```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>

#include <SoftwareSerial.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <ArduinoJson.h>

#define SCREEN_WIDTH 128    // OLED display width, in pixels
#define SCREEN_HEIGHT 64    // 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);
#define IN1POMPA D3 // D3 disambungkan ke IN1 Relay untuk mengatur pompa

#define RE D5
#define DE D6


const char* ssid = "keluarga";
const char* password = "helmanpolisi";

//const char* host = "tanicerdas.com";  //IP Address Server
String server = "http://192.168.1.18/destryweb/public";
String sendVal, getRelay, setPump;

unsigned long lastTime = 0;
unsigned long timerDelay = 5000;
```

```
const byte nitro[] = {0x01, 0x03, 0x00, 0x1e, 0x00, 0x01, 0xe4, 0x0c};
const byte phos[] = {0x01, 0x03, 0x00, 0x1f, 0x00, 0x01, 0xb5, 0xcc};
const byte pota[] = {0x01, 0x03, 0x00, 0x20, 0x00, 0x01, 0x85, 0xc0};

String apiKeyValue = "tPmAT5Ab3j7F9";
byte values[11];
SoftwareSerial mod(D7, D8);
int nilaiSoil, count = 0;

void setup() {
 //Set URL
 sendVal = server + "/nodemcu/update";
 getRelay = server + "/nodemcu/relay";
 setPump = server + "/nodemcu/pump/";

 // coding untuk relay pompa
 pinMode(IN1POMPA, OUTPUT);
 Serial.begin(9600);
 mod.begin(9600);
 pinMode(RE, OUTPUT);
 pinMode(DE, OUTPUT);


 display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //initialize with the I2C addr 0x3C (128x64)
 delay(500);
 display.clearDisplay();
 display.setCursor(25, 15);
 display.setTextSize(1);
```

```
 display.setTextColor(WHITE);
 display.println(" NPK Sensor");
 display.setCursor(25, 35);
 display.setTextSize(1);
 display.print("Initializing");
 display.display();
 delay(3000);

 Serial.begin(9600);

 WiFi.begin(ssid, password);
 Serial.println("Connecting");
 while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
 }
 Serial.println("");
 Serial.print("Connected to WiFi network with IP Address: ");
 Serial.println(WiFi.localIP());

  Serial.println("Timer set to 5 seconds (timerDelay variable), it will take 5 seconds before publishing the first reading.");
}

void loop() {

  // coding untuk sensor soil moisture
  nilaiSoil = map(analogRead(A0), 1023, 0, 0, 100);// untuk membaca data dari sensor soil moisture
  Serial.println(nilaiSoil);
```

```
display.setTextSize(2); // untuk tampilan sensor pada oled
display.setCursor(0, 5);
display.print("Soil Moisture: ");
display.print(nilaiSoil);
display.setTextSize(1);
display.print(" %");
display.clearDisplay();
delay(250);

byte val1, val2, val3;
val1 = nitrogen();
delay(250);
val2 = phosphorous();
delay(250);
val3 = potassium();
delay(250);



Serial.print("Nitrogen: ");
Serial.print(val1);
Serial.println(" mg/kg");
Serial.print("Phosphorous: ");
Serial.print(val2);
Serial.println(" mg/kg");
Serial.print("Potassium: ");
Serial.print(val3);
Serial.println(" mg/kg");
```

```
if (count == 60) {
  sendData((String)val1, (String)val2, (String)val3);
  count = 0;
} else {
  count++;
}
Serial.print("Counter : ");
Serial.println(count);
cekMode();

display.clearDisplay();



display.setTextSize(2);
display.setCursor(0, 5);
display.print("N: ");
display.print(val1);
display.setTextSize(1);
display.print(" mg/kg");

display.setTextSize(2);
display.setCursor(0, 25);
display.print("P: ");
display.print(val2);
display.setTextSize(1);
display.print(" mg/kg");

display.setTextSize(2);
display.setCursor(0, 45);
```

```
  display.print("K: ");
  display.print(val3);
  display.setTextSize(1);
  display.print(" mg/kg");

  display.display();

}

void sendData(String N, String P, String K) {
  WiFiClient client;
  HTTPClient http;
  http.begin(client, sendVal.c_str());
  http.addHeader("Content-Type", "application/x-www-form-urlencoded");
  String httpRequestData = (String)"api_key=" + apiKeyValue + "&nilai_N=" + N
                           + "&nilai_P=" + P + "&nilai_K=" + K + "&nilai_kelembaban=" + nilaiSoil;
  Serial.print("httpRequestData: ");
  Serial.println(httpRequestData);
  int httpResponseCode = http.POST(httpRequestData);
  String payload = http.getString();
  Serial.println(payload);
  Serial.print("HTTP Response code: ");
  Serial.println(httpResponseCode);
  http.end();
}

void cekMode() {
  WiFiClient client;
  HTTPClient http;
```

```
http.begin(client, getRelay.c_str());
//http.addHeader("Content-Type", "application/x-www-form-urlencoded");
//String httpRequestData = (String)"api_key=" + apiKeyValue + "&nilai_N="+ N
//+"&nilai_P="+ P +"&nilai_K="+ K +"&nilai_kelembaban="+ nilaiSoil;
//Serial.print("httpRequestData: ");
//Serial.println(httpRequestData);
int httpResponseCode = http.GET();
String payload = http.getString();
Serial.println(payload);
Serial.print("HTTP Response code: ");
Serial.println(httpResponseCode);
http.end();
if (httpResponseCode == 200) {
  DynamicJsonDocument doc(1024);
  deserializeJson(doc, payload);
  JsonObject obj = doc.as<JsonObject>();
  String modePompa = obj["relay"].as<String>();
  String setPompa = obj["nilai"].as<String>();
  if (modePompa == "manual") {
    bool val = (setPompa == "0") ? true : false;
    digitalWrite(IN1POMPA, val);
  } else {
    nilaiSoil = map(analogRead(A0), 1023, 0, 0, 100);// untuk membaca data dari sensor soil moisture
    Serial.println(nilaiSoil);
    if (nilaiSoil<60)digitalWrite(IN1POMPA, LOW);
    else if (nilaiSoil>80)digitalWrite(IN1POMPA, HIGH);
    WiFiClient client;
    HTTPClient http;
    String getVAL = digitalRead(IN1POMPA) ? "0" : "1";
```

```
    String pompa = setPump + getVAL;
    http.begin(client, pompa.c_str());
    int httpResponseCode = http.GET();
    Serial.print("HTTP Response code Set Pompa: ");
    Serial.println(httpResponseCode);
    http.end();
   }
  }
}

byte nitrogen() {
  digitalWrite(DE, HIGH);
  digitalWrite(RE, HIGH);
  delay(10);
  if (mod.write(nitro, sizeof(nitro)) == 8) {
    digitalWrite(DE, LOW);
    digitalWrite(RE, LOW);
    for (byte i = 0; i < 7; i++) {
      //Serial.print(mod.read(),HEX);
      values[i] = mod.read();
      //Serial.print(values[i],HEX);
    }
    //Serial.println();
  }
  return values[4];
}

byte phosphorous() {
  digitalWrite(DE, HIGH);
```

```
  digitalWrite(RE, HIGH);
  delay(10);
  if (mod.write(phos, sizeof(phos)) == 8) {
    digitalWrite(DE, LOW);
    digitalWrite(RE, LOW);
    for (byte i = 0; i < 7; i++) {
      //Serial.print(mod.read(),HEX);
      values[i] = mod.read();
      // Serial.print(values[i],HEX);
    }
    //Serial.println();
  }
  return values[4];
}

byte potassium() {
  digitalWrite(DE, HIGH);
  digitalWrite(RE, HIGH);
  delay(10);
  if (mod.write(pota, sizeof(pota)) == 8) {
    digitalWrite(DE, LOW);
    digitalWrite(RE, LOW);
    for (byte i = 0; i < 7; i++) {
      //Serial.print(mod.read(),HEX);
      values[i] = mod.read();
      //Serial.print(values[i],HEX);
    }
    //Serial.println();
  }
```

```
  return values[4];


}
```