TEAM MANAGEMENT FOR AGILE PLANNING

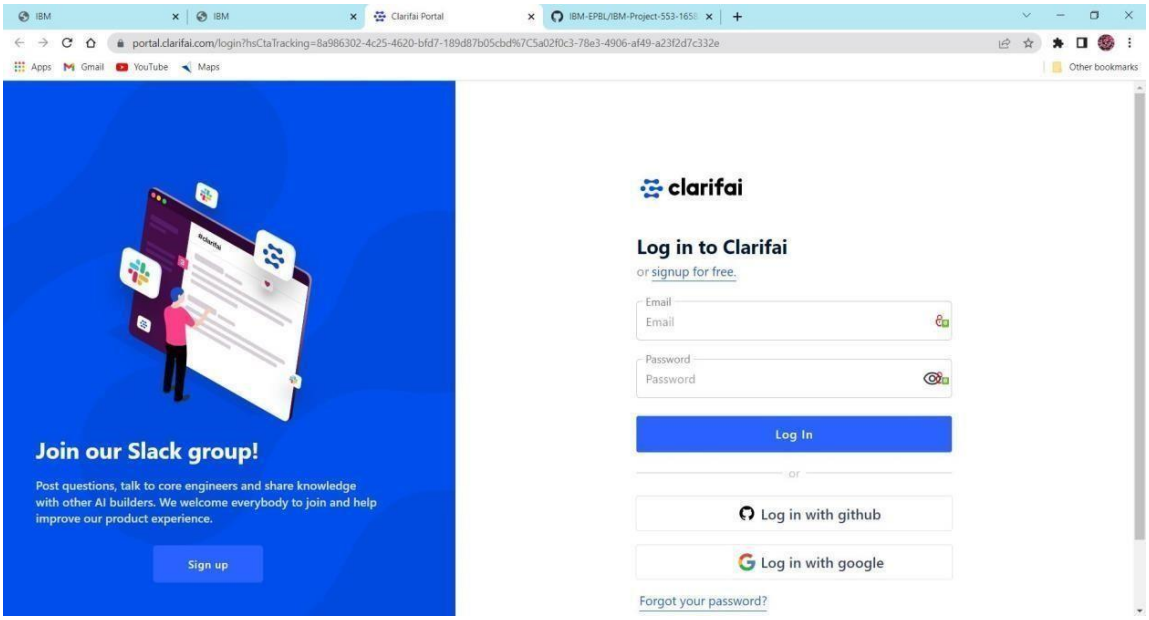
TEAM ID : PNT2022TMID17378

CLARIFAI:

Clarifai provides an end-to-end platform with the easiest to use UI and API in the market. Clarifai Inc. is an artificial intelligence (AI) company that specializes in computer vision and uses machine learning and deep neural networks to identify and analyse images and videos. The company offers its solution via API, mobile SDK, and on-premise solutions.

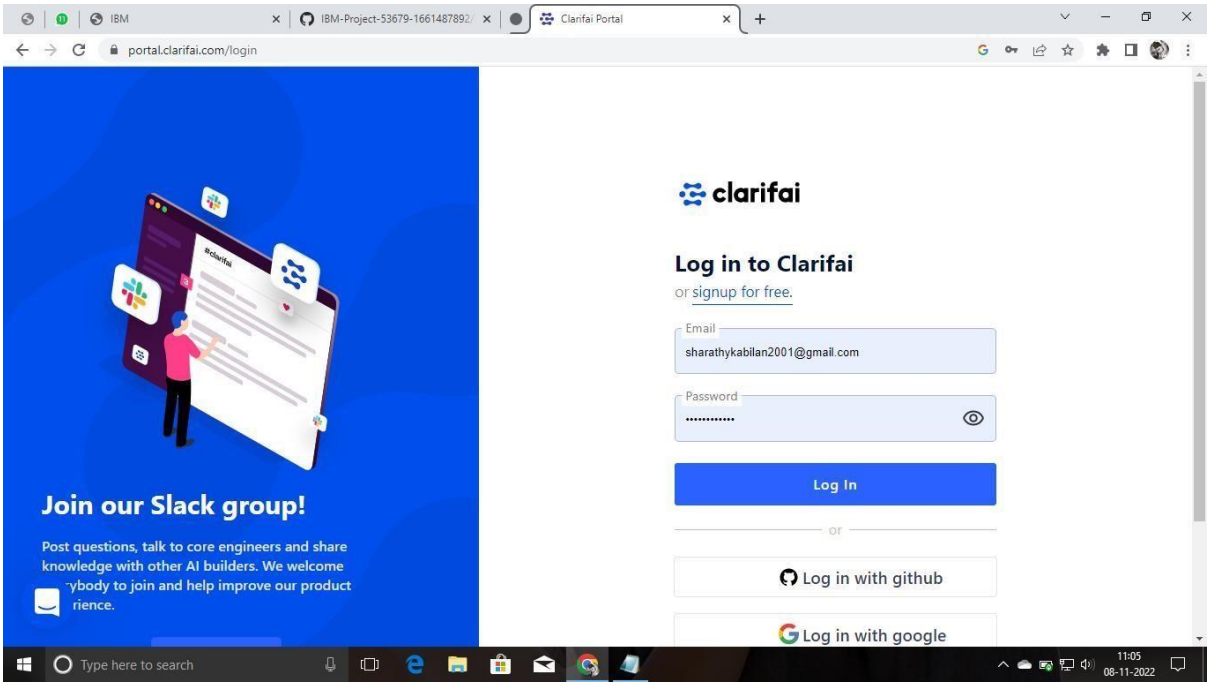
STEP 1:

- Open Clarifai portal in web browser.



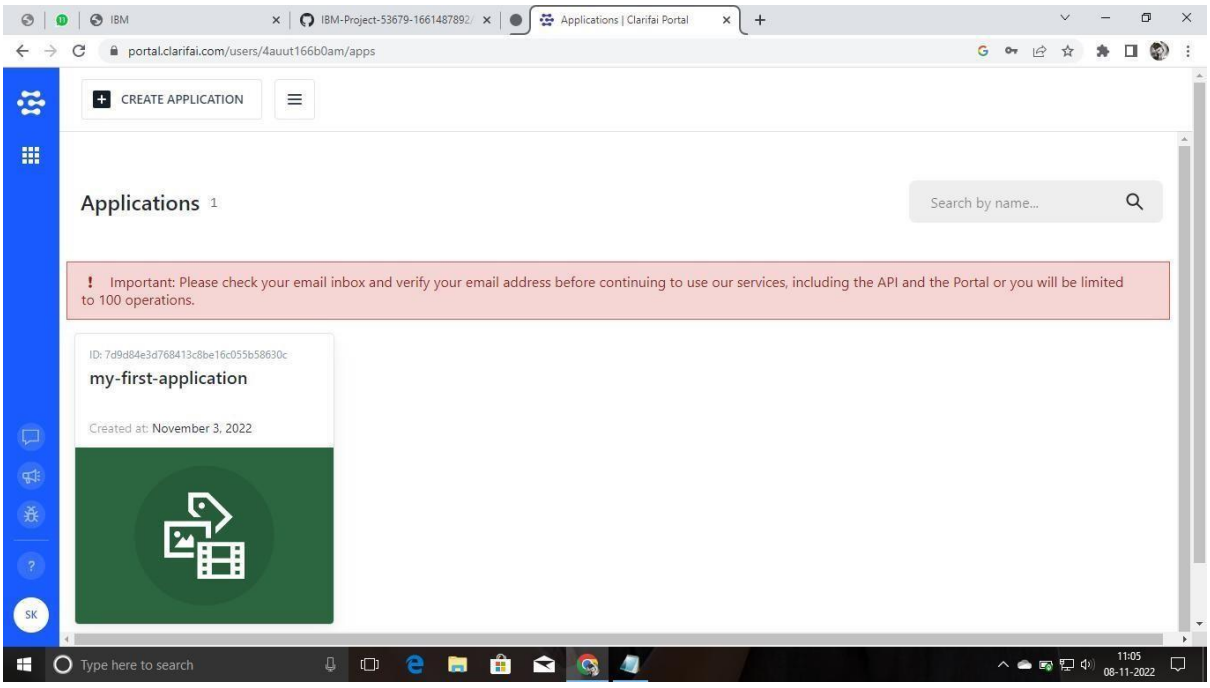
STEP 2:

- Signup using the required user mail and password



STEP 3:

Finally, Created an account



IBM WATSON PLATFORM:

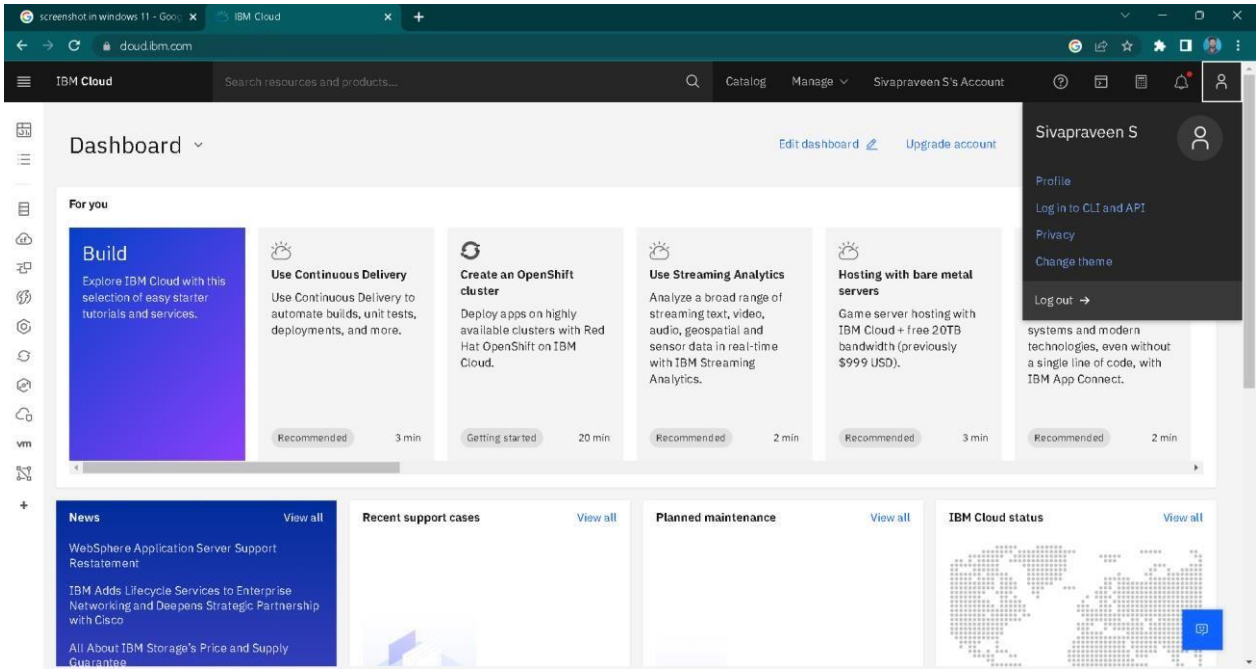
We need to have basic knowledge of the following cloud services:

- IBM Watson IoT Platform

- Node-RED Service
- Cloudant DB

We need to create an IBM Cloud Account to complete this project.

LOGIN:

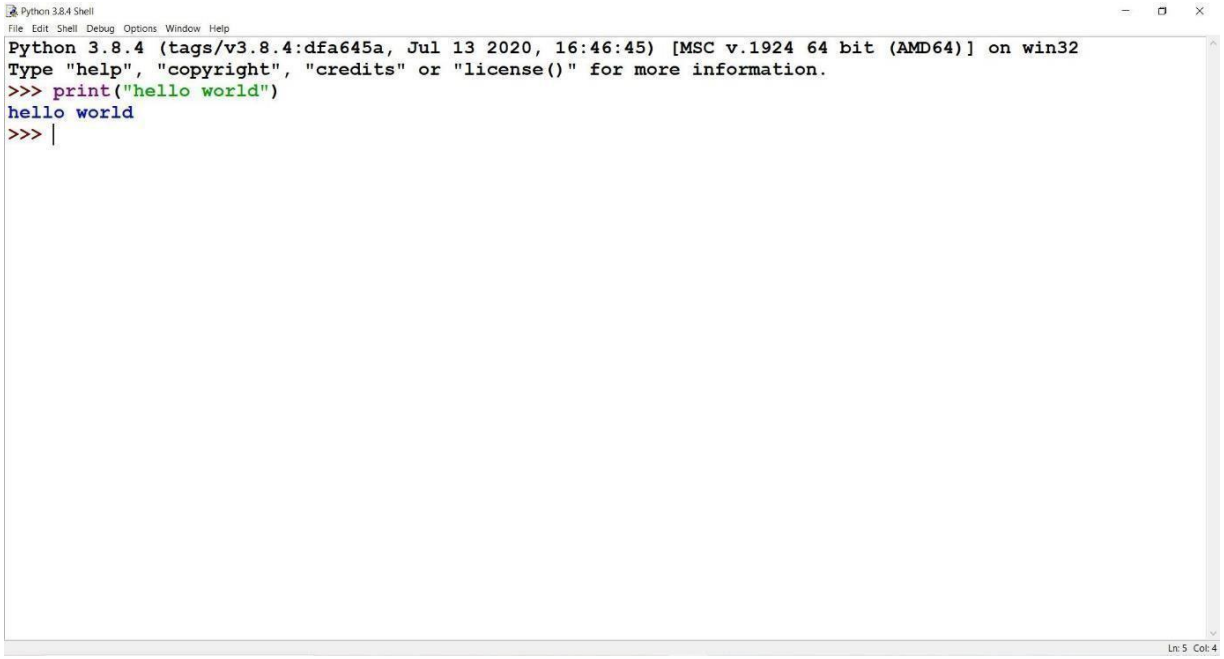


PYTHON IDLE INSTALLATION:

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a generalpurpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems.

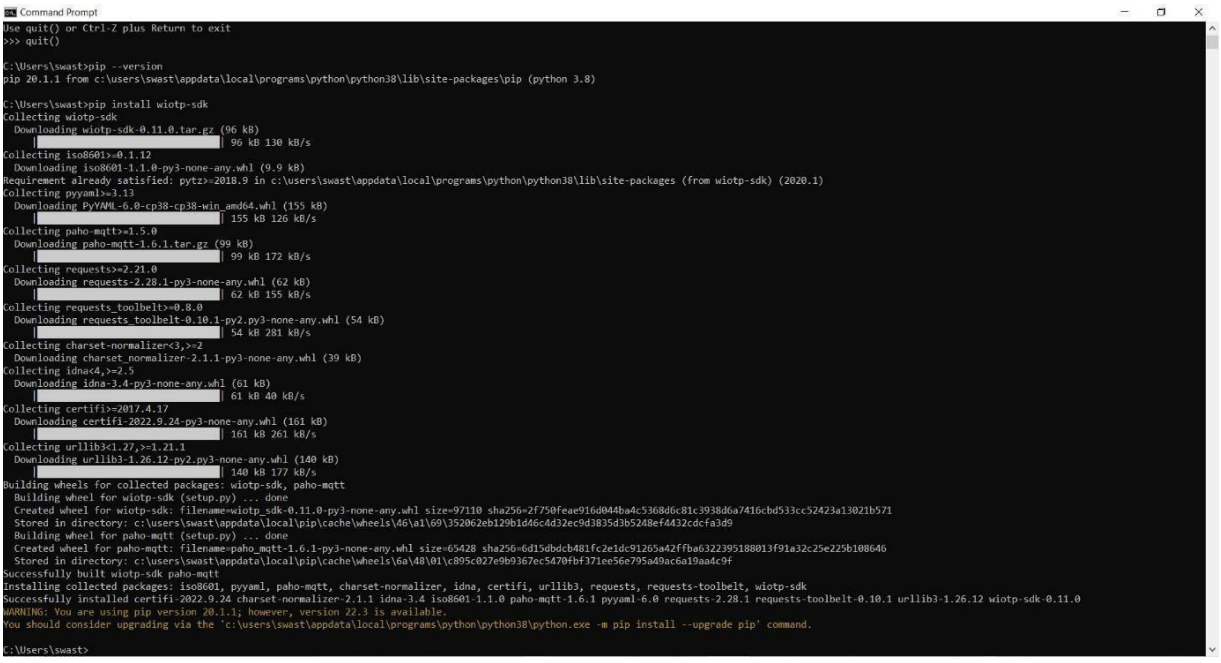
STEP 1:

- Python is installed successfully

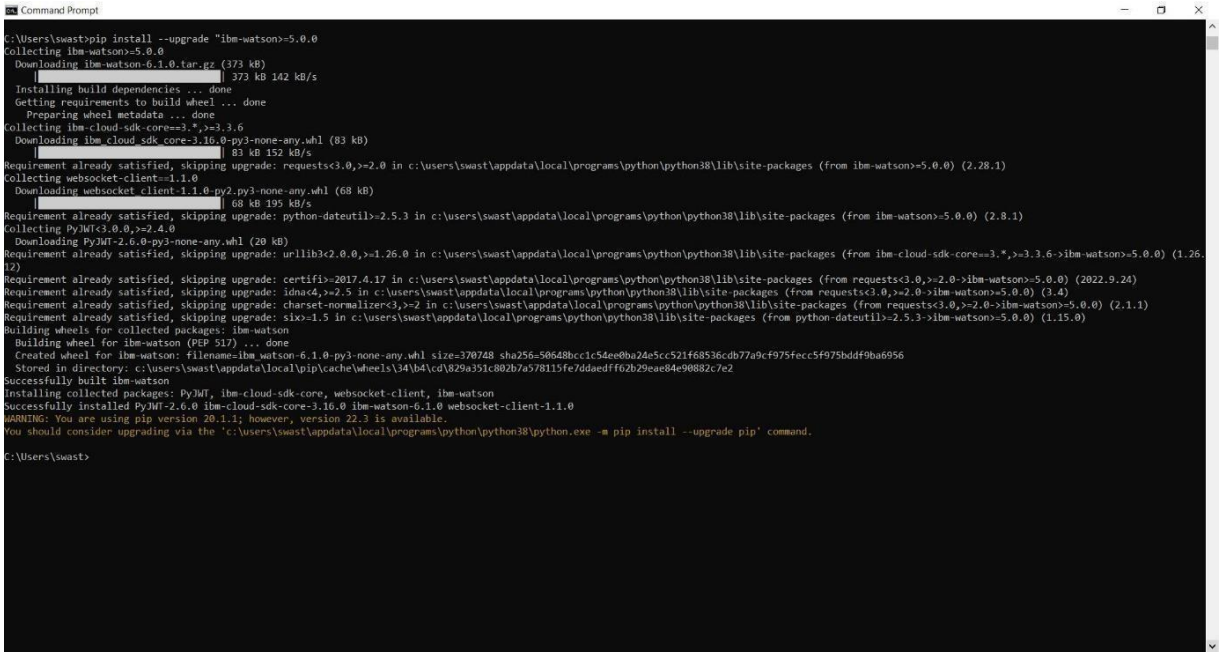


STEP 2:

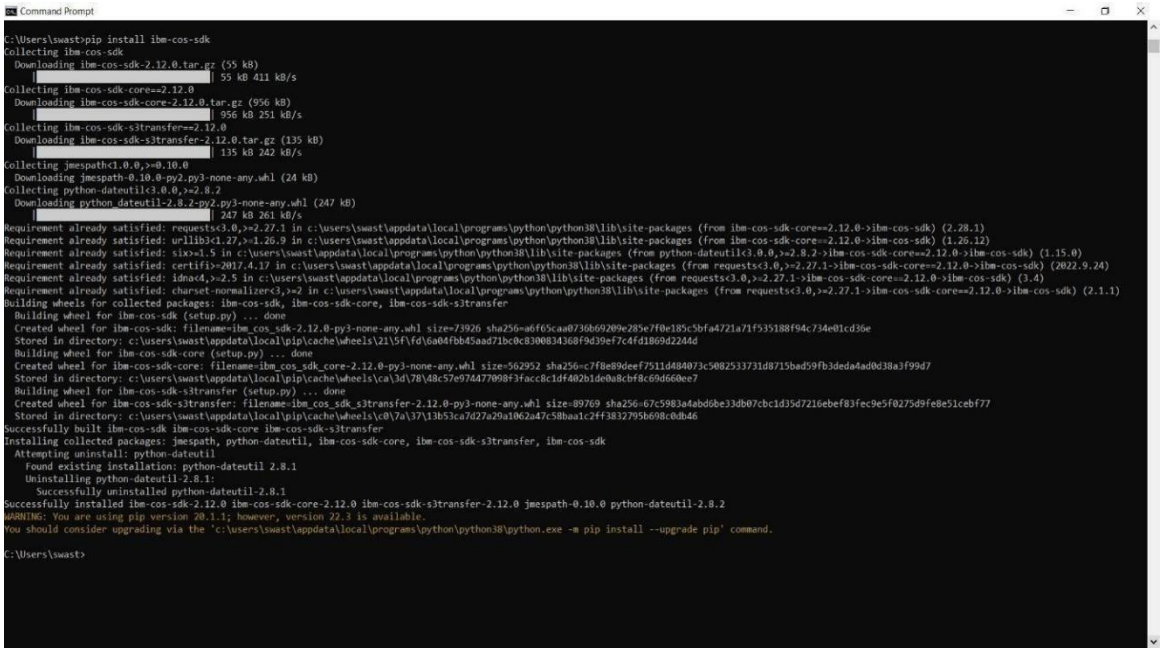
- The required python libraries are installed.
- Watson Python SDK to connect to IBM Watson Platform using python code is installed
- pip install wiotp-sdk



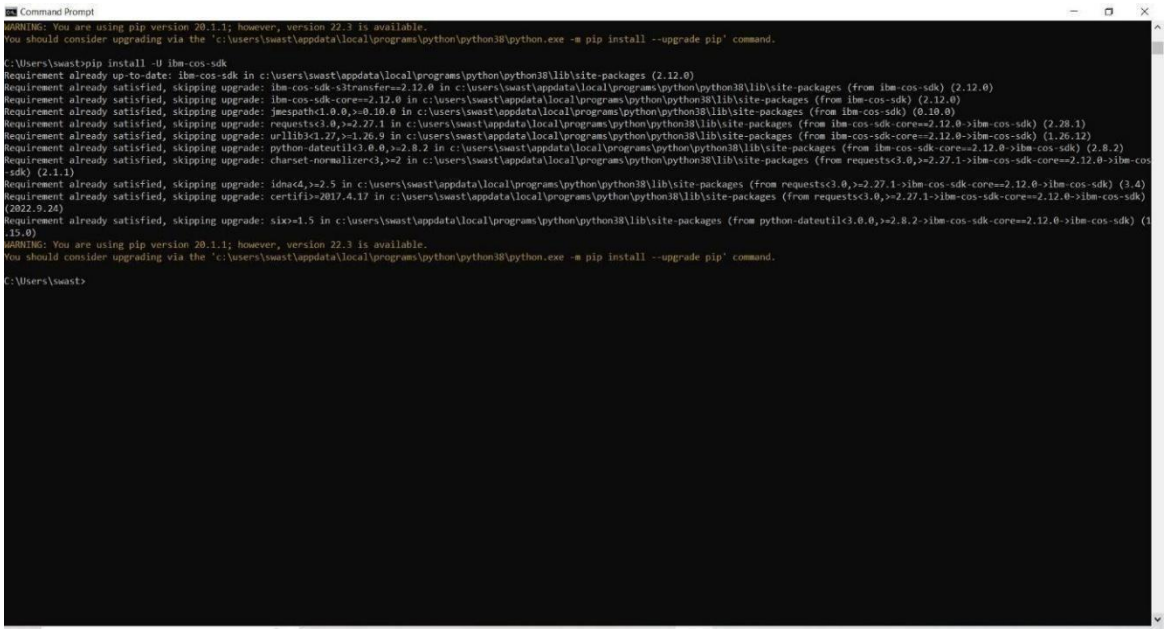
- Python client library for IBM Text to Speech is installed
- pip install --upgrade "ibm-watson>=5.0.0



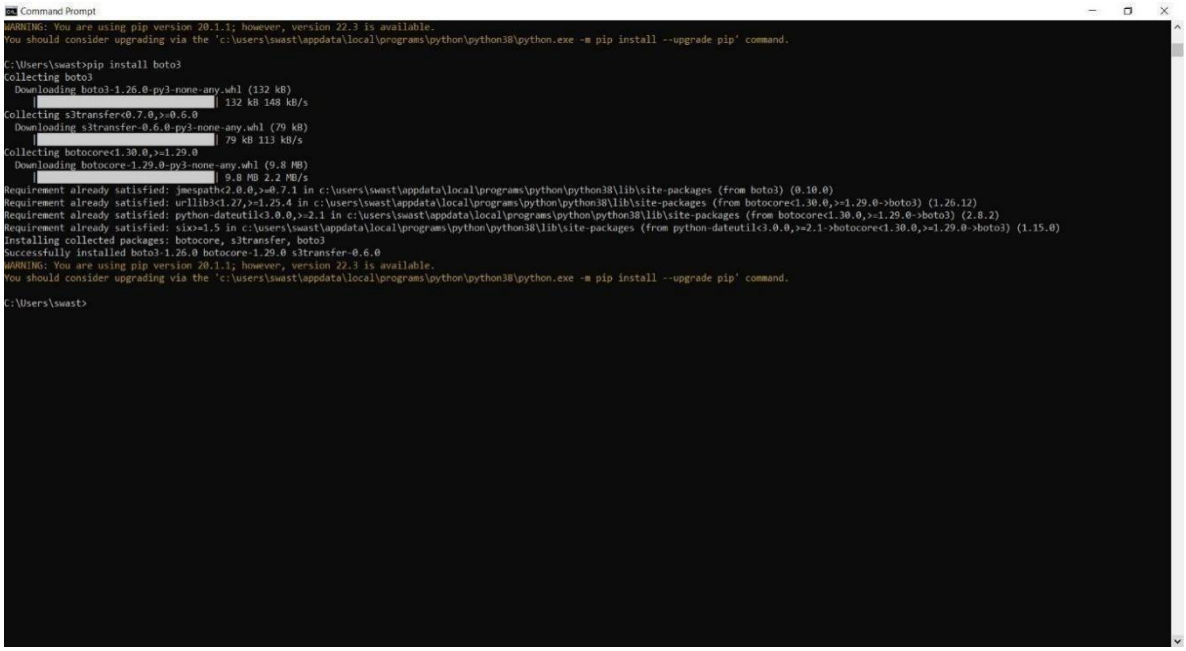
- Required Libraries for cloud object storage is installed.
- pip install ibm-cos-sdk



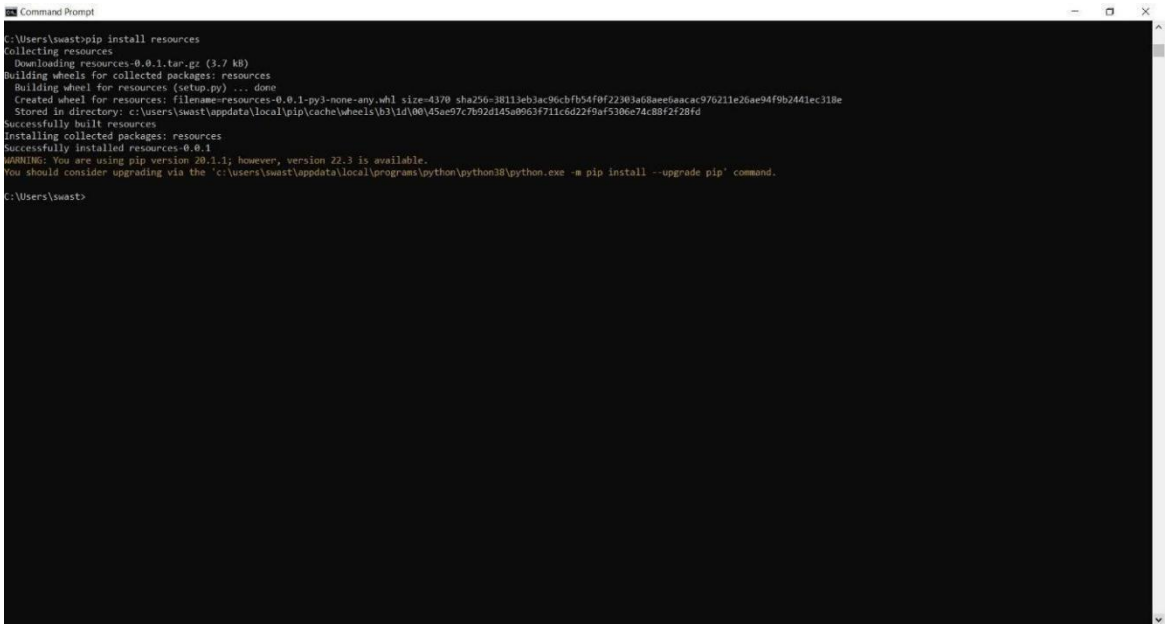
- pip install -U ibm-cos-sdk



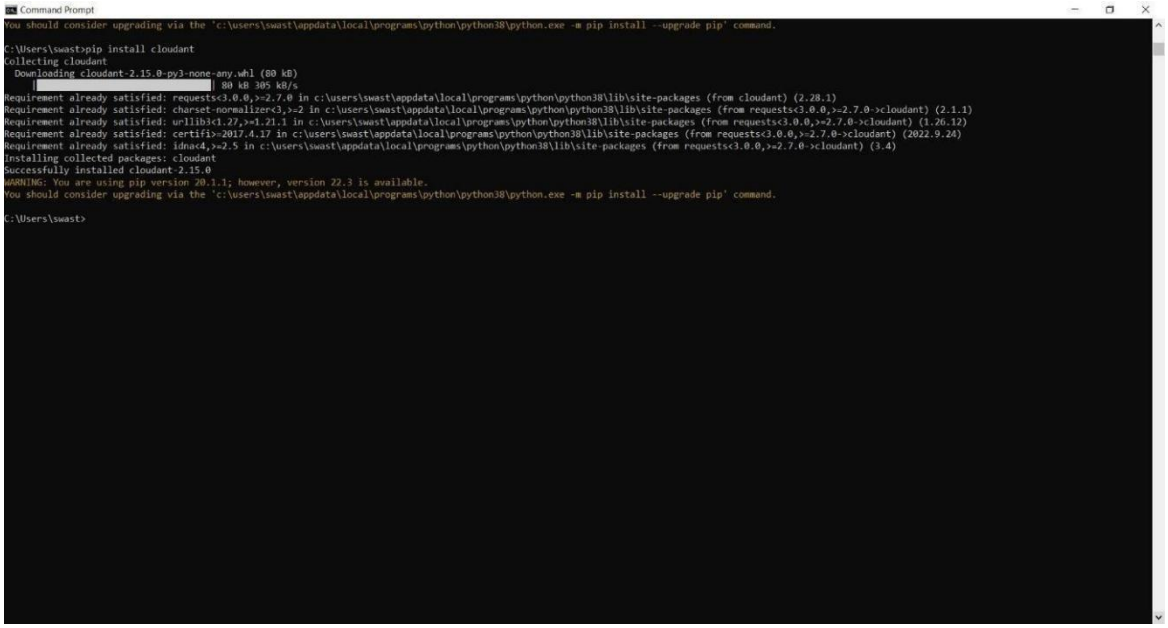
- pip install boto3



- pip install resources



- pip install cloudant



FROM PYTHON TO IBM:

Python code to generate random data and pass it to IBM Watson IoT platform **SourceCode:** import time import sys import ibmiotf.application import

ibmiotf.device import

random

#Provide your IBM Watson Device Credentialsorganization = "wu5b55" deviceType = "crop1" deviceId = "1234" authMethod = "token" authToken = "1234567890"

Initialize GPIOtry:

deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken} deviceCli =

ibmiotf.device.Client(deviceOptions)

#.....

except Exception as e:

print("Caught exception connecting device: %s" % str(e))sys.exit()

Connect and send a datapoint "hello" with value "world" into the cloud as an event of type"greeting" 10 times deviceCli.connect()while True:

temp=random.randint(0,

```
        100)
        Hum=random.randint(0,1
        00)
        moisture=random.randint
        (0,100)

        data = { 'temperature' : temp, 'Humidity': Hum, 'Moisture':moisture }



def myOnPublishCallback():
        print ("Temperature = " + str(temp)+" C Humidity = " +
    str(hum)+ " moisture = " +str(moisture) + "to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor",
    "json", data, qos=0,on_publish=myOnPublishCallback)
        if not success:
          print("Not connected to
        IoTF")time.sleep(10)


        deviceCli.commandCallback = myCommandCallback

    # Disconnect the device and application from
    the clouddeviceCli.disconnect()
```

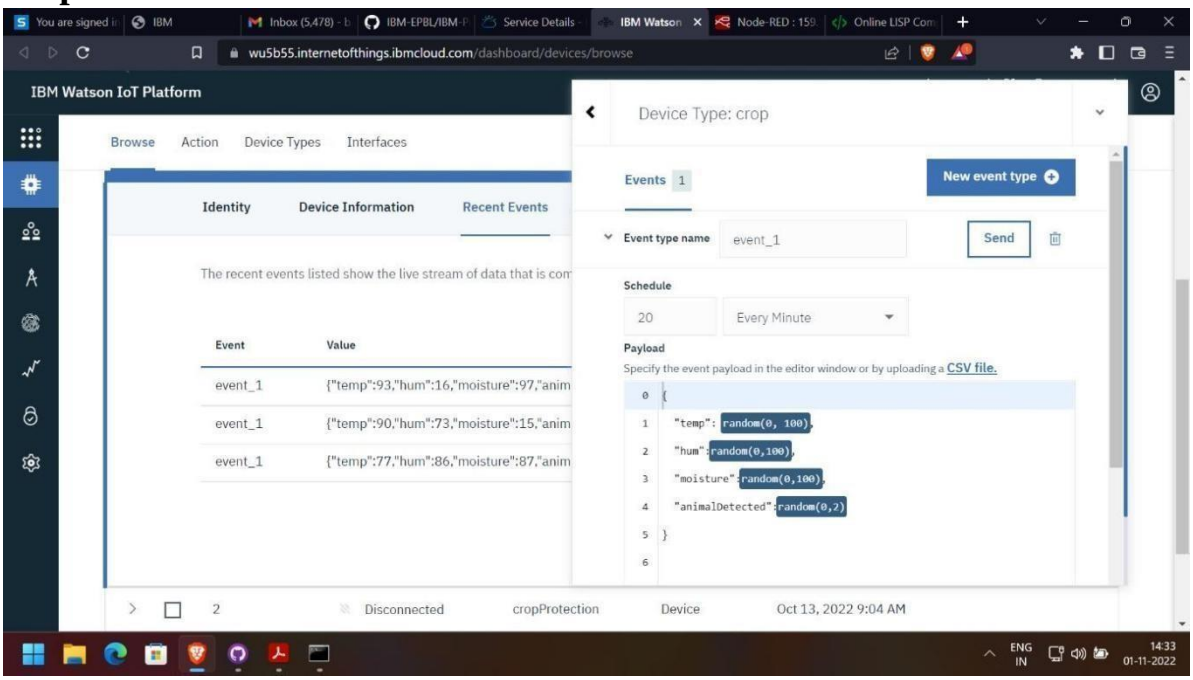
DATA GENERATION PLATFORM:

Source code is deployed on IBM Watson IoT platform to generate sensor data.

SourceCode:

```
{
        "temperature": random(0, 100),
        "humidity": random(0, 100),
        "moisture": random(0, 100),
        “animalDetected”:random(0,2)
}
```

Output:



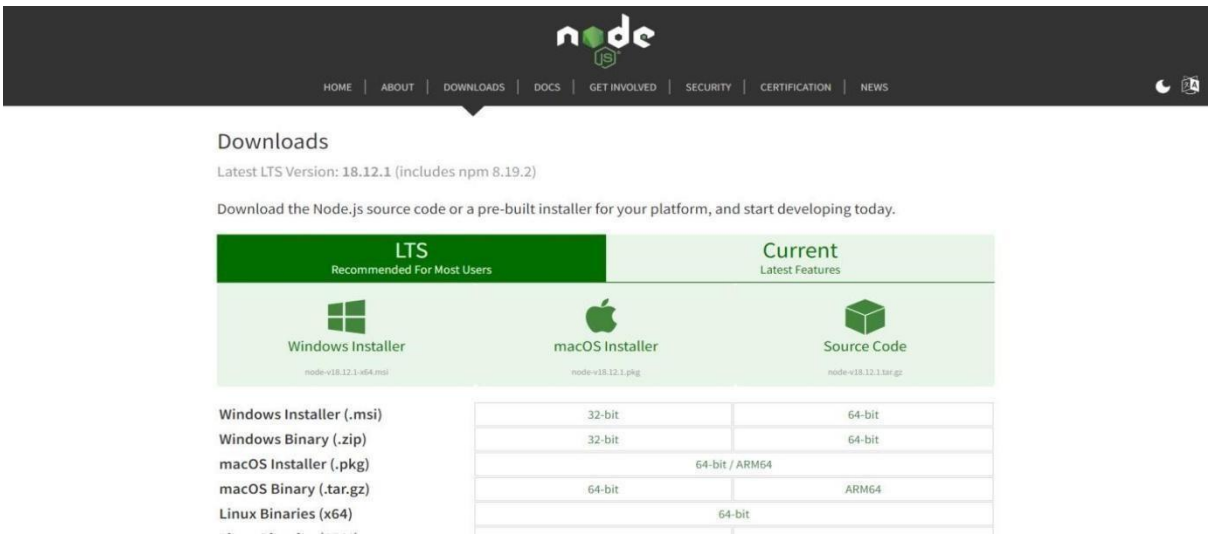
PYTHON CODE TO IBM:

```
import time
import sys

import ibmiotf.application import
ibmiotf.device import random



#Provide your IBM Watson Device Credentials organization
= "wu5b55" deviceType =
"crop1" deviceId = "1234" authMethod = "token"
authToken = "1234567890"


# Initialize GPIO



try: deviceOptions={"org":organization,"type":deviceType,"id":

deviceId, "auth-method": authMethod, "auth-token": authToken}deviceCli
        = ibmiotf.device.Client(deviceOptions) #.....


except Exception as e:

        print("Caught exception connecting device: %s" % str(e))sys.exit()


# Connect and send a datapoint "hello" with value "world" into thecloud as
an event of type "greeting" 10 times

deviceCli.connect() while
True:

          #Get Sensor Data from DHT11
```

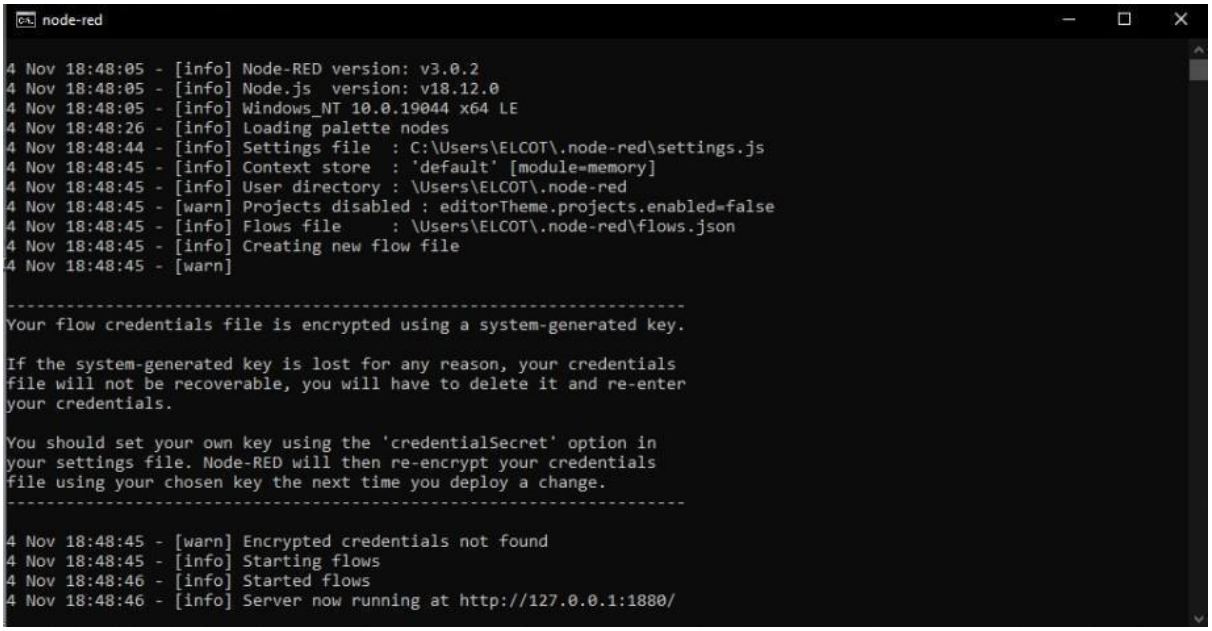
```
        temp=random.randint(0,100)
        Hum=random.randint(0,100)
        moisture=random.randint(0,100)


        data = { 'temperature' : temp, 'Humidity': Hum,
'Moisture':moisture }
#print data def myOnPublishCallback():
            print ("Temperature = " + str(temp)+" C Humidity = " +
str(hum)+ " moisture = " + str(moisture) + "to IBM Watson")


        success = deviceCli.publishEvent("IoTSensor", "json", data,qos=0,
on_publish=myOnPublishCallback) if
        not success:
            print("Not connected to IoTF")
        time.sleep(10)


        deviceCli.commandCallback = myCommandCallback


# Disconnect the device and application from the cloud
deviceCli.disconnect()
```

NODE-JS CONNECTION:

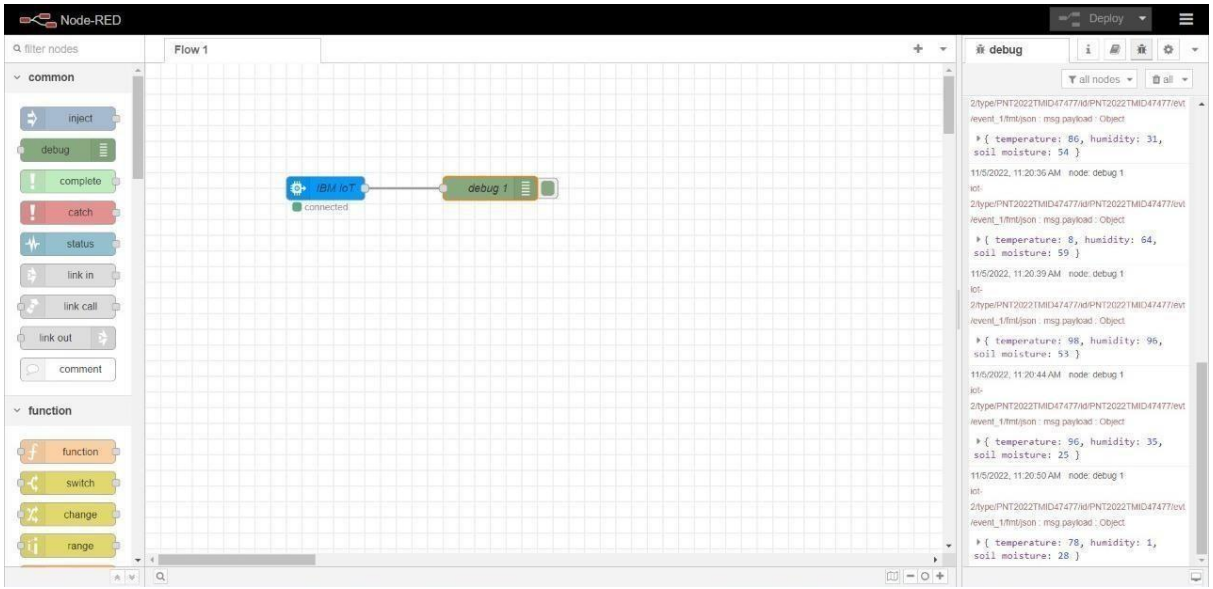
STEP1: Download and Install NODE JS.



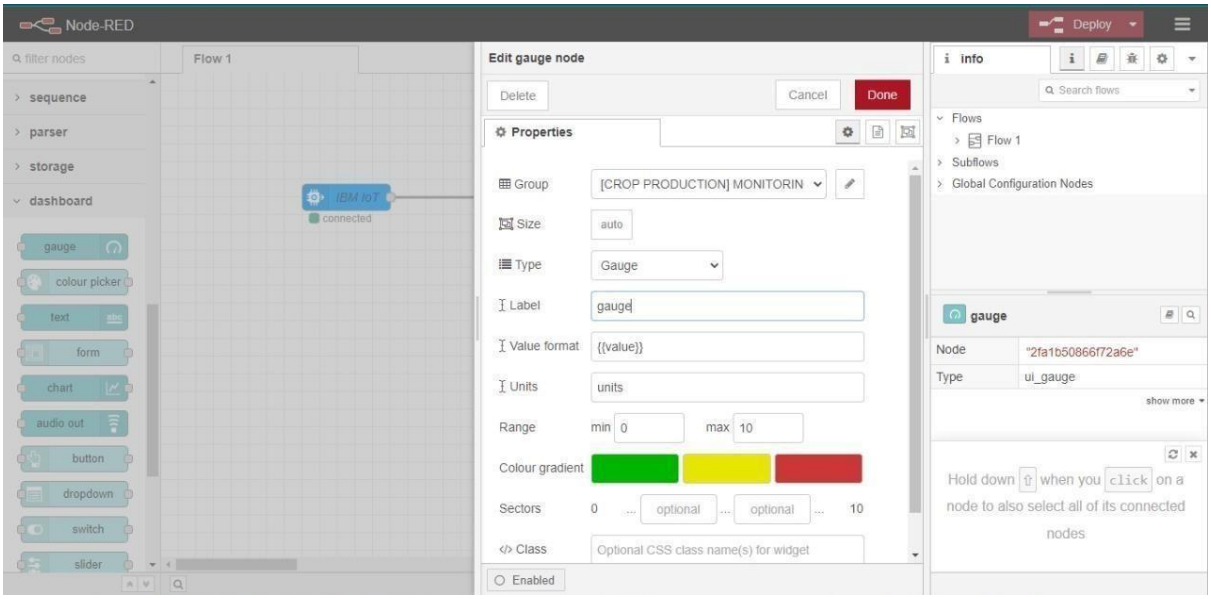
STEP2: Setup node.js and configure command prompt for error check .open node-red from the generated link.

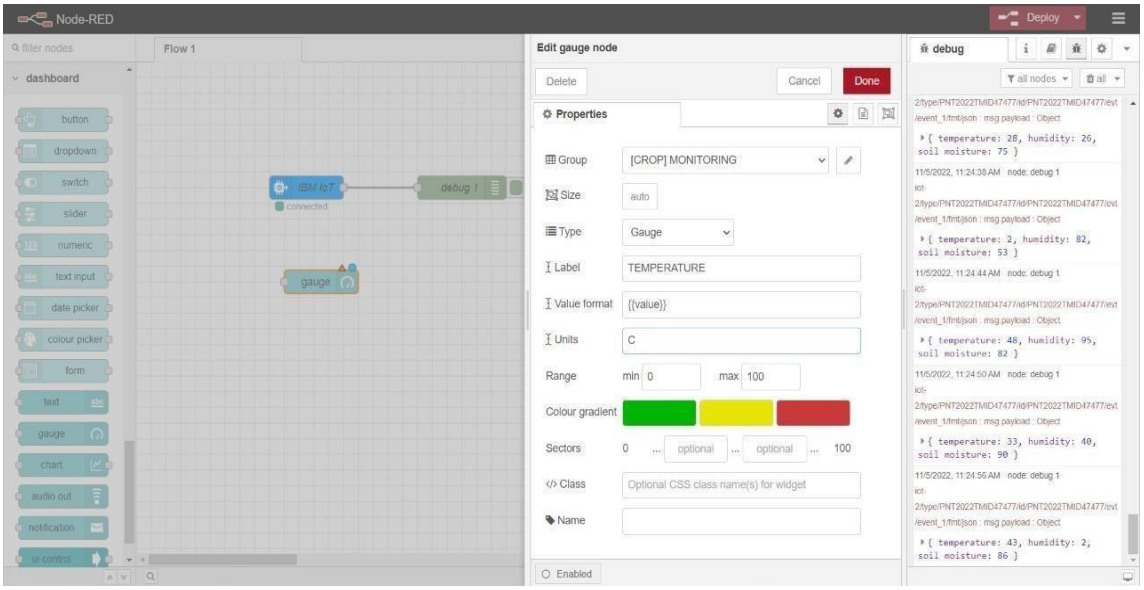


STEP3: Connect IBM IOT in and Debug 1 and Deploy.



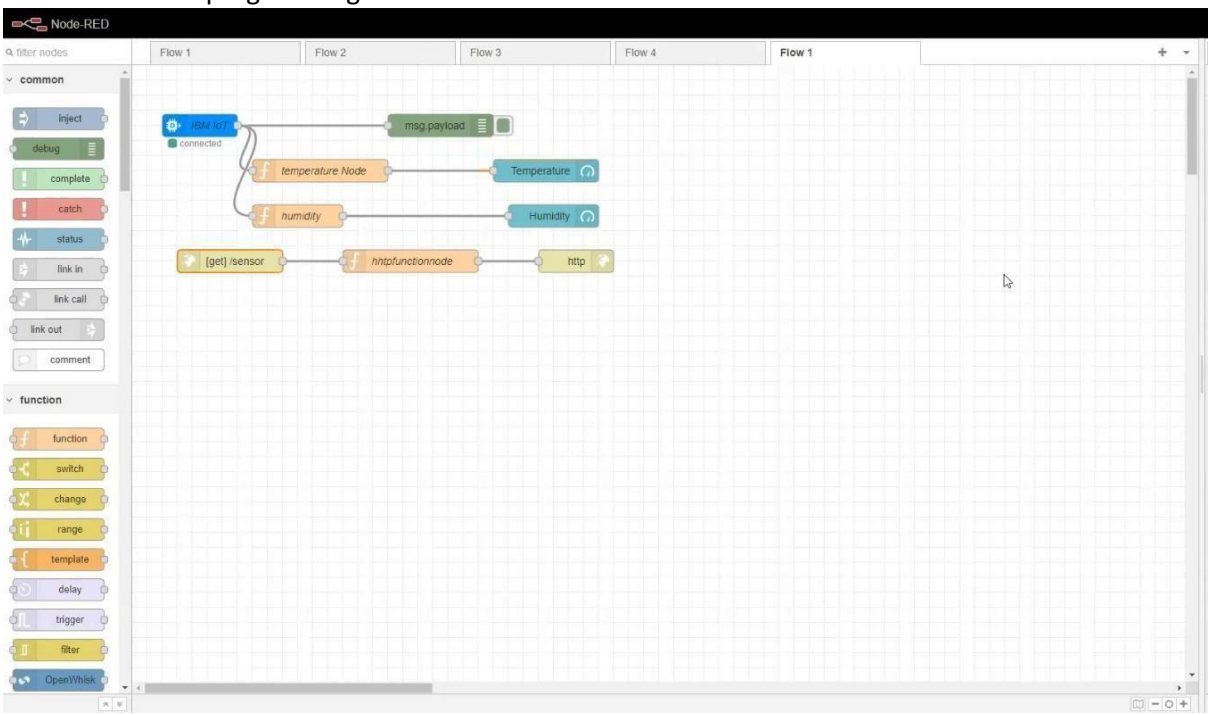
STEP4: Edit gauge node (Here the gauge nodes are named as Temperature, Humidity and Soilmoisture).



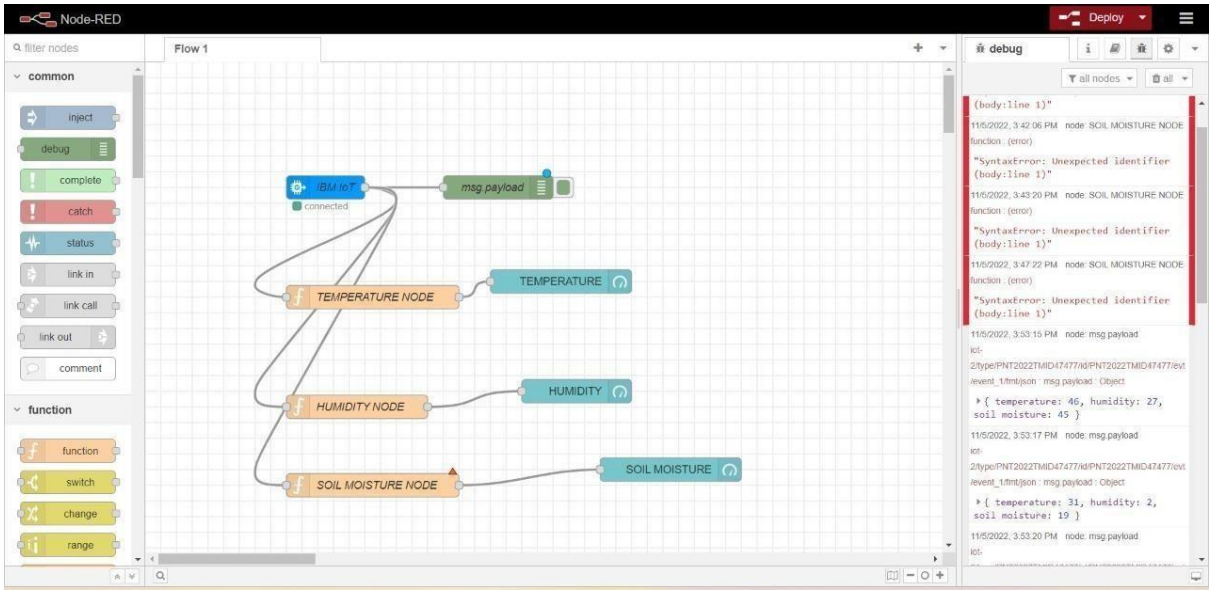


SIMULATION:

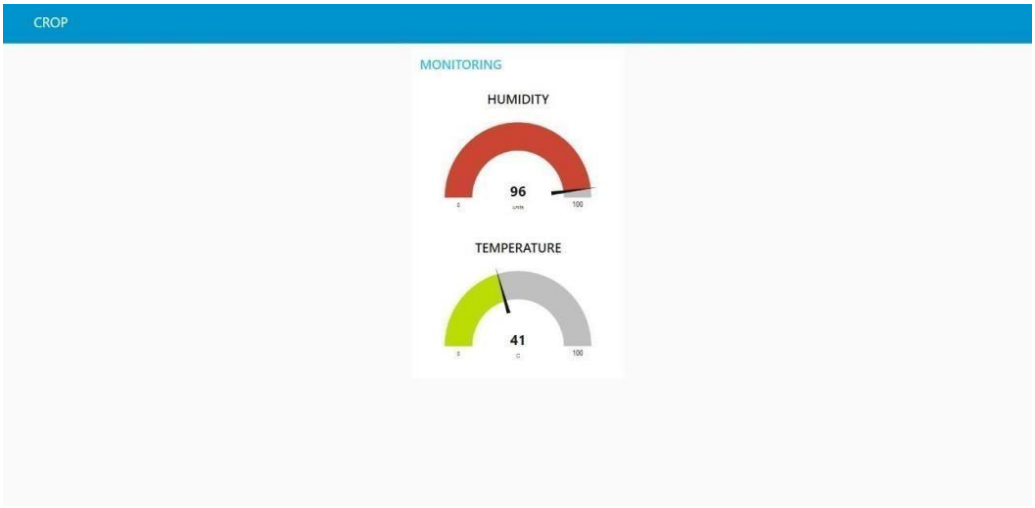
STEP1: Simulated program to get the random values

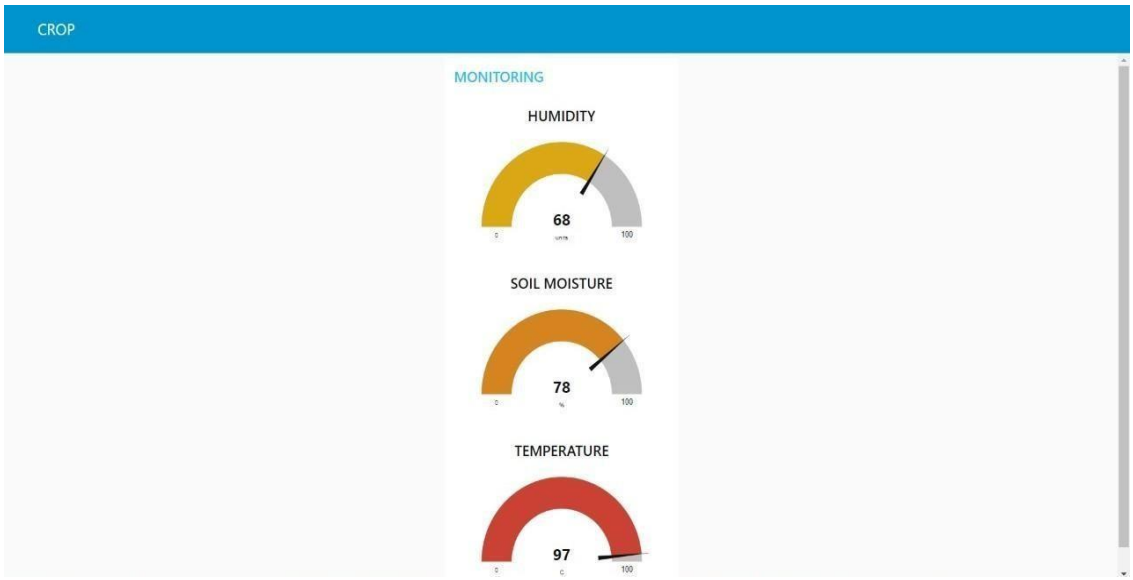


STEP2: Generate debug message from IBM Watson IoT Platform and connect the nodes.



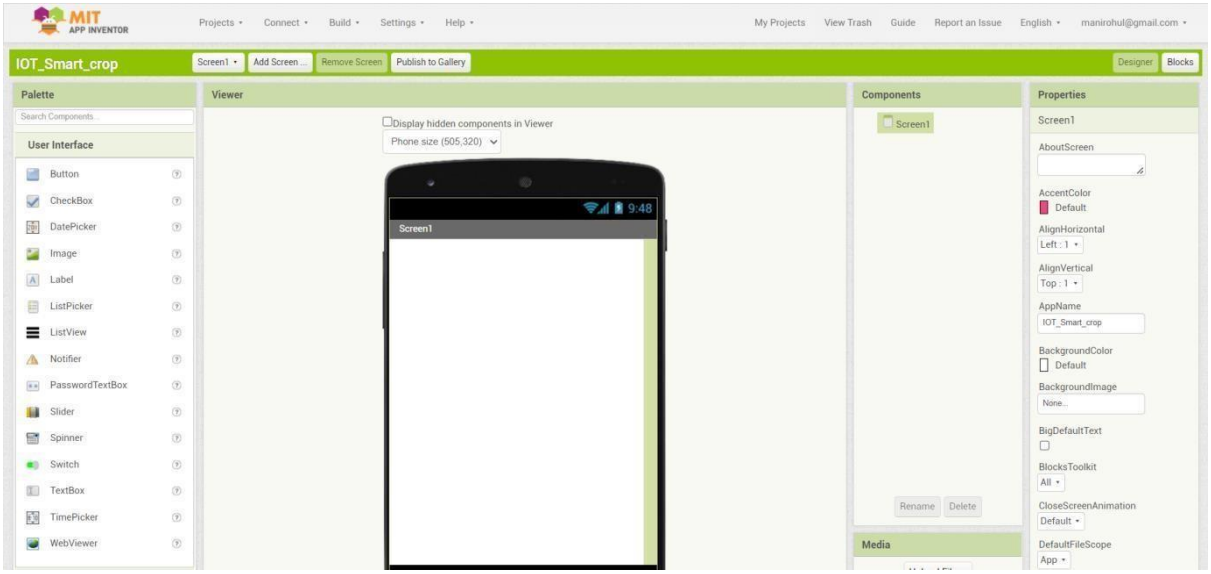
STEP3: Generate the some output from recent events.



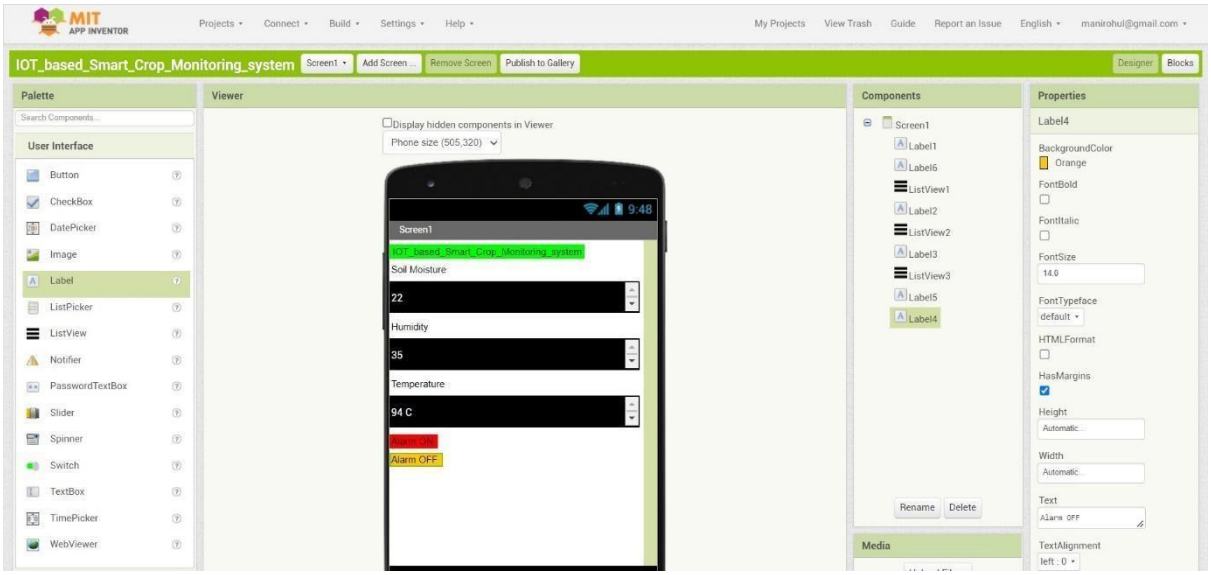


MIT APP INVENTOR:

STEP 1: MIT APP inventor to design the APP.



STEP 2: Customize the App interface to Display the Values.



-