#define BLYNK\_TEMPLATE\_ID "TMPL3cwgULoW1"

#define BLYNK\_TEMPLATE\_NAME "ESP32 Monitoring"

#define BLYNK\_AUTH\_TOKEN "ZvP9EGRJ4SYMyc-V40tLC2PDB3L2X5wp"

char ssid[] = "abi";

char pass[] = "abinaya@2003";

emptyTankDistance = 10 ; //Distance when tank is empty

int fullTankDistance = 2 ; //Distance when tank is full

int triggerPer = 20 ; //alarm will start when water level drop below triggerPer

#include <Adafruit\_SSD1306.h>

#include <WiFi.h>

#include <WiFiClient.h>

#include <BlynkSimpleEsp32.h>

#include <AceButton.h>

using namespace ace\_button;

#define TRIGPIN 27 //D27

#define ECHOPIN 26 //D26

#define wifiLed 2 //D2

6

#define ButtonPin1 12 //D12

#define BuzzerPin 13 //D13

#define GreenLed 14 //D14

#define VPIN\_BUTTON\_1 V1

#define VPIN\_BUTTON\_2 V2

#define SCREEN\_WIDTH 128 // OLED display width, in pixels

#define SCREEN\_HEIGHT 32 // OLED display height, in pixels

#define OLED\_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire,

OLED\_RESET);

float duration;

float distance;

int waterLevelPer;

bool toggleBuzzer = HIGH; //Define to remember the toggle state

char auth[] = BLYNK\_AUTH\_TOKEN;

ButtonConfig config1;

AceButton button1(&config1);

void handleEvent1(AceButton\*, uint8\_t, uint8\_t);

BlynkTimer timer;

void checkBlynkStatus()

{

7

bool isconnected = Blynk.connected();

if (isconnected == false) {

//Serial.println("Blynk Not Connected");

digitalWrite(wifiLed, LOW);

}

if (isconnected == true) {

digitalWrite(wifiLed, HIGH);

//Serial.println("Blynk Connected"); } }

BLYNK\_CONNECTED() {

Blynk.syncVirtual(VPIN\_BUTTON\_1);

Blynk.syncVirtual(VPIN\_BUTTON\_2);

}

void displayData(int value){

display.clearDisplay();

display.setTextSize(4);

display.setCursor(8,2);

display.print(value);

display.print(" ");

display.print("%");

display.display(); }

8

void measureDistance(){

digitalWrite(TRIGPIN, LOW);

delayMicroseconds(2);

digitalWrite(TRIGPIN, HIGH);

delayMicroseconds(20);

digitalWrite(TRIGPIN, LOW);

duration = pulseIn(ECHOPIN, HIGH);

distance = ((duration / 2) \* 0.343)/10;

if (distance > (fullTankDistance) && distance < emptyTankDistance ){

waterLevelPer = map((int)distance ,emptyTankDistance, fullTankDistance, 0, 100);

displayData(waterLevelPer);

Blynk.virtualWrite(VPIN\_BUTTON\_1, waterLevelPer);

Blynk.virtualWrite(VPIN\_BUTTON\_2, (String(distance) + " cm"));

// Print result to serial monitor

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");

if (waterLevelPer < triggerPer){

digitalWrite(GreenLed, HIGH);

9

if (toggleBuzzer == HIGH){

digitalWrite(BuzzerPin, HIGH); } }

if (distance < fullTankDistance){

digitalWrite(GreenLed, LOW);

if (toggleBuzzer == HIGH){

digitalWrite(BuzzerPin, HIGH); } }

if (distance > (fullTankDistance + 5) && waterLevelPer > (triggerPer + 5)){

toggleBuzzer = HIGH;

digitalWrite(BuzzerPin, LOW); } }

delay(100);

}

void setup() {

Serial.begin(115200)

pinMode(ECHOPIN, INPUT);

pinMode(TRIGPIN, OUTPUT);

pinMode(wifiLed, OUTPUT);

pinMode(GreenLed, OUTPUT);

pinMode(BuzzerPin, OUTPUT);

pinMode(ButtonPin1, INPUT\_PULLUP);

digitalWrite(wifiLed, LOW);

10

digitalWrite(GreenLed, LOW);

digitalWrite(BuzzerPin, LOW);

config1.setEventHandler(button1Handler);

button1.init(ButtonPin1);

if(!display.begin(SSD1306\_SWITCHCAPVCC, 0x3C)) {

Serial.println(F("SSD1306 allocation failed"));

for(;;);

}

delay(1000);

display.setTextSize(1);

display.setTextColor(WHITE);

display.clearDisplay();

WiFi.begin(ssid, pass);

timer.setInterval(2000L, checkBlynkStatus); // check if Blynk server is connected every 2

seconds

Blynk.config(auth);

delay(1000);

}

void loop() {

measureDistance();

Blynk.run();

11

timer.run(); // Initiates SimpleTimer

button1.check();

}

void button1Handler(AceButton\* button, uint8\_t eventType, uint8\_t buttonState) {

Serial.println("EVENT1");

switch (eventType) {

case AceButton::kEventReleased:

//Serial.println("kEventReleased");

digitalWrite(BuzzerPin, LOW);

toggleBuzzer = LOW;

break;

}

}