Hands-on-lab

Create a People Counting solution with Azure Percept DK and Azure Percept Vision

Prepared by Microsoft AEDPLS Customer Success Team

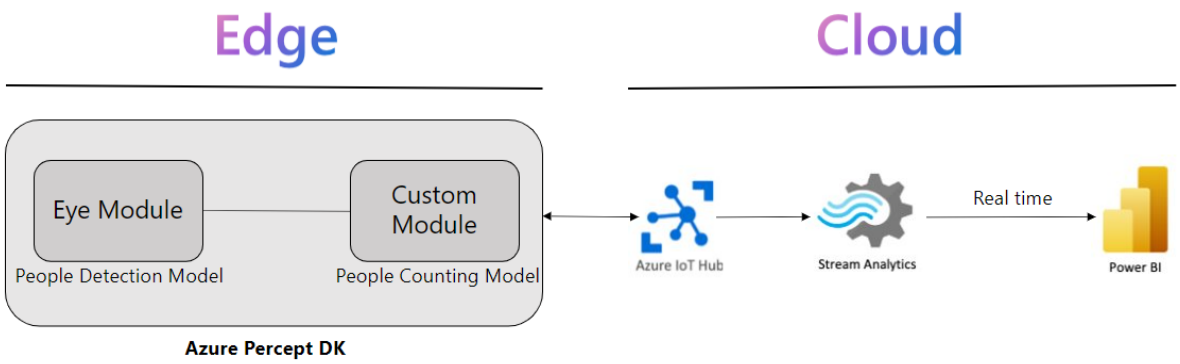
11-10-2021

Table of Contents

Lab Overview

This lab will focus on detecting and counting people using the Azure Percept DK hardware, Azure IoT Hub, Azure Stream Analytics, and Power BI dashboard

Solution Architecture



Prerequisites

1. Azure Percept DK
2. Azure Percept Vision
3. Azure Subscription
4. Azure Percept DK setup experience - you connected your devkit to a Wi-Fi network, created an IoT Hub, and connected your devkit to the IoT Hub
5. Download and install VS Code
6. Download and Install Git
7. Install the IoT Hub Extension in VS Code
8. Install the Azure IoT Tools Extension in VS Code
9. Download and Install Docker Desktop (Will require PC restart)
10. (**Only for Windows Users**) Install WSL2 by running the following command in Windows PowerShell or Terminal (on MacOS) (Will require a PC restart)

    wsl --install

    wsl --set-default-version 2

Set up your Azure Percept DK and Vision

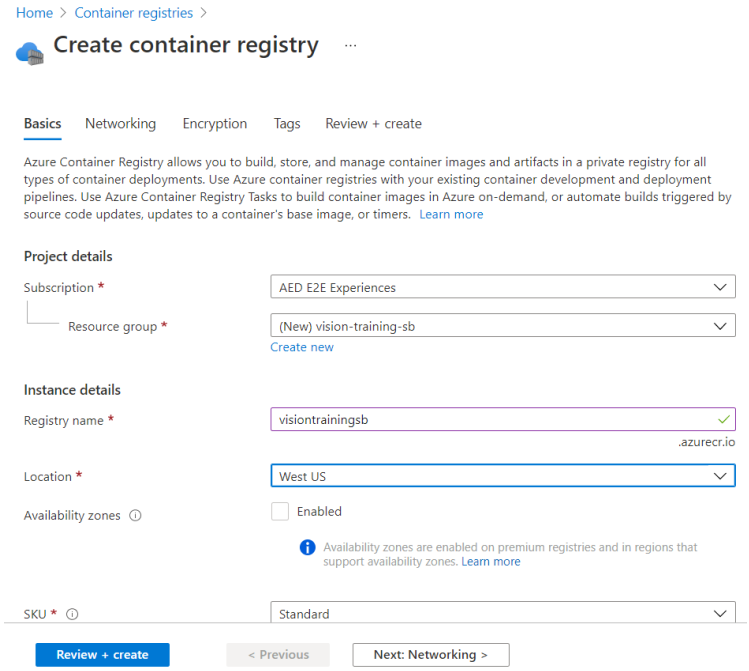
1. Power on your Azure Percept DK
2. Connect the camera module to the Azure Percept DK via the USB-C cable
3. Open Command Prompt (on Windows) or Terminal (on MacOS) and execute the command-

   git clone https://github.com/leannhuang/people-counting-with-azure-percept-vision.git

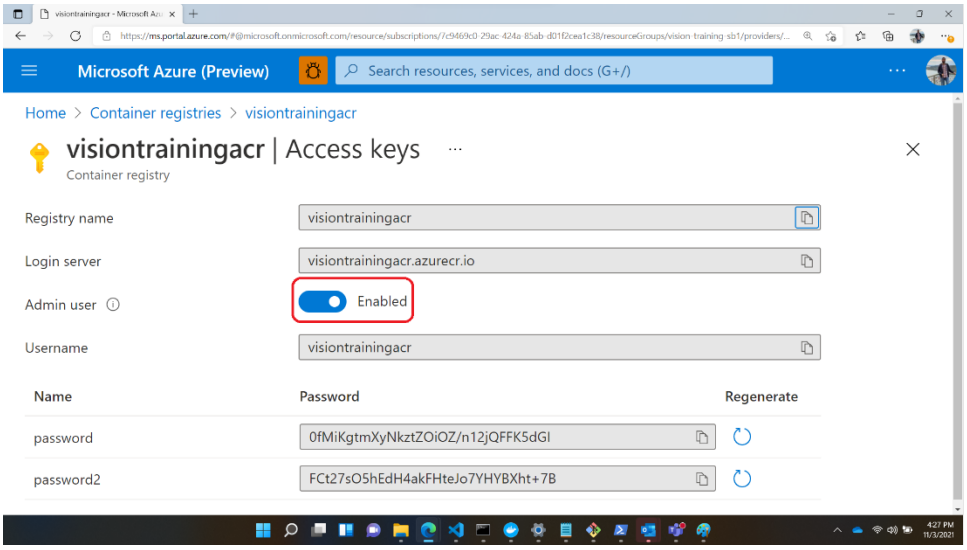
Note- If you do not have Docker installed, please proceed to Exercise 2 – Task 3

Exercise 1 - Create a Container Registry resource (5 min)

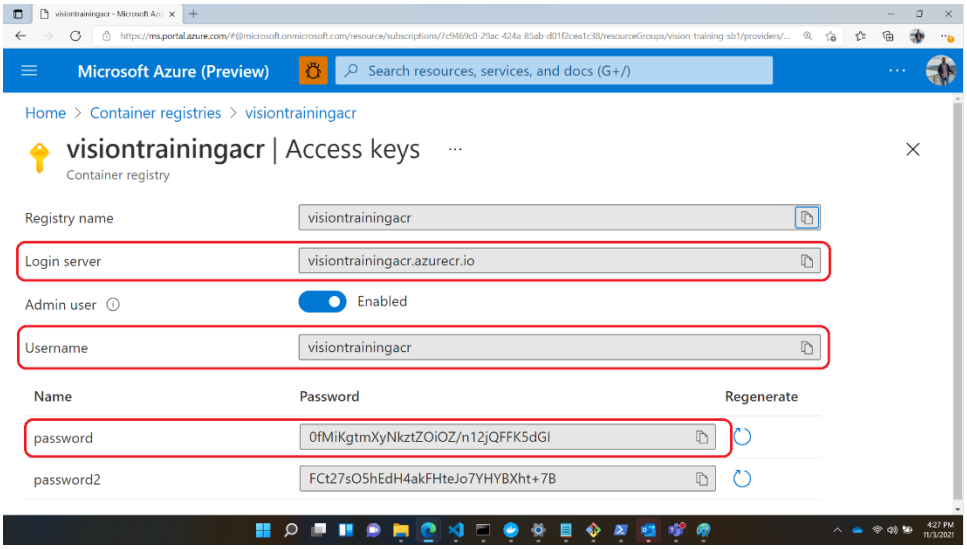
1. Login to Azure Portal https://portal.azure.com/
2. To create a Container Registry, go to Create container registry - Microsoft Azure
    a. Select your Azure Subscription in the **Subscription** drop down box
    b. Select your preferred resource group from the **Resource group** drop down menu. If you would like to create a new resource group to use with your voice assistant, click **Create new** under the dropdown menu and follow the prompts
    c. Provide a unique **Registry Name**
    d. Under **Location**, select the region to deploy resource (We suggest select **West US**)
    e. **Availability Zones** - disabled
    f. For **SKU**, select **Standard**
    g. Keep all other tab as default and click **Review + create** at the bottom of the screen. Once the validation passes, click **Create**. This will create your Container Registry.



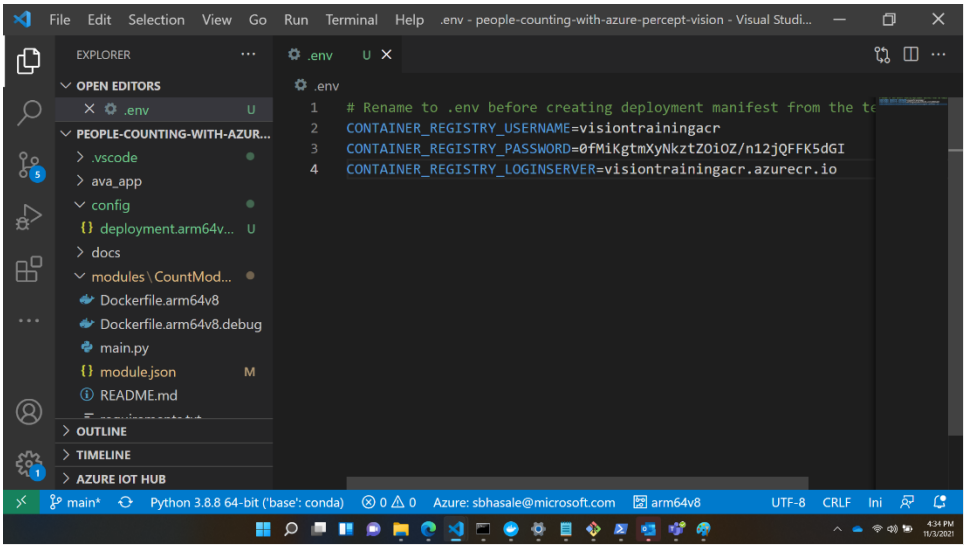
3. After successful resource deployment go to your container registry resource. On the left scroll panel select **Access Keys** under **Settings** and **enable** the **Admin user**



4. Make a note of the **Login Server**, **Username,** and **password**



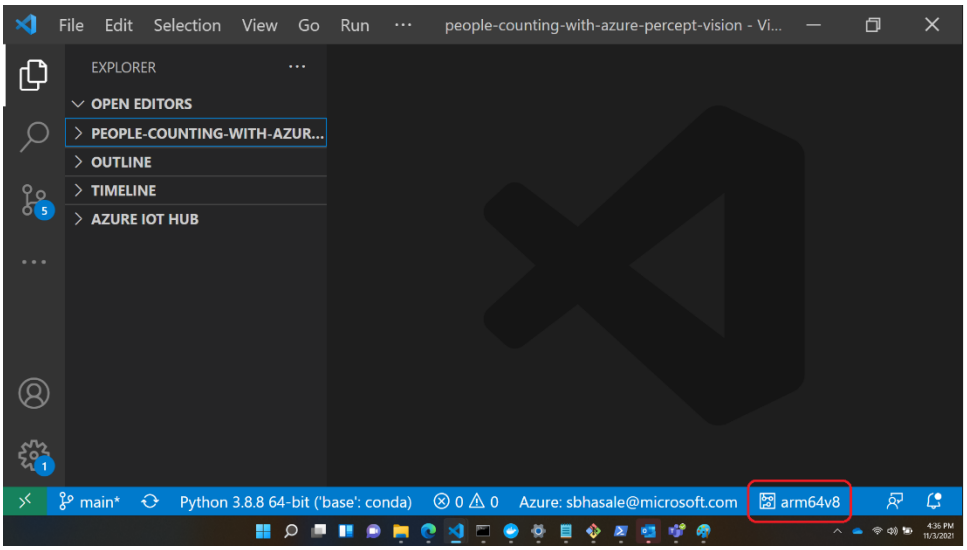
5. Go to the git repository that you cloned in VS Code. Rename the file **envtemplate** to **.env** open the file and fill in the following details-
    a. CONTAINER_REGISTRY_USERNAME=<your container registry Username>
    b. CONTAINER_REGISTRY_PASSWORD=<your container registry Password>
    c. CONTAINER_REGISTRY_LOGINSERVER=<your container registry Login Server>



Exercise 2 - Setup for Edge Deployment (20 min)

Ex 2. - Task 1 - Build and push your IoT edge solution (10 min)

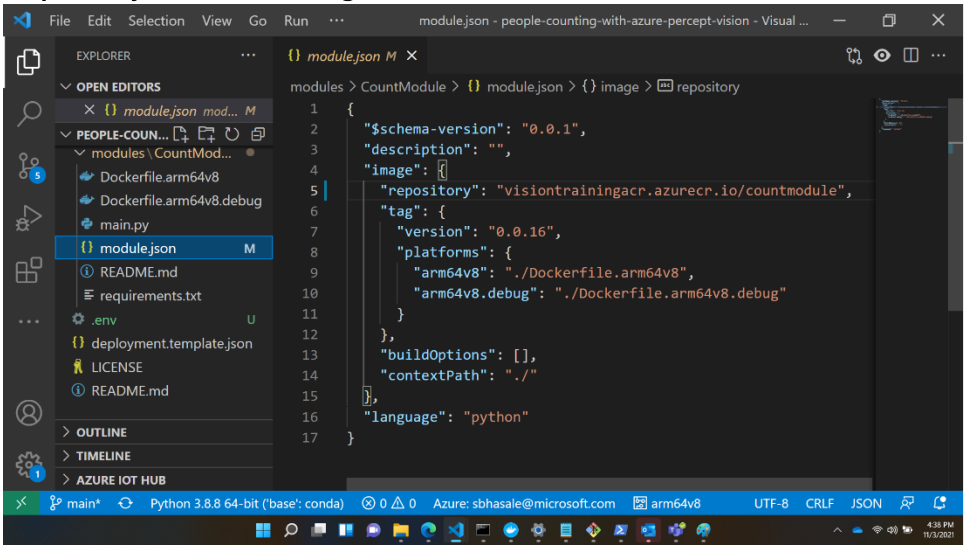
1. Open VS Code, at the bottom of the screen ensure you have **arm64v8** as the **Default Platform for IoT Edge Solution** selected (if not, then please click and select **arm64v8** from the list)



2. Go to **modules/CountModule/** directory and open **module.json**. Fill in your **Container registry address** (same as the **Login server** saved earlier) and followed by a **repository name** (Note- please make sure your repository name is all lowercase)

   "repository": "<Your container registry login server/repository name>" will change as follows, for example-

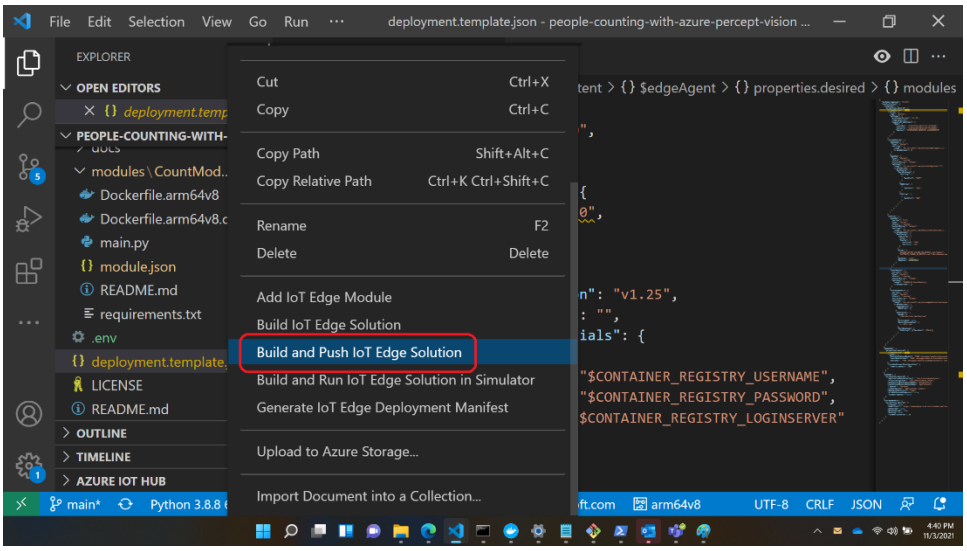
   "repository": "visiontrainingacr.azurecr.io/countmodule"



3. Now you will build the module image and push it to your container registry. Open Visual Studio Code integrated terminal by selecting **View > Terminal**
4. Sign into Docker with the Azure Container registry (ACR) credentials that you saved after creating the registry using below command in terminal-

   docker login -u <ACR username> -p <ACR password> <ACR login server>

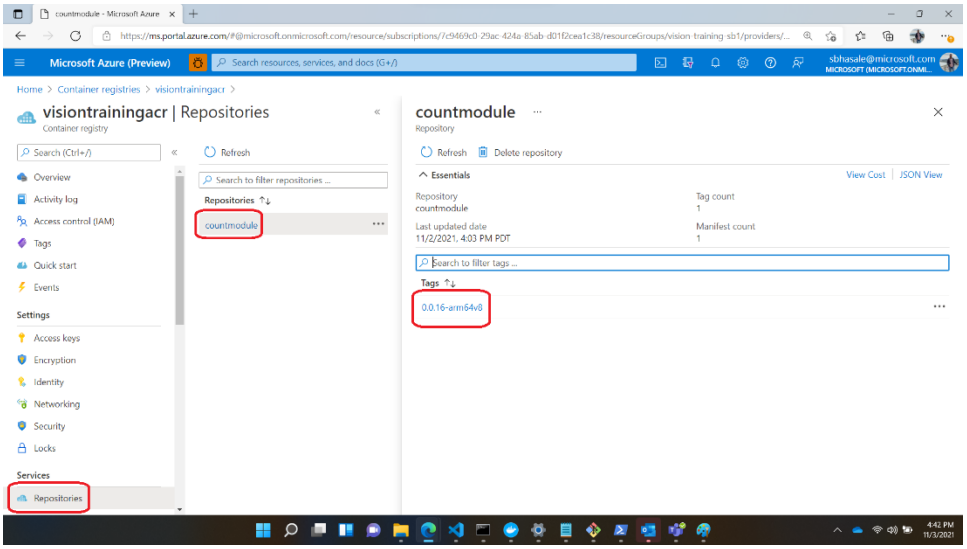
5. Visual Studio Code now has access to your container registry. In the next steps you will turn the solution code into a container image. In Visual Studio Code explorer, right click the **deployment.template.json** file and select **Build and Push IoT Edge Solution**



   The build and push command starts three operations. First, it creates a new folder in the solution called **config** that holds the full deployment manifest, built out of information in the deployment template and other solution files. Second, it runs **docker build** to build the container image based on the appropriate docker file for your target architecture. Then, it runs **docker push** to push the image repository to your container registry. This process may take several minutes the first time but is faster the next time that you run the commands.

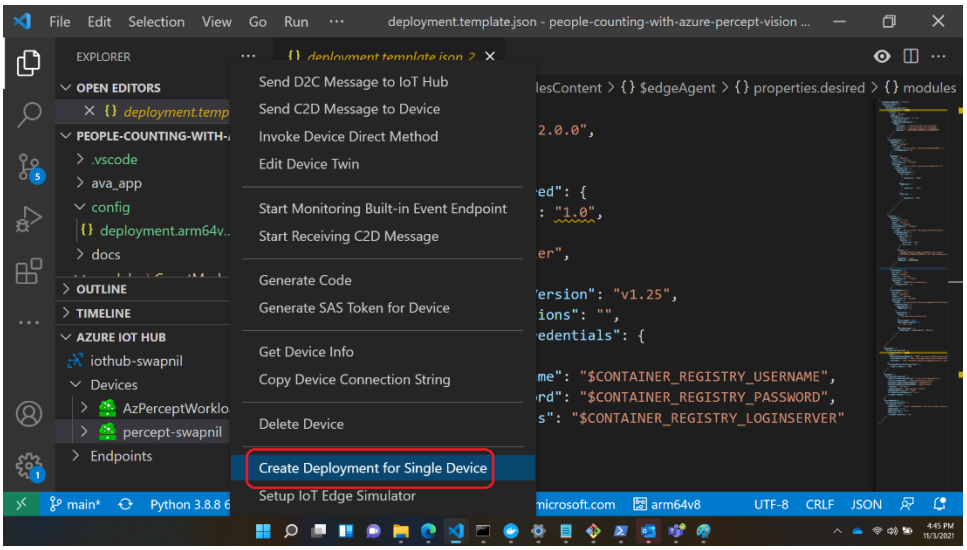
6. Open the **deployment.arm64v8.json** file in the newly created **config** folder. The filename reflects the target architecture, so it will be different if you choose a different architecture.
7. Notice that the two parameters that had placeholders now are filled in with their proper values. The **registryCredentials** section has your registry username and password pulled from the .env file. The **CountModule** has the full image repository with the **name**, **version**, and **architecture** tag from the **module.json** file.

8. To further verify what the build and push command did, go to the Azure portal, and navigate to your container registry. In your container registry, select **Repositories** then **countmodule**



Ex 2. - Task 2 - Deploy edge module to device (10 min)

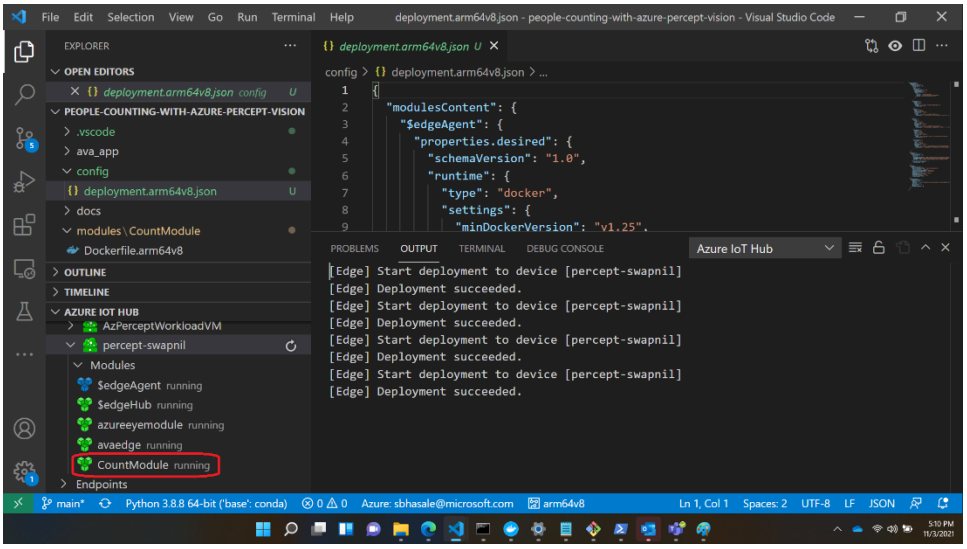
1. In the Visual Studio Code explorer, under the **Azure IoT Hub** section, expand **Devices** to see your list of IoT devices
2. Right-click the IoT Edge device that you want to deploy to, then select **Create Deployment for Single Device**



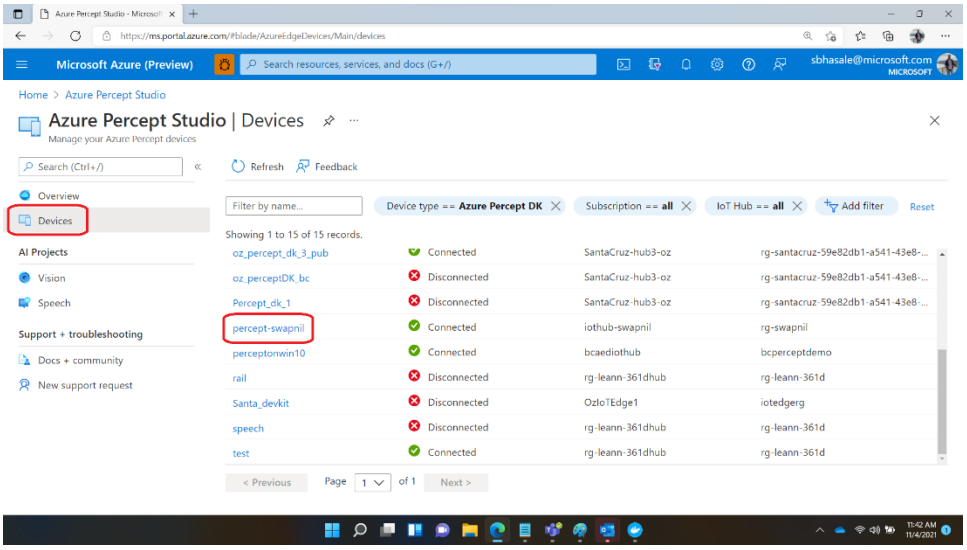
3. In the file explorer, navigate into the **config** folder then select the **deployment.amd64v8.json** file and click **Select Edge Deployment Manifest**.

   Do not use the **deployment.template.json** file, which does not have the container registry credentials or module image values in it.

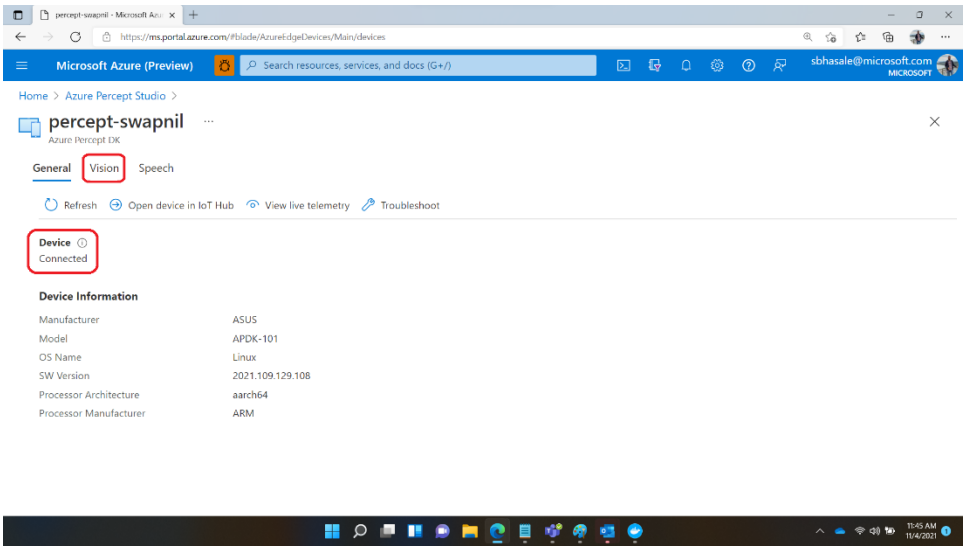
4. Under your device, expand **Modules** to see a list of deployed and running modules. Click the refresh button. You should see the **CountModule** running on your device.



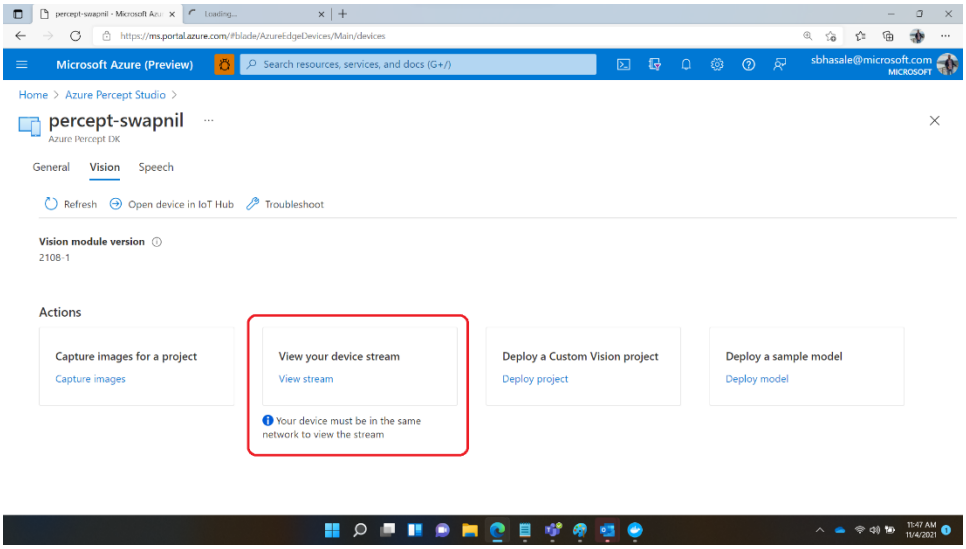
5. Go to [Azure Percept Studio] and on the left panel, select **Devices**, then select your Azure Percept device



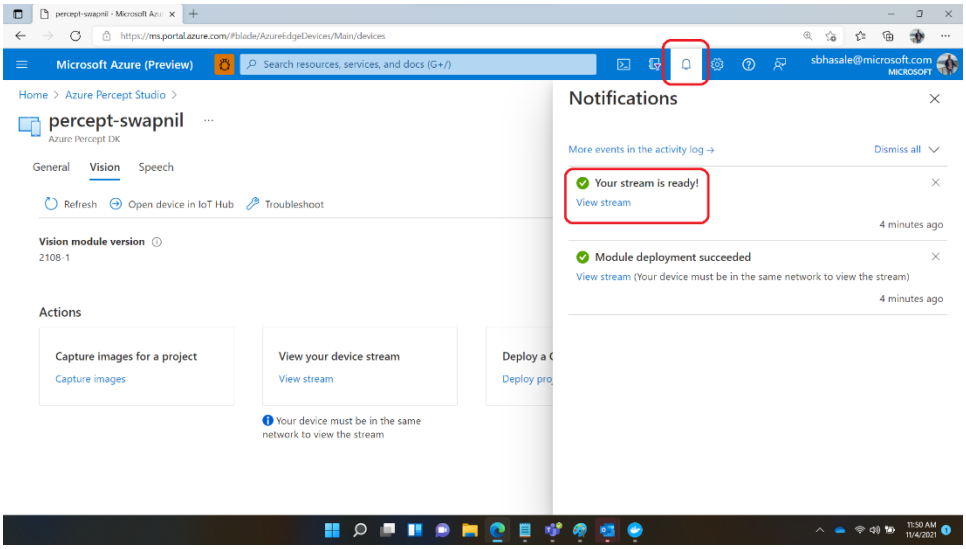
6. Ensure that your device is **Connected**. Click on **Vision**



7. Click View your device stream

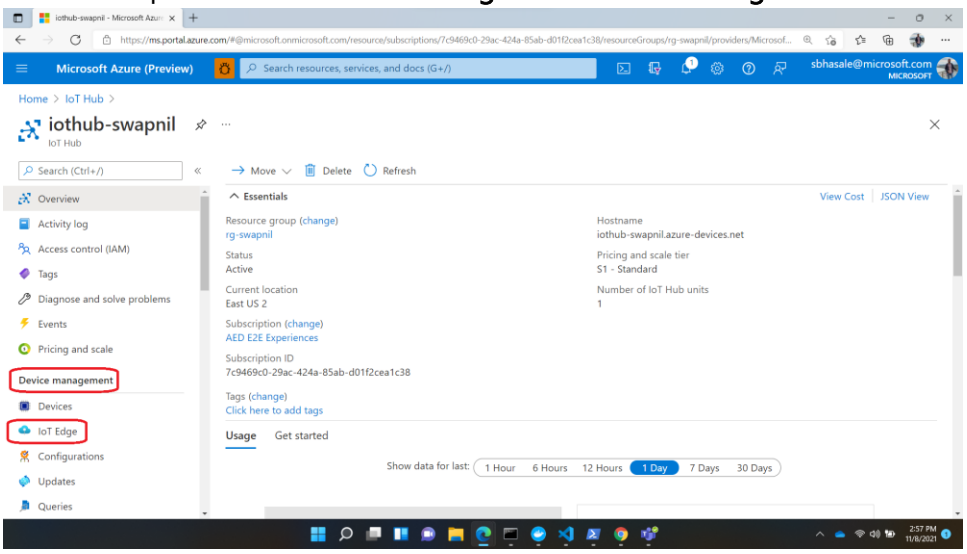


8. The previous step will deploy modules to your device. In the **Notifications** tab click **View Stream**. This will open a new tab in your browser, please verify that you see the video stream. If you point the camera module to a person then you will see the person detection with bounding box

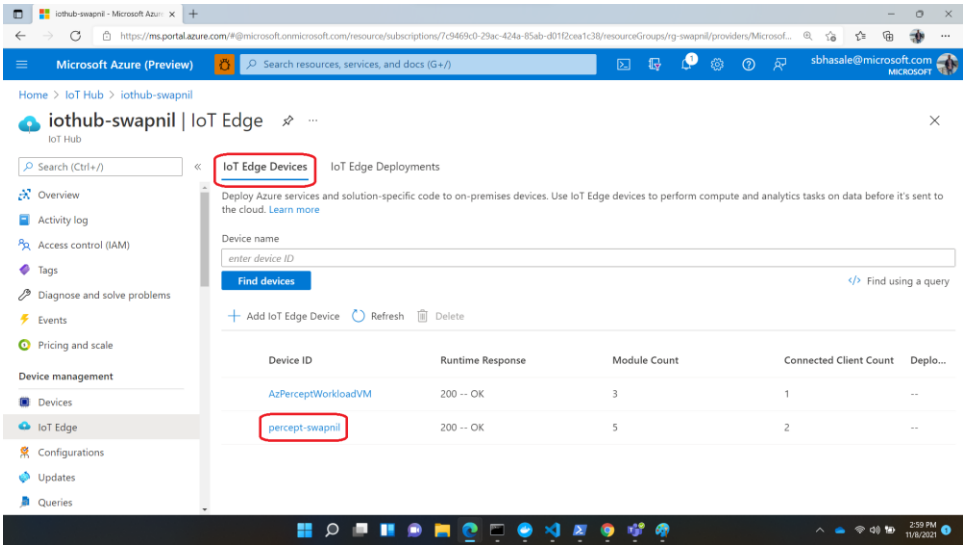


9. After verifying the video stream and bounding boxes, please **close** the **webstream** browser tab.

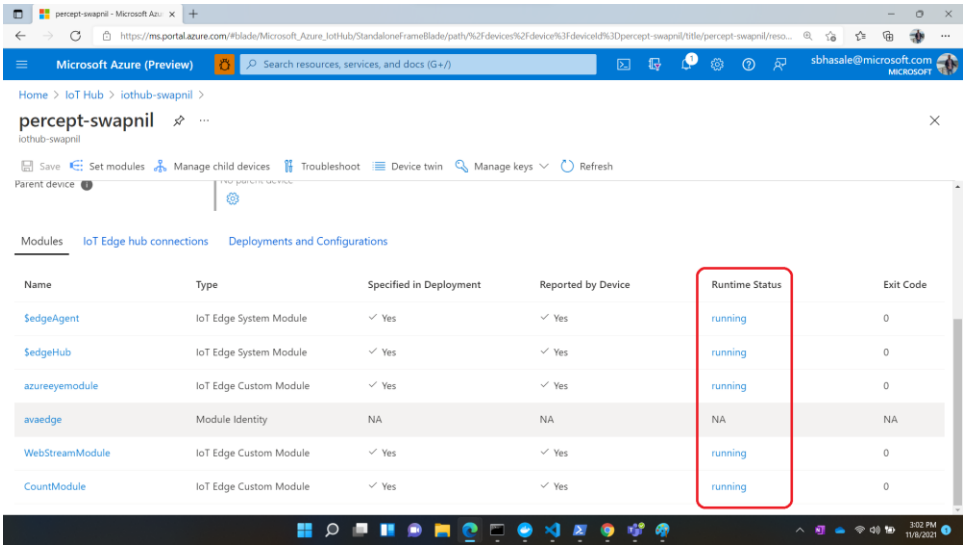
10. To ensure the Count Module is setup correctly, in the Azure Portal go to your IoT Hub. On the left panel under **Device management** select **IoT Edge**



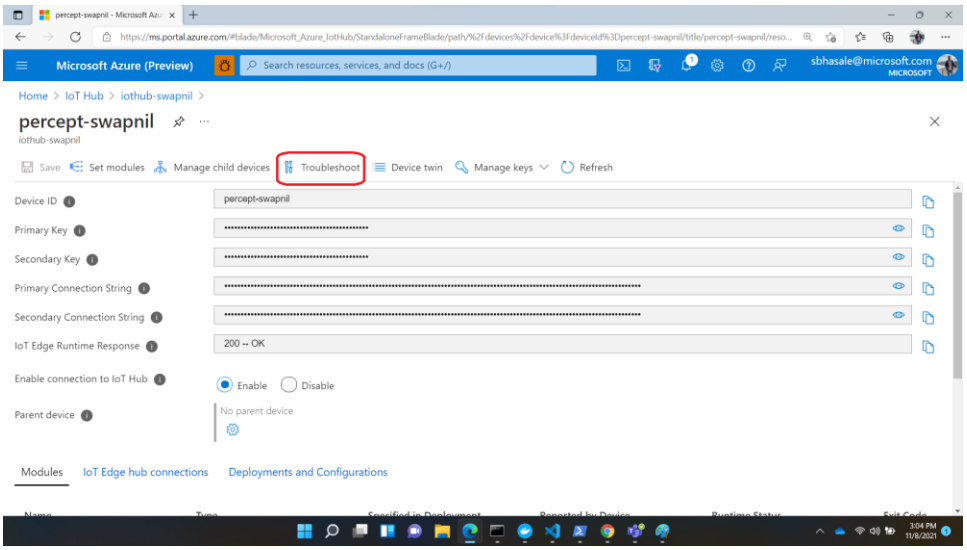
11. From the IoT device list click on your Azure Percept DK device



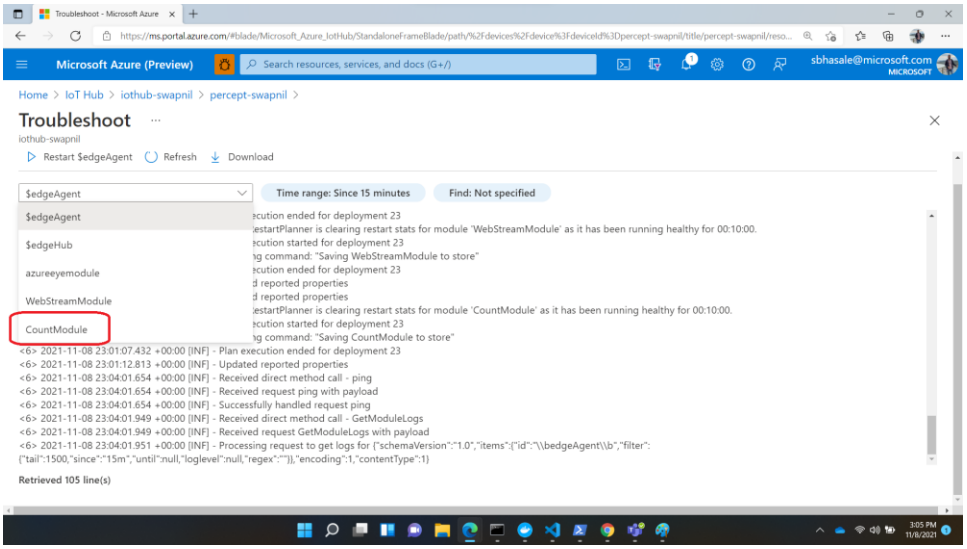
12. Scroll down to check if all deployed modules are in **running** status



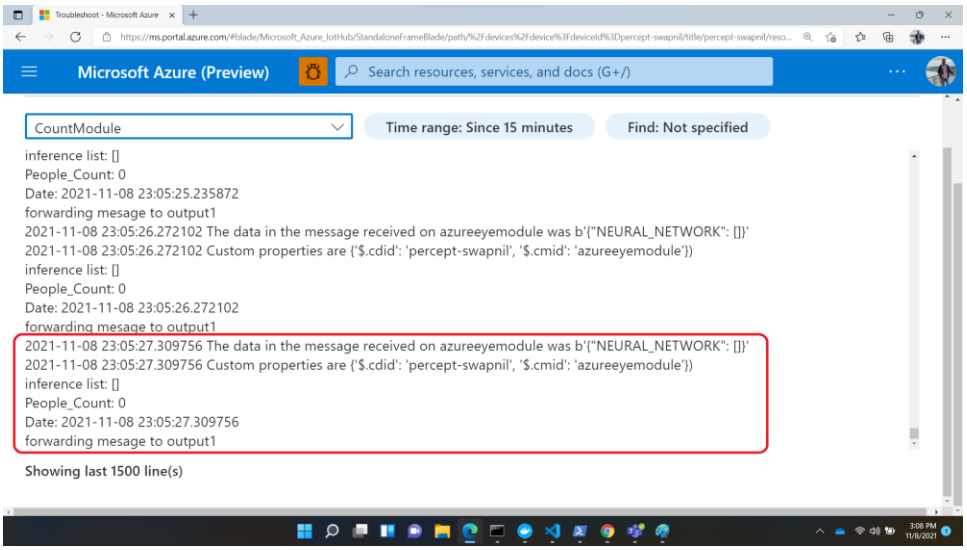
13. Click **Troubleshoot**



14. From the drop-down list select **CountModule**



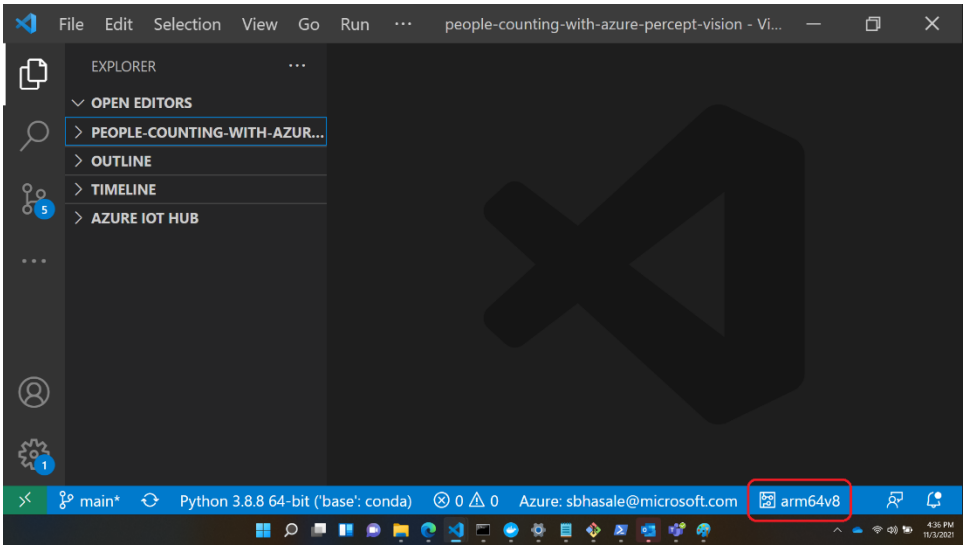
15. Ensure you see **People_Count** logs as follows-



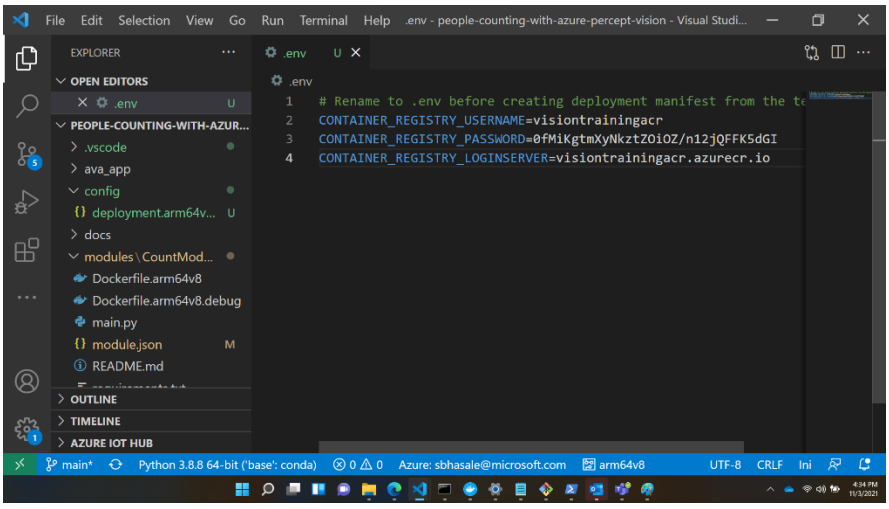
Note- The next task is only for participants who do not have Docker installed. If you have Docker installed, please proceed to Exercise 3.

Ex 2. - Task 3 - Deploy edge modules to device (No Docker steps) (10 min)

1. Open VS Code, at the bottom of the screen ensure you have **arm64v8** as the **Default Platform for IoT Edge Solution** selected (if not, then please click and select **arm64v8** from the list)



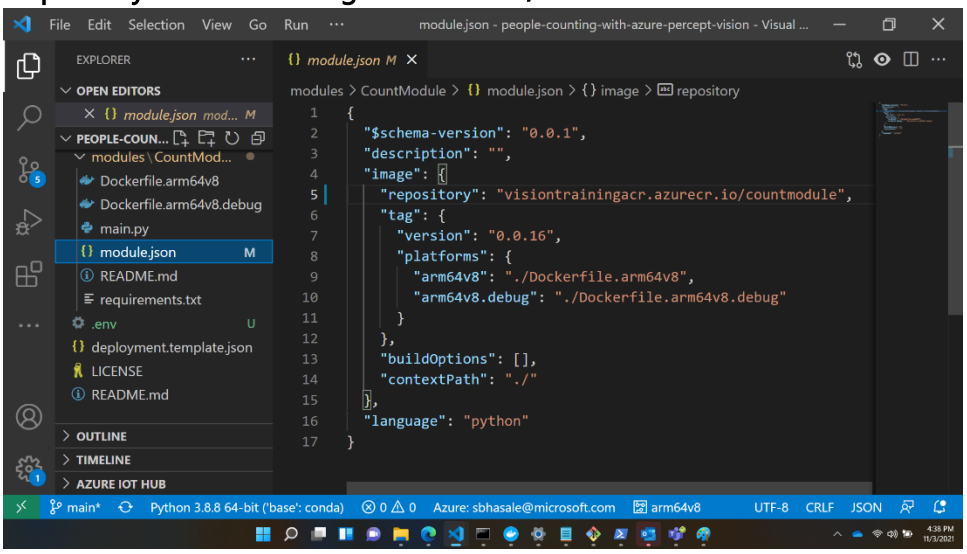
2. Go to the git repository that you cloned in VS Code. Rename the file **envtemplate** to **.env** open the file and fill in the following details
    a. CONTAINER_REGISTRY_USERNAME=visiontrainingacr
    b. CONTAINER_REGISTRY_PASSWORD=0fMiKgtmXyNkztZOiOZ/n12jQFFK5dGI
    c. CONTAINER_REGISTRY_LOGINSERVER=visiontrainingacr.azurecr.io



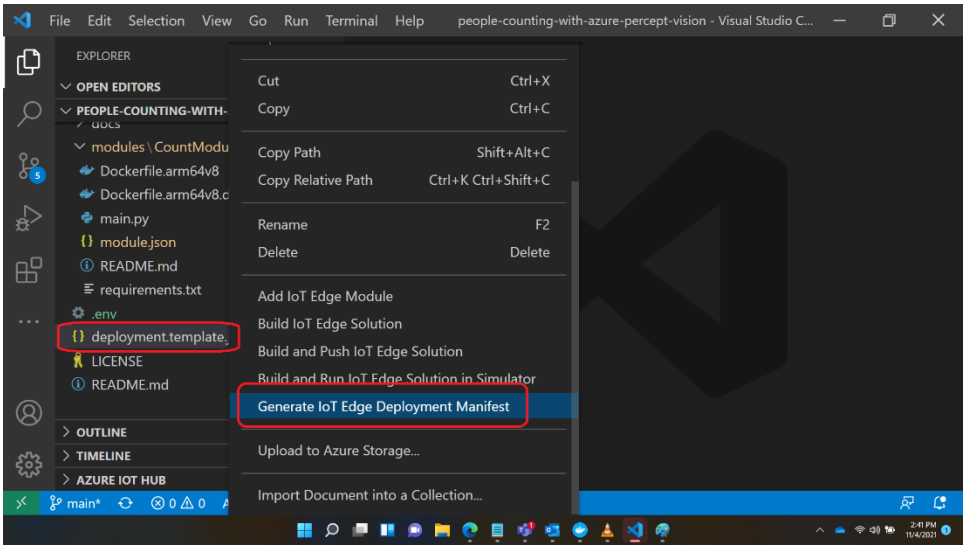
3. In the repository go to **modules/CountModule/** directory and open **module.json**. Fill in your **Container registry address** followed by a **repository name**

   "repository": "<Your container registry login server/repository name>" will change as follows-

   "repository": "visiontrainingacr.azurecr.io/countmodule"



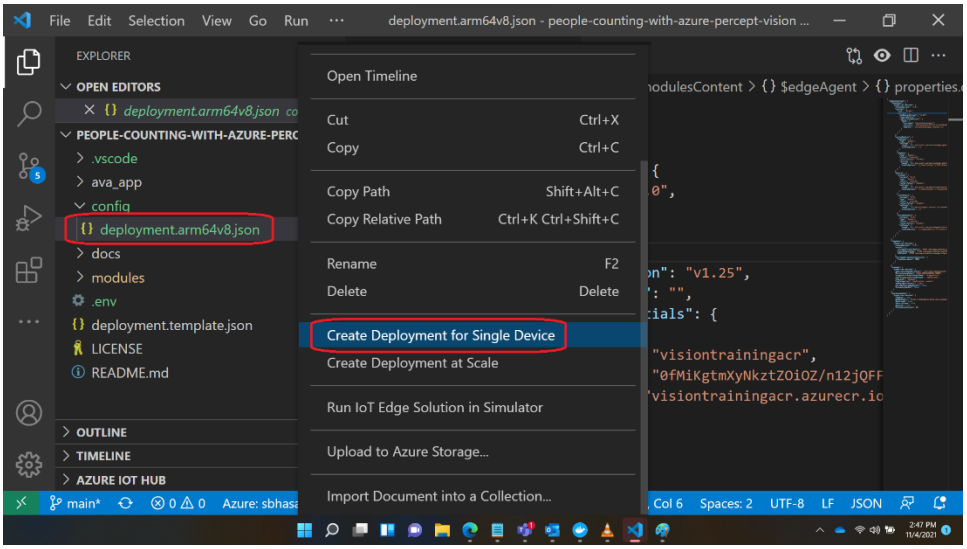
4. Right-click the **deployment.template.json** file and select **Generate IoT Edge Deployment Manifest** option



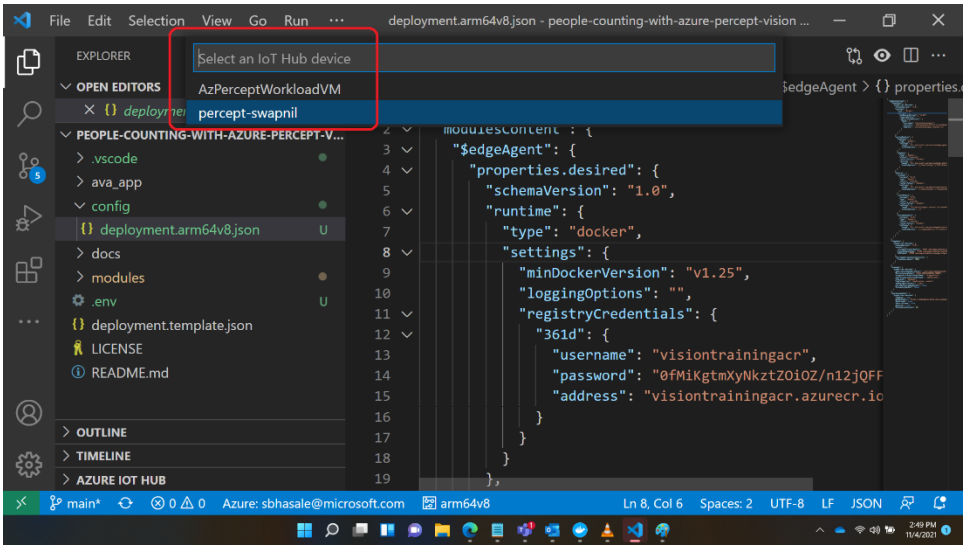
5. The above step will create a **config** folder in your project directory. Within this folder there is a file called **deployment.arm64v8.json**

6. Go to the **config** folder and right-click the **deployment.arm64v8.json** file and select **Create Deployment for Single Device**

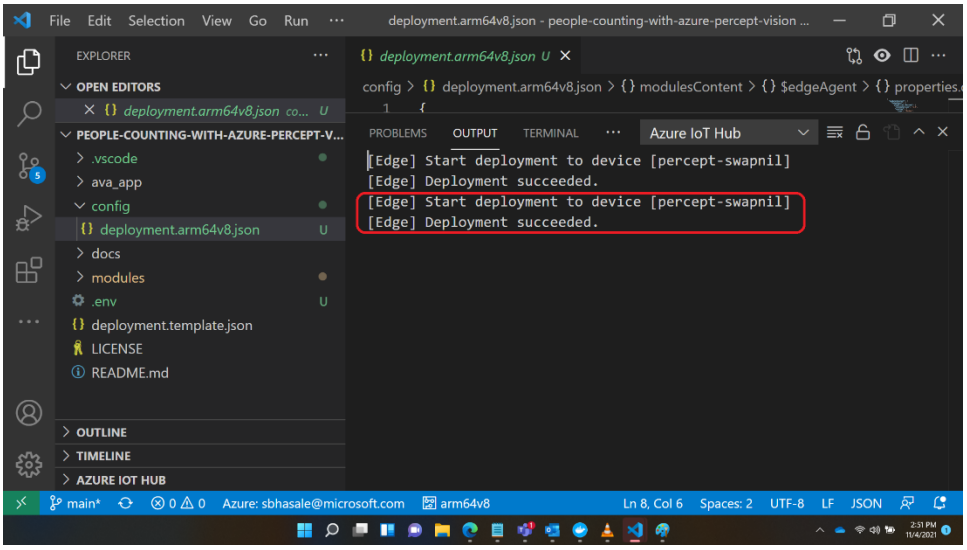
   Do not use the **deployment.template.json** file, which does not have the container registry credentials or module image values in it.



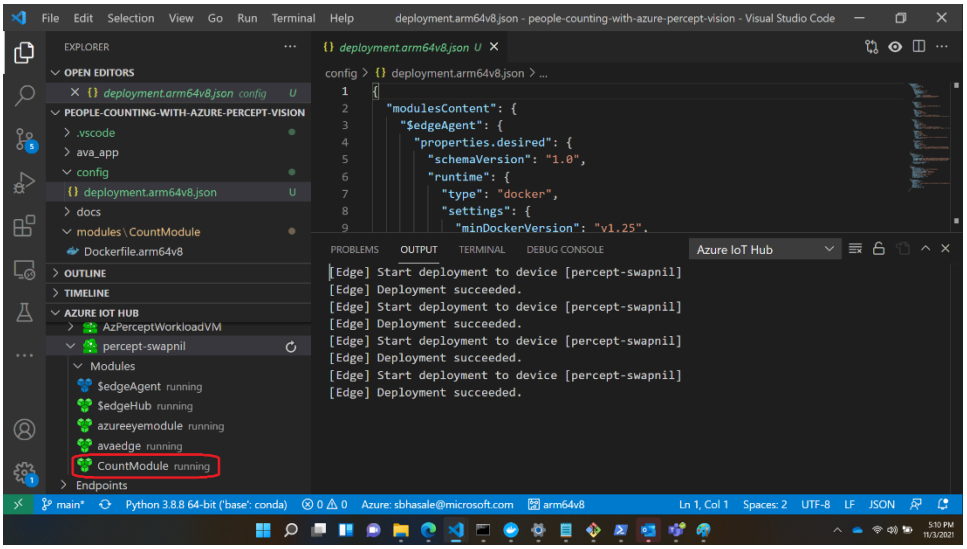
7. Select your IoT hub device



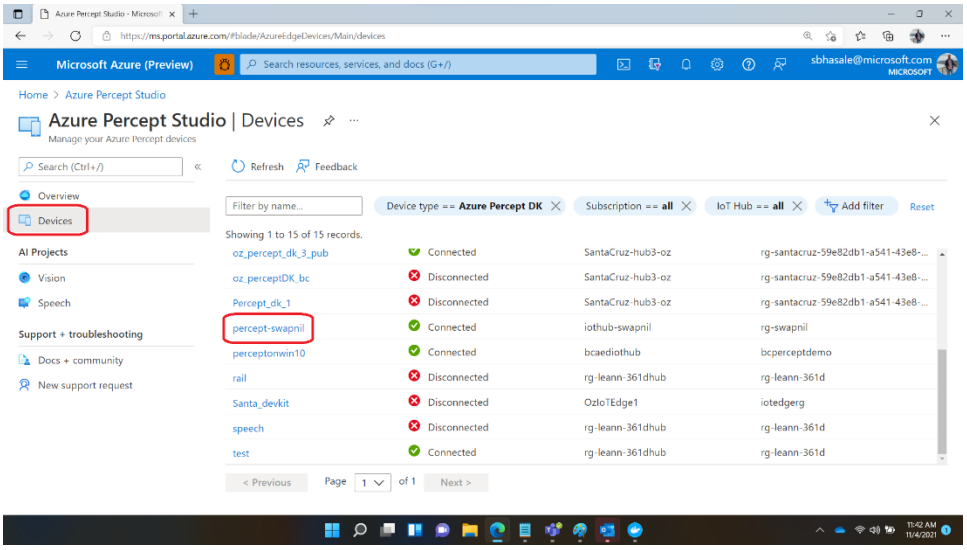
8. Once you select your IoT hub device the deployment will start, ensure that the deployment succeeds in the terminal window



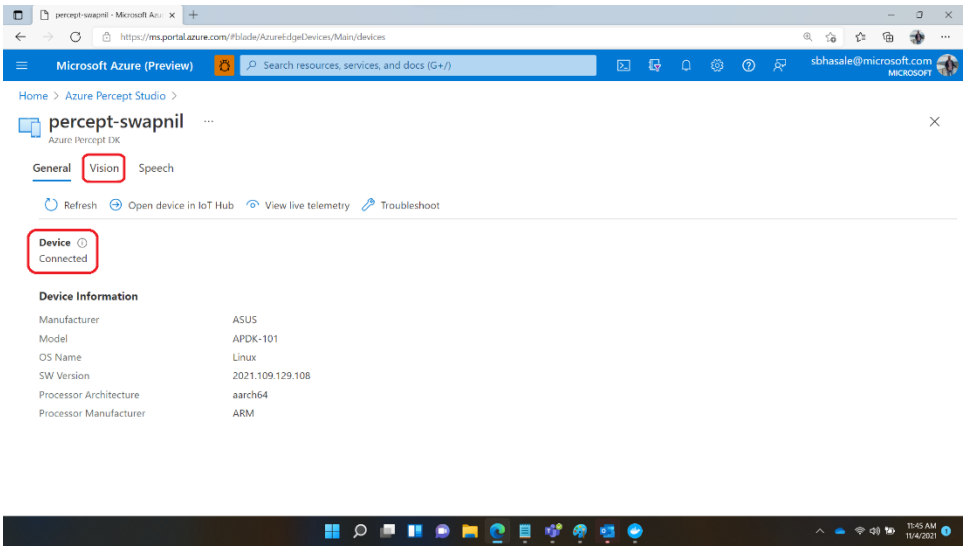
9. Under your device, expand **Modules** to see a list of deployed and running modules. Click the refresh button. You should see the **CountModule** running on your device.



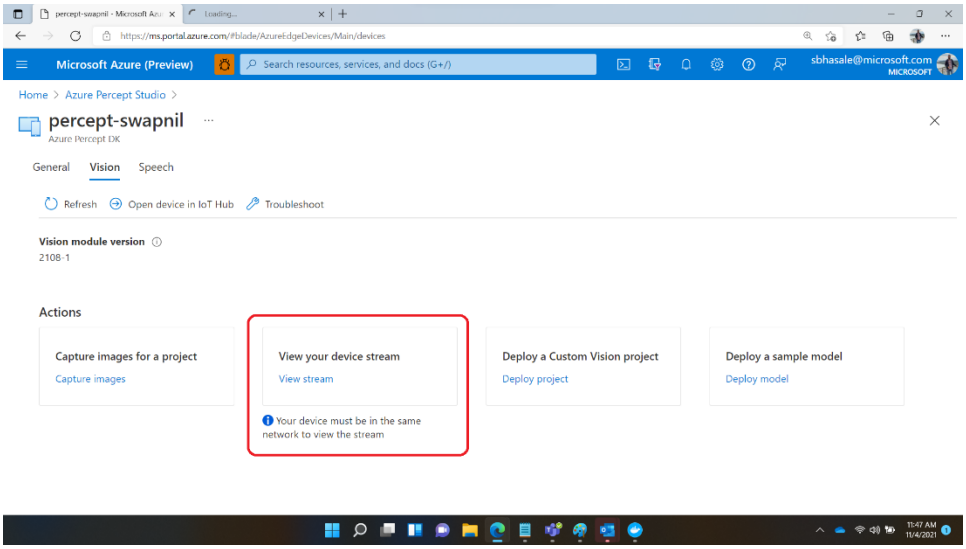
16. Go to Azure Percept Studio and on the left panel, select **Devices**, then select your Azure Percept device



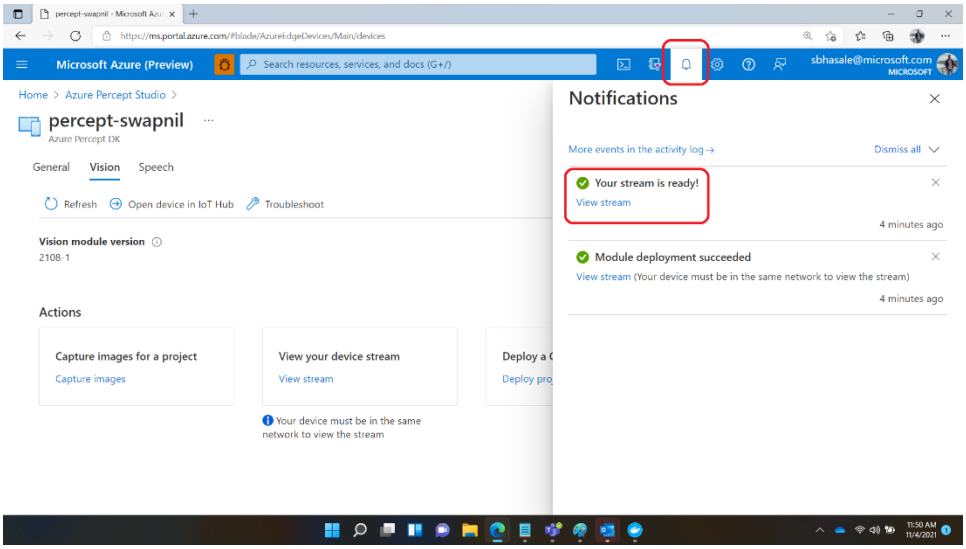
17. Ensure that your device is **Connected**. Click on **Vision**



18. Click View your device stream

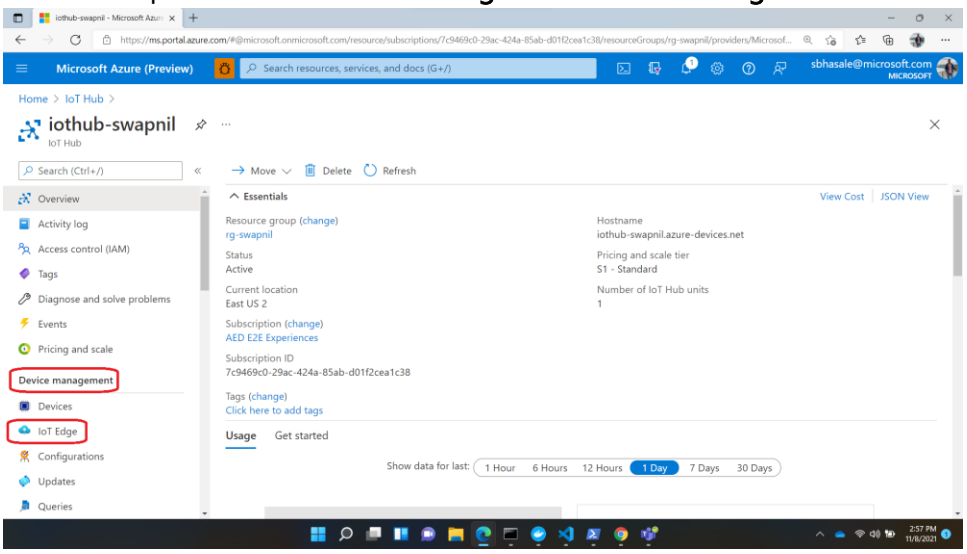


19. The previous step will deploy modules to your device. In the **Notifications** tab click **View Stream**. This will open a new tab in your browser, please verify that you see the video stream. If you point the camera module to a person then you will see the person detection with bounding box

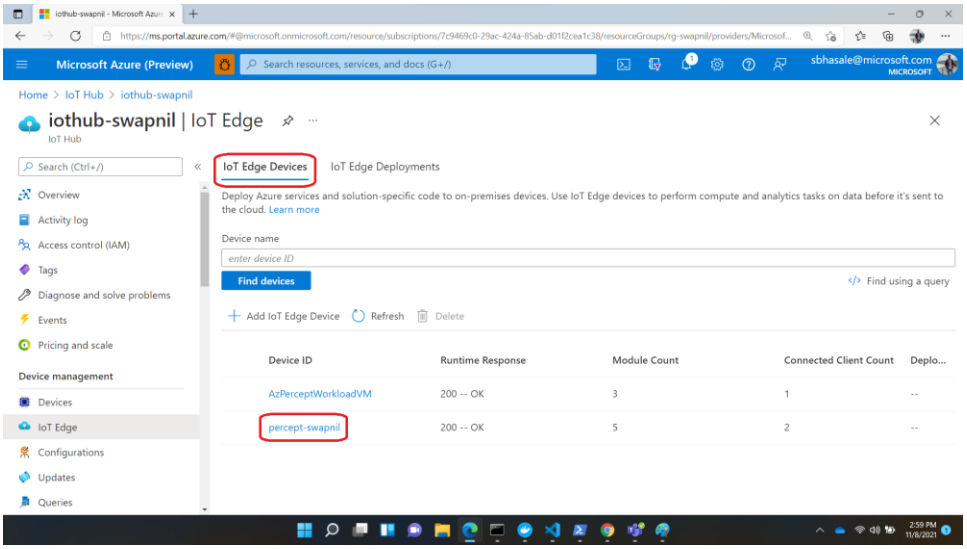


20. After verifying the video stream and bounding boxes, please **close** the **webstream** browser tab.

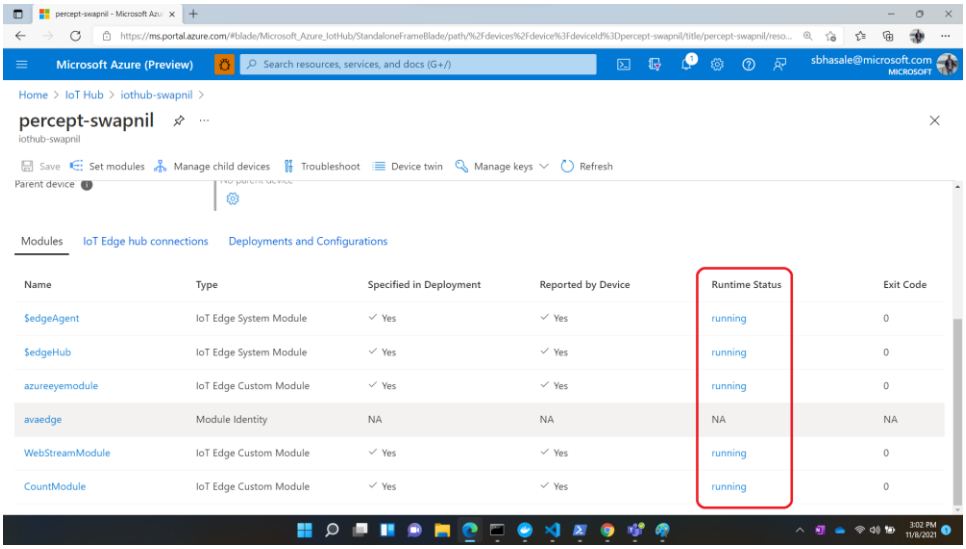
21. To ensure the Count Module is setup correctly, in the Azure Portal go to your IoT Hub. On the left panel under **Device management** select **IoT Edge**



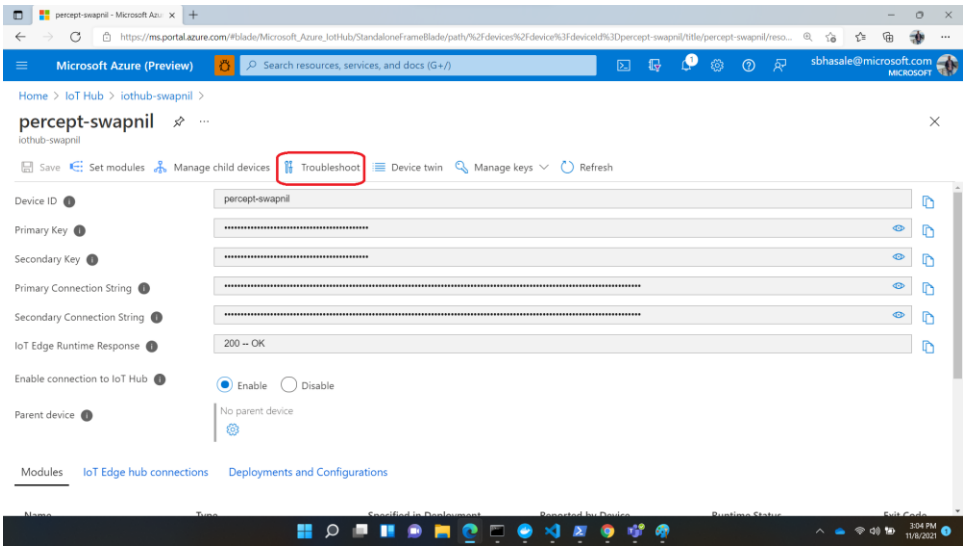
22. From the IoT device list click on your Azure Percept DK device



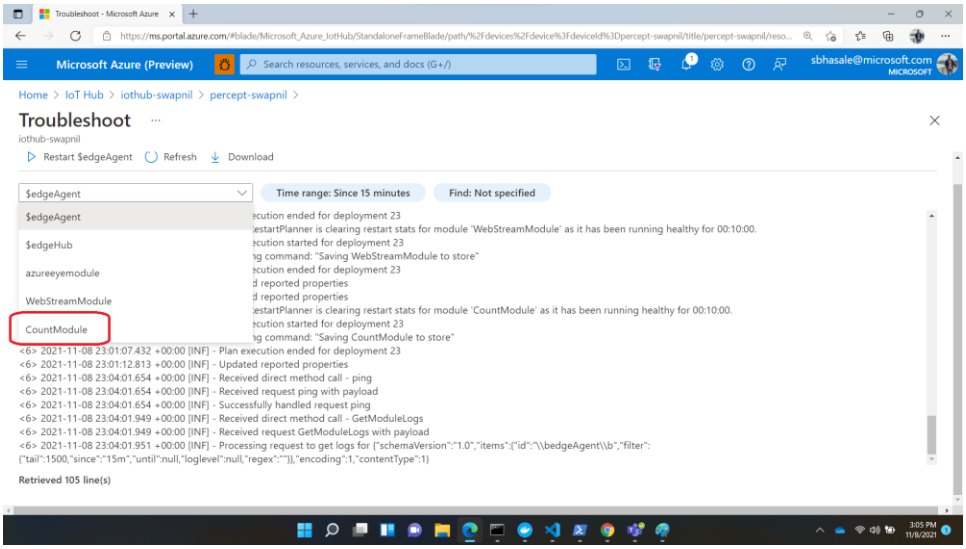
23. Scroll down to check if all deployed modules are in **running** status



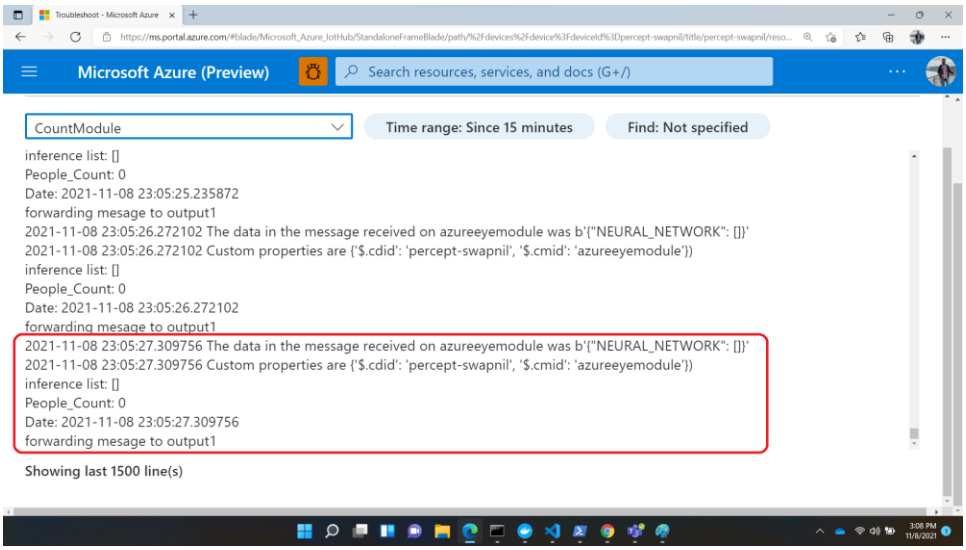
24. Click **Troubleshoot**



25. From the drop-down list select **CountModule**



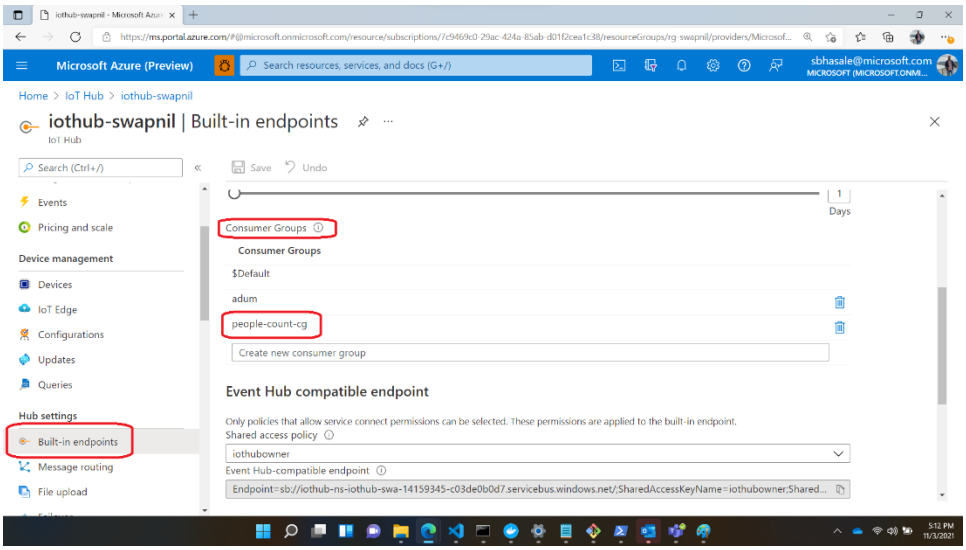
26. Ensure you see **People_Count** logs as follows-



Exercise 3 - Add a consumer group to your IoT hub (5 min)

Consumer groups provide independent views into the event stream that enable apps and Azure services to independently consume data.

1. In the Azure portal, go to your IoT hub
2. On the left pane, select **Hub settings > Built-in endpoints**. Enter a name for your new consumer group in the text box under **Consumer groups**

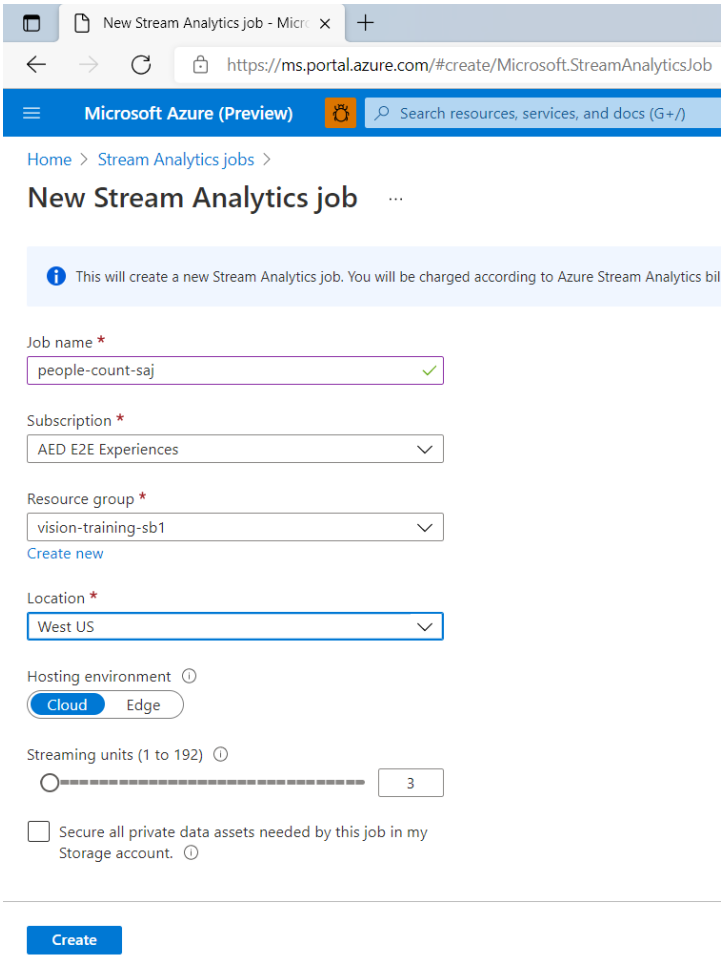


3. Click anywhere outside the text box to save the consumer group

Exercise 4 – Set up Stream Analytics (25 min)

Ex. 4 - Task 1 – Create a Stream Analytics Job (5 min)

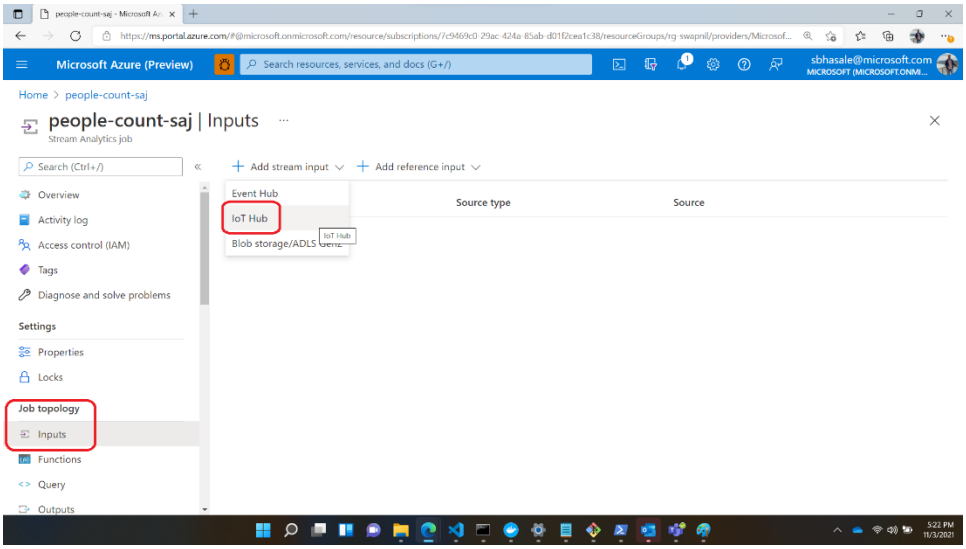
1. Go to [New Stream Analytics job - Microsoft Azure]
2. Enter the following information for the job -
    a. **Job name** - The name of the job. The name must be globally unique.
    b. **Resource group** - Use the same resource group that your IoT hub uses.
    c. **Location** - Use the same location as your resource group.



3. Click **Create**

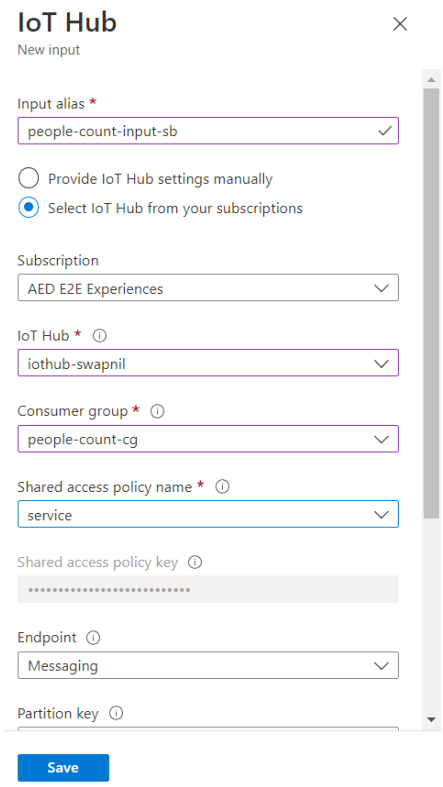
Ex.4 - Task 2 - Add an input to the Stream Analytics job (5 min)

1. Open the previously created Stream Analytics job. Under **Job topology**, select **Inputs**
2. In the **Inputs** pane, select **Add stream input**, then select **IoT Hub** from the drop-down list.



3. On the new input pane, enter the following information -
    a. **Input alias** - Enter a unique alias for the input
    b. **Select IoT Hub from your subscription** - Select this radio button
    c. **Subscription** - Select the Azure subscription you are using for this lab
    d. **IoT Hub** - Select the IoT Hub you are using for this lab
    e. **Consumer group** - Select the consumer group you created previously
    f. **Shared access policy name** - Select the name of the shared access policy you want the Stream Analytics job to use for your IoT hub. For this lab, you can select **service**
    g. **Shared access policy key** - This field is auto filled based on your selection for the shared access policy name
    h. **Endpoint** - Select Messaging

    Leave all other fields as default



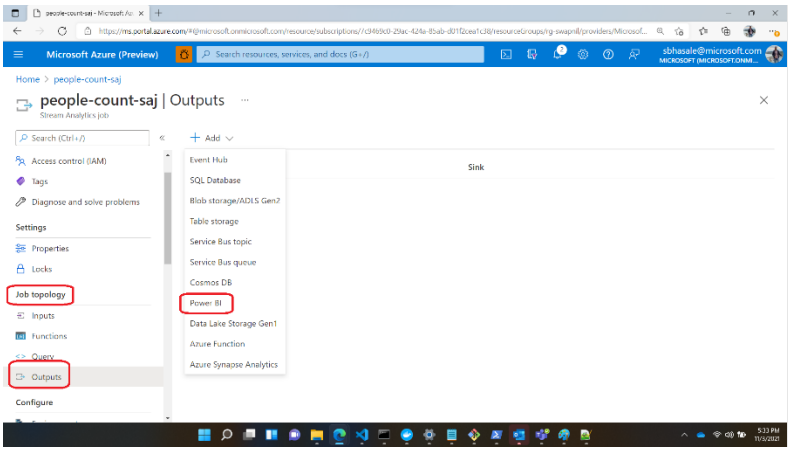
4. Click **Save**

Ex. 4 - Task 3 - Add an output to the Stream Analytics job (5 min)

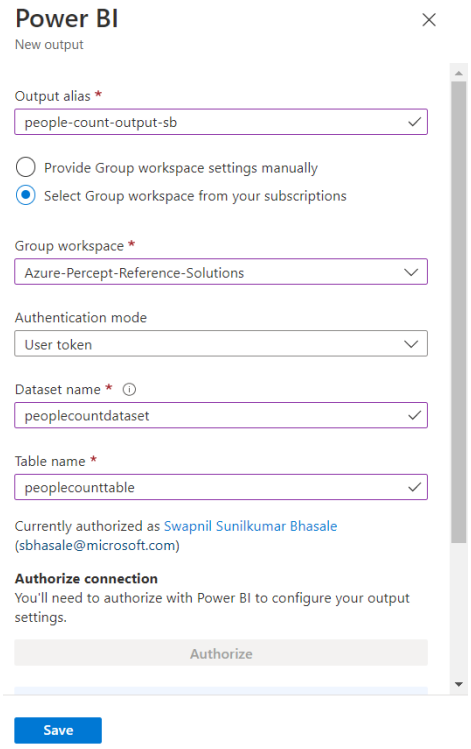
Create a Group Workspace, take the following steps to create one-

a. In a new Web Browser tab open [Power BI]
b. On the left panel click on **Workspaces** > **Create a workspace**
c. Give your workspace a name and description (optional) and click **Save**
d. Go back to the Azure Portal and go to the Stream Analytics job

1. Under **Job topology**, select **Outputs**
2. In the **Outputs** pane, select **Add**, and then select **Power BI** from the drop-down list



3. Enter the following information -
    a. **Output alias** - A unique alias for the output
    b. **Group workspace** - Select your target group workspace.
    c. **Dataset name** - Enter a dataset name
    d. **Table name** - Enter a table name
    e. **Authentication mode** - Leave at the default



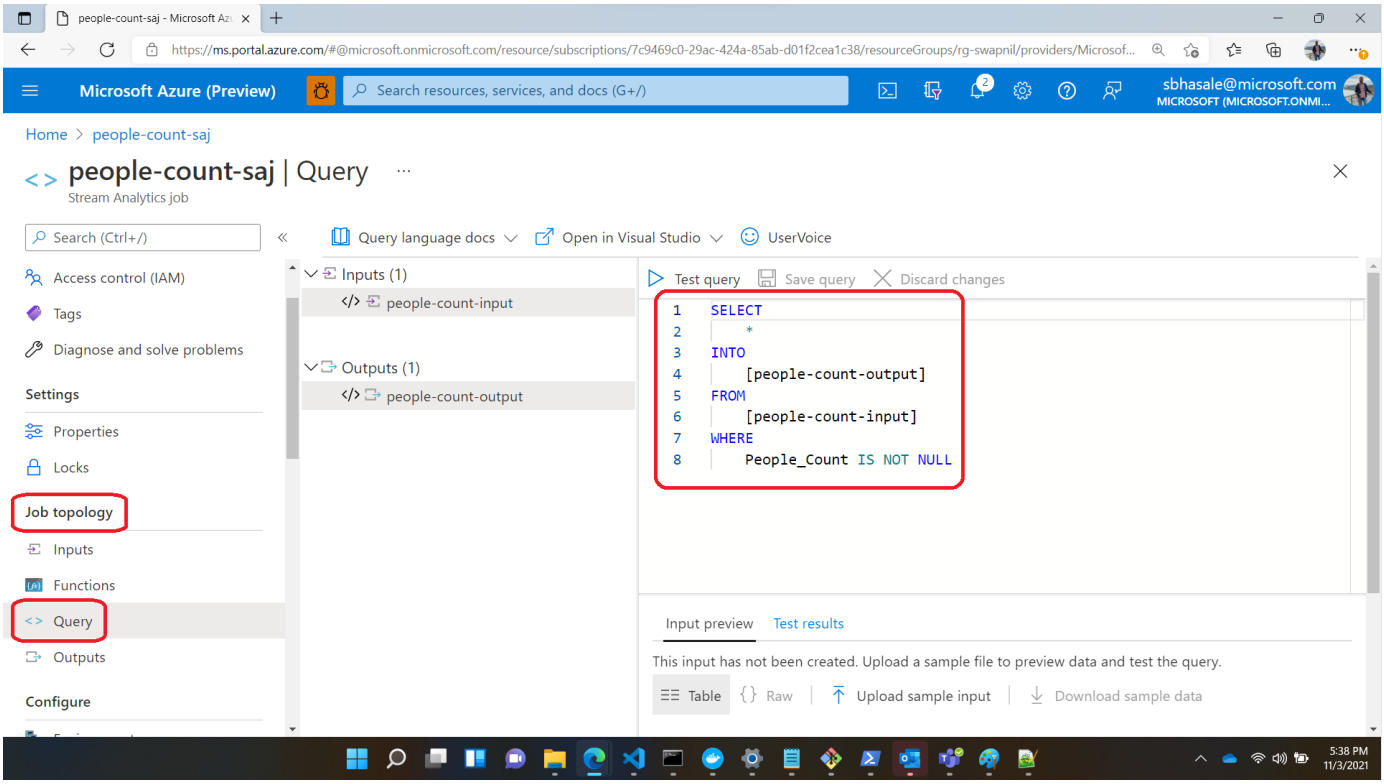
4. On the **Power BI - New output** pane, select **Authorize** and follow the prompts to sign into your Power BI account
5. Click **Save**

Ex. 4 - Task 4 - Configure the query of the Stream Analytics job (5 min)

1. Under **Job topology**, select **Query**
2. Replace **[YourInputAlias]** with the input alias of the job
3. Replace **[YourOutputAlias]** with the output alias of the job
4. Add the following **WHERE** clause as the last line of the query. This line ensures that only messages with a **People_Count** property will be forwarded to Power BI.

   WHERE People_Count IS NOT NULL

5. The query will look as follows -

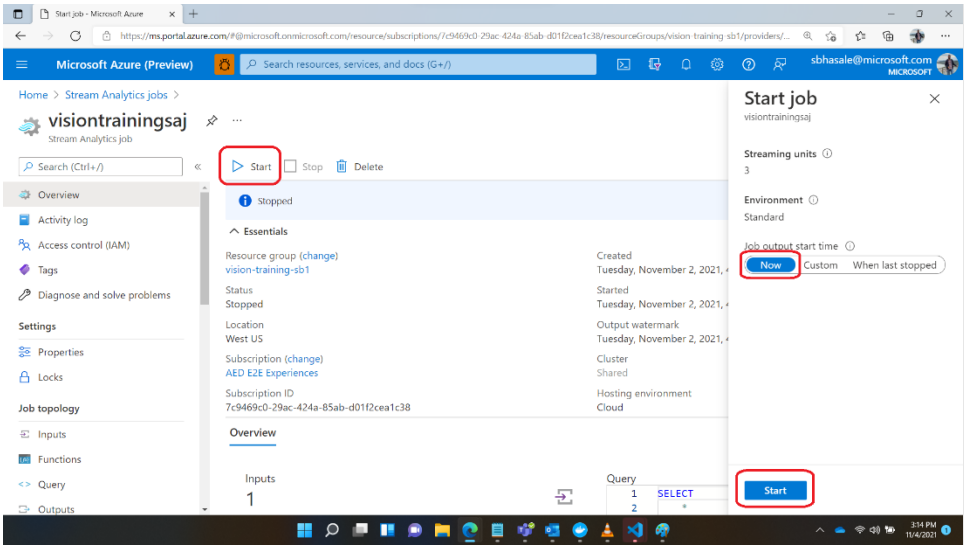


6. Click **Save Query**

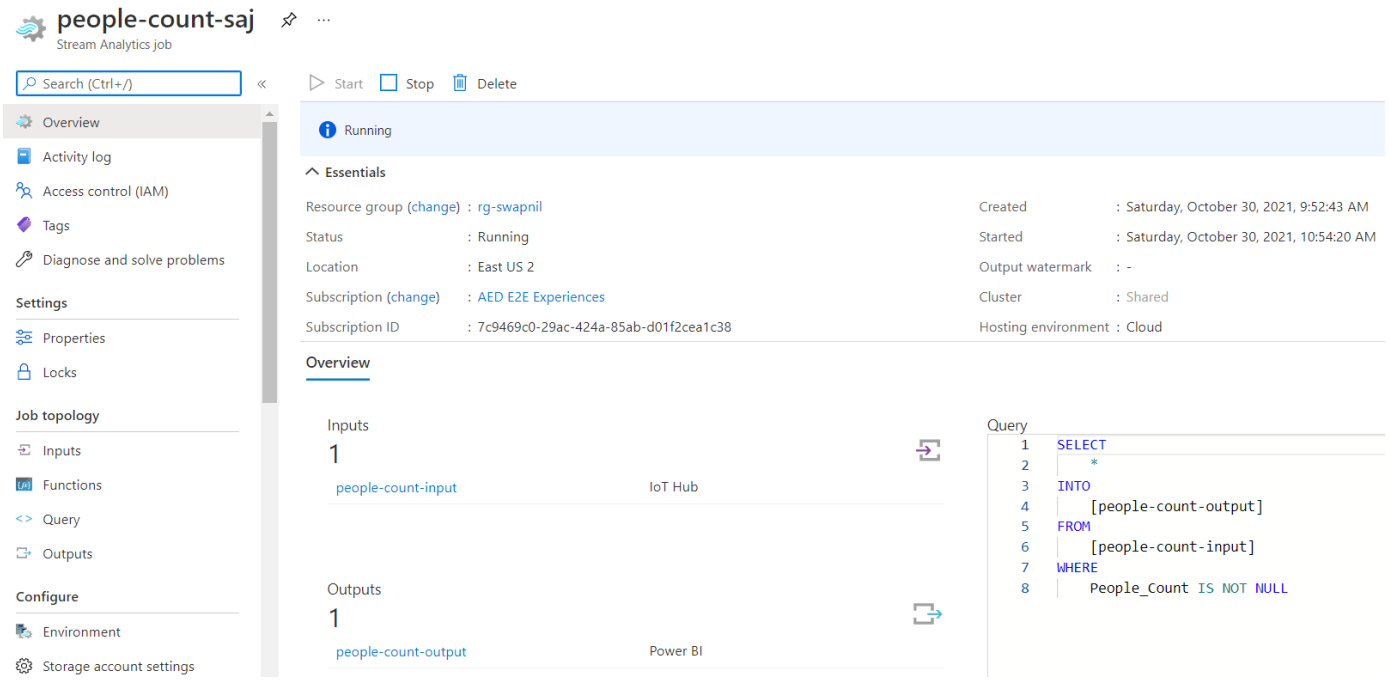
Note- The People_Count property is sent from the countmodule to the IoT hub and is forwarded to the Stream Analytics job.

Ex. 4 - Task 5 - Run the Stream Analytics job (2 min)

1. In the Stream Analytics job, select **Overview**, then select **Start > Now > Start**.

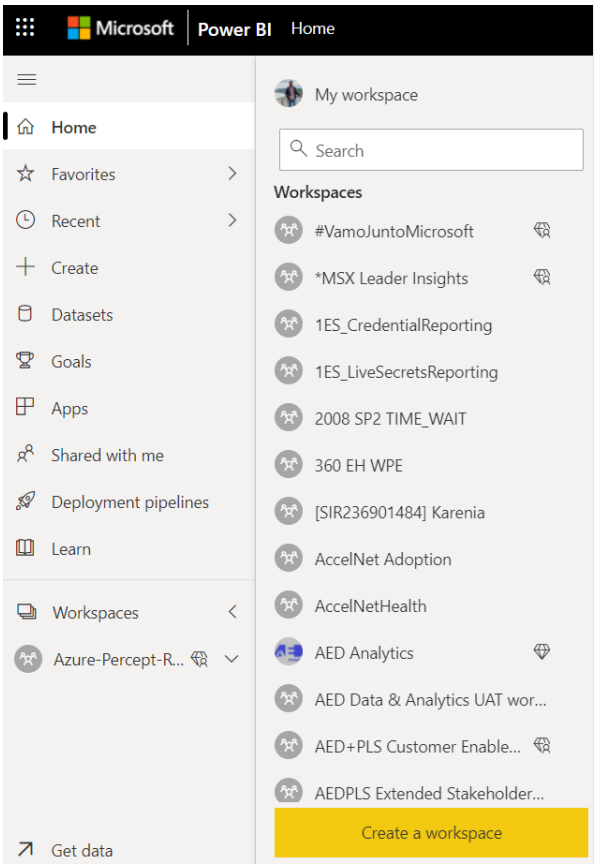


2. Once the job successfully starts, the job status changes from **Stopped** to **Running**

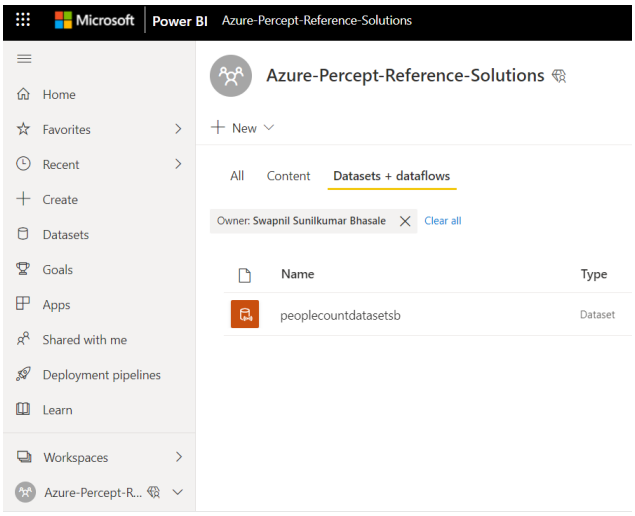


Exercise 5 – Create and publish a PowerBI report to visualize data (5 min)

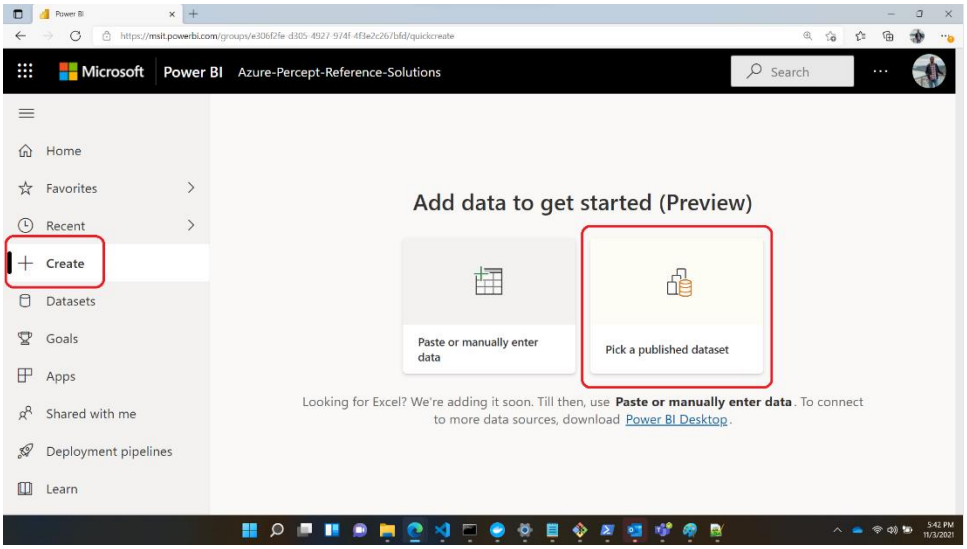
1. Log in to [Power BI] and select your Workspace (this is the same Group Workspace you used while creating the Stream Analytics job output)



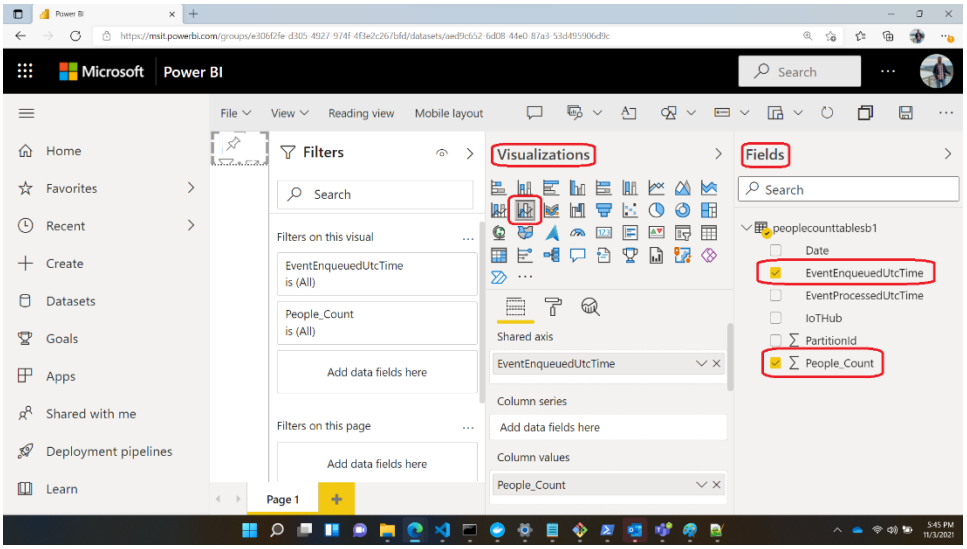
2. Verify that you see your dataset



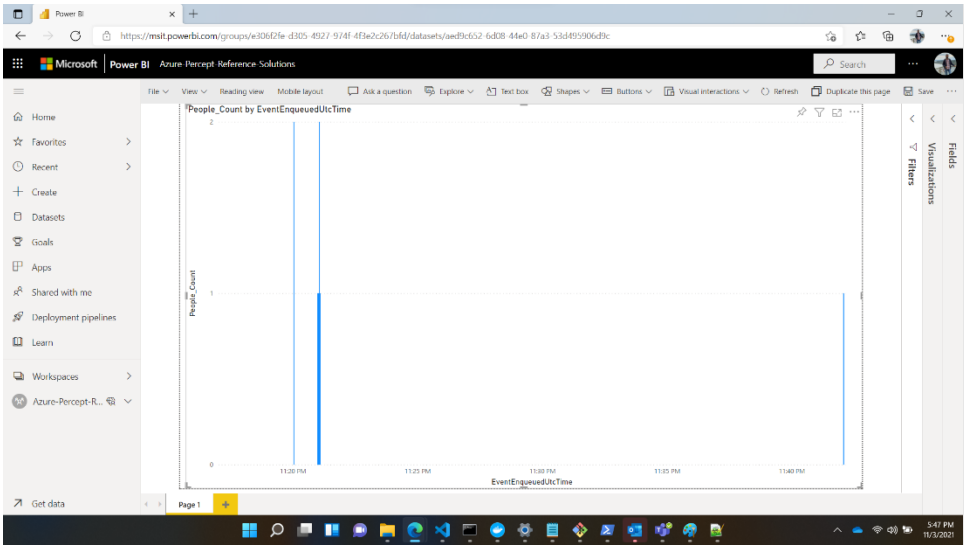
3. On the left scroll panel select **+ Create** and then click **Pick a published dataset**



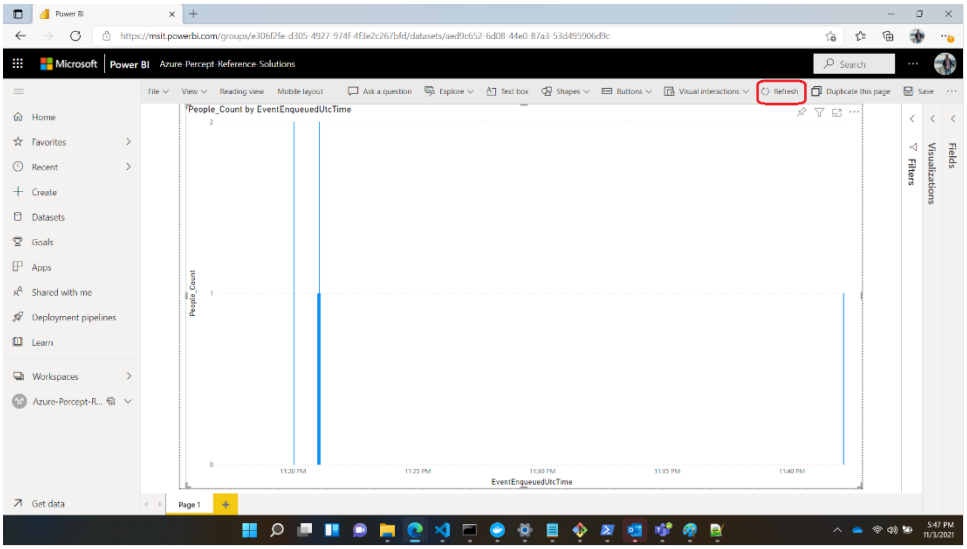
4. Select your dataset and click **Create**
5. On the right, expand the **Fields** dropdown and select **EventEnqueuedUtcTime** and **ΣPeople_Count**
6. Under **Visualizations** select **Line and clustered column chart**



7. This will generate a graph as follows -



8. Click refresh periodically to update the graph



Summary

In this lab you created a people detection and counting solution using the Azure Percept DK and the Azure Percept Vision module. You used PowerBI dashboard to visualize the results.