8/12/2020

IOT ANALYTICS IN HEALTH MONITORING

DONE BY,
NEHA SATISH

1) INTRODUCTION

1.1 Overview

Health Monitor is used to check if a person is following a healthy regime. It is used by collecting data on regular components of health such as body temperature or water level in the body.

1.2 Purpose

More skilful patient administration can help use the assets of the clinic all the more astutely and set aside cash. It is simpler to utilize the framework for patients and clinical experts. The checking framework is particularly helpful to screen patients with interminable sicknesses. Most ailments are serious, so it is important to screen the condition of the patient while at home, and rapidly react if well-being markers compound.

2) LITERATURE SURVEY

2.1 Existing problem

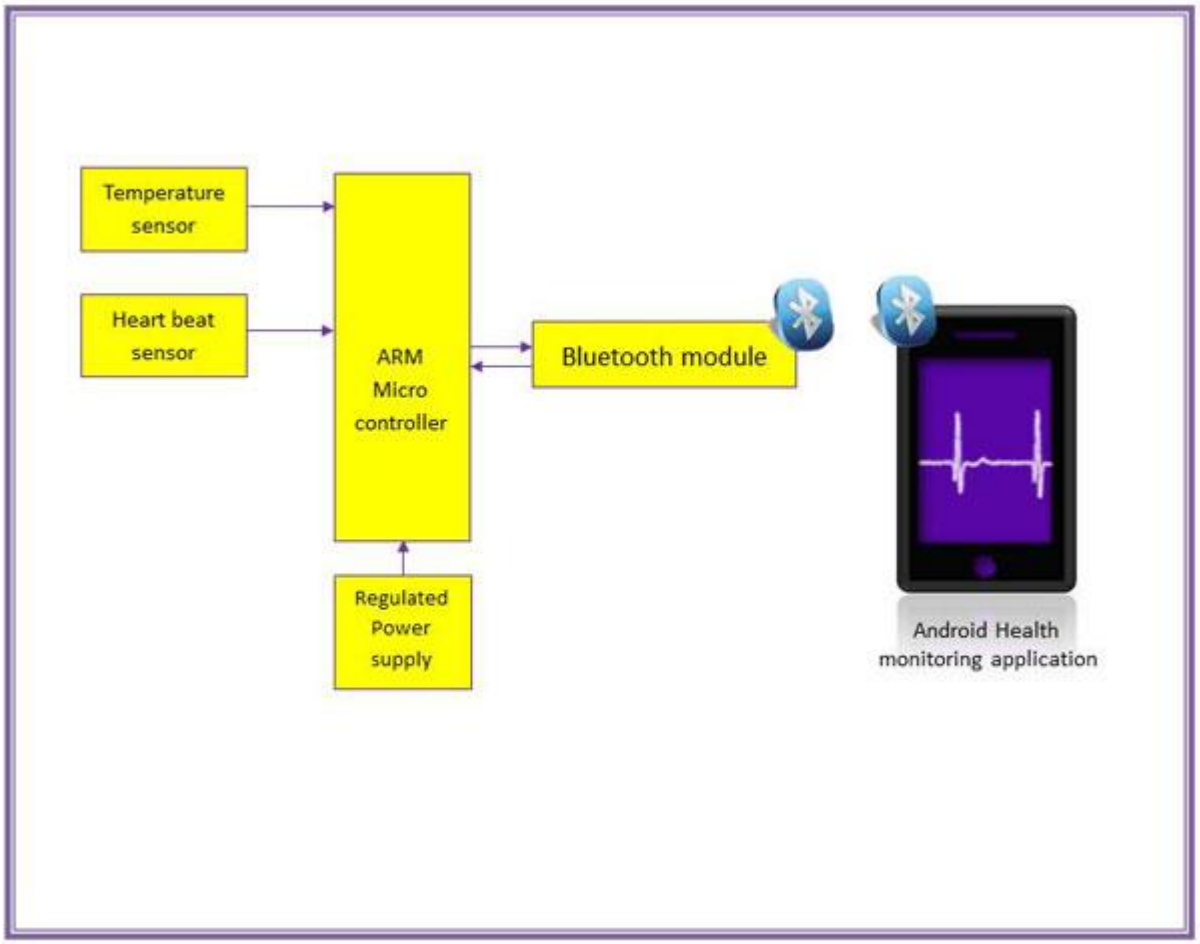
Many people run to the hospitals for a regular health check-up. Due to the Covid19 Pandemic, most of the patients couldn't visit the hospital for regular check-ups. This has increased panic in people and also ailments which people do not find cure to.

2.2 Proposed solution

The health Monitor is built to give people a better exposure to hoe they can have a regular check-up at home. Using the health monitor they can check their body temperature and know if their water levels are proper, and if not get appropriate cure.

3) THEORITICAL ANALYSIS

3.1 Block diagram



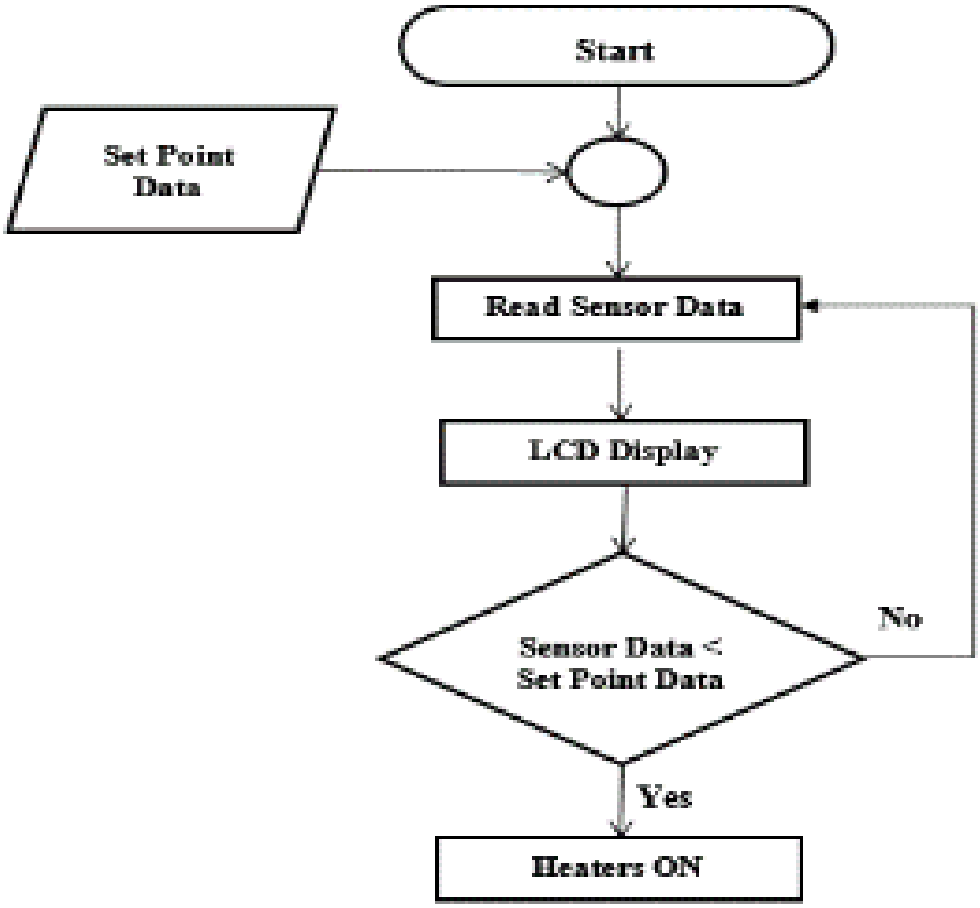
3.2 Software designing

Software components used are : 1.IBM cloud 2.IBM IOT platform 3.IBM Watson 4.node-red 5.Tinker cad 6.Python IDLE 7.MIT app inventor

4) EXPERIMENTAL INVESTIGATIONS

People who felt sick, immediately checked their temperature using the health monitor and found cure.

5) FLOWCHART



6) RESULT

Health monitor was built using IBM watson assistant, IOT sensor simulator to check the health parameters of a person.

7) ADVANTAGES & DISADVANTAGES

PROS:

- Many people can be helped at the same time in a short while.
- Health Monitor cannot spread virus and diseases to the people.

CONS:

- The Health monitor requires Internet and power supply. In most rural and remote areas, reliable source of power is a major challenge.

8) APPLICATIONS

- Allows sending data from patients to health professionals in real time
- Improves patients' lifestyle.
- Makes healthcare more available
- Saves money.
- Timely detection and action for specific conditions which require quick attention.
- Assisted and rapid diagnoses that may help arrive at logical conclusions.
- Reduction in hospitalization and related time, effort, and costs.
- Better adherence to the medication schedule.
- Home or familiar premises may be more amenable for several patients than hospitals.

9) CONCLUSION

Health monitor app is available on your phones. Any time, any place it can be accessible. It is very feasible and eco-friendly. This helps a lot of people to get their regular medical check-ups done and follow a healthy regime.

10) FUTURE SCOPE

With more High-end hardware and software, the Health Monitor can be customized and can be upgraded and improved for more efficiency and success rate.

11) BIBILOGRAPHY

a.Github

b.ubuntupit

c.youtube

d.smart internz

12) **APPENDIX**

A. SOUURCE CODE

1. DHT HUMIDITY AND TEMPERATURE:

```
#include <dht.h>
#define dht_apin A0

dht DHT;

void setup(){

  Serial.begin(9600);
  delay(500);
  Serial.println("DHT11 Humidity & temperature Sensor\n\n");
  delay(1000);

}

void loop(){
    DHT.read11(dht_apin);
    Serial.print("Current humidity = ");
    Serial.print(DHT.humidity);
    Serial.print("%  ");
    Serial.print("temperature = ");
    Serial.print(DHT.temperature);
    Serial.println("C  ");
    delay(5000);

}
```

2. BUZZER ALARM:

```
// Declaring Pins
const int buzzerPin = 5;
const int ledPin = 6;
const int motionPin = 7;
const int buttonPin = 12;

// Setting Buzzer mode to False
```

```
boolean buzzer_mode = false;

// For LED
int ledState = LOW;
long previousMillis = 0;
long interval = 100;  // Interval at which LED blinks

void setup()
{
  //The Following are our output
  pinMode(ledPin,OUTPUT);
  pinMode(buzzerPin,OUTPUT);

  //Button is our Input
  pinMode(buttonPin, INPUT);

  // Wait before starting the alarm
  delay(5000);
}

void loop()
{
  // To chech whether the motion is detected or not
  if (digitalRead(motionPin)) {
    buzzer_mode = true;
  }

  // If alarm mode is on,blink our LED
  if (buzzer_mode){
    unsigned long currentMillis = millis();
    if(currentMillis - previousMillis > interval) {
      previousMillis = currentMillis;
      if (ledState == LOW)
        ledState = HIGH;
      else
        ledState = LOW;
    // Switch the LED
    digitalWrite(ledPin, ledState);
    }
    tone(buzzerPin,1000);
  }

  // If alarm is off
```

```
  if (buzzer_mode == false) {

    // No tone & LED off
    noTone(buzzerPin);
    digitalWrite(ledPin, LOW);
  }

  // If our button is pressed Switch off ringing and Setup
  int button_state = digitalRead(buttonPin);
  if (button_state) {buzzer_mode = false;}
}
```

3. Connecting NodeMCU to WIFI:

```
#include<ESP8266Wifi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
const char* ssid = "<yourWIFIssid>";
const char* password = "<yourWIFIpassword>";

#define ORG "xyz1kg"
#define DEVICE_TYPE "<Arduino"
#define DEVICE_ID "<ard123"
#define TOKEN "12345678"
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

const char eventTopic[] = "iot-2/evt/status/fmt/json";
const char cmdTopic[] = "iot-2/cmd/led/fmt/json";



WiFiClient wifiClient;
void callback(char* topic, byte* payload, unsigned int payloadLength)
{
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < payloadLength; i++) {
    Serial.print((char)payload[i]);
  }
```

```
  Serial.println();

  // Switch on the LED if an 1 was received as first character
  if (payload[0] == '1') {
    digitalWrite(BUILTIN_LED, LOW);   // Turn the LED on (Note
that LOW is the voltage level
    // but actually the LED is on; this is because
    // it is acive low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, HIGH);  // Turn the LED off by
making the voltage HIGH
  }

}
PubSubClient client(server, 1883, callback, wifiClient);

int publishInterval = 5000; // 5 seconds//Send adc every 5sc
long lastPublishMillis;

void setup() {
  Serial.begin(9600); Serial.println();
  pinMode(LED_BUILTIN, OUTPUT);
  wifiConnect();
  mqttConnect();
}

void loop() {
  if (millis() - lastPublishMillis > publishInterval) {
    publishData();
    lastPublishMillis = millis();
  }

  if (!client.loop()) {
    mqttConnect();
  }
}

void wifiConnect() {
  Serial.print("Connecting to "); Serial.print(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

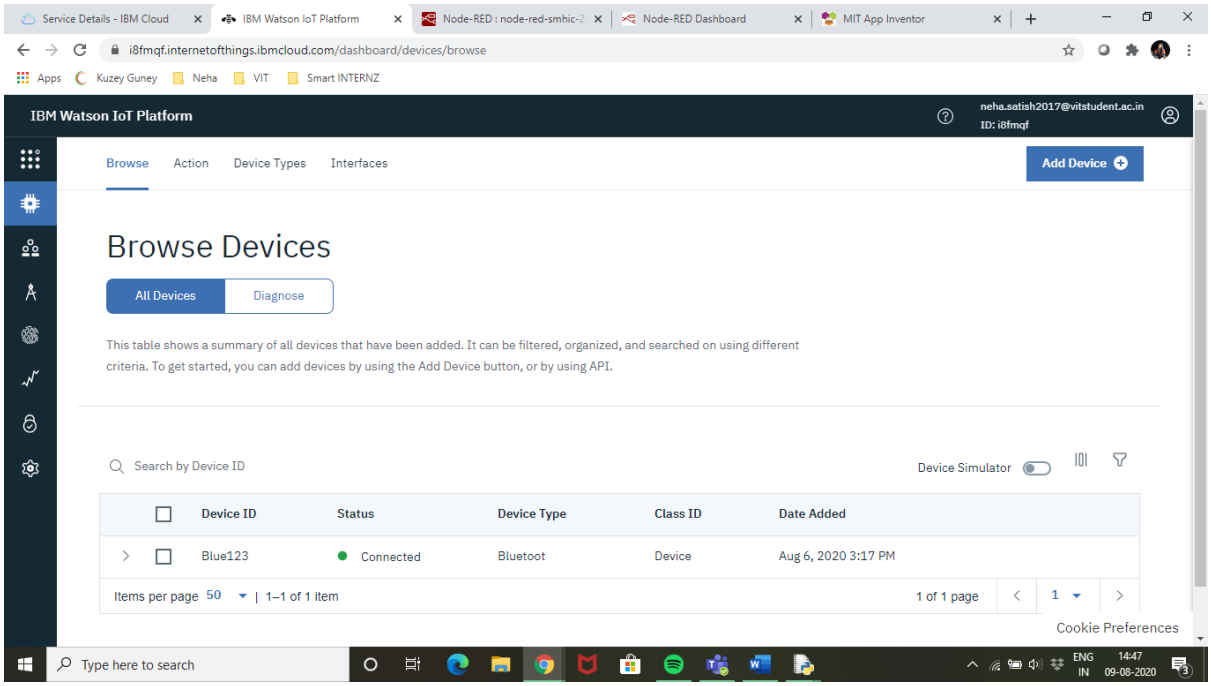
```
  }
  Serial.print("nWiFi connected, IP address: ");
Serial.println(WiFi.localIP());

}

void mqttConnect() {
  if (!!!client.connected()) {
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
    if (client.subscribe(cmdTopic)) {
      Serial.println("subscribe to responses OK");
    } else {
      Serial.println("subscribe to responses FAILED");
    }
    Serial.println();
  }
}


void publishData() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);

  String payload = "{\"d\":{\"adc\":";
  payload += String(sensorValue, DEC);
  payload += "}}";

  Serial.print("Sending payload: "); Serial.println(payload);

  if (client.publish(eventTopic, (char*) payload.c_str())) {
    Serial.println("Publish OK");
  } else {
    Serial.println("Publish FAILED");
  }
}
```

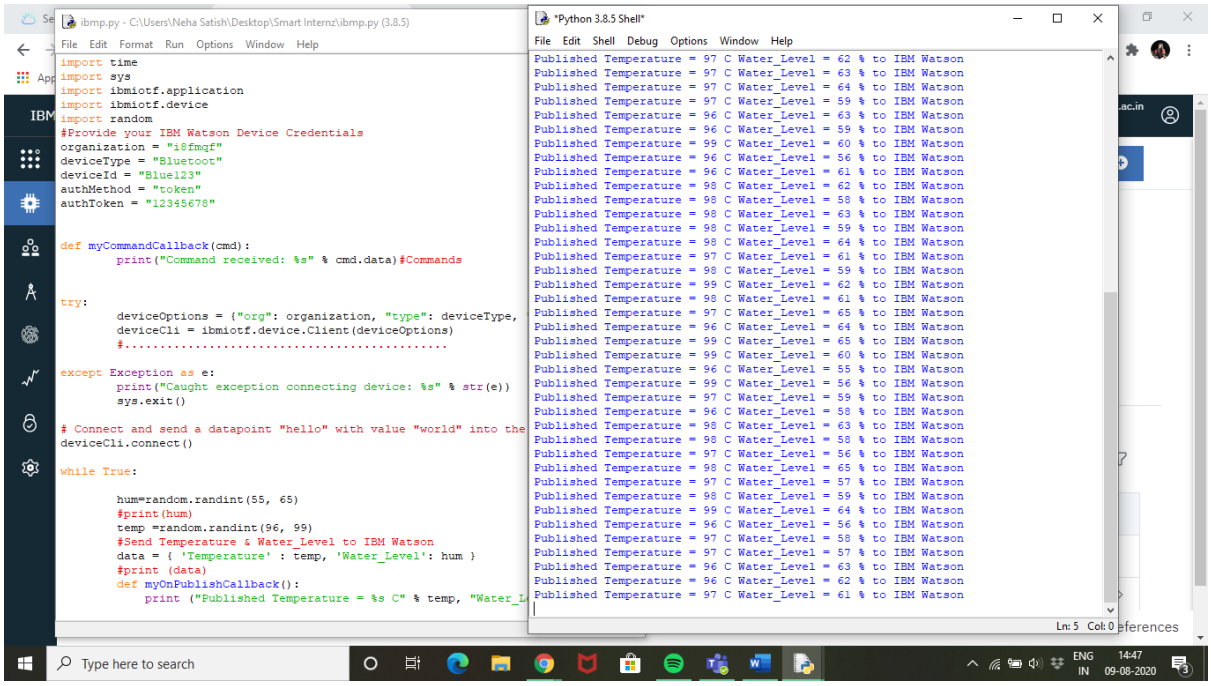
4. For Publishing the data using MQTT:

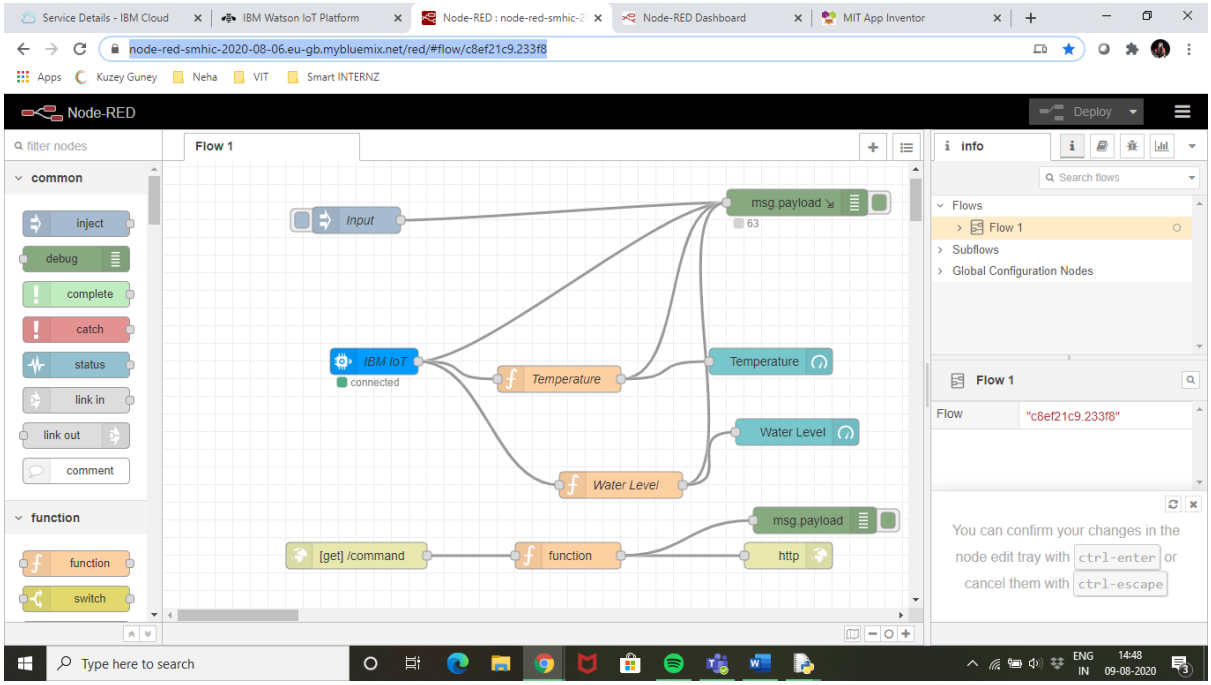
```
int Publish(char* payload, int payload_size) {
  int rc = -1;
  MQTTClient client = {0};
  MQTTClient_connectOptions conn_opts =
MQTTClient_connectOptions_initializer;
  MQTTClient_message pubmsg = MQTTClient_message_initializer;
  MQTTClient_deliveryToken token = {0};

  MQTTClient_create(&client, opts.address, opts.clientid,
                    MQTTCLIENT_PERSISTENCE_NONE, NULL);
  conn_opts.keepAliveInterval = 60;
  conn_opts.cleansession = 1;
  conn_opts.username = k_username;
  conn_opts.password = CreateJwt(opts.keypath, opts.projectid, opts.algorithm);
  MQTTClient_SSLOptions sslopts = MQTTClient_SSLOptions_initializer;

  sslopts.trustStore = opts.rootpath;
  sslopts.privateKey = opts.keypath;
  conn_opts.ssl = &sslopts;

  unsigned long retry_interval_ms = kInitialConnectIntervalMillis;
  unsigned long total_retry_time_ms = 0;
  while ((rc = MQTTClient_connect(client, &conn_opts)) !=
MQTTCLIENT_SUCCESS) {
    if (rc == 3) {  // connection refused: server unavailable
      usleep(retry_interval_ms * 1000);
      total_retry_time_ms += retry_interval_ms;
      if (total_retry_time_ms >= kMaxConnectRetryTimeElapsedMillis) {
        printf("Failed to connect, maximum retry time exceeded.");
        exit(EXIT_FAILURE);
      }
      retry_interval_ms *= kIntervalMultiplier;
      if (retry_interval_ms > kMaxConnectIntervalMillis) {
        retry_interval_ms = kMaxConnectIntervalMillis;
      }
    } else {
      printf("Failed to connect, return code %d\n", rc);
      exit(EXIT_FAILURE);
    }
  }
```

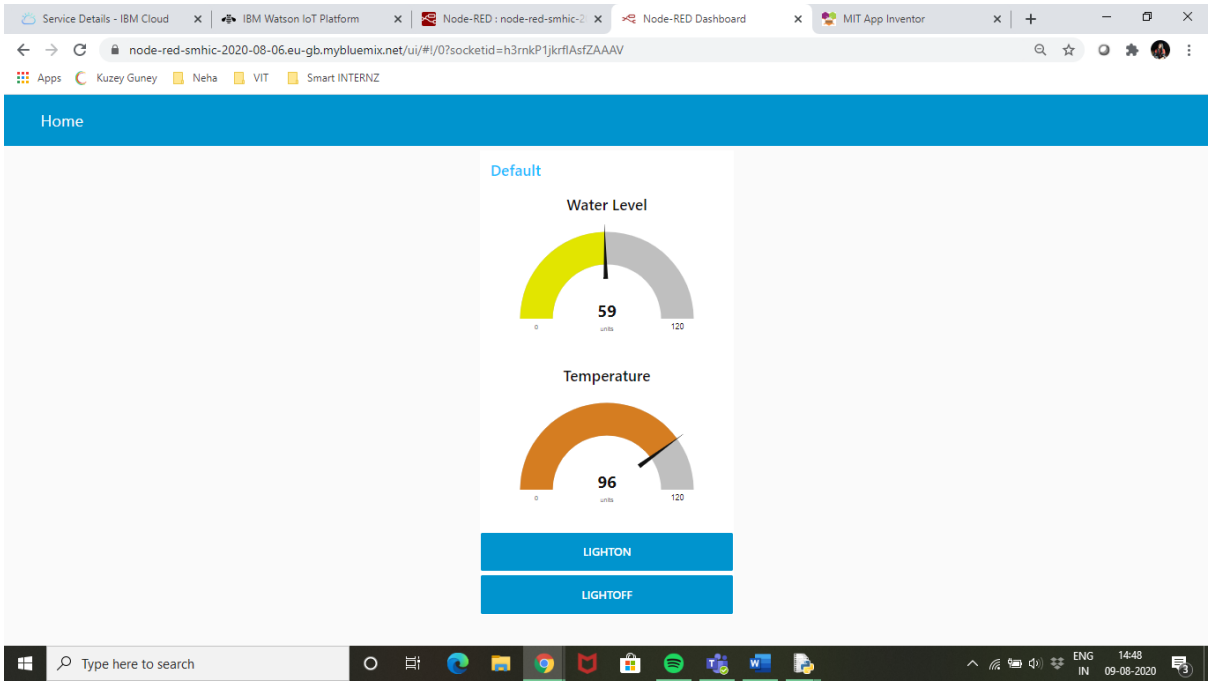
```
 pubmsg.payload = payload;
 pubmsg.payloadlen = payload_size;
 pubmsg.qos = kQos;
 pubmsg.retained = 0;
 MQTTClient_publishMessage(client, opts.topic, &pubmsg, &token);
 printf(
     "Waiting for up to %lu seconds for publication of %s\n"
     "on topic %s for client with ClientID: %s\n",
     (kTimeout / 1000), opts.payload, opts.topic, opts.clientid);
 rc = MQTTClient_waitForCompletion(client, token, kTimeout);
 printf("Message with delivery token %d delivered\n", token);
 MQTTClient_disconnect(client, 10000);
 MQTTClient_destroy(&client);

 return rc;
}
```

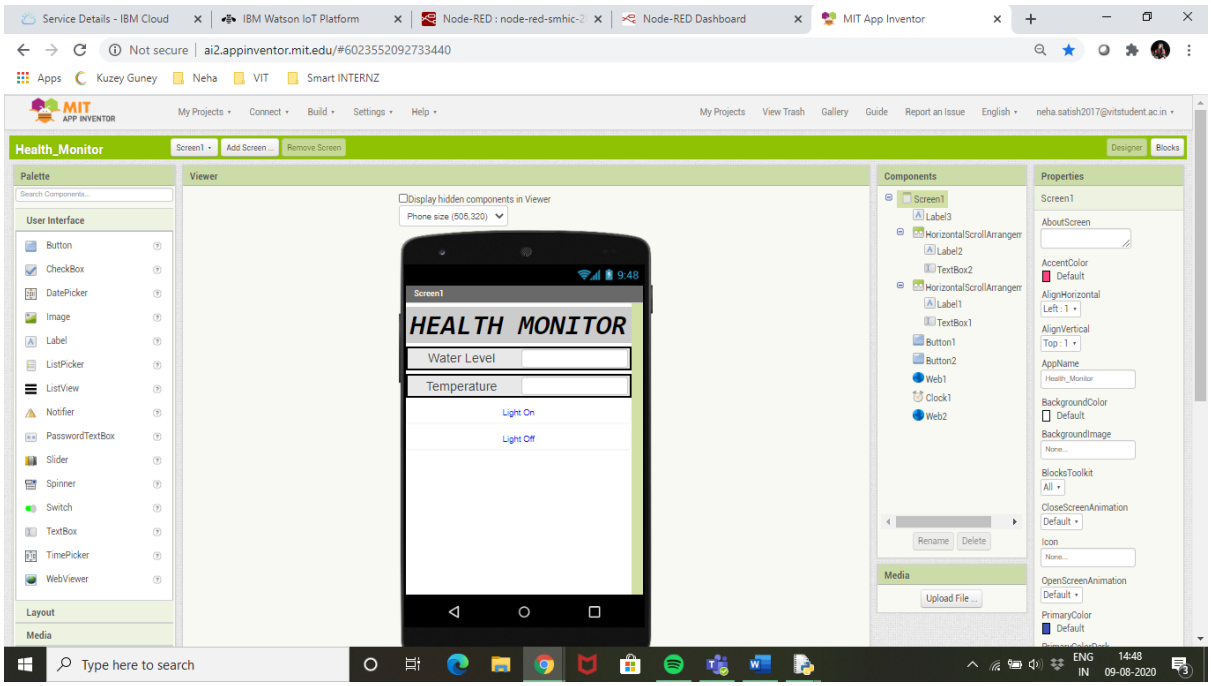
B. UI output Screenshot.

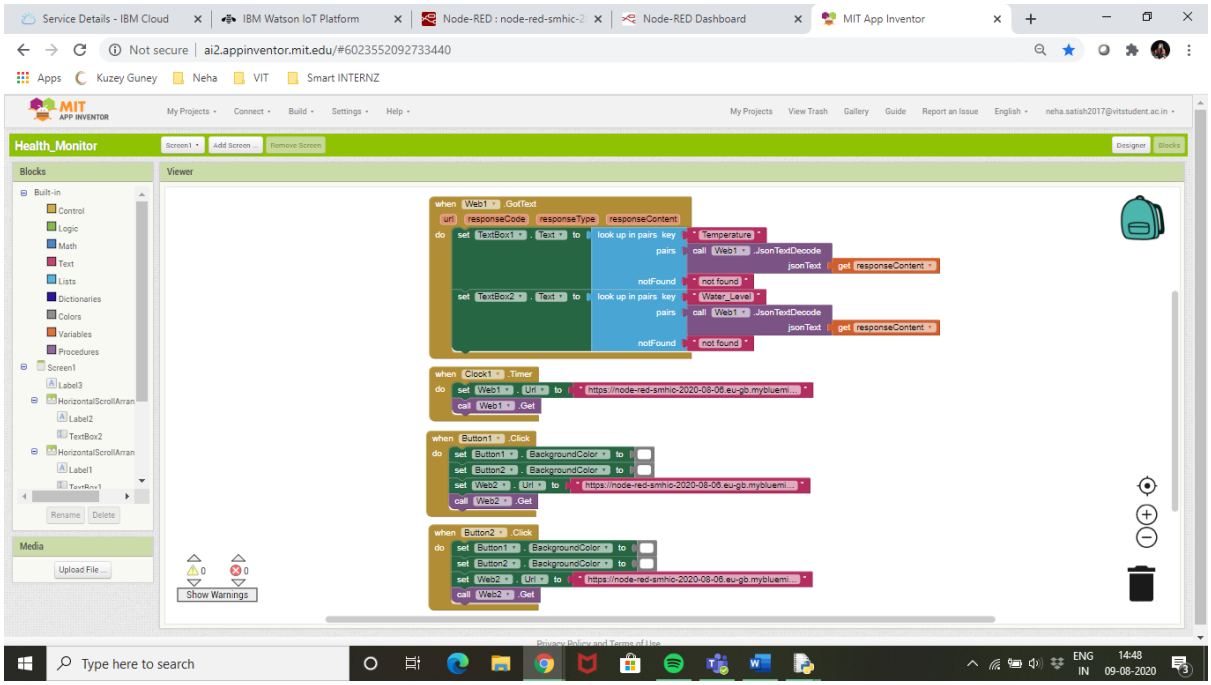



ibmp.py - C:\Users\Neha Satish\Desktop\Smart Interns\ibmp.py (3.8.5)
File Edit Format Run Options Window Help
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "i8fmqf"
deviceType = "Bluetoot"
deviceId = "Blue123"
authMethod = "token"
authToken = "12345678"
def myCommandCallback(cmd):
print("Command received: %s" % cmd.data)#Commands
try:
deviceOptions = {"org": organization, "type": deviceType,
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
deviceCli.connect()
while True:
hum=random.randint(55, 65)
#print(hum)
temp =random.randint(96, 99)
#Send Temperature & Water_Level to IBM Watson
data = { 'Temperature' : temp, 'Water_Level': hum }
#print (data)
def myOnPublishCallback():
print ("Published Temperature = %s C" % temp, "Water_L
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Published Temperature = 97 C Water_Level = 62 % to IBM Watson
Published Temperature = 97 C Water_Level = 63 % to IBM Watson
Published Temperature = 97 C Water_Level = 64 % to IBM Watson
Published Temperature = 97 C Water_Level = 59 % to IBM Watson
Published Temperature = 96 C Water_Level = 63 % to IBM Watson
Published Temperature = 96 C Water_Level = 59 % to IBM Watson
Published Temperature = 99 C Water_Level = 60 % to IBM Watson
Published Temperature = 96 C Water_Level = 56 % to IBM Watson
Published Temperature = 96 C Water_Level = 61 % to IBM Watson
Published Temperature = 98 C Water_Level = 62 % to IBM Watson
Published Temperature = 98 C Water_Level = 58 % to IBM Watson
Published Temperature = 98 C Water_Level = 63 % to IBM Watson
Published Temperature = 98 C Water_Level = 59 % to IBM Watson
Published Temperature = 98 C Water_Level = 64 % to IBM Watson
Published Temperature = 97 C Water_Level = 61 % to IBM Watson
Published Temperature = 98 C Water_Level = 59 % to IBM Watson
Published Temperature = 99 C Water_Level = 62 % to IBM Watson
Published Temperature = 98 C Water_Level = 61 % to IBM Watson
Published Temperature = 97 C Water_Level = 65 % to IBM Watson
Published Temperature = 96 C Water_Level = 64 % to IBM Watson
Published Temperature = 99 C Water_Level = 65 % to IBM Watson
Published Temperature = 99 C Water_Level = 60 % to IBM Watson
Published Temperature = 96 C Water_Level = 55 % to IBM Watson
Published Temperature = 99 C Water_Level = 56 % to IBM Watson
Published Temperature = 97 C Water_Level = 59 % to IBM Watson
Published Temperature = 96 C Water_Level = 58 % to IBM Watson
Published Temperature = 98 C Water_Level = 63 % to IBM Watson
Published Temperature = 98 C Water_Level = 58 % to IBM Watson
Published Temperature = 97 C Water_Level = 56 % to IBM Watson
Published Temperature = 98 C Water_Level = 65 % to IBM Watson
Published Temperature = 97 C Water_Level = 57 % to IBM Watson
Published Temperature = 98 C Water_Level = 59 % to IBM Watson
Published Temperature = 99 C Water_Level = 64 % to IBM Watson
Published Temperature = 96 C Water_Level = 56 % to IBM Watson
Published Temperature = 97 C Water_Level = 58 % to IBM Watson
Published Temperature = 97 C Water_Level = 57 % to IBM Watson
Published Temperature = 96 C Water_Level = 63 % to IBM Watson
Published Temperature = 96 C Water_Level = 62 % to IBM Watson
Published Temperature = 97 C Water_Level = 61 % to IBM Watson
Ln: 5 Col: 0
Type here to search
ENG IN 14:47 09-08-2020


Service Details - IBM Cloud
IBM Watson IoT Platform
Node-RED : node-red-smhic-2
Node-RED Dashboard
MIT App Inventor
node-red-smhic-2020-08-06.eu-gb.mybluemix.net/red/#flow/c8ef21c9.233f8
Apps Kuzey Guney Neha VIT Smart INTERNZ
Node-RED
Deploy
filter nodes
Flow 1
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
Input
msg.payload
63
IBM IoT
connected
Temperature
Temperature
Water Level
Water Level
msg.payload
[get] /command
function
http
info
Search flows
Flows
Flow 1
Subflows
Global Configuration Nodes
Flow 1
Flow
"c8ef21c9.233f8"
You can confirm your changes in the node edit tray with ctrl-enter or cancel them with ctrl-escape
Type here to search
ENG IN 14:48 09-08-2020


Service Details - IBM Cloud
IBM Watson IoT Platform
Node-RED : node-red-smhic-2
Node-RED Dashboard
MIT App Inventor
node-red-smhic-2020-08-06.eu-gb.mybluemix.net/ui/#!/0?socketid=h3rnkP1jkrflAsfZAAAV
Apps
Kuzey Guney
Neha
VIT
Smart INTERNZ
Home
Default
Water Level
59
120
Temperature
96
120
LIGHTON
LIGHTOFF
Type here to search
ENG
IN
14:48
09-08-2020


Service Details - IBM Cloud
IBM Watson IoT Platform
Node-RED : node-red-smhic-2
Node-RED Dashboard
MIT App Inventor
Not secure
ai2.appinventor.mit.edu/#6023552092733440
Apps
Kuzey Guney
Neha
VIT
Smart INTERNZ
MIT APP INVENTOR
Health_Monitor
Palette
Viewer
Components
Properties
Screen1
HEALTH MONITOR
Water Level
Temperature
Light On
Light Off
Type here to search
ENG
IN
14:48
09-08-2020


Service Details - IBM Cloud
IBM Watson IoT Platform
Node-RED : node-red-smhic-2
Node-RED Dashboard
MIT App Inventor
Not secure | ai2.appinventor.mit.edu/#6023552092733440
Apps
Kuzey Guney
Neha
VIT
Smart INTERNZ
MIT APP INVENTOR
My Projects
Connect
Build
Settings
Help
View Trash
Gallery
Guide
Report an Issue
English
Health_Monitor
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Viewer
Built-in
Control
Logic
Math
Text
Lists
Dictionaries
Colors
Variables
Procedures
Screen1
Label3
HorizontalScrollArran
Label2
TextBox2
HorizontalScrollArran
Label1
Rename
Delete
Media
Upload File ...
when Web1 .GotText
responseCode
responseType
responseContent
set TextBox1 . Text to look up in pairs key Temperature
pairs call Web1 .JsonTextDecode
jsonText get responseContent
notFound not found
set TextBox2 . Text to look up in pairs key Water_Level
pairs call Web1 .JsonTextDecode
jsonText get responseContent
notFound not found
when Clock1 .Timer
set Web1 . Url to https://node-red-smhic-2020-08-06.eu-gb.mybluemi...
call Web1 .Get
when Button1 .Click
set Button1 . BackgroundColor to
set Button2 . BackgroundColor to
set Web2 . Url to https://node-red-smhic-2020-08-06.eu-gb.mybluemi...
call Web2 .Get
when Button2 .Click
set Button1 . BackgroundColor to
set Button2 . BackgroundColor to
set Web2 . Url to https://node-red-smhic-2020-08-06.eu-gb.mybluemi...
call Web2 .Get
Show Warnings
Type here to search
ENG
IN
14:48
09-08-2020