Lab 15. (Approval Advanced) Implementing the Flow Controller Pattern

Author: Serge Luca aka "Doctor Flow"

Updated by: Dattatray-Patil

Learning objective: Create state machines and have Flows that can run beyond the 30 days limitation by using the Flow Controller pattern.

Duration: 30 minutes.

Scenario: When a user submits a new expense in a SharePoint list, the expense will have to be approved by a first user (line manager); if this user doesn't react on time, another user (big boss) will react. If the big boss doesn't react on time, the system will ask the big boss to react again and again. The Flow should work even if the whole process takes more than 30 days (current Flow limitation).

Tasks:

This lab illustrates an implementation of the Flow Controller pattern with the HTTP action. This requires a premium license, but an implementation with a flow started from a message stored into a SharePoint list will be available soon and has been tested years ago.

This controller pattern Flow provides more flexibility in the Flow architecture design; we will apply this concept to have Flows that can run beyond the 30 days limitation (Flow run and approval), but also to implement state machines even though the current Flow designer doesn't support state machines yet.

In this lab, we will create 3 Flows:

- The launcher Flow
- The controller Flow
- The generic approval Flow

Setup

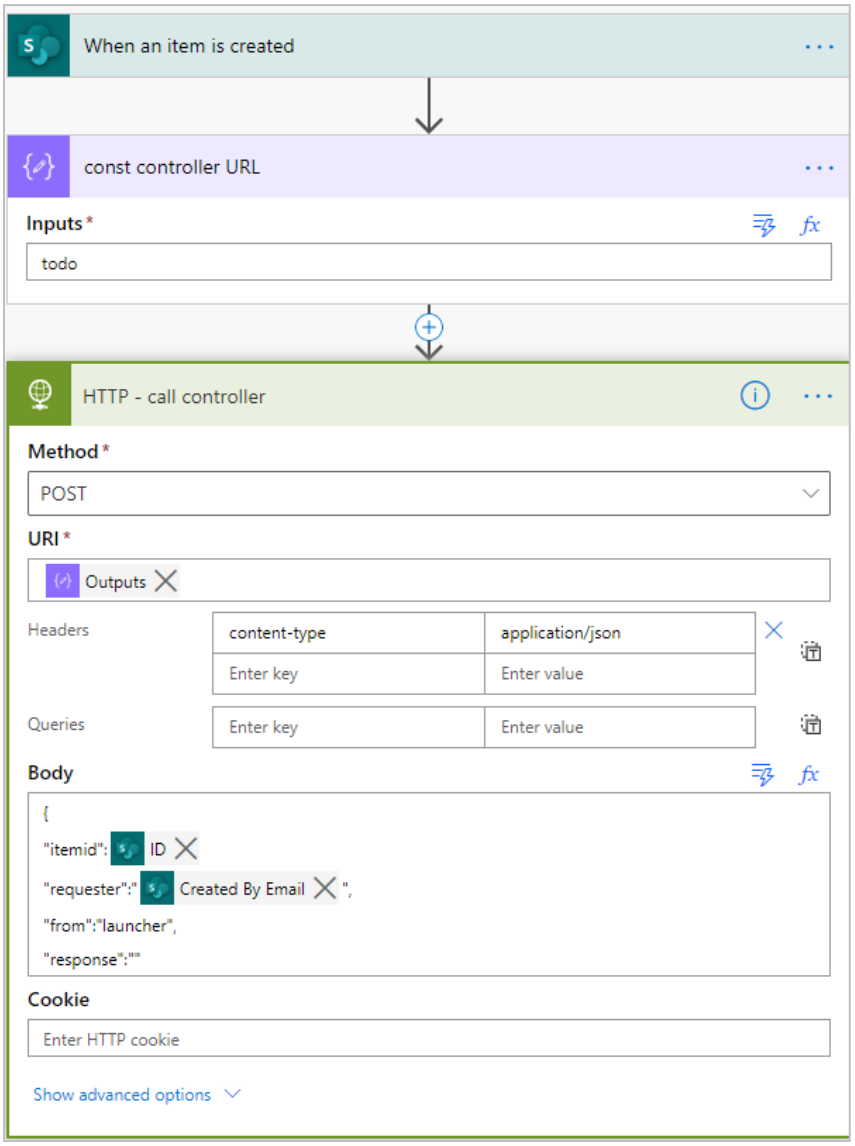
1. Create a SharePoint list named "**Expenses**" with 3 columns: the default column (**title**), a column **Amount** (type number) and a column named **Status** (choice with the values **In progress**, **Rejected**, **Accepted**).

Launcher Flow implementation

2. Create a Flow named **Approval launcher** that will start when a new expense is submitted in this list.
3. The trigger of this Flow is **When an item is created** (SharePoint Connector)
4. Add a **compose** action, call it **const controller URL** and set its value to "todo". We will update it afterward.
5. Add an **HTTP action**, name it **HTTP - Call Controller**
    a. Set the method POST
    b. Set the URI to the Output of **const controller URL**

c. In the Headers set **content-type** to **application/Json**
d. Set **Body** like as described below:

You can notice that ID is not surrounded by double quotes because it is an integer.



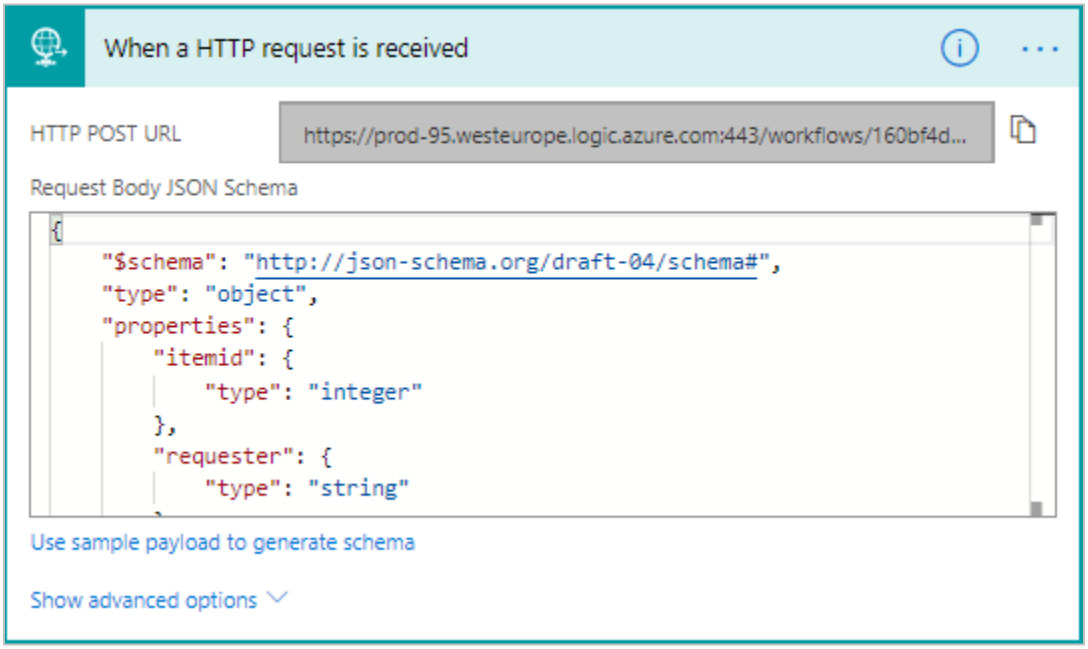
The Flow controller will be implemented in the last place. Let us focus on the Approval logic.

Approval generic Stage implementation

1. Create a new Flow called **Approval generic stage** that starts with the trigger **When an HTTP request is received**.
2. In the Request body of the trigger, copy and paste the following JSON schema:

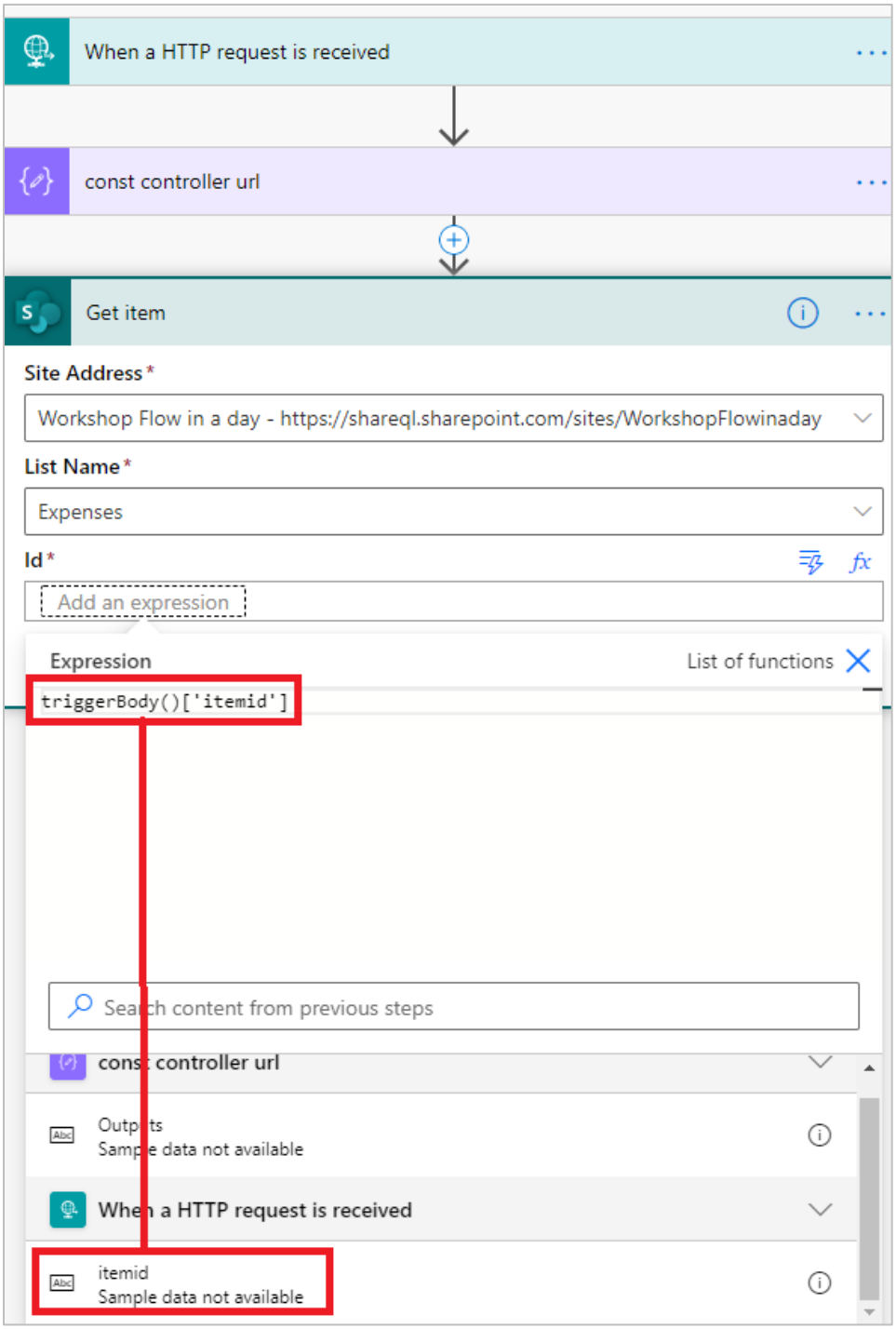
```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "itemid": {
      "type": "integer"
```

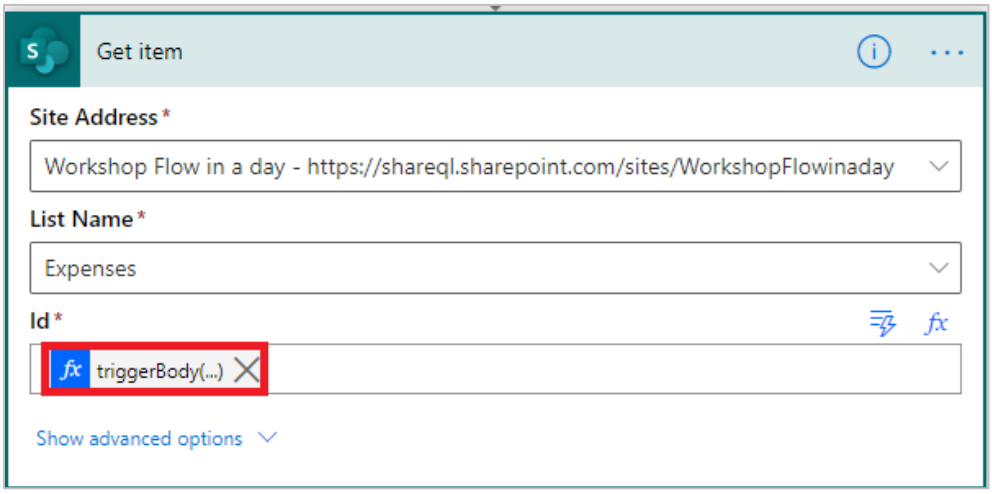
```
        },
        "requester": {
            "type": "string"
        },
        "approver": {
            "type": "string"
        },
        "stagename": {
            "type": "string"
        }
    },
    "required": [
        "requester",
        "itemid",
        "approver",
        "stagename"
    ]
}
```



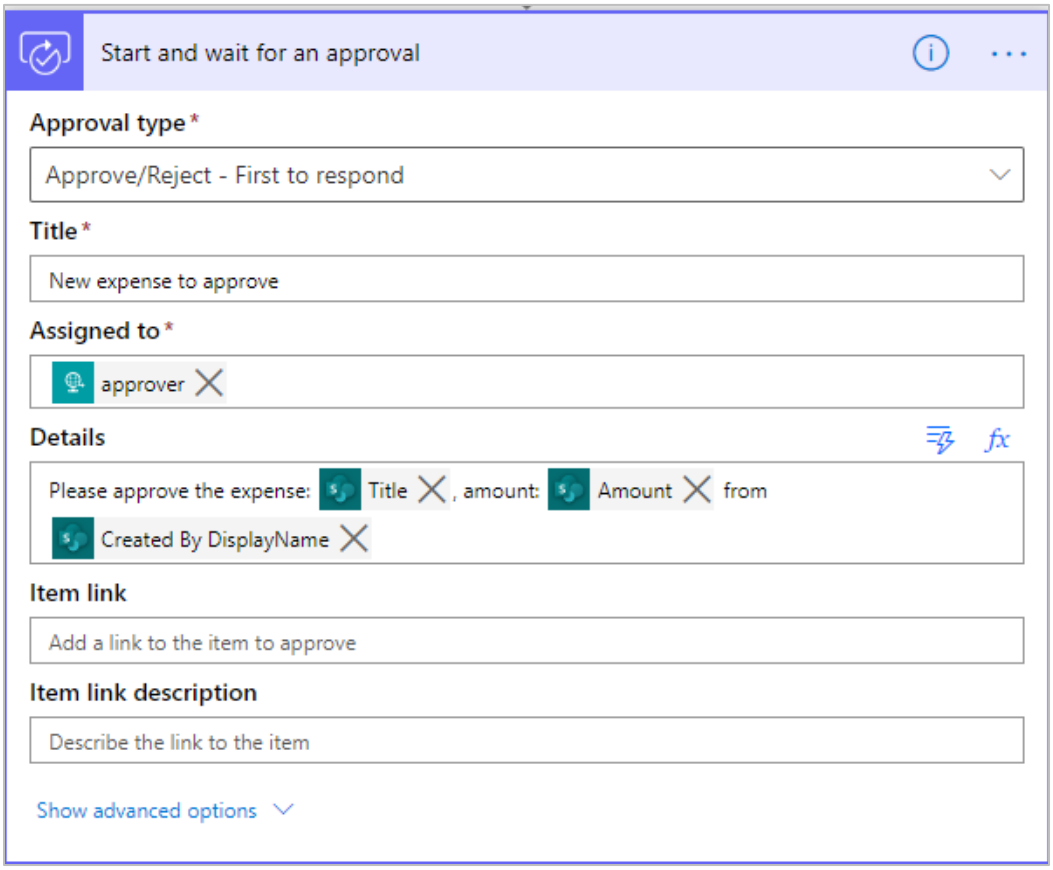
3. Add a **Compose** action named **const controller URL**. We still have to generate the controller; in the meantime, store a string “todo” in this action.

4. Add a SharePoint **Get item** action to retrieve you expense details; the id we must be grabbed from the trigger “itemid” parameter:


When a HTTP request is received
const controller url
Get item
Site Address *
Workshop Flow in a day - https://shareql.sharepoint.com/sites/WorkshopFlowinaday
List Name *
Expenses
Id *
Add an expression
Expression
List of functions
triggerBody()['itemid']
Search content from previous steps
const controller url
Outputs
Sample data not available
When a HTTP request is received
itemid
Sample data not available



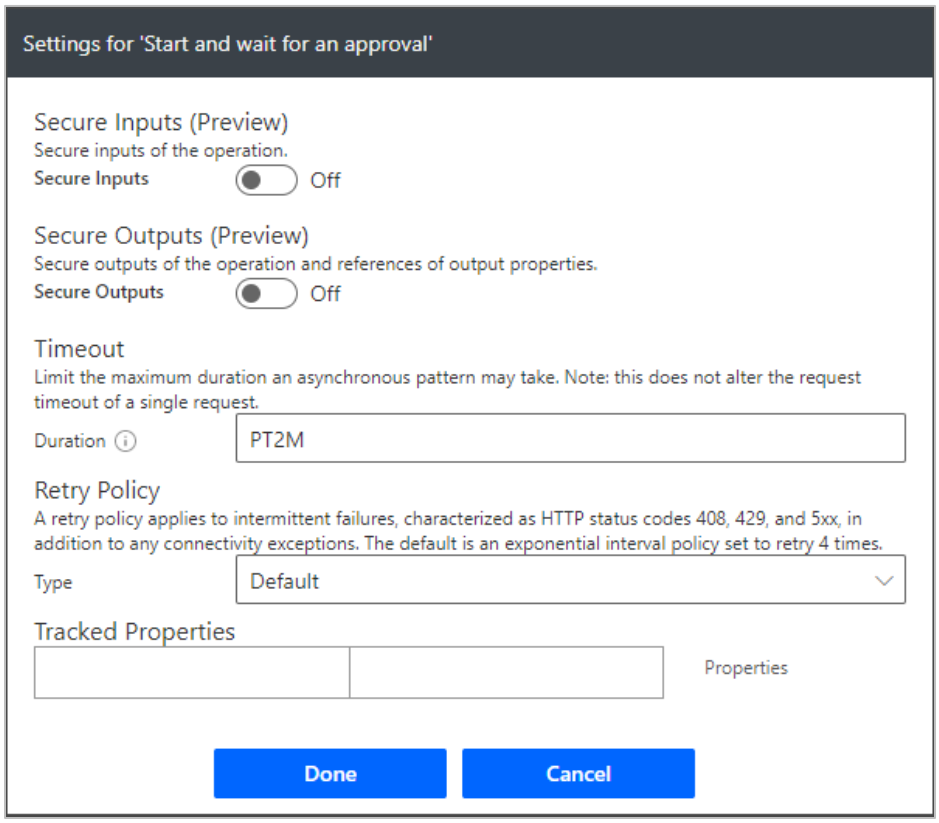
5. Add a **Start and wait for an approval action** and ask the line manager to approve:



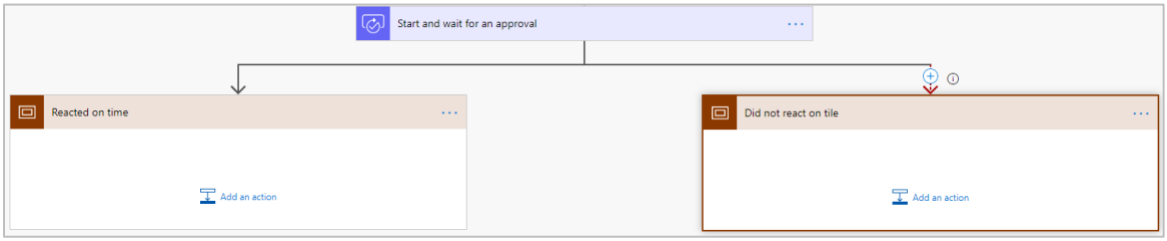
There are now 2 options: the approver reacts on time, or he does not.

a) If he reacts on time, the process is completed.
b) If he does not react, the approval will be escalated to the big boss.

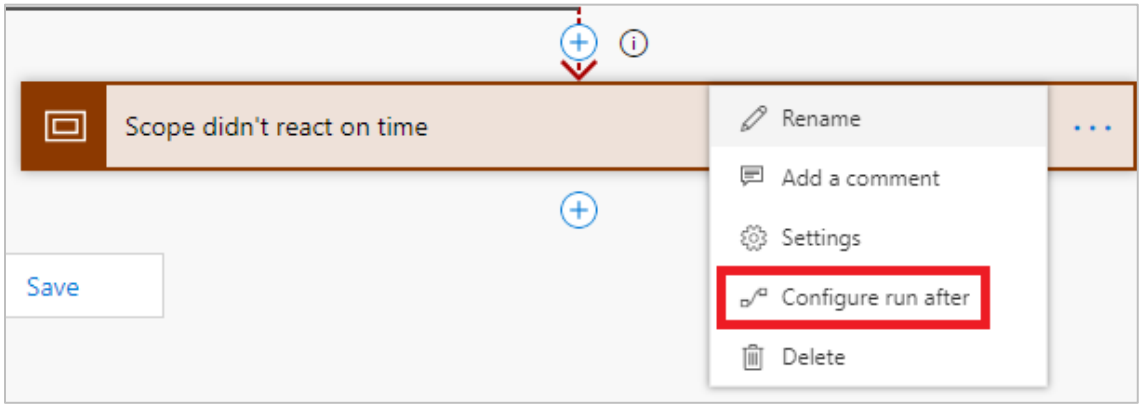
6. Select the **Start and wait for an approval** action setting and set the timeout to PT2M (2 minutes):

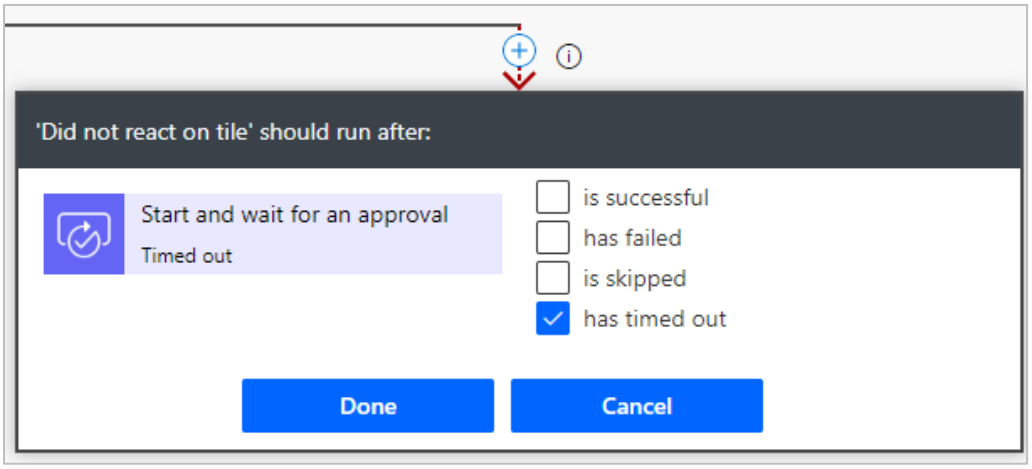


7. Add a parallel branch with 2 scopes and rename them accordingly:

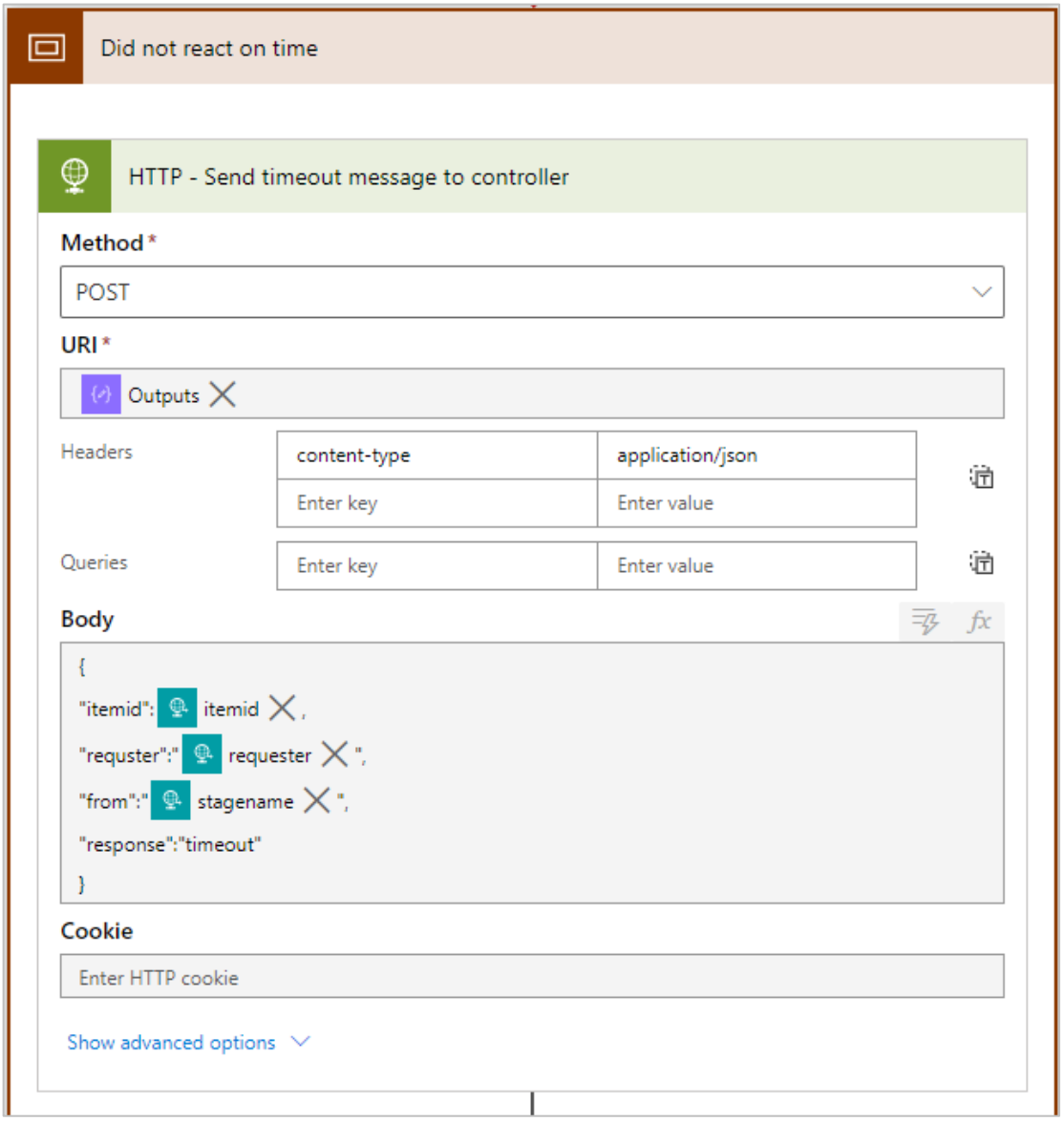


8. Select the "Scope didn't react on time" **Configure run after**:

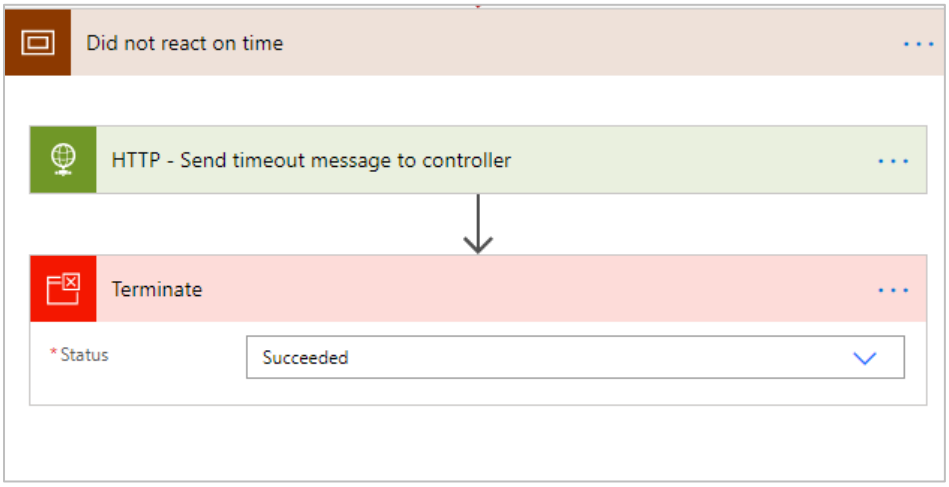




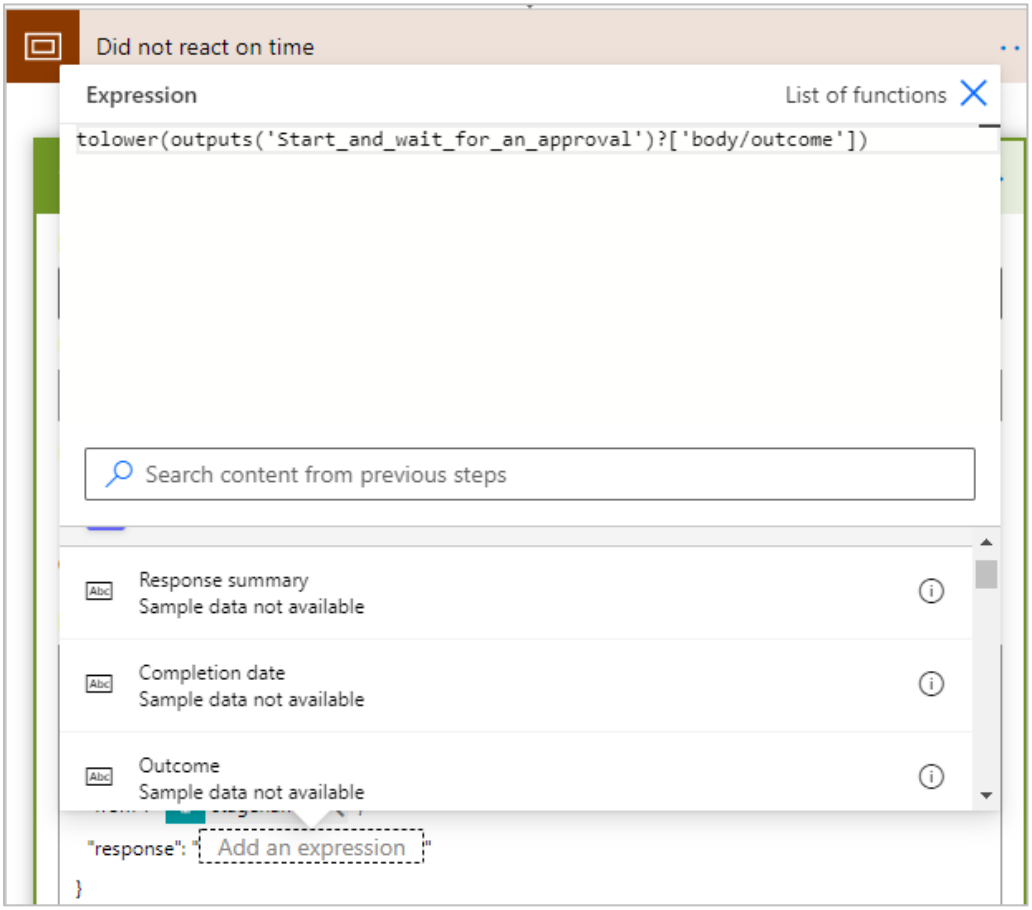
9. The check **has timed out**.
10. In the same scope, add an **HTTP** action and set its settings as follows:

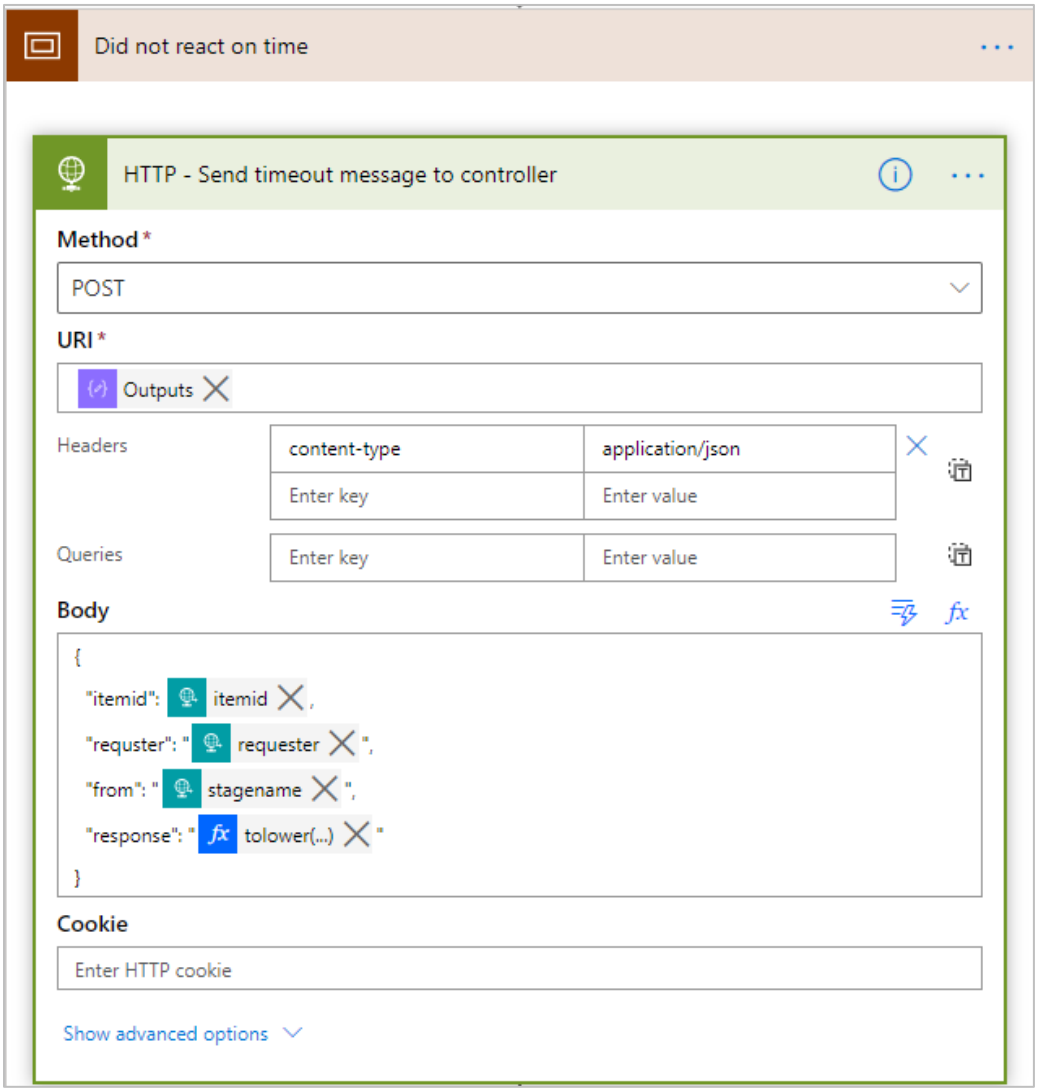


11. Add a **Terminate** action with a Succeeded status:

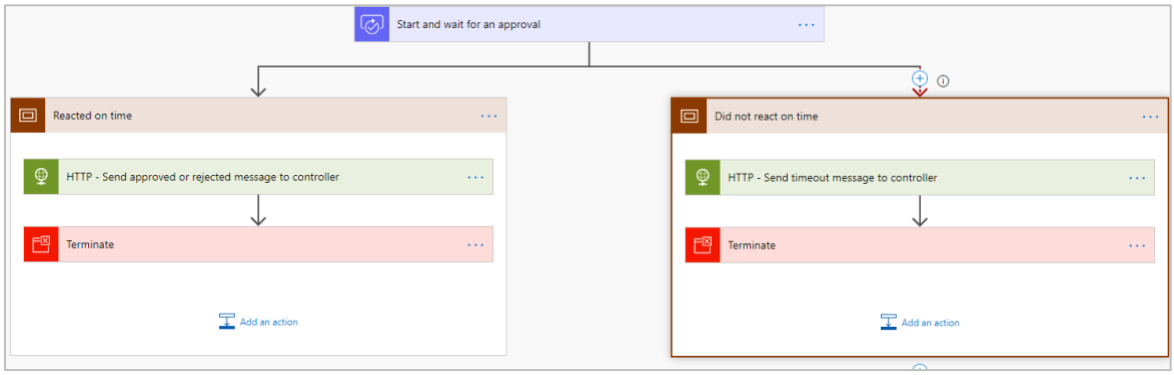


12. In the other scope, add an **HTTP** action (or copy the one you just created), but this time the "response" value should be grabbed from the Approval outcome:





13. Add a Terminate action just after the HTTP action.
14. The 2 branches should look like this:



15. Save the Flow.

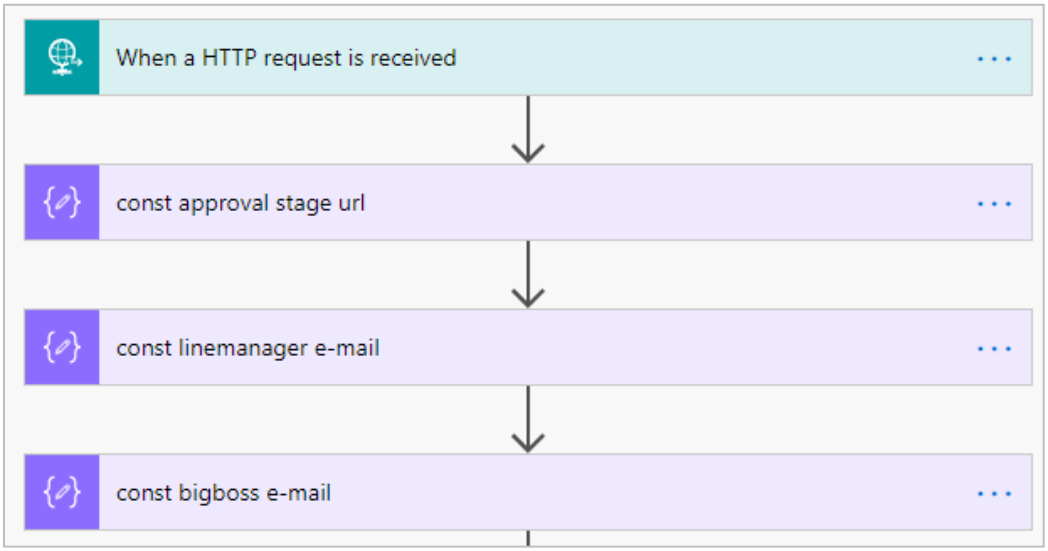
Approval Controller Flow implementation

1. We will now create the Flow controller Flow. Create a Flow called **Approval Controller**.
2. The trigger of this Flow must be **When HTTP Request is received**

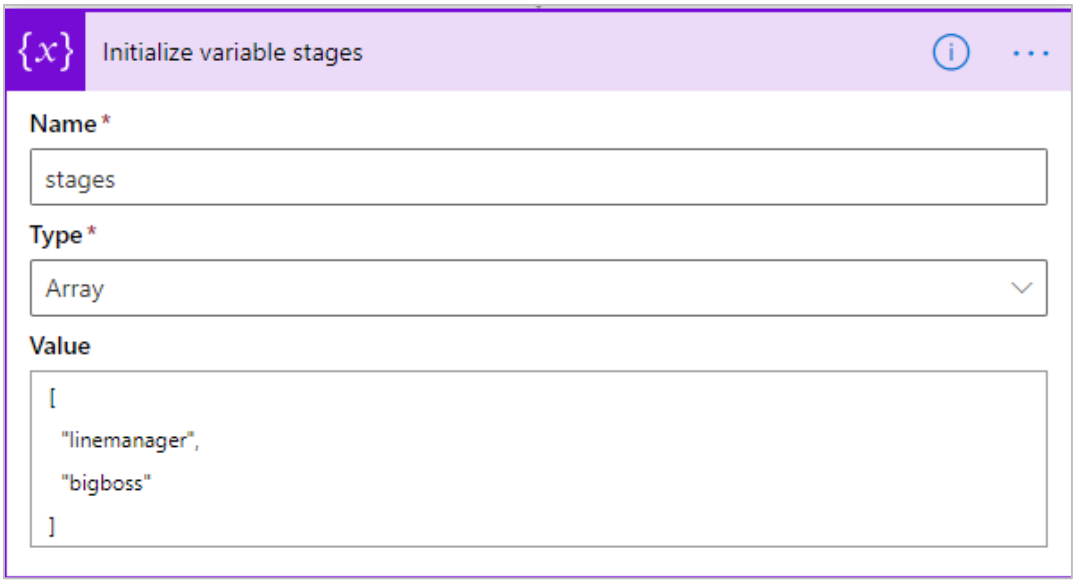
3. Copy and paste the following schema into the request trigger JSON schema:

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "itemid": {
      "type": "integer"
    },
    "requester": {
      "type": "string"
    },
       "from": {
         "type": "string"
       },
       "response": {
         "type": "string"
       }
    },
    "required": [
      "requester",
      "itemid",
      "from"
    ]
  }
```

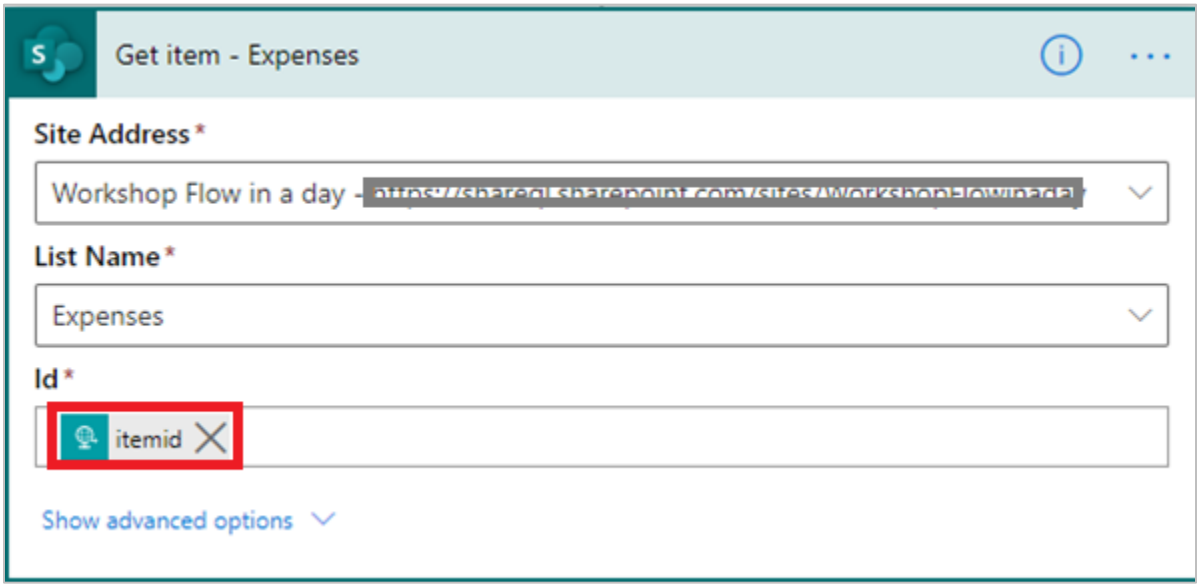
4. Add 3 **Compose** actions and name them as illustrated below:



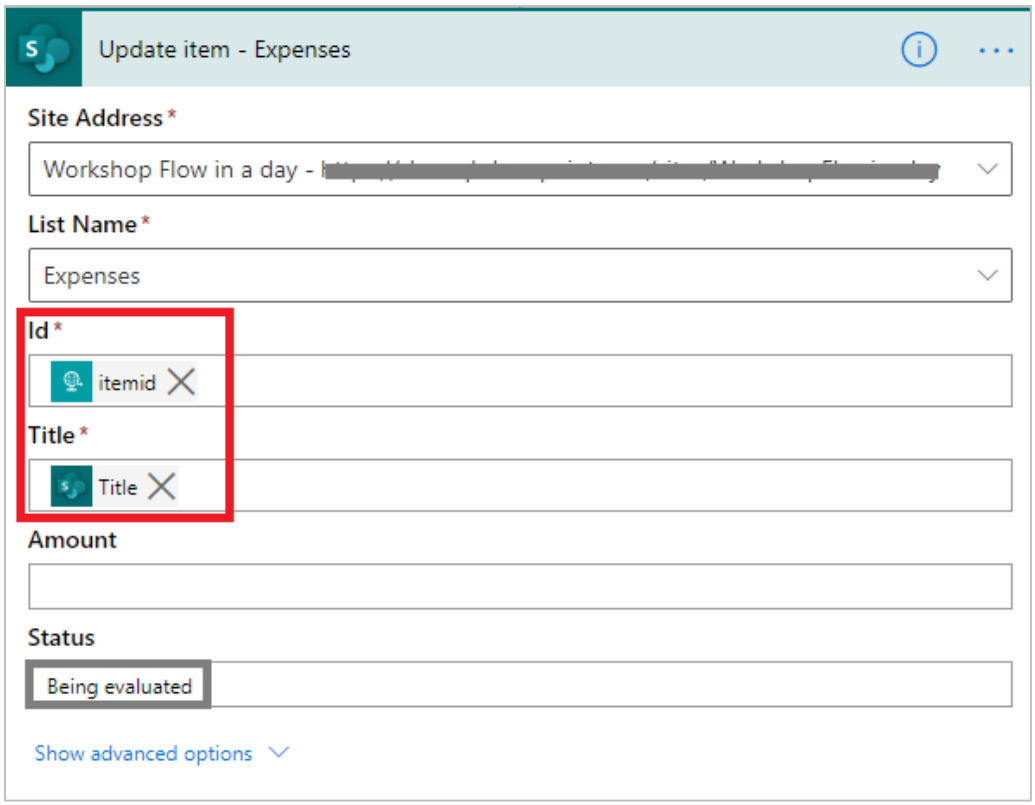
5. Store the e-mails addresses of the **linemanager** and **bigboss** in the 2 compose actions
6. Add "todo" in the **const approval stage URL**.
7. Define a variable of type **Arra**y:



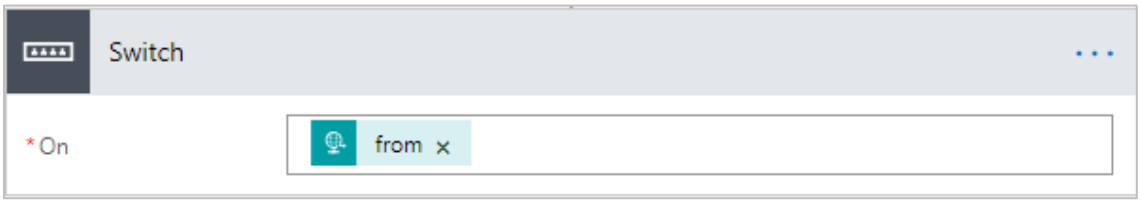
8. Add a SharePoint **Get item** action:



9. Add an **update item** action to change the status of the current expense and rename it **Update expense status**:

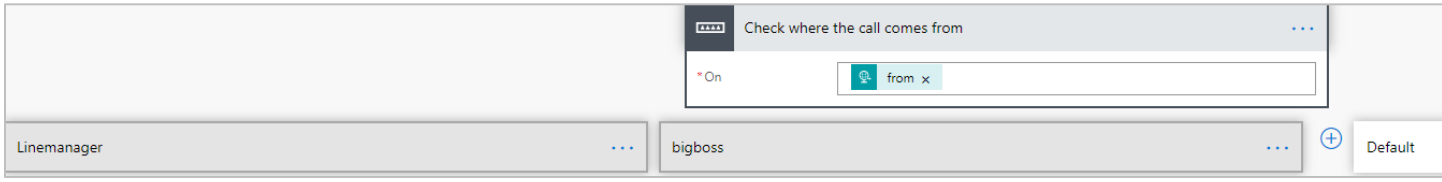


10. After the Update expense status, add a **Switch** action and check the **from** value:



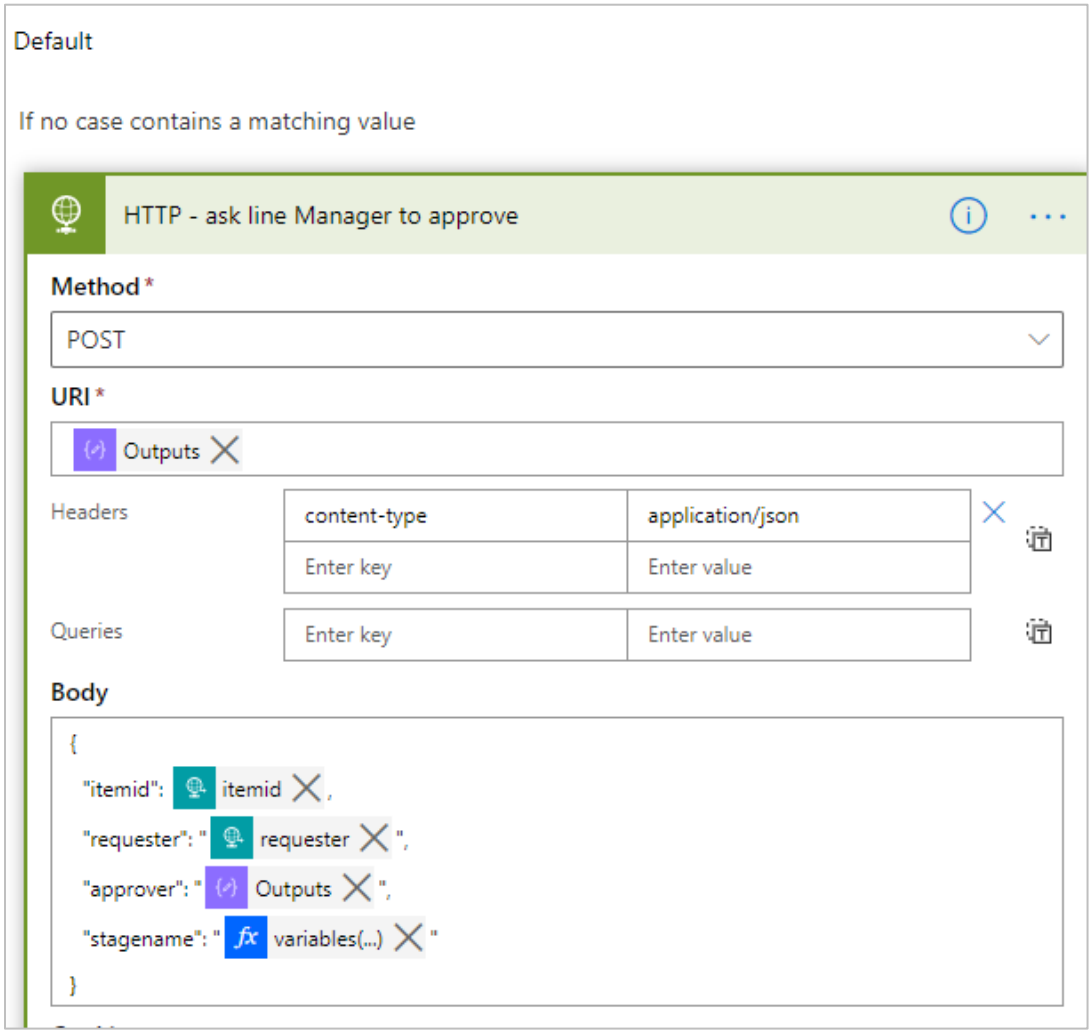
11. Rename the switch **Check where the call comes from**.
12. In the switch, we will analyze 3 scenarios:

    The **from** comes from the **linemanager** approval, from the **bigboss** approval or it is empty:



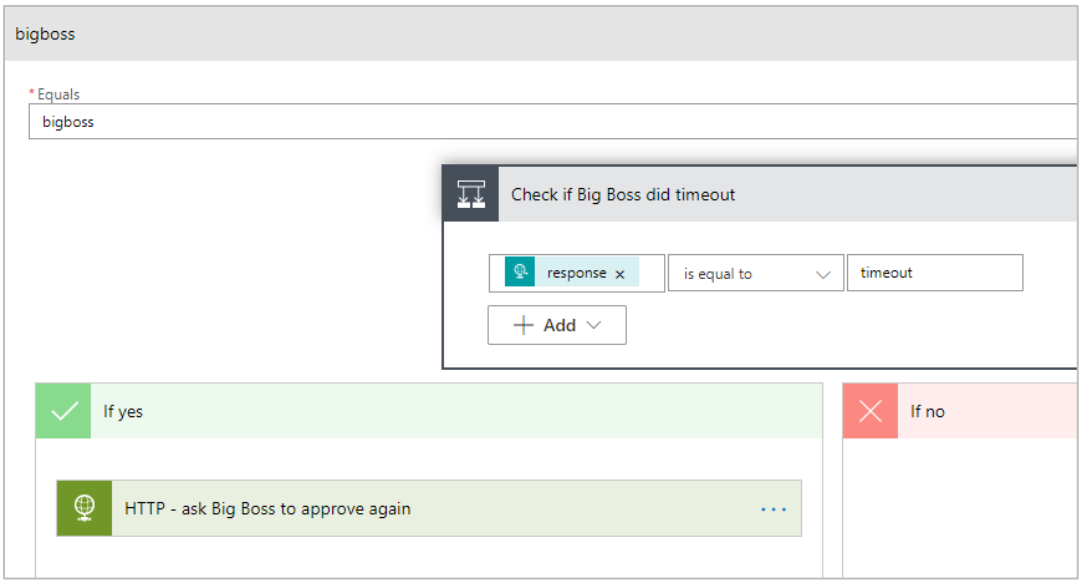
If **from** is empty, we will go into the **Default branch** where we will start the first approval (in this case, line manager approval):

13. Add an **HTTP** request in the default branch:

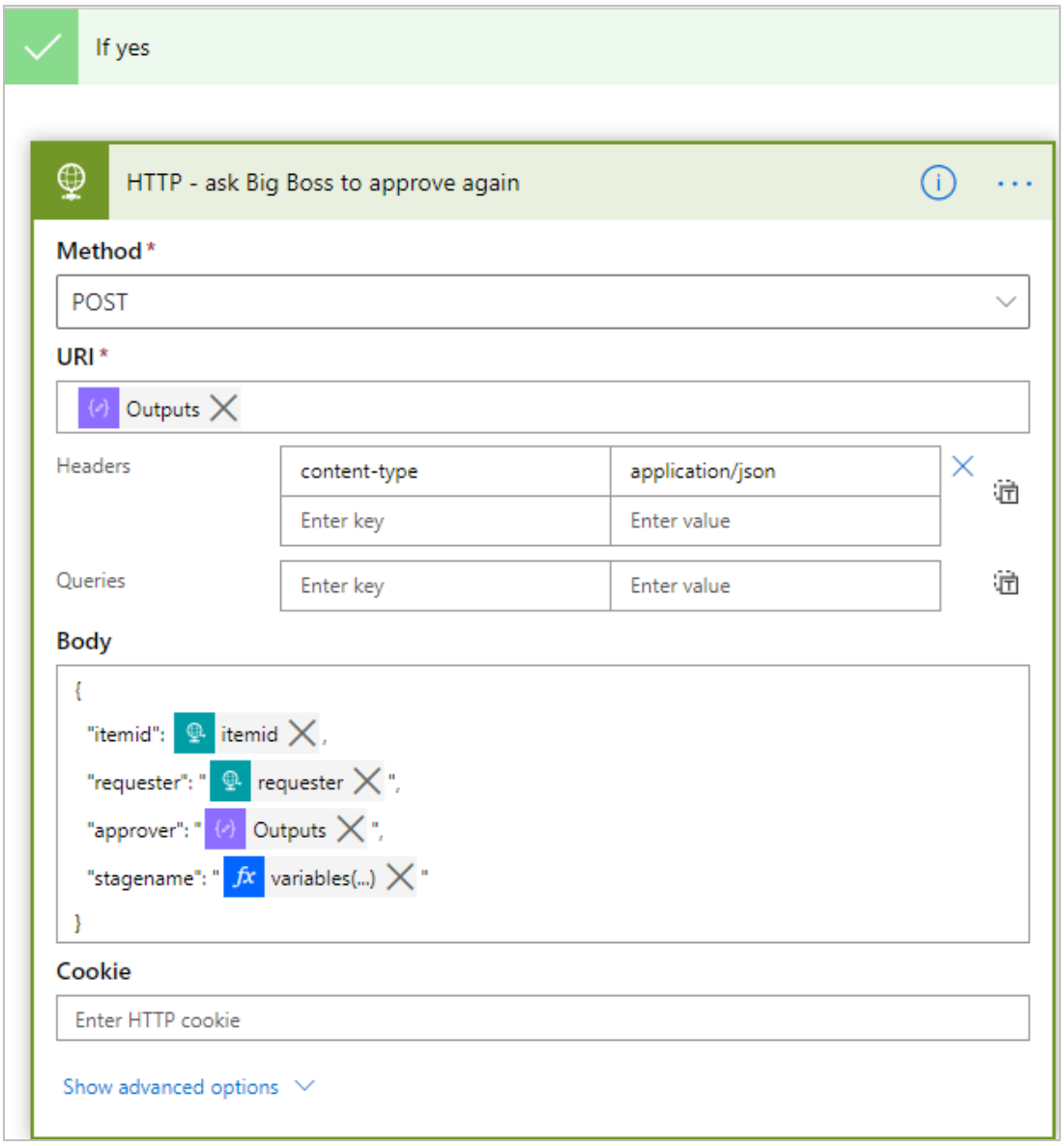


- In the URI: define the output value of the Compose **const approval stage URL.**
- In the **approver**, define the output of the Compose **const linemanager email**.
- In the stagename, type the expression `variables('stages')[0]`

14. Here is what we must do if the value of from is "**bigboss**": (**Big Boss Branch**)
    We must check if the response is timeout; if that is the case, then we need to call Big Boss again (new HTTP action):

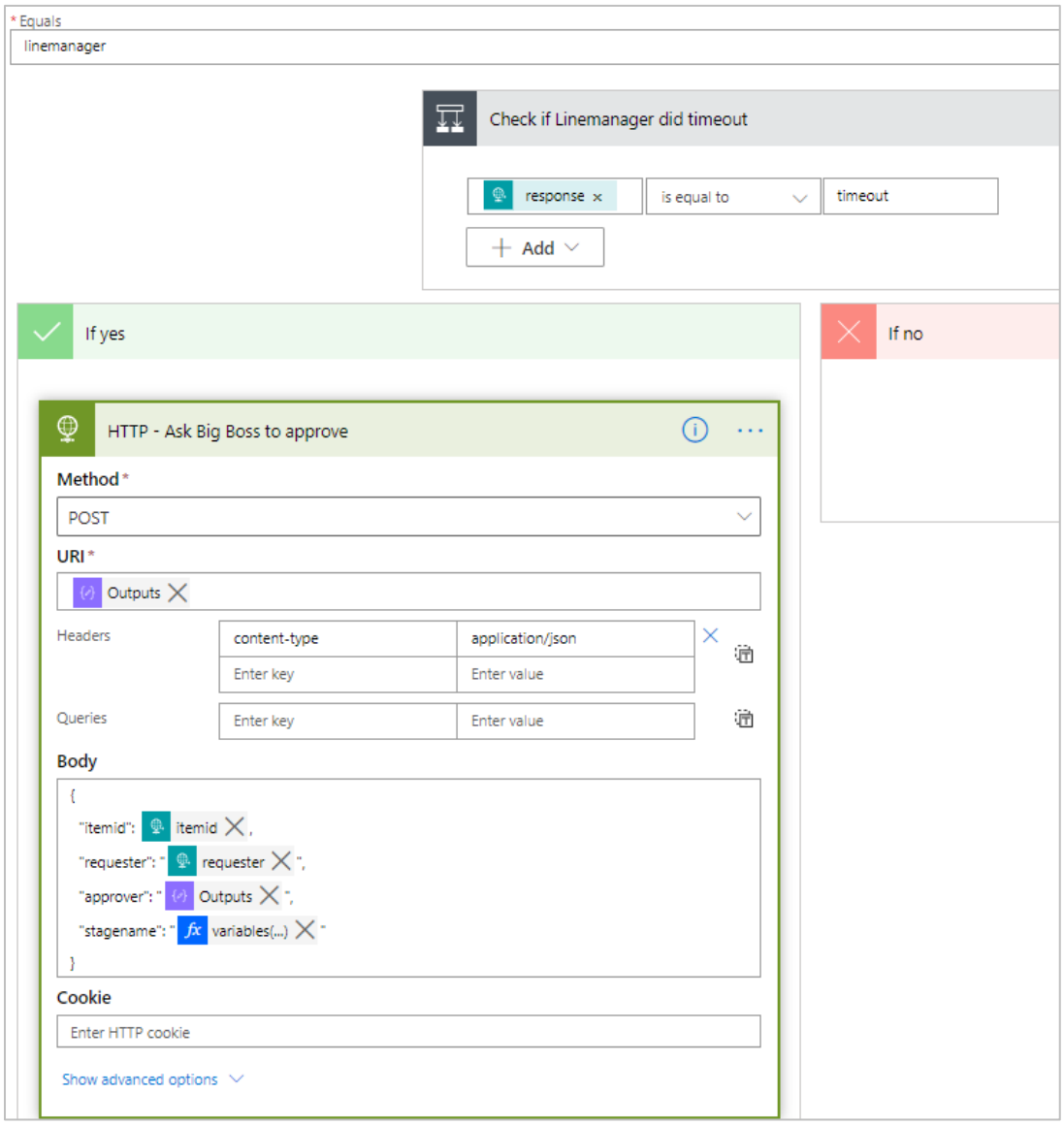


15. The implementation of the HTTP action **ask Big Boss to approve again** is the following:

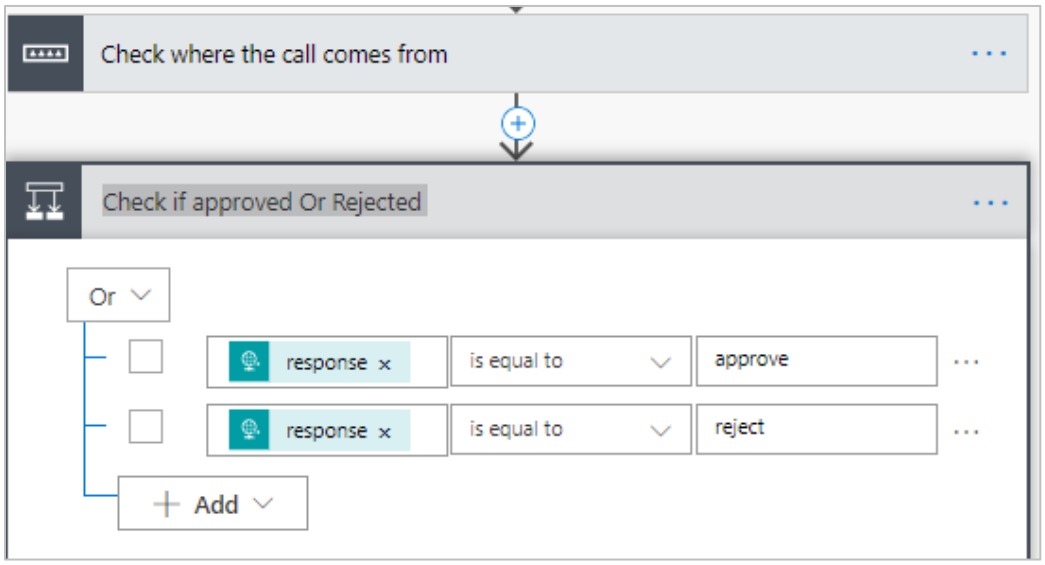


Where the "**stagename**" value is the expression: `variables('stages')[1]`

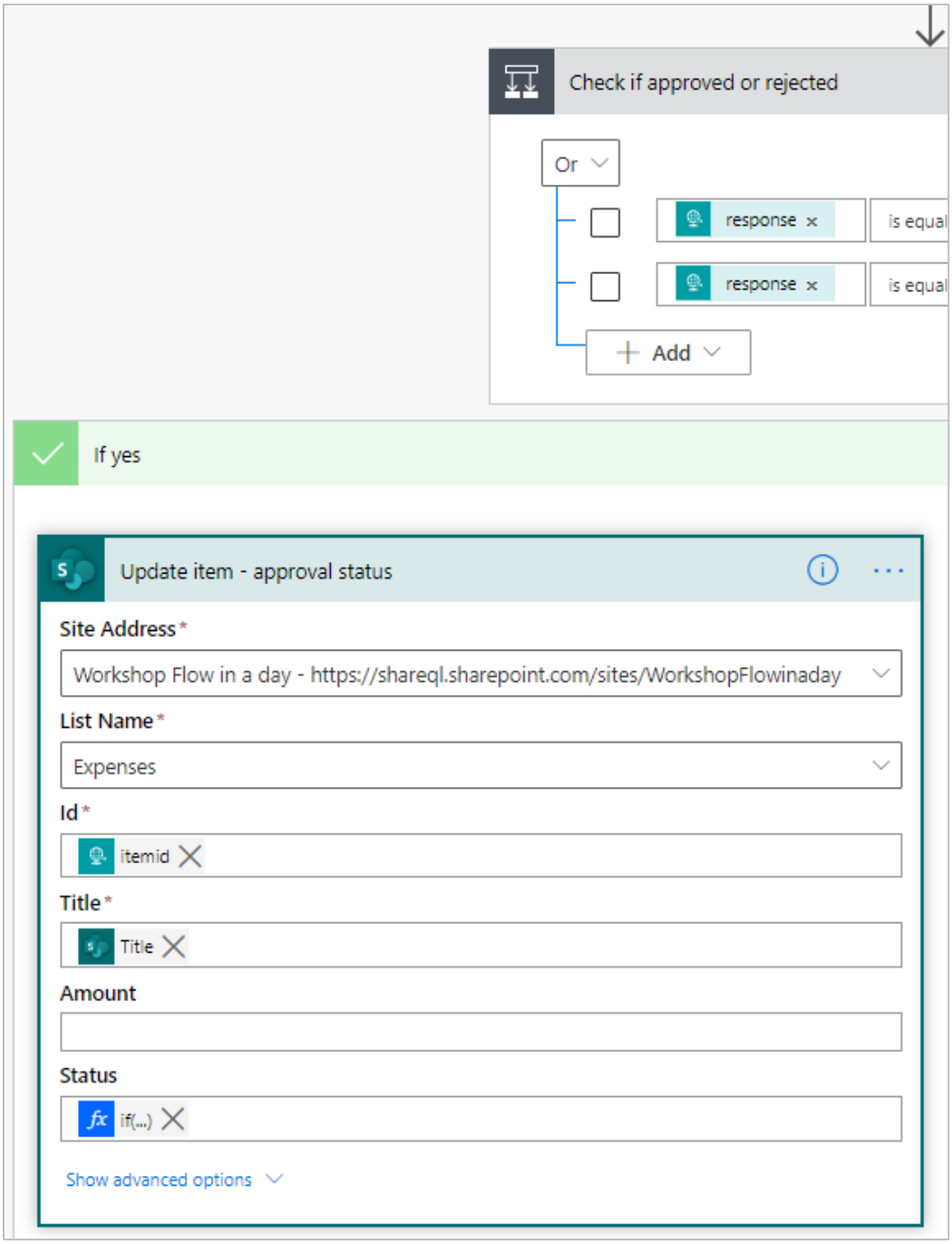
16. Let's implement the **LineManager branch**. If the message is timed-out, we must call the big boss again. It is pretty much what we have implemented in the big boss branch:



17. After the switch, we test if the response was “approve” or “reject”: add a condition and name it **Check if approved Or Rejected**



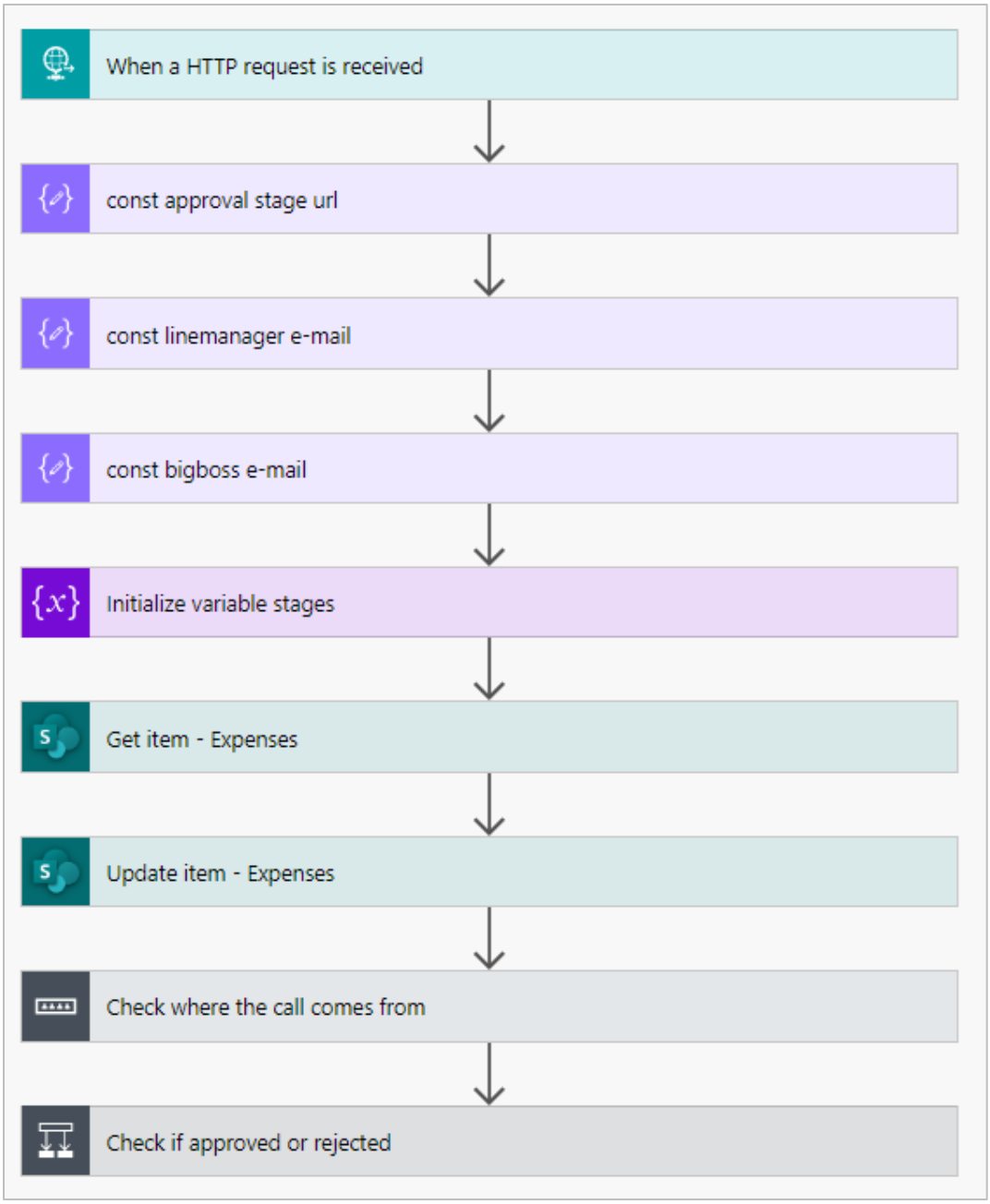
18. In the left branch of the condition update the SharePoint item status:



The expression that updates the Status Value is the following:

```
if(equals(triggerBody()?['response'], 'approved'),'Approved','Rejected')
```

Your Flow should look like this:



19. Save the Flow and open it again the generate the associated public URL in the request trigger (and copy the URL):

20. Open the Generic Stage Flow and paste this URL in the **Compose const Controller URL**.
21. Copy the Generic Stage public URL and paste it in the controller Flow (in the Compose const approval stage URL).
22. Test you Flow by adding a new expense in the SharePoint list.

We need your feedback

Do you want to report an issue or to suggest something? We need your feedback: https://github.com/Power-Automate-in-a-day/Training-by-the-community/issues