LAMPIRAN

1. Surat Izin Penelitian

PEMERINTAH KABUPATEN LABUHANBATU
DINAS PENDIDIKAN
SEKOLAH MENENGAH PERTAMA NEGERI 3 RANTAU UTARA
Jalan Padang Matinggi Rantauprapat, Padang Matinggi, Rantau Utara, Labuhanbatu
Sumatera Utara, Kode Pos : 21411 Telepon : (0624) – E-mail: smpn3ratu@gmail.com
NSS : 201070708050 NPSN : 10205238 Akreditasi : A

SURAT PERNYATAAN PENELITIAN
Nomor : 400.3.5/580/TU/SMPN.3/RU/2025

Yang bertanda tangan dibawah ini, Kepala Satuan Pendidikan SMPN 3 Rantau Utara Kecamatan Rantau Utara Kabupaten Labuhanbatu Provinsi Sumatera Utara, dengan ini menerangkan bahwa :

Nama	: FANI WULANDARI RAMBE
NIM	: 2208100023
Program Studi	: Teknologi Informasi
Fakultas	: Sains dan Teknologi

Berdasarkan surat Universitas Labuhanbatu Fakultas Sains dan Teknologi dengan nomor :1356/TI/FST-ULB/XII/2025 tanggal 15 Desember 2025 Perihal memohon izin melakukan Penelitian Tugas Akhir maka Mahasiswi tersebut diatas diberi izin untuk melaksanakan Penelitian Tugas Akhir di SMPN 3 Rantau Utara dengan judul: **"Rancang Bangun Sistem Monitoring Daya Listrik Berbasis IoT dengan Notofikasi WhatsApp pada SMPN 3 Rantau Utara"**

Demikian surat pernyataan penelitian ini diperbuat untuk dapat dipergunakan seperlunya.

Rantauprapat, 16 Desember 2025
Kepala Satuan Pendidikan
SMPN 3 Rantau Utara

SRI ZULIANI RITONGA, SE
NIP. 19760730 201212 2 001

2. Source Code Program ESP32

```
#include <PZEM004Tv30.h>
#include <LiquidCrystal_I2C.h>
#include <WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
// ===== WIFI =====
const char* ssid     = "SOLIN_PROJECT";
const char* password = "17AGUSTUS2025";
// ===== MQTT =====
const char* mqttServer   = "broker.hivemq.com";
const int   mqttPort     = 1883;
const char* topicSensor  = "fani/sensor";
const char* topicStatus  = "esp32/relay/status";
const char* topicCommand = "esp32/relay/command";
const char* topicLimit   = "fani/limit";
const char* topicNotif   = "fani/notifikasi";
// ===== PIN =====
#define RXD2 16
#define TXD2 17
#define RELAY_PIN 26
// ===== BATAS DAYA =====
float batasDaya = 900.0;
// ===== FLAG =====
bool relayManualOFF = false;
bool relayJustON    = false;  // flag: relay baru dihidupkan
int  skipCount      = 0;      // hitung skip pembacaan PZEM
// ===== SHARED VARIABLES =====
volatile bool  statusTrip      = false;
volatile float voltageShared   = 0;
volatile float currentShared   = 0;
volatile float powerShared     = 0;
unsigned long  tripStartTime   = 0;
bool           delaySelesai    = false;
bool           sudahKirimNotif = false;
// ==== OBJECT =====
PZEM004Tv30       pzem(Serial2, RXD2, TXD2);
LiquidCrystal_I2C lcd(0x27, 20, 4);
WiFiClient        espClient;
PubSubClient      mqttClient(espClient);
SemaphoreHandle_t xMutex;
// CUSTOM CHARACTER LCD
// Index 0: icon petir/kilat (voltage)
byte iconVolt[8] = {
  0b00010,
  0b00100,
  0b01000,
  0b11111,
  0b00010,
  0b00100,
  0b01000,
  0b00000
};
// Index 1: icon gelombang (current/arus)
byte iconArus[8] = {
  0b00000,
  0b01010,
```

```
  0b10101,
  0b10001,
  0b01010,
  0b00100,
  0b00000,
  0b00000
};
// Index 2: icon bohlam (power/daya)
byte iconDaya[8] = {
  0b01110,
  0b10001,
  0b10001,
  0b10001,
  0b01110,
  0b01110,
  0b00100,
  0b00000
};
// Index 3: icon batas/limit (segitiga peringatan)
byte iconLimit[8] = {
  0b00100,
  0b00100,
  0b01110,
  0b01010,
  0b11011,
  0b11111,
  0b00000,
  0b00000
};
// Index 4: icon centang / relay ON
byte iconON[8] = {
  0b00000,
  0b00001,
  0b00011,
  0b10110,
  0b11100,
  0b01000,
  0b00000,
  0b00000
};
// Index 5: icon X / relay OFF
byte iconOFF[8] = {
  0b00000,
  0b10001,
  0b01010,
  0b00100,
  0b01010,
  0b10001,
  0b00000,
  0b00000
};
// Index 6: icon WiFi / MQTT connected
byte iconWifi[8] = {
  0b01110,
  0b10001,
  0b00100,
  0b01010,
  0b00000,
```

```
  0b00100,
  0b00000,
  0b00000
};
// MQTT CALLBACK
void mqttCallback(char* topic, byte* payload, unsigned int length) {
  String msg = "";
  for (int i = 0; i < length; i++) msg += (char)payload[i];
  Serial.println("MQTT IN [" + String(topic) + "]: " + msg);
  if (String(topic) == topicCommand) {
    if (msg == "ON") {
      // Hidupkan relay DULU sebelum restart
      // Supaya saat ESP32 nyala lagi, relay sudah dalam posisi ON
      relayManualOFF = false;
      statusTrip     = false;
      digitalWrite(RELAY_PIN, HIGH);
      mqttClient.publish(topicStatus, "ON");
      Serial.println("Relay ON - ESP32 restart untuk reset PZEM...");
      delay(500); // beri waktu MQTT publish terkirim dulu
      ESP.restart(); // software reset - PZEM akan baca volt normal setelah restart
    }
    else if (msg == "OFF") {
      relayManualOFF = true;
      digitalWrite(RELAY_PIN, LOW);
      mqttClient.publish(topicStatus, "OFF");
      Serial.println("Relay OFF by WA command");
    }
  }
  if (String(topic) == topicLimit) {
    float newLimit = msg.toFloat();
    if (newLimit > 0) {
      batasDaya = newLimit;
      Serial.println("Limit daya diubah: " + String(batasDaya) + " W");
    }
  }
}// RECONNECT MQTT
void reconnectMQTT() {
  while (!mqttClient.connected()) {
    Serial.println("Connecting MQTT...");
    if (mqttClient.connect("ESP32-Fani")) {
      Serial.println("MQTT Connected");
      mqttClient.subscribe(topicCommand);
      mqttClient.subscribe(topicLimit);
    } else {
      Serial.println("MQTT gagal, coba lagi 3 detik...");
      delay(3000);
    }
  }
}// TASK CORE 0: SENSOR & LCD
void taskPembacaan(void *parameter) {
  while (true) {
    // Kalau relay baru dihidupkan, skip 3 pembacaan pertama
    // PZEM butuh waktu stabilisasi setelah relay ON
    if (relayJustON) {
      skipCount++;
      Serial.println("Skip pembacaan ke-" + String(skipCount) + " (PZEM stabilisasi)");
      if (skipCount >= 3) {
        relayJustON = false;
```

```
    skipCount   = 0;
    Serial.println("PZEM siap, mulai baca normal.");
   }
   delay(1000);
   continue;
  }
  float voltage = pzem.voltage();
  float current = pzem.current();
  float power   = pzem.power();
  if (xSemaphoreTake(xMutex, portMAX_DELAY)) {
   // Simpan nilai terakhir yang valid (tidak 0/nan)
   if (!isnan(voltage) && voltage > 0) voltageShared = voltage;
   if (!isnan(current))                currentShared = current;
   if (!isnan(power))                  powerShared   = power;
   xSemaphoreGive(xMutex);
  }
  // ===== LCD =====
  lcd.setCursor(0, 0);
  lcd.write(byte(0));          // icon petir
  lcd.print(" V:");
  lcd.print(isnan(voltage) ? 0.0 : voltage, 1);
  lcd.print("V ");
  lcd.setCursor(13, 0);
  lcd.write(byte(6));          // icon wifi
  lcd.print(mqttClient.connected() ? " OK  " : " --  ");
  // Baris 1: [gelombang] I: 0.45 A
  lcd.setCursor(0, 1);
  lcd.write(byte(1));          // icon arus
  lcd.print(" I:");
  lcd.print(isnan(current) ? 0.0 : current, 2);
  lcd.print(" A       ");
  // Baris 2: [bohlam] P: 98.2 W
  lcd.setCursor(0, 2);
  lcd.write(byte(2));          // icon daya
  lcd.print(" P:");
  lcd.print(isnan(power) ? 0.0 : power, 1);
  lcd.print(" W       ");
  // Baris 3: [limit] Lmt:900W [status]
  lcd.setCursor(0, 3);
  lcd.write(byte(3));          // icon limit
  lcd.print(" Lmt:");
  lcd.print(batasDaya, 0);
  lcd.print("W ");
  // Status relay di pojok kanan bawah
  lcd.setCursor(13, 3);
  if (relayManualOFF || statusTrip) {
   lcd.write(byte(5));         // icon X
   lcd.print(" MATI");
  } else {
   lcd.write(byte(4));         // icon centang
   lcd.print(" HIDUP");
  }
  // ===== PUBLISH SENSOR KE MQTT =====
  if (mqttClient.connected()) {
   StaticJsonDocument<200> doc;
   doc["voltage"] = isnan(voltage) ? 0 : voltage;
   doc["current"] = isnan(current) ? 0 : current;
   doc["power"]   = isnan(power)   ? 0 : power;
```

```
    doc["limit"]   = batasDaya;
    doc["trip"]    = statusTrip;
    char buf[200];
    serializeJson(doc, buf);
    mqttClient.publish(topicSensor, buf);
  }
  // ===== LOGIKA TRIP =====
  // Skip logika kalau relay dikunci manual OFF
  if (relayManualOFF) {
    delay(1000);
    continue;
  }
  // Deteksi overcurrent HANYA saat relay hidup (power valid)
  // Jangan cek power saat statusTrip=true karena relay sudah mati -> power pasti 0/NaN
  if (!statusTrip && !isnan(power) && power > batasDaya) {
    statusTrip      = true;
    tripStartTime   = millis();
    delaySelesai    = false;
    sudahKirimNotif = false;
    digitalWrite(RELAY_PIN, LOW); // matikan relay
    Serial.println("TRIP! Power: " + String(power) + "W");
  }
  if (statusTrip) {
    // Relay tetap mati selama menunggu
    digitalWrite(RELAY_PIN, LOW);
    // Setelah 5 detik, kirim notifikasi lalu kunci relay
    if (!delaySelesai && (millis() - tripStartTime >= 5000)) {
      delaySelesai = true;
      if (!sudahKirimNotif && mqttClient.connected()) {
        sudahKirimNotif = true;
        // Simpan data sebelum publish
        float v, i, p;
        if (xSemaphoreTake(xMutex, portMAX_DELAY)) {
          v = voltageShared;
          i = currentShared;
          p = powerShared;
          xSemaphoreGive(xMutex);
        }
        StaticJsonDocument<200> notif;
        notif["type"]    = "OVERCURRENT";
        notif["voltage"] = v;
        notif["current"] = i;
        notif["power"]   = p;
        notif["limit"]   = batasDaya;
        char buf[200];
        serializeJson(notif, buf);
        mqttClient.publish(topicNotif, buf);
        Serial.println("Notifikasi overcurrent dikirim");
        // Kunci relay mati, tunggu perintah WA
        statusTrip     = false;
        relayManualOFF = true;
        Serial.println("Relay dikunci mati. Ketik hidupkan di WA.");
      }
    }
  }
  delay(1000);
 }
}
```

```
// ============================
// TASK CORE 1: MQTT LOOP
// ============================
void taskMQTT(void *parameter) {
  while (true) {
    if (!mqttClient.connected()) reconnectMQTT();
    mqttClient.loop();
    delay(10);
  }
}
// ============================
// SETUP
// ============================
void setup() {
  Serial.begin(9600);
  Serial2.begin(9600, SERIAL_8N1, RXD2, TXD2);
  pinMode(RELAY_PIN, OUTPUT);
  digitalWrite(RELAY_PIN, HIGH); // relay hidup saat pertama nyala
  lcd.init();
  lcd.backlight();
  lcd.createChar(0, iconVolt);
  lcd.createChar(1, iconArus);
  lcd.createChar(2, iconDaya);
  lcd.createChar(3, iconLimit);
  lcd.createChar(4, iconON);
  lcd.createChar(5, iconOFF);
  lcd.createChar(6, iconWifi);
  lcd.clear();
  lcd.setCursor(3, 1);
  lcd.print("CONNECTING WIFI");
  xMutex = xSemaphoreCreateMutex();
  WiFi.begin(ssid, password);
  int wifiTimeout = 0;
  while (WiFi.status() != WL_CONNECTED && wifiTimeout < 40) {
    delay(500);
    wifiTimeout++;
  }
  lcd.clear();
  if (WiFi.status() == WL_CONNECTED) {
    lcd.setCursor(4, 1);
    lcd.print("WIFI  OK!");
    Serial.println("WiFi: " + WiFi.localIP().toString());
  } else {
    lcd.setCursor(3, 1);
    lcd.print("WIFI  GAGAL!");
  }
  mqttClient.setServer(mqttServer, mqttPort);
  mqttClient.setCallback(mqttCallback);
  reconnectMQTT();
  delay(2000);
  lcd.clear();
  xTaskCreatePinnedToCore(taskPembacaan, "Sensor", 10000, NULL, 1, NULL, 0);
  xTaskCreatePinnedToCore(taskMQTT,      "MQTT",   10000, NULL, 1, NULL, 1);
  Serial.println("System Started!");
}
void loop() {
  delay(1000);
}
```