Workflow Automation with Microsoft Power Automate

Achieve digital transformation through business automation with minimal coding

Aaron Guilmette

Workflow Automation with Microsoft Power Automate

Achieve digital transformation through business automation with minimal coding

Aaron Guilmette

Packt

BIRMINGHAM - MUMBAI

Workflow Automation with Microsoft Power Automate

Commissioning Editor: Pavan Ramchandani
Acquisition Editor: Heramb Bhavsar
Content Development Editor: Aamir Ahmed
Senior Editor: Hayden Edwards
Technical Editor: Shubham Sharma
Copy Editor: Safis Editing
Project Coordinator: Kinal Bari
Proofreader: Safis Editing
Indexer: Rekha Nair
Production Designer: Jyoti Chauhan

First published: Septemeber 2020

Production reference: 1180920

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-83921-379-3

www.packt.com

I want to dedicate this book to my kids: Liberty Maranatha, Hudson Asher, Glory Grace, Anderson Valor, and Victory Elizabeth. They're five of the reasons I write since I need a way to pay for all their shoes.

I also want to thank the staff at Packt Publishing, including Aamir Ahmed, Divij Kotian, and Hayden Edwards. They've suffered through four books with me. I'm not sure which of us is more a glutton for punishment.

Finally, this book is dedicated to everyone who wants to be more productive. While many people say that the best technology is invisible, I like to think that the best technology is that which lets you spend less time with technology and more time with the people you care about.

- Aaron

Packt.com

Subscribe to our online digital library for full access to over 7,000 books and videos, as well as industry leading tools to help you plan your personal development and advance your career. For more information, please visit our website.

Why subscribe?

- Spend less time learning and more time coding with practical eBooks and Videos from over 4,000 industry professionals
- Improve your learning with Skill Plans built especially for you
- Get a free eBook or video every month
- Fully searchable for easy access to vital information
- Copy and paste, print, and bookmark content

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.packt.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at customercare@packtpub.com for more details.

At www.packt.com, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

Contributors

About the author

Aaron Guilmette, a Microsoft Teams technical specialist at Microsoft, provides guidance and assistance to customers adopting the Microsoft 365 platform. He primarily focuses on collaborative technologies, including Microsoft SharePoint Online, Microsoft Exchange, and Microsoft Teams. He also works with identity and scripting solutions.

He has been involved with technology since 1998 and has provided consulting services for customers in the commercial, educational, and government sectors internationally. Aaron has also worked on technical certification exams and instructional design for Microsoft and other organizations.

Aaron lives in the Detroit, Michigan area with his five children. When he's not busy solving technical problems, writing, or running his children to events, he's likely to be making a pizza.

About the reviewers

Mohamed Abdel Ghaffar is a technical consultant with over 14 years' experience with Microsoft technologies and platforms (including web development, integration architecture, data analytics, the Power Platform, cloud technologies, Azure, and AI) who has architected, designed, and developed enterprise solutions for governmental, banking, and telecommunication entities in Egypt and the Gulf region. Mohamed has achieved certifications related to the cloud, data analytics, and the Power Platform. In his free time, Mohamed likes to play chess, football, and spend quality time with his family.

Himanshu Churiwala is an experienced technical architect with expertise in Microsoft technologies, including Microsoft 365/Office 365, Azure, handling big cloud migration projects, and SI projects. He has designed, architected, developed, and integrated multiple enterprise solutions on the Microsoft technology stack. He has conducted several boot camps and training on new technological advances in the Microsoft 365 stack. Throughout his career, he has performed multiple roles, including those of mentor, developer, tech architect, trainer, and project manager. He is certified in Microsoft technologies and has numerous certifications under his belt. He is very keen on learning new technologies and is very passionate about his work.

Packt is searching for authors like you

If you're interested in becoming an author for Packt, please visit `authors.packtpub.com` and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Table of Contents

Preface

Microsoft Power Automate is a business productivity tool designed to bring automation capabilities to cloud and on-premises applications. The ubiquity and proliferation of cloud-based software-as-a-service technologies has resulted in organizations using a myriad of disconnected tools as part of their business operations.

Throughout this book, you'll learn how Power Automate can be used to drive efficiency in automation and connecting all the parts to reduce the time spent performing repetitive tasks.

Who this book is for

This book is for individuals who are new to the Microsoft Power Platform (Power Automate, Power Apps, and Power BI). Readers can have little or no experience in coding, automation, or software practices. This book will walk users through very basic tasks to gain an understanding of the tools and leave the readers with completed workflows that can be exported and used in their daily work, or further refined and adapted to produce more complex processes.

What this book covers

This book is divided into sixteen chapters. It starts by covering the basic terminology and providing a general overview of the interface and proceeds to build increasingly detailed and complex workflows.

`Chapter 1`, *Introducing Power Automate*, introduces the basic concepts of workflow and automation.

`Chapter 2`, *Getting Started with Power Automate*, dives into the Power Automate interfaces and application components, and explains how to create a basic flow to see how it all works.

`Chapter 3`, *Working with Email*, explains how to configure Power Automate to interact with email.

`Chapter 4`, *Copying Files*, covers using Power Automate to copy files between cloud services.

`Chapter 5`, *Creating Button Flows*, explains how to use button (also known as manual or instant) flows and shows how they can be executed from mobile devices.

`Chapter 6`, *Generating Push Notifications*, explains how to create a push notification to a mobile device.

`Chapter 7`, *Working with Team Flows*, covers how to share flows with your peers.

`Chapter 8`, *Working with Conditions*, explains how to use conditional execution to cause different outcomes for flows.

`Chapter 9`, *Getting Started with Approvals*, discusses how to use approvals workflows for documents.

`Chapter 10`, *Working with Multiple Approvals*, expands your knowledge of approvals with added branches or sequential approvals.

`Chapter 11`, *Posting Approvals to Teams*, explains how to integrate approval workflows with Microsoft Teams.
`Chapter 12`, *Using a Database*, covers connecting to a database and using it to store or retrieve information for a process.

`Chapter 13`, *Working with Microsoft Forms*, explains how to use Power Automate to process data from Microsoft Forms.

`Chapter 14`, *Accepting User Input*, explores configuring a button or manual flow to accept a variety of user input.

`Chapter 15`, *Exporting, Importing, and Distributing Flows*, discusses how to import, export, and transport your flows between environments.

`Chapter 16`, *Monitoring and Troubleshooting Flows*, contains troubleshooting tips for when things go wrong.

To get the most out of this book

To get the most out of this book, you should have a Microsoft 365 subscription with access to Power Automate Per-App or Per-User plans to enable premium connectors, as well as an internet-connected computer with a modern browser such as Microsoft Edge, Google Chrome, or Mozilla Firefox.

You can obtain a 30-day trial of Microsoft 365 by going to `https://www.microsoft.com/en-us/microsoft-365/try`.

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "In this case, we're going to use an existing folder called `Archive`, but then create a new folder path based on the date of the received email."

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Launch the Power Automate web portal (`https://flow.microsoft.com`) and select **+ Create flow**."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at `customercare@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packtpub.com/support/errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packt.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit `packt.com`.

1
Section 1 - What is Power Automate?

In this section, the reader will become familiar with the concepts of Power Automate.

This section comprises the following chapter:

- `Chapter 1`, *Introducing Power Automate*

1
Introducing Power Automate

Business activities in the information age are filled with repetitive tasks: receive an email, generate a purchase order, send a message, route a document, approve a time-off request. In many cases, these activities don't generate real value, though they do need to get done to help support business goals.

Computers introduced the promise of helping us do more, but a lot of that has resulted in there being more to do in order to get the same value. What if we could use technology to handle routine tasks and save our strength for doing the things that require skill and thinking?

In this book, we're going to learn the basics of Microsoft Power Automate, a tool designed to help you automate repetitive tasks and get you back to generating value.

This chapter focuses on getting an understanding of some of the basic concepts of Power Automate:

- What is Power Automate?
- Choosing what to automate
- Reviewing general terminology
- Learning Power Automate terminology

By the end of this chapter, you should have an understanding of the components of Power Automate and how they can be used to streamline daily operations.

What is Power Automate?

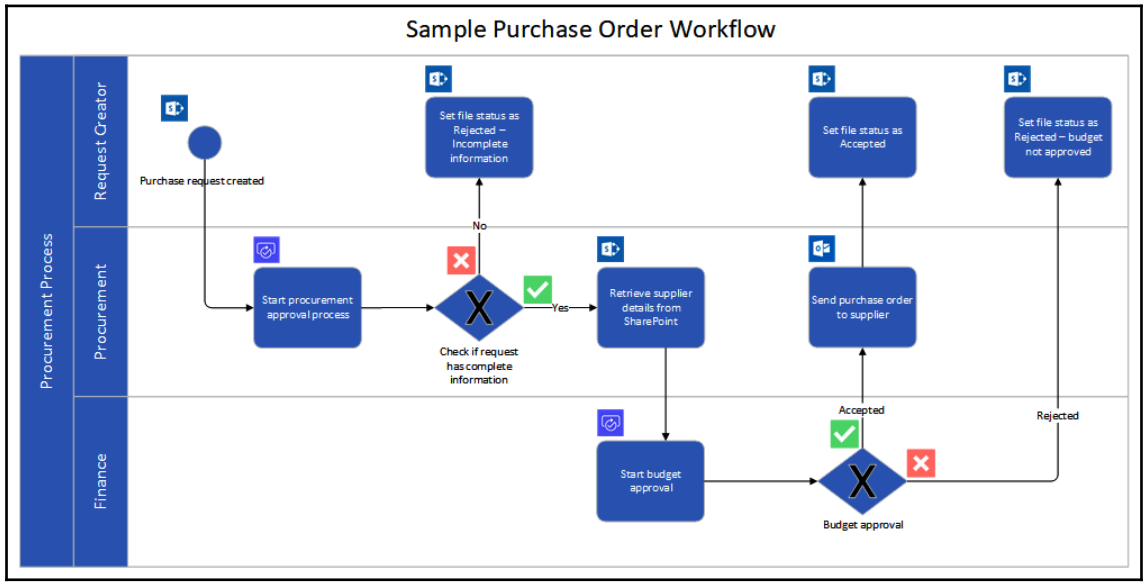
Power Automate, part of the Power Platform family of products, is a workflow engine that can be used to automate common business processes or sequences based on a number of conditions or scenarios. Power Automate (formerly known as *Microsoft Flow*) is a web-based tool designed to interface with a growing library of software from both Microsoft and other vendors.

Many readers may be familiar with the concept of *SharePoint workflows*. In the SharePoint world, you can use products such as SharePoint Designer and Workflow Manager to kick off business processes based on activities such as a document being checked in to a particular library. One of the great things about SharePoint workflows is that they can automate business processes and tasks inside the SharePoint environment. However, one of the drawbacks of SharePoint workflows is that they can really only automate business processes and tasks *inside the SharePoint environment*.

While some organizations (or even vendors) have created bridge or integration packages to tie SharePoint workflows to external products, many of them are limited to interfacing with data inside SharePoint.

This is the power of the Power Automate platform—it has native connectivity to hundreds of applications out of the box. It's also extensible, meaning you can develop your own connectivity solutions to work with your custom apps. You can even use HTTP to interact with nearly any REST-based interface for any application. Power Automate's capabilities are limited only by your imagination and the services offered by the applications you wish to integrate.

In the following diagram, a sample purchase order workflow ties together the SharePoint, Outlook, and Microsoft Approvals apps:

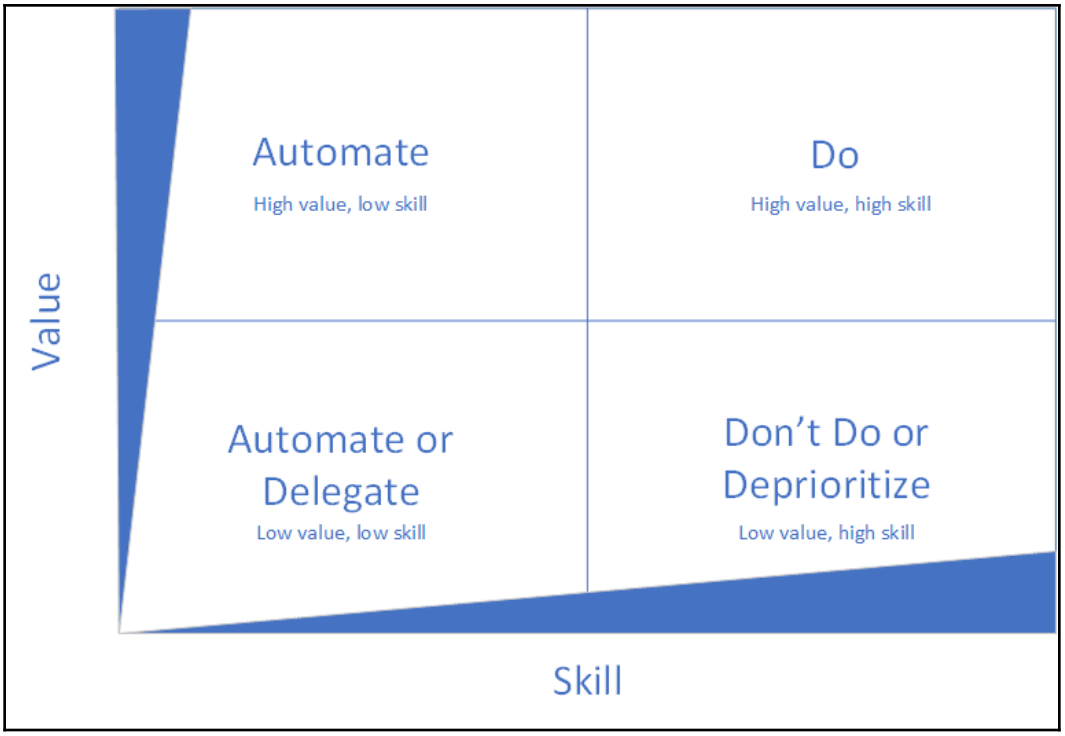


Choosing what to automate

Frequently, people do work that they don't need to do. It's important to differentiate between "work that doesn't need to be done at all" and "work that doesn't require you to do it."

From both the administrative and end user perspectives, there are a number of activities, processes, and tasks in Office 365 and other line-of-business applications that can be automated through the use of Power Automate.

Automation is a key business technology to reduce the impact of repetitive, low-skill tasks on the workforce. Consider the following diagram:



When looking at business processes, they can generally be divided into one of the four quadrants:

- **High value, low skill**: Requires minimal specialist skill, but is critical or produces high-value output
- **High value, high skill**: Requires human intelligence or processing to determine value, skills, and relationships
- **Low value, low skill**: Requires minimal specialist skill and also produces an output of small value
- **Low value, high skill**: Requires a high degree of focus or skill but produces an output of small value.

As you examine tasks in your daily routine, you can evaluate them against the preceding matrix to help understand whether it is something that *can* or *should* be automated. Items that fall into the "low skill" quadrants are very good candidates for automation.

Look at the following examples:

- Running a report of the previous day's sales totals is a repetitive task that requires low specialist skill. If possible, you should seek to automate this task.
- Calling a customer for follow-up on a demo unit that was sent. This is a high-touch, high-value activity, but requires the personalization and complexity of human relationship management to execute effectively. This task is not a good candidate for automation.

In the upcoming sections and chapters, we're going to get familiar with the terminology and interface of Power Automate, and then learn how to connect to common applications to solve business problems.

Reviewing general terminology

You've already seen a few terms, and if you're familiar with SharePoint or other collaboration tools, they may be recognizable. But just to make sure we have a solid foundation on which to build, we're going to go over some basic terminology, and then we'll start getting into specific Power Automate terminology.

Business process

A business process is any sequence of tasks needed to accomplish the business's purpose. This may be something as simple as submitting a time card or getting a signature on a purchase order. Business processes generally fall into three categories:

- **Primary or operating processes**: These typically result in some sort of customer value delivery, such as a customer placing an order and the business shipping a product. They also may include things such as product design and engineering.
- **Support**: These processes are necessary for the primary or operating processes to take place. For example, purchasing materials to make a product, or training an employee would generally be regarded as support processes.
- **Management**: These are processes to oversee the operating and support processes, or to improve those processes. Examples of management processes might be reviewing and making recommendations about the procurement or employee onboarding processes.

Automation can be used with processes in all of these categories. Power Automate can be used to automate some or all parts of many types of business processes.

Workflow

A workflow can be thought of as the individual steps to achieve a particular business process. For example, *employee onboarding* may be a business process, and *ordering a new computer* may be a workflow task associated with completing the *employee onboarding* business process.

Power Automate can be used to automate some or all parts of a workflow.

REST

REST is an industry acronym for **RE**presentational **S**tate **T**ransfer. It is used to describe a method for interacting with computer systems. In a REST-based system, client devices generally send HTTP verbs (or actions) to a target system **uniform resource identifier** (**URI**) to input or retrieve data.

For more information on the history and design principles of REST architectures, refer to `https://restfulapi.net/`.

Like many technologies on the Microsoft 365 platform, Power Automate utilizes REST to enable a high volume of performant transactions.

Next, we'll look at Power Automate-specific terminology.

Learning Power Automate terminology

As we begin working with Power Automate, it will be important to understand the core terminology that is being used. You'll need to be able to differentiate between the following terms so you can choose where to apply the correct business logic and processes.

Flow

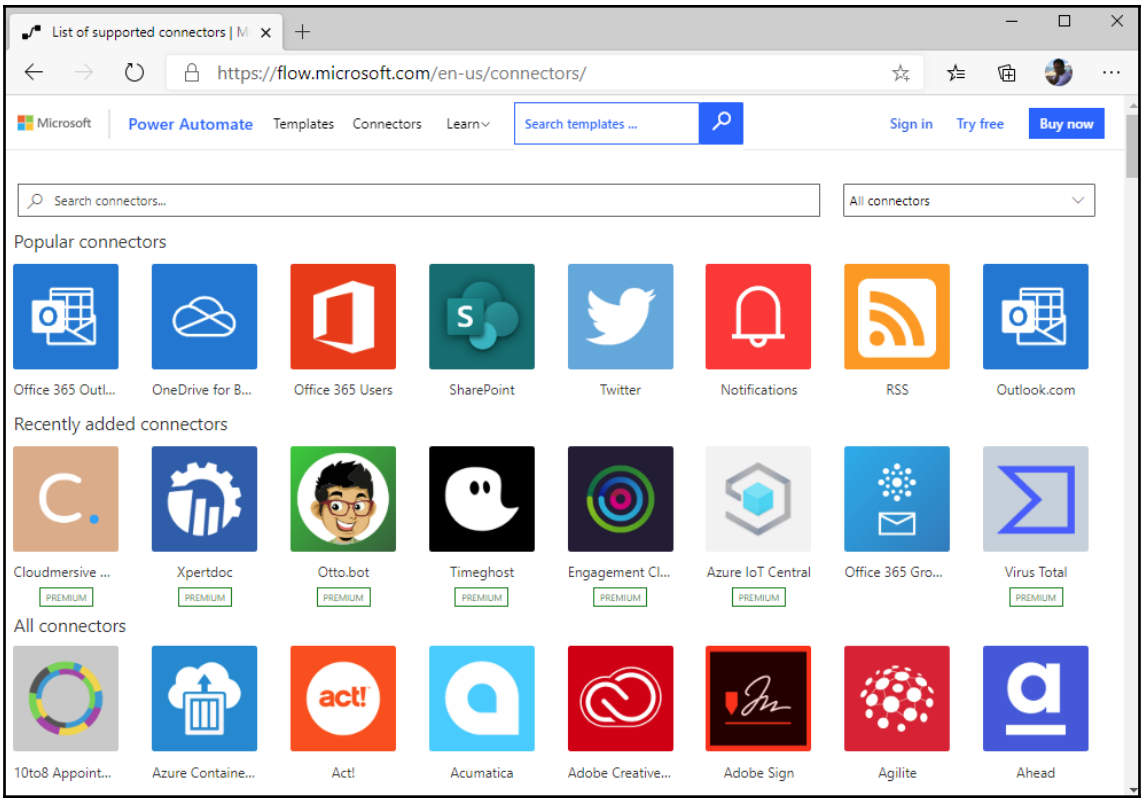
Flow is simply the logical grouping of connectors, triggers, conditions, and actions used to automate actions. Flows are currently divided into the following categories:

- **Automated**: Flows that happen based on triggers or events
- **Button**: Also known as Instant or Manual, these occur when initiated by a user
- **Scheduled**: Timed events that occur at specific intervals
- **Approval**: A process where requests are routed through an approval chain
- **Business process**: A high-level process comprising smaller tasks and workflows
- **UI flows**: Also known as **Robotic Process Automation**, or **RPA**, used to automate legacy (usually non-REST-based) apps

Each flow has different use cases, triggers, and configuration capabilities.

Connectors

Connectors are the components that are used to directly interface with both source and target systems. Connectors contain the information required to interact with applications. Examples of connectors are shown in the following screenshot:



Connectors are generally broken down into two tiers: **standard** and **premium**. Standard connectors are generally included with all Power Automate plans (such as Power Automate for Office 365), while premium connectors have additional costs associated with them.

Triggers

Triggers are the activities that cause the flow to begin. The core types of triggers are as follows:

- **Automated**: An automated flow happens automatically based on a particular type of event (such as a new file being uploaded to a site or an email being received).
- **Instant**: Also known as a manually-triggered flow or a button flow, instant flows are triggered on demand by a user.
- **Scheduled**: This time-based option happens on a recurring basis.

The available triggers may depend on which connectors are being used.

Actions

Actions describe the types of activities performed by a flow (such as copying a file, posting to a Teams channel, or sending an email).

Branching

Branching is used to describe the concept of different series of business logic or actions that can happen. Branches may be invoked through the use of conditions (in the form of the `if-then` construct) or through the use of parallel branches (different sets of actions or logic that are simultaneously executed).

Conditions

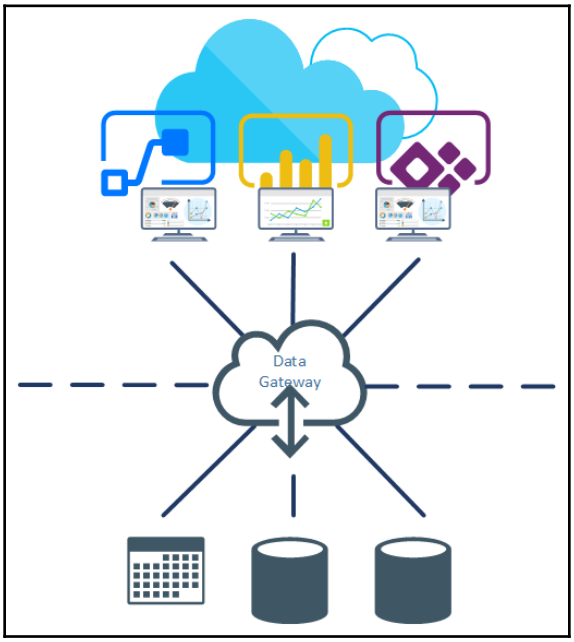
Conditions are used to evaluate and select the circumstances under which actions will be performed. Conditions may take the form of time or schedule constraints, values received through user input or reading files, or other calculated values. Conditions can lead to branches or different sets of logic that can be executed depending on the results of items as they are evaluated.

Common Data Service

Common Data Service is a storage mechanism similar to database tables that allows organizations to store business data. Data is stored as an *entity*, which is a set of related records and fields. A variety of applications, such as Dynamics 365, Power Apps, Power Automate, and Power BI, can use data stored in Common Data Service. Advanced Power Automate scenarios, such as working with Dynamics 365 entities, may require access to Common Data Service.

Gateways

A **gateway** (also known as a data gateway) is a software application installed on an on-premises computer. This application is used to facilitate access for the Power Platform services to data sources located in an on-premises environment. For example, a data gateway may be used to allow Power Automate to read items from an on-premises SharePoint Server list. The following diagram shows the data gateway presenting on-premises data to Power Automate, Power BI, and Power Apps services:



The data gateway, shown in the previous diagram, acts as a conduit between the on-premises data sources and Power Platform services (such as Power Automate, Power BI, and Power Apps).

Steps

Steps are the individual evaluations and actions that a flow executes. Steps are ordered in a methodical manner. Here are some examples of steps:

- Posting a Teams channel message
- Reading an email

- Saving a file
- Determining whether a variable has a particular value

Every flow is made up of one or more steps.

Templates

A template comprises a set of connectors, triggers, and actions designed to accomplish a predefined purpose. Templates allow standardized deployments and the reuse of common components and configurations. Templates also allow you to be able to create flows in sandbox or test environments and then export and deploy them in production environments, mitigating the potential risks of a misconfigured flow.

Understanding the terminology, both business- and Power Automate-specific, will help you as we move toward creating flows.

Summary

In this chapter, we discussed the high-level concepts of Power Automate, including how it relates to solving problems (such as business processes or workflows). We also covered introductory knowledge concepts such as actions, triggers, and conditions, and how they work together to produce desired outcomes. Understanding the types of triggers and connectors will help you design flows and business processes that reduce repetitive work and produce value in your organization.

In the next chapter, we're going to begin exploring the Power Automate interface and use it to create simple flows.

2
Section 2 - Basic Flow Concepts

In this section, the reader will become familiar with navigating the Power Automate interface and using connectors, triggers, actions, and templates to automate tasks.

This section comprises the following chapters:

- `Chapter 2`, *Getting Started with Power Automate*
- `Chapter 3`, *Working with Email*
- `Chapter 4`, *Copying Files*
- `Chapter 5`, *Creating Button Flows*
- `Chapter 6`, *Generating Push Notifications*
- `Chapter 7`, *Working with Team Flows*

2
Getting Started with Power Automate

In the first chapter, we introduced some of the basic concepts of Microsoft Power Automate. Now that you're familiar with some of the ideas and terminology behind Power Automate, let's take a look at navigating the interface.

As a Power Automate user, you'll likely have access to the web portal interface (most commonly used) as well as the mobile interface. If you administer a Microsoft 365 tenant, you'll also have some administrative features available.

In this chapter, we'll tackle the following:

- Logging in to Power Automate
- Creating your first flow

You'll learn the features of the interfaces, and then we'll create a basic flow to post to a Teams channel.

Let's get started!

Logging in to Power Automate

Power Automate has three distinct experiences:

- **End user web portal**: User or creator interfaces with the purpose of designing, importing, saving, exporting, and executing flows
- **Mobile app**: User or creator interfaces with the purpose of designing, importing, saving, exporting, and executing flows, formatted specifically for mobile devices

- **Admin**: The overall administration of the Power Automate environment for your tenant, including the number of executions or runs and data gateway configurations

Power Automate tenant administration is largely outside the scope of what we're going to be learning from a design aspect, but should you ever need to administer the Power Automate environment, you'll know where to start.

Microsoft Power Automate is a cloud-only solution—there is no on-premises counterpart. Microsoft Power Automate is available in the following Microsoft 365 clouds:

- Worldwide/Commercial: `https://flow.microsoft.com`
- United States Government Community Cloud (Moderate): `https://gov.flow.microsoft.us`
- United States Government Community Cloud (High): `https://high.flow.microsoft.us`

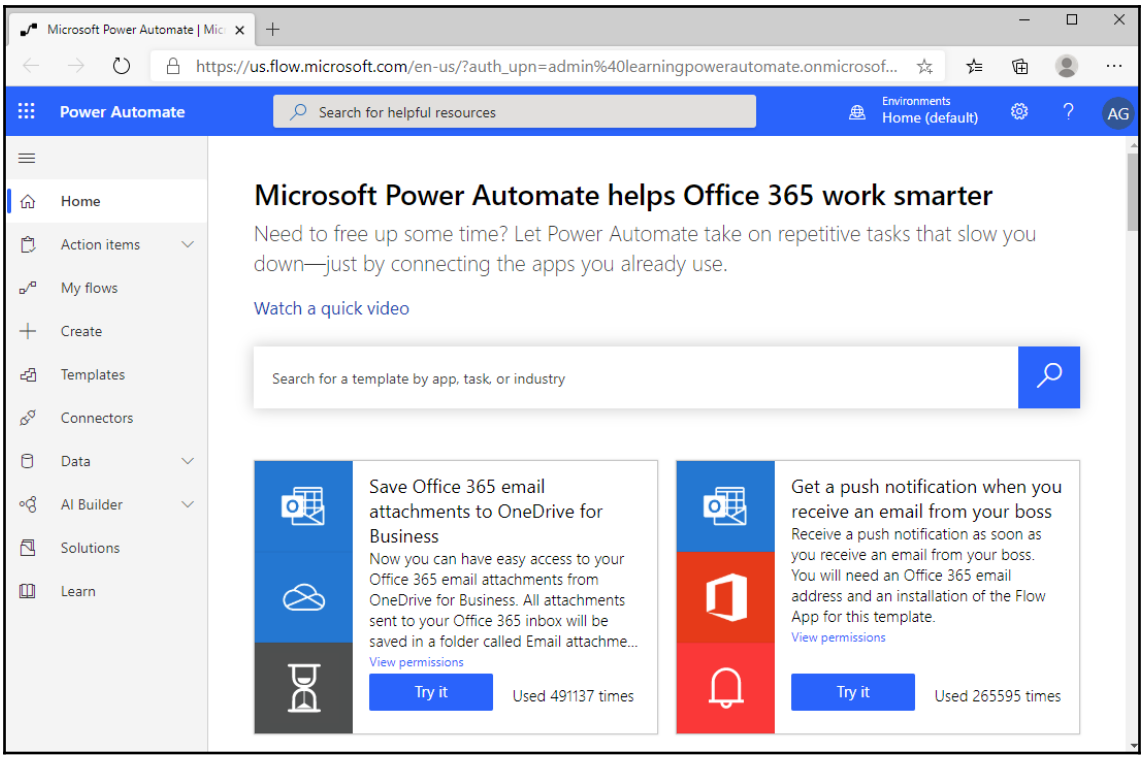
For the purposes of this book, we'll focus on the features of the worldwide commercial solution. Power Automate also contains the concepts of *standard* and *premium* connectors. Standard connectors are available for all Power Automate plans, including the Power Automate plans that come with Office 365. Premium connectors will require the use of a standalone Power Automate license. For up-to-date licensing details, see `https://docs.microsoft.com/en-us/power-platform/admin/powerapps-flow-licensing-faq`.

End user web portal interface

The end user web portal interface is where you'll design, manage, and execute your flows. To access it, you log in to `http://flow.microsoft.com`, or log in to the Microsoft 365 end user portal (`https://portal.office.com`) and click on the **Power Automate** tile. If you're using one of the other sovereign clouds, use the links in the previous section or contact your reseller for specific links.

If you don't have access to Microsoft Power Automate within your organization, you'll want to go start a Microsoft 365 trial so you can gain access to it. To start a trial subscription, navigate to `https://www.microsoft.com/en-us/microsoft-365/try` and click the **Try one month free** link.

The web portal interface is depicted in the following screenshot:



The left part of the page features a navigation menu that includes options to review action items (approvals and business process flows), as well as creating and managing your own flows and searching for templates and connectors. The following list describes the options available:

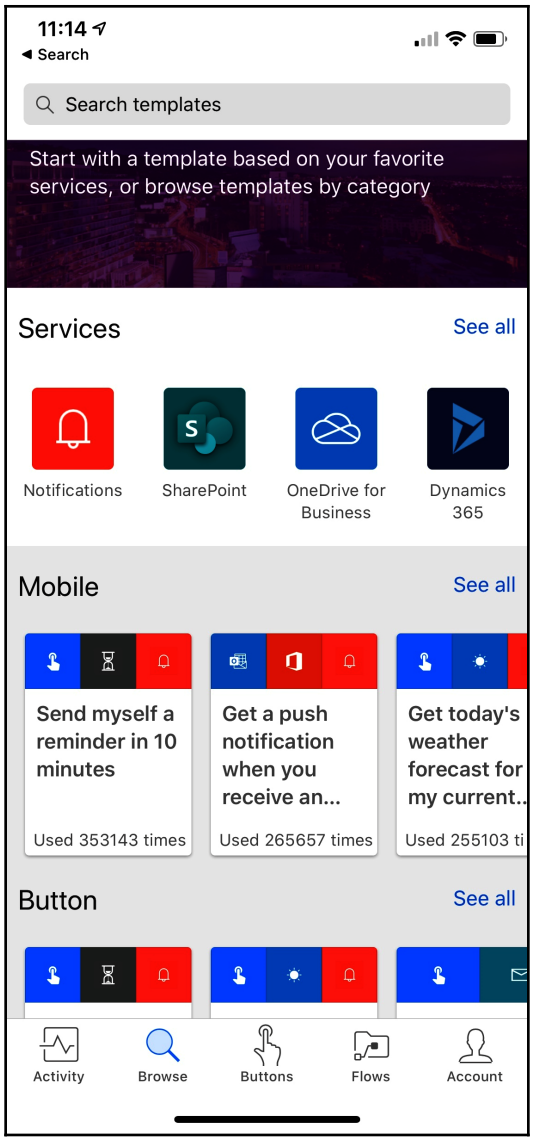
- **Home**: The dashboard displayed upon first logging in
- **Action items**: The top-level menu item containing links to **Approvals** (approval workflows) and **Business process flows** (multi-step task-oriented flows)
- **My flows**: Flows that you have created (or that have been shared with you)
- **Create**: The interface for creating new flows
- **Templates**: Pre-canned flows comprising connectors that only need to be customized

- **Data**: A top-level menu item containing links to the following:
    - **Entities** (objects in the Common Data Service)
    - **Connections** (authenticated configurations for connectors)
    - **Custom connectors** (custom interfaces to REST API applications)
    - **Gateways** (basic information about existing Data Gateways to connect to on-premises data sources)
- **AI Builder**: A top-level menu item containing links to **Build** (AI Builder tools to create flows that predict outcomes) and **Models** (AI Builder models that you can use to build in AI-based flows)
- **Solutions**: Containers for managing the transporting and life cycle of Power Platform applications
- **Learn**: A link to the landing page for Power Automate on `docs.microsoft.com`

In this book, we'll explore some of the built-in templates and components to use when building your own flows. You should spend some time looking through the connectors (`https://flow.microsoft.com/en-us/connectors/`) and templates (`https://flow.microsoft.com/en-us/templates/`) to start getting ideas of what types of applications and services you can integrate with Power Automate. You'll want to focus on connectors and templates that help you integrate apps that your organization uses on a regular basis.

Mobile app interface

The Power Automate mobile app, available for both the iOS (`https://apps.apple.com/us/app/microsoft-flow/id1094928825`) and Android (`https://play.google.com/store/apps/details?id=com.microsoft.flow`) platforms, offers a similar design aesthetic and experience, as shown in the following screenshot:



The following options are available on the home screen:

- **Activity**: Shows a list of recently completed flows and approval activities
- **Browse**: Search connectors and templates to begin creating a new flow
- **Buttons**: Allows you to create button (or instant) flows to be triggered from your mobile device

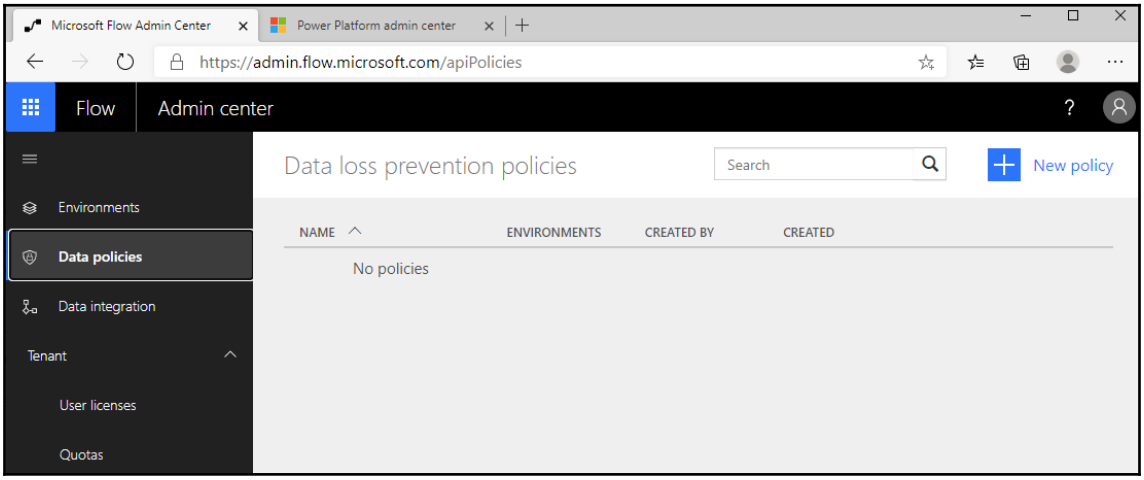
- **Flows**: Lists your own and your team's (that is, shared) flows
- **Account**: Your Office 365 tenant account sign-in information

The mobile experience focuses on surfacing flow components specifically tailored to mobile devices. You can design and review flows from both the mobile and web platforms. The majority of this book will focus on the end user web portal experience, but there will be a few opportunities to use the mobile app.

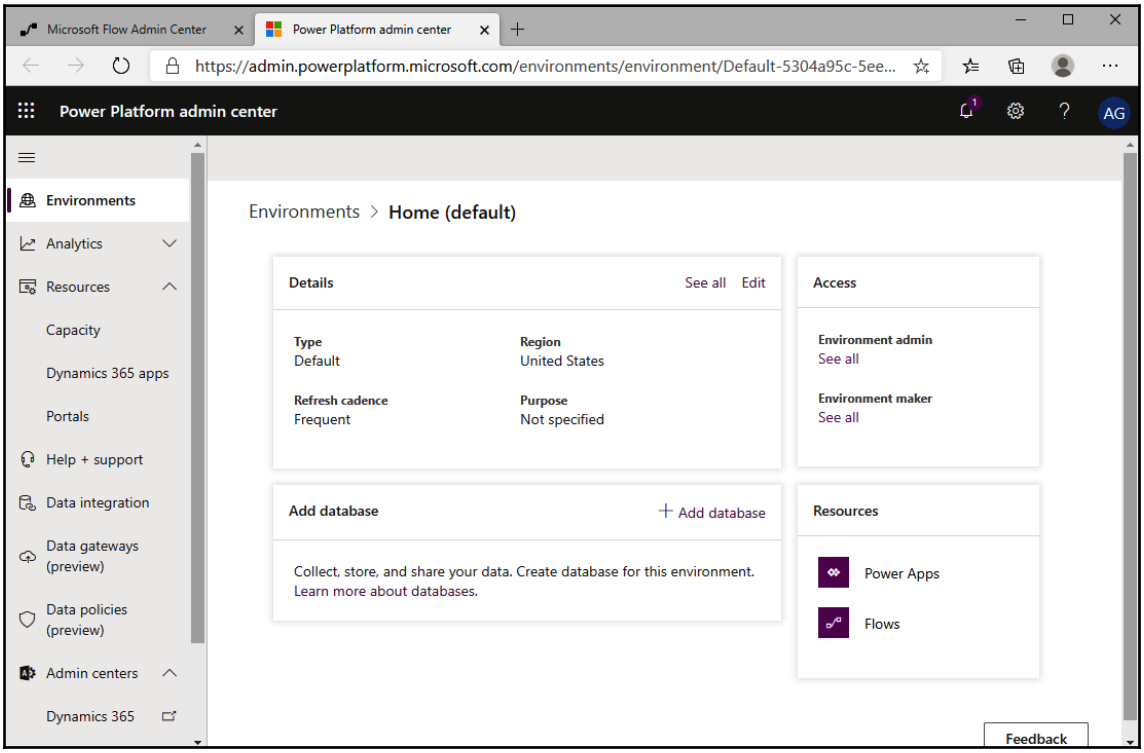
Admin interface

The admin controls for Power Automate are spread across two interfaces: the **Power Automate Admin center** (also known as the **Flow Admin center**) and the new **Power Platform Admin center**. You'll only be able to access these interfaces if you're a global admin of a Microsoft 365 tenant or have been granted Power Platform admin rights.

The Power Automate Admin center (`https://admin.flow.microsoft.com`) allows you to keep track of runs executed tenant-wide, view license information, view overall environment data, and configure **Data Loss Prevention** (**DLP**) policies:



The Power Platform Admin center (`https://admin.powerplatform.microsoft.com`) contains additional administrative tools and configuration data that apply not only to Power Automate, but also to Dynamics 365, Power BI, and Power Apps:



While neither of the admin interfaces will be used extensively in this book, you should understand that they exist and that administrators of the overall Microsoft 365 environment can modify configurations and view statistical data across the organization.

Now that you've seen the various interfaces for Power Automate, let's start creating!

Creating your first flow

The best way to see Power Automate in action is to start creating a flow. In this example, we're going to create a flow that monitors Twitter for a certain hashtag and then posts a notification to a Teams channel. Such a flow might be useful if you're trying to gauge or capture customer sentiment for a product or service, track trending public health topics related to certain keywords, monitor engagement activity, or other topic-based alerts on a social media platform.

Understanding the flow components

This particular flow is going to rely upon a few components:

- An identity for Twitter (username and password)
- A trigger that monitors Twitter for certain words or phrases
- An identity for Office 365 (username and password)
- A Microsoft Teams team
- An action that posts to Microsoft Teams

To complete this flow, we'll need to configure or obtain access to both Twitter and Office 365 identities, as well as a Microsoft Teams team where messages will be posted.

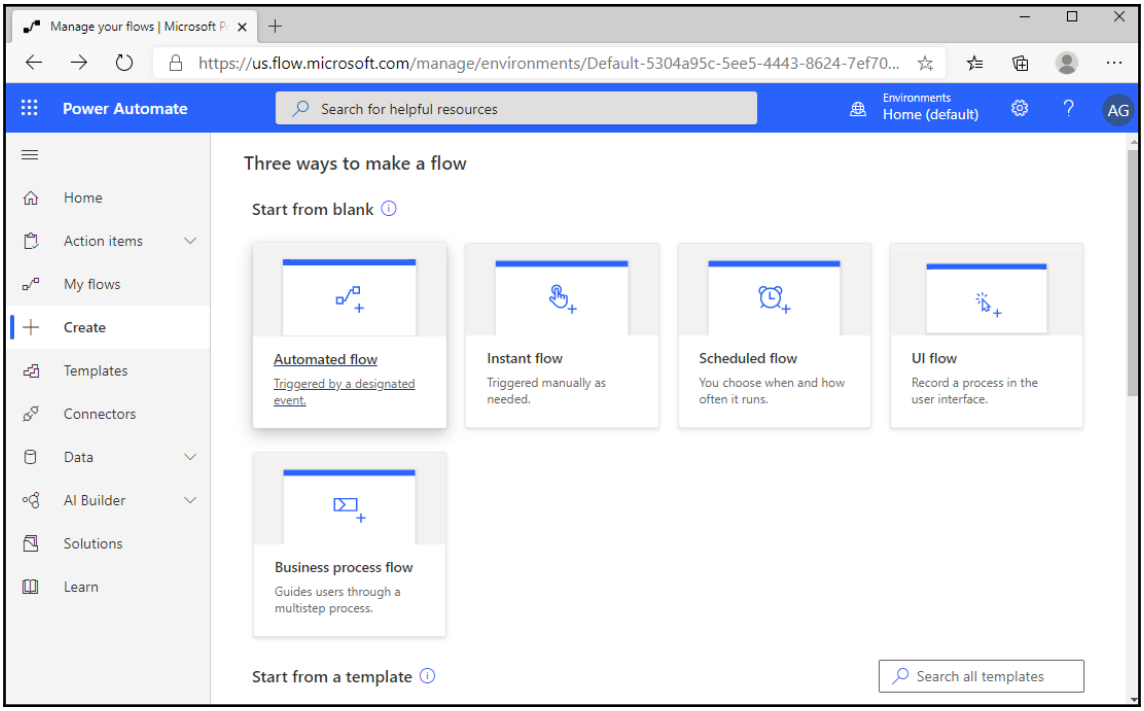
Then, we'll start creating the flow!

Creating and executing the flow

Once we have the prerequisite components for the flow (that is, the aforementioned identities and a team), we can begin the configuration. To build this sample flow, follow these steps:

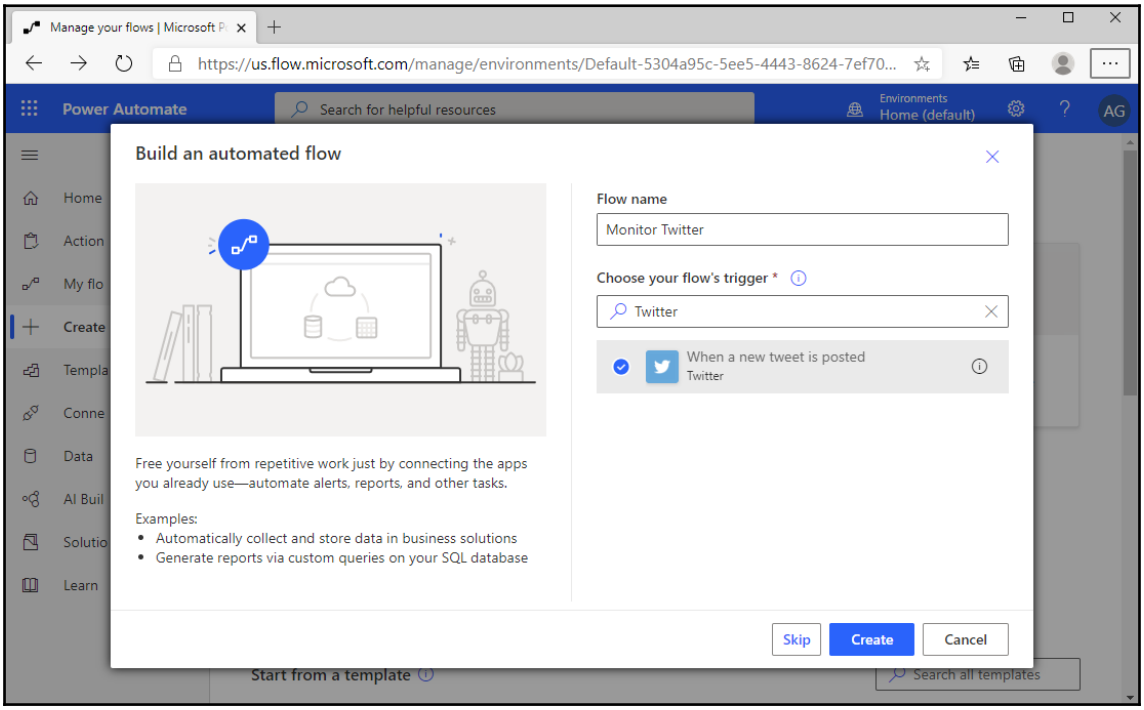
1. Log in to the Power Automate web portal interface (`https://flow.microsoft.com`).
2. On the left-hand side of the page, click **+ Create**.

3. On the **Manage your flows** page, under **Start from blank**, click **Automated flow**:

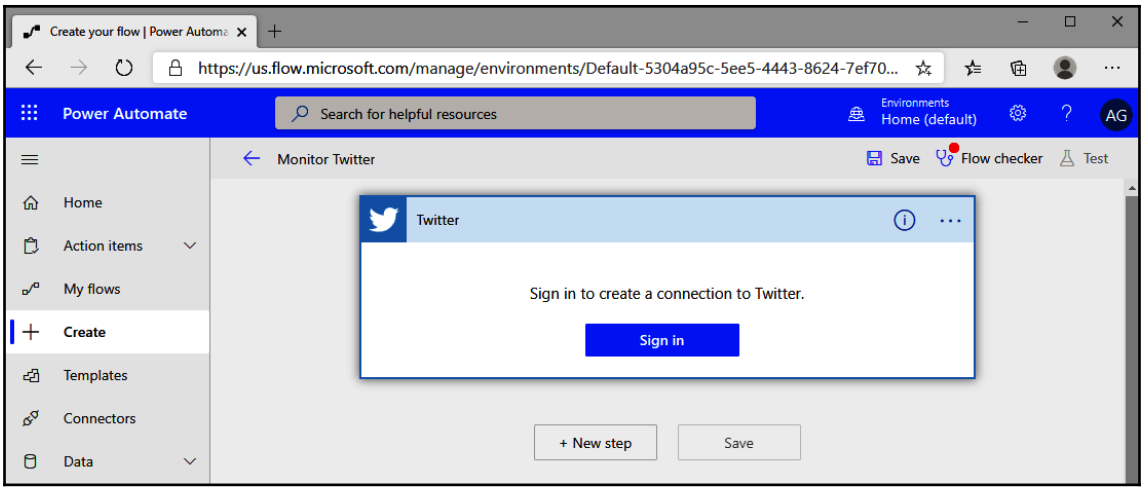


4. Enter a name for the flow.

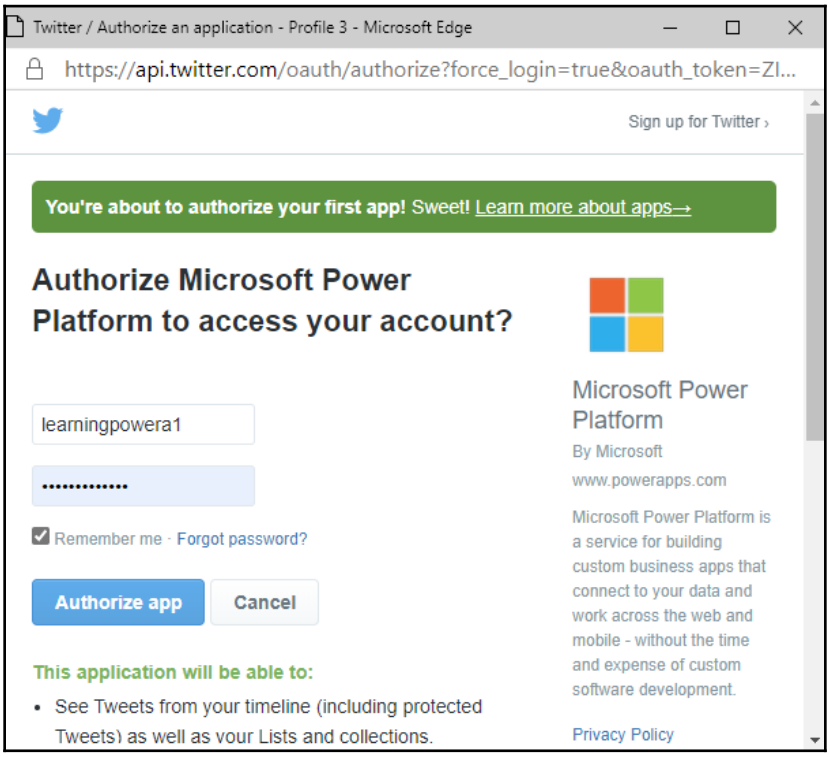
5. In the **Choose your flow's trigger** box, start typing `Twitter` to filter triggers related to Twitter. Select **When a new tweet is posted** and click **Create**:



6. Next, click **Sign in** to provide the necessary Twitter credentials to access the Twitter API:

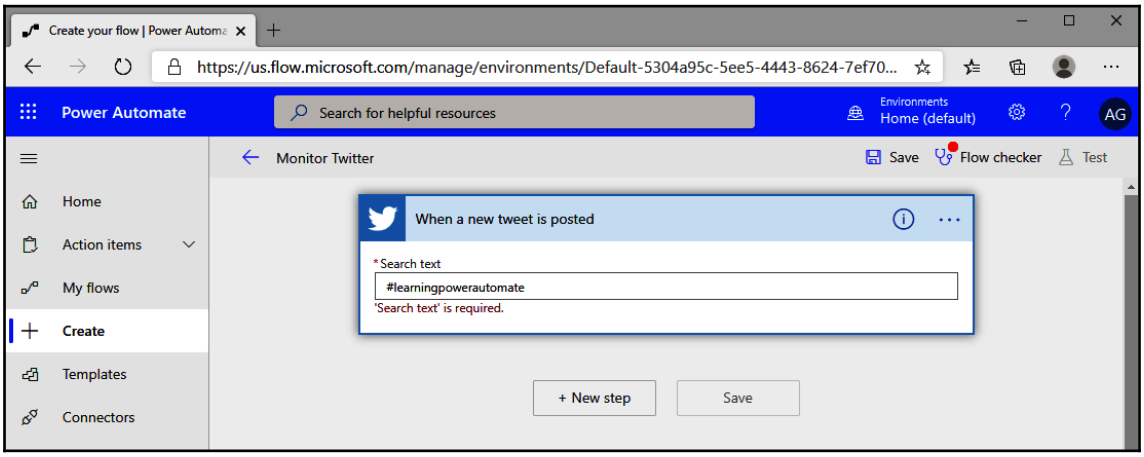


7. Enter the credentials of a Twitter account and then select **Authorize app**:

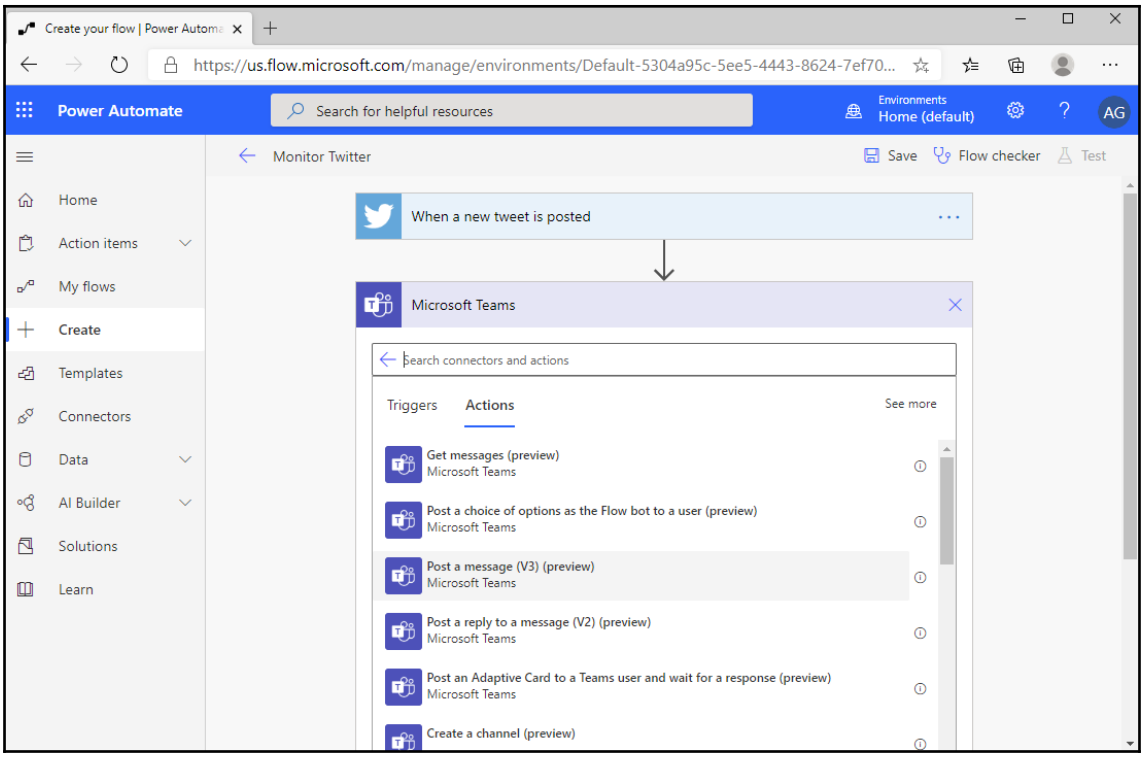


8. In the **Search text** box, enter the text that you want to search for. In this case, the flow will be configured to monitor new instances of `#learningpowerautomate`:

All Twitter interactions with the hashtag will be queried, not just tweets posted by the account whose credentials were provided.

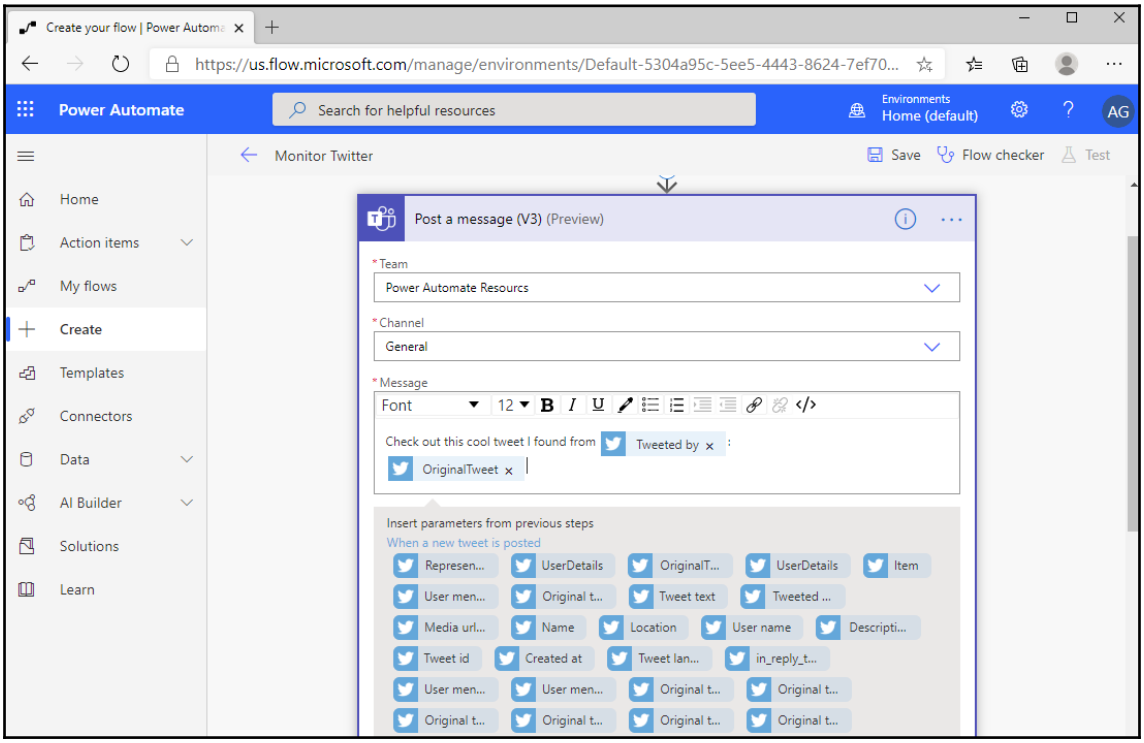


9. Click **+ New step**.
10. Under **Actions**, search for the Microsoft Teams **Post a message** action and select it:



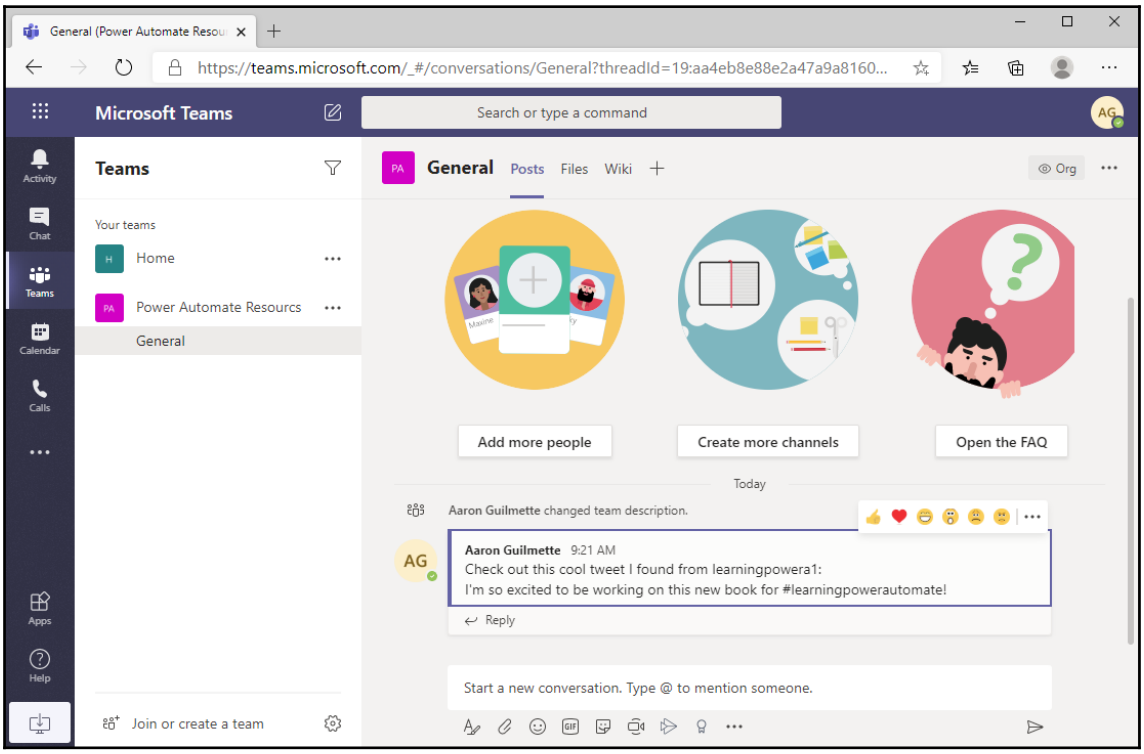
11. Select a **Team** and a **Channel** where you have access to post messages.

12. Populate the **Message** field. Many flows will give you access to *dynamic content* (data that is automatically populated, calculated, or retrieved) based on the context of the flow's connected data sources and actions. In this example, the dynamic content values, **Tweeted by** and **OriginalTweet**, have been selected to surface data provided by Twitter:

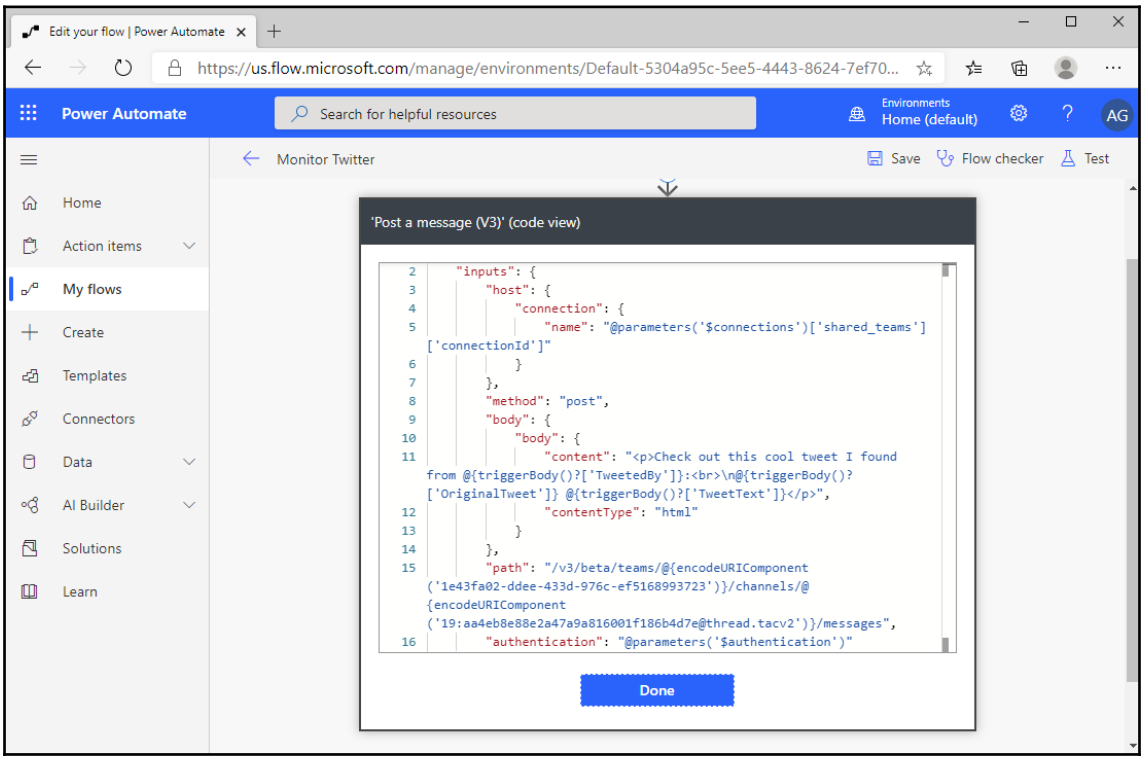


13. Scroll to the bottom of the page and select **Save**.

Now that the flow has been created and saved, you can test it by generating content and then checking to see whether the content gets posted to the Microsoft Teams channel as you'd expect:



At any time, you can use a feature called **Peek code** that allows you to look at the underlying JSON code that controls how a step functions. To access the JSON, click the ellipsis (**...**) for a step and then select **Peek code**:



You can't directly edit them in the Power Automate user interface, but you can export it and edit it in an editor such as Visual Studio.

This flow built on your previous knowledge of flow components by adding dynamic content. We'll get into more dynamic content concepts in later chapters, but as you can see, being able to use dynamic content placeholders and references will increase the richness of the flows you create.

Summary

In this chapter, you learned about the various Microsoft Power Automate interfaces (web, mobile, and administration) and some of the features available in each of them. Using the knowledge of Power Automate and the workflow concepts introduced in `Chapter 1`, *Introducing Power Automate*, we built a simple flow to generate a Teams channel message based on Twitter keywords.

Our simple flow illustrated the power of automation to help organizations stay connected and generate actionable information in a relatable manner.

In the next chapter, we're going to begin using Power Automate to interact with email.

3
Working with Email

Over the years, many business processes have been developed that use email as a storage, tracking, or processing mechanism. These processes have relied upon a combination of manual activities, third-party plug-ins, **component object model** (**COM**) add-ins, and **Visual Basic for Applications** (**VBA**) scripting. As you learned in `Chapter 1`, *Introducing Power Automate*, connectors are used to attach to data sources and endpoints (either to receive, retrieve, store, or send data). Connectors are structured configuration files (typically formatted as **JavaScript Object Notation** or **JSON**) that define how Power Automate will interact with another service for automation tasks.

So what makes Power Automate different from previous automation technologies?

A Power Automate-based solution strength lies in the cloud, as it doesn't rely on any desktop or local computer components to be installed or running, isn't dependent on particular versions of an Outlook or other desktop client, and can be built in a codeless (or near codeless) manner using an extensible, componentized architecture.

This chapter will focus on using Power Automate to manage email:

- Learning about email connectors and actions
- Working with email

By the end of the chapter, you should be able to connect Power Automate to an email account and perform common tasks such as saving and sending messages.

Learning about email connectors and actions

One of the most common scenarios for end user automation is email. Microsoft Power Automate can connect natively to different types of email systems, such as the following:

- Office 365
- Outlook.com
- Gmail

Additionally, there are third-party plug-ins such as MailParser that further enhance the ability of Microsoft Power Automate to process messages.

When processing a mailbox, there are several types of actions available, depending on the data source. Popular actions include the following:

- Creating or deleting a calendar event
- Creating or deleting a contact
- Flagging an email (such as the Importance flag)
- Forwarding and replying to an email or sending a new one
- Marking an email as read

As you saw in the previous chapter, you were able to access dynamic content—properties, metadata, and content related to a unique item. When manipulating mailbox content, you have many types of objects available to you as parameters. For example, when retrieving messages using the **Get Emails (V3)** action (`https://docs.microsoft.com/en-us/connectors/office365/#get-emails-(v3)`), you can then work with these parameters (among others) of a message:

- **Folder** (folder containing a message)
- **To** (To recipients of a message)
- **CC** (CC recipients of a message)
- **From** (sender of a message)
- **Importance** (message importance)
- **Subject** (message subject)
- **Search Query** (message body filtering)

By working with those parameters or fields, you can create flows with customized or personalized content, as well as evaluate the parameters to ensure flow only executes when certain conditional constraints are met.

For a complete list of the hundreds of connectors and actions available, see `https://docs.microsoft.com/en-us/connectors/`.

In the next section, we're going to start exploring some of the possibilities of the Office 365 email connector (though many of the functions and concepts will be applicable when working with Outlook or Gmail connectors).

Working with email

The challenge of automating email has been around almost as long as email itself. As we mentioned at the beginning of the chapter, many solutions have been developed to address this over the years using technologies such as VBA scripting, Outlook rules, and COM add-ins. These solutions have required both programming skills to develop and a dependence on a desktop computer always running to be able to execute the commands.

With Power Automate and Microsoft 365, Outlook.com, or Gmail services, the necessary components are "always on," so you don't need to worry about power outages, software updates, or network connectivity issues disrupting a business process. In this section, we're going to look at options for handling incoming emails, processing attachments, and sending messages.

Receiving email

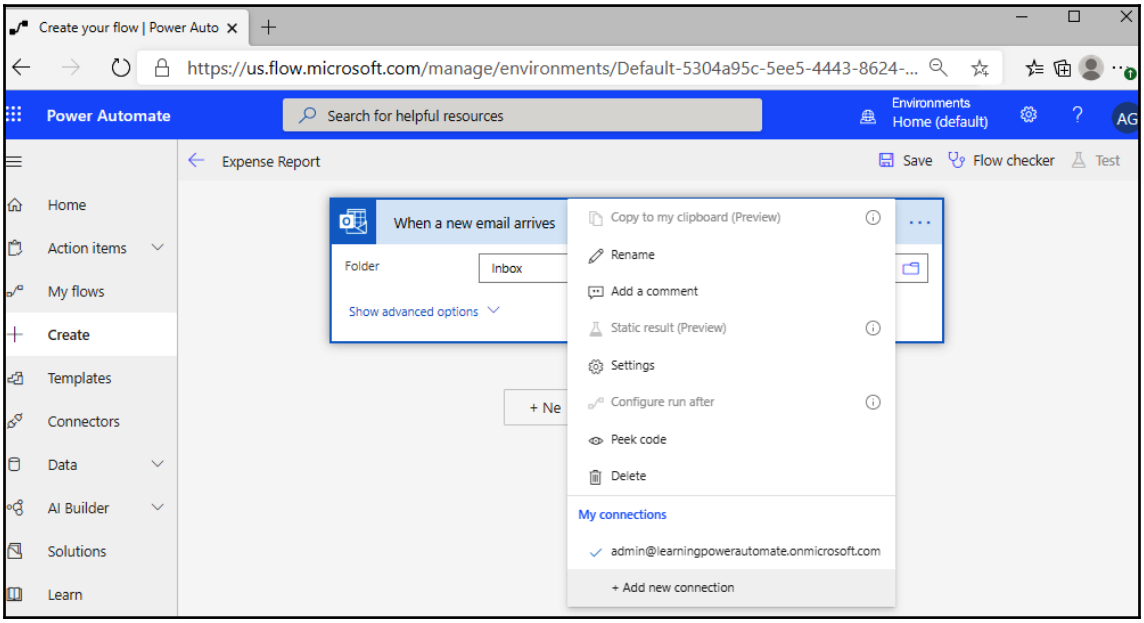
Many organizations use the receiving of an email (typically, to some sort of shared mailbox) as a sort of trigger to start a business process. This could be a sales order, a service ticket request, or a request for some sort of information. In this example, we're going to look at some of the filtering and selection options available when processing an incoming message.

To work with this flow, we're going to need an Exchange Online mailbox and a Microsoft Power Automate subscription. For this scenario, we're just going to look at messages we receive that could indicate they are related to an expense report. Follow these steps:

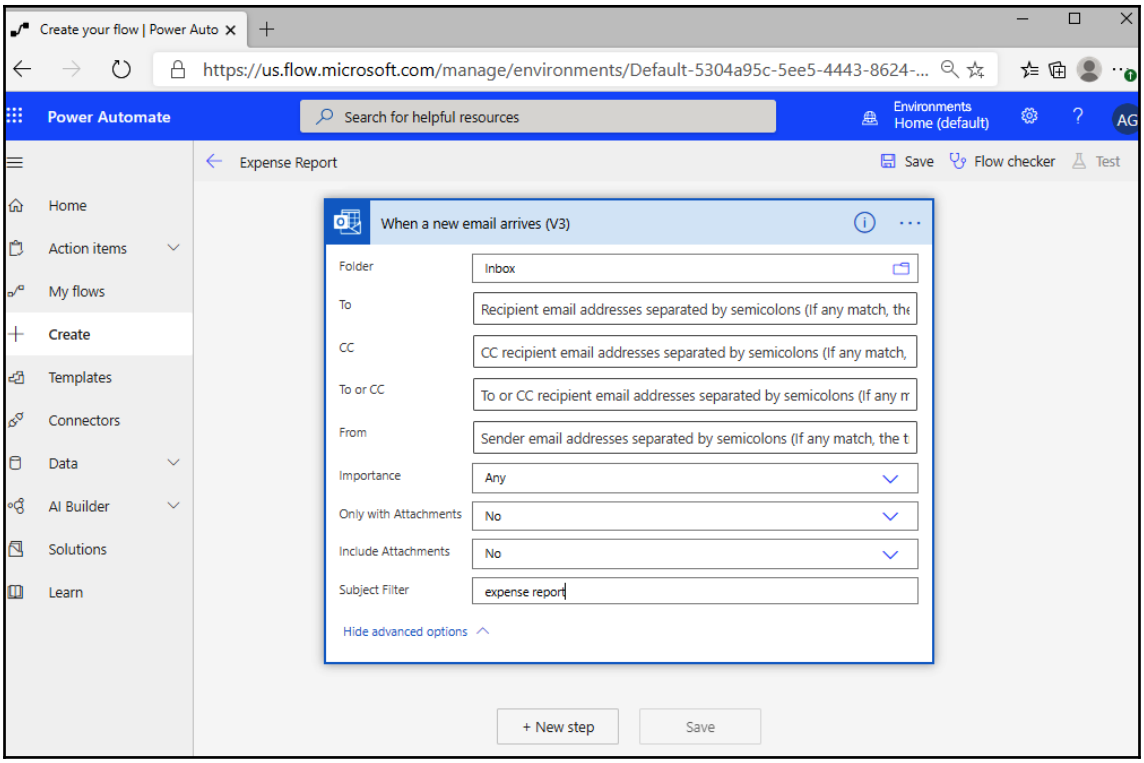
1. The easiest way to start is to log into the Power Automate web portal (`https://flow.microsoft.com`), click **+ Create**, and then select **Automated flow**.
2. Enter a value for **Flow name**, and then search the triggers for **When a new email arrives (V3)**. There are also triggers for **When a new email arrives in a shared mailbox (V2)** *and* **When a new email arrives to a group**. Any of them can be used, but you'll need to ensure the account you use as a credential has access to the shared mailbox or is a member of the group being monitored.

3. Click **Create**.

4. By default, the flow is going to connect to the Outlook mailbox for the currently logged in account. If you need to monitor a different mailbox, select the ellipsis (**...**) icon on the **When a new email arrives (V3)** step, and then click **+ Add new connection** to provide the necessary mailbox name and credentials:



5. Click **Show advanced options**.
6. In the **Subject Filter** box, enter a value such as `expense report`:



The filtering options at this level are basic, but give you options to specify sender or recipient values that you can use as conditions to proceed with your flow. From here, you can add additional processing steps.

When you were creating the flow, you probably also noticed other email-based triggers that looked similar:

- When a new email mentioning me arrives (V2)
- When a new email mentioning me arrives (V3)

The triggers for "new email mentioning me arrives" use a special Microsoft Graph API call to check for messages with an @mention (at mention) for your identity.

We'll take a look at some additional conditional filtering concepts in `Chapter 8`, *Working with Conditions*.

Handling attachments

The ability to save and manipulate attachments is a popular automating requirement. This type of request is frequently seen with mailboxes used to receive things such as purchase orders or expense reports. You can use Power Automate to save those items to OneDrive or a SharePoint Document Library.

In this example, we're going to put together a sample flow that does the following:

- Monitors a shared mailbox for messages that contain the words "expense report"
- Saves the attachment to a SharePoint Library called ExpenseReports
- Marks the email as Read
- Marks the email as Complete

To prepare for this example, you'll need the following:

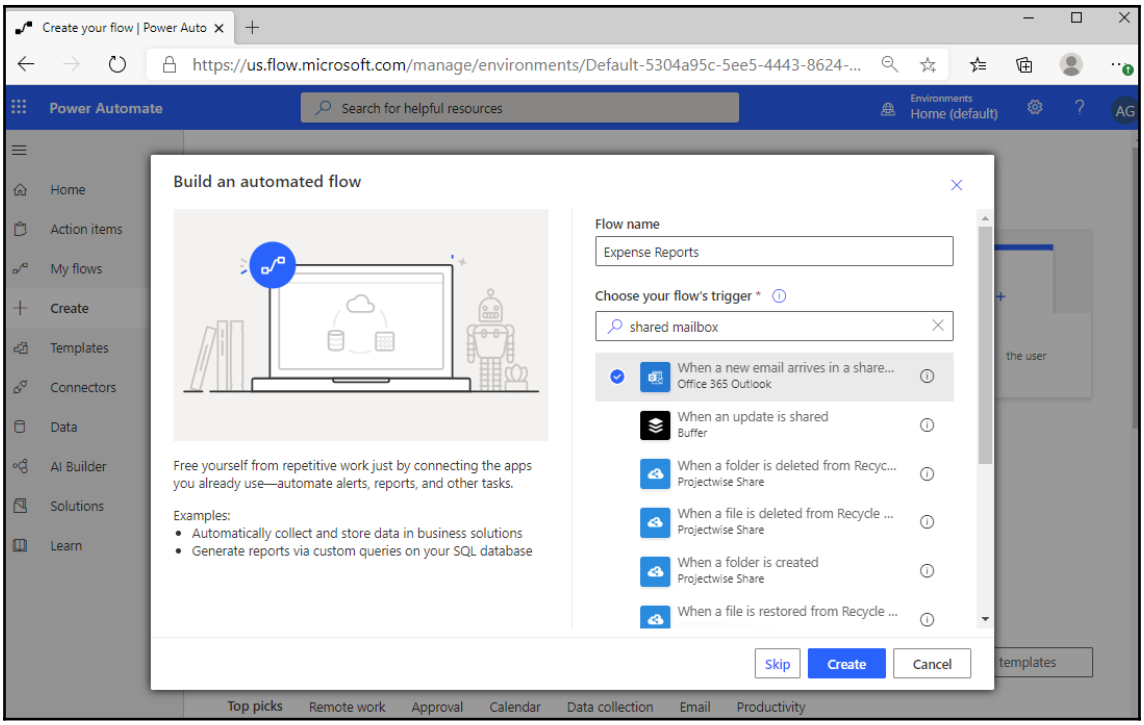
- A SharePoint site to which you can save documents
- A shared mailbox to which you've been granted management permissions

With those requirements met, you can begin creating the flow.

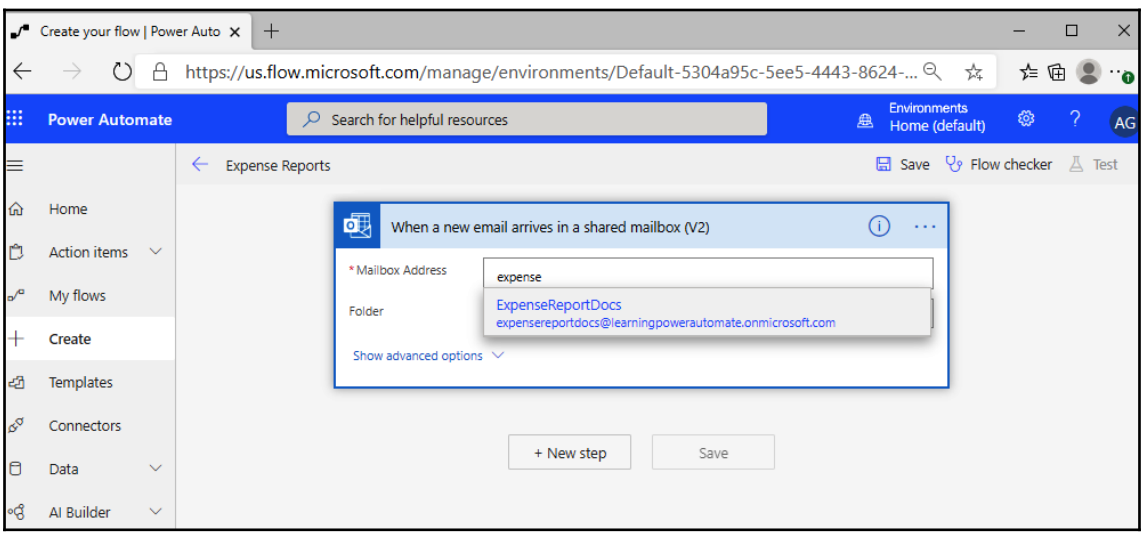
Creating the flow

In order to configure this example, follow these steps:

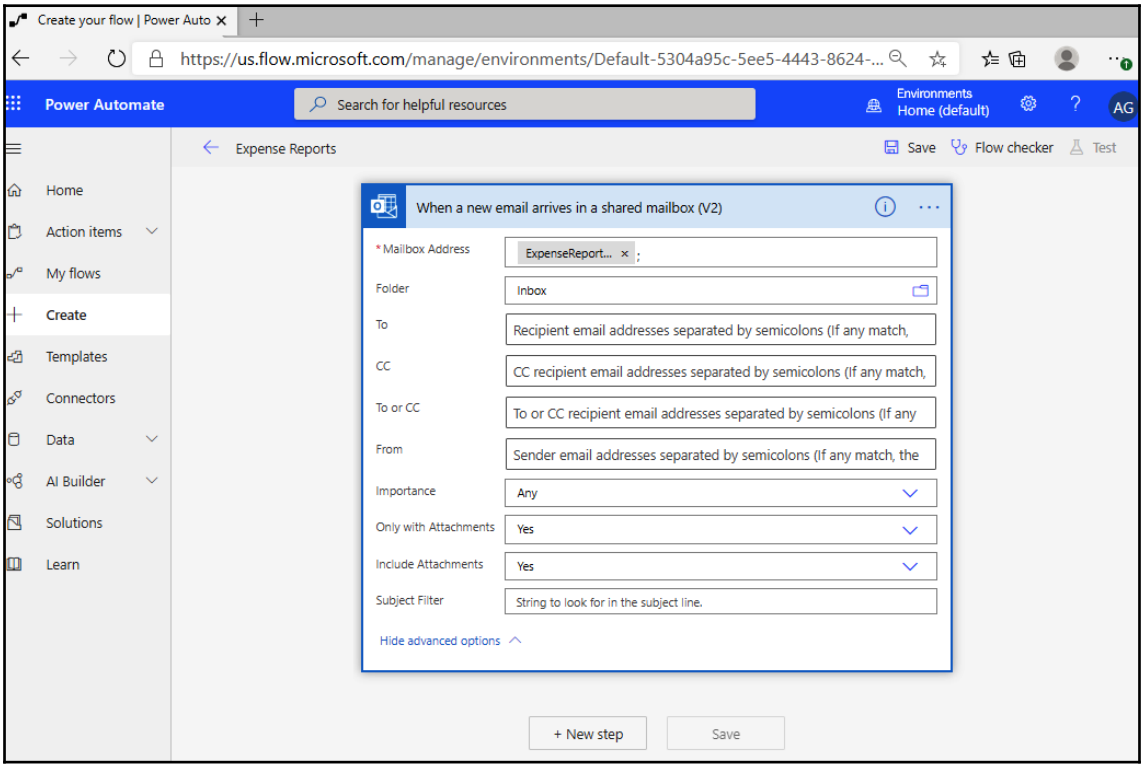
1. Log into the Power Automate web portal (`https://flow.microsoft.com`), click **+ Create**, and then select **Automated flow**.
2. Enter a value for **Flow name**, and then search the triggers for **When a new email arrives in a shared mailbox (V2)**:



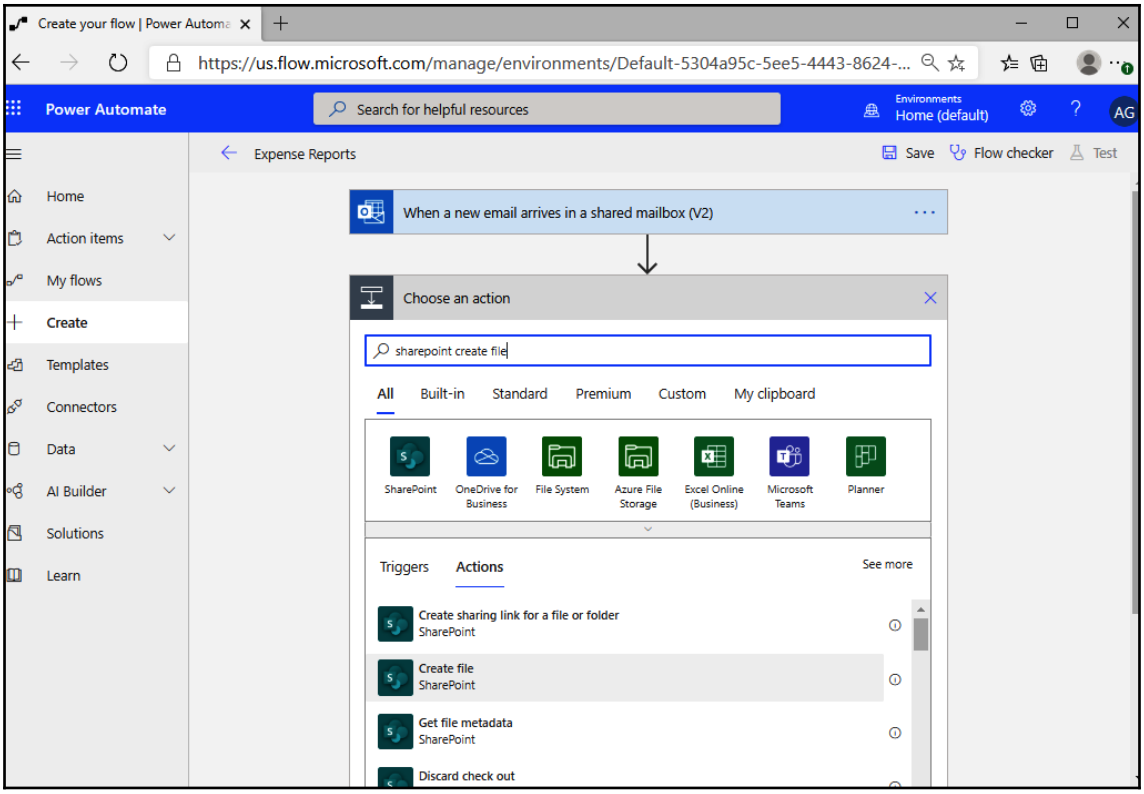
3. In the **Mailbox Address** box, enter the address of the shared mailbox you wish to monitor:



4. Click **Show advanced options** to expand the options.
5. In the **Only with Attachments** box, select **Yes**. In the **Include Attachments** box, select **Yes**. These two options will configure the flow to only trigger if the message has an attachment and will make the attachment available for the next step:

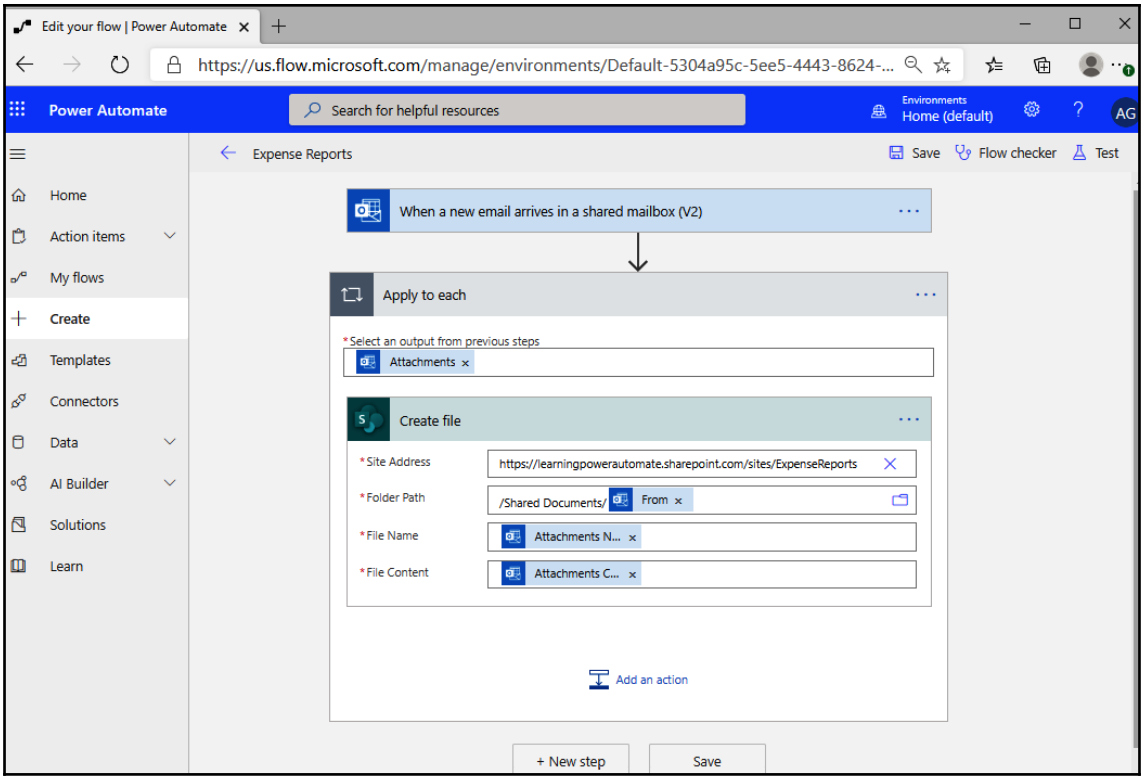


6. In the **Subject Filter** box, enter `expense report`. This will place a further constraint on the rule to only work if all three of the conditions are met.
7. Click **+ New step**.
8. Search for `Sharepoint Create file` and select the **Create file** action for SharePoint:



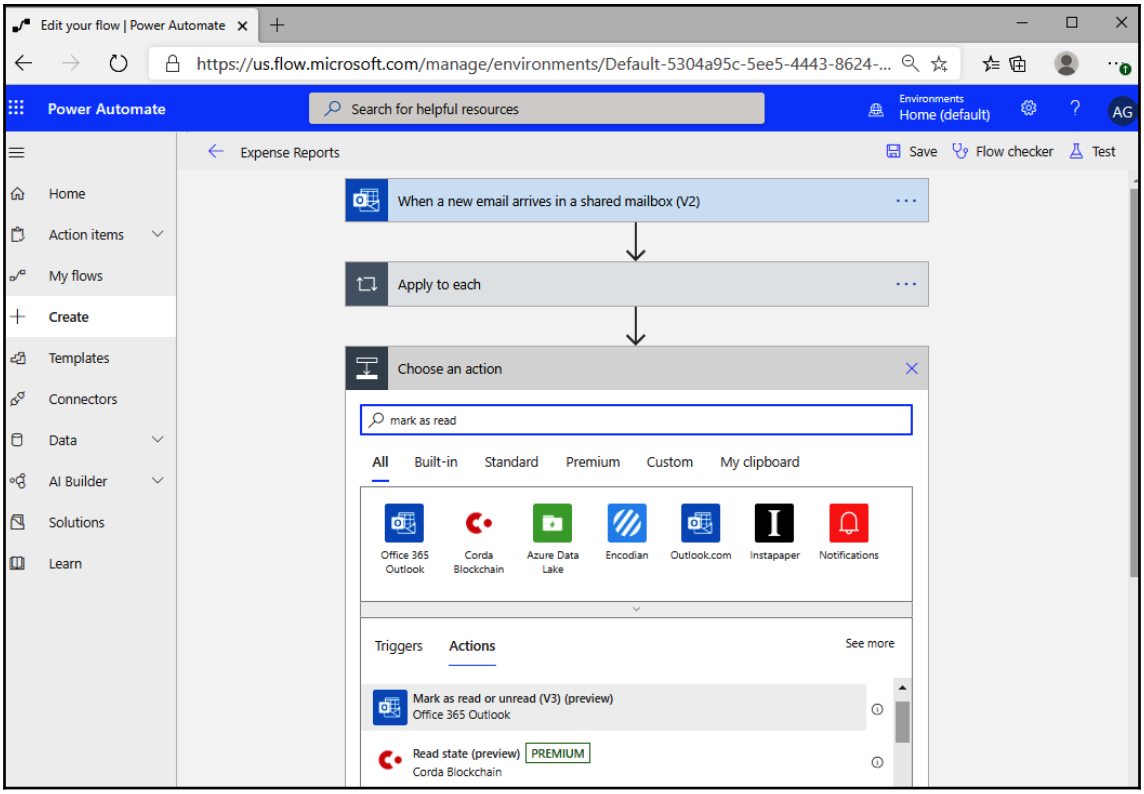
9. In the **Site Address** box, add the URL for the SharePoint site that will store the document.
10. In the **Folder Path** box, click the folder icon to browse for a document library folder such as **Shared Documents**. In this example, we added a trailing forward-slash (/) and then used the dynamic content variable **From** to create a subfolder for each email sender.
11. In the **File Name** box, select the dynamic content object **Attachments Name**.

12. In the **File Content** box, select the dynamic content object **Attachments Contents**:

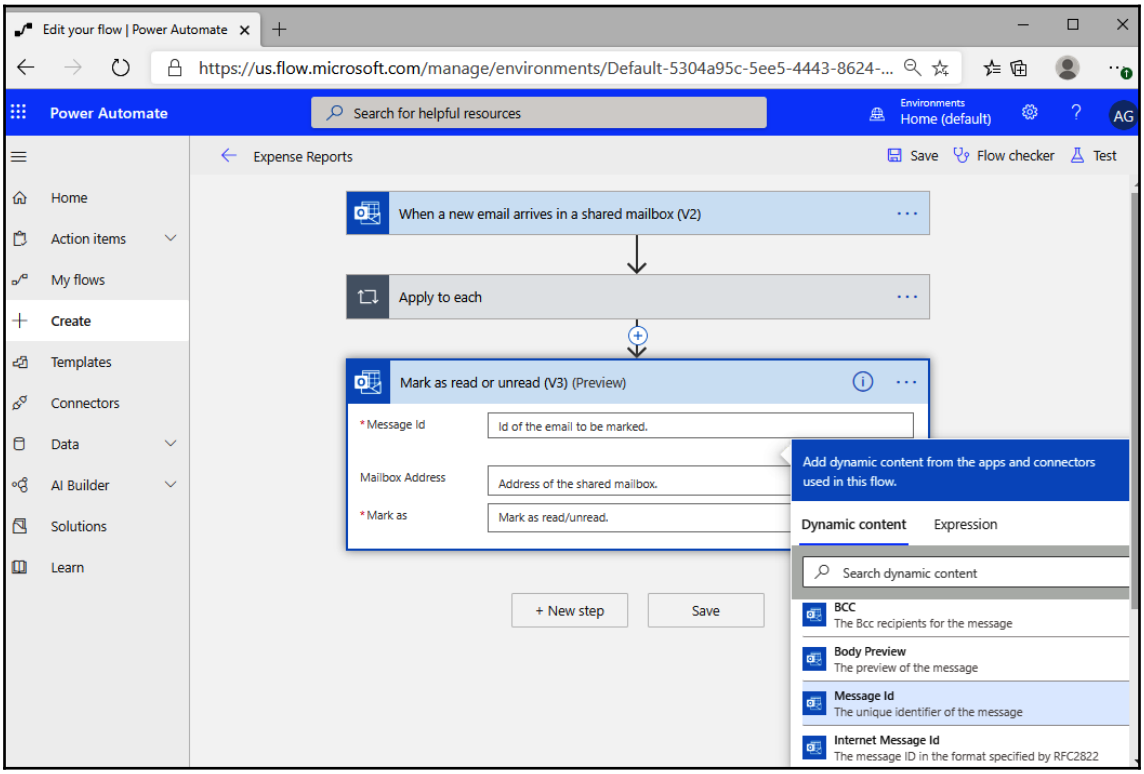


13. Since Power Automate knows that an email message may have one or more items attached, it will wrap this action in an **Apply to each** function that will save each attachment separately.
14. Click **+ New step**.

15. Locate **Mark as read or unread (V3) (preview)** and select it:

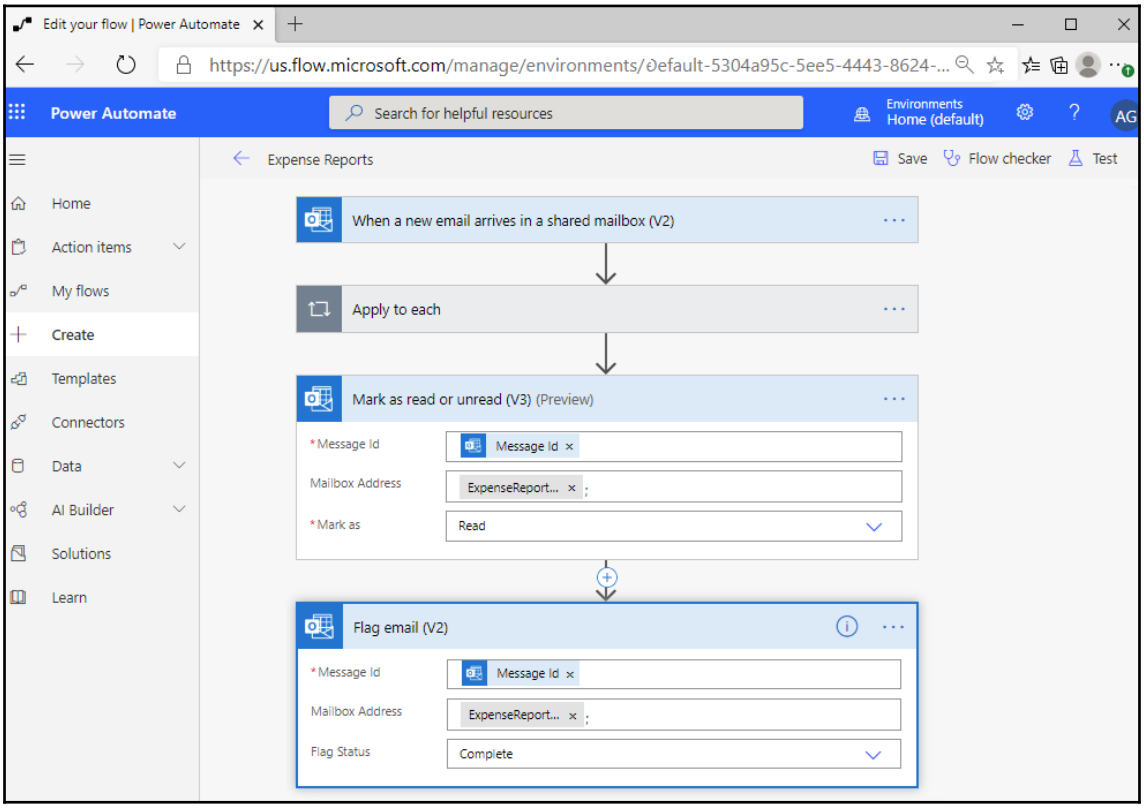


16. In this step, select the **Message Id** box, and then browse to the dynamic content object **Message Id**. This will select the message that we're currently processing:



17. In the **Mailbox Address** box, locate the shared mailbox.
18. In the **Mark as** box, select **Read**. We want to mark this so that in the future if someone reviews the shared mailbox, they'll know this item has been read. However, this may not be enough, given that multiple people may access the mailbox and someone may accidentally change the status by just clicking on the message.
19. Add another step. Search for **Flag email (V2)**.
20. In the **Message ID** box, select the **Message Id** dynamic content object.
21. In the **Mailbox Address** box, locate the shared mailbox.

22. In the **Flag Status**, select **Complete**. This will help verify that the message has indeed been processed:



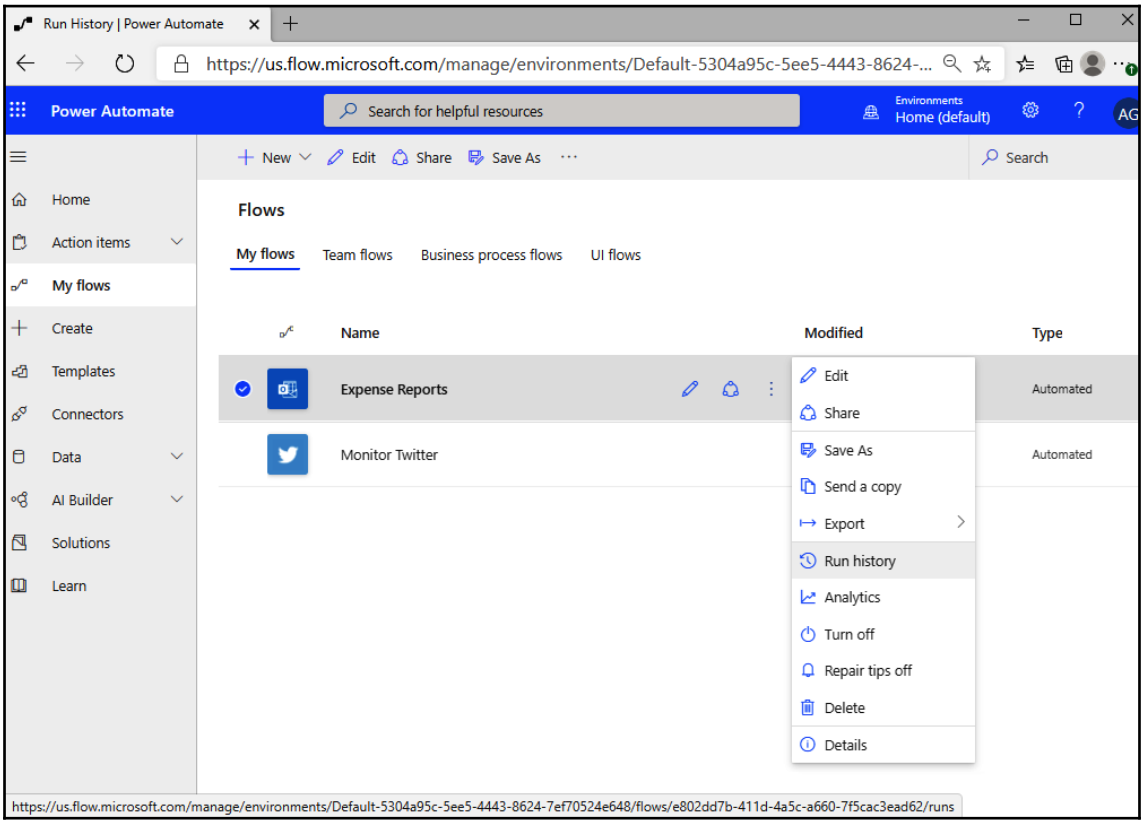
23. Click **Save**.

Time to test it out! Send a message to the shared mailbox meeting the requirements (an attachment and the subject of **expense report**). After you've submitted an email message, verify that the flow has completed successfully.

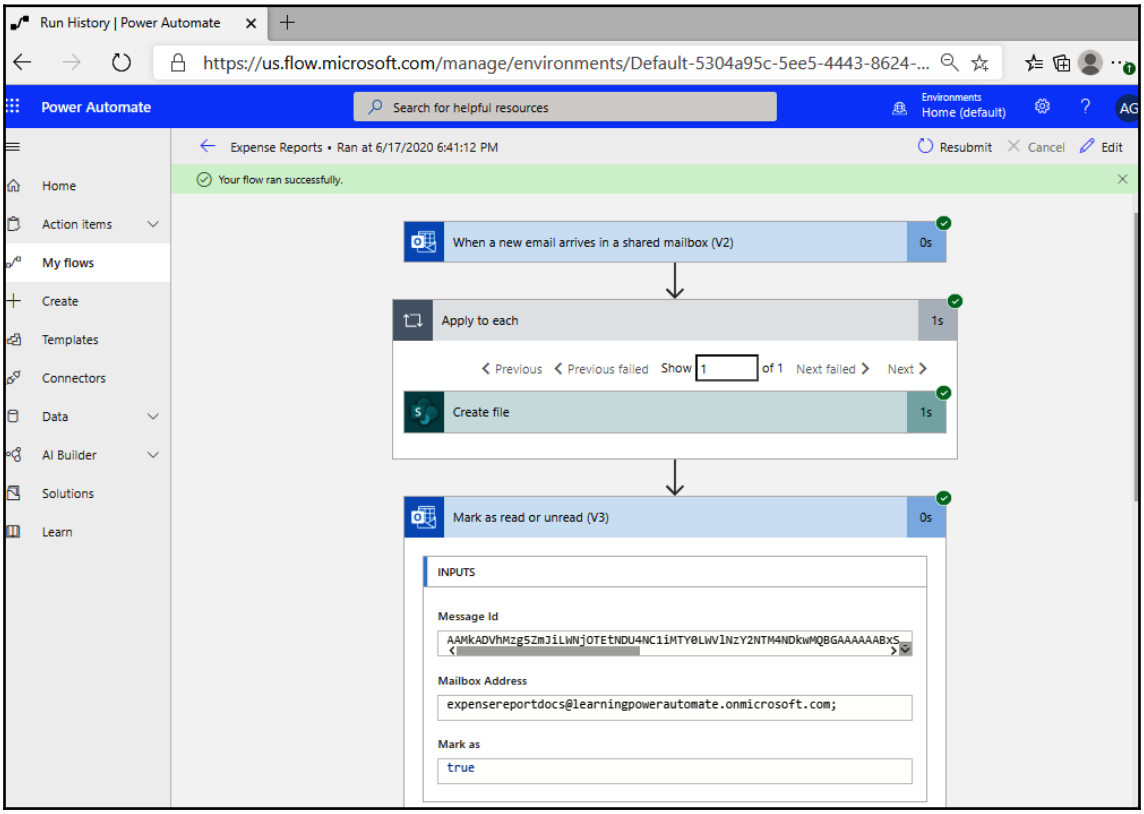
Verifying completion

To ensure that things have completed successfully, you'll want to review the output and verify it. You can use these steps to verify this particular flow:

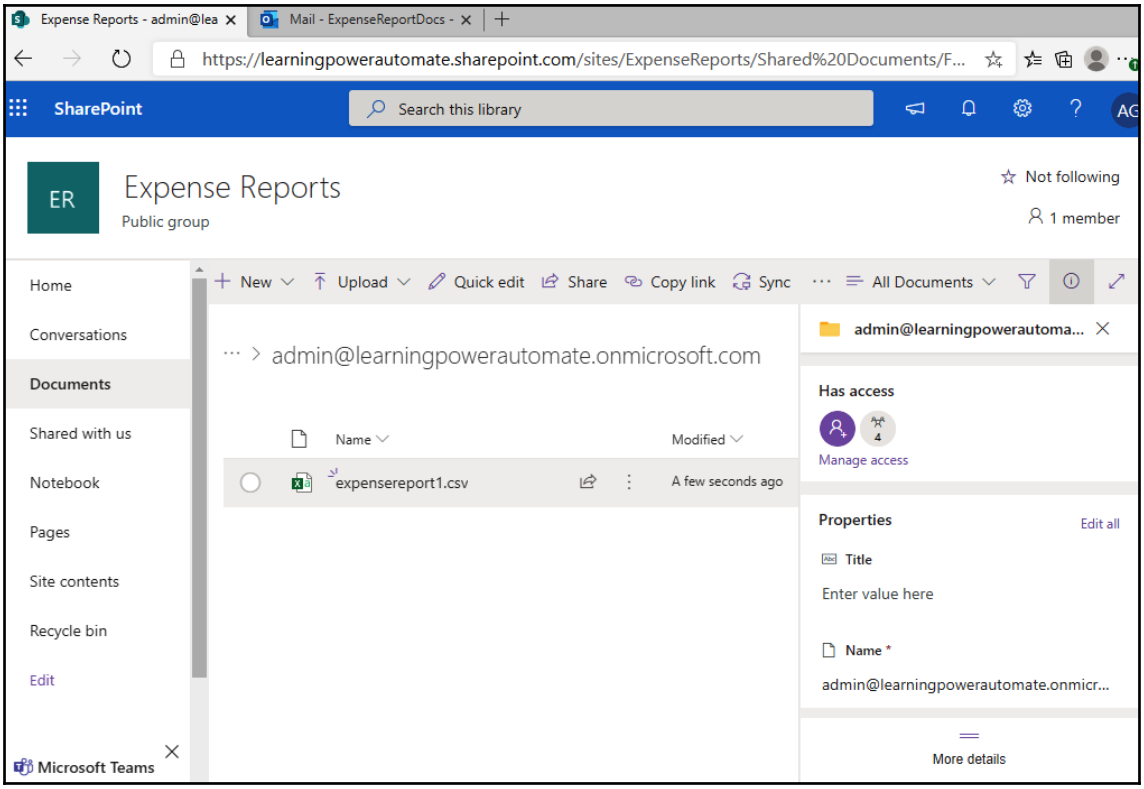
1. From the Power Automate web portal, you can select **My flows**, and then select the ellipsis next to the flow and click **Run history**:



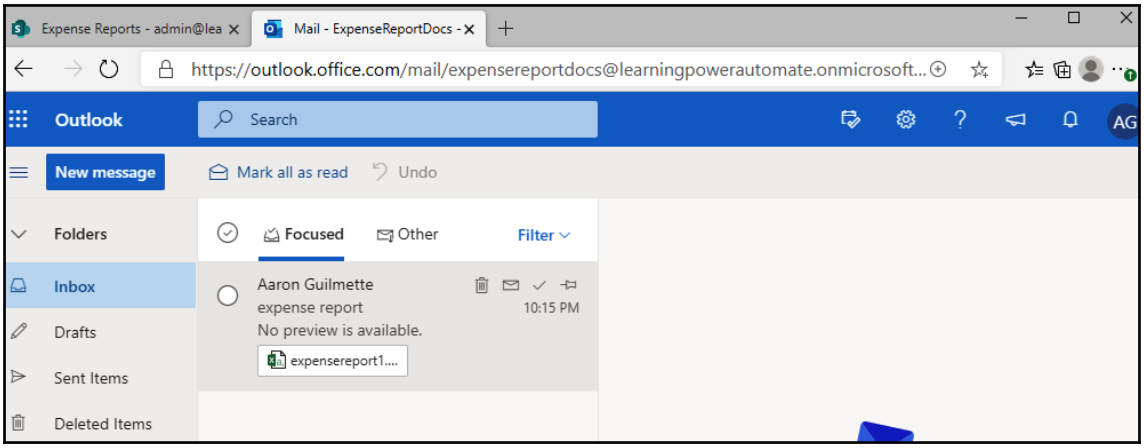
2. Select the run, and then look at the history. You can click on each step to expand and verify the values that Power Automate used during processing:



3. To check the connected SharePoint site, navigate to the SharePoint site containing the document library that you specified when creating the flow. Verify that the folder structure and file exists:



4. Next, you can open the shared mailbox that the flow was configured to use. Look for the message and verify that it has been marked as read *and* the flag column shows a checkmark:



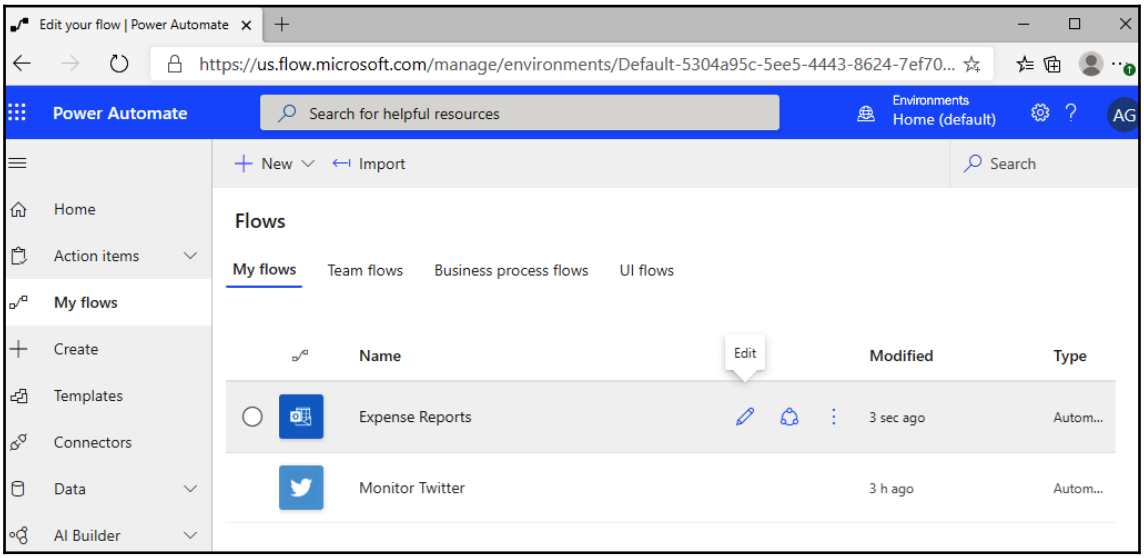
At this point, you can be confident that your attachment handling flow is operating correctly. Next, we'll finish up the configuration by sending a confirmation email.

Sending email

Since sending things to a shared mailbox may seem like a bit of a black box, many organizations choose to send some sort of a confirmation email to let the sender know that the message has been received and processed. Sending an email is also a common task for Power Automate, and can be used in a number of circumstances, such as generating an email-based alert based on certain conditions or responding to a sender, like we're going to do in this example. The process is the same regardless.

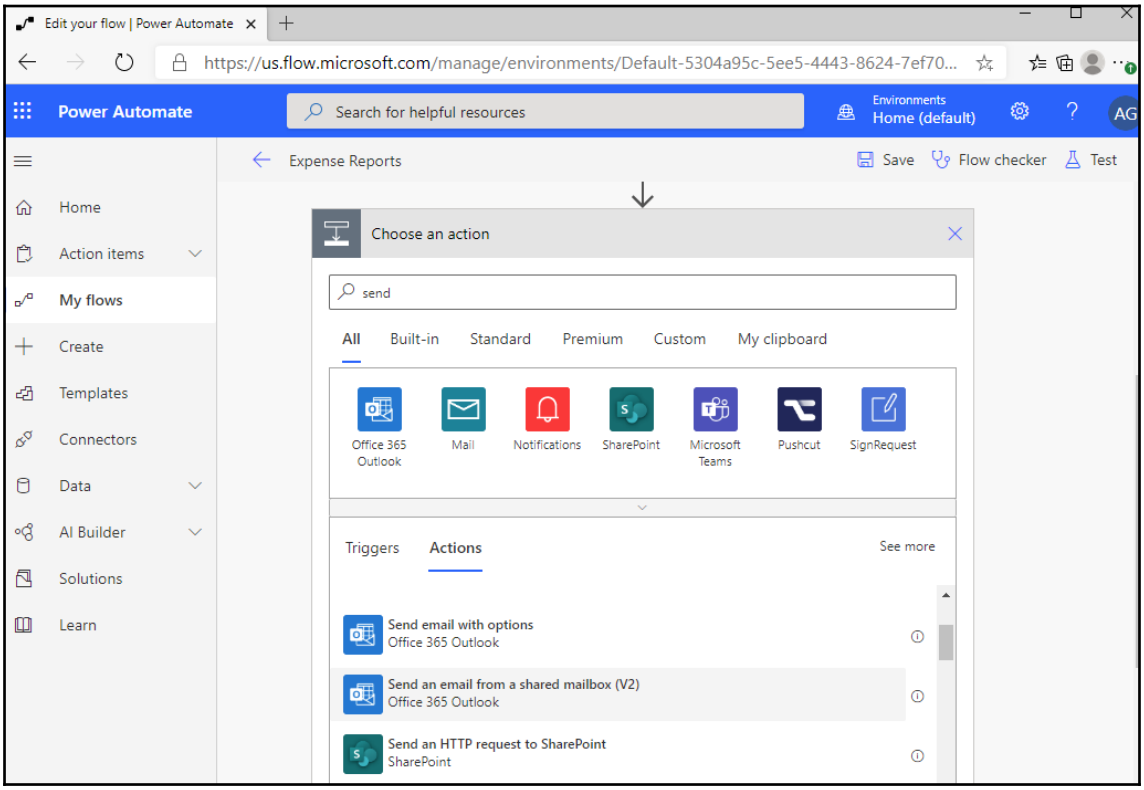
For this task, we're going to re-use the Expense Report attachment flow and reply with a message that their file was received:

1. Navigate to the Power Automate web interface (`https://flow.microsoft.com`) and select **My flows**.
2. Select the Expense Report flow previously created, and then select the pencil icon to edit it:



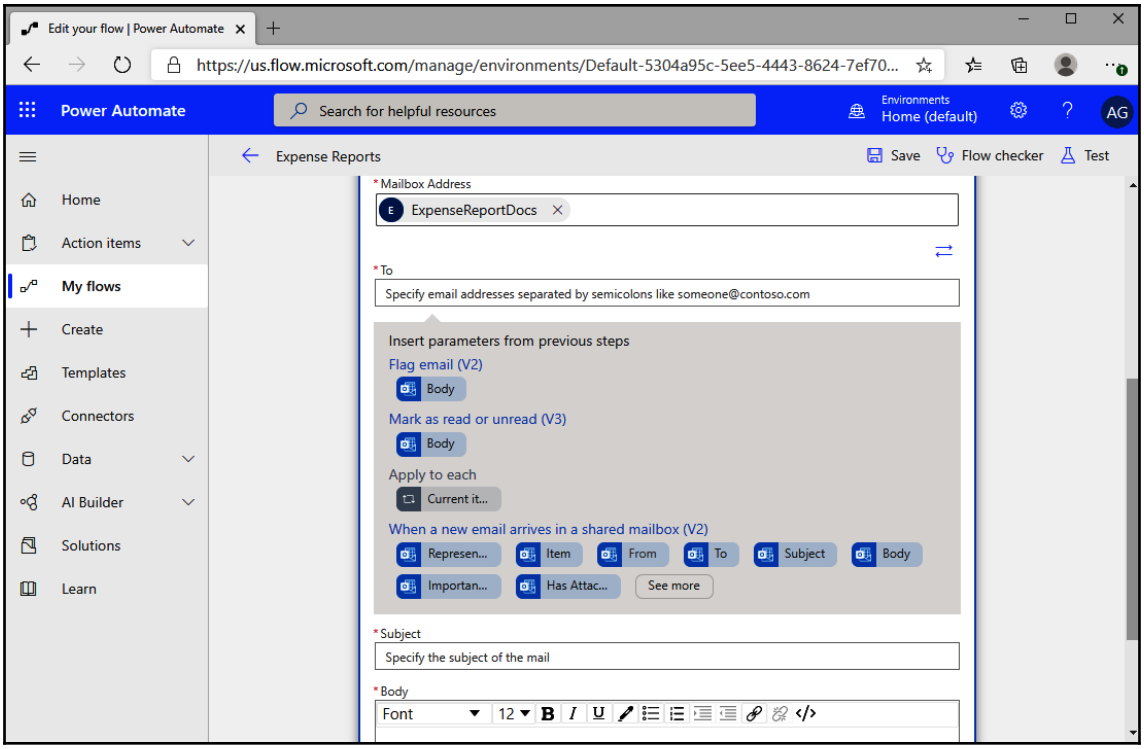
3. At the end of the flow, select **+ New step** to add a new step.

4. Since the Expense Reports flow is using a shared mailbox, select the **Send an email from a shared mailbox (V2)** action:

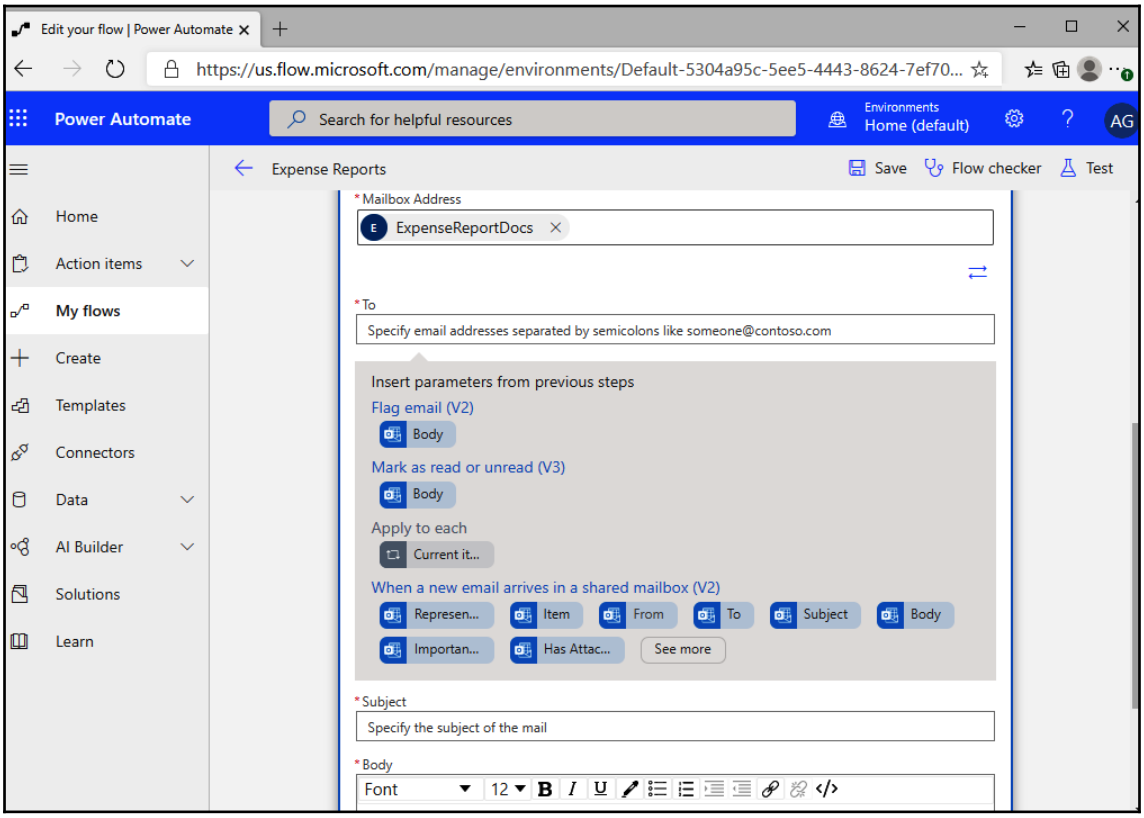


5. In the **Mailbox Address** box, enter the value for the shared mailbox that will be generating the outbound message.

6. In the **To** box, click the **Add dynamic content +** button:

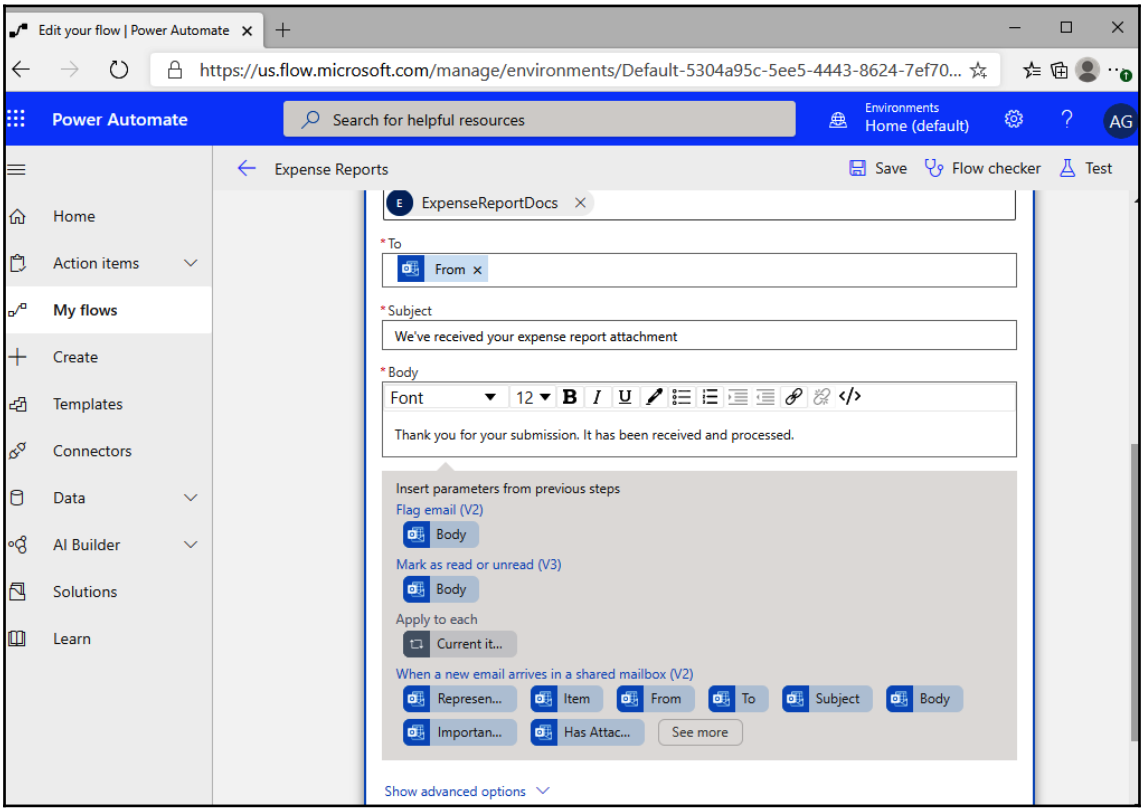


7. Under **When a new email arrives in the shared mailbox (V2)** section, select **From** to use the address of the original sender:



8. In the **Subject** field, add a value that will be used as the subject of the outgoing message.

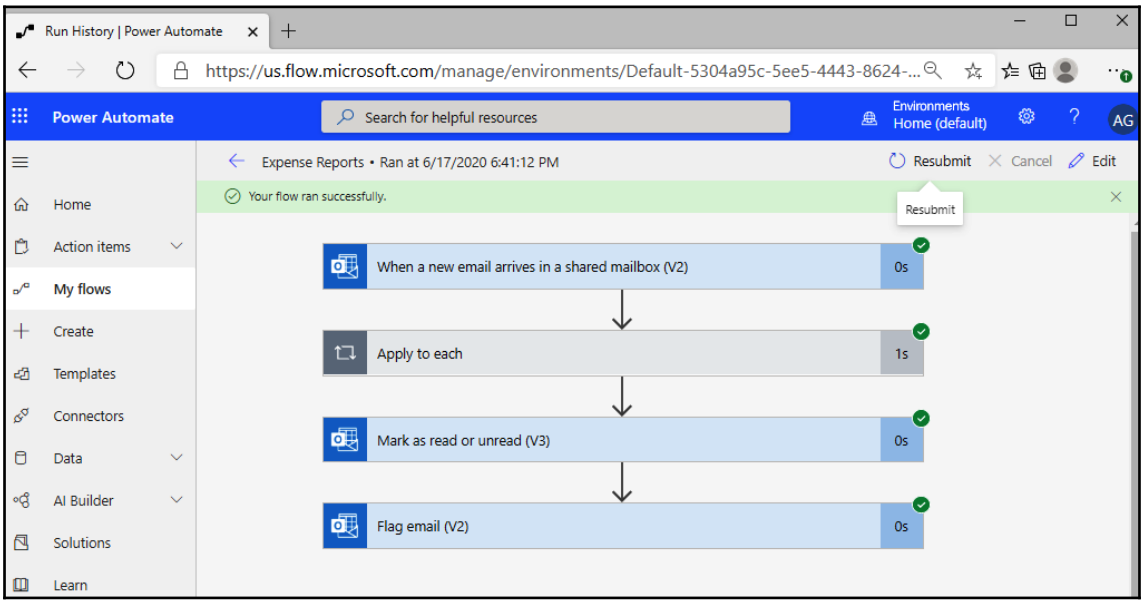
9. In the **Body** field, add any applicable text. You can use dynamic content objects here, if desired, as well as the rich text editor to change fonts or text properties:



10. If you want to set additional options such as fixed Cc or Bcc members, additional attachments, or message importance, select **Show advanced options**.
11. Scroll to the bottom of the message and click **Save**.

At this point, the flow has been modified to send back a response message upon successful receipt and processing.

When modifying or performing iterative development on a flow, you may just want to use the option to resubmit or retry the last run with its parameters. You can use the **My flows** | **<Flow>** | **Run history** option to resubmit the last job and test it:



It's important to note that the run history shows the steps for the *last completed run*. When you re-submit, it will run the flow in its current state, so any new steps added will be included.

You can check the **Sent items** of the shared mailbox or the **Inbox** of the original sender's mailbox to see the email confirmation.

Summary

This chapter built on some of the foundational concepts in previous chapters and introduced several new concepts:

- Connecting to standard and shared mailboxes
- Saving attachments to SharePoint sites
- The **Apply to each** function
- Using the **Message Id** dynamic content object to mark items as read or flag them as completed
- Sending a message as a shared mailbox

Using all of those tools together allowed you to build a simple flow that can process inbound messages and their attachments, and then respond to the sender that their message was received.

In the next chapter, we'll use Power Automate to copy files between storage locations.

4
Copying Files

In the previous chapter, you learned about a few basic concepts you can use to handle emails, such as receiving, sending, and handling attachments. Frequently, a work process requires that you copy or move files between locations in order to make data available to new groups of users or to integrate them into another business process. For example, another flow may need to interact with the email attachment that was saved or a notification may need to be sent, stating that a file has been received.

Processing files as part of a flow can mean interacting with the file itself, as well as its properties. Power Automate provides us with the functionality to manipulate a file's properties, its contents, and the object as a whole.

In this chapter, we're going to focus on some basic file-related concepts:

- Learning about file connectors and actions
- Working with files

By the end of this chapter, you should be familiar with the common file connectors and actions and be able to use them to copy data, both inside the Microsoft 365 environment and to external storage locations.

Learning about file connectors and actions

In `Chapter 3`, *Working with Email*, you saw that Power Automate can work with files as attachments to emails. Work processes that require us to automate file attachments may also benefit from automating further processing with those files. Files may need to be copied or moved, uploaded to a new service, ingested into another platform, archived, or require approvals. Learning how to reference files and their properties is an important building block to building more complex flows.

Here's a quick rundown of some of the file storage mechanisms that we can use:

- SharePoint Online
- OneDrive for Business
- OneDrive
- Azure File Storage
- Google Drive
- Box
- Dropbox

Most organizations will use Power Automate primarily within the context of SharePoint Online and OneDrive for Business. However, most organizations also have access to or use other file hosting and sharing mechanisms as part of their manual business processes when working with partners, customers, or vendors, so it's important to be able to interact with as many platforms as possible.

File connectors typically have the following types of trigger actions available, allowing you to create automated flows:

- When a file is created
- When a file is modified
- When a file's properties are modified

Similar to the email connector actions, file connectors allow you to perform the following types of actions (depending on the connector and what APIs the service has available):

- Update a file.
- Copy a file.
- Get the metadata of a file.
- Delete a file.
- List files or folders.
- Create or delete folders.
- Extract an archive file.
- Create file sharing links.
- Check in or check out a file.

Finally, the connectors allow you to manage and manipulate the content and metadata properties of the files, such as the following:

- The name or DisplayName of a file or folder
- The unique ID of a file or folder
- The path
- The LastModified date and time
- The MediaType of the file

In addition to being able to use the available metadata of the files themselves, you may be able to include dynamic properties from other parts of the flow, as well as include them as part of your file handling procedure.

Building on the concepts of dynamic content that we saw in `Chapter 3`, *Working with Email*, we're going to introduce a new concept: **expressions**. While dynamic content is used in the context of Power Automate to describe content properties derived from object metadata, expressions can be used to perform further processing in programmatic methods. If you are familiar with using Microsoft Excel formulas, then you'll notice that the concepts and formatting are similar.

Expressions are frequently used to calculate and format values. We're going to use an expression (or formula) to create custom folder names as part of working with files in the next section.

Working with files

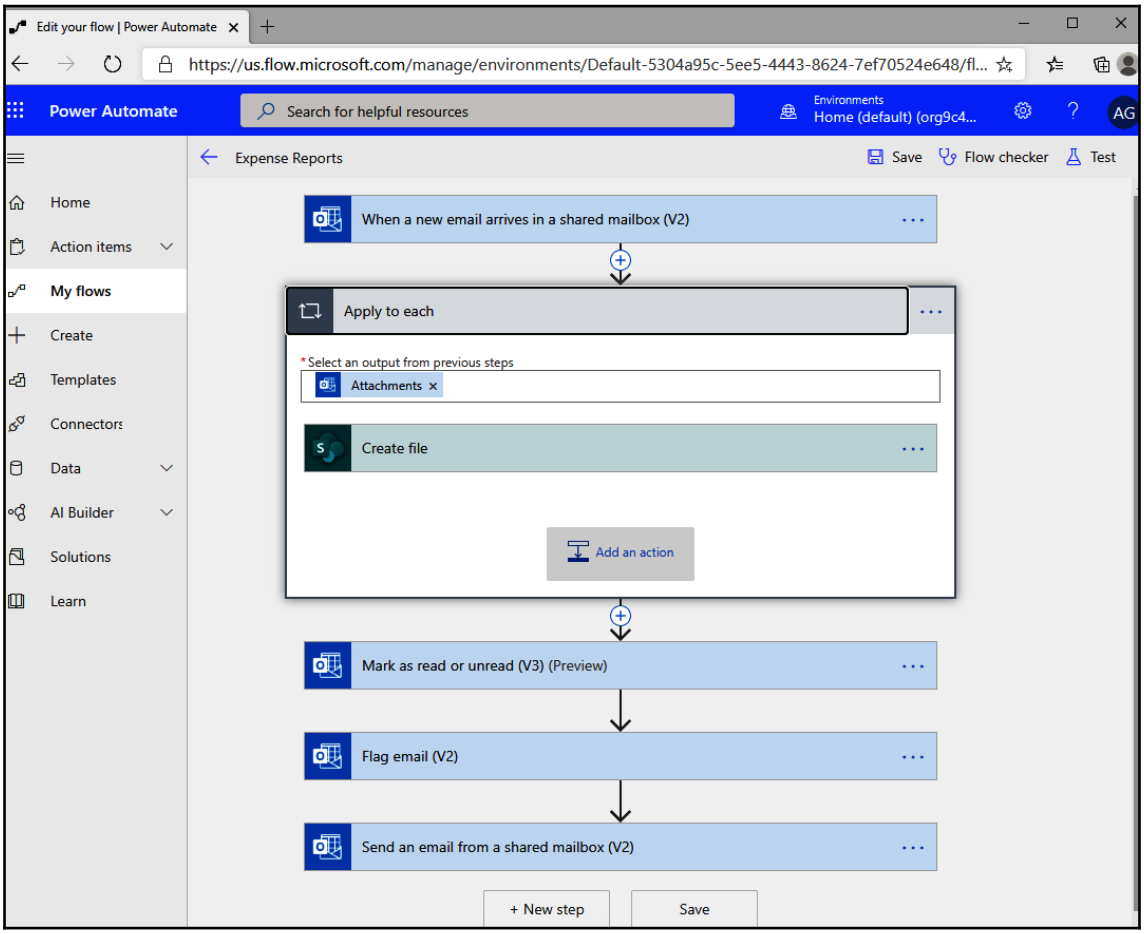
Files, from the perspective of Microsoft Power Automate, are objects that can be both the product or result of a workflow (as we saw in `Chapter 3`, *Working with Email,* when we looked at the Expense Report flow) or moved around through one. In this section, we're going to build on the Expense Report flow from `Chapter 3`, *Working with Email*, and copy the file from the Expense Report document library to another SharePoint site for archival purposes.

We'll also use Power Automate with another business scenario: publishing content to an outside storage service.

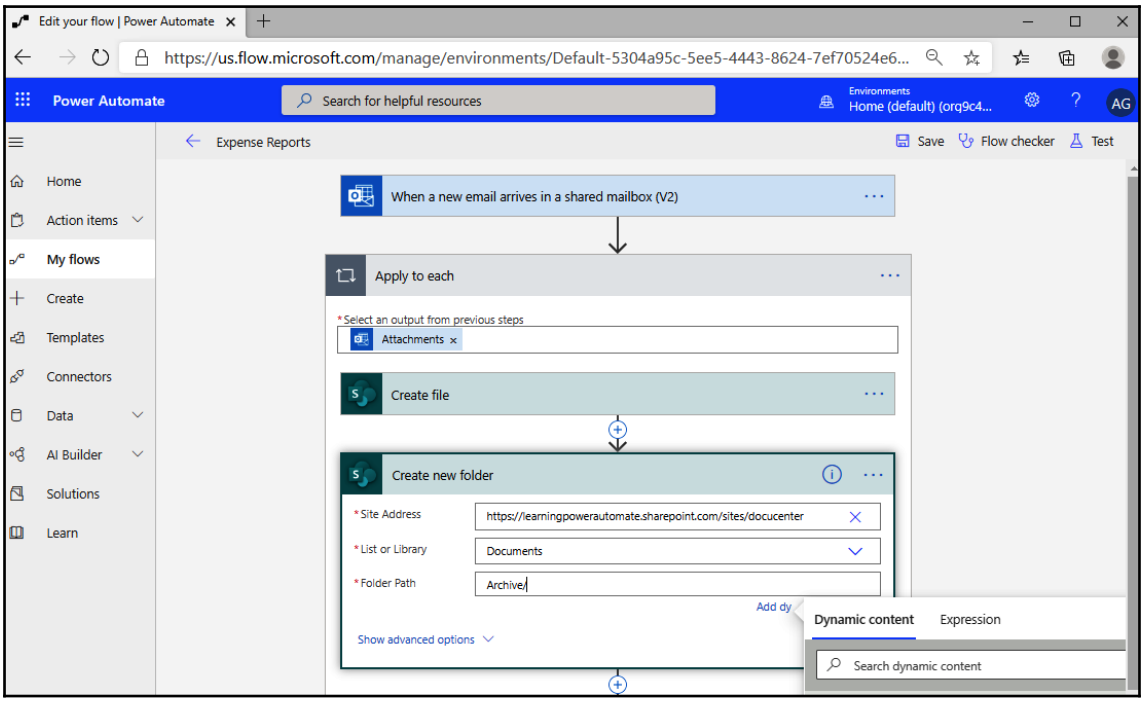
Copying files to SharePoint

Suppose your organization has a requirement to make an archival copy of all the content that passes through a particular work process. In this example, we're going to build on the flow we created in `Chapter 3`, *Working with Email*, in order to save email attachments and copy those file attachments to a new folder. Follow these steps to do so:

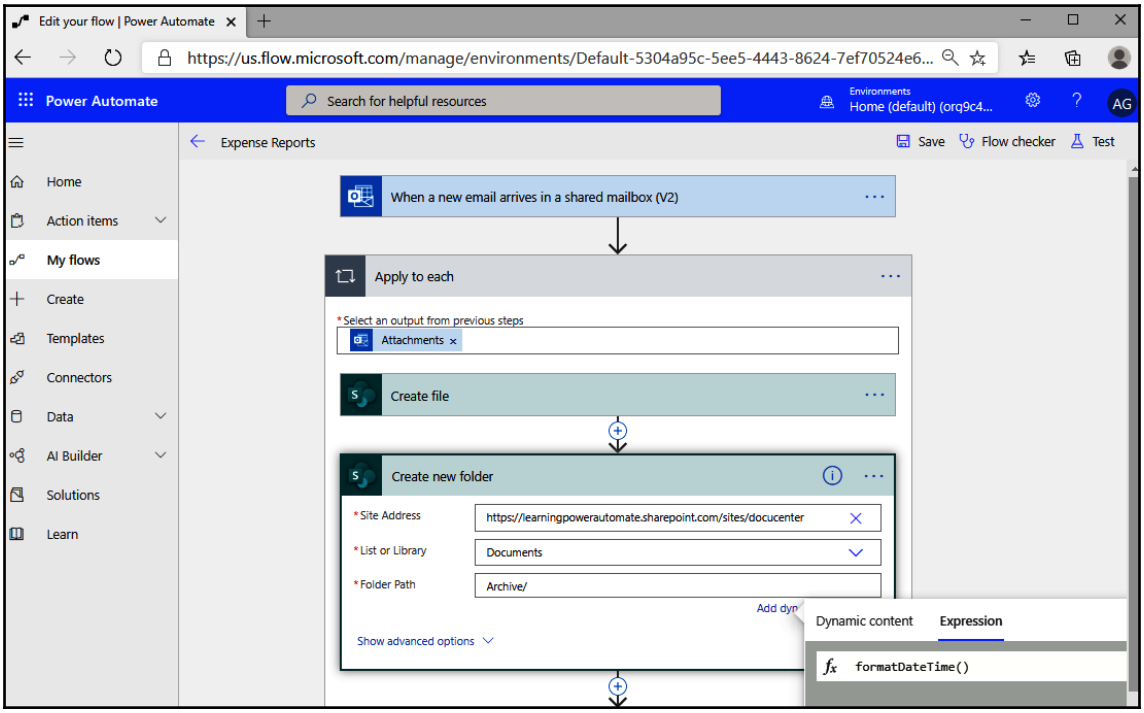
1. Launch the Power Automate web portal (`https://flow.microsot.com`) and select **My flows**.
2. Select the ellipsis next to the **Expense Reports** flow you created in `Chapter 3`, *Working with Email*, and click **Edit**.
3. Click the **Apply to each** step to expand it:



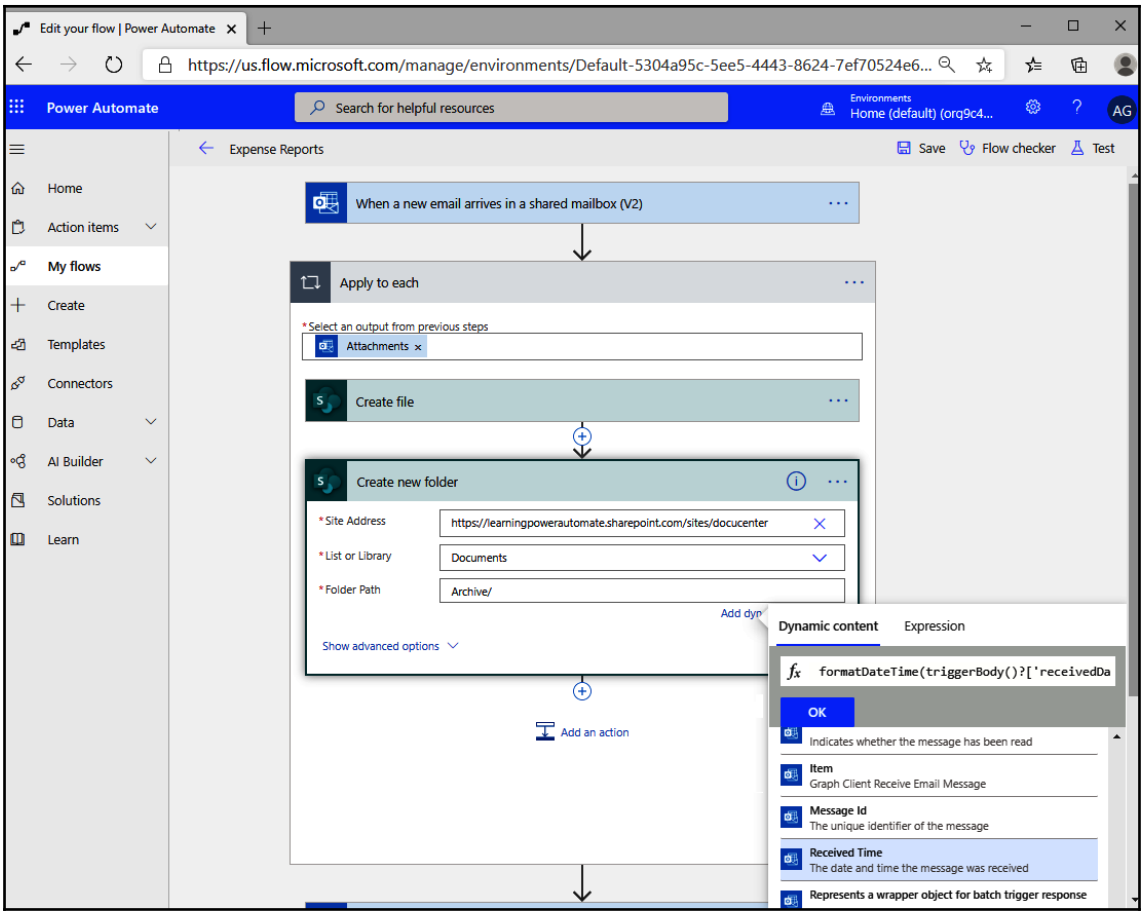
4. Click **Add an action**.
5. In the **Search connectors and actions** box, type `Sharepoint create new folder` and select the **Create new folder** SharePoint action.
6. In the **Site Address** box, enter the destination SharePoint site. If you haven't used this site in the flow before, you may need to enter it manually.
7. In the **List or Library** box, select a document library that you wish to store the files in.
8. In the **Folder Path** box, enter a folder structure. In this case, we're going to use an existing folder called `Archive`, but then create a new folder path based on the date of the received email. If you're creating subfolders here, enter the top-level folder path with a trailing forward slash (`/`) character. Then, select **Add dynamic content**:



9. In this example, we're going to create a folder based on the date/time the message was received and use that as part of the folder name. To do this, select **Dynamic content**, then click the **Expressions** tab. Begin typing in the date and select the **formatDateTime** function. Then, enter trailing parentheses (`()`), as shown in the following screenshot:

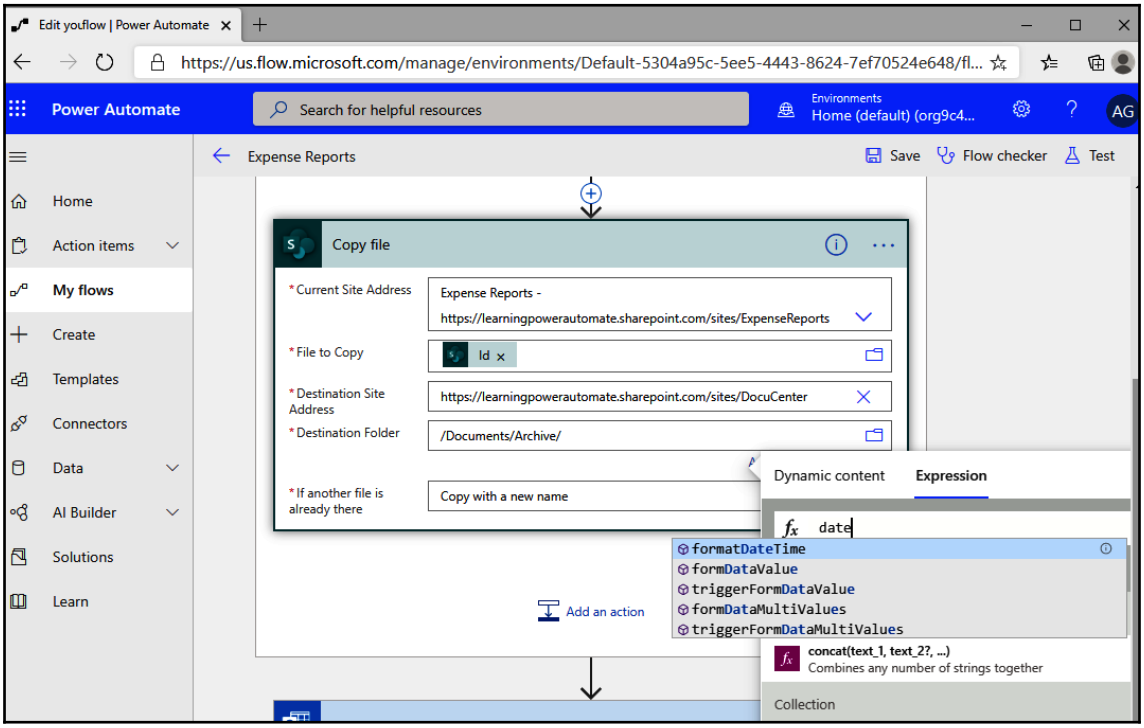


10. Select the **Dynamic content** tab.
11. Place your mouse cursor between the parentheses and select the **Received Time** dynamic content object.
12. Move the cursor to the end of the line before the closing parenthesis and add `'yyyy-MM-dd'`.
13. The expression in the box should update to `formatDateTime(triggerBody()?['receivedDateTime'],'yyyy-MM-dd')`. Click **OK**:

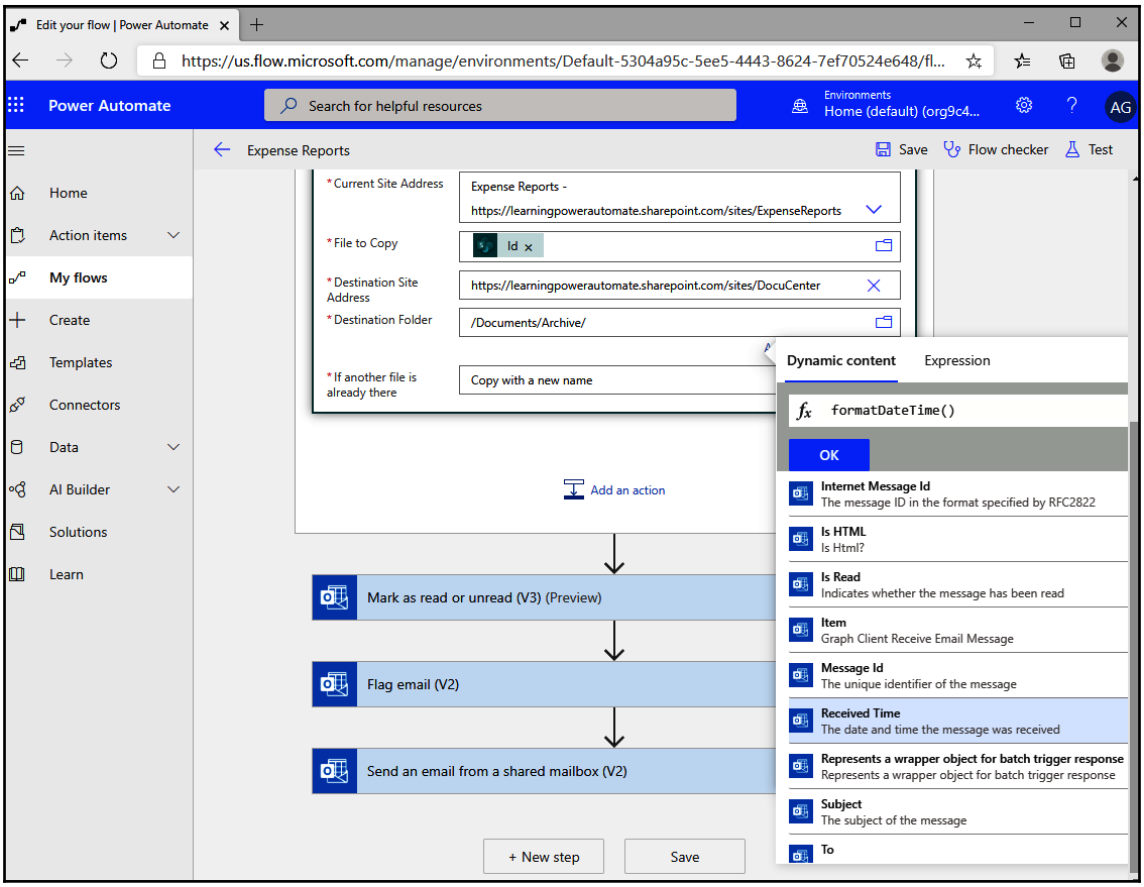


14. Click the **Add an action** button after the **Create new folder** action.
15. In the **Search connectors and actions** box, type `Sharepoint copy file` and select the **Copy file** action from the list.
    In the **Current Site Address** box, select the existing SharePoint site where the Expense Reports flow is configured to save the file.
16. In the **File to Copy** box, select the dynamic content object that represents the ID of the file saved in the **Create file** action.
    From the **Destination Site Address** drop-down, select a SharePoint URL where you wish to save this new archival copy of the file. If you haven't used this URL before, you may need to type it in manually.

17. From the **Destination Folder** drop-down, select a folder to hold the destination file. You can use a base folder name or use a dynamic content variable or function to name the folder. In this example, we're going to create a folder based on the date the message was received and use that as part of the folder name. To do this, select **Dynamic content**, then click the **Expressions** tab. Begin typing `date`, and then select the **formatDateTime** function:

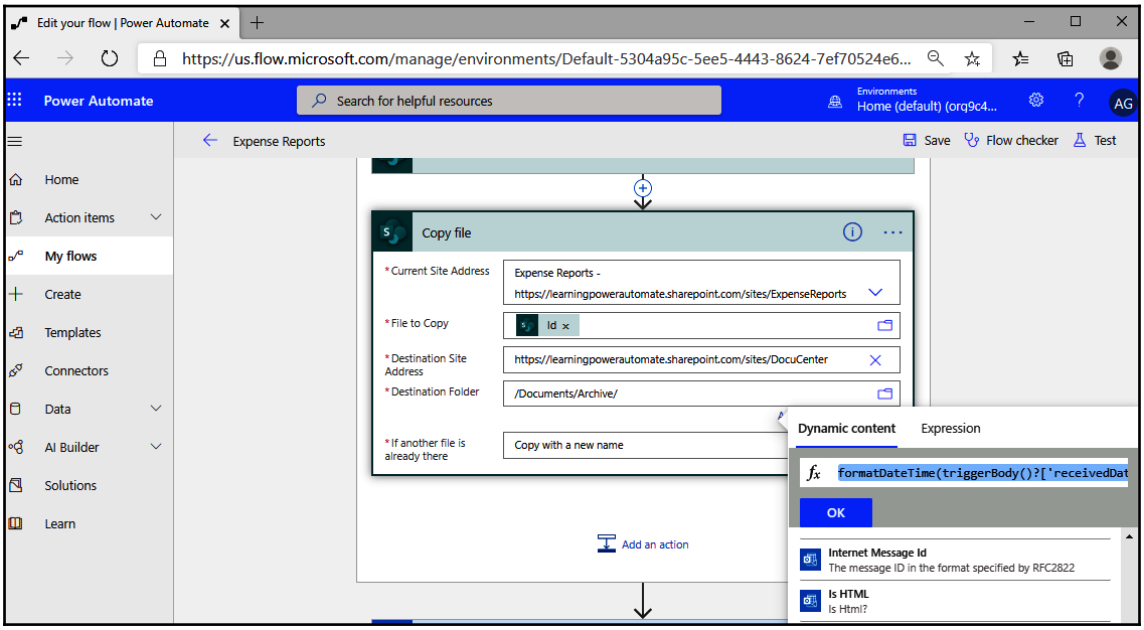


18. Next, add left and right parentheses to the end of the expression and select the **Dynamic content** tab.
19. Place your mouse cursor between the parentheses and select the **Received Time** dynamic content object:



20. Move the cursor to the end of the line before the closing parenthesis, and then add `'yyyy-MM-dd'`.

21. The expression in the box should update to `formatDateTime(triggerBody()?['receivedDateTime','yyyy-MM-dd'])`. Click **OK**:



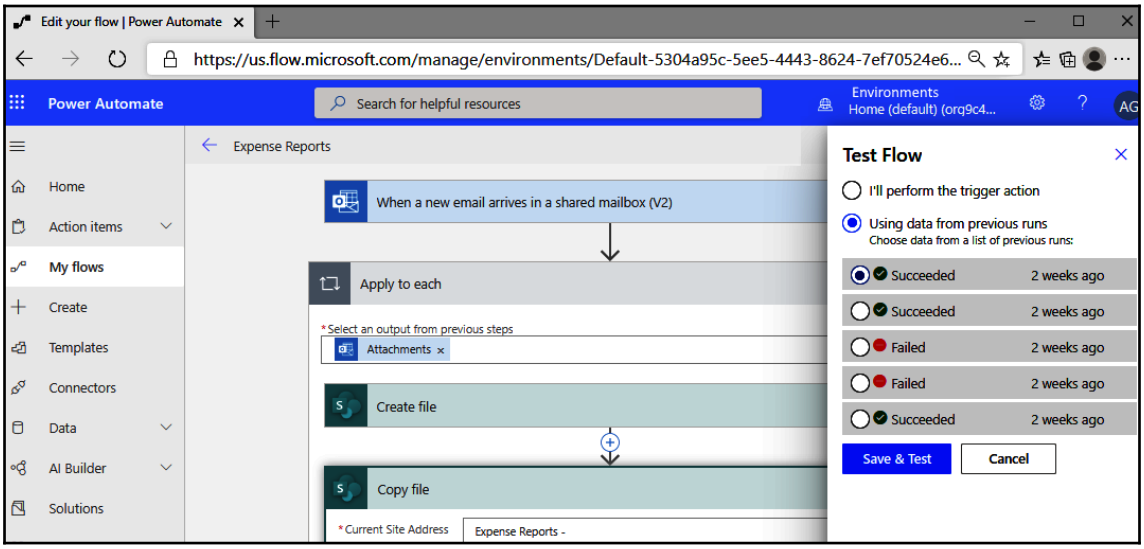
22. In the **If another file is already there** box, you can choose from four options: **Copy with a new name**, **Fail this action**, **Replace**, and **Enter a custom value**. Which option you choose depends on your business requirements. In this case, we're going to select the **Copy with a new name** option. You can use the **Enter a custom value** option to build your own naming convention using expressions and dynamic content, as we did in the previous step, as well.
23. Click **Save** to save the updated flow.

Next, we'll verify the results of the changes.

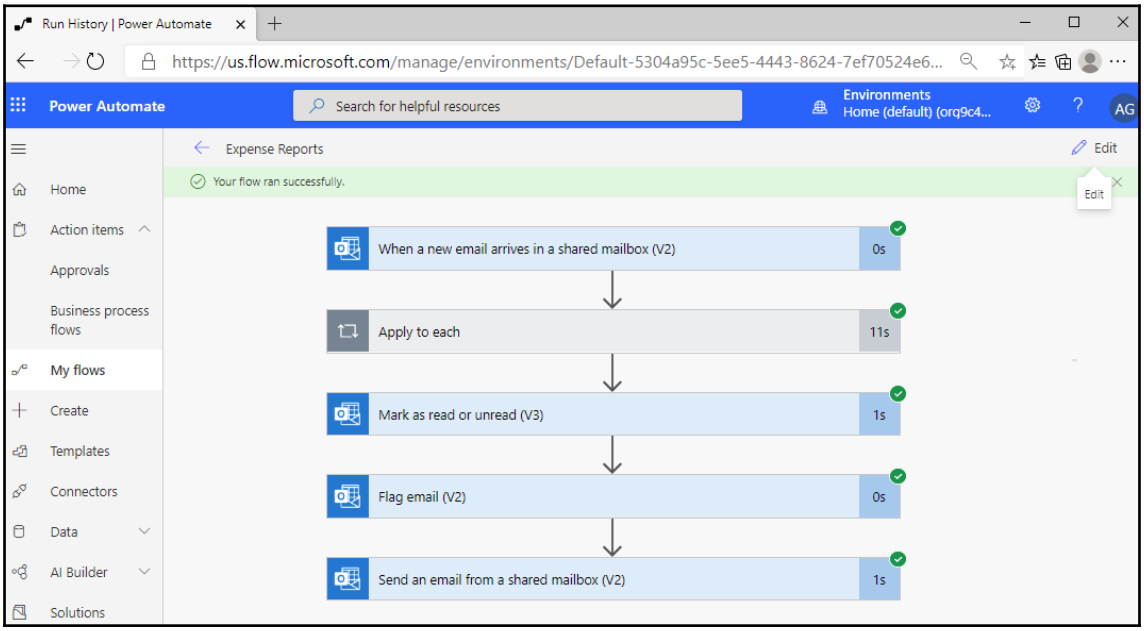
Verifying the results with the test flow

It's important to make sure the syntax is correct for when we make any updates to configurations. To do so, you can use the **Test Flow** function at the top of the page. Follow these steps to do so:

1. Click **Test** to test it and ensure that you get the expected result. You can choose to trigger this manually or reuse data from a previous run. In this case, we're going to use the results from a previously successful run. Select the run and click **Save & Test**:



2. The results should come back successful. If not, review the flow configuration and make any necessary updates:

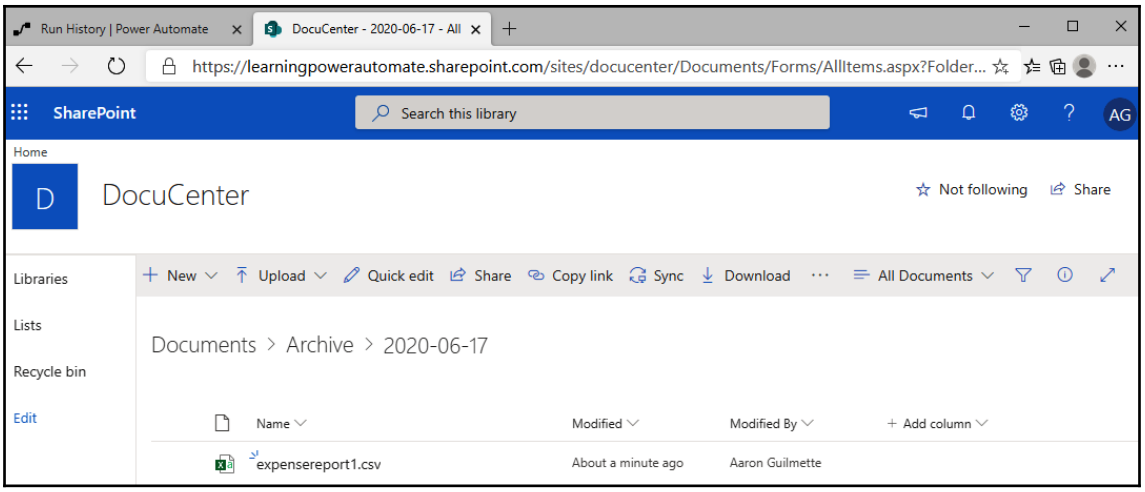


However, it's also good to verify that the updated changes are where you anticipate them to be. We'll look at this next.

Verifying the results manually

While the Test Flow function may state that it was successful, that doesn't necessarily mean it's *correct*. It just means it completed without *errors*. When it comes to formatting expressions or functions, many are case-sensitive. If, for example, you are processing a date such as 20 June 2020 and you use **YYYY-MM-DD** instead of **yyyy-MM-dd** in the **formateDateTime** expression, you'll get a folder named **YYYY-06-DD** instead of **2020-06-20**.

To verify that you achieved the expected result, you can expand and examine the resulting JSON values at each of the steps. Alternatively, you can simply navigate the target library, as shown in the following screenshot:



With that, the file that was attached to the recently processed expense report will be deposited into the correctly named folder.

Next, we'll look at using the Dropbox connector to work with files.

Publishing files to Dropbox

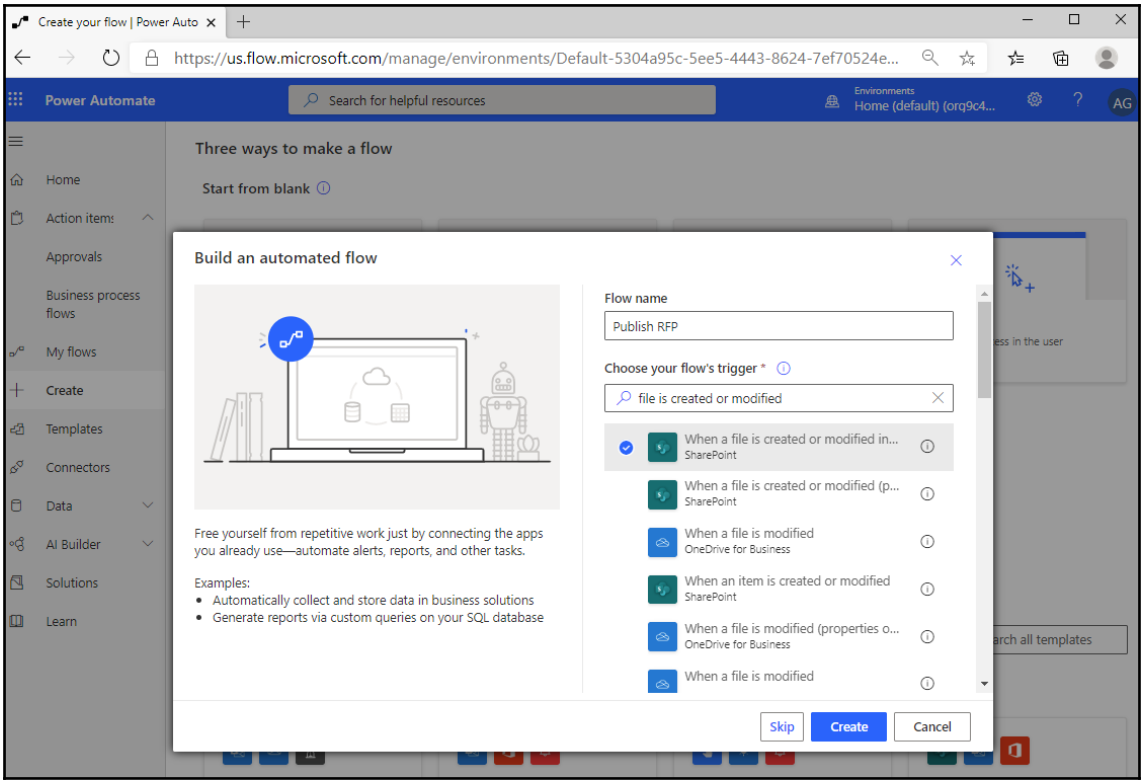
Another popular file copy scenario is moving files to repositories *outside* of SharePoint Online. The public site option for SharePoint Online was deprecated several years ago, which makes it more difficult, in some instances, to make information broadly available.

For example, suppose your organization hosts a SharePoint Online site where individuals save **Requests for Proposals** (**RFPs**) that have been prepared to go out to bid, and you need to make those automatically available on a public download site for potential vendors. You can create a flow to monitor that SharePoint Online site and copy contents to the Dropbox site to deal with this.

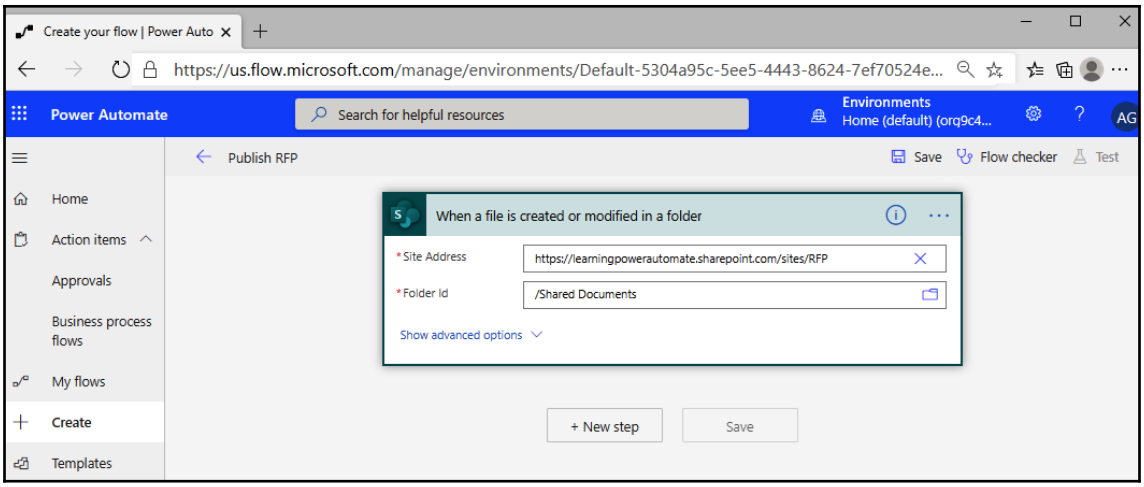
To get started, follow these steps:

1. Launch the Power Automate web portal (`https://flow.microsoft.com`) and select **+ Create flow**.
2. Select **Automated flow**.

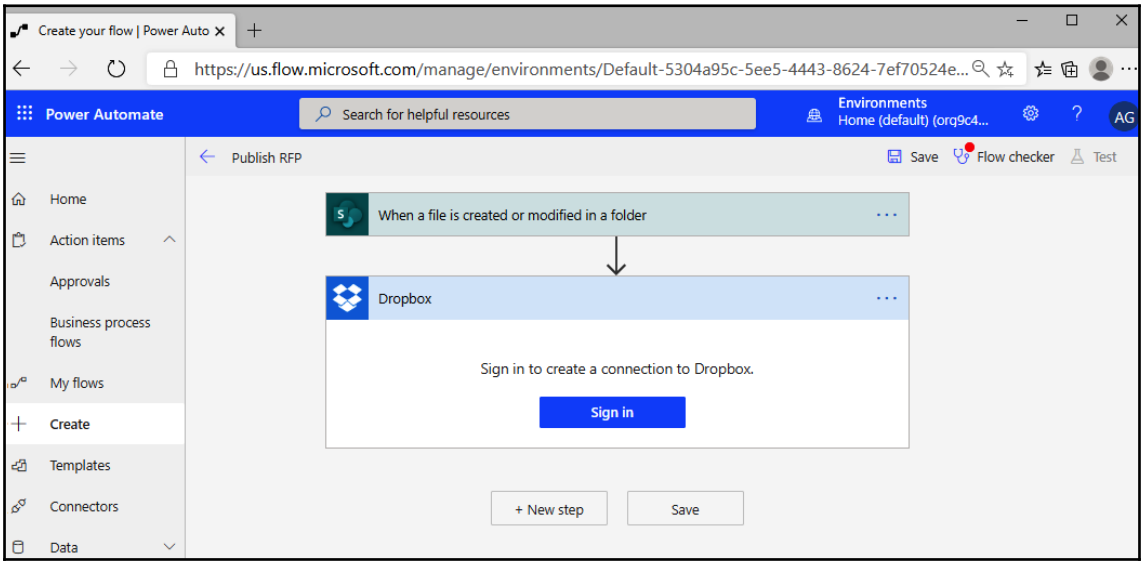
3. Enter a name for the flow (in this case, we'll use `Publish RFP`) and search for the SharePoint trigger called **When a file is created or modified**. Select it and click **Create**:



4. In the **Site Address** box, enter the URL for the SharePoint site that will be monitored for new files.
5. In the **Folder Id** box, select the folder that will be monitored:



6. Click **+ New step**.
7. Then, select the **Choose an action** box, search for `Dropbox`, and select the **Create file** action.
8. Click **Sign in** and provide the necessary credentials to the service:



9. If a CAPTCHA or other identity challenge is presented, provide it.

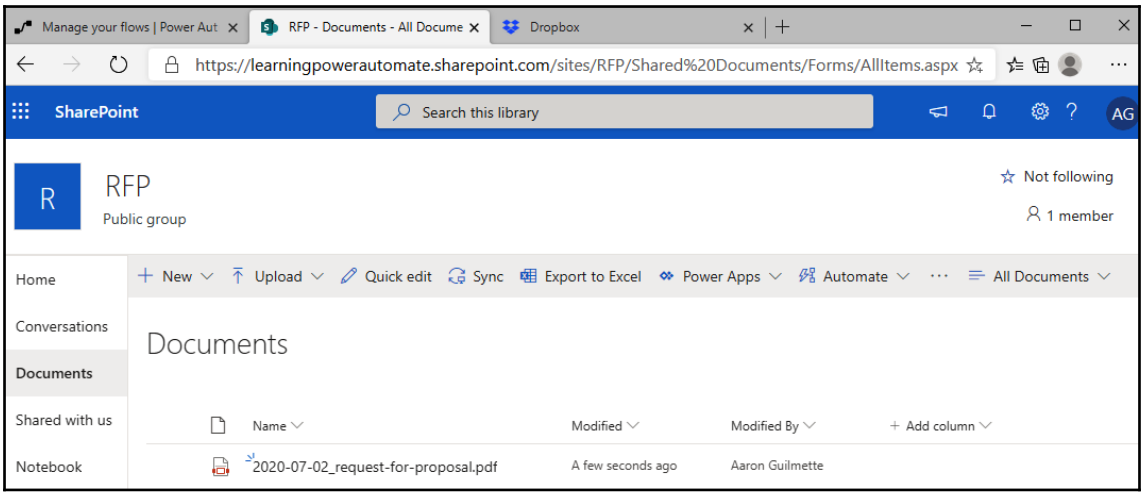
10. Once authentication has been verified, in the **Folder Path** box, select the folder path. / indicates the top-level folder.
11. In the **File Name** box, select **Add dynamic content** and select the **File name** dynamic content item.
12. In the **File Content** box, select **Add dynamic content** and select the **File Content** dynamic content item.
13. Click **Save**.

At this point, the flow is ready to test.

Verifying the results

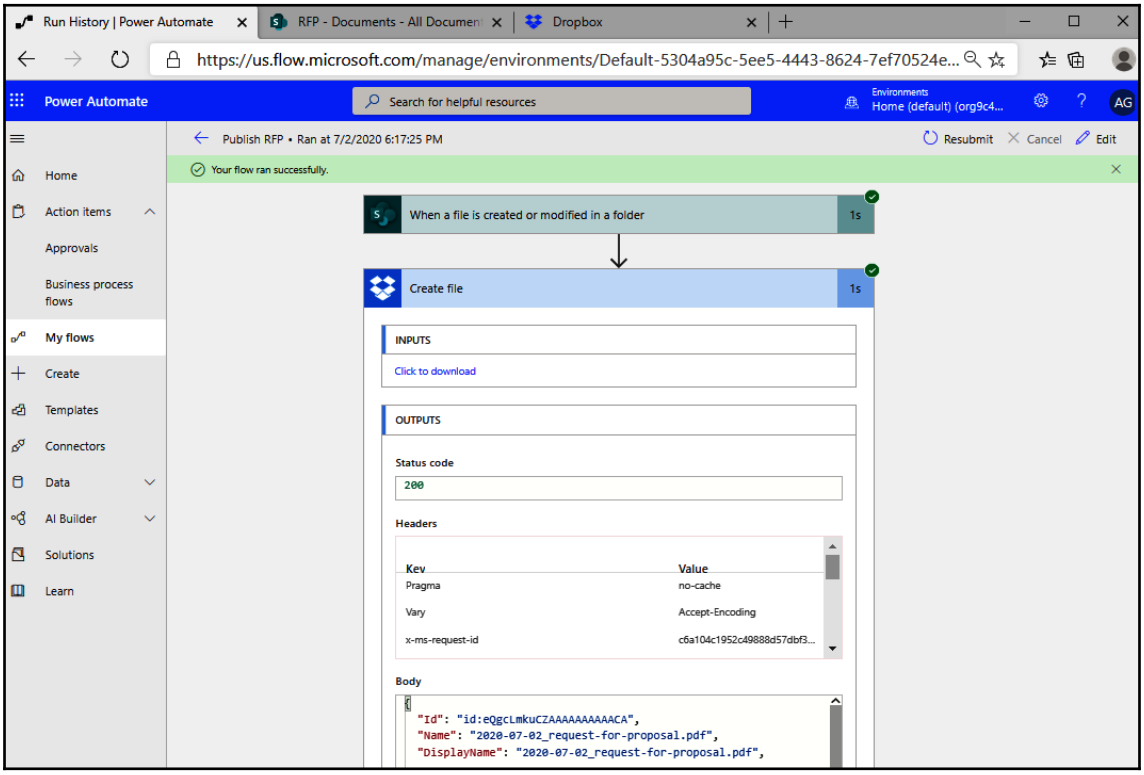
To verify the functionality of this flow, follow these steps:

1. Upload a document to the monitored SharePoint document library:

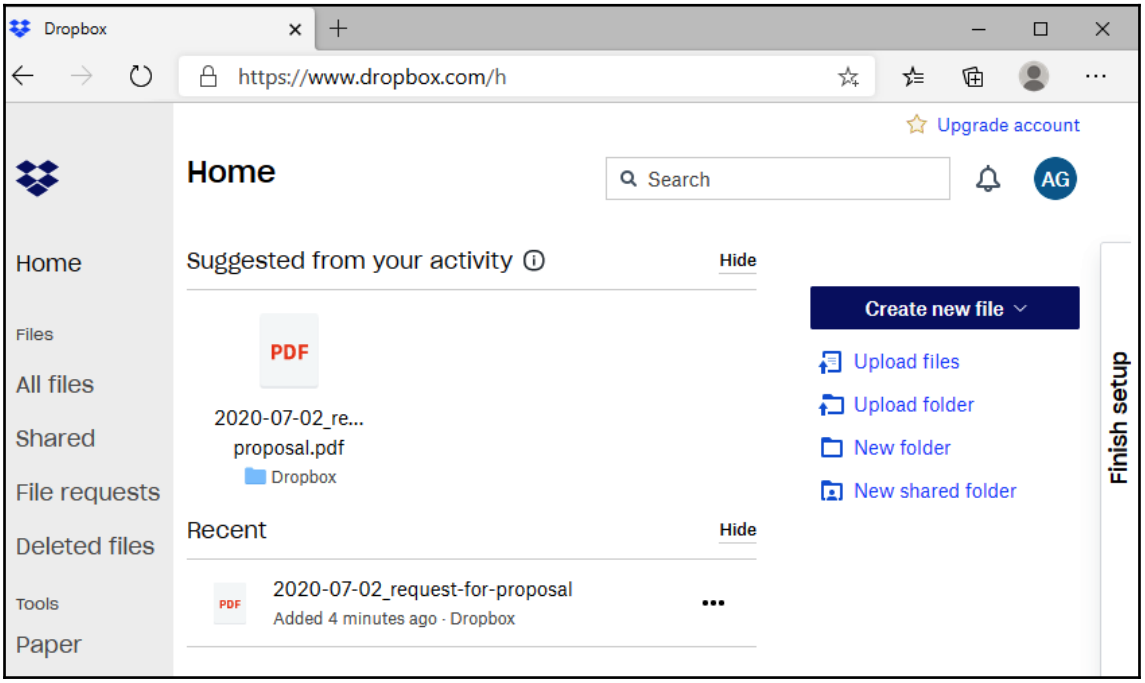


2. Next, from the Power Automate web portal, navigate to **My flow**.
3. Select the ellipsis next to the newly created flow and select **Run history**.

4. Select the most recent run to review all activity:



5. Next, log into the Dropbox site and review the recent files:



At this point, you can be confident that your publishing flow is working correctly.

Summary

This chapter introduced a few new actions (Create folder and Copy file) for the SharePoint connector, a new connector to an external service (Dropbox), as well as the concept of *expressions* (or functions). Expressions can be used to further create custom content or extract content from entities by following the steps of a flow.

Using these new components, you were able to copy files internally between SharePoint sites, as well as copy them to an external service. Both types of flows (copying files internally and externally) can be used as part of solving very common business automation problems, and increasing your familiarity with expressions will help you make detailed flows with customizable data outputs.

In the next chapter, we'll shift the focus to creating button flows.

5
Creating Button Flows

Up to this point, we've worked primarily with *automated flows* – that is, flows that happen based on Power Automate detecting, or being notified of, a change and then acting upon it. In this chapter, we're going to shift gears to create a different type of flow – one that happens when you manually start it.

Since the original launch of Power Automate as Microsoft Flow, *button flow* has been known by a few other names, including *manual flow* and *instant flow*. While this terminology is largely interchangeable, the term *button flow* is typically used to indicate a flow that has been published to a mobile device that can be initiated by tapping a button in the Power Automate mobile app experience, while the term *instant flow* is typically seen in the Power Automate web portal.

In this chapter, we're going to cover the following topics:

- Learning about button flows
- Creating a button flow to email a manager
- Executing a button flow from the Power Automate mobile app

By the end of this chapter, you'll be familiar with creating and using button flows. Let's get started!

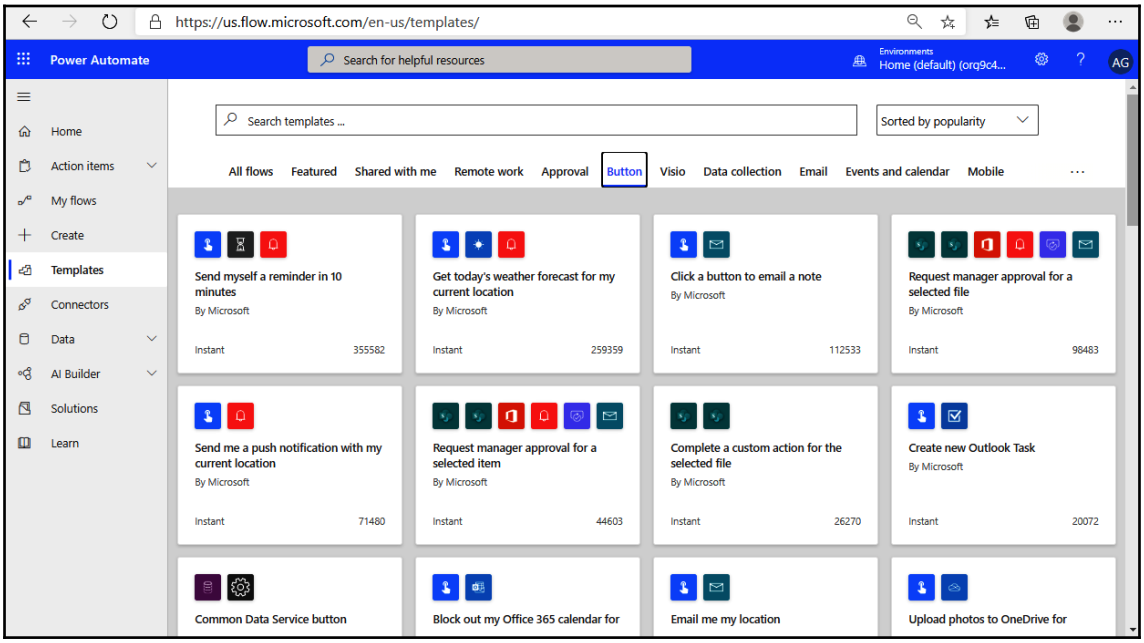
Learning about button flows

Button flows (or instant flows), as previously mentioned, are manually triggered flows. They don't monitor anything, nor do they have any sort of REST-based trigger that instantiates them. Button flows can be used for a variety of low-impact tasks, such as executing reminders or notifying individuals.

A button or instant flow can be created two different ways:

- From a button flow template
- From a blank template

Button flows can be created from either the Power Automate mobile app or the Power Automate web portal. You can see examples of button flow templates in the following screenshot of the web portal:



While most button flows are designed to perform a one-time task with simplicity, more complex instant or button flows can be used that interact with data from SharePoint lists, documents, or data stored in the Common Data Service.

In the next section, we'll create a button flow from the web portal.

Creating a button flow to email a manager

There are times when you just need a quick form letter to respond or send someone a notification, such as when you're out of the office due to holiday or illness. In this section, we're going to configure this simple button flow to email your manager that you're going to be out of the office. Like the flows you've created in the last few chapters, we're going to take advantage of dynamic content to make sure we retrieve the value stored in the manager property for an account in Office 365.

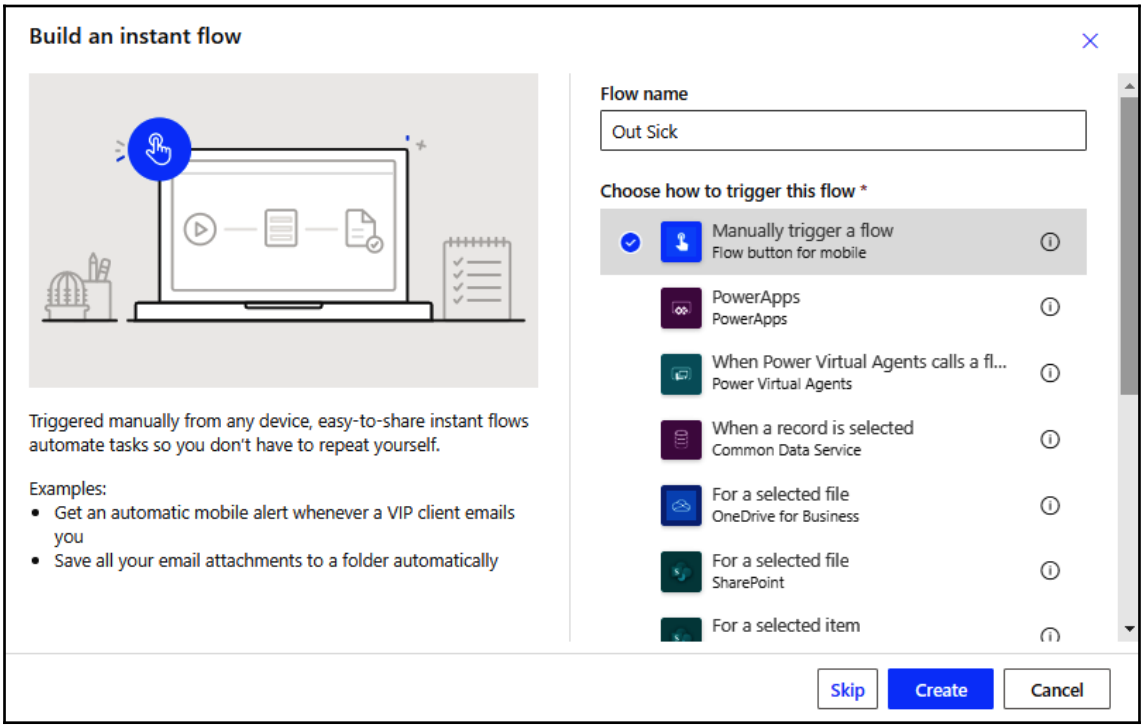
In order to do this, we're going to introduce a new connector and two new actions:

- The **Office 365 Users** connector: This connector allows you to connect to Azure Active Directory to retrieve information pertaining to user object data. You can learn more about the Office 365 Users connector here: `https://docs.microsoft.com/en-us/connectors/office365users/`
- The **Get my profile (V2)** action: This action will retrieve your specific details from Azure Active Directory. You can learn more about what properties this action can access here: `https://docs.microsoft.com/en-us/graph/api/resources/user?view=graph-rest-1.0#properties`
- The **Get manager (V2)** action: This action retrieves the manager property of the selected user. In order to use this, the **Manager** Exchange property must be configured. You can learn more about the Get manager (V2) action here: `https://docs.microsoft.com/en-us/connectors/office365users/#get-manager-(v2)`

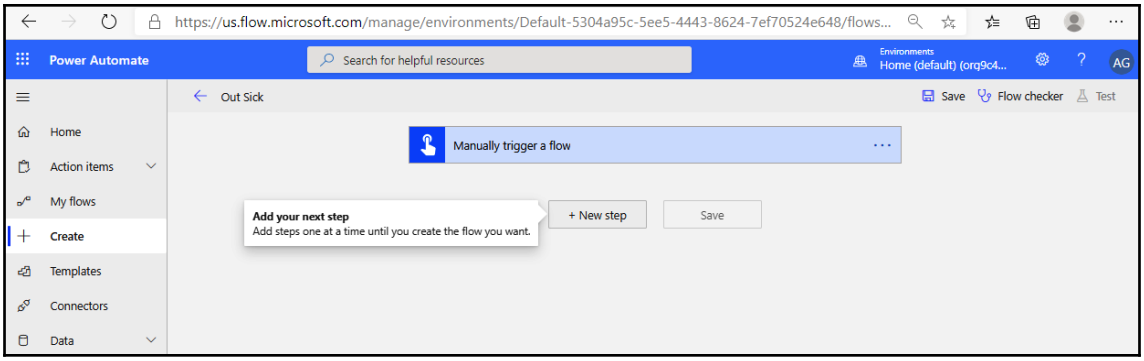
With these new components, we'll be able to retrieve your manager data and send off a pre-populated email. As you work through the example, take careful note of the order of the steps when working with Office 365 data – in this case, retrieving your profile data and then retrieving the manager value.

Follow these steps to complete the example:

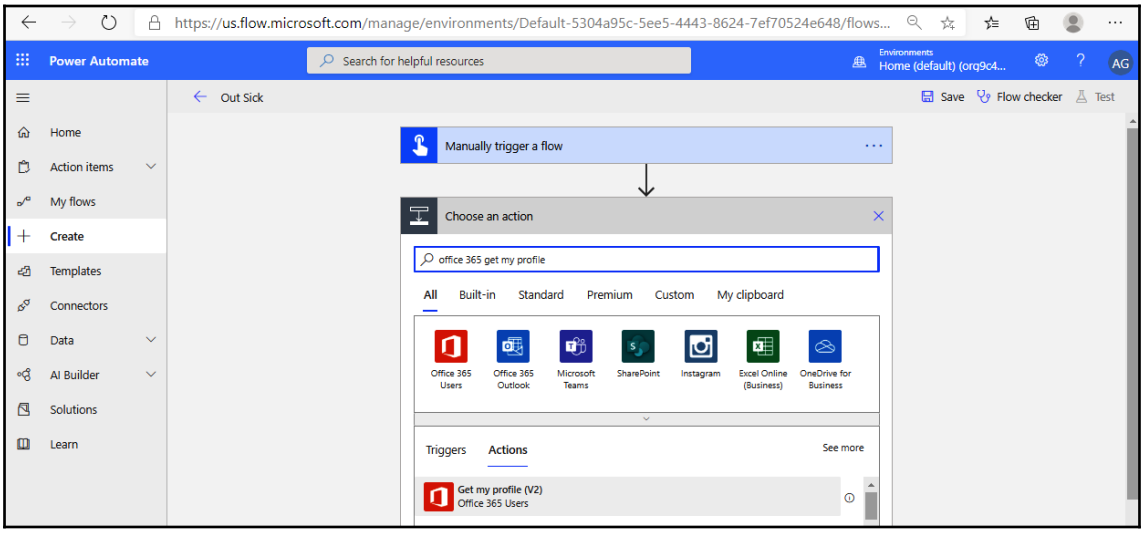
1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **+ Create** and select the **Button flow**.
2. Enter a name for the flow and select **Manually trigger a flow**. Then, click **Create**:



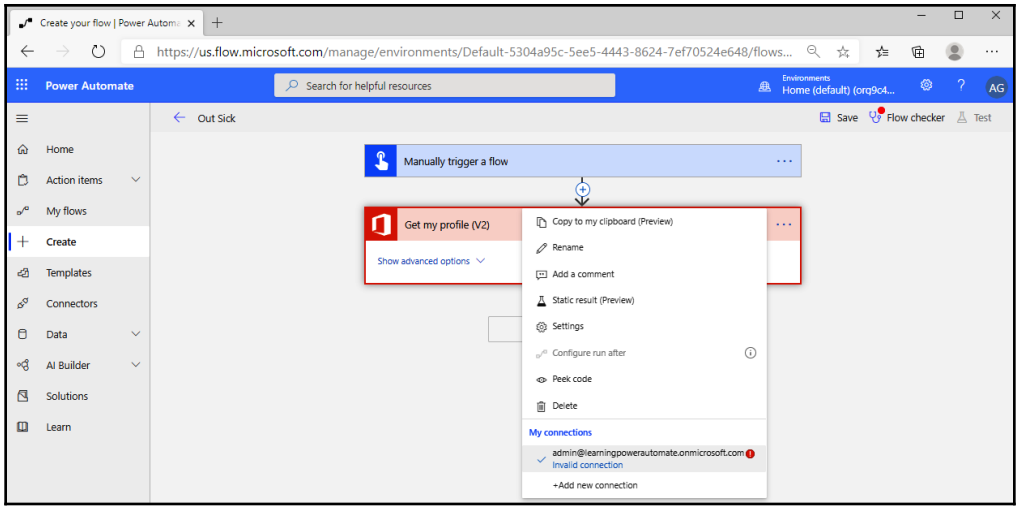
3. Click **+ New step** to add a new step:



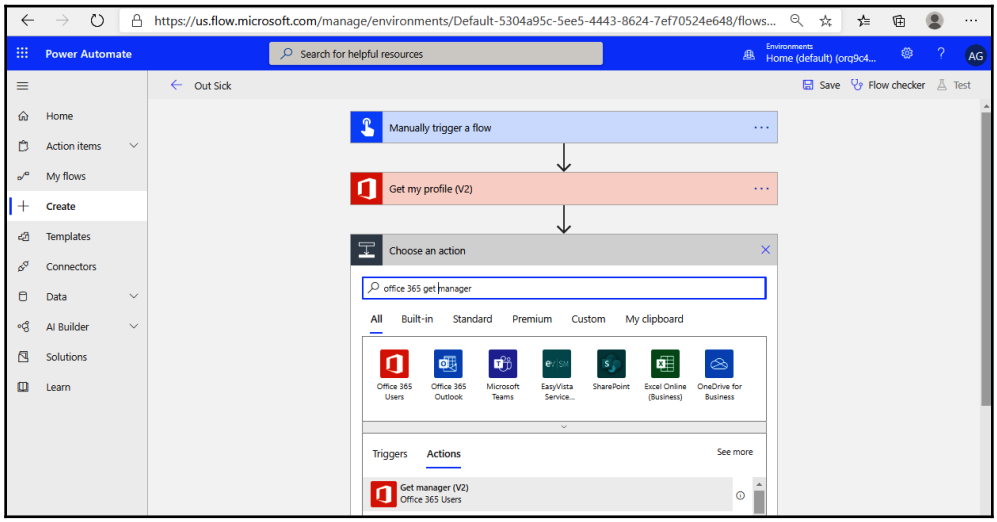
4. In the **Choose an action** search box, enter `office 365 get my profile` and select the **Get my profile (V2)** action:



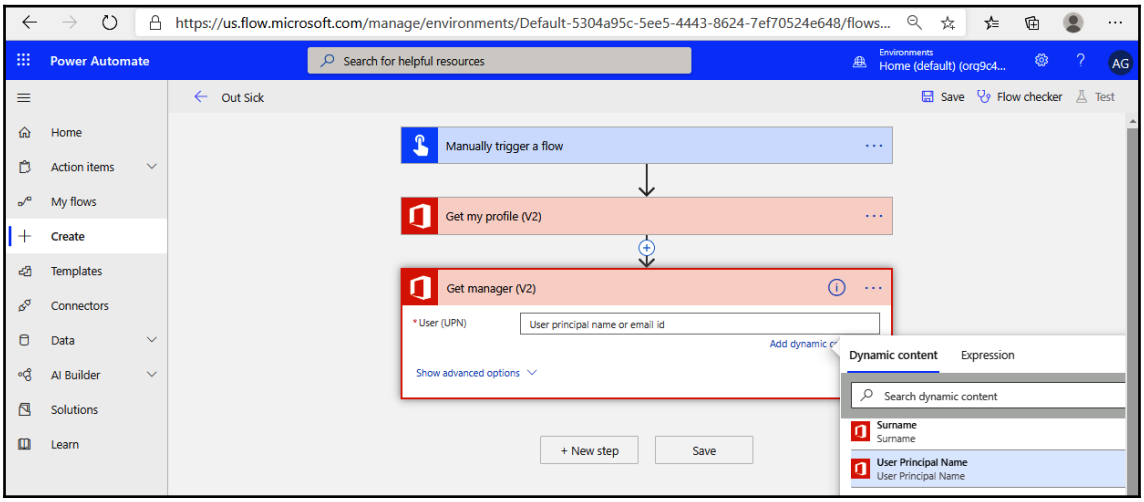
5. If you haven't used or fully configured any of the Office 365 connector objects before, you'll need to authenticate. Select the ellipsis for the **Get my profile (V2)** action, and then, under **My connections**, either select your existing connection (if it was detected) or click **Add new connection**. Enter any credentials:



6. Once your authentication is successful, click **+ New step** to add a new step.
7. In the **Choose an action** search box, enter `office 365 get manager` and select the **Get manager (V2)** action from the list:



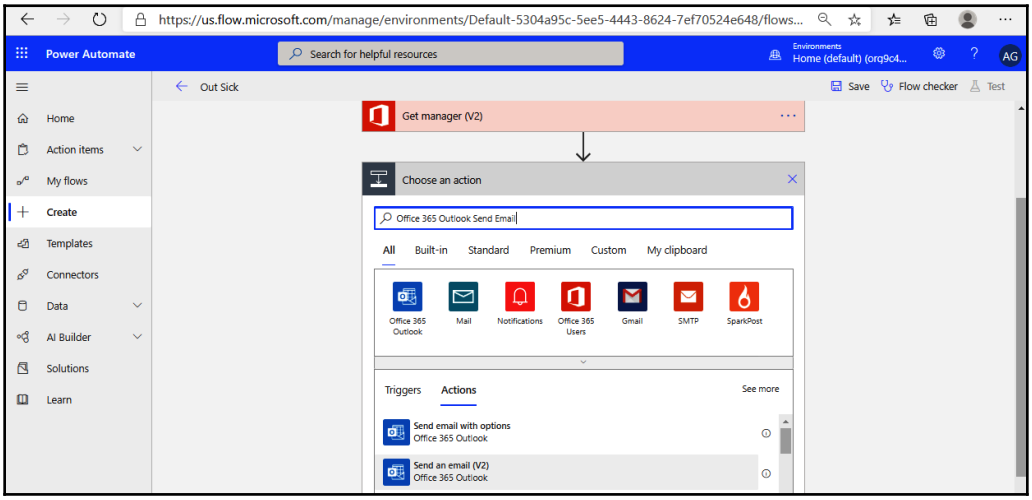
8. Click in the **User (UPN)** value box, and then select the **User Principal Name** value under the **Get my profile (V2)** section. This option will retrieve the `UserPrincipalName` property that was discovered as a result of the **Get my profile (V2)** action previously:



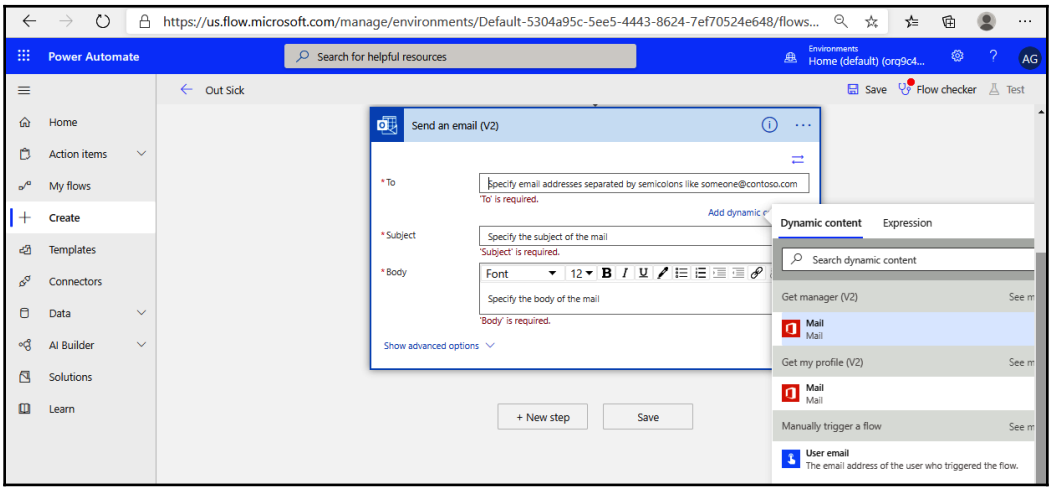
Once you have added the dynamic content value, **User Principal Name**, you can hover over it to see the configuration. It should say `body('Get_my_profile_(V2)')?['userPrincipalName']`, indicating that it is using the `userPrincipalName` value from the **Get my profile (V2)** action.

9. Click **+ New step**.

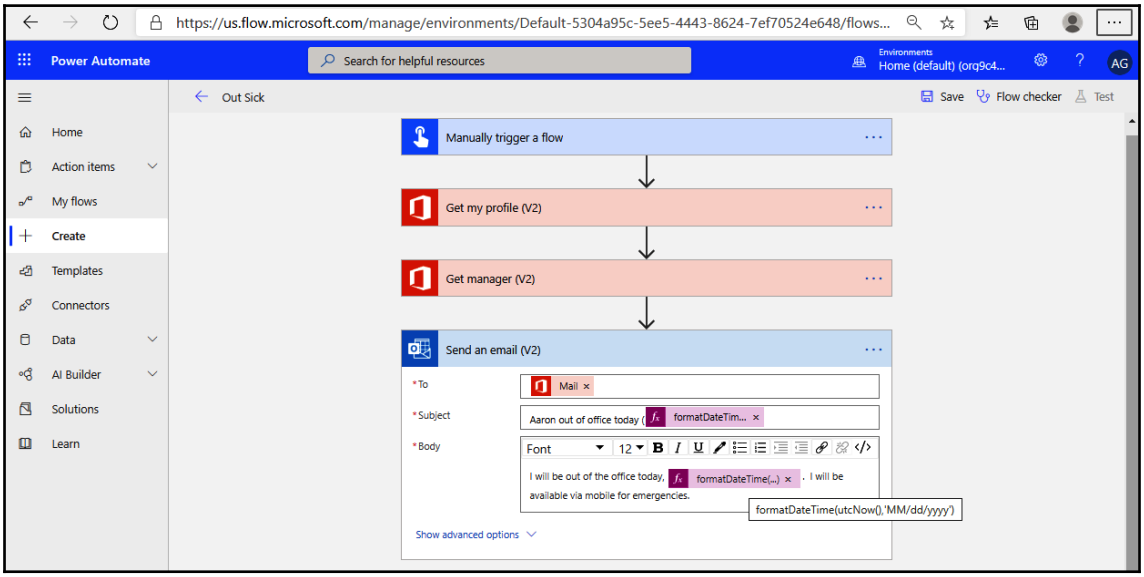
10. In the **Choose an action** search box, enter `Office 365 Outlook Send Email` and select the **Send an email (V2)** action:



11. At this point, you've retrieved the profile value for your manager. You can then use the dynamic content object to populate the **To** address field. Power Automate is aware of the property type restrictions of actions and, by default, automatically selects the most appropriate or likely options. In this case, select the **Mail** property under the **Get manager (V2)** section:



12. Complete the flow by filling out the **Subject** and **Body** fields. This can be a good opportunity to experiment by combining new expressions. For example, one way you can retrieve today's date and format it in **month/day/year** format is by combining the `formatDateTime()` expression and the `utcNow()` expression. Select the `formatDateTime()` expression, and then supply the expression `utcNow()` as the first argument and supply the date format as the second argument: `formatDateTime(utcNow(),'MM/dd/yyyy')`
13. Verify that all of the fields of the action have been completed:



14. Click **Save**.

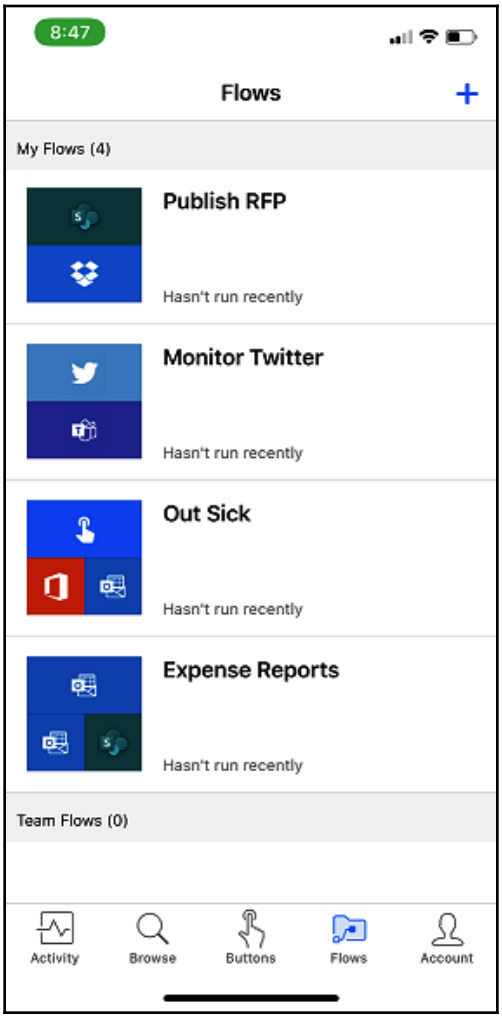
At this point, you should have a properly configured flow that will email your manager. You can view the flow in **My flows** and select the **Play** icon to launch it if desired.

However, since this is designed to be a button that you can execute from anywhere, we'll test it from the mobile app.

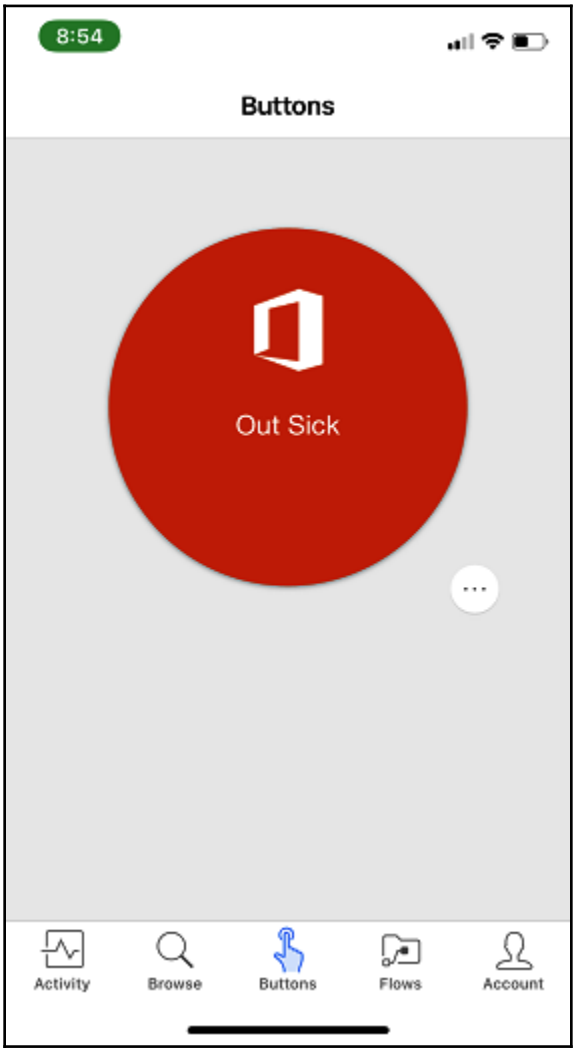
Executing a button flow

Once you have created a button flow, the best way to see its usefulness is by launching the Power Automate mobile app. Once you have signed in, follow these steps to execute the newly-created flow:

1. Verify that the new button flow appears. After launching the Power Automate app, select **Flows**. You should see all of the flows associated with your signed-in account:

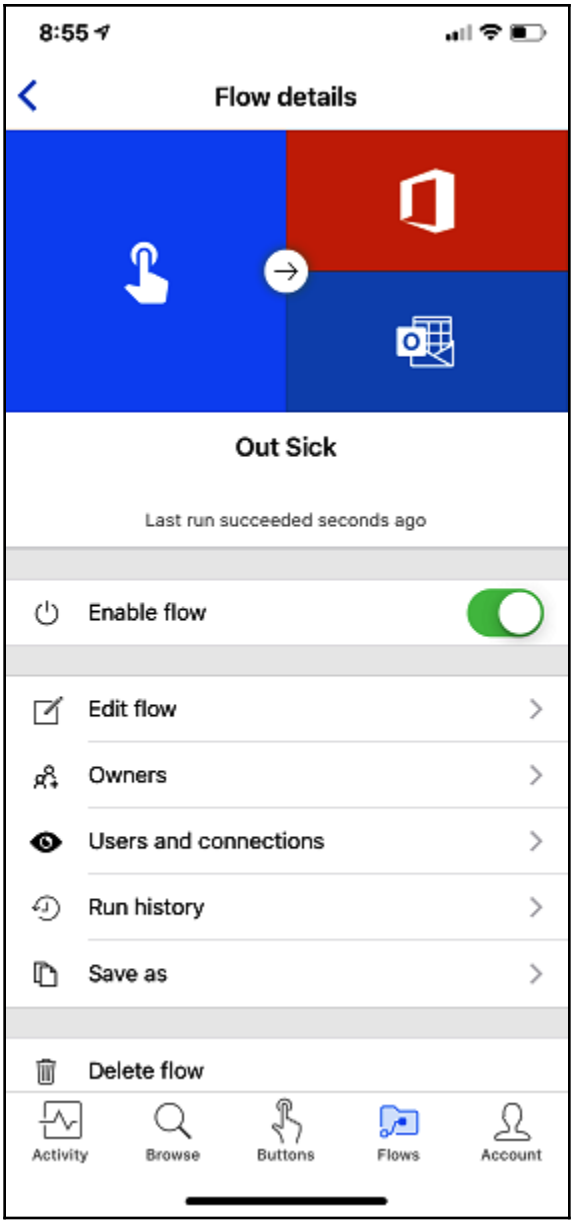


2. After verifying that you can see the new flow, tap on the **Buttons** icon. The new flow should appear:

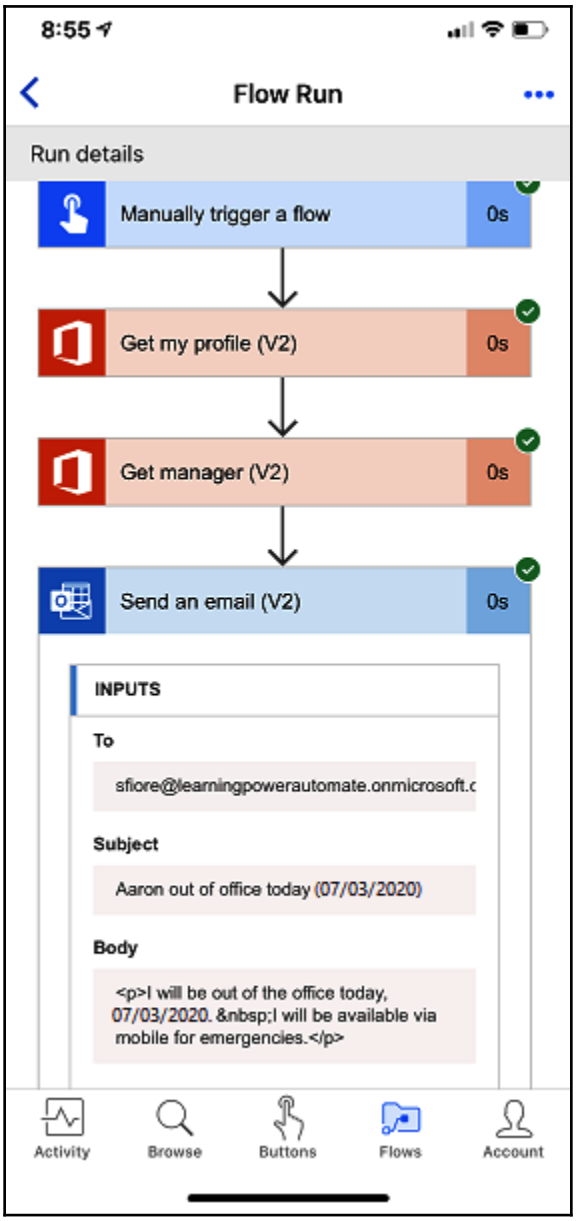


3. Tap the button to execute the flow.
4. After it has completed, tap **Flows** in the tray and then tap the button flow to open the **Flow details** page.

5. Tap **Run history** to display recent runs:



6. Select the run and then examine the details to ensure that it has completed successfully. You can expand any of the steps to view the results of any step, including the expansion of dynamic content or the evaluation of expressions:



The flow is complete. You can perform further verification, if desired, by checking the **Sent Items** folder of the sender's mailbox. Since the flow is authenticated, outgoing emails will be sent from, and saved in, the sender's mailbox.

Summary

In this chapter, we covered new connectors and actions, as well as an example of nesting an expression inside another expression by formatting the value from `utcNow()` with the `formatDateTime()` expression. By combining multiple expressions, you can display variable data in a format that makes sense for the flow.

As you've seen, button flows can be useful for quick tasks that can be executed from either the web portal or a mobile device. While button flows can be created from mobile devices, you may want to consider creating complex flows from the web interface so that you can see more components on the screen at the same time.

In the next chapter, we're going to look at using push notifications to increase visibility and draw attention to noteworthy items.

6
Generating Push Notifications

Push notifications are a type of communication mechanism used to deliver a message (popup, banner, or other types of notification message) to an end user's device. Push notifications are a handy way to keep users connected to data and activities, drive engagement scenarios, or prompt users to perform an action or related app task.

Microsoft Power Automate enables users to configure their own push notifications as part of a flow. For example, in addition to processing an email attachment, you may wish for a flow to send a notification to your device when an email attachment is processed.

In this chapter, we'll look at how push notifications work and how to integrate notifications into your flows. The topics we'll cover are as follows:

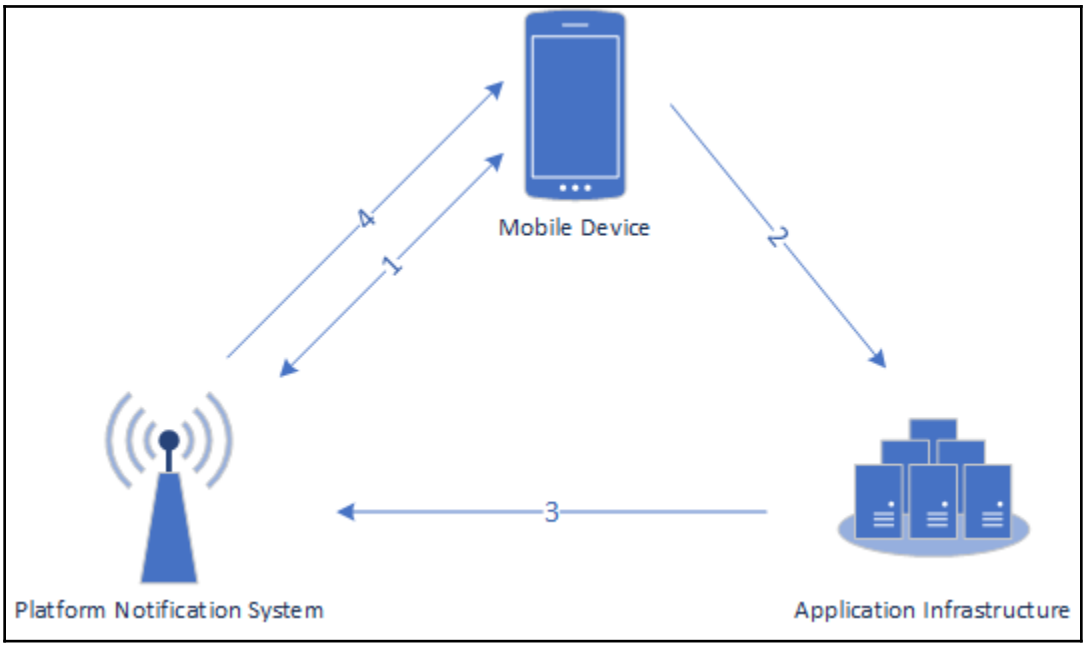
- Learning about push notifications
- Configuring a notification for emails from your manager
- Reviewing a notification

Using notifications effectively can help you stay on top of important tasks. Let's begin!

Learning about push notifications

Push notifications are delivered using subscriber-based **Platform Notification Systems** (**PNSes**). PNSes, as the name indicates, are unique to mobile platforms. Platforms include Firebase for Android devices, Apple Push Notification services for iOS platform devices, and Windows Notification services for Windows-based devices. Vendors that develop multi-platform applications must integrate with each PNS for which they wish to offer push notifications.

The following diagram depicts a basic overview of a PNS:



In order to receive a push notification, the following actions take place:

1. The user configures an application to receive a push notification. When this happens, the application contacts the push notification system for a *handle*, or a unique token that identifies the app and device being subscribed. These identifiers are a unique pairing, meaning that the token for one app cannot be used for another app, even if both apps are installed on the same device.
2. The application then saves this unique handle or token on the application server's infrastructure.
3. When an application event triggers a push notification, the application's backend server contacts the PNS, supplying the handle to tell the PNS which device/app pair to contact.
4. The PNS contacts the application on the target device.

Power Automate is the application that will request the handle for the notification that you'll configure, and the trigger for the notification will be the condition you configure in the flow to activate the notification.

Power Automate notifications can also be configured to send an email as well. Notifications are really just actions that can be configured whenever you need to raise an alert, either via email or a push notification to a device. A notification action can be added any time a set of conditions or circumstances require immediate attention by a user, whether internal or external to the Power Automate and Office 365 environment.

Next, you'll see how to configure a basic flow that integrates push notifications.

Configuring a notification for emails from your manager

Configuring a flow that uses a push notification is very straightforward. You can add push notifications to existing flows or create a new flow that utilizes a push notification.

In this example, we're going to create a new flow that sends a push notification when the monitored mailbox receives an email from the user's manager. We're going to use the following components that we utilized in the previous chapter:

- **Connectors**: Office 365 Users and Outlook
- **Actions**: Get my profile (V2) and Get manager

We're also going to introduce a new connector and new actions:

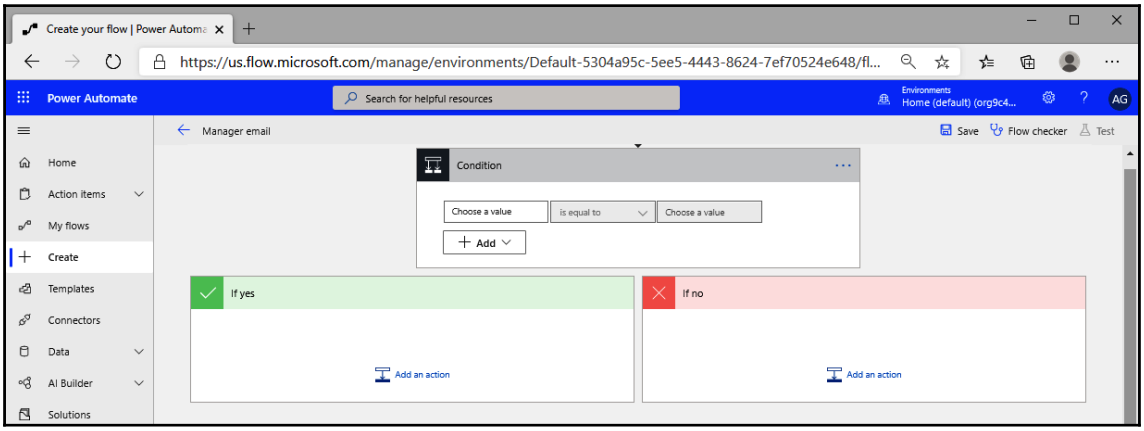
- **Notifications connector**: This connector allows you to interact with the notifications service. You can learn more about the notifications connector here: `https://docs.microsoft.com/en-us/connectors/flowpush`.
- **Send me a mobile notification action**: This action sends a push notification to a mobile device. You can learn more about the Send me a mobile notification action here: `https://docs.microsoft.com/en-us/connectors/flowpush/#send-me-an-email-notification`.
- **Condition control**: This allows you to perform different actions based on a condition, such as the value of a property.

Depending on how your applications are configured, you may be able to configure notifications to take advantage of *deep linking*, in which the notification takes the user to the relevant item (as opposed to a dashboard of items). You can learn more about deep linking here: `https://powerapps.microsoft.com/en-us/blog/powerapps-deep-linking/`.

You'll also notice that we're introducing a new type of action – a *condition*. Rather than being tied to the capabilities of a particular connector, it can help your flow make decisions on what actions to perform.

Introducing conditions

Conditions, as indicated, are a type of control action that allows you to build more complex flows based on the values of certain properties, expressions, calculations, or inputs. In traditional programming or scripting languages, you might see this referred to as an `If` statement. The following screenshot illustrates an empty condition action:



In programming or scripting, the `If` construct allows the application to follow different paths (or branches) based on the result of a previous command or choice. Common examples include `if [condition equals true], then [do action]` or `if [condition equals true], then [do action], else [do other action].`

The condition control in Power Automate functions in the same way. You can control the set of actions a flow will execute depending on the values present in the flow.

Power Automate conditions can perform several different types of evaluations, as shown in the following list of operators:

- **contains**
- **does not contain**
- **is equal to**
- **is not equal to**
- **is greater than**
- **is greater than or equal to**
- **is less than**
- **is less than or equal to**
- **starts with**
- **does not start with**
- **ends with**
- **does not end with**

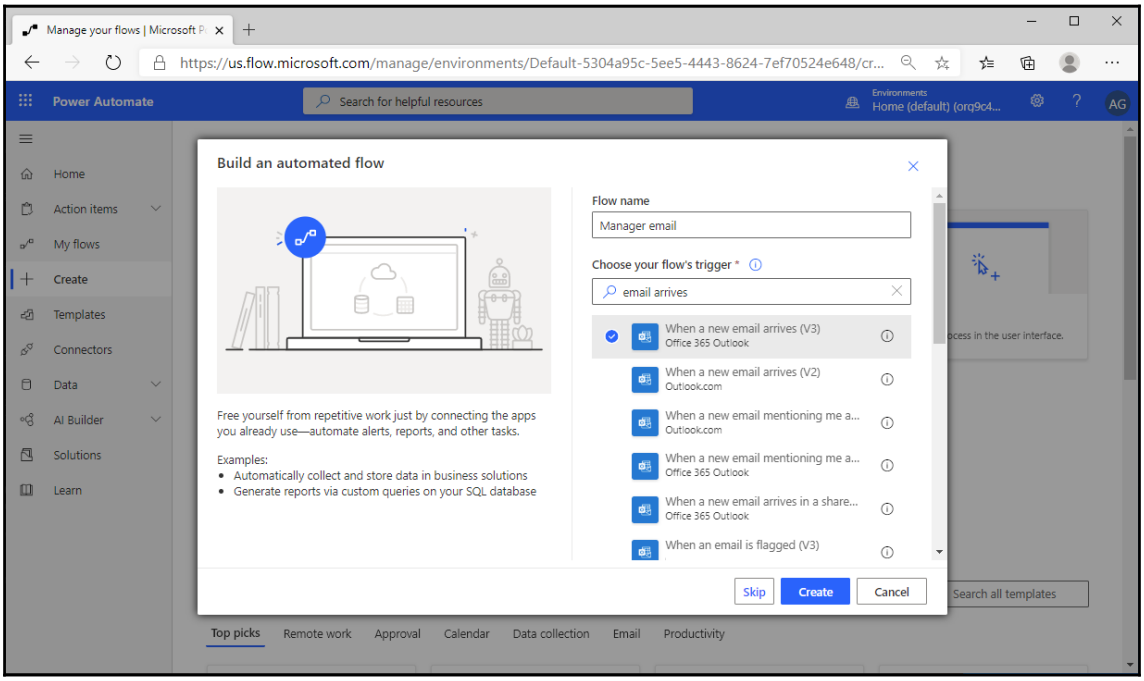
When evaluating values against each other, you'll want to make sure you're using the right operator for the type of data that you're evaluating. For example, you would use operators such as **is greater than** or **is less than or equal to** when comparing or testing numeric values.

Creating the flow

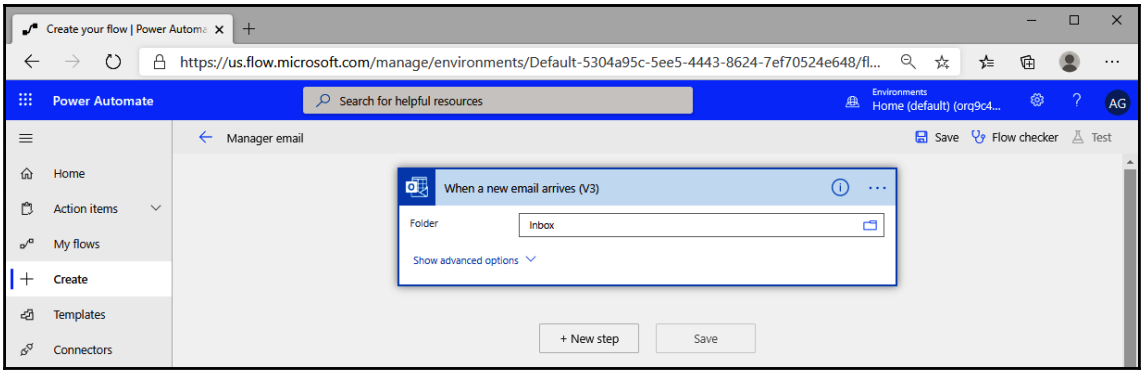
In this example, we'll use a condition to control when a notification will be sent. Follow these steps to see how the condition control is used:

1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **+ Create** and select **Automated flow**.

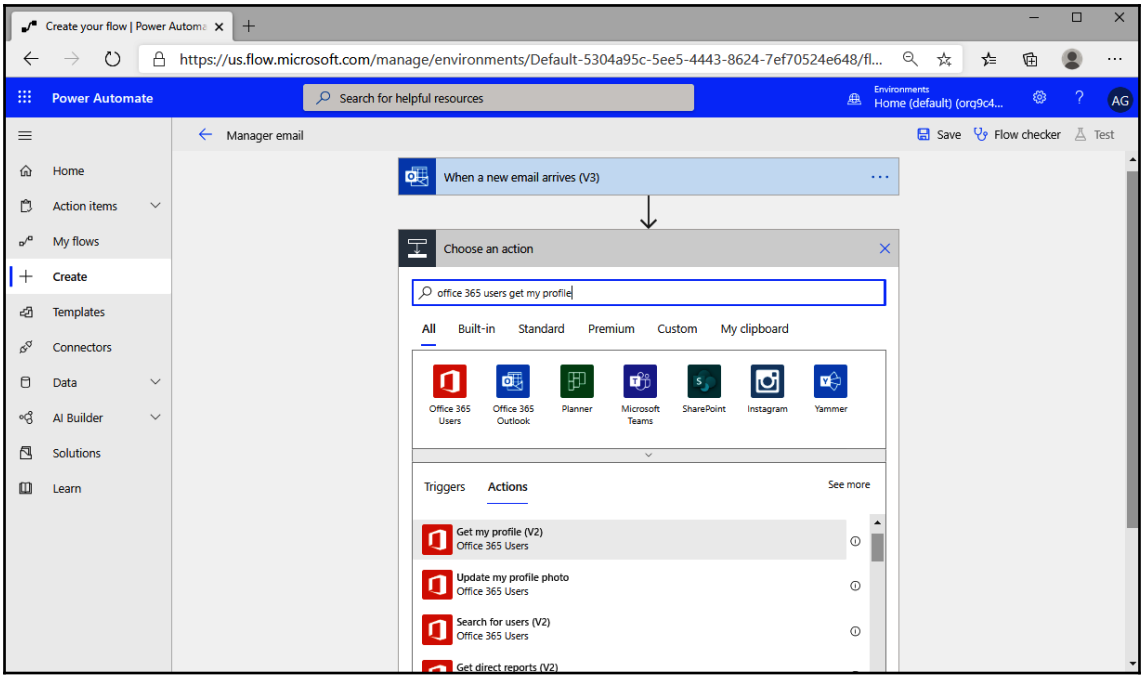
2. Enter a name and then select the **When a new email arrives (V3)** trigger. Click **Create**:



3. Click **+ New step**:



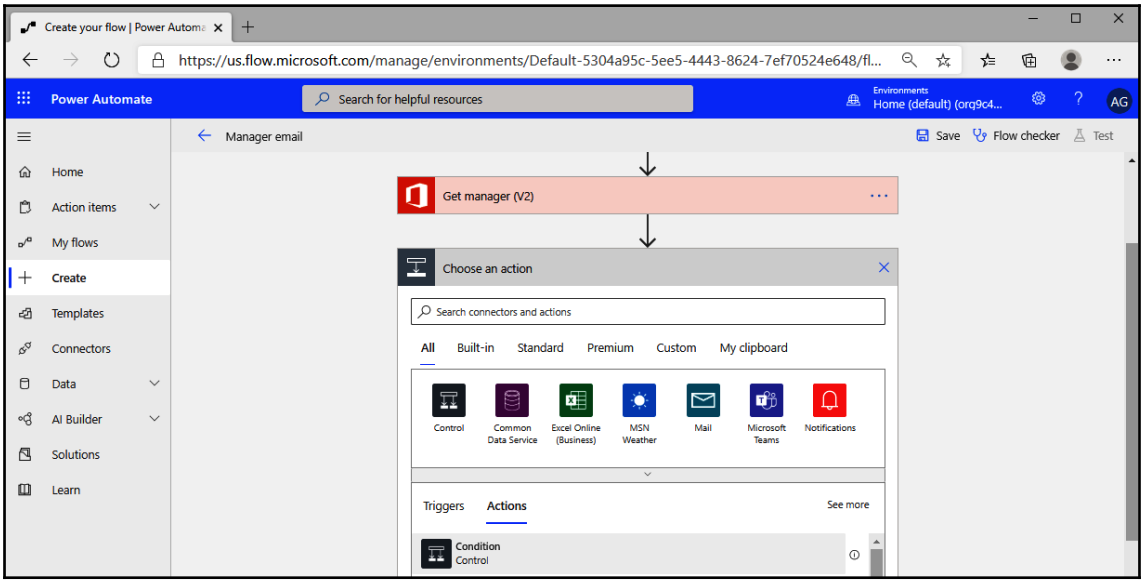
4. In the **Search connectors and actions** box, search for the `Office 365 Users Get my profile (V2)` action and select it:



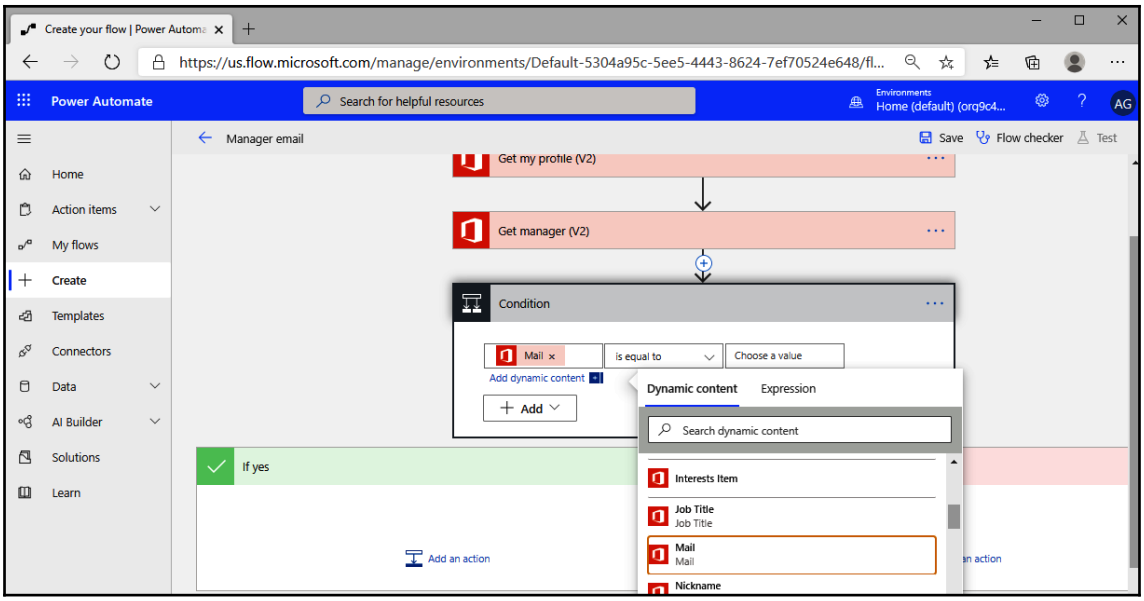
5. Click **+ New step**.
6. In the **Search connectors and actions** box, search for the `Office 365 Users Get manager` action and select it.
7. In the **User (UPN)** box of the **Get manager (V2)** action, select the **User Principal Name** value under the **Get my profile (V2)** section.
8. Click **+ New step**

.

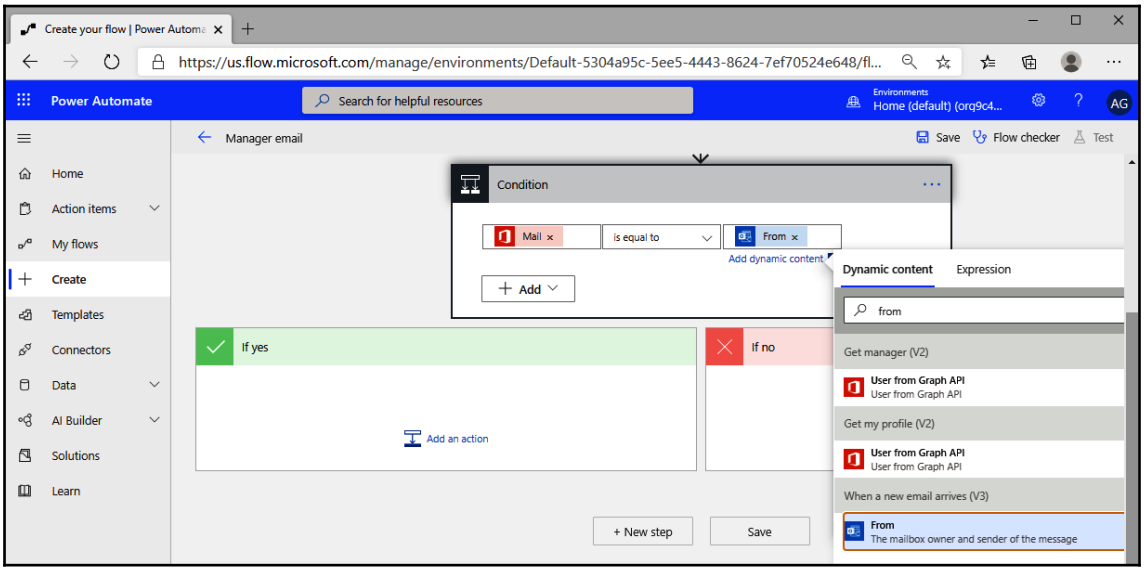
9. In the **Search connectors and actions** box, search for the `Condition` action and select it:



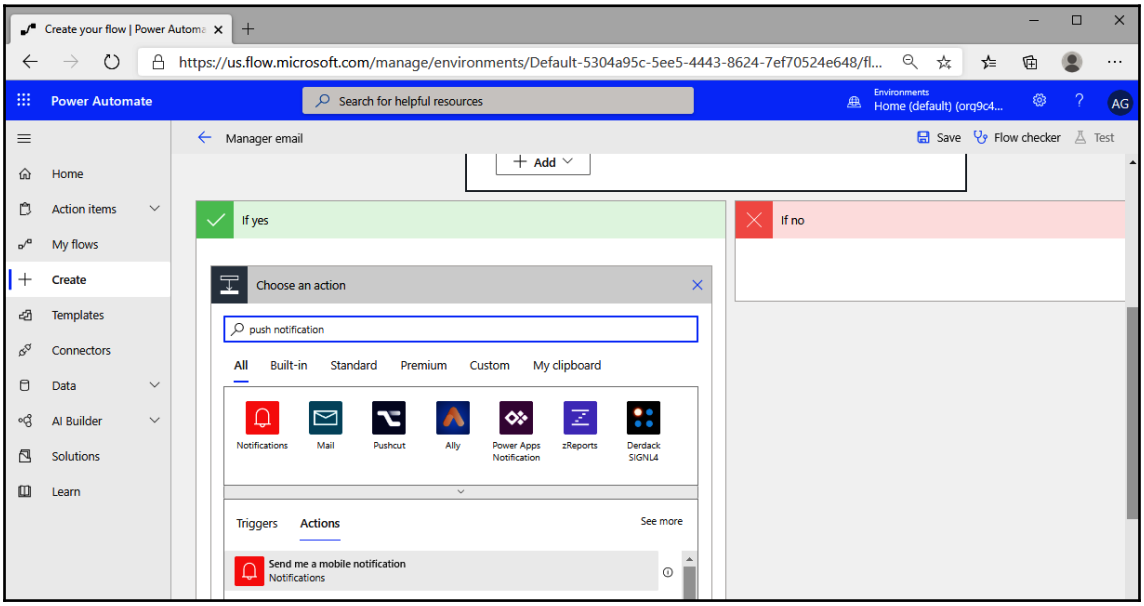
10. The goal of this step is to check to see whether the value of the email sender matches the result of the **Get manager (V2)** action. In the **Choose a value** box on the left side of the condition, select the **Mail** value under **Get manager (V2)**:



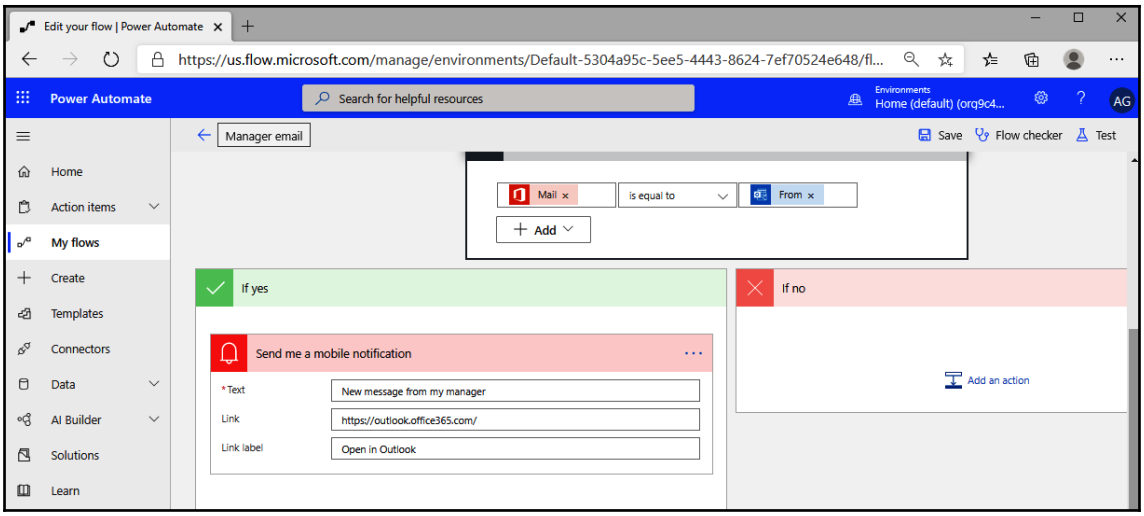
11. In the **Choose a value** box on the right side of the condition, select the **From** dynamic content value under **When a new email arrives (V3)**:



12. Under the **If yes** branch, select **Add an action**.
13. In the **Search connectors and actions box**, select **Send me a mobile notification**:



14. Customize the notification text and URL. Depending on which platform you are creating a notification for, you may be able to create a deep link, which enables you to open a related item or message directly. In this example, we're not going to use a deep link and will simply use the URL `https://outlook.office365.com` as the destination:

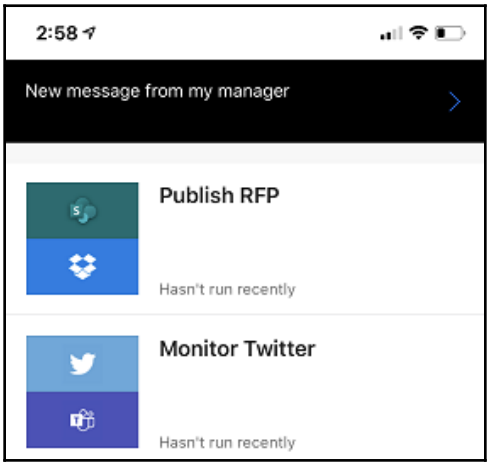


15. Click **Save**.

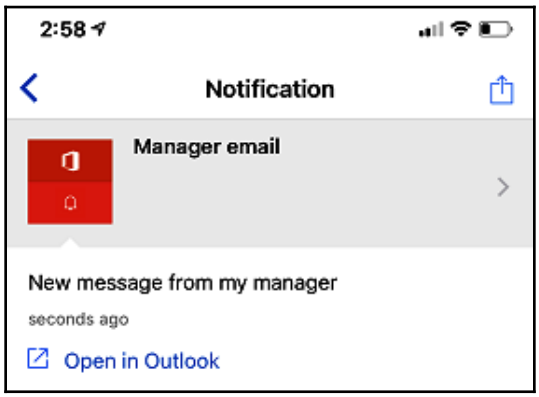
The flow has been created. You can now wait for (or ask) your manager to send you an email to trigger the notification.

Reviewing a notification

When you receive a message that meets the conditions and triggers the notification, you should get a push notification on your device, as shown in the following screenshot:



You can tap on the notification banner to be taken to the Power Automate mobile app notifications page:



From here, you can use the **Open in Outlook** link (or whatever text you specified in the push notification link text) to open the URL/application specified.

Summary

In this chapter, you learned about creating flows that utilize push notification actions. Push notifications can be useful for drawing a user's attention to a particular event or item that requires prompt action.

You also started learning about the power of conditions and how they can be used to enable flows to perform different actions based on the data that is received. Conditions are necessary to ensure that processing only happens when certain values are detected. Conditions are part of the Power Automate system itself, and as such, can be used with any connectors. Conditions can use expressions and dynamic content as well, just like other actions in a flow.

In the next chapter, you'll learn about sharing flows. Sharing flows will allow your team members to build on each other's work, increasing both individual and group productivity.

7
Working with Team Flows

Up to this point, all of the flows that you have created or seen have been associated with your particular Office 365 user account. These types of flows are individualized and may rely on your identity to authenticate.

However, if you are part of a team or share responsibility for a work process, you may find it necessary to create flows that others can also see and manage. This is where the *team flows* fit in.

In this chapter, you'll learn the basics of team flows:

- Understanding team flows
- Sharing a flow with your team
- Sharing a flow with run-only permissions
- Managing team flows

Team flows are important to enable business continuity in the event that one or more parties of a flow changes roles or leaves the organization altogether. By the end of this chapter, you'll know how to share and manage team flows.

Let's begin!

Understanding team flows

As mentioned in the introduction, team flows enable multiple individuals to manage the activity and configuration of a flow. Team flows also enable multiple individuals to view the run history and provide credentials for connectors.

There are, however, some interesting features and caveats team flows:

- Other users, groups, or SharePoint lists can be made co-owners of a flow. The creator of the flow *cannot* be removed by another co-owner. If you are granting access to a SharePoint list, that site must be connected to the flow (such as being used to save a file).
- Co-owners of a flow can view a flow's run history.
- Co-owners of a flow can add or delete actions or conditions.
- Co-owners of a flow can manage the properties of a flow (for example, whether it's enabled or disabled, descriptions, and other general properties).
- Co-owners can delete a flow.
- Connections that are part of a flow can only be used in the context of the shared or team flow; they can't be used by other co-owners in their own flows.

Any flow can be made into a team flow. No special design considerations are necessary.

Credential and connection information are part of the team flow, so it may be possible for others to update the flow in such a way as to gain access to other information. When using shared or team flows, you should ensure that the account and connection configurations only have access to the minimum amount of data necessary to process the flow.

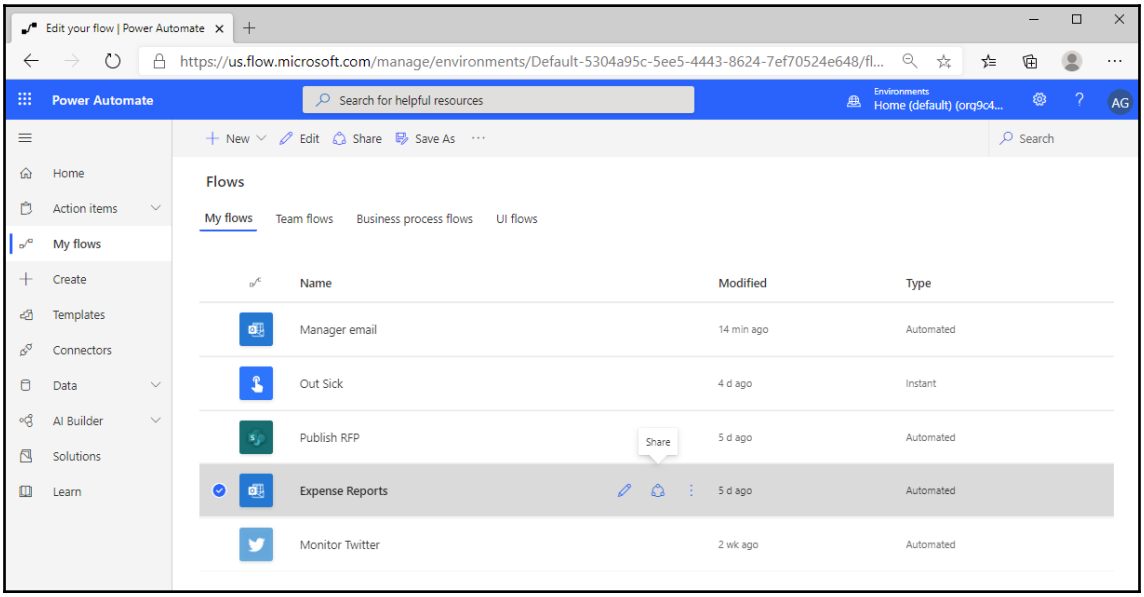
Next, we'll look at converting a standard user flow into a team flow.

Sharing a flow with your team

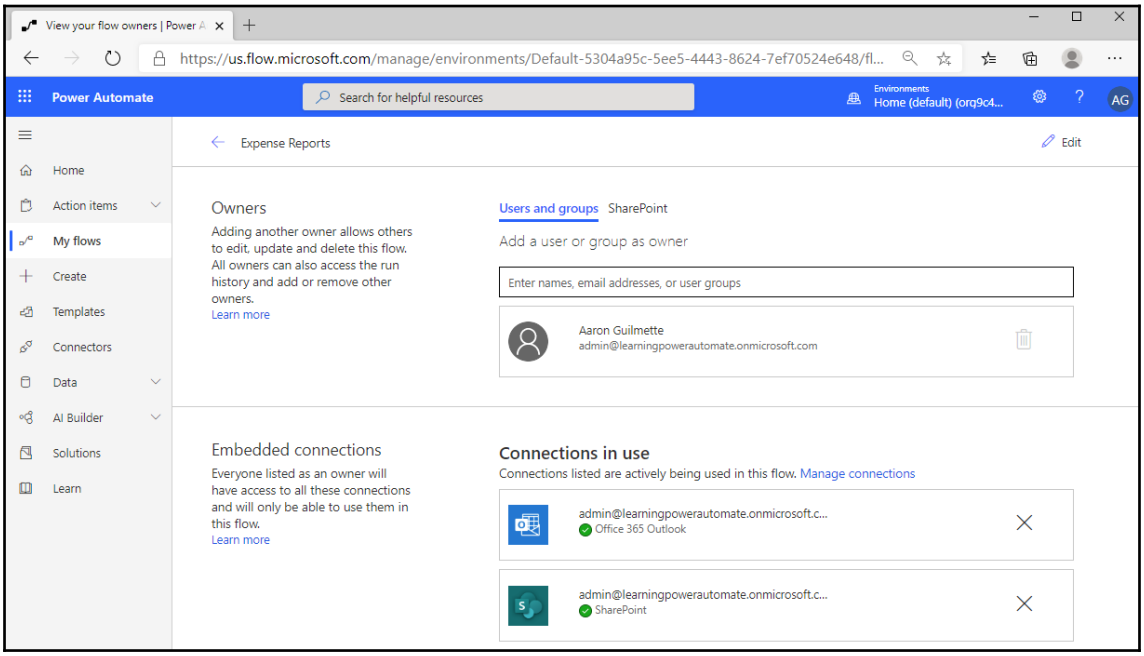
Now that you've got an understanding of what additional features a team flow provides, let's work on sharing a flow with some team members.

In this example, we're going to take the Expense Reports flow we created in `Chapter 3`, *Working with Email*, and update it to be a team flow:

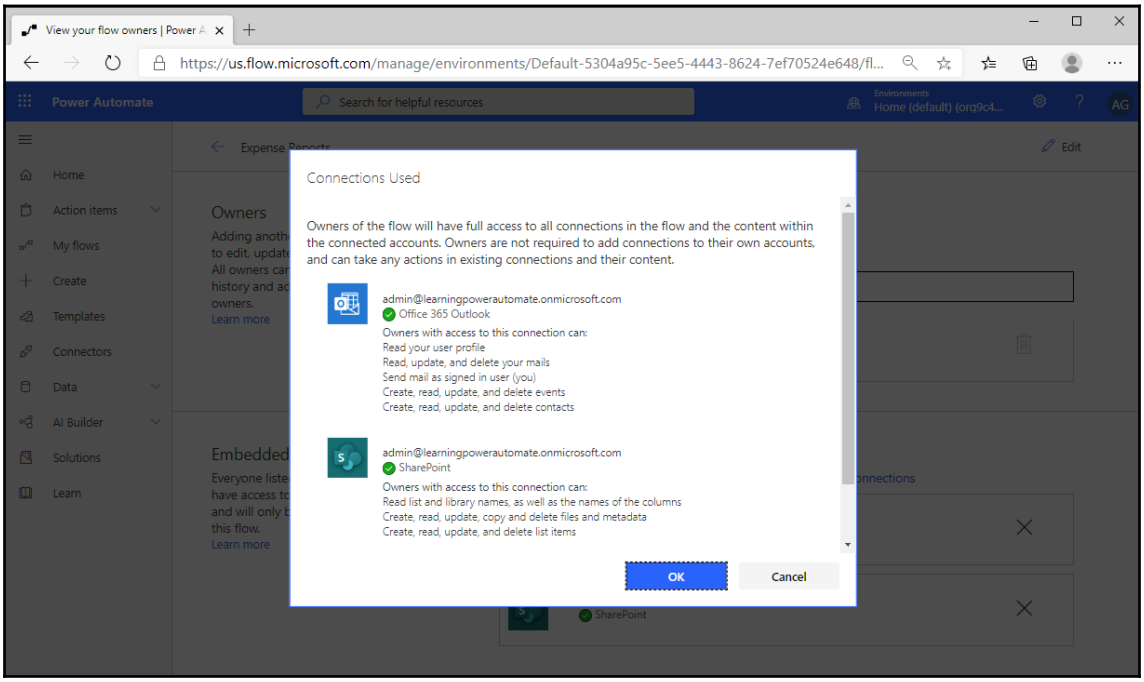
1. Log in to the Power Automate web portal (`https://flow.microsoft.com`) and select **My flows**.
2. Select the **Expense Reports** flow you created in `Chapter 3`, *Working with Email*, and then click the share icon. You can also click the ellipsis and select **Share** from the context menu:



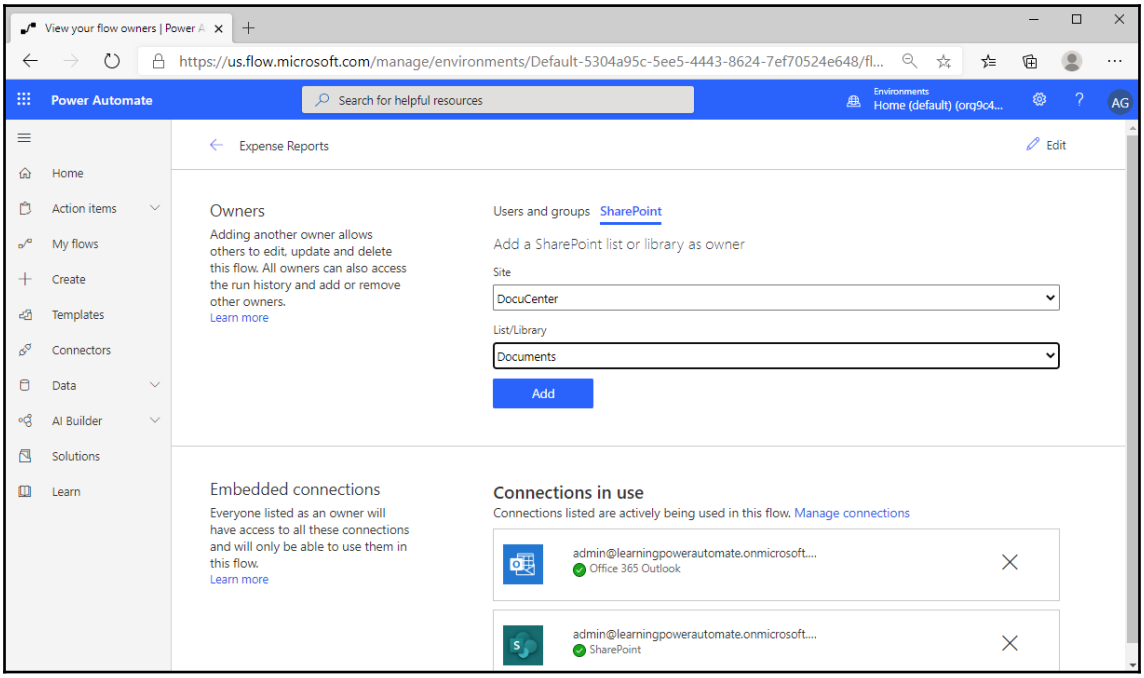
3. Under **Users and groups**, enter the name or address of a user or group to whom you will grant access. Click **Add**:



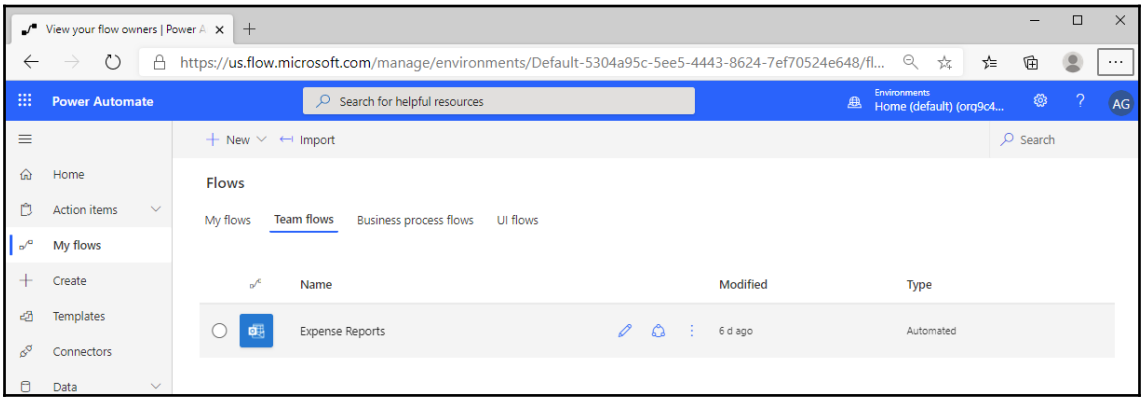
4. When adding new members, you'll be prompted to click **OK** to confirm the permissions:



5. If your flow is connected to a SharePoint site, you can grant permissions to users who have edit access to a SharePoint list or library. Select **SharePoint** and then select the connected site and list or document library. Click **Add**:



6. The flow will be moved from the **My flows** view of **My flows** to the **Team flows** view of **My flows**, as shown in the following screenshot:



The flow has been shared.

In some cases, you may not wish to share a flow with full co-owner permissions. Next, we'll look at how to share with run-only permissions.

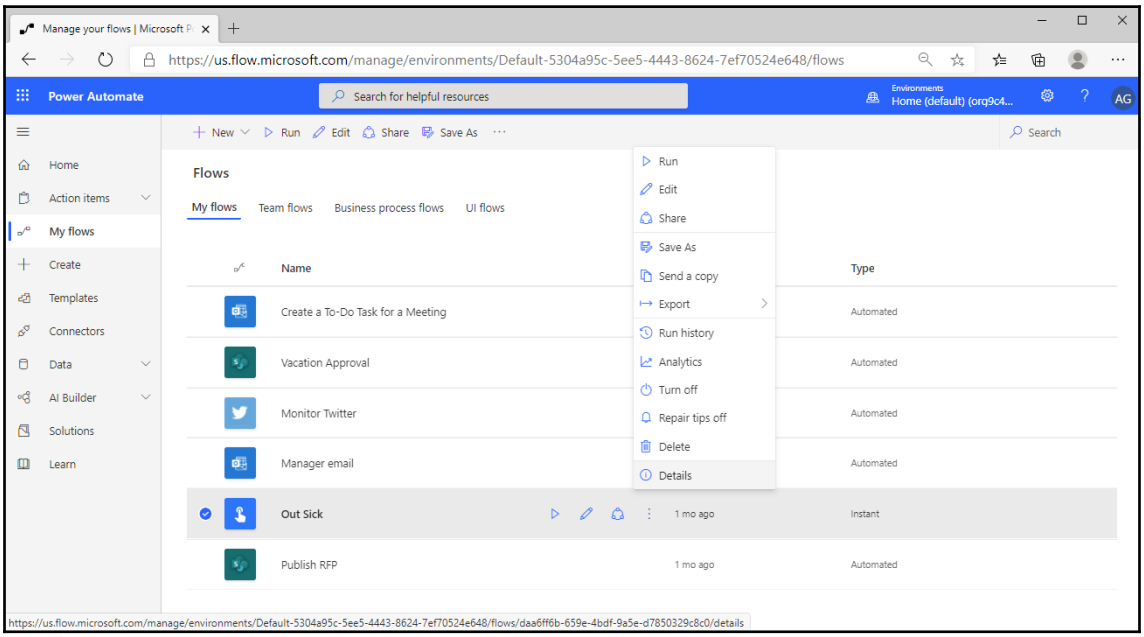
Sharing a flow with run-only permissions

The **run-only permissions** feature allows you to grant a very limited set of permissions to manually-triggered flows (such as button and instant flows). With this option, users can only execute the flow. They cannot edit or modify any part of the flow and will only be granted permission to trigger it.

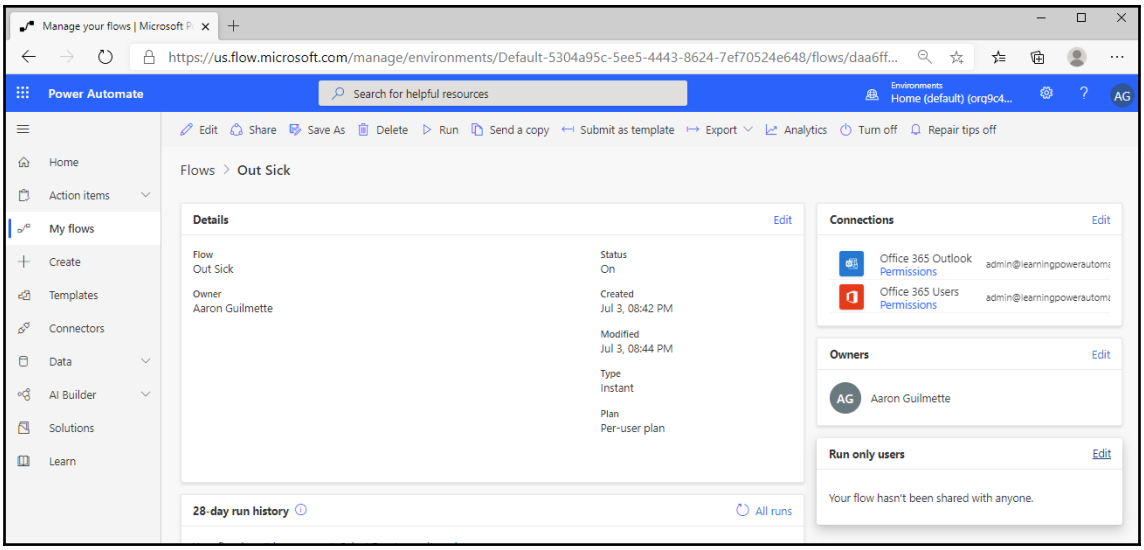
To grant run-only permission to a button or instant flow, follow these steps:

1. Log in to the Power Automate web portal (`https://flow.microsoft.com`) and select **My flows**.

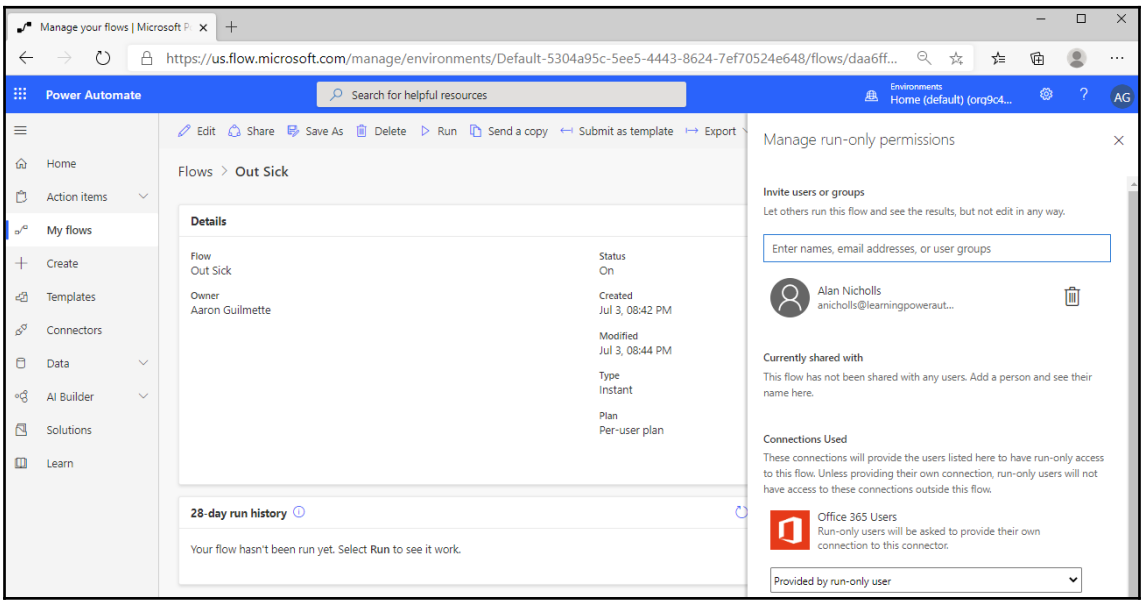
2. Select the ellipsis for a button or instant flow, such as the **Out Sick** instant flow you created in `Chapter 5`, *Creating Button Flows*. Select **Details** to open the **Details** page:



3. In the **Run only users** section, click **Edit**:



4. In the **Manage run-only permissions** panel, add the user or group for which you want to grant run-only access. For connections that require credentials to data sources, the run-only user will need to provide them. The credentials provided are only in the context of this flow and not tied to any other flows a user may have:



5. Scroll to the bottom of the panel and click **Save**.
6. The **Details** tab will show the updated configuration with run-only users.

The flow has been configured with run-only permissions for the specified users and groups.

When a flow is configured with just run-only permissions, it is not shown under the **Team flows** tab. Only flows that have been configured with co-owners will be moved to the **Team flows** tab.

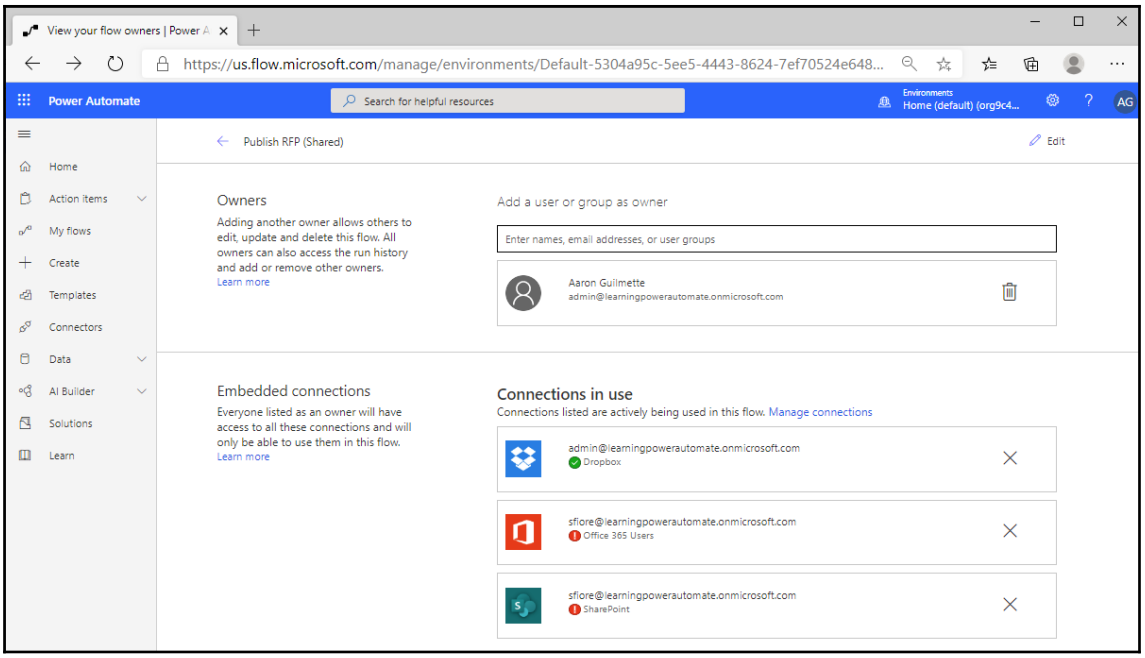
Next, we'll look at some tasks related to managing team flows.

Managing team flows

All co-owners of a flow can manage its various properties. In this exercise, we'll look at how a team flow looks to a sharing recipient.

Once a flow has been shared with you, you can manage it like any other flow using the **My flows** | **Team flows** page of the Power Automate web portal. One of the most important changes comes when the *creator* of the flow is no longer available. After the account of a creator has been removed from the organization, connections that utilize that credential will need to be updated.

In the following screenshot, notice how the status of the Office 365 Users and SharePoint connections are displayed with a red exclamation mark to denote failing credentials:

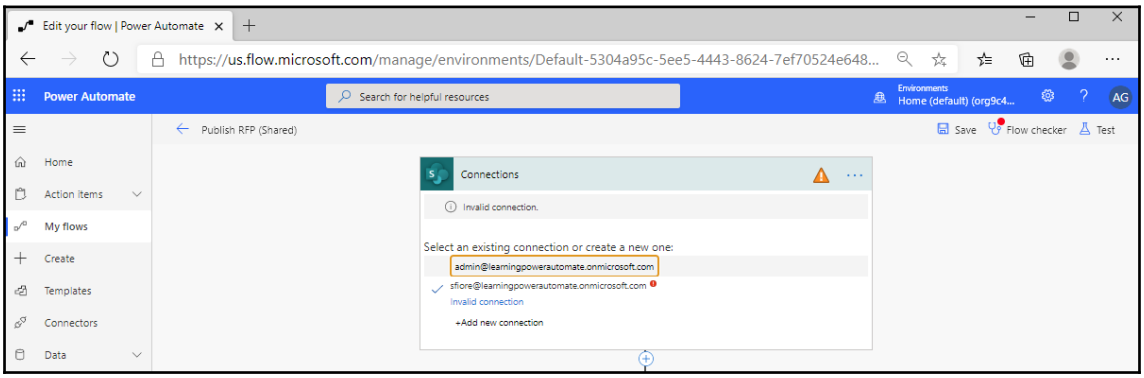


To resolve the situation and update the credentials, follow these steps:

1. In the **Embedded connections** section, select **Manage connections**.
2. Select the account in error.
3. Click the ellipsis and then click **Switch account**.
4. Select a new account (or enter credentials).

You may also edit the flow to update the credential using this procedure:

1. From the **Team flows** page, select the flow to be updated and click the pencil icon.
2. Select the object in error to expand the connection selection:



3. Select a connection to use or click **+Add new connection** to create a new authenticated connection.
4. Save the flow.

You'll need to repeat this unique process whenever the account of a flow's creator is removed. This credential and authentication management process is important to successfully ensure team flows continue to work.

At the time of writing this, once a flow has been shared, co-ownership or sharing information can be removed, but the flow will stay in the **Team flows** section of the portal.

Summary

In this chapter, we introduced the concept of team flows. Team flows allow you to share the management and ownership of a flow in your organization. Team flow owners can manage all aspects of the flow. You also learned how to manage the credentials of a team flow after the owner's account is removed from the Microsoft 365 organization. This process is critical to allow organizations to continue running shared flows.

In the next chapter, we're going to expand on using conditions in flows.

3
Section 3 - Intermediate Flow Concepts

Using the knowledge and skills acquired in *Section 2*, the reader will start using intermediate techniques to create slightly more complex solutions.

This section comprises the following chapters:

- `Chapter 8`, *Working with Conditions*
- `Chapter 9`, *Getting Started with Approvals*
- `Chapter 10`, *Working with Multiple Approvals*
- `Chapter 11`, *Posting Approvals to Teams*
- `Chapter 12`, *Using a Database*
- `Chapter 13`, *Working with Microsoft Forms*
- `Chapter 14`, *Accepting User Input*

8
Working with Conditions

In `Chapter 6`, *Generating Push Notifications*, you were introduced to the concept of **conditions**. Conditions can be used to evaluate parameters and data inside a flow to inform a decision. Conditions are an important part of building more complex flows.

Conditions, as you'll see in this chapter, can be used to expand the capability of flows or potentially combine multiple single flows into a larger, more robust flow. In this chapter, we're going to expand your understanding of conditions with the following topics:

- Understanding condition operators
- Using expressions and multiple conditions

This chapter will give you the opportunity to see how conditions can fine-tune your flows.

Let's go!

Understanding condition operators

Operators indicate the types of calculations used when evaluating components. You might be familiar with common mathematical operators such as the following:

- **+** (addition)
- **-** (subtraction)
- ÷ (division)
- × (multiplication)
- **<** (less than)
- **>** (greater than)

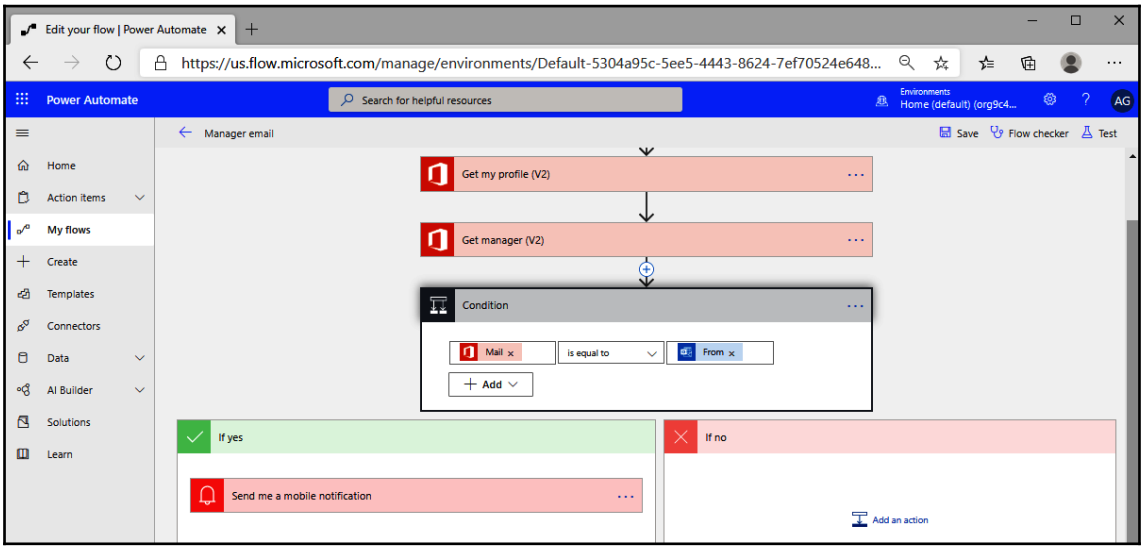
These operators are used for evaluating numeric values. When manipulating objects and text in a flow, simply using mathematical operators may not be sufficient. While you can use numeric expressions in some places, it's also important to understand other operators that can be used to determine whether a value meets a certain condition.

Power Automate conditions can perform the evaluation of conditions using the following operators:

- **contains**
- **does not contain**
- **is equal to**
- **is not equal to**
- **is greater than**
- **is greater than or equal to**
- **is less than**
- **is less than or equal to**
- **starts with**
- **does not start with**
- **ends with**
- **does not end with**

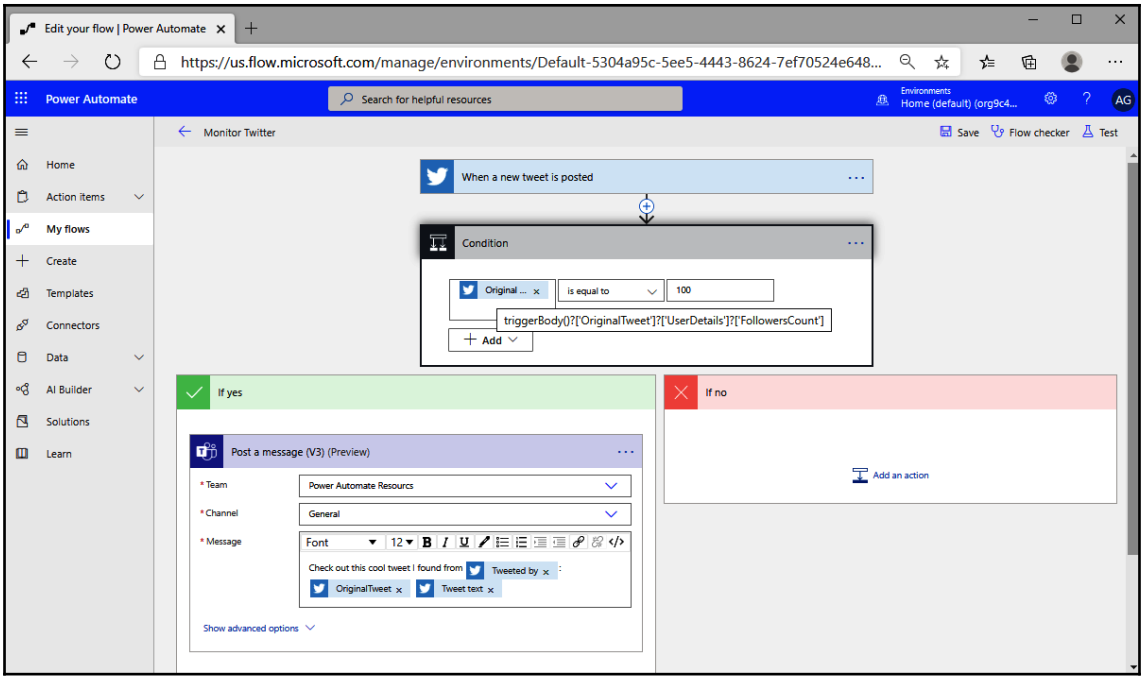
Some of these can function with numerals (such as **is greater than or equal to**), while some work with alphabetical characters (such as **starts with**).

Reviewing the flow you created in `Chapter 6`, *Generating Push Notifications*, you can see how conditions can be used to compare dynamic content values:



In this case, we didn't want to send a mobile notification *every* time we received an email – we only wanted to generate a notification if the user's manager value was the same as the email sender's, effectively putting a condition or filter on what would generate a notification. This was accomplished with the **is equal to** condition, which can be used to evaluate strings or numerals.

In the following example (based on the *Monitor Twitter* flow from `Chapter 2`, *Getting Started with Power Automate*), you can see that we have updated it. The modified flow now includes a condition that evaluates whether the account that posted the tweet has more than 100 followers:



As you can see, evaluating text or numeric strings is relatively straightforward with a condition.

Next, we'll look at using an expression as part of a condition's equation.

Using expressions and multiple conditions

Perhaps evaluating conditions based on static values (such as the sender of an email message) doesn't provide the granularity or flexibility that a flow requires. In that case, you can also use expressions in a condition.

You've already seen examples using expressions as part of dynamic content, such as using `formatDateTime()` and `utcNow()` in `Chapter 4`, *Copying Files*. In that example, expressions were used to help generate the value for the folder name to store an expense report.

You can also use expressions or functions as part of a condition.

Conditions that perform more than one evaluation are commonly referred to as **advanced conditions**. You can simply use the **+Add** button, and continue adding criteria. When adding multiple criteria to evaluate, there are two core operators available: **AND** and **OR**. Selecting **AND** will require all conditions to evaluate as `true` in order for the control to evaluate as `true`. Selecting **OR** will only require one of the conditions to evaluate as `true` in order for the whole condition control to evaluate as `true`.

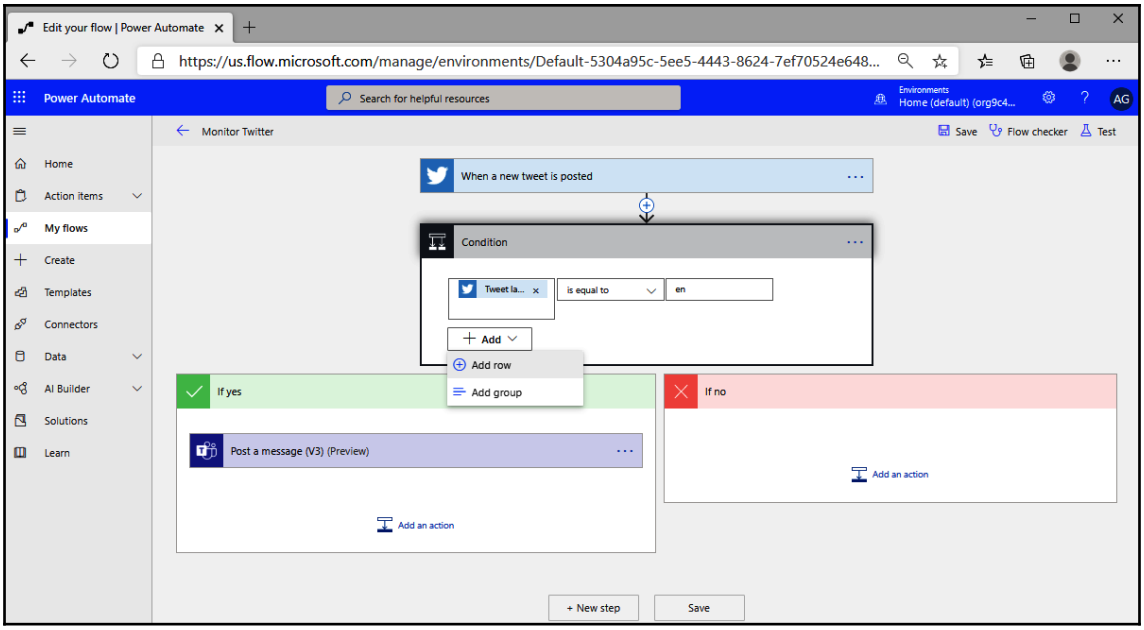
In the following sections, we'll look at two scenarios for creating an advanced condition utilizing multiple criteria. The first will use multiple conditions together, and the section will use condition groups.

Adding multiple conditions

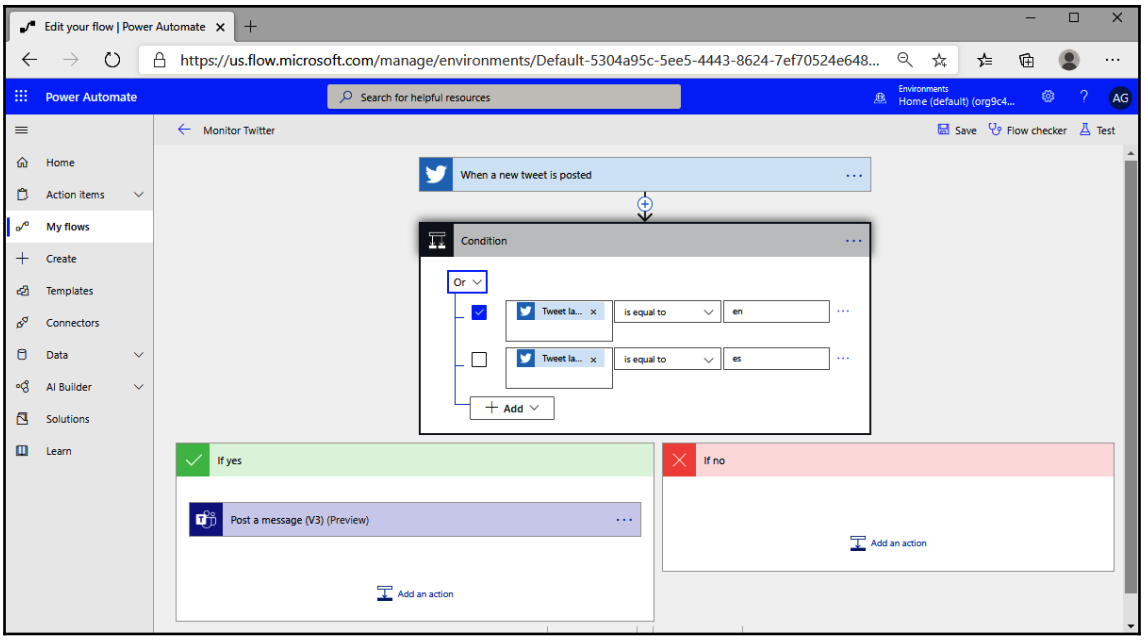
The following example shows adding a condition to evaluate whether a tweet's language is in either English or Spanish:

1. From the Power Automate portal, select **My flows** and then select the **Monitor Twitter** flow for editing.
2. Add a step and select **Condition**.
3. From the **Dynamic content** menu, select the **Tweet language** token, **is equal to** operator, and then enter `en` as the value. This adds English as a condition in order for the control to evaluate as true.

4. Select the **+Add** icon, and then click **+Add row**:



5. Select the **Tweet language** dynamic content token, the **is equal to** operator, and then enter `es` as the value (or another two-digit language code). This adds Spanish as a condition in order for the control to evaluate as true. At this point, the **Tweet language** must evaluate to *both* English and Spanish in order for the condition to be true. While this is a valid configuration from a syntax perspective, it will result in no content being able to pass the condition.
6. Select the drop-down box under the **Condition** header and then select the **Or** operator:



At this point, a tweet will need to be posted where the language is *either* English *or* Spanish in order for the condition to evaluate as true.

In the next section, we'll introduce the concept of condition groups and how they can be used to further control the processing of a flow.

Adding condition groups

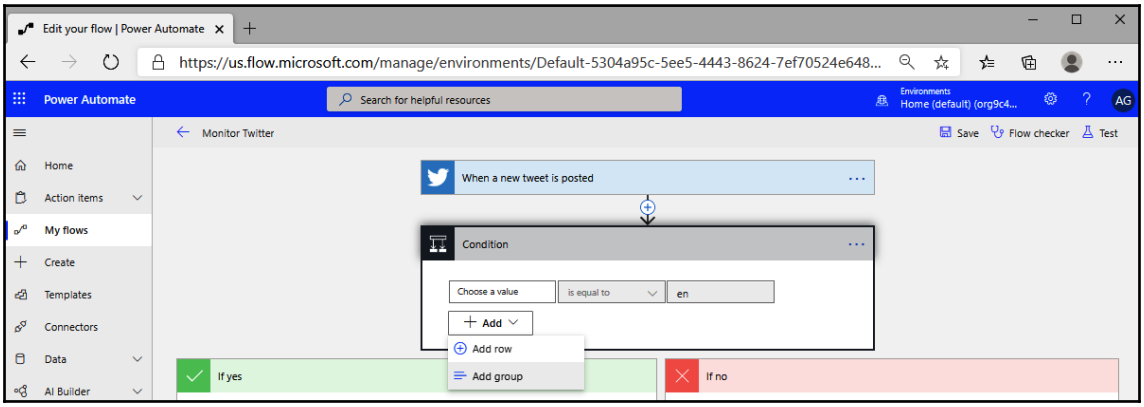
In the previous example, we configured a condition where only one criterion needed to be met in order to process the flow. However, there may be scenarios where you need to select multiple values from one or more groups of conditions. This is where *groups* fit in. Groups allow you to specify one or more conditions from one or more groups that are met in order to satisfy the control.

In this example, we'll reconfigure the flow to meet one of each of the following conditions:

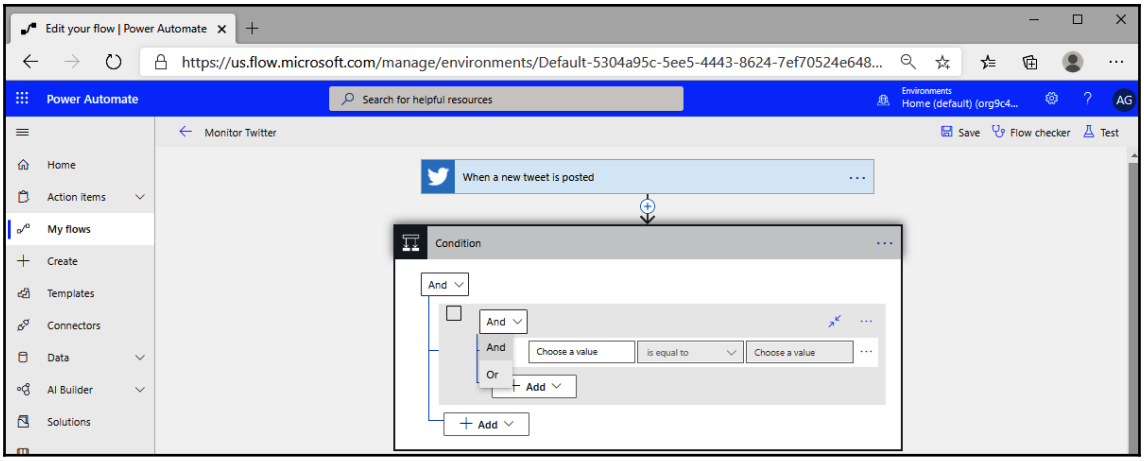
- The tweet language is either English or Spanish.
- The account tweeting must have accumulated more than 100 followers, 100 friends, or favorited more than 100 items.

To accomplish this, we'll create two logical groups, one that will evaluate the languages and one that will evaluate the location. Follow these steps to see how to create condition groups:

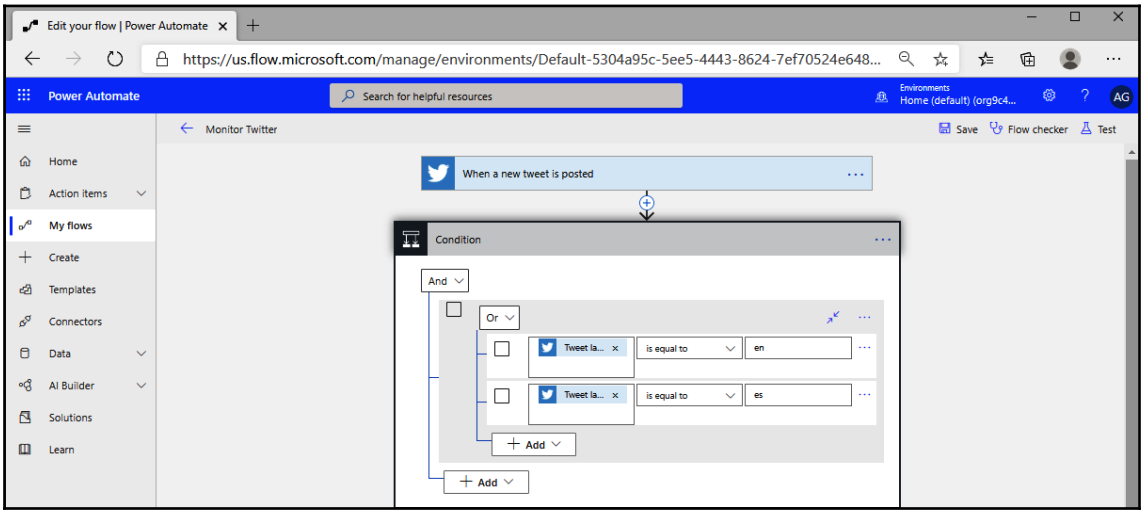
1. From the Power Automate portal, select **My flows** and then select the **Monitor Twitter** flow for editing.
2. Add a step and select **Condition**.
3. From the **Dynamic content** menu, select the **Tweet language** token, the **is equal to** operator, and then enter `en` as the value. This adds English as a condition in order for the control to evaluate as true.
4. Select the **+Add** icon, and then click **Add group**:



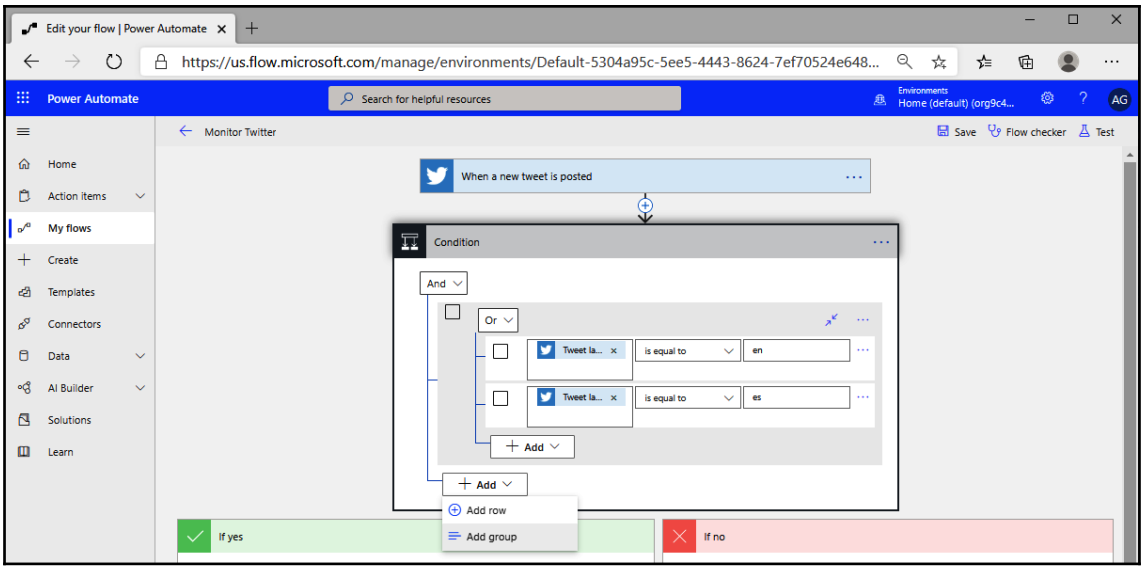
5. In the shaded box, select the inner operator dropdown and then click **Or**. This is necessary to identify that we need to meet only one of these conditions:



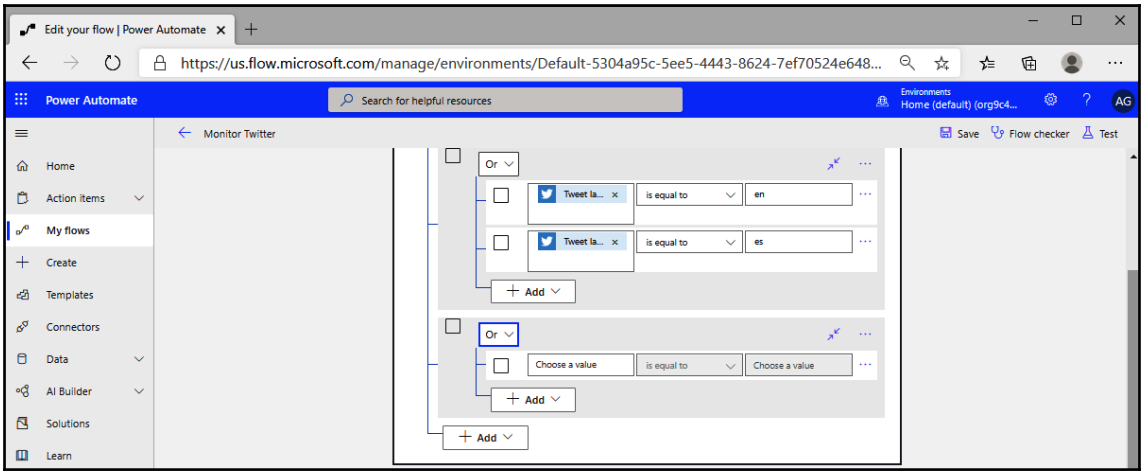
6. Add the first set of conditions (rows) using the **Tweet language** dynamic content tokens, as shown in the following screenshot:



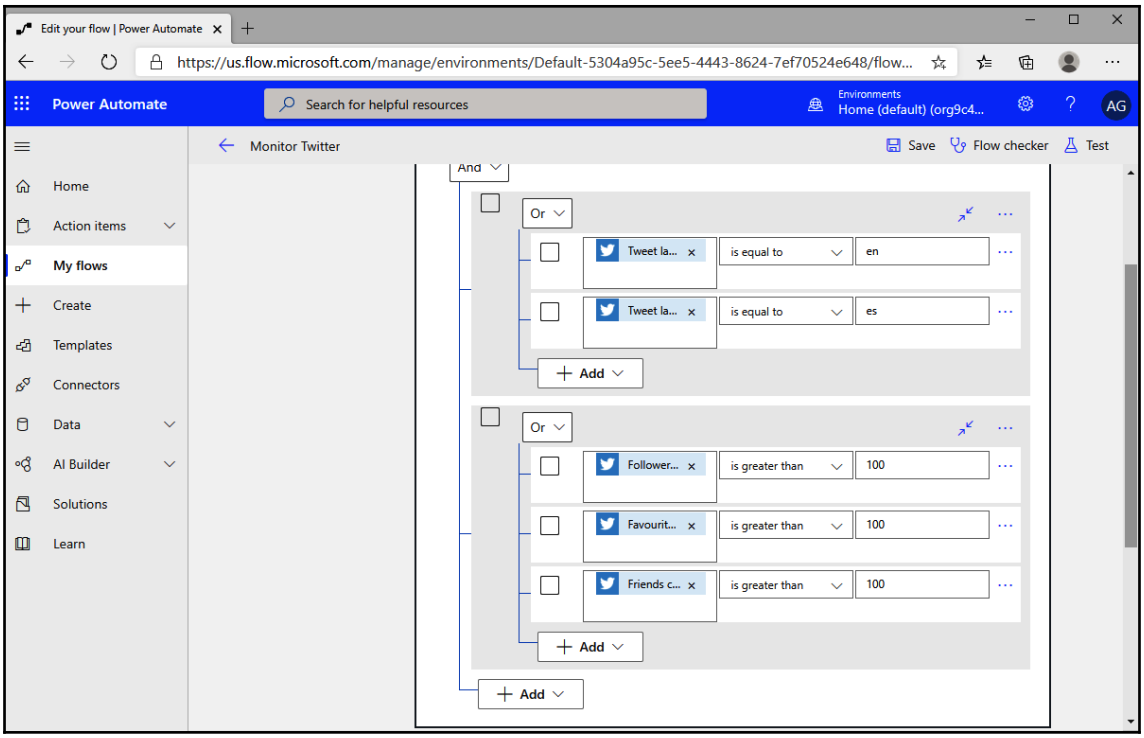
7. Select the outer **+ Add** button at the bottom of the condition control and add a new group:



8. In the new shaded box, click the inner operator dropdown and choose **Or**. This will indicate that the flow needs to evaluate only one of these conditions as true in order to render the whole condition true:

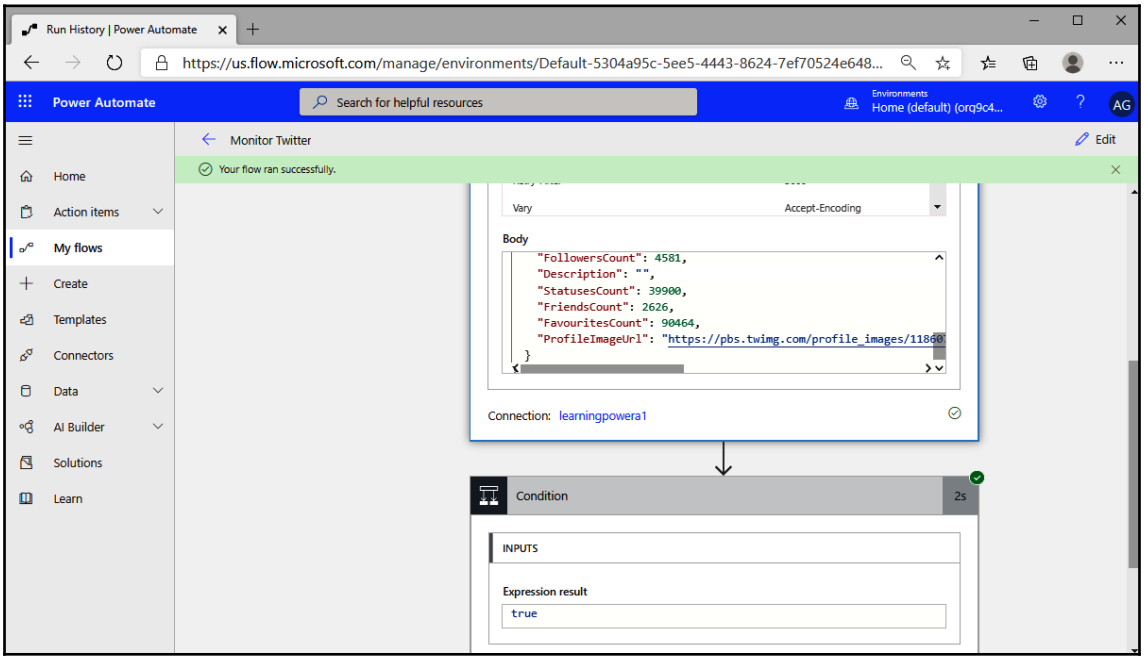


9. Click the inner **+Add** button in this condition box, select the **Location** dynamic content token, and then enter `us` in the value box. Repeat for each additional location that meets the task requirement. The finished result should look similar to the following screenshot:



10. Click **Save**.
11. Using another browser window, prepare a tweet that meets the criteria to trigger the flow.
12. In the Power Automate window, click **Test**, select the **I'll perform the trigger action** radio button, and then click **Test**.
13. Switch back to the previous browser window and send the tweet.

14. Return to the Power Automate window and verify the results by expanding both the trigger and conditions:



As you can see from the previous screenshot, at least one of each condition group was successfully met and the condition evaluated as true, causing the *Yes* branch of the condition to execute.

Summary

In this chapter, you learned how to work with advanced conditions. Advanced conditions allow you to use multiple dynamic content tokens and expressions in order to restrict what data entities are processed by a flow. Using the AND and OR operators, you were able to combine multiple filters and expressions to ensure that only content meeting one of each of the condition groups passed through the flow. Conditional processing is important because it can be used to limit how many flow executions get processed and billed to your account, as well as ensuring that all business criteria or requirements are met prior to execution.

In the next chapter, we'll begin working with approval workflows.

9
Getting Started with Approvals

The flows you've worked with up to this point are largely automated (such as for detecting new files or emails) or triggered by a user action (such as the button flow to send a template email). In this chapter, we're going to begin exploring flows that require additional user action to continue, such as an *approval* flow.

Approval flows require a special service that's used to hold state information pertinent to the flow. **Common Data Service**, or **CDS**, is the foundation for more complex flows such as approvals.

Specifically, this chapter will cover the following topics:

- Understanding Common Data Service
- Creating an approval flow
- Responding to approvals

By the end of this chapter, you'll have an understanding of Common Data Service and will be able to create basic approval flows.

Let's dive in!

Understanding Common Data Service

CDS can be thought of in similar terms to a database. It is comprised of data objects called **entities**. Compared to a database, entities can be described in terms of *database tables*. Whereas database tables have the concept of *columns*, the corresponding object in an entity is called an *attribute*.

CDS has a number of standard entities that cover a lot of usage scenarios. Some entities, such as activities, are multidimensional as well. There are standard entities in several groups or categories, as listed here:

Entity Type	Example Entities	Description
Customer	Account, Contact, CustomerAddress	Entities that are used for managing and referring to customer account information, such as individuals and addresses.
Activity	Email, Task, Fax, PhoneCall, ActivityParty, ActivityPointer	Entities that are used when users perform interactions or activities with customers, such as leaving phone messages or sending emails.
Annotation	Note	Captures note details (text, attachments) related to a record in CDS.
Calendar	Appointment	Calendar entities can be used to keep track of the schedule and availability of a service or resource.
Queue	Public, Private	Queues are used to manage and organize how activities or tasks are processed for a particular customer. Queues are containers for multiple entities, such as tasks.
Subject	Knowledgebase, Incident	Subject entities are used to categorize records, such as sales artifacts, in a hierarchical fashion.
Category	Category	The category entity can be used to tag records to aid in searchability and discoverability within the system.
Feedback	Feedback	The feedback entity enables users to store data related to customer feedback for an object, such as a product or service.
Template	DocumentTemplate, KbArticleTemplate, MailMergeTemplate	Template entities help organizations ensure consistency across other entities created in the system.

Along with these standard entities, the CDS allows for the creation of custom entities to support specific business processes or goals. For a full list of entities and the actions that can be performed against them, see the Entity reference: `https://docs.microsoft.com/en-us/powerapps/developer/common-data-service/reference/about-entity-reference`.

As it relates to an approval flow, CDS is used to store state information in entities throughout the duration of the flow. Without this data source, the approval flow would not have a mechanism to track the status or state of any of the approval branches.

While approval flows rely on CDS to operate, end users are not responsible for interacting with the CDS to create the flows. All of the integration, read, and write operations regarding the CDS are handled internally, without the user needing to work with them.

Next, we'll work through the steps used to create an approval flow.

Creating an approval flow

Approval flows can cover a number of scenarios, such as requesting time away, manager sign-off on an expense report, or business owner sign-off on product marketing materials that are to be published. Approval flows typically fall into two categories:

- **Automated**, such as when a new item is added to a SharePoint document library or list. You might use an automated flow to start the approval process for an expense report, which is triggered when a document is uploaded to a site.
- **Instant**, such as when a user manually starts one from an existing document, list item, or application. You might configure an instant approval to start a process for approving an individual piece of content for external publication.

Regardless of the use case and configuration, an approval flow will require the Approvals connector. The Approvals connector has a number of actions and object types that the flow can interact with, though mostly, we'll be working with *responses* and *outcomes*. You can learn more about the Approvals connector here: `https://docs.microsoft.com/en-us/connectors/approvals/`.

When starting an approval, the workflow process sends an email to the approver and adds an item to the approver's **Approvals** node, under **Action items**. For the requester, the **Approvals** node shows actions that have been initiated or sent. Approvals may be responded to via email or the Approvals action in the Power Automate portal.

In this example, we're going to create a sample approval for a vacation request. This flow has the following prerequisites:

- Azure Active Directory user identities with the **Manager** property need to be populated.
- SharePoint Modern Team Site with a modern list, including two Date/Time columns for **Start** and **End** dates.

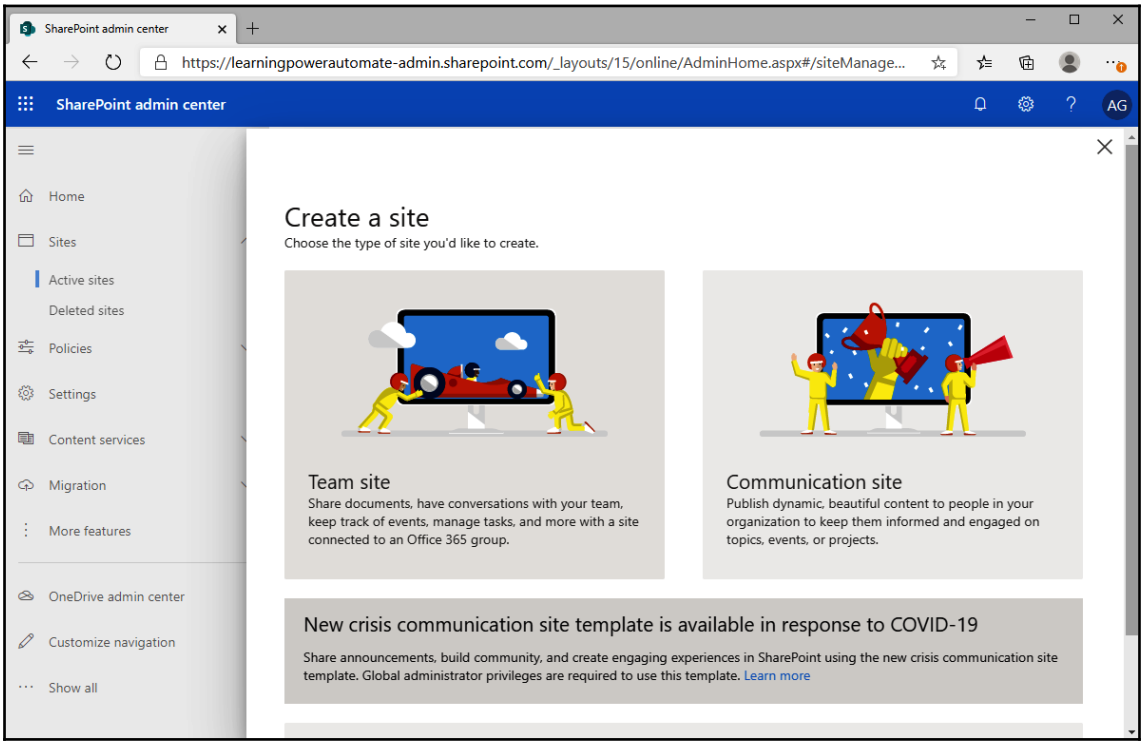
In the upcoming sections, we'll set up an environment that supports the approval and then create the approval flow.

First, we'll create the SharePoint site and list that will be used in the flow.

Creating a SharePoint site and list

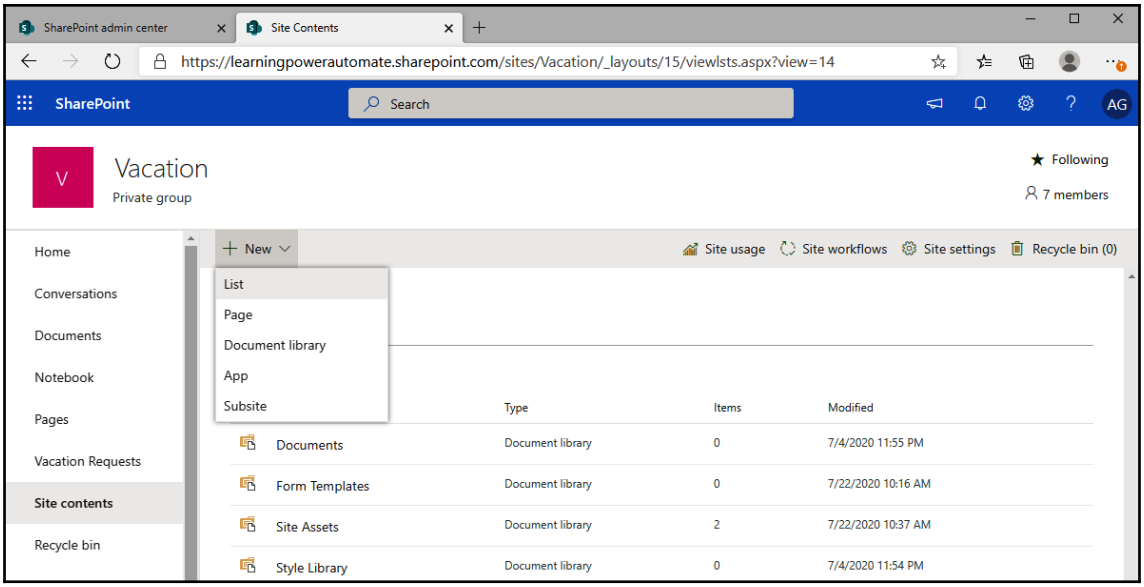
In this instance, the requests will be added to a SharePoint list. You can create a site with a list by following these steps:

1. Navigate to the Microsoft 365 Admin Center (`https://admin.microsoft.com`), log in with an administrative identity, and navigate to the SharePoint Admin Center (**Admin centers** | **SharePoint**).
2. Select **Sites**, and then click **+Create**.
3. Select **Team site**:

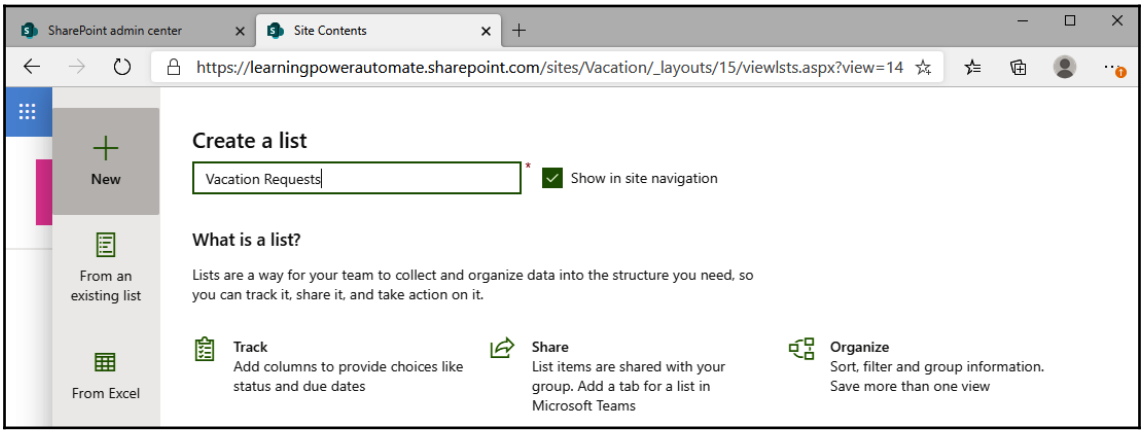


4. Fill in the properties for the name of the site, address, owners, and privacy, and then click **Next**.
5. Add any additional members or owners and click **Finish**.
6. Navigate to the new site.

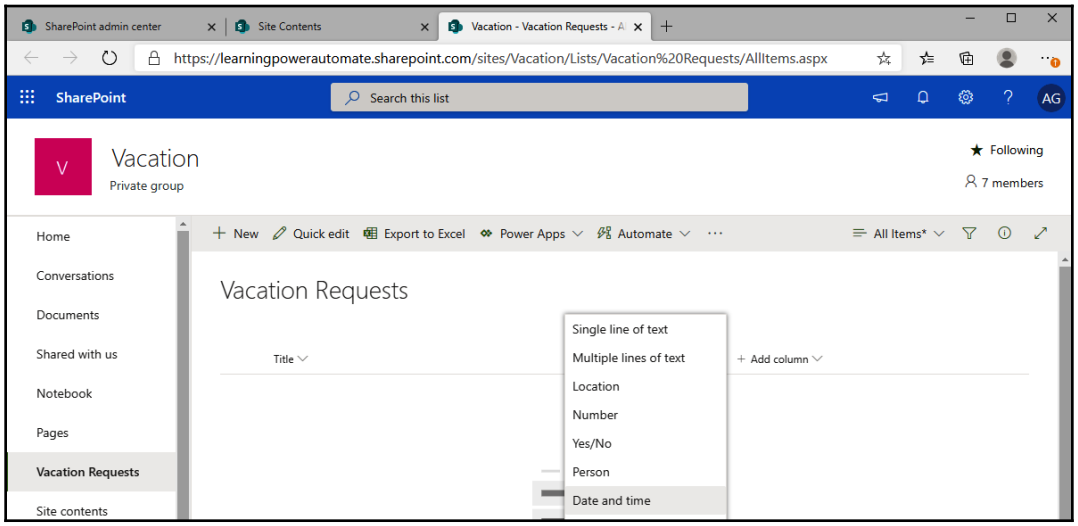
7. Select **Site contents**, and then select **+New | List**:



8. Enter a name for the list, scroll to the bottom of the page, and click **Create**:



9. Click **+Add column**, select **Date and time**, and then create a column for the start date (such as **From**):



10. Add additional columns based on the following table (and any additional columns for details you'd like):

Name	Type	Possible Values	Default Value
To	Date and time	[NA]	[NA]
Status	Choice	Pending, Approved, Denied	Pending
Details	Multiple lines of text	[NA]	[NA]

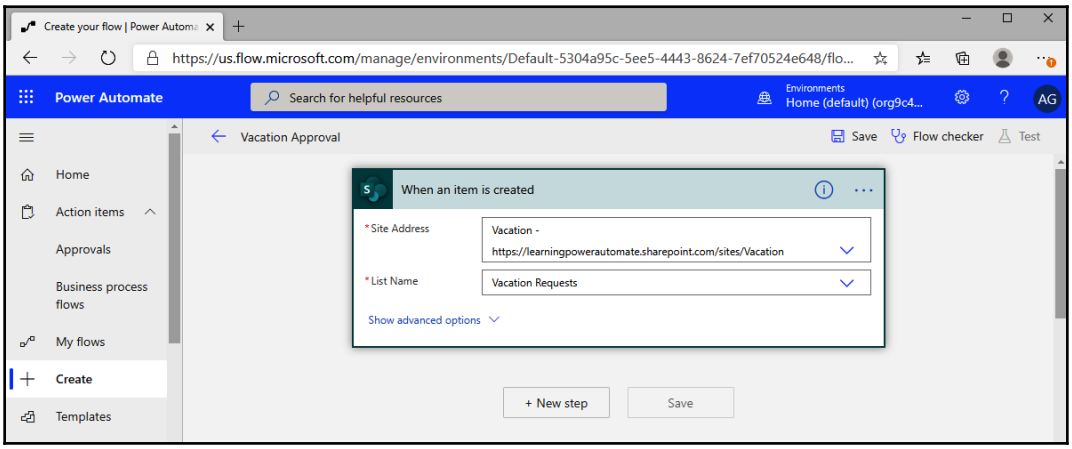
Your SharePoint site and list are now ready to take vacation requests. Next, we'll work on creating the actual approval flow.

Creating an approval

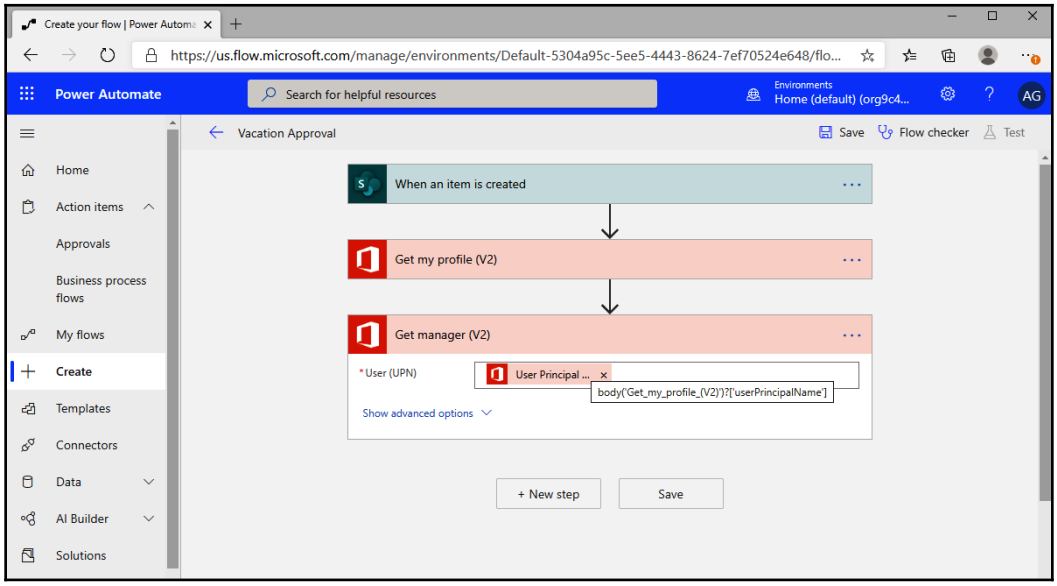
Once the SharePoint site and list have been created, we'll need to create the actual approval that will process requests as they appear on the list. Follow these steps to create the approval flow:

1. Log into the Power Automate web portal (`https://flow.microsoft.com`). Click **+ Create** and under **Start from blank**, select **Automated flow**.
2. Enter a flow name, such as `Vacation Approval`. Search for the **When an item is created** SharePoint trigger. Select it and click **Create**.

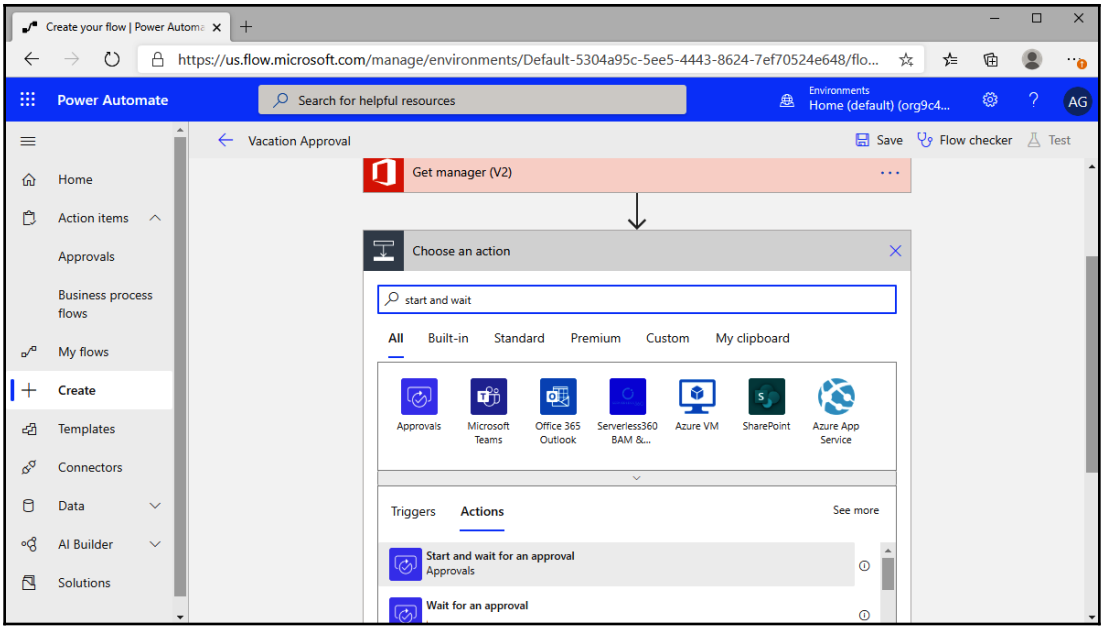
3. Inside the **When an item is created** trigger, select the SharePoint site you created in the prerequisite step for the **Site Address** value and select the SharePoint list you created in the prerequisite step for the **List Name** value:



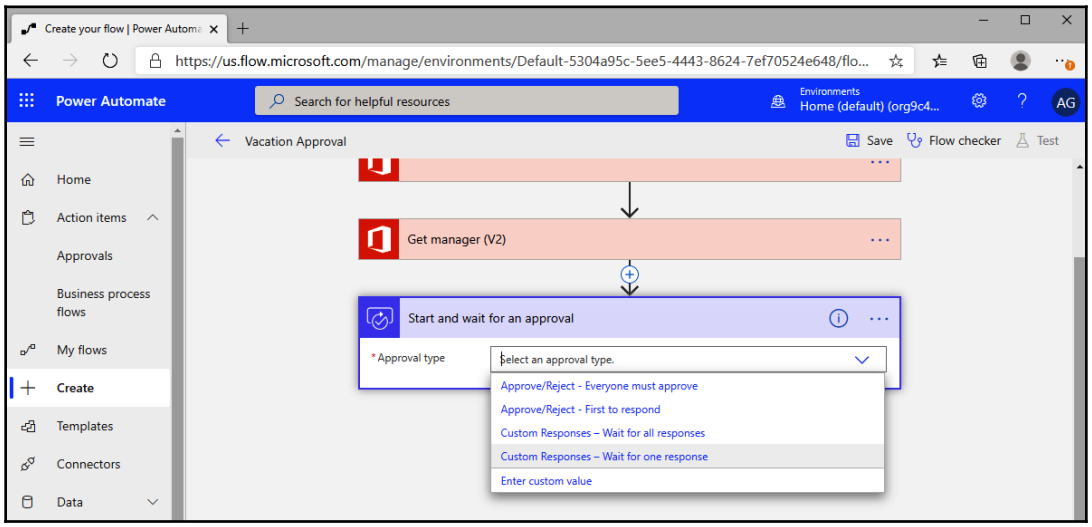
4. Click **+New step**. Add the **Get my profile (V2)** action.
5. Click **+New step**. Add the **Get manager (V2)** action.
6. In the **User (UPN)** value box of the **Get manager (V2)** action, add the dynamic content token for **Get my profile (V2).User Principal Name**:



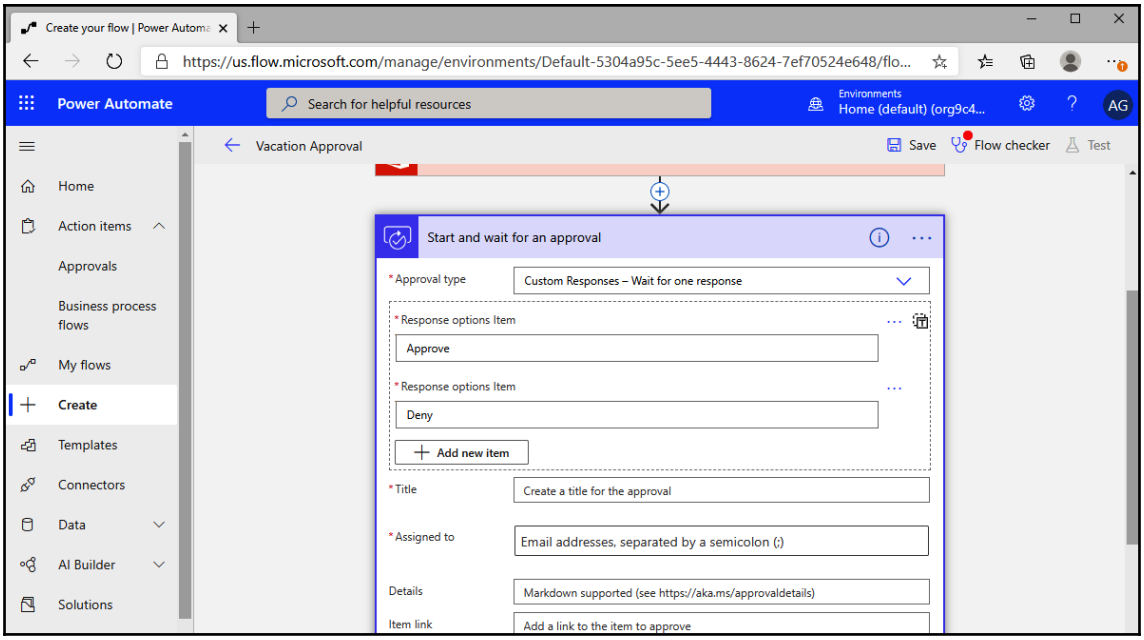
7. Click **+New step** and add the **Start and wait for an approval** action:



8. In the **Approval type** box, select the **Custom Responses - Wait for one response** option:

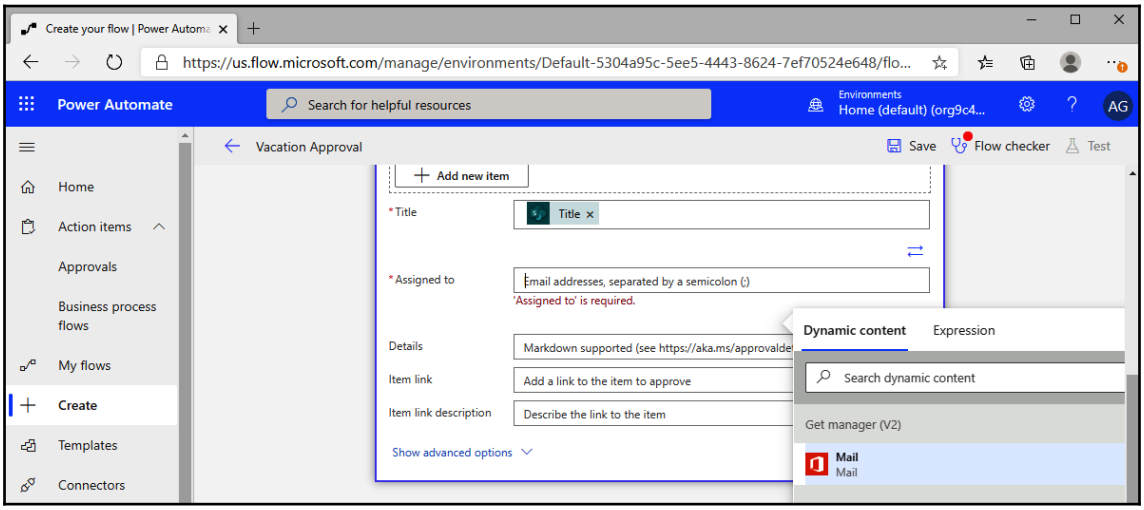


9. Fill out the form as appropriate. For **Response options Item**, enter `Approve`. Then, select **+Add new item** and add another option for **Deny**. These are the options that will be displayed when the approval is sent to the approver:

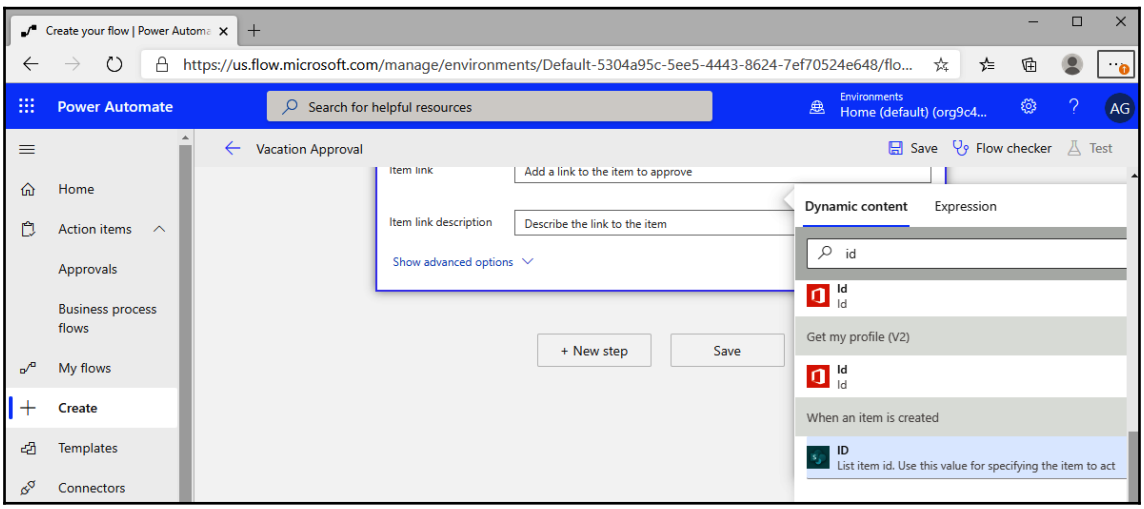


10. In the **Title** box, you can create a title that will be displayed for the approval. You can also use dynamic content tokens to populate it.

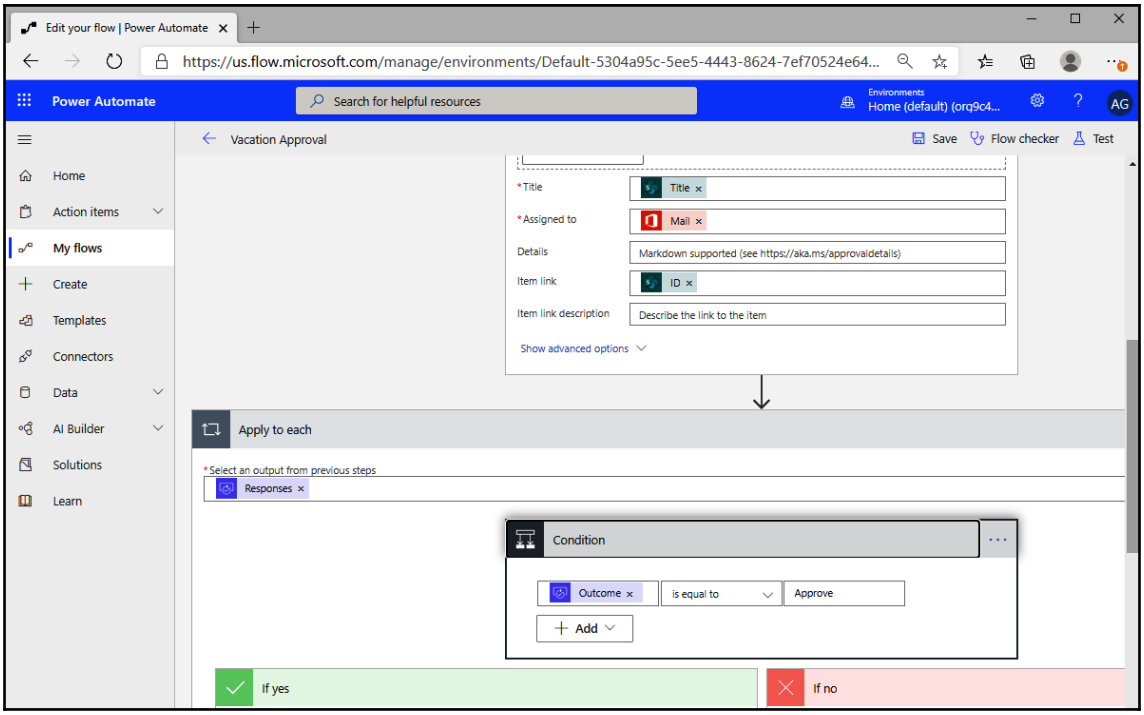
11. In the **Assigned to** box, select the **Mail** dynamic token under the **Get manager (V2)** section:



12. In the **Details** box, enter any additional details or text you wish to include in the approval notification.
13. In the **Item link** box, add the dynamic content token **ID** under the **When an item is created** section:



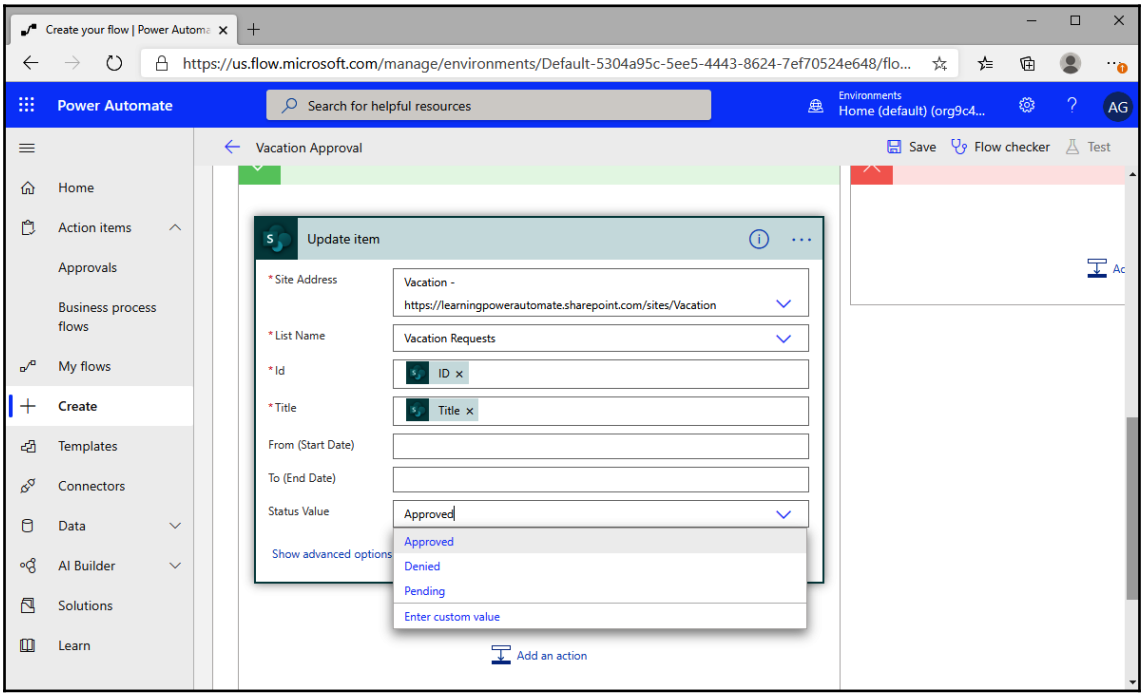
14. Click **+New step** and add the **Condition** control.
15. In the **Choose a value** box, add the **Responses** dynamic content token. Power Automate should create an **Apply to each** container.
16. Select the title bar of the condition card inside the **Apply to each** container. On the left-hand side of the condition control, in the **Choose a value** box, add the **Outcome** dynamic content token. On the right-hand side, enter the `Approve` value. This text should match the **Response** options text for the **Approve** option we entered in *step 9*:



17. Under the **If yes** branch of the condition, click **Add an action**.
18. Select the SharePoint **Update item** action.
19. On the card for the **Update item** action, in the **Site Address** box, add where the vacation request was submitted.
20. On the card for the **Update item** action, in the **List Name** box, add the list containing the vacation request item.
21. On the card for the **Update item** action, in the **ID** box, select the dynamic content token for **ID** under the SharePoint **When an item is created** section.

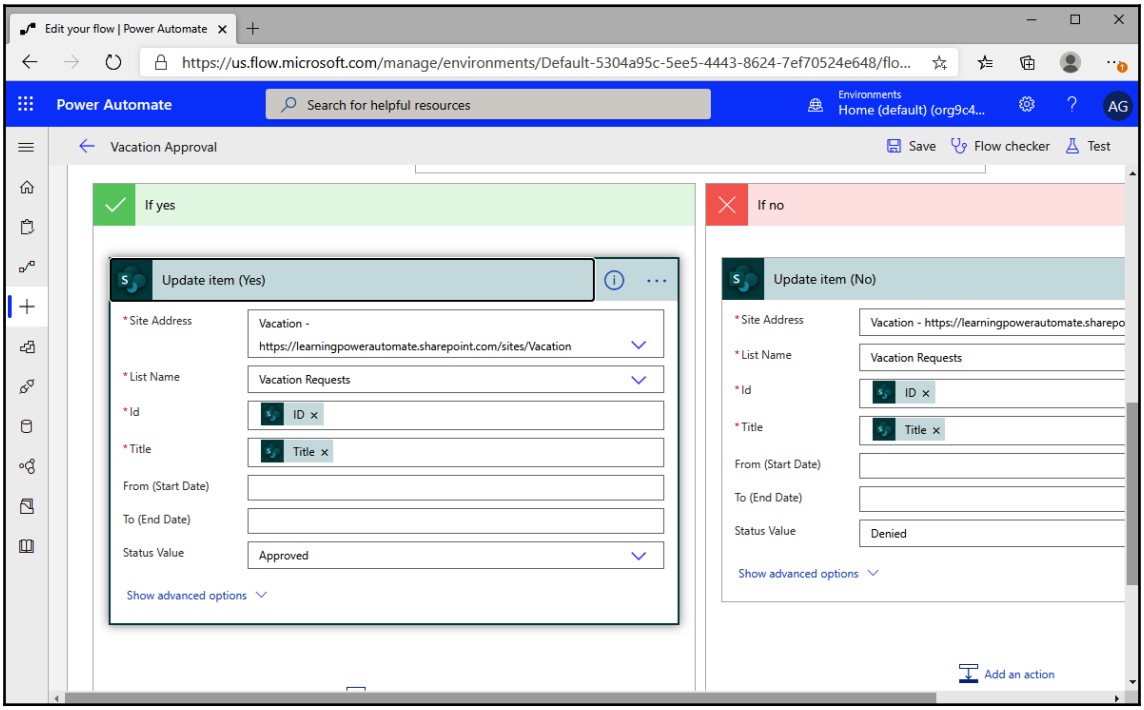
22. On the card for the **Update item** action, in the **Title** box, select the dynamic content token for **Title** under the **SharePoint When an item is created** section.

23. On the card for the **Update item** action, in the **Status Value** box, select the `Approved` value:



24. Under the **If no** branch, repeat *steps 17* through *22*.

25. On the card for the **Update item** action, in the **Status Value** box, select the `Denied` value:



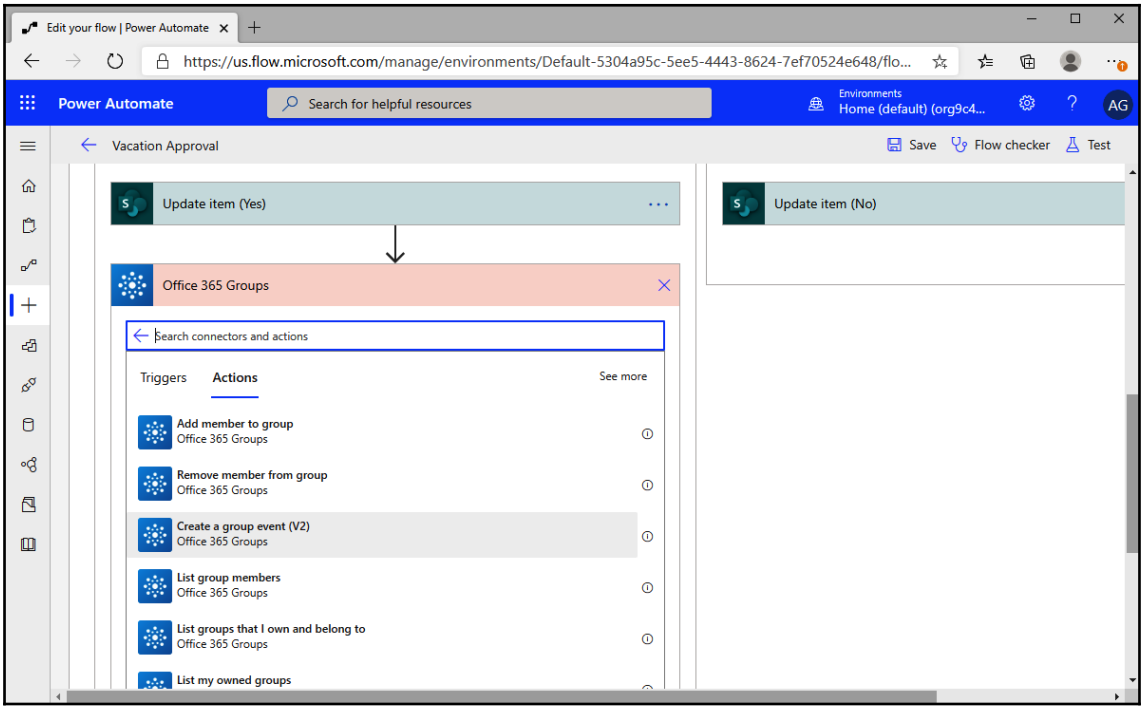
26. Click **Save** to save the flow.

For an advanced modification to this, you can also add individual or group calendar events, as you'll see in the next section.

Adding calendar events and notifications

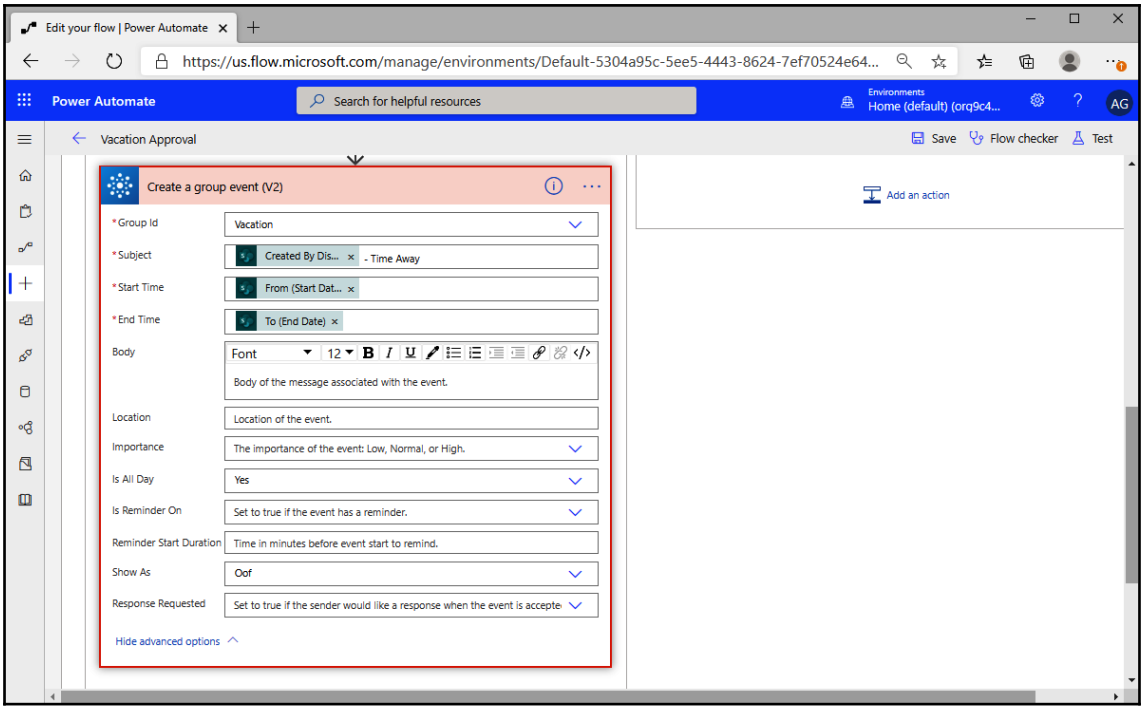
You can increase this approval's usefulness by adding automatic calendar appointments to the **If yes** branch. You can also include additional information in emails to the request originator if desired. Use the following optional steps to modify and extend this approval with a shared calendar appointment:

1. With **Vacation Approval** open, navigate to the end of the **If yes** branch.
2. Click **Add an action** and select the **Create a group event (V2)** action under **Office 365 Groups**:



3. On the **Create a group event (V2)** card, in the **Group Id** box, select the Office 365 group corresponding to the SharePoint site for the vacation's approval. You can also select another group as long as the connected account has access.

4. On the **Create a group event (V2)** card, in the **Subject** box, enter any value consisting of dynamic content tokens or static content.
5. On the **Create a group event (V2)** card, in the **Start Time** box, select the dynamic content token representing the start time from the SharePoint **When an item is created** section.
6. On the **Create a group event (V2)** card, in the **End Time** box, select the dynamic content token representing the end time from the SharePoint **When an item is created** section.
7. On the **Create a group event (V2)** card, expand the advanced options. Set the **Is All Day** to **Yes** and the **Show As** option to **Oof**. Edit any additional options to your preference:



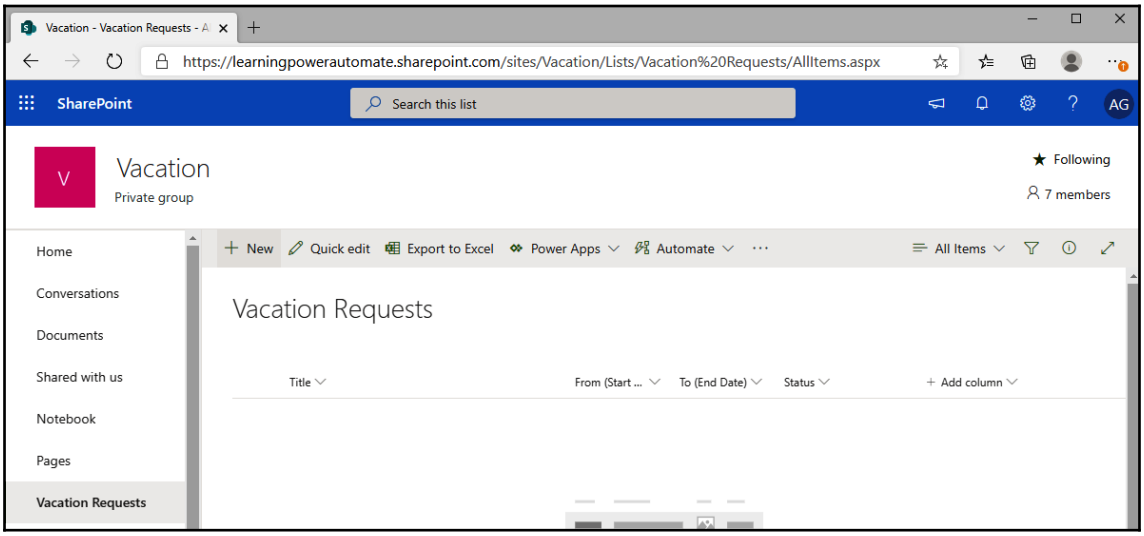
8. Click **Save** to save the flow.

The Office 365 group will now be updated with a calendar showing the requester's approved time off. Next, we'll learn how the flow works as a requester.

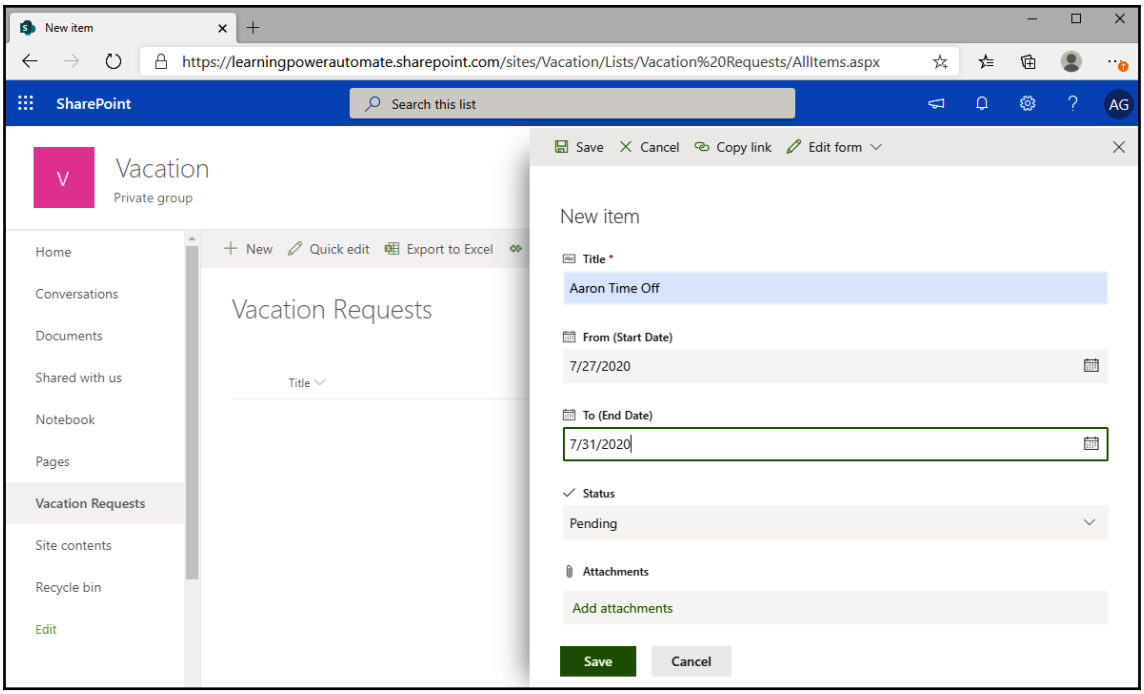
Starting the flow

The easiest way to test this flow is to start an approval process. To do so, follow these steps:

1. Navigate to the SharePoint list that was created as part of this exercise.
2. On the list, click **+New** to create a new request item:



3. Fill out the form. Add a title and select the start and end dates, and then click **Save**:



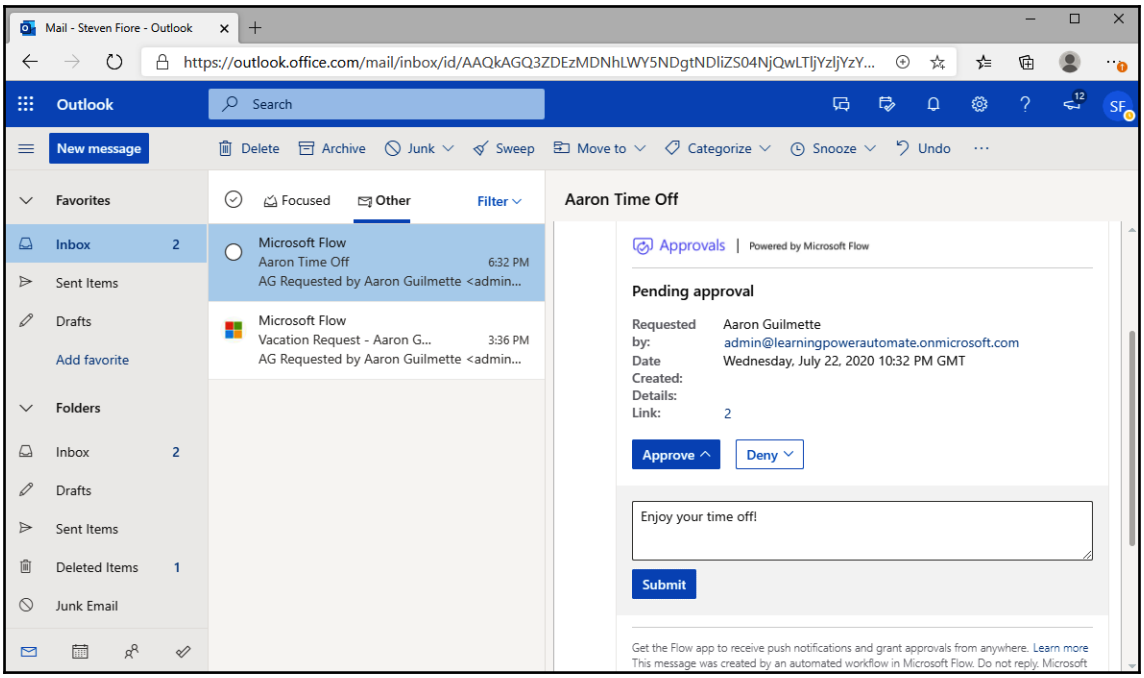
Clicking **Save** will trigger the flow to begin.

At this point, you have created a SharePoint list that can be used as the trigger for the approval flow, as well as the approval flow itself. Finally, you triggered the flow. Next, we'll see how it looks from the approver's side.

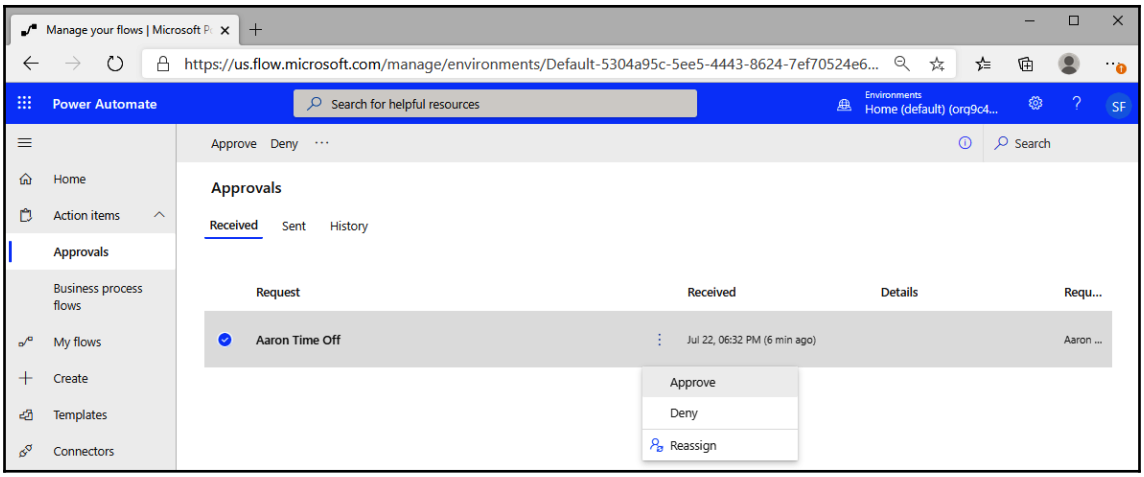
Responding to approvals

As an approver, you'll receive an email notification stating that action is required. Approvals will also show up in the **Approvals** section of both the Power Automate web portal and the mobile app. An approval can be processed from any of those locations.

The approval has buttons based on the various **Response option items** selected in the approval flow. In the following screenshot, you can see that the options we selected when creating the approval flow are displayed in the body of the email. The approver can select **Approve** or **Deny**, enter any optional information, and then click **Submit**:

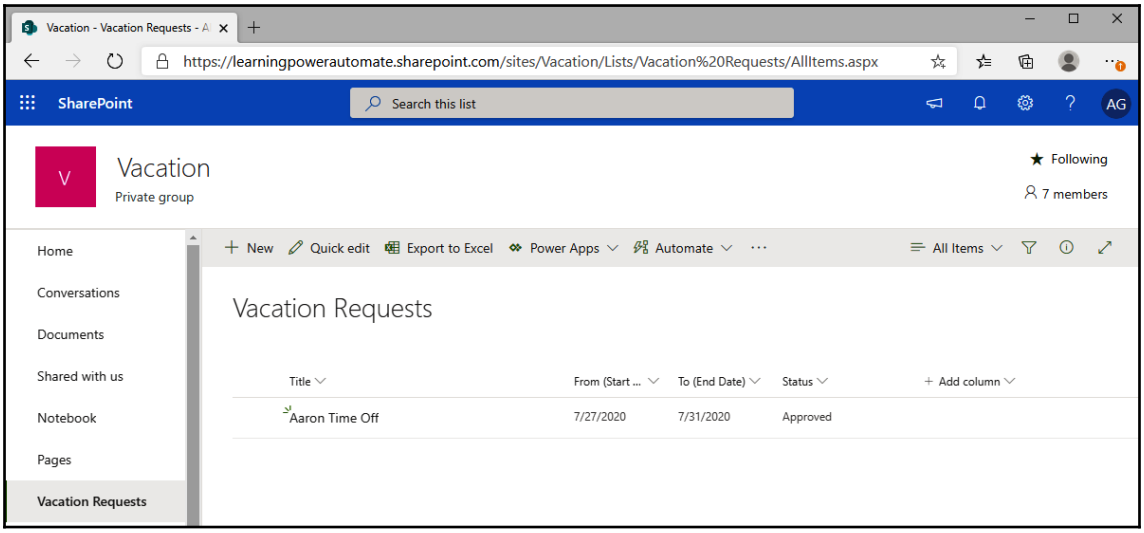


The approver can also log into the Power Automate web portal and navigate to **Action items** | **Approvals** to see and act on any pending approval requests:

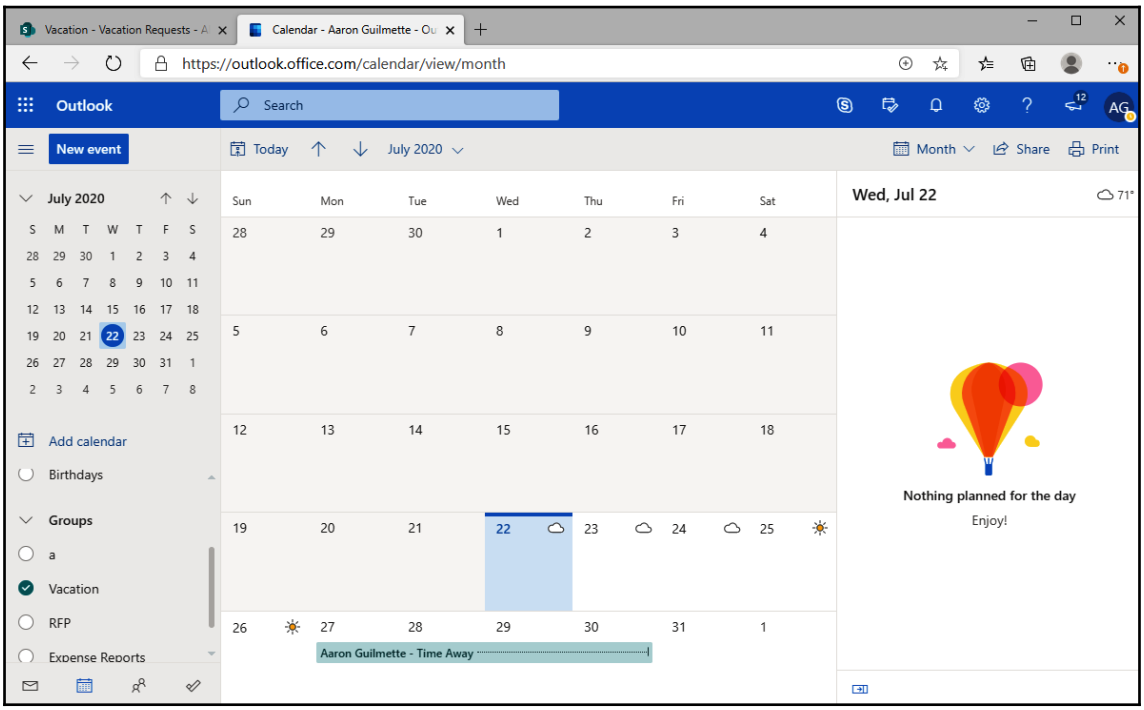


As an approver in the web portal, you can choose to **Approve**, **Deny**, or **Reassign** the approval to another user.

After choosing an option, the item is updated in the SharePoint Online list, as shown in the following screenshot:



If you updated the flow with the group calendar event action, you will also see the calendar item through the Outlook web app:



You can continue editing and refining the approval request flow by adding email notifications, attachments, or even integrating it into other flows.

Summary

In this chapter, you learned about one of the most popular and useful flow capabilities available in Power Automate – the approval flow. Approval flows frequently utilize the CDS to store state information. Approval flows can be used for simple tasks such as vacation requests or expense forms, and can also be integrated into more complex tasks such as document publishing and content automation. Finally, you learned how to create an approval flow, trigger the approval, and respond to it as an approver.

In the next chapter, we'll begin using Power Automate to interact with Microsoft Forms.

10
Working with Multiple Approvals

In `Chapter 9`, *Getting Started with Approvals*, you learned the basic mechanics of an approval flow with a single approver. While that scenario is very common, it doesn't address all of the likely requirements that you'll meet. In many instances, you may need to have multiple individuals approve items, either in sequence or in parallel. This is where multiple approvals fit.

In this chapter, we're going to look at addressing two common multiple approval scenarios:

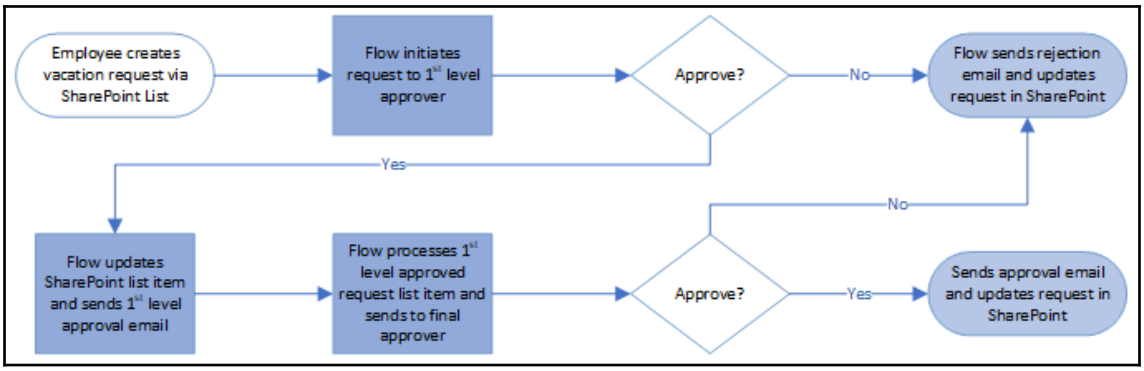
- Working with sequential approvals
- Working with parallel approvals
- Working with advanced scenarios
- Creating a basic sequential approval

By the end of this chapter, you'll understand how you can use these techniques independently or together to make more complex approval workflows.

Working with sequential approvals

A sequential approval is a multiple approval scenario requiring various stages of approval before a final approver can accept or reject it. Common scenarios include expense reports or vacation requests where an immediate manager approves an item, which is then routed to a second-level manager.

An example sequential approval sequence is shown here:



The previous diagram depicts this order of events:

1. The user adds an item to a SharePoint list.
2. Power Automate, which was configured to use a new item appearing on the list as a trigger, initiates the approval flow and sends the approval request to the first approver.
3. The first approver responds to the request (either *Approves* or *Rejects*).
4. The Power Automate flow processes the approval answer. However, unlike a single approval ending the flow, Power Automate instead starts the second stage (or final stage) of the approval.
5. The second approver responds to the request (either *Approves* or *Rejects*).

An approval process can have any number of approval actions and is not just limited to first and second stages.

Next, we'll look at an overview of parallel approvals.

Working with parallel approvals

Whereas sequential approvals process as a single thread in a row or stages, parallel approvals happen independently. A vacation request may also take the form of a parallel approval: you may need two individuals to approve time off (such as your immediate manager as well as an HR representative), but those individuals might be in different management or reporting structures and their approval processes are independent of each other.

Parallel approval processes allow different branches of approvals to happen independently. Logic at the end is used to evaluate whether the overall approval is successful or not.

Working with advanced scenarios

There are some additional configuration options that are available for creating even more complex approval workflows. These options are sometimes used in large organizations where one or more individuals or groups can interact with the approval process. Two such options we'll briefly look at include **mixed approval types** and **everyone must approve** options.

Mixed approval types

For very complex scenarios, an approval may contain multiple levels of sequential approvals, a mix of parallel and sequential approvals, or even a sequential approval inside a parallel approval branch. These types of approvals are known as **mixed approval**. When evaluating outcomes and responses, you'll need to make sure you're selecting the right dynamic content tokens.

Everyone must approve

When creating an approval workflow, it may be desirable to only send a final response if one of two conditions is met:

- *Everyone* included in the approval chain approves
- *Anyone approver* rejects the approval request

These types of approvals are known as **everyone must approve**.

To demonstrate some of these techniques, we'll create an approval that uses a series (or sequence) of approvals.

Creating a basic sequential approval

In this section, we're going to create a basic sequential approval that involves first- and second-stage approval. These types of approvals are common in scenarios where an individual requests more than a certain number of vacation days or expenditure that reaches a certain threshold.

Like other workflows, you can use a SharePoint list as the starting point. You can also take input from other methods, such as reading content from a file or even requesting user input (which you'll learn about in `Chapter 14`, *Accepting User Input).*

For this example, you'll be creating a list with a number of different columns and data types. The columns will be used to store information as it moves through the process. The overall process will look like this:

- Configuring the prerequisites
- Creating the flow
- Testing the flow

The design of the flow will stipulate that the first-level approver is the requester's manager, while the second-level approver is a user called Finance Manager. The request will need to be approved by both users in order for it to be marked as **Approved**.

We'll configure the prerequisites next.

Configuring prerequisites

Follow these steps to require a second purchase approval if the final dollar amount is over $1,000:

1. Navigate to a SharePoint site that will contain the list used to start this workflow. If you don't have a suitable site, you can create one.
2. Select **Site contents** from the navigation menu, and then click **+ New** and select **List**.
3. Create a SharePoint list with the following columns and types:

Column Name	**Column Type**
Title	Single line of text
Modified	Date and time
Created	Date and time
Purchase Amount	Currency

Comments	Single line of text
Pre-approval required	Yes/No (set **No** as default)
Pre-approved	Yes/No (set **No** as default)
Approved	Yes/No (set **No** as default)
Created by	Person or group
Modified by	Person or group

Once the columns have been created, you can move on to creating the approval flow. Be sure to note the URL of the SharePoint site and list that you're using, as you'll need them in the next section.

Creating the flow

Once your SharePoint list has been created, it's time to start working with the flow itself. We'll create this flow in several sections:

- Creating the trigger
- Creating the first-stage approval
- Creating the second-stage approval
- Completing the flow

Let's go!

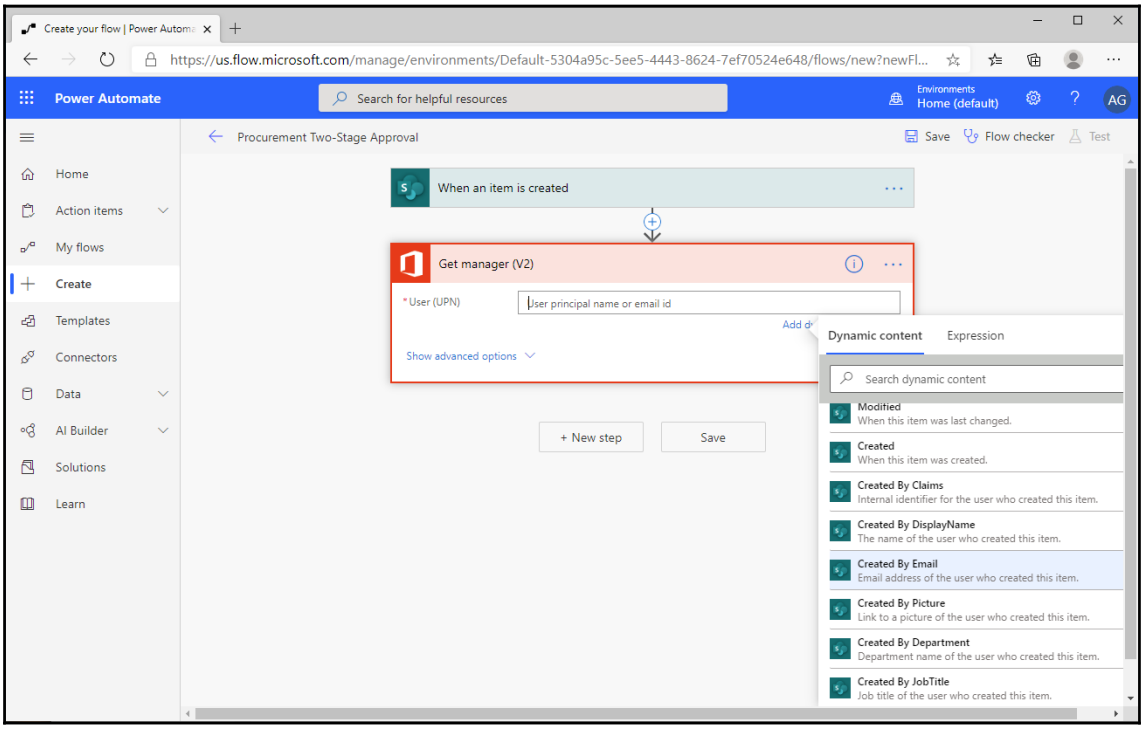
Creating the trigger

The trigger will be used to initiate the workflow. This flow's trigger action is a user placing a purchase request in the SharePoint list.

To create the flow, follow these steps:

1. Navigate to the Power Automate web portal (`https://flow.microsoft.com`), log in, and click **+ Create**.
2. Select **Automated**.
3. Enter a name for the flow (such as `Procurement Two-Stage Approval`).
4. From the list of triggers, select the **When an item is created** SharePoint trigger and then click **Create**.
5. Expand the trigger by clicking on the title bar. In the **Site Address** field, select the URL for the SharePoint site where the list is stored. If the site does not appear in the drop-down, select **Custom value,** and enter the site address.

6. In the **List Name** field, select the name of the list you created in the prerequisites section.
7. Click **+ New step**.
8. Select the **Get manager (V2)** Office 365 Users action.
9. In the **User (UPN)** field, add the dynamic content token for **Created by Email** in the **When an item is created** section:



10. Click **+ New step**.
11. Select **Condition** control.
12. On the left side of the condition, select the **Purchase Amount** dynamic content token from the **When an item is created** SharePoint section. Choose **is greater than** for the evaluation type. Enter `1000` on the right side of the condition.
13. Click **Save** to save your progress.

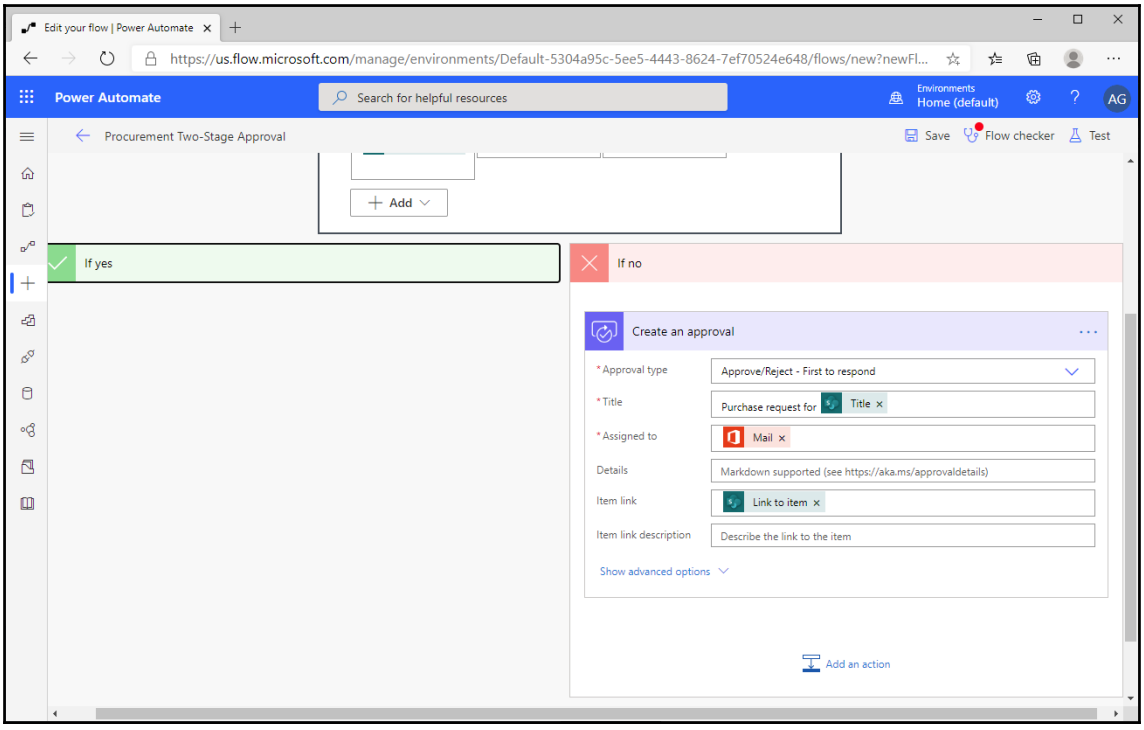
Now that the trigger and condition have been created, it's time to proceed to create the first stage of the approval process.

Creating the first-stage approval

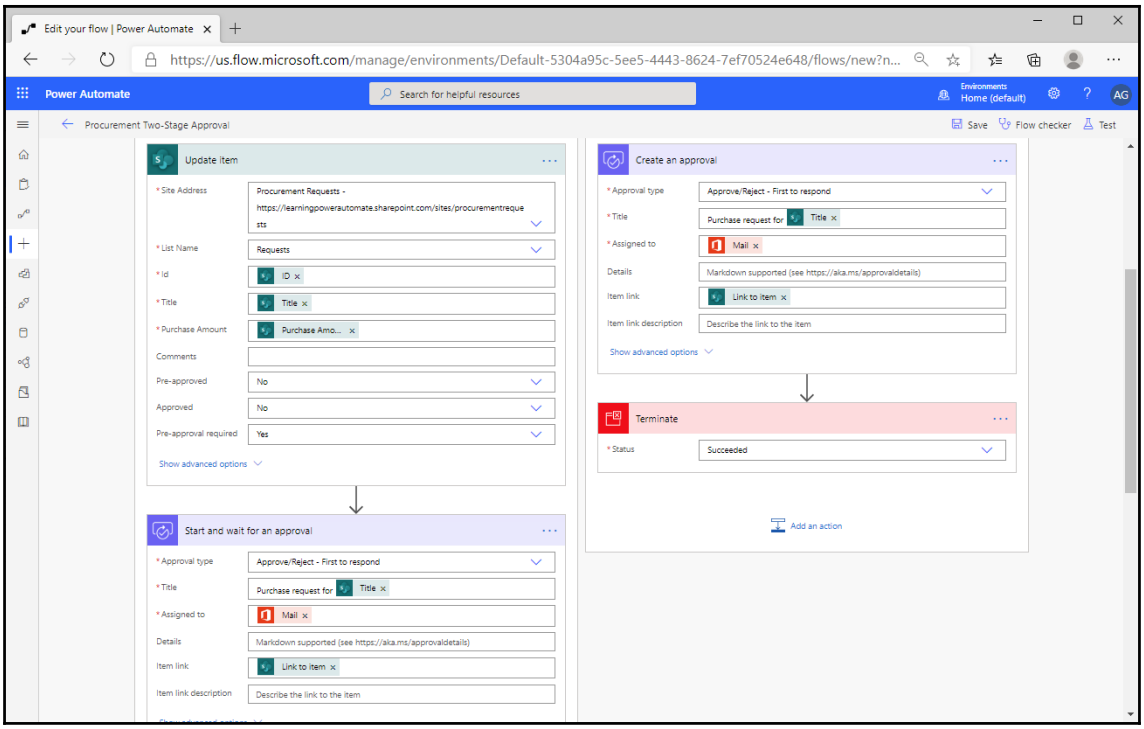
This stage of the approval is routed to the user's immediate manager and can take two paths: if the dollar value is over **$1,000**, it will proceed down a branch that will require a second stage approval. If it is less than $1,000, it can be approved by the user's immediate manager.

To continue working on the approval, follow these steps:

1. With the flow open, expand the **If no** condition branch. This is the condition branch that will execute if the purchase price is $1,000 or less. Click **Add an action**, and then select the **Create an approval** Approvals action.
2. For the **Approval type**, select **Approve/Reject - First to respond**.
3. For the **Title**, include the dynamic content token for **Title** under the **When an item is created** section, along with any other descriptive text.
4. For **Assigned to**, select the **Mail** dynamic content token under the **Get manager** section.
5. For **Item link**, select the **Link to item** dynamic content token under the **When an item is created** section:

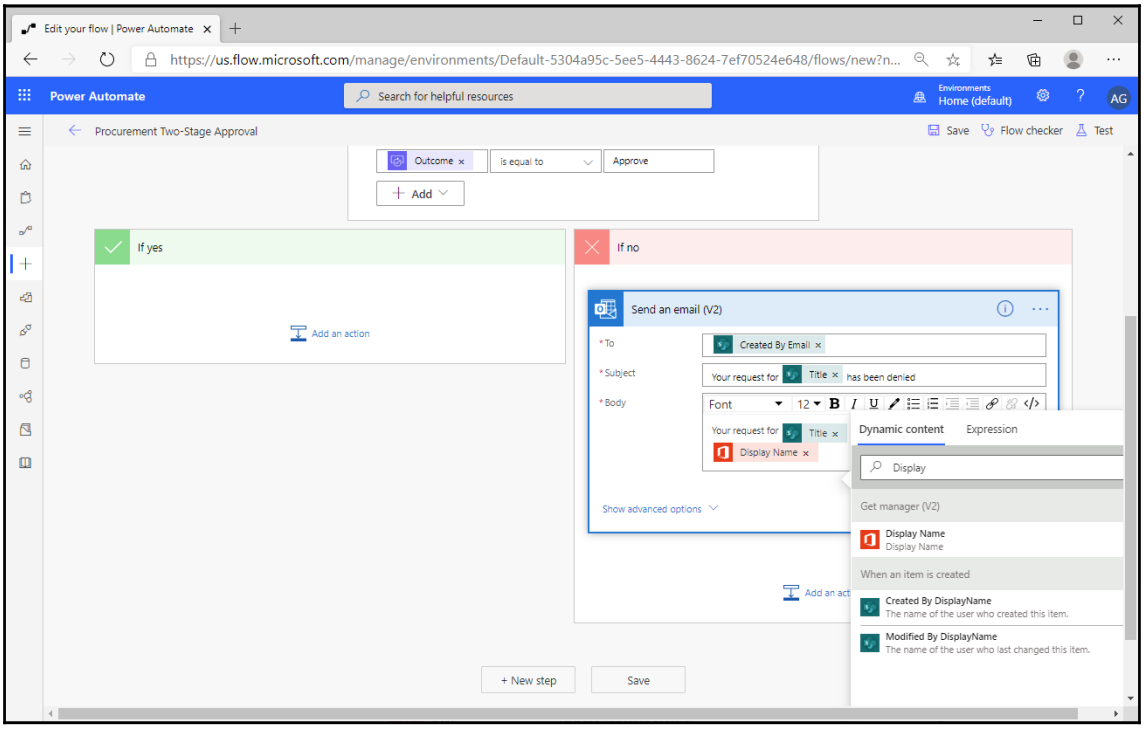


6. Click **Add an action**.
7. Select the **Terminate** control.
8. In the **Terminate** control **Status** box, select **Succeeded**. This will stop further steps from executing.
9. Expand the **If yes** condition branch.
10. Click **Add an action**, and then select the **Update item** SharePoint action.
11. In the **Site Address** field, select the site containing the SharePoint list used for this workflow.
12. In the **List Name** field, select the list used for this workflow.
13. In the **Id** field, select the **ID** dynamic content token in the **When an item is created** section.
14. Set the **Title** field to the **Title** dynamic content token in the **When an item is created** section.
15. Set the **Purchase Amount** field to the **Purchase Amount** dynamic content token in the **When an item is created** section.
16. Ensure the **Pre-approved** dropdown is set to **No**.
17. Ensure the **Approved** dropdown is set to **No**.
18. Set the **Pre-approval required** dropdown to **Yes**.
19. Click **Add an action**, and then select the **Start and wait for an approval** Approvals action.
20. For **Approval type**, select **Approve/Reject - First to respond**.
21. For **Title**, include the dynamic content token for **Title** under the **When an item is created** section, along with any other descriptive text.
22. For **Assigned to**, select the **Mail** dynamic content token in the **Get manager** section.
23. For **Item link**, select the **Link to item** dynamic content token in the **When an item is created** section. The completed condition steps should look similar to the following screenshot:



24. Click **Save** and **+ New step**.
25. Select **Condition** control. In the **Condition** control, select the **Choose a value** box on the left, and then select the **Outcome** dynamic content token in the **Start and wait for an approval** section.
26. In the **Choose a value** box on the right, enter the text `Approve`.
27. In the **If no** branch, click **Add an action** and select the **Send an email (V2)** Office 365 Outlook action.
28. Add the **Created By Email** SharePoint dynamic content token to the **To** field.
29. Add a deny message to the **Subject** field. You may want to include the **Title** SharePoint dynamic content token.

30. Add a deny message to the **Body** field. You may want to use the **Title** SharePoint dynamic content token as well as the **Display Name** dynamic content token from the **Get manager** action:



31. Click **Add an action**, and then select the **Terminate** control.
32. Select **Succeeded** as the **Status**.
33. Click **Save**.

At this point, the first stage of the approval process is complete. For purchases under $1,000, the immediate manager can approve or reject the request. If it is approved, it can move on to the second stage of the approval process.

By default, a step is labeled with the name of the action. However, in long or complex flows with several conditions or branches, it may be helpful to rename particular steps to help document where you are in the process. To rename a step, click on the ellipsis (...) in the title bar of the step and select **Rename**. Each step in a flow must have a unique name value.

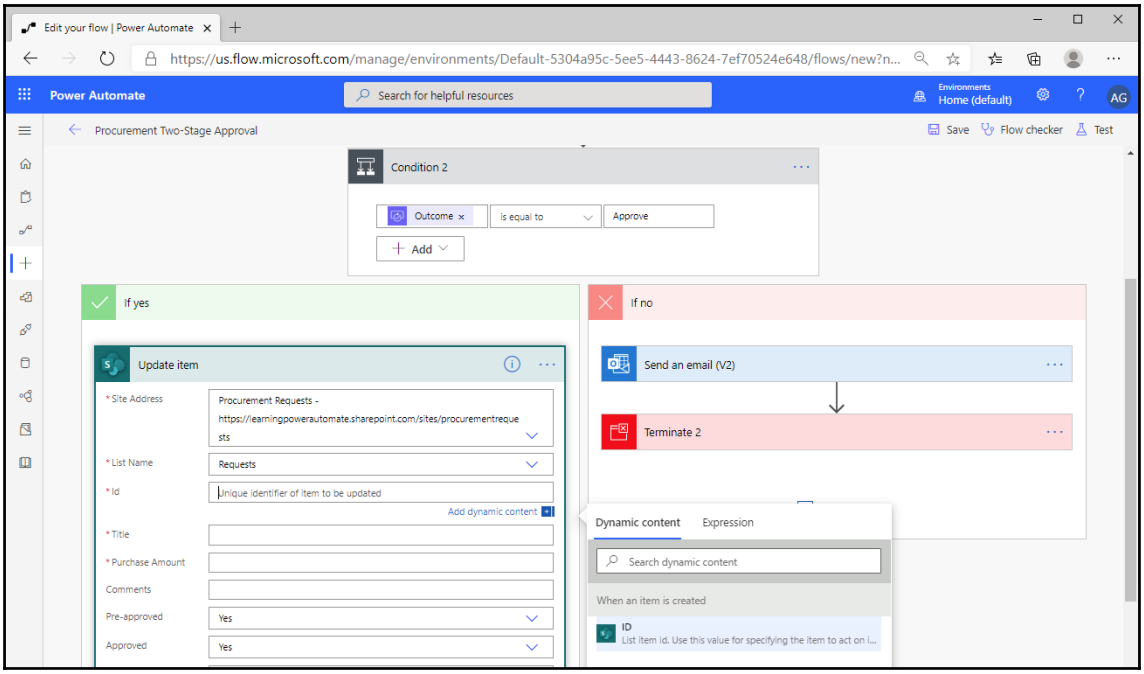
Now, let's move on to creating the next stage of approval.

Creating the second-stage approval

In this section of the approval process, the request is sent to the second-level or second-stage approver. The second-stage approver will also have the ability to approve or deny the request.

To complete the flow, follow these steps:

1. With the flow open, select the **If yes** branch of the condition and then click **Add an action**.
2. Select the **Update item** SharePoint action.
3. In the **Site Address** field, select the site containing the SharePoint list used for this workflow.
4. In the **List Name** field, select the list used for this workflow.
5. In the **Id** field, select the **ID** dynamic content token under the **When an item is created** section:



6. Set the **Title** field to the **Title** dynamic content token under the **When an item is created** section.

7. Set the **Purchase Amount** field to the **Purchase Amount** dynamic content token in the **When an item is created** section.
8. Set the **Pre-approved** dropdown to **Yes**.
9. Ensure the **Approved** dropdown is set to **No**.
10. Ensure the **Pre-approval required** dropdown is set to **Yes**.
11. Click **Add an action**.
12. Select the **Start and wait for an approval** Approvals action.
13. For the **Approval type** field, select **Approve/Reject - First to respond**.
14. For the **Title** field, select the **Title** dynamic content token from the **When an item is created** section.
15. For the **Assigned to** field, enter the **Finance Manager** email address. Alternately, you can use the output of another **Get manager** action to determine the pre-approver's manager.
16. Click **Save**.

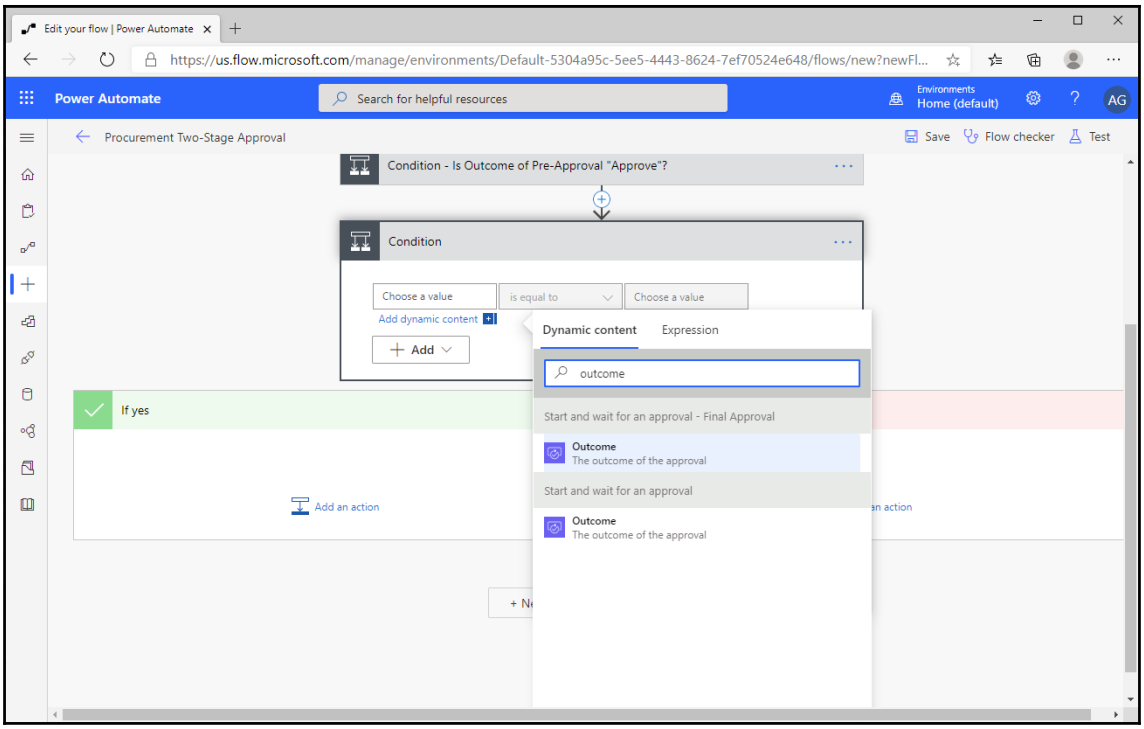
You've now completed the second-stage approval section. Next, we'll move on to completing the flow.

Completing the flow

In this final section of the flow creation, we'll follow a process similar to the pre-approval: using a condition to check the value of **Outcome** and then updating the SharePoint item accordingly.

To complete this flow, follow these steps:

1. With the flow open, click **+ New step**.
2. Add a **Condition** control.
3. For the **Choose a value** box on the left side of the condition, select the **Outcome** value in the **Start and wait for an approval** section for the final approval:



4. Select **is equal to** as the evaluation method for the condition.
5. Set the **Choose a value** box on the right side of the condition control to `Approve`.
6. In the **If no** condition branch, click **Add an action**.
7. Select the **Send an Email (V2)** Office 365 Outlook action.
8. In the **To** field, add the **Created By Email** dynamic content value under the **When an item is created** section.
9. In the **Subject** field, add a message. Optionally, add the **Title** dynamic content value.
10. In the **Body** field, add a message body. Optionally, add the **Title** dynamic content value and any other values (such as the **Approver** name).
11. Click **Add an action** and select the **Update item** dynamic content value in the **When an item is created** section.
12. Configure the **Site Address**, **List Name**, **Id**, **Title**, and **Purchase Amount** values using the dynamic content value tokens from the **When an item is created** section.
13. Set the **Pre-approved** field to **No**.
14. Set the **Approved** field to **No**.

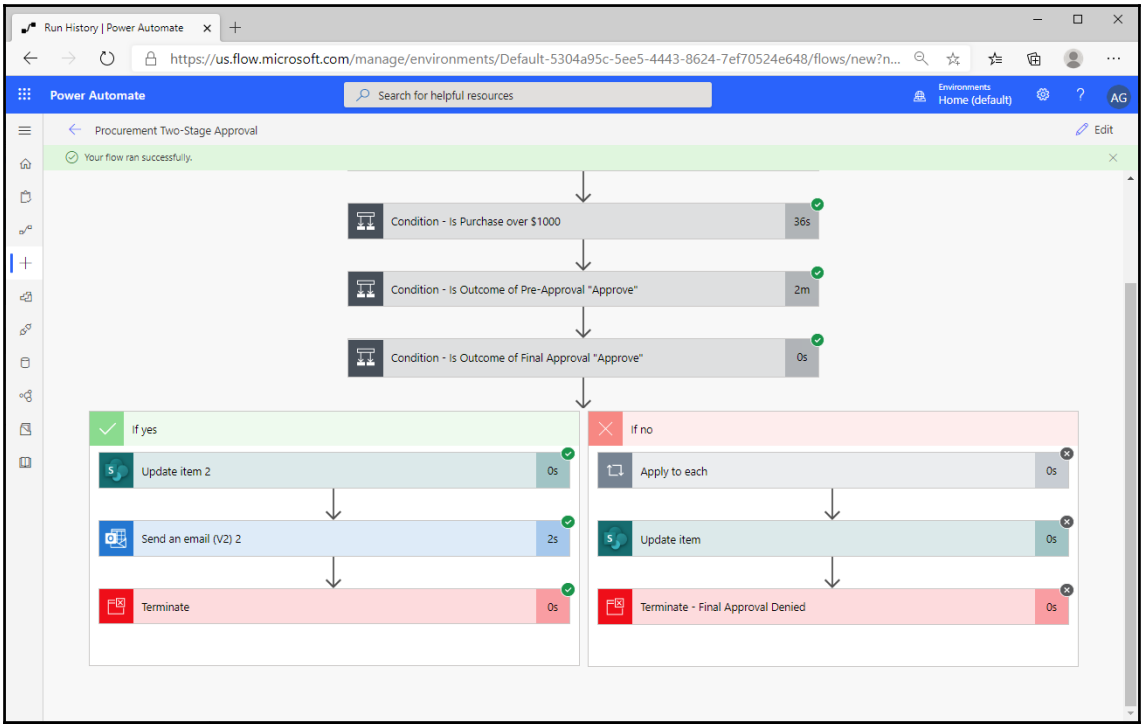
15. Set the **Pre-approval required** field to **Yes**.
16. Click **Add an action** and then select the **Terminate** control.
17. Set the **Status** value to **Succeeded**.
18. Select the **If yes** condition branch, and then select **Add an action**.
19. Select the **Update item** SharePoint action.
20. Configure the **Site Address**, **List Name**, **Id**, **Title**, and **Purchase Amount** values using the dynamic content value tokens from the **When an item is created** section.
21. Set the **Pre-approved** field to **Yes**.
22. Set the **Approved** field to **Yes**.
23. Set the **Pre-approval required** field to **Yes**.
24. Select the **Send an Email (V2)** Office 365 Outlook action.
25. In the **To** field, add the **Created By Email** dynamic content value under the **When an item is created** section.
26. In the **Subject** field, add a message. Optionally, add the **Title** dynamic content value.
27. In the **Body** field, add a message body. Optionally, add the **Title** dynamic content value and any other values (such as the **Approver** name).
28. Click **Add an action** and select the **Terminate** control.
29. Set the **Status** field to **Succeeded**.
30. Click **Save**.

The approval has been successfully created! Since this is an especially long and complex flow, you may want to review the items and update any descriptions or names of steps to clarify them for others who may look at the flow (or for yourself if you need to perform troubleshooting).

When finished, it's time to test.

Testing the flow

To test this flow, navigate to the SharePoint site containing the list being monitored and add values. In order to verify that it works as designed, you may need to run several tests, including requests over and under $1,000, and then approve and deny them at either the first or second stage:



If any of the steps error, review the flow steps to ensure you're using the appropriate dynamic content tokens and that condition evaluation statements are correct.

As you become more familiar and confident with Power Automate concepts, you'll discover new ways to customize the outputs or steps, including using complex expressions, additional formatting, and combining dynamic content values to produce rich output.

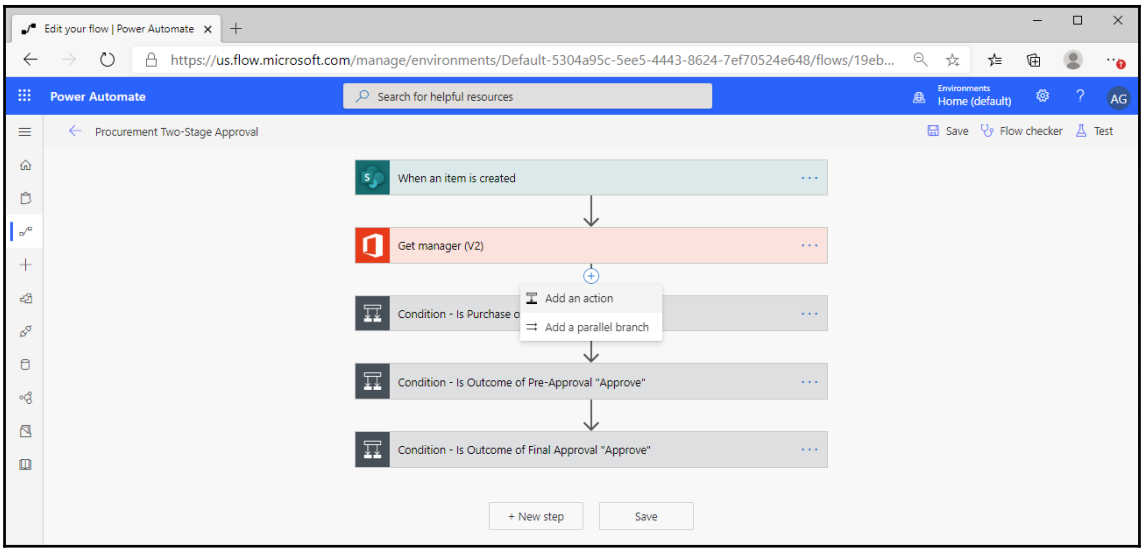
Exploring further options

For long-running approval processes or approval processes involving multiple users and approval steps, you may want to send periodic email reminders to users. One way to do this is by populating variables at each approval stage and checking whether they're complete using a parallel approval branch. You'll do this with five components:

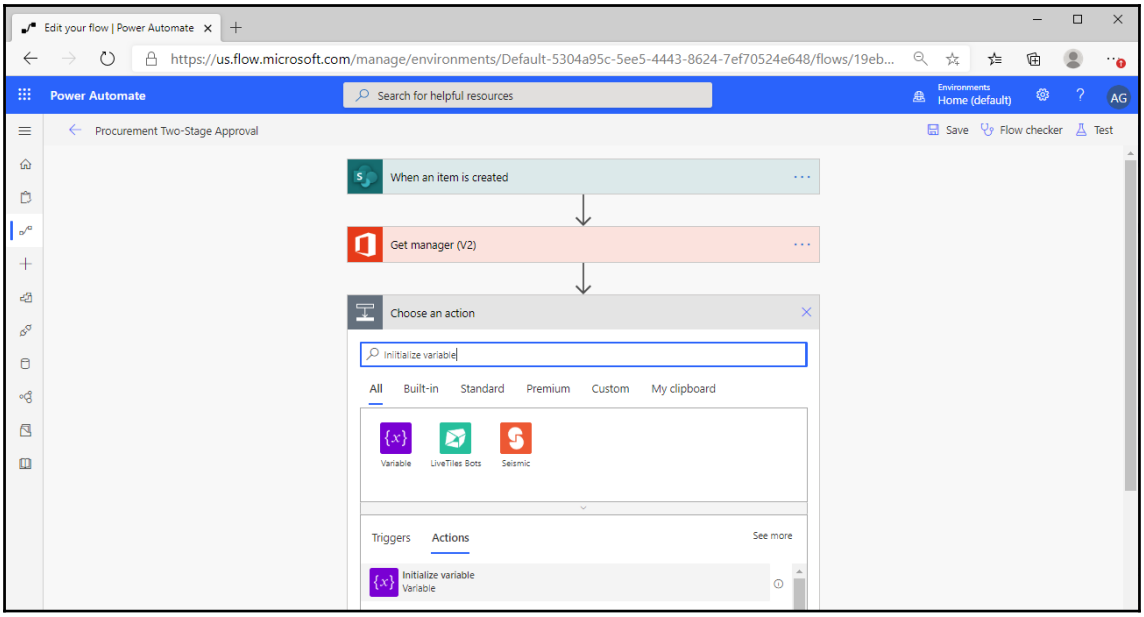
- The **Initialize variable** action will be used to configure a variable (such as **FirstApprovalDone**) that you'll check throughout the approval. The variable will be initialized to **false**, and then only set to **true** once that approver approves the request.
- The **Do until** control will be used to create a loop that periodically checks to see whether the variable (**FirstApprovalDone**, in the example) has changed from **false** to **true**.
- The **Delay** action will allow you to delay the sending of the email by some period of time (to give the approver a chance to respond before getting a notification).
- The **Send an email** action that will send the reminder email to the approver.
- The **Set variable** action will be used to update the variable (**FirstApprovalDone**) to signify that the approval has been completed.

You can use the following steps to add reminder emails:

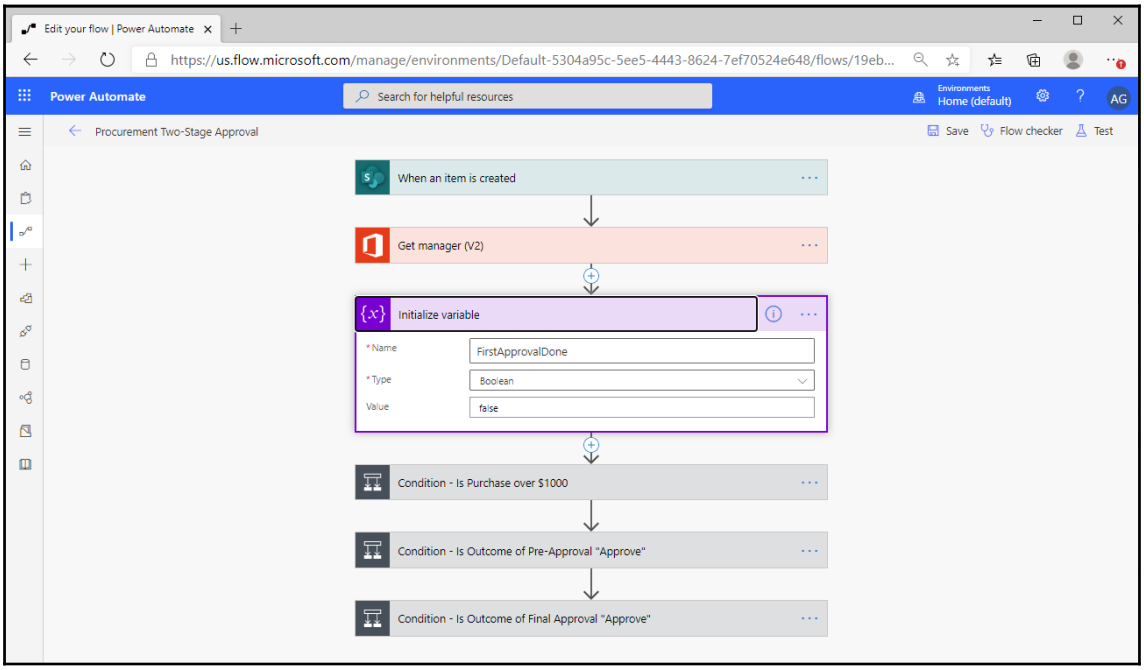
1. Using the Power Automate web portal interface (`https://flow.microsoft.com`), edit the approval.
2. Navigate to the **Get manager (V2)** step, hover over the arrow below it, select the **+** icon, and select **Add an action**:



3. Select the **Initialize variable** action:

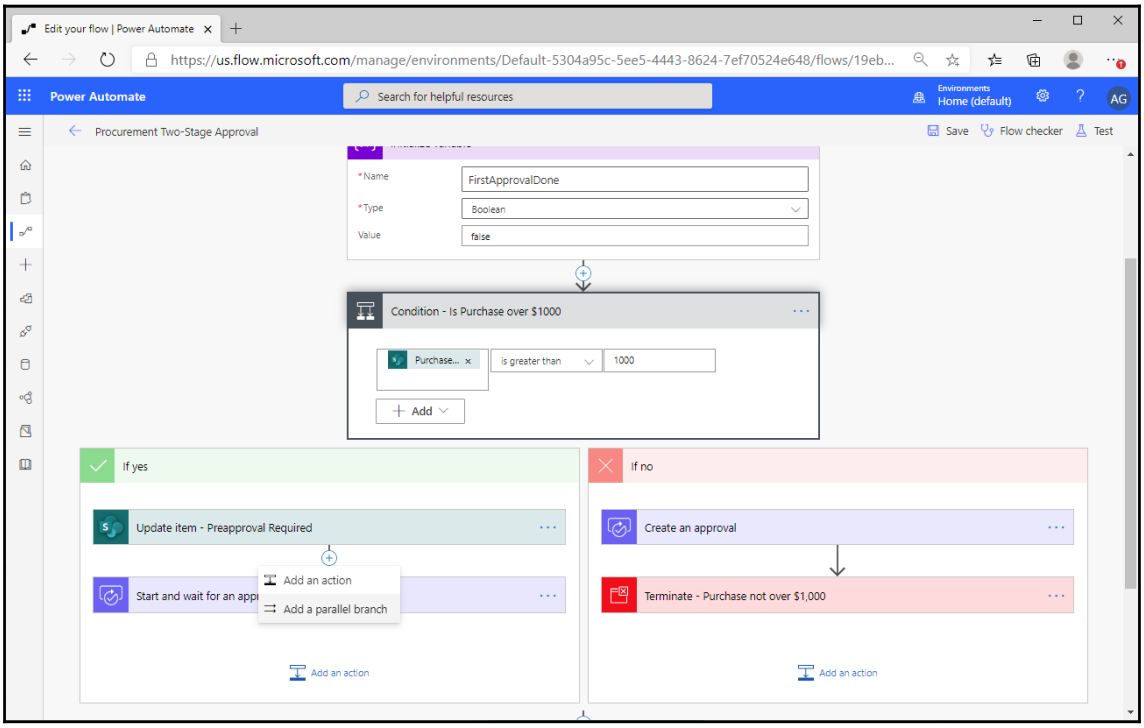


4. Name the variable something that will help you keep track of which approval you're working with. Set the type to **Boolean** and enter a starting value of `false`. In this case, we're going to start sending notification emails for the first approval, so we'll name it **FirstApprovalDone**:

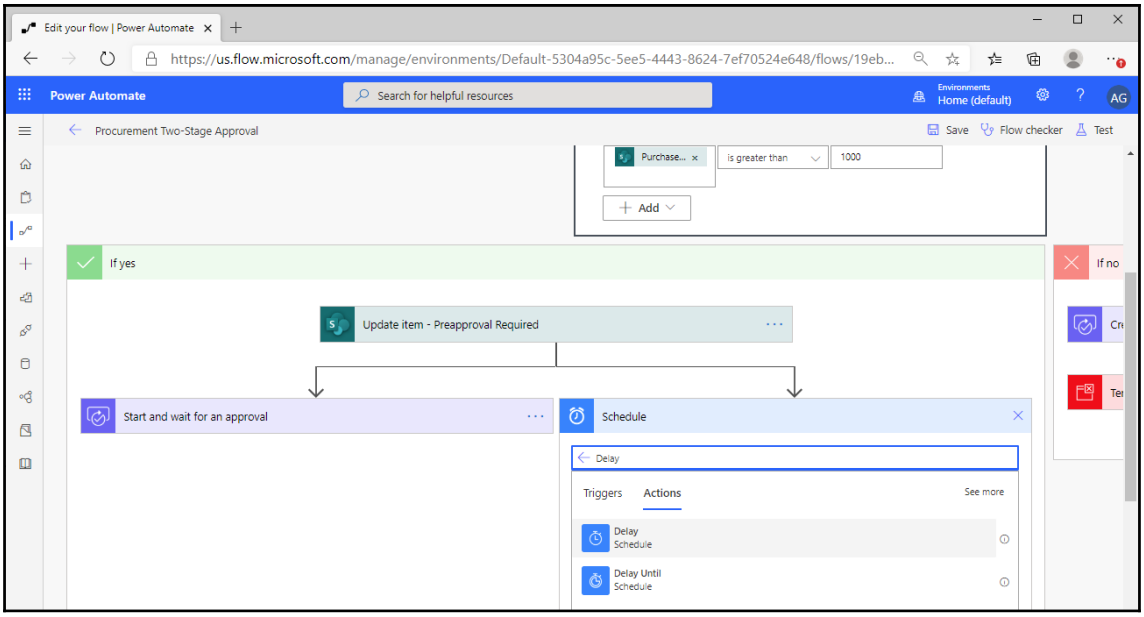


5. Next, expand the condition where the first approval is created.

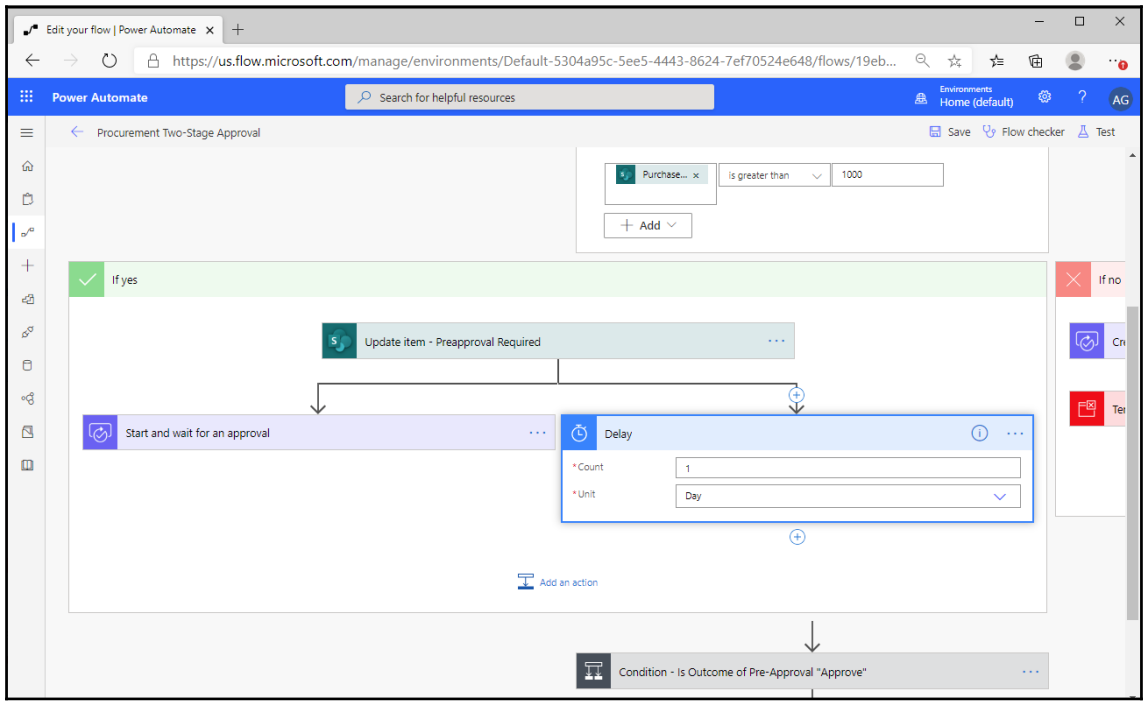
6. Select the **+** icon between the **Update item** and **Start and wait for an approval** actions, and then select **Add a parallel branch**:



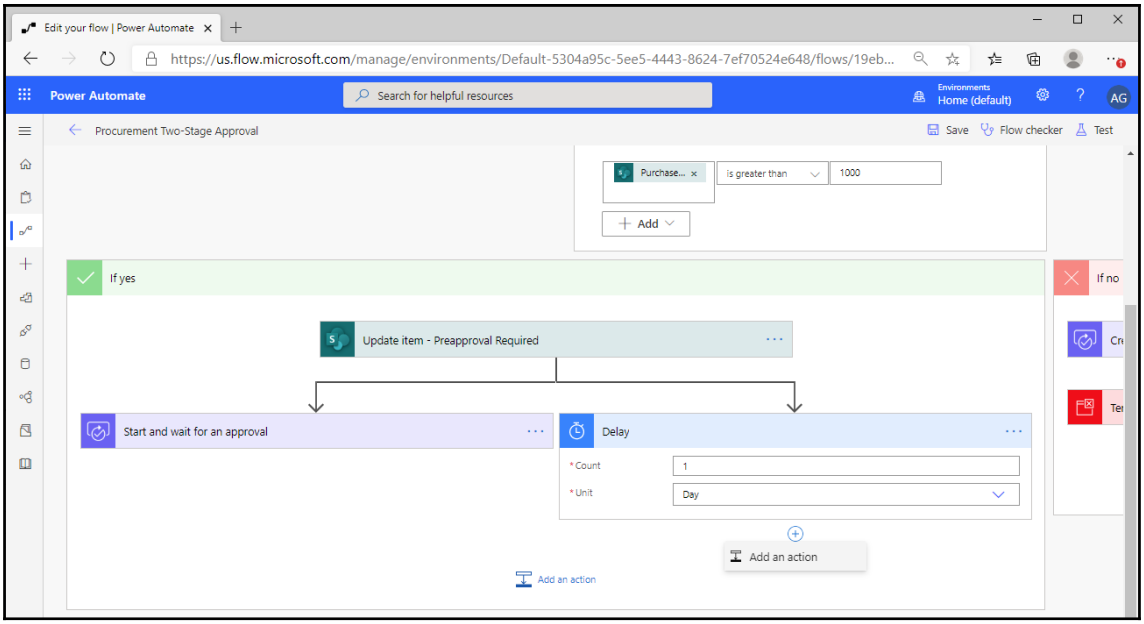
7. In the new branch, select the **Delay Schedule** action:



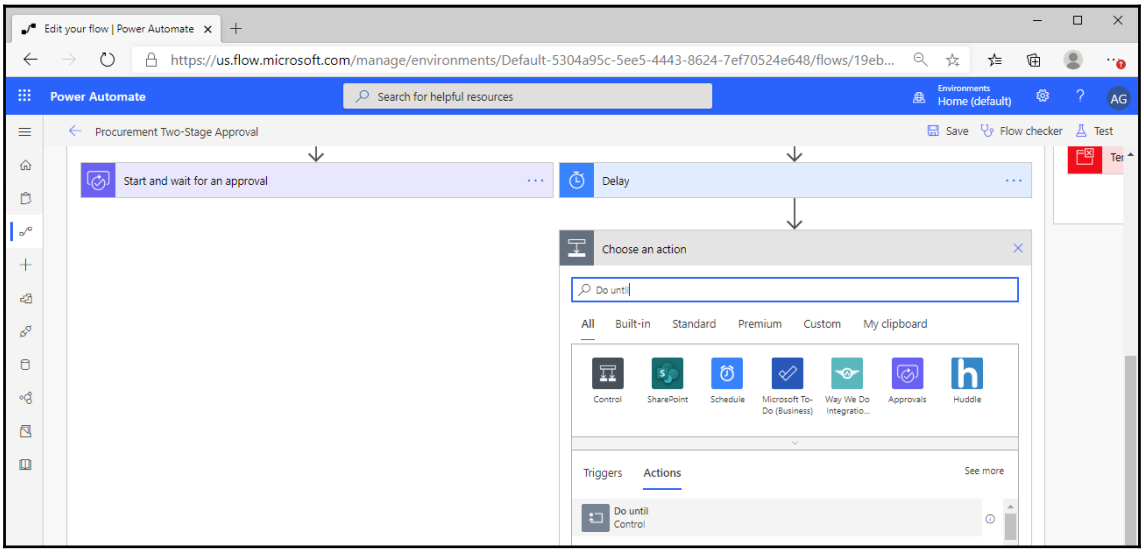
8. Enter a number for **Count** and a time unit, which you will use as a delay for sending the first reminder email. In this example, we select **1** as the count and **Day** as the unit:



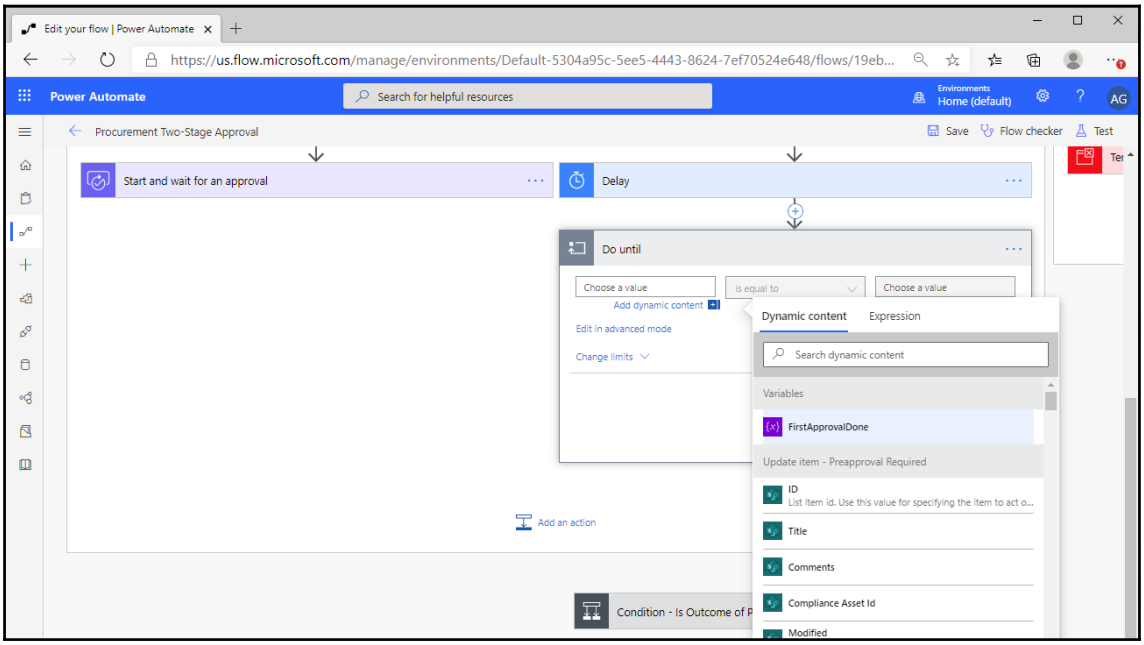
9. Click **+** to insert a step under **Delay** and select **Add an action**:



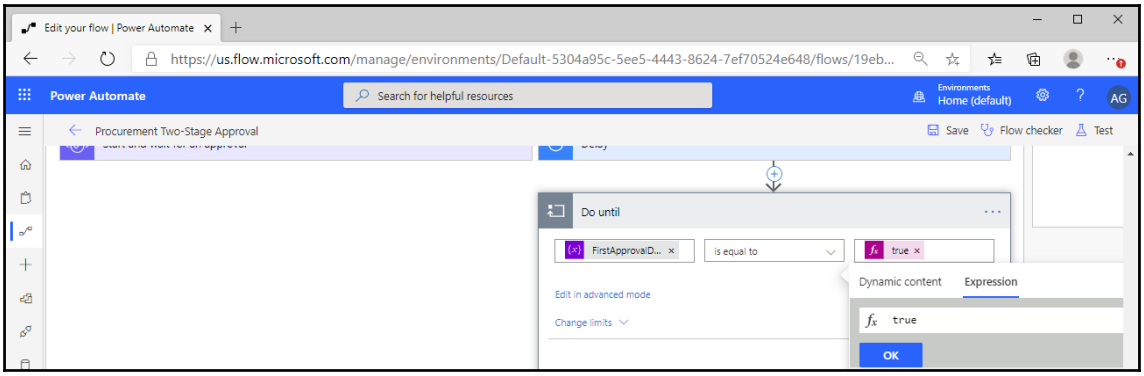
10. Select the **Do until** control:



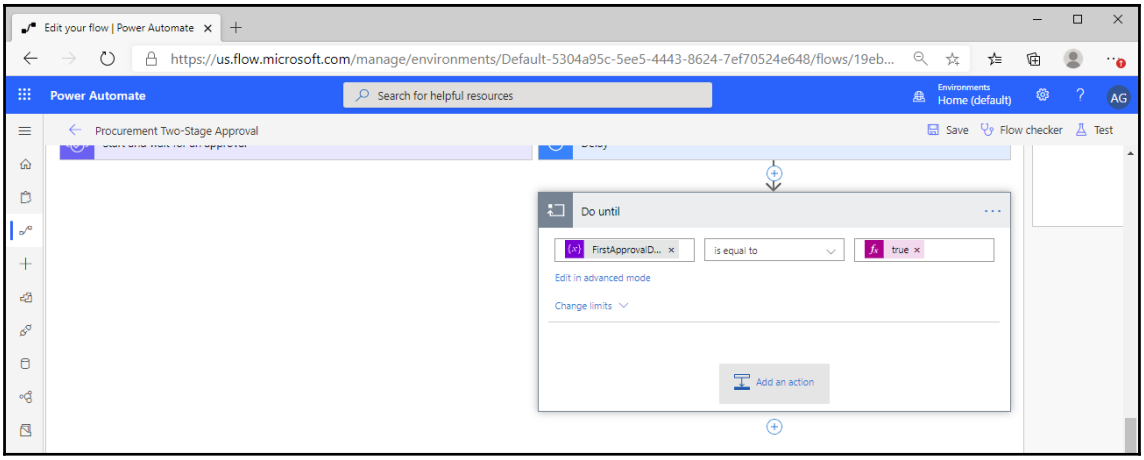
11. In the **Choose a value** box on the left side of the **Do until** action, add the dynamic content token for **FirstApprovalDone**:



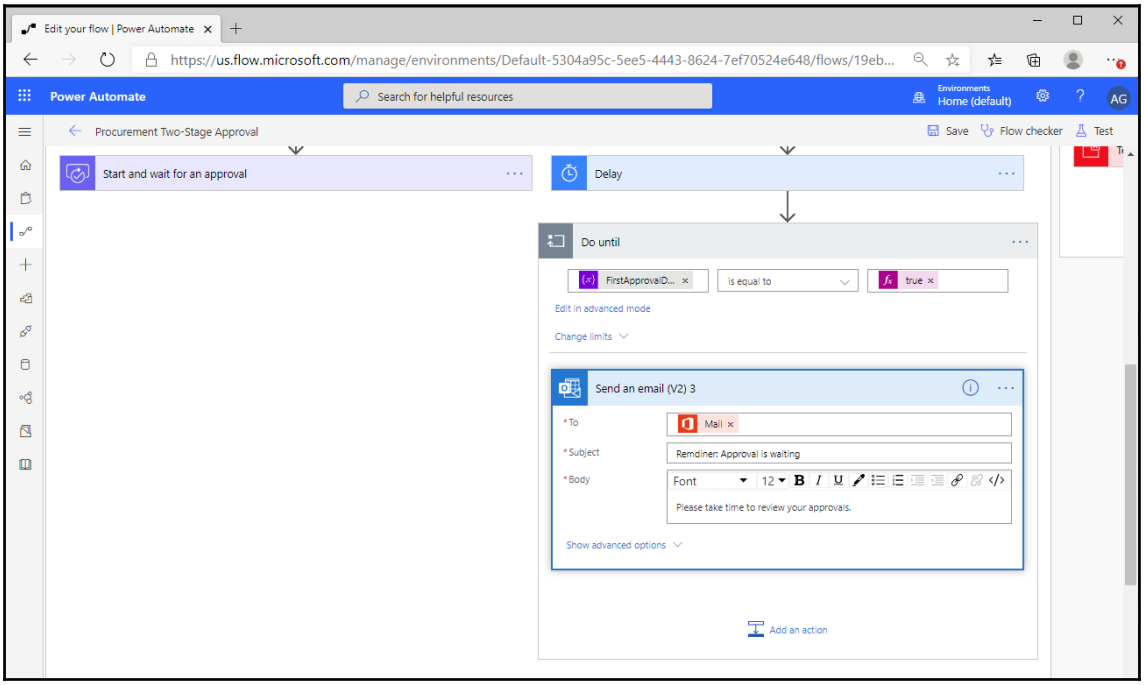
12. In the **Choose a value** box on the right side of the **Do until** action, select the expression for **true**:



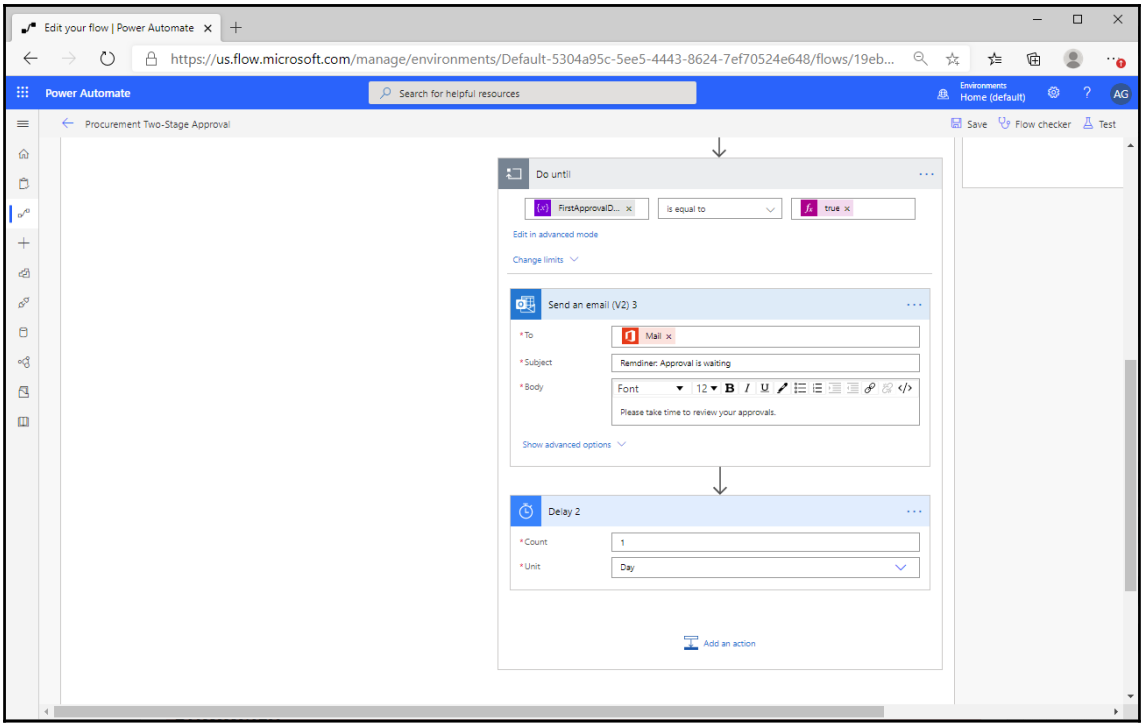
13. Click **Add an action** inside the **Do until** action card:



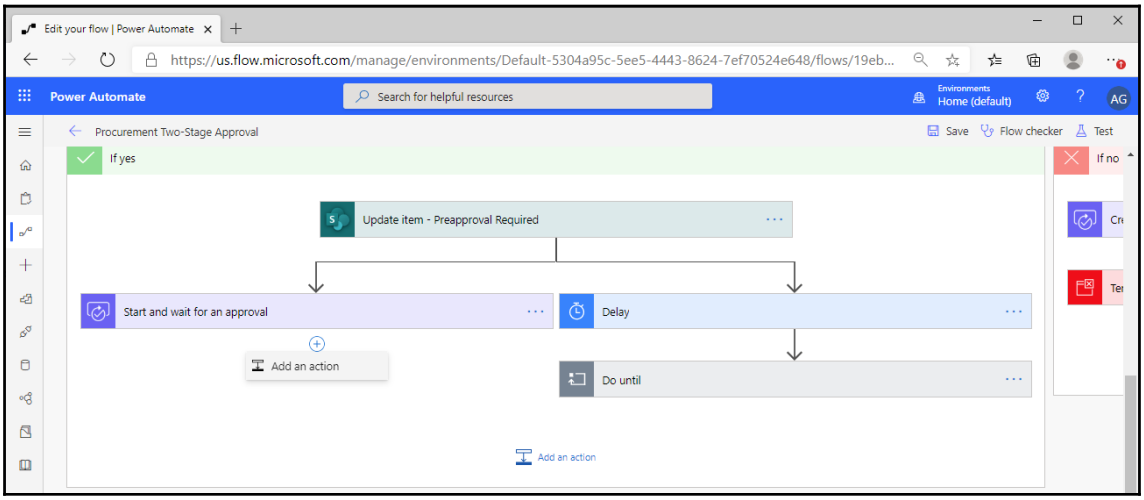
14. Select the **Send an email action**.
15. Fill out the email using the **Mail** dynamic content token value from the **Get Manager (V2)** action. Enter a subject and an email body:



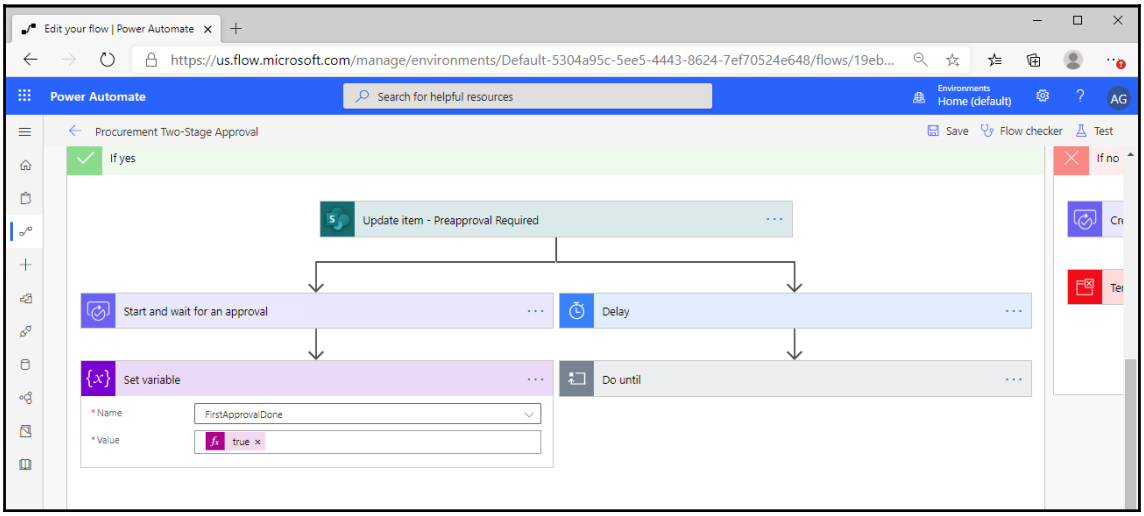
16. Select **Add an action** inside the **Do until** action card.
17. Select the **Delay** schedule action. As before, enter a **Count** value and a unit (interval). This will determine *how often* reminder emails get sent:



18. Minimize the **Do until** card. Select the **+** icon to add a new step under the first branch containing **Start and wait for an approval**:



19. Add the **Set variable** action.
20. Select the **FirstApprovalDone** variable from the variable dropdown, and then add the **true** expression:



21. Click **Save**.

Depending on how many reminder branches you want to instantiate, you can repeat this process for each approval throughout the flow.

Summary

In this chapter, you learned how to start creating approval flows that require multiple stages or approvals to be successful. These types of complex approval request flows are frequently used in medium-and large-sized organizations and reflect real-world scenarios. Becoming familiar with these types of approvals will help you create flows that more accurately meet your organization's needs in the future.

In the next chapter, we'll look at integrating the approvals process with Microsoft Teams.

11
Posting Approvals to Teams

Microsoft has positioned Teams as the "hub for teamwork." With that in mind, we're going to apply the approval flow skills you acquired in `Chapter 9`, *Getting Started with Approvals*, and use them to build an approval experience in Microsoft Teams. You'll be able to use these skills to help streamline the approvals process for members of your organization that are using Teams as a platform.

In this chapter, we're going to cover the following topics:

- Understanding the flow
- Configuring prerequisites
- Configuring an approval flow to use Teams
- Testing the flow

By the end of this chapter, you'll be able to integrate a Power Automate approval flow with Microsoft Teams.

Let's go!

Understanding the flow

Before we configure an approval process, you'll need to make sure that all of the components that support the process exist. In the example flow, we're going to configure a **Purchase Requisition** approval that begins when a document is posted to a SharePoint site. The approval will show up in Teams. If the requisition is approved, the document will be moved to the `Approved` folder. If it's rejected, it will be moved to the `Rejected` folder. The requester should receive an email at the end telling them the status.

This complex flow is going to use several familiar components, including the following:

- A SharePoint **When a file is created in a folder** trigger
- A SharePoint **Get file metadata** action
- A SharePoint **Get file properties** action
- An Office 365 Users **Get user profile (V2)** action
- An Office 365 Users **Get manager (V2)** action
- An Approvals **Create an approval (V2)** action
- Conditions

In this chapter, we're also introducing **the Microsoft Teams connector**. With the Microsoft Teams connector, you'll be able to perform actions such as these:

- **Post your own adaptive card as the Flow bot to a user**: Adaptive cards are a new way to present content directly in a Microsoft Teams conversation or experience. Adaptive cards can contain components such as graphs, images, and formatted text.
- **Post a message as the Flow bot to a user**: This action will allow Power Automate and the Flow bot to communicate with the requester of an approval.

These new actions will allow Power Automate to interact on the Microsoft Teams platform. In order to improve business productivity, the approvers should be able to respond to the approvals inside the Microsoft Teams application.

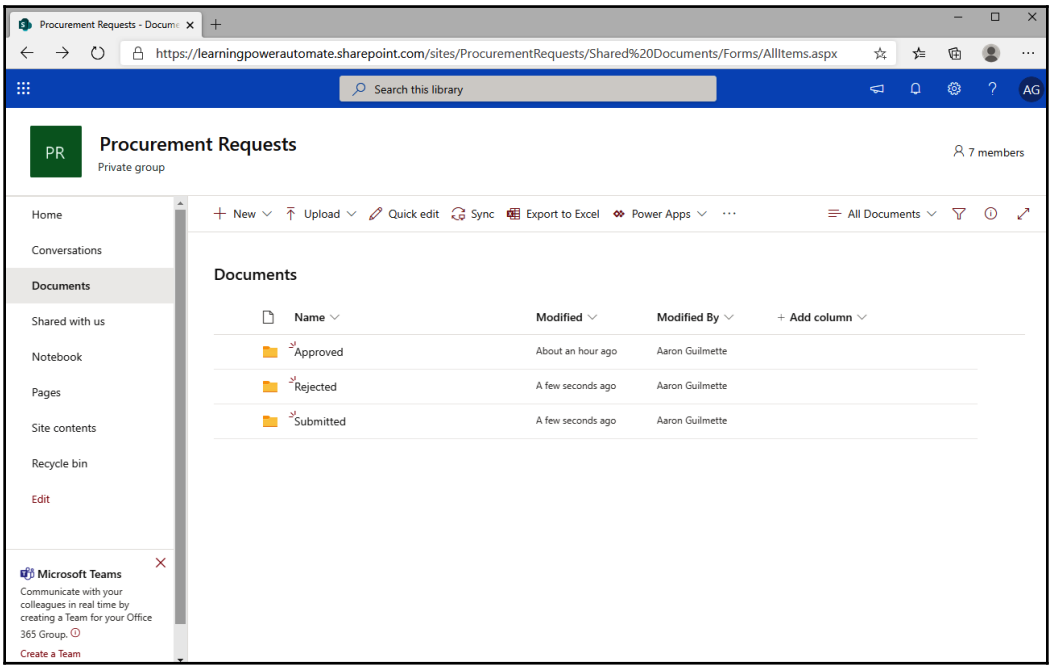
Now that you are familiar with the components that we're going to use, let's move on to configuring the prerequisites.

Configuring prerequisites

This approval flow will require several components and prerequisites. You're already familiar with many of them, but we'll review the exact requirements for this flow. The next two sections will describe the requirements

A SharePoint site

Like many flows, this one uses SharePoint as a document repository. This flow will require a SharePoint site with a document library that contains `Submitted`, `Approved`, and `Rejected` folders, as shown in the following screenshot:

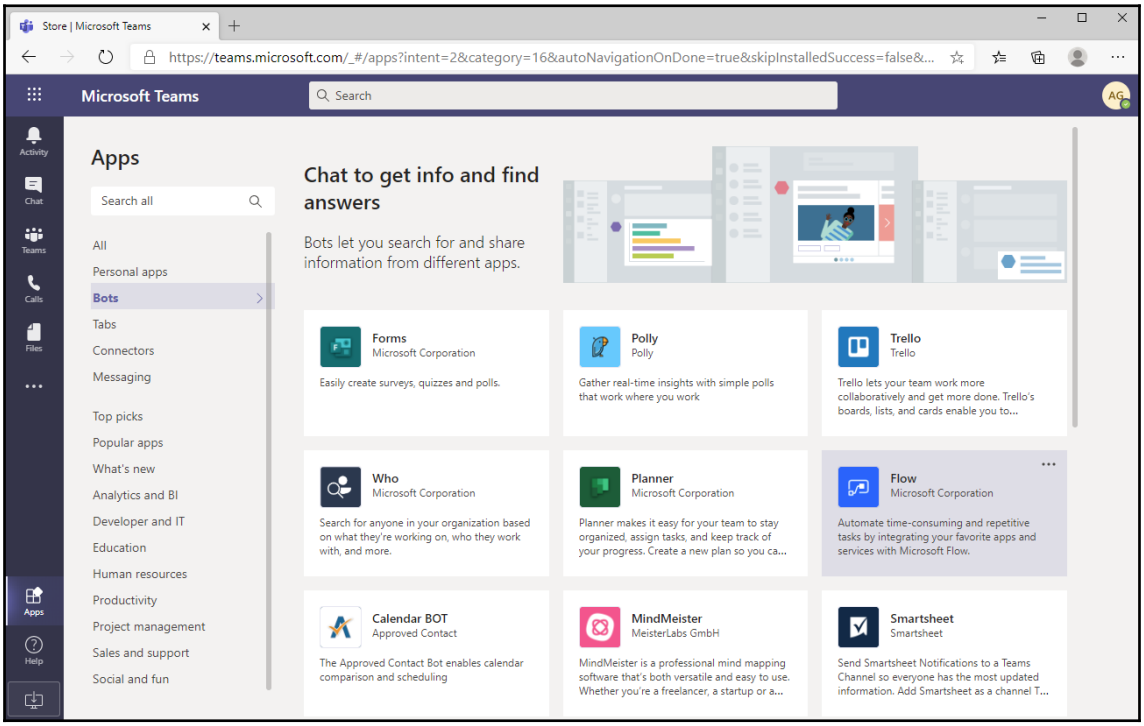


For this example, we'll be using a SharePoint site called **Procurement Requests**, but you can use any SharePoint site that you like.

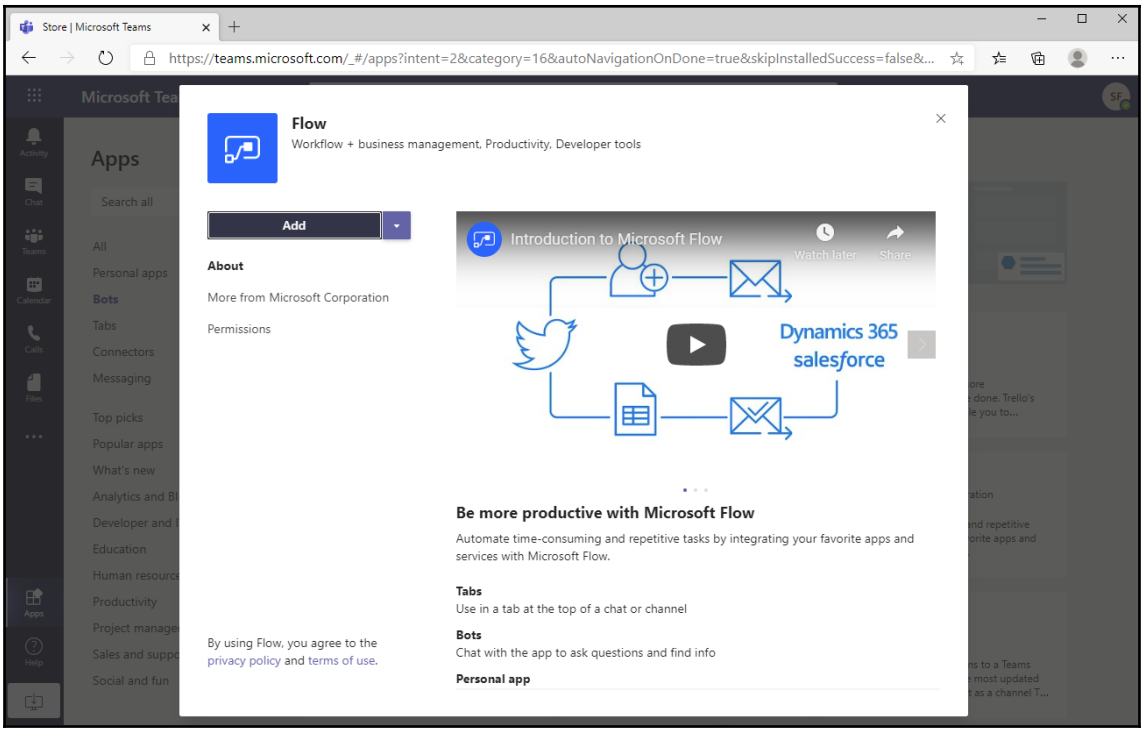
Flow user bot for Teams

The Flow Bot for Teams allows Power Automate to interact with a Microsoft Teams user. To install the bot, you can follow these steps:

1. Launch Microsoft Teams (either the desktop application version or the web version at `https://teams.microsoft.com`) and sign in.
2. On the left rail, select **Apps**, and then select **Bots**. Click the **Flow** bot:



3. On the bot app page, click **Add**:



Once the bot has been configured for the test users, you should be ready to begin configuring the flow.

Configuring an approval flow to use Teams

As we mentioned in the *Understanding the flow* section, this flow is going to be initiated by a request or a file being placed on a SharePoint site and result in approval notifications being sent to both the approver and the requester inside Teams.

Since this is a longer flow, we'll complete it in a few sections:

- Getting the request's information
- Creating the approval
- Returning the response

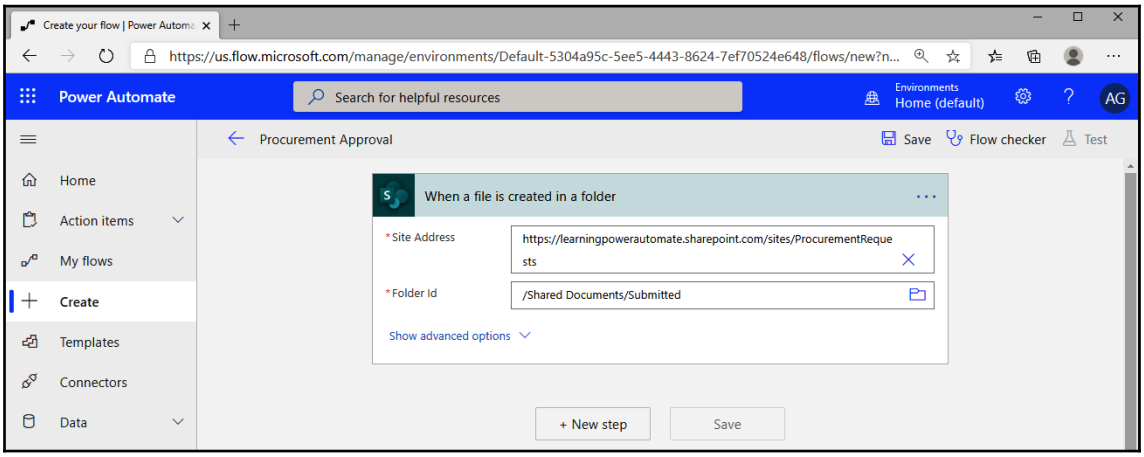
Let's begin!

Getting the request's information

In this first section, we're going to configure the trigger for the flow to detect a new file and obtain information about the file and the requester.

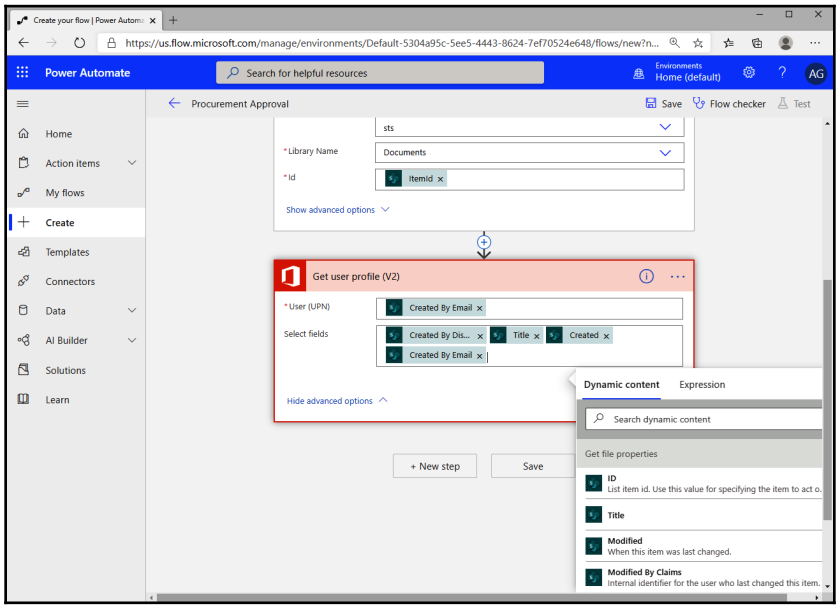
To configure the flow, follow these steps:

1. Navigate to the Power Automate web portal (`https://flow.microsoft.com`), select **+ Create**, and then select **Automated flow** under the **Start from blank** section.
2. Add a name, such as `Procurement Approval`, and then choose the **When a file is created in a folder** trigger.
3. Edit the **Site Address** field and select the site you created in the prerequisites.
4. Select the `Submitted` folder in the **Folder Id** field:



5. Click **+ New step**.
6. Select the **Get file metadata** SharePoint action. In the **Site Address** field, select the SharePoint site containing the request. In the **File Identifier** field, select the **x-ms-file-id** dynamic content token under the **When a file is created in a folder** section.
7. Click **+ New step**.

8. Select the **Get file properties** SharePoint action. In the **Site Address** field, select the SharePoint site containing the request. In the **Library Name** field, select the document library containing the `Submitted` folder. In the **Id** field, select the **ItemId** dynamic content token under the **Get file metadata** section.
9. Click **+ New step**.
10. Select the **Get user profile (V2)** Office 365 Users action. In the **User (UPN)** field, select the **Created by Email** dynamic content token under the **Get file properties** section. In the **Select fields** field, select the **Created by Displayname**, **Title**, **Created**, and **Created by Email** properties:



11. Click **+ New step**.
12. Select the **Get manager (V2)** Office 365 Users action.
13. In the **User (UPN)** field, select the **User Principal Name** dynamic content token under the **Get user profile (V2)** section.
14. Click **Save**.

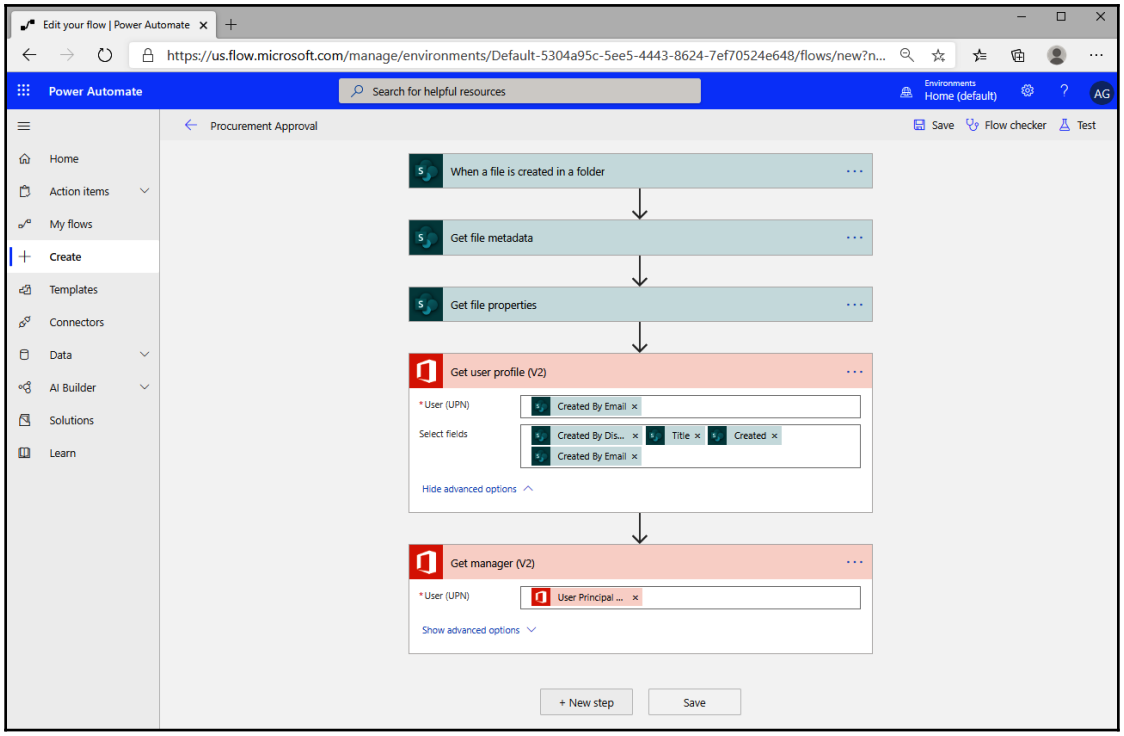
At this point, you will have created the part of the flow that gathers information about the request and the requester.

Next, we'll configure the approval request.

Creating the approval

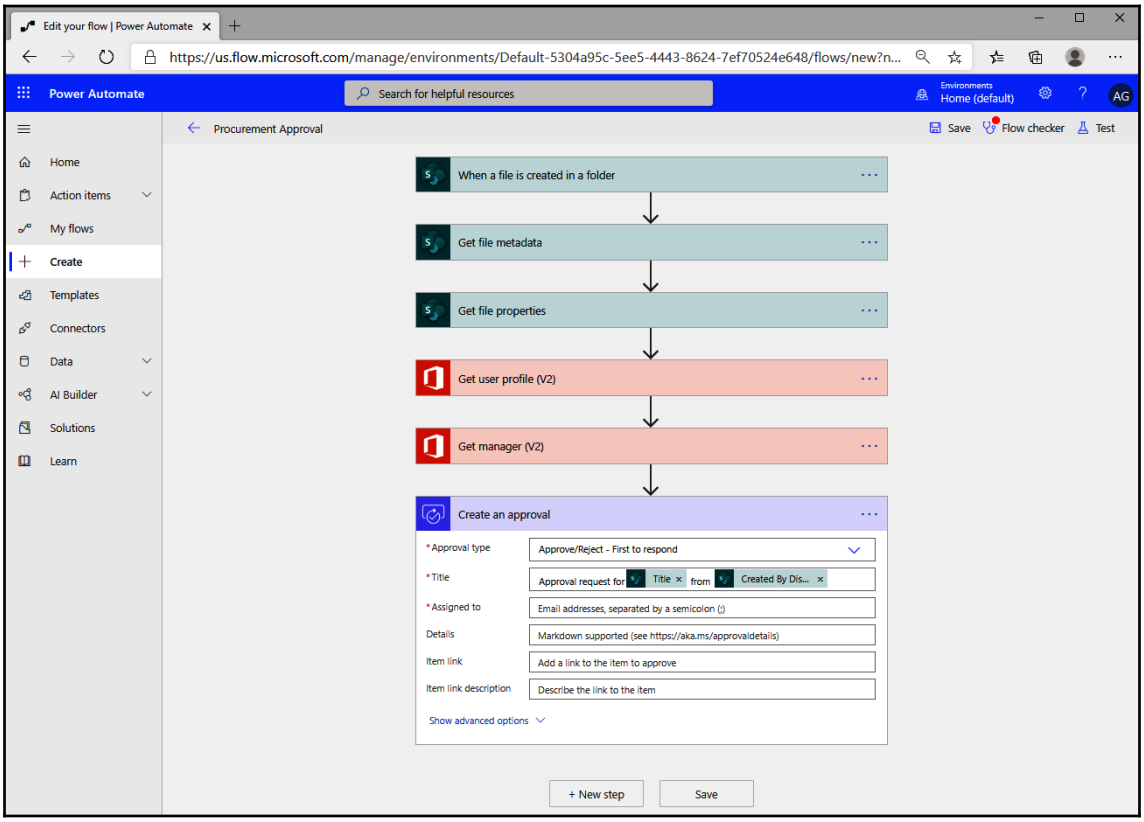
In this section, you'll continue creating the approval from the previous section by adding and configuring the approval action. To complete this part of the flow, ensure that the flow from the previous section is open for editing, and follow these steps:

1. Click **+ New step** after the **Get manager (V2)** action:

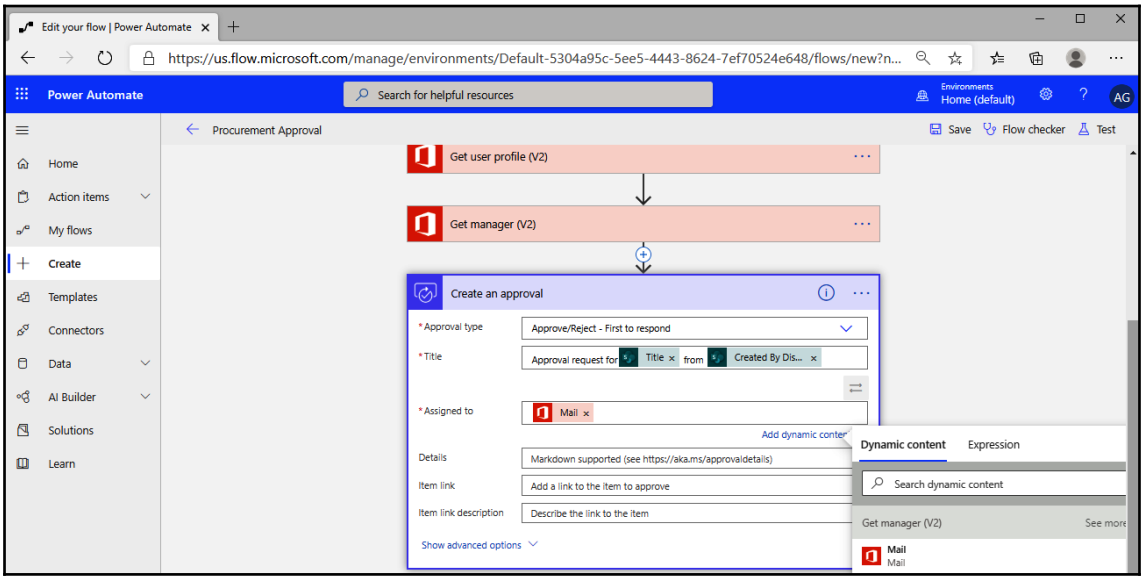


2. Select the **Create an approval** Approvals action.
3. Under **Approval type**, select **Approve/Reject - First to respond**.

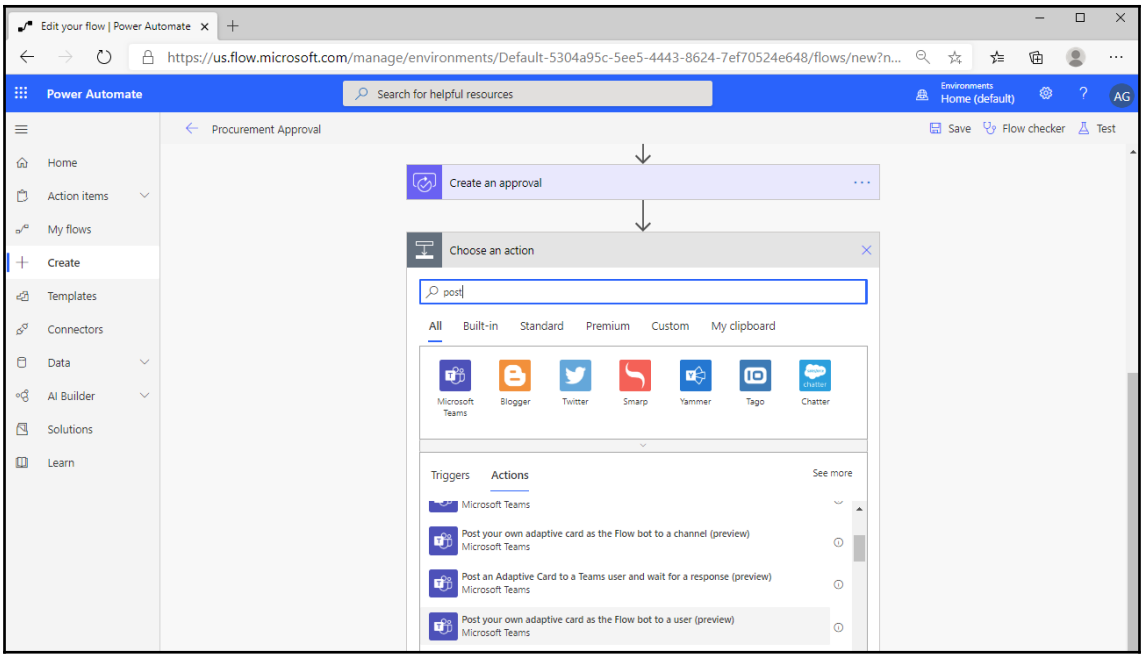
4. Under **Title**, enter the text that uniquely identifies this request. In this example, we selected the **Title** and **Created by Displayname** dynamic content tokens:



5. In the **Assigned to** field, select the dynamic content token value for **Mail** under the **Get manager** section:

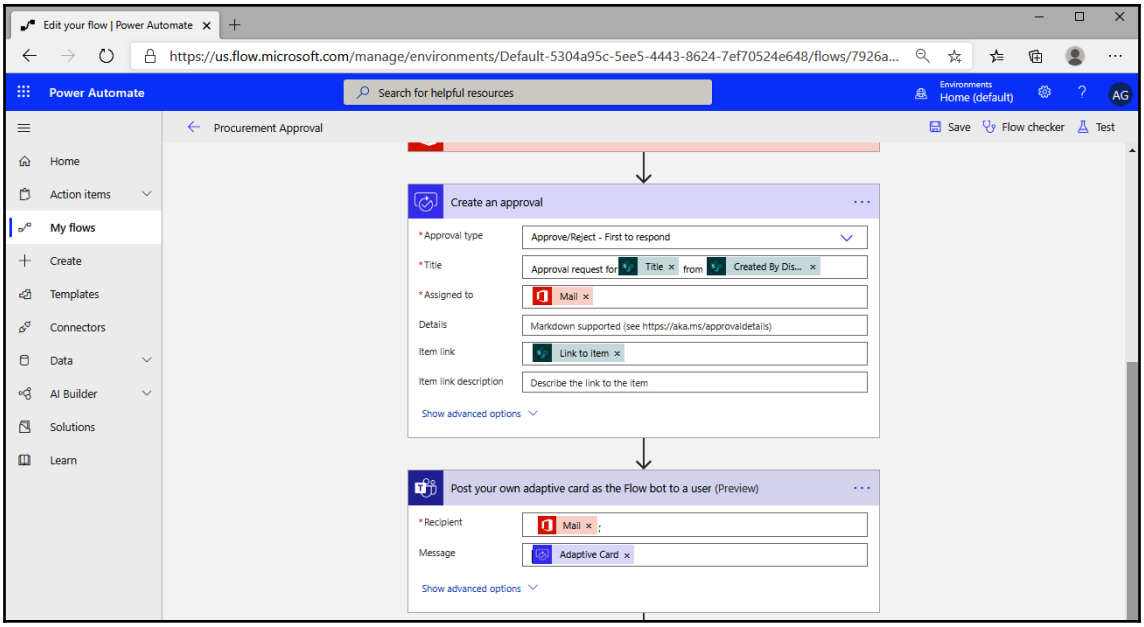


6. In the **Item link** field, you can optionally add an item, such as the **Link to item** dynamic content token under the **Get file properties** section.
7. Click **+ New step**.
8. Select the **Post your own adaptive card as the Flow bot to a user (preview)** Microsoft Teams action:

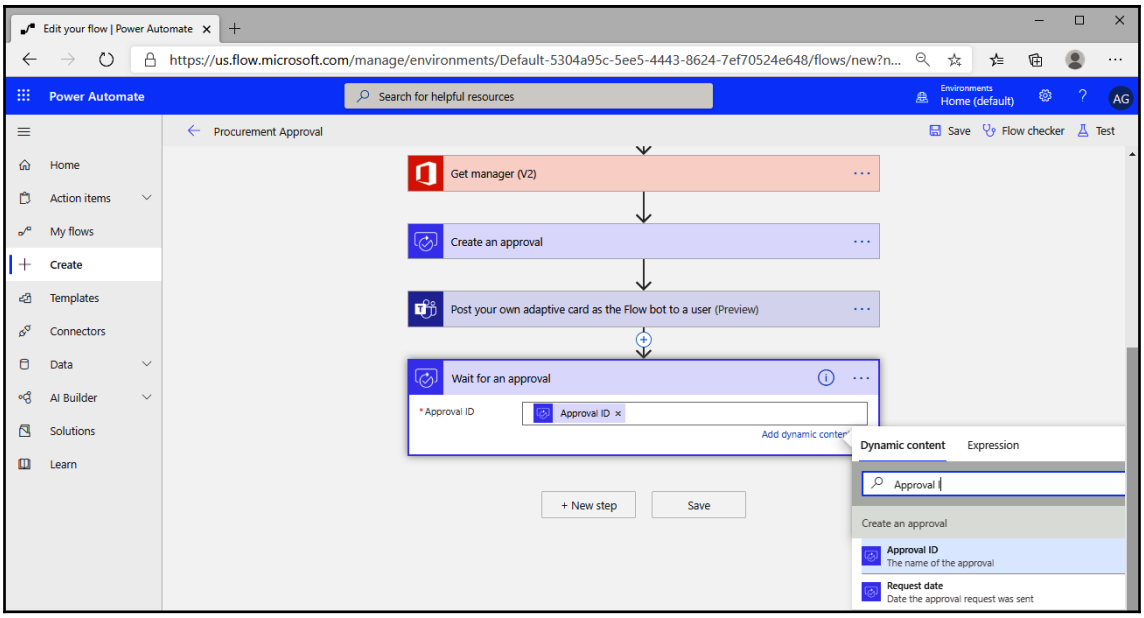


9. In the **Recipient** field, select the dynamic content token for **Mail** under the **Get manager** action.

10. In the **Message** field, select the **Adaptive Card** dynamic content token:



11. Click **+ New step**.
12. Select the **Wait for an approval** Approvals action.
13. In the **Approval ID** field, select the **Approval ID** dynamic content token under **Create an approval**:



14. Click **Save**.

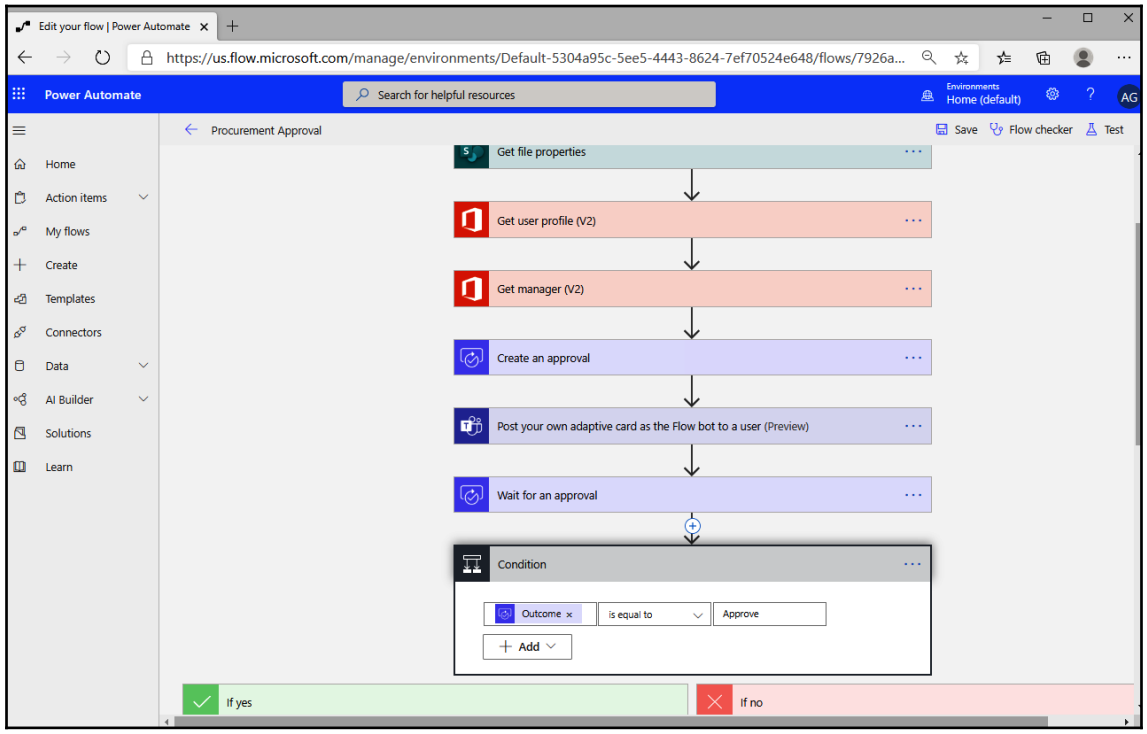
You've now configured an approval workflow to begin, post a card to the Teams user, and wait for their response. In the next section, we'll evaluate the response and decide what actions to take.

Returning the response

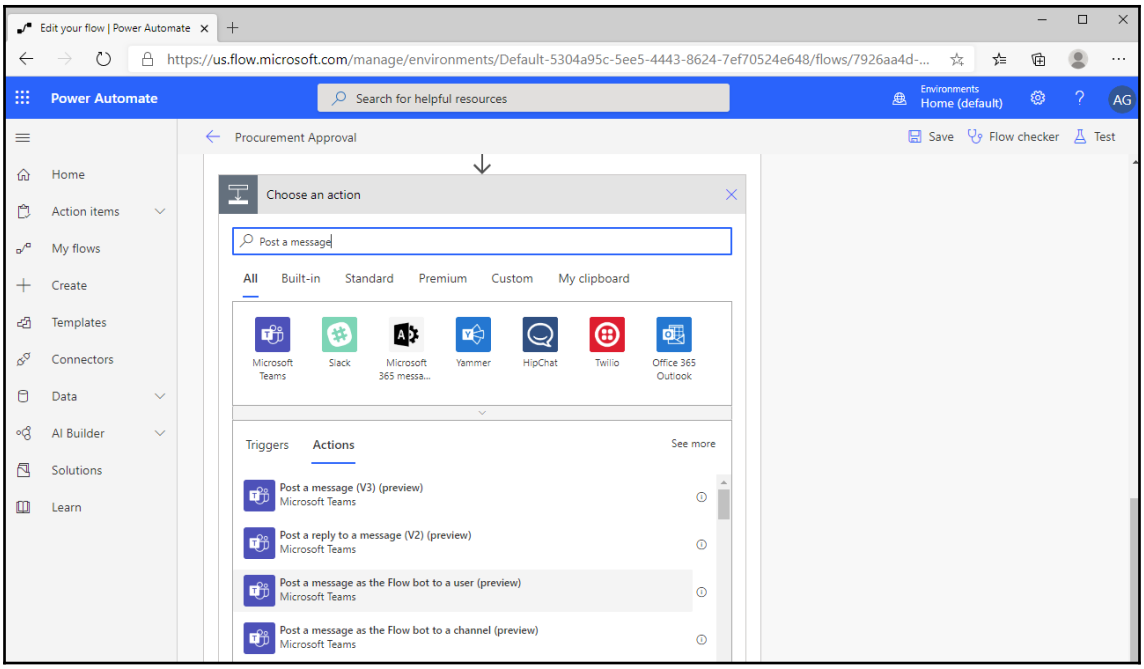
In this final section, we'll configure the actions based on the approver's response. To complete this part of the flow, ensure that the flow from the previous section is open for editing, and follow these steps:

1. After the **Wait for an approval** action, click **+ New step**.
2. Select the **Condition** control.

3. In the **Choose a value** box on the left side of the control, select the **Outcome** dynamic content token:

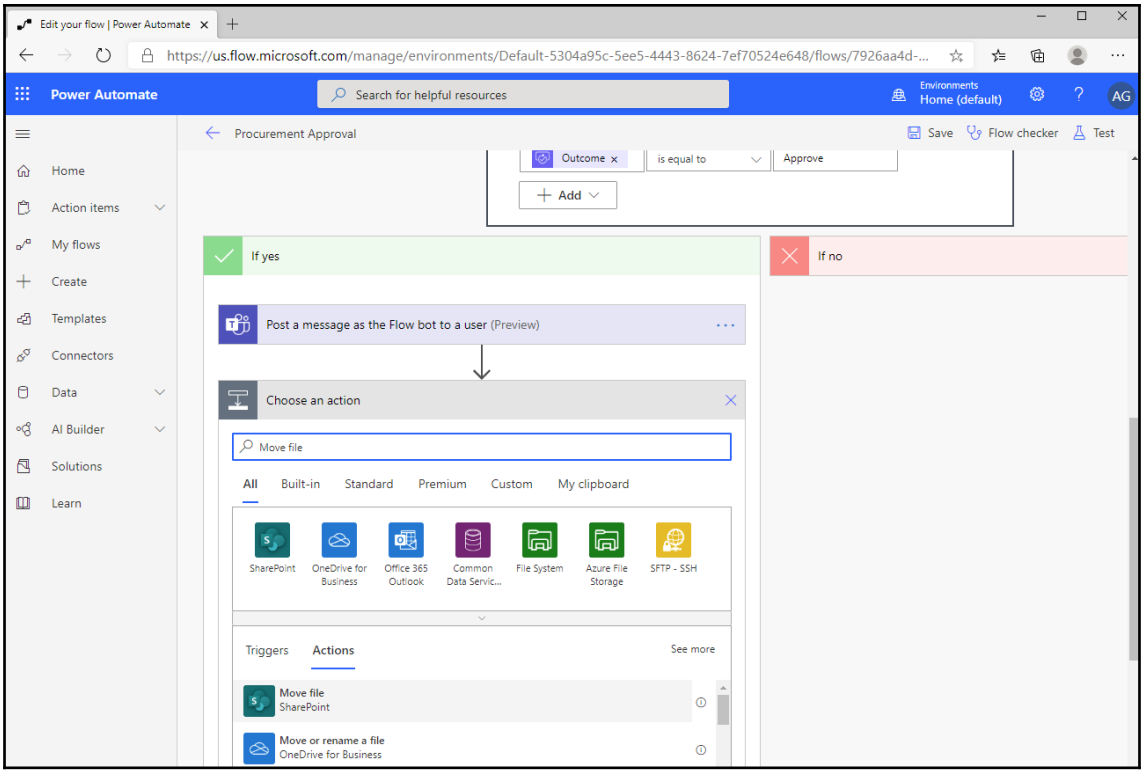


4. Select the **is equal to** evaluation method for the condition.
5. Enter the text `Approve` in the right-hand box of the condition.
6. In the **If yes** branch of the condition, click **Add an action**.
7. Select the **Post a message as the Flow bot to a user (Preview)** action:

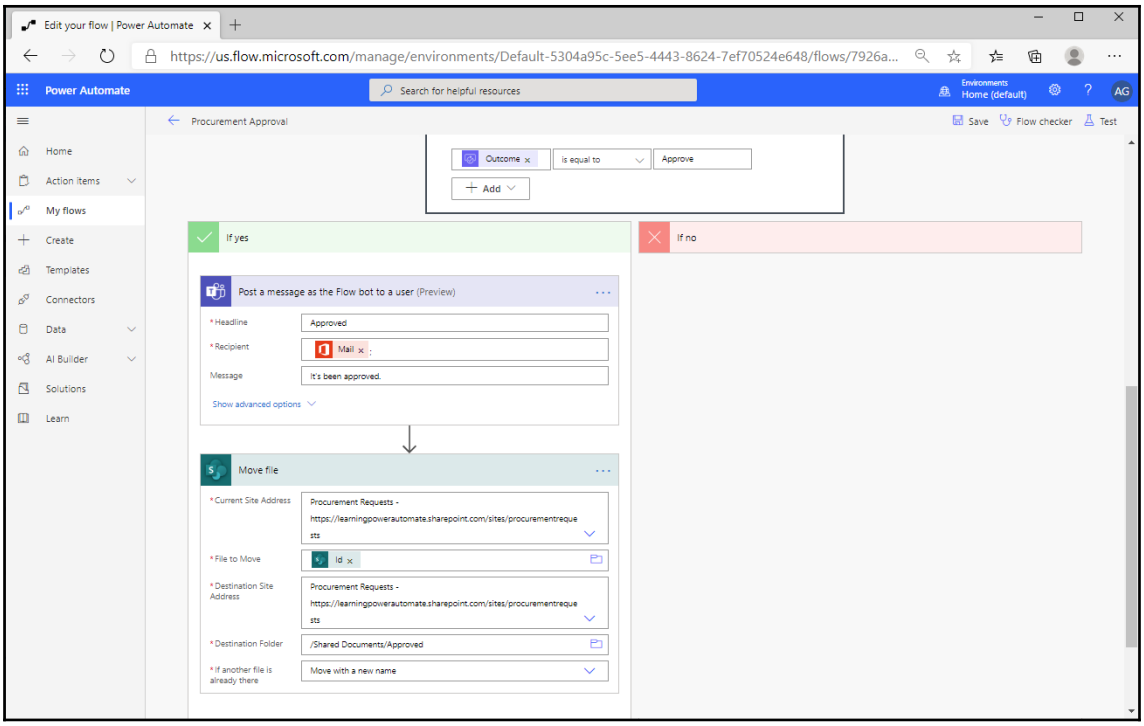


8. In the **Headline** field, add a value that will be displayed as the headline or title of the message sent to the *requester*.
9. In the **Recipient** field, add the **Mail** dynamic content token under the **Get user profile** section. This will send the response to the *requester*.
10. In the **Message** field, enter a value that will be posted in the body of the message to the *requester*.
11. In the **Yes** branch, select **Add an action**.

12. Select the **Move file** SharePoint action:

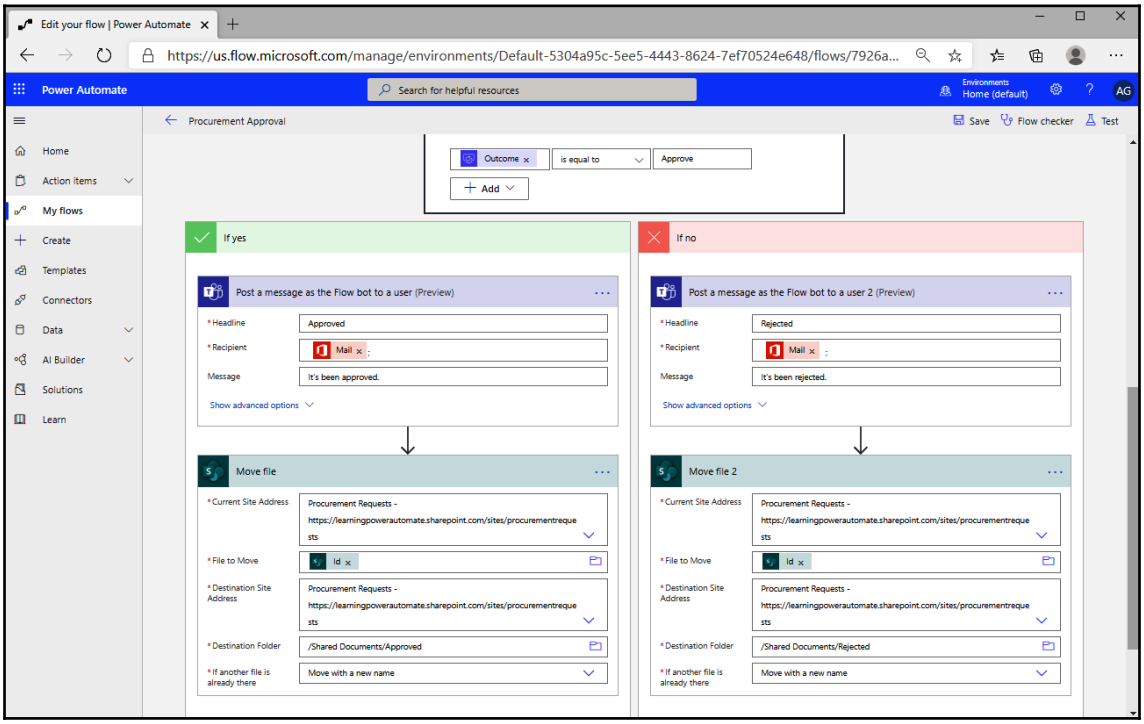


13. In the **Current Site Address** field, select the SharePoint site used in the flow.
14. In the **File to Move** field, select the **Id** dynamic content token that represents the file that triggered the flow (you can use a number of tokens, such as **Id** or **Identifier**). In the **Destination Site Address** field, select the SharePoint site used in the flow.
15. For **Destination Folder**, select the `Approved` folder. The completed branch should look similar to the following screenshot:



16. In the **If no** branch, select **Add an action**.
17. In the **Headline** field, add a value that will be displayed as the headline or title of the message sent to the *requester*.
18. In the **Recipient** field, add the **Mail** dynamic content token under the **Get user profile** section. This will send the response to the *requester*.
19. In the **Message** field, enter a value that will be posted in the body of the message to the *requester*.
20. In the **If no** branch, select **Add an action**.
21. Select the **Move file** SharePoint action.
22. In the **Current Site Address** field, select the SharePoint site used in the flow.

23. In the **File to Move** field, select the **Id** dynamic content token that represents the file that triggered the flow (you can use a number of tokens, such as **Id** or **Identifier**).
24. In the **Destination Site Address** field, select the SharePoint site used in the flow.
25. For **Destination Folder**, select the `Rejected` folder. The complete branches should look similar to the following screenshot:



26. Click **Save**.

You've now configured the flow to post a message to the requester and move the file that triggered to flow to the `Approved` or `Rejected` folder, depending on the outcome.

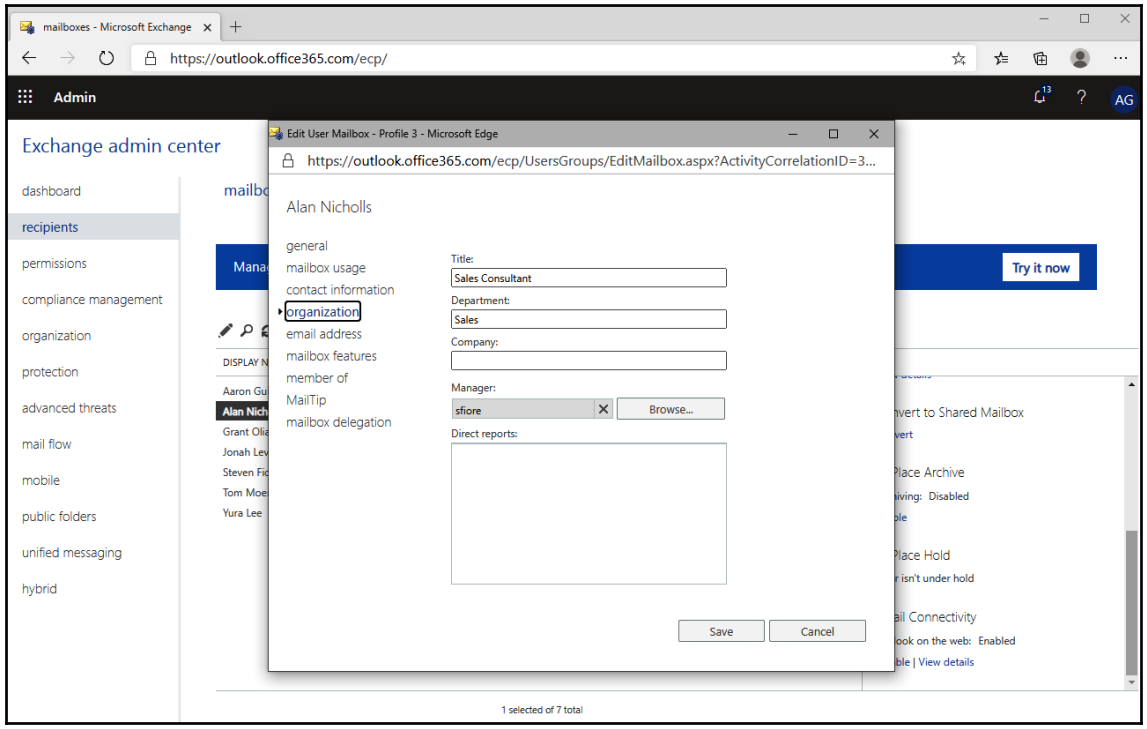
Next, we'll test the flow!

Testing the flow

In order to test this flow, you'll need to meet the following prerequisites:

- At least two users with Exchange Online and Microsoft Teams enabled.
- Users should have the Flow bot enabled.
- One of the users should be configured as a *manager* and one of the users should be configured to be the *direct report*.
- At least two browser sessions, one logged in as *manager or approver*, and one logged in as *direct report or requester*.

You can use the Exchange Online `Set-User` cmdlet (`https://docs.microsoft.com/en-us/powershell/module/exchange/set-user`) to set the manager property and establish the manager/direct relationship. You can also use the Exchange Online Admin Center to set the manager property of a user, which configures the manager/direct report relationship:



If you don't have the capability to set the manager property, you may need to edit the flow to use a particular email address of a user to use as the approver.

The testing process will involve three core steps:

- Requesting the approval
- Approving the request
- Reviewing the response

Let's start testing!

Requesting the approval

The process must be started by the requester. The requester must have the manager field populated in Azure AD. Once this requirement is met, follow these steps:

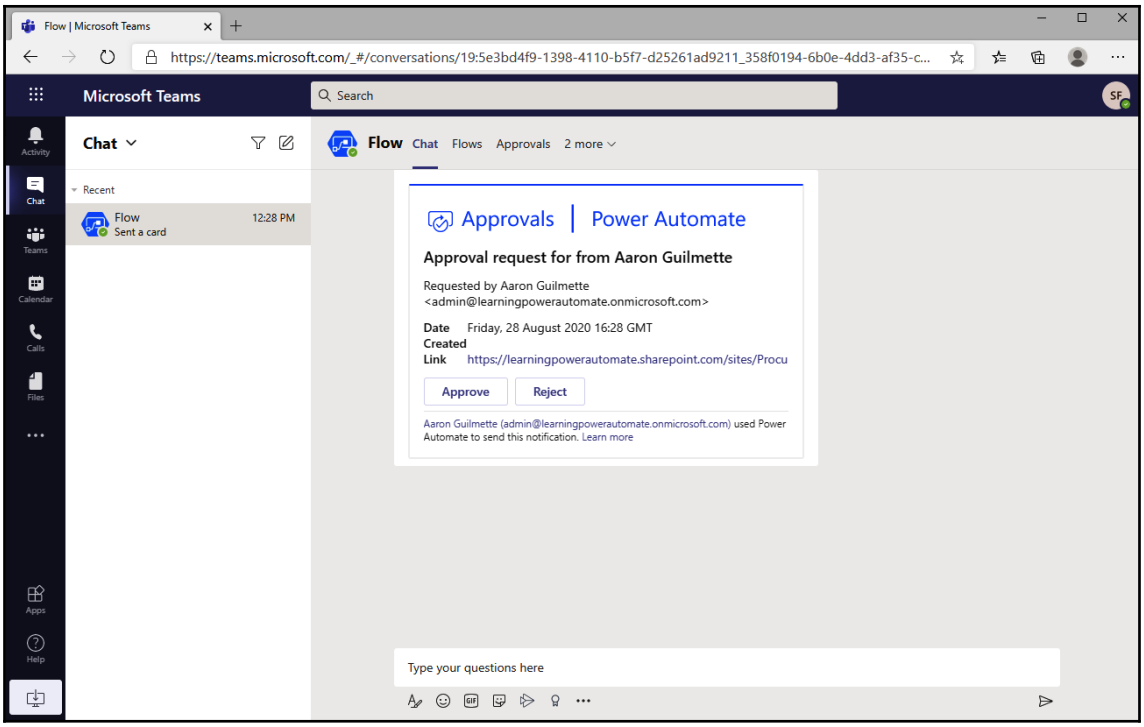
1. Using a separate browser, log in to Office 365 as the *requester* user.
2. Browse to the SharePoint site that is being monitored by the flow.
3. Upload a file to the `Submitted` folder in the SharePoint site being monitored for a file.

This activity will trigger the flow.

Approving the request

For this process, you must have a separate browser session logged in as the *approver*:

1. When logged in to a browser session to Office 365 as the *approver*, open Microsoft Teams (`https://teams.microsoft.com`).
2. Select **Chat** from the left rail and select a message from the Flow Bot.
3. To approve or reject the request, click the **Approve** or **Reject** button on the adaptive card:



4. Optionally, enter an approval or rejection comments. Click **Submit**.

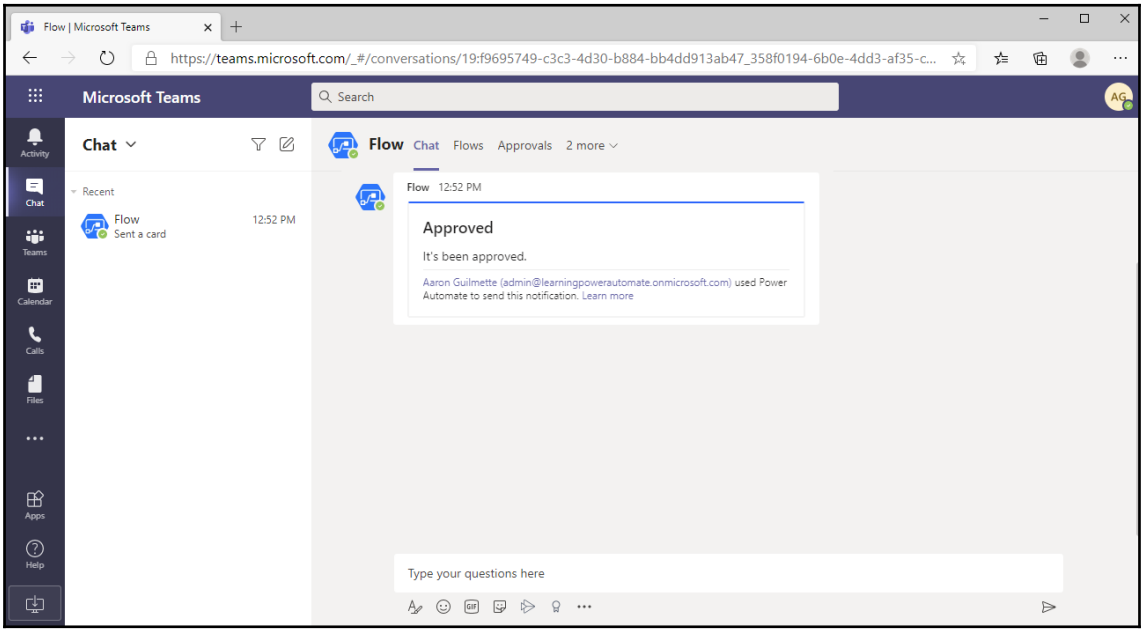
Once this is done, it will trigger the next part of the flow.

Reviewing the response

In this final step, the requester should be able to view the approver's response:

1. When logged in to a browser session to Office 365 as the *requester*, launch Teams (`https://teams.microsoft.com`).
2. Select **Chat** from the left rail and select a message from the Flow bot.

3. Review the message:



The approval process has been completed.

The adaptive cards and flow messages can be customized. As you gain familiarity and confidence with dynamic content value tokens, formatting, and expressions, you can build on these examples to make cards and messages more personalized and detailed.

Summary

In this chapter, you built on the knowledge that you've gained throughout the book to build an approvals integration into Microsoft Teams. You used familiar components, such as the SharePoint file creation trigger and conditions, and expanded into using adaptive cards and the Flow bot for Microsoft Teams. Using this type of design keeps approvers from task-switching or application-switching out of Microsoft Teams to complete approvals, which will help your organization improve productivity.

Finally, you were able to test the flow and see it from the perspective of both the requester and the approver.

In the next chapter, you'll learn how to start using databases in flows.

12
Using a Database

Databases are critical infrastructure components that are used to compile information about tasks, individuals, services, products, or any other entity that needs to be tracked. Common database-driven applications include **customer relationship management** (**CRM**), product catalogs, **enterprise resource planning** (**ERP**), accounting packages, storefronts, and inventory management.

In the world of Power Automate, you might use a database to store information about survey or form result responses or to manage scheduled jobs. You may even use one as a logging repository for actions taken by other applications and users. In this chapter, we're going to look at some of the basics of connecting to a database and writing data entries. Specifically, we'll learn about the following topics:

- Understanding database connectors, triggers, and actions
- Adding database content
- Creating a connection to a database
- Creating a flow to add content to a database

By the end of this chapter, you will be able to understand the basics of how to connect and send data to a database with Power Automate.

Off we go!

Understanding database connectors, triggers, and actions

Power Automate can connect to a number of databases, such as Microsoft SQL Server, Azure SQL Datawarehouse, Postgres SQL, MySQL, DB2, Oracle, and Xoaa Blockchain (as well as the CDS, which we discussed back in `Chapter 9`, *Getting Started with Approvals*). Each of these database technologies shares similar mechanics and concepts. In the examples provided in this book, we're going to focus on using Microsoft SQL Server databases and the related connector, but you may want or need to use others, depending on what you have available.

As with other connectors, the SQL connector has both triggers and actions. We'll examine these next.

Triggers

The Microsoft SQL Server connector supports two triggers (at the time of writing):

- When an item is created (v2)
- When an item is modified (v2)

When working with both of these connectors, you'll need to supply a server name, database name, and table name, as well as optional select and filter query statements to help refine your queries.

Actions

When working with a database, there are many actions you can perform. The most common actions that we'll be working with are as follows:

- **Get row/Get rows**: Retrieves one or more rows from a table
- **Execute a SQL query**: Runs a SQL query
- **Execute a stored procedure**: Runs an existing stored procedure
- **Delete row**: Deletes a row from a table
- **Get tables**: List tables in a database
- **Insert row**: Adds a new row to a table
- **Update row**: Modifies an existing row in a database table

Complex transactions can be made using the "Execute a SQL query" and "Execute a stored procedure" actions.

Now that you know what triggers and actions are available, let's get connected to a database and make a simple transaction.

Adding database content

In this section, we'll look at creating a simple button flow that adds a row of data to a database table. To do this, you'll need a database and a table. In the following example, we'll create a SQL instance and a database that we'll use for our Power Automate configuration.

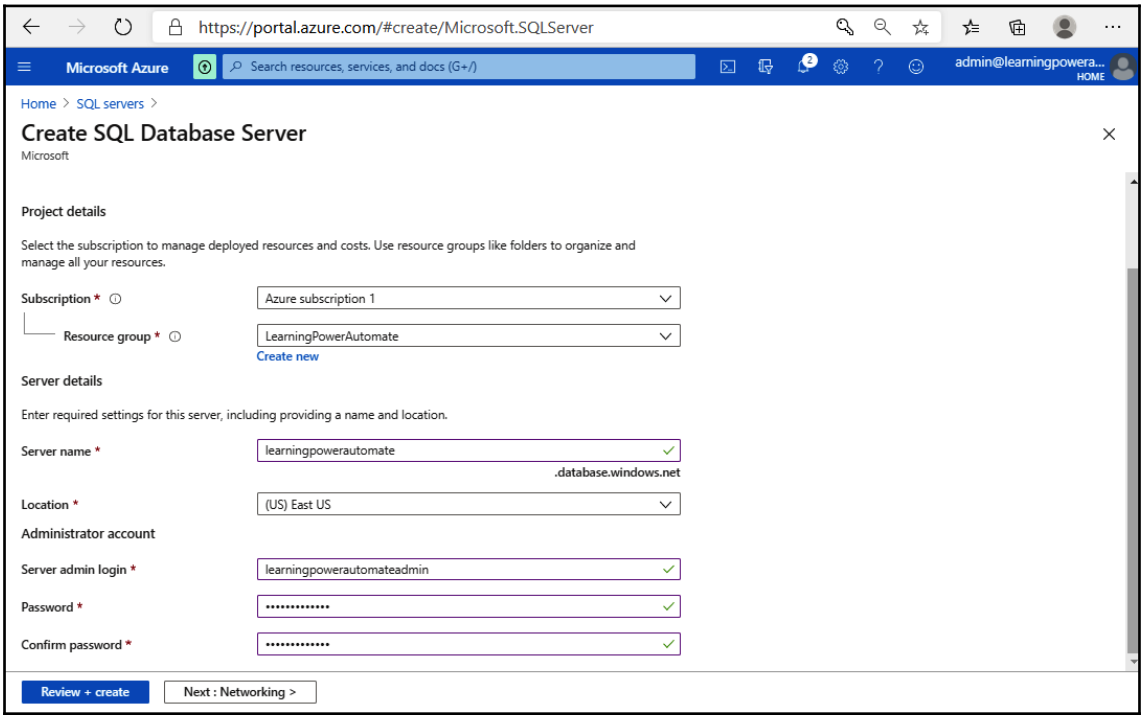
Creating a server

While you can use any supported database technology with Power Automate, we're going to focus on SQL Server. If you already have an existing SQL server available, you can skip this step. Otherwise, you can provision the necessary components in Azure to start working immediately.

To create a SQL server in Azure, follow these steps:

1. Navigate to `https://portal.azure.com` and sign into a tenant using an account with administrative privileges. (If you don't currently have an Azure subscription, you can sign up for a trial).
2. From the search bar, begin typing `SQL` and select **SQL Servers**.
3. Click **+Add**.
4. Select a subscription and a resource group. If you do not have a resource group, you can create one.

5. Enter a server name, location, and credentials for administration. Then, click **Next: Networking >**:



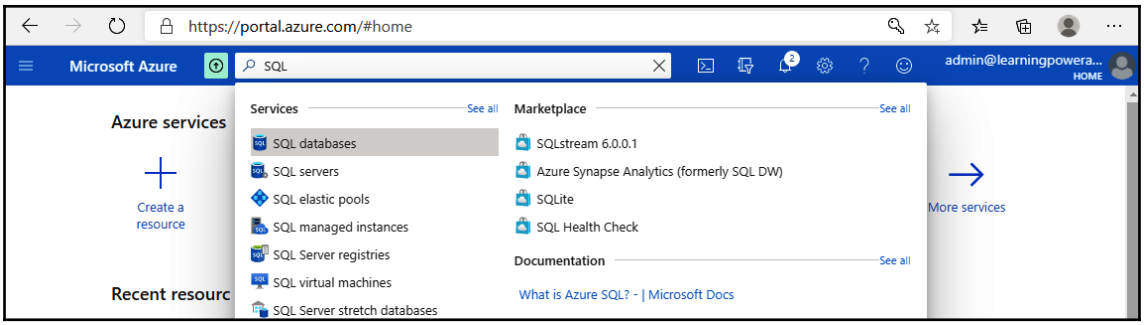
6. On the **Networking** tab, enable the slider for firewall rules to allow access to Azure services.
7. Click **Review + create** to create your server.

Now that you have finished creating the server, you can start creating a database for it.

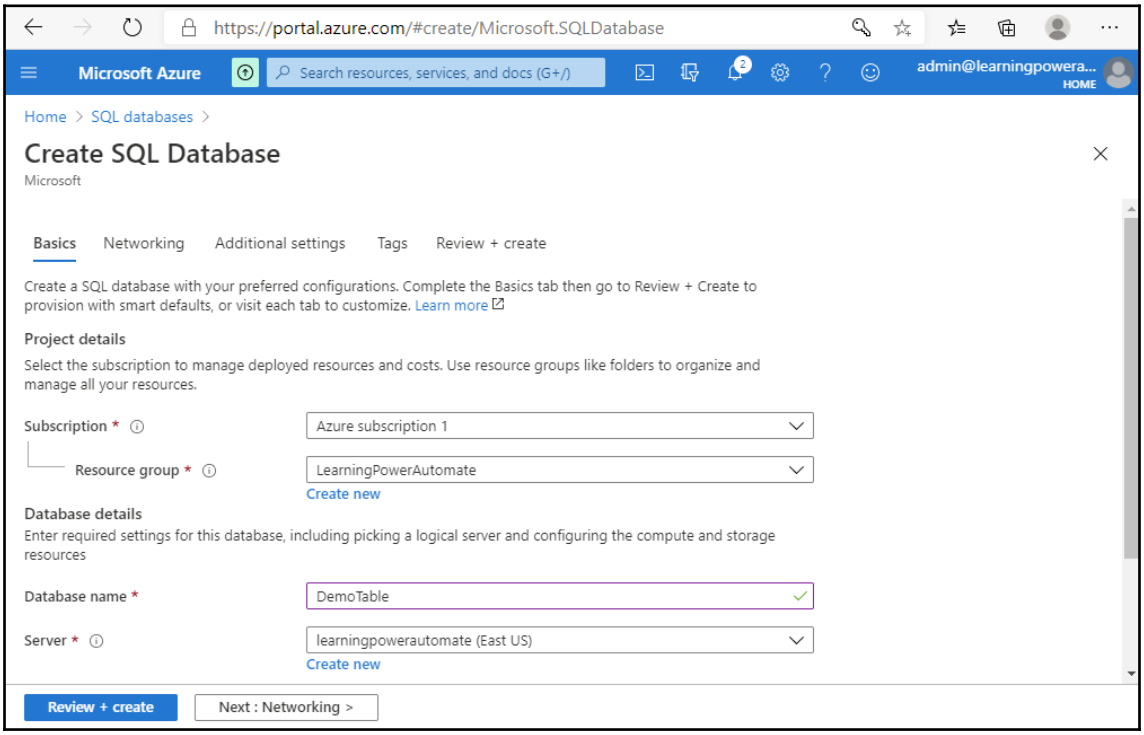
Creating a database

Now that you have a SQL server instance provisioned, you'll need a database. If you already have a database that you can use, you can skip this step. Otherwise, you can use the following process to create a new database on the instance that you provisioned in the previous subsection:

1. From the Azure portal, click the search bar and begin typing `SQL databases`. Select the **SQL databases** option:



2. Click **+ Add** to create a new database.
3. Under **Project details**, select a subscription and a resource group. For simplicity's sake, use the same subscription and resource group that the server resides in.
4. Under **Database details**, enter a value for the new database name. Select a server. If you created a server in the previous subsection, select that server here. Click **Review + create** to continue:



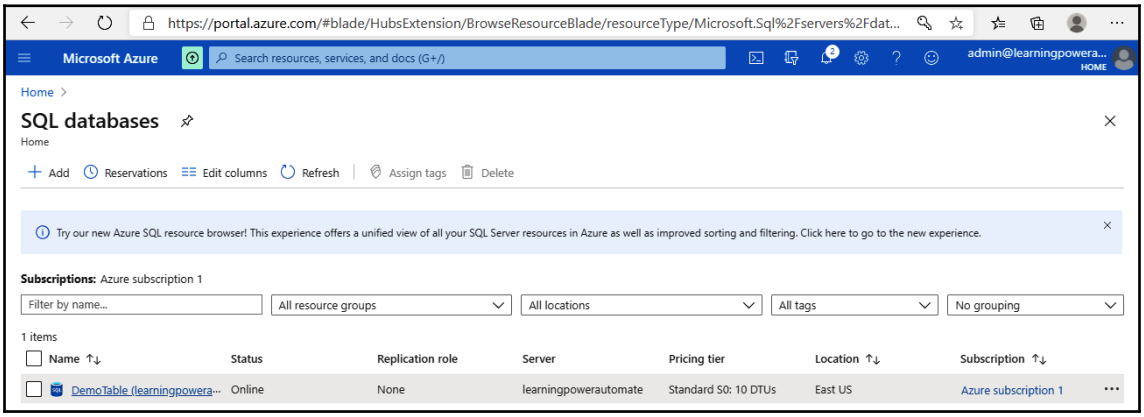
5. Click **Create**.

The wizard will complete the process and create a database.

Creating a database table

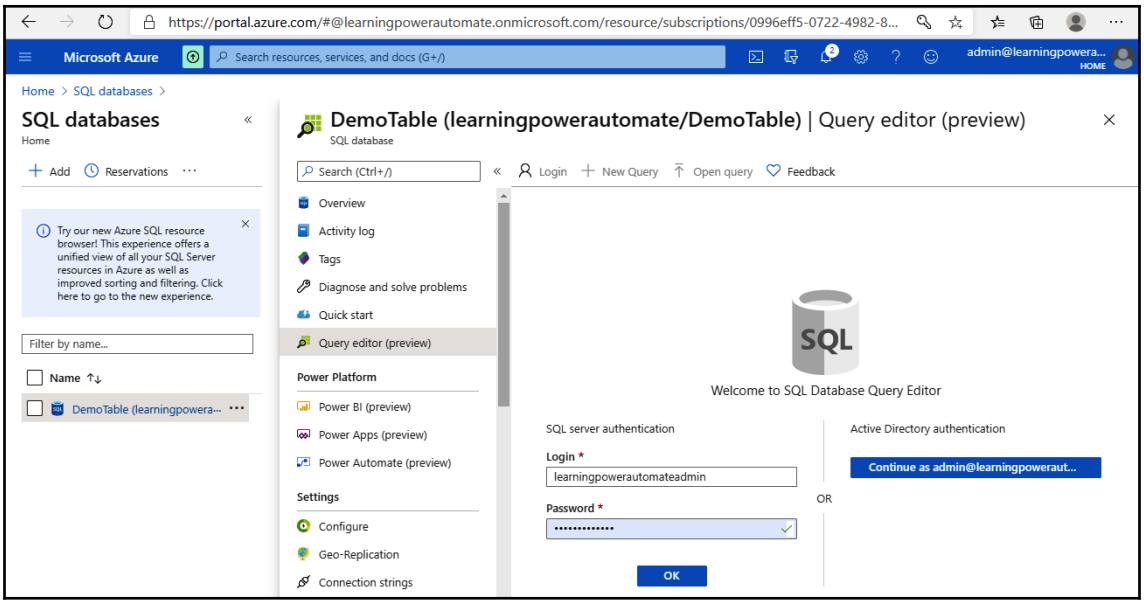
In order for you to have a place to add content, the database will need a database table. You can create a database table by using the Query editor and following these steps:

1. From the Azure portal, click the search bar and begin typing `SQL databases`. Select the **SQL databases** option.
2. Select the database that you created in the previous subsection:



3. Select **Query editor (preview)**. Enter your login credentials to access the database and then click **OK**:

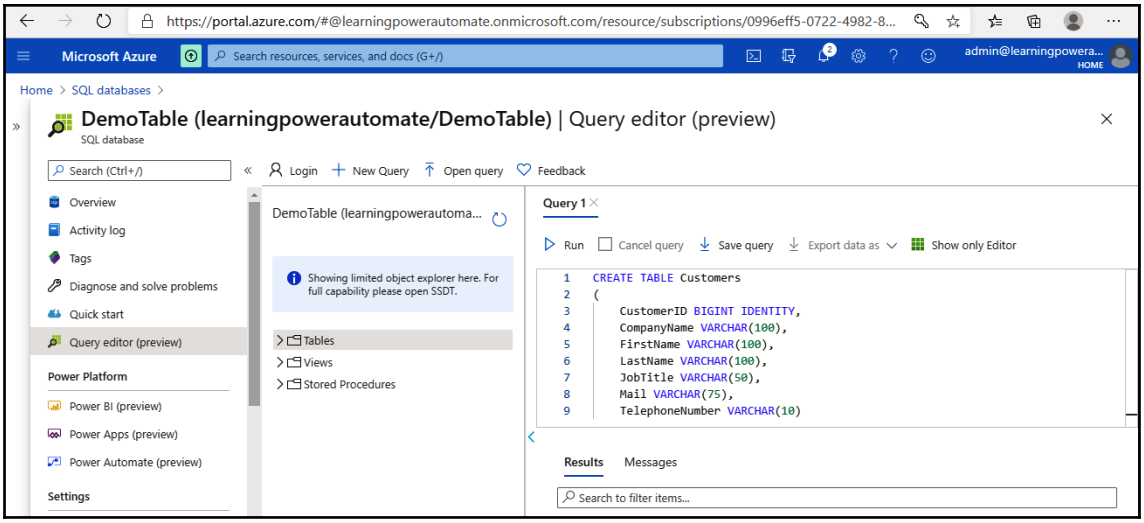
If prompted, you may need to enable a firewall rule to allow your local computer to communicate with the Query editor.



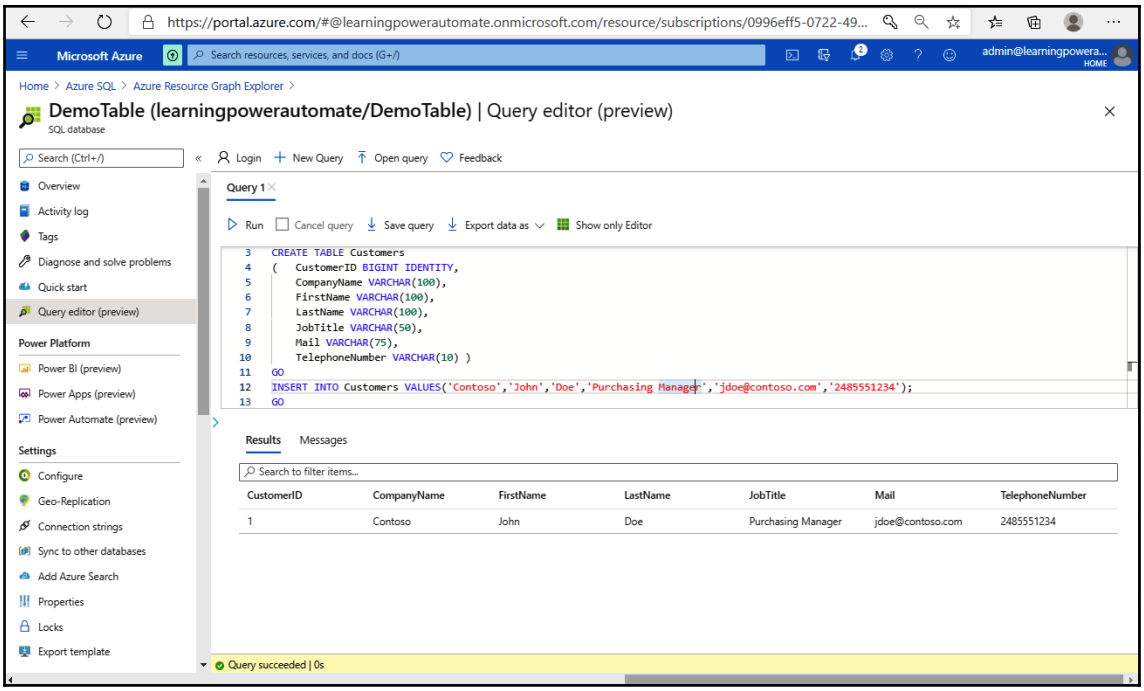
4. Copy and paste the following code into **Query editor**:

```
CREATE TABLE Customers
(
    CustomerID BIGINT IDENTITY,
    CompanyName VARCHAR(100),
    FirstName VARCHAR(100),
    LastName VARCHAR(100),
    JobTitle VARCHAR(50),
    Mail VARCHAR(75),
    TelephoneNumber VARCHAR(10)
)
GO
INSERT INTO Customers VALUES('Contoso','John','Doe','Purchasing
Manager','jdoe@contoso.com','2485551234');
GO
```

5. Click **Run**:



At this point, you should now have a server, a database, and a table that contains some content. This database table should have seven columns (`CustomerID`, `CompanyName`, `FirstName`, `LastName`, `JobTitle`, `Mail`, and `TelephoneNumber`) and one row of data. To verify that the table contains data, in the Query window, enter `SELECT * FROM Customers;` and click **Run**. The result is shown in the **Results** section of the following screenshot:



If everything was successful, you'll have a sample database configured and ready to use as a destination for Power Automate.

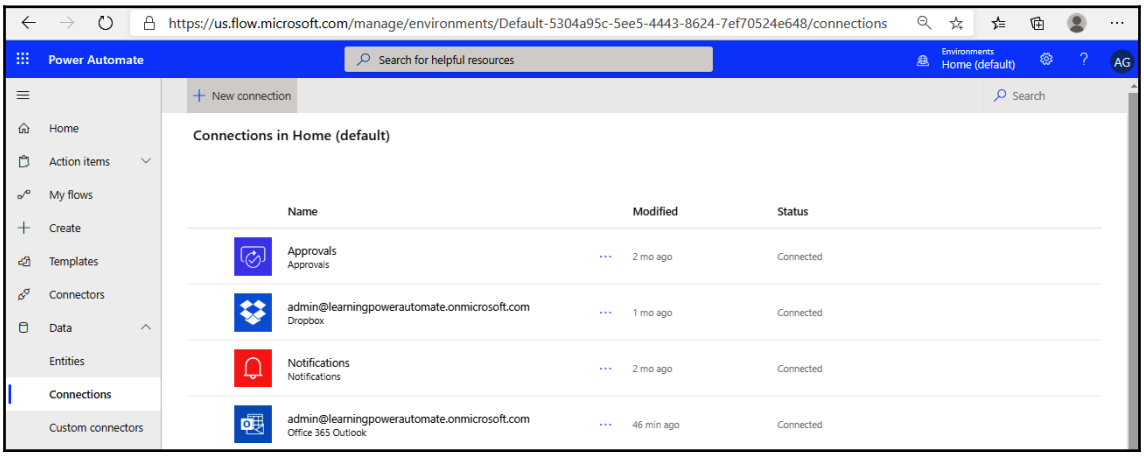
Next, we'll establish a connection to the database in Power Automate.

Creating a connection to a database

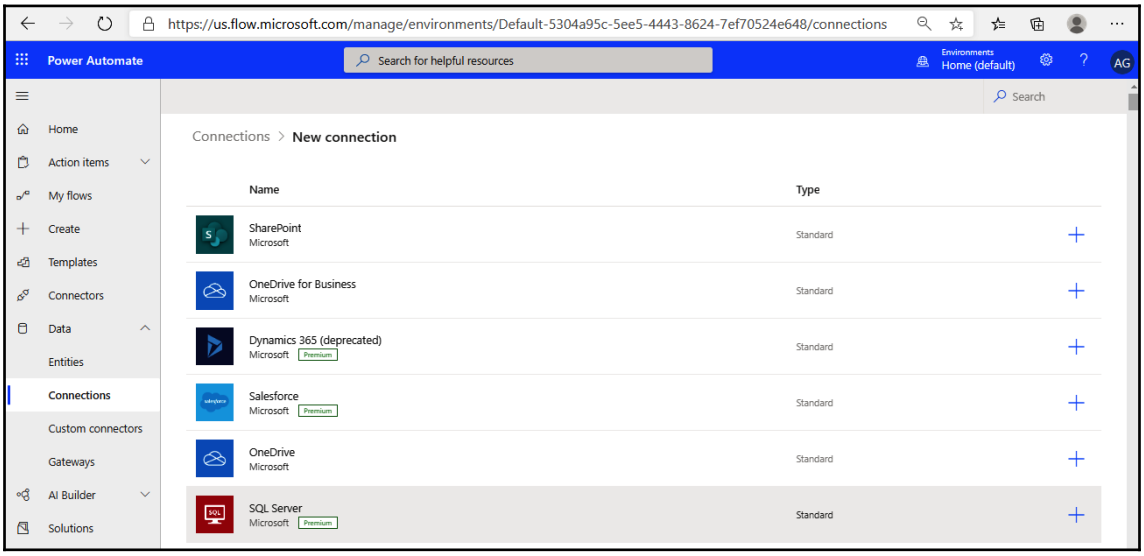
When working with SQL Server (or other database technologies), you'll have to establish a connection to a database. Connection details will depend on the type of database you're connecting to, but the most common fields or properties that you'll need to populate include a server name or IP address, credentials, port numbers, and a database name.

To connect to a database in Power Automate, follow these steps:

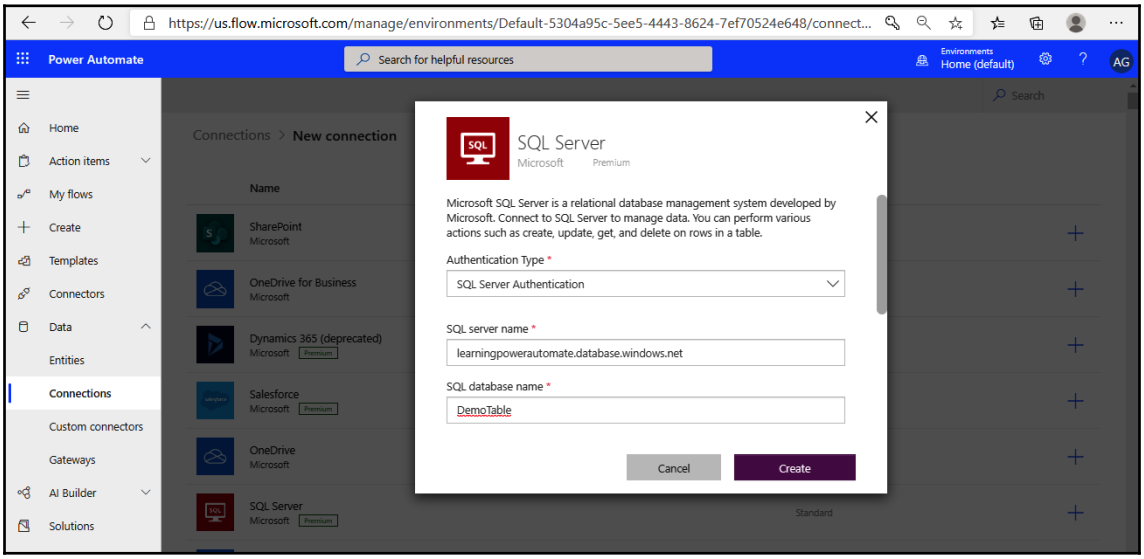
1. Navigate to the Power Automate web portal (`https://flow.microsoft.com`). Expand **Data** and click **Connections**.
2. Select **+ New connection**:



3. Select **SQL Server** from the list of connection types:



4. Fill out the dialog box using the server name, database name, and the credentials you created in the previous sections:



5. Click **Create**.

Now that a connection has been established, you can create a simple button flow in order to place information in the database.

Adding content to a database

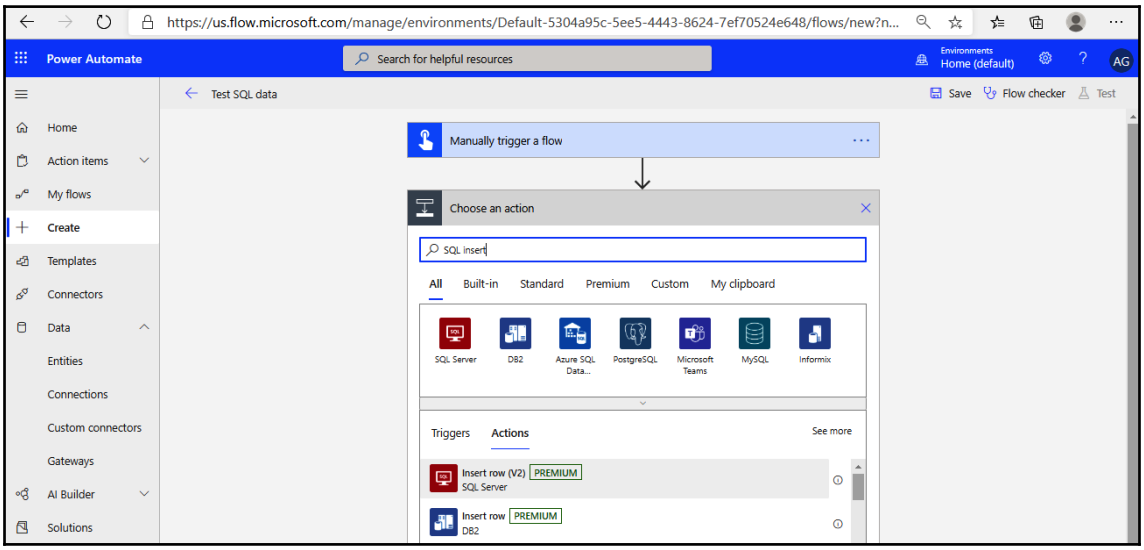
Now that you have configured a server, a database, and a table with columns, you get to see all the pieces working together. In this section, you'll create a flow, execute it, and then verify that it wrote the data successfully.

First, let's create a flow so that we can add content.

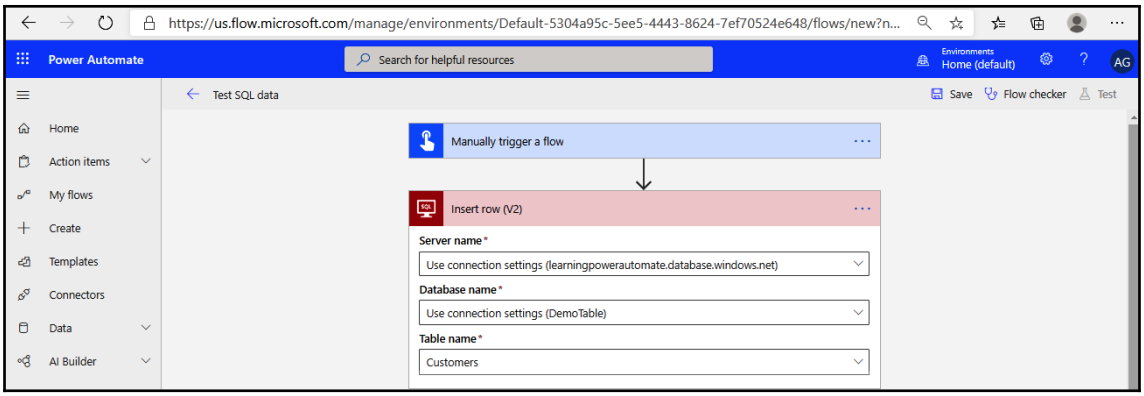
Creating the flow

To verify that you've configured database connectivity, we'll walk through creating a sample button or instant flow so that we can post data to the database. Follow these steps to create the flow:

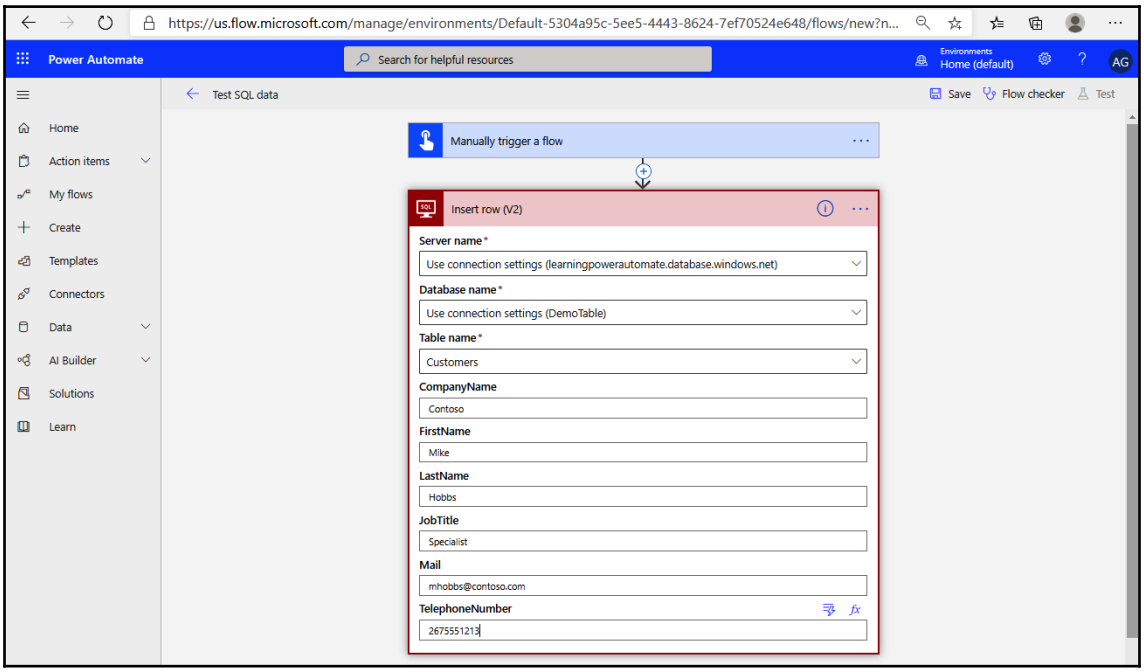
1. From the Power Automate web portal (`https://flow.microsoft.com`), select **+ Create**.
2. Under the **Start from blank** section, select **Instant flow**.
3. Enter a name for the flow, and then select **Manually trigger a flow** as the trigger type.
4. Click **Create**.
5. Click **+ New step**.
6. In the search box, enter `SQL insert` and then select the **Insert row (V2)** action:



7. From the dropdowns for the action, select the server name, database name, and the table name you created in the *Adding database content* section:



8. Once you've selected the items from your connection, the action will update to display all the available columns. Add some test data and click **Save**:

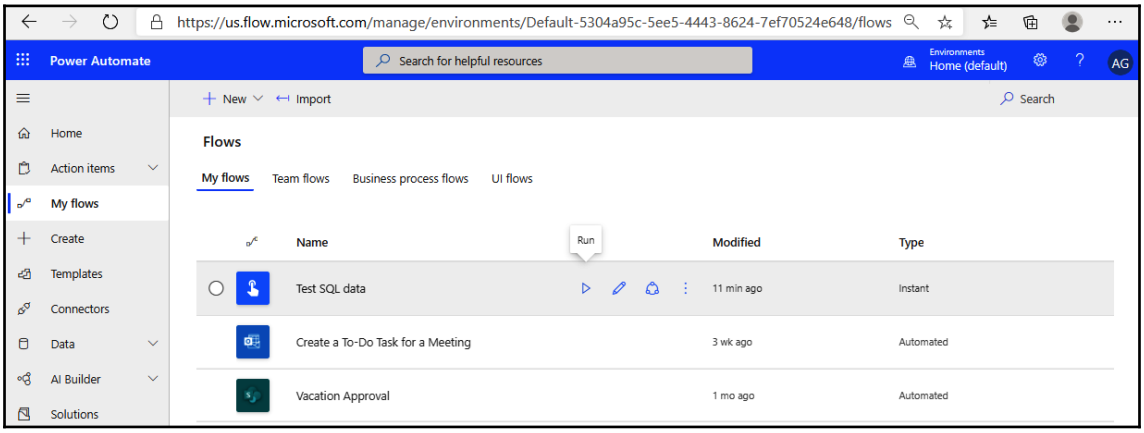


The flow is now ready to test.

Executing the flow

Use the process you learned about in `Chapter 5`, *Creating Button Flows*, to execute the flow. As a refresher, you can follow these steps:

1. Navigate to the Power Automate web portal (`https://flow.microsoft.com`) and select **My flows**.
2. Hover over the **Test SQL data** flow and click the **Run** button to execute the flow:



3. On the **Run flow** flyout panel, click **Continue**.
4. Click **Run flow**.
5. Click **Done**.

With that, the flow has been successfully executed.

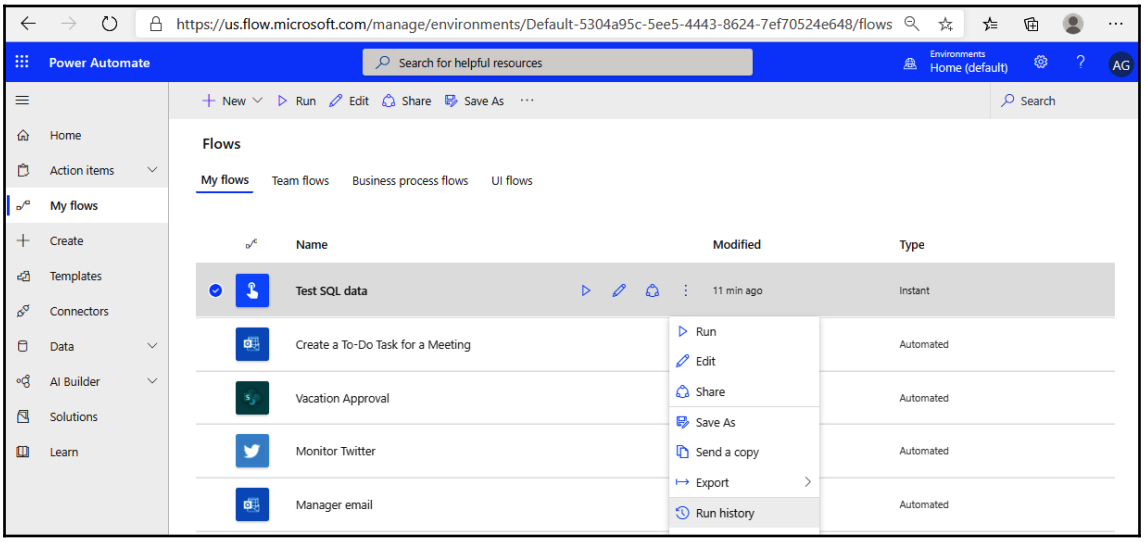
Verifying the flow

You can check whether the content was written to the flow by examining the flow's run history or by querying the database table in SQL.

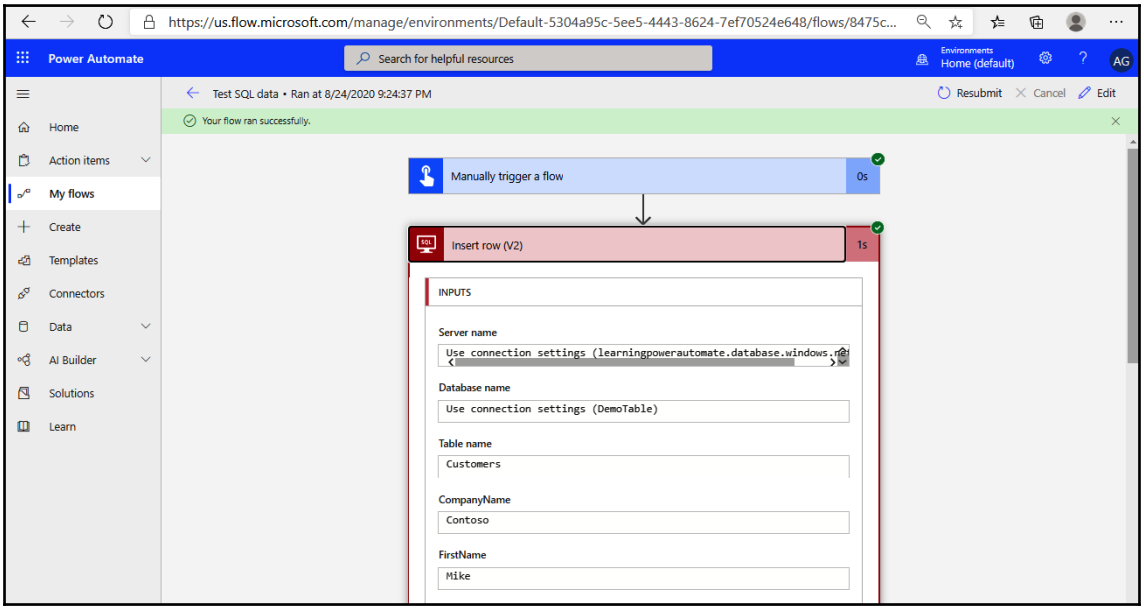
Reviewing the run history

The easiest and quickest way to verify whether the flow is successful is to examine its run history. The flow's run history will show the steps that were performed during the flow's execution. You can review the run history for the flow by using the following process:

1. Expand the ellipses for the flow and then select **Run history**:



2. Select the date and time of the run to display the history.
3. Expand the SQL action to view the data:



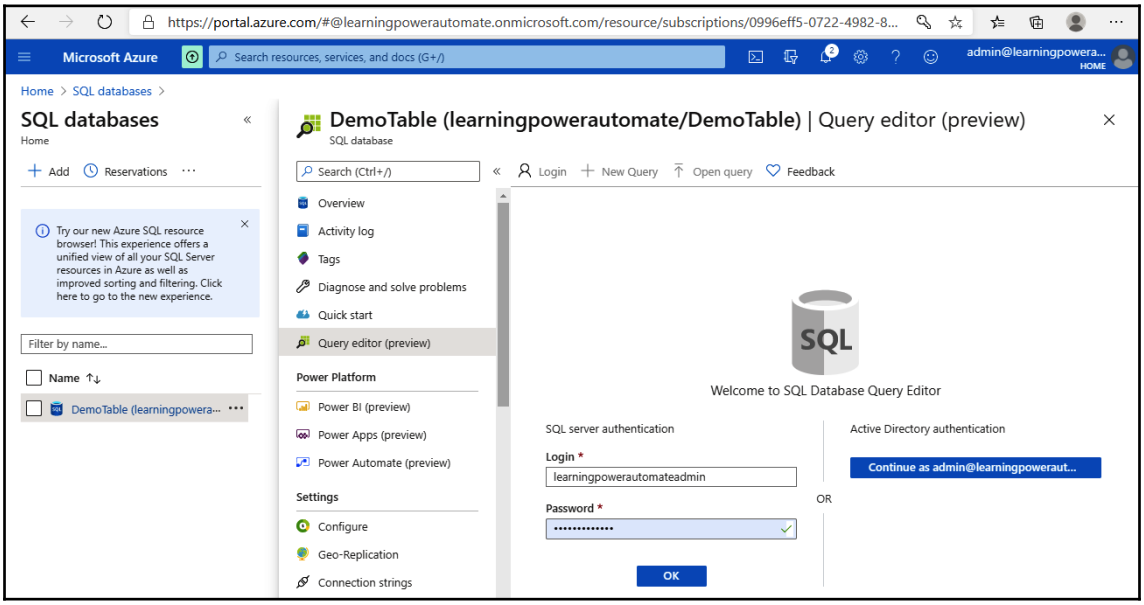
The expanded action will show the data that was inserted into the table.

Reviewing the SQL data

Using the SQL Query editor, you can also view the newly added data. To view the SQL data directly, follow these steps:

1. From the Azure portal (`https://portal.azure.com`), navigate to **SQL databases**.
2. Select **Query editor (preview)**. Enter your login credentials to access the database and then click **OK**:

If prompted, you may need to enable a firewall rule to allow your local computer to communicate with the Query editor.

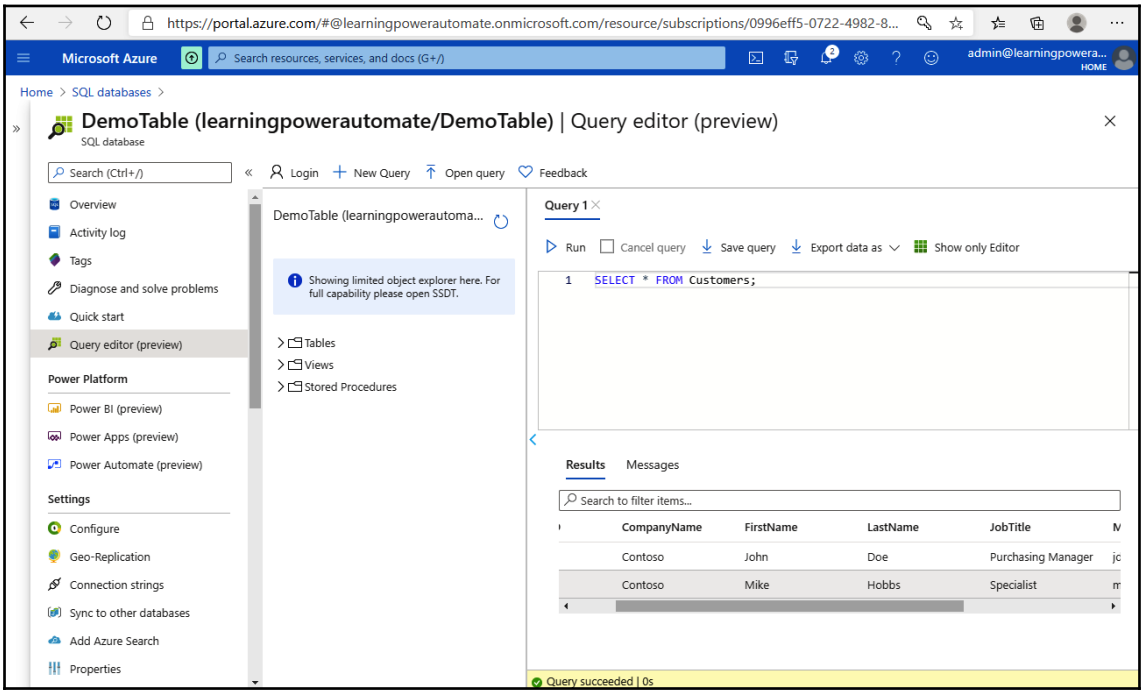


3. Copy and paste the following code into **Query editor**:

```
SELECT * FROM Customers;
```

4. Click **Run**.

5. Compare the data in the **Results** section to the values you entered in the *Creating the flow* section:



No matter how you choose to verify that the flow was executed successfully, you should be able to see the same data.

Summary

In this chapter, you learned about some of the database interaction capabilities of Power Automate. Following the steps in this chapter, you were able to successfully create an Azure SQL Server instance, a database, and a database table to store content. You were able to connect to the database, create a flow, and insert a new row of data. Finally, you examined the run history of the flow and learned how to query the database directly.

The next chapter will build on this knowledge by showing you how to take Microsoft Forms data and insert it into a database using Power Automate.

13
Working with Microsoft Forms

Gathering and processing feedback from users or customers is an important part of business operations. Microsoft Forms is a survey tool that can be used to capture that information, both with fixed questions and free-form text entry options.

In this chapter, we're going to use some of the concepts we've already learned about conditions (`Chapter 8`, *Working with Conditions*) and adding content to a database (`Chapter 12`, *Using a Database*) to create flows based on input from Microsoft Forms. Specifically, we'll look at the following topics:

- Understanding the Forms connector triggers and actions
- Creating a basic form
- Processing a form with Power Automate

When you finish this chapter, you'll have an understanding of how you can connect Forms and SQL with Power Automate.

Let's dig in!

Understanding the Forms connector triggers and actions

Before we can begin crafting a flow that involves Microsoft Forms, it's important to understand what kinds of triggers and actions are available for the Forms connector. As a reminder, triggers are activities that can initiate a flow, and actions are the activities that a flow can perform.

Triggers

As mentioned in the introduction, Microsoft Forms is a survey and information gathering tool. An end user's interaction with Forms ends when they complete and submit the form. As such, the Forms connector currently has only one trigger: **When a new response is submitted**. This trigger is activated when a user submits a form to the service.

Actions

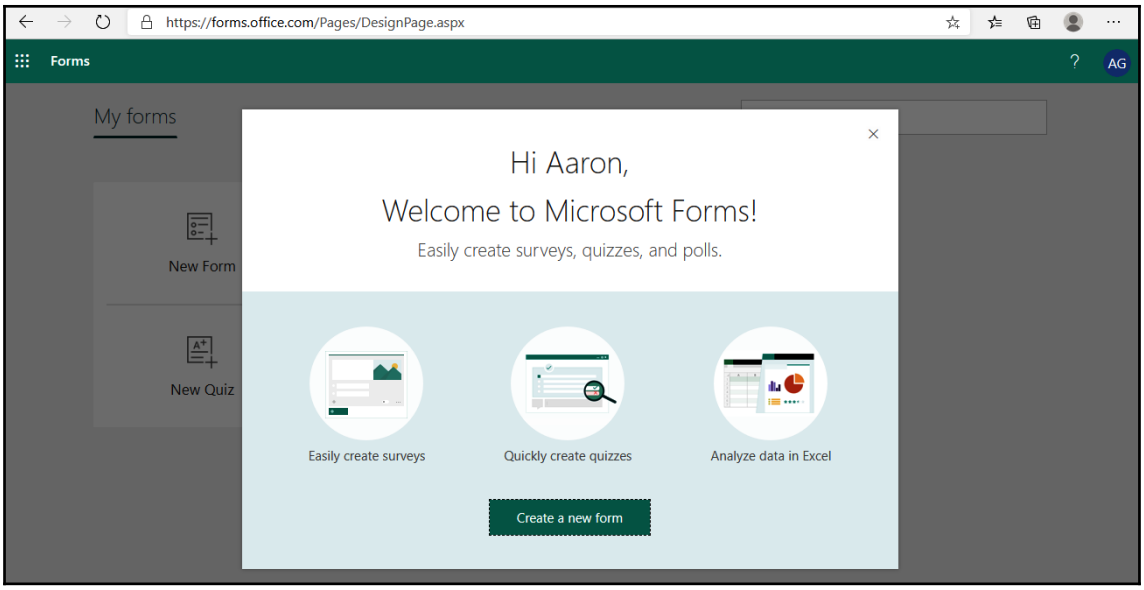
The Forms connector also has only a single action: **Get response details**. Using this action, Power Automate can retrieve the data submitted by the user and use the field values in a flow.

Before we can use Power Automate to process a form, we'll need a form to work with. In the next section, we'll create a form that will collect basic contact information.

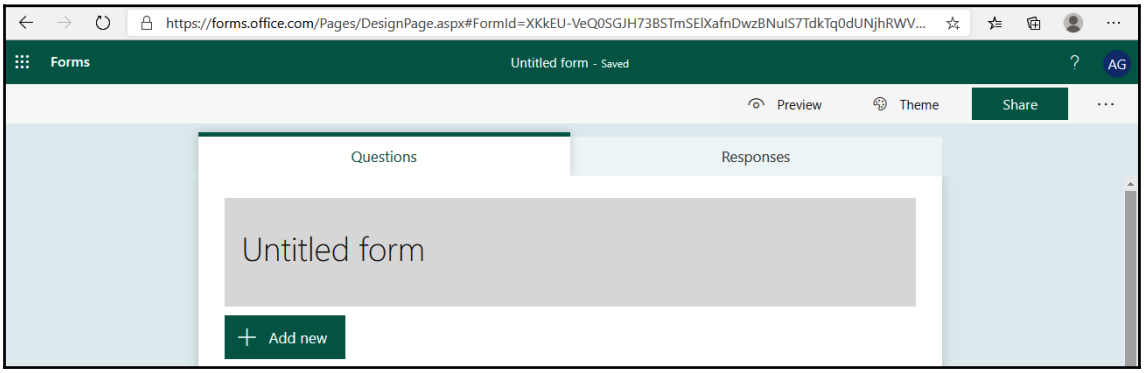
Creating a basic form

The Forms application is relatively straightforward to use. In this section, we'll create a basic form to collect user information. To create the form, follow these steps:

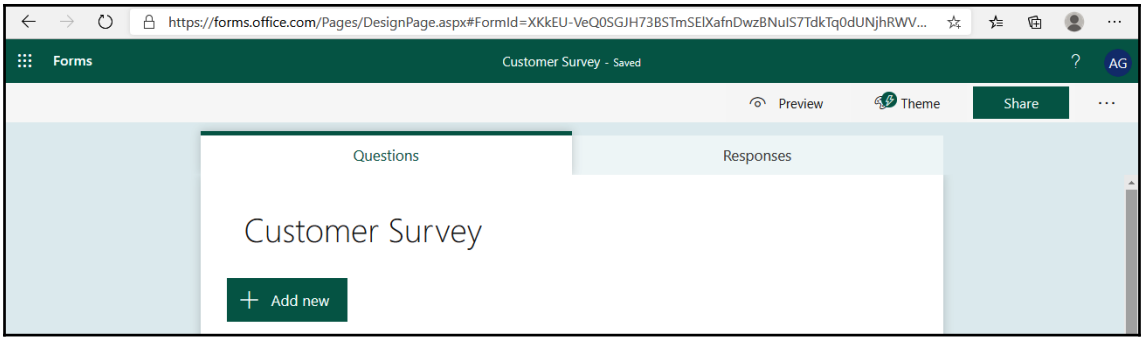
1. Launch the Forms application by navigating to `https://forms.office.com` and signing in.
2. If this is your first time signing into Forms, click the **Create a new form** button on the splash page. If you've logged into Forms previously, you can click the **New Form** button on the dashboard:



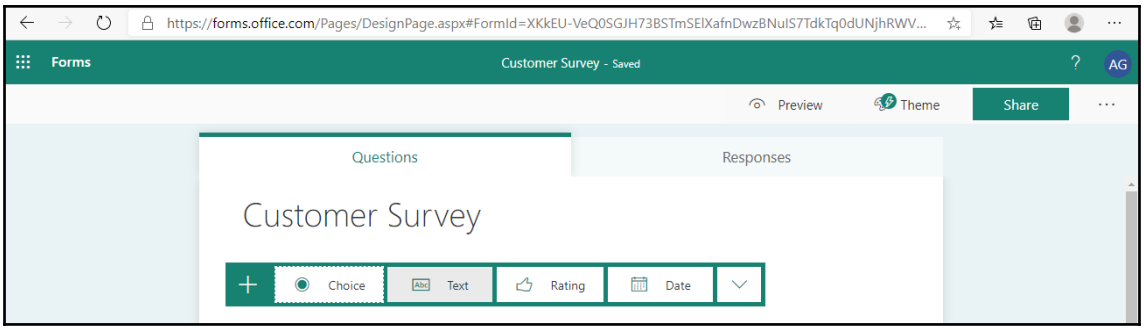
3. Click the title area (**Untitled form**) text box and enter a name for the form, such as **Customer Survey**:



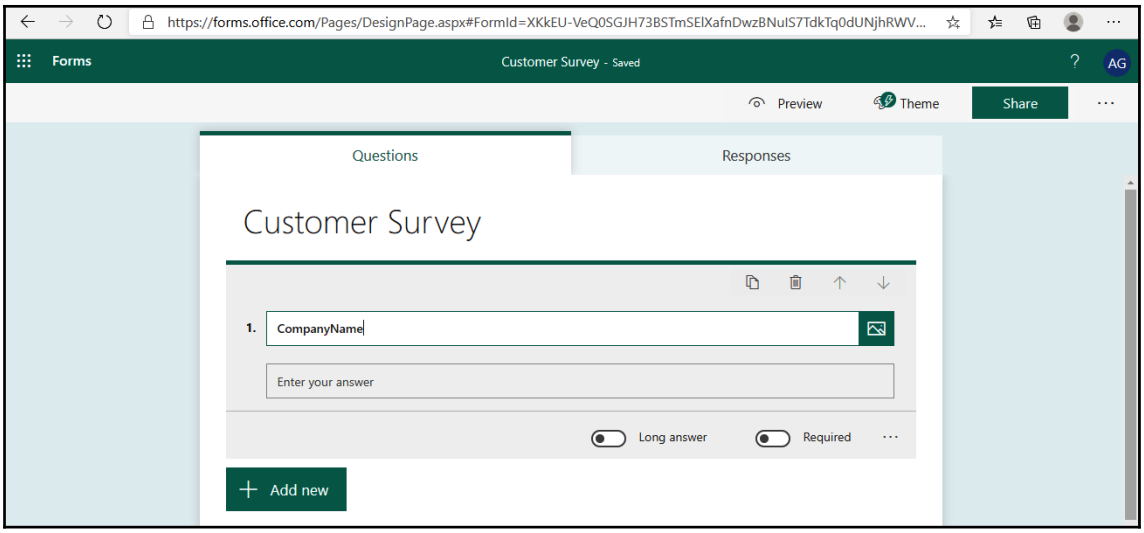
4. Click **+ Add new** to add a new item:



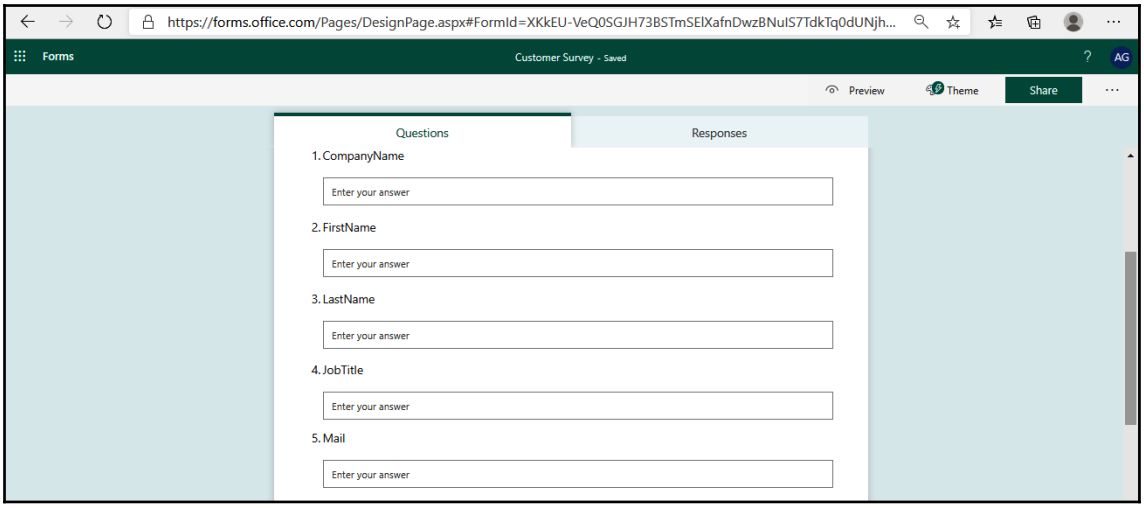
5. Select the **Text** option:



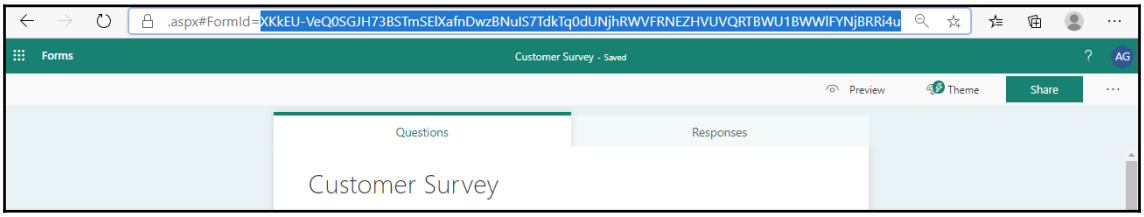
6. Enter the value `CompanyName`. You can enter something more descriptive if desired, but make sure it is easily identifiable as the company name value:



7. Repeat *steps 4-6*, adding fields for `FirstName`, `LastName`, `JobTitle`, `Mail`, and `TelephoneNumber`:



8. In your browser's URL bar, select the value for `FormID` (after `FormID=`) and copy it to Notepad or some other location. It will look like this: `XKkEU-VeQ0SGJH73BSTmSElXafnDwzBNuIS7TdkTq0dUNjhRWVFRNEZHVUVQRTBWU1BWWlFYNjBRRi4u`:



When you're finished, you should have a form with fields that resemble the column names for the database you created in `Chapter 12`, *Using a Database*.

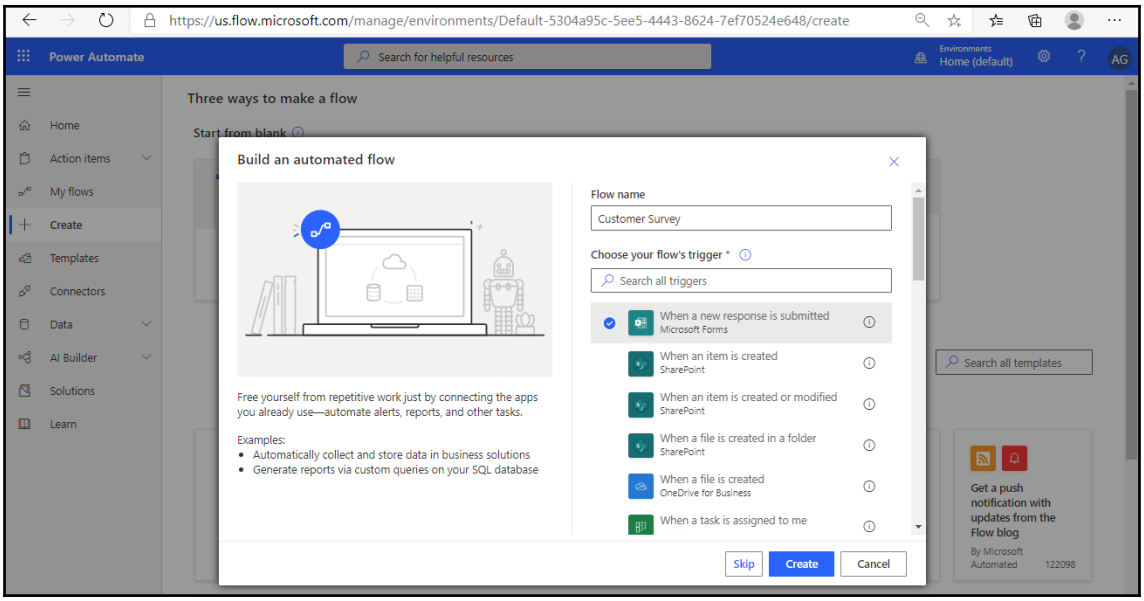
Next, we'll use Power Automate to save these form responses to a database for later analysis.

Processing a form with Power Automate

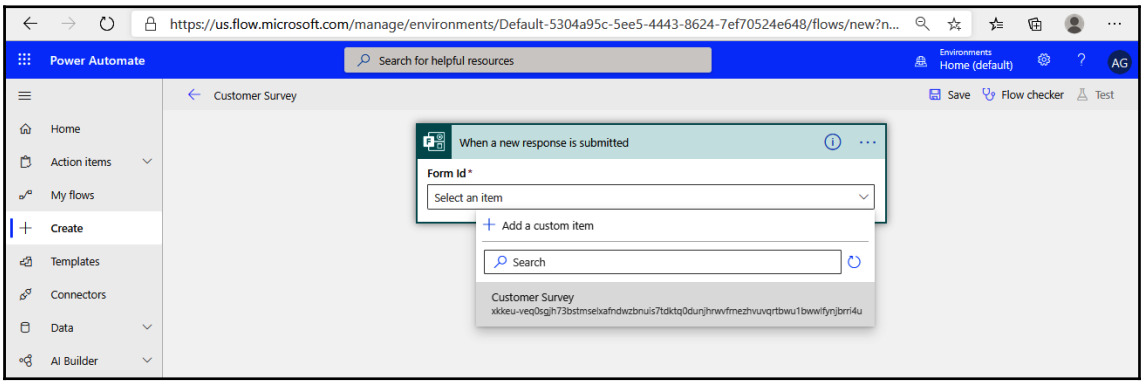
In this exercise, we'll configure a flow to process the responses and save them into the database you created in `Chapter 12`, *Using a Database*. The flow will utilize the `FormID` value you saved in the previous exercise, the **When a new response is submitted** trigger, and the **Get response** details action for the Microsoft Forms connector.

Follow these steps to configure the flow:

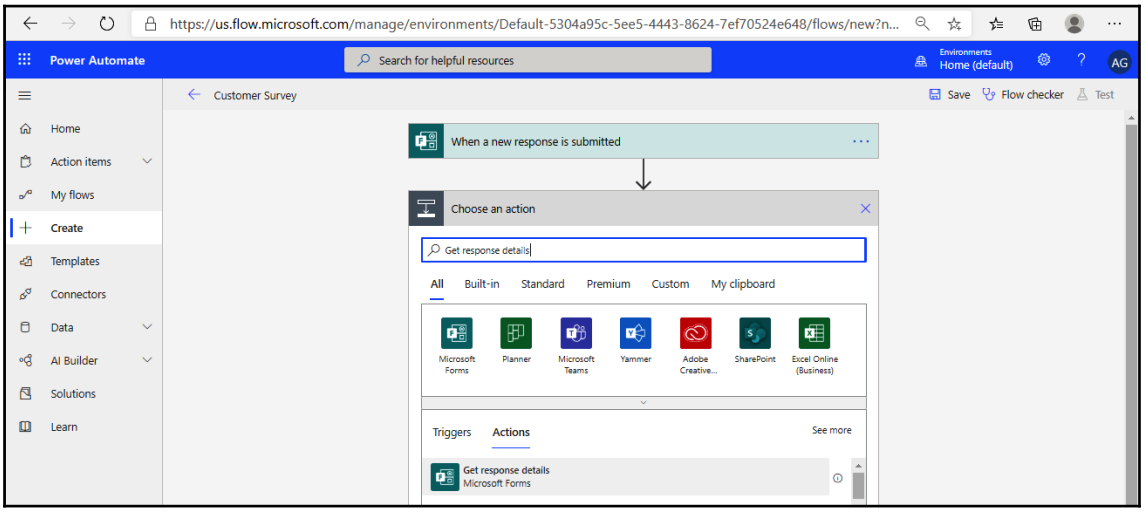
1. Log into the Power Automate web portal (`https://flow.microsoft.com`) and click **+ Create**.
2. Under the **Start from blank** section, select **Automated flow**.
3. Enter a name for the flow (such as `Customer Survey`) and select the **When a new response is submitted** trigger for Microsoft Forms. Click **Create**:



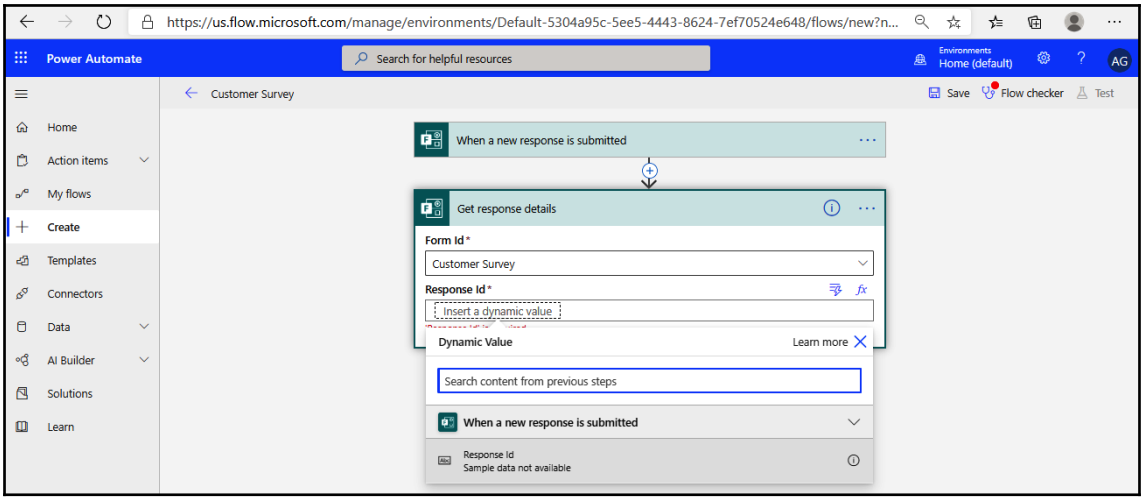
4. In the details for the trigger, select the Form if it is displayed. If the form you created is not displayed, select **+ Add a custom item** and paste in the `FormID` value you saved in the previous exercise:



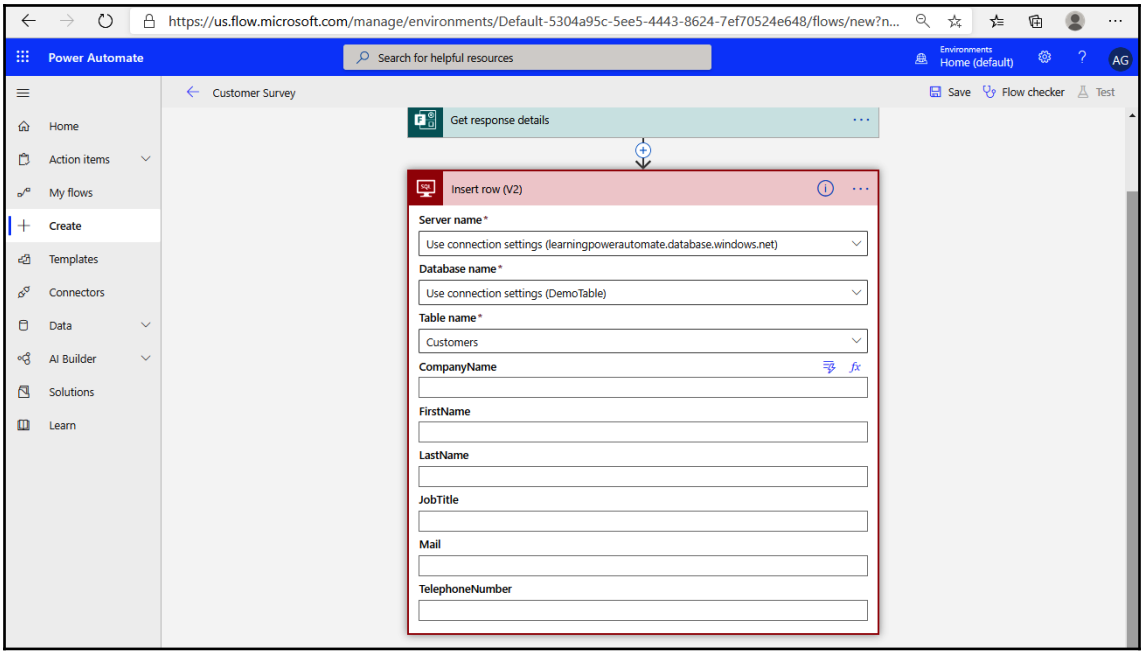
5. Click **+ New step**.
6. In the search bar, start typing `Get response details` and select the action for Microsoft Forms:



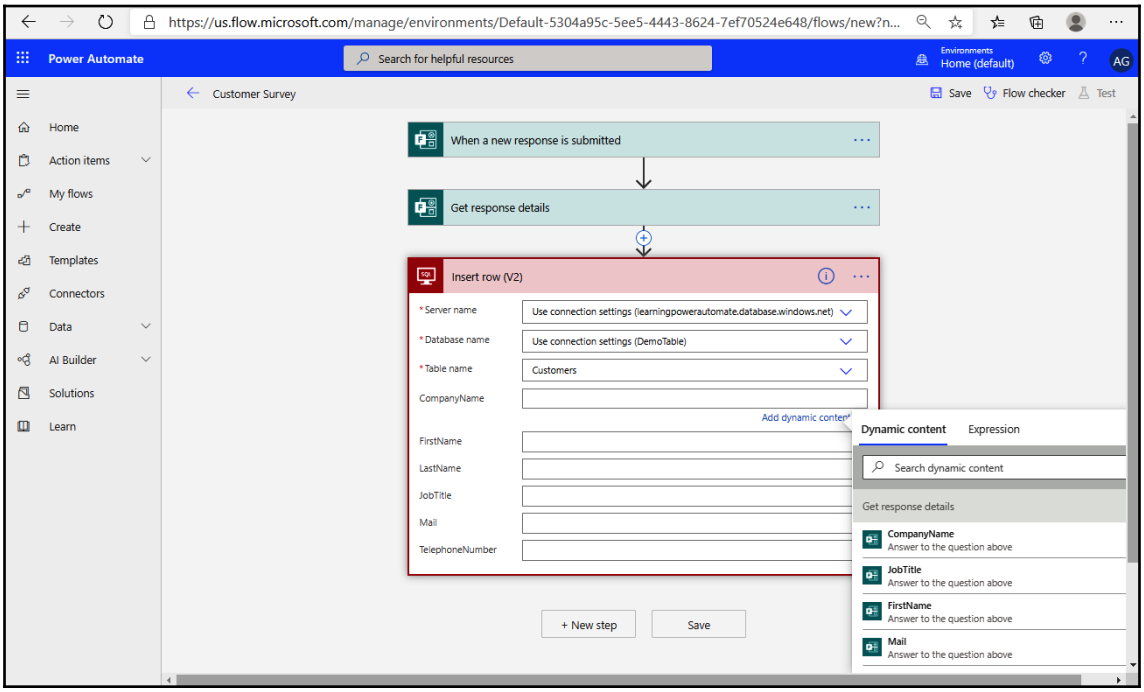
7. In the **Get response details** action, select the `FormID` value representing the form. Under **Response Id**, select **Dynamic content** and then select the **Response Id** dynamic content token:



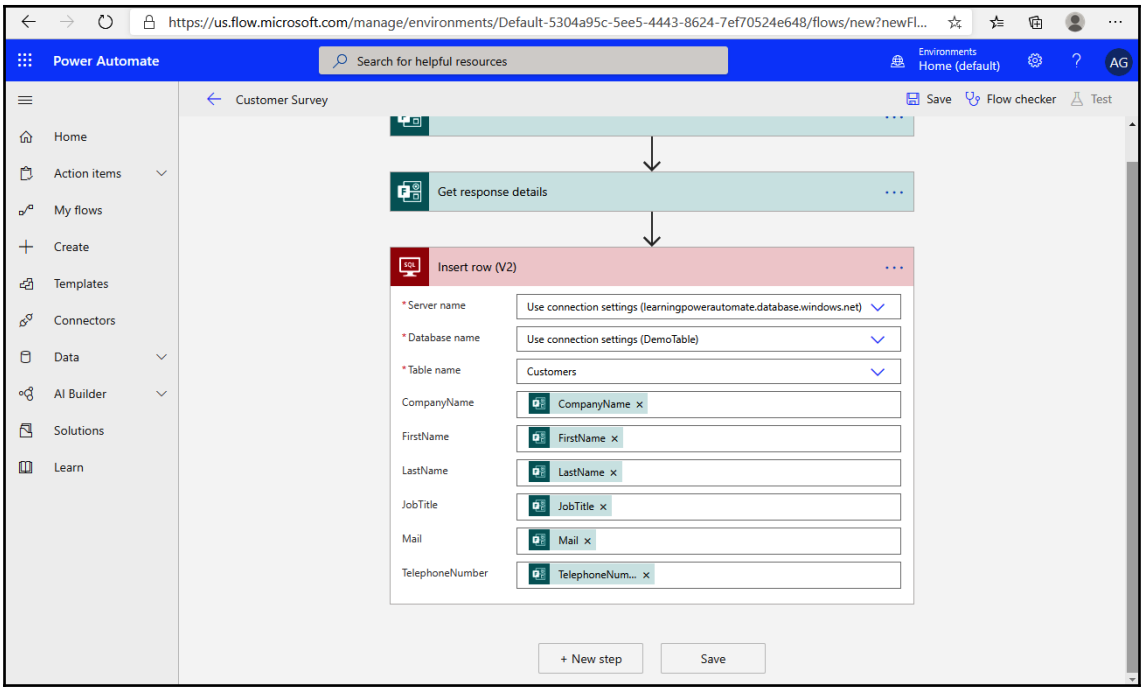
8. Click **+ New step**.
9. Select the **Insert row (V2)** action for **SQL Server**.
10. Select the **Server name**, **Database name**, and **Table name** values from the connection settings you created in `Chapter 12`, *Using a Database*. After a moment, the column values should be displayed, as shown in the following screenshot:



11. Select the **CompanyName** text box, select **Dynamic content**, and then select the **CompanyName** answer:



12. Repeat *step 11* for the remaining database column fields, matching them to the response values. Click **Save**:

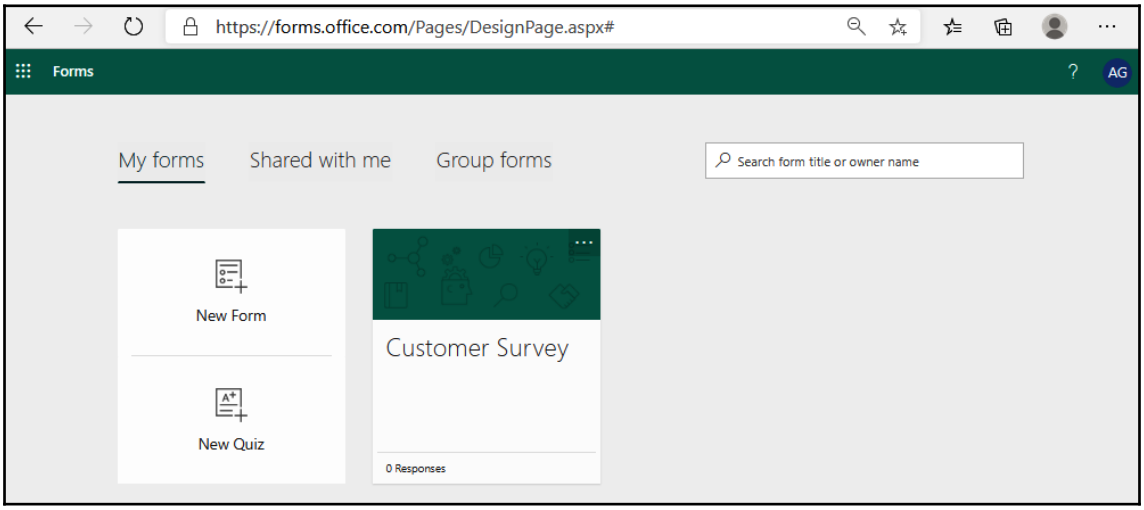


The flow has been created. This flow will be triggered on a new form response and save the form data directly to a database. Next, we'll submit a form and then look for the results in the database.

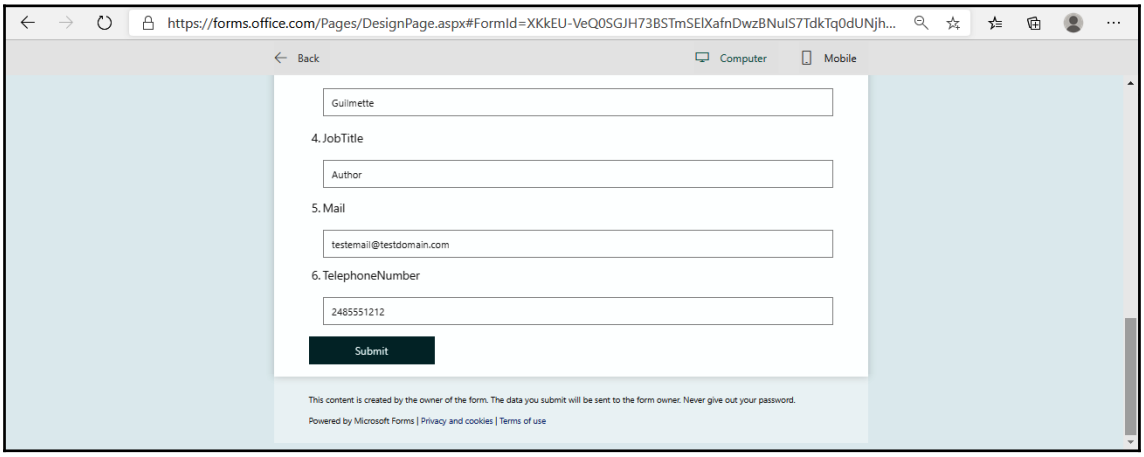
Testing the flow

In order to test the flow, you will need to submit a form response. To submit a form response, follow these steps:

1. Navigate to the Microsoft Forms dashboard (`https://forms.office.com`) and select your form under **My forms**:



2. Select **Preview**.
3. Fill out the form and click **Submit**:



After the form has been submitted, you will receive a confirmation page indicating that your submission was successful.

Next, we'll verify that the flow ran successfully.

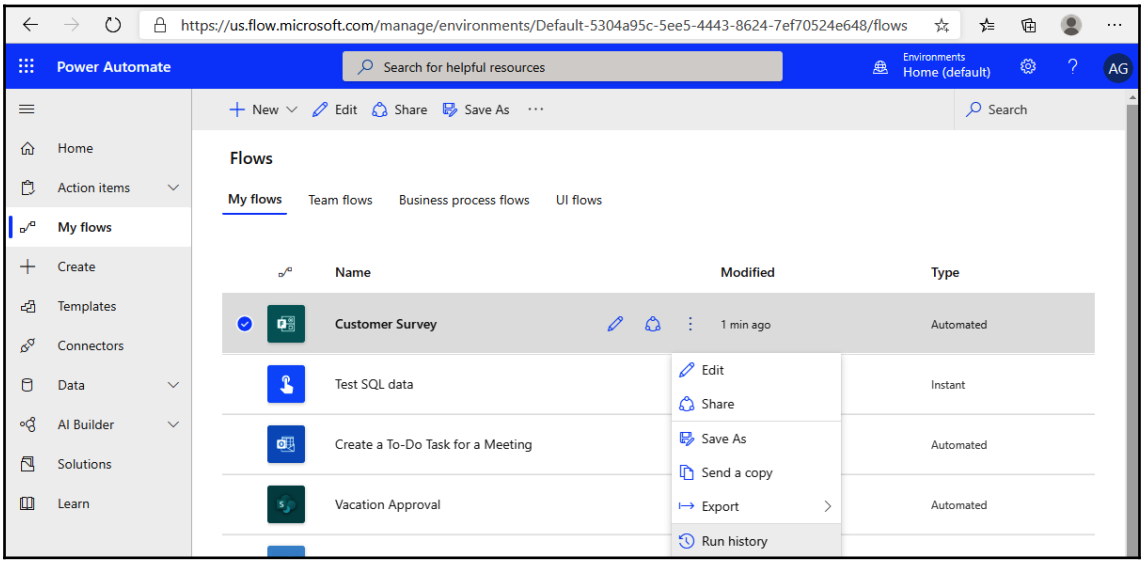
Verifying the result

Finally, you'll want to ensure that the flow has completed as you anticipated. You can check to see if the content was written to the flow by examining the flow's run history or by querying the database table in SQL.

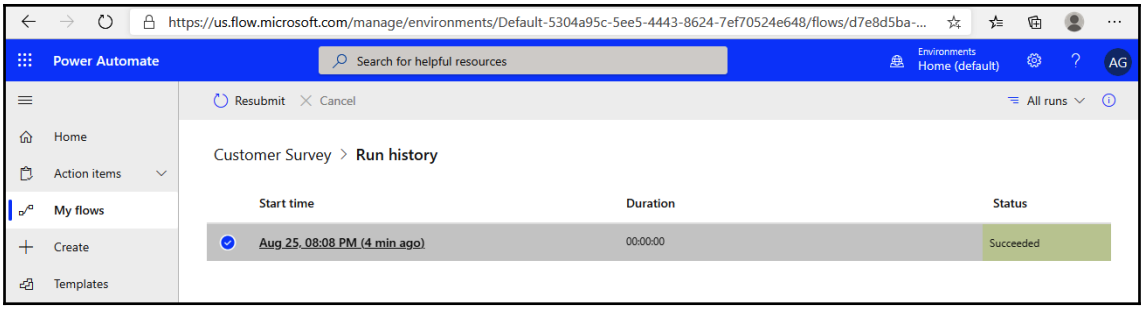
Reviewing the run history

The easiest and quickest way to verify if the flow is successful is to examine the run history. The run history will show the steps performed during the flow's execution. You can review the run history for the flow by using the following process:

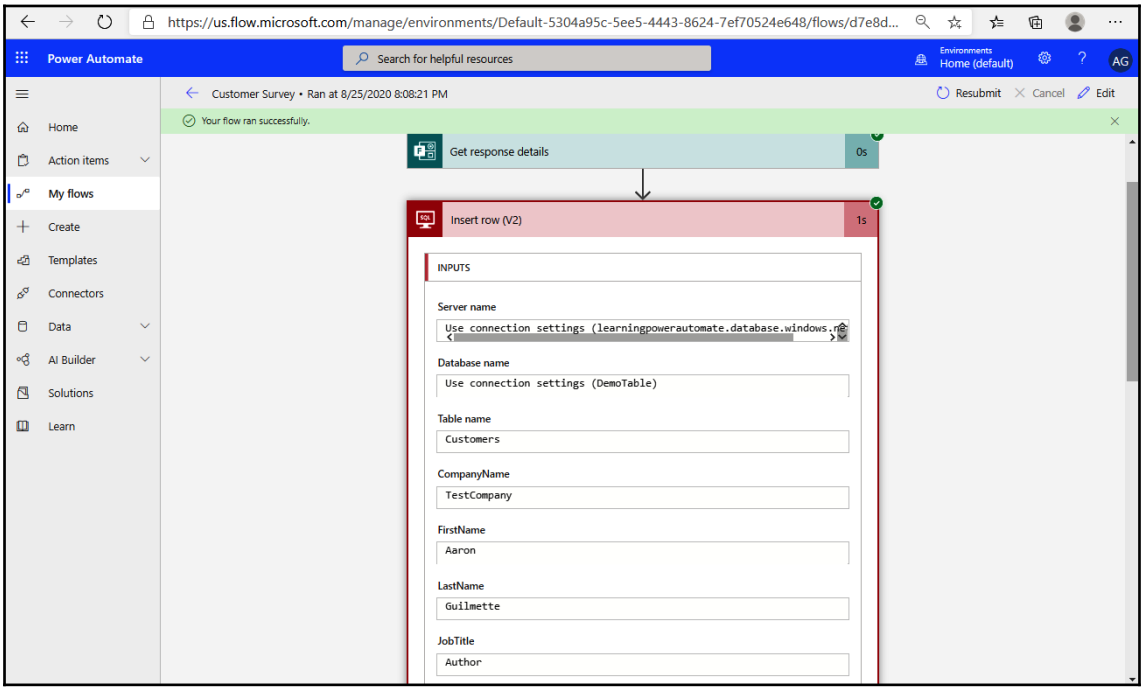
1. Expand the ellipses for the flow and then select **Run history**:



2. Click on **Run**:



3. Expand the **Insert row (V2)** action and review the data:



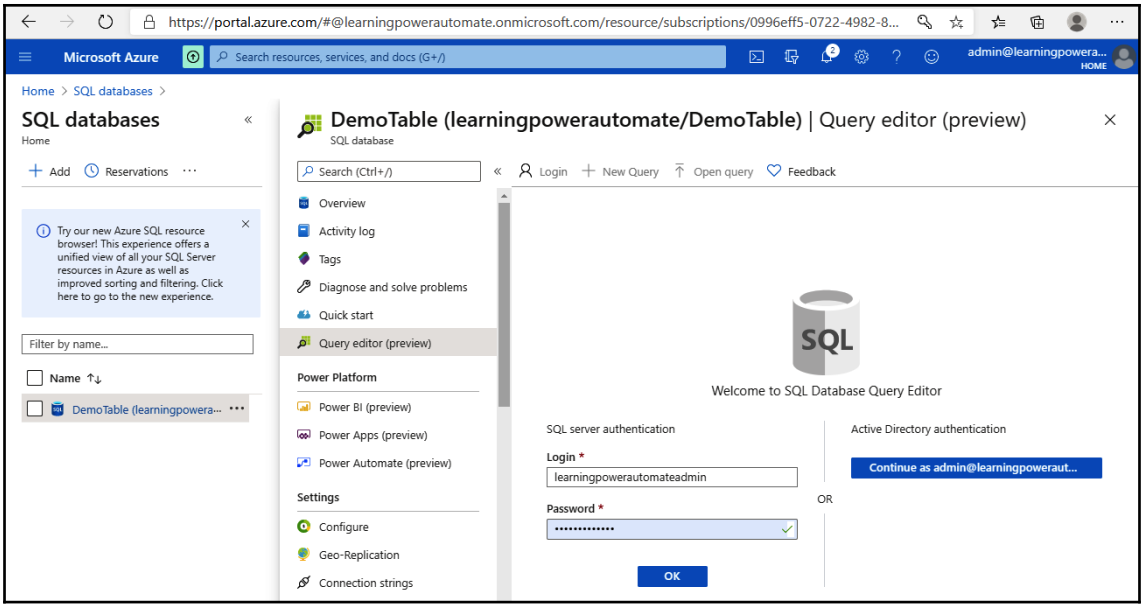
The expanded action will show the data that was processed from the form and inserted into the table.

Reviewing the SQL data

Using the SQL Query editor, you can also view the newly added data. To view the SQL data directly, follow these steps:

1. From the Azure portal (`https://portal.azure.com`), navigate to **SQL databases**.
2. Select **Query editor (preview)**. Enter credentials to access the database and then click **OK**:

If prompted, you may need to enable a firewall rule to allow your local computer to communicate with the Query editor.

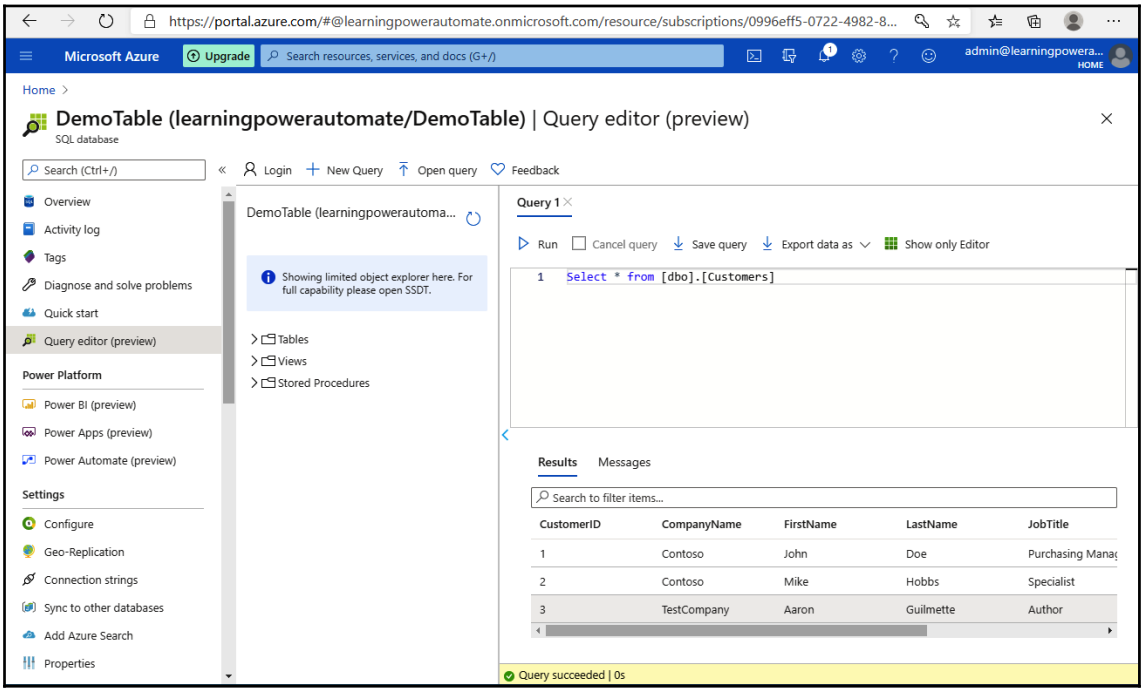


3. Copy and paste the following code into the Query editor:

```
SELECT * FROM [dbo].[Customers]
```

4. Click **Run**.

5. Compare the data in the **Results** section to the values you entered in the *Testing the flow* section:



No matter which way you choose to verify the results of the flow's execution, you should be able to see the same data. Verify that the data in the table matches the data submitted in the form.

Summary

In this chapter, you learned how to create a basic form using Microsoft Forms. Additionally, you were able to build upon the skills acquired in `Chapter 12`, *Using a Database*, to save data from Microsoft Forms into a SQL database. In real-world scenarios, you could expand this flow further to generate a confirmation email back to the submitter or use a tool such as Excel or Power BI to sort the data. Depending on the type of data gathered in the form, you could even send this data to Azure Cognitive Services for sentiment analysis.

In the next chapter, we'll look at using Power Automate to gather user input directly.

14
Accepting User Input

Many types of business processing require some sort of user input. Input can come in the form of data typed in by a user, a choice prompt where the user selects "Yes" or "No," or even uploading a file. Power Automate provides an interface for users to supply information as part of the flow. Connected applications, such as Power Apps, can also be used to supply information to a flow.

In this chapter, you'll be introduced to gathering different types of user inputs that can be used later in a flow. Specifically, this chapter will cover the following topics:

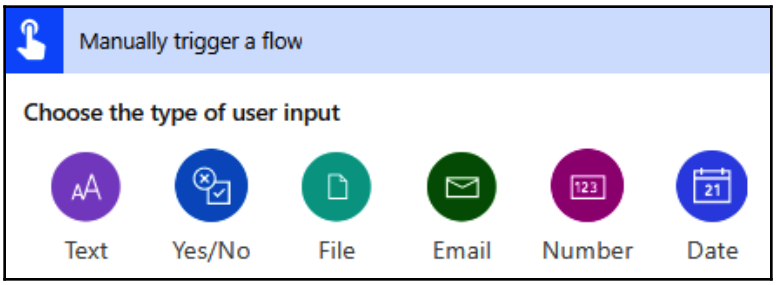
- Understanding the user input options
- Creating a flow that uses all input types

By completing the example exercise at the end of this chapter, you'll become familiar with the different types of user input options and how they can be used to create interactive flows.

Onward!

Understanding the user input options

User input can take many forms. When creating a manually triggered flow (also known as a *button* or *instant* flow), Power Automate allows flow designers to accept six types of user input. To add input capability to a manually triggered flow, click the title bar labeled **Manually trigger a flow**. The trigger will expand and present the input options, as shown in the following screenshot:



The types of input are as follows:

- **Text**: The **Text** input option supports several configurations, including a free-form text field, a single drop-down selectable list, and a multi-select list.
- **Yes/No**: The **Yes/No** input option can be used to allow users to utilize a Yes/No toggle switch.
- **File**: The **File** input option provides the ability for the user to upload a file from their device. If the user is accessing the flow from a mobile device, they can also access the camera to capture and upload a picture.
- **Email**: The **Email** input option is designed to accept content formatted as an email address or allow the user to select from the organization's existing Exchange Online address book.
- **Number**: The **Number** input option allows users to enter numerals.
- **Date**: The **Date** option can be used to allow users to input a date, providing consistently and correctly formatted date values.

A manually triggered flow can use one or more input options and can accept multiple instances of the same input object type. For example, you may need to create a flow that has a series of Yes/No questions as well as providing an option to upload a file at the end.

Additionally, all of the user input options can be configured as "required" or "optional," providing a high degree of flexibility without the need for complex "it/then" coding statements.

Now that you have a basic understanding of the types of input types, let's create a flow!

Creating a flow that uses all input types

In this example, we're going to configure a simple flow that uses all six input types, and populate a SharePoint list. You could also try this example to save content to a database.

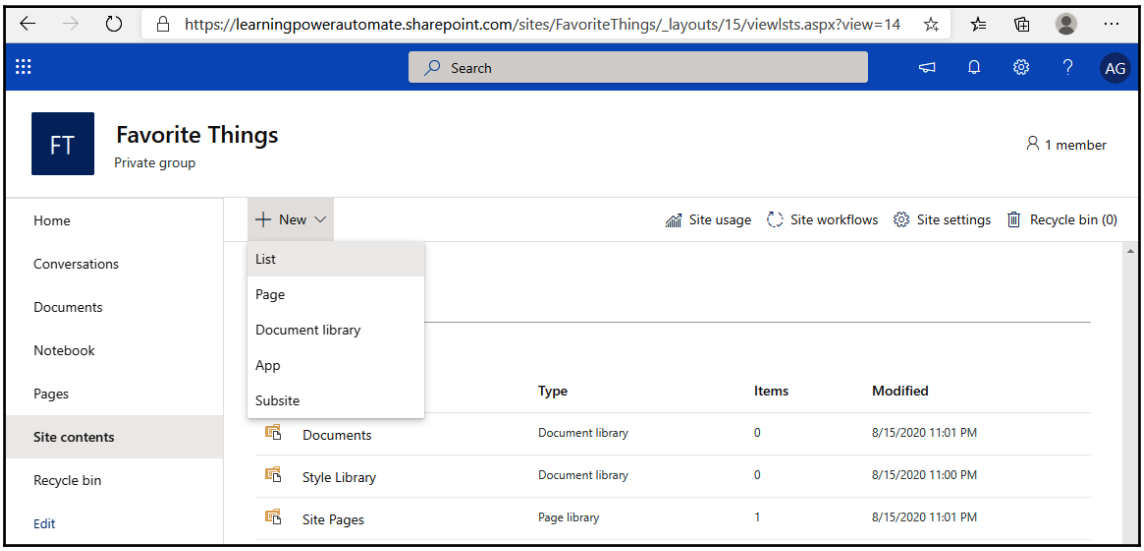
Before you begin configuring the flow, you'll need to configure some basic prerequisites, such as a SharePoint list that contains the columns to store the various data values. Once the prerequisites have been completed, you'll be able to move on to creating and testing the flow.

Configuring the prerequisites

Since the data for this flow will be stored in a SharePoint list, you'll need to configure a SharePoint list if you don't already have one that you can use. In this example, we'll create a new list and then add one column to store each of our values.

To create the list, follow these steps:

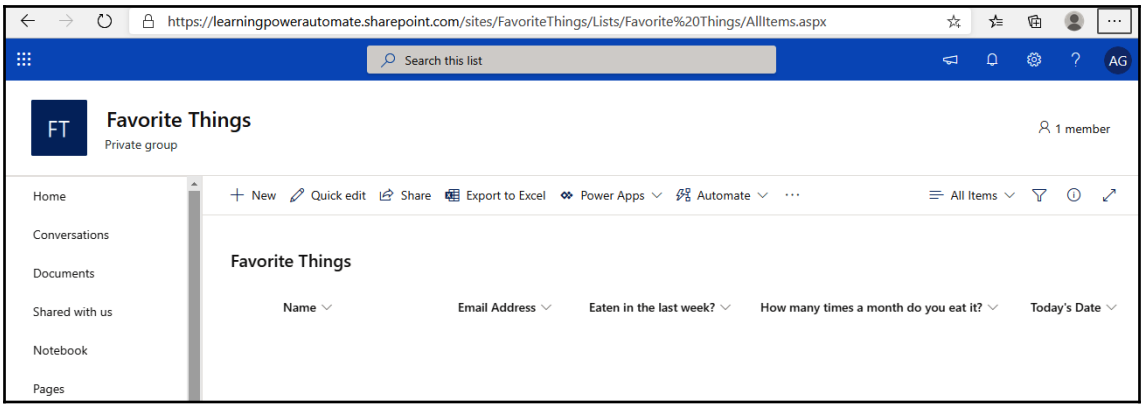
1. Navigate to a SharePoint site, click **Site contents**, and then click **+ New**. Select **List** from the drop-down menu:



2. Enter a name for the list and click **Create**.
3. Click **Quick Edit**, and then select **Title** and click **Rename Column**. Rename the column to **Name**.
4. Click **+ Add column** and create columns for each of the items/content types:

Column Name	**Column Type**
Email Address	Text
Eaten in the last week?	Yes/No
How many times a month do you eat it?	Number
Today's Date	Date

When you are finished, it should look similar to this:



5. Click **Exit quick edit** to save your changes.

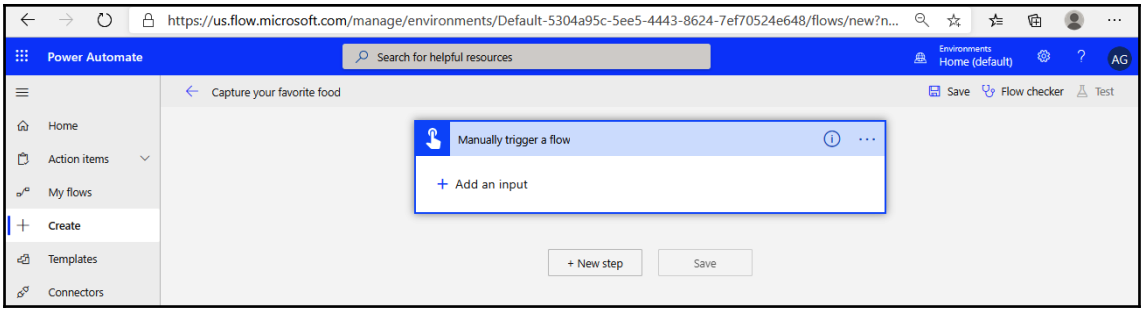
After a moment, your configured list should be saved. You're ready to move on to the next step.

Creating the flow

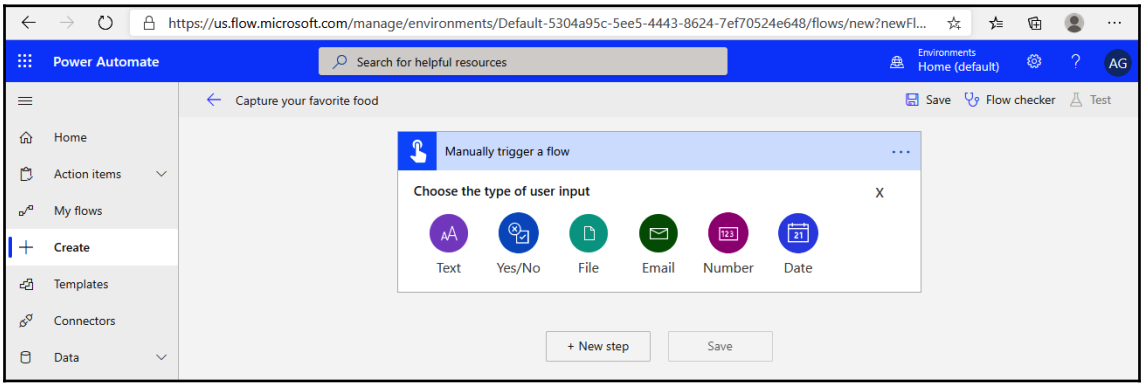
Now that you have a place to save the data, you can begin creating the flow. Follow these steps to create a manually triggered flow that captures user input:

1. Navigate to the Power Automate web portal (`https://flow.microsoft.com`) and click **+ Create**.
2. Select **Instant flow**.
3. Enter a name for the flow, select the **Manually trigger a flow** trigger, and click **Create**.

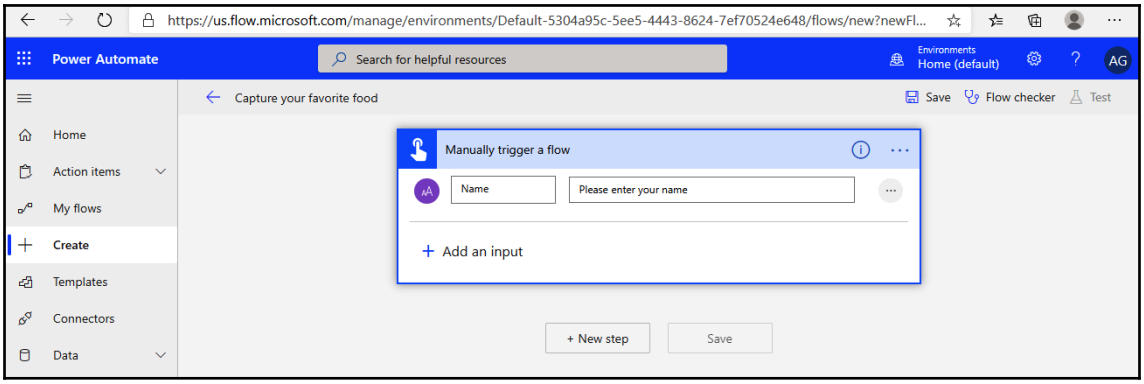
4. Click the title bar for the trigger to display the **+ Add an input** option:



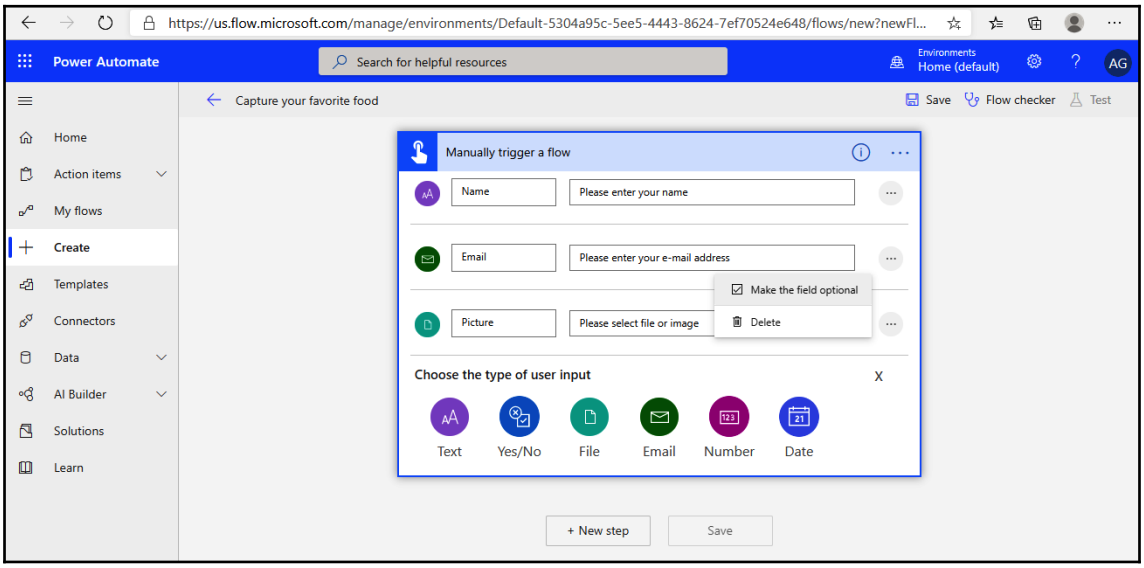
5. Click **+ Add an input** and select **Text**:



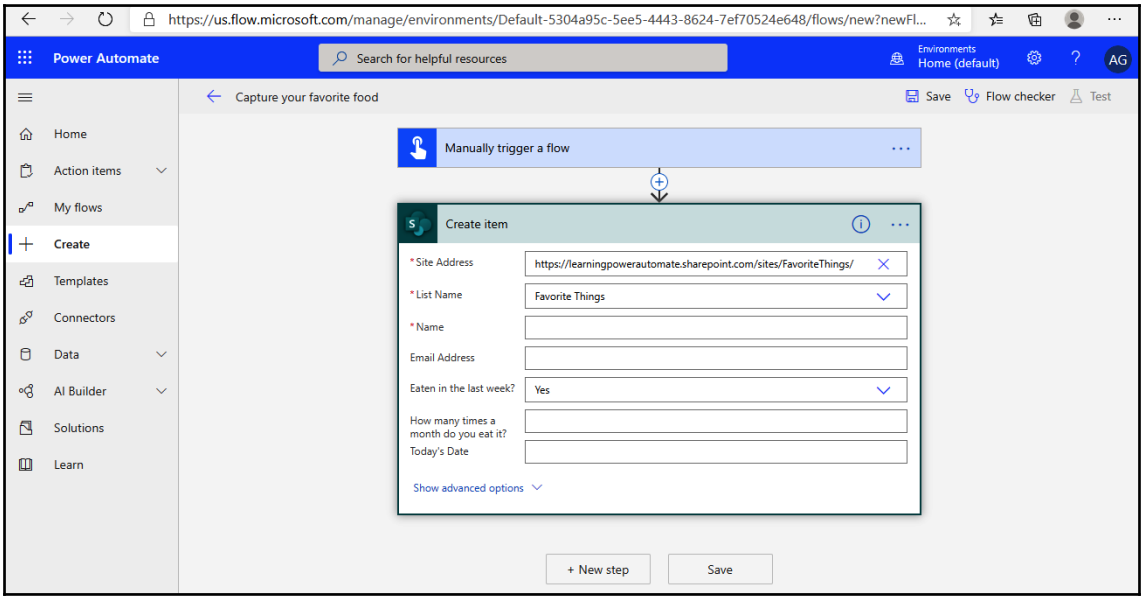
6. Update the label with the text `Name`, and then update the prompt to say `Please enter your name`:



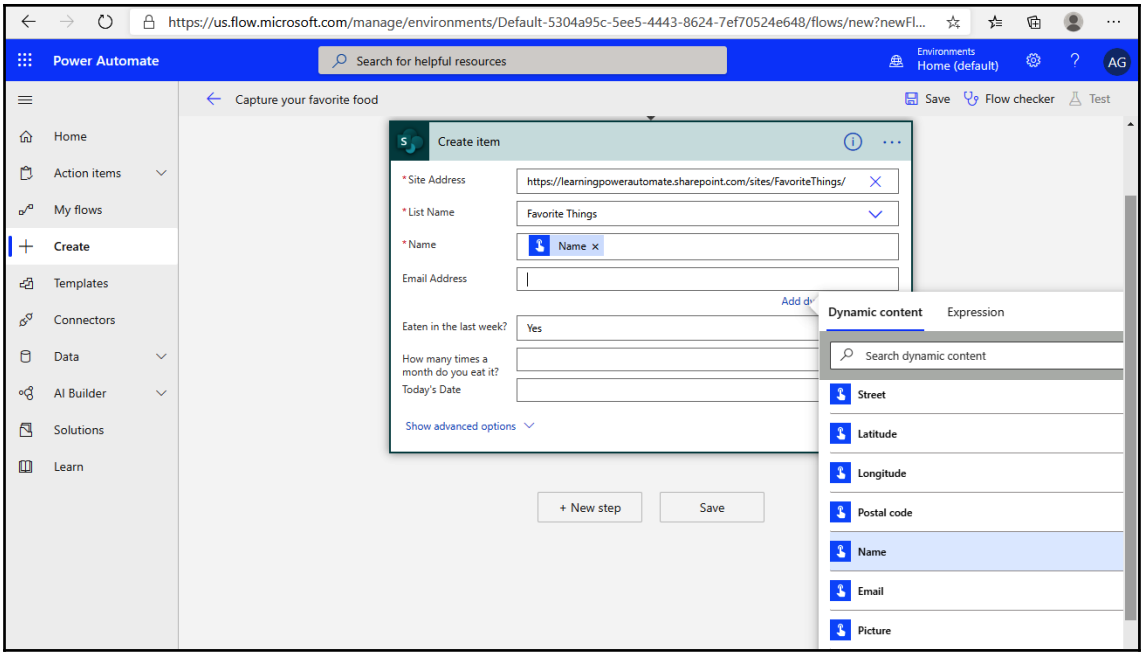
7. Click **+ Add an input** and select **Email**. Update the prompt to say `Please enter your e-mail address`.
8. Click **+ Add an input** and select **File**. Update the label to say `Picture`. Click the ellipsis (**...**) for the file item, and then toggle the option to **Make the field optional** or **Make the field required**, as desired:



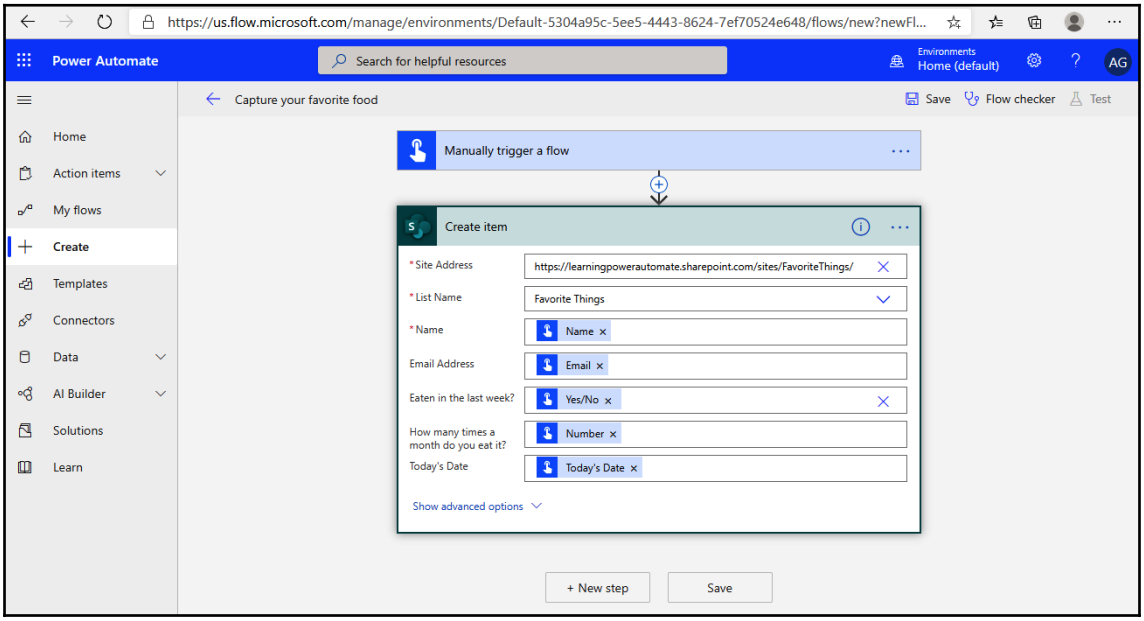
9. Click **+ Add an input** and select **Yes/No**. Update the label to say `Have you eaten it in the last week?`
10. Click **+ Add an input** and select **Number**. Update the prompt to say `Please enter the number of times per month you normally eat it.`
11. Click **+ Add an input** and select **Date**. Update the label to say `Today's Date`. Update the prompt to say `Please select today's date in YYYY-MM-DD format.`
12. Click **+ New step**.
13. Select the SharePoint **Create item** action.
14. In the **Site Address** box, select the site containing the list you created for this exercise. If it is not displayed, click **Enter custom value** and add the URL for the SharePoint site.
15. Select the list. After a moment, the column names on the list should appear:



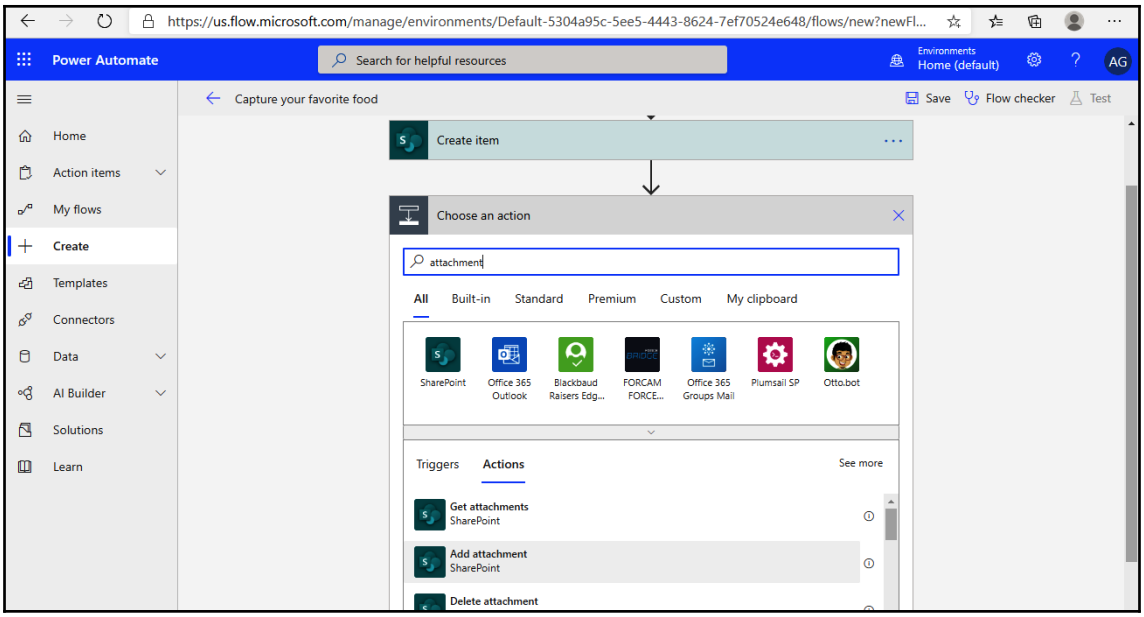
16. Click the **Name** field, and then select the dynamic value token for **Name**:



17. Repeat the process for each of the remaining fields displayed in the **Create item** action card. For the **Eaten in the last week?** Yes/No field, select **Custom value**, and then you'll be able to select the appropriate dynamic content token. When finished, it should look similar to the following screenshot:

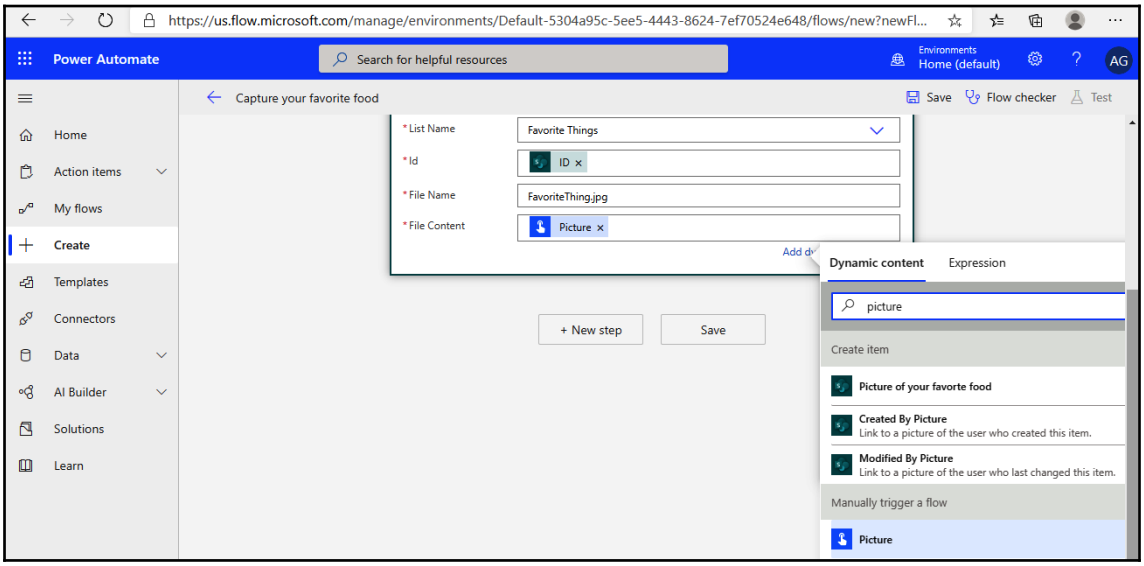


18. Click **+ New step**.
19. In the search box, type `attachment`. Select the SharePoint **Add attachment** action:



20. In the **Site Address** box, select the site containing the list. Select the list you used previously in the **List Name** box.
21. In the **Id** box, select the dynamic content token for **ID** under the **Create item** section.
22. In the **File Name** box, enter a file name to save the file as, such as `FavoriteThing.jpg`.

23. In the **File Content** box, select the **Picture** dynamic content token from the **Manually trigger a flow** section:



24. Click **+ New step**. Select the **Send an email (V2)** Office 365 Outlook action.
25. In the **To** box, select the **Email** dynamic content token.
26. In the **Subject** box, enter a subject such as `Thank you for your favorite food submission!`
27. In the **Body** box, enter any text and select any dynamic content tokens you wish to include.
28. Click **Save**.

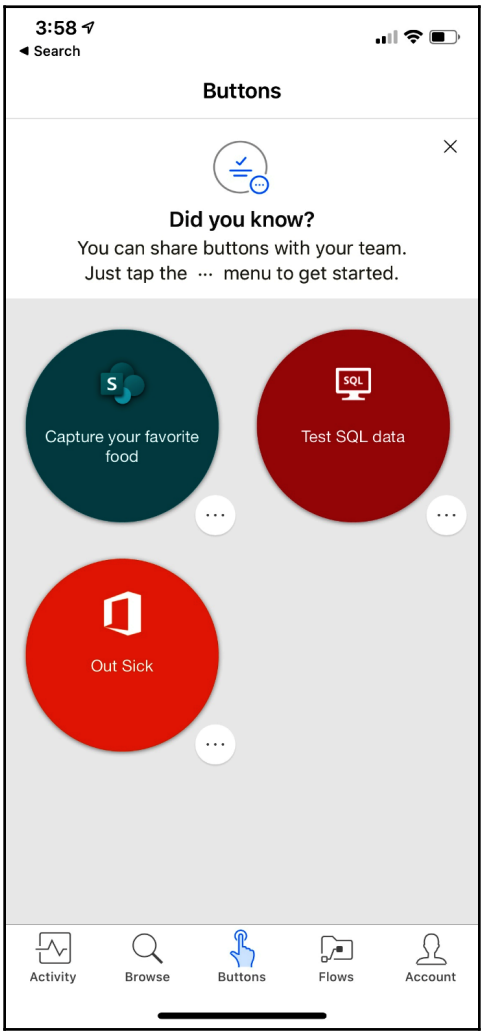
At this point, you have configured a flow that requests a number of input variables from a user and saves them to a SharePoint list. After creating the flow, you can click **Flow checker** to examine your flow for any errors.

Next, we'll test out the flow on a mobile device with the Power Automate app installed.

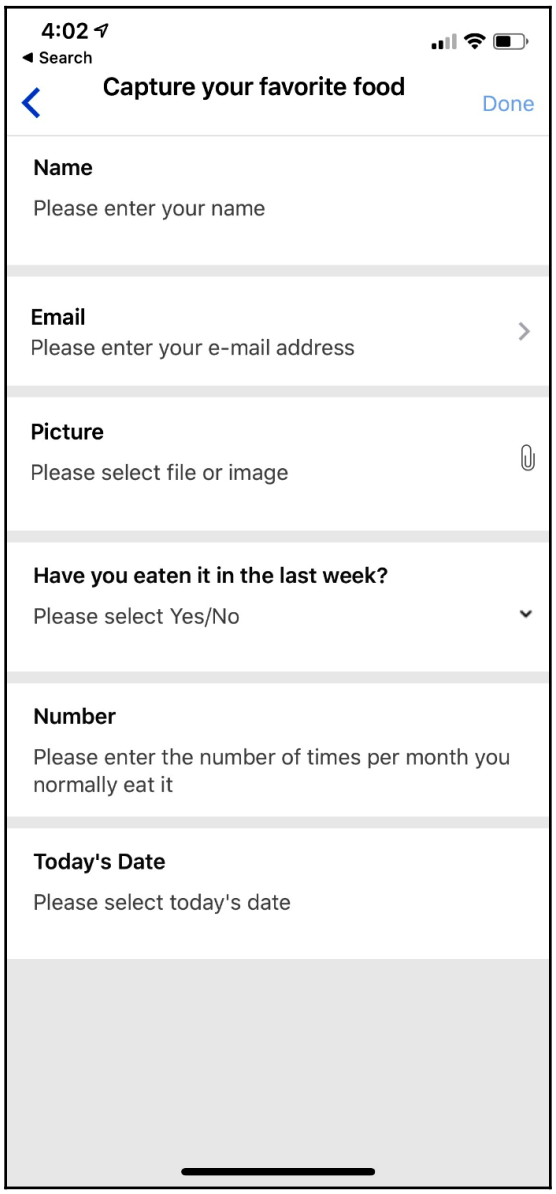
Executing the flow

To test this flow, we'll use a mobile device with the Power Automate app installed. To walk through the process, follow these steps:

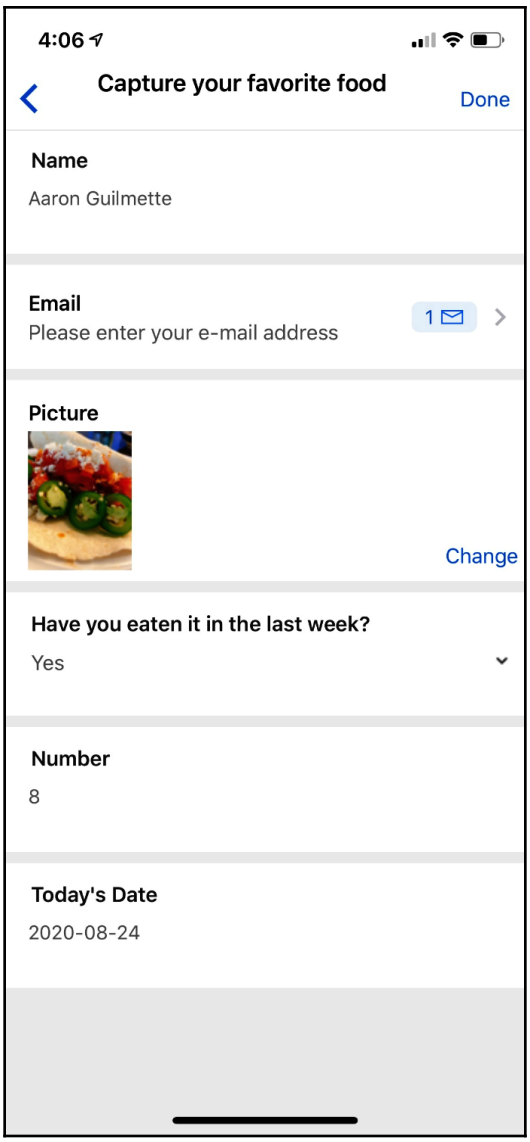
1. On a mobile device, launch the **Power Automate** app.
2. Select **Buttons**.
3. Select the **Capture your favorite food** button flow:



4. Fill out the user input fields:



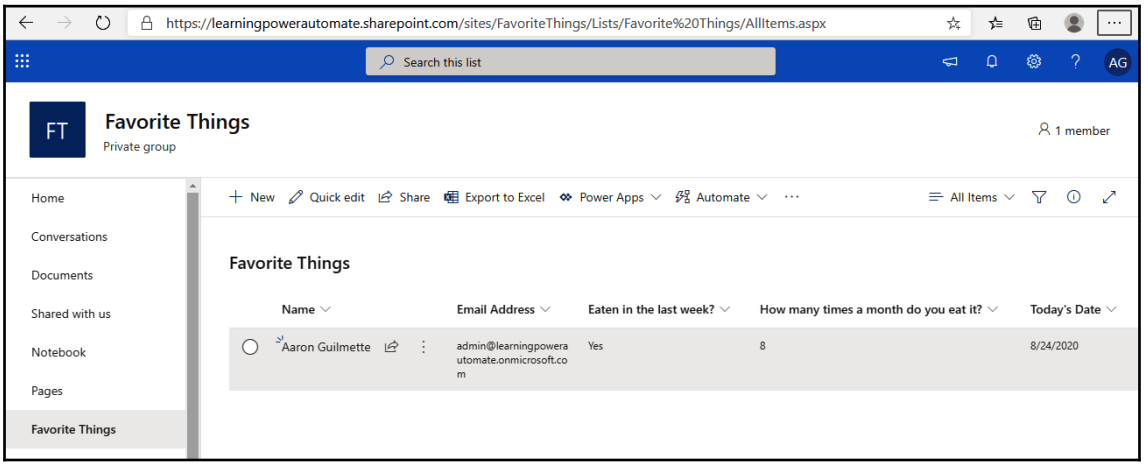
5. When finished, tap **Done**:



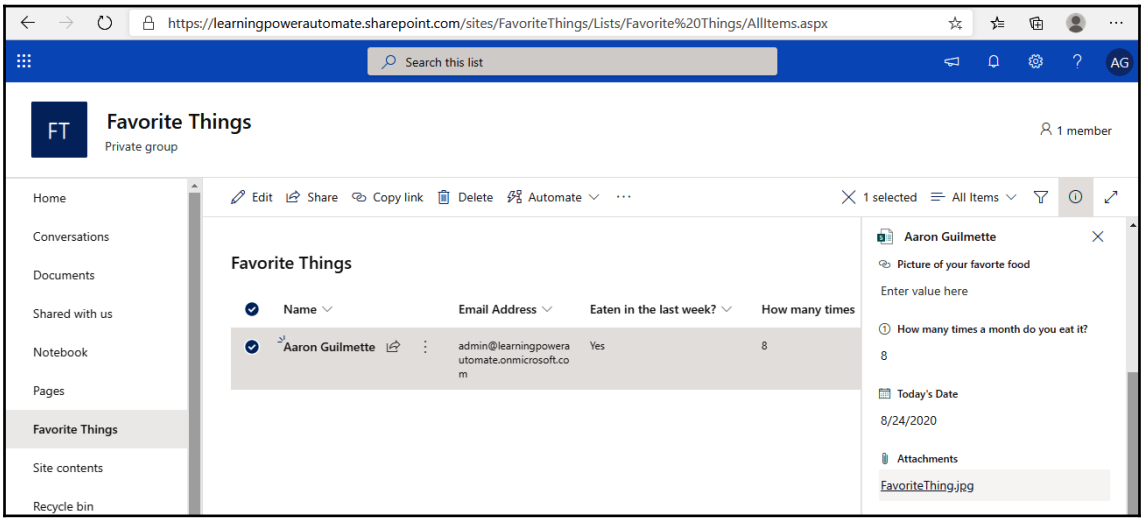
The app will return you to the **Button flow** page. At this point, it's time to verify that the flow completed successfully.

Verifying the flow execution

Once the flow has been submitted, you can review the run history for the flow (**Flow** | **My flows** | **...** | **Run history**), or you can browse to the SharePoint list where the content is saved:



You can click on the information panel icon to expose the details of the item and even view the uploaded picture under **Attachments**:



In this series of exercises, you created a SharePoint list to act as a data repository, built a flow to capture a sample of all six user input types, tested it, and verified that the data was indeed saved.

Summary

In this chapter, you learned about the six different kinds of user input that Power Automate can accept, including text, numbers, Yes/No choice boxes, and binary data. You also configured a flow that utilizes each of these methods and saved the content to a SharePoint list.

Requesting user input is a very useful feature of Power Automate, enabling a number of self-service and data gathering operations for your organization. With these types of input capabilities, you can enable workflow scenarios that rely on unique data provided at runtime, such as photos of objects or other types of content that can't be obtained without user interaction.

In the next chapter, we'll look at how you can save, export, distribute, and import flows.

4

Section 4 - Administering the Power Automate Environment

Shifting gears, the reader will learn about the administration side of the Power Automate environment.

This section comprises the following chapters:

- `Chapter 15`, *Exporting, Importing, and Distributing Flows*
- `Chapter 16`, *Monitoring, Managing, and Troubleshooting Flows*

15
Exporting, Importing, and Distributing Flows

At some point, it may be necessary (or useful) to be able to make a backup copy of a flow for archival purposes or move it between environments. There could be several reasons for this, including:

- Promotion from development or test environments to production
- Sharing with other teams or organizations
- Archival or backup
- Publishing to the Power Automate community

Whatever your requirements, Power Automate gives you the ability to back up or export, import, and share your flows.

In this chapter, we're going to explore the following concepts:

- Exporting a flow
- Importing a flow
- Distributing a flow

By the end of this chapter, you should be comfortable with the processes for exporting (or backing up), restoring (or importing), and sharing flows that others can then import.

Exporting a flow

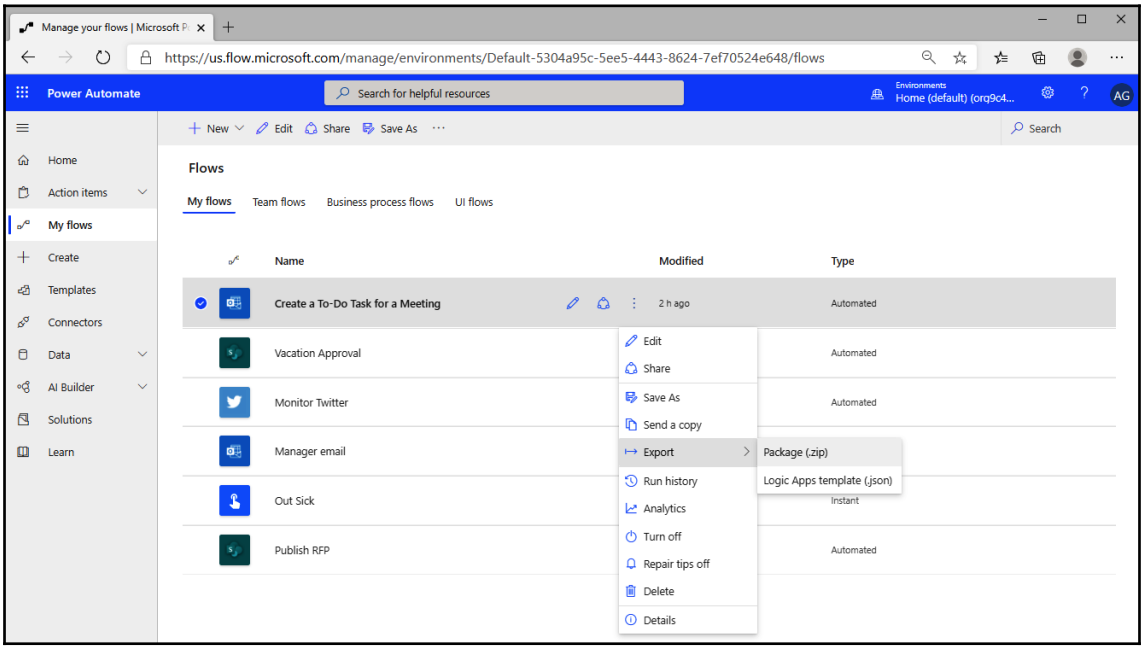
If you want to back up or prepare a flow for use in another environment, you'll need to export it. Exporting a flow makes it portable and gives you the ability to move it. A flow can be exported into different formats:

- A package (`.zip` for importing into Power Automate)
- A Logic Apps template (`.json` for importing into Logic Apps)

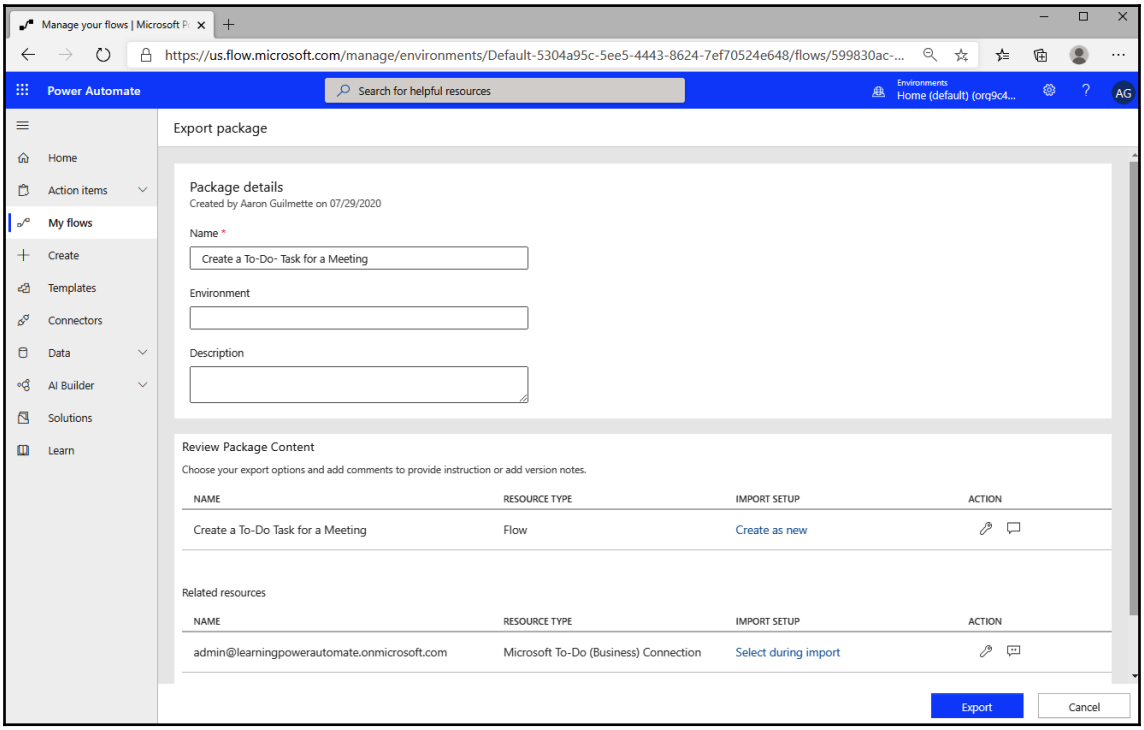
The export format you choose depends on what the target environment is. The formats are very similar, as Power Automate is built on the Logic Apps platform. In this example, we'll look at the steps for exporting a template for reuse in Power Automate (in either the same or different environments).

To export a flow, follow these steps:

1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **My flows.**
2. Click the ellipsis next to the flow you wish to export, point to **Export**, and then select **Package (.zip)**:



3. On the **Export package** page, enter a name for the package. You can add comments to each of the resource types by clicking on the corresponding speech bubble icon in each resource row. In the **IMPORT SETUP** column for the **Flow** resource type, click and select **Create as new** to create the flow as new in the target environment or **Update** to update an existing flow in the target environment:



4. Click **Export**. Power Automate will build a ZIP file and trigger the browser to initiate the download.

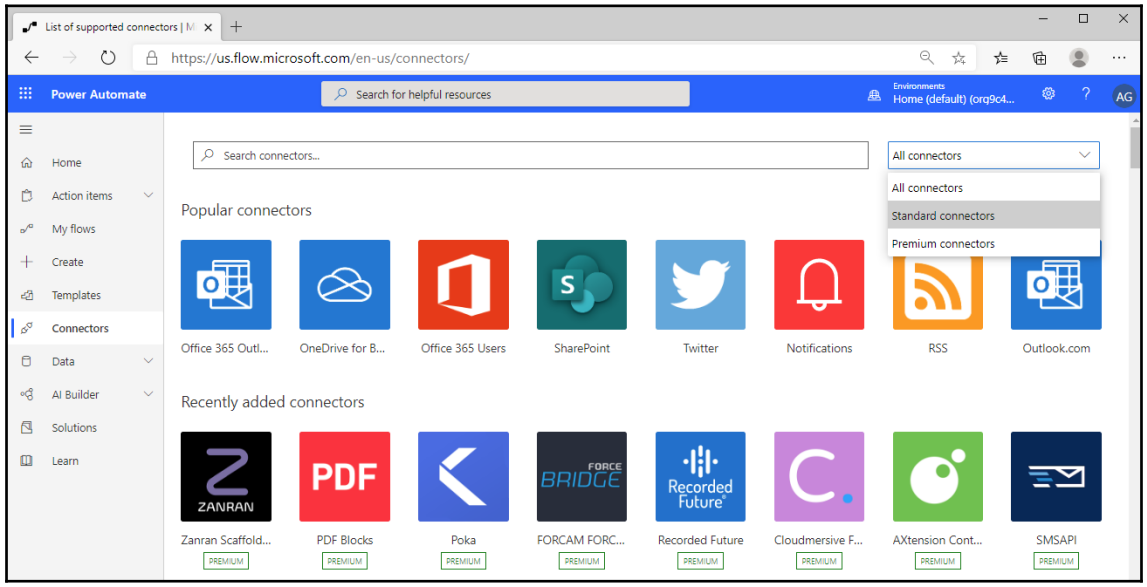
Once the export has finished downloading, you can distribute it as desired or store a copy for backup purposes.

Next, we'll look at importing a flow into the environment.

Importing a flow

Importing a allows you to take the exported contents from one environment and reuse the logic. The process is very simple, allowing you to move flows easily between environments.

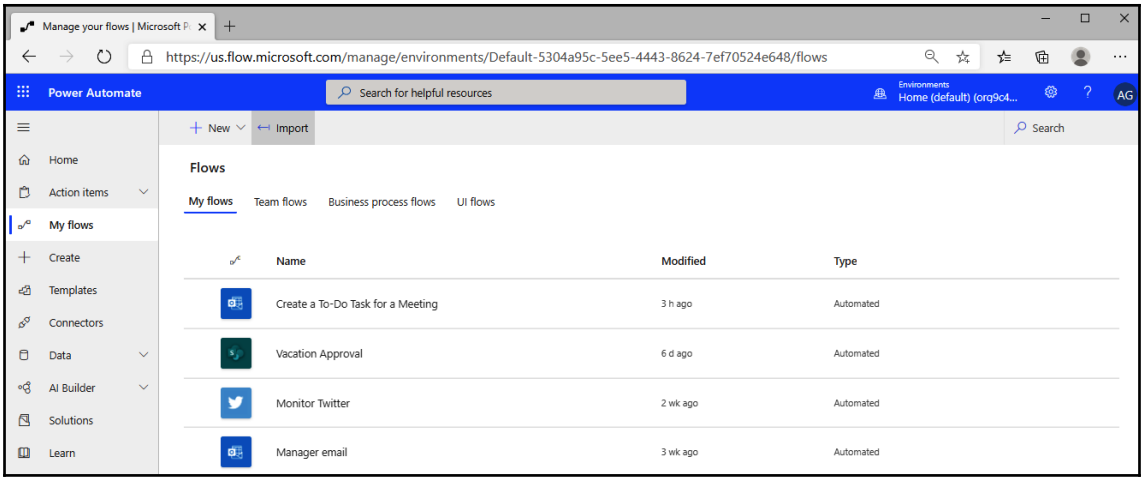
You'll want to make sure that the flows you are importing are supported by your environment (for example, the differences between standard connectors that come with Power Automate for Office 365 versus premium connectors only available for paid plans). Premium connectors are denoted by the word **PREMIUM** underneath the connector name, as shown in the following screenshot:



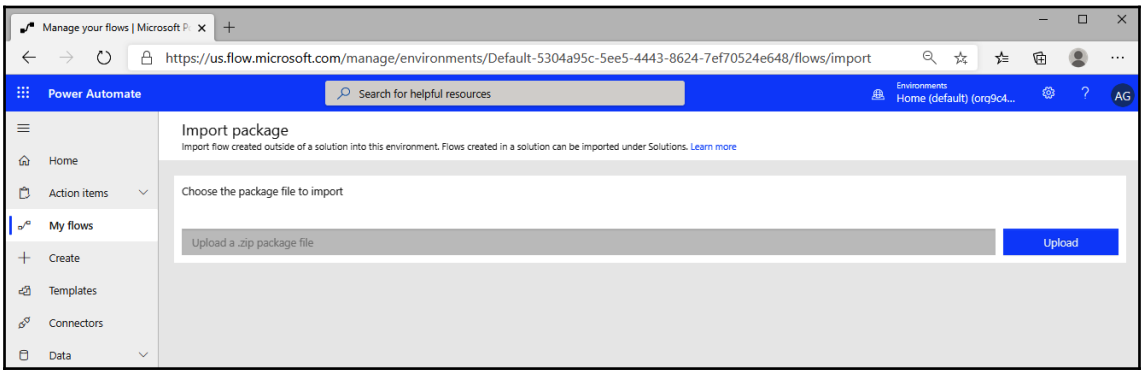
For examples and information on standard and premium connectors, refer to the following link: `https://flow.microsoft.com/en-us/connectors/`.

To import a flow, follow these steps:

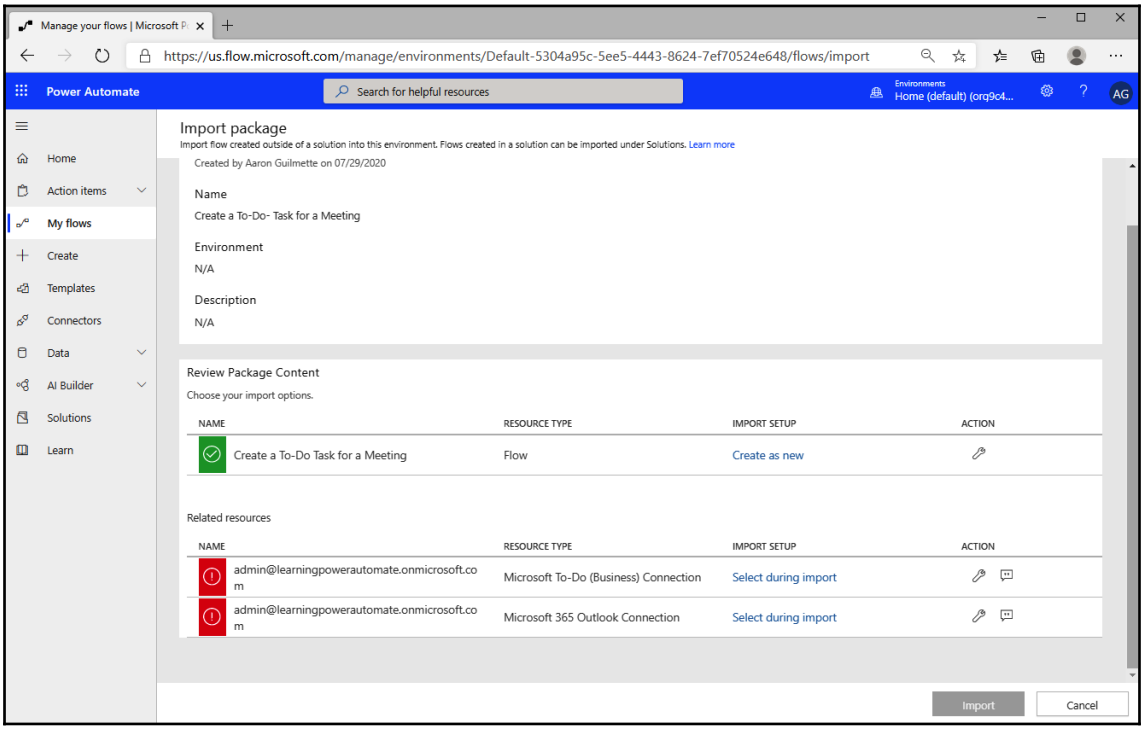
1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **My flows**.
2. Click **Import**:



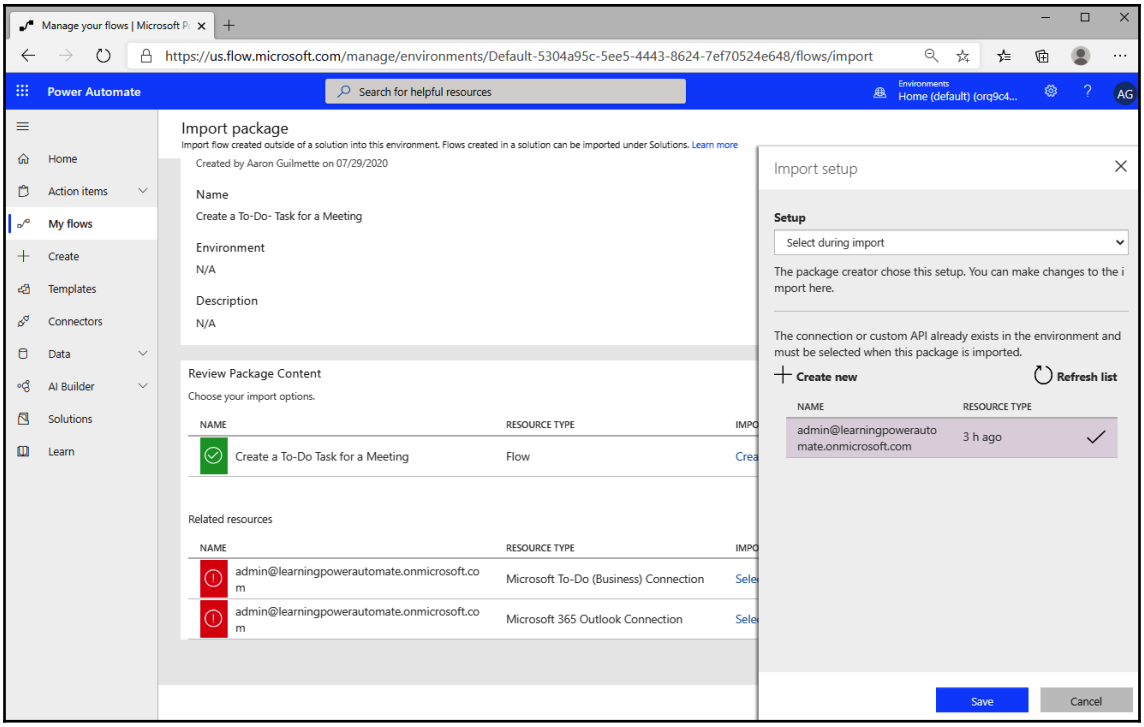
3. Click **Upload** to open a File Explorer window and browse to the `.zip` package file containing the flow you wish to import:



4. Click the wrench icon under the **ACTION** column to update any items (typically credential objects) that have a red exclamation icon:



5. Select an existing resource or click **+ Create new** to add a new resource of the correct type. Then, click **Save**:



6. When finished, click **Import**.

The flow will be created in the environment. As with any flow template import or creation, edit it to ensure that any variables or pick lists are updated with the correct objects to ensure the flow will work as expected. If a service or resource type is not currently available in the target environment, you'll need to create it during the import step. You'll need to repeat that process for every service used in a flow that isn't already configured in the target environment.

Next, we'll look at ways to distribute a flow.

Distributing a flow

If you have exported a flow and want to share it with others, you can do so in several ways:

- By sending the exported `.zip` file as an email attachment
- By posting the exported `.zip` file on a file hosting or sharing site such as OneDrive or DropBox
- By sending a flow via Power Automate
- By publishing the flow as a template to the gallery

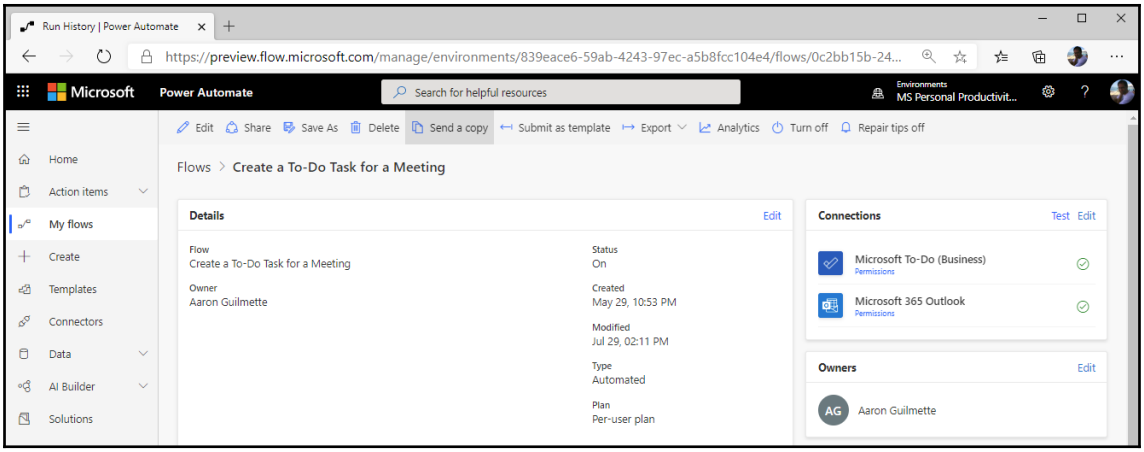
As the first two are relatively simple, we'll focus on the steps necessary to perform the last two methods: sending a flow via Power Automate and publishing the flow as a template to the gallery. Both of these methods will allow users to download and install their own copy of the flow, as you'll see.

Sending a flow via Power Automate

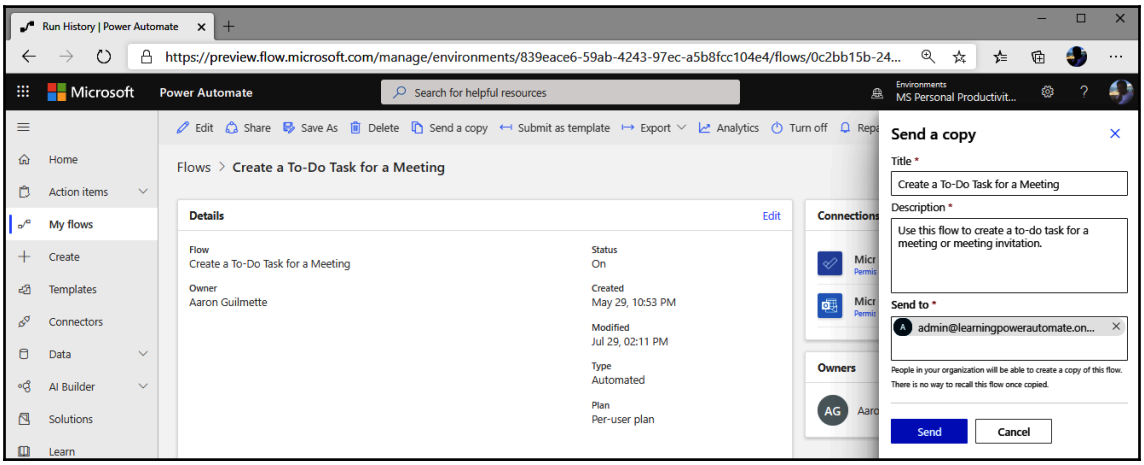
When you send a flow via Power Automate, you are allowing the recipients to download and create a copy of the flow. This is *different* to sharing a flow and converting it to a team flow, as you learned about in `Chapter 7`, *Working with Team Flows*. With a team flow, all users have access to the *same* flow instance. When you *send* a flow, each recipient gets a copy of it that they can personalize.

To send a flow via Power Automate, follow these steps:

1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **My flows** and then select the flow you wish to publish.
2. Select the **Send a copy** option:

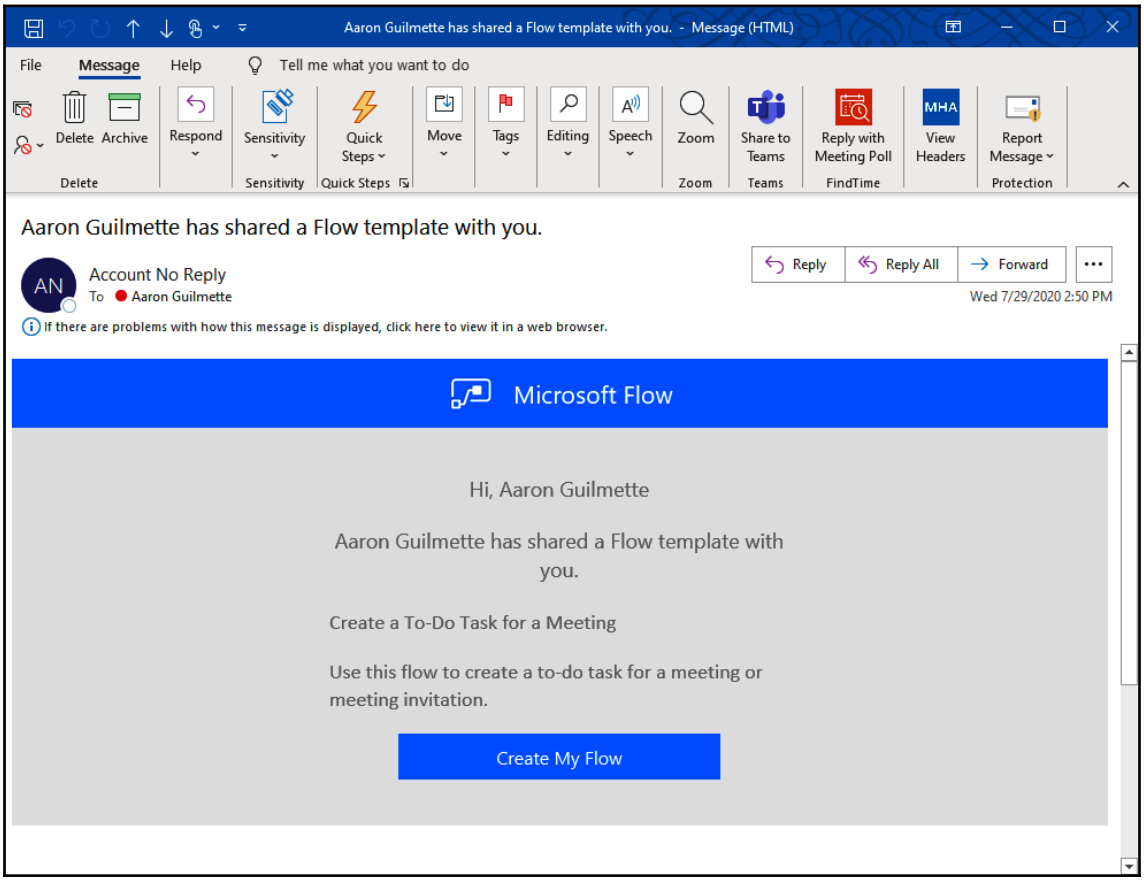


3. Enter the email addresses of users to whom you wish to send the flow, and then click **Send**:

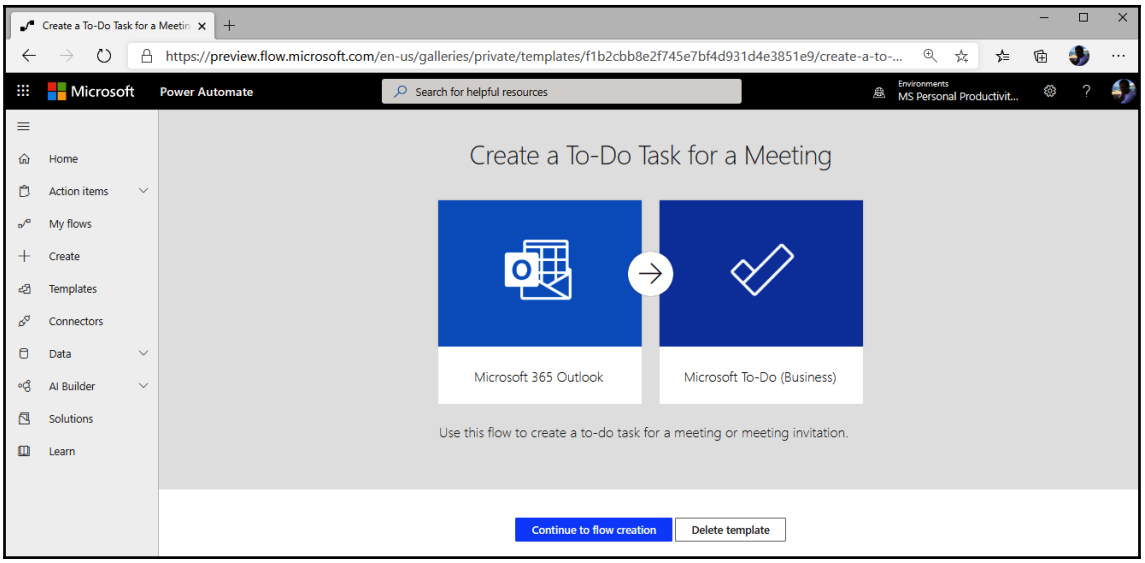


4. The recipient should receive an email prompting them to add the flow to their environment.

5. If you are the recipient of a sent flow, select the **Create My Flow** button in the email to start the import and configuration process:



6. Click the **Continue to flow creation** button once the browser window displays the flow details:



7. Update the credentials as necessary and then click **Create Flow**.

The flow will be created in the environment. As with any flow template import or creation, edit it to ensure any variables or pick lists are updated with the correct objects to ensure that the flow will work as expected.

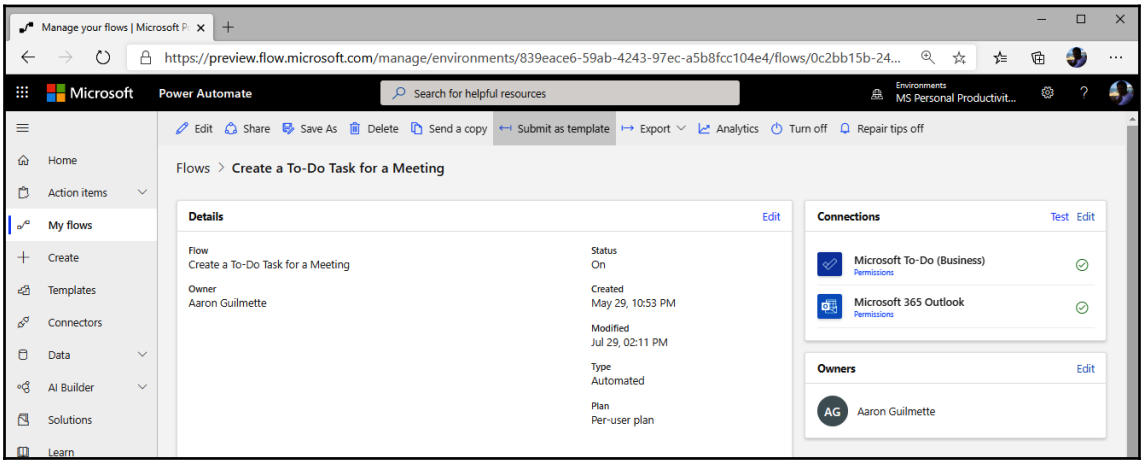
Publishing a flow to the template gallery

In order to submit a flow to the gallery, the most recent run *must* have been successful. This helps ensure that the flow will be successful for other users as well.

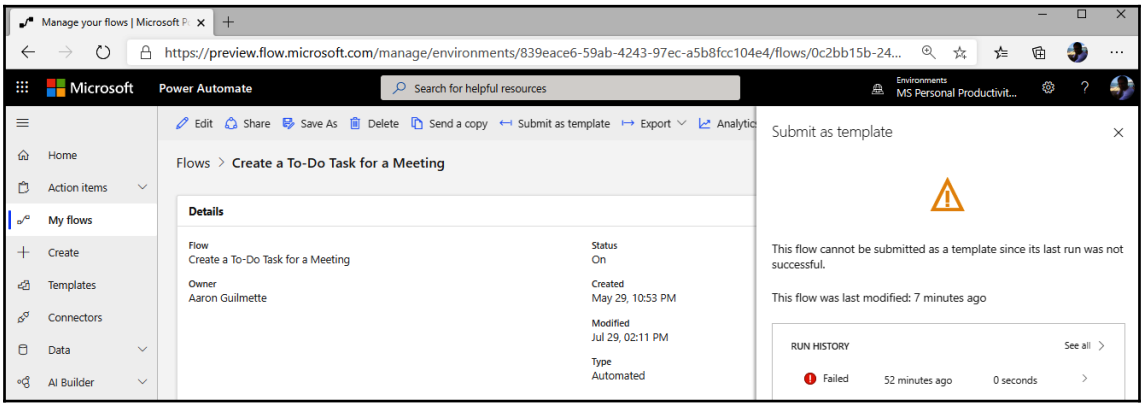
To submit a flow as a template to the gallery, follow these steps:

1. Log in to the Power Automate web portal (`https://flow.microsoft.com`). Click **My flows** and then select the flow you wish to publish.

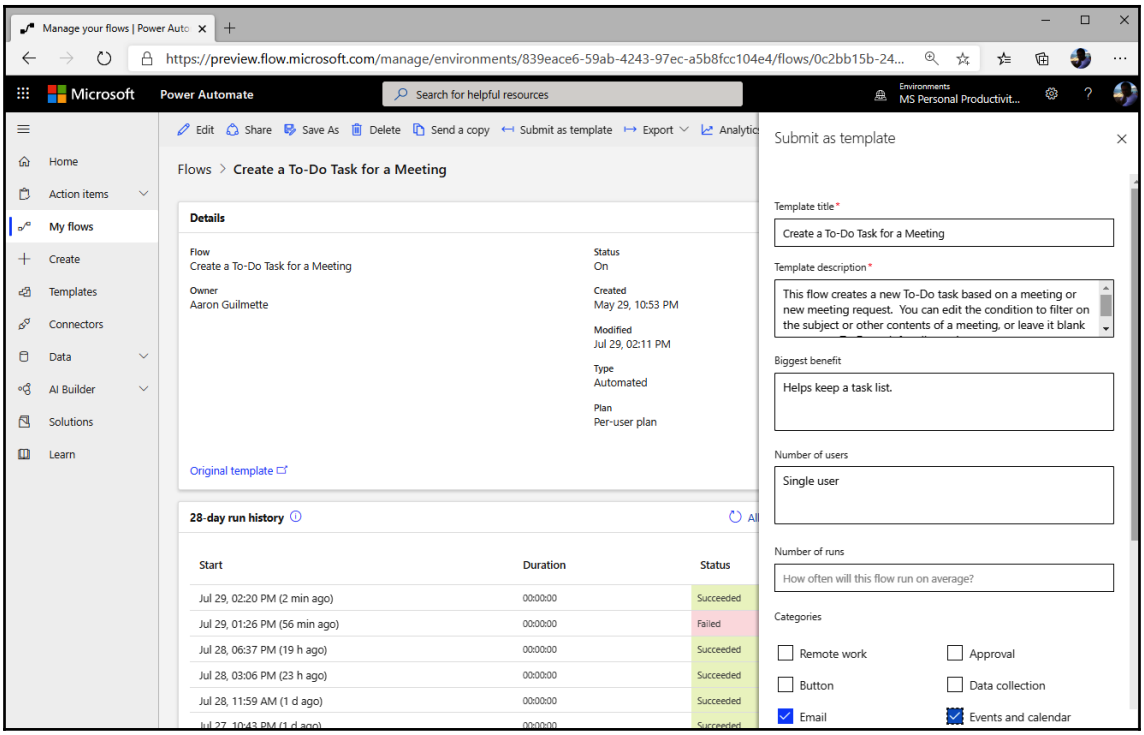
2. Select the **Submit as template** option. Power Automate will evaluate the run history of your flow:



3. If the run history shows the last run as a failure, you will be unable to submit it. Correct the flow, test it, and ensure that it is successful:



4. If the last flow run was successful, you will be prompted to fill out details, such as a title, description, and categories. When finished, scroll to the bottom of the panel and click **Submit**:



If accepted, your flow will be selectable from the Power Automate template gallery.

Summary

In this chapter, you learned how to export, import, send, and publish flows using the Power Automate interface. Exporting allows you to save a copy for archival purposes, send to colleagues or external entities, or use it to reimport the flow into another environment. Importing a flow allows you to take a previously exported flow and bring it into the current environment. The final tasks you learned about were the two ways in which you can distribute flows in Power Automate – by using either the **Send flow** or **Publish** capabilities to make the flow available to others.

In the next chapter, we'll look at monitoring and troubleshooting flows.

16
Monitoring and Troubleshooting Flows

Despite your best efforts, at times, there will be flows that fail. It could be something as simple as an incorrect or expired credential or something more challenging to troubleshoot, such as a third-party vendor updating their interface and rendering your process invalid.

In this chapter, we're going to review some of the troubleshooting features and capabilities of Power Automate. These tips and techniques will help you troubleshoot failing flows. We'll cover the following:

- Monitoring flows
- Reviewing email error reports
- Resolving authentication errors
- Examining detailed errors with the flow checker
- Understanding error codes
- Finding additional resources

By the end of this chapter, you'll understand how to troubleshoot common errors and know where to go for additional help.

Let's go!

Monitoring flows

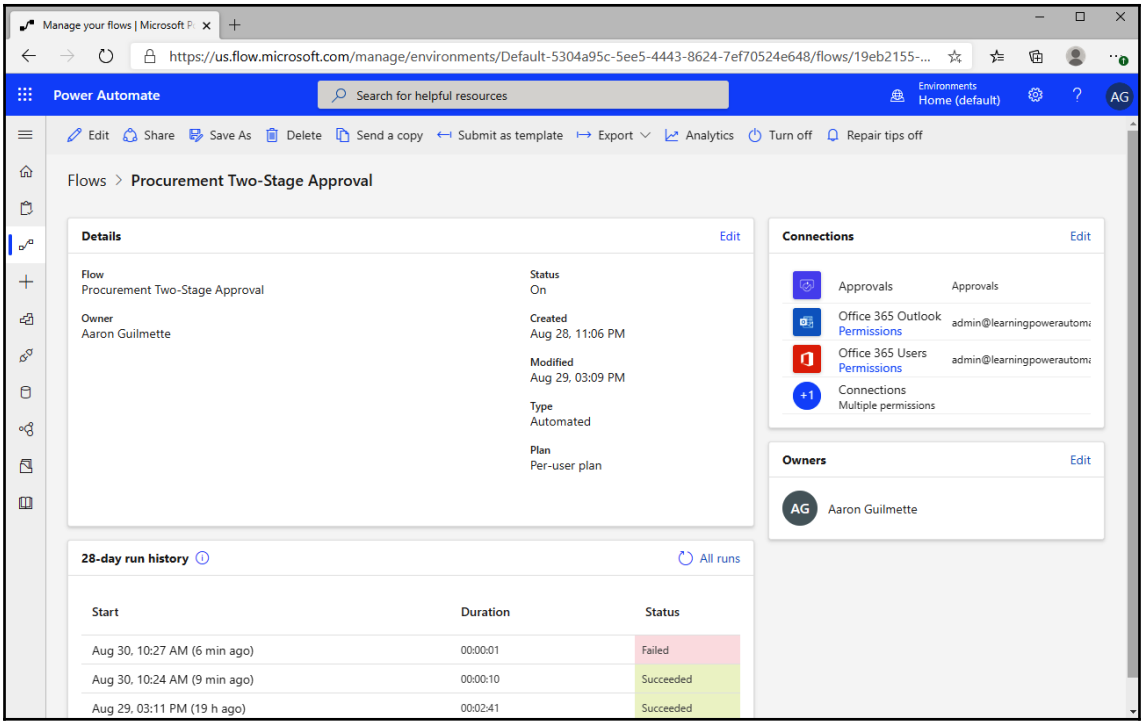
Throughout this book, you've seen examples of the run history for a particular flow. In this section, we'll review the run history of a flow that is experiencing errors and troubleshoot it.

For this example, we'll use one of the purchase request approvals that we created previously that relies upon checking for a user's manager. To experience an error, you need to submit the flow as a user that does not have a manager. In this example, we'll use the **Procurement Two-Stage Approval** from `Chapter 10`, *Working with Multiple Approvals*.

Viewing a run history

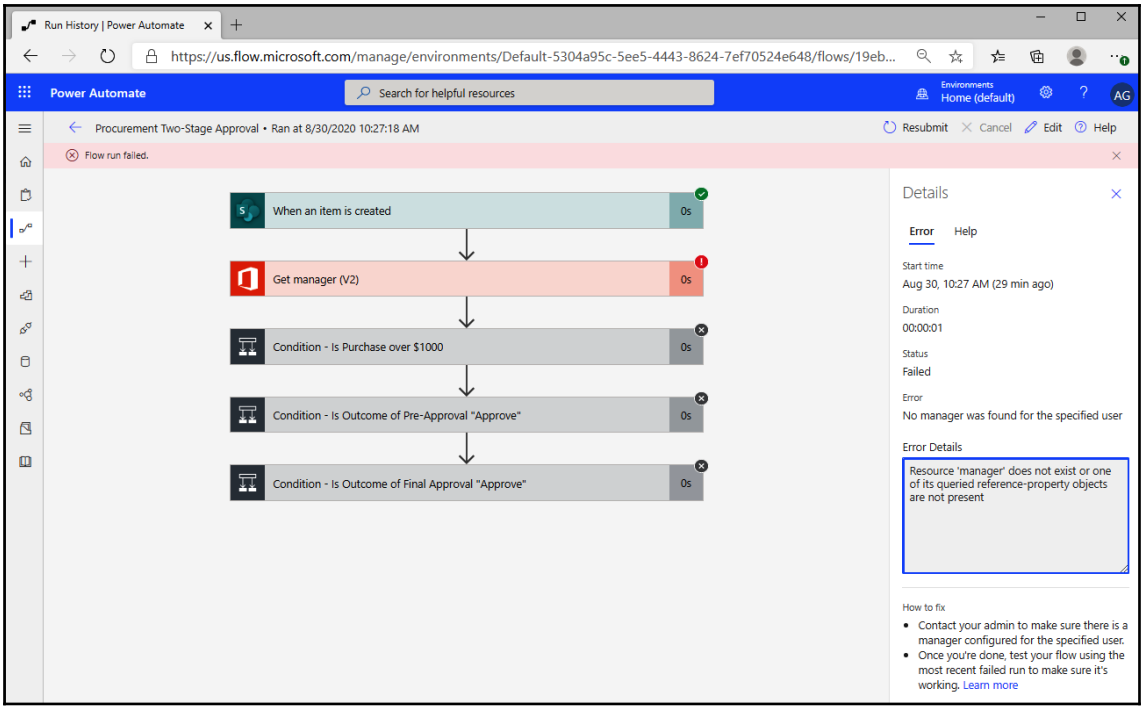
The run history will display a log of all the times that a flow has executed over the previous 28 days. To view the run history, follow these steps:

1. Using one browser session, log in to Office 365 (`https://portal.office.com`) as a user without a manager value assigned.
2. Navigate to the procurement approval site and submit a request. If you haven't created this workflow, review `Chapter 10`, *Working with Multiple Approvals*, and create a similar workflow.
3. Using a second browser session, navigate to the Power Automate web portal (`https://flow.microsoft.com`) and click **My flows**.
4. Select the **Procurement Two-Stage Approval** workflow. The run history will be displayed toward the bottom of the page:

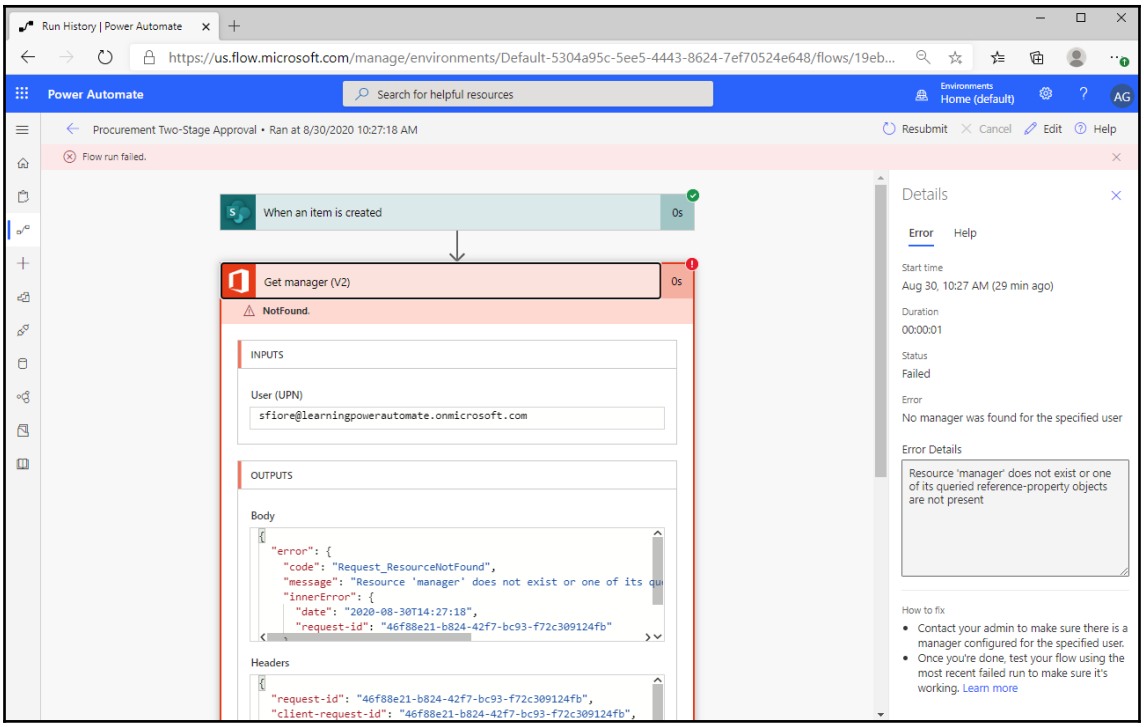


5. Under the **Start** column in the **28-day run history** section, click the link for the date.

6. The flow run will be displayed. Errors will be displayed in the fly-out panel on the right side of the page. The step-in error will have a red exclamation mark on it as well:



7. Most errors should have detailed information regarding the issue with the failed step. You can also click the title bar of the failed step in the main window to review the data that was evaluated during the step:



Most errors provide details in easy-to-understand language, helping you clearly understand the source of the problem.

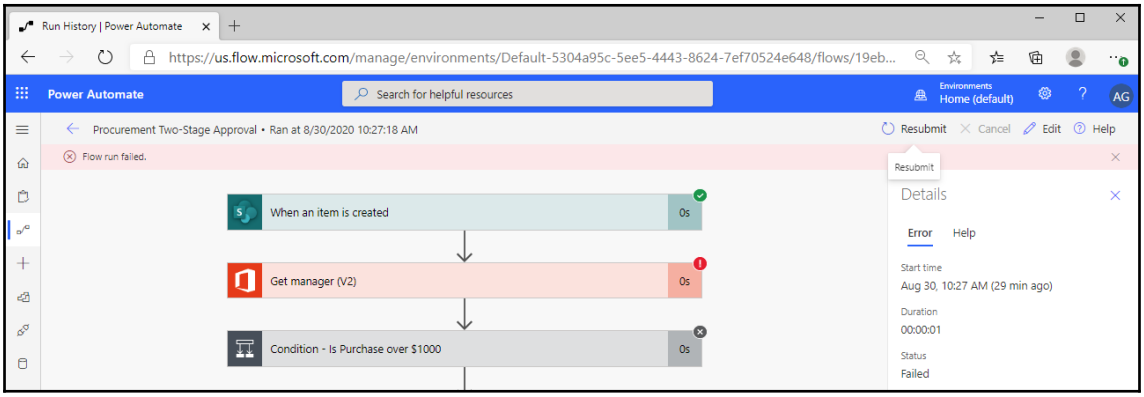
Next, we'll see how to work with flows where you've resolved the issue.

Resubmitting a flow

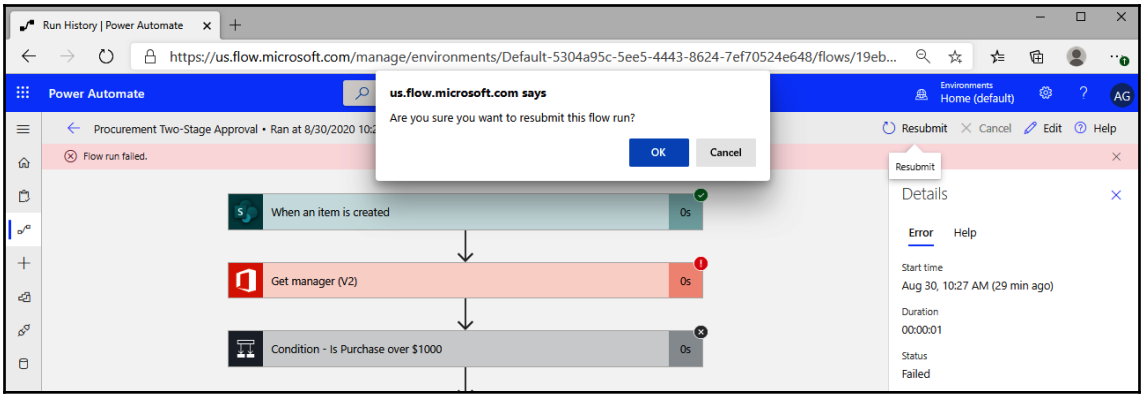
Once you have resolved the error (in this instance, assigning a "manager" value to the user who submitted the flow), you can have Power Automate attempt to process the flow again by resubmitting it.

To resubmit a flow, follow these steps:

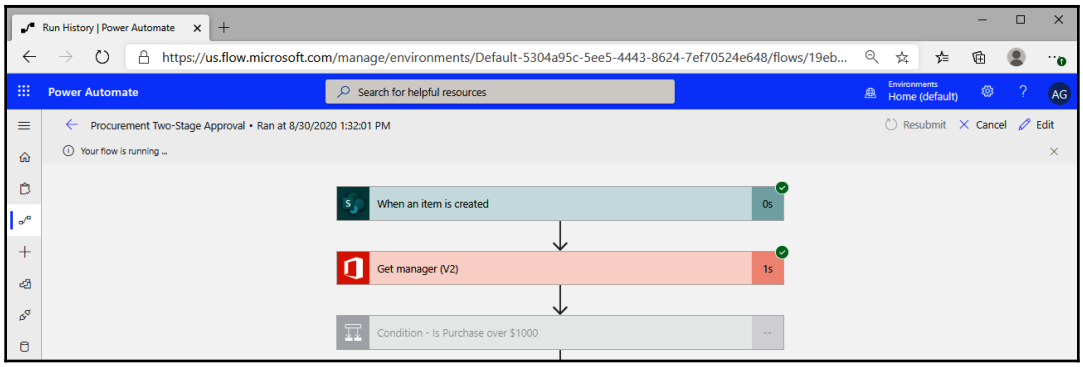
1. Open the failed run from the flow's run history.
2. Click the **Resubmit** button:



3. Confirm the popup by clicking **OK**:



4. Confirm that the flow is functioning as expected:

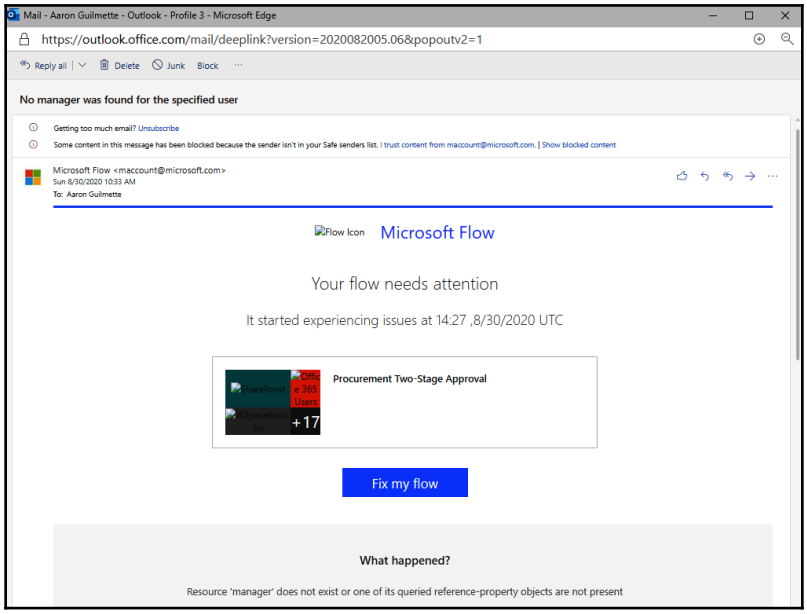


If further errors are encountered, review the run history and examine the error details.

Next, we'll look at the email error reports that are generated when a flow fails.

Reviewing email error reports

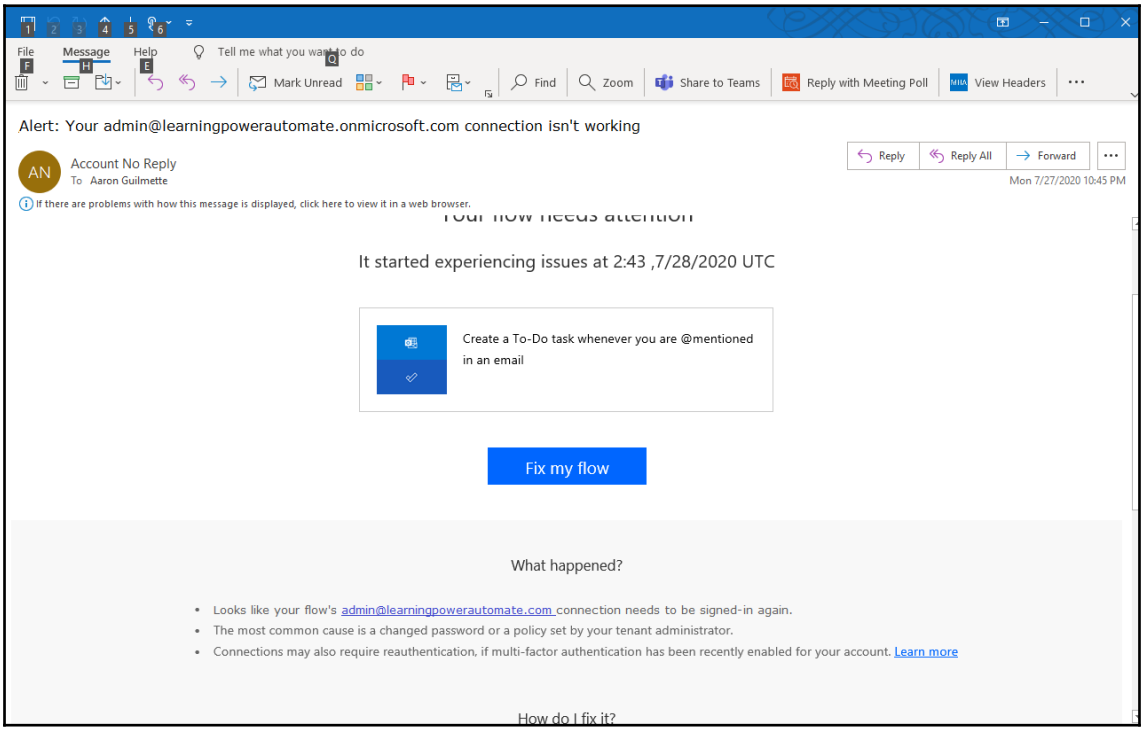
When you are the creator or owner of a flow and it experiences an error, Power Automate will generate an email and send it to you. An example of an email error report is displayed in the following screenshot:



The message will contain information similar to what is displayed in the details panel of a run history error, as shown in the previous section. Clicking on **Fix my flow** will direct you to the **Edit your flow** page.

Resolving authentication errors

One of the most common flow errors is due to an authentication failure. Consider the following error report email:



This error can occur when multifactor authentication is newly enabled for a sign-in or your credentials have changed.

Common authentication error messages include the term **Unauthorized** or may display an error code of **401** or **403**. These error codes generally indicate that the username and password combination is incorrect. If the username and password are correct, a 403 error code may mean that the account you're attempting to use to access a resource does not have the required permissions.

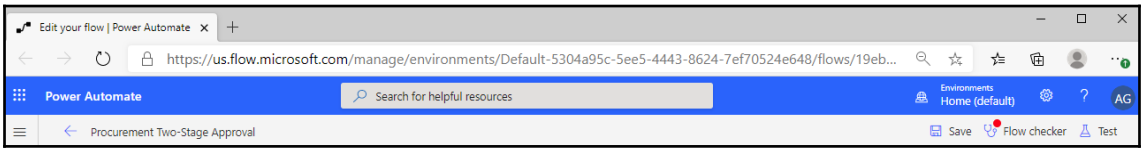
To resolve authentication errors, sign in to Power Automate, edit the flow, and then update the connection settings. On the run history detail page, you may be able to click the **View Connections** link to update the credentials being used.

In the next section, we'll see how the flow checker can help you identify issues.

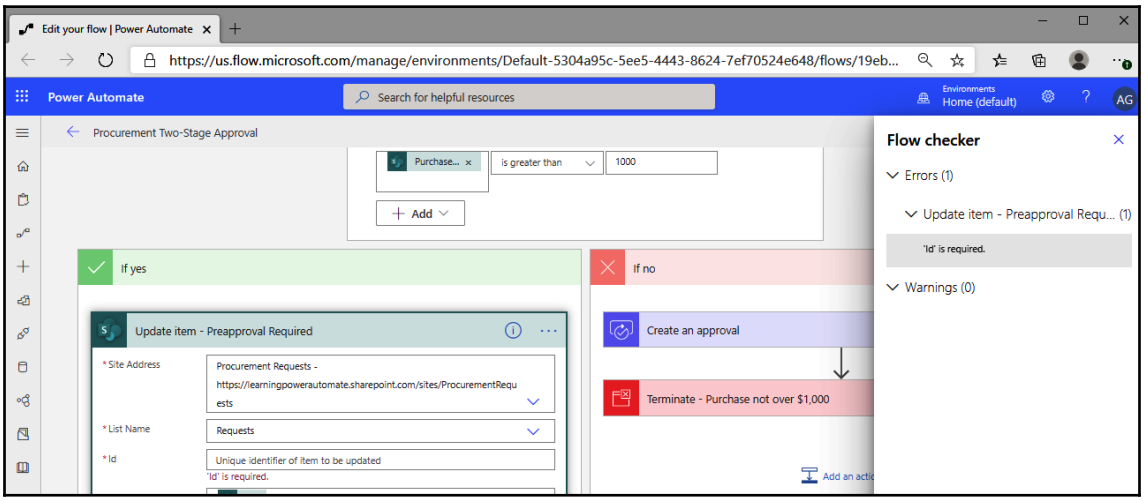
Examining detailed errors with the flow checker

The flow checker is a verification and troubleshooting tool built into the Power Automate interface. It can be used during the development and troubleshooting process to help identify potential issues. The flow checker continuously evaluates your flows for potential performance, reliability, and configuration issues.

If the flow checker determines that there is a potential error, a red bubble is displayed next to the **Flow checker** icon:



To review the errors, warnings, or issues detected by the flow checker, click on the link. The details will be displayed in the details fly-out pane. In this example, the checker displays an error that indicates a required field has not been configured:



Update the items that the indicator identifies. You can then save the flow and re-run the checker by clicking on the icon.

Next, we'll look at additional error codes that may be present when troubleshooting flows.

Understanding error codes

You may also encounter other errors when interacting with services. Usually, Power Automate will return the actual error code it encounters when accessing a service. Here are some example error codes and some potential causes:

Error code	Details	Recommendation
401, 403	Unauthorized. This is usually due to incorrect credentials, but it could also indicate that an account does not have the correct permissions.	Verify that the account credentials are correct and that the account has access to the resource.

400	Bad request. This error usually means that the type of data being submitted is of the incorrect type or format (such as sending CSV-type data if a field is anticipating a binary image file). It could also happen if you are attempting to send data to an incorrect URL endpoint or using an incorrect method (for example, using an HTTP GET method when you should be using an HTTP POST method).	Verify the type of data a particular flow action requires. Verify that the correct URL is being used. If using an advanced HTTP connector, verify that the correct method (GET or POST) is being used.
404	Not found. This error means that the file or resource specified in the action is not found at the URI or URL specified.	Verify that the URI or URL paths specified are valid.
409	Conflict. This error may occur when two processes are attempting to update the same resource simultaneously.	Verify that only one action or process is attempting to update a given resource at a time.
423	File locked. This error can occur when you are attempting to access a resource that is locked for editing by another person or is checked out.	Verify that the file is not currently open by another user or process.
429	Rate limit exceeded. This can happen if you encounter an API interface request limit (too many requests in too short a period of time).	Check the flow run history to see how many runs are being executed. Check the connected service for any rate limit warnings. Reduce the number of steps or calls in the flow or run the flow less frequently.
502	Bad gateway. This can occur when you specify a URL as HTTP instead of HTTPS or if you are attempting to access a resource protected by a proxy server that is not responding correctly.	Verify the correct HTTP protocol. If the service is hosted behind a firewall or proxy server, ensure that it is configured correctly.
503	Service unavailable. This error indicates that the service you are attempting to reach is unresponsive, partially offline, or under heavy load.	Wait and retry. If the error persists, contact the service vendor to see whether the service is available.
504	Gateway timeout. This error can occur for services that are protected by a firewall or proxy and the connection times out or is reset. Additionally, this can also indicate that a service is being interrupted by an intrusion protection service.	Verify that any proxy or firewall services protecting a resource are configured correctly and available and that no intrusion protection services are blocking requests.

If some of these error codes look familiar, it is due to Power Automate being built on standard web service protocols and accessing REST-based API resources. Review the errors encountered in your flow to determine whether these common errors are present.

Next, we'll review some additional resources for troubleshooting.

Finding additional resources

There are a number of resources available to help troubleshoot and resolve Power Automate issues:

Resource	**URL**
Power Automate documentation	`https://docs.microsoft.com/en-us/power-automate/`
Power Automate community	`https://powerusers.microsoft.com/t5/Microsoft-Power-Automate/ct-p/MPACommunity`
Power Automate user group	`https://www.automateug.com/home`

If you're using third-party apps as part of your flows, you'll also want to review any REST API documentation that they provide. That is an invaluable resource when troubleshooting connectivity and functionality with service providers.

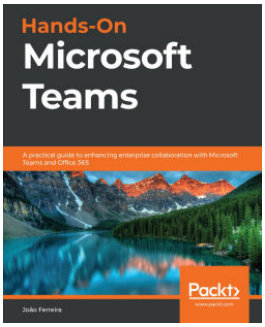
Summary

In this chapter, you learned how to use some of the troubleshooting resources available in Power Automate, including the run history and email error reports. Additionally, you learned how the flow checker can help identify potential problems for your flows. Finally, you learned about other resources available to help expand your knowledge and skills with Power Automate.

Congratulations on making it all the way to the end of the book! We hope you walk away with an increased knowledge of how Power Automate can help improve your productivity as well as having some real-world examples of how to make Power Automate part of your toolbox.

Other Books You May Enjoy

If you enjoyed this book, you may be interested in these other books by Packt:



Hands-On Microsoft Teams
João Ferreira

ISBN: 978-1-83921-398-4

- Create teams, channels, and tabs in Microsoft Teams
- Explore the Teams architecture and various Office 365 components included in Teams
- Perform scheduling, and managing meetings and live events in Teams
- Configure and manage apps in Teams
- Design automated scripts for managing a Teams environment using PowerShell
- Build your own Microsoft Teams app without writing code

Learn Microsoft PowerApps
Matthew Weston

ISBN: 978-1-78980-582-6

- Design an app by simply dragging and dropping elements onto your canvas
- Understand how to store images within PowerApps
- Explore the use of GPS and how you can use GPS data in PowerApps
- Get to grips with using barcodes and QR codes in your apps
- Share your applications with the help of Microsoft Teams and SharePoint
- Use connectors to share data between your app and Microsoft's app ecosystem

Leave a review - let other readers know what you think

Please share your thoughts on this book with others by leaving a review on the site that you bought it from. If you purchased the book from Amazon, please leave us an honest review on this book's Amazon page. This is vital so that other potential readers can see and use your unbiased opinion to make purchasing decisions, we can understand what our customers think about our products, and our authors can see your feedback on the title that they have worked with Packt to create. It will only take a few minutes of your time, but is valuable to other potential customers, our authors, and Packt. Thank you!

Index

D

E

F

G

I

M

O

P

R

S

T

U

W