Contents

1 PL-900: Power Platform Fundamentals

- **Download Latest Student Handbook and AllFiles Content**
- **Are you a MCT?** - Have a look at our GitHub User Guide for MCTs
- **Need to manually build the lab instructions?** - Instructions are available in the MicrosoftLearning/Docker-Build repository

1.1 What are we doing?

- To support this course, we will need to make frequent updates to the course content to keep it current with the Power Platform services used in the course. We are publishing the lab instructions and lab files on GitHub to allow for open contributions between the course authors and MCTs to keep the content current with changes in the platform.
- We hope that this brings a sense of collaboration to the labs like we've never had before - when the Power Platform changes and you find it first during a live delivery, go ahead and make an enhancement right in the lab source. Help your fellow MCTs.

1.2 How should I use these files relative to the released MOC files?

- The instructor handbook and PowerPoints are still going to be your primary source for teaching the course content.

- These files on GitHub are designed to be used in conjunction with the student handbook, but are in GitHub as a central repository so MCTs and course authors can have a shared source for the latest lab files.
- It will be recommended that for every delivery, trainers check GitHub for any changes that may have been made to support the latest Azure services, and get the latest files for their delivery.

1.3 What about changes to the student handbook?

- We will review the student handbook on a quarterly basis and update through the normal MOC release channels as needed.

1.4 How do I contribute?

- Any MCT can submit a pull request to the code or content in the GitHub repro, Microsoft and the course author will triage and include content and lab code changes as needed.
- You can submit bugs, changes, improvement and ideas. Find a new Azure feature before we have? Submit a new demo!

1.5 Notes

1.5.1 Classroom Materials

1.6 It is strongly recommended that MCTs and Partners access these materials and in turn, provide them separately to students. Pointing students directly to GitHub to access Lab steps as part of an ongoing class will require them to access yet another UI as part of the course, contributing to a confusing experience for the student. An explanation to the student regarding why they are receiving separate Lab instructions can highlight the nature of an always-changing cloud-based interface and platform. Microsoft Learning support for accessing files on GitHub and support for navigation of the GitHub site is limited to MCTs teaching this course only.

1.7 title: Online Hosted Instructions permalink: index.html layout: home

2 Content Directory

Hyperlinks to each of the lab exercises and demos are listed below.

2.1 Labs

{% assign labs = site.pages | where_exp:"page", "page.url contains '/Instructions/Labs'" %} | Module | Lab | | --- | --- | {% for activity in labs %}| {{ activity.lab.module }} | [{{ activity.lab.title }}{% if activity.lab.type %} - {{ activity.lab.type }}{% endif %}](/home/ll/Azure_clone/Azure_new/PL-900-Microsoft-Power-Platform-Fundamentals/{{ site.github.url }}{{ activity.url }}) | {% endfor %}

2.2 Demos

2.3 {% assign demos = site.pages | where_exp:"page", "page.url contains '/Instructions/Demos'" %} | Module | Demo | | --- | --- | {% for activity in demos %}| {{ activity.demo.module }} | [{{ activity.demo.title }}](/home/ll/Azure_clone/Azure_new/PL-900-Microsoft-Power-Platform-Fundamentals/{{ site.github.url }}{{ activity.url }}) | {% endfor %}

2.4 lab: title: 'Lab: Validate lab environment' module: 'Module 0: Course introduction'

3 Module 0: Course introduction

3.1 Lab: Validate lab environment

3.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs. We expect to have our content fully up to date very soon.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

3.2 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visitors are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In this Module 0 lab, you will acquire a Power Platform trial tenant and access the Power Platform admin center. In the admin center, you will then create your **Practice** environment that you will do the majority of your lab work in.

3.3 Exercise 1 – Setup

3.3.1 Task 1 - Acquire your Power Platform trial tenant

1. Copy your **Microsoft 365 credentials** from the Authorized Lab Hoster.
2. Navigate to https://powerapps.microsoft.com and click **Start free.**
3. Under **Get started**, enter the email address from your Microsoft 365 credentials in the text box that says **Enter your work email address.**
4. You see a prompt that you have an existing account with Microsoft. Select **Sign in.**
5. Enter the password provided by the Authorized Lab Hoster.
6. Select **Yes** to stay signed in.

3.3.2 Task #2 – Create environment

1. Access https://admin.powerplatform.microsoft.com and log in with your Microsoft 365 credentials if prompted again.
2. Select **Environments** and click **+New.**

- For **Name**, enter [**My Initials**] **Practice.** (Example: AJ Practice.)
- For **Type**, select **Trial** (do not select the Trial (subscription-based) option).
- Change the toggle on **Create a database for this environment?** to **Yes.**
- Leave all other selections as default and click **Next.**
- On the next tab, leave all selections to default and click **Save.**

3. Your **Practice** environment should now show in the list of Environments.

   Your environment may take a few minutes to provision. Refresh the page if needed.

4 Exercise #2: Provision a Power Apps portal

Objective: Provisioning a Power Apps portal can take some time. In this exercise, you will create your Power Apps portal in your environment so that the provisioning process can be initiated. You will use this portal in a later lab.

4.1 Task #1: Create Power Apps portal

1. Sign in to https://make.powerapps.com
2. If the **Environment** displayed in the top right is not your Practice environment, click to select your Environment.
3. On the Home page, click on the **Portal from blank** panel under **Make your own app**

   If you do not see this option, try zooming out.
4. Provide new portal details
   - Enter **Bellows College Visitors** as the portal **Name**
   - Provide a unique URL; **something**.powerappsportals.com (if the name has been taken, choose a different one)
   - Select a base portal **Language**
   - Click **Create**

   The Portal provisioning process will run anywhere from 30 to 45 minutes. You do not have to wait, as this will continue while moving on to the next module.

4.2 lab: title: 'Lab 1: Data Modeling' module: 'Module 2: Introduction to Microsoft Dataverse'

5 Module 2: Introduction to Microsoft Dataverse

5.1 Lab: Data Modeling

5.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

6 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visits are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In this lab you will access your environment, create a Microsoft Dataverse database, and create a solution to track your changes. You will also create a data model to support the following requirements:

- R1 – Track the locations (buildings) of the campus visits
- R2 – Record basic information to identify and track the visitors
- R3 – Schedule, record, and manage visits

Finally, you will import sample data into Microsoft Dataverse.

7 High-level lab steps

To prepare your learning environments you will:

- create a solution and publisher
- add both new and existing components required to meet the application requirements. Refer to the data model document for the metadata description (tables and relationships). You can hold CTRL+click or right click the link to open the data model document in a new window.

Your solution will contain several tables upon completion of all the customizations:

- Contact
- Building
- Visit

7.1 Prerequisites:

- Completion of **Module 0 Lab 0 - Validate lab environment**

7.2 Things to consider before you begin:

- Naming convention
- Data types, restrictions (e.g. max length of a name)
- Datetime formatting to support easy localization

8 Exercise #1: Create Solution

8.1 Task #1: Create Solution and Publisher

1. Create Solution
   - Navigate to https://make.powerapps.com. You may need to reauthenticate - click **Sign in** and follow instructions if needed.
   - Select your environment by clicking on **Environment** on the upper right corner of the screen and choosing your environment from the drop-down menu.
   - Select **Solutions** from the left menu and click **New Solution**.
   - Enter [**Your Last Name**] **Campus Management** for **Display Name**.
2. Create Publisher
   - Click on the **Publisher** dropdown and select **+ Publisher**
   - In the window that pops up, enter **Bellows College** for **Display Name**

    - Enter **bc** for **Prefix**
    - Click **Save and Close**
    - Click **Done** in the pop-up window.
3. Complete the solution creation.
    - Now, click on the **Publisher** dropdown and select the **Bellows College** publisher you just created.
    - Validate that **Version** is set to **1.0.0.0**
    - Click **Create**.

9 Exercise #2: Add Existing and Create New Tables

Objective: In this exercise, you will add the standard Contact table and create new custom tables for Buildings and Visits in the solution.

9.1 Task #1: Add Existing Table

1. Click to open your **Campus Management** solution you just created.
2. Click **Add Existing** and select **Table**.
3. Locate **Contact** and select it.
4. Click **Next**.
5. Click **Select Components** under Contact.
6. Select the **Views** tab and select the **Active Contacts** view. Click **Add**.
7. Click **Select Components** again.
8. Select the **Forms** tab and select the **Contact** form.
9. Click **Add**.

    You should have **1 View** and **1 Form** selected.
10. Click **Add** again. This will add the Contact table with the selected View and Form to the newly created solution.

```
> Your solution should now have one table: Contact.
```

9.2 Task #2: Create Building Table

1. You should still have your browser open to your Campus Management solution. If not, open the Campus Management solution by following these steps:
    - Sign in to https://make.powerapps.com (if you are not already signed in)
    - Select **Solutions** and click to open the [**Your Last Name**] **Campus Management** solution you just created.
2. Create Building table
    - Click **New** and select **Table**.
    - Enter **Building** for **Display Name**
    - Click **Create**. This will start provisioning the table in background while you can start adding other tables and columns.

9.3 Task #3: Create Visit Table and Columns

The **Visit** table will contain information about the campus visits including the building, visitor, scheduled and actual time of each visit.

We would like to assign each visit a unique number that can be easily entered and interpreted by a visitor when asked during the visit check-in process.

We use **Time zone independent** behavior to record date and time information, because time of a visit is always local to the location of the building and should not change when viewed from a different time zone.

1. Select your **Campus Management** solution
2. Create Visit table
   - Click **New** and select **Table**.
   - Enter **Visit** for **Display Name**
   - Click **Create**. This will start provisioning the table in background while you can start adding other columns.
3. Create Scheduled Start column
   - Make sure you have the **Columns** tab selected and click **Add column**.
   - Enter **Scheduled Start** for **Display Name**.
   - Select **Date and Time** for **Data Type**.
   - In **Required**, select **Required**.
   - Expand **Advanced options** section.
   - In **Behavior**, select **Time zone independent**.
   - Click **Done**.
4. Create Scheduled End column
   - Click **Add column**.
   - Enter **Scheduled End** for **Display Name**.
   - Select **Date and Time** for **Data Type**.
   - In **Required**, select **Required**.
   - Expand **Advanced options** section.
   - In **Behavior**, select **Time zone independent**.
   - Click **Done**.
5. Create Actual Start column
   - Click **Add column**.
   - Enter **Actual Start** for **Display Name**.
   - Select **Date and Time** for **Data Type**.
   - In **Required**, leave this as **Optional**.
   - Expand **Advanced options** section.
   - In **Behavior**, select **Time zone independent**.
   - Click **Done**.
6. Create Actual End column
   - Click **Add column**.
   - Enter **Actual End** for **Display Name**.
   - Select **Date and Time** for **Data Type**.
   - In **Required**, leave this as **Optional**.
   - Expand **Advanced options** section.
   - In **Behavior**, select **Time zone independent**.
   - Click **Done**.
7. Create Code column

- Click **Add column**.
- Enter **Code** for **Display Name**.
- Select **Autonumber** for **Data Type**.
- Select **Date prefixed number** for **Autonumber type**.
- Click **Done**.

8. Click **Save Table**

10 Exercise #3: Create Relationships

Objective: In this exercise, you will add relationships between the tables.

10.1 Task #1: Create Relationships

1. Ensure that you are still viewing the **Visit** table of your **Campus Management** solution. If not, navigate there.
2. Create Visit to Contact relationship
   - Select the **Relationships** tab.
   - Click **Add Relationship** and select **Many-to-one**
   - Select **Contact** for **Related (One)**
   - Enter **Visitor** for **Lookup column display name**
   - Click **Done**.
3. Create Visit to Building relationship
   - Click **Add Relationship** and select **Many-to-one**
   - Select **Building** for **Related (One)**
   - Click **Done**.
4. Click **Save Table**.
5. Select **Solutions** from the top menu and click **Publish all customizations.**

11 Exercise #4: Import Data

Objective: In this exercise you will import sample data into the Dataverse database.

11.1 Task #1: Import solution

In this task you will import a solution that contains the Power Automate flow required to complete data import.

1. You should have the **DataImport__managed.zip** file stored on your Desktop. Download Data Import Solution if you do not.
2. Sign in to https://make.powerapps.com.
3. Select your [**my initials**] **Practice** environment at the top right, if it is not already selected.
4. Select **Solutions** in the left navigation panel.
5. Click **Import**, then click **Browse**. Browse and select **DataImport__managed.zip** from your Desktop, and then press **Next**.

   You may receive the following message:

   There are missing dependencies. Install the following solutions before installing this one...

   That message indicates that either the data model is not complete, the publisher prefix is not **bc**, or the **Building** and **Visit** table names differ from the names listed in the steps above.
6. Press **Next**. You should be prompted to re-establish connections.

7. Expand the **Select a connection** dropdown and select **+ New Connection**.
8. The new browser window or tab will open. Select **Create** when prompted to the connection. Sign in if required to complete creating the connection.
9. Close the current tab so that you are now back to the previous **Import a Solution** tab.
10. Ensure the connection you just created is selected. If you do not see your connection, click **Refresh** to refresh the list of connections.
11. Press **Import**.
12. Wait until the import is complete.

11.2 Task #2: Import Data

1. Open **Data Import** solution.
2. Check the **Status** of the **Import Data** flow.
3. If **Status** is **Off**, select **...** next to **Import Data** then select **Turn On**.

    Important: If you receive an error message, verify that the tables and columns you created match the instructions above.

4. Open **Import Data** component. A new tab will open Power Automate.
5. Click **Get Started** if presented with a popup.
6. Click **Run** then click **Run flow** when prompted.
7. Click **Done**.
8. Wait until the flow instance completes the run. You can refresh the **28-day run history** table to see when the flow has run.

    The purpose of this flow was to generate example data for the upcoming labs. In the next task, you will verify that the data import was successful.

11.3 Task #3: Verify Data Import

1. Navigate back to the previous Power Apps tab. Click **Done** on the popup.
2. Select **Solutions** on the left navigation bar and open your **Campus Management** solution.
3. Click to open the **Visit** table, then select the **Data** tab.
4. Click **Active Visits** in the top right-hand corner to display the view selector, then select **All columns**. This will change the view that is being used to display the Visit data.

    If you do not see **Active Visits** due to smaller resolution, you should see an eye icon in the same location.

    If the import was successful, you should see a list of visit rows.

5. Click on any value in the **Building** column, confirm that the Building form opens in a separate window.
6. Close the recently launched window.
7. Click on any value in the **Visitor** column (you may need to scroll the view to the right), confirm that the Contact form opens in a separate window.
8. Close the recently launched window.

12 Challenges

- Would you consider using *appointment* activity as part of the solution? What would it change?
- How could you enforce the scheduled end to be after the scheduled start?
- How could you add support for multiple meetings during a single visit?
- How could you secure the building access not only for external contacts but for internal staff member as well?

- How could you make visits to certain buildings require management approval? What would the approval process change in the data model?

12.1 lab: title: 'Lab 2: How to build a canvas app, Part 1' module: 'Module 3: Get started with Power Apps'

13 Module 3: Get started with Power Apps

13.1 Lab: How to build a canvas app, Part 1

13.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

14 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visits are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In part 1 this lab, you will design a Power Apps canvas app that college staff can use to manage visits for their guests.

15 High-level lab steps

We will follow the below outline to design the canvas app:

- Create the app from data using the phone form factor template
- Configure a detail page with visit info
- Configure an edit page to create to visits
- Configure a gallery control to show the visits
- Add filtering on the gallery data source to show only future visits

15.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**

15.2 Things to consider before you begin

- What is the most prevalent form factor for the target audience?
- Estimate the number of records in the system
- How to narrow the records selected to improve app performance and user adoption

16 Exercise #1: Create Staff Canvas App

Objective: In this exercise, you will create a canvas app from a template and then modify it to include required data.

16.1 Task #1: Create Canvas App

In this task, you will create a canvas app using the phone layout template based on Microsoft Dataverse. Using Visits as a selected table from Dataverse, the template will generate a Gallery - View - Edit app to manage campus visits.

1. View the apps in your environment.
   - Sign in to https://make.powerapps.com
   - Select your **environment** at the top right if it is not already set to your Practice environment.
   - Select **Apps**.
2. Create new canvas application
   - Click **New app** and select **Canvas**.
   - Select **Phone layout** under **Common Data Service**.
3. Select **Create** under the **Common Data Service** connection
4. Select **Visits** table
5. Click **Connect**
6. The **Welcome to Power Apps Studio** window may appear. Click **Skip**.
7. Save application
   - Click **File** > **Save**.
   - Enter [**Your Last Name**] **Campus Staff** as the app name.
   - Press **Save**.

16.2 Task #2: Configure Visits Detail Form

In this task, you will configure the Detail form to view information about individual visit records.

1. Select the **Back** arrow at the top left to go back to the app definition.
2. Expand **DetailScreen1** under **Tree view**
3. Select **DetailForm1**
4. Select **Edit fields** next to **Fields** in the right-hand panel.
5. Click **Add field**
6. Select the following fields:
   - Actual End
   - Actual Start
   - Building
   - Code
   - Scheduled End
   - Scheduled Start
   - Visitor
7. Click **Add**
8. Rearrange fields in the **Fields** pane by dragging and dropping field names up or down. Recommended order is:
   - Code, Name, Building, Visitor, Scheduled Start, Scheduled End, Actual Start, Actual End

Tip: You can collapse each field by clicking the down arrow beside the field name.

9. Remove the **Created On** field by clicking the ellipses (**...**) beside the field name and selecting **Remove**.
10. Close the **Fields** pane.
11. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.

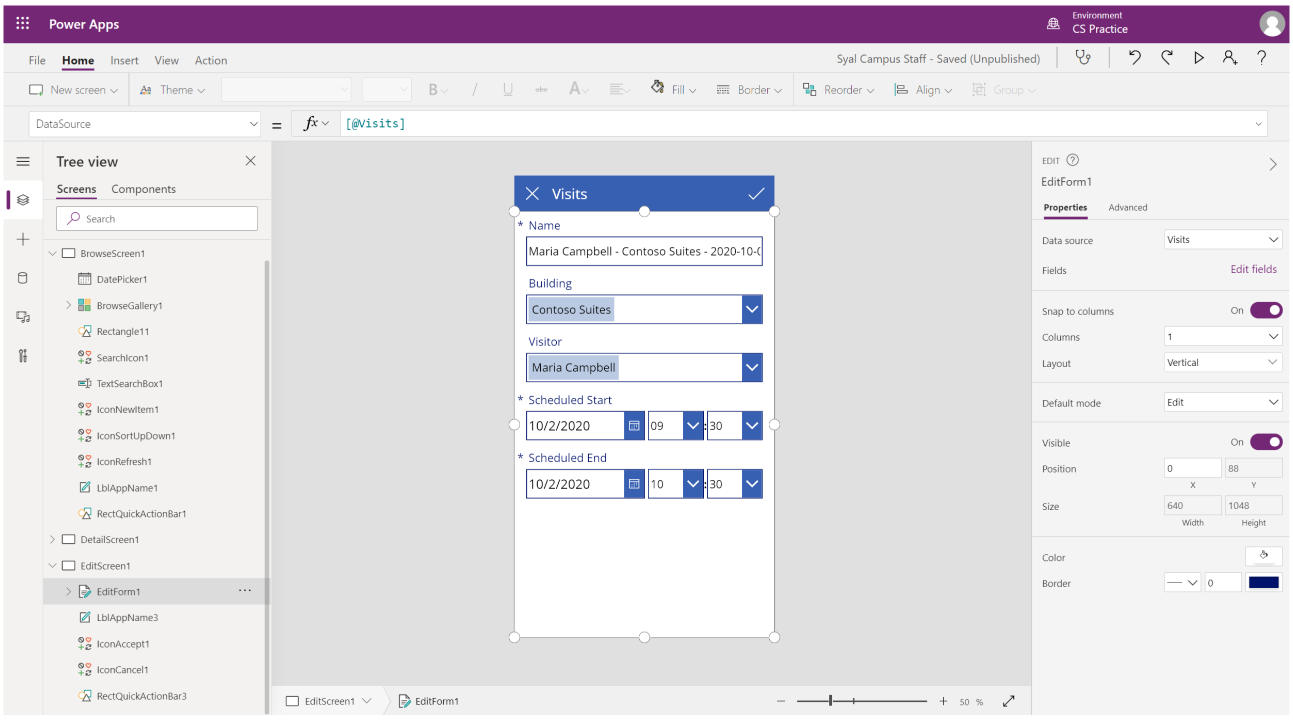
16.3 Task #3: Configure Visits Edit Form

In this task, you will configure a form to edit information about individual visit rows.

1. Expand **EditScreen1** under **Tree view**
2. Select **EditForm1**
3. Select **Created On** field and press **Del** key to remove the field
4. Select **Edit fields** in the properties panel
5. Click **Add field**
6. Select the following fields:
   - Building
   - Scheduled End
   - Scheduled Start
   - Visitor
7. Click **Add**
8. Rearrange fields in the **Fields** pane by dragging and dropping field names up or down. Recommended order is:
   - Name, Building, Visitor, Scheduled Start, Scheduled End

   Tip: You can collapse each field by clicking the down arrow beside the field name.
9. Close the **Fields** pane.
10. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.

Your screen should look approximately like the following:



DataSource = [@Visits]

Visits

* Name: Maria Campbell - Contoso Suites - 2020-10-(

Building: Contoso Suites

Visitor: Maria Campbell

* Scheduled Start: 10/2/2020 09 :30

* Scheduled End: 10/2/2020 10 :30

Data source	Visits
Fields	Edit fields
Snap to columns	On
Columns	1
Layout	Vertical
Default mode	Edit
Visible	On
Position	0 (X) 88 (Y)
Size	640 (Width) 1048 (Height)
Color	
Border	0

16.4 Task #4: Configure Visits gallery

In this task, you will configure the pre-generated gallery to display the title, start date and end date for the visit.

1. Expand **BrowseScreen1** under **Tree view**
2. Select **BrowseGallery1**
3. Select **TemplateSize** property from in the Advanced Properties panel on the right
4. Replace the expression with the following `Min(150, BrowseGallery1.Height - 60)`. That will ensure sufficient space for additional information.
5. In the app preview, select the first Date Time field in the gallery.
6. In the formula bar at the top, change **ThisItem.'Created On'**to `ThisItem.'Scheduled Start'`
7. Select the field again
8. Press **CTRL-C** then **CTRL-V** to create a copy of the field.
9. Using either mouse or keyboard, move the copied control down and align it with the other controls in the gallery, beneath the other Date Time field.
10. In the formula bar at the top, change **ThisItem.'Scheduled Start'** to `ThisItem.'Scheduled End'`
11. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.

16.5 Task #5: Add date filter

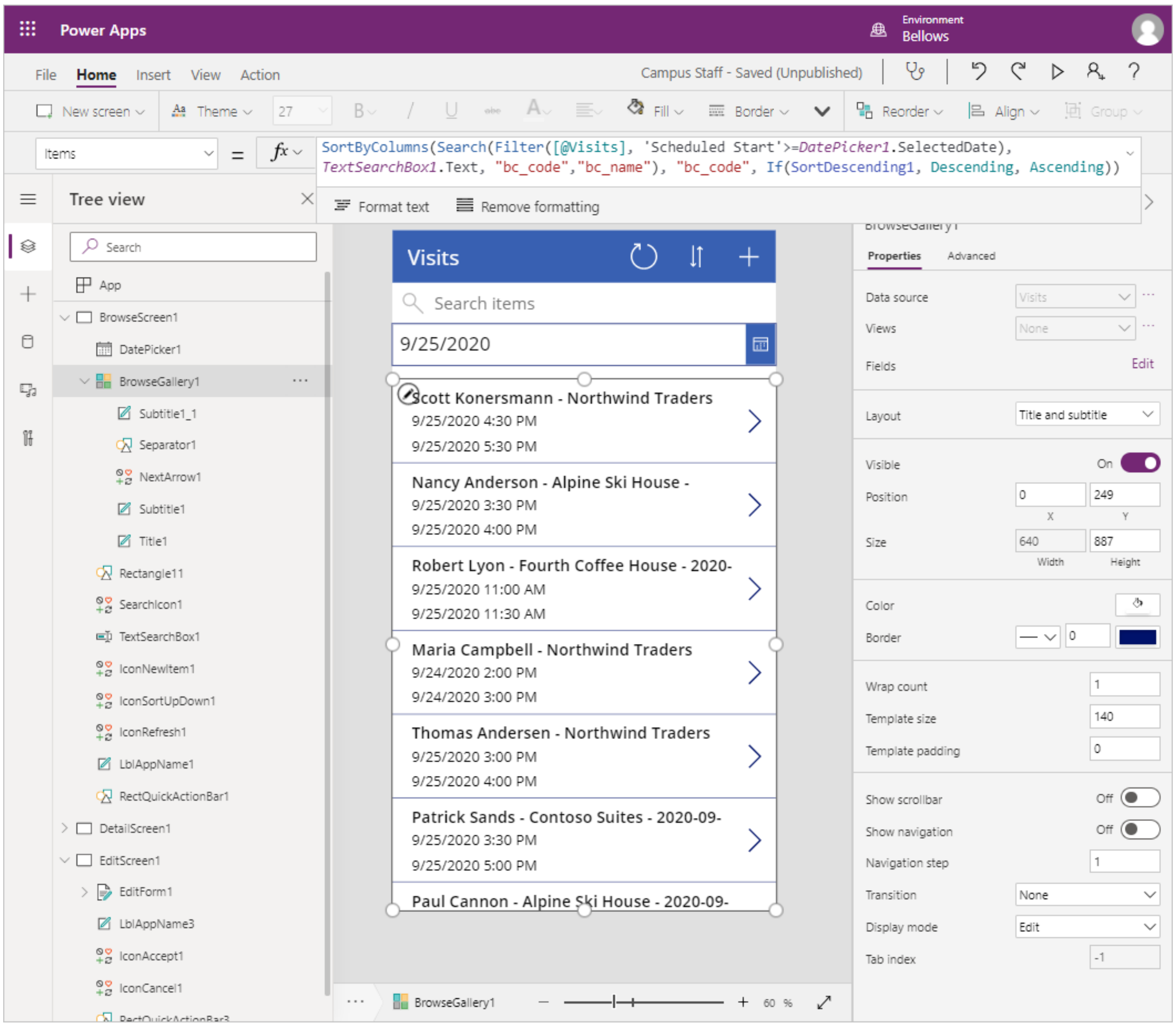
Because number of visits continuously grows, users need a feature to filter the visits gallery. For example, the user may want to see only the future visits. In this task, you will add ability to show visits only after a date selected by the user.

1. Select **BrowseScreen1**
2. Select **Insert** menu at the top.
3. Click **Input** and select **Date picker**.
4. Using either keyboard or mouse, position the control below the search box.
5. Select **BrowseGallery1**
6. Resize and move the gallery control so that it is located under the date picker and covers the screen. You can do this by clicking the resize icon at the top center of the gallery control and resizing the control to start after the date picker.
7. With **BrowseGallery1** selected, click the **Advanced** tab of the Properties pane.
8. Locate the **Items** property and click in the text box.
9. In the expression, locate [**@Visits**] and replace it with `Filter(Visits,'Scheduled End' >= DatePicker1.SelectedDate)`. The full expression should look like the following:

```
SortByColumns(
 Search(
     Filter(
         Visits,
         'Scheduled End' >= DatePicker1.SelectedDate
        ),
        TextSearchBox1.Text,
     "bc_code","bc_name"
    ),
  "bc_code",
  If(SortDescending1, Descending, Ascending)
)
```

10. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.

Your screen should look approximately like the following:



```
SortByColumns(Search(Filter([@Visits], 'Scheduled Start'>=DatePicker1.SelectedDate),
TextSearchBox1.Text, "bc_code","bc_name"), "bc_code", If(SortDescending1, Descending, Ascending))
```

Visits

Search items

9/25/2020

Scott Konersmann - Northwind Traders
9/25/2020 4:30 PM
9/25/2020 5:30 PM

Nancy Anderson - Alpine Ski House -
9/25/2020 3:30 PM
9/25/2020 4:00 PM

Robert Lyon - Fourth Coffee House - 2020-
9/25/2020 11:00 AM
9/25/2020 11:30 AM

Maria Campbell - Northwind Traders
9/24/2020 2:00 PM
9/24/2020 3:00 PM

Thomas Andersen - Northwind Traders
9/25/2020 3:00 PM
9/25/2020 4:00 PM

Patrick Sands - Contoso Suites - 2020-09-
9/25/2020 3:30 PM
9/25/2020 5:00 PM

Paul Cannon - Alpine Ski House - 2020-09-

Data source	Visits
Views	None
Fields	Edit
Layout	Title and subtitle
Visible	On
Position	0 (X), 249 (Y)
Size	640 (Width), 887 (Height)
Border	0
Wrap count	1
Template size	140
Template padding	0
Show scrollbar	Off
Show navigation	Off
Navigation step	1
Transition	None
Display mode	Edit
Tab index	-1

17 Exercise #2: Complete the App

In this exercise you will test the application and, once successful, you will add it to your solution.

17.1 Task #1: Test App

1. Start the application
   - Select the **BrowseScreen1** and press Function **F5**, or click the **Play** icon at the upper-right corner to preview the app.
   - The application should load and show a list of visits.
   - Test the filter by selecting different dates in the date picker control
   - Select a visit and verify that display form is working properly
   - Return to the gallery and press + to create a new visit. Verify that edit form contains required columns including visitor, building, and scheduled start and end dates.
   - Fill in the information and submit. Verify that the new record appears in the gallery.
   - Create at least 2 more visits.
   - Press **ESC** key or click the **X** icon to close preview mode.
2. Save and publish the application

- Click **File** and, if the Save button is displayed, click **Save**.
- Click **Publish**.
- Click **Publish this Version**.
- Click the **Back** arrow to navigate back to the app.
- Close the **Designer** browser window or tab.
- Click **Leave** if prompted when tried to close the browser window.

17.2 Task #2: Add App to Solution and publish

1. Open the Campus Management solution.
   - Sign in to https://make.powerapps.com
   - If the Environment displayed in the top right is not your Practice environment, select your **Environment**.
   - Select **Solutions**.
   - Click to open your **Campus Management** solution.
2. Select **Add existing**, then click **App**, and then click **Canvas app**.
3. Select **Outside solutions** tab.
4. Select your **Campus Staff** app, click **Add**.
5. Select **Publish all customizations**.

18 Challenges

- Calendar view of all visits and filtering by date range
- Ability to create and manage contacts as part of the app
- How to display multiple meetings during a single visit

18.1 lab: title: 'Lab 3: How to build a canvas app, part 2' module: 'Module 3: Get started with Power Apps'

19 Module 3: Get started with Power Apps

19.1 Lab 2: How to build a canvas app, part 2

19.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs. We expect to have our content fully up to date very soon.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

20 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visits are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In part 2 of this lab, you will create design and build a Power Apps canvas app that the security personnel will use at the building entrances to quickly confirm and register the visitors.

21 High-level lab steps

You will follow the below outline to design the canvas app:

- Create the app using the phone form factor
- Connect to Dataverse as a data source
- Capture the input (visitor code) and locate the visitor row
- Configure a form viewer control to show the visitor information
- Use a Dataverse view to populate the gallery
- Handle checking-in and checking-out process for a visitor

21.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**

21.2 Things to consider before you begin

- What information would a security officer need quick access to?
- What should happen if visitor code is invalid?
- What should happen if the visitor arrives outside of the scheduled hours?

22 Exercise #1: Create Security Canvas App

Objective: In this exercise, you will create a canvas app.

22.1 Task #1: Create Canvas App

1. Open your Campus Management solution.
    - Sign in to https://make.powerapps.com
    - If the Environment displayed in the top right is not your Practice environment, select your **Environment**.
    - Select **Solutions**.
    - Click to open your **Campus Management** solution.
2. Create new canvas application
    - Click **New** and select **App | Canvas App | Phone Form Factor**. This will open the App Editor in a New window.
    - Click **Skip** if presented with the Welcome to Power Apps Studio dialogue.
3. Save the canvas app
    - Click **File** and select **Save As**.
    - Check if **The cloud** is selected.
    - Enter [**Your Last Name**] **Campus Security** for Name and click **Save**.
    - Click the **Back** arrow at the top left (below Power Apps) to return to the app.
4. Connect to data source (Visits)
    - Click **View | Data sources**
    - Click **+ Add Data**
    - Click **See all entities** (or tables)

- Select **Visits** and wait for the Visit table to display on the Data tab.

5. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.

22.2 Task #2: Display Visitor information

1. Add search box
   - Select the **Tree View** tab on the left navigation bar.
   - Select **Screen1**.
   - Go to the **Insert** tab.
   - Click **Text** and select **Text input**.
2. Edit the text input object
   - While still selecting the Text input object, select the text in the **Default** property and clear the value.
   - Select **Hint Text** property and enter `"Enter visitor code"` as the value (including double quotes)
   - Click on **...** next to the control name in tree view (TextInput1), select **Rename**, change the name to `textCode`
3. Add a form view
   - On **Insert** tab click **Forms** then select **Display** (you may need to click the down arrow on the right side of the ribbon to see Forms)
   - Drag to position the form and align with the bottom of the screen
   - While still selecting the new form, select **DataSource** property and select **Visits**
   - In the properties pane select **Horizontal** as **Layout**
4. Edit form view
   - While still selecting the new form, click **Edit fields**
   - Remove both the **Name** and **Created On** fields
   - Click **Add field** and select the following fields: **Actual End**, **Actual Start**, **Building**, **Scheduled End**, **Scheduled Start**, **Visitor**
   - Press **Add**
   - Change the order of the selected fields by dragging the field cards in the list. Recommended order is: Visitor, Building, Scheduled Start, Scheduled End, Actual Start, Actual End (you can collapse the fields to make them easier to drag)
   - Click the **X** to close the Fields pane
5. While still selecting the form view, select the Advanced tab on the Properties pane. Select **Item** property and enter `LookUp(Visits, Code = textCode.Text)`
6. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.
7. Prepare to test the app
   - Switch to the browser tab containing the solution
   - Click **Done** in the pop-up window
   - Select **Visit** table
   - Select **Data** tab
   - Open the View Selector in the top right by clicking the current View name, **Active Visits**
   - Change the View to **All columns**
   - Locate a Visit row that does not have an Actual Start or Actual End value (i.e., both columns are blank). Select and copy the **Code** for this Visit.
8. Test the app

- Switch to the browser tab with the app, press **F5** or click the **Play** icon at the upper-right corner to preview the app.
- Paste the copied value into the search textbox, verify that the record is displayed in the form

9. Clear the search textbox contents.
10. Press **ESC** to exit the running app.

22.3 Task #3: Add Check In and Check Out Buttons

In this task, we will create buttons for the user to check in and check out of their Visit.

1. Save search results in a variable to reuse across the control
   - Select **textCode** control
   - In the properties pane, select the **Advanced** tab and select **OnChange** property
   - Enter the following expression `Set(Visit, LookUp(Visits, Code = textCode.Text))`

     This will save the visit in a global variable when a user searches in the textCode searchbox. That allows us to use the variable *Visit* throughout the app without the need to re-enter the entire lookup expression.
2. Add Check In Button
   - Select **Insert** tab
   - Click **Button**
   - In the properties pane, change the button **Text** property to ”`Check In`” (you can type within the existing quotes)
   - Click on **...** next to the button name in tree view (Button1), select **Rename**, change the name to `CheckInButton`
3. Add Check Out Button
   - Click **Button** on the Insert tab to insert another button
   - In the properties pane, change the button **Text** property to ”`Check Out`” (you can type within the existing quotes)
   - Rename the button as `CheckOutButton`
   - Position the buttons below the search box, with **Check In** above **Check Out**

22.4 Task #4: Enable and disable buttons depending on visit data

Once users have looked up the visit, we would like them to use the Check In button to check in for that visit. We would like to enable **Check In** button when the visit record has been located (not blank), record status is active, and the visit has not started yet, i.e. the actual start value is blank.

1. Select the **Check In button** and click on the **Display Mode** property of the button in the Properties tab
2. Enter the expression below in the function bar:

   ```
   If(!IsBlank(Visit)
   && Visit.Status = 'Status (Visits)'.Active
   && IsBlank(Visit.'Actual Start'),
       DisplayMode.Edit,
       DisplayMode.Disabled
   )
   ```

   The expression can be broken down as following:

   - **!IsBlank(Visit)** - visit record was found
   - **&&** - logical AND operator
   - **Visit.Status = ’Status (Visits)’.Active** status of the record is *Active*
   - **IsBlank(Visit.’Actual Start’)** - Active Start field does not have any data in it

- **DisplayMode.Edit, DisplayMode.Disabled** - If the above conditions are met, the button will become editable. If not, the button will remain disabled.

We would like to enable **Check Out** button when the visit record has been located (not blank), record status is active, and the visit has already started, i.e. the actual start value is not blank.

3. Select the Check Out button and click on the **Display Mode** property of the button in the Properties tab
4. Enter the expression below in the function bar:

   ```
   If(!IsBlank(Visit)
   && Visit.Status = 'Status (Visits)'.Active
   && !IsBlank(Visit.'Actual Start'),
       DisplayMode.Edit,
       DisplayMode.Disabled
   )
   ```

5. To preserve work in progress, click **File** then click **Save**. Use the back arrow to return to the app.
6. Press **F5** to run the app.
7. Both buttons should be disabled. Enter the code value you copied previously and press **Tab** to move the focus away from the textbox (or click outside of the textbox). The **Check In** button should become enabled.
8. Clear the search box contents.
9. Press **ESC** to exit the running app.

22.5 Task #5: Complete Check In and Check Out Process

To perform the check in and check out process we need to update Dataverse visit data as following:

- When visitor checks in, set *Actual Start* field to the current date and time
- When visitor checks out, set *Actual End* field to the current date and time.
- After check out, set the record status to inactive, indicating that the visit has been completed

1. Select **Check In** button.
2. Set **OnSelect** property on the Advanced tab to the following expression.

   ```
   Patch(
       Visits,
       Visit,
       {'Actual Start': Now()}
   );
   Refresh([@Visits]);
   Set(Visit, LookUp(Visits, Code = textCode.Text));
   ```

   This expression contains the following parts:

   - **Patch(Visits, Visit, {'Actual Start': Now()});**. *Patch* method updates **Visits** table, the row identified by **Visit** variable (which is the current visit). The expression sets the value of *Actual Start* column to the current date and time (*Now()* method).
   - **Refresh(**[**@Visits**]**);**. This expression refreshes the visit rows, as the underlying values have changed
   - **Set(Visit, LookUp(Visits, Code = textCode.Text));** This expression updates the *Visit* variable with fresh data from Dataverse.

     When a user clicks this button, the Actual Start of the Visit will be set to the current date and time and the data will refresh.

3. Select **Check Out** button.
4. Set **OnSelect** property on the Advanced tab to the following expression:

   ```
   Patch(
       [@Visits],
       Visit,
       {
   ```

```
            'Actual End': Now(),
            Status: 'Status (Visits)'.Inactive
        }
);
Refresh([@Visits]);
Set(Visit, LookUp(Visits, Code = textCode.Text));
```

When a user clicks this button, the Actual End will be set to the current date and time, the Status of the Visit will be set to Inactive, and the data will refresh.

5. To preserve work in progress, click **File** then click **Save**. Use the **Back** arrow to return to the app.
6. Press **F5** or click the Play button to run the app. Enter the code value you copied previously and press **Tab** to move the focus away from the textbox. The **Check In** button should become enabled.
7. Press **Check In** button. The following should happen:
   - **Actual Start** is set to the current date and time
   - **Check In** button is disabled
   - **Check Out** button is enabled
8. Press **Check Out** button.
   - **Actual End** is set to the current date and time
   - Both buttons are disabled
9. Clear the search box contents.
10. Press **ESC** to exit the running app.

22.6 Task #6: Add visual indicators

Usability of a mobile app significantly improves when visual indicators are provided. In this task, we will add an icon indicating if a visitor can be checked in or checked out.

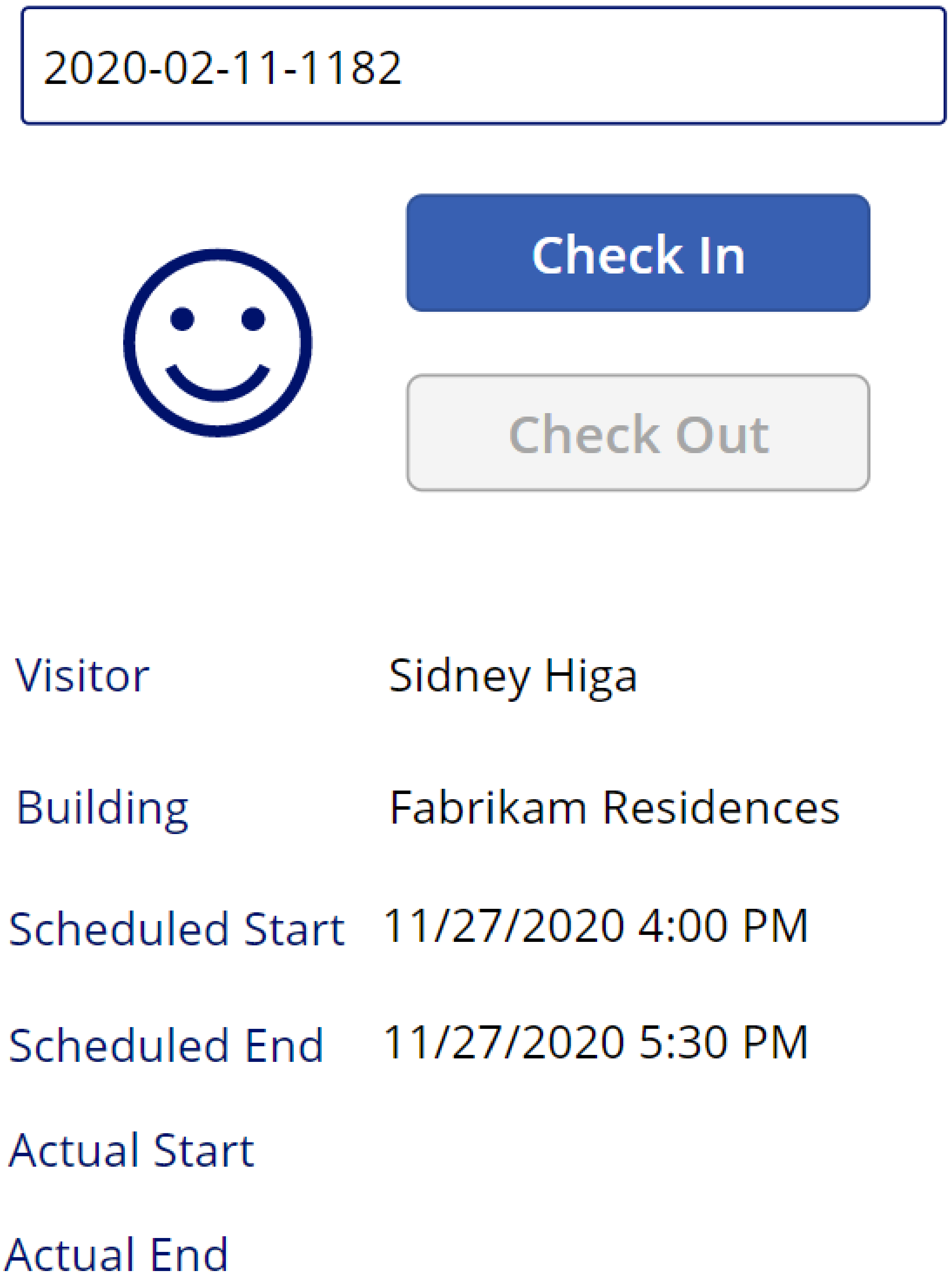
1. Select **Insert** tab
2. Select **Icons | Add**. Select an Icon. At this point it does not matter which icon we select as we want the value to be dynamic.
3. Resize and place the icon to the left of the buttons
4. In the Advanced tab for the Icon, select **Icon** property (in the Design section) and enter the following expression

```
If(
   CheckInButton.DisplayMode = DisplayMode.Disabled
&& CheckOutButton.DisplayMode = DisplayMode.Disabled,
    Icon.EmojiFrown,
    Icon.EmojiSmile
)
```

5. To preserve work in progress, click **File** then click **Save**. Use the **Back** arrow to return to the app.
6. Press **F5** to run the app. Enter the code value you copied previously and press **Tab** to move the focus away from the textbox. Verify the icon displays a frown emoji.
7. Find a different code value that has not been used before (it should not have an Actual Start or Actual End value).

   > You can navigate to the previous tab to copy another Code from one of the Visits you have created. You also have the option to run your **Campus Staff** app created previously to create new visit records. Verify the icon displays a smile emoji for this code.

Your running app should look approximately like the following:


2020-02-11-1182

Visitor	Sidney Higa
Building	Fabrikam Residences
Scheduled Start	11/27/2020 4:00 PM
Scheduled End	11/27/2020 5:30 PM
Actual Start	
Actual End	

8. Press **ESC** to exit the running app.

22.7 Task #7: Publish the app

1. You should still have the Campus Security app open in your browser. If not, select **Campus Security** app and click **Edit**.
2. Select **File | Publish**
3. Select **Publish this version**

23 Challenges

- Avoid manual entry of the visit code
- Add building validation for the visit
- Add validation of the visit actual time vs visit scheduled time (too early, too late, etc)
- Add detailed status of the visit, e.g. email display and validation for the visitor, reason for denying building access, etc
- Multiple buildings/meetings/checkings during a single campus visit. For example, someone may visit campus for a day and during that day they will meet staff members in multiple buildings at different time of the day. Would you consider bringing *appointment* entity into the solution?

23.1 lab: title: 'Lab 4: How to build a model-driven app' module: 'Module 3: Get started with Power Apps'

24 Module 3: Get started with Power Apps

24.1 Lab 3: How to build a model-driven app

24.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs. We expect to have our content fully up to date very soon.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

25 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visitors are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In this lab, you will build a Power Apps model-driven app to allow the backoffice campus staff to manage visit records across the entire campus.

26 High-level lab steps

As part of creating the model-driven app, you will complete the following:

- Create a new model-driven app named Campus Management
- Edit the app navigation to reference the required tables

- Customize the forms and views of the required tables for the app

We will work with the following components:

- **Views**: Views allow the user to display the existing data in the form table.
- **Forms**: This is where the user creates/updates new rows in the tables.

Both will be integrated to the model-driven app for a better user-experience.

26.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**

26.2 Things to consider before you begin

- What changes should we make to improve the user experience?
- What should we include in a model-driven app based on the data model we have built?
- What customizations can be made on the sitemap of a model-driven app?

27 Exercise #1: Customize Views and Forms

Objective: In this exercise, you will customize views and forms of the custom created tables that will be used in the model-driven app.

27.1 Task #1: Edit Visit Form

1. Sign in to https://make.powerapps.com if you are not already signed in.
2. Select your **environment.**
3. Select **Solutions**.
4. Click to open your **Campus Management** solution.
5. Click to open the **Visit** entity.
6. Select the **Forms** tab and click to open the **Main** form type.

   By default, the form has two fields: Name (Primary Field) and Owner.
7. Select + **Form field** and ddd the following fields below the **Owner** field by dragging columns to the form or simply clicking column names:
   - **Building**
   - **Visitor**
   - **Scheduled Start**
   - **Scheduled End**
   - **Actual Start**
   - **Actual End**
8. Drag the **Code** column and drop it in the form header.

   The header is the top right area of the form. You may need to minimize the Properties panel on the right side of the screen to see the field on the form.
9. With the **Code** field still selected, check the checkbox for **Read-only** in the Properties panel.
10. Select **Owner** field. In the Properties panel, change the **Field label** to **Host**
11. Click **Save** at the top right and wait for the save to complete.
12. Click **Publish** at the top right and wait for the publishing to complete.
13. Click **Back** at the top left of the screen. You should now be back to the Visit entity Forms Tab.

27.2 Task #2: Edit Visit Views

In this task, we will modify the default Active Visits view and create a new view for today's visits.

1. Select the **Views** tab and click to open the **Active Visits** view.
2. Add the following fields to the view by either clicking or dragging and dropping the fields:
   - **Code**
   - **Visitor**
   - **Building**
   - **Scheduled Start**
   - **Scheduled End**
3. Click the **Created On** column and select **Remove**. Field **Created On** will now be removed from the view.
4. Click the **Name** column and select **Remove**. Field **Name** will now be removed from the view.
5. In the Properties panel on the right, click **Sort by ...** and select **Scheduled Start**. Click on **Scheduled Start** again to change the order to descending.
6. Resize the individual column widths to fit the data.
7. Click **Save** and wait until the changes are saved.
8. Click **Publish** and wait for the publishing to complete.

Now, we will clone the view to create a new view for today's visits.

9. Press **Edit filters** link in the Properties panel.
10. Click **Add**, select **Add row**.
11. Select **Scheduled Start** as a field, then select **Today** as the condition in the drop-down.
12. Click the **...** on the **Status** row and click **Delete**.
13. Press **Ok** to save the condition. The view is now filtered to show only records where the Scheduled Start date is today.
14. Add **Actual Start** and **Actual End** fields to the view.

```
> **Note:** Since we no longer filter on the view status, we will get all today's visits including comp
```

15. Click on the **dropdown arrow** by the Save button (be careful not to press the button itself) and select **Save As**.
16. Change the name to **Today's Visits** and press **Save**.
17. Click **Publish** and wait for the publishing to complete.

28 Exercise #2: Create Model-Driven Application

Objective: In this exercise, you will create the model-driven app, customize the sitemap, and test the app.

You will see several fields not addressed as you build out your application, particularly on the sitemap steps. We have taken some short cuts in the interest of doing the labs. In a real implementation, you would give these items logical names.

28.1 Task #1: Create Application

1. Open your Campus Management solution if you are not already in it.
   - Sign in to https://make.powerapps.com
   - While in your environment, click to open your **Campus Management** solution.
2. Create the Model-Driven Application
   - Click **New** and select **App** and then **Model-driven app**. This will open a new tab.
   - Enter [**Your Last Name**] **Campus Management** for Name.

   - Select **Use existing solution to create the App** checkbox
   - Select **Next**
   - Select your **Campus Management** solution
   - Click **Done**
3. Click the pencil icon next to **Site Map.**
4. Edit the default titles
   - Select **New Area**.
   - Change the Title of the New Area to **Campus** in the properties pane on the right.
   - Select **New Group**.
   - Change the Title of the New Group to **Security** in the properties pane on the right.
5. Add the Contact table to the sitemap
   - Select **New Subarea**.
   - In the **Properties** pane, select **Entity** from the dropdown for **Type**.
   - Search for **Contact** table from the dropdown for **Entity**.
6. Add the Visit table to the sitemap
   - Select **Security** group and click **Add**.
   - Select **Subarea**.
   - Go to the **Properties** pane.
   - Select **Entity** from the dropdown for **Type** and search for **Visit** table from the dropdown for **Entity**.
7. Add the Building table to the sitemap
   - Select **Campus** area and click **Add**.
   - Select **Group**.
   - Enter **Settings** for **Title** in the **Properties** pane.
   - With the **Settings** Area still selected, click **Add**.
   - Select **Subarea**.
   - Go to the **Properties** pane.
   - Select **Entity** from the dropdown for **Type** and search for **Building** table from the dropdown for **Entity**.
8. Click **Save**. This will show the loading screen while the changes are getting saved.
9. Click **Publish** to publish the sitemap and wait for the publishing to complete.
10. Click **Save and Close** to close the sitemap editor.

```
> You will see the assets for the entities that were added to the sitemap are now in the application.
```

11. Click **Save** on the App Designer.
12. Click **Validate** to validate the changes done in the application.

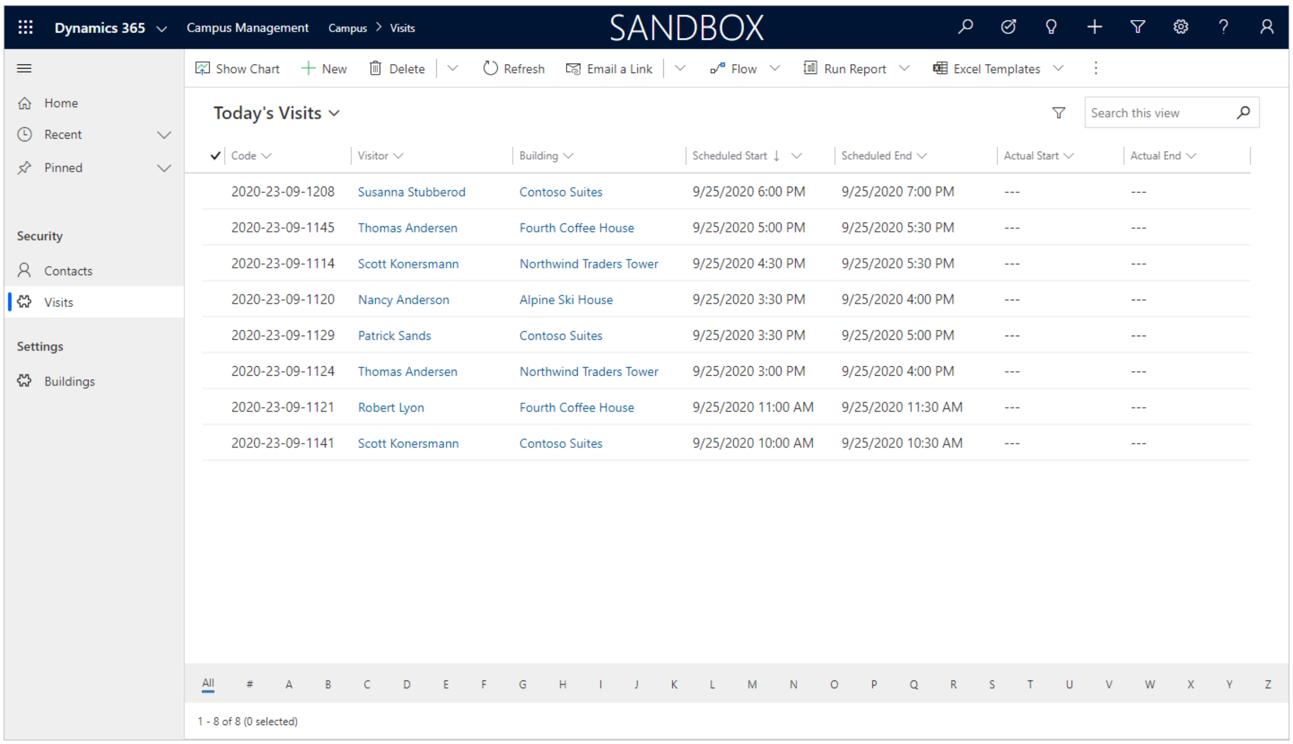
```
>  This will show some warnings but we can ignore them, since we have not referenced a specific View an
```

13. Click **Publish**
14. Click **Save and Close** to close the app designer.
15. Click **Done**.
16. Select **Solutions** and select **Publish all Customizations.**
17. Select **Apps** and your application should now be listed.

28.2 Task #2: Test Application

1. Start the application
   - Select **Apps** and click on your **Campus Management** app. (If you don't see your app at first, you may need to refresh your browser.)
   - The application should open in a new window.
2. Create new Contact
   - The app should open to the **Active Contacts** view
   - Click **New** from the top menu.
   - Provide **First Name** as `John` and **Last Name** as `Doe`.
   - Provide your personal email as **Email**. This will be used in a future lab.
   - Click **Save and Close**.
   - You should now see the created contact on the **Active Contacts** view.
3. Create new Building
   - Select **Buildings** from the sitemap.
   - Click **New**.
   - Enter the **Name** as `Microsoft Building`
   - Click **Save and Close**. This will show the newly created record on the Active Buildings View.
4. Create new Visit
   - Select **Visits** from the sitemap.
   - Click **New**.
   - Enter the fields as following
     - **Name**: `New test visit`
     - **Building**: select Microsoft Building
     - **Visitor**: select John Doe
     - **Scheduled Start**: select tomorrow's date and 2:00 PM as start time
     - **Scheduled End**: select tomorrow's date and 3:30 PM as end time
   - Click **Save and Close**. This will create the Visit and you should be able to see it on the Active Visits View.
   - Change view to **Today's Visits**. You should no longer see the new visit in the view, since it is scheduled for tomorrow.
5. You may add more test records.

   Your running app should look approximately like the following:



Code	Visitor	Building	Scheduled Start	Scheduled End	Actual Start	Actual End
2020-23-09-1208	Susanna Stubberod	Contoso Suites	9/25/2020 6:00 PM	9/25/2020 7:00 PM	---	---
2020-23-09-1145	Thomas Andersen	Fourth Coffee House	9/25/2020 5:00 PM	9/25/2020 5:30 PM	---	---
2020-23-09-1114	Scott Konersmann	Northwind Traders Tower	9/25/2020 4:30 PM	9/25/2020 5:30 PM	---	---
2020-23-09-1120	Nancy Anderson	Alpine Ski House	9/25/2020 3:30 PM	9/25/2020 4:00 PM	---	---
2020-23-09-1129	Patrick Sands	Contoso Suites	9/25/2020 3:30 PM	9/25/2020 5:00 PM	---	---
2020-23-09-1124	Thomas Andersen	Northwind Traders Tower	9/25/2020 3:00 PM	9/25/2020 4:00 PM	---	---
2020-23-09-1121	Robert Lyon	Fourth Coffee House	9/25/2020 11:00 AM	9/25/2020 11:30 AM	---	---
2020-23-09-1141	Scott Konersmann	Contoso Suites	9/25/2020 10:00 AM	9/25/2020 10:30 AM	---	---

29 Challenges

- Select specific views and forms for Visits and Buildings
- Security personnel typically work in a single building. How would you provide an easy way for them to display visits only for a selected building?
- Restrict access to specific entities, e.g. Buildings should be read-only for all staff members except the administrators
- What Dashboards would you consider adding to the app?

29.1 lab: title: 'Lab 5: How to build a Power Apps portal' module: 'Module 3: Get started with Power Apps'

30 Module 3: Get started with Power Apps

30.1 Lab 4: How to build a Power Apps portal

30.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs. We expect to have our content fully up to date very soon.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

31 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visits are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to provide the visitors with the information about the buildings on campus. The visitors will be able to view the buildings list on a website, which will be built using a Power Apps portal.

In this lab, you will provision a Power Apps portal and create a portals webpage that will show a listing of the buildings on campus.

32 High-level lab steps

You will follow the below outline to design the Power Apps portal:

- Provision a Power Apps portal in the Dataverse environment
- Create and configure a webpage to show a list of the buildings
- Create a new theme and apply it to the portal

32.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**

32.2 Things to consider before you begin

- Power Apps portals apps are always started from a template instead of a blank application. Your portal should have been created in Module 0 Lab 0. Once you provision a portal, it will already have pages, menus and a default theme.

33 Exercise #1: Create a Portal Webpage

Objective: In this exercise, you will create a new webpage that will display some static content as well as a list of buildings from Dataverse.

33.1 Task #1: Navigate to Portal

1. Navigate to https://make.powerapps.com.
2. Verify that you are in your Practice Environment. If you are not, change the environment at the top right.
3. Click on **Apps**
4. Locate the app that has the **Type** of **Portal**
5. Click on the app name to open the portal

   You should be redirected to your portal website landing page with a welcome message. Navigate your portal to see what was created by default when you provisioned your portal.

33.2 Task #2: Create a Webpage

1. Open Power Apps portals Studio
   - Sign in to https://make.powerapps.com (you may still have this open in your tabs)
   - Select **Apps**
   - Locate the app that has the **Type** of **Portal**
   - Click on the ellipses (**...**) to the right of the portals app name and choose **Edit**

   You are now in the Power Apps portals Studio. This is where you can modify and create portal content.
2. Create a new page
   - From the command bar, select **New page**
   - Mouse over **Fixed layouts** and choose **Page with title**
3. In the properties pane, under **Display** change the **Name** from **New page (1)** to `Building Directory`
4. In the **Partial URL** change the value to `building-directory`, press the Tab key (to initiate auto-save)

The title of the page should now read **Building Directory**

33.3 Task #3: Add Static Content

1. Add a section to the webpage
   - On the canvas (area showing webpage), select the **Page Copy** section. This is the large box around the 2 sentences of text in the middle of your page.
   - On the toolbelt (left side), select the **Components** icon
   - Choose **Two columns section** from the **Section layout** area
2. Add Static Text
   - On the canvas (area showing webpage), select the left column
   - On the toolbelt (left side), select the **Components** icon
   - Choose **Text** from the **Portal components** area
   - In the new text area, enter the following text:

     ```
     The following is the building directory.
     ```
   - Select the text box above the one you just edited, and click **Delete** on the command bar to remove the default text.
3. Add an Image
   - On the canvas (area showing webpage), select the right column
   - On the toolbelt (left side), select the **Components** icon
   - Choose **Image** from the **Portal components** area
   - In the properties pane, click **Select an image**. Locate and select the **Product A.png**
   - In the properties pane, click the **Formatting** section drop-down and change the **Width** to 70% (be sure to type the %). You can play around with the sizing of the image until it is as desired.
4. Click **Browse website** to view the page so far. Notice that there is now the **Building Directory** option on the main menu.

   You may need to configure your browser to allow pop-ups.

33.4 Task #4: Add a List Component

1. Navigate to the previous tab and continue to step #2. If not available, follow the below steps to return to this location.
   - Sign in to https://make.powerapps.com (you may still have this open in your tabs)
   - Locate the app that has the **Type** of **Portal**
   - Click on the ellipses (**...**) and choose **Edit**
   - On the toolbelt (left hand side), choose the **Pages** option
   - Locate and select the **Building Directory** page you created earlier
2. Add a list component to the Building Directory page
   - Select the section with two columns.
   - On the toolbelt (left side), select the **Components** icon
   - Choose **One column section** from the **Section layout** area (a section will appear below the image and text on the webpage)
   - Select the new column section on the canvas
   - On the toolbelt (left side), select the **Components** icon
   - Choose **List** from the **Portal Components** area (a list component will appear in the new section)

3. Configure the list component
   - Select the list component on the canvas
   - In the properties pane (right side), enter in `Buildings List` in the **Name** field
   - In the **Table** field, choose **Building (bc_building)** from the drop-down list
   - In the **Views**, choose **Active Buildings**
   - Leave the remaining default settings
4. Click **Browse website** to view the page.

   You should see the list of Buildings from your Dataverse database appear on the webpage.

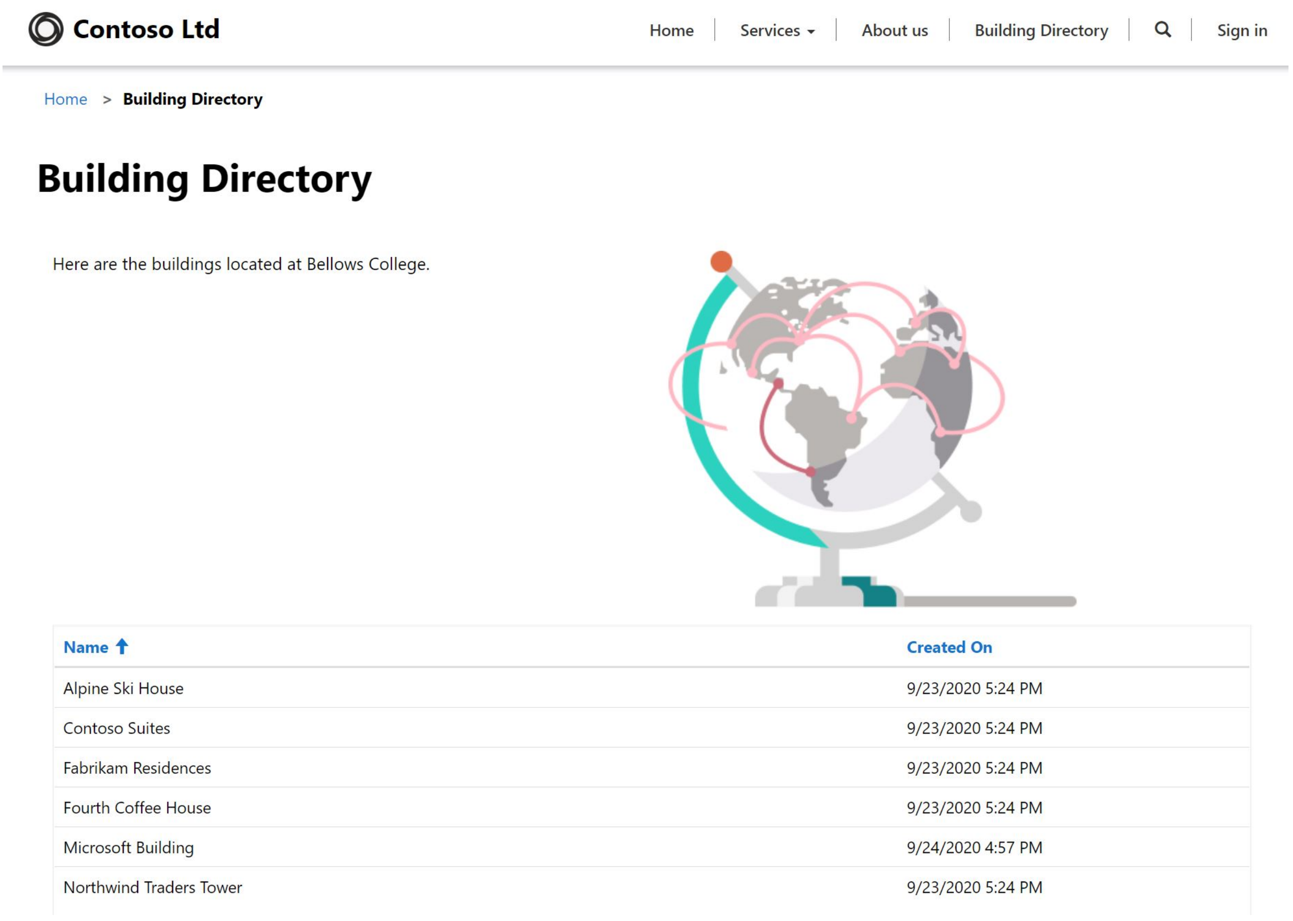
34 Exercise #2: Change the Portal Theme

Objective: In this exercise, you will create a new theme that will alter the color scheme of your portal.

34.1 Task #1: Apply and Edit a Theme

1. Navigate to the previous tab and continue to step #2. If not available, follow the below steps to return to this location.
   - Sign in to https://make.powerapps.com (you may still have this open in your tabs)
   - Locate the app that has the **Type** of **Portal**
   - Click on the ellipses (**...**) and choose **Edit**
2. Apply and customize a basic theme
   - On the toolbelt (left side), select the **Themes** icon
   - Click the toggle for **Enable basic theme** to turn this feature on.
   - On one of presets, click the ellipses (**...**) and choose **Customize**
   - A copy of the basic theme has been created.
   - On the properties pane, play around with changing the colors and exploring the impact of these changes to your portal.
   - Rename your theme
3. On the command bar, click **Sync configuration**

Your app layout should look similar to the following structure:



Building Directory

Here are the buildings located at Bellows College.

Name ↑	Created On
Alpine Ski House	9/23/2020 5:24 PM
Contoso Suites	9/23/2020 5:24 PM
Fabrikam Residences	9/23/2020 5:24 PM
Fourth Coffee House	9/23/2020 5:24 PM
Microsoft Building	9/24/2020 4:57 PM
Northwind Traders Tower	9/23/2020 5:24 PM

35 Challenges

- Create a different view of Buildings that just displays the Building Name. You will need to select **Browse website** from the Portal studio to see the changes.
- On the toolbelt, click on the **Themes** icon and edit the CSS of your custom theme.
- Create a page with the **Form** component and modify a **List** component to add or edit Dataverse rows with the form.
- Enable **Entity Permissions** in a **List** component **Settings**, what happens to the data?
- In the Portal studio, select the Source Code Editor icon `</>` to view the page source. If you are comfortable with HTML, make some modifications and view the results.

35.1 lab: title: 'Lab 6: How to build an automated solution' module: 'Module 4: Get Started with Power Automate'

36 Module 4: Get Started with Power Automate

36.1 Lab: How to build an automated solution

36.1.1 Important Notice (Effective November 2020):

Common Data Service has been renamed to Microsoft Dataverse. Some terminology in Microsoft Dataverse has been updated. For example, entity is now table. Fields and records in Dataverse databases are now referred to as columns and rows.

While the applications are in the process of updating their user experience, some references to terminology for Microsoft Dataverse like entity (now **table**), field (now **column**), and record (now **row**) may be out of date. Please keep this in mind as you work through the labs. We expect to have our content fully up to date very soon.

For more information and for a complete list of affected terms, please visit What is Microsoft Dataverse?

36.2 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visitors are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In this lab, you will create Power Automate flows to automate various parts of the campus management.

37 High-level lab steps

The following have been identified as requirements you must implement to complete the project:

- The unique code assigned to each visitor must be made available to them prior to their visit.
- Security personnel need to receive notifications of visitors overstaying their scheduled timeslots.

37.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**
- Campus Staff app created in **Module 3 Lab 2 – How to build a canvas app, part 2** (for testing)
- John Doe contact created with a personal email address in **Module 3 Lab 4 - How to build a model-driven app** (for testing)

37.2 Things to consider before you begin

- What is the most appropriate distribution mechanism for the visitor codes?
- How could overstays be measured and strict policies enforced?

38 Exercise #1: Create Visit Notification flow

Objective: In this exercise, you will create a Power Automate flow that implements the requirement. The visitor should be sent an email that includes the unique code assigned to the visit.

38.1 Task #1: Create flow

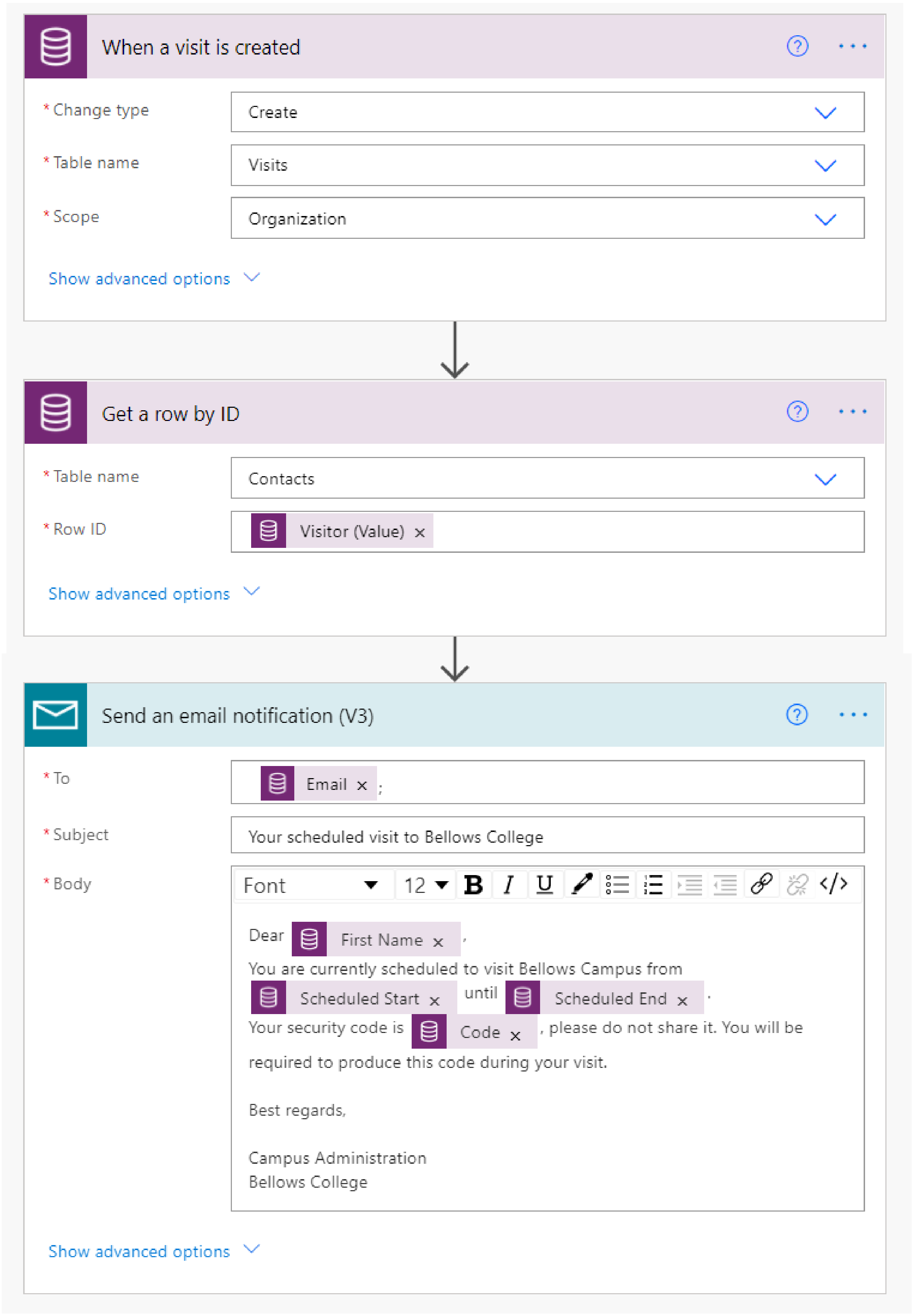
1. Open your Campus Management solution.
   - Sign in to https://make.powerapps.com
   - Select your **environment.**
   - Select **Solutions**.
   - Click to open your **Campus Management** solution.
2. Click **New** and select **Cloud flow**. This will open the Power Automate flow editor in a new window.
3. Search for *Current* and select **Common Data Service (Current Environment)** connector.
4. Select the trigger **When a Record is Created, Updated or Deleted**.
   - Select **Create** for **Trigger condition**
   - Select **Visits** for **Table name**
   - Select **Organization** for **Scope**
   - On the trigger step, click the ellipsis (**...**) and click **Rename**. Rename this trigger **"When a visit is created"**. This is a good practice, so you and other flow editors can understand the purpose of the step without having to dive into the details.
5. Click **New Step**. This step is required to retrieve visitors information, including email address.

6. Search for *Current* and select **Common Data Service (Current Environment)** connector.
7. Select **Get a row by ID** action.
   - Select **Contacts** as **Table name**
   - In the **Row ID** field, select **Visitor (Value)** from the Dynamic content list.
   - On this action, click the ellipsis (**...**) and click **Rename**. Rename this action **"Get the Visitor"**. This is a good practice, so you and other flow editors can understand the purpose of the step without having to dive into the details.
8. Click **New Step**. This is the step that will create and send email to the visitor.
9. Search for *mail*, select **Mail** connector and **Send an email notification** action
   - If asked to Accept terms and conditions for using this action, click **Accept**.
   - Select **To** field, select **Email** from the Dynamic content list. Notice that it is beneath the **Get the Visitor** header. This means you are selecting the Email that is related to the Visitor that you looked up in the previous step.
   - Enter **Your scheduled visit to Bellows College** in the **Subject** field.
   - Enter the following text in **Email Body**:

     > Dynamic content needs to be placed where fields are named in brackets. It is recommended to copy & paste all text first and then add dynamic content in the correct places.

     ```
     Dear {First Name},

     You are currently scheduled to visit Bellows Campus from {Scheduled Start} until {Scheduled En

     Your security code is {Code}, please do not share it. You will be required to produce this cod

     Best regards,

     Campus Administration
     Bellows College
     ```
10. Select the **Untitled** flow name at the top and rename it to `Visit notification`
11. Press **Save**

    Leave this flow tab open for the next task. You flow should look approximately like the following:



When a visit is created

* Change type	Create
* Table name	Visits
* Scope	Organization

Show advanced options

Get a row by ID

* Table name	Contacts
* Row ID	Visitor (Value)

Show advanced options

Send an email notification (V3)

* To	Email ;
* Subject	Your scheduled visit to Bellows College

* Body

Dear First Name ,
You are currently scheduled to visit Bellows Campus from Scheduled Start until Scheduled End .
Your security code is Code , please do not share it. You will be required to produce this code during your visit.

Best regards,

Campus Administration
Bellows College

Show advanced options

38.2 Task #2: Validate and test the flow

1. Open a new tab in your browser and navigate to https://make.powerapps.com
2. Click **Apps** and select the **Campus Staff** app you created
3. Leaving this tab open, navigate back to the previous tab with your flow.
4. On the command bar, click **Test**. Select **Manually** and then **Save & Test**.
5. Leaving the flow tab open, navigate back to the previous tab with the **Campus Staff** app.
6. Press + to add a new Visit record
7. Enter **John Doe** as **Name** and choose any **Building**
8. Choose **John Doe** as the **Visitor**
9. Choose the **Scheduled Start** and **Scheduled End Dates** to any dates in the future.
10. Press the **Checkmark** icon to save the new visit
11. Navigate back to the previous tab with the flow being tested. Watch as the flow is run. If there are any errors, go back and compare your flow to the example above. If the email is sent successfully, you will receive it in your inbox.
12. Click the back arrow on the command bar
13. In the **Details** section, notice that the **Status** is set to **On**. This means your flow will run whenever a new Visit is created, until you turn it off. Any time the flow runs, you will see it added to the **28-day run history** list.
14. Turn the flow off by clicking **Turn off** on the command bar. You may need to press the ellipses (**...**) to see this option.
15. Close this window.

39 Exercise #2: Create Security Sweep flow

Objective: In this exercise, you will create a Power Automate flow that implements the requirement. A security sweep needs to be performed every 15 minutes, and security should be notified if any of the visitors overstayed their scheduled time.

39.1 Task #1: Create flow to retrieve records

1. Open your Campus Management solution.
   - Sign in to https://make.powerapps.com
   - Select your **Environment.**
   - Select **Solutions**.
   - Click to open your **Campus Management** solution.
2. Click **New** and select **Cloud flow**. This will open the Power Automate flow editor in a new window.
3. Search for *recurrence*, select **Schedule** connector, and then select the **Recurrence** trigger.
4. Set **Interval** to **15 minutes**
5. Click **New step**. Search for *Current* and select **Common Data Service (Current Environment)** connector. Select **List rows** action.
   - Enter **Visits** as **Table name**
   - Click **Show advanced options**
   - Enter the following expression as **Filter rows**

   ```
   statecode eq 0 and bc_actualstart ne null and bc_actualend eq null and Microsoft.Dynamics.CRM.Ol
   ```

   - To break it down:
     - **statecode eq 0** filters active visits (where Status equal Active)

  - **bc__actualstart ne null** restricts search to visits where Actual Start has a value, i.e. there was a checkin
  - **bc__actualend eq null** restricts search to visits where there was no check out (Actual End has no value)
  - **Microsoft.Dynamics.CRM.OlderThanXMinutes(PropertyName='bc__scheduledend',PropertyV** restricts visits where visits meant to complete more than 15 minutes ago.

- On this action, click the ellipsis (**...**) and click **Rename**. Rename this action **"List active visits that ended more than 15 minutes ago"**. This is a good practice, so you and other flow editors can understand the purpose of the step without having to dive into the details.

6. Click **New step**. Search for *Apply*, select **Apply to each** action
7. Select **value** from dynamics content in the **Select an output from previous steps** field. Notice that it is beneath the **List active visits that ended more than 15 minutes ago** gray header. This means you are selecting the list of visits that you looked up in the previous step.
8. Retrieve Building data for related record
   - Click **Add an action** inside the Apply to Each loop.
   - Search for *Current* and select **Common Data Service (Current Environment)** connector.
   - Select **Get a row by ID** action.
   - Select **Buildings** as **Entity name**
   - Select **Building (Value)** as **Item ID** from the Dynamic content
   - Click **...** beside **Get a record**, select **Rename**. Enter **Get building** as step name
9. Retrieve Visitor data for related record
   - Click **Add an action** inside the Apply to Each loop.
   - Search for *Current* and select **Common Data Service (Current Environment)** connector.
   - Select **Get a row by ID** action.
   - Select **Contacts** as **Entity name**
   - Select **Visitor (Value)** as **Item ID** from the Dynamic content
   - Click **...** beside **Get a record**, select **Rename**. Enter **Get visitor** as step name
10. Send email notification
    - Click **Add an action** inside the Apply to Each loop. Add **Send an email notification** action from **Mail** connection.
11. Enter your email address as **To**
12. Enter the following in the **Subject** field. **Full Name** is a dynamic content from the **Get visitor** step.

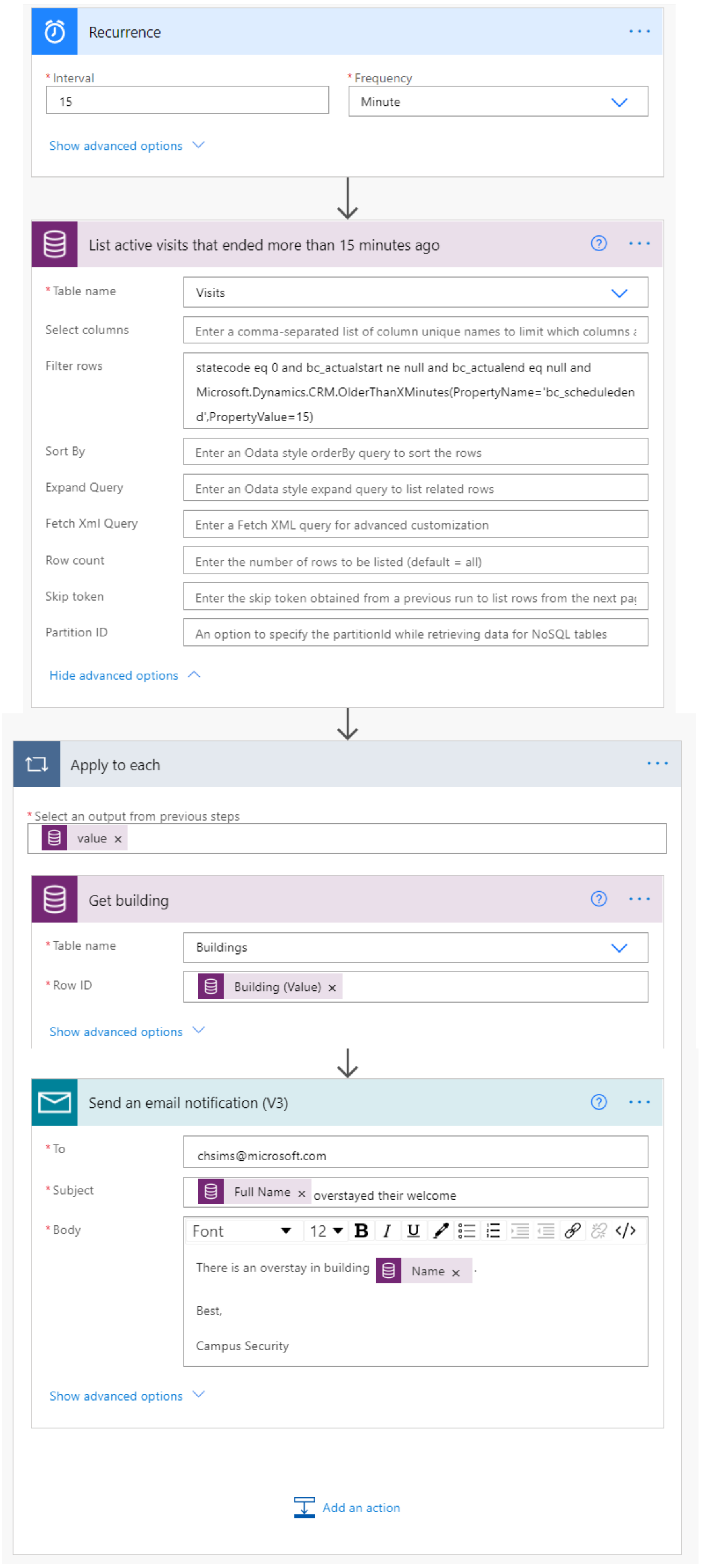
```
{Full Name} overstayed their welcome
```

14. Enter the following in the **Body** field. **Name** is a dynamic content from **Get building** step.

```
There is an overstay in building {Name}.

Best,

Campus Security
```

17. Select flow name **Untitled** in the upper left corner and rename it to **Security Sweep**
18. Press **Save**

```
Your flow should look approximately like the following:
```



Recurrence

* Interval	15
* Frequency	Minute

Show advanced options

List active visits that ended more than 15 minutes ago

* Table name	Visits
Select columns	Enter a comma-separated list of column unique names to limit which columns a
Filter rows	statecode eq 0 and bc_actualstart ne null and bc_actualend eq null and Microsoft.Dynamics.CRM.OlderThanXMinutes(PropertyName='bc_scheduledend',PropertyValue=15)
Sort By	Enter an Odata style orderBy query to sort the rows
Expand Query	Enter an Odata style expand query to list related rows
Fetch Xml Query	Enter a Fetch XML query for advanced customization
Row count	Enter the number of rows to be listed (default = all)
Skip token	Enter the skip token obtained from a previous run to list rows from the next pa
Partition ID	An option to specify the partitionId while retrieving data for NoSQL tables

Hide advanced options

Apply to each

* Select an output from previous steps: value ×

Get building

* Table name	Buildings
* Row ID	Building (Value) ×

Show advanced options

Send an email notification (V3)

* To	chsims@microsoft.com
* Subject	Full Name × overstayed their welcome
* Body	There is an overstay in building Name × . Best, Campus Security

Show advanced options

Add an action

39.2 Task #2: Validate and test the flow

Your flow will begin sending you emails (to the email you specified when creating the John Doe contact previously) if there are visits that meet the requirements laid out in the flow.

1. Validate that you have visit records that:

    1. Have active status

    2. Scheduled End is in the past (by more than 15 minutes)

    3. Actual Start has a value.

        Note: To view this data, navigate to make.powerapps.com in a new tab. Click Solutions on the left pane to locate your solution. Select the Visit entity, then select the Data tab. Click Active Visits in the top right-hand corner to display the view selector, then select All fields.

2. Navigate to your **Security Sweep** flow, if not already there.

3. When your flow opens, click **Test**.

4. Select **Manually**.

5. Click **Save & Test** and **Run Flow**.

6. When flow competes, click **Done**.

7. Expand **Apply to each**, then expand the **Send an email notification** step. Check the **Subject**, **Email Body** values.

8. Select the back arrow to the Security Sweep flow details. Select **Turn off** on the command bar. This is to prevent flow from executing on a schedule on the test system.

40 Challenges

- Add Actual Start and Scheduled End to the email body.
- How could you ensure user-friendly date formatting is used in the email body?
- Is it possible to generate a table with overstay information and send only a single email?
- Can you generate barcode for the visit code? When will that be useful?

40.1 lab: title: 'Lab 7: How to build a simple dashboard' module: 'Module 5: Get Started with Power BI'

41 Module 5: Get Started with Power BI

41.1 Lab: How to build a simple dashboard

42 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visitors are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Campus administration would like to modernize their visitor registration system where access to the buildings is controlled by security personnel and all visits are required to be pre-registered and recorded by their hosts.

Throughout this course, you will build applications and perform automation to enable the Bellows College administration and security personnel to manage and control access to the buildings on campus.

In this lab, you will build a Power BI dashboard that visualizes data about campus visits.

43 High-level lab steps

We will follow the below steps to design and create the Power BI dashboard:

- Connect to Dataverse

- Transform the data to include user-friendly descriptions for the related rows (lookups)
- Create and publish a report with various visualizations of the campus visits information
- Utilize a user natural language query to build additional visualizations
- Build a mobile view of the Power BI dashboard

43.1 Prerequisites

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**

43.2 Things to consider before you begin

- Who is the target audience of the report?
- How will the audience consume the report? Typical device? Location?
- Do you have sufficient data to visualize?
- What are the possible characteristics you can use to analyze data about the visits?

44 Exercise #1: Create Power BI Report

Objective: In this exercise, you will create a Power BI report based on data from your Dataverse database.

44.1 Task #1: Install Power BI Desktop / Prepare Power BI service

1. Follow the below instructions to setup Power BI:
   - If Power BI Desktop is **already** installed, please skip to Task #2.
   - If you do not have Power BI Desktop installed, complete **Step #2**.
   - If you do not have required permissions or encounter issues with running Power BI Desktop, continue to **Step #4**.
2. Navigate to https://aka.ms/pbidesktopstore to download and install Power BI Desktop.

   > [!IMPORTANT] If you experience issues installing Power BI Desktop using Microsoft Store, try standalone installer that can be downloaded from https://aka.ms/pbiSingleInstaller.
3. If you successfully installed Power BI Desktop, you can now skip to Task #2; otherwise, continue to the next step.

   > If you do not have required permissions to install desktop applications or experience difficulties in running or configuring Power BI Desktop, complete the task steps below.
4. Download visits.pbix and save on your computer.
5. Navigate to https://app.powerbi.com/ and click **Sign in**.
6. Click **My Workspace**.
7. If presented with the **Get Data** page, click **Skip**.
8. Expand **+New** and select **Upload a file**.

   > [!IMPORTANT] If you don't see **+New**, you may need to activate the new look for Power BI. Be sure to toggle the **New look** to **On** at the top of your screen.
9. Select **Local File**.
10. Locate and select **visits.pbix** file you've downloaded earlier.
11. Once data load is complete, select **visits** report (notice that the Type is set to **Report**).
12. Click **Edit**. If **Edit** menu item is not visible click **...** and then select **Edit**.
13. You have now setup Power BI service to use for your labs. Continue to Task #3, but use the online Power BI service at https://app.powerbi.com instead of Power BI Desktop throughout the lab.

44.2 Task #2: Prepare Data

1. Find out your organization URL
   - In a new tab, navigate to Power Platform Admin Center at https://admin.powerplatform.com
   - In the left navigation page, select Environments, and then open your Practice environment.
   - Right mouse click **Environment URL** on the **Details** panel, then select **Copy link address**.
2. Open Power BI Desktop, sign in with your provided credentials if prompted.
3. Select **Get data**.
4. Select **Power Platform** on the left, then select **Common Data Service**, and press **Connect**.
5. Paste the environment URL you copied earlier into the **Server URL** field, press **OK**.
6. Expand **Entities** node, select **bc_Building** and **bc_Visit** entities, click **Load**.
7. Click **Model** icon on the left vertical toolbar.
8. Drag **bc_buildingid** column from **bc_Building** table and drop it to **bc_building** column in **bc_Visit** table. That will create a relationship between the two tables that Power BI will be able to use to display related data.
9. Select **Report** icon on the left toolbar.
10. Expand **bc_Visit** node in the **Fields** panel.
11. Click **...** beside **bc_Visit** and select **New Column**.
12. Complete the formula as following:

    ```
    Column = RELATED(bc_Building[bc_name])
    ```

    and press ENTER. That will add a new field with the building name into the visits data.
13. Click **...** next to the **Column** field that you just created and select **Rename**. Enter **Building** as the field name.
14. Click **...** next to the **bc_visitid** field and select **Rename**. Enter **Visit** as the field name.
15. Click **...** next to the **bc_scheduledstart** field and select **Rename**. Enter **Start** as the field name.
16. Save work in progress by pressing **File | Save** and entering a filename of your choice.

44.3 Task #3: Create Chart and Time Visualizations

1. Press the pie chart icon in the **Visualizations** panel to insert a chart.
2. Drag the **Building** field and drop it into **Legend** box.
3. Drag the **Visit** field and drop it into **Values** target box.
4. Resize the pie chart using corner handles so that all chart components are visible.
5. Click on the report outside of the pie chart to deselect it and select stacked column chart in **Visualizations** pane.
6. Drag **Visit** field and drop it into **Values** target box.
7. Drag **Start** field and drop it into **Axis** target box.
8. In the Visualizations pane, click **x** next to **Day** and **Quarter** to leave only **Year** and **Month** totals for the Axis.
9. Resize the chart as desired using the corner handles.
10. Test the report interactivity:
    - Select various building slices on the pie chart and observe changes on the time report.
    - Click on the column chart. Press the down arrow to turn on **Drill down** mode, then press the column to drill down to the next level (months). Another way to do this is to click **Data/Drill | Expand next level** on the ribbon.

- Drill up and down and select various bars on the time column chart to observe changes on the pie report.

11. Save work in progress by pressing **File | Save**.

45 Exercise #2: Create Power BI Dashboard

45.1 Task #1: Publish Power BI Report

1. Press **Publish** button on the Home tab of the ribbon.
2. Select **My workspace** as the destination, then press **Select**.
3. Wait until publishing is complete and click **Open <name of your report>.pbix in Power BI**.

45.2 Task #2: Create Power BI Dashboard

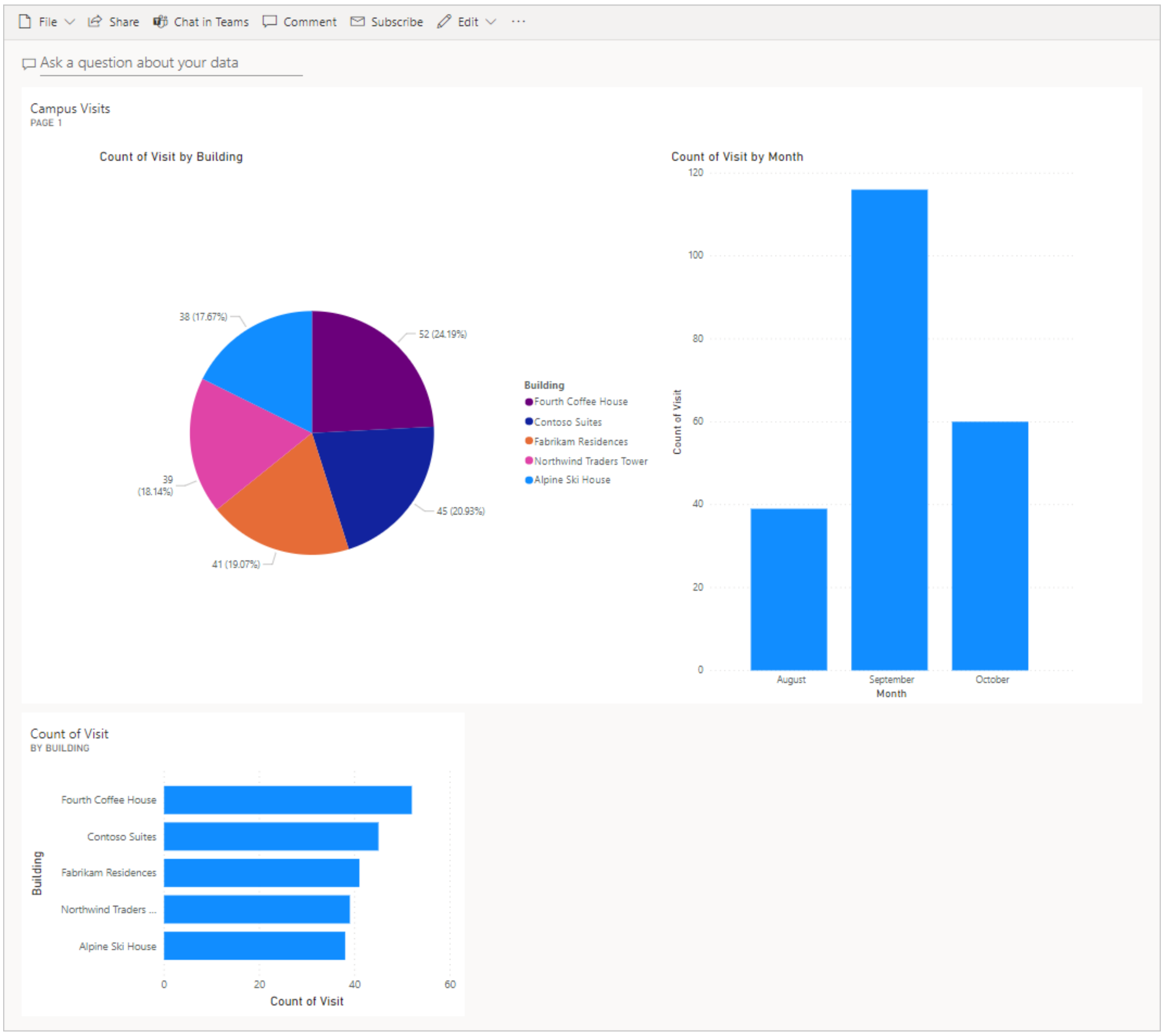
1. You should have the report open from the previous task.
2. Select **Pin to a dashboard** on the menu. Depending on the layout you may need to press **...** to show additional menu items.
3. Select **New dashboard** on **Pin to dashboard** prompt.
4. Enter [**Your Last Name**] **Campus Management** as a **Dashboard name**, press **Pin live**.
5. Select **My workspace** at the top, select [**Your Last Name**] **Campus Management** dashboard.
6. Test interactivity of the pie and bar charts displayed.

45.3 Task #3: Add Visualizations Using Natural Language

1. Within your **Campus Management** dashboard, select **Ask a question about your data** bar at the top.
2. Enter **buildings by number of visits** in Q&A area. The bar chart will be displayed.
3. Select **Pin visual**.
4. Select **Existing dashboard**, select your [**Your Last Name**] **Campus Management** dashboard, press **Pin**.
5. Click **Exit Q&A**.

Your [**Your Last Name**] **Campus Management** dashboard should be displayed. You may have to scroll down to see the new Q&A visual.

Your dashboard should look similar to the following:



45.4 Task #4: Build Mobile Phone View and Share a Report with a QR Code

1. In the Dashboard, select **Edit** | **Mobile View**.
2. Rearrange tiles as desired.
3. Click **Phone view** at the top right and change the View to **Web view**.
4. Select **My Workspace** at the top, and select your **Report**.
5. Select **Edit** and then select **...** | **Generate QR Code**.
6. *Optional:* If you have a mobile device, scan the code using a QR scanner app available on both iOS and Android platforms, or the camera app if your phone supports it. Log in to your account if prompted. Navigate and explore the report on a mobile device.

46 Challenges

- Dashboards and reports to include your campus and building plans
- Report and analyze visiting patterns and trends
- Overstaying visualization
- Streaming Power BI for near real-time processing for a large campus

46.1 lab: title: 'Lab 8: How to build a basic chatbot' module: 'Module 6: Intro to Power Virtual Agents'

47 Module 6: Intro to Power Virtual Agents

47.1 Lab: How to build a basic chatbot

48 Scenario

Bellows College is an educational organization with multiple buildings on campus. Campus visits are currently recorded in paper journals. The information is not captured consistently, and there are no means to collect and analyze data about the visits across the entire campus.

Like most organizations, Bellows College is quickly responding to concerns with misinformation about COVID-19, best practices, schedules, and so on. In this lab, you will build a Power Virtual Agent chatbot that will point to the Center for Disease Control page with questions and answers regarding the current status of the pandemic. The college will want this setup so that it can be embedded on their portal site, as well made available ad hoc as departments make their own planned reopening.

48.1 High level steps

We will follow the below outline to build our Power Virtual Agent:

- Sign up for a trial of Power Virtual Agent
- Build a bot using FAQs
- Test the bot
- Change the default greeting
- Publish the bot
- **Bonus Challenge:** Embed bot in your portal

48.2 Prerequisites

The following have been identified as requirements you must implement to complete the project:

- Completion of **Module 0 Lab 0 - Validate lab environment**
- Completion of **Module 2 Lab 1 - Introduction to Microsoft Dataverse**
- Bonus exercise only: Completion of **Module 6 Lab 4 - Introduction to Power Apps portals**

48.3 Things to consider before you begin

Bots can be very useful in many different scenarios. Based on what you know so far about Bellows College, consider where else in the organization a bot might be of use.

49 Exercise #1: Sign up for PVA and Create a New Bot

In this exercise, you will sign up for Power Virtual Agents trial.

1. Navigate to Power Virtual Agents
2. Click **Start Free Trial**.
3. Sign in, if required.
4. The **Create new bot** window should appear.
5. Enter **Crisis Bot** for **Name** and select a language.
6. Select your Practice environment to create the bot in and click **Create**. Wait for the bot to be created. Click **Explore bot** if prompted.
7. Test the bot. Type **Hello** in the message box and click **Send**. The bot should greet you and tell you what it can do.

8. Close the **Chat**.
9. Select **Topics**. The bot comes with some sample user topics and some system topics. The default greeting came from the system topics.

   In the next exercise, you will generate your own topics from the CDC FAQ site. Do not navigate away from this browser window.

50 Exercise #2: Create Topics

In this exercise, you will generate topics from the CDC FAQ site.

1. In a new tab, navigate to the CDC FAQ site and examine what is on the site. You will generate your topics from these FAQs.
2. Copy the URL.
3. Go back to Power Virtual Agents and make sure you still have **Topics** selected.
4. Select the **Suggested** tab, below **Topics**.
5. Click **Get started.**
6. Paste the URL you copied in the **Link to online content** textbox and click **Add**.
7. Click **Start** and wait. This can take a few minutes.
8. You should get some suggested topics created for you. Click to open one of the suggested topics.
9. You should see the trigger phrase and what the bot reply will be. **Click Add to topics.**
10. The suggested topic should be added to your topics. Select all the suggested topics and click **Add to topics**

    > You can select all topics by using the icon to the left of the Name column. If you receive an error message, please try again.

11. Once the Suggested topics have been added, select the **Existing** tab. You should see the new topics with their status set to Off.
12. Use the toggle button in the **Status** column to turn some of the topics to **On**.
13. Write down the trigger phrase for one of the topics that you have turned on so that you can test with this later.

    Do not navigate away from this browser window.

51 Exercise #3: Test Topics

In this task, you will test the topics you added.

1. Click **Test your bot** at the bottom left.
2. Click **Reset**.
3. Type the trigger phrase that you recorded from the previous task and click **Send**.
4. The bot should give you the correct information and ask if it answered your question. Click **Yes**.
5. The bot should ask you the rate how it did. Give it an excellent rating.
6. The bot should ask if it can help you with anything else. Click **No, thanks**.
7. The bot should conclude the chat session.
8. Type **hello** and click **Send**.
9. The bot should greet you and tell you what it can do. Your bot can now help users with COVID-19 FAQs, so you will need to change the greeting message in the next task. Do not navigate away from this browser window.

52 Exercise #4: Change the Greeting

In this task, you will change the greeting to COVID-19 specific.

1. Make sure you have **Topics** selected and select the **Existing** tab.
2. Collapse the **User Topics** section.
3. Click to open the **Greeting** topic of the System Topics. You can also use the search box to **Search existing topics**.
4. The greeting topic has 52 trigger phrases, click **Go to authoring canvas**.
5. Go to the first message and replace it with `Hi, I'm a virtual agent. I can tell you about how COVID-19 spreads, how to protect yourself, preparing your home and family for COVID-19, symptoms, testing, and more.`
6. Click **Save**.
7. Click **Test Bot** if your bot is not still open. Click **Reset** to reset the chat.
8. Type hello and click **Send**.
9. The bot should now reply with the new greeting.

53 Exercise #5: Publish the Bot

In this exercise, you will publish the bot.

1. Select **Publish** on the left navigation bar.
2. Click **Publish**.
3. Click **Publish** again and wait for the publishing to complete.
4. Expand **Manage** on the left navigation bar and select **Channels**.
5. You will get list of available channels you can publish your bot to. Select **Demo website**.
6. Change the welcome message to `Try my COVID-19 FAQ bot.`
7. Enter the following in **Conversation starters**:

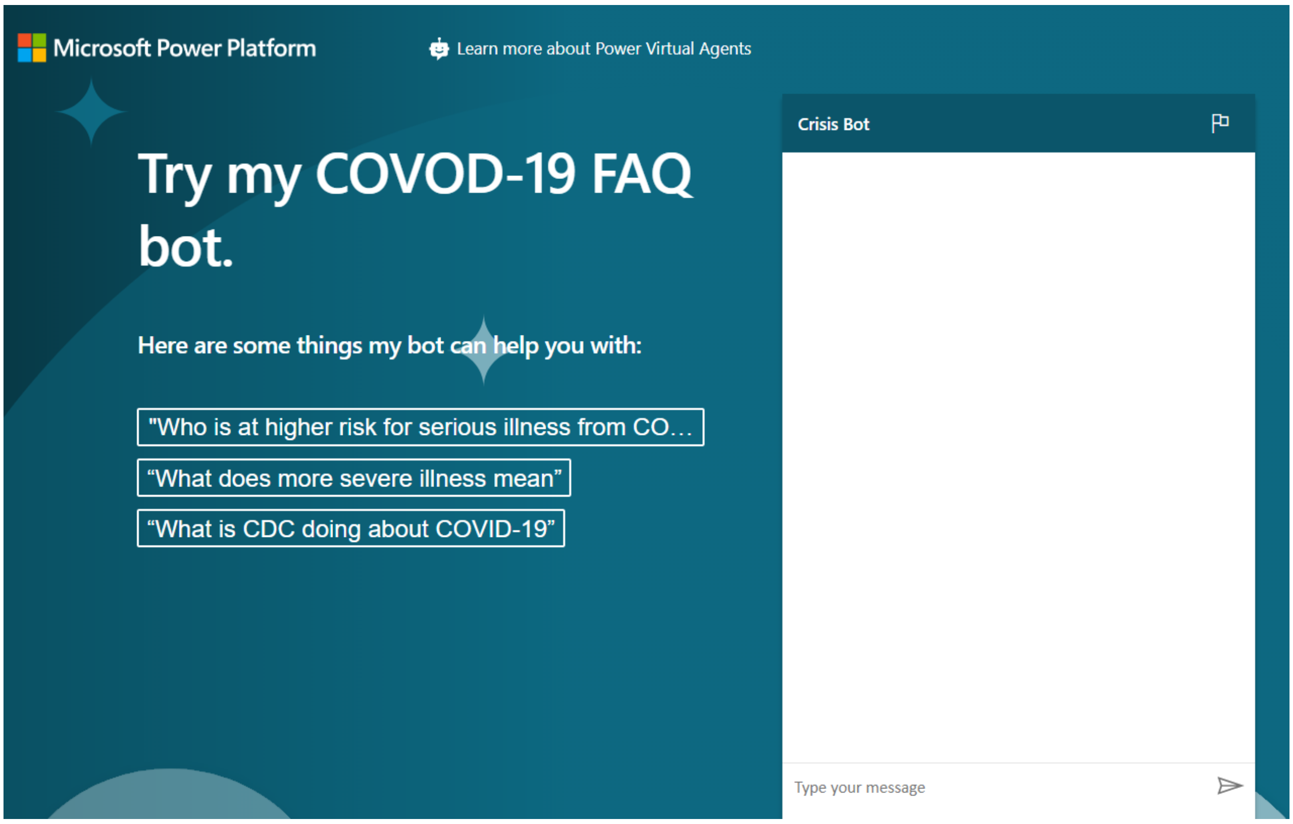
   ```
   "Who is at higher risk for serious illness from COVID-19"
   "What does more severe illness mean"
   "What is the CDC doing about COVID-19"
   ```

8. Click **Save**.
9. Copy the **URL**.

   You can share the URL with your colleagues and get feedback from them.

10. Start a new browser window or tab and navigate to the URL you copied. The demo website should like the image below.
11. Go ahead and start chatting with the bot.

When completed, your published bot should look similar to this:


Microsoft Power Platform
Learn more about Power Virtual Agents
Try my COVOD-19 FAQ bot.
Here are some things my bot can help you with:
"Who is at higher risk for serious illness from CO…
“What does more severe illness mean”
“What is CDC doing about COVID-19”
Crisis Bot
Type your message

54 Challenges

- Embed your chatbot on your Bellows College Visitors portal (more information on how to do this under **Add bot to Power Apps** here.)