

# Power Platform Workshop

## Scenario

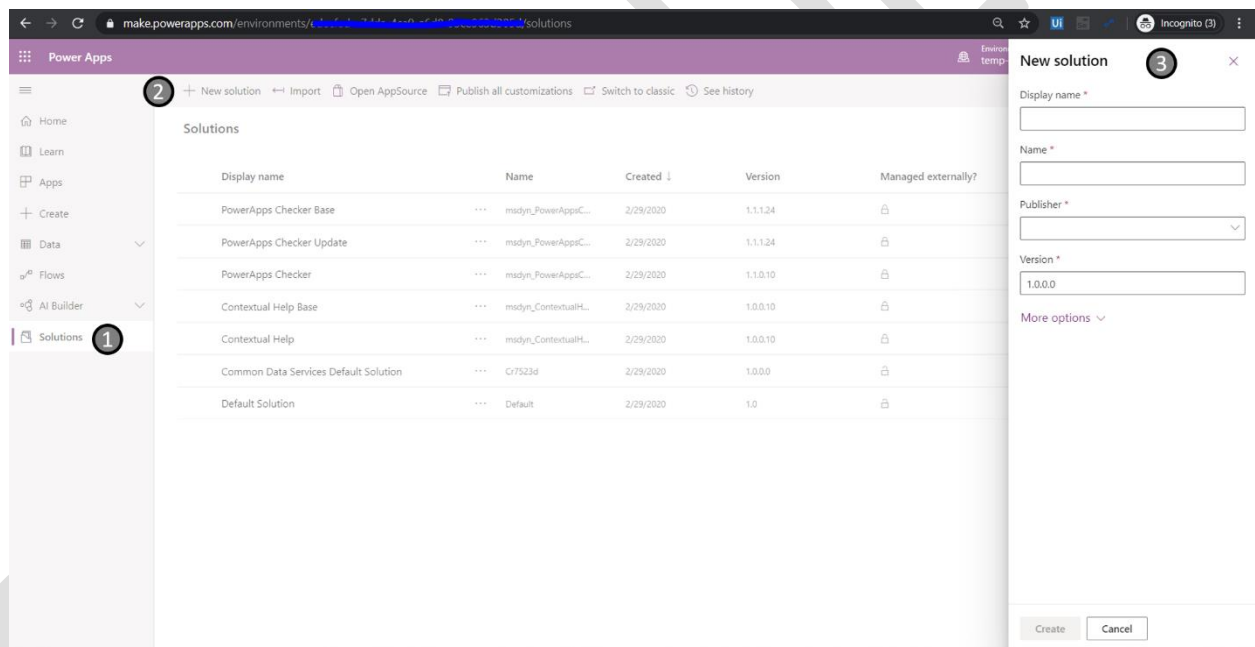
In this workshop, we will build a low code no code solution using the entire Power Platform. Towards the end, we will look at:

1. A service front that is built using PVA that interacts with a user and gathers information.
2. A flow in Power Automate that validates this information and stores it on a data source (SharePoint/ CDS).
3. A Canvas App that allows internal users to perform CRUD operations on the data.
4. A PowerBI report to visualize the data.
5. Lastly, the PVA will be installed on a web page, the canvas app, model driven app and the Power BI report on a Teams channel for better collaboration on work.

## Module 1: Data

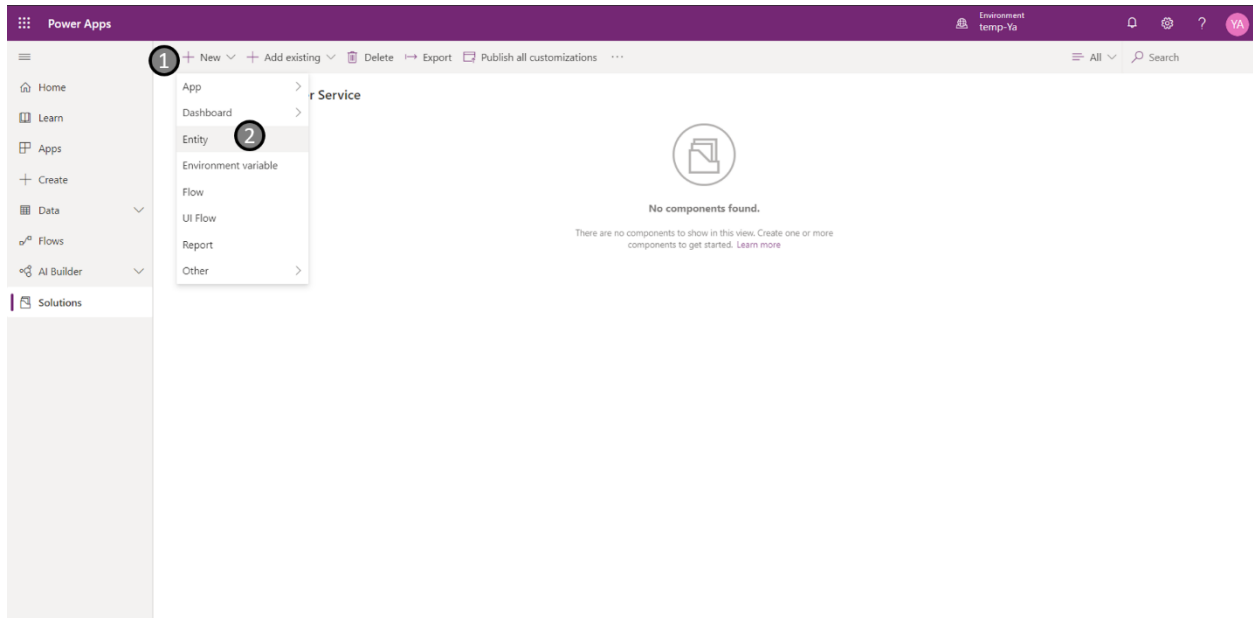
In this module, we will create data attributes in CDS for capturing data of the customer.

Step 1: Once logged into the correct environment, in Power Apps, navigate to Solutions and create a new solution.



1. Navigate to solutions
2. Select '+ New Solution'
3. Provide details and create the solution (Customer Service Request is the name for the solution that we are building as a part of this workshop)

Step 2: Once the solution is created, create a new entity in the solution and the metadata as listed below.



1. In the newly created solution, select new
2. From the drop down, select entity to create a new entity.

Entity: Customer Requests

Fields: Customer Name (Use Name Field)

Customer Email (SLT)

Customer Concern (MLT)

Agent Comments (MLT)

Request ID – AutoNumber (Select text prefix of the form REQ-0001)

Status Active:

Status Reasons:

Registered (Default)

In Progress

Status Inactive:

Status Reasons:

Resolved

Cancelled

If working with Share Point, below is the list of attributes that need to be created.

SP List: Customer Requests

Columns: Customer Name (use title column)

Customer Email (SLT)

Customer Concern (MLT)

Agent Assigned (Person)

Agent Comments (MLT)

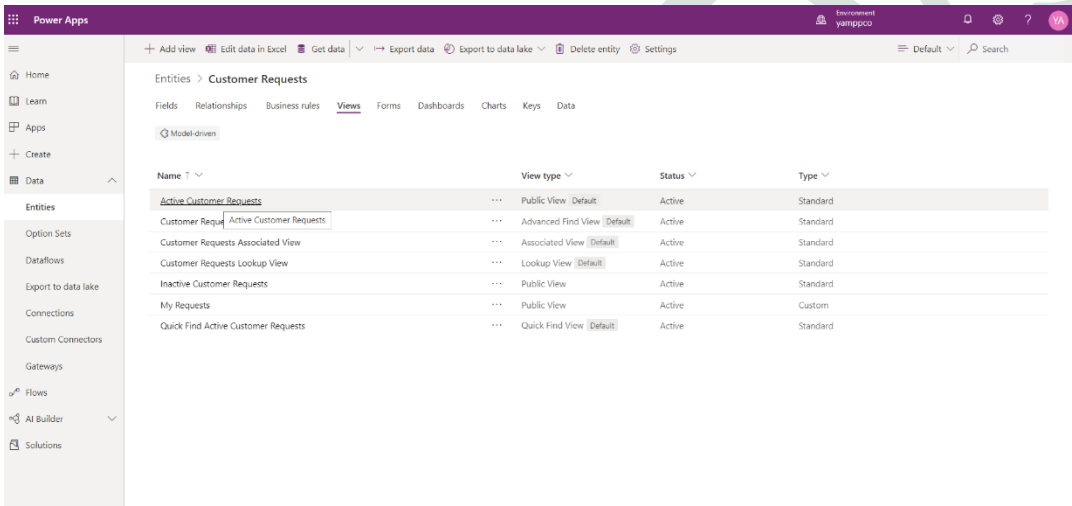
Status (Registered, In Progress, Complete, Cancelled)

## Module 2: Power Apps

In this section, we will look at creating a model driven app for the above entity and configure views, forms etc. on the entity and generate the model driven app. We will also create two canvas apps using the CDS data source and build a master app to view all requests and an individual app so that individual users can view the items they are working on.

Model Driven App: We will now create a model driven app based on the entity and configure the views and forms for the entity.

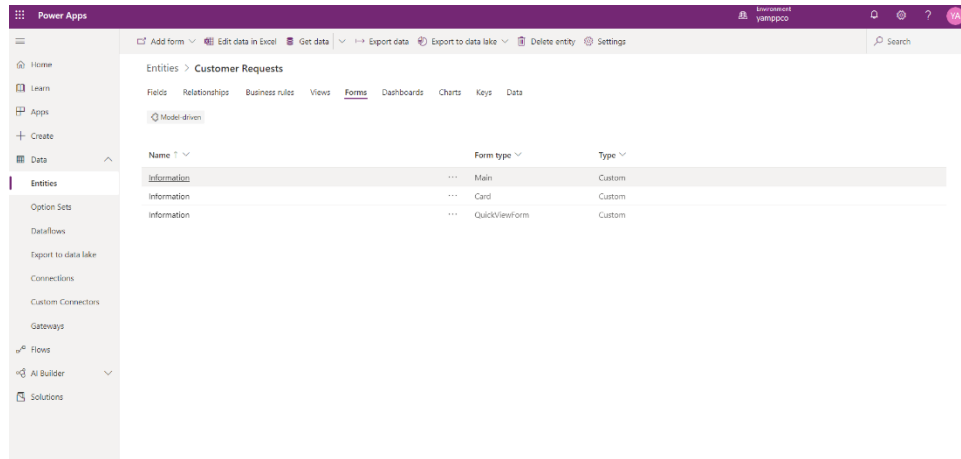
Step 1: Views: Open the Views Tab inside the entity. Select **Active <EntityName> View**.



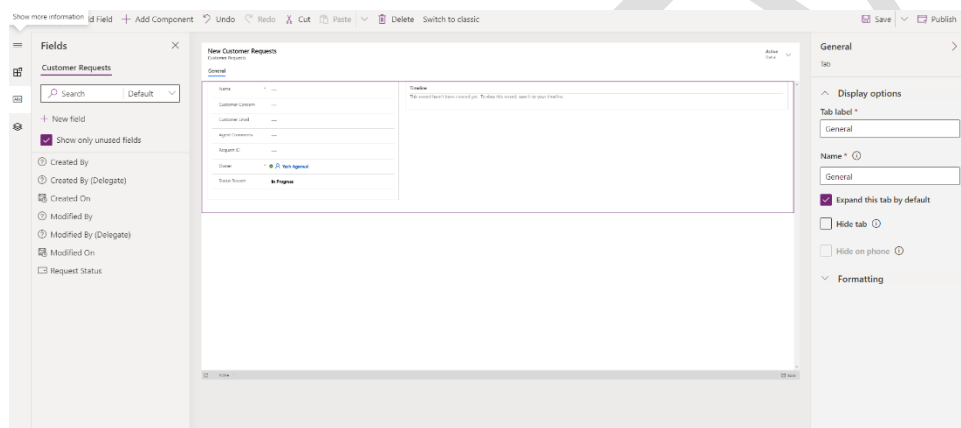
Select the Name, Customer Concern, Customer Email, Agent Comments and Request ID columns. Repeat the same steps for **Inactive <Entity Name> View**.

Name	Customer ...	Customer ...	Agent Co...	Request ID
Abe	Broken items in t...	abe@yamppco.o...		REQ-0001
Ben	Packaging was ta...	ben@yamppco.o...		REQ-0007
Han	I have not receiv...	han@yamppco.o...	The issue is been...	REQ-0002
Sam	Order misplaced	sam@yamppco....	Looking into the ...	REQ-0004
Test				REQ-0010
Test				REQ-0011
Tom	There seems to b...	tom@yamppco....	MyTest	REQ-0005
Yash	Package is tamp...	yash@testmppc...	Solved!	REQ-0009

Step 2: Form: Open the Forms tab inside Entity and select the information form of type main.



Drag and drop attributes on the form as per below image.



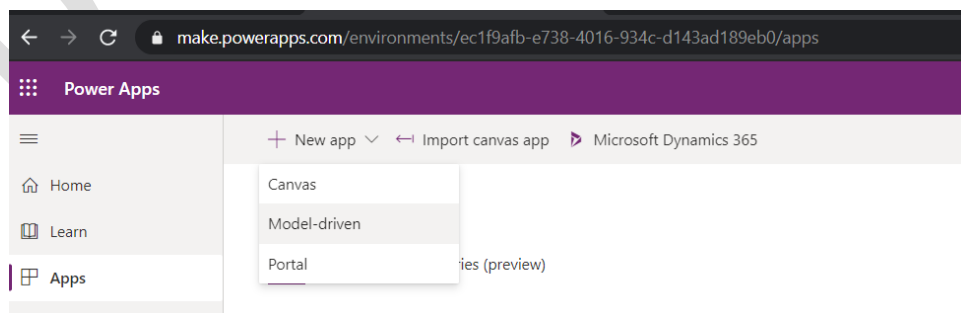
To add the timeline, you can follow the below steps:

1. Select Switch to Classic.
2. Select the General section and select Change Properties from the top menu. Switch to the Formatting tab and select it as 3 columns.
3. Select Insert tab and select Timeline.

### Step 3: Model Driven App

Navigate to <https://make.powerapps.com/> and make sure that you are in the correct environment.

1. Select "New App".
2. Select Model-Driven from the dropdown.



3. Give your app a name, **Customer Request Tracking <YourLastName>**.

**Create a New App**  
Create and publish your own app in minutes. You can start simple and add more components later.

Name: \* Customer Request Tracking

Unique Name: \* cr157\_CustomerRequestTracking

Description:

Icon: ☒ Use Default Image

Unified Interface URL: [https://yampco.crm.dynamics.com/Apps/uniquename/cr157\\_CustomerRequestTracking](https://yampco.crm.dynamics.com/Apps/uniquename/cr157_CustomerRequestTracking)

☐ Use existing solution to create the App

☐ Choose a welcome page for the app

☐ Enable Mobile Offline

App Title: Customer Request Tracking

#### Step 4: Sitemap

When the model driven app is configured, select Edit icon next to Site Map.

1. Select "New Subarea".
2. On the right menu, select the type as Entity and Entity as Customer Requests (created as part of Step 2 in Module 1).

App Designer > Sitemap Designer  
Customer Request Tracking

Group 1

Requests

Customer Requests

Components

SUB AREA

General

Type: Select a type

Entity: Customer Requests

URL:

Default Dashboard: Select a dashboard

Title (1033):

Icon: Use Default Image

3. Select Save.
4. Select Publish.
5. Select App Designer on top-left side of the screen.
6. On the Model Driven Editor page, select Validate.
7. Select Publish.
8. Select Play.

Screen: Once the app is configured and played, it looks like the below screenshot.

**Dynamics 365** Customer Request Tracking Group 1 > Customer Requests

Active Customer Requests

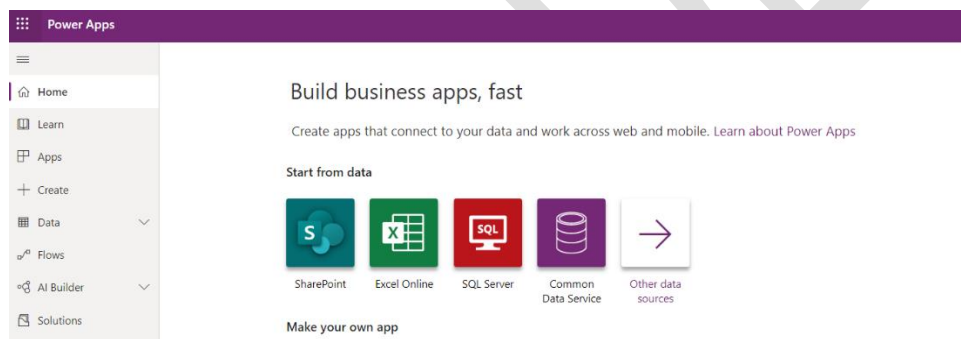
Name	Customer Con...	Customer Email	Agent Comm...	Request ID	Owner	Status	Status Reason	Created On
Abe	Broken Item...	abe@yampp...	---	REQ-0001	Yash Agarwal	Active	In Progress	2/20/2020 1:07 ...
Ben	Packaging w...	ben@yampp...	---	REQ-0007	Yash Agarwal	Active	In Progress	2/20/2020 2:01 ...
Test	---	---	---	REQ-0010	Yash Agarwal	Active	In Progress	3/19/2020 12:31...
Test	---	---	---	REQ-0011	Yash Agarwal	Active	In Progress	3/19/2020 1:57 ...
Yash	Package is L...	yash@itestm...	Solved!	REQ-0009	Yash Agarwal	Active	In Progress	2/29/2020 1:22 ...

1 - 5 of 5 (0 selected)

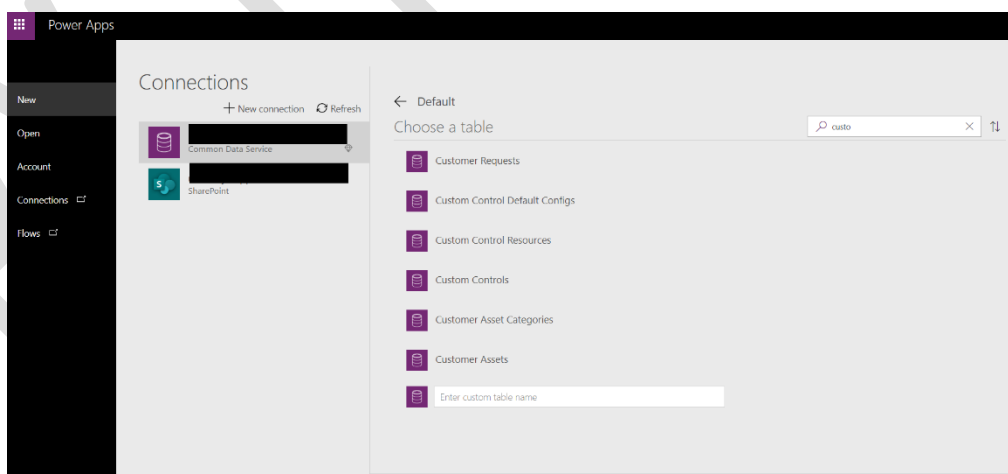
Canvas App 1 (Master App): In this app, we will use the template app and directly create it from data by pointing the app to the CDS entity. This is an autogenerated app from Power Apps that has a standard layout and does not require much customization.

Step 1: Navigate to <https://make.powerapps.com/> and make sure that you are in the correct environment.

Step 2: Select Common Data Service under Start with Data.

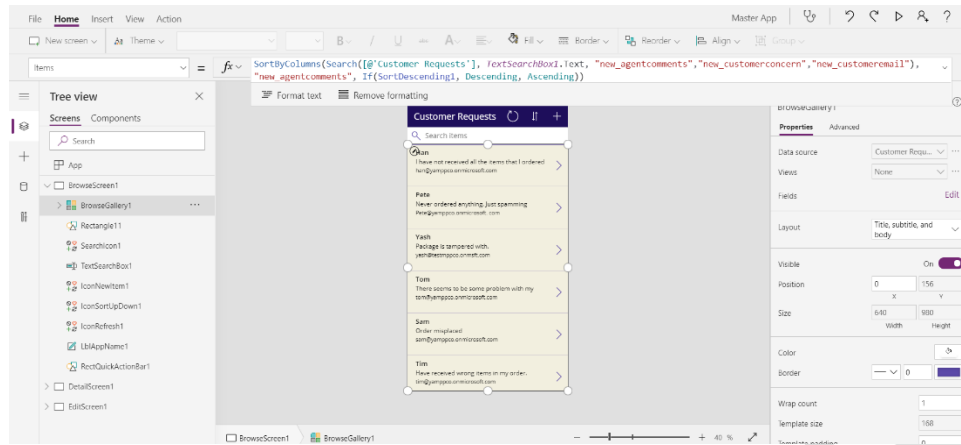


Step 3: Select the Customer Requests Entity and select **Connect**.

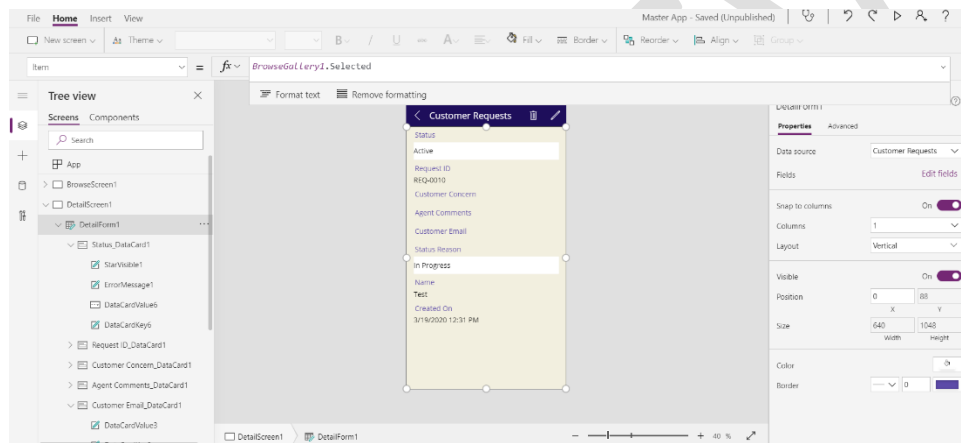


Once the app is created, you should notice three different screens created.

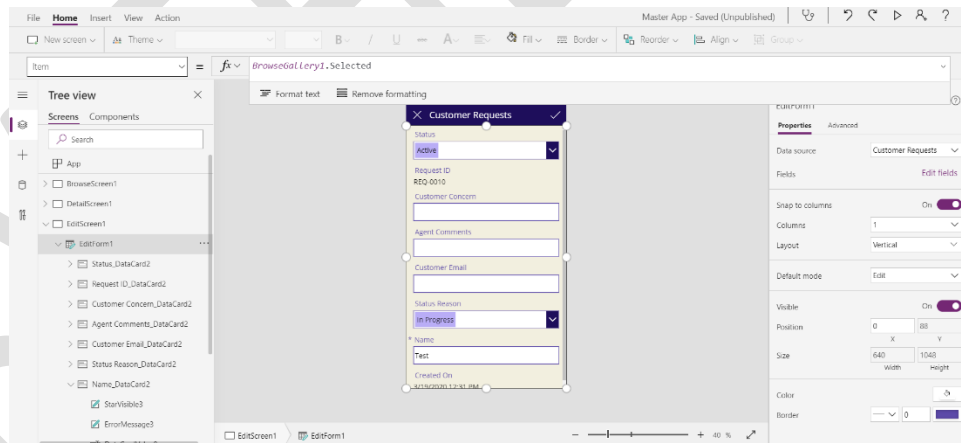
Screen 1: Browse Screen -> This screen enables the users to see all the customer requests.



Screen 2: Details Screen -> This screen allows the users to see details specific to the selected customer request.



Screen 3: Edit Screen -> This screen allows the users to edit the selected customer request.



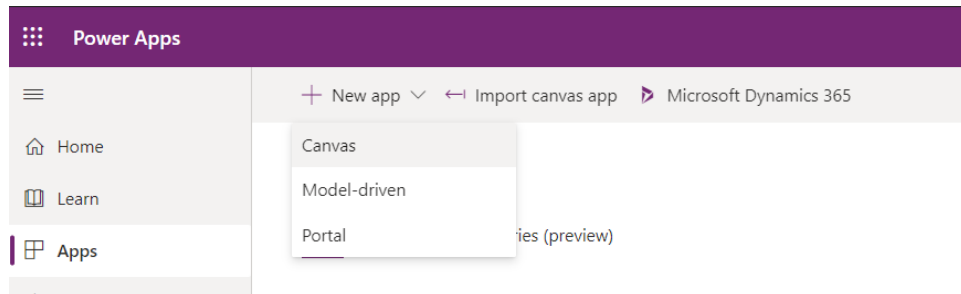
Step 4: Select **Save** and add an app name. The app is auto published on first save.

Canvas App 2 (Individual App): In this app, we will create an app from scratch and learn about individual controls. We will display records assigned to the user who is using the app (based on the logged in user) and allow them to edit those records. We will also add some basic branding to the app and modify the controls.

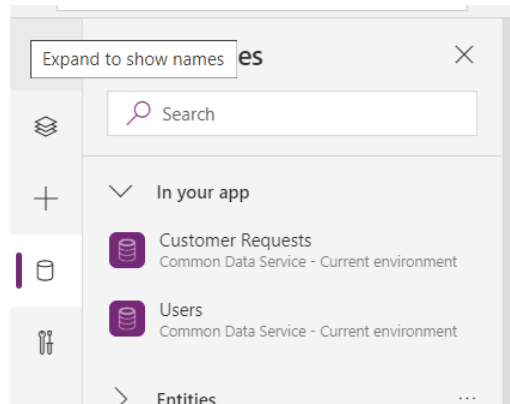
Step 1: Navigate to <https://make.powerapps.com/> and make sure that you are in the correct environment.

Step 2: Select Apps from the Left Menu.

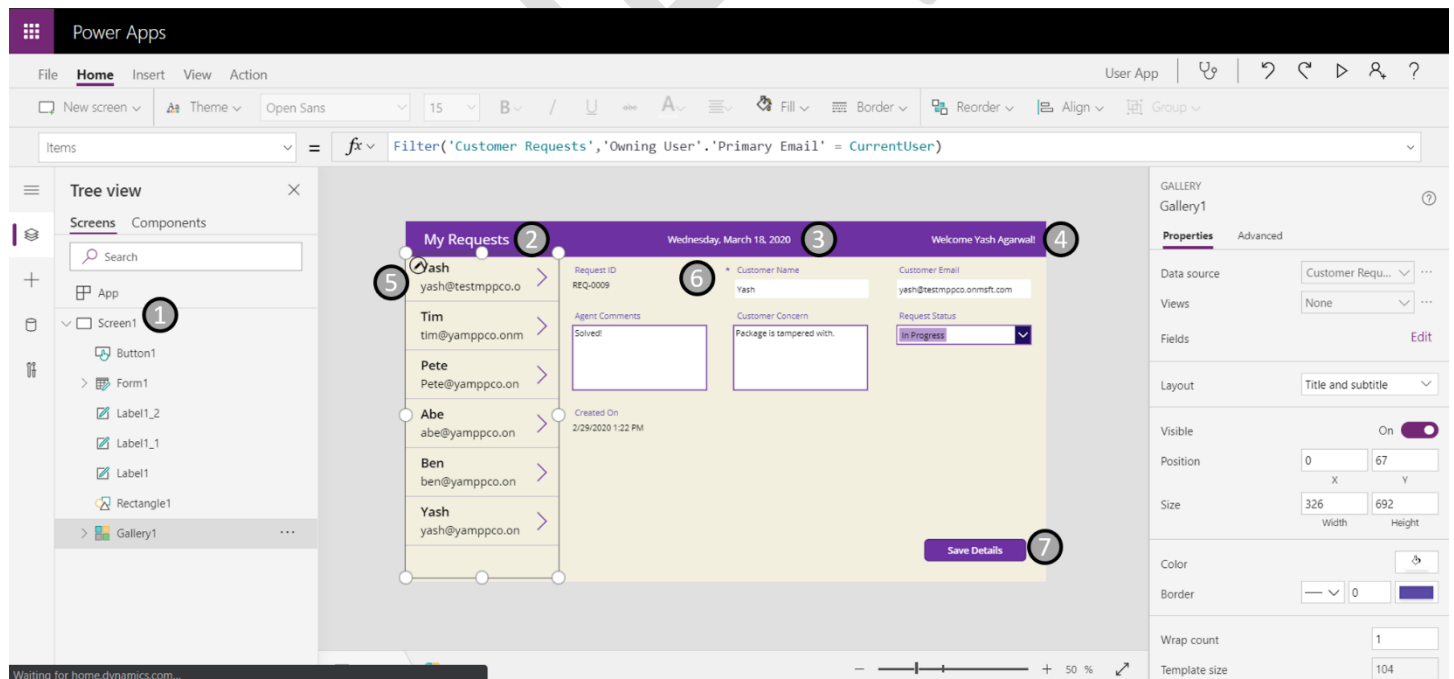
Step 3: Select “New App” and then select Canvas from the dropdown.



Step 4: Select Database icon on the left menu. Search and Select Customer Requests and the User entity.



Screen 1:



1. Set the OnStart property of the app to store user details as a variable.



Expression

Used:

`Set(CurrentUser,User().Email)`

2. This is label to show the App Name. Add a label control from the top menu. Set the “Text” property of the label to: "My Requests".
3. This is a label control to show today’s date. Expression used on the “Text” property of the label is: Text(Today(),DateTimeFormat.LongDate)
4. This is a label control to show a welcome message to current logged in user. The “Text” property of the control is set to: "Welcome "&CurrentUser.FullName&"!"
5. Add a Gallery control which will display the details from Customer Requests entity.

Items = fx Filter('Customer Requests','Owning User'.Primary Email' = CurrentUser)

Items = fx Filter('Customer Requests','Owning User'.Primary Email' = CurrentUser)

Expression used on the “Items” property is: Filter('Customer Requests','Owning User'.Primary Email' = CurrentUser)

6. Add an Edit Form to show select item details. Select the DataSource as the Customer Requests entity. Set the Item property of the Edit Form to: LookUp('Customer Requests','Customer Requests' = Gallery1.Selected.'Customer Requests')

Item = fx LookUp('Customer Requests','Customer Requests' = Gallery1.Selected.'Customer Requests')

Select the Request ID datacard, now select **Unlock the properties** from right menu. Set the Display Mode of the datacard to **View** from the right menu. This makes these fields un-editable.

7. This is a button control which saves the details back to CDS. Set the Text property of the button control to "Save Details". Set the OnSelect property of the button control to SubmitForm(Form1);Refresh('Customer Requests')

OnSelect = fx SubmitForm(Form1);Refresh('Customer Requests')

Step 5: Select File from top menu and then select Save.

Step6: Provide a name to the App, example -> My Requests. Click Save. The app is auto published when saved for the first time.

## Module 3: Power Automate

In this section, we will look at creating Flows in Power Automate that can be triggered from Power Virtual Agents as actions to store the data from the customer responses. We will create three flows:

NOTE: While creating flows for Power Virtual Agents, it is necessary that the flows are created in a solution that is in the same environment as the Power Virtual Agent. If this configuration is not met, the virtual agent will not be able to identify the flow to invoke as an action.

Therefore, as an initial step, while creating a new flow for the virtual agent, navigate to the Solutions tab and then click on the “Customer Service Request” solution created in the first step and then click new and click on Flow.


1. Create a new request: In this flow we will get inputs from the PVA to create a new record in CDS and provide the customer with a request ID.
  - a. Step 1: Trigger: Select the trigger as When a Power Virtual Agent calls a Flow. On the trigger, add 3 text input properties one each to record the customer name, email and the concern.
  - b. Step 2: Action: Now that we have the inputs configured, add an action to create a new record in CDS. Select the appropriate environment and then map the inputs to the respective fields from the dynamic selector.


- c. Step 3: Action: Return a response to Power Virtual Agents, we now have to return a response to the virtual agents. So, configure an output property of type text to send the Request ID from the CDS record.

The first flow is now ready. Name it as “Create a new request” and click save. Navigate back to the solution and check if the flow appears in the solution.

Once the above steps are completed, the flow should look something like the screenshot below:

BY THE DEV'S


 Power Virtual Agents ...



Name

Please enter your input


...



Email

Please enter your input

...




Concern

Please enter your input

...

[+ Add an input](#)



 Create a new record ...

\* Environment


▼

\* Entity Name

Customer Requests


▼

\* Name


 Name x

Agent Comments

Customer Concern

 Concern x

Customer Email

 Email x

Import Sequence Number

Sequence number of the import that created this record.

Record Created On

Date and time that the record was migrated.

Request ID

Request Status Value

▼

Status Reason Value

Reason for the status of the Customer Requests

▼

Time Zone Rule Version Number

For internal use only.

UTC Conversion Time Zone Code

Time zone code that was in use when the record was created.

Owner

Owner Id

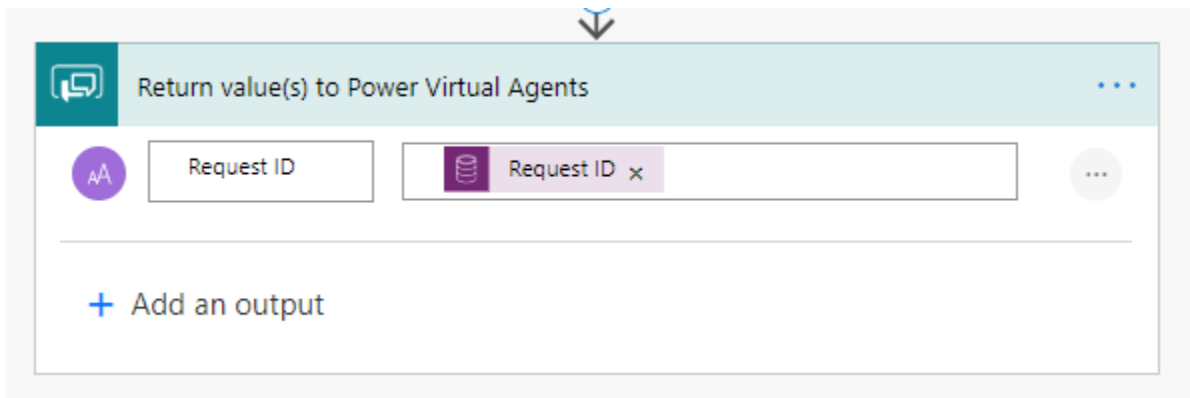
Owner Type

Owner Id

▼

Hide advanced options

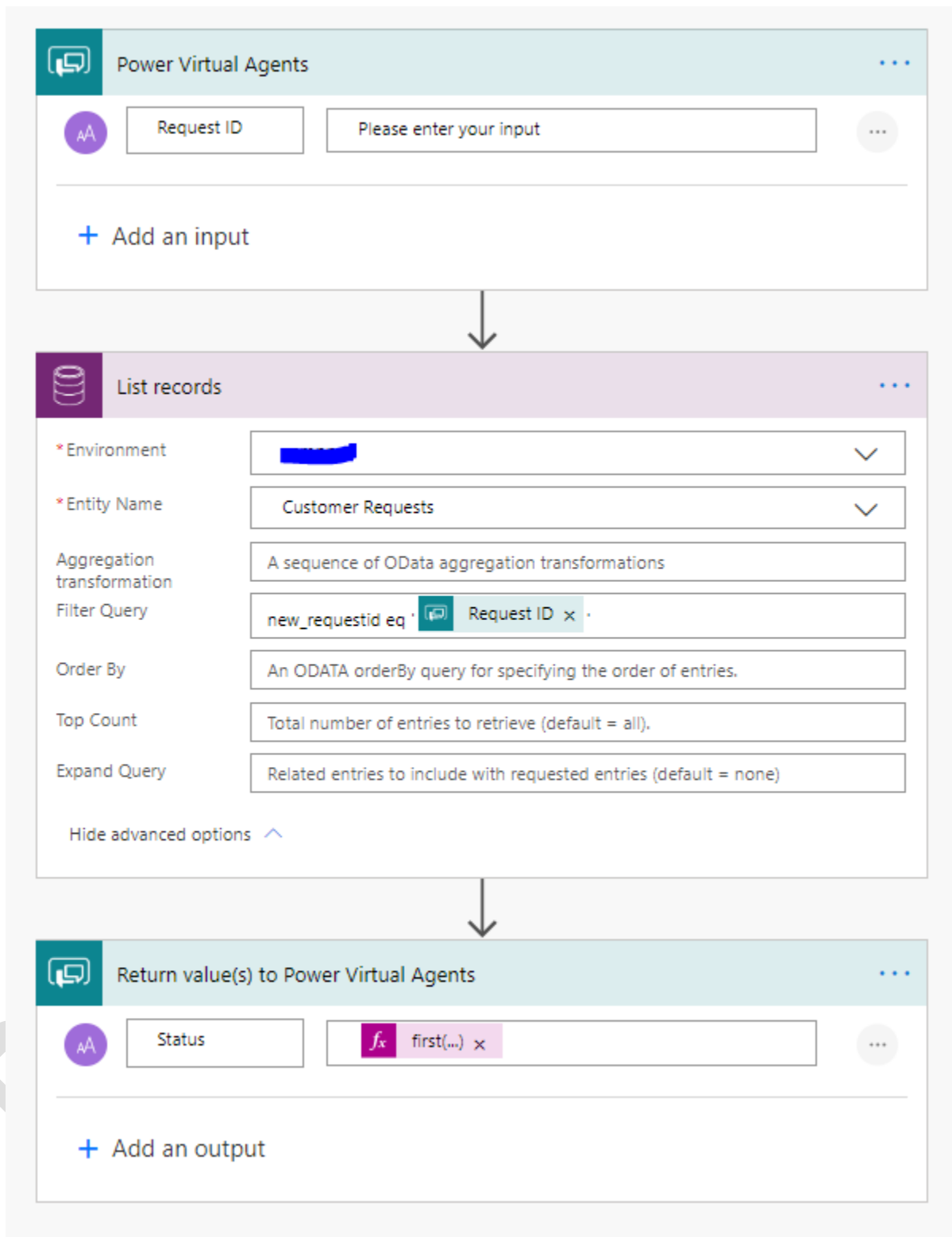
^



2. Track a previous request: In this flow, we will get the request ID from the PVA and query the CDS to get the details on the customer request.
  - a. Step 1: Trigger: Select the trigger as When a Power Virtual Agent calls a Flow. On the trigger, add a text input property to record the request ID from the customer.
  - b. Step 2: Action: Now that we have the customer request ID, add an action to list records from CDS. Select the appropriate environment and use a filter query to get the record that matches the request ID. Expression: `new_requestid eq '{@triggerBody()['text']}` (Here, new\_requestid is the logical name of the Request Id field from CDS. You can get that by navigating to the entity and on the field information in CDS. After eq ' you can select the input from the dynamic selector to point to the input value configured in the trigger).
  - c. Step 3: Action: Return a response to Power Virtual Agents, we now have to return a response to the virtual agents. So, configure an output property of type text to send the agent comments to the customer. As the list records action returns a list of records in the form of an array, we need to use the first function to get the first item from the array. Expression: `first(outputs('List_records')['body/value'])['new_agentcomments']` (Here, we are extracting the agent comments from the first value returned in the list from the list records action. New\_agentcomments is the logical name of the Agent Comments field that can be retrieved again by navigating to the entity in the CDS and check for the field information)

The second flow is now ready. Name it as "Get request" and click save. Navigate back to the solution and check if the flow appears in the solution.

Once the above steps are completed, the flow should look something like the screenshot below:

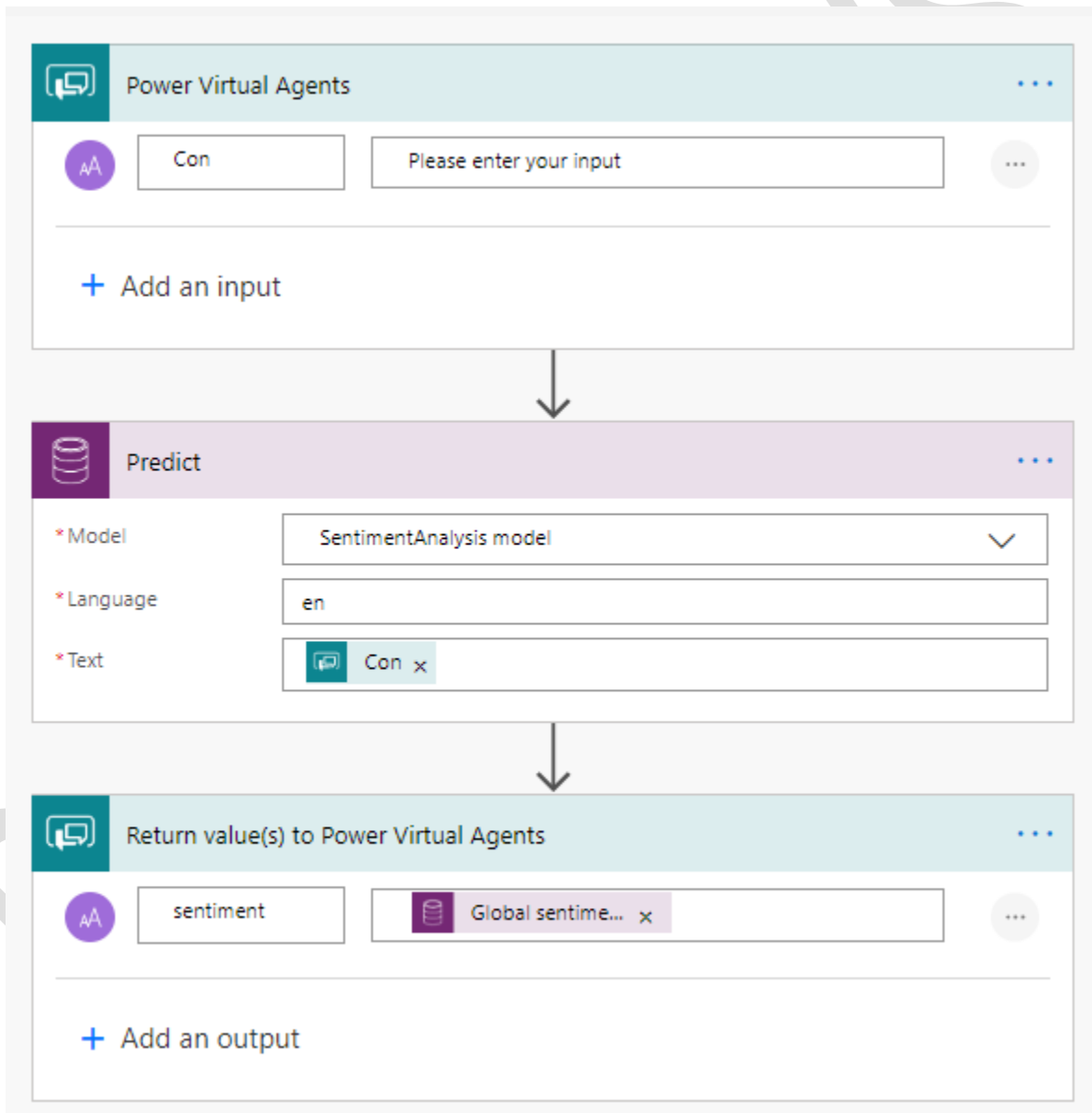


3. Sentiment analysis: In this flow, we will get an input from the PVA and perform a sentiment analysis on the concern and pass the results to the PVA.
  - a. Step 1: Trigger to trigger the flow when it is called from a power virtual agent. Configure an input of type text to store the concern of the customer.

- b. Step 2: Action: Search for the “Predict” action and select it. Select the “Sentiment Analysis” model and then provide language option as “en” for English. Now pass the concern from the above step to the text property.
- c. Step 3: Action: Pass a response to the virtual agent and configure the output property as text. Select the Global Sentiment value from the dynamic selector to pass the sentiment to the virtual agent. The values passed on this variable are either of positive, neutral and negative.

The third flow is now ready. Name it as “Sentiment Analysis” and click save. Navigate back to the solution and check if the flow appears in the solution.

Once the above steps are completed, the flow should look something like the screenshot below:

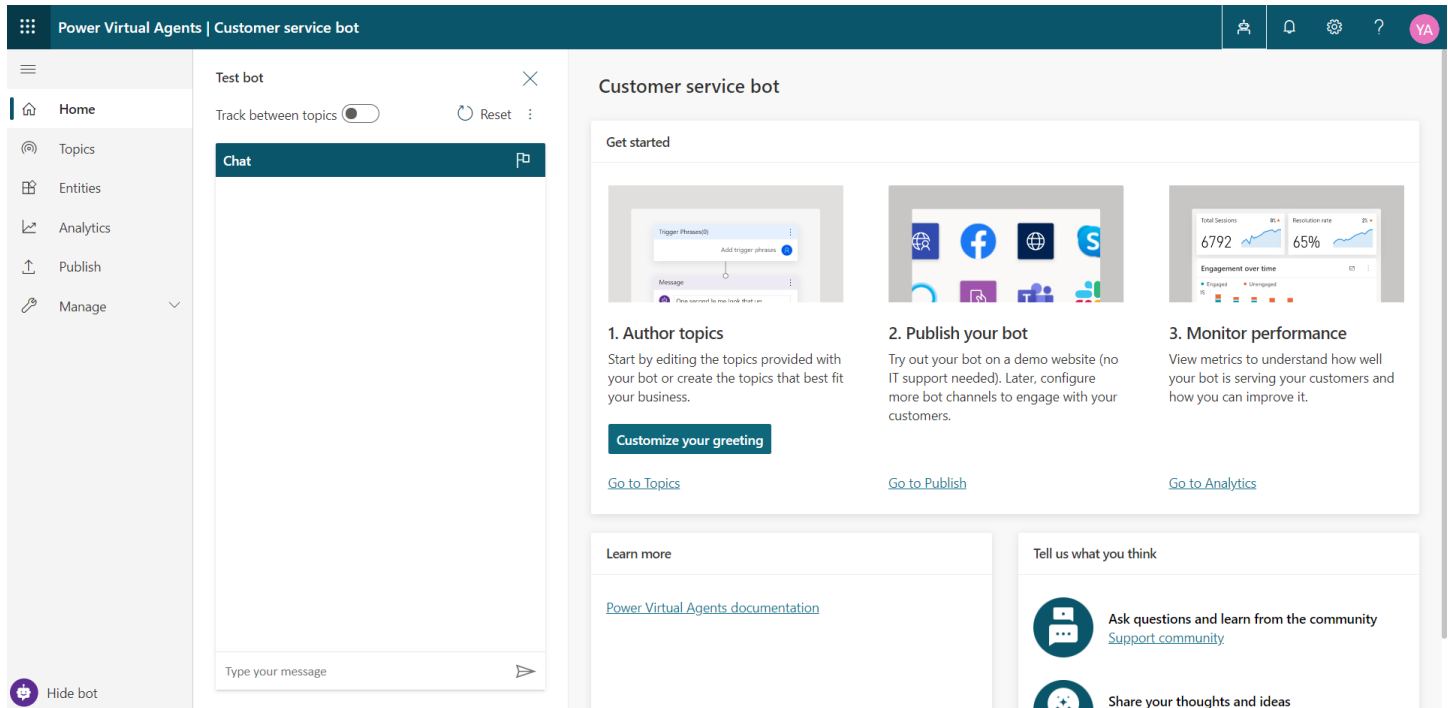


## Module 4: Power Virtual Agents

In this section, we will look at provisioning a Power Virtual Agent that the customer can interact with and provide information. Flows created in the above section will be invoked as action to store data and perform sentiment analysis on the user input.

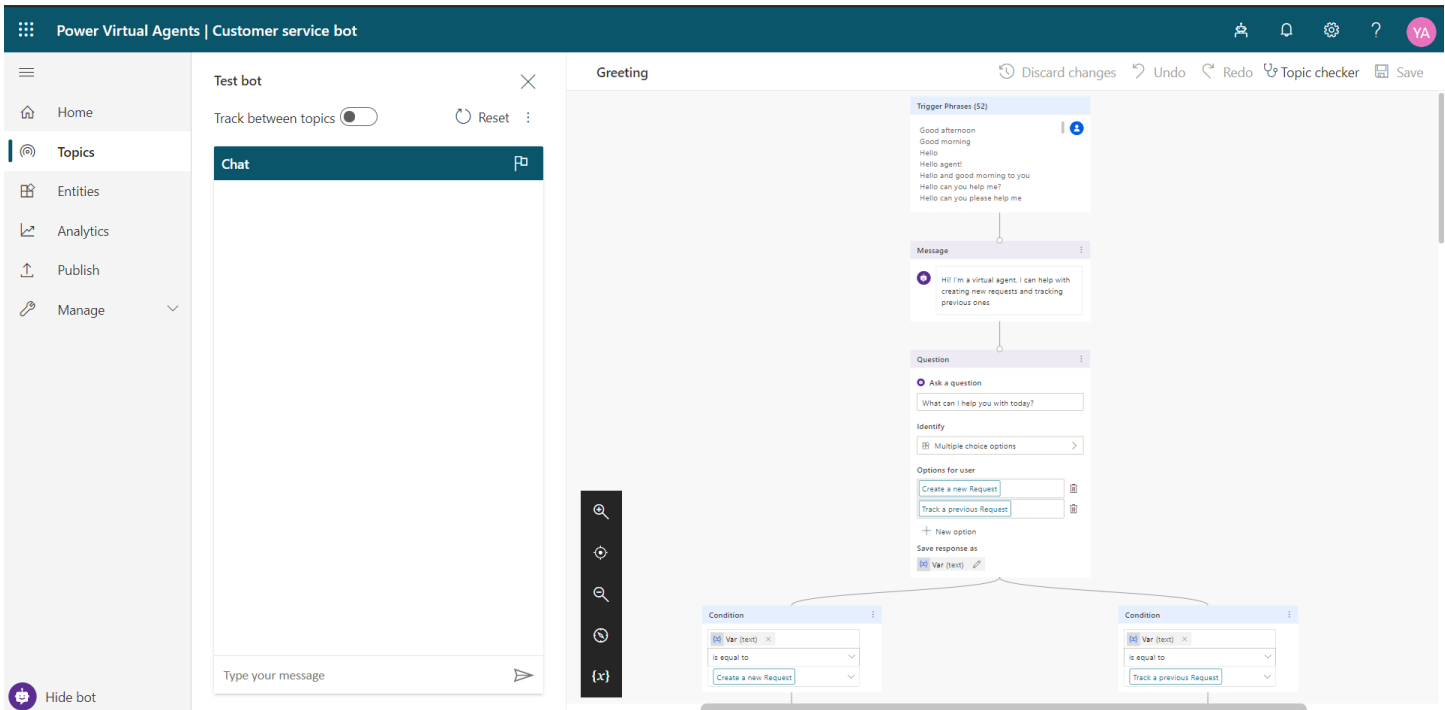
Step 1: logon to <https://powervirtualagents.com> and create a new bot by clicking the + sign. Ensure that you select the correct environment (the one in which you have created the flows/ solution in the previous modules).

Step 2: It may take up to 30 minutes for the bot to provision at the most and then once the bot is provisioned, you should be able to see the screen as shown in the screenshot below.



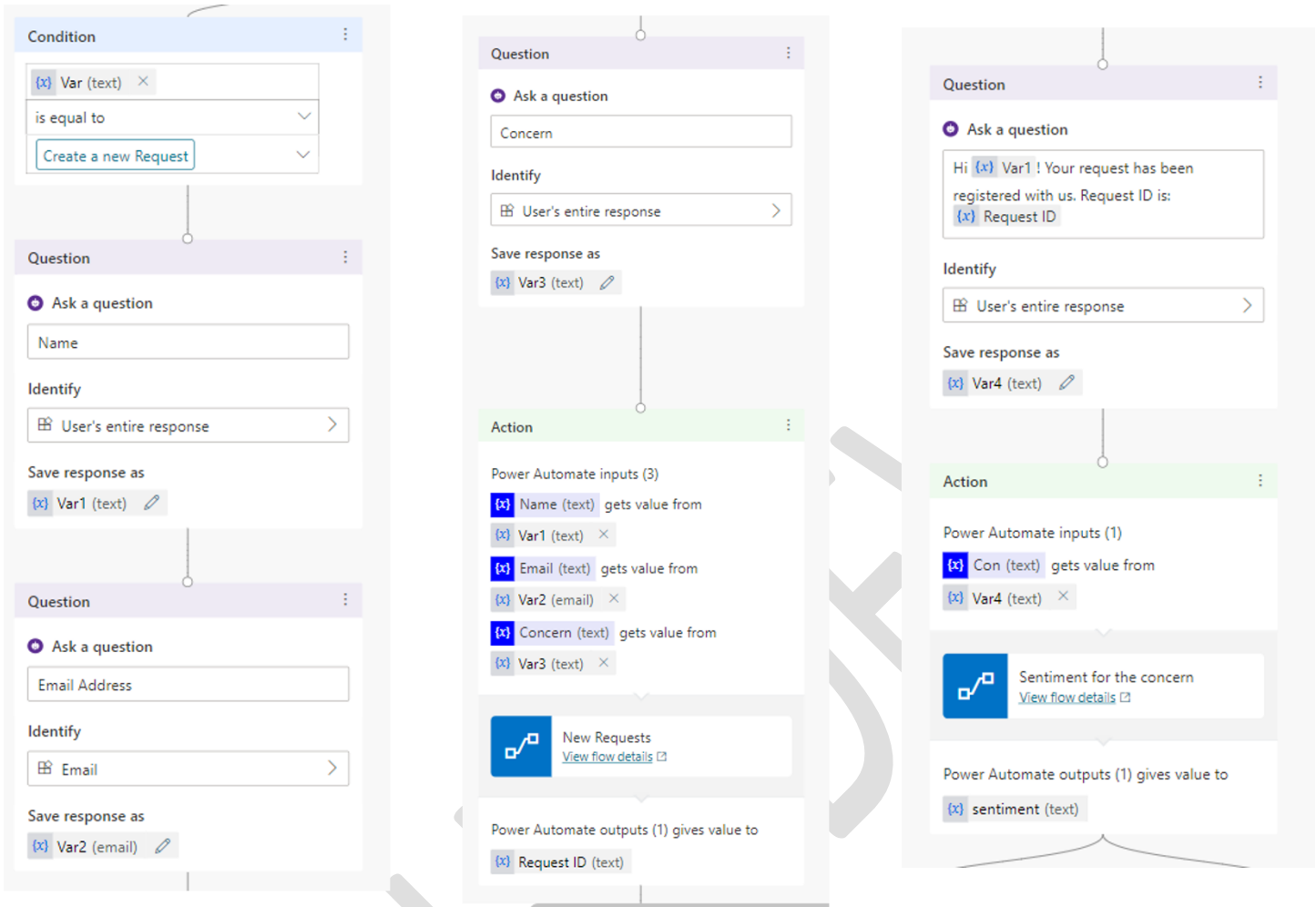
Step 3: Click on the customize greeting button and navigate to the canvas to start building the bot. Hide the test bot by clicking the 'X' or the Hide Button at the bottom left.

Step 4: Now customize the welcome message and start adding steps/ nodes to the virtual agent. As this is a customer request tracking bot, we will build two paths. One for creating a new ticket and the other for tracking a previous ticket. After customizing the welcome message, add a node as ask a question and identify the response as a multiple choice. Now add the first choice as Create a new request and the second choice as Track a previous request. Once the two options are added, the flow creates two paths and branches out as shown in the image below:

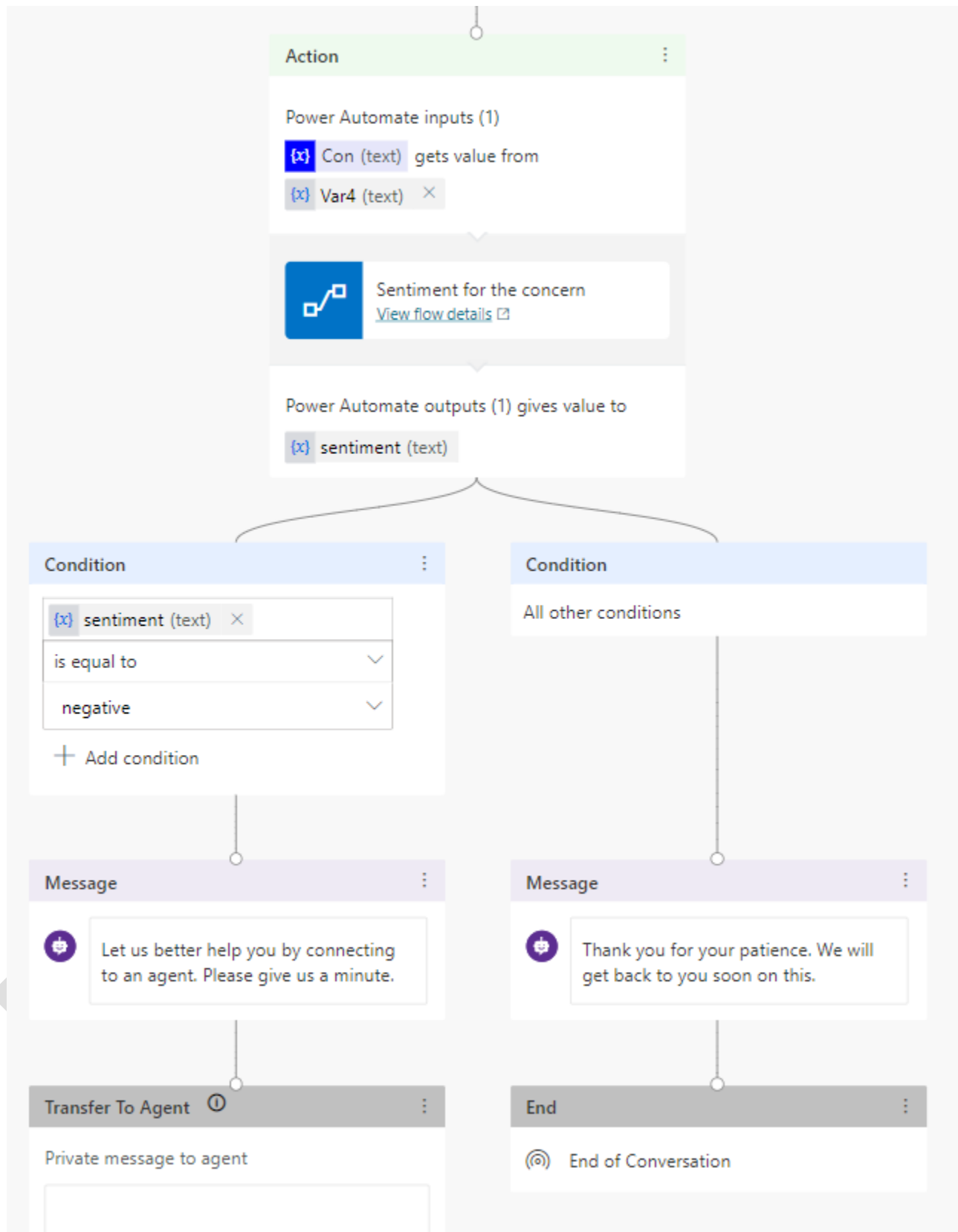


Step 5: Now start building the individual path items for each option. First let's build for the Create new request:

1. Question: To get the name of the customer. (Identify the entire response of the user as a string)
2. Question: To get the email of the customer. (identify the response as an email. This validates the user response and performs a check if the input is a valid email or not)
3. Question: To get the concern from the customer. (Identify the entire response of the user as a string)
4. Action: Invoke a flow from Power Automate. Select the create a new request flow and provide all the input parameters as shown in the image below. The flow returns with a request ID and the output can be used to display the message to the customer as shown in the image below.



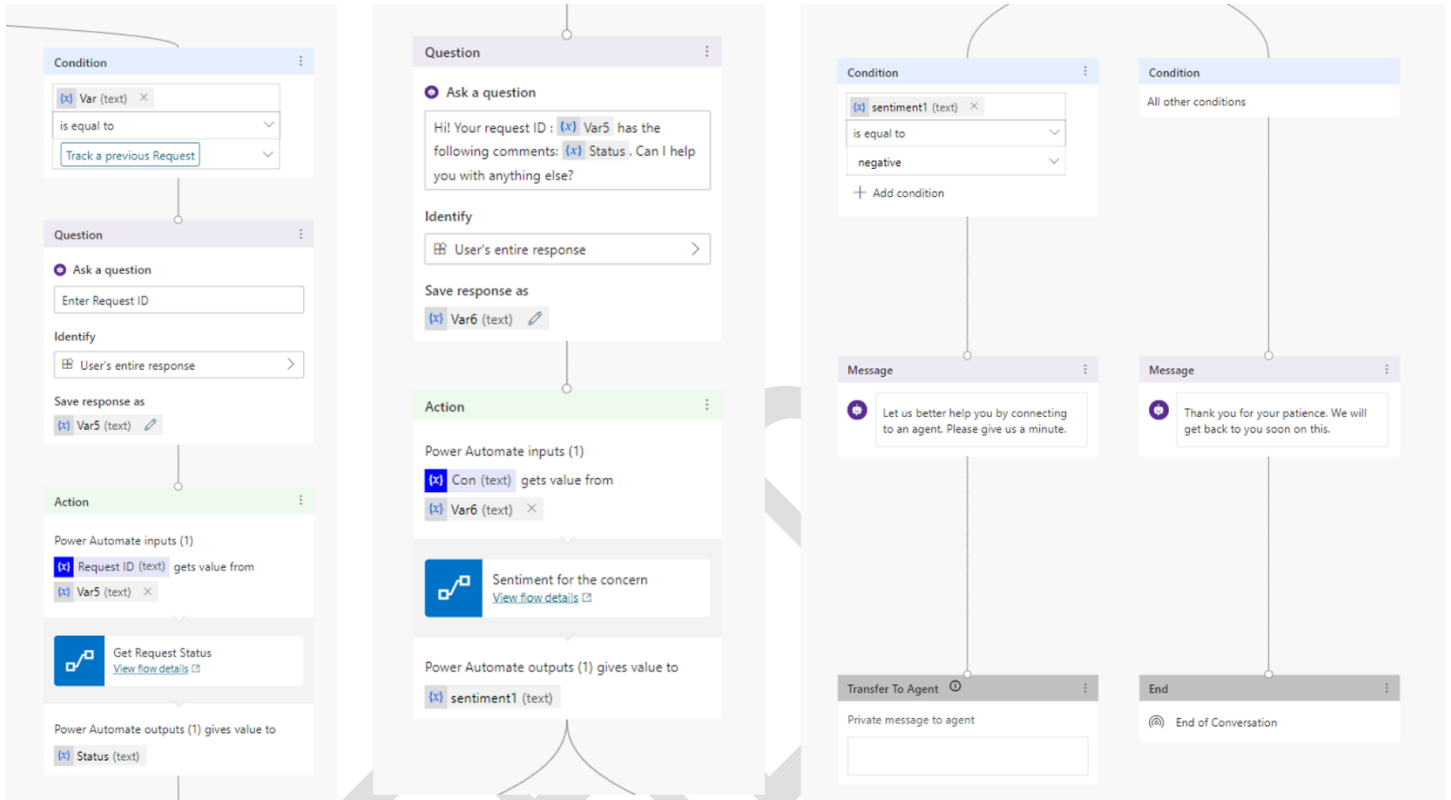
5. Question: Once the request id has been provided, check if the customer has any further queries. Identify the entire response as a string.
6. Action: Invoke a flow from Power Automate: Select the Sentiment analysis flow this time and pass the user input to the input parameters. The flow then performs the sentiment analysis on the concern and returns with a sentiment score that can be further used to route the conversation in the bot.
7. Identify the output from the flow action and select the condition action. Check for the negative sentiment and show the appropriate message to the customer and end the conversation by selecting transfer to agent.
8. Similarly, for all other values in the condition, show the message and then end the conversation with a survey.



Step 6: Now start building the individual path items for the track a request option:

1. Question: To get the request ID from the customer. Identify the entire response as a string as it will then be passed to the flow.

2. Action: Invoke a flow from Power Automate. Select the get request status flow and pass the request id in the input parameters as shown in the image below.
3. From the outputs of the flow, show the message to the customer with the appropriate message and ask a question. Identify the entire response of the customer as a string.
4. Pass the response to the sentiment analysis flow as shown in the image below and get the sentiment value.
5. Based on the condition for the sentiment value, route the chat as done in the Step 5.



## Module 5: Power BI

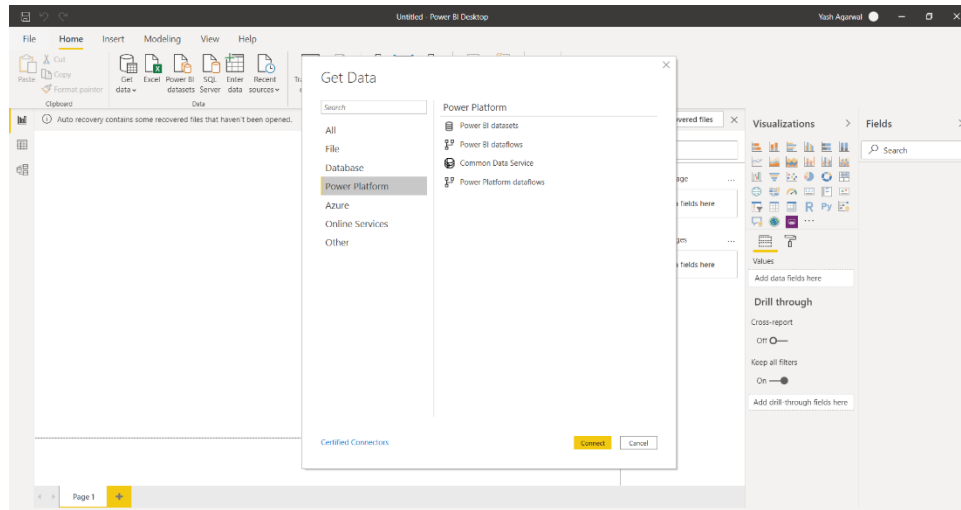
In this section, we will look at creating a Power BI report and publish it to the workspace. We will create a report from the data in CDS and visualize the customer requests, turnaround time on the requests etc.

Step 1: Install PowerBI Desktop if not already. <https://powerbi.microsoft.com/en-us/downloads/>

Step 2: Open the PowerBI Desktop application and Sign in to your account.

Step 3: Select Get Data from the top menu and select more.

Step 4: Select Power Platform in the opened window and select Common Data Service. Select Connect.

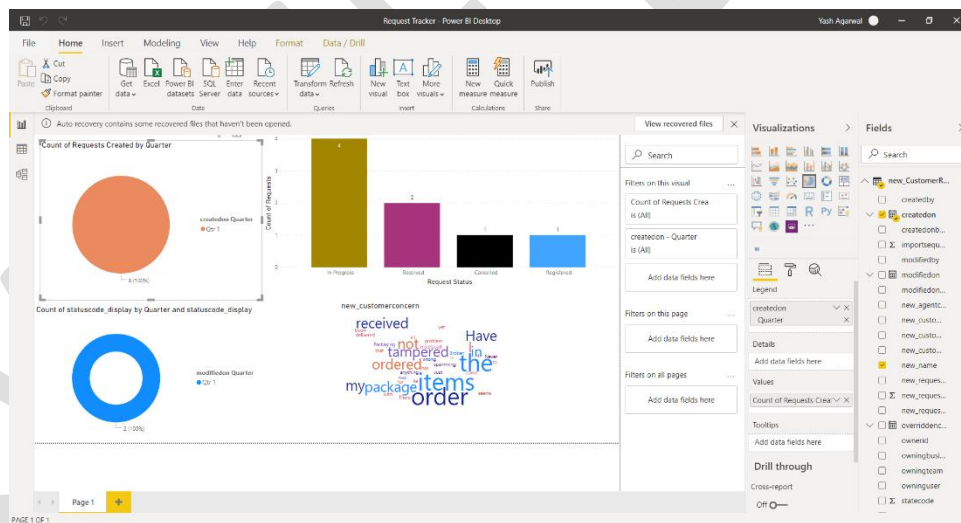


Step 5: To get the Server URL. Navigate to <https://make.powerapps.com/> and select the setting icon on the right menu. Select Advanced setting. Copy the URL opened in the new window and paste it in a Notepad. Keep the URL as: <https://organizationIdentifier.crm.dynamics.com/>. Paste this in the Server URL. Select OK.

Step 6: Sign in to your organizational account. Select Connect.

Step 7: Expand the entities and select the Customer Requests entity. Select Load.

Step 8: To create visuals, select the visual from the Visualizations. Drag and drop the attributes in Legend and Values for the selected visual.



Note: To create a word cloud, you can import a visual from the marketplace.

Chart 1: This chart shows the number of requests created per quarter.

Chart 2: This chart shows the number of resolved requests per quarter.

Chart 3: This bar graph shows the count of Customer Requests by their status.

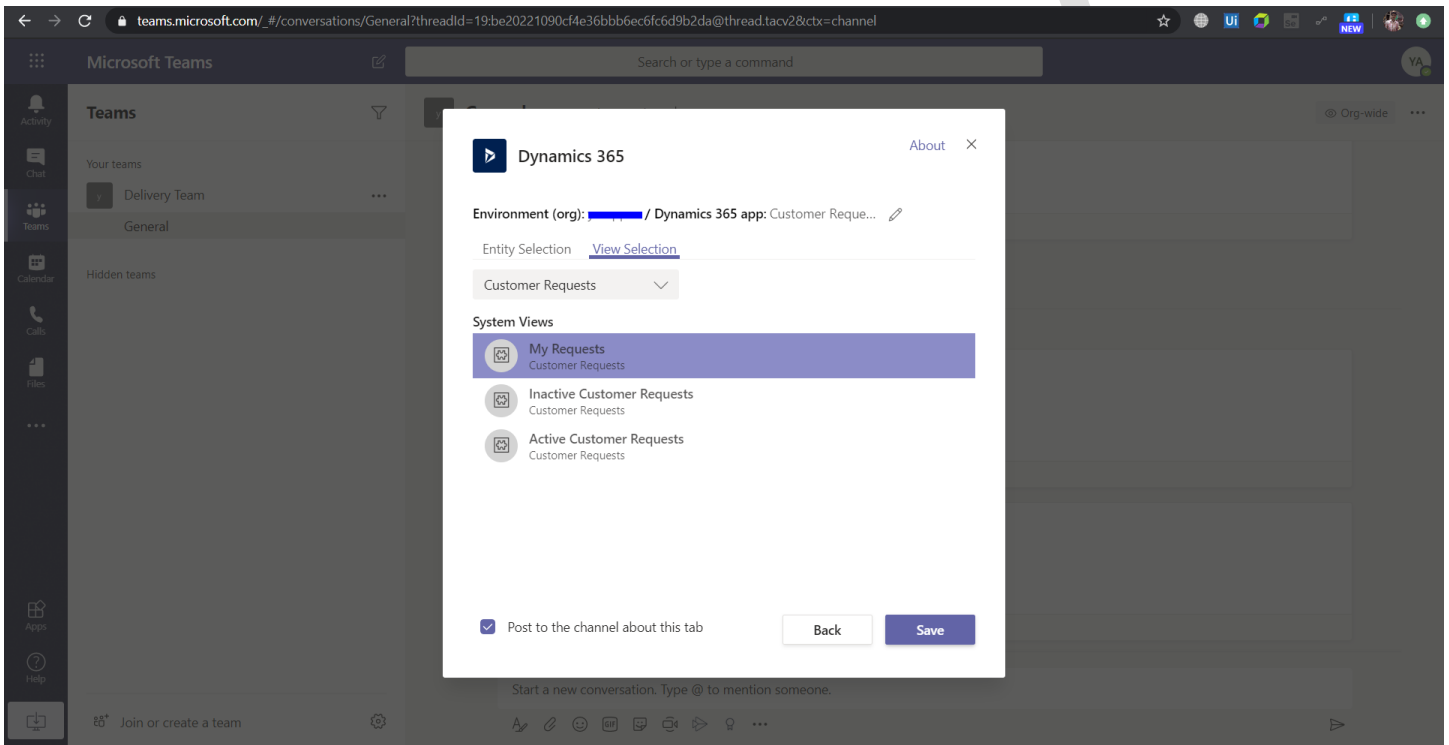
Chart 4: This is a word cloud to show the most used keyword in customer comments.

Step 9: Once the report is complete, Select **Publish** from the top menu.

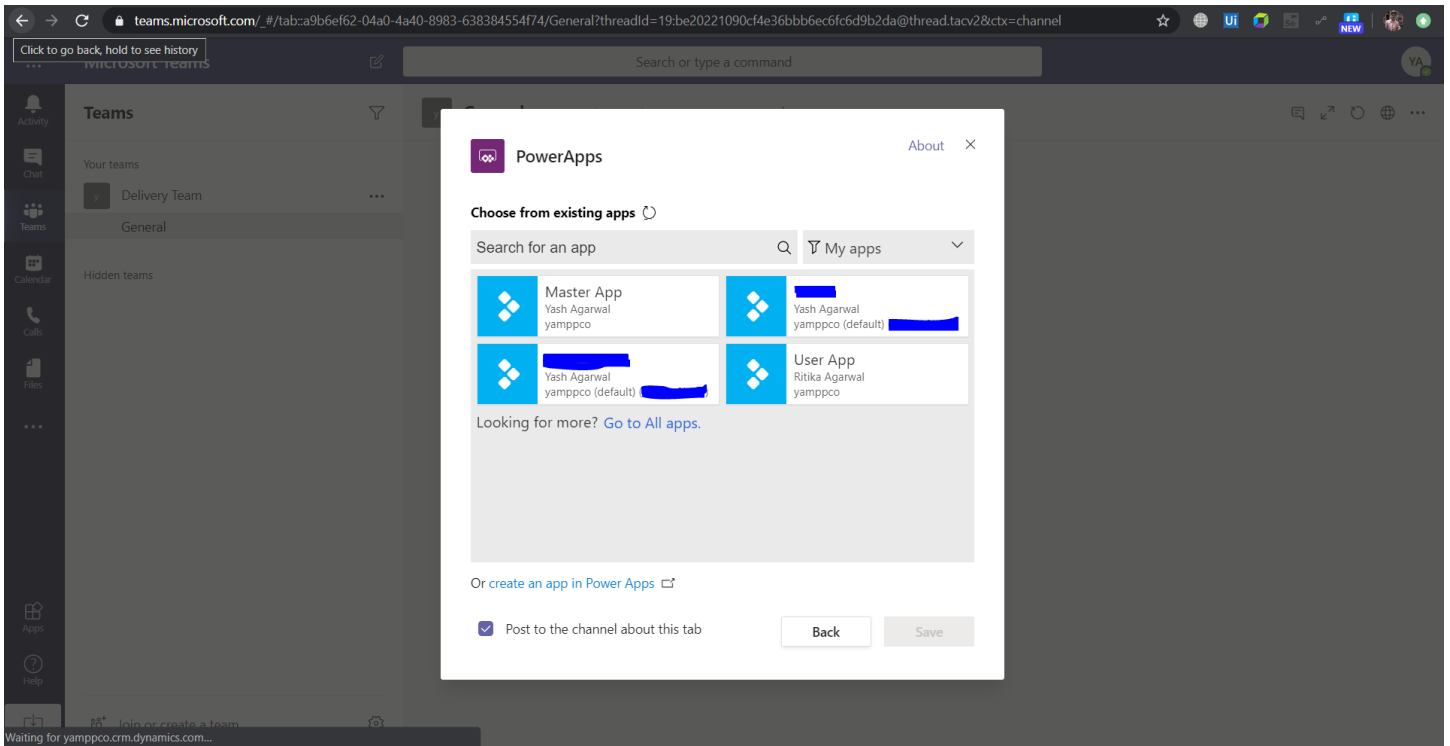
## Module 6: Teams Integration

In this section, we will look at the team integration aspect of the solution. Once the data from customer is stored in CDS, we want the internal customer service agents to work on those tickets and update them so that the updated status can be provided to the customer through the virtual agents. We will integrate the Model Driven App view so that a user can view the requests that they are working on, the master and individual canvas apps and the Power BI report.

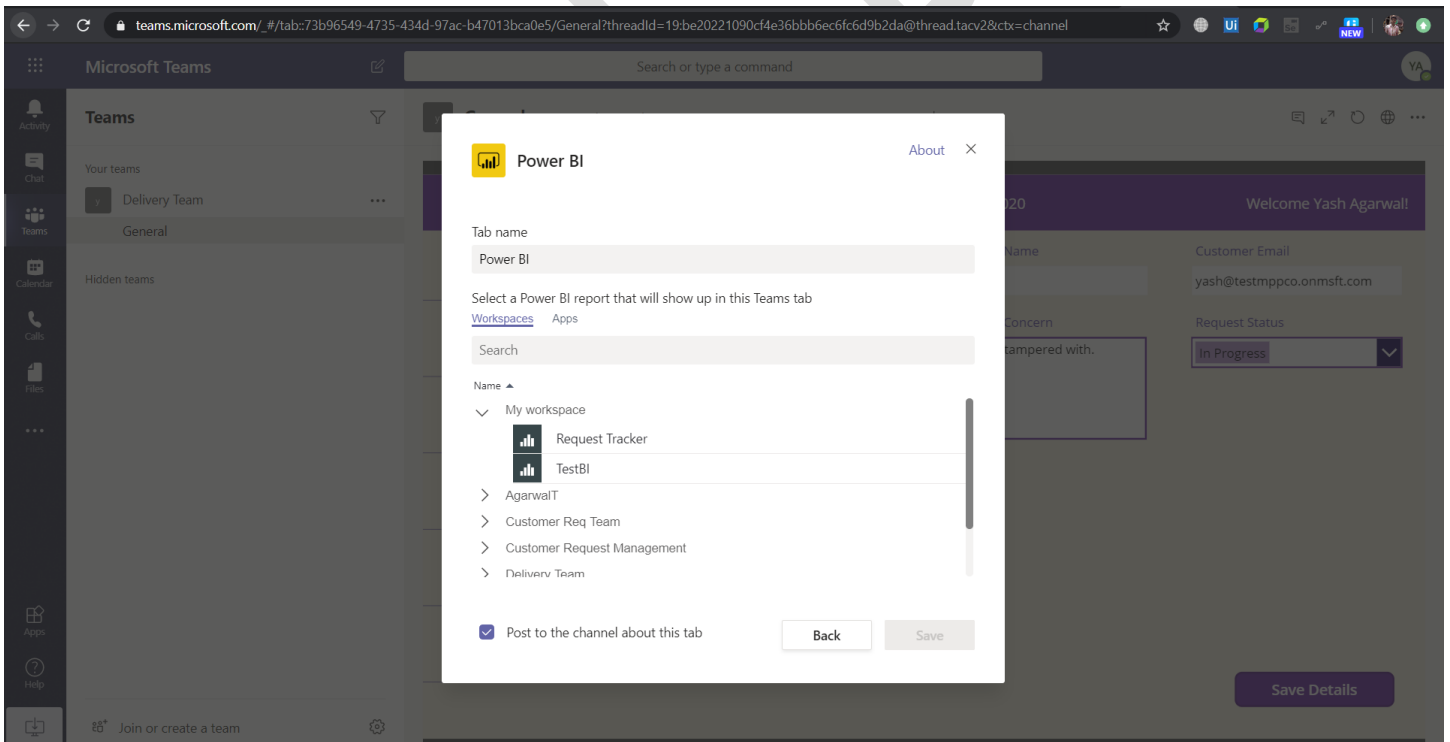
**Step 1: Add the Model driven app to the tab:** Navigate to the teams channel and select the '+' to add a new tab. Search for Dynamics 365 tab and then select the appropriate environment and the model driven app. Switch to the view selection and select the My Requests view and click on save. This will add the My Requests view as a tab to the teams channel.



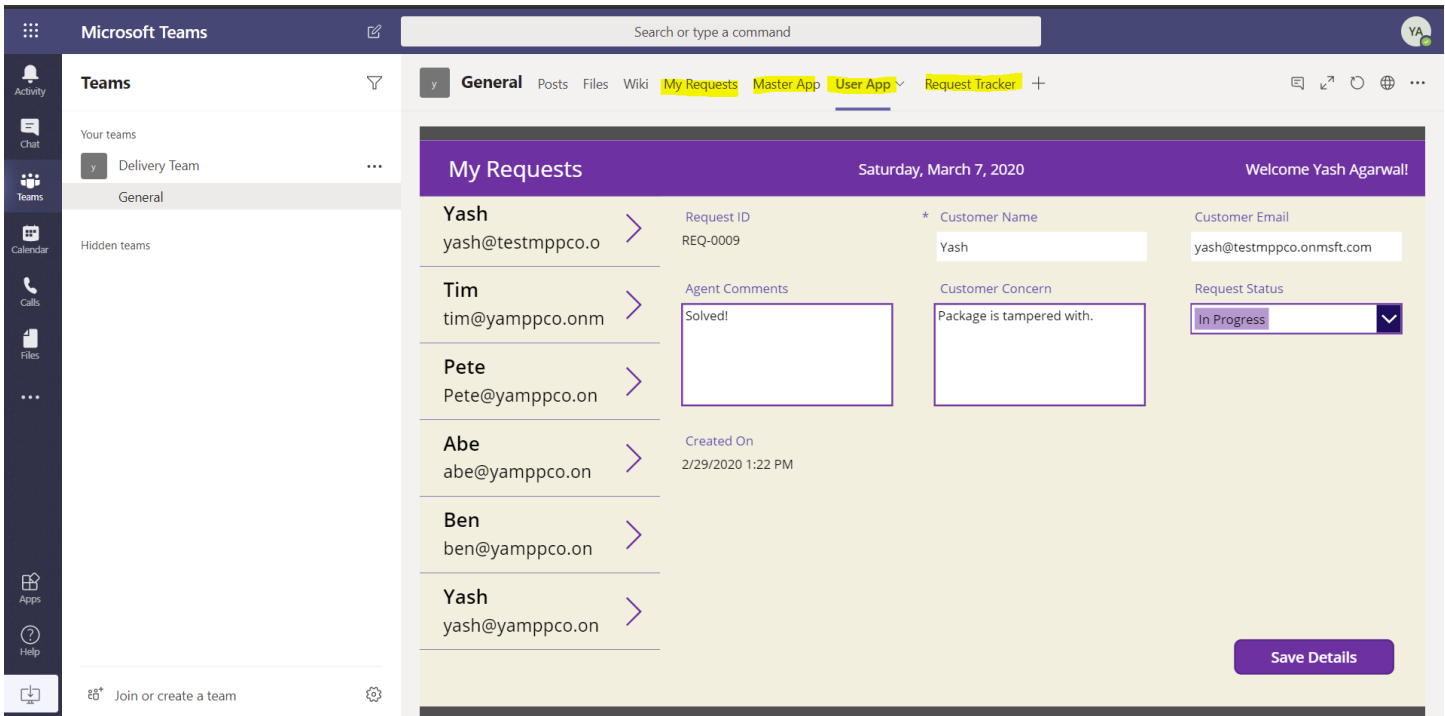
**Step 2: Add the canvas apps to the channel:** Navigate to the teams channel and select the '+' to add a new tab. Search for Power Apps and then select the app that you want to add to the teams tab. To add the other app, repeat the same steps.



Step 3: Add the Power BI report to the channel: Navigate to the teams channel and select the '+' to add a new tab. Search for Power BI and then select the appropriate workspace and then the report to embed on the tab.



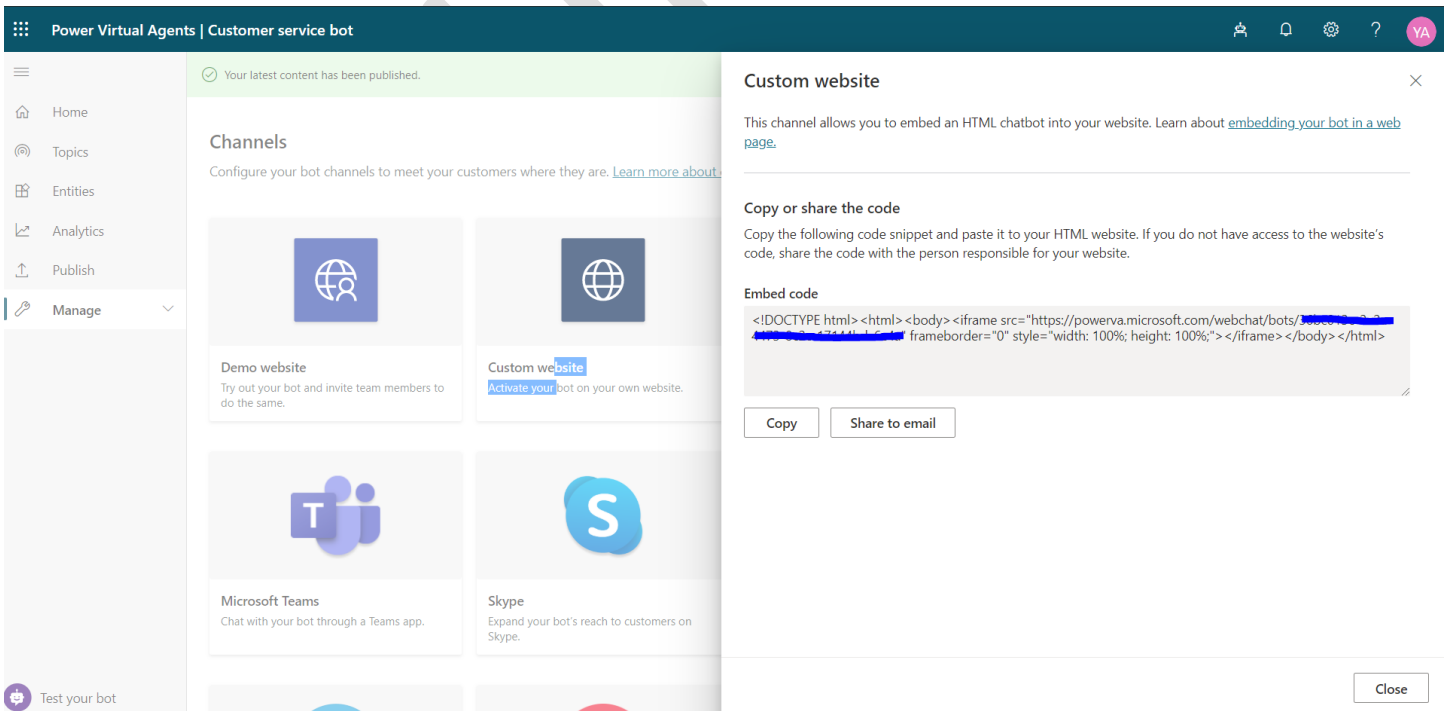
Once completed, all the above components display on the teams channel as shown in the screenshot below:



## Module 7: Deploy PVA to a web app

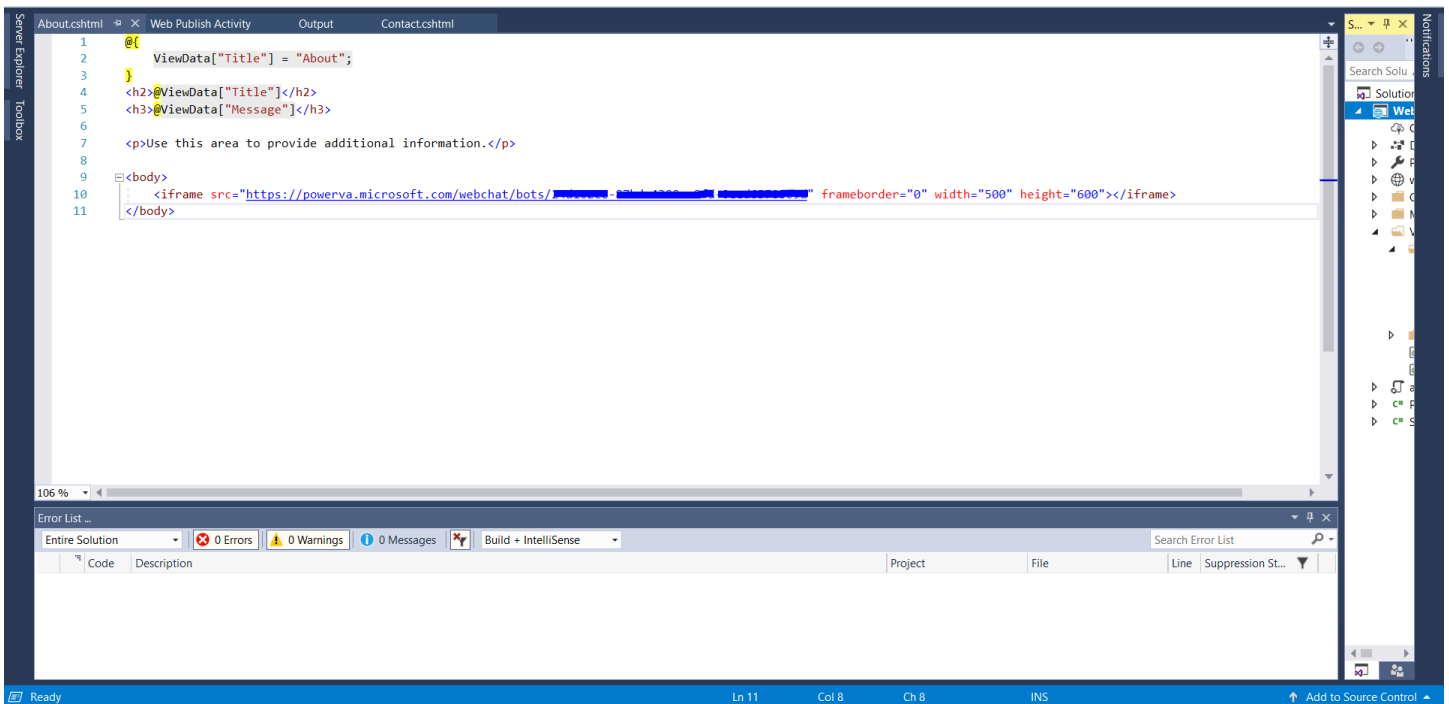
In this section, we will look at publishing and deploying the PVA to a web app using the iframe tag.

Step 1: Publish the Power Virtual Agent and select the channel as a custom website. This will generate the iframe code snippet to embed the bot. Once the code snippet is generated, copy it.



Step 2: As this is a customer service request bot, we will deploy it on the contacts page of the web app. Open visual studio and create a new web application project. This will create all the project files and then open the file for the page that you

want to embed the bot into in visual studio. Trim the above code and just copy the portion from <iframe> to </iframe> and paste it along with the code on the web page as shown in the screenshot below.



The screenshot shows the Visual Studio IDE with the 'About.cshtml' file open. The code is as follows:

```
1 @{  
2     ViewData["Title"] = "About";  
3 }  
4 <h2>@ViewData["Title"]</h2>  
5 <h3>@ViewData["Message"]</h3>  
6  
7 <p>Use this area to provide additional information.</p>  
8  
9 <body>  
10     <iframe src="https://powerva.microsoft.com/webchat/bots/..." frameborder="0" width="500" height="600"></iframe>  
11 </body>
```

The error list at the bottom is empty, showing 0 errors, 0 warnings, and 0 messages.

Step 3: Save all the files and now run the code. Navigate to the contact page and you should see the virtual agent render on the screen as shown in the screenshot below.

