

LAMPIRAN

Lampiran 1. Daftar Nama Responden Mahasiswa

Daftar nama mahasiswa sebagai kelompok kecil

No	Nama Mahasiswa	NIM	Kode
1	Nyoman Darma Wijaya	2115061009	R1
2	Made Wida Kurniawan	2115061003	R2
3	Putu Andi Surya Shaputra	2115061005	R3
4	Kadek Arya Prasmanta Diva	2115061006	R4
5	Komang Aditya Pramana Putra	2115061024	R5

Daftar nama mahasiswa sebagai kelompok besar

No.	Nama	NIM	Kode
1	Komang Aditya Pramana P.	2115061024	R1
2	Kadek Arya Prasmanta Diva	2115061006	R2
3	Nyoman Darma Wijaya	2115061009	R3
4	Jesika Putri	2215061010	R4
5	Tiara Elok Riskyta	2215061008	R5
6	Komang Endra Adiyana	2315061007	R6
7	Dilon Diliano Patopang	2315061010	R7
8	I Gede Nyoman Bagiada	2315061001	R8
9	Muhammad Devan Maulana Akbar	2315061009	R9
10	Gede Dita Suadnyana	2315061016	R10
11	Made Subawa Ada	2315061011	R11
12	Komang Wahyu Udayana	2315061005	R12
13	Gede Nanda Putra Dinatha	2315061008	R13
14	Gede Suta	2315061002	R14
15	Rizki Wahyudi	2315061013	R15
16	Elfarezhel Agung Dwisantra	2315061004	R16
17	I Komang Widi Adi Putra	2315061003	R17
18	Wayan M.Dhiya Ul'Khoir	2315061015	R18
19	Moh. Fenska Pramanta Ryanto	2315061012	R19
20	I Putu Aditya Sunu Pradnyana	2315061014	R20

Lampiran 2. Kode Program

Kode Program *Node Sensor Transmitter*

```

include "LoRa_E220.h"
#include <SoftwareSerial.h>
#include <DHT.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define MQ2PIN A0
#define FLAMEPIN 6

#define DESTINATION_ADDH 0xFF //65535
#define DESTINATION_ADDL 0xFF //65535
#define CHANNEL 23

SoftwareSerial mySerial(2, 3);
LoRa_E220 e220ttl(&mySerial);
DHT dht(DHTPIN, DHTTYPE);

unsigned long previousMillis = 0;
const long intervalPengiriman = 500;

void setup() {
  Serial.begin(9600);
  delay(500);
  e220ttl.begin();
  dht.begin();
  pinMode(FLAMEPIN, INPUT);
}

void loop() {
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= intervalPengiriman) { //1000
    - 500 = interval pengiriman
    previousMillis = currentMillis;

    float suhu = dht.readTemperature();
    float kelembapan = dht.readHumidity();

    int asap = analogRead(MQ2PIN);

    int flame = digitalRead(FLAMEPIN);
    int flameDetected = (flame == 0 ? 1 : 0);

    String status = statusKebakaran(suhu, asap, flame);
  }
}

```

```
// Untuk mengganti identitas node cukup mengganti identitas node_1/2/3
// di bawah
String data = "Node_2," + String(suhu) + "," + String(kelembapan)
+ "," + String(asap) + "," + String(flameDetected) + "," + status +
"\n";

Serial.println("Mengirim: " + data);
ResponseStatus rs = e220ttl.sendFixedMessage(DESTINATION_ADDH,
DESTINATION_ADDL, CHANNEL, data);
Serial.print("Status Pengiriman: ");
Serial.println(rs.getResponseDescription());
}
}

String statusKebakaran(float suhu, int asap, int flame) {
  if (flame == 0 && suhu > 30 && asap > 60) return "BAHAYA";
  if (flame == 0 && suhu > 31) return "BAHAYA";
  if (suhu > 29 && asap > 100) return "BAHAYA";
  if ((flame == 0) && (suhu > 29 && suhu <= 30) && (asap > 50 && asap
<=60)) return "WASPADA";
  if ((flame == 0) && (suhu > 30 && suhu <= 31)) return "WASPADA";
  if ((suhu > 28) && (asap > 50 && asap <=100)) return "WASPADA";
  return "AMAN";
}
```

Kode program node receiver gateway

```
/* KODE PADA TEMPLATE BLYNK */
#define BLYNK_TEMPLATE_ID "TMPL6FlzOW_tT"
#define BLYNK_TEMPLATE_NAME "Fire Detector Berbasis Wireless Sensor
Network"

/* PENAMBAHAN LIBRARY */
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SoftwareSerial.h>
#include <LoRa_E220.h>
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

/* TOKEN AUTENTIKASI, WIFI DAN PASSWORD */
char auth[] = "4-a11SFY9jiiJ5S85XiboatyEvJSv4w1";
char ssid[] = "fire detector";
char pass[] = "WSN12345";

/* DEKLARASI PIN DAN TIPE LCD */
LiquidCrystal_I2C lcd(0x27, 16, 2);
SoftwareSerial LoRa(D5, D6);
const int relay = D4;
```

```

/* VARIABEL SENSOR */
float suhu1, suhu2, suhu3;
float asap1, asap2, asap3;
float hum1, hum2, hum3;
int api1, api2, api3;
String status1, status2, status3;

/* TIMING PROGRAM */
unsigned long previousMillis1 = 0; //TIMING PEMBACAAN PENERIMAAN DATA
DARI LORA
unsigned long previousMillis2 = 0; //TIMING PERGANTIAN TAMPILAN DI
LCD
const long interval1 = 500; //INTERVAL PEMBACAAN PENERIMAAN
DATA DARI LORA
const long interval2 = 3000; //INTERVAL PERGANTIAN TAMPILAN DI
LCD
int currentNode = 1; //OPERASI UNTUK PERGANTIAN NODE
SECARA BERURUTAN

/* TIMEOUT NODE */
unsigned long waktuTerimaTerakhir1 = 0; //WAKTU PENERIMAAN DATA
NODE SENSOR 1
unsigned long waktuTerimaTerakhir2 = 0; //WAKTU PENERIMAAN DATA
NODE SENSOR 2
unsigned long waktuTerimaTerakhir3 = 0; //WAKTU PENERIMAAN DATA
NODE SENSOR 3
const unsigned long batasWaktuHabis = 15000; //DURASI WAKTU HABIS
KONEKSI LORA

/* RELAY */
bool statusBahaya = true;

/* KONDISI AWAL NODE SENSOR */
bool koneksi_node1 = false;
bool koneksi_node2 = false;
bool koneksi_node3 = false;

String receivedData = "";

void updateBlynk();

void setup() {
  Serial.begin(9600);
  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Node Receiver");
  lcd.setCursor(0, 1);
  lcd.print("Connect WiFi...");

```



```

/* Koneksi Awal WiFi */
WiFi.begin(ssid, pass);
int wifi_attempts = 0;
while (WiFi.status() != WL_CONNECTED && wifi_attempts < 20) {
    delay(500);
    wifi_attempts++;
}
lcd.clear();
if (WiFi.status() == WL_CONNECTED) {
    lcd.print("WiFi Terhubung");
    Blynk.config(auth);
    Blynk.connect();
} else {
    lcd.print("WiFi Terputus");
}
delay(1500);
lcd.clear();

LoRa.begin(9600);
Serial.println("LoRa initialized");

pinMode(relay, OUTPUT);
digitalWrite(relay, HIGH);
}

void loop() {
    if (WiFi.status() == WL_CONNECTED) Blynk.run();
    unsigned long currentMillis = millis();

    /** 1. Terima & Baca Data LoRa menggunakan interval1 */
    if (currentMillis - previousMillis1 >= interval1) {
        previousMillis1 = currentMillis;

        if (LoRa.available()) {
            receivedData = LoRa.readStringUntil('\n');
            receivedData.trim();

            int sIdx = receivedData.indexOf("Node_");
            if (sIdx > 0) receivedData = receivedData.substring(sIdx);

            int idx1 = receivedData.indexOf(',');
            int idx2 = receivedData.indexOf(',', idx1 + 1);
            int idx3 = receivedData.indexOf(',', idx2 + 1);
            int idx4 = receivedData.indexOf(',', idx3 + 1);
            int idx5 = receivedData.indexOf(',', idx4 + 1);

            if (idx1 > 0 && idx2 > idx1 && idx3 > idx2 && idx4 > idx3 &&
            idx5 > idx4) {
                String node = receivedData.substring(0, idx1);
                float suhu = receivedData.substring(idx1 + 1, idx2).toFloat();
            }
        }
    }
}

```

```

float hum = receivedData.substring(idx2 + 1, idx3).toFloat();
float asap = receivedData.substring(idx3 + 1, idx4).toFloat();
int api = receivedData.substring(idx4 + 1, idx5).toInt();
String status = receivedData.substring(idx5 + 1);

if (node == "Node_1") {
    suhu1 = suhu;
    hum1 = hum;
    asap1 = asap;
    api1 = api;
    status1 = status;
    waktuTerimaTerakhir1 = currentMillis;
    koneksi_node1 = true;
} else if (node == "Node_2") {
    suhu2 = suhu;
    hum2 = hum;
    asap2 = asap;
    api2 = api;
    status2 = status;
    waktuTerimaTerakhir2 = currentMillis;
    koneksi_node2 = true;
} else if (node == "Node_3") {
    suhu3 = suhu;
    hum3 = hum;
    asap3 = asap;
    api3 = api;
    status3 = status;
    waktuTerimaTerakhir3 = currentMillis;
    koneksi_node3 = true;
}
updateBlynk();
}
}
}

/** 2. Waktu habis koneksi LoRa seluruh node */
if (currentMillis - waktuTerimaTerakhir1 > batasWaktuHabis) {
    koneksi_node1 = false;
    suhu1 = 0;
    hum1 = 0;
    asap1 = 0;
    api1 = 0;
    status1 = "";
}
if (currentMillis - waktuTerimaTerakhir2 > batasWaktuHabis) {
    koneksi_node2 = false;
    suhu2 = 0;
    hum2 = 0;
    asap2 = 0;
    api2 = 0;
    status2 = "";
}
}

```

```

if (currentMillis - waktuTerimaTerakhir3 > batasWaktuHabis) {
  koneksi_node3 = false;
  suhu3 = 0;
  hum3 = 0;
  asap3 = 0;
  api3 = 0;
  status3 = "";
}

/** 3. Menampilkan data masing-masing Node Sensor di Serial Monitor
**/
Serial.println("===== DATA NODE SENSOR =====");
Serial.print("Node 1 | Suhu: ");
Serial.print(suhu1);
Serial.print("°C | Hum: ");
Serial.print(hum1);
Serial.print("% | Api: ");
Serial.print(api1);
Serial.print(" | Asap: ");
Serial.print(asap1);
Serial.print(" | Status: ");
Serial.println(status1);

Serial.print("Node 2 | Suhu: ");
Serial.print(suhu2);
Serial.print("°C | Hum: ");
Serial.print(hum2);
Serial.print("% | Api: ");
Serial.print(api2);
Serial.print(" | Asap: ");
Serial.print(asap2);
Serial.print(" | Status: ");
Serial.println(status2);

Serial.print("Node 3 | Suhu: ");
Serial.print(suhu3);
Serial.print("°C | Hum: ");
Serial.print(hum3);
Serial.print("% | Api: ");
Serial.print(api3);
Serial.print(" | Asap: ");
Serial.print(asap3);
Serial.print(" | Status: ");
Serial.println(status3);
Serial.println("=====\\n");

delay(2000);

/** 4. Update Tampilan data node sensor pada LCD dengan timing
interval2**/

```

```

if (currentMillis - previousMillis2 >= interval2) {
    previousMillis2 = currentMillis;
    lcd.clear();

    statusBahaya = (status1 == "BAHAYA" || status2 == "BAHAYA" ||
status3 == "BAHAYA");

    digitalWrite(relay, statusBahaya ? LOW : HIGH); //menggunakan
relay Active LOW (bernilai LOW untuk menghidupkan relay)

    /* Prioritas tampilan LCD 1: KETIKA NODE SENSOR MENUNJUKKAN STATUS
BAHAYA */
    if (statusBahaya) {
        if (status1 == "BAHAYA") {
            lcd.setCursor(0, 0);
            lcd.print("N1 BAHAYA |API:" + String(api1));
            lcd.setCursor(0, 1);
            lcd.print("T:" + String(suhu1) + " ASAP:" + String(asap1));
            Blynk.logEvent("node_sensor_1", String("Terdeteksi Kebakaran
di Node Sensor 1! Suhu: ") + String(suhu1) + " °C | Asap : " +
String(asap1) + " ppm | Api: " + String(api1));
        } else if (status2 == "BAHAYA") {
            lcd.setCursor(0, 0);
            lcd.print("N2 BAHAYA |API:" + String(api2));
            lcd.setCursor(0, 1);
            lcd.print("T:" + String(suhu2) + " ASAP:" + String(asap2));
            Blynk.logEvent("node_sensor_2", String("Terdeteksi Kebakaran
di Node Sensor 2! Suhu: ") + String(suhu2) + " °C | Asap : " +
String(asap2) + " ppm | Api: " + String(api2));
        } else {
            lcd.setCursor(0, 0);
            lcd.print("N3 BAHAYA |API:" + String(api3));
            lcd.setCursor(0, 1);
            lcd.print("T:" + String(suhu3) + " ASAP:" + String(asap3));
            Blynk.logEvent("node_sensor_3", String("Terdeteksi Kebakaran
di Node Sensor 3! Suhu: ") + String(suhu3) + " °C | Asap : " +
String(asap2) + " ppm | Api: " + String(api3));
        }
    }

    /* Prioritas tampilan LCD 2: KETIKA NODE SENSOR MENUNJUKKAN STATUS
WASPADA */
    else if (status1 == "WASPADA" || status2 == "WASPADA" || status3
== "WASPADA") {
        if (status1 == "WASPADA") {
            lcd.setCursor(0, 0);
            lcd.print("N1 WASPADA|API:" + String(api1));
            lcd.setCursor(0, 1);
            lcd.print("T:" + String(suhu1) + " ASAP:" + String(asap1));
        } else if (status2 == "WASPADA") {
            lcd.setCursor(0, 0);
            lcd.print("N2 WASPADA|API:" + String(api2));

```

```

        lcd.setCursor(0, 1);
        lcd.print("T:" + String(suhu2) + " ASAP:" + String(asap2));
    } else {
        lcd.setCursor(0, 0);
        lcd.print("N3 WASPADA|API:" + String(api3));
        lcd.setCursor(0, 1);
        lcd.print("T:" + String(suhu3) + " ASAP:" + String(asap3));
    }
}

/* Rotasi normal tampilan data Node Sensor dan keterangan koneksi
Node sensor */
else {
    switch (currentNode) {
        case 1:
            if (koneksi_node1) {
                lcd.setCursor(0, 0);
                lcd.print("Node 1 T:" + String(suhu1));
                lcd.setCursor(0, 1);
                lcd.print("API:" + String(api1) + " ASAP:" +
String(asap1));
            } else {
                lcd.setCursor(0, 0);
                lcd.print("Node 1 TERPUTUS ");
                lcd.setCursor(0, 1);
                lcd.print(" ");
            }
            break;

        case 2:
            if (koneksi_node2) {
                lcd.setCursor(0, 0);
                lcd.print("Node 2 T:" + String(suhu2));
                lcd.setCursor(0, 1);
                lcd.print("API:" + String(api2) + " ASAP:" +
String(asap2));
            } else {
                lcd.setCursor(0, 0);
                lcd.print("Node 2 TERPUTUS ");
                lcd.setCursor(0, 1);
                lcd.print(" ");
            }
            break;

        case 3:
            if (koneksi_node3) {
                lcd.setCursor(0, 0);
                lcd.print("Node 3 T:" + String(suhu3));
                lcd.setCursor(0, 1);
                lcd.print("API:" + String(api3) + " ASAP:" +
String(asap3));
            } else {

```

```

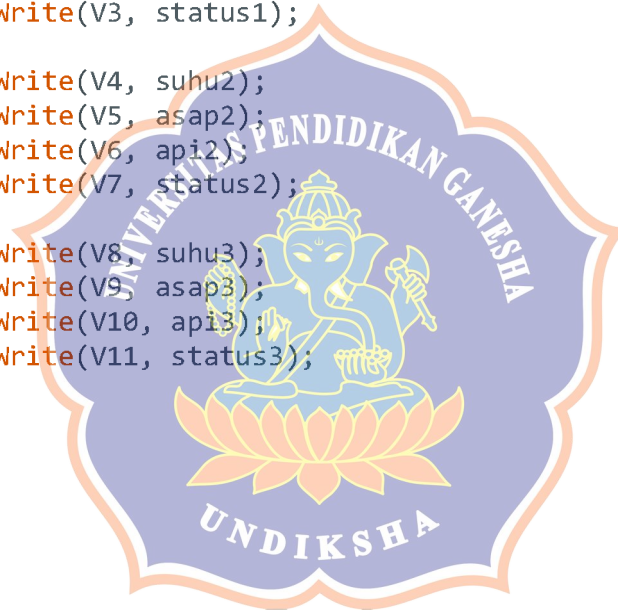
        lcd.setCursor(0, 0);
        lcd.print("Node 3 TERPUTUS ");
        lcd.setCursor(0, 1);
        lcd.print("                ");
    }
    break;
}
currentNode = (currentNode % 3) + 1;
}
}
}

/* ===== */
void updateBlynk() {
    Blynk.virtualWrite(V0, suhu1);
    Blynk.virtualWrite(V1, asap1);
    Blynk.virtualWrite(V2, api1);
    Blynk.virtualWrite(V3, status1);

    Blynk.virtualWrite(V4, suhu2);
    Blynk.virtualWrite(V5, asap2);
    Blynk.virtualWrite(V6, api2);
    Blynk.virtualWrite(V7, status2);

    Blynk.virtualWrite(V8, suhu3);
    Blynk.virtualWrite(V9, asap3);
    Blynk.virtualWrite(V10, api3);
    Blynk.virtualWrite(V11, status3);
}

```



Lampiran 3. Dokumentasi Proses Penelitian

FOTO	KETERANGAN
	Wawancara bersama dosen pengampu mata kuliah sensor dan aktuator
 	Bimbingan bersama dosen pembimbing 1



Bimbingan bersama
dosen pembimbing 2



	Proses pembuatan media pembelajaran
---	--

	Uji coba mandiri
	Uji coba validasi ahli media



Uji coba pemakaian
oleh kelompok kecil



Uji coba pemakaian
oleh kelompok besar