**Create a Power App from scratch**

Create a SharePoint list - **Service Desk Requests** list

* IssueTitle
* Issue Status - (Active/New/Resolved)
* Date
* Description
* Location
* Customer
* Assign To

In this exercise you will build a similar app from scratch, using the same SharePoint list as the data source. The purpose of this exercise is for you to learn how to build an app from scratch.

In the prior lab exercise when Power Apps created the app for the Service Desk Ticketing system, it automatically created the screens used in the app – the browse screen, the detail screen, and the edit screen. When you create a canvas app from scratch, you will manually create each of these screens instead:

* **Screen 1 – Browse screen.** The Browse screen will display the list of records from your SharePoint list, but it will only display a limited set of fields from each record.
* **Screen 2 – Detail screen.** The Detail screen will display all the fields for one specific record. If you select a record on the Browse screen, the detailed information for that record will be displayed on the Detail screen. This screen will also enable you to delete the record that you are viewing.
* **Screen 3 – Edit screen.** The Edit screen has data entry fields for adding information for a new record or changing information for an existing record.

When you initially create these screens, you will just create their forms with their basic structure. Once the screens are in place, you will then add trigger mechanisms to these screens to add, view, edit, and delete records in your SharePoint list.

### Task 1: Create the Browse screen for your Power App

In your role as Holly Spencer, you will begin the process of creating a canvas app from scratch by building the app’s Browse screen.

1. You should still be logged into your Domain Controller VM (LON-DC1) as **ADATUM\Administrator** and password **Pa55w.rd**; if not, then do so now.
2. In Internet Explorer, open a new tab in the browser and enter the following URL in the address bar: [**https://make.powerapps.com**](https://make.powerapps.com/)
3. This will open the **Power Apps studio**. On the main body of the **Home** page, under the **Make your own app** section, select **Canvas app from blank**.
4. On the **Canvas app from blank** window that appears, enter a name for your app (anything that you want) in the **App name** field, and in the **Format** section, select the **Phone** option.
5. Scroll down to the bottom of the window and select **Create**. If a **Welcome to Power Apps studio** dialog box appears, select **Skip**.
6. Notice how in the **Tree view** pane, none of the browse, detail, or edit screens appear under the **Screens** tab. Those screens were automatically created when you created an app from an existing data source, as you did in the prior lab exercise. But when creating a canvas app from scratch, as you are doing here, you must manually create those screens.  
   ‎  
   ‎**Screen1** is the only screen in the Tree view pane since it is the name of the blank screen that appears on the canvas. By default, the font size on the screen is so small (20%) that you cannot read it. Use the slider to increase the font size. While the text on the screen says **Add an item from the insert pane or connect to data**, do not select either of these options at this time. You will eventually configure the properties of this screen from the **Insert** pane, but for now, let’s begin by renaming this app screen.  
   ‎  
   ‎In the **Tree view** pane, select the **ellipses** icon that appears to the right of **Screen1**. In the menu that appears, select **Rename**, and then enter **BrowseScreen1** as the new screen name (this will keep the screen name consistent with the naming convention that is used when Power Apps automatically creates screens for an app).
7. Now you will access the **Insert** pane to configure screen properties. On the menu bar at the top of the Power Apps Studio screen, select **Insert**.
8. On the **Insert ribbon** that appears below the menu bar, select **Gallery**, and in the menu that appears, select **Vertical**.
9. On the canvas pane, you will see the vertical gallery and a **Select a data source** window. Note that in the **Tree view** pane, the default name of the vertical gallery you just added is **Gallery1.** Gallery1 represents the list of records that will be displayed in BrowseScreen1 from your data source. Since you have yet to define a data source, you can see that it simply fills in sample records. Once you select your data source, you will see the actual records from your data source in the gallery of the Browse screen.  
   ‎  
   ‎In the **Select a data source** window, enter **Share** (short for SharePoint) in the field (this is a search box). This will display a list of data sources starting with **Share**. In the list, select **SharePoint**. Alternatively, you could select the down arrow next to **Connectors** and select **SharePoint**.  
   ‎  
   ‎**Important:** If you happened to click somewhere else on the screen before you performed the prior step of entering **Share** in the Search box, the **Select a data source** window will disappear. **Do not worry!** You will simply perform this step using the **Properties pane**. First, in the **Tree view** pane, select **Gallery1** if it is not already selected. In the **Gallery1 Properties pane** on the right, in the **Data source** field, select the drop-down arrow. In the window that appears, select **Connectors** and then select **SharePoint**.
10. In the **SharePoint** window that appears, select the **SharePoint** connection that is listed (this displays the administrator’s email associated with the SharePoint connection).
11. A **Connect to a SharePoint site** window opens on the right side of the screen that lists the available SharePoint sites at the bottom of the window. Select **IT services** **–**[**https://m365xZZZZZZ.sharepoint.com/sites/ITServices**](https://m365xzzzzzz.sharepoint.com/sites/ITServices) (where ZZZZZZ is your tenant suffix ID provided by your lab hosting provider).
12. In the **Choose a list** window, you must select the SharePoint list for this site that you want to use as the data source for this app. Select the check box next **Service Desk Requests** and then select **Connect** on the bottom of the screen.  
    ‎  
    ‎**Note:** As soon as Power Apps connects to the **IT Services** site and the **Service Desk Requests** list, note how the gallery is updated with the items from the list.
13. On the canvas for **BrowseScreen1**, with **Gallery1** still selected in the **Tree view** pane, reduce the screen size on the slider to about 40% so that you can see more of the screen. Scroll from the top to the bottom of the screen and notice how the Gallery’s border starts at the very top of **BrowseScreen1** and goes a little past the middle of the screen.  
    ‎  
    ‎Since Holly wants to add a couple of controls to the top of this screen (an Add icon and a Refresh icon), you need to make room at the top of the screen for these controls. Therefore, you need to move the Gallery down to provide space at the top of the screen for the controls.  
    ‎  
    ‎To do this, first scroll down to the bottom of the screen (so that you can see the bottom of **BrowseScreen1**). Then hold your mouse over the middle circle in the bottom border line of the Gallery; this will change your mouse icon to an up and down arrow. With your left mouse button held down, drag the bottom border of the gallery to the very bottom of **BrowseScreen1**.  
    ‎  
    ‎Next, scroll to the very top of **BrowseScreen1** and repeat this process with the top border line of the gallery. However, in this case, move it down to about where the top of the second sample record begins. This leaves basically the height of the first sample record space to insert the two control icons later in this exercise.  
    ‎  
    ‎**Note:** Depending on where your slider is set, you may not be able to see the blank space that you just created at the top of the screen, even after scrolling to the top. If this occurs, set your slider to 100% to see the top of the screen. You will need to set it to 100% in later tasks when adding your controls. For now, you can set the slider to a smaller percentage if you wish to see more records in the Gallery.
14. On the **Gallery1 Properties pane** on the right side of the screen, select the **down arrow** next to the **Layout** property and select the option labeled **Title, subtitle, and body**. Notice how the gallery is updated based on this selection.
15. Holly now wants to change which fields that are displayed in the gallery. In the **Properties pane**, select **Edit** that appears to the right of the **Fields** property. A new **Data** window opens which shows the **Body**, **Subtitle**, and **Title** fields; these are the three fields you selected for the **Layout** property in the prior step. For each field, you can select which field from the SharePoint list you want displayed in that field. Power Apps selects a default SharePoint field for each, but you can change them to display whatever fields that you want.  
    ‎  
    ‎By default, the **Body1** field is mapped to the **Issue Title**, the **Subtitle1** field is mapped to the **Date**, and the **Title1** field is mapped to the **Issue Status**.  
    ‎  
    ‎Holly has decided to customize the browse screen to change the **Body1** field so that it maps to the **Assigned To** column of the SharePoint list. Select the drop-down arrow for the **Body1** field. Note how the field is selected in the Gallery image. Select the drop-down arrow again for the **Body1** field to display the list of available fields. Scroll to the top of the list and select **Assign To**.
16. In the **Data** window, notice two changes - the value in **Body1** is now **Assign To**, and a **Display value** field has appeared below the **Assign To** value. Select the drop-down arrow for the **Display value** field, and in the menu list, select **DisplayName**.
17. Notice how the records displayed in the gallery have been updated to reflect the **Assign To** field.
18. Close the **Data** window by selecting the **X** in the upper right corner.
19. You have now finished creating the browse screen for your app. While you can continue creating the Detail and Edit screens for your app, as a best practice it is recommended that you save each screen in the app as you complete it.  
    ‎  
    ‎Select **File** from the top menu bar of the PowerApps screen, and then select **Save** from the navigation pane that appears on the left side of the screen. Since the app has yet to be saved, a **Save as** screen appears. You want to accept the default setting on this screen that will save it to Power Apps in the cloud, and you want to keep the name that you previously assigned; therefore, simply select the **Save** button.
20. In the **<app name>** window, it should display a message indicating that all changes were saved. Since you have not finished creating all the necessary screens for your app, you do **NOT** want to publish it at this point in time. **Do NOT** select the **Publish** button. Instead, at the top of the navigation pane, select the back arrow to return to the Power Apps studio.
21. You have now finished creating the browse screen for your new app. Leave the Power Apps tabs open in your browser and proceed to the next task to create the Detail screen.

### Task 2 – Create the Detail screen for your Power App

Holly is now ready to create screen #2 for her app, which is the Detail screen. In a Power App, the browse screen displays a summary of the items from the selected data source, which in this case is the Service Desk Requests list for the IT Services site. If the user selects an item from this browse list, the app will display all the fields for that item within a Detail screen. You will create the Detail screen for your app in this task.

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the Browse screen for your app in the prior task. The tab should be displaying the **Insert** tab from the menu bar; if not, select it now.  
   ‎  
   ‎In the **Insert ribbon** that appears below the menu bar, select **New Screen**. In the menu of available screen types that appear, select **Blank**. This opens a new blank screen on the canvas pane that is titled **Screen2**.
2. In the **Tree view** pane under the **Screens** tab, select the **ellipses** next to **Screen2**, select the **Rename** option from the menu that appears, and then enter **DetailScreen1** in the name field.
3. Verify that **DetailScreen1** is selected in the **Tree view** pane (if not, select it now). On the **Insert ribbon**, select **Forms**, and in the menu that appears, select **Display**. The Detail form will display the fields for an individual record, but it will not allow you to edit the record (Note – in a later task, you will create the Edit screen, and for that screen, you will select Edit as the form type rather than Display).  
   ‎  
   ‎**Note:** In the **Tree view** pane, this action has added a display form titled **FormViewer1** under **DetailScreen**1.
4. If necessary, in the screen canvas in the middle of the page, scroll to the top to see the top of the form box. The form box is inserted by default at the top of page. Since you want to leave the top of the page open to add additional controls, you should move the empty form box down on the page to make space available at the top.  
   ‎  
   ‎To move the form box, hover your cursor over the circle that appears in the middle of the top border line for the form box. Note how the cursor image changes to an up and down arrow. At this point, hold the left mouse key and move the box down to the center of the page. Leave some space on the top of the page to add some additional controls later in this task.  
   ‎  
   ‎**Note:** You can repeat this process by grabbing the bottom border line of the form box and dragging it to the bottom of the form to provide additional space to display the fields for a given record (to avoid having to scroll through the fields if at all possible).
5. With the **FormViewer1** display form still selected, go to the **Properties pane** and select the down arrow in the **Data source** field. In the pop-up window that appears, select **Service Desk Requests**.  
   ‎  
   ‎In the canvas in the middle of the page, the form should immediately update to reflect the column headers from this SharePoint list; however, no data values are displayed at this point.
6. When a user selects an item on the Browse screen, the fields associated with that item will be displayed in this Detail screen. To accomplish that, you must associate the properties of the item selected from the SharePoint list with the fields in this Detail screen. You will do this by configuring a function property that is associated with each item in the form.  
   ‎  
   ‎Below the **Insert ribbon** is a **Function property** **bar**. The field on the left side of the function currently displays **Data Source.** Select the down arrow that appears in this field. A menu appears that displays the properties for the display form that you just added. Select **Item**.
7. Notice how the **function box** that appears to the right of the **Item** field is blank. The item property for the display form represents the record from the data source that will be displayed. In this case we want the item to be the record the user selects from the gallery in **BrowseScreen1**. If you will recall, the gallery of records displayed in BrowseScreen1 is titled **Gallery1**.  
   ‎  
   ‎In the function box, enter **Gallery1.Selected** and then press Enter.  
   ‎  
   ‎**Note:** By entering this function, the display form in the canvas will populate with values that map to the columns from the SharePoint list. The display form will select the values from the first record in the SharePoint list. In practice, when a user selects a record from the list of items in **BrowseScreen1**, the details of that record will be displayed in this **DetailScreen1** screen.
8. In the **Tree view** pane, under **DetailsScreen1**, you will see the control named **FormViewer1** (this is the display form). Select the arrow **(>)** that appears to the left of this control to expand it. Notice all the **Data Cards** that appear under this form control; these represent the columns of the SharePoint list that are displayed in **DetailsScreen1**.  
   ‎  
   ‎**Note:** These Data Card controls in the **Tree view** pane enable you to customize the fields displayed on this screen. For example, you remove any of the fields if you do not want to see them in this Detail screen by selecting the Data Card in the Tree view pane, selecting the ellipsis icon that appears to the right of it, and them selecting the Delete option. Because Holly wants to display all the SharePoint fields, do not remove any of them from the Detail screen.  
   ‎  
   ‎For this app, Holly wants to rearrange the order of the fields. Select the **FormViewer1** control (if necessary) in the **Tree view** pane, then go to the **Properties pane** and select **Edit Fields**, which appears next to the **Fields** property.
9. In the **Fields** window that appears, the fields appear in the order in which they will be displayed in this form. Holly has decided to move the **Custome**r field up so that it appears after the **Date** field.  
   ‎  
   ‎Hover your cursor over the **Customer** field and note how an **ellipses** icon appears to the right of the field. Select this **ellipses** icon. In the menu that appears, select **Move up**. As you do this, notice how the **Customer** field moves up one spot in the list, above the **Location** field.  
   ‎  
   ‎Repeat this process until the **Customer** field is below the **Date** field.
10. Close the **Fields** window by selecting the **X** in the upper right corner.
11. You have now finished creating the detail screen for your app, which you want to save before making any additional changes.  
    ‎  
    ‎Repeat the process that you learned earlier to save the file and then return to the Power App studio. Do **NOT** publish the app yet as you still have additional changes to make.
12. Leave the Power Apps tabs open in your browser and proceed to the next task to assign a trigger action to the Detail screen.

### Task 3: Assign a trigger action to view the Detail screen

In the prior task, you created the DetailScreen1 that will be used to display the detailed record information for an item that the user selects in BrowseScreen1. However, this will do you no good until you configure a trigger action within the app that instructs it to display DetailScreen1 when the user selects a record in BrowseScreen1. You must also configure a trigger action that enables you to return from DetailScreen1 back to BrowseScreen1. You will create both trigger actions in this task.

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the detail screen for your app.  
   ‎  
   ‎In the **Tree view** pane, it should currently be displaying **FormViewer1**. Scroll up in the list until **BrowseScreen1** appears. If **BrowseScreen1** is not expanded, then select the arrow **(>)** that appears to the left of it to expand it now.
2. Under **BrowseScreen1**, select **Gallery1**.
3. Below the **Insert ribbon** is the function property that you configured in the prior task for **FormViewer1** (it displays **Items** in the field on the left and the function to the right of it). In this task, you will replace these values with a function property specific to Gallery1. This property will provide the trigger action to display the Detail screen when the user selects an item in the Browse screen.  
   ‎  
   ‎In the prior step you selected **Gallery1** (under **BrowseScreen1)**. If you recall, Gallery1 represents the summarized list of items in the Browse screen. You will now configure this function property to create the trigger action that controls what to do when an item within Gallery1 is selected on the Browse screen.  
   ‎  
   ‎In the **Function property** **bar**, select the down-arrow for the field on the left side of the function, and in the menu that appears, scroll down and select **On Select**.
4. On the formula input box, enter **Navigate(DetailScreen1)** and then press Enter.
5. You have now created the trigger action so that when a user selects a record from the gallery in BrowseScreen1, the app will display the details for that record in DetailScreen1.  
   ‎  
   ‎To test whether you have properly configured this trigger action, **press F5** to go into Preview mode. Select any record in the gallery and Power Apps should display the record details in DetailsScreen1. Close Preview mode by selecting the **X** in the upper right corner of the screen.
6. Review the list of records displayed in the gallery in BrowseScreen1. On the right side of each record is a right arrow (>), which is referred to as the **NextArrow** control. If a user selects this arrow, it will also display the details for the record.  
   ‎  
   ‎So if you are wondering why you configured an **OnSelect** trigger action to display the details of a record when you can simply select the **NextArrow** control, the answer is that the trigger action simply provides another option for displaying the record details.  
   ‎  
   ‎The difference between the two options is that using the **Next Arrow** control requires the user to specifically select the arrow key to navigate to the details screen. By configuring the **OnSelect** trigger action as you did, the user can instead select anywhere on the record.
7. Now that you can navigate from BrowseScreen1 to DetailsScreen1, you need to make sure you have a way to navigate back to BrowseScreen1. You will do this by adding a **Back** arrow to **DetailScreen1**, and then you will configure the arrow so that when it’s selected, it returns **BrowseScreen1**.  
   ‎  
   ‎In the **Tree view pane**, scroll down and select **DetailsScreen1.**
8. In the **Insert ribbon**, select **Icons**. In the menu that appears, scroll down and select **Back.**  
   ‎  
   ‎**Note:** In the **DetailScreen1** image, notice how the **Back arrow** icon (the **<** inside the circle) appears on the screen.
9. In the **Tree view** pane, notice how under **DetailScreen1** it now displays the **Icon1** control. The **Icon1** control is selected in the **Tree view** pane because when you inserted the **Back arrow** in the earlier step, Power Apps studio inserted the icon in DetailScreen1 and left it selected.
10. After reviewing the Detail screen, you decide that you want to make two cosmetic changes to the **Back arrow** icon – you want to change its color, and you want to display a tool tip that says “Back” when you hover your mouse over it.  
    ‎  
    ‎Since the **Icon1** control is selected in the **Tree view** pane (if not, select it now), the **Properties** pane displays the properties for this control. In the **Properties** pane, scroll down to the **Color** field, to the right of which are two boxes. The first box displays a capital **“A”** with a colored line below it. This line is currently black.  
    ‎  
    ‎To change the color of the **Back arrow** icon, select this field and then select the color of your choice from the color menu that appears. Note that you can select different colors to see which one you prefer. Each time you select a color, the color of the **Back arrow** icon in the **DetailScreen1** image changes to that selection.  
    ‎  
    ‎Once you have selected the color that you want to use, press the **Esc** key on your keyboard to close the color menu. This will keep **Icon1** selected (if you select anywhere else, it will also close the color menu, but **Icon1** will be unselected and you will have to select it again to add a tool tip to the icon).

‎Scroll further down the **Properties pane** for the until you get to the **Tooltip** property. Select the **Tooltip** box, enter **Back**, and press Enter.

‎**Note:** Hovering your mouse over the **Back arrow** icon on DetailScreen1 does not display the tool tip. You will test this out when you finish all other edits in this task and use Preview mode to test the changes that you made.

1. You now need to configure this **Back arrow** icon so that when the user selects it, **BrowseScreen1** is returned. In the **Tree view** pane, select the **Icon1** control under **DetailScreen1** if it’s not already selected.  
   ‎  
   ‎In the **Function property bar**, the field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Back arrow** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
2. On the formula input box, enter **Navigate(BrowseScreen1)** and then press Enter.  
   ‎  
   ‎You have now created the trigger action so that when a user selects the **Back arrow** icon in **DetailScreen1**, the app will return to **BrowseScreen1**.
3. To test whether you have properly configured this trigger action, **press F5** to go into Preview mode. The record that was selected in the gallery will be displayed in **DetailScreen1**.  
   ‎  
   ‎You should begin by hovering your mouse over the **Back arrow** icon, where the tool tip of **Back** should appear. Select the **Back arrow** icon, which should return you to **BrowseScreen1**.  
   ‎  
   ‎Close Preview mode by selecting the **X** in the upper right corner of the screen.
4. You have now finished creating the trigger mechanism for the Detail screen for your app, which you want to save before making any additional changes.  
   ‎  
   ‎Repeat the process that you learned earlier to save the file and then return to the Power App studio. Do **NOT** publish the app yet as you still have additional changes to make.
5. Leave the Power Apps tabs open in your browser and proceed to the next task to create the Edit screen.

### Task 4 - Create the Edit screen for your Power App

Holly is now ready to create screen #3 for her app, which is the Edit screen. This screen will enable users to add new Service tickets to the SharePoint list and edit and delete existing ones. You will create the Edit screen for your app in this task.

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the trigger actions for the Detail screen in the prior task. The tab should be displaying the **Insert** tab from the menu bar; if not, select it now.  
   ‎  
   ‎In the **Insert ribbon** that appears below the menu bar, select the drop-down arrow key next to **New Screen**. In the menu of available screen types that appear, select **Blank**. This opens a new blank screen on the canvas pane that is titled **Screen1**.
2. On the **Tree view** pane, select the **ellipses** next to **Screen1** and rename it to **EditScreen1**.
3. **EditScreen1** should be selected in the **Tree view** pane (if not, select it now). On the **Insert ribbon**, select **Forms**, and in the menu that appears, select **Edit**.  
   ‎  
   ‎**Note:** In the **Tree view** pane, this action has added an edit form titled **Form1** under **EditScreen1.**
4. If necessary, in the screen canvas in the middle of the page, scroll to the top to see the top of the form box. The form box is inserted by default at the top of page. Since you want to leave the top of the page open to add additional controls, you should move the empty form box down on the page to make space available at the top.  
   ‎  
   ‎To move the form box, hover your cursor over the circle that appears in the middle of the top border line for the form box. Note how the cursor image changes to an up and down arrow. At this point, hold the left mouse key and move the box down to the center of the page. Leave some space on the top of the page to add some additional controls later in this task.
5. With the **Form1** edit form still selected, go to the **Properties pane** and select the down arrow in the **Data source** field. In the pop-up window that appears, select **Service Desk Requests**.  
   ‎  
   ‎In the canvas in the middle of the page, the form should immediately update to display data entry boxes that are associated with each of the column headers from this SharePoint list.
6. Below the **Insert ribbon** is the **Function property** **bar**. The field on the left side of the function currently displays **Data Source.** Select the down arrow that appears in this field. A menu appears that displays the properties for the edit form that you just added. Select **Item**.
7. Notice how the **function input box** that appears to the right of the **Item** field is blank. The item property for the edit form represents the record that will be added to the data source that will be displayed.  
   ‎  
   ‎In the function box, enter **Gallery1.Selected** and then press Enter.  
   ‎  
   ‎**Note:** By entering this function, the edit form in the canvas will populate with values that map to the last selected item in the gallery of BrowseScreen1.
8. You have now finished creating the edit screen for your app, which you want to save before making any additional changes.  
   ‎  
   ‎Repeat the process that you learned earlier to save the file and then return to the Power App studio. Do **NOT** publish the app yet as you still have additional changes to make.
9. Leave the Power Apps tabs open in your browser and proceed to the next task to assign a trigger action to the Detail screen.

### Task 5 - Assign trigger actions to create a new entry

The Edit screen that you created in the prior task provides the form needed to enter the data for a new entry. However, while the form is ready for entering data, you must create several trigger mechanisms to process this form; three of the triggers are associated with specific icons being added to screens, and the another facilitates automatic navigation between screens:

* **Add icon -** You must add a trigger mechanism to display the Edit screen when you are on the Browse screen and you select an **Add** icon to add a new record.
* **Save icon -** You must add a trigger mechanism to submit the data for processing once the user has completed entering data in the Edit screen and selects a **Save** icon.
* **Navigation -** You must add a trigger mechanism to navigate away from the Edit screen and back to the prior screen that initiated the Edit screen (either the Browse screen when adding a record and the Detail screen when editing a record) if the data has been successfully submitted.
* **Cancel icon -** You must add a trigger mechanism to navigate away from the Edit screen and back to the prior screen if the user selects a **Cancel** icon.  
  ‎

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the Edit screen in the prior task. The tab should be displaying the **Insert** tab from the menu bar; if not, select it now.
2. In the **Tree view** pane, select **BrowseScreen1**.
3. In the earlier task when you first created **BrowseScreen1**, the Gallery’s border went to the very top of the screen. Since you knew you would be adding controls to the screen later on, you moved the Gallery’s top border down to provide enough space to add these controls.  
   ‎  
   ‎The first control you want to add is an **Add** icon for creating a new item. Scroll to the top of **BrowseScreen1** so that you can see the top of the screen with the blank space you configured for the controls.  
   ‎  
   ‎**Note:** Depending on where your slider is set, you may not be able to see the blank space at the top of the screen, even after scrolling to the top. If this occurs, set your slider to 100% to see the blank space at the top of **BrowseScreen1.**  
   ‎  
   ‎On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Add**. Notice how the **Plus sign (+)** icon shows up on the screen, and in the **Tree view** pane, **Icon2** appears below **BrowseScreen1**.
4. You now must associate a trigger mechanism to this **Plus sign (+)** icon so that when the user selects it, **EditScreen1** will be returned.  
   ‎  
   ‎Below the **Insert ribbon** is the **Function property** **bar**. The field on the left side of the function currently displays **OnSelect.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Plus sign (+)** **sign** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
5. In the function box, enter **NewForm(Form1); Navigate(EditScreen1)** and then press Enter.  
   ‎  
   ‎This function actually encapsulates two separate actions:
   * **NewForm(Form1)** - changes the form mode to New to enable the entry of a new record in **Form1**, which is the name of the edit form under **EditScreen1**.
   * **Navigate(EditScreen1) –** navigates the user to the screen (EditScreen1) which contains the edit form (Form1).
6. Now that the **Plus sign (+)** icon has been configured, you decide that you want to make some cosmetic changes to it. First, you decide that you want to change the location of the icon from the top left corner of **BrowseScreen1** to the top right. To do this, select the border of the **Plus sign (+)** icon (so that an image of crossed arrows appears) and drag it across to the right corner.
7. Next, you want to change the color of the icon and assign it a tool tip.  
   ‎  
   ‎If the **Plus sign (+)** icon is not selected, then click on it now (this will select **Icon2** under **BrowseScreen1** in the **Tree view** pane and display the **Properties pane** for this icon on the right side of the page).  
   ‎  
   ‎Repeat the processes that you learned earlier to select a color and to add a tooltip for this icon. Choose whichever color you wish and enter **Create new item** for the tooltip.
8. You must now enter a second trigger mechanism that will be associated with a **Save** icon. This icon and trigger mechanism will ensure that the data the user enters in the Edit screen can get saved to the data source, which for this app is the **Service Desk Requests** SharePoint list.  
   ‎  
   ‎In the **Tree view** pane, select **EditScreen1**.
9. On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Check**. Notice how the **check mark** icon shows up on the screen, and in the **Tree view** pane, **Icon3** appears below **EditScreen1**.
10. Now that you have added the **check mark** icon to **EditScreen1**, you must associate a trigger mechanism to the check mark so that the data is submitted when the **check mark** icon is selected. In the **Tree view** pane, the **Icon3** control should be selected; if not, then do so now.  
    ‎  
    ‎In the **Function property** **bar**, the field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **check mark** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
11. In the function box, enter **SubmitForm(Form1)** and then press Enter.
12. You have just associated a trigger mechanism to submit the data in the edit form (**Form1**) for processing when the check mark icon is selected. You must now create another trigger mechanism to navigate away from **Form1** and back to the prior screen once the data has been successfully submitted. This is a navigation trigger; there is no icon associated with it.  
    ‎  
    ‎In the **Tree view** pane, under **EditScreen1**, select **Form1**.
13. In the **Function property** **bar**, the field on the left side of the function currently displays **Data Source**. Select the down arrow for this field and select **On Success** from the menu.
14. In the function box, enter **Back()** and then press Enter.
15. You must now add one final trigger mechanism to the data entry form (Form1), which is the ability to cancel the data entry operation and return to the prior screen. This trigger will be associated with a **Cancel** icon.  
    ‎  
    ‎In the **Tree view** pane, under **EditScreen1**, select **Form1** if it’s not already selected.
16. On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Cancel**. Notice how the **“X”** (Cancel) icon shows up on the screen, and in the **Tree view** pane, **Icon4** appears below **EditScreen1**.
17. You notice that the “X” (Cancel) icon is partially placed over the check mark icon in the upper left corner of the screen. To fix this, you want to move the “X” icon to the upper right corner of the form. To do this, select the border of the “X” icon (so that an image of crossed arrows appears) and drag it across to the right corner. You will also need to drag it up a bit so that it’s on the same line as the check mark icon.
18. Now that you have added the **Cancel** icon to **EditScreen1**, you must associate a trigger mechanism to the icon so that EditScreen1 is reset and the user is returned back to the most recently processed screen when the **“X”** icon is selected. In the **Tree view** pane, the **Icon4** control should be selected; if not, then do so now.  
    ‎  
    ‎In the **Function property** **bar**, the field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Cancel** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
19. In the function box, enter **ResetForm(Form1); Back()**.  
    ‎  
    ‎This function actually encapsulates two separate actions:
    * **ResetForm(Form1)** - resets **Form1**, which is the name of the edit form under **EditScreen1**.
    * **Back() -** takes the user back to the most recently displayed screen.
20. After reviewing the Edit form, you decide that you want to change the colors of the two icons that you just added to **EditScreen1**, and you want to add a tooltip for each as well. With prior icons, you selected the color of your choice. However, for these two icons, the IT Consultant has suggested to Holly that she use two specific colors – green for the check mark and red for the “X” icon.  
    ‎  
    ‎Since you cannot remember which control in the **Tree view** pane is associated with each icon, on the canvas for **EditScreen1**, select the **check mark** icon. In the **Tree view** pane, you notice this selects **Icon3** under **EditScreen1**. The **Properties pane** on the right also displays the properties for this icon.  
    ‎  
    ‎Repeat the processes that you learned earlier to select a color for this icon and to add a tool tip. At the bottom of the Color menu under **Standard colors**, select the **Green (RGBA 54, 176, 75, 1)** box. For the **Tooltip**, enter **Save**.  
    ‎  
    ‎Next, select the “X” icon on the **EditScreen1** canvas, which selects **Icon4** under **EditScreen1** in the **Tree view** pane. The **Properties pane** on the right also displays the properties for this icon.  
    ‎  
    ‎Repeat the processes for selecting a color and adding a tool tip for this icon. At the bottom of the Color menu under **Standard colors**, select the **Red (RGBA 255, 0, 0, 1)** box. For the **Tooltip**, enter **Cancel**.
21. To test whether you have properly configured these trigger actions, select **BrowseScreen1** in the **Tree view** pane and then press **F5** on your keyboard to run the app in Preview mode.
22. Because you selected **BrowseScreen1** prior to pressing **F5**, the browse screen will appear, and it will display the existing records from the **Service Desk Requests** list. You want to add a new record, so select the **Plus (+) sign** icon in the upper right corner of the form. This should display the edit form (**EditScreen1**).
23. Populate the form with the following data:
    * Issue Status – **Active**
    * Date – **June**
    * Issue Title – **Test record**
    * Description – **Testing trigger actions**
    * Location – **Home Office**
    * Customer – enter **Megan** and select **Megan Bowen** from the list
    * Assign To – leave this field blank
24. To test the **Save** functionality, begin by hovering your mouse over the green **check mark** icon, which should display the tool tip of **Save.** Once you have verified the tool tip displays properly, select the **check mark** icon to submit the form.
25. Since the **Assign To** field is a required field, an error message should be displayed at the top of the screen indicating that one or more fields have errors. Scroll down through the form and note the specific error message below the **Assign To** field.  
    ‎  
    ‎In the **Assign To** field, enter **Allan** in the field and then select **Allan Deyoung** from the list.
26. Select the **check mark icon** to submit the form.
27. This time, all the data should have been successfully processed, the record should have been added to the **Service Desk Requests** list, and **BrowseScreen1** should be returned.
28. You should now verify that the record has been added to the **Service Desk Requests** list in the **IT Services** site.  
    ‎  
    ‎Select the **IT Services – Service Desk Requests** tab in your browser, and then select the **Refresh** icon on the right side of the address bar. Scroll down through the list and verify the record that you entered appears in the list.
29. Switch back to the **<app name> - Saved (Unpublished) – Power Apps** tab in your browser. You should still be on the browse screen for the app (**BrowseScreen1**) and it should still be running in Preview mode.
30. Select the **Plus (+) sign** icon to display the edit form.
31. To test the **Cancel** functionality, begin by hovering your mouse over the red **X** icon, which should display the tool tip of **Cancel**. Once you have verified the tool tip displays properly, select the **X** icon; this should return you to the **BrowseScreen1.**
32. To stop running the app in Preview mode, select the **X** in the circle that appears in the upper right corner of the screen.
33. You have now finished creating the trigger mechanisms for adding and canceling records in your app, which you want to save before making any additional changes.  
    ‎  
    ‎Repeat the process that you learned earlier to save the file and then return to the Power App studio. Do **NOT** publish the app yet as you still have additional changes to make.
34. Leave the Power Apps tabs open in your browser and proceed to the next task to add Edit and Delete functionality to the Edit screen.

### Task 6 - Assign trigger actions to edit and delete a record

Now that Holly has created the ability to add a new record into the SharePoint list, she wants to be able to edit and delete existing records as well. In this task, you will create the triggers needed to edit and delete a record in your app.

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the Edit screen in the prior task. The tab should be displaying the **Insert** tab from the menu bar; if not, select it now.
2. In the **Tree view** pane, select **DetailScreen1**.
3. If necessary, scroll to the top of the canvas so that you can see the top of the **DetailScreen1** form. On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Edit**. Notice how the **Edit** **(Pencil)** icon shows up on the screen, and in the **Tree view** pane, **Icon5** appears below **DetailScreen1**.
4. You notice that the **Pencil** icon is partially placed over the **Back arrow** icon in the upper left corner of the screen. To fix this, you want to move the Pencil icon to the upper right corner of the form. However, since you also want to add a Delete icon to this screen in a later step, you do NOT want to move the Pencil icon into the far right corner. In the right corner of the form, you want to display both the Edit icon and the Delete icon, where the Edit icon appears to the left of the Delete icon. Therefore, you want to move the Pencil icon to the right, but still leave room to the right of it for the Delete icon.  
   ‎  
   ‎To do this, select the border of the Pencil icon (so that an image of crossed arrows appears) and drag it across to the right corner, making sure you leave enough room to the right of it for the Delete icon. You will also need to drag it up a bit so that it’s on the same line as the **Back arrow** icon.
5. Now that you have added the **Edit** icon to **EditScreen1**, you must associate a trigger mechanism to it so that EditScreen1 is displayed when the **Edit** icon is selected. In the **Tree view** pane, the **Icon5** control should be selected; if not, then do so now.  
   ‎  
   ‎In the **Function property** **bar**, the field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Edit** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
6. In the function box, enter **EditForm(Form1); Navigate(EditScreen1)** and then press Enter.  
   ‎  
   ‎This function actually encapsulates two separate actions:
   * **EditForm(Form1)** - changes the form mode to Edit to enable updates to an existing record, where **Form1** is the name of the edit form under **EditScreen1**.
   * **Navigate(EditScreen1) –** navigates the user to the screen (**EditScreen1**) containing the edit form.
7. You now want to add another icon to **DetailScreen1** that enables you to delete a record. In the **Tree view** pane, select **DetailScreen1**.
8. If necessary, scroll to the top of the canvas so that you can see the top of the **DetailScreen1** form. On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Trash**. Notice how the **Trash** **(trash can)** icon shows up on the screen, and in the **Tree view** pane, **Icon6** appears below **DetailScreen1**.
9. You notice that the **Trash** icon is partially placed over the **Back arrow** icon in the upper left corner of the screen. To fix this, you want to move the **Trash** icon to the upper right corner of the form and to the right of the **Pencil** icon. To do this, select the border of the Trash icon (so that an image of crossed arrows appears) and drag it across to the right corner, to the right of the **Pencil** icon. You will also need to drag it up a bit so that it’s on the same line as the **Pencil** icon.  
   ‎  
   ‎**Note:** If you did not leave enough room for the Trash icon when you originally added the Pencil icon, you may need to move the Pencil icon to allow enough room for the Trash icon.
10. Now that you have added the **Trash** icon to **DetailScreen1**, you must associate a trigger mechanism to it so that the selected record is deleted from the SharePoint list when the **Trash** icon is selected. In the **Tree view** pane, the **Icon6** control should be selected; if not, then do so now.  
    ‎  
    ‎In the **Function property** **bar**, the field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Edit** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
11. In the function box, enter **Remove('Service Desk Requests', Gallery1.Selected); If(IsEmpty(Errors('Service Desk Requests', Gallery1.Selected)), Back())** and then press Enter.  
    ‎  
    ‎This function actually encapsulates two separate actions:
    * **Remove('Service Desk Requests', Gallery1.Selected) -** removes from the SharePoint list titled Service Desk Requests the entry that was selected from the gallery in BrowseScreen1.
    * **If(IsEmpty(Errors('Service Desk Requests', Gallery1.Selected)), Back()) -** checks for the existence of errors when editing the SharePoint list. Errors can occur for many reasons, including network outages, inadequate permissions, and edit conflicts. If there are no errors (the error table is empty) then the app will go back to the most recently displayed screen. The “If” statement does not include any action if there are errors in the error table; however, PowerApps will, by default, display a message.
12. After reviewing the Detail form, you decide that you want to change the colors of each icon and display a tool tip when you hover your mouse over them.  
    ‎  
    ‎Since you cannot remember which control in the **Tree view** pane is associated with each icon, on the canvas for **DetailScreen1**, select the **Pencil** icon. In the **Tree view** pane, you notice this highlights **Icon5** under **DetailScreen1**. The **Properties pane** on the right also displays the properties for this icon.  
    ‎  
    ‎Repeat the processes that you learned earlier to select a color and to add a tooltip for this icon. Choose whichever color you wish and enter **Edit** for the tooltip.  
    ‎  
    ‎Next, select the **Trash** icon on the canvas. In the **Tree view** pane, you notice this highlights **Icon6** under **DetailScreen1**. The **Properties pane** on the right also displays the properties for this icon.  
    ‎  
    ‎Repeat the processes to select a color and add a tooltip for this icon. Choose whichever color you wish and enter **Delete** for the tooltip.
13. To test whether you have properly configured these trigger actions, select **BrowseScreen1** in the **Tree view** pane and then press **F5** on your keyboard to run the app in Preview mode. **BrowseScreen1** will display the existing records from the **Service Desk Requests** list.
14. You must first select a record from the gallery. Find a record, and then select the **>** icon. This should display the record in the Detail screen.
15. To test the Edit functionality, begin by hovering your mouse over the **Pencil** icon, which should display the **Edit** tool tip. Select the **Pencil** icon, which should display the Edit form with the data for the selected record pre-filled in the fields.  
    ‎  
    ‎Make a change to any of the fields in this record (**Hint:** change something that will be easy to find in the SharePoint list and BrowseScreen1, such as changing the value of the **Issue Status** to **EDIT TEST**). Then hover your mouse over the green check mark icon, which should display the **Save** tool tip. Select the **check mark** icon to save your changes.  
    ‎  
    ‎If the change was successful, the Detail screen should return. You should see your changes in the information displayed on the screen.
16. You should now verify that the record has been changed in the **Service Desk Requests** list in the **IT Services** site.  
    ‎  
    ‎Select the **IT Services – Service Desk Requests** tab in your browser, and then select the **Refresh** icon on the right side of the address bar. Scroll down through the list and verify the record that you entered appears in the list.
17. Switch back to the **<app name> - Saved (Unpublished) – Power Apps** tab in your browser. You should still be on the Detail screen for the app (**DetailScreen1**) and it should still be running in Preview mode (if not, select **DetailScreen1** in the **Tree view** pane and then press F5 on the keyboard). The record that you just changed should still be displayed.
18. To test the Delete functionality, you should delete the record that you just changed, which will make it easy to determine that the record no longer exists in the SharePoint list.  
    ‎  
    ‎Begin by hovering your mouse over the **Trash** icon, which should display the **Delete** tool tip. Select the **Trash** icon, which should delete the record from the SharePoint list and then take you back to the browse screen (**BrowseScreen1**).
19. You should now verify that the record has been deleted from the **Service Desk Requests** list in the **IT Services** site.  
    ‎  
    ‎Select the **IT Services – Service Desk Requests** tab in your browser, and then select the **Refresh** icon on the right side of the address bar. Scroll down through the list and verify the record that you deleted does not appear in the list.
20. Switch back to the **<app name> - Saved (Unpublished) – Power Apps** tab in your browser. You should still be on the Browse screen for the app (**BrowseScreen1**) and it should still be running in Preview mode. You should see that the deleted record no longer appears in this list as well.  
    ‎  
    ‎To stop running the app in Preview mode, select the **X** in the circle that appears in the upper right corner of the screen.
21. You have now finished creating the trigger mechanisms for editing and deleting records in your app, which you want to save before making any additional changes.  
    ‎  
    ‎Repeat the process that you learned earlier to save the file and then return to the Power App studio. Do **NOT** publish the app yet as you still have additional changes to make.
22. Leave the Power Apps tabs open in your browser and proceed to the next task to add a control that refreshes the connected data source.

### Task 7 - Assign trigger to refresh the connected data source

With the three screens for her app now in place, Holly wants to add one more control to the Browse screen. This control will refresh the connected data source (the SharePoint list) to ensure that what the user sees in the Browse screen is always current.

1. You should still have the **<app name> - Power Apps** tab open in your browser from when you finished creating the Edit and Delete triggers in the prior task. The tab should be displaying the **Insert** tab from the menu bar; if not, select it now.
2. In the **Tree view** pane, select **BrowseScreen1**.
3. In the earlier task when you first created **BrowseScreen1**, the Gallery’s border went to the very top of the screen. Since you knew you would be adding controls to the screen later on, you moved the Gallery’s top border down to provide enough space at the top of the screen to add these controls. You then added the **Add** icon for creating a new record, and you moved it to the top right corner of the screen. You now want to add the **Reload** icon and move it to the left of the **Add** icon.  
   ‎  
   ‎Scroll to the top of **BrowseScreen1** so that you can see the top of the screen with the **Add** icon.  
   ‎  
   ‎**Note:** Depending on where your slider is set, you may not be able to see the space at the top of the screen that contains the **Add** icon, even after scrolling to the top. If this occurs, set your slider to 100% to see the **Add** icon at the top of **BrowseScreen1.**  
   ‎  
   ‎On the **Insert ribbon**, select **Icons**, and in the menu that appears, select **Reload**. Notice how the **Reload** icon shows up on the screen, and in the **Tree view** pane, **Icon7** appears below **BrowseScreen1**.
4. You now must associate a trigger mechanism to this **Reload** icon so that when the user selects it, the list of data that is displayed in the Gallery will be refreshed.  
   ‎  
   ‎Below the **Insert ribbon** is the **Function property** **bar**. The field on the left side of the function currently displays **On Select.** While this value is left over from the prior task, it’s also the trigger action that you want to associate with this **Reload** icon. If **On Select** is not displayed in this field, select the drop-down arrow and select it from the menu.
5. In the function box, enter **Refresh('Service Desk Requests')** and then press Enter.  
   ‎Now that the **Reload** icon has been configured, you decide that you want to make some cosmetic changes to it. First, you decide that you want to change the location of the icon from the top left corner of **BrowseScreen1** to the left of the **Add** icon. To do this, select the border of the **Reload** icon (so that an image of crossed arrows appears) and drag it across to so that it’s to the left of the **Add** icon.
6. Next, you want to change the color of this icon and assign it a tool tip.  
   ‎  
   ‎If the **Reload** icon is not selected on the canvas for **BrowseScreen1**, then select it now. This will select **Icon7** below **BrowseScreen1** in the **Tree view** pane and display the **Properties pane** for this icon.  
   ‎  
   ‎Repeat the processes that you learned earlier to select a color and to add a tooltip for this icon. Choose whichever color you wish and enter **Refresh list** for the tooltip.
7. To test whether you have properly configured the Reload trigger action, select **BrowseScreen1** in the **Tree view** pane and then press **F5** on your keyboard to run the app in Preview mode. **BrowseScreen1** will display the existing records from the **Service Desk Requests** list.
8. You must now add a new record into the SharePoint list. Select the **IT Services – Service Desk Requests** tab in your browser and then add a new record to the list.
9. Once you have added this record, switch back to the **<app name> - Saved (Unpublished) – Power Apps** tab in your browser. You should still be on the Browse screen for the app (**BrowseScreen1)**, and it should still be running in Preview mode.
10. Select the **Reload** icon and then verify that the new record that you added to the **Service Desk Requests** list appears in the list of records.  
    ‎  
    ‎To stop running the app in Preview mode, select the **X** in the circle that appears in the upper right corner of the screen.
11. You have now finished creating the trigger mechanism for refreshing the record list in your Browse screen. This completes the process of creating a Power App from scratch, so you’re ready to save your final changes and then publish the app.  
    ‎  
    ‎Repeat the process that you learned earlier to save the file; however, after saving the changes, you now want to select the **Publish** button. In the **Publish** dialog box that appears, select **Publish this version**.
12. Now that the final changes have been saved and the app is published, the **<app name>** window provides you with the ability to share the app with additional users. We will not be sharing this app, since you already shared the prior app that you created with the IT Consultant’s MOD Administrator account.
13. In your browser, close all the Power Apps-related tabs that are open.

**Congratulations! You have created a Power App from scratch that provides the ability to add, edit, delete, and view records from a data source.**