TRAINOCATE

TRAINOCATE

Microsoft Solutions Partner

WELCOME TO

Class: Power Automate (Advanced) Intensive Workshop

5 Nov 2025

9.00AM – 4.00PM

Instructor: K. Phakkhaphong Krittawat

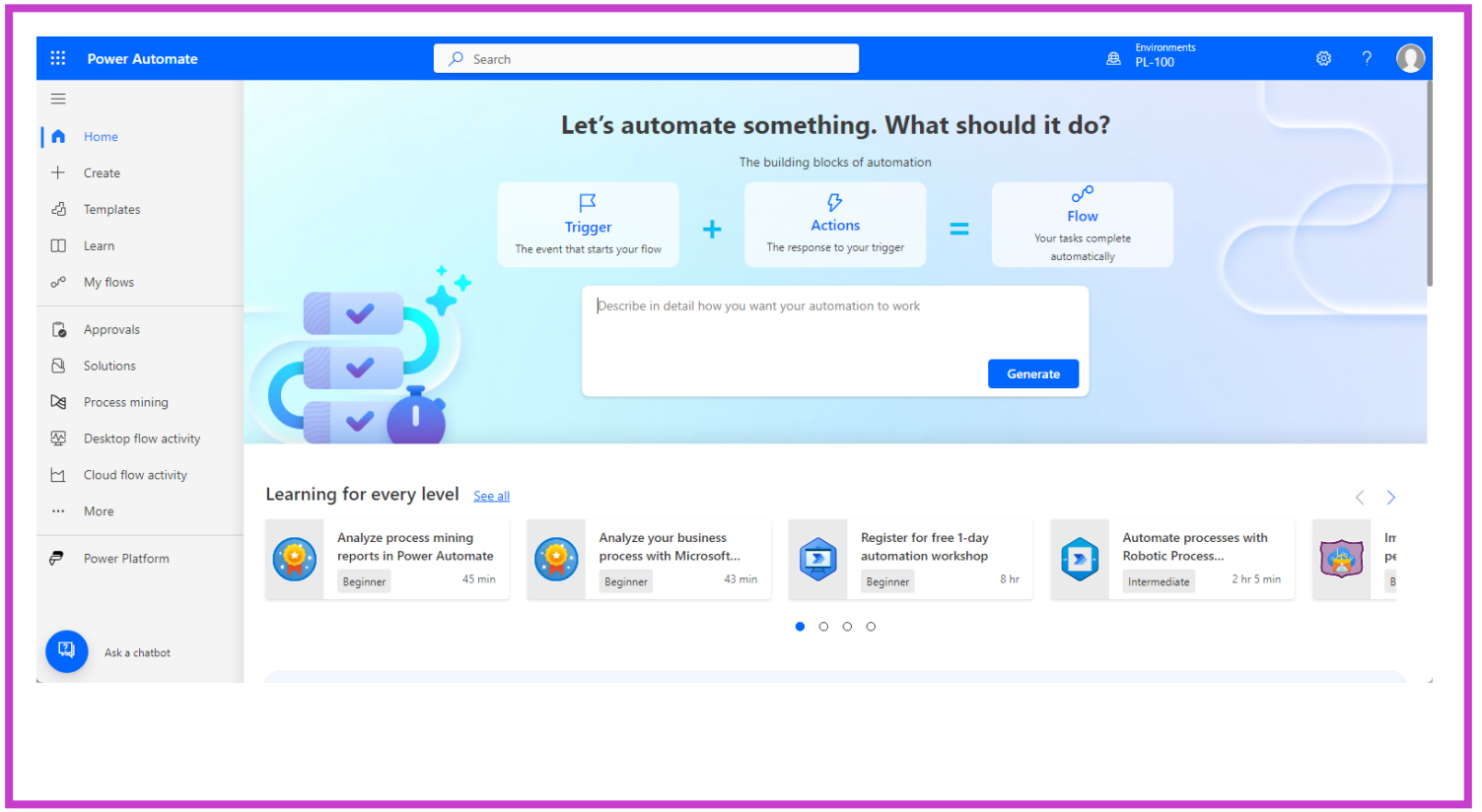
TRAINOCATE

Module 1

Building Flows from Scratch

What is Power Automate?

Power Automate is an online workflow service that automates actions across the most common apps and services



TRAINOCATE

What can you do with Power Automate?

Power Automate automates workflows between your favorite applications and services, sync files, get notifications, collect data, and more

For example, you can automate these tasks:

- Instantly respond to high-priority notifications or emails
- Capture, track, and follow up with new sales leads
- Copy all email attachments to your OneDrive for Business account
- Collect data about your business, and share that information with your team
- Automate approval workflows

Key concepts

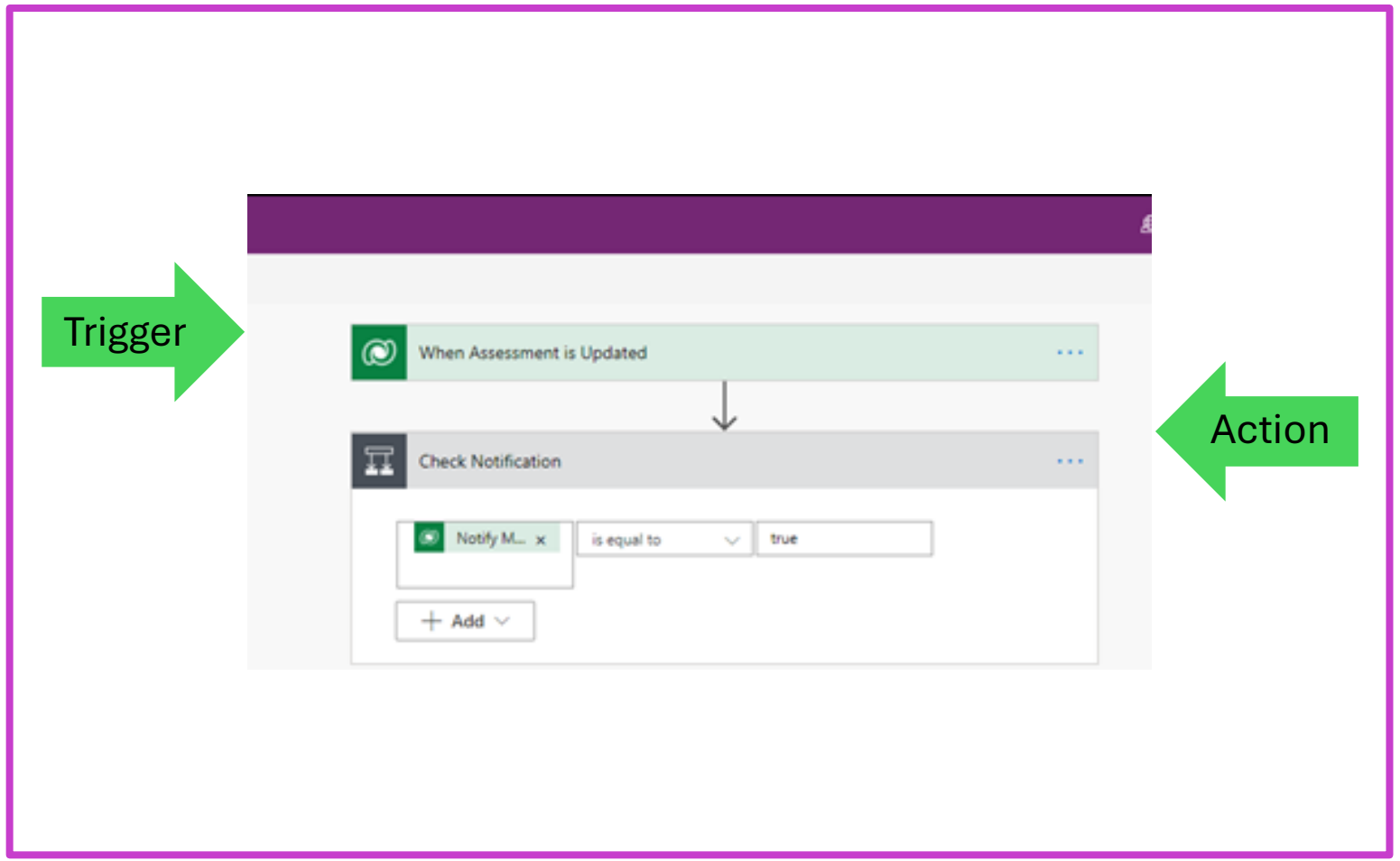
Every cloud flow has two main parts:

A Trigger

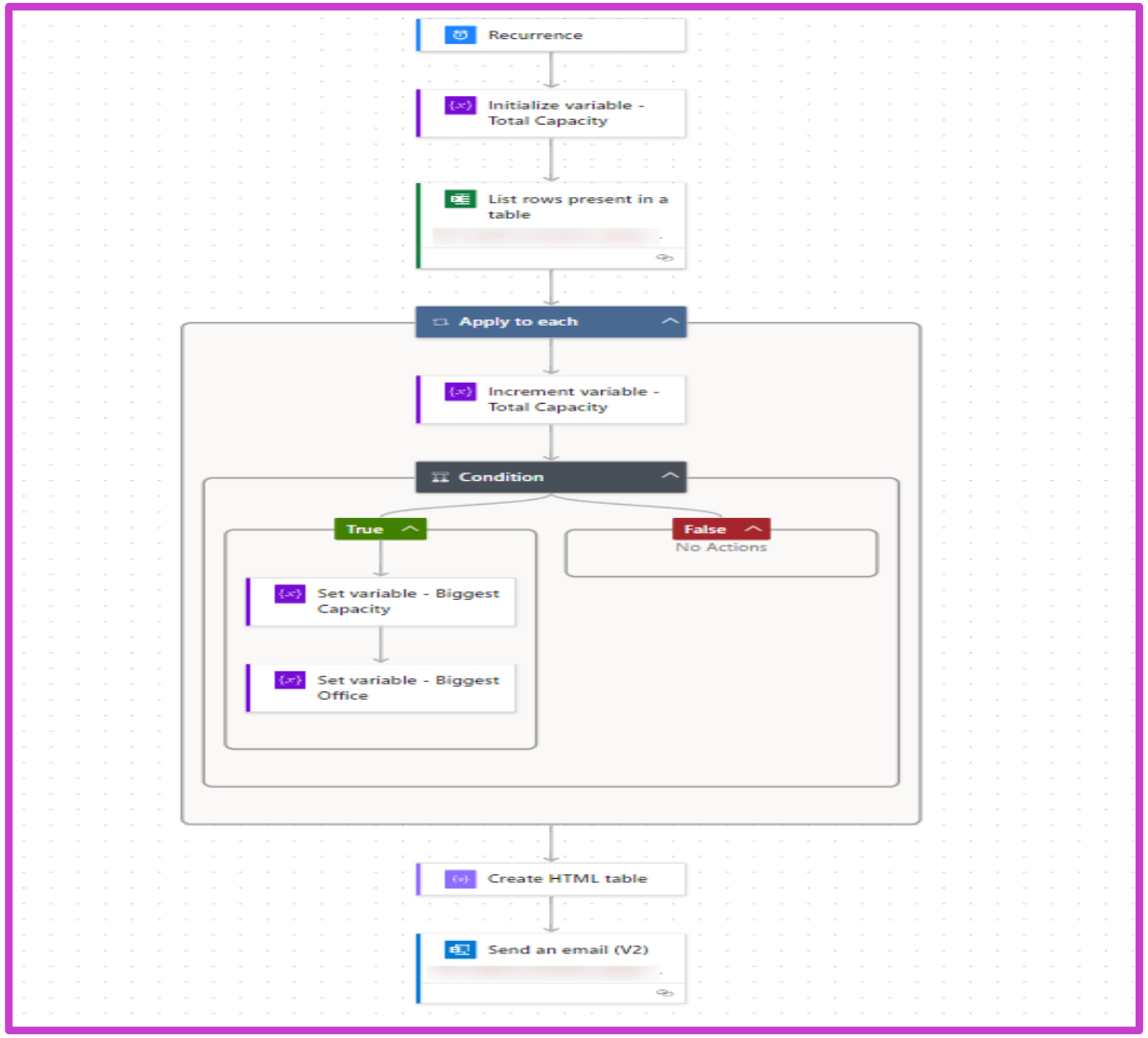
The starting action of the flow such as new email arriving in your inbox or a new item being added to a SharePoint list

One or more Actions

What you want to happen when a trigger is invoked such as start the action of creating a new file on OneDrive for Business

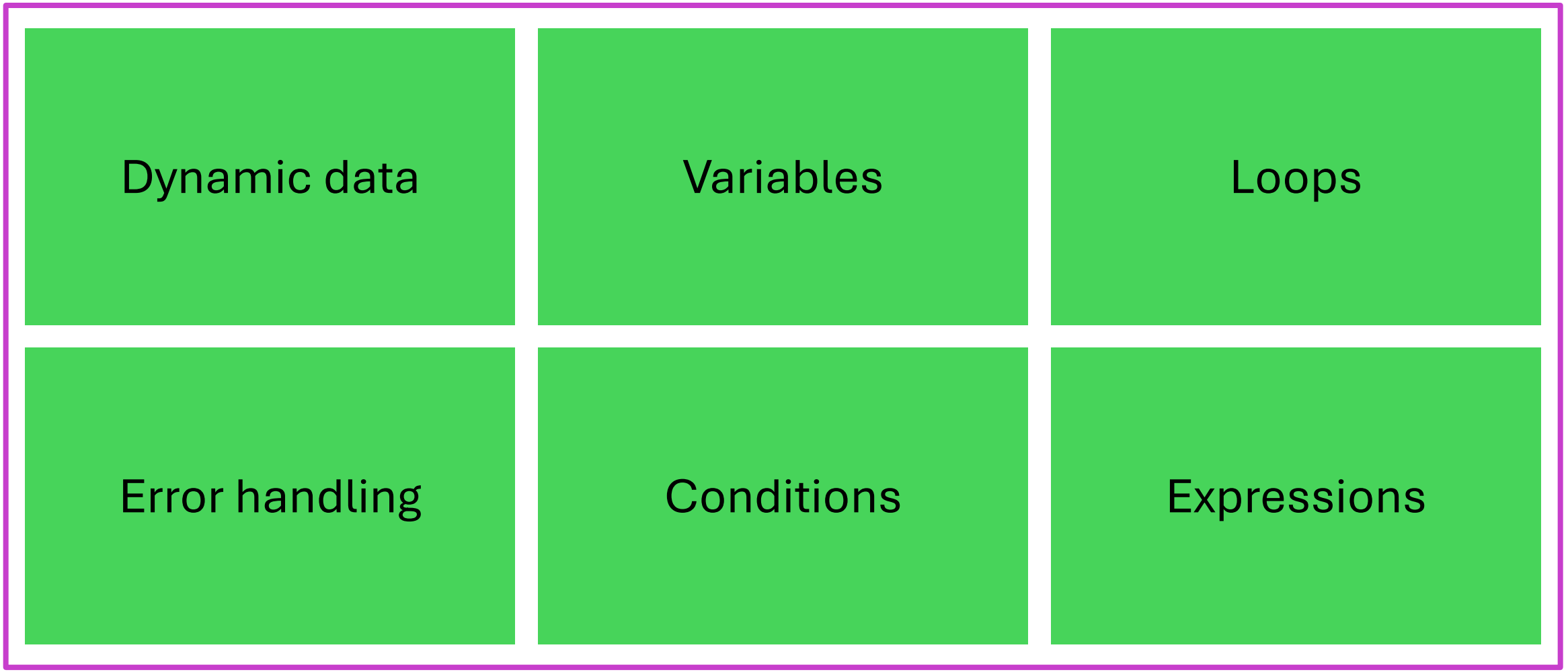


Example – Follow up on a message



TRAINOCATE

Flow building fundamentals



Ways to start a cloud flow

With Copilot

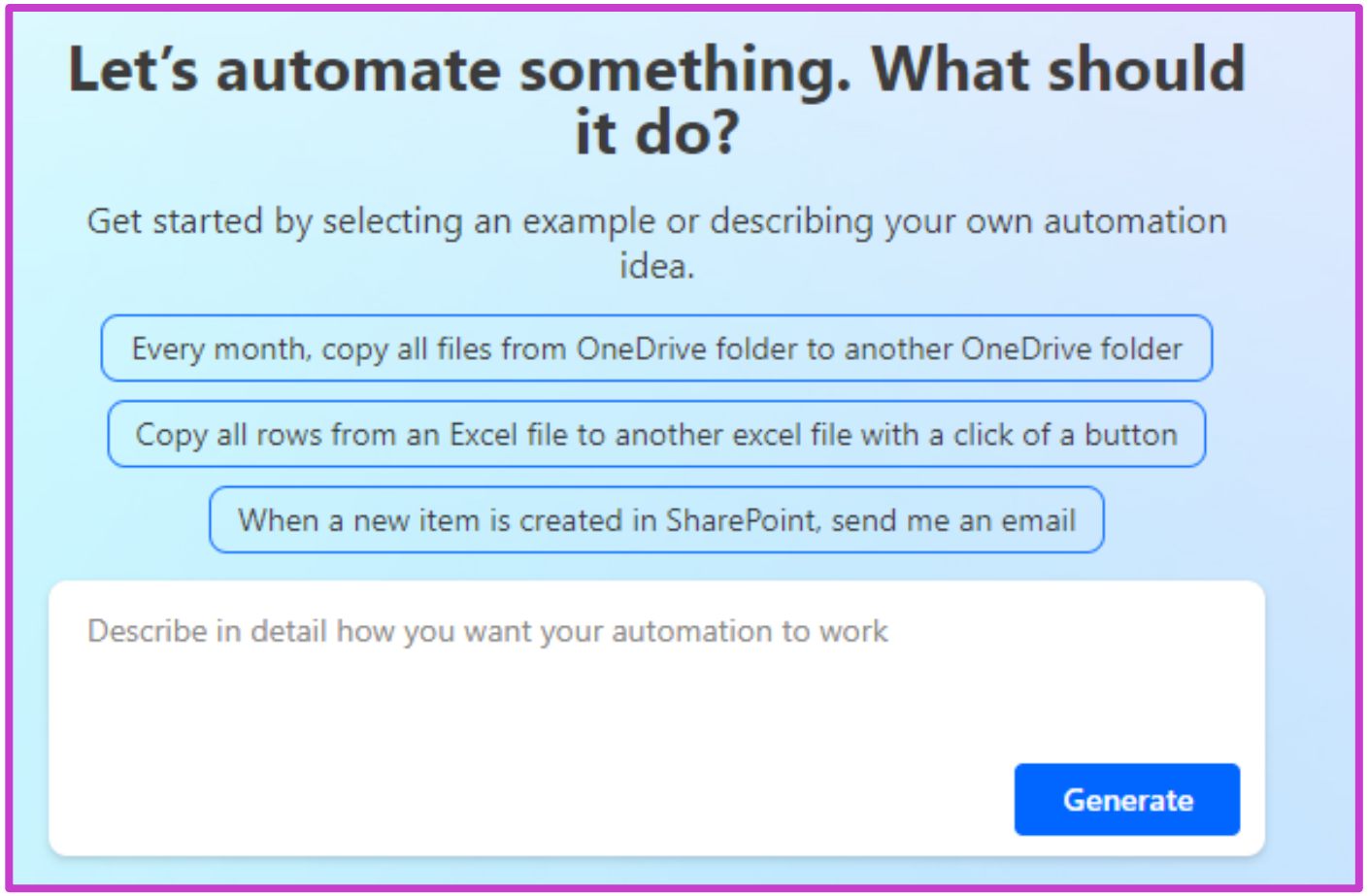
From a template

From blank

Create a cloud flow

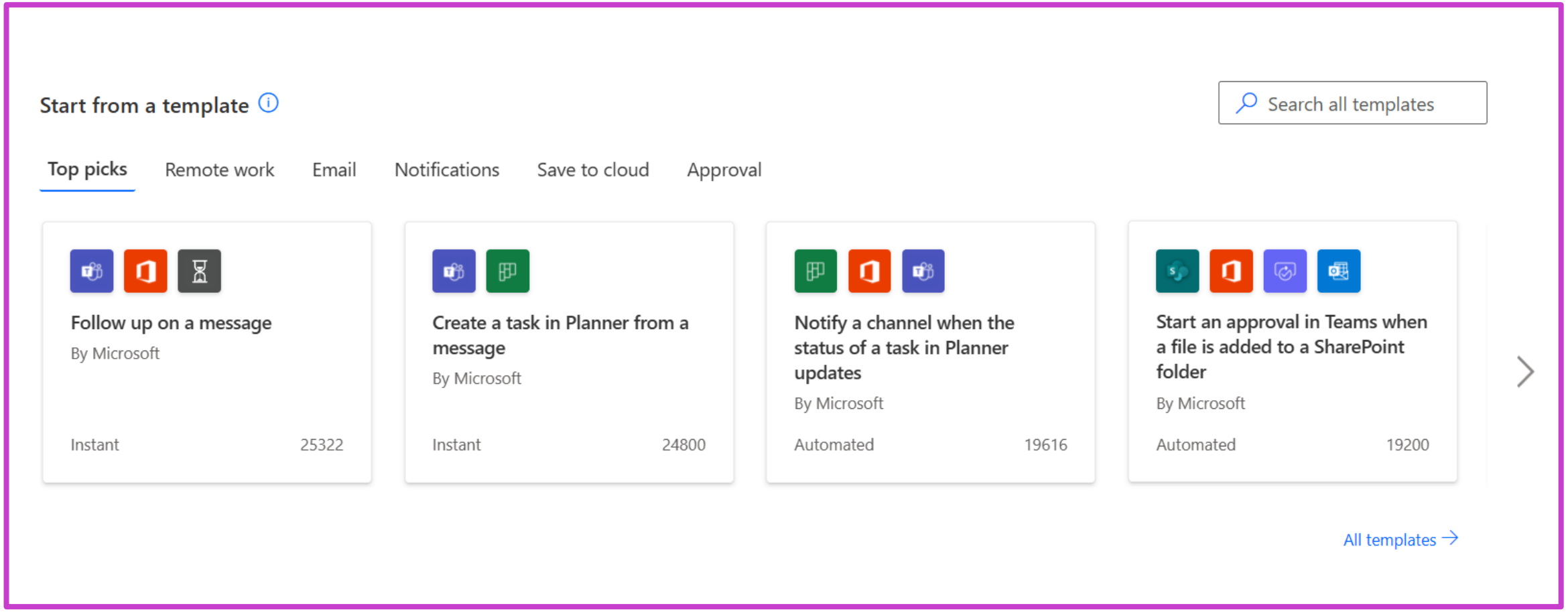
Start with Copilot

- Build flows through conversation



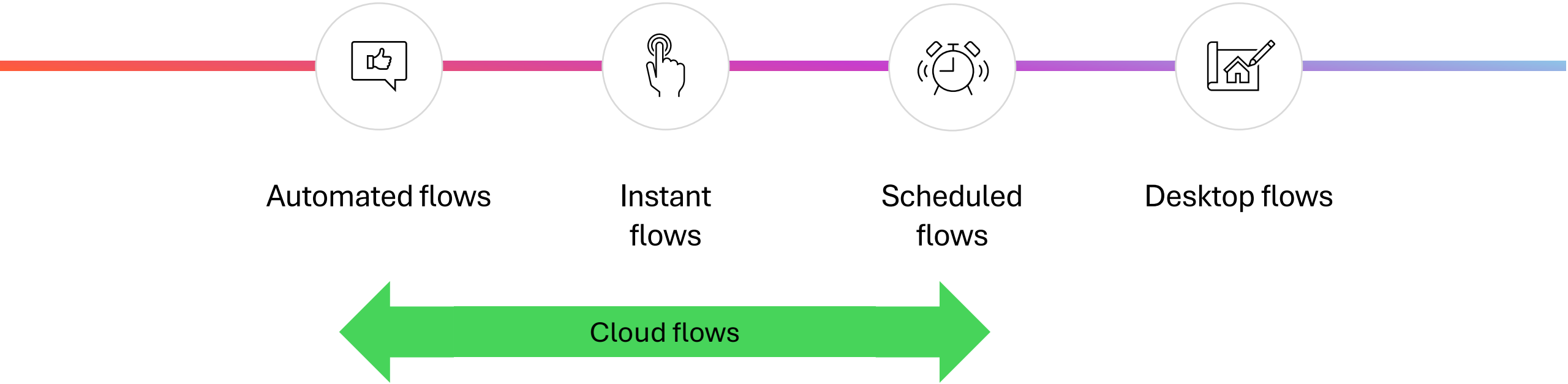
Create a cloud flow

Templates provide a quick start for common scenarios



Power Automate

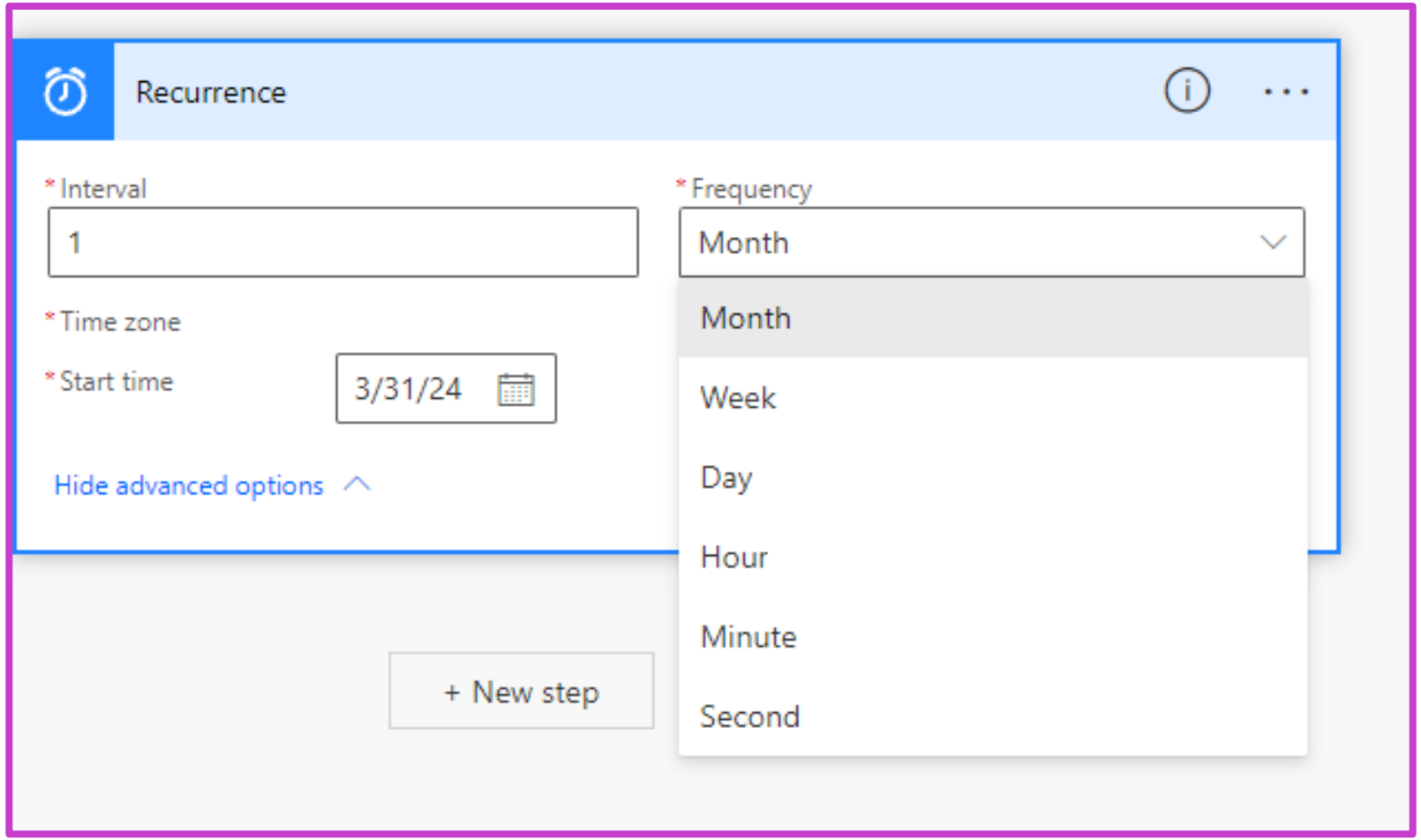
Types of Power Automate flow



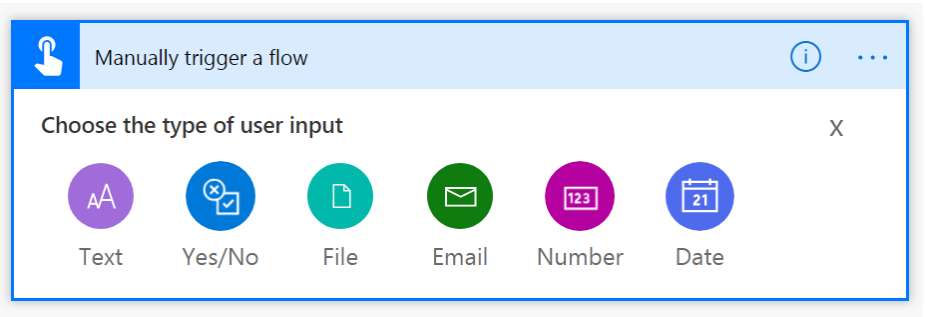
Create a scheduled flow

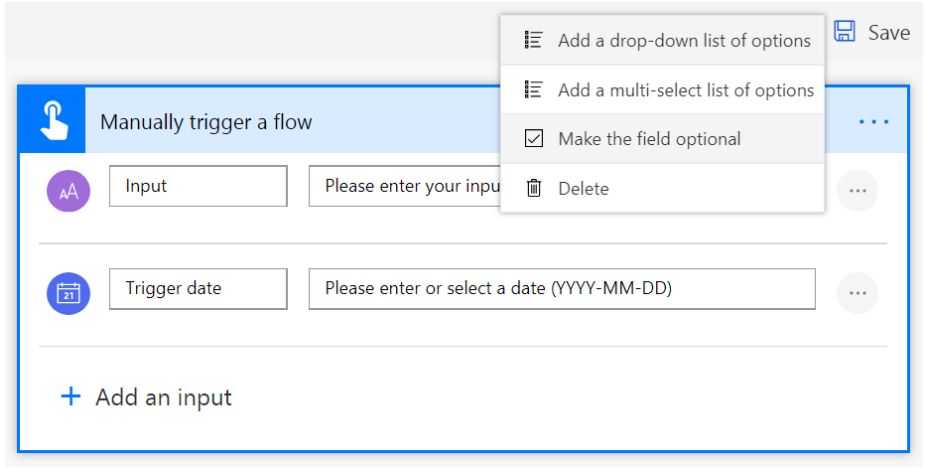
Recurrence trigger

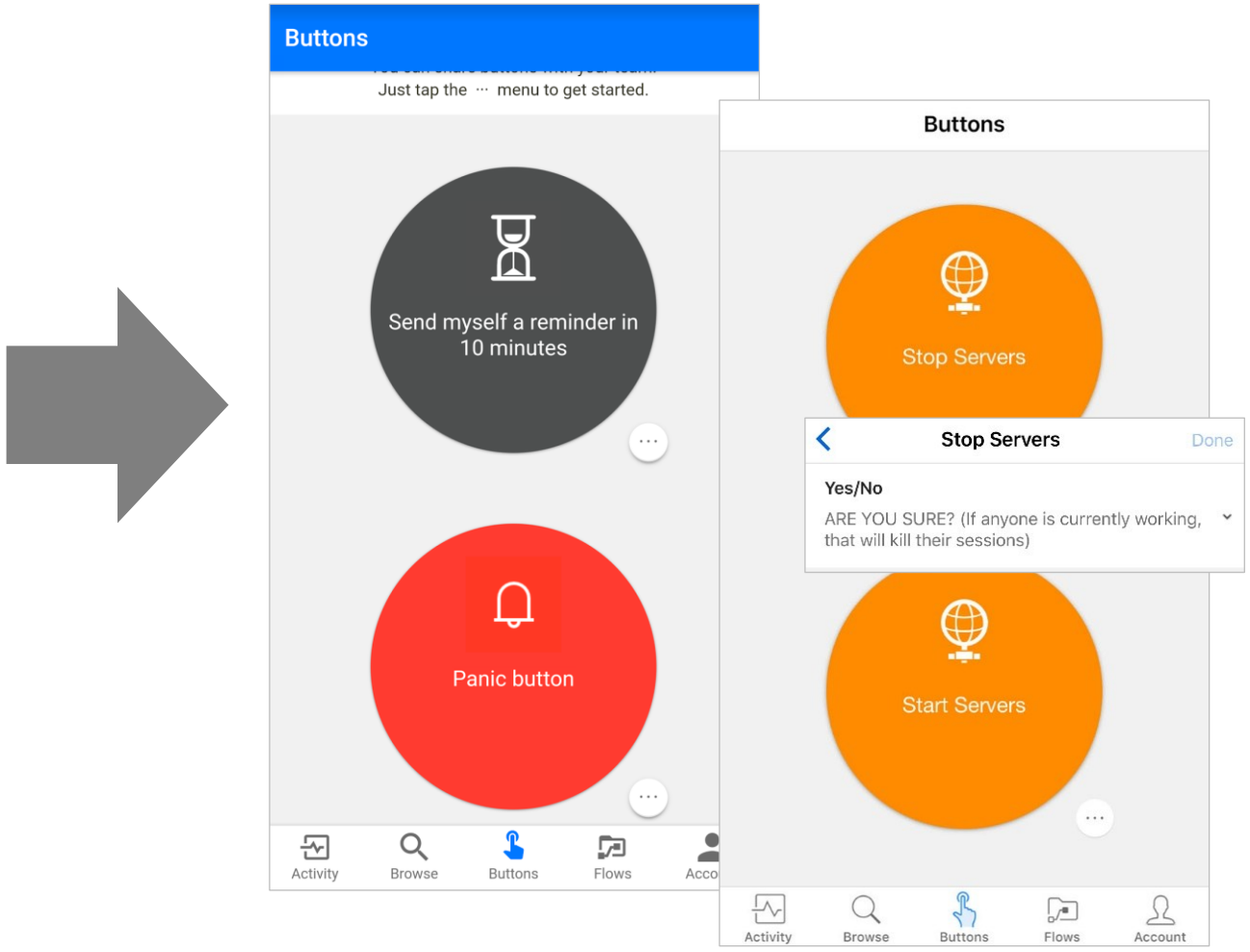
- Runs on a pre-defined schedule
- Can also run manually from Power Automate portal



Create an instant flow





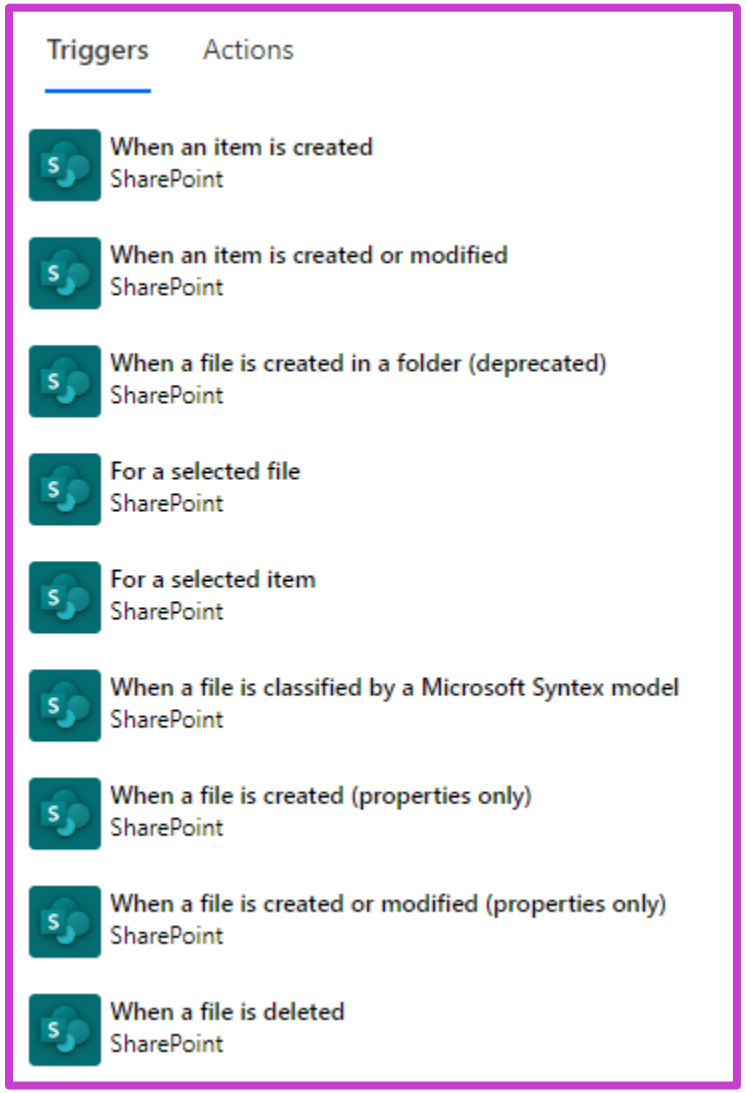


TRAINOCATE

Create an automated flow

Event-based trigger

- Most connectors have triggers
- Tabular connectors
  - Created/Added
  - Updated/Modified
  - Read/Get
  - List/Guery
  - etc
- Function based triggers
  - Vary by connector

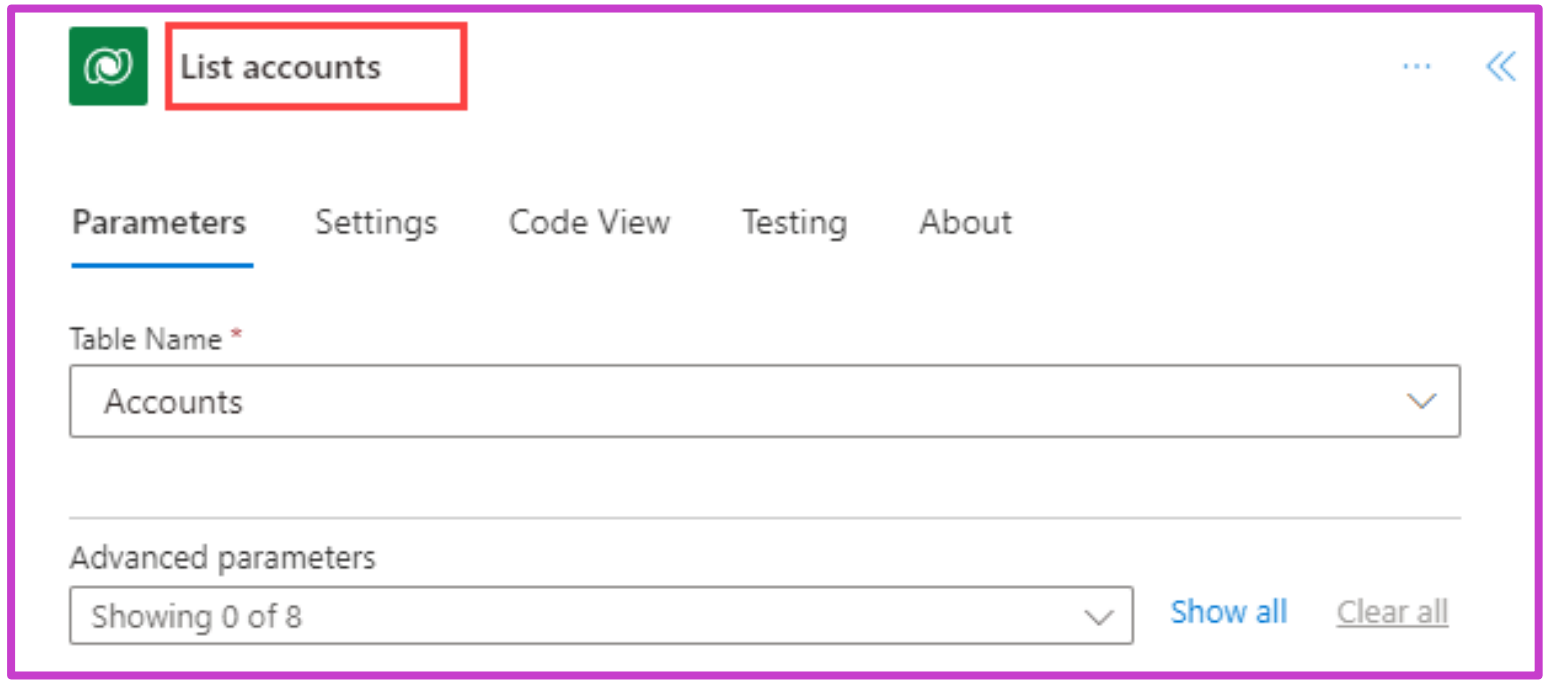


Naming

Naming flows
Most important for instant flows, since they will be selected by end users from a list of flows

All other flows will need clear names for administrators to tell what has run/failed

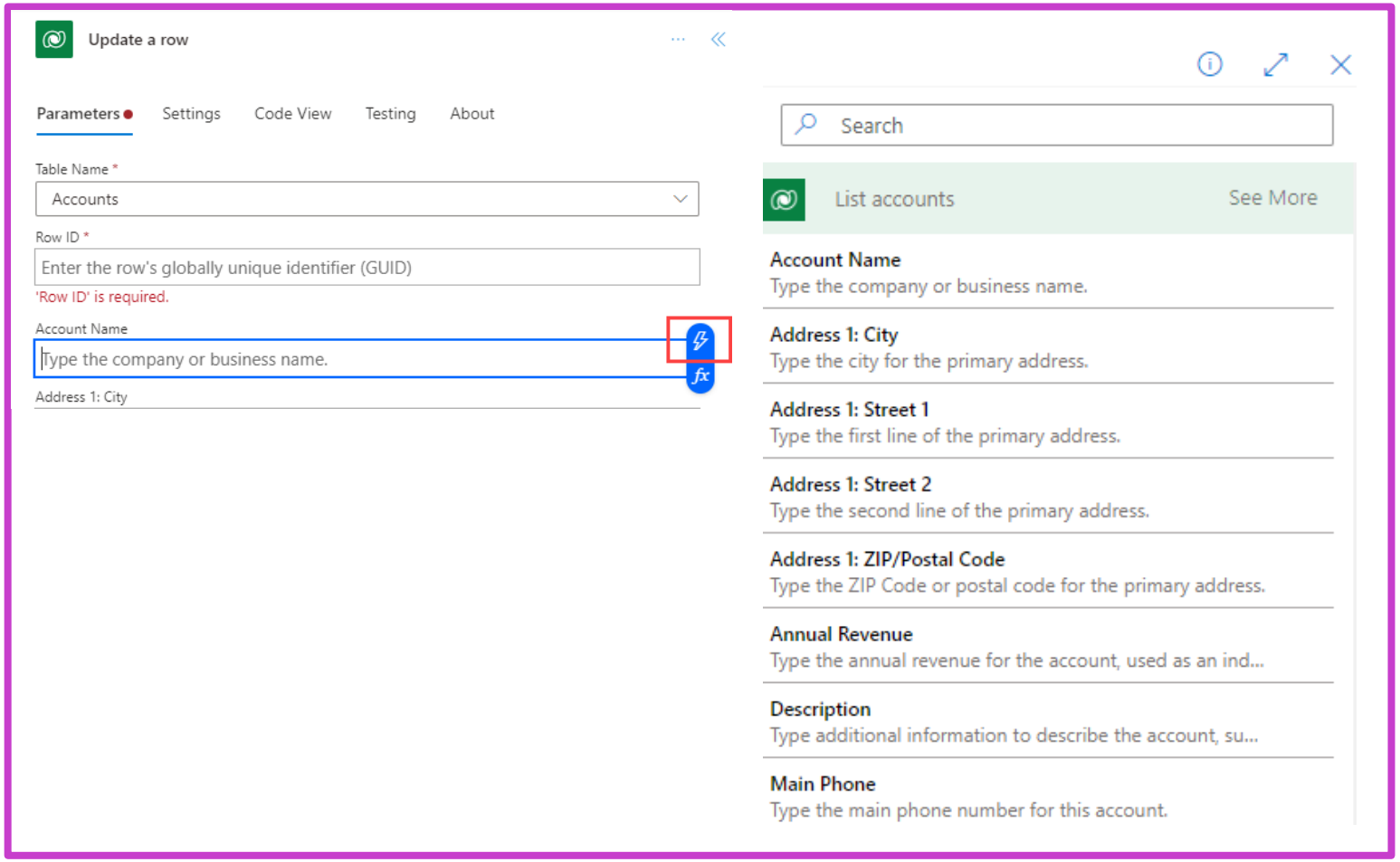
Naming flow actions
Every action gets default name



Dynamic data

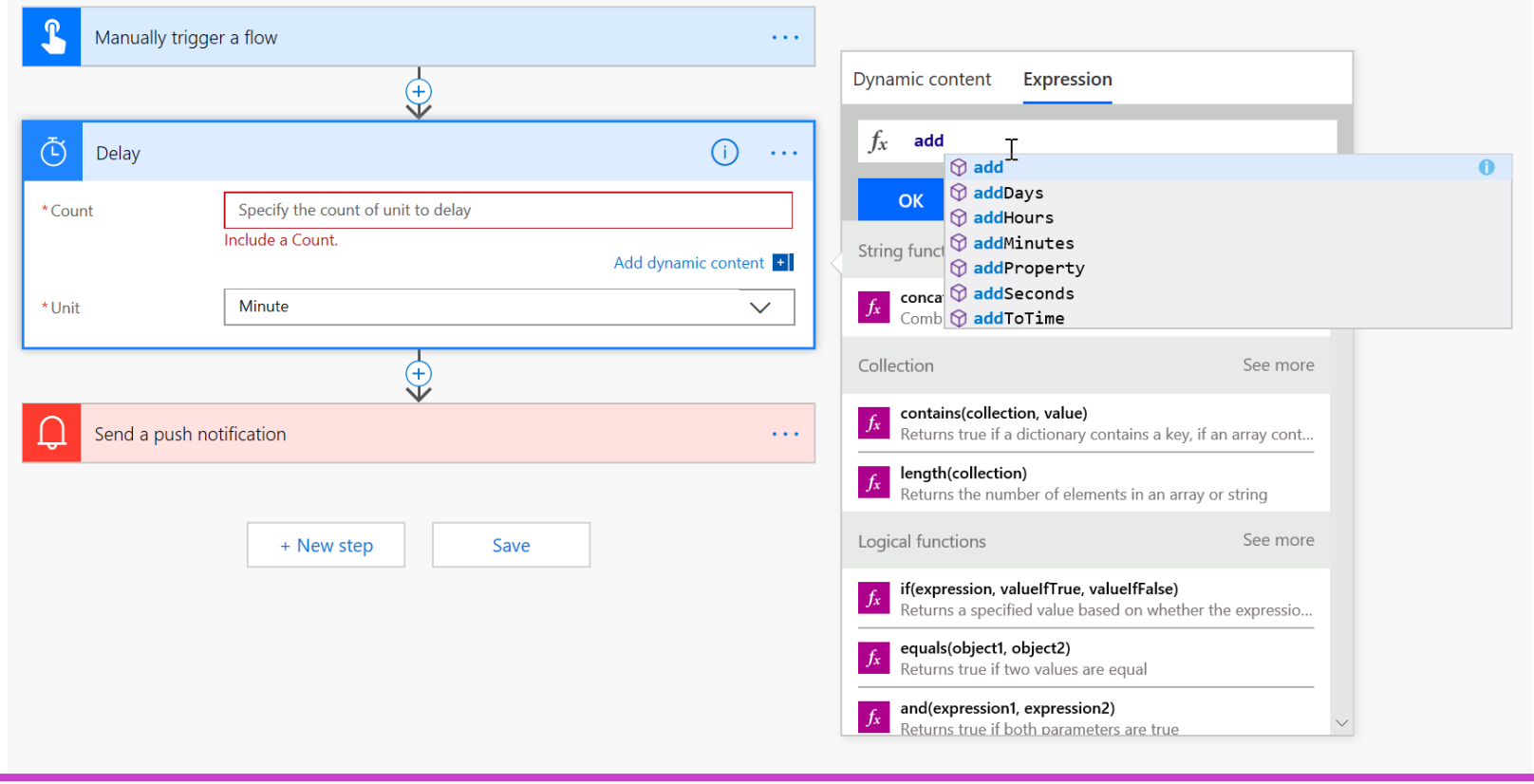
Use dynamic content panel to reference data from prior steps in a flow and dynamically bind it to current properties

Dynamic content panel shown when clicking in a property value on a step flow



Expressions

To write an expression in Power Automate, select a field to open the Dynamic content menu and then select the Expression tab.



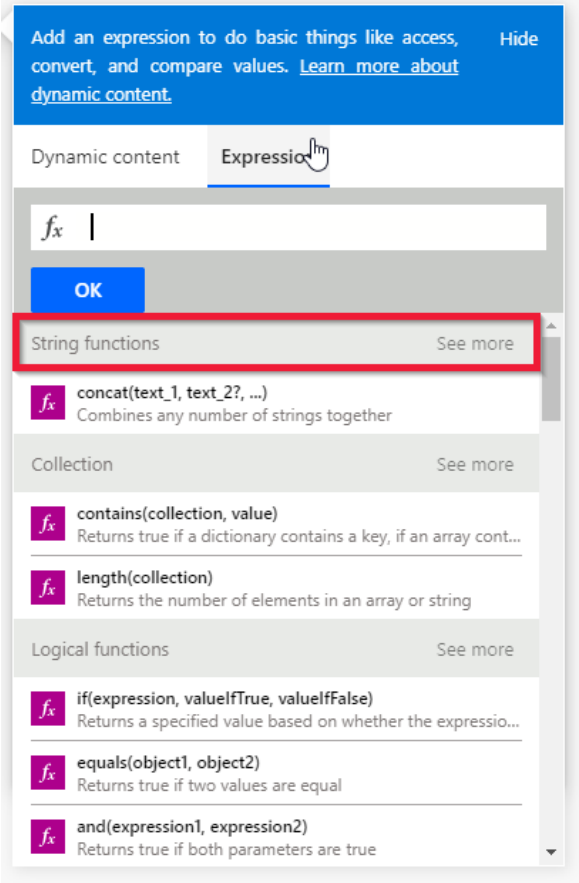
TRAINOCATE

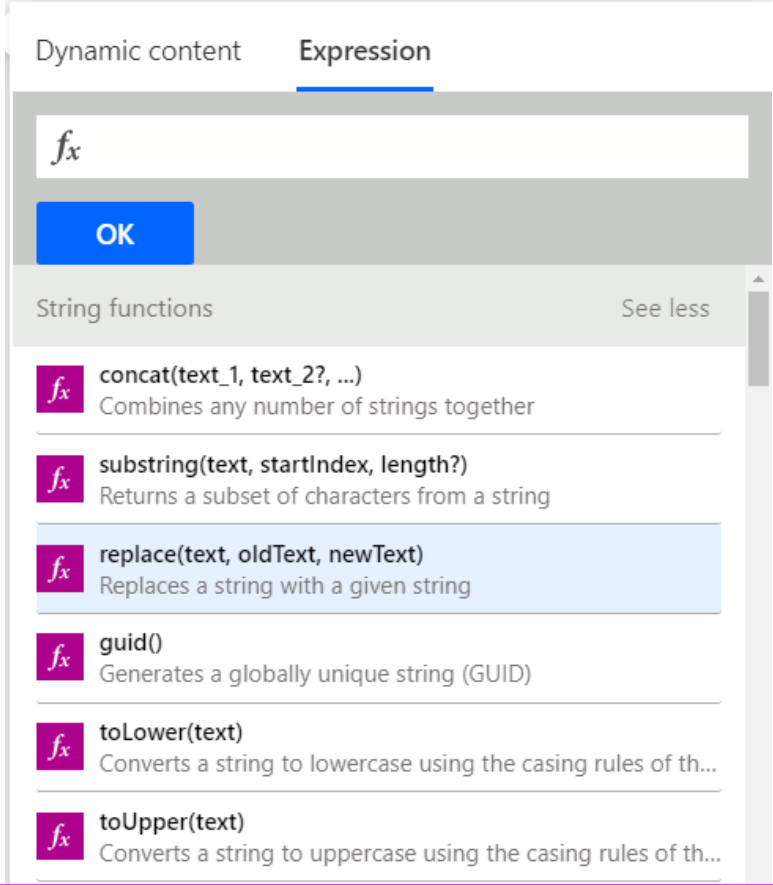
Many reasons to use expressions

- Change Data Type
- Do Simple Math or Text Work
- Create Dynamic Values (random, GUID,...)
- Check for Empty Data
- Make Decisions ("If" conditional statements)
- Work with Data Sets (lists)

Types of expressions

Expressions are grouped into 10 different categories like math and logic.





TRAINOCATE

Core expressions

- **outputs(stepName)** – Returns object with outputs from a step
- **body(stepName)** – Returns object with properties from a step
- **triggerBody()** – Like body() but for trigger output
- **item()** – Used to reference the current item in Apply to each
- **variables(variableName)** – Reference a variable
- **parameters(parameterName)** – Environment variable

Handling null values

Trying to use a property that's not set, "null", will cause the flow to fail

Two steps to handle

- Use the "**?**" character after selecting the property (this happens automatically)
- Then, use the coalesce() function to provide a default value

```
coalesce(body('Get_record')?
['content'],'Default Value')
```

Data Types and Formats

Text – aka Strings

- Normal – ‘Example string’
- Email, URL, phone number, etc…
- Base64
- Binary content
- Data URI
- URI component

For strings use single quotes

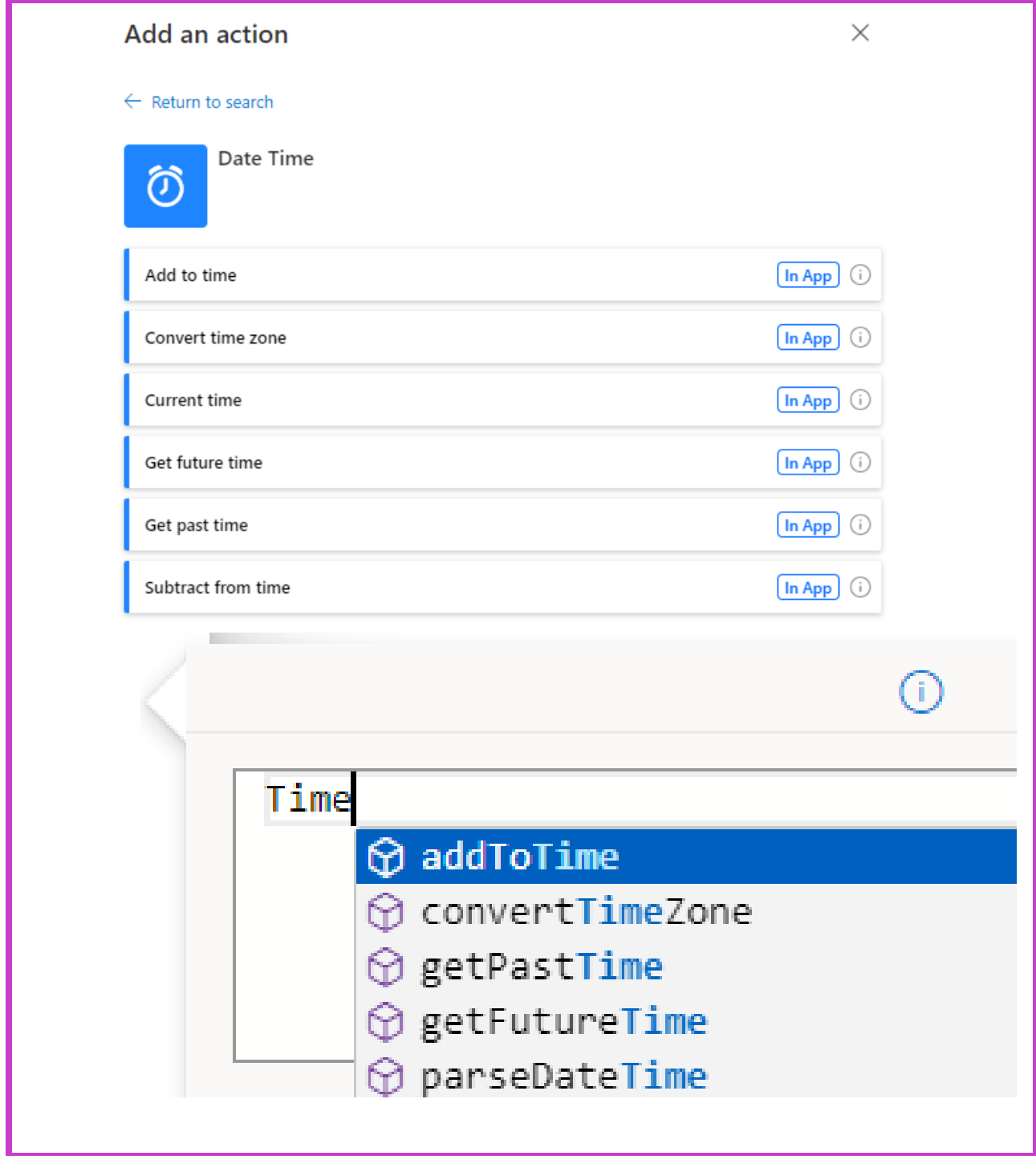
Non-text

- Floating-point number – 9.0
- Integer number – 137
- Boolean – True or false
- Array – A list of items
- Object – A record with key-value pairs
- XML Content

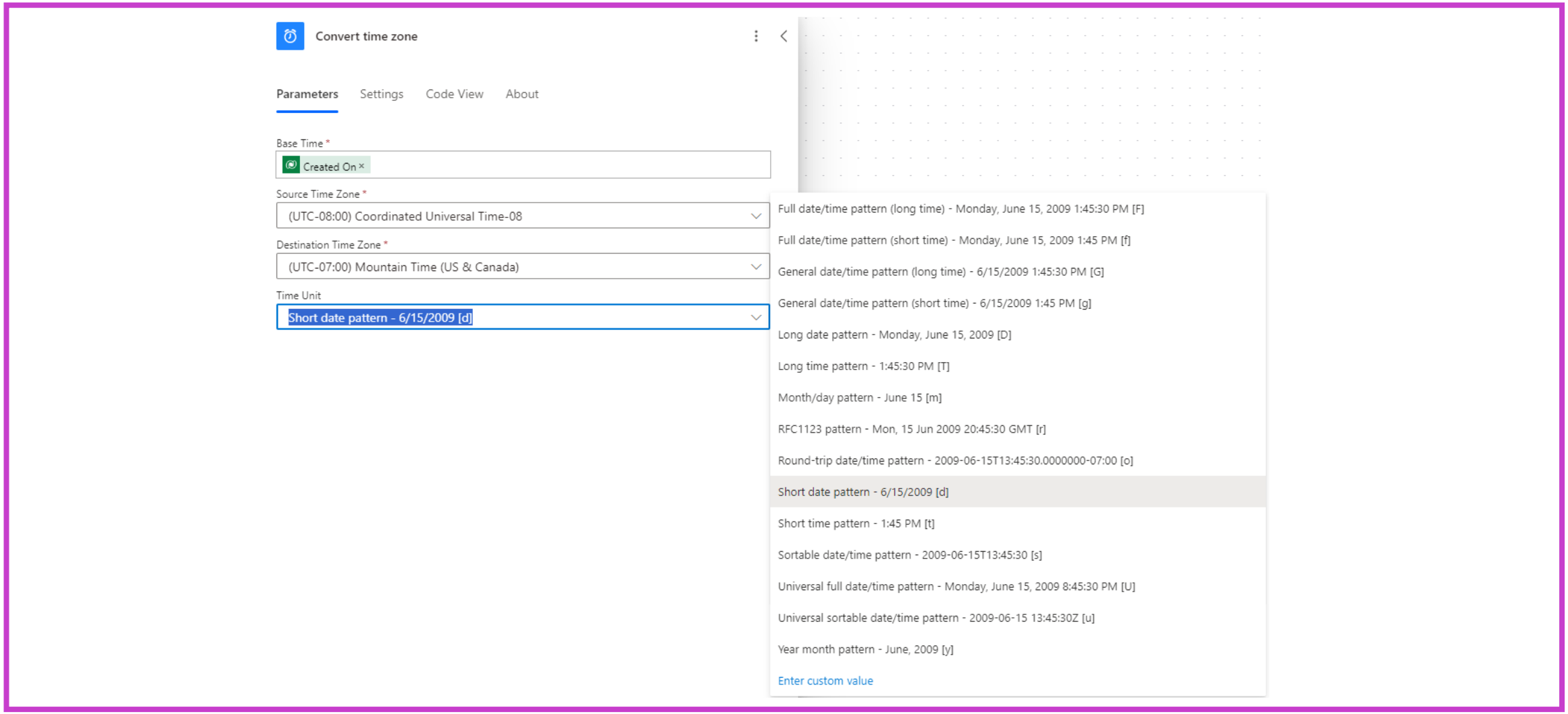
Do *not* use single quotes for numbers, integers or Booleans: ‘true’ *does not equal* true

Working with Dates

- No date time datatype
- Date time is in ISO 8601 format (json)
  - 2022-11-30T20:46:27+10:00
- Date Time connector
- Or expressions
- Date parameter includes only date without time or timezone. To compare parameter dates
  - < ≤ ≥ > == ≠ all bad as you're comparing strings
  - Convert to ticks
  - Use substring to restrict what are we comparing, e.g., first 10 characters would give you the date only



Preparing Dates for Display



Examples Using Expressions with Dates

Get days since account was created

```
int(
  div(
    sub(
      ticks(utcNow()),
      ticks(triggerBody()?['entity']?['createdon'])),
    864000000000
))
```

Check if date is between business hours

```
and(
  greaterOrEquals(
    formatdatetime(utcnow(),'HH:ss’),
    formatdatetime('7:00','HH:ss')),
  lessOrEquals(
    formatdatetime(utcnow(),'HH:ss’),
    formatdatetime('15:00','HH:ss’))
)
```

Escaping Rules

1. When you are directly using the expression language you may need to escape certain characters

2. Using strings with a single quote character

   @substring('It''s A Great Day!',1,5)

3. Using spaces in action names: @body('Name_with_spaces')

5. Building an object you need to escape '@'

   - {
   - "@@Odata.type" : ""
   - }

Common expressions

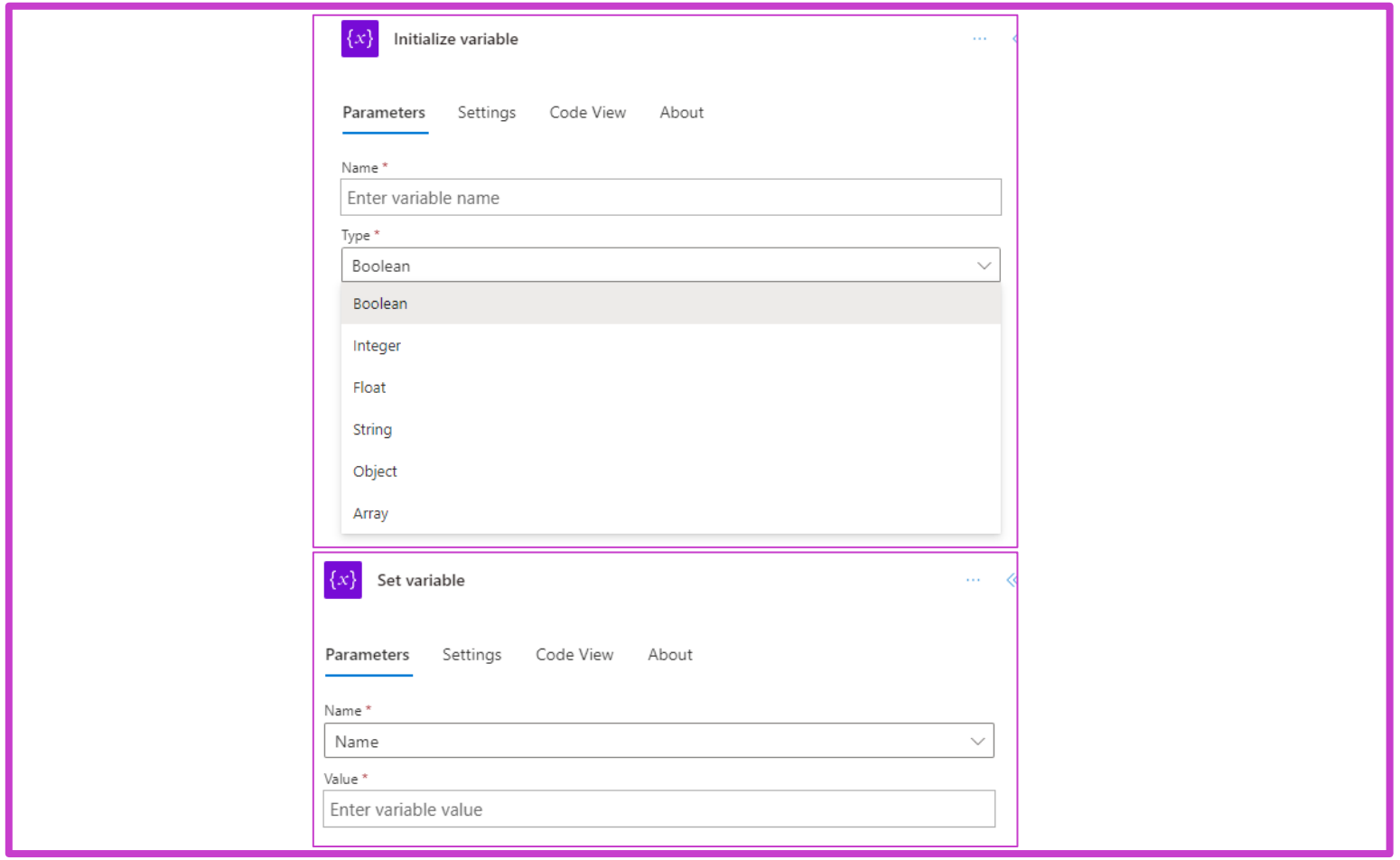
- **coalesce(<object_1>, <object_2>, ...)** – Return first non-null value
- **guid()** – Generate a GUID
- **replace(string, old, new)** – Replace old with new in string
- **equals(left, right)** – Returns true if left equals right
- **utcnow('yyyy-mm-dd')** – Generate a date/time
- **ticks('<timestamp>')** – Returns the number of ticks since January 1, 0001
- **string()** – Convert to plain/text
- **json()** – Convert to application/json – Can parse like JSON
- **xml()** – Convert to application/xml
- **xpath(<xml>, <expression>)** – Execute Xpath expression
- **if(<condition>, <true>, <false>)** – Set value based on condition
- **result(<scope>)** – Return the run result for a scope of actions

Variables

Initialize at top of flow and select data type

Set during flow

Reference using Dynamic data



TRAINOCATE

Module 2: JSON in Workflows

The Core of Power Automate

JSON

- A data storage format that is easy for computers to understand.
- Comparable to an Excel table written in code.
- A programming language used for communication between various systems.
- Every **Action** in **Power Automate** uses **JSON**.

TRAINOCATE

JSON Basics: Objects

- ***Key-Value*** Pair Storage
- **Example**:

```
{
  "name": "สมชาย" ,
 "age": 20,
  "city": "กรุงเทพ"
}
```

- "name" = **Key** (ชื่อของข้อมูล)
- "สมชาย" = **Value** (ค่าของข้อมูล)

JSON Basics: Arrays

- A list of multiple data items, arranged in a sequence
- **Simple Array** :

  ["แอปเปิ้ล", "ส้ม", "กล้วย"]

- **Array of Objects**:

```
[
{ "id": 1 , "name": "สมชาย" } ,
{ "id": 2 , "name": "สมหญิง" }
]
```

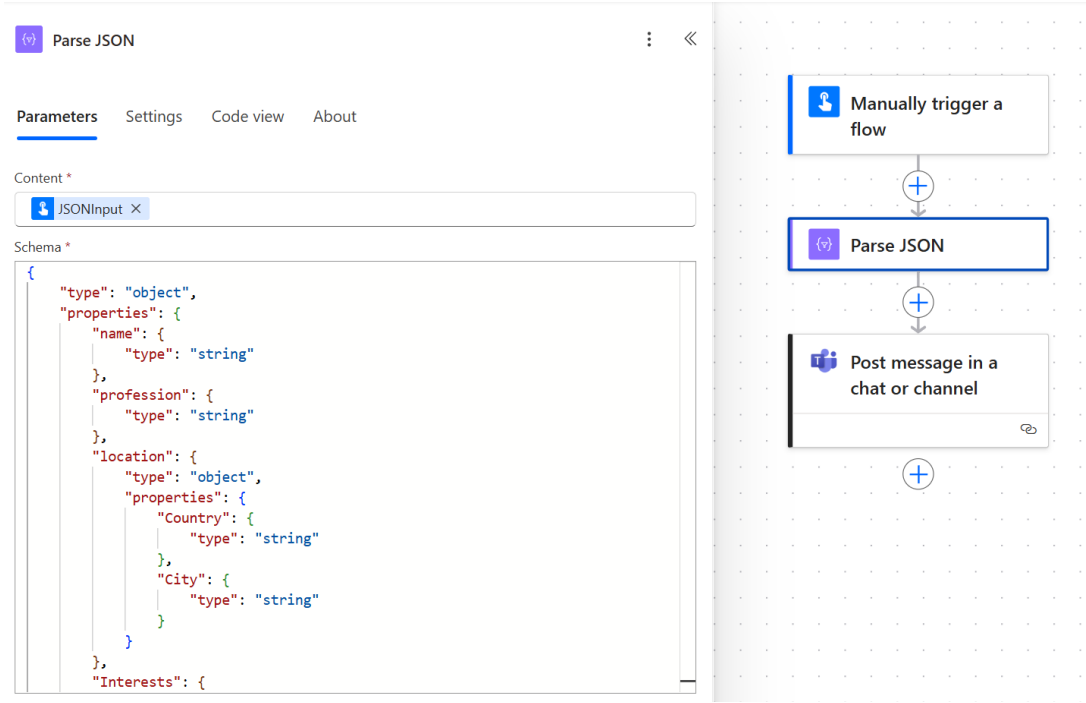
TRAINOCATE

Key-Value ใน Power Automate

- Field Name = ***Key***
- Dynamic Content = ***Value***

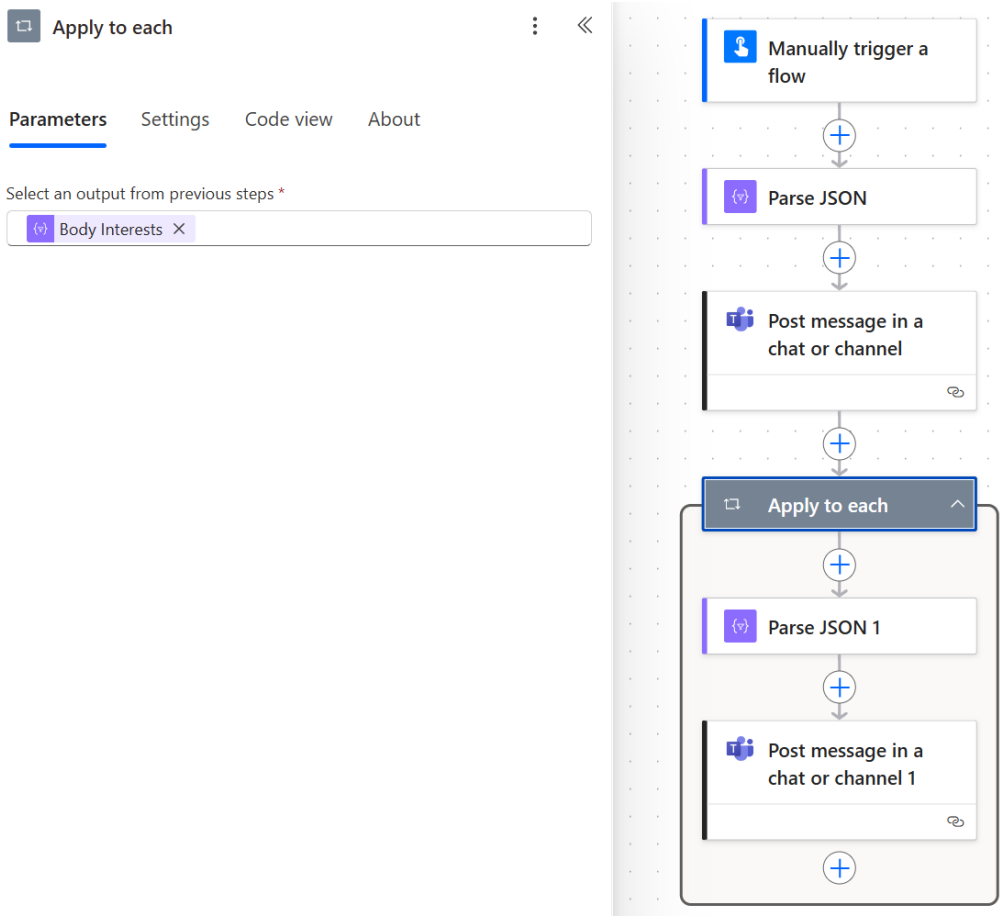
Parse JSON Action

- **Common Issues:**
  - **JSON data is often a String** (text)
  - It **must be converted into an Object** (data structure) before use
- **The Method (in Power Automate):**
  1. Use the **Parse JSON Action**
  2. Input the **JSON String** to be converted
  3. Define the **Schema** (Structure)



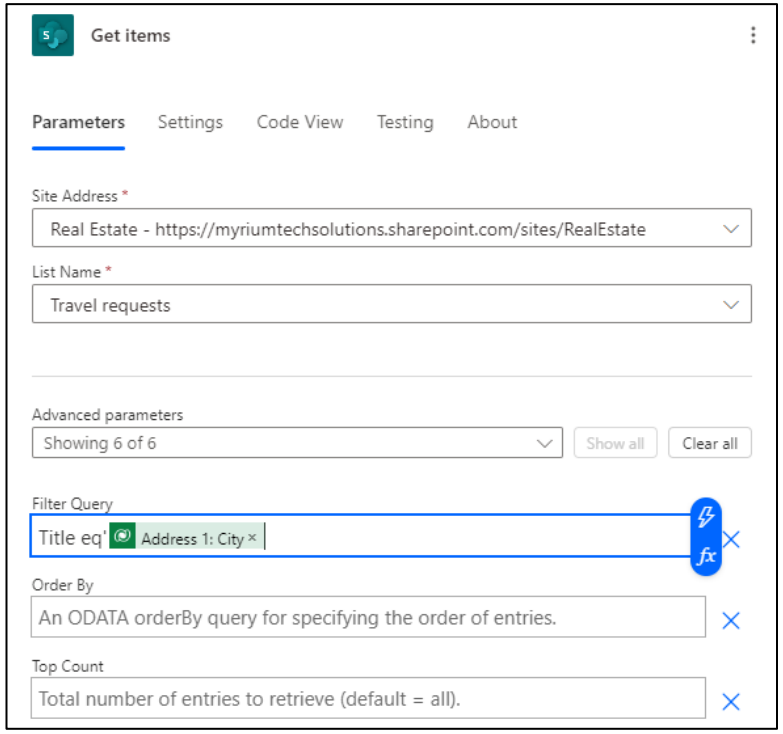
Parse JSON (Array)

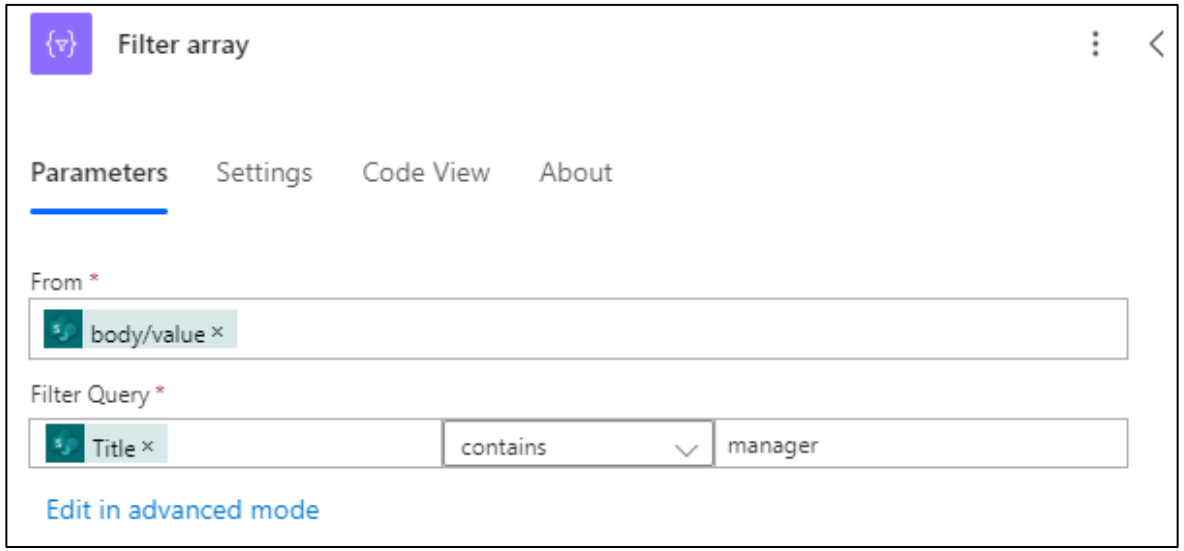
1. **Steps (Configuration):**
   - Use the **Parse JSON Action**.
   - Input the JSON String into the **Content field**.
   - **Generate Schema** from a **sample payload**.
2. **After Parse JSON:**
   - body('Parse_JSON')?['items'] (Array of items)
3. **extraction Operation:**
   - Extract items



Filtering

- Have the connector do the filtering for you – **Recommended**
- Built-in flow action – Filter array
  - Select the array in the From field
  - Can either use simple or advanced mode just like for conditions
- If you need the first N items you can use take() or skip()

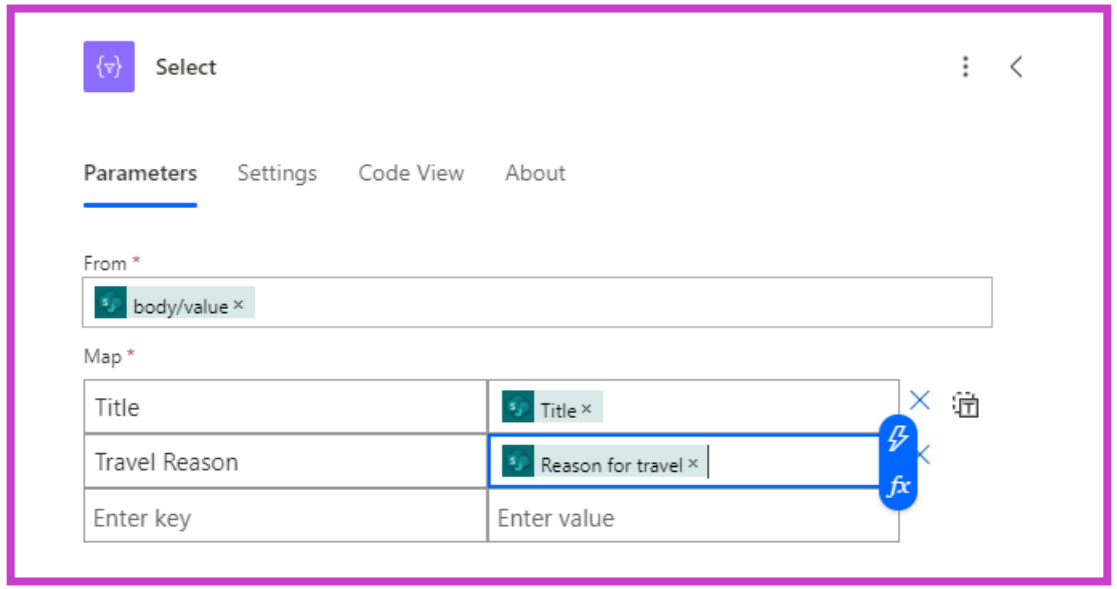




Selecting Arrays

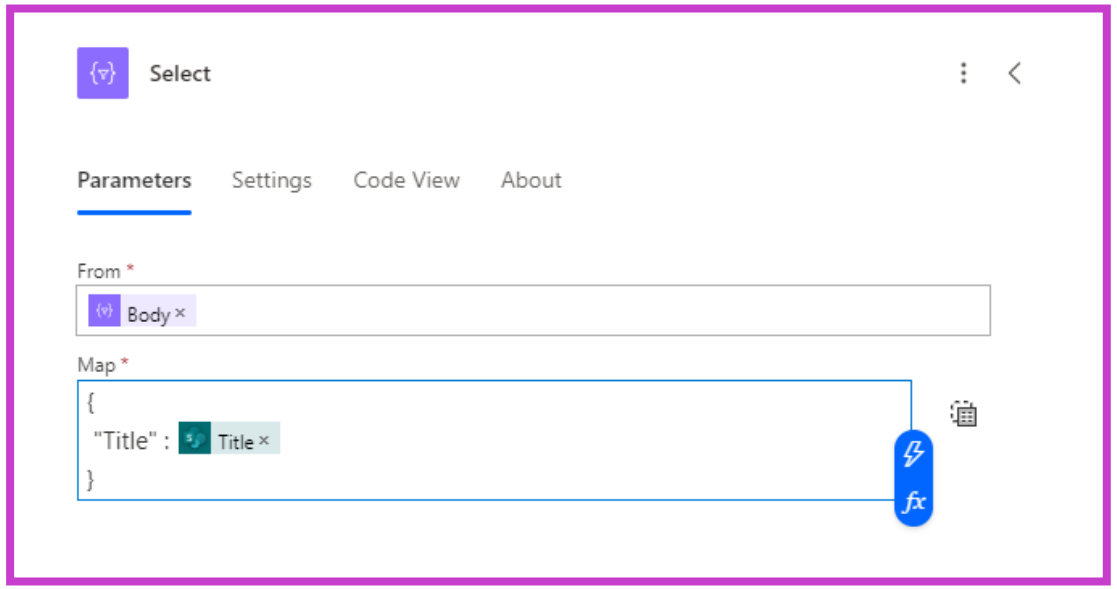
Two input modes: Fill key-value pairs, or typing directly

Create an array of objects



Useful for passing this array to another action

Create an array strings, numbers, Booleans etc.



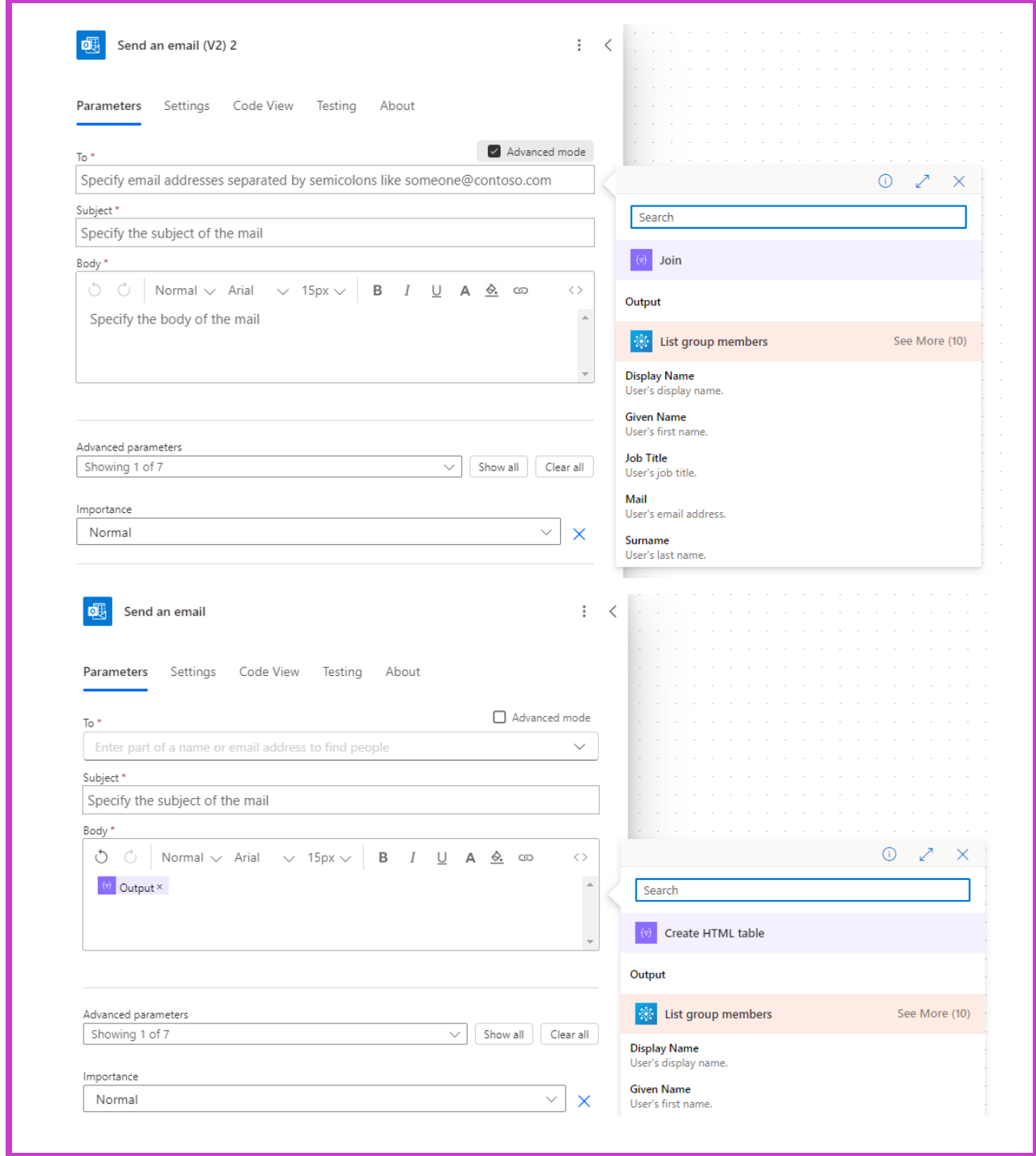
Useful for getting a simple list, for example, of email addresses

TRAINOCATE

Convert the List to a String

- **Join –** Use join to get a simple list, for example, if you have a list of email addresses*
- **Create HTML/CSV table –** Convert a list of objects to a tabular display format, for example, for inclusion in the body

* **Note**: You must have a list of strings, not a list of objects



TRAINOCATE

Module 3

Taking Control with Flow Controls

Conditional logic

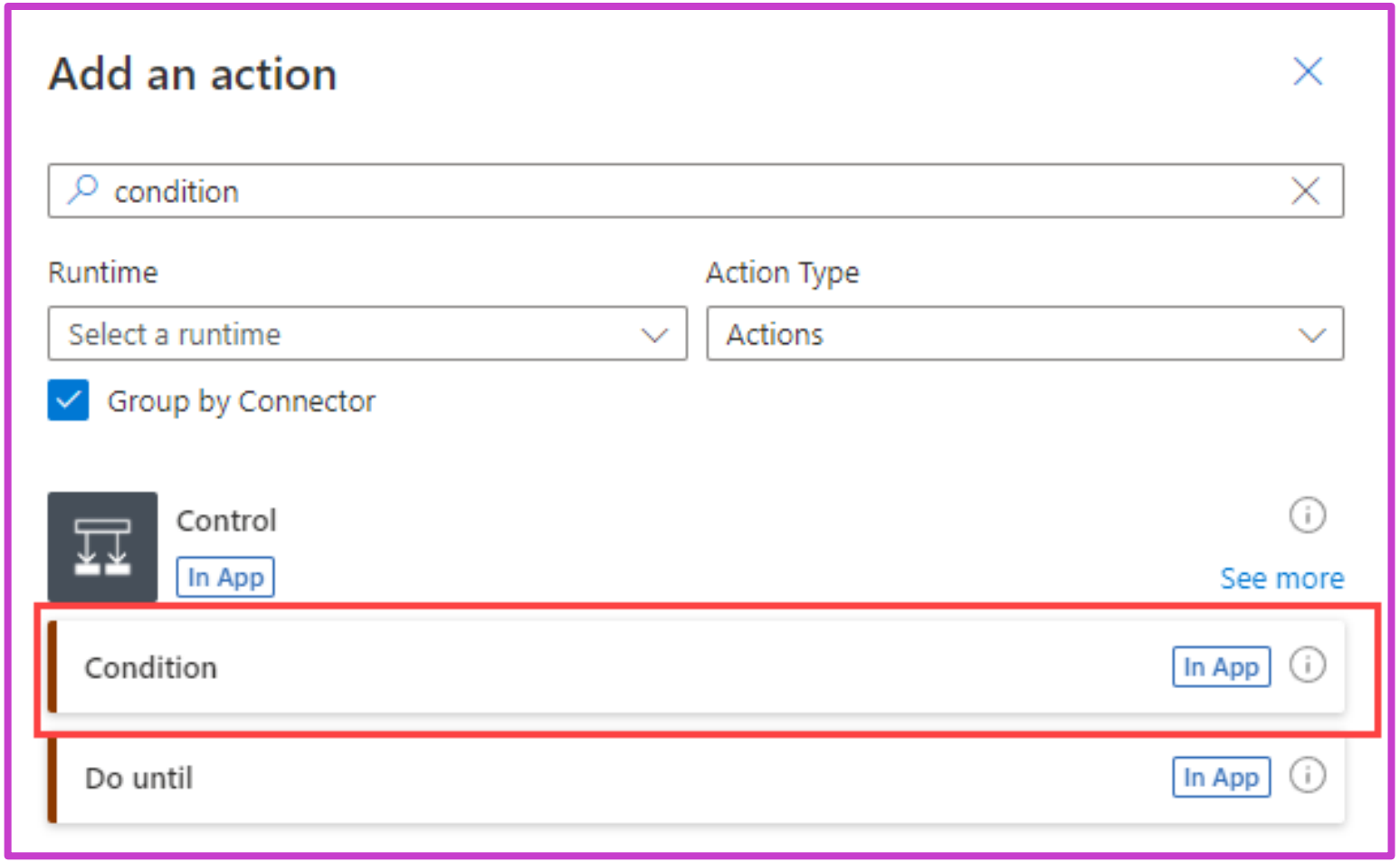
Conditions and switches offer ways to introduce conditional branching into a flow

Using conditions

Search actions on “Condition” and select

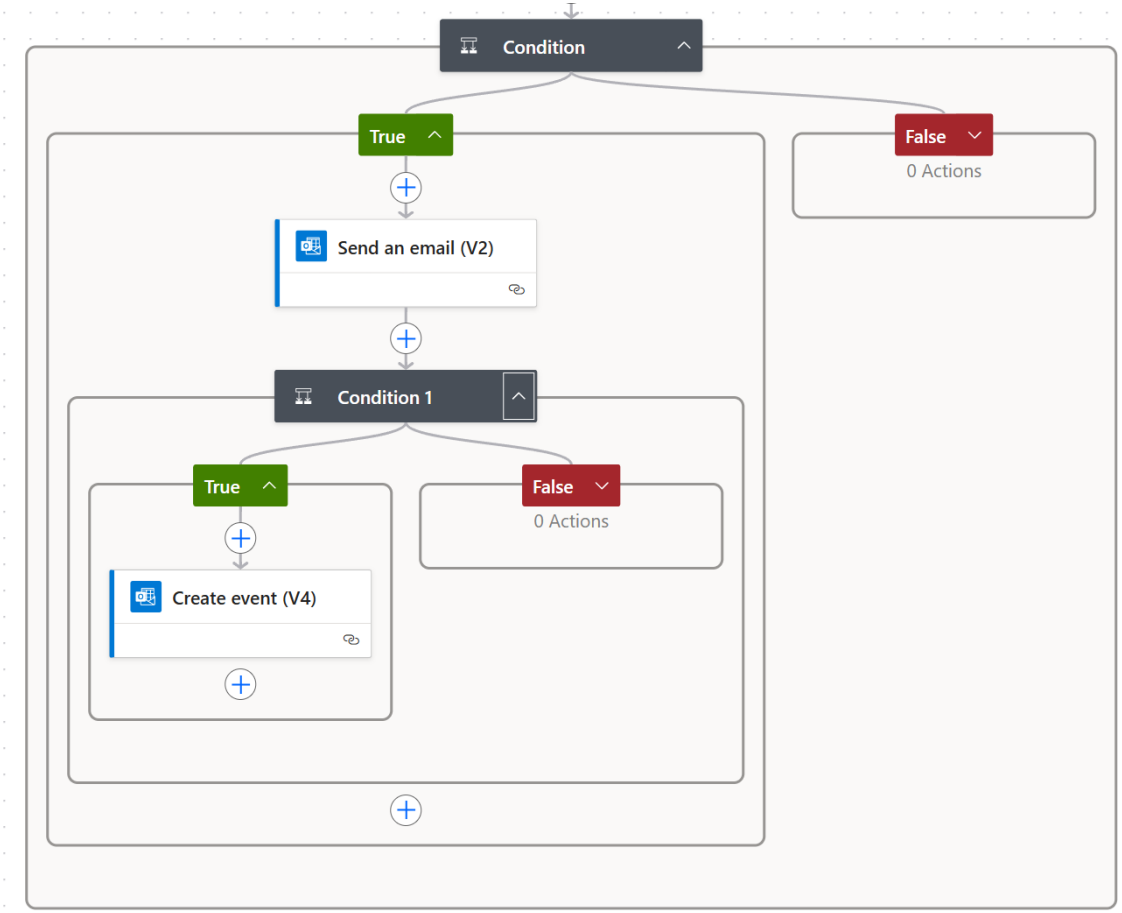
Using switches

Searching on “Switch” and add to flow



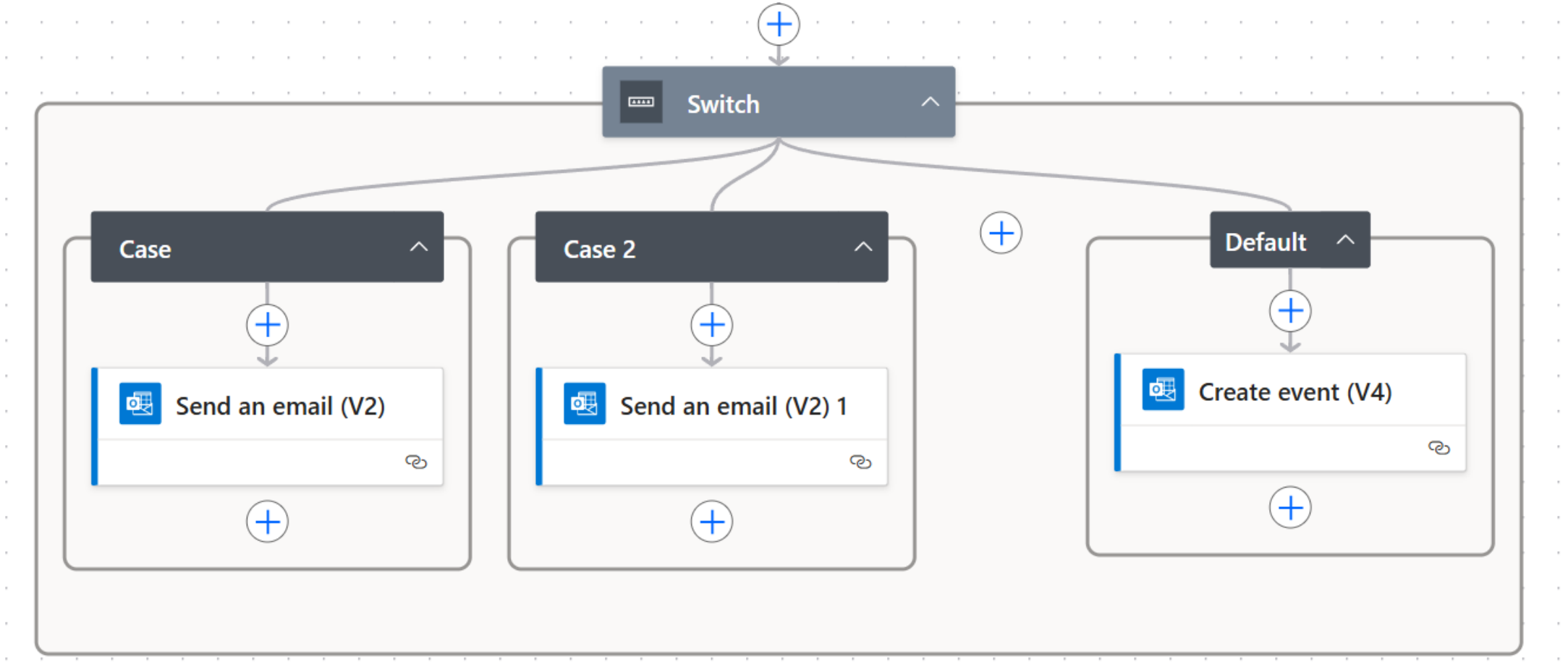
TRAINOCATE

Nest Condition



TRAINOCATE

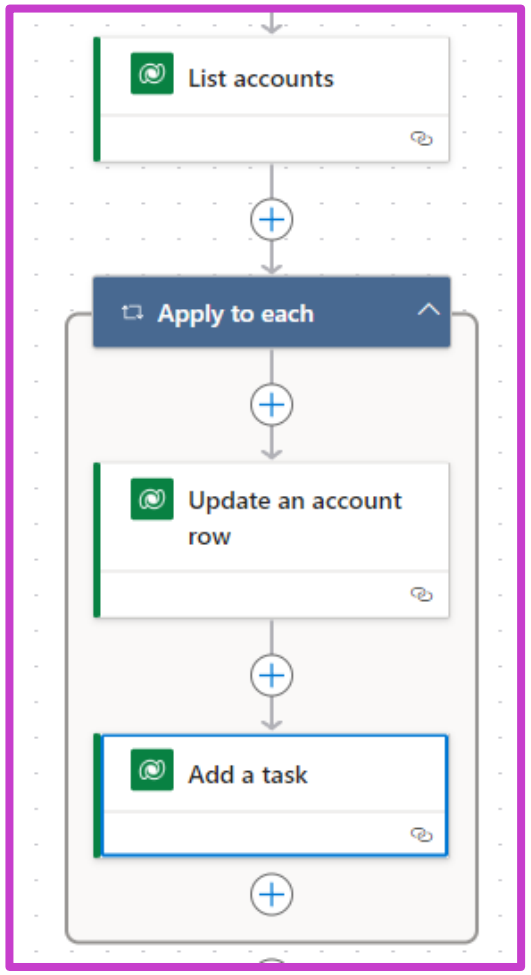
Switch



Loops

Apply for each

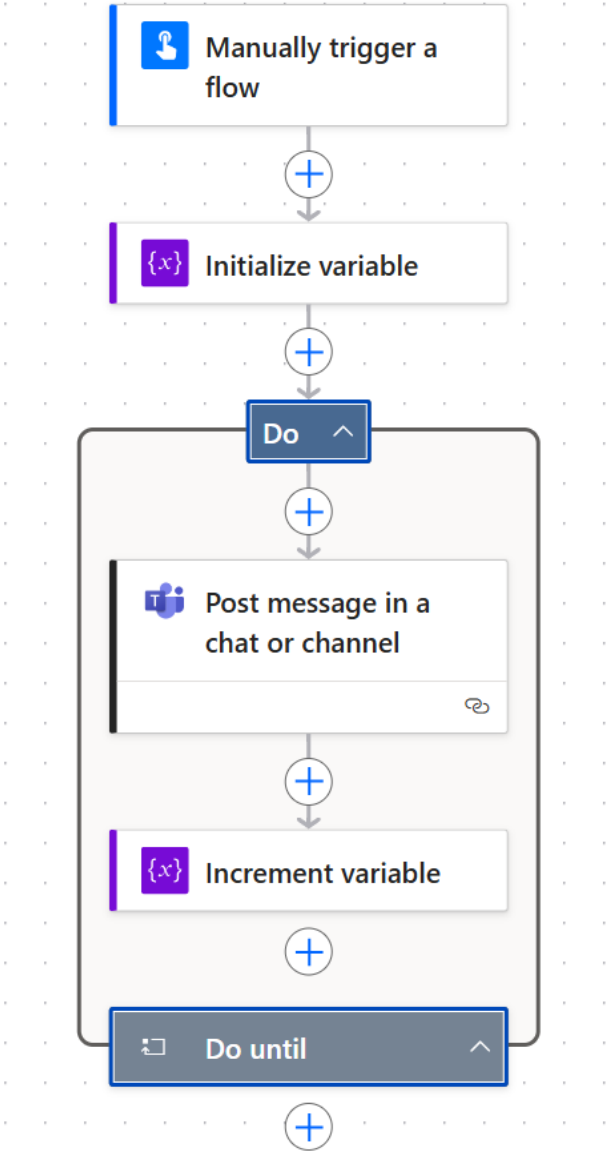
- Executes for each item in an array
- Often used after a query step



TRAINOCATE

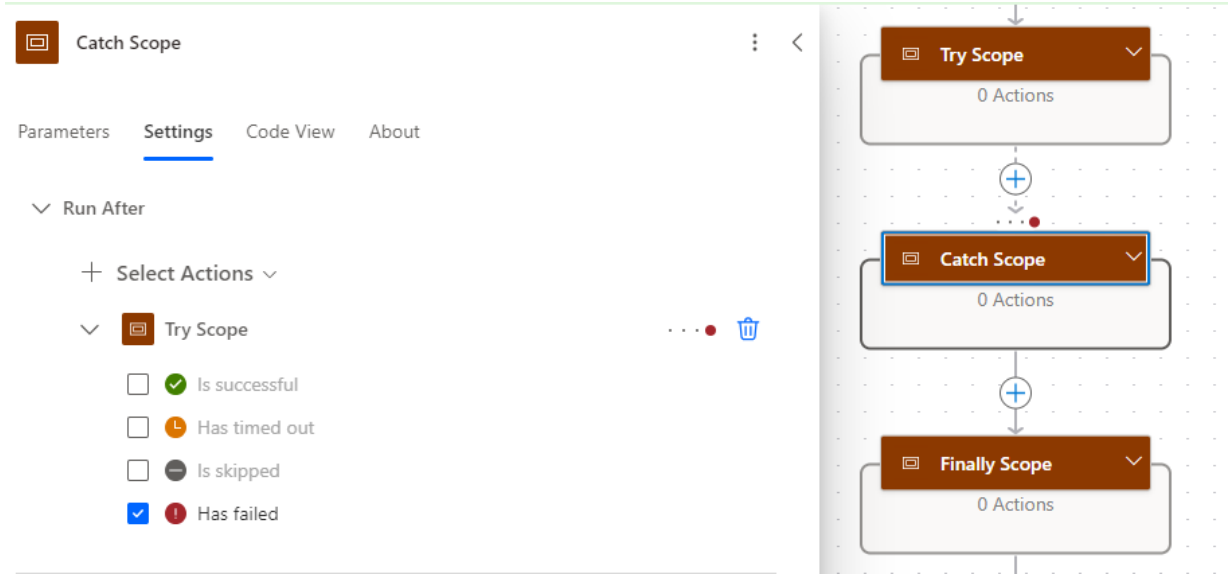
Loops Do-Until

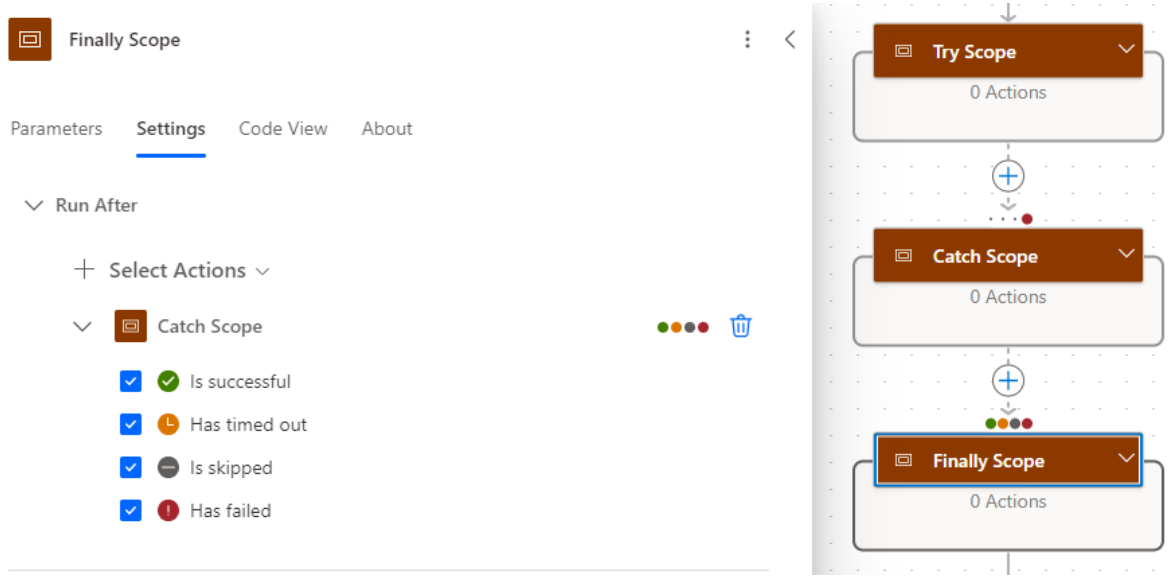
Executes until a condition is met



Scope for Action Grouping

- Catch only runs when Try has failed
- When Try runs successfully, Catch scope is skipped
- Finally runs regardless of the Catch execution including when it is skipped

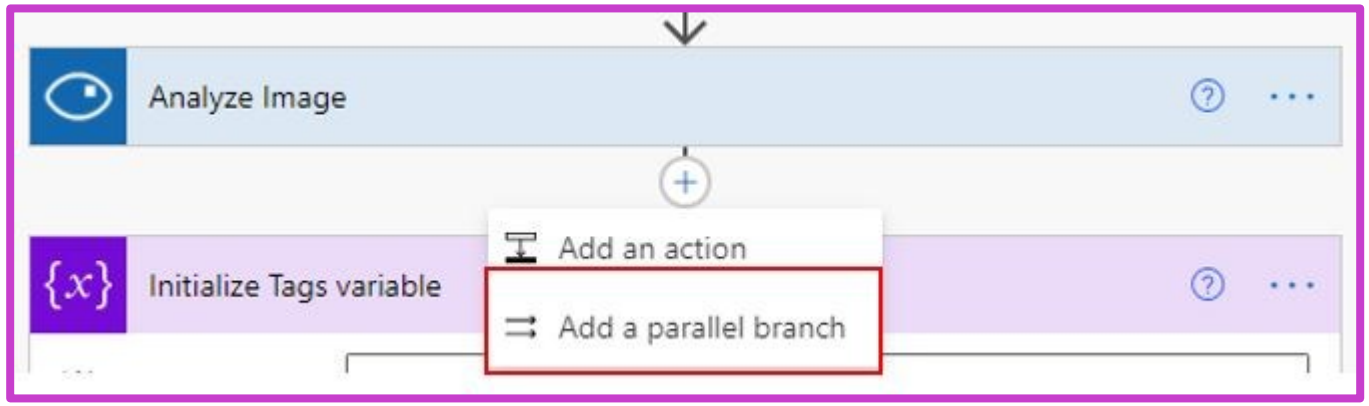




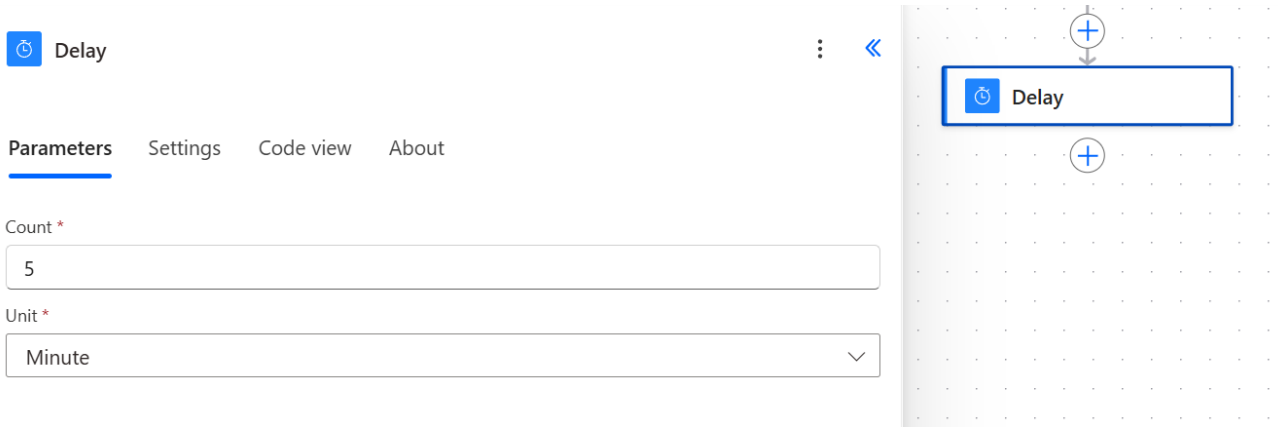
Branching

Parallel branches

- Error handling
- Parallel approvals



Delay / Delay Until / Timeout



TRAINOCATE

Module 4

Advanced Approvals

Approvals actions

Actions

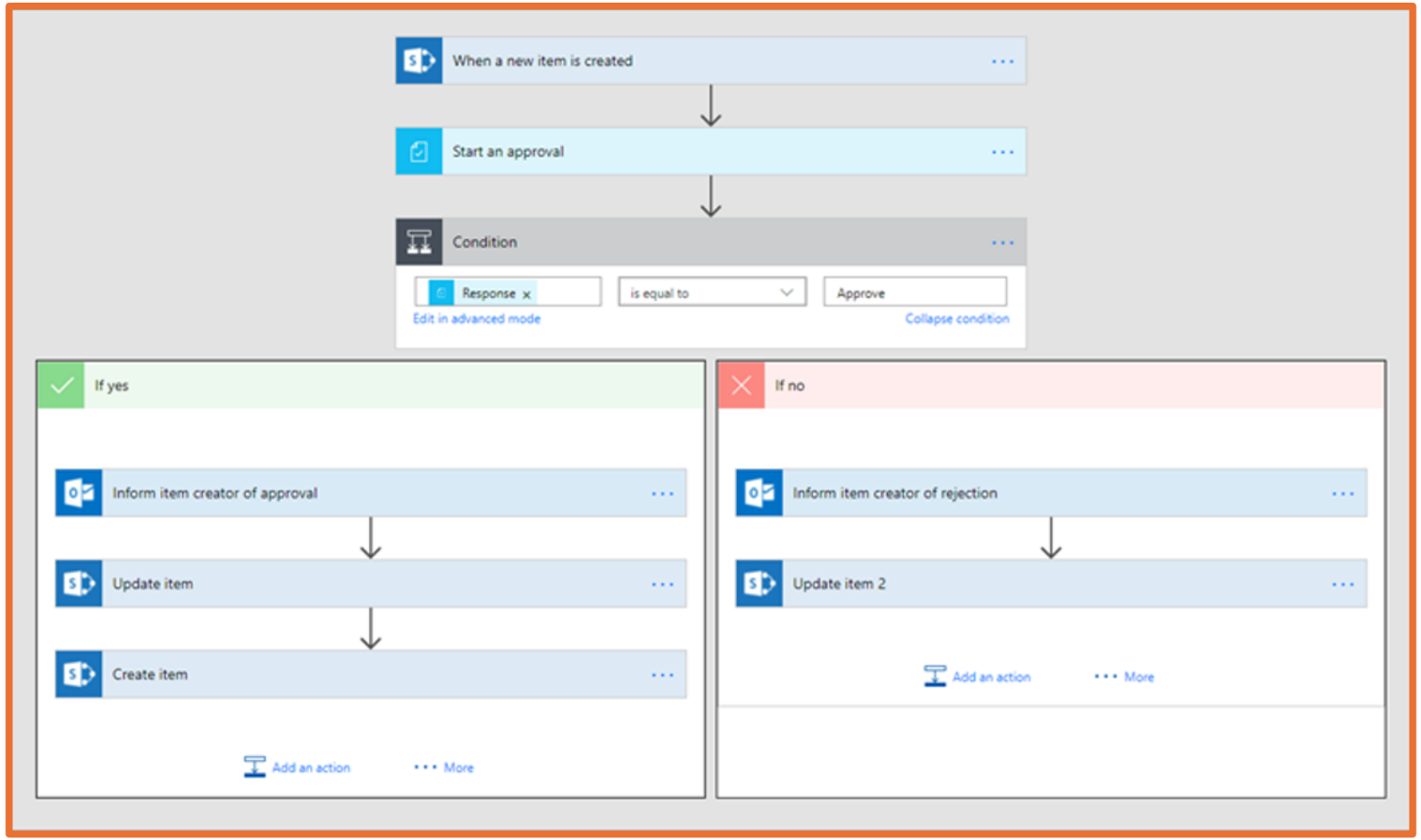
With the approvals capability in Power Automate, you can automate sign-off requests and combine human decision-making for workflows. Some popular cases where approvals can be used include:

- Approving vacation time requests
- Approving documents that need sign-off
- Approving expense reports

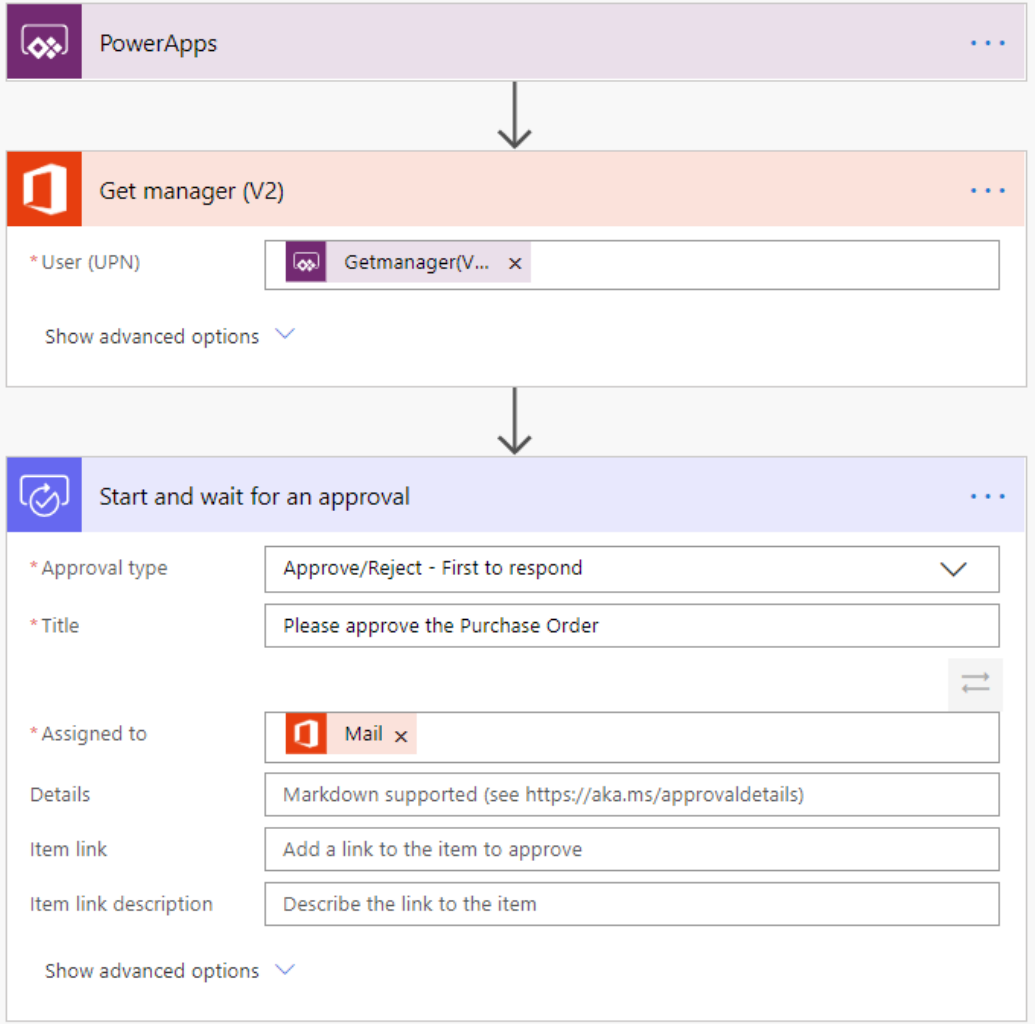
Approvals components

Components

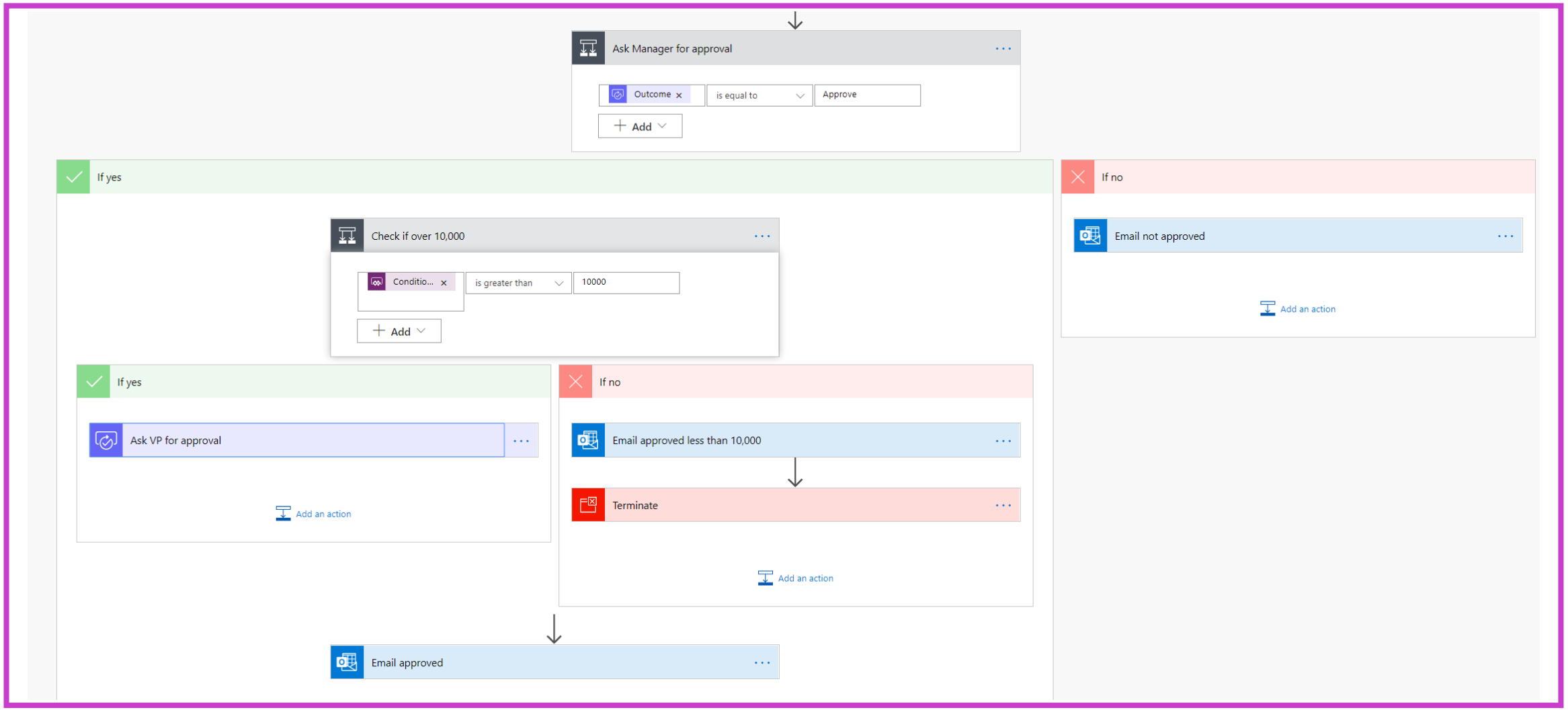
- Approval Action
- Approval Card
- Flow waits for Response
- Condition on Response



Example: Purchase order approval



Example: Purchase order approval

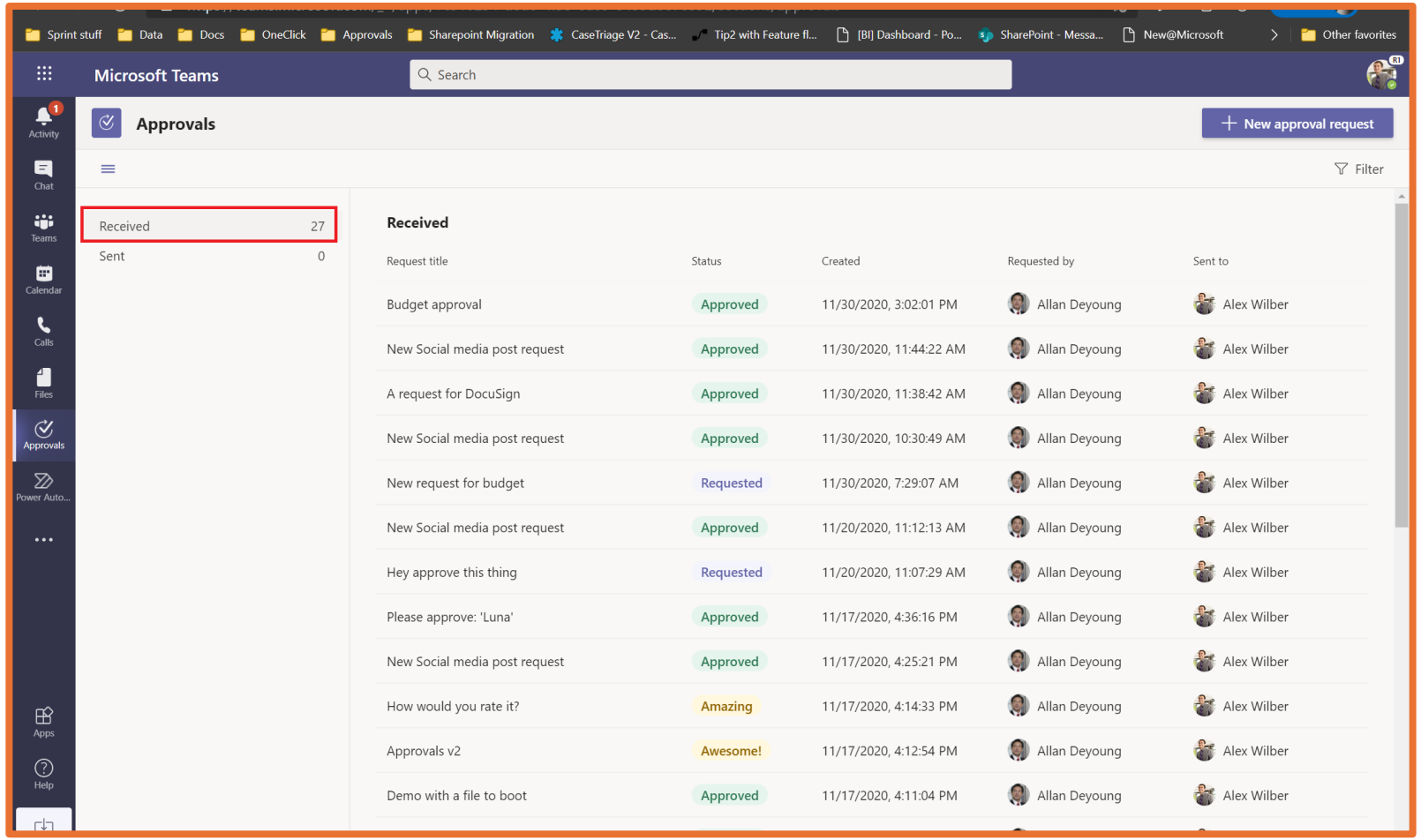


TRAINOCATE

Approvals

Process approvals

- Power Automate portal
- Power Automate mobile
- Microsoft Teams



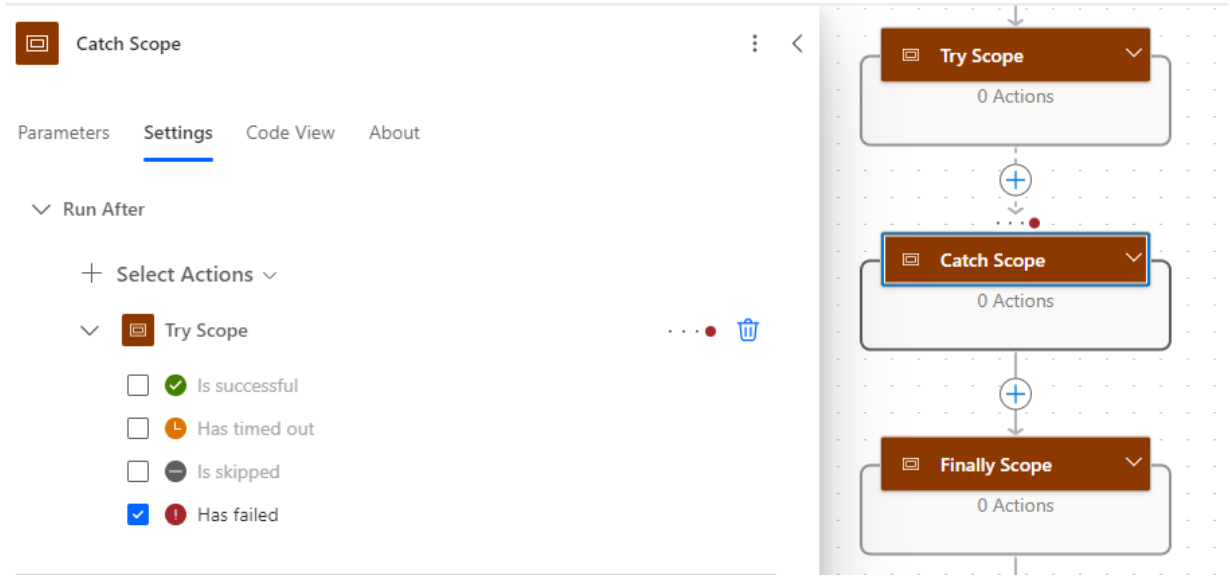
TRAINOCATE

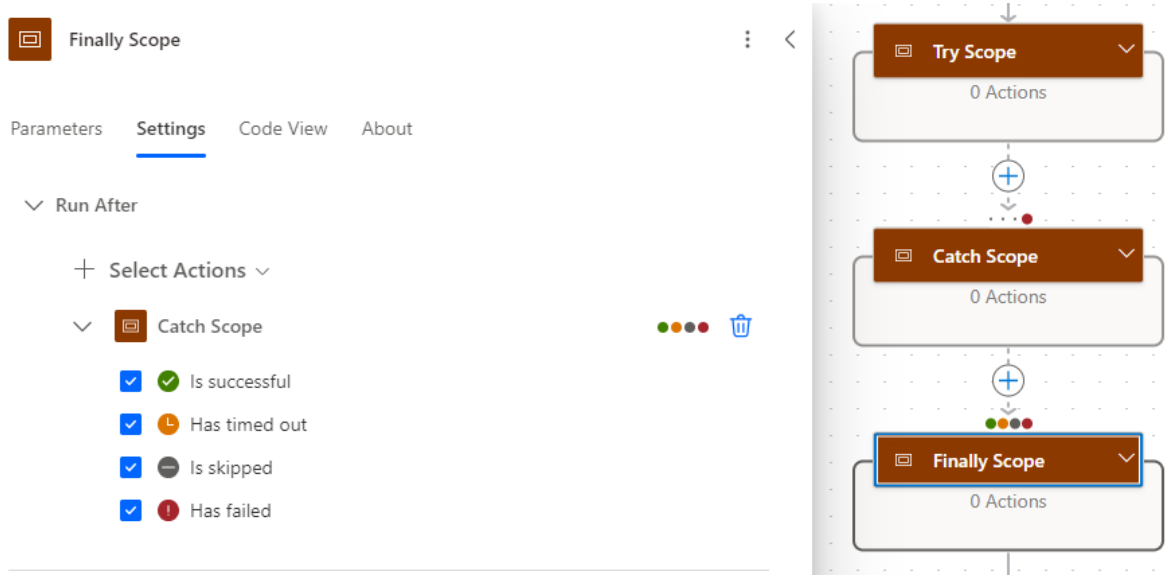
Module 5

Best Practices and Error Handling

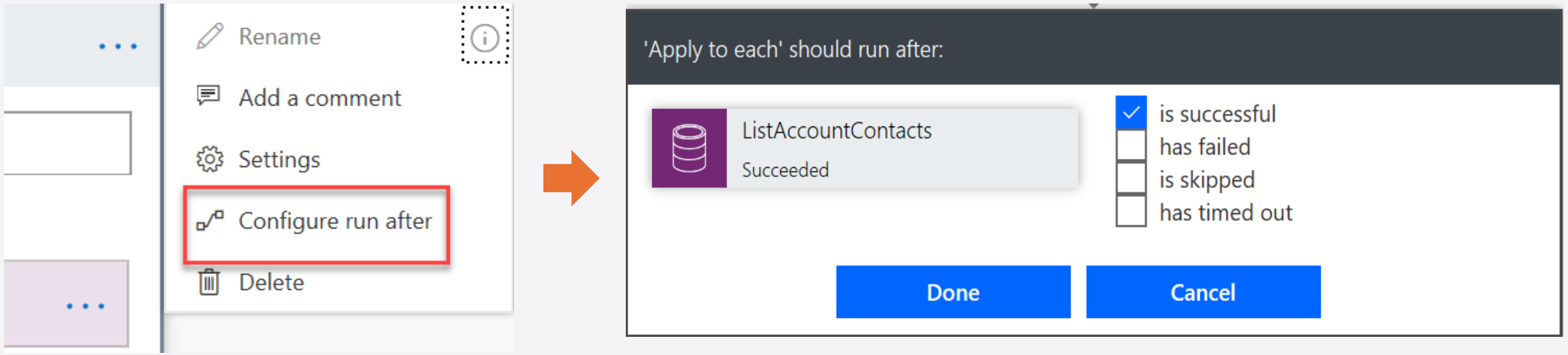
Try – Catch – Finally

- Catch only runs when Try has failed
- When Try runs successfully, Catch scope is skipped
- Finally runs regardless of the Catch execution including when it is skipped





Error handling



Configure run after

By default, flow will keep running through its step as long as prior actions have been successful

Errors

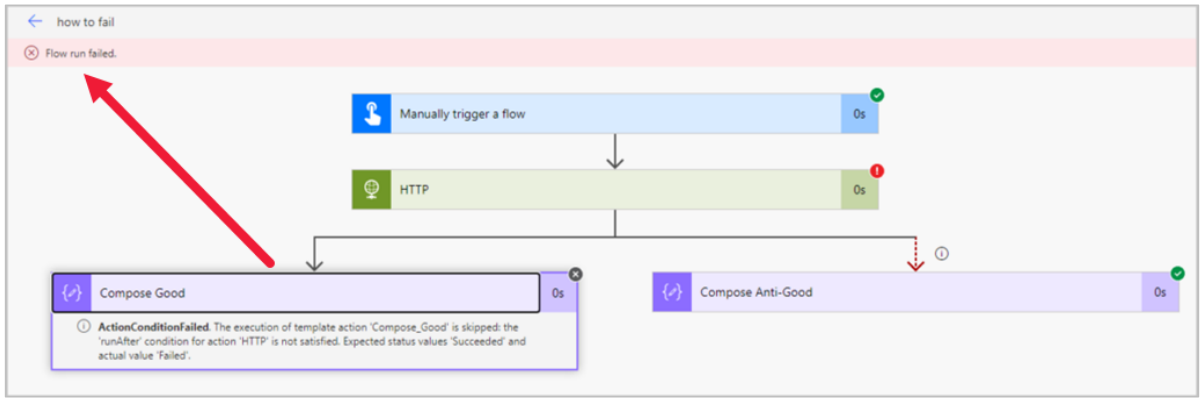
Failed

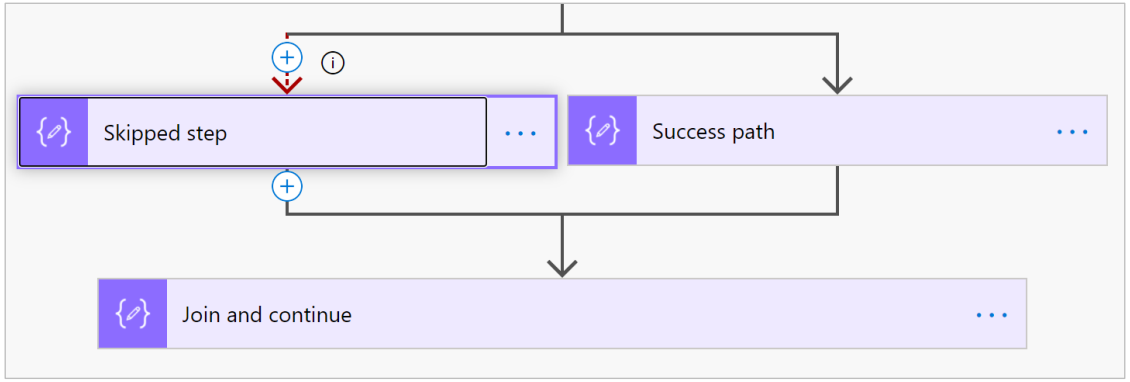
Skipped

Timed out

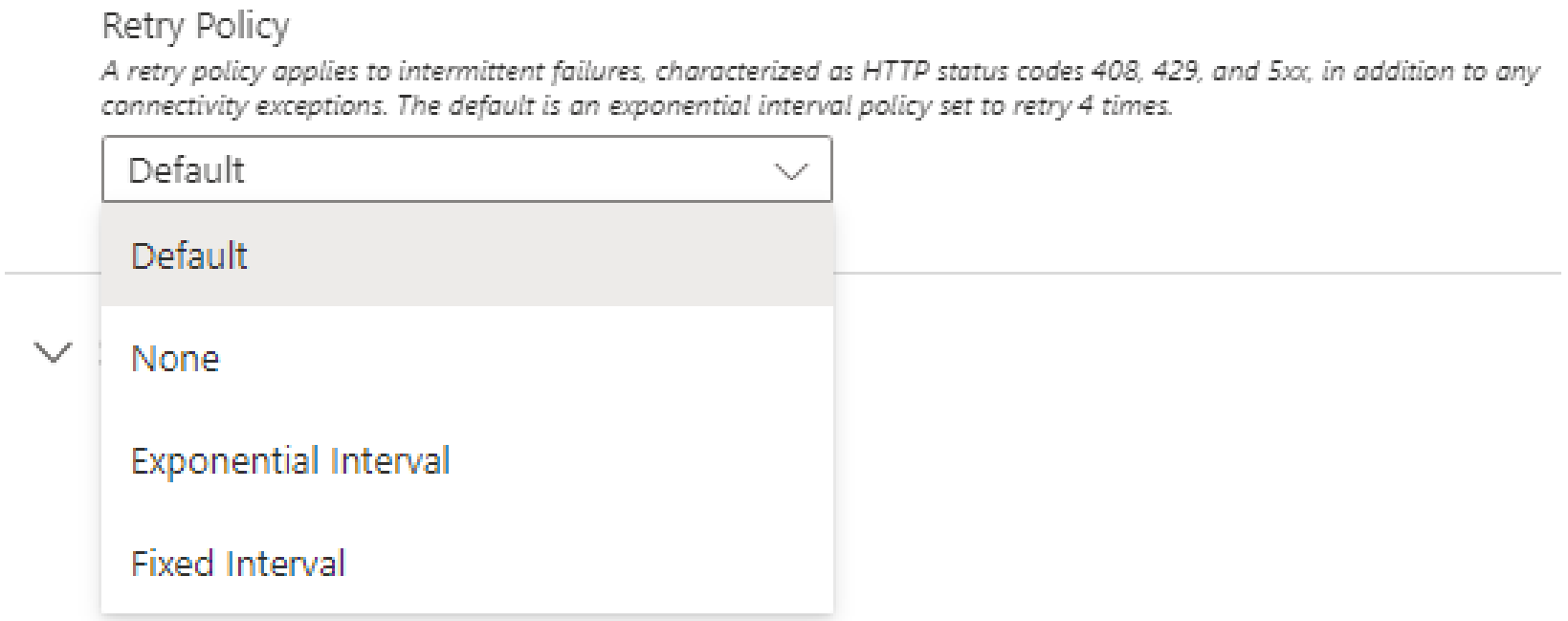
Errors and Flow Success

- For a flow run to be a success, all execution paths must complete
- That includes paths that contain skipped actions
- Solutions
  - Join the paths
  - Terminate the branch





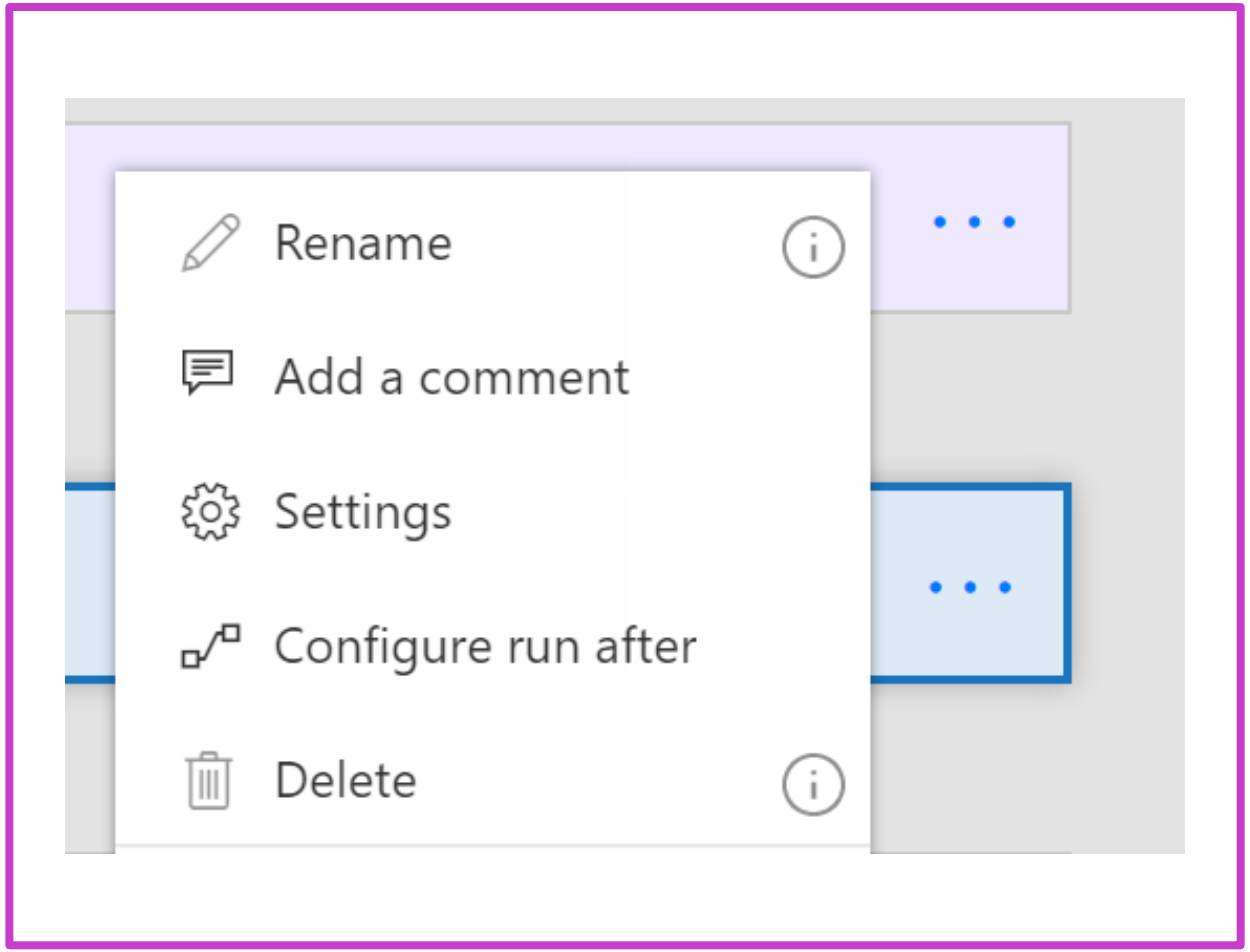
Changing Retry Policy



- Retry policy applies to intermittent failures, characterized as HTTP status codes 408, 429, and 5xx, in addition to any connectivity exceptions
- Adjust the retry policy to handle unique service scenarios

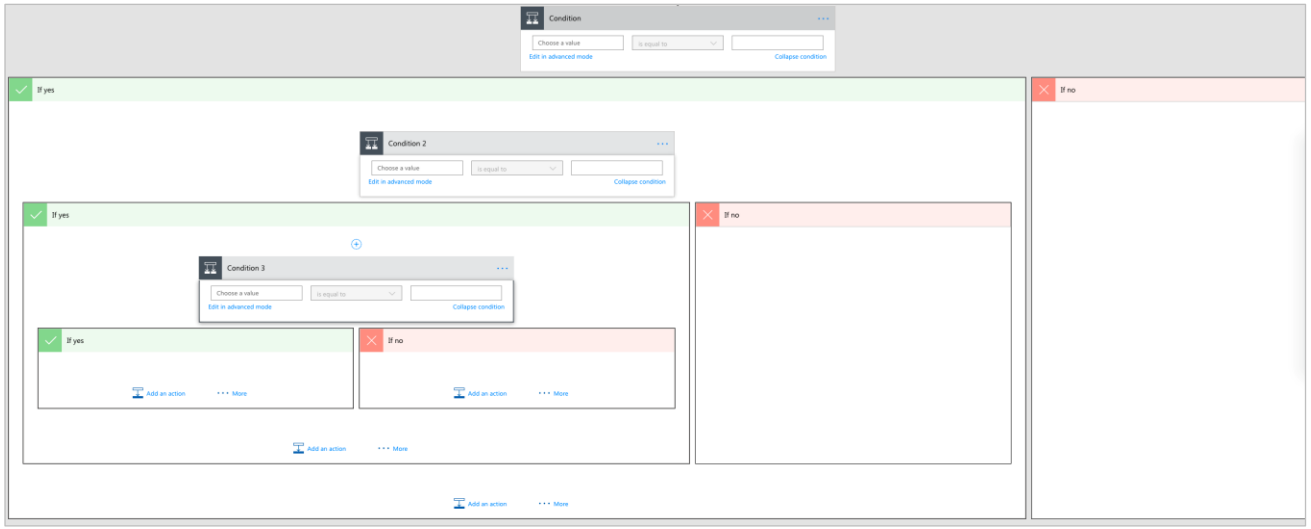
Action Settings

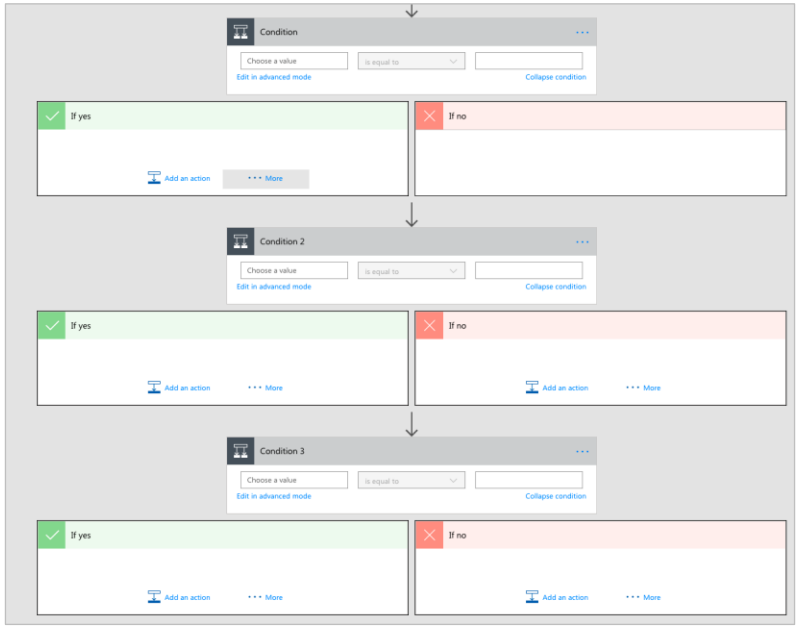
- The "..." menu contains entry points to Settings and Configure run after
- Settings lets you configure
  - Async actions
  - Timeout
  - Retry policy
  - Sequential
  - And more!



Terminate Flows

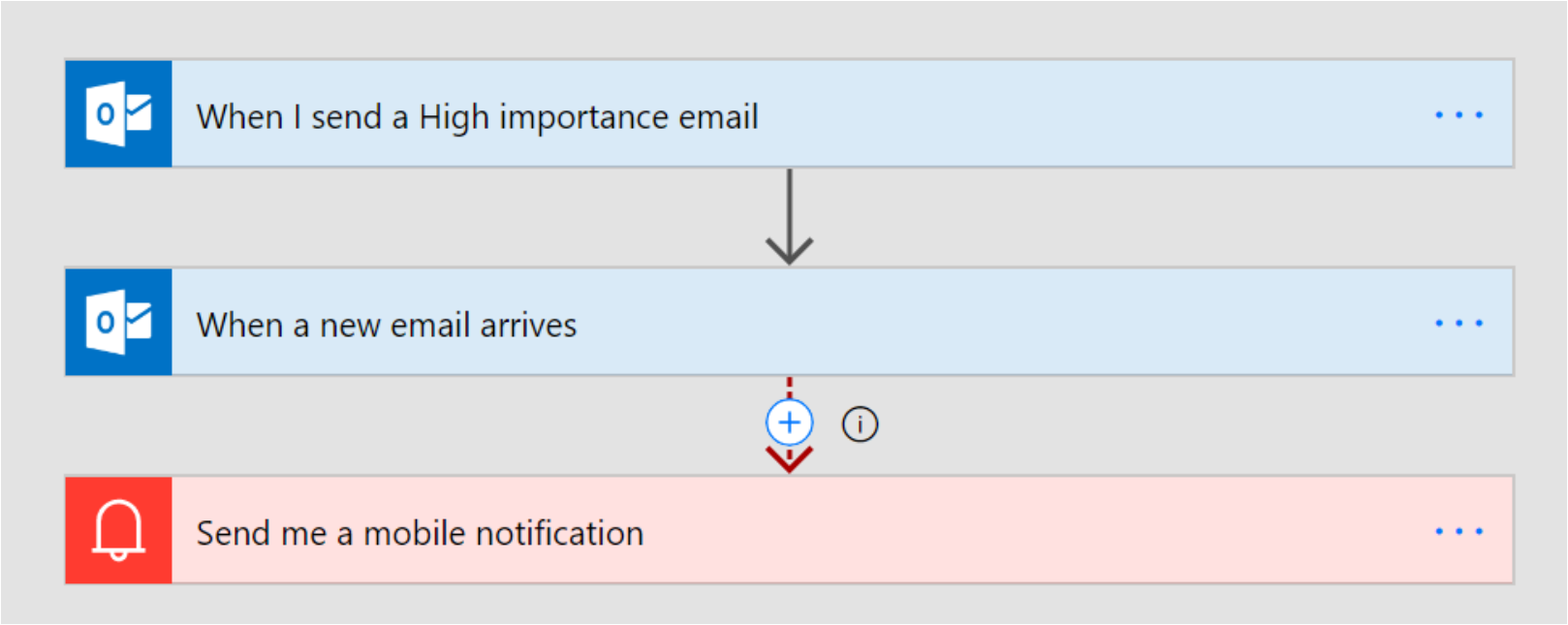
You can use this action to avoid nested conditions by ending in the middle





TRAINOCATE

Asynchronous Actions

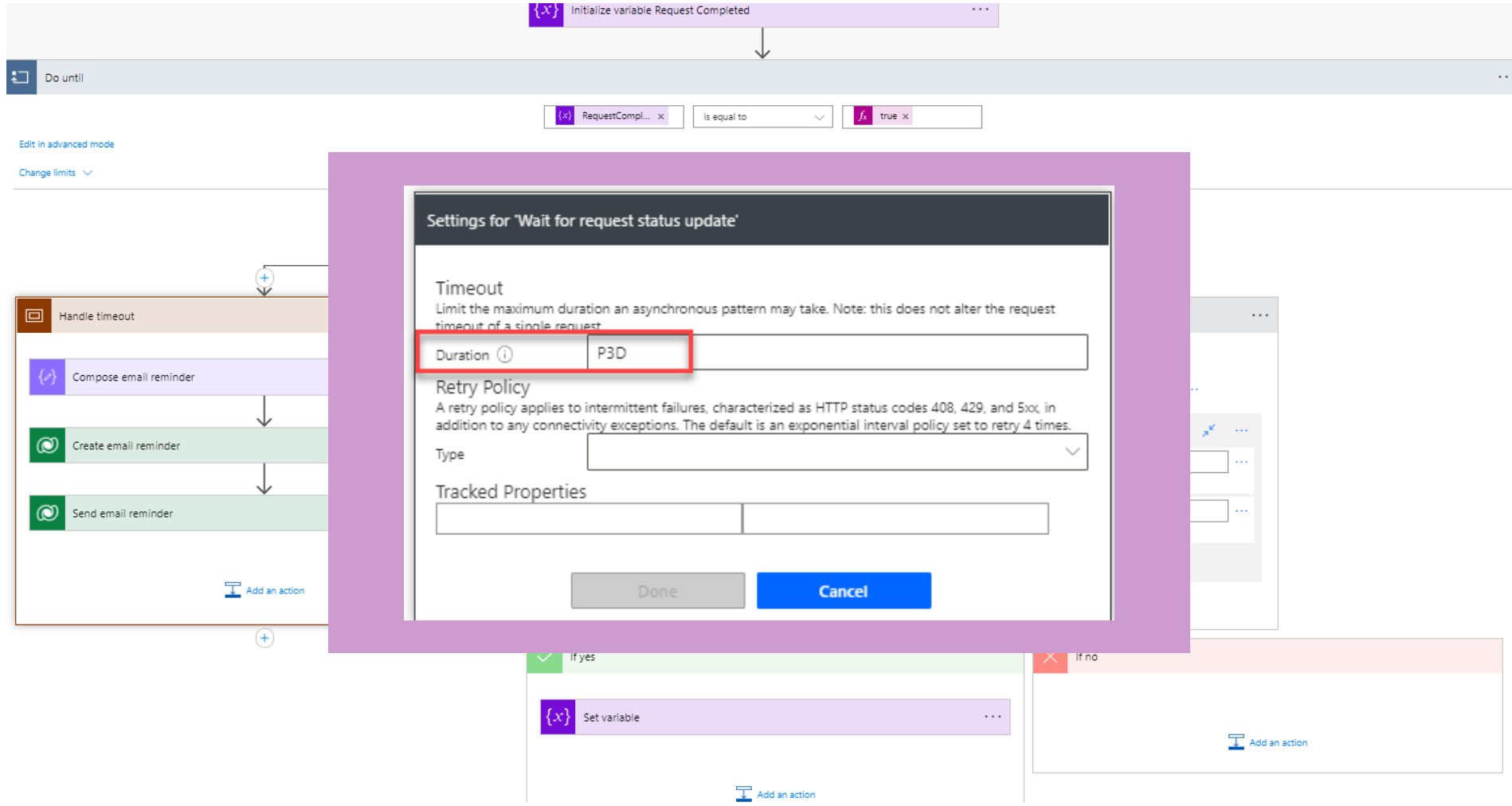


Any “trigger” can be in the middle of a flow as well

- For example, create a flow that waits for a reply to a certain thread in the middle

You probably want to configure a time out and handle that case

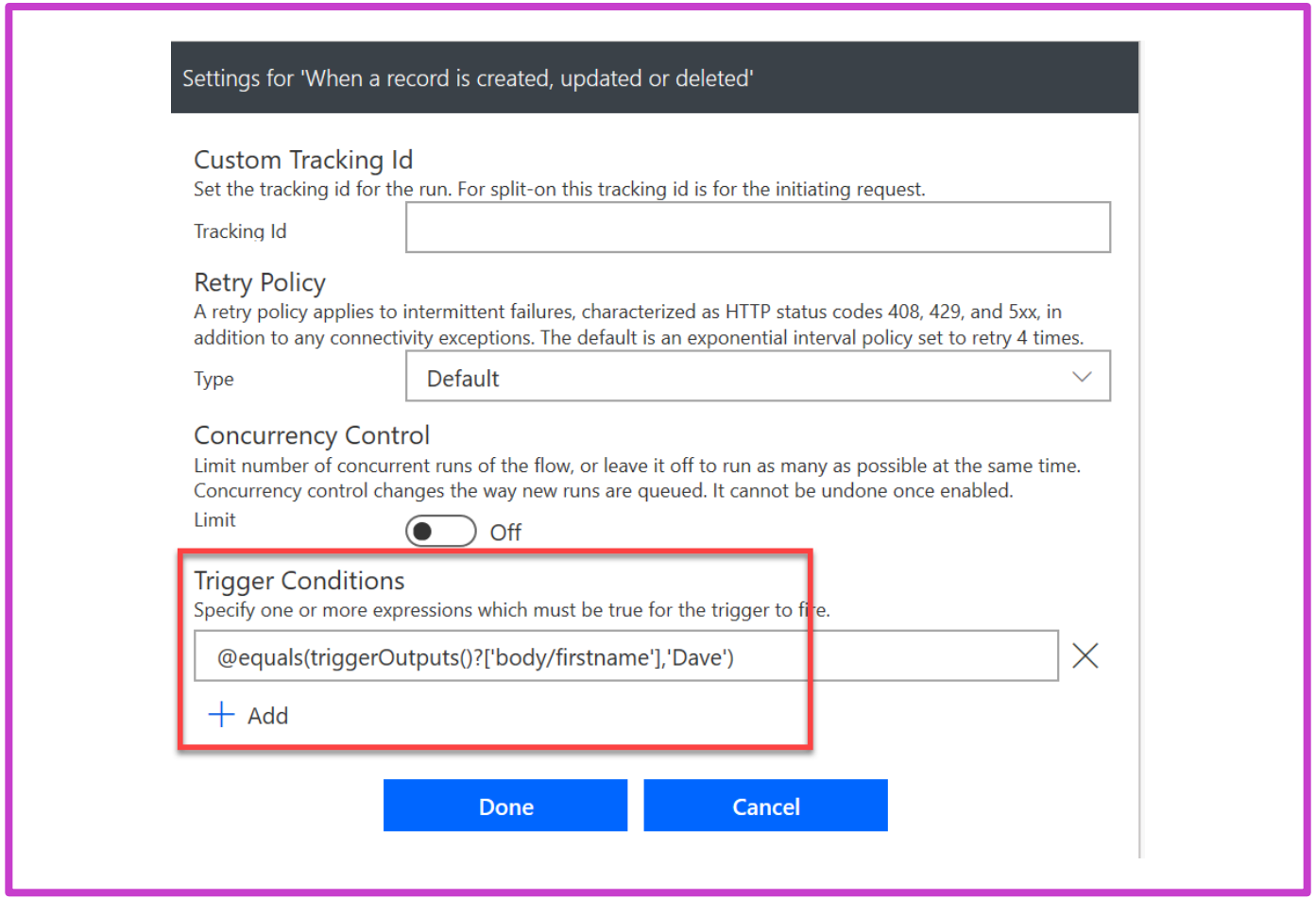
Wait Until Pattern



TRAINOCATE

Using Settings → Trigger Conditions

- Use flow expressions for filter
- Works on create and update
- Not available if trigger is used later in flow



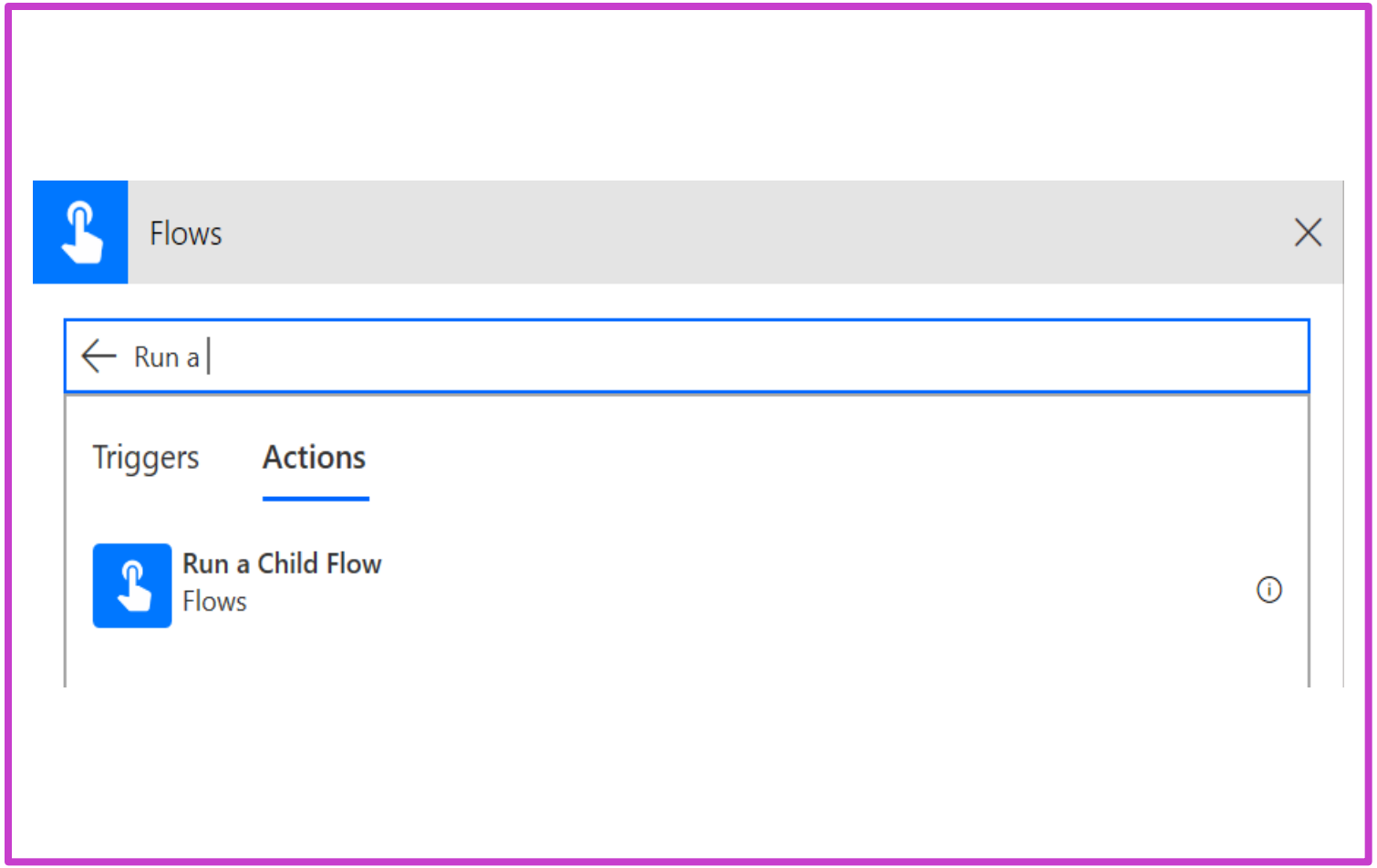
TRAINOCATE

Module 6

Parent and Child Flows

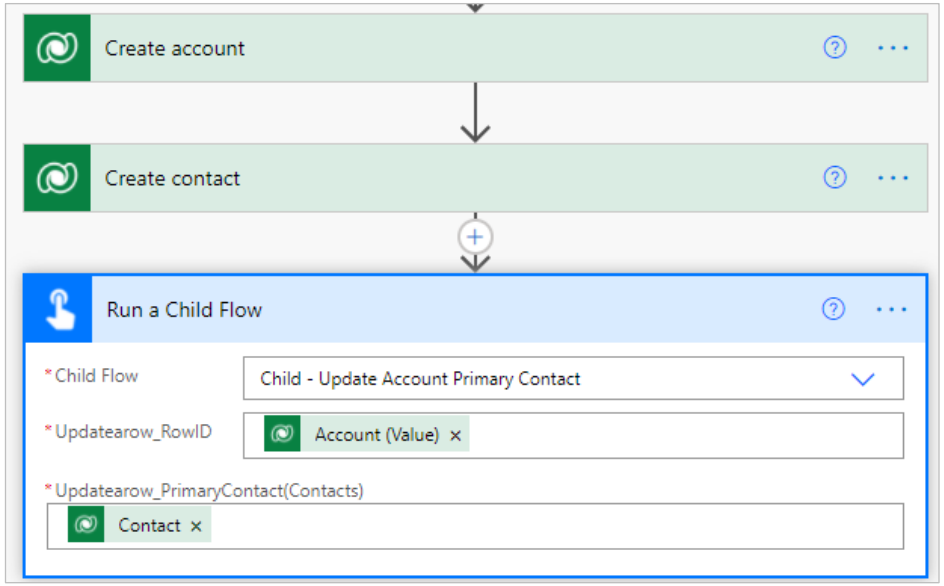
About Child Flows

- Allows breaking out parts of a flow into a reusable child flow
- Trigger support of child flows
  - Manually triggered button flow
  - Power Apps
  - HTTP request
- Respond with Power Apps or HTTP response
  - If no response action, parent won't wait
- Solutions are required for this to work

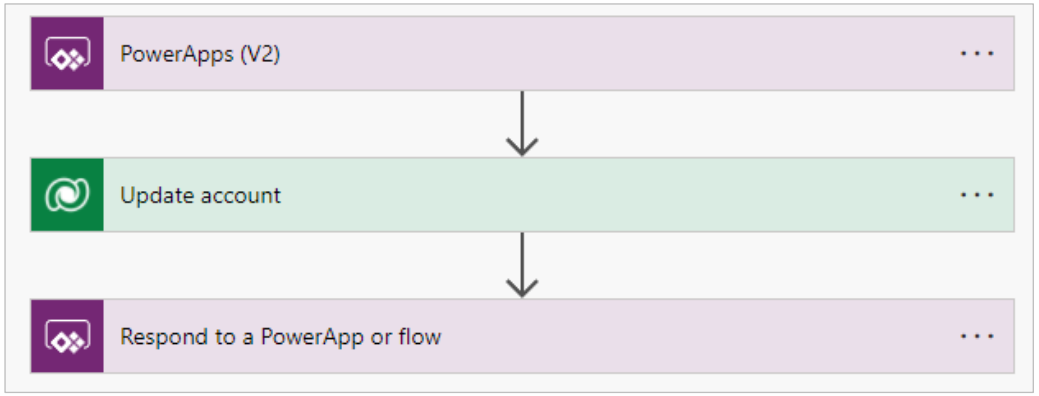


Example – Using a Child Flow

Parent flow



Child flow



TRAINOCATE

Module 7

Governance, Security & Monitoring

Flow Naming Conventions

- **Standardizing Flow Names**
- **Format**: [Environment]-[Module]-[Function]-[Version]
- **Examples**:
  - PROD-Finance-PurchaseOrderApproval-v1
  - DEV-HR-LeaveRequestProcessing-v2
  - TEST-IT-UserProvisioning-v1

- **Best Practices**:
- Use names that clearly describe the function
- Avoid names that are too short or too long
- Include a Version Number

Action Naming

- **Clearly Naming Actions**
- **Format**: [Action Type] - [Description]
- **Examples**:
  - HTTP Request - Get User Data
  - Parse JSON - Parse Order Response
  - Condition - Check Approval Status

- **Best Practices**:
  - Name the action to describe its operation.
  - Avoid default names like "Condition 2"

Variable Naming

- **Standardizing Variable Names**
- **Format**: [Prefix][Description]
- **Prefixes**:
  - str: String (e.g., strCustomerName)
  - int: Integer (e.g., intOrderCount)
  - bool: Boolean (e.g., boolIsApproved)
  - arr: Array (e.g., arrOrderItems)
  - obj: Object (e.g., objOrderData)

Environments

- **Environment Types**:
  - **Development**: For development and initial testing.
  - **Test/UAT**: For User Acceptance Testing.
  - **Production**: For live, operational usage.
- **Best Practices**:
  - Use separate Environments aligned with the SDLC (Software Development Life Cycle).
  - Limit Access based on Role.
  - Utilize Environment Variables for Configuration.

Solutions

- **Managing and Deploying Flows**
- **Purpose**: To manage and deploy Flows between Environments.
- **Solution Structure**:
  - **Components**: Flows, Connections, Variables.
  - **Dependencies**: External Dependencies.
  - **Versioning**: Version Management.

- **Deployment Process**:
  - Create Solution in Source Environment.
  - Add Components.
  - Export Solution.
  - Import to Target Environment.
  - Configure Connections and Variables.

Environment Variables

- **Storing Configuration Values**
- **Use Cases**:
  - API Endpoints
  - Connection Strings
  - Feature Flags
  - Threshold Values
- **Benefits**:
  - Environment-Specific Configuration.
  - Easy Updates.
  - Security (Sensitive Data).

TRAINOCATE

Security Roles

- **Defining Security Roles**
- **Security Roles**:
  - **Flow Owner**: Flow owner with modification rights.
  - **Flow User**: User who can run the Flow.
  - **Environment Admin**: Environment maintainer/administrator.
  - **Solution Admin**: Solution maintainer/administrator.

Access Control

- **Setting Access Permissions**
  - **Share Flow**: Share Flow with Users or Groups.
  - **Permissions**:
    - **Can Edit**: Can modify the Flow.
    - **Can View**: Can view the Flow.
    - **Can Run**: Can execute the Flow.
- **Best Practices**:
  - Apply the **Principle of Least Privilege**.
  - Use **Groups** instead of Individual Users.
  - Review Permissions regularly.

TRAINOCATE

Optimization: Reduce Nested Loops

- Before

Apply to Each: items

Apply to Each: subItems

Apply to Each: details

Process Detail

- After

Filter Array: items where status = "Active"

Select: Transform to required format

Apply to Each: filteredItems

Process Item

TRAINOCATE

Optimization: Reduce Nested Loops

- Before

Apply to Each: allItems

Condition: If item.status = "Active"

Process Item

- After

Filter Array: allItems where status = "Active“

Apply to Each: activeItems

Process Item

Before we finish
Today's class

Take Class Evaluation : Click to Evaluation

Scan now :

TRAINOCATE

THANK YOU

The Offices at CentralWorld, 16 Floor
Pathumwan, Bangkok, 10330, Thailand

(+66) 2 613 1033

thailand@trainocate.com

www.trainocate.com