Microsoft Power Platform Microsoft Teams

Building solutions with Dataverse for Teams

Lab 3 – Power Automate

Workshop Version: 1.4, Published: 02-2023

Table of contents

Exercise 1:
Announcement Flow

Important – Working with lab tenant

- All the labs in this course require you to use the latest version of Edge or Chrome in Incognito/InPrivate mode.
- Use Office 365 credentials retrieved from Skillabe (Labs on Demand) in Lab 0.
- Always remember to replace M365xXXXXXX with your lab tenant prefix.
- If you are experiencing any problems with working in your lab tenant – please, notify your instructor as soon as possible.

Objectives:

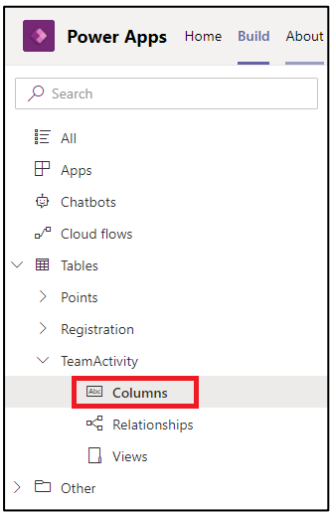
- Create a basic flow in Dataverse for Teams
- Utilize adaptive cards to send rich messages to Team channel

Estimated time:

20 minutes

Task 1: Prepare an Adaptive Card

1. **Navigate to Dataverse for Teams content** of your Team
2. In the left navigation, expand **Tables** -> **TeamActivitity** and select **Columns**

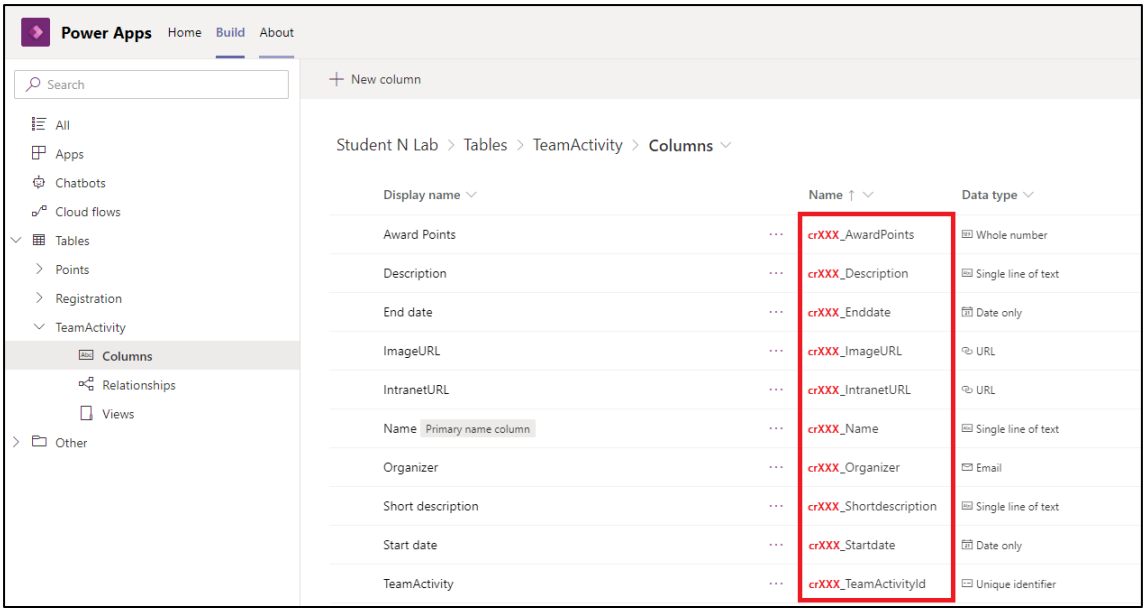


3. List of columns will be displayed.
   Notice **DisplayName** and **Name** columns:

   - **DisplayName** – is how the column name is usually displayed to user
   - **Name** – is an actual internal name of the column that we will use in Power Automate and Power BI labs

   Custom columns have prefix (PublisherId) *crXXX* – **please, memorize it or copy it somewhere.**

   You may use Custom view to see only custom columns.
   Example:



This prefix is unique for the environment and is used for internal names of all non-default assets, that you create in your Dataverse for Teams (e.g., apps, flows, tables, columns)

4. Navigate to https://raw.githubusercontent.com/Vas-MSFT/building-solutions-with-dft/main/AdaptiveCard.json

5. In browser, press **Ctrl+S** to save **AdaptiveCard.json** file locally

6. Open downloaded file in **Visual Studio Code** (or any preferred text editor)

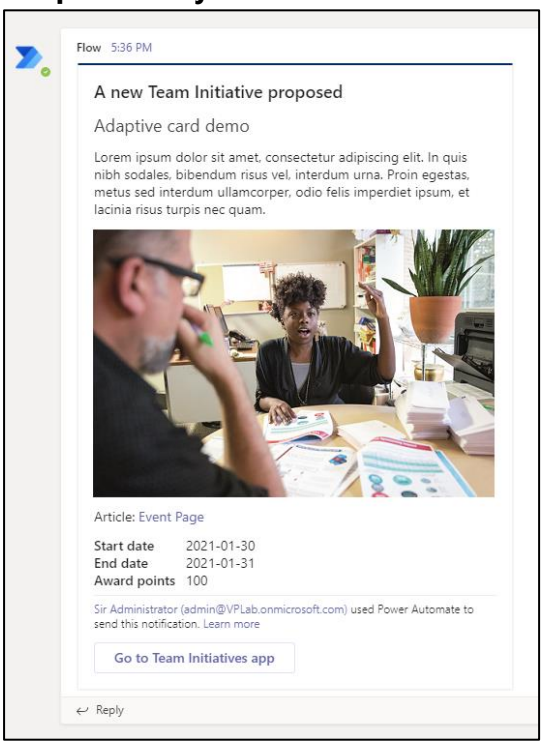
Adaptive cards

Adaptive Cards' content is declared using JSON, that relies on unified adaptive cards schema.

Schema explorer: [Schema Explorer | Adaptive Cards]

Designer tool: [Designer | Adaptive Cards]

AdaptiveCard.json file describes this card:

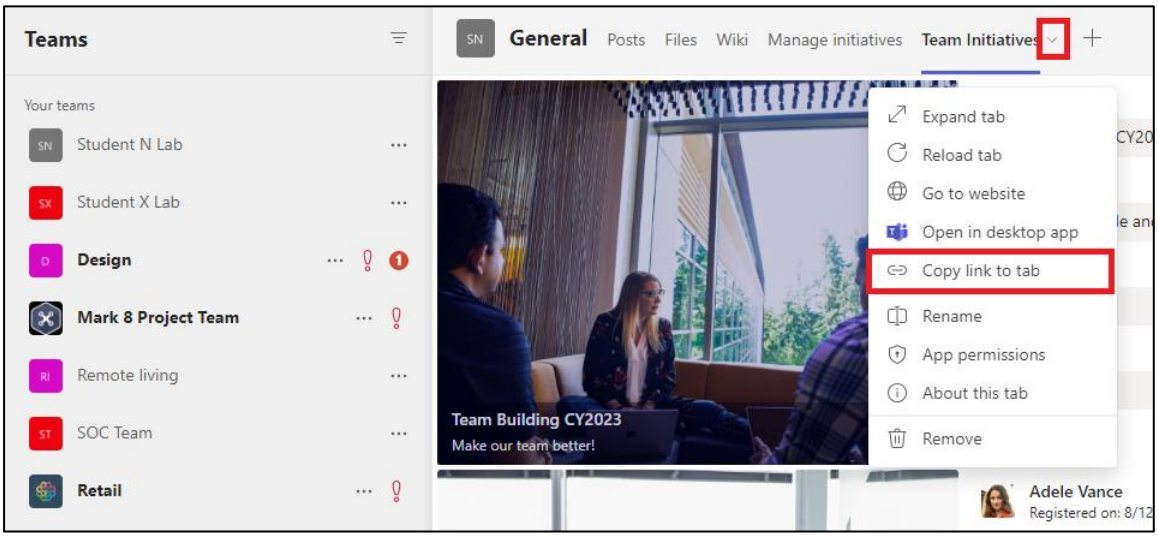


You can do **optional task** in the end of this exercise, if you want to know more about how it's done or edit this card.

7. Press **Ctrl+H** to open Replace dialog and **replace all** *crXXX* with your Dataverse for Teams prefix (from Step 3)

8. In **Microsoft Teams**, go to your team and the channel, where **Team Initiatives app** is published.

Click **ellipsis (…)** in the top right corner and select **Copy link to tab**



9. Scroll to the end of **AdaptiveCard.json** file and paste the link, copied in the previous step, instead of *<<PUT DEEP LINK HERE>>*:

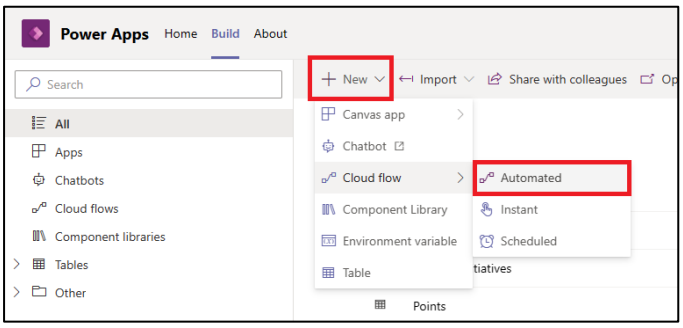
```
"actions": [
    {
      "type": "Action.OpenUrl",
      "title": "Go to Team Initiatives app",
      "url": "<<PUT DEEP LINK HERE>>"
    }
  ]
```

   Button on the card will redirect to this URL now.

10. Save the file. **Task is completed.**

Task 2: Create Initiative announcement flow and set up a trigger

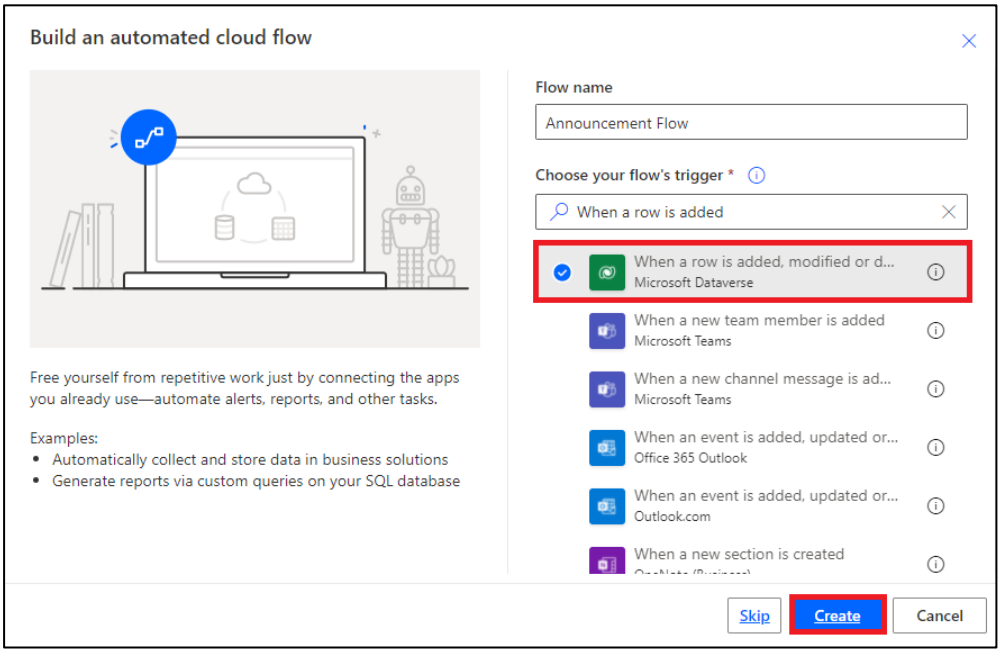
1. **Navigate to Dataverse for Teams content** of your Team
2. To create a new flow, click **+New -> Cloud Flow -> Automated**



3. Set **Flow name** as *Announcement Flow*

   Search for *"When a row is added"*

   Select **When a row is added, modified, or deleted** trigger from **Microsoft Dataverse** connector
   Click **Create** button

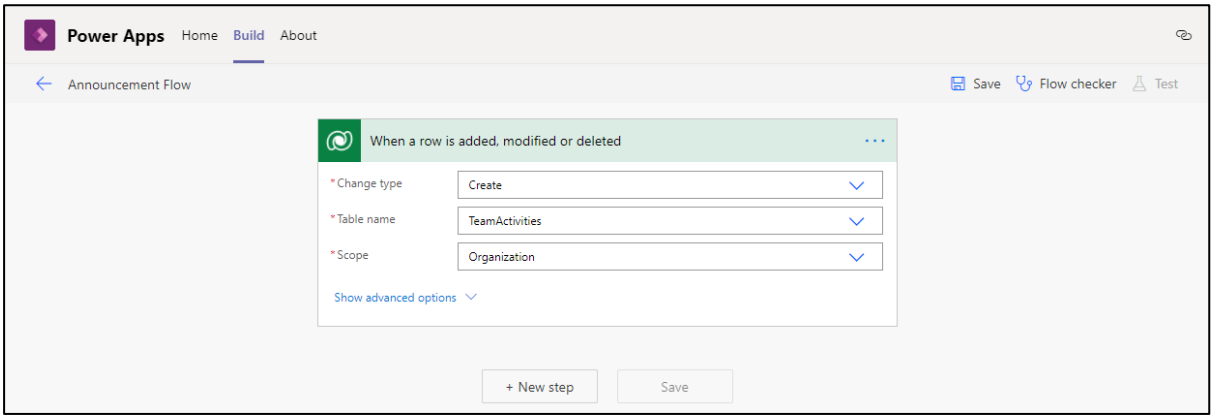


4. Set properties for the trigger:

   Change type: Create

   Table name: TeamActivities

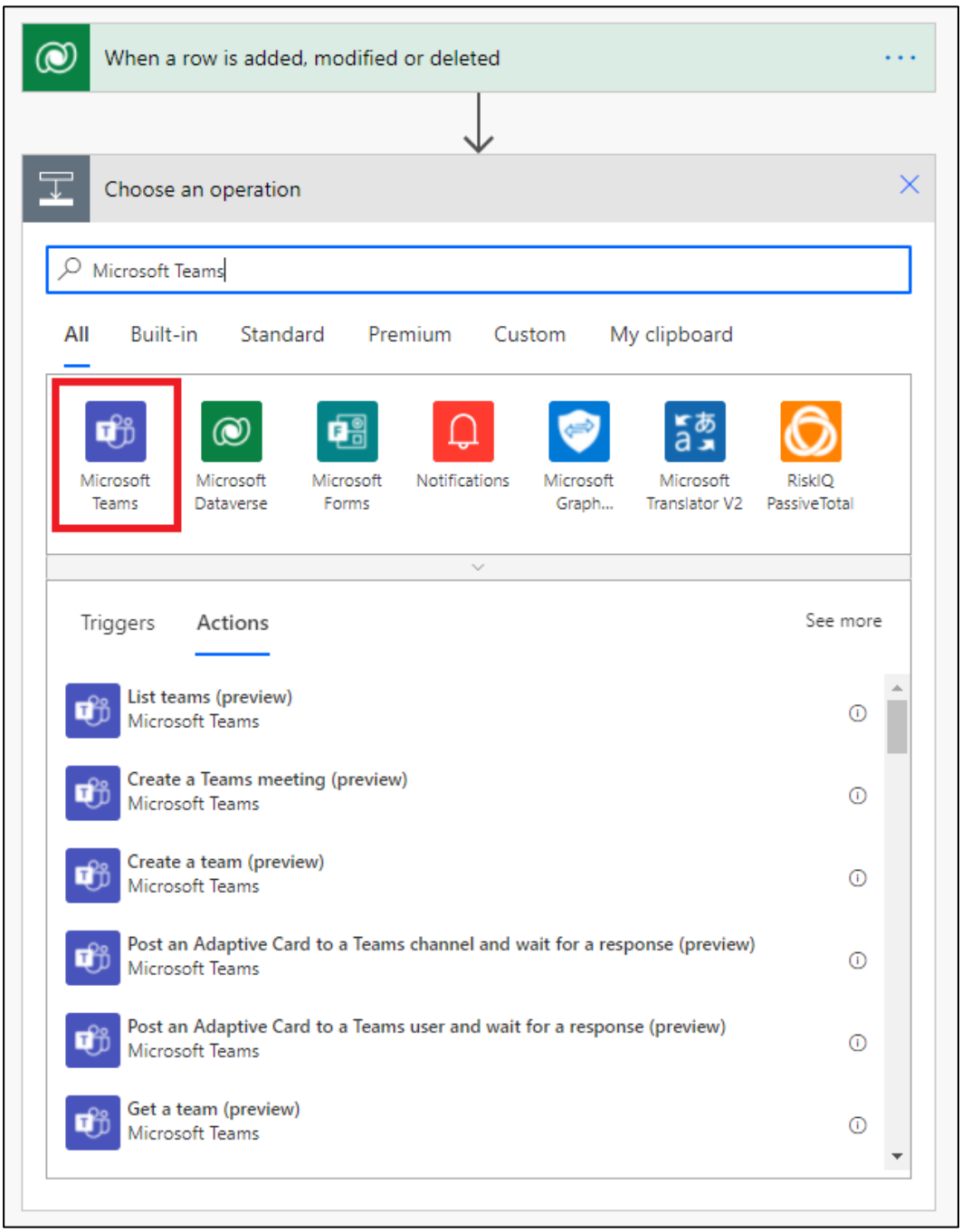
   Scope: Organization



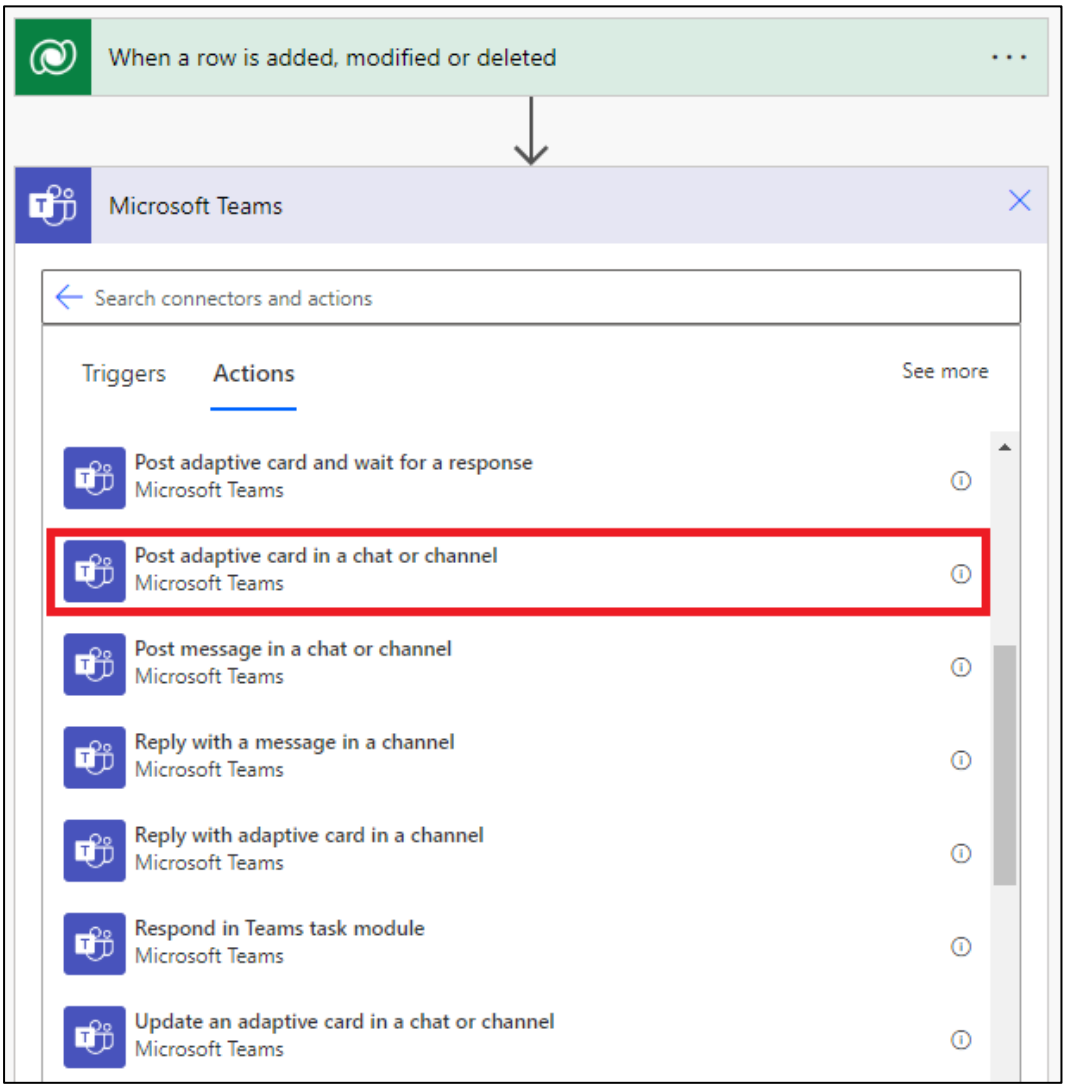
Task is completed.

Task 3: Create an announcement message using Adaptive Card

1. In **Announcement Flow**, click **+New step** button
   Select **Microsoft Teams** connector by clicking on it
   (use Search, if this connector is not present)



2. Choose **Post adaptive card in a chat or channel** action by clicking on it



3. Set properties for this action as

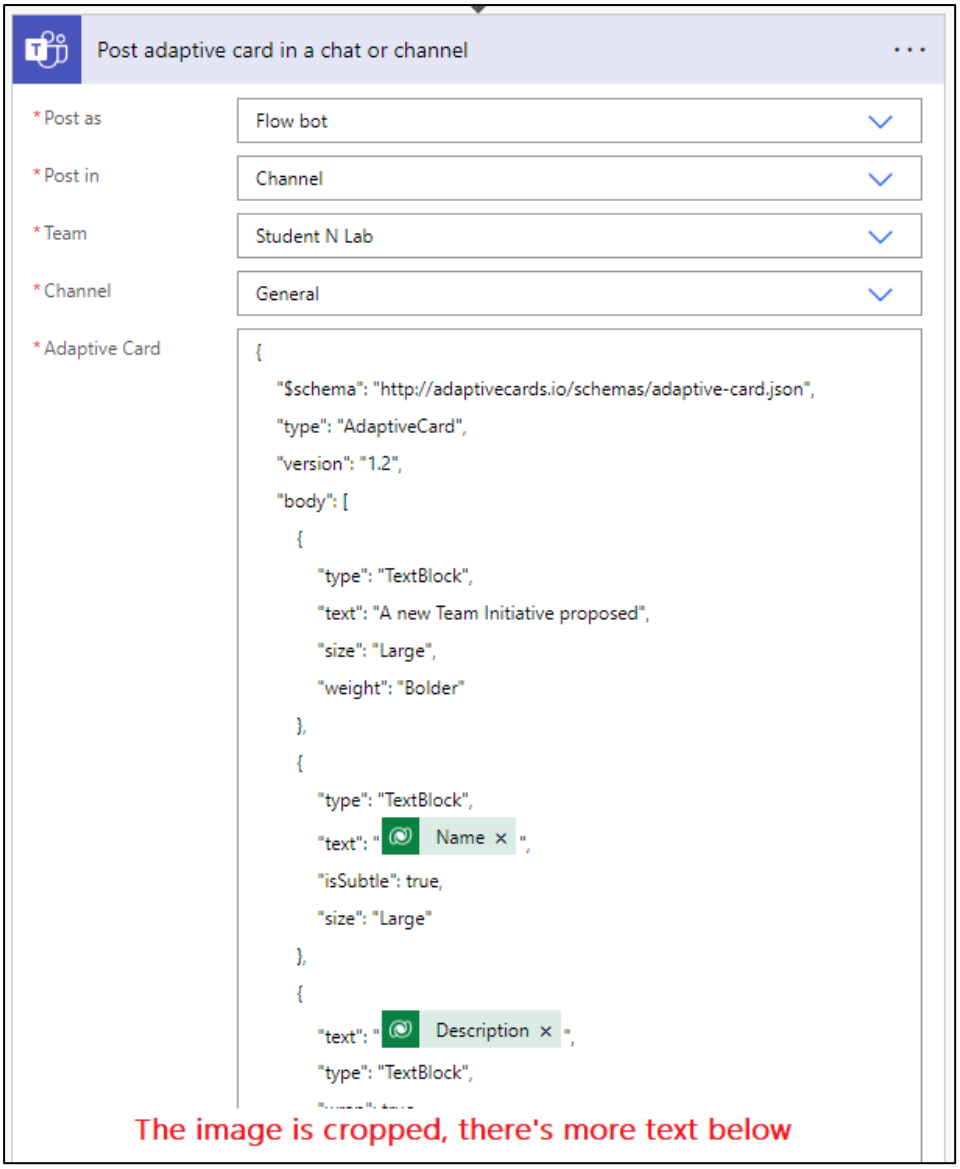
Post as: Flow bot

Post in: Channel

Team: Student N Lab (your Team name)

Channel: General

Message: *copy the contents of AdaptiveCard.json file here*



Trigger output references

`@{triggerOutputs()?['body/crXXX_name']}` is displayed as Name ×

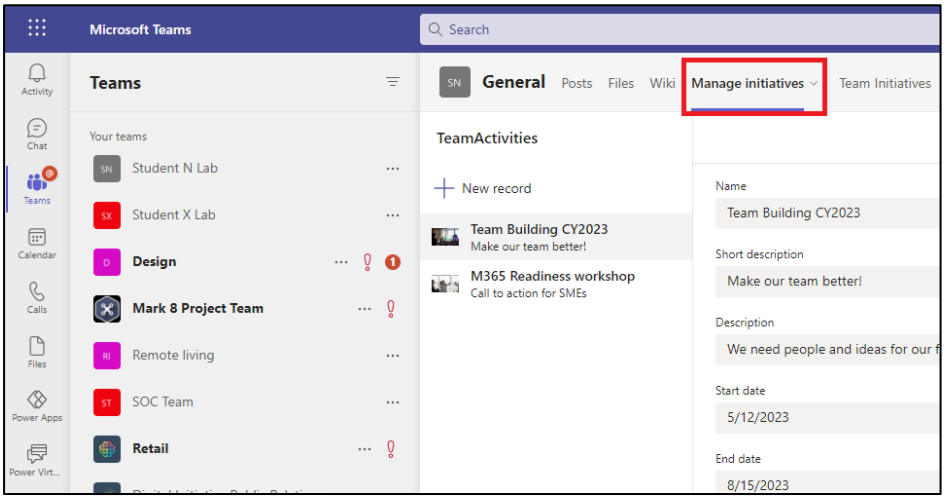
You could add outputs of trigger manually to the card, but we spared some time by using Workflow Definition Language (WDL) to reference trigger outputs.

You can use these expressions to reference any output property in Power Automate:

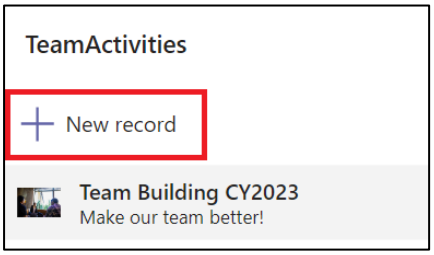
- triggerBody()?['PropertyName'] – reference to trigger output property
- body('ActionName')?['PropertyName'] – reference to action output property. There will be more than one action in your flows, that's why you need to define action name.

4. Check **Flow checker** and **Save** your flow.

5. Navigate to **Manage initiatives** app, that was published to your team **General** channel in **Lab 1**



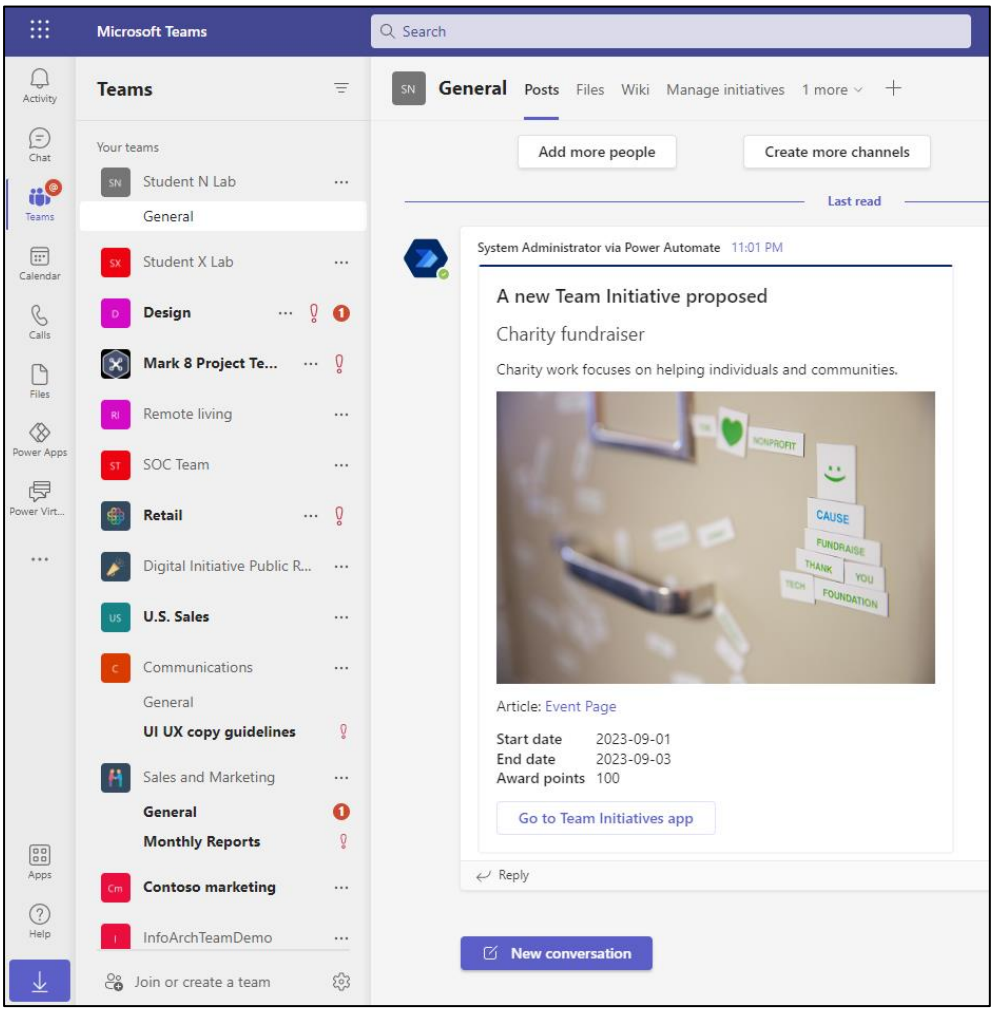
6. Click **+New record** to create a new initiative



7. Fill in the form with the value given below and click ✓ in the top right corner.

Column	Value
Name	Charity fundraiser
Short description	Your chance to make world a better place!
Description	Charity work focuses on helping individuals and communities.
Start date	12/01/2023
End date	01/01/2024
Award Points	100
IntranetURL	https://vas-msft.github.io/building-solutions-with-dft/Activity-Page-03
ImageURL	https://raw.githubusercontent.com/Vas-MSFT/building-solutions-with-dft/main/LabImages/ActivityImage03.png
Organizer	admin@m365xXXXXXX.onmicrosoft.com (replace xXXXXXX)

8. Grab some coffee, while flow gets kicked off by a new record created in TeamActivities table (up to 5 minutes).
9. **Adaptive card** with the announcement should appear in **General** channel of your Team



Task is completed.

Optional task: Understanding Adaptive Card

1. Open locally downloaded **AdaptiveCard.json** file

   Let's review its content step-by-step.

2. Notice, that before adding the content of the card, **schema information** is defined. It is used by client application to understand how this card should be rendered.

   Later versions of schema support more card features.

   In February 2023, the latest schema version is 1.6 (preview) and Microsoft Teams supports cards of 1.4 version.

   Schema info

   ```
   "$schema": "http://adaptivecards.io/schemas/adaptive-card.json",
   "type": "AdaptiveCard",
   "version": "1.2",
   ```

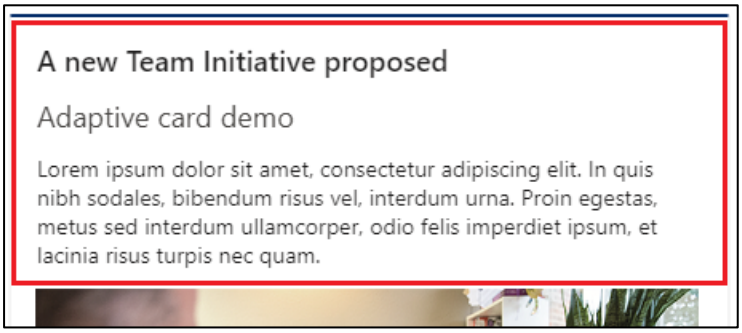
3. **Body** of the card is the collection of different types of content blocks

   For example, first three blocks contain simple text (**TextBlock** type) with different formatting

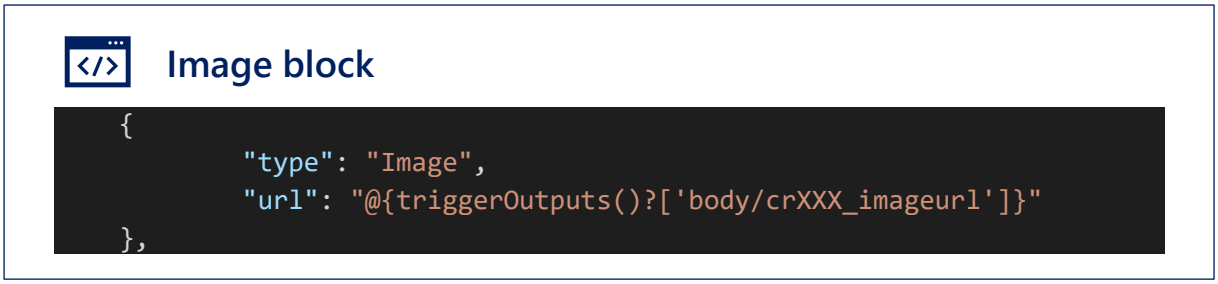
   Text blocks

   ```
   "body": [ // Body collection is opened
           {
               "type": "TextBlock",
               "text": "A new Team Initiative proposed",
               "size": "Large",
               "weight": "Bolder"
           },
           {
               "type": "TextBlock",
               "text": "@{triggerOutputs()?['body/crXXX_name']}",
               "isSubtle": true,
               "size": "Large"
           },
           {
               "text": "@{triggerOutputs()?['body/crXXX_description']}",
               "type": "TextBlock",
               "wrap": true
           },
   ```

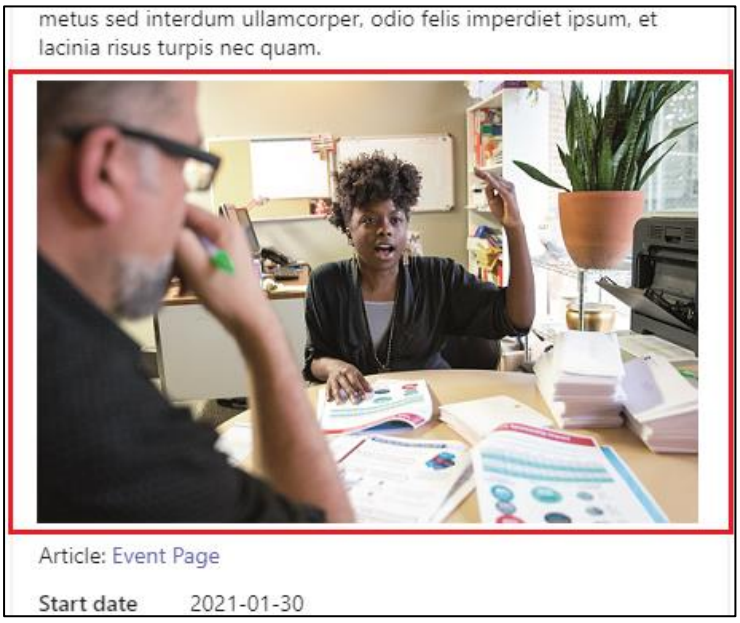
First text block contains static text, the next two text blocks are defined by the trigger outputs and will display **Name** and **Description** from **TeamActivities**:



4. Image is added by adding block of **Image** type



URL points to the web resource, that contains an image. In this case it's a value of **ImageURL** column in **TeamActivities** record.



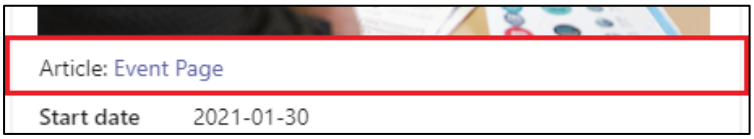
5. Next part is also just a **TextBlock**, but this one contains a **clickable link**.

TextBlock with hyperlink

```
{
 "type": "TextBlock",
 "text": "Article: [Event Page](@{triggerOutputs()?['body/crXXX_intraneturl']})"
}
```

Syntax: `[Link Text](Link URL)`

It redirects user to SharePoint page defined in **IntranetURL** column of **TeamActivities** record.

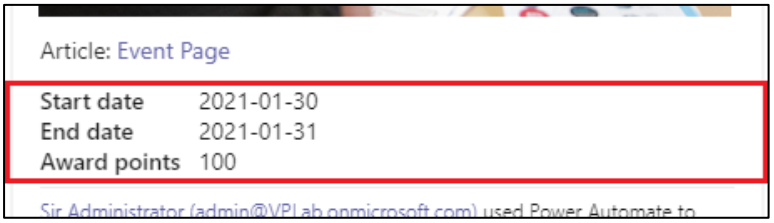


6. **FactSet** block easily displays key-value pairs.

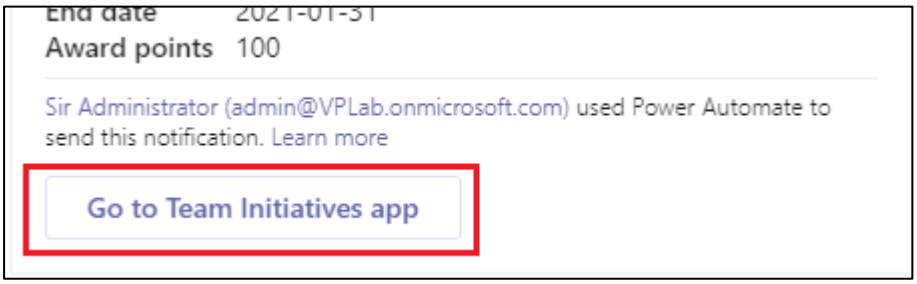
FactSet block

```
    {
            "facts": [
                {
                    "title": "Start date",
                    "value": "@{triggerOutputs()?['body/crXXX_startdate']}"
                },
                {
                    "title": "End date",
                    "value": "@{triggerOutputs()?['body/crXXX_enddate']}"
                },
                {
                    "title": "Award points",
                    "value": "@{triggerOutputs()?['body/crXXX_awardpoints']}"
                }
            ],
            "type": "FactSet"
    }
    ], // Body collection is closed
```

In flow, all fact values are pulled from corresponding **TeamActivity** record properties.



7. After **Body** collection is closed, you can notice **Actions** collection, that contains only one button in this case. This button simply redirects user to certain URL.



Task is completed.

Exercise 2:
Approval Flow

Objectives:

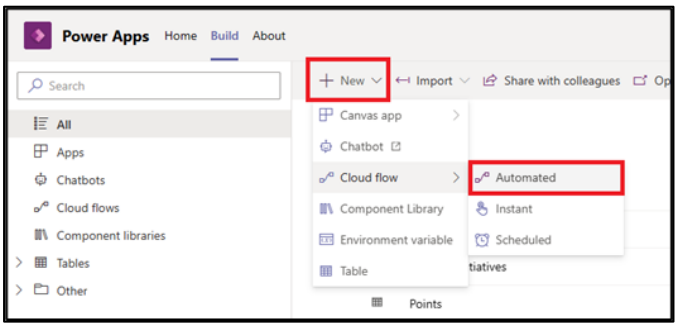
- Create an approval flow in Dataverse for Teams

Estimated time:

20 minutes

Task 1: Create an approval flow in Dataverse for Teams

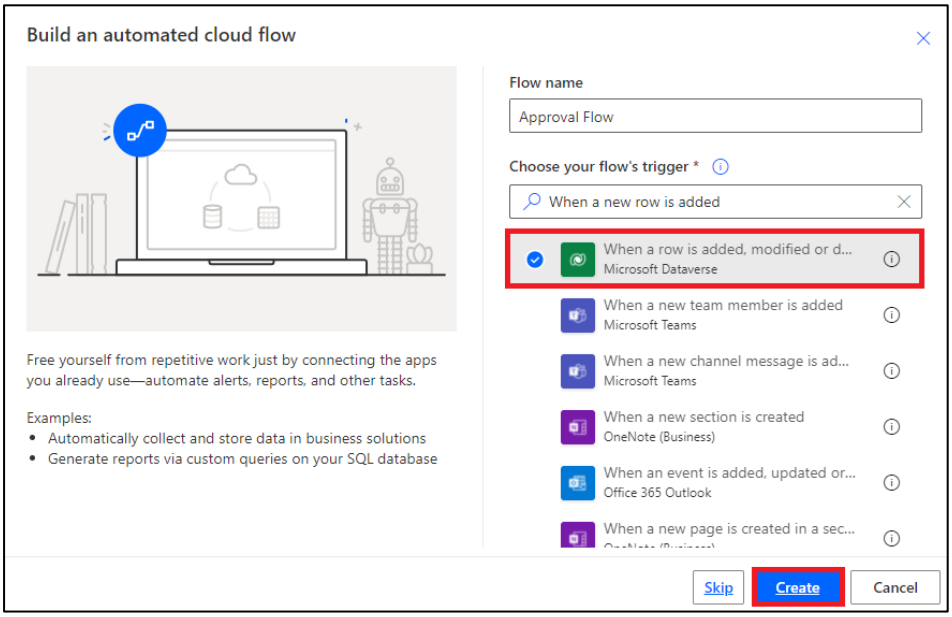
1. **Navigate to Dataverse for Teams content** of your Team
2. To create a new flow, click **+New -> Cloud Flow -> Automated**



3. Set **Flow name** as *Approval Flow*

   Search for *"When a row is added"*

   Select **When a row is added, modified, or deleted** trigger from **Microsoft Dataverse** connector
   Click **Create** button

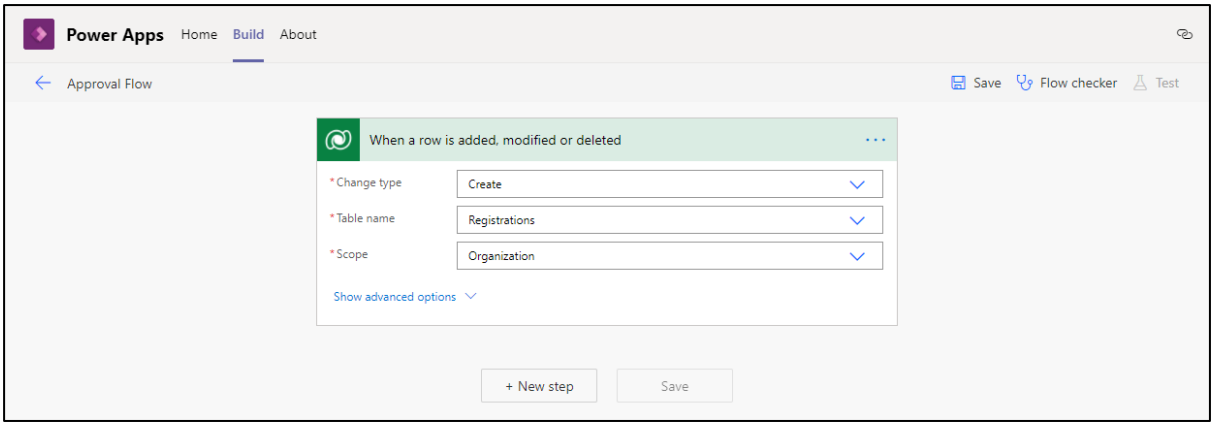


4. Set properties for the trigger:

   Change type: Create

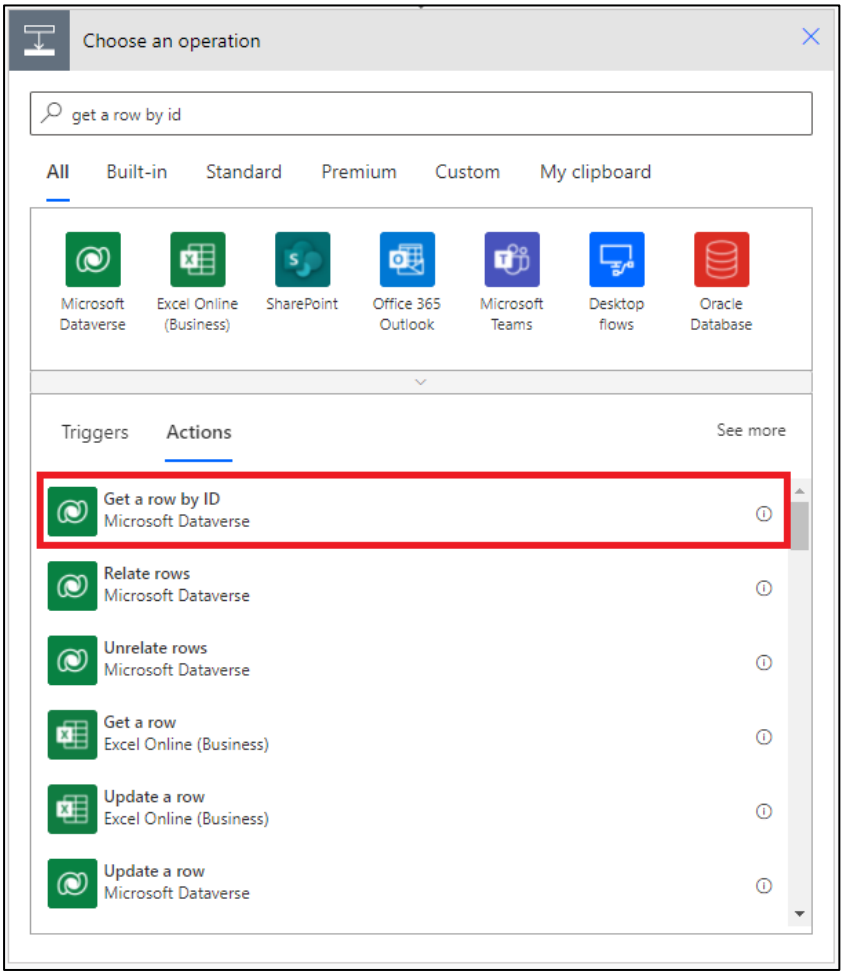
   Table name: Registrations

   Scope: Organization



5. To show more details about the related initiative, get a row from **TeamActivities** table

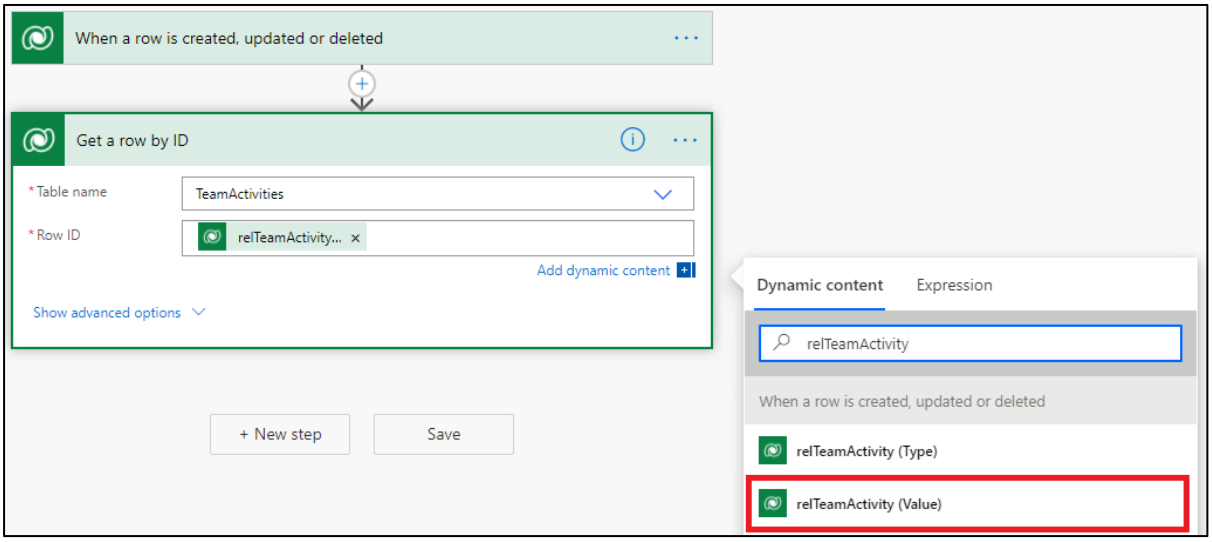
   Click **+New step** button
   Search for **Get a row by ID** action from **Microsoft Dataverse connector**



6. Set properties for **Get a row by ID** action as:

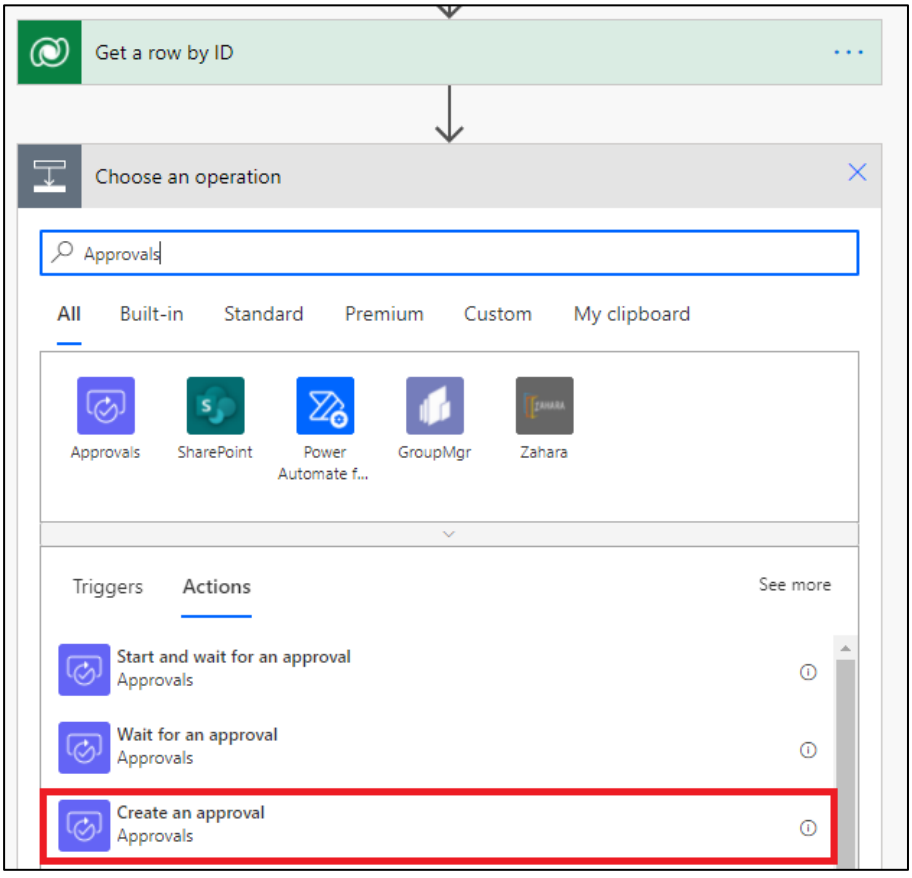
   Table name: TeamActivities

   Row ID: *relTeamActivity (Value)*



   This action outputs will contain the details of related **TeamActivity**.

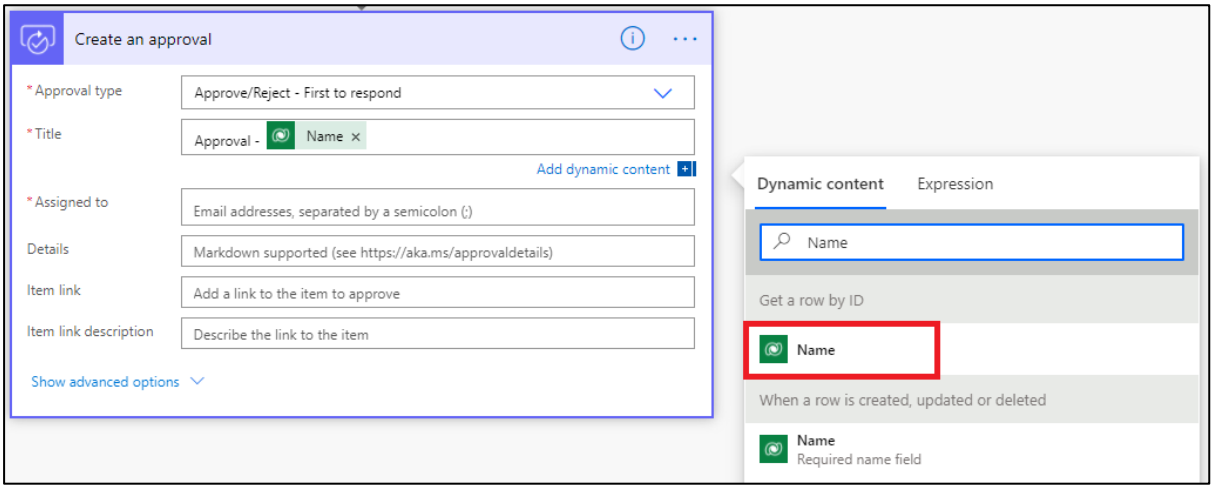
7. Click **+New step** button
   Select **Create an approval** action in **Approvals** connector by clicking on it (use Search, if this connector or action is not present)



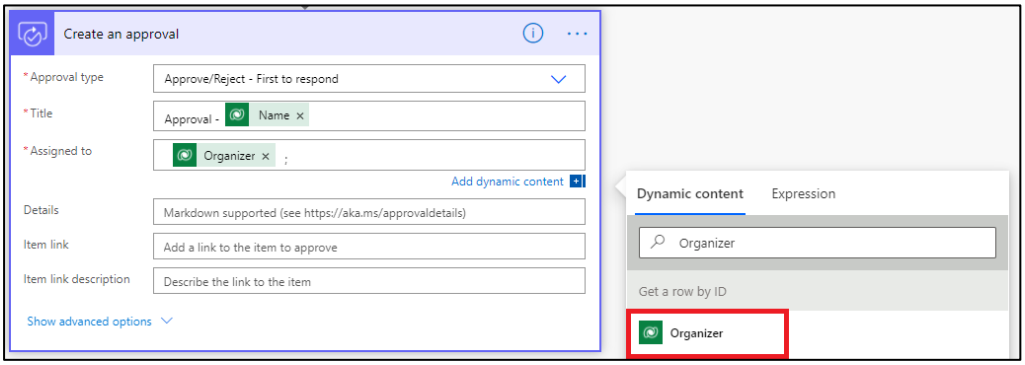
8. Set properties for **Create an approval** as:

   Approval type: Approve/Reject – First to respond

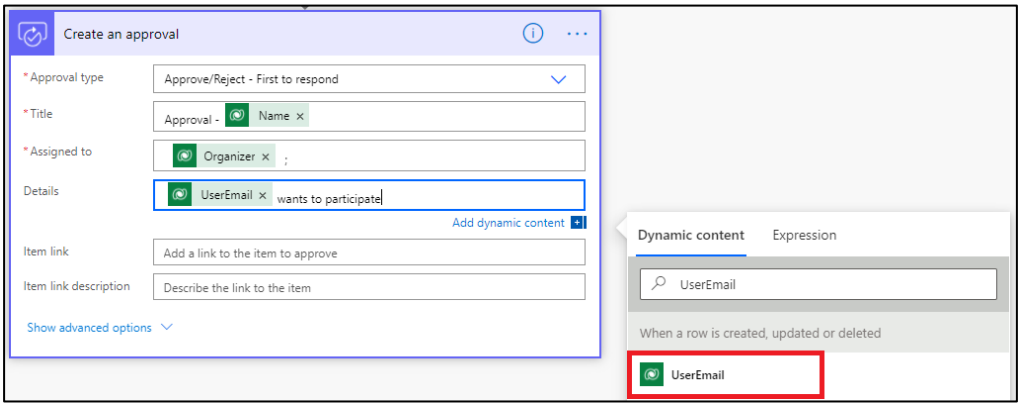
   Title: Approval – *Name (from Get a row by ID)*



   Assigned to: *Organizer (from Get a row by ID)*

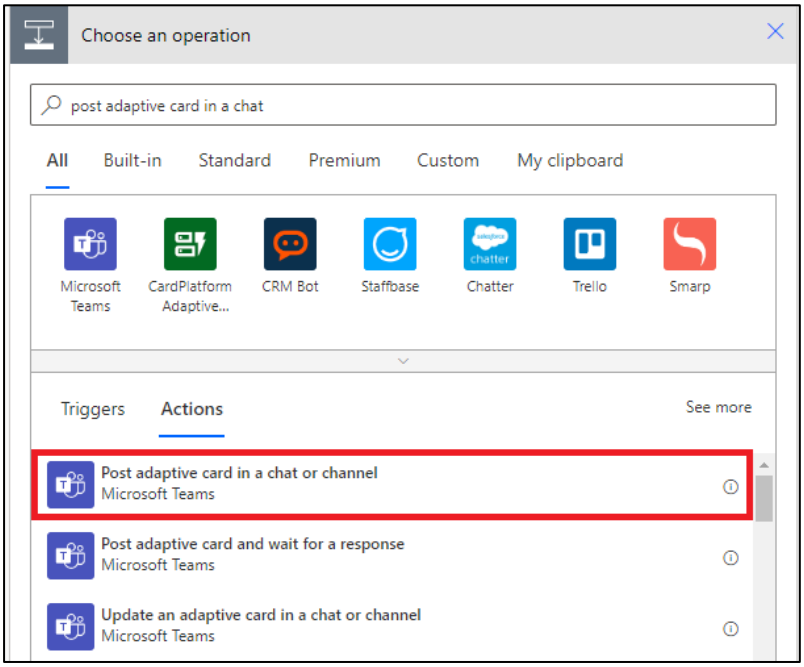


   Details: *UserEmail (from trigger)* wants to participate



9. Let's use an adaptive card as one more way for the user to approve request.
   Click **+New step** button
   Select **Post adaptive card in a chat or channel** action in **Microsoft Teams**

connector by clicking on it
(use Search, if this connector or action is not present)



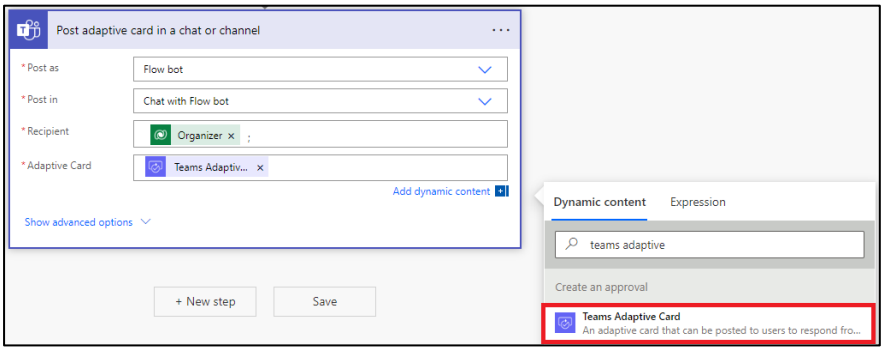
10. Luckily, the adaptive card was pre-created, when we started an approval.
    Set properties for **Post adaptive card in a chat or channel** as

    Post as: Flow bot

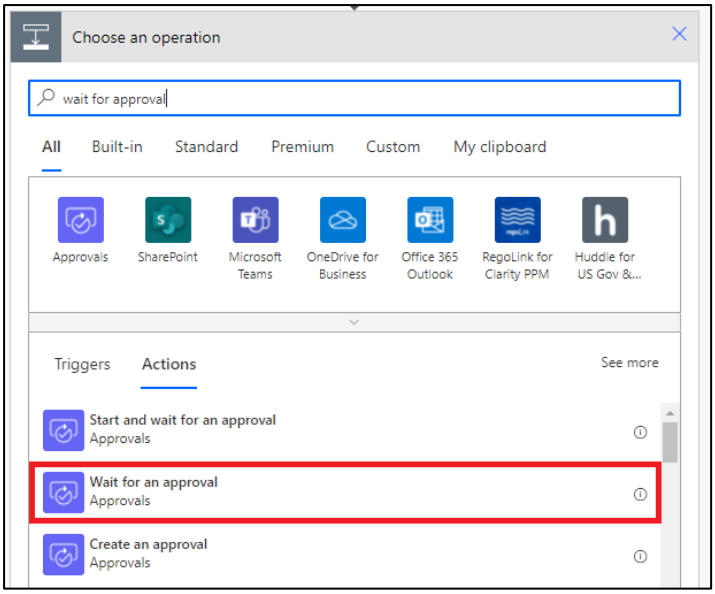
    Post in: Chat with Flow bot

    Recipient: *Organizer (from Get a row by ID)*

    Message: *Teams Adaptive Card (from Create an approval)*

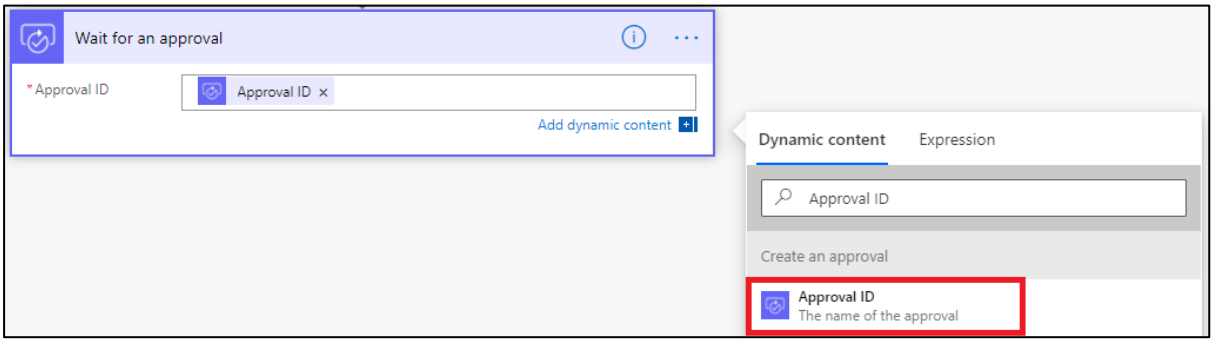


11. Click **+New step** button
    Select **Wait for approval** action in **Approvals** connector by clicking on it
    (use Search, if this connector or action is not present)

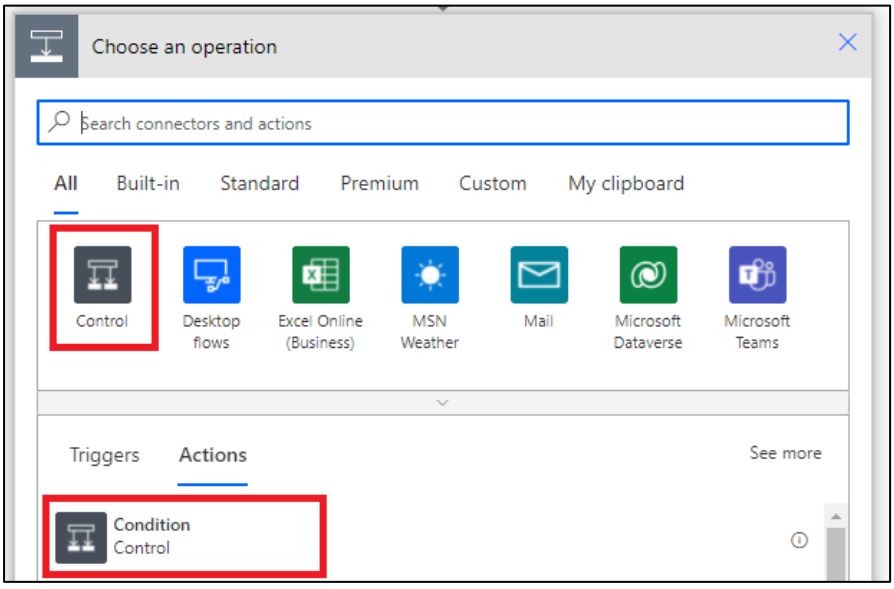


12. Set properties for **Wait for an approval** as:

    Approval ID: *Approval ID (from Create an approval)*

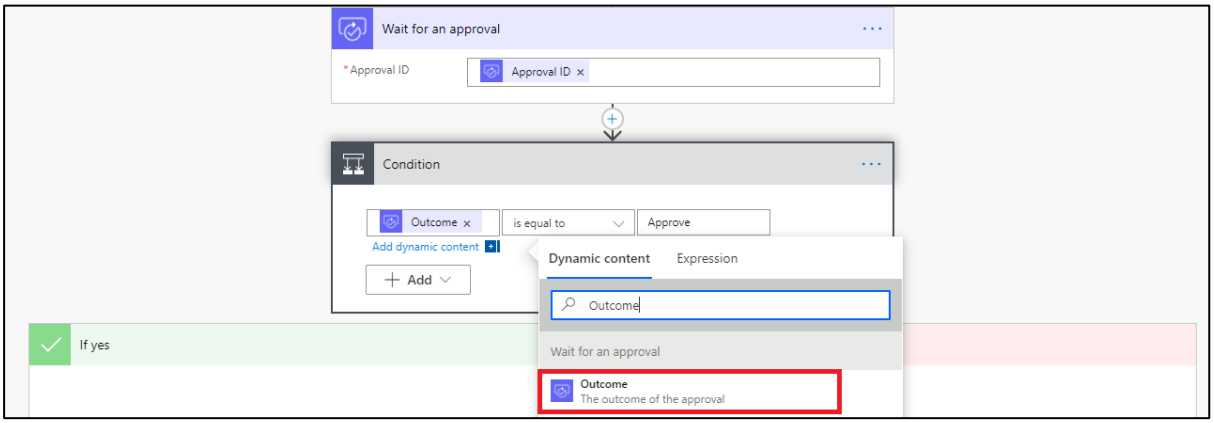


13. Click **+New step** button
    Select **Condition** action in **Control** "connector" by clicking on it (use Search, if this connector or action is not present)

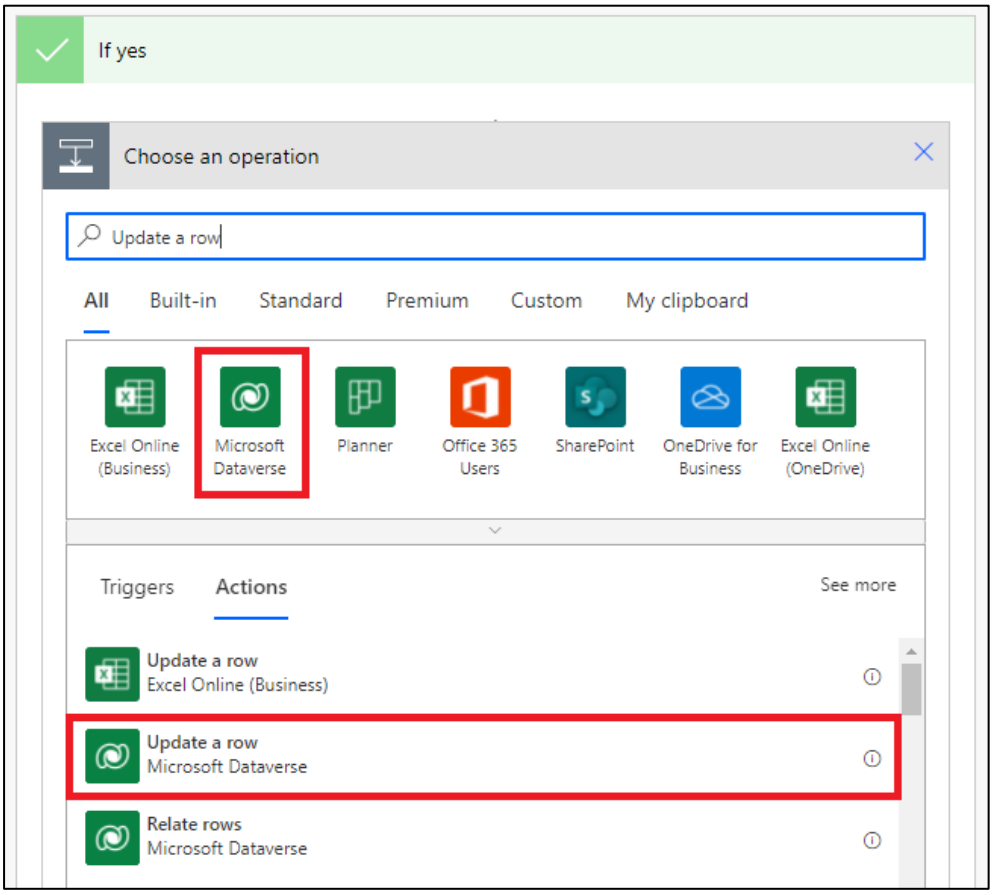


14. Set **Condition** as *Outcome (from Wait for an approval)* is equal to **Approve**

    Because of **Approval type** "Approve/Reject – First to respond", **Outcome** can contain one of two string values "Approve" or "Reject"



15. In **If yes** scope, click **Add an action** button
    Select **Update a row** action in **Microsoft Dataverse** connector (use Search, if this connector or action is not present)

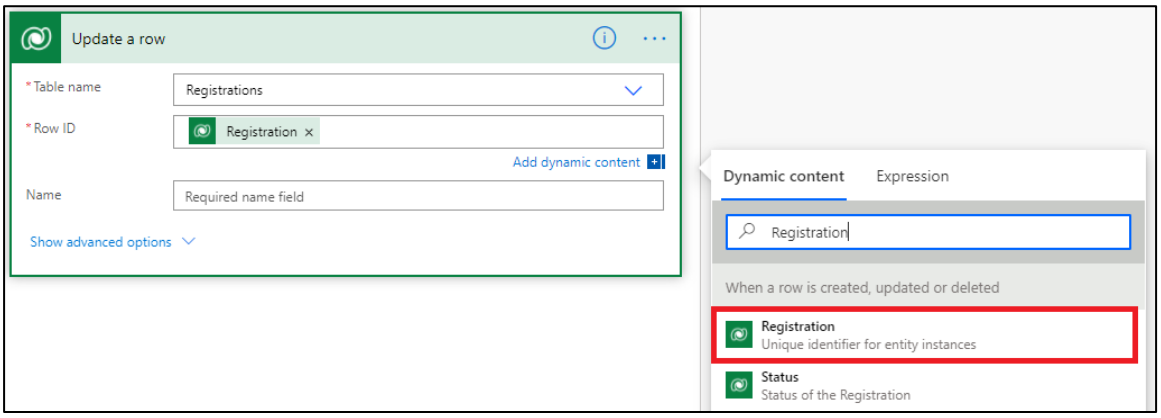


16. Let's confirm user's participation by updating **Registration** values.

    Set properties for **Update a row** action as:

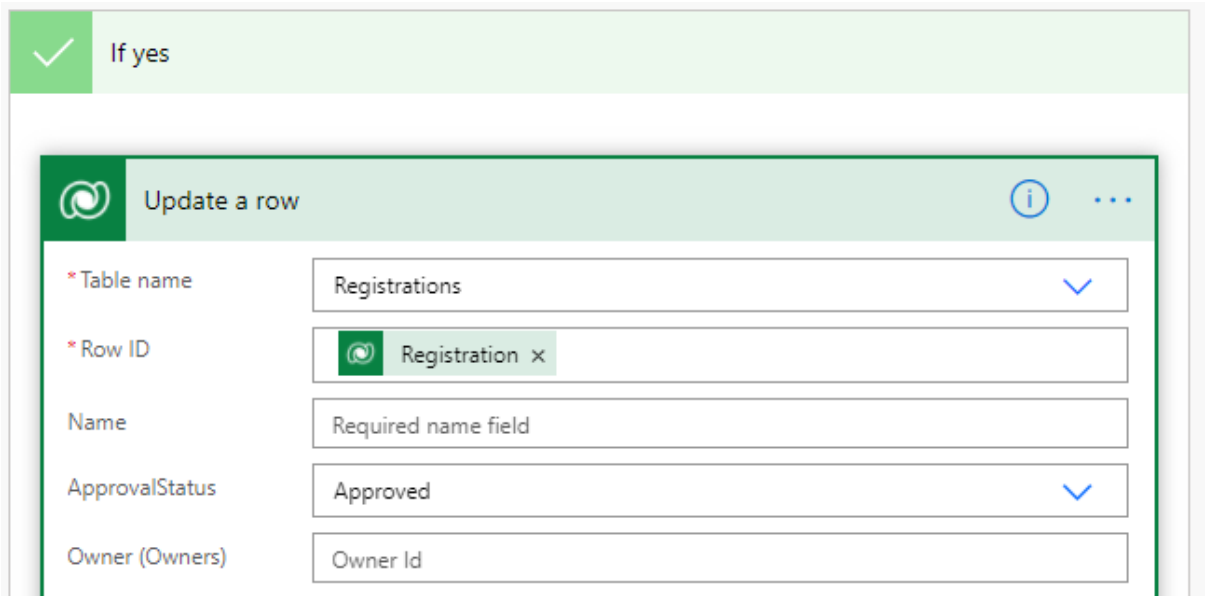
    Table name: Registrations

    Row ID: *Registration (from trigger)*



    Expand **Show advanced options**

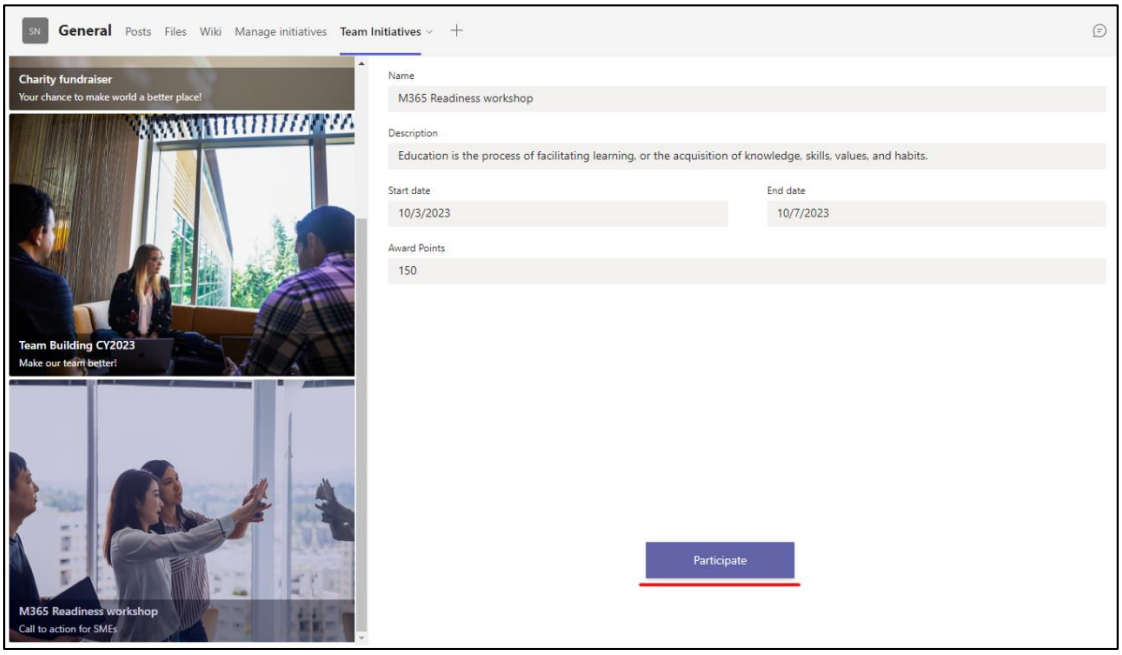
    ApprovalStatus: Approved



    Please, do not change any other fields, if you don't want to change their values – empty fields won't be updated at all.

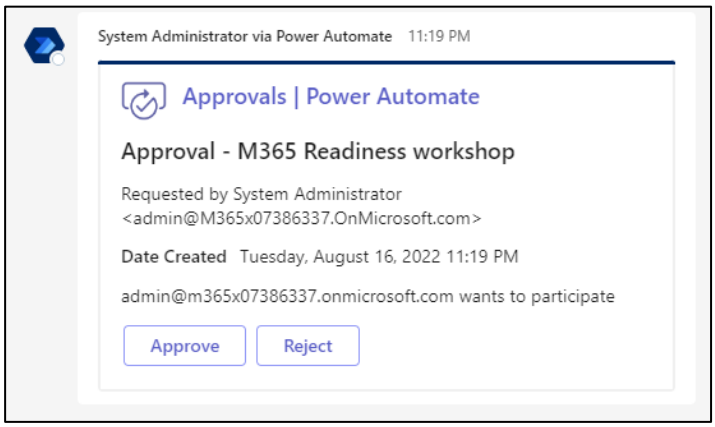
17. Make sure, that there are no notifications in **Flow checker** and **Save** this flow

18. In **Team Initiatives app**, click **Participate** for any initiative.



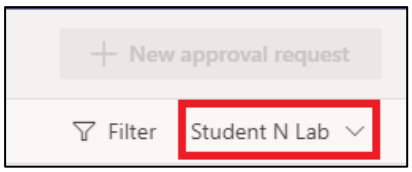
19. Give some time for flow to start, then check an approval request in:

    a. **Adaptive cards** in a chat (we sent it in this flow)

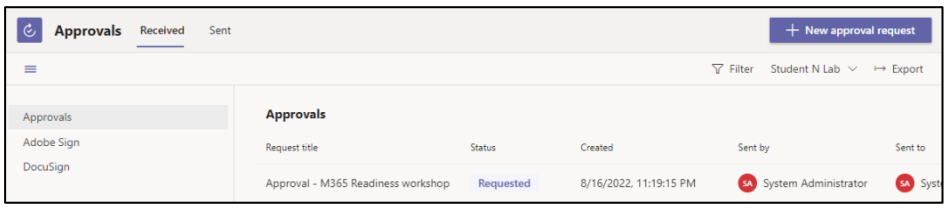


    b. Or use **Approvals** teams app

    Select your **Dataverse for Teams** environment



    Click on the approval to check its details



    c. Check **Office 365 Outlook** mailbox for an approval request

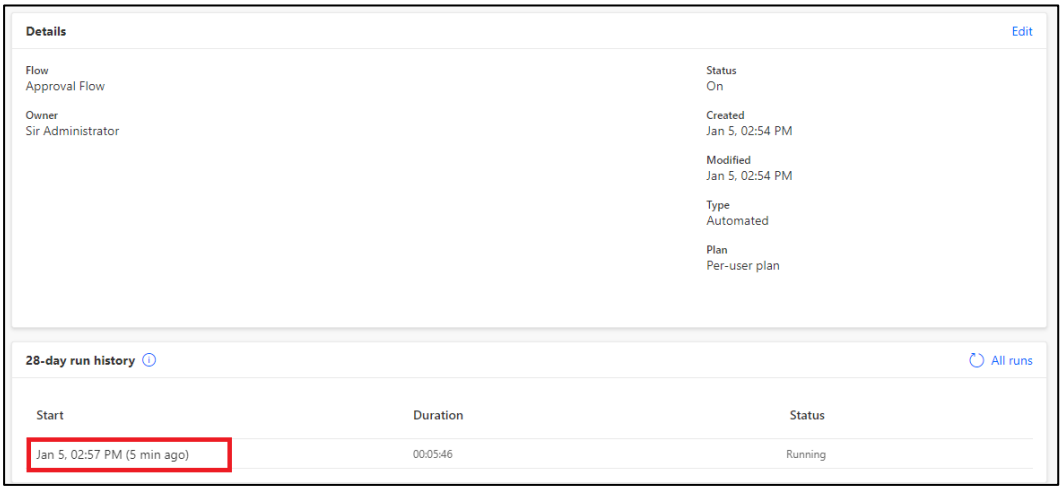
20. Approve the request and go back to **Team Initiatives app**
    Check if registration status changed to **Approved**

21. If the flow doesn't seem to work, then
    navigate to **Dataverse for Teams content of your Team**

    Select **Cloud Flows** tab to see all the flows in the environment

    Select **Approval Flow**

    Investigate flow issues by clicking on the flow start time in **28-day run history**



> **If the flow is running, but no Approvals received**
>
> It may mean that Approvals provisioning in a new environment is not fully completed yet.
>
> Flow Checker will display warning: "The approvals database provisioning job is still in progress" – so give it some time for the first run.

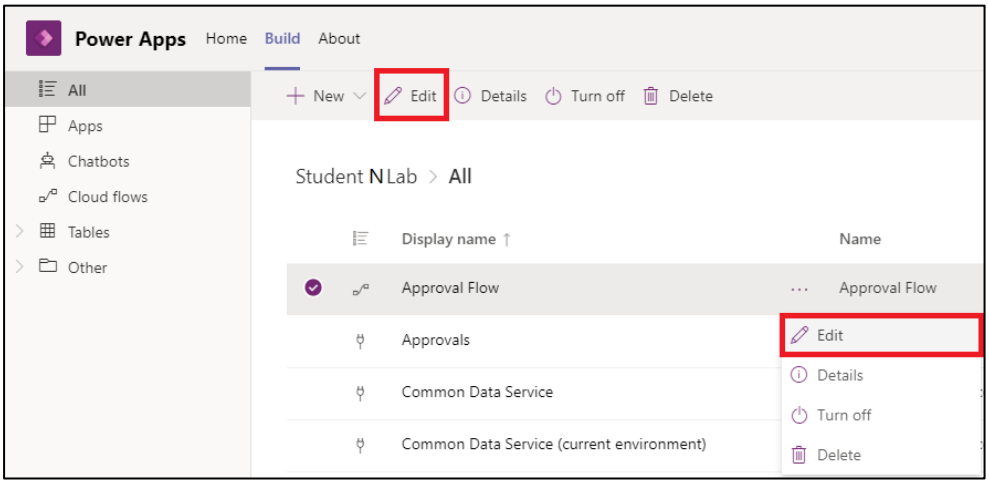
Task is completed.

Optional task: Query Dataverse tables to update Points balance

Prerequisites required

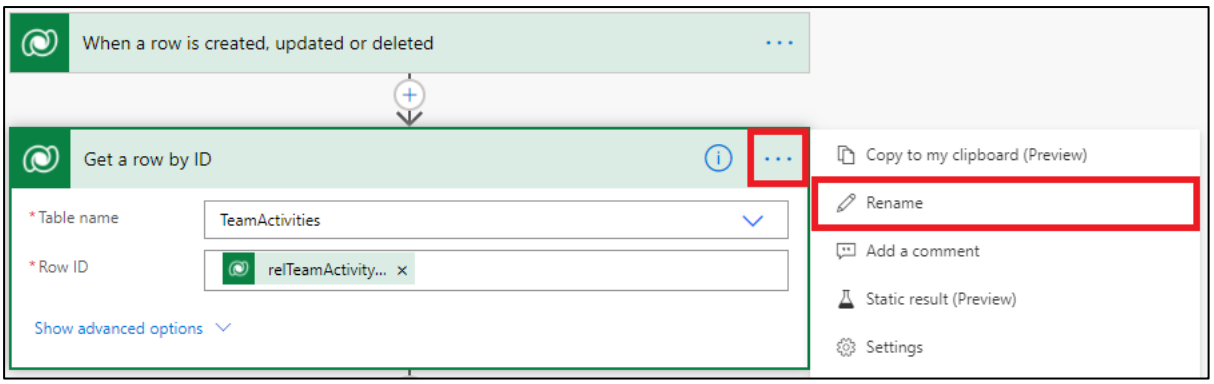
Other optional tasks are prerequisites for this task:
- Points table must exist in order to complete this task (Lab 2 Exercise 1)

1. **Navigate to Dataverse for Teams content** of your Team
2. Select **Approval Flow** and click **Edit** on toolbar or in the context menu

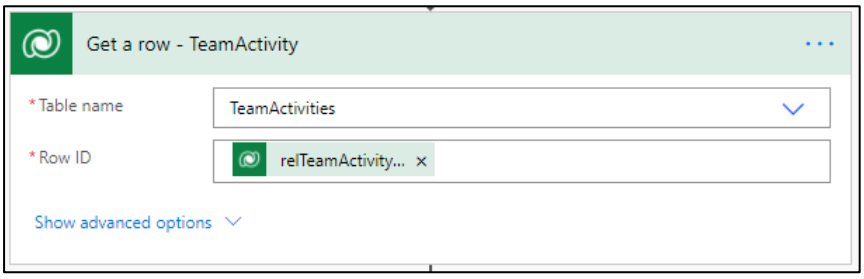


3. It is a good practice to rename actions of the same type in a flow to avoid ambiguity, when building complex flows.

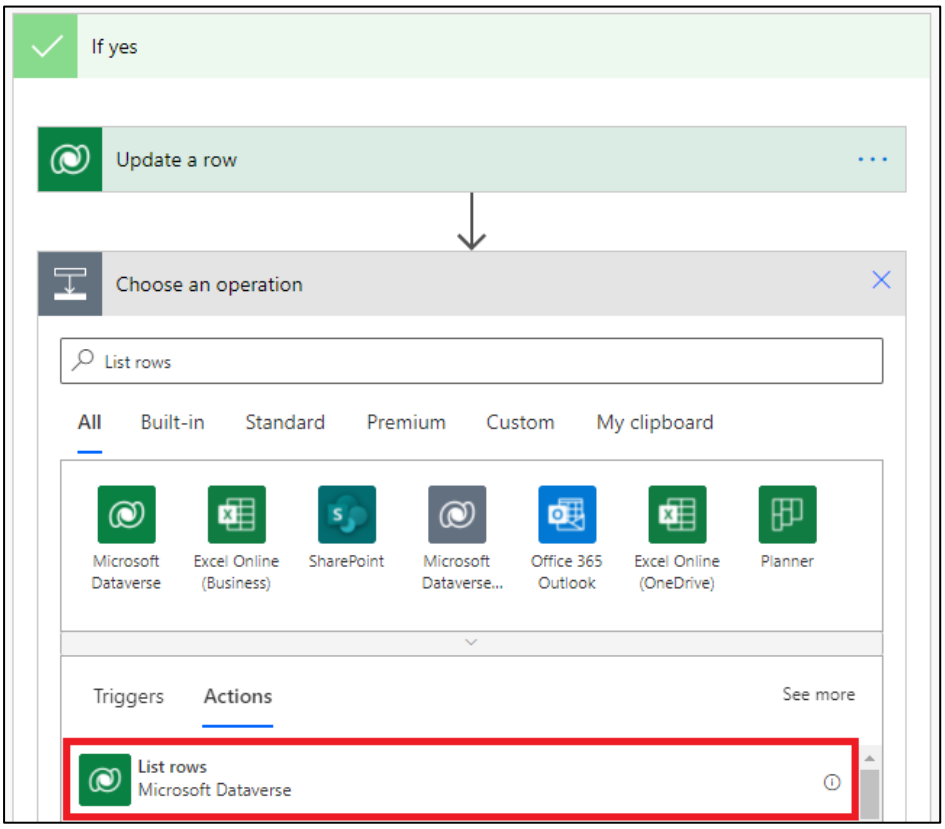
   Find **Get a row by ID** action and click **ellipsis(...)** in the right corner to open a context menu of the action, select **Rename**



4. Set action name as **"Get a row – TeamActivity"**
   It will help you easily recognize dynamic outputs of this action when you build a flow.



5. In **If yes** scope, click **Add an action** button
   Select **List rows** action in **Microsoft Dataverse** connector by clicking on it (use Search, if this connector or action is not present)



6. **List rows** is a very special action, that allows to query Dataverse tables using OData or FetchXML queries.

OData and FetchXML queries

OData stands for the Open Data Protocol.
It's a standard that defines practices for building and consuming RESTful Web APIs.

In Power Automate, OData queries can help you to retrieve only the data that you need from Dataverse tables using List Rows action.

Developer reference:
Query Data using the Web API (Microsoft Dataverse) - Power Apps | Microsoft Docs

---

FetchXML is a proprietary query language that is used in Dataverse. It can be handy for experienced Dynamics developers, or when you build especially complex queries.

Developer reference:
Use FetchXML to query data (Microsoft Dataverse) - Power Apps | Microsoft Docs

List rows returns a collection of objects that may be empty, contain one or more objects.

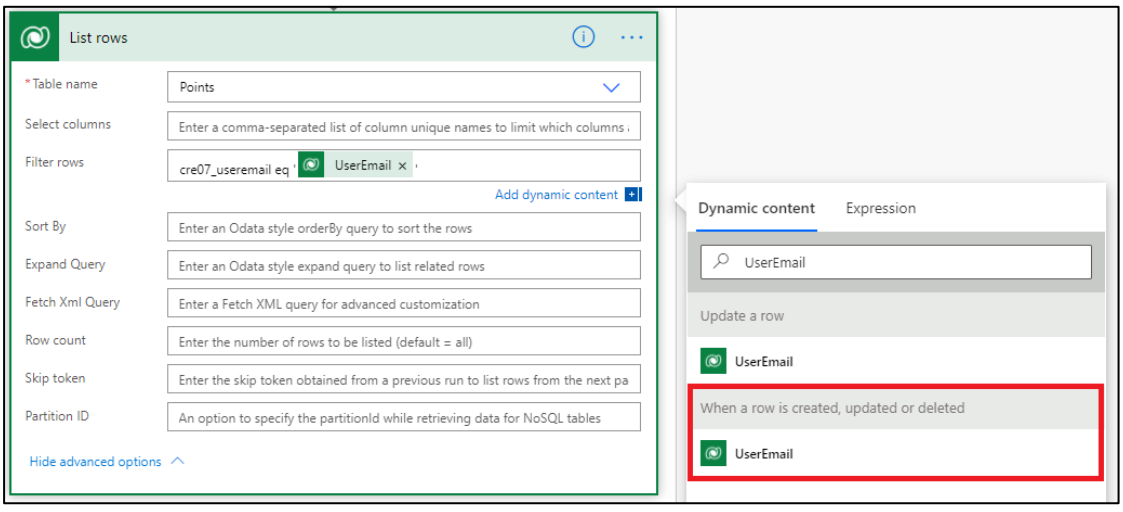
Set properties for **List rows** action as:

Table name: Points

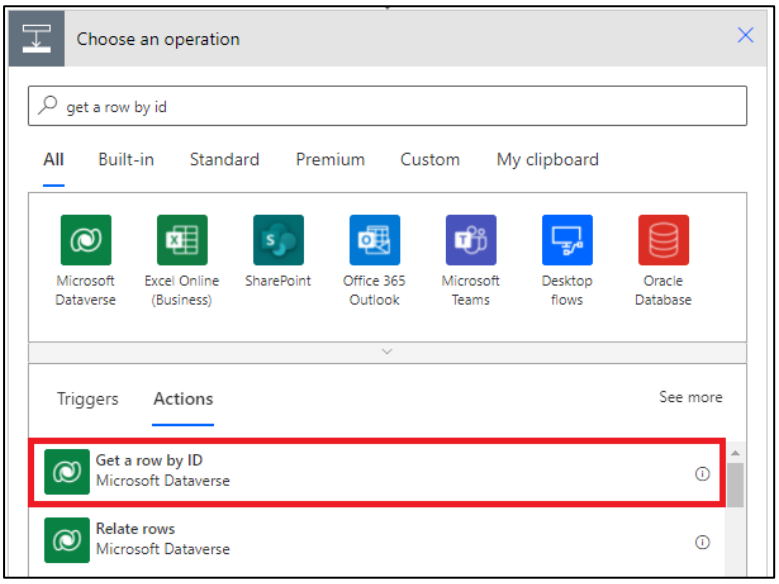
Filter rows: crXXX_useremail eq '*UserEmail*' *(from trigger)*

crXXX is a prefix (publisherId) the same as in **Exercise 1 Task 1**.

Pay attention that column name is written in lower case and there are single quotes around UserEmail (from trigger). Double-check syntax.



7. After **List rows** add an action and select **Get a row by ID** action from in **Microsoft Dataverse** connector by clicking on it
(use Search, if this connector or action is not present)

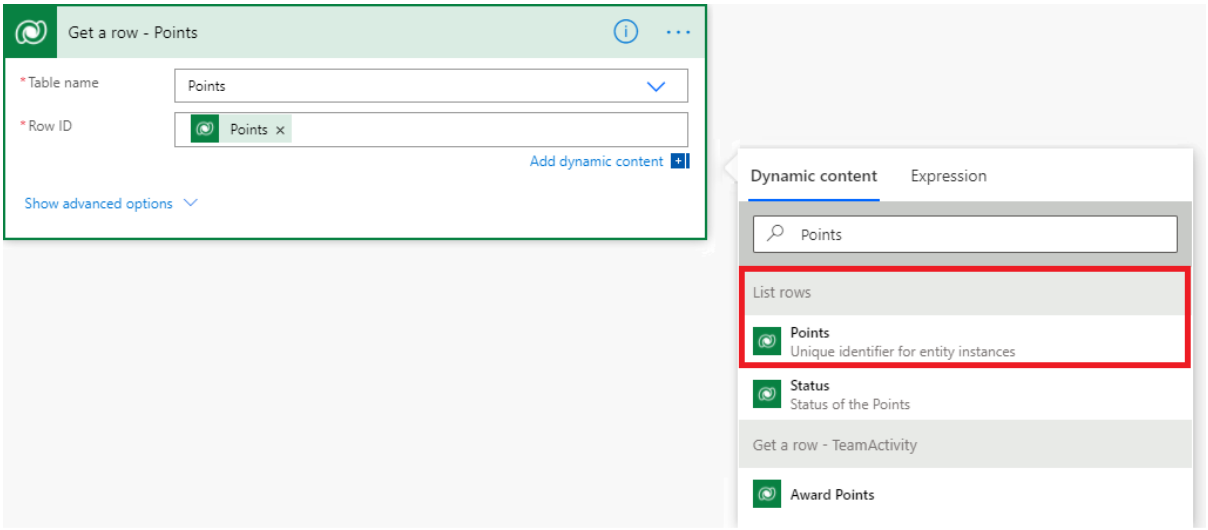


8. Rename action as **"Get a row – Points"**

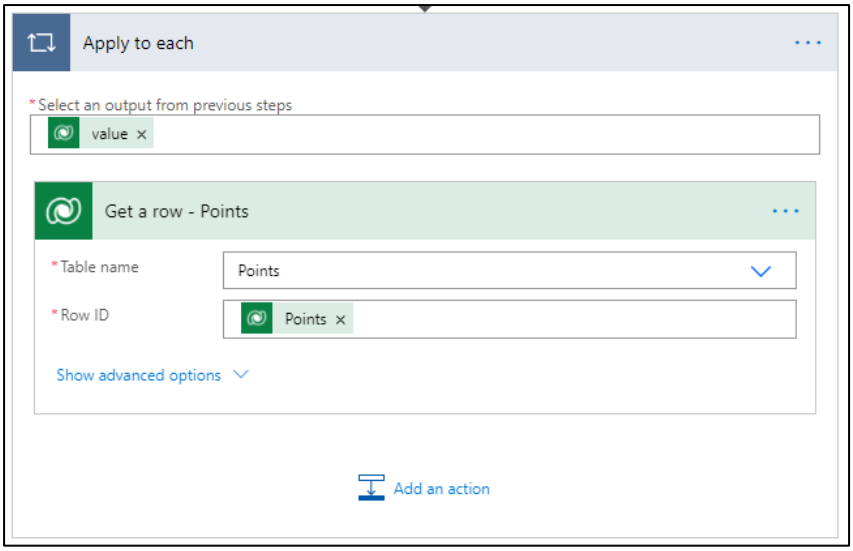
   Set properties for **Get a row - Points** action as:

   Table name: Points

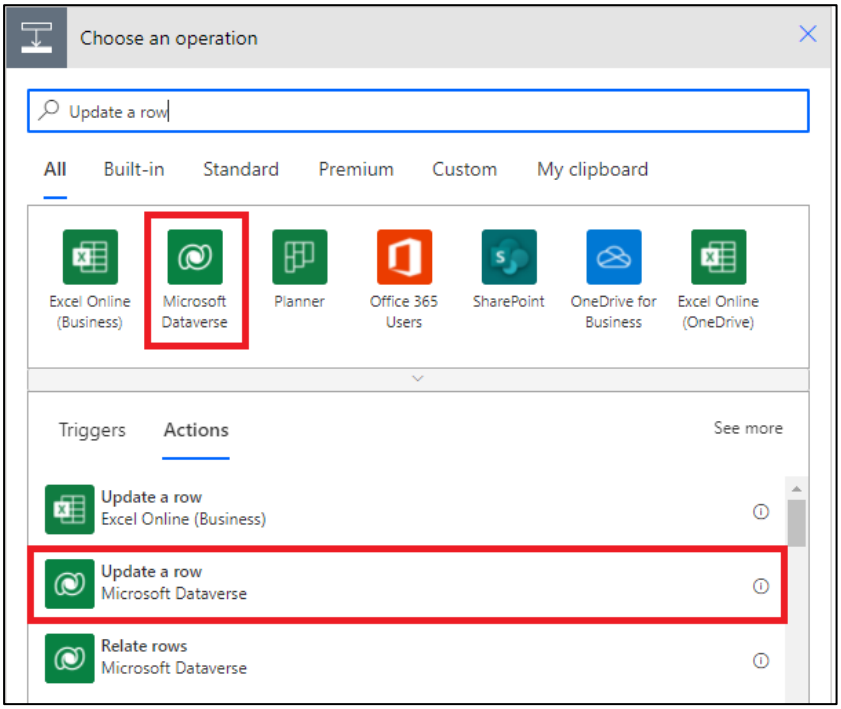
   Row ID: Points *(from List rows)*



9. Notice that **Apply to each** loop was automatically added.
   It happened, because, even if **List rows** action has returned a single entry as we expect – it's still a single entry in a collection of objects.
   In this case this loop will work just once.



10. **Add an action** within **Apply to each** loop.
    Select **Update a row** action in **Microsoft Dataverse** connector (use Search, if this connector or action is not present)



11. Set properties for **Update a row 2** action as:

    Table name: Points

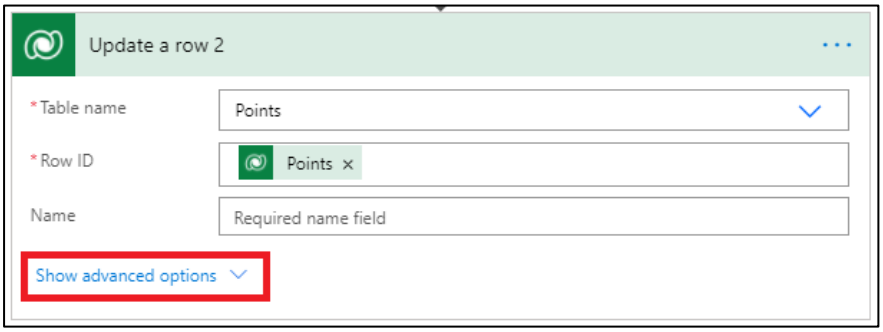
    Row ID: *Points (from Get a row - Points)*

12. Now it's time to TeamActivity add award points to user's points balance.

    Balance (new) = Balance (current) + Award Points (of TeamActivity)

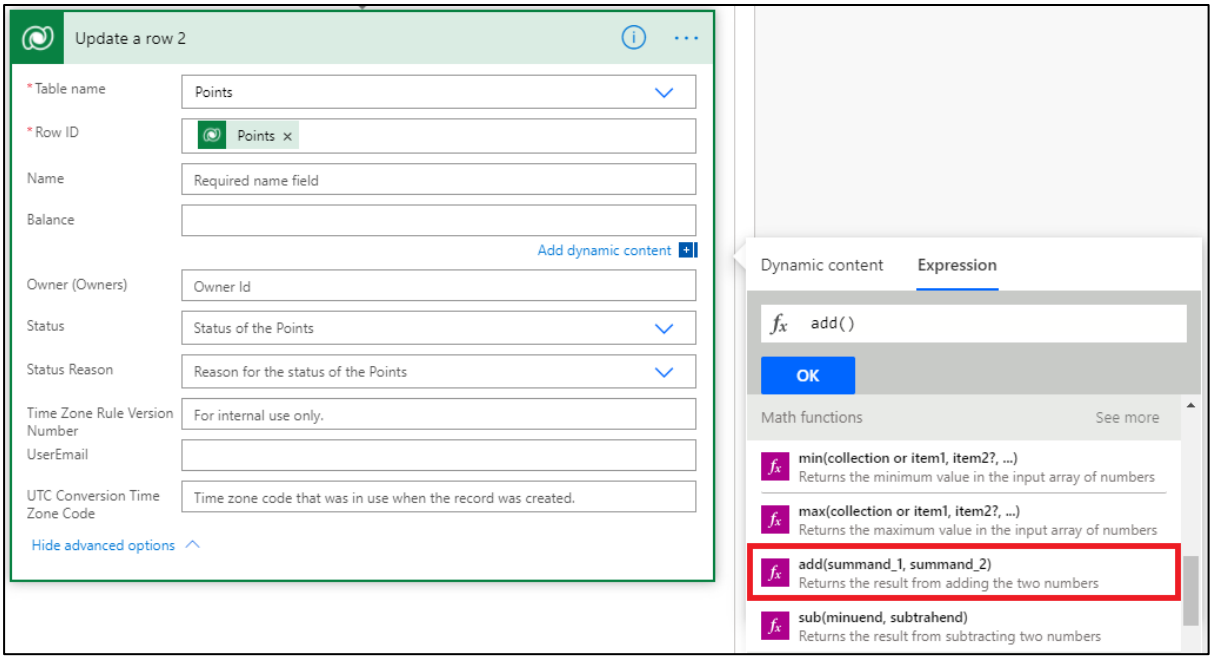
In Power Automate expression language (WDL),
to get sum we need to use *add(operand1, operand2)* function.

Click **Show advanced options** in **Update a row 2** to expand action properties.

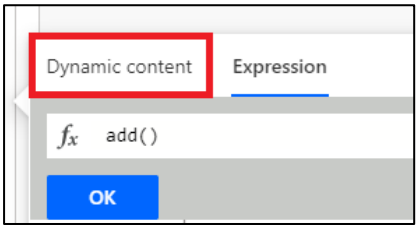


13. Select **Balance** property and in popup switch to **Expression** tab

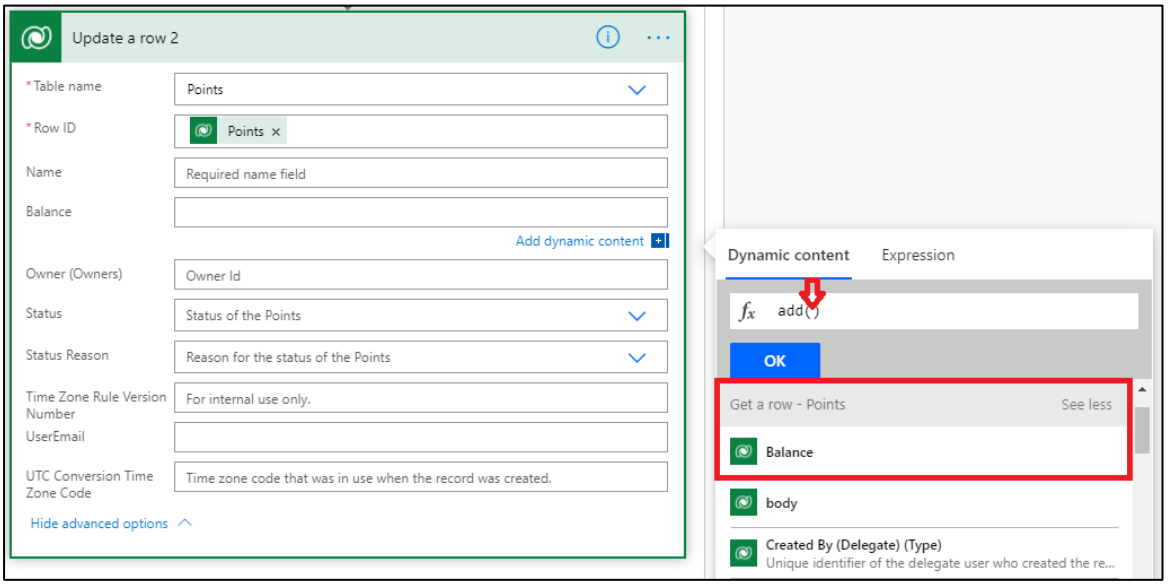
    Type *add()* into expression bar or select *add()* function in Math functions.



14. Switch back to **Dynamic content** tab



15. Put cursor between brackets in *add()* and select *Balance (from Get a row – Points*) in menu below (click See more, if you don't see Balance)



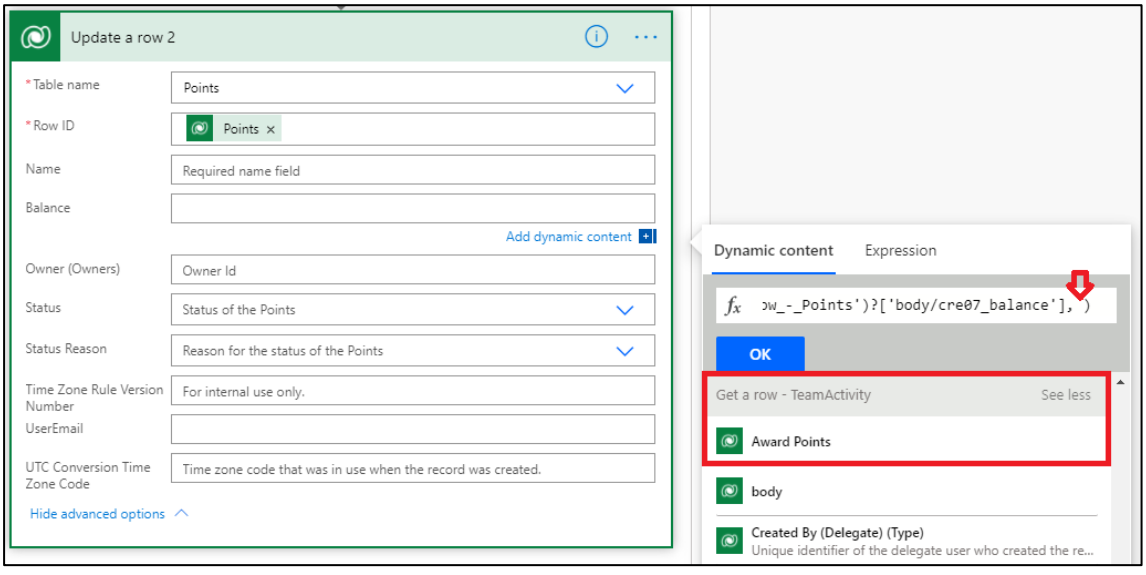
Now expression should look like:

add(outputs('Get_a_row_-_Points')?['body/crXXX_balance'])

16. Add comma before the closing bracket:

add(outputs('Get_a_row_-_Points')?['body/crXXX_balance'],)

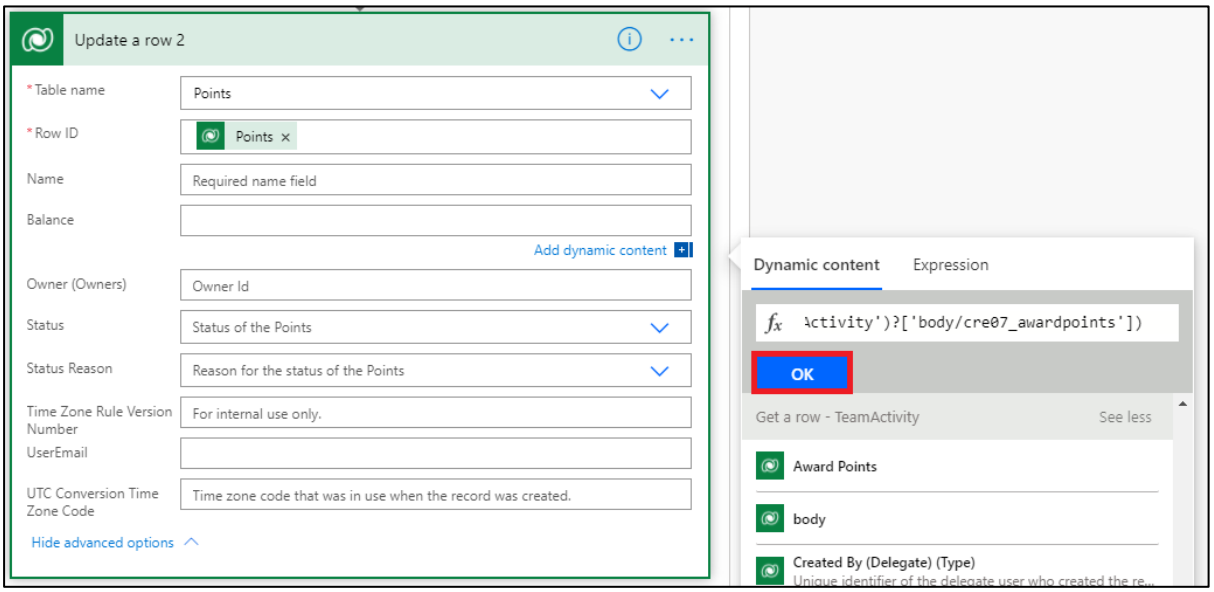
17. Put cursor between after brackets in *add()* and select *Award Points (from Get a row – TeamActivity*) in menu below (click See more, if you don't see Award Points)



18. Resulting expression should have both needed parameters to get sum from *add(operand1, operand2)* look like:

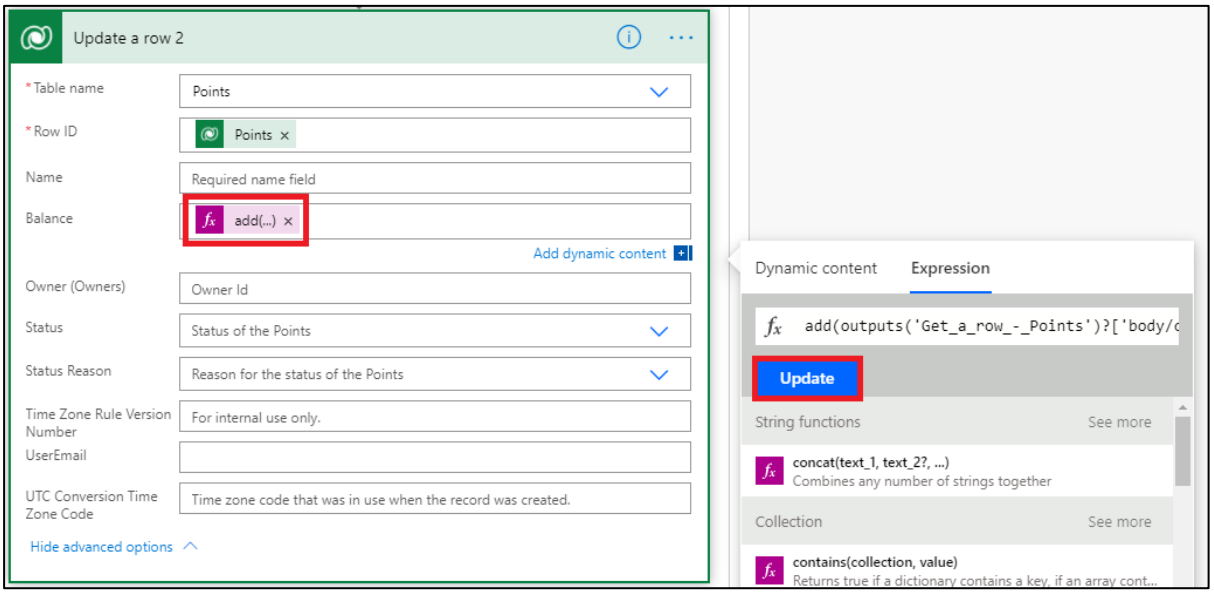
    add(outputs('Get_a_row_-_Points')?['body/crXXX_balance'], outputs('Get_a_row_-_TeamActivity')?['body/crXXX_awardpoints'])

    Click **OK** button



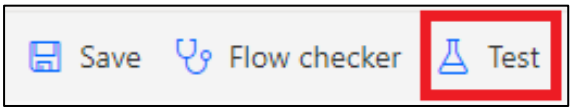
19. Double-check that expression is valid and added to Balance field in **Update a row 2.**

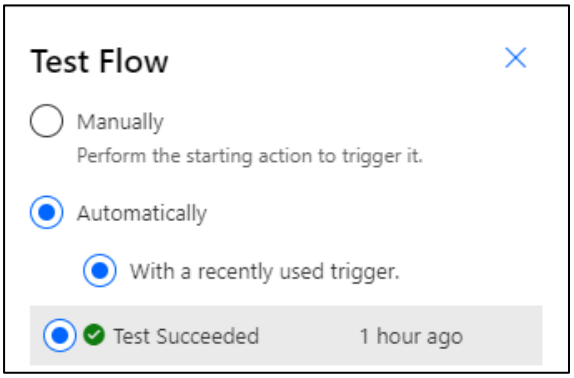
    If you need to change syntax of expression, just click on it, do changes, and click **Update** button.



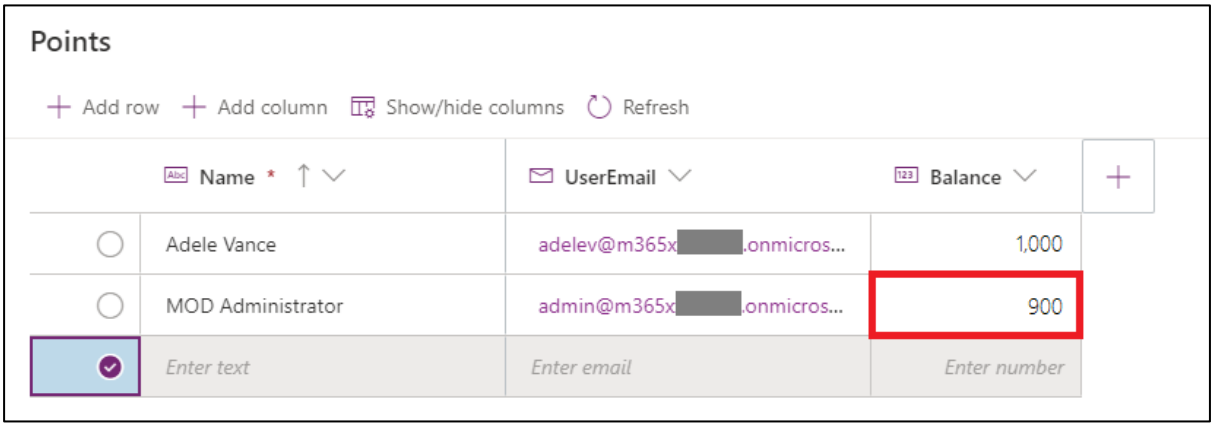
20. Make sure, that there are no notifications in **Flow checker** and **Save** this flow

21. Click **Test**, select **Automatically**, and then select one of previous successful runs





22. **Approve the request (like in Exercise 2 Steps 15-17)**
Navigate to Dataverse for Teams content of your Team and check **Points** table.
Balance of **Administrator** should increase from 750 to a different value.



Task is completed.