

PowerApps and Azure

Hands-on Lab

Table of Contents

Lab Overview & Objective.....	1
Objective:	1
Overview	1
Environment	1
Main components of the lab	2
Exercise 1 – Build the API App.....	3
Exercise 2 – Deploy to Azure	6
Exercise 3 – Create custom connector	10
Exercise 4 – Create canvas app	18
Copyright.....	28

Lab Overview & Objective

Objective:

This lab has two key objectives:

- To demonstrate how PowerApps platform can help unlock the potential of untapped assets within an Enterprise (legacy APIs, data sources, processes) with a low-code / no code approach.
- To demonstrate how the above pattern can, in turn, engender Azure consumption in the enterprise.

Overview

Most enterprises have a majority of their core business data trapped in several silos (legacy databases & apps - Systems of Records). It is essential for digital transformation to break those data silos and make the information trapped in those systems available to employees and business teams as needed.

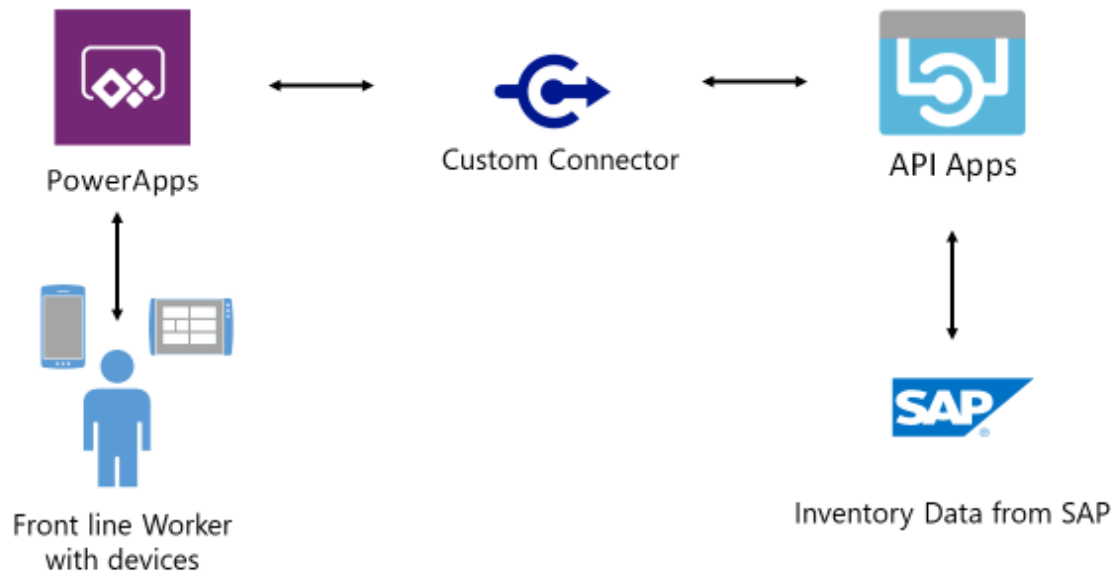
It is also important to modernize the user experience of the enterprise apps in order to drive productivity. PowerApps along with Azure could be an effective way to bridge this gap, by enabling an RESTful API layer powered by Azure which could encapsulate and standardize the enterprise data for use with apps + PowerApps as the Rapid application development User Interface where business apps can be built with agility and in collaboration with the business.

Environment

The lab requires access to a PowerApps environment with permission to create apps and custom connectors.

Note: Exercise 1 & 2 is targeted toward Azure developers. If you are not comfortable with Azure you can skip Exercise 1 and 2 & go straight to Exercise 3 & 4. There is a pre-configured API provided for completing Exercise 3 & 4.

Main components of the lab



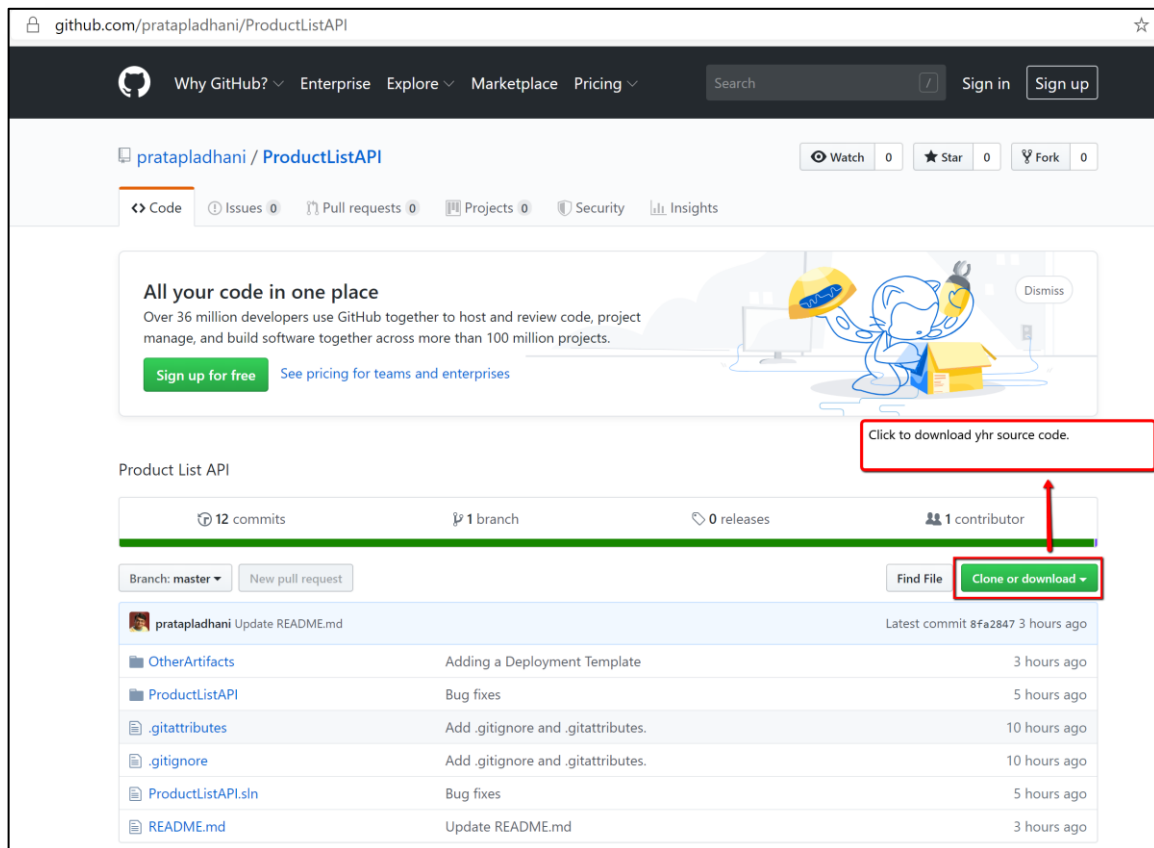
There are three main components of the lab:

1. An **Azure API App** which represents a Mock API connecting to back-end enterprise data (like SAP) and exposes a REST API endpoint for performing CRUD operations against that data.
2. A **custom connector** which exposes this API endpoint to PowerApps and Microsoft Flow makers
3. A **PowerApps canvas app** which in-turn uses the custom connector and provides a User Interface for end-users to performs CRUD operations. This app can be accessed from any device, browser or embedded experiences (like SharePoint webparts, Microsoft Search, Teams etc.)

Exercise 1 – Build the API App

Step 1. Download the source code or clone the GitHub repo:

<https://github.com/pratapladhani/ProductListAPI>



github.com/pratapladhani/ProductListAPI

Why GitHub? Enterprise Explore Marketplace Pricing Search Sign in Sign up

pratapladhani / ProductListAPI Watch 0 Star 0 Fork 0

<> Code Issues 0 Pull requests 0 Projects 0 Security Insights

All your code in one place
Over 36 million developers use GitHub together to host and review code, project manage, and build software together across more than 100 million projects.
[Sign up for free](#) [See pricing for teams and enterprises](#)

Product List API

12 commits 1 branch 0 releases 1 contributor

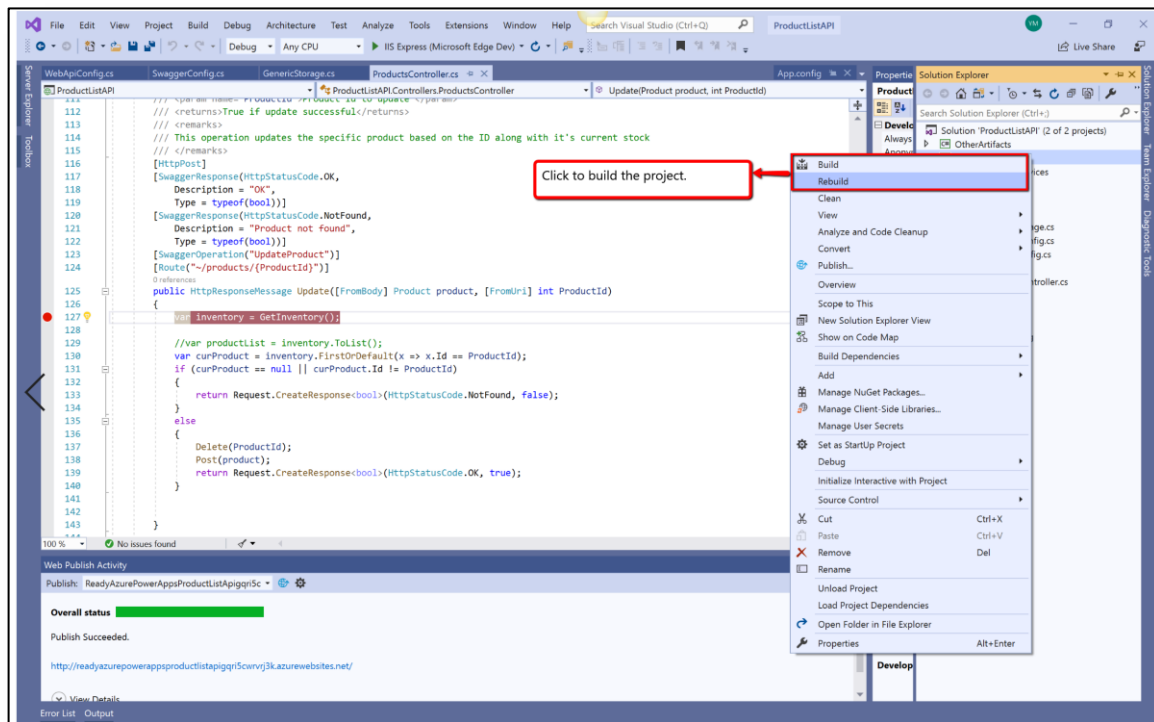
Branch: master New pull request Find File **Clone or download**

pratapladhani Update README.md Latest commit 8fa2847 3 hours ago

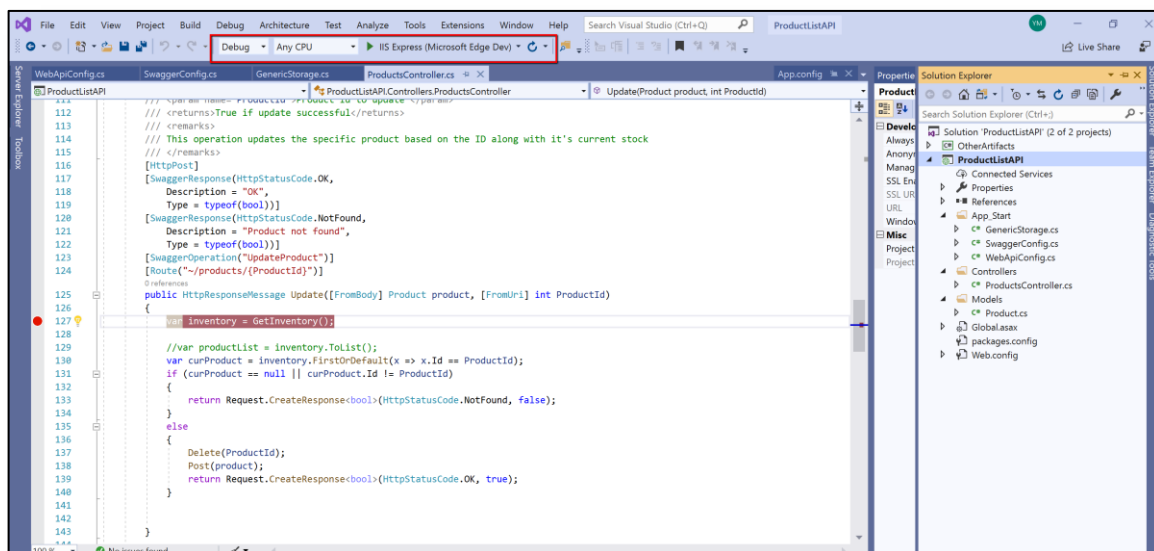
OtherArtifacts	Adding a Deployment Template	3 hours ago
ProductListAPI	Bug fixes	5 hours ago
.gitattributes	Add .gitignore and .gitattributes.	10 hours ago
.gitignore	Add .gitignore and .gitattributes.	10 hours ago
ProductListAPI.sln	Bug fixes	5 hours ago
README.md	Update README.md	3 hours ago

Click to download yhr source code.

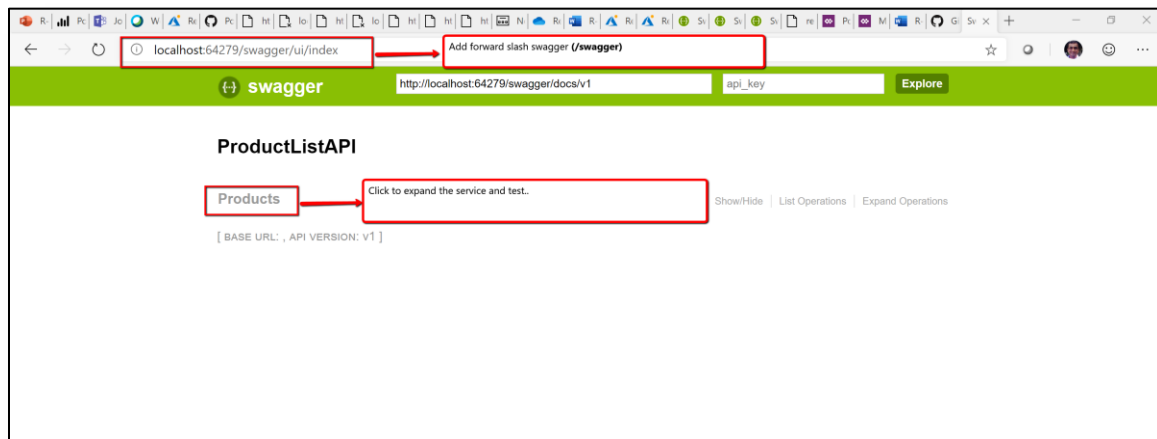
Step 2. Open the source code in Visual Studio and compile the source code.



Step 3. Run the Service in Local IIS Express of Visual Studio

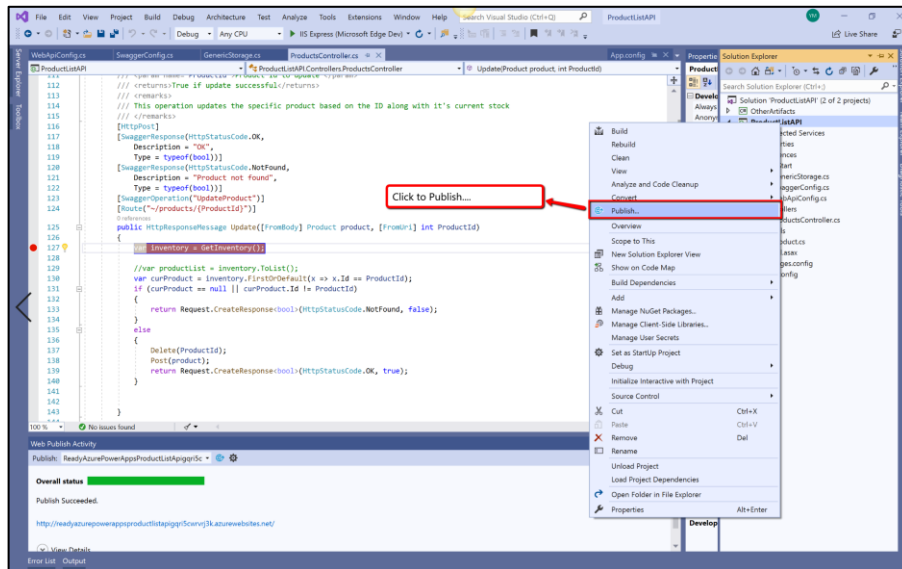


Step 4. Test the Service in the Browser. Append “/swagger” to the URL if the swagger UI is not automatically loaded.

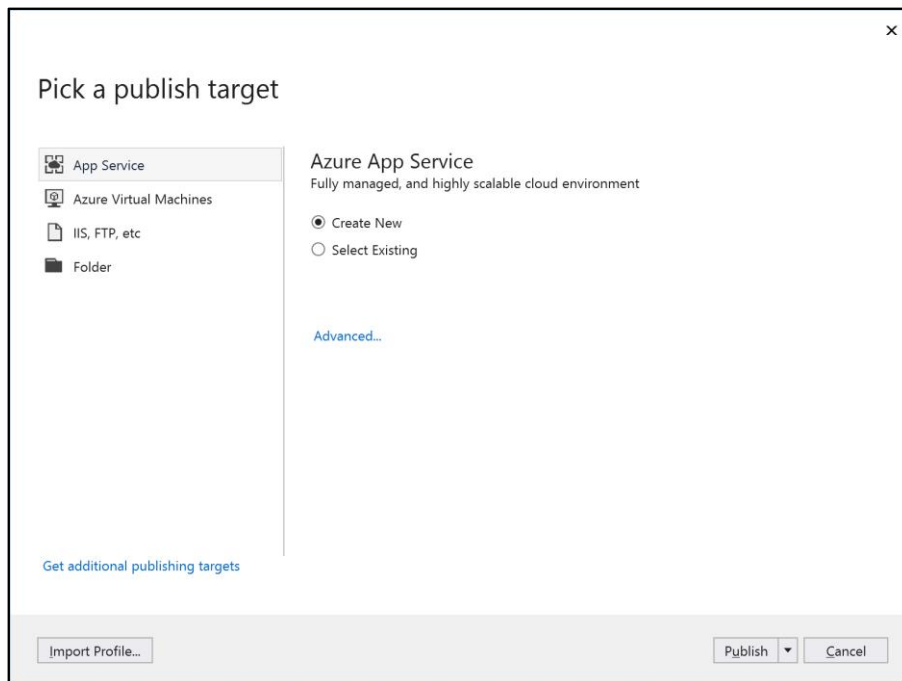


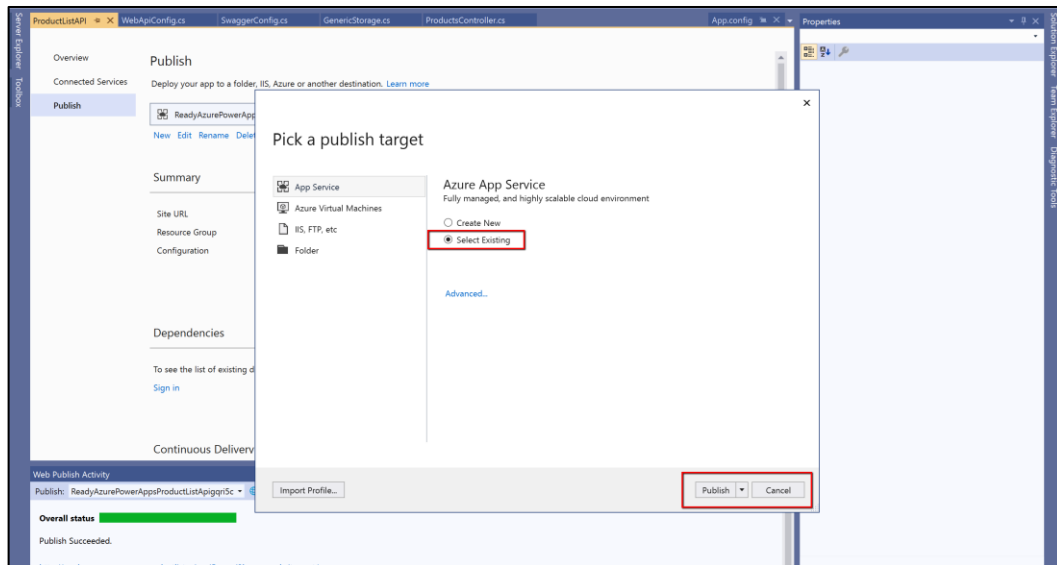
Exercise 2 – Deploy to Azure

Step 5. Open Visual Studio and then right click your publish your App to Azure.

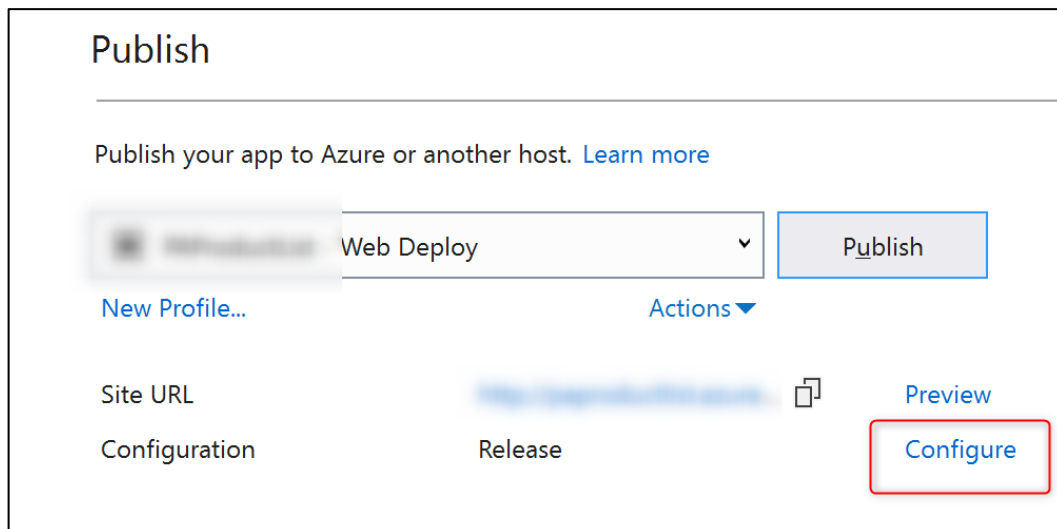


Step 6. Open the Azure publishing dialog to either create a new App Service OR select an existing App Service.

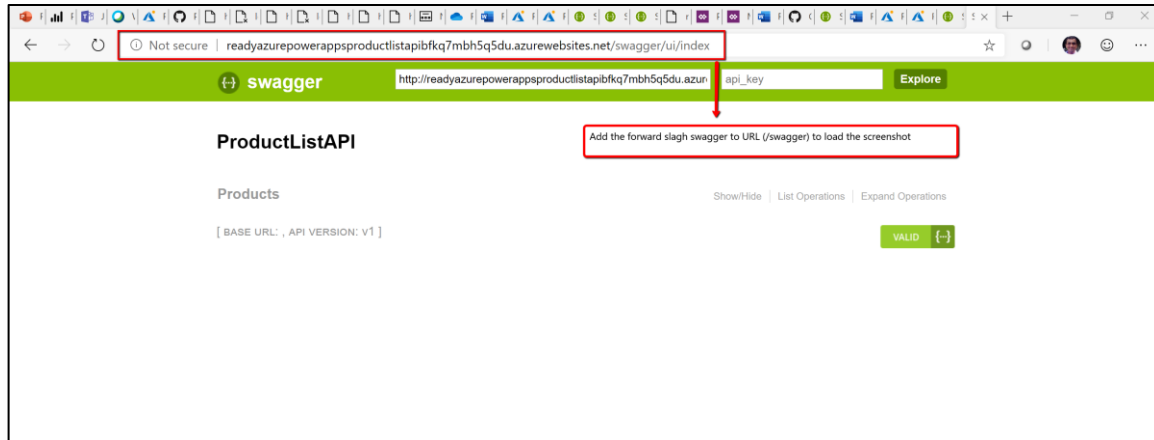




Step 7. Click on Configure > Edit the Site URL and append “/swagger” at the end of the Site URL, so that it automatically loads that URL once published for debugging purposes.



Step 9. After the application is published, if the swagger UI page is not displayed - append forward slash swagger (/swagger) to the URL to load the swagger UI.

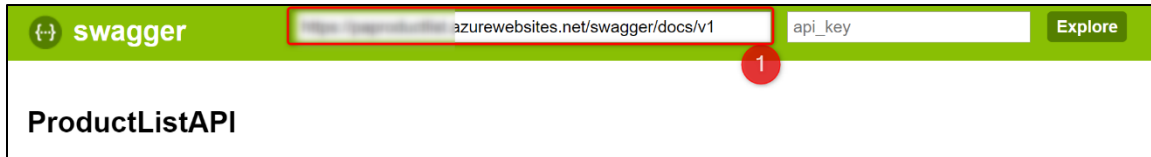


Step 10. Test the API using the Swagger UI Link by visiting the URL on the browser and trying out the different operations.

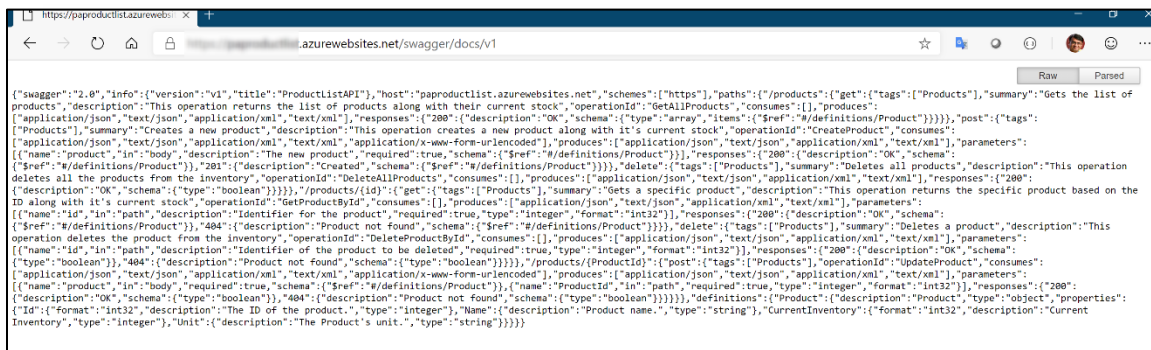


Exercise 3 – Create custom connector

Step 11. Copy the URL from the Swagger UI page top center.



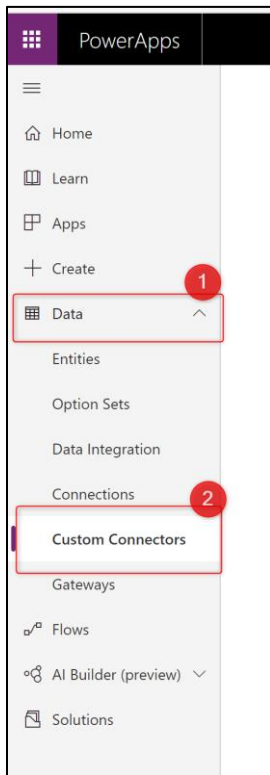
Step 12. Paste that URL in a new browser tab and press enter.



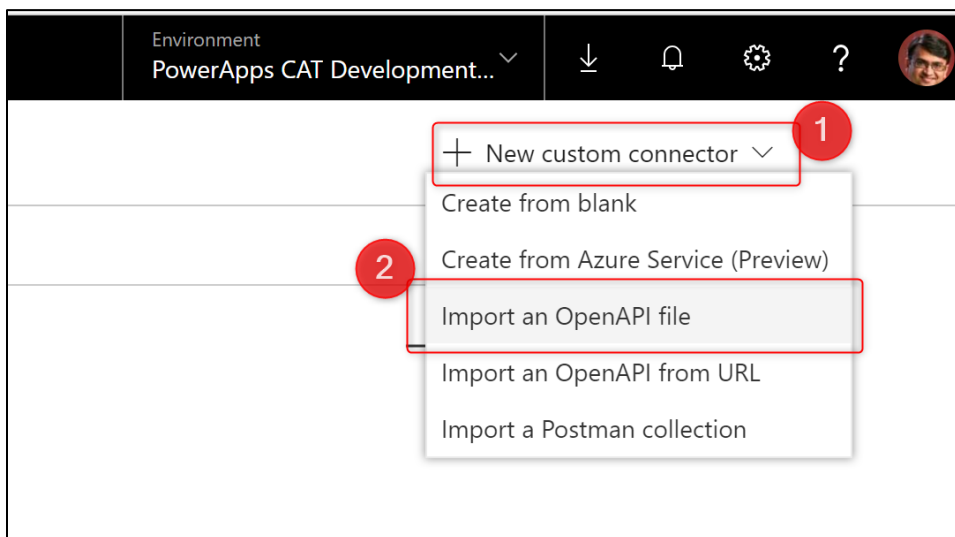
Step 13. Copy all the content of the page and paste it in Notepad and save the file somewhere on your local machine with the name “Swagger.json”.

Step 14. Open up your favorite browser and navigate to <https://make.powerapps.com>. Login if required. This lab doesn't require CDS, so feel free to pick up any environment where you have maker access.

Step 15. Click on Data > Custom Connectors on the Left Navigation.



Step 16. Click on the + New Custom Connector link on the top right.



Step 17. Give a name to the Custom Connector - for e.g. **“PowerApps Product List API”**.

Step 18. Click on the **Import** button and select the **“Swagger.json”** file that you had saved earlier for Import an Open API file and click on **Continue**.

Create a custom connector

Connector name

PowerApps Product List API

Import an OpenAPI file

Swagger.json

Import

Continue

Cancel

Step 19. (Optional) Upload a picture for the Custom Connector. I have one uploaded here, if you want to reuse:

[https://github.com/pratapladhani/ProductListAPI/blob/master/OtherArtifacts/Swagger/Products list icon.jpg](https://github.com/pratapladhani/ProductListAPI/blob/master/OtherArtifacts/Swagger/Products%20list%20icon.jpg) Provide a color code of your choice for e.g. #ffffff and provide a description.

General information

Upload connector icon

Supported file formats are PNG and JPG. (< 1MB)

Icon background color

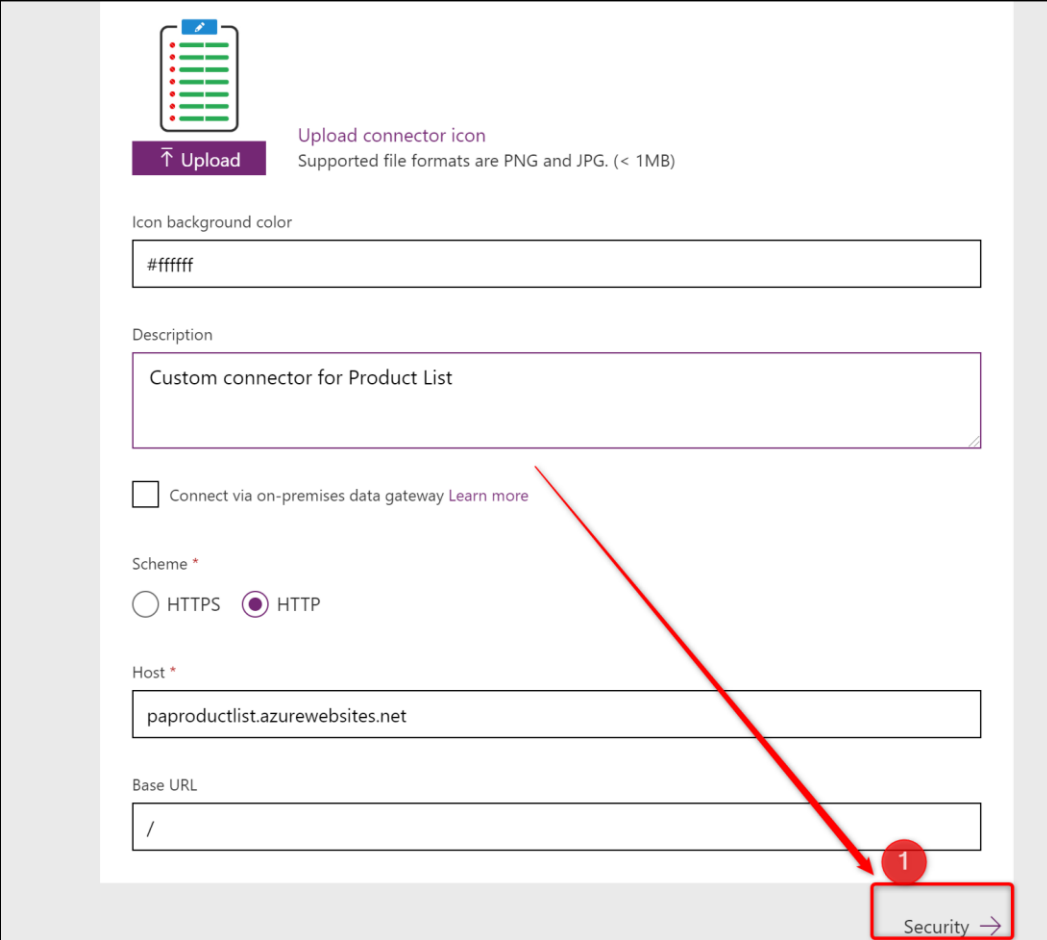
#ffffff

Description

Custom connector for Product List

☐ Connect via on-premises data gateway [Learn more](#)

Step 20. Click on Security > in the bottom right.



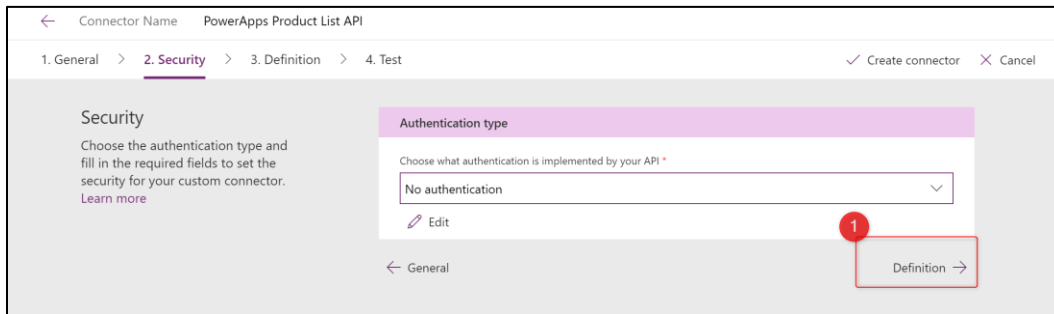
The screenshot shows a web form for uploading a connector icon. At the top, there is a purple 'Upload' button with an upward arrow icon. To its right, the text reads 'Upload connector icon' and 'Supported file formats are PNG and JPG. (< 1MB)'. Below this, there is a text input field for 'Icon background color' containing '#ffffff'. Underneath is a larger text area for 'Description' containing 'Custom connector for Product List'. A checkbox labeled 'Connect via on-premises data gateway' is followed by a 'Learn more' link. Below that, the 'Scheme' section has two radio buttons: 'HTTPS' (unselected) and 'HTTP' (selected). The 'Host' field contains 'paproductlist.azurewebsites.net'. The 'Base URL' field contains '/'. At the bottom right, a red arrow points from the 'Connect via on-premises data gateway' checkbox area to a red circle containing the number '1', which is positioned above a button labeled 'Security' with a right-pointing arrow.

Step 21. Keep the default “No authentication” option in the next screen.

Note:

This is not advisable in Production APIs – we are skipping this only for saving time in this lab. Refer to this article : <https://docs.microsoft.com/en-us/connectors/custom-connectors/azure-active-directory-authentication> for the guidance for protecting the API endpoint with Azure AD.

Step 22. Click on Definition to go to the next tab.

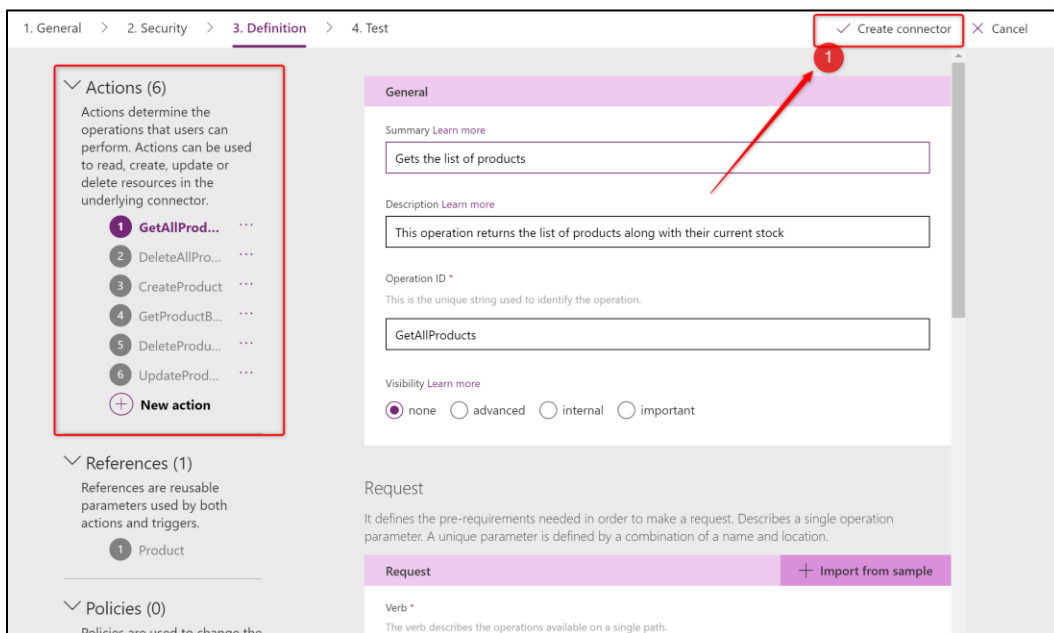


Step 23. Validate that all the methods have the right Summary and Description.

Note:

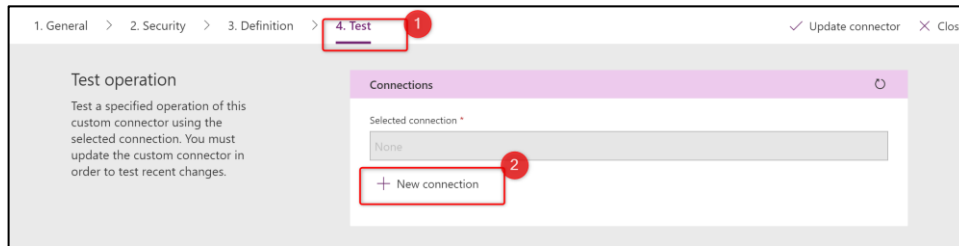
Since we had generated Summary and Description through XML comments for all our operations, we don't have to type in the summary and description for most methods.

Step 24. Click on Create connector on the top right.

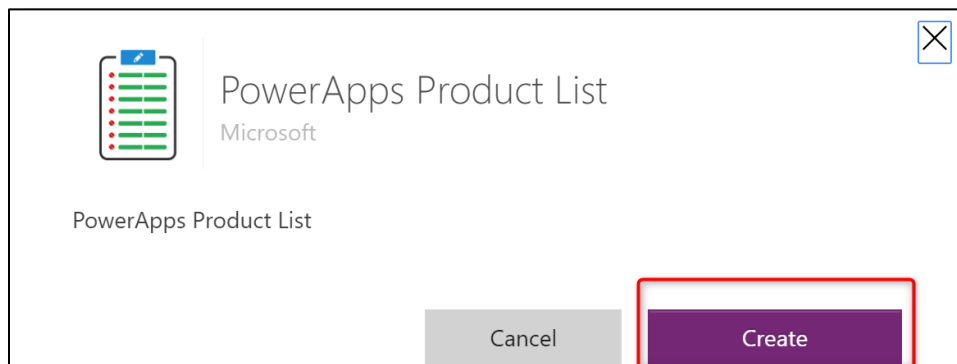


Step 25. Once the connector is saved successfully, click on **4. Test**.

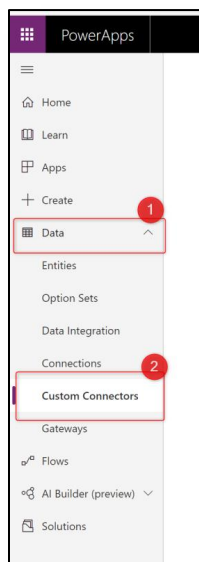
Step 26. Click on **+ New connection**.



Step 27. Click on **Create**.



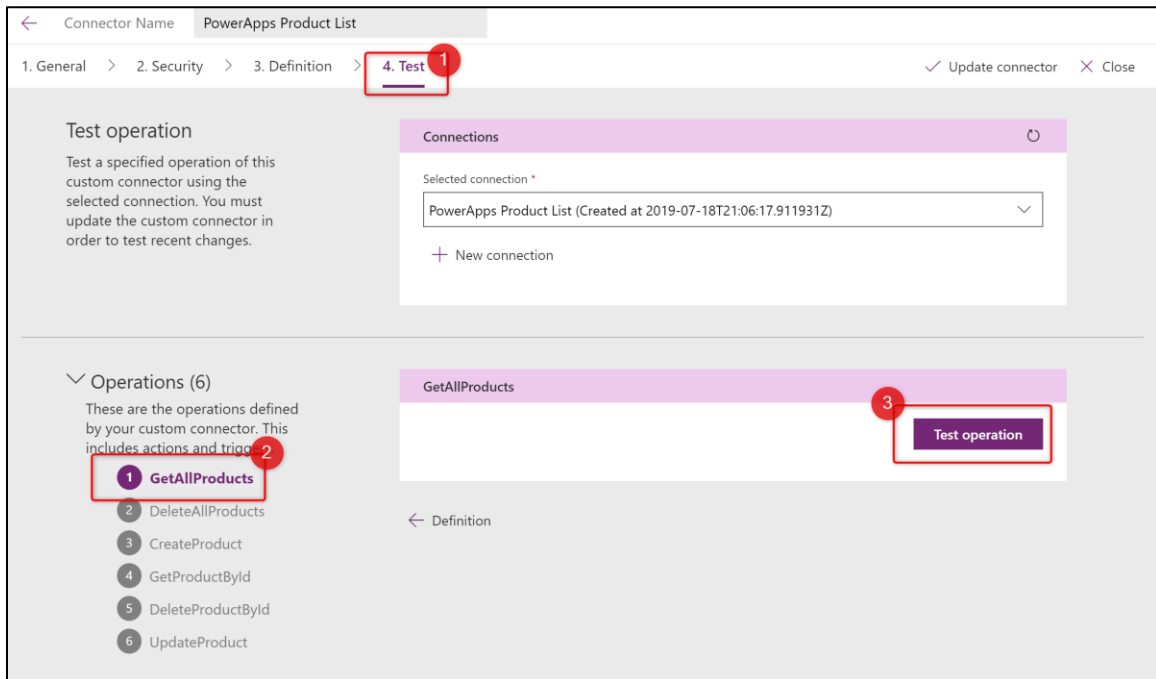
Step 28. After the connection is created, it will navigate you to the connections tab. Click on the **Left Nav > Data > Custom Connectors** link again.



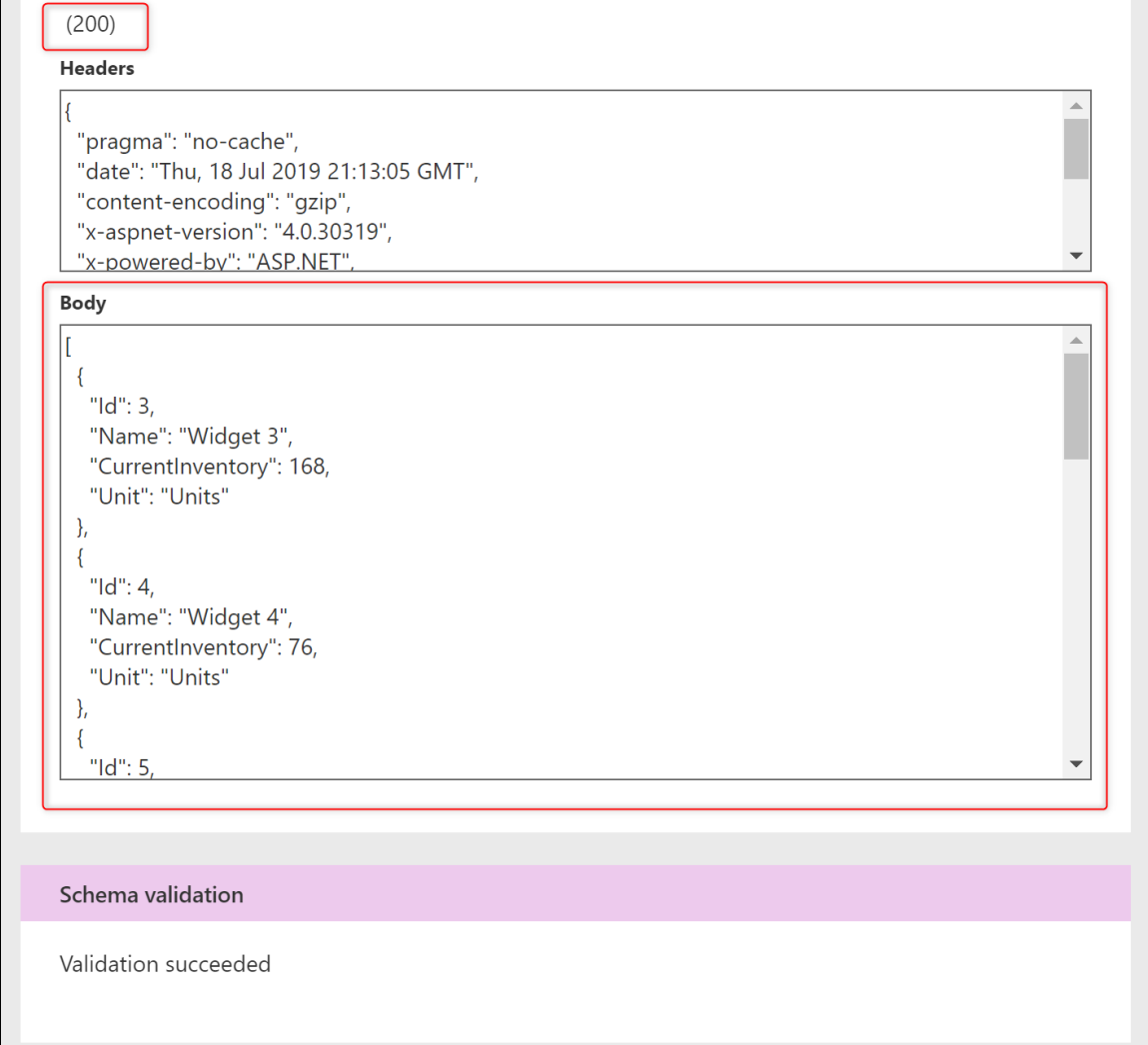
Step 29. Select the connector you just created and click the Edit icon to edit the connector.



Step 30. Click on 4. Test tab. Now the connection that you created in the previous step must have been populated. Click the GetAllProducts operation and click on the Test operation.



Step 31. You should be able to see a 200 status with the JSON of the results from your API showing in the Test Results.



The screenshot displays a REST client interface with the following components:

- Status:** A red-bordered box at the top left contains the text "(200)".
- Headers:** A section titled "Headers" containing a JSON object:

```
{  "pragma": "no-cache",  "date": "Thu, 18 Jul 2019 21:13:05 GMT",  "content-encoding": "gzip",  "x-aspnet-version": "4.0.30319",  "x-powered-by": "ASP.NET",}
```
- Body:** A section titled "Body" containing a JSON array of three objects:

```
[  {    "Id": 3,    "Name": "Widget 3",    "CurrentInventory": 168,    "Unit": "Units"  },  {    "Id": 4,    "Name": "Widget 4",    "CurrentInventory": 76,    "Unit": "Units"  },  {    "Id": 5,
```
- Schema validation:** A purple header bar with the text "Schema validation".
- Validation result:** Below the header, the text "Validation succeeded" is displayed.

Step 32. The next step is to create a canvas app using this custom connector.

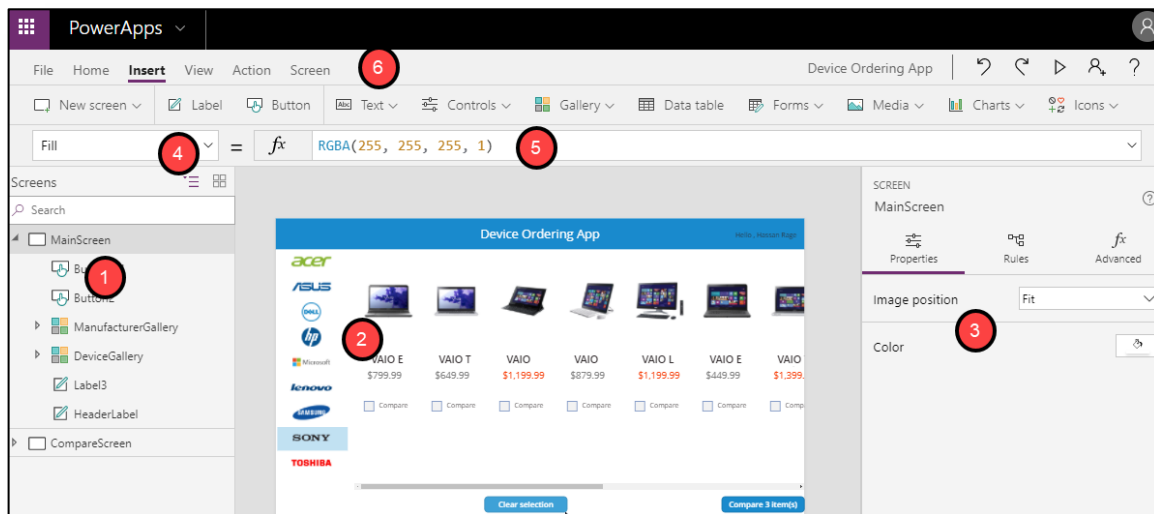
Exercise 4 – Create canvas app

PowerApps Canvas Studio Layout

PowerApps Canvas Studio is available as a web application (<http://make.powerapps.com>) that you can use in any modern browser.

PowerApps Studio is designed to have a user interface familiar to users of the Office suite. It has three panes and a ribbon that make app creation feel **like building a slide deck in PowerPoint**. Formulas are entered within a function fx bar that is like Excel. Studio components:

1. **Left navigation bar**, which shows all the screens and controls in your app
2. **Middle pane**, which contains the app screen you are working on
3. **Right-hand pane**, where you configure properties for controls, bind to data, create rules, and set additional advanced settings
4. **Property** drop-down list, where you select the property for the selected control that you want to configure
5. **Formula bar**, where you add formulas (like in Excel) that define the behavior of a selected control
6. **Ribbon**, where you perform common actions including customizing design elements



Locale-specific difference in formulas



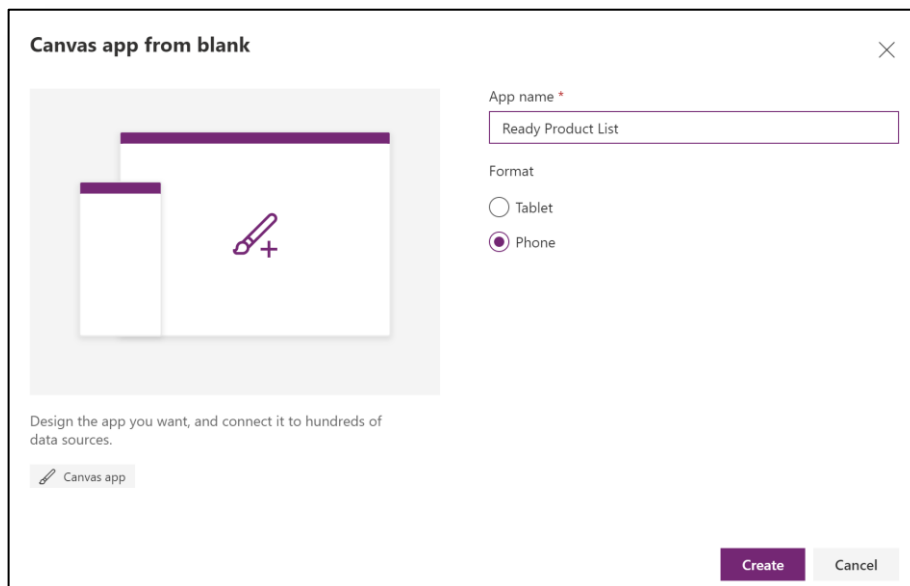
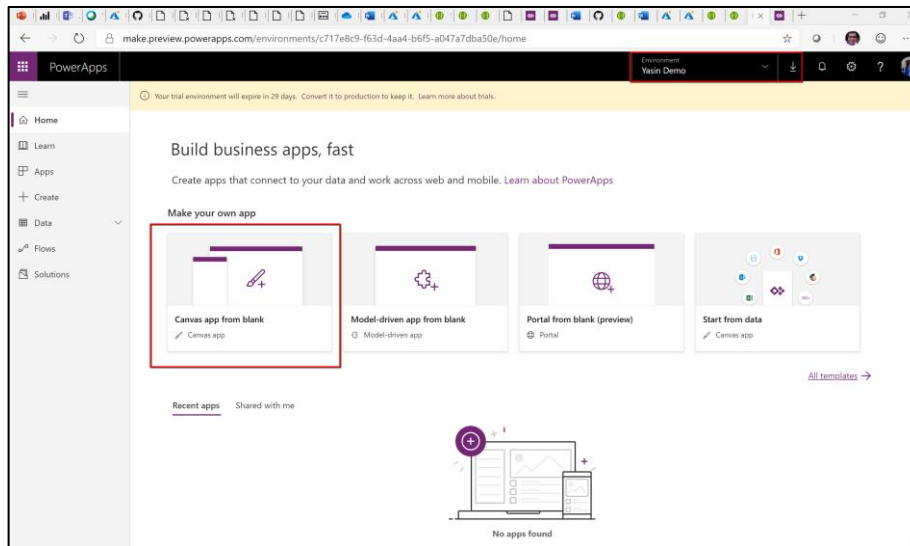
Before you begin, please note that if your computer has its regional settings set to use the comma ',' for its decimal separator (like in much of Europe) your formulas will need to use a semicolon ';' instead of a comma in your formulas. For example:

En-US `Filter(Machines, OEMsGallery.Selected.MFR=MFR)`

de-DE `Filter(Machines; OEMsGallery.Selected.MFR=MFR)`

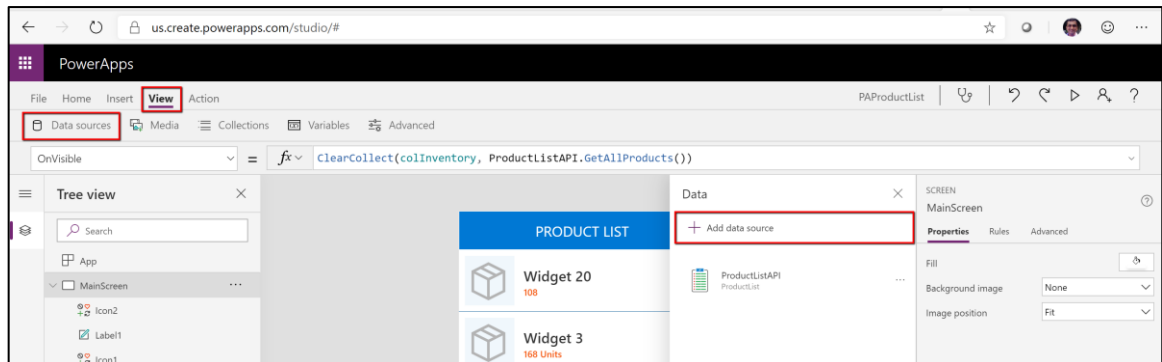
Step 33. Create a Canvas App with Phone Layout

Go To <http://make.powerapps.com> and select your environment where you deployed custom connector in Exercise 3.



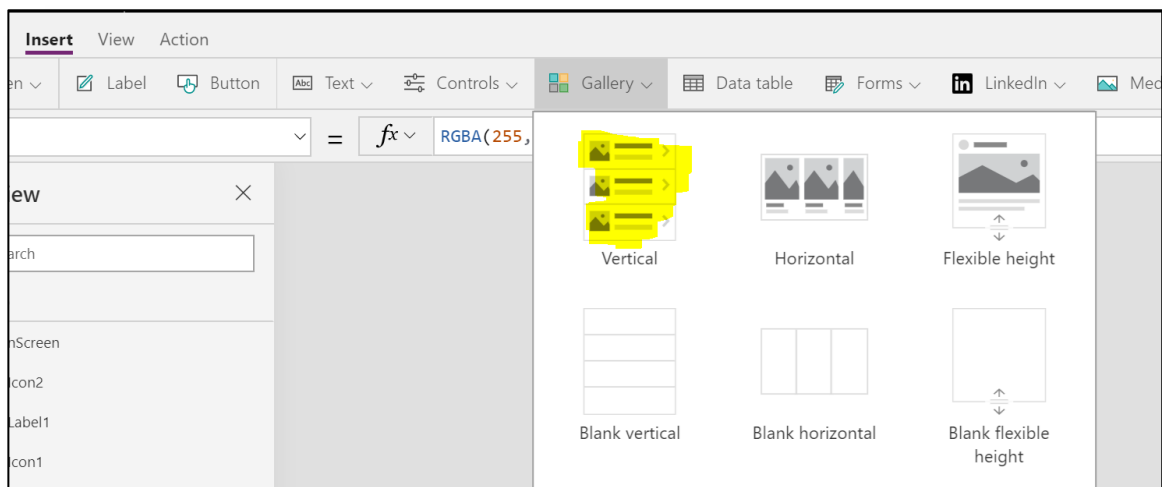
Step 34. Add your custom connector that you configured in Exercise 3

1. Go to View->Data Sources. Select “+ Add data source.”
2. Select “+ New Connection”
3. Search for the Custom Connector You created in the previous steps
4. Select and Create the connection



Step 35. Bind Data to the Gallery

1. Rename Screen1 to MainScreen
2. Add a Header
 - a. Insert a Label
 - b. Change Text to “PRODUCT LIST”
 - c. Change Fill Color to Blue
 - d. Change Color to White
3. Insert a vertical Gallery Control



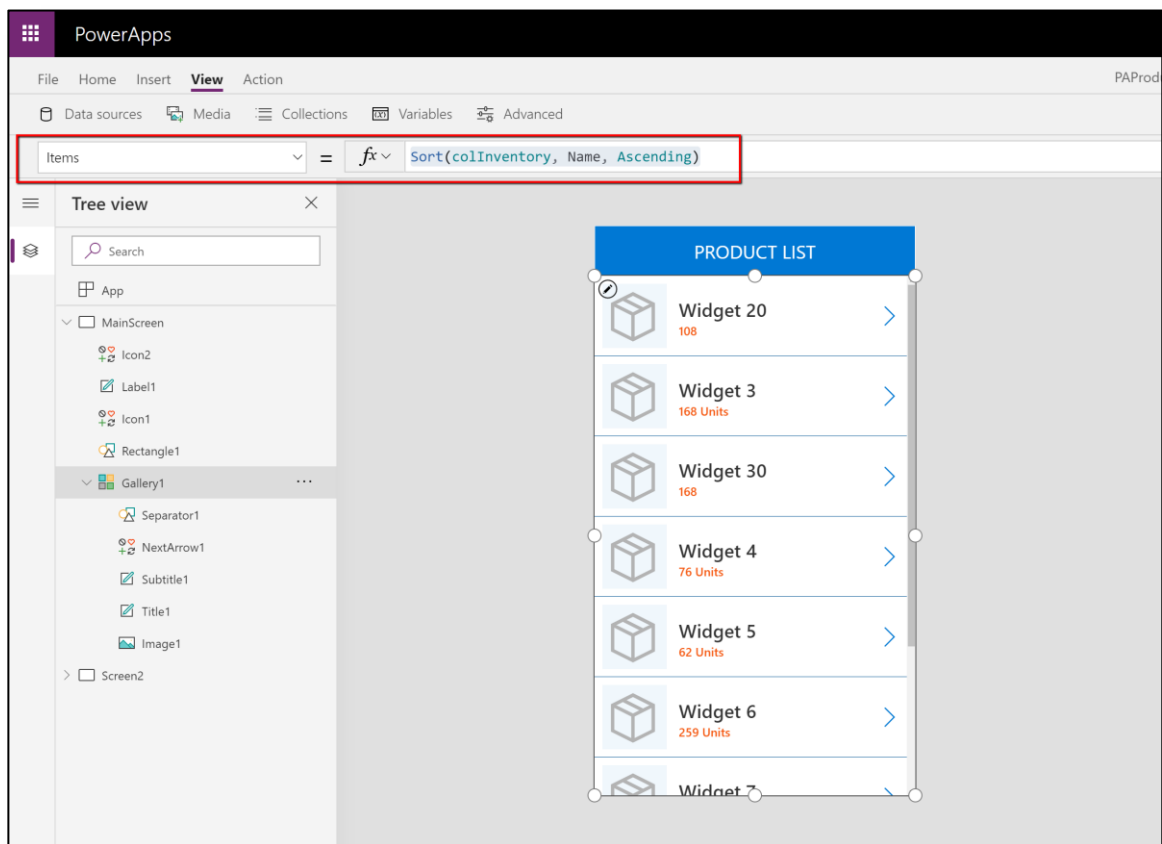
4. Go mainScreen > OnVisible >
5. Enter the following formula

```
ClearCollect(colInventory, ProductListAPI.GetAllProducts())
```

(This will call the API and save data in a local collection called **colInventory**)

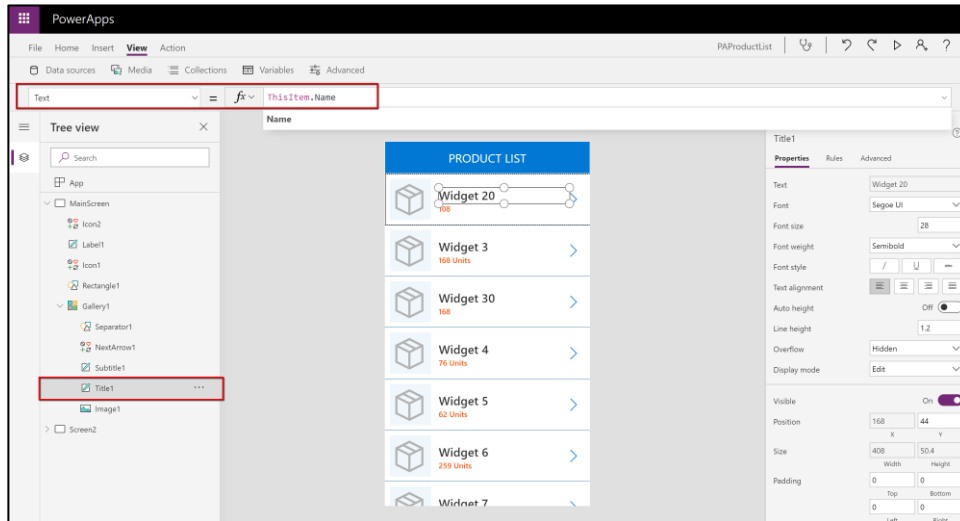
6. Create Screen 2 . Select Screen 2 and then select Screen1. This will ensure the OnVisible function will be triggered and the data is saved to the collection.
7. Select Gallery1=> Items property and type the following: >

```
Sort(colInventory, Name, Ascending)
```



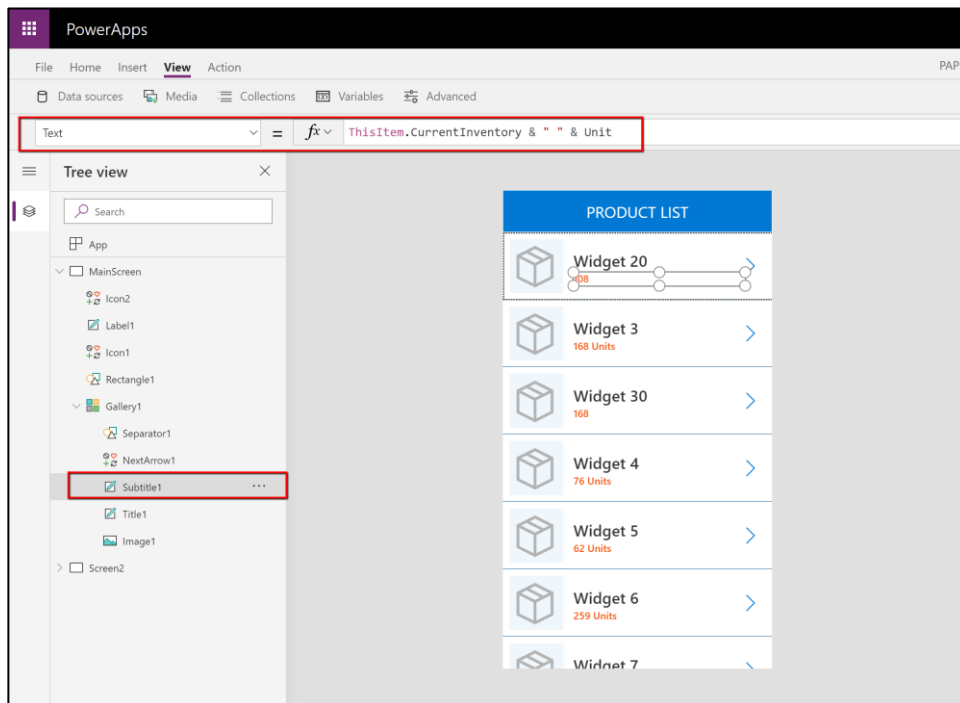
8. Set the label item formula >

ThisItem.Name

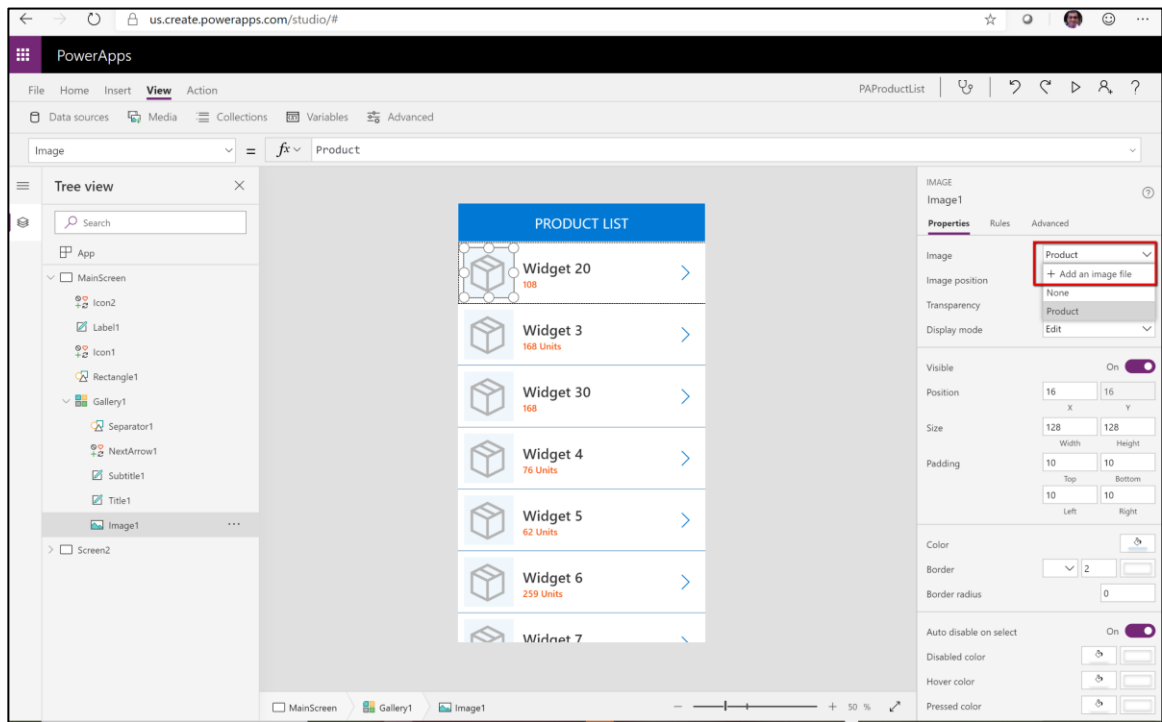


9. Set the label item formula >

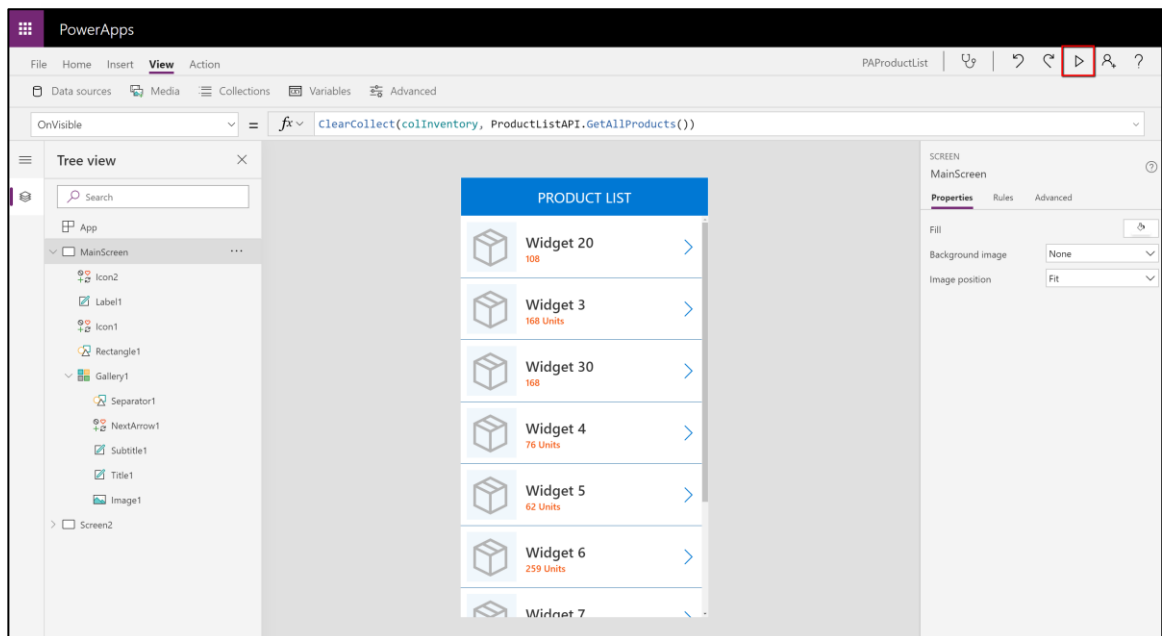
ThisItem.CurrentInventory & " " & Unit



10. Select a default image product.svg



11. Slect the play button to test your app.



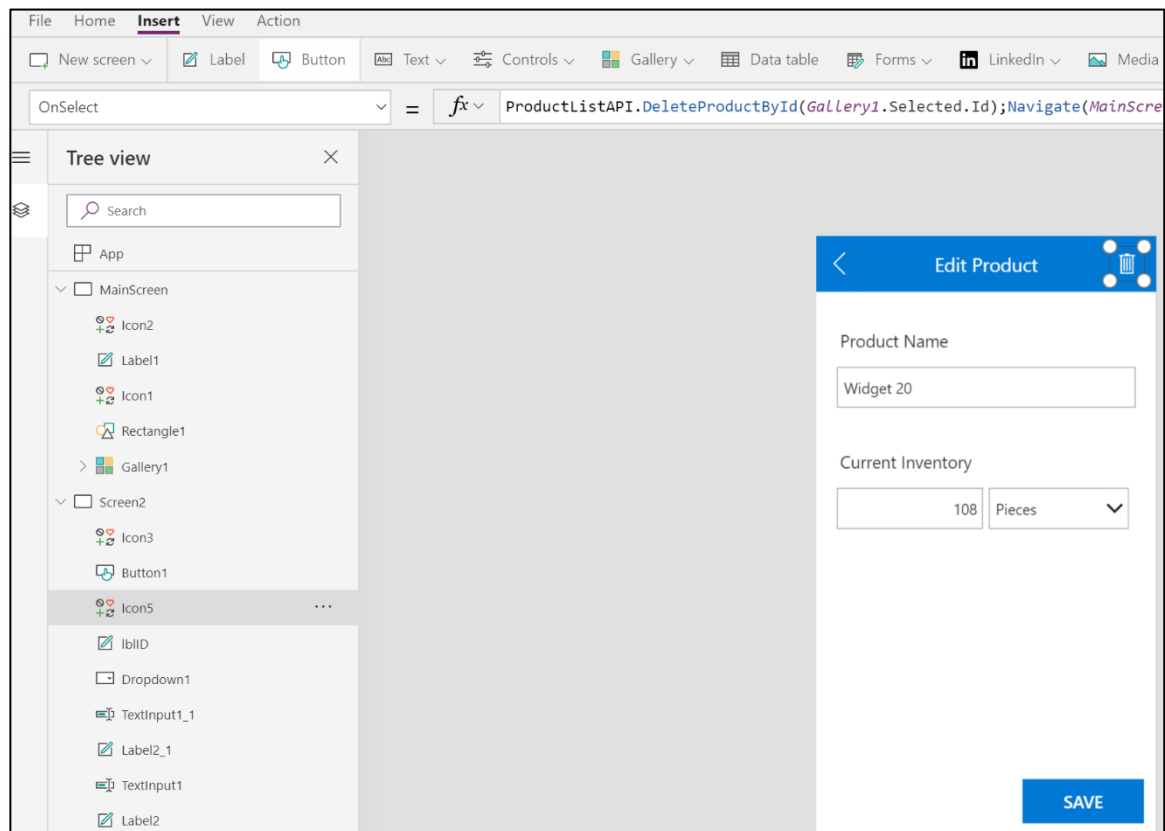
Step 36. Create the Detail Screen to Edit and Delete Products

1. Copy Header control from mainScreen and paste into Screen2
2. Add the Back icon, Select the OnSelect property and type

```
Back(ScreenTransition.UnCoverRight)
```

3. Add Delete icon and enter the following formula:

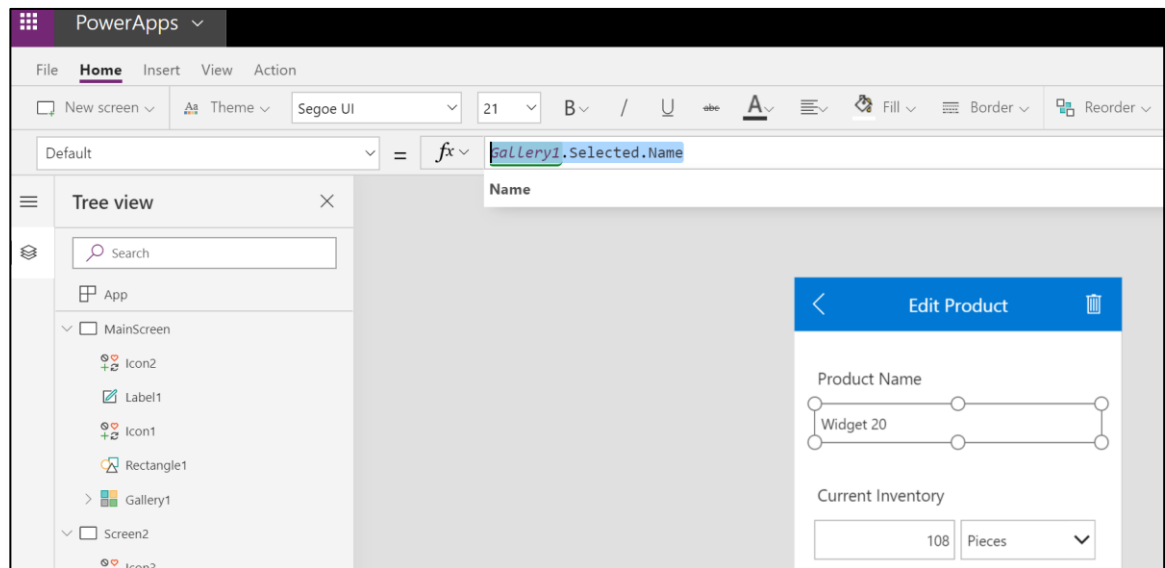
```
ProductListAPI.DeleteProductById(Gallery1.Selected.Id);Navigate(MainScreen,ScreenTransition.UnCover)
```



4. Add the Label for "Product Name"

5. Add TextInput and bind Default property to

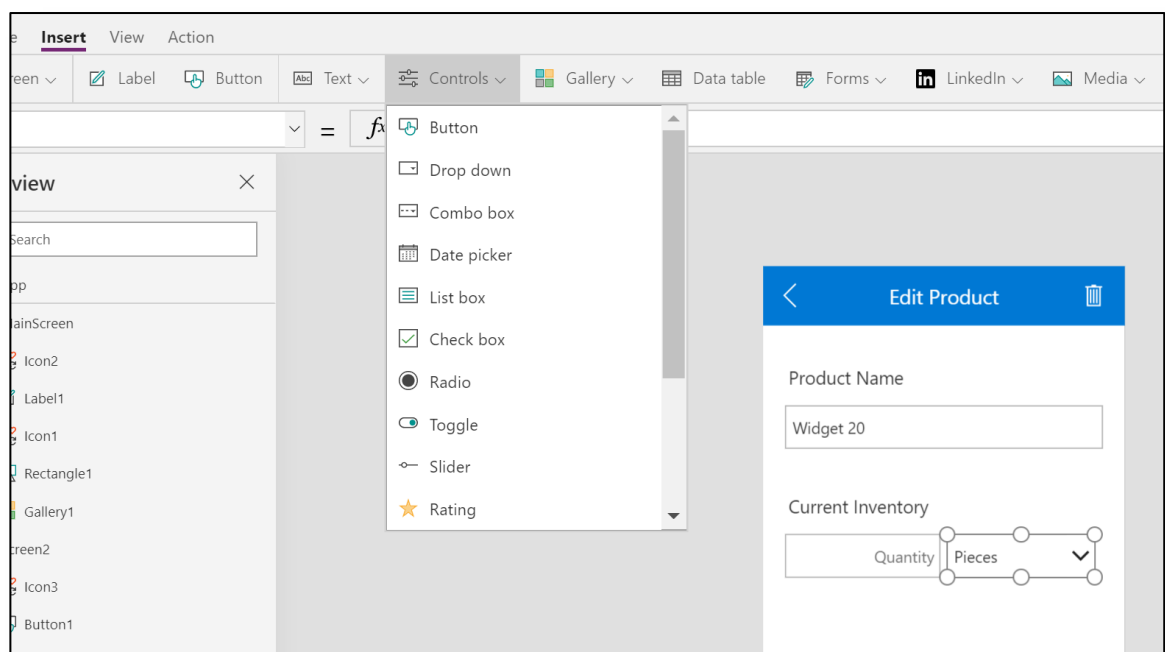
Gallery1.Selected.Name



6. Add the Label for “Current Inventory”
7. Add TextInput and bind the Default property to

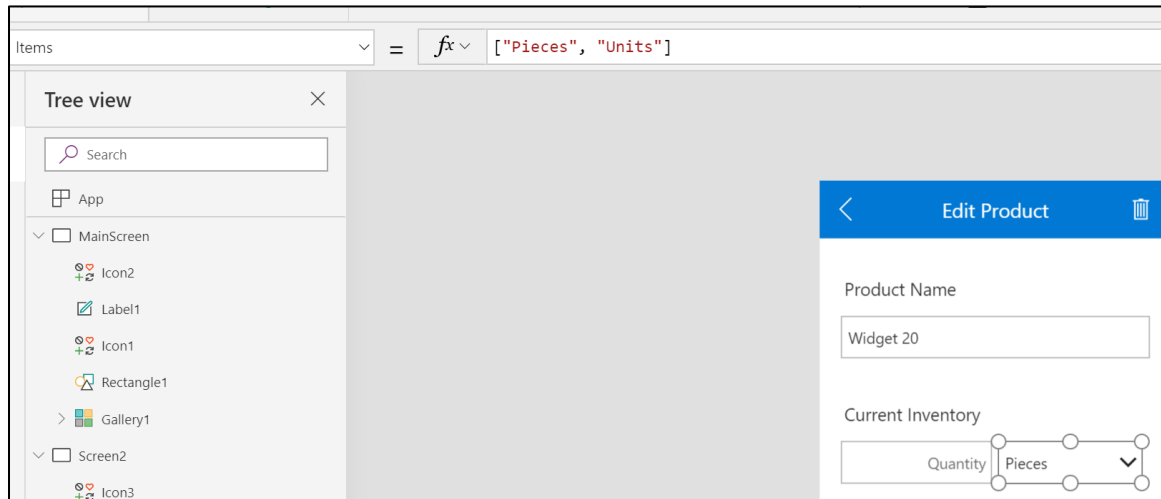
Gallery1.Selected.CurrentInventory

8. Add a dropdown control



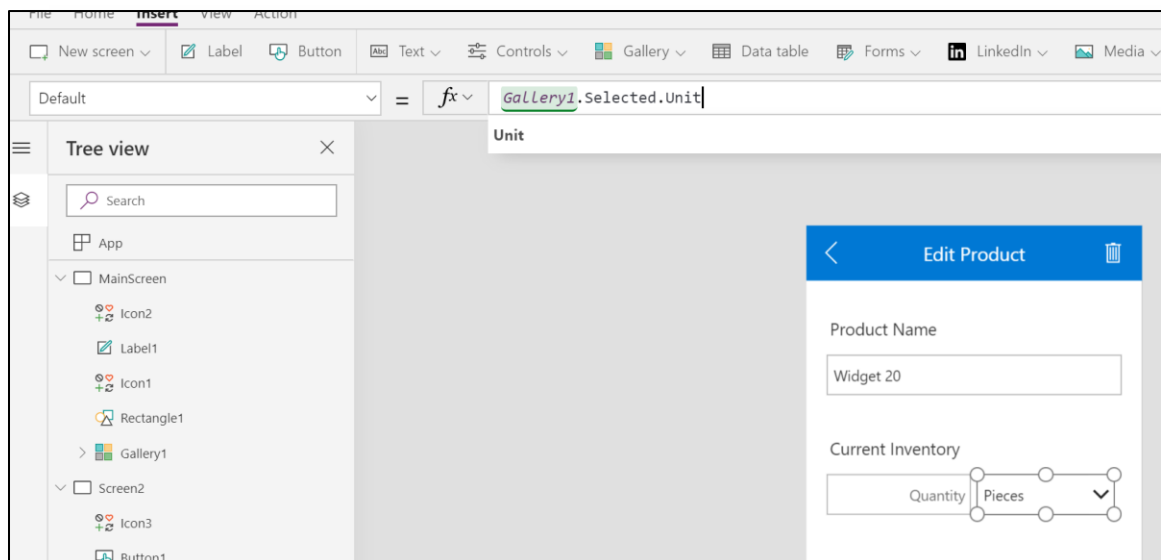
9. Bind the Items property to

["Pieces", "Units"]



10. Set the Default property

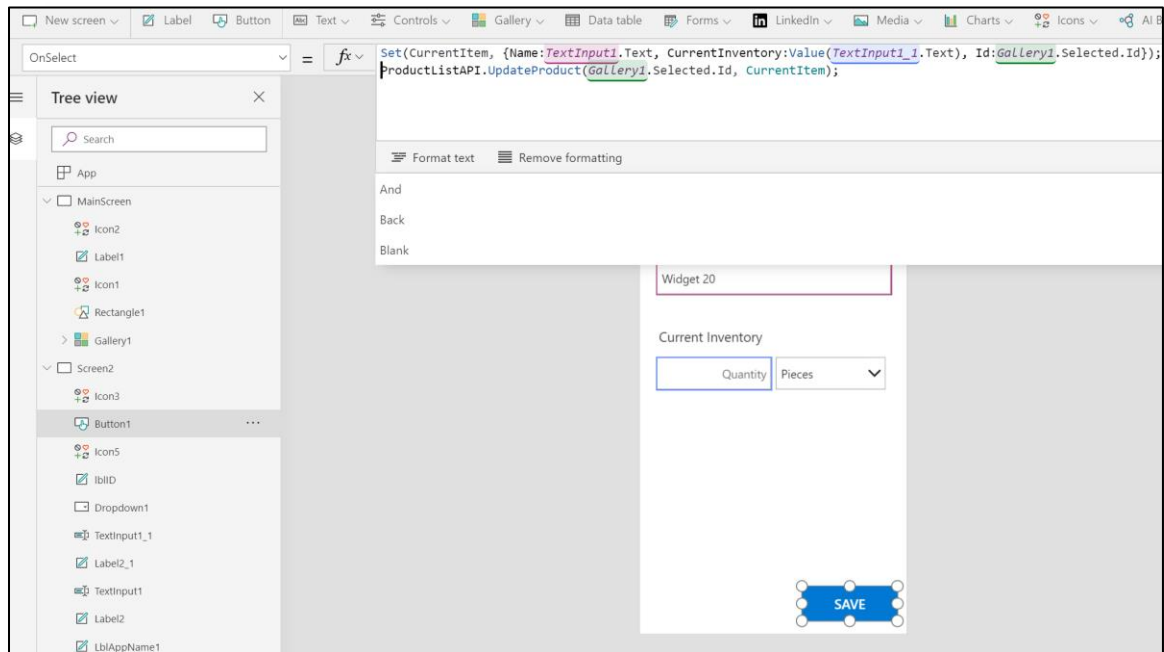
Gallery1.Selected.Unit



11. Add Save Button

12. Bind the OnSelect to

```
Set(CurrentItem, {Name:TextInput1.Text,  
CurrentInventory:Value(TextInput1_1.Text),  
Id:Gallery1.Selected.Id});  
ProductListAPI.UpdateProduct(Gallery1.Selected.Id, CurrentItem);
```



Copyright

© 2019 Microsoft Corporation. All rights reserved.

By using this demo/lab, you agree to the following terms:

The technology/functionality described in this demo/lab is provided by Microsoft Corporation for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the demo/lab to evaluate such technology features and functionality and provide feedback to Microsoft. You may not use it for any other purpose. You may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this demo/lab or any portion thereof.

COPYING OR REPRODUCTION OF THE DEMO/LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED.

THIS DEMO/LAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE. THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS DEMO/LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK. WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS. YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCTIONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

FEEDBACK. If you give feedback about the technology features, functionality and/or concepts described in this demo/lab to Microsoft, you give to Microsoft, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a Microsoft software or service that includes the feedback. You will not give feedback that is subject to a license that requires Microsoft to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

MICROSOFT CORPORATION HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE DEMO/LAB, INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. MICROSOFT DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF DEMO/ LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE DEMO/LAB FOR ANY PURPOSE.

DISCLAIMER

This demo/lab contains only a portion of new features and enhancements in Microsoft PowerApps. Some of the features might change in future releases of the product. In this demo/lab, you will learn about some, but not all, new features.