Oracle Fusion Cloud Sales Automation

How do I configure the Sales dashboard?

Oracle Fusion Cloud Sales Automation
How do I configure the Sales dashboard?

F90641-03

Author: Jiri Weiss

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Use help icons ⓘ to access help in the application. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessibility Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, suggest *ideas* for product enhancements, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 Before You Begin

Before You Configure the Sales Dashboard

Before you can configure the Sales Dashboard, you must first set up Visual Builder Studio and configure it. You only need to complete these steps once per implementation.

For a complete list of implementation steps, see *Before You Modify Pages in Visual Builder Studio*.

Enable Opportunity Pipeline Attributes in Adaptive Search

For the Sales Dashboard to display values in the Opportunity Pipeline amount, Win Probability, and Amount fields, the following 3 fields must be enabled in Adaptive Search. These fields aren't enabled by default. After you enable the fields, you must run the Partial Publish process to activate your changes.

You enable the fields in the **Setup** > **Advanced** tab of the **Configure Adaptive Search** setup task:

- Primary Revenue

  (Navigation: **Opportunity** > **Primary Revenue** > **Primary Revenue**)
- Win Probability

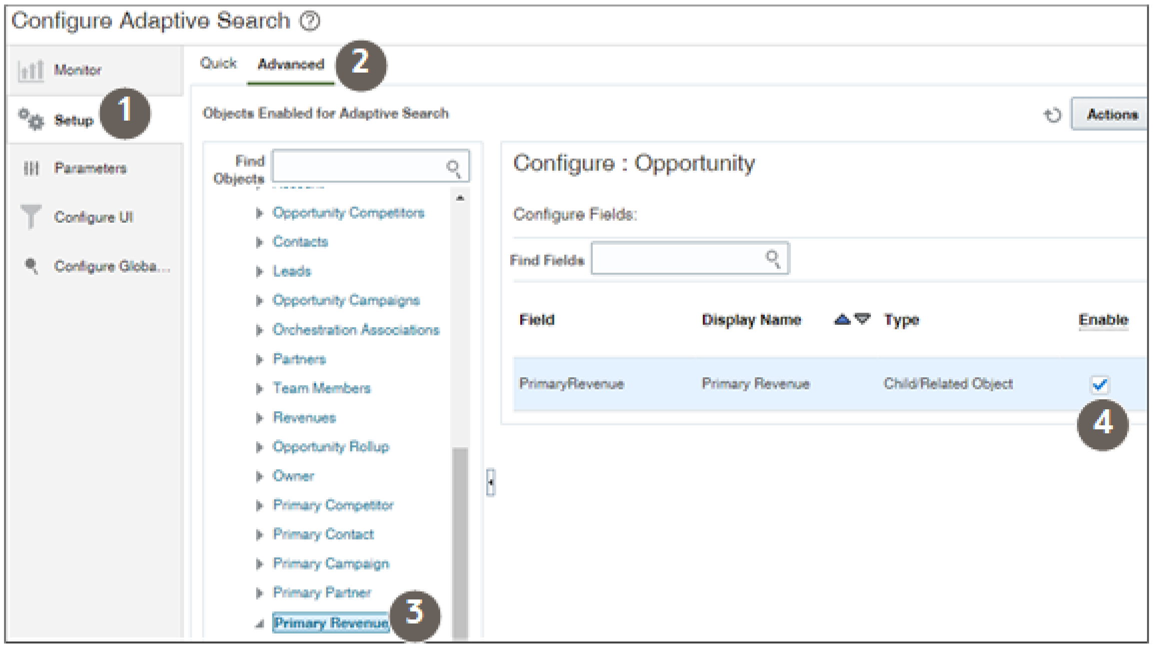
  (Navigation **Opportunity** > **Primary Revenue** > **Opportunity Revenue** > **Primary Revenue** > **Win Probability**)
- Amount

  (Navigation: **Opportunity** > **Primary Revenue** > **Opportunity Revenue** > **Amount**)

Here are the detailed steps:

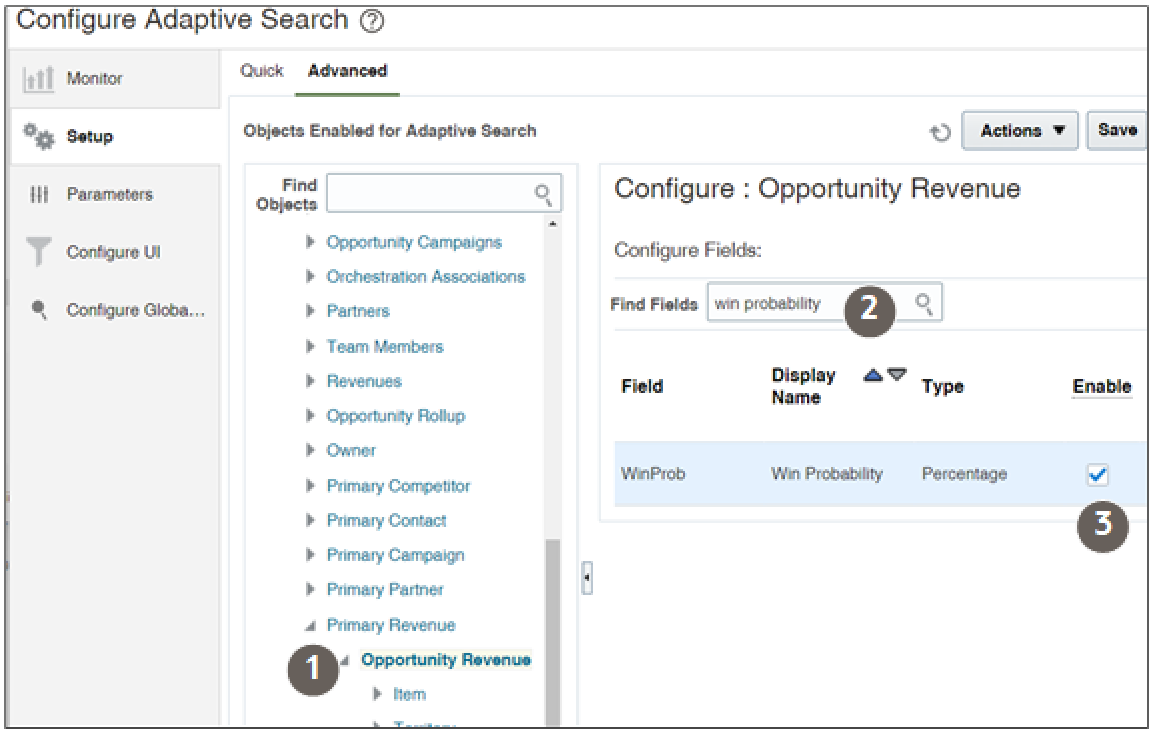
1. In the Setup and Maintenance work area, open the **Configure Adaptive Search** task:
   - Offering: Sales
   - Functional Area: Sales Foundation
   - Show: All Tasks
   - Task: Configure Adaptive Search
2. Click **Setup** > **Advanced**.
3. On the Advanced tab, click **Primary Revenue**.

4. In the Configure: Primary Revenue pane, select the **Enable** checkbox for the **Primary Revenue** field.



Field	Display Name	Type	Enable
PrimaryRevenue	Primary Revenue	Child/Related Object	✓

5. Click **Primary Revenue** > **Opportunity Revenue**.



Field	Display Name	Type	Enable
WinProb	Win Probability	Percentage	✓

6. In the Configure: Opportunity Revenue pane, use the **Find Fields** field to search and enable the **Amount** field and the **Win Probability** field.
7. Click **Save**.

8. Publish the new attributes by running the Partial Publish process:
    a. Click **Actions** > **Partial Publish**.
    b. Select **Opportunity**.
    c. Click **Proceed with Partial Publish**.
    d. In the Partial Publish window, click **Publish**.

2 Visualization Configurations

Access Visualization Configurations

Visualization configurations are viewed in the *mobile application*, or on the *sales dashboard*.

The following roles have permissions to work with visualizations.

- ORA_ZCA_CUSTOMER_RELATIONSHIP_MANAGEMENT_APPLICATION_ADMINISTRATOR_JOB (Customer Relationship Management Administrator)
- ORA_ZBS_SALES_ADMINISTRATOR_JOB (Sales Administrator)

How to Access the Visualization Tool

1. From the Navigator, choose **Configuration**.
2. Choose **Application Composer**.
3. Choose **Visualization Configuration**. If you don't see a visualization configuration option, then verify that your role has the proper permissions.

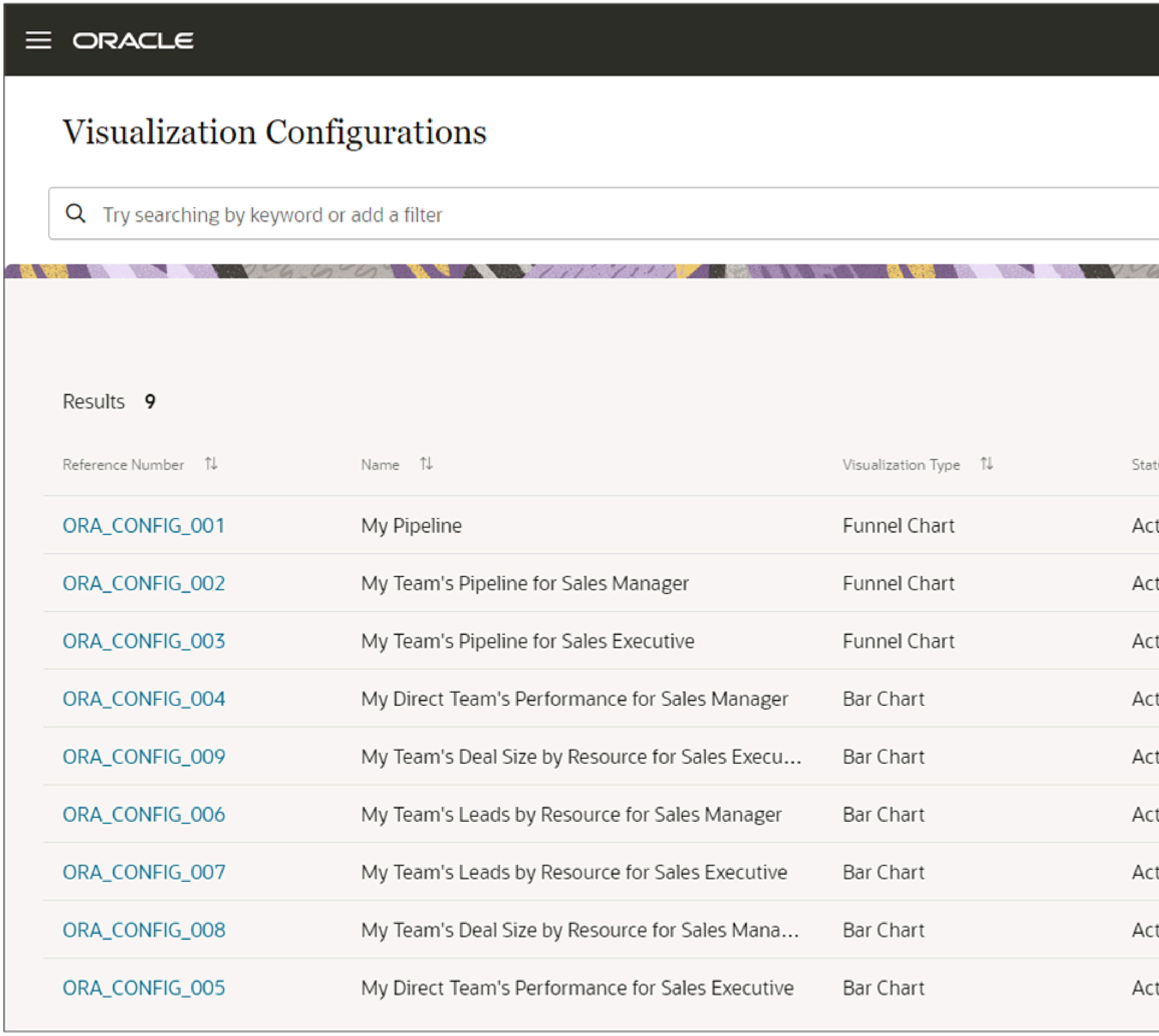
View a Visualization Configuration

On the Visualization Configuration page, you see a list of all the configurations that administrators have built, as well as predefined configurations.

How to View a Visualization Configuration

Click the visualization reference number to see the details of that configuration.

If an administrator doesn't have transactional data that they own, visualizations for predefined configurations might show a warning message that the source doesn't return data. This could also be validated in the source analysis results.



Visualization Configurations

Try searching by keyword or add a filter

Results 9

Reference Number	Name	Visualization Type	Stat
ORA_CONFIG_001	My Pipeline	Funnel Chart	Act
ORA_CONFIG_002	My Team's Pipeline for Sales Manager	Funnel Chart	Act
ORA_CONFIG_003	My Team's Pipeline for Sales Executive	Funnel Chart	Act
ORA_CONFIG_004	My Direct Team's Performance for Sales Manager	Bar Chart	Act
ORA_CONFIG_009	My Team's Deal Size by Resource for Sales Execu...	Bar Chart	Act
ORA_CONFIG_006	My Team's Leads by Resource for Sales Manager	Bar Chart	Act
ORA_CONFIG_007	My Team's Leads by Resource for Sales Executive	Bar Chart	Act
ORA_CONFIG_008	My Team's Deal Size by Resource for Sales Mana...	Bar Chart	Act
ORA_CONFIG_005	My Direct Team's Performance for Sales Executive	Bar Chart	Act

Create Visualization Configurations

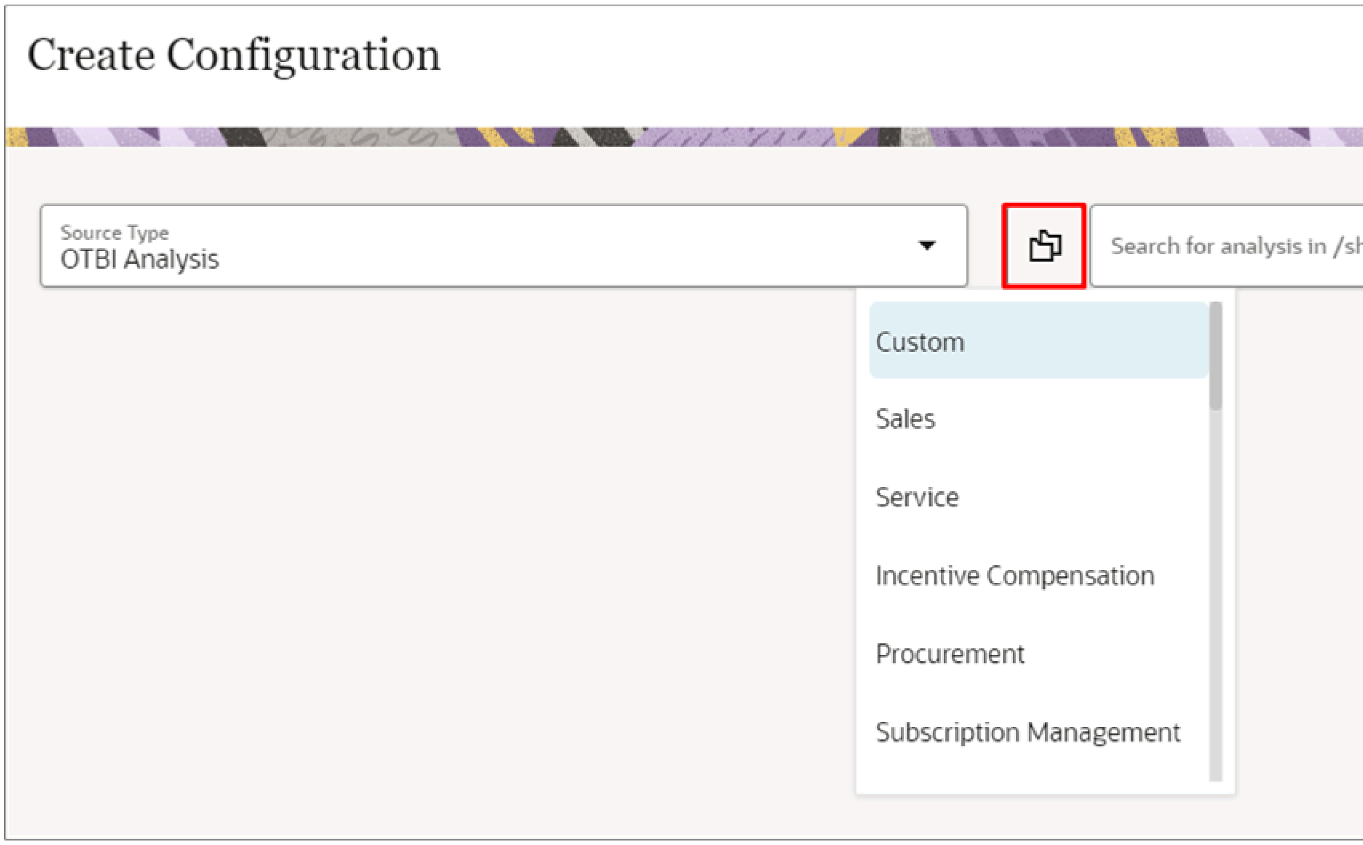
You can create a visualization for an *OTBI analysis* or for a saved search.

You must be a setup user or a user with one of the following job roles to create a visualization:

- ORA_ZCA_CUSTOMER_RELATIONSHIP_MANAGEMENT_APPLICATION_ADMINISTRATOR_JOB (Customer Relationship Management Administrator)
- ORA_ZBS_SALES_ADMINISTRATOR_JOB (Sales Administrator)

How to Create a Visualization Configuration From the Catalog

1. Navigate to **Configuration** > **Application Composer**.
2. Click **Visualization Configuration**.
3. Click **Add** to create a new configuration. The Create Configuration page displays the source type and source.
4. To create a configuration from an OTBI analysis:
   a. Select **OTBI Analysis** as Source Type.
   b. Select a folder by clicking the catalog icon and navigate to where you have saved the OTBI analysis. By default, the Custom folder is selected.



Create Configuration

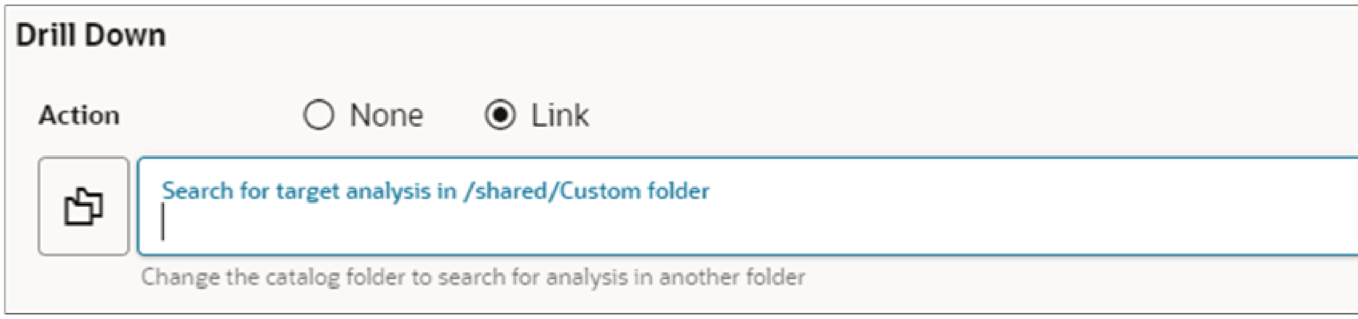
Source Type
OTBI Analysis

Search for analysis in /sh

Custom
Sales
Service
Incentive Compensation
Procurement
Subscription Management

5. In the Source field, enter your search criteria. You're searching the catalog for an existing analysis under the selected catalog folder.
6. Select a source analysis from the search results. All the fields get auto-selected and the visualization shows on the page. If the selected source analysis has more columns than required, there could be a warning message that the visualization might not show the correct data. To resolve this issue, remove the columns that aren't used and reload the configuration.
7. For the Sales Dashboard only, if you want to enable drill-down functionality, then you need to take a few more steps:
   a. Select a folder by clicking the Select Catalog folder icon where you've saved the Target OTBI analysis.

b. In the Target field, enter your search criteria. You're searching the BI catalog for an existing target analysis from the selected catalog folder. The target analysis should have a prompted filter applied on the same field as selected in the source analysis. The target analysis opens in a new window showing the drilled-down report. If you want to create a configuration using a predefined BI Analysis, then you should select the Catalog folder as Sales and then search for the source.


Drill Down

Action: None [✓ Link]

Search for target analysis in /shared/Custom folder

Change the catalog folder to search for analysis in another folder

c. After enabling drill-down, you can validate the drill-down functionality by clicking any component of the visualization.

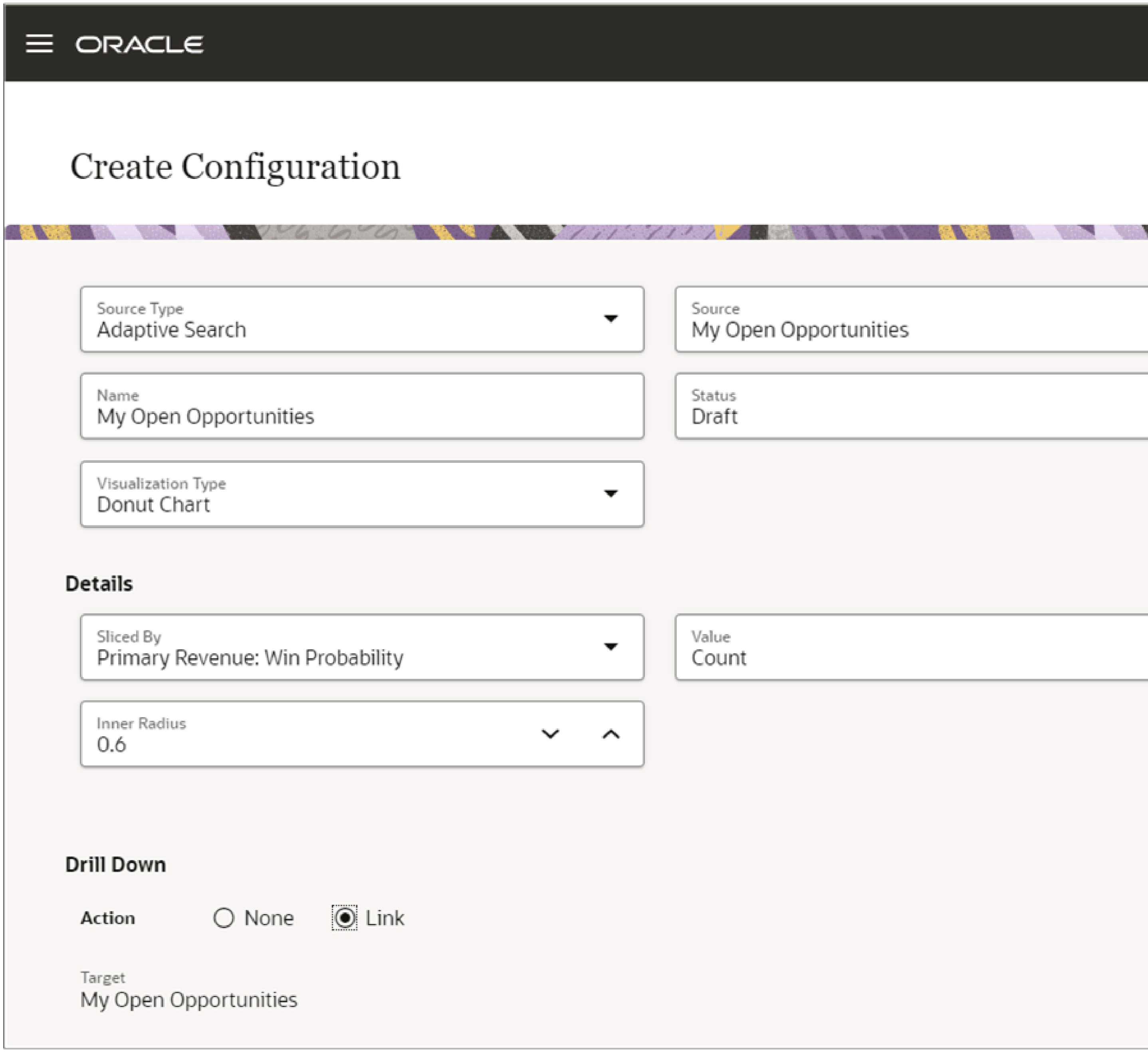
8. Click **Create** to save the configuration.

After enabling drill-down, you can validate functionality by clicking any component of the visualization.

How to Create a Configuration Visualization from a Saved Search

1. Navigate to **Configuration** > **Application Composer**.
2. Click **Visualization Configuration**.

3. Select the **Adaptive Search** source type from the Create Configuration page.



Create Configuration

Source Type: Adaptive Search	Source: My Open Opportunities
Name: My Open Opportunities	Status: Draft
Visualization Type: Donut Chart	

Details

Sliced By: Primary Revenue: Win Probability	Value: Count
Inner Radius: 0.6	

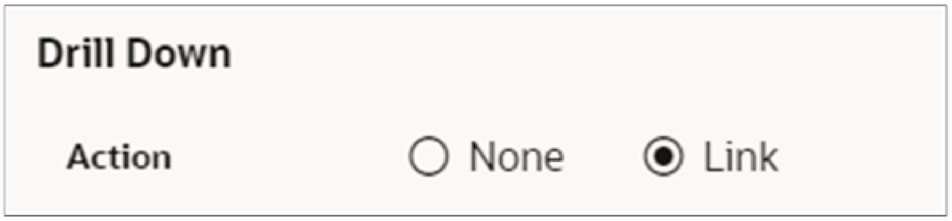
Drill Down

Action: None [✓ Link]

Target: My Open Opportunities

4. In the Source Type field, enter your search criteria. You're searching for an existing saved search.
5. Select a source analysis from the search results to create your visualization. All the fields get auto-selected and the visualization displays on the right side upon selection.

6. To enable drill-down functionality for the visualization (applicable only for the Sales Dashboard), select the drill-down action as **Link**.


Drill Down

Action ○ None [✓ ● Link]

7. After enabling drill-down, you can validate the drill-down functionality by clicking any component of the visualization.
8. Click **Create** to save the configuration.

You can now add this visualization configuration to a Sales Dashboard *Add a Visualization to the Redwood Sales Dashboard*, or on the mobile dashboard using the Mobile Application Setup in Application Composer. See *Configure the Reports Page Layouts*.

Import and Export Visualization Configurations

You can import and export your visualization configurations using the Export Management tool.

How to Import and Export Visualization Configurations

To import and export visualization configurations, refer to the Understanding Import and Export Management for Sales and Fusion Service guide.

Object: Visualization Configuration

Sample Script:

- To export a specific configuration: ReportConfigNumber = 'CDRM_1002'
- To export all configurations: ReportConfigNumber IS NOT NULL

3 Configure the Redwood Sales Dashboard

Overview of the Redwood Sales Dashboard

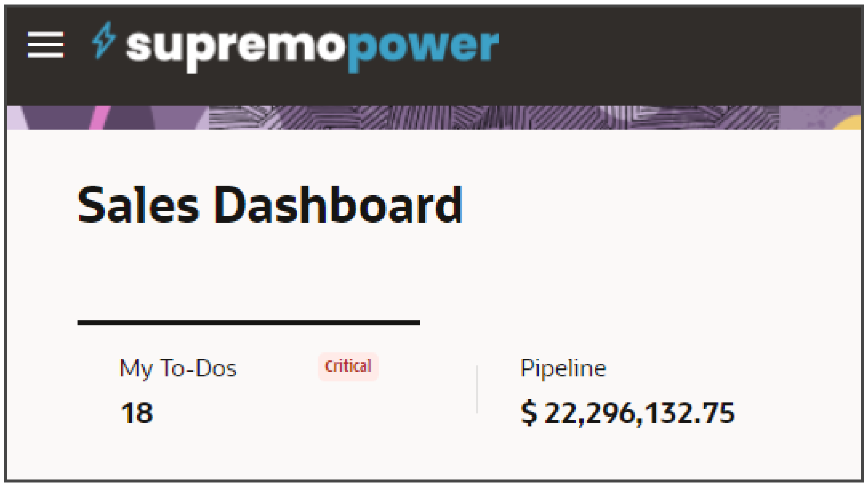
The Sales Dashboard is a visual representation of key sales performance metrics and KPIs (Key Performance Indicators) designed to provide a quick and easy-to-understand overview of a sales team's to do list and pipeline. Sales representatives and managers can review items that need their immediate attention, such as upcoming appointments and overdue tasks. The dashboard can display lists and tables of information, as well as visualizations from OTBI.

In the Redwood user experience of Sales, the Sales Dashboard is constructed using Dashboard fragments, which makes it easier to modify, if needed. Fragments are building blocks that help you configure pages more quickly with a minimal amount of manual coding. This chapter applies only to Sales Dashboards that use Dashboard fragments.

Oracle delivers two Sales Dashboards: one for the sales representative and one for the sales manager. Each version of the dashboard has two pages, by default: My To-Dos and Pipeline.

Dashboard Pages

By default, the Sales Dashboard is divided into two different pages via a set of metric cards at the top of the page: My To-Dos and Pipeline.



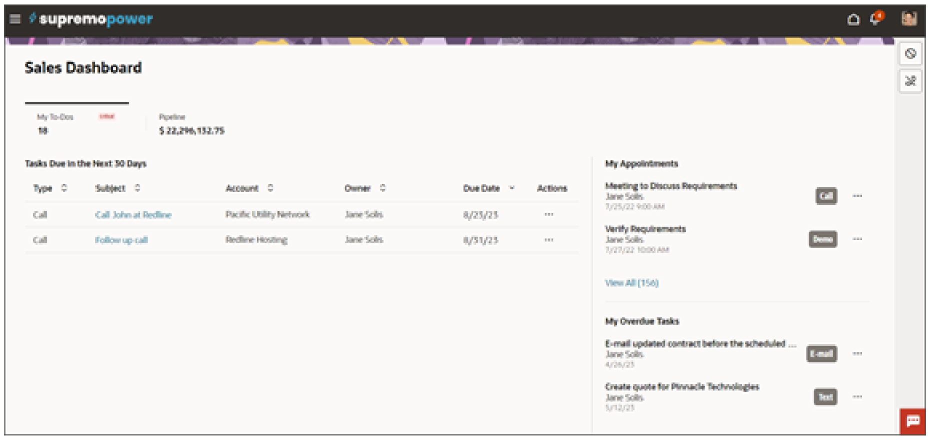
Users can click each metric card to toggle between pages. In this way, metric cards act like tabs because users can click a metric card to view a different page of the dashboard. Each metric card also includes an aggregate sum of items returned by a saved search.

When users click a metric card, they can see a dashboard page that includes a set of dashboard components. Each dashboard page displays unique information that's critical and relevant to either sales representatives or sales managers.

A dashboard page can display up to 5 dashboard components. As an administrator, you can use Oracle Visual Builder Studio to configure which components display on the page. You can also add new pages by creating custom metric cards.

Sales Dashboard for the Sales Representative

Here's the default layout for the My To-Dos dashboard page for sales representatives:



Sales Dashboard

My To-Dos 18 | Critical | Pipeline $ 22,296,132.75

Tasks Due in the Next 30 Days

Type	Subject	Account	Owner	Due Date	Actions
Call	Call John at Redline	Pacific Utility Network	Jane Solis	8/23/23	···
Call	Follow up call	Redline Hosting	Jane Solis	8/31/23	···

My Appointments

Meeting to Discuss Requirements
Jane Solis
7/25/22 9:00 AM
Call ···

Verify Requirements
Jane Solis
7/27/22 10:00 AM
Demo ···

View All (156)

My Overdue Tasks

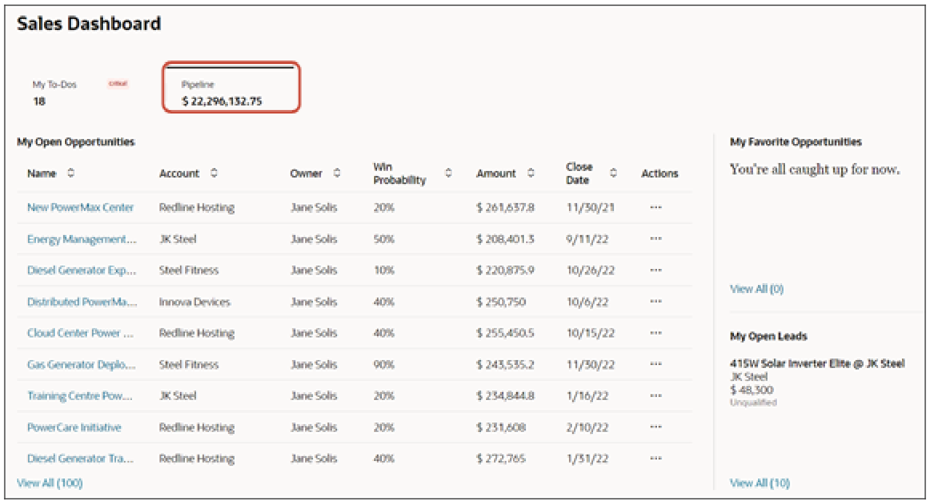
E-mail updated contract before the scheduled ...
Jane Solis
4/26/23
E-mail ···

Create quote for Pinnacle Technologies
Jane Solis
5/12/23
Text ···

The My To-Dos page for the sales representative includes these components:

- Tasks Due in the Next 30 Days
- My Appointments
- My Overdue Tasks

Here's the default layout for the Pipeline dashboard page for sales representatives:



Sales Dashboard

My To-Dos 18 | Critical | Pipeline $ 22,296,132.75

My Open Opportunities

Name	Account	Owner	Win Probability	Amount	Close Date	Actions
New PowerMax Center	Redline Hosting	Jane Solis	20%	$ 261,637.8	11/30/21	···
Energy Management...	JK Steel	Jane Solis	50%	$ 208,401.3	9/11/22	···
Diesel Generator Exp...	Steel Fitness	Jane Solis	10%	$ 220,875.9	10/26/22	···
Distributed PowerMa...	Innova Devices	Jane Solis	40%	$ 250,750	10/6/22	···
Cloud Center Power ...	Redline Hosting	Jane Solis	40%	$ 255,450.5	10/15/22	···
Gas Generator Deplo...	Steel Fitness	Jane Solis	90%	$ 243,535.2	11/30/22	···
Training Centre Pow...	JK Steel	Jane Solis	20%	$ 234,844.8	1/16/22	···
PowerCare Initiative	Redline Hosting	Jane Solis	20%	$ 231,608	2/10/22	···
Diesel Generator Tra...	Redline Hosting	Jane Solis	40%	$ 272,765	1/31/22	···

View All (100)

My Favorite Opportunities

You're all caught up for now.

View All (0)

My Open Leads

415W Solar Inverter Elite @ JK Steel
JK Steel
$ 48,300
Unqualified

View All (10)

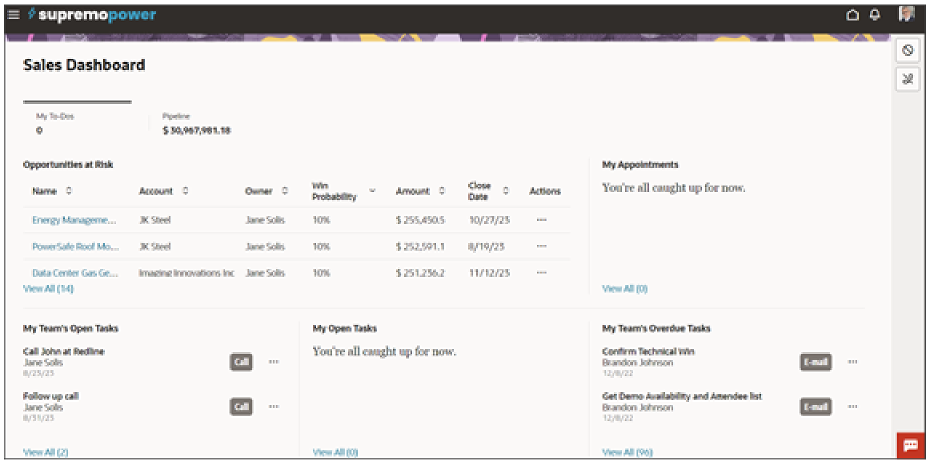
The Pipeline page for the sales representative includes these components:

- My Open Opportunities

  Sales representatives can take an action on an item in the list by clicking **Actions** (the three dot icon), or they can click **View All** to see the complete list of open opportunities.
- My Favorite Opportunities
- My Open Leads

Sales Dashboard for the Sales Manager

Here's the default layout for the My To-Dos dashboard page for sales managers:



Sales Dashboard

My To-Dos
0

Pipeline
$ 30,967,981.18

Opportunities at Risk

Name	Account	Owner	Win Probability	Amount	Close Date	Actions
Energy Manageme...	JK Steel	Jane Solis	10%	$ 255,450.5	10/27/23	...
PowerSafe Roof Mo...	JK Steel	Jane Solis	10%	$ 252,591.1	8/19/23	...
Data Center Gas Ge...	Imaging Innovations Inc	Jane Solis	10%	$ 251,236.2	11/12/23	...

View All (14)

My Appointments

You're all caught up for now.

View All (0)

My Team's Open Tasks

Call John at Redline
Jane Solis
8/25/23
Call ...

Follow up call
Jane Solis
8/31/23
Call ...

View All (2)

My Open Tasks

You're all caught up for now.

View All (0)

My Team's Overdue Tasks

Confirm Technical Win
Brandon Johnson
12/8/22
E-mail ...

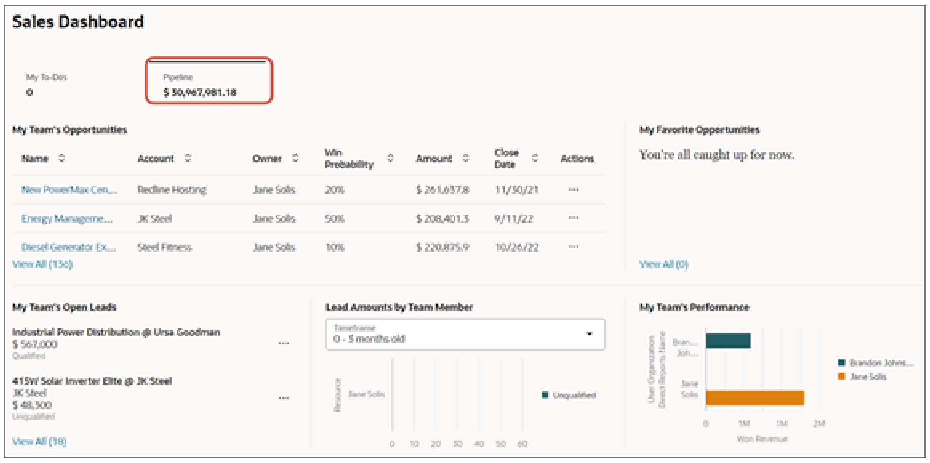
Get Demo Availability and Amendee list
Brandon Johnson
12/8/22
E-mail ...

View All (96)

The My To-Dos page for the sales manager includes these components:

- Opportunities at Risk
- My Appointments
- My Team's Open Tasks
- My Open Tasks
- My Team's Overdue Tasks

Here's the default layout for the Pipeline dashboard page for sales managers:



Sales Dashboard

My To-Dos
0

Pipeline
$ 30,967,981.18

My Team's Opportunities

Name	Account	Owner	Win Probability	Amount	Close Date	Actions
New PowerMax Cen...	Redline Hosting	Jane Solis	20%	$ 261,637.8	11/30/21	...
Energy Manageme...	JK Steel	Jane Solis	50%	$ 208,401.3	9/11/22	...
Diesel Generator Ex...	Steel Fitness	Jane Solis	10%	$ 220,875.9	10/26/22	...

View All (136)

My Favorite Opportunities

You're all caught up for now.

View All (0)

My Team's Open Leads

Industrial Power Distribution @ Ursa Goodman
$ 567,000
Qualified ...

415W Solar Inverter Elite @ JK Steel
JK Steel
$ 48,300
Unqualified ...

View All (18)

Lead Amounts by Team Member

Timeframe
0 - 3 months old

My Team's Performance

The Pipeline page for the sales manager includes these components:

- My Team's Opportunities

  Sales managers can take an action on an item in the list by clicking **Actions** (the three dot icon), or they can click **View All** to see the complete list of open opportunities.

- My Favorite Opportunities

- My Team's Open Leads
- Lead Amounts by Team Member
- My Team's Performance

Get Started with Configuring the Redwood Sales Dashboard

As an administrator, you can use Oracle Visual Builder Studio to configure the components that display on the Sales Dashboard, for both the My To-Dos and Pipeline metric cards. You can also add new metric cards that link to new dashboard pages.

Prerequisite

Before you can configure the Sales Dashboard, you must enable it so that users can navigate to it.

See *Enable the Redwood Sales Dashboard*.

What You Can Modify

The Sales Dashboard displays metric cards at the top of the page. Each metric card is associated with a dashboard layout.

- You can configure the components that display on a dashboard page.

  See *Configure Redwood Sales Dashboard Layouts*.
- You can add new metric cards, each with its own corresponding dashboard page.

  See *Overview of Metric Cards*.

What Components You Can Add

For each dashboard page, you can:

- Add and remove predefined components, and change their display order.
- Add new components.
- Add visualizations you created from saved searches or from Oracle Analytics Cloud analyses.

Here's a list of the types of components that you can add to the Sales Dashboard:

- Metric card

  See *Overview of Metric Cards*.
- List, My List, Revenue list

  See *Add a List to the Redwood Sales Dashboard*.
- Table, Revenue table

See *Add a Table to the Redwood Sales Dashboard*.

- Visualization

  To display visualizations created from saved searches and reports, you use the Chart template type.

  You can choose from a predefined set of visualizations to add to a dashboard. Or you can create your own custom visualizations using the Visualization Configurations page. See *Create a Visualization of a Saved Search or an Analysis*. (You can add the same visualizations to both the Sales Dashboard and to CX Sales Mobile layouts.)

Predefined Sales Dashboard Components

The below components are already available for the Sales Dashboard. You can add these components to any custom Sales Dashboard layout, as needed.

- Deals At Risk saved search table
- Lead Amounts by Team Member chart
- My Appointments saved search list
- My Favorite Opportunities saved search list
- My List
- My Open Leads saved search list
- My Open Opportunities saved search list
- My Open Opportunities saved search table
- My Open Tasks saved search list
- My Overdue Tasks saved search list
- My Team's Open Leads saved search list
- My Team's Open Tasks saved search list
- My Team's Opportunities saved search table
- My Team's Overdue Tasks saved search list
- My Team's Performance chart
- Tasks Due in the Next 30 Days saved search table

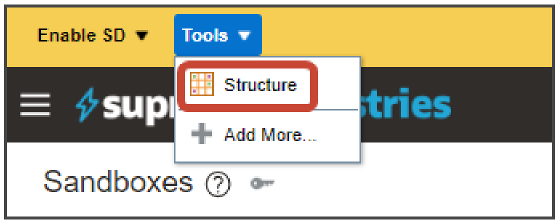
Enable the Redwood Sales Dashboard

Before sales users can access the Sales Dashboard, you must enable it.

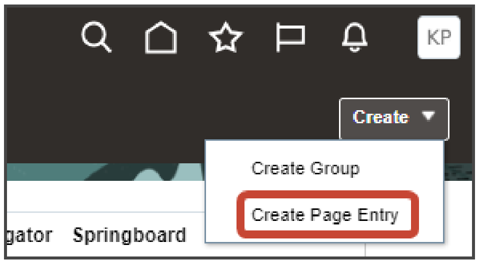
To enable the Sales Dashboard:

1. Navigate to **Configuration** > **Sandboxes** .
2. On the Sandboxes page, click **Create Sandbox**.
3. On the Create Sandbox page, enter a sandbox name.
4. For the Publishable option, click **Yes**.

5. In the All Tools table, select the **Structure** check box.
6. Click **Create and Enter**.
7. On the Sandboxes page, from the **Tools** menu, select **Structure**.



8. On the Navigation Configuration page, click **Create** > **Create Page Entry**.

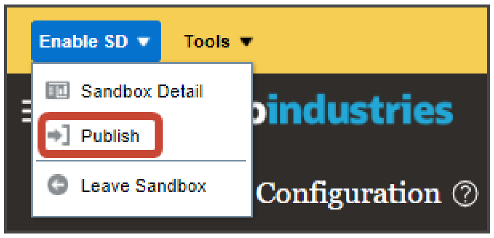


9. On the Create Page Entry page, enter these field values:

Create Page Entry Field Values for Sales Dashboard

Field	Value
Name	Sales Dashboard
Icon	Select your desired icon
Group	(Top Level)
Show on Navigator	Yes
Show on Springboard	Yes
Mobile Enabled	Yes
Link Type	Dynamic URL
Web Application	ORA_FSCM_UI
Destination for Web Application	/redwood/cx-sales/application/container/dashboards/sales-dashboard

10. Click **Save and Close**.
11. Test and confirm that you can access the Sales Dashboard using both the Navigator, as well as from the home page.
12. When ready, publish the sandbox by clicking the sandbox name at the top of the page > **Publish**.



Configure Redwood Sales Dashboard Layouts

To modify the Sales Dashboard, you must first duplicate a predefined Sales Dashboard layout. Let's look at an example of adding a component to a dashboard page.

Predefined Dashboard Layouts

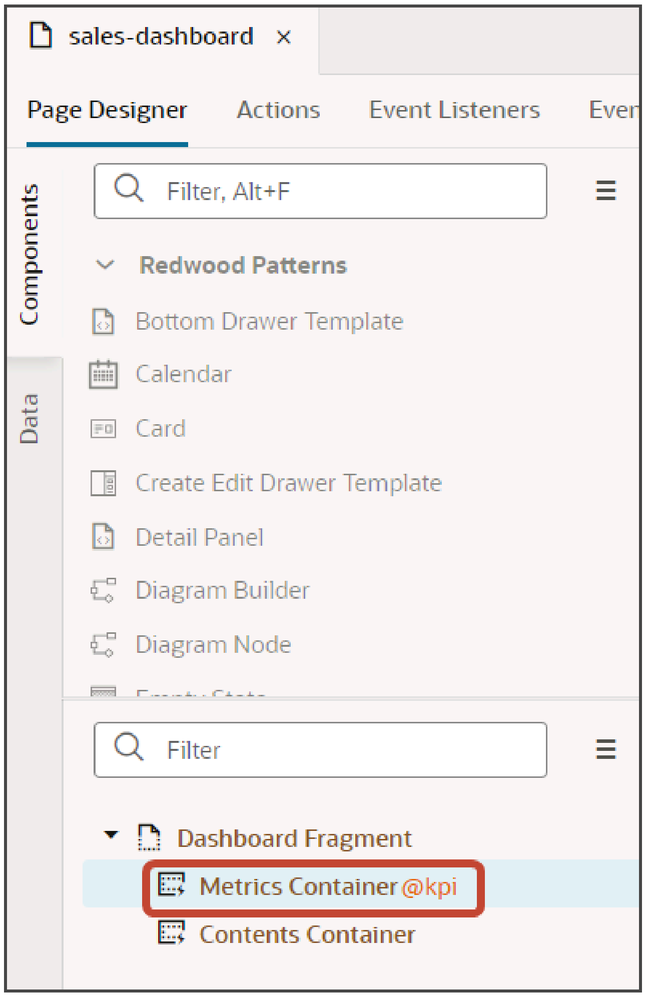
The Sales Dashboard is constructed using a Dashboard fragment, which is divided into two containers that hold dashboard layouts.

Each layout is read-only. To make changes to a layout, you must first duplicate it.

The Dashboard fragment includes these two containers:

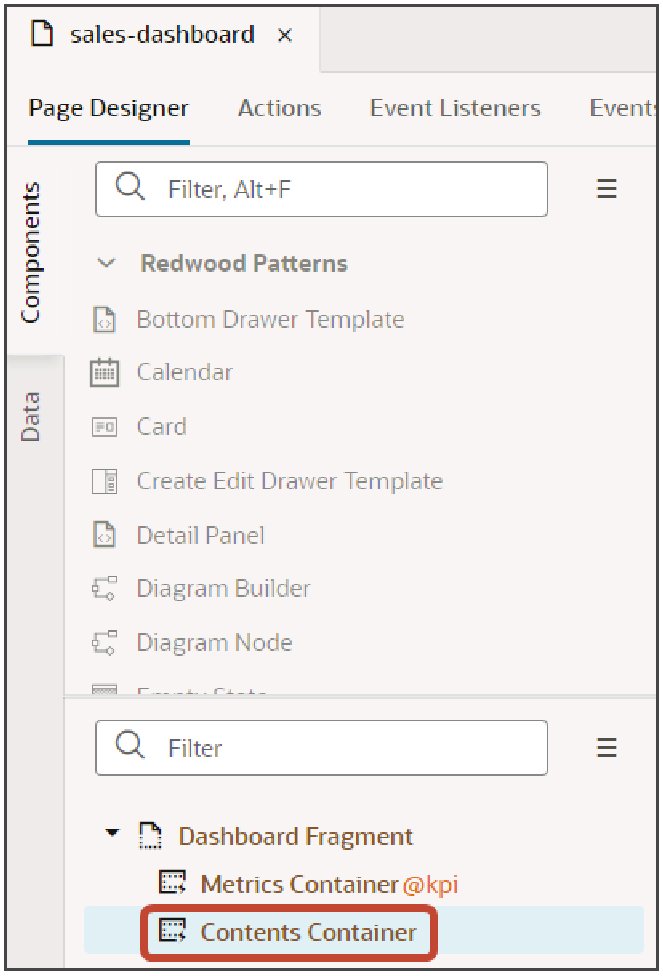
- Metrics Container

  This container holds read-only metric card layouts. Use this container to add new metric cards to metric card layouts.



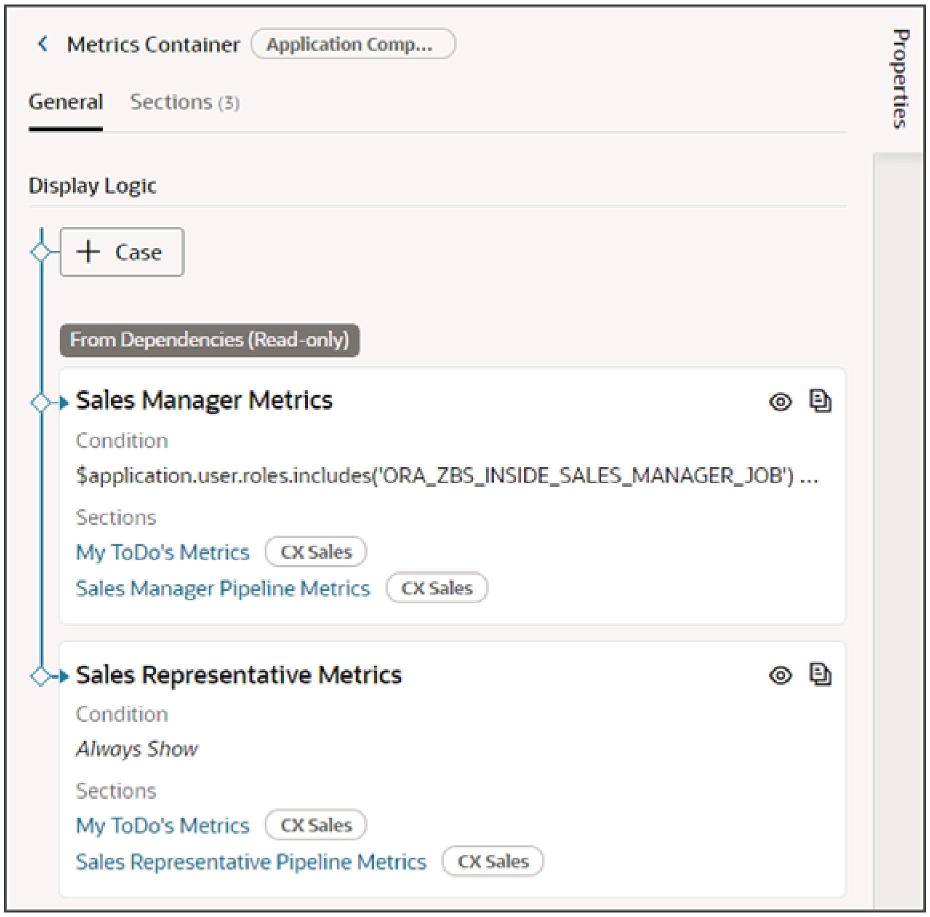
- Contents Container

  This container holds read-only dashboard page layouts. Use this container to add new components to dashboard pages.



The Metrics Container includes two read-only layouts:

- Sales Manager Metrics
- Sales Representative Metrics



Display Logic

Sales Manager Metrics

Condition

$application.user.roles.includes('ORA_ZBS_INSIDE_SALES_MANAGER_JOB') ...

Sections

My ToDo's Metrics — CX Sales

Sales Manager Pipeline Metrics — CX Sales

Sales Representative Metrics

Condition

Always Show

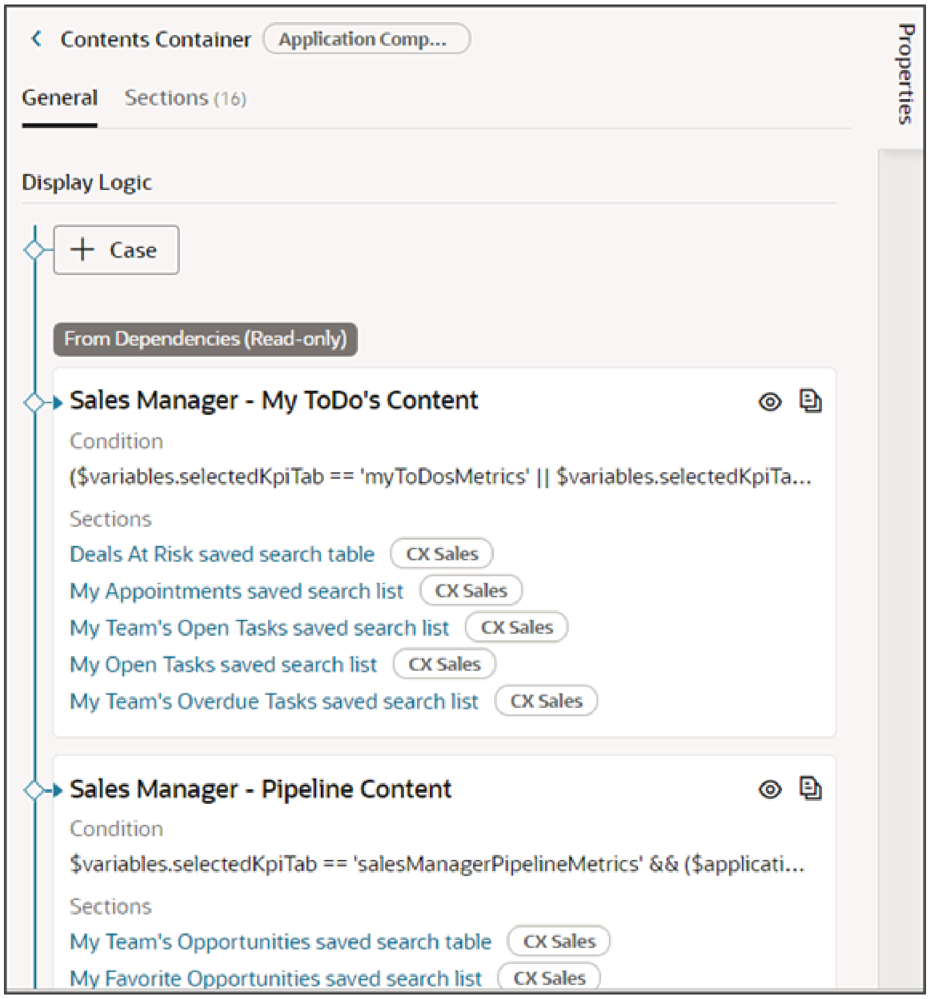
Sections

My ToDo's Metrics — CX Sales

Sales Representative Pipeline Metrics — CX Sales

The Contents Container includes four read-only layouts:

- Sales Manager - My ToDo's Content
- Sales Manager - Pipeline Content
- Sales Representative - My ToDo's Content
- Sales Representative - Pipeline Content



Display Logic

Sales Manager - My ToDo's Content

Condition

($variables.selectedKpiTab == 'myToDosMetrics' || $variables.selectedKpiTa...

Sections

Deals At Risk saved search table CX Sales
My Appointments saved search list CX Sales
My Team's Open Tasks saved search list CX Sales
My Open Tasks saved search list CX Sales
My Team's Overdue Tasks saved search list CX Sales

Sales Manager - Pipeline Content

Condition

$variables.selectedKpiTab == 'salesManagerPipelineMetrics' && ($applicati...

Sections

My Team's Opportunities saved search table CX Sales
My Favorite Opportunities saved search list CX Sales

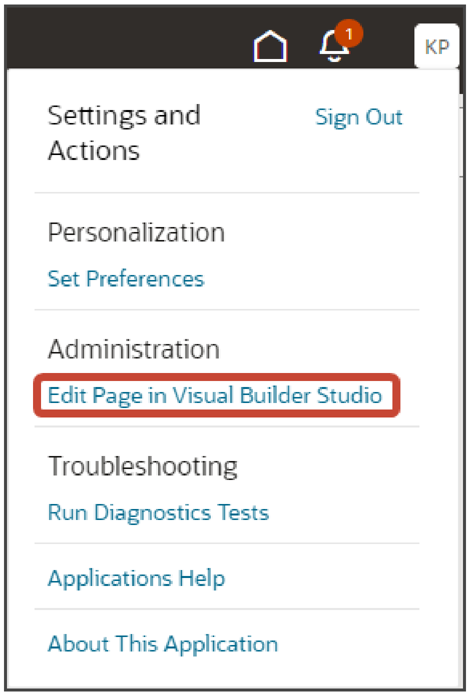
Duplicate a Layout

In this example, let's add a predefined My Open Tasks component to the sales representative's version of the Sales Dashboard.

To start, you must first duplicate an existing layout.

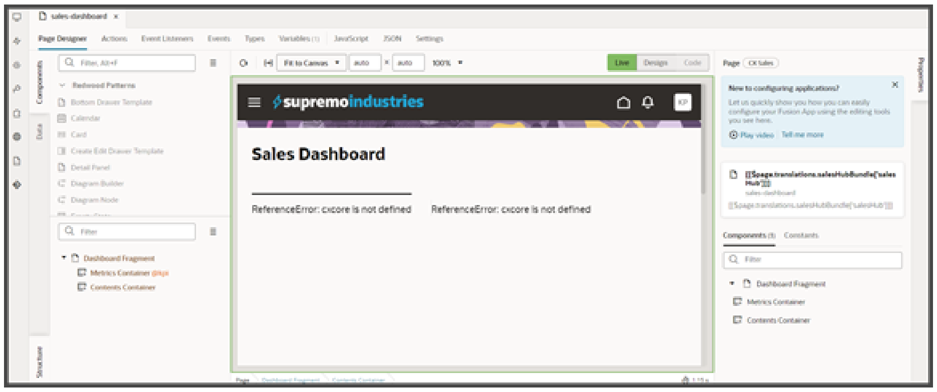
1. Navigate to the Sales Dashboard.

2. Under the Settings and Actions menu, select **Edit Page in Visual Builder Studio**.



3. Select the Redwood Sales project that's already set up for you. If only one project exists, then you will automatically land in that project.
4. If you're working in an active sandbox when you launch Visual Builder Studio from Digital Sales, then Visual Builder Studio looks for a workspace that's associated with your sandbox. If you're not working in a sandbox when you launch Visual Builder Studio, then Visual Builder Studio looks for a workspace without a sandbox. You might have to select a workspace if more than one workspace exists. If no workspace exists, then Visual Builder Studio automatically creates one for you.
5. When you enter into your workspace in Visual Builder Studio, you land on the Page Designer. This is where you create your application extension.

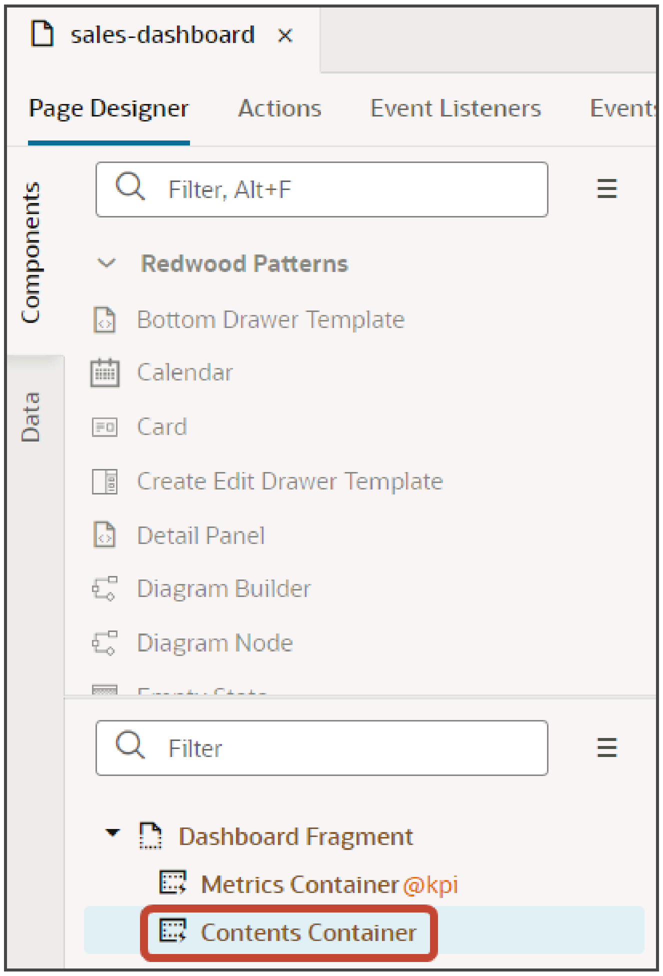
   The Sales Dashboard is displayed in the main design area called the canvas.



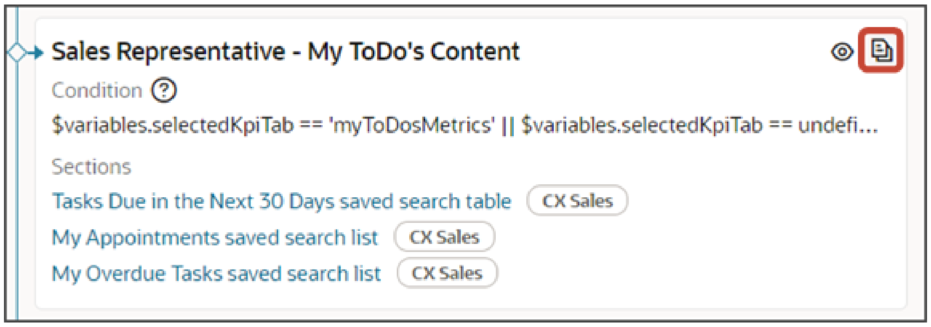
ReferenceError: cxcore is not defined

ReferenceError: cxcore is not defined

6. In this example, let's add a My Open Tasks component to the sales representative's version of the Sales Dashboard.
   On the Structure pane, click the Contents Container node.



7. On the Properties pane, scroll down to the **Sales Representative - My ToDo's Content** layout.
8. Click the **Duplicate** icon to make a copy of the layout.


Sales Representative - My ToDo's Content
Condition
$variables.selectedKpiTab == 'myToDosMetrics' || $variables.selectedKpiTab == undefi...
Sections
Tasks Due in the Next 30 Days saved search table CX Sales
My Appointments saved search list CX Sales
My Overdue Tasks saved search list CX Sales

   Visual Builder Studio makes a copy of the layout named **Sales Representative - My ToDo's Content (Copy)**.
9. Review the layout's condition to ensure that the correct metric card ID and role are specified for this layout. You can either accept the metric card value inherited from the original layout that you duplicated, or you can update the metric card ID.

Use this format for the layout condition:

```
$variable.selectedKpiTab == '<metric card id>'
```

For example:

```
$base.variables.selectedKpiTab == 'myToDosMetrics'
```

Here's an example that includes a role:

```
($variables.selectedKpiTab == 'myToDosMetrics' || $variables.selectedKpiTab
== undefined) && ($user.roles.includes('ORA_ZBS_INSIDE_SALES_MANAGER_JOB') ||
$user.roles.includes('ORA_ZBS_SALES_MANAGER_JOB'))
```

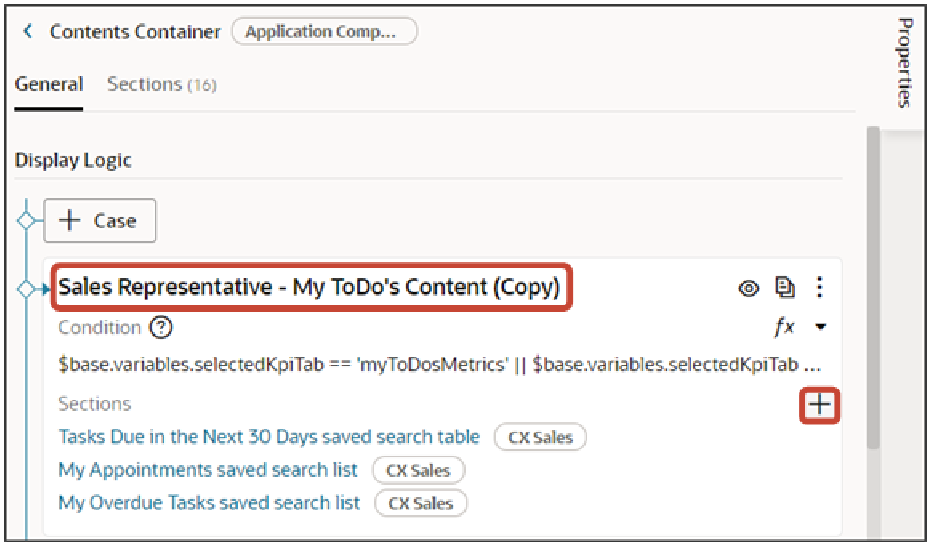
> **Note:** The conditions for all sales manager layouts specify a job role. Notice, however, that the sales representative layouts don't have any role specified. This means that if the signed-in user isn't a sales manager, then they'll always see the sales representative version of the Sales Dashboard. Moreoever, if the sales representative layout is listed first, before the sales manager layout, then the sales representative layout will always display because (1) it's evaluated first and (2) its condition applies to any user regardless of their role. To read more about how layouts are evaluated for display, see *Understand Layout Order*.

You're now ready to make changes to the dashboard layout.

Modify the Layout

Once you have a layout that's ready to modify, you can make a change.

1. Click the Add Section icon to add a component.



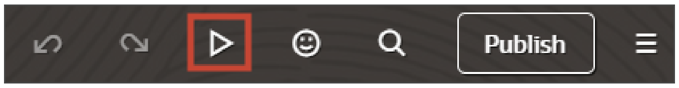
$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.variables.selectedKpiTab ...

Sections
Tasks Due in the Next 30 Days saved search table CX Sales
My Appointments saved search list CX Sales
My Overdue Tasks saved search list CX Sales

2. From the list of available components that display, click **My Open Tasks saved search list**.

3. Test the modified dashboard layout by previewing your application extension.

   a. Click the Preview button to see your changes in your runtime test environment.



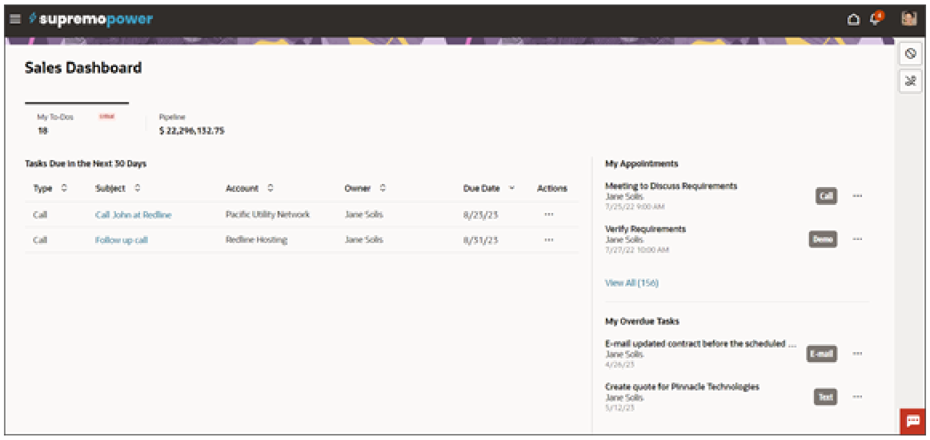
   b. The resulting preview link will be:

```
https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
```

   c. Change the preview link as follows:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
```

   Here's the default layout for the My To-Dos dashboard page for sales representatives:


supremopower

Sales Dashboard

My To-Dos 18

Pipeline $ 22,296,132.75

Tasks Due in the Next 30 Days

Type	Subject	Account	Owner	Due Date	Actions
Call	Call John at Redline	Pacific Utility Network	Jane Solis	8/23/23	···
Call	Follow up call	Redline Hosting	Jane Solis	8/31/23	···

My Appointments

Meeting to Discuss Requirements
Jane Solis
7/25/22 9:00 AM
Call

Verify Requirements
Jane Solis
7/27/22 10:00 AM
Demo

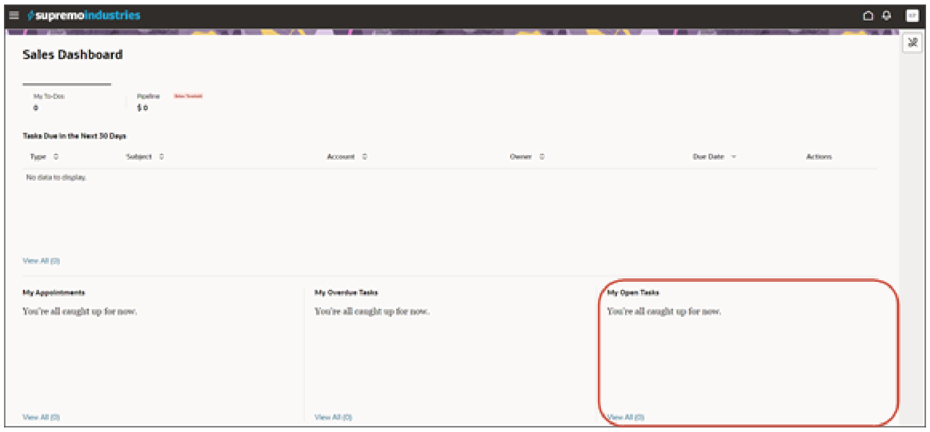
View All (156)

My Overdue Tasks

E-mail updated contract before the scheduled ...
Jane Solis
4/26/23
E-mail

Create quote for Pinnacle Technologies
Jane Solis
5/12/23
Text

   The screenshot below illustrates what the modified dashboard layout looks like with the new My Open Tasks component.


supremoIndustries

Sales Dashboard

My To-Dos 0

Pipeline $ 0

Tasks Due in the Next 30 Days

Type	Subject	Account	Owner	Due Date	Actions
No data to display.					

View All (0)

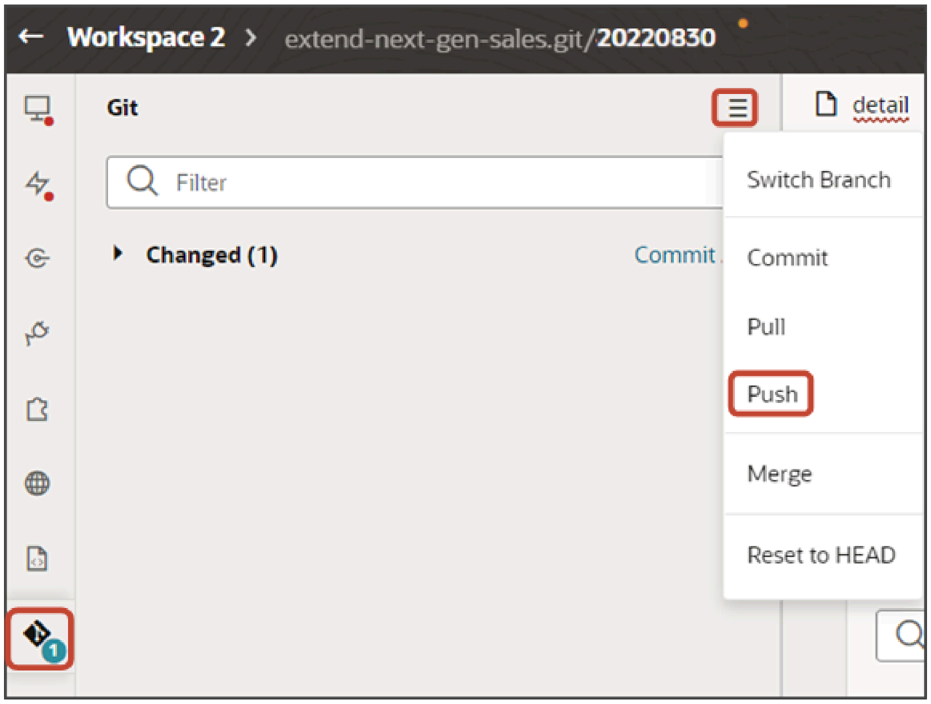
My Appointments
You're all caught up for now.
View All (0)

My Overdue Tasks
You're all caught up for now.
View All (0)

My Open Tasks
You're all caught up for now.
View All (0)

4. Save your work by using the Push Git command.

   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).

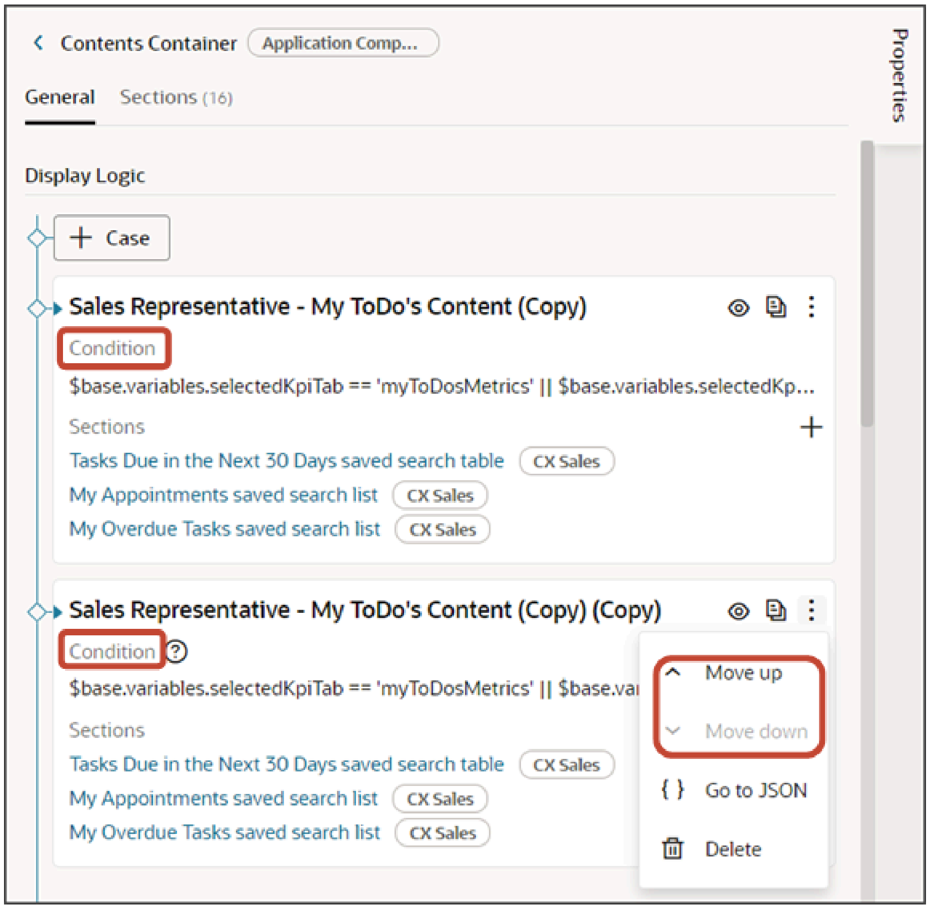


Understand Layout Order

When you duplicate multiple layouts, keep in mind that the sequence of those layouts is meaningful.

The sequence is meaningful because Visual Builder Studio evaluates the conditions of each layout to determine which one to display to the user. The first layout to meet the signed-in user's conditions is the layout that's displayed.

After creating multiple layouts, you can always change the sequence of layouts using the **Move up** and **Move down** options.


Contents Container
Application Comp...
Properties
General
Sections (16)
Display Logic
Case
Sales Representative - My ToDo's Content (Copy)
Condition
Sections
Tasks Due in the Next 30 Days saved search table
CX Sales
My Appointments saved search list
CX Sales
My Overdue Tasks saved search list
CX Sales
Sales Representative - My ToDo's Content (Copy) (Copy)
Condition
Sections
Tasks Due in the Next 30 Days saved search table
CX Sales
My Appointments saved search list
CX Sales
My Overdue Tasks saved search list
CX Sales
Move up
Move down
Go to JSON
Delete

$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.variables.selectedKp...

$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.vai

Add a List to the Redwood Sales Dashboard

On the Sales Dashboard, data can be rendered in a list format. You can add a new list to the Sales Dashboard using one of the predefined lists available for use. Or, add a new list that pulls its data from a custom saved search. This topic illustrates a simple example.

What Lists Can You Add?

The below list components are already available for the Sales Dashboard. You can add these components to any custom Sales Dashboard layout, as needed.

- My Appointments saved search list
- My Favorite Opportunities saved search list
- My List
- My Open Leads saved search list
- My Open Opportunities saved search list
- My Open Tasks saved search list
- My Overdue Tasks saved search list

- My Team's Open Leads saved search list
- My Team's Open Tasks saved search list
- My Team's Overdue Tasks saved search list

If none of these lists meet your needs, then you can also define a custom saved search and then add it as a new list to the Sales Dashboard.

Prerequisites

1. Create the custom saved search and share it with the job roles who view the dashboard. *Create Saved Searches for the Sales Organization*.
2. Capture the UUID of the saved search. You'll use the UUID when adding the table to the Sales Dashboard in Visual Builder Studio.

   For details see the topic *How can I find the UUID for a saved search in Sales in the Redwood User Experience?*
3. Duplicate a Sales Dashboard layout so that you can modify it.

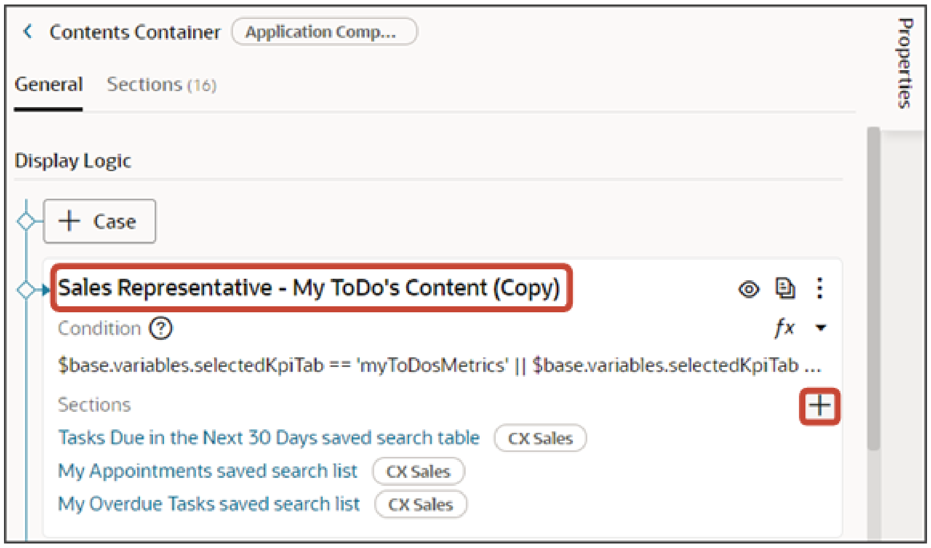
   See *Configure Redwood Sales Dashboard Layouts*.

Display the Custom Saved Search as a List in the Sales Dashboard

Let's add a list to a dashboard layout for a sales representative.

Before you start, duplicate the **Sales Representative - My ToDo's Content** dashboard layout as described in *Configure Redwood Sales Dashboard Layouts* so that you have a layout that's ready to modify. Once that's done, move on to these steps:

1. On the Properties pane for the **Sales Representative - My ToDo's Content (Copy)** layout, click the **+** Add Section icon > **+ New Section**.

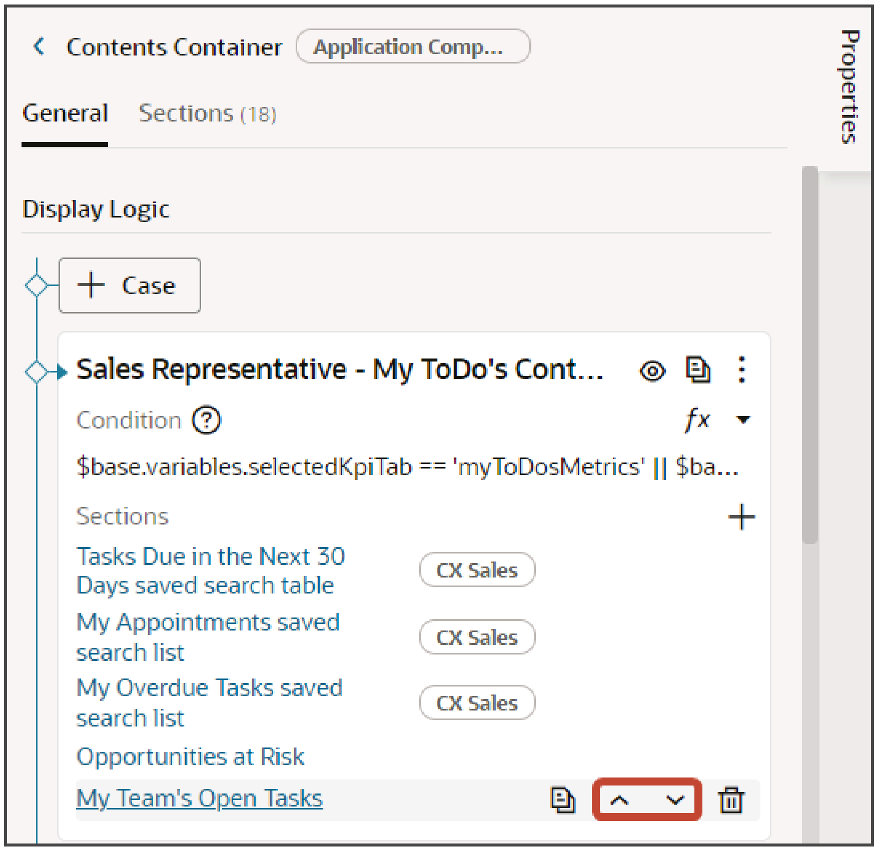

Contents Container
Application Comp...
General
Sections (16)
Display Logic
Case
Sales Representative - My ToDo's Content (Copy)
Condition
fx
Sections
Tasks Due in the Next 30 Days saved search table
CX Sales
My Appointments saved search list
CX Sales
My Overdue Tasks saved search list
CX Sales
Properties

$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.variables.selectedKpiTab ...

2. In the Create Section dialog, in the Title field, enter `My Team's Open Tasks`, and click **OK**.

   Note: The title is the name of the component, not how the component will appear at runtime on the dashboard. The runtime title comes from the panel component that you add to the dashboard.

   The newly added section is added to the bottom of the layout. Use the Move Up arrow to move the new section to the desired location.



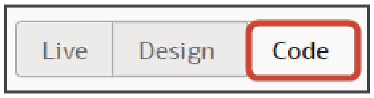
$base.variables.selectedKpiTab == 'myToDosMetrics' || $ba...

Tasks Due in the Next 30 Days saved search table	CX Sales
My Appointments saved search list	CX Sales
My Overdue Tasks saved search list	CX Sales
Opportunities at Risk	
My Team's Open Tasks	

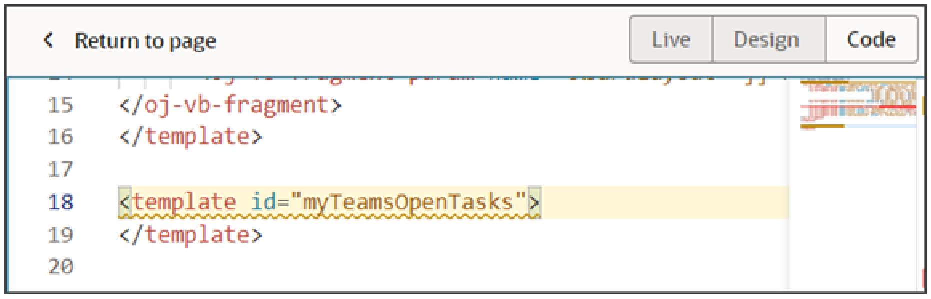
3. Click the **My Team's Open Tasks** link.

   The template editor opens.

4. Click the Code button.



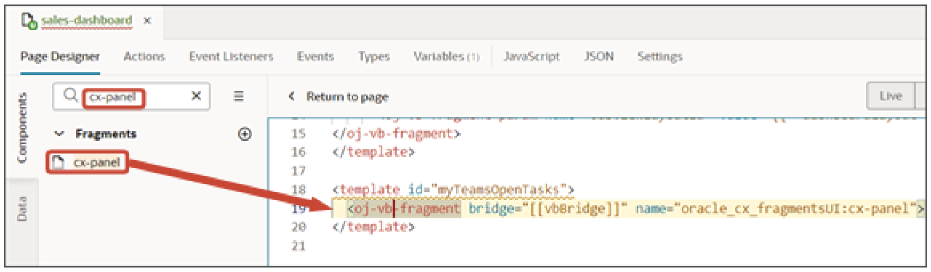
Your new **My Team's Open Tasks** section displays with empty placeholder `template` tags.



```
</oj-vb-fragment>
</template>

<template id="myTeamsOpenTasks">
</template>
```

5. On the Components palette, in the Filter field, enter `cx-panel`.
6. Drag and drop the cx-panel fragment to the template editor, between the template tags.



```
</oj-vb-fragment>
</template>

<template id="myTeamsOpenTasks">
  <oj-vb-fragment bridge="[[vbBridge]]" name="oracle_cx_fragmentsUI:cx-panel">
</template>
```

7. Add the following parameters to the fragment code so that the code looks like the below sample:

```
<template id="myTeamsOpenTasks">
 <oj-vb-fragment name="oracle_cx_fragmentsUI:cx-panel" class="oj-flex oracle-cx-fragmentsUI-cx-fragment-
full-height" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="resource" value='[[ {"name": "activities", "primaryKey": "ActivityId",
 "puid": "ActivityNumber","endpoint": "cx" } ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="query" value='[[ [{"type": "savedSearch", "params": [{"key": "queryUuid",
 "value": "8168e14c-3e70-4357-afa4-b8617eaf23bb" }]},
 {"type": "qbe", "params": [{"key": "RecordSet", "value": "RECORDS_SUBORDINATES_OWN" }]},
 {"type": "qbe", "params": [{"key": "ActivityFunctionCode", "value": "TASK" }]},
 {"type": "qbe", "params": [{"key": "StatusCode", "operator": "in", "value":
 ["NOT_STARTED","IN_PROGRESS","ON_HOLD"] }]},
 {"type": "qbe", "params": [{"key": "DueDate", "operator": "gte", "value": "now/d" }]},
 {"type": "qbe", "params": [{"key": "DueDate", "operator": "lte", "value": "now/d+30d" }]}] ]]'></oj-vb-
fragment-param>
 <oj-vb-fragment-param name="sortCriteria" value='[[ [{"attribute": "DueDate","direction": "asc" }] ]]'>
 </oj-vb-fragment-param>
 <oj-vb-fragment-param name="style" value="[[ 'dashboard' ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="title"
 value="[[$page.translations.salesHubBundle['listofMyTeamsOpenTasks']]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="panelCardLayoutId" value="[[ 'taskDashboardCardLayout' ]]"></oj-vb-
fragment-param>
</oj-vb-fragment>
```

```
</template>
```

In your fragment code, be sure to replace `activites`, `ActivityId`, and `ActivityNumber` with the appropriate values for your object name, primary key, and puid.

Also, replace the queryUuid value with the saved search's UUID value. Review the values for the "query," "sortCriteria," and "title" parameters, and adjust as necessary.

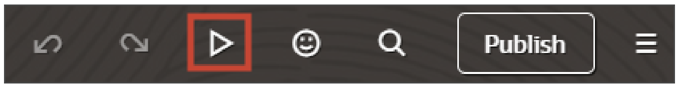
This table describes some of the parameters that you can provide for a list.

Parameters for Dashboard List

Parameter Name	Description
resource	Specify resource details for the saved search, including object name, primary key, puid, and endpoint.
query	Include criteria for querying the data.
sortCriteria	Specify how to sort the data.
title	Specify the title of the table. This title displays on the dashboard at runtime.

8. Test the modified dashboard layout by previewing your application extension.

   a. Click the Preview button to see your changes in your runtime test environment.

   b. The resulting preview link will be:

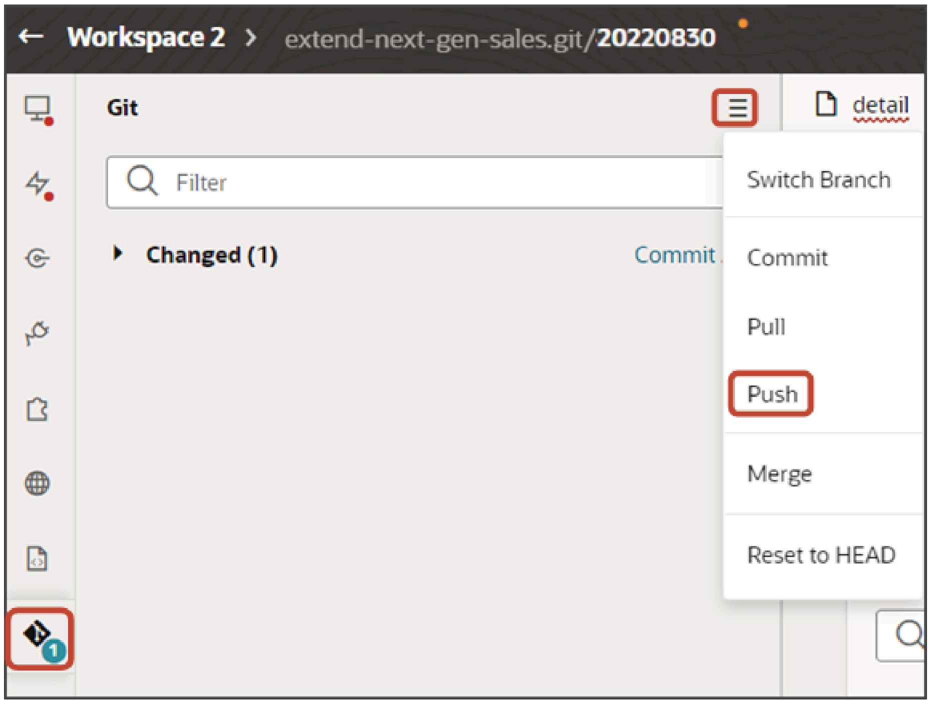
   ```
   https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
   ```

   c. Change the preview link as follows:

   ```
   https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
   ```

9. Save your work by using the Push Git command.

   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).



Resource Details for Sales Objects

Here's a list of resource details for common Sales objects. Use the details below for the list fragment's "resource" parameter.

Resource Details for Sales Objects

Object	Resource Details
Account	`[[ {"name": "accounts", "primaryKey": "PartyId", "puid": "PartyNumber", "endpoint": "cx" } ]]`
Contact	`[[ {"name": "contacts", "primaryKey": "PartyId", "puid": "PartyNumber", "endpoint": "cx" } ]]`
Lead	`[[ {"name": "leads", "primaryKey": "LeadId", "puid": "LeadNumber", "endpoint": "cx" } ]]`
Opportunity	`[[ {"name": "opportunities", "primaryKey": "OptyId", "puid": "OptyNumber", "endpoint": "cx" } ]]`
Appointment	`[[ {"name": "activities", "primaryKey": "ActivityId", "puid": "ActivityNumber", "endpoint": "cx" } ]]` **Note:** The distinction between a list of appointments vs. tasks is made via the filter in the underlying saved search.

Object	Resource Details
Task	`[[ {"name": "activities", "primaryKey": "ActivityId", "puid": "ActivityNumber", "endpoint": "cx" } ]]` **Note:** The distinction between a list of appointments vs. tasks is made via the filter in the underlying saved search.
Custom	`[[ {"name": "Objectname_c", "primaryKey": "Id", "puid": "Id", "endpoint": "cx-custom" } ]]`

Add a My List Component

In addition to adding lists, you can also add a My List component to the Sales Dashboard. The My List component includes data from all objects. For example, My List can include data from both opportunities and leads.

To add a My List component, the steps are the same as adding a list, but the fragment code is different. Add the following fragment code so that the code looks like the below sample. (The template name can be whatever you select.)

```
<template id="myList">
<oj-vb-fragment name="oracle_cx_fragmentsUI:cx-heterogeneous-list"
 class="oj-flex oracle-cx-fragmentsUI-cx-fragment-full-height" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="query" value="[[ [{type: 'qbe', params: [{key: '_userRelevantItems', value:
 true}]}] ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="style" value="[[ 'dashboard' ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="title" value="[['My List']]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="entities" value="[[ ['accounts','contacts','leads','opportunities'] ]]"></oj-
vb-fragment-param>
</oj-vb-fragment>
</template>
```

In your fragment code, be sure to update the values in the "entities" and "title" parameters, as necessary.

Add a Revenue List Component

You can also add a revenue list to the Sales Dashboard.

To add a revenue list, the steps are the same as adding a list, but the fragment code is different. The fragment code includes a "child" parameter.

Add the following fragment code so that the code looks like the below sample. (The template name can be whatever you select.)

```
<template id="revenueList">
<oj-vb-fragment name="oracle_cx_fragmentsUI:cx-panel" class="oj-flex oracle-cx-fragmentsUI-cx-fragment-full-
height" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="resource" value='[[ {"name": "opportunities", "primaryKey": "OptyId", "puid":
 "OptyNumber", "endpoint": "cx" } ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="child" value='[[ {"name": "ChildRevenue", "primaryKey": "RevnId"} ]]'></oj-vb-
fragment-param>
 <oj-vb-fragment-param name="query" value='[[ [{"type": "savedSearch", "params": [{"key": "queryUuid",
 "value": "73b21b33-db08-4327-bf30-88c3c9e0f70d" }]}] ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="extensionId" value="{{ 'oracle_cx_salesUI' }}"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="style" value="[[ 'dashboard' ]]"></oj-vb-fragment-param>
```

```
     <oj-vb-fragment-param name="title" value="[['Revenue']]"></oj-vb-fragment-param>
     <oj-vb-fragment-param name="panelCardLayoutId" value="[[ 'dashboardCardLayout' ]]"></oj-vb-fragment-param>
     </oj-vb-fragment>
    </template>
```

In your fragment code, be sure to replace the queryUuid value with the saved search's UUID value and update the "title" parameter.

Add a Table to the Redwood Sales Dashboard

On the Sales Dashboard, data can be rendered in a table format. You can add a new table to the Sales Dashboard using one of the predefined tables available for use. Or, add a new table that pulls its data from a custom saved search. This topic illustrates a simple example.

What Tables Can You Add?

The below table components are already available for the Sales Dashboard. You can add these components to any custom Sales Dashboard layout, as needed.

- Deals At Risk saved search table
- My Open Opportunities saved search table
- My Team's Opportunities saved search table
- Tasks Due in the Next 30 Days saved search table

If none of these tables meet your needs, then you can also define a custom saved search and then add it as a new table to the Sales Dashboard.

Prerequisites

1. Create the custom saved search and share it with the job roles who view the dashboard. *Create Saved Searches for the Sales Organization*.
2. Capture the UUID of the saved search. You'll use the UUID when adding the table to the Sales Dashboard in Visual Builder Studio. For details see the topic *How can I find the UUID for a saved search in Sales in the Redwood User Experience?*
3. Duplicate a Sales Dashboard layout so that you can modify it.

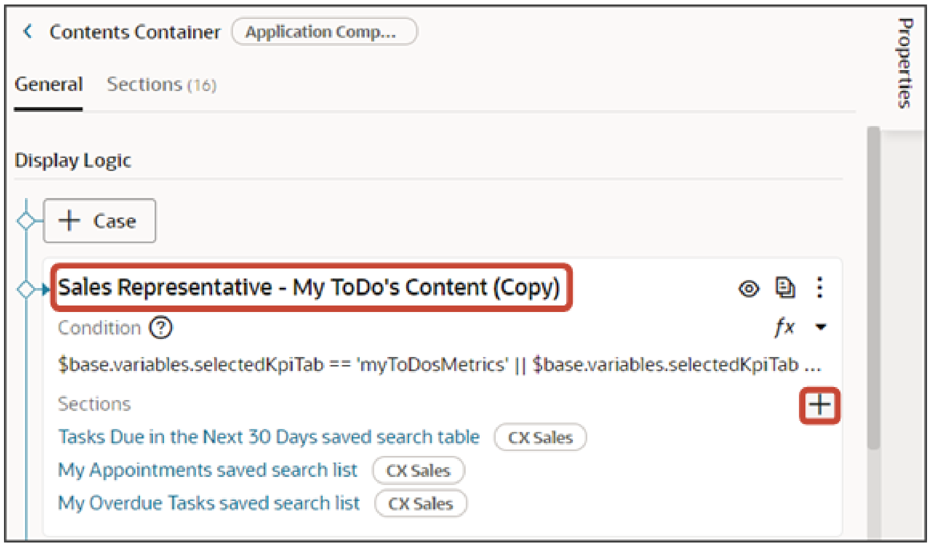
   See *Configure Redwood Sales Dashboard Layouts*.

Display the Custom Saved Search as a Table in the Sales Dashboard

Let's add a table to a dashboard layout for a sales representative.

Before you start, duplicate the **Sales Representative - My ToDo's Content** dashboard layout as described in *Configure Redwood Sales Dashboard Layouts* so that you have a layout that's ready to modify.

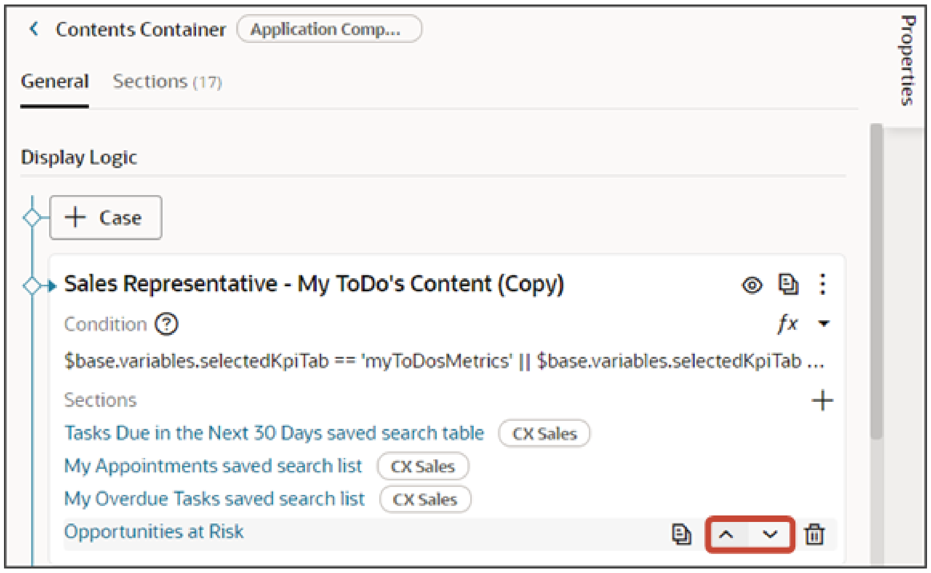
1. On the Properties pane for the **Sales Representative - My ToDo's Content (Copy)** layout, click the **+** Add Section icon > **+ New Section**.


Contents Container
Application Comp...
Properties
General
Sections (16)
Display Logic
+ Case
Sales Representative - My ToDo's Content (Copy)
Condition
fx
$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.variables.selectedKpiTab ...
Sections
Tasks Due in the Next 30 Days saved search table
CX Sales
My Appointments saved search list
CX Sales
My Overdue Tasks saved search list
CX Sales

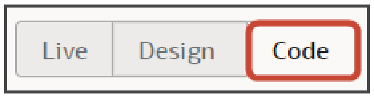
2. In the Create Section dialog, in the Title field, enter `Opportunities at Risk`, and click **OK**.

   Note: The title is the name of the component, not how the component will appear at runtime on the dashboard. The runtime title comes from the subview component that you add to the dashboard.

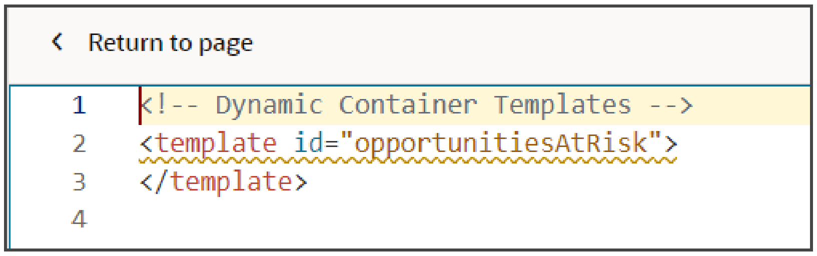
   The newly added section is added to the bottom of the layout. Use the Move Up arrow to move the new section to the desired location.


Contents Container
Application Comp...
Properties
General
Sections (17)
Display Logic
+ Case
Sales Representative - My ToDo's Content (Copy)
Condition
fx
$base.variables.selectedKpiTab == 'myToDosMetrics' || $base.variables.selectedKpiTab ...
Sections
Tasks Due in the Next 30 Days saved search table
CX Sales
My Appointments saved search list
CX Sales
My Overdue Tasks saved search list
CX Sales
Opportunities at Risk

3. Click the **Opportunities at Risk** link.
   The template editor opens.
4. Click the Code button.

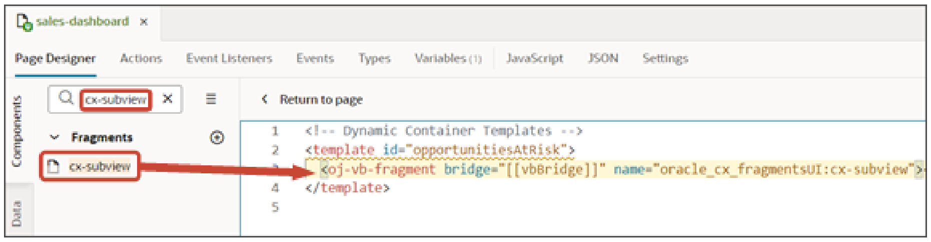


Your new **Opportunities at Risk** section displays with empty placeholder `template` tags.



```
<!-- Dynamic Container Templates -->
<template id="opportunitiesAtRisk">
</template>
```

5. On the Components palette, in the Filter field, enter `cx-subview`.
6. Drag and drop the cx-subview fragment to the template editor, between the template tags.



```
<!-- Dynamic Container Templates -->
<template id="opportunitiesAtRisk">
  <oj-vb-fragment bridge="[[vbBridge]]" name="oracle_cx_fragmentsUI:cx-subview">
</template>
```

7. Add the following parameters to the fragment code so that the code looks like the below sample:

```
<template id="opportunitiesAtRisk">
 <oj-vb-fragment name="oracle_cx_fragmentsUI:cx-subview" class="oj-flex oracle-cx-fragmentsUI-cx-
fragment-full-height" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="resource" value='[[ {"name": "opportunities", "primaryKey": "OptyId",
 "puid": "OptyNumber", "endpoint": "cx" } ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="query" value='[[ [{"type": "savedSearch", "params": [{"key": "queryUuid",
 "value": "e7e61371-9c42-47bb-a16d-8e6c4235cf1c" }]},
 {"type": "qbe", "params": [{"key": "WinProb", "operator": "lt", "value": 40 }]},
 {"type": "qbe", "params": [{"key": "EffectiveDate", "operator": "gte", "value": "now/d-30d" }]},
 {"type": "qbe", "params": [{"key": "EffectiveDate", "operator": "lte", "value": "now/d+90d" }]}] ]]'></
oj-vb-fragment-param>
 <oj-vb-fragment-param name="sortCriteria" value='[[ [{"attribute": "WinProb","direction": "asc" },
 {"attribute": "Revenue","direction": "desc" }] ]]'> </oj-vb-fragment-param>
 <oj-vb-fragment-param name="style" value="[[ 'dashboard' ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="title" value="[['Opportunities at Risk']]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="subviewLayoutId" value="[[ 'dashboardLayout' ]]"></oj-vb-fragment-param>
</oj-vb-fragment>
```

```
</template>
```

In your fragment code, be sure to replace `opportunities`, `OptyId`, and `OptyNumber` with the appropriate values for your object name, primary key, and puid.

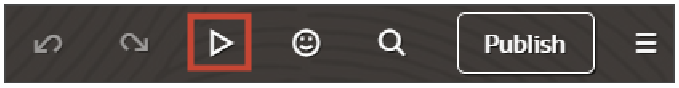
Also, replace the queryUuid value with the saved search's UUID value. Review the values for the "query," "sortCriteria," and "title" parameters, and adjust as necessary.

This table describes some of the parameters that you can provide for a table.

Parameters for Dashboard Table

Parameter Name	Description
resource	Specify resource details for the saved search, including object name, primary key, puid, and endpoint. For a list of resource details for common Sales objects, see "Resource Details for Sales Objects" in *Add a List to the Redwood Sales Dashboard*.
query	Include criteria for querying the data.
sortCriteria	Specify how to sort the data.
title	Specify the title of the table. This title displays on the dashboard at runtime.

8. Test the modified dashboard layout by previewing your application extension.
   a. Click the Preview button to see your changes in your runtime test environment.



   b. The resulting preview link will be:

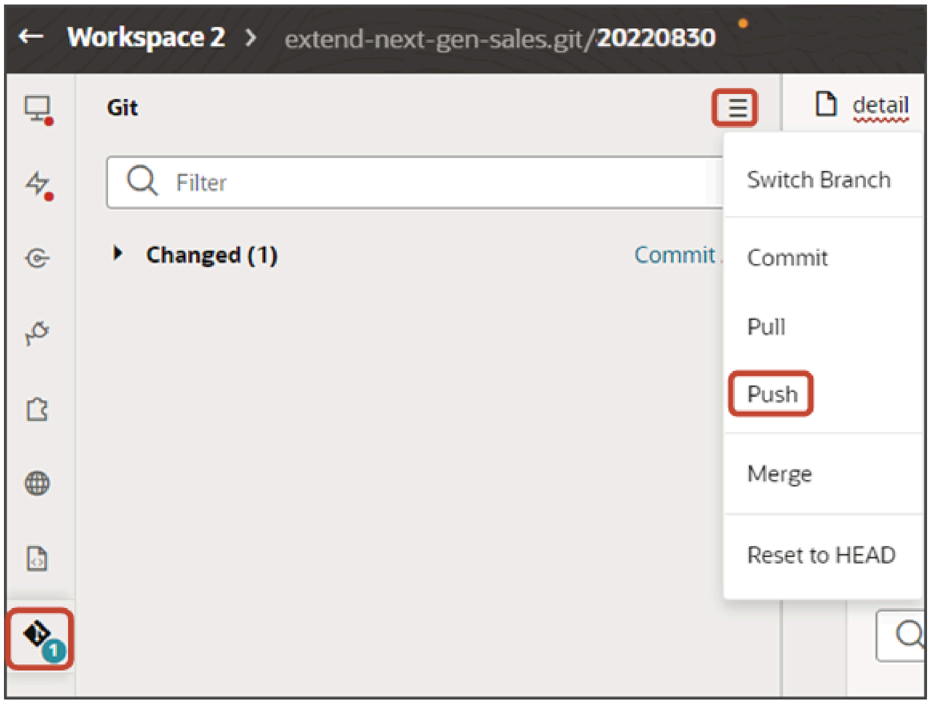
```
https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
```

   c. Change the preview link as follows:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
```

9. Save your work by using the Push Git command.

   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).



Add a Revenue Table Component

You can also add a revenue table to the Sales Dashboard.

To add a revenue table, the steps are the same as adding a table, but the fragment code is different. The fragment code includes a "child" parameter.

Add the following fragment code so that the code looks like the below sample. (The template name can be whatever you select.)

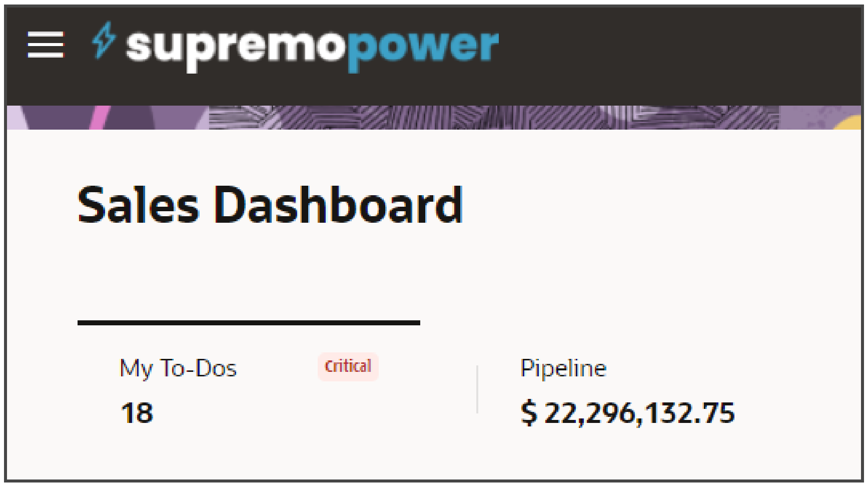
```
<template id="revenueTable">
<oj-vb-fragment name="oracle_cx_fragmentsUI:cx-subview" class="oj-flex oracle-cx-fragmentsUI-cx-fragment-
full-height" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="resource" value='[[ {"name": "opportunities", "primaryKey": "OptyId", "puid":
 "OptyNumber", "endpoint": "cx" } ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="child" value='[[ {"name": "ChildRevenue", "primaryKey": "RevnId"} ]]'></oj-vb-
fragment-param>
 <oj-vb-fragment-param name="query" value='[[ [{"type": "savedSearch", "params": [{"key": "queryUuid",
 "value": "73b21b33-db08-4327-bf30-88c3c9e0f70d" }]}] ]]'></oj-vb-fragment-param>
 <oj-vb-fragment-param name="style" value="[[ 'dashboard' ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="title" value="[['Revenue']]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="subviewLayoutId" value="[[ 'dashboardLayout' ]]"></oj-vb-fragment-param>
 </oj-vb-fragment>
</template>
```

In your fragment code, be sure to replace the queryUuid value with the saved search's UUID value and update the "title" parameter.

Overview of Metric Cards

By default, the Sales Dashboard is divided into two different pages via a set of metric cards at the top: My To-Dos and Pipeline. Users can click each metric card to toggle between pages. Metric cards are also useful because they display an aggregate sum of items returned by a saved search. For example, a metric card can display - at-a-glance - how many overdue tasks exist.

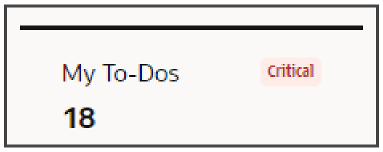
Here's a screenshot of the two metric cards that display at the top of the Sales Dashboard: My To-Dos and Pipeline.



Metric cards act like tabs because users can click a metric card to view a different page of the dashboard. If required, you can create your own metric cards and corresponding dashboard pages, as well.

Each metric card also includes an aggregate sum of items returned by a saved search:

- My To-Dos

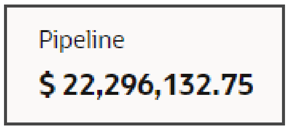
  

  The My To-Dos card lists the total number of the sales user's open tasks. This number is populated from a default "My Open Tasks" saved search.

  Note that for sales representatives, the My To-Dos page of the Sales Dashboard doesn't include a My Open Tasks component. Instead, the dashboard highlights the sales representative's upcoming and overdue tasks. You can add the My Open Tasks component using Visual Builder Studio, if needed.

  For sales managers, however, the My To-Dos page of the Sales Dashboard does include a My Open Tasks component.

- Pipeline

  The Pipeline card lists the total amount of open opportunities for the sales representative. The sales manager, meanwhile, can see the total amount of open opportunities for the sales team as a whole.

What You Can Configure

A metric card is associated with a dashboard page that includes a set of components.

As an administrator, you can:

- Configure which components display on the page for a given metric card.
- Create custom metric cards with its own saved search and set of dashboard components.

  See *Configure Metric Cards*.

  When you create a metric card, you can also optionally define a set of badges that can display under certain conditions.

Metric Card Badges

Metric cards can optionally display badges that visually alert salespeople when data meets certain conditions, which you can set.

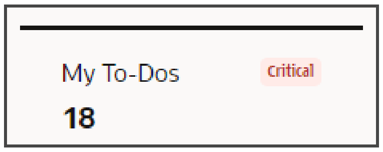
For example, the My To-Dos metric card displays a badge under certain conditions. This metric card has two conditions and displays a different badge depending on the number of overdue tasks.

My Overdue Tasks Metric Card Badges

Condition: Number of Overdue Tasks	Badge Text
>= 5 and <=10	Warning
>10	Critical

If there are fewer than 5 overdue tasks, then no badge displays on the metric card.

In the below example, the card displays a **Critical** badge.



Configure Metric Cards

The Sales Dashboard comes with the My ToDos and Pipeline metric cards and corresponding pages, but you can add more. This topic illustrates how to create a custom metric card and associated dashboard layout. You do this configuration in Oracle Visual Builder Studio.

Prerequisites

A metric card displays data from a saved search. Before you add a metric card to the Sales Dashboard and configure it, you must:

- Create the saved search
- Share the saved search with the appropriate job role
- Capture the UUID of the saved search. For details see the topic *How can I find the UUID for a saved search in Sales in the Redwood User Experience?*

Overview of Setup

To add a metric card, you must:

1. Duplicate a metric card layout.

   See *Duplicate a Layout*.
2. Add a new metric card and make note of the metric card ID.

   See *Add a New Metric Card*.
3. Optionally add a set of badges to a metric card.

   See *How to Add Metric Card Badges*.
4. Associate a dashboard page layout to the new metric card by adding the metric card ID to the dashboard page layout's condition.

   See *Associate a Dashboard Layout to Your New Metric Card*.

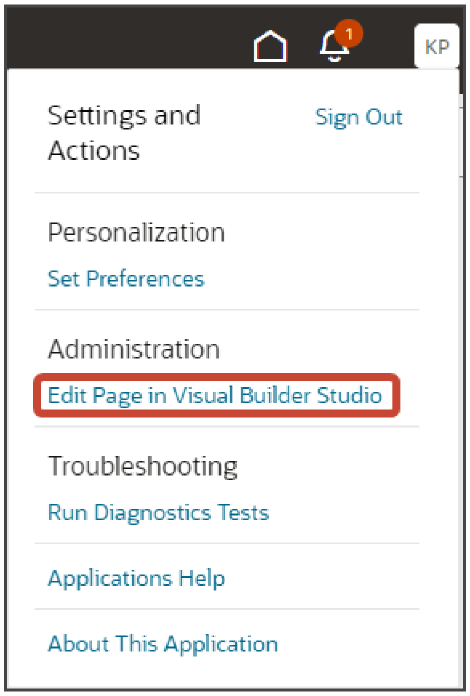
Duplicate a Layout

In this example, let's add a My Todos metric card to the sales representative's version of the Sales Dashboard.

To start, you must first duplicate an existing layout.

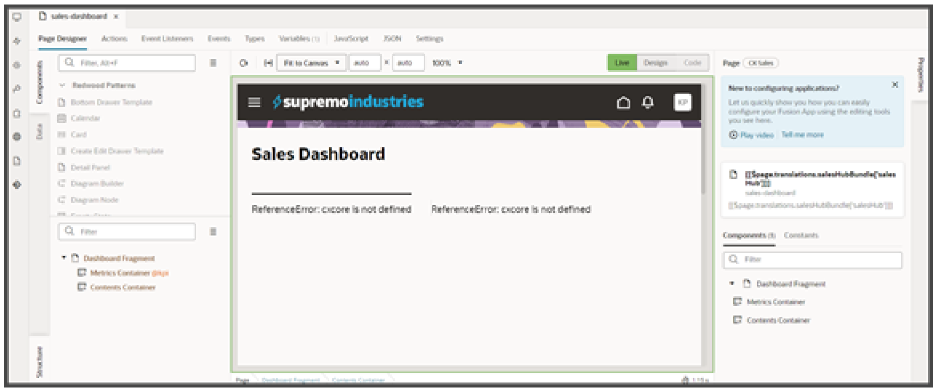
1. Navigate to the Sales Dashboard.

2. Under the Settings and Actions menu, select **Edit Page in Visual Builder Studio**.



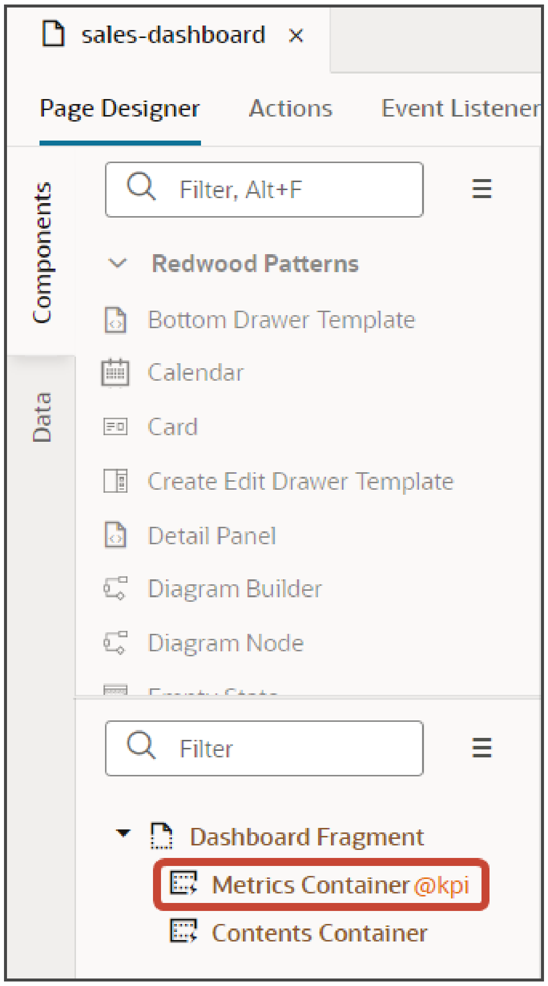
3. Select the Redwood Sales project that's already set up for you. If only one project exists, then you will automatically land in that project.
4. If you're working in an active sandbox when you launch Visual Builder Studio from Digital Sales, then Visual Builder Studio looks for a workspace that's associated with your sandbox. If you're not working in a sandbox when you launch Visual Builder Studio, then Visual Builder Studio looks for a workspace without a sandbox. You might have to select a workspace if more than one workspace exists. If no workspace exists, then Visual Builder Studio automatically creates one for you.
5. When you enter into your workspace in Visual Builder Studio, you land on the Page Designer. This is where you create your application extension.

   The Sales Dashboard is displayed in the main design area called the canvas.

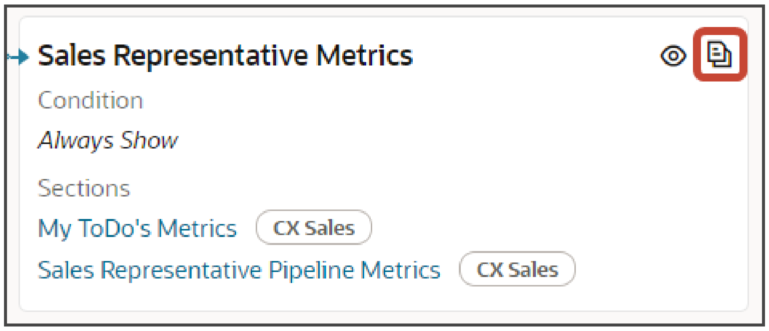


6. In this example, let's add a My Todos metric card to the sales representative's version of the Sales Dashboard.

   On the Structure pane, click the Metrics Container node.



7. On the Properties pane, scroll down to the **Sales Representative Metrics** layout.
8. Click the **Duplicate** icon to make a copy of the layout.


Sales Representative Metrics
Condition
Always Show
Sections
My ToDo's Metrics
CX Sales
Sales Representative Pipeline Metrics
CX Sales

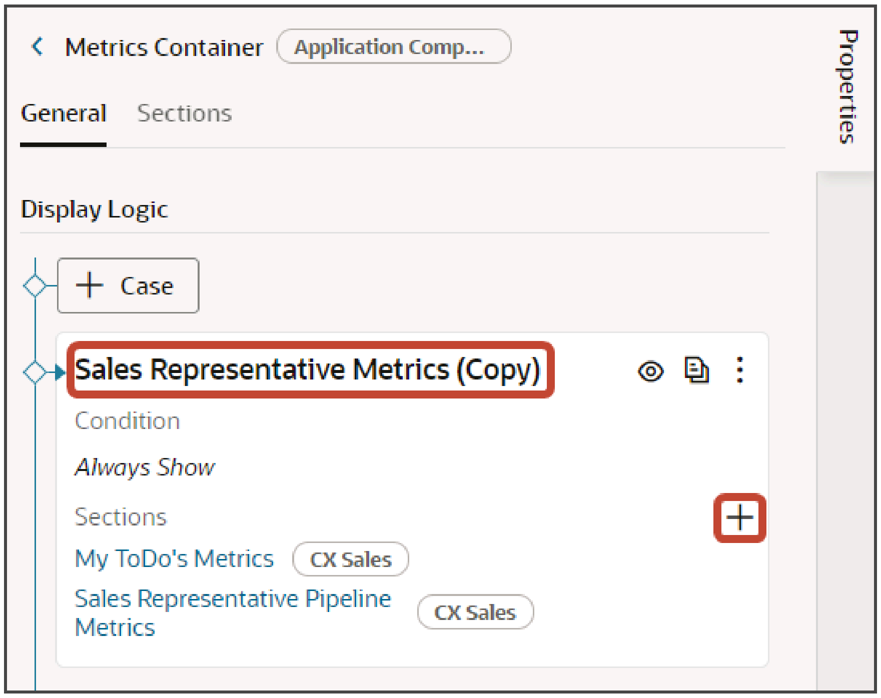
   Visual Builder Studio makes a copy of the layout named **Sales Representative Metrics (Copy)**.

You're now ready to make changes to the dashboard layout.

Add a New Metric Card

Let's add a metric card.

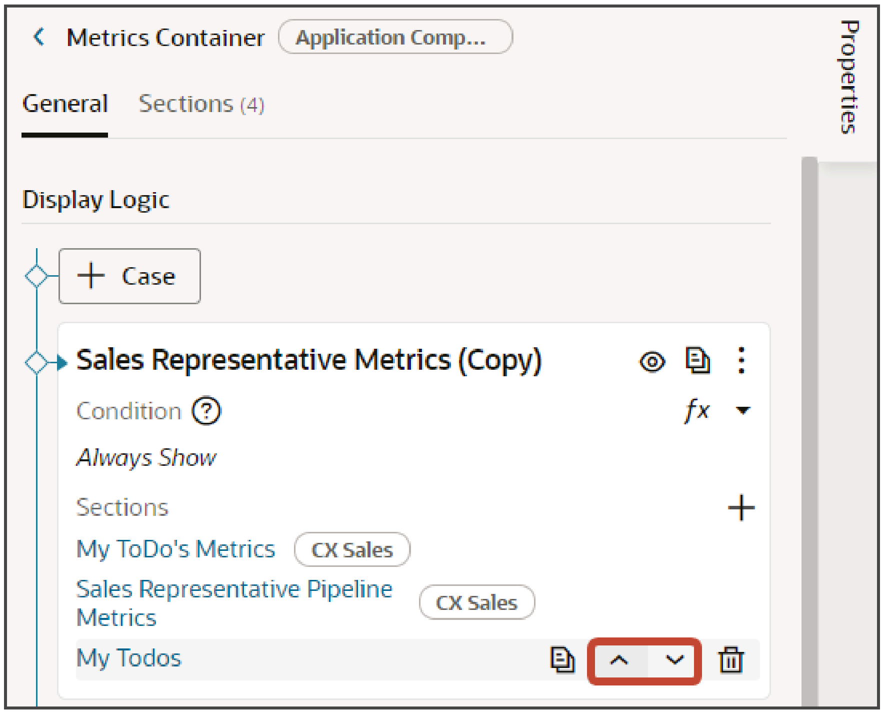
1. On the Properties pane for the **Sales Representative Metrics (Copy)** layout, click the **+** Add Section icon > **+ New Section**.



2. In the Create Section dialog, in the Title field, enter `My Todos`, and click **OK**.

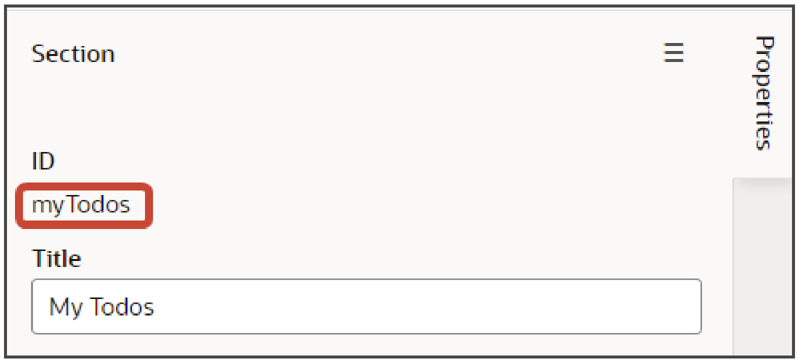
   > **Note:** The title is the name of the metric card, not how the metric card will appear at runtime on the dashboard. The runtime title comes from the cx-metric-card fragment that you add to the dashboard.

   The newly added metric card is added to the bottom of the layout. Use the Move Up arrow to move the new metric card to the desired location.

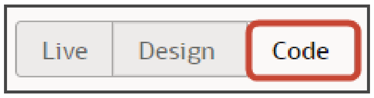
   

3. Click the **My Todos** link.

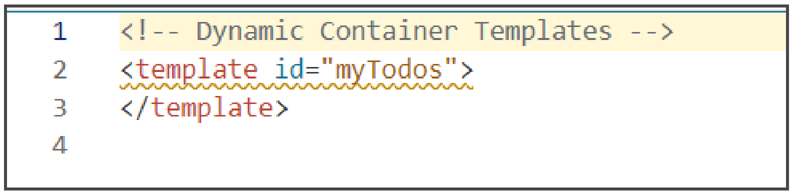
   The template editor opens.
4. Make a note of the metric card's ID.

5. Click the Code button.

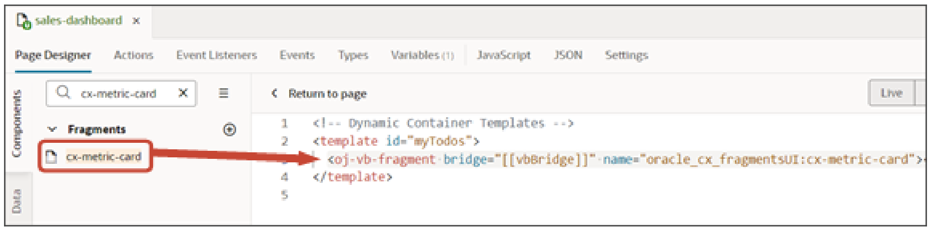


Your new **My Todos** section displays with empty placeholder `template` tags.



```
<!-- Dynamic Container Templates -->
<template id="myTodos">
</template>
```

6. On the Components palette, in the Filter field, enter `cx-metric-card`.
7. Drag and drop the cx-metric-card fragment to the template editor, between the template tags.



```
<!-- Dynamic Container Templates -->
<template id="myTodos">
  <oj-vb-fragment bridge="[[vbBridge]]" name="oracle_cx_fragmentsUI:cx-metric-card">
</template>
```

8. Add the following parameters to the fragment code so that the code looks like the below sample:

```
<template id="myTodos">
 <oj-vb-fragment name="oracle_cx_fragmentsUI:cx-metric-card" bridge="[[ vbBridge ]]">
 <oj-vb-fragment-param name="resource" value="[[ {"name": "activities", "primaryKey": "ActivityId",
 "endpoint": "cx" } ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="query" value="[[ [{"type": "savedSearch", "params": [{"key": "queryUuid",
 "value": "b91c2d92-bb85-4a34-82b8-7091e604b697"}]}] ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="aggregate" value="[[ {"field": "PrimaryRevenue.RevnAmount","functionType":
 "sum" } ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="badgeItem" value="[[ [{status : "success", text : "On-Track", min :
 "0"}] ]]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="titleItem" value="[['My Todos']]"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="footerItem" value="Appointments"></oj-vb-fragment-param>
</oj-vb-fragment>
</template>
```

In your fragment code, be sure to replace `activities` and `ActivityId` with the appropriate values for your object name and primary key.

Also, replace the queryUuid value with the saved search's UUID value. Review the values for the other parameters, and adjust as necessary.

This table describes some of the parameters that you can provide for a metric card.

Parameters for Metric Card

Parameter Name	Description
resource	Specify resource details for the saved search, including object name, primary key, and endpoint.

Parameter Name	Description
	For a list of resource details for common Sales objects, see "Resource Details for Sales Objects" in *Add a List to the Redwood Sales Dashboard*.
query	Include criteria for querying the data. The metric card will display a count of records unless you include the "aggregate" parameter.
aggregate	Specify the function to aggregate the data for the metric card. Use this format: `[[ [{"field": "DealAmount","function": "sum/min/max etc" }] ]]` Use this parameter only if you don't want the metric card to provide a count of records. Instead, you want to provide a numerical value derived from the underlying saved search. Available functions to use in this parameter are: **`sum`**, **`min`**, **`max`**, and **`avg`**. The **`min`** and **`max`** functions display either the lowest or highest value of records returned from the saved search.
badgeItem	A metric card can include a badge. You can configure metric cards so that different badges display depending on certain conditions. For example, if a metric card displays 10 open leads, then maybe you'd like the warning badge to display. But if there are only 2 open leads, then maybe you don't want a badge to display at all. See *How to Add Metric Card Badges*.
titleItem	Specify the title of the metric card. This title displays on the dashboard at runtime.
footerItem	Optionally specify a footer for the metric card. The footer displays as a description at the bottom of the metric card.

How to Add Metric Card Badges

This setup is optional. If you skip this setup for a metric card, then no badges or descriptions will ever display on the card.

To configure badges on a metric card, you first determine the display conditions. Each condition is a range of aggregate sum numbers that display on the metric card. For each range, you decide which type of badge to display. For example, you can configure this type of setup:

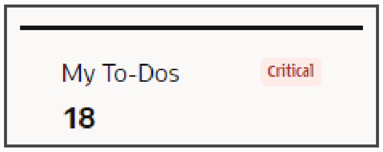
- If fewer than 5 overdue tasks exist, then don't show a badge at all.
- If 5-10 overdue tasks exist, then show a warning badge.
- If more than 10 overdue tasks exist, then show a critical badge.

When you define the display conditions for a metric card, the order of those conditions is important. Each condition is evaluated in the order in which you define them, from top to bottom. Whichever condition is satisfied first is the condition that's applied.

You can also configure the metric card so that the card always displays the same badge regardless of the aggregate sum on the card.

The following table lists the badge properties that you can use to define a set of badges and display conditions:

Badge Properties

Badge Property	Description
`status`	• danger • info • neutral • success • warning Each type of badge (`status`) displays with a specific color, which you can't change. Here's an example of a "danger" badge with the text "Critical": 
`min`	Lower limit of the display condition for a specific badge.
`max`	Upper limit of the display condition for a specific badge.
`text`	Text that displays on the badge. Each type of badge (`status`) displays with default text, but you can use this property to change it.

Here's a sample of the code that you can use to define a set of badges and display conditions:

```
[{status : "success", text : "All Good", min : "0", max : "10"}, {status : "warning", text : "Attention", min :
"11", max : "50"},{status : "danger", text : "Critical", min : "51"}]
```

- Specify the type of badge to display (`status`).
- Define the conditions (`min` and `max`) under which each badge (and optionally a description) will display.

- Indicate the badge text (`text`). For example, does the badge say **Warning** or **Problem**?

Tips:

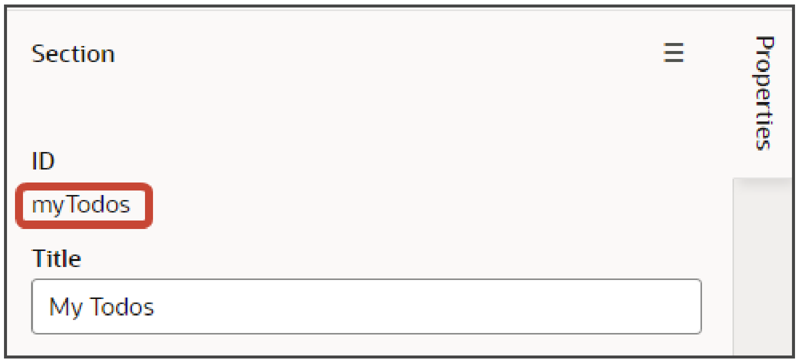
- You can specify only a lower limit without an upper limit (to indicate an aggregate sum that's greater than 10, for example).
- You can add metaText without any range conditions. So, if none of the ranges apply, then show a description on the metric card.

Associate a Dashboard Layout to Your New Metric Card

After you add a new metric card to a metric card layout, you must associate a dashboard layout to the card. Do this so that, when users click the metric card at runtime, the right dashboard page displays.

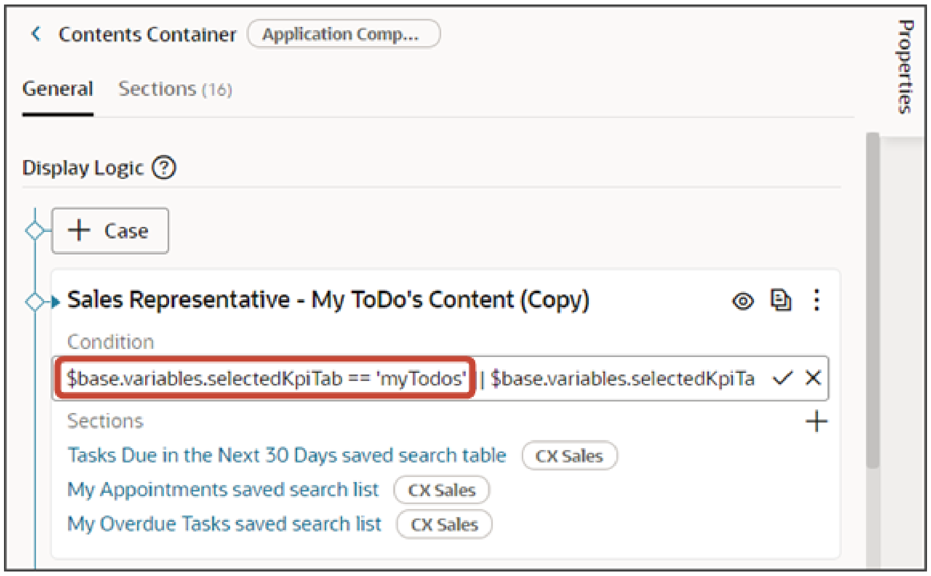
To associate a metric card with a dashboard layout:

1. Retrieve the metric card's ID that you earlier saved.



ID
myTodos
Title
My Todos

2. Navigate to the dashboard layout that you want to associate with the new metric card. For example:



Sales Representative - My ToDo's Content (Copy)
Condition
$base.variables.selectedKpiTab == 'myTodos' | $base.variables.selectedKpiTa
Sections
Tasks Due in the Next 30 Days saved search table CX Sales
My Appointments saved search list CX Sales
My Overdue Tasks saved search list CX Sales

3. Click the dashboard layout's condition to replace the existing metric card ID with the new ID.

   Use this format for the layout condition:

   ```
   $variable.selectedKpiTab == '<metric card id>'
   ```

   For example:

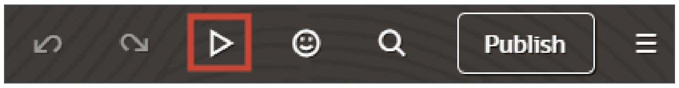
   ```
   $base.variables.selectedKpiTab == 'myTodos'
   ```

Test Your Changes

Test the new metric card to confirm the aggregate sum displays correctly. Also confirm that the right dashboard page displays when you click the card.

1. Test the modified dashboard layout by previewing your application extension.

   a. Click the Preview button to see your changes in your runtime test environment.

   b. The resulting preview link will be:

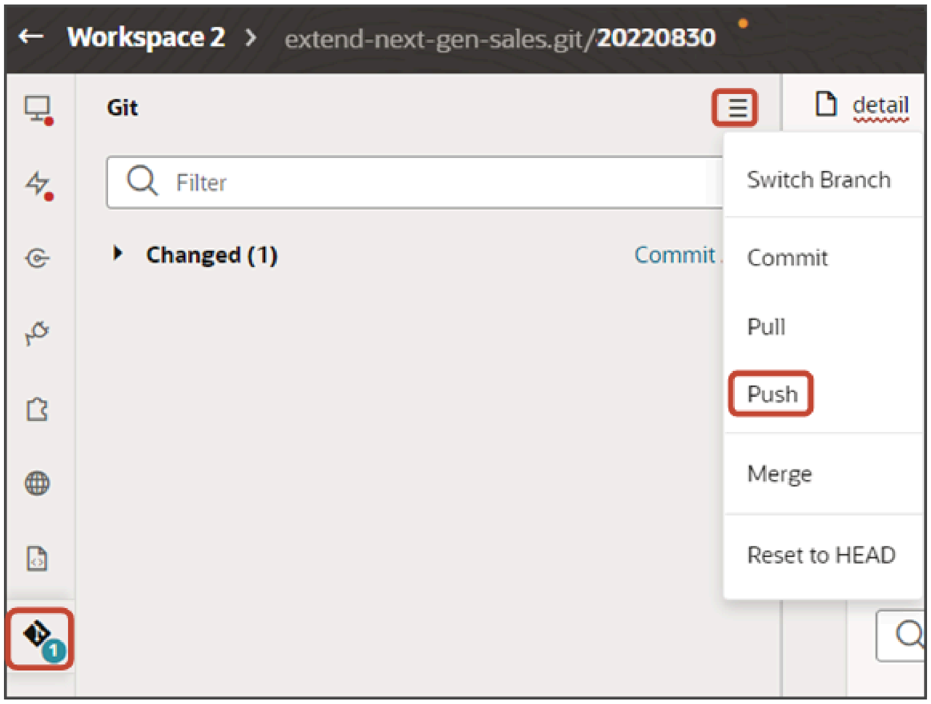
   ```
   https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
   ```

   c. Change the preview link as follows:

   ```
   https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
   ```

2. Save your work by using the Push Git command.

   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).

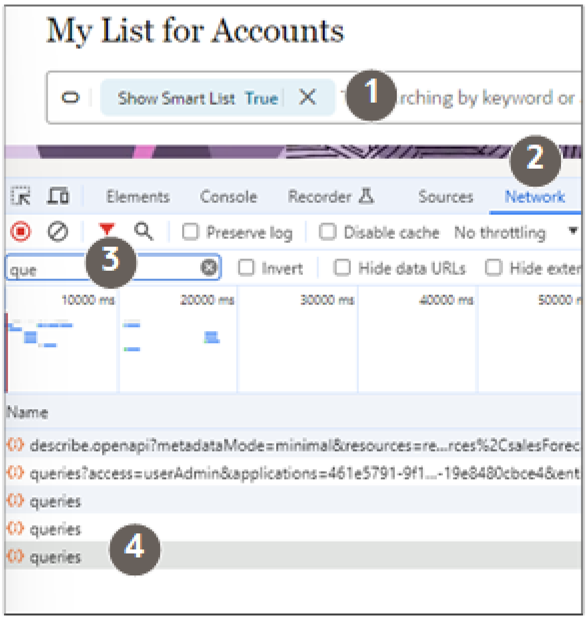


How can I find the UUID for a saved search in Sales in the Redwood User Experience?

Here's how to find the Universally Unique Identifier (UUID) for a saved search. This unique ID is required if you want to create a metric card from the saved search for the Sales Dashboard, for example. A UUID is a is a 36-character alphanumeric string that can be used to identify information, such as rows of data within a database table.

1. Open the list page of the object in the Chrome browser. If you need the UUID for accounts, for example, click **Accounts** on the home page.
2. Click in the action bar (callout 1 in the following screenshot) and select any saved search other that the one you're looking for.
3. Right-click the page and select **Inspect** from the Chrome browser menu.
4. Click the **Network** tab (callout 2).
5. Reload the browser page.
6. In the action bar, switch to the saved search you want.
7. Click the **Network** tab again.
8. Enter **que** in the filter field to filter out the queries (callout 3).

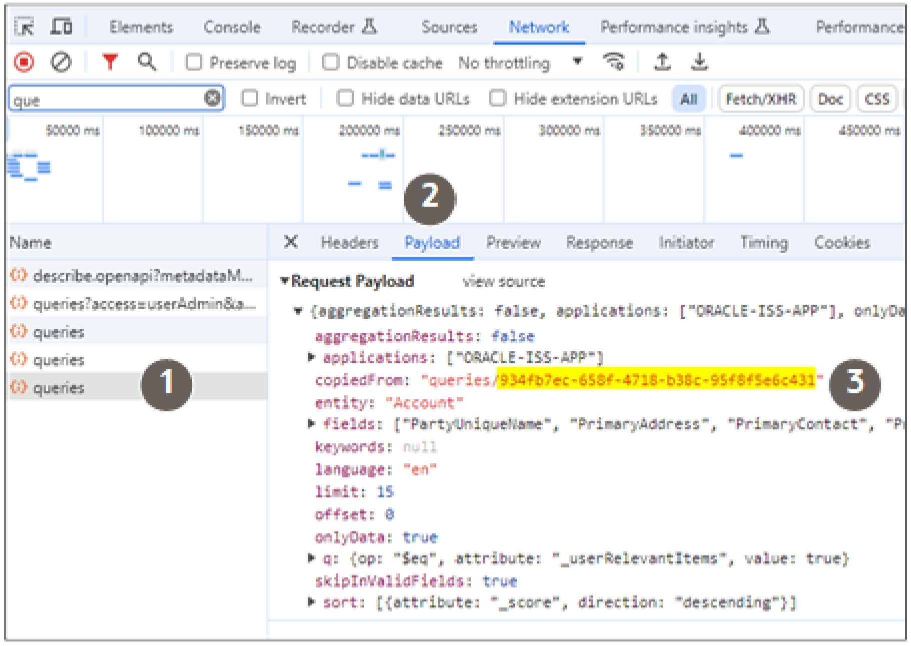
9. Select the last **queries** entry in the list (callout 4).



```
describe.openapi?metadataMode=minimal&resources=re...rces%2CsalesForec
queries?access=userAdmin&applications=461e5791-9f1...-19e8480cbce4&ent
queries
queries
queries
```

10. With the last queries item selected (callout 1 in the following screenshot), click the **Payload** tab (callout 2).

11. The UUID is the string of characters following **queries/** in the **copiedFrom:** line. For example: `934fb7ec-658f-4718-b38c-95f8f5e6c431` (highlighted).



```
describe.openapi?metadataM...
queries?access=userAdmin&a...
queries
queries
queries
```

```
▼Request Payload      view source
 ▼ {aggregationResults: false, applications: ["ORACLE-ISS-APP"], onlyDa
     aggregationResults: false
   ▶ applications: ["ORACLE-ISS-APP"]
     copiedFrom: "queries/934fb7ec-658f-4718-b38c-95f8f5e6c431"
     entity: "Account"
   ▶ fields: ["PartyUniqueName", "PrimaryAddress", "PrimaryContact", "P
     keywords: null
     language: "en"
     limit: 15
     offset: 0
     onlyData: true
   ▶ q: {op: "$eq", attribute: "_userRelevantItems", value: true}
     skipInValidFields: true
   ▶ sort: [{attribute: "_score", direction: "descending"}]
```

Add a Visualization to the Redwood Sales Dashboard

You can add visualizations of saved searches or Oracle Analytics Cloud analyses to the Sales Dashboard. You can pick from a set of predefined visualizations, or you can configure custom visualizations using the Visualization Configurations page. This topic illustrates how to add both predefined and custom visualizations to the Sales Dashboard.

Types of Visualizations That You Can Add

You can add both predefined and custom visualizations to the Sales Dashboard:

- You can add these predefined visualizations:
  - Lead Amounts by Team Member Chart
  - My Team's Performance Chart

  See *Add a Predefined Visualization to the Sales Dashboard*.
- You can add custom visualizations that you create using the Visualization Configurations page.

  See *Add a Custom Visualization to the Sales Dashboard*.

  > **Note:** You can also add reports created using the Express Reports tool. For more informaiton about express reporting, contact your Oracle representative.

Prerequisites

- To add a custom visualization to the Sales Dashboard, you must first create it.

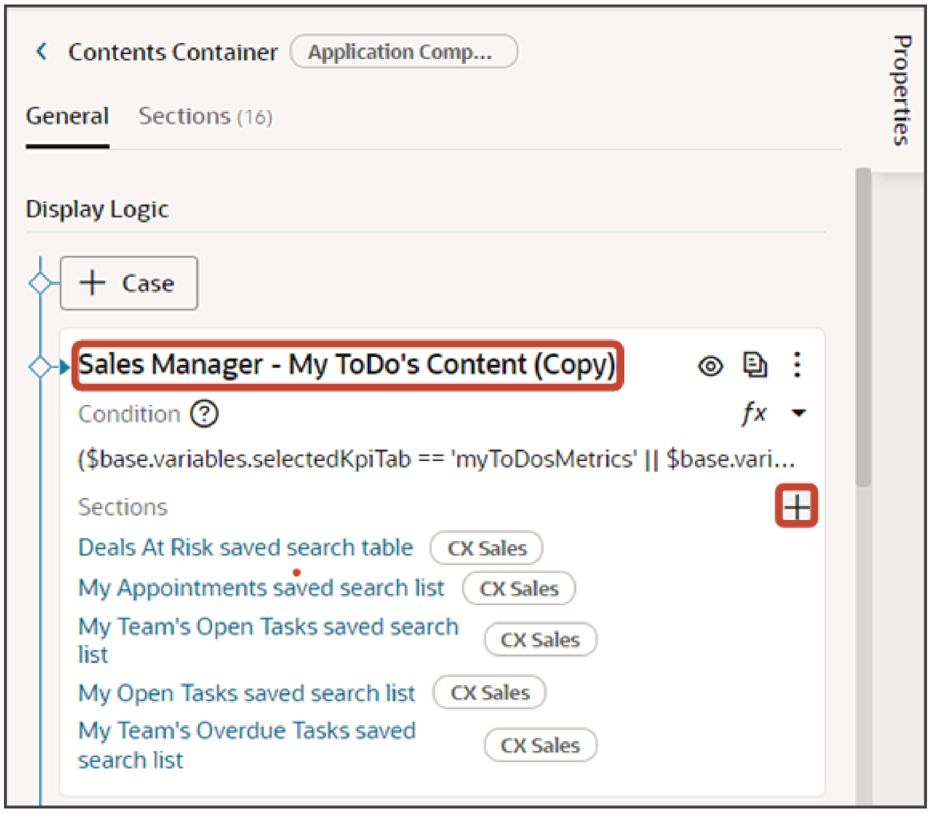
  See *Create a Visualization of a Saved Search or an Analysis*.
- Sales Dashboard layouts are read only in Visual Builder Studio. To modify a layout, you must first create a copy.

  Duplicate the **Sales Manager - My ToDo's Content** dashboard layout as described in *Configure Redwood Sales Dashboard Layouts* so that you have a layout that's ready to modify.

Add a Predefined Visualization to the Sales Dashboard

This example illustrates how to add a predefined visualization to the Sales Dashboard for sales managers.

1. On the Properties pane for the **Sales Manager - My ToDo's Content (Copy)** layout, click the **+** Add Section icon.

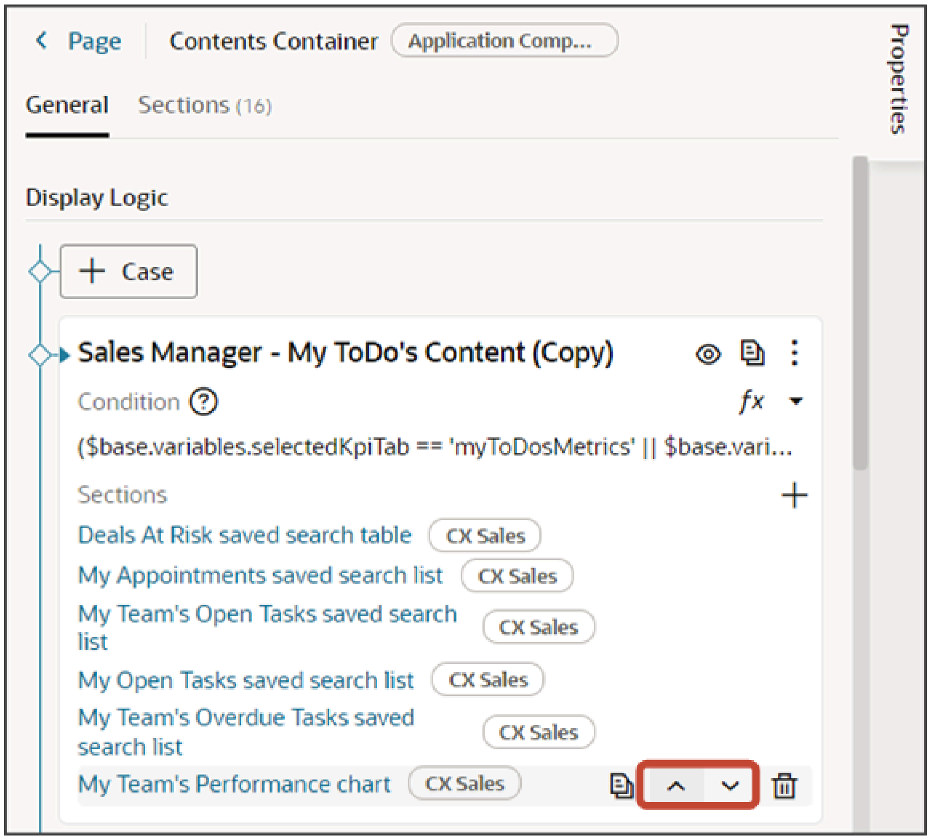


($base.variables.selectedKpiTab == 'myToDosMetrics' || $base.vari...

Deals At Risk saved search table	CX Sales
My Appointments saved search list	CX Sales
My Team's Open Tasks saved search list	CX Sales
My Open Tasks saved search list	CX Sales
My Team's Overdue Tasks saved search list	CX Sales

2. From the list of available components that display, click one of the predefined visualizations, such as **My Team's Performance Chart**.

   The newly added chart is added at the bottom of the layout. You can change the order the sections are displayed by using the Move Up and Move Down arrows in the Sections region.



< Page | Contents Container | Application Comp...

General | Sections (16)

Properties

Display Logic

+ Case

Sales Manager - My ToDo's Content (Copy)

Condition ? fx

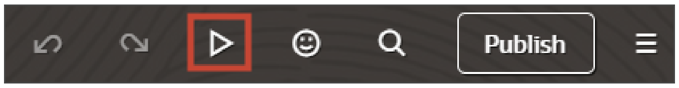
($base.variables.selectedKpiTab == 'myToDosMetrics' || $base.vari...

Sections +

Deals At Risk saved search table	CX Sales
My Appointments saved search list	CX Sales
My Team's Open Tasks saved search list	CX Sales
My Open Tasks saved search list	CX Sales
My Team's Overdue Tasks saved search list	CX Sales
My Team's Performance chart	CX Sales

3. Test the modified dashboard layout by previewing your application extension.

   a. Click the Preview button to see your changes in your runtime test environment.



   b. The resulting preview link will be:

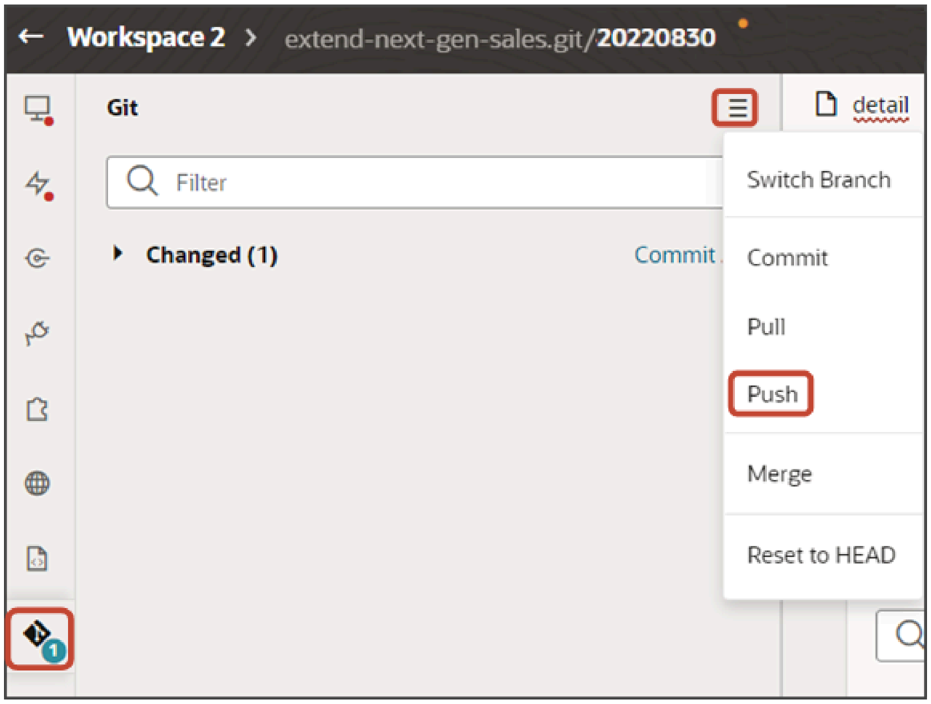
```
https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
```

   c. Change the preview link as follows:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
```

4. Save your work by using the Push Git command.

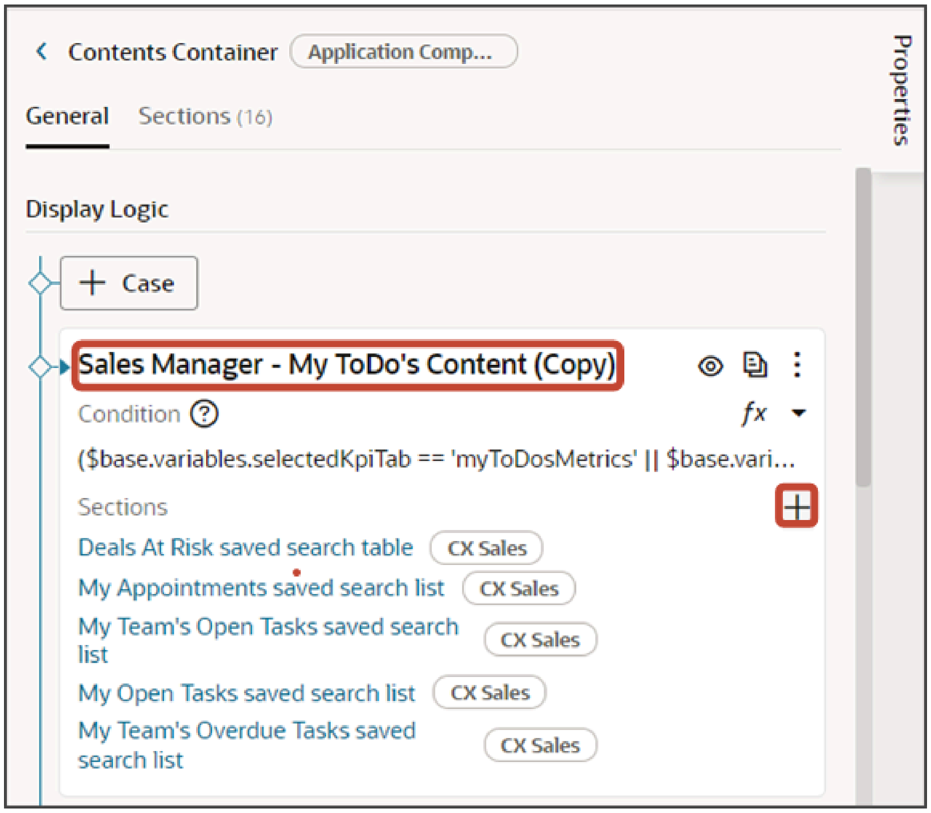
   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).



Add a Custom Visualization to the Sales Dashboard

If you previously created your own visualizations using the Visualization Configurations page, then you can add them to any Sales Dashboard layout:

1. On the Properties pane for the **Sales Manager - My ToDo's Content (Copy)** layout, click the **+** Add Section icon > **+ New Section**.

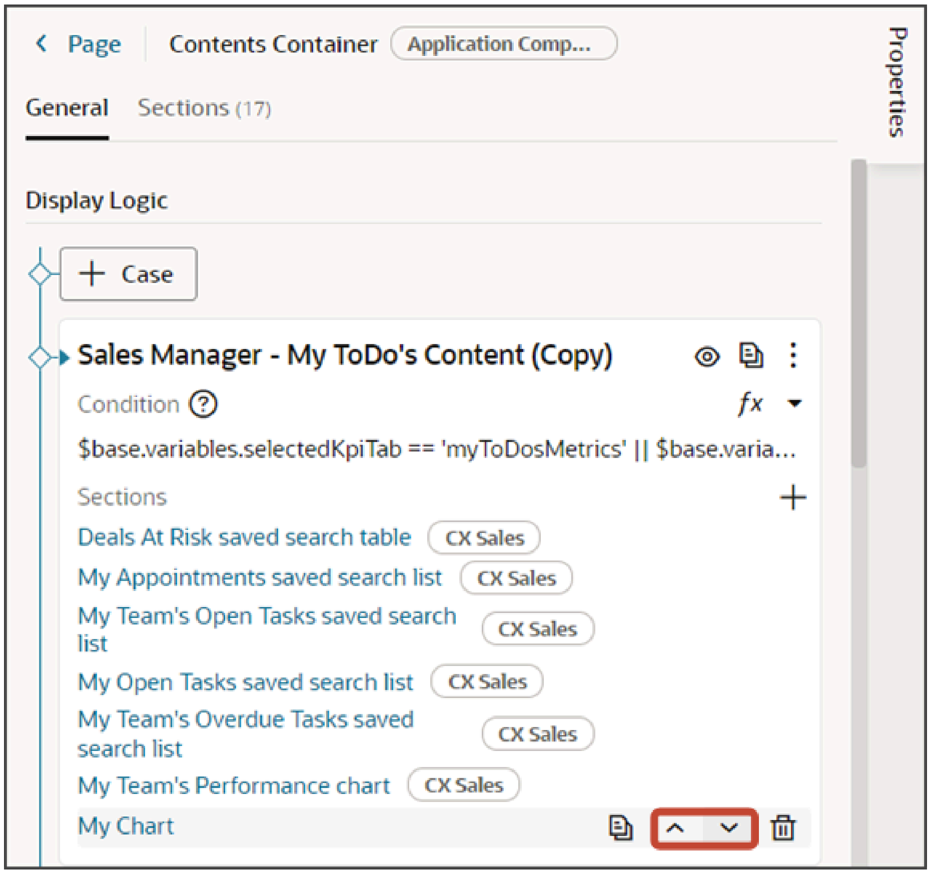

Contents Container
Application Comp...
General
Sections (16)
Properties
Display Logic
Case
Sales Manager - My ToDo's Content (Copy)
Condition
fx
Sections
Deals At Risk saved search table
CX Sales
My Appointments saved search list
CX Sales
My Team's Open Tasks saved search list
CX Sales
My Open Tasks saved search list
CX Sales
My Team's Overdue Tasks saved search list
CX Sales

($base.variables.selectedKpiTab == 'myToDosMetrics' || $base.vari...

2. In the Create Section dialog, in the Title field, enter `My Chart`, and click **OK**.

   > **Note:** The title is the name of the component, not how the component will appear at runtime on the dashboard. The runtime title comes from the `dashboardTitle` parameter that you add to the visualization fragment. See step 7.

   The newly added section is added to the bottom of the layout. Use the Move Up arrow to move the new section to the desired location.

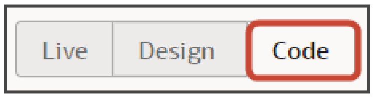
   $base.variables.selectedKpiTab == 'myToDosMetrics' || $base.varia...

   Deals At Risk saved search table
   My Appointments saved search list
   My Team's Open Tasks saved search list
   My Open Tasks saved search list
   My Team's Overdue Tasks saved search list
   My Team's Performance chart

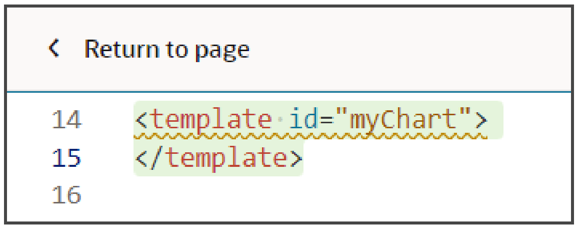
3. Click the **My Chart** link.

   The template editor opens.

4. Click the Code button.

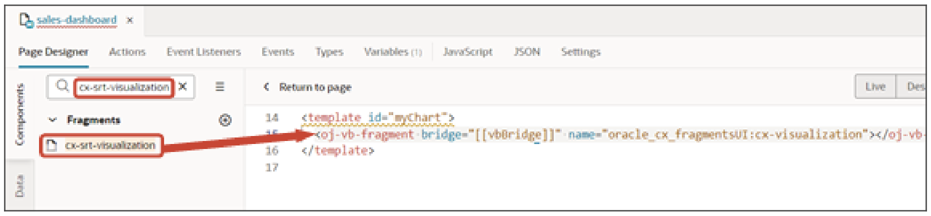


Your new **My Chart** section displays with empty placeholder `template` tags.



```
<template id="myChart">
</template>
```

5. On the Components palette, in the Filter field, enter `cx-srt-visualization`.
6. Drag and drop the cx-srt-visualization fragment to the template editor, between the template tags.



```
<template id="myChart">
<oj-vb-fragment bridge="[[vbBridge]]" name="oracle_cx_fragmentsUI:cx-visualization"></oj-vb-
</template>
```

7. Add the following parameters to the fragment code so that the code looks like the below sample:

```
<template id="myChart">
 <oj-vb-fragment name="oracle_cx_fragmentsUI:cx-visualization" class="oj-flex oracle-cx-fragmentsUI-cx-
fragment-full-height" bridge="[[vbBridge]]">
 <oj-vb-fragment-param name="reportNumber" value="CDRM_1"></oj-vb-fragment-param>
 <oj-vb-fragment-param name="context" value='{"mode": "dashboard","source":"DV","dashboardTitle":"DV
 Adaptive Chart"}'></oj-vb-fragment-param>
 </oj-vb-fragment>
</template>
```

In your fragment code, be sure to replace the values for the `reportNumber`, `source`, and `dashboardTitle` parameters with the appropriate values for your custom visualization.

This table describes some of the parameters that you can provide for a custom visualization.

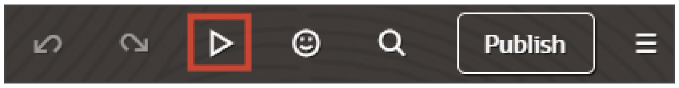
Parameters for a Custom Visualization

Parameter Name	Description
reportNumber	Enter the reference number of the visualization that you previously created.
mode	This value should always be `dashboard`.
source	Enter either `DV` or `SRT`.

Parameter Name	Description
	`SRT` refers to reports created using the Express Reports tool. Contact your Oracle representative for more information. DV charts are charts that you created on the Visualization Configurations page.
dashboardTitle	Specify the title of the chart. This title displays on the dashboard at runtime.

8. Test the modified dashboard layout by previewing your application extension.

   a. Click the Preview button to see your changes in your runtime test environment.

   b. The resulting preview link will be:

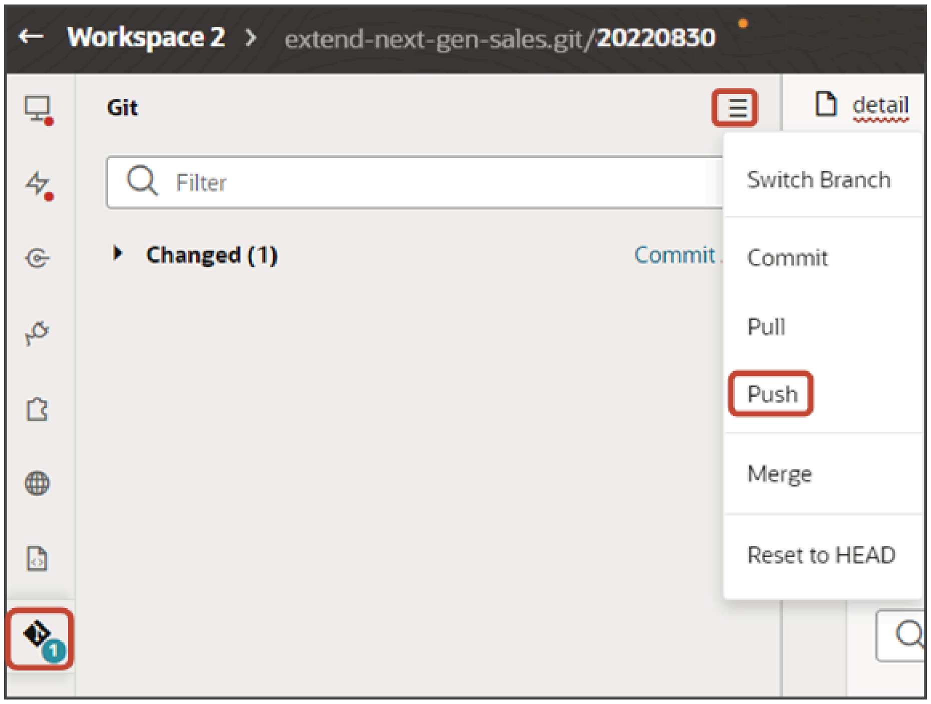
   ```
   https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
   ```

   c. Change the preview link as follows:

   ```
   https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
   ```

9. Save your work by using the Push Git command.

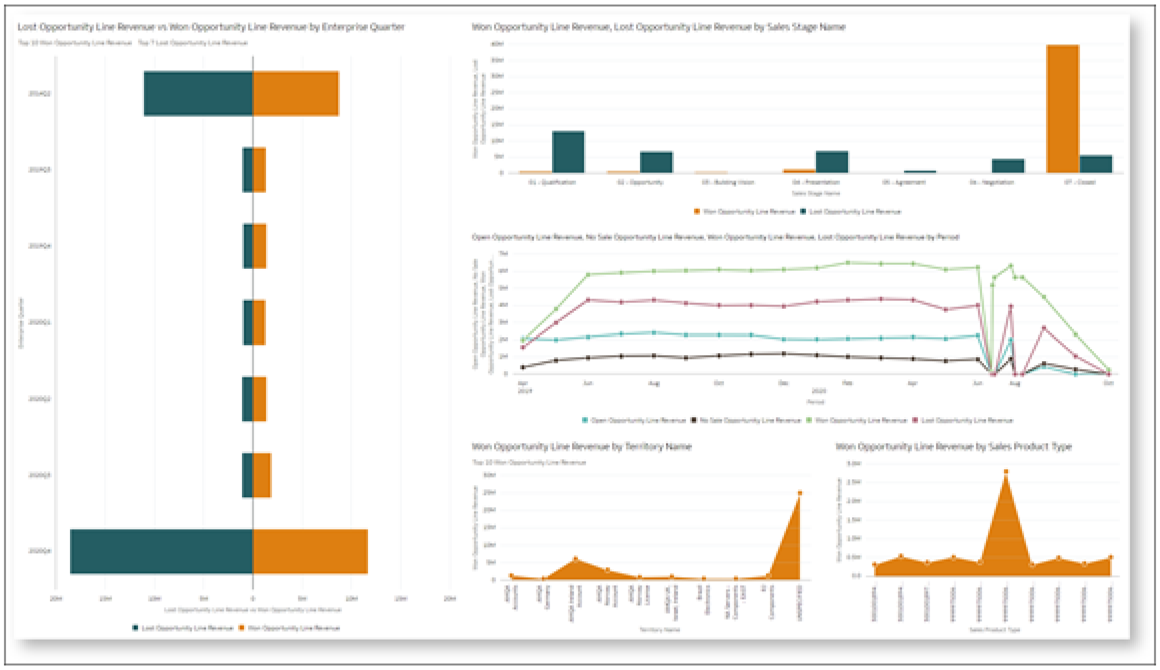
   Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).



Add an Oracle Analytics Component to the Redwood Sales Dashboard

You can embed content created in Oracle Analytics Cloud into the Sales Dashboard using Oracle Visual Builder Studio.

Here's an example of an Oracle Analytics component:


Lost Opportunity Line Revenue vs Won Opportunity Line Revenue by Enterprise Quarter
Won Opportunity Line Revenue, Lost Opportunity Line Revenue by Sales Stage Name
Open Opportunity Line Revenue, No Sale Opportunity Line Revenue, Won Opportunity Line Revenue, Lost Opportunity Line Revenue by Period
Won Opportunity Line Revenue by Territory Name
Won Opportunity Line Revenue by Sales Product Type

For more information about the Oracle Analytics embedding framework, see the Oracle Cloud Visualizing Data and Building Reports in Oracle Analytics Cloud guide.

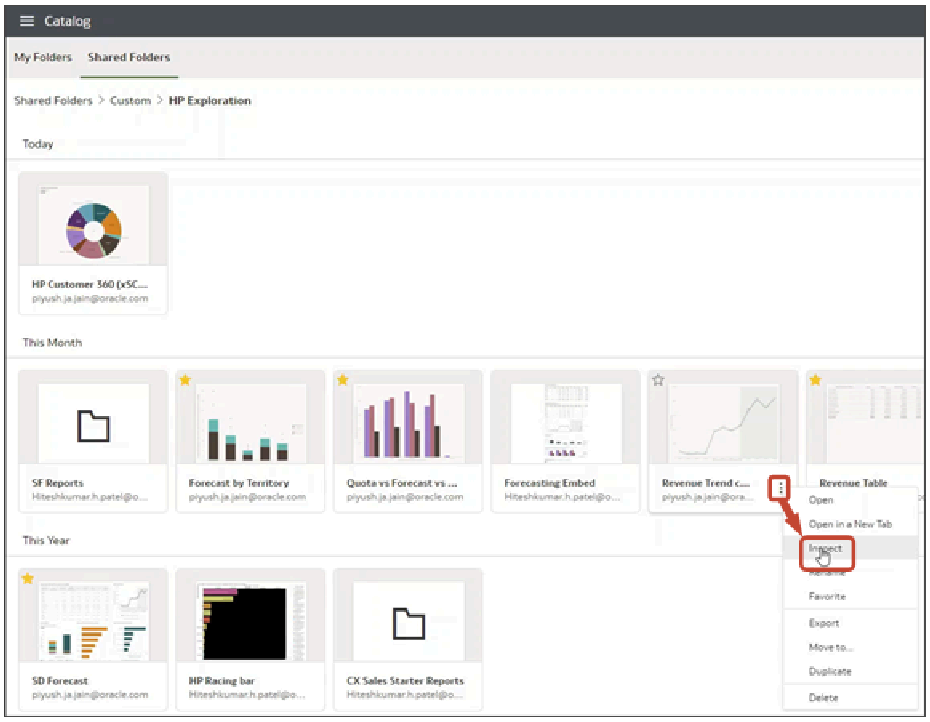
Prerequisites

Before adding an Oracle Analytics component to the Sales Dashboard, you must:

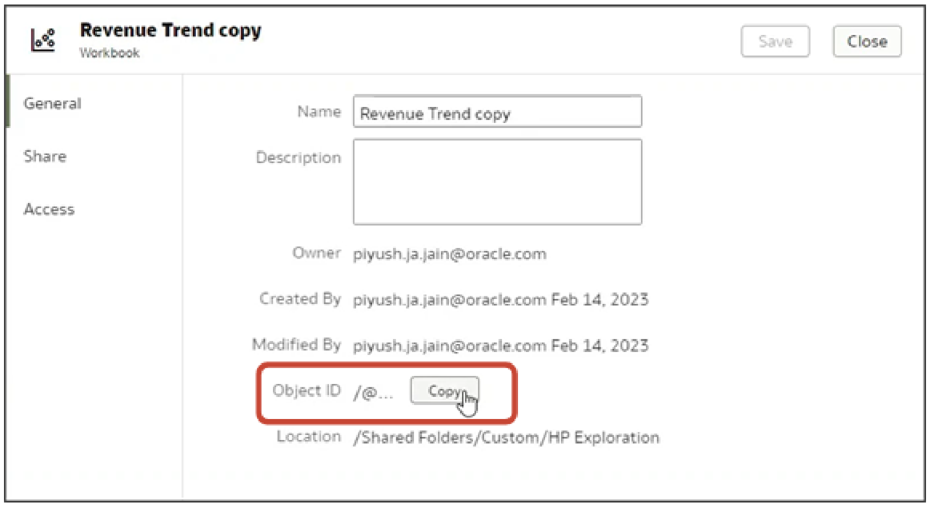
1. Retrieve the Object ID from your existing Oracle Analytics content:

   a. Navigate to the Oracle Analytics Host where the analytics workbook is saved.

   b. Click the Actions menu > **Inspect**.



   c. Next to the Object ID, click **Copy**.



Revenue Trend copy
Workbook

Save
Close

General
Share
Access

Name: Revenue Trend copy
Description:
Owner: piyush.ja.jain@oracle.com
Created By: piyush.ja.jain@oracle.com Feb 14, 2023
Modified By: piyush.ja.jain@oracle.com Feb 14, 2023
Object ID: /@... Copy
Location: /Shared Folders/Custom/HP Exploration

   d. Paste the Object ID, along with the Oracle Analytics Host URL, into a separate file for later use. For example:

      i. Oracle Analytics Host:

```
https://oac01-gse00010001-px.analytics.ocp.oraclecloud.com/
```

ii. Object ID:

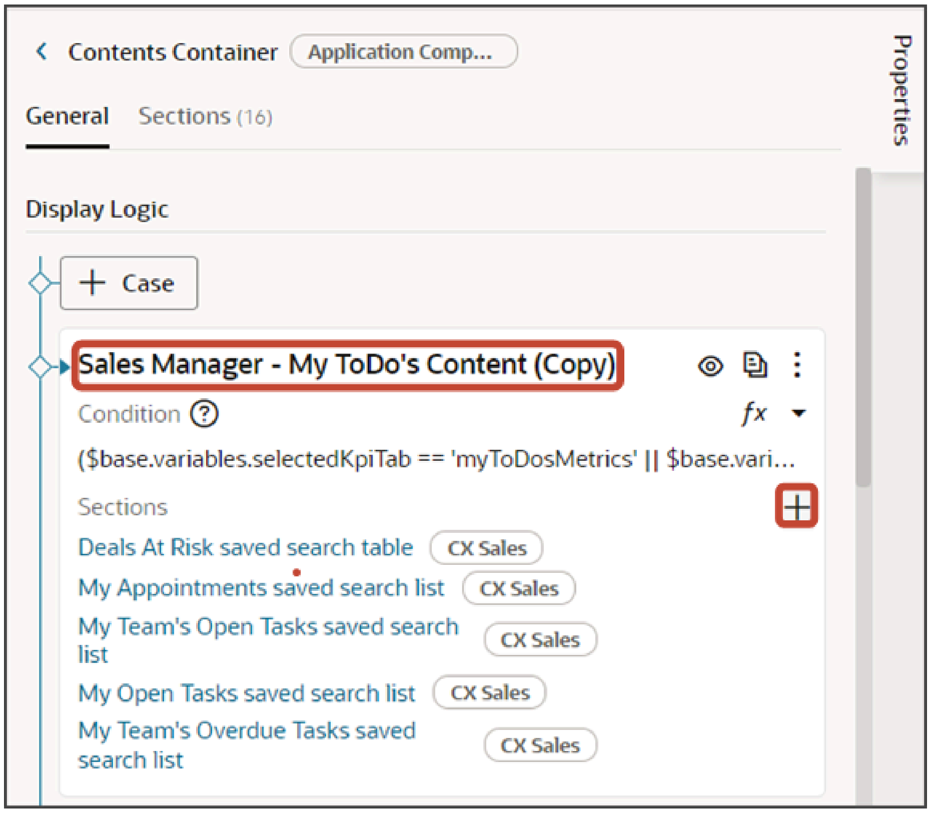
```
/@Catalog/shared/Custom/HP Exploration/Revenue Trend copy
```

2. Sales Dashboard layouts are read only in Visual Builder Studio. To modify a layout, you must first create a copy.

   Duplicate the **Sales Manager - My ToDo's Content** dashboard layout as described in *Configure Redwood Sales Dashboard Layouts* so that you have a layout that's ready to modify.

Add an Oracle Analytics Component to the Sales Dashboard

1. On the Properties pane for the **Sales Manager - My ToDo's Content (Copy)** layout, click the **+** Add Section icon.

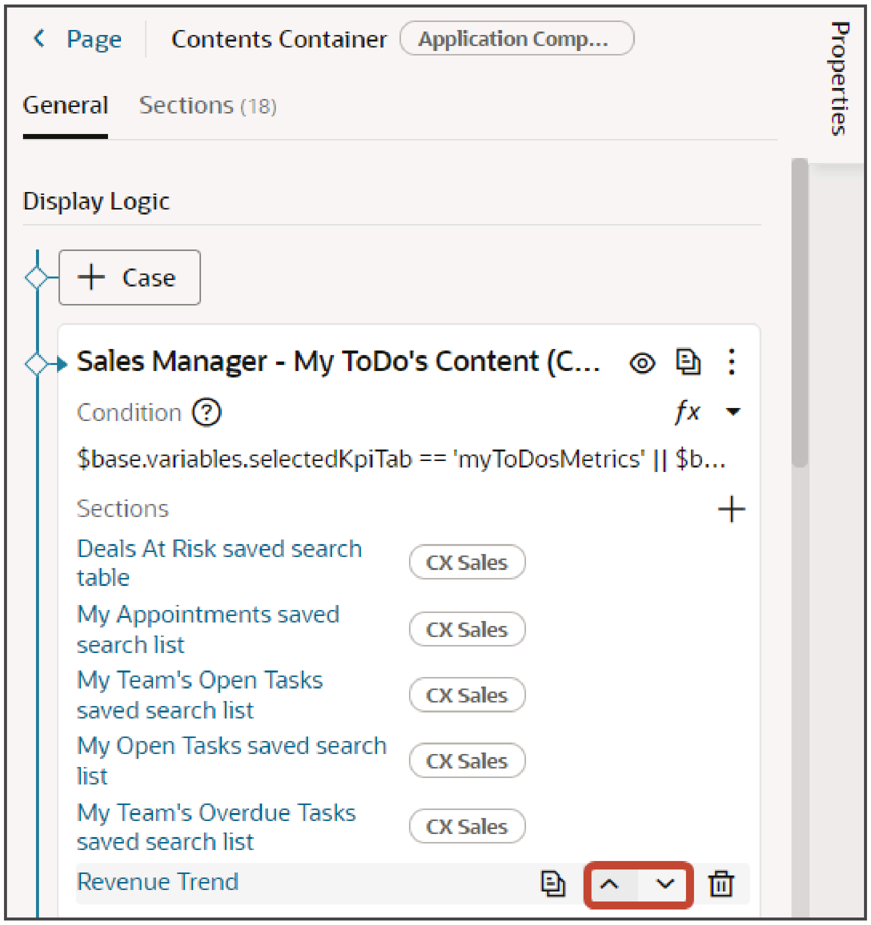

Contents Container
Application Comp...
Properties
General
Sections (16)
Display Logic
Case
Sales Manager - My ToDo's Content (Copy)
Condition
fx
Sections
Deals At Risk saved search table
CX Sales
My Appointments saved search list
CX Sales
My Team's Open Tasks saved search list
CX Sales
My Open Tasks saved search list
CX Sales
My Team's Overdue Tasks saved search list
CX Sales

($base.variables.selectedKpiTab == 'myToDosMetrics' || $base.vari...

2. In the Create Section dialog, in the Title field, enter `Revenue Trend`, and click **OK**.

   > **Note:** The title is the name of the component, not how the component will appear at runtime on the dashboard. The runtime title comes from the `Panel Title` property that you specify for the Dashboard Panel fragment. See step 10.

   The newly added section is added to the bottom of the layout. Use the Move Up arrow to move the new section to the desired location.



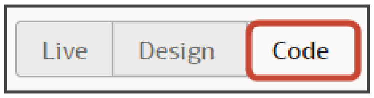
$base.variables.selectedKpiTab == 'myToDosMetrics' || $b...

Deals At Risk saved search table	CX Sales
My Appointments saved search list	CX Sales
My Team's Open Tasks saved search list	CX Sales
My Open Tasks saved search list	CX Sales
My Team's Overdue Tasks saved search list	CX Sales
Revenue Trend	

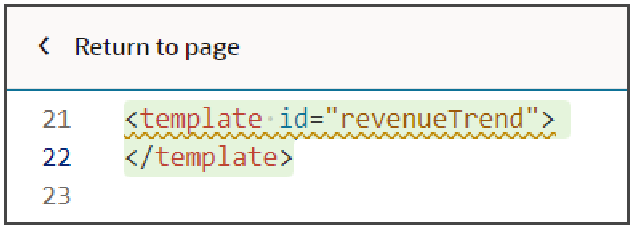
3. Click **Revenue Trend**.

   The template editor opens.

4. Click the Code button.

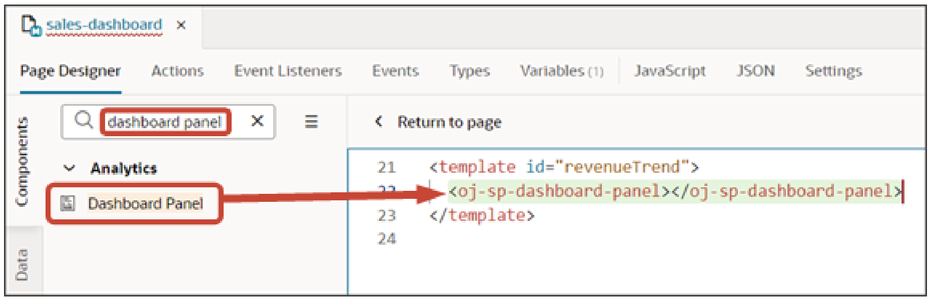


Your new **Revenue Trend** section displays with empty placeholder `template` tags.



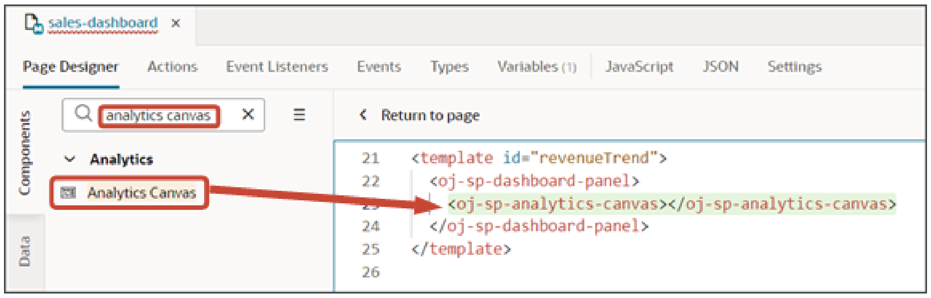
```
<template id="revenueTrend">
</template>
```

5. On the Components palette, in the Filter field, enter `dashboard panel`.
6. Drag and drop the Dashboard Panel fragment to the template editor, between the template tags.



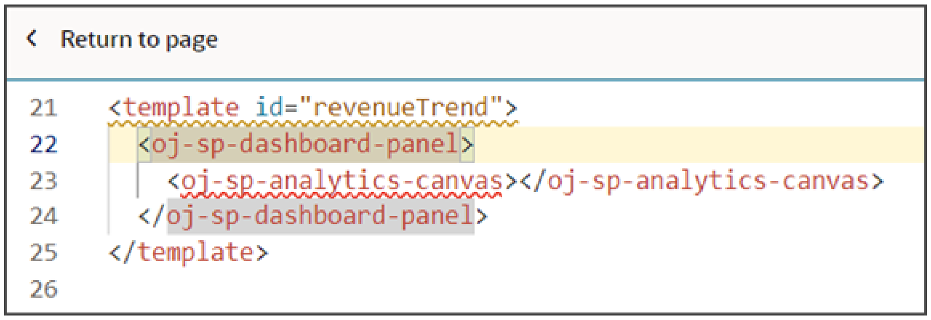
```
<template id="revenueTrend">
  <oj-sp-dashboard-panel></oj-sp-dashboard-panel>
</template>
```

7. On the Components palette, in the Filter field, enter `analytics canvas`.
8. Drag and drop the Analytics Canvas fragment to the template editor, between the `oj-sp-dashboard-panel` template tags.



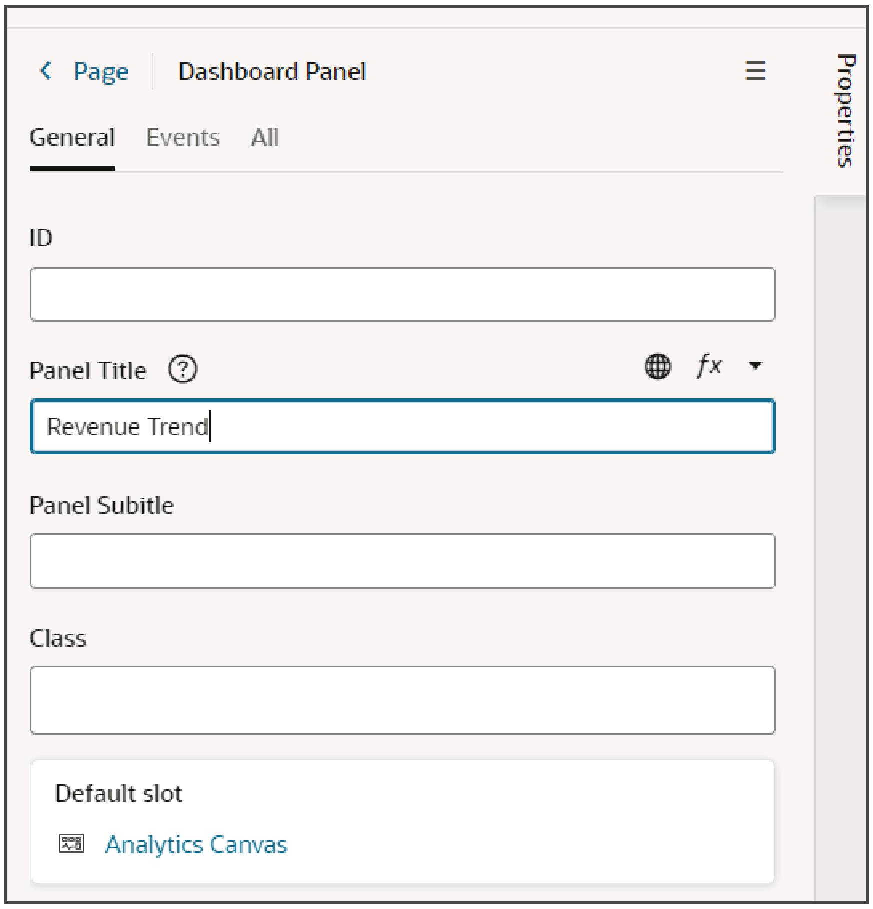
```
<template id="revenueTrend">
  <oj-sp-dashboard-panel>
    <oj-sp-analytics-canvas></oj-sp-analytics-canvas>
  </oj-sp-dashboard-panel>
</template>
```

9. In the template editor, click the `oj-sp-dashboard-panel` template tags.



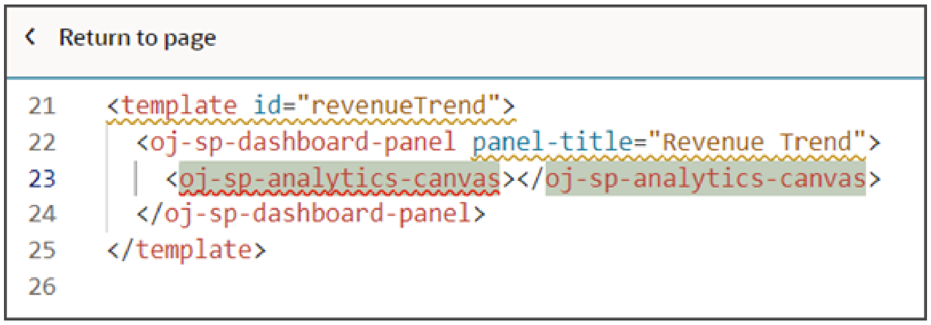
```
<template id="revenueTrend">
  <oj-sp-dashboard-panel>
    <oj-sp-analytics-canvas></oj-sp-analytics-canvas>
  </oj-sp-dashboard-panel>
</template>
```

10. On the Properties pane, in the Panel Title field, enter the title that you want to display on the dashboard for the analytic, such as `Revenue Trend`.



Panel Title: Revenue Trend

11. In the template editor, click the `oj-sp-analytics-canvas` template tags.

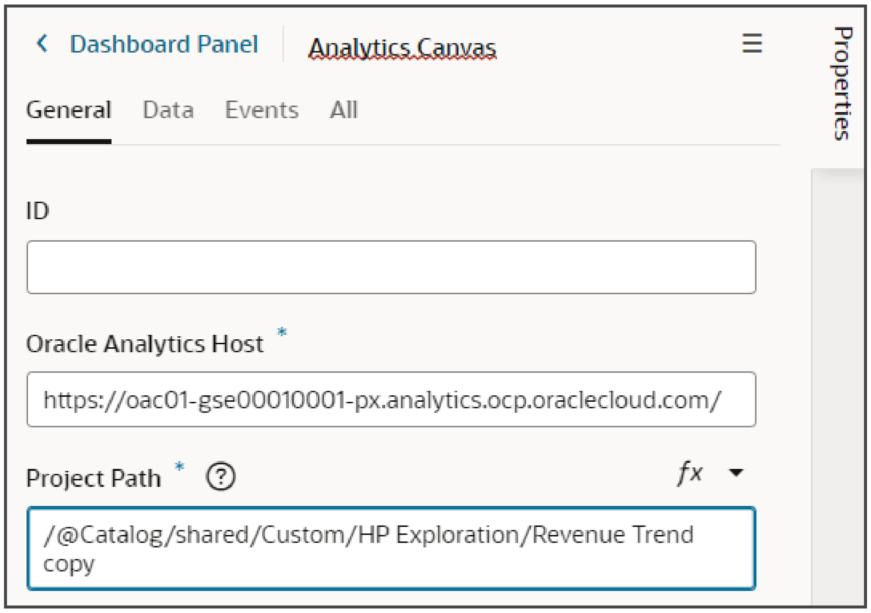


```
<template id="revenueTrend">
  <oj-sp-dashboard-panel panel-title="Revenue Trend">
    <oj-sp-analytics-canvas></oj-sp-analytics-canvas>
  </oj-sp-dashboard-panel>
</template>
```

12. On the Properties pane, in the Oracle Analytics Host field, enter the host URL that you previously saved, such as `https://oac01-gse00010001-px.analytics.ocp.oraclecloud.com/`.

13. In the Project Path field, enter the repository path of the project to render, such as `/@Catalog/shared/Custom/HP Exploration/Revenue Trend copy`.

    Note: This is the Object ID of the Oracle Analytics workbook that you previously saved.



ID

Oracle Analytics Host *

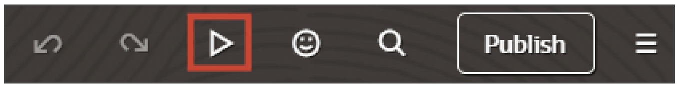
https://oac01-gse00010001-px.analytics.ocp.oraclecloud.com/

Project Path * ? fx

/@Catalog/shared/Custom/HP Exploration/Revenue Trend copy

14. Test the modified dashboard layout by previewing your application extension.

    a. Click the Preview button to see your changes in your runtime test environment.



    b. The resulting preview link will be:

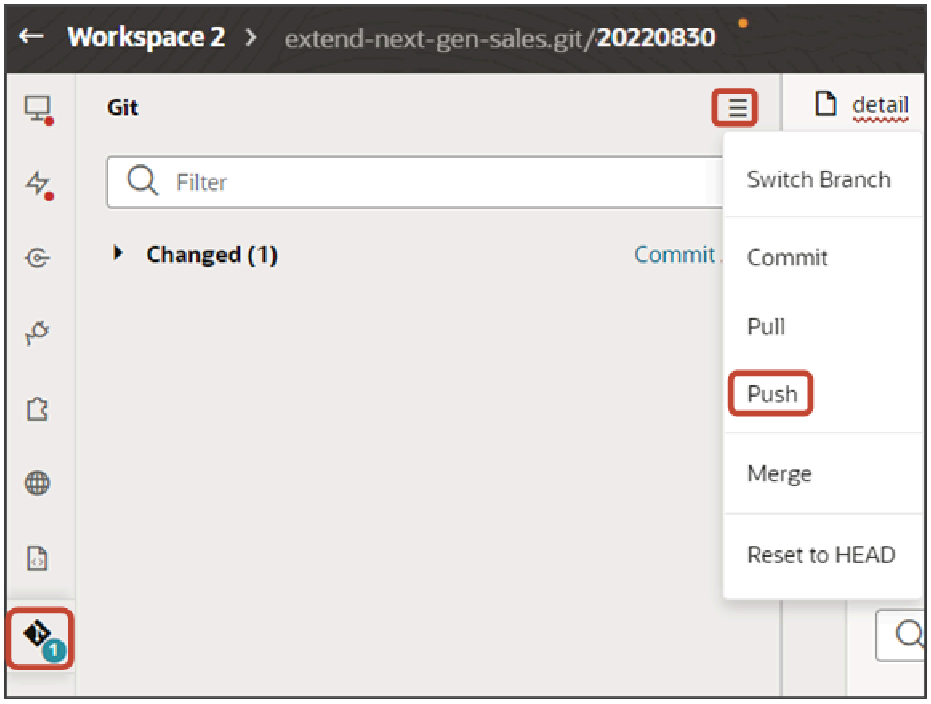
```
https://<servername>/fscmUI/redwood/cx-sales/dashboards/sales-dashboard
```

    c. Change the preview link as follows:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/dashboards/sales-dashboard
```

15. Save your work by using the Push Git command.

    Navigate to the Git tab, review your changes, and do a Git push (which does both a commit and a push to the Git repository).


Workspace 2
extend-next-gen-sales.git/20220830
Git
Filter
Changed (1)
Commit
detail
Switch Branch
Commit
Pull
Push
Merge
Reset to HEAD