

CODING:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
#include <LiquidCrystal_I2C.h>
#include "DHT.h"// Library for dht11
#define DHTPIN 15      // what pin we're connected to
#define DHTTYPE DHT11  // define type of sensor DHT 11
#define LED 2
DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and type of
dht connected void callback(char* subscribetopic, byte* payload, unsigned int
payloadLength);

//-----credentials of IBM Accounts-----

#define ORG "64yf7x"//IBM ORGANIZATION ID
#define DEVICE_TYPE "b11m3edevicetype"//Device type mentioned in ibm watson
IOT Platform
#define DEVICE_ID "b11m3edeviceid"//Device ID mentioned in ibm watson IOT
Platform
#define TOKEN "-&EMtr7l-v-Gz2G))e"      //Token
String data3=""; int buzz= 13;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of
event perform and format in which data to be send char subscribetopic[] =
"iot-2/cmd/command/fmt/String";// cmd REPRESENT command type 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
LiquidCrystal_I2C lcd(0x27,16,2);

//-----
WiFiClient wifiClient; // creating the instance for wifiClient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand
wificredential
void setup()// configuring the
ESP32
{

    Serial.begin(115200);
    dht.begin();    pinMode(buzz,
OUTPUT);
    pinMode(LED,OUTPUT);
```

```

delay(10);
Serial.println();
wificonnect();
mqttconnect();
} void loop()// Recursive
Function
{ if (!client.loop())
{ mqttconnect();
}
}
}

```

```

/*.....retrieving to
Cloud.....*/
void PublishData(float temp, float humid) {
mqttconnect();//function call for connecting to ibm
} 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 defination for
wificonnect
{
Serial.println();
Serial.print("Connecting to ");

WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
the 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("Medicine Name: "+ data3);
    if(data3 != "")
    {
        lcd.init();
        lcd.print(data3);
        digitalWrite(LED,HIGH);
        tone(buzz, 100, 1000);
        delay(2000);
        digitalWrite(LED,LOW);
        noTone(buzz);
        delay(1000);

    }
    else {
        digitalWrite(LED,LOW);
    }
    data3="";
}

```

Output:

WOKWI SAVE SHARE Medicine Reminder Docs

WOKWI PROJECT: 348198638815543891

diagram.json libraries.txt Library Manager

```

1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #include <LiquidCrystal_I2C.h>
4 #include "DHT.h" // Library for dht11
5 #define DHTPIN 15 // what pin we're connected to
6 #define DHTTYPE DHT11 // define type of sensor DHT 11
7 #define LED 2
8 DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type of dht
9 void callback(char* topic, byte* payload, unsigned int payloadLength);
10
11 //-----credentials of IBM Accounts-----
12
13 #define ORG "64yf7x" //IBM ORGANIZATION ID
14 #define DEVICE_TYPE "b11m3edevicetype" //Device type mentioned in ibm watson IOT
15 #define DEVICE_ID "b11m3edeviceid" //Device ID mentioned in ibm watson IOT Platform
16 #define TOKEN "-8EMtr71-v-Gz2G)" //Token
17 String data3="";
18 int buzz= 13;
19
20 //----- Customise the above values -----
21 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
22 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event
23 char subscribeTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command
24 char authMethod[] = "use-token-auth"; // authentication method
25 char token[] = TOKEN;
26 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
27 LiquidCrystal_I2C lcd(0x27,16,2);
28
29 //-----
30 WiFiClient wifiClient; // creating the instance for wifiClient
31 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefined
32
33 void setup() // configuring the ESP32
34 {
35

```

Simulation

00:21.421 89%

Medicine Name: acetaminophen
callback invoked for topic: iot-2/cmd/command/fmt/String
Medicine Name: acetaminophen
callback invoked for topic: iot-2/cmd/command/fmt/String
Medicine Name: acetaminophen
callback invoked for topic: iot-2/cmd/command/fmt/String
Medicine Name: acetaminophen