LAMPIRAN

Lampiran 1. Surat Kesediaan Membimbing TA Dosen Pembimbing I

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini:

Nama	: Arif Rakhman, SE, S.Pd, M.Kom
NIPY	: 05.016.291
Jabatan Struktural	: Dosen Tetap
Jabatan Fungsional	: Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut:

Nama	: Sadam Amsar
NIM	: 22040107
Program Studi	: Diploma III Teknik Komputer

Judul TA : RANCANG BANGUN SISTEM OTOMATISASI PEMANTAUAN KUALIATS AIR KOLAM IKAN KOI DENGAN PENGURASAN DAN PENGISIAN OTOMATIS BERBASIS ESP32

Dengan ini menyatakan ini dibuat agar dilaksanakan sebagaimana mestinya

Tegal, 17 Februari 2025

Mengetahui

Ka Prodi DIII Teknik Komputer,	Dosen Pembimbing I,
Ida Ariliana, ST, M.Kom	Arif Rakhman, SE, S.Pd,M.Kom
NIPY. 12.013.168	NIPY. 05.016.291

Lampiran 2. Surat Kesediaan Membimbing TA Dosen Pembimbing II

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini:

Nama	: Arfan Haqiqi Sulasmoro, M.Kom
NIPY	: 02.009.054
Jabatan Struktural	: -
Jabatan Fungsional	: Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing II pada Tugas Akhir mahasiswa berikut:

Nama	: Sadam Amsar
NIM	: 22040107
Program Studi	: Diploma III Teknik Komputer

Judul TA : RANCANG BANGUN SISTEM OTOMATISASI PEMANTAUAN KUALITAS AIR KOLAM IKAN KOI DENGAN PENGURASAN DAN PENGISIAN OTOMATIS BERBASIS ESP32

Dengan ini menyatakan ini dibuat agar dilaksanakan sebagaimana mestinya.

Tegal, 5 Mei 2025

Mengetahui

Ka Prodi DIII Teknik Komputer,

Ida Afridiana, ST, M.Kom

NIPY. 12.013.168

Dosen Pembimbing II,

Arfan Haqiqi Sulasmoro, M.Kom

NIPY. 02.009.054

Lampiran 3. *Source Code* Program

```
#include <ESPAsyncWebServer.h>
#include <AsyncTCP.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <ArduinoJson.h>

// LCD Setup
LiquidCrystal_I2C lcd(0x27, 16, 2);

// WiFi Configuration
const char* ssid = "JITOE";
const char* password = "takondepan";
const String serverUrl = "http://192.168.51.219:8080/api/sensor";

// Pin Definitions
#define PH_PIN        34
#define TDS_PIN       35
#define TURBIDITY_PIN 33
#define TRIG_PIN      5
#define ECHO_PIN      18
#define RELAY_POMPA   26
#define RELAY_VALVE   27

// Threshold Values
#define PH_MIN        5.0
#define PH_MAX        9.0
#define TDS_MAX       500.0
#define TURBIDITY_MAX 60.0
#define WATER_MIN     10.0
#define WATER_MAX     20.0

// Variables
float pH = 0, tds = 0, turbidity = 0, ketinggian_cm = 0;
unsigned long lastOperationMillis = 0;
const long operationInterval = 5000;
int lcdState = 0;

String modeKontrol = "auto";
bool statusPompa = false;
bool statusValve = false;

float mapFloat(float x, float in_min, float in_max, float out_min,
float out_max) {
  return (x - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min;
}

void setup() {
  Serial.begin(115200);
  lcd.init();
  lcd.backlight();
```

```
  pinMode(RELAY_POMPA, OUTPUT);
  pinMode(RELAY_VALVE, OUTPUT);
  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);

  digitalWrite(RELAY_POMPA, HIGH);
  digitalWrite(RELAY_VALVE, HIGH);

  lcdPrint("Sistem Koi", "Memulai...");

  connectWiFi();
}

void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
  }

  if (millis() - lastOperationMillis > operationInterval) {
    lastOperationMillis = millis();
    readAllSensors();
    sendDataToServer();
    printSensorData();
  }

  updateLCD();

  if (modeKontrol == "auto") {
    autoControl();
  } else {
    manualControl();
  }
}

void connectWiFi() {
  lcdPrint("Connecting", "WiFi...");
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("\nWiFi Connected");
  lcdPrint("WiFi", "Connected");
  delay(1000);
}

void readAllSensors() {
  readpH();
  readTDS();
  readTurbidity();
  readWaterLevel();
}
```

```
void readpH() {
  int sensorValue = analogRead(PH_PIN);
  float voltage = sensorValue * (3.3 / 4095.0);
  pH = 2.0 * voltage + 0.5; // Kalibrasi kasar, nanti bisa
adjustment saat tes
}

void readTDS() {
  int sensorValue = analogRead(TDS_PIN);
  float voltage = sensorValue * (3.3 / 4095.0);
  tds = (133.42 * voltage * voltage * voltage
        - 255.86 * voltage * voltage
        + 857.39 * voltage) * 0.5;
}

void readTurbidity() {
  float voltageSum = 0;
  int sampleCount = 10;

  for (int i = 0; i < sampleCount; i++) {
    int sensorValue = analogRead(TURBIDITY_PIN);
    float voltage = sensorValue * (3.3 / 4095.0);
    voltageSum += voltage;
    delay(50);
  }

  float averageVoltage = voltageSum / sampleCount;

  // Debug

  if (averageVoltage > 1.8) {
    turbidity = mapFloat(averageVoltage, 1.8, 3.3, 0, 9);
  } else if (averageVoltage > 1.7 && averageVoltage <= 1.8) {
    turbidity = mapFloat(averageVoltage, 1.7, 1.8, 10, 15);
  } else if (averageVoltage > 1.5 && averageVoltage <= 1.7) {
    turbidity = mapFloat(averageVoltage, 1.5, 1.7, 16, 25);
  } else if (averageVoltage > 1.0 && averageVoltage <= 1.5) {
    turbidity = mapFloat(averageVoltage, 1.0, 1.5, 26, 40);
  } else if (averageVoltage >= 0.5 && averageVoltage <= 1.0) {
    turbidity = mapFloat(averageVoltage, 0.5, 1.0, 41, 100);
  } else {
    turbidity = 100; // Di bawah 0.5 volt, sangat keruh sekali
  }
}

void readWaterLevel() {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  long duration = pulseIn(ECHO_PIN, HIGH);
  float distance = (duration * 0.0343) / 2.0;

  ketinggian_cm = constrain(30.0 - distance, 0, 30);
```

```
}

void sendDataToServer() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    if (!http.begin(serverUrl)) {
      Serial.println("HTTP Gagal Begin");
      return;
    }

    http.addHeader("Content-Type", "application/json");

    DynamicJsonDocument doc(256);
    doc["pH"] = pH;
    doc["tds"] = tds;
    doc["turbidity"] = turbidity;
    doc["ketinggian"] = ketinggian_cm;
    doc["statusPompa"] = digitalRead(RELAY_POMPA) == LOW ? 1 : 0;
    doc["statusValve"] = digitalRead(RELAY_VALVE) == LOW ? 1 : 0;

    String json;
    serializeJson(doc, json);

    int httpCode = http.POST(json);

    if (httpCode > 0) {
      String payload = http.getString();
      Serial.println("Respon Server: " + payload);
      DynamicJsonDocument resDoc(128);
      deserializeJson(resDoc, payload);
      if (resDoc.containsKey("mode")) {
        modeKontrol = resDoc["mode"].as<String>();
      }
    } else {
      Serial.println("HTTP          Error:          "          +
http.errorToString(httpCode));
    }

    http.end();
  }
}

void autoControl() {
  // Pompa menyala jika salah satu kondisi terpenuhi: pH tidak
normal, TDS tinggi, atau turbidity tinggi
  // Tetapi pompa harus mati jika ketinggian air terlalu rendah
  bool shouldPumpBeOn = ((pH < PH_MIN || pH > PH_MAX) ||
                         (tds > TDS_MAX) ||
                         (turbidity > TURBIDITY_MAX)) &&
                         (ketinggian_cm >= WATER_MIN);S
  // Selenoid valve menyala jika air kurang dari level minimum
  // Selenoid valve mati jika air sudah mencapai level maksimum
  bool shouldValveBeOn = (ketinggian_cm < WATER_MIN) ||
                         (ketinggian_cm    <    WATER_MAX    &&
digitalRead(RELAY_VALVE) == LOW);
```

```
  // LOW untuk relay menyala (aktif), HIGH untuk relay mati (non-
aktif)
  digitalWrite(RELAY_POMPA, shouldPumpBeOn ? LOW : HIGH);
  digitalWrite(RELAY_VALVE, shouldValveBeOn ? LOW : HIGH);
}

void manualControl() {
  digitalWrite(RELAY_POMPA, statusPompa ? LOW : HIGH);
  digitalWrite(RELAY_VALVE, statusValve ? LOW : HIGH);
}

void updateLCD() {
  static unsigned long lastLCDMillis = 0;
  if (millis() - lastLCDMillis > 3000) {
    lastLCDMillis = millis();
    lcd.clear();

    switch (lcdState) {
      case 0:
        lcdPrint("pH", String(pH, 2));
        break;
      case 1:
        lcdPrint("TDS", String(tds, 1) + " ppm");
        break;
      case 2:
        lcdPrint("Turbidity", String(turbidity, 1) + " NTU");
        break;
      case 3:
        lcdPrint("Tinggi", String(ketinggian_cm, 1) + " cm");
        break;
    }
    lcdState = (lcdState + 1) % 4;
  }
}

void lcdPrint(String line1, String line2) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(line1);
  lcd.setCursor(0, 1);
  lcd.print(line2);
}

void printSensorData() {
  Serial.println("\n--- Sensor Readings ---");
  Serial.printf("Turbidity: %.2f NTU\n", turbidity);
  Serial.printf("pH: %.2f\n", pH);
  Serial.printf("TDS: %.2f ppm\n", tds);
  Serial.printf("Ketinggian: %.1f cm\n", ketinggian_cm);
  Serial.printf("Mode: %s | Pompa: %s | Valve: %s\n",
                modeKontrol.c_str(),
                digitalRead(RELAY_POMPA) == LOW ? "ON" : "OFF",
                digitalRead(RELAY_VALVE) == LOW ? "ON" : "OFF");
}
```