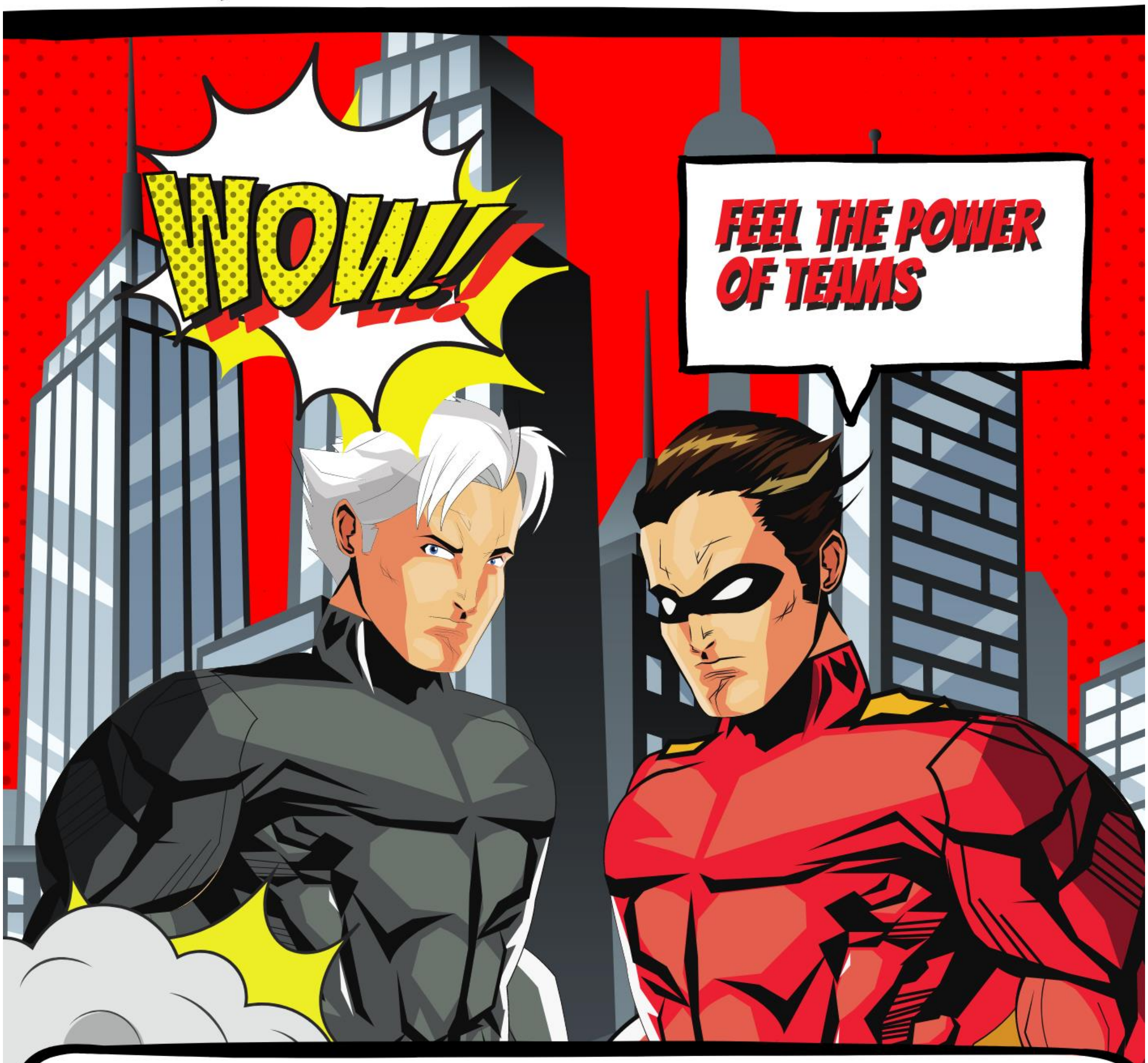
GETTING STARTED WITH DATAVERSE FOR TEAMS!



Priyaranjan KS

Contents

Getting Started with Dataverse for Teams

Introduction

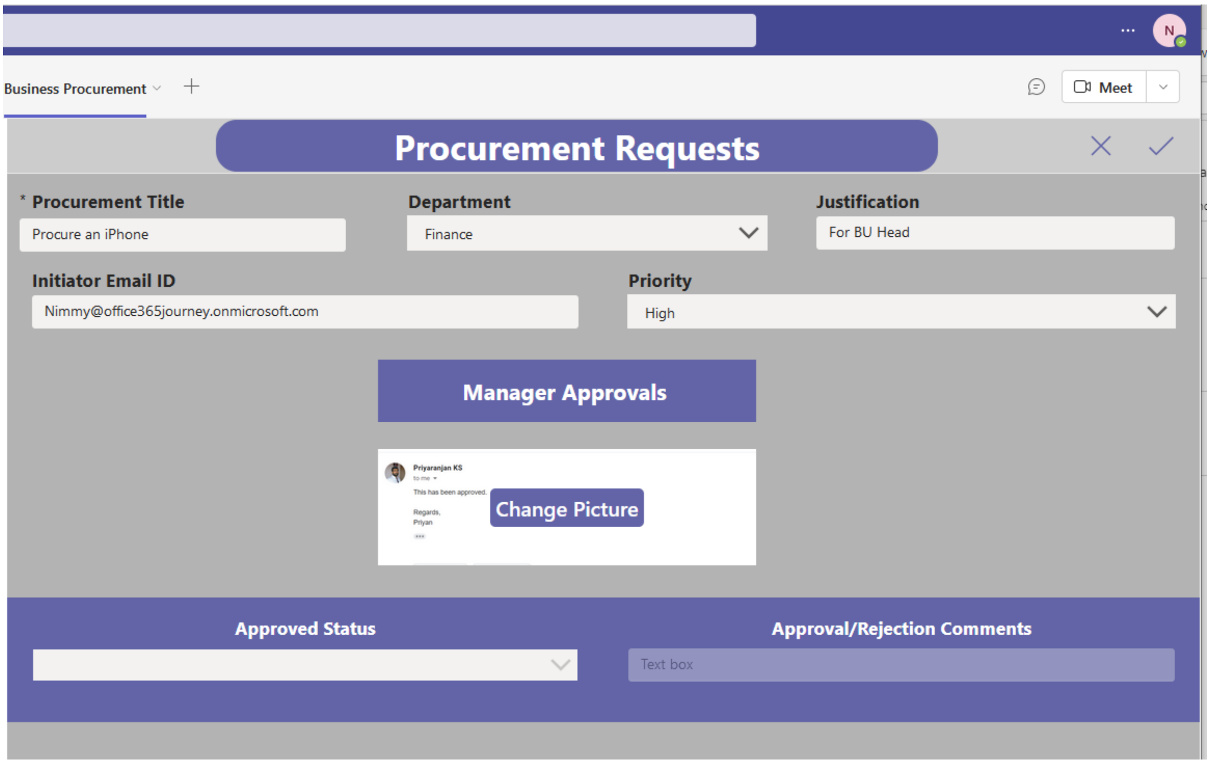
Stand alone power apps are a great choice for Individual and Team Productivity. However, to take the collaboration to the next level, having the Power App built and available within Teams is going to serve the purpose better. We can either create a Power App and embed it in Teams or we can create the Power App from scratch in Teams itself.

In this article, we will see how to get started with Power App Creation in Microsoft Teams.

Scenario

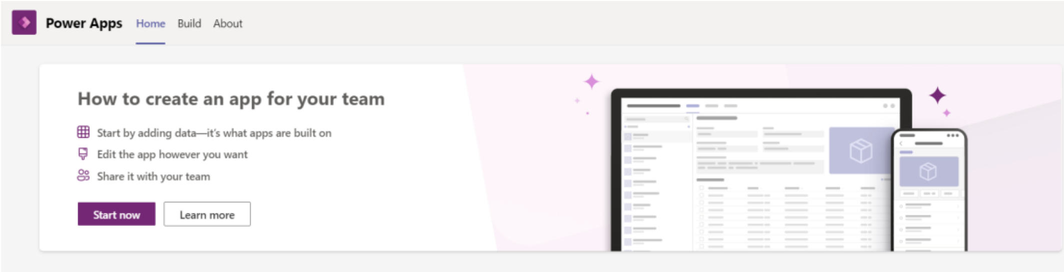
We will create a Power App within Microsoft teams to take care of the Procurement requests from different departments and having it within Teams will help boose collaboration due to the ease of access of the App.

We will have an input form to add the details as well as an option to approve Manager Approval images and submit the records. We will ensure that the Procurement records created by a user is available for only his/her viewing and only the admin will have the option to approve and add comments to the records which can be seen by the Initiator.

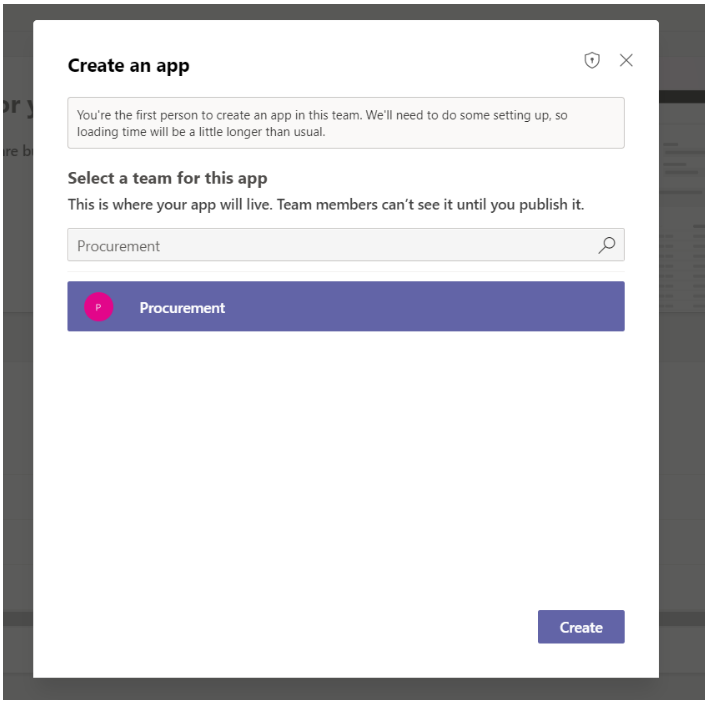


Implementation

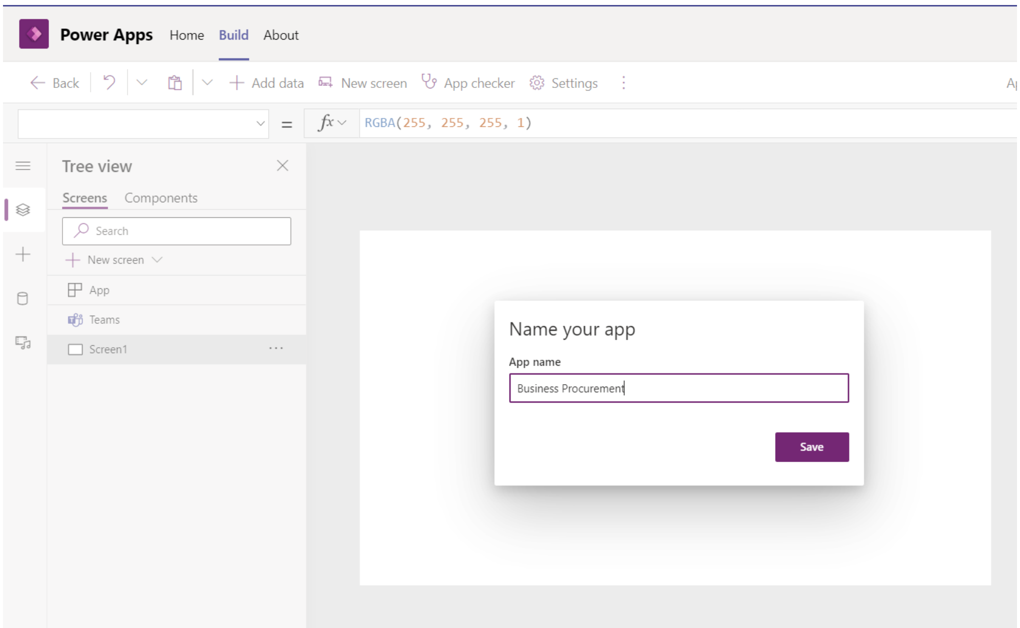
To get started, add Power App from the App Collection to Teams. Click on Start now to get started with the App Creation Experience.



Before heading over to the App Creation Interface , select to which team, the app has to be added.

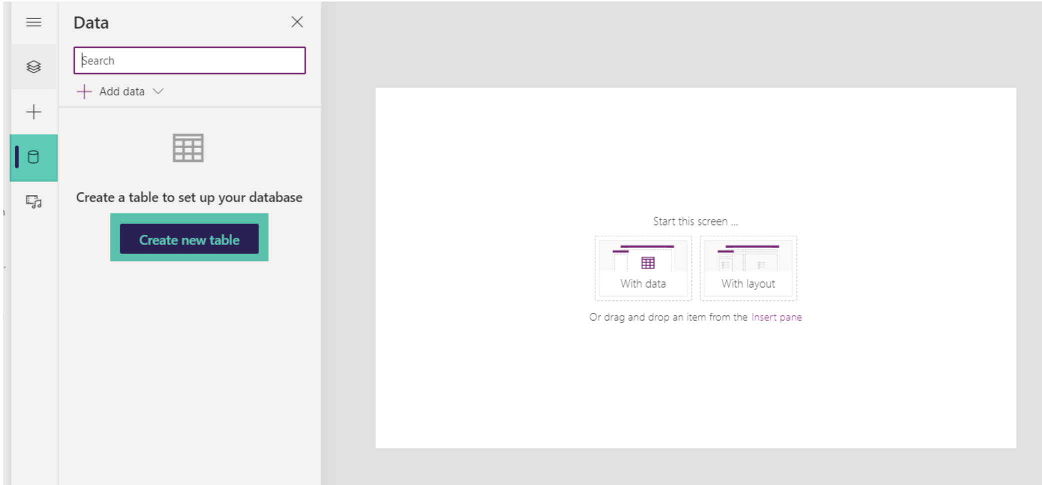


Clicking on Create will open the App maker where we can name the app.

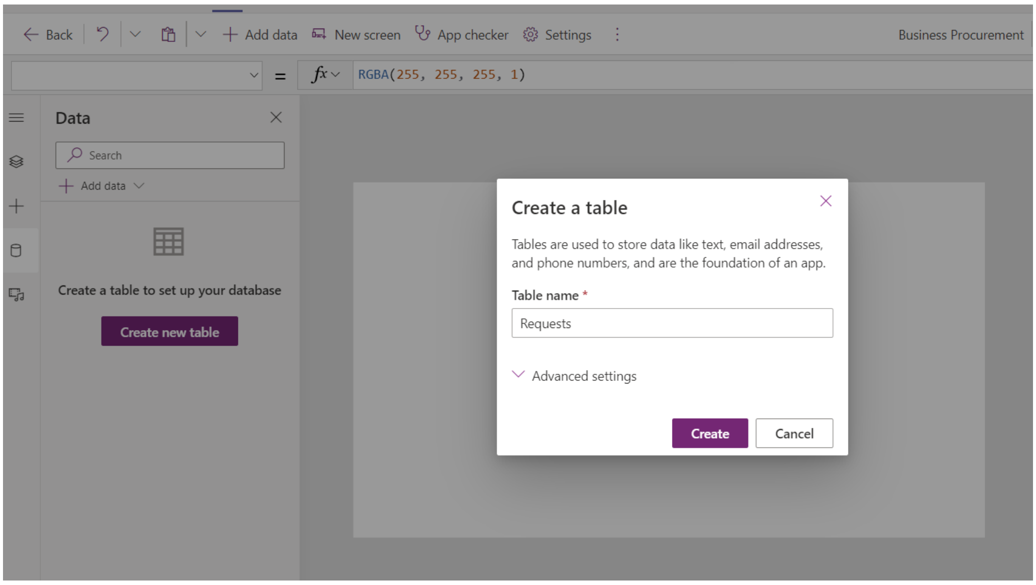


Create Table

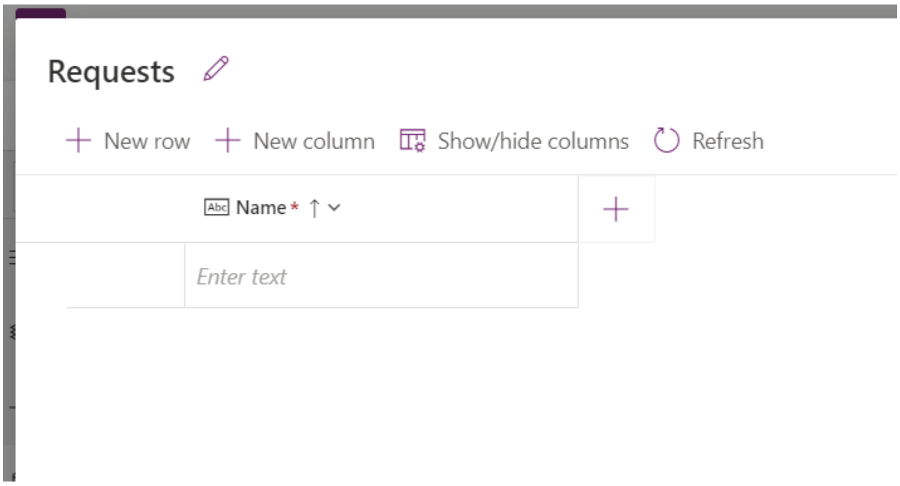
Once we have created the App, lets create a Table in Dataverse which will hold the information collected using the App.



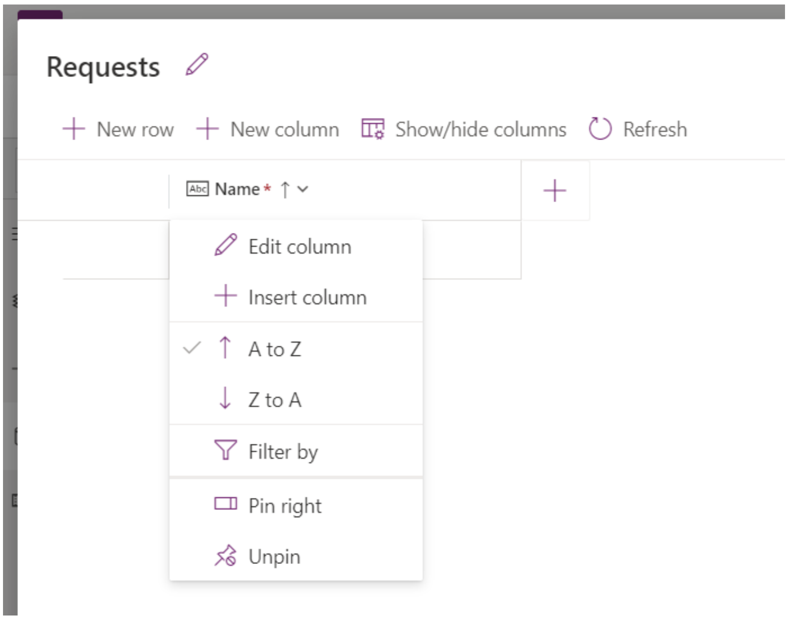
Let’s name the table as Requests.



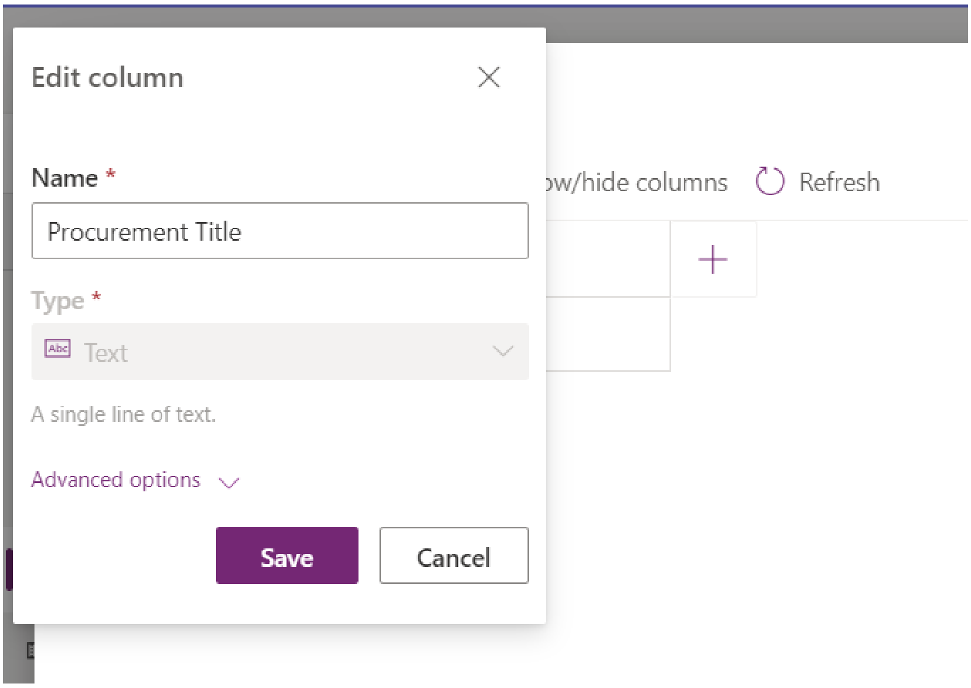
It will by default add a column by the name “Name”.



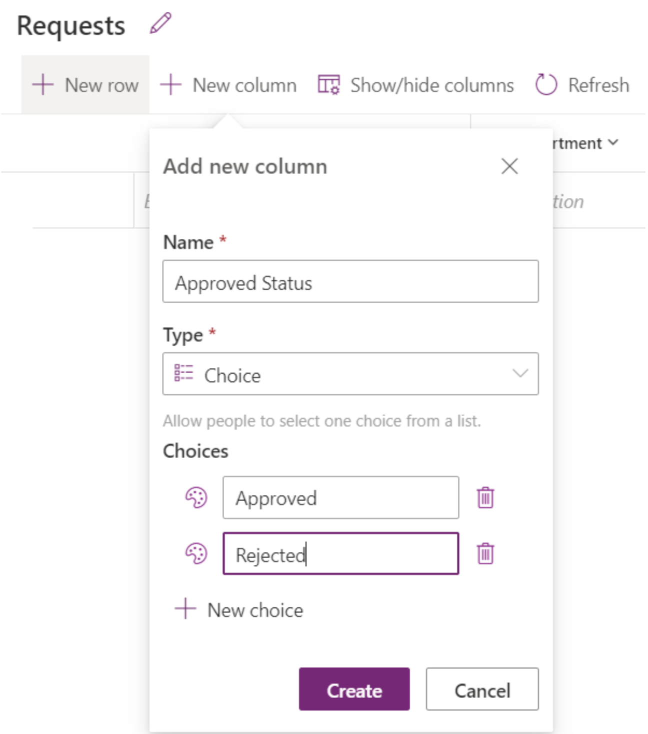
We can either rename it or add a new column. For time being, we will rename it as Procurement Title



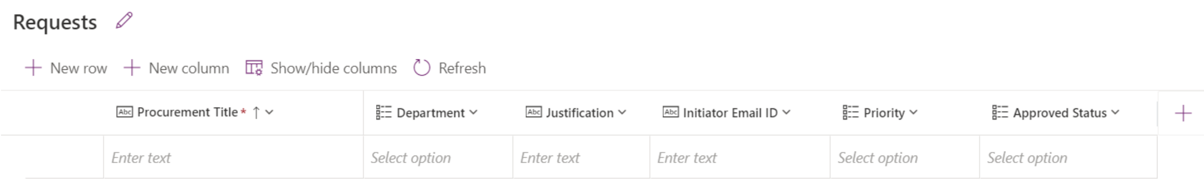
Since we are editing an existing column, it does not let us edit the Data Type. So, ensure that you are clear on the data type you would want to use for a column you are creating in a Dataverse table.



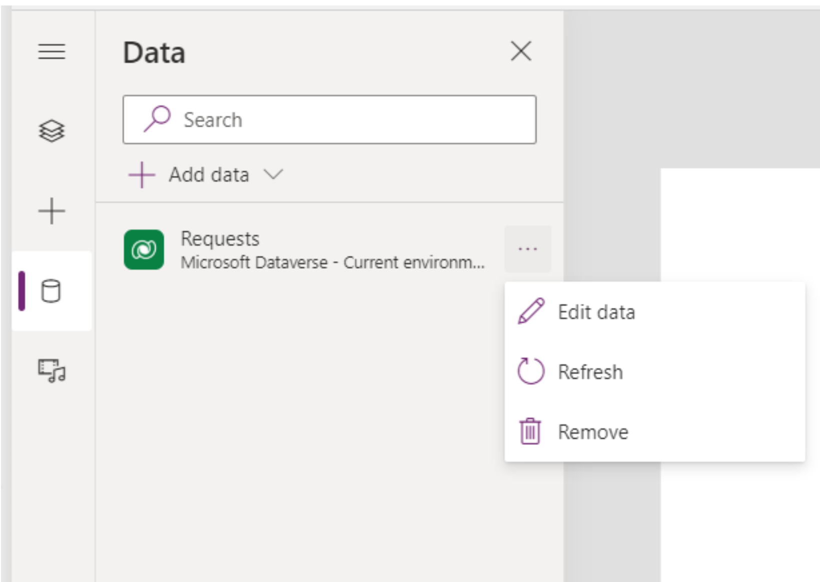
Let's go ahead and create few more columns like Department, Justification, Initiator Email ID, Priority. We will also add a column named Approval Status as a choice column with Approved and Rejected Values.



The table with the created columns will look like:

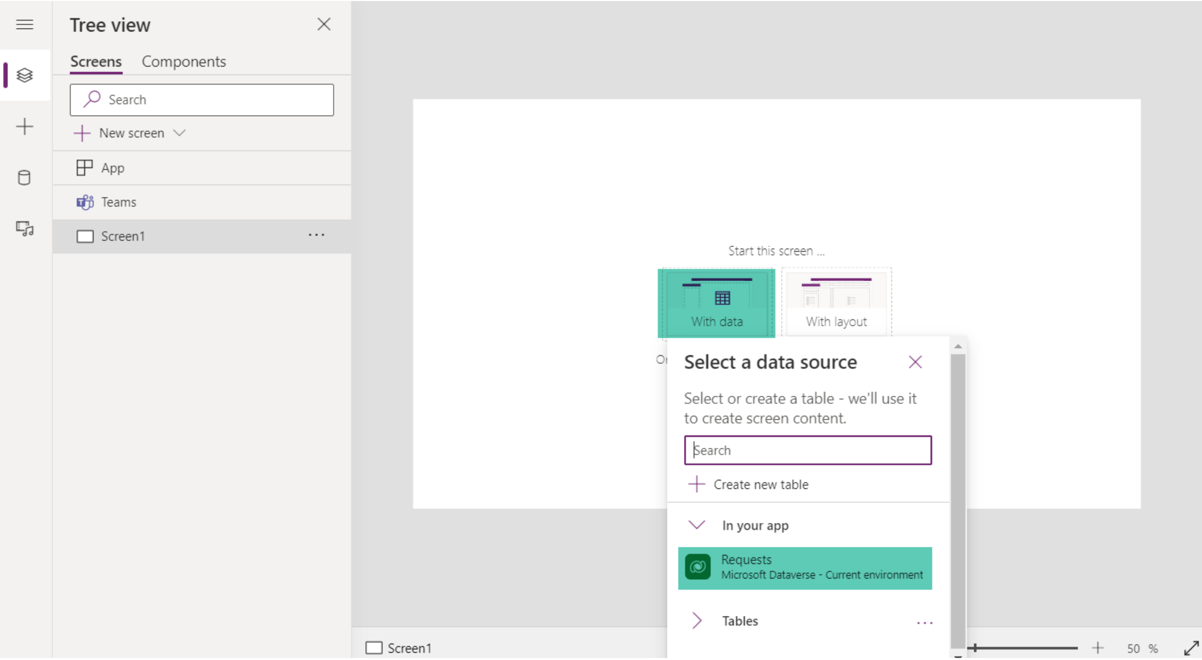


Now, we can see the newly created table in the Data tab. If we want to add data to the table at some point, we can click on Edit Data to update the table.

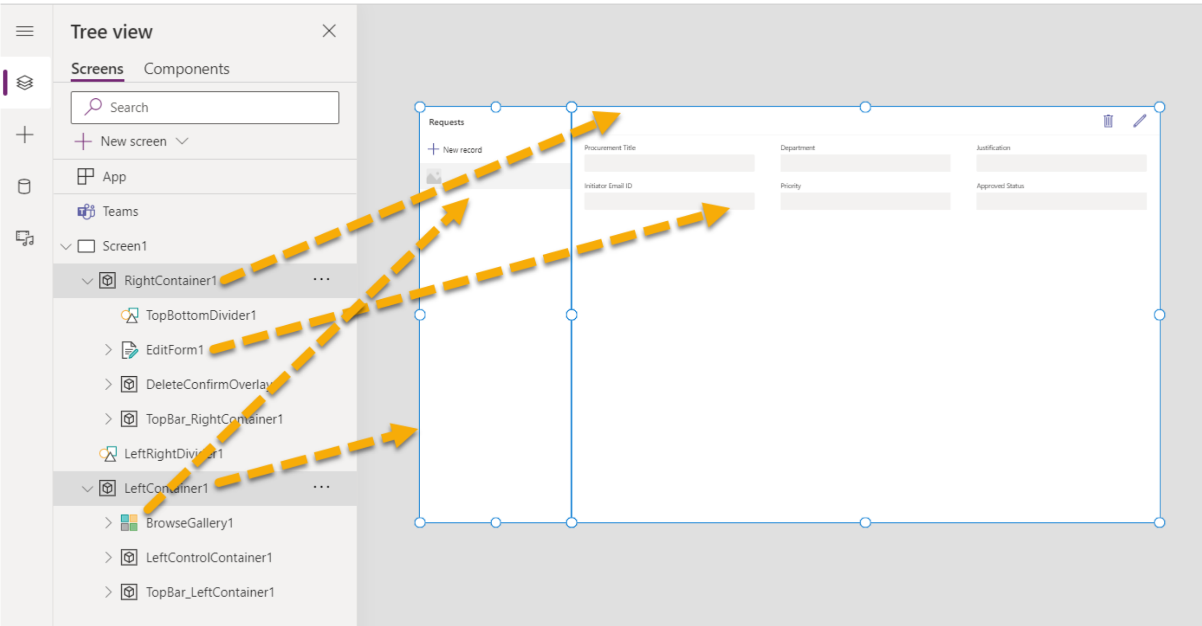


Building the screen

By default we have a blank screen available in the app , but since we have already created a back end table

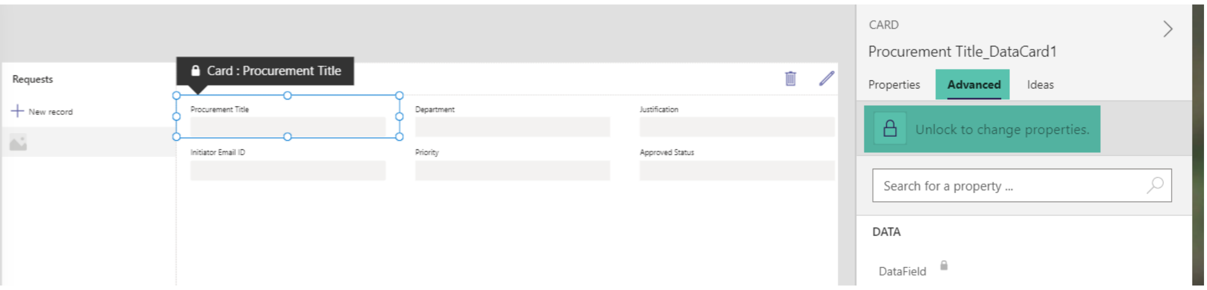


Clicking on "with data option" in the screen will update the screen with a template with which we can interact with the Requests table. It create a Left Container with a Gallery that lists the Items from the recently created table. It also has an option to add a new record. To the right, we have a Right Container which houses an Form which toggles between New and Display form when we click on the "New Record" or the edit Icon at the top right corner. The delete icon, let you delete the current record from the table.

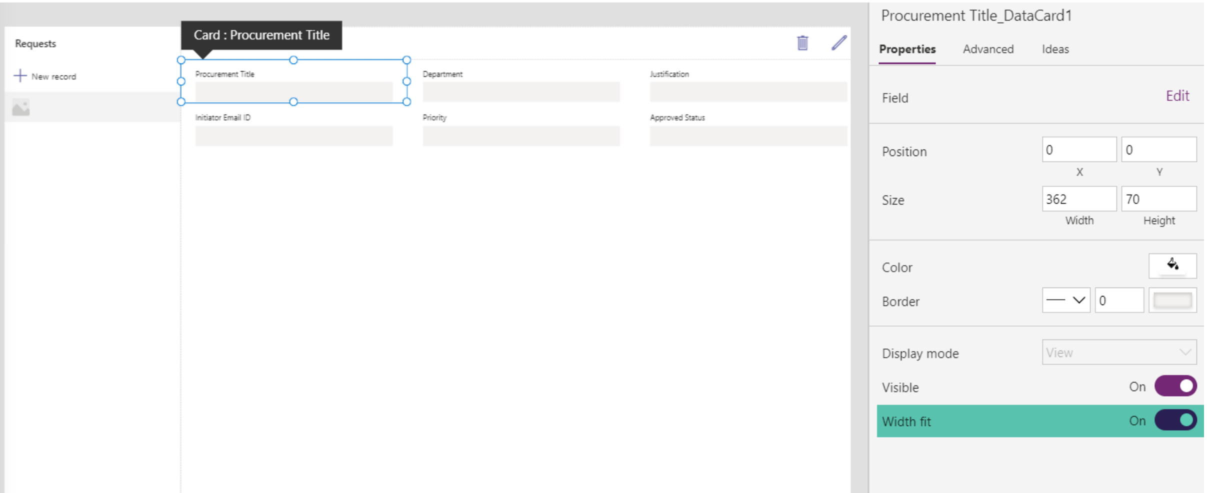


Subtle Design Changes

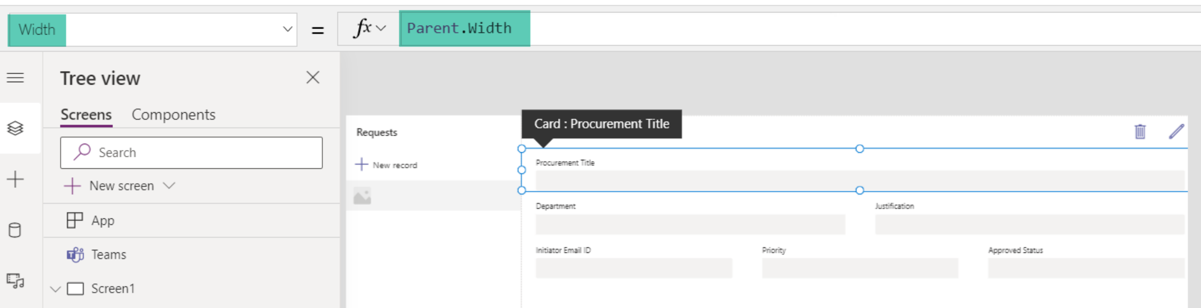
In case we want to modify the data cards and change the font size of update the width of the controls,we can do that by Selecting “Unlick to change properties” option from the advanced tab.



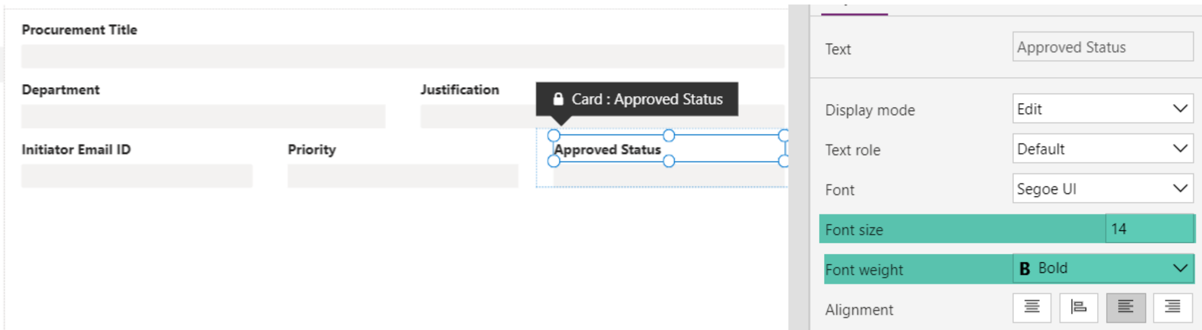
To change the width of the control, set the Width fit to Off to play around with the control.



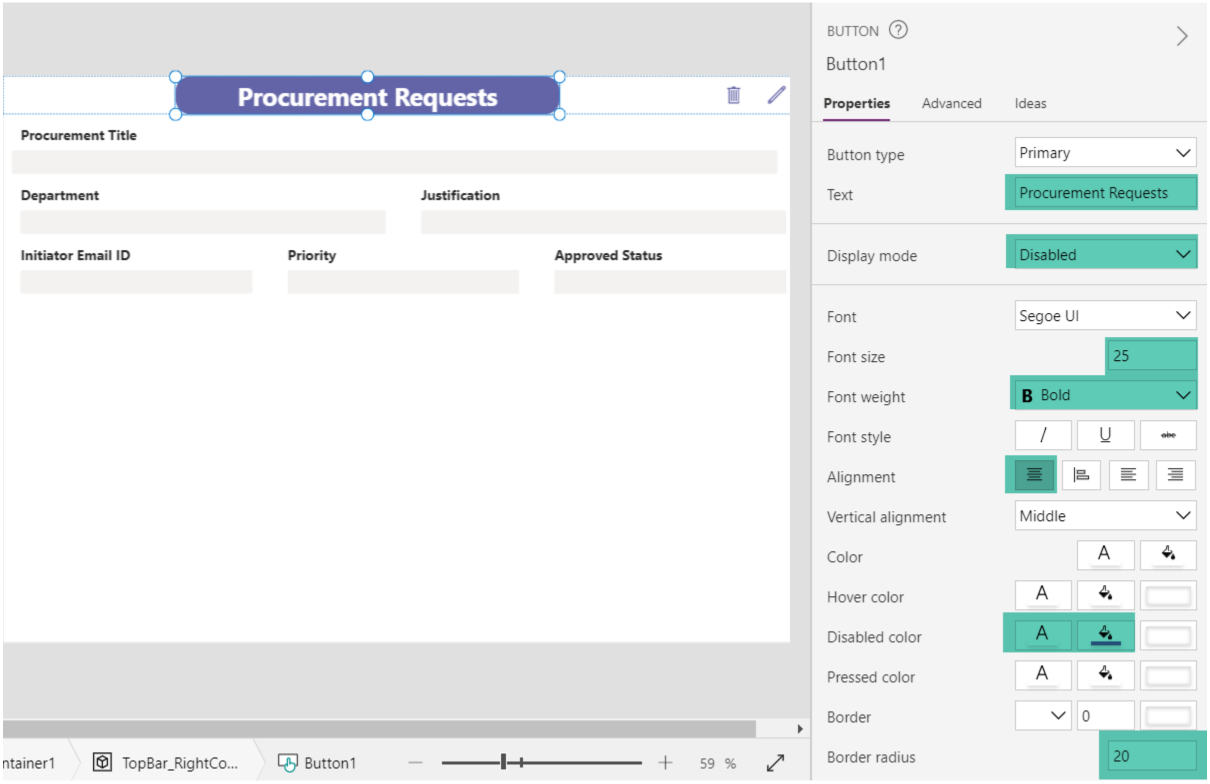
In case we want to set the width of control to occupy the full screen, we can set its width to Parent.Width after turning of Width fit.



We have also updated the Title’s Font Size and Font Weight

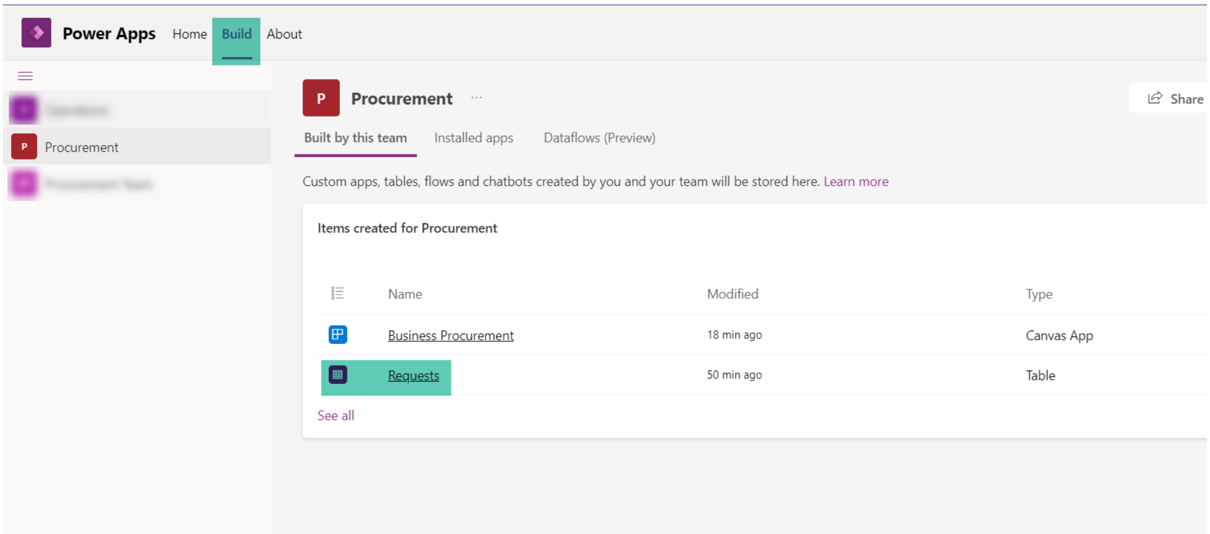


Let’s also add a Button to the Top Container so that it can be considered as an App Name. We will set the button to be disabled so that it will non clickable and behaves like a label that can be styled. We will also set its Disabled Colour to as shown in the settings below and add a border radius to it. Reason for using this approach is because, this provides a better flexibility in styling the component to give a header feel.

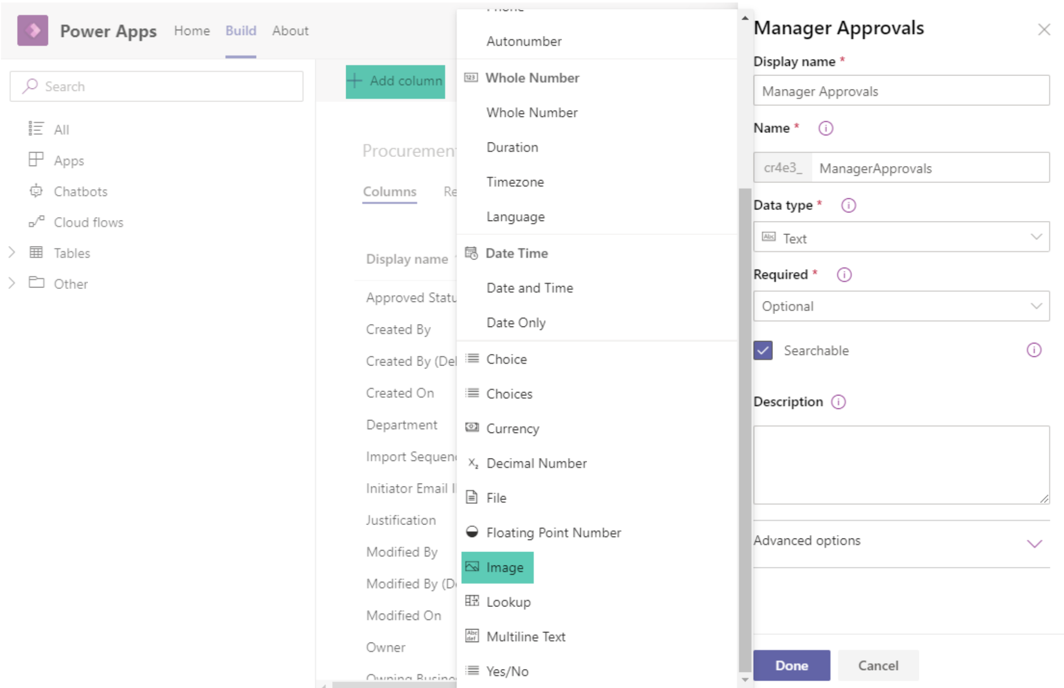


Adding an Image Column

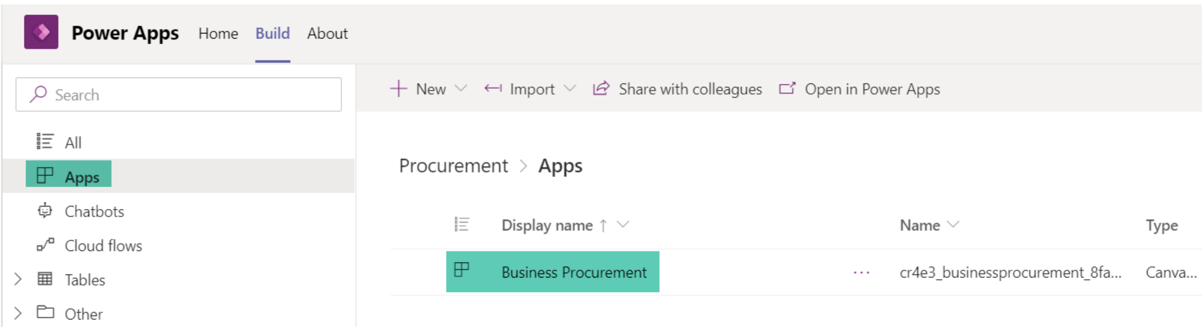
Now let’s see how we can add a new image column to the already created Dataverse table so that we can use it to store Manager approval for the Procurement requests. To do this, head over to the Build Tab in the top menu and select the Requests table that we had created for the app.



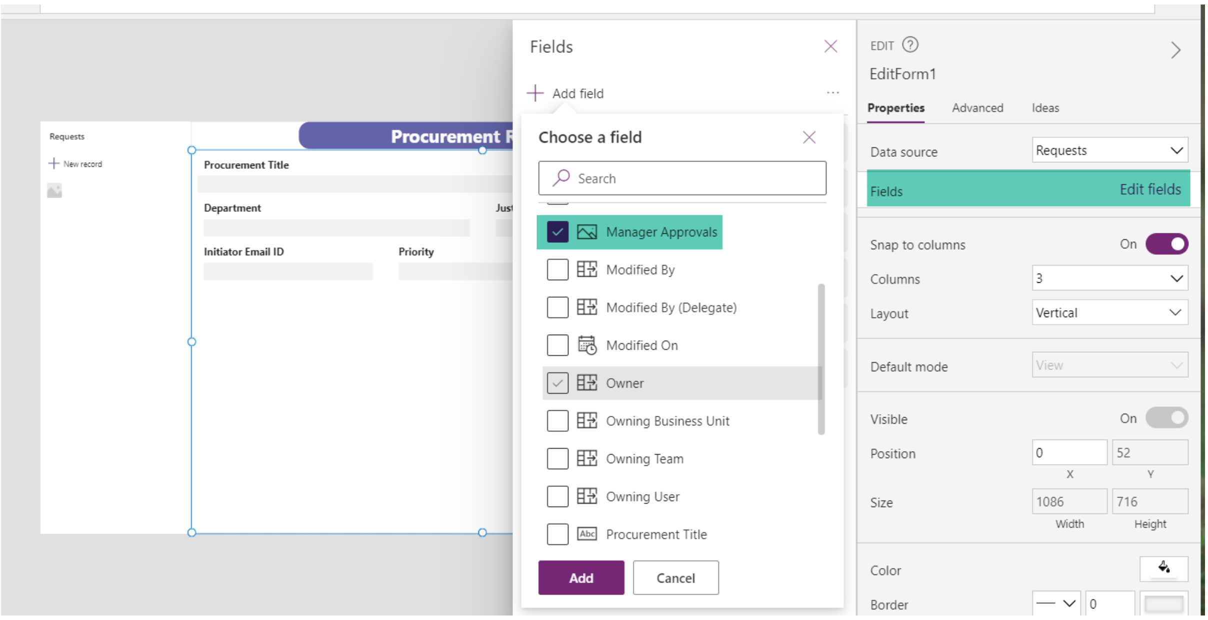
Select Add Column and we can see that there is lot more data types available from this Build Section view than in the Table view we used for column creation last time. Select the Image data type and click on Done. Finally Click on Save Table.



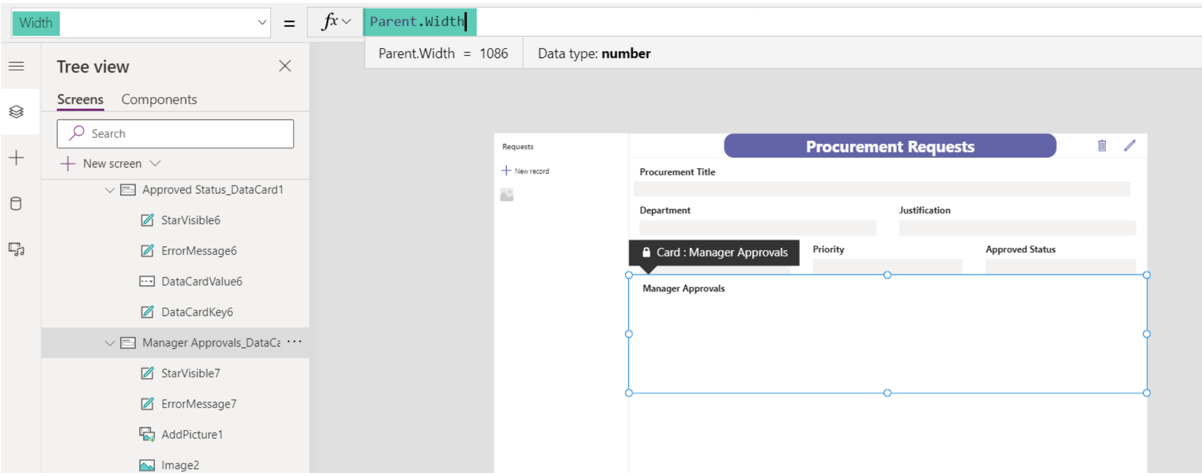
Now lets head back to the App that we were developing by clicking on Apps -> <App name>



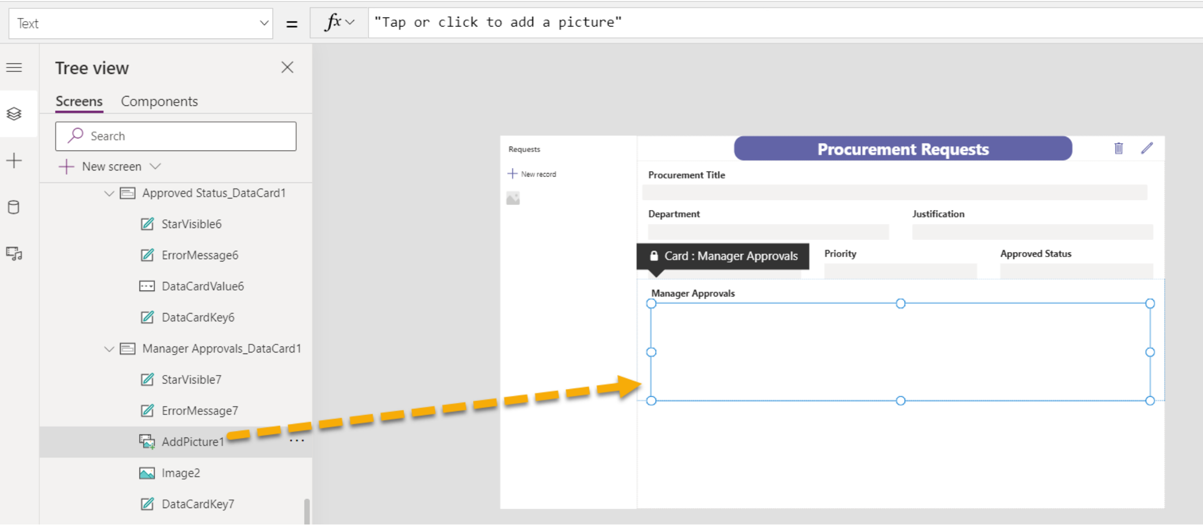
We will now add the newly added Manager Approvals Image field to the form , by clicking on Edit fields and selecting the Manager Approvals field.



This will add the Data card to the form, we will stretch it to fit the Parent Width using the below expression in the Width property.

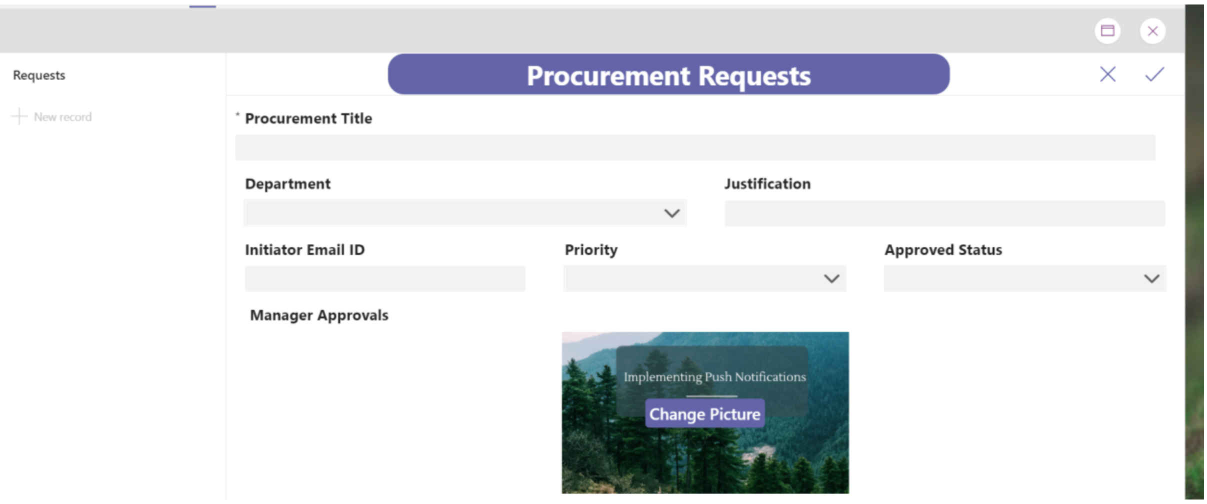


We can see that the Datacard for Manager Approvals Image field has a label named Manager Approvals as well as an AddPicture control.

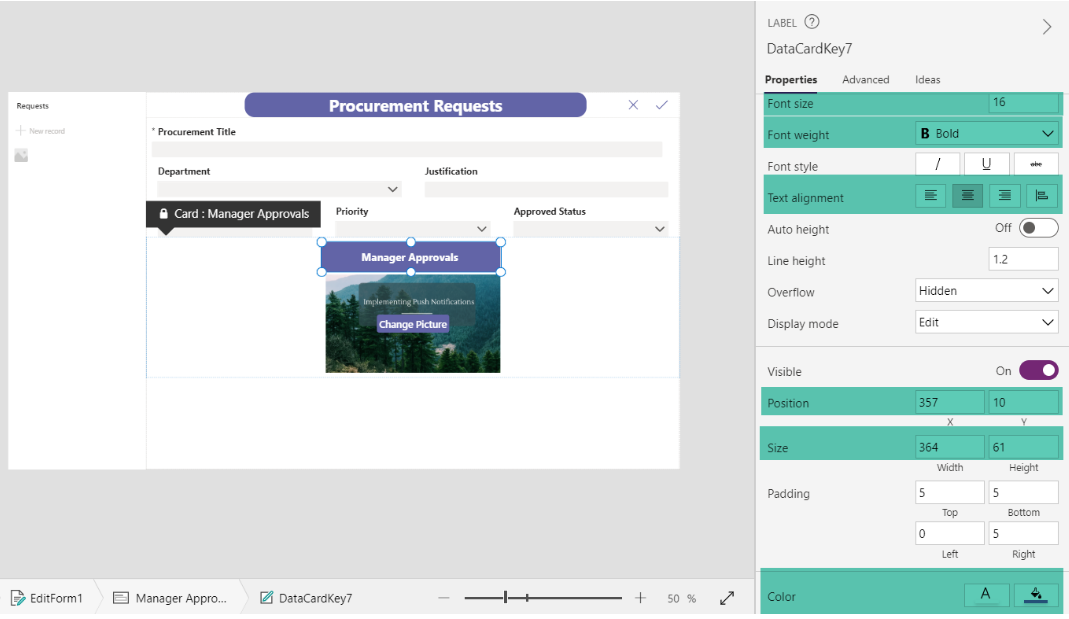


Since the form is in display mode now, we will not see the Add Picture Control button in action. But once we preview it and go into New or Edit more, we can see a Button that will let us explore the file system and upload the image. In case of mobile app, it will also let us use the camera to take a snap of the images.

I have just previewed the app and clicked on new record which shows the Button to add the image and on browsing and selecting the image it will be shown like below.

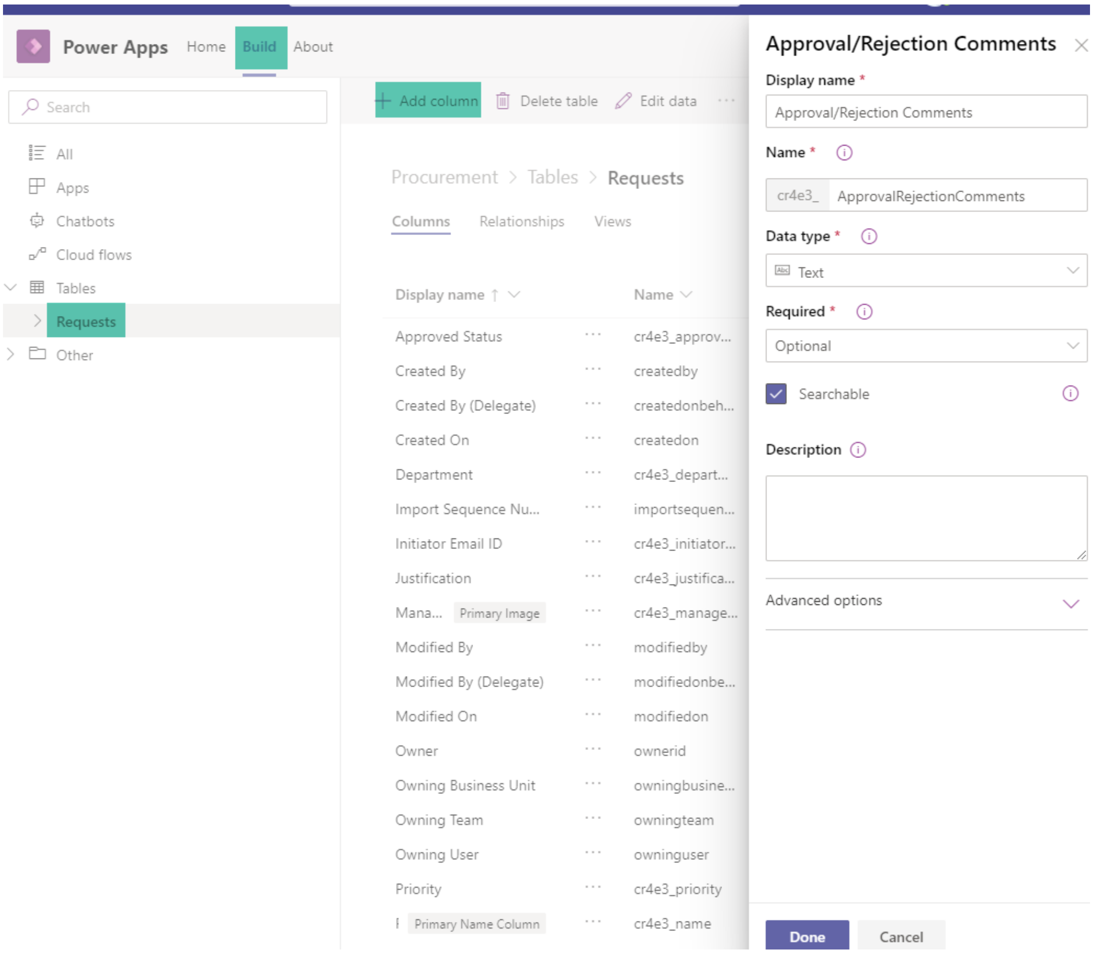


Now let's, make a change to the Manager Approvals label and centre align it to give it a better aesthetic look. We have applied the below settings to the Label of the Data Card to look like below.

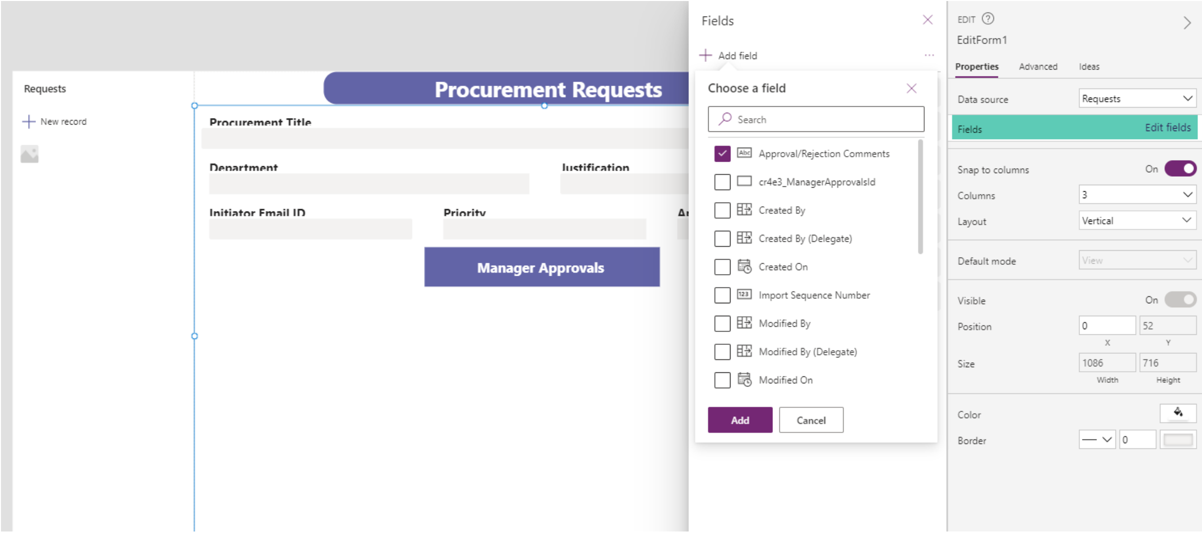


Approval/Rejection Comments

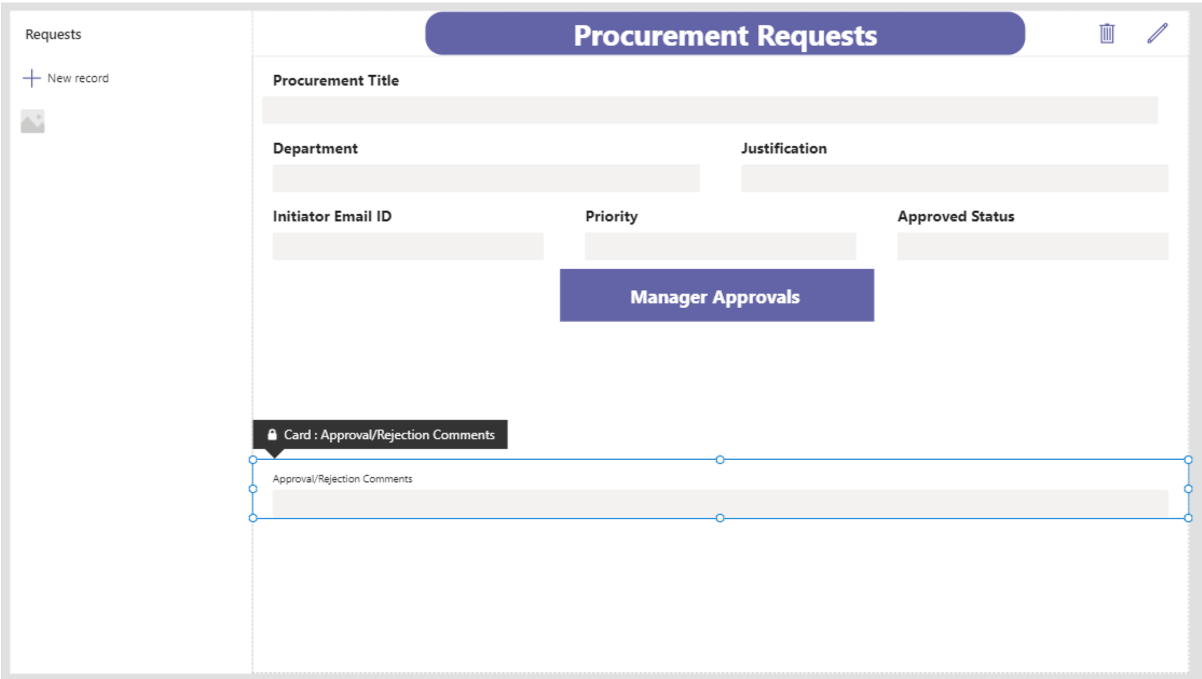
We will head to the Build tab, select the table and add one more column to capture the approval/rejection comments.



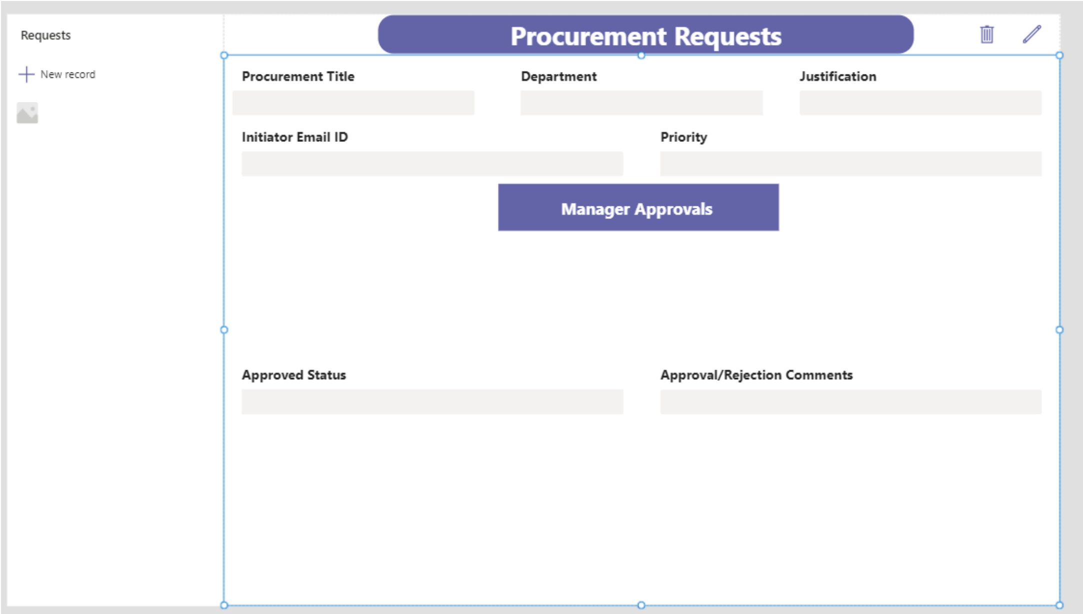
Click on Done, followed by that Click on Save Table and lets head back to the app. We will add the recently added column to the Screen by selecting Edit fields of the Edit form.



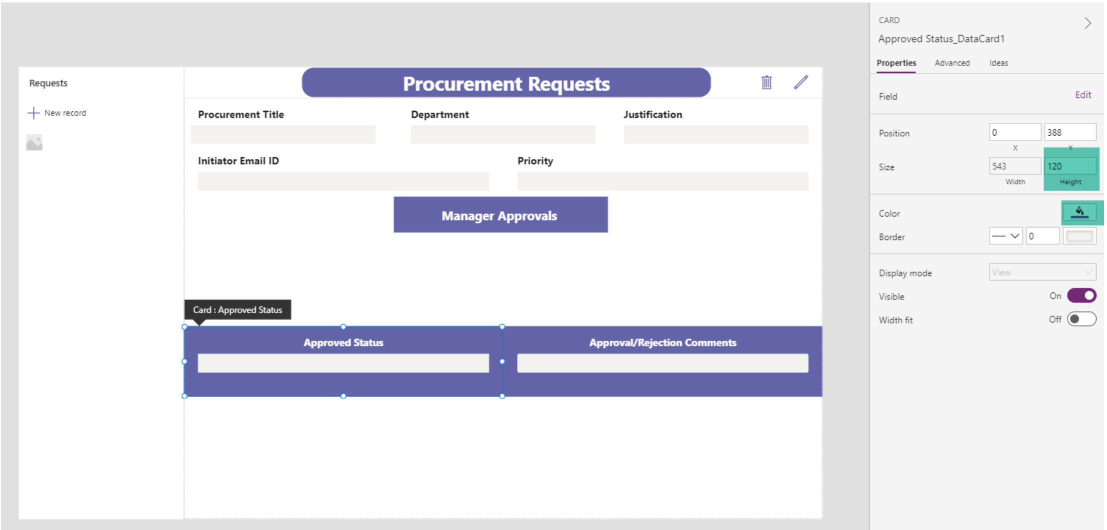
The newly added table column is now available in the Screen.



We will bring down Approved Status and align it along with Comments field.

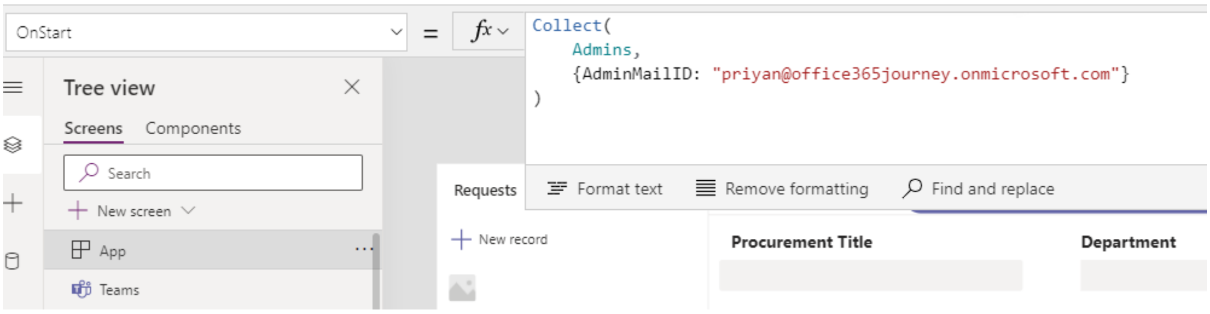


Let's give some styling to both Approved Status and Approval/Rejection Comments labels by applying a background colour and increasing its height



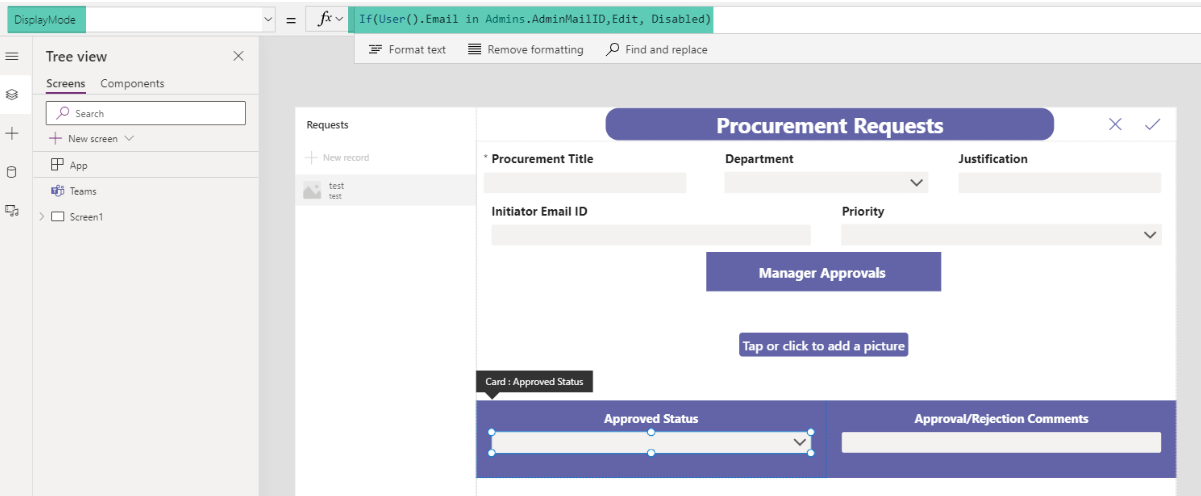
Control Visibility

The Approved Status and Comments fields has to be editable only for the admins. To achieve this, we will add some conditional rendering expressions. To get started with that lets add the the Admin email ids to a collection in the App OnStart.

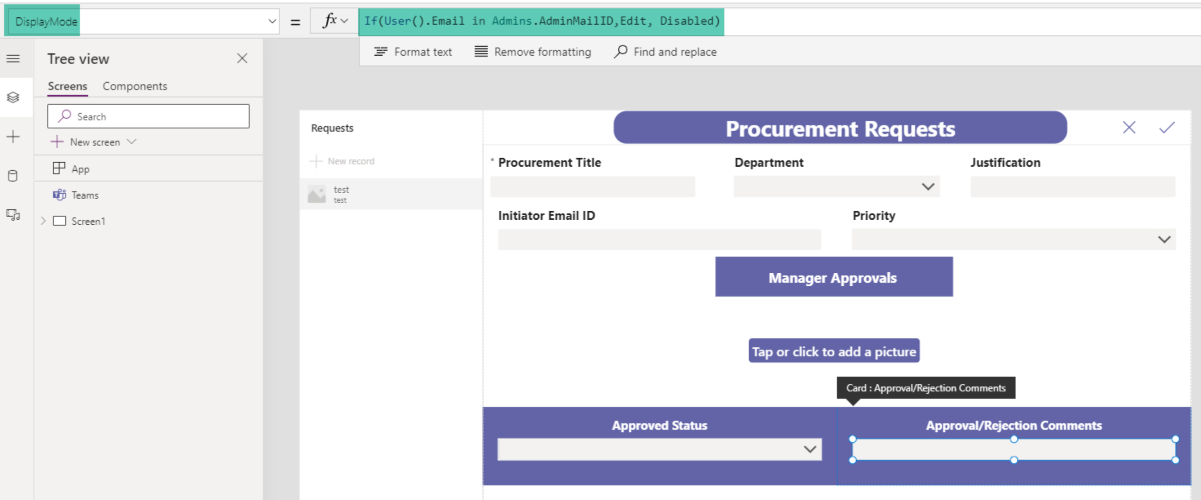


Now we will select the Approved Status Data Card and set the DisplayMode property using the below expression

If(User().Email in Admins.AdminMailID,DisplayMode.Edit, DisplayMode.Disabled)

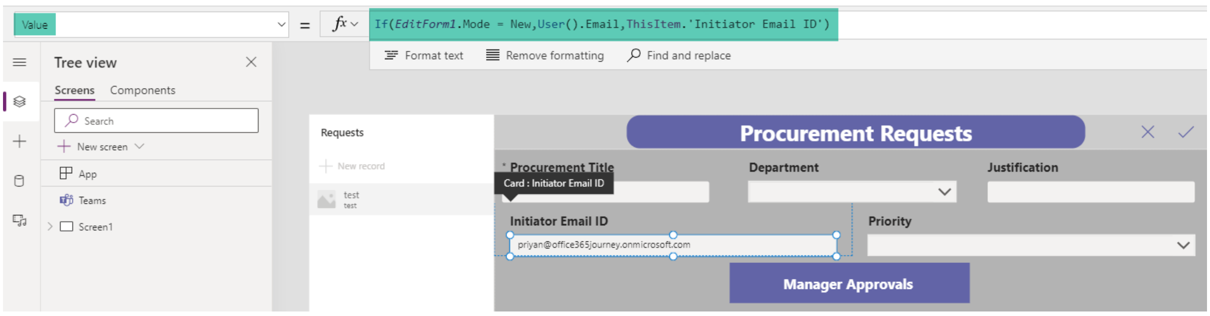


Same expression goes into the DisplayMode for Comments fields as well



Filtering Requests based on Signed in User

We will now add an additional functionality to show the records based on signed in user. So as to do that, lets assign the initiator email Id with the current logged in user's email so that whenever a new records is created, the field is automatically populated with the email value. We will then use this email id to filter the gallery in the left side.



If(EditForm1.Mode = FormMode.New,User().Email,ThisItem.'Initiator Email ID')

This expression will check if the form is in New Form Mode,if so it will assign the current logged in user email id to the field, else it will retain the existing value.

We will now head over to the Gallery in the left container and set its Items property to the below expression

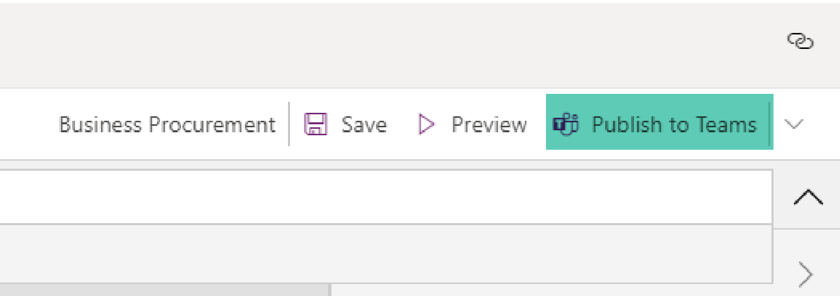
If(User().Email in Admins.AdminMailID,Requests, Filter(Requests,'Initiator Email ID' = User().Email))

It checks if the user is an admin, if so , the entire records will be show, else only the requests where the signed in user email id is present in the Initiator email id field will be shown. This way, we can show only the current user's records.

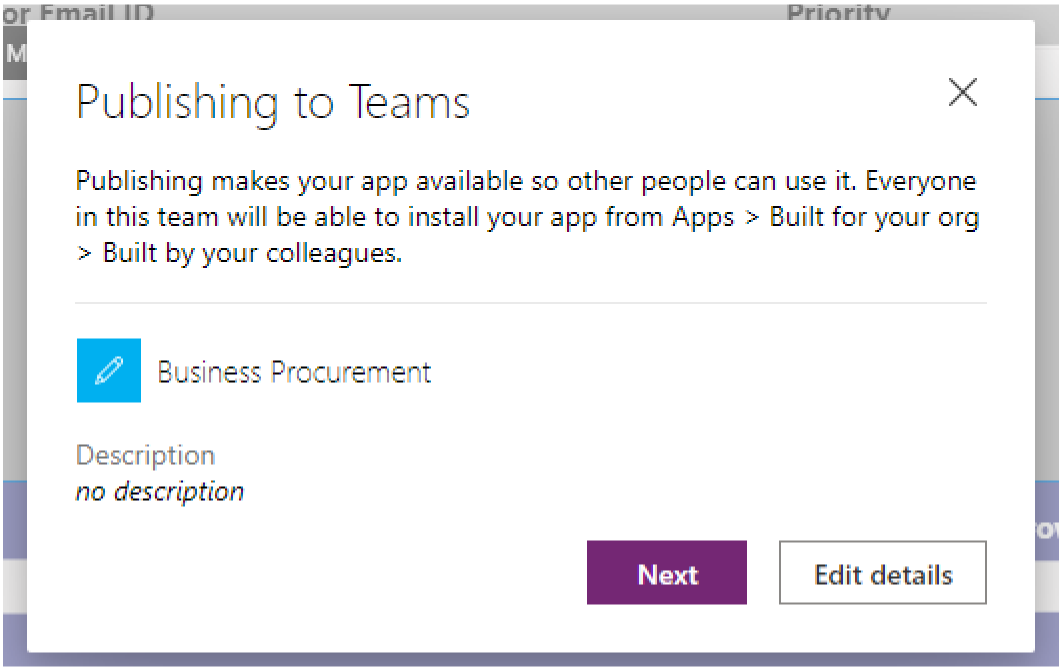
Thus we have completed the creation of a simple procurement Requests App , lets go ahead and Publish it.

Publish the App

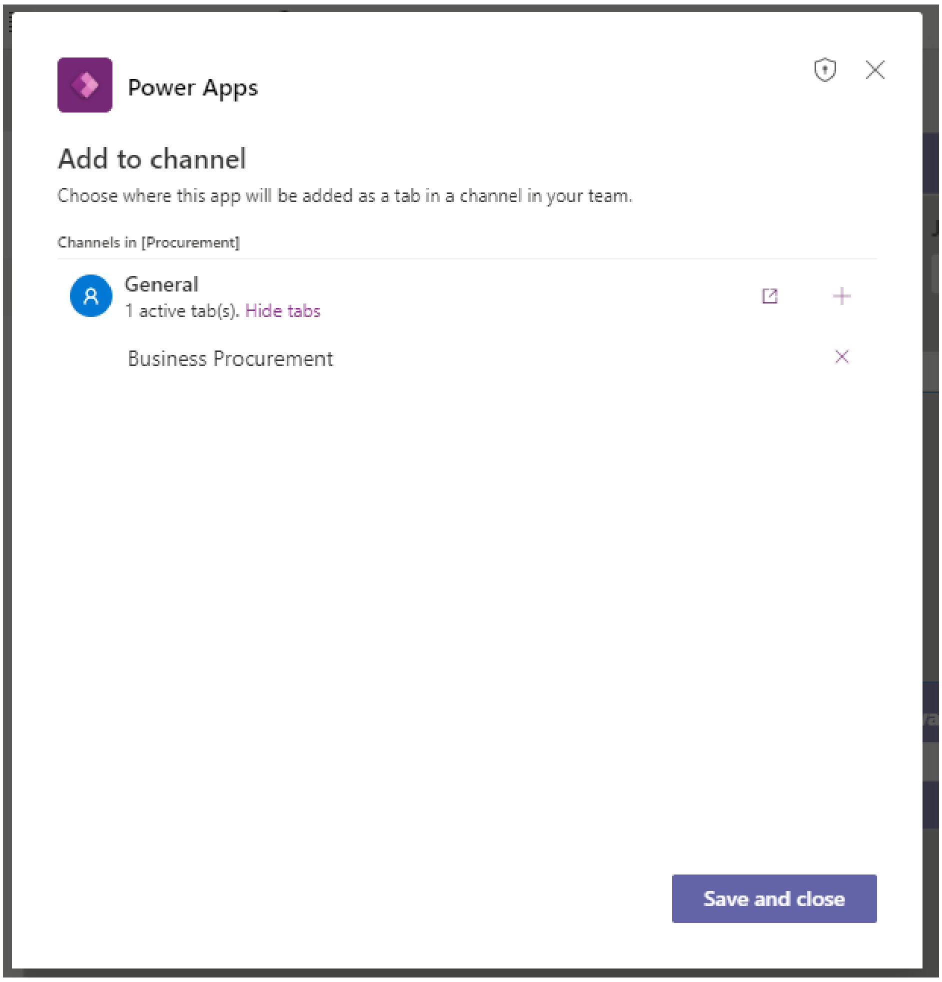
We can Publish the App so that it becomes available to other users.



Click on Next

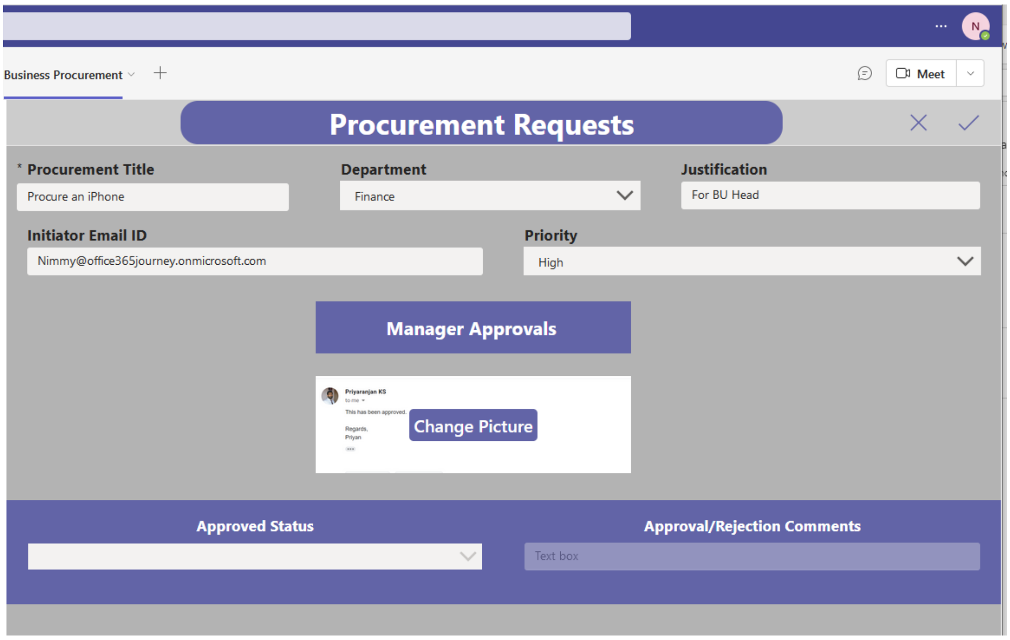


Select the + icon to add it as Tab to the Teams Channel and Save

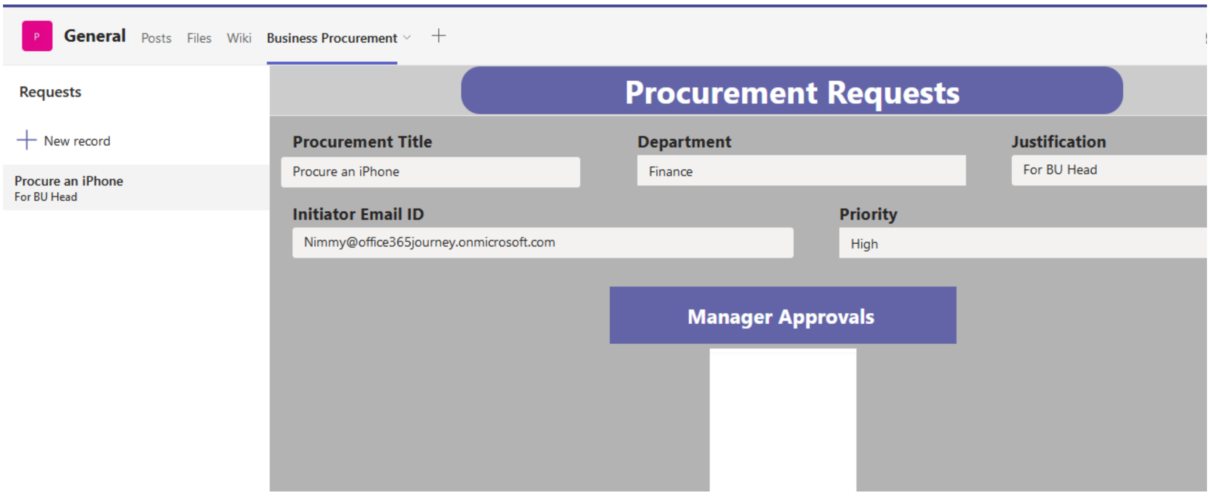


Test the Teams App

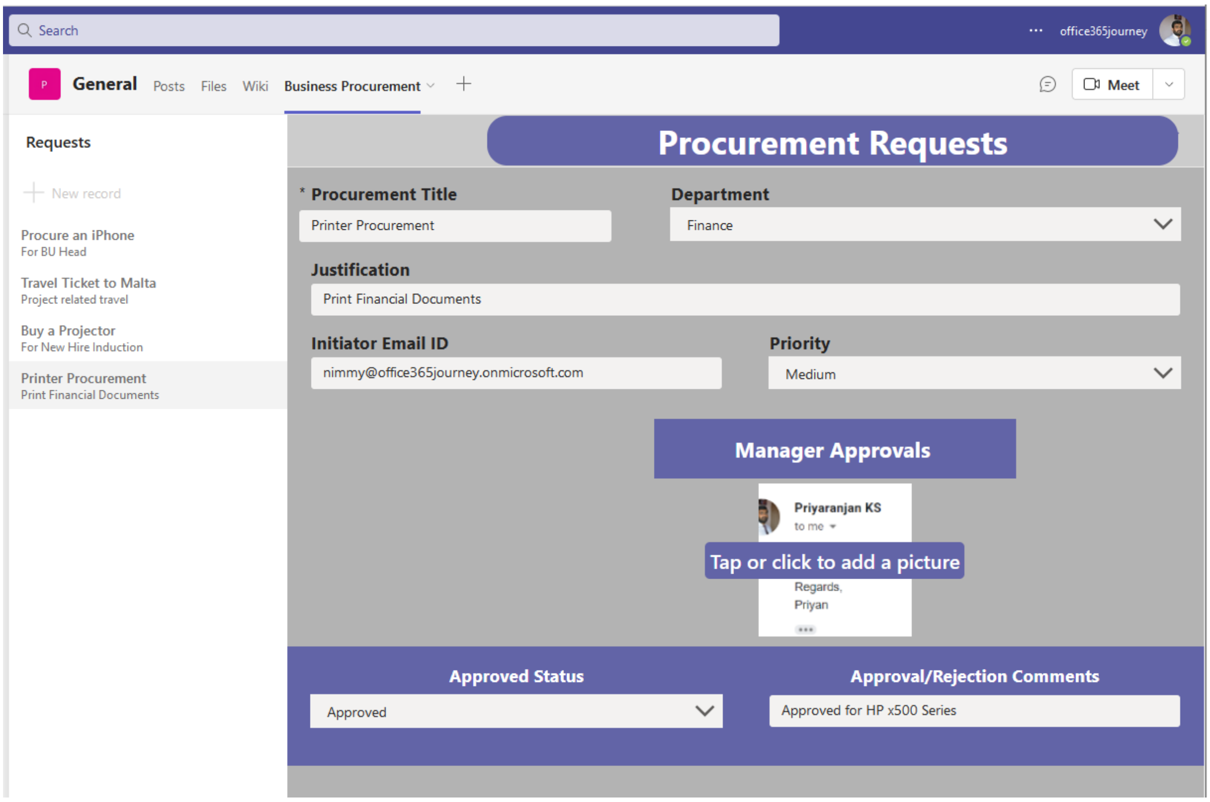
Let's head over to the Channel and test the app by creating a new record. I have logged in as a normal user who is creating a Procurement request.



The request will be submitted on Saving it. We can even come back later and edit it. The approved Status and Comments will be read only as its only available for admins to edit. The current user will be able to see on his/her record in the left gallery



When we login as the admin, we can see the entire records created by various users and we can go ahead and approve and add comments as well.



Summary

Thus, we saw how to create a Procurement Power App from scratch directly within the Teams environment and set conditional access to the records created in the app.

Sharing a Dataverse for Teams App

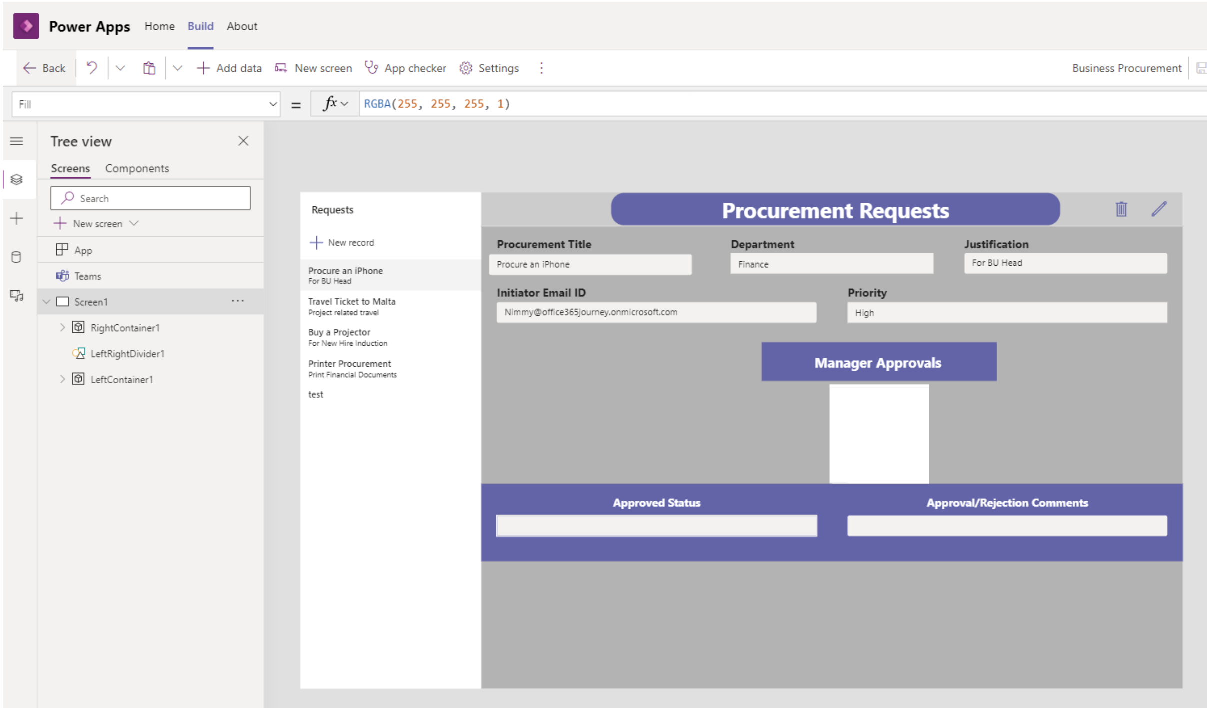
Introduction

We saw how to create a Power Apps for teams in the previous section. With the ability to create Power Apps from scratch within Teams help better accessibility and boost collaboration between teams. We also have the possibility to share the app with multiple teams.

In this section we will see how to share an already created Power App with other colleagues.

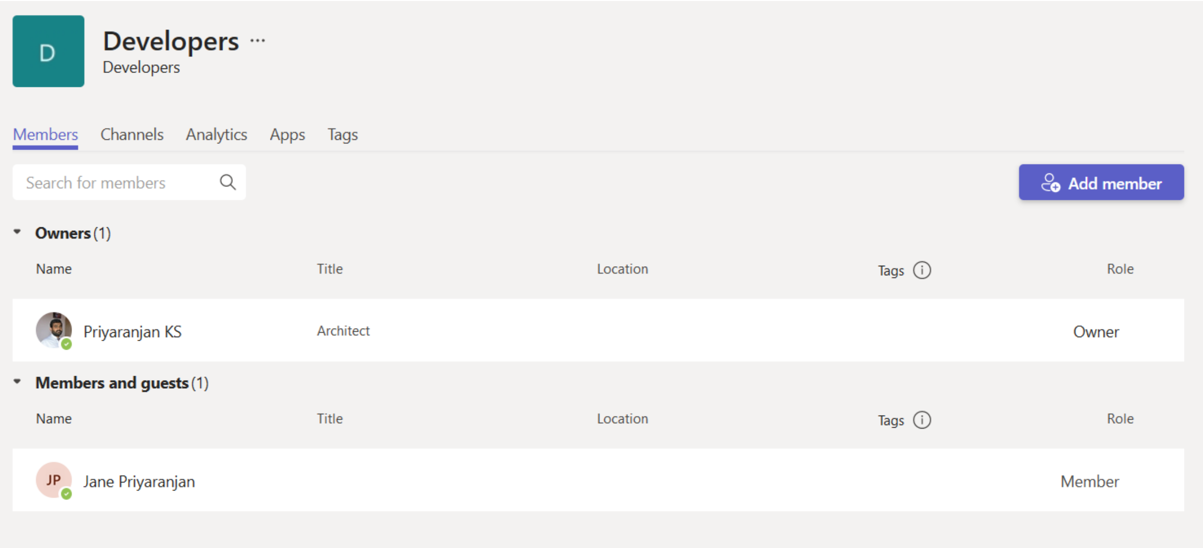
Scenario

We have created a Power App for accepting Procurement requests which is added to the Microsoft Team by the Power App Developers who initially created the App. We need to share the app with another group of Global Users so that they can start using the app. Let's see how we can share the below app with the end users.

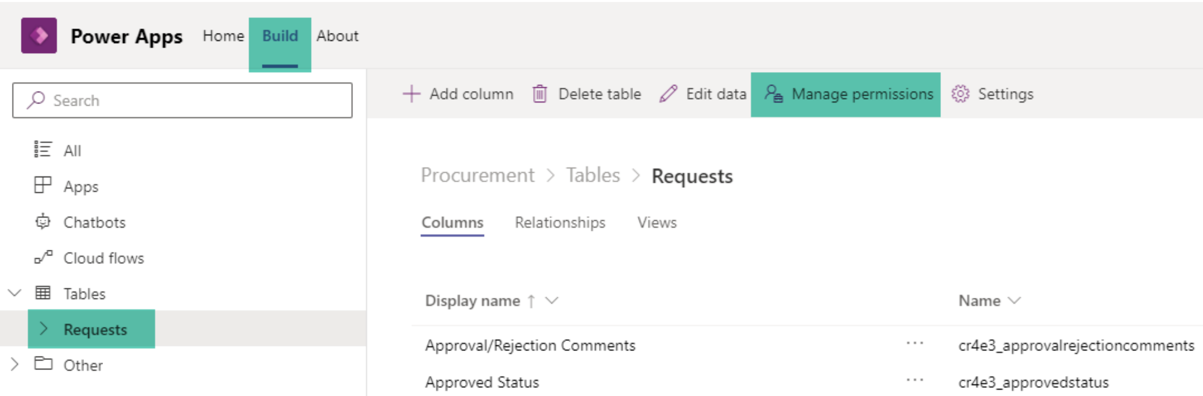


Sharing the App

By default any app that is created and shared with a Team will follow the security model defined for that Team. A team usually have 3 roles Owners, Members and Guests



The default permission levels for these Roles in the current Apps' Dataverse tables can be viewed by visiting the Build tab and selecting the table and clicking on Manage Permissions.

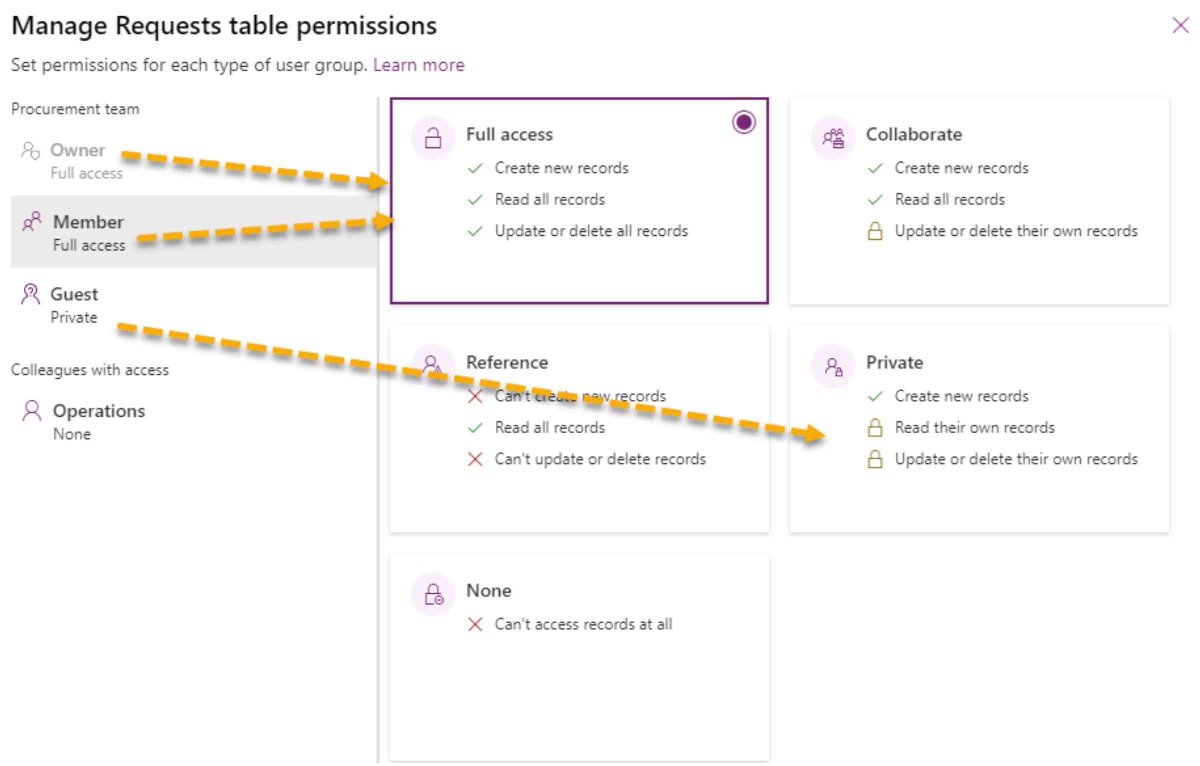


Here we can see the 4 permission levels defined for the Table:

- Full access – Allows Users to View and Edit all records in the table.
- Collaborate – Allows Users to View all records but edit their own records.
- Reference – Allows users to View all Users records.
- Private -– Allows users to only view and edit their own records.

By default, the Owner and Members have full access to the table which lets them do a full CRUD operation on the table. However, Guests can only Create new records and Read/Update/Delete their own records.

Using this Permission management pane, we can change the permissions for existing Roles in the Team, say for instance we can bring down the Members permission to Collaborate or Private so that they don't have the permission to update other's records.

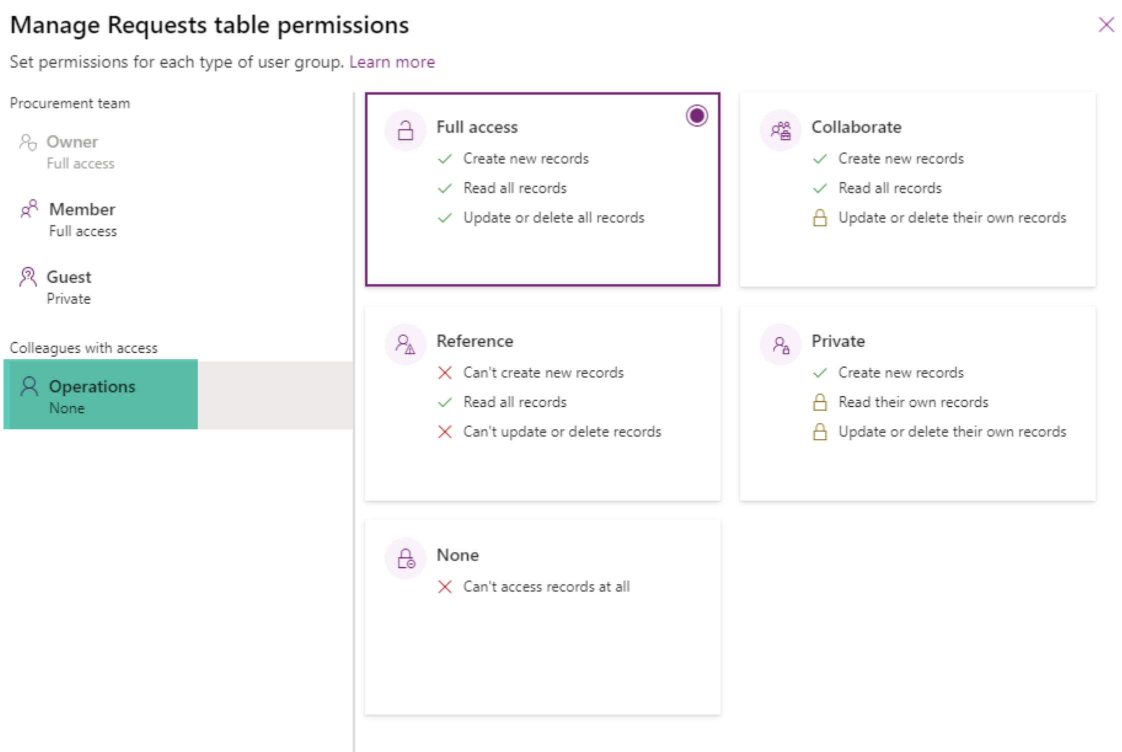


Set permissions for Users outside the Team

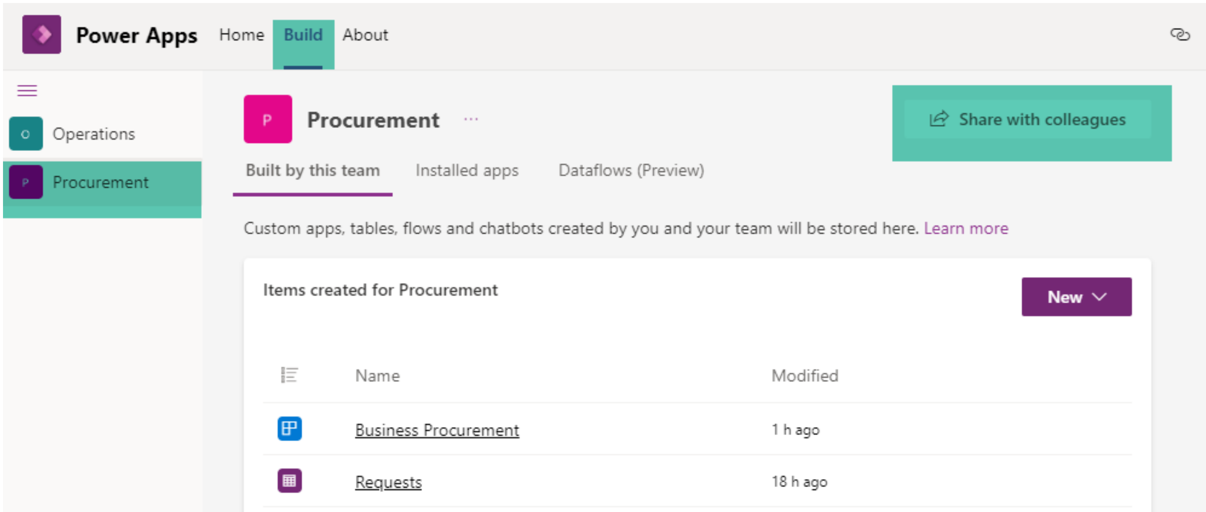
To grant permissions to users outside the team in which the app is shared, we have another section called the Colleagues with access which by default is set to None. We can Edit it and assign the required permissions. Since we are sharing the App and Table with Developers, we will assign Full access to Colleagues with access role.

Note: We are not specifying any Group/Colleagues in this page. We are just setting the permissions for those colleagues. Who are those colleagues, will be specified in the upcoming steps

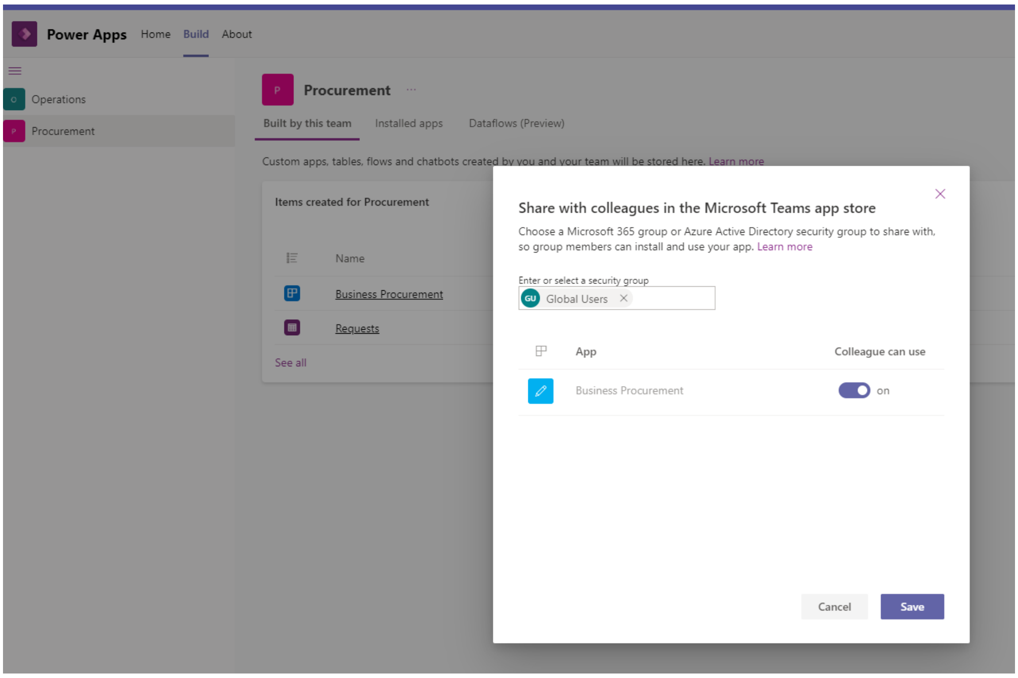
We will assign the full permission for Colleagues with access. Operations is the name for a previously entered group, we will see how to change that and assign it to Global Users.



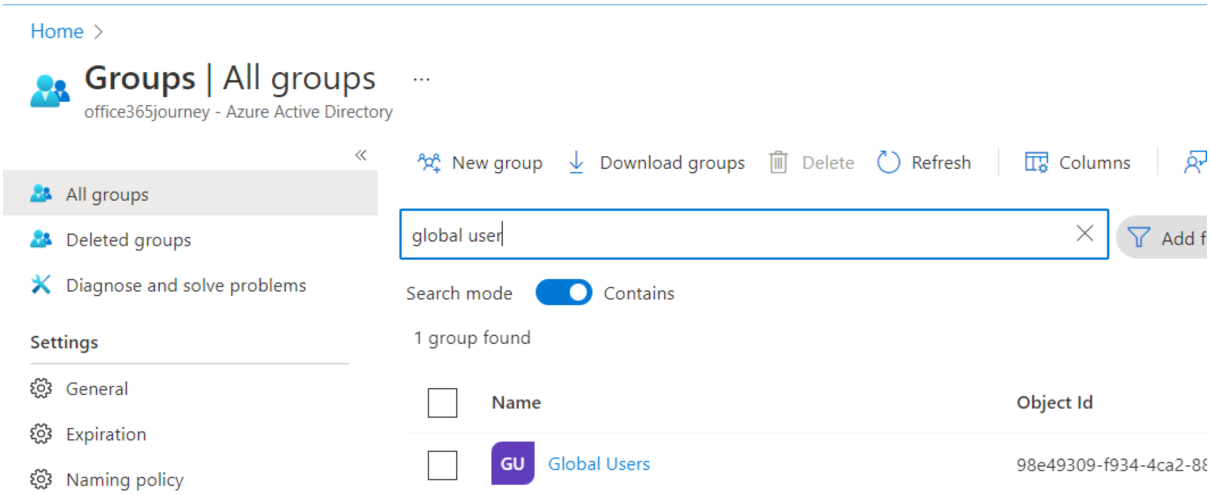
To assign the Colleagues who will have full access, Select the Build tab and click on the Teams that we had shared the App with. Here we can see an option to "Share with colleagues"



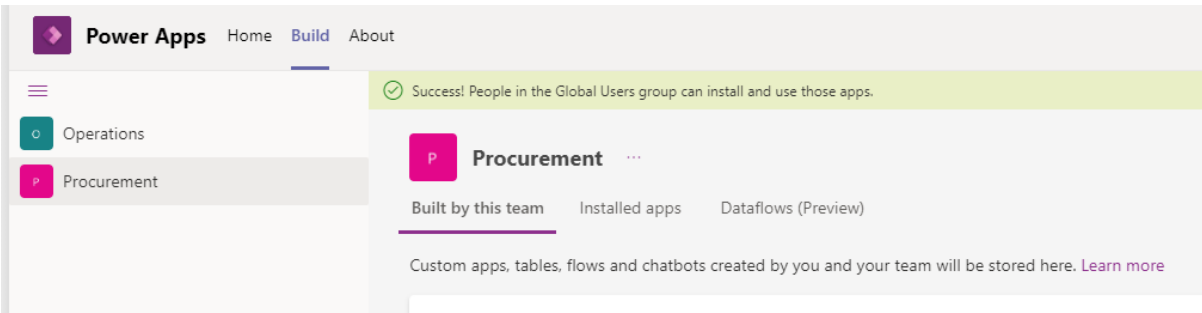
This will open a window where we can specify a Microsoft 365 group or an Azure AD security group and users in those groups will get access to the App that we are trying to share.



we have added the Global Users Azure AD Security Group so that they can access the app and table.

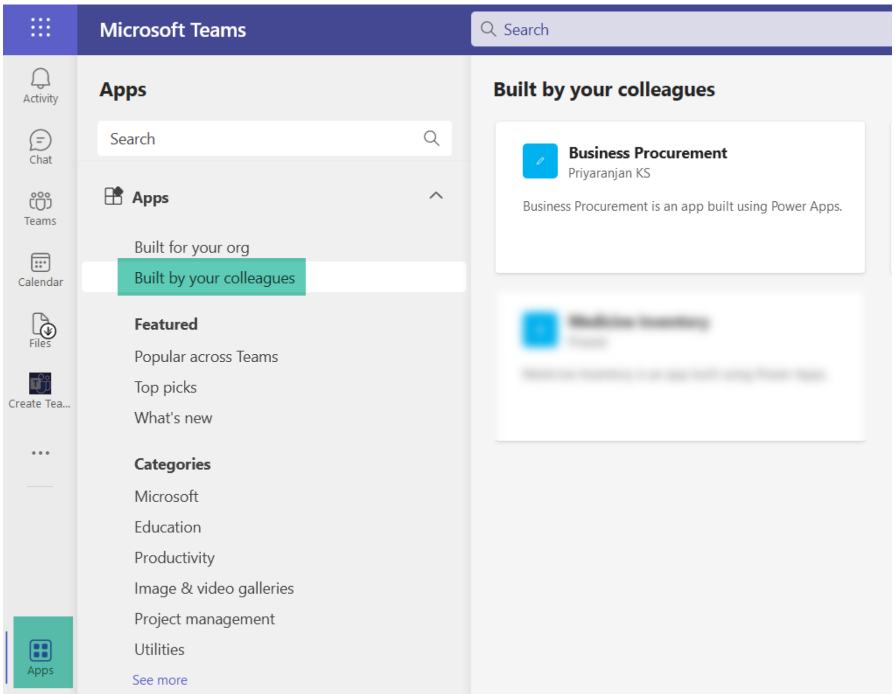


Thus, we have got a success message as well indicating the permissions has been granted.

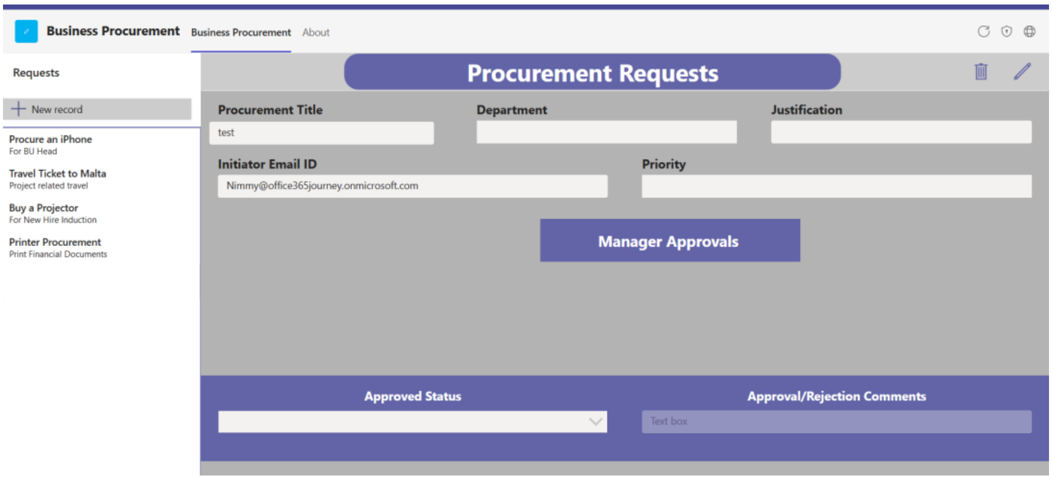


Test the Sharing of App

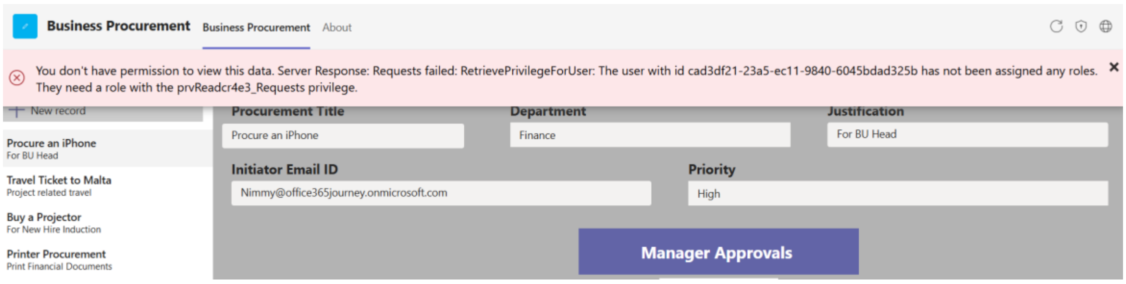
Now let’s test the sharing of the app, by logging in as one of the Developers who have been granted access. Logging in to teams, Clicking on Apps and Selecting “Build by your colleagues” will list the recent access granted app which we can add to the tab by selecting it.



We should now be able to use the App even outside the Teams with which it was shared initially.



Note: If the permissions were not granted correctly to the table, say for instance you want the delete a record and you have granted the Reference permission will throw the below error.



Summary

Thus, we saw how to share a Power App Created in Microsoft Teams to be shared with a Global audience outside the Teams.

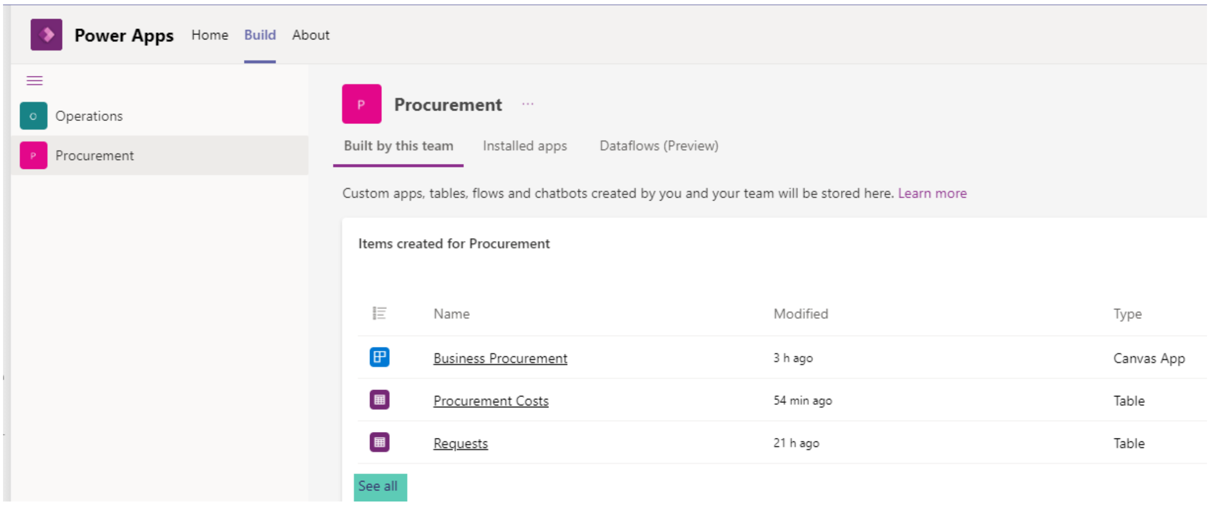
Build Power BI reports from Dataverse for Teams Table

Introduction

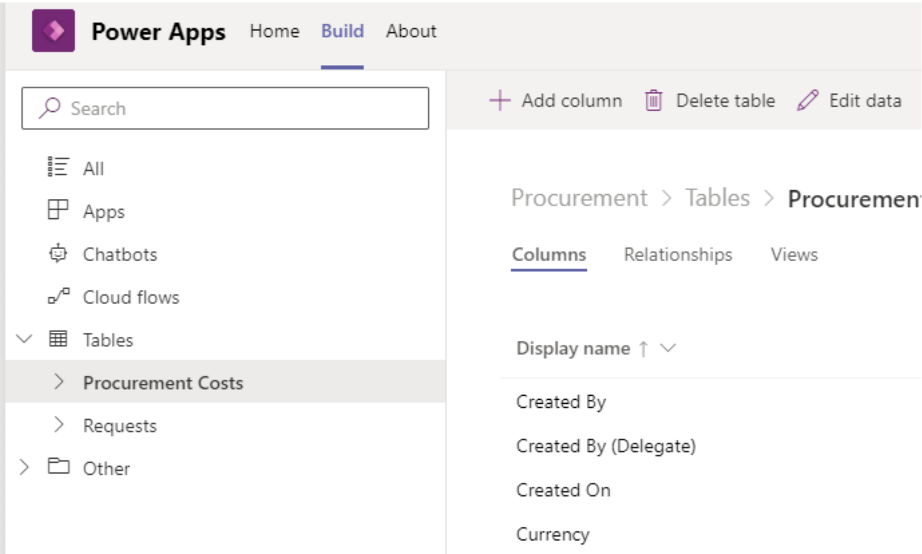
In this section we will see how to connect Power BI desktop to one of such Dataverse for teams table data to pull the data and build reports.

Dataverse Table for Teams

We can explore and view the Dataverse Table in Teams from the Power Apps personal app. Head over to Build and select See All which will list all the Tables and Apps.

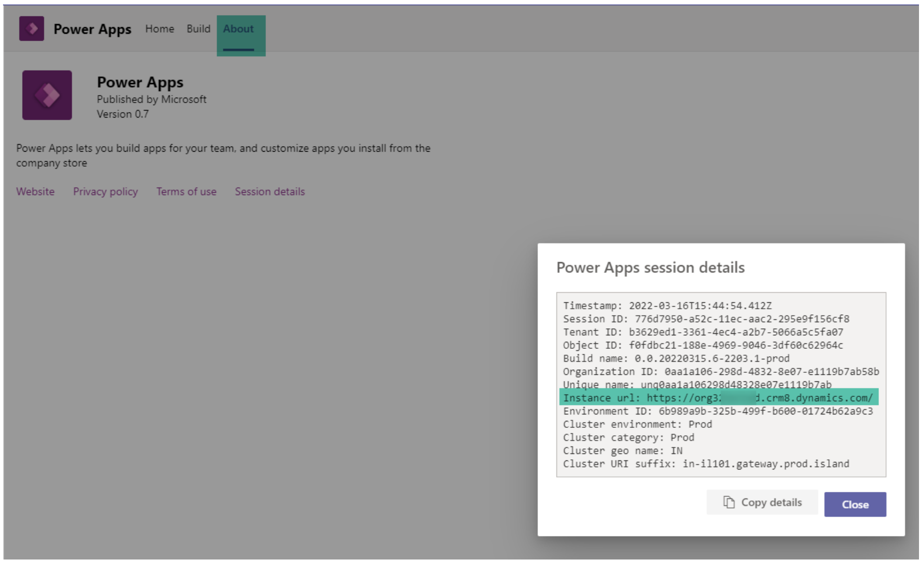


We can Edit and make changes to the Tables from here.



Fetch the Environment URL

To connect to this table, If we are using the Legacy Common Data Service connector,we will first have to get the URL for the Dataverse for Teams Environment which we can get from the About tab.

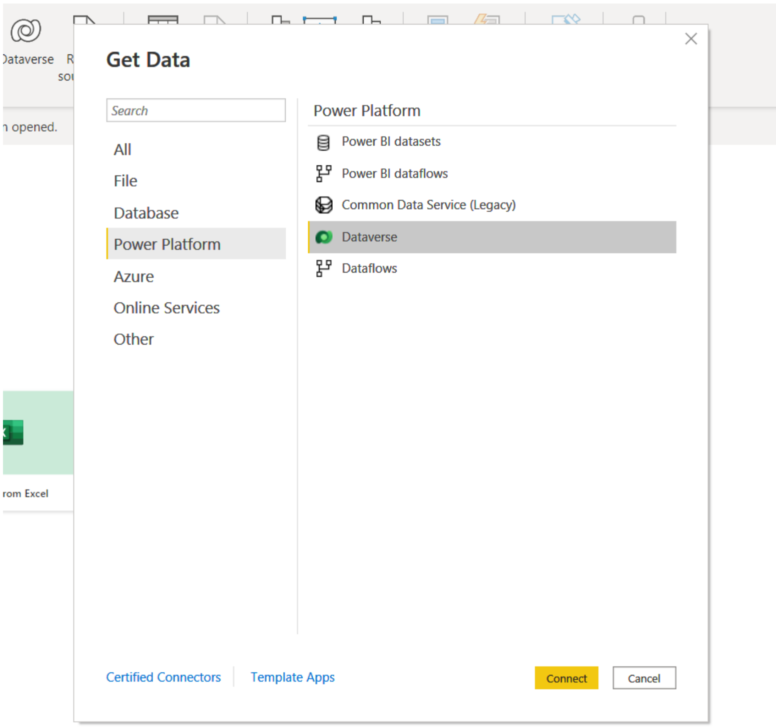


Copy the Instance URL which is the Dataverse for Teams environment URL.

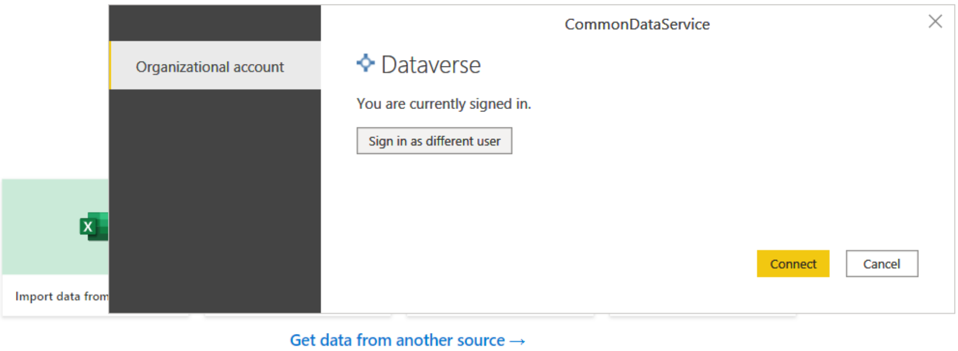
Note: If we are using the modern Dataverse connector, the environment URL is not needed as it connects to the Environment using Azure AD authentication and lists all the Dataverse tables available in the tenant.

Connect to Power BI

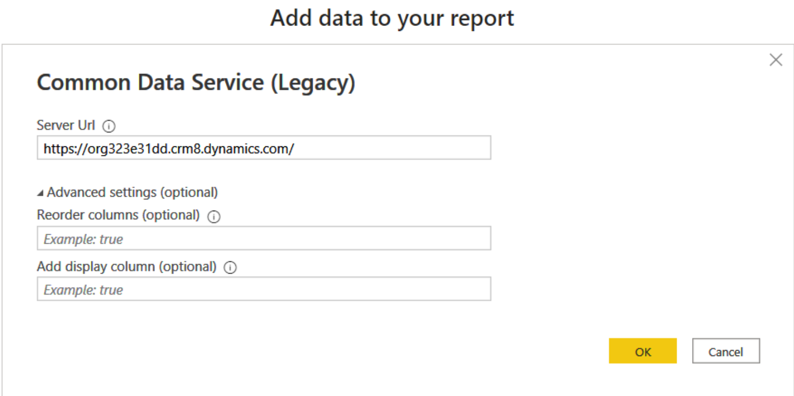
We will now use this URL to connect from Power BI. Open Power BI Desktop, Select File -> Get Data and Select Dataverse



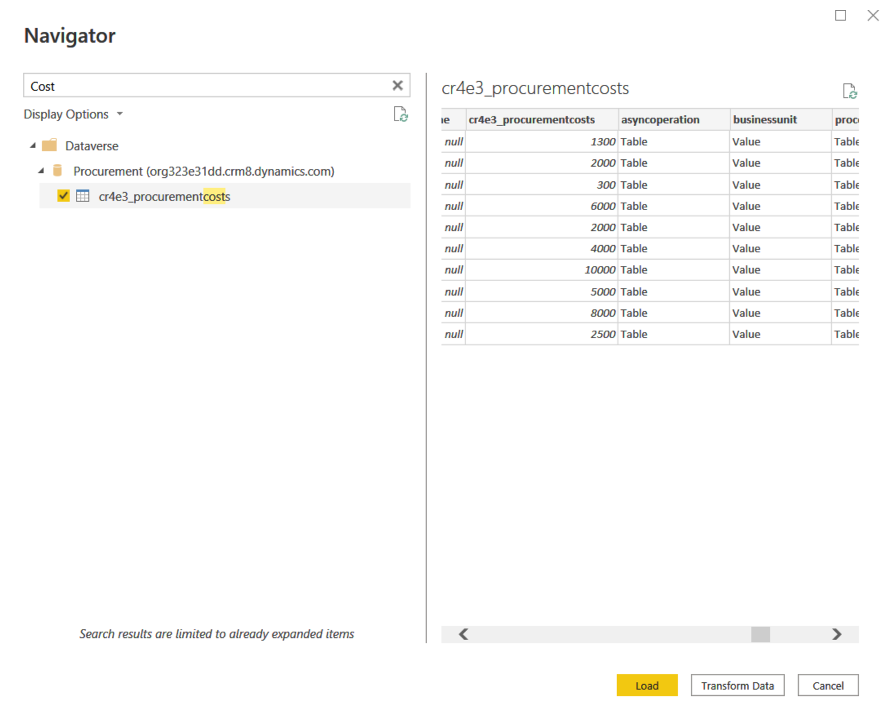
Sign in to connector using the Microsoft 365 account and click on Connect.



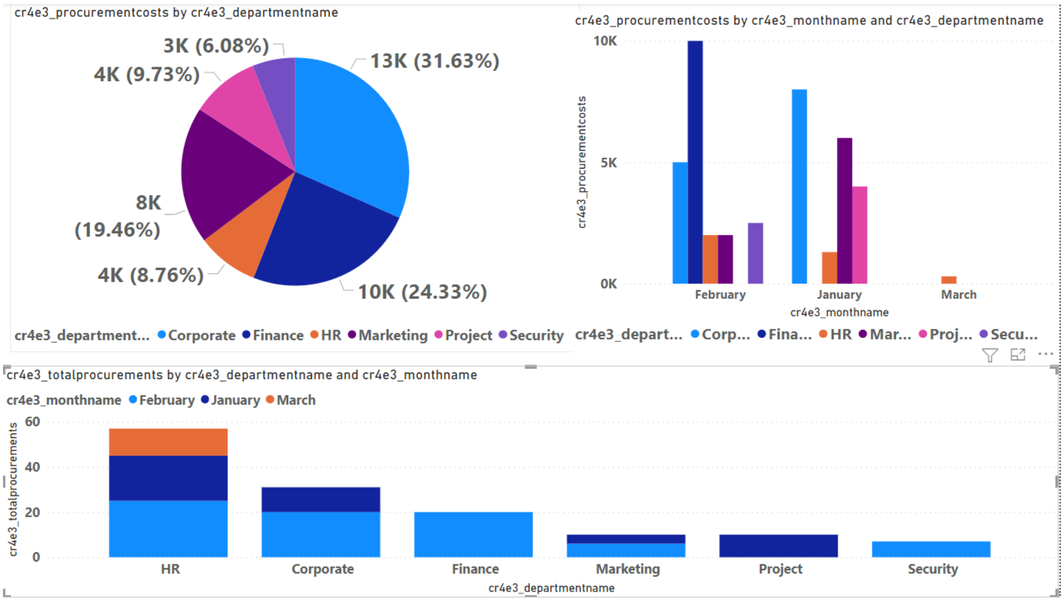
Note : In case if we are using the Legacy Common Data Service Connector, we can supply the environment URL that we fetched earlier to make the connection



Since we are using the Datverse connector, it seamlessly lists all the available dataverse tables for consumption. I have selected the Dataverse for teams table and will click on Load to import the data to Power BI workspace.



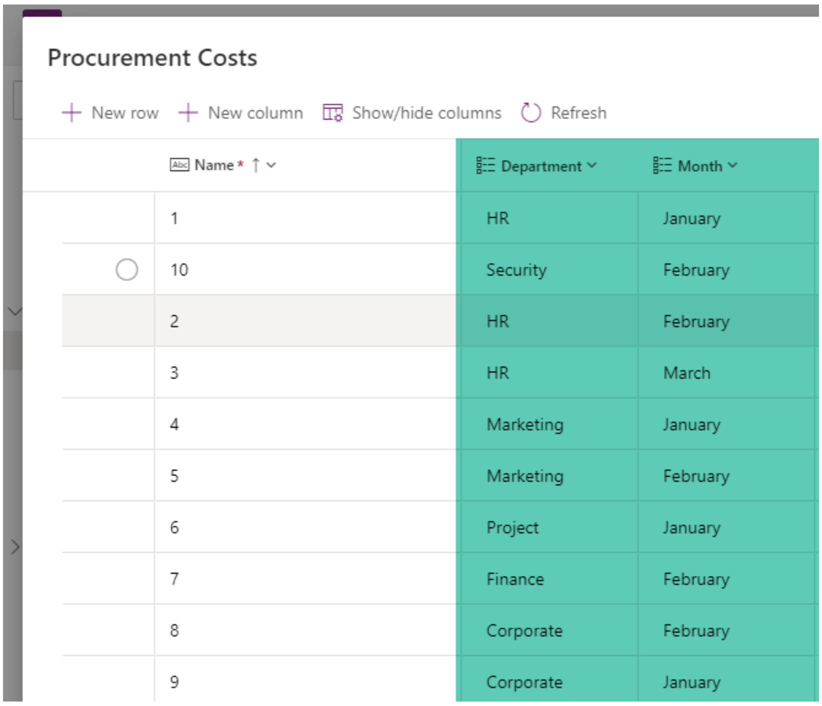
Once the data is inside Power BI, we can transform and make changes to the data and create stunning reports using the available Power BI Visualizations



Note: In case if you encounter issues with Choice Field from Dataverse table being empty in Power BI refer to the below section.

Possible Issue

However, while connecting to Dataverse for Teams tables, there can be a situation where your choice columns in the dataverse table does not get loaded as expected in Power BI. Say for instance I have two choice columns in the Dataverse table

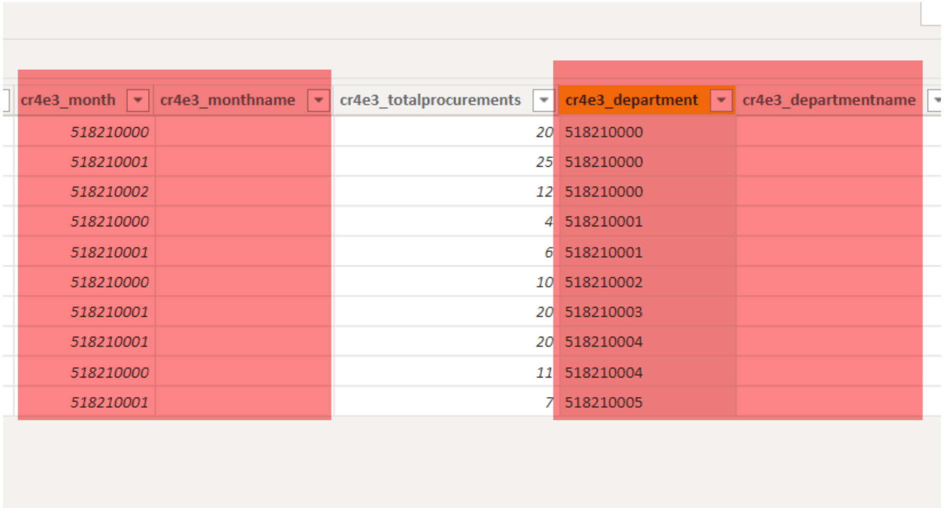


Procurement Costs

New row | New column | Show/hide columns | Refresh

Name *	Department	Month
1	HR	January
10	Security	February
2	HR	February
3	HR	March
4	Marketing	January
5	Marketing	February
6	Project	January
7	Finance	February
8	Corporate	February
9	Corporate	January

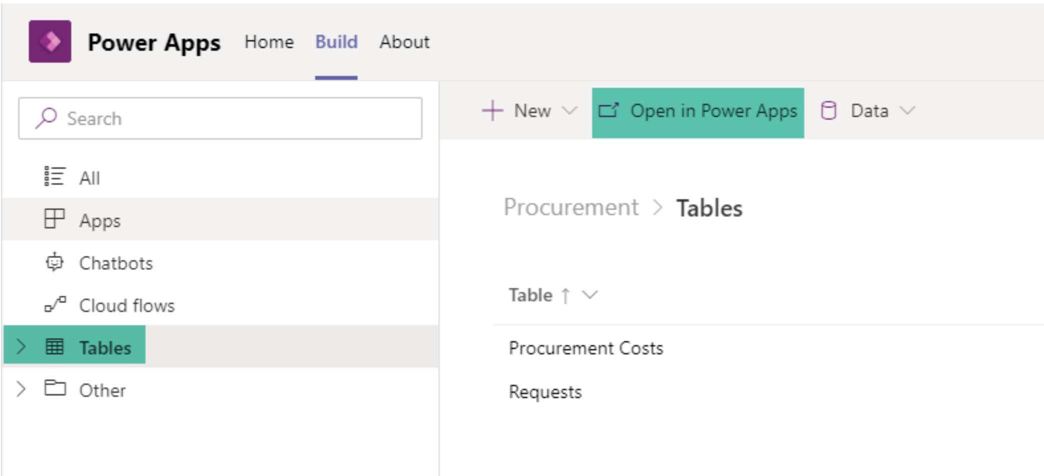
While importing to the Power BI, it creates a Choice Number column as well as a Choice Name column which is empty.



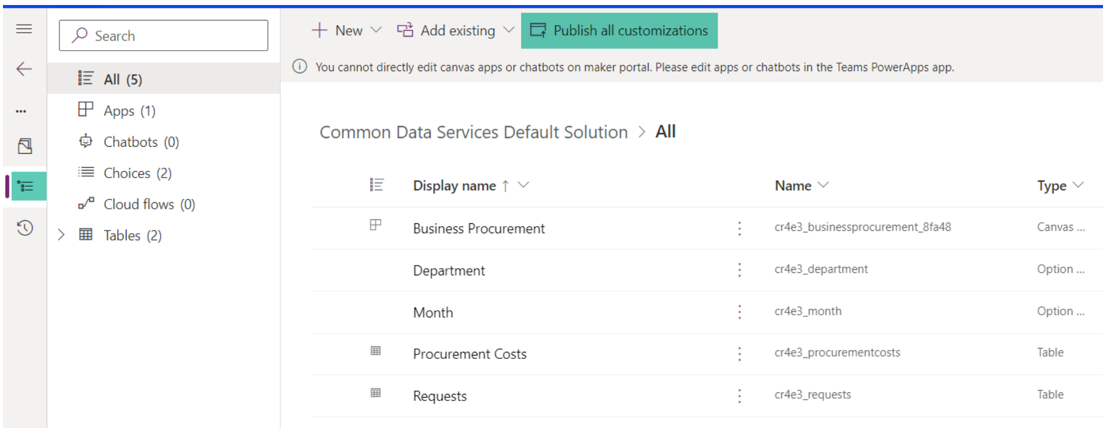
cr4e3_month	cr4e3_monthname	cr4e3_totalprocurements	cr4e3_department	cr4e3_departmentname
518210000		20	518210000	
518210001		25	518210000	
518210002		12	518210000	
518210000		4	518210001	
518210001		6	518210001	
518210000		10	518210002	
518210001		20	518210003	
518210001		20	518210004	
518210000		11	518210004	
518210001		7	518210005	

Solution

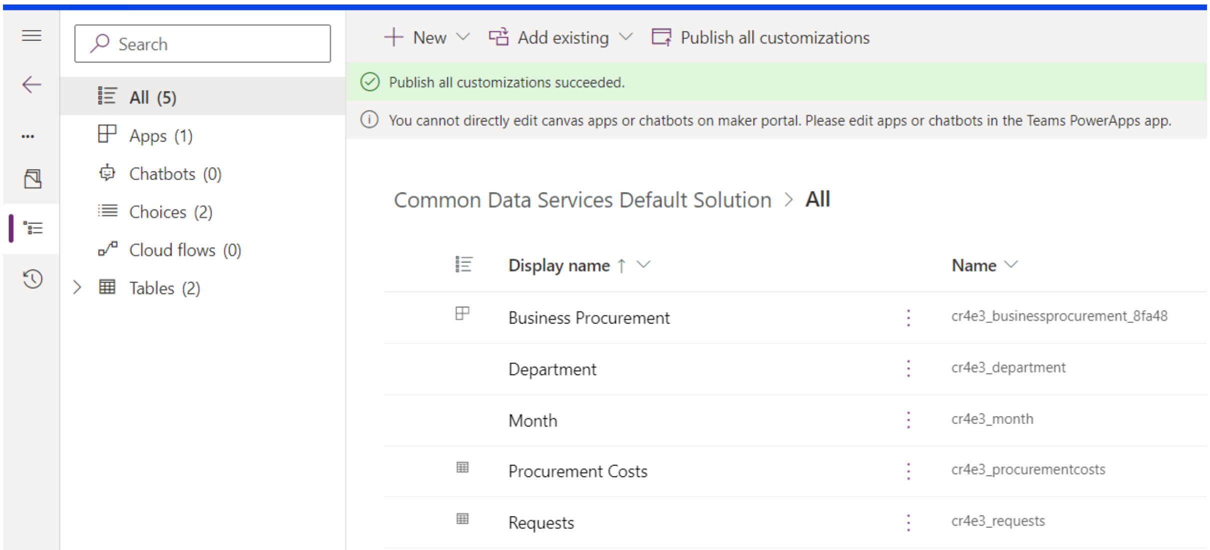
To dig deep into this issue and find a solution, lets head over to the Build tab of the Power Apps personal App in Microsoft Teams and select the table section. It lets you open it in Power Apps.



Here in the left pane, we can see the Objects tab. Select it and click on Publish all customizations in the top ribbon.



We can see that in a while, it has published the customizations.



Test the Data in Power BI

Re import the data to Power BI by connecting to Dataverse for Teams table and we can see that the Choice value texts are now populated without any issues

cr4e3_month	cr4e3_monthname	cr4e3_totalprocurements	cr4e3_department	cr4e3_departmentname
518210000	January	20	518210000	HR
518210001	February	25	518210000	HR
518210002	March	12	518210000	HR
518210000	January	4	518210001	Marketing
518210001	February	6	518210001	Marketing
518210000	January	10	518210002	Project
518210001	February	20	518210003	Finance
518210001	February	20	518210004	Corporate
518210000	January	11	518210004	Corporate
518210001	February	7	518210005	Security

Summary

Thus, we saw an issue related to Dataverse for Teams choice table column coming up as empty in Power BI and how to resolve this issue