**PHASE 4:**

**In this technology project you will continue building your project by developing the platform as per project requirement. Use web development technologies wherever needed. After performing the relevant activities create a document around it and share the same for assessment.**

**ABOUT AZURE:**

Azure is Microsoft's cloud computing platform offering a broad range of cloud services, including infrastructure, platforms, and software. It enables businesses to build, deploy, and manage applications and services in Microsoft-managed data centers. Azure supports virtual machines, databases, AI, IoT, and more, with global data center coverage, robust security, and a pay-as-you-go pricing model. Its scalability, developer tools, and integration with Microsoft services make it a popular choice for organizations seeking cloud solutions to drive innovation, streamline operations, and meet their digital transformation needs.

**Code to display Real time Transit Information azure:**

pip install requests

import requests

import azure.functions as func

def main(req: func.HttpRequest) -> func.HttpResponse:

# Replace with your actual transit API endpoint

api\_endpoint = "https://your-transit-api.com/realtime"

# Replace with your API credentials if needed

api\_headers = {"Authorization": "Bearer YOUR\_API\_KEY"}

response = requests.get(api\_endpoint, headers=api\_headers)

if response.status\_code == 200:

transit\_data = response.json()

return func.HttpResponse(transit\_data, mimetype="application/json")

else:

return func.HttpResponse("Failed to retrieve transit data", status\_code=500)

**Code for Interfacing Data with Azure:**

import machine

import utime

from azure.iot.device import IoTHubDeviceClient, Message

# Define Azure IoT Hub connection string and device ID

CONNECTION\_STRING = "HostName=TrafficManagement.azure-devices.net;DeviceId=RaspPi;SharedAccessKey=7ryCImqSUWqhp5ChKRrkZxNCwpjqtJHL2AIoTGkwJBs="

DEVICE\_ID = "RaspPi"

# Initialize the IoT Hub client

client = IoTHubDeviceClient.create\_from\_connection\_string(CONNECTION\_STRING)

# GPIO pins for the HC-SR04 sensor

trigger\_pin = machine.Pin(2, machine.Pin.OUT)

echo\_pin = machine.Pin(3, machine.Pin.IN)

# Traffic light control pins (simulated)

red\_light = machine.Pin(10, machine.Pin.OUT)

yellow\_light = machine.Pin(11, machine.Pin.OUT)

green\_light = machine.Pin(12, machine.Pin.OUT)

def measure\_distance():

trigger\_pin.value(0)

    utime.sleep\_us(2)

    trigger\_pin.value(1)

    utime.sleep\_us(10)

    trigger\_pin.value(0)

    while echo\_pin.value() == 0:

        pulse\_start = utime.ticks\_us()

    while echo\_pin.value() == 1:

        pulse\_end = utime.ticks\_us()

    pulse\_duration = utime.ticks\_diff(pulse\_end, pulse\_start)

    distance = (pulse\_duration \* 0.0343) / 2  # Speed of sound is approximately 343 meters per second

    return distance

def control\_traffic\_lights(distance):

if distance < 10:

red\_light.value(0)

yellow\_light.value(1)

green\_light.value(0)

elif 10 <= distance < 20:

red\_light.value(1)

yellow\_light.value(0)

green\_light.value(0)

else:

red\_light.value(0)

yellow\_light.value(0)

green\_light.value(1)

while True:

distance = measure\_distance()

control\_traffic\_lights(distance)

# For simulation purposes, print the distance and the traffic light state

print("Distance: {:.2f} cm".format(distance))

# Send distance data to IoT Hub

telemetry\_data = {"distance\_cm": distance}

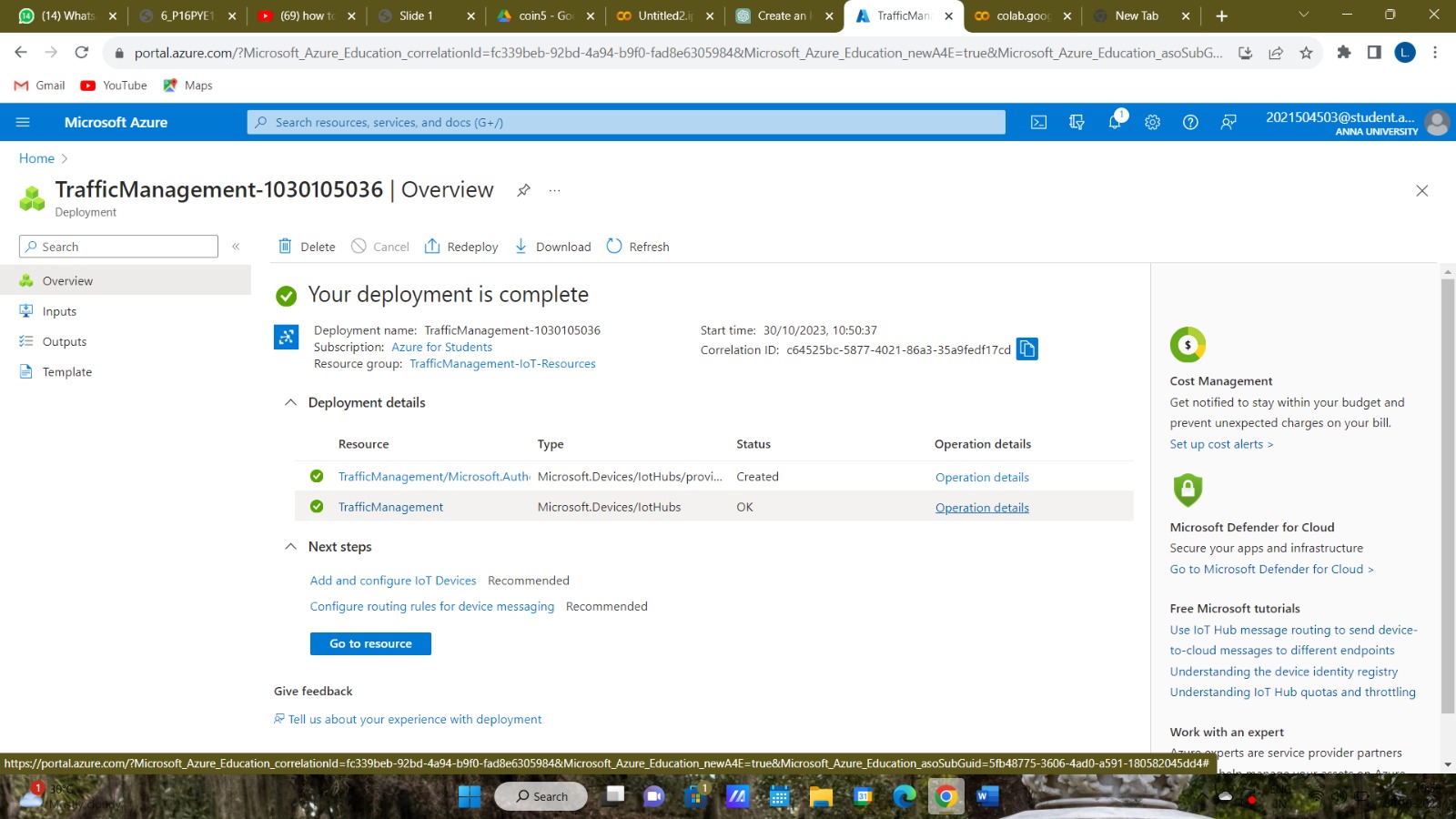
message = Message(telemetry\_data)

client.send\_message(message)

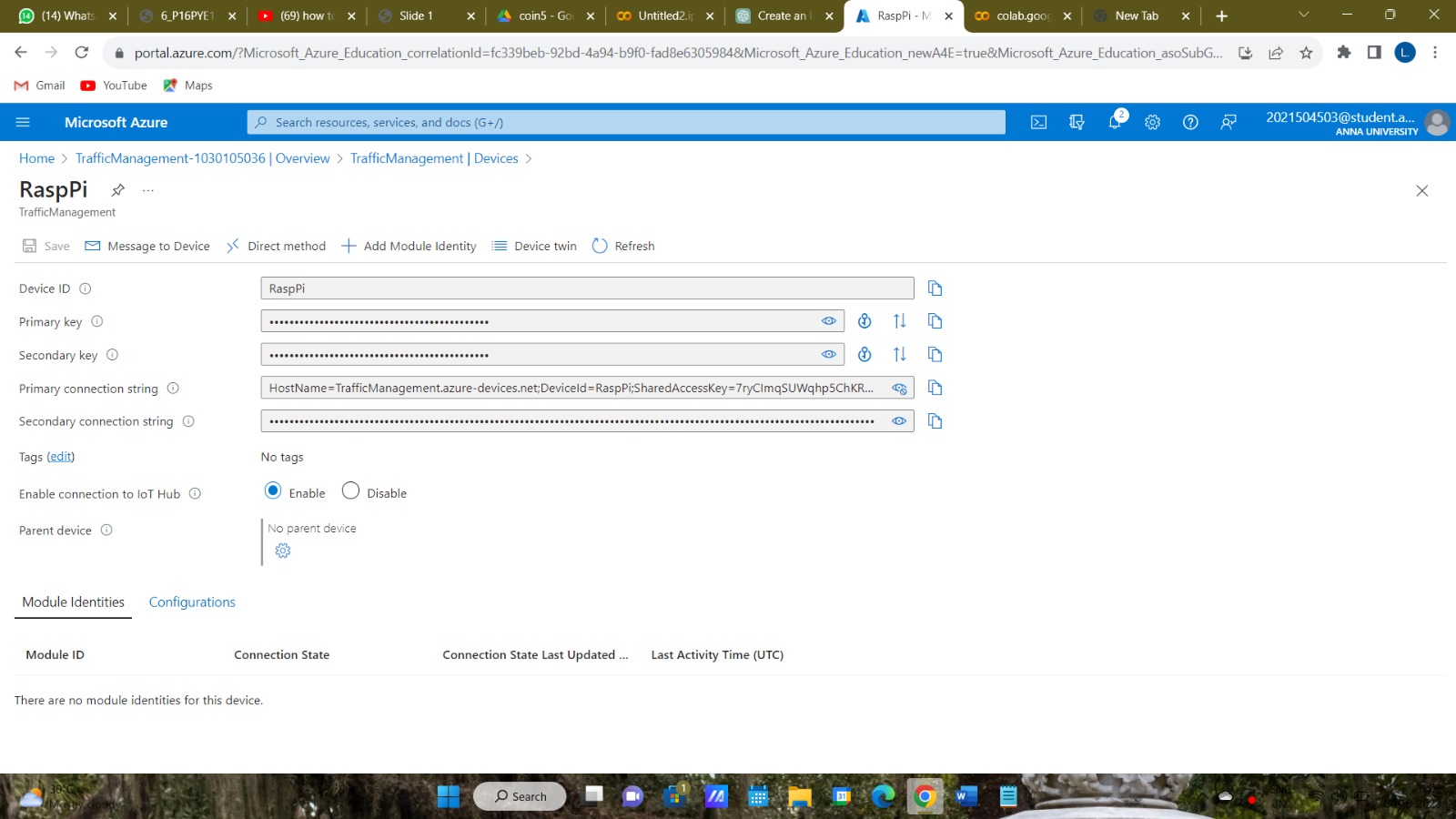
utime.sleep(2) # Adjust the sleep duration as needed

**Steps we followed while working with Azure:**

1. We created an IoT Hub in Microsoft Azure and Deployed it.



1. We created a Device Id, RaspPi and created a Primary Connection String, by using this, we interfaced it with our Wowki Code, to establish the web application:



**DeviceID:** RaspPi

**Primary Connection string**:HostName=TrafficManagement.azure-devices.net;DeviceId=RaspPi;SharedAccessKey=7ryCImqSUWqhp5ChKRrkZxNCwpjqtJHL2AIoTGkwJBs=

The remaining step is to integrate this code in wokwi, importing azure.iot.device which is already installed using the command**:**

**pip intall azure-iot-devices**