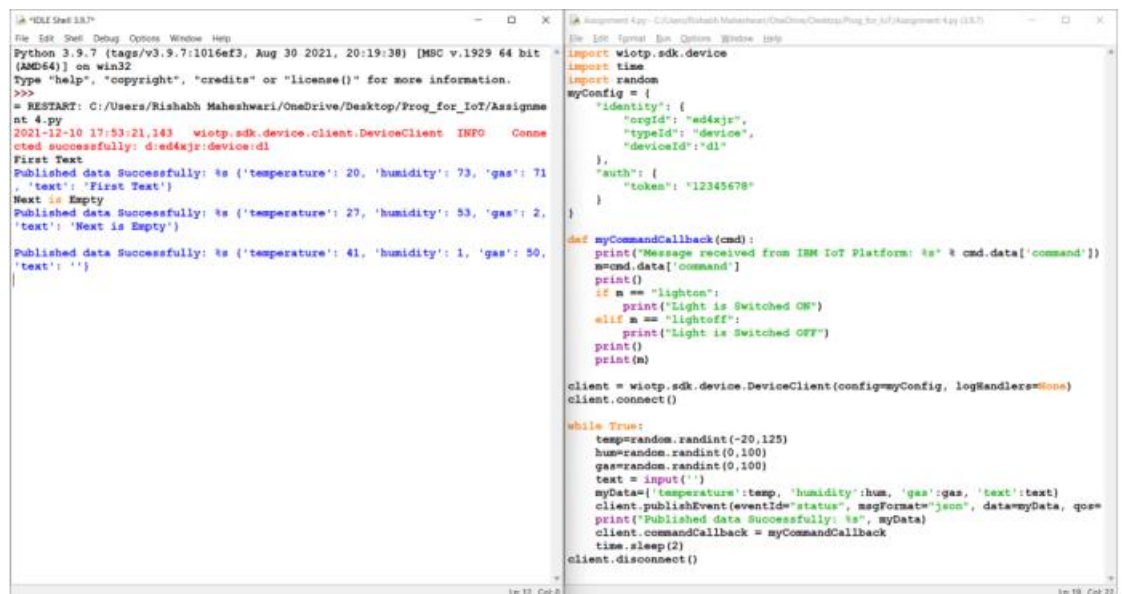


Assignment 4

Anjali Singh 19BCG10003

Develop a mobile application that takes the user input and sends it to IoT device (python code). print the received data in python shell. Keep a text box to accept the user input. Integrate a submit button. Whenever user enters the text input in text box and clicks the button the data should be sent to IBM cloud using URL (HTTP API).

- The Program that we have developed would work and update the Temperature, Humidity and Gas variables continuously while if the user enters some text, then it would be visible on the IDLE Shell.



The image shows two windows from a Python IDE. The left window is the IDLE Shell, displaying the execution of a script. It shows a restart message, connection status, and three successful data publications with JSON payloads for temperature, humidity, and gas. The right window shows the source code of the script. The code imports the `wiottp` library, defines a configuration dictionary with device credentials, and implements a `myCommandCallback` function to handle incoming commands like 'lighton' and 'lightoff'. A `while` loop continuously sends random sensor data and prompts for user input, which is then sent to the IoT platform via the `publishEvent` method.

```
Python 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021, 20:19:38) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Rishabh Maheshwari/OneDrive/Desktop/Prog_for_IoT/Assignment 4.py
2021-12-10 17:53:21.143 wiottp.sdk.device.client.DeviceClient INFO Connected successfully: d-ed4kjr:device:d1
First Text
Published data Successfully: %s ('temperature': 20, 'humidity': 73, 'gas': 71, 'text': 'First Text')
Next is Empty
Published data Successfully: %s ('temperature': 27, 'humidity': 53, 'gas': 2, 'text': 'Next is Empty')
Published data Successfully: %s ('temperature': 41, 'humidity': 1, 'gas': 50, 'text': '')

Assignment 4.py - C:/Users/Rishabh Maheshwari/OneDrive/Desktop/Prog_for_IoT/Assignment 4.py (33,7)
from wiottp.sdk.device import DeviceClient
import time
import random

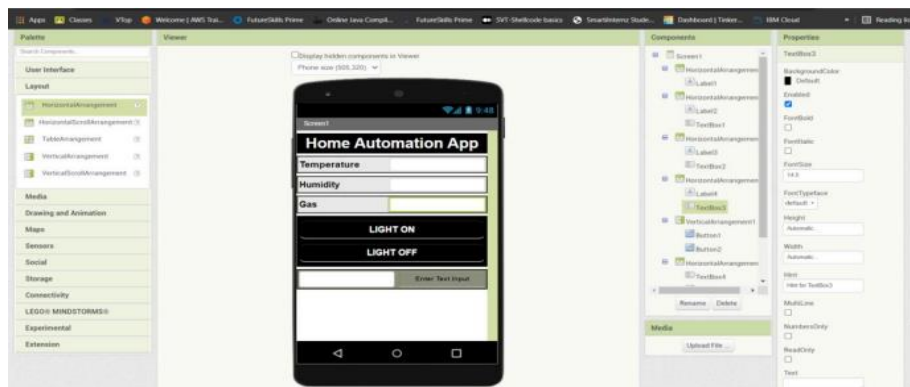
myConfig = {
    'identity': {
        'orgId': "ed4kjr",
        'typeId': "device",
        'deviceId': "d1"
    },
    'auth': {
        'token': "12345678"
    }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']
    print()
    if m == "lighton":
        print("Light is Switched ON")
    elif m == "lightoff":
        print("Light is Switched OFF")
    print()
    print(m)

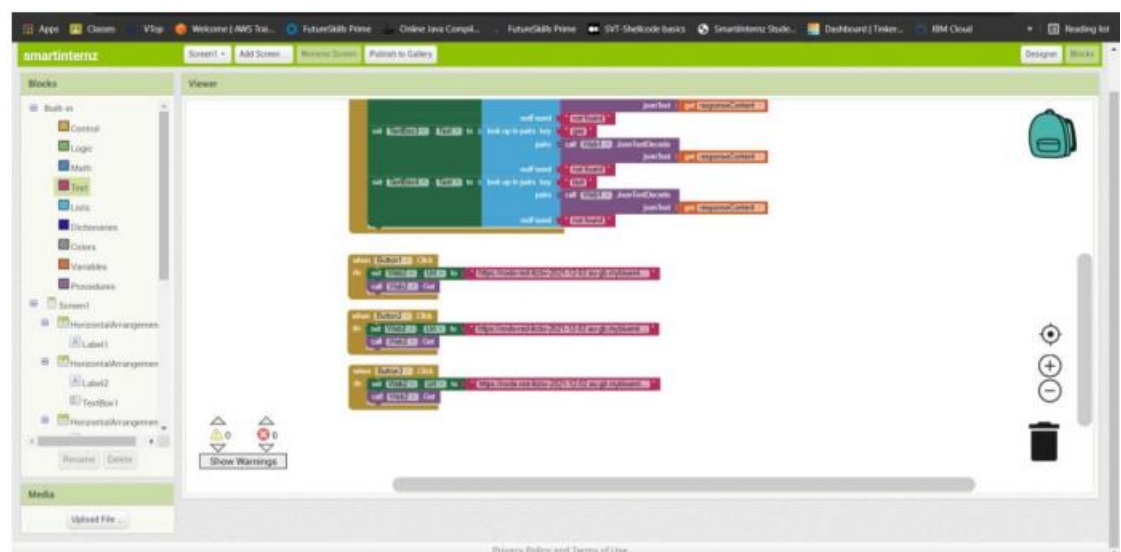
client = wiottp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

while True:
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    gas=random.randint(0,100)
    text = input('')
    myData={'temperature':temp, 'humidity':hum, 'gas':gas, 'text':text}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
    client.disconnect()
```

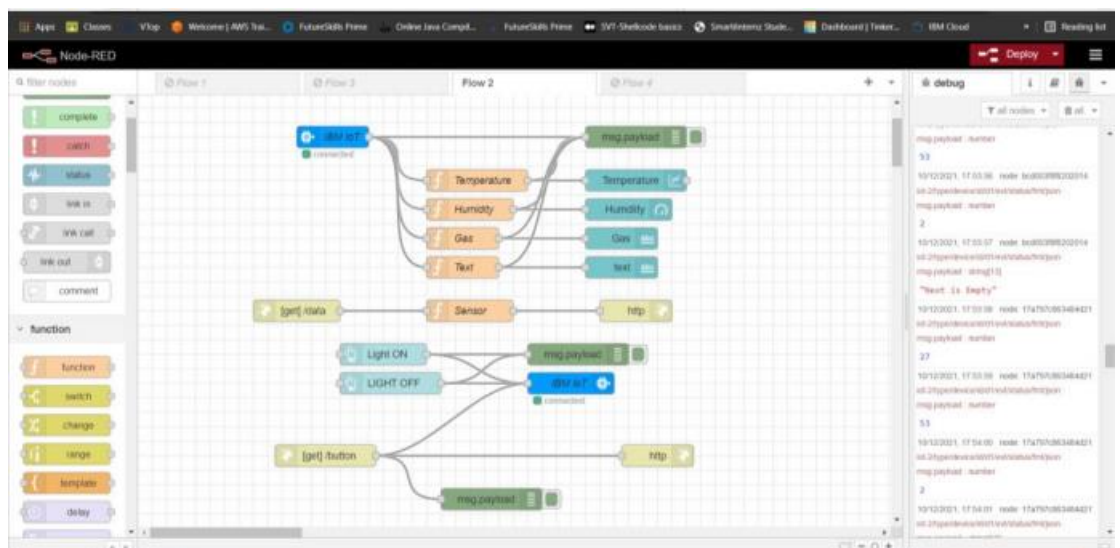
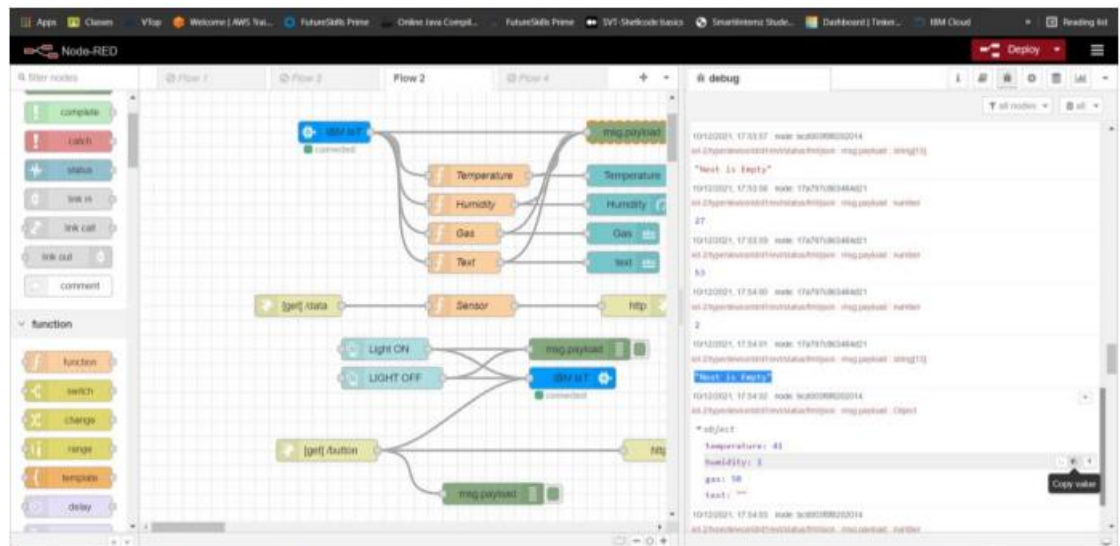
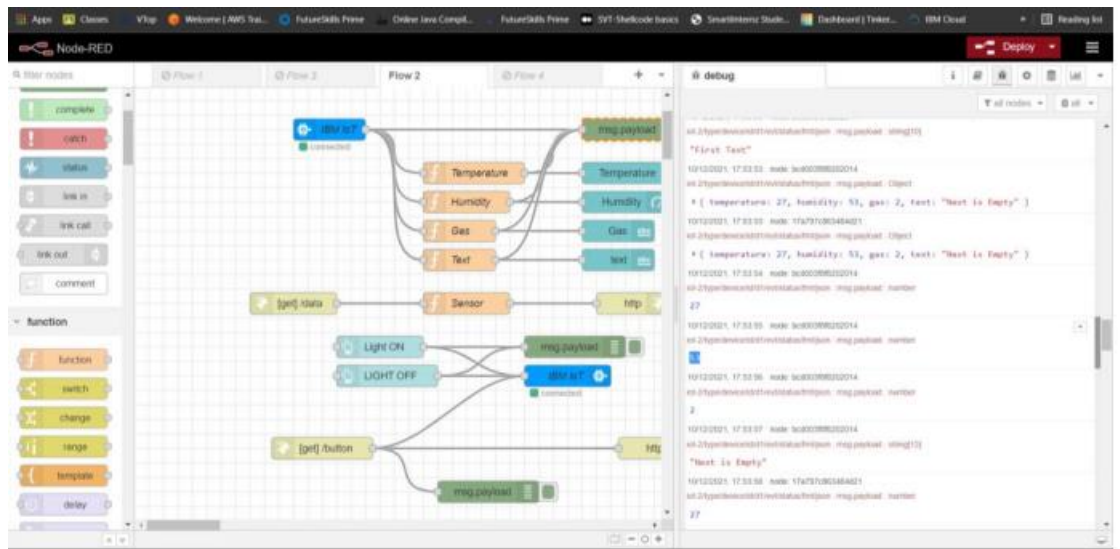
- Using MIT App Inventor first design the mobile app layout.

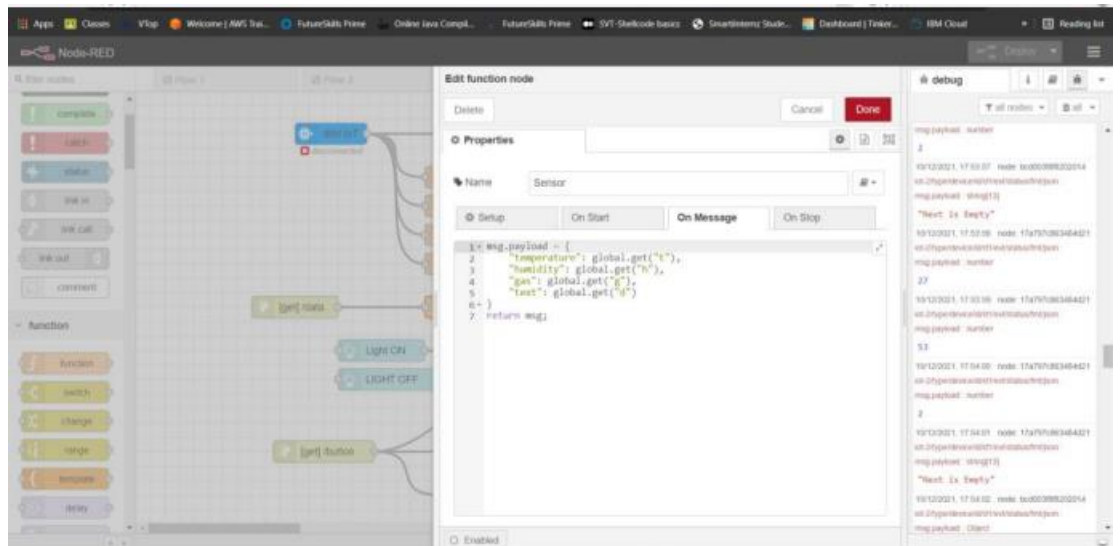


-

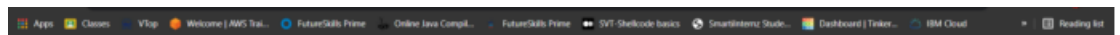


- Configure the Data Flow and Structure of the program on Node-RED application. Output will also be available on Node-RED debug console as shown below.





- Output available on web page using URLs.



```
{"command": "First Text"}
```