#include <Wire.h>

//#include <LiquidCrystal\_I2C.h>

#include <LCD\_I2C.h>

#include <WiFi.h>

#include <ThingSpeak.h>

#define WIFI\_SSID "dammy"

#define WIFI\_PASSWORD "uranus1234"

#include <PZEM004Tv30.h>

#define RXD2 16

#define TXD2 17

PZEM004Tv30 pzem(&Serial2);

LCD\_I2C lcd(0x27, 16, 2);

unsigned long channelID = 2075996;

const char \*apiKey = "QOU1C2F4LDLJDY2A";

WiFiClient client;

uint8\_t happy[8] =

{

0b00000,

0b10001,

0b00000,

0b00000,

0b10001,

0b01110,

0b00000,

0b00000,

};

#define grid\_restored 5

void setup() {

Serial.begin(115200);

lcd.begin();

// WiFi.begin(ssid,pass);

lcd.backlight();

lcd.createChar(0, happy);

pinMode(grid\_restored ,OUTPUT);

digitalWrite(grid\_restored ,HIGH);

delay(2000);

digitalWrite(grid\_restored ,LOW);

delay(500);

pinMode(grid\_restored ,OUTPUT);

digitalWrite(grid\_restored ,HIGH);

delay(2000);

digitalWrite(grid\_restored ,LOW);

delay(500);

lcd.setCursor (0,0);

lcd.print(" POWER RESTORED ");

delay (1000);

lcd.clear();

// Uncomment in order to reset the internal energy counter

// pzem.resetEnergy()

lcd.setCursor(1,0);

lcd.print(" IOT ENERGY ");

lcd.setCursor(4,1);

lcd.print("METER");

delay(2000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("initializing...");

lcd.setCursor (1,7);

lcd.write(0);

lcd.write(0);

lcd.write(0);

delay(3000);

//ssss buzz();

lcd.clear();

// Connect to WiFi network

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

Serial.print("Connecting to WiFi...");

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

delay(1000);

Serial.print(".");

}

Serial.println("Connected!");

// Initialize ThingSpeak library with channel ID and API key

ThingSpeak.begin(client);

}

void loop() {

check();

}

void check (){

Serial.print("Custom Address:");

// Serial.println(pzem.readAddress(), HEX);

// Read the data from the sensor

float voltage = pzem.voltage();

float current = pzem.current();

float power = pzem.power();

float energy = pzem.energy();

float frequency = pzem.frequency();

float pf = pzem.pf();

// Check if the data is valid

if(isnan(voltage)){

Serial.println("Error reading voltage");

} else if (isnan(current)) {

Serial.println("Error reading current");

} else if (isnan(power)) {

Serial.println("Error reading power");

} else if (isnan(energy)) {

Serial.println("Error reading energy");

} else if (isnan(frequency)) {

Serial.println("Error reading frequency");

} else if (isnan(pf)) {

Serial.println("Error reading power factor");

} else {

// Print the values to the Serial console

Serial.print("Voltage: "); Serial.print(voltage, 2); Serial.println("V");

Serial.print("Current: "); Serial.print(current); Serial.println("A");

Serial.print("Power: "); Serial.print(power); Serial.println("W");

Serial.print("Energy: "); Serial.print(energy,3); Serial.println("kWh");

Serial.print("Frequency: "); Serial.print(frequency, 1); Serial.println("Hz");

Serial.print("PF: "); Serial.println(pf);

float cost = energy \*50.8;

lcd.setCursor(0,0);

lcd.print ("V I E");

lcd.setCursor(0,1);

lcd.print( voltage,0);

lcd.setCursor(5,1);

lcd.print( current,2);

lcd.setCursor(11,1);

lcd.print(energy);

ThingSpeak.setField(1, voltage);

ThingSpeak.setField(2, current);

ThingSpeak.setField(3, power);

ThingSpeak.setField(4, energy);

ThingSpeak.setField(5, cost);

int status = ThingSpeak.writeFields(channelID, apiKey);

if (status == 200) {

Serial.println("Data sent to ThingSpeak!");

} else {

Serial.println("Failed to send data to ThingSpeak...");

}

}

Serial.println();

//delay(2000);

}