FLOOD MONITORING PHASE -04 DEVELOPMENT PART-II

Connecting the Hardware in Wokwi

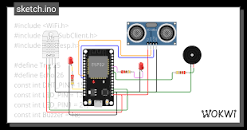
1. In the Wokwi simulator, you can add components like the ESP32, Ultrasonic sensor ( Hcsr -04), DHT22 , led light and buzzer by dragging them from the components panel onto the virtual breadboard.

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

3. Connect the ultrasonic sensor and DHT22 to an analog input pin on the ESP32.

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

HARDWARE CONNECTION



Source code

#include <ThingSpeak.h>

#include <WiFi.h>

#include "DHT.h"

#define DHTPIN  15                      //here we are initialising a pin for DHT22

#define DHTTYPE DHT22

#define ledPin 14

#define CM\_TO\_INCH 0.393701

DHT dht(DHTPIN, DHTTYPE);

const int trigPin = 23;

const int echoPin = 22;

int statusCode;

// defines variables

long duration;

float distance;

float distanceInch;

float Humidity;

float Temperature;

// wifi

const char \*ssid =  "Wokwi-GUEST"; //your network SSID (name)

const char \*pass =  ""; //your network password

WiFiClient client;

//thingspeak settings

unsigned long mychannelNumber = 2308333; //your channel ID number\*\* dari channel thingspeak yg telah kita buat

const char \*myWriteAPIkey = "FMXC5QW3BFQ8TI6L"; //your channel write API Key

//int lum, i = 0;

void setup() {

    pinMode(ledPin, OUTPUT);

    pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

    pinMode(echoPin, INPUT); // Sets the echoPin as an Input

**Serial**.begin(115200);

    delay(10);

    // Initialize the DHT sensor

  dht.begin();

    //connect to WiFi

**Serial**.print("Connecting to: "); **Serial**.println(ssid);

    WiFi.begin(ssid, pass);

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

      delay(500);

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

    }

**Serial**.println("\nWiFi connected\n");

    ThingSpeak.begin(client); //initialize ThingSpeak

}

void loop() {

  float T = dht.readTemperature();

  float H = dht.readHumidity();

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance cm

distance= duration\*0.034/2;

distanceInch = distance \* CM\_TO\_INCH;

 if(distance<20){

    digitalWrite(ledPin, HIGH);

    delay(1000);

    digitalWrite(ledPin, LOW);

    delay(100);

  }

  else{

    digitalWrite(ledPin, LOW);

  }

// Prints the distance on the Serial Monitor

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

**Serial**.print(H);

**Serial**.print(("%  Temperature: "));

**Serial**.print(T);

**Serial**.print("Water level (Cm): ");

**Serial**.println(distance);

// i++;

 // lum = analogRead(34);

       if(distance<20){

**Serial**.println("WARNING!");

**Serial**.println("waterlevel(cm): " + String(distance));

  }

  else{

**Serial**.println("SAFE");

  }

  //tahan selama 1 detik, program tidak menjalankan yang lain

  delay(2000);

writeData();

}

void writeData()

{

  float T = dht.readTemperature();

  float H = dht.readHumidity();

  ThingSpeak.setField(1, distance);

  ThingSpeak.setField(2, Temperature);

  ThingSpeak.setField(3,Humidity);

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 :" +

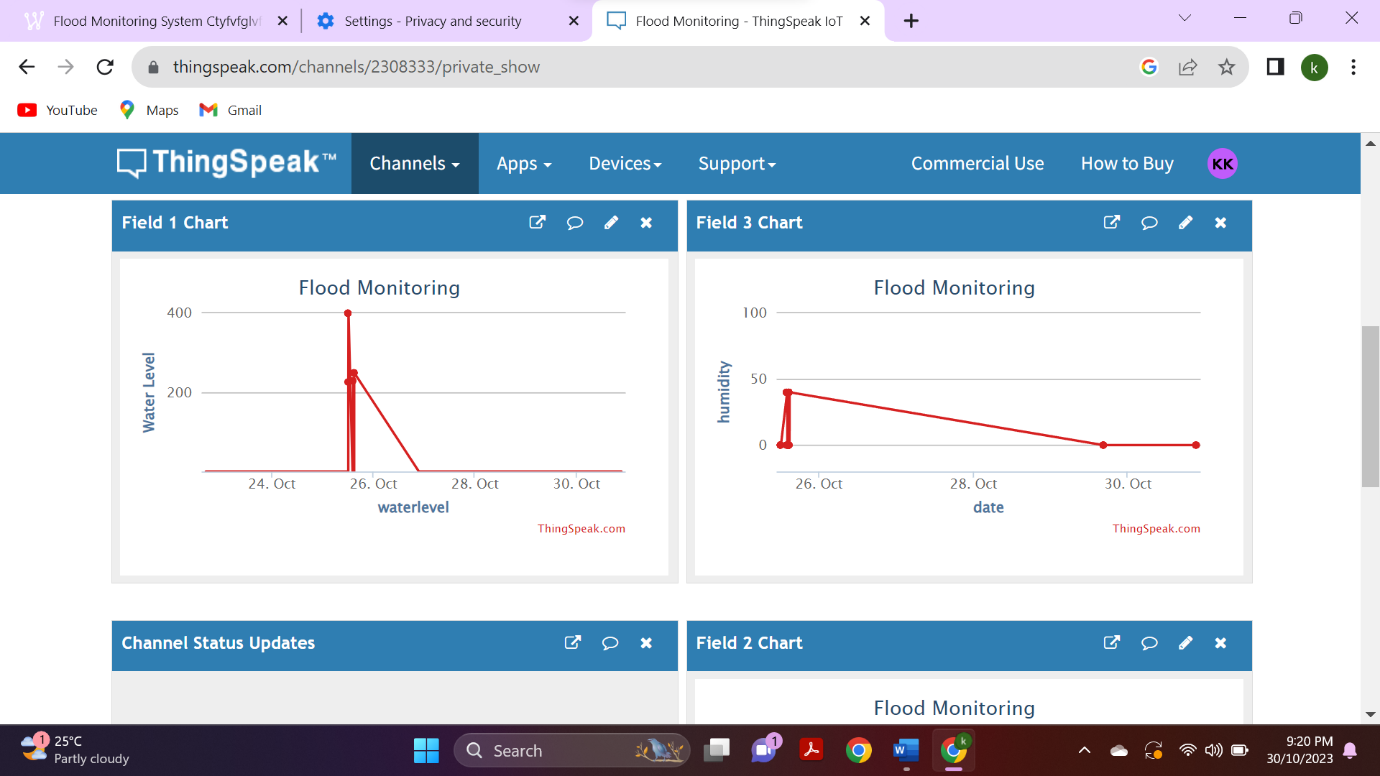
String(statusCode));

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

}

CONNECTING TO THE CLOUD

In wokwi platform simulation happen then it sends data into the ThingSpeak . It is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualization of live data, and send alerts.



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 FAMS(Flood Monitoring ALERT SYSTEM)

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

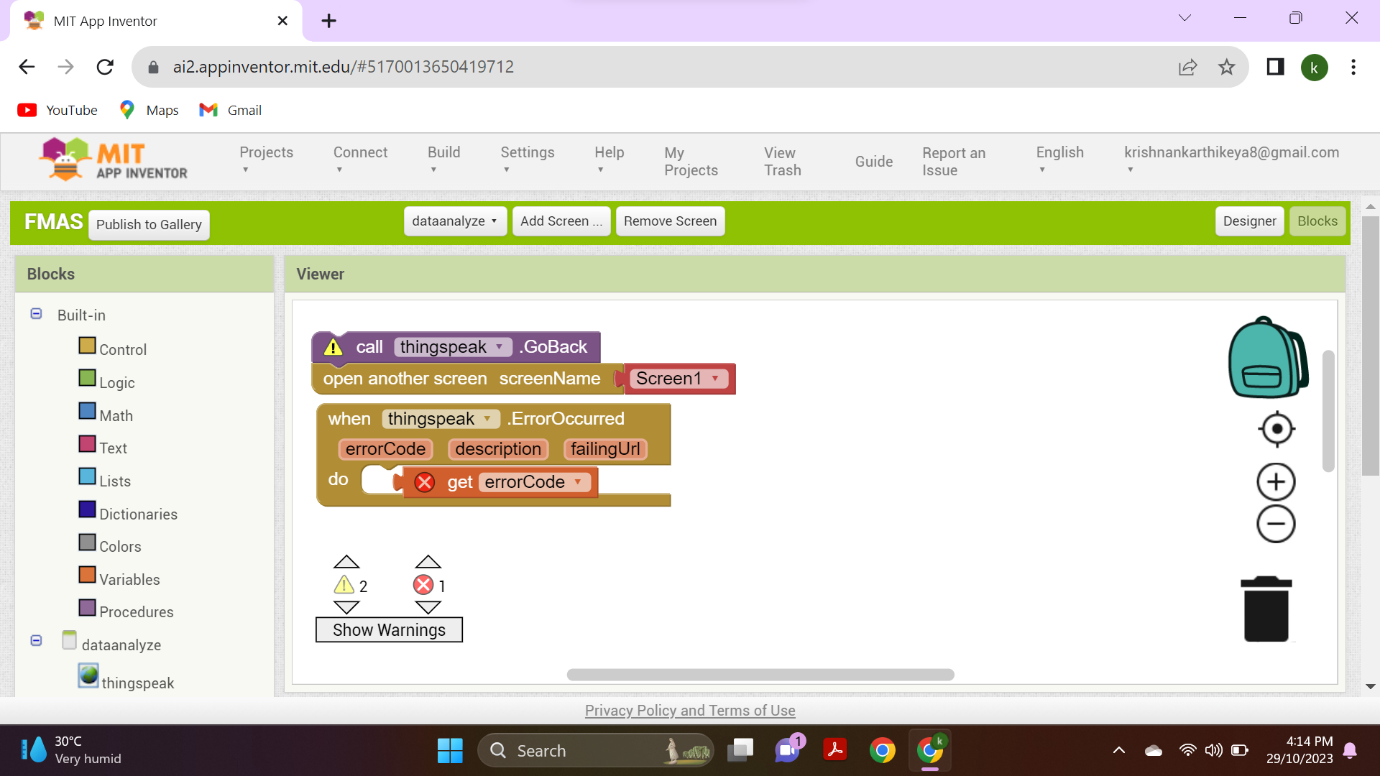
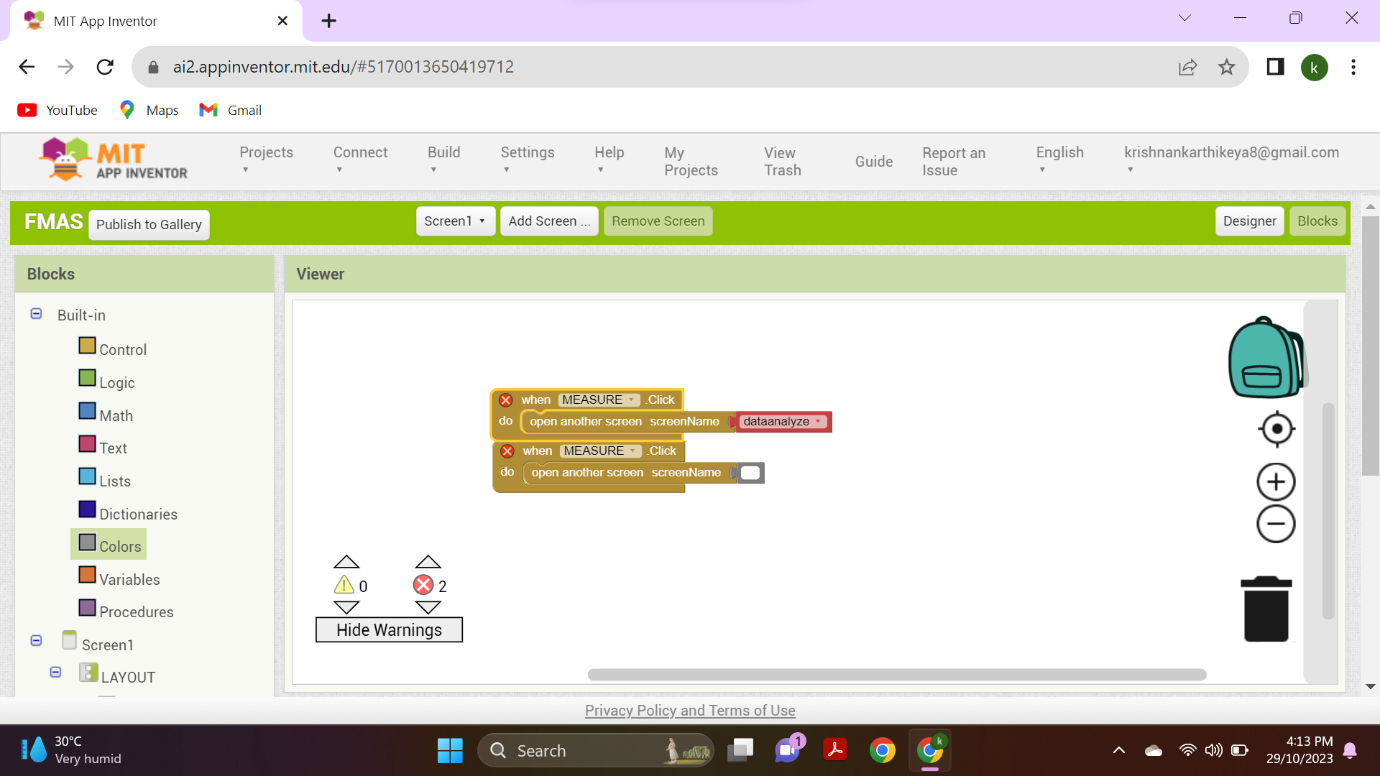
FAMS contains two screen

SCREEN 1

It is the open Desktop for the FAMS

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 .

