Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya

DEVLOP A WEB-BASED APPLICATION USING NODE-RED

SOFTWARE REQUIREMENTS

The web-based application is deployed by consolidating the essential requirements to built IOT web-based platform. The following credential are needed to deploy the web application using NODE-RED which are stated as below

1. Node-Red

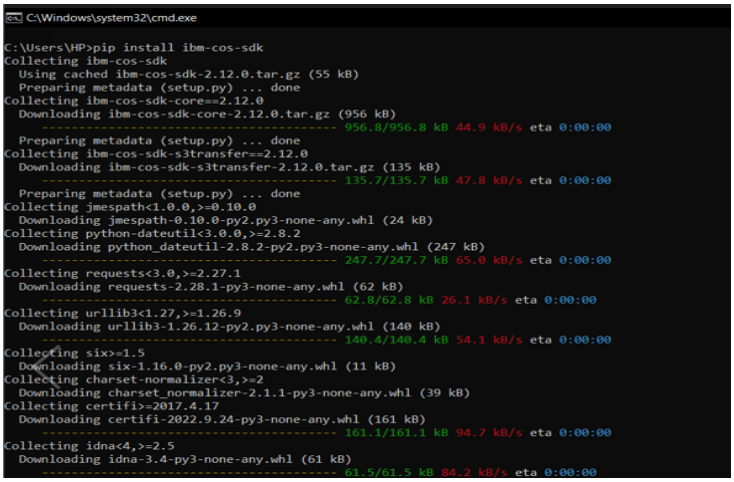
Node-RED is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs, and online services as part of the Internet of Things. Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions.

2. IBM Watson IoT Platform

A fully managed, cloud-hosted service with capabilities for device registration, connectivity, control, rapid visualization, and data storage. IBM Watson IoT Platform is a managed, cloud-hosted service designed to make it simple to derive value from your IoT devices.

3. Python Editor

Various packages are installed to work IBM platform and for deployment of IOT device

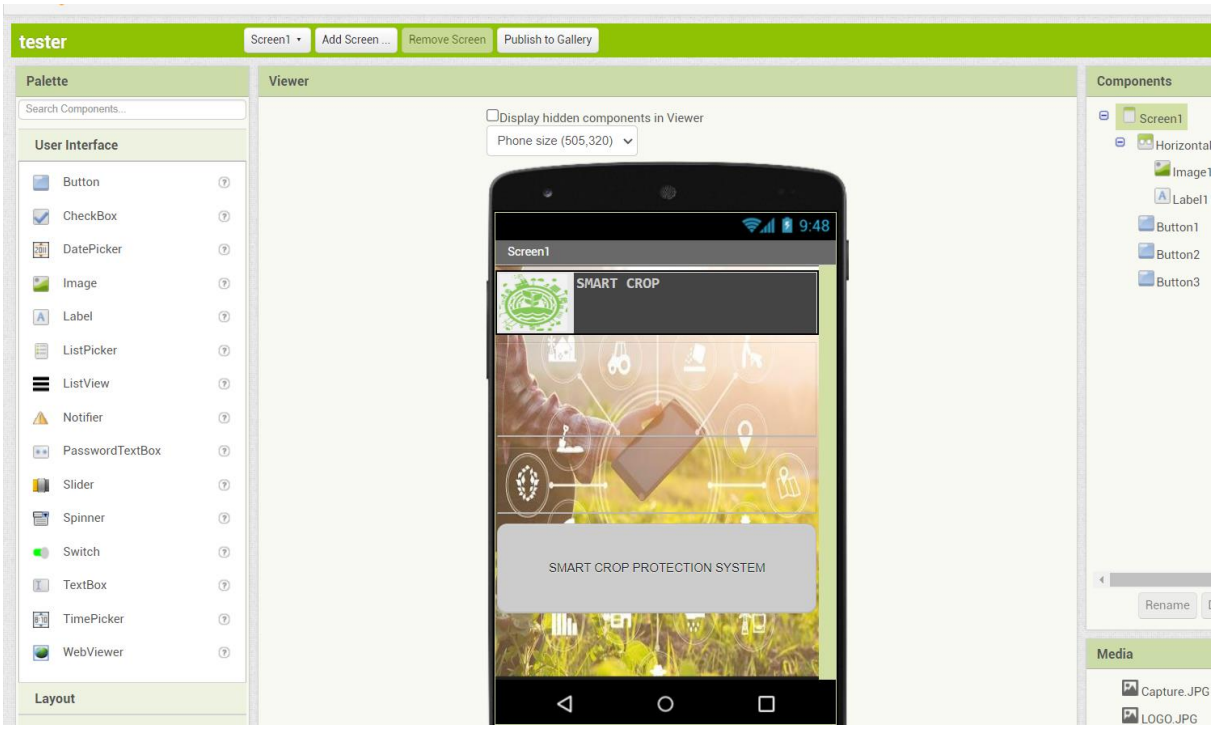


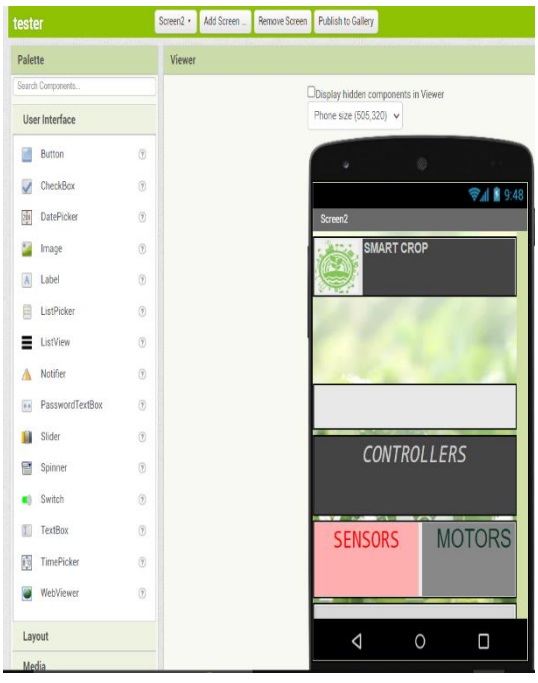
4. MIT-APP INVENTOR

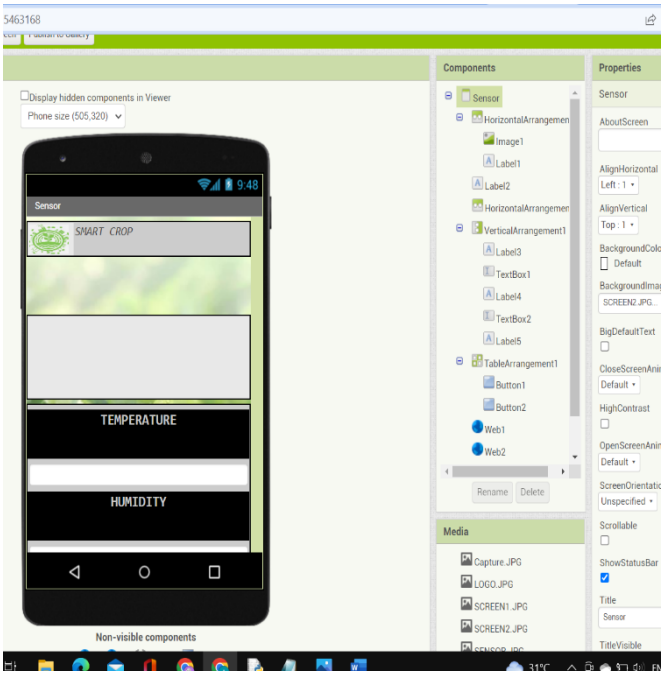
Simulation is carried on by using MIT-App inventor for Node-Red web application we have created a simulator with three modules they are:

- HOME SCREEN
- CONTROLLER SCREEN
- DATA SCREEN

There will be a designer and block panel separately and features and styles are being done in designer panel where we use to drag and drop desired icons. Similarly block panel here we give functionality to the features which are done by drag and drop method



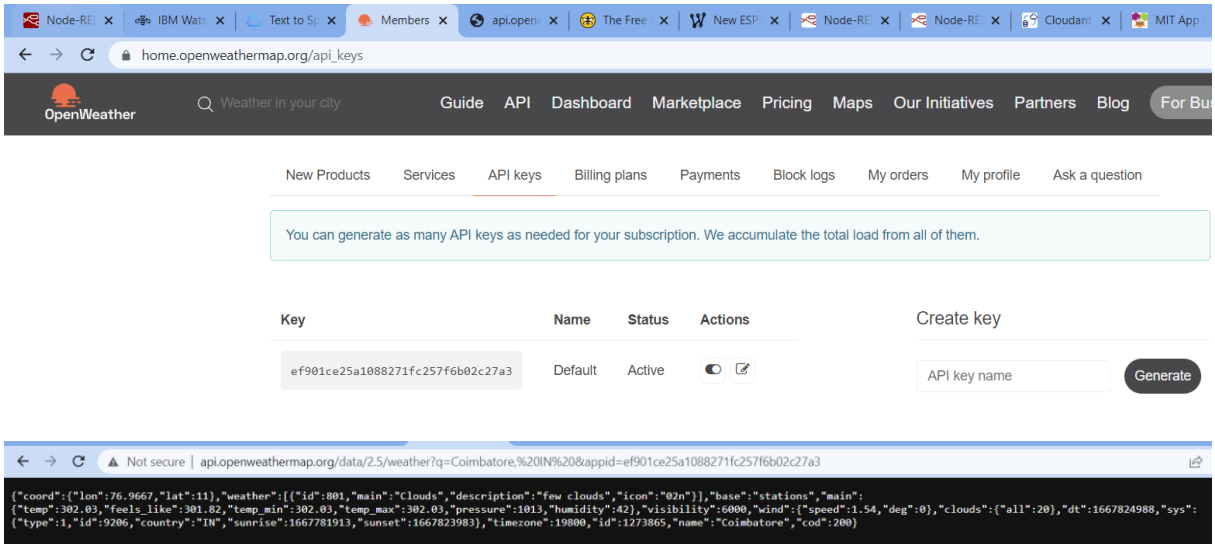




5. Open Weather API

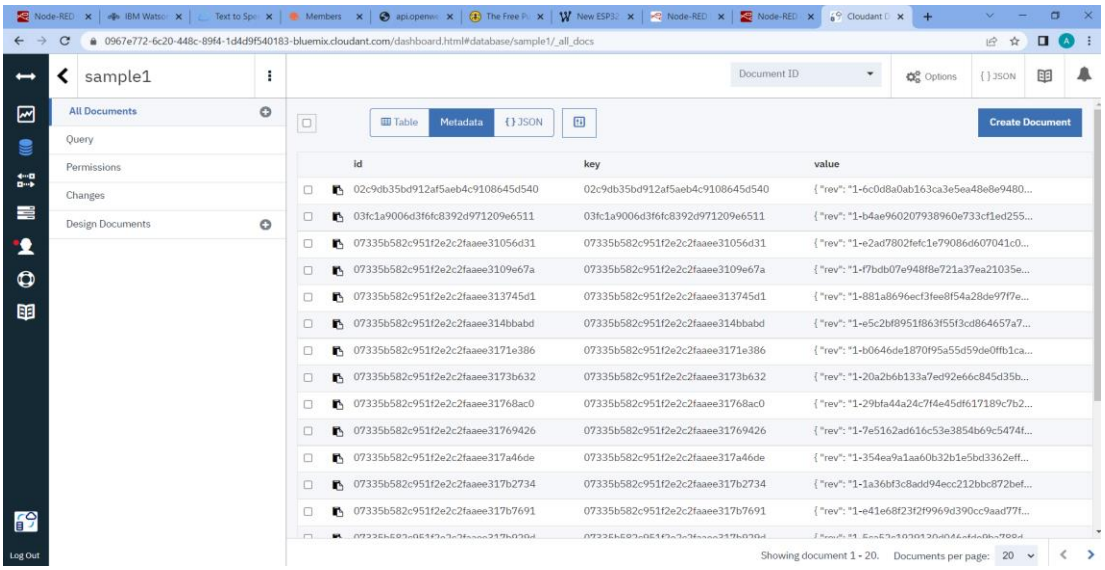
It is an online service that provides weather data. It provides current weather data, forecasts, and historical data to more than 2 million customers. Website link: https://openweathermap.org/guide

To configure: Create account in Open Weather and find the name of your city by searching then create API key to your account after that replace “city name” and “your API key” with your city and API key



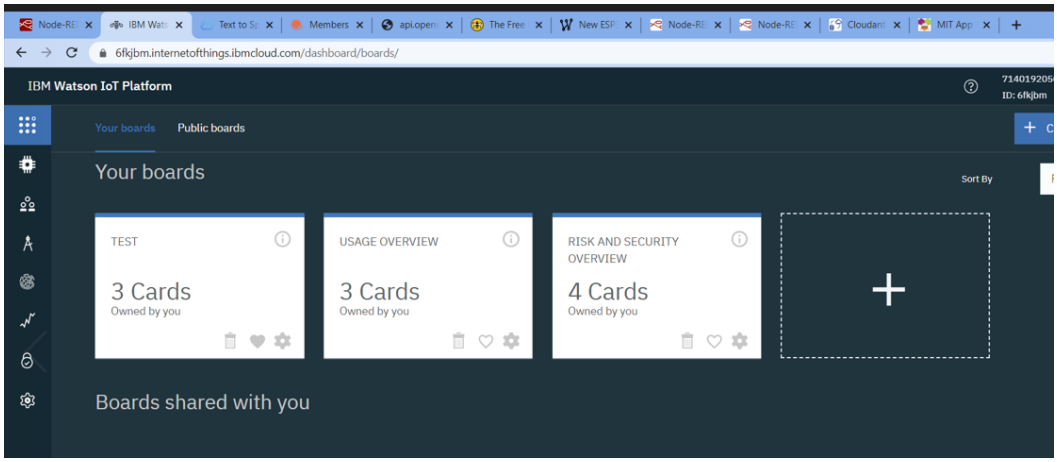
6. IBM Cloudant

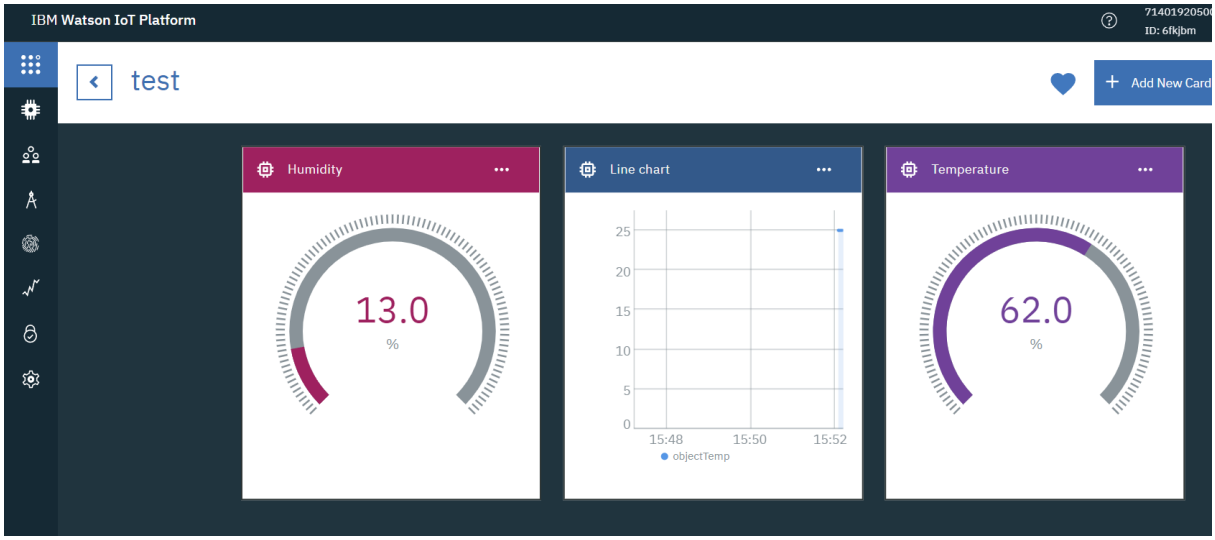
In order to store the NOD-RED value or any service provider we need to have a Database here the cloudant database is created inorder to store the values of the sensor reading of the devices

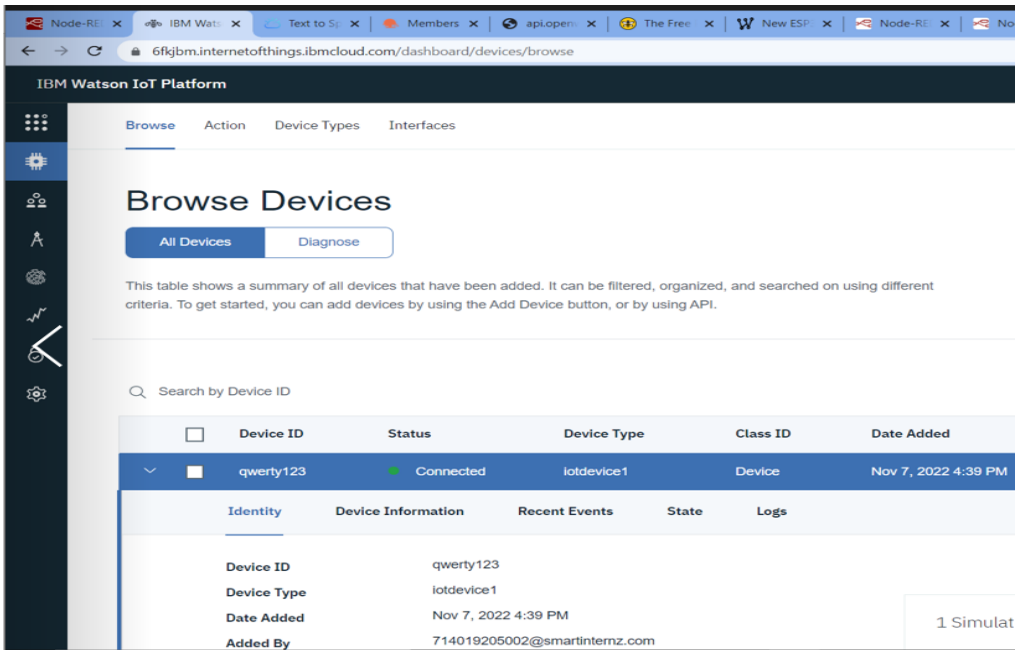


Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya

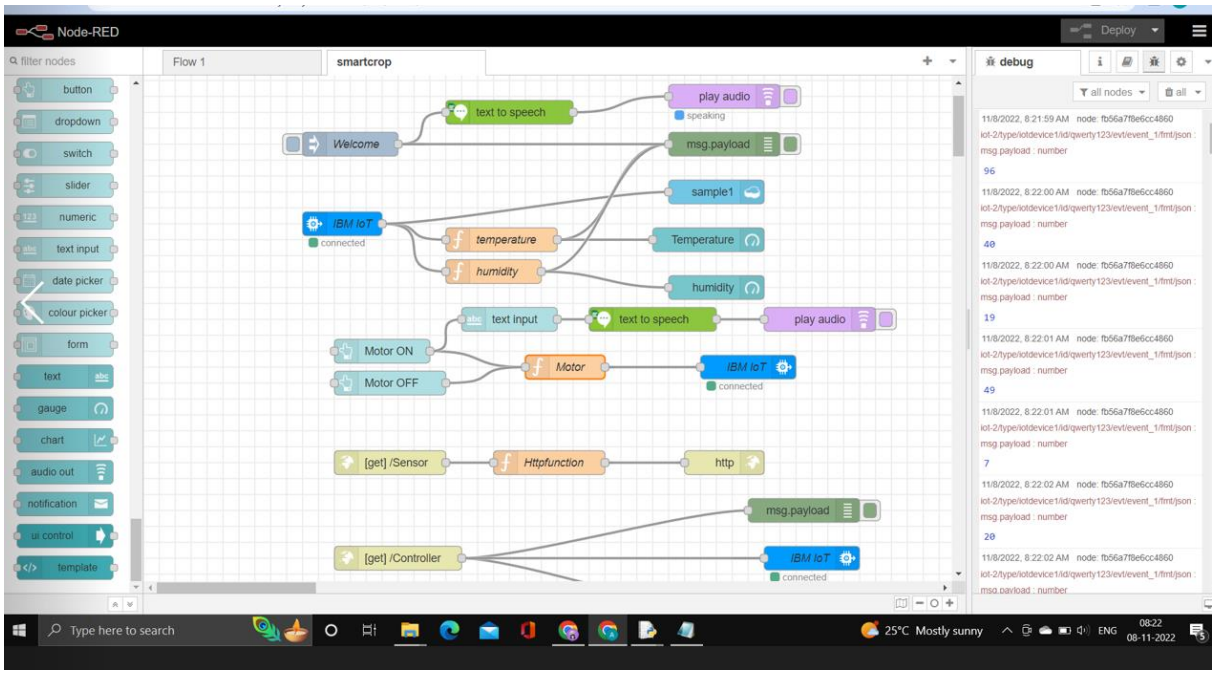
OUTPUT OF WEB-BASED APPLICATIION USING NODE-RED



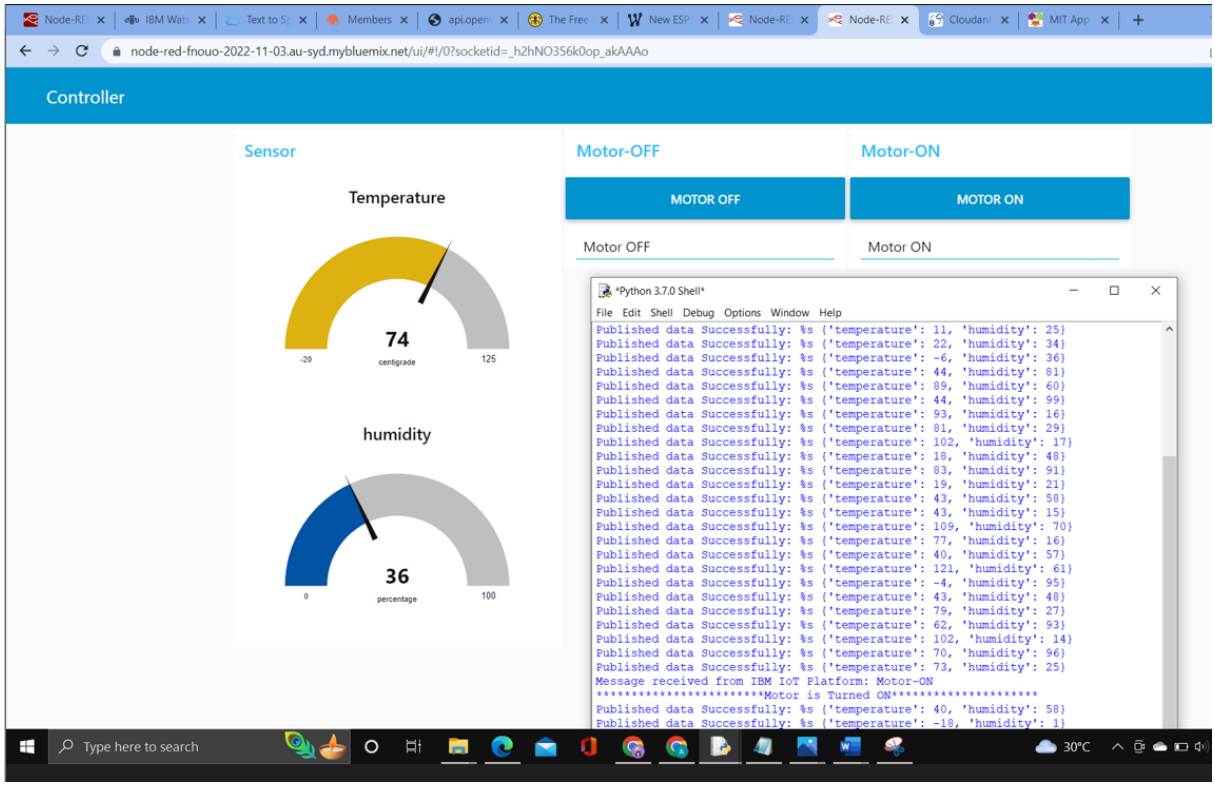




Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya



NODE-RED FLOW EDITOR WITH TEXT TO SPEECH



NODE-RED WEB GUI

Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya

SIMULATOR DEPLOYMENT OUTPUT

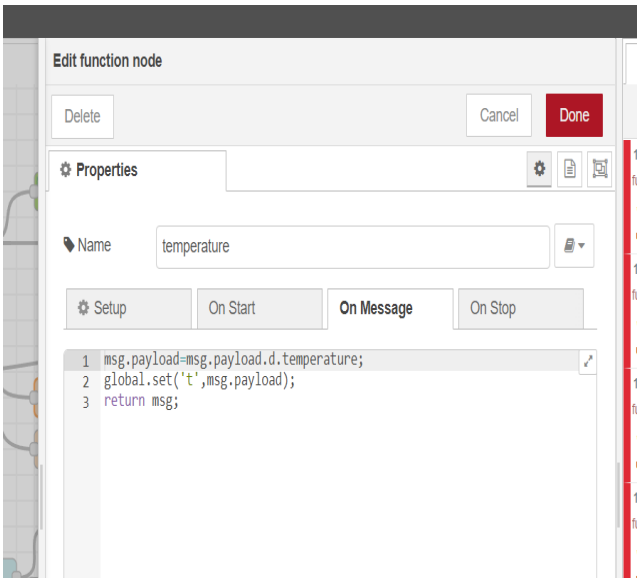
HOME SCREEN

CONTROLLER SCREEN

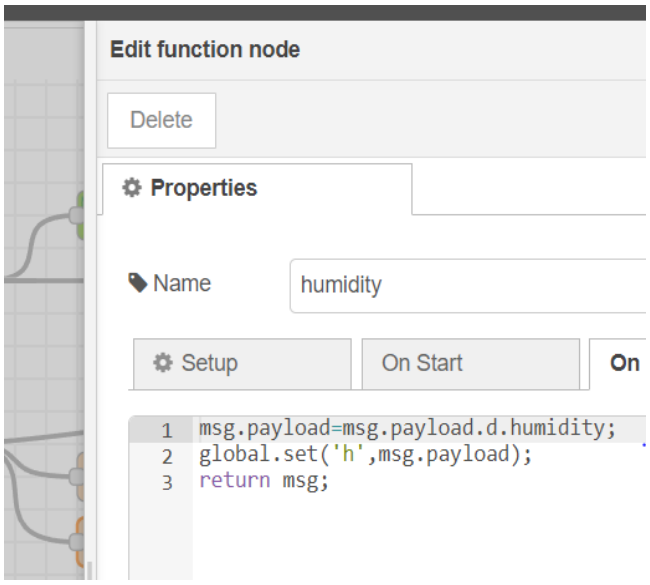
Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya

CODE FOR NODE-RED DEPLOYMENT

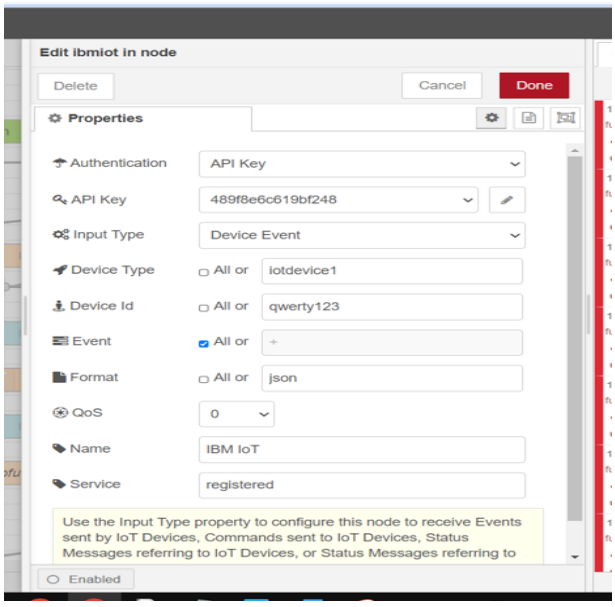
TEMPERATURE READING



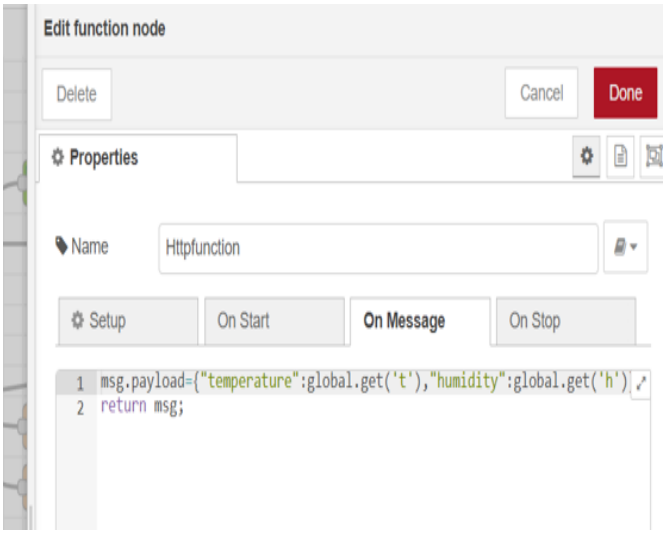
HUMIDITY READING



NODE-RED TO IOT WASTON

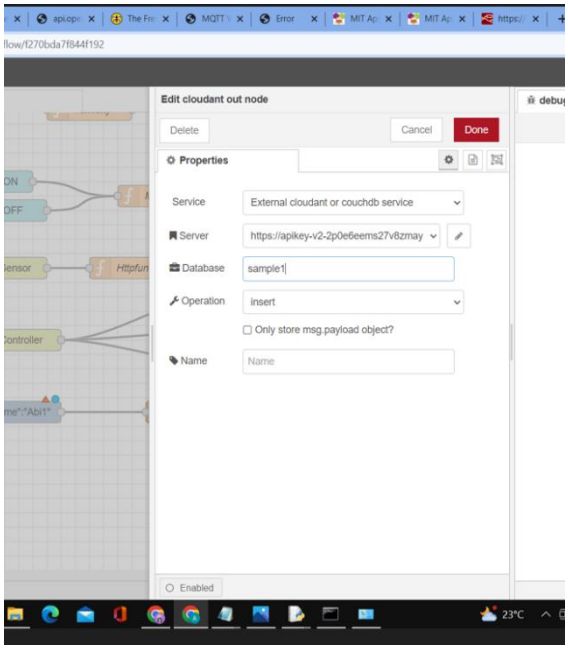


HTTP CONNECTION TO MIT-APP

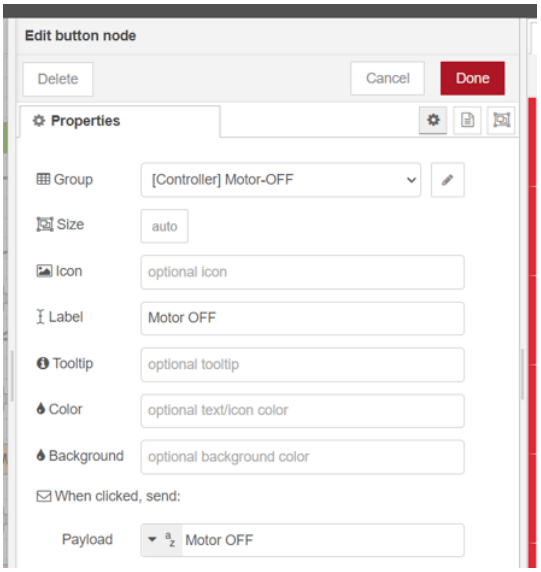


Team Member ID/Roll No: 714019205002
Name: Abirami Pavisya

NODE-RED TO CLOUDANT CONNECTIVITY



MOTOR ON AND OFF



CODE FOR PYTHON EDITOR

```
#IBM Watson IOT Platform
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "6fkjbm",
        "typeId": "iotdevice1",
        "deviceId":"qwerty123"
    },
    "auth": {
        "token": "johnyjohnyyespapa"
    }
}


def myCommandCallback(cmd):
```

```
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    if(m=="Motor-ON"):
        print("**********************Motor is Turned ON**********************")
    else:
         print("**********************Motor is Turned OFF**********************")
client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()


while True:
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'temperature':temp, 'humidity':hum}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
authenticator =
IAMAuthenticator('M_u6yEvEGJylj_ysbL_pG0ZOKuRCQW1LgXUtv_IcBPCR')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
)
text_to_speech.set_service_url('https://api.au-syd.text-to-
speech.watson.cloud.ibm.com/instances/23724eb6-a096-4a3a-b914-da0e442c1c5f')
with open('hello_world.wav', 'wb') as audio_file:
    audio_file.write(
        text_to_speech.synthesize(
```

```
        'Alert',
        voice='en-US_AllisonV3Voice',
        accept='audio/wav'
    ).get_result().content)
```

CODE MIT-APP INNVENTOR(SCREEN 3)