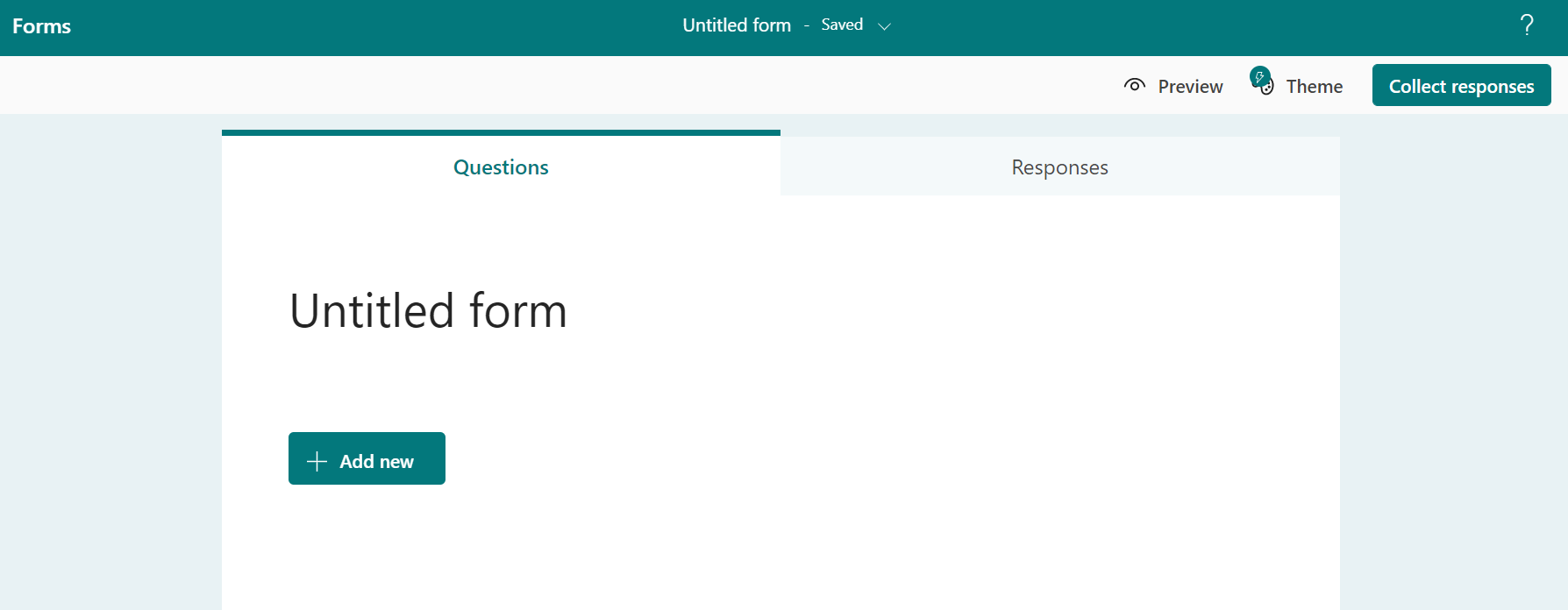
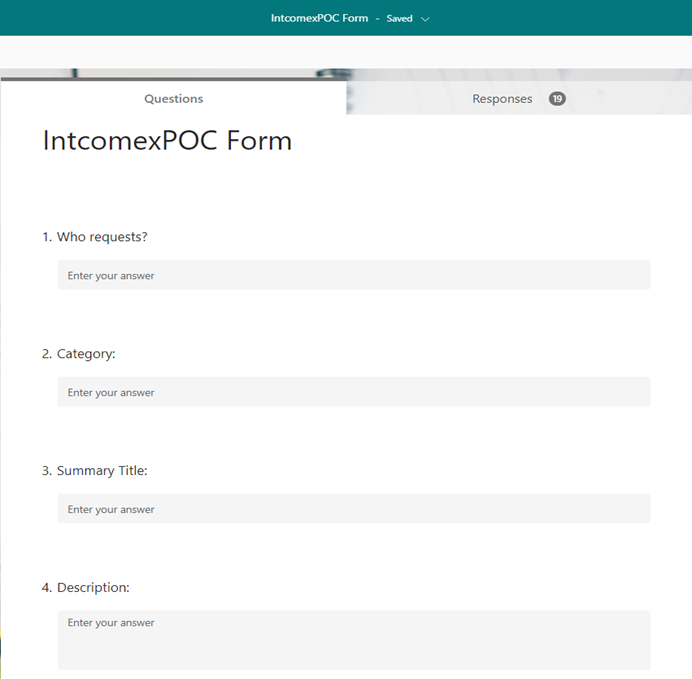
**JIRA-AZUREDEVOPS INTEGRATION**

**Microsoft forms:**

* Take a new Microsoft form
* Add required fields as per the requirement



* Fill the form
* Submit the response

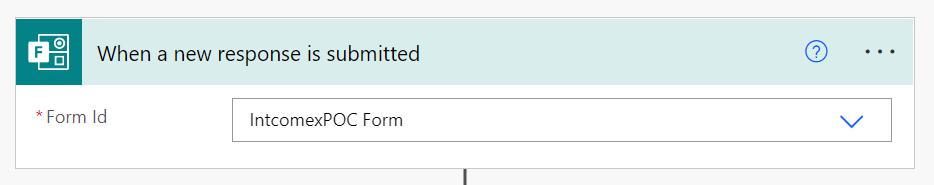


**Power Automate flows: Jira-AzureDevops**

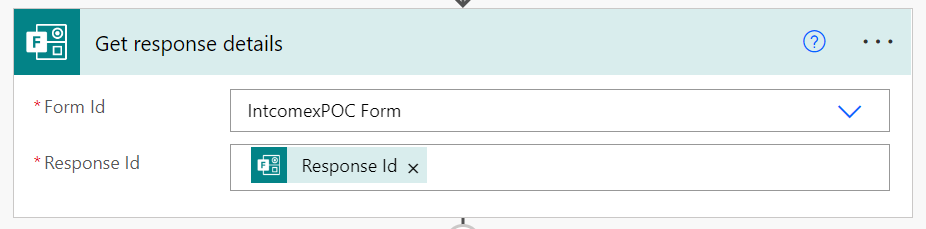
1. **Create an issue in Jira after the form is submitted and after the approval of Jira Issue, flow creates an epic in Azure Devops:**

* When a new response is submitted, the flow will create an issue in Jira.
* It will send an email for approval.
* It delays the flow for the scheduled time and after the approval, the flow will create an epic in AzureDevops.

**The following steps explain the flow:**

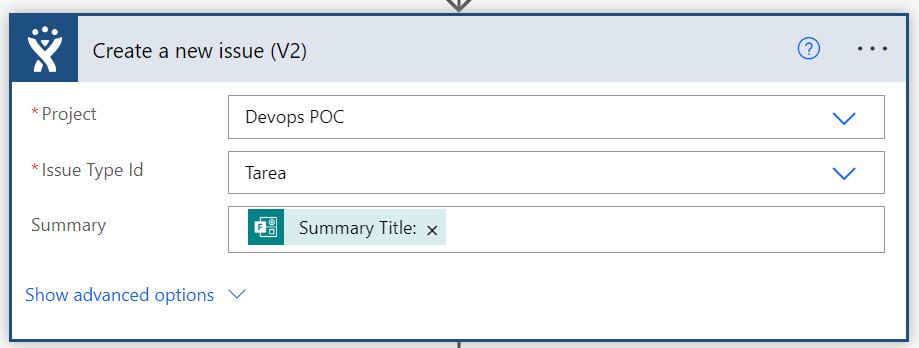


This operation triggers a flow when a new response is submitted.

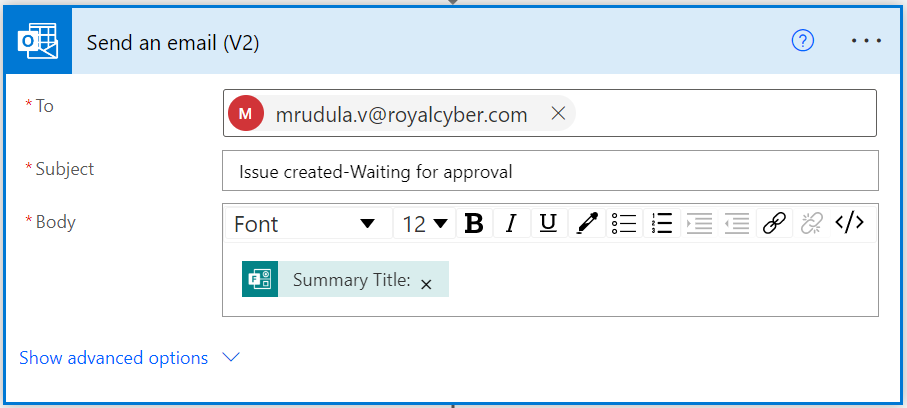


This action retrieves a form response, the Response Id takes the response when a form is

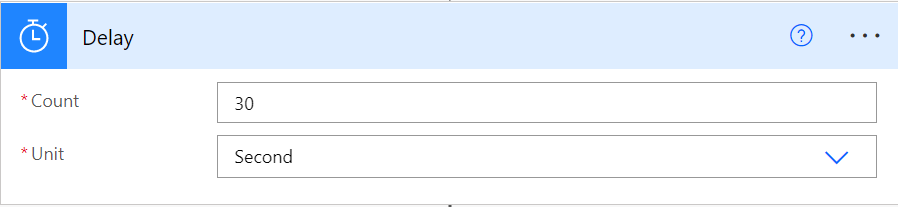
submitted.



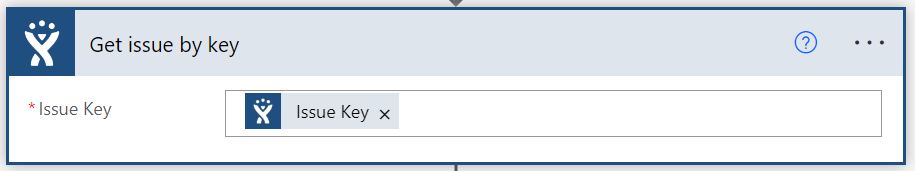
This operation is used to create a new issue. We can select the project and Issue type from Jira, since it is connected by establishing the connection between Power Automate and Jira. Summary takes the dynamic response from the form submitted.



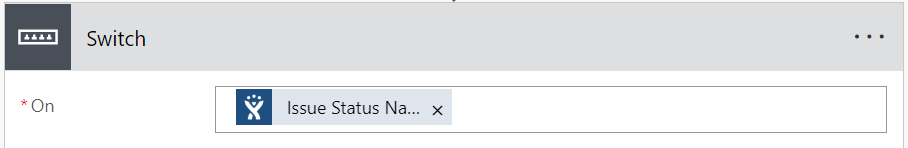
This operation sends an email to the approver, notifying that issue has been created and waiting for approval. Body of the email contains the dynamic content from the form response.



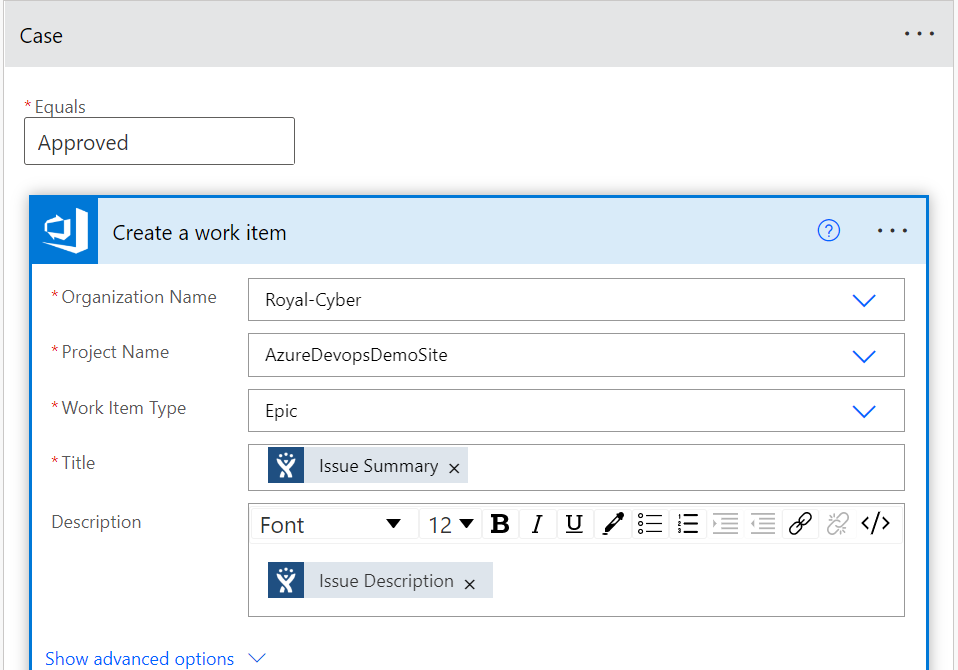
This operation delays the flows to review the Issues in Jira. We can define the count and unit as per the requirement.



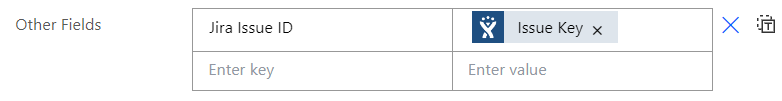
This operation is used to retrieve the issue object for a given issue Key.



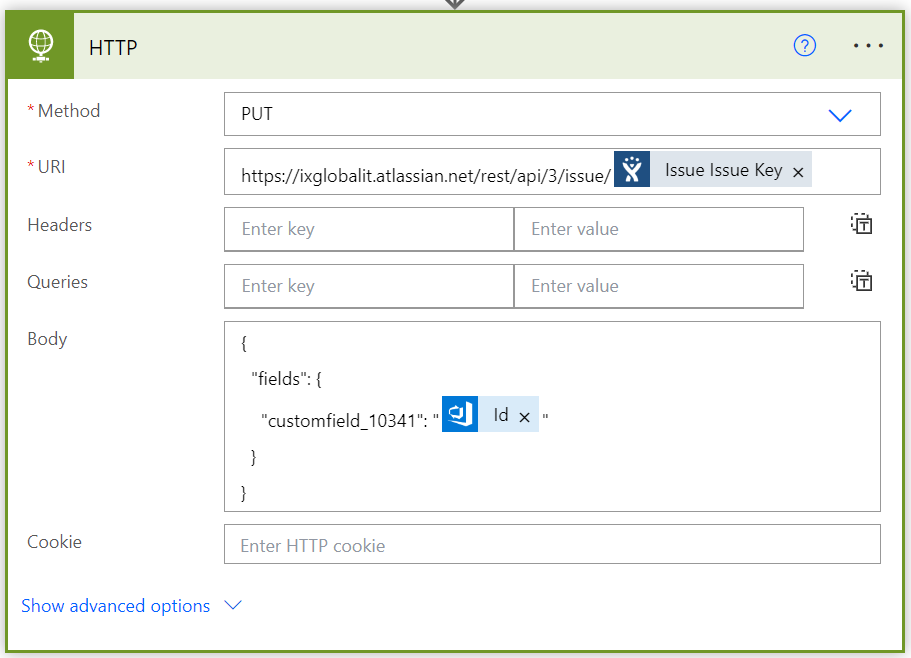
The switch case take the status of the Jira issue and we can create actions for different cases.



When the ‘Issue status’ response is ‘Approved’, this operation creates a new work item with the provided attributes in this case. The dynamic values received from the Jira issue will be updated to fields of the work item. Here we have chosen work item to be Epic and the title of the Epic will be the Jira Issue summary and the Description will take the response same as the Issue description.



While creating work item, we have given that Jira issue ID field in AzureDevops will take the value of Jira Issue key.

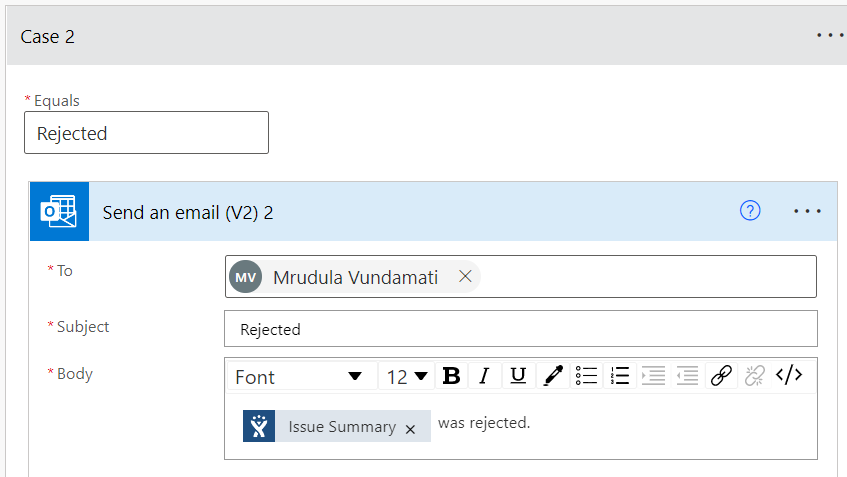


This operation updates the Jira Issue using HTTP action. Here the PUT method applies the changes to the Jira issue, which was linked to work item, by the provided URL of the Jira issues. The body contains the JSON text, which tells the HTTP action to update the Jira field (AzureDevops Epic ID) by the work item ID so that the work item and Jira are linked to each other.

HTTP actions enable you to interact with APIs and send web requests that perform various operations.

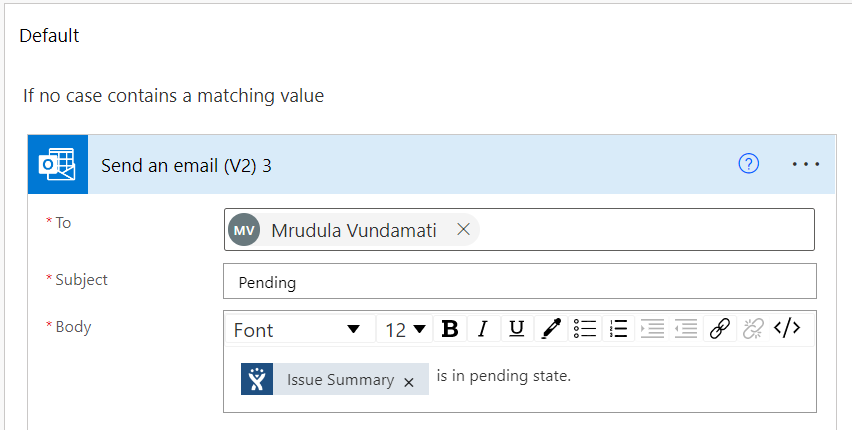
To send an API request, like POST, GET, PUT, or DELETE, use the Invoke web service action.

In the action's properties, you must populate the service's URL and the appropriate HTTP method.



In this when the Jira Issue status name is ‘Rejected’, this action will send a message by email to the responder, that the form was rejected.

In other case of the issue status, the form will be in pending state and it sends email to responder saying it is still in pending state.

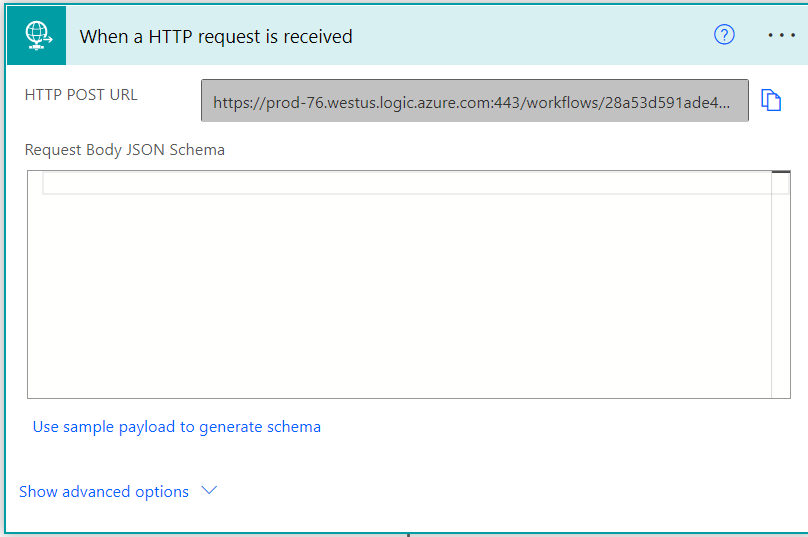


1. **Recurrence flow (for the issues which are yet to approve):**

* The flow will run for scheduled intervals and checks whether any issue is updated.
* If the issue is approved and it will create an epic in Azure Devops
* Updates the epic id in Jira issue.

**The following steps explain the flow:**

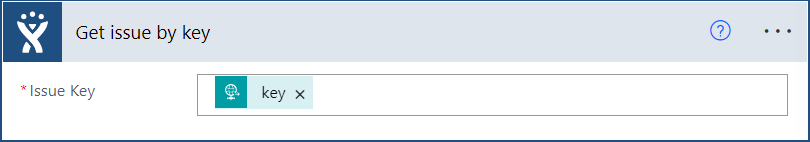
**When an HTTP request is received:**



When first adding the ‘When an HTTP request is received’ trigger, to a flow it starts with an HTTP POST URL informing us that the URL will be generated after the Flow has been saved. This means that initially creating the Flow will not allow us to be able to provide/use the URL that is required to trigger the Flow.

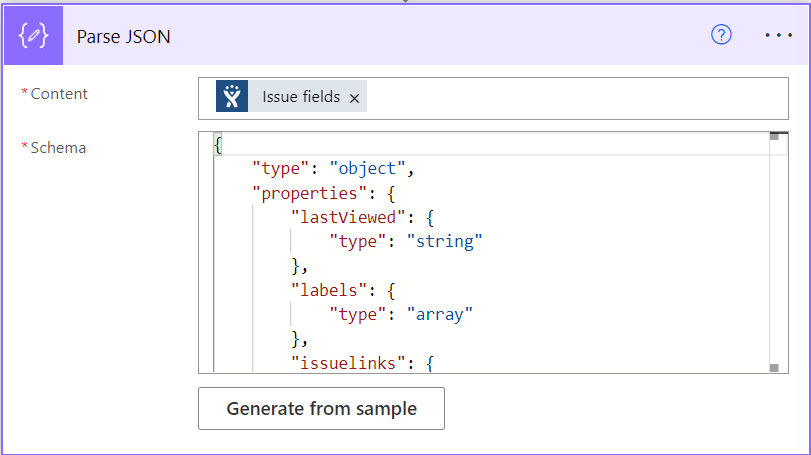
After entering JSON Schema into the box and hit done, the schema will be created and displayed in the ‘Request Body JSON Schema’ section as shown above. The above step retrieves the data from Jira.

**Get Issue by Key:**



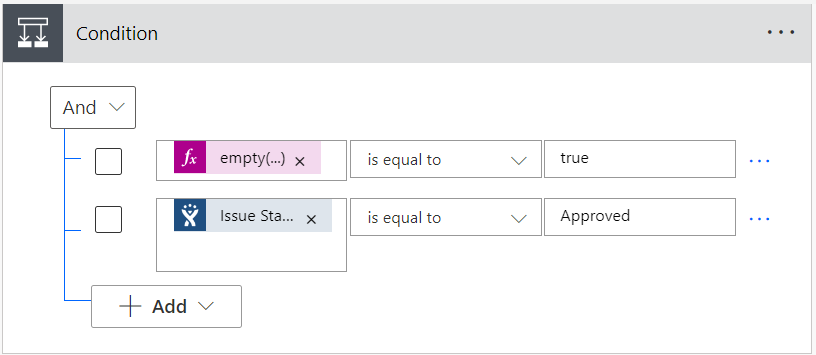
This is a Jira Connector, which gives the data about the Jira Issue Key from the key which is passed from the HTTP request connector.

Parse JSON:



In this step, Fields of Jira Issue will be parsed and gives the data of each field that is associated with a Jira Issue Key.

**Condition:**



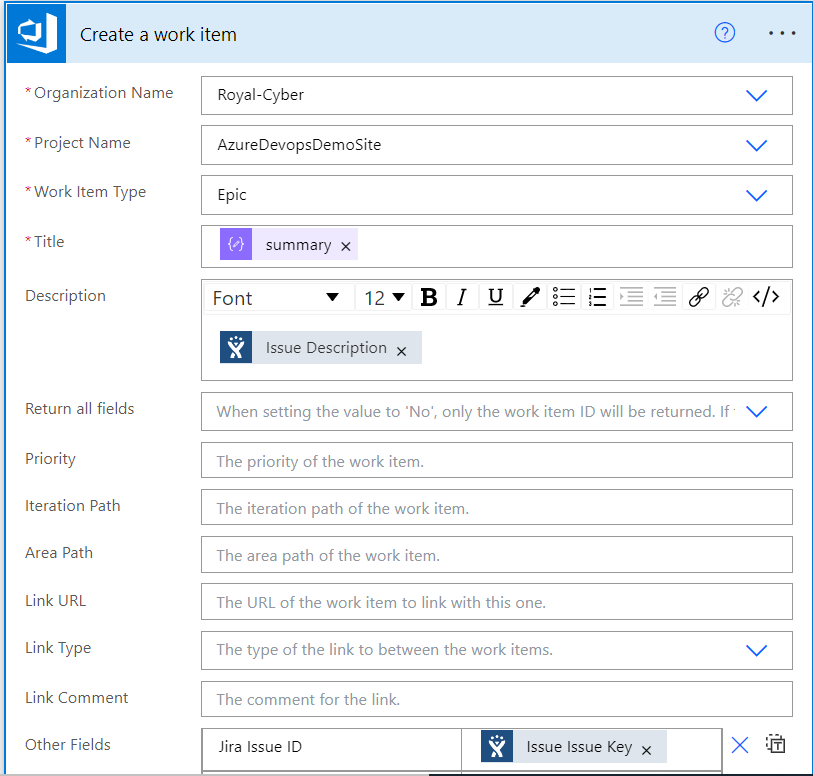
Then, there are two conditions these 2 must be validated.

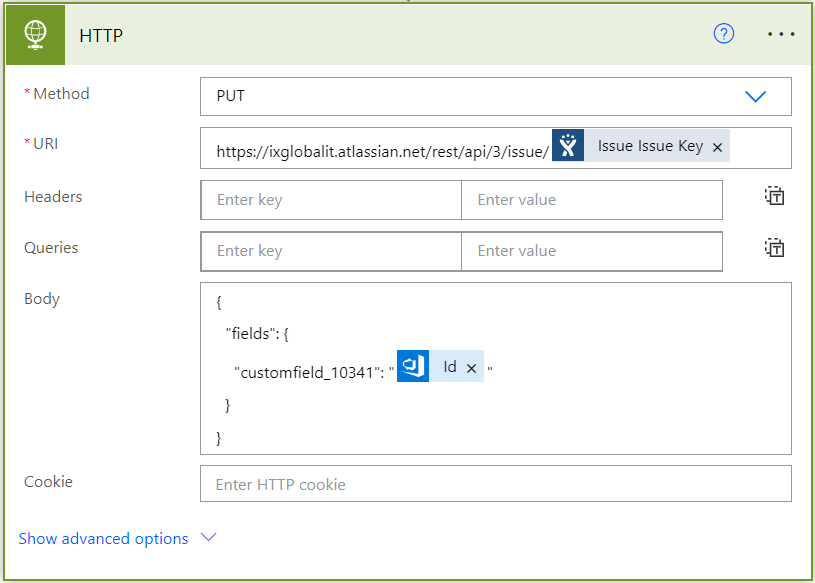
1. Empty (): This operation is to check whether any value is assigned or not, this should be true.
2. Jira Issue Status Name: This field is to check whether the status is in the approved state or not.

If the condition Yes then, there are 2 steps as follows:

**Create a work item:**

Create a new work item with the provided attributes.

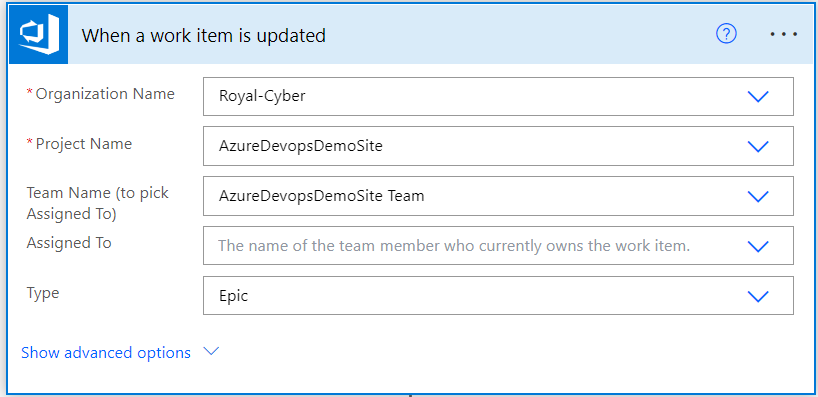


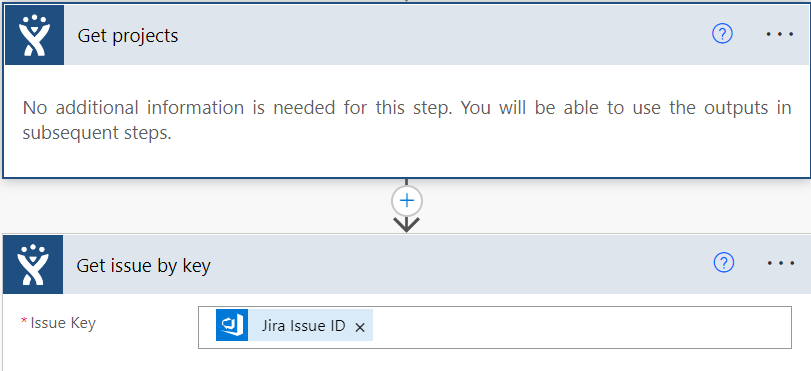


This operation updates the Jira Issue using HTTP action. Here the PUT method applies the changes to the Jira issue, which was linked to the work item, by the provided URL of the Jira issues.

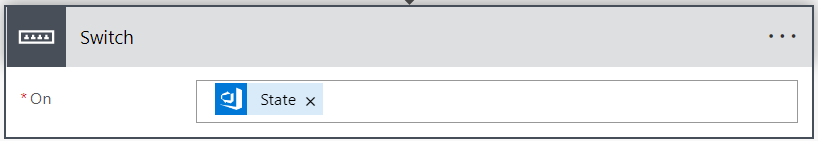
1. **When epic is updated in AzureDevops, flow updates the Jira issue:**

* When an epic is updated (the state changes), the flow will edit the status of Jira via HTTP request.

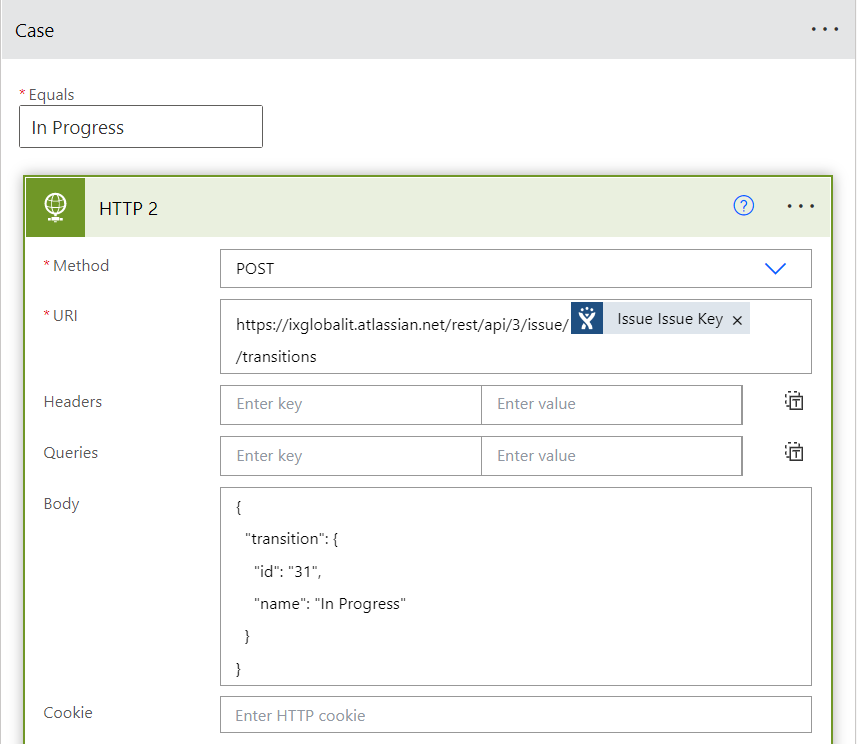
  
Triggers a flow when a work item which matches provided criteria is updated.

  
“Get projects" operation is used to retrieve a list of projects for your JIRA instance.

The “Get issue by key” operation is used to retrieve the issue object for a given issue Key. Here Issue key field is filled by the AzureDevops field “Jira Issue Id” since the Jira issue ID is the same as the Jira Issue key, it retrieves the same information.

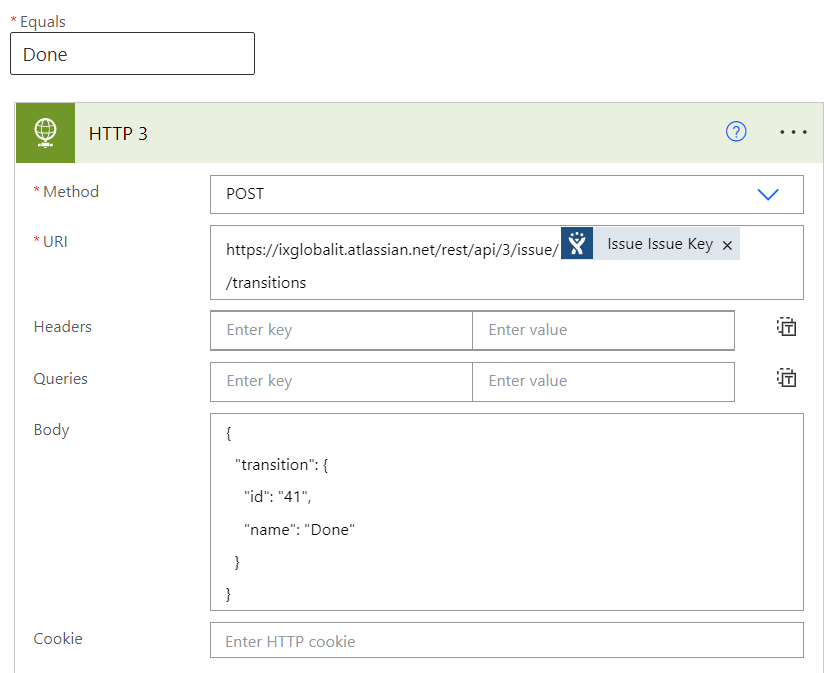


Here the “Switch” operation takes the value of “State” field of AzureDevops Work item.

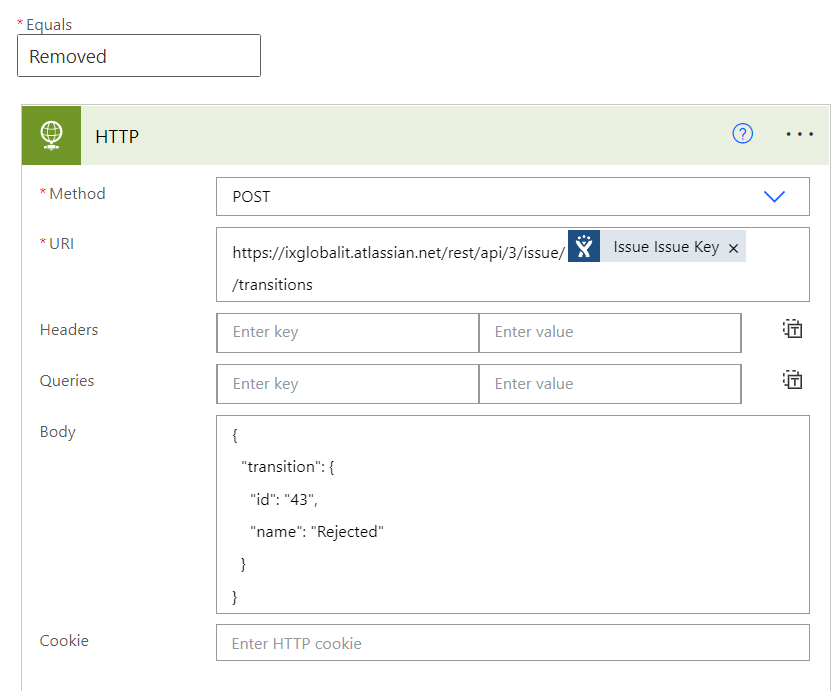


In this case, when the “State” of the work item is “In Progress”, the “HTTP” operation updates the status of the Jira issue by the POST method. The URL here used is for transitioning the Jira issue. The body is in Json format, the id indicates what would be the status name of the Jira issue.

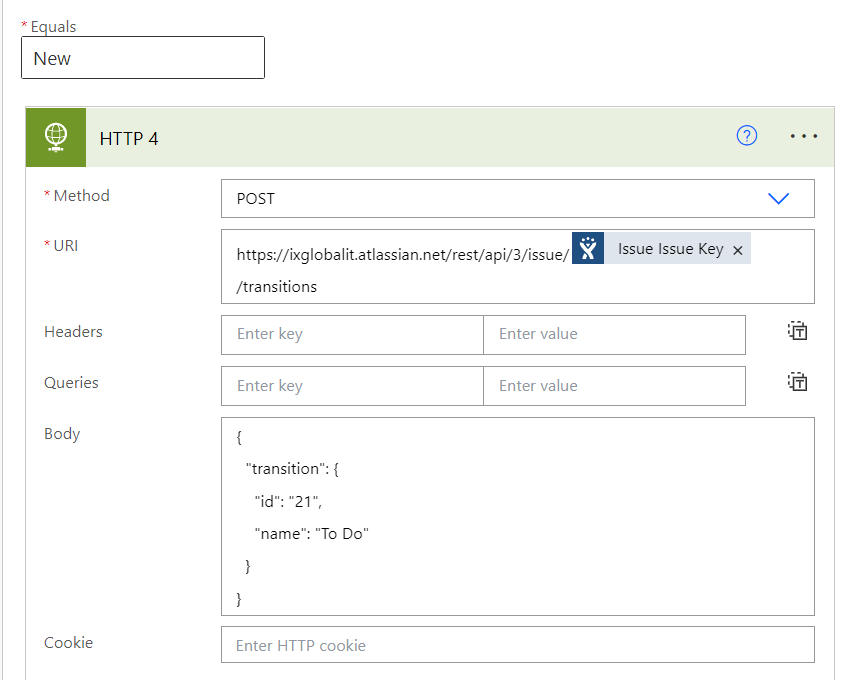
In Case of “Done” state:



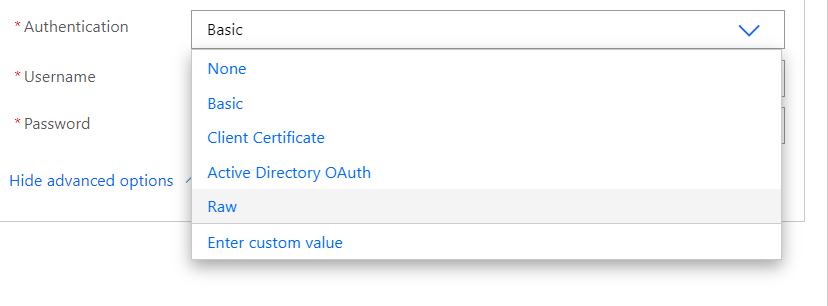
In Case of “Removed” state:



In Case of “New” State:



There are different types of authentications. Here we have used basic authentication, which requires email id and Personal access token.



**Connectors required for Jira-AzureDevops Integration:**

