

Source Code:

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQTT
#include "SoundData.h"
#include "XT_DAC_Audio.h"
XT_Wav_Class Sound("voice_command.wav");
XT_DAC_Audio_Class DacAudio(2,0);
uint32_t DemoCounter=0;
void callback(char* subscribtopic, byte* payload, unsigned int payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "ut4tn5"//IBM ORGANITION ID
#define DEVICE_TYPE "Arduino"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "nitish123"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "123456789" //Token
String data3;
float h, t;
//----- 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 subscribtopic[] = "iot-2/cmd/test/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
//-----
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()// configureing the ESP32
```

```

{
Serial.begin(115200);
delay(10);
Serial.println();
wificonnect();
mqttconnect();
}

void loop()// Recursive Function
{
delay(1000);
if (!client.loop()) {
mqttconnect();
}
}

/*.....retrieving to Cloud.....*/

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("data: " + data3);
if(data3=="announce")

```

```
{  
Serial.println(data3);  
for(int i=0;i<5;i++){  
DacAudio.FillBuffer();  
if(Sound.Playing==false)  
DacAudio.Play(&Sound);  
Serial.println(DemoCounter++);  
}  
}  
else  
{  
pass;  
}  
data3="";  
}
```