

Building Flows to Manage Content and Approvals



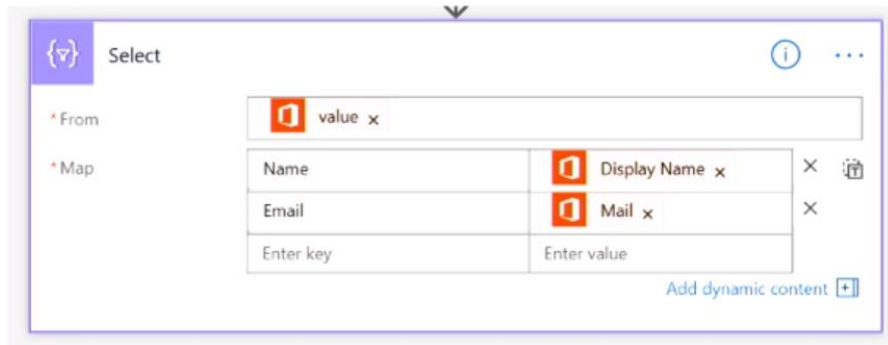
Agenda

- Processing Documents and Images
- Using Flow to Generate Word and PDF Files
- Integrating Flow with Microsoft Forms
- Using Flow to Automate Approval Processes
- Managing Approvals using Approvals Center



Transforming Arrays

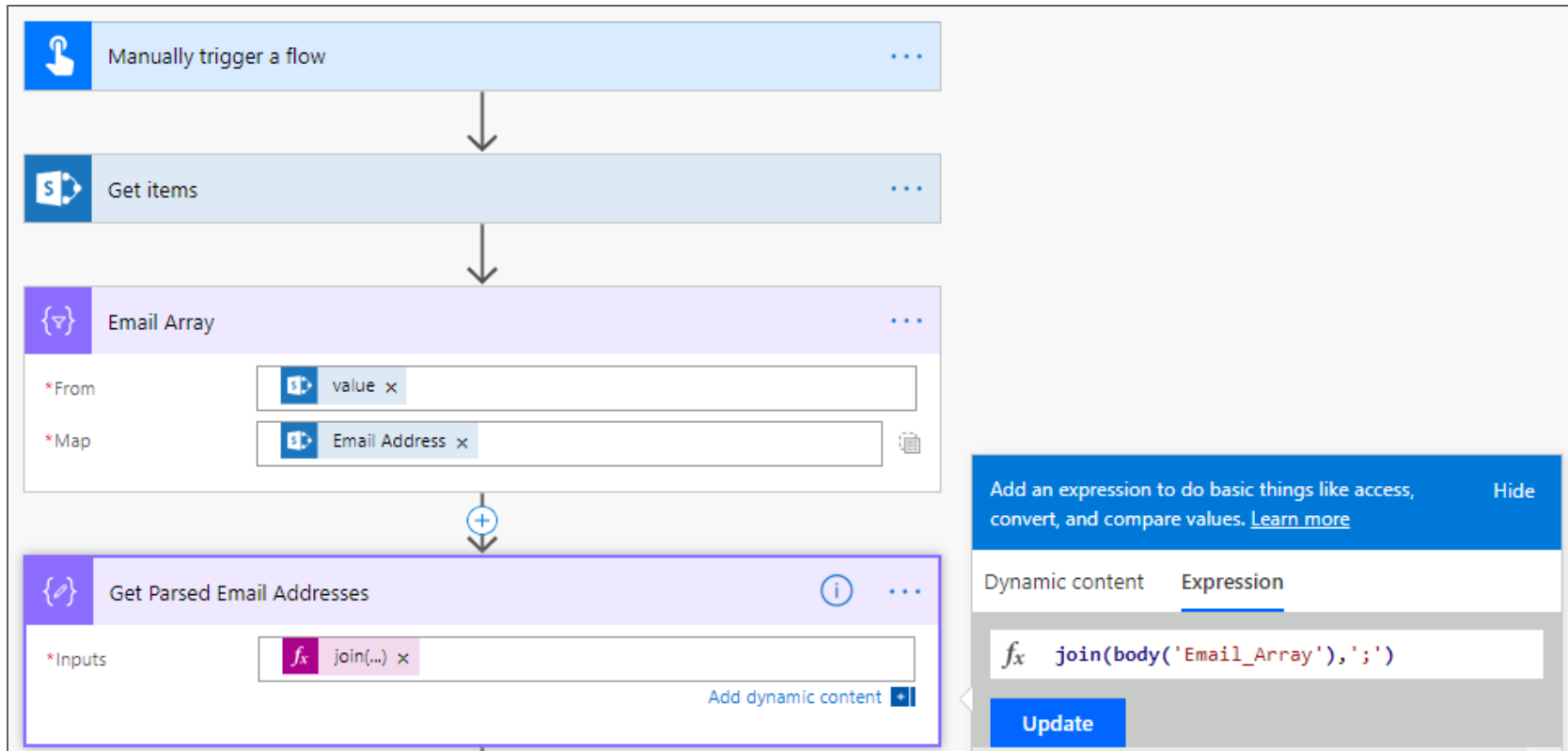
- Use Select action
 - Two input modes: fill key-value pairs or typing directly
- Create array object objects
 - Useful for passing array to another action



- Create a simple array of strings, numbers, Booleans, etc
 - Useful for creating simple list (e.g. email addresses)

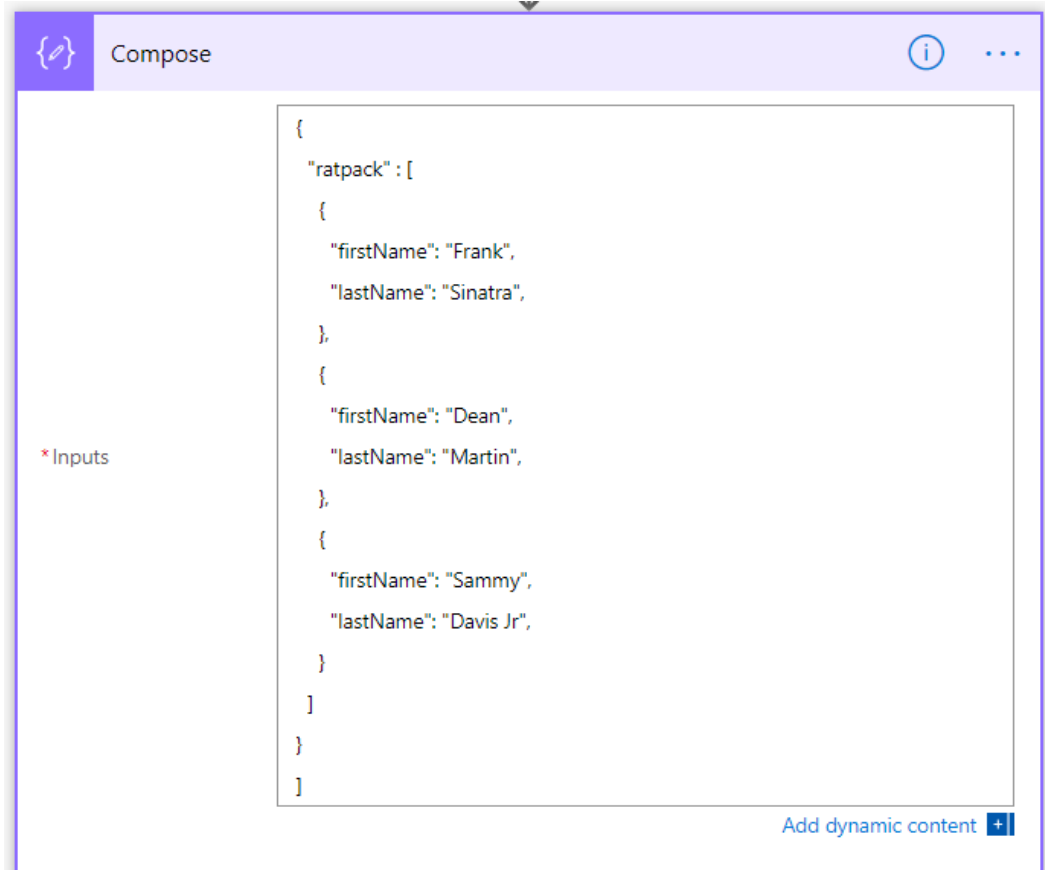
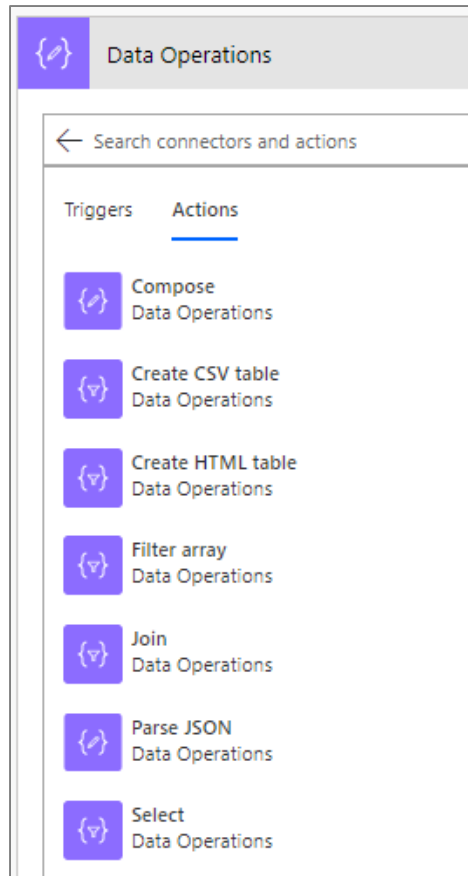


Converting an Array using Select



Data Operations

- Used to process data and prepare content



Handling Type Conversion

- Some conversion is automatic
 - Sometimes conversions are performed for you
 - In other cases, you must explicitly convert between types



string(value)

Convert the parameter to a string



float(value)

Convert the parameter argument to a floating-point number



bool(value)

Convert the parameter to a Boolean



base64(value)

Returns the base 64 representation of the input string



base64ToBinary(value)

Returns a binary representation of a base 64 encoded string



base64ToString(value)

Returns a string representation of a base 64 encoded string



binary(value)

Returns a binary representation of a value



dataUriToBinary(value)

Returns a binary representation of a data URI



dataUriToString(value)

Returns a string representation of a data URI



dataUri(value)

Returns a data URI of a value



uriComponent(value)

Returns a URI encoded representation of a value



uriComponentToBinary(value)

Returns a binary representation of a URI encoded string



uriComponentToString(value)

Returns a string representation of a URI encoded string



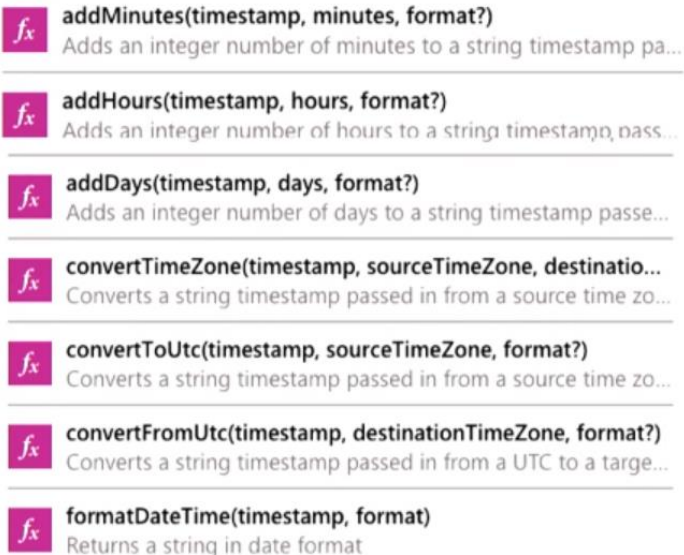
Flow Type Conversion Matrix

To From	UI adds automatically					Floating-point	Integer	Bool.	Array	JSON Object	XML content
	String	Base 64	Binary content	Data URI	URI comp.						
String	Yes	base64()	binary()	dataUri()	uriComponent()	float()	int()	bool()	split() json()	json()	xml()
Base 64	base64ToString()	Yes	base64ToBinary()	*	*	*	*	*	*	*	*
Binary content	string()	base64()	Yes	dataUri()	uriComponent()	*	*	*	*	*	*
Data URI	dataUriToString()	*	dataUriToBinary()	Yes	*	*	*	*	*	*	*
URI comp.	uriComponentToString()	*	uriComponentToBinary()	*	Yes	*	*	*	*	*	*
Floating-point	Yes	base64()	binary()	dataUri()	uriComponent()	Yes	No	No	No	No	No
Integer	Yes	base64()	binary()	dataUri()	uriComponent()	Yes	Yes	No	No	No	No
Bool.	Yes	base64()	binary()	dataUri()	uriComponent()	No	No	Yes	No	No	No
Array	join() string()	*	*	*	*	No	No	No	Select Action	Select or Compose	xml()
JSON object	string()	*	*	*	*	No	No	No	Select or Compose	Compose Action	xml()
XML content	string()	*	*	*	*	No	No	No	xpath()	xpath()	Logic apps only



Working with Dates and Time

- Get Greenwich Meantime using **utcnow()**
- Use **add*()** functions to move time back/forward
- **convertTimeZone()** used to handle local times
- **formatDateTime()** used to format



A screenshot of a code editor window displaying a list of date and time functions. Each function is preceded by a small icon of a pink square with a white 'fx' symbol. The functions listed are: **addMinutes(timestamp, minutes, format?)** (Adds an integer number of minutes to a string timestamp passed in), **addHours(timestamp, hours, format?)** (Adds an integer number of hours to a string timestamp passed in), **addDays(timestamp, days, format?)** (Adds an integer number of days to a string timestamp passed in), **convertTimeZone(timestamp, sourceTimeZone, destinationTimeZone, format?)** (Converts a string timestamp passed in from a source time zone to a destination time zone), **convertToUtc(timestamp, sourceTimeZone, format?)** (Converts a string timestamp passed in from a source time zone to UTC), **convertFromUtc(timestamp, destinationTimeZone, format?)** (Converts a string timestamp passed in from a UTC to a target time zone), and **formatDateTime(timestamp, format)** (Returns a string in date format).

fx **addMinutes(timestamp, minutes, format?)**
Adds an integer number of minutes to a string timestamp passed in

fx **addHours(timestamp, hours, format?)**
Adds an integer number of hours to a string timestamp passed in

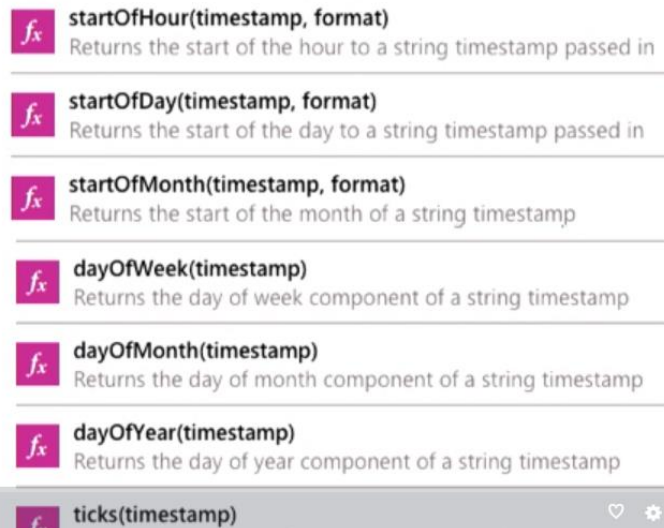
fx **addDays(timestamp, days, format?)**
Adds an integer number of days to a string timestamp passed in

fx **convertTimeZone(timestamp, sourceTimeZone, destinationTimeZone, format?)**
Converts a string timestamp passed in from a source time zone to a destination time zone

fx **convertToUtc(timestamp, sourceTimeZone, format?)**
Converts a string timestamp passed in from a source time zone to UTC

fx **convertFromUtc(timestamp, destinationTimeZone, format?)**
Converts a string timestamp passed in from a UTC to a target time zone

fx **formatDateTime(timestamp, format)**
Returns a string in date format



A screenshot of a code editor window displaying a list of date and time functions. Each function is preceded by a small icon of a pink square with a white 'fx' symbol. The functions listed are: **startOfHour(timestamp, format)** (Returns the start of the hour to a string timestamp passed in), **startOfDay(timestamp, format)** (Returns the start of the day to a string timestamp passed in), **startOfMonth(timestamp, format)** (Returns the start of the month of a string timestamp), **dayOfWeek(timestamp)** (Returns the day of week component of a string timestamp), **dayOfMonth(timestamp)** (Returns the day of month component of a string timestamp), **dayOfYear(timestamp)** (Returns the day of year component of a string timestamp), and **ticks(timestamp)**. At the bottom right of the editor, there are icons for a heart, a gear, a speaker, and a star.

fx **startOfHour(timestamp, format)**
Returns the start of the hour to a string timestamp passed in

fx **startOfDay(timestamp, format)**
Returns the start of the day to a string timestamp passed in

fx **startOfMonth(timestamp, format)**
Returns the start of the month of a string timestamp

fx **dayOfWeek(timestamp)**
Returns the day of week component of a string timestamp

fx **dayOfMonth(timestamp)**
Returns the day of month component of a string timestamp

fx **dayOfYear(timestamp)**
Returns the day of year component of a string timestamp

fx **ticks(timestamp)**



dataUriToBinary()

- PowerApps photos require conversion
 - Allows you to upload photos to SharePoint
 - Accomplished using **dataUriToBinary()** function

```
dataUriToBinary(triggerBody()['Createfile_FileContent'])
```

The screenshot displays a PowerApps flow with two actions: 'Get File Name' and 'Create file'.

Get File Name Action:

- Inputs:** `concat(...)`

Create file Action:

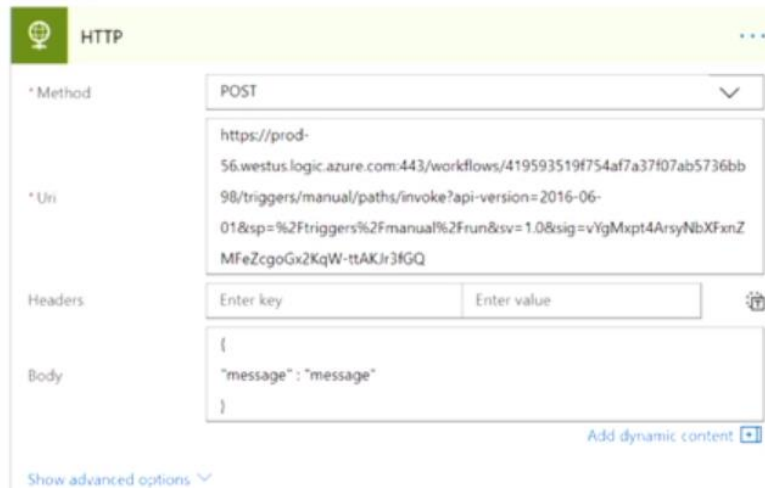
- Site Address:** Critical Path Training Labs Team Site - `https://msd0910.sharepoint.com/`
- Folder Path:** /My Photos
- File Name:** Output
- File Content:** `dataUriToBinary(...)`

Dynamic content / Expression editor:

- Expression:** `dataUriToBinary(triggerBody()['Createfile_`
- Update button:** Update
- String functions:** `concat(text_1, text_2?, ...)` Combines any number of strings together.

Calling Flows using the HTTP action

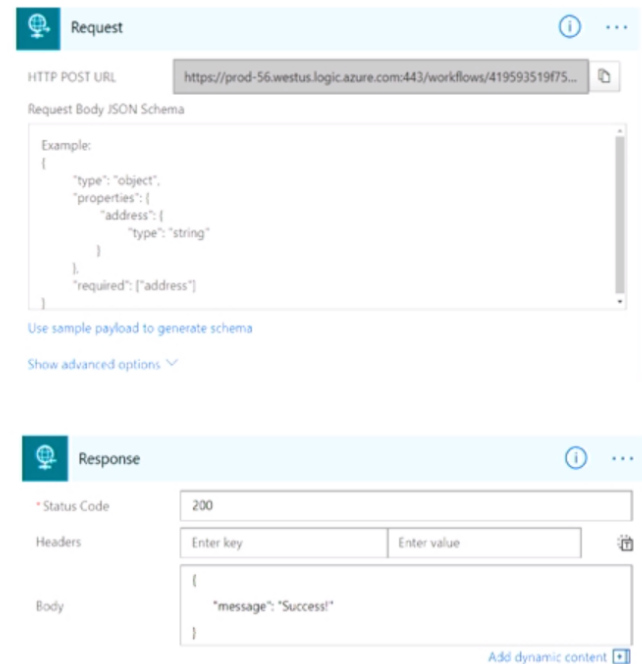
In the parent workflow:



The screenshot shows the configuration for an HTTP action in a parent workflow. The action is named "HTTP" and is set to the "POST" method. The URI is a long URL starting with "https://prod-56.westus.logic.azure.com:443/workflows/419593519f754af7a37f07ab5736bb98/triggers/manual/paths/invoke?api-version=2016-06-01&sp=%2Ftriggers%2Fmanual%2Frun&sv=1.0&sig=vYgMxpt4ArsyNbXfanZMFeZcgoGx2KqW-ttAKJr3fGQ". The headers section is empty, and the body contains a JSON object: {"message": "message"}. There is a link to "Show advanced options" and a button to "Add dynamic content".



In the child workflow:

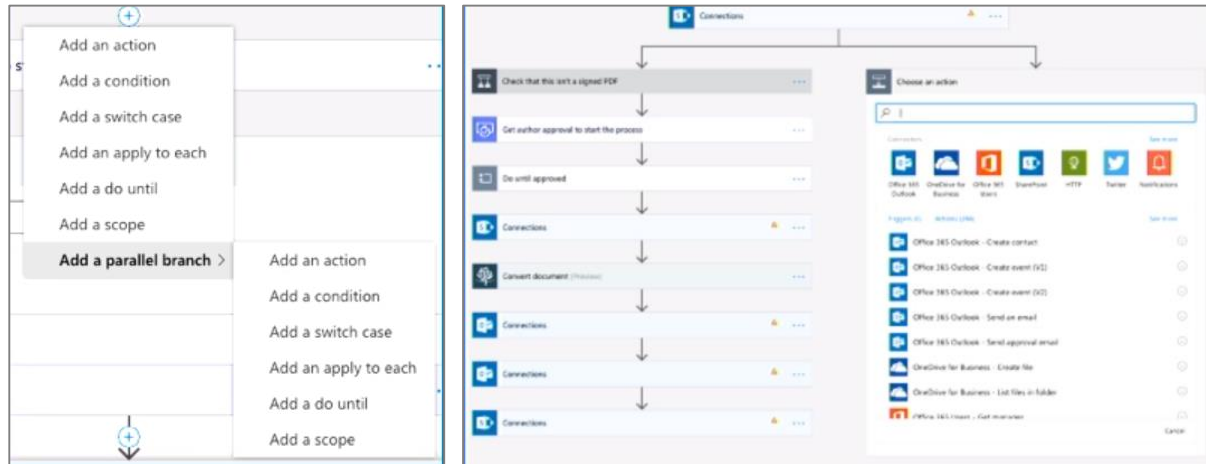


The screenshot shows the configuration for an HTTP action in a child workflow. The action is named "Request" and is set to the "POST" method. The URI is a long URL starting with "https://prod-56.westus.logic.azure.com:443/workflows/419593519f75...". The headers section is empty, and the body contains a JSON object: {"message": "Success!"}. There is a link to "Show advanced options" and a button to "Add dynamic content".

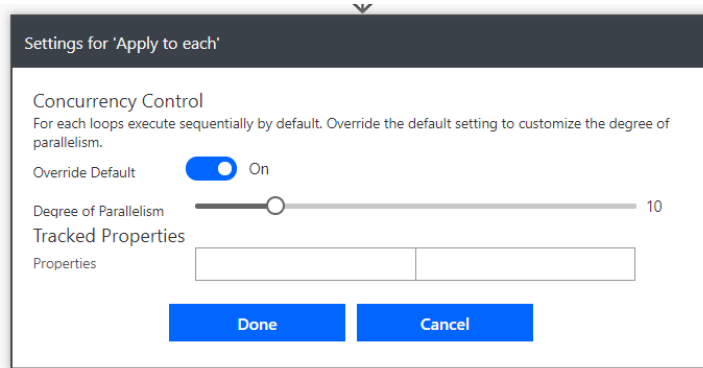


Parallel Execution

- Add parallel branch from above using ⊕



- Apply to each is sequential by default
 - Adding parallel execute to Apply to each



Agenda

- ✓ Processing Documents and Images
- Using Flow to Generate Word and PDF Files
 - Integrating Flow with Microsoft Forms
 - Using Flow to Automate Approval Processes
 - Managing Approvals using Approvals Center





DEMO

Generating PDF Files with the Word Online Connector

Agenda

- ✓ Processing Documents and Images
- ✓ Using Flow to Generate Word and PDF Files
- Integrating Flow with Microsoft Forms
 - Using Flow to Automate Approval Processes
 - Managing Approvals using Approvals Center





DEMO

Creating a Flow to Process Forms Created using Microsoft Forms

Agenda

- ✓ Processing Documents and Images
- ✓ Using Flow to Generate Word and PDF Files
- ✓ Integrating Flow with Microsoft Forms
- Using Flow to Automate Approval Processes
 - Managing Approvals using Approvals Center



Creating a Flow for Device Request Approval

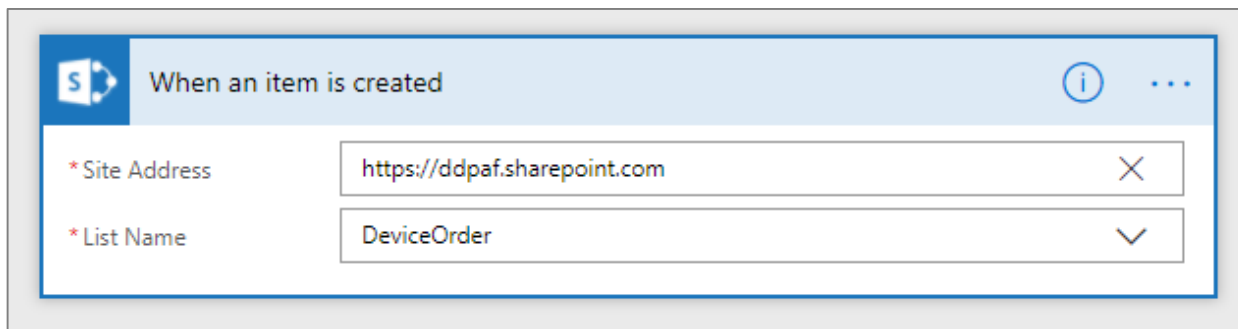
- Create a flow from blank



- Select a SharePoint trigger for **When an item is created**

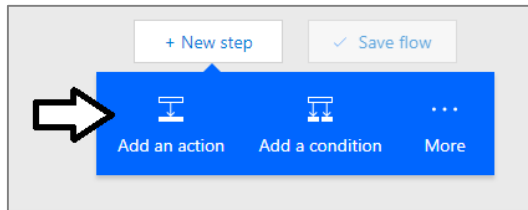


- Configure the trigger to use the **DeviceOrder** list

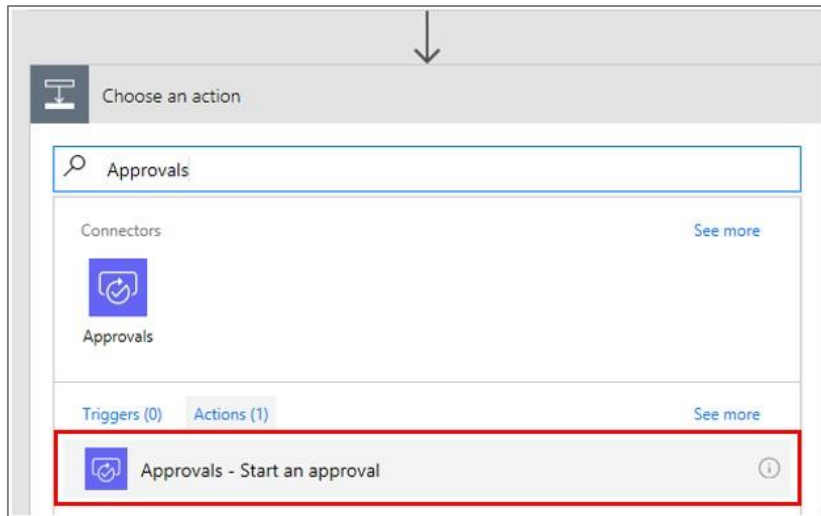


Add the Start an Approval Action

- An **Approval** process is added as an action




- Select the Approvals action named **Start an approval**



Approval Types

- There are two types of approvals
 - Determine behavior when there are two or more approvers
 - "Anyone" allows single approver to complete approval process
 - "Everyone" requires all approver to approve the request



The screenshot shows a 'Start an approval' dialog box. The title bar is light blue with a circular arrow icon on the left and an information icon and three dots on the right. Below the title bar, there is a label '* Approval type' followed by a dropdown menu. The dropdown menu is open, showing four options: 'Who should approve this?' (with a downward arrow), 'Anyone from the assigned list' (highlighted with a red border), 'Everyone from the assigned list', and 'Enter custom value'.



Building Out The Start an Approval Action

- You provide data which is sent to approver

Start an approval

* Approval type: Anyone from the assigned list

* Title: New device request for Title x

* Assigned to: Approver x ;

Details: A new device has been requested
Title x
\$ Price x
Comments: Comments x

Item link: Link to item x

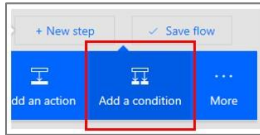
Item link description: This is the requested device

[Add dynamic content](#) +

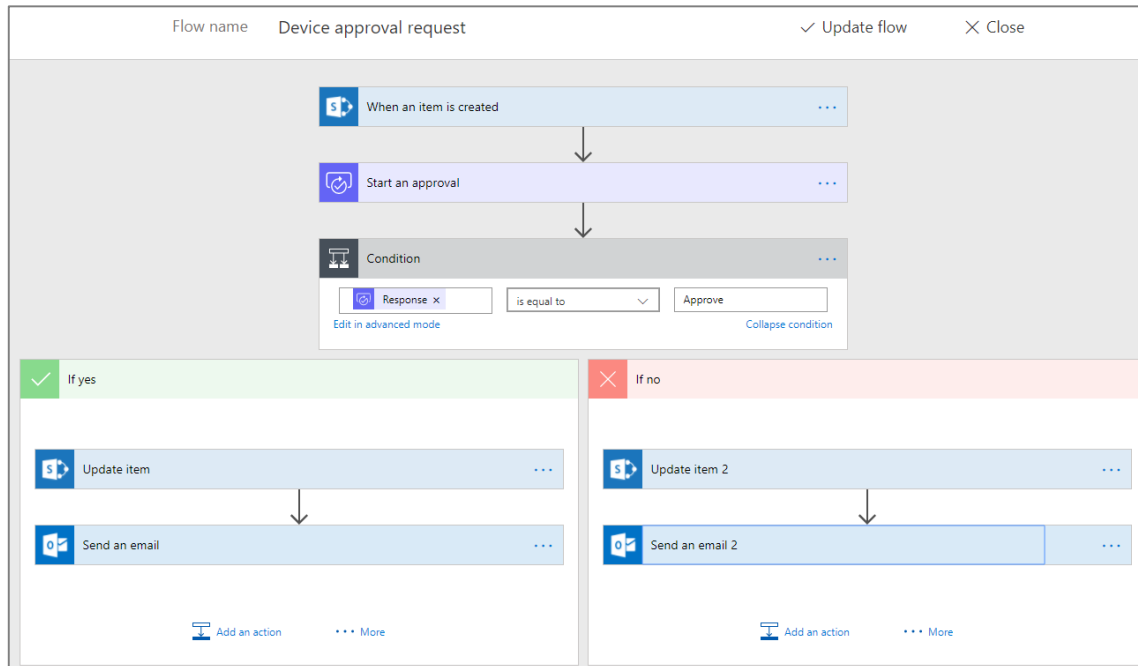


Responding to the Approval Response

- Start an Approval action followed by a condition
 - Allows flow to determine if approval was accepted or rejected

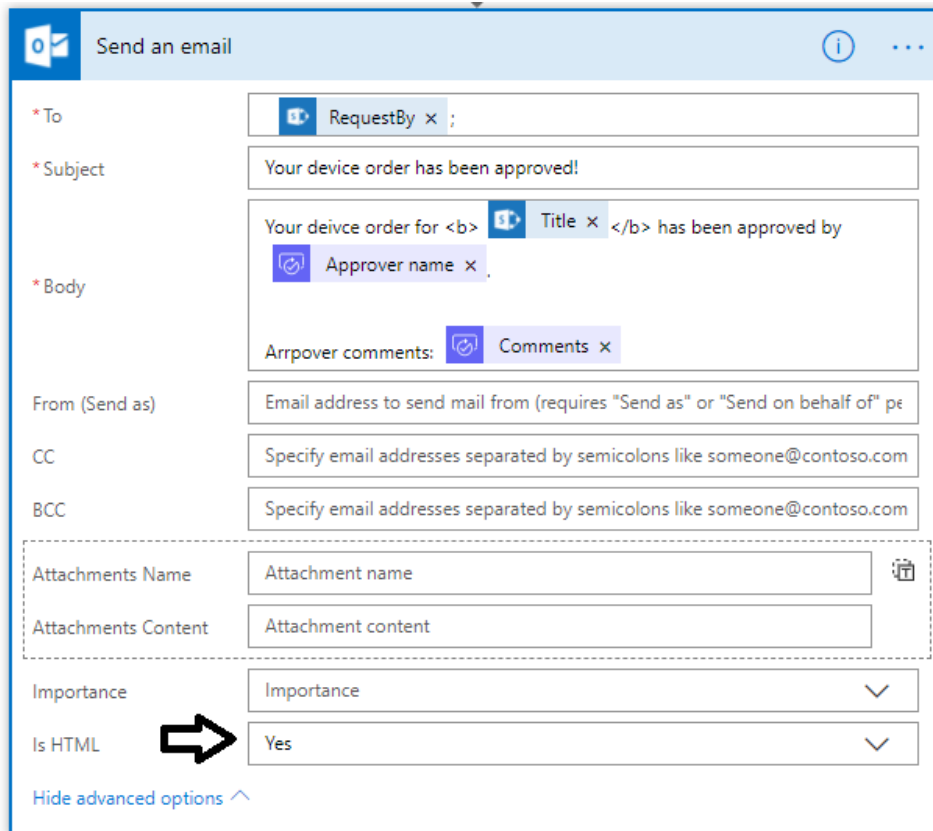


- Condition has If yes branch and If no branch for both outcomes



Implementing If yes Branch using Email

- If request is approved, send notification email to requestor



The screenshot displays the 'Send an email' dialog box with the following fields and values:

- To:** RequestBy x ;
- Subject:** Your device order has been approved!
- Body:**
 - Your device order for Title x has been approved by
 - Approver name x
 - Approver comments: Comments x
- From (Send as):** Email address to send mail from (requires "Send as" or "Send on behalf of" pe
- CC:** Specify email addresses separated by semicolons like someone@contoso.com
- BCC:** Specify email addresses separated by semicolons like someone@contoso.com
- Attachments:**
 - Name:** Attachment name
 - Content:** Attachment content
- Importance:** Importance
- Is HTML:** Yes (indicated by a large black arrow)

At the bottom left, there is a link: [Hide advanced options ^](#)



Testing the Approval Flow

- Start by creating a new device request

The screenshot shows the 'Device Ordering App' interface. On the left, three product cards are displayed: 'ProBook 4440s' (\$679.00), 'ProBook 4545s' (\$499.00), and 'Compaq Pro 4300' (\$859.00). The 'ProBook 4545s' card is highlighted. On the right, a form is filled out with the following details:

- Title: HP - ProBook 4545s
- Price: 499
- Approver: student@DDPAF.onmicrosoft.com
- Comments: I really need this laptop
- RequestBy: Student@DDPAF.onmicrosoft.com

A 'Submit device request' button is located at the bottom right of the form.

- Adding item to SharePoint list triggers approval process to start

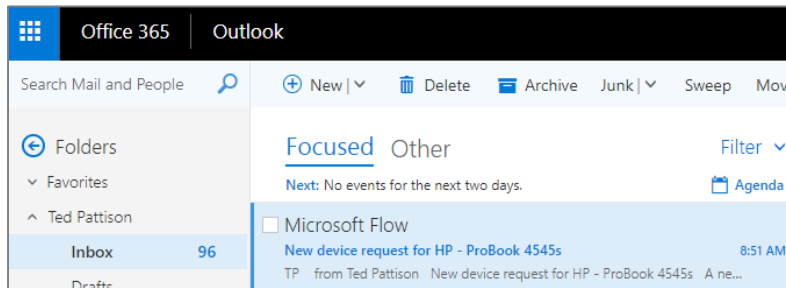
The screenshot shows a SharePoint list titled 'DeviceOrder' on the 'Critical Path Labs Team Site'. The list contains one item with the following details:

Title	DeviceID	Price	RequestBy	Approver	ApprovalStatus	Comments
HP - ProBook 4545s	45	\$499.00	Student@DDPAF.onmicrosoft.com	student@DDPAF.onmicrosoft.com	InReview	I really need this laptop

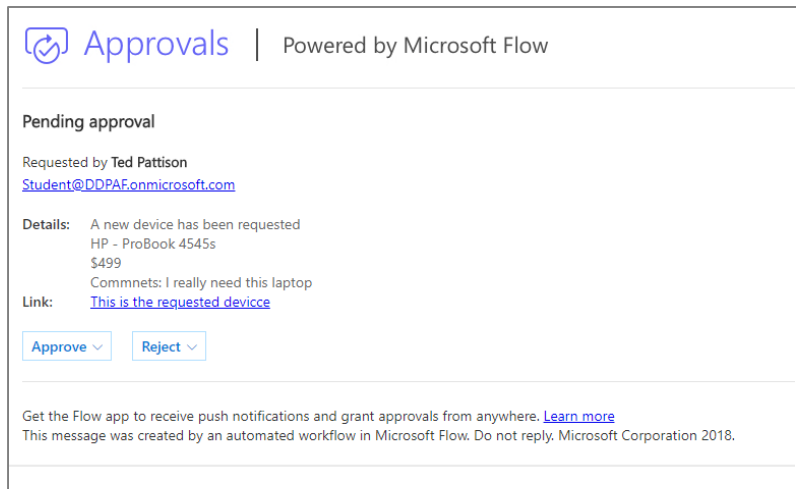


Sending Email Notification to an Approver

- The flow sends notification email to the approver
 - Flow execution currently paused inside **Start an Approval** action

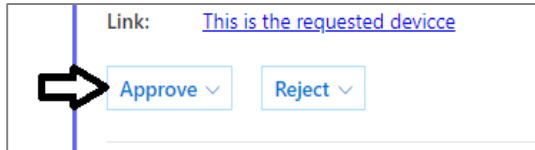


- Email allows approver to approve or reject approval request

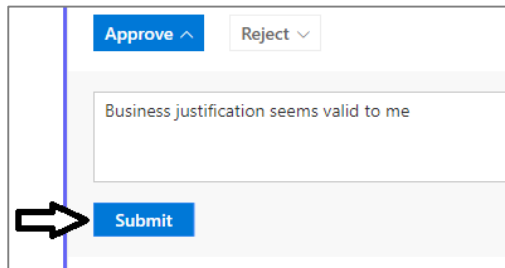


Approving an Approval Request

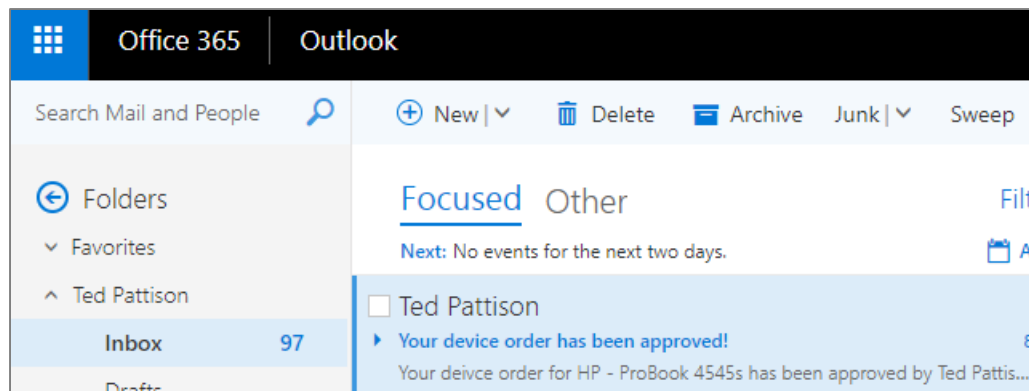
- Notification email provides button to approve or reject request



- Approver can enter comment and submit approval (or rejection)




- Approval or rejection unblocks flow which continue down appropriate branch
 - Approval response determines whether to send approval email or rejection email




Rejecting an Approval Request

- In the case of a rejected request...




Details: A new device has been requested
Toshiba - Portege Z935-ST4N02
\$979.99
Comments: I already have a great laptop but I want another one for no good reason.

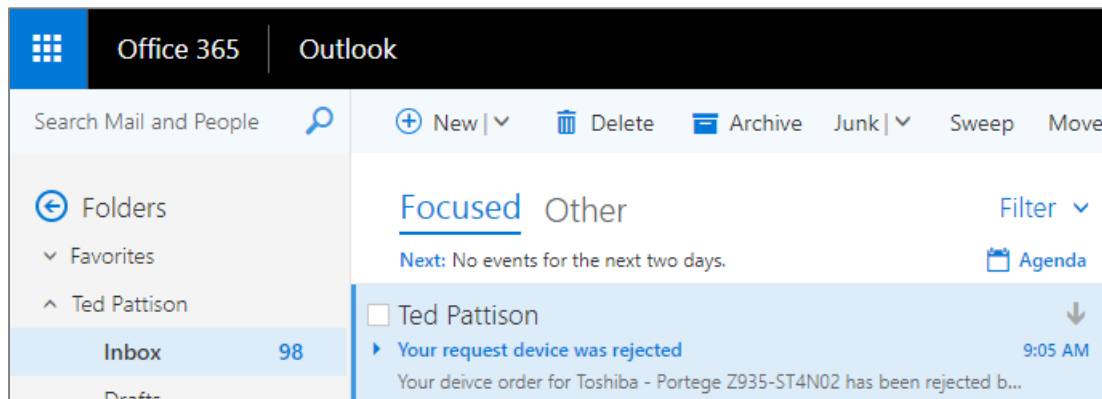
Link: [This is the requested device](#)

Approve  Reject ^

You only need one laptop!

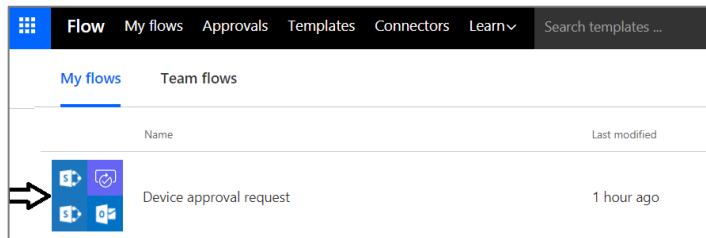
 Submit

- Approval flow sends a notification about rejection

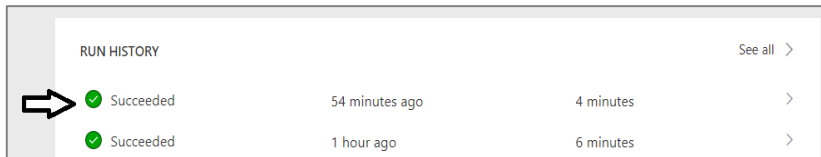


Run History

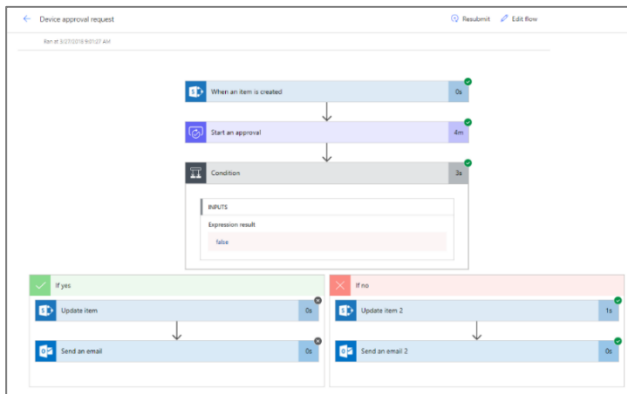
- Flow tracks run history of flow that have started
 - Click on a flow to see its RUN HISTORY list



- RUN HISTORY list has entry for each flow that has started



- Drilling into flow run history shows execution path and data



Updating a SharePoint List Item

- Approval flow can update SharePoint list items
 - Used to show SharePoint users the outcome of approval process

Home

CP Critical Path Labs Team Site

+ New Quick edit Export to Excel Flow PowerApps ...

DeviceOrder

Title ▾	DeviceID ▾	Price ▾	RequestBy ▾	Approver ▾	ApprovalStatus ▾
HP - ProBook 4545s	45	\$499.00	Student@DDPAF.onmicrosoft.com	student@DDPAF.onmicrosoft.com	Approved
Toshiba - Portege Z935-ST4N02	98	\$979.99	Student@DDPAF.onmicrosoft.com	student@DDPAF.onmicrosoft.com	Rejected



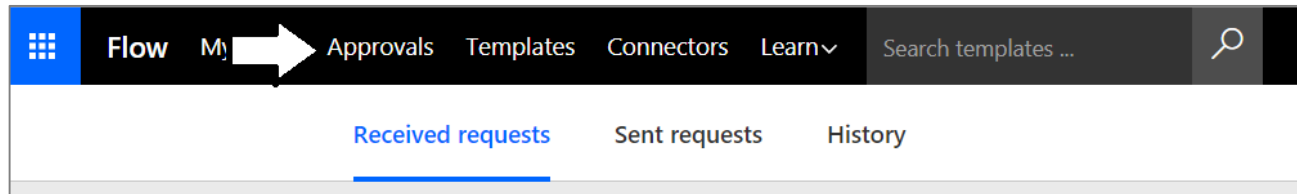
Agenda

- ✓ Processing Documents and Images
- ✓ Using Flow to Generate Word and PDF Files
- ✓ Integrating Flow with Microsoft Forms
- ✓ Using Flow to Automate Approval Processes
- Managing Approvals using Approvals Center



Approvals Center

- Microsoft Flow provides Approvals Center
 - Provides alternative to email for approve/reject processing
 - Accessible through browser



- Provides monitoring of completed approvals and pending approvals

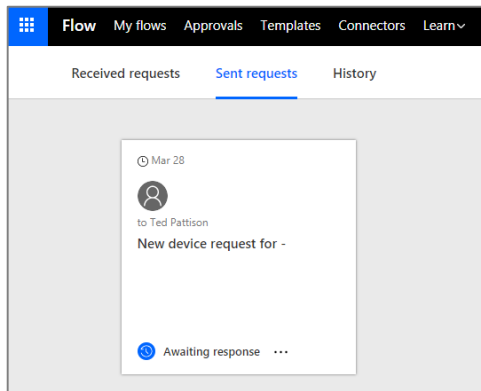
This screenshot shows the 'History' tab in the Microsoft Flow Approvals Center. It features a search bar labeled 'Filter by title' and a dropdown menu set to 'Received'. Below this is a table with four columns: 'REQUESTER', 'TITLE', 'DATE', and 'OUTCOME'. The table contains two rows of data.

REQUESTER	TITLE	DATE	OUTCOME
TP Ted Pattison	New device request for Toshiba - Portege Z935-S...	52 minutes ago	Rejected
TP Ted Pattison	New device request for HP - ProBook 4545s	1 hour ago	Approved

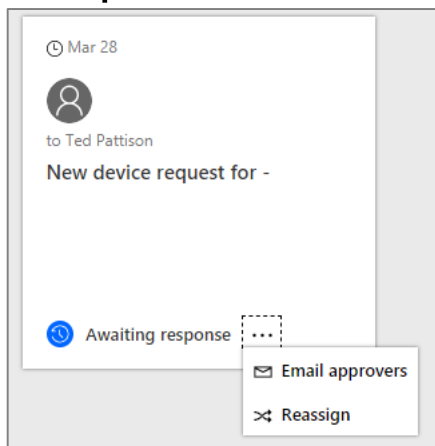


Examining Sent Request

- Requester can view requests he/she has submitted

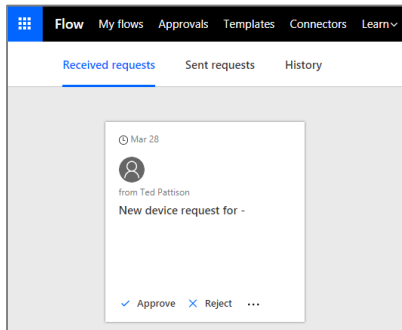


- Requester can email approver(s) or resign to different approver

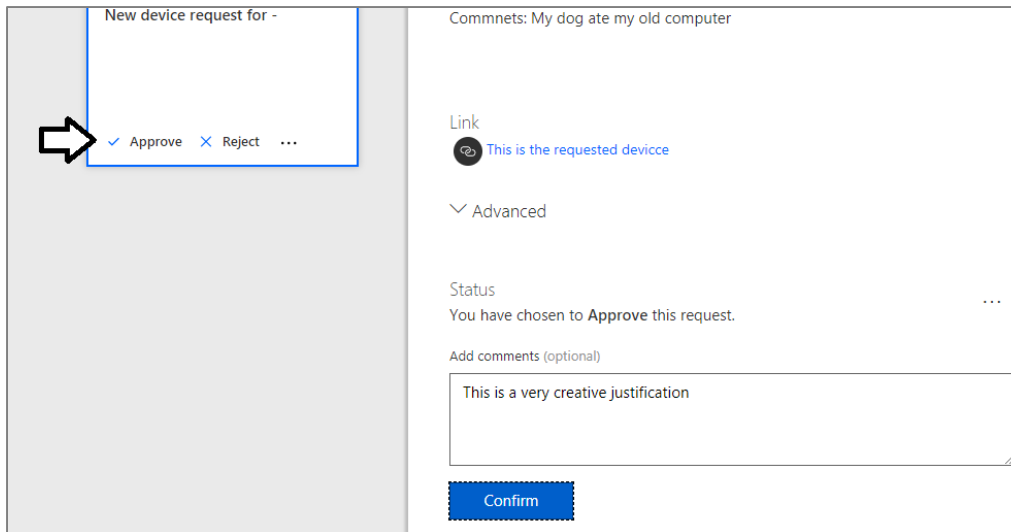


Examining Received Requests

- Approvers can see list of all their approval requests



- Approver can approve or reject approval request



Summary

- ✓ Processing Documents and Images
- ✓ Using Flow to Generate Word and PDF Files
- ✓ Integrating Flow with Microsoft Forms
- ✓ Using Flow to Automate Approval Processes
- ✓ Managing Approvals using Approvals Center

