

# LAMPIRAN

Lampiran 1. Surat kesediaan dosen pembimbing 1

**SURAT KESEDIAAN MEMBIMBING TA**

Yang bertanda tangan di bawah ini:

Nama : Mohammad Humam, M.Kom  
NIDN : 0618117901  
NIPY : 12.002.007  
Jabatan Struktural : Dosen Tetap  
Jabatan Fungsional : Lektor

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

Nama : Mohamad Fikri Faqih  
NIM : 21040009  
Program Studi : Teknik Komputer

Judul TA : RANCANG BANGUN ALAT PEMBERSIH DAN  
PENYORTIR UKURAN TELOR ASIN  
BERBASIS MICROCONTROLLER

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 18 Maret 2024

Mengetahui  
Ka Prodi DIII Teknik Komputer,



Ida Afriliana, ST, M.Kom  
NIPY. 12.013.168

Dosen Pembimbing I,

Mohammad Humam, M.Kom  
NIPY. 12.002.007

Lampiran 2. Surat kesediaan dosen pembimbing 2

**SURAT KESEDIAAN MEMBIMBING TA**

Yang bertanda tangan di bawah ini:

Nama : M. Teguh Prihandoyo, M.Kom  
NIDN : 0607117001  
NIPY : 02.005.012  
Jabatan Struktural : Dosen Tetap  
Jabatan Fungsional : Asisten Ahli

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

Nama : Mohamad Fikri Faqih  
NIM : 21040009  
Program Studi : Teknik Komputer

Judul TA : RANCANG BANGUN ALAT PEMBERSIH DAN  
PENYORTIR UKURAN TELUR ASIN BERBASIS  
MIKROKONTROLLER

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, Juni 2024

Mengetahui  
Ka. Prodi DIII Teknik Komputer,

Dosen Pembimbing II,



Ida Afriliana, ST, M.Kom  
NIPY. 12.013.168

M. Teguh Prihandoyo, M.Kom  
NIPY. 02.005.012

### Lampiran 3. Surat Observasi



**POLITEKNIK HARAPAN BERSAMA**  
The First Vocational Computer

D-3 Teknik Komputer

No. : 008.03/KMP.PHB/VI/2024  
Lampiran : -  
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Pimpinan KTTI ( Kelompok Tani Ternak Itik ) Berkah Abadi  
Jln. Mataram, Pesurungan lor, Kec. Margadana , Kota Tegal

Dengan Hormat,

Schubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester VI (Genap) Program Studi D III Teknik Komputer Politeknik Harapan Bersama Tegal, Maka dengan ini kami mengajukan izin observasi pengambilan data di KTTI ( Kelompok Tani Ternak Itik ) Berkah Abadi yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No.	NIM	Nama	No. HP
1	21040009	MOHAMAD FIKRI FAQIH	085943651480
2	21040039	YOGI WAHYUNI KASANDRA	085716306632

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

Tegal, 10 Juni 2024  
Ka. Prodi DIII Teknik Komputer  
Politeknik Harapan Bersama Tegal  
  
**Ida Afriliana, ST, M.Kom**  
NIPY. 12.013.168

## Lampiran 4 Surat Balasan Observasi

**Pimpinan KTTI (Kelompok Tani Ternak Itik) Berkah Abadi**  
Jln. Mataram ,Pesurungan Lor,Kec.Margadana,Kota Tegal  
085725383974

Tegal, 21 Mei 2024

Kepada Yth.  
Politeknik Harapan Bersama  
Jln. Mataram No.9 Pesurungan Lor Kec.Margadana,Kota Tegal

Dengan hormat,

Terima kasih atas surat permintaan observasi yang telah Anda kirimkan kepada kami. Kami sangat menghargai minat dan perhatian Anda terhadap proses pengolahan telur asin yang kami lakukan di **KTTI (Kelompok Tani Ternak Itik) Berkah Abadi**

Dengan senang hati, kami menginformasikan bahwa permintaan observasi Anda kami terima dengan baik. Berikut adalah beberapa informasi terkait observasi yang akan dilakukan:

**Tanggal Observasi** : 21 Mei 2024

**Waktu Observasi** : 15:40 WIB

**Alamat Lokasi Observasi** : Jln. Mataram ,Pesurungan Lor,Kec.

Margadana, Kota Tegal

Terima kasih atas perhatian dan kerjasamanya. Kami menantikan kehadiran Anda di **KTTI (Kelompok Tani Ternak Itik) Berkah Abadi**

Hormat kami,

Pimpinan KTTI (Kelompok Tani Ternak Itik) Berkah Abadi.



Radjum



Lampiran 5. Dokumentasi



## Lampiran 6. *Source Code*

```
#include <Arduino.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
#include "HX711.h"
#include <ESP32_Servo.h>
#include <ArduinoJson.h>
#include <EEPROM.h>
#ifdef ESP32
#include <WiFi.h>
#include <HTTPClient.h>
#include <AsyncTCP.h>
#elif defined(ESP8266)
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ESPAsyncTCP.h>
#endif
#include <ESPAsyncWebServer.h>

const int pinDT_loadcell = 4;
const int pinSCK_loadcell = 5;
HX711 scale_loadcell(5, 4);;
Servo servoESP32;
int port_servoESP32 = 17;
int relay_conveyor = 18;
int relay_pompa = 19;
int relay_pembersih = 15;
extern const char* default_nama_ssid = "wifi-iot";
extern const char* default_password = "password-iot";
extern const char* default_server =
"http://labrobotika.go-web.my.id/server.php?apikey=";
extern const char* default_apikey =
"4a9f24c30244c2b1a77dad07dbe01ed7";
String nama_ssid;
String password;
String server_url;
String apikey;
AsyncWebServer server(80);

void lcd_i2c(String text = "",int kolom = 0, int baris =
0)
{
byte bar[8] =
{B11111,B11111,B11111,B11111,B11111,B11111,B11111,};
if (text == "")
{
lcd.begin(); //jika error pakai lcd.init();
lcd.backlight();
lcd.createChar(0, bar);
lcd.setCursor(0,0);
lcd.print("Loading..");
for (int i=0; i<16; i++)
{
lcd.setCursor(i, 1);
```

```

        lcd.write(byte(0));
        delay(100);
    }
    delay(50);
    lcd.clear();
}
else
{
    lcd.setCursor(kolom,baris);
    lcd.print(text + "                ");
}
}

float baca_loadcell() {
return scale_loadcell.get_units(10);
}
void posisi_servoESP32(int posisi)
{
servoESP32.write(posisi);
}
void relay_conveyor_on()
{
digitalWrite(relay_conveyor ,LOW);
}
void relay_conveyor_off()
{
digitalWrite(relay_conveyor ,HIGH);
}
void relay_pompa_on()
{
digitalWrite(relay_pompa ,LOW);
}
void relay_pompa_off()
{
digitalWrite(relay_pompa ,HIGH);
}
void relay_pembersih_on()
{
digitalWrite(relay_pembersih ,LOW);
}
void relay_pembersih_off()
{
digitalWrite(relay_pembersih ,HIGH);
}
void debug(String message) {
    Serial.println(message);
    //tampilkan jika menggunakan lcd
    lcd.clear();
    lcd_i2c(message);
}
void writeStringToEEPROM(int address, const String &str)
{
    int len = str.length();
    EEPROM.write(address, len);
    for (int i = 0; i < len; i++) {
        EEPROM.write(address + 1 + i, str[i]);
    }
}

```



```

    }
}
String readStringFromEEPROM(int address) {
    int len = EEPROM.read(address);
    char data[len + 1];
    for (int i = 0; i < len; i++) {
        data[i] = EEPROM.read(address + 1 + i);
    }
    data[len] = '\0';
    return String(data);
}
void saveCredentialsToEEPROM() {
    EEPROM.begin(512);
    writeStringToEEPROM(0, nama_ssid);
    writeStringToEEPROM(64, password);
    writeStringToEEPROM(128, server_url);
    writeStringToEEPROM(192, apikey);
    EEPROM.commit();
    debug("Konfigurasi yang disimpan ke EEPROM:");
    debug("nama_ssid: " + nama_ssid);
    debug("Password: " + password);
    debug("Server URL: " + server_url);
    debug("API Key: " + apikey);
}
void loadCredentialsFromEEPROM() {

    EEPROM.begin(512);
    nama_ssid = readStringFromEEPROM(0);
    password = readStringFromEEPROM(64);
    server_url = readStringFromEEPROM(128);
    apikey = default_apikey;
    if (nama_ssid.length() == 0) {
        nama_ssid = default_nama_ssid;
        debug("SSID Default.");
    } else {
        debug("SSID EEPROM.");
    }
    if (password.length() == 0) password =
default_password;
    if (server_url.length() == 0) server_url =
default_server;
    if (apikey.length() == 0) apikey = default_apikey;
    debug("SSID :" + nama_ssid);
    delay(1000);
    debug("PASS :" + password);
    delay(1000);
    debug("URL :" + server_url);
    delay(1000);
    debug("API :" + apikey);
    delay(1000);
}
void setupWiFi() {
    WiFi.begin(nama_ssid.c_str(), password.c_str());
    int attempts = 0;
    while (WiFi.status() != WL_CONNECTED && attempts < 20)
{

```

```

        delay(2000);
        debug("Connect WiFi(" + (String)attempts + ")");
        attempts++;
    }
    if (WiFi.status() == WL_CONNECTED) {
        debug("Terhubung ke Wi-Fi");
        debug("nama_ssid: " + String(WiFi.SSID()));
        debug("IP: " + WiFi.localIP().toString());
        debug("System Ready");
        proses_iot("");

        } else {
            debug("Gagal terhubung..");
            WiFi.softAP("wifi-ESP");
            debug("AP: Wifi-ESP");
            delay(5000);
            debug("IP:" + WiFi.softAPIP().toString());
        delay(5000);
        server.on("/", HTTP_GET, [] (AsyncWebServerRequest
*request){
            String nama_ssidValue = (nama_ssid.length() > 0) ?
nama_ssid : default_nama_ssid;
            String passwordValue = (password.length() > 0) ?
password : default_password;
            String serverValue = (server_url.length() > 0) ?
server_url : default_server;
            String apiKeyValue = (apikey.length() > 0) ? apikey
: default_apikey;
            String htmlContent = R"(
                <!DOCTYPE html>
                <html>
                <head>
                <title>ESP32 WiFi Configuration</title>
                <style>
                body {
                    font-family: Arial, sans-serif;
                    margin: 20px;
                }
                input[type="text"],
                input[type="password"] {
                    width: 100%;
                    padding: 10px;
                    margin: 5px 0;
                    display: inline-block;
                    border: 1px solid #ccc;
                    border-radius: 4px;
                    box-sizing: border-box;
                }

                input[type="submit"]:hover {
                    background-color: #45a049;
                }
                .container {
                    padding: 20px;
                    border-radius: 5px;
                    background-color: #f2f2f2;

```

```

    }
    </style>
</head>
<body>
    <div style="max-width: 600px; margin: 20px
auto; padding: 20px; border: 1px solid #ccc; border-
radius: 5px; background-color: #f9f9f9;">
    <div class="container">
        <h2>ESP WiFi Configuration</h2>
        <form action="/save" method="post">
            <label
SSID:</label>
                for="nama_ssid">WiFi
            <input
                type="text"
                id="nama_ssid"
                name="nama_ssid"
                value=")" + nama_ssidValue + R"(
                required)><br>
            <label
Password:</label>
                for="password">WiFi
            <input
                type="text"
                id="password"
                name="password"
                value=")" + passwordValue + R"(
                required)><br>
            <label for="server">Server URL:</label>
            <input
                type="text"
                id="server"
                name="server"
                value=")" + serverValue + R"(
                required)><br>
            <label for="apikey">API Key:</label>
            <input
                type="text"
                id="apikey"
                name="apikey"
                value=")" + apiKeyValue + R"(
                required)><br>
            <input
                style=" width: 100%;color: #fff;
                background-color: green; padding: 10px 20px; text-
                decoration: none; border-radius: 4px;"
                type="submit"
                value="SAVE CONFIGURATION">
            </form>
            <br>
            <br>
            Kembali Ke pengaturan Awal :
            <a href="/reset" style="color: #fff;
background-color: red; padding: 10px 20px; text-
decoration: none; border-radius: 4px;">RESET DEFAULT</a>
        </div>
    </div>
</body>
</html>
    );
    request->send(200, "text/html", htmlContent);
};
server.on("/save",
                                HTTP_POST,
[] (AsyncWebServerRequest *request){
    if(request->args() > 0){ // Pastikan ada argumen
yang disampaikan
        for(uint8_t i = 0; i < request->args(); i++){
            if(request->argName(i) == "nama_ssid"){
                nama_ssid = request->arg(i);
            } else if(request->argName(i) == "password"){
                password = request->arg(i);
            } else if(request->argName(i) == "server"){

```

```

        server_url = request->arg(i);
    } else if(request->argName(i) == "apikey"){
        apikey = request->arg(i);
    }
}
saveCredentialsToEEPROM(); // Simpan konfigurasi
ke EEPROM
request->send(200, "text/html", R"(
    <div style="max-width: 600px; margin: 20px
auto; padding: 20px; border: 1px solid #ccc; border-
radius: 5px; background-color: #f9f9f9;">
        <h2 style="color: #4CAF50;">Konfigurasi
Berhasil Disimpan</h2>
        <p><br>Klik tombol dibawah ini untuk restart
esp <br><br><br><a href="/restart" style="color: #fff;
background-color: #4CAF50; padding: 10px 20px; text-
decoration: none; border-radius: 4px;">RESTART
ESP</a></p>
    </div>
</body>
)");
} else {
    request->send(400, "text/html", "Bad Request:
Tidak ada data yang disampaikan.");
}
});
server.on("/reset", HTTP_GET,
[] (AsyncWebServerRequest *request) {

    nama_ssid = default_nama_ssid;
    password = default_password;
    server_url = default_server;
    apikey = default_apikey;
    saveCredentialsToEEPROM();
    request->send(200, "text/html", R"(
        <div style="max-width: 600px; margin: 20px
auto; padding: 20px; border: 1px solid #ccc; border-
radius: 5px; background-color: #f9f9f9;">
            <h2 style="color: RED;">Konfigurasi Berhasil
Di Reset</h2>
            <p><br>Klik tombol dibawah ini untuk restart
esp <br><br><br><a href="/restart" style="color: #fff;
background-color: red; padding: 10px 20px; text-
decoration: none; border-radius: 4px;">RESTART
ESP</a></p>
        </div>
    )");

});
server.on("/restart", HTTP_GET,
[] (AsyncWebServerRequest *request) {

    request->send(200, "text/html", R"(
        <head>
        <meta http-equiv="refresh" content="5;url=/">
        </head>

```

```

<body>
  <div style="max-width: 600px; margin: 20px auto;
padding: 20px; border: 1px solid #ccc; border-radius:
5px; background-color: #f9f9f9;">
    <p><br>ESP Restart... <br><br></p>
  </div>
</body>
  )");

  delay(1000); // Tambahkan jeda sebelum merestart
  ESP.restart(); // Restart ESP
  request->redirect("/");
  });
}
server.begin();
}

int start_conveyor = 0;
int treshold = 1;
void proses_iot(String nilai) {
  if (WiFi.status() != WL_CONNECTED) return;

  WiFiClient client;
  HTTPClient http;
  String url = server_url + apikey + nilai; //
Menggunakan server_url
  url.replace(" ", "%20");
  Serial.println("Request URL: " + url);
  http.begin(client, url);
  int httpStatusCode = http.GET();
  if (httpStatusCode == HTTP_CODE_OK) {
    const size_t capacity = JSON_OBJECT_SIZE(1024);
    DynamicJsonDocument jsonDoc(capacity);
    String jsonResponse = http.getString();
    DeserializationError error =
deserializeJson(jsonDoc, jsonResponse);
    if (error) {
      Serial.println("Error parsing JSON: " +
String(error.c_str()));
      return;
    }
    for (int i = 1; i <= 10; i++) {
      String out = jsonDoc["out_" +
String(i)].as<String>();
      Serial.println("out_" + String(i) + ": " + out);
    }

    start_conveyor = jsonDoc["out_1"].as<int>();
    treshold = jsonDoc["out_2"].as<int>();
    Serial.println(start_conveyor);
    Serial.println(treshold);
  } else {
    Serial.println("Error Code: " +
String(httpStatusCode));
  }
  http.end();
}

```



```

}
void setup()
{
Serial.begin(9600);
lcd_i2c();
servoESP32.attach(port_servoESP32);
posisi_servoESP32(90);
pinMode(relay_conveyor, OUTPUT);
relay_conveyor_off();
pinMode(relay_pompa, OUTPUT);
relay_pompa_off();
pinMode(relay_pembersih, OUTPUT);
relay_pembersih_off();
EEPROM.begin(512);
loadCredentialsFromEEPROM();
setupWiFi();
  Serial.println("Before setting up the scale:");
  Serial.print("read: \t\t");
  Serial.println(scale_loadcell.read());      // print a
raw reading from the ADC

  Serial.print("read average: \t\t");
  Serial.println(scale_loadcell.read_average(20));  //
print the average of 20 readings from the ADC

  Serial.print("get value: \t\t");
  Serial.println(scale_loadcell.get_value(5));      //
print the average of 5 readings from the ADC minus the
tare weight (not set yet)

  Serial.print("get units: \t\t");
  Serial.println(scale_loadcell.get_units(5), 1);
  // print the average of 5 readings from the ADC
  minus tare weight (not set) divided
  scale_loadcell.set_scale(2280.f);                //
this value is obtained by calibrating the scale with
known weights; see the README for details
  scale_loadcell.tare();                          // reset the
scale to 0
  Serial.println("After setting up the scale:")
  Serial.print("read: \t\t");
  Serial.println(scale_loadcell.read());
  // print a raw reading from the ADC
  Serial.print("read average: \t\t");
  Serial.println(scale_loadcell.read_average(20));
  // print the average of 20 readings from the ADC
  Serial.print("get value: \t\t");
  Serial.println(scale_loadcell.get_value(5));      //
print the average of 5 readings from the ADC minus the
tare weight, set with tare()
  Serial.print("get units: \t\t");
  Serial.println(scale_loadcell.get_units(5), 1);
  // print the average of 5 readings from the ADC minus
tare weight, divided
  // by the SCALE parameter set with
  set_scale

```

```

        Serial.println("Readings:");
    }
    void servo_netral()
    {
        posisi_servoESP32(90);
    }
    void servo_besar()
    {
        posisi_servoESP32(0);
    }
    void servo_kecil()
    {
        posisi_servoESP32(180);
    }
    int looping ;
    int besar ;
    int kecil ;
    void loop(){
    float loadcell = baca_loadcell();
    if (loadcell<0)
    {
        loadcell = 0;
    }
    Serial.println("Berat: " + (String)loadcell + " gram");
    lcd_i2c("Berat: " + (String)loadcell + " g");
    looping = looping +1;
    if (looping > 5)
    {
        proses_iot("&besar=" + (String)besar + "&kecil=" +
        (String)kecil + "&loadcell=" + (String)loadcell);
        looping = 0;
    }
    if (start_conveyor == 1)
    {
        relay_conveyor_on();
        relay_pembersih_on();
        relay_pompa_on();
    }
    else
    {
        relay_conveyor_off();
        relay_pembersih_off();
        relay_pompa_off();
    }

    if (loadcell> 0.5 && loadcell< treshold )
    {
        kecil = kecil +1;
        servo_kecil();
        delay(2000);
        servo_netral();
    }
    else if (loadcell>treshold )
    {
        besar = besar +1;
        servo_besar();
    }

```

```

        delay(2000);
        servo_netral();
    }
    scale_loadcell.power_down();           // put the ADC
    in sleep mode
    delay(500);
    scale_loadcell.power_up();
    //servo_netral();
    //delay(4000);
    //servo_besar();
    //delay(4000);
    //servo_netral();
    //delay(4000);
    //servo_kecil();
    //delay(4000);
    //relay_conveyor_off();
    //delay(1000);
    //relay_conveyor_on();
    //delay(1000);
    //
    //relay_pompa_off();
    //delay(1000);
    //relay_pompa_on();
    //delay(1000);
    //
    //relay_pembersih_off();
    //delay(1000);
    //relay_pembersih_on();
    //delay(1000);
    //
    //delay(3000);
}

```