



edunet  
foundation



## LAB MANUAL 2

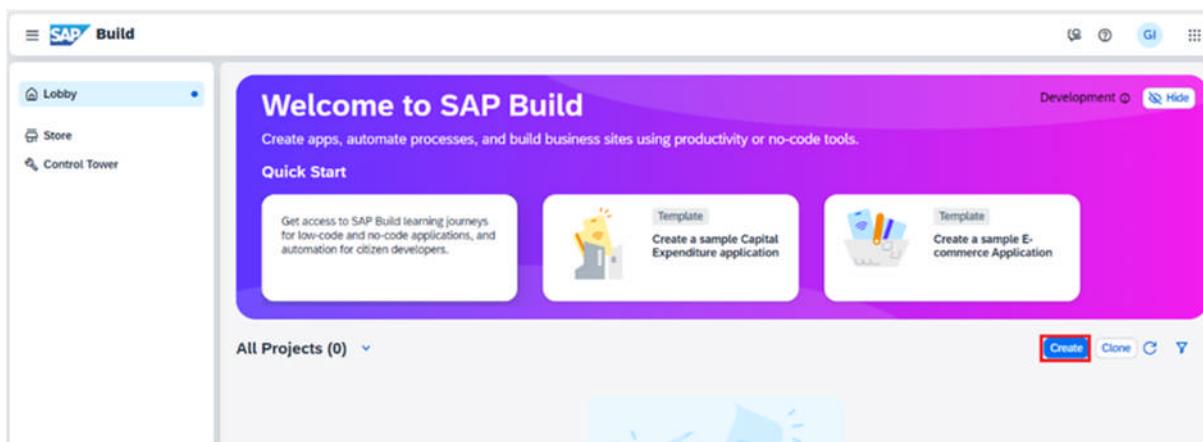
# Create a Full-Stack SAP Fiori Application with Joule in SAP Build Code

Disclaimer: The content is curated from online/offline resources and used for educational purpose only

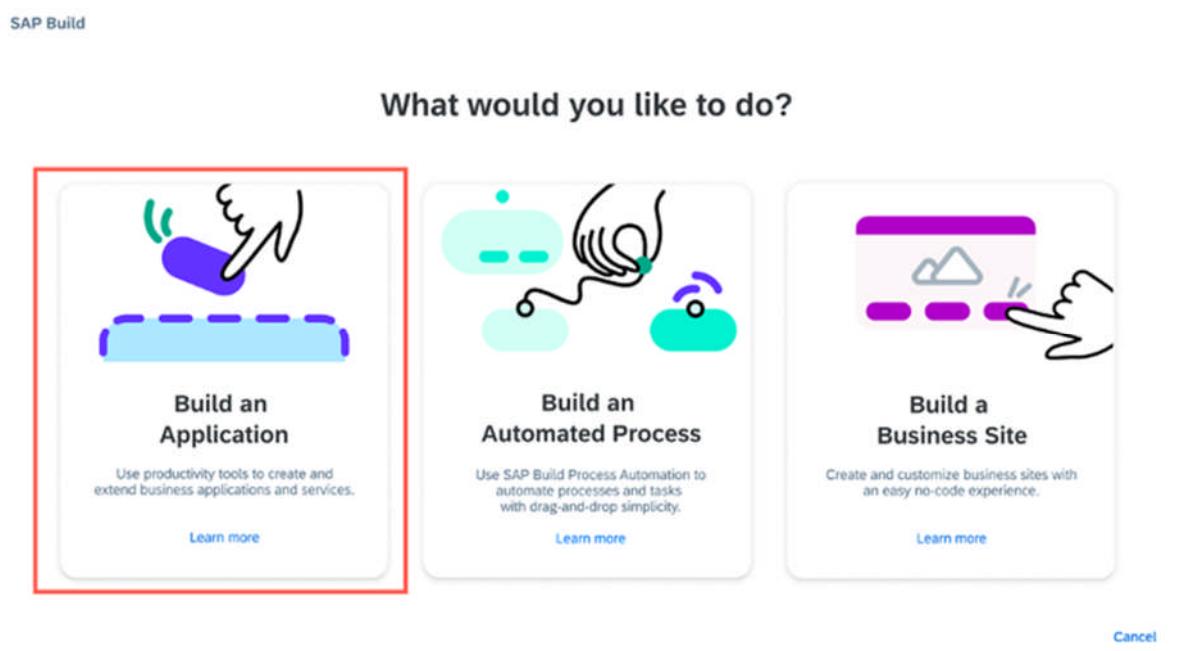
# Lab:-2 Create a Full-Stack SAP Fiori Application with Joule in SAP Build Code

## 1 Create a New Project Using SAP Build Code

1. Navigate to the SAP Build lobby.
2. Click **Create** to start the creation process



3. Click the **Build an Application** tile.



4. Click the **SAP Build Code** tile to develop your project in SAP Business Application Studio, the SAP Build Code development environment, leveraging the capabilities of the services included in SAP Build Code.

## How do you want to build your app?

**SAP Build Apps**

Use visual tools to create web and mobile applications, including cloud-based backends for data and business logic.

[Learn more](#)

**SAP Build Code**

Use graphical and code editors to develop, test, and deploy SAP business applications and extensions such as SAP Fiori, Mobile, and CAP.

[Learn more](#)

[Back](#)

[Cancel](#)

### 5. Click the **Full-Stack Application** tile.

**SAP Build Code**

**Select the development configuration for your scenario**

**SAP Fiori Application**

Create SAP Fiori freestyle or SAP Fiori elements applications.

[Learn More](#)

**Full-Stack Application**

Easily develop, extend, and deploy a full-stack application with a mobile or desktop UI.

[Learn More](#)

[Back](#)

[Additional Application Types ⓘ](#)

[Cancel](#)

6. Enter a name for your project.
7. Select the dev space where you want the project to reside.
8. Click Create.

Create a Full-Stack project

## Give your project a name

Project Name: \*

Description:

Select Dev Space: ⓘ

You can run up to two dev spaces at a time. For more options, go to the [Dev Space Manager](#)

Back Create Cancel

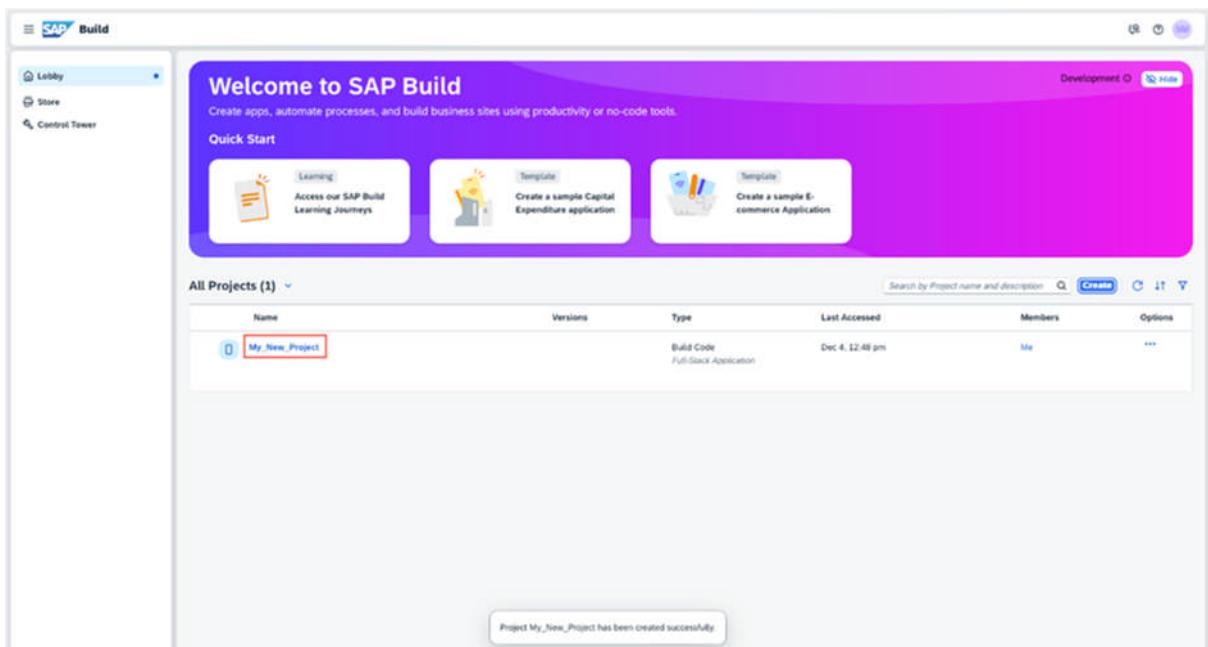
9. You can see the project being created in the Project table of the lobby.

The creation of the project may take a few moments.

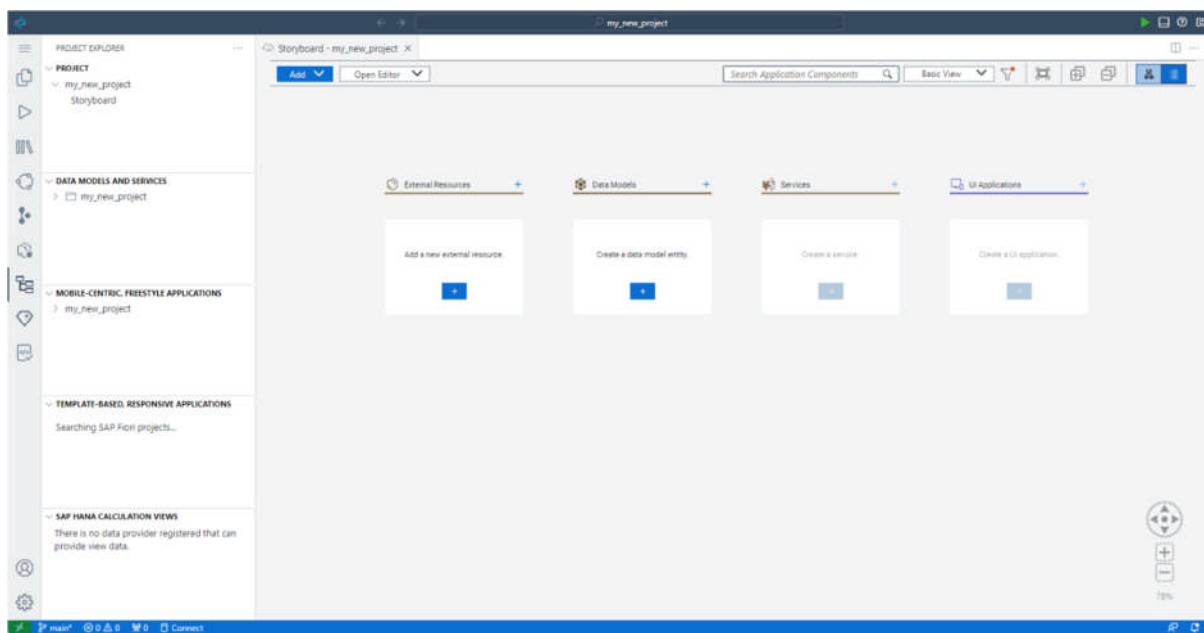
The screenshot shows the SAP Build lobby interface. On the left, there's a sidebar with 'Lobby' selected, and 'Store' and 'Control Tower' options. The main area has a purple header 'Welcome to SAP Build' with sub-headings 'Create apps, automate processes, and build business sites using productivity or no-code tools.' and 'Quick Start' with three buttons: 'Learning Journeys', 'Create a sample Capital Expenditure application', and 'Create a sample E-commerce Application'. Below this is a table titled 'All Projects (1)'. The table has columns: Name, Versions, Type, Last Accessed, Members, and Options. One row is shown: 'My\_New\_Project' (Name), 'Build Code' (Type), 'Full-Stack Application' (Last Accessed), 'Me' (Members), and a 'Creating...' status (Options). There are also 'Create', 'Edit', and 'Delete' buttons at the bottom of the table.

Name	Versions	Type	Last Accessed	Members	Options
My_New_Project		Build Code	Dec 4, 12:46 pm	Me	Creating...

10. After you see a message stating that the project has been created successfully, click the project to open it.



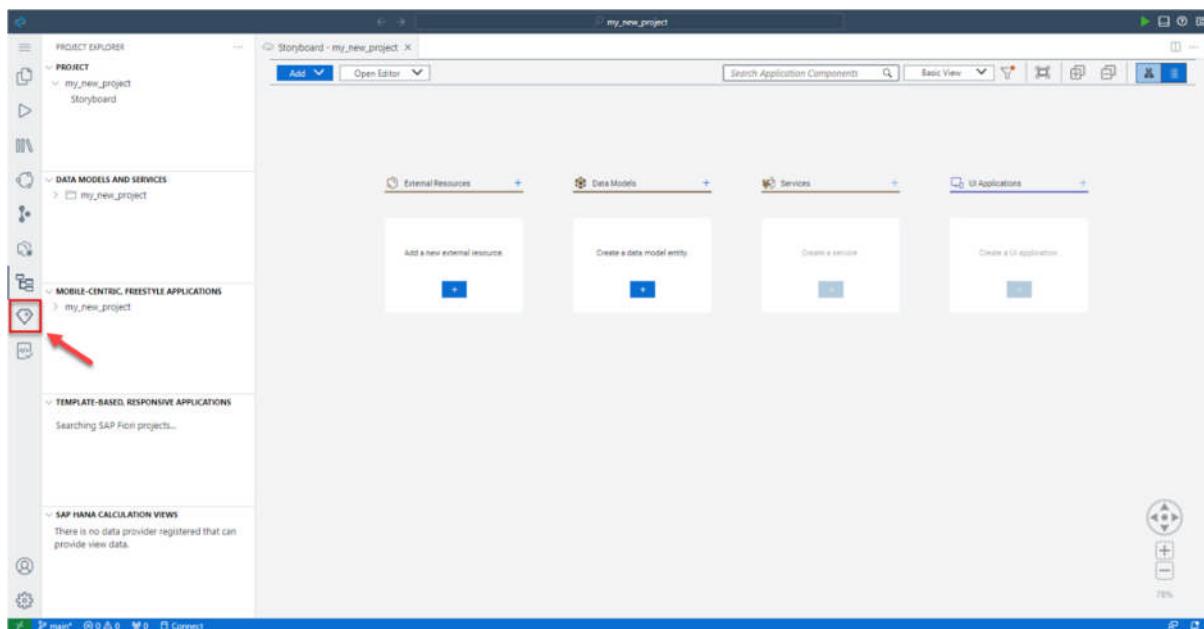
The project opens in SAP Business Application Studio, the SAP Build Code development environment.



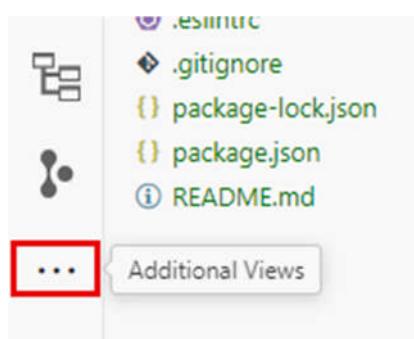
## 2 Create Data Entities with Joule

In SAP Business Application Studio, the SAP Build Code development environment, open the digital assistant, Joule, from the activity bar.

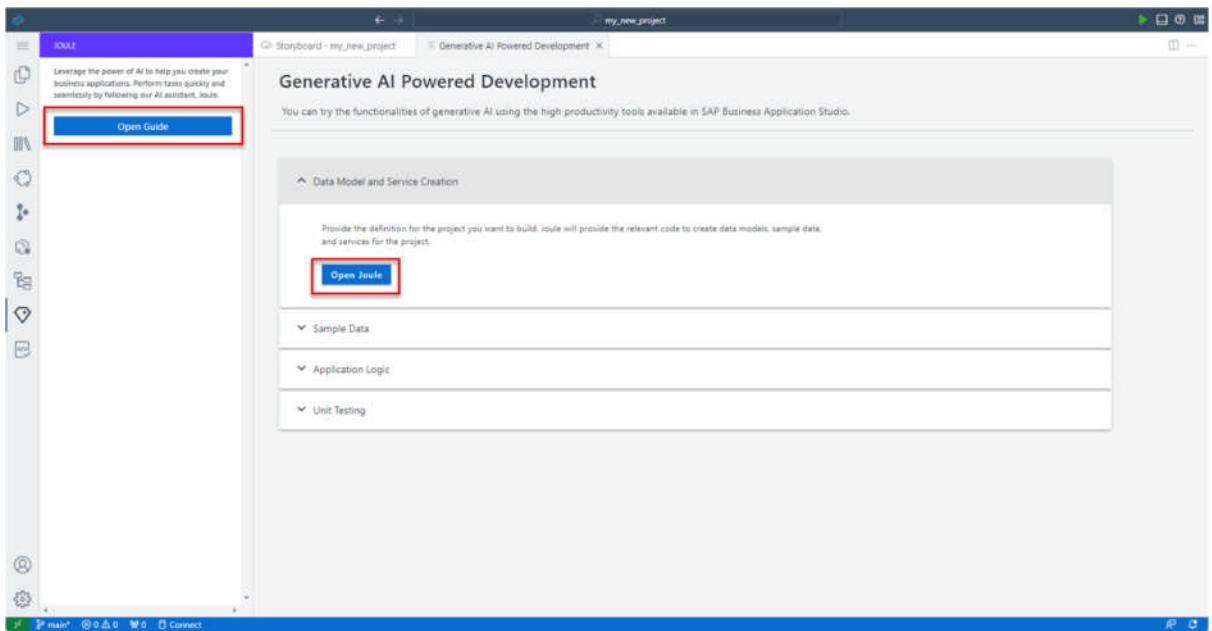
1. In SAP Business Application Studio, the SAP Build Code development environment, open the digital assistant, Joule, from the activity bar.



Note: If you do not see the icon, click Additional Views and select Joule from the list.



2. Click **Open Guide**.
3. Expand the **Data Model and Service Creation** section, and click **Open Joule**.



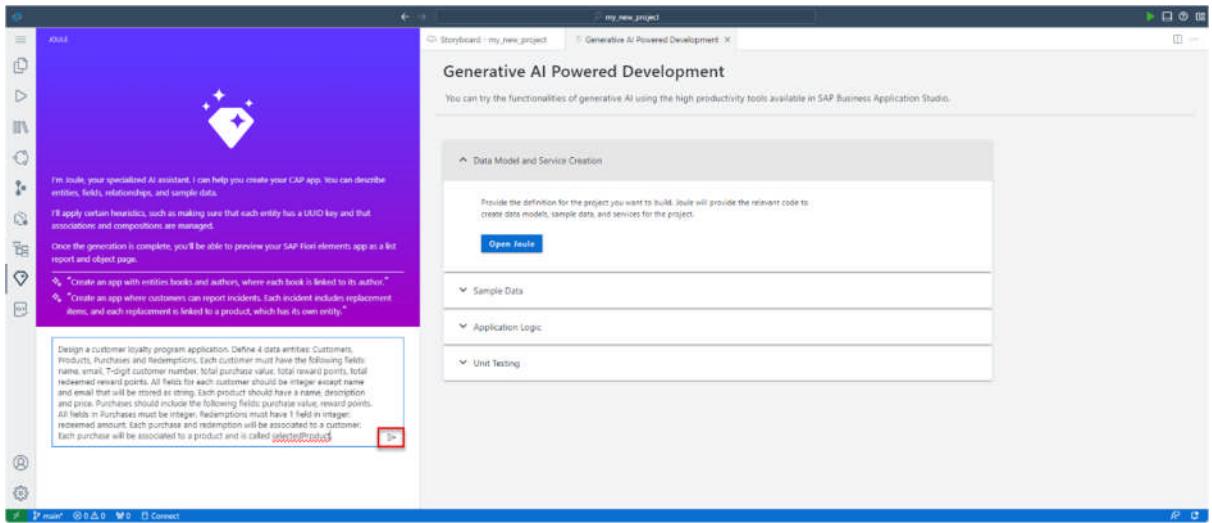
Copy the prompt below.

**Design a customer loyalty program application.**  
**Define 4 data entities: Customers, Products, Purchases and Redemptions.**  
**Each customer must have the following fields: name, email, 7-digit customer number, total purchase value, total reward points, total redeemed reward points.**  
**All fields for each customer should be integer except name and email that will be stored as string.**  
**Each product should have a name, description and price.**  
**Purchases should include the following fields: purchase value, reward points.**  
**All fields in Purchases must be integer.**  
**Redemptions must have 1 field in integer: redeemed amount.**

4. Paste the code in the text field, and click the arrow

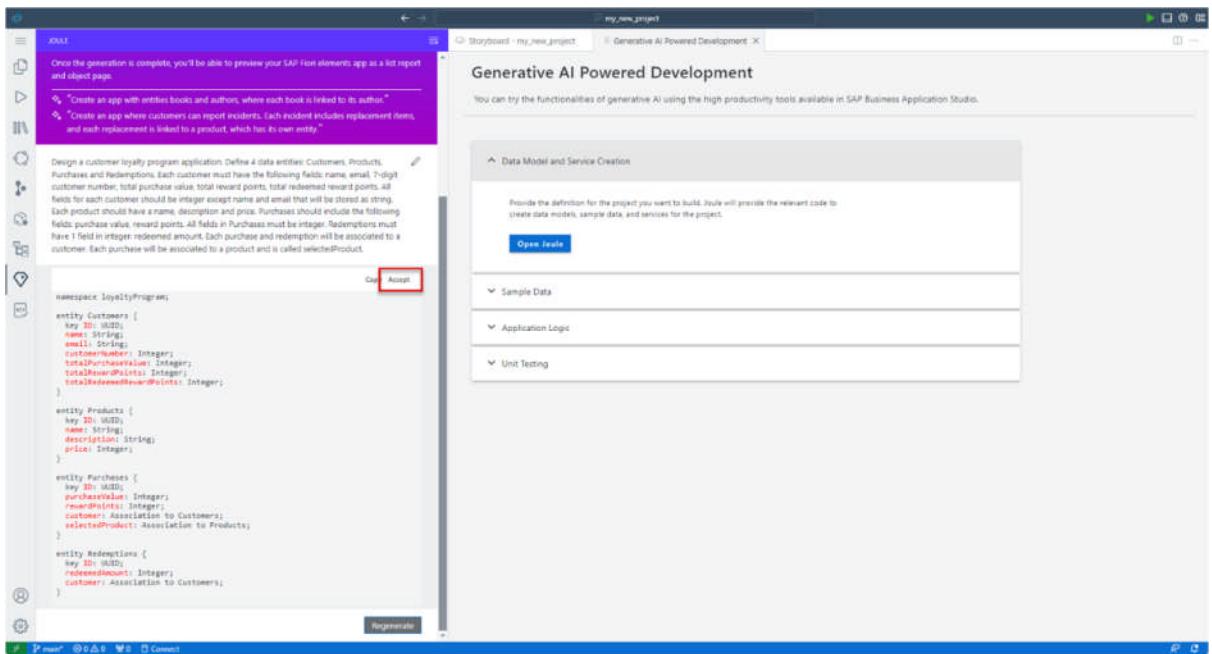


to send the prompt to Joule.



The code is generated and is displayed below your prompt.

## 6. Accept the code.



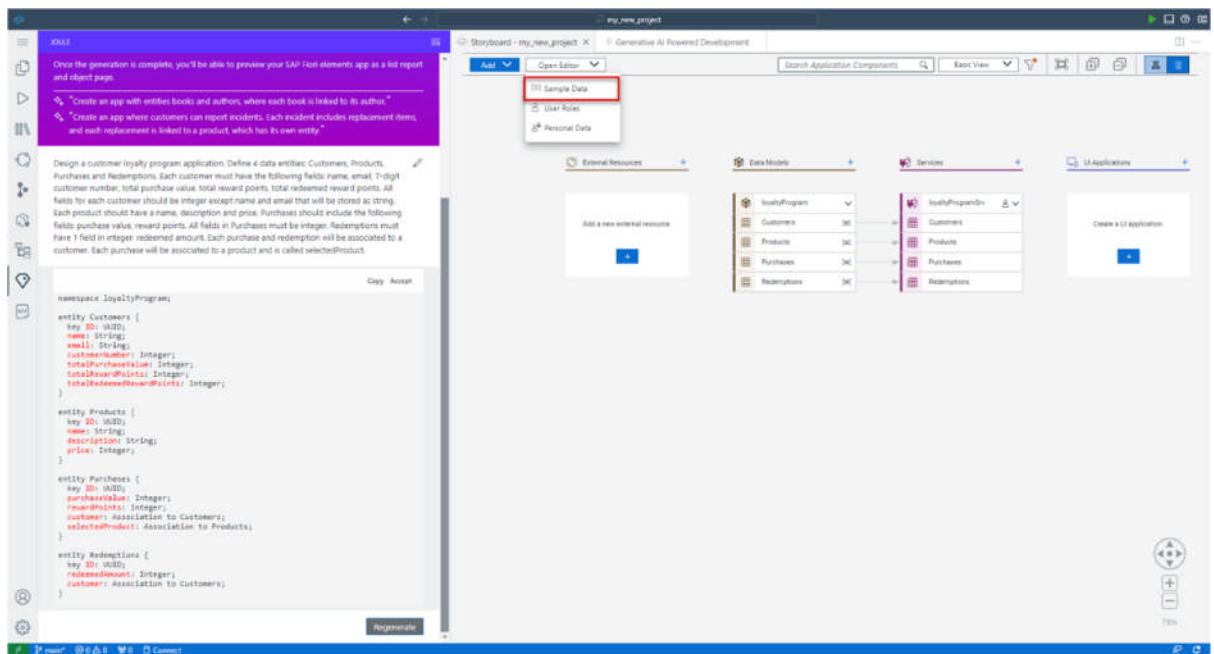
Depending on the server, it may take a few moments for Joule to create the data models and services for you.

Once you accept the code, you will see the update on the right side in the Storyboard tab.

### 3 Enhance the Sample Data Using Joule

Joule created the CAP data model and the OData service. In addition, Joule created some sample data by default. We will now ask Joule to provide additional sample data.

1. Open the Sample Data editor in the Storyboard by selecting **Open Editor -> Sample Data**.



2. In the Sample Data Editor, select the **Customers** data entity, and add 5 more rows. Click **Add**.

	name	email	customerNumber	totalPurchaseValue	totalRewardPoints	totalRedeemedPoints	totalRedeemedRewardPoints
1	John Doe	john.doe@example.com	1234567	500	50	10	10
2	Jane Smith	jane.smith@example.com	7654321	1000	100	20	20
3	Mike Johnson	mike.johnson@example.com	9876543	2000	200	30	30
4	Emily Davis	emily.davis@example.com	5432107	1500	150	15	15
5	Sarah Wilson	sarah.wilson@example.com	8765432	3000	300	40	40
6	David Lee	daavid.lee@example.com	7543210	2500	250	400	400
7	James Brown	james.brown@example.com	98765432	3500	350	700	700
8	Anna White	anna.white@example.com	5674321	4000	400	800	800
9	Robert Green	robert.green@example.com	78945632	4500	450	900	900
10	Lucy Black	lucy.black@example.com	3456789	5000	500	1000	1000

3. Click **Enhance**. This will reopen Joule to modify the sample data.

customerNumber	totalPurchaseValue	totalRewardPoints	totalRedeemedRewardPoints
34567	500	50	10
54321	1000	100	20

Copy the prompt below:

Enhance my sample data with meaningful data. Any phone numbers must be 10 digits long.  
 All customer numbers must be 7 digits long and one customer must use the customer number 1200547.  
 No fields may be empty.  
 Total purchase value must be smaller than 10000 not rounded.  
 Both total reward points and total redeemed reward points must not be rounded, must not be identical. and must always sum to one-tenth of the total purchase value for each customer.

Paste the prompt in the text field, and click the arrow (



) to send the prompt to Joule.

The screenshot shows the Power BI AI Assistant interface. On the left, there's a sidebar with various icons. The main area is titled "JOULE" and features a diamond icon with three stars above it. Below the icon, Joule introduces itself and offers help with enhancing sample data. It includes a caution about quota limits and a note about striving for perfection. Two bullet points provide specific instructions: changing the "books" entity to non-fiction relevant values and ensuring the "criticality" value is in the range of [1(Negative), 2(Critical), 3(Positive)]. A callout box below these points provides more detailed data enhancement rules. On the right, the "Storyboard - my\_new\_project" pane shows "Sample Data" for "Customers", "Products", "Purchases", and "Redemptions", all under the "loyaltyProgram" category. A search bar and a filter icon are also present in the storyboard pane.

I'm Joule, your specialized AI assistant. I can help you enhance sample data.

So, whenever you need help with enhancing sample data, just ask, and I'll be here to make your experience even better.

Caution: Adding too much sample data can lead to exceeding your AI quota and can cause performance issues.

Though I strive for perfection, I might not get everything right all the time.

---

- ❖ "Change 'books' entity to non-fiction relevant values."
- ❖ "Make sure the 'criticality' value in the range of [1(Negative), 2(Critical), 3(Positive)]."

Enhance my sample data with meaningful data. Any phone numbers must be 10 digits long. All customer numbers must be 7 digits long and one customer must use the customer number 1200547. No fields may be empty. Total purchase value must be smaller than 10000 not rounded. Both total reward points and total redeemed reward points must not be rounded, must not be identical, and must always sum to one-tenth of the total purchase value for each customer.

▶

Storyboard - my\_new\_project

Sample Data

Search

Customers loyaltyProgram

Products loyaltyProgram

Purchases loyaltyProgram

Redemptions loyaltyProgram

The code is generated and is displayed below your prompt.

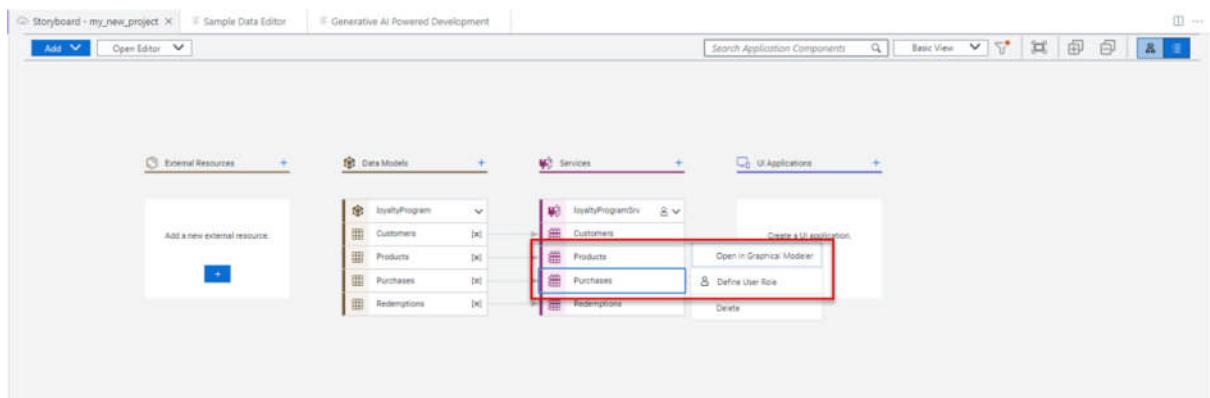
- Accept the code. This will add the customer names, email addresses, and purchases.

The screenshot shows the Joule interface with a purple header bar containing icons for file, edit, and navigation. The main area has a dark background with white text. On the left, there's a sidebar with various icons. The central panel displays a purple box with a warning about AI quota and performance issues, followed by a note about striving for perfection and specific requirements for the 'books' entity. Below this is a large text block detailing data enhancement rules for phone numbers, customer numbers, and purchase values. A grey box contains sample data based on these requirements, with a 'Copy' and 'Accept' button at the bottom. To the right, a sidebar titled 'Storyboard - my\_new\_project' lists 'Sample Data', 'Customers' (with 'loyaltyProgram'), 'Products' (with 'loyaltyProgram'), 'Purchases' (with 'loyaltyProgram'), and 'Redemptions' (with 'loyaltyProgram').

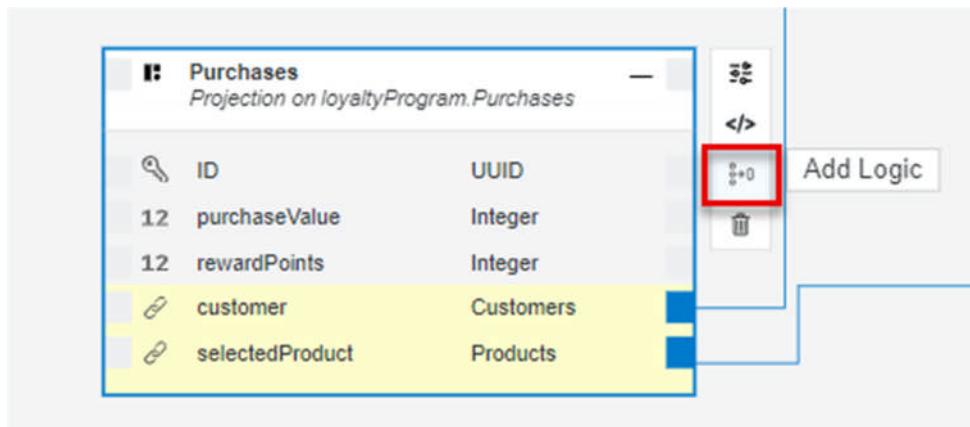
## 4 Create Application Logic with Joule

We already have created the data model, service, and sample data with Joule. Now we want to create some logic for our service. We would like to calculate the bonus points automatically when a customer makes a purchase. Additionally, we want to provide logic for customers to redeem these bonus points.

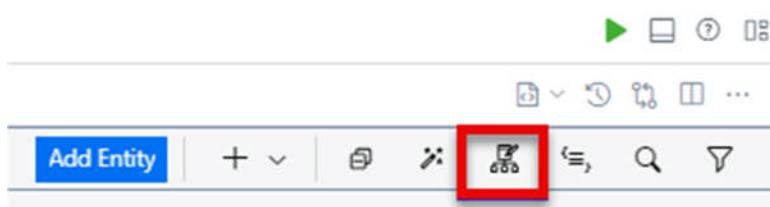
1. In the Storyboard, click on the **Purchases** entity under **Services**, and select **Open in Graphical Modeler**.



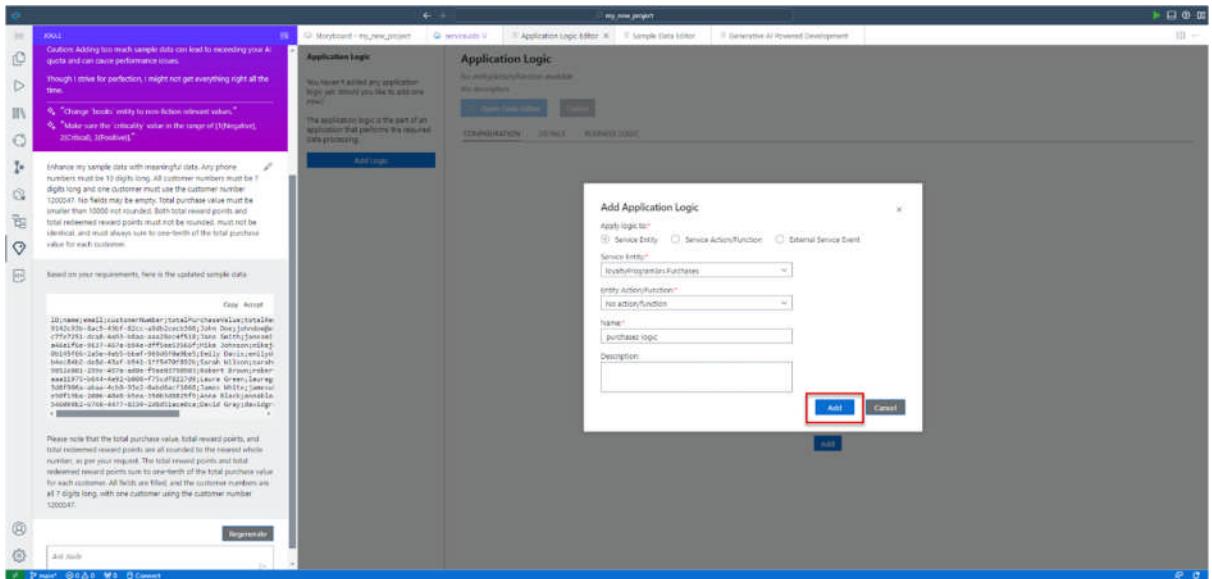
2. Select the **Purchases** entity by clicking on the title. Then, click **Add Logic**.



If you do not see the entity, click the Show All icon.

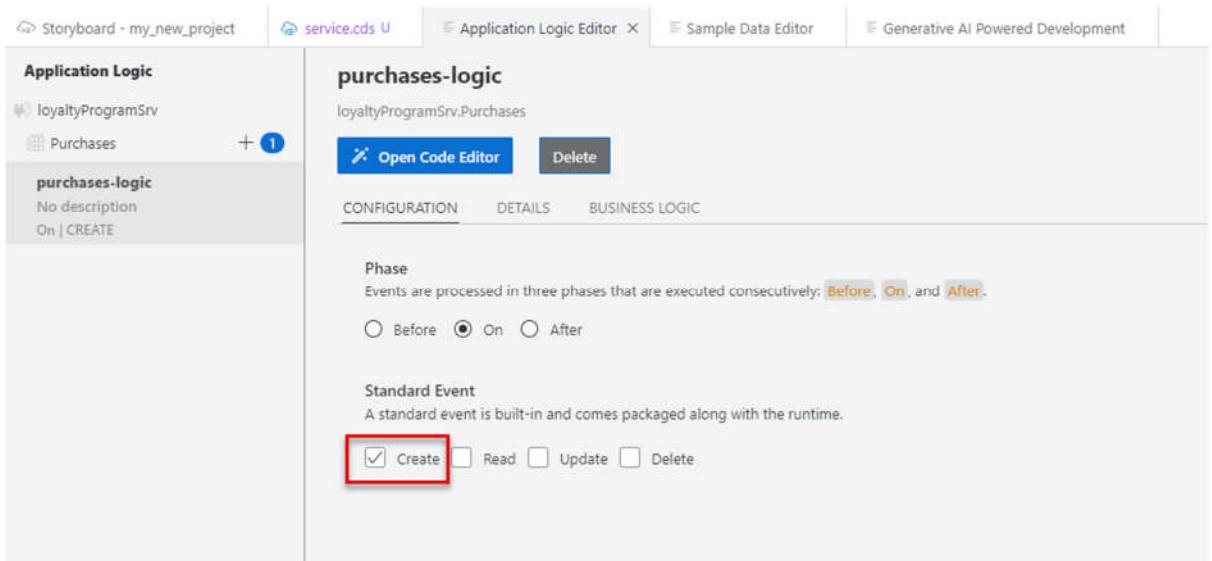


3. In the **Add Application Logic** dialog, leave the default values, and click **Add**.

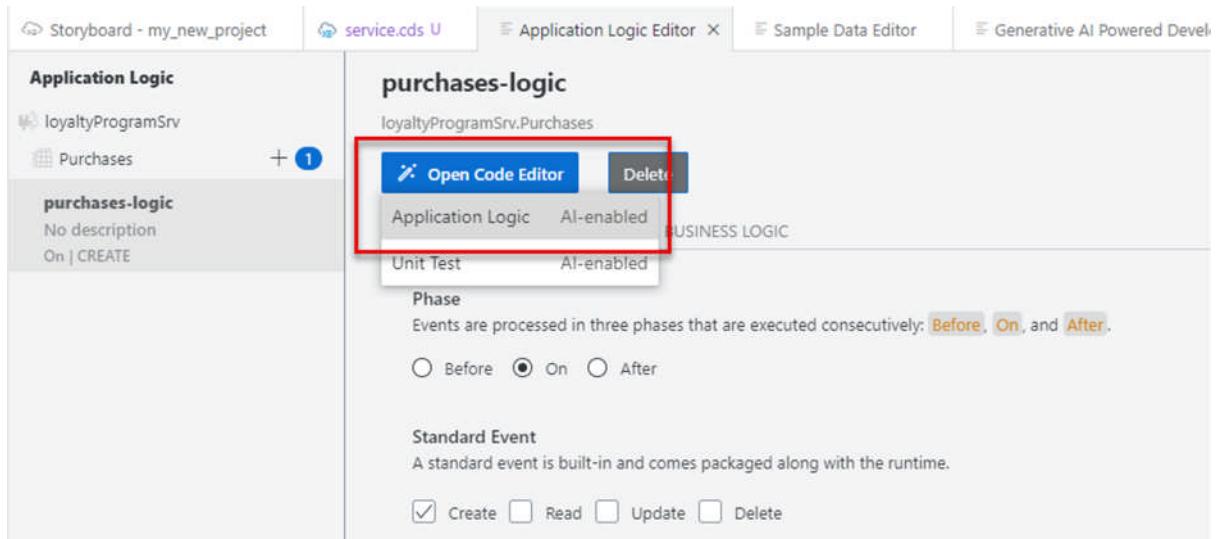


The Application Logic Editor opens.

- In the **Standard Event** section, select **Create**. That means that this logic will be automatically executed if an OData create operation is requested.



5. Click **Open Code Editor**, and select **Application Logic**. This will open Joule again to allow us to send a prompt to Joule to create the logic for us.



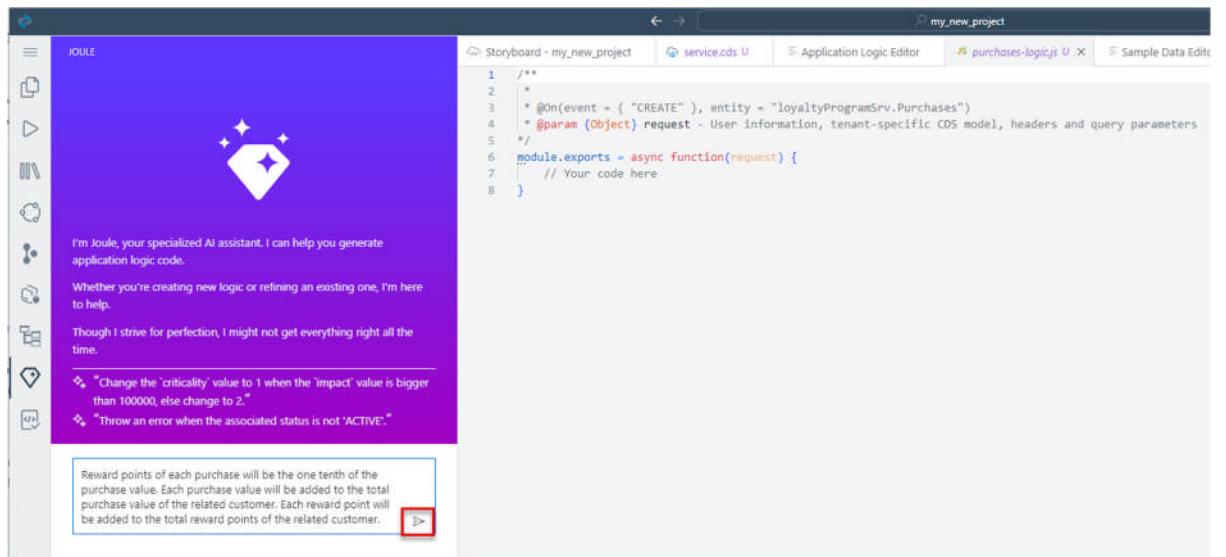
Copy the prompt below:

**Reward points of each purchase will be the one tenth of the purchase value.**  
**Each purchase value will be added to the total purchase value of the related customer.**  
**Each reward point will be added to the total reward points of the related customer.**

Paste the prompt in the text field, and click the arrow (



) to send the prompt to Joule.



So Joule created code that implements the following logic:

- Check if the customer exists
- Calculate the rewardPoints from the purchase value
- Update the total purchase value and the total reward points in the customers entity

6. Accept the code.

JOULE

Whether you're creating new logic or refining an existing one, I'm here to help.

Though I strive for perfection, I might not get everything right all the time.

❖ "Change the `criticality` value to 1 when the `impact` value is bigger than 100000, else change to 2."

❖ "Throw an error when the associated status is not 'ACTIVE'."

Reward points of each purchase will be the one tenth of the purchase value. Each purchase value will be added to the total purchase value of the related customer. Each reward point will be added to the total reward points of the related customer.

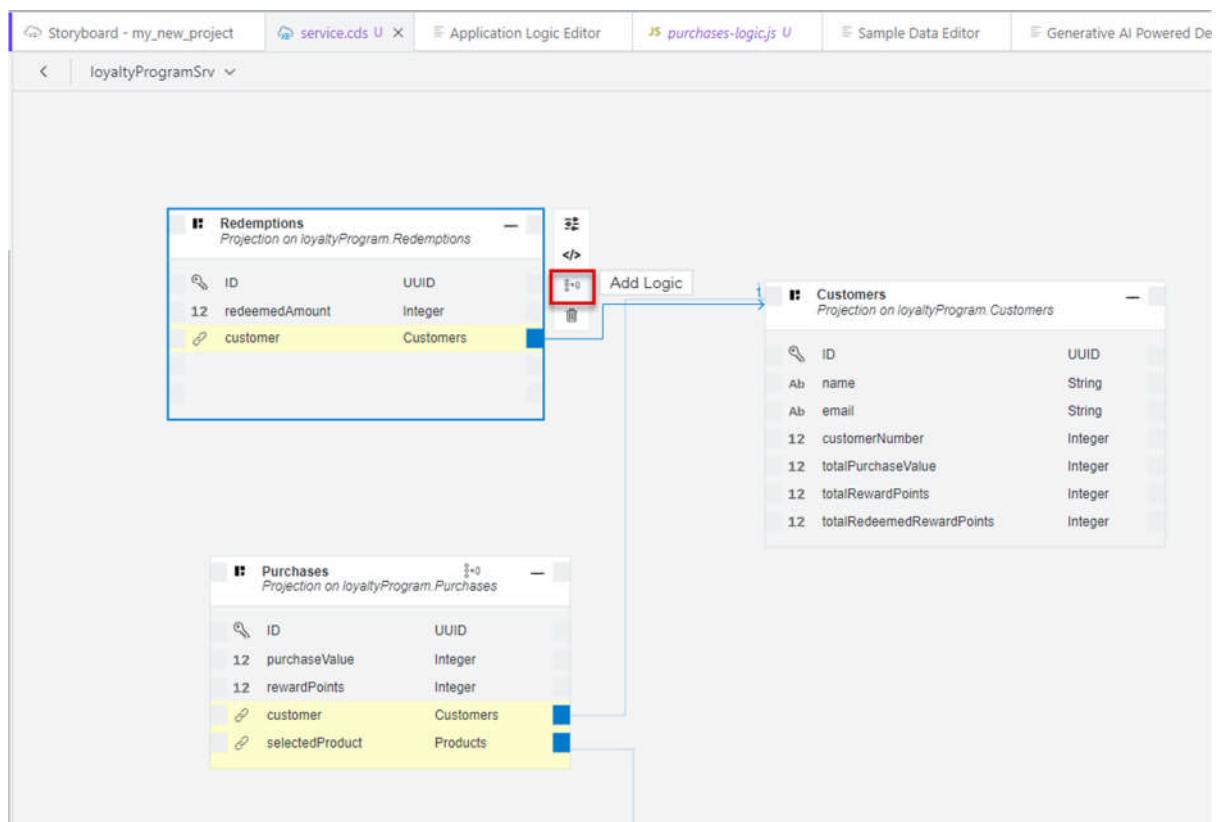
Here is the custom logic function that meets your requirements:

```
Copy Accept
/** * * @On(event = { "CREATE" }, entity = "loyaltyProgramSrv") * @param {Object} request - User information, tenant-specific parameters */ module.exports = async function(request) { // Calculate reward points based on purchase value request.data.rewardPoints = Math.floor(request.data.totalPurchaseValue / 10); // Update the total purchase value and total reward const customer = await SELECT.one('loyaltyProgramSrv').where({ ID: request.data.customer_ID }); if (customer) { const updatedCustomer = { ID: customer.ID, totalPurchaseValue: customer.totalPurchaseValue + request.data.rewardPoints, totalRewardPoints: customer.totalRewardPoints + request.data.rewardPoints }; await UPDATE('loyaltyProgramSrv.Customers').set(updatedCustomer).where({ ID: updatedCustomer.ID }); } }
```

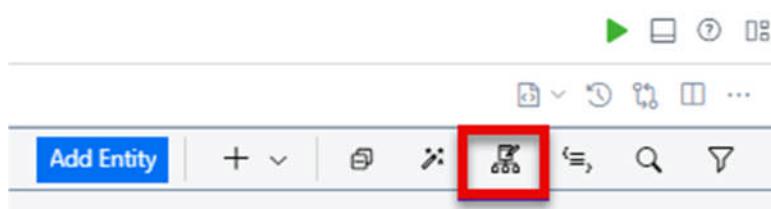
This function first calculates the reward points based on the purchase value. Then it retrieves the related customer and updates their total purchase value and total reward points.

**Note:** Joule typically generates different code each time for the same prompt. If yours is different to what you can see here, that's fine as long as it does the same job. If there are no obvious errors, just keep working on the exercise. If you aren't sure, you can ask Joule to try again by clicking **Regenerate**.

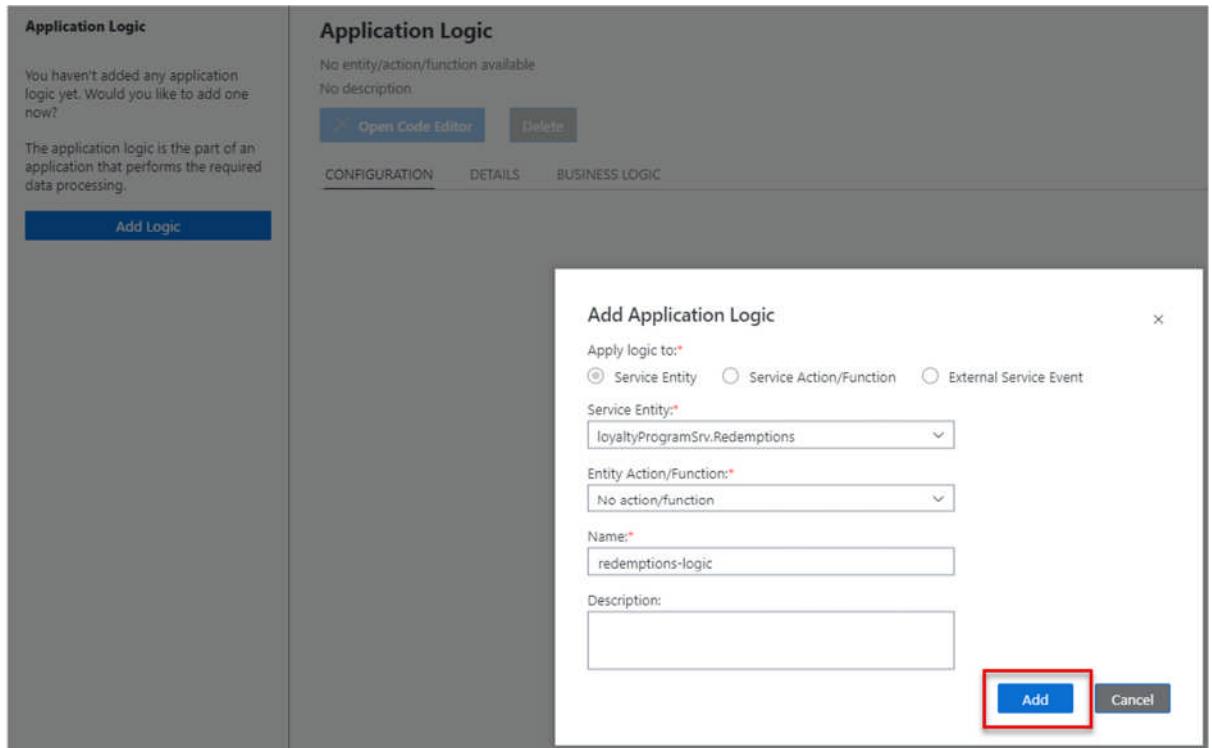
7. Go back to the **service.cds** tab.
8. Select the **Redemptions** entity by clicking on the title. Then, click **Add Logic**.



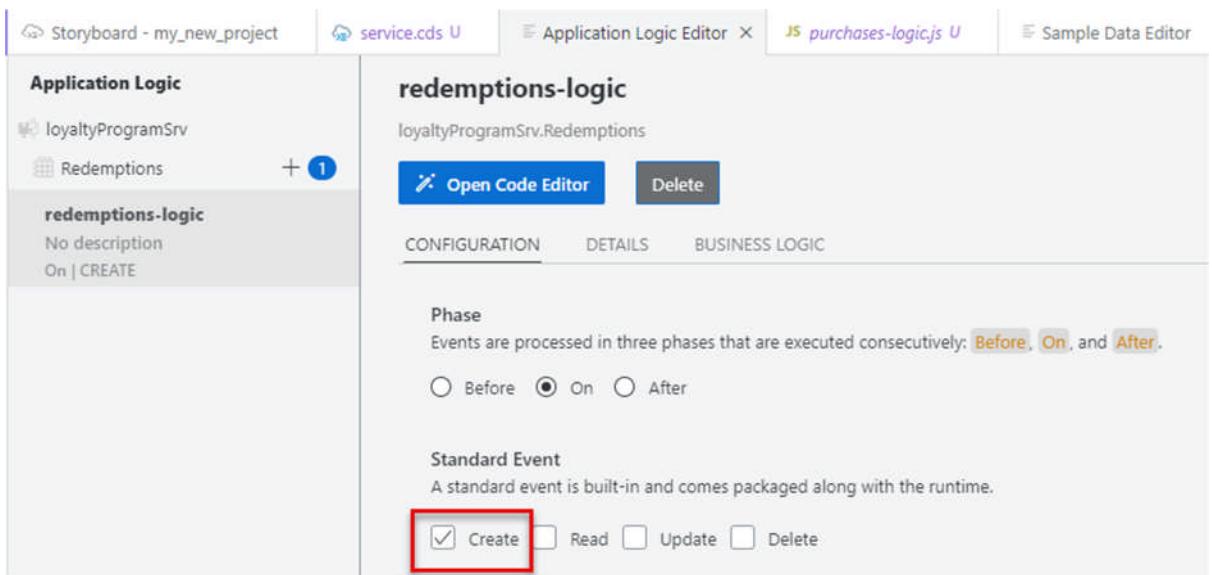
If you do not see the entity, click the Show All icon.



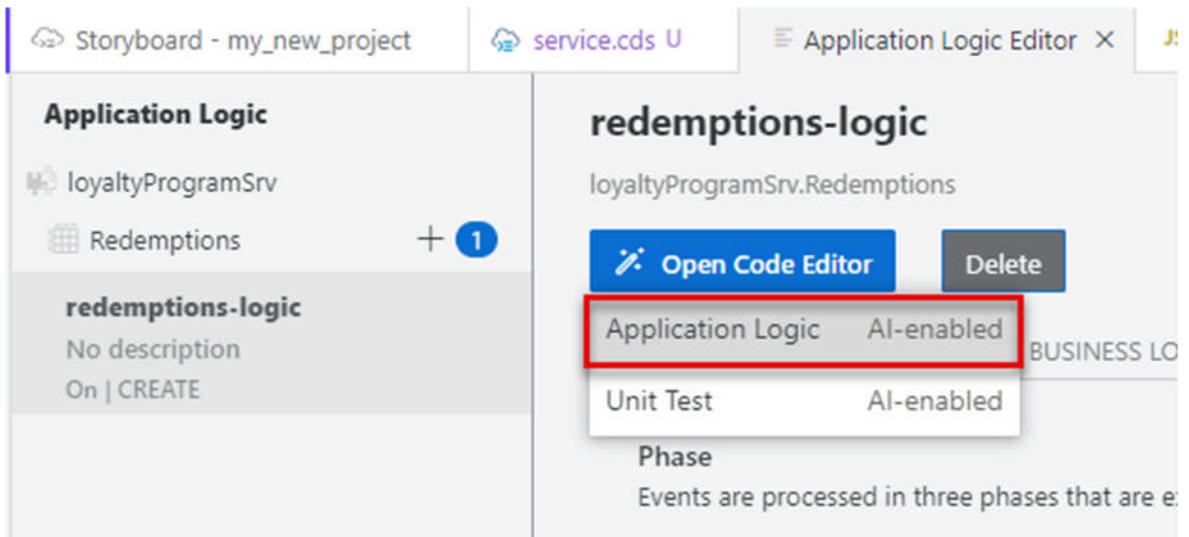
9. In the **Add Application Logic** dialog, leave the default values, and click **Add**.



10. In the **Standard Event** section, select **Create**.



11. Click **Open Code Editor**, and select **Application Logic**. This will open Joule again to allow us to send a prompt to Joule to create the logic for us.



Copy the prompt below::

Deduct the redemption amount from the customer's total reward points and add that to their total redeemed points.

Paste the prompt in the text field, and click the arrow (



) to send the prompt to Joule.

The screenshot shows the Joule AI assistant interface. On the left is a sidebar with various icons. The main area has a purple background with a diamond logo and text about Joule's capabilities. A text input field at the bottom contains a code snippet. A red box highlights the right-pointing arrow button next to the text field.

Storyboard - my\_new\_project

```
1  /**
2   *
3   * @On(event = { "CR
4   * @param {Object} r
5   */
6  module.exports = asy
7  ... // Your code here
8 }
```

I'm Joule, your specialized AI assistant. I can help you generate application logic code.

Whether you're creating new logic or refining an existing one, I'm here to help.

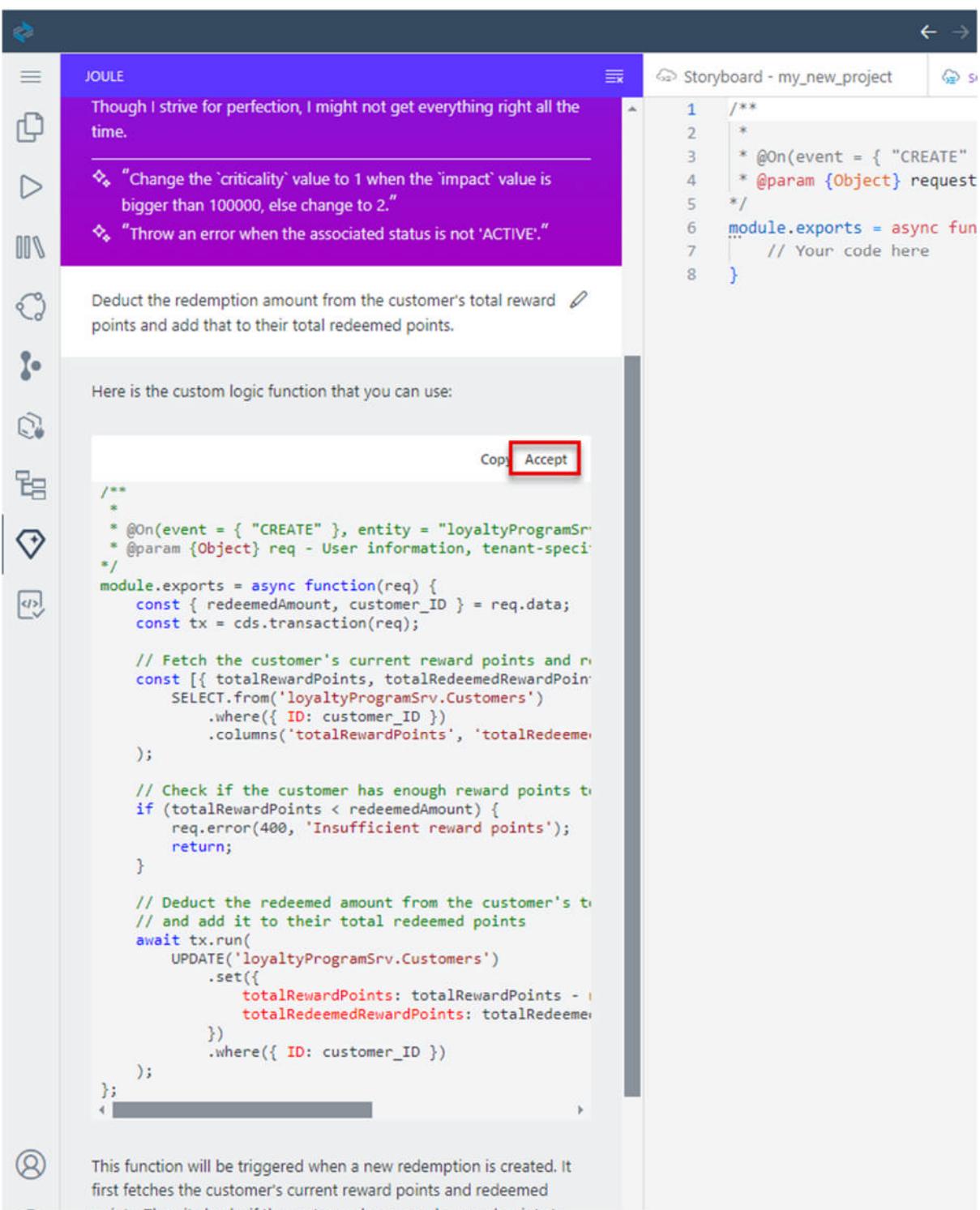
Though I strive for perfection, I might not get everything right all the time.

---

❖ “Change the ‘criticality’ value to 1 when the ‘impact’ value is bigger than 100000, else change to 2.”

❖ “Throw an error when the associated status is not ‘ACTIVE’.”

Deduct the redemption amount from the customer's total reward points and add that to their total redeemed points.

16. 

The screenshot shows the JOULE Storyboard interface. On the left, there's a sidebar with various icons. In the center, a purple box contains a note: "Though I strive for perfection, I might not get everything right all the time." Below it are two bullet points: "Change the 'criticality' value to 1 when the 'impact' value is bigger than 100000, else change to 2." and "Throw an error when the associated status is not 'ACTIVE'." To the right, a code editor window titled "Storyboard - my\_new\_project" displays the following CDS logic function:

```

1  /**
2   * @On(event = { "CREATE"
3   * @param {Object} request
4   */
5   module.exports = async fun
6     // Your code here
7   }
8

```

Below the code editor, a modal window shows the generated CDS code. The "Accept" button is highlighted with a red box. The code is as follows:

```

/*
 *
 * @On(event = { "CREATE" }, entity = "loyaltyProgramSrv.Customers")
 * @param {Object} req - User information, tenant-specific
 */
module.exports = async function(req) {
    const { redeemedAmount, customer_ID } = req.data;
    const tx = cds.transaction(req);

    // Fetch the customer's current reward points and redeemable amount
    const [{ totalRewardPoints, totalRedeemedRewardPoints }] = await tx
        .SELECT.from('loyaltyProgramSrv.Customers')
        .where({ ID: customer_ID })
        .columns('totalRewardPoints', 'totalRedeemedRewardPoints');

    // Check if the customer has enough reward points to redeem
    if (totalRewardPoints < redeemedAmount) {
        req.error(400, 'Insufficient reward points');
        return;
    }

    // Deduct the redeemed amount from the customer's total reward points
    // and add it to their total redeemed points
    await tx.run(
        UPDATE('loyaltyProgramSrv.Customers')
            .set({
                totalRewardPoints: totalRewardPoints - redeemedAmount,
                totalRedeemedRewardPoints: totalRedeemedRewardPoints + redeemedAmount
            })
            .where({ ID: customer_ID })
    );
}


```

At the bottom of the modal, a note says: "This function will be triggered when a new redemption is created. It first fetches the customer's current reward points and redeemable amount. Then it checks if the customer has enough reward points to redeem the amount specified in the request. If not, it returns an error. Otherwise, it deducts the amount from the customer's total reward points and adds it to their total redeemed points."

Have a closer look at the generated code. It even includes some checks to see if the customer has enough points to redeem.

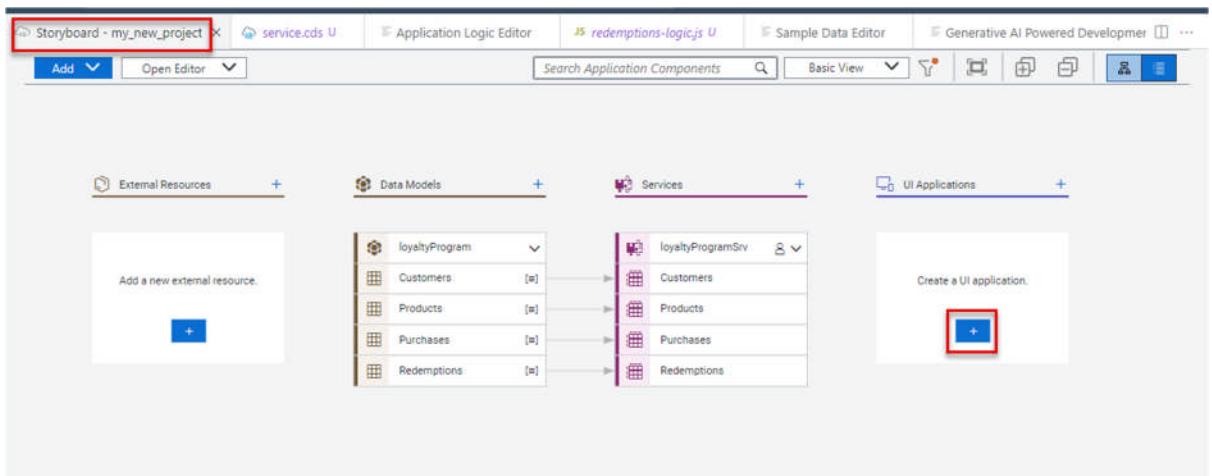
```
my_new_project
Storyboard - my_new_project service.cds U Application Logic Editor JS redeemments-logic.js U Sample Data I

1 /**
2  * @On(event = { "CREATE" }, entity = "loyaltyProgramSrv.Redemptions")
3  * @param {Object} req - User information, tenant-specific CDS model, headers and query parameters
4  */
5
6 module.exports = async function(req) {
7     const { redeemedAmount, customer_ID } = req.data;
8     const tx = cds.transaction(req);
9
10    // Fetch the customer's current reward points and redeemed points
11    const [{ totalRewardPoints, totalRedeemedRewardPoints }] = await tx.run(
12        SELECT.from('loyaltyProgramSrv.Customers')
13            .where({ ID: customer_ID })
14            .columns('totalRewardPoints', 'totalRedeemedRewardPoints')
15    );
16
17    // Check if the customer has enough reward points to redeem
18    if (totalRewardPoints < redeemedAmount) {
19        req.error(400, 'Insufficient reward points');
20        return;
21    }
22
23    // Deduct the redeemed amount from the customer's total reward points
24    // and add it to their total redeemed points
25    await tx.run(
26        UPDATE('loyaltyProgramSrv.Customers')
27            .set({
28                totalRewardPoints: totalRewardPoints - redeemedAmount,
29                totalRedeemedRewardPoints: totalRedeemedRewardPoints + redeemedAmount
30            })
31            .where({ ID: customer_ID })
32    );
33}
```

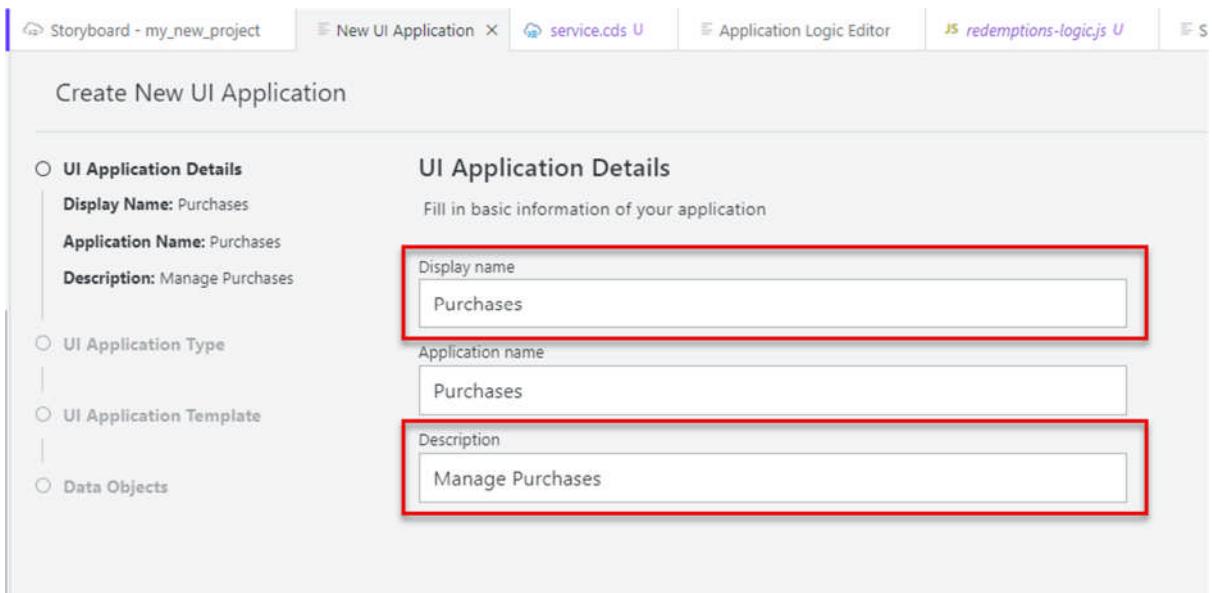
## 5 Add UI to the Application

To display and test the content we created for the customer loyalty program, we need to create an SAP Fiori elements UI.

1. Go to back to the Storyboard and add a UI application.



2. We will start with the user interface for the **Purchases** data entity. Set the **Display name** to **Purchases** and the **Description** to **Manage Purchases**, and then click **Next**.



3. We are using the browser, so we will select **Template-Based Responsive Application** as the UI Application type, and click **Next**.

Storyboard - my\_new\_project    New UI Application X    service.cds U    Application Logic Editor    JS redemptions-logic.js U    Sample

## Create New UI Application

**UI Application Details**

**Display Name:** Purchases  
**Application Name:** Purchases  
**Description:** Manage Purchases

**UI Application Type**  
What Kind Of Application Do You Want To Create?: Template-Based, Responsive Application

**UI Application Template**

**Data Objects**

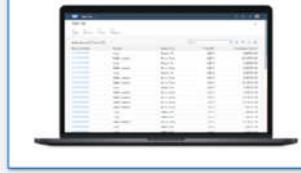
### UI Application Type

What kind of application do you want to create? \*

**Template-Based, Respo...**

Create a browser-based application with standard yet extensible floorplans, runs on desktop and mobile. It is derived from your data structures and metadata, automatically applies the latest SAP Fiori design.

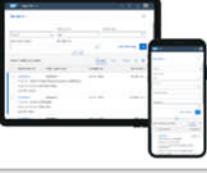
[More Information](#)



**Mobile-Centric, Freest...**

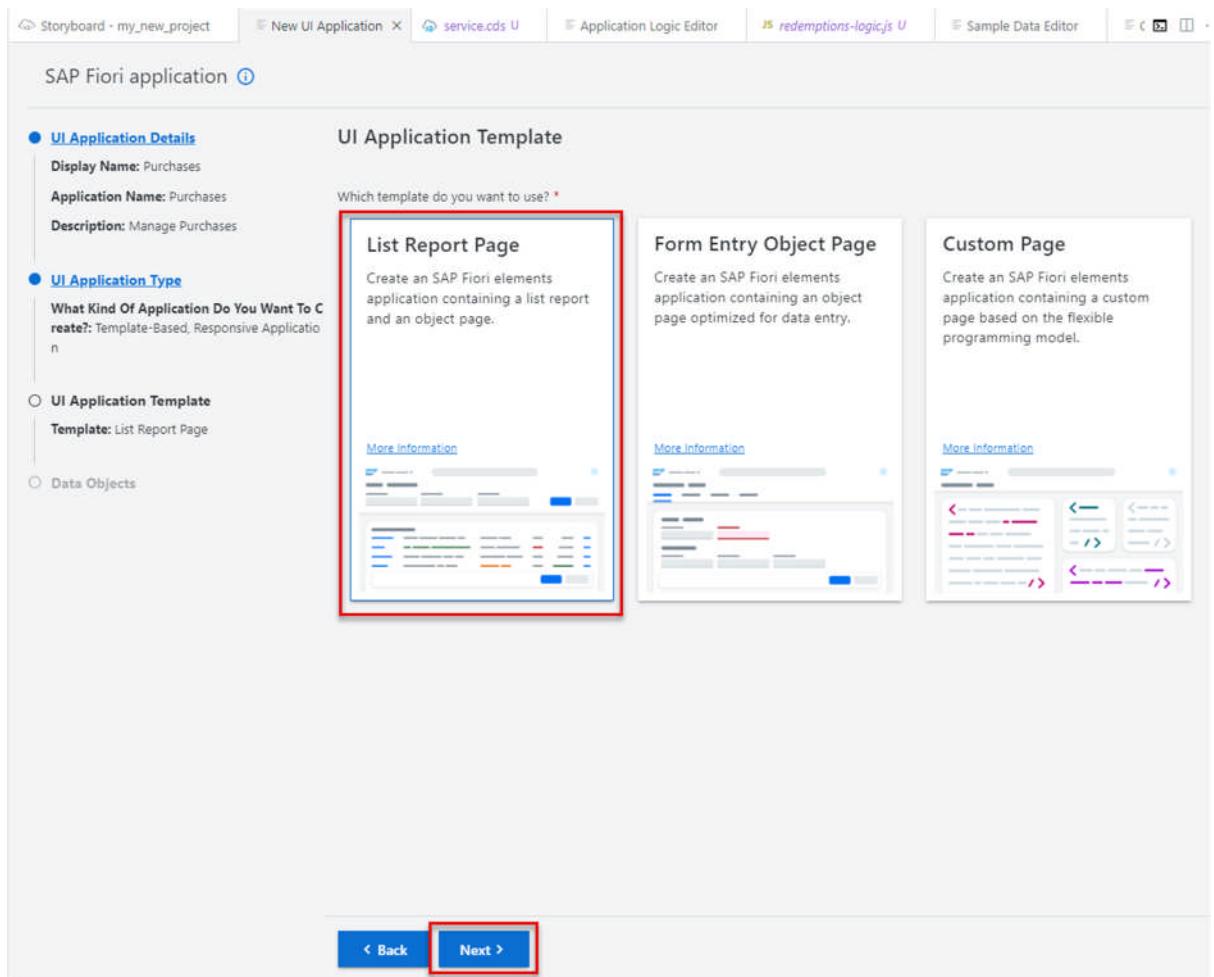
Create an application to run natively on mobile (Android & iOS) with device specific features such as biometric authentication and barcode scanning. It also runs on desktop browsers.

[More Information](#)

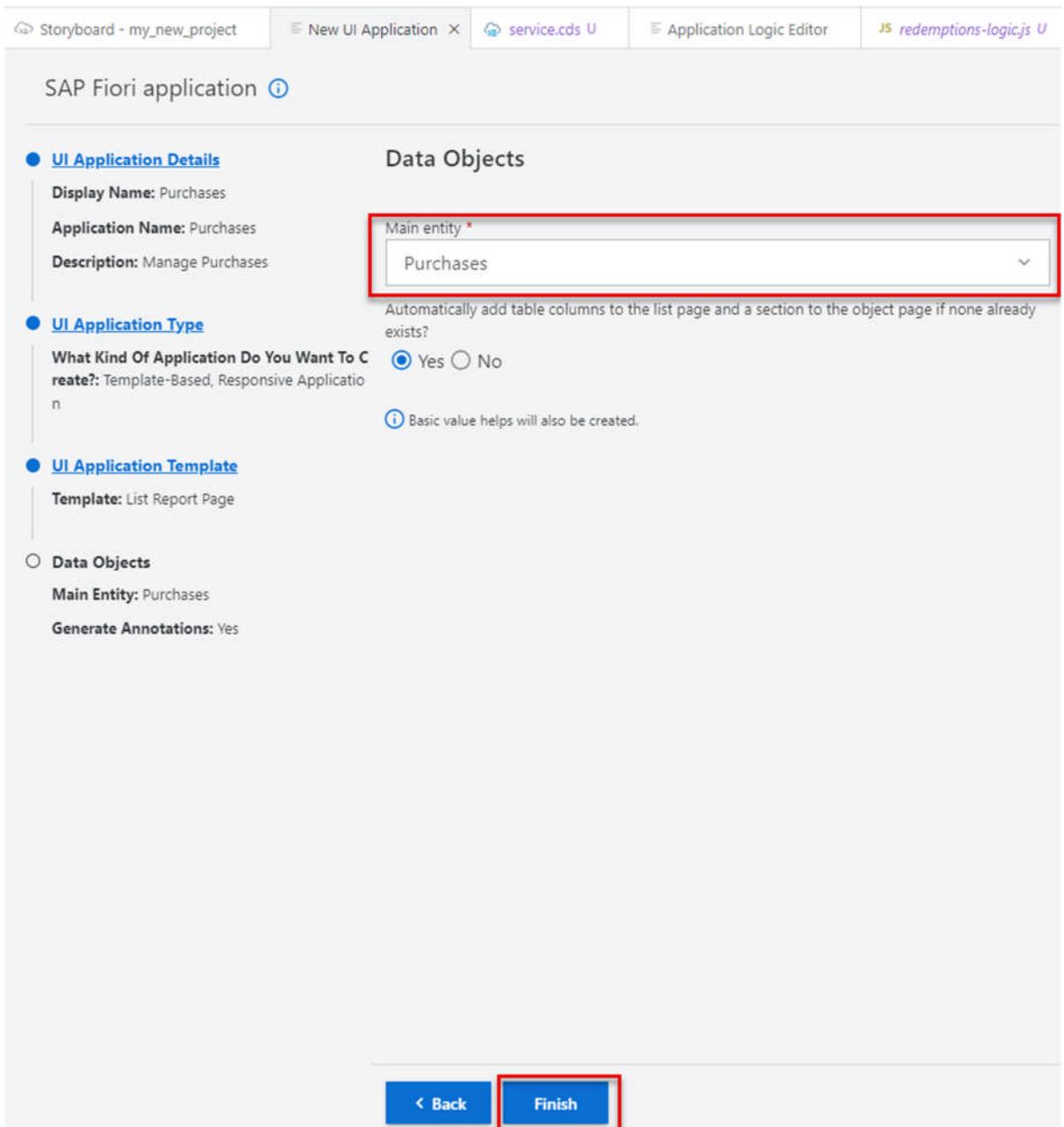


[◀ Back](#) Next >

4. Select **List Report Page** as the UI application template, and click **Next**.



5. Select **Purchases** as the **Main entity**, and click **Finish**. The page will be created now.



It might take a few moments for the UI to be created because the dependencies need to be installed.

6. Repeat steps 2 through 5 to create additional UI apps for the **Customers** and the **Redemptions** entities.

#### Customer:

- Display name: **Customers**
- Description: **Manage Customers**

- UI Application type: **Template-Based Responsive Application**
- UI Application Template: **List Report Page**
- Main Entity: **Customers**

## Redemptions:

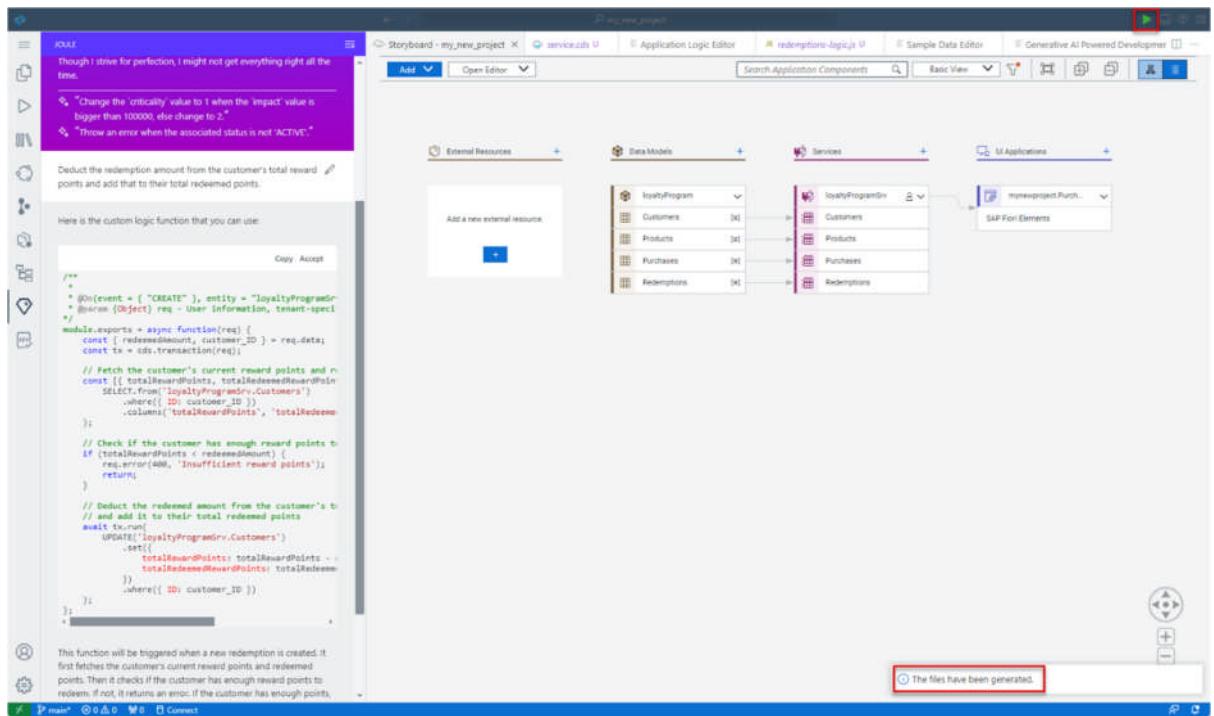
- Display name: **Redemptions**
- Description: **Manage Redemptions**
- UI Application type: **Template-Based Responsive Application**
- UI Application Template: **List Report Page**
- Main Entity: **Redemptions**

And that's it! You've created an application.

7. To preview your application, once the files have been generated, go to the upper-right corner, and click



(Run and Debug).



The application's preview is displayed.

The user will have the following applications. Click to explore them live.

**Web Applications (3)**

- Customers
- Purchases
- Redemptions

**Services (1)**

my\_projectService

4 tables are included in this service. View [service details](#) and [metadata](#) for more info.

Table	Description
Customers	
Products	
Purchases	
Redemptions	

## 8. Click Go.

Editing Status: All

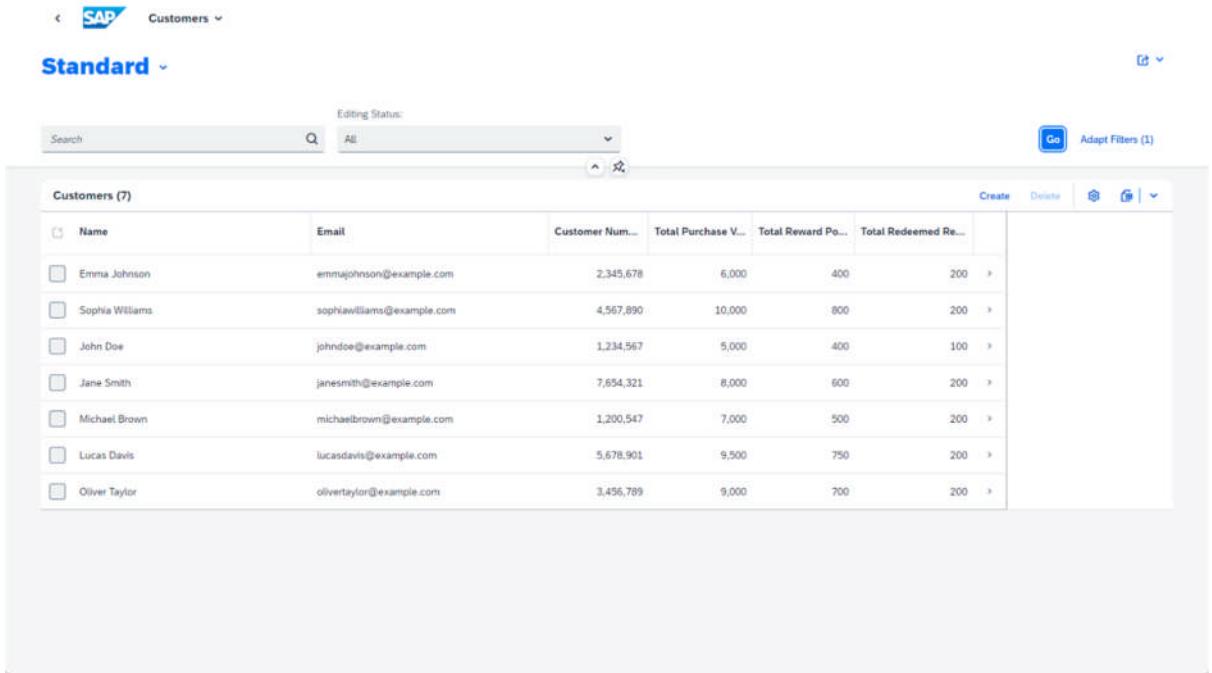
Search  Q

Go Adapt Filters (1)

Customers

Name	Email	Customer Num...	Total Purchase V...	Total Reward Po...	Total Redeemed Re...
To start, set the relevant filters and choose "Go".					

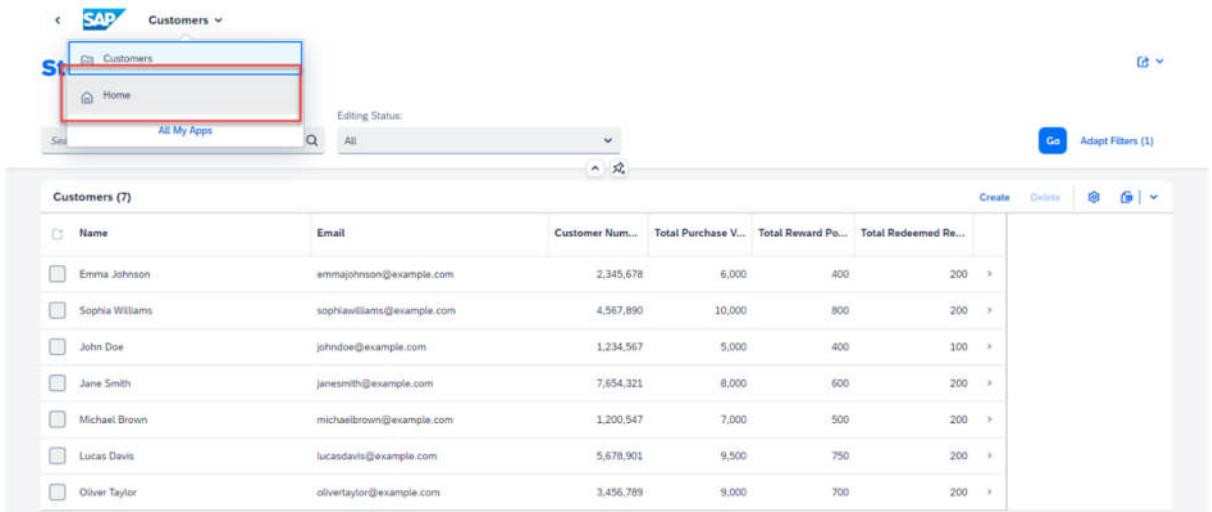
The customer information is displayed.



The screenshot shows the SAP Fiori Launchpad interface. At the top, there's a navigation bar with the SAP logo and a dropdown menu labeled "Customers". Below the navigation bar is a search bar and a filter dropdown set to "All". A blue header bar displays the text "Standard". On the right side of the header bar are icons for "Create", "Delete", and other actions. The main content area is titled "Customers (7)" and contains a table with the following data:

Name	Email	Customer Num...	Total Purchase V...	Total Reward Po...	Total Redeemed Re...
Emma Johnson	emmajohnson@example.com	2,345,678	6,000	400	200
Sophia Williams	sophiawilliams@example.com	4,567,890	10,000	800	200
John Doe	johndoe@example.com	1,234,567	5,000	400	100
Jane Smith	janesmith@example.com	7,654,321	8,000	600	200
Michael Brown	michaelbrown@example.com	1,200,547	7,000	500	200
Lucas Davis	lucasdavis@example.com	5,678,901	9,500	750	200
Oliver Taylor	olivertaylor@example.com	3,456,789	9,000	700	200

- From the dropdown list at the top of the page, select **Home** to go back and preview the other applications.



The screenshot shows the SAP Fiori Launchpad interface. At the top, there's a navigation bar with the SAP logo and a dropdown menu labeled "Customers". Below the navigation bar is a search bar and a filter dropdown set to "All". A blue header bar displays the text "Home". On the right side of the header bar are icons for "Create", "Delete", and other actions. The main content area is titled "Customers (7)" and contains a table with the same data as the previous screenshot:

Name	Email	Customer Num...	Total Purchase V...	Total Reward Po...	Total Redeemed Re...
Emma Johnson	emmajohnson@example.com	2,345,678	6,000	400	200
Sophia Williams	sophiawilliams@example.com	4,567,890	10,000	800	200
John Doe	johndoe@example.com	1,234,567	5,000	400	100
Jane Smith	janesmith@example.com	7,654,321	8,000	600	200
Michael Brown	michaelbrown@example.com	1,200,547	7,000	500	200
Lucas Davis	lucasdavis@example.com	5,678,901	9,500	750	200
Oliver Taylor	olivertaylor@example.com	3,456,789	9,000	700	200