LAPORAN KERJA PRAKTEK

International Course Program (ICP) Report

Creating an ESP 8266 Program for Web Monitoring Connectivity

Di Buat Oleh :

Andre Nugraha Wageardoyo 228160028

TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS MEDAN AREA

2025

LEMBAR PENGESAHAN

LAPORAN KERJA PRAKTEK

CREATING AN ESP 8266 PROGRAM FOR WEB MONITORING CONNECTIVITY

Diajukan Untuk Memenuhi Salah Satu Syarat Mata Kuliah Kerja Praktik Jenjang Studi S – 1 Program Studi Teknik Informatika

Oleh :

Andre Nugraha Wageardoyo (228160028)

Medan, 14 September 2025

Menyetujui

Dosen Pembimbing

Mahasiswa

Andre Nugraha Wageardoyo
NPM:228160028

Andre Hasudungan Lubis, S.Ti, M.Sc
NIDN:0126029101

Mengetahui,
Ketua Program Studi Teknik Informatika

Rizki Muliono, S.Kom, M.Kom
NIDN:0109038902

UNIVERSITAS MEDAN AREA
FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK INFORMATIKA

Kampus I : Jalan Kolam Nomor 1 Medan Estate ☎ (061) 7360168, Medan, 20223
Kampus II : Jalan Setiabudi Nomor 79 / Jalan Sei Serayu Nomor 70 A ☎ (061) 42402994, Medan, 20122
Website: www.teknik.uma.ac.id E-mail: univ_medanarea@uma.ac.id

BERITA ACARA DAN NILAI SEMINAR KERJA PRAKTEK

Pada hari ini 26 Juli 2025 telah diselenggarakan Seminar Kerja Praktek Program Studi Teknik Informatika untuk Tahun Akademik 2025/2026 atas :

Nama	: **Andre Nugraha Wageardoyo**
NIM	: 228160028
Program Studi	: Teknik Informatika
Jenjang Pendidikan	: S1 (Sarjana)
Judul Kerja Praktek	: *Creating an ESP 8266 Program for Web Monitoring Connectivity*
Tempat Seminar	: Ruang Seminar Fakultas Teknik
Tanda Tangan Pembawa Seminar	:
Nilai Pembawa Seminar	: 90 (A)

Seminar Kerja Praktek bersangkutan disetujui/tidak disetujui dengan catatan perubahan seperti yang tercantum pada tabel berikut :

Saran:	*Andre Hasudungan Lubis S.Ti., M.Sc* *Pembimbing Kerja Praktek*
Persetujuan Seminar :	
Saran:	*Rizki Muliono S.Kom, M.Kom* *Ka. Prodi*
Persetujuan Seminar :	

PANITIA SEMINAR KERJA PRAKTEK:

No.	Jabatan	Nama Dosen	Tanda Tangan
1	Pembimbing Kerja Praktek	Andre Hasudungan Lubis S.Ti., M.Sc	1
2	Ka. Prodi	Rizki Muliono S.Kom, M.Kom	2

Medan, 26 Juli 2025
Ketua Prodi.

Rizki Muliono S.Kom, M.Kom

ABSTRACT

The Design of an Automatic Solar Panel Web Monitoring System project aimed to address the inefficiency of manual solar panel monitoring by creating an accessible, real-time solution. This project, executed as part of the International Course Program, involved the design and implementation of an integrated system on a miniature house prototype. The methodology centered on programming an ESP8266 microcontroller to serve as both the system's brain and a web server. The microcontroller was developed to automatically track the sun's position using light sensors to control servo motors, thereby maximizing energy absorption. A static index.html web page was created and hosted directly on the ESP8266 to display real-time data on the panel's position, light intensity, and operational status. The project successfully demonstrated the system's ability to provide automated tracking and remote data access, proving the practical application of IoT technology. While the project achieved its core objectives, its limitation lies in the static nature of the web interface. Future recommendations include developing a more dynamic web application with a database for long-term data storage and trend analysis, and scaling the system for a real-world installation to test its full effectiveness. Keywords: Solar Panel, Monitoring, Real-Time, Web Interface, ESP8266, IoT

DAFTAR ISI

CHAPTER I
INTRODUCTION

1.1 Background

Electricity consumption continues to grow in line with economic development and technological innovation. Consequently, there has been an increase in the use of fossil fuels, which still serve as the primary fuel for power generation. Given the limited availability of fossil fuels, the search for alternative renewable energy sources, such as solar energy, wave energy, wind energy, and geothermal energy, is underway (L. Putriyana, 2021).

Sunlight energy, especially in Indonesia's tropical climate, is abundant. One limitation of solar energy is that it cannot be used directly to generate electricity. Solar energy must first be converted into electricity using devices such as solar panels, solar cell controllers, batteries, and inverters (Birhane, Modeling and Analysis of Photo-Voltaic Solar Panels under Constant Electric Load, 2019).
This solar energy is harnessed through a device called a solar panel (solar cell) by converting the absorbed sunlight into electrical energy (Anglistia, 2019).

Solar cells offer the advantage of being a practical energy source, as they don't require transmission and can be installed modularly at any location where needed.Unlike hydroelectric power plants, which can only be installed in specific watersheds, solar cells can be installed using renewable energy like sunlight, making them an environmentally friendly and sustainable alternative solution. However, existing solar-powered lighting systems are not yet fully efficient, especially in optimizing lighting during the day and at night. Therefore, research is needed into the design and development of innovative lighting systems, including those integrating reflective mirrors to maximize natural lighting during the day and solar-powered automatic lights that can turn on independently at night.

1.2 Activity Objectives

a) To improve access to lighting in rural areas not yet connected to conventional
electricity networks using renewable energy solutions.

b) To design an energy-efficient, sunlight-based lighting system by optimizing the
use of reflective mirrors to increase solar panel efficiency.

c) Develop automatic solar-powered lights that can turn on automatically at night, making them energy efficient and practical for residents.

d) Promote energy independence in rural communities by utilizing abundant local energy sources (solar energy) and reducing dependence on fossil fuels or PLN electricity.

1.3 Benefits of the Activity

a) Can save operational costs after initial installation.

b) Can learn about the use of solar energy, which can reduce carbon emissions and dependence on fossil fuels, while supporting environmental conservation efforts.

c) Can improve energy efficiency by using reflective mirrors that optimize the absorption of sunlight by solar panels, thereby increasing electrical output without increasing panel size.

CHAPTER II
LITERATURE REVIEW

The term solar panel is used colloquially for photovoltaic (PV) modules. A PV module is an assembly of photovoltaic cells mounted in a frame for installation. Photovoltaic cells use sunlight as an energy source and generate direct current electricity. A collection of PV modules is called a PV panel, and the panel system is an array. The array of photovoltaic systems supplies solar electricity to electrical appliances (Byregowda K. C, 2020).

Photovoltaic modules use light energy (photons) from the sun to generate electricity through the photovoltaic effect. Most modules use wafer-based crystalline silicon cells or thin-film cells. The structural (load-bearing) part of the module can be either a top layer or a back layer. The cells must be protected from mechanical damage and moisture. Most modules are rigid, but semi-flexible modules based on thin-film cells are also available. The cells are electrically connected in series with each other to achieve the desired voltage, and then in parallel to increase the electrical current. The wattage of a module is the mathematical product of the module's voltage and current. Manufacturing specifications for solar panels are obtained under standard operating conditions, which do not reflect the actual operating conditions encountered by the solar panels at the installation site.

Modules are electrically connected in series to achieve the desired output voltage or in parallel to provide the desired current (amperage) capacity of the solar panel or PV system (Pratik Pawar, 2018).

Electricity consumption continues to grow in line with economic development and technological innovation. Consequently, there has been an increase in the use of fossil fuels, which still serve as the primary fuel for power generation. Given the limited availability of fossil fuels, the search for alternative renewable energy sources, such as solar energy, wave energy, wind energy, and geothermal energy, is underway (L. Putriyana, 2021).

Solar energy, especially in Indonesia's tropical climate, is abundant. One limitation of solar energy is that it cannot be directly used to generate electricity. Solar energy must first be converted into electricity using devices such as solar panels, solar cell controllers, batteries, and inverters (Birhane, Modeling and Analysis of Photo-Voltaic Solar Panels under Constant Electric Load, 2019).

CHAPTER III
METHODOLOGY

To realize this innovative idea, the research was conducted using a prototype design and development approach. This method was chosen because it allowed us to first build and test the hybrid lighting system concept on a controlled scale.The core of this methodology is the creation of a mock-up or miniature house, which serves as a mini-laboratory to test the system's effectiveness before its implementation in the field.

3.1. Time and Place of Implementation

All activities, from concept design to final testing, were centered on the Medan Area University campus. This activity was carried out over a six-month period, from January to June 2025. This location selection ensured the availability of tools and a supportive environment for the development and experimentation process.

3.2. Tools and Materials

To realize this prototype, a series of tools and materials are required, divided into two main groups.

3.2.1. Tools

This equipment is used for the assembly process of electronic models and circuits, including:

- Scissors and Cutter
- Glue Gun and Korean Glue
- Ruler and Marker
- Soldering Iron, Screwdriver, and Pliers
- Paintbrush

3.2.2. Materials

The materials used consist of materials for building the physical model and electronic components that form the brain of the system.

- Materials for Making a House Model:
  - Plywood, cardboard, and popsicle sticks to form the building structure.
  - Paint for the finishing touch to make the model look representative.

- Hybrid Lighting System Components:

  • Panel Surya (Solar Cell): Komponen utama yang menangkap energi matahari untuk diubah menjadi Listrik

  • Cermin Reflektif: Solusi sederhana untuk memantulkan cahaya matahari ke dalam ruangan sebagai pencahayaan alami di siang hari.

  • Baterai dan Dudukan Baterai: Berfungsi sebagai unit penyimpan daya listrik yang akan digunakan pada malam hari.

  • Modul Cas (Solar Charge Controller): Perangkat pintar yang melindungi baterai dari pengisian berlebih dan memastikan efisiensi daya.

  • Lampu LED: Dipilih karena konsumsi dayanya yang rendah namun cahayanya terang.

  • Sensor Cahaya (LDR): Mata otomatis bagi sistem, yang memerintahkan lampu untuk menyala saat lingkungan menjadi gelap.

  • Saklar dan Kabel: Komponen penghubung yang mengalirkan listrik ke seluruh sistem

3.3. Activity Procedures

This project is executed through four systematic stages to ensure optimal results.

1. **Literature Study & Design Phase:** The initial step was to build a foundation of knowledge by reviewing various sources related to solar

energy and rural technology. After that, we translated our ideas into concrete designs, both for a mock-up of the house and for the hybrid system. The mock-up design aimed to replicate the conditions of a rural home that would be the "patient" of this technology.

2. **Implementation Stage on the Mockup:** At this stage, the design is translated into a physical prototype. The solar-powered automatic lighting circuit is meticulously assembled. In parallel, reflective mirrors are mounted on the mockup at calculated angles to maximize light reflection. These two systems are then integrated on the mockup, creating a unified lighting system.
3. **Testing Phase:** This was the moment to prove it. The prototype model was tested in two scenarios:

- Daytime Simulation: The model was placed in sunlight to measure how effectively the reflective mirrors illuminated the interior of the house.
- Nighttime Simulation: The automatic function of the lights was tested. We observed whether the solar panel successfully charged the battery and whether the light sensor could automatically turn on the light when it got dark.

3.4 Analysis Techniques

The data obtained from the testing was analyzed descriptively. This means we didn't just look at the numbers but also interpreted the observations. For example, we described the increased visual brightness in the mock-up thanks to the mirrors and the reliability of the automatic lighting system. This analysis served as the basis for concluding whether the system was feasible and had the potential for further development to help communities in remote areas.

CHAPTER IV
RESULTS AND DISCUSSION

4. 228160049 (Creating an ESP 8266 Program for Web Monitoring Connectivity)

Name : Andre Nugraha Wageardoyo NPM : 228160028 Study Program : Informatics Engineering Task : Creating an ESP 8266 Program for Web Monitoring Connectivity
ESP8266 Programmer for Web Monitoring Connectivity
#include <ESP8266WiFi.h> #include <ESP8266WebServer.h> #include <Servo.h> #include <FS.h> #include <LittleFS.h> // ===== KONFIGURASI ACCESS POINT ===== const char* ap_ssid = "Solar_Tracker"; const char* ap_password = "12345678"; // ===== SERVO MOTOR ===== Servo horizontal; Servo vertical; // ===== POSISI SERVO DAN TRACKING ===== int servohori = 90;

```
int servovert = 45;
int prevHori = 90;
int prevVert = 45;

// ===== DATA GERAKAN =====
String lastMovement = "Tidak ada gerakan";
unsigned long lastMoveTime = 0;
int totalMovements = 0;
String movementHistory[10]; // Simpan 10 gerakan terakhir
int historyIndex = 0;

// ===== SIMULASI TRACKING (untuk demo) =====
bool autoTracking = true;
unsigned long lastTrackingTime = 0;
int trackingDirection = 1;
float simulationSpeed = 0.5; // Kecepatan simulasi

// ===== VARIABEL MONITORING =====
unsigned long lastLogTime = 0;
unsigned long lastUpdateTime = 0;

ESP8266WebServer server(80);

// ===== FUNGSI UNTUK MENDETEKSI GERAKAN =====
void detectMovement() {
  if (servohori != prevHori || servovert != prevVert) {
    String movement = "";

    // Deteksi gerakan horizontal
    if (servohori != prevHori) {
      int horiDiff = servohori - prevHori;
      if (horiDiff > 0) {
```

```
      movement += "Kanan " + String(horiDiff) + "°";
    } else {
      movement += "Kiri " + String(abs(horiDiff)) + "°";
    }
  }

  // Deteksi gerakan vertikal
  if (servovert != prevVert) {
    int vertDiff = servovert - prevVert;
    if (movement.length() > 0) movement += ", ";

    if (vertDiff > 0) {
      movement += "Naik " + String(vertDiff) + "°";
    } else {
      movement += "Turun " + String(abs(vertDiff)) + "°";
    }
  }

  // Simpan gerakan
  lastMovement = movement;
  lastMoveTime = millis();
  totalMovements++;

  // Tambah ke history
    movementHistory[historyIndex] = String(millis()/1000) + "s: " +
movement;
  historyIndex = (historyIndex + 1) % 10;

  // Update posisi sebelumnya
  prevHori = servohori;
  prevVert = servovert;
```

```
    Serial.println("GERAKAN: " + movement);
  }
}

// ===== SIMULASI TRACKING MATAHARI =====
void simulateTracking() {
  if (!autoTracking) return;

  if (millis() - lastTrackingTime > 3000) { // Update setiap 3 detik
    lastTrackingTime = millis();

    // Simulasi gerakan mengikuti matahari
    int oldHori = servohori;
    int oldVert = servovert;

    // Gerakan horizontal (simulasi rotasi bumi)
    servohori += trackingDirection * random(1, 4);

    // Batasi gerakan
    if (servohori >= 170) {
      servohori = 170;
      trackingDirection = -1;
    } else if (servohori <= 10) {
      servohori = 10;
      trackingDirection = 1;
    }

    // Gerakan vertikal (simulasi elevasi matahari)
    if (servohori < 90) {
      // Pagi: naik
      servovert += random(0, 2);
    } else if (servohori > 90) {
```

```
      // Sore: turun
      servovert -= random(0, 2);
    } else {
      // Siang: sedikit variasi
      servovert += random(-1, 2);
    }

    // Batasi vertikal
    servovert = constrain(servovert, 1, 100);

    // Gerakkan servo fisik
    horizontal.write(servohori);
    vertical.write(servovert);

    // Deteksi gerakan
    detectMovement();
  }
}

// ===== FUNGSI ARAH =====
String getPanelDirection() {
  if (servohori < 45) return "Kiri Jauh";
  if (servohori < 75) return "Kiri";
  if (servohori < 105) return "Tengah";
  if (servohori < 135) return "Kanan";
  return "Kanan Jauh";
}

String getPanelHeight() {
  if (servovert < 25) return "Sangat Rendah";
  if (servovert < 45) return "Rendah";
  if (servovert < 65) return "Sedang";
```

```
  if (servovert < 85) return "Tinggi";
  return "Sangat Tinggi";
}

String getSunStatus() {
  if (servohori < 60) return "Matahari Pagi";
  if (servohori > 120) return "Matahari Sore";
  return "Matahari Siang";
}

// ===== SIMPAN DATA CSV =====
void saveToCSV() {
  String timestamp = String(millis() / 1000);
   String logEntry = timestamp + "," + String(servohori) + "," +
String(servovert) + "\n";

  File file = LittleFS.open("/data.csv", "a");
  if (file) {
    file.print(logEntry);
    file.close();
  }
}

void initCSV() {
  if (!LittleFS.exists("/data.csv")) {
    File file = LittleFS.open("/data.csv", "w");
    if (file) {
      file.println("Waktu,Horizontal,Vertikal");
      file.close();
    }
  }
}
```

```
void setup() {
  Serial.begin(115200);

  if (!LittleFS.begin()) {
    Serial.println("File system error");
    return;
  }

  initCSV();

  // ===== SERVO =====
  horizontal.attach(D1);
  vertical.attach(D2);
  horizontal.write(servohori);
  vertical.write(servovert);

  // ===== WIFI =====
  WiFi.mode(WIFI_AP);
  WiFi.softAP(ap_ssid, ap_password);

  // ===== WEB SERVER =====
  server.on("/", handleRoot);
  server.on("/data", handleData);
  server.on("/download", handleDownload);
  server.on("/clear", handleClear);
  server.on("/toggle", handleToggle);
  server.begin();

  Serial.println("Solar Tracker Started!");
  Serial.println("Connect to: " + String(ap_ssid));
  Serial.println("Open: http://" + WiFi.softAPIP().toString());
```

```
  // Inisialisasi history
  for (int i = 0; i < 10; i++) {
    movementHistory[i] = "";
  }

  saveToCSV();
}

void loop() {
  server.handleClient();

  // Simulasi tracking
  simulateTracking();

  // Auto save setiap 5 detik
  if (millis() - lastLogTime > 5000) {
    saveToCSV();
    lastLogTime = millis();
  }
}

void handleRoot() {
  String html = R"rawliteral(
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Panel Surya Real-Time</title>
        <meta name="viewport" content="width=device-width, initial-
scale=1">
    <style>
```

```
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    margin: 0;
    padding: 10px;
    background: #f8f9fa;
    color: #333;
    min-height: 100vh;
}

.container {
    max-width: 900px;
    margin: 0 auto;
    padding: 0 10px;
}

/* Card Styles */
.card {
    background: white;
    border-radius: 15px;
    padding: 25px;
    margin: 20px 0;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.08);
    border: 1px solid #e9ecef;
    transition: all 0.3s ease;
    position: relative;
}
```

```
.card::before {
    content: '';
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    height: 4px;
    background: #007bff;
    border-radius: 15px 15px 0 0;
}

.card:hover {
    transform: translateY(-3px);
    box-shadow: 0 8px 25px rgba(0, 123, 255, 0.15);
}

/* Header */
h1 {
    text-align: center;
    color: #007bff;
    font-size: 2.5em;
    font-weight: 700;
    margin-bottom: 30px;
}

/* Status Card */
.status-card {
    background: linear-gradient(135deg, #007bff, #0056b3);
    color: white;
    text-align: center;
}
```

```
.status-card::before {
    background: #007bff;
}

.status-card .big-text {
    font-size: 2.2em;
    font-weight: 700;
    margin: 15px 0;
}

.status-card .medium-text {
    font-size: 1.4em;
    margin: 8px 0;
    opacity: 0.9;
}

/* Position Card */
.position-card {
    background: white;
}

.status-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: 20px;
    margin: 25px 0;
}

.status-item {
    background: #f8f9fa;
    padding: 20px;
```

```
        border-radius: 12px;
        text-align: center;
        transition: all 0.3s ease;
        border: 2px solid #e9ecef;
    }

    .status-item:hover {
        background: #e3f2fd;
        border-color: #007bff;
        transform: scale(1.02);
    }

    .status-label {
        font-size: 1em;
        color: #6c757d;
        margin-bottom: 10px;
        font-weight: 500;
    }

    .status-value {
        font-size: 2em;
        font-weight: 700;
        color: #007bff;
    }

    /* Progress Bars */
    .progress {
        margin: 20px 0;
    }

    .progress-label {
        font-size: 1.1em;
```

```
        font-weight: 600;
        margin-bottom: 10px;
        color: #495057;
    }

    .progress-bar {
        width: 100%;
        height: 30px;
        background: #e9ecef;
        border-radius: 15px;
        overflow: hidden;
        position: relative;
        border: 1px solid #dee2e6;
    }

    .progress-fill {
        height: 100%;
        background: linear-gradient(90deg, #007bff, #0056b3);
        transition: width 0.5s ease;
        border-radius: 15px;
    }

    .progress-text {
        position: absolute;
        top: 50%;
        left: 50%;
        transform: translate(-50%, -50%);
        font-weight: 700;
        color: white;
        font-size: 1em;
    }
```

```
/* Movement Card */
.movement-card {
    background: #e3f2fd;
    border-left: 4px solid #007bff;
}

.movement-card::before {
    background: #007bff;
}

.movement-text {
    font-size: 1.3em;
    font-weight: 600;
    margin: 15px 0;
    color: #0056b3;
}

.movement-time {
    font-size: 1em;
    color: #6c757d;
}

.stats-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(130px, 1fr));
    gap: 15px;
    margin: 20px 0;
}

.stat-item {
    background: white;
    padding: 15px;
```

```
        border-radius: 12px;
        text-align: center;
        transition: all 0.3s ease;
        border: 2px solid #e9ecef;
    }

    .stat-item:hover {
        background: #f8f9fa;
        border-color: #007bff;
        transform: translateY(-2px);
    }

    .stat-value {
        font-size: 1.4em;
        font-weight: 700;
        color: #007bff;
    }

    .stat-label {
        font-size: 0.9em;
        color: #6c757d;
        margin-top: 5px;
    }

    /* History Card */
    .history-item {
        background: #f8f9fa;
        padding: 12px 15px;
        border-radius: 10px;
        margin: 8px 0;
        font-size: 0.95em;
        color: #495057;
```

```
    border-left: 4px solid #007bff;
    transition: all 0.3s ease;
}

.history-item:hover {
    background: #e3f2fd;
    transform: translateX(5px);
}

/* Buttons */
.button {
    background: #007bff;
    color: white;
    border: none;
    padding: 12px 20px;
    border-radius: 8px;
    font-size: 1em;
    font-weight: 600;
    cursor: pointer;
    margin: 8px;
    transition: all 0.3s ease;
    box-shadow: 0 3px 10px rgba(0, 123, 255, 0.3);
}

.button:hover {
    background: #0056b3;
    transform: translateY(-2px);
    box-shadow: 0 5px 15px rgba(0, 123, 255, 0.4);
}

.button.danger {
    background: #dc3545;
```

```
        box-shadow: 0 3px 10px rgba(220, 53, 69, 0.3);
    }

    .button.danger:hover {
        background: #c82333;
        box-shadow: 0 5px 15px rgba(220, 53, 69, 0.4);
    }

    .button.success {
        background: #28a745;
        box-shadow: 0 3px 10px rgba(40, 167, 69, 0.3);
    }

    .button.success:hover {
        background: #1e7e34;
        box-shadow: 0 5px 15px rgba(40, 167, 69, 0.4);
    }

    .center {
        text-align: center;
    }

    /* Live Indicator */
    .live-indicator {
        position: fixed;
        top: 20px;
        right: 20px;
        background: #007bff;
        color: white;
        padding: 10px 15px;
        border-radius: 20px;
        font-size: 0.9em;
```

```
        font-weight: 600;
        z-index: 1000;
        box-shadow: 0 4px 15px rgba(0, 123, 255, 0.3);
    }

    .live-indicator::before {
        content: "●";
        margin-right: 8px;
        animation: blink 1s infinite;
    }

    @keyframes blink {
        0%, 50% { opacity: 1; }
        51%, 100% { opacity: 0.3; }
    }

    /* Responsive Design */
    @media (max-width: 768px) {
        .container {
            padding: 0 5px;
        }

        .card {
            padding: 20px;
            margin: 15px 0;
            border-radius: 15px;
        }

        h1 {
            font-size: 2em;
            margin-bottom: 20px;
        }
```

```
.status-grid {
    grid-template-columns: 1fr;
    gap: 15px;
}

.stats-grid {
    grid-template-columns: repeat(2, 1fr);
    gap: 10px;
}

.status-card .big-text {
    font-size: 1.8em;
}

.status-card .medium-text {
    font-size: 1.2em;
}

.status-value {
    font-size: 1.6em;
}

.button {
    padding: 10px 15px;
    font-size: 0.9em;
    margin: 5px;
}

.live-indicator {
    top: 10px;
    right: 10px;
```

```
        padding: 8px 12px;
        font-size: 0.8em;
    }
}

@media (max-width: 480px) {
    body {
        padding: 5px;
    }

    .card {
        padding: 15px;
        margin: 10px 0;
    }

    h1 {
        font-size: 1.6em;
    }

    .status-card .big-text {
        font-size: 1.5em;
    }

    .status-card .medium-text {
        font-size: 1.1em;
    }

    .status-value {
        font-size: 1.4em;
    }

    .stats-grid {
```

```
            grid-template-columns: 1fr;
        }

        .button {
            width: 100%;
            margin: 5px 0;
        }
    }

    @media (min-width: 1200px) {
        .container {
            max-width: 1100px;
        }

        .stats-grid {
            grid-template-columns: repeat(3, 1fr);
        }
    }
  </style>
</head>
<body>
  <div class="live-indicator">LIVE</div>

  <div class="container">
    <h1>☼ Panel Surya Real-Time</h1>

    <div class="card status-card">
      <div class="big-text" id="sunStatus">Matahari Siang</div>
      <div class="medium-text" id="panelDirection">Tengah</div>
      <div class="medium-text" id="panelHeight">Sedang</div>
    </div>
```

```
<div class="card position-card">
  <div class="status-grid">
    <div class="status-item">
      <div class="status-label">Posisi Kiri-Kanan</div>
      <div class="status-value" id="horiValue">90°</div>
    </div>
    <div class="status-item">
      <div class="status-label">Posisi Atas-Bawah</div>
      <div class="status-value" id="vertValue">45°</div>
    </div>
  </div>

  <div class="progress">
    <div class="progress-label">Gerakan Kiri ↔ Kanan</div>
    <div class="progress-bar">
      <div class="progress-fill" id="horiBar"></div>
      <div class="progress-text" id="horiText">50%</div>
    </div>
  </div>

  <div class="progress">
    <div class="progress-label">Gerakan Atas ↕ Bawah</div>
    <div class="progress-bar">
      <div class="progress-fill" id="vertBar"></div>
      <div class="progress-text" id="vertText">45%</div>
    </div>
  </div>
</div>

<div class="card movement-card">
  <h3>🎯 Gerakan Terakhir</h3>
      <div  class="movement-text"  id="lastMovement">Tidak  ada
```

```
gerakan</div>
        <div class="movement-time" id="lastMoveTime">-</div>

        <div class="stats-grid">
          <div class="stat-item">
            <div class="stat-value" id="totalMovements">0</div>
            <div class="stat-label">Total Gerakan</div>
          </div>
          <div class="stat-item">
            <div class="stat-value" id="trackingStatus">ON</div>
            <div class="stat-label">Auto Tracking</div>
          </div>
          <div class="stat-item">
            <div class="stat-value" id="updateTime">0s</div>
            <div class="stat-label">Update Terakhir</div>
          </div>
        </div>
      </div>

      <div class="card">
        <h3>📜 Riwayat Gerakan</h3>
        <div id="movementHistory">
          <div class="history-item">Memuat riwayat...</div>
        </div>
      </div>

      <div class="card center">
        <h3>🎮 Kontrol & Data</h3>
          <button class="button success" onclick="toggleTracking()">🔄
Toggle Auto</button>
        <button class="button" onclick="downloadData()">📥 Download
```

```
Data</button>
            <button class="button danger" onclick="clearData()">🗑️ Hapus
Data</button>
        </div>
    </div>

    <script>
        function updateData() {
            fetch('/data')
                .then(response => response.json())
                .then(data => {
                    // Update positions
                            document.getElementById('horiValue').textContent =
data.horizontal + '°';
                            document.getElementById('vertValue').textContent =
data.vertical + '°';

                    // Update progress bars
                    const horiPercent = ((data.horizontal - 5) / 170) * 100;
                    const vertPercent = ((data.vertical - 1) / 99) * 100;

                              document.getElementById('horiBar').style.width =
Math.max(0, Math.min(100, horiPercent)) + '%';
                               document.getElementById('vertBar').style.width =
Math.max(0, Math.min(100, vertPercent)) + '%';

                            document.getElementById('horiText').textContent =
Math.round(horiPercent) + '%';
                            document.getElementById('vertText').textContent =
Math.round(vertPercent) + '%';

                    // Update status
```

```
                document.getElementById('sunStatus').textContent =
data.sunStatus;
                document.getElementById('panelDirection').textContent =
data.direction;
                document.getElementById('panelHeight').textContent =
data.height;

            // Update movement info
                document.getElementById('lastMovement').textContent =
data.lastMovement;
                document.getElementById('lastMoveTime').textContent =
data.lastMoveTime;
                document.getElementById('totalMovements').textContent =
data.totalMovements;
                document.getElementById('trackingStatus').textContent =
data.autoTracking ? 'ON' : 'OFF';
                document.getElementById('updateTime').textContent =
Math.floor(Date.now() / 1000) + 's';

            // Update history
                const historyDiv =
document.getElementById('movementHistory');
            historyDiv.innerHTML = '';
            data.history.forEach(item => {
              if (item && item.trim() !== '') {
                const div = document.createElement('div');
                div.className = 'history-item';
                div.textContent = item;
                historyDiv.appendChild(div);
              }
            });
```

```
            if (historyDiv.children.length === 0) {
                const div = document.createElement('div');
                div.className = 'history-item';
                div.textContent = 'Belum ada gerakan';
                historyDiv.appendChild(div);
            }
        })
        .catch(error => {
            console.error('Error:', error);
        });
}

function toggleTracking() {
    fetch('/toggle')
        .then(response => response.text())
        .then(data => {
            console.log(data);
        });
}

function downloadData() {
    window.location.href = '/download';
}

function clearData() {
    if (confirm('Hapus semua data?')) {
        fetch('/clear')
            .then(response => response.text())
            .then(data => {
                console.log(data);
            });
    }
```

```
        }

        // Auto update every 1 second
        updateData();
        setInterval(updateData, 1000);
    </script>
</body>
</html>
)rawliteral";

  server.send(200, "text/html", html);
}

void handleData() {
  // Buat array history untuk JSON
  String historyJson = "[";
  for (int i = 0; i < 10; i++) {
    if (movementHistory[i].length() > 0) {
      if (historyJson.length() > 1) historyJson += ",";
      historyJson += "\"" + movementHistory[i] + "\"";
    }
  }
  historyJson += "]";

  String timeAgo = "";
  if (lastMoveTime > 0) {
    int secondsAgo = (millis() - lastMoveTime) / 1000;
    timeAgo = String(secondsAgo) + " detik yang lalu";
  } else {
    timeAgo = "Belum ada gerakan";
  }
```

```
  String json = "{";
  json += "\"horizontal\":" + String(servohori) + ",";
  json += "\"vertical\":" + String(servovert) + ",";
  json += "\"direction\":\"" + getPanelDirection() + "\",";
  json += "\"height\":\"" + getPanelHeight() + "\",";
  json += "\"sunStatus\":\"" + getSunStatus() + "\",";
  json += "\"lastMovement\":\"" + lastMovement + "\",";
  json += "\"lastMoveTime\":\"" + timeAgo + "\",";
  json += "\"totalMovements\":" + String(totalMovements) + ",";
  json += "\"autoTracking\":" + String(autoTracking ? "true" : "false") +
",";
  json += "\"history\":" + historyJson;
  json += "}";

  server.send(200, "application/json", json);
}

void handleToggle() {
  autoTracking = !autoTracking;
  String response = autoTracking ? "Auto tracking ON" : "Auto tracking
OFF";
  server.send(200, "text/plain", response);
}

void handleDownload() {
  File file = LittleFS.open("/data.csv", "r");
  if (!file) {
    server.send(404, "text/plain", "File tidak ada");
    return;
  }

  server.setContentLength(file.size());
```


Document Accepted 17/10/25

```
  server.send(200, "text/csv", "");

  WiFiClient client = server.client();
  while (file.available()) {
   client.write(file.read());
  }
  file.close();
}

void handleClear() {
  LittleFS.remove("/data.csv");
  initCSV();

  // Reset movement data
  totalMovements = 0;
  lastMovement = "Tidak ada gerakan";
  lastMoveTime = 0;
  for (int i = 0; i < 10; i++) {
   movementHistory[i] = "";
  }
  historyIndex = 0;

  server.send(200, "text/plain", "Data dan riwayat gerakan dihapus");
}
```

ESP8266 Program Code Explanation

A solar tracker system is a technology that optimizes solar energy absorption by moving solar panels so they always face the light source. This project aims to create a prototype solar tracker system using an ESP8266 microcontroller. This system not only controls the movement of two servo motors to simulate the movement of the sun, but also provides an interactive web interface (dashboard). This dashboard allows real-time monitoring and remote control via a local Wi-Fi network, and serves as a database for storing panel movement history in a CSV file.

1. Hardware Description

This project is designed with a minimalist yet effective hardware configuration.

ESP8266 Microcontroller: The ESP8266 was chosen as the main component due to its affordability and built-in Wi-Fi module. This allows the device to function as a standalone web server without the need for additional devices.

Two Servo Motors:

Horizontal Servo (Pin D1): Responsible for the lateral movement of the panel, simulating the sun's movement from east to west. The range of movement is limited to 10° to 170° to maintain the panel's mechanics.

Vertical Servo (Pin D2): Responsible for the panel's elevation movement, simulating the rising and falling of the sun throughout the day. Its range of movement is limited to 1° to 100°.

Power Supply: To ensure the servo motor can move stably, an adequate external

power supply is required. The ESP8266 microcontroller is also powered by the same source.

2. Software Architecture

The software is built with a modular architecture, with each part having a specific task:

a. Network Configuration (Access Point)

This program configures the ESP8266 as an Access Point (AP) with the name "Solar_Tracker" and the password "12345678". This creates a local Wi-Fi network that does not require an internet connection, making it a standalone and portable solution. Other devices, such as mobile phones or laptops, can connect directly to this network to access the dashboard.

b. Tracking System (Simulation)

The simulateTracking() function is the core of the movement logic. Instead of using a light sensor, the program uses a time-based algorithm that changes the servo position periodically (every 3 seconds). This movement is designed to mimic the actual path of the sun:

The horizontal position will gradually increase in the morning and decrease in the afternoon.

The vertical position will adjust, increasing in the morning and decreasing in the

afternoon.

This logic provides a clear picture of how the solar tracking system can operate automatically.

c. Interactive Web Dashboard

The web interface is built using HTML, CSS, and JavaScript.

HTML: Serves as the basic structure of the page, containing elements such as the title, status card, progress bar, and control buttons.

CSS: Used to design the page to make it look modern and user-friendly. The design is responsive, so it displays well on various screen sizes, from mobile phones to desktops.

JavaScript: Plays a crucial role in making the dashboard dynamic. This script periodically retrieves data from the ESP8266 via the /data endpoint and updates all elements on the web page in real-time without requiring a page reload.

d. Data Management (LittleFS)

This program utilizes the LittleFS file system to persistently store movement data.

The initCSV() function will create a data.csv file with headers (Time, Horizontal, Vertical) if one doesn't already exist.

The saveToCSV() function will append a new row every 5 seconds, recording the current servo position.

Users can download this file via the /download endpoint. This endpoint will send

the entire CSV file as an HTTP response, which can then be saved by the user's web browser.

There is also a handleClear() function to clear all log data, which is useful for starting a new log.

e. Control and Feedback

This program provides comprehensive feedback to the user:

Last Movement: The detectMovement() function records and visualizes each change in servo position. This information, such as "Right 5°," is displayed on the dashboard along with the time it occurred.

Movement History: The program stores the last 10 movements in an array (movementHistory) and displays it on a web page, providing a historical overview of the panel's movements.

Automatic Status: The "Toggle Auto" button allows users to switch between automatic and idle tracking modes. The status (ON or OFF) is also displayed on the dashboard.

3. Conclusion and Potential for Development

Overall, this project has successfully created a solar tracking system that is not only mechanically functional but also sophisticated in terms of monitoring and control. The combination of simple hardware and software

CHAPTER V
CLOSING

5.1. Conclusion

Based on this treatment, it can be concluded that the Hybrid Project: Design and Development of a Sunlight-Based Lighting System Using Reflective Mirrors, an Automatic Solar-Powered Lamp for Rural Lighting is an innovative and sustainable solution in addressing the challenge of limited access to electrical energy in rural areas. By combining two solar-based lighting systems—namely the use of reflective mirrors for natural lighting during the day and automatic solar-powered lamps for the night—this system is able to provide efficient, environmentally friendly, and cost-effective lighting.

5.2 Suggestions

To ensure this hybrid lighting system can be implemented more widely and sustainably, it is recommended that the system's design and efficiency be further developed, particularly in terms of the use of reflective light and the energy storage capacity of the solar panels. Adaptations to local geographic and socio-cultural conditions also need to be considered to ensure the system functions optimally according to the needs of each region.

BIBLIOGRAPHY

Anglistia, S. (2019). Prototype Sistem Pelacakan Sinar Matahari Pada Sistem Pembangkit Listrik Tenaga Surya Berbasis Arduino. Universitas Muhammadiyah Surakarta.

Birhane, E. M. (2019). Modeling and Analysis of Photo-Voltaic Solar Panel under Constant Electric Load. J. Renew. Energy.

Birhane, E. M. (2019). Modeling and Analysis of Photo-Voltaic Solar Panel under Constant Electric Load. J. Renew. Energy, 1-9.

Byregowda K. C, R. V. (2020). Study and Analysis of Arduino Based Solar Tracking Panel. International Research Journal of Engineering and Technology, 7.

Fauzi, K. W. (2018). Fauzi, Kodrat Wirawan, TePerancangan dan Realisasi Solar Tracking System Untuk Peningkatan Efisiensi Panel Surya Menggunakan Arduino Uno. TELKATelekomunikasi, Elektronika, Komputasi dan Kontrol, 63-74.

Hermita, R. (2024). Perancangan Pembuatan Maket Ruang Guru Pada Sekolah Budi Agung Medan. Teknologias, 2-6. L.

Putriyana, A. F. (2021). Simulasi Numerik Lapangan Panas Bumi Sirung Pantar, Kabupaten Alor, Nusa Tenggara Timur. Ketenagalistrikan dan Energi Terbarukan, 1, 1-12.

Pratik Pawar, A. Y. (2018). Solar Tracking System Using Arduino. International Journal of Scientific & Engineering Research, 8(9).