

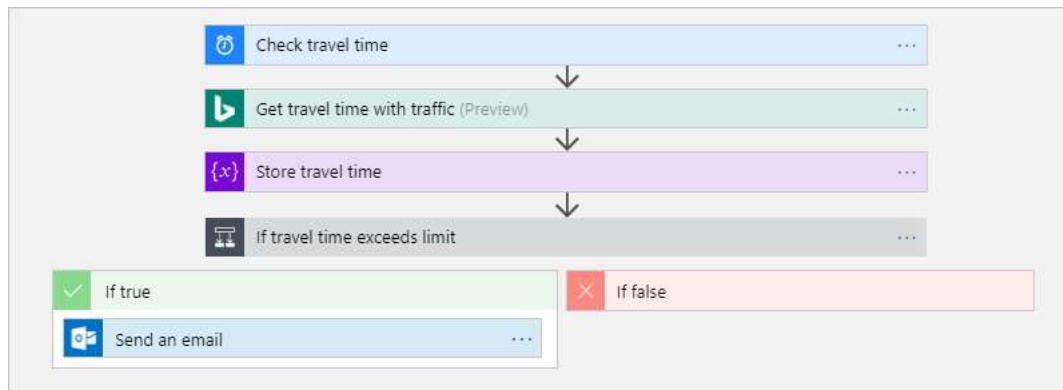
Module 14 - Lab 5: Logic App Workflow (Advanced - Check traffic on a schedule with Azure Logic Apps)

- ❓ Azure Logic Apps helps you automate workflows that run on a schedule. This lab shows how you can build a logic app with a scheduler trigger that runs every weekday morning and checks the travel time, including traffic, between two places. If the time exceeds a specific limit, the logic app sends email with the travel time and the extra time necessary for your destination.

In this task, you learn how to:

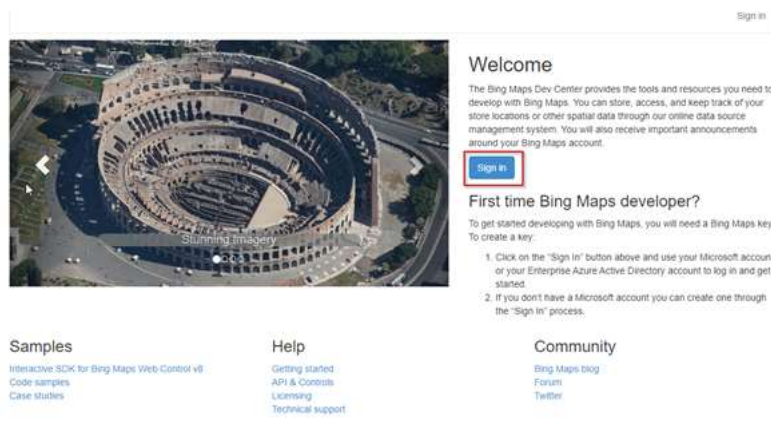
- Create a blank logic app.
- Add a trigger that works as a scheduler for your logic app.
- Add an action that gets the travel time for a route.
- Add an action that creates a variable, converts the travel time from seconds to minutes, and saves that result in the variable.
- Add a condition that compares the travel time against a specified limit.
- Add an action that sends email if the travel time exceeds the limit.

When you're done, your logic app looks like this workflow at a high level:



Task 1: Create a Bing Maps Key

- ☐ 1. Open a browser and go to the Bing Maps Dev Center by navigating to the following URL: <https://www.bingmapsportal.com/>
- ☐ 2. Sign in with the username sheikhnasirN2MWV@gdcs1.com and password [3DQyVZD7kZWgEdf7](#)



- ☐ 3. Select **Yes, lets create a new account.**

Welcome to the Bing Maps Dev Center!

This is the first time you have logged into Bing Maps Dev Center using . If you would like to continue to create your new Bing I continue. If you already have a Bing Maps Dev Center account please sign in using your original login information.

[Yes, let's create a new account](#)

[Sign in with another account](#) (This will sign you out of , so you can sign in with another account or create a new one.)

- ☐ 4. Enter your details on the Create Account page then click **Create**.

Create account

Account details

Account name *

Enter account name

Account name is required.

Contact name

Enter contact name

Company name

Enter company name

Email address * -This email address will receive important service announcements and notifications.

Enter email

Phone number

Enter phone number

☐ I agree to the [Bing Maps Platform APIs' Terms of Use \(TOU\)](#).

Contact Preferences

I would like information, tips and offers about Bing Maps. [Privacy Statement](#)

☐ Email ☐ Telephone

Create

* Required field

- ☐ 5. On the Update Account Details page click **Save**.

Update Account Details

Contact Preferences

☐ I would like information, tips and offers about Bing Maps. [Privacy Statement](#)

Save

- ☐ 6. Under **My Account**, Select **My keys**.



- ☐ 7. Select the option to create a new key and provide the following information to create a key:
- **Application name:** AzureLab
 - **Application URL:** Leave blank
 - **Key type:** Required. Basic
 - **Application type:** Dev/Test

Click the Create button. The new key displays in the list of available keys. Use this key to authenticate your Bing Maps application as described in the documentation for the Bing Maps API you are using.

My keys

Create key

Application name *

AzureLab

Application URL

Enter application URL

Key type *

[What's This](#)

Basic ▾

Application type *

Dev/Test ▾

Create

Cancel

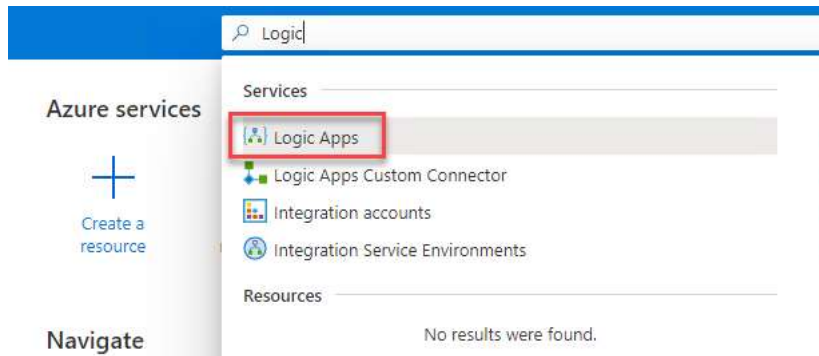
* Required field

To create Education, Broadcast or Not-for-Profit keys, please contact the Bing Maps account team at monet@microsoft.com.

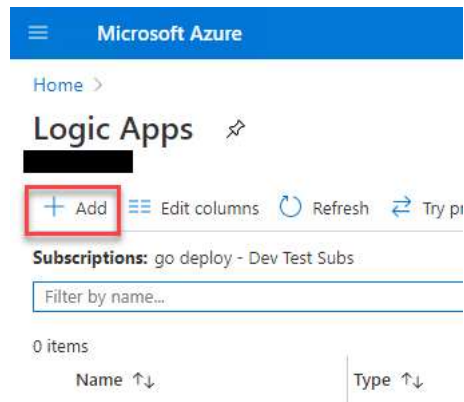
- ☐ 8. Leave the Bing Maps Portal open for later in this task.

Task 2: Create your logic app

Login to the **Azure Portal**  <https://portal.azure.com> search for and select **Logic App**.

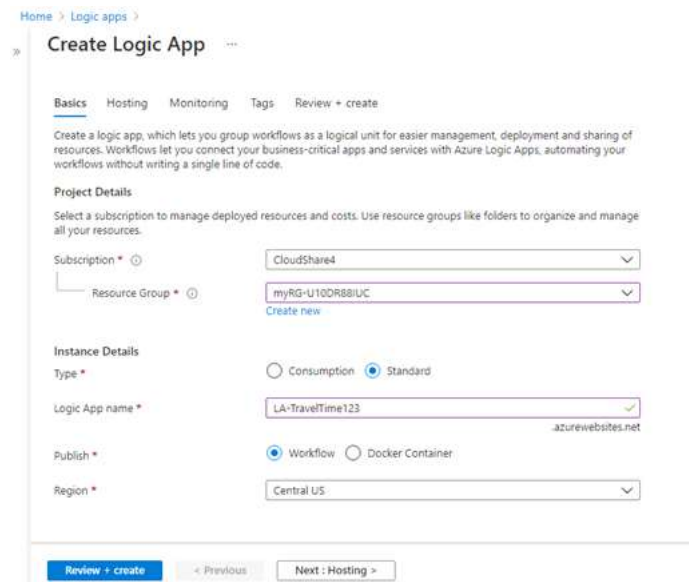


- ☐ 1. Click + **Add** to create a new Logic App.



- ☐ 2. Under **Create logic app**, provide this information about your logic app as shown and described then click **Review + create** and then select **Create**.

Setting	Value	Description
Subscription	<your-Azure-subscription-name>	The name for your Azure subscription
Resource group	myRG-GOPLXY6K24	The name of the Resource Group that was deployed for you
Name	LA-TravelTimeXXX	The name for your logic app (Replacing XXX with something unique)
Region	East US	The region where to store information about your logic app



- ☐ 3. After Azure deploys your Logic App, click **Go to Resource** and the Logic Apps Overview opens. Under **Workflows**, choose **Workflows**.

Home > Microsoft.Web-LogicApp-Portal-13b0b041-a1f8 >

LA-TravelTime123

Logic App (Standard)

Search (Ctrl+/) << Browse Refresh Stop Restart Swap

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Security

Events (preview)

Workflows

Workflows

Connections

Parameters

Artifacts

Click here to access Application Insights for monitoring and pro

Essentials

Resource group (change) : myRG-U10DR88IUC

Status : Running

Location : East US

Subscription (change) : CloudShare4

Subscription ID : ae4932b2-c869-49c1-a952-74e5247

Tags (change) : [Click here to add tags](#)

[See more](#)

Features (8) **Notifications (1)**

Filter by name...

- ☐ 4. Select **+Add** to create a new workflow.

Home > Microsoft.Web-LogicApp-Portal-13b0b041-a1f8 > LA-TravelTime123

LA-TravelTime123 | Workflows

Logic App (Standard)

Search (Ctrl+/) << Add Refresh Enable Disable Delete

Filter by name...

Name ↑

No results.

- ☐ 5. On the **New Workflow** window type in **LogicAppWorkFlow** as the Workflow name. Ensure the state type is set to **Stateful** and select **Create**.

New workflow

Create a new workflow in this logic app.

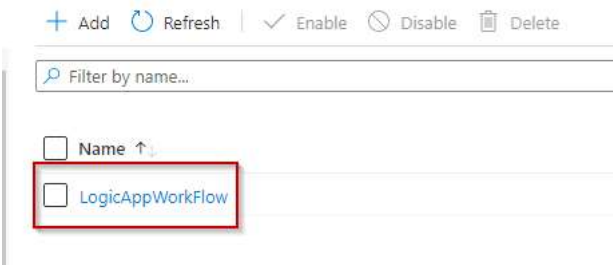
Workflow Name *

State type *

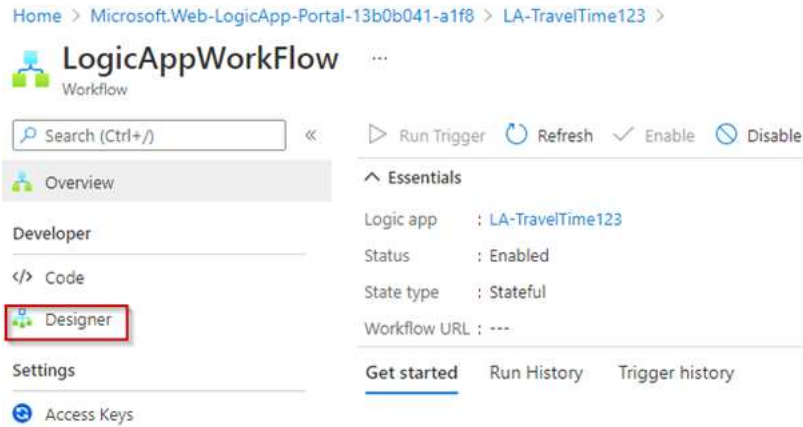
☒ Stateful: Optimized for high reliability, ideal for process business transitional data.

☐ Stateless: Optimized for low latency, ideal for request-response and processing IoT events.

- ☐ 6. On the Workflows blade, Select **LogicAppWorkFlow**.

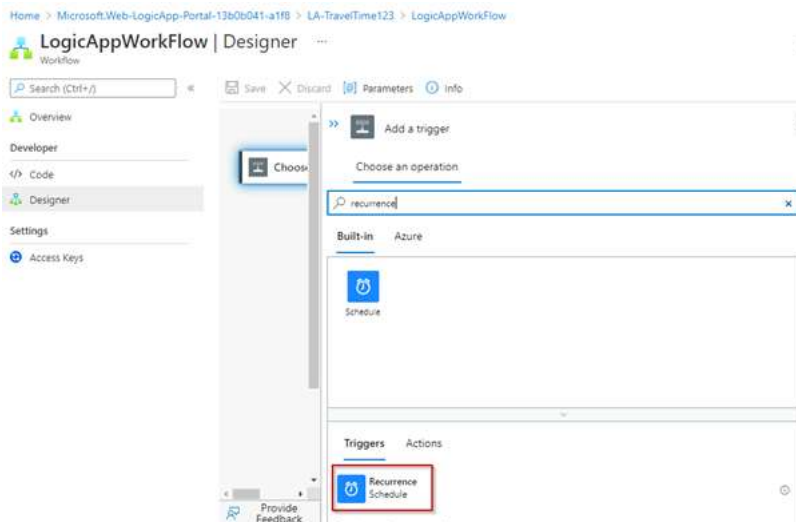


- ☐ 7. Select **Designer** under **Developer** in the resource menu on the left side.

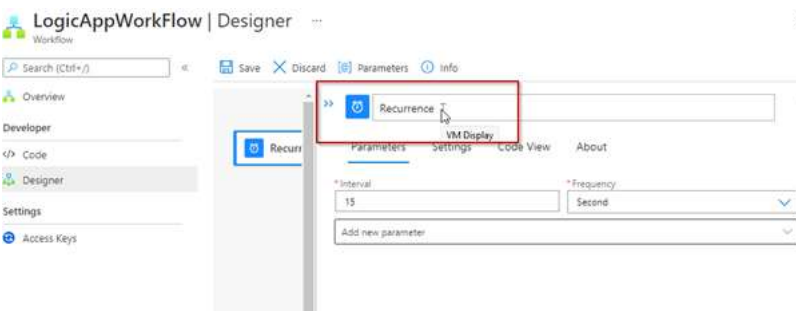


Task 3: Add the Recurrence trigger

- ☐ 1. On the Logic App Designer, in the search box, enter "recurrence" as your filter. From the **Triggers** list, select the **Recurrence** trigger.



- ☐ 2. Select the **Recurrence** text, and then rename the trigger with this description: **Check travel time every weekday morning**



- ☐ 3. Inside the trigger, change these properties.

Property	Required	Value	Description
Interval	Yes	1	The number of intervals to wait between checks
Frequency	Yes	Week	The unit of time to use for the recurrence

☐ 4. Under **Interval** and **Frequency**, open the **Add new parameter** list, and select these properties to add to the trigger.

- **On these days**
- **At these hours**
- **At these minutes**

Microsoft Azure | Search resources, services, and docs (G+)

Home > Microsoft.Web-LogicApp-Portal-037ac1a4-9880 > raklogicapp2 > LogicAppWorkflow

LogicAppWorkflow | Designer

Workflow

Search (Ctrl+/) Save Discard Parameters Info

Overview

Developer

Code

Designer

Settings

Access Keys

Check travel time every weekday morning

Parameters Settings Code View About

*Interval: 1 *Frequency: Week

On these days: Monday

At these hours: Example: 0, 10

At these minutes: Enter the valid minute values (from 0 to 59) separated by comma, e.g., 15,30

Preview: Runs on Monday every week

Add new parameter

ENG US 2:10 AM 4/12/2022

- ☐ 5. Now set the values for the additional properties as shown and described here.

Microsoft Azure | Search resources, services, and docs (G+)

Home > Microsoft.Web-LogicApp-Portal-037ac1a4-9880 > raklogicapp2 > LogicAppWorkflow

LogicAppWorkflow | Designer

Workflow

Search (Ctrl+/) Save Discard Parameters Info

Overview

Developer

Code

Designer

Settings

Access Keys

Check travel time every weekday morning

Parameters Settings Code View About

*Interval: 1 *Frequency: Week

On these days: Monday,Tuesday,Wednesday,Thursday,Friday

At these hours: 7,8,9

At these minutes: 0,15,30,45


Preview: Runs at 7:00, 8:00, 9:00 on Monday, Tuesday, Wednesday, Thursday, Friday every week

Add new parameter

Google Chrome

ENG US 2:15 AM 4/12/2022


Property	Value	Description
On these days	Monday,Tuesday,Wednesday,Thursday,Friday	Available only when Frequency is set to "Week"

Property	Value	Description
At these hours	7,8,9	Available only when Frequency is set to "Week" or "Day". Select the hours of the day to run this recurrence. This example runs at the 7, 8, and 9-hour marks.
At these minutes	 0,15,30,45	Available only when Frequency is set to "Week" or "Day". Select the minutes of the day to run this recurrence. This example runs every 15 minutes starting at the zero-hour mark.

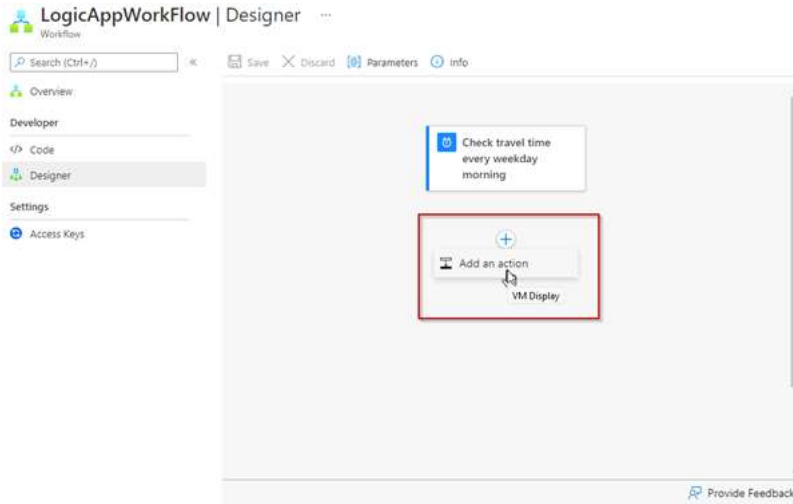
This trigger fires every weekday, every 15 minutes, starting at 7:00 AM and ending at 9:45 AM. The **Preview** box shows the recurrence schedule.


- ☐ 6. Save your logic app. On the designer toolbar, select **Save**.

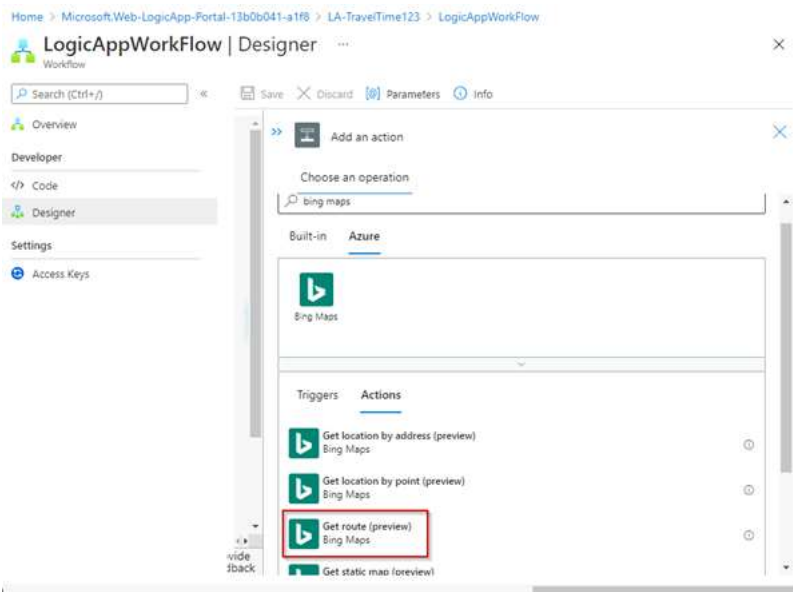
Task 4: Get the travel time for a route

 Now that you have a trigger, add an action that gets the travel time between two places. Logic Apps provides a connector for the Bing Maps API so that you can easily get this information. Before you start this task, make sure that you have a Bing Maps API key as described in this tutorial's prerequisites.

- ☐ 1. Minimize the trigger by clicking on the >> next to the trigger name. In the Logic App Designer, Under the trigger you just created, Select the + button follow by **Add an action** to create a new step.



- ☐ 2. Under **Choose an action**, select **Azure**. In the search box, enter  **bing maps** as your filter, and select the **Get route** action.



- ☐ 3. If you don't have a Bing Maps connection, you're asked to create a connection. Provide these connection details, and select **Create**.

Save Discard Parameters Info


>> Get route


Create Connection

* Connection name BingMapsConnection

* API Key

Create

Property	Required	Value	Description
Connection Name	Yes	 BingMapsConnection	Provide a name for your connection. This example uses "BingMapsConnection".
API Key	Yes	<your-Bing-Maps-key>	Enter the Bing Maps key that you previously created.

- ☐ 4. Rename the action with this description:  Get route and travel time with traffic
- ☐ 5. Inside the action, open the **Add new parameter list**, and select these properties to add to the action.
- Optimize
 - Distance unit
 - Travel mode
- ☐ 6. Now set the values for the action's properties as shown and described here.


Property	Required	Value	Description
Waypoint 1	Yes	<start-location>	Your route's origin
Waypoint 2	Yes	<end-location>	Your route's destination
Optimize	No	timeWithTraffic	A parameter to optimize your route, such as distance, travel time with current traffic, and so on. Select the "timeWithTraffic" parameter.
Distance unit	No	<your-preference>	The unit of distance for your route. This example uses "Mile" as the unit.
Travel mode	No	Driving	The travel mode for your route. Select "Driving" mode.

For more information about these parameters, see [Calculate a route](#).


- ☐ 7. Save your logic app.

Next, create a variable so that you can convert and store the current travel time as minutes, rather than seconds. That way, you can avoid repeating the conversion and use the value more easily in later steps.

Task 5: Create a variable to store travel time

-  Sometimes, you might want to run operations on data in your workflow, and then use the results in later actions. To save these results so that you can easily reuse or reference them, you can create variables to store those results after processing them. You can create variables only at the top level in your logic app.


By default, the previous **Get route** action returns the current travel time with traffic in seconds from the **Travel Duration Traffic** property. By converting and storing this value as minutes instead, you make the value easier to reuse later without converting again.

- ☐ 1. Under the Get route action, select the + icon and select **Add an action**.
- ☐ 2. Under **Choose an action**, select **Built-in**. In the search box, enter "variables", and select the **Initialize variable** action.
- ☐ 3. Rename this action with this description:  Create variable to store travel time
- ☐ 4. Provide the details for your variable as described here:

Property	Required	Value	Description
Name	Yes	travelTime	The name for your variable. This example uses "travelTime".
Type	Yes	Integer	The data type for your variable
Value	No	An expression that converts the current travel time from seconds to minutes (see steps under this table).	The initial value for your variable

- ☐ 5. To create the expression for the **Value** property, click inside the box so that the dynamic content list appears. If necessary, widen your browser until the list appears. In the dynamic content list, select **Expression**.

When you click inside some edit boxes, the dynamic content list appears. This list shows any properties from previous actions that you can use as inputs in your workflow. The dynamic content list has an expression editor where you can select functions to run operations. This expression editor appears only in the dynamic content list.



- ☐ 6. In the expression editor, enter this expression:  `div(,60)`.
- ☐ 7. Put your cursor inside the expression between the left parenthesis () and the comma (,). select **Dynamic content**.
- ☐ 8. In the dynamic content list, select **Travel Duration Traffic**.
- ☐ 9. After the property value resolves inside the expression, select **OK**.

The **Value** property now appears as shown here:

- ☐ 10. **Save** your logic app.

Next, add a condition that checks whether the current travel time is greater than a specific limit.

Task 6: Compare the travel time with limit

- ☐ 1. Under the **Create variable** action, select the + icon and select **Add an action**.
- ☐ 2. Under **Choose an action**, select **Built-in**. In the search box, enter "condition" as your filter. From the actions list, select the **Condition** action.
- ☐ 3. Rename the condition with this description:  `If travel time exceeds limit`
- ☐ 4. Build a condition that checks whether the **travelTime** property value exceeds your specified limit as described and shown here:
- ☐ 5. In the condition, click inside the **Choose a value** box on the condition's left side.
- ☐ 6. From the dynamic content list that appears, under **Variables**, select the **travelTime** property.
- ☐ 7. In the middle comparison box, select the **is greater than** operator.
- ☐ 8. In the **Choose a value** box on the condition's right side, enter this limit:  `15`

When you're done, the condition looks like this example:

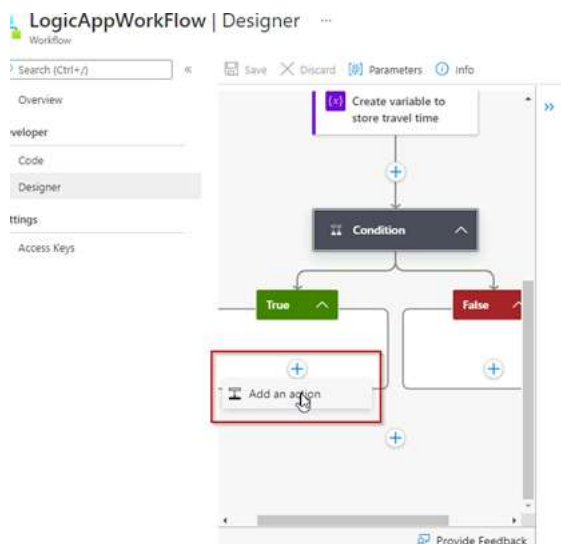
- ☐ 9. **Save** your logic app.

Next, add the action to run when the travel time exceeds your limit.

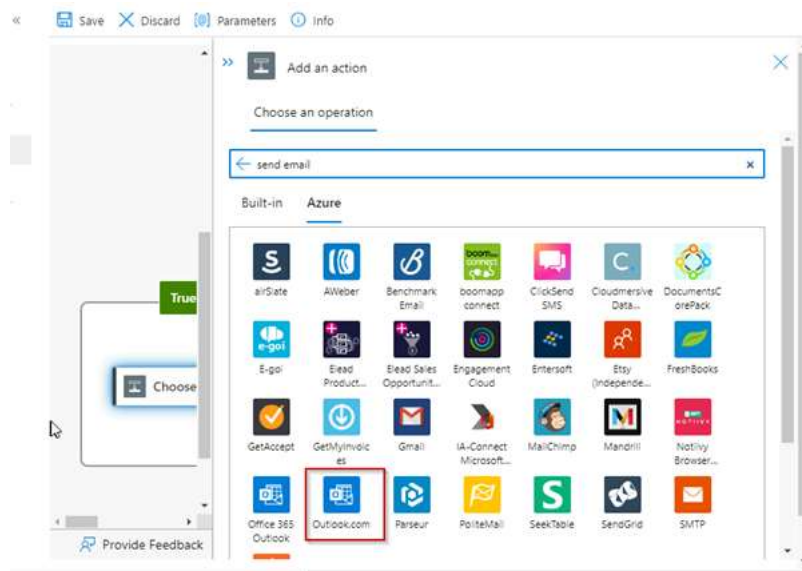
Task 7: Send email when limit exceeded

- ☐ ? Now, add an action that emails you when the travel time exceeds your limit. This email includes the current travel time and the extra time necessary to travel the specified route.

- ☐ 1. In the condition's **If true** branch, select **Add an action**.



- ☐ 2. Under **Choose an operation**, select **Azure**. In the search box, enter  `send email`. The list returns many results, so first select the email connector that you want, for example:






- For personal Microsoft accounts, select **Outlook.com**.

- ☐ 3. When the connector's actions appear, select "send email action" that you want to use, for example:
- ☐ 4. If you don't already have a connection, you're asked to sign in to your email account.

Logic Apps creates a connection to your email account.

⚠ For this task please use your personal email

- ☐ 5. Rename the action with this description:  Send email with travel time
- ☐ 6. In the **To** box, enter the recipient's email address. For testing purposes, use your email address.
- ☐ 7. Enter the text  Current travel time (minutes); with a trailing space.
- ☐ 8. In the **Subject** box, specify the email's subject, and include the **travelTime** variable.
- ☐ 9. In the dynamic content list, under **Variables**, select **See more**.
- ☐ 10. After **travelTime** appears under **Variables**, select **travelTime**.
- ☐ 11. In the **Body** box, specify the content for the email body.
- ☐ 12. Enter the text  Add extra travel time (minutes); with a trailing space.
- ☐ 13. In the dynamic content list, select **Expression**.
- ☐ 14. In the expression editor, enter this expression so that you can calculate the number of minutes that exceed your limit: `sub(,15)`
- ☐ 15. Put your cursor inside the expression between the left parenthesis (and the comma (,). Select **Dynamic content**.
- ☐ 16. Under **Variables**, select **travelTime**.
- ☐ 17. After the property resolves inside the expression, select **OK**.

The **Body** property now appears as shown here:

- ☐ 18. **Save** your logic app.

Next, test your logic app, which now looks similar to this example:

Task 8: Run your logic app

- ☐ 1. To manually start your logic app, on the **Overview** toolbar bar, select **Run Trigger** then **Run**.
 - If the current travel time stays under your limit, your logic app does nothing else and waits or the next interval before checking again.
 - If the current travel time exceeds your limit, you get an email with the current travel time and the number of minutes above your limit. Here is an example email that your logic app sends:

If you don't get any emails, check your email's junk folder. Your email junk filter might redirect these kinds of mails.

✓ Congratulations, you've now created and run a schedule-based recurring logic app.

To create other logic apps that use the **Recurrence** trigger, check out these templates, which available after you create a logic app:

- Get daily reminders sent to you.
 - Delete older Azure blobs.
 - Add a message to an Azure Storage queue.
-