

# Power Automate Bootcamp: Basic

Hands-On Lab Guide

Published (Jan 2025)



# Disclaimer

The content, code samples, and guidance provided in this lab document (the “Hands-On Lab Guide”) are for informational and educational purposes only. The Materials are provided “as is” and reflect general practices and platform capabilities at the time of publication. While reasonable efforts have been made to ensure accuracy, no representation or warranty, express or implied, is made as to the completeness, accuracy, reliability, or suitability of the Materials for any purpose.

The Materials are not intended to be used as-is in production environments. It is the sole responsibility of the user to evaluate and test all solutions, configurations, and code examples in a controlled, non-production environment prior to any implementation in a live setting. Use of these Materials in a production environment is done entirely at the user’s own risk.

No warranties or guarantees of any kind are provided, including but not limited to warranties of merchantability, fitness for a particular purpose, non-infringement, or the absence of errors or defects. Under no circumstances shall the authors, contributors, or any affiliated parties be held liable for any damages, including but not limited to direct, indirect, incidental, consequential, or special damages, arising out of or in connection with the use of the Materials.

By using or adapting any portion of the lab document in any capacity, including in production environments, you expressly acknowledge and agree that you are assuming full responsibility and liability for such use, and that you release the authors and affiliated parties from any and all claims or liabilities that may arise as a result.

## Contents

Disclaimer .....	2
Exercise 1: Experimenting Advanced Power Automate Settings .....	1
Objectives .....	1
Estimated Time .....	1
Task 1: Uploading Dummy Excel File to OneDrive .....	1
Task 2: Creating a Manually Triggered Flow .....	2
Task 3: Pagination and Threshold settings .....	3
Task 4: Concurrency Settings .....	6
Task 5: Run after Settings .....	9
Exercise 2: Create Approval Workflow to Run from Power Apps .....	13
Objectives .....	13
Estimated Time .....	13
Task 1: Creating Environment Variables .....	14
Task 2: Creating a New Flow to Run from Power App .....	18
Task 3: Handling Approval Timeout .....	23
Task 4: Adding Flow to App .....	26
Optional Task: Use HTML Table Instead of Multiple Messages .....	<b>Error!</b>
<b>Bookmark not defined.</b>	
Exercise 3: Deep Linking for Direct Access .....	29
Objectives .....	29
Estimated Time .....	29
Task 1: Duplicate the existing Screen .....	30
Task 2: Handle Deep Link ID in App .....	32
Task 2: Include link in Power Automate Approval .....	35
Exercise 4: Utilising Adaptive Card and Child Flow .....	36
Objectives .....	36
Estimated Time .....	36
Task 1: Change Response Options in Approval Workflow .....	37
Task 2: Design Adaptive card to Get User Input .....	40
Task 3: Create a Child Flow to Get More Information .....	44
Task 4: Add Child Flow to the Main Flow .....	49

# Exercise 1: Experimenting Advanced Power Automate Settings

In this hands-on exercise, participants will create an **instant cloud flow** to deal with large excel data and play with the stage settings to be able to deal with it.

## Objectives

After completing this exercise, participants will be able to:

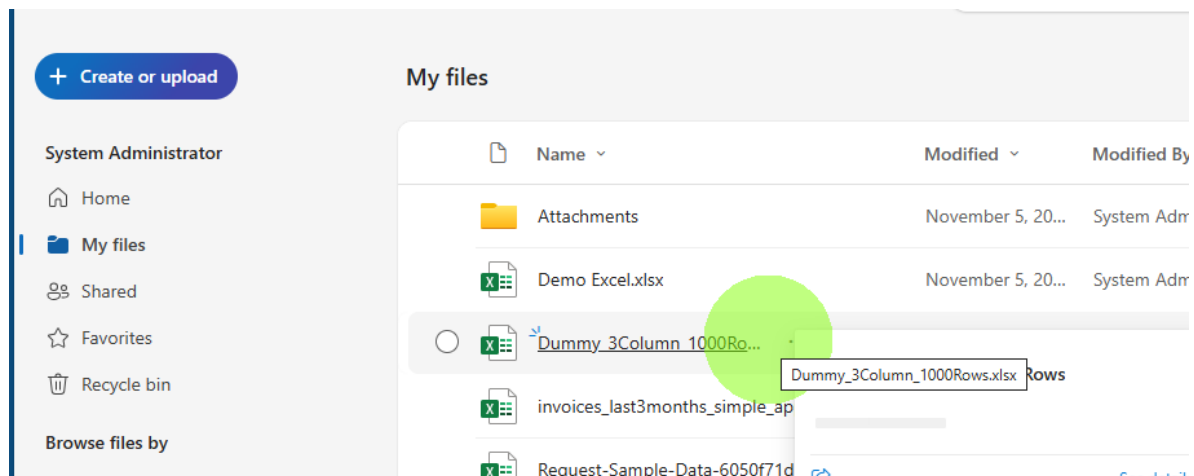
- Have a better understanding of Advanced Power Automate stage settings
- Perform Error Handling in Power Automate

## Estimated Time

40–50 mins

## Task 1: Uploading Dummy Excel File to OneDrive

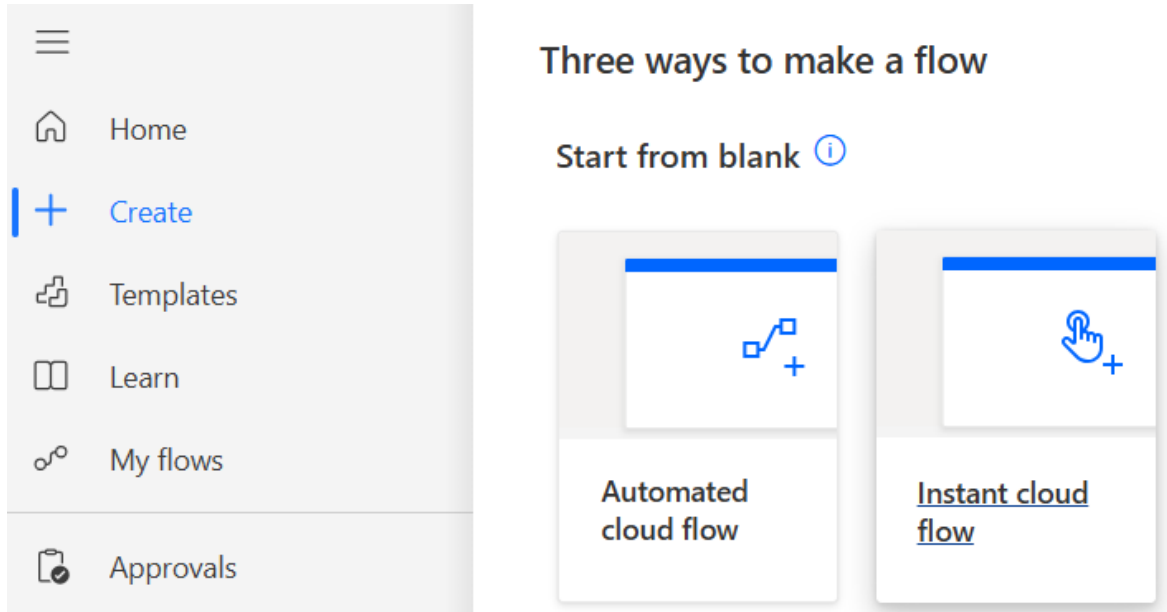
1. Go to **OneDrive for Business** and upload Dummy\_3Column\_1000Rows.xlsx file.



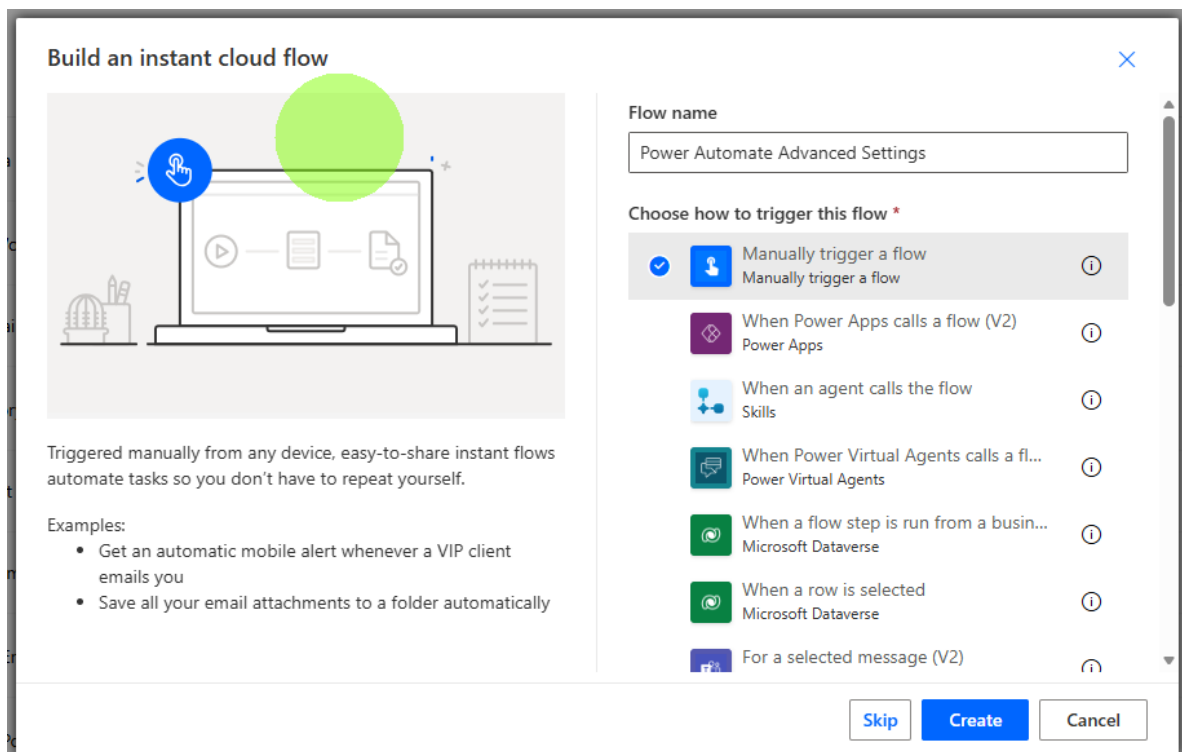
----- End of Task -----

## Task 2: Creating a Manually Triggered Flow

1. Go to Power Automate and click **Create > Instant cloud flow**



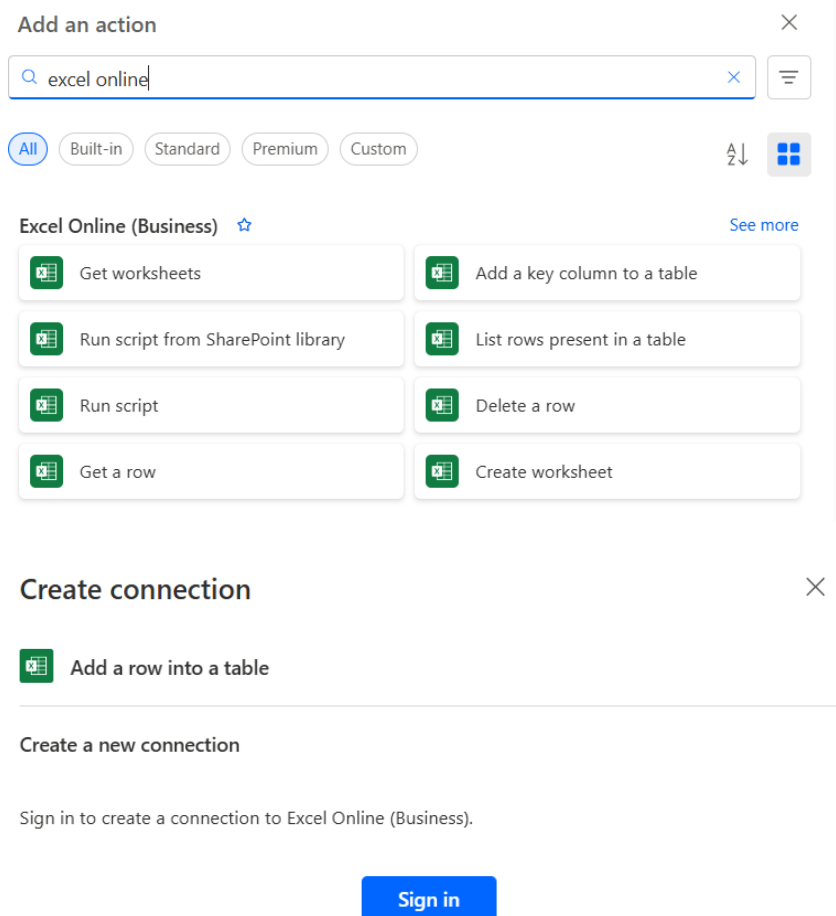
2. Name your flow: **Power Automate Advanced Settings**  
Choose the trigger: **Manually trigger a flow**, then click **Create**



----- End of Task -----

## Task 3: Pagination and Threshold settings

1. Click + **New Step**, search for **Excel Online (Business)**, and choose **List rows present in a table** (Note: Sign in and create connection if it is the first time)



**Add an action**

Search: excel online

Filters: All Built-in Standard Premium Custom

**Excel Online (Business)** [See more](#)

- Get worksheets
- Run script from SharePoint library
- Run script
- Get a row
- Add a key column to a table
- List rows present in a table
- Delete a row
- Create worksheet

**Create connection**

[Add a row into a table](#)

**Create a new connection**

Sign in to create a connection to Excel Online (Business).

[Sign in](#)

2. Select your OneDrive Excel file and the table name as shown below

**List rows present in a table**

Parameters Settings Code view Testing About

Location \*  
OneDrive for Business

Document Library \*  
OneDrive

File \*  
/Dummy\_3Column\_1000Rows.xlsx

Table \*  
Table1

3. Add a compose stage with below formula to observe the number of rows read.

```
length(outputs('List_rows_present_in_a_table')?['body/value'])
```

**Compose**

Parameters Settings Code view About

Inputs \*  
Inputs

length(outputs('List\_rows\_present\_in\_a\_table')?['body/value'])

4. You can save and test the flow and notice in Compose stage output that it doesn't read all the rows from the file. That's due to Power Automate default settings to retrieve 256 rows.


**OUTPUTS** Show raw outputs >

Outputs

256

5. Select List rows present in a table stage and switch to Settings tab. Turn on Pagination and set Threshold to 5000.



 List rows present in a table ⋮ <<

Parameters **Settings** Code view Testing About

▼ General

Action timeout ⓘ  
Specify the duration in ISO-8601 format

▼ Networking

Pagination ⓘ  
Retrieve items to meet the specified threshold by following the continuation token. Due to connector's page size, the number returned may exceed the threshold.  
☒ On  
Threshold

6. You can run again and will notice that it is now returning full list.

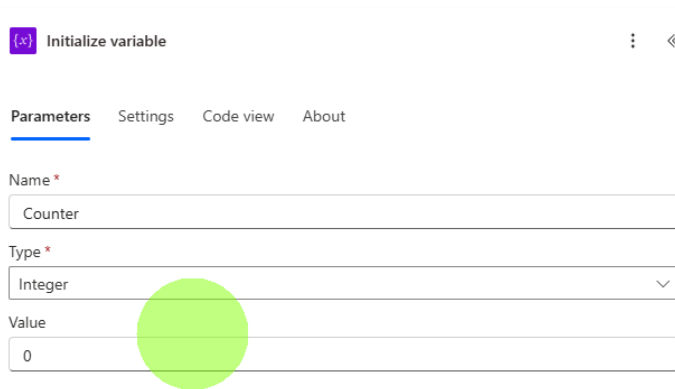
**OUTPUTS** Show raw outputs >

Outputs  
 

----- **End of Task** -----

## Task 4: Concurrency Settings

1. Add a new stage Initialize Variable and initialize a variable name Counter with integer as data type and 0 as value.



Initialize variable

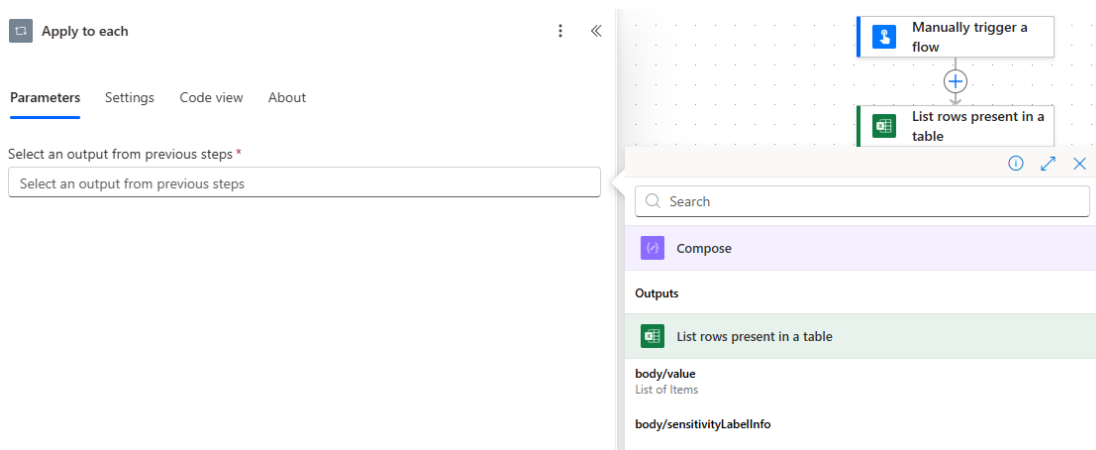
Parameters Settings Code view About

Name \*  
Counter

Type \*  
Integer

Value  
0

2. Add a new stage Apply to each and choose body/value output from List rows present in a table.



Apply to each

Parameters Settings Code view About

Select an output from previous steps \*

Select an output from previous steps

Search

Compose

Outputs

List rows present in a table

body/value  
List of Items

body/sensitivityLabelInfo

3. Add a new stage Increment variable inside Apply to each. Select Counter and set increment value to 1.

**Increment variable**

Parameters Settings Code view About

Name \*

Counter

Value

1

4. Add a compose stage after Apply to each with below formula to observe the final variable value.

**Compose 1**

Parameters Settings Code view About

Inputs \*

Counter

Inputs

fx

5. You can save and test the flow and notice in Compose 1 stage output that the final incremented value is 1000. And will notice that the flow runs for a long time. That's because Apply to each runs sequentially by default

**OUTPUTS**

Show raw outputs >

Outputs

1,000

6. Select Apply to each stage and switch to Settings tab. Turn on Concurrency and set degree of parallelism to 50.

Apply to each



Parameters **Settings** Code view About

General

Concurrency control ⓘ

For each loops execute sequentially by default. Override the default setting to customize the degree of parallelism.

Limit  
☒ On

Degree of parallelism

50

7. You can run again and will notice that the flow runs faster and incremented value is still 1000.

**OUTPUTS** [Show raw outputs >](#)

Outputs

1,000

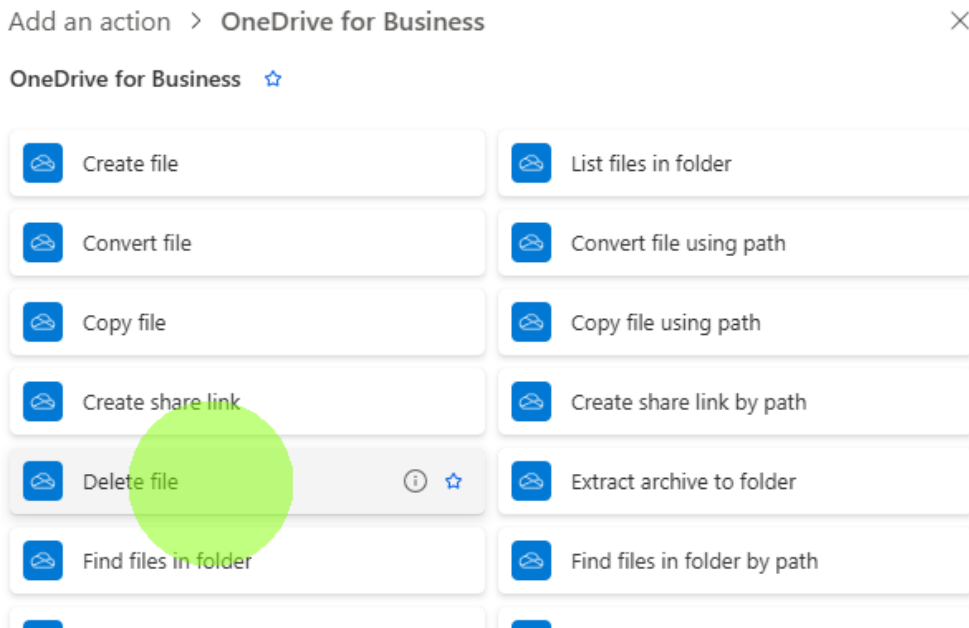
28-day run history ⓘ [Edit columns](#) [All runs](#)

Start	Duration	Status
Feb 4, 08:06 PM (2 min ago)	00:00:44	Test succeeded
Feb 4, 07:58 PM (10 min ago)	00:02:56	Test succeeded

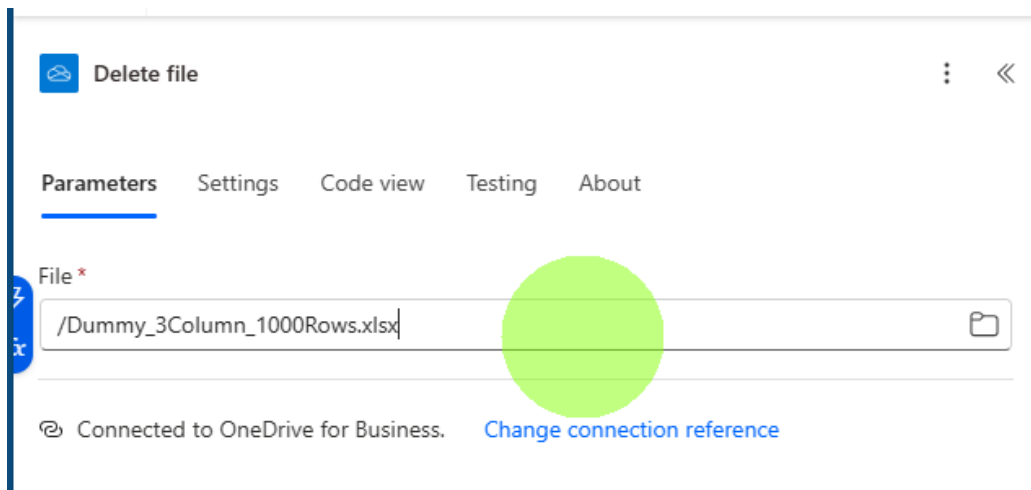
----- End of Task -----

## Task 5: Run after Settings

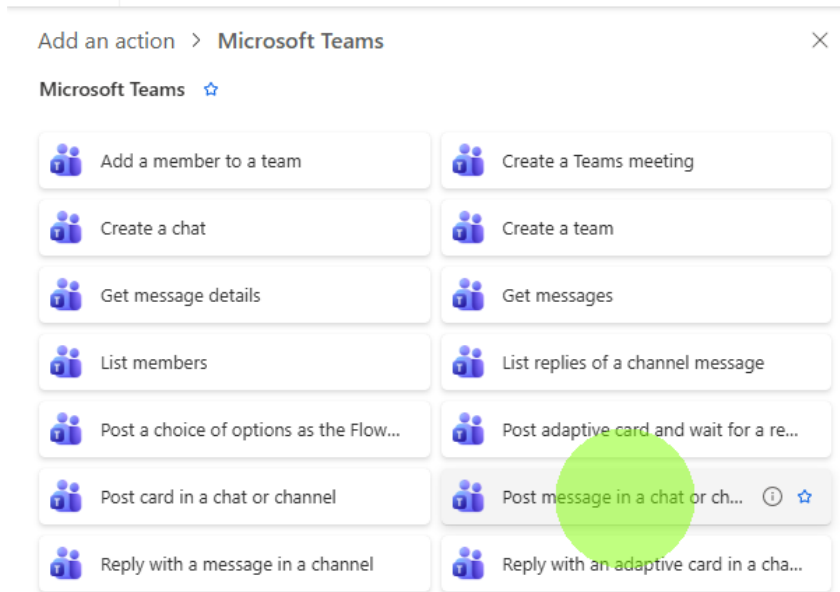
1. Add a new step with OneDrive for Business. Select Delete File.



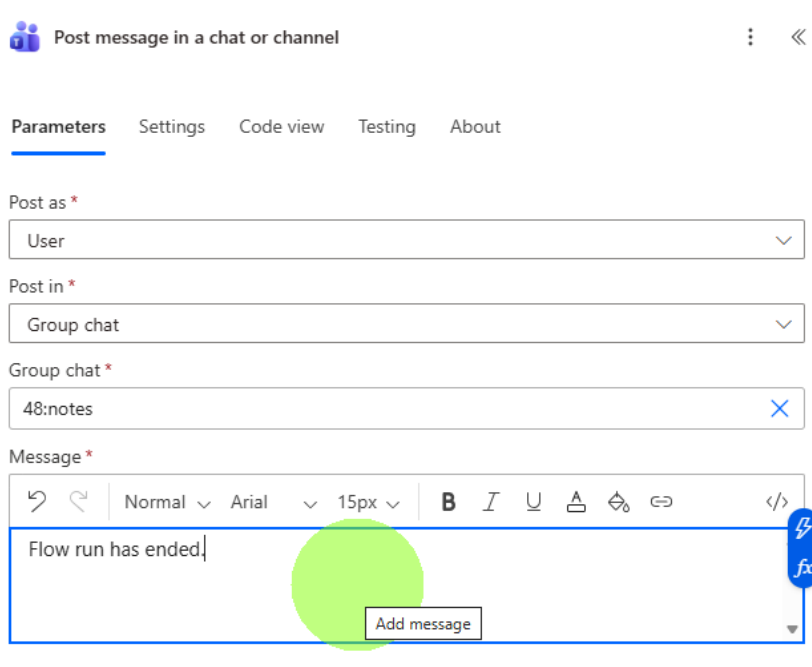
2. Select the excel file uploaded.



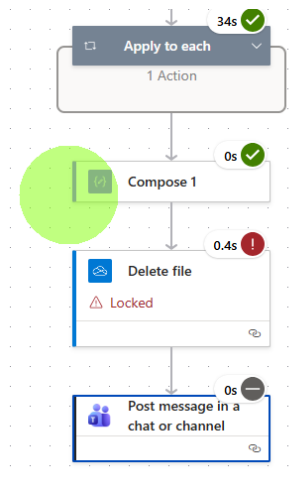
3. Add a new stage Post Message in a Chat or Channel under Teams.



4. Set the parameters as below.



5. Save and run the flow. You will notice Delete file fails as the file is locked for reading and the step Post message in a chat or channel is skipped as the step above failed.



- Select Post message in a chat or channel stage and switch to Settings tab. Check Has failed under Run after settings to make sure that the stage still runs even if the Delete file fails.

Post message in a chat or channel

Parameters Settings Code view Testing About

General

Action timeout ⓘ  
Specify the duration in ISO-8601 format  
Example: P1D

Networking

Retry policy ⓘ  
A retry policy applies to intermittent failures, characterized as HTTP status codes 408, 429, and 5xx, in addition to any connectivity exceptions. The default is an exponential interval policy and the number of retries is based on your performance profile. [Learn more](#)  
Default

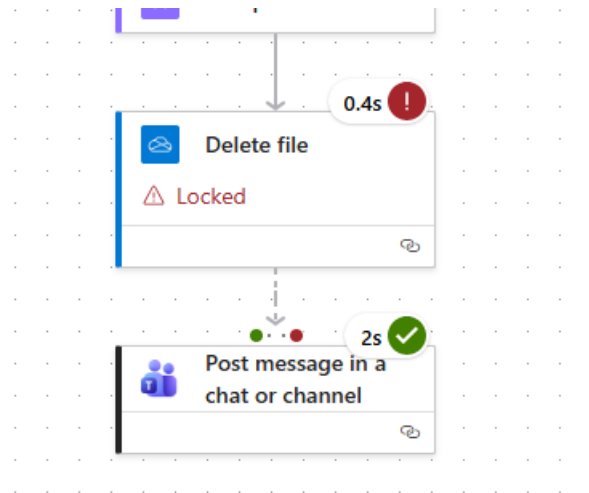
Run after

+ Select actions

Delete file

- ☒ Is successful
- ☐ Has timed out
- ☐ Is skipped
- ☒ Has failed

- Save and run the flow again. You will notice the stage now runs regardless of Delete file status.



----- End of Task -----



# Exercise 2: Create Approval Workflow to Run from Power Apps

In this exercise, you'll create an Approval Workflow to be called from the Finance Invoice App created in Day 1 lab guide.

## Objectives

After completing this exercise, participants will be able to:

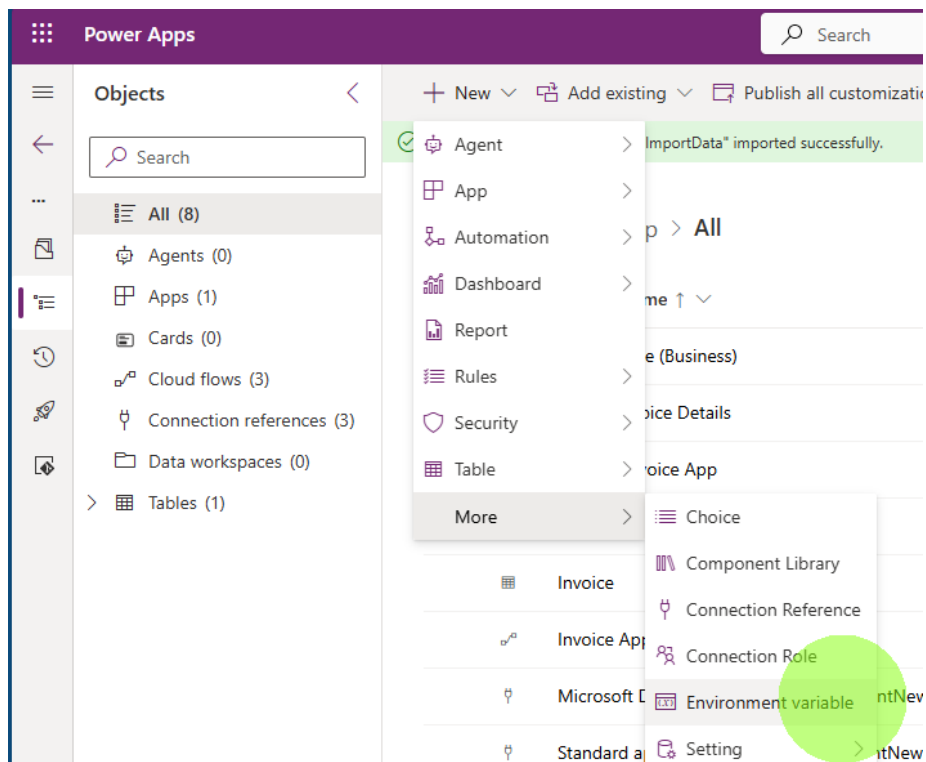
- Create workflows that integrate with Power Apps with input parameters
- Utilize environment variables in their solutions
- Send out approval request using Power Automate

## Estimated Time

30–40 mins

## Task 1: Creating Environment Variables

1. Go Invoice Processing App solution and a new Environment Variable in the solution.



2. Call the variable name as ApproverEmail and set your own email address as the value.

### New environment variable ✕

Environment variables can have different values when re-used, enter information about this variable so that future users can understand its purpose. [Learn more](#)

**Display name \***

**Name \* ⓘ**

**Description**

**Data Type \***

Text ▼

**Default Value ⓘ**

**Current Value**

Override the default value by setting the current value for your environment.

**Advanced ▼**

Save

Cancel

### 3. Get Canvas App URL from details page.

InvoiceProcessingApp > All

	Display name ↑		Name ↓
	ApproverEmail		cr61e_ApproverEmail
	Excel Online (Business)		new_sharedexcelonlinebusiness_62f...
	Extract Invoice Details		Extract Invoice Details
✓	Finance Invoice App		cr61e_financeinvoiceapp_2ee86
	Import Invoice Data	Edit	
	Invoice	Play	
	Invoice Approval Workflow	Live monitor	
	Microsoft Dataverse InventoryManagmentNew...	Details	
		Share	

Finance Invoice App

Details Versions Connections Flows Analytics (preview)

Owner

System Administrator

Description

Not provided

Created

18/01/2026, 6:32:18 pm

Modified

04/02/2026, 7:21:11 pm

Web link

https://apps.powerapps.com/play/e/035deb13-fd0f-e924-96c2-5542c8a33489/a/16c46905-a5df-44d9-87b2-13b6f56331ba?tenantId=444bd310-9e85-476b-bb08-40c5cd1a0fbc&hint=a22eb0bb-5f0f-40d9-b451-4da0cf2a2958&sourceTime=1770209217461

...

Copy link to clipboard

4. And create a new environment variable called AppURL.

### New environment variable ✕

Environment variables can have different values when re-used, enter information about this variable so that future users can understand its purpose. [Learn more](#)

**Display name \***

**Name \*** ⓘ

cr61e\_ AppURL

**Description**

**Data Type \***

Text ▾

**Default Value** ⓘ

**Current Value**

Override the default value by setting the current value for your environment.

**Advanced** ▾

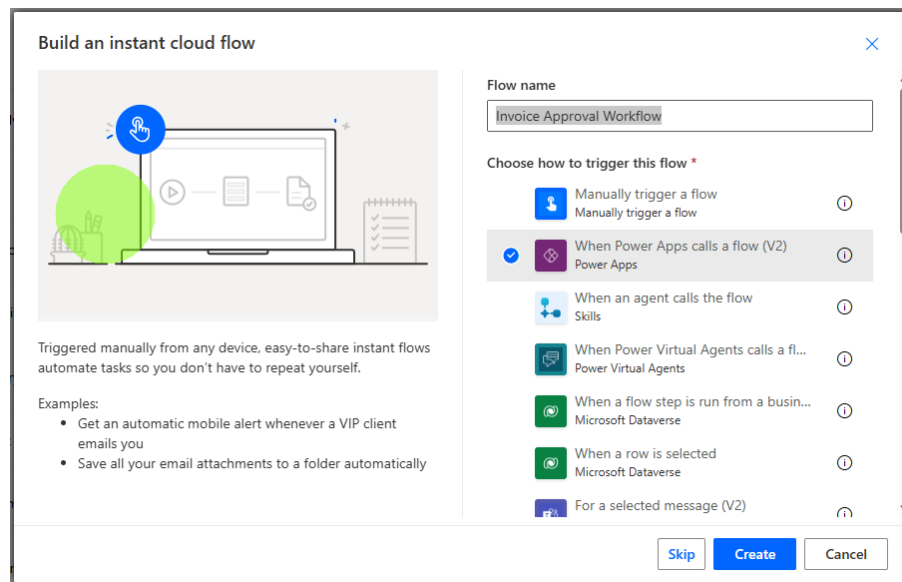
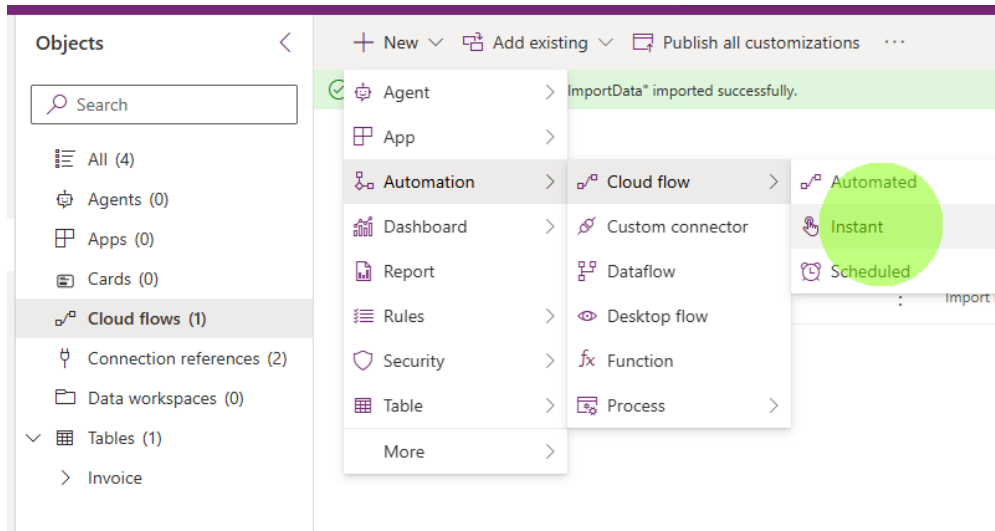
Save

Cancel


----- End of Task -----

## Task 2: Creating a New Flow to Run from Power App

1. Go Invoice Processing App solution and create a new **instant cloud flow**, Name it: **Invoice Approval Workflow**  
Choose **When Power Apps calls a flow (V2)**, then click **Create**



5. Create input parameters to the trigger as below.








 When Power Apps calls a flow (V2) ⋮ ⏪

Parameters

Settings

Code view

About

	VendorName	Please enter your input	...
	InvoiceId	Please enter your input	...
	Amount	Please enter a number	...
	InvoiceDate	Please enter or select a date (YYYY-MM-DD)	...
	RaisedBy	Please enter your input	...
	RaisedByEmail	Enter an email address.	...
	RowID	Please enter your input	...

+ Add an input

6. Add a step **Update a row** from the **Dataverse** connector. And set parameters as below.

Update Approval Status - Pending

⋮ ⏪

Parameters

Settings

Code view

Testing

About

Table name \*

Invoice

Row ID \*

RowID

Advanced parameters

Showing 2 of 14

Show all

Clear all

ApprovalRaisedDate

utcNow()

×


ApprovalStatus

Pending

×




7. Add a step: **Start and Wait for an Approval** from the **Approval** connector. And set parameters as below.

 **Start and wait for an approval** ⋮

**Parameters** Settings Code view Testing About

Approval type \*

Title \*

Assigned to \* 

Details

Vendor Name:

Invoice ID:

Invoice Amount:


Invoice Date:

Raised By:

Item link





Item link description

8. Add a Compose stage with the formula below to determine the Approval Outcome.  
`if(equals(outputs('Start_and_wait_for_an_approval')?['body/outcome'], 'Approve'), 'Approved', 'Rejected')`

 **Compose Outcome** ⋮ ⏪


**Parameters** Settings Code view About

Inputs \*

```
if(equals(outputs('Start_and_wait_for_an_approval')?['body/outcome'], 'Approve'), 'Approved', 'Rejected')
```

9. Add a step **Update a row** from the **Dataverse** connector. And set parameters as below.


 Update Approval Status ⋮ ⏪

**Parameters** Settings Code view Testing About

Table name <sup>\*</sup>  

Invoice ⌵


Row ID <sup>\*</sup>  

 RowID ×


Advanced parameters  

Showing 2 of 14 ⌵ Show all Clear all

ApprovalReceivedDate  

 Completion date × ×

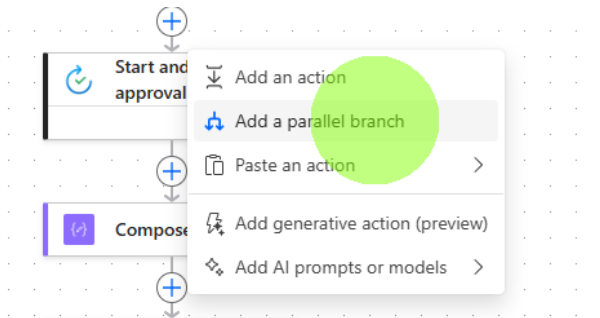
ApprovalStatus  

 Body × ×

----- End of Task -----

## Task 3: Handling Approval Timeout

1. Add a parallel branch between Approval and Compose Step.



2. Add a step **Update a row** from the **Dataverse** connector. And set parameters as below.

**Update Approval Status - Exception**

Parameters Settings Code view Testing About

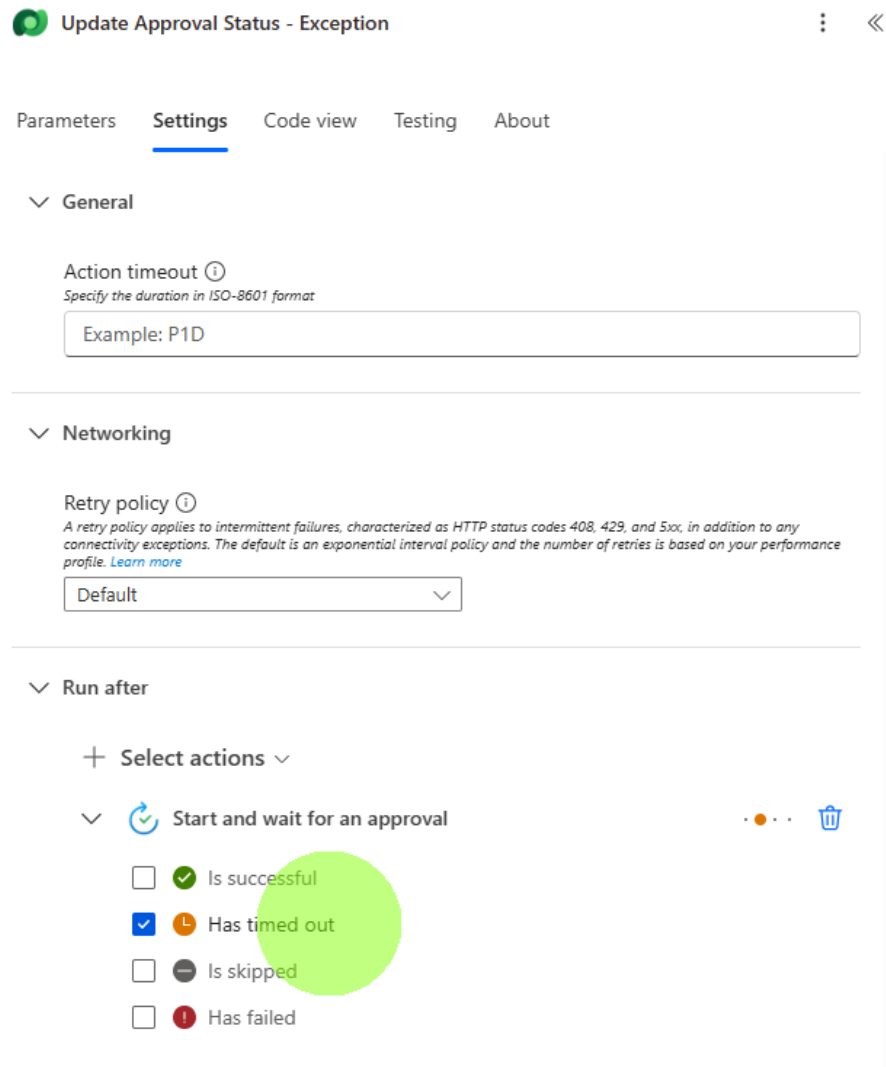
Table name \*  
Invoice

Row ID \*  
RowID

Advanced parameters  
Showing 1 of 14 Show all Clear all

ApprovalStatus  
Exception

3. Open Settings and set Run after to Has timed out. So this step will only run when Approval stage time outs (i.e. 30 days by default)



Update Approval Status - Exception

Parameters **Settings** Code view Testing About

General

Action timeout ⓘ  
Specify the duration in ISO-8601 format

Example: P1D

Networking

Retry policy ⓘ  
A retry policy applies to intermittent failures, characterized as HTTP status codes 408, 429, and 5xx, in addition to any connectivity exceptions. The default is an exponential interval policy and the number of retries is based on your performance profile. [Learn more](#)

Default

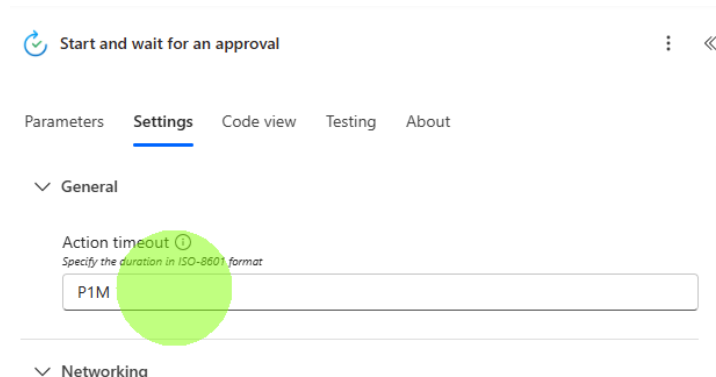
Run after

+ Select actions

Start and wait for an approval

- ☐ Is successful
- ☒ Has timed out
- ☐ Is skipped
- ☐ Has failed

4. For the purpose of testing in lab, we will change Action timeout of Approval stage to 1 minute (P1M in ISO-8601 format) as below.



Start and wait for an approval

Parameters **Settings** Code view Testing About

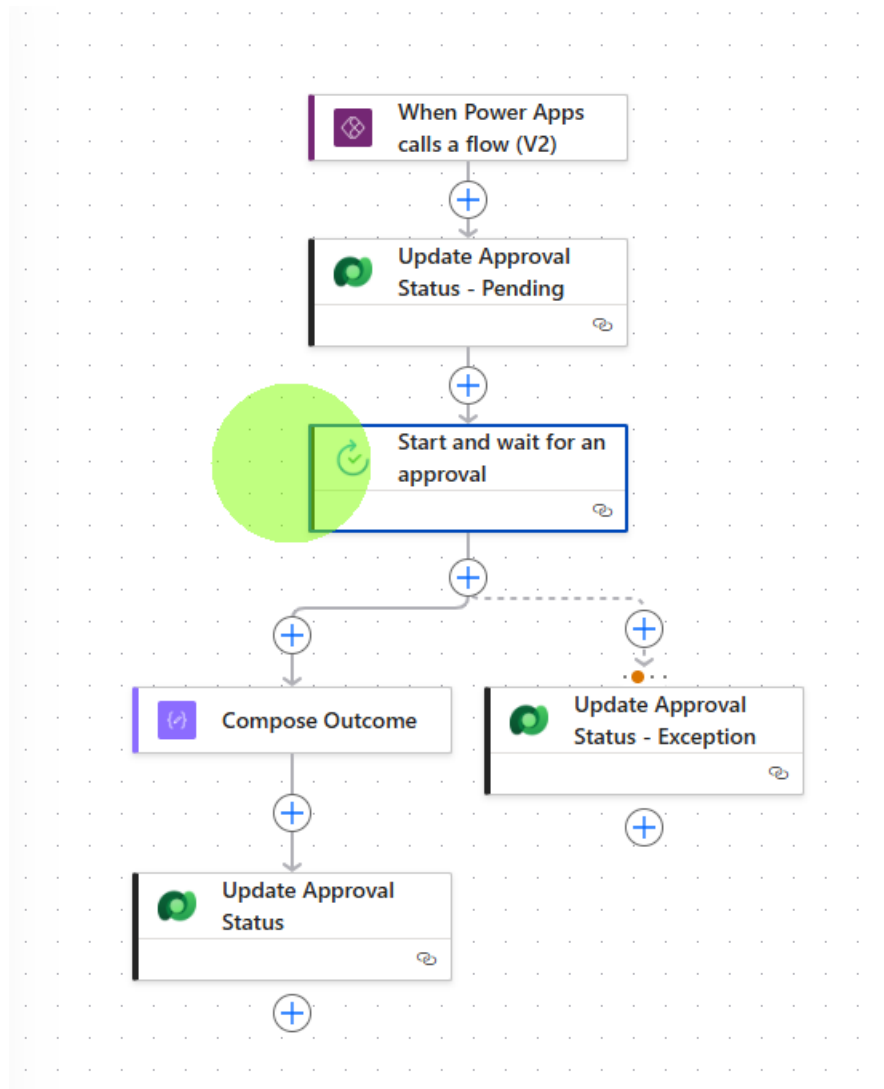
General

Action timeout ⓘ  
Specify the duration in ISO-8601 format

P1M

Networking

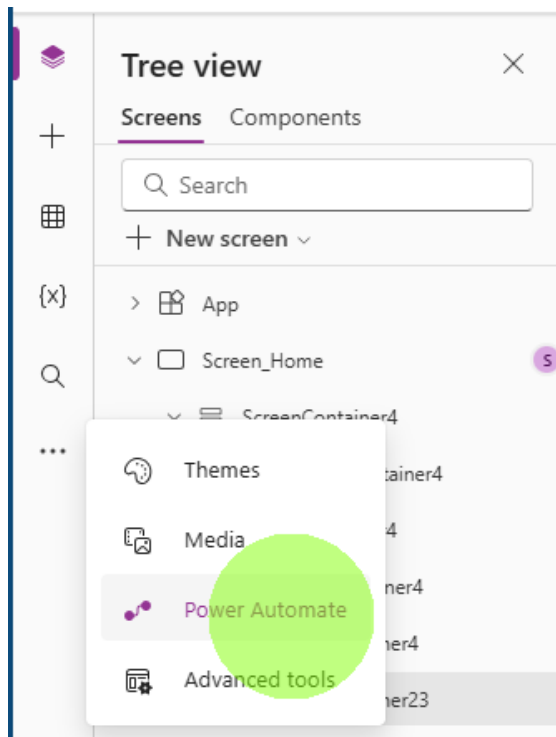
Completed flow should look like this.



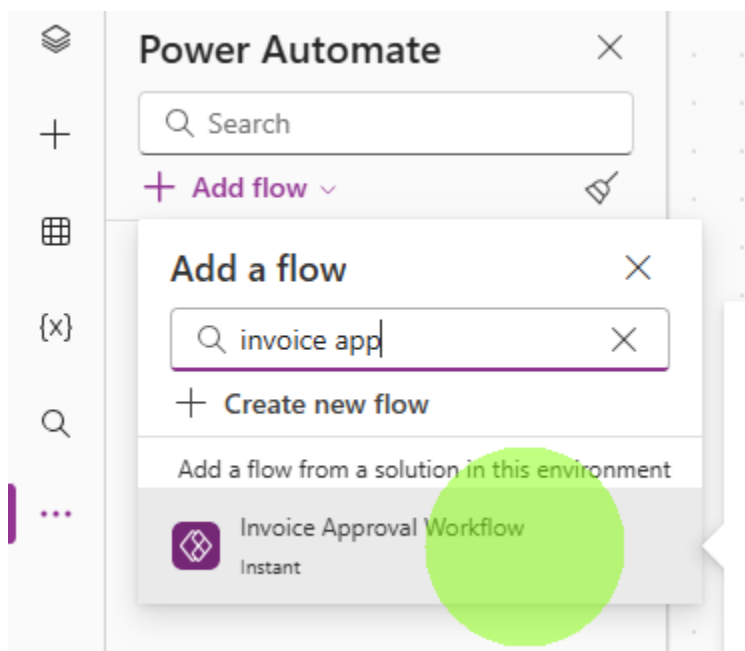
----- End of Task -----

## Task 4: Adding Flow to App

1. Open Finance Invoice App and select Power Automate.

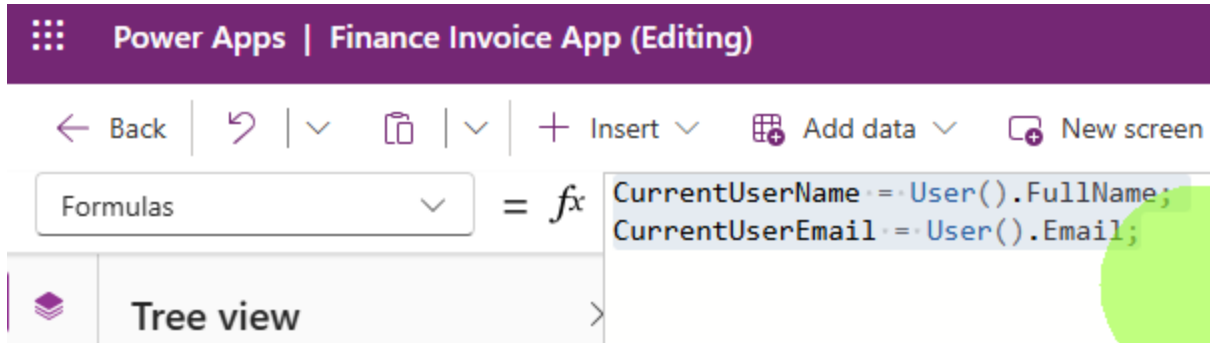


2. Under Add flow, look for Invoice Approval Workflow.



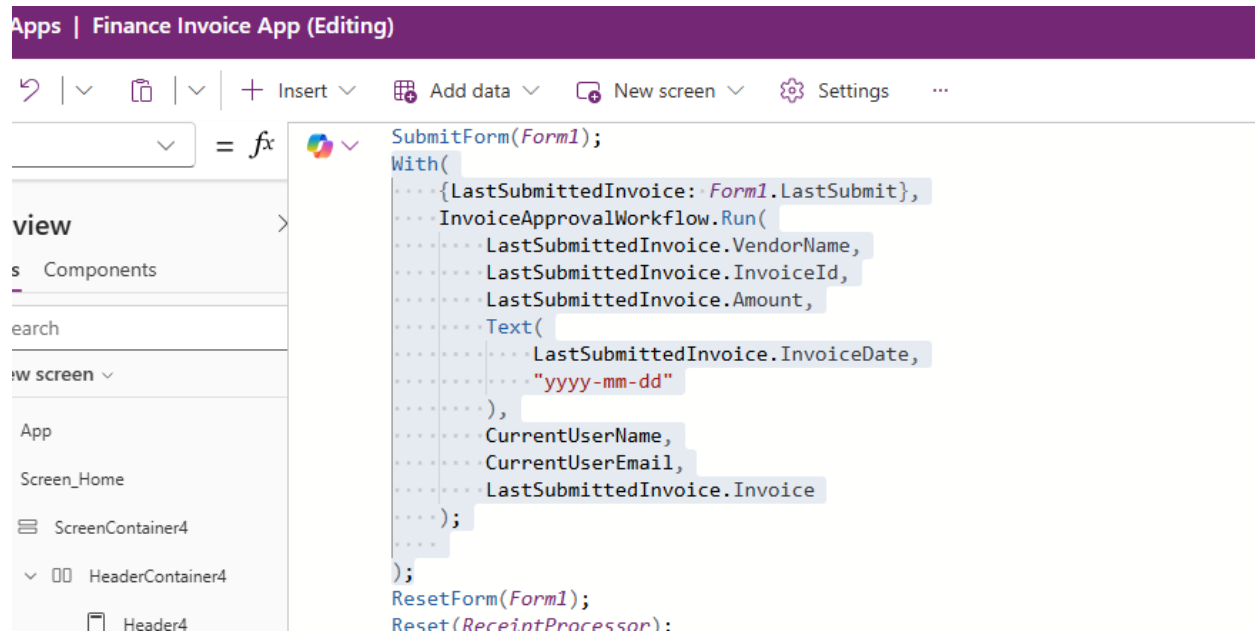
- Go to App → Formulas property and insert these named formulas.

```
CurrentUserName = User().FullName;
CurrentUserEmail = User().Email;
```



- Go to OnSelect property of Submit button and insert this formula between SubmitForm and ResetForm.

```
With(
    {LastSubmittedInvoice: Form1.LastSubmit},
    InvoiceApprovalWorkflow.Run(
        LastSubmittedInvoice.VendorName,
        LastSubmittedInvoice.InvoiceId,
        LastSubmittedInvoice.Amount,
        Text(
            LastSubmittedInvoice.InvoiceDate,
            "yyyy-mm-dd"
        ),
        CurrentUserName,
        CurrentUserEmail,
        LastSubmittedInvoice.Invoice
    );
);
```



4. You can now save the app and test Submit invoices to see if the approval flow is triggered.

----- **End of Task** -----



# Exercise 3: Deep Linking for Direct Access

In this exercise, participants will enhance the existing approval flow and app to link the approval request item in the message.

## Objectives

After completing this exercise, participants will be able to:

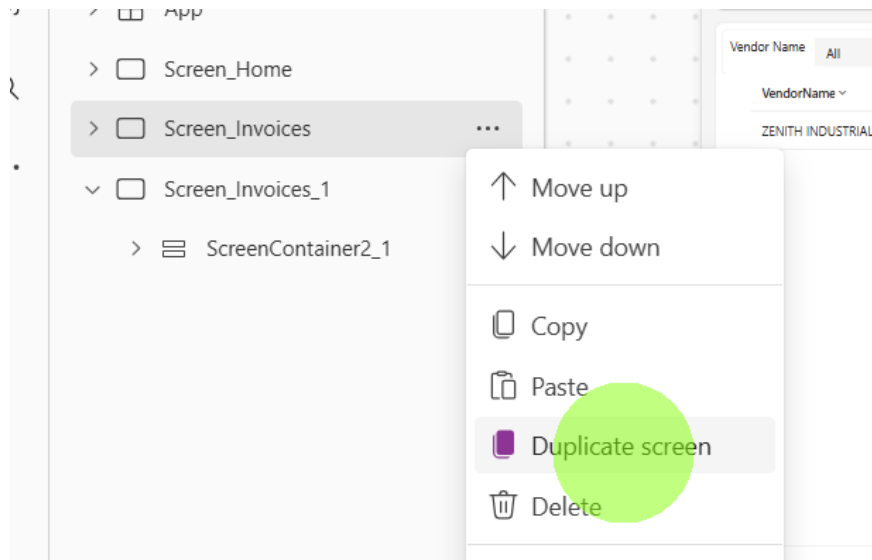
- Perform deep linking in Power App by URL

## Estimated Time

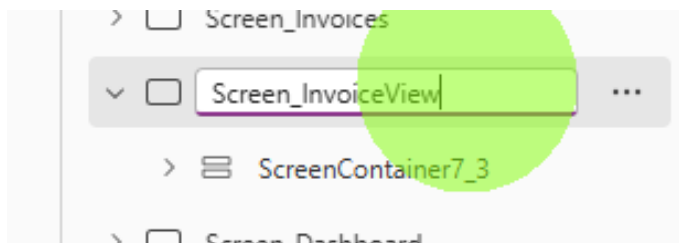
20-30 mins

## Task 1: Duplicate the existing Screen

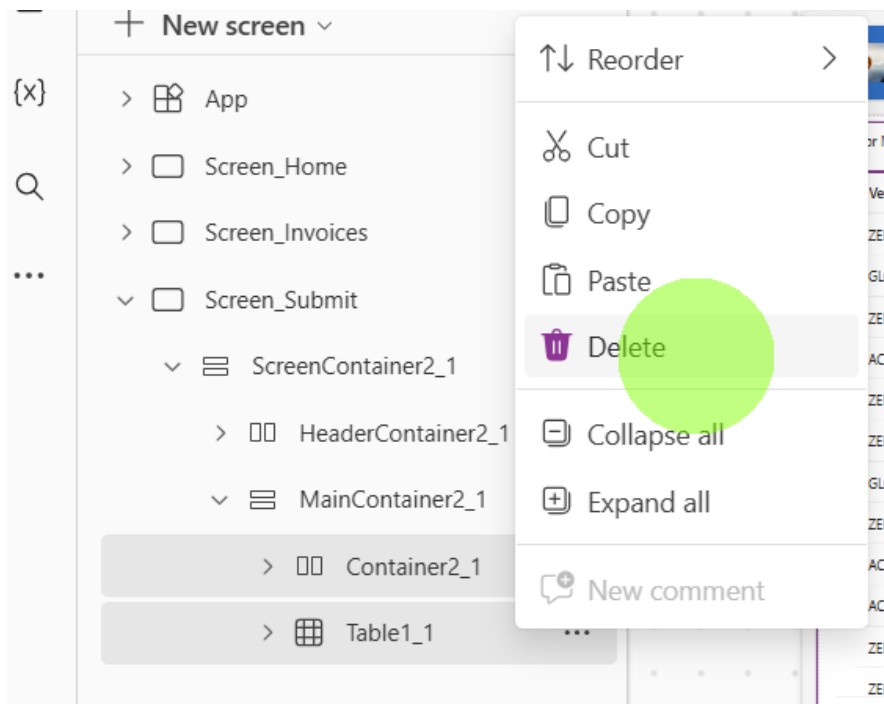
1. Duplicate Screen\_Invoices by following selecting the screen, right clicking it, and then selecting the 'Duplicate Screen' option.



Rename the new screen to Screen\_InvoiceView.



2. Expand the Main Container and delete the controls inside.

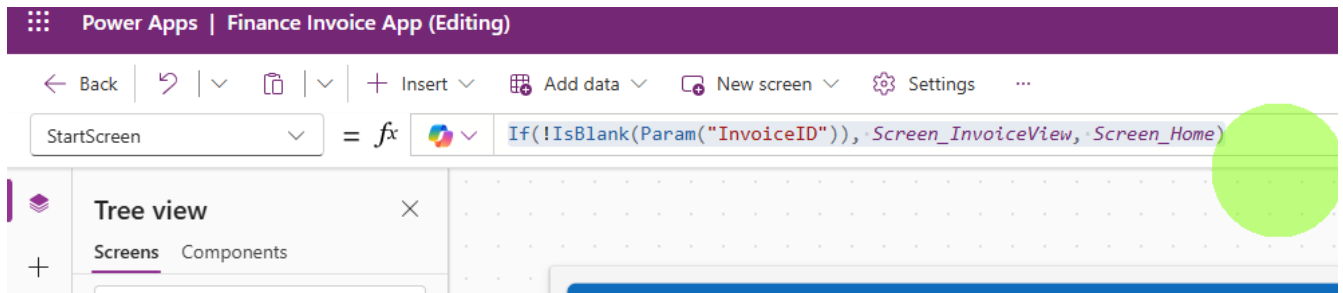


----- **End of Task** -----

## Task 2: Handle Deep Link ID in App

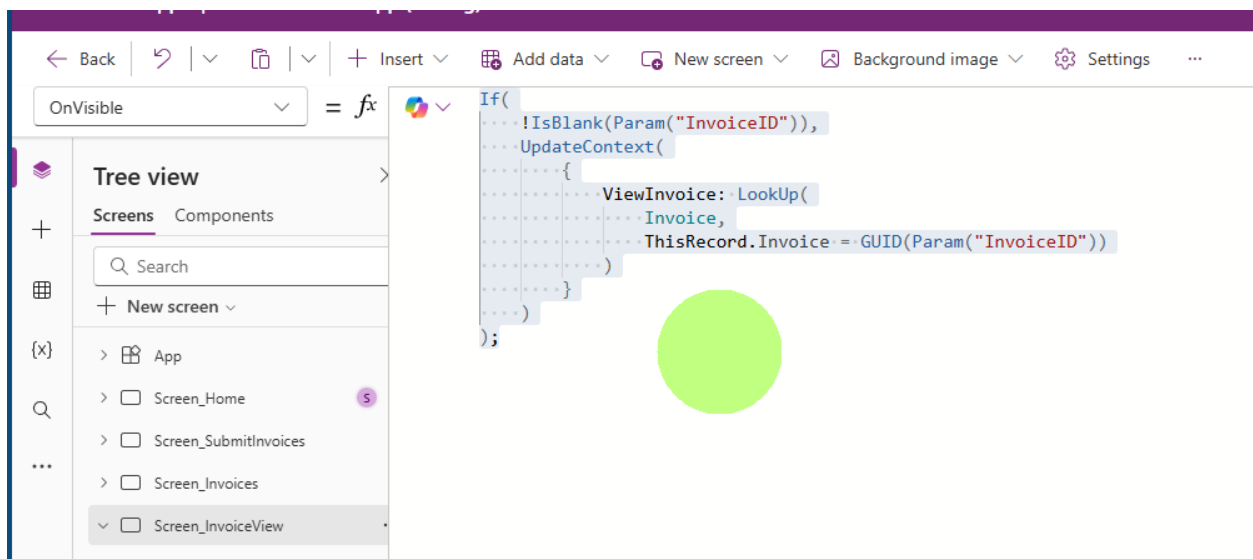
1. Go to App → StartScreen property and put the formula below.

```
If(!IsBlank(Param("InvoiceID")), Screen_InvoiceView, Screen_Home)
```

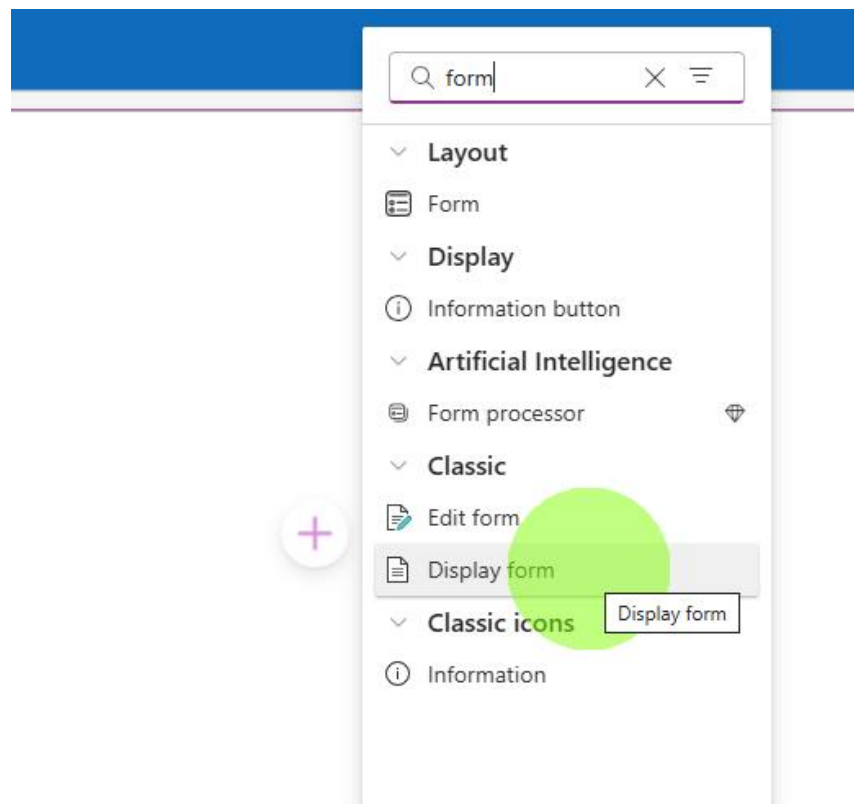


2. Go to Visible property under Screen\_InvoiceView and put the formula below.

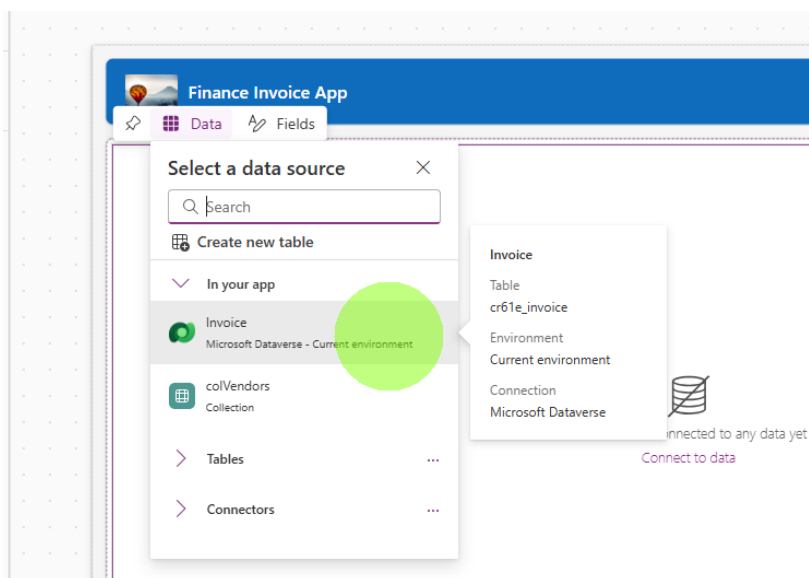
```
If(  
  !IsBlank(Param("InvoiceID")),  
  UpdateContext(  
    {  
      ViewInvoice: LookUp(  
        Invoice,  
        ThisRecord.Invoice = GUID(Param("InvoiceID"))  
      )  
    }  
  )  
);
```



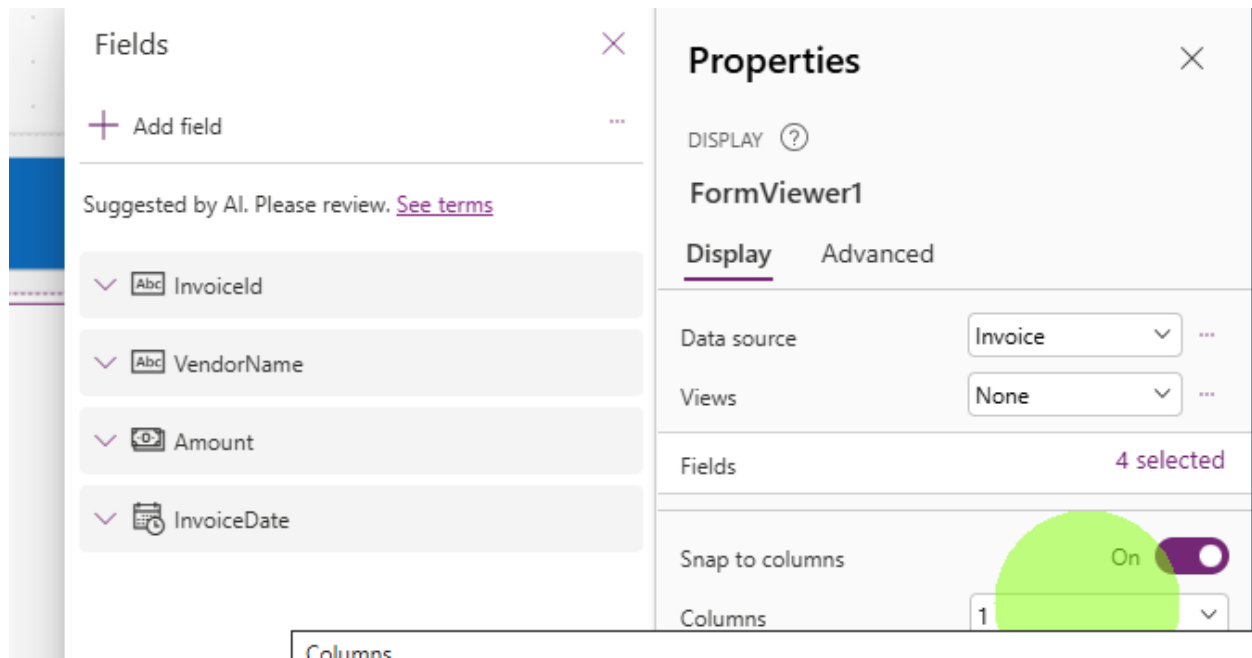
3. Add Display form control to the container.



4. Select Invoice as DataSource.

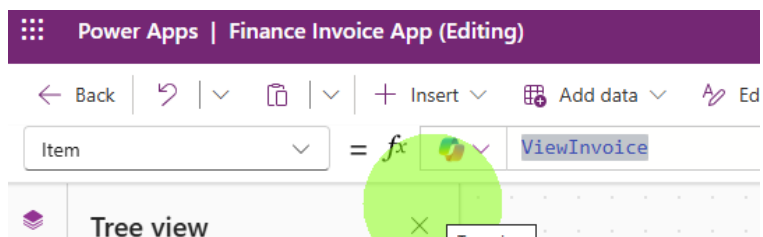


- Set the Columns to 1 and Edit the fields to show only 4 columns.

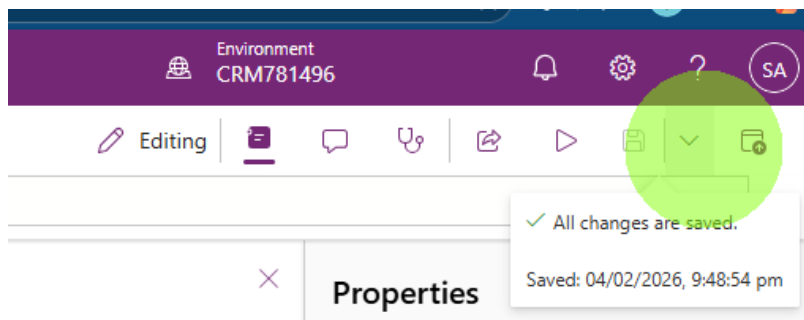


- Set Item property of the form to below.

ViewInvoice



- Save and publish the app.



End of Task

## Task 2: Include link in Power Automate Approval

1. In Invoice Approval Flow, select Start and wait for an approval stage.
2. Update Item Link and Item link description as below.

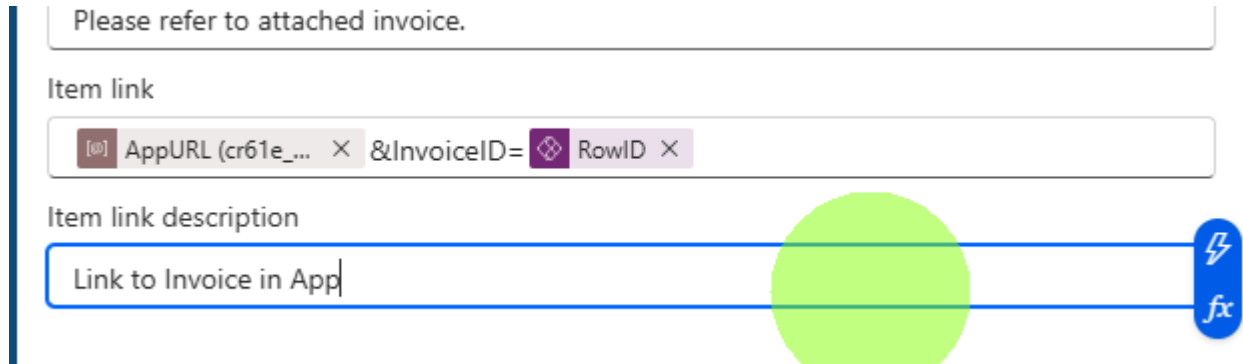
Please refer to attached invoice.

Item link

AppURL (cr61e\_... X &InvoiceID= RowID X

Item link description

Link to Invoice in App



3. Save and try submitting a new invoice via app. The new approval request will contain a link.

**Approvals**  
Approval request details

Requested

**Invoice Approval Request (Invoice ID: INV-3005)**

Vendor Name: ACME PTE LTD  
Invoice ID: INV-3005  
Invoice Amount: 4621.6  
Invoice Date: 13 May 2025  
Raised By: System Administrator

Please refer to attached invoice.

▼ Attachments

Link to Invoice in App  
https://apps.powerapps.com/play/...

apps.powerapps.com

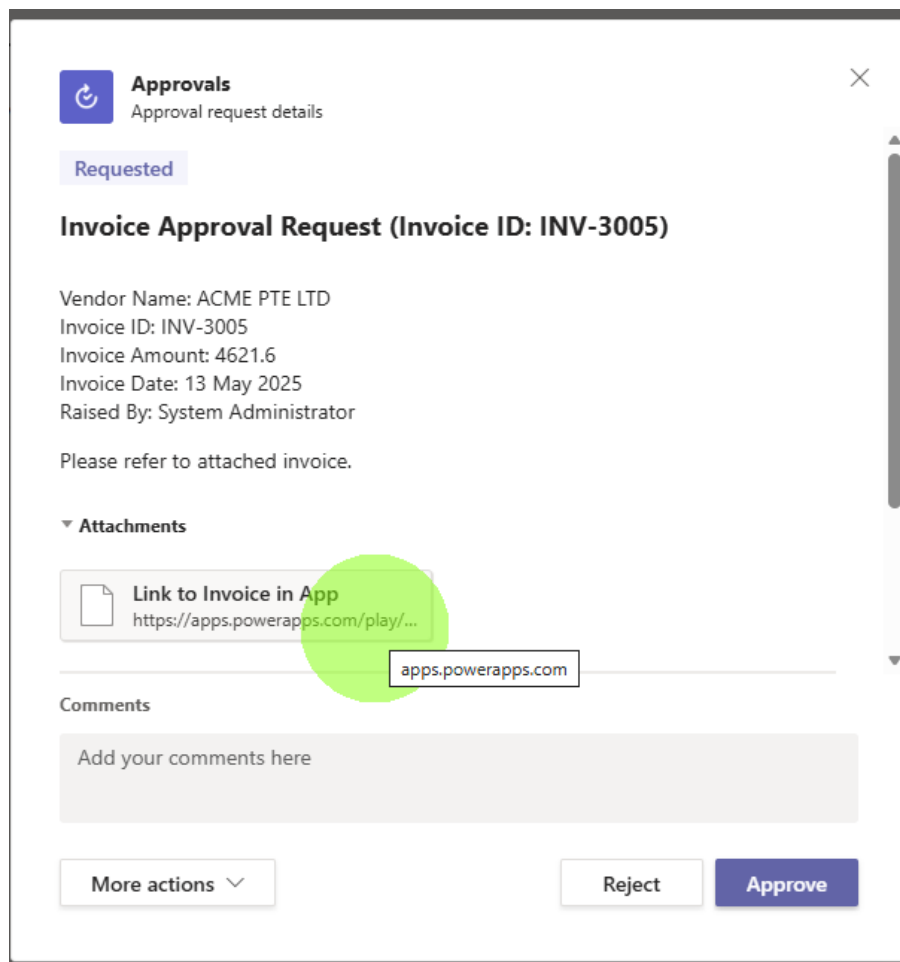
Comments

Add your comments here

More actions ▼

Reject

Approve



----- End of Task -----

# Exercise 4: Utilising Adaptive Card and Child Flow

In this exercise, participants will explore sending Teams messages using Adaptive Card in a child flow.

## Objectives

After completing this exercise, participants will be able to:

- Use adaptive cards to send out Teams notification and ask for user input
- Modularise their workflows by using Child Flows

## Estimated Time

20-30 mins



## Task 1: Change Response Options in Approval Workflow

1. Select Start and wait for an approval and change Approval type and Response options as below.

← Back Invoice Approval Workflow

Start and wait for an approval

Parameters Settings Code view Testing About

Approval type \*

Custom Responses – Wait for one response

Response options \*

\* Response options Item - 1

Approve

\* Response options Item - 2

Reject

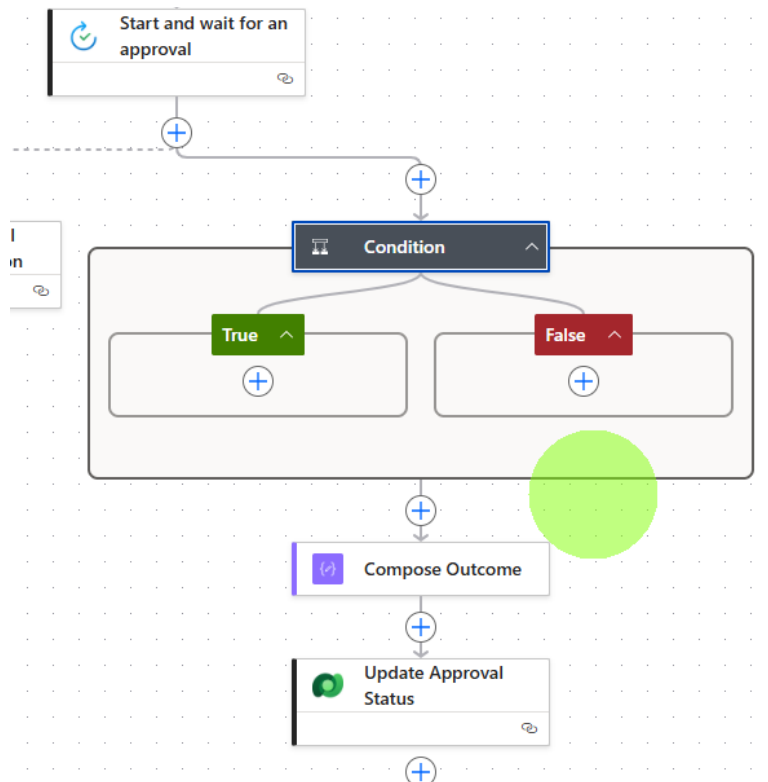
\* Response options Item - 3

Ask for more information

+ Add new item

Title \*

2. Add a Condition stage before Compose Outcome.



3. Set Condition as below.

**Condition**

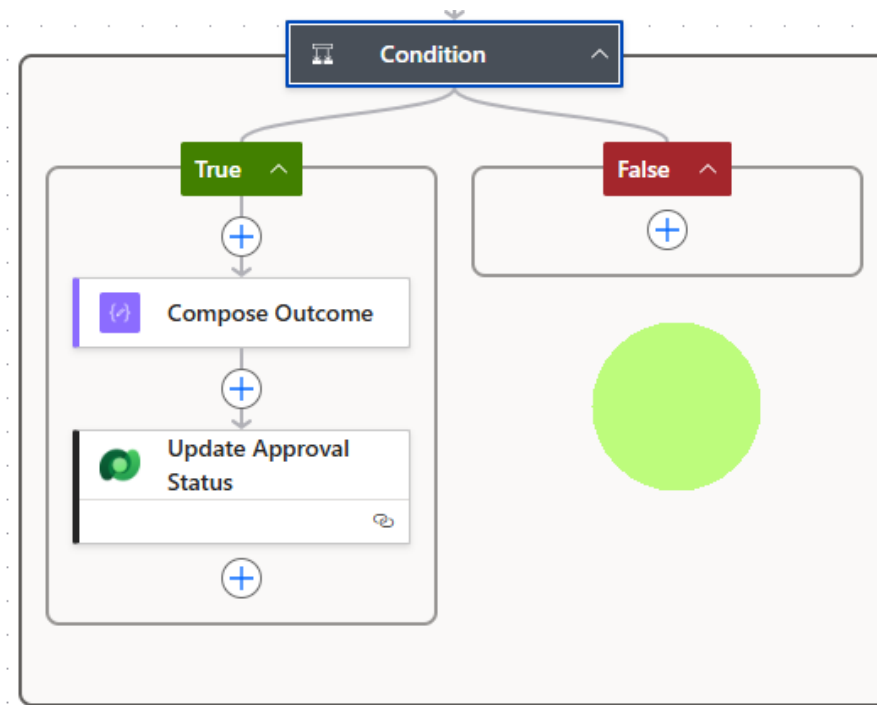
Parameters Settings Code view About

Condition expression \*

Provide the values to compare and select the operator to use.

Or				
<input type="checkbox"/>	Outcome X	is equal to	Approve	...
<input type="checkbox"/>	Outcome X	is equal to	Reject	...
<input type="checkbox"/>	Choose a value	is equal to	Choose a value	...
<input type="button" value="+ Add row"/> <input type="button" value="v"/>				

4. Move Compose and Update Approval Status stages to True branch.

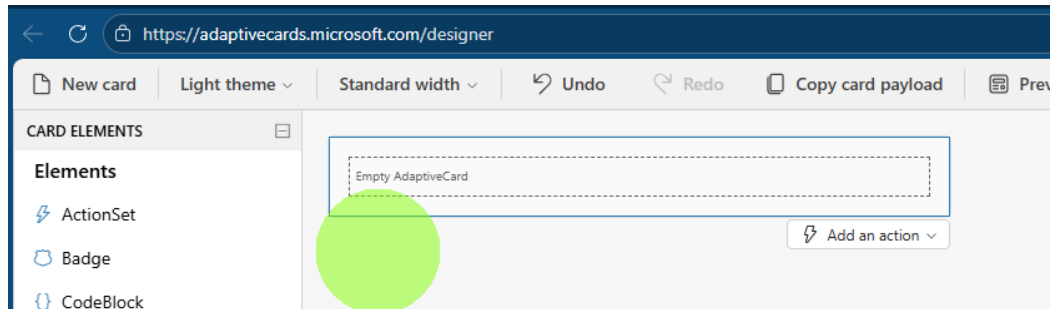


5. Save the flow first. We will create a child flow to put in False branch in the next task.

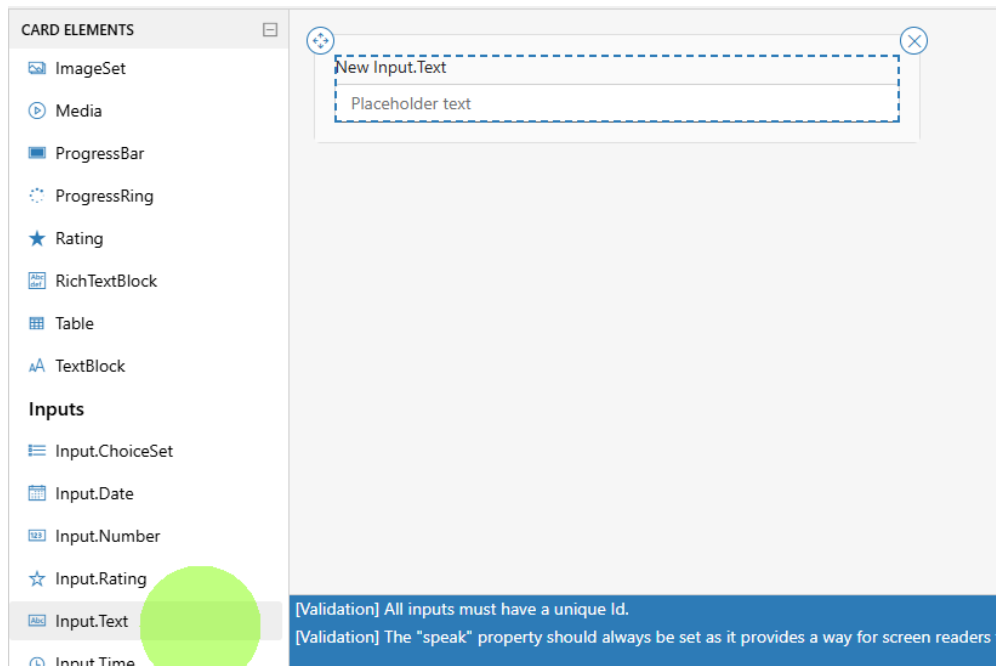
----- **End of Task** -----

## Task 2: Design Adaptive card to Get User Input

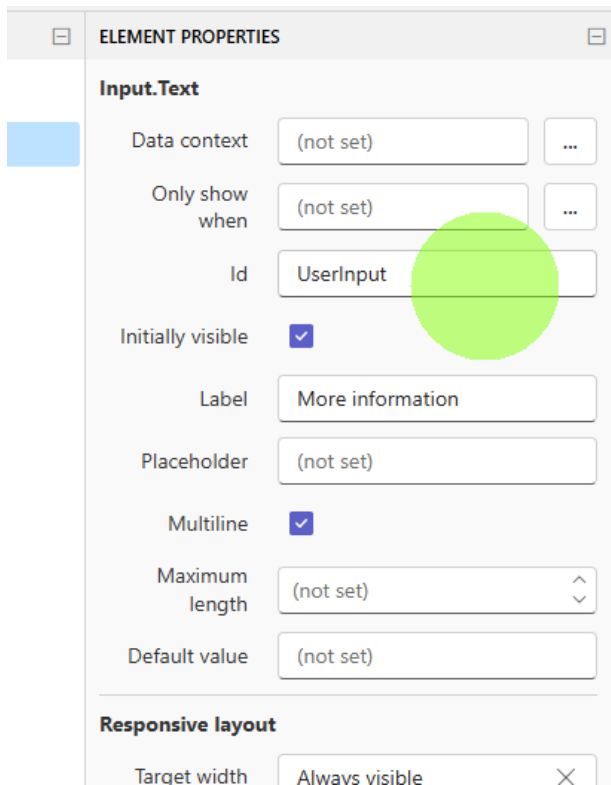
1. Go to <https://adaptivecards.microsoft.com/designer>.



2. Add Input.Text to the empty card.



3. Set these values in Element Properties.



**ELEMENT PROPERTIES**

**Input.Text**

Data context (not set) ...

Only show when (not set) ...

Id **UserInput**

Initially visible ☒

Label More information

Placeholder (not set)

Multiline ☒

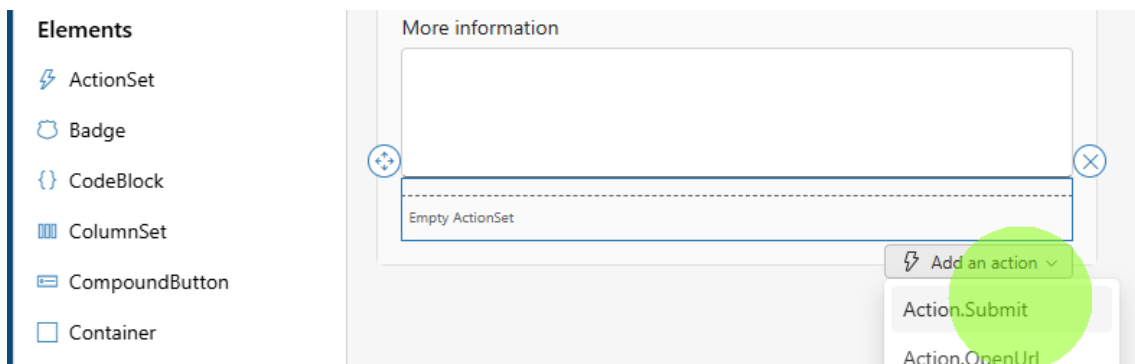
Maximum length (not set) ^ v

Default value (not set)

**Responsive layout**

Target width Always visible X

4. Add ActionSet and select Action.Submit.



**Elements**

- ⚡ ActionSet
- 🏷 Badge
- { } CodeBlock
- 📊 ColumnSet
- 📄 CompoundButton
- ☐ Container

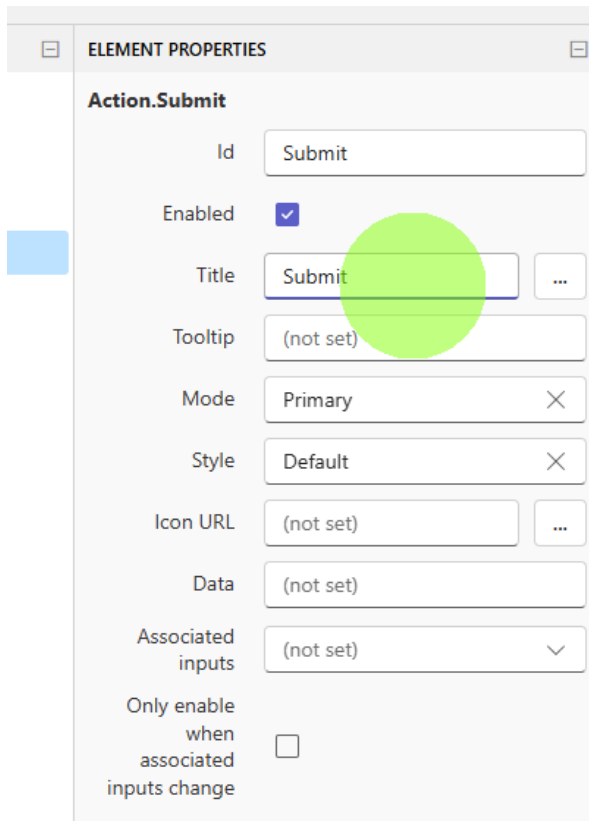
**More information**

Empty ActionSet

⚡ Add an action v

- Action.Submit
- Action.OpenUrl

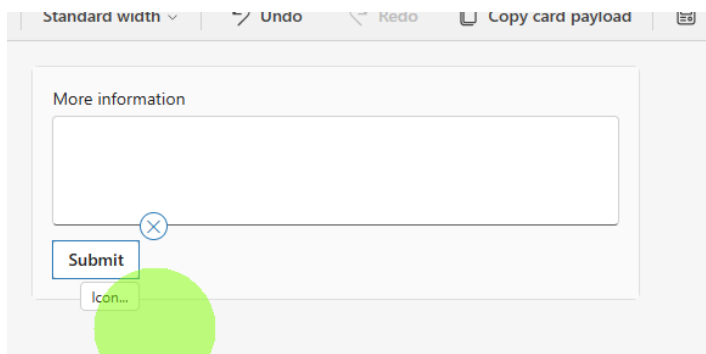
5. Set these values in Element Properties.



The screenshot shows the 'ELEMENT PROPERTIES' panel for an 'Action.Submit' element. A green circle highlights the 'Title' field, which contains the text 'Submit'. Other visible properties include:

- Id:** Submit
- Enabled:** ☒
- Title:** Submit
- Tooltip:** (not set)
- Mode:** Primary
- Style:** Default
- Icon URL:** (not set)
- Data:** (not set)
- Associated inputs:** (not set)
- Only enable when associated inputs change:** ☐

6. Your card should look like below. Copy the text in Card Payload Editor below.



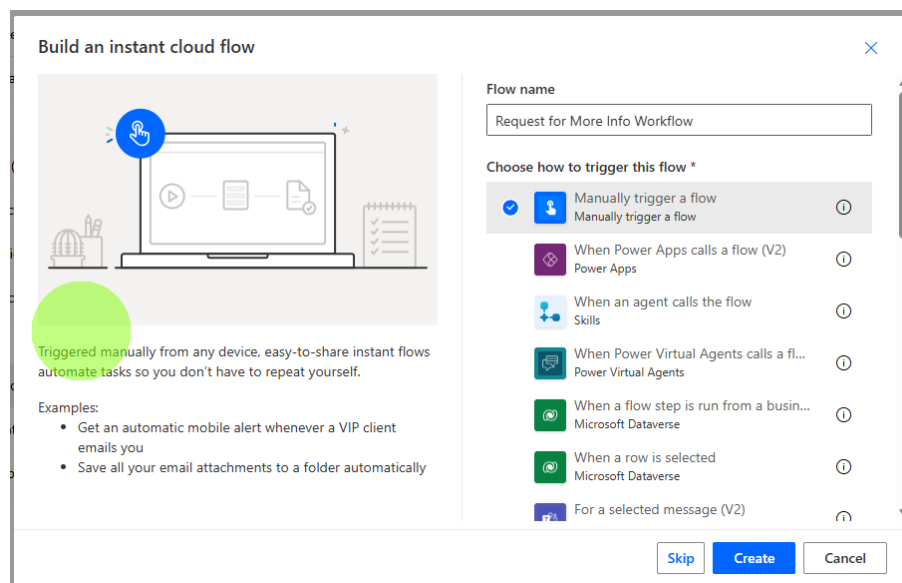
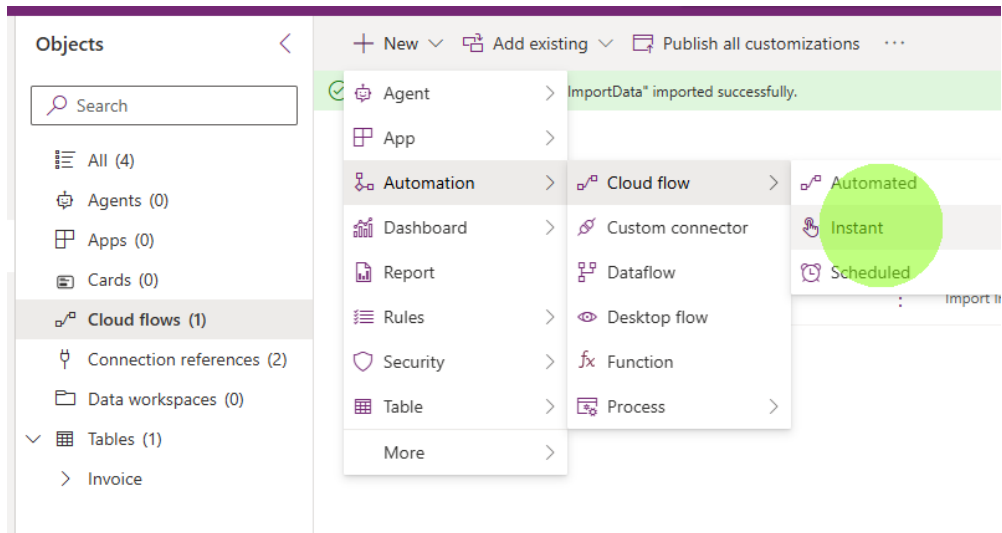
The screenshot shows a card layout in the Card Payload Editor. The card has a title 'More information' and a large text input field. Below the input field, there is a 'Submit' button and an 'Icon...' button. A green circle highlights the 'Submit' button. The top of the editor shows a toolbar with 'Standard width', 'Undo', 'Redo', and 'Copy card payload' options.

```
CARD PAYLOAD EDITOR
1 {
2   "type": "AdaptiveCard",
3   "$schema": "https://adaptivecards.io/schemas/adaptive-card.json",
4   "version": "1.6",
5   "body": [
6     {
7       "type": "Input.Text",
8       "id": "UserInput",
9       "label": "More information",
10      "isMultiline": true
11    },
12    {
13      "type": "ActionSet",
14      "actions": [
15        {
16          "type": "Action.Submit",
17          "title": "Submit",
18          "id": "Submit"
19        }
20      ]
21    }
22  ]
23 }
```

----- End of Task -----

## Task 3: Create a Child Flow to Get More Information

1. Go Invoice Processing App solution and create a new **instant cloud flow**, Name it: **Request for More Info Workflow**  
Choose **Manually trigger a flow**, then click **Create**





2. Add a new stage Post adaptive card and wait for a response under Teams connector.  
Message will come from Adaptive Card payload body we copied in Task 2.  
**Change version to "1.5" manually as Teams cannot display with "1.6".**

 Post adaptive card and wait for a response ⋮ <

Parameters Settings Code view Testing About

Post as \*  
Flow bot

Post in \*  
Chat with Flow bot

Message \*  

```
{
  "type": "AdaptiveCard",
  "$schema": "https://adaptivecards.io/schemas/adaptive-card.json",
  "version": "1.5",
  "body": [
    {
      "type": "Input.Text",
      "id": "UserInput",
      "label": "More information",
      "isMultiline": true
    },
    {
      "type": "ActionSet",
      "actions": [
        {
          "type": "Action.Submit",
          "title": "Submit"
        }
      ]
    }
  ]
}
```

Recipient \*  ⌵  
 RaisedByEmail × ;

Advanced parameters  
Showing 1 of 2 ⌵ Show all Clear all

Update message

3. Do a test run to see how the response body looks like.

```
    },  
    "submitActionId": "Submit",  
    "messageId": "1770216422844",  
    "messageLink": "https://teams.micr  
    "data": {  
        "UserInput": "hi"  
    }  
}
```

4. Add one more step Respond to a Power App or flow.

**Add an action**

🔍 respond

**All** Built-in Standard Premium Custom

**Microsoft Teams** ☆

Respond in Teams task module Post adapt

Post a choice of options as the Flow...

**Office 365 Outlook** ☆

Respond to an event invite (V2) Send email

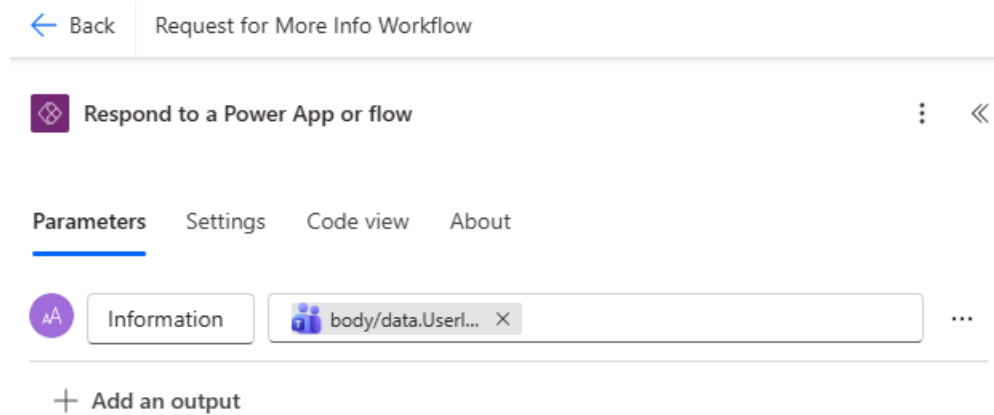
**Power Apps** ☆

Respond to a Power App or ... ⓘ ☆

5. Create an output variable named Information and paste in the formula below.

`outputs('Post_adaptive_card_and_wait_for_a_response')['body/data']['UserInput']`

Note `['UserInput']` requires manual typing as Power Automate doesn't provide the value automatically.




- Go back to Details page of the flow. Scroll down to look for Run-only user on the right-hand side of the screen.

**Solutions** ⓘ

Solution Name	Managed
InvoiceProcessingApp	No

**Process mining (preview)** ⓘ [Improve your flow](#)



Average run duration  
**00:01:08**

**Run-only user** [Edit](#)

Your flow hasn't been shared with anyone.


**Associated apps and flows** ⓘ [Edit](#)

You don't have any apps associated with this flow. [Learn more](#)

- Click edit and change the connection used for Microsoft Teams dropdown value to a connection (Use this connection .....).

**Connections Used**

These connections will provide the users listed here to have run-only access to this flow. Unless providing their own connection, run-only users will not have access to these connections outside this flow.

 **Microsoft Teams**  
Run-only users will be asked to provide their own connection to this connector.

Provided by run-only user ▼

Provided by run-only user

Use this connection (admin@CRM781496.onmicrosoft.com)

Save Cancel

- Click OK when prompted.

**Connections Used**


Users with run-only access will not control or have access to these connections outside of this flow.

OK Cancel

- Click Save.

**Connections Used**

These connections will provide the users listed here to have run-only access to this flow. Unless providing their own connection, run-only users will not have access to these connections outside this flow.

 **Microsoft Teams**  
Run-only users will be asked to provide their own connection to this connector.

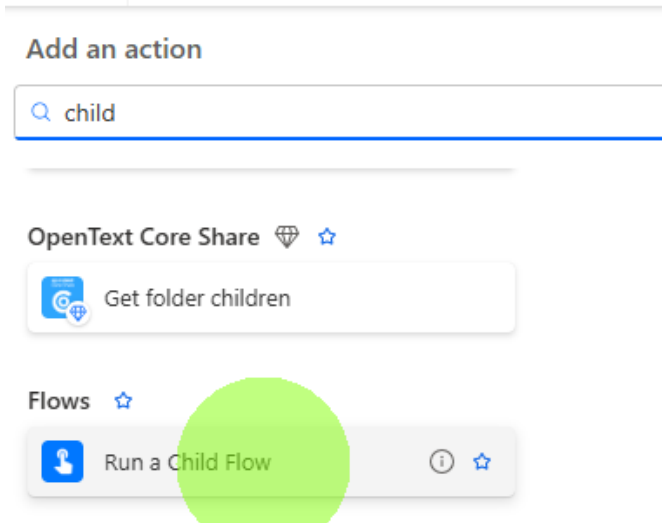
Use this connection (admin@CRM781496.onmicrosoft.com) ▼

Save Cancel

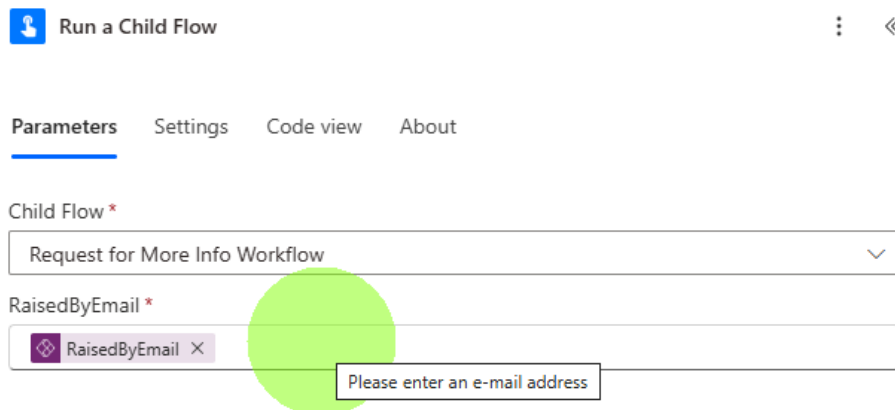
----- **End of Task** -----

## Task 4: Add Child Flow to the Main Flow

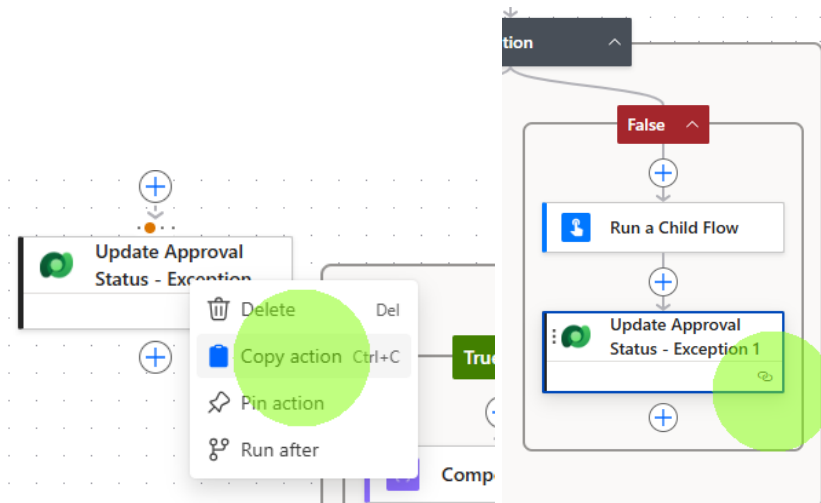
1. Go back to Invoice Approval Workflow.
2. Under False condition, add **Run a child flow**.



3. Select the child flow we have created in Task 3 and provide input parameter.

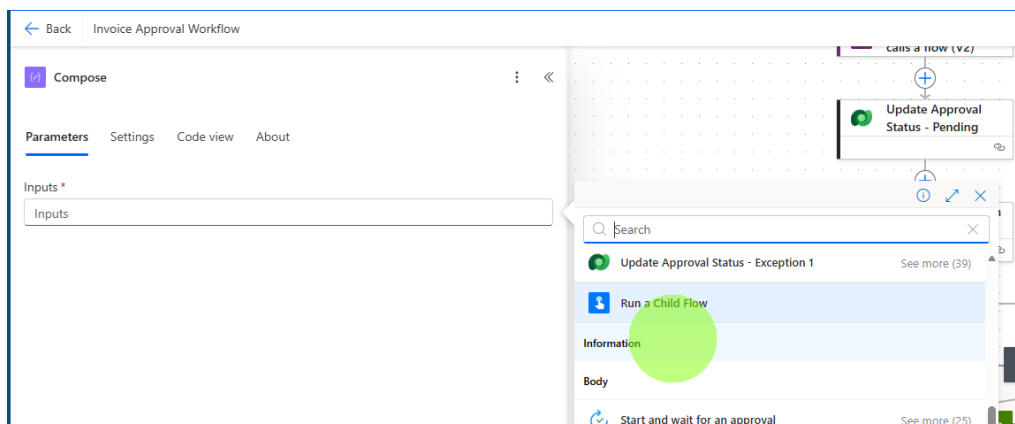


4. Copy Update Approval Status – Exception and paste after child flow stage.



Note: For simplicity of the lab exercise, it's just updating back the Dataverse record with exception status. In real life, you can design the workflow to trigger another round of approval along with the more information provided by the user.

5. Add a compose stage to view the information returned from the child flow.



----- End of Task -----