**Flood Monitoring and Early Warning System**

**Wokwi Project**: Downloaded from https://wokwi.com/projects/380203525580196865

Simulate this project on <https://wokwi.com>

**Sketch.ino:** //Early Flood Detection Using IOT ~ A project by Sabyasachi Ghosh

//<LiquidCrystal.h> is the library for using the LCD 16x2

#include <LiquidCrystal.h>

#include <WiFi.h>

#include <ThingSpeak.h>

//"DHT.h" is the library for using the Temperature sensor DHT22

#include "DHT.h"

char ssid[] = "Wokwi-GUEST";

char pass[] = "";

WiFiClient client;

unsigned long myChannelNumber =2326372;

const char \* myWriteAPIKey = "4Y93X5I7NC0K6SIO";

#define DHTPIN A0                       //here we are initialising a pin for DHT22

#define DHTTYPE DHT22                   //We have to declare the type of DHT sensor we are using for its correct functionality

LiquidCrystal lcd(2,3,4,5,6,7);         // Create an instance of the LiquidCrystal library

DHT dht(DHTPIN, DHTTYPE);               // Create an instance of the DHT library for the DHT22 sensor

const int in=8;                         //This is the ECHO pin of The Ultrasonic sensor HC-SR04

const int out=9;                        //This is the TRIG pin of the ultrasonic Sensor HC-SR04

// Define pin numbers for various components

const int green=10;

const int orange=11;

const int red=12;

const int buzz=13;

int statusCode;

void setup()

{

  // Start serial communication with a baud rate of 9600

  Serial.begin(9600);

  //WiFi.mode(WiFi.h);

  ThingSpeak.begin(client);

  // Initialize the LCD with 16 columns and 2 rows

  lcd.begin(16, 2);

  // Set pin modes for various components

  pinMode(in, INPUT);

  pinMode(out, OUTPUT);

  pinMode(green, OUTPUT);

  pinMode(orange, OUTPUT);

  pinMode(red, OUTPUT);

  pinMode(buzz, OUTPUT);

  // Initialize the DHT sensor

  dht.begin();

  // Set initial states for LEDs and buzzer to LOW (off)

  digitalWrite(green,LOW);

  digitalWrite(orange,LOW);

  digitalWrite(red,LOW);

  digitalWrite(buzz,LOW);

  // Display a startup message on the LCD

  lcd.setCursor(0, 0);

  lcd.print("Flood Monitoring");

  lcd.setCursor(0,1);

  lcd.print("Alerting System");

  // Wait for 5 seconds and then clear the LCD

  delay(5000);

  lcd.clear();

}

void loop() {

connectToCloud();

computeData();

writeData();

delay(1000);

}

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(1000);

}

}

Serial.println("\nConnected.");

}

}

void computeData(){

distance = getDistanceInCentimeters();

data = dhtSensor.getTempAndHumidity();

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

Serial.println("Humi: " + (data.H));

Serial.println("Temp: " + (data.T));

Serial.println("Dist: " + String(per));

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

}

void writeData(){

ThingSpeak.setField(1, data.H);

ThingSpeak.setField(2, data.T);

ThingSpeak.setField(3, per);

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(15000); // data to be uploaded every 15secs

}

void loop()

{

  // Read temperature and humidity from the DHT22 sensor

  float T = dht.readTemperature();

  float H = dht.readHumidity();

  // Check if the sensor data is valid

  if (isnan(H) && isnan(T)) {

    lcd.print("ERROR");

    return;

  }

  float f = dht.readTemperature(true);

  // Read distance from the ultrasonic sensor (HC-SR04)

  long dur;

  long dist;

  long per;

  digitalWrite(out,LOW);

  delayMicroseconds(2);

  digitalWrite(out,HIGH);

  delayMicroseconds(10);

  digitalWrite(out,LOW);

  dur=pulseIn(in,HIGH);

  dist=(dur\*0.034)/2;

  // Map the distance value to a percentage value

  per=map(dist,10.5,2,0,100);

  // Ensure that the percentage value is within bounds

  if(per<0)

  {

    per=0;

  }

  if(per>100)

  {

    per=100;

  }

  // Print sensor data and percentage value to serial

  Serial.print(("Humidity: "));

  Serial.print(H);

  Serial.print(("%  Temperature: "));

  Serial.print(T);

  Serial.print("%   Water Level:");

  Serial.println(String(per));

  lcd.setCursor(0,0);

  lcd.print("Temperature:");

  lcd.setCursor(0,1);

  lcd.print("Humidity   :");

  lcd.setCursor(12,0);

  lcd.print(T);

  lcd.setCursor(12,1);

  lcd.print(H);

  delay(1000);

  lcd.clear();

  lcd.print("Water Level:");

  lcd.print(String(per));

  lcd.print("%  ");

  // Check water level and set alert levels

  if(dist<=3)

  {

      lcd.setCursor(0,1);

      lcd.print("Red Alert!   ");

      digitalWrite(red,HIGH);

      digitalWrite(green,LOW);

      digitalWrite(orange,LOW);

      digitalWrite(buzz,HIGH);

      delay(2000);

      digitalWrite(buzz,LOW);

      delay(2000);

      digitalWrite(buzz,HIGH);

      delay(2000);

      digitalWrite(buzz,LOW);

      delay(2000);

  }

  else if(dist<=10)

  {

      lcd.setCursor(0,1);

    lcd.print("Orange Alert!  ");

    digitalWrite(orange,HIGH);

    digitalWrite(red,LOW);

    digitalWrite(green,LOW);

    digitalWrite(buzz,HIGH);

    delay(3000);

    digitalWrite(buzz,LOW);

    delay(3000);

  }else

  {

     lcd.setCursor(0,1);

    lcd.print("Green Alert!  ");

    digitalWrite(green,HIGH);

    digitalWrite(orange,LOW);

    digitalWrite(red,LOW);

    digitalWrite(buzz,LOW);

  }

}

**Diagram.json:**

{

  "version": 1,

  "author": "S Isabel",

  "editor": "wokwi",

  "parts": [

    {

      "type": "wokwi-arduino-uno",

      "id": "uno",

      "top": -64.64,

      "left": 105.33,

      "rotate": 90,

      "attrs": {}

    },

    { "type": "wokwi-dht22", "id": "dht1", "top": -305.53, "left": 410.32, "attrs": {} },

    { "type": "wokwi-lcd1602", "id": "lcd1", "top": -312.02, "left": 508, "attrs": {} },

    {

      "type": "wokwi-led",

      "id": "led1",

      "top": -45.08,

      "left": 439.42,

      "attrs": { "color": "red" }

    },

    {

      "type": "wokwi-hc-sr04",

      "id": "ultrasonic1",

      "top": -56.1,

      "left": 677.5,

      "attrs": { "distance": "7" }

    },

    {

      "type": "wokwi-slide-potentiometer",

      "id": "pot1",

      "top": -339.66,

      "left": 767.48,

      "rotate": 270,

      "attrs": { "travelLength": "30" }

    },

    {

      "type": "wokwi-buzzer",

      "id": "bz1",

      "top": -83.14,

      "left": 364.46,

      "attrs": { "volume": "0.1" }

    },

    {

      "type": "wokwi-led",

      "id": "led2",

      "top": -44.56,

      "left": 483.32,

      "attrs": { "color": "orange" }

    },

    {

      "type": "wokwi-led",

      "id": "led3",

      "top": -45.47,

      "left": 527.48,

      "attrs": { "color": "limegreen" }

    }

  ],

  "connections": [

    [ "lcd1:D4", "uno:4", "magenta", [ "v0" ] ],

    [ "lcd1:D5", "uno:5", "magenta", [ "v0" ] ],

    [ "lcd1:D6", "uno:6", "magenta", [ "v0" ] ],

    [ "lcd1:D7", "uno:7", "magenta", [ "v0" ] ],

    [ "led3:A", "uno:10", "red", [ "v0" ] ],

    [ "led2:A", "uno:11", "orange", [ "v0" ] ],

    [ "led1:A", "uno:12", "green", [ "v0" ] ],

    [ "bz1:2", "uno:13", "gray", [ "v0" ] ],

    [ "uno:GND.1", "led1:C", "black", [ "h0" ] ],

    [ "uno:GND.1", "led2:C", "black", [ "h0" ] ],

    [ "uno:GND.1", "led3:C", "black", [ "h0" ] ],

    [ "uno:GND.1", "bz1:1", "black", [ "h0" ] ],

    [ "uno:GND.2", "lcd1:VSS", "black", [ "h-27.73", "v-236.01", "h405.87" ] ],

    [ "uno:GND.2", "lcd1:RW", "black", [ "h-37.21", "v-223.22", "h460.88" ] ],

    [ "uno:5V", "lcd1:VDD", "red", [ "h-18.3", "v-203.63", "h13.46" ] ],

    [ "lcd1:RS", "uno:2", "magenta", [ "v0" ] ],

    [ "lcd1:E", "uno:3", "magenta", [ "v0" ] ],

    [ "uno:5V", "lcd1:A", "red", [ "h-46.76", "v-186.9", "h566.43" ] ],

    [ "uno:GND.2", "lcd1:K", "black", [ "h-55.24", "v-183", "h584.41" ] ],

    [ "uno:GND.3", "ultrasonic1:GND", "black", [ "h-26.54", "v101.57", "h653.32" ] ],

    [ "uno:5V", "ultrasonic1:VCC", "red", [ "h-37.14", "v129.59", "h633.92" ] ],

    [ "ultrasonic1:TRIG", "uno:9", "cyan", [ "v0" ] ],

    [ "ultrasonic1:ECHO", "uno:8", "cyan", [ "v0" ] ],

    [ "uno:A0", "dht1:SDA", "white", [ "h-64.26", "v-286.12", "h344.25" ] ],

    [ "uno:5V", "dht1:VCC", "red", [ "h-72.21", "v-246.3", "h342.7" ] ],

    [ "uno:GND.3", "dht1:GND", "black", [ "h-81.23", "v-213.9", "h380.52" ] ],

    [ "lcd1:V0", "pot1:SIG", "yellow", [ "v39.74", "h14.45" ] ],

    [ "uno:5V", "pot1:VCC", "red", [ "h-56.08", "v146.93", "h757.73" ] ],

    [ "uno:GND.3", "pot1:GND", "black", [ "h-46.52", "v120.3", "h800.57", "v-607.79", "h-52.4" ] ]

  ],

  "dependencies": {}

}

**Libraries:** # Wokwi Library List

# See https://docs.wokwi.com/guides/libraries

LiquidCrystal

DHT22

DHT sensor library

wifi

thingspeak

A computer screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a graph

Description automatically generated