TurboEngines

Table of Contents

Release Notes

Review the latest TurboEngines release notes that describe the system requirements and supported platforms, new features, enhancements, resolved issues, and known issues.

December21.12.23 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.12.23 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.4 | 2.225.4
Conga Configuration & Pricing **(New)**	13.2.1969.140 | 13.1969.140

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issue fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
00792520	CPQ-55571	The 'Validate Querymodel' action takes a long time to open the pop-up window.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8708	The progress bar appears on the cart page when you change the quantity through the client side.
LS-8577 / CPQ-54027	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8856 / CPQ-53987	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8249 / CPQ-54022	The attribute-based pricing does not work for the numeric expression.
LS-8477 / CPQ-53772	The product attribute value field does not clear after clicking the **Validate** button.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-56097	You encounter Apex CPU time limit exceeded errors while performing the following actions in the large cart with 5k lines. • Copying or deleting products. • Relaunching the finalized cart.

DOC ID: CTEDEC21PRN20221223

December21.06.24 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.06.24 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.4 | 2.225.4
Conga Configuration & Pricing **(New)**	13.2.1969.99 | 13.1969.99

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issue fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
00790886	CPQ-55236	In the TurboPricing flow, CPQ does not display the warranty for some instruments on the cart page.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8577 / CPQ-54027	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8856 / CPQ-53987	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8249 / CPQ-54022	The attribute-based pricing does not work for the numeric expression.
LS-8477 / CPQ-53772	The product attribute value field does not clear after clicking the **Validate** button.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.

Conga Internal ID	Description
CPQ-56097	You encounter Apex CPU time limit exceeded errors while performing the following actions in the large cart with 5k lines. • Copying or deleting products. • Relaunching the finalized cart.

DOC ID: CTEDEC21PRN20220624

December21.06.03 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.06.03 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.4 | 2.225.4

Product	Latest Certified Version *(Version Name | Version Number)*
Conga Configuration & Pricing **(New)**	13.2.1969.93 | 13.1969.93

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issue fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
00797910	CPQ-56046	CPQ intermittently displays blank cart lines while working on turbo quotes.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expressions in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. "Could not resolve remote action: saveAdjustmentLineItems".
LS-8249	The attribute-based pricing does not work for the numeric expression.

Conga Internal ID	Description
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.

Conga Internal ID	Description
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.

Conga Internal ID	Description
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	CPQ does not display available promotions under the Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized back to SFDC with auto-sync.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.
CPQ-54169	A constraint rule is taking time to trigger a specific quote.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing errors while passing masterdata as consumer name.
CPQ-54380	When you reconfigure the cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54535	In the case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies a promotion, the net price is incorrect.
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.

Conga Internal ID	Description
CPQ-54875	After you complete the price override, CPQ does not set Grand total and adjustments correctly.
CPQ-54898	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.
CPQ-56097	You encounter Apex CPU time limit exceeded errors while performing the following actions in the large cart with 5k lines: • Copying or deleting products. • Relaunching the finalized cart.

DOC ID: CTEDEC21PRN20220603

December21.04.29 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.04.29 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library **(New)**	2.2.225.3 | 2.225.3
Conga Configuration & Pricing **(New)**	113.2.1969.72 | 13.1969.72

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release.

Case Number	Conga Internal ID	Description
PST-2294	CPQ-55531	The constraint rule still appears on the cart page even after selecting the relevant option.
00791886	CPQ-55266	CPQ displays the incorrect Turbo Net adjustment % on Summary lines for miscellaneous line items.
00793694	CPQ-55570	When you add new options to the existing product model, the sequence in the cart does not match the sequence in the Configurator structure.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.

Conga Internal ID	Description
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. "Could not resolve remote action: saveAdjustmentLineItems".
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.

Conga Internal ID	Description
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,

Conga Internal ID	Description
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	CPQ does not display available promotions under the Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized backed to SFDC with auto sync.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.
CPQ-54169	A constraint rule is taking time to trigger a specific quote.

Conga Internal ID	Description
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing errors while passing masterdata as consumer name.
CPQ-54380	When you reconfigure cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54535	In case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54875	After you complete price override, CPQ does not set Grand total and adjustments are correctly.
CPQ-54898	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.

Conga Internal ID	Description
CPQ-54160	Auto-inclusion does not work when a sub-bundle and its option are auto-included using a constraint rule.

DOC ID: CTEDEC21PRN20220429

December21.04.15 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.04.15 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name | Version Number)*
Conga Base Library	2.2.225.2 | 2.225.2
Conga Configuration & Pricing **(New)**	113.2.1969.63 | 13.1969.63

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release.

Case Number	Conga Internal ID	Description
NA	CPQ-55413	You encounter a number of issues, including pending configuration, deselected options being selected again, and exclusion rules not being triggered when reconfiguring the Cart.
NA	CPQ-55168	After you select default options, default constraint rules are not triggered on the Service Configuration page.
NA	CPQ-54162	CPQ does not execute all required constraint rules in Turbo quote.

The following table lists the known issues fixed from the previous release.

Conga Internal ID	Description
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as New instead of Saved or Finalised.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled.
CPQ-54506	CPQ displays the following error on the Cart page even after you revalidate product with deleted primary price list item: *Entity is deleted*
CPQ-54690	In Turbo Engines Admin (app), when you click **Manage Sync** against any action, you get the following error: *Enter a valid URL and try Again*
CPQ-54826	After making the whole bundle as Is optional, CPQ still displays the config pending icon on the Cart.
CPQ-54970	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54045	When you have enabled the **Select Many Options** functionality, duplicate radio buttons are selected by default.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.

Conga Internal ID	Description
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. "Could not resolve remote action: saveAdjustmentLineItems".
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.

Conga Internal ID	Description
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.

Conga Internal ID	Description
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	CPQ does not display available promotions under the Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.

Conga Internal ID	Description
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized backed to SFDC with auto sync.
CPQ-54160	Auto-inclusion does not work when a sub-bundle and its option are auto-included using a constraint rule.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.
CPQ-54169	A constraint rule is taking time to trigger a specific quote.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing errors while passing masterdata as consumer name.
CPQ-54380	When you reconfigure cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54535	In case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54875	After you complete price override, CPQ does not set Grand total and adjustments are correctly.

Conga Internal ID	Description
CPQ-54898	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.

DOC ID: CTEDEC21PRN20220418

December21.04.01 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.04.01 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> ⓘ This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library **(New)**	2.2.225.2 | 2.225.2
Conga Configuration & Pricing **(New)**	113.2.1969.62 | 13.1969.62

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release.

Case Number	Conga Internal ID	Description
NA	CPQ-55047	When you relate an asset in the Service CPQ flow, CPQ intermittently displays the error "*Required Fields are Missing*".
00791361	CPQ-55170	CPQ is updating the values incorrectly in summary groups after repricing the cart in turbo flow.
PST-2330	CPQ-55095	Auto-inclusion is not triggering as intended when you switch the option on the cart page.

Case Number	Conga Internal ID	Description
PST-2321	CPQ-54976	When you reopen the cloned quote, you observe a deselected auto-inclusion option from the original quote.
NA	CPQ-54824	The constraint rule message does not appear on the cart page after deselecting IsOptional for optional products if the rule includes Min/Max criteria.

The following table lists the known issues fixed from the previous release.

Conga Internal ID	Description
CPQ-55023	CPQ does not display messages for copied products on the cart page and for constraint rules on the Configuration page.
CPQ-55017	CPQ intermittently displays a blank page after you perform BCP or CSP.
CPQ-54722	In Service CPQ, when you configure the same service product for the second time, an Exclusion constraint rule is not working.
CPQ-55045	The price of the same product mismatch in Classic flow and Turbo Flow.
CPQ-55031	When you return to the cart page from a custom Validate Cart page, CPQ does not load the cart.
CPQ-55030	CPQ does not update **Grand Total** when you reprice for the first time after applying adjustment or promotion.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as *New* instead of *Saved* or *Finalised.*
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.

Conga Internal ID	Description
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. "Could not resolve remote action: saveAdjustmentLineItems".
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.

Conga Internal ID	Description
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.

Conga Internal ID	Description
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54045	When you have enabled the Select Many Options functionality, duplicate radio buttons are selected by default.
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	CPQ does not display available promotions under the Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized backed to SFDC with auto sync.
CPQ-54160	Auto-inclusion does not work when a sub-bundle and its option are auto-included using a constraint rule.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.
CPQ-54169	A constraint rule is taking time to trigger a specific quote.

Conga Internal ID	Description
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing errors while passing masterdata as consumer name.
CPQ-54380	When you reconfigure cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54506	CPQ displays "*Entity is deleted*" error on the Cart page even after you revalidate product with deleted primary price list item.
CPQ-54535	In case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54690	In Turbo Engines Admin (app), when you click Manage Sync against any action, you get the following error: Enter a valid URL and try Again
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54826	After making the whole bundle as Is optional, CPQ still displays the config pending icon on the Cart.
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54875	After you complete price override, CPQ does not set Grand total and adjustments are correctly.
CPQ-54898 / CPQ-54970	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.

Conga Internal ID	Description
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-54976	When you clone a quote, CPQ automatically adds the auto-included option that you removed in the original quote.
CPQ-55005	When you add new products to the cart that auto-include other products, CPQ removes existing auto-included products from the cart.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.
CPQ-55016	In the TurboConfig flow, when you change the condition product that auto-included those options, CPQ does not deselect the auto-included options.

DOC ID: CTEDEC21PRN20220401

December21.03.23 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.03.23 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.60 | 13.1969.60

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the known issue fixed from the previous release.

Case Number	Conga Internal ID	Description
NA	CPQ-55047	When you relate an asset in the Service CPQ flow, CPQ intermittently displays the error "*Required Fields are Missing*".

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as *New* instead of *Saved* or *Finalised.*
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.

Conga Internal ID	Description
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.

Conga Internal ID	Description
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.

Conga Internal ID	Description
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54045	When you have enabled the Select Many Options functionality, duplicate radio buttons are selected by default.
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	CPQ does not display available promotions under the Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized backed to SFDC with auto sync.

Conga Internal ID	Description
CPQ-54160	Auto-inclusion does not work when a sub-bundle and its option are auto-included using a constraint rule.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.
CPQ-54169	A constraint rule is taking time to trigger a specific quote.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing errors while passing masterdata as consumer name.
CPQ-54380	When you reconfigure cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54506	CPQ displays "*Entity is deleted*" error on the Cart page even after you revalidate product with deleted primary price list item.
CPQ-54535	In case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54690	In Turbo Engines Admin (app), when you click Manage Sync against any action, you get the following error: Enter a valid URL and try Again
CPQ-54722	In Service CPQ, when you configure the same service product for the second time, an Exclusion constraint rule is not working.
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54826	After making the whole bundle as Is optional, CPQ still displays the config pending icon on the Cart.

Conga Internal ID	Description
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54875	After you complete price override, CPQ does not set Grand total and adjustments are correctly.
CPQ-54898 / CPQ-54970	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-54976	When you clone a quote, CPQ automatically adds the auto-included option that you removed in the original quote.
CPQ-55005	When you add new products to the cart that auto-include other products, CPQ removes existing auto-included products from the cart.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.
CPQ-55016	In the TurboConfig flow, when you change the condition product that auto-included those options, CPQ does not deselect the auto-included options.
CPQ-55017	CPQ intermittently displays blank page after you perform BCP or CSP.
CPQ-55023	CPQ does not display messages for copied product on Cart page and for constraint rule on Configuration page.
CPQ-55030	CPQ does not update **Grand Total** when you reprice for the first time after applying adjustment or promotion.
CPQ-55031	When you return to Cart page from a custom Validate Cart page, CPQ does not load the cart.

Conga Internal ID	Description
CPQ-55045	The price of the same product mismatch in Classic flow and Turbo Flow.

DOC ID: CTEDEC21PRN20220323

December21.03.15 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.03.15 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.59 | 13.1969.59

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

The following feature is new to the TurboEngines in this release.

Stop Pricing the Saved Turbo Cart on a Relaunch

In this release, The CPQ does not delete the turbo cart when you relaunch the cart. Instead, it invokes a new restore API passing all the new line items. Hence, the UI fetches these line items from salesforce in the lighter SO form. The turbo restore API finds the updated and deleted lines and prices the cart as needed and loads the cart faster. To see the updated pricing calculation when redirecting back or relaunching saved configuration, you must set the pricing status to Pending for the line items you are modifying externally with dependent line items from backend or custom code.

Enhancements

There are no new enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
NA	CPQ-54920	In the TurboConfig and TurboPricing flow, the custom field on the line item does not work in the constraint rule condition.
00786974	CPQ-54568	If you apply the usage tier to the line item, the approval status for the line items that were added or deleted to the cart does not reflect at the cart level.

Case Number	Conga Internal ID	Description
PST-2236	CPQ-54888	The exclusion rules are not working as intended while reconfiguring the cart.
NA	CPQ-54847	While configuring the copied product, CPQ does not display the configuration pending message and price pending icon for the original product, and the constraint rule message is not displayed on the configuration page.
NA	CPQ-54827	CPQ does not display the constraint rule message If you select auto inclusion or Show Messgage option first and enable IsOptional for them.
NA	CPQ-54733	In the service CPQ flow, CPQ display all the service products that are available for selection while relating the products.
NA	CPQ-54464	CPQ does not display the constraint rule message after removing the product from cart.
NA	CPQ-54065	You encounter the following error while configuring the products. `"Unable to obtain exclusive access to this record of 1 record".`

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as *New* instead of *Saved* or *Finalised.*

Conga Internal ID	Description
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8695	The product configuration does not get updated when you copy the product.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8685	Duplicate charge types are created for products with multiple charge types when you relaunch the saved cart.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8666	You observe a delay in removing the constraint rule error messages after selecting the required product or option.
LS-8588	The cart page does not update the grand total when you relaunch the saved turbo quote with more than 5k lines after deleting some products during the product configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.

Conga Internal ID	Description
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.

Conga Internal ID	Description
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.

Conga Internal ID	Description
CPQ-53494	In case of Include(AND) and Include with Bundle Option(AND), CPQ does not apply promotions as expected.
CPQ-53938	Multiple conditions with attributes are not fulfilling the criteria and getting incorrect base price and net price values on the Cart.
CPQ-54002	A constraint rule where Action Type = Inclusion and Action Intent = Show Message is not triggered, and a message is not disappearing when the user is trying to deselect options inside a sub-bundle.
CPQ-54028	CPQ does not display conditional charge types with multiple charge type criteria on the line item
CPQ-54045	When you have enabled the Select Many Options functionality, duplicate radio buttons are selected by default.
CPQ-54057	In a constraint rule where Action Type = Exclusion and Action Intent = Prompt, when the user adds a specific product to the Cart, CPQ does not display a prompt to remove that product.
CPQ-54066	CPQ does not trigger constraint rules again if you remove the condition product and add it again to the cart.
CPQ-54084	it is not available to remove under coupons tab of apply promotion dialog CPQ does not display available promotions under Coupons tab on the Apply Promotions pop-up. This rendered you to unable to remove the applied coupons.
CPQ-54101	When you remove promotions from the line items on the Cart page, CPQ sometimes does not complete the removal and keeps the promotions attached.
CPQ-54117	Sometimes pricing is not synchronized backed to SFDC with auto sync.
CPQ-54160	Auto-inclusion does not work when a sub-bundle and its option are auto-included using a constraint rule.
CPQ-54164	After data sync, auto-publish does not trigger automatically for a few products.

Conga Internal ID	Description
CPQ-54169	A constraint rule is taking time to trigger on a specific quote.
CPQ-54170	CPQ is taking almost 30 seconds to open the Configuration page.
CPQ-54288	Multiple APIs are throwing error while passing masterdata as consumer name.
CPQ-54380	When you reconfigure cart, CPQ does not display any products on the Cart page. This happens when the price list item of an auto-included product is inactive.
CPQ-54506	CPQ displays "*Entity is deleted*" error on the Cart page even after you revalidate product with deleted primary price list item.
CPQ-54535	In case of For Every X Get X promotion, when you add the same product twice to the Cart, you see weird pricing on the newly added line item.
CPQ-54690	In Turbo Engines Admin (app), when you click Manage Sync against any action, you get the following error: Enter a valid URL and try Again
CPQ-54722	In Service CPQ, when you configure the same service product for the second time, an Exclusion constraint rule is not working.
CPQ-54732	On the Deal Guidance pop-up, the Price From and Price To values are incorrect.
CPQ-54787	When Rounding Mode is set to HALF_DOWN and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.
CPQ-54826	After making the whole bundle as Is optional, CPQ still displays the config pending icon on the Cart.
CPQ-54838	When Rounding Mode is set to HALF_UP and Custom Rounding flag is set to True, if the user applies Cumulative Range Matrix with % Discount, the net price is incorrect.
CPQ-54840	When Rounding Mode is set to UP and Custom Rounding flag is set to True, if the user applies promotion, the net price is incorrect.

Conga Internal ID	Description
CPQ-54875	After you complete price override, CPQ does not set Grand total and adjustments are correctly.
CPQ-54898 / CPQ-54970	In the TurboConfig flow, CPQ delays or does not execute the constraint rules when you use Service CPQ.
CPQ-54902	CPQ updates **Selling Term** for options where auto-cascade for **Selling Term** is set to false.
CPQ-54976	When you clone a quote, CPQ automatically adds the auto-included option that you removed in the original quote.
CPQ-55005	When you add new products to the cart that auto-include other products, CPQ removes existing auto-included products from the cart.
CPQ-55013	When you reconfigure a cart after finalizing the proposal, CPQ does not display the pop-up on the click of the "i" icon next to the bundle product to open the price details.
CPQ-55016	In the TurboConfig flow, when you change the condition product that auto-included those options, CPQ does not deselect the auto-included options.
CPQ-55017	CPQ intermittently displays blank page after you perform BCP or CSP.
CPQ-55023	CPQ does not display messages for copied product on Cart page and for constraint rule on Configuration page.
CPQ-55030	CPQ does not update **Grand Total** when you reprice for the first time after applying adjustment or promotion.
CPQ-55031	When you return to Cart page from a custom Validate Cart page, CPQ does not load the cart.
CPQ-55045	The price of the same product mismatch in Classic flow and Turbo Flow.
CPQ-55047	When you relate an asset in the Service CPQ flow, CPQ intermittently displays the error "*Required Fields are Missing*".

DOC ID: CTEDEC21PRN20220315

December21.03.04 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.03.04 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	**Latest Certified Version** ***(Version Name / Version Number)***
Conga Base Library	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.53 | 13.1969.53

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

The following section describes existing features that are changed (or are no longer supported) in this release.

Enhancements to Optional Products Inclusion Rules and Auto-Inclusion Rule Actions

- The products that are marked optional using the **IsOptional** checkbox are not considered while evaluating inclusion rules. In addition, CPQ displays an error or warning message when the auto-inclusion rule of type error or warning is set to optional. If the parent bundle of the products is optional, any product that is optional by the rule will not display an error or warning message.
- The products that are marked optional using the **IsOptional** checkbox are not considered while evaluating Auto-Inclusion Rule Actions. In addition, CPQ displays a warning message if you include products from a show message inclusion rule or check on a finalize inclusion rule of type warning is set to optional. If the parent bundle of the products is optional, any product that is optional by the rule will not display an error or warning message.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
00785953	CPQ-54567	CPQ allows you to make adjustments even though Allow Manual Adjustment option is disabled at the PLI level.

Case Number	Conga Internal ID	Description
NA	CPQ-53717	When you apply a summary group adjustment on grand total, you observe a difference in net price resulting in a trigger for approval check.
00787112	CPQ-54582	If you deactivate some products in the saved cart, while revalidating the cart, CPQ displays the following error. `"Some items in the cart may have changed. Please revalidate the cart".`
PST-2230	CPQ-54551	The inclusion and exclusion rules parity do not match after clicking relate button while the configuring service
PST-2277	CPQ-54550	The contract numbers revert back to the original value post pricing.
PST-2283	CPQ-54549	The constraint rules are not working as intended.
PST-2269	CPQ-54548	The constraint rule does not work as intended while configuring the product.
00783897	CPQ-54173	You intermittently encounter the following error while banding the cart. `"UNABLE_TO_LOCK_ROW"`
NA	CPQ-54168	The promotion codes are still visible on the cart page and get recalculated even after removing the promotion code. **Resolution**: Perform the Quick Save or Reprice actions.
PST-2249	CPQ-54162	Pending Configuration or Deselected option gets selected again and exclusion rules are not triggered while reconfiguring the cart.
NA	CPQ-54643	The incentive code is not consistently applied on non-eligible line items in the turbo flow.
NA	CPQ-54753	The proposal gets finalized after selecting the Is Optional checkbox for an optional product.
NA	CPQ-54740	You observe incomplete constraint rule messages while reconfiguring products.

Case Number	Conga Internal ID	Description
NA	CPQ-54752	CPQ displays the constraint rule error or warning message after selecting the IsOptional Checkbox for the ParentBundle (Sub bundle).
NA	CPQ-54685	CPQ does not display the constraint rule message after Selecting Auto Included Option product IsOptional Checkbox.
NA	CPQ-54081	In Service CPQ flow, when you want to associate the same bundle asset and bundle service, CPQ does not work properly in this use case.
NA	CPQ-54062	The inclusion rule for service option product does not trigger if you enable the Service asset and Releated lines flag.
NA	CPQ-54033	The inclusion rule for service option products does not trigger as intended.
NA	CPQ-54463	CPQ does not display constraint rule message on Config page when there is an inclusion show message rule with options in the Parent and sub bundle.

The following table lists the known issue fixed from the previous release.

Conga Internal ID	Description
CPQ-54622	When you reconfigure the approved cart, the status of the cart is set to the *Approval Required* instead of *Approved.*

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as *New* instead of *Saved* or *Finalised.*
LS-8789	When the default quantity of options has a numeric expression, the cart page does not calculate the option price for the bundle.
LS-8728	The cart calculates the grand total incorrectly and gets corrected after reconfiguring the product.
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8695	The product configuration does not get updated when you copy the product.
LS-8798	Multiple Adjustment does not copy to cloned Quote.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8685	Duplicate charge types are created for products with multiple charge types when you relaunch the saved cart.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8666	You observe a delay in removing the constraint rule error messages after selecting the required product or option.

Conga Internal ID	Description
LS-8588	The cart page does not update the grand total when you relaunch the saved turbo quote with more than 5k lines after deleting some products during the product configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8806	The cart page does not display the constraint rule errors or warnings intermittently after reconfiguring the quote saved with constraint rule error or warning messages.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8755	The empty cart is displayed when you navigate to the cart page from the Validate Cart page.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8857	After copying the line items, the cart page displays the incorrect error message on the progress bar.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.

Conga Internal ID	Description
LS-8838	You encounter the following error while finalizing the revalidated cart if the revalidation callback is configured in your org. `"entity is deleted".`
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.

Conga Internal ID	Description
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,

Conga Internal ID	Description
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
CPQ-52177	In the TurboPricing flow, the progress bar appears a second time even after sync action has started.
CPQ-52179	On the Cart page, CPQ displays are vertical scroll on the progress bar.
CPQ-52180	In the TurboPricing flow, the sync icon tooltip message is displayed in multiple lines masking some fields in the cart.
CPQ-52186	In the TurboPricing flow, when you performed save action and update a line item, CPQ calculates pricing though you have not clicked Reprice.
CPQ-52224	In TurboConfig and TurboPricing flows, CPQ displays the pricing pending icon for a few seconds after you perform mass updates.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21PRN20220304

December21.02.18 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.02.18 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.46 | 13.1969.46

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
PST-2223	CPQ-54171	You encounter the following error on the cart page if you perform the copy action. `"Error executing [saveCartTotals]: Delete failed. First exception on row 0; first error: ENTITY_IS_DELETED, entity is deleted".`
00781965	CPQ-54166	When usage price Tiers are applied for a product in the cart, the **Reprice** button becomes unresponsive.
PST-2260	CPQ-54159	The cart page becomes unresponsive while configuring the products.
PST-2145	CPQ-54491 & LS-9034	If you reprice the copied bundle, the status of the completely configured bundle changes to *Pending for Config*.
NA	CPQ-54163	The cart page creates duplicate sequences on relauching.
NA	CPQ-54465	Auto Inclusion and exclusion rules do not work for the menu bundles in the turbo flow.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`

Conga Internal ID	Description
CPQ-54620	After saving or finalizing the product configuration, the status of the product configuration is displayed as *New* instead of *Saved* or *Finalised.*
CPQ-54622	When you reconfigure the approved cart, the status of the cart is set to the *Approval Required* instead of *Approved.*
LS-8789	When the default quantity of options has a numeric expression, the cart page does not calculate the option price for the bundle.
LS-8728	The cart calculates the grand total incorrectly and gets corrected after reconfiguring the product.
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8695	The product configuration does not get updated when you copy the product.
LS-8798	Multiple Adjustment does not copy to cloned Quote.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8685	Duplicate charge types are created for products with multiple charge types when you relaunch the saved cart.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8666	You observe a delay in removing the constraint rule error messages after selecting the required product or option.

Conga Internal ID	Description
LS-8588	The cart page does not update the grand total when you relaunch the saved turbo quote with more than 5k lines after deleting some products during the product configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8806	The cart page does not display the constraint rule errors or warnings intermittently after reconfiguring the quote saved with constraint rule error or warning messages.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8755	The empty cart is displayed when you navigate to the cart page from the Validate Cart page.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8857	After copying the line items, the cart page displays the incorrect error message on the progress bar.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.

Conga Internal ID	Description
LS-8838	You encounter the following error while finalizing the revalidated cart if the revalidation callback is configured in your org. `"entity is deleted".`
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.

Conga Internal ID	Description
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.

Conga Internal ID	Description
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
CPQ-52177	In the TurboPricing flow, the progress bar appears a second time even after sync action has started.
CPQ-52179	On the Cart page, CPQ displays are vertical scroll on the progress bar.
CPQ-52180	In the TurboPricing flow, the sync icon tooltip message is displayed in multiple lines masking some fields in the cart.
CPQ-52186	In the TurboPricing flow, when you performed save action and update a line item, CPQ calculates pricing though you have not clicked Reprice.
CPQ-52224	In TurboConfig and TurboPricing flows, CPQ displays the pricing pending icon for a few seconds after you perform mass updates.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21PRN20220218

December21.02.04 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.02.04 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

All Conga customers have FREE access to getting started content and release training in the Conga Learning Center. To take your training further, ensure your organization has access to the Conga Learning Pass, which is a training subscription service. Click here to learn more.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.40 | 13.1969.40

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

There are no new features in this release. Keep checking the Conga Documentation Portal for new updates.

Enhancements

There are no enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
PST-2244	LS-9007	You encounter the following error while adding the service a second time to the cart. `updateCartLineItems: Attempt to de-reference a NULL object.`
NA	LS-8964	If the custom revalidation callback is enabled, you encounter duplicate remote calls (getCustomRevalidation) while configuring the services or products.
PST-2216	LS-8944	The user-selected options get removed with nested bundles of three-level after calculating service price.
NA	LS-9003	You encounter the following error while adding the service product to the cart. `'System.NullPointerException: Attempt to de-reference a null object'`

Case Number	Conga Internal ID	Description
PST-2225	LS-9020	You encounter the following error while adding the service product to the cart. `'System.NullPointerException: Attempt to de-reference a null object'`
PST-2086	LS-8631	If you remove the bundle and try to reconfigure the same bundle, you intermittently encounter the following error. `'Error executing [updateCartLineItems]: Attempt to de-reference a null object'`
NA	LS-8951	You encounter the InvalidActorNameException as Child actor is not unique if the snapshot fails.
00781965	LS-8952	The **Reprice** button is not responsive after applying the usage price Tiers to a product in the cart.
00783889	LS-8975	The custom formula field (%) is calculated incorrectly for a warranty charge type. The cart displays the non-zero value for the custom formula field margin instead of 0 when you add charge type *Lease* using a product attribute.
PST-2233	LS-8988	The grand total on the cart is not updating after CSP for NONPOS flow after changing quantity, billing plan, and start or End dates.
00783889	LS-8980	The Enddate of the option gets removed and replaced by the proposed end date If you delete and recreate the cart.
00783500	LS-7901	The cart approvals are not working on the product configuration object if the *Order* object is a parent object.
NA	LS-8992	The constraint rule is not working for the non-POS quotes which contain products that are not part of the current product structure.
NA	LS-8950	You encounter ReverseSync ActorNotFound expectation error while resolving the ActorSelection.

Case Number	Conga Internal ID	Description
00783500	LS-8940	If you apply a base price override adjustment to the bundle main charges type line item during the bundle configuration, the same adjustment gets applied to the other chare type line items.
PST-2210	LS-8930	Changing the default selections in a service configuration removes most of the auto selection from the cart page after executing the pricing.
NA	LS-8896	You encounter a browser's WebSocket API error if you try to send a message when the WebSockets handle is in the connecting state.
PST-2203	LS-8895	The constraint rule still appears on the cart page even after selecting the relevant option for menu bundles.
NA	LS-8860	If you apply the discounts through the summary group, the Approval check does not work as intended.
NA	LS-8820	The deal guidance displays the incorrect results due to the incorrect formula field result.
PST-2158	LS-8721	You observe multiple discrepancies in the cart page after calculating the service price.
PST-2067	LS-8530	You cannot trigger the eligibility rules for a few products.
00775003	LS-8875	You observe the following error while configuring the product. `"Price List Item not Found".`

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
LS-8789	When the default quantity of options has a numeric expression, the cart page does not calculate the option price for the bundle.
LS-8728	The cart calculates the grand total incorrectly and gets corrected after reconfiguring the product.
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.
LS-8695	The product configuration does not get updated when you copy the product.
LS-8798	Multiple Adjustment does not copy to cloned Quote.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8685	Duplicate charge types are created for products with multiple charge types when you relaunch the saved cart.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8666	You observe a delay in removing the constraint rule error messages after selecting the required product or option.
LS-8588	The cart page does not update the grand total when you relaunch the saved turbo quote with more than 5k lines after deleting some products during the product configuration.

Conga Internal ID	Description
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.
LS-8806	The cart page does not display the constraint rule errors or warnings intermittently after reconfiguring the quote saved with constraint rule error or warning messages.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8755	The empty cart is displayed when you navigate to the cart page from the Validate Cart page.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8857	After copying the line items, the cart page displays the incorrect error message on the progress bar.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8838	You encounter the following error while finalizing the revalidated cart if the revalidation callback is configured in your org. `"entity is deleted".`

Conga Internal ID	Description
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.

Conga Internal ID	Description
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.

Conga Internal ID	Description
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
CPQ-52177	In the TurboPricing flow, the progress bar appears a second time even after sync action has started.
CPQ-52179	On the Cart page, CPQ displays are vertical scroll on the progress bar.
CPQ-52180	In the TurboPricing flow, the sync icon tooltip message is displayed in multiple lines masking some fields in the cart.
CPQ-52186	In the TurboPricing flow, when you performed save action and update a line item, CPQ calculates pricing though you have not clicked Reprice.
CPQ-52224	In TurboConfig and TurboPricing flows, CPQ displays the pricing pending icon for a few seconds after you perform mass updates.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21PRN20220204

December21.01.07 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.01.07 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

All Conga customers have FREE access to getting started content and release training in the Conga Learning Center. To take your training further, ensure your organization has access to the Conga Learning Pass, which is a training subscription service. Click here to learn more.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library **(New)**	2.2.225.1 | 2.225.1
Conga Configuration & Pricing **(New)**	113.2.1969.17 | 13.1969.17

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

The following is a new feature in this release.

Stop recreating turbo cart on redirects or reopens

In this release, The CPQ does not delete the turbo cart when you relaunch or reopen the cart. Instead, it invokes a new restore API passing all the new line items. Hence, the UI fetches these line items from salesforce in the lighter SO form. The turbo restore API finds the updated and deleted lines and prices the cart as needed and loads the cart faster. To see the updated pricing calculation when redirecting back or relaunching saved configuration, you must set the pricing status to *Pending* for the line items you are modifying externally with dependent line items from backend or custom code.

Enhancements

There are no enhancements in this release.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
NA	LS-8701	The cart page does not update summary group formula fields when the turbo formula filed flag is set to *false.*
NA	LS-8807	Turbo cart does not re-validate approval checks after relaunching.
00781372	LS-8744	The promotions applied on main charges automatically apply for other chart types.
PST-2148	LS-8705	The Inclusion rule is not triggering while configuring the services in the turbo flow.
NA	LS-8689	The cloned cart does not load summary groups when you relaunch it.
PST-2136	LS-8659	The constraint rules are not triggering as intended while configuring the products.

Case Number	Conga Internal ID	Description
PST-2121	LS-8632	The cart page removes the option added using the copy function in the turbo flow.
NA	LS-8631	When you remove the bundle and reconfigure the same bundle again, you intermittently encounter the following error. `"Error executing [updateCartLineItems]: Attempt to de-reference a null object"`
NA	LS-8627	Promotion does not apply to the rollup charge line when performing pricing for the first time.
PST-2079	LS-8561	You observe a configuration pending icon after copying 260000 menu bundles in the turbo flow.
PST-2078	LS-8531	The publish option does not work for bundles when you enable the UseDataSync option.
NA	LS-8521	The revalidation callback is not working resulting in a JavaScript error in the browser console.
NA	LS-8511	The cart does not display the products when you re-open the saved cart.
00775003	LS-8245	You encounter the following error while configuring the product. `"Price List Item not found"`

The following table lists the known issues fixed from the previous release.

Conga Internal ID	Description
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8455	Even after completing the data sync, the cart page displays the following error after adding the product: `"insufficient access rights on cross-reference id".`

Conga Internal ID	Description
LS-8429	When you delete, expire, or inactivate the primary PLI of the option product and revalidate, the cart does not update the total value.
LS-8254	After saving or finalizing the cart, the usage price tier data is not stored on the LineItem object.
LS-8223	After selecting an option during configuration, when you click the Add More Products, the following error shows on the catalog page. • `"Id not specified in an update call"`
LS-8491	The COF rule does not trigger when you use the product fieldset in condition criteria.
LS-8428	The price list rules are not working as intended when the action criteria have conditions based on the price list.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8818	You encounter the following error after deleting the bundle or subbundle or option with PAR. `"Getting Error executing [deleteLineItems]: Attempt to de-reference a null object"`
LS-8789	When the default quantity of options has a numeric expression, the cart page does not calculate the option price for the bundle.
LS-8728	The cart calculates the grand total incorrectly and gets corrected after reconfiguring the product.
LS-8708	When you change the quantity through the client-side, the progress bar appears on the cart page.

Conga Internal ID	Description
LS-8695	The product configuration does not get updated when you copy the product.
LS-8798	Multiple Adjustment does not copy to cloned Quote.
LS-8692	You can not select the options that have a long product description during the product configuration.
LS-8685	Duplicate charge types are created for products with multiple charge types when you relaunch the saved cart.
LS-8674	The cart becomes unresponsive and does not allow copying products during the configuration.
LS-8672	Duplicate radio buttons are selected by default during the product configuration if the **Select Many Options** functionality is enabled. **Resolution:** Click the **Confirm Option Selections** to remove the duplicate radio buttons selection.
LS-8666	You observe a delay in removing the constraint rule error messages after selecting the required product or option.
LS-8588	The cart page does not update the grand total when you relaunch the saved turbo quote with more than 5k lines after deleting some products during the product configuration.
LS-8864	The cart displays the grand total when you revalidate the expired or inactive price list items with custom fields.
LS-8577	When you relaunch the saved cart, the cart displays the incorrect progress bar message.
LS-8600	The pending configuration icon is displayed next to the products when you relaunch the saved quote after updating attributes with numeric expression in the product configuration.
LS-8786	The cart becomes unresponsive when you relaunch the cart in a read-only mode after making the changes to line items.

Conga Internal ID	Description
LS-8806	The cart page does not display the constraint rule errors or warnings intermittently after reconfiguring the quote saved with constraint rule error or warning messages.
LS-8727	You observe inconsistent behavior of the progress bar while launching the cart.
LS-8755	The empty cart is displayed when you navigate to the cart page from the Validate Cart page.
LS-8856	When the *saveAdjustmentLineItems* action is in progress, you observe a top progress bar instead of the Save action button.
LS-8857	After copying the line items, the cart page displays the incorrect error message on the progress bar.
LS-8852	You encounter the following error message when you complete the pricing after adding the 1k lines to the cart. `"Could not resolve remote action: saveAdjustmentLineItems".`
LS-8249	The attribute-based pricing does not work for the numeric expression.
LS-8838	You encounter the following error while finalizing the revalidated cart if the revalidation callback is configured in your org. `"entity is deleted".`
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.

Conga Internal ID	Description
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.

Conga Internal ID	Description
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
CPQ-52177	In the TurboPricing flow, the progress bar appears a second time even after sync action has started.
CPQ-52179	On the Cart page, CPQ displays are vertical scroll on the progress bar.

Conga Internal ID	Description
CPQ-52180	In the TurboPricing flow, the sync icon tooltip message is displayed in multiple lines masking some fields in the cart.
CPQ-52186	In the TurboPricing flow, when you performed save action and update a line item, CPQ calculates pricing though you have not clicked Reprice.
CPQ-52224	In TurboConfig and TurboPricing flows, CPQ displays the pricing pending icon for a few seconds after you perform mass updates.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21PRN20220107

December21.12.17 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December21.12.17 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> ⓘ This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library	2.2.223 | 2.223
Conga Configuration & Pricing **(New)**	13.2.1969.3 | 13.1969.3

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

The following are new features in this release.

Support for Validation Constraint Rule on Service CPQ

TurboConfig supports the following client-side constraint rules. In addition, you can use this functionality to apply constraint rules services in Service Catalog.

Type	Constraint Rule
Validation	Match In Service Assets
	Match In Releated Lines

For more information, see Configuring Service CPQ and Creating Constraint Rule Actions.

Enhancements

The following section describes existing features that are changed (or are no longer supported) in this release.

Enhancement to the Turbo Assessment Application

The turbo assessment application is enhanced to include the details of TurboPricing asset-based ordering (ABO) features.

For more information, see Installing TurboPricing Assessment Package.

Enhanced Price Waterfall Display Setup

You can create multiple waterfalls for a price pipeline. When you select the price pipeline, the waterfall is displayed based on the criteria you defined during the waterfall setup. The User Criteria is added to the waterfall criteria pop-up to define the user criteria. Currently, you can use only 'user role' as a criteria field.

For more information, see Configuring Price Waterfall to a Price Pipeline.

Provision to View and Update Line Item Fields in the Price Waterfall Display UI

You can view and update the Line Item Fields in the Price Waterfall page as required without navigating to the cart page. However, you must add the required Line Item fields during the waterfall setup. Also if you want to give the user an ability to edit, mark one or more of the Line Item Fields as editable. This feature helps you validate the values of key Line Item Fields such as base price, net price, adjustments, option price, and extended price without navigating to the cart page.

For more information, see Working with Price Waterfall.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
NA	LS-8415	When the socket handle is in the CONNECTING state and you try to send a message, the WebSocket API of the browser throws an error. **Resolution Info:** Add an admin setting *APTS_PricingSocketKeepAliveInterval* with the Value as 45.
NA	LS-8493	When you open inflight quotes after catalog changes or data migration in Turbo mode, some field expressions are not working properly.
NA	LS-8508	In TurboPricing and TurboConfig mode, CPQ does not load the Config page when you configure multiple big bundles and the Pricing callback stops responding when you land on the Cart causing the loss of products added earlier.
NA	LS-8512	CPQ redirects you to the Catalog page from a custom page instead of redirecting you to the Cart page.
NA	LS-8562	In TurboPricing and TurboConfig mode, Auto Inclusion and Auto Exclusion rules are not working for Menu bundles for turbo quotes.
00779317	LS-8579	When you try to update and save Usage Price Tiers, the following error is displayed: *Error - Could not resolve remote action: saveUsagePriceTiers*
NA	LS-8586	CPQ displays a blank page after you click Return to Cart in both Turbo and Non-Turbo flows.
NA	LS-8630	An option that is disabled and selected in the Classic mode is disabled and unselected in the Turbo mode.

The following table lists the known issues fixed from the previous release.

Conga Internal ID	Description
LS-8484	Performing mass updates on the cart page causes delay in progress messages.
LS-8457	The AutoCascading Selling Term for MultiCharge Type is not working as intended.

Conga Internal ID	Description
LS-8518	The constraint rules do not work as intended on the configuration page.
LS-8445	An error icon appears beside the bundle product even though the **Isoptional** option is checked.
LS-8356	The exclusion or inclusion rule error message displays the incorrect condition product when selecting the asset option as a conditional product.
LS-8347	The revalidation message does not appear in the turbo flow while reconfiguring the quote.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8455	Even after completing the data sync, the cart page displays the following error after adding the product: `"insufficient access rights on cross-reference id".`
LS-8429	When you delete, expire, or inactivate the primary PLI of the option product and revalidate, the cart does not update the total value.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.

Conga Internal ID	Description
LS-8254	After saving or finalizing the cart, the usage price tier data is not stored on the LineItem object.
LS-8223	After selecting an option during configuration, when you click the Add More Products, the following error shows on the catalog page. • `"Id not specified in an update call"`
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8567	PFStoPF rule does not trigger when you re-configure the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8491	The COF rule does not trigger when you use the product fieldset in condition criteria.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8428	The price list rules are not working as intended when the action criteria have conditions based on the price list.

Conga Internal ID	Description
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,

Conga Internal ID	Description
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
CPQ-52177	In the TurboPricing flow, the progress bar appears a second time even after sync action has started.
CPQ-52179	On the Cart page, CPQ displays are vertical scroll on the progress bar.
CPQ-52180	In the TurboPricing flow, the sync icon tooltip message is displayed in multiple lines masking some fields in the cart.
CPQ-52186	In the TurboPricing flow, when you performed save action and update a line item, CPQ calculates pricing though you have not clicked Reprice.
CPQ-52224	In TurboConfig and TurboPricing flows, CPQ displays the pricing pending icon for a few seconds after you perform mass updates.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21RN20211217

December '21 Release Notes

In these release notes, you can find packages, requirements, features, enhancements, resolved issues, and known issues for the TurboEngines December '21 release. For documentation updates, see What's New in TurboEngines Documentation and What's New in Data Sync Documentation.

> This documentation may contain descriptions of software features that are optional and for which you may not have purchased a license. As a result, your specific software solution and/or implementation may differ from those described in this document. Please contact your CSM or AE for information on your specific features and licensing.

Packages

The following packages and dependent packages are required to upgrade to this release to utilize all the new features of this release. These are the *minimum* required versions; later versions are also supported. Separate prerequisites for each feature can be found in the respective guides. The packages marked as **(New)** are new in this release.

Product	Latest Certified Version *(Version Name / Version Number)*
Conga Base Library **(New)**	2.2.223 | 2.223
Conga Configuration & Pricing **(New)**	13.1969

System Requirements and Supported Platforms

For information pertaining to the requirements and recommendations, you must consider before you proceed with the installation of the Conga product suite delivered on the Salesforce platform, see System Requirements and Supported Platforms Matrix.

New Features

The following are new features in this release.

TurboConfig: Support for Service CPQ

Service CPQ comprises of Installed Products (Purchased Products - Assets), Service Catalog (Quoting), Service Configuration (Config), and Service Cart modules for the service management.

Service CPQ allows customers to tackle service flows for Conga's existing or newly purchased products. With Service CPQ, customers can define and associate services to products so their sales teams can easily find applicable services and add them to their quotes. Starting from Assets, sales teams will utilize an intuitive and familiar flow, going through a catalog of services, configuring those services, and pricing them in their cart. In this release, TurboConfig supports the following client-side Constraint rules. In addition, you can use this functionality to apply constraint rules services in Service Catalog.

Type	Constraint Rule
Inclusion	Match In Service Assets
	Match In Releated Lines
Exclusion	Match In Service Assets
	Match In Releated Lines

For more information, see Configuring Service CPQ and Creating Constraint Rule Actions.

TurboPricing: Ability to set the approvals and guidance based on the adjustments to Options in Bundle and Summary Group Net Adjustment %

Ability to roll up the net adjustment percentage at bundle level if any adjustments are applied at option level. For this, you must define value as *True* for *APTS_RollUpOptionNetAdjustment*. For more information, see Configuring Option Net Adjustment Rollup to Bundle and Pricing Products.

TurboPricing: Smart Search - Ability to add the weightage to the products for Smart Search

The weightage allocation enables the Smart Search to control the ranking of the search results based on customer needs. The total weightage is 100%, and you can split it into any number of Product Fields that participate in the search criteria.

For more information, see Smart Search Settings.

TurboPricing: Asset Based Ordering - Ability to change the Standalone and Bundle asset for the configurable fields

You can change the standalone assets for the configurable fields. For example, you can update the values for the following fields on the cart page using a single Change action.

- Attribute Value
- Quantity
- Adjustment Type
- Price
- Start Date
- End Date
- Billing Frequency
- Billing Rule
- Billing Preference
- Payment Term
- Selling Term

For more information, see Changing an Asset and Changing Configurable Bundle Assets.

TurboPricing: Asset-Based Ordering - Ability to terminate the asset

You can terminate the standalone, fixed bundle, and configurable bundle assets. The status of the canceled asset line items shows *Pending Cancellation* on the Installed Products page. For more information, see Terminating an Asset.

TurboPricing: Asset-Based Ordering - Ability to edit the cart fields during the termination of an asset

You can edit the custom fields on the termination page if the administrator configures **Editable Fields for Cancelled Lines** and **Display Columns Settings.** For more information, see Terminating an Asset.

> If the Conga Billing package is not installed in your org, it is recommended to display the Confirm Termination pop-up instead of the Confirm Termination page. In addition, you can add the **Termination Date** field in **Display Column Settings** for the **Display Type** *Asset Termination*. For more information, see Configuring Display Columns Settings.

TurboPricing: Asset-Based Ordering - Ability to renew an asset

You can renew the Standalone Asset with single and multiple charge types and Bundle Asset with options. For more information, see Renewing an Asset.

TurboPricing: Asset-Based Ordering - Ability to configure the Auto Renew, Renewal Adjustment Type, and Renewal Adjustment Amount on pricing

You can configure Auto-Renew, Renewal Adjustment Type, and Renewal Adjustment Amount on pricing at the Price List Item level, cart level, or asset level.

- **Auto Renew**: Enable this setting when you want CPQ to automatically close the renewal quote with an **Auto Renew = True asset**. On the expiry date of the asset, CPQ automatically processes the renewal quote and renews the asset for the next term. The quote status is set to *Accepted* without any intervention (Touchless Renewal). This setting is used to group assets into different renewal quotes, along with the **Renewal Group Fields**.
- **Auto Renewal Type**: CPQ allows you to configure an asset as renewable or non-renewable at the price list item level. The supported values are:
    - Fixed
    - Evergreen
    - Do Not Review
- Fixed - select this to make the asset renewable, and CPQ can also pull the product into an auto and On-demand renewal quote.
- Evergreen & Do Not Review- Select this to make the asset non-renewable, and CPQ can not pull the product into an auto or on-demand renewal quote.

If you configure the **Renewal Adjustment Type** and **Renewal Adjustment Amount** columns for the cart of the current flow, you will see those columns.

- In the **Renewal Adjustment Type** drop-down for a line item, select *% Uplift* by how much you want to uplift the asset price during renewal. In this case, enter the required percentage in the **Renewal Adjustment Amount** field of the line item.
- In the **Renewal Adjustment Type** drop-down for a line item, select *Uplift Amount* by how much you want to uplift the asset price during renewal. In this case, enter the required uplift amount in the **Renewal Adjustment Amount** field of the line item.

For more information, see Applying the Default Pricing to Products and Renewing Assets Manually.

TurboPricing: Asset-Based Ordering - Renewal Line Item association with Config Line Item ID

The system generates the auto-renewal quote when you set the following for a Price List Item.

- Enable the Auto-renewal Flag and select *Fixed* as the Auto-renewal Type. If none is selected, the system considers it as *Fixed* only.
- If required, update the Auto-renewal Term, or the system considers *Default Renewal Term* or *Selling Term* as per the precedence.

The Config Line Item ID of the Price List Item remains the same in the renewal quote. The Installed Products page displays the renewed quote when you cloned the renewed quote. You must configure the Installed product Settings as mentioned below.

- **Default Renewal Price List**: The Price list currently used for the Proposal
- **Renewal Business Object Type:** Proposal
- **Renewal Default Price Book**: Standard Price Book
- **Renewal Execution Mode**: Auto
- **Alert Asset Related To Renewal Cart:** Enabled

TurboPricing: Asset-Based Ordering - Selective display of cotermination options during renewal

In this release, you can define the list of options to display while renewing the asset through the **Cotermination Preference During Renewal** option available on the **Installed Products** settings page. For more information, see Configuring Installed Products Settings.

TurboPricing: Asset-Based Ordering - Ability to set the precedence of the terms during renewal

You can define the auto-renewal term that CPQ must consider while renewing assets. CPQ considers the **Default Renewal Term** specified in Installed Product Settings if you do not specify any value. If both Auto-Renewal Term and Default Renewal Term are null, CPQ considers the selling term on the original asset during renewal. You are allowed to configure this setting at the price list item level, which will cascade to the cart and asset level. For more information, see Applying the Default Pricing to Products.

TurboPricing: Asset-Based Ordering - Quote Renewals & Agreement Renewals: Auto-Closure process

When you perform the auto-renewal for any asset, the system generates the auto-renewable quote for which you have to set the Asset Start Date (n-3) and Asset End Date (n-2), 'n' is the current system date. The renewal agreement set for auto closure expires within 24 hours. A new **Auto Closure Renewal Proposal** link is added to the system-generated auto-renewed quote in this release. Clicking the **Auto Closure Renewal Proposal** link closes the Auto-renewal quote. In addition, it creates Asset Line Items and Order Line Items with the status *Activated* and **Is Auto Accepted** flag is set to *True.* You can view the status of Asset Line Items in the **Temp Renew Asset Group** section during pricing calculation. For more information, See Configuring Agreement-Based Renewals and Managing Assets in the Contract Flow.

TurboPricing: Asset-Based Ordering - Ability to remove the option from bundle during the change operation

You can remove options from the bundle during the change operation. For more information, see Changing Configurable Bundle Assets.

TurboPricing: Asset-Based Ordering - Support for Same Day Cancellation with Calendar Method during Asset termination

CPQ sets the start date as the end date when you terminate an asset before the end date. Therefore, you must provide the asset end date as the termination date. If the *Same Day Cancelation* flag is set to *True,* then the asset end date -1 is considered for pricing calculations. If the *Same Day Cancelation* flag is set to *False, the* asset end date is considered for pricing calculations. For more information, see Same Day Cancellation.

TurboPricing and TurboConfig: Option Group Min/Max Validation

- If an option is marked as optional using **Is Optional** checkbox, that option is not considered while validating Option Groups' Min/Max criteria on the Configuration page. Therefore, you must select other options that are not optional to satisfy the Min/Max criteria.
- The optional lines in an optional bundle are not considered in the Option Group Min/Max validation on the Configuration page. Therefore, you can add options regardless of the Min/Max criteria.
- The optional lines in an Optional Group where the min Quantity is greater than or equal to '0' (>=0) and max quantity are greater than equal to '1' (>=1) are not considered in the Option Group Min/Max validation on the Configuration page.

Enhancements

The following section describes existing features that are changed (or are no longer supported) in this release.

TurboPricing: Ability to Apply Buy X Get Y Promotions on Multiple Line Items of Y Products

You can apply promotions on multiple line items of benefit products (Y) in Buy X Get Y scenarios. For more information, see Applying Promotions on Line Items in the cart.

Data Sync: TurboPricing instance provisioning to the single currency orgs

Exchange rates are required for currency conversion during pricing and, therefore, must be synced by TurboEngines. Since single currency orgs do not have a currencies table, any attempt to sync pricing data will fail. You can add multiple currencies to data sync if they enable multiple currencies later. For more information, see Onboarding Data Sync Services.

Data Sync: Onboarding a new non-existing consumer profile post-seed data through OnboardNewProfile API

You can use OnboardNew Profile API to onboard non-existing consumer profile post-seed data. First, invoke this API with the input (consumer name) to check if the profile already exists or not. If the entered consumer profile does not exist, it triggers onboarding for the given profile.

> ⓘ While onboarding, ensure that the *IsFirstTimeProcessing* flag is set to true and copy all out-of-the-box documents from the given consumer profile folder.

Data Sync: Ability to enable or disable criteria based sync for any entity

You can enable or disable the criteria-based sync for any entity. If enabled, the data sync considers the *turboexpiry_c* date while triggering the initial sync and delta-sync.

Data Sync: Enhancement to the sync status messages

The sync status messages are now enhanced to understand the error messages better in the Run History UI. For more information, see Working with Data Sync Run History.

Data Sync: Renaming the Synced to Processed

The Synced is renamed Processed in this release to differentiate validation errors in the Run History UI and Run Details UI. For more information, see Working with Data Sync Run History.

Data Sync: Initiate resync after adding a new formula field through UI

This release introduces a new pop-up to notify Data Sync admin about auto-Resync when adding a new formula field for any object through UI. For more information, see Adding Objects and Fields for Sync.

Data Sync: Ability to search objects by API name

The Add Object pop-up is now enhanced with the API Name column to differentiate the objects from the display and API names. This feature enables you to identify the duplicate display name created in Salesforce. For more information, see Adding Objects and Fields for Sync.

Data Sync: Ability to view the Sync Record Details

A *Details* link has been added to the Run Details page to view the status of records processed for DataSync. You can click the link to display a Sync Record Details pop-up with Datasync and Object record details. For more information, see Working with Data Sync Run History.

Fixed Issues

The following table lists the issues fixed in this release. If any actions are required, they will be listed in this table.

Case Number	Conga Internal ID	Description
00777951	LS-8432	The sum of net price on Line Item and cart are not matching in the summary group.
NA	LS-8481	In the turbo flow, you encounter an error after clicking the **Go To Pricing** button.
PST-2030	LS-8363	When the same option exists in parent and sub bundle levels, TurboEngine selects both options irrespective of the Constraint Rule.

Case Number	Conga Internal ID	Description
NA	LS-8307	The saved tier details do not appear on the tier pop-up window.
PST-1955	LS-8210	The product configuration page is not loading after clicking the **Config** button.
00773225	LS-8208	You encounter a discrepancy between classic pricing & TurboPricing while calculating the option rollup price & net price with Contract Pricing & ProductOptionPrice Records.
00771454	LS-8188	In the Turbo flow, the cart does not respond when you use the combination of field expression and AVM (Attribute value Matrix).
00772598	LS-8187	You intermittently encounter a flickering screen issue when you add Attribute Value Matrix to the product.
NA	LS-8167	When the other actions are in progress, the value of approval status disappears automatically from both configuration and cart pages.
NA	LS-8166	The **Submit for Approval** button is set as the primary action button and staus as *Approval Required* even though the cart is approved.
NA	LS-8093	The rollup lines appear in the cart when you revalidate after removing an option product's price list item.
00769811	LS-8075	The following error appears on the cart page after selecting or changing the location or performing Reprice. `Error - Could not resolve remote action: updateCartLineItemsQuietly'.`
GHGCP-340	LS-7999	The cart does not respond when you reprice after adding the discount for any product in the Turbo flow.
00769811	LS-7959	You encounter the following error If you click the **Reprice** button after selecting or changing the location. *"Error - Could not resolve remote action: updateCartLineItemsQuietly"*
PST-1580	LS-7790	The options in the configuration page get deselected automatically after performing the configuration action.

Case Number	Conga Internal ID	Description
PST-1755	LS-7778	You cannot view the updated values on the cart page due to the delay in the Salesforce data update.
PST-1664	LS-7723	In the turbo flow, the catalog page displays the Incorrect code for the product.
PST-1775	LS-7222	CPQ triggers an additional constraint rule in the turbo flow while configuring the model.
NA	LS-7843	You encounter the following error on every pricing request. `EntityMetadata not found for: 'Apttus_Config2__AssetAttributeValue__c'`
NA	LS-7839	You cannot find the UOM conversion when multiple charge types with different UOMs are resolved in the cart.
PST-1814	LS-7867	An additional inclusion rule triggers while configuring the product.
PST-1795	LS-7786	The options do not reflect in the quote when you reopen the quote.
PST-1796	LS-7781	The pending configuration symbol appears on the copied product.
NA	LS-7839	UOM conversion is not found when the multiple charge types with different UOM are resolved in the cart.
NA	LS-7224	You cannot add the object to the *Customer Specific Tables* consumer profile.
GHGCP-352	LS-7311	When the user configured the bundle using a mini cart, the configuration page was not loading as intended.
GHGCP-346	LS-7285	The cart page incorrectly displays the option groups' sequence in the turbo flow.
00115121	LS-7124	The promotions are applied to all bundles in the turbo flow even though you applied them to one bundle.

Case Number	Conga Internal ID	Description
GHGCP-364	LS-7545	The cart page did not display the attributes from the **Product Attribute Value extension** object for the products.
PST-1832	LS-7947	The default select option is not getting selected in a nested bundle.
00771752	LS-7696	The cart page triggers the constraint rule even though the condition is not met.
00771733	LS-7972	In the TurboConfig flow, the pricing gets stuck for specific products.
00770951	LS-7952	In the TurboConfig flow, the option does not get default for a few bundles.

Known Issues

The following table provides the cumulative list of known issues up to this release.

Conga Internal ID	Description
LS-8455	Even after completing the data sync, the cart page displays the following error after adding the product: `"insufficient access rights on cross-reference id".`
LS-8429	When you delete, expire, or inactivate the primary PLI of the option product and revalidate, the cart does not update the total value.
LS-8515	The **Reprice** option is displayed as a primary action instead of **Finalize** on the cart page.
LS-8484	Performing mass updates on the cart page causes delay in progress messages.
LS-8477	The product attribute value field does not clear after clicking the **Validate** button.
LS-8379	You encounter a blank cart when you set the custom view as a default view.

Conga Internal ID	Description
LS-8349	The validation error message disappears when you navigate from the configuration page to the cart page.
LS-8254	After saving or finalizing the cart, the usage price tier data is not stored on the LineItem object.
LS-8223	After selecting an option during configuration, when you click the Add More Products, the following error shows on the catalog page. • `"Id not specified in an update call"`
LS-8564	In the Turbo ABO Change flow, the cart miscalculates the pricing after adding an option that has an auto-inclusion rule.
LS-8452	You encounter an error when you apply multiple promotions with inclusion criteria.
LS-8520	The coupon applied on LineItem is not available under the coupons tab.
LS-8457	The AutoCascading Selling Term for MultiCharge Type is not working as intended.
LS-8567	PFStoPF rule does not trigger when you re-configuring the quote.
LS-8528	When you deselect the options from the sub-bundle, the inclusion show message rule is not triggered, resulting in an error message appearing on the configuration page.
LS-8518	The constraint rules do not work as intended on the configuration page.
LS-8495	The deactivated product appears on the cart page even after revalidation.
LS-8491	The COF rule does not trigger when you use the product fieldset in condition criteria.
LS-8458	An incorrect inclusion rule is triggered when you use four product fieldset in the condition criteria.

Conga Internal ID	Description
LS-8456	The constraint rule is not triggered as intended when you exclude the option using Min/Max Match Rule or include an option using another constraint rule.
LS-8445	An error icon appears beside the bundle product even though the **Isoptional** option is checked.
LS-8443	The cart page does not respond when you trigger the validation rule criteria.
LS-8428	The price list rules are not working as intended when the action criteria have conditions based on the price list.
LS-8390	When multiple sub bundles have a common option, the action product for any constraint rule, the constraint rule with auto include is selecting or deselecting correctly.
LS-8370	You intermittently encounter a cart issue when the pricing is in progress.
LS-8356	The exclusion or inclusion rule error message displays the incorrect condition product when selecting the asset option as a conditional product.
LS-8347	The revalidation message does not appear in the turbo flow while reconfiguring the quote.
LS-8329	When the PLI of the rule added product is inactive, the cart page does not display products even after reconfiguring.
LS-7961	When you copy or delete products in the section view, you encounter the following issues. • Entity deleted error while performing delete action on the cart page. • Incorrect line sequence of the products after deleting the product. • Disables the dropdown option beside Reprice on the cart page. • The deleted products are visible in other views.
CPQ-47814	In the TurboPricing flow, auto pricing occurs if you adjust any field on the cart while pricing is in progress.
CMS-700	CPQ displays an error after adding the condition and action for different countries when using Match In Location.

Conga Internal ID	Description
CMS-699	When you delete the condition of an Action, the action is also getting deleted along with the condition.
CMS-701	When configuring Match In Location, the error message is displayed continuously after deleting the actions.
CMS-702	In TurboConfig flow, CPQ displays an error after selecting the cloned condition.
CMS-703	When you update the quantity of cloned options after cloning it, the blank values are displayed for all the attributes of cloned options.
CMS-704	The original and cloned actions are deselected and added an extra clone for action after cloning a condition and action products.
CMS-705	Only one success message appears after adding the condition twice,
CMS-706	When performing repeat inclusion, an error message is not displayed after deleting the action from the mini cart.
CMS-707	When configuring Repeat Inclusion, the error message displayed for the prompt disappears.
CPQ-44694	The price factor (Bundle Only) adjustment is not supported.
CPQ-44665	The formula field on the charge type displays the incorrect values on the cart.
CPQ-44599	Promotions with the "Contains" operator in the criteria do not work.
LS-4895	Some objects in seed data are not protected (made read-only) when the option "Include all fields including future fields" is enabled.

DOC ID: CTEDEC21RN20211207

TurboEngines Documentation

Select one of the following for more information.

- About TurboEngines
- What's New in TurboEngines Documentation
- TurboEngines for Administrators

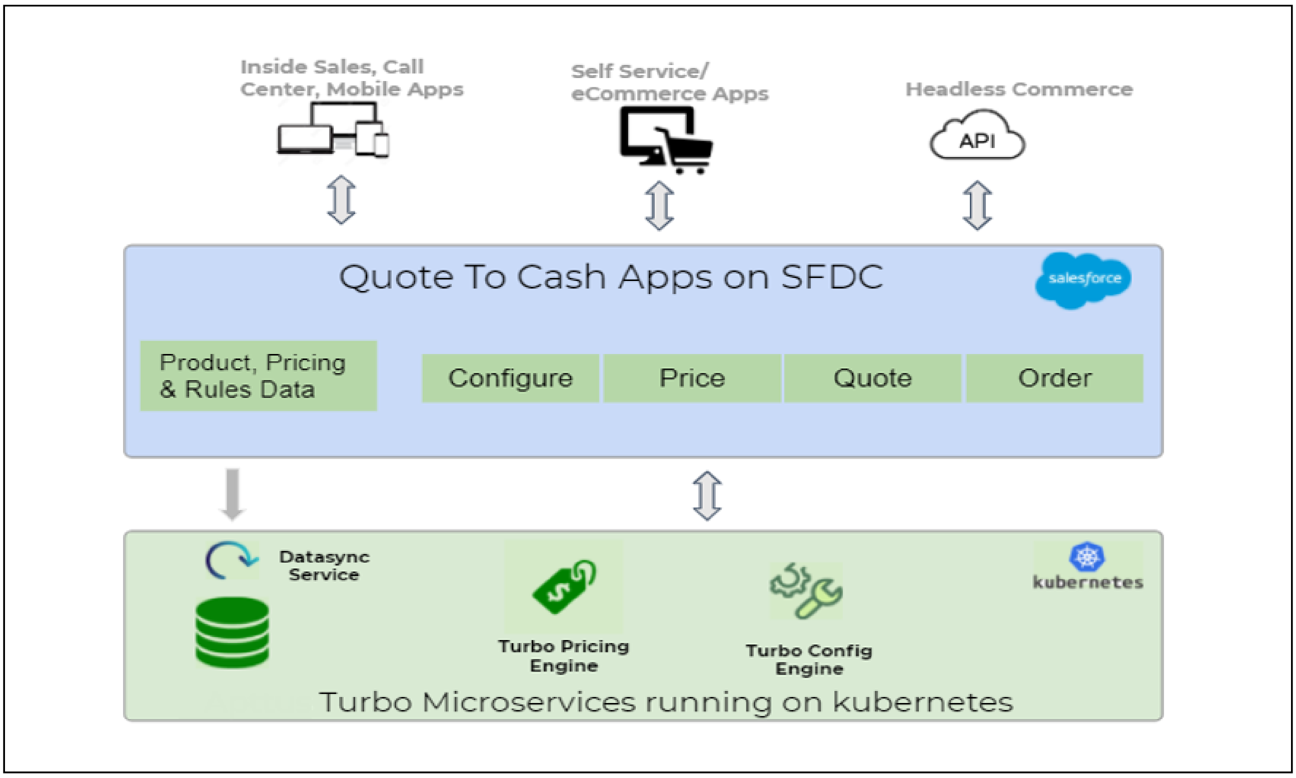
About TurboEngines

Conga TurboEngines is a concurrent processing engine provided by Conga comprising various microservices that process product configurations (TurboConfig), pricing calculations (TurboPricing), and other product-related business data, such as promotions. Conga TurboEngines offload the computation workload from the Salesforce platform to the Conga Flexible Compute Platform to reduce the processing time on the cart. In addition, processing the computation workload in the Conga Flexible Compute Platform reduces the interaction costs and the quote turnaround time, specifically during peak load or large transactions.

TurboEngines scale on the following dimensions:

- Number of users
- Size of transaction
- The complexity of the product and rules

TurboEngines also provide a critical component called TurboEngines Data Sync services that offer a high-performance mechanism to sync pricing and config master data at regular, scheduled intervals (or on-demand) between Salesforce and the Conga Flexible Compute Platform. Data is pushed to TurboPricing and TurboConfig consumer endpoints and made available for processing to take advantage of the performance improvements offered by the TurboEngines platform.



About TurboConfig

TurboConfig is a configuration engine created to process product configuration rules when products and bundles are configured on a cart and when finalizing the quote. TurboConfig offloads the computation workload from the Salesforce platform to the Conga Flexible Compute Platform built using microservices to reduce the processing time of the configuration rules. Computation workload includes the processing of rules defined on the products. For example, in a TurboConfig enabled flow, when the Sales rep adds the product or the favorite configuration to the cart, the constraint rules associated with them are offloaded to the Conga Flexible Compute Platform to process. TurboConfig engine executes the rules, maintains rule states, and avoid unnecessary line item processing.

TurboConfig is recommended when you have a large number of rules or highly complex configuration rules to be applied while selecting a product or configuring a bundle.

To get started enabling TurboConfig for your org, refer to Enabling TurboEngines in an Org. To learn more about the TurboConfig service, refer to Frequently Asked Questions (TurboConfig).

Supported Features in TurboConfig

The following features and their capabilities are supported when TurboConfig mode is enabled. For information on the listed features, see *CPQ Documentation.*

Feature	Capability	Supported
Constraint Rules	Inclusion rules	Yes
	Exclusion rules	Yes
	Validation rules	Yes
	Recommendation rules	Yes
	Replacement rules	Yes
	Product Scope: Product, Product Group, Product Family, Product Field Set	Yes
	Product Option Group scope	Yes It supports *condition* only.
	Match in Primary Lines or Options	Yes
	Match in Location	Yes
	Match in Asset	Yes
	Match in Service Asset	Yes
	Match in Cart Options	Yes
	Repeat Inclusion	Yes
	Condition Association	Yes
	Condition Criteria	Yes

Feature	Capability	Supported
	Action Criteria	Yes
	Match in Related Lines	Yes
	Is Bundle Context	Yes
Option Configuration	Min/Max Options	Yes
	Min/Max Total Quantity	Yes
	Is Hidden	Yes
	Is Picklist	Yes
	Modifiable Type	Yes
	Option Sequencing	Yes
	Default / Required Option	Yes
	Inclusion criteria	No
	Min/Max Quantity	Yes
	Quantity: Default, Modifiable	Yes
	Quantity: Auto Update	Yes
	Allow Cloning	Yes
	Config Type	No

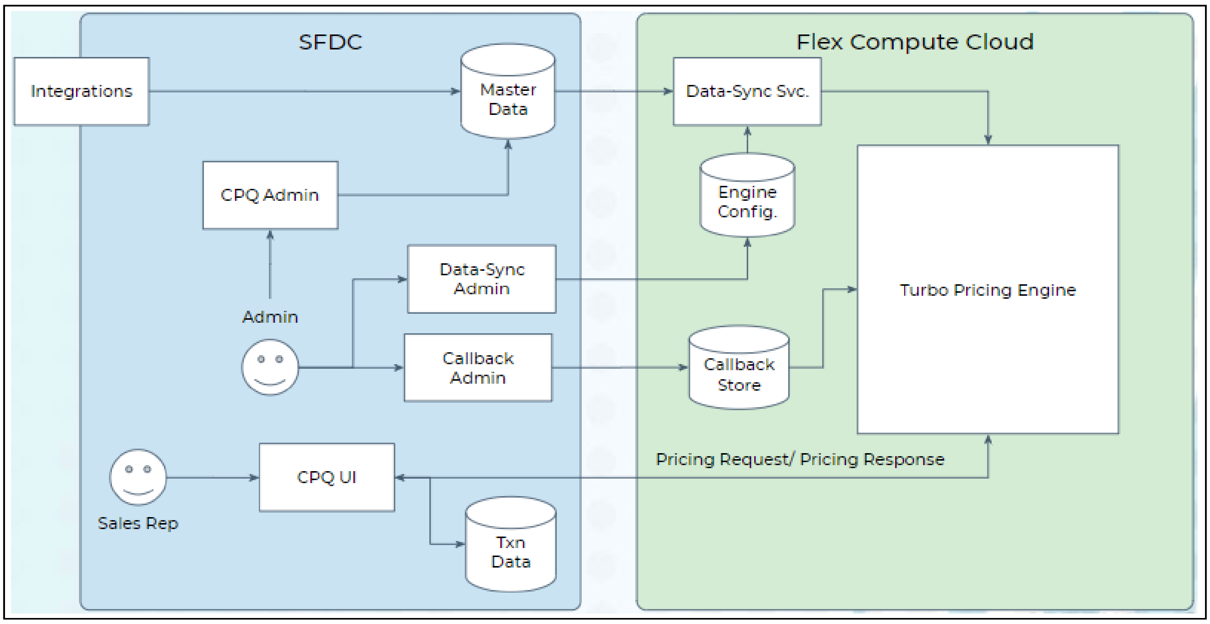
Feature	Capability	Supported
Product Attributes Display	Attribute: Read-Only, Hidden, Primary	Yes
	Two-column Attribute Display	Yes
	Three column Attribute Display	Yes
Product Attribute Rules	Product Scope: Product, Product Family, Product Group	Yes
	Filter Criteria	Yes
	Action Types: Allow, Default, Hidden, Disabled, Required, Reset	Yes
	Target field: Product Attribute Value, Line Item, Product, Pricelist, Product Configuration	Yes
Attribute Value Matrices	Product Scope: Product, Product Family, Product Group, Location	Yes
	Application Type: Default, Constraint, Force Set	Yes
Field Expressions / Rollup	Evaluation Context: Constraint Rule action, Record Update, Default Quantity, Rollup	Yes
	Update Product Attribute and Line Item object fields	Yes
	Rollup Group By Field: Line Item, Product Attribute	Yes
Callbacks	Option Filter Callback	No

Feature	Capability	Supported
Service CPQ		
Constraint Rules	Inclusion Rules: Match In Related Lines, Match In Service Asset	Yes
	Exclusion Rules: Match In Related Lines	Yes
	Validation Rules: Match In Related Lines, Match In Service Asset	Yes
Others	Formula field support (Condition/action)	Yes
	Lookups (For example, Attributes)	No
	TurboConfig Data Sync	Yes
	TurboPricing Integration	Yes
	ABO Flow support	Yes
	Service CPQ	Yes
	Multi-language support: For products & categories	Yes

About TurboPricing

TurboPricing is a pricing engine built using microservices to reduce the processing time on the cart. You can enable TurboPricing to offload complex pricing computation workload from the Salesforce platform to the Conga Flexible Compute Platform. It reduces time to submit prices to customers, improves user experience, and improves user adoption with a more responsive user interface.

The following diagram shows how data flows between a Salesforce org and Conga Flexible Compute Platform:



Supported Features in TurboPricing

The following table lists the features supported or not supported. For information on the listed features, refer to *CPQ on Salesforce Administrator Guide.*

Feature	Sub-Feature	Available in TurboPricing
Price Waterfall	Manage Price Pipeline	Yes
	Manage Price Pipeline Ruleset	Yes
Price Calculation		
Bundle Pricing	Price included in bundle set at PLI	Yes
	Price included in bundle set at Bundle	Yes
	Price Adjustments at Bundle	Yes
	Rollup method Flat at Bundle level	Yes
	Rollup method Per Unit at Bundle level	Yes

Feature	Sub-Feature	Available in TurboPricing
	Auto sequencing of options	Yes. To use this feature, you must enable it in custom settings.
	Contract pricing when the same option exists in multiple bundles	No
Defaulting Quantity	Defaulting quantity for Bundles/ Multiple Charges	Yes
	Defaulting quantity FROM Product Attribute	Yes
	Default quantity derived FROM Advanced Formula	Yes
Defaulting Term	Defaulting term for Bundles/Multiple Charges	Yes
Price List Item		
Price Method	Use of Tiered Rates*	Yes
	Use of Per Unit Price method	Yes
	Use of Flat price Price method	Yes
Frequency	Use of Daily Frequency	No
	Use of Weekly Frequency	No
	Use of Frequencies Monthly, Quarterly, Half Yearly, Yearly	Yes
	Use of any Custom Frequency	No

Feature	Sub-Feature	Available in TurboPricing
Price Type	Use of Price Type - Included Usage	No
	Use of Price Type - One Time	Yes
	Use of Price Type - Per unit	Yes
	Use of Price Type - Usage	Yes
	Use of Price Type - Recurring	Yes
LineItem Update	Read-only quantity	Yes
	Read-only Selling term	Yes
Price Methods		
Proration	Allow Proration set on PLI	Yes
Price Method Per Unit	Use of Price Method Per Unit	Yes
Price Method Flat Price	Use of Price Method Flat price	Yes
Pricing Methods		
Min/Max Price	Min/Max Price applies to BasePrice	Yes
	Min/Max Price applies to BaseExtendedPrice	No
	Min/Max Price applies to ExtendedPrice	Yes
Price Ramps	Use of Price Ramps	No
	Use of Auto Ramp creation	No
	Use of Price ramp overlap	No

Feature	Sub-Feature	Available in TurboPricing
	Use of Price Escalators	No
	Use of Ramp Option without ramping bundle	No
Defer Pricing	Defer Pricing	No
Cost Models		
Cost Models	Cost Models	No
Conversions		
Currency Conversion	Use of Multi-Currencies	Yes

Feature	Sub-Feature	Available in TurboPricing
	Use of Dated Exchange Rates	Yes, it supports the following scenarios. • List Price • Price Matrices • Product Option Price • Conditional charge type • Custom field Pricing • Manual Adjustments • Related Pricing • Price Rule Sets (if and only if currency conversion is turned on) • Price Tiers • Formula Pricing (With Reference Type) • Rounding mode • Promotion • Proration • Contract Price
	Disable Currency Conversion Rate for a Price List	Yes
	Currency conversion enabled at a rule see the level	Yes

Feature	Sub-Feature	Available in TurboPricing
UOM Conversion	Use of UOM Conversions	Yes
	Use of Product specific conversion rates	Yes
	Use of Product Family-specific conversion rates	Yes
Price Method		
Price Method	Use of Price Method Percentage	No. You must switch to Related Price List Items.
Related Pricing		
Related Pricing	Use of Related Price Lists in Pricing	No. You must switch to Related Price List Items.
	Source PLI set on Related Price List Item	Yes
	Source product and charge typeset on Related Price List Item	Yes
	Source product group and charge typeset on Related Price List Item	Yes
	Source product family and charge typeset on Related Price List Item	Yes
	Source Custom Group and charge typeset on Related Price List Item	Yes
Adjustments in Related Pricing	Adjustment defined on PLI	Yes

Feature	Sub-Feature	Available in TurboPricing
	Adjustment defined on Related Price List Item	Yes
Price Rule Set		
Header Scope and Criteria	Use of Effectivity period and Active flag	Yes
	Use of Scope Fields - price list, charge type, product family, product category, product group	Yes
	Use of Advanced Criteria	Yes
	Use of Advanced Criteria with Line Item Reference Fields	Yes
	Use of Wildcards in Advanced Criteria	Yes
	Application Level Bundle or Line Item	Yes
	Application Level Aggregate	No
	Use of StopProcessingMoreRules flag	Yes
Dimension based Price Rules	Use of StopProcessingMoreRules flag	Yes
	Use of Adjustment applies to - Base Price	Yes
	Use of Adjustment applies to - Base Extended Price	Yes
	Use of Adjustment applies to - Extended Price	No

Feature	Sub-Feature	Available in TurboPricing
Criteria based Price Rules	Use of StopProcessingMoreRules flag	Yes
	Use of Adjustment applies to - Base Price	Yes
	Use of Adjustment applies to - Base Extended Price	Yes
	Use of Adjustment applies to - Extended Price	No
	Match in Product Group	Yes
	Match in Asset	No
Price Dimension		
Use of Un-supported Price Dimension Types	Use of Un-supported Price Dimension Types - any type Except Line Item, Product Attribute, and Formula Field	No
Use of Custom Price Dimension Types	Use of Custom Price Dimension Types	No. You must convert this to Formula Field.
Service CPQ	Service CPQ	No
Adjustments		
Manual Adjustments	Line level adjustments	Yes
	Group adjustments	Yes
	Group adjustment spread	Yes
	Line-level adjustment of usage price tiers	Yes

Feature	Sub-Feature	Available in TurboPricing
	Usage Tier Modifiable	Yes
	Misc Charge Types	Yes
Adjustments	Adjustment Bucketing	No
	Ability to create multiple Adjustments	No
	Auto-refresh Usage price tiers	No
	Calculate Net Adjustment % before rounding the Net Price	Yes
Bundle/Option level manual Adjustments	Line-level adjustments	Yes
Bundle/Option level Adjustments	Group adjustments	Yes
Asset-Based Ordering		
Terminate	Terminate an asset (Standalone and Bundle)	Yes
	Edit fields Capability in the intermediate page of Terminate Operation	Yes
	ABO Cancellation after performing other operations	Yes
	Use of 'Same day cancellation' flag	Yes
	Terminate an asset from Original Asset Start Date post Change or Renew Operation (It requires Billing Services Integration in case the customer is using Conga Billing Services)	No (Net price in the cart does not consider Asset TCV values)

Feature	Sub-Feature	Available in TurboPricing
	Cancellation of Options during Change Operation gives accurate Asset TCV, Net price, and Billing schedules (It requires Billing Services Integration in case the customer is using Conga Billing Services)	No (For example, rounding off of Net Price is expected USD 6.00023, but being shown as USD 6.00000 in cart)
Change	Change an asset (Standalone and Bundle)	Yes
	Changing the Configuration of an asset (Configuration fields, Remove Options, Attributes)	Yes
Renew	Renewing an asset (Standalone and Bundle), Single and Multiple Charge Types	Yes
	Auto-renewal of an asset with Renewal Adjustment Type and Renewal Adjustment Amount	Yes
	Precedence within Auto-renewal Term, Default Renewal Term, and Selling Term while Auto-renewing an asset	Yes
	Selective display of Cotermination options during renewal	Yes
	Auto-closure Process with Quote and Agreement	Yes
Incentives		
Other application types except Promotion	Use of Price Program, Loyalty, Rebate, Milestone Incentive, or Custom	No

Feature	Sub-Feature	Available in TurboPricing
Promotions applied on Line Item and Summary Group	Promotions applied on Line Item and Summary Group	Yes
Promotions applied on other items	Promotions applied on other items except Line Item and Summary Group	No
Support for Promotion type - Own every X Get Y, Support for Promotion type - Own every X Get Y	Support for Promotion type - Own every X Get Y	No
Support for other Promotion types - except Own every X Get Y, Support for Promotion types - except Own every X Get Y	Support for other Promotion types - except Own every X Get Y	Yes
Incentive Limits	Support for Promotion Limits	Yes
Incentive Coupons	Support for Coupon Limits	No
Sales Promotions	Sales Promotions	No
Advanced Criteria set in Price Ruleset	Support for Incentive Criteria on Price Ruleset	Yes
Advanced Criteria with Reference Fields on Price Ruleset	Advanced Criteria with Reference Fields on Price Ruleset	Yes
Quotes		
Quote collaboration	Quote collaboration	No
Carts		
Favorite Configurations	Use of Favorite Configurations	Yes
Smart Carts	Use of Smart Carts	No
Submit for Approval	Submit for Approval	Yes

Feature	Sub-Feature	Available in TurboPricing
Copy	Copy products	Yes
Cart Line Item		
Revalidate	Revalidate**	Yes, Partially supported in TurboPricing.
Totaling and Summary Groups		
Adhoc Totaling	Adhoc Totaling	Yes
Deal Guidance		
Deal Guidance	Deal Guidance	Yes
Callbacks		
Callbacks	Pricing Callback	Yes
	Validation Callback	Yes
	Cart Approval Callback	Yes
	Advanced Approval Callback	No
	Loyalty Cycle Callback	No
	Bulk Loyalty Point Callback Class	No
	Adjustment Spread Callback	Yes. You can use the new TurboPricing Callback instead.
	Loyalty Point Callback	No

Feature	Sub-Feature	Available in TurboPricing
	Related Pricing Callback	Yes
	Pricing Extension Callback	Yes. You can use the new TurboPricing Callback instead.
User Experience		
Bundle-specific option line item update	Read-only quantity	Yes
	Read-only selling term	Yes
LineItem Update	Line level adjustments	Yes
	Group adjustments	Yes
Smart Search		
	Adding weightage to the products	Yes
	Using Typeahead feature	Yes

Installing TurboPricing Assessment Package

Apttus_TPAssessment package contains the TurboPricing Assessment page and related resources. Installing this package creates a TurboPricing Assessment tab. This tab includes a list of features that are currently used and supported features in TurboPricing.

> This package depends on the CPQ package (Spring '19 or later).

Pre-requisites

Rename the instance names for *Config System Properties* and *Config Custom Classes* as mentioned below.

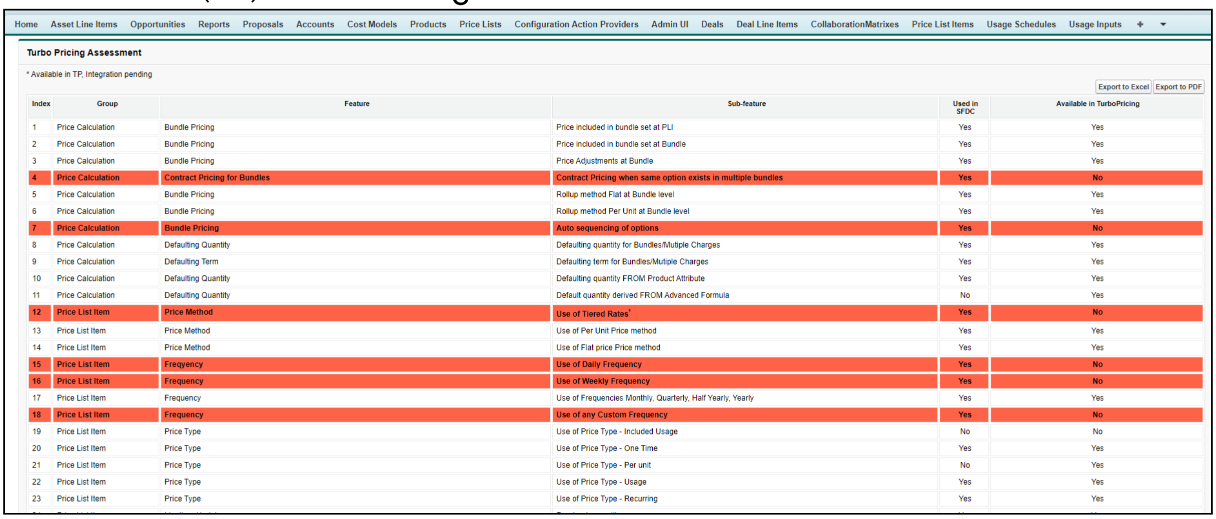
Custom Setting Name	Instance Name
Config System Properties	System Properties
Config Custome Classes	Custom Classes

To install TP_Assessment Package

1. Click the https://login.salesforce.com/packaging/installPackage.apexp?p0=04t3i000002ScHF link. The Salesforce login page is displayed.
   If you are trying to install in a sandbox or test environment, enter https://test.salesforce.com/ at the beginning of the URL.
2. Enter your user name and password, and click **Log In**. The package may require a password and the default password is *12345*.
3. After installing the TurboPricing Assessment package, provide necessary permissions to the required profiles.

Navigating the TurboPricing Assessment Tab

1. Go to https://salesforce.com/
2. Enter your user name and password, and click **Log In**.
3. Click All Tabs (➕) > **TurboPricing Assessment**.



Index	Group	Feature	Sub-feature	Used in SFDC	Available in TurboPricing
1	Price Calculation	Bundle Pricing	Price included in bundle set at PLI	Yes	Yes
2	Price Calculation	Bundle Pricing	Price included in bundle set at Bundle	Yes	Yes
3	Price Calculation	Bundle Pricing	Price Adjustments at Bundle	Yes	Yes
4	Price Calculation	Contract Pricing for Bundles	Contract Pricing when same option exists in multiple bundles	Yes	No
5	Price Calculation	Bundle Pricing	Rollup method Flat at Bundle level	Yes	Yes
6	Price Calculation	Bundle Pricing	Rollup method Per Unit at Bundle level	Yes	Yes
7	Price Calculation	Bundle Pricing	Auto sequencing of options	Yes	No
8	Price Calculation	Defaulting Quantity	Defaulting quantity for Bundles/Mutiple Charges	Yes	Yes
9	Price Calculation	Defaulting Term	Defaulting term for Bundles/Mutiple Charges	Yes	Yes
10	Price Calculation	Defaulting Quantity	Defaulting quantity FROM Product Attribute	Yes	Yes
11	Price Calculation	Defaulting Quantity	Default quantity derived FROM Advanced Formula	No	Yes
12	Price List Item	Price Method	Use of Tiered Rates*	Yes	No
13	Price List Item	Price Method	Use of Per Unit Price method	Yes	Yes
14	Price List Item	Price Method	Use of Flat price Price method	Yes	Yes
15	Price List Item	Freqyency	Use of Daily Frequency	Yes	No
16	Price List Item	Frequency	Use of Weekly Frequency	Yes	No
17	Price List Item	Frequency	Use of Frequencies Monthly, Quarterly, Half Yearly, Yearly	Yes	Yes
18	Price List Item	Frequency	Use of any Custom Frequency	Yes	No
19	Price List Item	Price Type	Use of Price Type - Included Usage	No	No
20	Price List Item	Price Type	Use of Price Type - One Time	Yes	Yes
21	Price List Item	Price Type	Use of Price Type - Per unit	No	Yes
22	Price List Item	Price Type	Use of Price Type - Usage	Yes	Yes
23	Price List Item	Price Type	Use of Price Type - Recurring	Yes	Yes

Upgrading TurboPricing Assessment package

The *Apttus_TPAssessment* package is an unmanaged package. Therefore, you can not upgrade the *Apttus_TPAssessment* package directly to the latest version. To upgrade the TurboPricing Assessment package, uninstall the old version from your org before installing the latest package.

What's New in TurboEngines Documentation

The following table lists changes in documentation to support each release.

December '21

Document	Publication Date	Topic	Description
December '21	14 May 2022	.Configuring Admin Settings	New Topic.
	25 Jan 2022	Checklist for Customized Custom Settings before Migration	New Topic.
	07 Dec 2021	Supported Features in TurboConfig	Updated features table with newly supported features. • Service CPQ • Match In Service Assets • Match In Related Lines

Document	Publication Date	Topic	Description
		Supported Features in TurboPricing	Updated features table with newly supported features. • **Asset-Based Ordering** • **Terminate** • Terminate an asset (Standalone and Bundle) • Edit fields Capability in the intermediate page of Terminate Operation • ABO Cancellation after performing other operations • Use of 'Same day cancellation' flag • **Change** • Change an asset (Standalone and Bundle) • Changing the Configuration of an asset (Configuration fields, Remove Options, Attributes) • **Renew** • Renewing an asset (Standalone and Bundle), Single and Multiple Charge Types • Auto-renewal of an asset with Renewal Adjustment Type and Renewal Adjustment Amount • Precedence within Auto-renewal Term, Default Renewal Term, and Selling Term while Auto-renewing an asset

Document	Publication Date	Topic	Description
			• Selective display of Cotermination options during renewal • Auto-closure Process with Quote and Agreement
		Creating a Connected App	Updated topic to add a new section 'Enabling the Connected App for JWT Authorization'.
		Verifying JWT Flow Settings and Access Level	New topic.

Summer '21

Document	Publication Date	Topic	Description
Summer '21	06 Jul 2021	Managing TurboEngines Callbacks	Updated topic. Added a note about the multi-tab feature.
		PricingHelper	Updated topic. Added new methods.
		Pricing Base Price Callback Interface	Updated topic. Added new extension points.
		Guidelines for TurboEngines Callbacks Coding	New topic.
		Migrating Salesforce Pricing Callback to TurboEngines Pricing Callback	New topic.

Spring '21

Document	Topic	Description
Spring '21 REV B	Supported features in TurboPricing	Updated topic. Updated the table with the newly supported feature.

Document	Topic	Description
Spring '21 REV A	Navigating the TurboEngines Callbacks Administrator User Interface	Updated topic. Updated the content and screens as per the new UI enhancements.
	Managing TurboEngines Callbacks	Updated topic. Updated the content and screens as per the new UI enhancements.
Spring '21	Supported features in TurboPricing	Updated topic. Updated the table with newly supported features.
	Supported Features in TurboConfig	Updated topic. Updated the table with newly supported features.
	Preparing Tenant Information	Updated topic.
	Version History	New Topic.
	Validate QueryModel	New Topic
	Downloading a Callback	New Topic.
	Pricing Callback Class for TurboPricing	Updated Topic. Updated the description.
	Pricing Base Price Callback Interface	Updated Topic. Added a note about the code snippet usability.
	Pricing Totalling Callback Interface	Updated Topic. Added a note about the code snippet usability.
	Related Pricing Callback Interface	Updated Topic. Added a note about the code snippet usability.
	Managing TurboEngines Callbacks	Updated Topic. Added the table with new features.

Winter '20

Document	Topic	Description
Winter 2020	Supported features in TurboPricing	Updated topic. Updated the table with newly supported features.
	Supported features in TurboConfig	Updated topic. Updated the table with newly supported features.
	DB Helper	Updated topic.
	CacheHelper	Updated topic.
	VaultHelper	Updated topic.
	Onboarding Data Sync Services	Updated topic.
	Pricing Base Price Callback Interface	Updated topic.
	Pricing Totalling Callback Interface	Updated topic.
	Related Pricing Callback Interface	New topic.
	Importing a Callback	New Topic. This feature provides a mechanism to import the callback project.
	Helper Functions for TurboEngines Callbacks	Renamed the topic from "Helper Functions for TurboPricing Callbacks".
	Configuring TurboEngines Callbacks	Renamed the topic from "configuring TurboPricing Callbacks".
	Navigating the TurboEngines Callbacks Administrator User Interface	Renamed the topic from "Navigating the TurboPricing Callbacks Administrator User Interface".
	Managing TurboEngines Callbacks	Renamed the topic from "Managing TurboPricing Callbacks".
	Syncing TurboConfig Data	Deleted Topic.

Summer '20

Document	Topic	Description
Summer 2020 Rev B	Validation Callback Class	Deleted Topic.
	Supported features in Apttus TurboPricing	Updated topic. Updated the table with newly supported features.
Summer 2020 Rev A	Configuring Data Sync for TurboPricing	Moved topic to "Onboarding data sync services".
	Navigating the TurboEngines Callbacks Administrator User Interface	Replaced screenshots.
	Managing TurboEngines Callbacks	Replaced screenshots.
	Helper Functions	Renamed the topic to "Helper Functions for TurboPricing Callbacks" and moved the new topic under "Configuring TurboPricing Callbacks".
	Helper Functions for TurboEngines Callbacks	Renamed from "Helper Functions" and moved out of "Pricing Callback Class for TurboPricing".
	CacheHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
	DBHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
	HttpHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".

Document	Topic	Description
	LogHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
	MetadataHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
	PricingHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
	VaultHelper	New topic, moved out of "Helper Functions for TurboPricing Callbacks".
Summer 2020	All topics	First release

TurboEngines for Administrators

Conga TurboEngines Administrator Guide provides information to configure TurboEngines: TurboConfig and TurboPricing. Application administrators and Conga customer administrators can also use content in this guide to perform updates to configurations, configure settings, and other microservice functionalities.

Topic	Description
What's Covered	This guide walks the administrator through the process of TurboEngines administration. It provides conceptual information, step-by-step instructions to deploy and configure Conga TurboEngines for integrated systems.
Primary Audience	• TurboEngines Implementation Teams • Customer Administrators
IT Environment	Refer to the latest *Conga TurboEngines Release Notes* for information on System Requirements and Supported Platforms.
Other Resources	• *Conga TurboEngines Data Sync Documentation* • *Conga CPQ Documentation*

This guide describes the following tasks:

- Reviewing the list of supported features for TurboConfig and TurboPricing
- Configuring Conga TurboEngines
- Enabling TurboEngines
- Completing Pre-Provisioning Tasks
    - Creating a Connected App
    - Preparing Tenant Information
- Completing Post-Provisioning Tasks for TurboConfig
    - Configuring Remote Site Settings
    - Configuring Custom Settings
    - Configuring Custom Flows
    - Syncing TurboConfig Product Data
- Completing Post-Provisioning Tasks for TurboPricing
    - Configuring TurboPricing Settings
    - Customizing TurboPricing Callbacks
    - Configuring data sync settings
    - Syncing TurboPricing Pricing Data

Before using TurboEngines, you must be familiar with the following:

- Basic Salesforce administration
- Salesforce Lightning experience
- Salesforce and Conga terms and definitions
- Conga TurboPricing Overview and Data Sync architecture
- Basic understanding of Conga TurboPricing and TurboConfig

DOC ID: CTESUM21AG20210706

Configuring Conga TurboEngines

The topics in this section provide information and step-by-step tasks for enabling TurboEngines for your organization.

> Please start with the Enabling TurboEngines in an Org topic and refer to the necessary steps you must take before and after the TurboEngines provisioning process.

- Enabling TurboEngines in an Org
- Creating a Connected App
- Preparing Tenant Information

- Post-Provisioning Tasks (TurboConfig)
- Post-Provisioning Tasks (TurboPricing)
- Onboarding Data Sync Services
- Logging APIs

Enabling TurboEngines in an Org

This topic provides a summary of the necessary steps for enabling TurboEngines (TurboConfig and TurboPricing) for your org.

An administrator can be any of the following persona: Customer Administrator, Partner Administrator, any other administrators assigned the responsibility of enabling TurboEngines for their org. In the table on this topic, this persona is referred to as the *Tenant Admin*.

Prerequisites

- Check the "Supported Features" topics (under About TurboEngines) for the service you want to enable. Make sure all of the features you want are included before making a provisioning request.
- You must have the appropriate TurboEngines license before turning on your org. If you do not have a license, please reach out to your Conga Account Executive.
- You must have the Summer 2020 or later build of Conga Configuration & Pricing (Conga CPQ) in the Salesforce org to enable TurboConfig and TurboPricing. Refer to "Packages" in the latest Conga *CPQ Release Notes*.

Enabling TurboEngines

To enable TurboEngines, perform the following steps for each org:

Step	Task	Owner	Description
Pre-Provisioning Tasks			
1	Set up Connected App in your org	Tenant Admin	Create a connected app to provide authentication and authorization to TurboConfig and TurboPricing Data Sync Service.
2	Prepare pre-provisioning tenant information	Tenant Admin	Gather all required information for provisioning your TurboConfig or TurboPricing org. Provide this information to Conga Technical Support to begin the provisioning process.
Post-Provisioning Tasks Perform the following steps only after receiving a notice from Conga Technical Support that the requested orgs are provisioned. You must have the new service URLs to proceed.			
3	Set up Remote Site Settings (TurboConfig)	Tenant Admin	Use the service URL you received from Conga Technical Support to set up the remote site settings for TurboConfig.

Step	Task	Owner	Description
4	Configure Services	Tenant Admin	For TurboConfig, update the following settings: 1. Set up the Config Execution Mode 2. Set up the custom flow and Configure Products button For TurboPricing, update the following settings: 1. Set up the Pricing Execution Mode 2. Set up the TurboPricing endpoint URL
5	Configure TurboEngines Callbacks	Tenant Admin	Configure TurboEngines Callbacks. • TurboPricing Callbacks
6	Configure data sync service	Tenant Admin	Configure specific settings to onboard data sync services.
7	Sync data to TurboEngines	TurboEngines Administrator (can be Tenant Admin)	Set up and schedule or activate data sync to sync master data.

Creating a Connected App

As part of the pre-provisioning process, you must configure a Connected App in your org to provide authentication and authorization for the following TurboEngines services:

- TurboConfig
- TurboPricing

> ⓘ The example in the following tasks is provided for TurboConfig but is the same process for any service configuration.

To create a Connected App

1. Navigate to **Setup > App Setup > Create > Apps.**
2. Scroll down and search for the **Connected Apps** related list and click **New** to create a new app.
3. Fill in the following details in the **Basic Information** section.

Field	Description
Connected App Name	Enter the name of the Connect App.
API Name	The API name is generated automatically based on the name of the Connected App.
Contact Email	Enter the email address of the administrator managing the Connected App.

4. Fill in the following details in the **API (Enable OAuth Settings)** section.

Fields	Description
Enable OAuth Settings	Select this to define the OAuth settings. For example, *TurboConfig*. When you enable this field, additional settings are displayed under **API (Enable OAuth Settings)** section.
Enable for Device Flow	Select this to enable the connected app for an external application.

Fields	Description
Callback URL	The **Callback URL** is generated by default when you select the field **Enable for Device Flow**. For example, *https://test.salesforce.com/services/oauth2/success* is generated based on the instance URL. You can also add other URLs in separate lines.
Selected OAuth Scope	Select all the entries under **Available OAuth Scopes** and move them to **Selected OAuth Scopes** by clicking the **Add** arrow.
Require Secret for Web Server Flow	Select this to require the connected app to provide a consumer secret for authorization.

5. You must leave all other fields blank. Click **Save**.

To capture Consumer Key

After you create a Connected App, CPQ generates **Consumer Key** and **Consumer Secret**. You must provide the value of **Consumer Key** to Conga Technical Support.

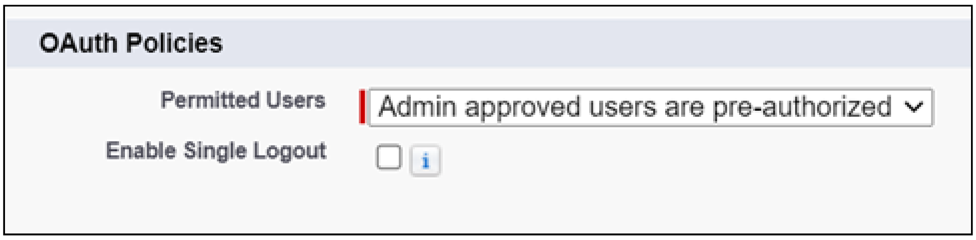
1. Navigate to **Setup > App Setup > Create > Apps.**
2. Scroll down and search for the **Connected Apps** related list.
3. Click the name of the Connected App you created in the previous topic.
4. Click **Copy** next to **Consumer Key**.
5. Store the information for the next part of the process.

Enabling the Connected App for JWT Authorization

Enabling the Connected App for JWT Authorization is mandatory for all the new consumers. For this, you require a Public Key (certificate) which you can collect from Conga TS or Conga Ops.

1. Setting up the Permitted Users.
    a. Navigate to **Setup** > **Apps** > **App Manager**.
    b. Click the icon of required **Connected App**> **Manage.**
    c. Click **Edit Policies**. The Connected App edit page is displayed.

d. In the OAuth Policies section, select **Admin approved users are pre-authorized** from the **Permitted Users** option.



e. Click **Save**.

2. Uploading the Public Key (certificate) to Connected App.
    a. Navigate to **Setup** > **Apps** > **App Manager**.
    b. Click the icon of required **Connected App**> **Edit**.
    c. Enable **Use Digital Signatures.**
    d. Click **Choose File** and upload the public key (certificate) provided by the *Conga TS* or *Cloud Ops.*

    e. Click **Save**.
3. Associating correct access for the Connected App.
    a. Navigate to **Setup** > **Apps** > **App Manager**.
    b. Click the icon of required **Connected App**> **Manage.**
    c. If you are using Profiles or Permission Sets for OAuth applications, Perform the following based on your organization policy.
        i. To associate with the profile, click **Manage Profiles** > click the checkbox next to the required profile or
        ii. To associate with the permission sets, click **Manage Permission Sets** > click the checkbox next to the required permission set.
        iii. Click **Save**.
4. Once the tenant onboarding is completed, you must verify a Connected App's JWT flow settings and access levels and ensure that the JWT related feature flag (**is-certbased-jwt-auth-enabled)** is enabled for your organization.

Preparing Tenant Information

Your provisioning request for TurboConfig or TurboPricing must include specific information related to your tenant. Before your org can be provisioned, you must gather the required information and provide it to Conga Technical Support. What information must be collected will differ depending on the service you are provisioning.

> ⓘ Configure a Connected App to use with TurboEngines before collecting the information described in this topic.

Refer to the following table for all required pre-provisioning information:

> ⓘ While implementing TurboPricing, any Id (15-character Id) that is returned from Salesforce is converted to an 18-character Id for the proper functioning of TurboPricing.

Configuration	Required for Service	Description
OrgId	TurboConfig, TurboPricing	This is the Salesforce Organization ID of the org to be provisioned for TurboEngine service. To locate your Organization ID: 1. Log in to the org to be provisioned. 2. Go to **Setup** > **Company Profile** > **Company Information** > **Salesforce.com Organization ID**. 3. Copy the 15-character ID (to be converted into 18 characters). You can add any random characters to the Org ID for conversion. For example, TenantId = *00d3i000000qn7xAAA*
Tier	TurboConfig, TurboPricing	The Tier (Gold, Silver, and Bronze) to be provisioned. If no tier is provided, then **Bronze** is selected by default.
Org Type	TurboConfig, TurboPricing	Org type to be provisioned (sandbox or production)
Tenant Name	TurboPricing, TurboConfig	The one-word tenant name used for the tenant endpoint (for example, *customername-sandbox*)
Consumer Key	TurboConfig, TurboPricing	The consumer key (`client-id` in OAuth 2.0) generated from your Connected App. Refer to Creating a Connected App.
Consumer Secret	TurboConfig, TurboPricing	The secret key (`client-secret` in OAuth 2.0) generated from your Connected App. Refer to Creating a Connected App. This is not required if you have enabled the JWT authorization.

Configuration	Required for Service	Description
Salesforce User Name	TurboConfig, TurboPricing	Admin username for the org to be provisioned with read/write access to Conga CPQ (used by Conga Technical Support for verifying settings)
Salesforce Password	TurboConfig, TurboPricing	Password for the Salesforce admin user. This is not required if you have enabled the JWT authorization.
Authority	TurboPricing	The URL used to verify session Id for TurboPricing (*login.salesforce.com*, *test.salesforce.com*, or a custom Salesforce domain)
InstanceURL	TurboPricing	The URL is given by the UI after logging into the org to be provisioned (for example, *customerturbo.my.salesforce.com*)
OAuthTokenURL	TurboConfig	Salesforce token endpoint URL (this will be *login.salesforce.com* or *test.salesforce.com*, depending on your Org Type)
Contact Email	TurboConfig, TurboPricing	The email to send the notifications on the successful provision.
Data Volume Settings	TurboConfig, TurboPricing	Provide the following entity information to Cloud Ops to perform the sizing accordingly. Refer to Data Volume Settings for Tenant Onboarding. • Estimated Total Number of Search Filters • Estimated Total Number of Asset Line Items • Estimated Total Number of Deal Guidance Rules • Estimated Total Number of Price Rules • Estimated Total Number of Price List Items • Estimated Total Number of Price Matrixes • Estimated Total Number of Products (count of product records in product2 object)

Configuration	Required for Service	Description
Activate Multiple Currencies	TurboPricing	Verify if the *Activate Multiple Currencies* option is selected for an org. • If the option is selected, add `"MultipleCurrenciesEnabled": true` in the tenant configuration file with the help of DevOps/CloudOps, • If the option is not selected, add `"MultipleCurrenciesEnabled": false` in the tenant configuration file with the help of DevOps/CloudOps.

After collecting all the required information, provide it with your tenant provisioning request to Conga Technical Support.

Checklist for Customized Custom Settings before Migration

This section provides information about additional steps to consider when migrating one environment to another.

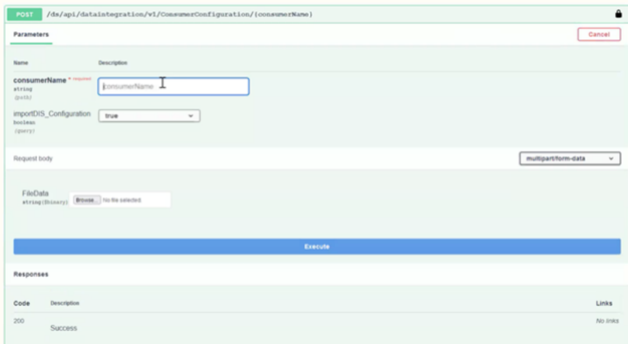
Generally, customers make a few additional configurations such as custom settings, pricing callback, etc., to use TurboPricing. When migrating to another environment, ensure that the additional configurations are saved and transferred to the second environment. It helps you avoid the rework and errors that can delay the project. You must take the additional steps, especially when Refreshing a sandbox from production, Migrating from one sandbox to another, and Promoting from Dev > QA (Quality Assurance) or QA > UAT (User Acceptance Testing) or UAT > Production.

S. No.	Step	Description
1	Saving TurboPricing Callbacks	• You must perform this step before raising the CloudOps ticket to re-provision the TurboPricing instance to the second environment. • Download the following Pricing callbacks through callbacks authoring user interface, which is a part of the base pack, and upload them in the second environment. • Pricing BasePrice Callback • Pricing Totaling Callback • Related Pricing Callback

S. No.	Step	Description
2	Export your changes to the DataSync service configuration	• You must perform this step before raising the CloudOps ticket to re-provision the TurboPricing instance to the second environment. • Invoke the following API to download a zip file with the configuration by providing a consumer profile as input. • `GET /ds/api/dataintegration/v1/ConsumerConfiguration/{consumerName}` • Export the zip file to the data sync service configuration for each consumer profile. For example, the Consumer Profiles of TurboPricing are *elasticsearch* and *actor.* This API is available from the TurboPricing version 21.8.0.

S. No.	Step	Description
3	Document Changes to Custom Settings specific to TurboPricing	You must make the changes to the documents if you have made the These custom settings changes my Suppose you find any changes in the TurboPricing in the custom settings and try to refresh the environment. You must export these settings to the new environment before the refresh. If there is already an established process to export and import custom settings, ensure these settings are part of the process. **3.1 Saving Custom Setting changes for Flow-specific turning on of TurboPricing** If TurboPricing is selectively used and managed through Flows, the system creates more Config System Property records. In contrast, Classic Pricing will have only one Config System Property file. **3.2 Saving Custom Setting changes made to move Formula Field processing to TurboPricing** Suppose there are formula fields on the line item displayed to the end-user in the cart or used for pricing. In that case, TurboPricing calculates those fields to enhance user experience due to multiple roundtrips while pricing the cart. These custom setting changes must be a part of your migration plan. However, if this is a refresh or the TurboPricing instance is only being re-provisioned, these settings must be saved or referenced to ensure TurboPricing runs smoothly in the newer environment. 1. Config System Properties > View Cart Custom Fields 2. Config System Properties > View Cart Custom Fields 2 3. Config LineItem Custom Fields The above settings provide information about the fields calculated by the TurboPricing engine. The third setting lists the reference fields used in the formula expression specified in the previous settings.

S. No.	Step	Description
4	Raising Cloud Ops tickets to re-provision the TurboPricing Instance	Refer to the Preparing Tenant Information of TurboPricing Documentation to provision TurboPricing to the second Salesforce Environment. If the salesforce environment is a single currency environment, you must provide that information in the Cloud Ops ticket. Also, If you perform the formula field process in TurboPricing, turn on this feature and specify the same in the Cloud Ops Ticket. The administrator must perform all the pre-provisioning steps before creating the Cloud Ops Ticket. After provisioning the instance. The administrator must perform the post-provisioning steps, except performing the initial data-sync as it should be performed after completing all the steps in this section.

S. No.	Step	Description
5	Making Custom Settings changes saved from the previous environment	Perform this step before initiating the initial data sync in the new environment. A forced resync may require if the initial data sync is performed before this step. Invoke the following API to import the Data Sync Service Configuration for each Consumer Profile. For example, the Consumer Profiles of TurboPricing are *elasticsearch* and *actor*. This API helps you download a zip file with the configuration of the Consumer Profile provided as input. `POST /ds/api/dataintegration/v1/ConsumerConfiguration/{consumerName}` This API is available from the TurboPricing version 21.8.0. If you are using a Swagger, set *importDIS_Configuration* to *false* to upload only tables and fields while skipping other configuration details that can be set afresh in the new environment. Next, browse and add the Zip file downloaded in step 2. 
7	Importing Callbacks	Refer to the Navigating the TurboEngines Callbacks Administrator User Interface and Managing TurboEngines Callbacks topic to import the callback projects.

Data Volume Settings for Tenant Onboarding

To handle the high volume data when onboarding a new tenant, update the *OverrideSettings* of *Onboarding API* with the help of Cloud Ops or Dev Ops.

- The following settings are required only when onboarding a new tenant.
- Upgrading an existing tenant will not update shards(elastic doesn`t support updating shards after creating index). However, onboarding API will not throw any error if you pass the records count.

To update the *OverrideSettings*:

1. Collect the total records count from Professional Services for the entities below and update the respective counts in the *OverrideSettings* object.

```
"OverrideSettings": {
    "APTTUS_APPROVAL__SEARCHFILTER__C": 10000,
    "APTTUS_CONFIG2__ASSETLINEITEM__C": 10000,
    "APTTUS_DEALMGR__DEALGUIDANCERULEENTRY__C": 10000,
    "APTTUS_CONFIG2__PRICERULEENTRY__C": 10000,
    "APTTUS_CONFIG2__PRICELISTITEM__C": 21000000,
    "APTTUS_CONFIG2__PRICEMATRIXENTRY__C": 10000,
    "PRODUCT2": 6100000
  }
```

2. Pass the updated *OverrideSettings* object in the body of the following API. This API adds shards for the new indices based on the record count and updates the elastic config for the respective index info section.

<table>
<tr><th>API Details</th><th>Example</th></tr>
<tr><td>POST /Pricing/dataadmin/onboardtenant
<table>
<tr><th>Parameter</th><th>Data Type</th><th>Required?</th></tr>
<tr><td>doTemplateOverride</td><td>Boolean</td><td>Yes</td></tr>
</table>
</td><td>POST /Pricing/dataadmin/onboardtenant</td></tr>
</table>

Enabling JWT Authorization

Currently, all turbo applications use the username and password authentication flow along with connected app secret for data sync. However, this flow is not recommended for server-to-server communication due to the following challenges:

- **Security Risks:** Sharing sensitive data as username and password may have a chance of uninvited access.
- **Authentication Issues:** Turbo applications will fail at authentication Whenever the user changes the password.
- **Maintenance:** The Conga Support teams (TS) team has to change passwords in the tenant config file whenever the user changes them.

To overcome these issues, the TurboData team recommends using a JSON Web Token (JWT) based authentication for server-to-server communication where the c*ertificate* is used instead of the password to enhance security and user experience. JWT is a JSON encoded representation of a request(s) transferred between two parties and it allows the users to share the information securely. However, the validity of this certificate is one year, and the admin has to update the certificate once a year. Thus, changing the password will not affect the authentication.

Advantages of JWT based authentication:

- **Reduces Security Risks:** Reduce security risk as no password or client secret is stored.
- **Low Maintenance:** Users can change their passwords as required. If the user changes the password, redeployment is not required for configuration changes.

Sample Tenant Config (click to expand)

```
{
  "TenantId": "00XXXXXXeau",
  "TenantName": "Customer1Prod",
  "OpenIdConfiguration": {
    "Authority": "https://login.salesforce.com",
    "SalesforceUsername": "pricXXXXXdev",
    "ClientId": "3MVG9vXXXXXXXmn609",
    "RSAPrivateKey": ""
  },
  "ProcessorConfiguration": {
    "Parallel": 8,
    "BufferSize": 500
  },
  "DataSource": {
```

```
    "Type": "ElasticSearch",
    "Configuration": {
      "ServiceUrl": "https://ls-ibm-engg-elasticdata.apttuscloud.io:9200",
      "SuperAdmin": {
        "UserName": "elastic",
        "Password": "ElasXXXXXX90"
      },
      "ReadonlyUser": {
        "UserName": "elastic",
        "Password": "ElastXXXX90"
      }
    }
  },
  "CacheConfiguration": {
    "ConnectionString": "XXXXX"
  },
  "LightsaberDatabase": {
    "ConnectionName": "lightsaberdb",
    "MongoDbName": "iclsdev1-lsdb",
    "MongoDbConnectionString": "mongodb://localhost:27017"
  },
  "DatasyncConfiguration": {
    "InstanceUrl": "https://XXXXX.my.salesforce.com",
    "OAuthTokenUrl": "https://login.salesforce.com/services/oauth2/token",
    "SalesforceUsername": "pXXXev",
    "SalesforcePassword": "XXX0",
    "ClientId": "3MXXXXX609",
    "ClientSecret": "87XXXX9",
    "MongoDbConnections": [
      {
        "ConnectionName": "syncconfiguration",
        "MongoDbName": "iclsdev1-ds",
        "MongoDbConnectionString": "mongodb://localhost:27017"
      }
    ],
    "KafkaConfiguration": {
      "EventsTopicName": "XXX-datasyncEvents",
      "TriggerTopicName": "XXX-datasyncTrigger",
      "SchedulerTopicName": "XXX-datasyncScheduler",
      "PushTopicCreatedTopicName": "XXX-datasyncPushTopic",
      "ConsumerOrchestratorTopicName": "XXX-consumerorchestrator"
    }
  },
```

```
  "ElasticAdminApiEndPoint": "http://data-admin-api",
  "ActorApiEndPoint": "http://XXXXX.local",
  "x-forwarded-host": "XXXX.apttuscloud.io",
  "CMSApiEndpoint": "<CMS-PublicURL>"
}
```

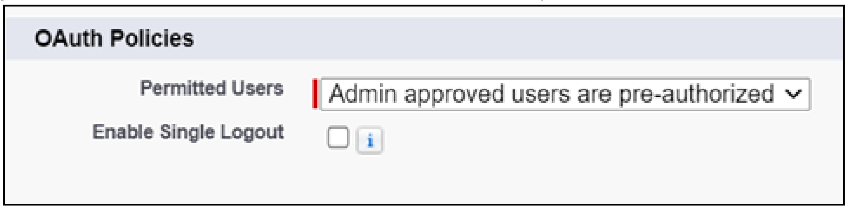
Prerequisites:

- **Public Key (Certificate)**: You must collect the key (certificate) from conga TS or Conga Ops.
- Existing users can invoke the following APIs (using datasync swagger) before enabling JWT authorization for verification purposes after enabling JWT.
  - `/ds/api/dataintegration/v1/Validate` - Trigger this API and copy the validation Id (guId) from the response.
  - `/ds/api/dataintegration/v1/Validate/GetValidationResults` - Trigger this API by passing validation id copied from the response of the above API. Then, save the response.

Enabling JWT authorization covers five major processes:

1. Identifying the Connected App
2. Setting up the Permitted Users
3. Uploading Public Certificate
4. Associating correct access for the connected app
5. Enabling JWT OAuth using a feature flag

Order	Action	Description
1	Identifying Connected App	If you are not aware of which connected app is currently used for Turbo, contact TS/PS/Conga Ops team to get the ClientId (also called Consumer Key) and verify against existing connected apps in Salesforce org.

Order	Action	Description
2	Setting up the Permitted Users	Once you have identified a connected app, perform the following. 1. Login to Salesforce Org > **Setup** > **Apps** > **App Manager**. 2. Click the icon of required **Connected App**> **Manage.** 3. Click **Edit Policies.** The Connected App edit page is displayed. 4. In the OAuth Policies section, select **Admin approved users are pre-authorized** from the **Permitted Users** option. 
3	Uploading Public Certificate	After receiving the Public Key from Conga TS or Conga Ops. 1. Login to Salesforce Org > **Setup** > **Apps** > **App Manager**. 2. Click the icon of required **Connected App**> **Edit**. 3. Enable **Use Digital Signatures.** 4. Click **Choose File** and upload the public key (certificate) provided by the *Conga TS* or *Cloud Ops*. 
4	Associating correct access for the Connected App	As per the organization's policy, select either Profile or Permission Sets, which has complete access to all turbo objects and fields configured in the consumer profiles. To associate the access for Connected App : 1. Login to Salesforce Org > **Setup** > **Apps** > **App Manager**. 2. Click the icon of required **Connected App**> **Manage.** 3. Do one of the following: a. To associate with the profile, click **Manage Profiles** > click the checkbox next to the required profile > **Save** or b. To associate with the permission sets, click **Manage Permission Sets** > click the checkbox next to the required permission set > **Save**.
Post-Onboarding Task		

Order	Action	Description
5	Enabling JWT OAuth using a feature flag	• All existing customers must perform this step after enabling JWT. • All new tenants must perform this after completing the onboarding tasks. After making the changes to the Connected App, contact Conga TS or Conga Ops for enabling the JWT OAuth using the feature flag called **is-certbased-jwt-auth-enabled**. Once this feature flag is enabled, run validation APIs against your org to check the access and permissions. For more information, see Verifying JWT Flow Settings and Access Level.

Post-Provisioning Tasks (TurboConfig)

The post-provisioning process for TurboConfig are divided into two main tasks:

- Configuring the service (custom settings and properties)
- Setting up and activating data sync for config master data from SFDC to TurboConfig

Refer to the following topics for step-by-step instructions to complete setup and configuration of TurboConfig:

- Configuring Remote Site Settings for TurboConfig
- Configuring TurboConfig Settings
- Configuring Custom Flows for TurboConfig

Configuring Remote Site Settings for TurboConfig

To create a Remote Site record

1. Go to **Setup** > **Administration Setup** > **Security Controls** > **Remote Site Settings**.
2. Click **New Remote Site**.
3. Enter the name in **Remote Site Name**. For example, *TurboConfig.*
4. Enter the URL for the remote site in the **Remote Site URL**.

- Contact Conga Technical Support for the Remote Site URL.

- Do not enter the '/' symbol at the end of the Remote Site URL.

5. Enable **Active**, if not selected by default.
6. Click **Save**.

Configuring TurboConfig Settings

You can enable TurboConfig either globally or for select CPQ flows. By default, the global **Contraint Rule Execution Mode** is set to *Client*. To enable the TurboConfig for select flows, refer to Configuring Custom Flows for TurboConfig. Follow the steps below to enable TurboConfig at the global level.

Take note of the following before you enable TurboConfig:

- Refer to Supported Features in TurboConfig topic to confirm that the features you use are supported in TurboConfig before enabling it globally.
- If you want to enable TurboConfig for certain types of quotes only, create a new flow with execution mode as *CMS*. Refer to Configuring Custom Flows for TurboConfig
- Once you create a quote using *CMS* execution mode you cannot switch to *Client* mode for that particular quote.
- The constraint rule execution mode (*Client* or *CMS*) is determined at the beginning of the quote. Once determined you cannot change them.
- The constraint rule execution mode *CMS* works on standalone products but the products must be published individually or in groups.

To enable TurboConfig on the quote

1. Go to **Setup** > **App Setup** > **Develop** > **Manage Custom Settings.**
2. Click **Config System Properties**. Click **Manage**.
3. Click **Edit** next to **System Properties**.
4. Define the fields as explained in the table below:

Field	Description
Constraint Rule Execution Mode	Enter the value *CMS* in the field.
CMS End Point URL	Enter the End Point URL for TurboConfig.

- CMS End Point URL is provided by Conga Technical Support.
- Do not enter the '/' symbol at the end of the CMS End Point URL.

5. Click **Save**.

Configuring Custom Flows for TurboConfig

You can enable TurboConfig for selective CPQ flows. You can use this functionality to avoid making TurboConfig as the default configuration engine and use the engine to process large and complex configuration rules. To enable TurboConfig for a specific flow, you must create a dedicated data set of **Config System Properties**.

To enable TurboConfig for specific flows

1. Create a custom flow. Refer to "Configuring Flow Settings" in *CPQ on Salesforce Administrator Guide*.
2. Create a formula action at the **Quote/Proposal** object for the flow you created in Step 1. Refer to "Creating Custom Buttons for Different Flows" in *CPQ on Salesforce Administrator Guide*.
3. Go to **Setup** > **App Setup** > **Develop** > **Custom Settings.**
4. Click **Config System Properties**. Click **Manage**.
5. Click **New** to create a new data set.
6. In the **Name** field, enter the name of the custom flow you created in Step 1.
7. Define **Constraint Rule Execution Mode** and **CMS End Point URL** as described in Step 4 in the topic Configuring TurboConfig Settings.
8. Click **Save**.

The Sales rep must use the custom flow that you created to configure the quote using TurboConfig.

Post-Provisioning Tasks (TurboPricing)

The post-provisioning process for TurboPricing are divided into four main tasks:

1. Configuring the service (custom settings and properties)
2. Configuring data sync service for use by administrators
3. Configure custom pricing callbacks
4. Setting up and activating data sync for pricing master data from SFDC to TurboPricing

Refer to the following topics for step-by-step instructions to complete setup and configuration of TurboPricing:

- Setting Up the TurboPricing Endpoint URL
- Setting Up the Pricing Execution Mode
- Configuring TurboEngines Callbacks
- Managing Turbo Formula Fields

Setting Up the TurboPricing Endpoint URL

This section provides information for setting up the TurboPricing endpoint URL in the org.

1. Click the **All Tabs** icon () and click **Admin**. The Home page is displayed.
2. Click **New**. The New Admin page is displayed.
3. In the **Name** field, enter *APTS_PricingServiceOverrideURl*.
4. In the **Value** field, enter the TurboPricing endpoint URL (without https://).
5. Click **Save**.

> Do not enter the forward-slash (/) symbol at the end of the Endpoint URL.

Setting Up the Pricing Execution Mode

This section provides information for setting up the Pricing Execution Mode in the org.

1. Go to **Setup** > **App Setup** > **Develop** > **Events** > **Custom Settings**.
2. Click **Config System Properties**. Click **Manage**.
3. Click **Edit** next to System Properties.
4. In the **Pricing Execution Mode**, enter *Turbo*.
5. Click **Save**.

Configuring TurboEngines Callbacks

This topic provides information on configuring TurboEngines callbacks.

Callbacks provide you with a mechanism to apply custom logic at the extension points provided and get invoked during the pricing computation by the TurboEngines. For example, you can apply custom pricing on the line items in the cart using the TurboEngines Callback Class. Callbacks are implemented using interfaces that are specific to each callback. These interfaces have various methods that you can use to achieve your task. You must implement the interface in a C# class and within that class, you may use your custom logic using the methods of the interface.

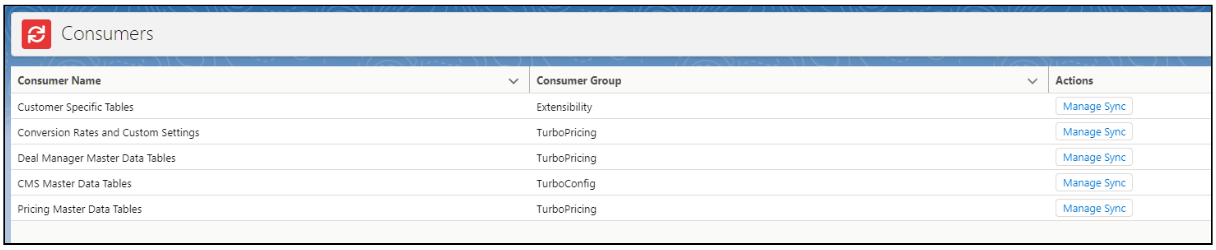
The sections in this topic provide information for:

- Navigating the TurboEngines Callbacks Administrator User Interface
- Managing TurboEngines Callbacks
- Helper Functions for TurboEngines Callbacks
- Pricing Callback Class for TurboPricing

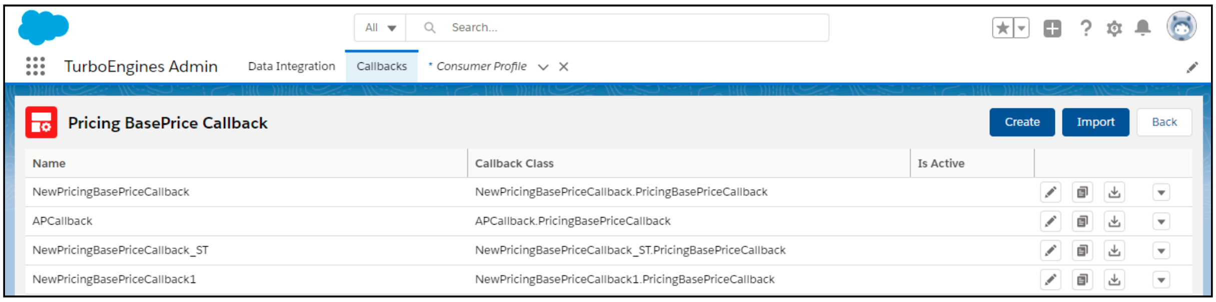
Navigating the TurboEngines Callbacks Administrator User Interface

This section provides information on navigating the TurboEngines callback administrator user interface.

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed.

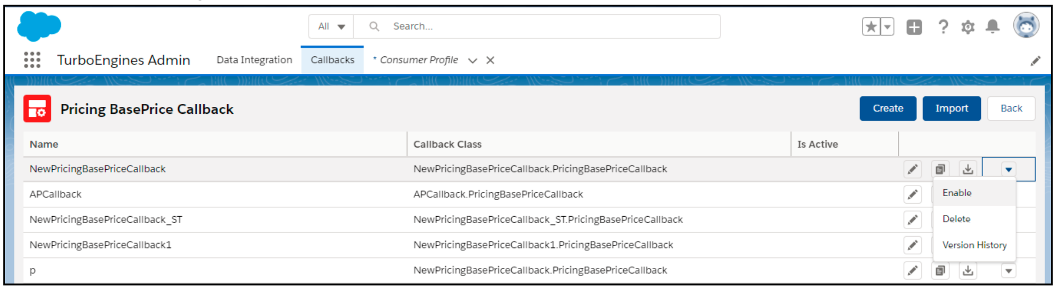


   If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.



6. Click **Create** to create a new project.
7. Click **Import** to import an existing project.
8. Click one of the following icons for the required callback:
   - Edit ( )
   - Clone ( )
   - Download ( )
   - Click the Action icon ( ) and select one of the following options:
     - Enable or Disable
     - Delete

- Version History



9. From the Explorer panel, create a file under the current project if required. See Navigating the TurboEngines Callbacks Administrator User Interface.
10. Test your code before saving it. See Navigating the TurboEngines Callbacks Administrator User Interface.
11. Click the **Code Difference** icon ( ) to see the difference between the original and modified code. The differences are highlighted for easy identification.
12. At the bottom of the code editor panel:
    - Click the **Collapse** icon ( ) to show the panel and **Expand** icon ( ) to hide the panel.
    - **Output**: This tab displays the output of your code. Whenever you execute a method, the returned result is displayed in this tab.
    - **Input**: This tab displays the input of your code. You can verify what values CPQ has set for parameters or what values CPQ retrieves by reference for a parameter when you execute some code.
    - **Profiler**: This tab displays the order of execution of methods and performance of methods. You can also check how long CPQ takes to execute each method.
    - **Console**: This tab displays the log statements that you have added in the callback code when you test the callback code using the authoring UI.

Managing Files and Folders in the Explorer Panel of the Edit Project Page

This section describes how you can manage files and folders in the Explorer panel of the edit project screen. You can click the Collapse icon ( ) to show the panel and Expand icon ( ) to hide the panel.

Adding an Item

1. Select a folder if any and click the **Add Item** icon ( ). The Add Item pop-up is displayed.
2. From the Type drop-down, select what type of item you want to add. The supported values are File and Folder.
3. Enter a name for the item in the **Name** field.

4. Click **Create**.

Renaming an Item

1. Select a file to be renamed.
2. Click the **Rename Item** icon ( ). The Rename Item pop-up is displayed.
3. In the **Name** field, enter a new name and click **Save**.

Deleting an Item

1. Select a file to be removed.
2. Click **Delete Item** icon ( ). The Delete Item pop-up prompting you to confirm deletion is displayed.
3. Click **Yes** to delete the item.

> If there is only one item, the Delete Item icon ( ) is disabled. You must have at least one item on the edit project page.

Testing a Callback Method by Executing the Code

CPQ allows you to test the code you have written for a callback method before saving it. In the **Test Run** panel, click the **Collapse** icon ( ) to show the panel and **Expand** icon ( ) to hide the panel.

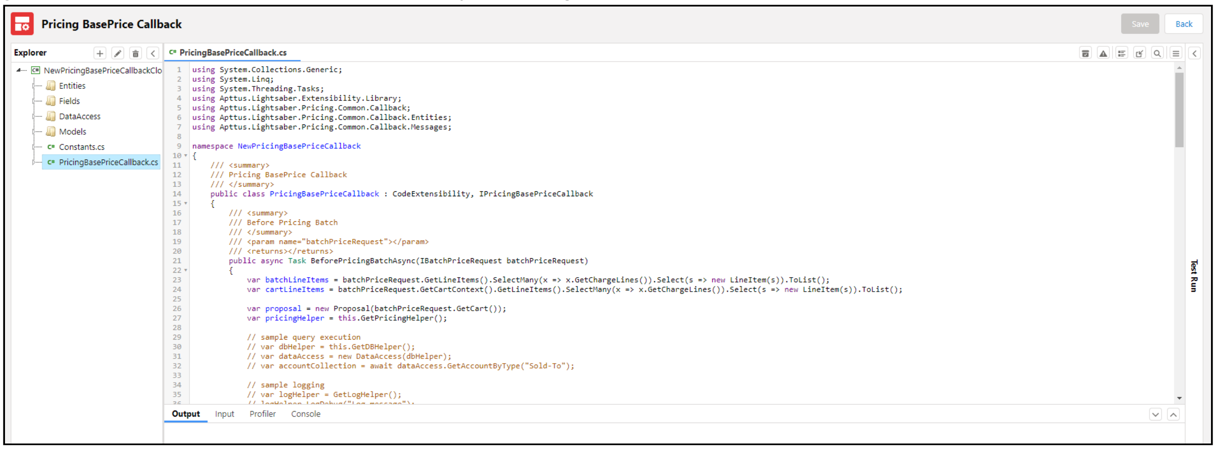
1. From the **Class** drop-down, select a class. It displays all classes available in the project callback.
2. From the **Method** drop-down, select a method. It displays all methods currently available in the class. You can execute a method of that particular class. CPQ does not display private methods on the **Method** drop-down.
3. In the **Parameters** field, enter the code.
4. Click the **Refresh project metadata** icon ( ) to refresh project data. For example, if you have an unsaved method and want to execute it for validation, click
5. the **Refresh project metadata** icon. The new method is listed in the **Method** drop-down.
6. Click the **Execute** icon ( ) to execute the method.

Managing TurboEngines Callbacks

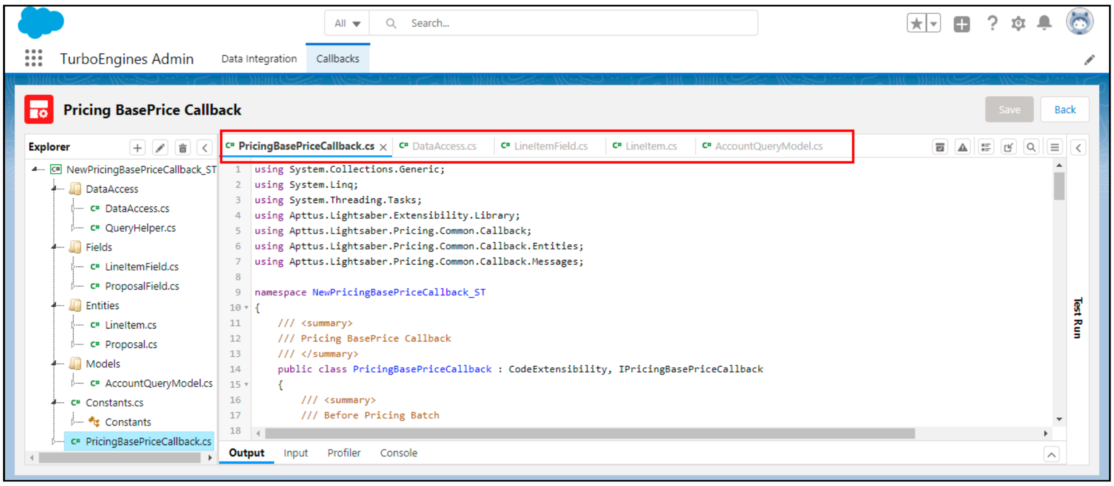
This section provides information on creating and importing callbacks and managing an existing TurboPricing callbacks.

Creating a Callback

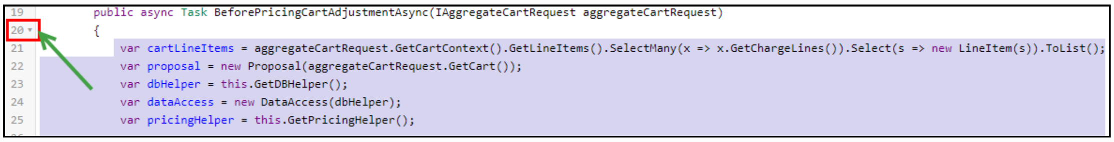
1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click **Create** to create a new project. The New Callback pop-up is displayed.
7. Enter a name for the callback project and click Create. A new project is created and you are redirected to the edit project page.



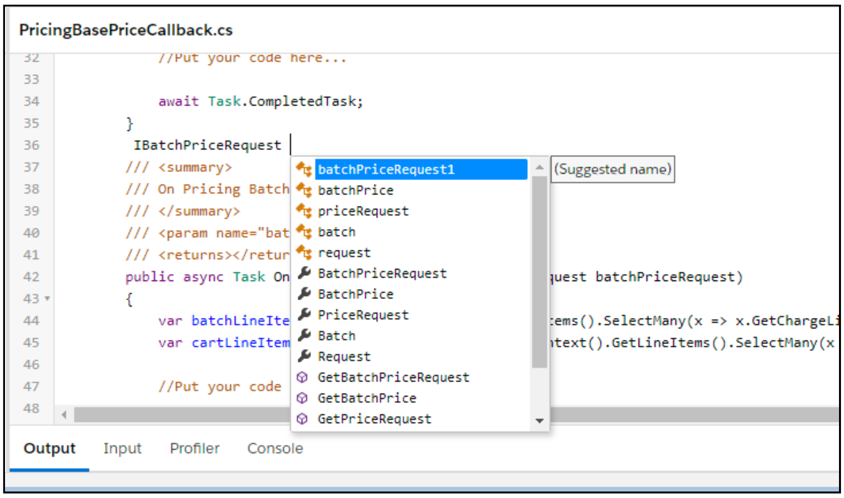
> ⓘ You can open multiple files from the explorer in the code editor.

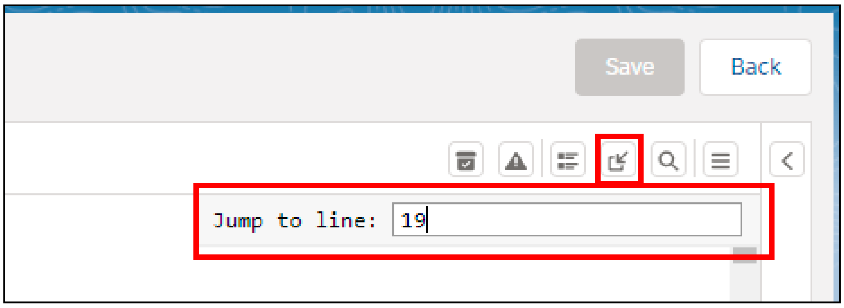
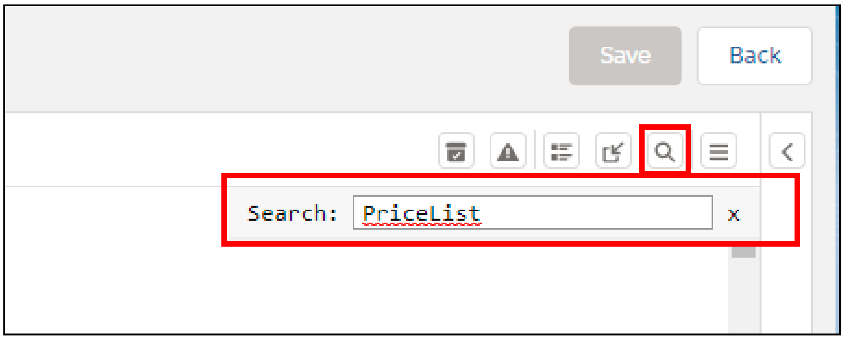


8. You can collapse the code blocks by clicking the **Collapse** icon ().

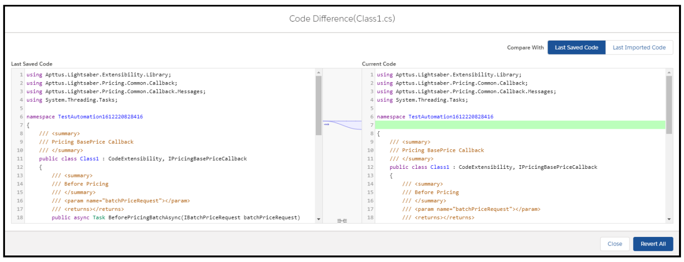


The following table describes the icons displayed on the code editor.

Icon or Feature	Name	Description
Auto Complete	Auto Complete	Press **Ctrl+Space** to view the auto-complete suggestions on the project code (C#) editor. 

Icon or Feature	Name	Description
	Validate QueryModel	To verify if the master entities or fields used in callback code are available in the extensibility data source or not.
	Code Warnings	Displays the code warnings as per the C# guidelines at the bottom.
	Format	Formats and aligns the code (C#) in the editor as per C# coding standards.
	Go To Line (Ctrl+G)	To jump to the specific line in the editor. 
	Find	To search a particular term in the editor. 

<table>
<tr><th>Icon or Feature</th><th>Name</th><th>Description</th></tr>
<tr><td></td><td>Code Difference</td><td>To compare and find the differences between Last Saved Code or Last Imported Code and Current Code.

a. Click the  icon. A Code Difference pop-up window is displayed with details of Last Saved Code and Current Code.


The code differences will be highlighted and can be identified easily.
b. To compare the current code with the last imported code, click Last Imported Code.
c. To retrieve the changes, click Revert All.</td></tr>
</table>

There is a sample template for you to create *IPricingBasePriceCallback*. Below is a basic implementation of *IPricingBasePriceCallback.* Once a project is created, it uses the callback template which has some pre-filled information for your reference. It enables the user to proceed with incorporating a custom logic.

```
using Apttus.Lightsaber.Extensibility.Library;
using Apttus.Lightsaber.Pricing.Common.Callback;
using Apttus.Lightsaber.Pricing.Common.Callback.Messages;
using System.Threading.Tasks;

namespace NewPricingBasePriceCallback_ST
{
    /// <summary>
    /// Pricing BasePrice Callback
    /// </summary>
     public class PricingBasePriceCallback : CodeExtensibility,
IPricingBasePriceCallback
     {
        /// <summary>
        /// Before Pricing
        /// </summary>
```

```
        /// <param name="batchPriceRequest"></param>
        /// <returns></returns>
        public async Task BeforePricingBatchAsync(IBatchPriceRequest
batchPriceRequest)
        {
            await Task.CompletedTask;
        }

        /// <summary>
        /// On Pricing
        /// </summary>
        /// <param name="batchPriceRequest"></param>
        /// <returns></returns>
        public async Task OnPricingBatchAsync(IBatchPriceRequest
batchPriceRequest)
        {
            await Task.CompletedTask;
        }

        /// <summary>
        /// After Pricing
        /// </summary>
        /// <param name="batchPriceRequest"></param>
        /// <returns></returns>
        public async Task AfterPricingBatchAsync(IBatchPriceRequest
batchPriceRequest)
        {
            await Task.CompletedTask;
        }
    }
}
```

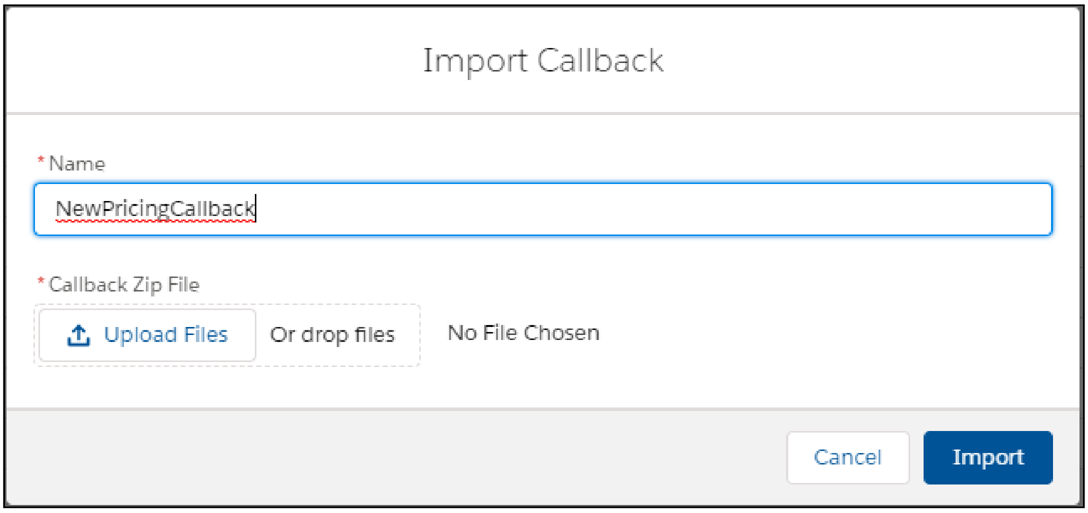
9. Create a callback using the template.
10. Click **Save**. If there are no errors, the code is saved successfully. If there is an error in the code, an error message is displayed. At the bottom of the editor panel, It displays the reason for the error. The user can click the error detail in the output tab and jump to a specific error line in the specific file. It helps the user in compiling the error.

Importing a Callback

The **Import** feature provides a mechanism to import the callback project (*.zip file*). Using this feature, you can move the callback code to a different environment if required.

1. Log in to the Salesforce org.

2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click **Import**. The Import Callback pop-up is displayed.
7. Enter a name for the callback project and click **Upload Files** or **drop files** to add the *.zip* file that you require to import.
8. Click **Import**. A new project is created and you are redirected to the edit project page.



Editing a Callback

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the **Edit** icon () for the required callback. The edit project page is displayed.
7. Perform the required edits to the project. See Creating a TurboPricing Callback.
8. Click **Save**.

Cloning a Callback

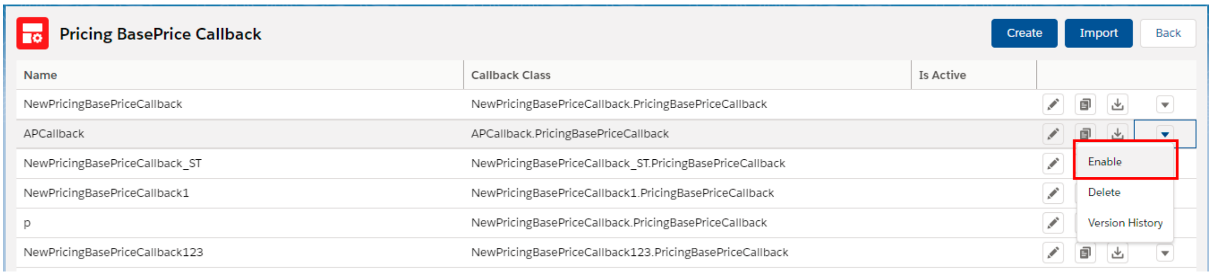
1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the **Clone** icon ( ) for the required callback. The New Callback pop-up is displayed. The name of the callback is <original_callback_name>Clone.
7. Enter a new name if you want.
8. Click **Clone**.

Downloading a Callback

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click the **Download** icon ( ) for the required callback. A <Callback Name>.zip file is downloaded to your system.

Enabling a Callback

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the **Action** icon ( ) for the required callback and select **Enable**. The project is enabled.



ⓘ When you enable a project, CPQ directly impacts pricing engine runtime because of real-time updates.

Disabling a Callback

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the Action icon (for the required callback and select **Disable**. The project is disabled.

Deleting a Callback

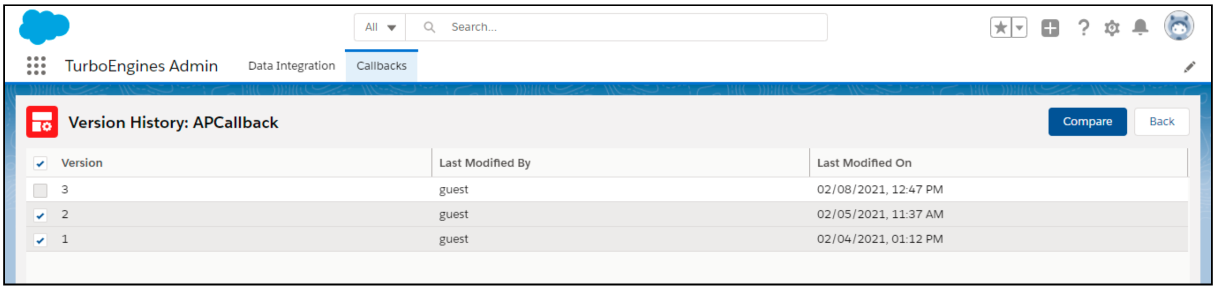
1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the **Action** icon ( ) for the required callback and select **Delete**. The Delete Callback pop-up prompting you to confirm deletion is displayed. Click **Yes** to delete the callback.

> ⓘ You cannot delete an enabled project. First, you need to disable the project and then you can delete it.

Version History

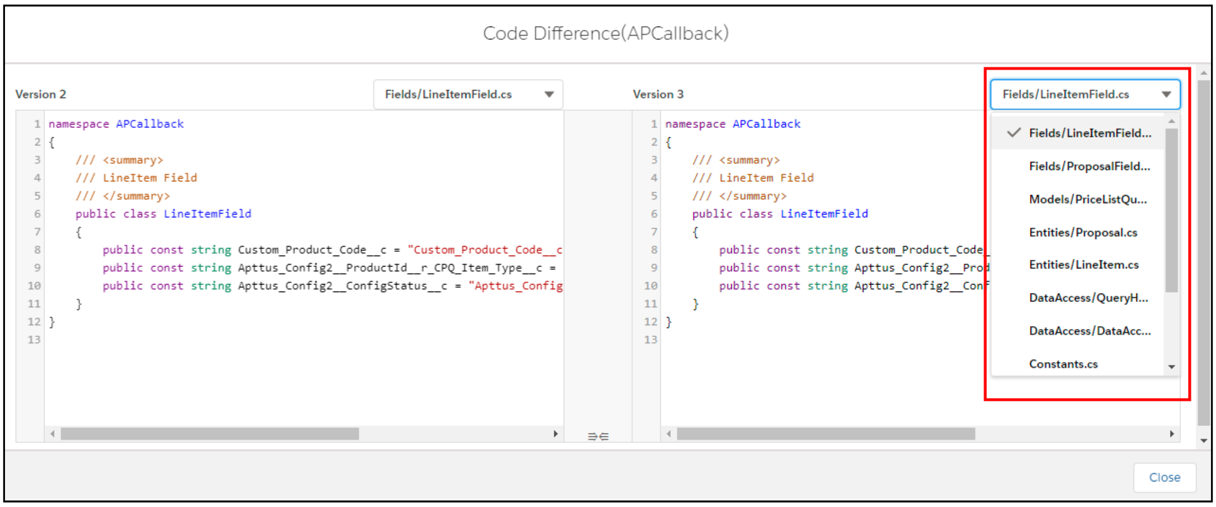
This feature allows the user to compare the different versions in order to identify the differences.

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the **Action** icon () for the required callback and select **Version History.** The detail page of the selected project with a list of versions related to the project is displayed.



7. Click the checkbox of the desired version and click **Compare**. The code difference of the selected callback project is displayed.

8. In the Code Difference popup window, click the dropdown available at the top of the code editor to change the code (*.cs) type.



Validate QueryModel

This feature allows the user to verify if the master entities or fields used in callback code are available in the extensibility data source or not.

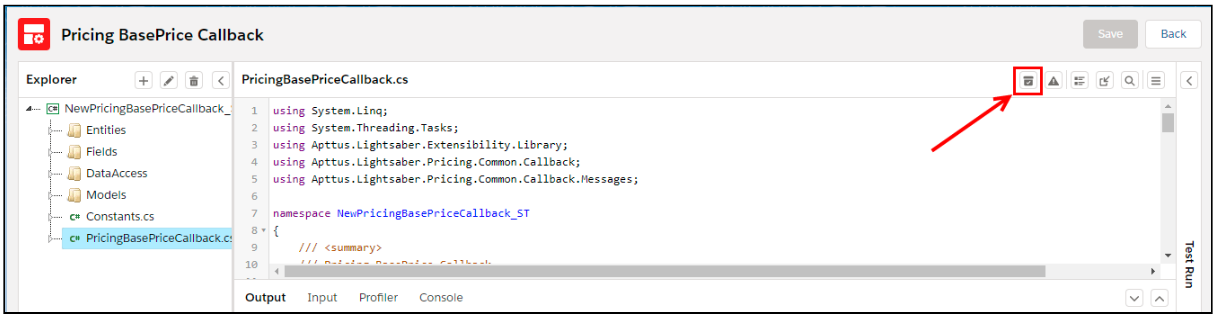
The user should follow such guidelines in order to use this feature accurately.

1. The QueryModel class should be decorated with the attribute [Entity="name of the entity"].
2. All the fields (select clause, where clause etc) used in the query must be specified in QueryModel class with the correct API names".

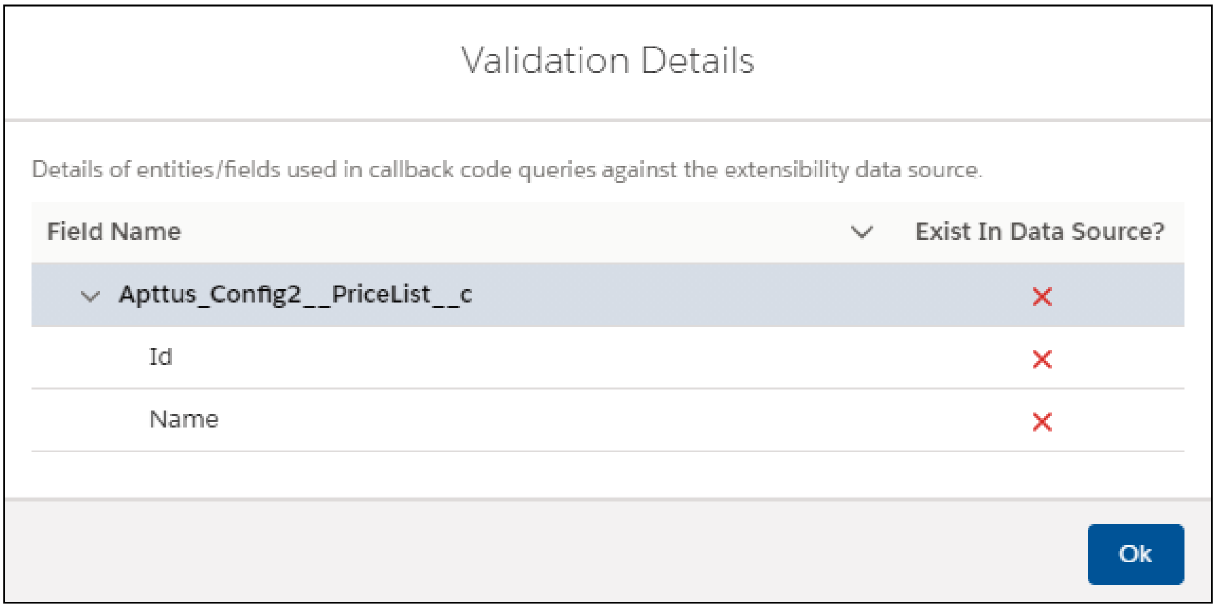
To verify the master entities or fields:

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.

3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed. If a callback is enabled, the project name is displayed in the **Callback Name** column.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click the Edit icon for the required project. You are redirected to the edit project page.



7. Click the **Validate QueryModel** ( ) icon. The Validation Details popup window is displayed.



Helper Functions for TurboEngines Callbacks

This section describes the Helper functions available for TurboPricing callbacks.

- DBHelper
- LogHelper
- PricingHelper
- MetadataHelper

- VaultHelper
- HttpHelper
- CacheHelper

DBHelper

The DBHelper function helps you fetch the required documents from the database based on Query and *FilterConditions*, and fields that you require from the documents that fulfill the conditions. You can get the instance of *IDBHelper* by calling the method *GetDBHelper()* and it provides you with the following methods:

Methods

```
Task<List<T>> FindAsync<T>(Query query);
Task<List<T>> FindAsync<T>(string entityName, Expression<Func<T, bool>>
filterCondition, int? limit, params string[] fields);
Task<List<T>> FindAsync<T>(string entityName, Expression<Func<T, bool>>
filterCondition, params string[] fields);
DBStatisticsInfo GetDBStatistics();
```

The first method helps you fetch the documents based on a *Query*, which comprises of the *EntityName(Name of the collection)*, list of *FilterConditions*, *Fields* to fetch, and *Limit* of results to retrieve.

The *GetDBStatistics()* provides you the information on the total number of queries and the time taken for each query during callback execution.

Note

GetDBStatistics() method should be used only when you are debugging or troubleshooting the issue. Once the debugging/troubleshooting is done, you must remove it from your callback code.

Definitions

```
public class Query
{
    ExpressionOperator ExpressionOperator = ExpressionOperator.AND;
    public string EntityName { get; set; }
    public List<FilterCondition> Conditions { get; set; }
    public Expression Criteria {get; set;}
    public string[] Fields { get; set; }
    public int? Limit { get; set; }
}
```

```
public class FilterCondition
{
    public string FieldName { get; set; }
    public ConditionOperator ComparisonOperator { get; set; }
    public object Value { get; set; }
}

public class Expression
{
    public ExpressionOperator ExpressionOperator { get; set; }
    public List<FilterCondition> Conditions { get; }
    public List<Expression> Filters { get; }
}

ComparisonOperator :
    EqualTo,
    GreaterThan,
    GreaterThanOrEqualTo,
    In,
    LessThan,
    LessThanOrEqualTo,
    NotEqualTo,
    NotIn,
    Contains
```

- *EntityName* is the name of the collection you want to fetch the records from, or the records of the entity that you want to query upon.
- *Conditions* are a list of *FilterConditions*, which compare a field or a column's value with a specified *Value*.
- *Fields* are the columns of the entity you want in your query results.
- *Limit* is the number of records you want to limit your result to.

The following are a few examples of different Query Models: The following are a few examples of executing queries using DBHelper. The examples are for your reference only and may not execute as is.

Simple Query Example

```
Query query = new Query()
{
  EntityName = "Account",
  Fields = new string[]{ "Id", "Name", "Type" },
```

```
  Limit = 5,
  Conditions = new List<FilterCondition>()
  {
    new FilterCondition()
    {
      ComparisonOperator = ConditionOperator.EqualTo,
      FieldName = "Type", Value = "Ship-To"
    }
  }
};
public class AccountQueryModel
{
public string Id {get; set;}
public string Name {get; set;}
public string Type {get; set;}
}
```

DB Call Example

```
var dbHelper = GetDBHelper();
List<AccountQueryModel> account = await dbHelper.FindAsync<AccountQueryModel>(query);
```

Output

After you execute *FindAsync,* it retrieves data from the database and returns a list of AccountQueryModel.

Complex Query Example

```
Query query = new Query()
{
   EntityName = "Account",
   Fields = new string[]{ "Id", "Name", "Type","BillingCity__c",
"BillingCountryCode__c" },
   Limit = 2,
   Criteria = new Expression(ExpressionOperator.AND)
};

FilterCondition nestedConditionOne = new FilterCondition()
{
     ComparisonOperator = ConditionOperator.EqualTo,
     FieldName = "Type",
     Value = "Sold-To"
};
```

```
query.Criteria.AddCondition(nestedConditionOne);

Expression complexExpression = new Expression(ExpressionOperator.OR);
FilterCondition complexExpressionConditionOne = new FilterCondition()
{
     ComparisonOperator = ConditionOperator.In,
     FieldName = "BillingCity__c",
     Value =  new string[] { "BEDFORD PARK", "FREMONT" }
};
FilterCondition complexExpressionConditionTwo = new FilterCondition()
{
     ComparisonOperator = ConditionOperator.EqualTo,
     FieldName = "BillingCountryCode__c",
     Value =  "US"
};
complexExpression.AddCondition(complexExpressionConditionOne);
complexExpression.AddCondition(complexExpressionConditionTwo);
query.Criteria.AddFilter(complexExpression);
```

DB Call Example

```
var dbHelper = GetDBHelper();
List<AccountQueryModel> accountList = await
dbHelper.FindAsync<AccountQueryModel>(query);
return account;
```

Result

After you execute *FindAsync,* it retrieves data from the database and returns a list of AccountQueryModel.

Note:

For better performance, consider the following scenarios when executing database queries.

1. Apply more filters to narrow down the search results.
2. Fetch only the required columns or fields.
3. Do not execute queries inside *for* loop.
4. Create appropriate indexes based on the fields used in the query. You can create indexes on an entity using datasync UI.

LogHelper

The LogHelper function helps you with the tracing or logging. The *ILogHelper* instance is returned when *GetLogHelper()* is called provides you with the following methods.

Methods Available in LogHelper

```
void LogCritical(string message, Exception ex);
void LogDebug(string message);
void LogDebug(object obj);
void LogError(string message, Exception ex);
void LogInformation(string message);
void LogTrace(string message);
void LogWarning(string message);
```

Example

```
var logHelper = GetLogHelper();
logHelper.LogTrace("Trace Statement");
logHelper.LogDebug("Debug Statement");
logHelper.LogWarning("Warning Statement");
logHelper.LogInformation("Info Statement");
AccountQueryModel account = new AccountQueryModel(){
  Id= "0013i0000043m9eAAA",
  Name =  "Account_B5F83C"
};
logHelper.LogDebug(account);
logHelper.LogError("Error Message", new Exception("SystemException"));

public class AccountQueryModel
{
  public string Id {get; set;}
  public string Name {get; set;}
  public string Type {get; set;}
}
```

Console Output

```
Trace Statement
Debug Statement
Warning Statement
Info Statement
{"Id":"0013i0000043m9eAAA","Name":"Account_B5F83C","Type":null}
Error Message
{
```



```
  "ClassName": "System.Exception",
  "Message": "SystemException",
  "Data": null,
  "InnerException": null,
  "HelpURL": null,
  "StackTraceString": null,
  "RemoteStackTraceString": null,
  "RemoteStackIndex": 0,
  "ExceptionMethod": null,
  "HResult": -2146233088,
  "Source": null,
  "WatsonBuckets": null
}
```


When debugging or troubleshooting callback related issues, follow the minimal logging approach and add log statements in the callback code and then remove log statements from the code once it is done.

PricingHelper

The PricingHelper function helps you perform pricing-related operations such as apply rounding and updating the Price Method for line items. To get the instance of *IPricingHelper*, you must call *GetPricingHelper()*, which provides the following methods:

Methods

```
decimal? ApplyRounding(decimal? value, int precision, RoundingMode roundingMode);
void UpdatePrice(ILineItemModel lineItemModel);
public decimal? ApplyRounding(decimal? value, int precision, RoundingMode
roundingMode);
public decimal? ApplyRounding(decimal? value, int? currencyPrecision = null);
public void UpdatePrice(ILineItemModel lineItemModel);
public bool AddPricePointAdjustmentValueToPriceWaterfall(ILineItemModel lineItem,
IPricePointAdjustmentValue newPricePointAdjValue, int? index = null);
public IPricePointAdjustmentValue CreatePricePointAdjustmentValue();
public void RecalculatePriceWaterfall(ILineItemModel lineItem);
public bool RemoveAddedAdjFromPriceWaterfall(ILineItemModel lineItem, bool
recalculate, string pricePoint, PricePointAdjustmentType type, string name, string
sourceId);
public IConfigSystemProperties GetSystemProperties();
public Task<IList<T>> GetRecordsAsync<T>(IQueryObject queryObject);
public IQueryObject CreateQueryObject();
```



```
public T CreateEntity<T>() where T : IBaseEntity;
```

Example

```
var pricingHelper = GetPricingHelper();
var logHelper = GetLogHelper();

var roundedValue = pricingHelper.ApplyRounding(1.657m,2,RoundingMode.DOWN);
logHelper.LogDebug(roundedValue);

roundedValue = pricingHelper.ApplyRounding(1.657m,2,RoundingMode.UP);
logHelper.LogDebug(roundedValue);
```

Console Output

```
1.65
1.66
```

MetadataHelper

The MetadataHelper function helps you fetch the metadata for an entity for a specified RecordType. The *IMetadataHelper* instance returned when *GetMetadataHelper()* is called performs the following method:

Method

```
Task<string> GetRecordTypeIdbyName(string entityName, string recordTypeName);
```

Example

```
var metadataHelper = GetMetadataHelper();
var logHelper = GetLogHelper();

var recordTypeId = await metadataHelper.GetRecordTypeIdbyName("Product2", "Master");
logHelper.LogDebug(recordTypeId);
```

Console Output

```
012000000000000AAA
```

VaultHelper

The VaultHelper function helps you fetch the KeyVault details for an application. The *IVaultHelper* instance returned when *GetVaultHelper()* is called fetches KeyVault values using the following method:

Method

```
Task<Dictionary<string, object>> GetValue(string AppName);
```

Example

```
var vaultHelper = GetVaultHelper();
var logHelper = GetLogHelper();

var vault = await vaultHelper.GetValue("MyApp");
logHelper.LogDebug(vault["AppKey"]);
logHelper.LogDebug(vault["AppSecrets"]);
```

Console Output

```
"Your_Key"
"Your_Secrets"
```

To set and update value in vault

Using the Vault API, you can set and update value in vault, which is stored in encrypted format. You cannot retrieve vault value using any API. You can retrieve vault values only using the VaultHelper function.

1. **Create:** POST: /pricing/api/admin/Vault
   It creates an application with the specified key value name and application name.
   Body:

   ```
   {
       "AppName": "MyApp",
       "Secrets": {
           "MySecretsKey1" : "MyValue1",
           "MySecretsKey2" : "MyValue2"
       }
   }
   ```


You can create multiple applications and store multiple secrets as KeyValue in a single application using the POST API and retrieve those secrete values using the VaultHelper function by passing AppName.

2. **Update:** PUT /pricing/api/admin/Vault/{appName}
   It updates an existing application vault information with a new value.
   Body:

```
{
    "AppName": "MyApp",
    "Secrets": {
        "MySecretsKey1" : "MyNewValue1",
        "MySecretsKey2" : "MyNewValue2"
    }
}
```

3. **Remove:** DELETE /pricing/api/admin/Vault/{appName}

It removes secretes from the vault storage.

HttpHelper

The HttpHelper function helps you with HTTP calls covering CRUD operations.

To get an instance of the *IHttpHelper*, you must call *GetHttpHelper()*, which returns you an instance of *IHttpHelper* allowing you to call the following methods:

> ⓘ It is not recommended to call salesforce API from Custom Code, due to performance reasons.

Methods

```
HttpContent GetHttpContenFromtXml<T>(T payload);
Task<T> GetAsync<T>(string requestUri);
Task<HttpResponseMessage> GetAsync(string requestUri);
Task<T> PostAsync<T>(string requestUri, HttpContent content);
Task<HttpResponseMessage> PostAsync(string requestUri, HttpContent content);
Task<T> PutAsync<T>(string requestUri, HttpContent content);
Task<HttpResponseMessage> PutAsync(string requestUri, HttpContent content);
Task<HttpResponseMessage> DeleteAsync(string requestUri);
```

Example: Invoke GET API call

```
var httpHelper = GetHttpHelper();
var logHelper = GetLogHelper();
```

```
string url = "https://jsonplaceholder.typicode.com/todos/1";
var response = await httpHelper.GetAsync<SampleResponse>(url);
logHelper.LogDebug(response);
```

Console Output

```
Complete-GetAsync(https://jsonplaceholder.typicode.com/todos/1), HttpStatus=OK,
Time=285.534
{"userId":1,"id":1,"title":"delectus aut autem","completed":false}
```

Example: Invoke POST API call

```
var httpHelper = GetHttpHelper();
var logHelper = GetLogHelper();

string url = "https://jsonplaceholder.typicode.com/todos/1";
var response = await httpHelper.GetAsync<SampleResponse>(url);

url = "https://jsonplaceholder.typicode.com/todos";
response.id = 0;
response.title = "Callback";
var contentToPost = new
 StringContent(JsonConvert.SerializeObject(response),Encoding.UTF8,"application/json")
;
var postResult = await httpHelper.PostAsync<SampleResponse>(url,contentToPost);
logHelper.LogDebug(postResult);
```

Console Output

```
Complete-GetAsync(https://jsonplaceholder.typicode.com/todos/1), HttpStatus=OK,
Time=285.534
{"userId":1,"id":1,"title":"delectus aut autem","completed":false}
Complete-PostAsync(https://jsonplaceholder.typicode.com/todos), HttpStatus=Created,
Time=400.073
{"userId":1,"id":201,"title":"Callback","completed":false}
```

CacheHelper

The CacheHelper function helps you cache data for faster retrievals. The *ICacheHelper* instance, which is returned when *GetCacheHelper()* is called, provides you with the following methods:

Methods

```
void Set<T>(string key, T value, string regionName);
T Get<T>(string key, string regionName);
bool Contains(string key, string regionName);
```

Example

```
public class Product2
{
  public string Id {get; set;}
  public string Name { get; set;}
}
```

Set Data in Cache

```
var cacheHelper = GetCacheHelper();

Product2 product = new Product2()
{
  Id= "RT001", Name =  "Router"
};
await cacheHelper.Set<Product2>("Product",product);
```

Console Output

```
{"Id":"RT001","Name":"Router"}
```

Get Data from Cache

```
var cacheHelper = GetCacheHelper();
var logHelper = GetLogHelper();

var productFromCache = await cacheHelper.Get<Product2>("Product");
logHelper.LogDebug(productFromCache);
```

Console Output

```
{"Id":"RT001","Name":"Router"}
```

Pricing Callback Class for TurboPricing

Pricing callback provides extensibility points in the Pricing Engine which can be used to extend or override existing behavior of the Pricing Engine based on customer requirements. Pricing callback classes allow you to add pricing logic to the cart that cannot be achieved by out-of-the-box pricing mechanisms, such as Price Rulesets and Price Matrices.

To use the Pricing Callback you must create a custom C# class that implements the following interfaces

Interface	Description
IPricingBasePriceCallback	This interface provides you a mechanism to define custom logic to be executed before, during, and after Base Price Calculation
IPricingTotallingCallback	This interface provides you a mechanism to define custom logic to be executed before, during, and after adjustment Calculation
IRelatedPricingCallback	This interface provides you a mechanism to define custom logic for calculating the pricing for related product line items

For accessing non-transactional/master data in callback code, you must only use the data synced in the Extensibility consumer profile. You can use the *DBHelper* functions to retrieve the synced master data from the TurboPricing data source. For accessing any transactional data such as line-item custom fields, proposal fields, cart fields, and so on, the line item custom fields should be configured in the custom setting "Configure LineItem Custom Fields" and Proposal/cart fields should be configured in the custom setting "Cart Header Criteria Fields" at Salesforce. Also, the line item's product or option relationship fields used in callback must be configured in the Price List Item entity as a part of the "Pricing Master Data Tables" consumer profile.

The following sections describe the interfaces:

- Coding Guidelines and Best Practices for TurboEngines Callbacks
- Pricing Base Price Callback Interface
- Pricing Totalling Callback Interface
- Related Pricing Callback Interface

- Migrating Salesforce Pricing Callback to TurboEngines Pricing Callback

Coding Guidelines and Best Practices for TurboEngines Callbacks

This section provides guidelines to follow while coding the TurboEngines compatible callbacks.

General Coding Guidelines for TurboEngines Callbacks

Guideline	Sub-topic	Description
Naming Conventions		
	Naming a Class	• Use Pascal casing standard for naming a class. ◦ For example, the class name should be *PricingBasePriceCallback* instead of *pricingBasePriceCallback.*
	Naming a Method	• Use Pascal casing standard for naming a method. ◦ For example, the method name should be *BeforePricingCartAdjustmentAsync* instead of *beforePricingCartAdjustmentAsync.*
	Naming a variable or method parameter	• Use Camel casing standard for naming a variable or method parameter. ◦ For example, the variable name should be cartLineItems instead of CartLineItems. • Use meaningful, descriptive names for naming a variable or method parameter. ◦ For example, the variable name should be cartLineItems instead of crtLnItms
	Naming a class property	• Use Pascal casing standard for naming a class property ◦ For example, the property name should be Id instead of id • Use meaningful, descriptive names for naming the property ◦ For example, the property name should be IsPrimaryLine instead of IsPrmryLn.

Coding Guidelines for TurboEngines Callbacks

You must follow the below guidelines while writing the callback code (C#) for TurboEngines.

> **When you create a new callback project using TurboEngines callback authoring UI, the sample callback project is auto-generated by the TurboEngine Extensibility framework as per the TurboEngine Callback coding guidelines. You can enhance it further with a custom implementation.**

<table>
<tr><th>Guideline</th><th>Description</th></tr>
<tr><td>Using Statements</td><td>

1. When you are writing the callback code:
   a. Add all the required using statements for the callback code.
   b. Below are the supported using statements in TurboEngine C# callback code:
      i. .NET core framework provided using statements
      ii. Newtonsoft.json provided using statements
      iii. TurboEngine Extensibility Framework provided using statements.
   c. For your reference, TurboEngine Extensibility Framework provides using statements are as follows. It would help if you used the required using statements.

```
using Apttus.Lightsaber.Pricing.Common.Callback;
using Apttus.Lightsaber.Pricing.Common.Callback.Entities;
using Apttus.Lightsaber.Pricing.Common.Callback.Messages;
using Apttus.Lightsaber.Pricing.Common.Callback.Enums;
using Apttus.Lightsaber.Pricing.Common.Callback.Models;
using Apttus.Lightsaber.Extensibility.Library;
using Apttus.Lightsaber.Extensibility.Library.Attributes;
using Apttus.Lightsaber.Extensibility.Library.Extension;
using Apttus.Lightsaber.Extensibility.Library.Interface;
using Apttus.Lightsaber.Extensibility.Library.Model;
```

</td></tr>
<tr><td>Namespace naming</td><td>

1. Namespace's name must have the following convention:
   Apttus.Lightsaber.{OrganizationName}.{OtherDetails}
   a. For example, when writing a pricing callback implementation for Totaling callback for organization X, write the namespace as follows
      Apttus.Lightsaber.X.Totalling

</td></tr>
</table>

<table>
<tr><th>Guideline</th><th>Description</th></tr>
<tr><td>Class Naming</td><td>

1. The main class implementing the callback interface must always inherit from **CodeExtensibility**.
2. The naming of the main class must be based on the interface name. So, remove "I" prefix from the interface name.
    a. For example, pricing callback implementation for Totaling callback must implement *IPricingTotallingCallback.* Hence, the class name must be *PricingTotallingCallback.*

</td></tr>
<tr><td>Methods</td><td>

1. Methods should follow async/await pattern implementation.
    a. Following is an example method representation without async/ await implementation.

```
public void MethodName() {
    return;
}
```

    b. Following is an example method implementation with async/await implementation.

```
public async Task MethodName() {
    await Task.CompletedTask;
}
```

</td></tr>
<tr><td>Variables</td><td>

- For better precision, use decimal datatype instead of double datatype.

</td></tr>
</table>

Guideline	Description
Database queries related	• All the database query invocation must be part of the class named **DataAccess** under the folder DataAccess. • Ensure that all the DB queries for fetching the master data records from TurboEngine Extensibility data source (For example, MongoDB) are written in **QueryHelper** under the DataAccess folder. • This provides a reference and a single place to identify all the queries used by a callback implementation. • The respective model class needs to be created as follows to hold the result/response of the query execution for each of the queries executed. When you create a new callback project, a sample model class is already provided in the auto-generated callback project. • Create all the model classes under the **Models** folder. • The name of the class must end with QueryModel. i. For example, the model class to hold the result/response of Account query execution should be named as AccountQueryModel. • Ensure that the field names of QueryModel match the field names used in the query. • Ensure that the QueryModel class is annotated with the Entity attribute. The entity name should be spelled the same as it is in Salesforce. i. For example, The AccountQueryModel class should have the Entity attribute added as **[Entity("Account")]**. Here Account is the entity name as available in Salesforce.
Constants	• Ensure that all the general-purpose constants used in the callback implementation are specified in the Constants class.
Custom Fields	• The classes under the Fields folder must hold the field name constants for any custom fields used in the callback implementation. These class names must end with Field. • For example, if the class to hold the proposal object's custom field must be ProposalField. • Generally, you are required to create such classes for the line item, proposal, and cart object fields.

Guideline	Description
LineItem and Proposal entity model objects	1. To easily access LineItem and Proposal fields for getting or setting the values in the callback implementation, LineItem and Proposal classes are auto-created under the Entities folder by the TurboEngine Extensibility framework. a. You must use those classes in your callback implementation and add the custom field details to your implementation. b. When adding custom field details to LineItem or Proposal class, i. provide the relationship field as shown in the following example. For example, If your relationship field is **Apttus_Config2__ProductId__r.APTS_field__c** then the field name is **Apttus_Config2__ProductId__r_APTS_field__c.** Here "." is replaced with "_" in the name. ii. In getter methods for the fields, use *GetLookupValue<T>()* method for relationship fields and use *GetValue<T>()* method for non-relationship or direct fields.
Extensibility Framework Helpers	For detailed Extensibility Framework Helpers information, refer to the Helper Functions for TurboEngines Callbacks topic.

Best Practices for TurboEngines Callback code

Consider the following practices while writing the TurboEngines compatible C# callback code.

Whenever you change specific fields within the callback code and send them to the TurboPricing for pricing, It only verifies the change in Net Price and does not verify each field to detect the difference. Hence, if you make any change in callback code that impacts pricing, perform the following.

TurboEngine uses a digest mechanism to detect if there are any changes to the line items. At the end of every pricing cycle, it recomputes the digest and compares it against the old digest. If they did not match, you could consider that the line item is changed. However, TurboEngines uses only a few selected fields (including NetPrice) to compute the digest but no custom field. Therefore, when the callback code makes any important change on a line item or custom field that TurboEngines might not detect, you must reset digest for that line item to avoid matching the new one. *For example, use* the following code to reset the digest for a line item with type *ILineItemModel*:

```
lineItem.Set("Digest", string.Empty);
```

Below is the sample pattern to be followed:

```
List<LineItem> lineItemsToResetDigest = new List<LineItem>();
foreach(var lineItem in allLineItems){
  // business logic that changes line item's field
  // add all such lines to above list
  lineItemsToResetDigest.Add(lineItem);
}

// reset the digest at the end in bulk
ResetDigest(lineItemsToResetDigest);
```

General Practices

1. Use class-level variables only when necessary and try to use the local variables as much as possible.
2. If you need a class-level variable, use *BeforePricingBatchAsync* (Base price callback) or *BeforePricingCartAdjustmentAsync* (Totalling callback) methods to initialize such class level variables.
3. Avoid using nested for loops.
4. Do not use **Try** or **Catch** in the callback code to explicitly log any unhandled exception. By default, an unhandled exception is logged by the TurboEngine Extensibility framework. However, you can also configure to display the unhandled exception on UI using the “Is Callback Exception Throw Enabled” (is-callback-exception-throw-enabled) feature flag.
5. For using Extensibility framework helpers (such as DBHelper, LogHelper, and so on) in your callback project, you must create a helper object in the callback implementation class (the main class of callback project which is implementing the callback interface). If you want to access helpers in any other supported class of your callback project, you must pass the helper instances from the main implementation class to such classes.
6. You must not use the PricingHelper's UpdatePrce() method on line items unless it is necessary. Therefore, design a callback implementation that can avoid such a method call.
7. Do not use concrete objects to set as a field value on the line item. In case if you want to set a concrete object as a value to the line items field, then stringify it before setting it on the line item field.

8. From TurboEngine callback implementation, do not make an SFDC API call using HttpHelper in any scenario. If such a scenario needs to be handled, get in touch with TurboEngine's product or technical team.
9. In Base price callback, always use batch line items for updating any value on line item as per your custom logic. Ensure that you do not update anything on cart line items when you are in the Base price callback.
10. Always use the async or await pattern for each method.
11. To access/update any field in callback code, the field API name/object API name must be spelled the same in Salesforce.
12. Keep your code formatted for easy readability. Callback authoring UI provides code formatting capability to format the code as per C# standards.
13. Keep your code to have 0 or minimum code warnings. Callback authoring UI provides the capability that shows you the code warnings per the C# standards for your callback project. Ensure that you have fixed them and incorporated them in your code to have 0 or minimum code warnings.

Pricing Base Price Callback Interface

The *IPricingBasePriceCallback* interface provides you a mechanism to define a custom logic to be executed before, during, and after **Base Price** calculation. The *IPricingBasePriceCallback* is executed in batches which consists of the batch line items. TurboPricing calls this interface for every batch.

The following methods are available in the *IPricingBasePriceCallback* interface:

Method Signature	Description
Task BeforePricingBatchAsync(IBatchPriceRequest batchPriceRequest)	You can use this method to define custom logic that must be executed before Base Price is calculated.
Task OnPricingBatchAsync(IBatchPriceRequest batchPriceRequest)	You can use this method to define custom logic that must be executed during the Base Price calculation. You can use the price list items to write the custom logic.
Task AfterPricingBatchAsync(IBatchPriceRequest batchPriceRequest)	You can use this method to define custom logic that must be executed after the Base Price is calculated.

Method Signature	Description
Task OnProductOptionPriceAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IProductOptionPrice> productOptionPrice)	You can use this extension point to modify product option prices for the given line item.

Extension Points

The following extension points are added to the default template.

Method Signature	Description

Method Signature	Description
Task OnPriceMatrixAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceMatrixEntry>> priceMatrixEntries)	You can use this extension point to modify resolved price matrix for the given line item if required. The below code snippet is for reference purposes only.

```
/// <summary>
/// On Price Matrix
/// </summary>
/// <param name="batchPriceRequest"></param>
/// <param name="priceMatrixEntries"></param>
/// <returns></returns>
public async Task
OnPriceMatrixAsync(IBatchPriceRequest
batchPriceRequest, IDictionary<string,
IEnumerable<IPriceMatrixEntry>>
priceMatrixEntries)
{
    var batchLineItems =
batchPriceRequest.GetLineItems().SelectMany(x
=> x.GetChargeLines()).Select(s => new
 LineItem(s)).ToList();
    var cartLineItems =
batchPriceRequest.GetCartContext().GetLineIte
ms().SelectMany(x =>
x.GetChargeLines()).Select(s => new
 LineItem(s)).ToList();

    var batchLineItem = batchLineItems[0];
    if(priceMatrixEntries.ContainsKey(batchLi
neItem.Id))
    {
        var priceMatrixEntriesEnumerable =
priceMatrixEntries[batchLineItem.Id];
        foreach(var pMatrixEntry in
priceMatrixEntriesEnumerable)
        {
            pMatrixEntry.AdjustmentAmount =
20;
        }
    }
```

Method Signature	Description
	`    await Task.CompletedTask;` `}`

<table>
<tr><th>Method Signature</th><th>Description</th></tr>
<tr><td>Task OnPriceRuleAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceRuleEntry>> priceRuleEntries)</td><td>You can use this extension point to modify the resolved price rule entries for the given line item if required.

The below code snippet is for reference purposes only.

<pre>/// <summary>
/// On PriceRule
/// </summary>
/// <param name="batchPriceRequest"></param>
/// <param name="priceRuleEntries"></param>
/// <returns></returns>
public async Task
OnPriceRuleAsync(IBatchPriceRequest
batchPriceRequest, IDictionary<string,
IEnumerable<IPriceRuleEntry>>
priceRuleEntries)
{
    var batchLineItems =
batchPriceRequest.GetLineItems().SelectMany(x
=> x.GetChargeLines()).Select(s => new
 LineItem(s)).ToList();
    var cartLineItems =
batchPriceRequest.GetCartContext().GetLineIte
ms().SelectMany(x =>
x.GetChargeLines()).Select(s => new
 LineItem(s)).ToList();

    var batchLineItem = batchLineItems[0];
    if(priceRuleEntries.ContainsKey(batchLine
Item.Id))
    {
        var priceRuleEntriesEnumerable =
priceRuleEntries[batchLineItem.Id];
        foreach(var pRuleEntry in
priceRuleEntriesEnumerable)
        {
            pRuleEntry.AdjustmentAmount = 20;
        }
    }
  await Task.CompletedTask;</pre></td></tr>
</table>

Method Signature	Description
	`` } ```

Method Signature	Description
Task OnPipelinePriceRuleAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceRuleEntry>> pipelinePriceRuleEntries)	You can use this extension point to modify the resolved price pipeline rule entries for the given line item if required The below code snippet is for reference purposes only. `/// <summary>` `/// On Pipeline PriceRule` `/// </summary>` `/// <param name="batchPriceRequest"></param>` `/// <param name="pipelinePriceRuleEntries"></param>` `/// <returns></returns>` `public async Task OnPipelinePriceRuleAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceRuleEntry>> pipelinePriceRuleEntries)` `{` `    var batchLineItems = batchPriceRequest.GetLineItems().SelectMany(x => x.GetChargeLines()).Select(s => new LineItem(s)).ToList();` `    var cartLineItems = batchPriceRequest.GetCartContext().GetLineItems().SelectMany(x => x.GetChargeLines()).Select(s => new LineItem(s)).ToList();` `    var batchLineItem = batchLineItems[0];` `    if(pipelinePriceRuleEntries.ContainsKey(batchLineItem.Id))` `    {` `        var pipelinePriceRuleEntriesEnumerable = pipelinePriceRuleEntries[batchLineItem.Id];` `        foreach(var ppRuleEntry in pipelinePriceRuleEntriesEnumerable)` `        {` `            ppRuleEntry.AdjustmentAmount = 20;`

Method Signature	Description
	`        }` `    }` `    await Task.CompletedTask;` `}`
Task OnPriceEscalatorAsync(IBatchPriceRequest batchPriceRequest, List<IPriceEscalator> priceEscalators)	You can use this extension point to modify the resolved price escalators if required.

Example Code

The below code snippet is for reference purposes only.

```
namespace Apttus.Lightsaber.Customer.Pricing
{
    public class PricingBasePriceCallback : CodeExtensibility,
IPricingBasePriceCallback
    {
        public async Task AfterPricingBatchAsync(IBatchPriceRequest
batchPriceRequest)
        {
            await Task.CompletedTask;
        }

        public async Task BeforePricingBatchAsync(IBatchPriceRequest
batchPriceRequest)
        {
            var batchLineItems = batchPriceRequest.GetLineItems().SelectMany(x =>
x.GetChargeLines()).Select(s => new LineItem(s)).ToList();

            //Example, setting custom field on line item before PLI resolution
            foreach(var batchLineItem in batchLineItems) {
                decimal extendedQuantity = batchLineItem.GetQuantity();
                decimal quantity = batchLineItem.GetQuantity();

                if(batchLineItem.IsOptionLine()) {
                    LineItem rootBundleLineItemModel = new
 LineItem(batchLineItem.GetRootParentLineItem().GetPrimaryLineItem());
```

```
                    decimal bundleQuantity = rootBundleLineItemModel.GetQuantity();
                    extendedQuantity = bundleQuantity * batchLineItem.GetQuantity();
                }

                batchLineItem.APTS_Extended_Quantity__c = extendedQuantity;
            }

            //You can also query DB here, and perform initial setup such as creating
required Dictionary, List and so on for later use in the callback code.

            await Task.CompletedTask;
        }

        public async Task OnPricingBatchAsync(IBatchPriceRequest batchPriceRequest)
        {
            var batchLineItems = batchPriceRequest.GetLineItems().SelectMany(x =>
x.GetChargeLines()).Select(s => new LineItem(s)).ToList();

            foreach(var batchLineItem in batchLineItems) {
                IPriceListItemModel priceListItemModel =
batchLineItem.GetPriceListItem();
                IPriceListItem priceListItemEntity = priceListItemModel.GetEntity();

                if(batchLineItem.PriceListId != priceListItemEntity.PriceListId) {
                       batchLineItem.APTS_Is_Contract_Pricing__c = true;
                }
            }
            await Task.CompletedTask;
        }
    }
}
```

Pricing Totalling Callback Interface

The *IPricingTotallingCallback* interface provides you a mechanism to defined custom logic to be executed before, during, and after adjustment calculation.
The *IPricingTotallingCallback* is invoked once for each pricing request to calculate the total.

The following methods are available in the *IPricingTotallingCallback* interface:

Method Signature	Description
Task BeforePricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)	You can use this method to define custom logic that must be executed before the adjustment is calculated.
Task AfterPricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)	You can use this method to define custom logic that must be executed after the adjustment is calculated.
Task OnCartPricingCompleteAsync(IAggregateCartRequest aggregateCartRequest)	You can use this method to define custom logic that must be executed after the pricing is calculated completely.

Example Code

The below code snippet is for reference purposes only.

```
namespace Apttus.Lightsaber.Customer.Totaling
{
    public class PricingTotallingCallback : CodeExtensibility,
IPricingTotallingCallback
    {
        public async Task BeforePricingCartAdjustmentAsync(IAggregateCartRequest
aggregateCartRequest)
        {
            var cartLineItems =
aggregateCartRequest.GetCartContext().GetLineItems().SelectMany(x =>
x.GetChargeLines()).Select(s => new LineItem(s)).ToList();

            foreach(LineItem cartLineItem in cartLineItems) {
                if(cartLineItem.IncentiveBasePrice.HasValue &&
cartLineItem.IncentiveBasePrice.Value != 0) {

                    decimal sellingTerm =
cartLineItem.GetValuetOrDefault(LineItemPropertyNames.SellingTerm, 1);
                    decimal lineItemQ = cartLineItem.GetQuantity();

                    decimal? unitIncentiveAmount = cartLineItem.BasePrice -
cartLineItem.IncentiveBasePrice;
```



```
                    cartLineItem.APTS_Unit_Incentive_Adjustment_Amount__c =
unitIncentiveAmount;
                    cartLineItem.IncentiveBasePrice = cartLineItem.BasePrice -
unitIncentiveAmount;
                    cartLineItem.IncentiveAdjustmentAmount = unitIncentiveAmount *
lineItemQ * sellingTerm * -1;
                }
            }

            await Task.CompletedTask;
        }

        public async Task AfterPricingCartAdjustmentAsync(IAggregateCartRequest
aggregateCartRequest)
        {
            //Example, Set custom fields on line item based on the adjusted price
            await Task.CompletedTask;
        }

        public async Task OnCartPricingCompleteAsync(IAggregateCartRequest
aggregateCartRequest)
        {
            var cartLineItems =
aggregateCartRequest.GetCartContext().GetLineItems().SelectMany(x =>
x.GetChargeLines()).Select(s => new LineItem(s)).ToList();
            foreach(var cartLineItem in cartLineItems) {
                if(cartLineItem.NetPrice < 1000) {
                    cartLineItem.APTS_Deal_Color = "Red";
                } else {
                    cartLineItem.APTS_Deal_Color = "Green";
                }
            }
            await Task.CompletedTask;
        }
    }
}
```

Pricing Baseprice Callback Interface Parameters

Batch Price Request Interface

Below is the interface used by the BasePrice callback.

```
/// <summary>
    /// Cart Price Request by Batch Object
```

```
    /// </summary>
    public interface IBatchPriceRequest : ICartBaseRequest
    {
        /// <summary>
        /// Unique batch ID
        /// </summary>
        public string BatchId { get; }

        /// <summary>
        /// Batch name
        /// </summary>
        public string Name { get; }

        /// <summary>
        /// List of lines present in this batch to be priced
        /// </summary>
        public List<IProductLineItemModel> GetLineItems();

        /// <summary>
        /// List of related lines or parent bundle lines present in this batch to be
priced
        /// </summary>
        public List<IProductLineItemModel> GetReferenceLines();

        /// <summary>
        /// Cart context.
        /// </summary>
        public ICartContext GetCartContext();

        /// <summary>
        /// Set error details for all lines in this batch. This also updates pricing
status to error
        /// </summary>
        /// <param name="errorDetails"></param>
        public void SetErrorDetails(string errorDetails, string lineItemId = null);

        /// <summary>
        /// Set error details for all lines in this batch. This also updates pricing
status to error
        /// </summary>
        /// <param name="ex"></param>
        public void SetErrorDetails(Exception ex, string lineItemId = null);
```

```
        /// <summary>
        /// List  the product lines that have thrown no errors so far during 
processing.
        /// Errors could be due to validation checks, invalid data setup or failure 
of dependent systems as well
        /// </summary>
        /// <returns></returns>
        public IEnumerable<IProductLineItemModel> GetValidProductLineItems();

        /// <summary>
        /// Gets the product line object using line number.
        /// Option Line can use it to get RootBundleLine LineItemModel.
        /// </summary>
        /// <param name="lineNumber">Line number</param>
        /// <returns></returns>
        public IProductLineItemModel GetRootParentBundleLine(int lineNumber);

        /// <summary>
        /// Gets the product line object using parent bundler number.
        /// Option Line can use it to get ParentBundleLine LineItemModel.
        /// </summary>
        /// <param name="parentBundleNumber">Parent bundle number</param>
        /// <returns></returns>
        public IProductLineItemModel GetParentBundleLine(int parentBundleNumber);

        /// <summary>
        /// Get Realted Line Items
        /// </summary>
        /// <returns></returns>
        public IEnumerable<ILineItemModel> GetRelatedLineItems();

    }
```

Method	Data Type	Description
BatchId	String	Provides the unique batch ID.
Name	String	Provides the batch name.
GetLineItems();	List<IProductLineItemModel>	Provides the details of lines to be priced in the batch.

Method	Data Type	Description
GetReferenceLines()	List<IProductLineItemModel>	Cart Price Request by Batch object provides the details of parent lineitems.
GetCartContext()	ICartContext	Provides the context of the cart.
SetErrorDetails(string errorDetails, string lineItemId = null);	Void	Sets error details for all lines in this batch and updates the pricing status to *error.*
SetErrorDetails(Exception ex, string lineItemId = null)	Void	Set error details for all lines in this batch and update the pricing status to *error.*
GetValidProductLineItems();	IEnumerable<IProductLineItemModel>	Provides all the line items which are not marked as error or valid lineitems in pricing.
GetRootParentBundleLine(int lineNumber);	IProductLineItemModel	Gets the root parent bundle for option lines.
GetParentBundleLine(int parentBundleNumber)	IProductLineItemModel	Gets the parent bundle line for options.
GetRelatedLineItems()	IEnumerable<ILineItemModel>	Provides the related line items

Pricing Totalling Callback Parameters

Aggregate Cart Request Interface

Below is the definition for the IAggregateCartRequest interface used in the Pricing Totalling callback.

```
public interface IAggregateCartRequest : ICartBaseRequest
    {
        /// <summary>
```

```
        /// Get Cart Context
        /// </summary>
        /// <returns></returns>
        public ICartContext GetCartContext ();

        /// <summary>
        /// Request Id
        /// </summary>
        public string RequestId { get; }
    }
```

Method	Description
ICartBaseRequest	This is the base Interface which *IAggregatecartrequest* inherits.
GetCartContext	This method provides the context of the cart.
RequestId	This property provides the request ID.

Cart Base Request Interface

Below is the definition of the *ICartBaseRequest* interface which *IAggregateCartRequest* inherits.

```
public interface ICartBaseRequest : ICallbackBaseRequest
    {
        /// <summary>
        /// Cart Id
        /// </summary>
        public string CartId { get; }

        /// <summary>
        /// Cart Object
        /// </summary>
        public IProductConfigurationModel GetCart();
    }
```

Method	Description
GetCart	This method provides the details of the cart.
CartId	This property provides the value of the cart id.

Cart Context

Below are the ICartContext interface details which is used by *GetCartContext()* method in *IAggregateCartRequest* interface.

```
public interface ICartContext
    {
        /// <summary>
        /// Cart header model
        /// </summary>
        public IProductConfigurationModel GetCart();

        /// <summary>
        /// All product lines present in the cart
        /// </summary>
        public List<IProductLineItemModel> GetLineItems();

        /// <summary>
        /// Get Summary Groups
        /// </summary>
        List<ISummaryGroup> GetSummaryGroups();
    }
```

Method	Description
GetCart	This method provides the details of the cart.
GetLineItems	This method provides the data about the lineitems inside the cart.
GetSummaryGroups	This method provides the details of all summary groups present in the cart.

IProductConfiguration

IProductConfigurationModel interface is used by GetCart() method in ICartBaseRequest interface.

```
Public interface IProductConfigurationModel : IBaseModel<IProductConfiguration>
    {

    }
```

```
IProductConfigurationModel  is inheriting IProductConfiguration so below is the
definition of IProductConfiguration.

public interface IProductConfiguration : IBaseEntity
    {
        /// <summary>
        /// Sold to Account Id.
        /// </summary>
        public string AccountId { get; set; }

        /// <summary>
        /// Price List Id.
        /// </summary>
        public string PriceListId { get; set; }

        /// <summary>
        /// Cart Approval Preview Status (pending or complete)
        /// </summary>
        public string ApprovalPreviewStatus { get; set; }

        /// <summary>
        /// Cart Approval Status
        /// </summary>
        public string ApprovalStatus { get; set; }

        /// <summary>
        /// Billing Preference Id
        /// </summary>
        public string BillingPreferenceId { get; set; }

        /// <summary>
        /// Pricing Date for the Cart.
        /// </summary>
        public DateTime? PricingDate { get; set; }

        /// <summary>
        /// Start date for the Cart.
        /// </summary>
        public DateTime? ExpectedStartDate { get; set; }

        /// <summary>
        /// End date for the Cart.
        /// </summary>
        public DateTime? ExpectedEndDate { get; set; }
```

```
        /// <summary>
        /// Contract Numbers for querying Price Lists.
        /// </summary>
        public string ContractNumbers { get; set; }

        /// <summary>
        /// CouponCodes
        /// </summary>
        public string CouponCodes { get; set; }

        /// <summary>
        /// Contains the Effective Price List Id (Quote PriceList)
        /// </summary>
        public string EffectivePriceListId { get; set; }

        /// <summary>
        /// To identify cart business object type
        /// </summary>
        public string BusinessObjectType { get; set; }

        /// <summary>
        /// Cart Status
        /// </summary>
        public string Status { get; set; }
    }
```

Method	Description
AccountId	This property provides the details of an account Id against the cart that has been created.
PriceListId	This property contains the details of a Price List id.
ApprovalPreviewStatus	This property contains the details of the status of a cart approval (pending or complete).
ApprovalStatus	This property contains the details of the status of the cart approval.
BillingPreferenceId	This property contains the details of the billing preference ID.

Method	Description
PricingDate	This property contains the details of the date and time of the pricing for the cart.
ExpectedStartDate	This property contains the details of the start date and time of the cart.
ExpectedEndDate	This property contains the details of the end date and time of the cart.
ContractNumbers	This property contains the details of the contact numbers for querying price lists.
CouponCodes	This property contains the details of the coupon codes applied on the cart.
EffectivePriceListId	This property contains the details of the effective price list id of the quote price list.
BusinessObjectType	This property contains the details of the identified business object type of the cart.
Status	This property contains the details of the cart status.

Related Pricing Callback Interface

The interface provides you the capability to define custom logic for calculating the pricing for related product line items. The prices in the related product line items are dependant on the price of other line items.

The following method is available in the *IRelatedPricingCallback* interface:

Method Signature	Description
Task<List<IRelatedPricingBatchResponse>> ComputeBasePriceBatchAsync(IRelatedPricingBatchRequest relatedPricingBatchRequest);	You can use this method to define custom logic that must be executed to specify a new base price for the related line item.

Example Code

The below code snippet is for reference purposes only.

```
namespace Apttus.Lightsaber.Customer.RelatedPricing
{
```

```
    public class RelatedPricingCallback : CodeExtensibility, IRelatedPricingCallback
    {
        public async Task<List<IRelatedPricingBatchResponse>>
ComputeBasePriceBatchAsync(IRelatedPricingBatchRequest relatedPricingBatchRequest)
        {
            List<IRelatedPricingBatchResponse> relatedPricingBatchResponseResult =
new List<IRelatedPricingBatchResponse>();

            List<ILineItemModel> relatedLineItems =
relatedPricingBatchRequest.GetRelatedLineItems();

            foreach(var relatedLineItem in relatedLineItems) {

                var priceBreakupRecords = relatedLineItem.GetPriceBreakupRecords();

                foreach(var priceBreakup in priceBreakupRecords) {
                    if(priceBreakup.BreakupType == "Total" &&
relatedLineItem.GetEntity().RelatedAdjustmentAmount > 5000) {
                        IRelatedPricingBatchResponse relatedPricingResponse =
relatedPricingBatchRequest.CreateRelatedPricingBatchResponse(relatedLineItem);
                        relatedPricingResponse.BasePrice = 500;

relatedPricingBatchResponseResult.Add(relatedPricingResponse);
                    }
                }
            }
            return await Task.FromResult(relatedPricingBatchResponseResult);
        }
    }
}
```

Migrating Salesforce Pricing Callback to TurboEngines Pricing Callback

This section provides high-level details on mapping the pricing callback from Salesforce to TurboEngines.

Mapping pricing callback from Salesforce to TurboEngines

This section summarizes the mapping process from the Salesforce pricing callback method to TurboEngines pricing callback methods. The 'Mode' plays an important role in the mapping process as the mode Salesforce pricing callbacks are now available in dedicated interfaces in TurboEngine pricing callback. *IPricingBasePriceCallback* for BASEPRICE mode and *IPricingTotallingCallback* for ADJUSTMENT mode.

The following table summarizes the **BASEPRICE** mode methods of Salesforce pricing callback.

BASEPRICE Mode in Salesforce	IPricingBasePriceCallback in TurboEngine
void start(ProductConfiguration cart);	Task BeforePricingBatchAsync(IBatchPriceRequest batchPriceRequest)
void setMode(PricingMode mode);	Not Applicable
void beforePricing(ProductConfiguration.LineItemColl itemColl);	Task BeforePricingBatchAsync(IBatchPriceRequest batchPriceRequest)
void onPriceItemSet(PriceListItem__c itemSO, LineItem lineItemMO);	Task OnPricingBatchAsync(IBatchPriceRequest batchPriceRequest)
void afterPricing(ProductConfiguration.LineItemColl itemColl);	Task AfterPricingBatchAsync(IBatchPriceRequest batchPriceRequest)
void finish();	Task AfterPricingBatchAsync(IBatchPriceRequest batchPriceRequest)
	Task OnProductOptionPriceAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IProductOptionPrice> productOptionPrice)
	Task OnPriceMatrixAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceMatrixEntry>> priceMatrixEntries)
	Task OnPriceRuleAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceRuleEntry>> priceRuleEntries)

BASEPRICE Mode in Salesforce	IPricingBasePriceCallback in TurboEngine
	Task OnPipelinePriceRuleAsync(IBatchPriceRequest batchPriceRequest, IDictionary<string, IEnumerable<IPriceRuleEntry>> pipelinePriceRuleEntries)
	Task OnPriceEscalatorAsync(IBatchPriceRequest batchPriceRequest, List<IPriceEscalator> priceEscalators)

The following table summarizes the **ADJUSTMENT** mode methods of Salesforce pricing callback.

ADJUSTMENT Mode in Salesforce	IPricingTotallingCallback in TurboEngine
void start(ProductConfiguration cart);	Task BeforePricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)
void setMode(PricingMode mode);	Not applicable
void beforePricing(ProductConfiguration.LineItemColl itemColl);	Task BeforePricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)
void beforePricingLineItem(ProductConfiguration.LineItemColl itemColl, LineItem lineItemMO);	Task BeforePricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)
void afterPricingLineItem(ProductConfiguration.LineItemColl itemColl, LineItem lineItemMO);	Task AfterPricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)
void afterPricing(ProductConfiguration.LineItemColl itemColl);	Task AfterPricingCartAdjustmentAsync(IAggregateCartRequest aggregateCartRequest)

ADJUSTMENT Mode in Salesforce	IPricingTotallingCallback in TurboEngine
void finish();	Task OnCartPricingCompleteAsync(IAggregateCartRequest aggregateCartRequest)

- The mapping shown above is for reference only and varies based on the use case implemented in the Salesforce pricing callback.
- Review the Salesforce pricing callback code and put functional instances in equivalent TurboEngine extension points.

Use Case

For example, in the Salesforce pricing callback's *BeforePricing* method, there is a **logic** set to a custom field value on a line item based on LineItem's resolved PLI's list price. The **logic** is not included any mode (BASEPRICE or ADJUSTMENT) check.

The migration process for the given SFDC pricing callback code to the TurboEngine callback code and where to insert the code in the TurboEngine callback code is explained below.

From a functional point of view, the logic is applicable for ADJUSTMENT mode's *BeforePricing* method. Because, in BASEPRICE mode's *BasePricing* method, LineItem's resolved PLI is not available. Hence, the most probable mapping point in TurboEngines is the BeforePricingCartAdjustmentAsync method of IPricingTotallingCallback. On the other hand, since the logic uses just resolved PLI's list price, another probable mapping point in TurboEngines is the *OnPricingBatchAsync* method *IPricingBasePriceCallback* where line item's resolved PLI details are available.

Managing Turbo Formula Fields

Refer to the following steps in order to use the Turbo Formula Fields.

1. Identifying the price impacting fields
2. Adding (Custom or Reference) fields in Line Item Criteria settings
3. Adding price Impacting fields in Line Item custom settings
4. Calling following TP LineItem custom settings and Formula Fields refresh API
5. Enabling Formula Fields feature flag

Identifying the price impacting fields

TurboEngine identifies the impacting pricing fields by referring to the Config LineItem Custom Fields and System Properties. Therefore, you must manually locate the impacting pricing fields in the *Display Column Settings* page and configure them in either *Config System Properties* or *Config LineItem Custom Fields*.

To view or identify the price impacting fields

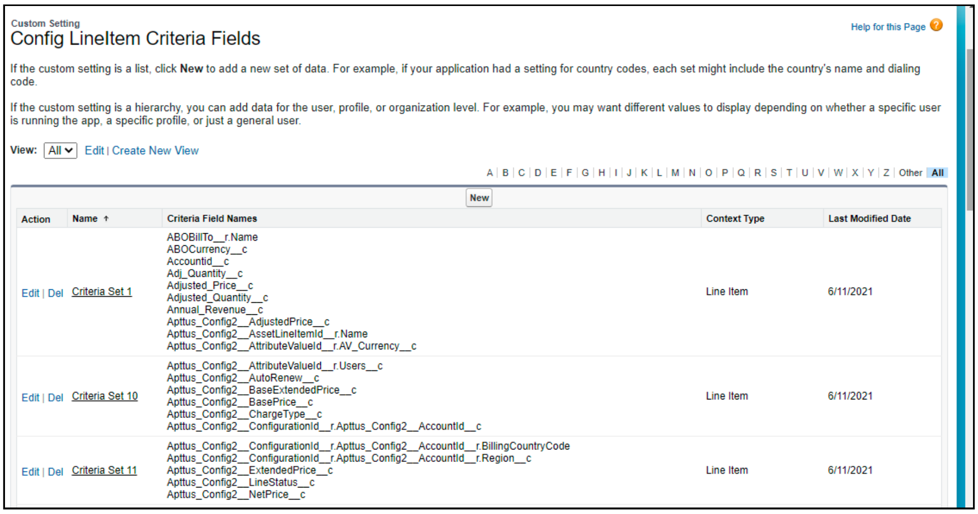
1. Log in to Salesforce org.
2. Click + icon > Config Settings >Display Column Settings. A Config Custom Display Column page is displayed with a list of formula fields and custom fields. Identify all price impacting fields in the list.

Adding (Custom or Reference) fields in Line Item Criteria settings

Suppose there are custom fields from the proposal or product configuration objects used in formula fields or call back code. In that case, you must add them to the Config LineItem Criteria Fields to make those fields available on line item transactional data.

To add custom fields in Line Item Criteria Settings

1. Go to **Setup** > **Custom Settings** > **Config LineItem criteria Fields**.



2. Click **Edit** to the desired Criteria name and enter the required custom field.
3. Click **Save.**

Adding price Impacting fields in Line Item custom settings

To configure price impacting fields

You can configure fields that impact pricing using one of the following methods.

Method 1:

1. Go to **Setup** > **Custom Settings** > **Config System Properties**.

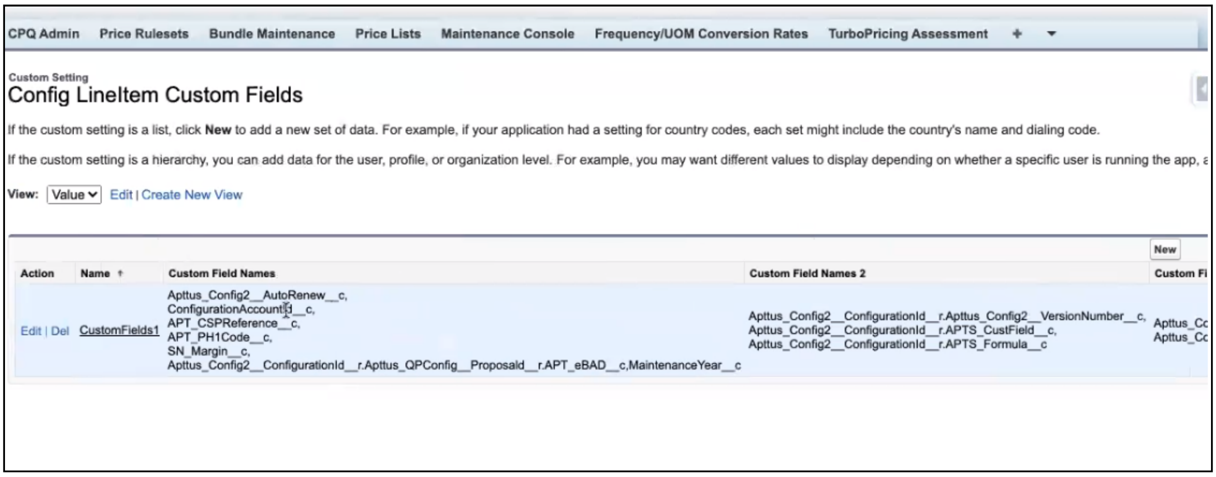
2. Click **System Properties**.

3. Click **Edit** and Add the price impacting field identified in the previous section in either *View Cart Custom Fields* or *View Cart Custom Fields 2.*

> ⓘ A comma or a new line must separate each field.

Method 2:

1. Go to **Setup** > **Custom Settings** > **Config LineItem Custom Fields** > **Manage.** A list of Custom Field names is displayed.



2. Click **Edit** > Add the price impacting field identified in the previous section.

   If the specific pricing formula filed internally refers to any custom field, you must add the pricing formula field and its reference custom field. The User Interface identifies the field from which TurboEngine identifies the field.

3. Click **Save**.

Calling TP LineItem custom settings and Formula Fields refresh API

Syncing Pricing Impacting fields in TurboPricing

Onboarding Data Sync Services

To complete post-provisioning for TurboConfig & TurboPricing, the tenant admin must configure settings for data sync services. TurboEngines data sync provides a high-performance mechanism to sync config and pricing master data at regular, scheduled intervals (or on-demand) between Conga CPQ on Salesforce, TurboPricing, and TurboConfig. Before the initial data sync, you must configure settings enable data sync services and give the administrator access to the TurboEngines Data Sync Admin user interface (UI) to set up and schedule or activate the sync.

Perform the following tasks to complete post-provisioning data sync tasks for TurboConfig and TurboPricing.

> You must run validate API **(***GET /ds/api/dataintegration/v1/Validate*) after first onboarding data sync, after every upgrade, and if there is a change in the object(s) or Salesforce user profile permissions. for more information, see Turbo Data Sync APIs.

Configuring Data Sync Specific Settings

You must configure the data sync service URL and a CSP Trusted Site entry so SFDC can communicate with an external server.

To configure the service URL

1. Go to **Setup** > **Custom Settings**.
2. Go to **Turbo Engine Admin Properties**.
3. Click **Manage**.
4. Click **New**.
5. Enter the following required properties:
    - Name: *LightsaberServiceUrl*.
    - TurboEngines service Endpoint – Endpoint URL provided by Conga CloudOps
6. Click **Save**.

> ⓘ Do not enter the forward-slash (/) symbol at the end of the Endpoint URL.

To configure the CSP Trusted Site

1. Go to **Setup > CSP Trusted Sites**
2. Click **New Trusted Site**.
3. Enter the following required properties:
   a. Enter a **Trusted Site Name** for the trusted site (for example, "TurboEngineAdminService")
   b. Enter the **Trusted Site URL** (this is the service URL from the previous task).
4. Click **Save**.

Enabling My Domain

As an additional prerequisite to using the Data Sync Admin, you must deploy the "My Domain" feature in Salesforce.

For more information, refer to https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/intro_reqs_my_domain.htm and https://help.salesforce.com/articleView?id=domain_name_overview.htm&type=5

Configure Permissions for Data Sync Admin User

Users who need to configure and run TurboEngines Data Sync must have permission to access and use the Data Sync Admin UI. This can be a user assigned to the System Administrator profile, or you can customize a profile and create one or more users in this role.

> ⓘ You must verify the JWT flow settings and access levels for your org after completing the tenant onboarding. For more information, see Verifying JWT Flow Settings and Access Level.

To check if the current user has the right permissions:

1. Log in to your organization as the admin user.
2. Open the Salesforce App Launcher (Lightning) and launch the **TurboEngines Admin** app.
3. If the **Data Integration** and **Callbacks** tabs are visible after launching the app, the user has the correct permissions. Otherwise, log back in as a system administrator and perform the following tasks to provide access to the user profile.

To provide access to the data sync app

1. Go to **Setup > App Manager**.
2. Find the **TurboEngines Admin** app in the list. Click the drop-down and the end of the row and select **Edit**.
3. Click **User Profiles**.
4. From the list of Available Profiles, search and select the app you want to add.
5. Click the right-facing arrow to move the profile from the list of Available Profiles to the list of Selected Profiles.
6. Click **Save**.

To make all tabs visible in the data sync app

1. Go to **Setup > Profiles**.
2. Search for the profile you want to configure and click **Edit**.
3. Under Custom Settings, make sure the following tabs are set as "Default On":
    - **Data Integration**: This tab serves as the starting point for managing all consumer profiles.
    - **Callbacks**: This tab allows you to manage pricing callbacks for TurboPricing.
    - **Consumer Profile**: This tab allows you to set up and configure data sync operations.
    - **Run Details**: This tab allows you to review run history for data sync and take action.

Configuring Single Currency Orgs

Exchange rates are required for currency conversion during pricing and therefore must be synced by TurboEngines. Since single currency orgs do not have a currencies table, any attempt to sync pricing data will fail. Take one of the following two actions when onboarding the org prior to the initial sync:

- Enable multiple currencies in the provisioned org (see Salesforce documentation).
- Raise a support ticket to have currencies removed from the Conversion Rates and Custom Settings consumer profile.

Salesforce orgs are provisioned with a single currency. In such cases, the currencies table will not exist. The currency tables are created and hold the exchange rates only when the administrator enables multiple currencies.

- You cannot change the currency flag for an org after provisioning the tenant.

- You cannot change the currency flag once you have changed it from Single currency to Multi-currency.

Setting up and Syncing TurboConfig Data

For complete information and the tasks required to administer TurboEngines data sync for TurboConfig and TurboPricing, refer to *Data Sync Administrator Guide.*

Verifying JWT Flow Settings and Access Level

You can verify the JWT flow settings and access level on a Connected App by invoking the following APIs.

Order	API	Response
1	ds/api/dataintegration/v1/Validate/SFDCDetails	Trigger this API to verify the response for `validate` and `jwtAuthFeatureFlagEnabled` elements. If the response is *true*, then the validation is successful and JWT Authorization is enabled. `{` `"configuredSFDCUrl": "https://someURL",` `"sfdcUrlFromToken": "https://someURL",` `"oAuthUrl": "https://someURL/services/oauth2/token",` `"userId": "someUserID",` `"clientId": "someText”` `"clientSecret": “someText ",` `"password": "****@**",` `"validate": true,` `"isInstanceUrlMatches": true,` `"rsaPrivateKey": "",` `"featureFlagInitlized": true,` `"jwtAuthFeatureFlagEnabled": true //feature flag` `}`
2	/ds/api/dataintegration/v1/HealthCheck/status	Trigger this API and verify the response, the status of SFDC_TOKEN_GENERATION and CERT_LOCATION_CHECK elements must be displayed as *success.*

Order	API	Response
3	/ds/api/dataintegration/v1/Validate	Trigger this API and copy the validation Id (guId) from the response.
4	/ds/api/dataintegration/v1/Validate/GetValidationResults	Trigger this API by passing validation id copied in step 3. • If the validation is successful, the JWT flow settings are working as intended. • If you find any difference in the result, you need to assign more privileged profiles or permission sets to a Connected App as required. Repeat steps 3 and 4 until validation success. Contact Conga TS if you encounter any issue in providing access.

Logging APIs

This section describes how to use logging APIs to enable logging and Search Logs.

Logging in TurboPricing

There are several systems that are used to record, Search, and analyze logs in TurboPricing. When the logging is enabled for the user, the user can check or search all the logs in TurboPricing using Logging APIs.

Log Types

Currently, TurboPricing supports two types of logs.

1. OOTB Logs
2. Logs that are written in Callbacks

In TurboPricing, all logs are categorized based on the priority (from low to high) as mentioned below.

- Trace (5)
- Debug (4)
- Information (3)
- Warning (2)
- Error (1)

- Critical (0)
- None (-1)

Each Log level above is cumulative of all logging levels. For example, if you enable the log level with "Error", you can see all other log levels "critical" and "None". On the other hand, if you allow the Log level with "Information", then you can see only two levels of logs, "Information", "Warning", "Error", "Critical", and "None".

OOTB Logging

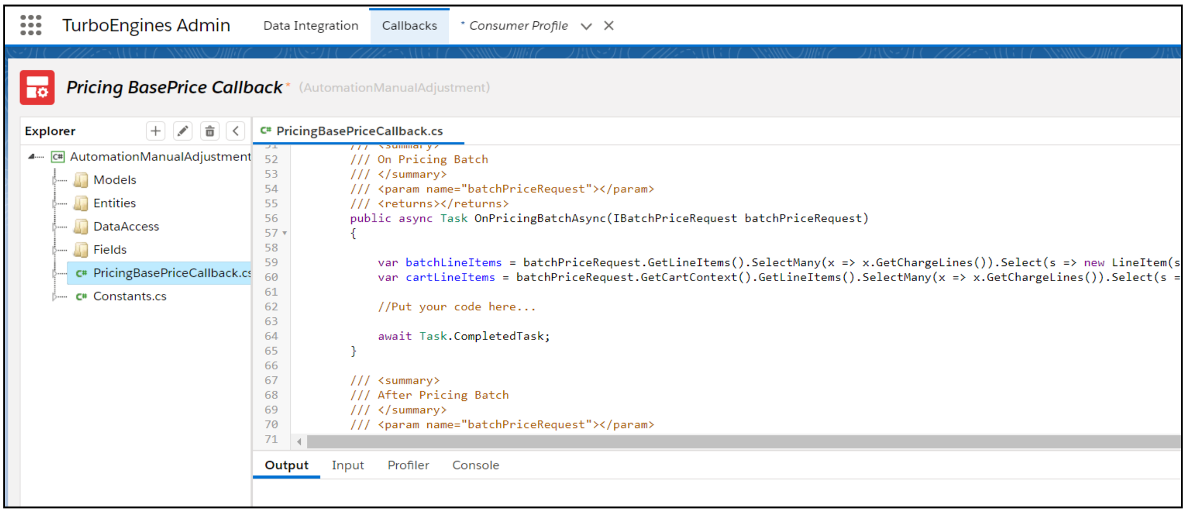
TurboPricing uses extensive logging in all Turbo sub-systems. If the logging is enabled, you can use these logs to find different TurboPricing flows. The logs are available for Metod Enter, Method Exit, and errors and exceptions.

Writing Logs in Callbacks

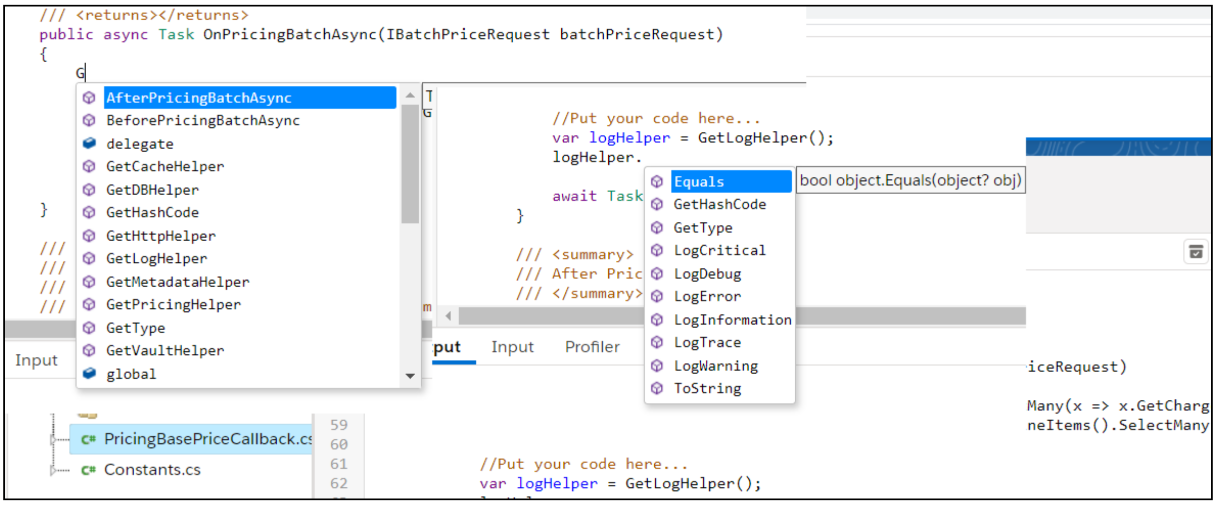
You can write your logs using the Conga Callback Authoring UI in SFDC org while implementing the callbacks like Pricing callback, Totalling callback, etc.

To write logs in Callbacks:

1. Log in to the Salesforce org.
2. Click **Switch to Lightning Experience**.
3. Open the **Salesforce App Launcher (Lightning)** and launch the **TurboEngines Admin** app.
4. Click the **Callbacks** tab. A list of callbacks is displayed.
5. Click **Manage** under the **Actions** column for the required callback to configure it. The detail page of the selected callback with a list of projects related to the callback is displayed.
6. Click **Create** to create a new project.



7. Add logger helper using methods in the Callback class. Click **Ctrl + Space Bar** and enter to see the Suggestions.
   Follow the same procedure to add a variable name. For example, enter *logHelper* and click **Ctrl + Space Bar** to see all available methods.



To enable and check the Logs

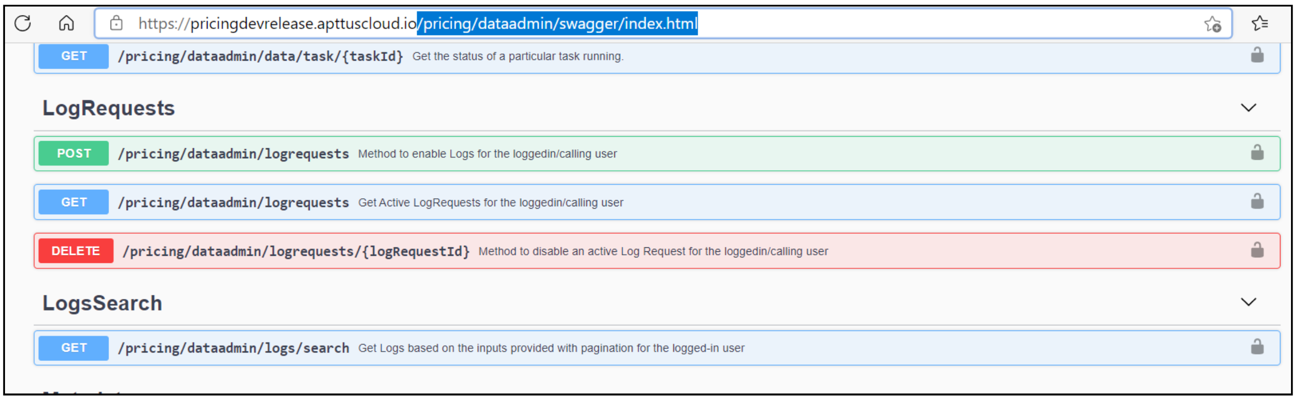
Refer to the following topics for step-by-step instructions to enable and check the logs you have written in the Callback code.

1. Enable Logging for the user
2. Perform TurboPricing Activities such as Create a Cart, Add Lines to the Cart, Get the Pricing Response, etc.
3. Use Search Log (`GET /pricing/dataadmin/logs/search`) to see the logs.

Enabling Logging APIs

Logging APIs allow you to enable or disable logging for a user and search or filter logs using keywords.

- To enable logging for the user
- To check the existing log requests
- To delete existing Log requests



To enable logging for the user

Invoke the following API to enable the Logging for the given user.

API Details	Example
POST /pricing/dataadmin/logrequests Input Parameters: None Authorization: Bearer <access token>	Type: POST /pricing/dataadmin/logrequests

Example Response

```
{
  "Success":true
}
```

To check the existing log requests

Invoke the following API to get the active LogRequests for the logged-in user.

API Details	Example
GET /pricing/dataadmin/logrequests Input Parameters: log-level Data Type: String Authorization: Bearer <access token>	Type: GET /pricing/dataadmin/logrequests

Example Response

To delete existing Log requests

Invoke the following API to delete any existing log enabling request using request ID.

API Details	Example
DELETE /pricing/dataadmin/logrequests Input Parameters: log Request ID Data Type: String Authorization: Bearer <access token>	DELETE /pricing/dataadmin/logrequests

Example Response

> ⚠ If you have subscribed with Log Level *Warning* and want to switch the log level to *Information*, you must delete the existing log request and re-create it with the new required log level.

Configuring Admin Settings

Admin Settings are system properties that have different values and you can use them for different purposes. Use the following admin and customs settings, to conform your implementation's behavior to your unique business requirements.

To create admin entries

1. Log in to Salesforce.

2. Click + and click **Admin.** All the out-of-the-box admin settings are displayed on the Admin home page. You can add or create new Admin entries based on your organization's requirements.
3. To create a new Admin entry, click **New.**
4. Type Name, Value, and Code for the admin setting and click **Save.** Your new admin entry is saved and added.

Admin Settings in TurboEngines

The following content provides information about admin entries available for TurboEngines along with their values and purposes.

APTS_RestoreLinesOnChange

Whenever you modify the line items after saving the cart, those modifications must display on the cart page whenever you launch the cart. To achieve this, you must configure the following admin setting to communicate line item changes between Salesforce and TurboPricing. The following property allows you to restore the line items when they are modified after saving the cart.

> If you relaunch the cart when the turbo cart does not exist, CPQ ignores this setting and restores the full cart in the turbo flow. This setting becomes effective from the subsequent cart launch.

Name	APTS_RestoreLinesOnChange
Data Type	String
Value	• **True:** CPQ sends price pending line items to turbo pricing on cart launch. • **False:** CPQ restores the complete payload for all line items.
Code	N/A

Frequently Asked Questions (TurboConfig)

What is TurboConfig and how does it work?

TurboConfig is a configuration engine to process product configuration rules while configuring products and finalizing a quote. TurboConfig offloads the computation workload from the Salesforce platform to the Conga Flexible Compute Platform built using microservices. The benefit of the TurboConfig is that users can sell complex configurations much faster because of significantly-optimized processing time. Also, it allows customers to expand the solution to other business units and sell faster.

For example, in a TurboConfig-enabled flow, when the Sales Representative adds a product or a favorite configuration to the cart, the application of complex constraint rules associated with them is offloaded to the Conga Flexible Compute Platform to process for faster response.

When do I need to use TurboConfig?

The Salesforce platform has limitations (such as heap size, CPU timeout limits, number of SOQL limits, and view state) that result in slower response times and usability issues. TurboConfig handles such complex rules and processes a volume of rules significantly faster.

TurboConfig is recommended when you have a large number of rules or highly complex configuration rules to be applied while selecting a product or configuring a bundle.

For example

- If you have more than 100 constraint rules (inclusion, exclusion, recommendation, and replacement rules) applicable across standalone and bundle products
- If you have more than 50 field expressions applicable across products and bundles
- If you have more than 100 product attribute value rules applicable across bundles
- If you have a complex bundle structure that includes more than 500 options and several option groups
- If you have complex bundles rules such as min/max, custom filter callback, repeat inclusions.

How do I enable TurboConfig?

You must have a license for TurboPricing or TurboConfig to enable either service. If you do not have a license, please contact your Conga Account Executive before you begin. After you acquire a license TurboConfig instances will be provisioned for you.

For detailed instructions on how to enable TurboConfig, refer to instructions on how to turn on TurboConfig.

Which version of CPQ should I be on to use TurboConfig?

You must be on CPQ on Salesforce Summer 2020 release or above.

What are the supported features in TurboConfig?

For a complete list of supported features on TurboConfig, refer to the feature matrix.

Is the TurboConfig available for all products or only select products?

You can use TurboConfig for all or select products. However, the TurboEngines data sync services configure the selected products at a regular frequency.

For instructions on how to onboard the data sync services to TurboConfig, refer to the instructions here.

How does TurboConfig work with Data Sync for synchronizing the data?

TurboEngines data sync provides a high-performance mechanism to sync the config and pricing master data at regular, scheduled intervals (or on-demand) between Conga CPQ on Salesforce, Turbo Pricing, and Turbo Config. To start using the data sync services, the tenant admin must configure settings for data sync services. For more information on configuring data sync settings, refer to Onboarding data sync services.

Is there a way to automate the data sync of all products and changes on a regular basis?

Yes, TurboConfig administrators can now make use of the TurboEngine Data Sync Admin application to configure and manage master data sync at regular scheduled intervals and on-demand.

Does TurboConfig work on existing quote or configurations, which were created using a different constraint rule execution mode?

Yes, if you have quotes in progress and if you have configured quotes using the *Client* execution mode, you can process the quotes using the *CMS* execution mode. However, if you have created a quote using the *CMS* execution mode when you switch to the *Client* execution mode, you may have to delete the line items and add them again.

Can I configure custom flows for TurboConfig?

Yes, You can enable TurboConfig for selective CPQ flows. You can use this functionality to avoid making TurboConfig as the default configuration engine and use the engine to process large and complex configuration rules.

For more information on configuring custom flows, refer to Configuring Custom Flows for TurboConfig.

How do I switch from Server Side/Client Side constraint rules to TurboConfig?

For detailed instructions on how to enable TurboConfig, refer to instructions on how to turn on. Also, refer to the feature parity matrix before you switch to TurboConfig. Note that custom callbacks are not supported in this release.

When I refresh my Salesforce Sandbox org, should I change any config settings?

When you refresh your sandbox, you must reconfigure TurboConfig after the refresh. Follow the onboarding process to enable TurboConfig in your sandbox after you have refreshed the sandbox.

I have done some customization in my org such as added formula fields, workflow rules, callbacks. Do my customizations work when I switch to TurboConfig?

There is no impact on any customizations you may have done on CPQ Objects. However, if you have written any configuration callbacks such as Option Filter Callbacks, you will be required to migrate your callback to TurboConfig using the microservice callback framework. Note that the callbacks are not supported in TurboConfig in the Winter 2020 release. Refer to the supported feature matrix before switching to TurboConfig engine.

Is TurboConfig supported to work with ABO and Service CPQ?

No. ABO and Service CPQ are not supported to work with TurboConfig in the Winter '20 release. For a complete list of supported features on TurboConfig, refer to the Feature Matrix.

Is TurboConfig security and privacy compliant?

TurboEngines run in a secure multi-tenant environment and TurboEngines are designed to provide full security and privacy with your data. The services are hosted in IBM Cloud, which is ISO 27001/2, SOC 1/2, GDPR compliant. Conga takes advantage of data encryption and access control features enabled by the cloud service provider. If you have any questions or need details, contact Conga Technical Support.

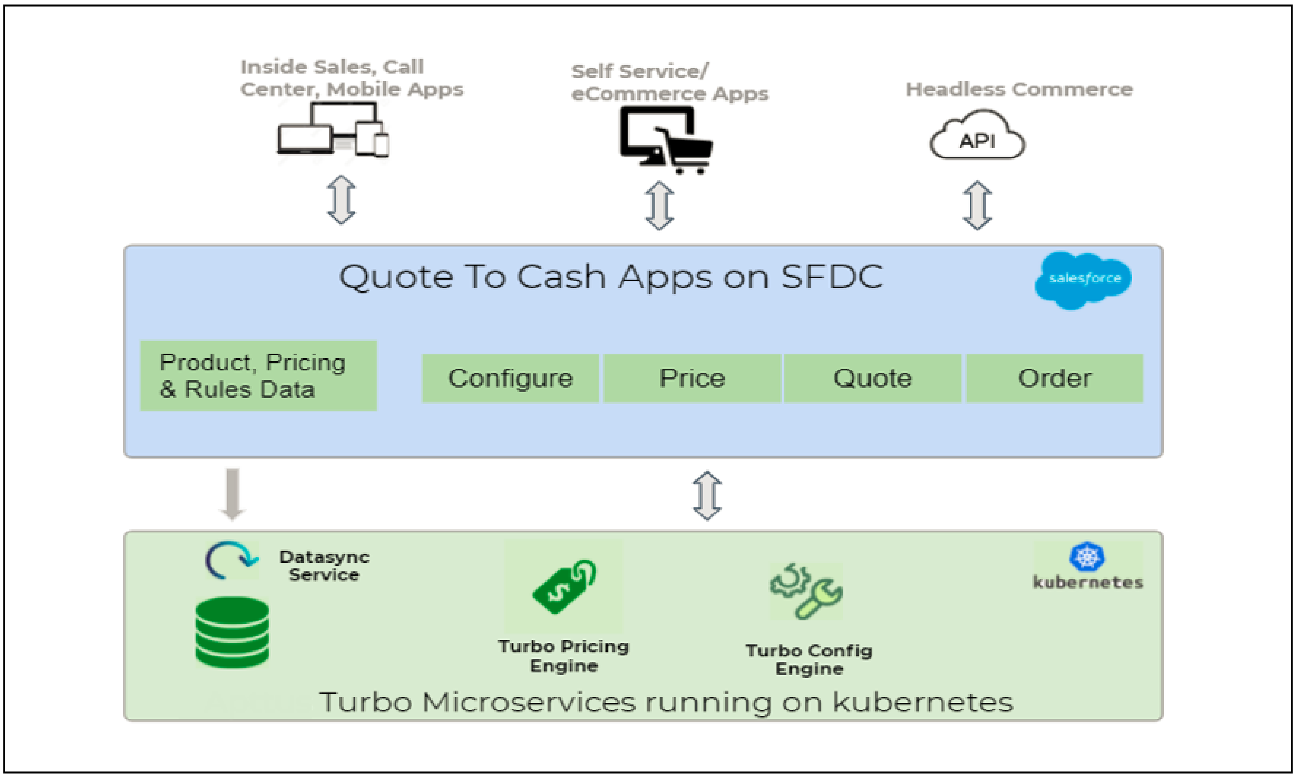
About TurboEngines (CPQ)

Conga TurboEngines is a concurrent processing engine provided by Conga comprising various microservices that process product configurations (TurboConfig), pricing calculations (TurboPricing), and other product-related business data, such as promotions. Conga TurboEngines offload the computation workload from the Salesforce platform to the Conga Flexible Compute Platform to reduce the processing time on the cart. In addition, processing the computation workload in the Conga Flexible Compute Platform reduces the interaction costs and the quote turnaround time, specifically during peak load or large transactions.

TurboEngines scale on the following dimensions:

- Number of users
- Size of transaction
- The complexity of the product and rules

TurboEngines also provide a critical component called TurboEngines Data Sync services that offer a high-performance mechanism to sync pricing and config master data at regular, scheduled intervals (or on-demand) between Salesforce and the Conga Flexible Compute Platform. Data is pushed to TurboPricing and TurboConfig consumer endpoints and made available for processing to take advantage of the performance improvements offered by the TurboEngines platform.



About TurboConfig (CPQ)

TurboConfig is a configuration engine created to process product configuration rules when products and bundles are configured on a cart and when finalizing the quote. TurboConfig offloads the computation workload from the Salesforce platform to the Conga Flexible Compute Platform built using microservices to reduce the processing time of the configuration rules. Computation workload includes the processing of rules defined on the products. For example, in a TurboConfig enabled flow, when the Sales rep adds the product or the favorite configuration to the cart, the constraint rules associated with them are offloaded to the Conga Flexible Compute Platform to process. TurboConfig engine executes the rules, maintains rule states, and avoids unnecessary line item processing.

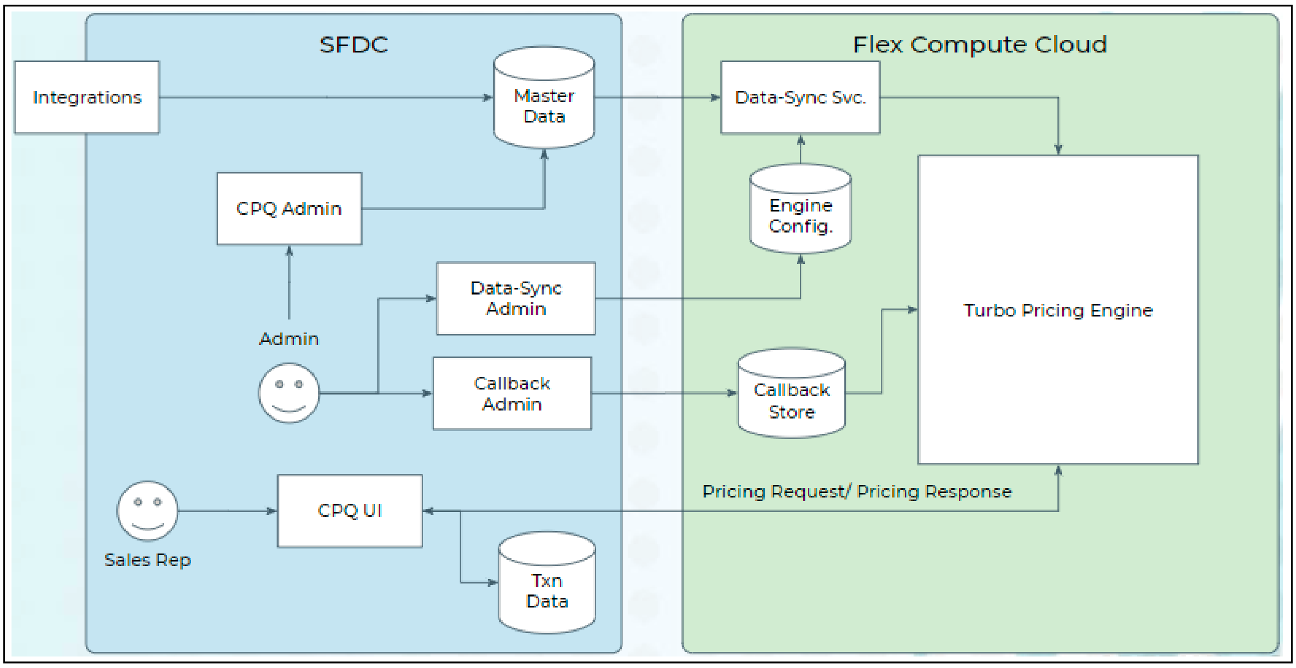
TurboConfig is recommended when you have a large number of rules or highly complex configuration rules to be applied while selecting a product or configuring a bundle.

To get started enabling TurboConfig for your org, refer to Enabling TurboEngines in an Org. To learn more about the TurboConfig service, refer to Frequently Asked Questions (TurboConfig).

About TurboPricing (CPQ)

TurboPricing is a pricing engine built using microservices to reduce the processing time on the cart. You can enable TurboPricing to offload complex pricing computation workload from the Salesforce platform to the Conga Flexible Compute Platform. It reduces time to submit prices to customers, improves user experience, and improves user adoption with a more responsive user interface.

The following diagram shows how data flows between a Salesforce org and Conga Flexible Compute Platform:



Supported Features in CPQ UI with TurboPricing Integration

Following are the features available on CPQ UI when you enable TurboPricing.

- Price Waterfall

Price Waterfall

About Price Waterfall

Price Waterfall is the industry standard for Price and Margin calculation. Price waterfall defines a sequence to handle different prices with the target of reaching the net price.

A well-defined Price Waterfall provides transparency and high visibility of pricing calculation and helps protect margins and avoid incorrect pricing.

Price Waterfall comes with features transparency and high visibility in pricing calculation and protecting margins, avoiding incorrect pricing. In addition, Price Waterfall defines a sequence to treat different prices with the end goal of arriving at the net price charged to the customer.

Price Pipeline enables administrators to configure the price execution flow and set the groundwork for Conga's Price Waterfall feature to display detailed pricing steps for end-users to visualize. Price waterfall is a tool that tells the user how much revenue the company is generating from a transaction. The waterfall is typically represented in a column chart consisting of critical price or margin points and any adjustments that were applied on the list price to arrive at the final price or margin.

The advantages of the price waterfall functionalities are as follows:

- Provides Sales Representatives with the ability to drill down on any price point visible to them so that they can better communicate the deal to customers giving them more leverage during a negotiation.
- Provides Approvers and Deal Desk with a complete picture of the different price points, preapproved and non-standard adjustments, and costs to better analyze the health of the deal.
- Provides Cost Accountants and Controllers to drill down on the cost to ensure that the true profitability is calculated for the deal to ensure the company's bottom line goals are met.
- Enable Admins to set up the price waterfall view so that the different types of end-users only view the data points of interest to them and what is allowed of them.

Configuring Price Waterfall

To configure price waterfall, perform the following steps:

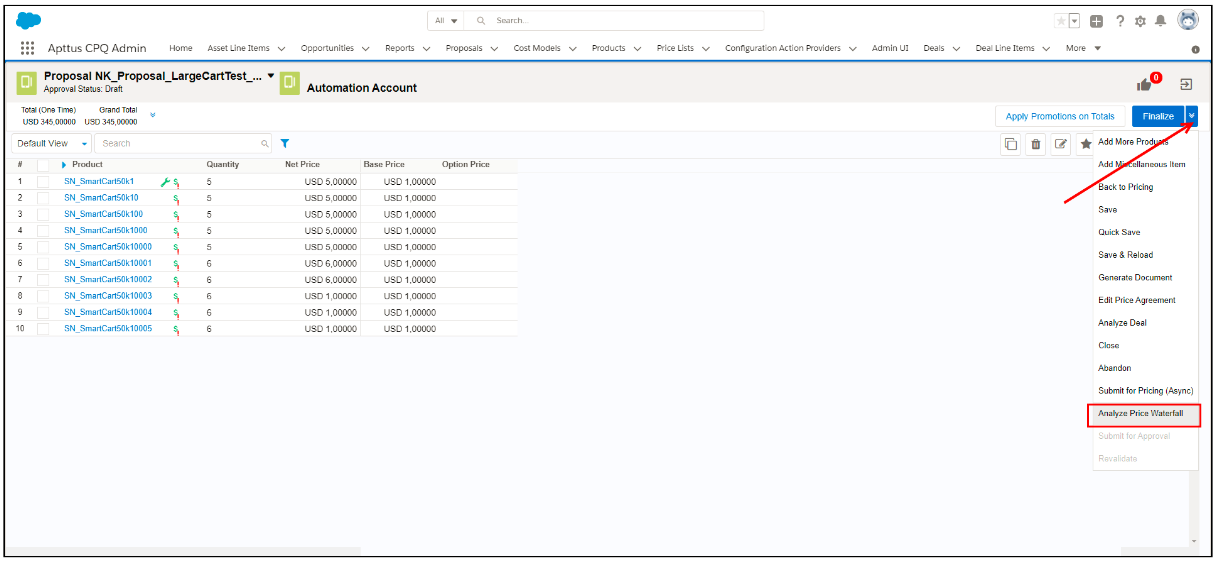
Step	Task	Description
1	Define Pricepoints	Create a new price point as a picklist and add new price points as new picklist values in the Price Rule object.
2	Configure Price Pipeline	Define price points and configure price pipelines.
3	Configure a Price Waterfall to a Price Pipeline	Configure a price waterfall to a price pipeline. Create additional waterfalls to a price pipeline, if required.

Analyzing Price Waterfall

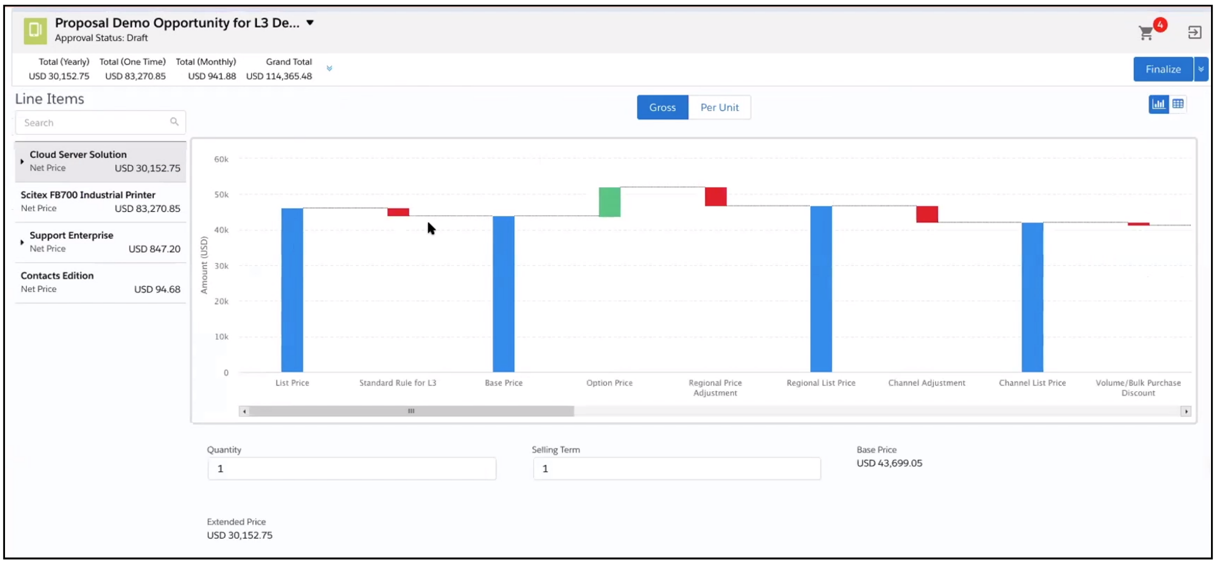
The Analyze Quote page enables the sales representative to view and analyze the price, view adjustments and how the net price is derived, the margin for the product, define the best possible price, analyze Key Performance Indicators, etc.

To analyze price waterfall:

1. Navigate to the Catalog.
2. Click **Add products**.
3. Click **Go To Pricing**. The Cart page is displayed.



4. Click **Analyze Price Waterfall**. The Analyze page is displayed.



The analyze Price Waterfall page displays the following information:

- **Price Waterfall Chart:** The waterfall chart is a bar graph where Line Item Fields are plotted vertically and Cost is plotted horizontally.
- **Per Unit and Gross tabs:** Per Unit displays price waterfall in terms of unit prices. Gross displays price waterfall in terms of the gross price.
- **Menu icons to display the Price waterfall information:** You can select whether to view price waterfall in the form of a bar chart () or in the form of a table ().
- **Price Waterfall Adjustment:** If the price point is marked as modifiable, then you can make relevant adjustments for various price points in the Create Manual Adjustment window.

Field	Description
Name	Enter a name for the adjustment.
Adjustment Type	Select adjustment type from the drop-down list.
Price Point source	Select the source of price point.
Amount	Enter the required adjustment amount.

Price Pipeline

A price pipeline is a sequence of price points.

Price rules will derive price points. You can create multiple price points, arrange them in a sequence to define the price pipeline, and map one price as net price. Base Price and List Price are mandatory price points.

When the pricing engine is pricing a line item, it has to calculate the per-unit amount associated with each price point and the gross amount (amount incorporating the quantity and term) if the transaction is a quote or order. The price point definition will describe how it is to be calculated and whether it is treated specially (for example, whether it is the net price used for integration with downstream systems such as billing.)

Defining Price Points

For creating new price points, Admin must add them as new picklist values in the Price Rule Object in Salesforce.

To create a Price Point as a new picklist:

1. Go to **Setup** > **App Setup** > **Create** > **Manage your custom objects** > **Price Rule**.
2. Under **Custom Fields & Relationships** section, click **New.**
3. Select **Picklist** from the Field Type.
4. Click **Next.**
5. Enter **Price Points** in the **Field Label**.
6. Enter values for the picklist.
7. Click **Next.**
8. Establish field-level security and click **Next.**
9. Click **Save.**

To add new price points as picklist values:

1. Go to **Setup** > **App Setup** > **Create** > **Manage your custom objects** > **Price Rule**.
2. Under the **Custom Fields & Relationships** section, click **Price Points**.
3. Under the **Values** section, click **New.**
4. Enter new price points in the textbox and click **Save.**

> The first two price points must be list price and base price and other price points can be defined as per use case.

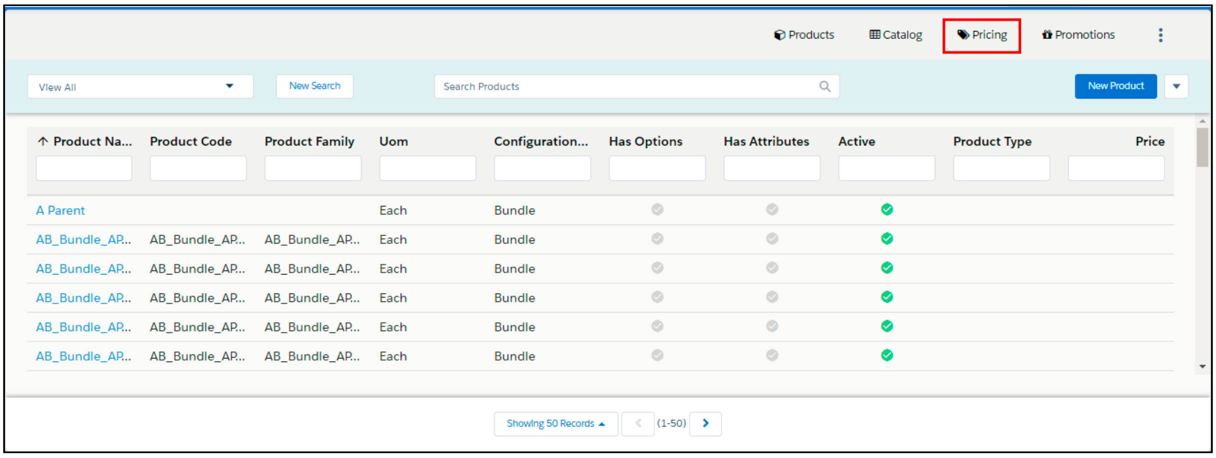
Configuring a price Pipeline

Pricing administrators or CPQ administrators define price points and configure price pipelines. Price pipelines are further used in the price waterfall and also define which price point is impacted by the price rule set.

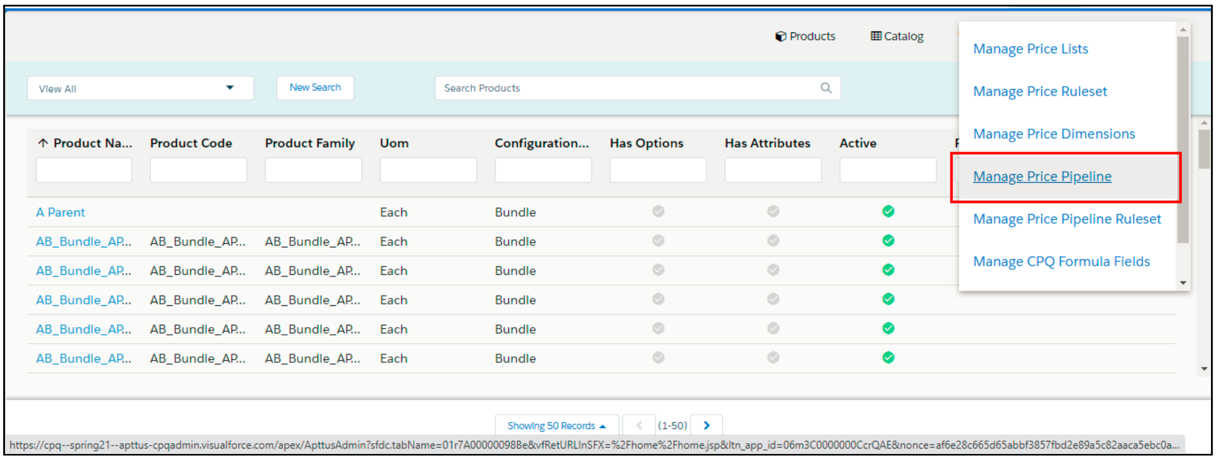
Ensure that you have enabled the price pipeline custom setting. For more information, refer to Configuring the Pricing Engine.

To add a price pipeline:

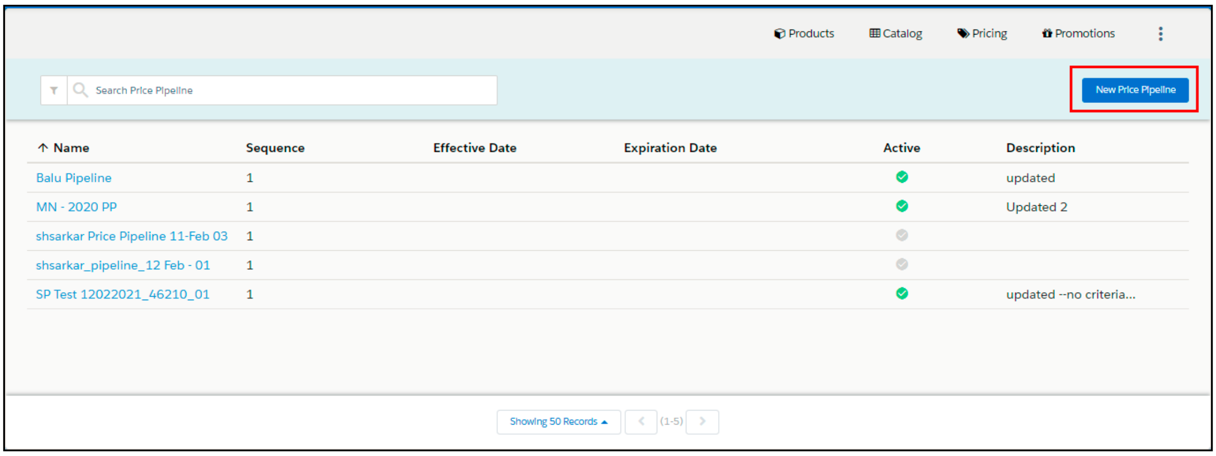
1. On **App Menu**, click **Apttus CPQ Admin** and then click the **Admin UI** tab. The new admin console is launched.



2. Select **Pricing** > **Manage Price Pipeline**.



The Manage Price Pipeline page is displayed.



3. Click **New Price Pipeline** and update the following fields:

Field Name	Description
Price Pipeline Name	Name of the price pipeline
Sequence:	
Is Active	Slide the slider to enable the price pipeline
Description	Description of the price pipeline
Effective Date	Enter a date to associate an effective start date with the price pipeline
Expiration Date	Enter a date to associate an effective end date with the price pipeline
Pipeline Criteria	Define the criteria for the price pipeline. The price pipeline is used based on the criteria defined by the user. If the given criteria are not met then the default pipeline is used. If there is no default pipeline, then a price waterfall chart is not created. Define the criteria for Line Items and Product Attributes as follows. • For Line Items, click the icon > Line Items > Add New Criteria > enter the desired Field (Select the field that must be used for the adjustment), Operator, and Value > Save. • For Product Attributes, click the icon > Product Attributes > Add New Criteria > enter the desired Field (Select the field that must be used for the adjustment), Operator, and Value > Save.

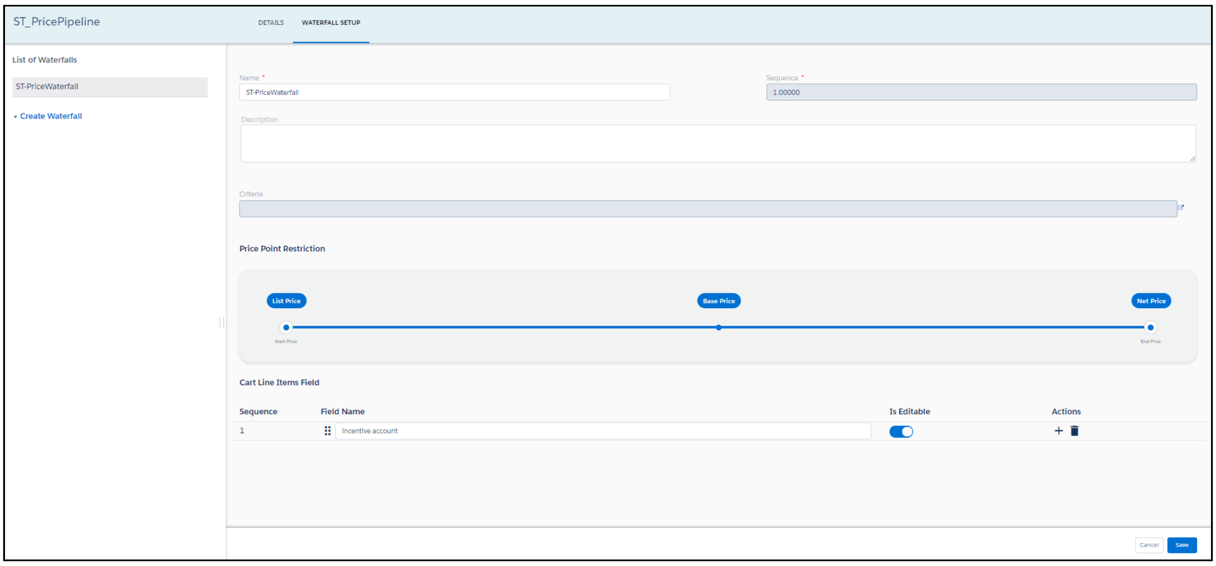
Field Name	Description
Define Price Points:	Define price points as follows. List price and Base Price are set by default. a. Click the text box in the name column and select the desired price point from the list. b. Is Modifiable: This allows you to modify the price point on the Waterfall page. For example, if you enable the modifiable option for invoice price, then the sales representative can modify the Invoice price of the product on the cart and apply it to the price. Move the slider to enable the Is Modifiable option. c. Is this Net Price: This allows you to make a price point as a net price. Move the slider to enable the option. Any manual adjustments that the user performs on the cart appear on the waterfall chart against this price point (which is marked as the net price). d. Click the '+' icon to add new price points.

4. Click **Save.**

> You must set at least one Price Point as a Net Price.

Configuring Price Waterfall to a Price Pipeline

1. On **App Menu**, click **Apttus CPQ Admin** and then click the **Admin UI** tab. The new admin console is launched.
2. Select **Pricing** > **Manage Price Pipeline**.
3. Click the required Price Pipeline > Click the **WATERFALL SETUP** tab.



4. Update the following details and click Save.

Field	Description
Name	Name of the price waterfall.
Description	Provide a description.
Criteria	Lists the criteria that are defined.
Price Point Restriction	Set the visibility restriction on price points. This option allows you to control the visibility of price points for the user. 
Cart Line Items Field	a. Click the text box in the name column and select the desired price point from the list. b. Is Editable: This allows you to modify the price point on the Waterfall page. For example, if you enable the editable option for *Invoice Price*, then the sales representative can modify the *Invoice price* of the product on the waterfall chart and apply it to the price. Move the slider to enable the Is Editable option. c. Click the '+' icon to add new field names. d. Click **Save.**

Creating a new Price Waterfall to a Price Pipeline

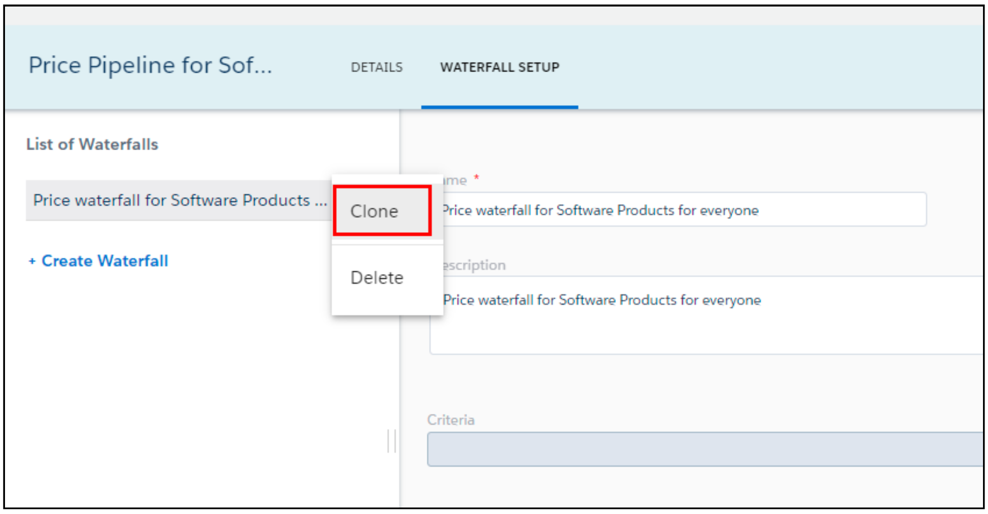
1. On **App Menu**, click **Apttus CPQ Admin** and then click the **Admin UI** tab. The new admin console is launched.
2. Select **Pricing** > **Manage Price Pipeline**.
3. Click the required Price Pipeline > Click the **WATERFALL SETUP** tab.
4. Click **Create Waterfall** and enter the required information.
5. Click **Save**.

> You can create a maximum of 10 waterfalls for each price pipeline.

Cloning the waterfall

Once you created a waterfall to price pipeline, you can replicate the waterfall using the clone feature. Later, you can modify the waterfall as per customer needs.

1. In the **List of Waterfalls** pane, place the cursor on desired waterfall name and click the (⋮) icon.
2. Click **Clone**.



3. Enter the details as required >Click **Save.**
   A cloned copy of waterfall is created as *Waterfall name-<Copy>.* You can have a maximum of 10 waterfalls for each price pipeline.

To Modify a Price Pipeline

1. On **App Menu**, click **Apttus CPQ Admin** and then click the **Admin UI** tab. The new admin console is launched.
2. Select **Pricing** > **Manage Price Pipeline**.
3. Click the required Price Pipeline > Modify the details and click **Save.**

Managing Price Pipeline Ruleset

Price Waterfall Feature

About Price Waterfall

Price Waterfall is the industry standard for Price and Margin calculation. Price waterfall defines a sequence to handle different prices with the target of reaching the net price.

A well-defined Price Waterfall provides transparency and high visibility of pricing calculation and helps protect margins and avoid incorrect pricing.

Price Waterfall comes with features for transparency and high visibility in pricing calculation and protecting margins, avoiding incorrect pricing. In addition, Price Waterfall defines a sequence to treat different prices with the end goal of arriving at the net price charged to the customer.

Price Pipeline enables administrators to configure the price execution flow and set the groundwork for Conga's Price Waterfall feature to display detailed pricing steps for end-users to visualize. Price waterfall is a tool that provides users detailed information on how much revenue the company is generating from a transaction and the pocket margin of that transaction. The waterfall is typically represented in a column chart consisting of critical price or margin points and any adjustments that were applied on the list price to arrive at the final price or margin.

The advantages of the price waterfall functionalities are as follows:

- Provides Sales Representatives with the ability to drill down on any price point visible to them so that they can make better decisions and better communicate the deal to customers giving them more leverage during a negotiation.
- Provides Approvers and Deal Desk with a complete picture of the different price points, preapproved and non-standard adjustments, and costs to better analyze the health of the deal.
- Provides Cost Accountants and Controllers' ability to drill down on the cost (if cost is included as part of the Price Pipeline) to ensure that the true profitability is calculated for the deal to ensure the company's bottom line goals are met.

Configuring Price Waterfall

To configure price waterfall, perform the following steps:

Step	Task	Description
1	Define Pricepoints	Create new price points as picklist values in the "Price Points" field in the Price Rule object in salesforce so that these price points can be used when defining any price pipeline.
2	Configure Price Pipeline	Define price points and configure price pipelines.
3	Configure a Price Waterfall to a Price Pipeline	Configure a price waterfall to a price pipeline.

Defining price Points

For creating new price points, Admin must add them as new picklist values in the "Price Points" field in the Price Rule Object in Salesforce.

To add new price points as picklist values:

1. Go to **Setup** > **App Setup** > **Create** > **Manage your custom objects** > **Price Rule**.
2. Under the **Custom Fields & Relationships** section, click **Price Points**.
3. Under the **Values** section, click **New.**
4. Enter new price points in the textbox and click **Save.**

Configuring a Price Pipeline

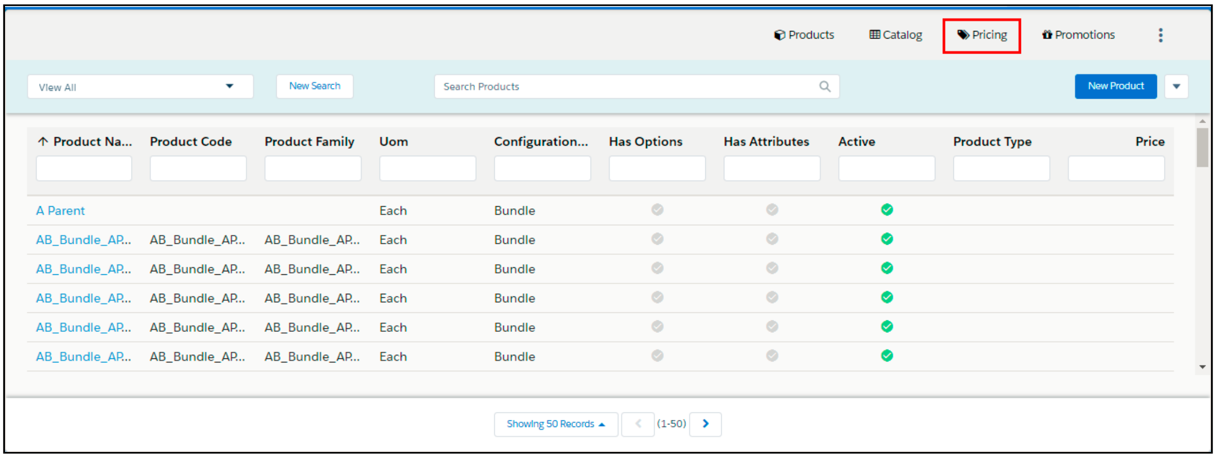
A price pipeline is a sequence of price points.

You can add multiple price points in a price pipeline definition, arrange them in a sequence to define the price pipeline, and map one price as net price. Base Price and List Price are mandatory price points, and hence those will already be added to every price pipeline by default. When the pricing engine is pricing a line item, it calculates the per-unit amount associated with each price point and the gross amount (amount incorporating the quantity and term) if the transaction is a quote or order. The price point definition describes how it should be calculated and whether it is treated specially (for example, identifying a particular price point as the Net Price point for integration with downstream systems such as billing.)

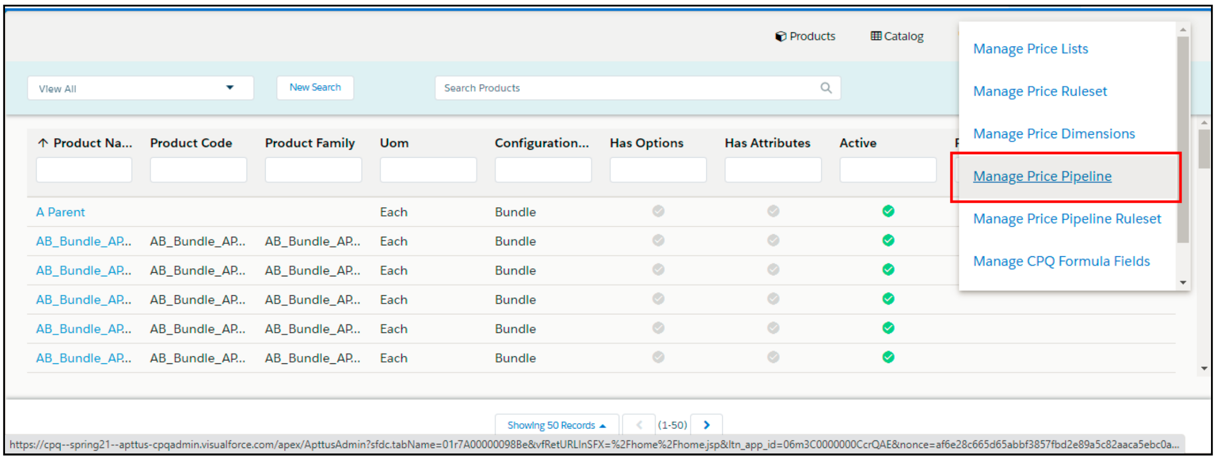
Pricing administrators or CPQ administrators define price points and configure price pipelines. Price Pipeline definition is further used in the Price Waterfall chart to determine whether certain features (such as manual adjustment to any price point) are allowed to the end-user or not. For more information, refer to Configuring the Pricing Engine.

To add a price pipeline:

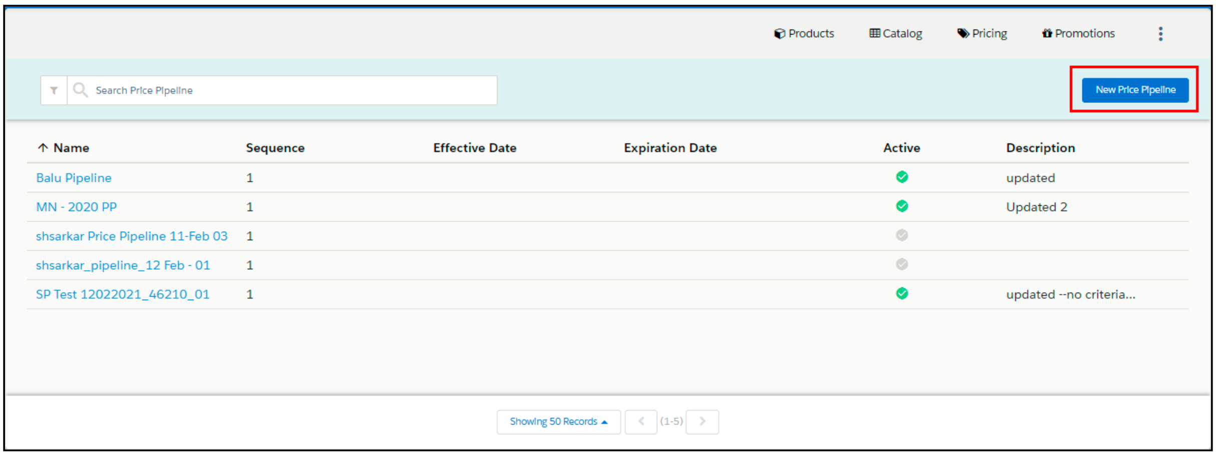
1. On **App Menu**, click **Apptus CPQ Admin** and click the **Admin UI** tab. The new admin console is launched.



2. Select **Pricing** > **Manage Price Pipeline**.



The Manage Price Pipeline page is displayed.



3. Click **New Price Pipeline** and update the following fields:

Field Name	Description
Price Pipeline Name	Name of the price pipeline
Sequence	The sequence allows you to determine the correct price pipeline to apply if multiple price pipelines match the criteria for a line item. The lowest matching sequence will have the highest priority.
Is Active	Slide the slider to enable the price pipeline
Description	Description of the price pipeline
Effective Date	Enter a date to associate an effective start date with the price pipeline
Expiration Date	Enter a date to associate an effective end date with the price pipeline

Field Name	Description
Pipeline Criteria	Define the criteria for the price pipeline. The price pipeline is used based on the criteria defined by the user. If the given criteria are not met for any line item, the price waterfall chart is not created. Define the criteria for Line Items and Product Attributes as follows. • For Line Items, click the icon > Line Items > Add New Criteria > enter the desired Field (Select the field that must be used for the adjustment), Operator, and Value > Save. • For Product Attributes, click the icon > Product Attributes > Add New Criteria > enter the desired Field (Select the field that must be used for the adjustment), Operator, and Value > Save. • For User Criteria, click the icon > User Criteria > Add New Criteria > enter the User Role, Operator, and Value > Save. Use the Advanced Filter Condition when multiple criteria are defined.
Define Price Points:	Define price points as follows. List price and Base Price are set by default. a. Click the text box in the name column and select the desired price point from the list. If the price point is not available, create a new price point. b. Is Modifiable: This feature allows you to modify the price point on the Waterfall page. For example, Suppose you enable the Is Modifiable option for a price point such as *Invoice Price. In that case,* the sales representative can only modify the *Invoice Price* of the product by adding manual adjustment to it on the Price Waterfall chart. c. Is this Net Price: This feature allows you to make a price point as a net price. Move the slider to enable the option. Any manual adjustments that the user performs on the cart appear on the waterfall chart against this price point (marked as the net price). d. Click **Add Price Point** to add a new price point.

4. Click **Save.**

ⓘ You must set at least one Price Point as a Net Price.

To Modify a Price Pipeline

1. On **App Menu**, click **Apttus CPQ Admin** and click the **Admin UI** tab. The new admin console is launched.

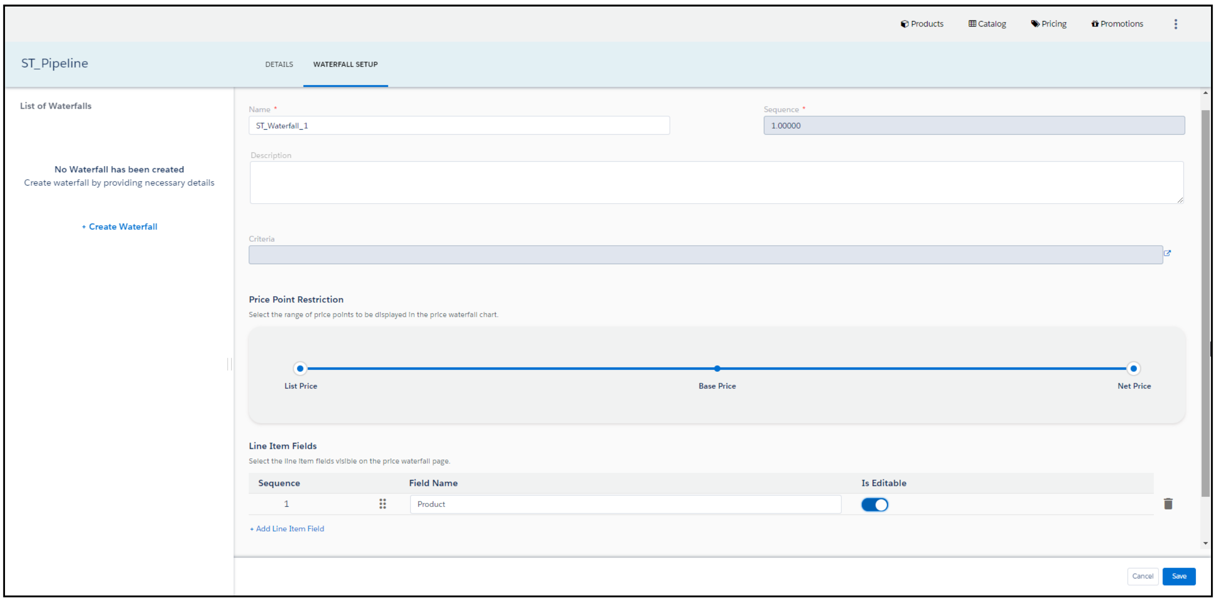
2. Select **Pricing** > **Manage Price Pipeline**.
3. Click the required Price Pipeline > Modify the details and click **Save.**

Configuring Price Waterfall to a Price Pipeline

After configuring the price pipeline, you must configure the price waterfall to a specific price pipeline.

To configure a price waterfall:

1. In the **App Menu**, click **Apttus CPQ Admin** and then click the **Admin UI** tab. The new admin console is launched.
2. Select **Pricing** > **Manage Price Pipeline**.
3. Click the required Price Pipeline > Click the **WATERFALL SETUP** tab.



4. Update the following details and click **Save**.

Field	Description
Name	Name of the price waterfall.
Description	Provide a description.
Criteria	Lists the criteria that are defined.

Field	Description
Price Point Restriction	Set the visibility restriction on price points. This option allows you to control the visibility of price points for the user. 
Line Items Fields	Select line item fields to display on the Price Waterfall page. a. Click the text box in the name column and select the desired Line Item field from the list. b. **Is Editable**: This enables you to modify the price point on the *Waterfall* page. For example, if you enable this option for the cart line item *Quantity,* then the sales representative can modify the quantity of the product on the waterfall chart and apply it to the price. Move the slider to enable the Is Editable option. c. Click **Add Line Item Field** to add a new field name. d. Click **Save**.

Configuring Price Pipeline Ruleset

Price Pipeline Ruleset allows pricing adjustments across a range of products through price rules. Price Pipeline Ruleset manages adjustments to specific price points, based on particular line item criteria or product attribute criteria. Price Pipeline rulesets are very similar to standard price rulesets, except that these contain price pipeline rules instead of the standard price rules. A price pipeline ruleset can contain one or more price pipeline rule entries and it can be dimensional, which employs a similar concept as Price Matrices. You can apply line item pricing adjustments as well as summary pricing adjustments.

To create a price pipeline ruleset

You must have an existing price list.

1. On **App Menu**, click **Apttus CPQ Admin** and then click the **CPQ Admin UI** tab. The new admin console is launched.
2. On the **Pricing** menu, click **Manage Price Pipeline Ruleset**.
3. Click **New Price Pipeline Ruleset**.
4. Enter details in one or more of the following fields, as required:

Field	Description
Price Pipeline Ruleset Name	Enter a mandatory price pipeline ruleset name and a mandatory sequence in which the system will evaluate multiple rulesets. Typically, you will perform line item adjustments first and then any summary adjustments
Is Active	Select this to set the ruleset as active.
Enable Currency	Select this to enable currency for the Price Pipeline Ruleset.
Currency	Select the currency for the Price Pipeline Ruleset. ⓘ If the Price Pipeline Ruleset and a line item have different currencies, CPQ applies the currency conversion rate (defined at **Setup** > **Company Profile** > **Manage Currencies**) when applying the rule adjustment.
Short Description	Enter a description for the ruleset.
Effective Date	Not required as long as the Price Pipeline Ruleset is Active, but can be used for promotional rules, and more.
Expiration Date	Select an expiration date.
Price List	Select a price list. The ruleset will only source product prices with the selected price list and then apply adjustment criteria.
Product Category	Select a category. The ruleset will only source product prices within the selected category.

Field	Description
Category	Select All, Agreement, or Proposal. This indicates if the ruleset is relevant to Agreements, Proposals, or both.
Application Level	Select a level to which this ruleset will be applied. The supported values are: • *Line Item*: Selecting this will apply the line item adjustment within the line item net price in the shopping cart. • *Bundle*: Selecting this will apply the adjustment to a bundle and its options and the adjustment is displayed in the Totals tab in the shopping cart. • *Aggregate:* Selecting this enables the **Application Method** field.
Application Method	This field is enabled if you selected *Aggregate* from the **Application Level** drop-down list. This indicates that you want the adjustment to select products in aggregate but apply an adjustment as a summary line in the Totals tab or spread the adjustment over numerous products. The supported values are: • *Apply to Line Items*: Applies the adjustment to line items on the cart. • *Create Summary Lines*: Applies the adjustment as a summary line in the Totals tab on the cart.
Ruleset Criteria	Allows you to set criteria for a line item rule or a bundle, depending on the Application level you have selected, such that the ruleset only applies when it satisfies a line-level field value or a product attribute value. Click **New** to fill in your criteria.
Enable Date Range	Selecting this enables you to set the effective date and expiration date on the rule entry.
Product Family	Select the Product Family. This is the Product Family field on the products object. The ruleset will only source product prices with the selected Product Family.
Product Group	Click to search and select a custom product group to the ruleset will apply the pricing adjustments. These custom product groups have no relation to a category, a Product Family, or any other product designation.

Field	Description
Charge Type	Select a charge type to which the ruleset will apply adjustments.

5. Click **Save**.

By filling out these criteria, the source products and prices are then designated and the ruleset can apply adjustments through price rules.

Creating Price Pipeline Rules

A *Price Pipeline Rule* enables you to provide target adjustments to specific price points in the price waterfall chart. Although the standard price rules are applied to the *Base Price*, you can now target any custom price point that you want by using the price pipeline rule. When defining any price pipeline rule, you must associate at least one price point to a price pipeline rule. These rules are will be executed (subject to fulfilling other criteria) and shown as adjustments specific to only those price points in the price waterfall chart

You can associate multiple pipeline rules to a price pipeline ruleset or through to a product. Price matrices can be defined in price rules as well as conditional price pipeline rules (rules that use Boolean logic such as AND/OR). The price pipeline rule can be dimensional, which uses a similar concept to Price Matrices or Conditional, that relates to fields for pricing adjustments.

One ruleset can have multiple rules and is evaluated in order of the rule sequence. A rule can be dimensional or conditional. A price pipeline rule determines the actual price adjustment made.

To create a price pipeline rule

You must have an existing Price Ruleset.

1. In the **App Menu**, click **Apttus CPQ Admin** and then click the **CPQ Admin** tab. The new admin console is launched.
2. On the **Pricing** menu, click **Manage Price Ruleset**.
3. Select a price ruleset.
4. On the **PRICE PIPELINE RULES** tab click **New Price Price Pipeline Rule**.
5. On the **DETAILS** tab, enter the following details.
    a. In **Rule Name**, enter a mandatory rule name.
    b. In **Short Description**, enter a description for the rule.

c. Click **Target Price Points** and select the price point from the list to associate with the rule.
d. From **Rule Type**, select a type. The supported values are *Dimension* and *Condition*.
e. From **Adjustment Applies To**, select an option to which the adjustment is applied. This is for line-item adjustments only. The supported values are List Price, Previous, and Prior Price Point.

Value	Description
List price	The source price for adjustment calculation is the List Price.
Previous	The source price for adjustment calculation is the current running price.
Prior Price Point	The source price for adjustment calculation is the prior price point.

f. By default, the price pipeline rule inherits the value of **Currency**, from the price ruleset.
g. The **Sequence** is auto-generated.
h. To make the rule active, select **Active**.
i. From **Allowable Action**, select an appropriate action for adjustments.
j. From **Adjustment Charge Type**, select an option to set adjustment as a charge type. You can change this in the price pipeline rule object.

6. Select the **DIMENSIONS** tab, enter the following details.
   a. From **Dimension**, select a price dimension.

      > ⓘ You can enter up to six dimensions, which bring in attributes from the line items or headers of any data type within Salesforce. For example, you can select a dimension on quantity.

   b. From **Dimension Value Type**, select **Discrete, Range, Cumulative Range,** or **Cumulative Range-Line Item.**
7. Click **Save**. The price pipeline rule is created and listed under the price pipeline rule list.
8. Click **Save**.

Working with Price Waterfall

The Price Waterfall page enables you to view and analyze the price for each line item. In addition to that, you can also make manual adjustments to the price points derived in the price pipeline to analyze the key performance indicators.

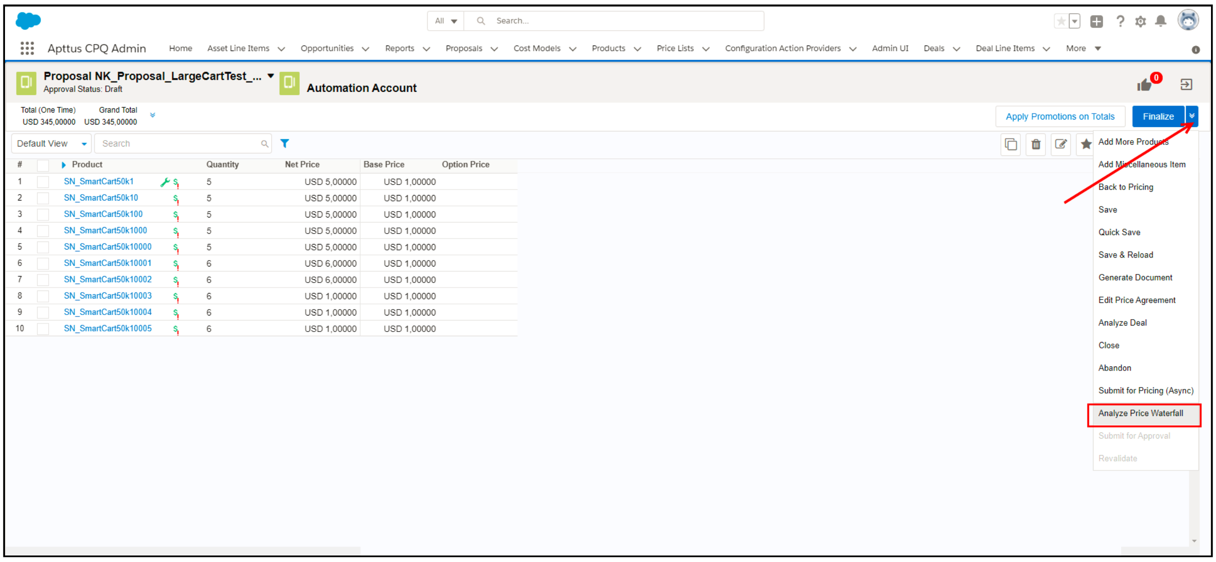
Launching Price Waterfall:

You can launch a price waterfall chart using one of the following.

- To Launch Price Waterfall from the Actions menu
- To Launch Price Waterfall for a specific Line Item

To Launch Price Waterfall from the Actions menu

1. Navigate to the Catalog.
2. Click **Add products**.
3. Click **Go To Pricing**. The Cart page is displayed.



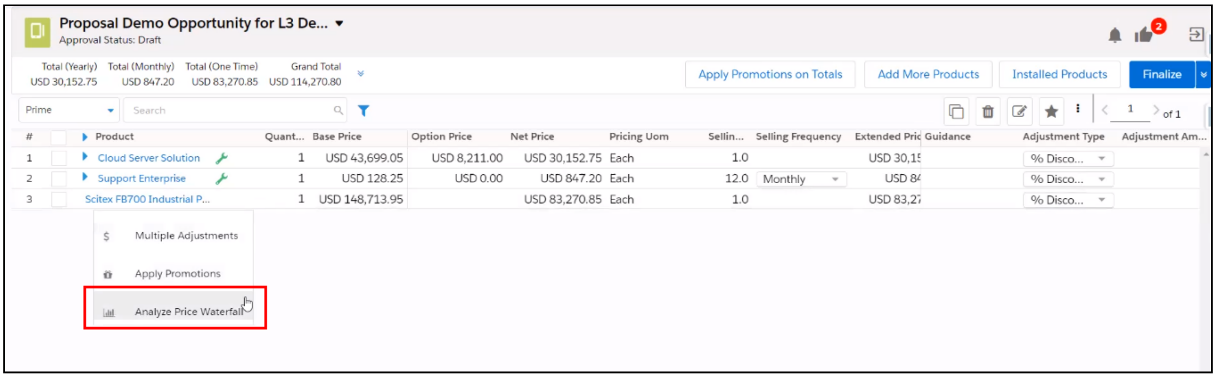
4. Click Analyze Price Waterfall. The Price Waterfall UI is displayed for the first line item.

	Field/ Icon/ Feature	Description
1	List of Line Items	Displays a list of Line items added to the cart and Enter a keyword to search a required product/option, and the search will show the product/parent bundle and its options.
2	Price Summary	The price summary of the cart line item.
3	Waterfall Chart	The waterfall chart is a bar graph where Line Item Fields are plotted vertically, and Cost is plotted horizontally.
4	Chart Options and Icons	**Per Unit and Gross tabs:** Per Unit displays price waterfall in terms of unit prices. Gross shows price waterfall in terms of the gross price.
		Menu icons to display the Price waterfall information: You can select whether to view price waterfall in the form of a bar chart ( ) or table ( ).
5	Action buttons	To Reprice or Finalize the cart.

	Field/ Icon/ Feature	Description
6	Line Item fields	The cart line item fields are displayed based on the waterfall setup to a price pipeline. You must enable the 'Is Editable' to a cart line item field to edit the value while analyzing the price waterfall chart. Working with cart line item fields: a. Enter the desired values in the editable cart line fields. b. Click **Reprice** or **Finalize**.

To Launch Price Waterfall for a specific Line Item

1. Navigate to the Catalog.
2. Click **Add products**.
3. Click **Go To Pricing**. The Cart page is displayed.
4. Place the cursor on the line item and click the kabab menu of desired line item.



5. Click **Analyze Price Waterfall**. The Price Waterfall UI is displayed for the specific line item.

Working with Manual Adjustments in the Price Waterfall UI

Suppose the price point is marked as modifiable in the price pipeline definition. In that case, you can make relevant manual adjustments to the particular price point in the

Create Manual Adjustment window. Any manual adjustments you perform on the cart appear against the price point marked as a net price.

To add manual adjustments:

1. Click the price point bar in the price waterfall chart.

   > You can click only if the price point is set as modifiable by the Admin in the price pipeline definition.

2. Enter the following fields and click **Apply** to view the updated price waterfall chart.

Field	Description
Name	Enter a name for the adjustment.
Adjustment Type	Select adjustment type from the drop-down list. You can only apply a *% Discount* as an Adjustment Type for the Net Price.
Amount	Enter the required adjustment amount.
Apply To	Select the source of price point from the following: • Previous: For the running price. • Prior Price Point: for the last point price. • List Price: For the List Price, which is the first price point in the waterfall chart.

Configuring Admin Entries

Admin Settings are system properties that have different values and you can use them for different purposes. Use the following admin settings, along with customs settings, to conform your implementation's behavior to your unique business requirements.

To create admin entries

1. Log in to Salesforce.

2. Click + and click **Admin.** All the out-of-the-box admin settings are displayed on the Admin home page. Based on your organization's requirements, you can add or create new Admin entries.
3. To create a new Admin entry, click **New.**
4. Type Name, Value, and Code for the admin setting and click **Save.** Your new admin entry is saved and added.

Admin Settings in TurboEngines

The following content provides information about admin entries available for TurboEngines along with their values and purposes.

APTS_RestoreLinesOnChange

Whenever you modify the line items after saving the cart, those modifications must display on the cart page whenever you launch the cart. To achieve this, you must configure the following admin setting to communicate line item changes between Salesforce and TurboPricing. The following property allows you to restore the line items when they are modified after saving the cart.

> For any reason, if you relaunch the cart when the turbo cart does not exist, CPQ ignores this setting and restores the full cart in the turbo flow. This setting becomes effective from the subsequent cart launch.

Name	APTS_RestoreLinesOnChange
Data Type	String
Value	• **True:** CPQ sends price pending line items to turbo pricing on cart launch. • **False:** CPQ restores the complete payload for all line items.
Code	N/A

TurboEngines Data Sync Documentation

Select one of the following topics for more information:

- About TurboEngines Data Sync
- What's New in Data Sync Documentation
- TurboEngines Data Sync for Administrators

About TurboEngines Data Sync

TurboEngines Data Sync service is a critical component of the **Conga TurboEngines** platform that provides a high-performance mechanism to sync master data at regular, scheduled intervals (or on-demand) between Conga CPQ on Salesforce and specific consumer endpoints using Conga's **Flexible Computing Platform**.

About TurboEngines

Conga TurboEngines is a concurrent processing engine provided by Conga that comprises various microservices that process product rules and configurations, pricing configurations and data, and other product-related business data. TurboEngine offloads the computation workload from the Salesforce platform to the Conga Flexible Compute Platform to reduce the processing time on the cart. Processing the computation workload in the Conga Flexible Compute Platform reduces the interaction costs and the quote turnaround time specifically during peak load or large transactions.

TurboEngines scales on 3 dimensions:

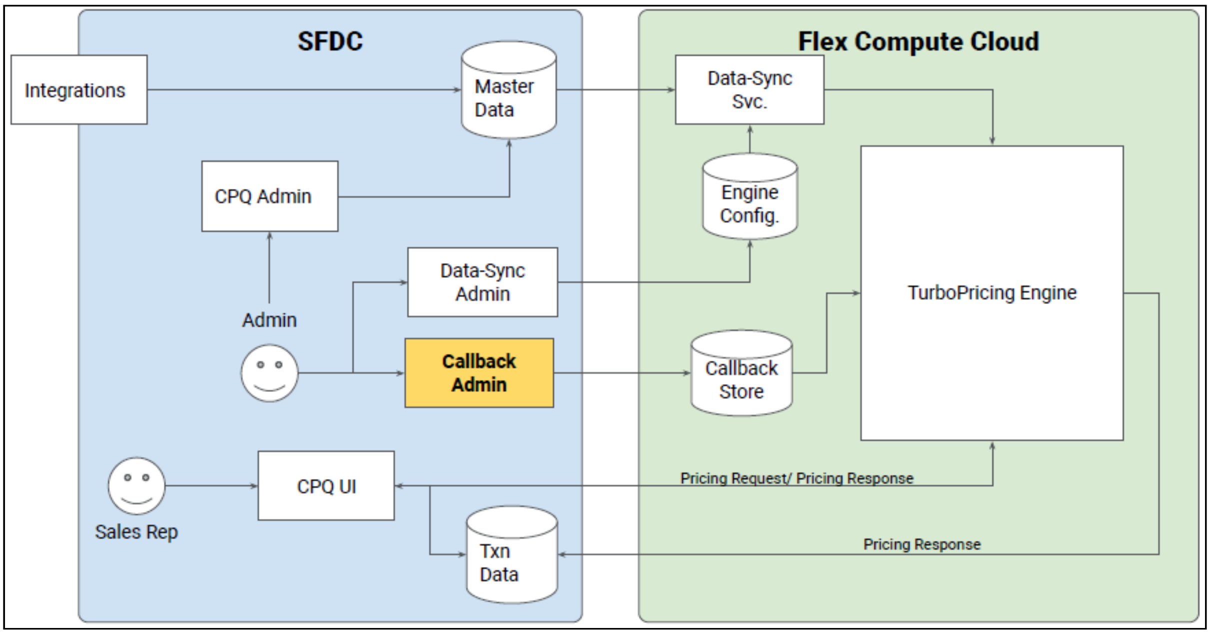
- Number of users
- Size of transaction
- The complexity of product and pricing configuration

For more information on how to get started with and configure **TurboEngines**, please refer to the *Conga TurboEngines Documentation* on the Conga documentation portal.

About the TurboEngine Data Sync Flow

TurboEngines Data Sync services comprise several components that work to pull data from Conga applications in Salesforce to a staging database that is then delivered using consumer services to various consumer endpoints (Consumer Profiles) based on their need for data. The data sync service pulls data from Salesforce at regular, scheduled intervals (using scheduler) and as needed based on Salesforce Push Topic configuration. A user can also retrieve data by invoking on demand sync from the Data Sync Admin UI.

Refer to the following diagram for a high-level flow of the data sync process between SFDC and TurboEngines.



About Consumer Profiles

A Consumer Profile as defined in Data Sync is essentially a master list and format definition for objects, fields, and the related objects and fields to be synced to a specific consumer endpoint. In the Data Sync Admin user interface, the Consumer Profile is defined as **Sync Settings**, comprising object and field data to be synced, details or indicators related to the profile, and sync frequency settings.

For more information on Consumer Profiles, refer to Navigating the Data Sync Admin User Interface.

Data Sync Prerequisites

Provisioning requirements for Conga TurboEngines must be met prior to managing data sync. Refer to Enabling TurboEngines in an Org in the *Conga TurboEngines Documentation* for more information.

Key Terms

Term	Definition
TurboEngines	A concurrent processing engine provided by Conga comprises various microservices that process and sync product configurations, pricing configurations, and data.
Flexible Computing Platform	A Conga-designed cloud platform built using microservices that offloads the computation workload from the Salesforce platform to reduce processing time on the cart, interaction costs, and the quote turnaround time specifically during peak load or large transactions.
TurboPricing	A pricing engine microservice on the Flexible Computing Platform that computes complex pricing computations and callbacks.
TurboConfig	A configuration engine microservice on the Flexible Computing Platform that computes complex product configurations and product rules.
Data Sync	The process is handled by TurboEngines Data Sync Services to sync master pricing data at regular, scheduled intervals or on-demand.
Run History	A log of all data successful and unsuccessful data sync executions provides a means of troubleshooting sync operations using helpful error messages.
Consumer Profile	A master list and format definition for objects, fields, and related objects and fields to be synced to various consumer endpoints.
Consumer Service	The service that delivers synced data to the consumer endpoint.
Consumer Endpoint	The destination for synced data is delivered by a Consumer Service and defined by the Consumer Profile.

What's New in Data Sync Documentation

The following table lists changes in documentation to support each release.

Document	Publication Date	Topic	Description
December '21 Rev A	18 Jan 2022	Criteria Based Sync	New Topic.
December '21	07 Dec 2021	Working with Data Sync Run History	Updated topic to • Add new status descriptions in the Run History and Run Details pages. • Add details about the Sync Record Details pop-up.
		Adding Objects and Fields for Sync	Updated topic to add details about, • Auto resync on selecting formula fields • API name column in the Add Object pop-up window
Summer '21	06 Jul 2021	Adding Objects and Fields for Sync	Modified topic based on changes to the Update Fields flows and changes in the Sync Settings UI.
		Creating Sync Indexes	Modified topic to add details for the sync index sequence.
		Working with Data Sync Run History	Modified topic to add details for the admin icon, resync tracking, new link (provides additional pricing master data sync details), in the Run History UI.
		Viewing and Evaluating Error Messages	Modified topic based on changes to downloading error messages in the Run History UI.
Spring '21 (Rev. A)		Adding Objects and Fields for Sync	Updated topic based on changes to the Manage Fields flow in the Sync Settings UI.
		Creating Sync Indexes	Updated topic based on changes to restrictions on Sync Indexes.

Document	Publication Date	Topic	Description
		Working with Data Sync Run History	Updated topic to add details for the Resync Sync Type.
		Data Sync Limitations	Updated topic to remove the limitation regarding syncing of future fields since the feature was removed.
Spring '21		NA	No updates.
Winter '20		Winter '20 Data Sync Administrator Guide	Updated topic to add a minor note about TurboConfig.
		About TurboEngines Data Sync	Updated topic to add a definition for TurboConfig.
		Navigating the Data Sync Admin User Interface	Updated topic to add consumer profile information for TurboConfig.
		Adding Objects and Fields for Sync	Updated topic to reflect changes to managing fields and related fields.
		Managing Email Notifications	New topic.
		Running Data Sync On-Demand	Updated topic to add a note about email notifications.
		Creating Sync Indexes	New topic.
		Enabling Objects for Push-Based Sync	New topic.
		Working with Data Sync Run History	Updated topic to add a note about email notifications and descriptions for new sync status values.

Document	Publication Date	Topic	Description
		Viewing and Evaluating Error Messages	Updated topic with new error messages.
		Retrying Data Sync	New topic.
		Data Sync Limitations	Updated topic name.
		Data Sync FAQ	New topic.
Summer 2020 (Rev. A)		Navigating the Data Sync Admin User Interface	Updated topic to reflect the name changed to the "TurboEngines Admin" app.
		Adding Objects and Fields for Sync	Updated topic to reflect minor changes to the simple and complex object tasks.
		Working with Data Sync Run History	Updated topic to include details for in-progress sync.
Summer 2020		All topics.	First release (internal only).

TurboEngines Data Sync for Administrators

This section provides Data Sync Administrators with the information required to manage master data sync for Conga Commerce implementations that have enabled **Conga TurboEngines** as the primary pricing and configuration engine.

Topic	Description
What's Covered	This guide provides information for administrators to manage Conga TurboEngines Data Sync.

Topic	Description
Primary Audience	• Conga Administrators • Conga Professional Services • Pricing Administrators • Customer Administrators
IT Environment	Refer to Getting Started for System Requirements and Supported Platforms.
Updates	For a comprehensive list of updates to this guide for each release, see the What's New in Data Sync Documentation topic.
Other Resources	• *Conga TurboEngines Documentation* • *Conga TurboEngines Release Notes* • *Conga CPQ Documentation*

This guide describes the following tasks:

- Managing Data Sync Settings
    - enabling and disabling data sync
    - adding objects and fields
    - adding sync actions and formats
    - running data sync
    - scheduling data sync
- Working with Data Sync Run History
    - viewing and evaluating run history
    - viewing error messages
- Troubleshooting Data Sync (known issues)

Before using TurboEngines Data Sync, you must be familiar with the following:

- Basic Salesforce administration
- Salesforce Lightning experience
- Salesforce and Conga terms and definitions
- Basic understanding of Conga TurboEngines
- Basic understanding of Conga TurboPricing
- Basic understanding of Conga TurboConfig

DOC ID: CTESUM21DSAG20210707

Navigating the Data Sync Admin User Interface

The TurboEngines Data Sync administrator user interface allows administrators to manage, modify, and monitor data sync operations of master data between Conga on Salesforce and various consumer endpoints. Administrators can make changes to data sync consumer profiles by adding, updating, or deleting objects and fields for sync. They can also monitor data sync jobs status and run history, as well as manually trigger data sync for any given consumer profile.

Navigating to TurboEngines Data Sync

To open the Data Sync Admin UI, go to **Salesforce App Launcher > TurboEngines Admin**.

Go to the **Data Integration** tab. From the list of Consumers, click **Manage Sync** to open the Admin UI.

> The **Callbacks** tab is used for managing TurboPricing Callbacks. Refer to the Conga TurboEngines December '21 Documentation for information on setting up and managing callbacks.

About Consumer Profiles

The standard and custom objects that are synced from Salesforce to TurboEngines are defined as Consumer Profiles. Consumer Profiles are a master list and format definition for objects, fields, and the related objects and fields to be synced to a specific consumer endpoint. These profiles are displayed in the Consumer list under the Data Integration tab. Consumer Profiles are classified into three Consumer Groups based on the consuming application (TurboPricing or TurboConfig).

TurboPricing Consumer Group

<table>
<tr><th>Consumer Profile</th><th>Data Sync Admin Access</th><th>Description</th></tr>
<tr><td>Pricing Master Data</td><td>Read / Write</td><td>Seeded master pricing data (Salesforce standard and Conga custom objects). This consumer profile can be modified from Sync Settings.

ⓘ The following objects will not be synced during initial sync and will be marked as Skipped in the Run History Detail page:
Complex_Apttus_Config2__PriceRule__c
Complex_Apttus_Config2__PriceMatrix__c
Apttus_Config2__ProductGroup__c
Complex_Apttus_Config2__PriceRuleset__c
Complex_Apttus_Config2__RelatedPriceListItem__c
Product2
These objects are synced as part of other objects in the form of denormalization.

ⓘ If the installed CPQ version is less than 13.1909 in org, remove the following fields from the Pricing Master Data consumer profile.
Apttus_Config2__PriceRuleset__c > Apttus_Config2__ParentPricePipelineId__c 2
Apttus_Config2__PriceRuleset__c > Apttus_Config2__StartPricePointId__c 3
Apttus_Config2__PriceRuleset__c > Apttus_Config2__EndPricePointId__c
Apttus_Config2__PriceRule__c > Apttus_Config2__PricePoints__c</td></tr>
<tr><td>Conversion Rates and Custom Settings</td><td>Read-only</td><td>Seeded custom settings for pricing. This consumer profile is read-only.</td></tr>
</table>

Consumer Profile	Data Sync Admin Access	Description
Deal Maximizer Master Data Tables	Read-only	Seeded master pricing data for Conga Deal Maximizer. You can modify the consumer profile from Sync Settings (no new objects can be added).

TurboConfig Consumer Group

Consumer Profile	Data Sync Admin Access	Description
Configuration Master Data	Read / Write	Seeded master product configuration data (Salesforce standard and Conga custom objects). This consumer profile can be modified from Sync Settings.

Extensibility

Consumer Profile	Data Sync Admin Access	Description
Customer-Specific Data	Read / Write	Seeded consumer profile with no data. This consumer profile must be modified from Sync Settings to add customer-specific data.

Overview of the Data Sync Admin UI

The TurboEngines Admin UI comprises two main tabs:

- **Sync Settings**: Manage objects and fields to sync for the selected consumer profile. You can view objects, fields and their hierarchy and make modifications as necessary. You can also define the frequency at which the data will be synced.

- **Run History**: View data sync run history and associated error messages. The tab displays a list of data sync Ids and other information, including whether or not the sync was successful. You can click on an entry in the run history to view any error messages or other details that are provided.

Managing Sync Settings

Refer to the topics in this section for step-by-step information on enabling, configuring, and executing data sync from Sync Settings in the Admin UI.

- Criteria Based Sync
- Enabling and Disabling Data Sync
- Adding Objects and Fields for Sync
- Creating Sync Indexes
- Running Data Sync On-Demand
- Managing Data Sync Frequency
- Managing Email Notifications

Criteria Based Sync

You can now enable the criteria-based sync for any entity. Criteria-based sync allows the administrators to decide which records should be synced for an entity. So, there is no need to spend time syncing the customer data, which is not required. The following are the key features available with the criteria-based sync.

- Ability to change the criteria at any time as the changes in filter criteria impact only the jobs that change the expiry date.
- Consumer API does not need to filter the records in their search APIs (Consumers need not worry about the criteria).

Criteria-based sync uses a condition that can be applied on the turboexpirydate__c field for the given Salesforce object. The data sync does not ship the turboexpirydate__c field for the Salesforce objects out of the box, and the consumer has to create this field or populate the required objects based on their filters. The turboexpirydate__c must be created with the same name, and it is case-sensitive. The turboexpirydate__c field should be an object with 'DateTime' and 'Indexed' to pull the records from Salesforce. In addition, the turboexpirydate__c field can have three values Null, Any Future Date, and Any Past Date.

DataSync Runtime would process the pulled records (from Salesforce) in the following manner. Currently, the records have been pulled based on the *'LastModifiedDate'* field.

- The record is considered *valid* and gets synced to the consumer-endpoint if the turboexpirydate__c field value is greater than the sync-cutoff-time.
- The record is considered *invalid* and gets deleted from the consumer-endpoint (if it exists) if the turboexpirydate__c field value is less than the sync-cutoff-time.
- The record is considered *ignored* if the turboexpirydate__c field value is Null.

The following table provides a summary of the scenarios to understand the outcome of criteria-based sync quickly.

Scenario	Criteria-Based Sync Outcome
The value of turboexpirydate__c is null during the earlier DataSync run, but it contains a future date and time for the current run.	The record gets synced to the consumer endpoint.
The value of turboexpirydate__c was past-DateTime during the earlier DataSync-Run, but it contains a future-DateTime for the current run.	The record gets synced to the consumer-endpoint.
The value of turboexpirydate__c was future-DateTime during the earlier DataSync-Run, but it contains a past date and time for the current run.	The record gets deleted from the consumer-endpoint.
The value of turboexpirydate__c was future-DateTime during the earlier DataSync-Run, but it contains null for the current run.	The record remains unchanged at the consumer-endpoint as data sync stops tracking that record if the *turboexpirydate__c* has a null value. Ensure that you avoid this kind of scenario as it leads to inconsistent data in the consumer point.

To enable criteria-based sync for an SFDC Object:

Assuming that the field turboexpirydate__c has been created and populated with the appropriate values. The following are the four possible use cases while enabling the criteria-based sync for an SFDC object.

1. The SFDC object is added to the customer profile for the first time.
2. The SFDC object is already present in the consumer profile and has gone through the **Initial Sync.**

3. The SFDC object is already present in the consumer profile and has not gone through the **Initial Sync**.
4. The SFDC object Is already present in the consumer profile and has gone through the **Initial Sync** using the existing filter-criteria-based sync.

<table>
<tr><th>S. NO.</th><th>Use Case</th><th>Recommended Actions</th></tr>
<tr><td>1</td><td>If the SFDC object is added to the customer profile for the first time</td><td>Invoke the following API to enable the object for criteria-based sync.
<table><tr><th>API Details</th><th>Example</th></tr><tr><td><code>POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/ enable</code>
This API enables the criteria-based sync fr the given object across all the consumer profiles. It is not possible to enable criteria-based sync for a specific consumer profile.</td><td><code>POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/enable</code></td></tr></table></td></tr>
</table>

<table>
<tr><th>S. NO.</th><th>Use Case</th><th>Recommended Actions</th></tr>
<tr><td>2</td><td>If the SFDC object is already added and has gone through the Initial Sync.</td><td>You can enable this object for Criteria-based sync without cleaning the existing data (Synced data). But, both the records (previously synced and newly synced) are stored in the consumer endpoint. Therefore, it is recommended to clean up the existing data of SFDC objects from both staging-DB and consumer endpoint before enabling the SFDC object for the criteria-based sync.

To clean up the existing data from DataSync's Internal DB, invoke the following API
<table>
<tr><th>API Details</th><th>Example</th></tr>
<tr><td><code>DELETE /ds/api/dataintegration/v1/SyncCriteria/</code>
<code>{entityName}/CleanDataForCriteriaBasedSync</code></td><td><code>DELETE /ds/api/dataintegration/v1/SyncCriteria/</code>
<code>{entityName}/CleanDataForCriteriaBasedSync</code></td></tr>
</table>

Clean the existing data from the consumer endpoint with the help of CloudOPS.
After cleaning up the existing data from both staging DB and consumer endpoint, invoke the following API to enable criteria-based sync for the given object.
<table>
<tr><th>API Details</th><th>Example</th></tr>
<tr><td>POST /ds/api/dataintegration/v1/SyncCriteria/
{entityName}/enable</td><td><code>POST /ds/api/dataintegration/v1/SyncCriteria/</code>
<code>entityName}/enable</code></td></tr>
</table>

</td></tr>
</table>

<table>
<tr><th>S. NO.</th><th>Use Case</th><th>Recommended Actions</th></tr>
<tr><td>3</td><td>The SFDC object is already present in the consumer profile and has not gone through the Initial-Sync</td><td>Invoke the following API to enable the object for criteria-based sync.
<table>
<tr><th>API Details</th><th>Example</th></tr>
<tr><td><code>POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/enable</code></td><td><code>POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/enable</code></td></tr>
</table></td></tr>
<tr><td>4</td><td>The SFDC object Is already present in the consumer profile and has gone through the Initial-Sync using the existing filter-criteria-based sync</td><td>The filter-criteria functionality remains the same without any impact even after the upgrading to the current release builds.
<blockquote>This scenario is applicable only for the tenants who have enabled Filter-based sync using the earlier builds. as that works differently than the current criteria-based-sync feature.</blockquote>

If the existing filter-criteria is similar to the filter that is being implemented in the pre-processing job, then you can remove the filter-criteria from the object by invoking the below API.
<table>
<tr><th>API Details</th><th>Example</th></tr>
<tr><td><code>DELETE /ds/api/dataintegration/v1/SyncCriteria/{consumerName}/SyncAction/{syncActionName}/FilterCriteria</code></td><td><code>DELETE /ds/api/dataintegration/v1/SyncCriteria/{consumerName}/SyncAction/{syncActionName}/FilterCriteria</code></td></tr>
</table>
Perform all steps mentioned in the use case-2.
</td></tr>
</table>

To disable the criteria-based sync for an SFDC Object

The following are the two possible use cases while disabling the criteria-based sync.

1. The object is present in the consumer profile and has not gone through the **Initial Sync.**
2. The object was earlier present in the consumer profile and has gone through the **Initial Sync** using **Criteria Based Sync**.

S.no	Use Case	Recommended Action
1	The object is present in the consumer profile and has not gone through the **Initial Sync.**	Invoke the following API `POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/disable`
2	The object was earlier present in the consumer profile and has gone through the **Initial Sync**.	Invoke the following API `POST /ds/api/dataintegration/v1/SyncCriteria/{entityName}/disable` After disabling the criteria-based sync for the SFDC object, clean up the already synced data by invoking the following API. `DELETE /ds/api/dataintegration/v1/SyncCriteria/{entityName}/CleanDataForCriteriaBasedSync`

To achieve criteria-based sync using Apex-jobs

Tenant Admin can exclude the invalid records from sync by setting the past date in the expiry date field. The expiry date is set based on the criteria condition.

1. Create a new turboexpirydate__c (case-sensitive) field with the appropriate date in the object on which you want to perform the Criteria-based sync.
2. Create a new custom metadata type.
   a. Log in to Salesforce.
   b. Go to **Setup** > **Custom Metadata Types** > **New Custom Metadata Type** and enter the following.

Field	Input
Label	DataSyncCutoffTimeList
Visibility	Public (All Apex code and APIs can use the type, and it's visible in Setup)
Object Name	DataSyncCutoffTimeList_mdt (auto-generated)

c. Click **Save.** A new custom metadata type field (*DataSyncCutoffTimeList)* page is displayed.

d. In the Custom fields section, click **New** and create the following custom fields.

Data Type	Field Lable
Text	EntityName__c
Date/Time	SyncCutoffTime__c

e. Create a new record as mentioned below.

Label	Product2
Name	Product2
EntityName__c	Product2
SyncCutoffTime__c	1900-01-01T12:00:00.000Z

3. Create a new Apex class.
   a. Go to **Setup** > **Apex Classes** > **New.**
   b. Enter the apex class code (ProductDataSyncPreProcessor2) as mentioned below.

> ⓘ The following Apex job code is just for reference purposes only. However, you can use this apex job code as a reference if your criteria are "Sync all products to turbo database if they have any price line item associated." For the products without Price Line Item association, the Apex job code updates the turboexpirydate__c field as the past date. Hence such products are not considered for syncing at the endpoints.

```
global class ProductDataSyncPreProcessor2 implements
 Database.Batchable<sObject>
{
    private DateTime newCutoffDate = null;

    /**
     * Constructor
     */
    public ProductDataSyncPreProcessor2(DateTime newCutoff)
    {
        this.newCutoffDate = newCutoff;
    }

    /**
    * Apex batch start method
    */
    global Database.QueryLocator start(Database.BatchableContext bc)
    {
        String strCurrentCutoffTime = newCutoffDate.formatGMT('yyyy-MM-
dd\'T\'HH:mm:ss.SSS\'Z\'');

        System.debug('strCurrentCutoffTime ::: ' + strCurrentCutoffTime +
'***');

        String soqlQuery =  'select Id ' +
                            'from   Product2 ' +
                            'where  turboexpirydate__c > ' +
strCurrentCutoffTime + ' ' +
                            '       and Id not in ( select
Apttus_Config2__ProductId__c ' +
                            '                       from
Apttus_Config2__PriceListItem__c ' +
                            '                       where
Apttus_Config2__ProductId__c != null)';

        System.debug('soqlQuery ::: ' + soqlQuery);
        return Database.getQueryLocator(soqlQuery);
    }

    /**
    * Apex batch execute method
```

```
    */
    global void execute(Database.BatchableContext BC, list<Product2> data)
    {
        Map<Id, Product2> prodList = new Map<Id, Product2>();

        for(Product2 sObj : data)
        {
            Product2 prod = new Product2(Id = sObj.Id);
            prod.turboexpirydate__c = DateTime.newInstance(2000, 1, 31, 14,
 0, 0);
            prodList.Put(prod.Id, prod);
        }

        if(!prodList.IsEmpty())
            update prodList.Values();
    }

    global void finish(Database.BatchableContext BC) {
        setLastSyncedTime('Product2', newCutoffDate);
    }

    public static void setLastSyncedTime(string objName, DateTime
cutoffTime)
    {
        List<DataSyncCutoffTimeList__mdt> historyRecs = [SELECT
EntityName__c, SyncCutoffTime__c, DeveloperName, MasterLabel
                                FROM    DataSyncCutoffTimeList__mdt
                                WHERE   EntityName__c = :objName
                                LIMIT 1];

        if (historyRecs != null && !historyRecs.isEmpty()) {
           DataSyncCutoffTimeList__mdt dsHistory = historyRecs[0];

            //create instance of Metadata.CustomMetadata
            Metadata.CustomMetadata metadataRec =  new
 Metadata.CustomMetadata();
            metadataRec.fullName = 'DataSyncCutoffTimeList__mdt.'+
dsHistory.DeveloperName;
            metadataRec.label = dsHistory.MasterLabel;

            //provide the value for the fields and add it to custom
metadata instance
            Metadata.CustomMetadataValue entityNameToUpdate = new
 Metadata.CustomMetadataValue();
```

```
                entityNameToUpdate.field = 'EntityName__c';
                entityNameToUpdate.value = objName;
                metadataRec.values.add(entityNameToUpdate);

                //provide the value for the fields and add it to custom
metadata instance
                Metadata.CustomMetadataValue cutoffTimetoUpdate = new
 Metadata.CustomMetadataValue();
                cutoffTimetoUpdate.field = 'SyncCutoffTime__c';
                cutoffTimetoUpdate.value = cutoffTime;
                metadataRec.values.add(cutoffTimetoUpdate);

                //Add the custom metadata instances in the container
                Metadata.DeployContainer mdContainer = new
 Metadata.DeployContainer();
                mdContainer.addMetadata(metadataRec);
                Id deployRequestId =
Metadata.Operations.enqueueDeployment(mdContainer, null);
                System.debug('deployRequestId $$$ '+deployRequestId);
            }
        }
}
```

c. Click **Save.**
d. Go to **Setup** > **Apex Classes** > **New.**
e. Enter the apex class code (ProductDataSyncPreProcessor1) as mentioned below.

> The following Apex code is just for reference purposes only.

```
global class ProductDataSyncPreProcessor1 implements
 Database.Batchable<sObject>
{
    private DateTime newCutoffDate = System.now();
    private Boolean isInitialRun = false;

    /**
     * Constructor
     */
    public ProductDataSyncPreProcessor1(Boolean isFirstRun)
    {
        this.isInitialRun = isFirstRun;
```

```
    }

    /**
    * Apex batch start method
    */
    global Database.QueryLocator start(Database.BatchableContext bc)
    {
        DateTime previousCutoffTime = getLastSyncedTime('Product2');

        String strCurrentCutoffTime = newCutoffDate.formatGMT('yyyy-MM-
dd\'T\'HH:mm:ss.SSS\'Z\'');
        String strPreviousCutoffTime = previousCutoffTime.formatGMT('yyyy-
MM-dd\'T\'HH:mm:ss.SSS\'Z\'');

        System.debug('strCurrentCutoffTime ::: ' + strCurrentCutoffTime +
'***');
        System.debug('strPreviousCutoffTime ::: ' + strPreviousCutoffTime
+ '***');

        String soqlQuery =  'select Apttus_Config2__ProductId__c ' +
                            'from   Apttus_Config2__PriceListItem__c ' +
                            'where
(Apttus_Config2__ProductId__r.turboexpirydate__c = null or
Apttus_Config2__ProductId__r.turboexpirydate__c < ' + strCurrentCutoffTime
+ ') ' +
                            '       and LastModifiedDate >= ' +
strPreviousCutoffTime + ' ' +
                            '       and LastModifiedDate <= ' +
strCurrentCutoffTime;

        System.debug('soqlQuery ::: ' + soqlQuery);
        return Database.getQueryLocator(soqlQuery);
    }

    /**
    * Apex batch execute method
    */
    global void execute(Database.BatchableContext BC,
list<Apttus_Config2__PriceListItem__c> data)
    {
        Map<Id, Product2> prodList = new Map<Id, Product2>();

        for(Apttus_Config2__PriceListItem__c sObj : data)
        {
```

```
            if(sObj.Apttus_Config2__ProductId__c != null)
            {
                //
sObj.put('Apttus_Config2__ProductId__r.turboexpirydate__c',
DateTime.newInstance(2099, 1, 31, 14, 0, 0));
                Product2 prod = new Product2(Id =
sObj.Apttus_Config2__ProductId__c);
                prod.turboexpirydate__c = DateTime.newInstance(2099, 1, 31,
 14, 0, 0);
                prodList.Put(prod.Id, prod);
            }
        }

        if(!prodList.IsEmpty())
            update prodList.Values();
    }

    global void finish(Database.BatchableContext BC) {
        System.debug('Executing ProductDataSyncPreProcessor1.Finish;'+isIn
itialRun+';'+newCutoffDate);
        if(isInitialRun)
        {
            setLastSyncedTime('Product2', newCutoffDate);
        }
        else
        {
            ProductDataSyncPreProcessor2 myBatchable = new
 ProductDataSyncPreProcessor2(newCutoffDate);
            Database.executeBatch(myBatchable, 10000);
        }
    }

    public static DateTime getLastSyncedTime(string objName){
        // Query custom metadata DataSyncCutoffTimeList__mdt as per the
object name
        DateTime timeToReturn = System.now();
        DataSyncCutoffTimeList__mdt dsCutOff = null;

        List<DataSyncCutoffTimeList__mdt> historyRecs = [SELECT
EntityName__c, SyncCutoffTime__c
                                FROM    DataSyncCutoffTimeList__mdt
                                WHERE   EntityName__c = :objName
```

```
                                LIMIT 1];

        if (historyRecs != null && !historyRecs.isEmpty()) {
           DataSyncCutoffTimeList__mdt dsHistory = historyRecs[0];
           if(dsHistory.SyncCutoffTime__c != null) {
              timeToReturn = dsHistory.SyncCutoffTime__c;
           }
        }

        return timeToReturn;
    }

    public static void setLastSyncedTime(string objName, DateTime
cutoffTime)
    {
        System.debug('Executing setLastSyncedTime;'+objName+';'+cutoffTime
);
        List<DataSyncCutoffTimeList__mdt> historyRecs = [SELECT
EntityName__c, SyncCutoffTime__c, DeveloperName, MasterLabel
                                FROM    DataSyncCutoffTimeList__mdt
                                WHERE   EntityName__c = :objName
                                LIMIT 1];

        if (historyRecs != null && !historyRecs.isEmpty()) {
           DataSyncCutoffTimeList__mdt dsHistory = historyRecs[0];

            //create instance of Metadata.CustomMetadata
            Metadata.CustomMetadata metadataRec =  new
 Metadata.CustomMetadata();
            metadataRec.fullName = 'DataSyncCutoffTimeList__mdt.'+
dsHistory.DeveloperName;
            metadataRec.label = dsHistory.MasterLabel;

            //provide the value for the fields and add it to custom
metadata instance
            Metadata.CustomMetadataValue entityNameToUpdate = new
 Metadata.CustomMetadataValue();
            entityNameToUpdate.field = 'EntityName__c';
            entityNameToUpdate.value = objName;
            metadataRec.values.add(entityNameToUpdate);

            //provide the value for the fields and add it to custom
metadata instance
```

```
            Metadata.CustomMetadataValue cutoffTimetoUpdate = new
 Metadata.CustomMetadataValue();
            cutoffTimetoUpdate.field = 'SyncCutoffTime__c';
            cutoffTimetoUpdate.value = cutoffTime;
            metadataRec.values.add(cutoffTimetoUpdate);

            //Add the custom metadata instances in the container
            Metadata.DeployContainer mdContainer = new
 Metadata.DeployContainer();
            mdContainer.addMetadata(metadataRec);
            Id deployRequestId =
Metadata.Operations.enqueueDeployment(mdContainer, null);
            System.debug('deployRequestId $$$ '+deployRequestId);
        }
    }
}
```

- ProductDataSyncPreProcessor1 and ProductDataSyncPreProcessor2 are interconnected and executed one after another.
    - ProductDataSyncPreProcessor1 is to update the turboexpirydate__c field of valid product records to a future date.
    - ProductDataSyncPreProcessor2 is to update the turboexpirydate__c field of invalid product records to a past date.
- In the first run, the ProductDataSyncPreProcessor1 is executed.
- In the second run, both ProductDataSyncPreProcessor1 and ProductDataSyncPreProcessor2 are executed.

4. Open the developer tool and execute the following statements.
    a. Run the following for the Initial Run:

```
ProductDataSyncPreProcessor1 myBatchable = new
 ProductDataSyncPreProcessor1(true); //Parameter 'true' is passed to
indicate that this is the first run
Database.executeBatch(myBatchable, 10000);  //The second parameter can be
altered based on the required batchsize.
```

    b. Run the following from the second run onwards:

```
ProductDataSyncPreProcessor1 myBatchable = new
 ProductDataSyncPreProcessor1(false); //Parameter false is passed to
indicate that this is not the first run
Database.executeBatch(myBatchable, 10000);  //The second parameter can be
altered based on the required batchsize.
```

Enabling and Disabling Data Sync

You can enable or disable data sync for a consumer profile from the **Sync Settings** tab.

To enable or disable data sync for a consumer profile

1. From the list of Consumers, click **Manage Sync** to manage data sync for the profile you want to enable or disable.
2. From the Sync Settings header, click **Sync Enable** toggle to enable or disable data sync. A message is displayed notifying you that the data sync is enabled or disabled.

> When data sync is disabled for a consumer profile:
> - The **Sync Now** button is disabled.
> - Any in-progress sync is not affected and completes as normal.
> - Any scheduled synchronizations are cancelled (note that the sync frequency setting for this profile is still set in the event the sync is re-enabled).
> - The admin can still modify objects and fields for sync.
> - The admin can still modify sync frequency.

Adding Objects and Fields for Sync

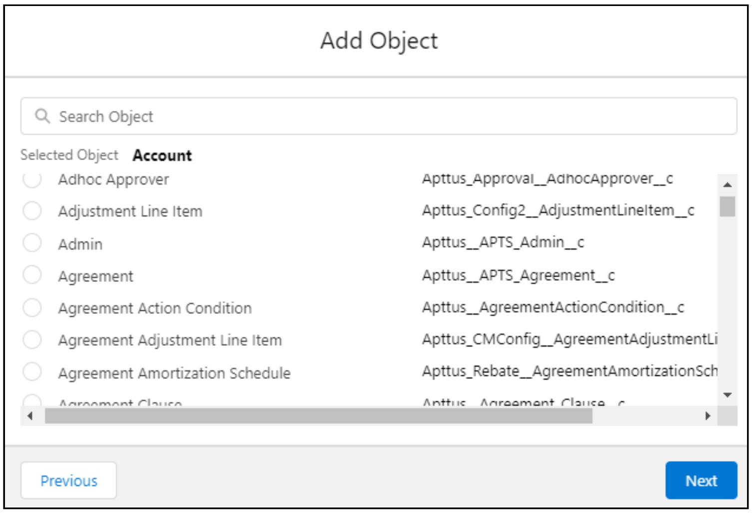
The initial list of objects to be synced is provided by the associated consumer profile and displayed on the Sync Settings page. When you need to add new objects or fields to the sync or update the existing data structure, you can do so from Sync Settings. You have the option to add simple (single object with no joins) or complex objects (single object with one or more joins). When adding new objects or managing existing objects, you can also select which fields to include or exclude from the sync operation.

> The Add Objects button is disabled for consumer profiles that are read-only.

To add a simple object to the sync

Add a simple object to the sync when you only need the object and its fields (included the Id of any reference fields).

1. Go to the **Sync Settings** page.
2. From the object list at the bottom of the page, click **Add Object**. The Add Object dialog is displayed.
3. Select **Simple Object**.
4. Click **Next**. An **Add Object** pop-up window is displayed with **Display Name** and **API Name** columns.

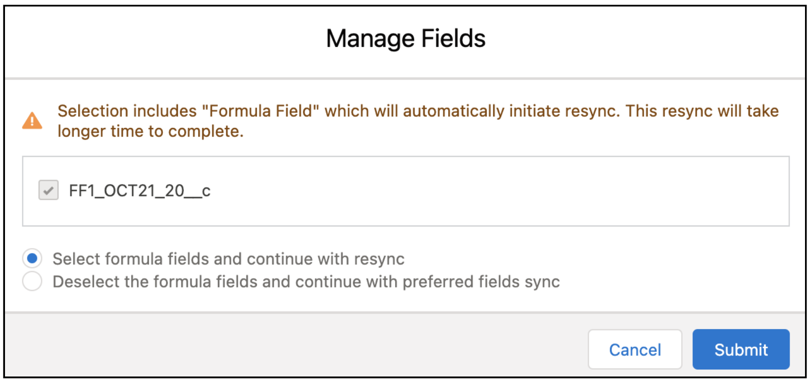


5. Search and select the object to sync. You can also search objects by API name.
6. Click **Next**.
7. Select **Data and metadata sync** to sync both object data and metadata, or select **Only metadata sync** to sync only metadata for the object.

> When you choose to sync only metadata, click **Save**. The sync profile is updated. No further action is required.

8. Click **Manage**. The Manage Object page is displayed.
9. Click **Manage Fields** to the right of the object name. The Manage Fields dialog is displayed.
10. Search and select one or more fields to add them to the sync. If you want to sync all fields for the selected object click checkbox to the left of Field Name in the table header. The Resync is initiated automatically if your select the formula fields.

> Formula fields are distinguished from non-formula fields by **[Data Type] (Formula Field)**.

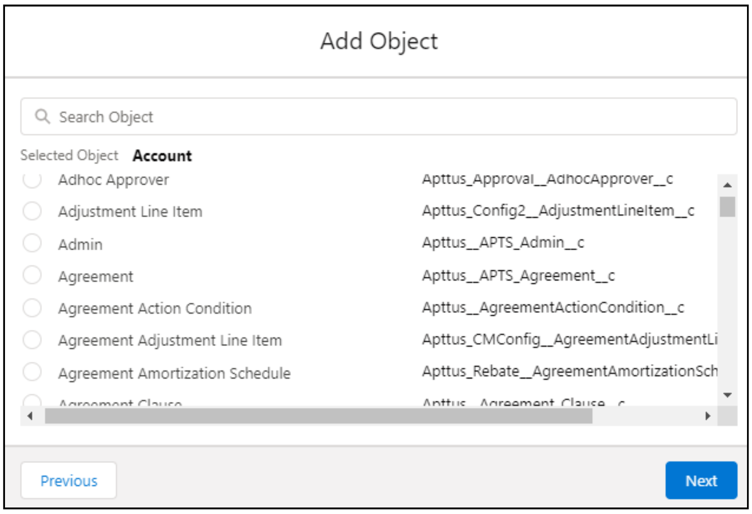


11. Select one of the following options as required and click **Submit**.
    - Select formula fields and continue with resync.
    - Deselect the formula fields and continue with preferred fields sync.
12. Click the chevron (>) next to the object name to expand the view to include the fields you just added.
13. Click **Submit**. The object is added to the list of objects on the Sync Settings page.

To add a complex object to the sync

Add a complex object to the sync when you want to include joins for related objects and their fields (reference, child, and nested relationships).

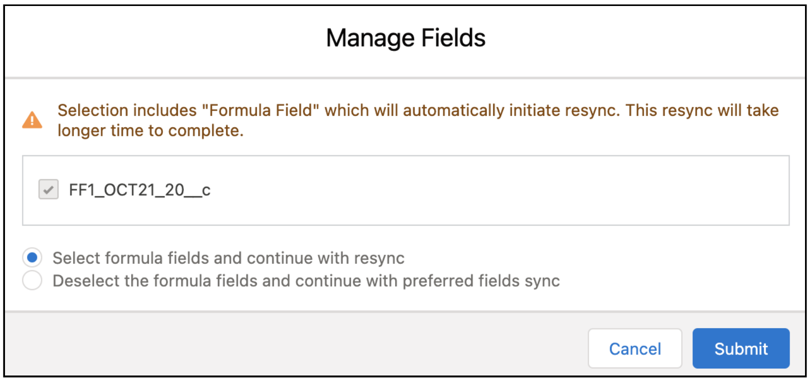
1. Go to the **Sync Settings** page.
2. From the object list at the bottom of the page, click **Add Object**. The Add Object dialog is displayed.
3. Select **Complex Object**.
4. Click **Next**. An **Add Object** pop-up window is displayed with **Display Name** and **API Name** columns.



5. Search and select an object to sync. You can also search objects by API name.
6. Click **Next**.

7. Enter the **Target Object Name**. This can be any user-friendly name for the object (do not include spaces).
8. Click **Manage**. The Manage Object page is displayed for the object you selected. By default, only the top-level object is displayed in the list.
9. Click **Manage Fields** to the right of the object name. The Manage Fields dialog is displayed.
10. Search and select one or more fields to add them to the sync. If you want to sync all fields for the selected object click checkbox to the left of Field Name in the table header. The Resync is initiated automatically if you select the formula fields.

    > Formula fields are distinguished from non-formula fields by **[Data Type] (Formula Field)**.

11. Select one of the following options as required and click **Submit**.
    - Select formula fields and continue with resync.
    - Deselect the formula fields and continue with preferred fields sync.
12. The Manage Objects page refreshes to display an expanded list of the object and fields to be synced.
13. To add reference or child objects, click **Manage Relations**. The Manage Relations dialog is displayed.
14. From the Reference Objects tab, search and select one or more lookup fields to add them to the sync. To sync all reference fields, click the checkbox to the left of Reference Object in the list header.
15. From the Child Objects tab, search and select one or more child objects to add them to the sync. To sync all child objects, click the checkbox to the left of Child Object in the list header.
16. Click **Submit** to add all selected reference and child objects.
17. Reference and child object fields can be added as joins to the nth level:
    a. To manage fields for a *reference* object you added, click the chevron (>) to the left of the reference field name and click **Manage Fields**. To manage additional reference and child objects, click **Manage Relations**.

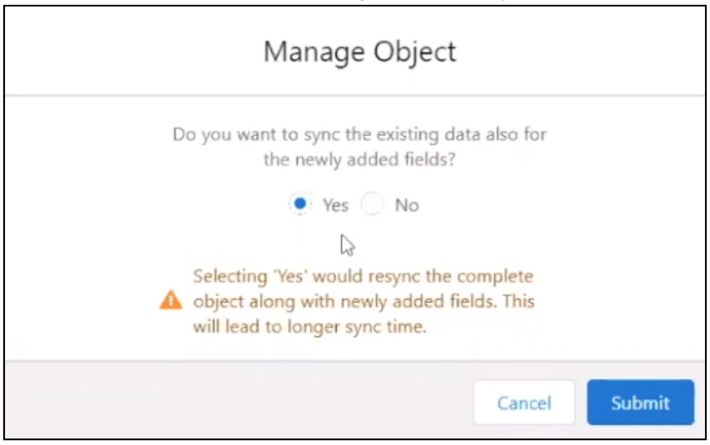
b. To manage fields for a *child* object you added click **Manage Fields** to the right of the object name. To manage additional reference and child objects, click **Manage Relations**.

18. Click **Submit**. Click the chevron ( ) to the left of a field name to view all fields, reference objects, and child objects you added for that field.
19. Click **Submit**. The object and its selected fields is added to the list of objects on the Sync Settings page.

To update an object for sync

Update objects when you want to include or exclude fields from the sync.

1. Go to the **Sync Settings** page.
2. Search for the object you want to modify.
3. Click the drop-down ( ) at the end of the row and select **Manage Object**. The Manage Object page is displayed.
4. Click **Manage Fields** to open the Manage Fields dialog. Click the checkbox next to any field in the list to enable or disable the field for sync and click **Submit**.
5. Click **Submit**. A new dialog is displayed.



6. Under **Do you want to sync existing data also for the newly added fields?** select Yes or No:
   - Select **Yes** to sync existing data for the entire object included any newly added fields.
   - Select **No** to decline the sync. The next sync will be run depending on any changes to fields marked for sync or the next scheduled or manual sync.

   > Be careful when choosing to resync the entire object as it will lead to a longer sync time.

7. Click **Submit** to return to the Manage Object page.

8. For complex objects, click **Manage Relations** to open the Manage Relations dialog. Click the checkbox next to any reference field or child object to enable or disable the field for sync. Click **Submit** to return to the Manage Object page.
9. Click **Submit** to save your changes and return to the Sync Settings page.
10. You can now trigger On-demand Sync to sync your changes to fields marked for sync.

To delete an object

1. Go to the **Sync Settings** page.
2. Search for the object you want to delete.
3. Click the drop-down at the end of the row ( ) and select **Delete**. A confirmation dialog is displayed.

   > You cannot delete seeded objects. You cannot delete fields marked with the lock ( ) icon.

4. Click **Yes** to confirm or **No** to cancel.

   > Be careful when deleting objects. To maintain sync integrity, delete child or reference objects first.

Enabling Objects for Push-Based Sync

You can enable a simple or complex object for Push-based sync. This takes advantage of PushTopic Events in Salesforce that notify the data sync service when one or more records have been created, updated, or deleted, or when changes have been made to a record based on a specific PushTopic query. When you enable an object for PushTopic sync, any time activity in Salesforce meets the criteria for a PushTopic Event, the corresponding object and its data will be synced for the applicable consumer profile.

To enable an object for PushTopic sync

1. Go to the Sync Settings page.
2. From the list of objects, search for the object you want to enable for PushTopic sync.
3. Select the drop-down at the end of the row ( ) and click **Enable Push Based Sync**. The Enable/Disable Push Based Sync dialog is displayed.
4. Click the **Push Based Sync** toggle to enable or disable the object for Push Based Sync.

5. Click **Submit**. The list of objects is updated and indicates the object's status under the Push Based Sync column. If enabled, the next time a PushTopic Event occurs, sync profile data is synced for Push Based Sync enabled objects.

> ⓘ Due to limitations imposed by Salesforce you cannot enable more than 50 objects for Push Based Sync. Please be aware of the following:
> - This is applicable across all consumer profiles. Meaning that any objects marked for Push Based Sync contribute to the overall limit.
> - For objects marked for Push Based Sync in multiple consumer profiles, the object only counts once towards the maximum.
> - For complex objects any child or reference objects also marked for Push Based Sync will be counted towards the maximum.
>
> In addition, if you exceed the Salesforce limit for PushTopic events in a 24-hour period (this is determined by your Salesforce edition – see link below) any objects marked for Push Based Sync will sync only as scheduled in the Data Sync Admin. After the 24-hour period expires (measured 24 hours from the initial PushTopic Event) Push Based Sync will resume until the limit is reached again.

For more information on Salesforce PushTopic Events, refer to the following documentation:

- Salesforce PushTopic overview: https://developer.salesforce.com/docs/atlas.en-us.api_streaming.meta/api_streaming/pushtopic_events_intro.htm
- PushTopic Event Allocations: https://developer.salesforce.com/docs/atlas.en-us.api_streaming.meta/api_streaming/limits.htm

Creating Sync Indexes

For objects with significantly large data sets, it may be good to consider creating a sync index so that pricing and configuration data for that object is more easily searchable after it has been synced to one or more consumer endpoints. You can set up indexes for any objects and specify fields that should be included in the index. You can configure indexes when you add an object for sync or when you manage any objects that are already part of the sync profile.

Sync indexes are most applicable for object pricing data subject to callbacks. For more information on configuring callbacks, refer to TurboEngines administrator documentation.

To add an index to a sync object

1. Go to the Sync Settings page.

2. From list of sync objects, search for the object and select the row drop-down (▾) and click **Manage Index**. The Manage Index page is displayed.
3. Click **Add Index**.
4. Enter a unique **Index Name** in the field provided.
5. Click **Next**.
6. Click the chevron (>) to the left of the object name to view the list of fields.
7. Click the check box next to a field name to add it to the index. Click the check box to the left of the Field Name column to select all visible fields. Note that child objects and reference objects can also be selected.
8. Click **Save**. The index is created and the index list is updated to display the new index. The Index Field column displays the added fields in sequence.
9. Click **Back** to return to the Manage Objects page.

Keep the following in mind when creating an index:

- After adding an index, you cannot make any modifications to it. If you need to make changes, delete and recreate the index.
- You can add a maximum of 32 fields (columns) to an index.
- You can create a maximum of 64 indexes per object.
- You cannot create an index for an object that is marked to "Include all fields including future fields" from the UI. See Data Sync Limitations for more information.

To delete a sync index

1. Go to the Sync Settings page.
2. From list of sync objects, search for the object and select the row drop-down (▾) and click **Manage Index**. The Manage Index page is displayed.
3. From the list of indexes, search for the index and select the row drop-down (▾) and click **Delete**. A confirmation dialog is displayed.
4. Click **Yes** to confirm the deletion or **No** to cancel.
5. Click **Submit** to save the updated index to Sync Settings.

Running Data Sync On-Demand

In cases where you need to initiate data sync manually, you perform on-demand data sync from the Sync Settings page.

Prerequisites:

- Data sync must be enabled for the given consumer profile.
- The consumer profile must include one or more simple or complex objects.

To run data sync manually

1. Go to the **Sync Settings** page.
2. Click **Sync Now**. The data sync process is begun and you are redirected to the Run History tab.
3. Review Run History to view sync details and any associated errors.

> - If sync is in progress when you invoke on-demand sync, the sync is queued and executes after the in-progress sync is complete.
> - If the sync fails for one or more objects, you can retry the sync. Refer to Retrying Data Sync for steps.

Managing Data Sync Frequency

For a given consumer profile, you can set and schedule master data to be synced at specified intervals. You can manage the frequency of data sync operations and set a start date and time for the first scheduled sync to begin.

To set the initial data sync frequency

1. Go to the **Sync Settings** page.
2. Click **Set New Frequency**. The Set Sync Frequency dialog is displayed.
3. From the **New Sync Frequency** drop-down, select the sync frequency.
4. To specify a start date and time for the first sync, click the checkbox next to **Schedule start date and time**.
    a. Select a date from the **Date** field.
    b. Select a time from the **Time** field.
5. Click **Submit**. The frequency for syncing the data for this consumer profile is set. If no date or start time was configured, the first sync is run. The Sync Settings page displays the newly configured frequency, any run currently in progress, and also the date and time of the next scheduled sync.

To change data sync frequency

1. Go to the **Sync Settings** page.

2. Click **Manage Frequency**. The Set Sync Frequency dialog is displayed.
3. From the **New Sync Frequency** drop-down, select the sync frequency.
4. Click **Submit**.

> ⓘ You cannot schedule a specific date and time for the next sync when you are updating a previously set frequency.

Managing Email Notifications

You can enable or disable email notifications from the Data Sync Admin UI. When enabled, an email notification is sent to the applicable recipients a sync fails or is partially successful ('Failed' or 'Partial Success').

> ⓘ **The From email corresponds to the email address specified in "My Email Settings" for the Salesforce profile associated with the Data Sync admin user.**

To configure email notifications

1. Go to the Sync Settings page.
2. From the upper-right hand corner of the Sync Settings tab, select the **settings drop-down** ( ) and click **Email Notification Settings**. The Email Notification Settings dialog is displayed.
3. Click the **Email Notification** toggle to enable or disable email notifications.
4. Enter a valid email address in the **Email** field and click the **Add** (+)icon. The email address is displayed below the field if added successfully. Click the delete icon (✕) to the right of an email address to remove it from the list.
5. Click **Submit**.

> ⓘ You must add at least one valid email address to enable email notifications.

Working with Data Sync Run History

After a manual or scheduled data sync is run, information about the sync is displayed in the **Run History** tab, regardless of whether or not the data sync was successful. You can view and interact with data sync run history to monitor the progress or status of the sync and any errors generated if the sync was unsuccessful.

The Run History table displays the following information.

Column	Description
Sync Id	The Id of a sync operation. Click the link to view object-level details.
Start Date and Time	The date and time the sync operation initiated. You can use the start date to understand how recently data was pulled from Salesforce to the consumer service.
End Date and Time	The date and time the sync operation completed or failed.
Duration	The duration of the data sync operation.

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>Status</td><td>
<table>
<tr><th>Status</th><th>Description</th></tr>
<tr><td>Success</td><td>The data sync operation was successful without any errors.</td></tr>
<tr><td rowspan="3">Partial Success</td><td>The data sync operation was successful for some objects but failed for one or more objects.</td></tr>
<tr><td>The data sync operation was successful for some objects, but sync failed and validation warning for one or more objects.</td></tr>
<tr><td>The data sync operation was failed, and the validation warning for one or more objects.</td></tr>
<tr><td>Success with Warning</td><td>The data sync operation was successful, but validation warning for one or more objects.</td></tr>
<tr><td>Failed</td><td>The data sync operation failed with one or more errors.</td></tr>
<tr><td>In Progress</td><td>The data sync operation is in progress for one or more objects.</td></tr>
</table>
You can click on the error ( ) icon to view a list of errors and suggested actions.

An email notification is automatically generated and sent to the appropriate administrator whenever sync fails or is partially successful. Email notifications are configurable and can be enabled or disabled. For more information, see Managing Email Notifications.
</td></tr>
</table>

Column	Description
Type	Indicates whether the sync was **Scheduled**, **Manual**, or **Resync**. A sync is labeled as a Resync in Run History when one of the following actions has triggered a sync: • An admin adds a new field for sync from the UI and selects "Yes" from the checkbox (to sync the entire object and any newly added fields). • A change in metadata is made to a formula field on the object or dependency object, and the field is currently marked for sync. • An Admin User icon ( ) is displayed next to the **Type** to identify the admin who triggered the corresponding data sync. You can click on the icon to display a pop-up with the admin user's username and email.

Only the most recent syncs are displayed on the page on your screen. To view older syncs, scroll down, and the list will refresh. You can also use the search function at the top to find a specific sync Id.

To view object-level details for a sync operation

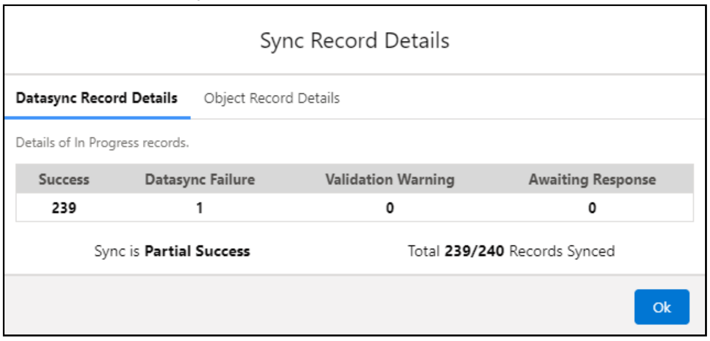
1. Go to the **Run History** page.
2. Locate and click the link to the corresponding **Sync Id**. The Details page is displayed. The Run History Details page displayed the following information.

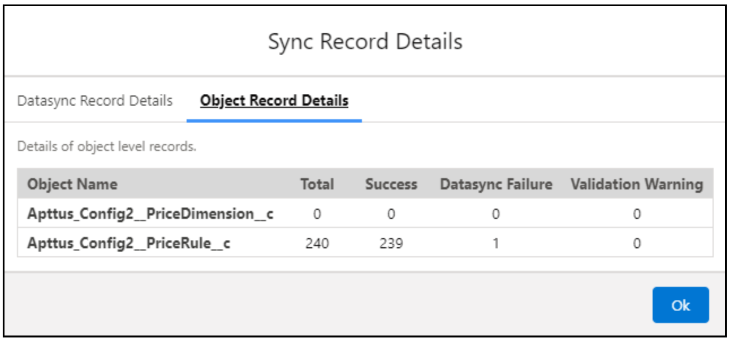
Column	Description
Object	The object of the record or records that were synced.
Type	The type of the object (Simple or Complex).

Column	Description
Status	The status of the synced object: • Preparing: The object has data that has been pulled from Salesforce but is not yet synced. • Successful: One or more records were successfully synced. • No Change: No changes were made to records for that object. • Partial Success: The sync was successful, but some records failed to sync. • Failed: The sync failed for the object – check for error message(s). • Aborted: An error occurred fetching object data from Salesforce – the operation to sync this object was not run. • Skipped: Sync was skipped for this object. This status is only present for certain objects in initial sync.

<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>Records Sync Status</td><td>
Displays the number of records processed for that object out of the total number of object records (for example, "50/50" Processed).
If there is any difference between the number of objects processed and the total number of objects, a Details link is displayed next to the Record Sync status.
you can click the details link to display a Sync Record Details pop-up with the following details.

Datasync Record Details: To view the details of In-progress records.
Object Record Details: To view the details of object-level records.





</td></tr>
</table>

Viewing and Evaluating Error Messages

The Run History tab allows you to view sync details for any sync and to investigate errors that have occurred for any during data sync operations in the list. From the list under Run History, you can view basic sync information and error details and suggested action to take. You can also drill down to the object level to view status for any object in the sync and download a list of error messages if needed for troubleshooting purposes.

To view and evaluate error messages in Run History

1. Go to the **Run History** tab.

2. Locate a **Sync Id** with the status **Failed**.
3. Click the **error** (⚠) icon. A dialog is displayed with a list of errors and suggestion actions to take.

The following table summarizes errors that can occur during a sync operation and suggested actions to take to resolve these errors.

Message	Suggested Action (Consumer Admin)
Unable to sync data to the Consumer Database.	Click "Sync Now" on the Sync Settings page to retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
An internal error has occurred with the Data Sync service.	Click "Sync Now" on the Sync Settings page to retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Unable to fetch Consumer Profile.	Contact Conga Technical Support with the Sync Id.
Unable to create bulk job on Salesforce.	Click "Sync Now" on the Sync Settings page to retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Unable to get initial results from Salesforce.	Please check that all objects in Sync Settings have read access. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Error occurred while retrieving the records from Salesforce.	Please check that all objects in Sync Settings have read access. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Unable to sync data to the target endpoint.	Click "Sync Now" on the Sync Settings page to retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Unable to fetch Data using the target object	Please contact Conga Technical support with the Sync Id.
Unable to sync data to the target endpoint.	Please contact Conga Technical support with the Sync Id.

Message	Suggested Action (Consumer Admin)
Unable to delete data from the target endpoint.	Please contact Conga Technical support with the Sync Id.
SFDC Api Limit exceeded	It may take up to 24 hours to restore SFDC Api limit. After the limit is restored retry the sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Metadata sync of an object failed	Retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).
Consumer profile sync of an object failed	Retry data sync. If the problem persists, please contact Conga Technical Support (you will need to provide the Sync Id).

To download a list of record-level errors

1. Click the **Sync Id** link for a sync with status 'Failed' or 'Partial Success'. The Run Details page is displayed.
2. From the list of objects synced, find the objects with Failed or Partial Success status.
3. Click the check box to include any errors from an object in the error report. Click the check box to the left of the Object column to select all such objects.
4. Click the **Error Messages** button at the top of the Details list to download a collection (*.zip* file) of error messages generated during the sync in JSON format. It generates the .zip file with *Consumer Profile Name_<Sync ID>_<DateTime>.zip* naming convention and It comprises the following files.
    - **Summary File:** Provides a summary of all the errors, reasons, number of records affected for each object, the corresponding number of columns, and resolution steps for each error.
    - **Details Error File:** Provides details at record level in a separate file. This JSON file is created for each object separately.

Retrying Data Sync

When an a sync fails or is partially successful, you have the option to retry the sync operation for each object that had one or more records fail to sync. After each retry, Run History is updated to reflect any additional records that were synced successfully.

To retry a failed sync

1. Go to the Run History page.
2. Find the sync that failed or was partially successful.
3. Click the **Sync Id** to open the object details page.
4. Click the check box next to an object in the list with the status 'Failed' or 'Partial Success'. Repeat for each object you want to retry. You can select all such objects in the list by clicking the check box to the left of the Object column in the header.
5. From the Details page header, click **Retry**. The on-demand sync executes and all selected objects are updated to 'In Progress' status.

   > ⓘ You cannot retry a sync with status "Aborted".

6. After the on-demand sync finishes, the Status field is updated for each object along with the number of records synced. If the sync retry is successful for all objects, the Status field on the Run History page is also updated.

   > ⓘ Note: When you retry a failed sync, no new entry is created on the Run History page. The entry for the original sync is updated to reflect any changes.

Appendices

Refer to the following appendices for additional important information about TurboEngines Data Sync.

- Data Sync Limitations
- Data Sync FAQ

Data Sync Limitations

The following known issues apply to TurboEngines Data Sync up to the current release. Please contact Conga Technical Support with any additional questions:

Issue: Adding Cross-Object Formula Fields

Data sync only supports up to eight levels in a cross-object formula field. Any levels beyond eight will be ignored.

Issue: Some Out-of-the-box Objects and Fields Can Cause Sync to Fail

Assumptions: Some objects exist in the list of objects to add that do not support Salesforce "PK chunking" for bulk data queries.

Scenario: Some SFDC out-of-the-box objects in the "Add Objects" list under Sync Settings could cause the data sync to fail if they are added to the consumer profile. There is no effect on custom objects (Conga and customer-specific) as they all support "PK chunking."

Workaround: As a workaround Conga can add objects as they are needed from the back end. Please contact Conga Technical Support if you have any questions regarding this limitation.

Issue: Deleting a Consumer Profile Field in SFDC

Assumptions: Changes are made to the data model for a consumer profile object in SFDC.

Scenario: When a field or related field of an object that is part of a consumer profile is removed, data sync will mark that field as inactive. If it is added back again with the same name, the status of the field remains inactive.

Workaround: Contact Conga Technical Support.

Data Sync FAQ

Q: How long does it take for the next scheduled data sync to complete?

A: This depends on the frequency and the volume of data being completed. The delay between syncs will always be based on the scheduled frequency plus the amount of time to sync the data.

Q: When I delete a field, reference, or object in Sync Settings, is the same data deleted from the consumer endpoint?

A: When you delete a field, reference, or object in Sync Settings, it is no longer enabled for sync. This has no effect on the data at the consumer endpoint other than that any new data for that field or object will no longer be synced.

Q: I have already added an object in sync settings but I want to add one or more fields? How can I be sure that the field will be synced the next time the sync is run?

A: Use the "Manage" feature for an object in sync settings to add additional fields or reference and child fields to the object and save. The next time the sync is run, existing data for those fields are synced to the consumer endpoint.

Q: Can I set up email notifications to send an email to the administrator when a sync completes successfully?

A: No. Email notifications are only sent on partial success or failure of a sync.

Q: As an end-user, I have changed the data and I want to quote the same in Turbo immediately?

A: No. Please check your data sync scheduler. You must complete the sync before using the Turbo functionality for quoting.

REST API Guide

Select one of the following topics for more information:

About REST API Guide

This guide provides an API reference for the TurboConfig Publisher API. Use this API to sync product configuration data from Salesforce to the TurboConfig product configuration engine for processing. You can sync product data to TurboConfig by bundle Id or category Id.

Topic	Description
What's Covered	This guide provides a reference for the TurboConfig Apttus Publisher REST API provided by Apttus.
Primary Audience	• Apttus TurboEngines Administrators • API Developers • Customer Administrators
IT Environment	Refer to the latest *Apttus TurboEngines Summer 2020 Release Notes* for information on System Requirements and Supported Platforms.
Updates	For a comprehensive list of updates to this guide for each release, see the What's New topic.
Other Resources	• *Apttus TurboEngines Summer 2020 Administrator Guide* • Apttus *CPQ on Salesforce Administrator Guide.*

This guide describes the following task:

- Syncing product data using TurboConfig Publisher API.

Before using TurboEngines, you must be familiar with the following:

- Understanding of REST architecture
- Knowledge of REST request and response payloads and operations

- Knowledge of Salesforce API calls
- Salesforce and Apttus terms and definitions
- Basic understanding of Apttus TurboConfig

DOC ID: CTESPR21APIG20210303

What's New in API Guide

The following table lists changes in documentation to support each release.

Document	Publication Date	Topic	Description
December '21	07 Dec 2021	N/A	No new APIs were introduced in this release. The guide was updated to reflect product name changes.
Summer '21	06 Jul 2021	N/A	No new APIs were introduced in this release. The guide was updated to reflect product name changes.
Spring '21		N/A	No new APIs were introduced in this release. The guide was updated to reflect product name changes.
Winter 2020		TurboConfig Publisher	Deleted Topic. A new Data Sync Service is introduced in this release. It provides a high-performance mechanism to sync master data at regular, scheduled intervals (or on-demand) between Conga CPQ on Salesforce and specific consumer endpoints using Conga's **Flexible Computing Platform**.

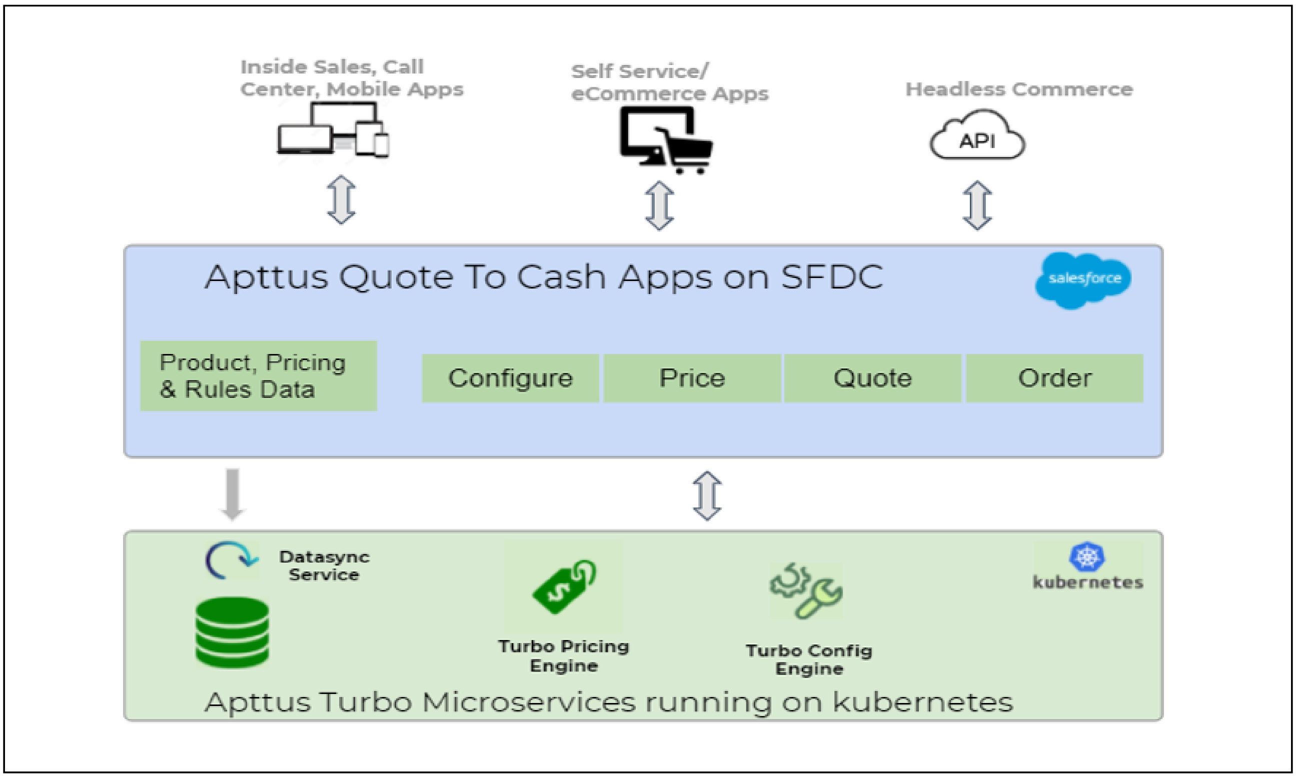
Document	Publication Date	Topic	Description
Summer 2020		All topics.	First release.

About TurboEngines - Rest API

Conga **TurboEngines** is a concurrent processing engine provided by Conga that comprises various microservices that process product configurations (**TurboConfig**), pricing calculations (**TurboPricing**), and other product-related business data, such as promotions. Conga TurboEngines offloads the computation workload from the Salesforce platform to the **Conga Flexible Compute Platform** to reduce the processing time on the cart. Processing the computation workload in the Conga Flexible Compute Platform reduces the interaction costs and the quote turnaround time specifically during peak load or large transactions.

TurboEngines scale on the following dimensions:

- Number of users
- Size of transaction
- The complexity of the product and rules



Turbo Data Sync APIs

(Open API documentation is only available to view online)

TurboEngines Feature by Release

Review the latest TurboEngines Features by Release document.

- Features by Release

Features by Release

This document contains an overview of features introduced in each major release of Conga TurboEngines. For more information, see TurboEngines Features by Release.

Conga Customer Community & Learning Center Resources

The Conga Customer Community is your one-stop shop for success!

After registering as a new member, you'll gain access to a wide variety of resources, from a personalized onboarding checklist to free expert-led webinars, to our thought-leadership blog! It's also your portal to manage your Conga account, access the Install Center, and submit support tickets.

You can also access the Conga Learning Center. All customers get access to a limited catalog of getting started courses. Consider upgrading to the Conga Learning Pass to unlock the premium training subscription.

Ready to get started?

Log into the Conga Customer Community with your credentials.

Not yet registered? No problem. Set up an account to receive your login credentials via our registration page.

After you log in, there are 2 ways to access the Conga Learning Center:

- On the main page, click the Learning Center tile.
- Navigate to the "Resources" dropdown menu at the top, then click the "Learning Center" option.