Oracle Fusion Cloud Sales Automation

How can I modify standard Sales pages in Oracle Visual Builder Studio?

Oracle Fusion Cloud Sales Automation
How can I modify standard Sales pages in Oracle Visual Builder Studio?

F88585-04

Author: Kristin Penaskovic

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 Modify Pages in Visual Builder Studio

Before your team can start creating application extensions, you must first set up Visual Builder Studio and configure it. You only need to complete these steps once per implementation.

Before you begin, make sure you have the following information:

- The URL of your instance of Visual Builder Studio

  You can obtain the URL by navigating from your Oracle Cloud Applications development instance. From the Navigator, under **Configuration**, select **Visual Builder**.

Setting up Visual Builder Studio includes these steps:

1. Obtain the required roles and privileges.
2. Set up Visual Builder Studio users.
3. Create a project and workspace.

   Or, create a workspace in an existing project.
4. Create the build pipeline to deploy your changes to your production environment.

Get the Required Roles and Privileges

To set up Visual Builder Studio and your Oracle Cloud Applications instances, make sure you're assigned the FND_ADMINISTER_SANDBOX_PRIV privilege. Each role listed below, for example, has this privilege:

- Application Administrator (ORA_FND_APPLICATION_ADMINISTRATOR_JOB)
- Application Developer (ORA_FND_APPLICATION_DEVELOPER_JOB)
- Sales Administrator (ORA_ZBS_SALES_ADMINISTRATOR_JOB)
- Customer Relationship Management Application Administrator (ORA_ZCA_CUSTOMER_RELATIONSHIP_MANAGEMENT_APPLICATION_ADMINISTRATOR_JOB)

In addition, in order to view the **Edit Page in Visual Builder Studio** option in the Settings and Actions menu, users must be assigned the View Administration Link privilege (FND_VIEW_ADMIN_LINK_PRIV). By default, this privilege is assigned to the predefined roles listed above. Grant this privilege to custom roles, as well, where required.

Set Up Visual Builder Studio Users in IDCS

First, you must authorize users so they can access Visual Builder Studio to do extension work. You do this in Oracle Identity Cloud Service (IDCS).

To sign in to the right instance of IDCS, complete these steps:

1. Navigate to www.oracle.com.
2. Click **View Accounts**.
3. Click **Sign in to Cloud**.
4. In the Cloud Account Name field, enter the name of your production pod, such as **cakp**. Click **Next**.

5. The Oracle Cloud Account Sign In page is specific to the Cloud Portal and is separate from your test and production pods. Enter your user credentials and click **Sign In**, or use the **Click here** link to reset your password.
   On the Oracle Cloud Dashboard, navigate to the Active Applications region and click **Identity Cloud**. Scroll down to view your IDCS service instances:
   - Identity
     IDCS instance associated with the Myservices portal.
   - CAKP (for example)
     IDCS instance associated with your production pod.
   - CAKPTEST (for example)
     IDCS instance associated with your developer pod.
6. To sign in to the IDCS instance associated with your test environment, click the Service Instance URL for your test pod (in this example, CAKPTEST), or click the **Open Service Console** link.

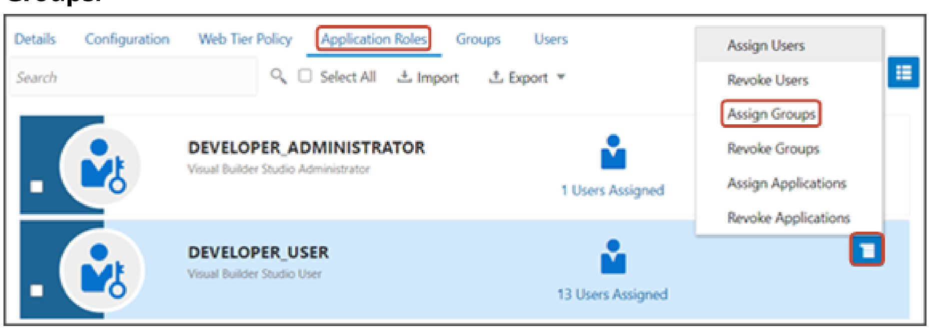
   Tip: Bookmark this URL so you have it for next time.

7. Sign in to IDCS directly, or click **Oracle Applications Cloud (Fusion)** at the bottom of the page to sign in using your Oracle Applications user credentials.

Once you're in the Identity Cloud Service console, you can now authorize users so they can access Visual Builder Studio. You can assign users, or groups of users, to the Visual Builder Studio administrator (DEVELOPER_ADMINISTRATOR) or Visual Builder Studio user (DEVELOPER_USER) roles. These two predefined roles are specific to working with Visual Builder Studio.

In this example, let's look at assigning a group to the Visual Builder Studio administrator (DEVELOPER_ADMINISTRATOR) role. This is a way to provide Visual Builder Studio access to Fusion Application roles and their users.

1. In the Identity Cloud Service console, expand the Navigation Drawer, and then click **Oracle Cloud Services**.
2. Click **DevServiceAppAUTO_<VBSTUDIO_INSTANCE>**.
3. Click the Application Roles tab.
4. On either the DEVELOPER_ADMINISTRATOR or DEVELOPER_USER row, click the menu icon, then click **Assign Groups**.



5. In the Assign Groups dialog, select the roles that you want to assign and click **OK**. For example:
   - Application Administrator
   - Sales Administrator
   - Customer Relationship Management Application Administrator

- Application Developer
- Any custom role, per your authorization plan

Users with the selected roles can now access Visual Builder Studio. Note that sometimes it may take up to 12 hours for Oracle Cloud Applications user updates to sync with IDCS.

For more information, see *Set Up VB Studio Users* in the Oracle Cloud Administering Visual Builder Studio guide.

Create Your Project and Workspace

If you haven't yet created a project and workpace in Visual Builder Studio, then you must do so before configuring a standard object's detail page.

- A **project** is a collection of resources that your team will use to create the application extension.

  > **Note:** The procedures in this chapter use fragments which are building blocks that help you to configure pages more quickly, with a minimal amount of manual coding. Create a new project to keep your fragment-based extension separate from extensions that don't use fragments.

- A **workspace** exists within a project, and is your private work area which connects to a Git repository, environment, and Oracle Cloud Applications sandbox.

If you **already** created a project, then you don't need to create a new one. Rather, you can create a new workspace within your existing project. See the next section.

To create a project and workspace at the same time:

1. In Visual Builder Studio, click the Organization side tab, then click the Projects subtab.
2. Click **+ Create**.
3. In the New Project dialog, enter a name and description for the project.
4. Set the security for your project, either **Private** or **Shared**.
5. Click **Next**.
6. Select **Empty Project** as the template for this project, and click **Next**.
7. On the Properties step, click **Next**.
8. On the Team step, add team members and select their membership levels.
9. Click **Finish**.

   Visual Builder Studio provisions your project, which could take a few minutes to complete. You will then be navigated to the project's home page.
10. Create a development environment for your project.
    a. Click the Environments side tab, then click **+ Create Environment**.
    b. Enter a name and description for the environment and click **Create**.
    c. Click **+ Add Instance**.
    d. In the Add Service Instances dialog, under Instance Type, click **Oracle Cloud Applications**.
    e. Under Authentication Method, click **Identity Domain**.
    f. Select the desired instance for the environment and click **Add**.
11. Create a new workspace.
    a. Click the Workspaces side tab, then click **New** > **New Application Extension**.
    b. In the New Application Extension dialog, in the Extension Name field, enter the extension name.
    c. In the Extension Id field, enter `site_cxsales_Extension`.

d. In the Workspace Name field, accept the default value or enter a new workspace name.
e. In the Development Environment field, select the environment you just created.
f. In the Base Oracle Cloud Application field, select **CX Sales (from CX Sales)**.
g. In the Git Repository region, click **Create new repository**.
h. Enter a name for both the repository and working branch.
i. Click **Create**.

Create a Workspace in an Existing Project

To create a new workspace in an existing project:

1. In Visual Builder Studio, click the Workspaces tab on the left pane, then click **Clone from Git**.
2. In the Clone from Git dialog, enter these values:

Field	Value
Workspace name	Enter the name of your workspace.
Repository Name	Select the Git repository connected to your team's project. Ask your administrator for this value.
Branch	Select **main**.
"New branch from selected" check box	Select this check box.
New branch name	Enter a name for your branch. For example, **<user_id>_<feature_name>**.
Development Environment	Select your development environment.
Sandbox (optional)	Select a sandbox, or leave this blank and attach a sandbox to your workspace later.

3. Click **Create**.

Create the Build Pipeline for Your Production Environment

Create the build pipeline to deploy your changes to your production environment.

When you created your project, a build pipeline was automatically created to deploy your application extension to your test environment. Now you must create a second pipeline for your production environment.

For complete instructions, see *Configure VB Studio to Run CI/CD Pipelines* and *Set Up the Project to Deploy to Production* in the Oracle Cloud Administering Visual Builder Studio guide.

Related Topics

- Setting Up Visual Builder Studio for Oracle Cloud Application Extensions

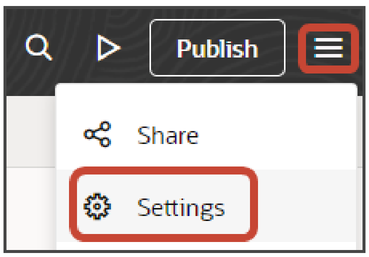
How can I change my project's Extension ID?

When working with a Sales application extension in Oracle Visual Builder Studio, your project's extension ID must be `site_cxsales_Extension`. This topic illustrates how to correct the extension ID, if required.

To change the extension ID for a project:

1. In Visual Builder Studio, from the left navigator, click **Environments** > **Deployments**.
2. Undeploy any deployments to target servers.

   See *Delete an Extension*.
3. Navigate to your workspace and, from the upper menu, click **Settings**.

4. In the **Extension ID** field, enter `site_cxsales_Extension`.
5. Build and deploy your extension once more.

   See *Build Your Applications* and *Package, Deploy, and Manage Extensions*.

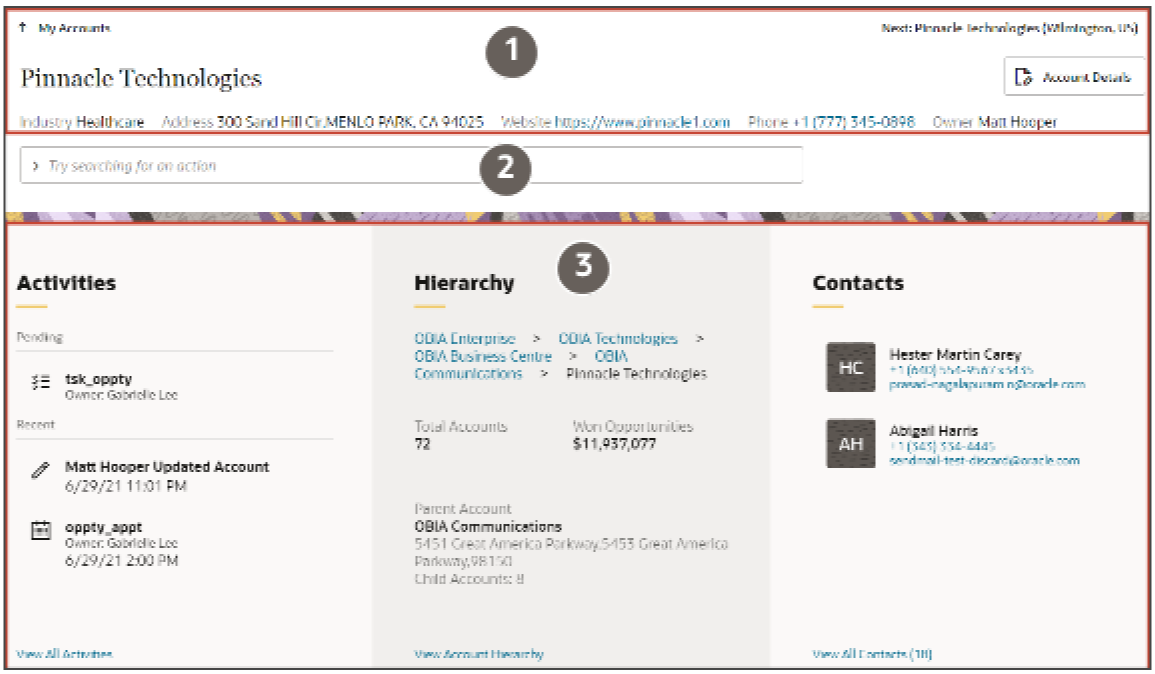
2 Configure Standard Object Pages

What You Can Configure

An individual detail page is divided into three sections: the header, the Action Bar, and a panels region (also known as the foldout). You can use Oracle Visual Builder Studio to configure a standard object's header and panels region so that the page shows the exact information that your users need. This chapter explains how to configure a standard object's detail page leveraging fragments. Fragments are building blocks that help you configure pages more quickly, with a minimal amount of manual coding.

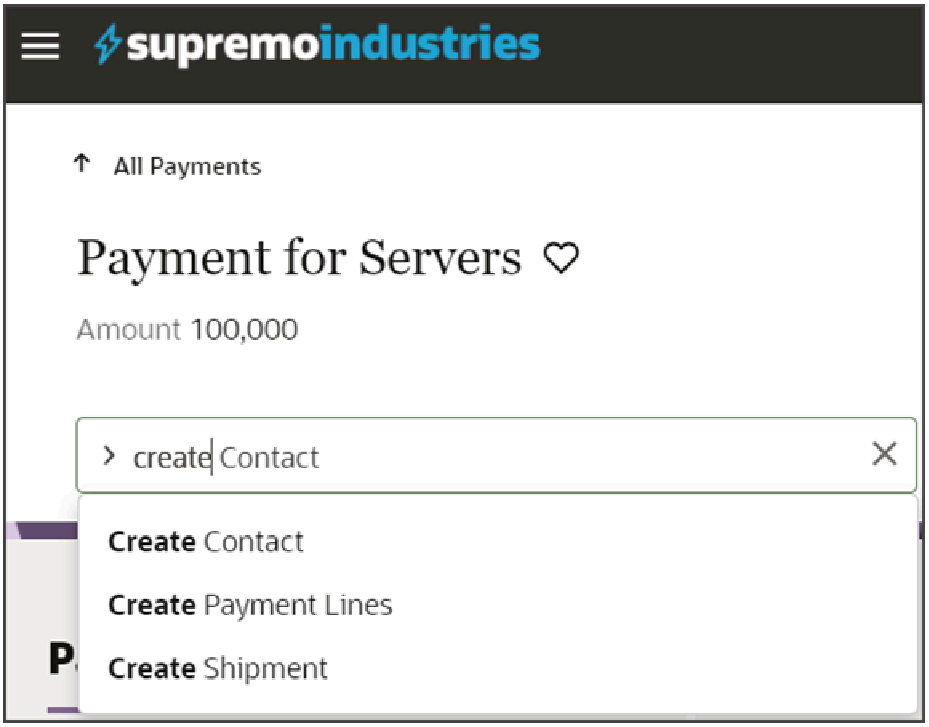
Note: In the runtime UI the detail page is called the overview page.

Here's a screenshot of the three main regions on a detail page. You can configure both the header and the panels region (callouts 1 and 3 in the screenshot). The topics in this chapter illustrate the various configuration options available.



Smart Actions in the Action Bar

The Action Bar is a field at the top of many pages where users can type keywords to access and update information, and take actions. For example, here's a screenshot of the Action Bar with suggested actions, after typing `create`:



Standard objects are delivered with a set of seeded smart actions, but you can create your own using Application Composer. See *Create Smart Actions*.

Note: If you previously created custom smart actions for a non-fragments implementation, then you don't need to create new smart actions for use with fragments. Instead, update existing UI-based custom smart actions to specify the action type, either **Add** or **Create**, as well as the target object and any required field mapping. For existing REST-based or object function-based smart actions that you already created, edit the action and then save without making any changes. These steps ensure that your custom smart actions still work with new fragment-based extensions.

Prerequisites

Before configuring a standard object's detail page, make sure you have a project and workspace ready in Visual Builder Studio.

For instructions about how to create a project and workspace, refer to the Before You Begin chapter.

Configure the Header on the Detail Page

The header on a record's detail page displays key details right at the top, so that users can quickly find what they need. The header displays a predefined set of attributes, but you can configure the region to display any attributes that your users are most interested in.

Where's the Header?

When you drill down to a record's detail page from a list page, the header displays at the top of the detail page.

Configure the Header Region Using a Rule Set

You can modify the header using Oracle Visual Builder Studio. To modify any region on a page, you create a new rule that returns a custom layout.

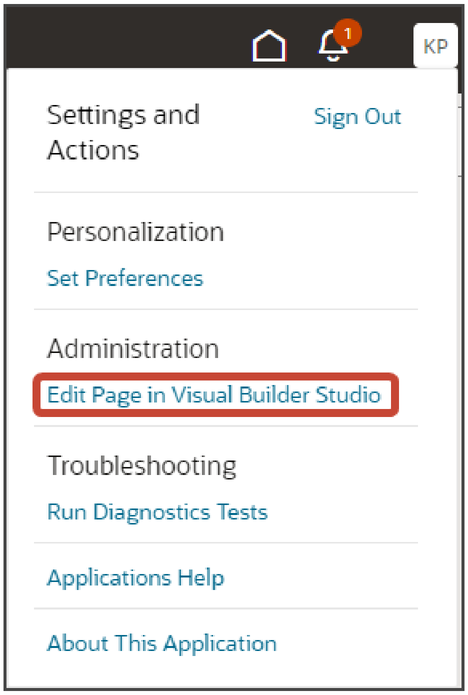
In this example, let's look at adding the Type field to the header on the account detail page.

1. Navigate to the page that displays the area you want to extend.

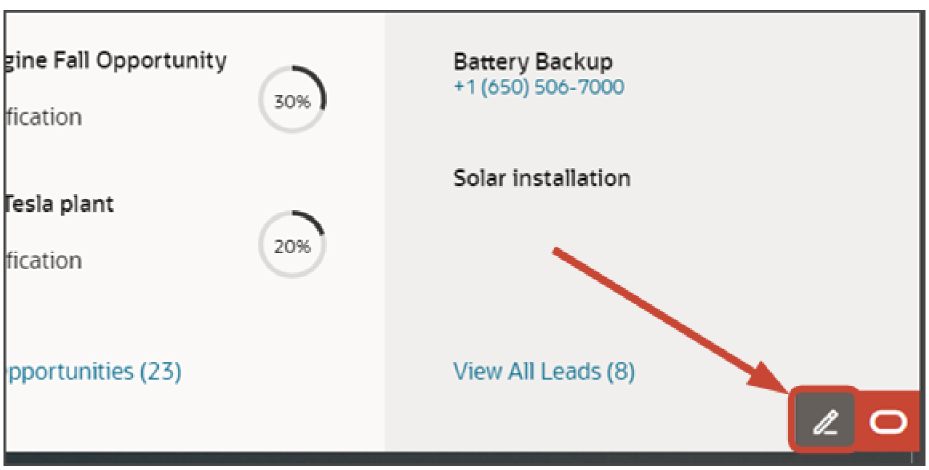
   In this example, navigate to the My Team's Accounts page and then click an account record.

2. To access Visual Builder Studio, you have one of two options:

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



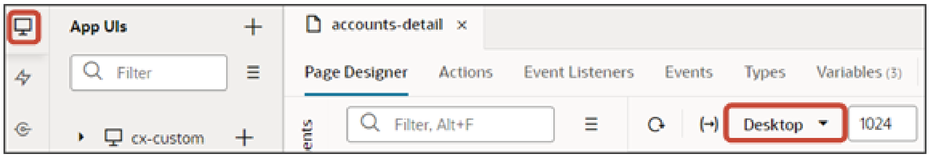
    - If you don't see the Settings and Actions menu, then in the lower right corner of the page, hover over the Ask Oracle icon and click **Edit Page in Visual Builder Studio**.



3. If you're working on multiple projects, then select the Extend Redwood CX Sales project to keep your fragments extension separate from non-fragments changes.
4. If you're working in an active sandbox when you launch Visual Builder Studio, then Visual Builder Studio looks for a workspace that's associated with your sandbox. If you're not working in a sandbox, 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, the accounts-detail page is displayed in the Page Designer canvas.

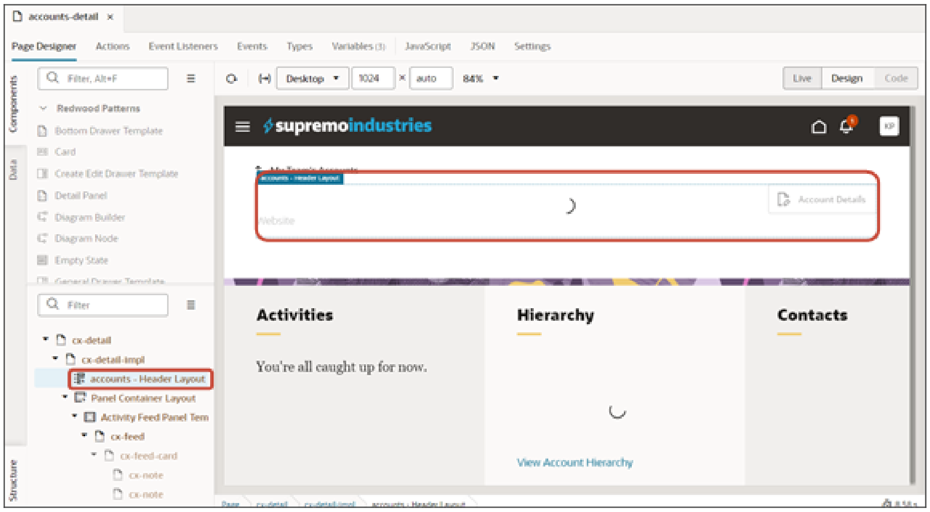
> **Tip:** To display more of the accounts-detail page in the canvas, collapse the left pane by clicking the App UIs tab on the left. Then, select **Desktop** or **Wide screen** from the list at the top of the Page Designer.



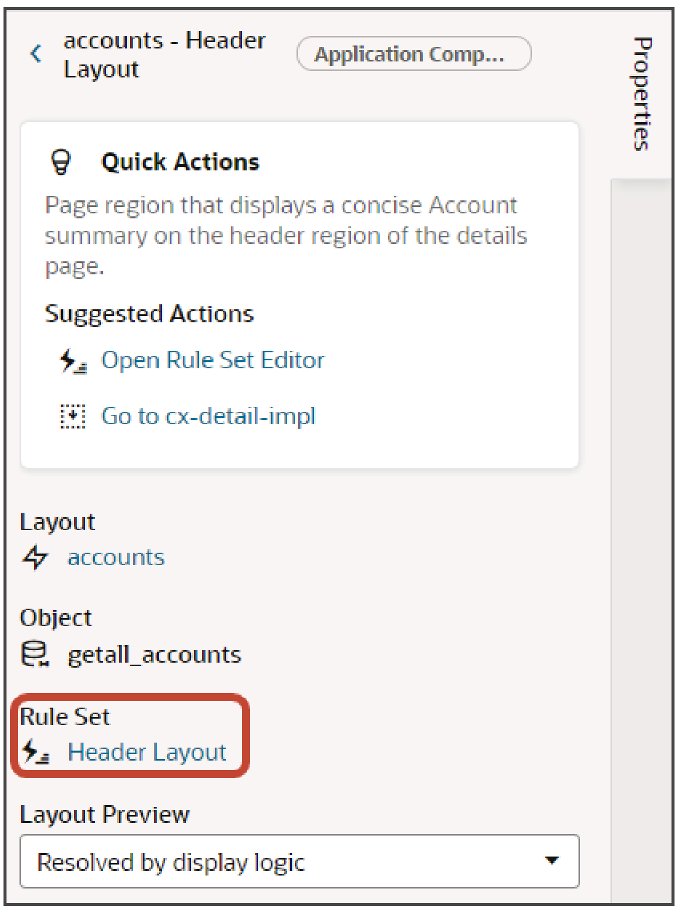
6. Click the Design button.

7. On the Structure panel, click the **accounts - Header Layout** node.

   The header is highlighted in the canvas.

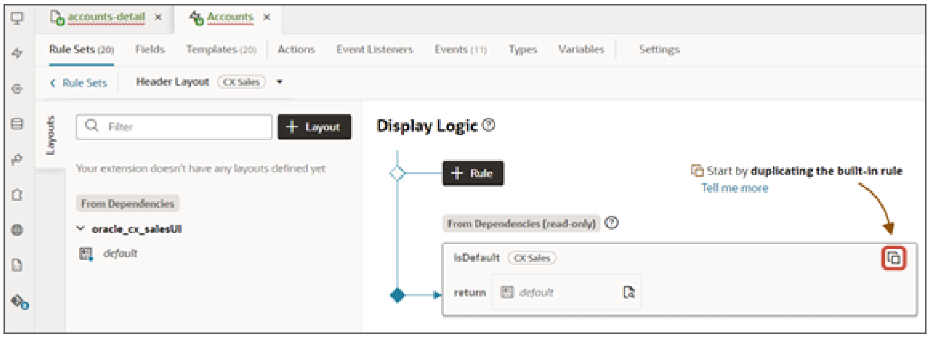


   On the Properties pane, the rule set is also displayed.



8. On the Properties pane, click the **Header Layout** rule set.

   The Accounts layout tab > Rule Sets subtab displays. The Rule Sets subtab includes both a default layout as well as a default rule. Let's create a new rule.

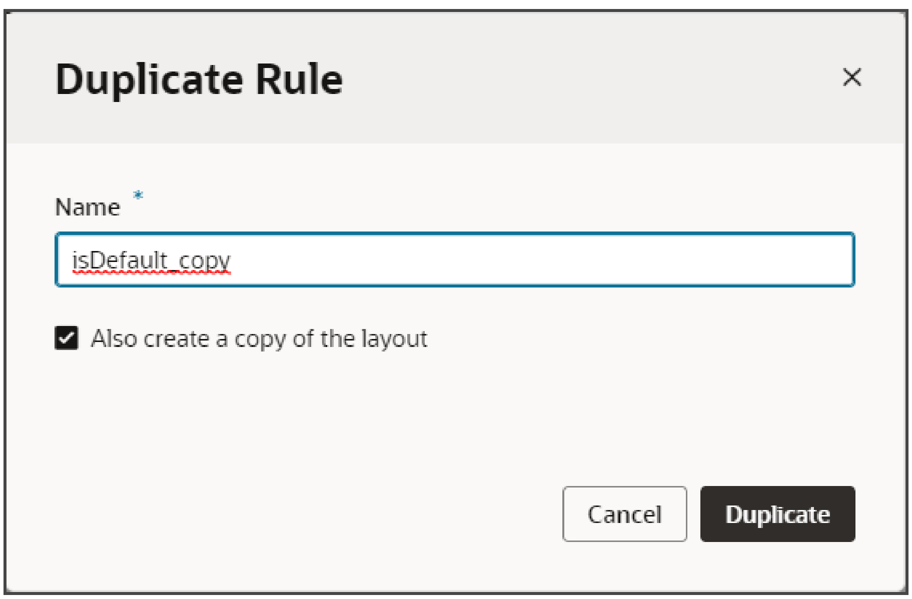
   

9. Click the Duplicate Rule icon.

10. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

    Also, make sure that the **Also create a copy of the layout** check box is selected.

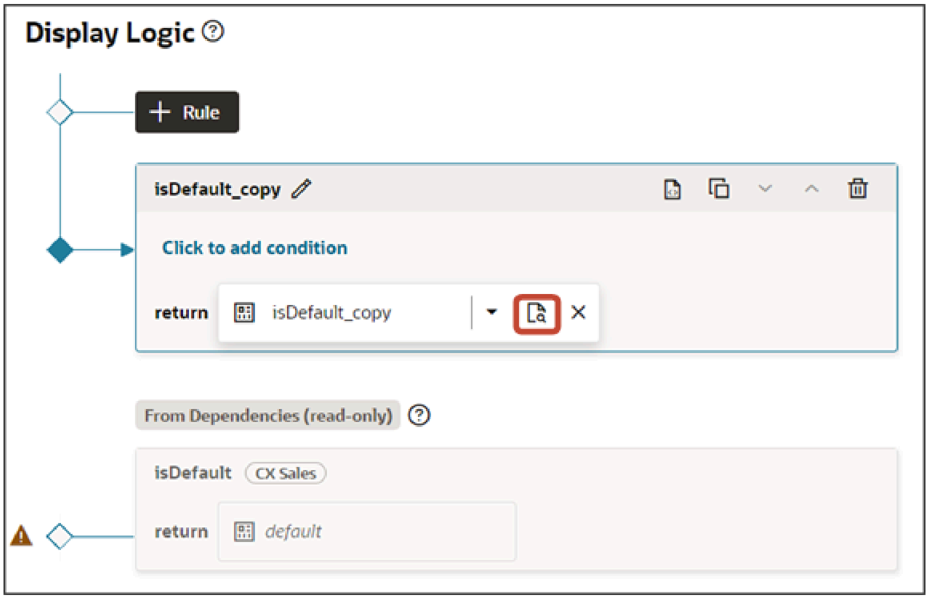
11. Click **Duplicate**.

The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

In this example, we're not adding any conditions which means that the associated layout will always be displayed.

12. Modify the rule's copied layout.

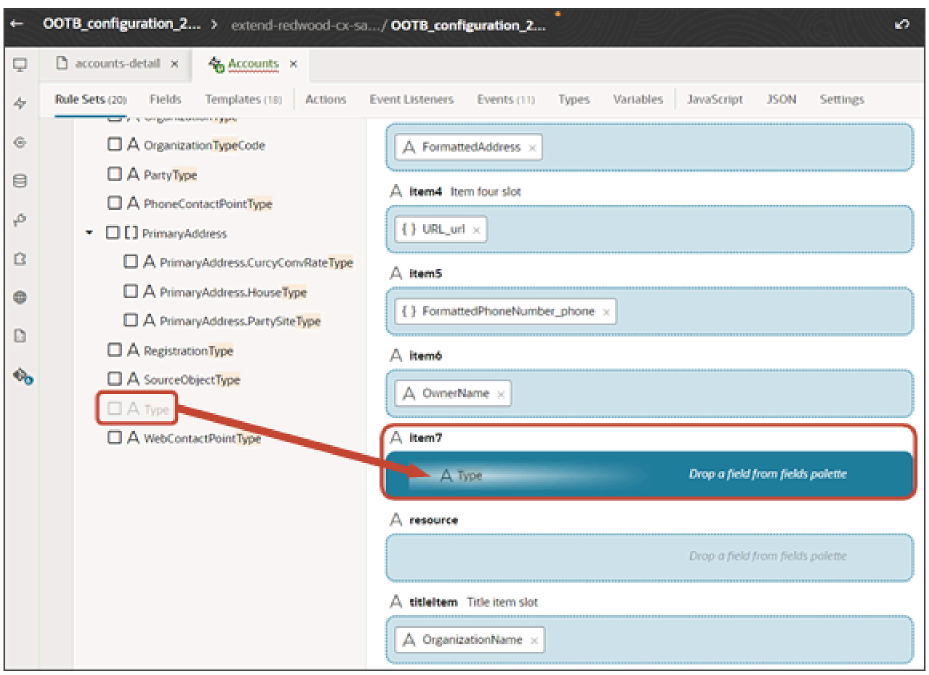
    a. Click the Open icon to edit the copied layout.



    b. In the Filter field, enter `type` to locate the Type field.

    c. Drag the Type field to an available section in the header.

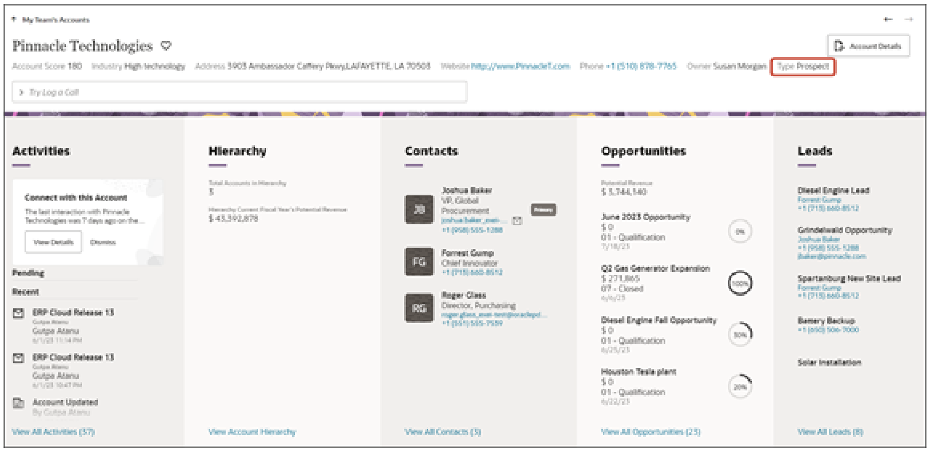
    In our example, drag the Type field to the item7 section.



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

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```



> **Note:** The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

Show or Hide a Field Based on Another Field's Value

In Oracle Visual Builder Studio, you can use rules and display logic to determine which page layout to display at runtime. You can also set up your rules so that a field only displays depending on the value of another field. For example, maybe you want the Owner field to always display for an account record, unless the account type is Prospect. This topic illustrates how to implement this scenario using display logic.

This topic also provides an alternative method for simple hide/show scenarios.

Prerequisite

In this example, we'll show or hide a field based on an account's type. Before you try the following procedure, you must add the Type field to the account detail page's header using the instructions documented in *Configure the Header on the Detail Page*.

Add Display Logic to a Rule

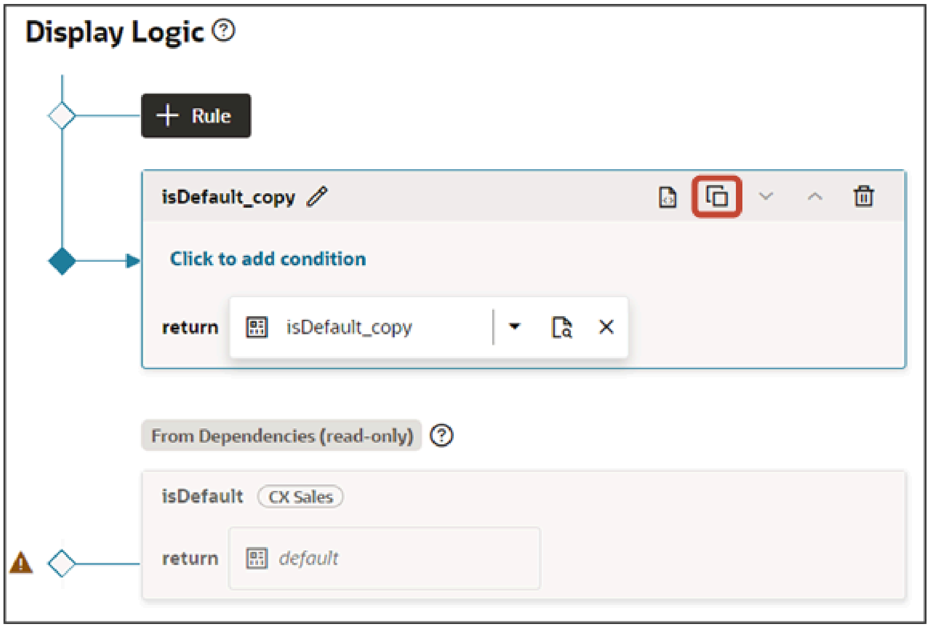
To show a field depending on the value of another field, we'll create two rules and add a condition to the first rule, as follows:

1. If an account's type is Prospect, then display a layout without the Owner field.
2. If an account's Type is not Prospect, then display a different layout that includes the Owner field.

Let's create the rule and corresponding layout for item #1.

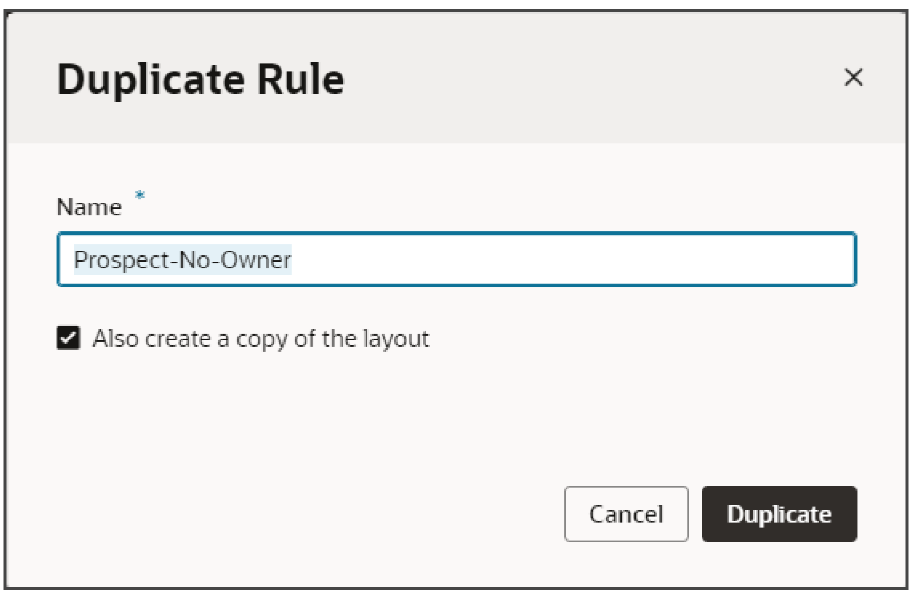
1. In Visual Builder Studio, on the Accounts layout tab > Rule Sets subtab, create a new rule by copying the rule that you created in the procedure documented in *Configure the Header on the Detail Page*.

   Click the Duplicate icon.



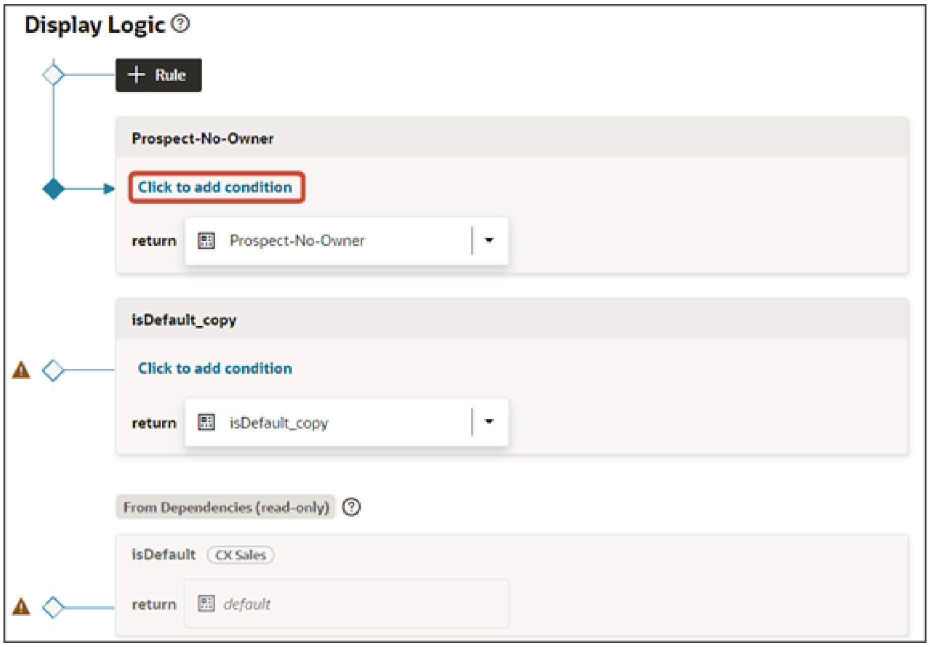
2. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you, such as `Prospect-No-Owner`.

   Also, make sure the **Also create a copy of the layout** check box is selected.



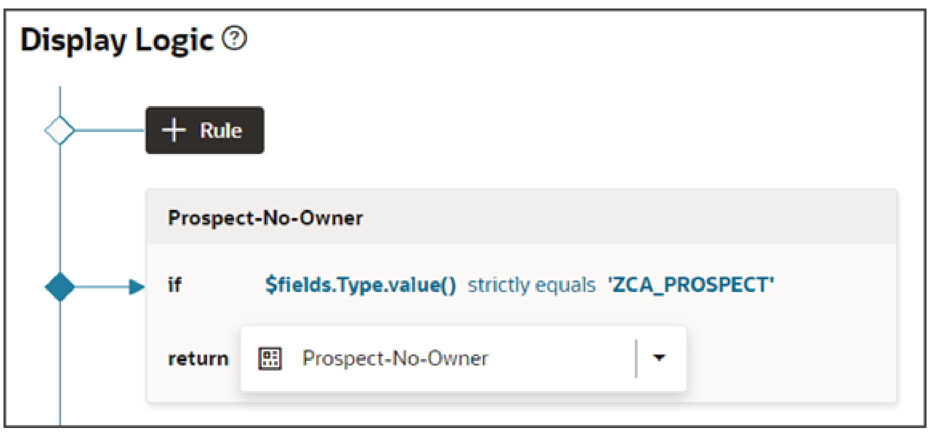
3. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user. Otherwise, the next rule will be evaluated.
4. Let's add a condition to the rule. Click **Click to add condition**.



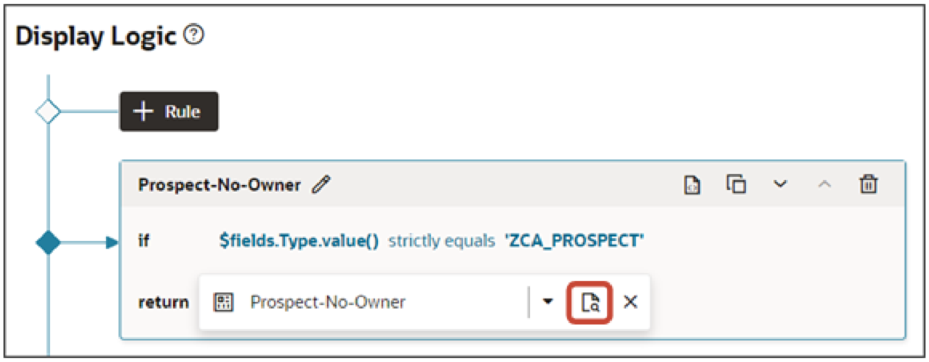
5. Add the condition to specify when you want the associated layout to display at runtime.

   In this example, we want the Type field to equal `Prospect`.

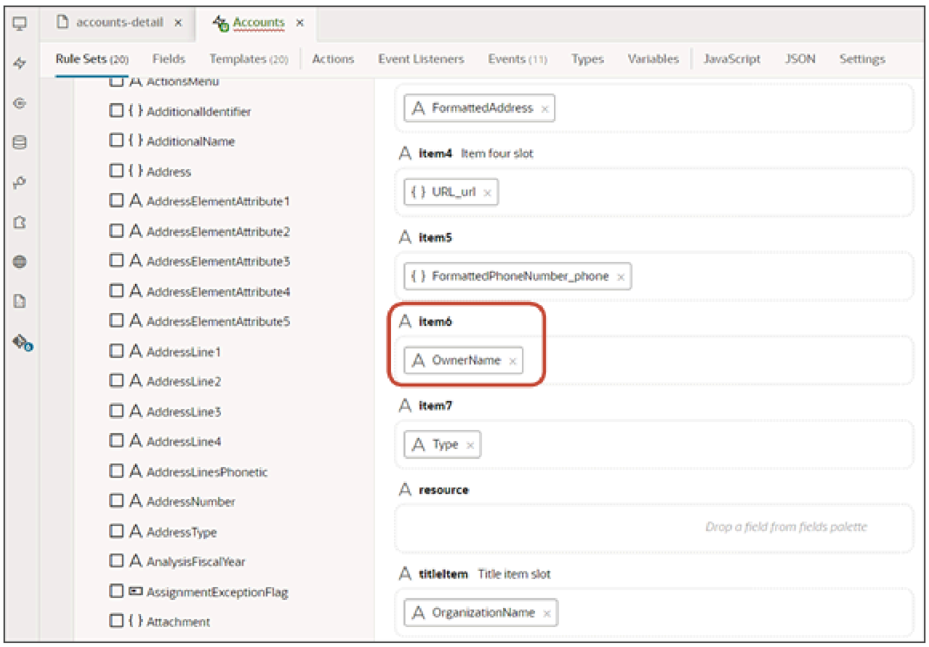


6. Edit the layout so that the Owner field is not included when the account type is Prospect.

    a. Click the Open icon to edit the copied layout.



    b. Delete the Owner field in the item6 slot.



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

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

In this example, we can see that this account is a prospect, and therefore there is no Owner field.

However, this account is a Customer, and thus we see the Owner field.

> **Note:** The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

> **Note:** The rule and corresponding layout that you previously created in *Configure the Header on the Detail Page* satisfy the conditions for item #2, "If an account's Type is not Prospect, then display a different layout that includes the Owner field."

Alternative Method for Simple Hide/Show Use Cases

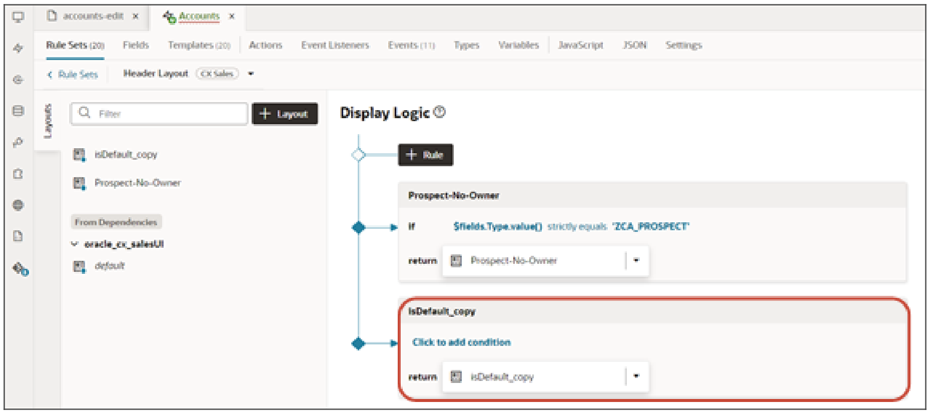
The previous example illustrates how to use rules, display logic, and custom layouts to show or hide a field based on another field's value. But, you can also directly modify the page template instead. This method is an alternative method that might be preferable for simple hide/show use cases.

Let's look at an example. In this next procedure, we'll hide the account's phone number if the account's industry is High Tech.

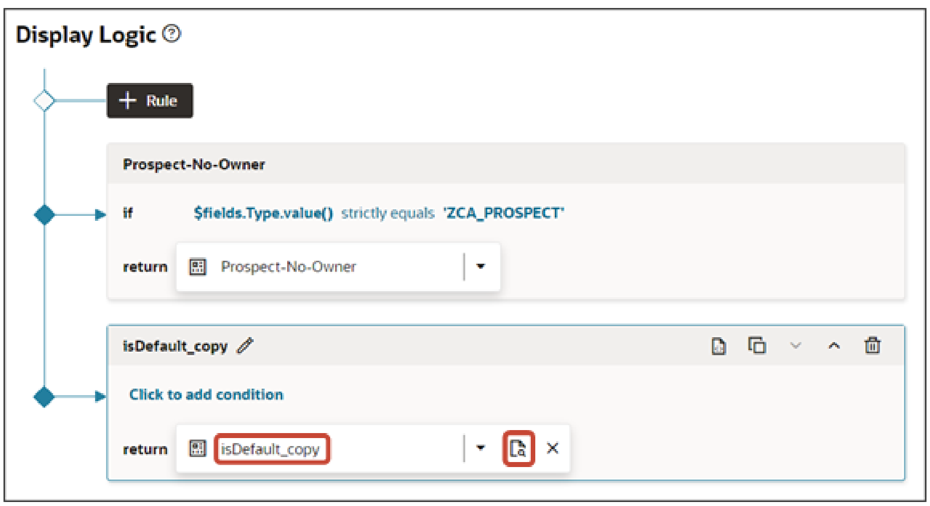
1. In Visual Builder Studio, on the Accounts layout tab > Rule Sets subtab, review the list of rules.

   We can ignore the Prospect-No-Owner rule because that rule is specifically for the instance when the Type field is Prospect.

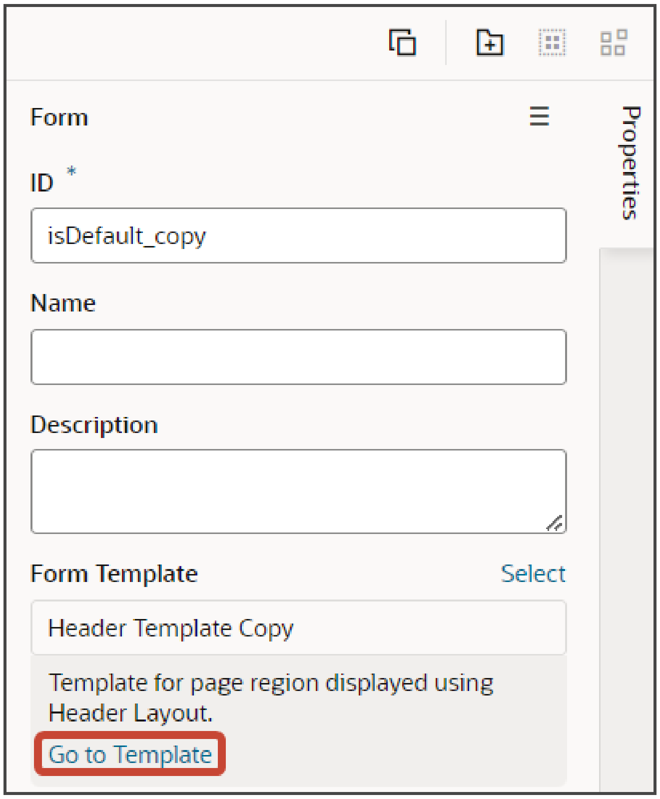
   Instead, we'll edit the rule that you created in the procedure documented in *Configure the Header on the Detail Page*.

2. Click the Open icon to edit the rule's copied layout.

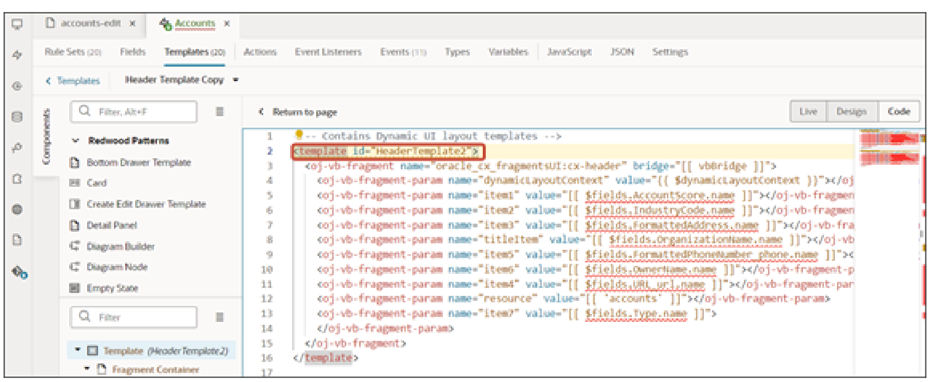
   

3. On the Properties pane, click **Go to Template**.

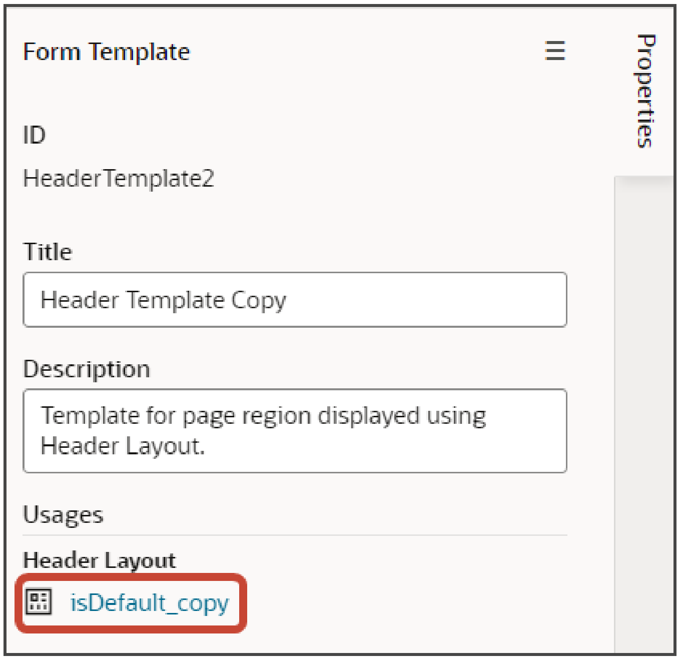


4. Click the Code button.

5. In the template editor, highlight the template tags for the header template.



6. On the Properties pane, confirm the layout name.



7. In the template editor, highlight the **FormattedPhoneNumber_phone** parameter.

```
<oj-vb-fragment-param name="item5" value="[[ $fields.FormattedPhoneNumber_phone.name ]]"></oj-vb-fragment-param>
```

8. Replace the highlighted parameter with this line:

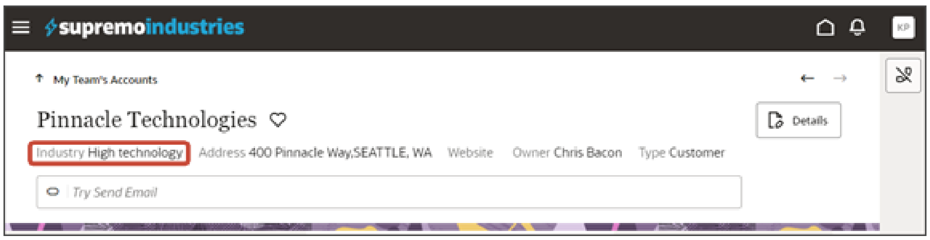
```
<oj-vb-fragment-param name="item5" value="[[ ( $fields.IndustryCode.value() =='HIGH_TECHNOLOGY') ? null : $fields.FormattedPhoneNumber_phone.name]]"></oj-vb-fragment-param>
```

9. From the accounts-detail page, click the Preview button to see your changes in your runtime test environment.

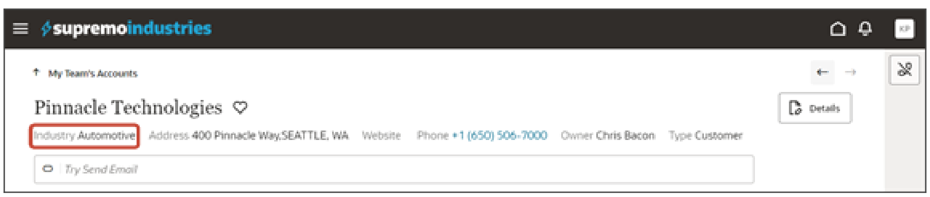
The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```

Let's look at an account. In this example, the account's industry is **High technology**, so the phone number doesn't display in the header.



If we change the industry to **Automotive**, notice that the phone number now displays.

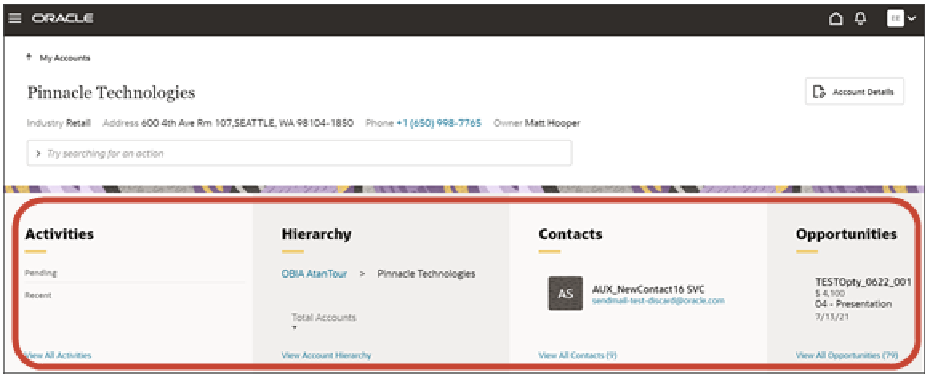


Note: The header displays a field only if the field has a value. For example, if an account doesn't have a Type specified, then the Type field won't display.

Configure Panels on the Detail Page

An individual record's detail page includes key information displayed in a region of panels. Panels display information related to the record, such as the most recent and pending activities or related contacts. As an administrator, you can use Oracle Visual Builder Studio to configure these panels to meet your users' requirements.

Here's an example of the panels that display for an account:



A different set of panels can display for different types of objects. For example, for an account, you'll probably see a hierarchy panel and a contacts panel. For an opportunity, meanwhile, you'll probably see a products panel and a team panel.

In Oracle Visual Builder Studio, these panels are rendered using a panel layout. Each Sales object has a default, read-only panel layout which you can duplicate and then configure. For example, you can:

- Add and remove panels.

  See *Add a Panel*.

  - If you add a panel, then you must add a subview, as well.

    See *Add a Subview*.

- Change the display order of the panels.

  See *Change the Display Order of a Panel*.

- Display a panel based on a condition.

  See *Display a Panel Based on a User's Role* and *Display a Panel and Subview Based on a Field Value*.

You can also configure the content inside most panels. See *Configure the Contents of a Panel*. Note, however, that you can't configure the Activities panel.

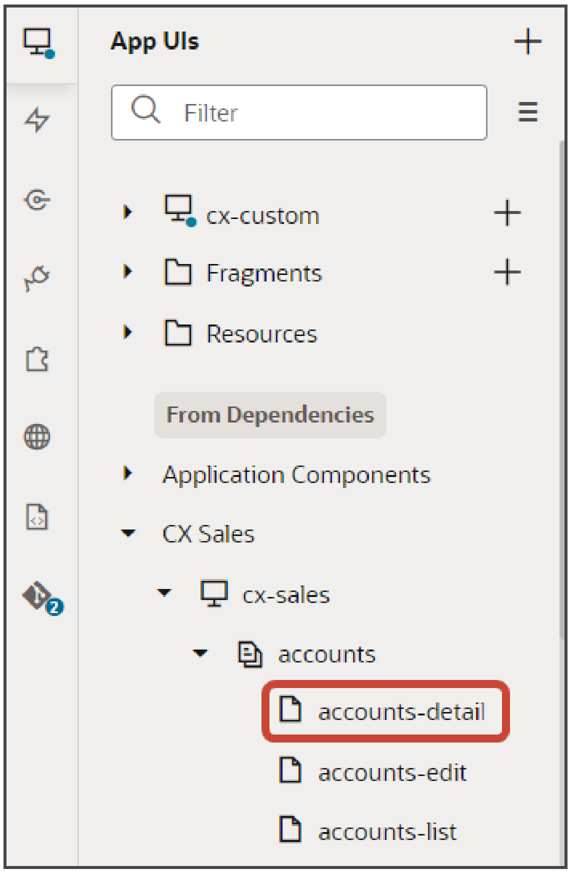
Add a Panel

Each Sales object comes with a set of panels. You can add a panel as needed.

To add a panel, you must first duplicate the object's read-only panel layout. Here's an example of duplicating the read-only panel layout for the Account detail page:

1. In Visual Builder Studio, navigate to the Page Designer tab on the accounts-detail page.

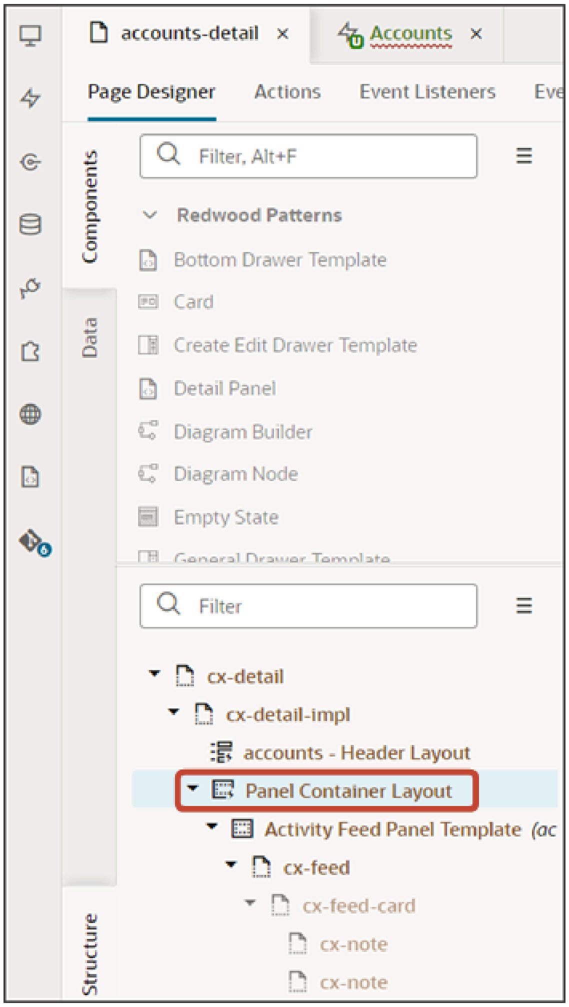
   You can navigate to the accounts-detail page either by navigating to Visual Builder Studio from the Account detail page in the application, or by navigating to the accounts-detail node on the App UIs tab in Visual Builder Studio.

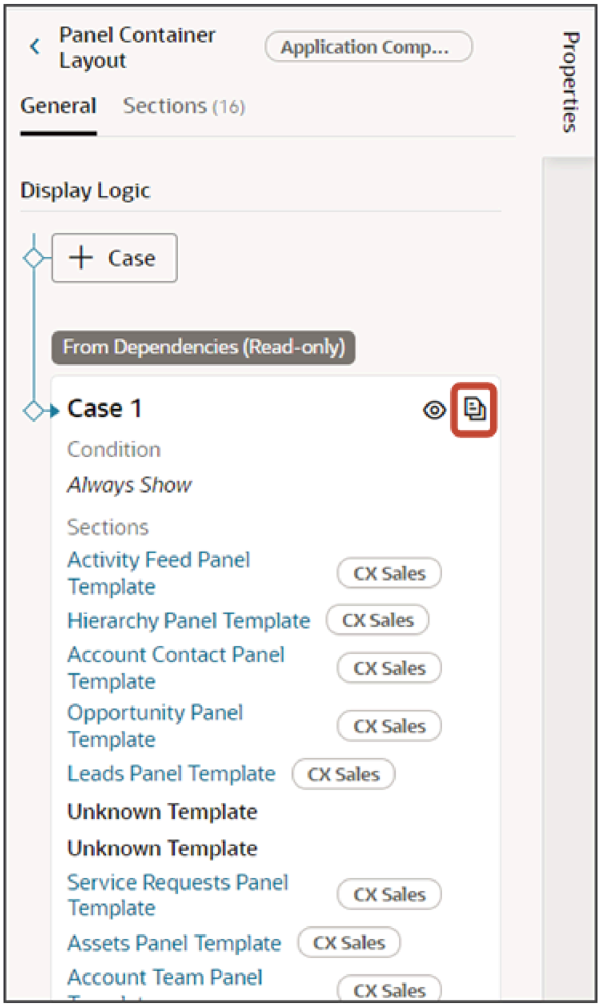
2. Click the Design button.

3. On the Structure panel, click the **Panel Container Layout** node.



4. On the Properties pane, next to **Case 1**, click the Duplicate icon.



After duplicating the panel container layout, you can now add a panel. There are different procedures to follow depending on the panel object (is it a standard or custom object?) as well as on the type of relationship between the two objects.

1. If the relationship between the detail page's object and the panel object is one-to-many, and the panel's target related object is a standard object, then see *Add a Standard Object Panel for Related Objects (One-to-Many)*.
2. If the relationship between the detail page's object and the panel object is one-to-many, and the panel's target related object is a custom object, then see *Add a Custom Object Panel for Related Objects (One-to-Many)*.
3. If the relationship between the detail page's object and the panel object is many-to-many, then see *Add a Panel for Related Objects (Many-to-Many)*.

   These instructions apply whether the panel object is a standard or custom object.

Note: After adding a panel, you must also add a related subview for the panel. See the next section, Add a Subview.

Add a Subview

If you add a panel to an object's detail page, then you must also add a related subview for the panel. The subview displays all records that can't fit into the limited size of the panel.

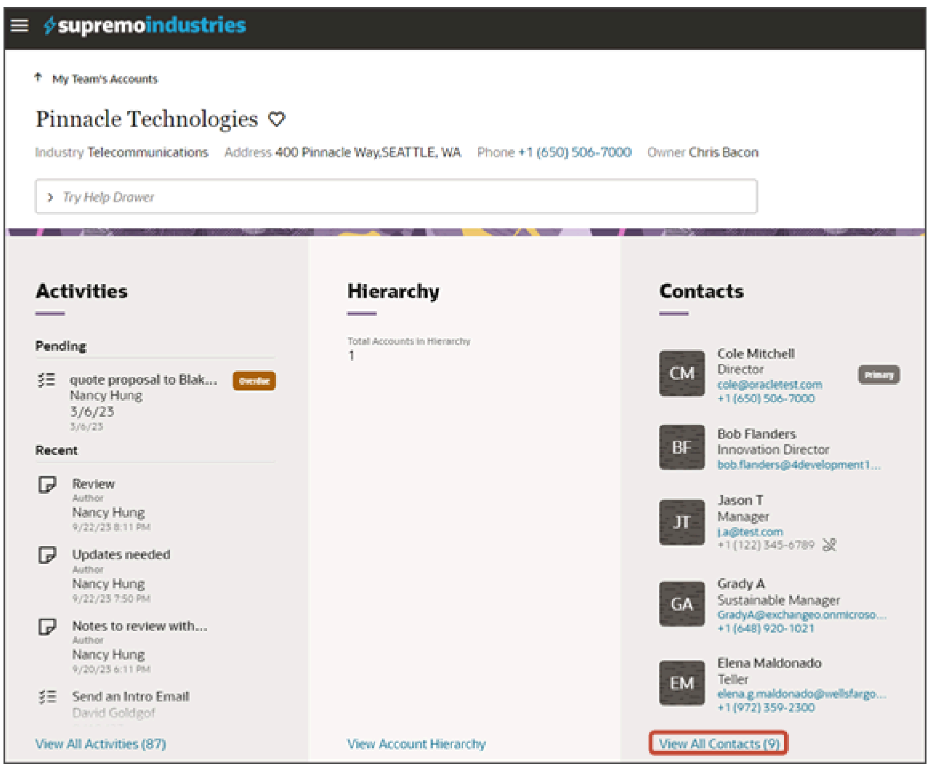
Before you can add a subview, you must first duplicate the object's read-only subview layout. Here's an example of duplicating the read-only subview layout for the Account detail page.

1. In your application, navigate to any record for the object that you just added a panel to.

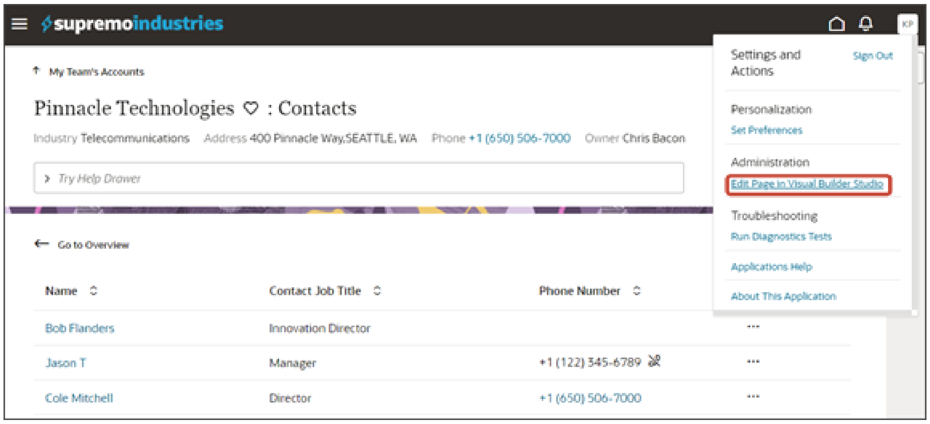
   For example, navigate to any account detail page.

   > **Note:** You must start from an application page first, before navigating to Visual Builder Studio. This ensures that the **Subview Container Layout** node is available, which you'll use in step 6.

2. On the account detail page, click the View All link at the bottom of any panel to navigate to the panel's subview.

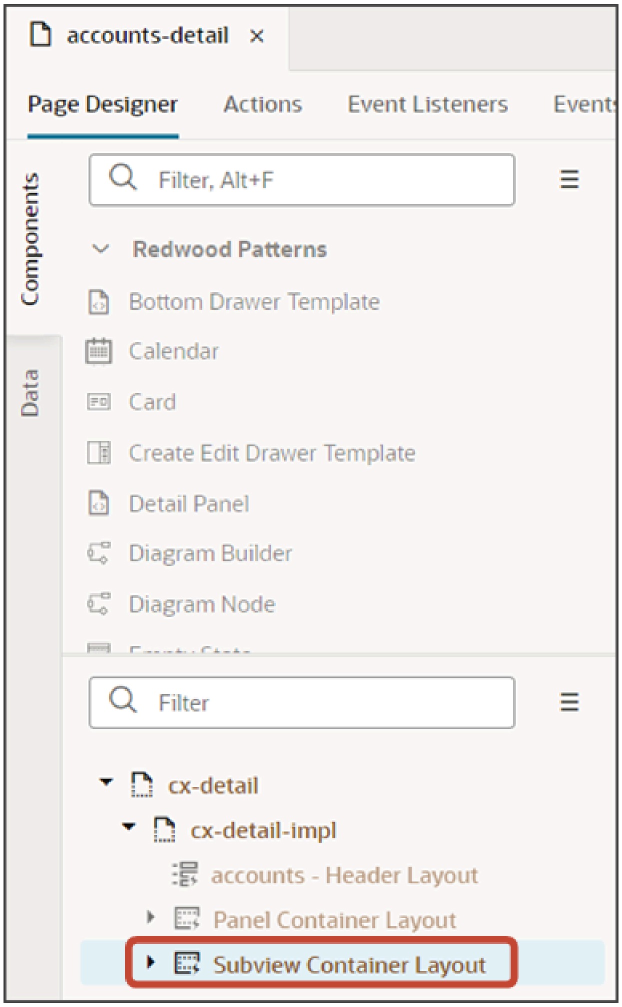


3. On the subview, under the Settings and Actions menu, select **Edit Page in Visual Builder Studio**.

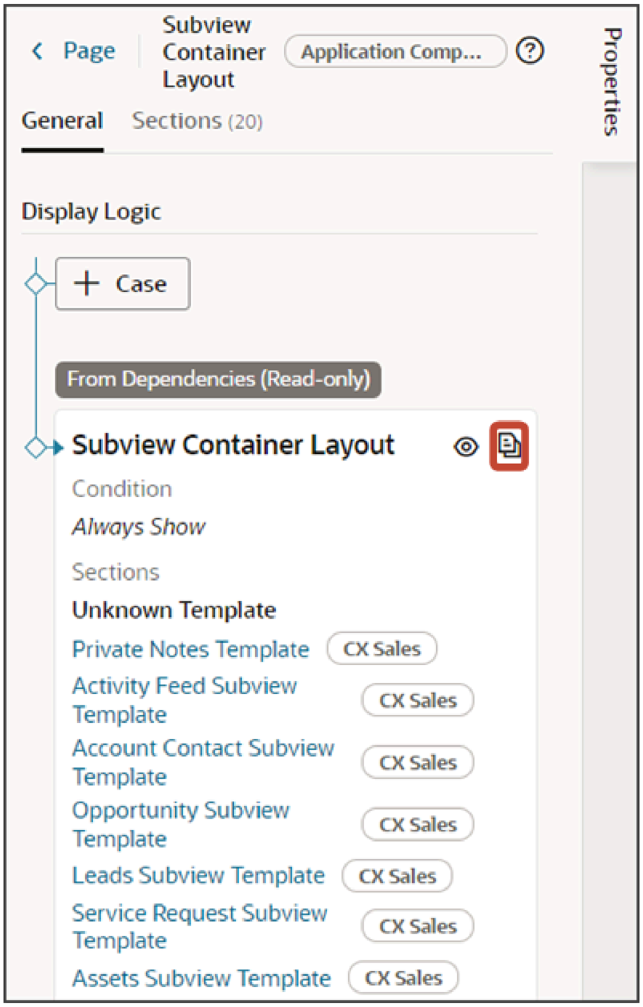


4. When you enter into your workspace in Visual Builder Studio, the accounts-detail page is displayed in the Page Designer canvas.
5. Click the Design button.

6. On the Structure panel, click the **Subview Container Layout** node.



7. On the Properties pane, next to **Subview Container Layout**, click the Duplicate icon.



After duplicating the read-only subview container layout, you can now add a subview. There are different procedures to follow depending on the subview object (is it a standard or custom object?) as well as on the type of relationship between the two objects.

1. If the relationship between the detail page's object and the subview object is one-to-many, and the subview's target related object is a standard object, then see the example documented in "Add a Subview for the Leads Panel" inside the *Add a Standard Object Panel for Related Objects (One-to-Many)* topic.
2. If the relationship between the detail page's object and the subview object is one-to-many, and the subview's target related object is a custom object, then see "Configure the Subview Layout" inside the *Configure the Subview for Related Objects (One-to-Many)* topic.
3. If the relationship between the detail page's object and the subview object is many-to-many, then see "Configure the Subview Layout" inside the *Configure the Subview for Related Objects (Many-to-Many)* topic.

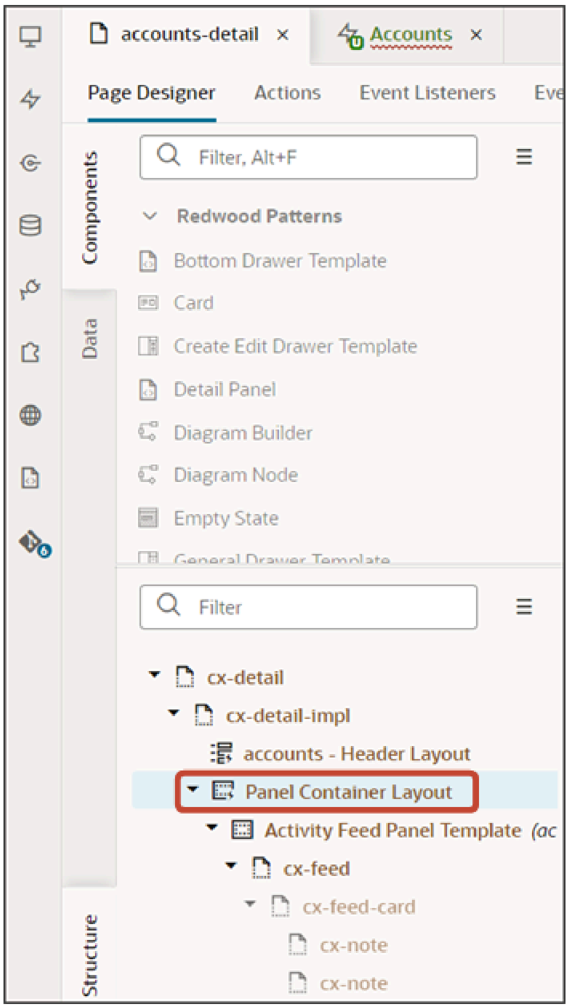
   These instructions apply whether the subview object is a standard or custom object.

Change the Display Order of a Panel

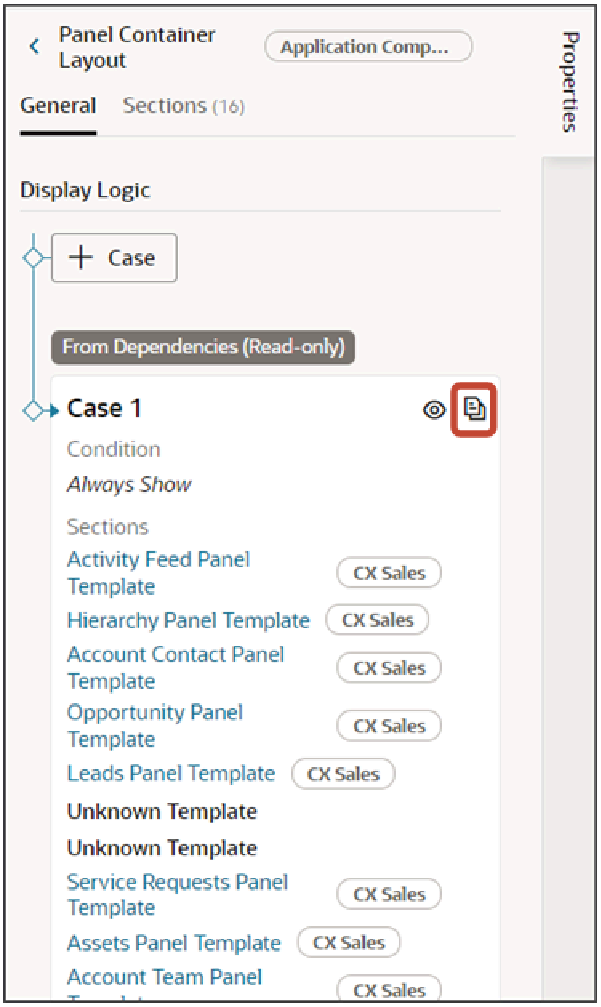
In this example, we'll move the Activities panel for an account so that it displays after the Hierarchy panel.

1. In Visual Builder Studio, navigate to the Page Designer tab on the accounts-detail page.
2. Click the Design button.

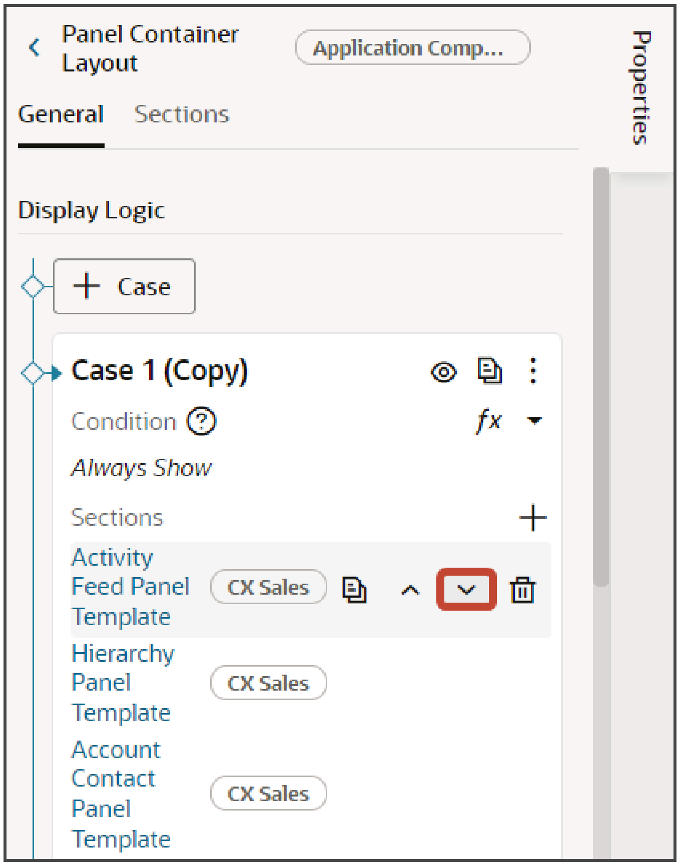
3. On the Structure panel, click the **Panel Container Layout** node.



4. On the Properties pane, next to **Case 1**, click the Duplicate icon.

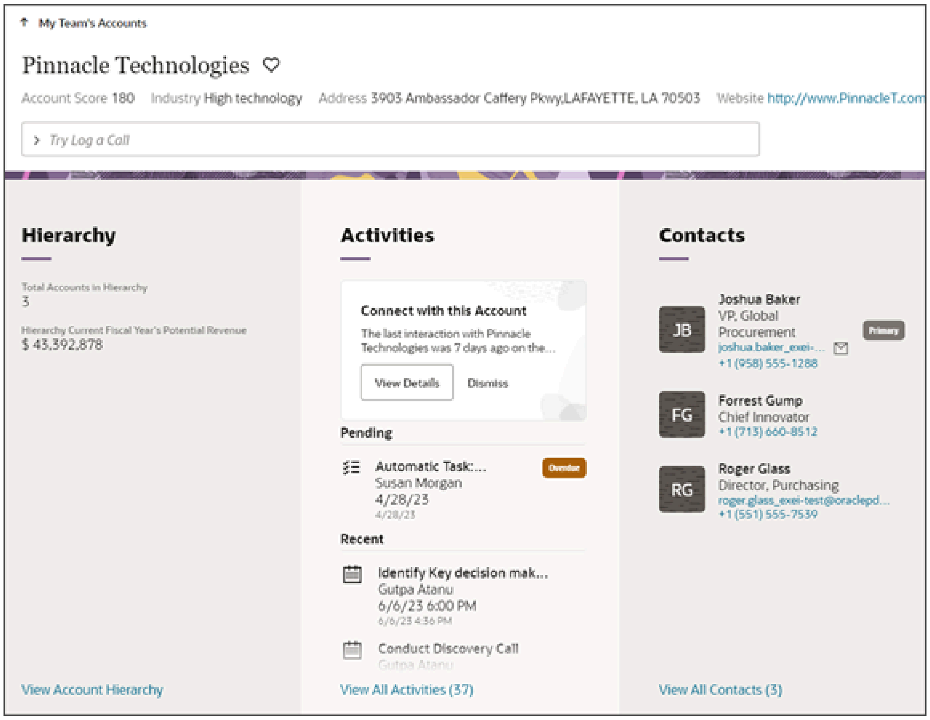


5. On the Properties pane for the copied layout, next to the Activity Feed Panel Template section, click the Move Down icon so that this section displays after the Hierarchy Panel Template section.



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

Here's a screenshot of the Activities panel which now displays right after the Hierarchy panel.



Related Topics

- Display a Panel Based on a User's Role
- Display a Panel and Subview Based on a Field Value

Display a Panel Based on a User's Role

To display a panel based on a user's role, you have two options: one option for a simple hide/show of a panel and a second option for more complicated scenarios such as reordering panels.

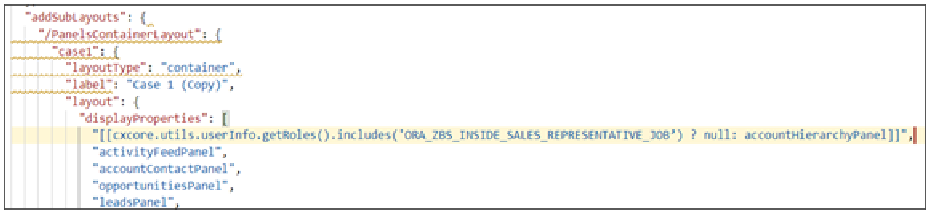
First, let's configure the panel region so that the Hierarchy panel displays only to sales managers, not sales representatives. This example is a simple hide/show use case, so it's easiest to update the accounts-detail page's JSON with the condition.

In the accounts-detail page's JSON, indicate that the Hierarchy panel must not display for sales representatives.

1. In Visual Builder Studio, click the App UIs side tab.
2. Navigate **CX Sales** > **cx-sales** > **accounts** > **accounts-detail**.
3. On the accounts-detail page, click the JSON subtab.
4. Scroll to the "/PanelsContainerLayout" section.

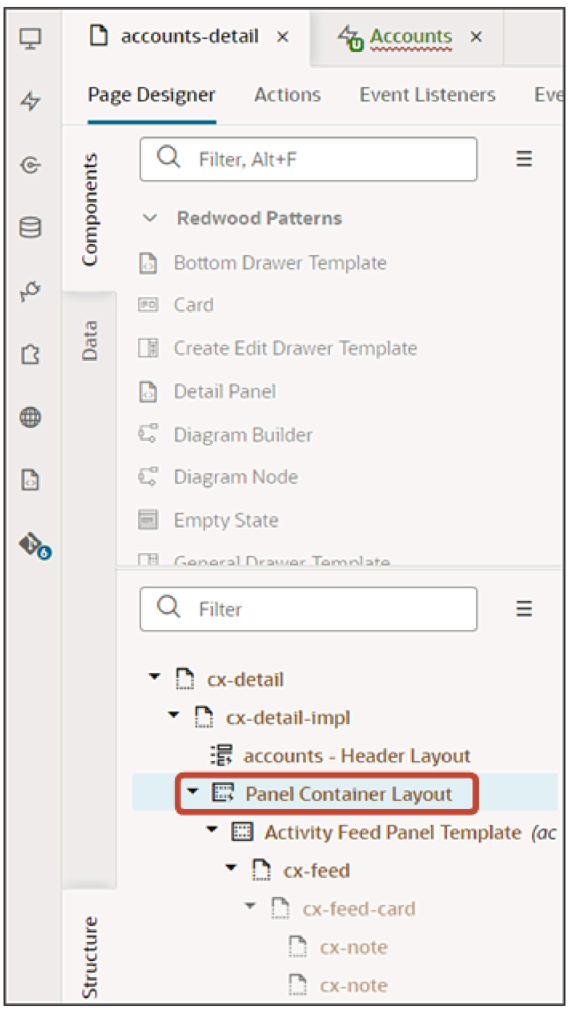
5. Replace the "accountHierarchyPanel" panel with the following expression:

```
"[[cxcore.utils.userInfo.getRoles().includes('ORA_ZBS_INSIDE_SALES_REPRESENTATIVE_JOB') ? null:
accountHierarchyPanel]]",
```

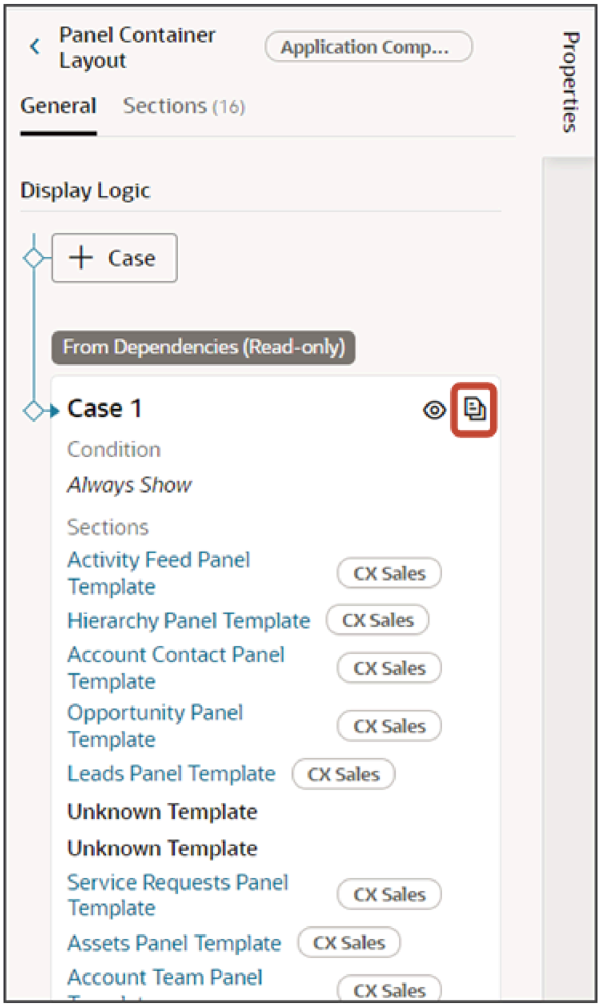


For more complex changes like reordering panels based on a user's role, you must create a panel layout with the new sequence and add a condition. For example, for the Case 1 (Copy) panel layout that we previously created, add the sales manager condition. This means that only sales managers can view the panel layout where the Activities panel displays after the Hierarchy panel.

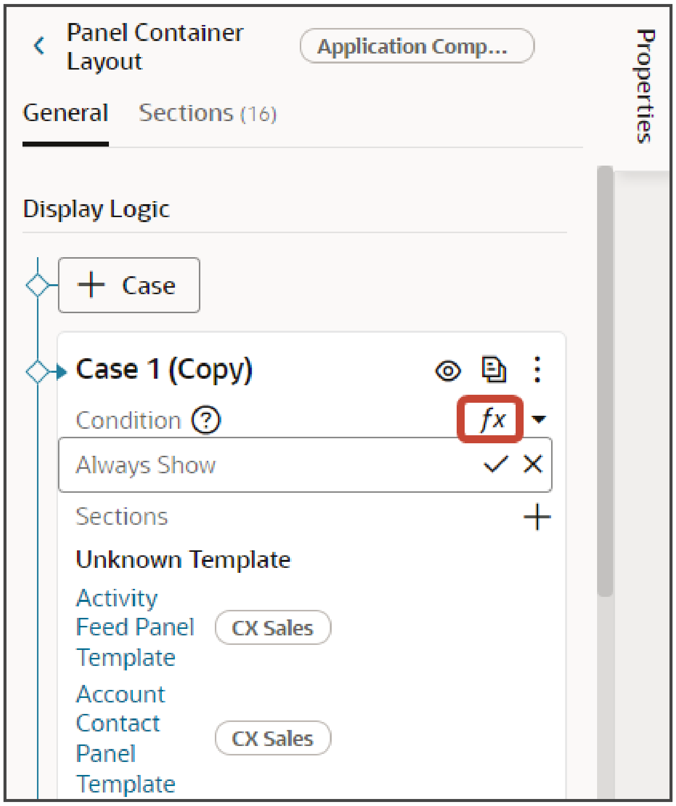
1. Navigate to the Page Designer tab on the accounts-detail page.
2. On the Structure panel, click the **Panel Container Layout** node.



3. On the Properties pane, next to **Case 1**, click the Duplicate icon.

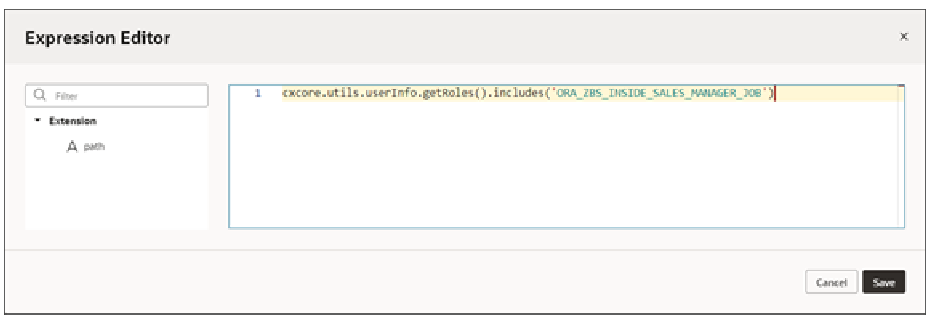


4. Next to the Case 1 (Copy) panel layout's condition, click the Expression Editor icon.



5. In the Expression Editor dialog, enter this expression:

```
cxcore.utils.userInfo.getRoles().includes('ORA_ZBS_INSIDE_SALES_MANAGER_JOB')
```



6. Click **Save**.

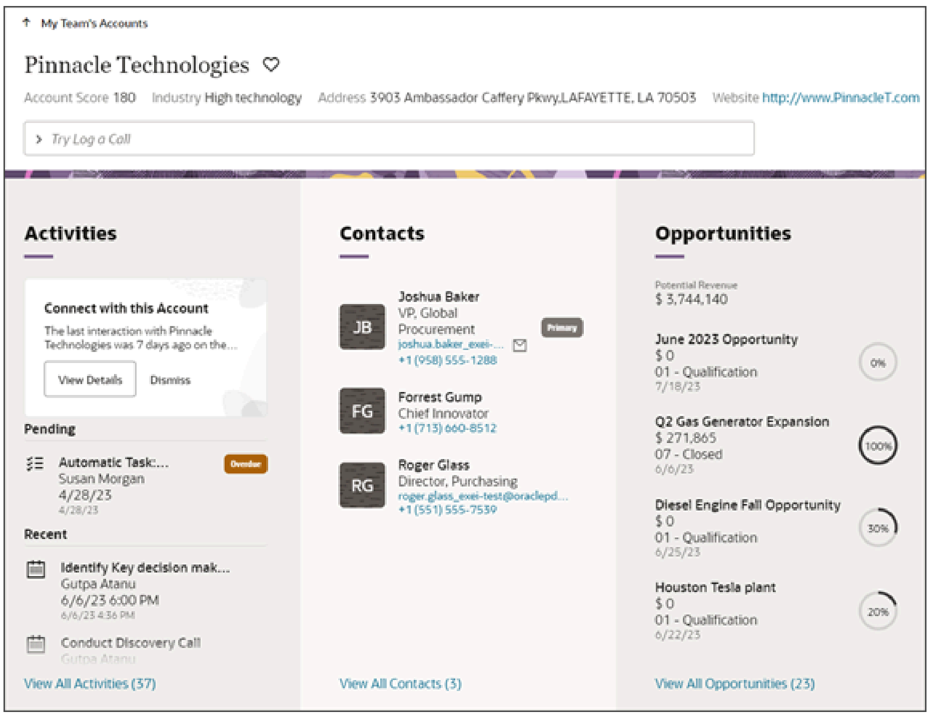
7. Let's test the sales representative change.

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

   The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```

   Here's a screenshot where the logged-in user is a sales representative. Notice that, on the account detail page, the Hierarchy panel doesn't display. This is due to the JSON update that you made previously.



   Next, try signing in as a sales manager. On an account detail page, the Activities panel should display after the Hierarchy panel.

Display a Panel and Subview Based on a Field Value

You can display different sets of panels (and their corresponding subviews) based on the value of a field.

To do this, create a panel layout or subview layout, and then add a field value condition. If a record's field matches the specified value, then the associated layout displays. If not, then a different layout displays.

This topic illustrates how an account's type, either Customer or Prospect, changes the panel and subview layout on an account detail page.

Prerequisite

To create a layout condition that references a field value, you must first enable this feature so that panels and subviews are loaded to the page only after evaluating the header.

1. In Visual Builder Studio, click the App UIs side tab.
2. Navigate **CX Sales** > **cx-sales** > **accounts** > **accounts-detail**.
3. On the accounts-detail page, click the Variables subtab.
4. In the Constants region, click the **deferRelatedDataLoad** constant.
5. On the Properties pane, in the Default Value field, select **True**.

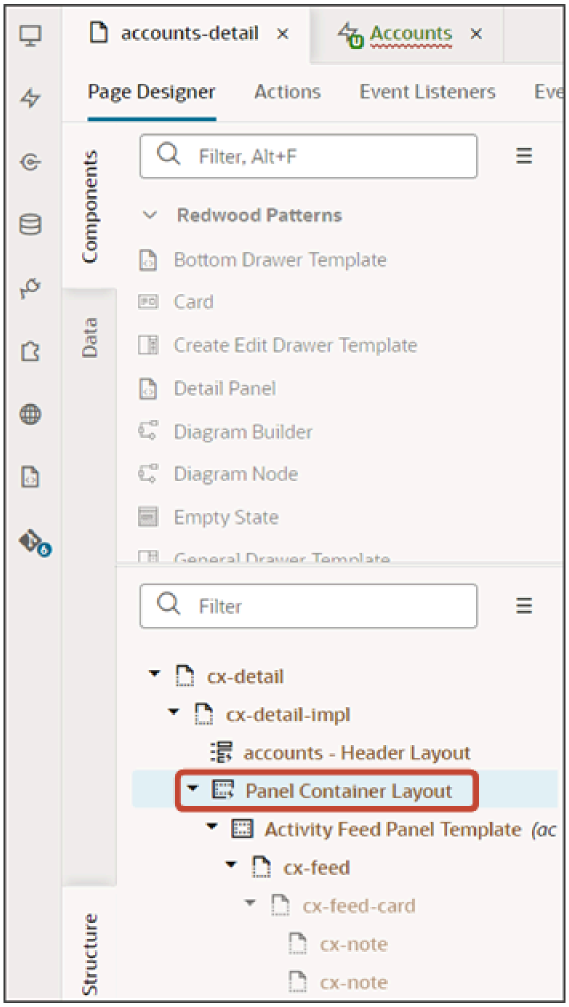
   If you want to add a field value condition to panel and subview container layouts, then you must set this value to true.

Create a New Panel Layout

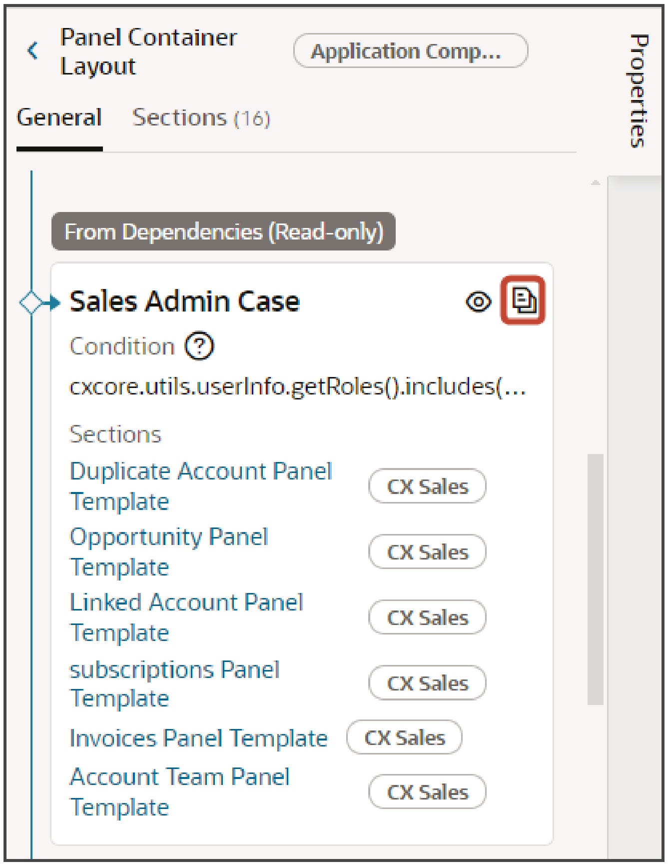
Once you have enabled the feature, you can now add a field value condition to a panel layout. Let's add a condition to the account detail page.

1. Navigate to Visual Builder Studio from an account record.
2. On the accounts-detail page, click the Page Designer subtab.

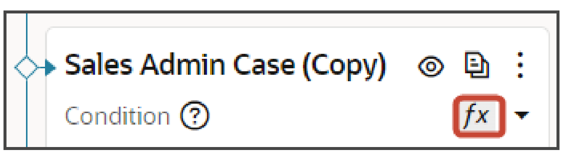
3. On the Structure panel, click the **Panel Container Layout** node.



4. On the Properties pane, next to **Sales Admin Case**, click the Duplicate icon.



5. Next to the Sales Admin Case (Copy) panel layout's condition, click the Expression Editor icon.



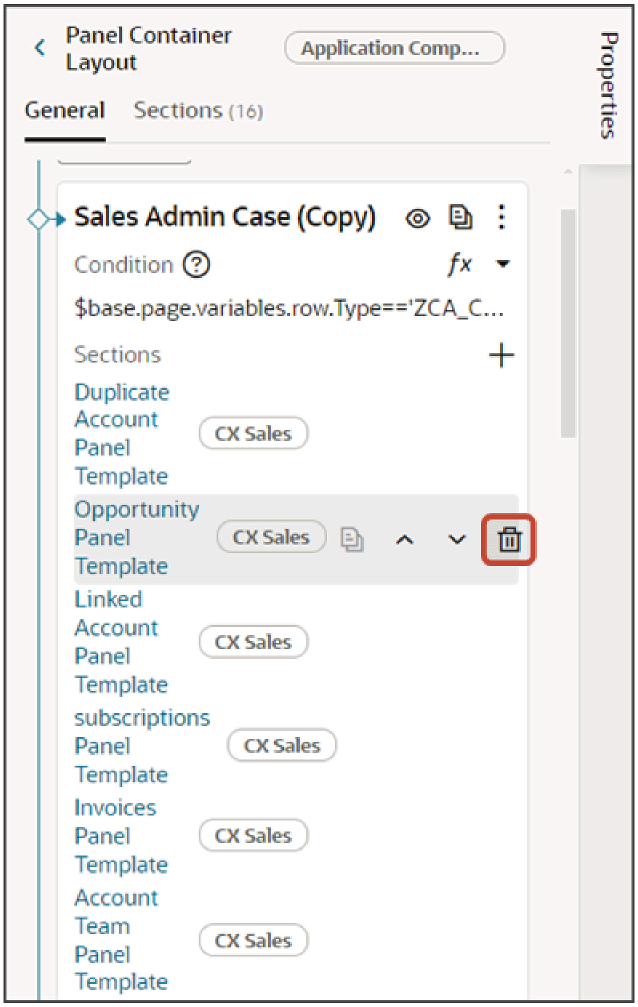
6. In the Expression Editor dialog, replace the existing expression with this new one, just for testing:

```
$base.page.variables.row.Type=='ZCA_CUSTOMER'
```

7. Click **Save**.

8. Delete the Opportunity Panel Template.

   With the field condition specified above, this means that accounts of type Customer won't see the Opportunities panel on the account detail page.



9. Let's test this change.

   From the accounts-list page, click the Preview button to see your changes in your runtime test environment.

10. The resulting preview link will be:

```
https://<servername>/fscmUI/redwood/cx-sales/accounts/accounts-list
```

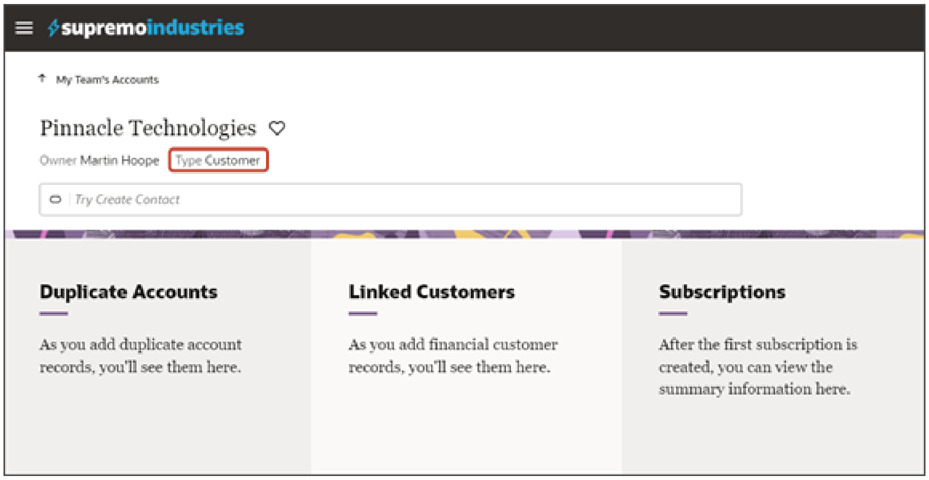
11. Change the preview link as follows:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-list
```

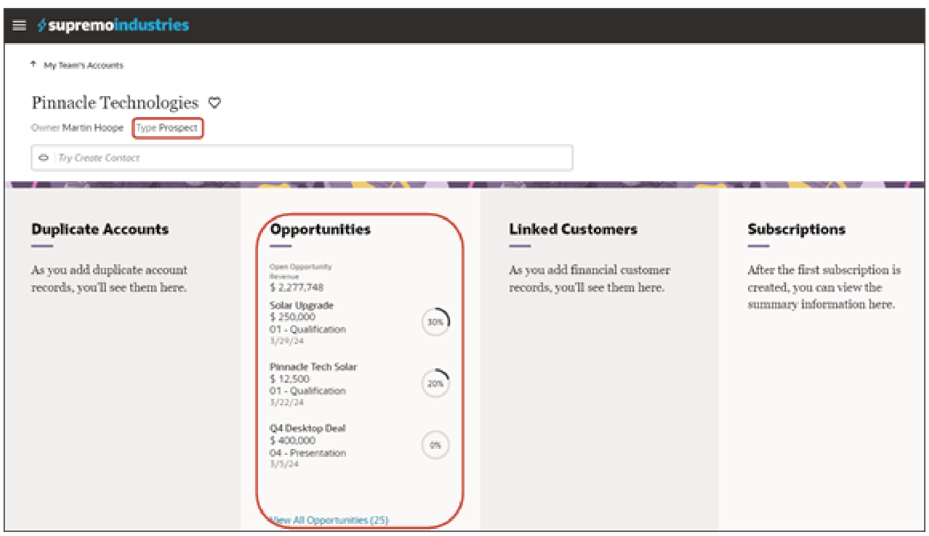
> **Note:** You must add `/application/container` to the preview link.

12. On the My Team's Accounts page, click any account.

    - If the account is of type Customer, then you won't see the Opportunities panel.

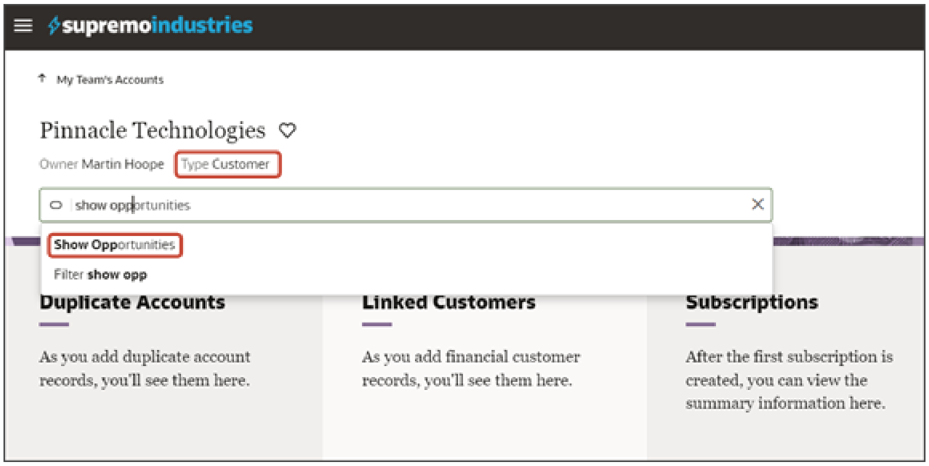


    - If the account is of type Prospect, then the Opportunities panel does display.



Create a New Subview Layout

Next, add the field condition to the subview layout, as well. It's important to add the field condition to the subview layout. Otherwise, the Show Opportunities smart action is still available from the Action Bar even when the account is a customer.



1. Navigate to Visual Builder Studio from any subview page, which you can navigate to from any panel on an account record.
2. On the accounts-detail page, click the Page Designer subtab.
3. On the Structure panel, click the **Subview Container Layout** node.
4. On the Properties pane, next to **Subview Container Layout**, click the Duplicate icon.
5. Next to the Subview Container Layout (Copy) subview layout's condition, click the Expression Editor icon.
6. In the Expression Editor dialog, add this expression:

```
$base.page.variables.row.Type=='ZCA_CUSTOMER'
```

7. Click **Save**.
8. Delete the Opportunity Subview Template.

   With the field condition specified above, this means that accounts of type Customer won't see the Show Opportunities smart action on the account detail page.
9. Let's test this change.

   From the accounts-list page, click the Preview button to see your changes in your runtime test environment.

10. The resulting preview link will be:

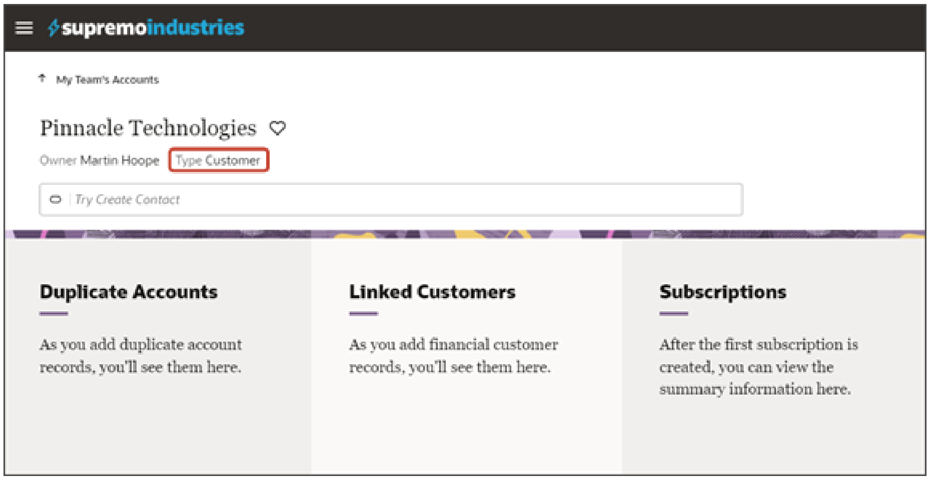
```
https://<servername>/fscmUI/redwood/cx-sales/accounts/accounts-list
```

11. Change the preview link as follows:

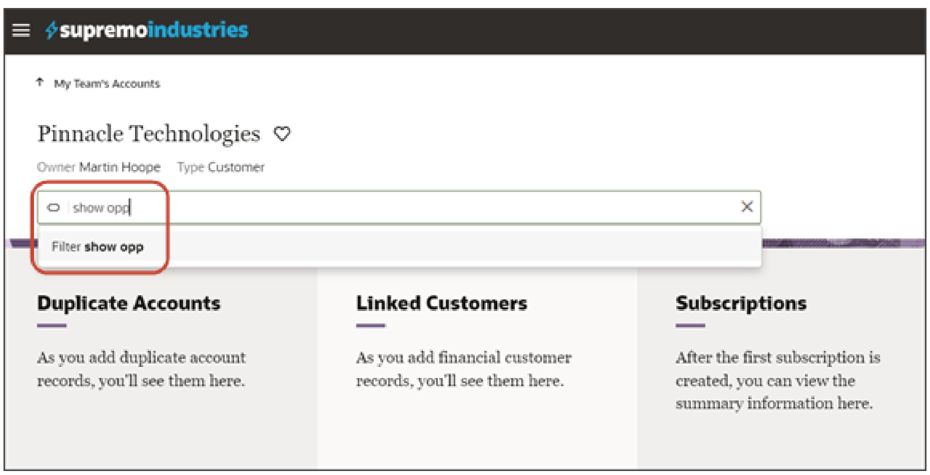
```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-list
```

> **Note:** You must add `/application/container` to the preview link.

12. On the My Team's Accounts page, click any account and make sure that the account is of type Customer. The Opportunities panel shouldn't display.



13. Test the field condition on the subview layout by checking to see if the Show Opportunities smart action is still available from the Action Bar. It shouldn't be visible anymore if the account is a customer.



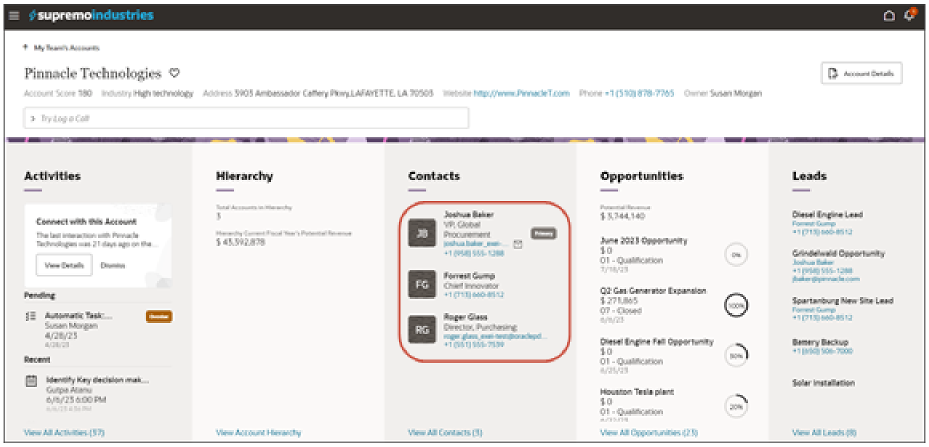
Configure the Contents of a Panel

An individual record's detail page includes key information displayed in a region of panels. Each panel contains information related to the record, such as related contacts and opportunities. Most panels display information in a list format. You can configure these lists using Oracle Visual Builder Studio.

Note, however, that you can't configure the Activities panel.

What's Inside a Panel?

A panel often contains a list, which you can configure. Here's an example of a list inside a panel:



Lists can display up to 5 records, depending on screen size. If the screen size is small, then the list automatically adjusts to display fewer records. However, users can click the View All link that displays at the bottom of the panel to navigate to a second page to see all records in the list. This second page is called the subview.

What Can You Change in a List?

In Visual Builder Studio, you can modify the information that displays in each list.

You can:

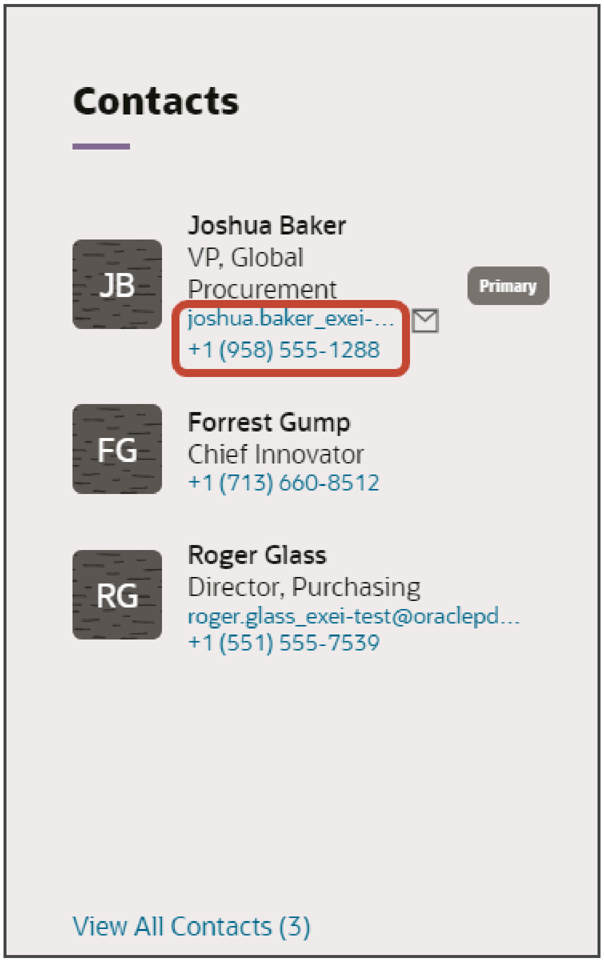
- Add and remove fields
- Change the display order of fields in the list

This topic illustrates how to change the display order of fields that display on panels on an account's detail page. We'll look at both the Contacts panel and Opportunities panel.

To configure the subview, see *Configure the Subview Layout*.

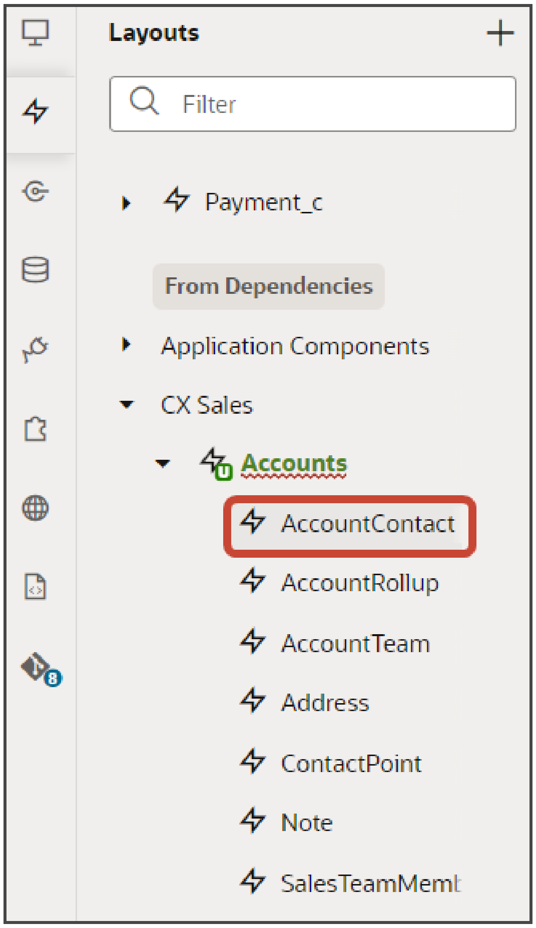
Change the Display Order of Contact Panel Fields

Let's change the display order of fields in a panel list. In this example, we'll switch the order of the email and phone number fields on the Contacts panel on the Account detail page.



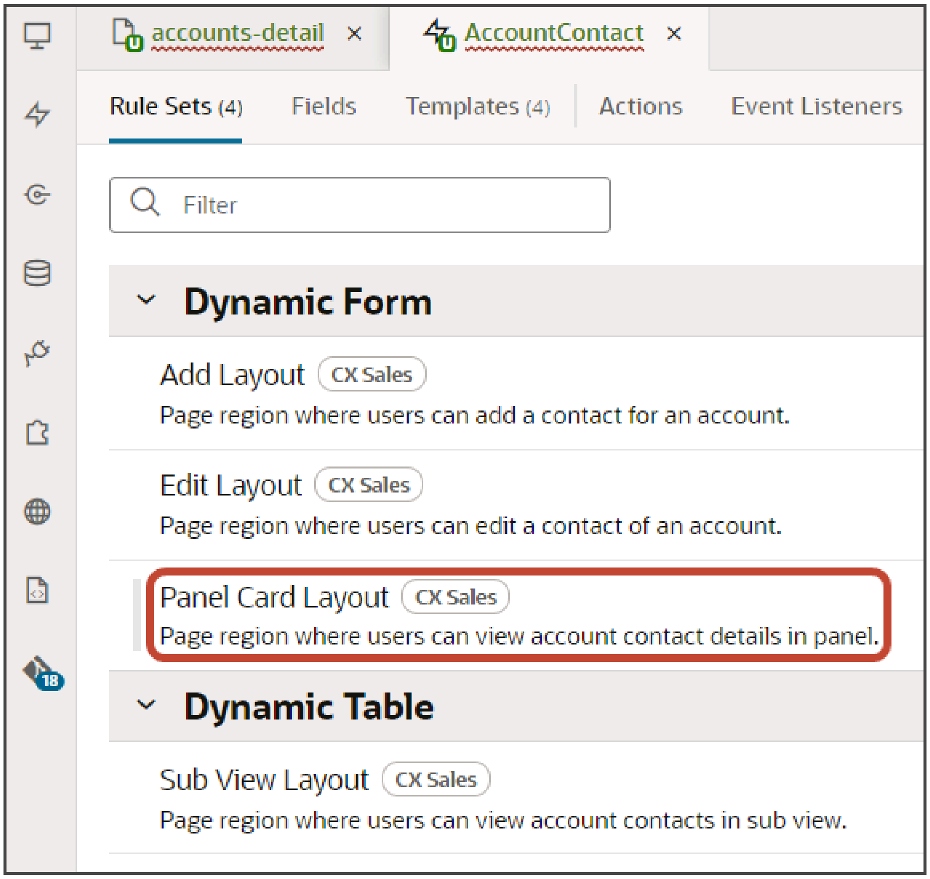
1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Accounts > AccountContact.

   The AccountContact node contains the rule sets for the Contacts panel on the Account object.

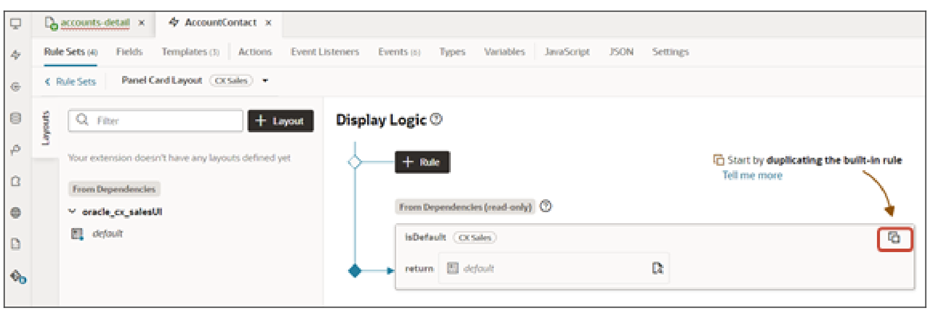


Note: When configuring the contents of a panel, consider what kind of relationship the panel's object has with the primary object. In this case, the Account object has a many-to-many relationship with Contact. This means that you'll find layouts for the Contact object on the AccountContact node, nested under the Accounts node.

2. On the AccountContact tab > Rule Sets subtab, click the Panel Card Layout.



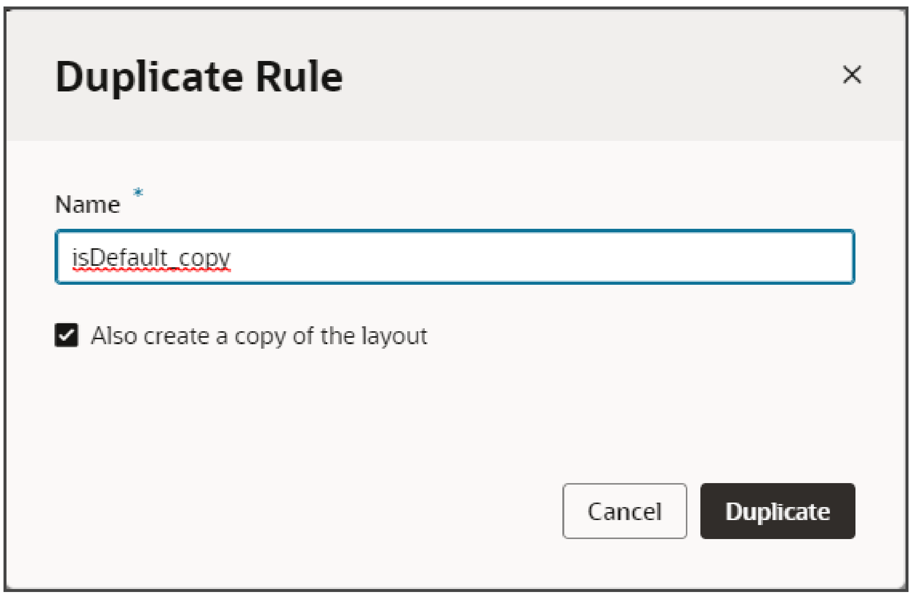
Both a default layout as well as a default rule are displayed for the Panel Card Layout.



3. Click the Duplicate Rule icon.

4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.



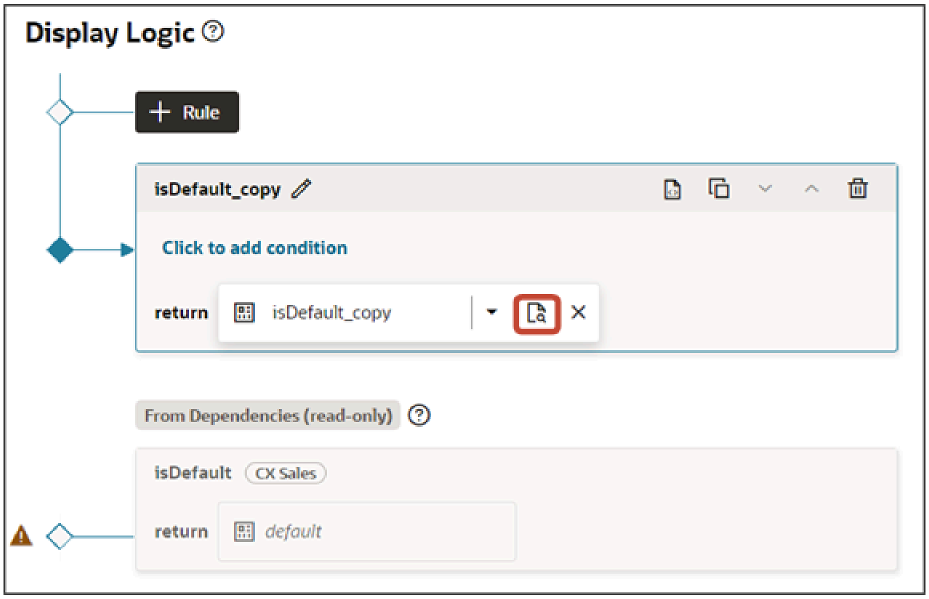
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

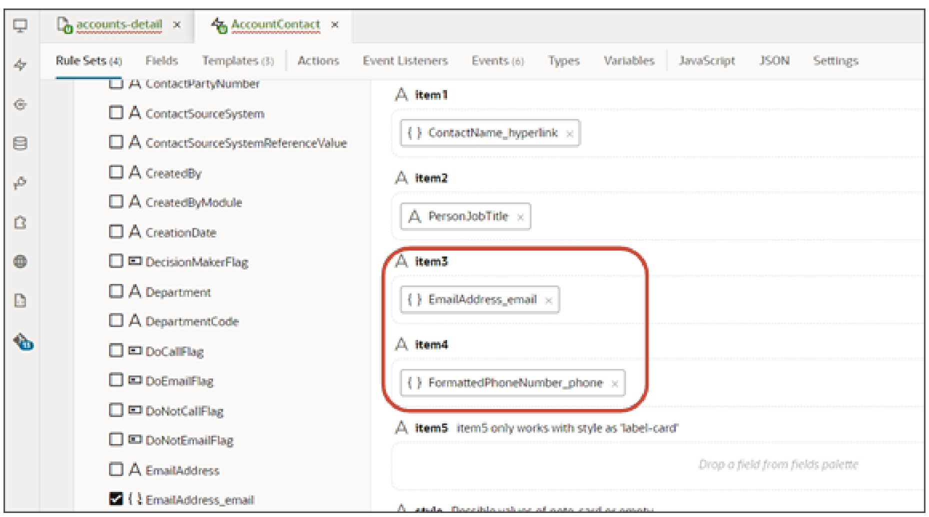
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

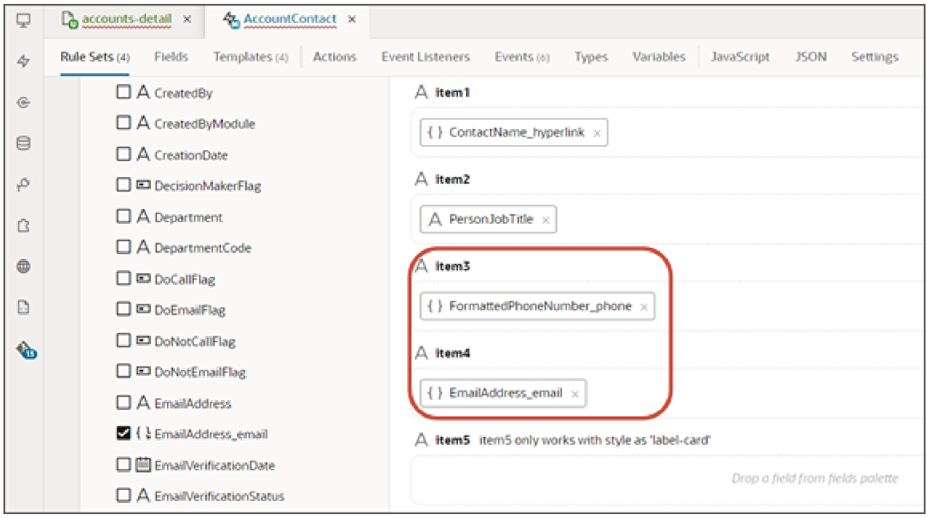
   a. Click the Open icon to edit the copied layout.



   b. Scroll down the list of fields in the layout until you locate the email and phone fields.



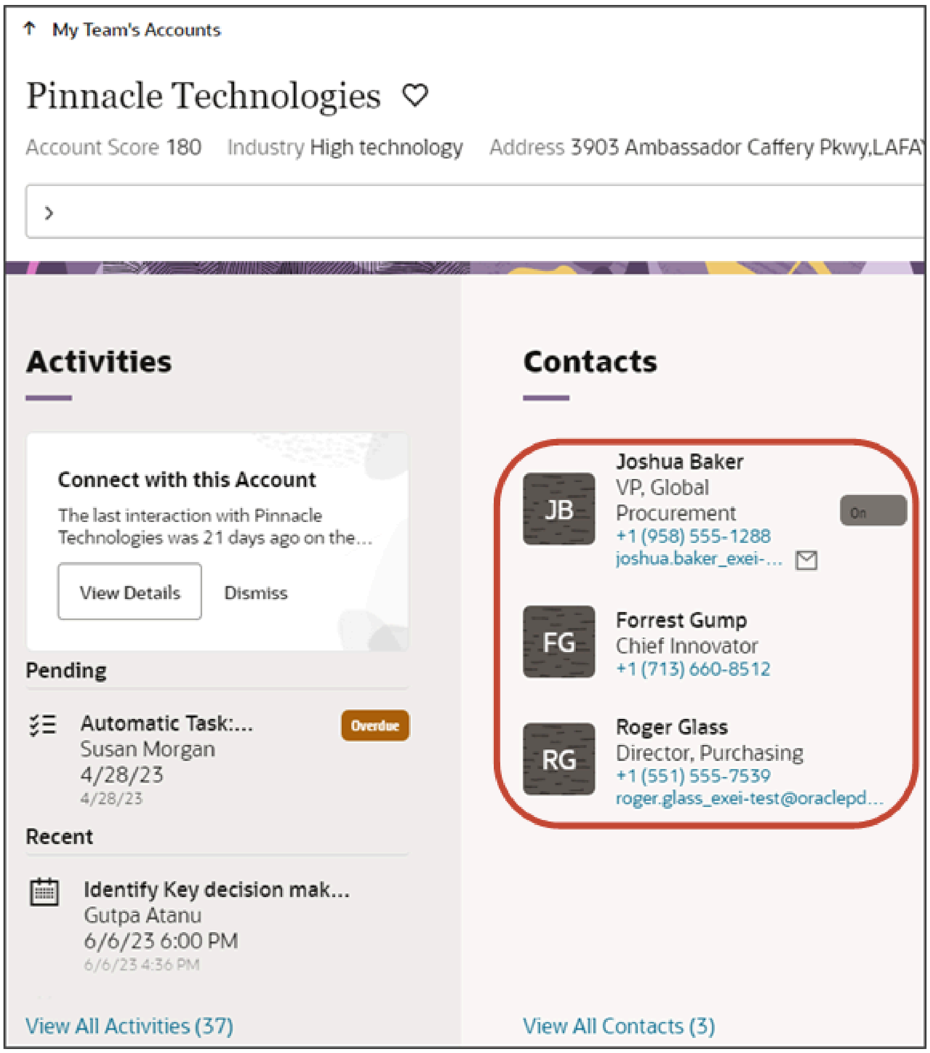
   c. Delete each field from the Item3 and Item4 slots, and then add the fields back. This time, however, switch the order so that the phone field is in the Item3 slot and the email field is in the Item4 slot.


accounts-detail
AccountContact
Rule Sets (4)
Fields
Templates (4)
Actions
Event Listeners
Events (6)
Types
Variables
JavaScript
JSON
Settings
CreatedBy
CreatedByModule
CreationDate
DecisionMakerFlag
Department
DepartmentCode
DoCallFlag
DoEmailFlag
DoNotCallFlag
DoNotEmailFlag
EmailAddress
EmailAddress_email
EmailVerificationDate
EmailVerificationStatus
item1
ContactName_hyperlink
item2
PersonJobTitle
item3
FormattedPhoneNumber_phone
item4
EmailAddress_email
item5 item5 only works with style as 'label-card'
Drop a field from fields palette

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

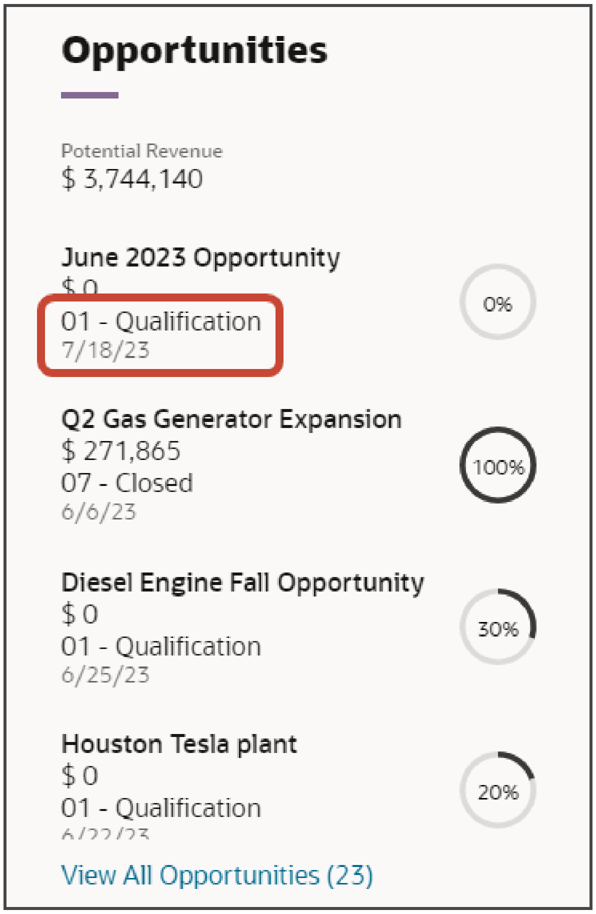
The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```



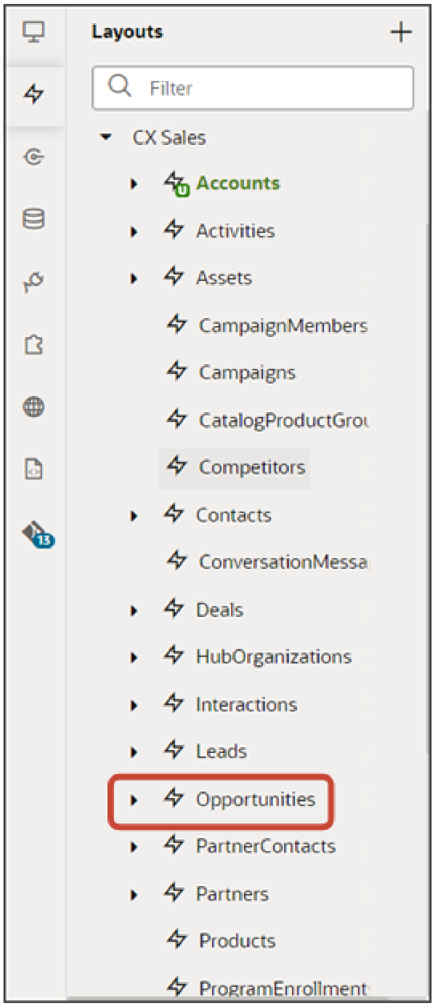
Change the Display Order of Opportunity Panel Fields

In this example, we'll switch the order of the sales stage and effective date fields on the Opportunities panel on the Account object.



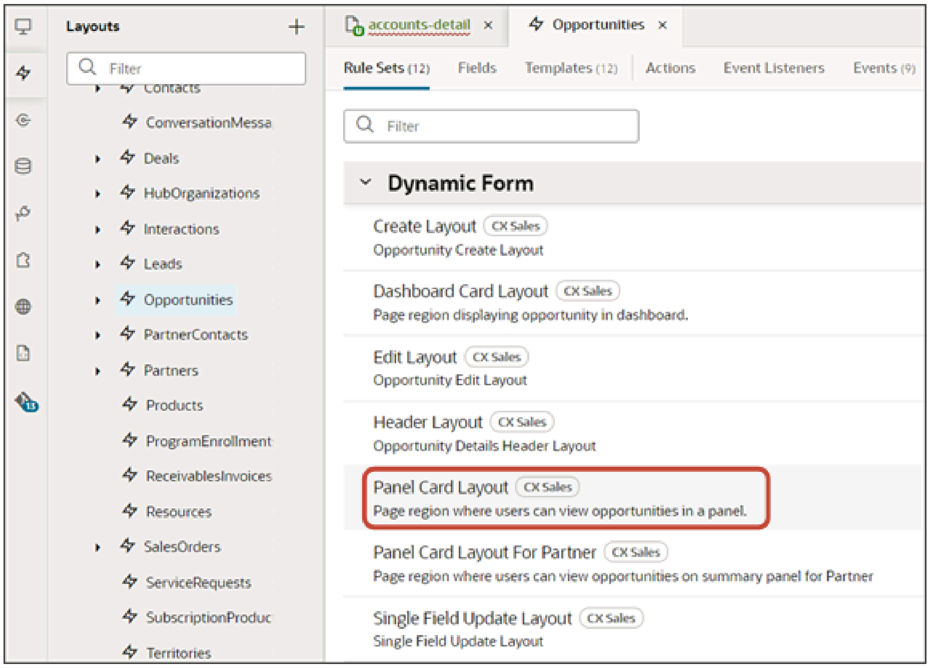
1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Opportunities.

   The Opportunities node contains the rule sets for the Opportunities panel on the Account object.

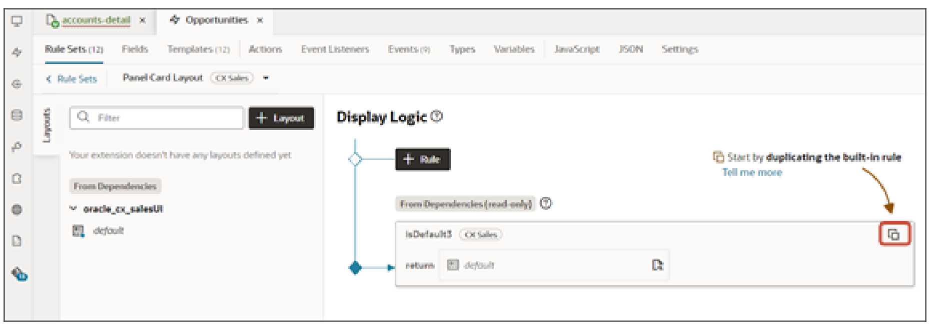


Note: When configuring the contents of a panel, consider what kind of relationship the panel's object has with the primary object. In this case, the Account object has a one-to-many relationship with Opportunity. This means that you'll find layouts for the Opportunity object on the Opportunities node.

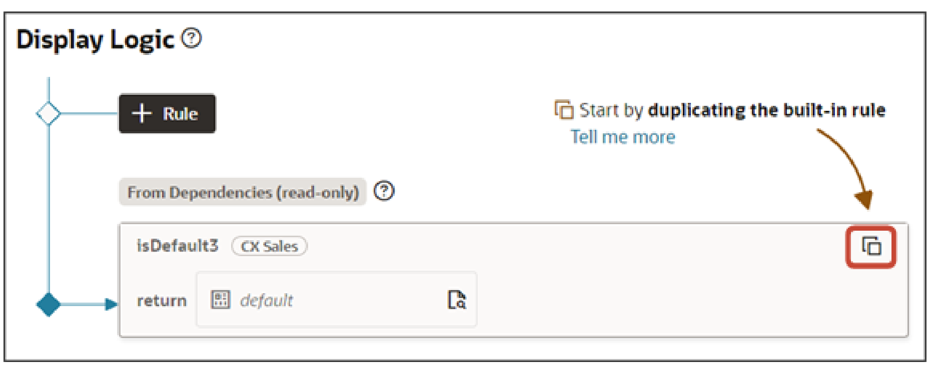
2. On the Opportunities tab > Rule Sets subtab, click the Panel Card Layout.



Both a default layout as well as a default rule are displayed for the Panel Card Layout.

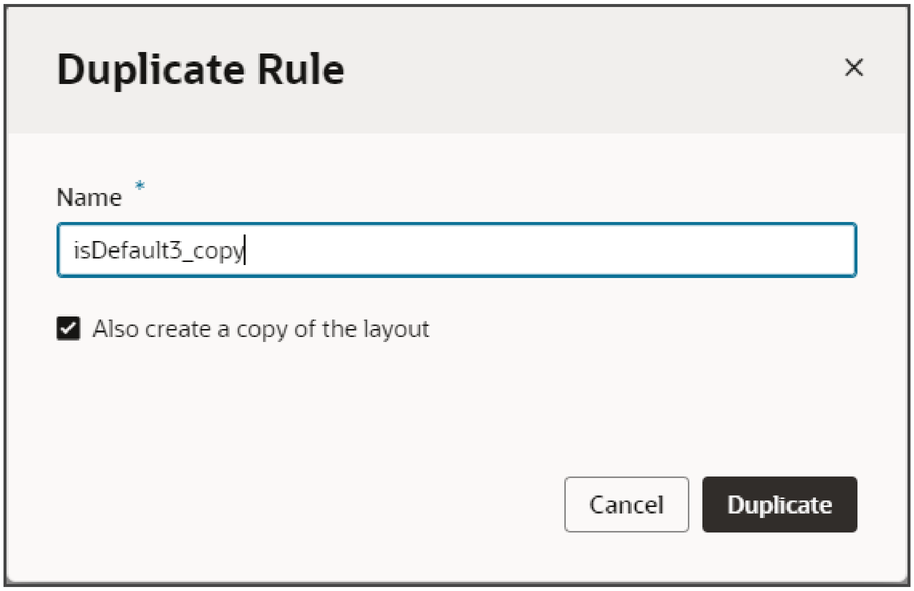


3. Click the Duplicate Rule icon.



4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.

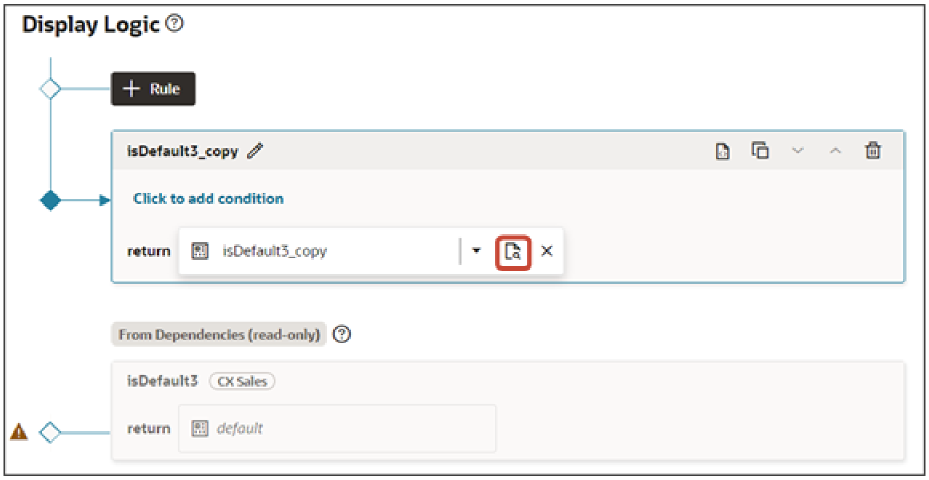
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

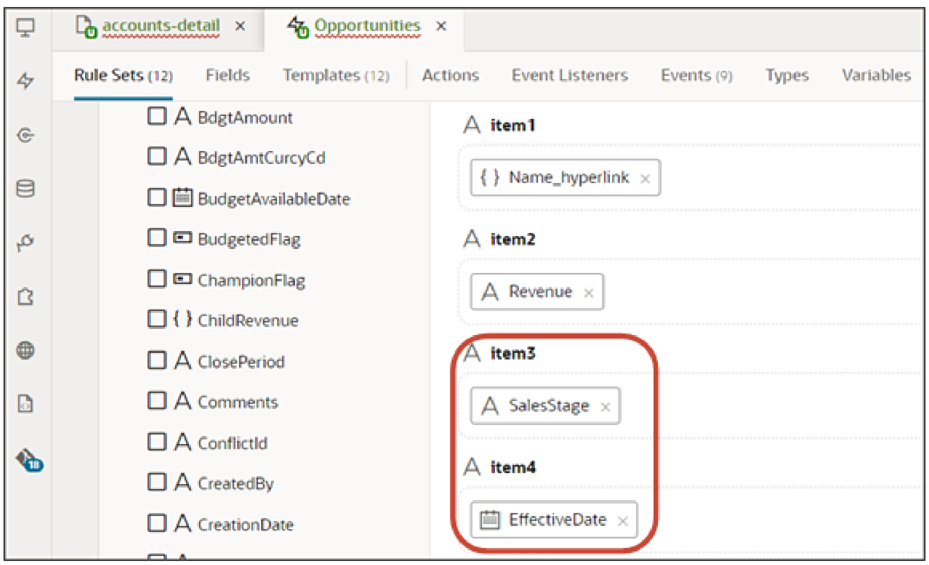
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

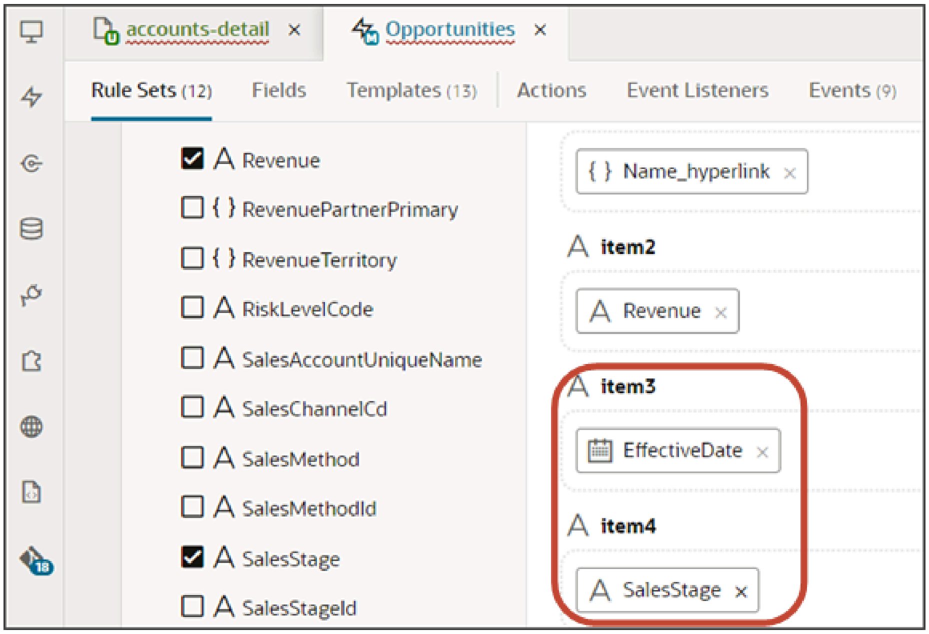
    a. Click the Open icon to edit the copied layout.



    b. Scroll down the list of fields in the layout until you locate the sales stage and effective date fields.



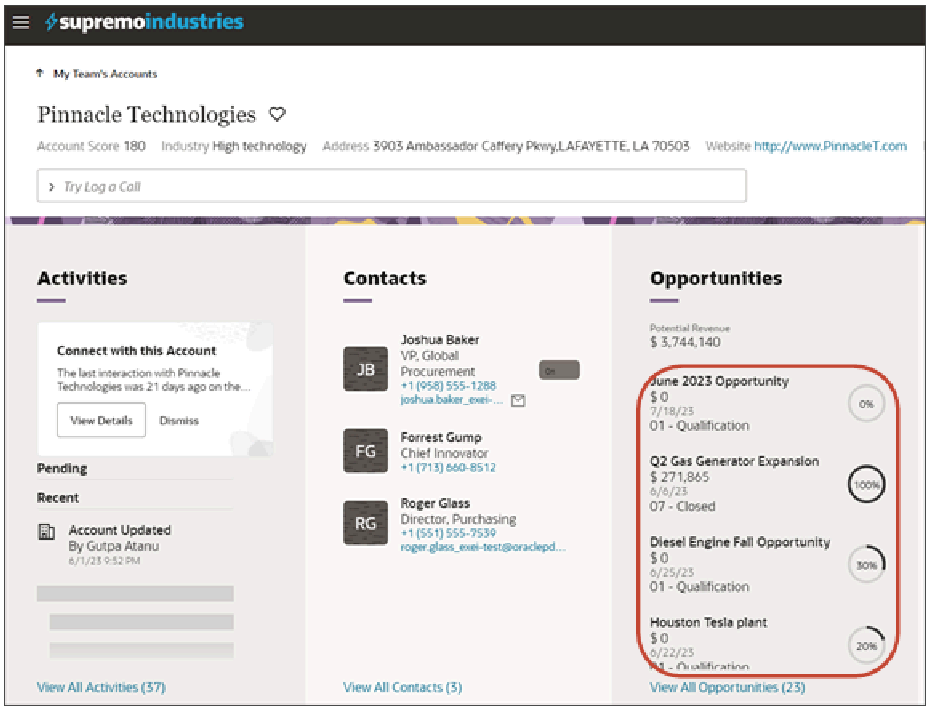
    c. Delete each field from the Item3 and Item4 slots, and then add the fields back but switch the order.


accounts-detail
Opportunities
Rule Sets (12)
Fields
Templates (13)
Actions
Event Listeners
Events (9)
Revenue
RevenuePartnerPrimary
RevenueTerritory
RiskLevelCode
SalesAccountUniqueName
SalesChannelCd
SalesMethod
SalesMethodId
SalesStage
SalesStageId
Name_hyperlink
item2
Revenue
item3
EffectiveDate
item4
SalesStage

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

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000003513233&puid=7050&view=foldout
```



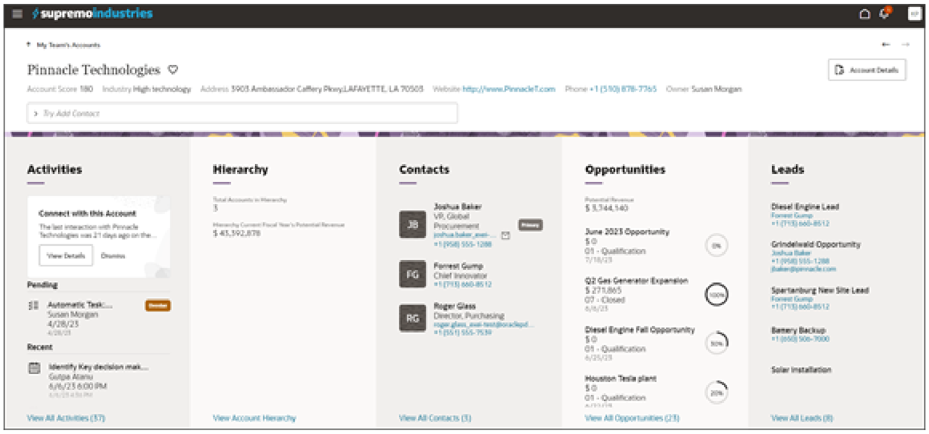
Configure the Subview Layout

An object's detail page includes a region of panels with information. Each panel, however, can display only a few records due to panel size. To see all records, users can navigate to a second page called a subview. This topic illustrates how to modify those subview pages using Oracle Visual Builder Studio.

What's Inside the Subview?

A subview contains a list of all records that the panel, due to limited real estate, can't display.

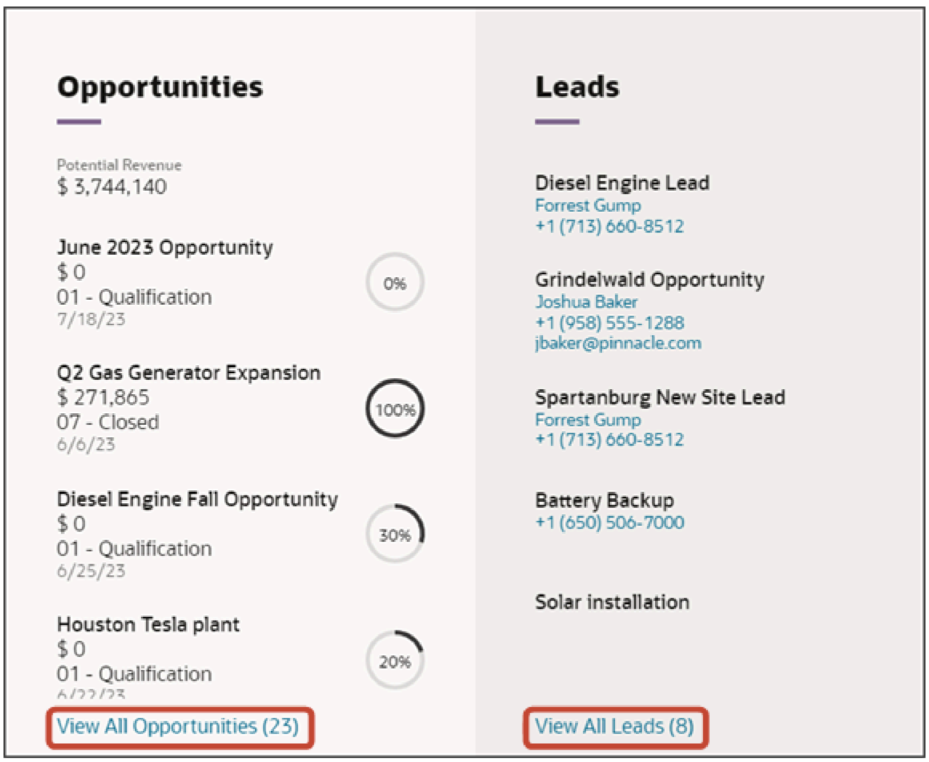
For example, here's an example of an account detail page with 5 panels:



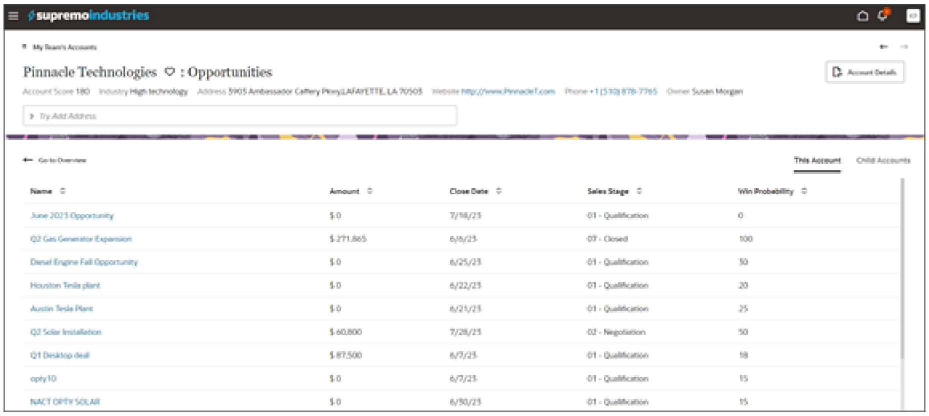
Notice how each panel displays only a few records.

To see all records, users can click the View All link that displays at the bottom of the panel.

Here's an example of some View All links. Note that after the link itself, a number indicates the number of total records listed on the subview.



The subview displays all those records in a table.



What Can You Change in a Subview Table?

In Visual Builder Studio, you can modify the information that displays in a subview table.

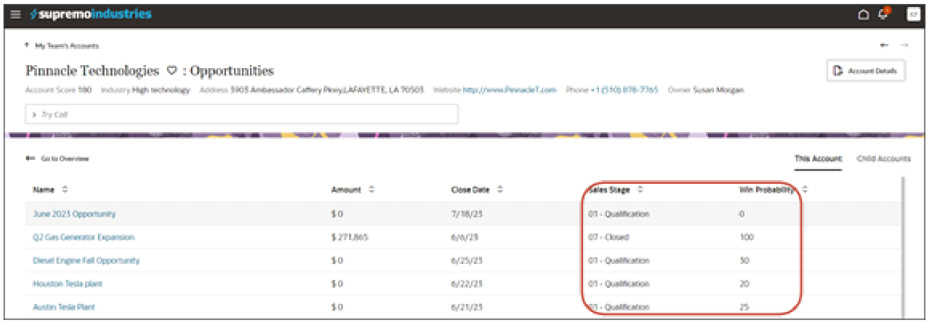
You can:

- Add and remove columns
- Change the display order of columns in the table

This topic illustrates how to change the display order of columns in a subview table. We'll look at the Opportunities subview that's available from an account detail page.

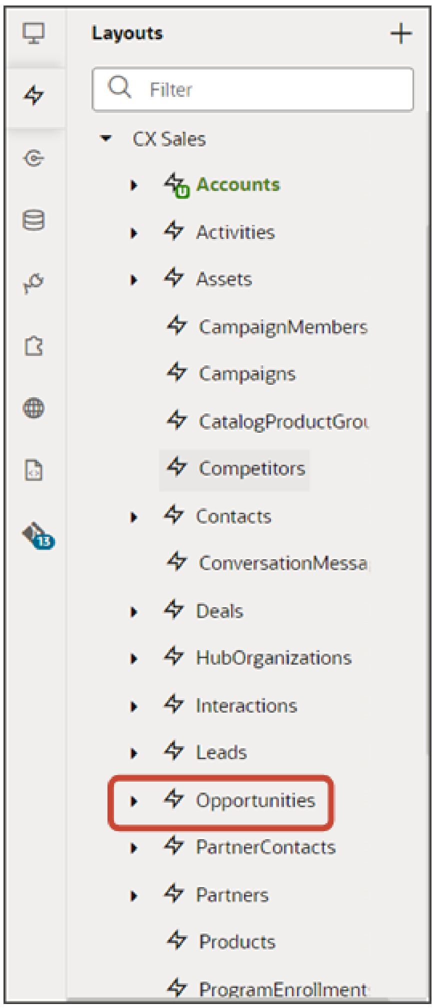
Change the Display Order of Opportunity Subview Columns

Let's change the display order of columns in a subview table. In this example, we'll switch the order of the sales stage and win probability columns on the Opportunities subview, accessed from the Opportunities panel on the Account detail page.

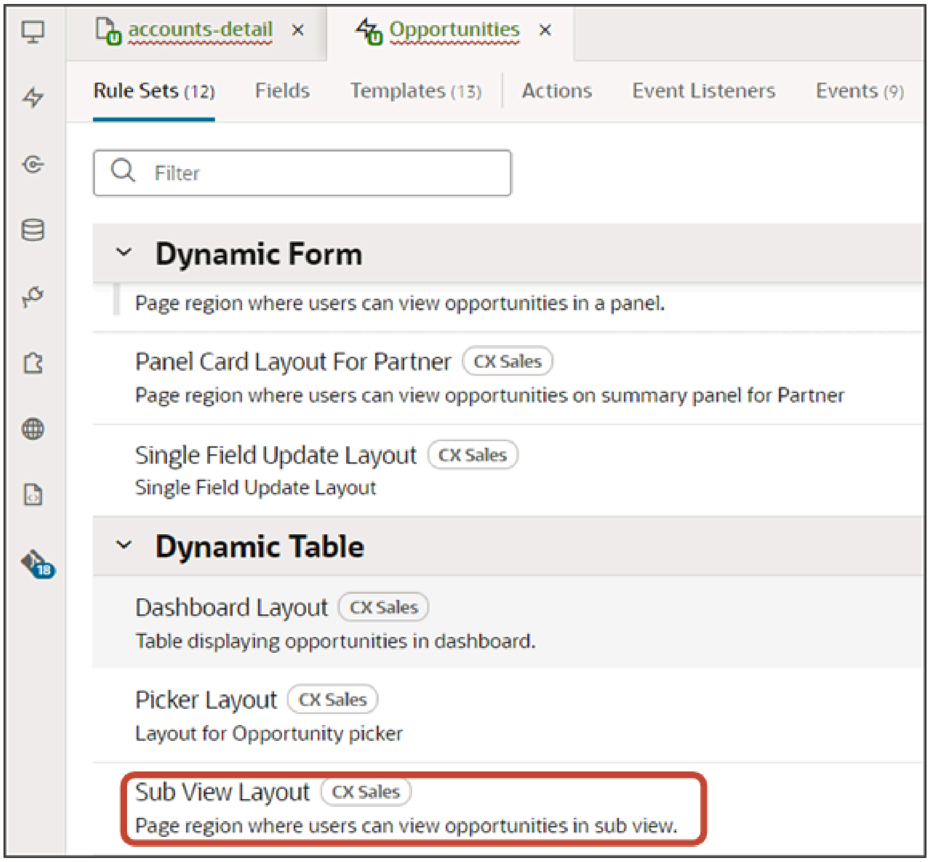


1. In Visual Builder Studio, navigate to the Layouts tab and expand the CX Sales node > Opportunities.

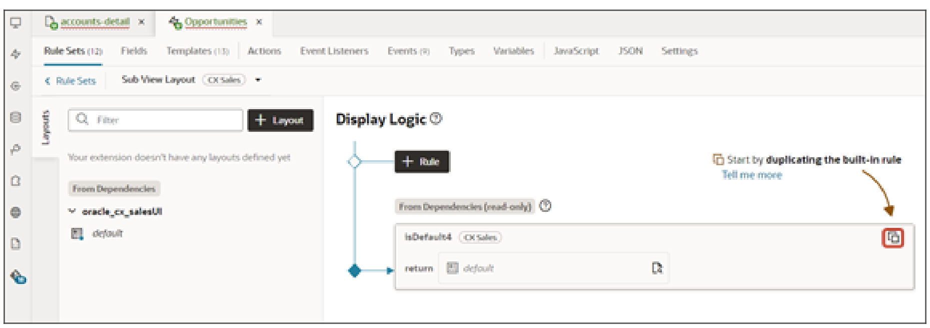
   The Opportunities node contains the rule sets for the Opportunities panel on the Account object.



2. On the Opportunities tab > Rule Sets subtab, click the Sub View Layout.



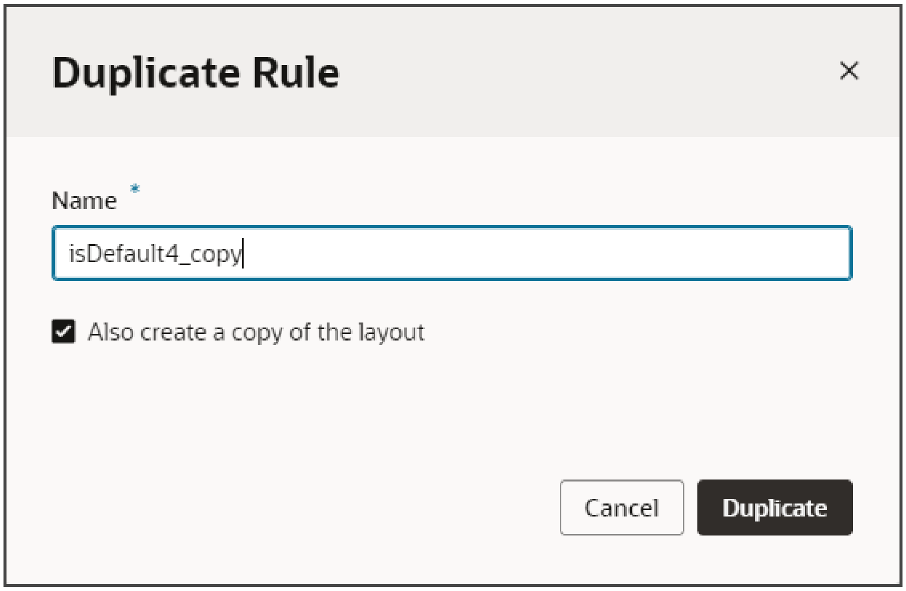
Both a default layout as well as a default rule are displayed for the Sub View Layout.



3. Click the Duplicate Rule icon.

4. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

   Also, make sure that the **Also create a copy of the layout** check box is selected.

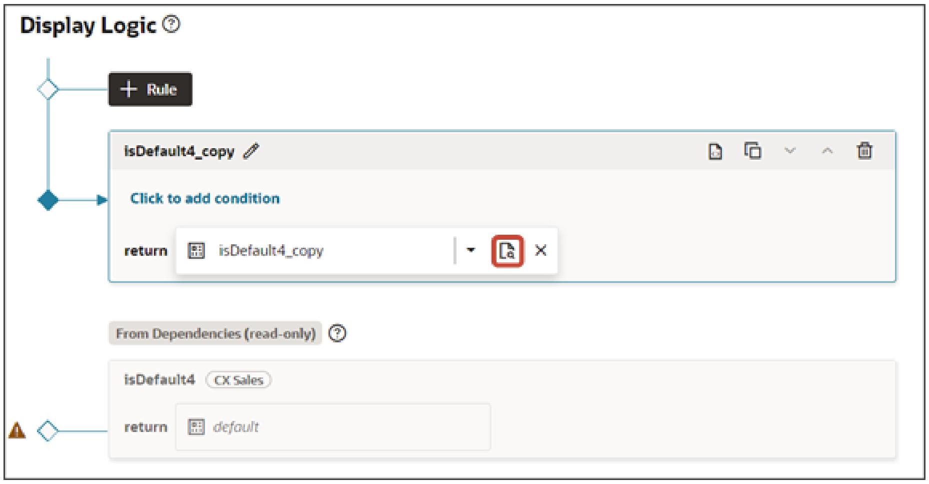
5. Click **Duplicate**.

   The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

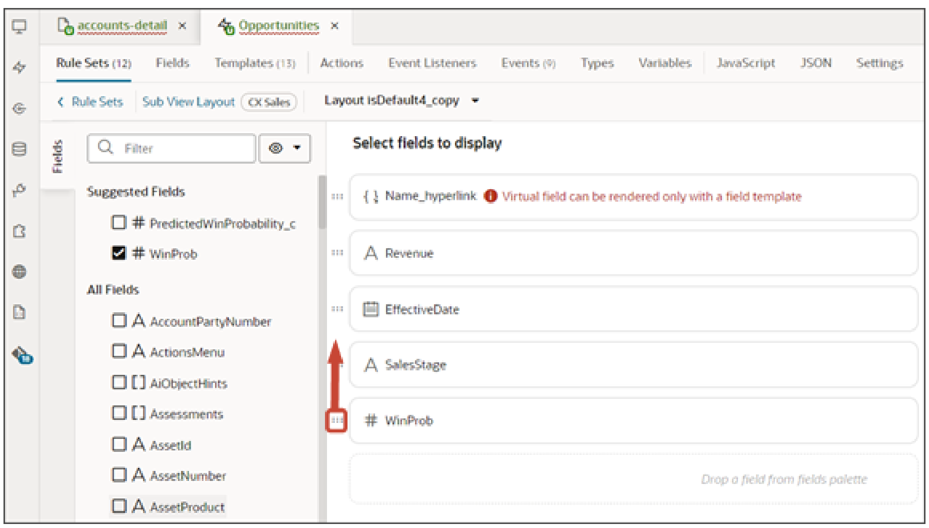
   In this example, we're not adding any conditions which means that the associated layout will always be displayed.

6. Modify the rule's copied layout.

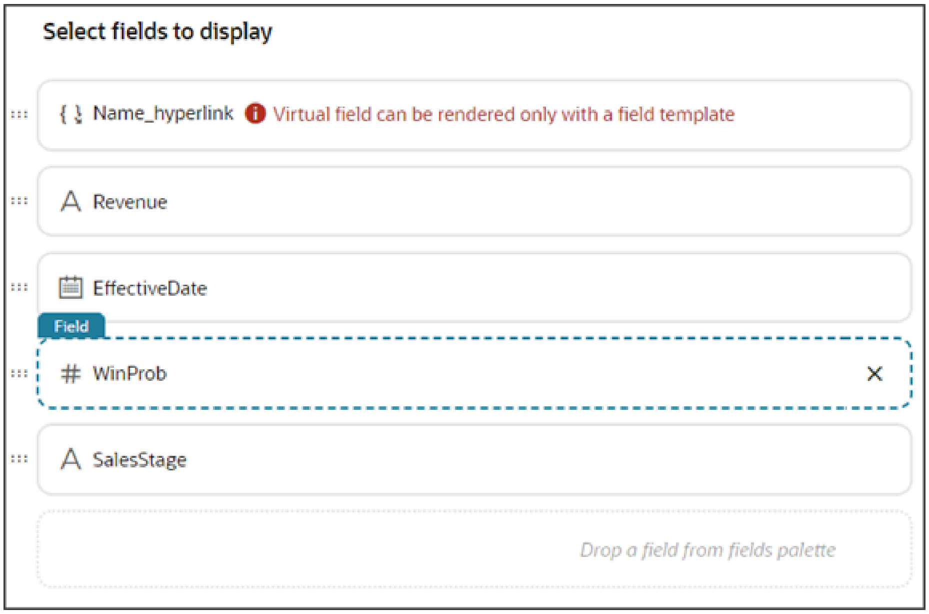
   a. Click the Open icon to edit the copied layout.



   b. In the list of fields in the layout, use the handle next to the win probability field to move it above the sales stage field.



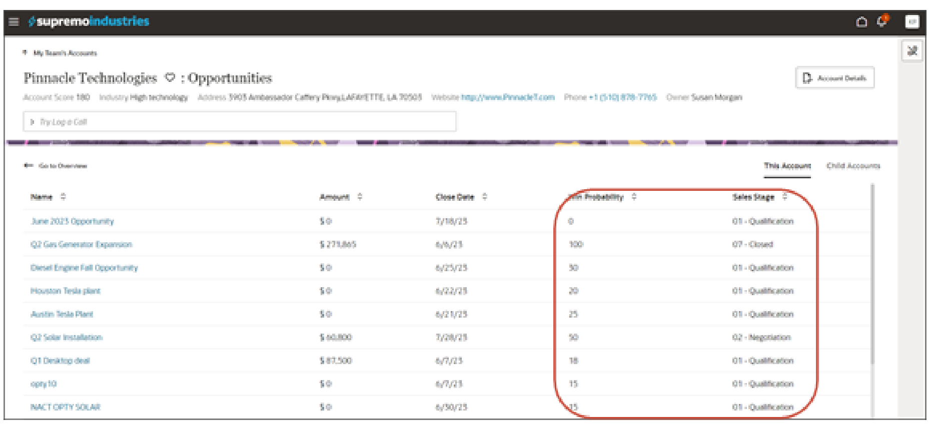
Here's a screenshot of the final location of the win probability field.



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

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?id=300000003513233&puid=7050&view=foldout
```

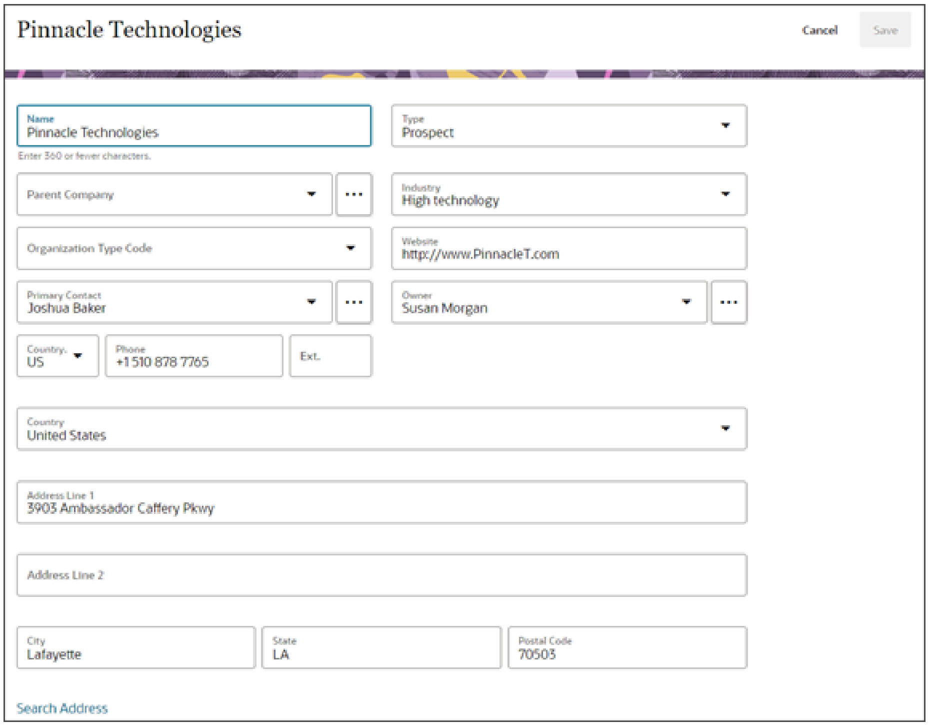


Configure the Create and Edit Layouts

This topic illustrates how to add or remove a field, and how to reorder fields, on the Create and Edit Account pages.

What's the Scenario?

Here's a screenshot of the Edit Account page.



In this example, we'll move the Website field to display to the right of the Name field.

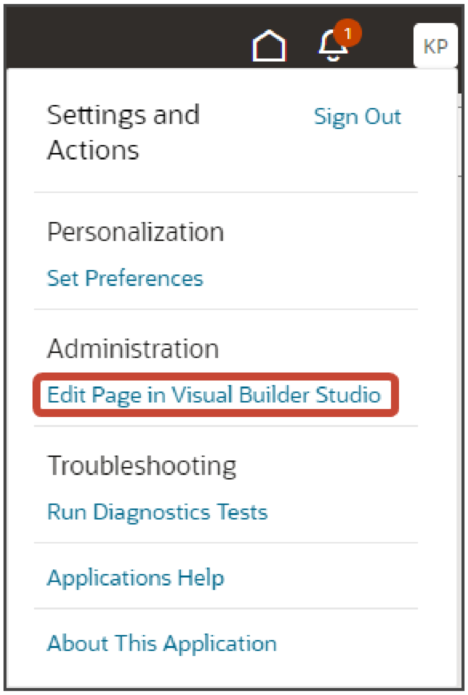
Modify the Edit Account Layout

1. Navigate to the page that displays the area you want to extend.

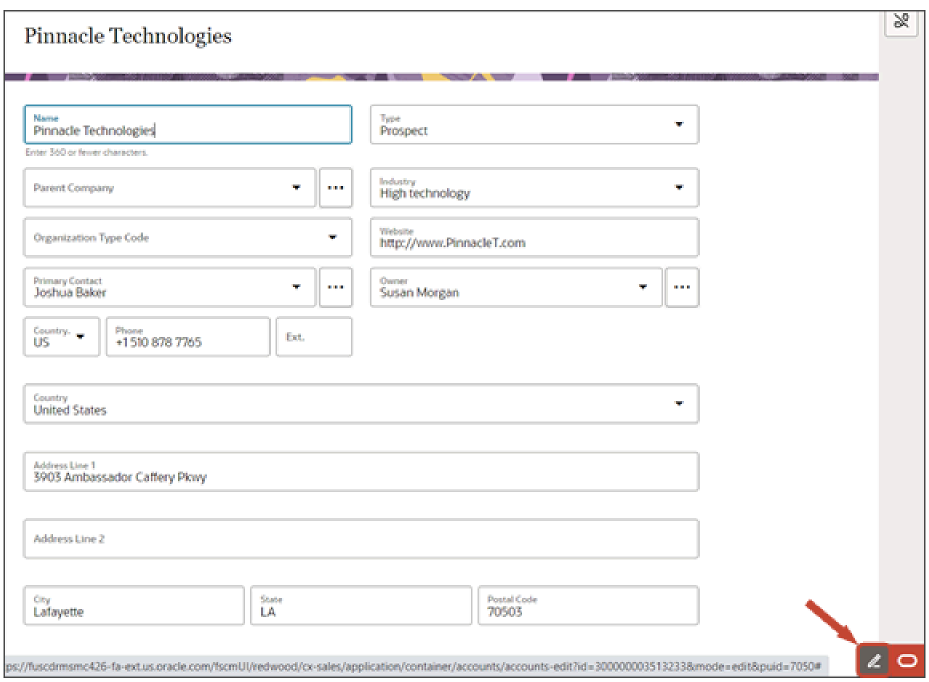
   In this example, navigate to the My Team's Accounts page, click an account record, then click **Account Details**.

2. To access Visual Builder Studio, you have one of two options:

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



    - If you don't see the Settings and Actions menu, then in the lower right corner of the page, hover over the Ask Oracle icon and click **Edit Page in Visual Builder Studio**.

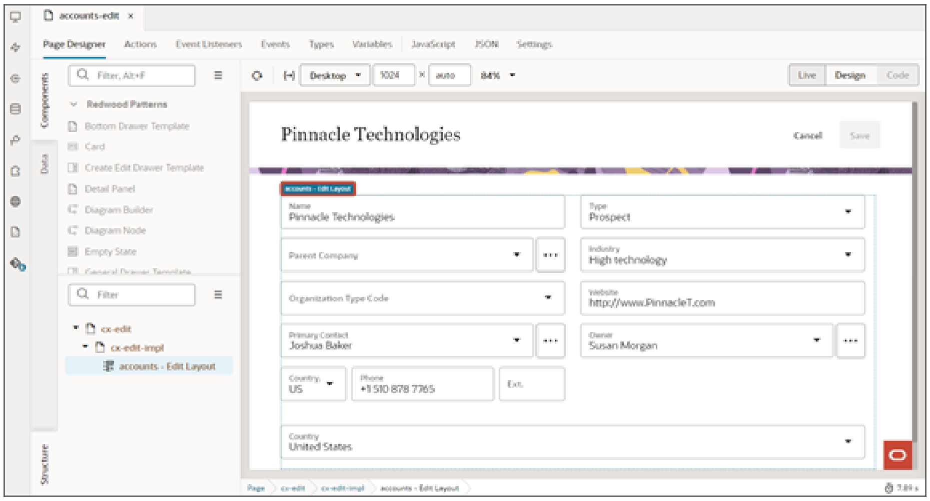


3. If you're working on multiple projects, then select the Extend Redwood CX Sales project to keep your fragments extension separate from non-fragments changes.

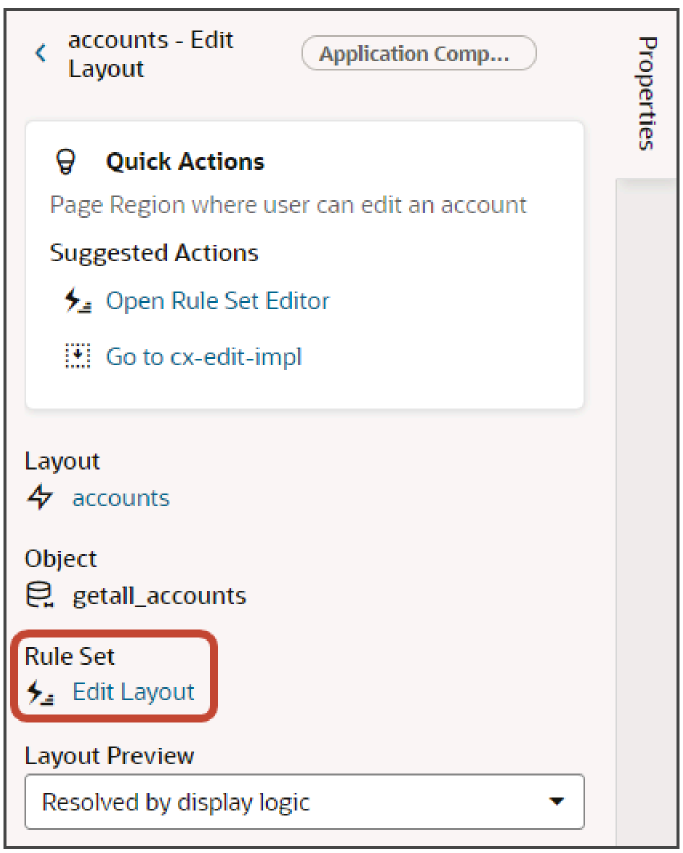
4. When Visual Builder Studio opens, the Edit Account page displays in the canvas.

   On the Structure panel, click the **accounts - Edit Layout** node.

   The edit layout is highlighted in the canvas.

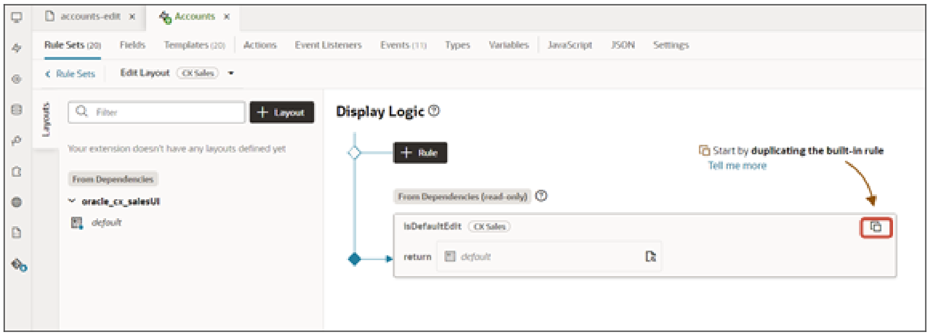


5. On the Properties pane, click the **Edit Layout** rule set.



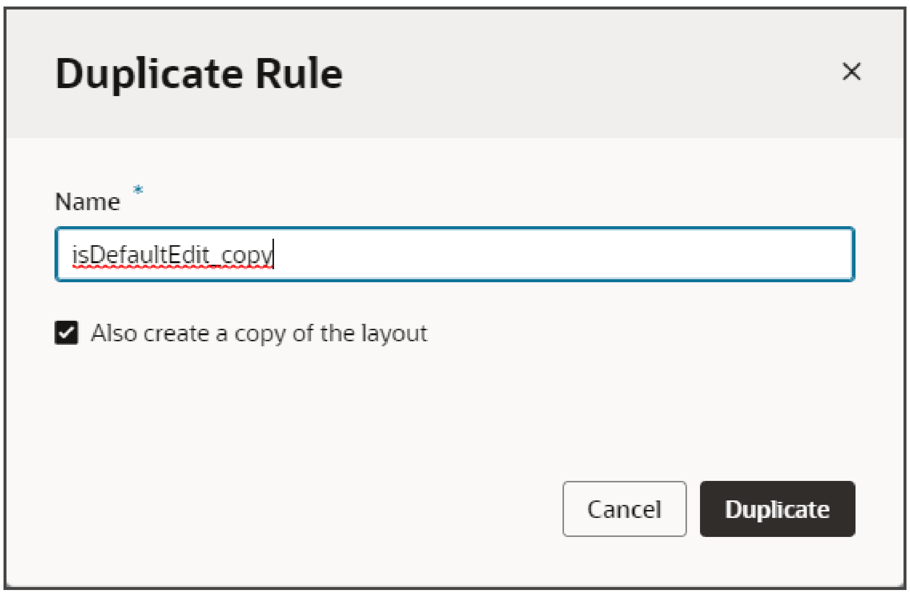
   The Accounts layout tab > Rule Sets subtab displays. The Rule Sets subtab includes both a default layout as well as a default rule. Let's create a new rule.

6. Click the Duplicate Rule icon.



7. In the Duplicate Rule dialog, accept the default rule name or enter a new name. The name you enter here is both the rule name and also the layout name, so enter a layout name that makes sense for you.

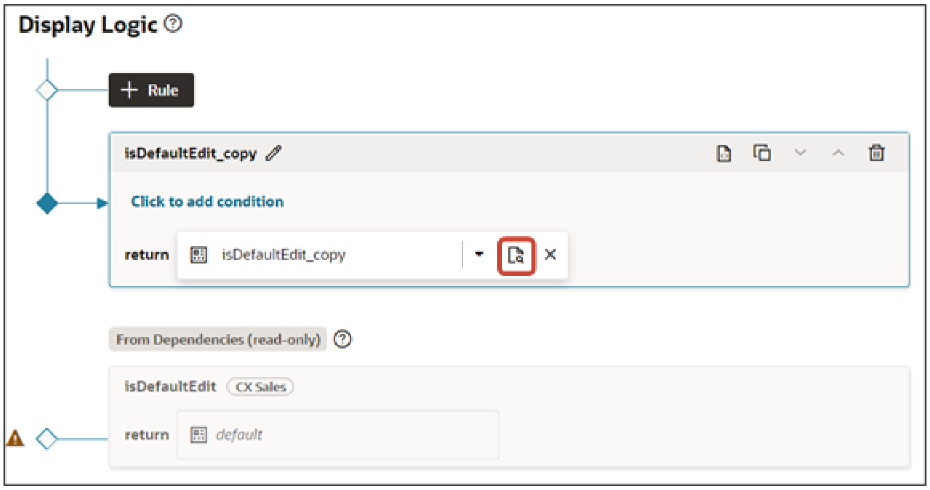
   Make sure that the **Also create a copy of the layout** check box is selected, and then click **Duplicate**.



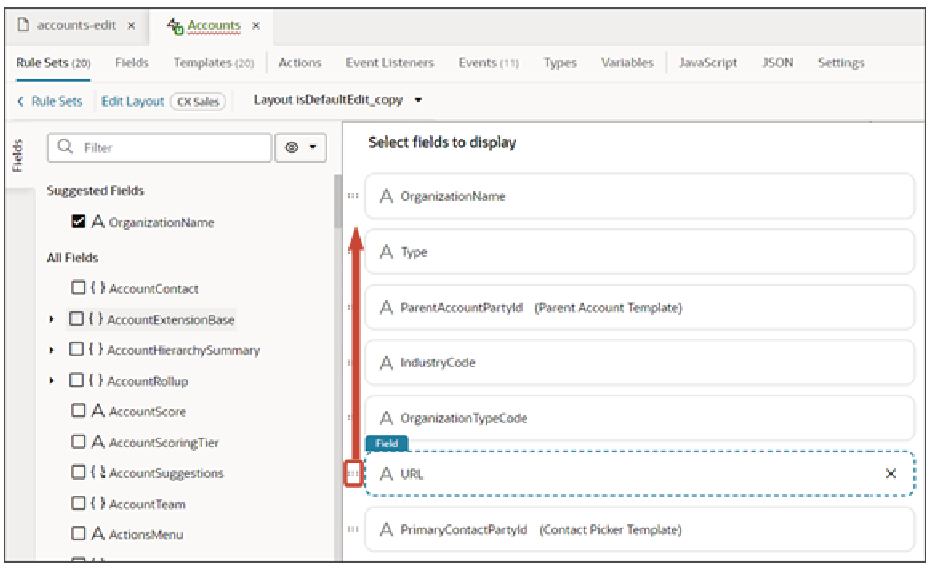
The new rule displays at the top of the list of existing rules, which means that this rule will be evaluated first at runtime. If the rule's conditions are met, then the associated layout is displayed to the user.

In this example, we're not adding any conditions which means that the associated layout will always be displayed.

8. Let's modify the rule's copied layout. Click the Open icon to edit the copied layout.



9. In this example, we want to move the position of the Website field. Locate the Website field and use the handle to reposition the field so that it displays after the Name field.

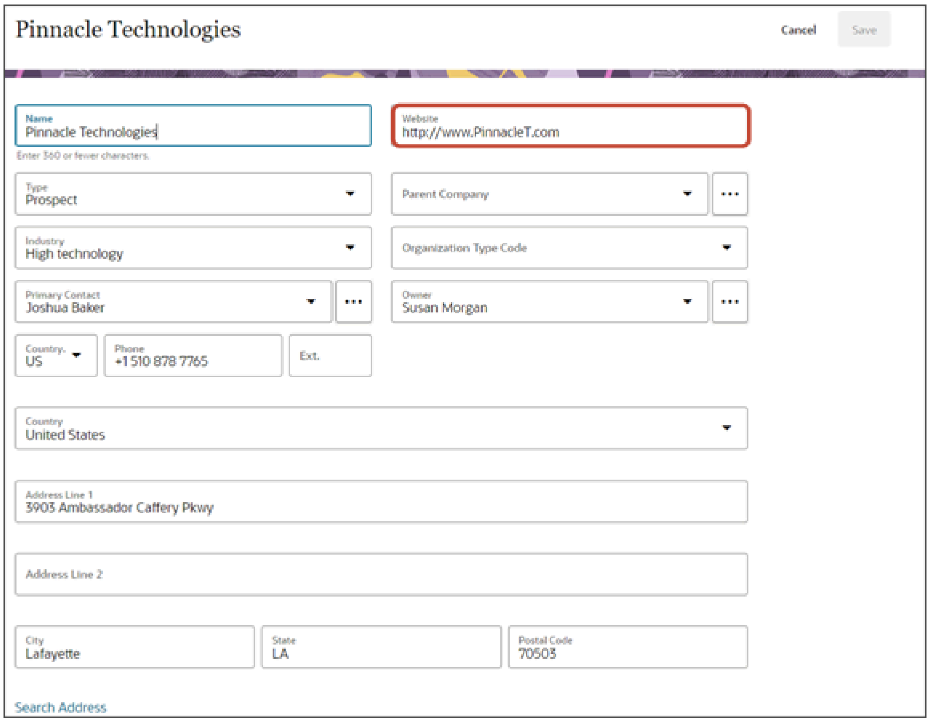


10. Click the Preview button to see your changes in your runtime test environment.

The preview link must include the `application/container` segments in the URL. If not, then change the preview link using the following example URL:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-edit?
id=300000003513233&mode=edit&puid=7050
```

This screenshot illustrates the new location of the Website field.



Link to a Smart Action Using a URL

You can construct a URL that calls a smart action in the Redwood version of Sales. Construct this URL whenever needed and then use it as a deeplink. Depending on the smart action added to the URL, clicking the link will either execute a smart action without involving a UI (to delete a record, for example) or navigate directly to an open drawer on a Sales page (to create a record, for example).

To construct the URL, append the smart action ID as a parameter to the detail page URL.

1. Obtain the smart action ID.
   You can retrieve the smart action ID from Application Composer.
2. Obtain the URL of the detail page.
   For example:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout
```

3. Append the smart action ID parameter as follows:

```
&actionId=<smart action ID>
```

4. The resulting URL can be used to link to a smart action:

   For example:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout&actionId=SDA-Delete-accounts
```

   Note that once the action is completed, the URL changes to:

```
https://<servername>/fscmUI/redwood/cx-sales/application/container/accounts/accounts-detail?
id=300000008600956&puid=38005&view=foldout&actionId=completed
```