**Steps for following up during the workshop**

**Create an azure IoT Hub resource:**

1.In the azure main page Go to Create a resource – which will redirect you to azure marketplace

2.In the left pane you will have a list of categories and you need to choose “internet of Things”.

3.This will show list of services and resources pertaining to IoT – Choose the IoT Hub from the list.

4.It will ask you few details which you can enter as below:

1. Basics:

1. Subscription:” Azure for Students” – if you have activated your student benefit or else “pay-as-you-go” if you’re using your own billing service.
2. Resource Group: if you don’t have any you can create a new resource group – these are logical grouping of resources for a project or region however you wish to group together.
3. Region: Choose “West US” which will be the fastest
4. IoT hub name: you can choose something like ‘<yourname>hub’
5. Click on Next: Size and Scale

2.Size and Scale:

1.Choose the F1 tier which will allow you one device and is free of cost. So happy learning without sweating too much of your azure credits.

2.Then click on Review + Create

3.Your IoT Hub resource will spin up in a few minutes.

4.After the deployment is complete click on go to resource

5.In the overview section you can see the resource details.

**Creating a IoT device in Azure IoT Hub:**

1.Click on your IoT hub instance.

2. In the leftmost pane you will see the options called overview, activity log, etc.

3. Scroll down to Explorers and then click on the IoT Devices option or you can search for the same in the leftmost pane using the search tab.

4. Once you are in there you will be able to see the number of devices in your Hub.

5.The limit for one IoT Hub is 1 million devices if you want further then you can contact azure support and then they will provide an increased quota based on request.

6. Click on +New to create a new device on the top bar of the page.

7. You will need to enter a couple of options:

1. Device ID – “pi” or anything of your choice.
2. Authentication type – Symmetric Type
3. Make sure that the Auto-generate keys are enabled
4. Click on Save.

8. After clicking save your first device in IoT Hub will be created.

9. Now after the device is created if you click on the device, then the details about the device will be shown to you like device id, primary key, secondary key, primary connection string, secondary connection string. Please make sure two important settings are fine – Connection to IoT Hub is in enabled state then you will need to use your primary connection string later in you simulated device code.

**Use azure cli to make sure the events are incoming to the Azure IoT service:**

1. Open Azure CLI – Click on the PowerShell like icon on the topmost navigation bar. On Hovering over it will display “Cloud Shell”
2. After clicking on the cloud shell, it will ask you to create a storage account before running any commands.
3. Choose the subscription and click on create storage button.
4. Also, azure will give you the option to open the shell in bash or PowerShell mode you can choose it based on your convenience for this tutorial we will be using the bash shell mode.
5. Once the bash shell is initiated successfully run the following commands:

$ az extension add --name azure-iot

$ az iot hub monitor-events --hub-name <yourhub> --device-id <id>

1. Once this is run, it will display Starting Event Monitor, filtering on device:” <id>”
2. Now you can move over to the git repo and navigate to the simulateddevice.py in the git local repo you cloned.
3. Now in the “CONNECTION\_STRING” variable- copy and paste the primary connection string from your IoT Device that you created.
4. Now create a virtual environment using pip or anaconda.
5. We will use anaconda to create virtual environment

Run “conda create -n azureenv python=3.7”

1. This will create an environement called azureenv with python 3.7 installed.
2. Now to enter the environment type conda activate azureenv
3. Once you are in the environement then install the following packages

pip install azure-iot-device

pip install azure-iot-hub

pip install azure-iothub-service-client

pip install azure-iothub-device-client

1. once these are installed you can now run the python program “simulateddevice.py” to send some sample data to your hub.
2. Navigate to the directory of the git folder and then run python simulateddevice.py
3. This will start sending the messages to you azure iot hub now move over to you cloud shell and now you can see your azure IoT Hub messages.

**Creating an Azure Stream analytics service:**

1. On the azure portal, click on create resource.
2. Search for Azure stream Analytics – this service allows you to handle large amounts of data in real time or you can find this service under the analytics tba in azure.
3. Now click on create.
4. On the New Stream analytics job page enter the following details:
   1. Job name: any arbitrary name
   2. Subscription
   3. Resource Group
   4. Location – East or West US
   5. Hosting environment – Cloud
   6. Streaming units – set to 1
5. Click on create – now just creating this service will not do anything now we have to configure it properly.
6. Now once the service deployment is done go to the overview page – and then in the left navigation pane go to the ‘Job Topology section’ under the section click on inputs and notice the different inputs that are accepted in this view.
7. Similarly look at the output tab and see the different options present in that section too.
8. Now we need to create an input source/Stream and an output sink.
9. For this we require two things alias and the service in an overview.
10. Now for the input stream page:
    1. Click on add new button
    2. Choose the IoT Hub option as we need to read data from the IoT Hub.
    3. Enter an input alias for the IoT Hub – preferably something small as we need to use in SAQL queries.
    4. Then in the radio button options choose Select IoT Hub directly from your subscription.
    5. Choose your subscription
    6. Choose the IoT Hub
    7. Endpoint – Messaging
    8. Leave the rest as defaults – and set the serialization to JSON
    9. Then just click on Save and your Input stream is ready
11. Now we follow similar steps for creating an output sink:
    1. For this exercise we just choose blob storage
    2. Give an output alias – blob
    3. Select Storage from existing subscription
    4. Then it will ask for a container. Since we have not created any containers we will choose the radio option – Create new
    5. And the container authentication mode needs to be set to Connection String and then make sure the serialization patter is in JSON.
    6. Then click on Save.
12. Now let’s create our query to push the data into the blob storage. In the already existing query click on the query and click on edit. Then in the place out output alias enter the blob storage alias and in the place of input alias enter the alias for the IoT Hub.
13. Then click on Save Query
14. Now once it is saved go back to the overview page and then start the job. Then put the start time as now and make sure you are pushing telemetry data from the earlier script that was provided and set up.
15. Now go to your storage account and navigate into containers and wait for the data to be present then you can view your files and then see the data there.
16. Once that you are done with the demo stop the job as you are billed by the minute the job is running and even as files are added into containers.

**Creating an Azure cosmos DB account and pushing data into cosmos DB:**

1. Since we have already set up an input stream now all that we need to add is the output sink and make some changes to our queries.
2. Now before we create an output sink for our cosmos DB, we need to create a cosmos DB account which will be used to create our new database and then add the data into it.
3. Now go back to the azure portal and then click on create resource or use the search bar on the main Navbar of the azure portal.
4. And then search for ‘cosmos DB’ and then click on ‘Azure Cosmos DB’
5. Then click on ‘+ Create’
6. It will take you to a page to choose your API – for this we will choose the Core (SQL) API.
7. And then click on create button under the Core API.
8. In the basics tab enter:
   1. Subscription – as earlier
   2. Resource Group – as earlier
   3. Enter an account name
   4. Location choose West US
   5. The capacity as provisioned mode
   6. Then click on apply Free Tier discount.
   7. Click next – we don’t need to enable any options under Global Distribution
   8. Networking: Connectivity Method: All Networks
   9. Backup Policy change the backup storage redundancy to Locally-redundant backup storage.
   10. Encryption choose service managed Key
   11. Then leave tags empty
   12. Then click on review and create.
   13. Once the validation passes click on create.
9. Once you have created your cosmos DB account let’s move on to the ASA to set your new output sink:
   1. Click on Add
   2. Then from the drop down choose the Cosmos DB
   3. Enter an output alias
   4. In the radio button options choose Cosmos DB from your subscriptions
   5. Then choose your right subscription and Account id
   6. Then in the database click on the option create new and give a name to your database.
   7. Then create new and enter a container name (this can be logically mapped to a table in SQL)
   8. Then click on Save.
10. Once you are done with this now is the time to tinker with the SAQL queries.
11. Now just have a look at your simulateddevice.py code and notice the MSG\_TXT section we now want to pass all that is entered here along with the time the telemetry data is enqueued into Stream analytics.
12. Now enter the following query:

SELECT

    deviceid, messageid, EventEnqueuedUtcTime as time, temperature, humidity

INTO

    <cosmos>

FROM

    <Iothub>

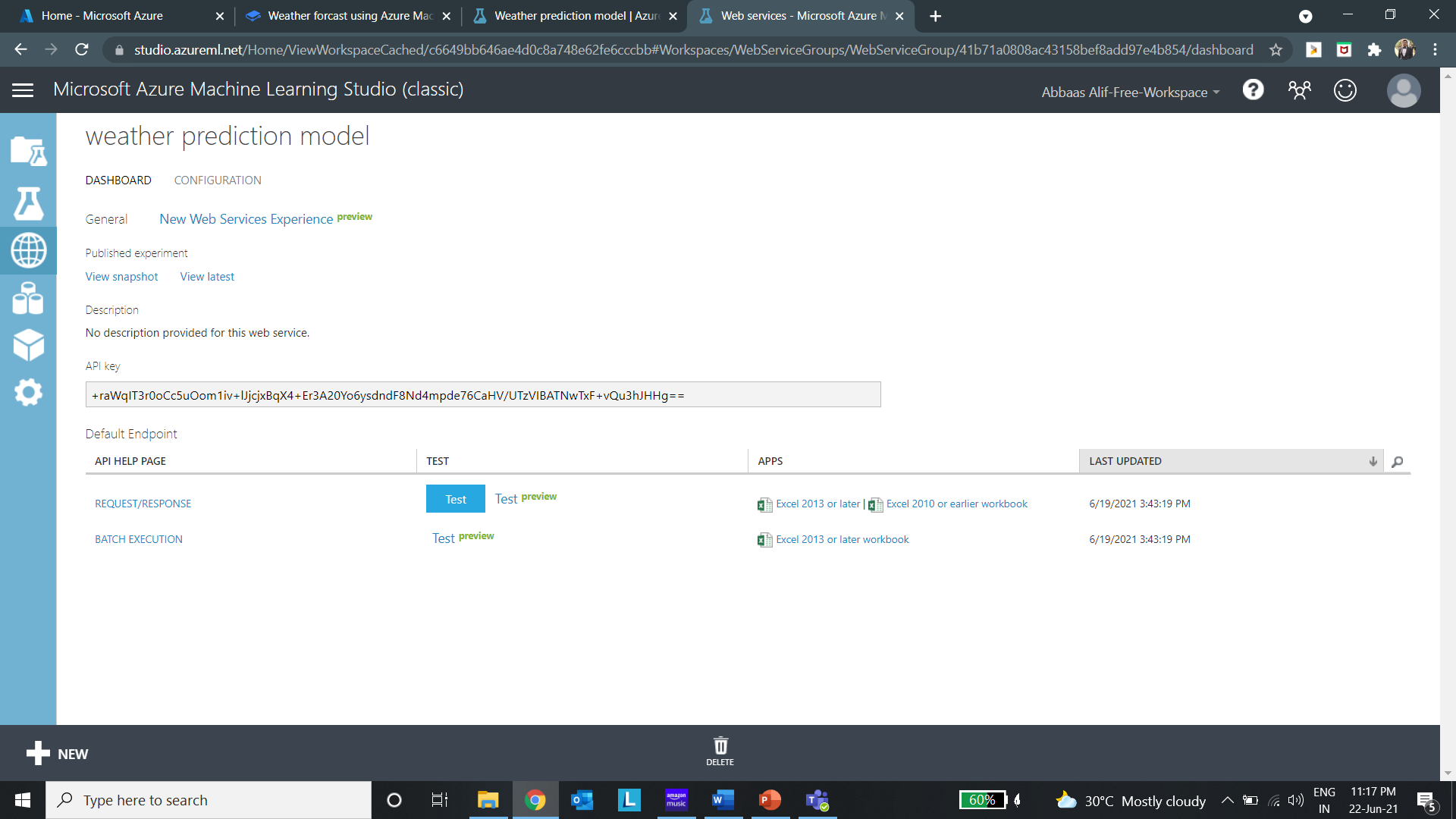
This will post all of the data into your cosmos DB. Now once you are done with this, we can move onto the cosmos DB and query your data with SQL but before all this don’t forget to start you ASA job and don’t forget to stop it once you are done with your work. ‘Time is “literally” money’ in the cloud era.

**Visualizing in Power BI**

1. For visualizations in power BI all you need to do is add another output sink to power BI and you have to add another query for power BI
2. For using power BI, you will have to have an organization email id like your university email address and you cannot use it with your personal email id.
3. Now click on the Add outputs in the Stream analytics service
4. Now from the drop-down list we will choose Power BI
5. Now enter the following details:
   1. Enter an output alias name
   2. In Group workspace choose “My Workspace”
   3. Authentication mode “User Token”
   4. Choose a Dataset name and a table name of your choice
   5. Then Click on the authorize button and enter the Power BI login details. Since this is a SSO service you will have to enter your university mail details to authorize your power BI for Stream Analytics Service.
6. After this, change the output sink alias in the query or copy paste and change below. For the query used above in Cosmos and then start running the job.
7. Now go to power BI and choose a line chart. Put the time in Axis and Temperature and Humidity in Values. This is a simple drag and drop based tool so it is very self-explanatory and easy to navigate around.

**Creating a custom machine learning function(rain prediction based on temperature and humidity) using Azure Machine Learning Studio pushing data to azure blob store**

1. Go to the [weather prediction model page](https://gallery.azure.ai/Experiment/Weather-prediction-model-1).
2. Now you will be redirected to the model page over there click on the Open in Studio button and authorize your azure account.
3. Once you have this you can view the steps that the model is performing. As we can see it performs a two-class logistics regression which is a very basic classification algorithm but the discussion of what is under the hood will not be the scope of this but rather how we can elevate the usage of stream analytics to perform real-time machine learning and provision that will be the aim we are using this.
4. Once this is done you need to hover on the Set-up web service icon in the bottom pane and the choose the deploy web service option from the list.
5. This will take you here:



Under this page you see the request and response and just click on the Excel 2010 or earlier workbook and wait for the download once this download is done open the Excel workbook.

1. Open the workbook and keep a note on the Web service URL and access key
2. One you have this go back to your Stream analytics service. And go the functions sections in the Job topology area.
3. Then click on ADD.
4. Mention the following details:
   1. Function alias
   2. Then click on the ‘Provide Azure Machine Learning function settings manually’ radio button
   3. In the URL section enter you Web service URL from your workbook and in the Key section enter the Access Key details in you
   4. Once you are done with this click on save.
   5. Now navigate to your outputs sections and now we are going to create a blob storage sink like earlier but this time we will use csv files instead of JSON files.
   6. Go to outputs and click on add.
   7. Then choose the blob storage option
   8. Give an output alias.
   9. Then click on Select Blob Storage/ADLS Gen2 from your subscription.
   10. Then choose your storage account
   11. Then create a new container – I will call it raincheck
   12. Then in the authentication mode choose connection string
   13. Leave everything as it is and then in the even serialization option click on CSV – comma (,) and then click on save.
5. Now we have our output sink ready now time to write the query:

WITH MachineLearningStep AS

(

    SELECT

        EVENTENQUEUEDUTCTIME AS time,

        temperature,

        humidity,

        <functionalias>(temperature, humidity) as result

    FROM

        <input>

)

SELECT

    time,

    CAST(result.[temperature] as FLOAT) as temparature,

    CAST(result.[humidity] as FLOAT) as humidity,

    CAST (result.[Scored Probabilities] AS FLOAT ) AS 'probabality of rain'

INTO <output>

FROM machinelearningstep

1. Now with this query in place we need to click on the start Job and then go the container in your storage account and then you will see the probabilities of rain based on the temperature value and humidity value.

**Additional take home content:**

Visualization using the Web App:

1. Power BI is not freeware though it is very easy to use it costs $10 dollars a month for each pro user or for large organization they provide a premium workspace. Around 20 dollars per user to 4,995 dollars per month per capacity of VM.
2. So, we can see how to create a web app using the azure IoT Hub data which is written by Microsoft. The links for this are provided [here](https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-live-data-visualization-in-web-apps).
3. I have also provided a doc in the same repo called web app instruction which you can follow to deploy the same on Azure Web app service (If you have doubts on this or any part of the content, please feel free to raise that as an issue in GitHub and I will try to guide you all to the best of my knowledge)
4. If you want to host the web app on your system then use docker and then navigate to the webapp folder in the repo and find the env-file.txt then replace the values of the IoTEventHub and Consumer group based on the steps for web app doc and then run:

$ docker build -t nodedocker .

$ docker run -d -p 3000:3000 –env-file env-file.txt nodedocker

Now navigate to localhost:3000 and your web page is up and running.

**Azure Anomaly Detection inbuilt function – link and script can be found in scripts.txt after you follow the documentation:**

1. This is also a very interesting topic to learn where you can use azure in built machine learning function for anomaly detection – read the docs [here](https://docs.microsoft.com/en-us/azure/stream-analytics/stream-analytics-machine-learning-anomaly-detection).
2. I have written the script and found the hyperparameters for this exercise which you can view use:

WITH AnomalyDetectionStep AS

(

    SELECT

        deviceid,

        messageid,

        EVENTENQUEUEDUTCTIME AS time,

        temperature,

        humidity,

        AnomalyDetection\_SpikeAndDip(CAST(temperature AS float), 95, 30, 'spikesanddips')

            OVER(LIMIT DURATION(second, 30)) AS SpikeAndDipScores

    FROM <input>

)

SELECT

    deviceid,

    messageid,

    time,

    temperature,

    humidity,

    CAST(GetRecordPropertyValue(SpikeAndDipScores, 'Score') AS float) AS

    SpikeAndDipScore,

    CAST(GetRecordPropertyValue(SpikeAndDipScores, 'IsAnomaly') AS bigint) AS

    IsSpikeAndDipAnomaly

INTO <output>

FROM AnomalyDetectionStep;