**ID: PNT2022TMID18845**

SPRINT-2

NOTIFICATION and STORE DATA

#include<WiFi.h>//library for wifi #include<PubSubClient.h>//library for MQTT

void callback(char\* subscribe topic, byte\* payload, unsigned int payload length);

//---------------credentials of IBM Account------

#define ORG "frpi8s"// IBM ORGANIZATION ID

#define DEVICE\_TYPE "NodeMCU"//DEVICE TYPE MENTIONED IN IOT WATSON

PLATFORM #define DEVICE\_ID "12345"//DEVICE ID MENTIONED IN IOT WATSONPLATEFORM

#define TOKEN "12345678"//Token String data3;float dist;

//------------customize the above value char server [] =ORG

".messaging.internetofthings.ibmcloud.com";//servername

char publish topic[]="ultrasonic/evt/Data/fmt/json";/\*topic name andtype

of event perform and format in which data to be send\*/

charsubscribetopic[]="ultrasonic/cmd/test/fmt/String";/\*cmd REPRESENT

Command tupe and

COMMAND IS TEST OF FORMAT STRING\*/

char authMethod[]="use-token-auth";//authentication method char token[]=TOKEN;

char clientid[]="d:" ORG ":" DEVICE\_TYPE":" DEVICE\_ID;//CLIENT ID

//

WiFiClient wifiClient;// creating an instance for wificlient PubSubClient client(server, 1883 , callback , wifiClient);/\*calling the predefined

client id by passing parameter like server id,portand wificredential\*/ int LED =4;

int trig =5; int echo=18; void setup(){

**Serial**.begin(115200); pinMode(trig,OUTPUT); pinMode(echo,INPUT); pinMode(LED,OUTPUT); delay(10); **Serial**.println(); wificonnect(); mqttconnect();

void loop() { digitalWrite(trig,LOW); digitalWrite(trig,HIGH); delayMicroseconds(10); digitalWrite(trig,LOW);

float dur=pulseIn(echo,HIGH); float dist=(dur \* 0.0343)/2;

**Serial**.print("distance in

cm"); **Serial**.println(dist); PublishData(dist); delay(1000);

if (!client.loop()){ mqttconnect();

}

}

/\* retriving to

cloud. ...................................................

\*/

void PublishData(float dist){ mqttconnect();//function call for connecting to ibm

/\*creating the string in form of JSON to update the data to ibm cloud\*/ String

object;if(dist<100)

{

digitalWrite(LED,HIGH); **Serial**.println("no object is near"); object="Near";

}

else

{

digitalWrite(LED,LOW); **Serial**.println("no object found"); object="No";

}

String payload="{\"distance\":"; payload +=dist; payload +="," "\"object\":\""; payload += object; payload += "\"}";

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

**Serial**.println(payload);

if(client.publish(publishtopic, (char\*) payload.c\_str())){ **Serial**.println("Publish ok");/\* if its sucessfully upload data on the cloud then it will print publish ok in serial monitor or else it will print publish

failed\*/

} else{

**Serial**.println("Publish failed");

}

}

void mqttconnect(){ if(!client.connected()){ **Serial**.print("Reconnecting client to "); **Serial**.println(server); while(!!!client.connect(clientid,authMethod,

token)){ **Serial**.print("."); delay(500);

}

initManagedDevice();

**Serial**.println();

}

}

void wificonnect()//function defenition for wificonnect

{

**Serial**.println(); **Serial**.print("Connecting to ");WiFi.begin("vivo 1816", "taetae95",6);//PASSING THE WIFI CREDIDENTIALS TO

ESTABLISH CONNECTION

while (WiFi.status() !=WL\_CONNECTED){ delay(500);

**Serial**.print(".");

}

**Serial**.println(""); **Serial**.println("WiFi connected"); **Serial**.println("IP address");

**Serial**.println(WiFi.localIP());

}

void initManagedDevice(){ if(client.subscribe(subscribetopic)){

**Serial**.println((subscribetopic)); **Serial**.println("subscribe to cmd OK");

}else{

**Serial**.println("subscribe to cmd failed");

}

}

void callback(char\* subscribetopic,byte\*payload,unsigned int payloadLength)

{

**Serial**.print("callback invoked for topic: ");

**Serial**.println(subscribetopic); for(int i=0; i< payloadLength; i++){

//Serial.print((char)payload[i]); data3 +=(char)payload[i];

}

//Serial.println("dta: "+ data3);

//if(data3=="Near")

//{

//Serial.println(data3);

//digitalWrite(LED,HIGH);

//}

//else //{

//Serial.println(data3);

//digitalWrite(LED,LOW);//} data3="";

}

CIRCUIT DIAGRAM:

