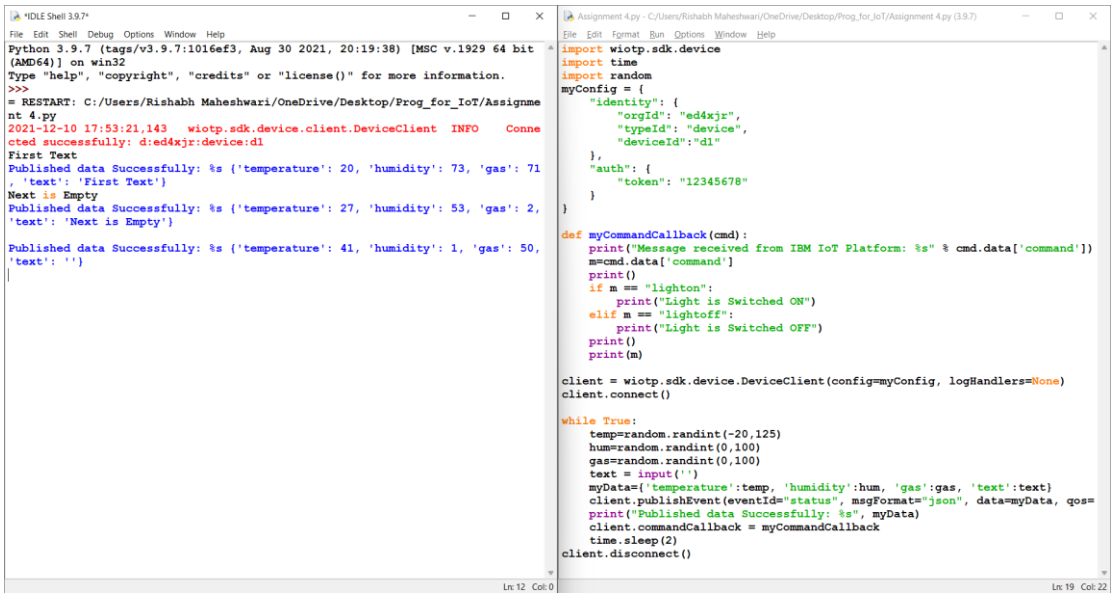
Assignment 4

Harsh Agarwal
19BCE10093

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).

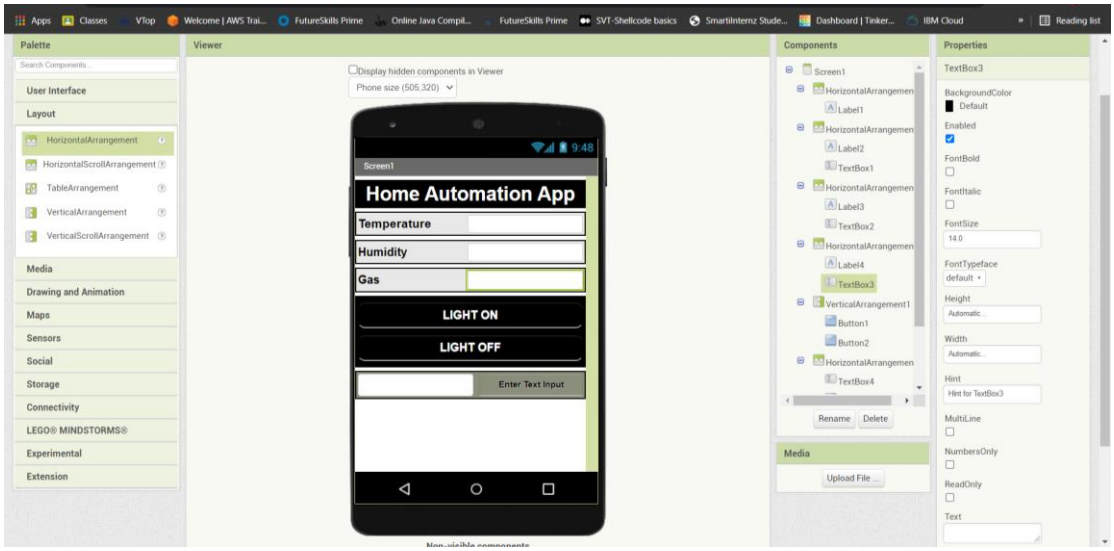
1. The Program that we have developed would work and update the Temperature, Humidity and Gas variables continuously while if the user enters some text then it would be visible on the IDLE Shell.



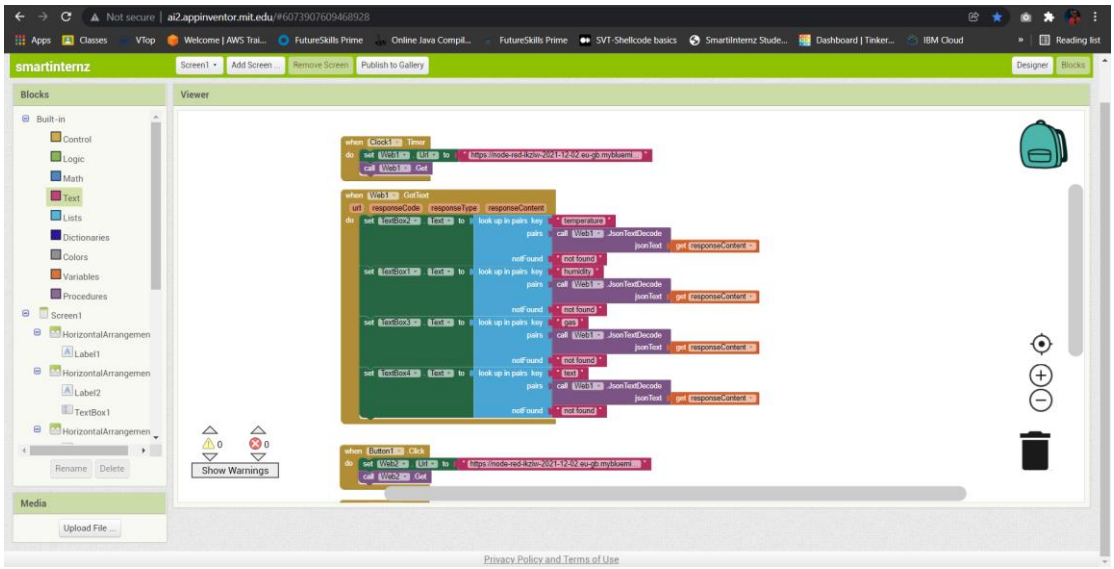
```
*IDLE Shell 3.9.7*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.9.7 (tags/v3.9.7:1016ef3, Aug 30 2021, 20:19:38) [MSC v.1929 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Rishabh Maheshwari/OneDrive/Desktop/Prog_for_IoT/Assignme
nt 4.py
2021-12-10 17:53:21,143   wiotp.sdk.device.client.DeviceClient  INFO    Conne
cted successfully: d:ed4xjr: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': ''}
```

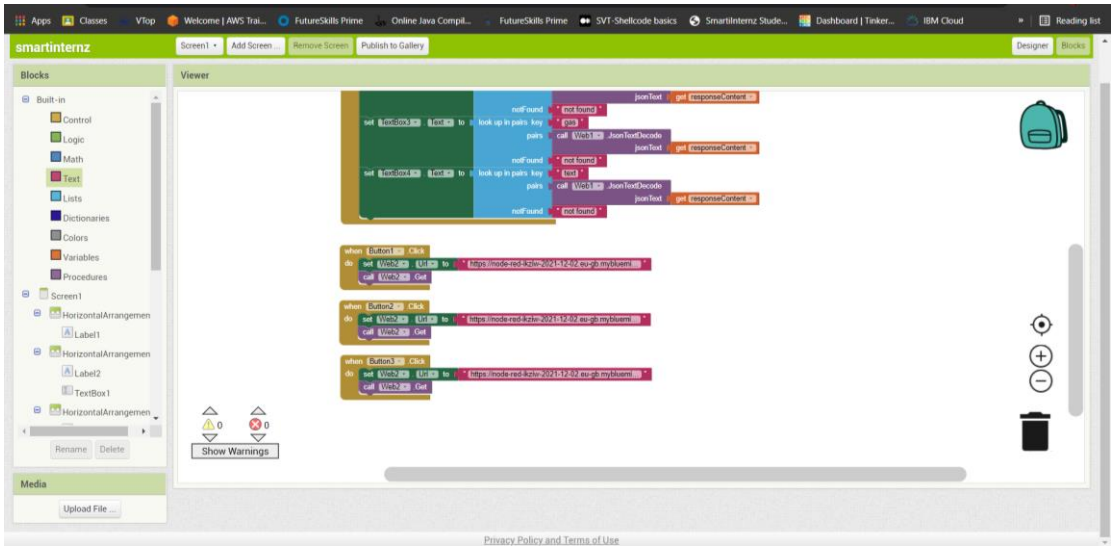
```
Assignment 4.py - C:/Users/Rishabh Maheshwari/OneDrive/Desktop/Prog_for_IoT/Assignment 4.py (3.9.7)
File  Edit  Format  Run  Options  Window  Help
import wiotp.sdk.device
import time
import random
myConfig = {
    "identity": {
        "orgId": "ed4xjr",
        "typeId": "device",
        "deviceId":"d1"
    },
    "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 = wiotp.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=
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
client.disconnect()
```

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

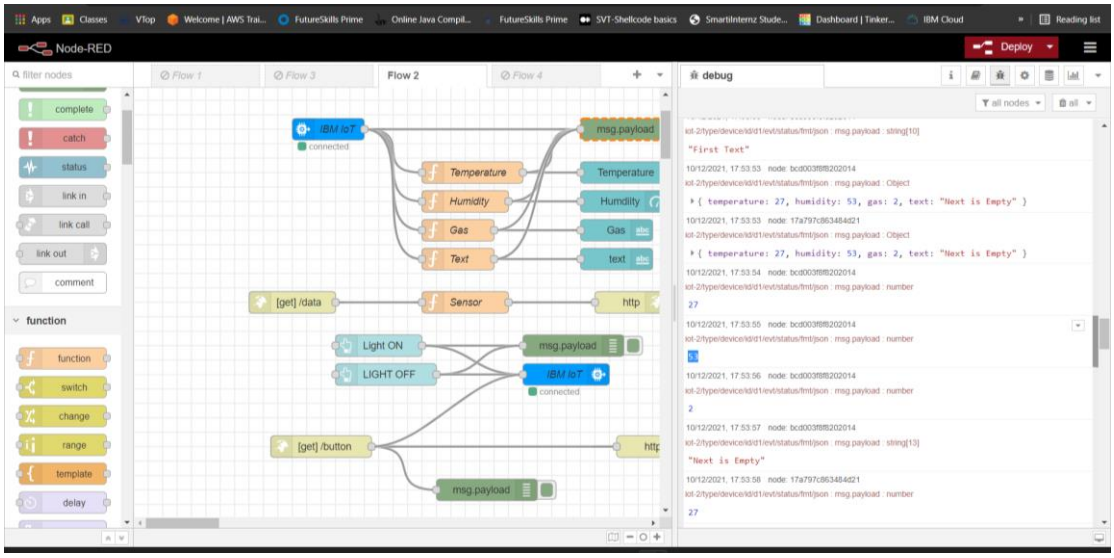


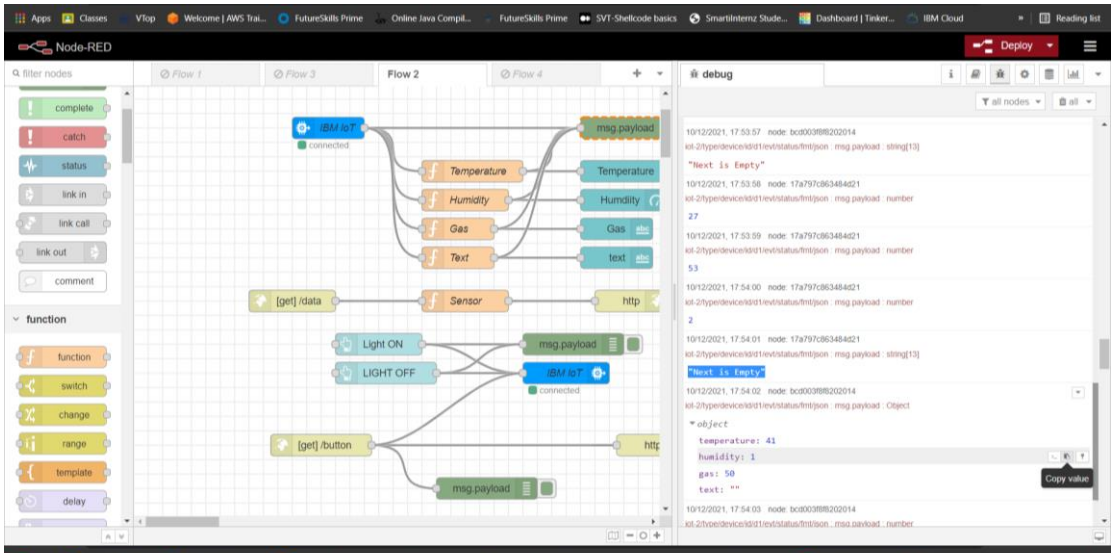
3. Design the working of the app using the Blocks available in the MIT App Inventor app.

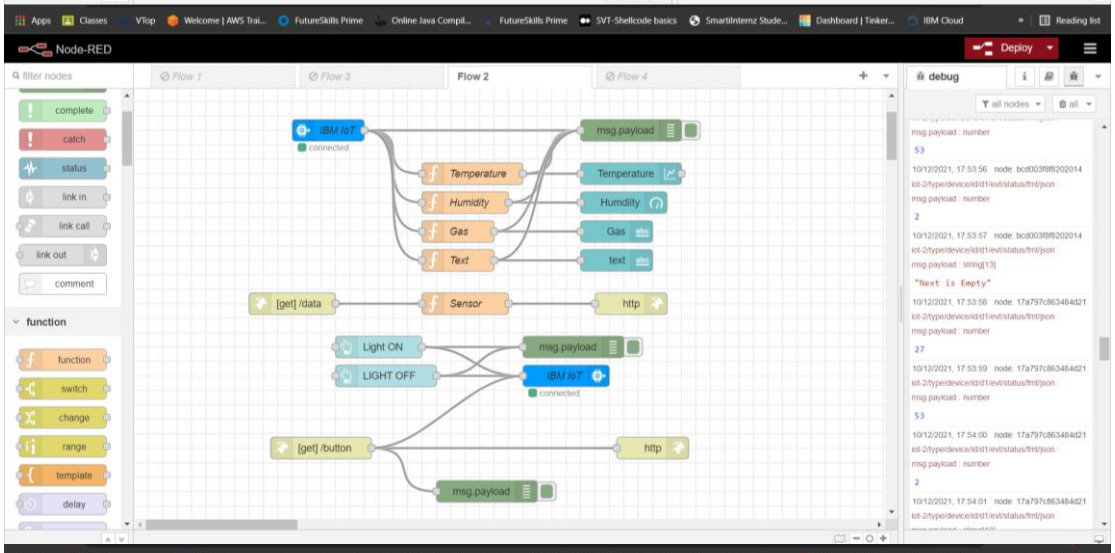


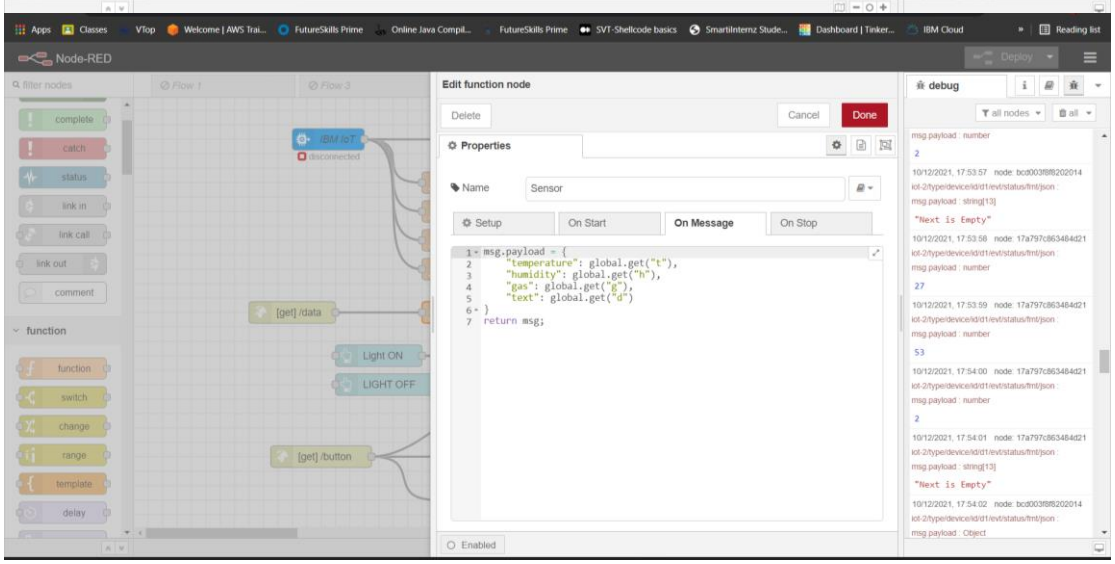


4. Configure the Data Flow and Structure of the program on Node-RED application. Output will also be available on Node-RED debug console as shown below.

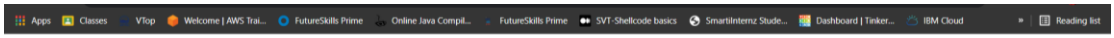



Node-RED
Deploy
Flow 1
Flow 3
Flow 2
Flow 4
debug
IBM IoT
connected
msg.payload
Temperature
Humidity
Gas
Text
[get] /data
Sensor
http
Light ON
LIGHT OFF
[get] /button
"Next is Empty"
object
temperature: 41
humidity: 1
gas: 50
text: ""
Copy value


Node-RED
Deploy
Flow 1
Flow 3
Flow 2
Flow 4
debug
IBM IoT
connected
msg.payload
Temperature
Humidity
Gas
Text
[get] /data
Sensor
http
Light ON
LIGHT OFF
[get] /button


Node-RED
Deploy
Edit function node
Delete
Cancel
Done
Properties
Name
Sensor
Setup
On Start
On Message
On Stop
msg.payload = {
"temperature": global.get("t"),
"humidity": global.get("h"),
"gas": global.get("g"),
"text": global.get("d")
}
return msg;
Enabled
debug
IBM IoT
disconnected
[get] /data
Light ON
LIGHT OFF
[get] /button

5. Output available on web page using URLs.



{"command":"First Text"}