AIR QUALITY MONITORING PHASE -04

DEVELOPMENT PART-II

Connecting the Hardware in Wokwi

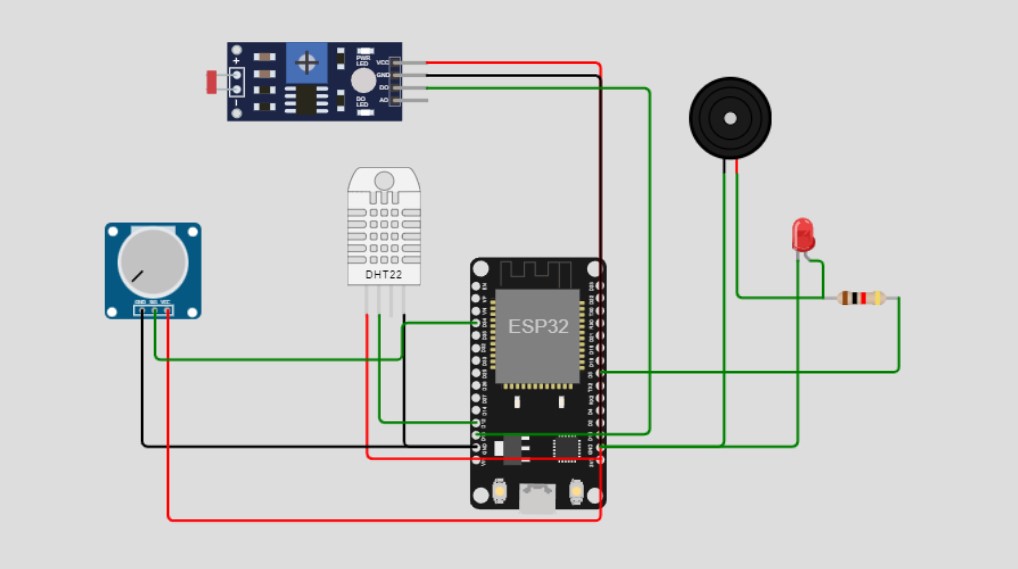
1. In the Wokwi simulator, you can add components like the ESP32, DHT22,Potentiometer ,Photoresistor Sensor,Resistor, led light and buzzer by dragging them from the components panel onto the virtual breadboard.

2. Connect the components using virtual jumper wires. Connect the power and ground pins appropriately.

3. Connect the Photoresistor Sensor,Potentiometer and DHT22 to an analog input pin on the ESP32.

4. Connect the buzzer and led through the resistor to a digital output pin on the ESP32.

HARDWARE CONNECTION



SOURCE CODE:

#include <WiFi.h>

#include <ThingSpeak.h>

#include "DHTesp.h"

#define ledPin 5

#define DO\_PIN 13

char ssid[] = "Wokwi-GUEST";

char pass[] = "";

WiFiClient client;

unsigned long myChannelNumber =  2311245;

const char \* myWriteAPIKey = "EUME5SWFX3BNSUWD";

DHTesp dhtSensor;

TempAndHumidity data;

const int DHT\_PIN = 12;

int statusCode;

void setup() {

**Serial**.begin(9600);

pinMode(32, INPUT);

pinMode(34, INPUT);

pinMode(ledPin, OUTPUT);

WiFi.mode(WIFI\_STA);

ThingSpeak.begin(client);

dhtSensor.setup(DHT\_PIN, DHTesp::DHT22);

}

void loop() {

  pinMode(DO\_PIN, INPUT);

connectToCloud();

aqi();

computeData();

writeData();

}

void connectToCloud(){

if(WiFi.status() != WL\_CONNECTED) {

**Serial**.print("Attempting to connect");

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

WiFi.begin(ssid, pass);

for(int i=0;i<5;i++) {

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

delay(5000);

}

}

**Serial**.println("\nConnected.");

}

}

void aqi(){

  int aqi=analogRead(34);

**Serial**.println("AQI INDEX=" + String(aqi));

  if (aqi<100){

**Serial**.println("Air Quality : Good");

  }

  else{

**Serial**.println("Air Quality : UnHealthy");

  }

  if(aqi<100){

    digitalWrite(ledPin, LOW);

    delay(1000);

    }

  else{

    digitalWrite(ledPin, HIGH);

    delay(5000);

  }

}

void computeData(){

data = dhtSensor.getTempAndHumidity();

**Serial**.println("Humi: " + String(data.humidity));

**Serial**.println("Temp: " + String(data.temperature));

int lightState = digitalRead(DO\_PIN);

  if (lightState == HIGH)

**Serial**.println("It is dark");

  else

**Serial**.println("It is light");

**Serial**.println("\-----------");

}

void writeData(){

int lightState = digitalRead(DO\_PIN);

int aqi=analogRead(34);

ThingSpeak.setField(3, data.humidity);

ThingSpeak.setField(2, data.temperature);

ThingSpeak.setField(4, lightState);

ThingSpeak.setField(5, aqi);

statusCode = ThingSpeak.writeFields(myChannelNumber,myWriteAPIKey);

if(statusCode == 200) //successful writing code

**Serial**.println("Channel update successful.");

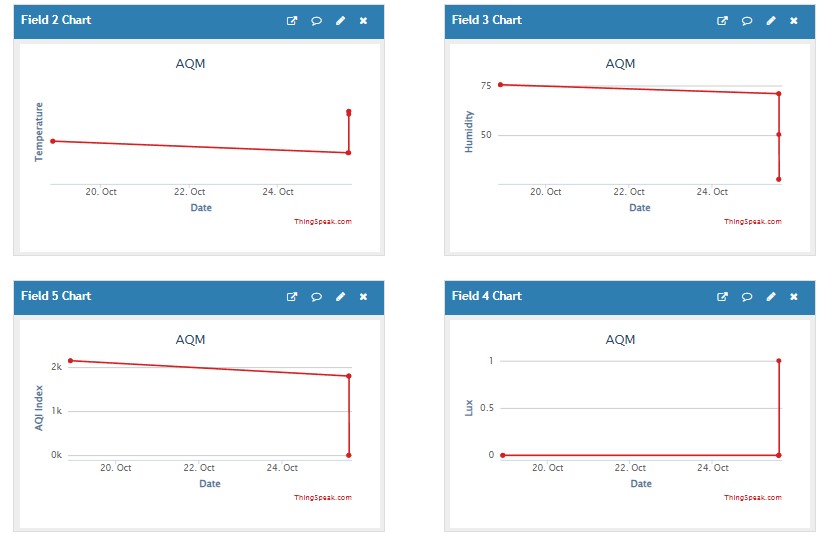
else

**Serial**.println("Problem Writing data. HTTP error code :" +

String(statusCode));

delay(5000); // data to be uploaded every 15secs

}



DEVELOP APK USING MIT APP INVENTOR

For every simulation in the wokwi platform the data can be update into personal channel created in the Thingspeak. We can use the data to know the difference level daily update and also live stream the data into the API interface using MIT APP INVENTOR

Using MIT app inventor we have to create app that can be named as AQM(Air Quality Monitoring)

AQM can be used to monitor and regular update from the cloud system

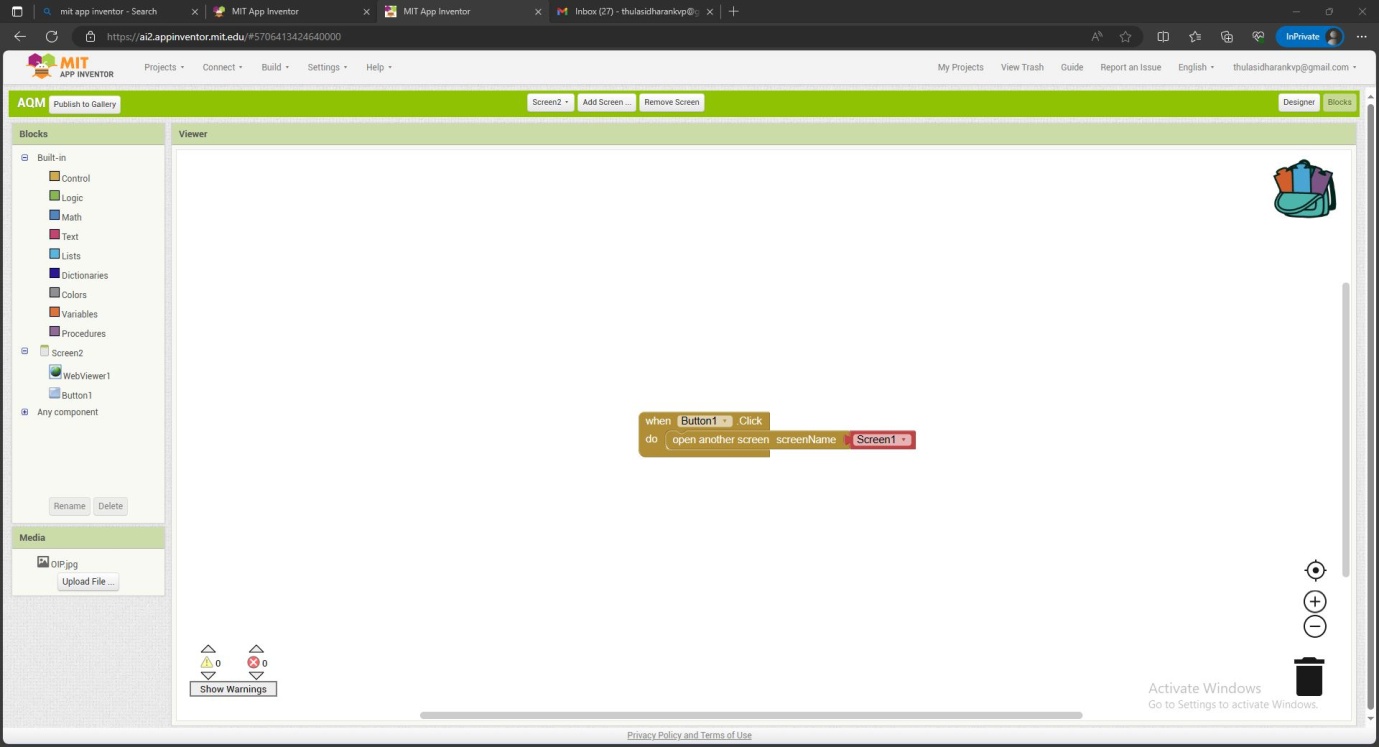
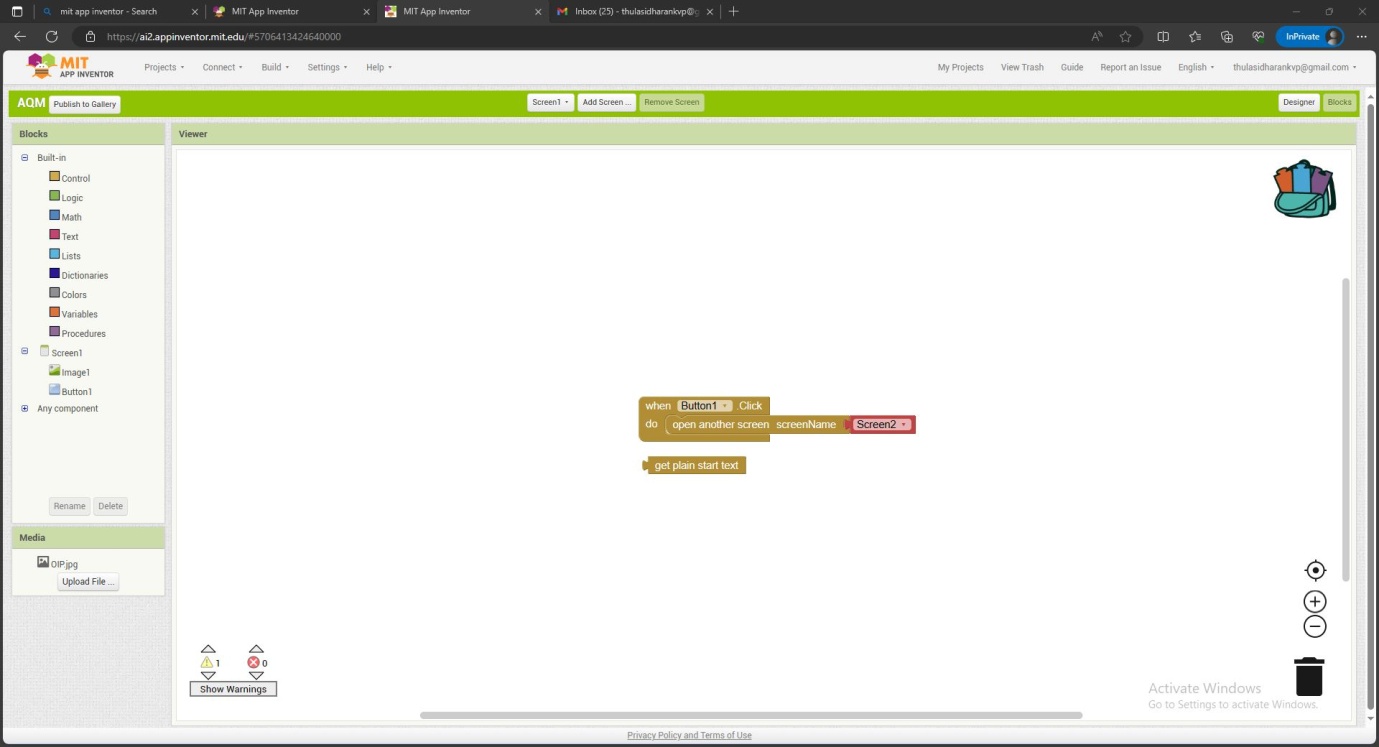
AQM contains two screen

SCREEN 1

It is the open Desktop for the AQM

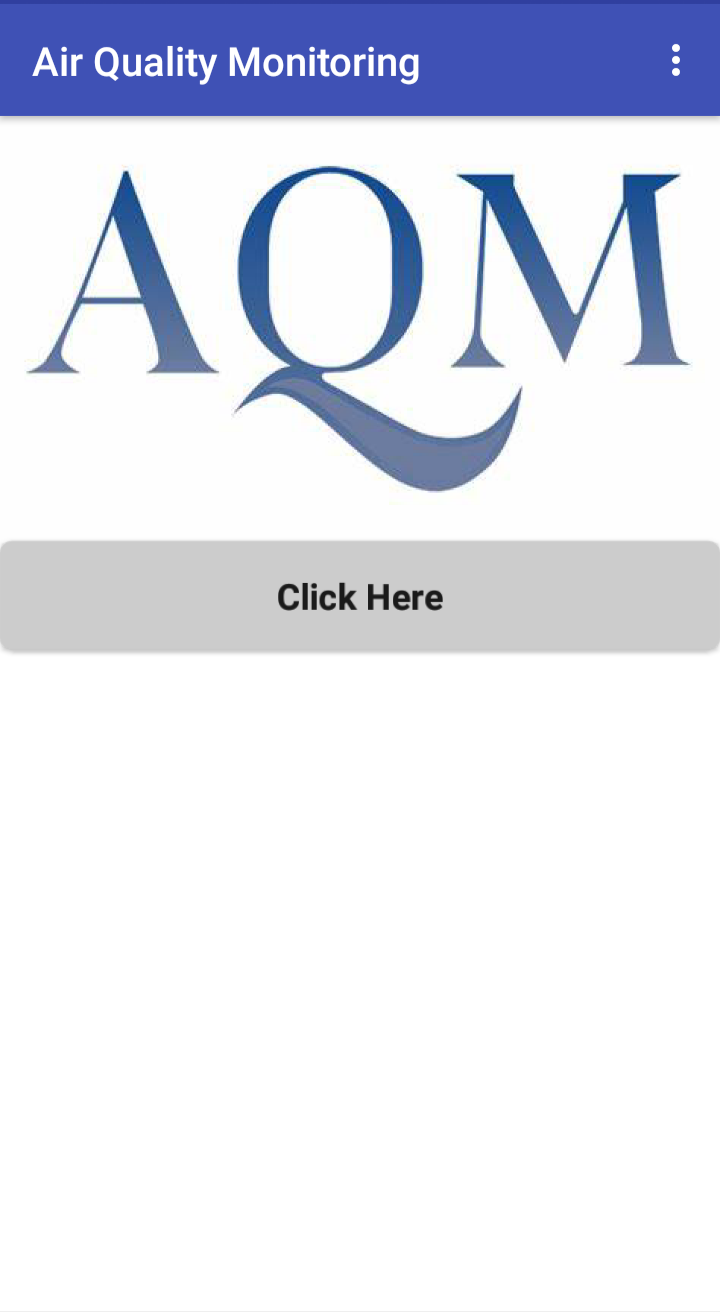
DATA ANALYST

It is used to collect the data from the cloud and provide alert message to the device



INSTALL THE APK MODE INTO THE MOBILE

1. USER INTERFACE



2.DATA ARRIVED FROM THINGSPEAK CONNECTED THROUGH THE MIT APP INVENTOR .

