

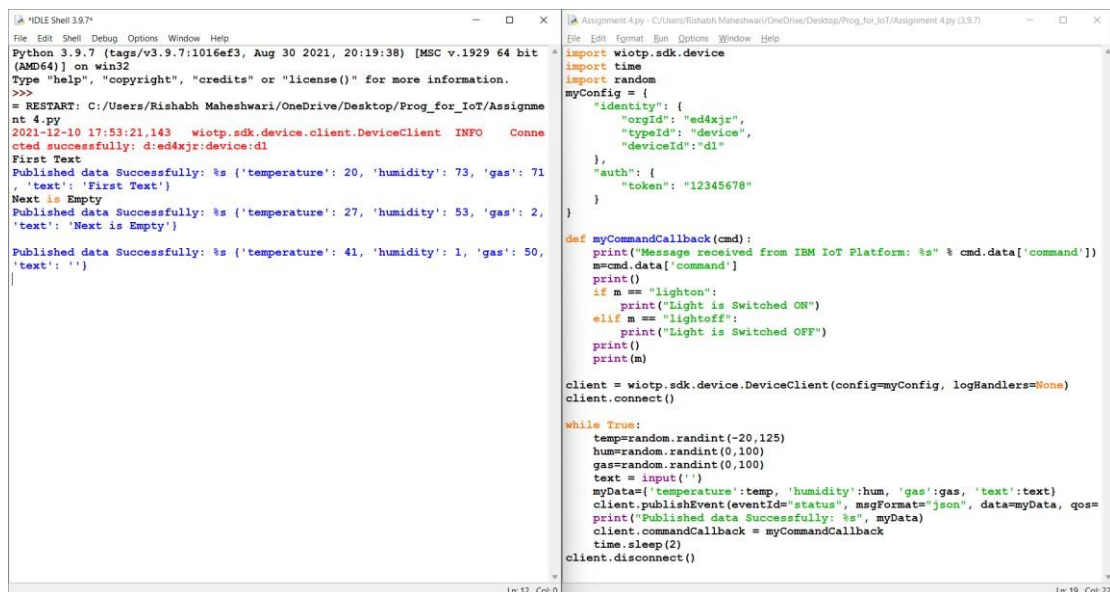
Assignment 4

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.



```
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 wiotpsdk.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': '')

import wiotpsdk.device
import time
import random

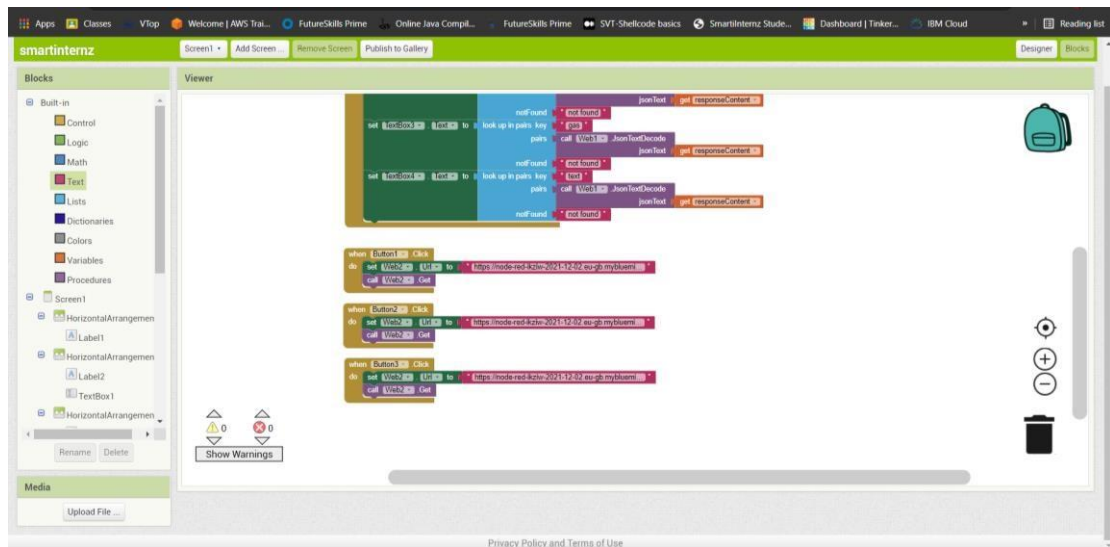
myConfig = {
    "identity": {
        "orgId": "ed4xjr",
        "typeId": "device",
        "deviceId": "d1"
    },
    "auth": {
        "token": "12345678"
    }
}

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)

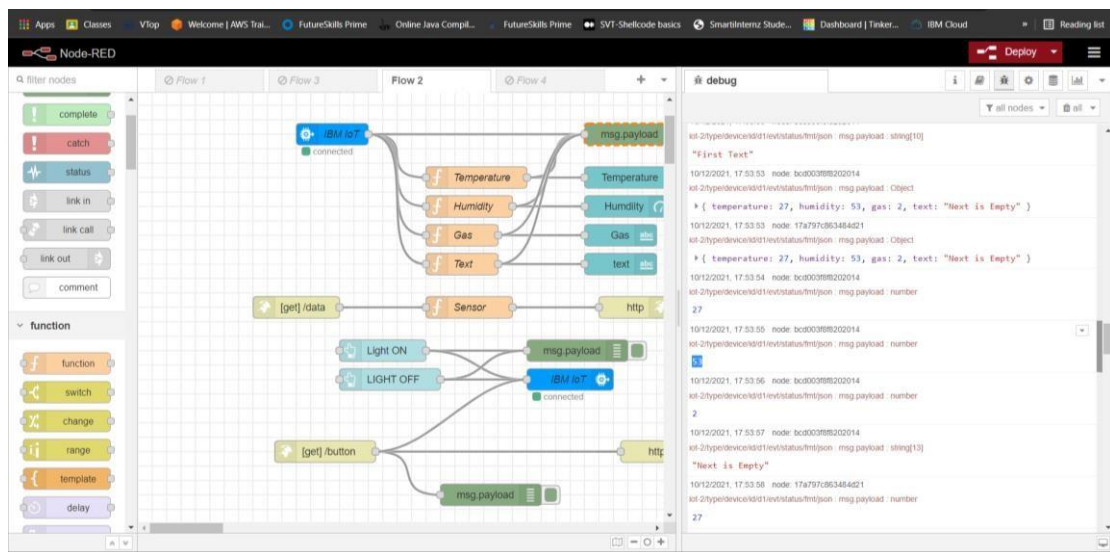
client = wiotpsdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

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.



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 interface showing a flow diagram and a debug console.

Flow Diagram:

- Flow 1:** `IBM IoT` node connected to `msg.payload`.
- Flow 2:** `IBM IoT` node connected to `Temperature`, `Humidity`, `Gas`, and `Text` nodes. These nodes are connected to `msg.payload`.
- Flow 3:** `Light ON` and `LIGHT OFF` nodes connected to `msg.payload`.
- Flow 4:** `Light ON` and `LIGHT OFF` nodes connected to `IBM IoT` node.
- Flow 5:** `[get] /data` node connected to `Sensor` node, which is connected to `http` node.
- Flow 6:** `[get] /button` node connected to `msg.payload` and `http` node.

Debug Console:

```
10/12/2021, 17:53:57 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: string[13]  
"Next is Empty"  
10/12/2021, 17:53:58 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
27  
10/12/2021, 17:53:59 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
53  
10/12/2021, 17:54:00 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
2  
10/12/2021, 17:54:01 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: string[13]  
"Next is Empty"  
10/12/2021, 17:54:02 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: Object  
{"temperature": 41, "humidity": 1, "gas": 50, "text": ""}  
10/12/2021, 17:54:03 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number
```

Node-RED interface showing a flow diagram and a debug console.

Flow Diagram:

- Flow 1:** `IBM IoT` node connected to `msg.payload`.
- Flow 2:** `IBM IoT` node connected to `Temperature`, `Humidity`, `Gas`, and `Text` nodes. These nodes are connected to `msg.payload`.
- Flow 3:** `Light ON` and `LIGHT OFF` nodes connected to `msg.payload`.
- Flow 4:** `Light ON` and `LIGHT OFF` nodes connected to `IBM IoT` node.
- Flow 5:** `[get] /data` node connected to `Sensor` node, which is connected to `http` node.
- Flow 6:** `[get] /button` node connected to `msg.payload` and `http` node.

Debug Console:

```
msg.payload: number  
53  
10/12/2021, 17:53:56 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
2  
10/12/2021, 17:53:57 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: string[13]  
"Next is Empty"  
10/12/2021, 17:53:58 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
27  
10/12/2021, 17:53:59 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
53  
10/12/2021, 17:54:00 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
2  
10/12/2021, 17:54:01 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: Object
```

Node-RED interface showing a flow diagram, a debug console, and an edit function node dialog.

Flow Diagram:

- Flow 1:** `IBM IoT` node connected to `msg.payload`.
- Flow 2:** `IBM IoT` node connected to `Temperature`, `Humidity`, `Gas`, and `Text` nodes. These nodes are connected to `msg.payload`.
- Flow 3:** `Light ON` and `LIGHT OFF` nodes connected to `msg.payload`.
- Flow 4:** `Light ON` and `LIGHT OFF` nodes connected to `IBM IoT` node.
- Flow 5:** `[get] /data` node connected to `Sensor` node, which is connected to `http` node.
- Flow 6:** `[get] /button` node connected to `msg.payload` and `http` node.

Debug Console:

```
msg.payload: number  
2  
10/12/2021, 17:53:57 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: string[13]  
"Next is Empty"  
10/12/2021, 17:53:58 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
27  
10/12/2021, 17:53:59 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
53  
10/12/2021, 17:54:00 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: number  
2  
10/12/2021, 17:54:01 node: 17a797c863484d21  
iot-2hypercendv01devstatus/rnt/json - msg.payload: string[13]  
"Next is Empty"  
10/12/2021, 17:54:02 node: bc003998202014  
iot-2hypercendv01devstatus/rnt/json - msg.payload: Object
```

Edit function node dialog:

Name: Sensor

Setup: On Start On Message On Stop

```
1: msg.payload = {  
2:   "temperature": global.get("t"),  
3:   "humidity": global.get("h"),  
4:   "gas": global.get("g"),  
5:   "text": global.get("d")  
6: }  
7: return msg;
```

5. Output available on web page using URLs.

A screenshot of a web browser window with a dark theme. The address bar shows a URL. The page content displays a JSON object: {"command": "First Text"}. The browser's tab bar at the top shows several open tabs, including 'Apps', 'Classes', 'Vtop', 'Welcome | AWS Tra...', 'FutureSkills Prime', 'Online Java Compil...', 'FutureSkills Prime', 'SVT-Shellcode basics', 'SmartInternz Shade...', 'Dashboard | Tinker...', 'IBM Cloud', and 'Reading list'.

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

Daksh Pathak

19BCG10043