Internet of Things

1. Azure IoT Foundation, theory

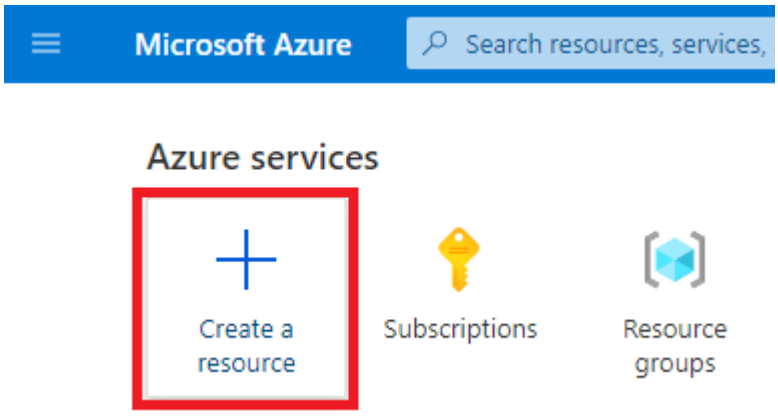
2. Getting familiar with Azure Portal https://portal.azure.com/

Exercise 1: IoT Hub provisioning

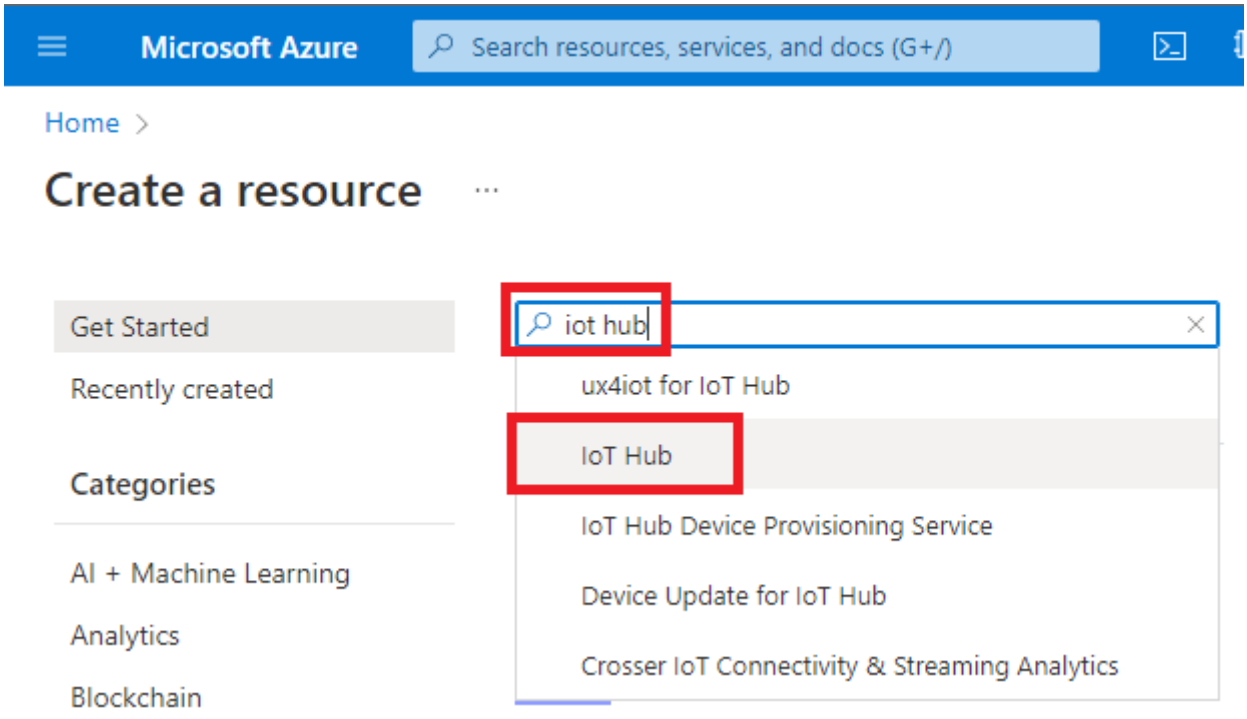
Task 1: Provision IoT Hub through the Portal

During this exercise you will use 3 different tools to create three different IoT Hubs, after this exercise we will delete two and continue the rest of the workshop with the first IoT Hub created through the Portal.

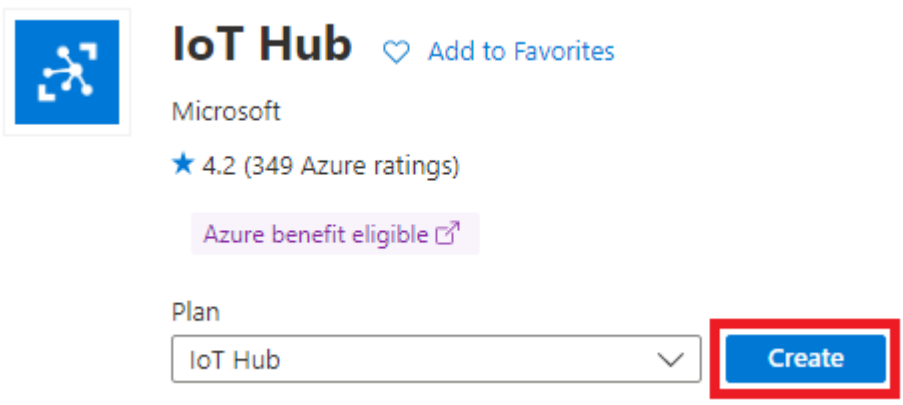
1. In your browser, navigate to the Azure portal, select **+Create a resource** in the navigation pane, enter `iot` into the **Search the Marketplace** box.



2. Select **IoT Hub** from the results

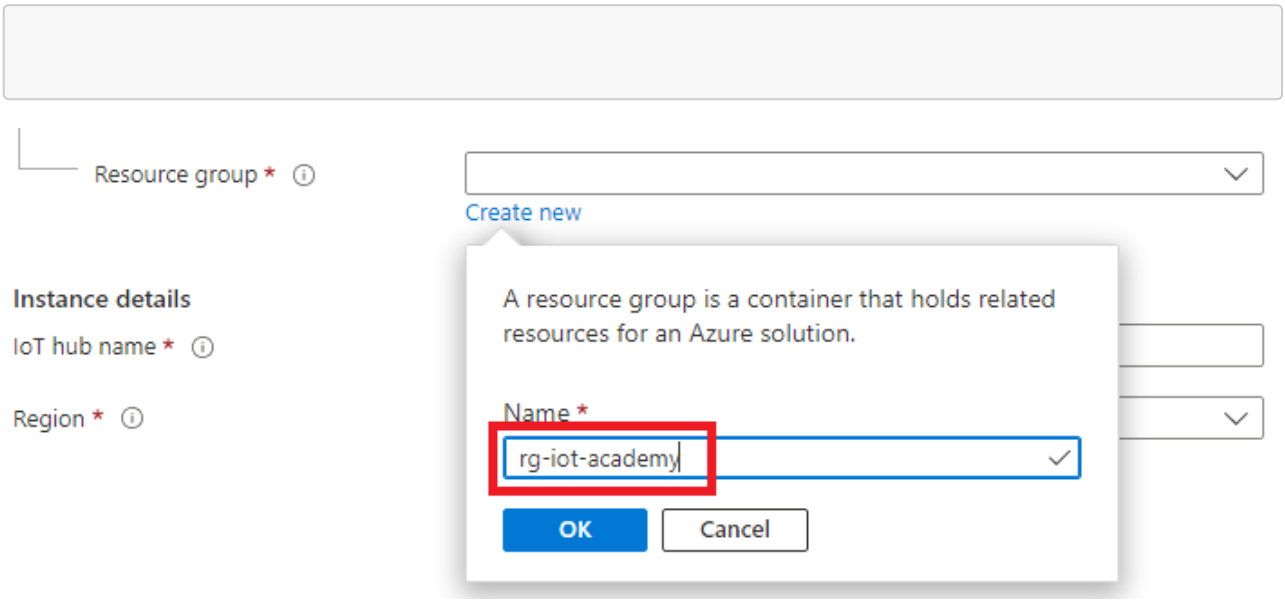


3. Select **Create**.



4. On the **IoT Hub** blade **Basics** tab, enter the following:

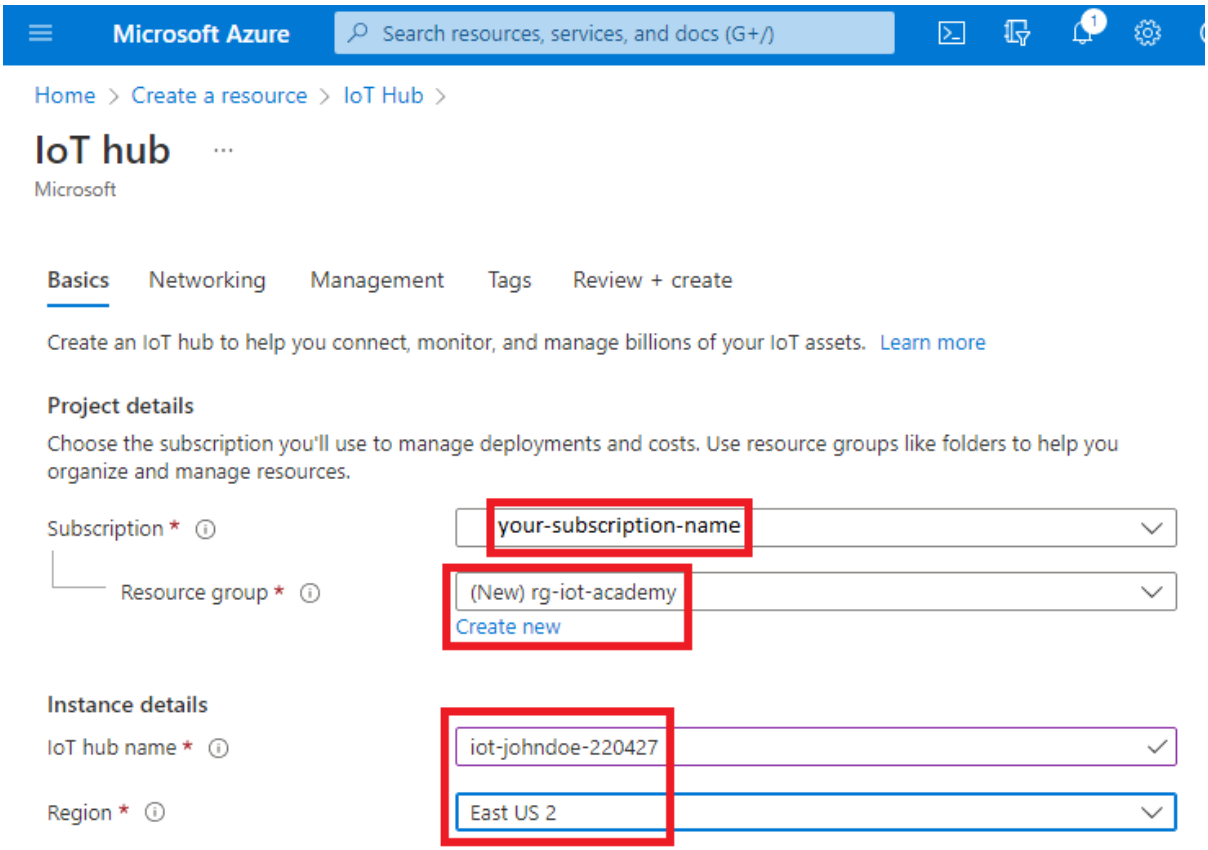
    - **Subscription**: Select the subscription you are using for this hands-on lab.
    - **Resource group**: Click Create new underneath Resource group. Enter the name `rg-iot-academy`



    - **Region**: Select the location you are using for this hands-on lab.
    - **IoT Hub Name**: Enter a unique name, such as `iot-academy-johndoe-220427`. The name follows best practices for naming resources in Azure. Note:

        1. the prefix `iot-`
        2. the inclusion of a name 'johndoe' and a date `220427(YYMMDD)` this combination such as `johndoe-220427` will be known as your **suffix**. You may want to copy this to your notepad so you can copy and paste it later.

        As some resources in Azure require unique names, the name and the date helps to avoid naming conflicts. More can be read concerning best practices for naming Azure resources at the following link: Azure Naming and Tagging For common resource prefixes refer to the following link: Azure Resource Abbreviations

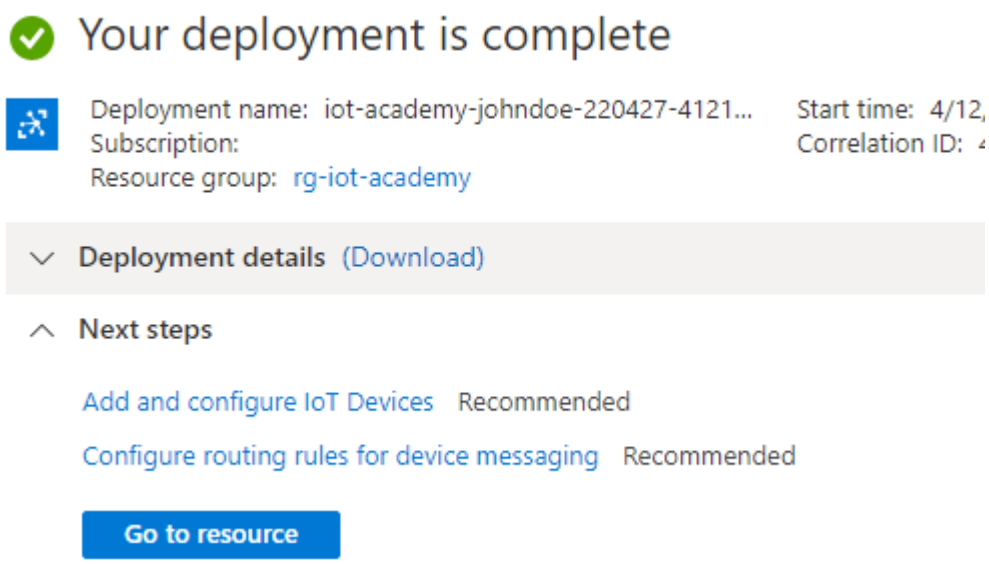


- Click **Next: Networking**.
- On the **Networking** tab ensure Public is selected
- Click **Next: Management**.
- On the **Management** tab

    1. **Pricing and scale tier**: ensure **S1: Standard tier** is selected
    2. **Number of S1 IoT hub units**: ensure **1** is selected
    3. **Defender for IoT**: set to **On**
    4. **Assign me to the IoT Hub Data Contributor role**: select the check box
    5. **Device-to-cloud partitions**: leave the default setting of **4**

- Click **Review + create**.
- Ensure validation passes and click **Create**.

5. After clicking create you were directed to a deployment overview page. When the deployment completes click the **Go to resource** button.

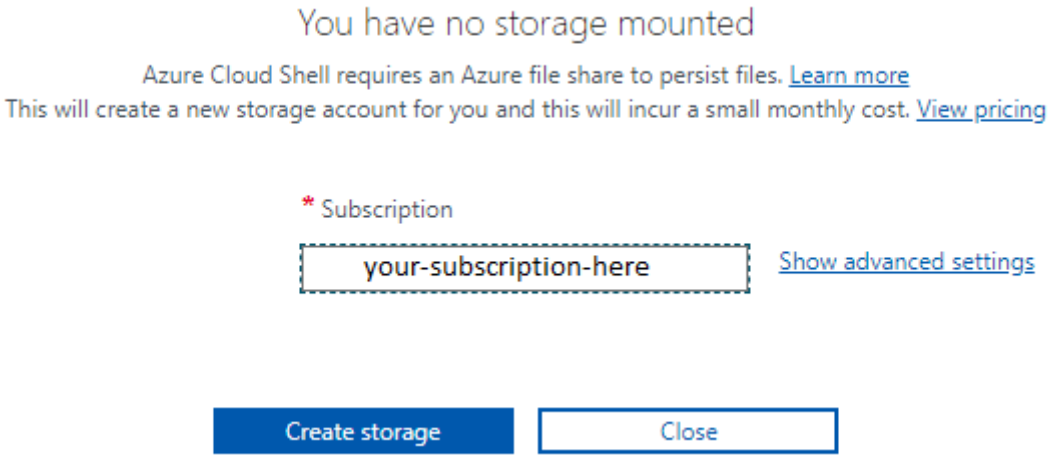


Task 2: Provision IoT Hub through CLI

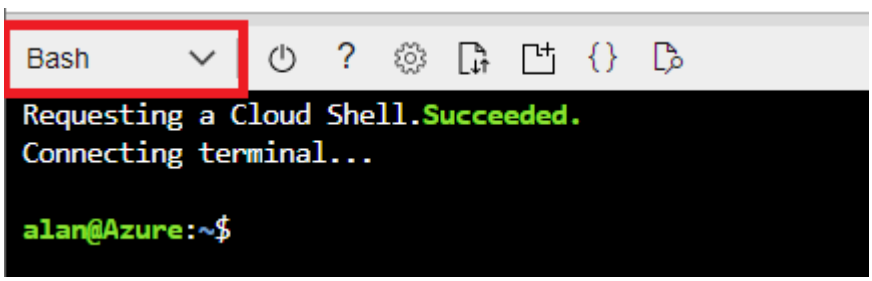
1. Open cloud with the below link

   https://shell.azure.com/

   If you've never used the Azure Cloud Shell before you will be prompted to mount a storage account, click **Create Storage** to continue. If you used Azure Cloud Shell before, you will skip this step.



2. Change to **Bash** access

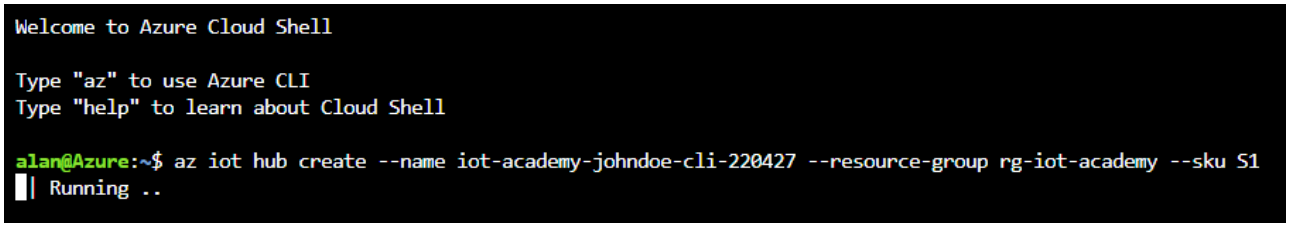


3. Once you are login run the following command to create an IoT Hub.

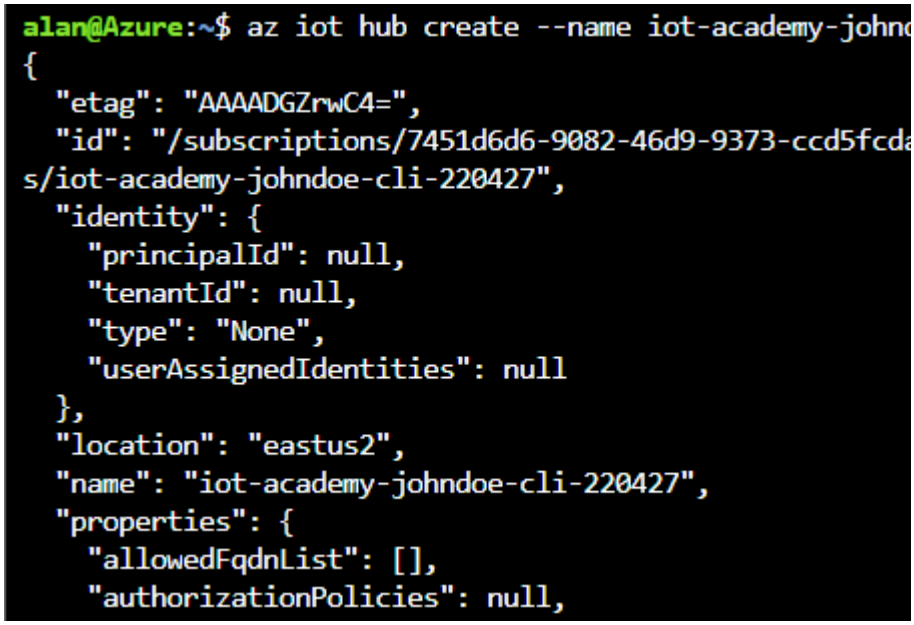
In the following command replace **iot-johndoe-cli-220427** with your iothub name, replacing johndoe and the appropriate date, of the form iot-{yourname}-cli-{YYMMDD}

```
az iot hub create --name iot-academy-johndoe-cli-220427 --resource-group rg-iot-academy --sku S1
```

As the command runs you'll observe the following result.



When the command completes you'll see output as follows



4. Browse to the Azure Portal to verify your newly created IoT Hub.

5. Delete the IoT Hub just created using the delete command.

   Again, replace johndoe and the appropriate date

   ```
   az iot hub delete --name iot-johndoe-cli-220427 --resource-group rg-iot-academy
   ```

   Also, delete the IoT Hub created through the portal

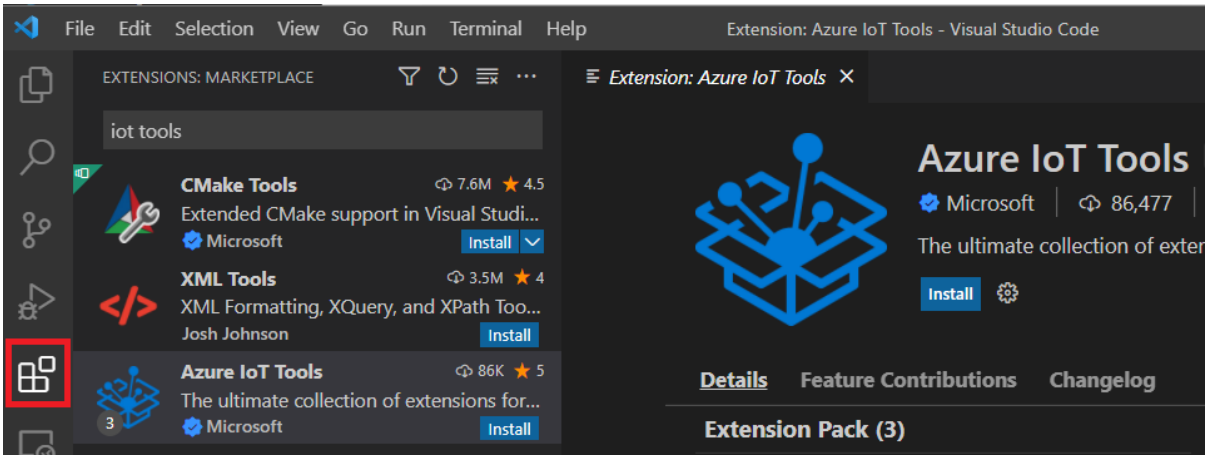
   ```
   az iot hub delete --name iot-academy-johndoe-220427 --resource-group rg-iot-academy
   ```

Task 3: Provision IoT Hub through VS Code

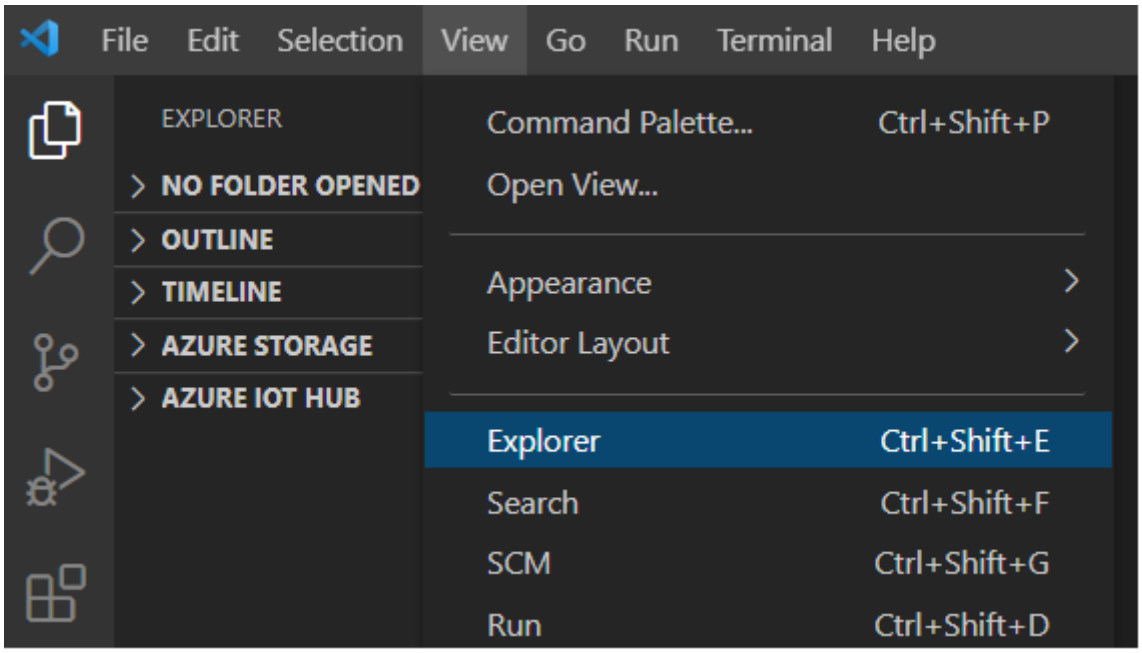
If you have not yet installed Visual Studio Code download and install from the following link: https://code.visualstudio.com/download

Our third way of creating an Azure resource, IoT Hub instance, is to use Visual Studio Code.

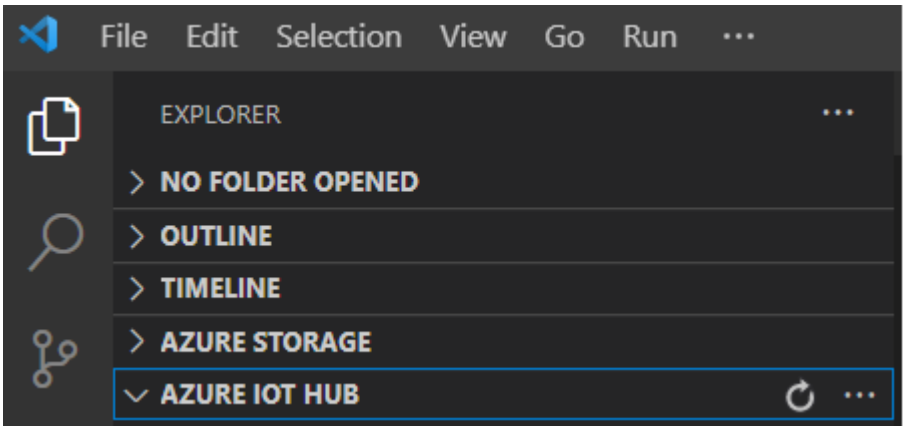
1. Install IoT Tools extension pack for VS Code in one of two ways:
    - Use the following URL https://marketplace.visualstudio.com/items?itemName=vsciot-vscode.azure-iot-tools
    - Use the extension tab (highlighted in red) in VS Code, search for **iot tools**, select **Azure IoT Tools**, click **Install**



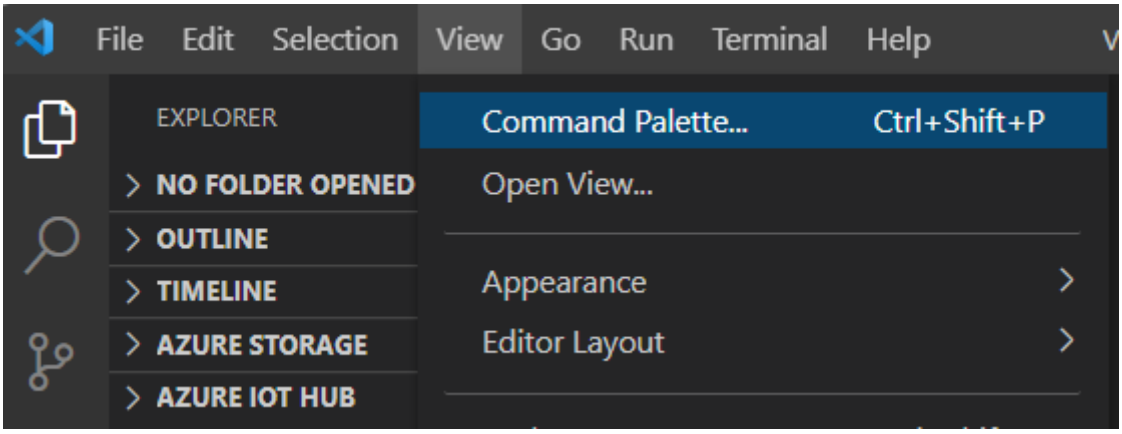
2. Click the View Menu and then Explorer



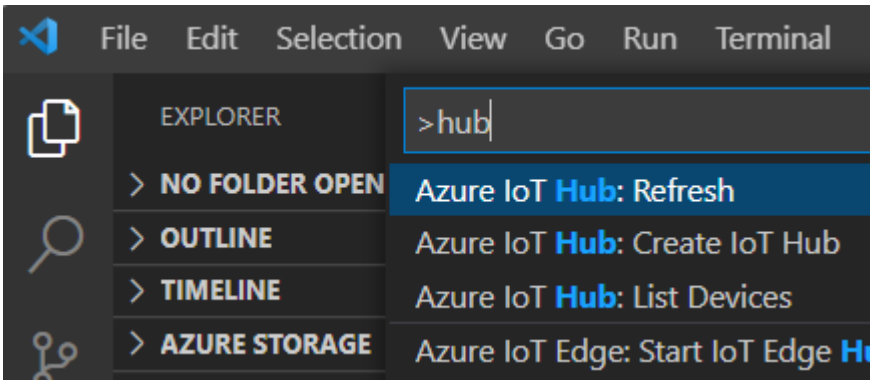
    - Now you should be able to see the **Azure IoT Hub**



- To create a new IoT Hub Go to the menu **View** on the top toolbar then select **Command Palette**.



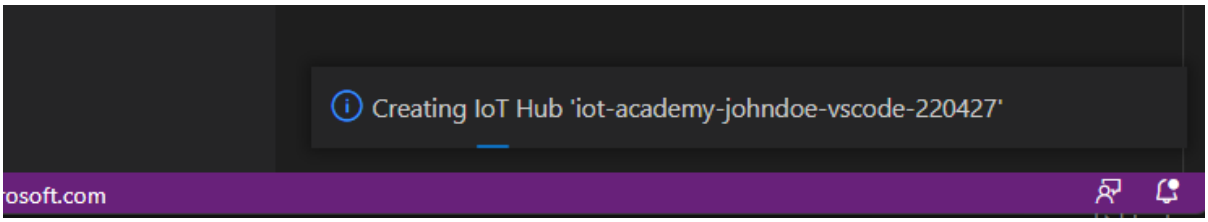
- Type **Azure IoT Hub** in the search bar, then you will see the list of commands available select **Azure IoT Hub: Create IoT Hub** and click Enter.



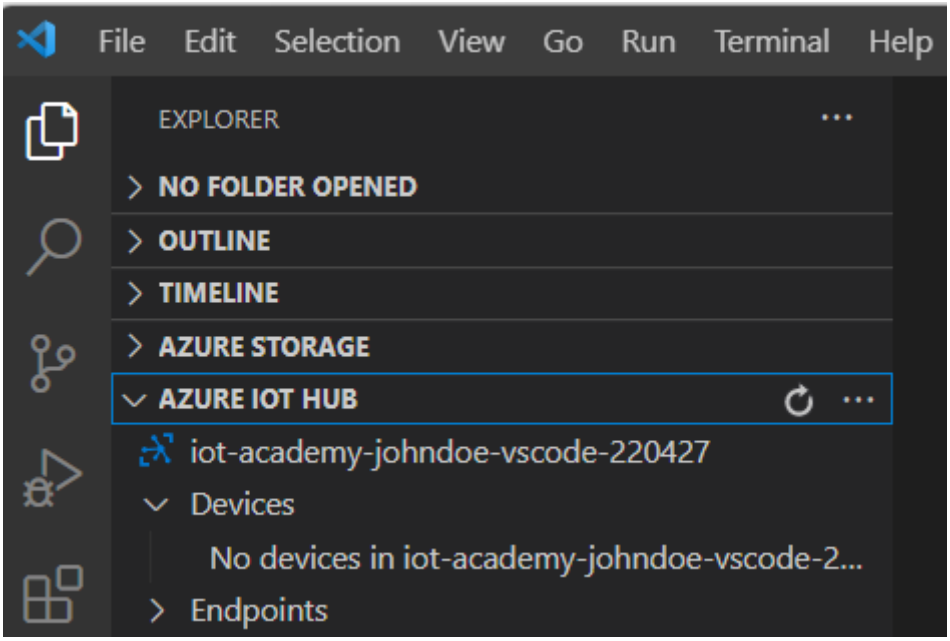
- Select the following parameters:

  - **Subscription**: Select your subscription.
  - **Resource group**: Use existing and select `rg-iot-academy`.
  - **Location**: Select the location you are using for resources in this hands-on lab.
  - **SKU**: Select **S1**.
  - **Name**: Assign a name to the IoTHub `iot-academy-johndoe-vscode-220427` change **johndoe** and **220427** using your name and the date YYMMDD.

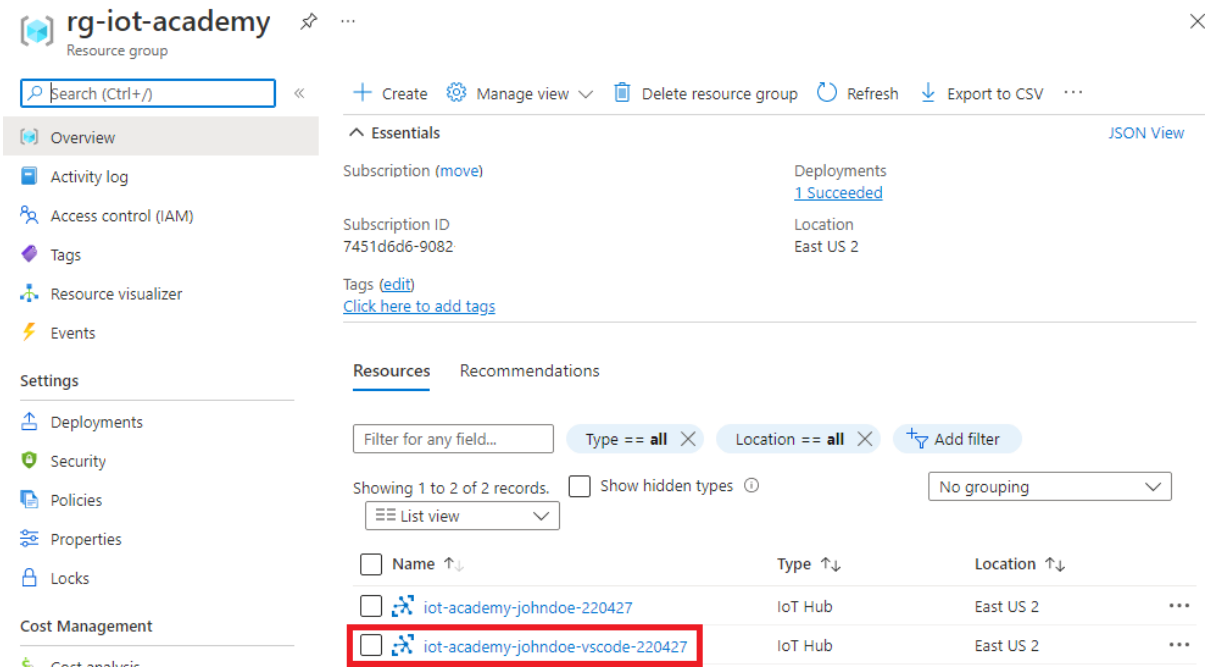
  As the extension creates the IoT Hub instance you should see a status message as follows.



- After the creation process you should be able to see the new IoT Hub in VS Code and the Azure Portal.



Azure Portal Resource Group View

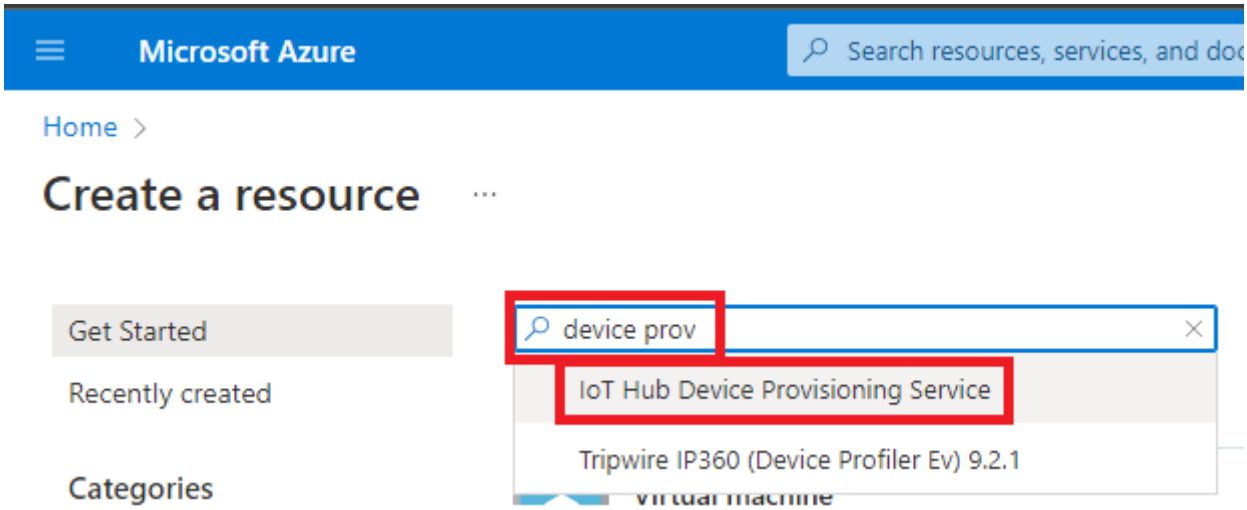


Exercise 2: Azure IoT Hub Device Provisioning Service (DPS)

Task 1: Deploy DPS

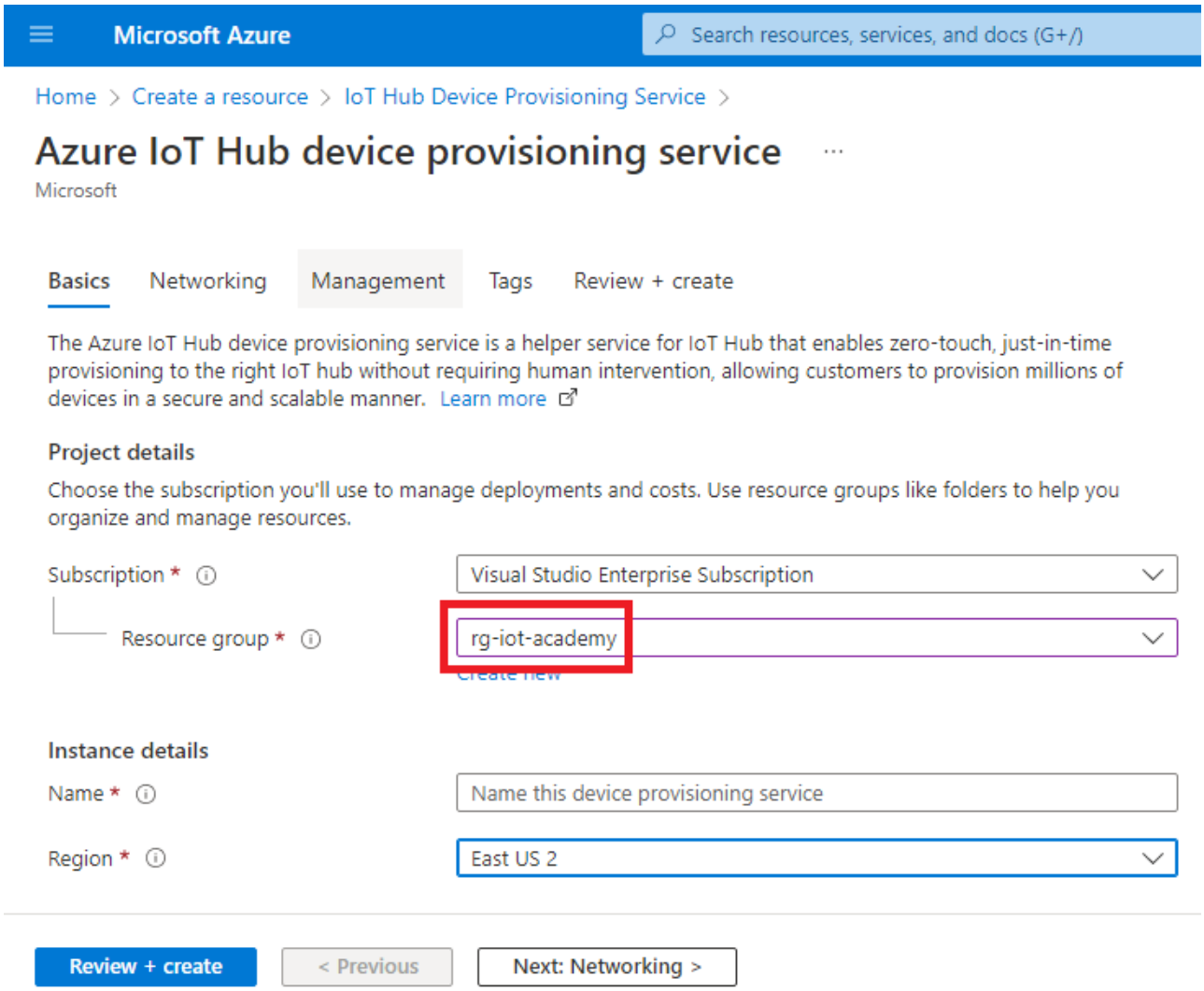
1. Create a DPS instance by:

- Go to the Azure Portal home page
- Click **Create a resource**
- Search for `device prov`
- Click **IoT Hub Device Provisioning Service**



2. Create Details

- Enter the following details
  - Resource group: `rg-iot-academy`
  - Name: `provs-iotacad-{SUFFIX}` e.g. `provs-iotacad-johndoe220427`
  - Region: `your region`



3. Click **Review and Create**, then click **Create**
4. When the deployment completes, click **Go to resource**
5. When the Overview page loads save the **ID Scope** to notepad

Task 2: Connect IoT Hub to DPS

1. Click **Linked IoT Hubs**, then click **Add**
2. Fill in the following details, then click **Save**
3. Click **Manage Allocation Policy**, review the options available

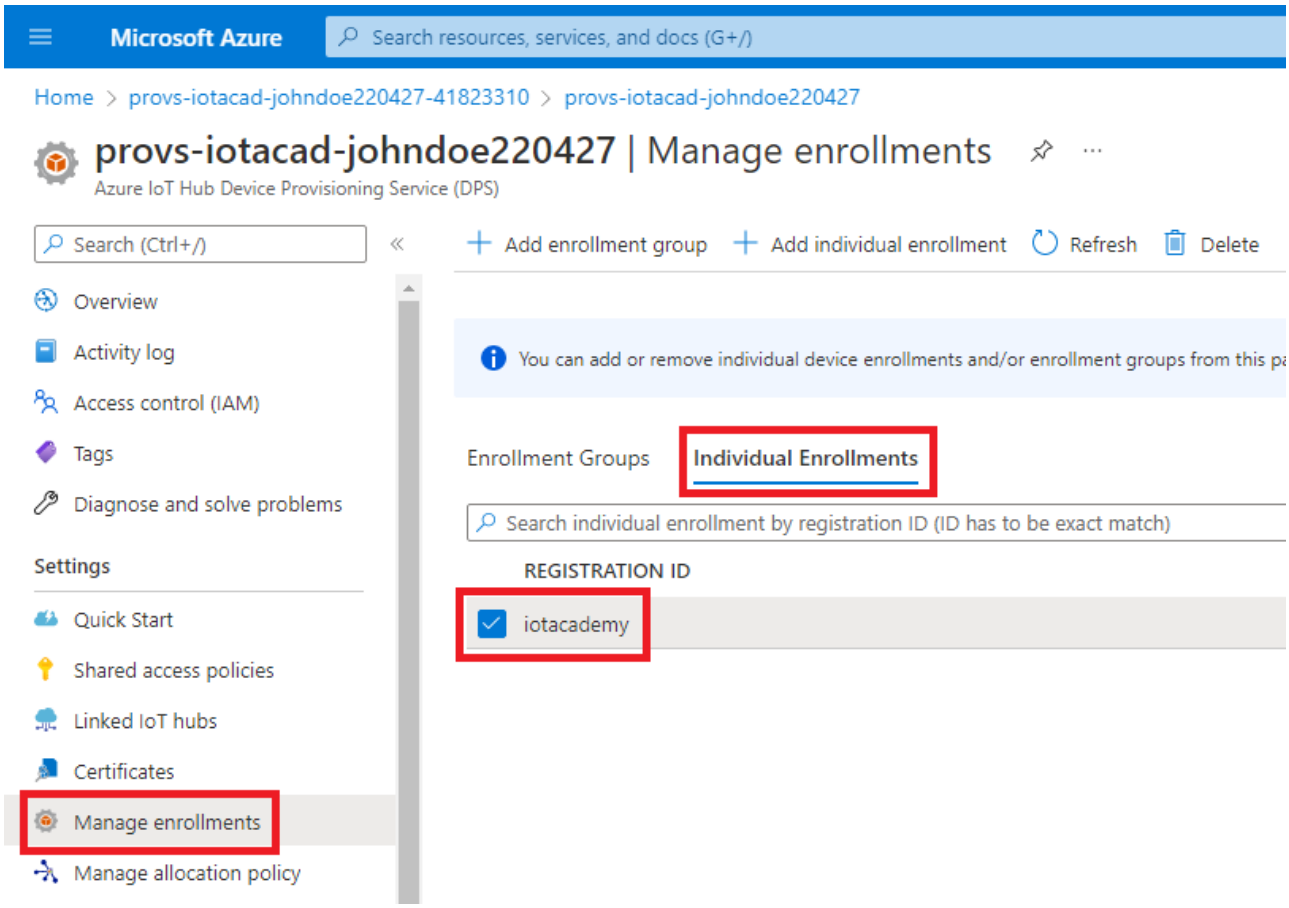
Task 3: Create an Individual Enrollment

1. Click **Add individual enrollment**
2. Fill in the following details
   - Mechanism: `Symmetric Key`
   - Auto-generate keys: `checked`
   - Registration ID: `iotacademy`
   - IoT Hub Device ID: `iot-academy-edge-device`
   - IoT Edge device: `True`
   - Select the IoT hubs this device can be assigned to: `select your hub` Leave all other values at default

3. Click **Save** at the top of the page

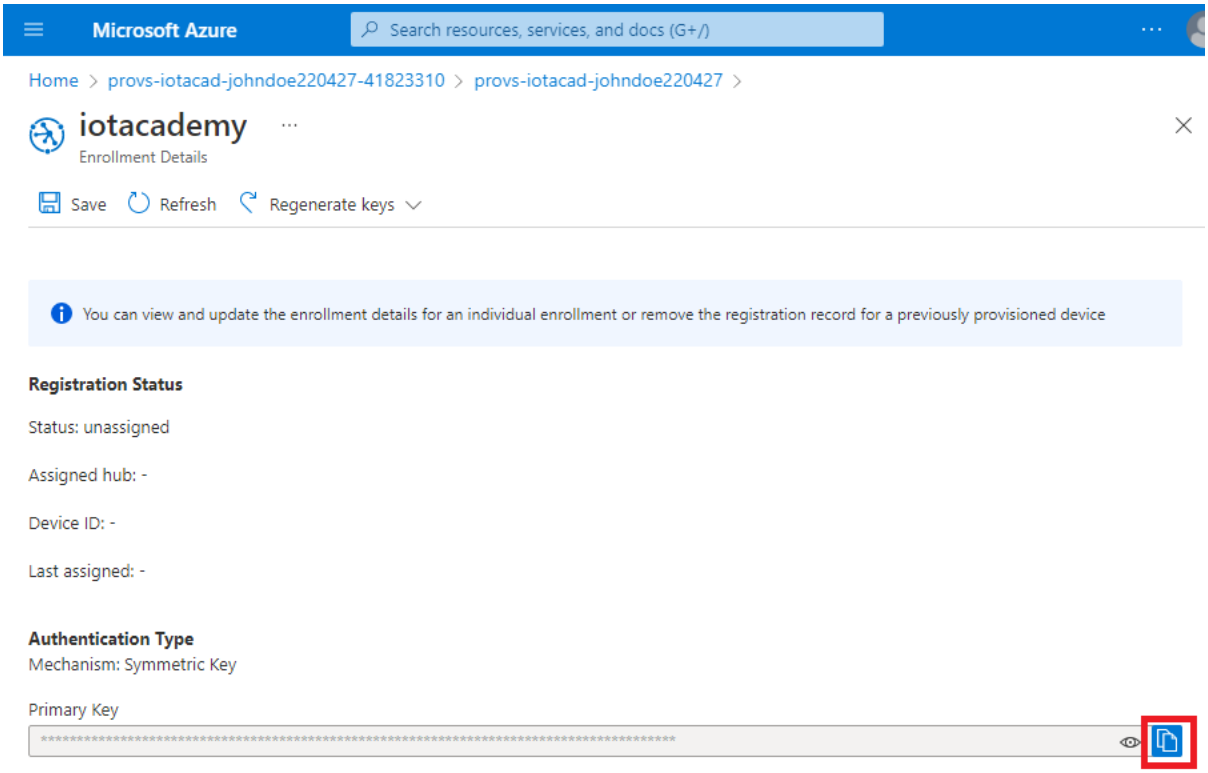
Task 4: Gather Individual Enrollment Details

1. Click **Manage enrollments**, then click **Individual Enrollments**, then click your enrollment **iotacademy**



2. Take note of the following values in your notepad
    - Registration ID
    - IoT Hub Device ID

- Primary Key (click the copy icon shown in the image below)



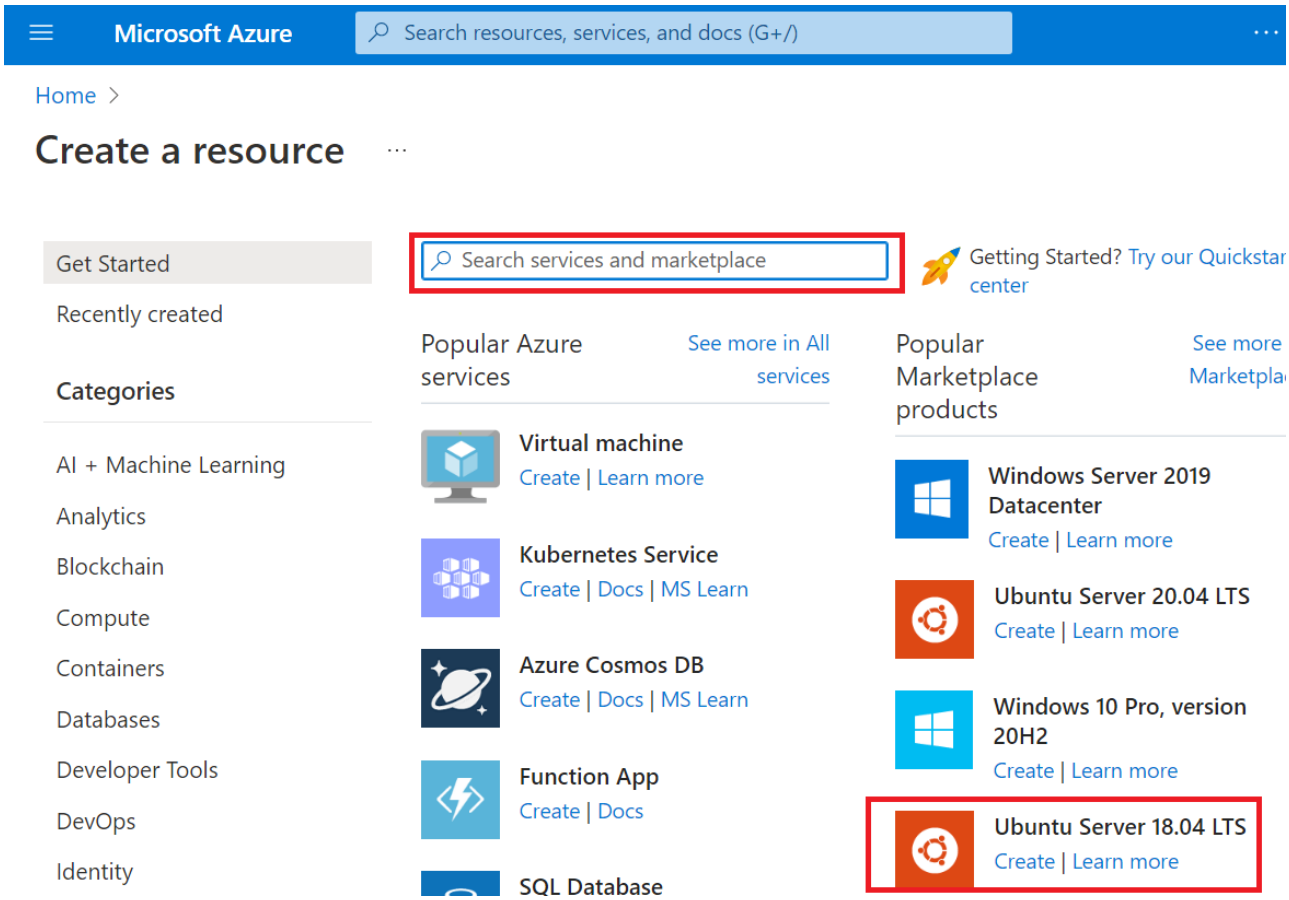
Exercise 3: Create an Ubuntu-based Azure IoT Edge Device

During this exercise you will learn how to set up an Azure IoT Edge device and connect it to IoT Hub to start streaming data.

Task 1: Creating a VM to host an IoT Edge Device

In this exercise we'll set up an IoT Edge device using an Ubuntu based VM.

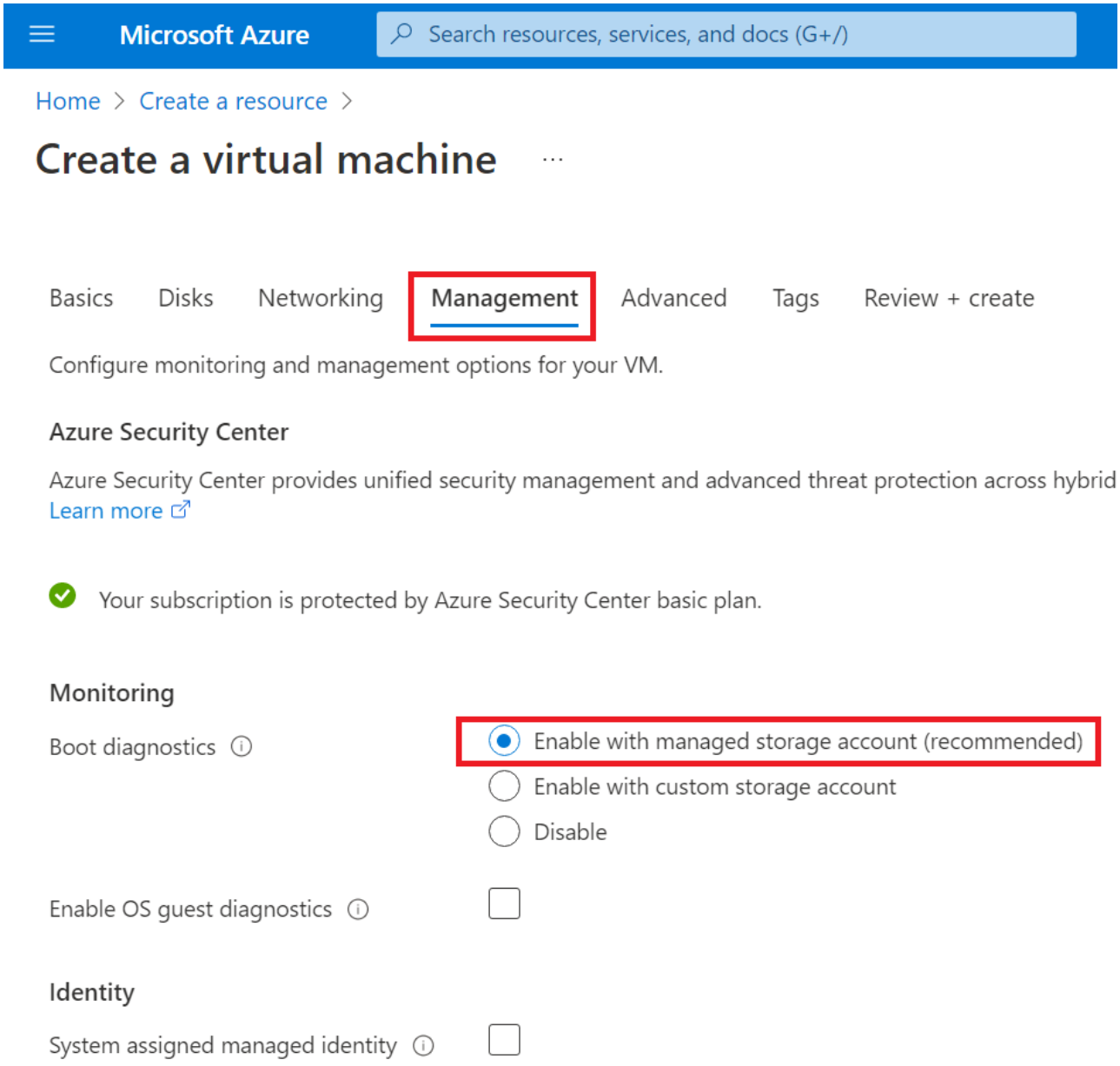
1. From Azure Portal select **Create resource** then from the most Popular list select **Ubuntu Server 18.04 LTS**. If you don't see it use the search box titled **Search services and marketplace** to search for **Ubuntu Server 18.04 LTS**.



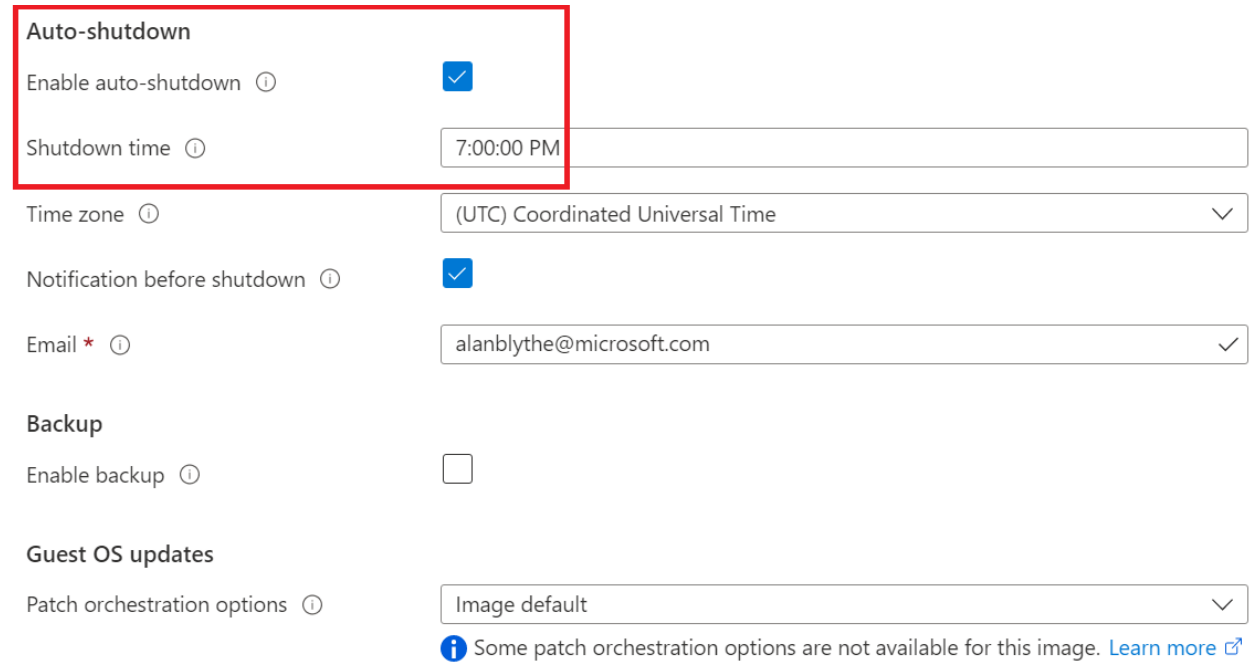
2. Then you will need to complete the following parameters in the **Basics** tab:

   - **Subscription**: Select the subscription you are using for this hands-on lab.
   - **Resource group**: Use existing and select your resource group, `rg-iot-academy`.
   - **Virtual Machine Name**: edgedevice+SUFFIX e.g. `edgedevice-johndoe-220427`
   - **Region**: Select the location you are using for resources in this hands-on lab.
   - **Availability Options**: Select **No Infrastructure redundancy required**.
   - **Image**: Keep default
   - **Size**: Keep default
   - **Authentication Type**: Select **Password**
   - **Username**: iotacademy
   - **Password**: MSFTacademy01!
   - **Public inbound ports**: None

3. Click the **Management** tab at the top of the pane.



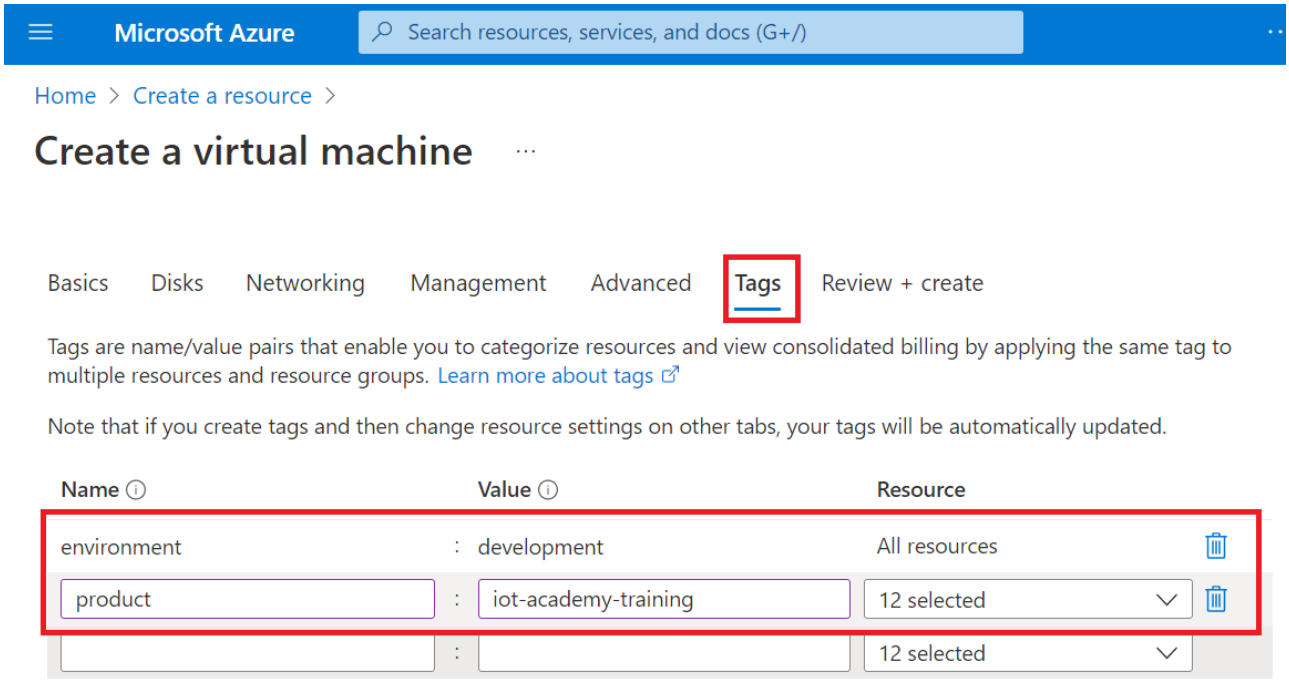
Notice the Auto-shutdown feature. This feature is a helpful to control costs for development or infrequently used virtual machines. When the VM is shutdown you do not incur compute costs.



4. Click the Tags tab Add the following two tags

- environment: development
- product: iot-academy-training

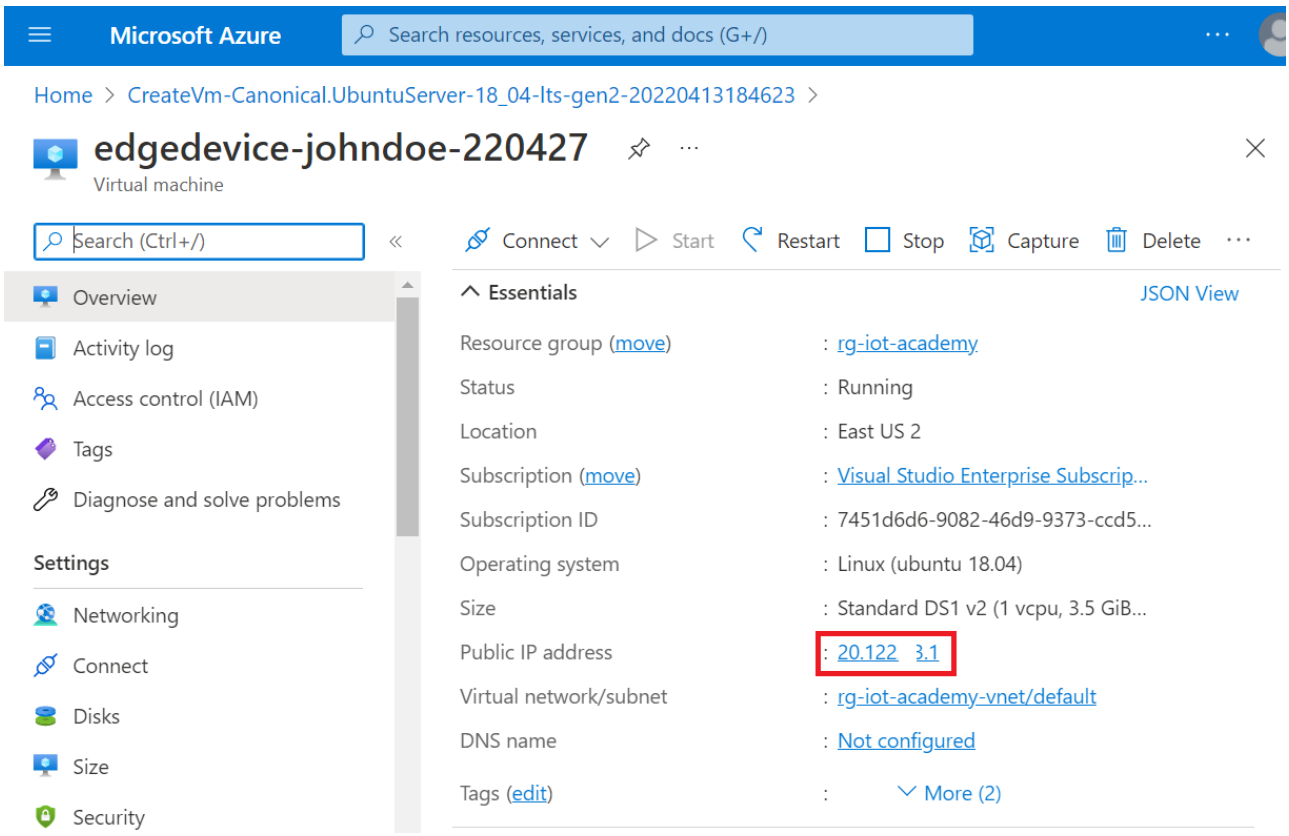
Diagram showing the tags tab while creating a VM



Tags are an important aspect for management, governance, and hygiene of Azure resources. It's not uncommon to have thousands of resources in mature organizations. The tags can be used for search,

reporting, and automation tools to ease management of large deployments. You can read more at Azure Tagging Strategy

5. Last select **Review + Create** after successfull validation you should be able to click **Create**
6. Once the resource is available click **Go to resource** to view the newly created Virtual Machine. You should see in the Overview section the Public IP to connect, copy the IP to your notepad.



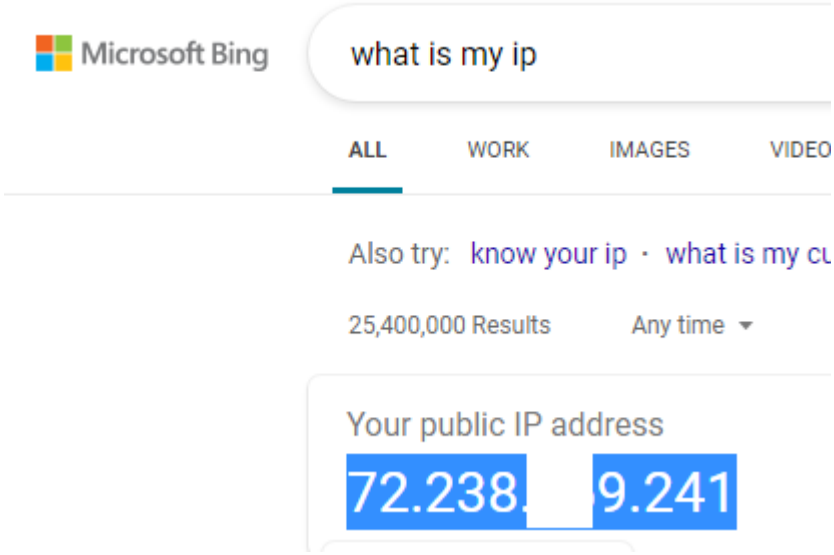
Task 2: Connecting to your Ubuntu Virtual Machine

An important aspect of building cloud infrastructure is doing it in a secure manner. As part of this exercise port 22 could be opened, for SSH, to allow quick connection to the VM. However, this could allow an attacker to attempt to breach this port. Two safer approaches could be used

- A safer approach would be to use an important feature of Azure Virtual Machines, Just-in-time (JIT) VM access. This feature allows enabling access to the VM for a specified amount of time. More information can found at Just-in-time VM Access
- Enabling SSH, port 22, access for just your IP address

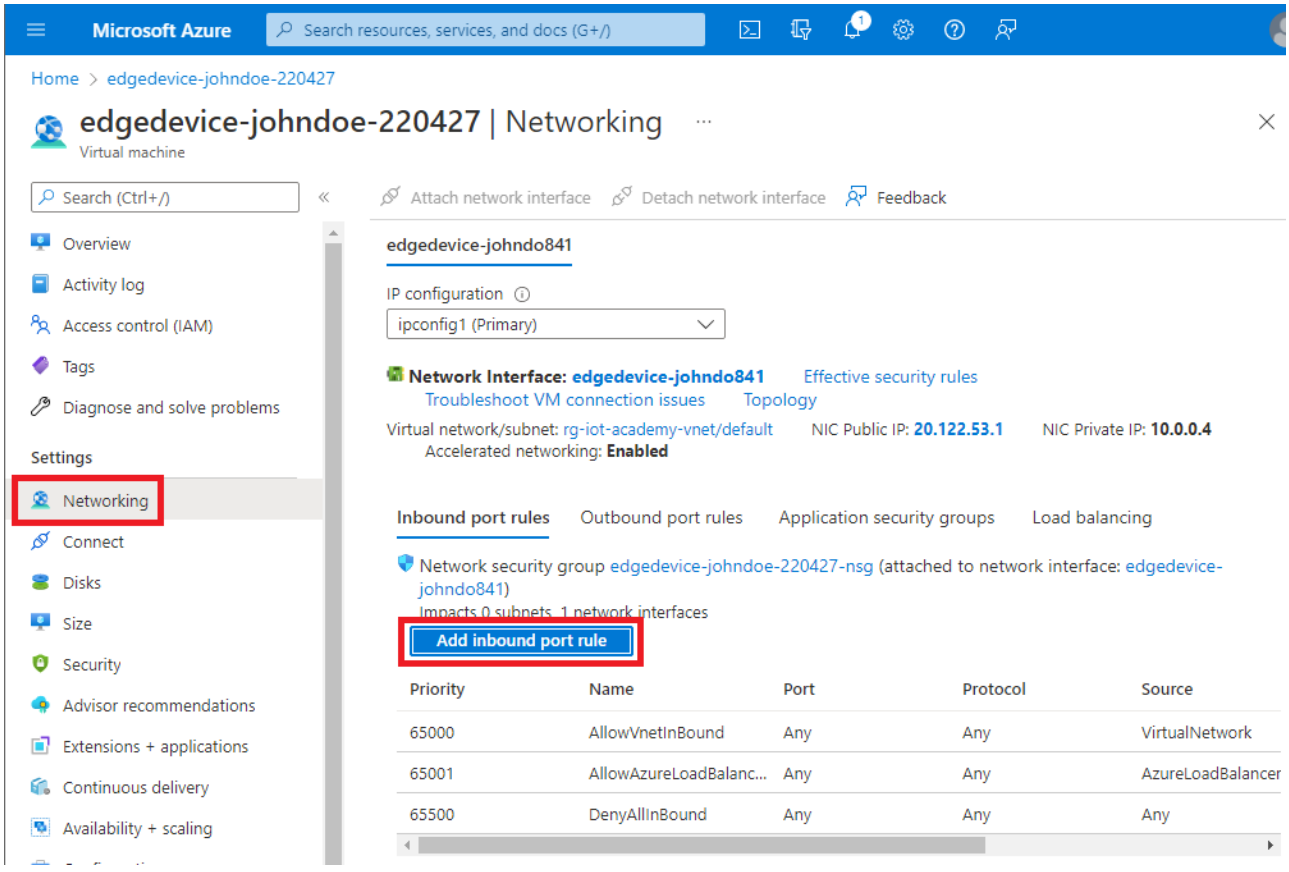
For simplicity the 2nd option will be used.

1. Go to www.bing.com in your browser and search for `what is my ip`



Copy your IP address to your notepad.

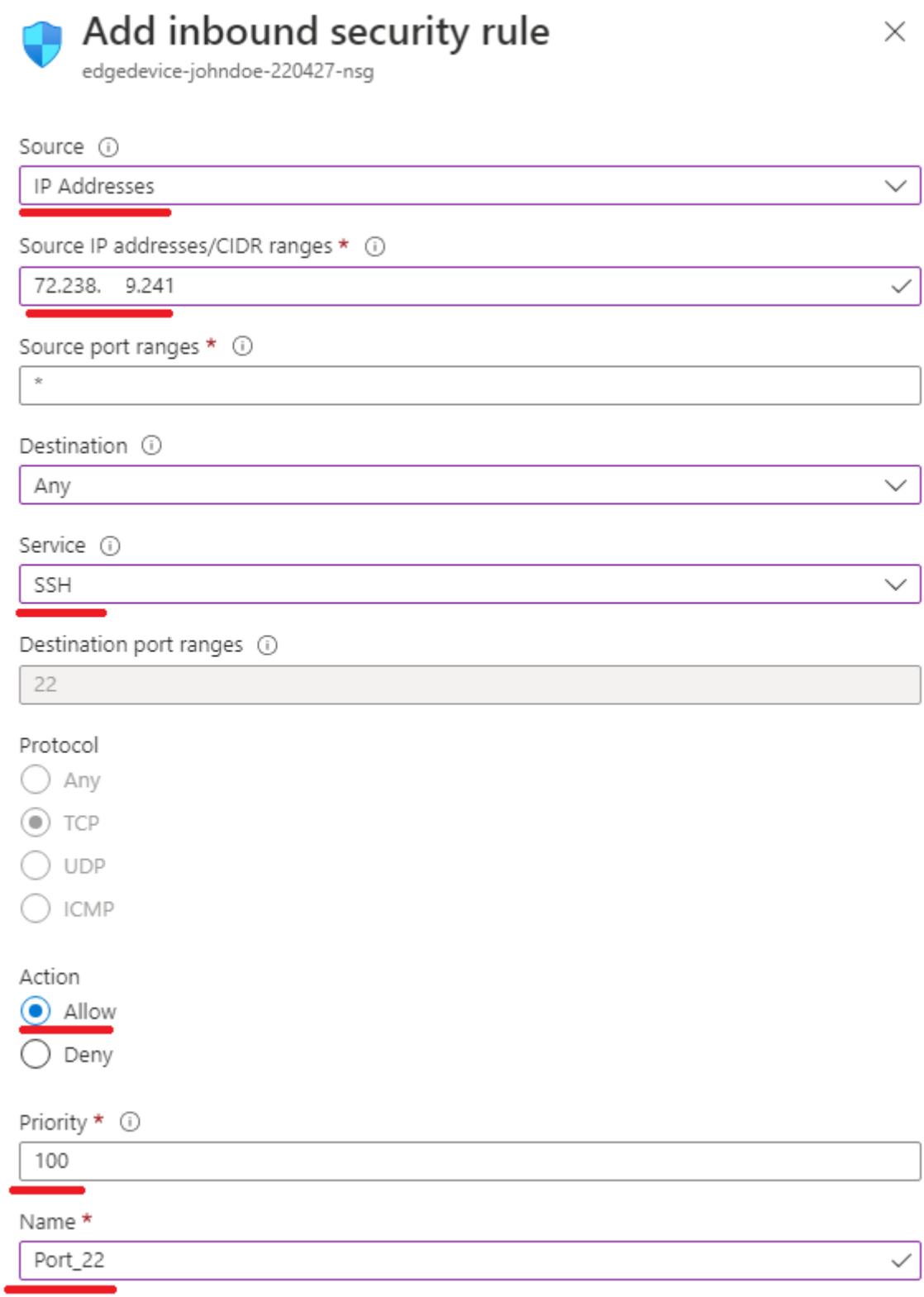
2. On your VM resource click the **Networking** tab. Click the `Add inbound port rule`



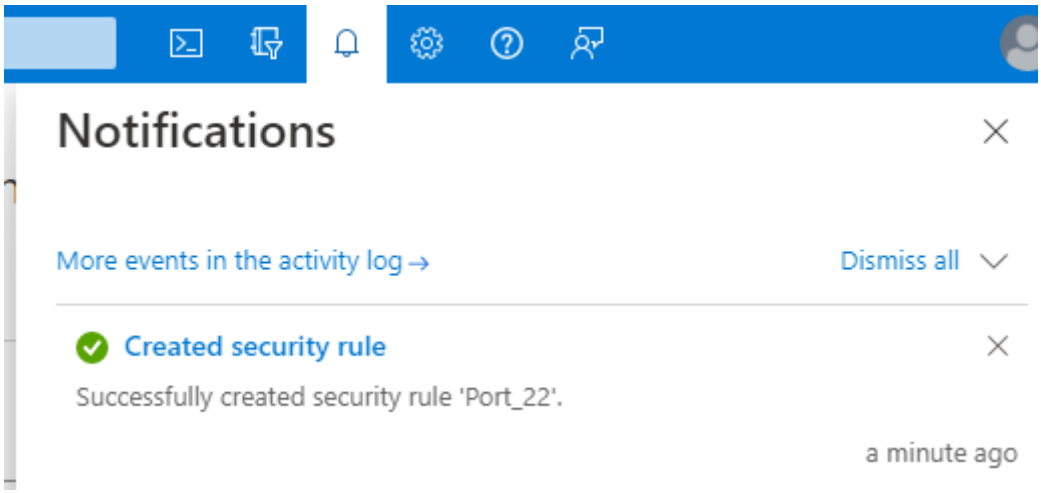
3. Set the details for the new inbound port rule, click Add.

   - Source: `IP Addresses`
   - Source IP addresses/CIDR ranges: `Your IP address you saved in notepad`
   - Source port ranges: `*`
   - Service: `SSH`
   - Action: `Allow`
   - Priority: `100`

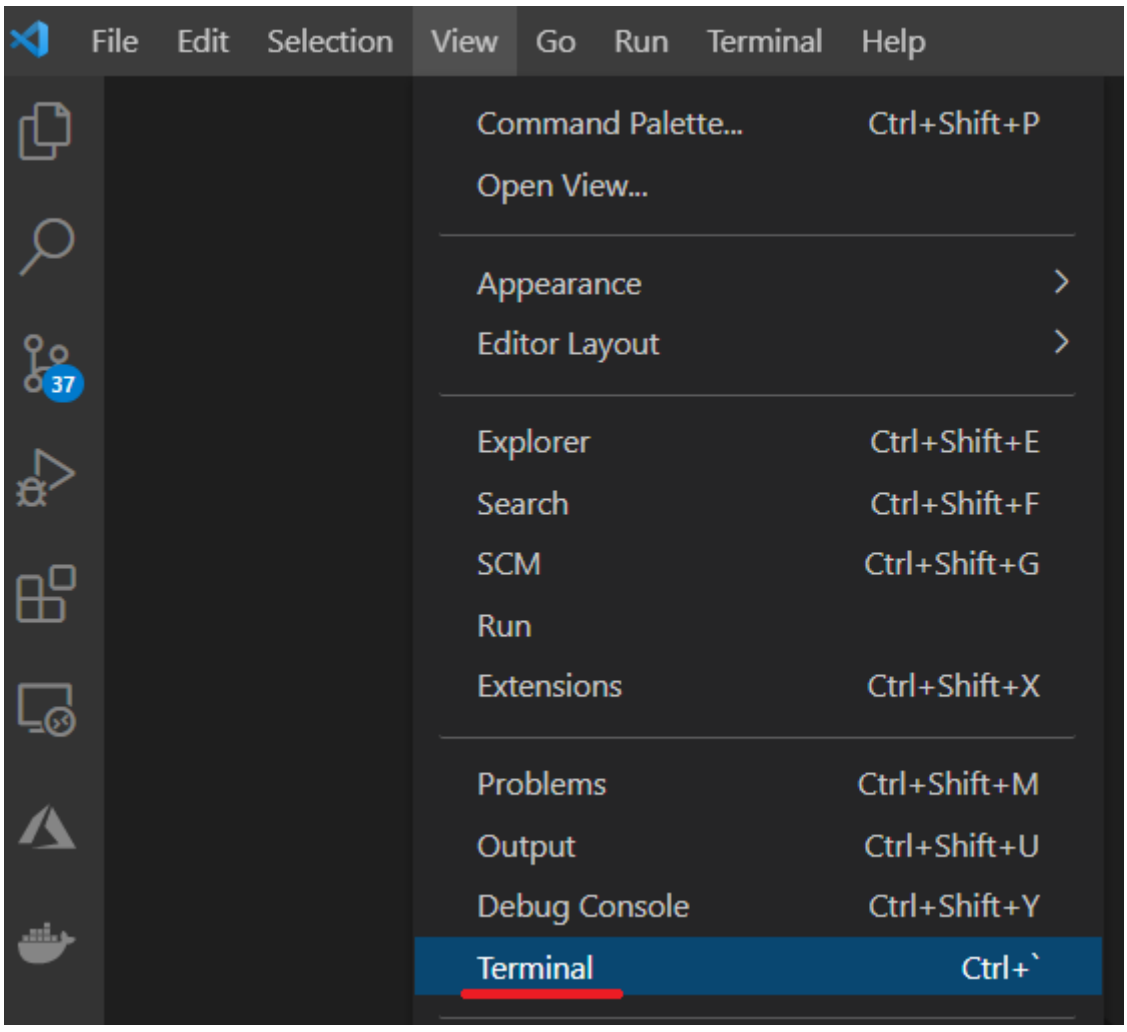
- Name: `Port_22`

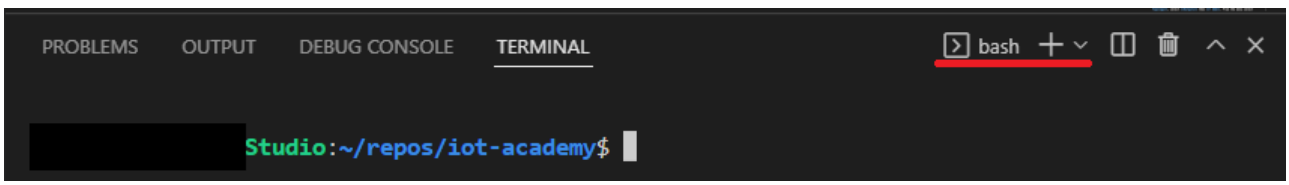


4. Watch for the notification that the security rule created successfully



5. Switch to VS Code, use the **View** menu and click **Terminal**

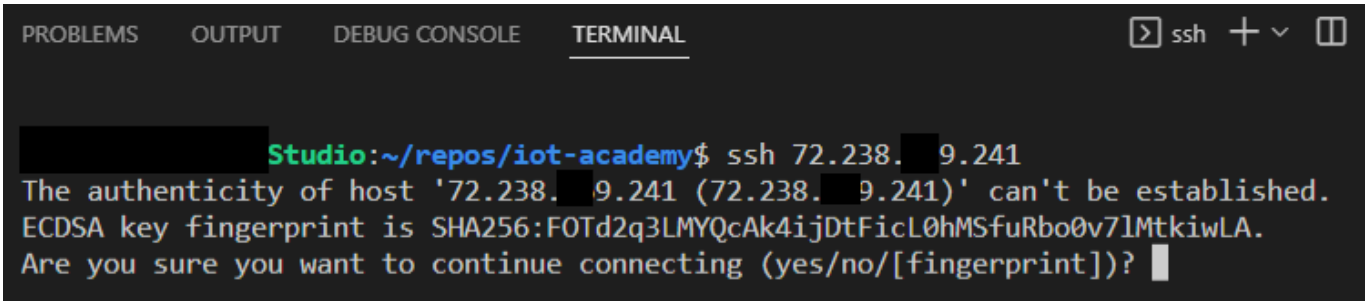




If you don't see a `bash` terminal at the top, click the `+` and click `bash`

6. Enter `ssh iotacademy@{the public IP address of your VM}` and press enter. You saved the VMs public IP address earlier in your notepad. Be sure to not confuse your public IP address with the IP address of the VM. e.g. `ssh iotacademy@20.122.53.2`

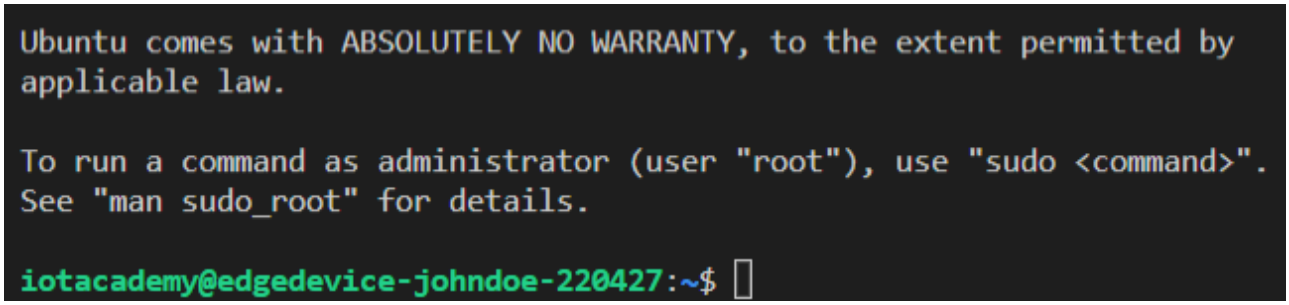
If this is your first time connecting you'll see a prompt asking `Are you sure you want to continue connecting? `. Enter `yes` and press enter.



7. Enter your password you defined earlier `MSFTacademy01!` and press enter. The password can be copied to your clipboard and pasted into the terminal by:

    - left clicking on the terminal window once, to focus the window
    - right clicking on the terminal window once, this pastes the clipboard contents

    After successfully connecting you'll see the following in your terminal



Task 3: Install the Azure IoT Edge Runtime and Connect the Device

1. Now logged into the VM, Install the Edge Runtime

    Install the repository configuration that matches your device operating system.

    ```
    curl https://packages.microsoft.com/config/ubuntu/18.04/multiarch/prod.list
    > ./microsoft-prod.list
    ```

    Copy the generated list to the sources.list.d directory.

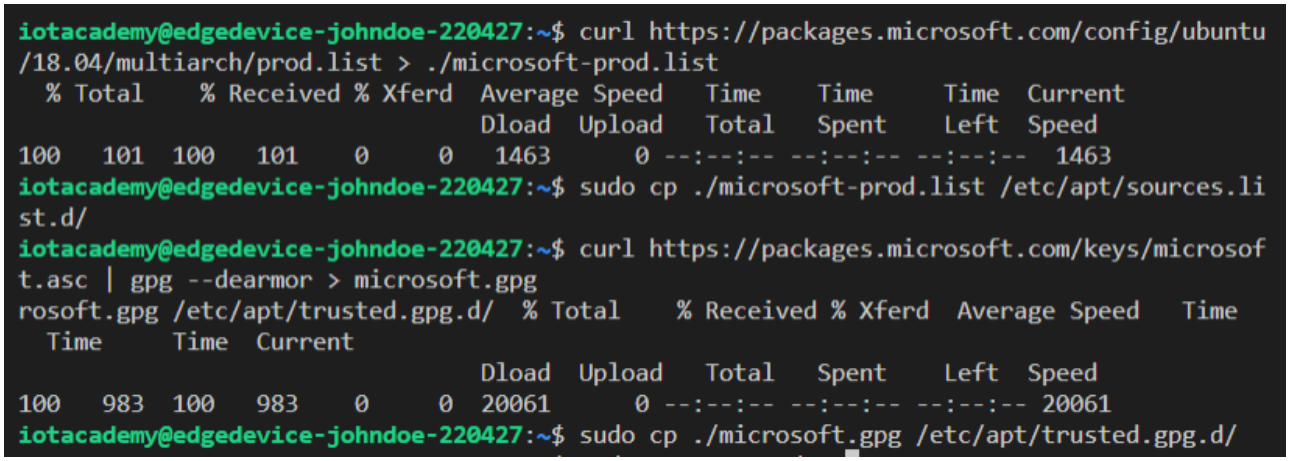
    ```
    sudo cp ./microsoft-prod.list /etc/apt/sources.list.d/
    ```

Install the Microsoft GPG public key.

```
curl https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor >
microsoft.gpg
sudo cp ./microsoft.gpg /etc/apt/trusted.gpg.d/
```

Azure IoT Edge software packages are subject to the license terms located in each package (usr/share/doc/{package-name} or the LICENSE directory). Read the license terms prior to using a package. Your installation and use of a package constitutes your acceptance of these terms. If you do not agree with the license terms, do not use that package.

After successfully running the previous commands you'll the following results depicted in the diagram.



2. Install a Container Engine

   Update package lists on your device.

   ```
   sudo apt-get update
   ```

   Install the Moby engine.

   ```
   sudo apt-get install moby-engine
   ```

3. Install the IoT Edge runtime package

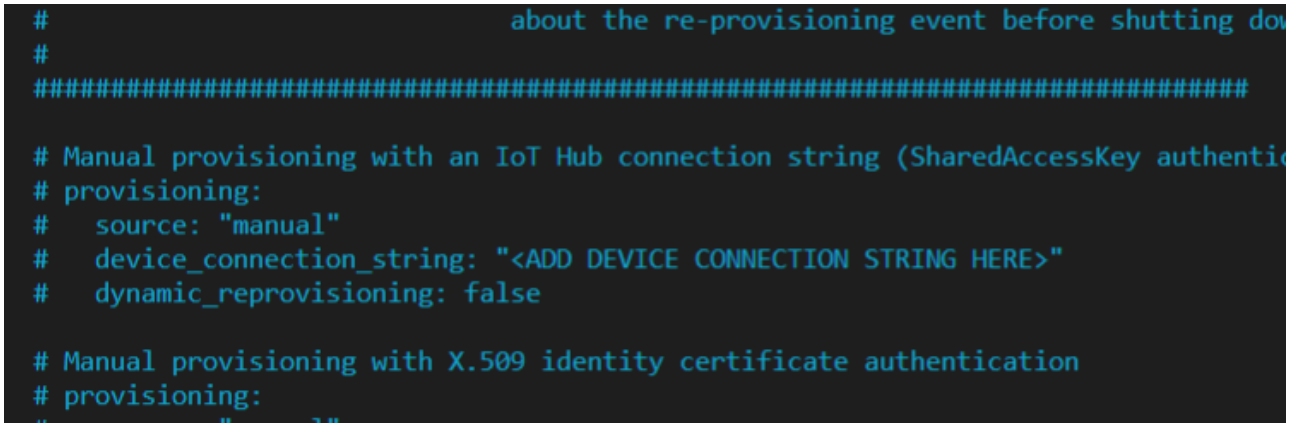
   ```
   sudo apt-get install iotedge
   ```

4. Edit the IoT Edge config.yaml, updating the provisioning information

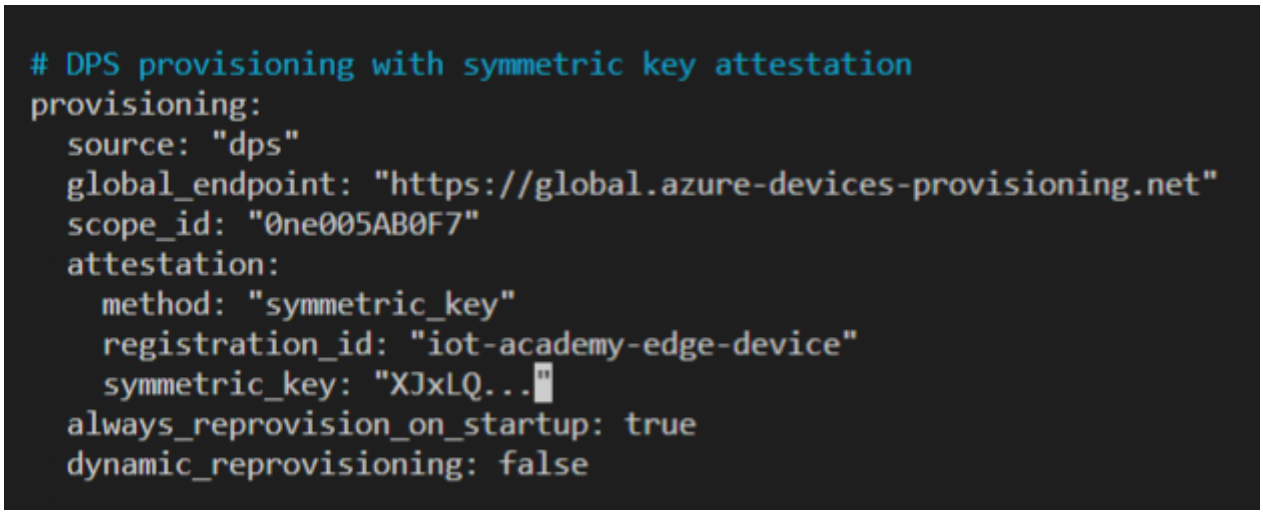
   - Configure the provisioning section with the information that we saved during the setup of DPS.

```
sudo nano /etc/iotedge/config.yaml
```

- Scroll down to **Manual Provisioning with an IoT Hub connection string** then comment out all the uncommented lines using the # symbol.



- Scroll down further to locate the **DPS provisioning with symmetric key attestation** section. Uncomment the section and set the following values
  - scope_id: saved in notepad
  - registration_id: saved in notepad
  - symmetric_key: the primary key saved in notepad



After making the edits:

- press **CrtL+X** to close the file
- select **Y** to save the changes
- press enter to accept the file name

Now restart your edge daemon

```
sudo systemctl restart iotedge
```

In a few minutes you should receive a **Running** status after executing the following command. You may need to run the command several times if enough time has not passed since restarting the IoT Edge runtime.

```
sudo iotedge list
```

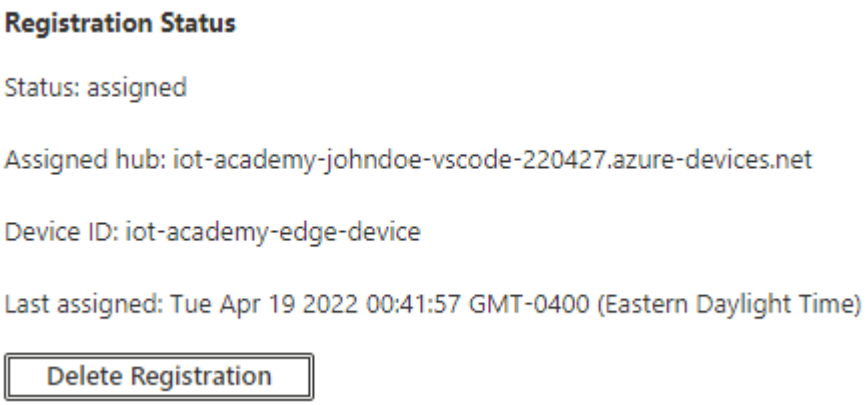
Another command that is useful is **check** that is shown below. The **check** command:

- checks the validity of the config.yaml file
- the container engine is functional
- time is correct
- connections to DPS and IoT Hub can be established

```
sudo iotedge check
```

Task 4: Observe the Enrollment and Device Status

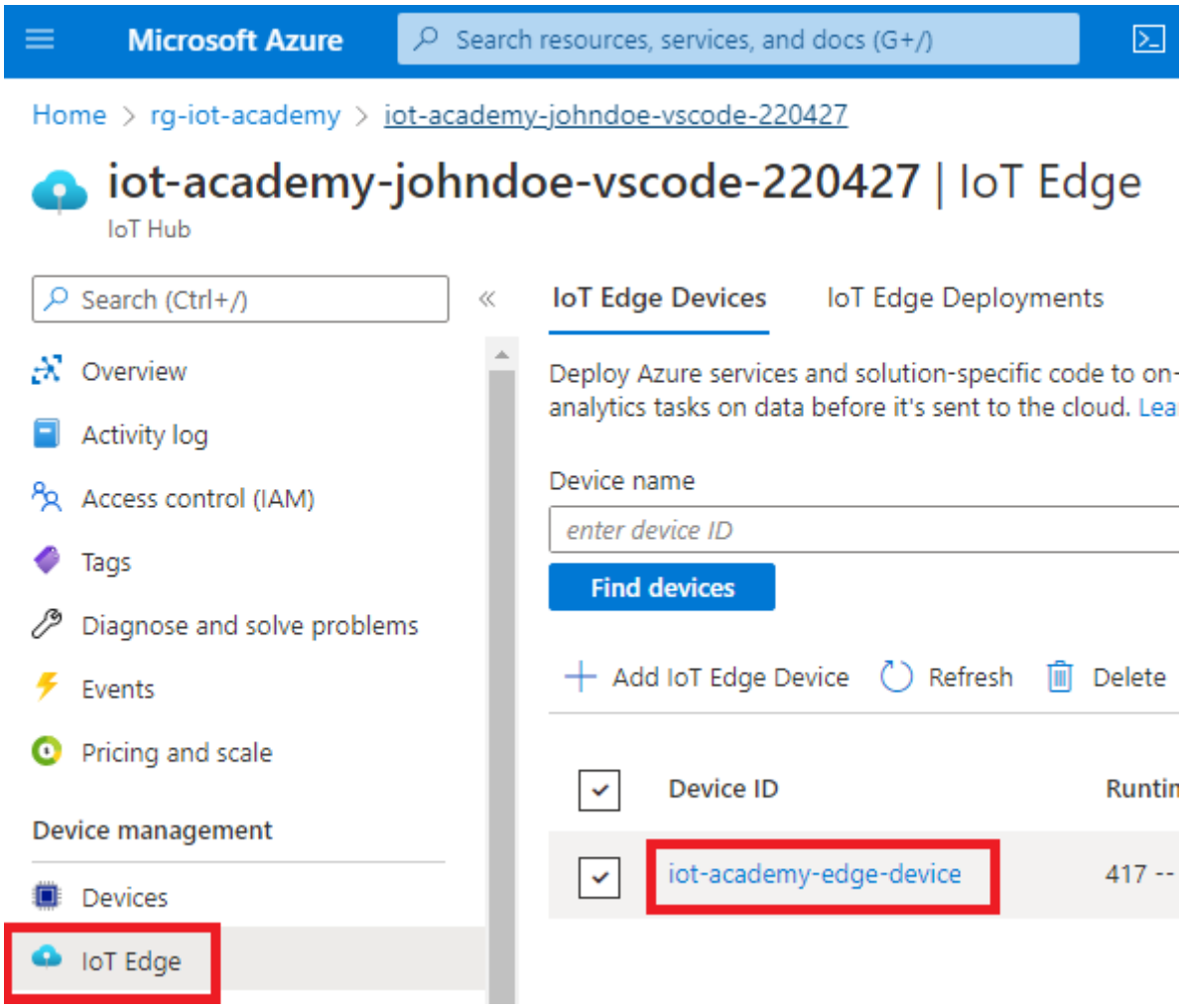
1. Navigate to your DPS instance by searching for `provs` and clicking your instance.
2. Click **Manage Enrollments**
3. Click **Individual Enrollments**
4. Click **iotacademy**



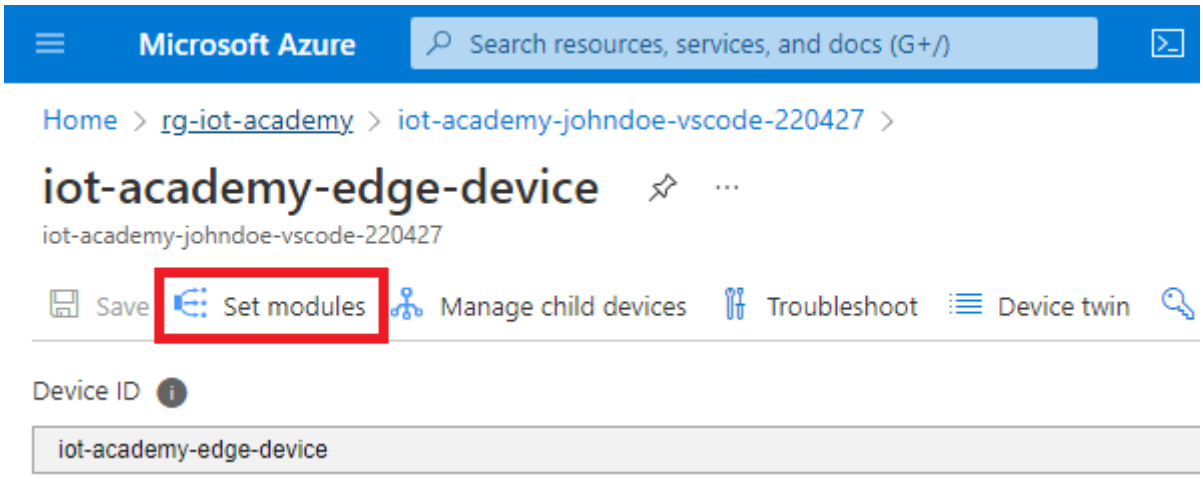
Exercise 4: Deploy and IoT Edge Module to Simulate Device Telemetry

Task 1: Use the IoT Edge Module Marketplace to Provision the Simulated Temperature Sensor Module

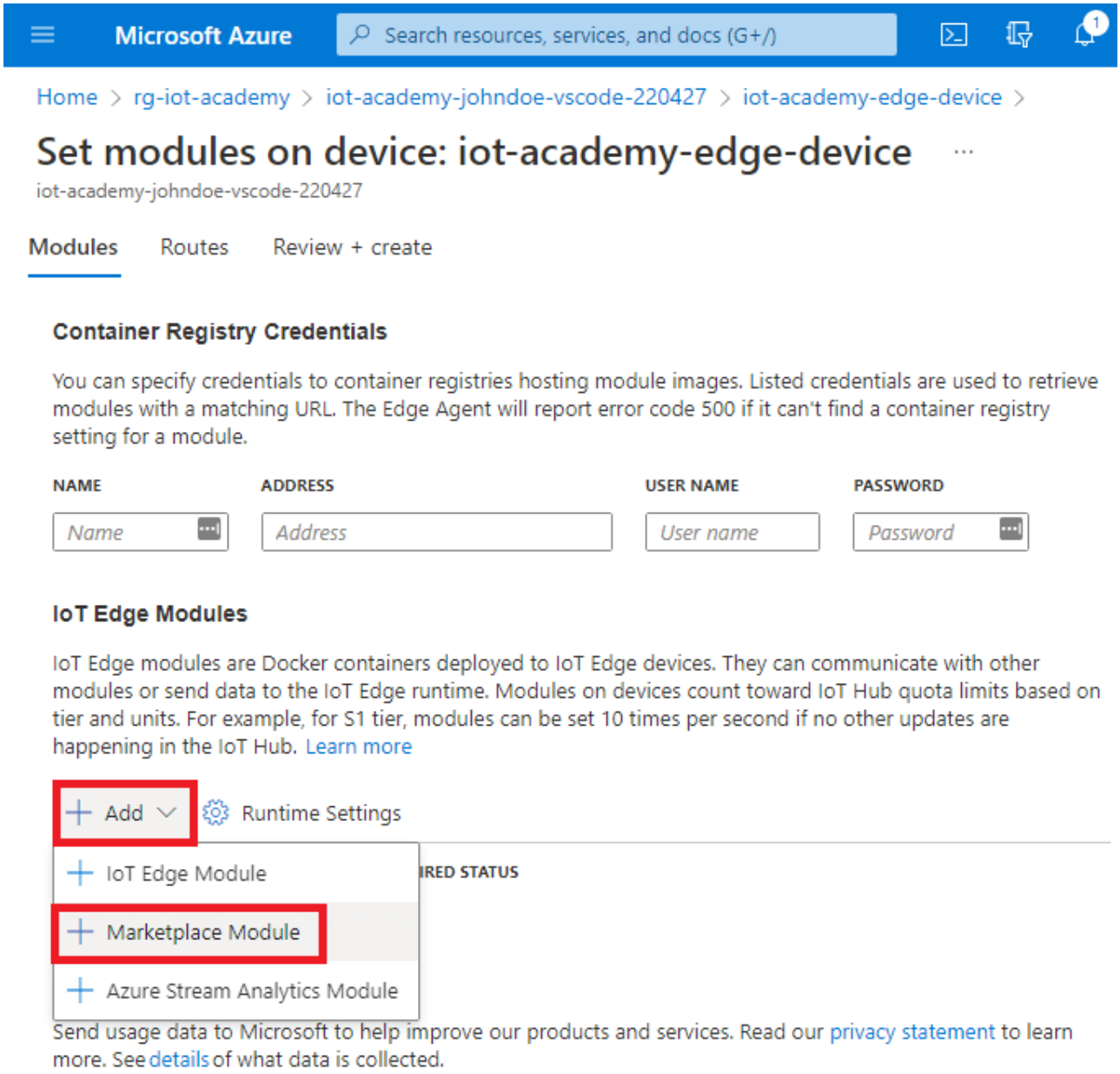
1. In Azure Portal, navigate to your IoT Hub created in previous steps, under **Device Management** click **IoT Edge**, then click your Edge Device



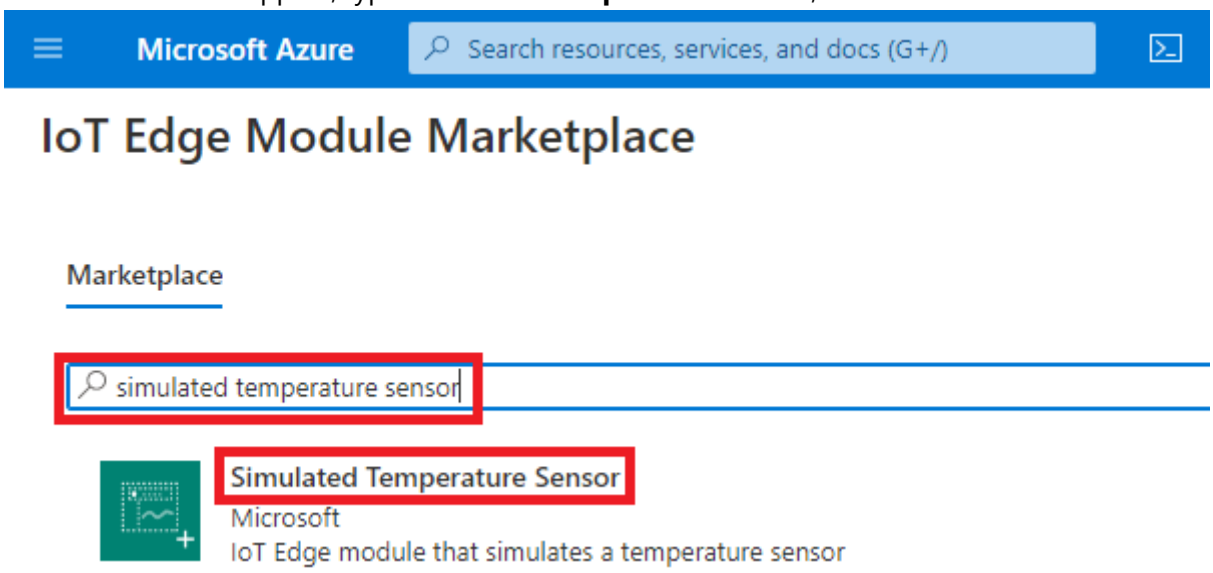
2. Click **Set Modules**



3. Click **Add**. Then click **Marketplace Module**



4. The search box will appear, type **Simulated Temperature Sensor**, Then click the module to add it



5. Review the presentation section **Routes** to learn about IoT Edge routing

6. Click **Review + create** at the bottom of the pane. Review the available settings for the Deployment

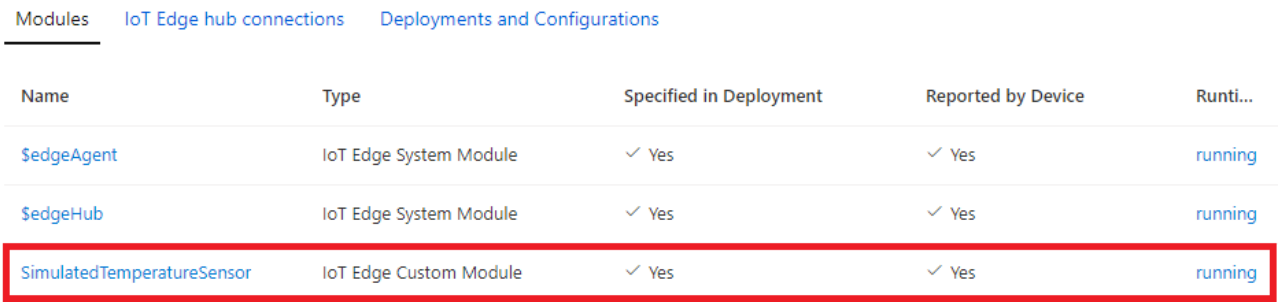
7. After validation passes click **Create** at the bottom of the pane

Task 2: Ensure the Module is Running

Review the running modules on your Edge device

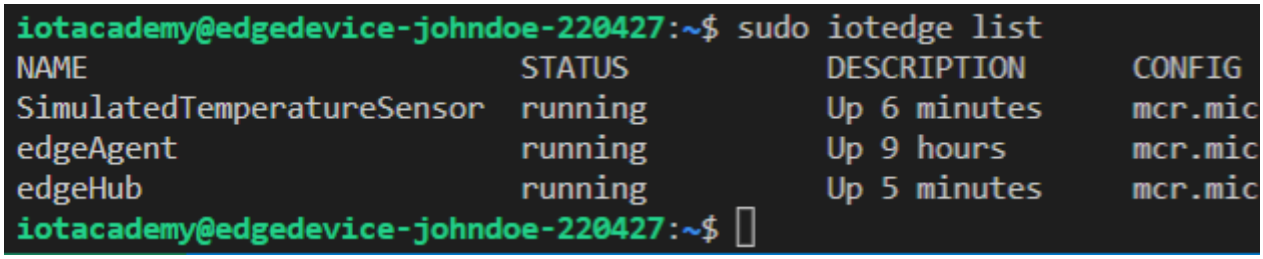
It may take a few minutes for the module to deploy and come to a running state. Running modules can be verified in a few ways:

- On device details review the **Modules** section at the bottom of the pane.



- Using SSH, connect to your VM and run the `iotedge list` command as shown below

```
sudo iotedge list
```

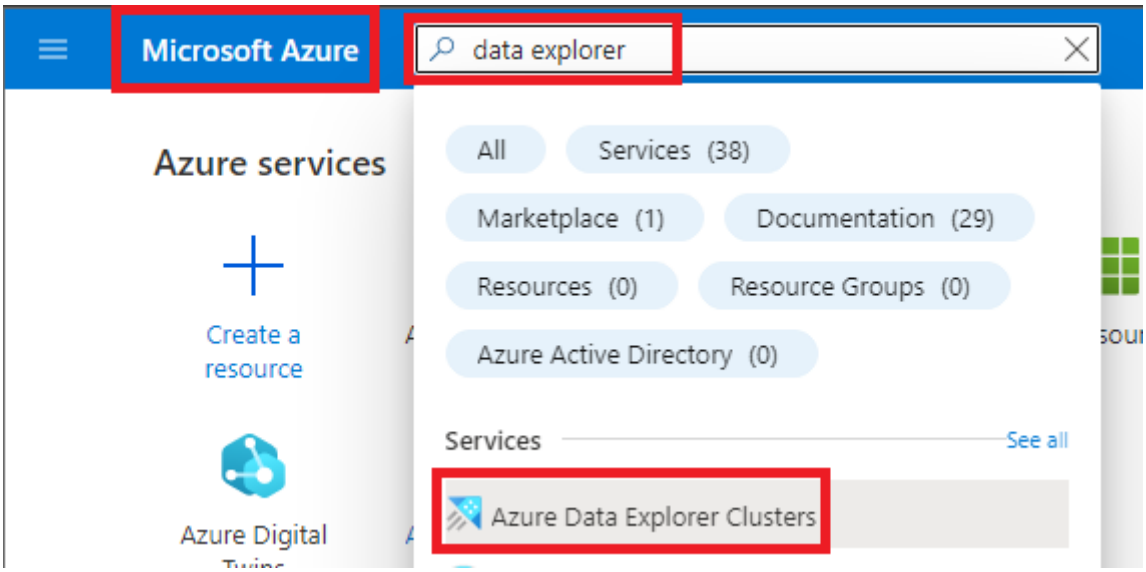


Exercise 5: Ingesting Telemetry Data with IoT Hub and Azure Data Explorer (ADX)

Task 1: Creating the ADX Cluster

1. Find the Azure Data Explorer service to create a new cluster

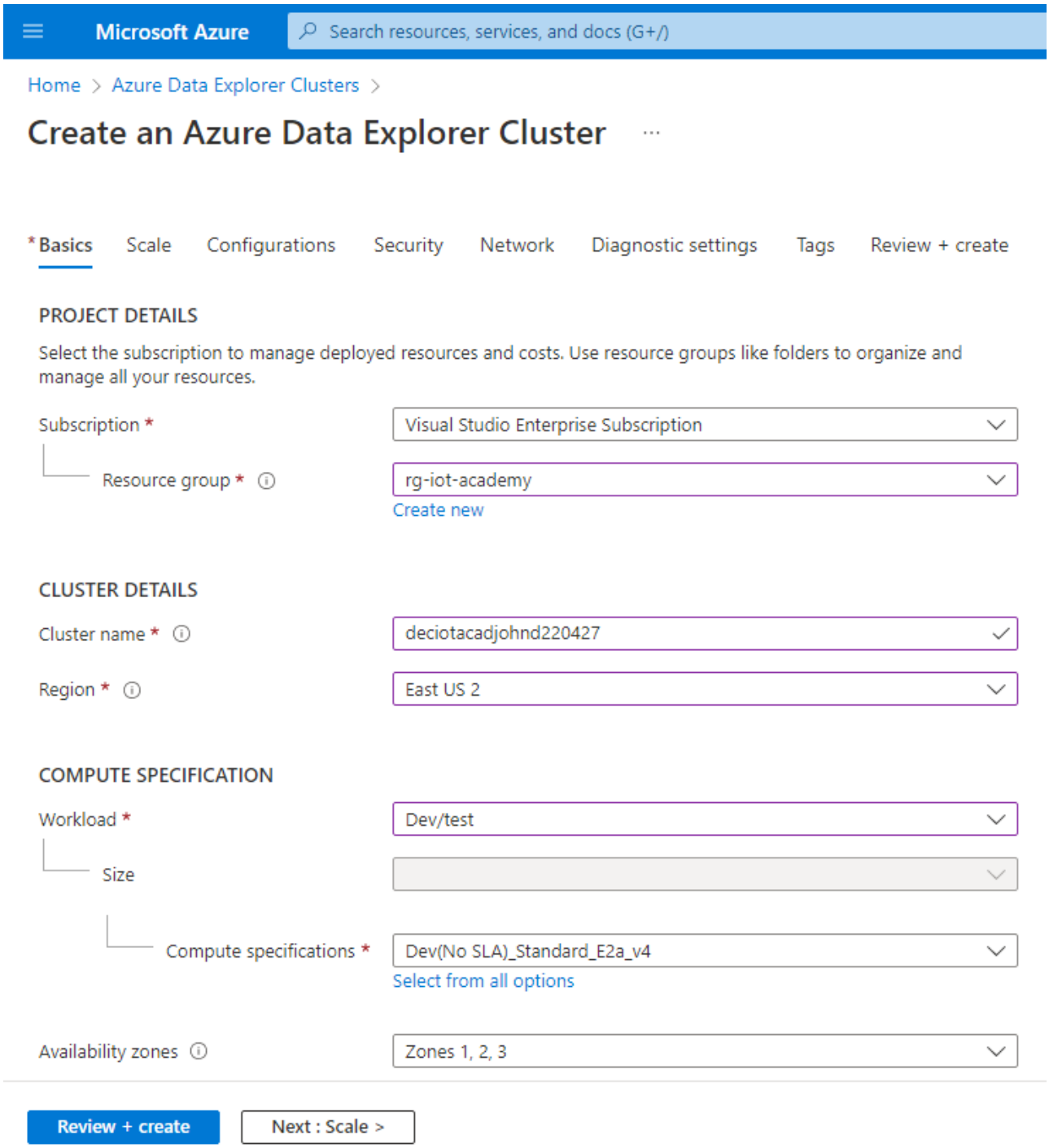
    - Go to the Azure Portal home page by selecting **Microsoft Azure** at the top of the window. Then click **Azure Data Explorer Clusters**



- Click **Create**

2. Enter the details for the new ADX cluster then click **Review + create**

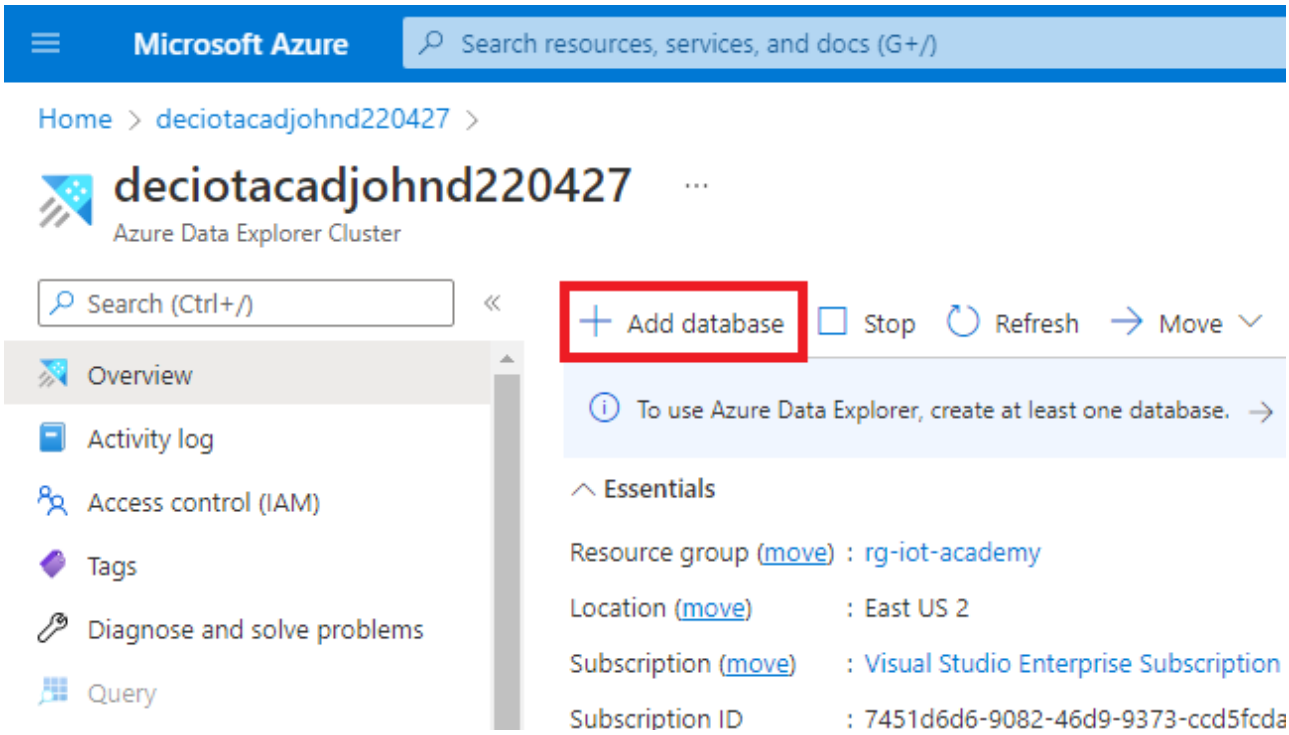
   - Details
     - Resource group: `rg-iot-academy`
     - Cluster name: `{prefix|iot|acad|johnd|220427}` e.g. `deciotacadjohnd220427`. This name must be globally unique, alphanumeric only, limited to 22 characters.
     - Region: `{Your region}` e.g. `East US 2`
     - Workload: `Dev/test`
     - Compute specifications: `Dev(No SLA)_Standard_E2a_v4`



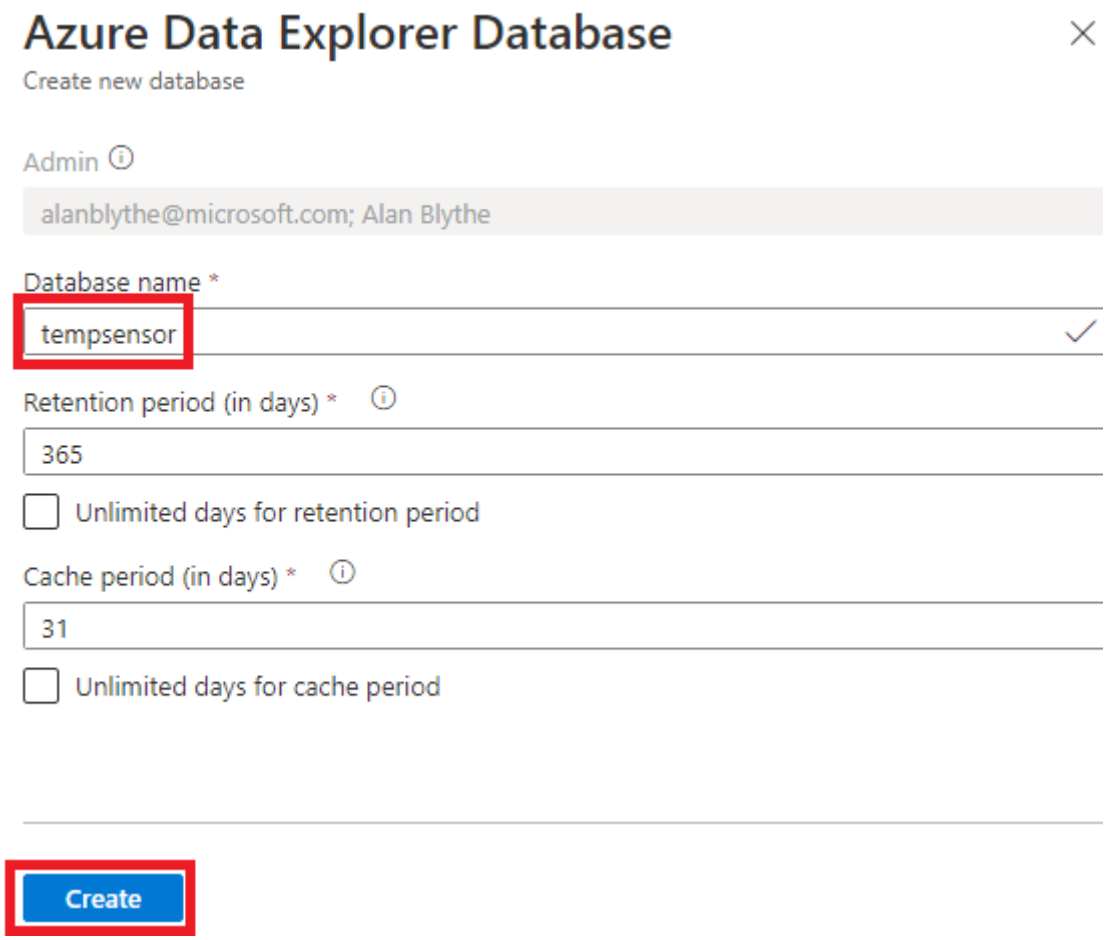
- After validation passes click **Create**
- Wait until the cluster is created and click **Go to resource**

Task 2: Adding an ADX Database

1. Now on the Overview tab for your new ADX cluster click **Add database**



2. Click **Create**



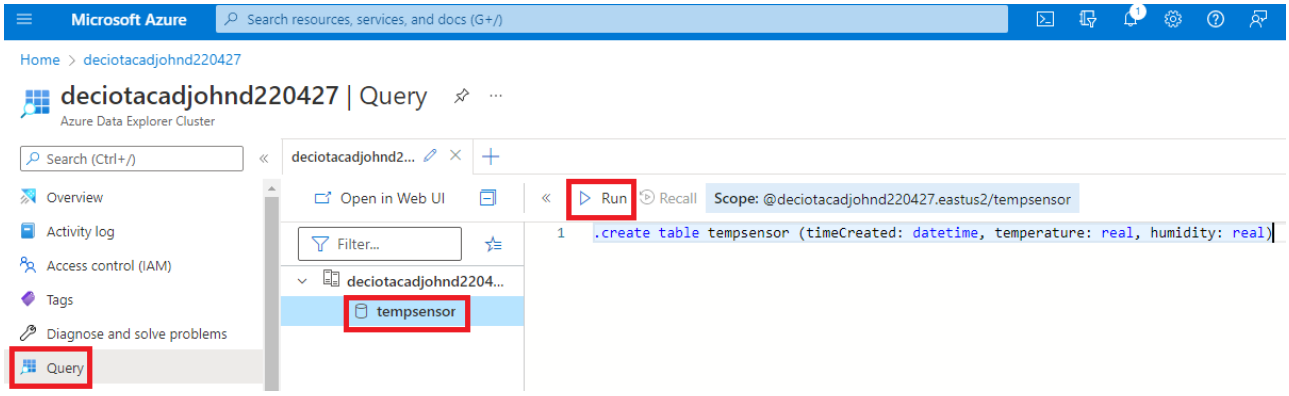
- After a few moments you'll be back at the overview page.

Task 3: Creating the tempsensor table

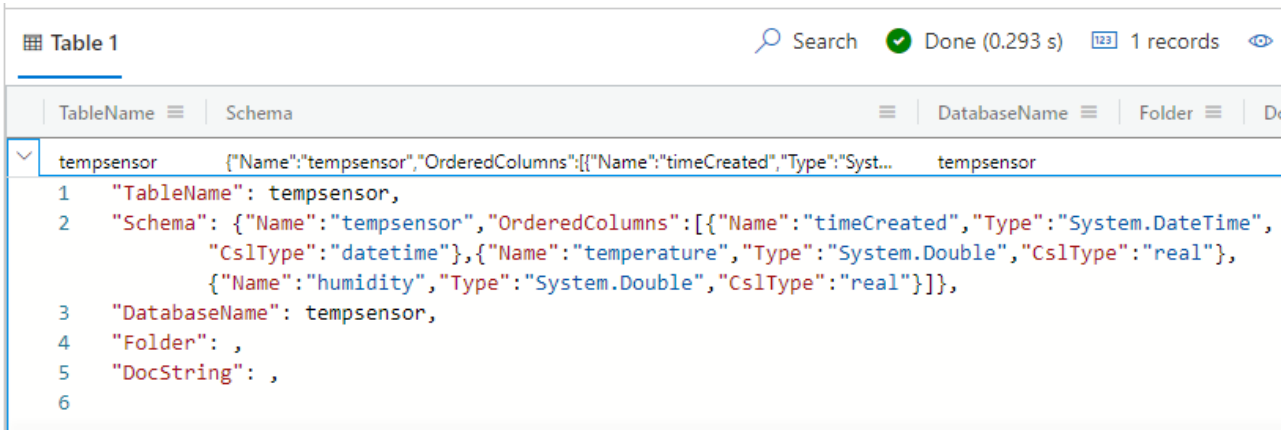
Create the tempsensor table with schema and mapping

- Click Query. Click your **tempsensor** database. Paste the create table code from below into the windows. Then click **Run**.

```
.create table tempsensor (timeCreated: datetime, temperature: real,
humidity: real)
```



- A message as follows will be received when successful.

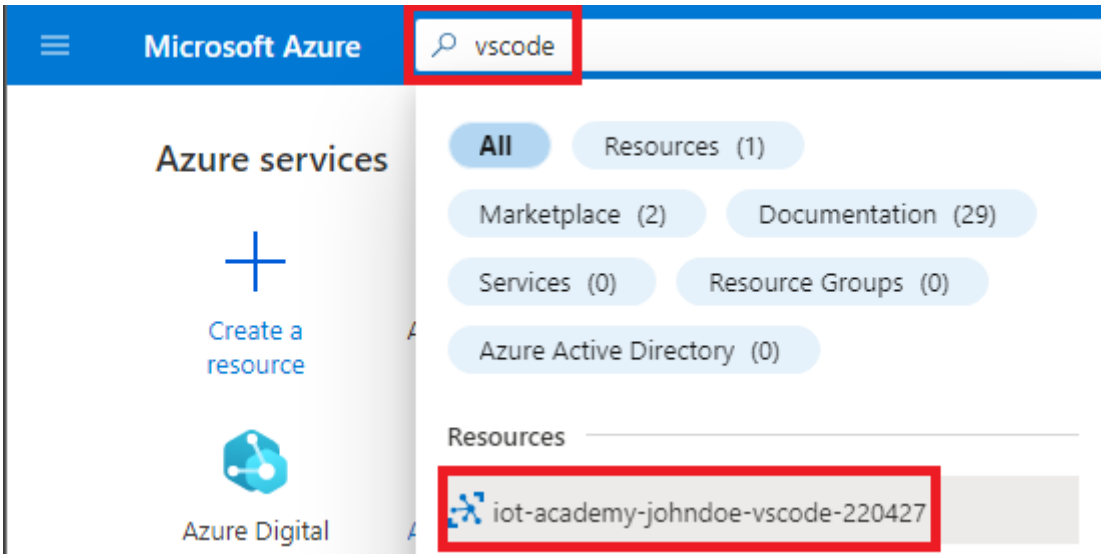


- Next, create the tempsensor ingestion mapping. Replace the command in the window with the command from the following code block. Then, click **Run**. When successfull a similar result from the last command will be observed.

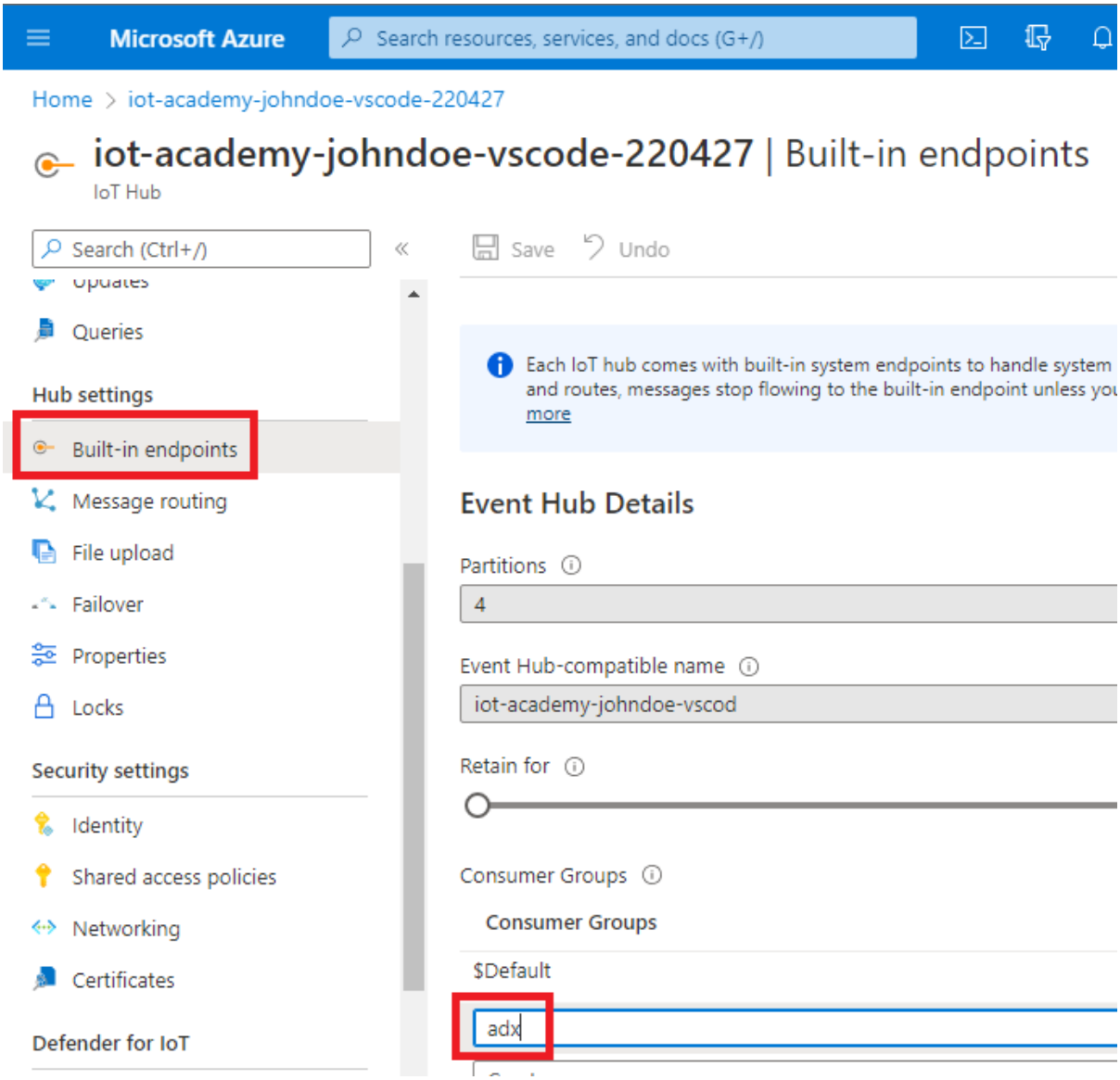
```
.create table tempsensor ingestion json mapping 'tempsensorMapping'
'[{"column":"timeCreated","path":"$.timeCreated","datatype":"datetime"},
{"column":"humidity","path":"$.ambient.humidity","datatype":"real"},
{"column":"temperature","path":"$.ambient.temperature","datatype":"real"}]'
```

Task 4: Connecting ADX to IoT Hub to Ingest Telemetry

1. Add a new Consumer Group to your IoT Hub
    - Use the Azure Portal search bar, type vscode. Your IoT Hub instance will be shown. Click your IoT Hub name to navigate to the resource.

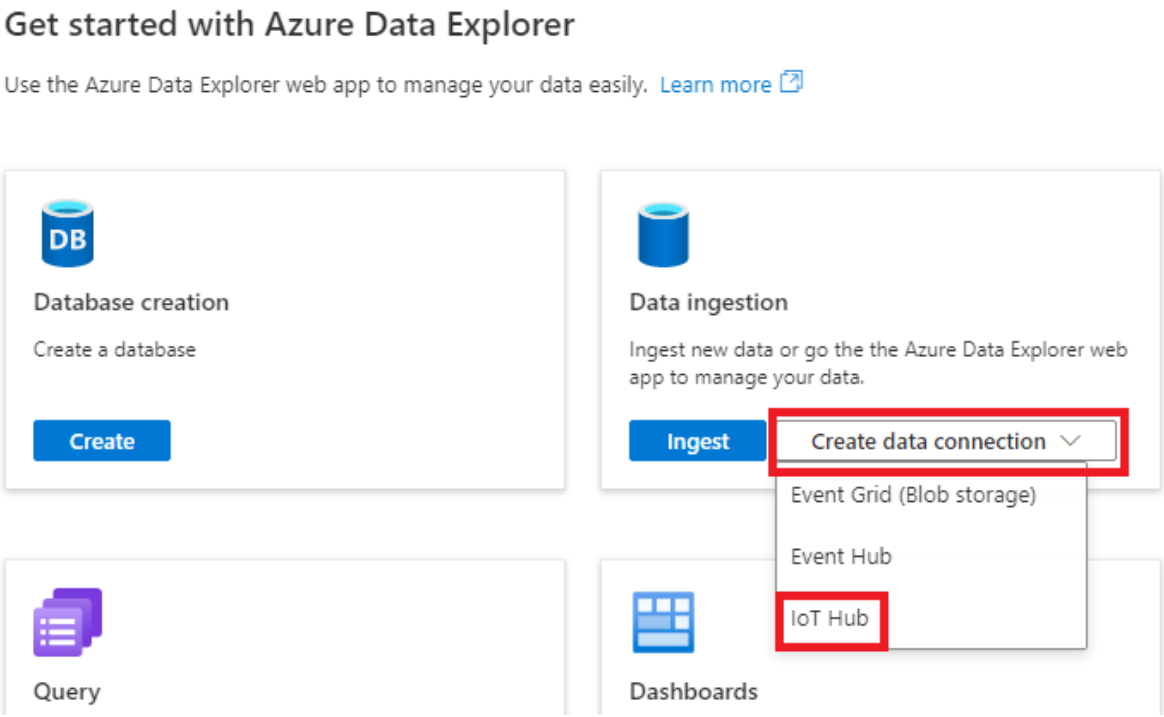


- Click **Built-in endpoints**. Add a new consumer group under the **Consumer Groups** section. Type adx in the box and press tab. When you press tab your entry will be saved.



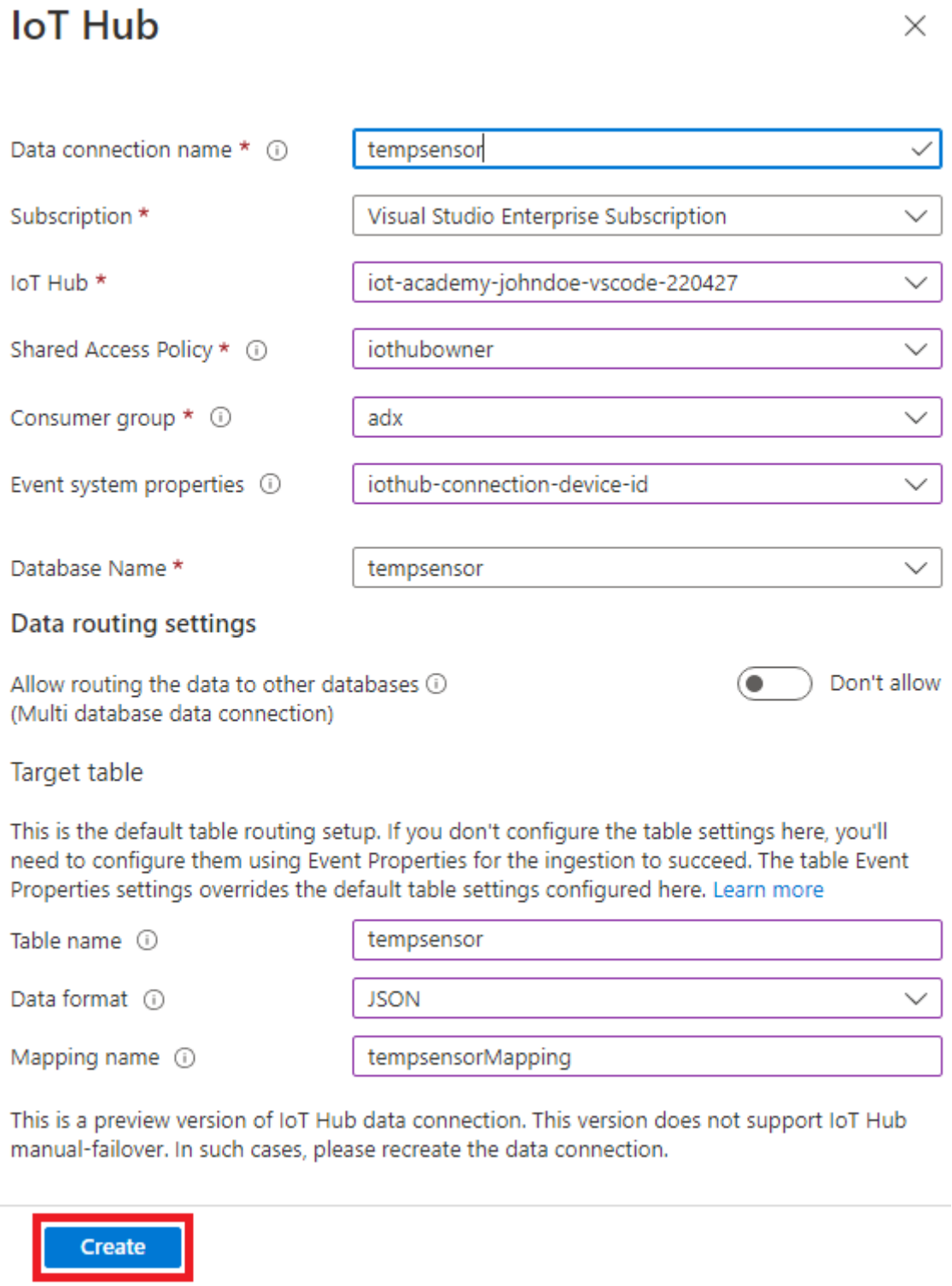
2. Create a data connection to ingest your tempsensor telemetry data

- Use the Azure Portal search bar to search for `dec`, the standard prefix for Azure Data Explorer resources. Then, click your instance to load the resource.
- Click **Create data connection** and select IoT Hub.

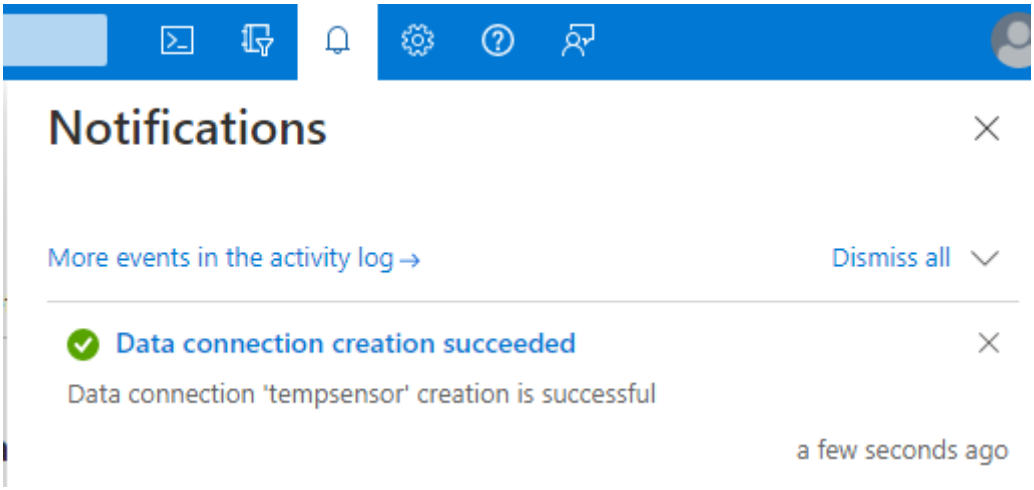


- Fill in all the fields as follows:
  - Data connection name: `tempsensor`
  - Subscription: {Your subscription}
  - IoT Hub: {Your IoT Hub that includes vscode in the name}
  - Shared Access Policy: {iothubowner}
  - Consumer Group: `adx`
  - Event system properites: Click the dropdown and select `iothub-connection-device-id`
  - Database Name: `tempsensor`
  - Table name: `tempsensor`
  - Data format: `JSON`
  - Mapping name: `tempsensorMapping`

- Click **Create**



- Watch for the notification for when the data connection creation is successful.



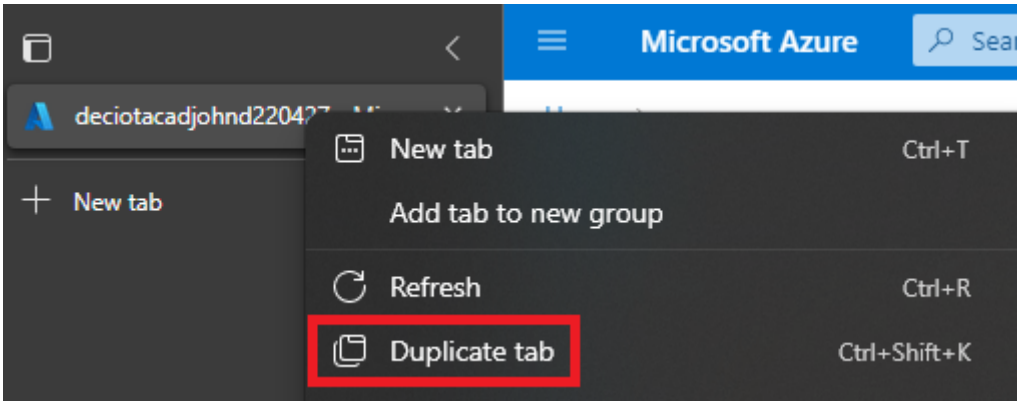
Exercise 6: Analyzing Data with Azure Data Explorer (ADX)

Task 1: Connecting ASA to IoT Hub

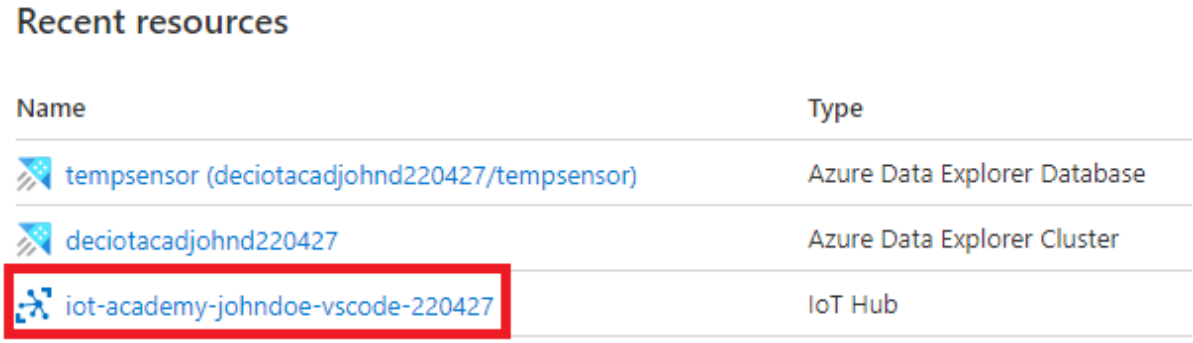
1. Restart the temperature sensor module to ensure fresh telemetry is flowing

   The `Simulated Temperature Sensor` module sends 500 telemetry messages, once every 5 seconds. This means ~40 minutes worth of messages. As our lab takes some time to work through we'll restart the module to ensure we have plenty of data to analyze with ADX.

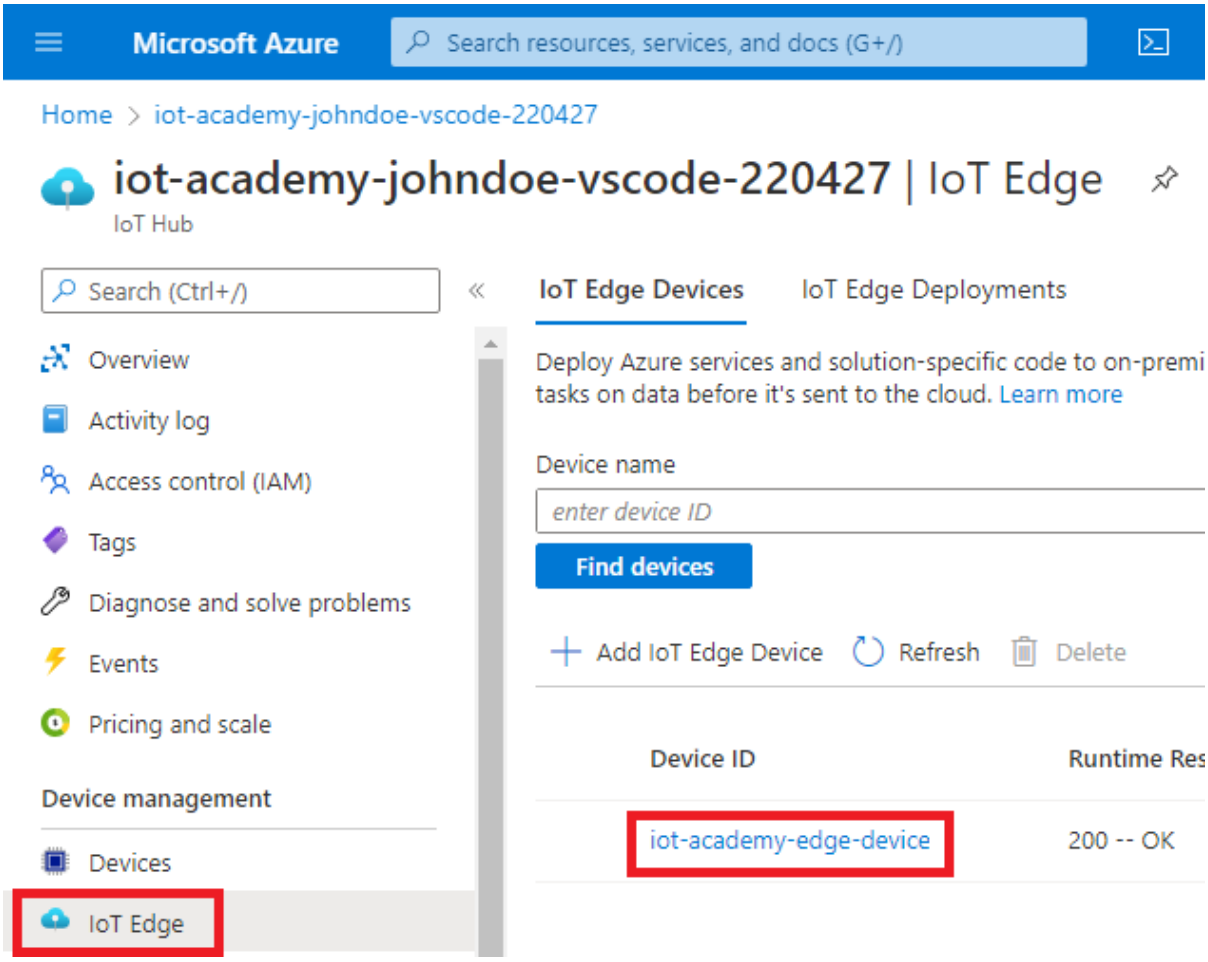
   - Create a 2nd browser tab to have both ADX and IoT Hub open in the Azure Portal. Right click the browser tab and click **Duplicate tab**.



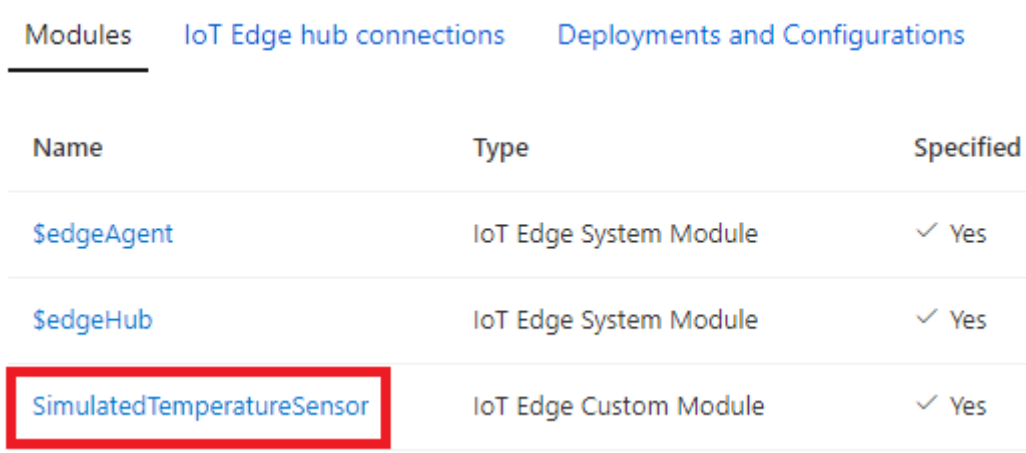
   - Switch to your 2nd tab, then click Microsoft Azure at the top of window to take you to the home page.
   - In your recent resources click your IoT Hub instance such as `iot-academy-johndoe-vscode-220427`



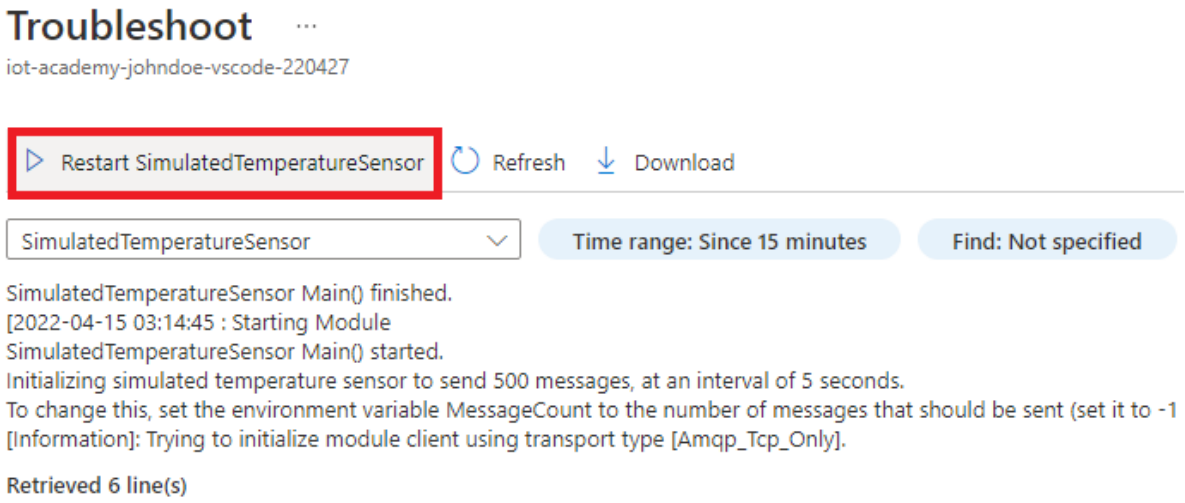
- Click **IoT Edge**, then click your device **iot-academy-edge-device**



- Click the **Simulated Temperature Sensor** module at the bottom of the window



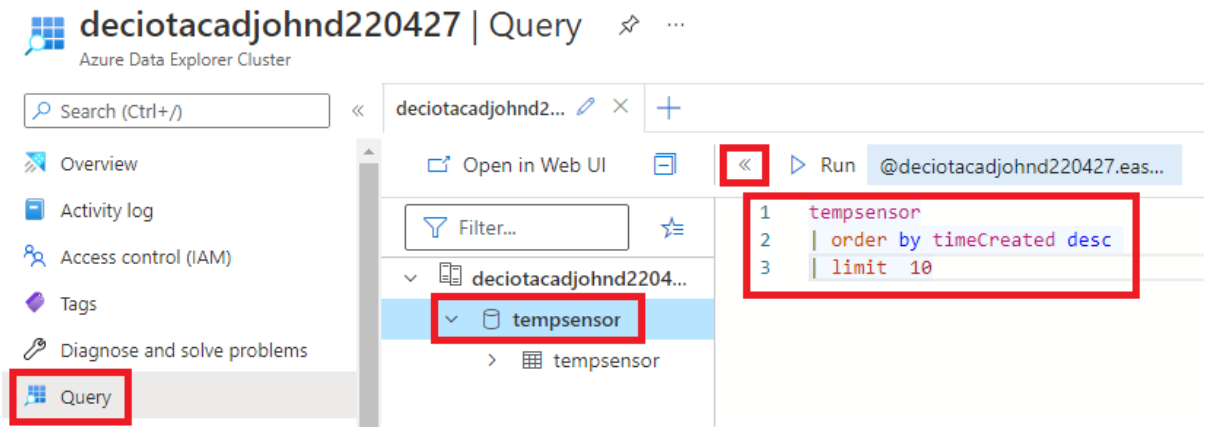
- Click **Restart Simulated Temperature Sensor** Notice the status that's received `Initializing simulated temperature sensor to send 500 messages, at an interval of 5 seconds`



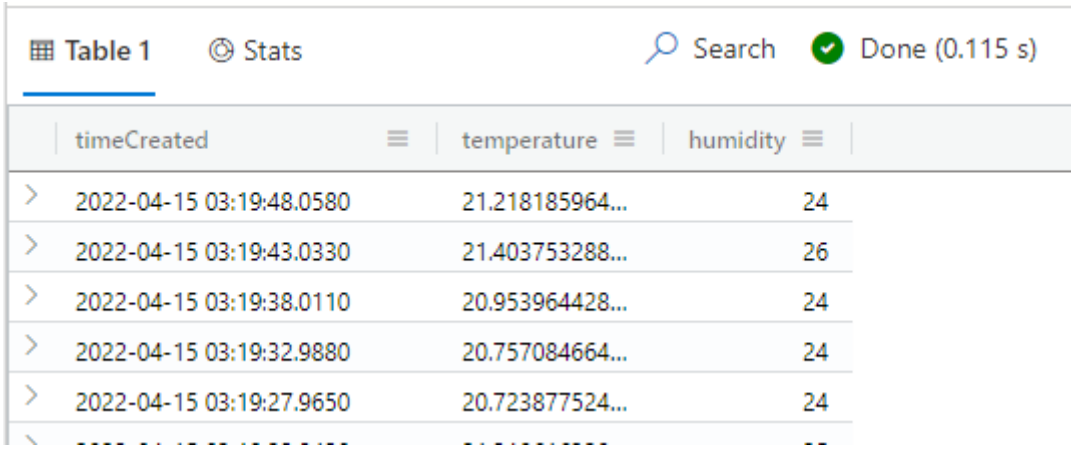
2. Review the data received by ADX

- Click your 1st browser tab where ADX is still loaded.
- Click **Query** and enter the following query. Also, click the `>>` and ensure the `tempsensor` database scope is selected.

```
tempsensor
| order by timeCreated desc
| limit  10
```



- Click the **Run** button. If you're used to SQL don't press F5 as it will refresh your browser.



Exercise 7: Processing Telemetry with Azure Stream Analytics (ASA)

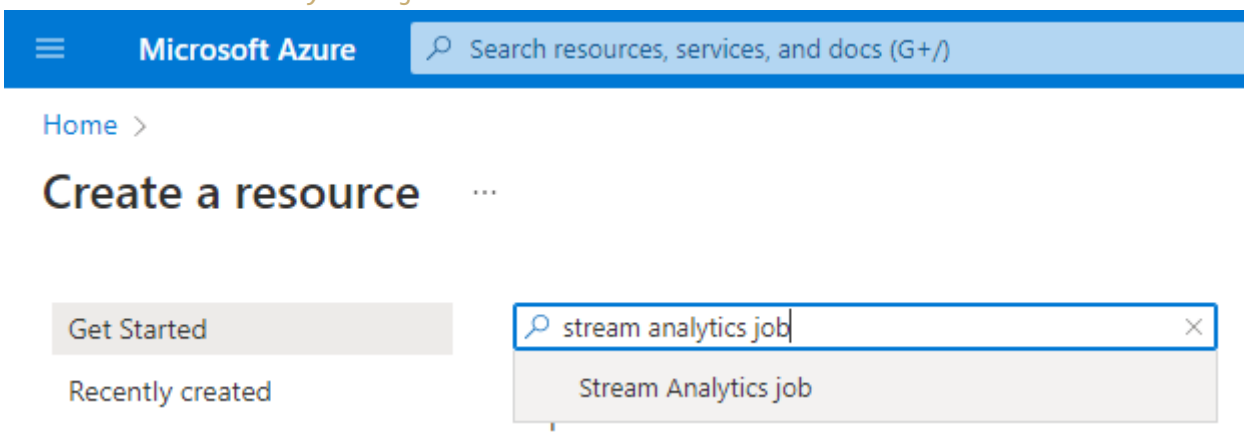
Task 1: Create a Table in ADX to Hold aggregated data

- In another Edge tab navigate to your ADX resource.
- Create a new table named `tempsensoragg` in the `tempsensor` database. Reference Exercise 5 - Task 3 if help is needed.

```
.create table tempsensoragg (WindowEnd: datetime, AverageTemperature: real)
```

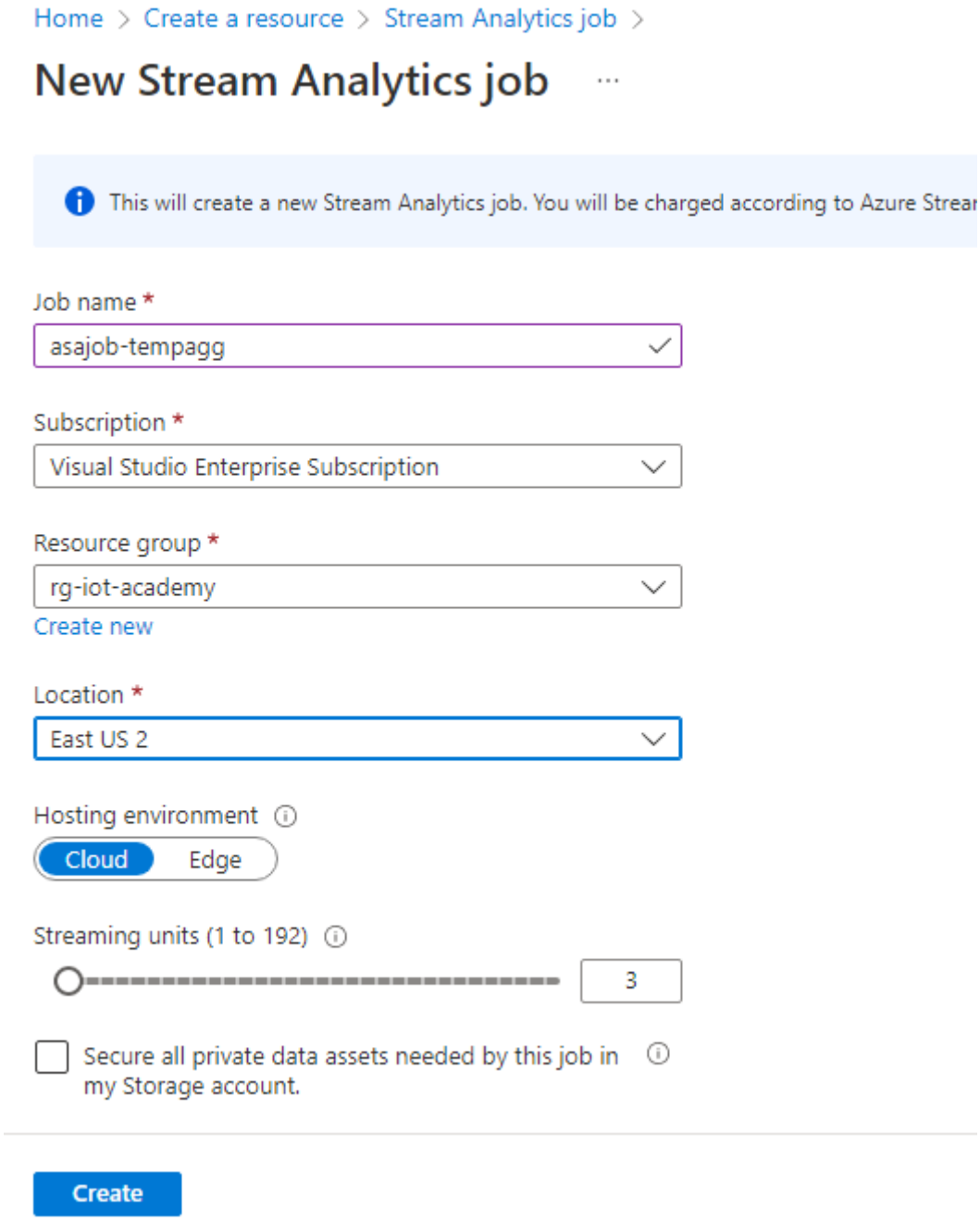
Task 2: Create a new ASA Job

1. Go to the Azure Portal home page
2. Click **Create a resource**
3. Search for `stream analytics job`



4. Click **Stream Analytics job**
5. Click **Create**
6. Enter the details for the job:
   - Job name: `asajob-tempagg`

- Resource group: `rg-iot-academy`
- Location: `East US 2`
- Hosting environment: `Cloud`
- Streaming units: `1`

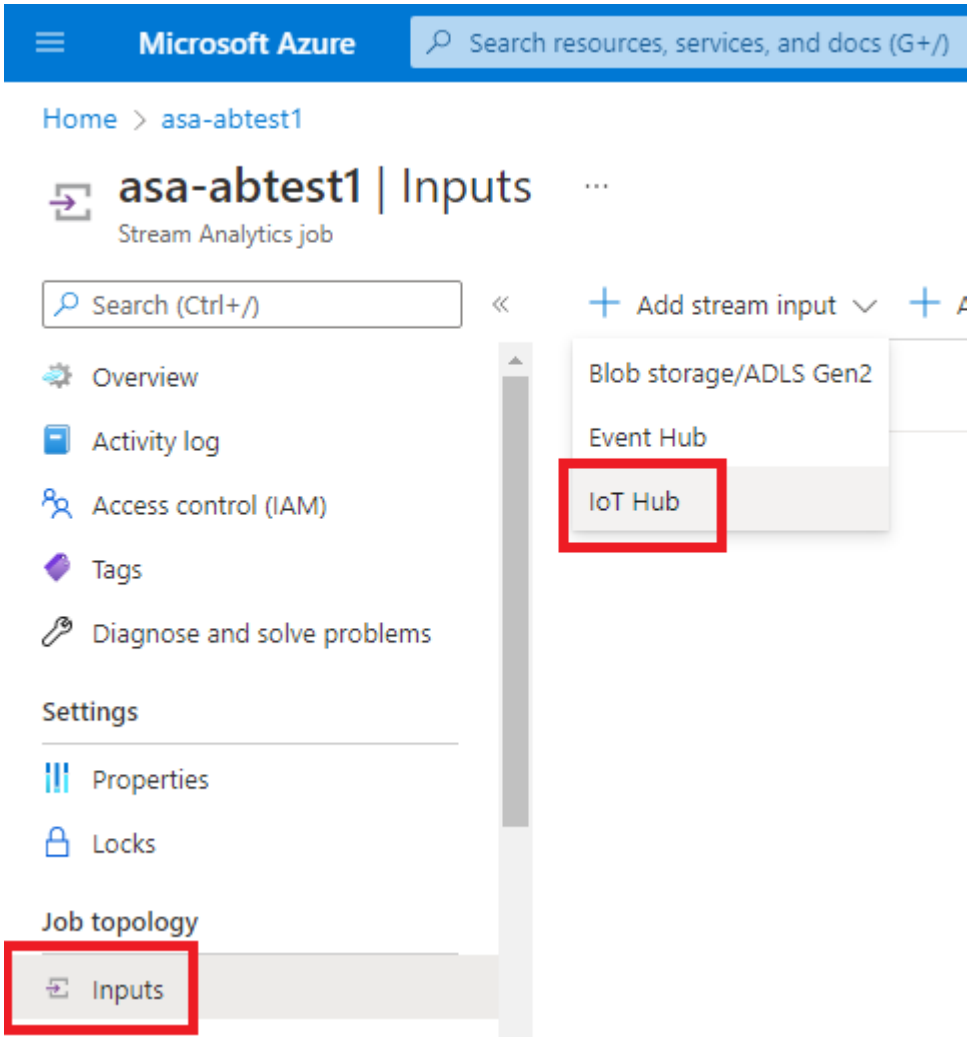


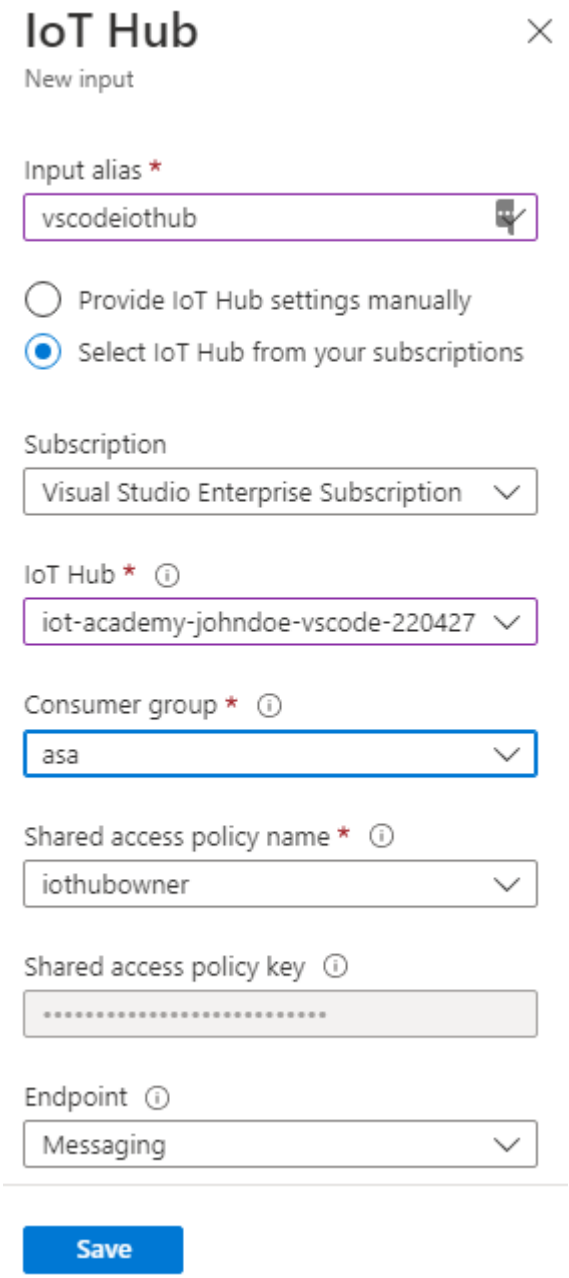
7. When the deployment is complete click **Go to resource**

Task 3: Creating an ASA Input

- In another Edge tab navigate to your IoT Hub resource.
- Create a new consumer group named `asa`. Reference Exercise 5 - Task 4 if help is needed.
- Go back to your tab that has your ASA instance.

- Click **Inputs**, Click **Add stream input**, Click **IoT Hub**



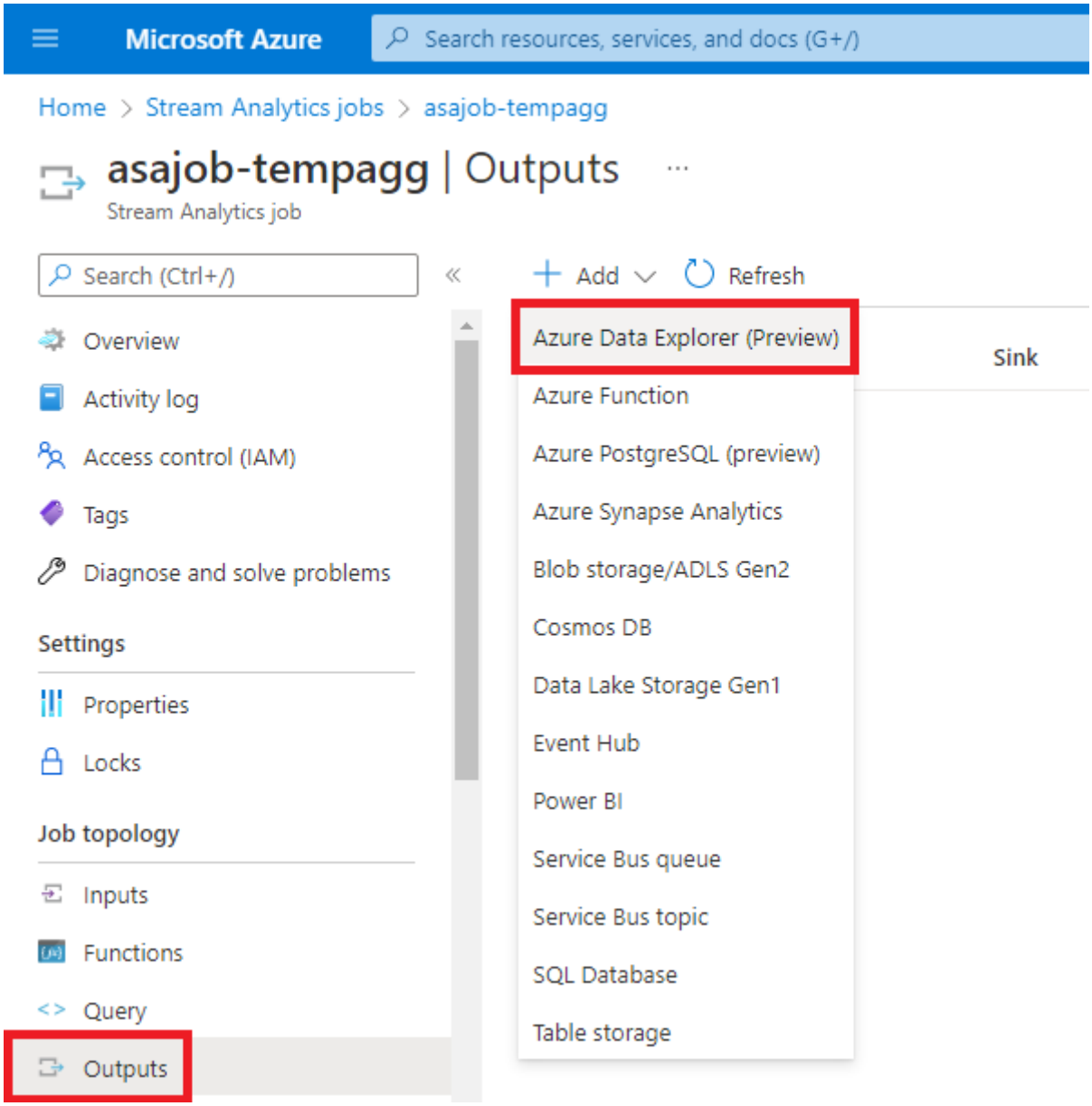


- Enter the input details:

Task 4: Creating an ASA Output

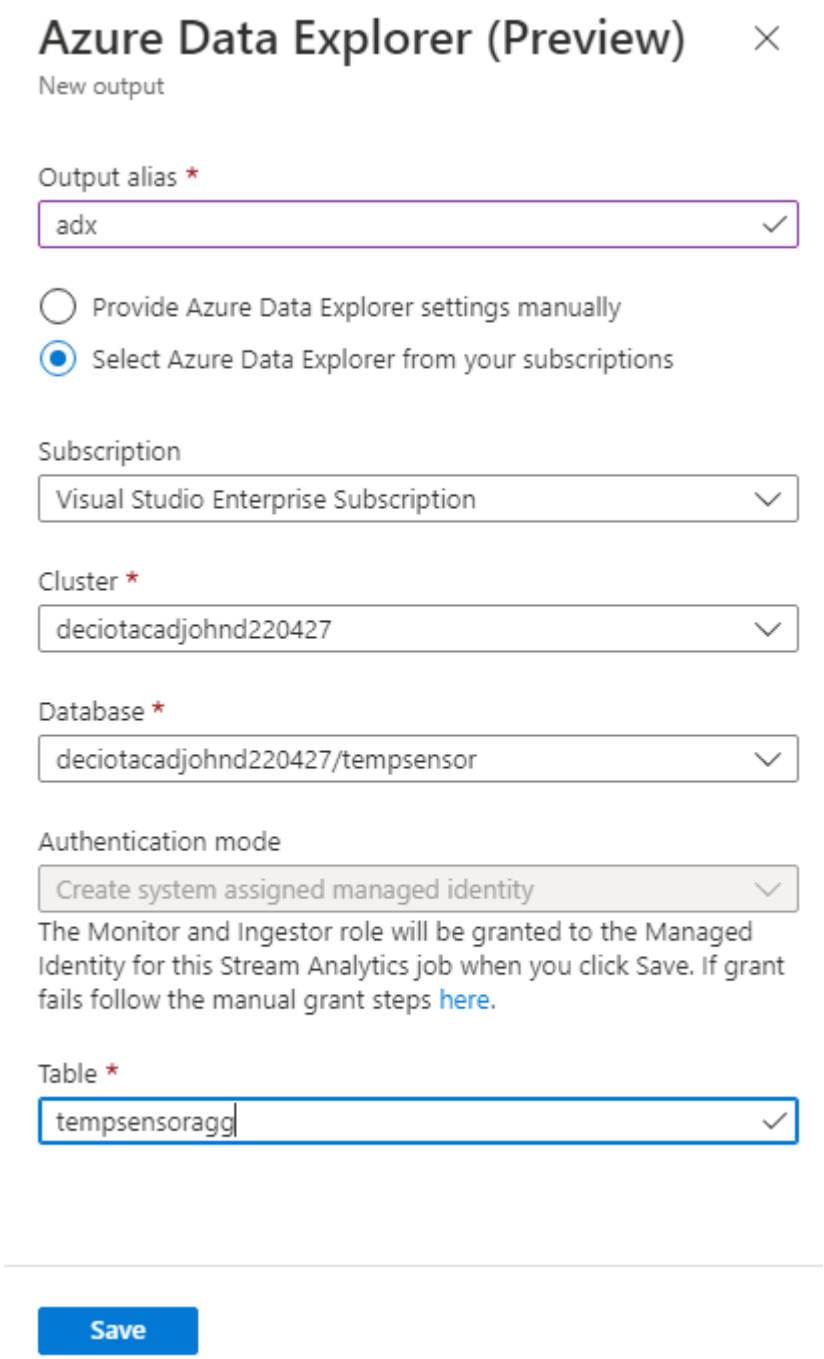
1. Click **Outputs**

2. Click **Add**, then click **Azure Data Explorer**



3. Enter the details as shown below:

   - Output alias: `adx`
   - Cluster: your cluster created earlier
   - Database: your database created earlier
   - Table: `tempsensoragg`



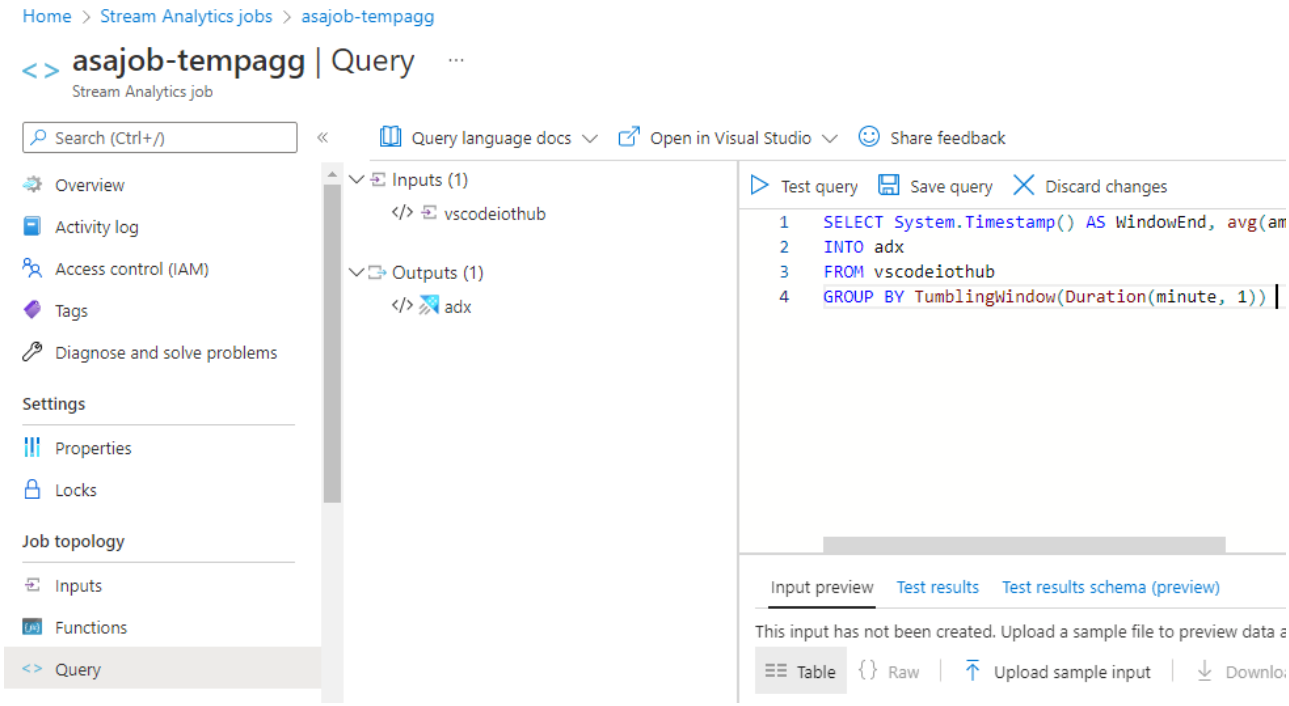
4. Click **Save**

Task 5: Writing an ASA Query

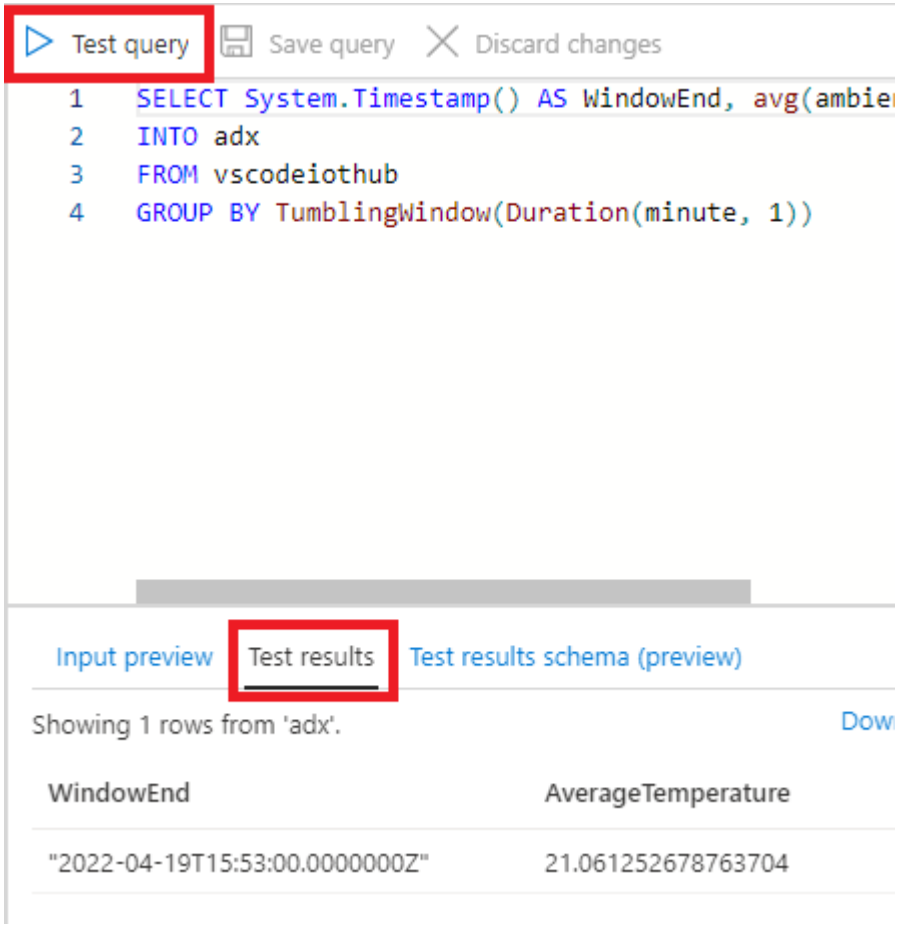
1. Click **Query**
2. Copy and paste the query below into the query window

```
SELECT System.Timestamp() AS WindowEnd, avg(ambient.temperature)
AverageTemperature
INTO adx
FROM vscodeiothub
GROUP BY TumblingWindow(Duration(minute, 1))
```

3. Click **Save query**

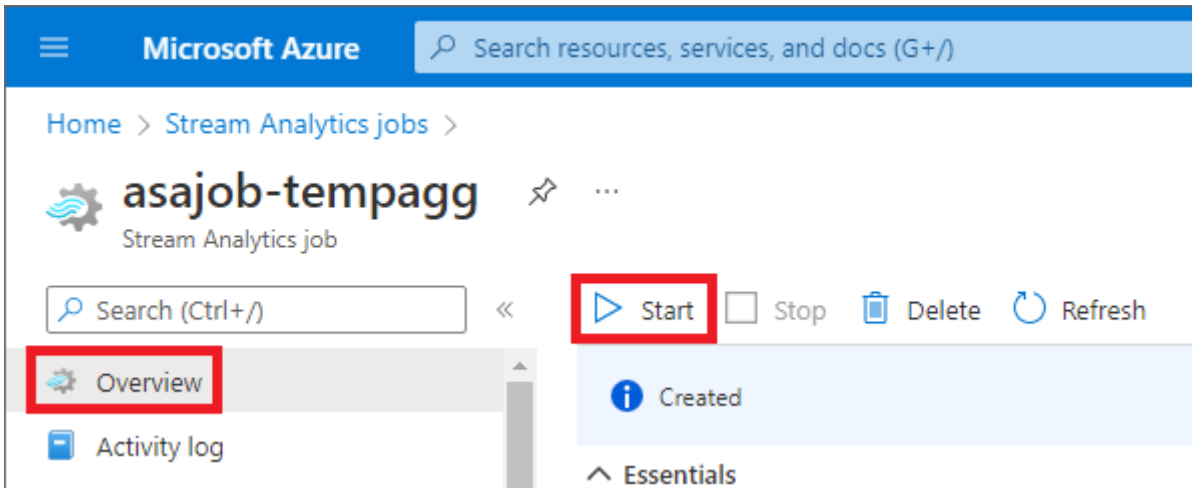


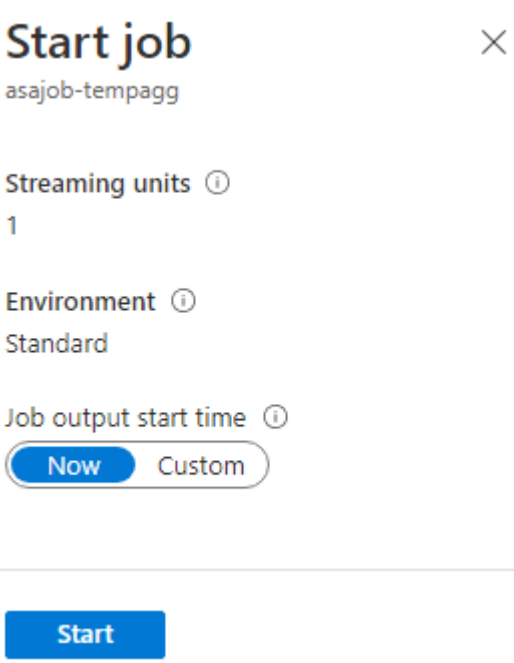
4. In another Edge tab navigate to your IoT Hub Instance
5. Reset the **SimulatedTemperatureSensor** module as described in Exercise 6 Task 1 to ensure telmetry being sent
6. Now back to your asa job, in the other open Edge tab
7. Click **Test Query**
8. Review the results in the output under the **Test Results** tab



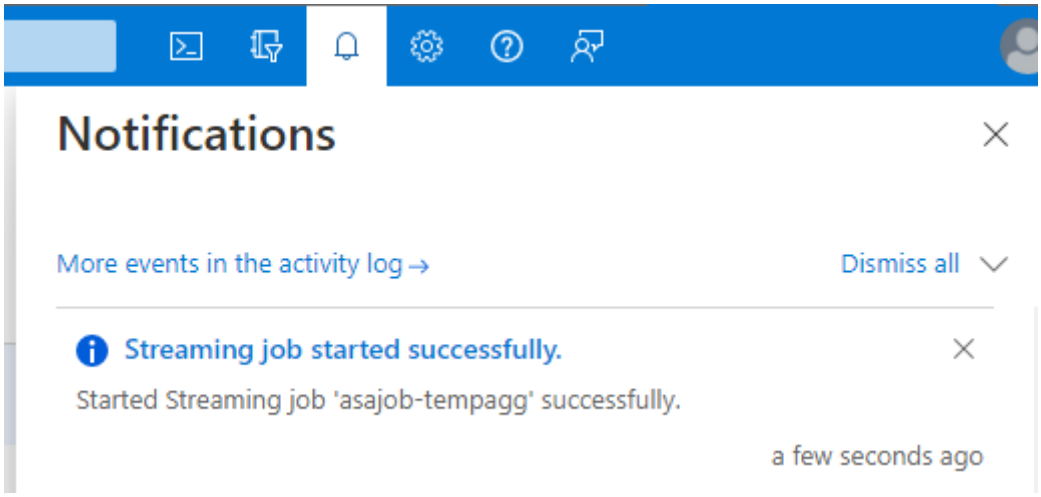
Task 6: Start the ASA Job

1. Go to the **Overview** tab
2. Click **Start**





3. Click **Start** once more
4. Watch the notification to ensure that the ASA job starts successfully.



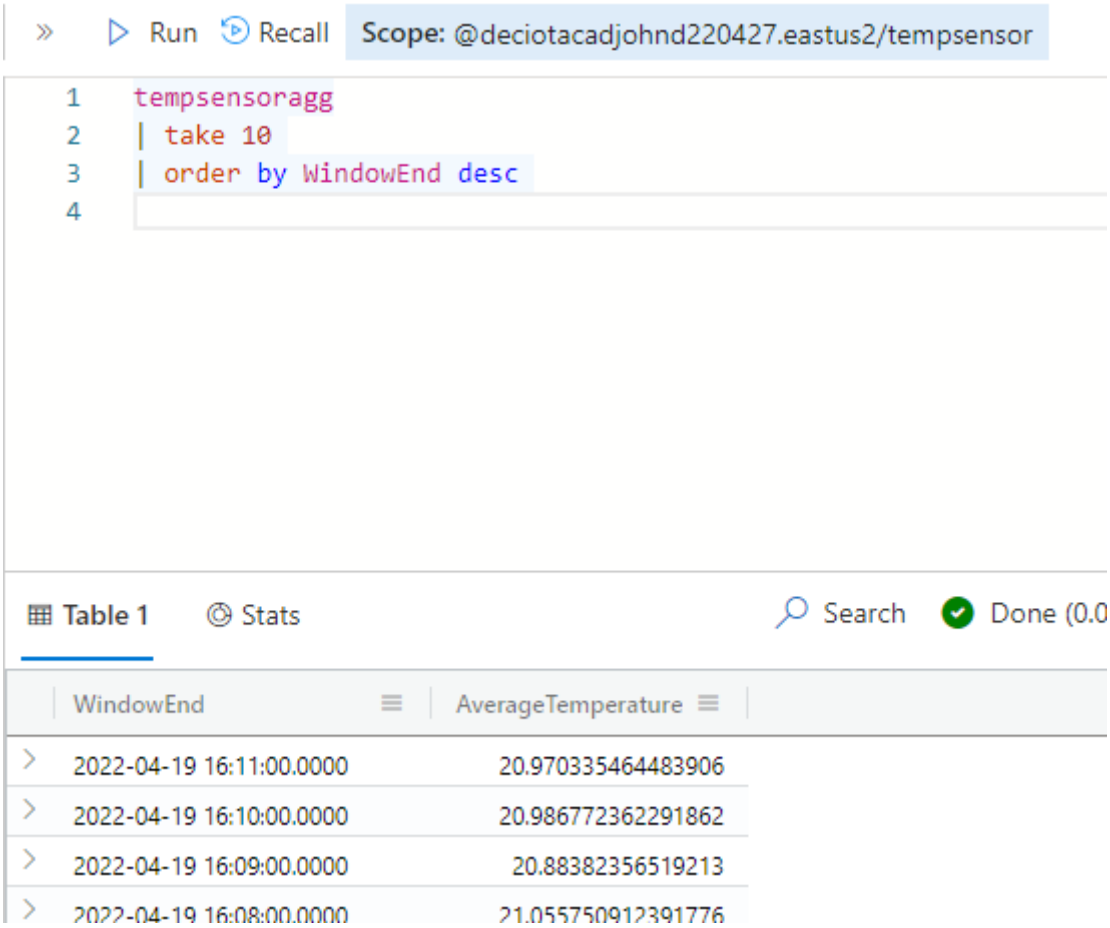
Task 7: Review Collected Aggregated in ADX

1. Navigate to your ADX resource in another Edge tab
2. Click **Query**
3. Enter the query below

```
tempsensoragg
| take 10
| order by WindowEnd desc
```

4. Click **Run**

5. Continue clicking **Run** occassionally for 10 - 15 minutes until you see your data appear in the results as shown



Exercise 8: Cleanup

It's important to not leave your Azure resources provisioned until the next Azure IoT Academy session as there may not be enough Azure credit for the next sessions.

1. Navigate to the Azure Portal home page
2. Click **Resource groups**
3. Click **rg-iot-academy**
4. Click **Delete resource group**