LAMPIRAN

Lampiran 1 Surat Kesediaan Membimbing TA Pembimbing 1

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini :

Nama	: Arif Rakhman, SE, S.Pd, M.Kom
NIDN	: 0623118301
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	: Dio Fadli Saputro
NIM	: 22040097
Program Studi	: Diploma III Teknik Komputer

Judul TA : RANCANG BANGUN ALAT MONITORING SUHU, KELEMBABAN, DAN AMONIA UNTUK KANDANG BEBEK DAY OLD DUCK DI KTTI BERKAH ABADI TEGAL BERBASIS IOT

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 11 April 2025

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

Lampiran 2 Surat Kesediaan Membimbing TA Pembimbing 2

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini :

Nama	: Achmad Sutanto, S.Kom.,M.Tr.T
NIDN	: 0618058902
NIPY	: 11.012.128
Jabatan Struktural	: Subag Jaringan dan Server
Jabatan Fungsional	: Asisten Ahli

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

Nama	: Dio Fadli Saputro
NIM	: 22040097
Program Studi	: Diploma III Teknik Komputer

Judul TA : RANCANG BANGUN ALAT MONITORING SUHU, KELEMBABAN, DAN AMONIA UNTUK KANDANG BEBEK DAY OLD DUCK DI KTTI BERKAH ABADI TEGAL BERBASIS IOT

Dengan pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 11 April 2025

Mengetahui	
Ka. Prodi DIII Teknik Komputer,	Dosen Pembimbing II,
Ida Afriliana.ST, M.Kom.	Achmad Sutanto, S.Kom.,M.Tr.T
NIPY. 12.013.168	NIPY. 11.012.128

Lampiran 3 Surat Observasi

POLITEKNIK HARAPAN BERSAMA

D-3 Teknik Komputer

Nomor : 224.03/KOM-PHB/II/2025
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Pimpinan KTTI BERKAH ABADI

Di Jl. Mataram, Pesurungan Lor, Kec. Margadana, Kota Tegal, Jawa Tengah 52147

Dengan Hormat,

Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester V Program Studi D III Teknik Komputer Politeknik Harapan Bersama Tegal, Maka dengan ini kami mengajukan izin observasi pengambilan data di KTTI BERKAH ABADI yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No	NIM	NAMA	NO HP
1	22040097	DIO FADLI SAPUTRO	087727914156
2	22040104	IRGI MIFTAKHUL HUSNI	088239554216

Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 19 Februari 2025
Ka. Prodi DIII Teknik Komputer
Politeknik Harapan Bersama Tegal

Ida Afriliana, ST, M.Kom
NIPY. 12.013.168

Jl. Mataram No. 9 Kota Tegal 52143, Jawa Tengah, Indonesia.
(0283)553000
komputer@politegal.ac.id
politegal.ac.id

Lampiran 4 Source Code

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

LiquidCrystal_I2C lcd(0x27, 16, 2);
const char* ssid = "hellopoco";
const char* password = "hellopoco";

String host = "https://dod-guard.dheimuts.web.id/api/";

String urlPostData = host + "postData";
String urlGetcontrol = host + "control/status";
String urlPostcontrol = host + "control/update";

const int DHTPIN = 4; // Pin sensor DHT22
const int DHTTYPE = DHT22; // Type sensor

DHT dht(DHTPIN, DHTTYPE);

const int MQ135_PIN = 33;
const float RL = 10.0;
const float RO = 8.5;

const int relayKipas = 23;
const int relayLampu = 19;
const int tombolKipas = 12;
const int tombolLampu = 13;
const int tombolMode = 14;

bool modeOtomatis = true;
bool kipasState = false;
bool lampuState = false;

const char* ntpServer = "pool.ntp.org";
const long gmtOffset_sec = 7 * 3600;
const int daylightOffset_sec = 0;
const float SUHU_MIN = 25.0;
const float SUHU_MAKS = 35.0;
const float NH3_THRESHOLD = 10.0;

unsigned long lastSendTime = 0;
const unsigned long sendInterval = 10000; // Kirim setiap 10 detik

const unsigned long debonce_delay = 50;

unsigned long lastManualStatusRequest = 0;
const unsigned long intervalAmbilStatus = 5000; //Ambil setiap 5
detik manual
```

```
void setup() {
 Serial.begin(115200);
 pinMode(relayKipas, OUTPUT);
 pinMode(relayLampu, OUTPUT);
 pinMode(tombolKipas, INPUT_PULLUP);
 pinMode(tombolLampu, INPUT_PULLUP);
 pinMode(tombolMode, INPUT_PULLUP);
 digitalWrite(relayKipas, HIGH); // Relay aktif LOW, jadi HIGH =
OFF
 digitalWrite(relayLampu, HIGH); // Relay aktif LOW, jadi HIGH =
OFF

 lcd.init();
 lcd.backlight();
 dht.begin();

 tampilBooting();
 connectToWiFi();
 syncTime();
 buatIkonLCD();

 Serial.println("Menunggu pemanasan sensor MQ135...");
 pemanasanSensorGas(10); // Waktu pemanasanDuckDash
 Serial.println("Sensor siap.");
}

void loop() {
 tampilkanLayarBergilir(); // Mengatur tampilan LCD bergantian
 cekTombolMode(); // Mengatur mode otomatis/manual
 if (!modeOtomatis) cekKontrolManual(); // Jika manual, cek tombol
control

 if (!modeOtomatis && millis() - lastManualStatusRequest >
intervalAmbilStatus) {
 ambilStatusKontrol();
 lastManualStatusRequest = millis();
 }
 if (millis() - lastSendTime >= sendInterval) {
 lastSendTime = millis();
 float suhu, hum, ppm;
 bacaSemuaSensor(suhu, hum, ppm);

 if (modeOtomatis) kontrolOtomatis(suhu, ppm);

 tampilkanDataSerial(suhu, hum, ppm); // Tampilkan data di Serial
Monitor
 kirimDataKeServer(suhu, hum, ppm); // KIRIM DATA KE LARAVEL
 }

 cekKoneksiWiFi(); // Memastikan koneksi WiFi tetap terjaga
 delay(100); // Tambahkan sedikit delay untuk stabilitas
}
```

```
void bacaSemuaSensor(float& suhu, float& hum, float& ppm) {
 suhu = dht.readTemperature();
 hum = dht.readHumidity();
 ppm = hitungPPM();

 for (int i = 0; i < 3 && isnan(suhu); i++) {
 delay(100);
 suhu = dht.readTemperature();
 }
 for (int i = 0; i < 3 && isnan(hum); i++) {
 delay(100);
 hum = dht.readHumidity();
 }
}

float hitungPPM() {
 int analogValue = analogRead(MQ135_PIN);
 float voltage = analogValue * (3.3 / 4095.0); // Asumsi VCC 3.3V
untuk ESP32 ADC
 if (voltage < 0.01) return 0; // Hindari pembagian dengan nol
atau nilai sangat kecil
 float rs = ((3.3 - voltage) / voltage) * RL;
 float ratio = rs / RO; // Rasio RS/RO

 return pow(10, (-1.8 * log10(ratio) + 1.5));
}
void kontrolOtomatis(float suhu, float ppm) {
 if (ppm > NH3_THRESHOLD) {
 digitalWrite(relayKipas, LOW); // Kipas ON
 kipasState = true;
 digitalWrite(relayLampu, HIGH); // Lampu OFF
 lampuState = false;
 } else if (suhu > SUHU_MAKS) {
 digitalWrite(relayKipas, LOW); // Kipas ON
 kipasState = true;
 digitalWrite(relayLampu, HIGH); // Lampu OFF
 lampuState = false;
 } else if (suhu < SUHU_MIN) {
 digitalWrite(relayLampu, LOW); // Lampu ON
 lampuState = true;
 digitalWrite(relayKipas, HIGH); // Kipas OFF
 kipasState = false;
 } else { // Kondisi normal
 digitalWrite(relayLampu, LOW); // Lampu ON
 lampuState = true;
 digitalWrite(relayKipas, HIGH); // Kipas OFF
 kipasState = false;
 }
}
```

```
void cekKontrolManual() {
 static bool lastKipas = HIGH, lastLampu = HIGH;
 static unsigned long lastDebounceTimeKipas = 0;
 static unsigned long lastDebounceTimeLampu = 0;

 bool readingKipas = digitalRead(tombolKipas);
 bool readingLampu = digitalRead(tombolLampu);
 unsigned long currentTime = millis();

 if (readingKipas != lastKipas && currentTime -
lastDebounceTimeKipas > debonce_delay) {
 lastDebounceTimeKipas = currentTime;
 if (readingKipas == LOW) { // Tombol ditekan
 kipasState = !kipasState;
 digitalWrite(relayKipas, kipasState ? LOW : HIGH);
 kirimStatusKeServer("fan", kipasState); // <- kirim ke server
 }
 }
if (readingLampu != lastLampu && currentTime -
lastDebounceTimeLampu > debonce_delay) {
 lastDebounceTimeLampu = currentTime;
 if (readingLampu == LOW) { // Tombol ditekan
 lampuState = !lampuState;
 digitalWrite(relayLampu, lampuState ? LOW : HIGH);
 kirimStatusKeServer("lamp", lampuState); // <- kirim ke server
 }
 }

 lastKipas = readingKipas;
 lastLampu = readingLampu;
}
void cekTombolMode() {
 static bool lastState = HIGH;
 static unsigned long lastDebounceTime = 0;

 bool currentState = digitalRead(tombolMode);
 unsigned long currentTime = millis();

 if (currentState != lastState && currentTime - lastDebounceTime >
debonce_delay) {
 lastDebounceTime = currentTime;
 if (currentState == LOW) { // Tombol ditekan
 modeOtomatis = !modeOtomatis;
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("Mode: ");
 lcd.print(modeOtomatis ? "Otomatis" : "Manual");
 delay(1000); // Feedback ke pengguna
 }
 }
 lastState = currentState;
}
```

```
void kirimDataKeServer(float suhu, float hum, float ppm) {
 if (WiFi.status() == WL_CONNECTED) {
 HTTPClient http;
 http.setTimeout(5000); // Timeout 5 detik

 http.begin(urlPostData);

 http.addHeader("Content-Type", "application/x-www-form-
urlencoded");

 String postData = "temperature=" + String(suhu) + "&humidity=" +
String(hum) + "&ammonia=" + String(ppm);

 int code = http.POST(postData);
 Serial.printf("Kirim ke server... Response: %d\n", code);
 if (code > 0) Serial.println("Respon: " + http.getString());
 else Serial.printf("Gagal kirim. Error: %s\n",
http.errorToString(code).c_str());
 http.end();
 } else {
 Serial.println("WiFi tidak tersambung! Tidak bisa kirim data.");
 }
}

void ambilStatusKontrol() {
 if (WiFi.status() == WL_CONNECTED) {
 HTTPClient http;
 http.setTimeout(5000); // Timeout request dalam 5 detik

 http.begin(urlGetcontrol);

 int httpCode = http.GET(); // Kirim request GET
 if (httpCode == 200) {
 String payload = http.getString();
 Serial.println("Respons status kontrol:");
 Serial.println(payload);

 StaticJsonDocument<256> doc;
 DeserializationError error = deserializeJson(doc, payload);

if (error) {
 Serial.print("Gagal parsing JSON: ");
 Serial.println(error.c_str());
} else {
 String kipas = doc["fan"];
 String lampu = doc["lamp"];
 if (!modeOtomatis) {
 bool serverKipasState = (kipas == "ON");
 bool serverLampuState = (lampu == "ON");
 if (serverKipasState != kipasState) {
 kipasState = serverKipasState;
 digitalWrite(relayKipas, kipasState ? LOW : HIGH);
 Serial.println("Sinkron kipas dari server: " + kipas);
 }
```

```
 if (serverLampuState != lampuState) {
 lampuState = serverLampuState;
 digitalWrite(relayLampu, lampuState ? LOW : HIGH);
 Serial.println("Sinkron lampu dari server: " + lampu);
 }
 }
 }
 } else {
 Serial.printf("Gagal ambil data kontrol. Kode: %d\n", httpCode);
 }
 http.end();
 } else {
 Serial.println("WiFi tidak terhubung. Tidak bisa ambil data
kontrol.");
 }
}

void kirimStatusKeServer(String perangkat, bool status) {
 if (WiFi.status() == WL_CONNECTED) {
 HTTPClient http;
 http.setTimeout(5000);

 http.begin(urlPostcontrol);

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

String payload = "{\"device\": \"" + perangkat + "\", \"status\":
\"" + (status ? "ON" : "OFF") + "\"}";

 int httpCode = http.POST(payload);
 String response = http.getString(); // Untuk debug

if (httpCode == 200) {
 Serial.println("✅ Status " + perangkat + " dikirim: " + (status
? "ON" : "OFF"));
 Serial.println("Respon server: " + response);
 } else {
 Serial.println("❌ Gagal kirim status ke server. Kode: " +
String(httpCode));
 }

 http.end();
 } else {
 Serial.println("❌ WiFi tidak terhubung.");
 }
}

void tampilkanDataSerial(float suhu, float hum, float ppm) {
 Serial.println("==================================");
 Serial.printf("Suhu : %.1f °C\n", suhu);
 Serial.printf("Kelembapan : %.1f %%\n", hum);
 Serial.printf("NH3 (PPM) : %.1f ppm\n", ppm);
 Serial.printf("Kipas : %s\n", kipasState ? "ON" : "OFF");
 Serial.printf("Lampu : %s\n", lampuState ? "ON" : "OFF");
 Serial.printf("Mode : %s\n", modeOtomatis ? "Otomatis" :
```

```
 struct tm timeinfo;
 if (getLocalTime(&timeinfo)) {
 Serial.printf("Waktu : %s, %02d-%02d-%04d %02d:%02d:%02d\n",
 hariIndo(timeinfo.tm_wday).c_str(),
 timeinfo.tm_mday, timeinfo.tm_mon + 1, timeinfo.tm_year + 1900,
 timeinfo.tm_hour, timeinfo.tm_min, timeinfo.tm_sec);
 } else {
 Serial.println("Waktu : Error mendapatkan waktu");
 }
 Serial.println("================================\n");
}

void tampilBooting() {
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("Smart DOD");
 lcd.setCursor(0, 1);
 lcd.print("Loading...");
 delay(2000);
 lcd.clear();
}

void connectToWiFi() {
 lcd.setCursor(0, 0);
 lcd.print("WiFi Connect...");
 Serial.print("Connecting to WiFi...");
 WiFi.begin(ssid, password);
 int attempts = 0;
 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 Serial.print(".");
 attempts++;
 if (attempts > 20) { // Coba 10 detik, jika gagal restart
 Serial.println("\nFailed to connect to WiFi. Restarting ESP...");
 lcd.clear();
 lcd.print("WiFi Error!");
 lcd.setCursor(0, 1);
 lcd.print("Restarting...");
 delay(2000);
 ESP.restart();
 }
 }
 lcd.clear();
 lcd.print("WiFi Terhubung!");
 Serial.println("\nWiFi Connected!");
 Serial.print("IP Address: ");
 Serial.println(WiFi.localIP());
 delay(1500);
 lcd.clear();
}
```

```
void syncTime() {
 configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
 struct tm timeinfo;
 lcd.setCursor(0, 0);
 lcd.print("Sync Waktu...");
 int attempts = 0;
 while (!getLocalTime(&timeinfo)) {
 Serial.print(".");
 delay(500);
 attempts++;
 if (attempts > 20) { // Coba 10 detik, jika gagal
 Serial.println("\nFailed to sync time. Continuing without time
sync.");
 lcd.clear();
 lcd.print("Time Sync Err!");
 delay(1000);
 break; // Lanjut tanpa waktu tersinkron
 if (getLocalTime(&timeinfo)) {
 lcd.clear();
 lcd.print("Waktu Disinkron!");
 delay(1000);
 }
}

void cekKoneksiWiFi() {
 static unsigned long lastCheck = 0;
 if (millis() - lastCheck >= 10000) { // Cek setiap 10 detik
 lastCheck = millis();
 if (WiFi.status() != WL_CONNECTED) {
 Serial.println("WiFi terputus! Mencoba menghubungkan
kembali...");
 lcd.clear();
 lcd.print("WiFi Lost!");
 lcd.setCursor(0, 1);
 lcd.print("Reconnecting...");
 WiFi.reconnect(); // Coba reconnect
 unsigned long reconnectStart = millis();
 while (WiFi.status() != WL_CONNECTED && millis() - reconnectStart
< 10000) { // Coba 10 detik
 delay(500);
 Serial.print(".");
 }
if (WiFi.status() == WL_CONNECTED) {
 Serial.println("\nWiFi reconnected!");
 lcd.clear();
 lcd.print("WiFi Reconnected!");
 delay(1500);
 } else {
 Serial.println("\nFailed to reconnect WiFi. Restarting
ESP32...");
 lcd.clear();
 lcd.print("Reconnect Fail!");
 lcd.setCursor(0, 1);
 lcd.print("Restarting...");
 delay(2000);
 ESP.restart();
```

```
 }
 }
 }
}

String hariIndo(int hari) {
 String namaHari[] = { "Minggu", "Senin", "Selasa", "Rabu",
"Kamis", "Jumat", "Sabtu" };
 return namaHari[hari % 7];
}

unsigned long lastSwitchTime = 0;
int currentScreen = 0;
const unsigned long screenInterval = 4000; // Ganti layar setiap 4
detik

void tampilkanLayarBergilir() {
 if (millis() - lastSwitchTime > screenInterval) {
 lastSwitchTime = millis();
 currentScreen = (currentScreen + 1) % 4; // Ada 4 layar (0, 1, 2,
3)
 lcd.clear();
 switch (currentScreen) {
 case 0: tampilkanStatusRelay(); break;
 case 1: tampilkanSensorDHT(); break;
 case 2: tampilkanGasAmonia(); break;
 case 3: tampilkanWaktu(); break;
 }
 }
}

void tampilkanStatusRelay() {
 lcd.setCursor(0, 0);
 lcd.write(byte(2)); // Ikon kipas
 lcd.print(" Kipas: ");
 lcd.print(kipasState ? "ON" : "OFF");
 lcd.setCursor(0, 1);
 lcd.write(byte(0)); // Ikon api/lampu (atau sesuaikan jika ada
ikon lampu khusus)
 lcd.print(" Lampu: ");
 lcd.print(lampuState ? "ON" : "OFF");
}

void tampilkanSensorDHT() {
 float suhu = dht.readTemperature();
 float hum = dht.readHumidity();
 lcd.setCursor(0, 0);
 lcd.write(byte(0)); // Ikon api/suhu
 lcd.print(" Suhu: ");
 // Tampilkan "N/A" jika pembacaan gagal
 if (isnan(suhu)) lcd.print("N/A");
 else {
 lcd.print(suhu, 1);
 lcd.print((char)223);
 lcd.print("C");
 }
```

```
lcd.setCursor(0, 1);
 lcd.write(byte(1)); // Ikon air/kelembaban
 lcd.print(" Lembap: ");
 if (isnan(hum)) lcd.print("N/A");
 else {
 lcd.print(hum, 1);
 lcd.print("%");
 }
}

void tampilkanGasAmonia() {
 float ppm = hitungPPM();
 lcd.setCursor(0, 0);
 lcd.print("NH3: ");
 lcd.print(ppm, 1);
 lcd.print(" ppm");
 lcd.setCursor(0, 1);
 lcd.write(byte(ppm > NH3_THRESHOLD ? 3 : 4)); // Ikon warning/cek
 lcd.print(ppm > NH3_THRESHOLD ? " Tinggi" : " Aman");
}

void tampilkanWaktu() {
 struct tm timeinfo;
 if (getLocalTime(&timeinfo)) {
 lcd.setCursor(0, 0);
 lcd.print(hariIndo(timeinfo.tm_wday));
 lcd.printf(", %02d-%02d", timeinfo.tm_mday, timeinfo.tm_mon + 1);
 lcd.setCursor(0, 1);
 lcd.printf("Jam %02d:%02d:%02d", timeinfo.tm_hour,
timeinfo.tm_min, timeinfo.tm_sec);
 } else {
 lcd.setCursor(0, 0);
 lcd.print("Waktu Error");
 lcd.setCursor(0, 1);
 lcd.print("NTP Failed");
 }
}

void buatIkonLCD() {
 byte ikonApi[] = { B00100, B01110, B01110, B11111, B11111,
B01110, B01110, B00000 };
 byte ikonAir[] = { B00100, B00100, B01010, B01010, B10001,
B10001, B01110, B00000 };
 byte ikonKipas[] = { B10101, B01010, B00100, B01010, B10101,
B01010, B00100, B00000 };
byte ikonWarning[] = { B00100, B00100, B00100, B00100, B00100,
B00000, B00100, B00000 };
 byte ikonCek[] = { B00000, B00001, B00010, B10100, B01000,
B00000, B00000, B00000 };

 lcd.createChar(0, ikonApi);
 lcd.createChar(1, ikonAir);
 lcd.createChar(2, ikonKipas);
 lcd.createChar(3, ikonWarning);
 lcd.createChar(4, ikonCek);
}
```

```
void pemanasanSensorGas(int durasiDetik) {
 lcd.clear();
 lcd.setCursor(0, 0);
 lcd.print("MQ135 Warm-up");

 for (int i = durasiDetik; i > 0; i--) {
 lcd.setCursor(0, 1);
 lcd.print("Tunggu ");
 lcd.print(i);
 lcd.print("s "); // Spasi untuk membersihkan sisa teks
 Serial.printf("Pemanasan MQ135... %d detik\n", i);
 delay(1000);
 }

 lcd.clear();
 lcd.print("Sensor Siap!");
 Serial.println("Sensor MQ135 siap digunakan.");
 delay(1000);
 lcd.clear();
}
```

Lampiran 5 Foto Dokumentasi