Building a Shopping Cart for Processing Orders

Lab Time: 60 minutes

Lab Folder: C:\Student\Modules\03_ShoppingCart\Lab

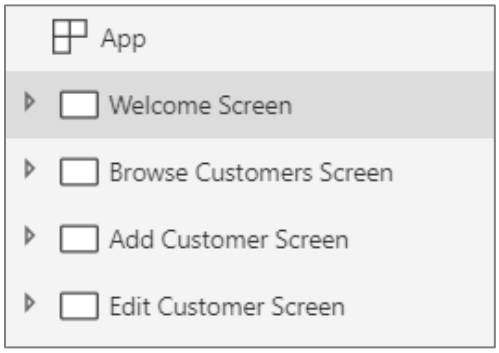
Lab Overview: In this lab you will extend the canvas app named **Customer Ordering** that you created in the previous lab by adding the behavior to add products into a shopping cart and submit an order.

Prerequisite: This lab assumes that you have already completed the previous lab titled **Building a Data-driven Canvas App**.

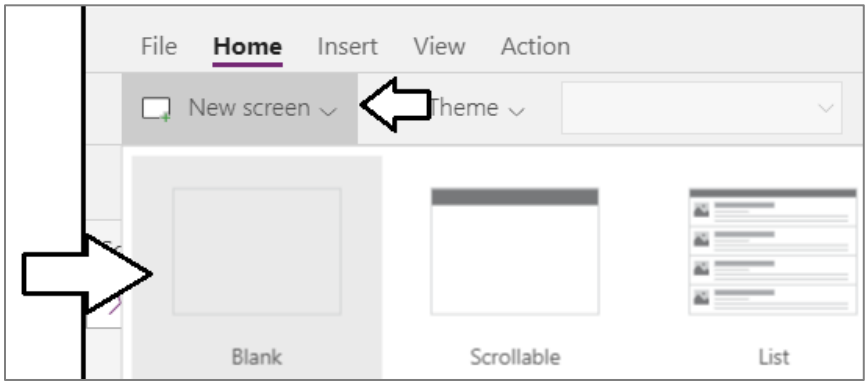
Exercise 1: Create The Browse Products Screen

In this exercise, you will add the **Browse Products Screen** which allows the user to view and filter a set of products.

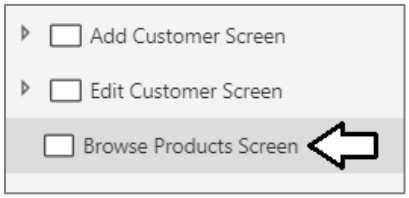
1. Open the canvas app named **Customer Ordering** that you created in lab 2 titled **Building a Data-driven Canvas App**.
   a) Navigate to the PowerApps portal at https://web.powerapps.com.
   b) Open the canvas app named **Customer Ordering** in edit mode.
   c) The **Customer Ordering** canvas app should currently contain four screens as shown in the following screenshot.



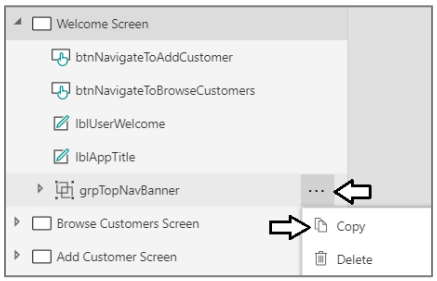
2. Create the **Browse Products Screen**.
   a) Use the **New screen** command from the **Home** tab to add a new **Blank** screen.



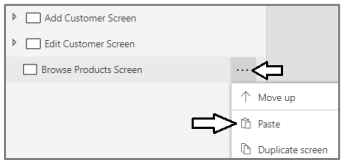
   b) Rename the new screen to **Browse Products Screen**.



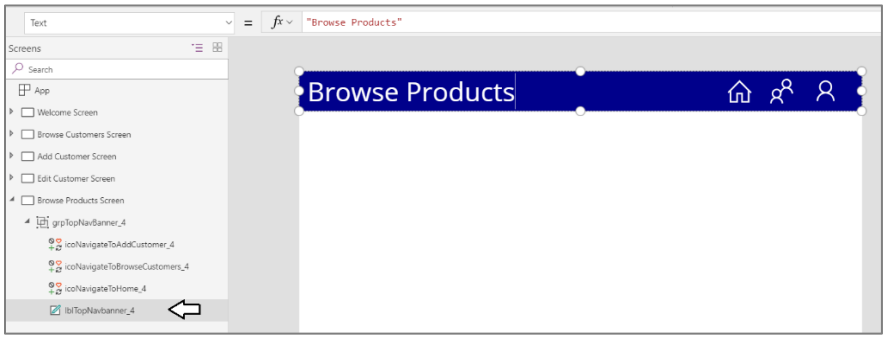
3. Copy and paste the group named **grpTopNavBanner** from the **Welcome Screen** to the **Browse Products Screen**.
   a) Expand the **Welcome Screen** in the left tree view.
   b) Drop down the context menu for **grpTopNavBanner** and select **Copy**.



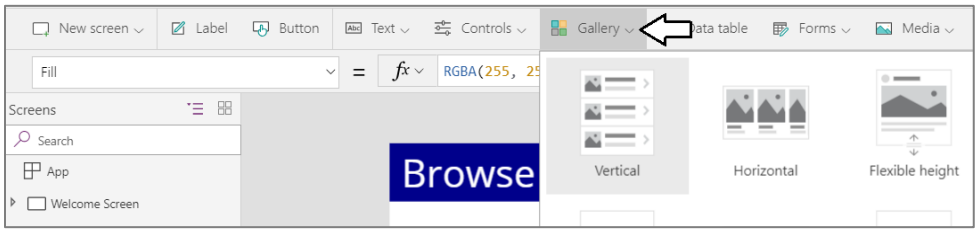
   c) Drop down the context menu for the **Browse Products Screen** and select **Paste**.



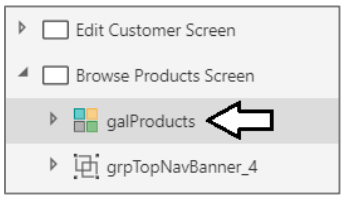
   d) Update the text on the Top Nav Banner to **"Browse Products"**.



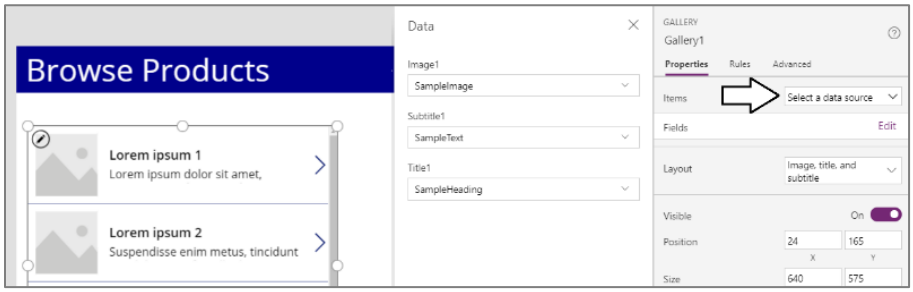
4. Add a new gallery to the **Browse Products Screen** for displaying products.
   a) Using the **Gallery** menu on the **Insert** tab, add a new **Vertical** gallery to the **Browse Products Screen**.



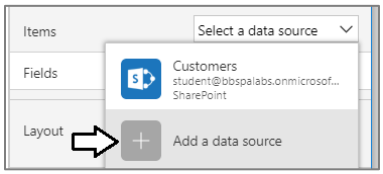
   b) Rename the new gallery to **galProducts**.



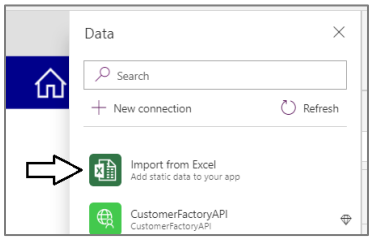
c) With **galProducts** selected in the left tree view, drop down the **Select a data source** menu in the **Properties** pane.



d) Select **Add a data source**.



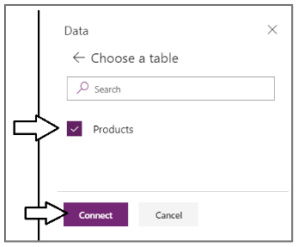
e) Select **Import from Excel**.



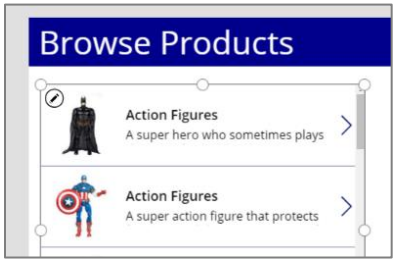
f) When the file **Open** dialog appears, select the Excel workbook named **ProductCatalog.xlsx** at the following path.

```
C:\Student\Extras\ProductCatalog.xlsx
```

g) When prompted to **Choose a table** on the **Data** pane, select the **Products** table and click **Connect**.

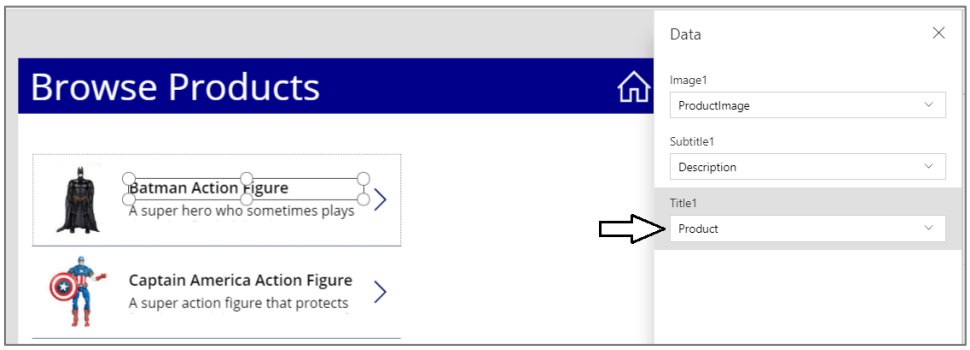


h) You should now see that he gallery is populated with products as shown in the following screenshot.

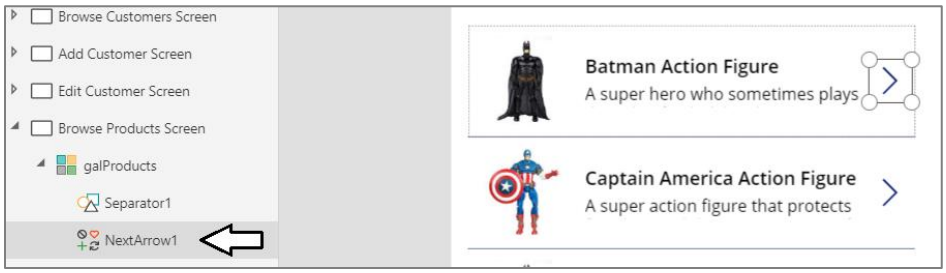


5. Configure the fields displayed by **galProducts**.
   a) Click the **Edit** link for the gallery **Fields** property to display the **Data** pane with the field mapping controls.

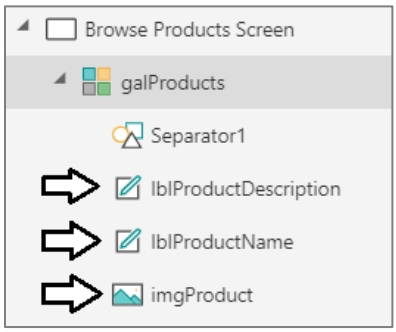
   b) In the **Data** pane, set the field for **Title1** to **Product**.



6. Configure the controls inside the item template of **galProducts**.
   a) Expand **galProducts** in the left tree view to see the controls inside the gallery template.
   b) Select and delete the **NextArrow** icon from the left tree view.



   c) Rename the **Title1** label to **lblProductName**.
   d) Rename the **Subtitle1** label to **lblProductDescription**
   e) Rename the **Image1** image to **imgProduct**.



f) With **galProducts** selected, click on the small button with the pen icon to enter edit mode for the gallery's item template.

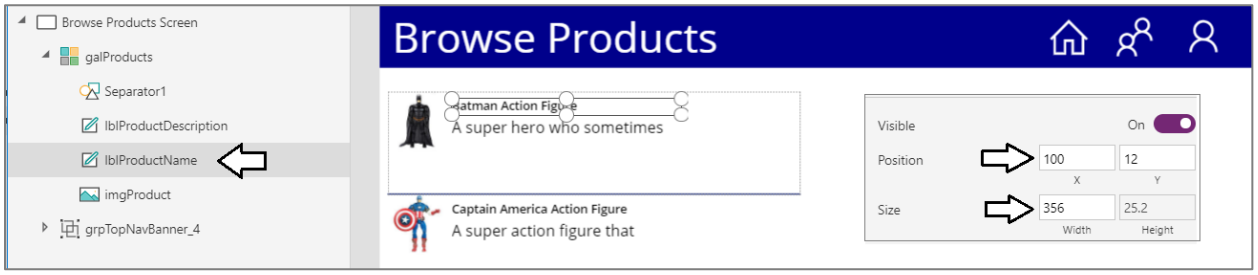
g) Select the control named **imgProduct** and update it **Height** property and **Width** property to a value of **92**.

h) Position **imgProduct** in the top left corner of the gallery template so the **X** property and the **Y** property both equal **4**.

i) Select **lblProductName** and update its **Size** property of to **14**.

j) Update the **X** property of **lblProductName** to **100** and the **Y** property to **12**.

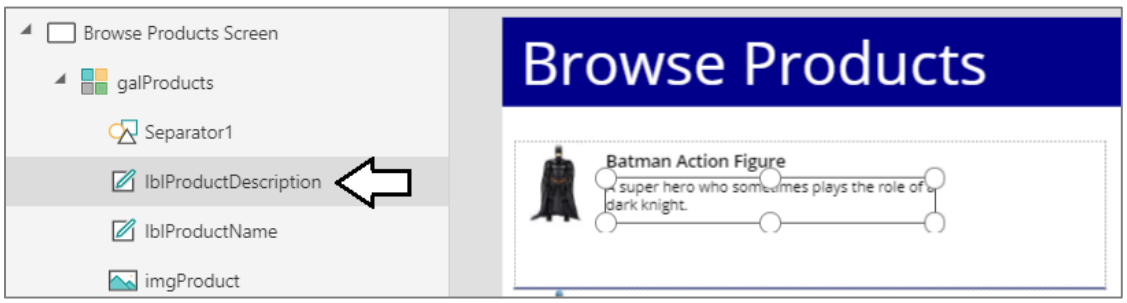
k) Update the **Width** property **lblProductName** to **356**.



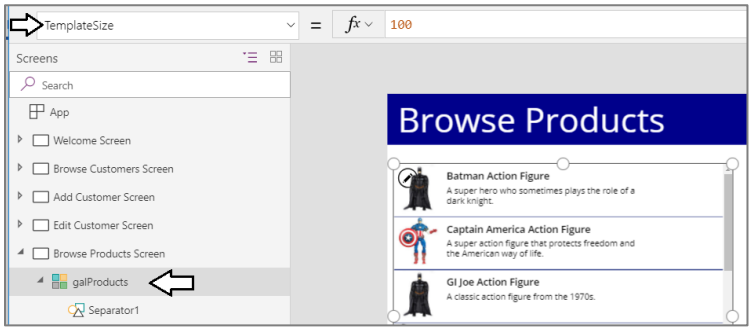
You should notice that the control named **lblProductDescription** automatically moves whenever you move **lblProductName**. This happens because the gallery template automatically includes dynamic expressions for the **Width**, **X** and **Y** properties of the subtitle label based on the **Width**, **X** and **Y** properties of the title control lab.

l) Select **lblProductDescription** and update its **Size** property to **12**.

m) Update the **Height** property of **lblProductName** to **48**.

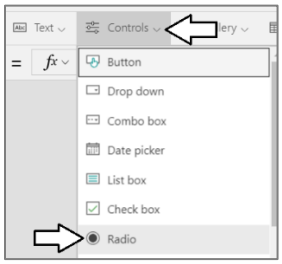


n) Update the height of the item template by selecting **galProducts** and updating the **TemplateSize** property to **100**.



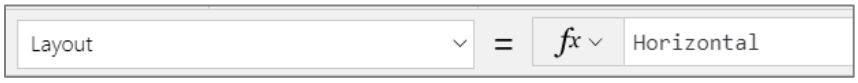
7. Add a set of radio buttons that allow the user to filter products based on product category.

a) With the **Browse Products Screen** select, using the **Controls** menu to add a new **Radio** control.



b) Rename the **Radio** control to **radProductFilter**.

c) Set the **Layout** property of **radProductFilter** to **Horizontal**.



d) Set the **Fill** property of **radProductFilter** to **LightGray**.

e) Update the **Width** property of **radProductFilter** using the following expression**.**

```
'Browse Customers Screen'.Width / 2
```

f) Position **radProductFilter** all the way to the left and just below the top nav banner as shown in the following screenshot.

g) Update the **BorderColor** property of **radProducts** to **Black** and the **BorderThickness** to **1**.

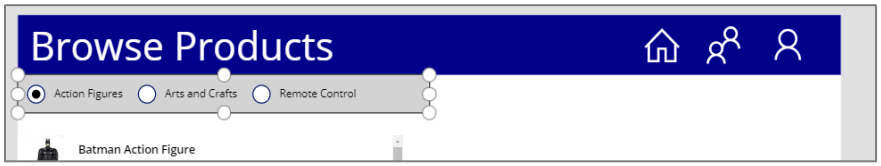
h) Update the **Items** property of **radProducts** using the following expression.

```
Distinct(Products, Category)
```

i) Update the **Default** property of **radProducts** using the following expression.

```
First(Distinct(Products, Category)).Result
```

j) Now **radProductFilter** should display three product categories and that the first category should be selected by default.



8. Update the Items property of **galProducts** to filter based on the category selected in **radProductFilter**.

a) Select **galProducts** and update its **Items** property using the following expression.

```
Filter(Products, Category=radProductFilter.SelectedText.Value)
```

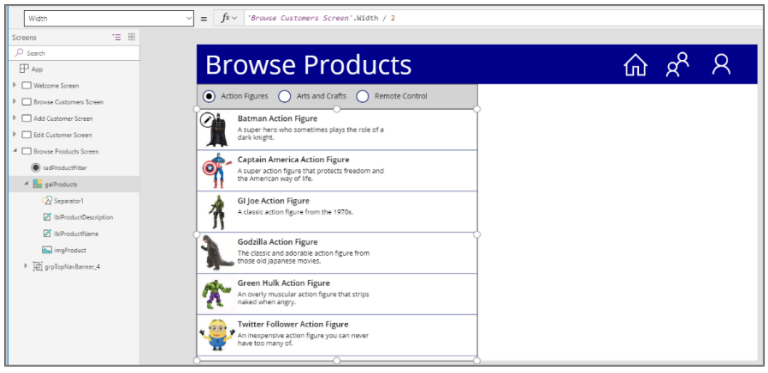
The items display in **galProducts** should now be filtered by the selected item in **radProductFilter**.

a) Update the **Width** property of **galProducts** using the following expression**.**

```
'Browse Customers Screen'.Width / 2
```

b) Update the **BorderColor** property of **galProducts** to **Black** and the **BorderThickness** to **1**.

c) Reposition **galProducts** underneath **radProductFilter** and expand its height to take up the remainder of the screen.

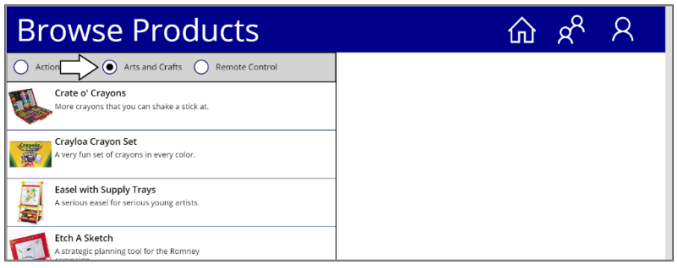


9. Test the **Browse Products Screen** to verify the product category filtering is working correctly.

a) Select the **Browse Products Screen** in the left tree view.

b) Click the Play button in the upper to start the app.

c) Once the **Browse Product Screen** has loaded, select **Arts and Crafts** to filter products by that category.



d) Select **Remote Control** to filter products by that category.

e) Once you have tested the filtering behavior, stop the app from running and return to edit mode in PowerApps Studio.

Exercise 2: Implement Shopping Cart Collection for Ordering Products

In this exercise, you will extend the **Browse Products Screen** to allow the user to add products to a shopping cart.

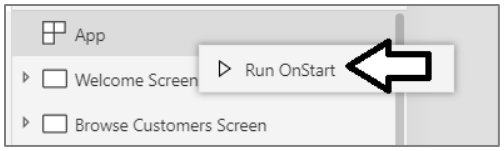
1. Add an expression to the **App OnStart** property to load product data into a new collection named **colProduct**.
   a) Select the **App** node in Tree view on the left and add the following expression to the **OnStart** property.

```
ClearCollect(colProducts, AddColumns(Products, "Quantity", 1))
```

   b) Your formula bar should match the following screenshot.

The reason for adding the **Quantity** column to **colProducts** is to enable adjusting the quantity on a product-by-product basis.

   c) Right-click the **App** node in tree view and then select the **Run OnStart** command to populate the **colProducts** collection.



You will now update **galProducts** so retrieves items from **colProducts** instead of directly from the **Products** data course.

2. Update the **Items** property of **galProducts** to retrieve items from **colProducts**.
   a) Select **Browse Products Screen** in Tree view on the left.
   b) On **Browse Products Screen**, select **galProducts** and inspect the **Items** property which is retrieving product items directly from the **Products** data source.

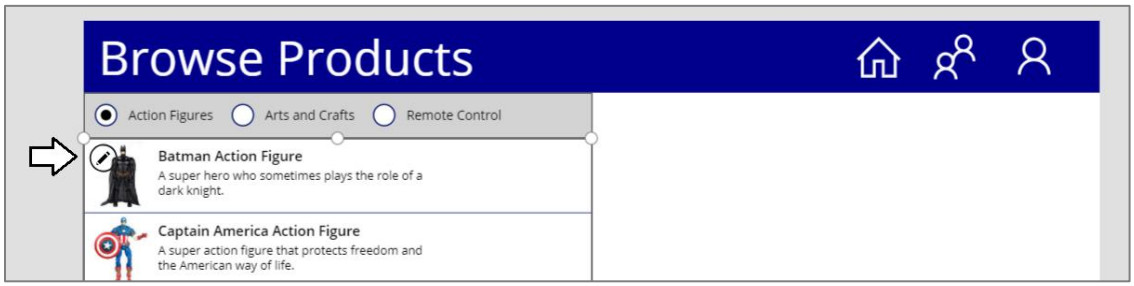
```
Filter(
    Products,
    Category = radProductFilter.Selected.Value
)
```

c) Update the expression for the **Items** property of **galProducts** by replacing **Products** with **colProducts**.

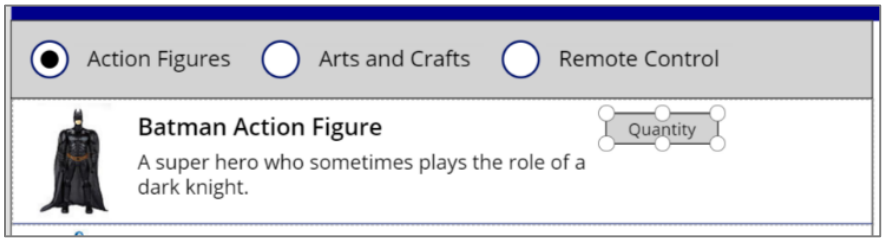
```
Filter(
    colProducts,
    Category = radProductFilter.Selected.Value
)
```

galProducts should work as before now that it is retrieving data from the **colProducts** collection instead of the **Products** data source.

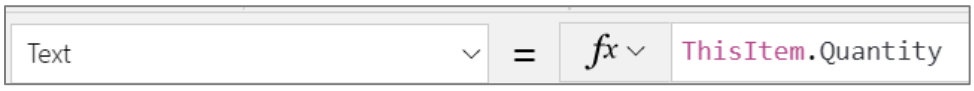
3. Add a **Quantity** caption label control to the product item template.
   a) Click on the pen icon of **galProducts** to edit the gallery's item template.



   b) Add a new **Label** control into the item template and rename it to **lblQuantityCaption**.
   c) Set the **Text** property of **lblQuantityCaption** to **"Quantity"**.
   d) Set the **X** property of **lblQuantityCaption** to **472** and the **Y** property to **12.**
   e) Set the **Width** property of **lblQuantityCaption** to **88** and the **Height** property to **32.**
   f) Set the **Fill** property of **lblQuantityCaption** to **LightGray**.
   g) Set the **BorderColor** property of **lblQuantityCaption** to **Black** and the **BorderThickness** to **1**.
   h) Set the **Size** property of **lblQuantityCaption** to **10**.
   i) Set the **Align** property of **lblQuantityCaption** to **Center**.
   j) The item template of **galProducts** should now match the following screenshot.



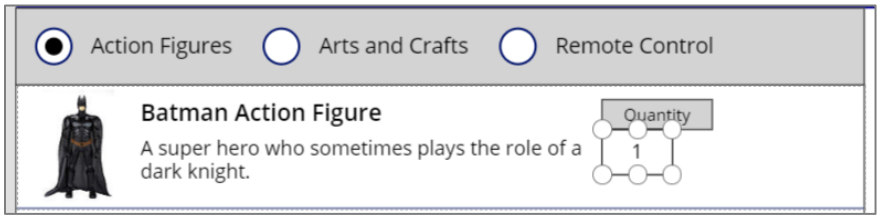
4. Add a second label to display the quantity value for the current product.
   a) Add a new **Label** control into the item template of **galProducts** and rename it to **lblQuantity**.
   b) Set the **Text** property of **lblQuantityCaption** to **ThisItem.Quantity**.



   c) Set the **X** property of **lblQuantity** to **472** and the **Y** property to **44.**
   d) Set the **Width** property of **lblQuantityCaption** to **56** and the **Height** property to **40.**
   e) Set the **Fill** property of **lblQuantityCaption** to **LightGray**.
   f) Update the **BorderColor** property of **lblQuantityCaption** to **Black** and the **BorderThickness** to **1**.
   g) Update the **Size** property of **lblQuantityCaption** to **12**.

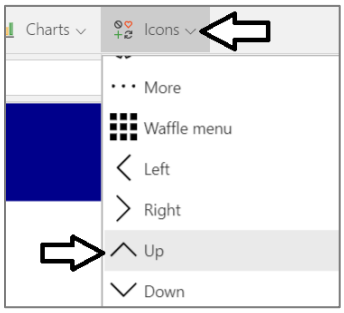
h) Update the **Align** property of **lblQuantityCaption** to **Center**.

i) The item template of **galProducts** should now match the following screenshot.



5. Add an icon button to increment the quantity for the current product.

a) Add a new **Up** icon control into the item template of **galProducts** and rename it to **icoIncrementQuantity**.



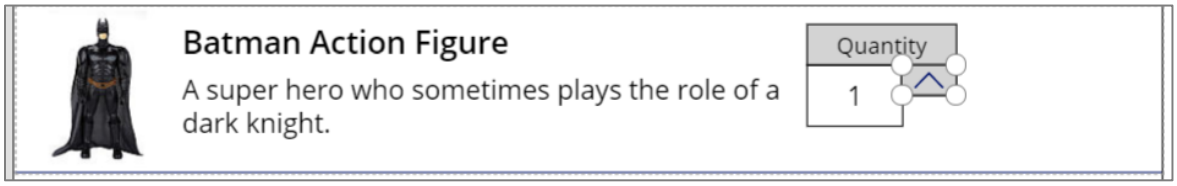
b) Set the **X** property of **icoIncrementQuantity** to **528** and the **Y** property to **44.**

c) Set the **Width** property of **icoIncrementQuantity** to **32** and the **Height** property to **20.**

d) Set the **Fill** property of **icoIncrementQuantity** to **LightGray**.

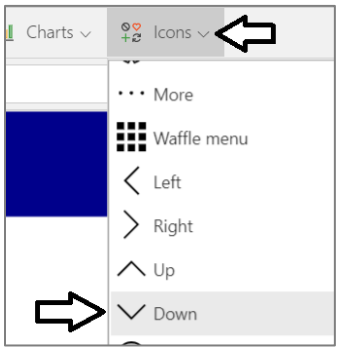
e) Update the **BorderColor** property of **icoIncrementQuantity** to **Black** and the **BorderThickness** to **1**.

f) The item template of **galProducts** should now match the following screenshot.



6. Add an icon button to increment the quantity for the current product.

a) Add a new **Down** icon control into the item template of **galProducts** and rename it to **icoDecrementQuantity**.



b) Set the **X** property of **icoDecrementQuantity** to **528** and the **Y** property to **64.**

c) Set the **Width** property of **icoDecrementQuantity** to **32** and the **Height** property to **20.**

d) Set the **Fill** property of **icoDecrementQuantity** to **LightGray**.

e) Update the **BorderColor** property of **icoDecrementQuantity** to **Black** and the **BorderThickness** to **1**.

f) The item template of **galProducts** should now match the following screenshot.

7. Add a button that allows the user to add products to the shopping cart.

a) Add a new button control to the item template and rename it o **btnAddToCart**..

b) Set the **X** property of **btnAddToCart** to **576** and the **Y** property to **12.**

c) Set the **Width** property of **btnAddToCart** to **96** and the **Height** property to **74.**

d) Set the **Size** property of **btnAddToCart** to **10**.

e) Update the **BorderColor** property of **btnAddToCart** to **Black** and the **BorderThickness** to **1**.

f) Update the **FocusedBorderThickness** to **1**.

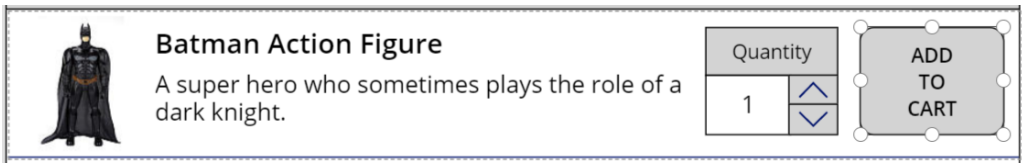
g) Update the **Fill** property of **btnAddToCart** to **LightGray** and the **Color** property to **Black**.

h) Update the **Text** property of **btnAddToCart** to the following expression.

```
"ADD
TO
CART"
```

In order to add a line break into the formula bar in PowerApps Studio, hold down the **SHIFT** key and then press **ENTER**.

i) The item template should now match the following screenshot.

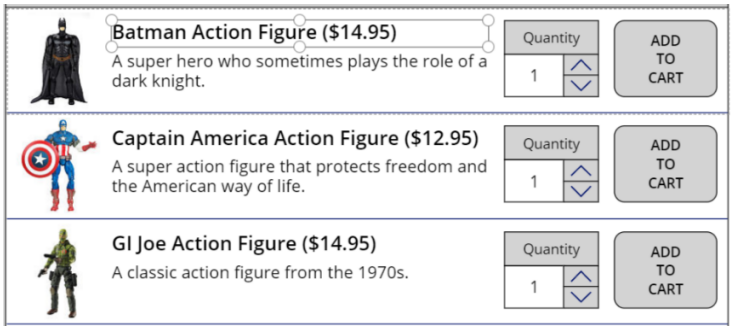


8. Add the product list price to the product label.

a) Update the **Text** property of **lblProductName** to the following expressions.

```
ThisItem.Product & " (" & Text(ListPrice, "$#,##0.00") & ")"
```

b) The item template should now show the formatted product list price after the product name.



9. Add behavior to increment and decrement the **Quantity** for a product.

a) Add an expression to the **OnSelect** property of **icoIncrementQuantity** to increment the **Quantity** for the current product.

```
Patch(colProducts, galProducts.Selected, {Quantity: Quantity + 1})
```

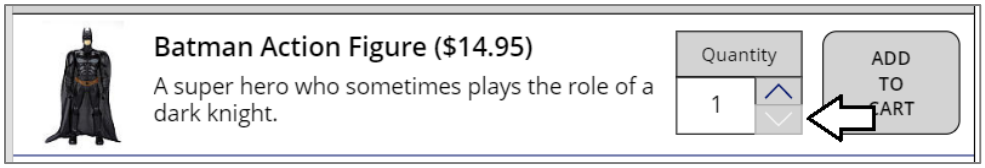
b) Add an expression to the **OnSelect** property of **icoDecrementQuantity** to decrement the **Quantity** for the current product.

```
Patch(colProducts, galProducts.Selected, {Quantity: Quantity - 1})
```

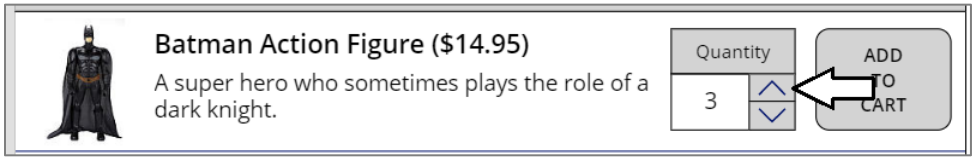
c) Update the **DisplayMode** property of **icoDecrementQuantity** to disable the control unless **Quantity** is greater than 1.

```
If(ThisItem.Quantity > 1, DisplayMode.Edit, DisplayMode.Disabled)
```

d) Run the application to test out your changes. You should see the decrement icon is initially disabled.



e) Click the increment button and verify that the action increments **Quantity** for the current product.



f) Click the decrement button and verify that the action decrements **Quantity** for the current product.

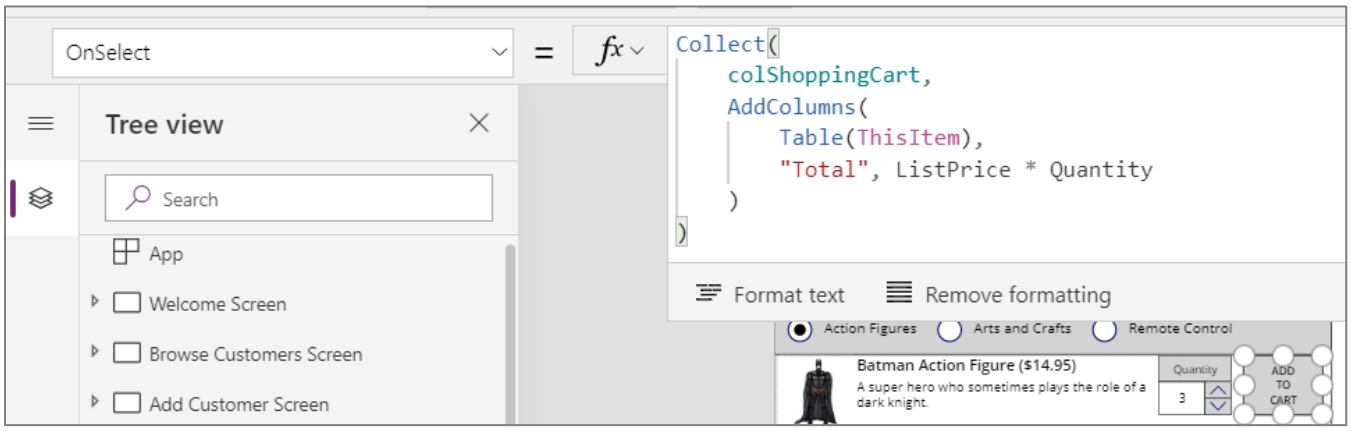
g) Once you are done with your testing, stop the app from running and return to the editor.

10. Add an expression to the **OnSelect** property of **btnAddToCart** to add an entry into a collection named

a) Add the following expression into the **OnSelect** property of **btnAddToCart**.

```
Collect(
    colShoppingCart,
    AddColumns(
        Table(ThisItem),
        "Total", ListPrice * Quantity
    )
)
```

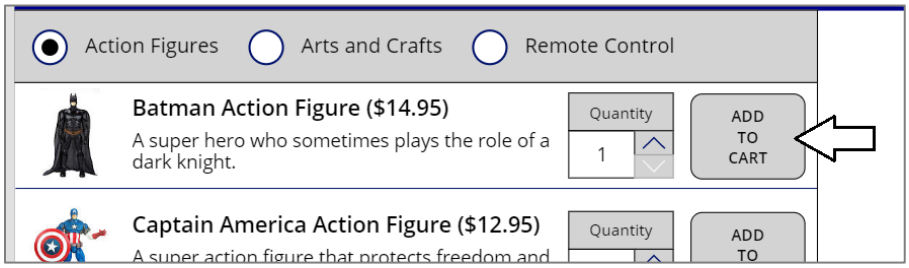
b) Your screen should match the following screenshot.



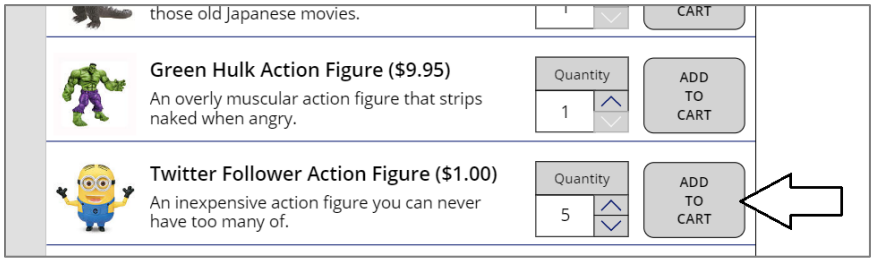
11. Test out the app by adding a few items to the shopping cart.

a) Select the **Browse Products Screen** in the left tree view and then start the app.

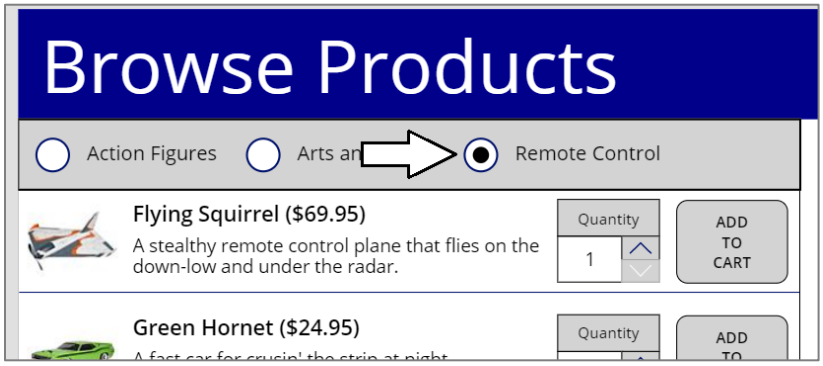
b) Change the **Quantity** for the **Batman Action Figure** to **1** and then click the **ADD TO CART** button.



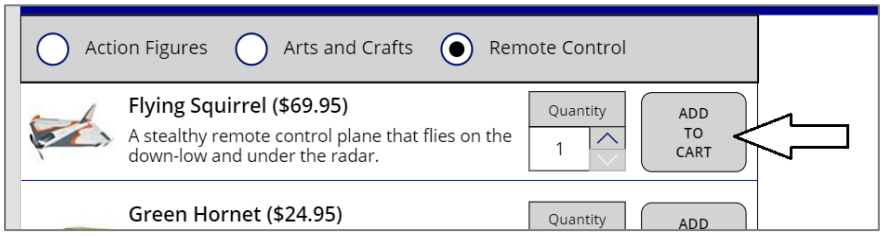
c) Update the **Quantity** for the **Twitter Follower Action Figure** to **5** and then click **ADD TO CART**.



d) Click the **Remote Control** category to change the filter on **galProducts**.



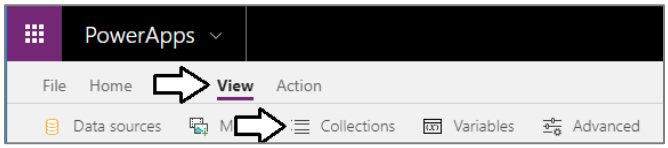
e) Click the **ADD TO CART** button to add the **Flying Squirrel**.



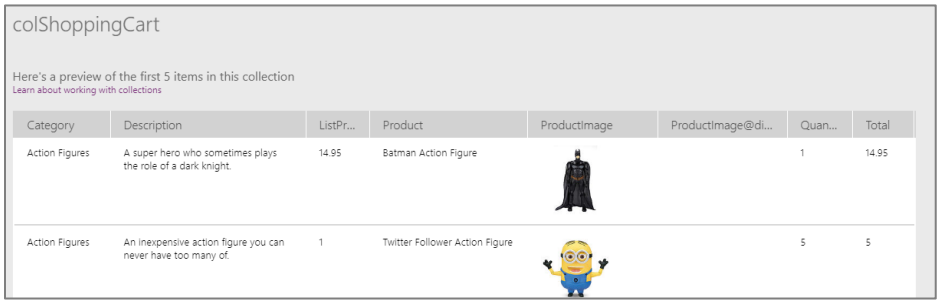
f) Stop the app from running and return to the PowerApps Studio editor.

12. Inspect what's inside the collection named **colShoppingCart**.

a) Activate the **View** tab in the ribbon and then click the **Collections** button.



b) You should now be able to see the data inside **colShoppingCart** which contains items with the properties from the **Products** table and additionally the extra two columns named **Quantity** and **Total**.



If you want, you could rewrite the expression for the **OnSelect** property of **btnAddToCart** to remove unnecessary shopping cart item properties such as **Description** and **ProductImage**. This step is not included in this lab to reduce complexity.

13. Add a label to display the current customer.

a) Add a new label to the **Browse Product Screen** and rename it to **lblCurrentCustomer**.

b) Set the **Size** property of **lblCurrentCustomer** to **18**.

c) Set the **Fill** property of **lblCurrentCustomer** to **LightBlue**.

d) Set the **PaddingLeft** property with **lblCurrentCustomer** to **10.**

e) Set the **Height** property of **lblCurrentCustomer** to **60**.

f) Set the **X** property of **lblCurrentCustomer** to the following formula.

```
'Browse Products Screen'.Width / 2
```

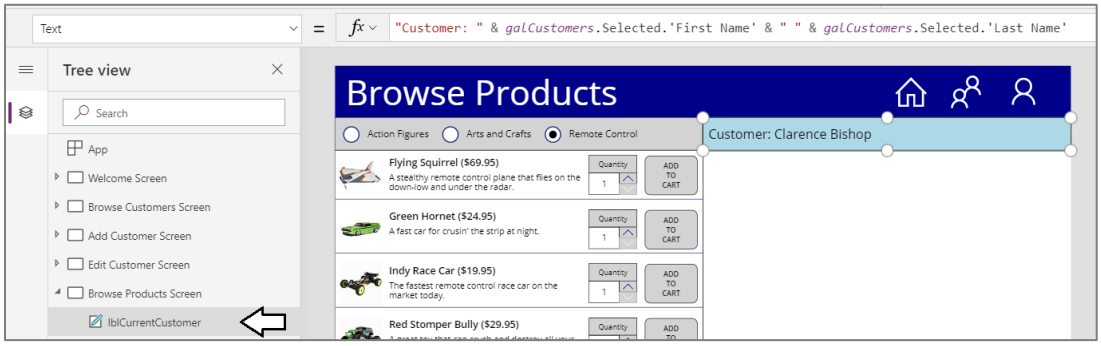
g) Set the **Width** property of **lblCurrentCustomer** to the following formula.

```
'Browse Products Screen'.Width / 2
```

h) Set the **Text** property of **lblCurrentCustomer** to the following expression.

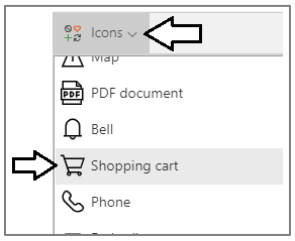
```
"Customer: " & galCustomers.Selected.'First Name' & " " & galCustomers.Selected.'Last Name'
```

i) The **Browse Products Screen** with **lblCurrentCustomer** should now match the following screenshot.



14. Update the **Browse Customers Screen** by adding a button to select a customer and navigate to the **Browse Products Screen**.
    a) Select the **Browse Customers Screen** in the left tree view.
    b) Select **galCustomers** and then click the pen icon button to edit the item template.

    c) Add a new **Shopping Cart** icon to the item template and rename it to **icoBrowseProducts**.

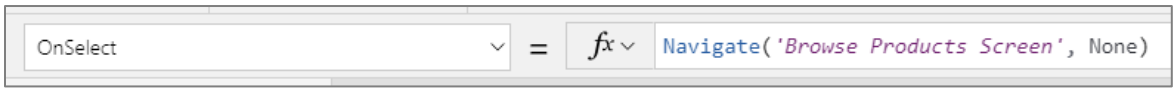


    d) Resize and reposition **icoBrowseProducts** to the left of the other two icons as shown in the following screenshot.

    e) Update the **OnSelect** property of **icoBrowseProducts** with the following expression.

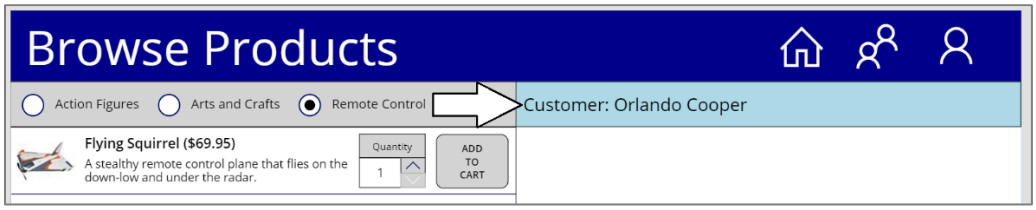
```
Navigate('Browse Products Screen', ScreenTransition.None)
```

    f) The formula bar should match the following screenshot.



15. Run the app to test your changes.
    a) With the **Browse Customers Screen** selected in the left tree view, start the app.
    b) Click on the shopping cart button for a specific customers.

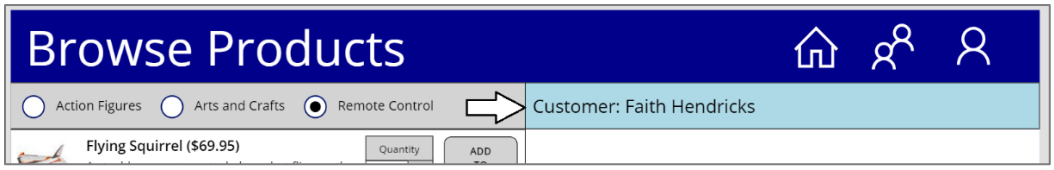
c) The app should navigate the **Browse Products Screen** and display the name of that customer.



d) Click the **People** button in the top nav bar to navigate back to the **Browse Customers Screen**.

e) Click the Shopping Cart button to select a different customers.

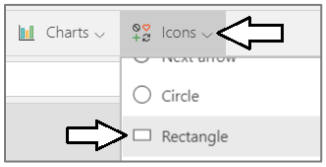
f) The app should navigate the **Browse Products Screen** and display the name of that customer.



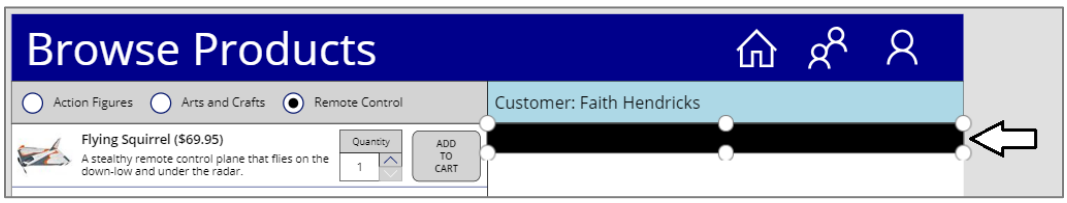
You have now implemented the behavior which allows the user to select a specific customer. Next, you are going to add a new gallery to display shopping cart items on the **Browse Products Screen**. However, before creating the gallery you will first create a set of labels to serve as column headers for the data displayed in the shopping cart gallery.

16. Add a set of labels to create the shopping cart column headers.

a) Add a new **Rectangle** icon to **Browse Products Screen** and rename it to **rectShoppingCartHeader**.



b) Position **rectShoppingCartHeader** underneath **lblCurrentCustomer** and give it a height of **42**.



c) Add a new label named **lblShoppingCartHeaderQuantity** and set its **Text** property to **No**.

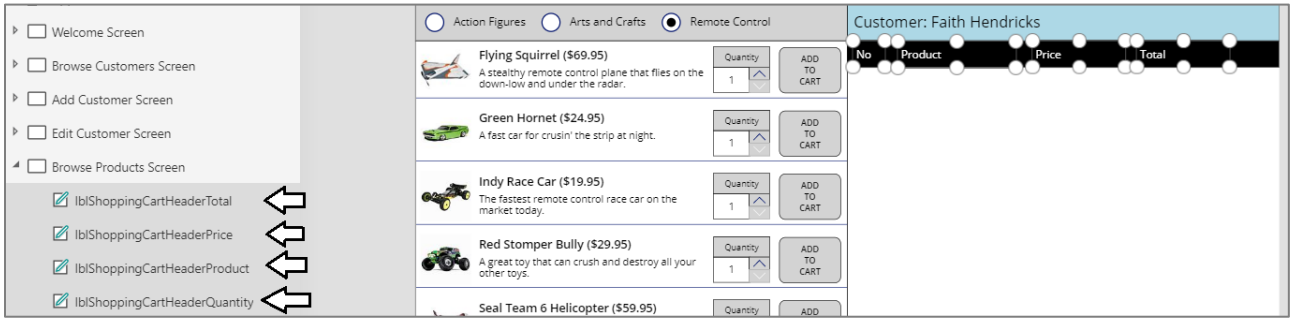
d) Add a new label named **lblShoppingCartHeaderProduct** and set its **Text** property to **Product**.

e) Add a new label named **lblShoppingCartHeaderPrice** and set its **Text** property to **Price**.

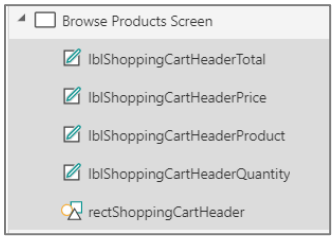
f) Add a new label named **lblShoppingCartHeaderQuantity** and set its **Text** property to **Total**.

g) Update the **Color** of all four labels to **White** and the **Size** to **13**.

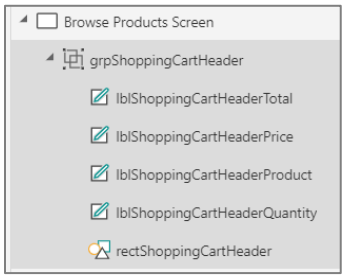
h) Reposition the four labels on top of **rectShoppingCartHeader** as shown in the following screenshot.



i) Select the four label along with **rectShoppingCartHeader** in tree view and then select the **Group** command to group them.

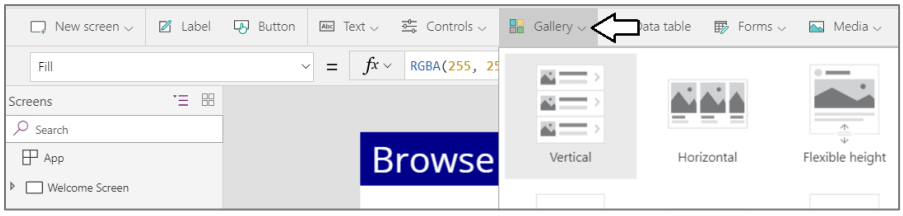


j) Once you have created the new group, rename the group to **grpShoppingCartHeader**.



17. Add a new gallery the **Browse Products Screen** for displaying the shopping cart.

a) Using the **Gallery** menu on the **Insert** tab, add a new **Vertical** gallery to the **Browse Products Screen**.

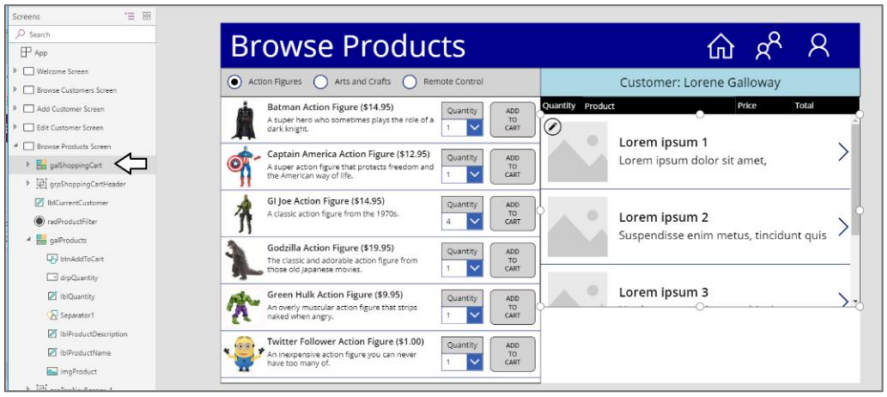


b) Rename the new gallery to **galShoppingCart** and reposition it just below **grpShoppingCartHeader**.

c) Set the **X** property of **galShoppingCart** to **683**.

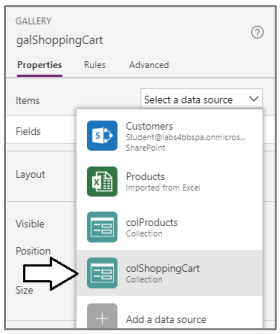
d) Set the **Width** property of **galShoppingCart** to **683**.

e) The **Browse Products Screen** should match the following screenshot.

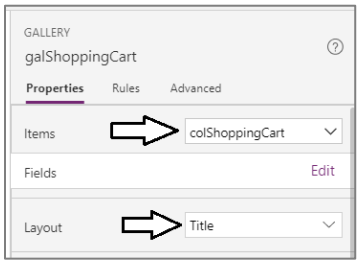


f) With **galShoppingCart** selected in the left tree view, drop down the **Select a data source** menu in the **Properties** pane.

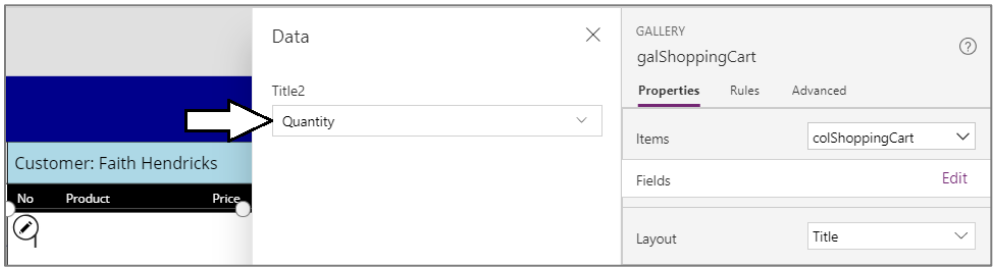
g) Select the collection named **colShoppingCart** as the source for **Items** property of **galShoppingCart**.



h) After assigning **colShoppingCart** to the **Items** property, update the **Layout** property to **Title**.

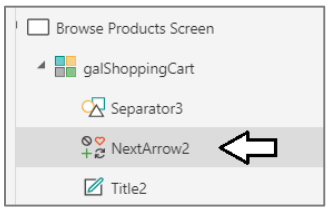


i) Once you have set the **Layout** property, click the **Edit** link for the **Fields** property and change the field for **Title2** to **Quantity**.

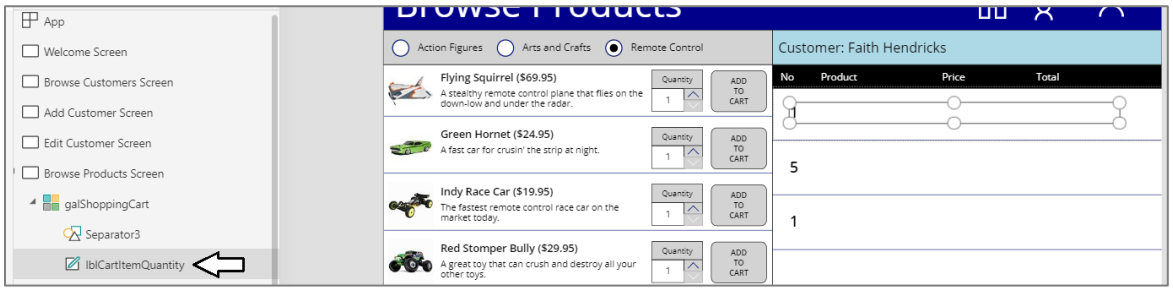


18. Design the item template for **galShoppingCart**.

a) Inside the item template for **galShoppingCart**, locate and delete the icon control named **NextArrow2**.



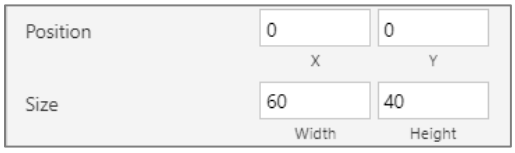
b) Rename the control named **Title2** to **lblCartItemQuantity**.



c) Change the **Size** property of **lblCartItemQuantity** to **16**.

d) Update the **X** and **Y** properties of **lblCartItemQuantity** to **0** and the **Align** property to **Center**.

e) Update the **Width** property of **lblCartItemQuantity** to **60** and the **Height** property to **40**.



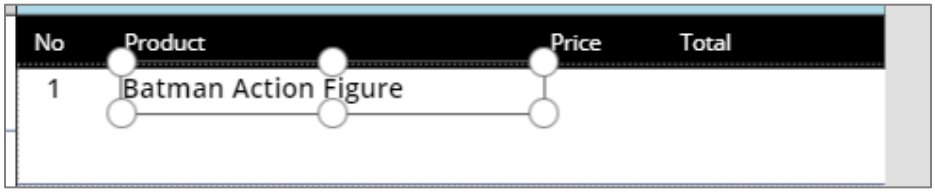
f) Update the **VerticalAlign** property of **lblCartItemQuantity** to **Middle**.

g) The item template for **galShoopingCart** should match the following screenshot.

h) Use a copy-and-paste operation to create a copy of the control named **lblCartItemQuantity**.

i) Rename the new copy of the label control to **lblCartItemProduct**.

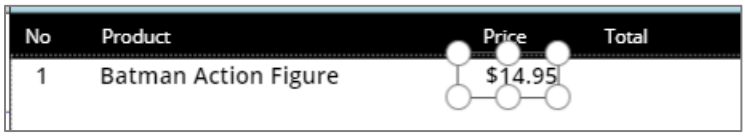
j) Change the **Text** property of **lblCartItemProduct** to **ThisItem.Product** and the **Align** property to **Left.**



k) Use another copy-and-paste operation to make another label and rename it to **lblCartItemPrice**.

l) Update the **Align** property of **lblCartItemPrice** to **Right** and the **Text** property using the following expression.

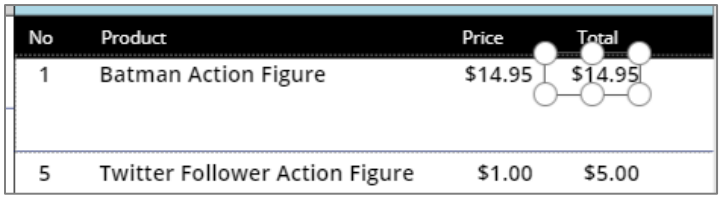
```
Text(ThisItem.ListPrice,"$#,##0.00")
```



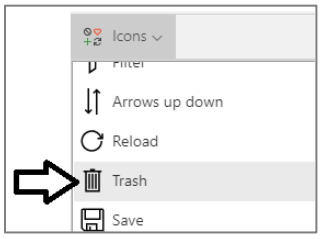
m) Use another copy-and-paste operation to make another label and rename it to **lblCartItemTotal**.

n) Update the **Align** property of **lblCartItemTotal** to **Right** and the **Text** property using the following expression.

```
Text(ThisItem.Total,"$#,##0.00")
```

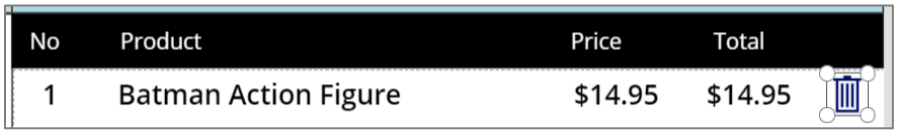


o) Add a new **Trash** icon to the **galShoppingCart** item template to provide a command for removing shopping cart items.



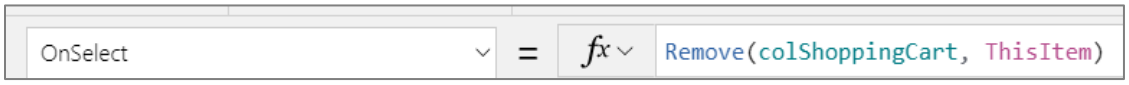
p) Rename the icon to **icoRemoveCartItem**.

q) Resize and reposition **icoRemoveCartItem** to match the following screenshot.

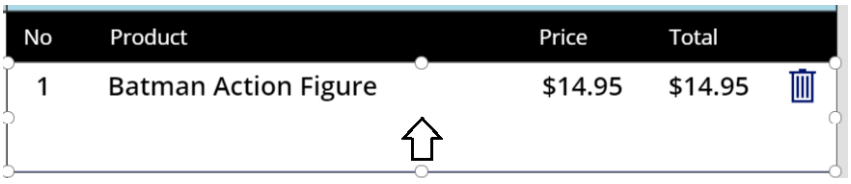


r) Update the **OnSelect** property of **icoRemoveCartItem** with the following expression to remove the current item.

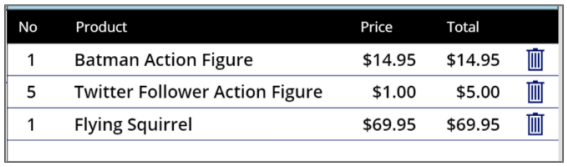
```
Remove(colShoppingCart, ThisItem)
```



s) Grab the bottom drag handle of the item template and move it up to reduce the **Template size** to **40**.

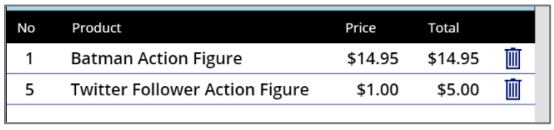


t) The shopping cart items displayed by **galShoppingCart** should now match the following screenshot.



19. Confirm that the user can delete a shopping cart item.
    a) With **Browse Products Screen** select in tree view, start up the app.
    b) Click on the button with the trash icon for the bottom item.

    c) You should be able to confirm that clicking the trash icon allows a user to remove a shopping cart item.

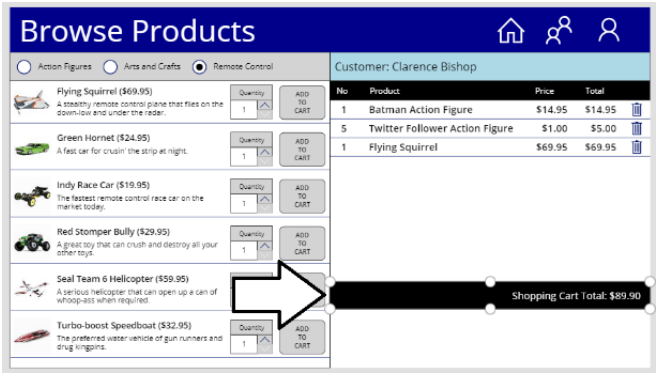


    d) Stop the app from running and return to the PowerApps Studio editor.

20. Create new label named **lblShoppingCartFooter**
    a) Add a new label and rename it to **lblShoppingCartFooter**.
    b) Set the **Fill** property to **Black** and the **Color** property to **White**.
    c) Set **Size** property to **16** and set the **Align** property to **Right**.
    d) Set the **PaddingRight** property to **20**.
    e) Update the **Text** property using the following formula.

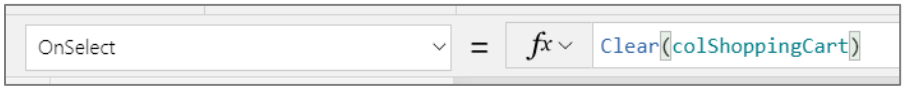
```
"Shopping Cart Total: " & Text(Sum(colShoppingCart, Total), "[$-en-US]$#,##0.00")
```

    f) Reposition **lblShoppingCartFooter** to match the layout shown in the following screenshot.

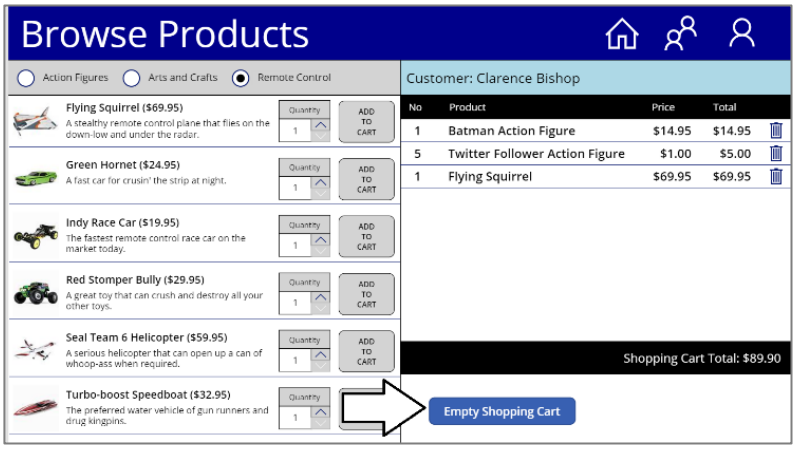


21. Add button which allows the user to empty the shopping cart.
    a) Add a button to the screen and rename it to **btnEmptyShoppingCart**.
    b) Update the **Text** property of **btnEmptyShoppingCart** to **Empty Shopping Cart**.
    c) Update the **OnSelect** property of **btnEmptyShoppingCart** using the following expression.

```
Clear(colShoppingCart)
```



    d) Reposition **btnEmptyShoppingCart** to match the layout shown in the following screenshot.



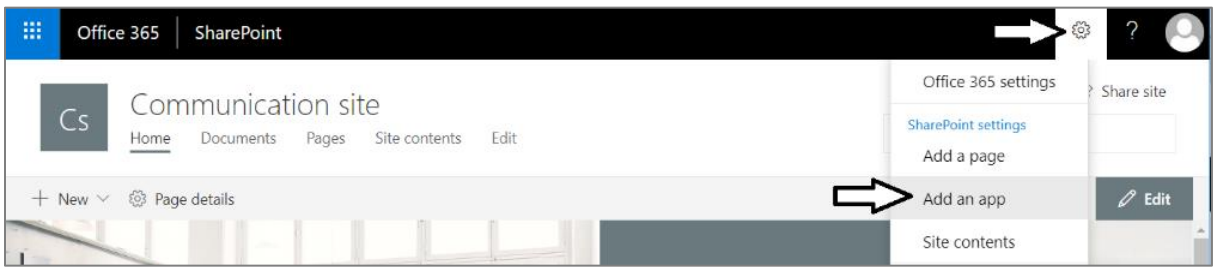
22. Test the behavior of **btnEmptyShoppingCart**.
    a) Start the app to display **Browse Products Screen** in run mode.
    b) Add a few items to the shopping cart.
    c) Click **btnEmptyShoppingCart** and verify that clicking the button clears the shopping cart by removing all items.

At this point, you have implemented behavior which allows to user to add and remove items build a shopping cart which will be used to submit an order. In the next exercise, you will create two lists in SharePoint Online to track orders and order details. In the exercise that follows, you will update the **Customer Ordering** canvas app to save order and shopping cart data into these two SharePoint lists.

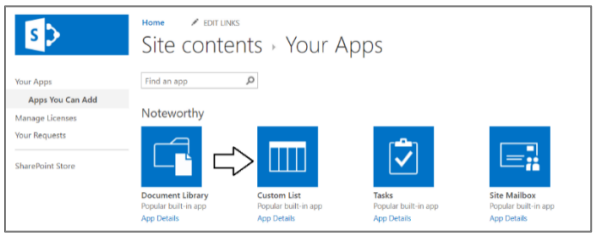
Exercise 3: Create SharePoint Lists for Orders and Order Details

In this exercise, you will create two new list in your SharePoint site named **Orders** and **OrderDetails**.

1. Create a new SharePoint custom list named **Orders**.
    a) Navigate to the SharePoint site in which you created the **Customers** list for the **Customer Ordering** Canvas app.
    b) Click on the gear icon and then click on **Add an app**

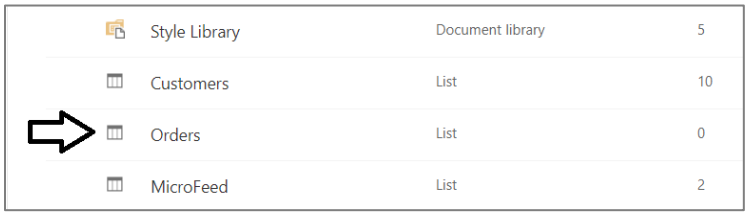


c) Click the **Custom List** tile to create a new custom list.

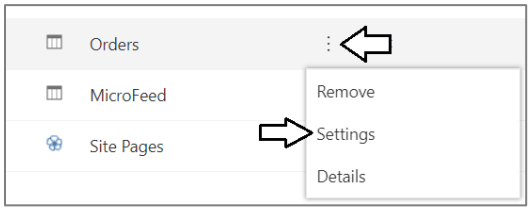


d) When prompted with the **Adding Custom List** dialog, add a **Name** of **Orders** and click **Create**.

e) Once the **Orders** list has been created, you should be able to locate this new list on the **Site contents** page.



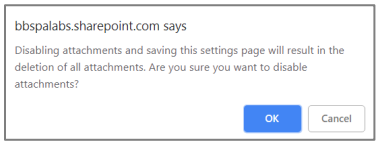
f) On the **Site contents** page, drop down the context menu to the right of the **Orders** list and then click **Settings**.



g) On the **Settings** page, click the **Advanced Settings** link to navigate to the **Advanced Settings** page.

h) On the **Advanced Settings** page, change the **Attachments** setting to **Disabled**.

i) When prompted with the warning about disabling attachments, click **OK** to continue.

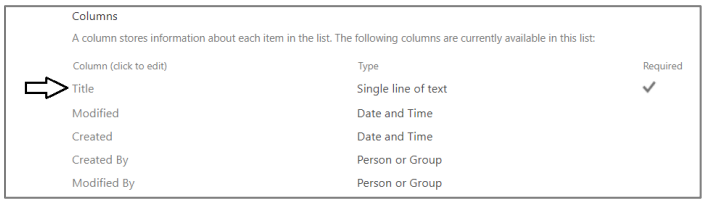


j) Scroll to the bottom of the **Advanced Settings** page and click **OK** to save your changes and return to the **Settings** page.

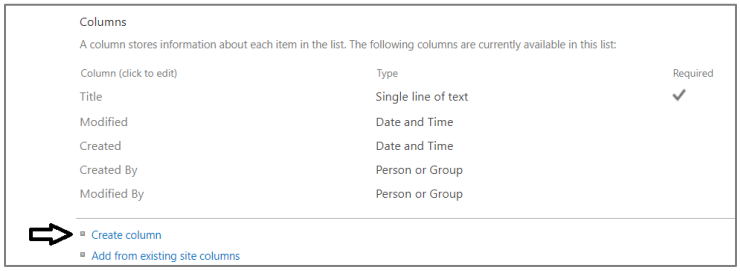
2. Configure the columns for the new **Orders** list.

a) In the **Setting** page, scroll down to the **Columns** section.

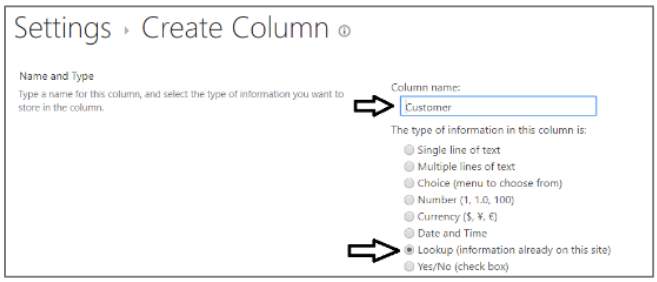
b) At this point, the **Columns** collection contains a field named **Title**.



c) Click on the **Create column** link to navigate to the **Create Column** page.

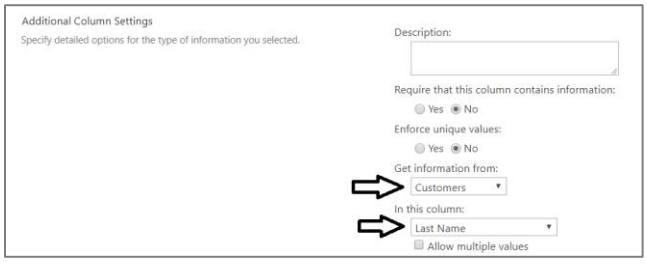


d) On the **Create Column** page, add a **Column name** of **Customer** and select the **Lookup** column type.

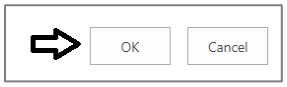


e) Move down on the **Create Column** page and locate the **Additional Column Settings** section.

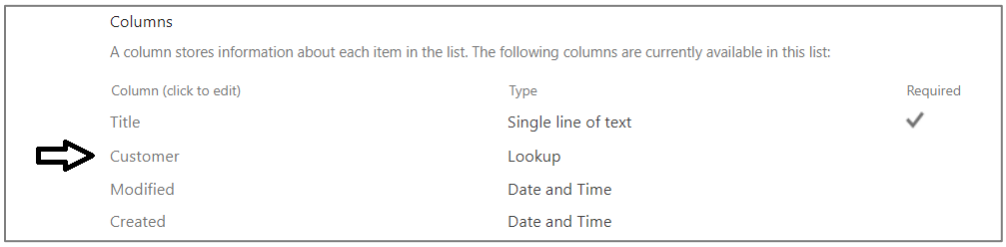
f) Set **Get information from** to **Customers** and **in this column** to **Last Name** as shown in the following screenshot.



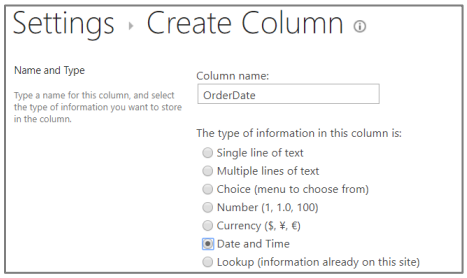
g) Move to the bottom of the **Create Column** page and click the **OK** button to create the new lookup column named **Customer**.



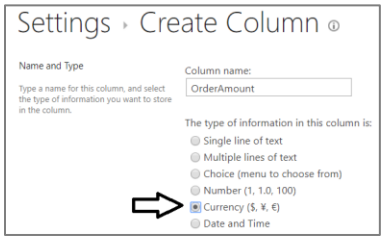
h) You should now be able to see the **Customer** column in the **Columns** collection of the **Orders** list.



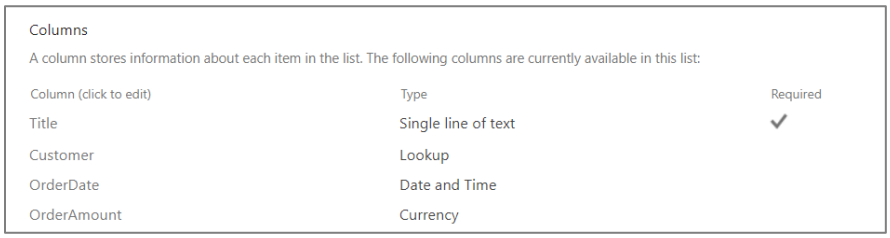
3. Add a new column named **OrderDate**.
   a) Click on the **Create column** link to navigate to the **Create Column** page.
   b) On the **Create Column** page, add a **Column name** of **OrderDate** and select the **Date and Time** column type.



   c) Move to the bottom of the **Create Column** page and click the **OK** button to create the new column named **OrderDate**.
4. Add a new column named **OrderAmount**.
   a) Click on the **Create column** link to navigate to the **Create Column** page.
   b) On the **Create Column** page, add a **Column name** of **OrderAmount** and select the **Currency** column type.

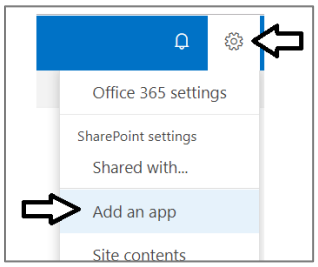


   c) Move to the bottom of the **Create Column** page and click the **OK** button to create the new column named **OrderDate**.
   d) You should now be able to see the **OrderDate** and **OrderAmount** columns in the **Columns** collection of the **Orders** list.

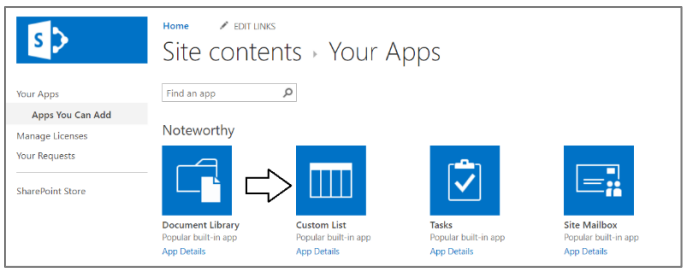


5. Create a new list named **OrderDetails**.

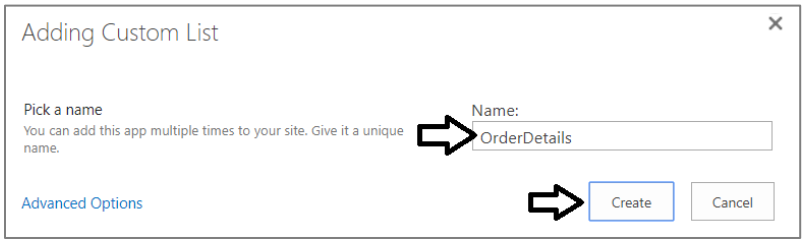
a) Click on the gear icon and then click on **Add an app**



b) Click the **Custom List** tile to create a new custom list.

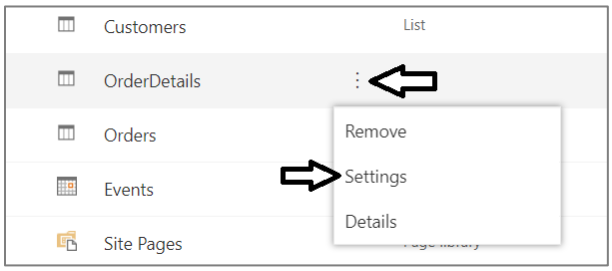


c) When prompted with the **Adding Custom List** dialog, add a **Name** of **OrderDetails** and click **Create**.



d) Once the **OrderDetails** list has been created, you should be able to locate this new list on the **Site contents** page.

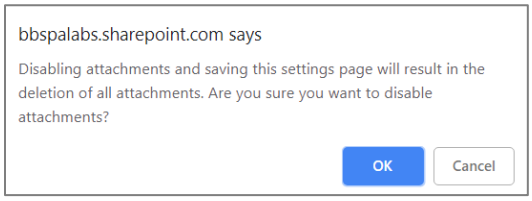
e) On the **Site contents** page, drop down the context menu to the right of the **OrderDetails** list and then click **Settings**.



f) On the **Settings** page, click the **Advanced Settings** link to navigate to the **Advanced Settings** page.

g) On the **Advanced Settings** page, change the **Attachments** setting to **Disabled**.

h) When prompted with the warning about disabling attachments, click **OK** to continue.



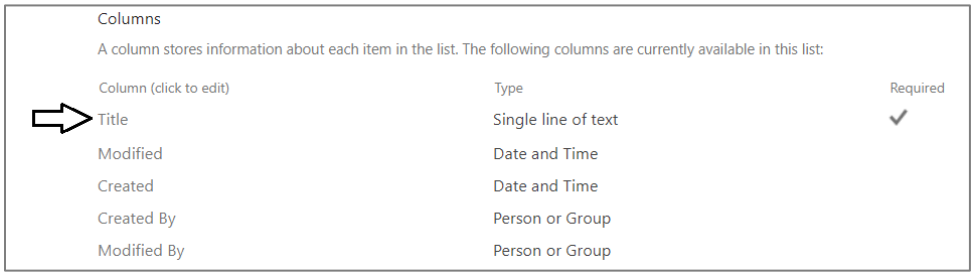
i) Scroll to the bottom of the **Advanced Settings** page and click **OK** to save your changes and return to the **Settings** page.

6. Configure the columns for the new **OrderDetails** list.

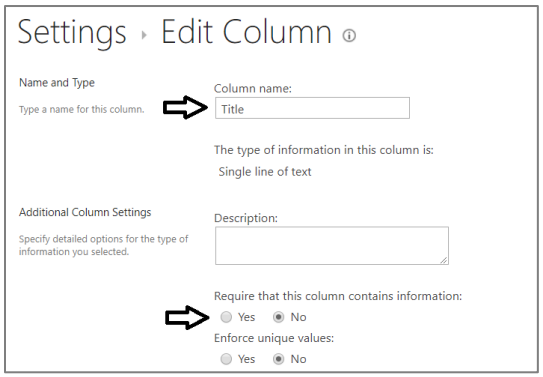
a) In the **Setting** page, scroll down to the **Columns** section.

b) At this point, the list contains a field named **Title**.

c) Click the link for the **Title** column to navigate to the **Edit Column** page.

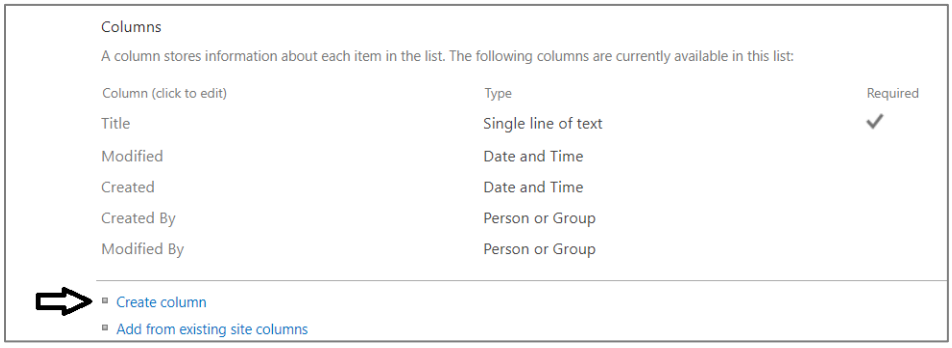


d) Select the **No** option for **Require that this column contains information**.

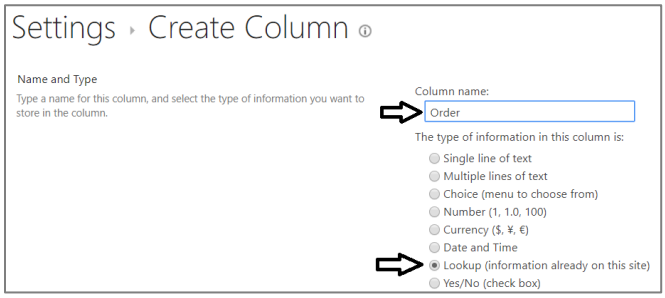


e) Click **OK** at the bottom of the **Edit Column** page to save your changes to the **Title** column.

f) Click on the **Create column** link to navigate to the **Create Column** page.

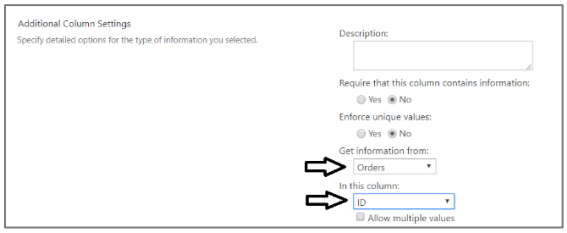


g) On the **Create Column** page, add a **Column name** of **Order** and select the **Lookup** column type.



h) Move down on the **Create Column** page and locate the **Additional Column Settings** section.

i) Set **Get information from** to **Orders** and **in this column** to **ID** as shown in the following screenshot.

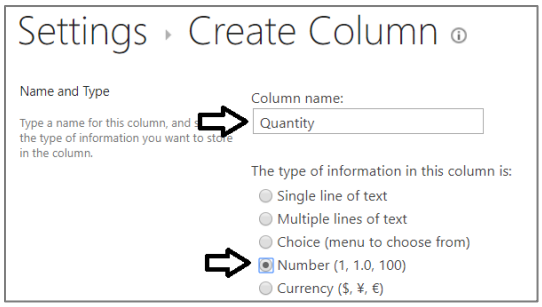


j) Move to the bottom of the **Create Column** page and click the **OK** button to create the new lookup column named **Order**.

7. Add a new column named **Quantity**.

a) Click on the **Create column** link to navigate to the **Create Column** page.

b) On the **Create Column** page, add a **Column name** of **Quantity** and select the **Number** column type.

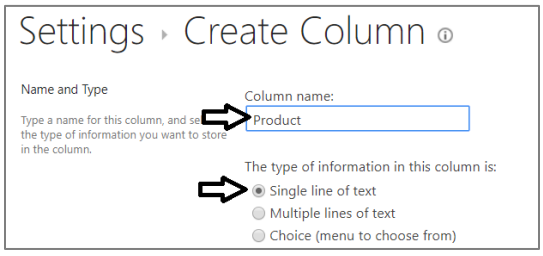


c) Move to the bottom of the **Create Column** page and click the **OK** button to create the new column named **Quantity**.

8. Add a new column named **Product**.

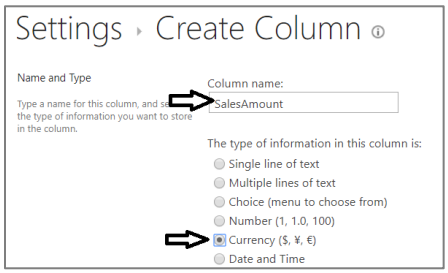
a) Click on the **Create column** link to navigate to the **Create Column** page.

b) On the **Create Column** page, add a **Column name** of **Product** and select the **Single line of text** column type.

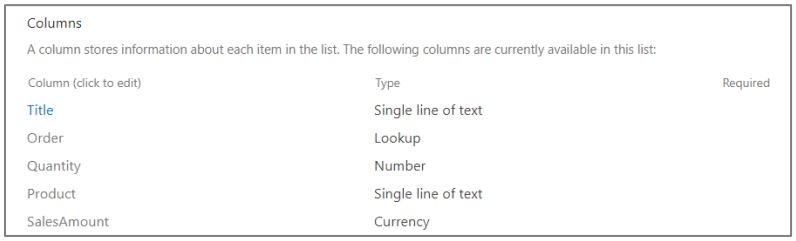


c) Move to the bottom of the **Create Column** page and click the **OK** button to create the new column named **Product**.

9. Add a new column named **SalesAmount**.
   a) Click on the **Create column** link to navigate to the **Create Column** page.
   b) On the **Create Column** page, add a **Column name** of **SalesAmount** and select the **Currency** column type.



   c) Move to the bottom of the **Create Column** page and click the **OK** button to create the new column named **Product**.
   d) You should now be able to see the **Order, Quantity, Product and SalesAmount** columns in the **Columns** collection.

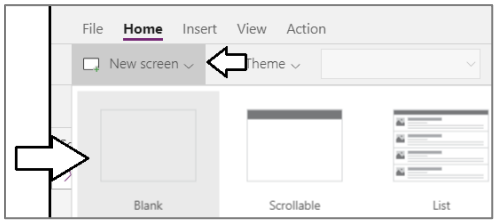


Now that you have created the lists you need in your SharePoint site, you will return to working on the **Customer Ordering** app in PowerApps Studio to add a new screen to submit orders and save order data and shopping cart data back to SharePoint.

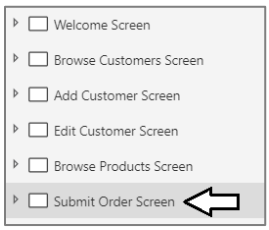
Exercise 4: Create the Submit Order Screen

In this exercise, you will extend the **Customer Ordering** app by adding the **Submit Order Screen** and implementing the behavior to save shopping cart data into the SharePoint lists you created in the previous exercise.

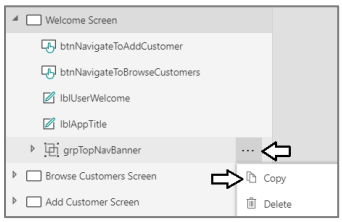
1. Return to the **Customer Ordering** canvas app in PowerApps Studio.
2. Create the **Submit Order Screen**.
   a) Use the **New screen** command from the **Home** tab to add a new **Blank** screen.



   b) Rename the new screen to **Submit Order Screen**.



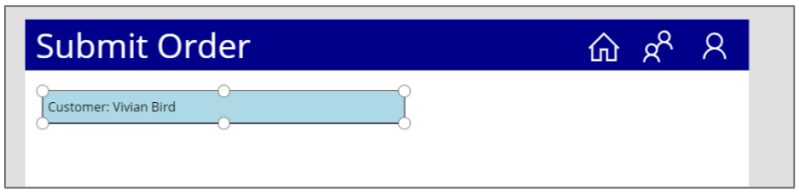
3. Copy and paste the group named **grpTopNavBanner** from the **Welcome Screen** to the **Submit Order Screen**.
   a) Expand the **Welcome Screen** in the left tree view.
   b) Drop down the context menu for **grpTopNavBanner** and select **Copy**.



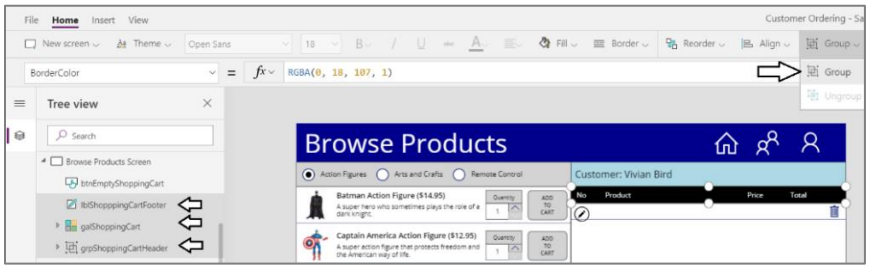
   c) Drop down the context menu for the **Submit Order Screen** and select **Paste**.
   d) Update the text on the Top Nav Banner to **"Submit Order"**.

4. Copy **lblCurrentCustomer** from the **Browse Product Screen** to the **Submit Order Screen**.
   a) From tree view of the left, select and copy **lblCurrentCustomer** from the **Browse Product Screen**.

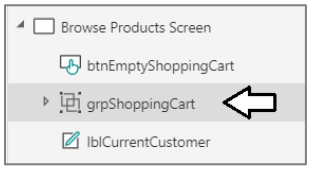
   b) From tree view on the left, select the **Submit Order Screen** and paste **lblCurrentCustomer** from the clipboard.



5. Copy the shopping cart controls from the **Browse Product Screen** to the **Submit Order Screen**.
   a) In tree view, select **grpShoppingCartHeader**, **galShoppingCart** and **lblShoppingCartFooter**.
   b) Use the **Group** command to add the selected controls into a single group.

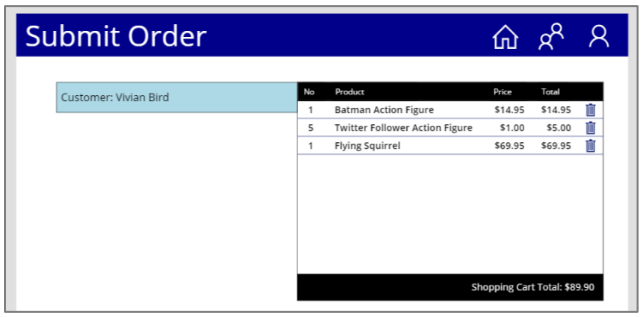


c) Rename the new group to **grpShoppingCart**.



d) Copy **grpShoppingCart** to the clipboard and then paste it into the **Submit Order Screen**.

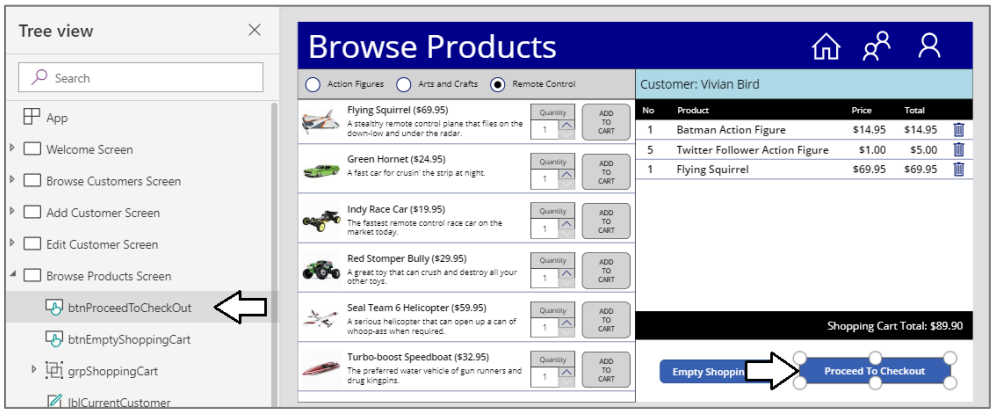
e) Rearrange the controls on the **Submit Order Screen** to match the layout of the following screenshot.



6. Add a new button to the **Browse Order Screen** to navigate to the **Submit Order Screen**.

a) Add a new button to the **Browse Products Screen** and rename it to **btnProceedToCheckOut**.

b) Position **btnProceedToCheckOut** in the bottom right of the screen as shown in the following screenshot.



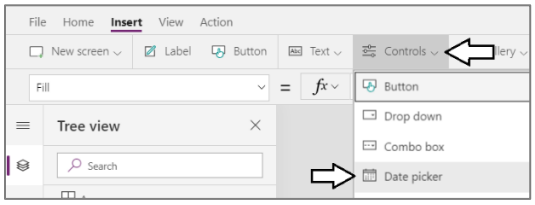
c) Update the **OnSelect** property of **btnProceedToCheckOut** with this expression to navigate to the **Submit Order Screen**.

```
Navigate('Submit Order Screen',ScreenTransition.None)
```

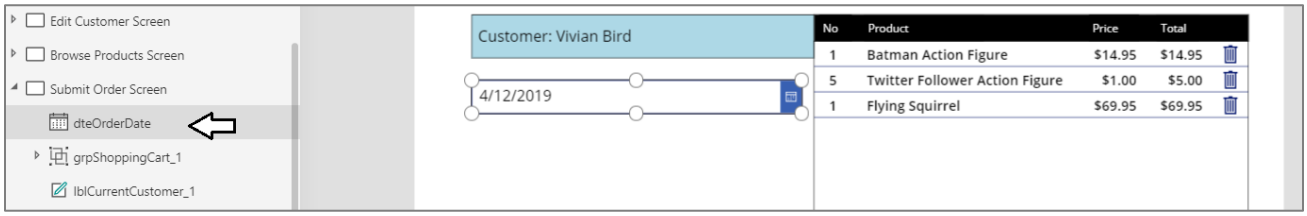
d) Run the app and click **btnProceedToCheckOut** to it allows the user to navigate to the **Submit Order Screen**.

7. Implement the behavior in the **Submit Order Screen** to save a new order records to the SharePoint list named **Orders**.

a) Add a new **Date picker** control to the **Submit Order Screen**.

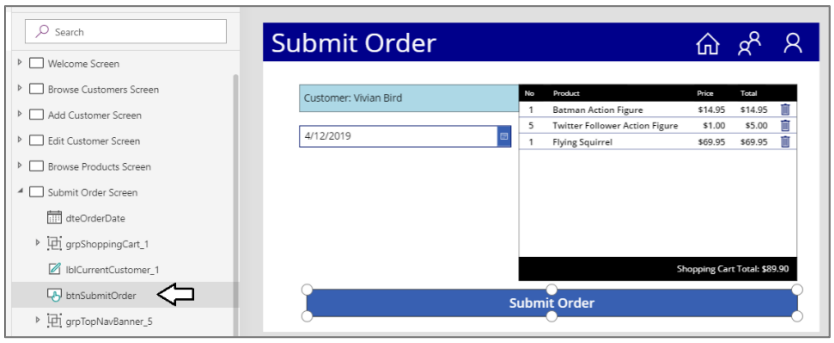


b) Rename the **Date picker** control to **dteOrderDate** and position it just below.



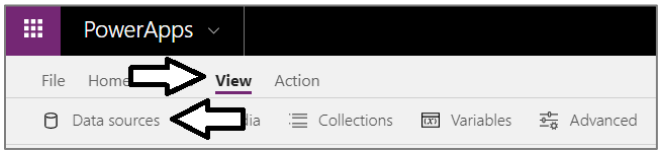
c) Add a new button to the bottom of **Submit Order Screen** and rename it to **btnSubmitOrder**.

d) Reposition **btnSubmitOrder** underneath the other controls as shown in the following screenshot.

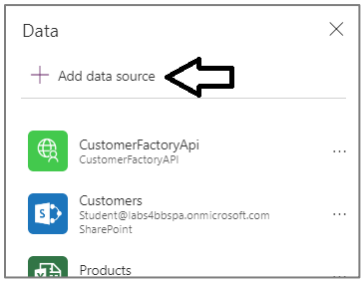


8. Add two new connections to the SharePoint lists named **Orders** and **OrderDetails**.

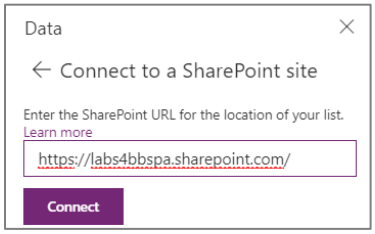
a) Select the **View > Data** sources command to display the **Data** pane.



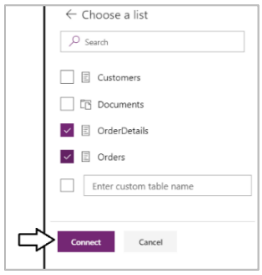
b) Click the **Add data source** link and then select the **SharePoint** connector.



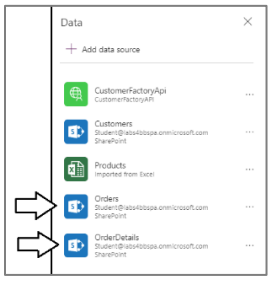
c) When prompted to **Connect to a SharePoint site**, add the URL to your SharePoint site and click **Connect**.



d) When prompted to **Choose a list**, select **Orders** and **OrderDetails** and then click **Connect**.



e) Your canvas app should now have connections to the **Orders** list and the **OrderDetails** list.



9. Add behavior to **btnSubmitOrder** to save order data to the SharePoint list named **Orders**.

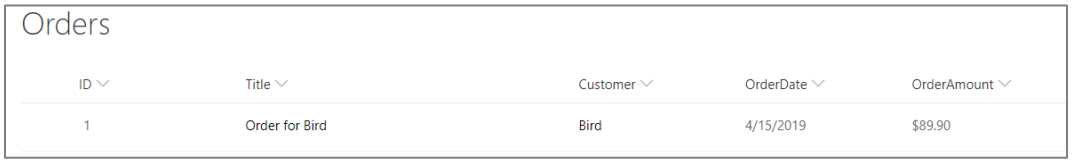
a) Update the **OnSelect** property of **btnSubmitOrder** with the following expression.

```
ClearCollect(
    colLastOrder,
    Patch(
        Orders,
        Defaults(Orders),
        {
            Title: "Order for " & galCustomers.Selected.'Last Name',
            OrderDate: dteOrderDate.SelectedDate,
            OrderAmount: Sum(colShoppingCart, Total),
            Customer: {
                '@odata.type': "#Microsoft.Azure.Connectors.SharePoint.SPListExpandedReference",
                Id: galCustomers.Selected.ID,
                Value: galCustomers.Selected.'First Name' & " " & galCustomers.Selected.'Last Name'
            }
        }
    )
);
```

You can see how this expression calls the **Patch** function to save order data to the SharePoint **Orders** list. The expressions also captures the record returned by the **Patch** function and adds it into a new collection named **colLastOrder**. After a user clicks the submit button to create a new **Order** record, you can query the record stored in **colLastOrder** to determine the **ID** of the new order.

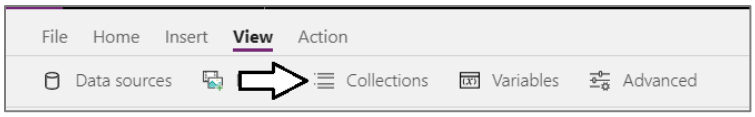
b) Test your work by starting the app and submitting an order, Before clicking the **Submit Order** button, make sure you have selected a customer and you have also added a few items into the shopping cart.

c) After submitting the order, navigate to the **Order** list in your SharePoint site and confirm a new list item has been added.

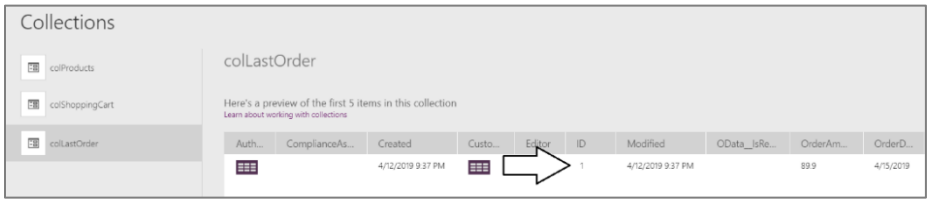


d) Return to the canvas app in PowerApps Studio.

10. Inspect the contents of **colLastOrder** to view information about the **Order** record which has just been added.
    a) Select the **View > Collections** command to display the collections for your app.



    b) Inspect **colLastOrder** and you should be able to see that **ID** that SharePoint assigned to the new **Order** item.



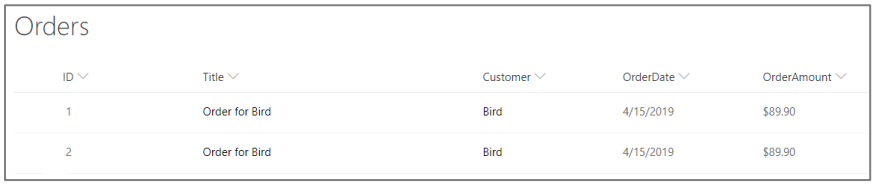
The reason you need to determine the **ID** of a new order has to do with saving shopping cart items to the **OrderDetails** list. Whenever you save an **OrderDetails** record, you must include the ID of the parent **Order** record.

11. Modify the **OnSelect** property of **btnSubmitOrder** to save **OrderDetails** records after saving the **Order** record.
    a) Modify the **OnSelect** property of **btnSubmitOrder** with the following expression.

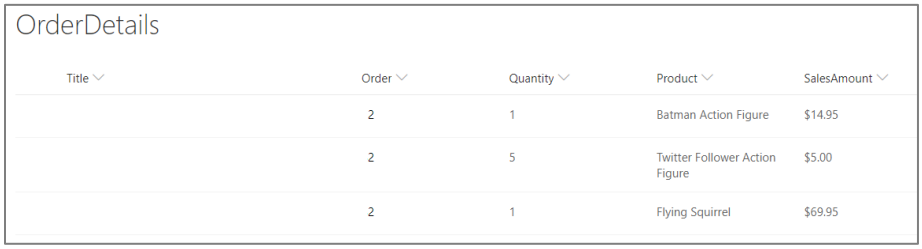
```
ClearCollect(
    colLastOrder,
    Patch(
        Orders,
        Defaults(Orders),
        {
            Title: "Order for " & galCustomers.Selected.'Last Name',
            OrderDate: dteOrderDate.SelectedDate,
            OrderAmount: Sum(
                colShoppingCart,
                Total
            ),
            Customer: {
                '@odata.type': "#Microsoft.Azure.Connectors.SharePoint.SPListExpandedReference",
                Id: galCustomers.Selected.ID,
                Value: galCustomers.Selected.'First Name' & " " & galCustomers.Selected.'Last Name'
            }
        }
    )
);
ClearCollect(
  colLastOrderDetails,
  ForAll(colShoppingCart,
    Patch(
      OrderDetails,
      Defaults(OrderDetails),
      {
        Product: Product,
        Quantity: Quantity,
        SalesAmount: Total,
        Order: {
             '@odata.type': "#Microsoft.Azure.Connectors.SharePoint.SPListExpandedReference",
              Id: First(colLastOrder).ID,
              Value: First(colLastOrder).ID
        }
      }
    )
  )
);
```

    b) Test your work by starting the app and submitting an order.

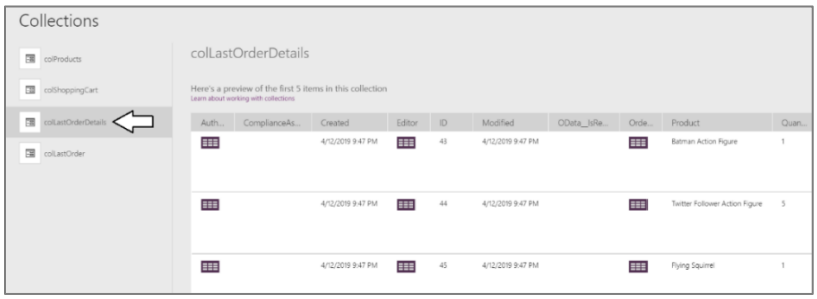
c) After submitting the order, navigate to the **Order** list in your SharePoint site and confirm a new list item has been added.



d) After verifying the **Order** has been created, navigate to the **OrderDetails** list and confirm list items have been added there.



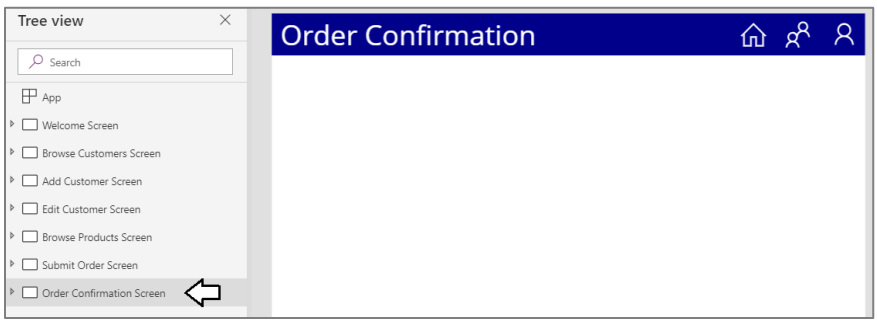
12. Inspect the contents of **colLastOrderDetails** to view information about the **Order** record which has just been added.
    a) Select the **View > Collections** command to display the collections for your app.
    b) Inspect the contents of **colLastOrderDetails** and confirm it contains the data for the new **OrderDetails** records in SharePoint.



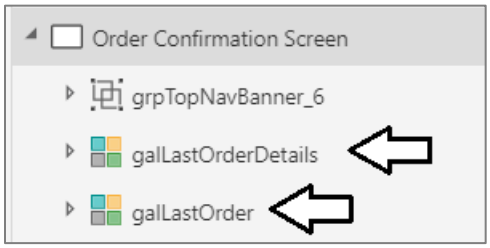
Exercise 5: Create the Order Confirmation Screen (If you have time)

This final exercise will involve you extending the **Customer Ordering** app by adding the **Order Confirmation Screen** on which you will display shopping cart data from **colLastOrder** and **colLastOrderDetails**. This is an extra credit exercise that will be different from the earlier exercises because it will not include step-by-step instructions. Instead you will only be provided with high-level instructions for creating galleries to display the order data that has been saved to the **Orders** list and the **OrderDetails** list.

1. Create the **Order Confirmation Screen**.
    a) Create a new **Blank** screen and rename it to **Submit Order Screen**.
    b) Copy **grpTopNavBanner** from **Welcome Screen** and paste it to **Order Confirmation Screen**.



2. Create two new galleries on the **Order Confirmation Screen** to display the data that has been written to SharePoint.
   a) Create two new **Vertical** galleries and rename then to **galLastOrder** and **galLastOrderDetails**.

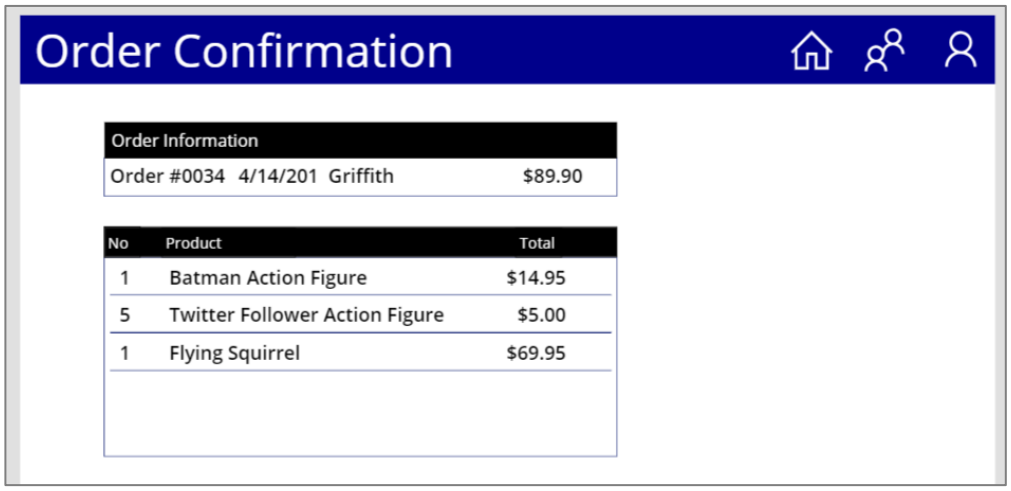


   b) Build the item template for **galLastOrder** to display the record that's been added to the collection named **colLastOrder**.
   c) Build the item template for **galLastOrderDetails** to display the records in the collection named **colLastOrderDetails**.
3. Update **btnSubmitOrder** to navigate from **Submit Order Screen** to **Order Confirmation Screen** after saving to SharePoint.
   a) Update the **OnSelect** property of **btnSubmitOrder** by adding a 3rd expression to navigate to the **Order Confirmation Screen**.

```
Navigate('Order Confirmation Screen',ScreenTransition.None)
```

When you modify the **OnSelect** property of **btnSubmitOrder**, make sure to leave the two expressions you added earlier which save the order data back to SharePoint. You should leave what's there and add a third expression to navigate to the **Order Confirmation Screen**. The behavior of the button should now navigate to the **Order Confirmation Screen** after saving data to SharePoint.

   b) Test your work by starting the app and submitting a new order.



   c) The **Order Confirmation Screen** should display all the data that has been written to SharePoint along with the Order ID that was created by SharePoint when the order was created.

You have now reach the end of this lab.