REAL TIME SYSTEM AND INTERNET OF THINGS FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA

GREEN BUILDING COMFORT

GROUP 13

AZRIEL DIMAS ASH-SHIDIQI	**2206059414**
NAKITA RAHMA DINANTI	**2206059401**
MUHAMMAD FAUZAN	**2206819054**
NAHL SYAREZA RAHIDRA	**2206830340**

PREFACE

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga kami dapat menyelesaikan proyek akhir yang berjudul "Green Building Comfort" ini dengan baik. Laporan ini disusun untuk memenuhi salah satu syarat dalam pemenuhan persyaratan tugas mata kuliah *Real Time System and Internet of Things* pada program studi Teknik Komputer, Fakultas Teknik Universitas Indonesia.

Dalam era modern ini, efisiensi energi dan kenyamanan di tempat publik menjadi perhatian utama dalam pengembangan teknologi berbasis *Internet of Things* (IoT). Salah satu inovasi yang sangat relevan adalah sistem otomatisasi untuk mengelola kenyamanan dan efisiensi energi. Proyek akhir ini bertujuan untuk merancang dan mengimplementasikan sebuah alat berbasis IoT yang dapat meningkatkan kenyamanan di tempat publik sekaligus menghemat energi listrik.

Sistem *Green Building Comfort* yang kami rancang memiliki kemampuan untuk secara otomatis mengatur kapan lampu di tempat publik harus dinyalakan atau dimatikan sesuai dengan tingkat penerangan yang dibutuhkan. Selain itu, sistem ini dapat mendeteksi kondisi keramaian di dalam ruangan. Jika suasana ruangan sangat ramai, alat ini akan secara otomatis mengaktifkan pendingin untuk menjaga kenyamanan pengunjung. Dengan demikian, sistem ini tidak hanya meningkatkan kenyamanan pengguna, tetapi juga membantu mengurangi konsumsi energi yang tidak diperlukan.

Kami menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang membangun sangat kami harapkan. Semoga laporan ini dapat memberikan manfaat dan kontribusi positif dalam pengembangan teknologi IoT untuk mendukung efisiensi energi dan kenyamanan di masa mendatang.

Depok, December 12, 2024

Group 13

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT

Di era modern ini, efisiensi energi dan kenyamanan di tempat publik menjadi salah satu aspek penting yang terus dikembangkan seiring dengan kemajuan teknologi. Tempat publik, seperti gedung perkantoran, pusat perbelanjaan, atau fasilitas umum lainnya, sering kali mengalami masalah dengan pengelolaan konsumsi energi yang kurang efisien, seperti penggunaan lampu dan pendingin ruangan yang tidak terkendali. Hal ini tidak hanya meningkatkan biaya operasional, tetapi juga memberikan dampak negatif terhadap lingkungan.

Selain itu, kenyamanan pengunjung menjadi prioritas utama dalam pengelolaan tempat publik. Ketika ruangan terlalu ramai, suasana menjadi kurang nyaman akibat panas dan sirkulasi udara yang buruk. Di sisi lain, pengaturan yang manual sering kali kurang responsif terhadap kondisi yang berubah dengan cepat, sehingga menurunkan kualitas pengalaman pengguna di tempat tersebut.

Salah satu solusi yang dapat diterapkan untuk menangani masalah ini adalah dengan memanfaatkan teknologi berbasis *Internet of Things* (IoT). Sistem otomatisasi berbasis IoT memungkinkan pengelolaan energi yang lebih efisien sekaligus memastikan kenyamanan pengunjung.

Dalam proyek ini, kami merancang sebuah sistem bernama *Green Building Comfort* dengan menggunakan berbagai komponen seperti ESP32, sensor cahaya (photoresistor), sensor IR, LED, motor DC, dan *motor driver* L298N. Sistem ini dirancang untuk mengatur pencahayaan secara otomatis berdasarkan tingkat penerangan yang dibutuhkan, serta mengaktifkan pendingin ruangan jika deteksi keramaian mencapai batas tertentu. Dengan menggunakan alat ini, kami berharap dapat menciptakan solusi yang tidak hanya hemat energi tetapi juga memberikan kenyamanan optimal di tempat publik.

1.2 PROPOSED SOLUTION

Proyek ini menggunakan berbagai komponen berbasis Internet of Things (IoT) untuk menciptakan sistem otomatisasi yang efisien dalam mengelola kenyamanan di tempat publik. Sistem ini memanfaatkan ESP32 sebagai unit pengendali utama yang terhubung dengan berbagai sensor dan aktuator untuk mengatur pencahayaan dan pendingin ruangan secara otomatis. Untuk mengatur pencahayaan, digunakan sensor cahaya (photoresistor) yang mendeteksi tingkat penerangan di ruangan. Berdasarkan data dari sensor ini, ESP32 akan mengontrol LED melalui breadboard dan menghidupkan atau mematikan lampu sesuai kebutuhan. Hal ini membantu menghemat energi dengan memastikan lampu hanya menyala saat dibutuhkan.

Selain itu, sistem ini juga dilengkapi dengan dua sensor IR untuk mendeteksi keberadaan dan tingkat keramaian di ruangan. Ketika sensor IR mendeteksi suasana ruangan yang ramai, ESP32 akan mengirimkan sinyal ke *motor driver* L298N untuk mengaktifkan motor DC yang menggerakkan pendingin ruangan. Dengan cara ini, alat dapat menjaga suhu ruangan tetap nyaman bagi pengunjung. Seluruh komponen terhubung melalui breadboard dan didukung oleh sumber daya dari baterai. Dengan menggunakan sistem otomatisasi ini, *Green Building Comfort* tidak hanya mampu menghemat energi tetapi juga memberikan kenyamanan optimal di tempat publik.

1.3 ACCEPTANCE CRITERIA

Tujuan dari proyek ini adalah:

1. Mengoptimalkan penggunaan listrik dengan mengatur pencahayaan secara otomatis sesuai kebutuhan dan hanya menyalakan lampu saat tingkat penerangan di ruangan kurang memadai.
2. Menjaga suhu ruangan tetap nyaman bagi pengunjung dengan mendeteksi tingkat keramaian dan mengaktifkan pendingin secara otomatis ketika diperlukan.
3. Menciptakan sistem otomatis berbasis IoT yang dapat berfungsi tanpa pengawasan manual, memastikan respons yang cepat dan akurat terhadap perubahan kondisi lingkungan.

1.4 ROLES AND RESPONSIBILITIES

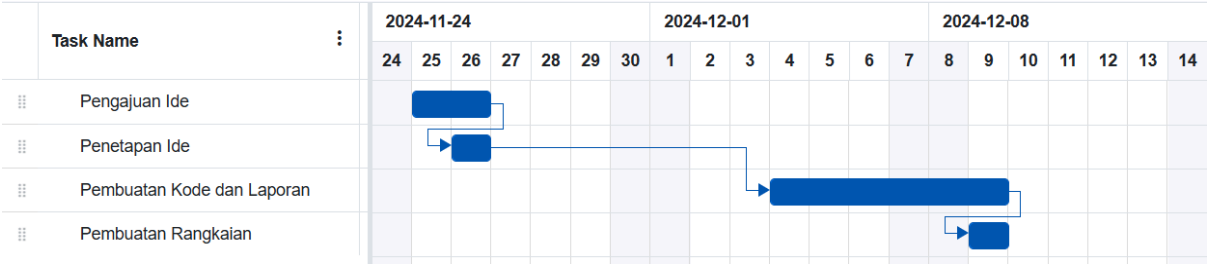
Berikut adalah peran dan tanggung jawab yang diberikan untuk tiap anggota:

Roles	Responsibilities	Person
Role 1	Membuat program dan merangkai rangkaian asli	Azriel Dimas Ash-Shidiqi
Role 2	Membuat program dan merangkai rangkaian asli	Nahl Syahreza Rahindra
Role 3	Membeli kebutuhan dan merangkai rangkaian asli	Nakita Rahma Dinanti
Role 4	Membeli kebutuhan dan merangkai rangkaian asli	Muhammad Fauzan

Table 1. Roles and Responsibilities

1.5 TIMELINE AND MILESTONES

Gantt Chart untuk timeline pengerjaan proyek akhir:



CHAPTER 2

IMPLEMENTATION

2.1 HARDWARE DESIGN AND SCHEMATIC

Untuk mengimplementasikan ide kami, terdapat beberapa komponen hardware yang akan digunakan, di antaranya yaitu:

- ESP32
- Breadboard
- LED
- DC Motor
- IR Sensor
- L298N Motor Driver
- Photoresistor
- Power Supply
- Kabel Jumper

Komponen dapat dibagi menjadi dua fungsi, yaitu fungsi input dan output. Komponen yang memiliki fungsi input adalah Photoresistor dan IR Sensor, sedangkan fungsi output terdapat pada LED, L298N Motor Driver, dan DC Motor. Terdapat komponen-komponen pendukung lainnya seperti Breadboard, Kabel Jumper, dan Power Supply. ESP32 akan bertindak sebagai pengendali dan media penghubung antara komponen input dan output.

Sensor akan memberikan data kepada ESP32 untuk dapat menentukan fungsionalitas dari alat kami. Photoresistor digunakan untuk mengatur tingkat keterangan dari LED dan IR Sensor digunakan untuk mengatur kekuatan dari DC Motor.

Desain dari hardware yang akan digunakan ialah menghubungkan Photoresistor kepada pin Analog, IR Sensor kepada pin Digital, LED kepada pin Analog, dan L298N kepada pin Digital. DC Motor akan dikendalikan oleh ESP32 melalui L298N Motor Driver.

Berikut adalah implementasi skematik dari desain hardware

2.2 SOFTWARE DEVELOPMENT

Ide kami bertujuan untuk mendukung penggunaan energi yang efisien dan otomatis kepada tempat-tempat publik. Alat kami akan menentukan output secara otomatis berdasarkan input-input yang diberikan oleh sensor yang ada pada alat kami. Maka dari itu diperlukan software berupa program yang dapat mengendalikan fungsionalitas dari tiap-tiap komponen. Program akan digunakan untuk menghubungi komponen input dengan komponen output, sehingga output yang dihasilkan dapat menyesuaikan dengan data yang diberikan oleh input pada saat itu. Terdapat sebuah algoritma yang dapat menentukan output yang sesuai berdasarkan input yang diberikan dari sensor.

Dalam merancang program tersebut, kami menggunakan library, fugnsi, ataupun software yang telah disediakan untuk memperluas kerja ESP32, misalnya

- FreeRTOS
- PWM
- Blynk

Terdapat sebuah program yang memiliki dua fungsi kerja, yaitu pengaturan untuk Photoresistor dan LED, kemudian IR Sensor dengan L298N Motor Sensor dan juga DC Motor. Kedua program ini kemudian akan diintegrasikan untuk dijalankan pada satu buah ESP32.

Program Photoresistor dan LED akan menerima input berupa bacaan cahaya. Input ini kemudian akan diproses dalam program melalui sebuah proses mapping. Mapping ini dilakukan untuk menyesuaikan antara input yang didapatkan dan output yang ingin diberikan. Konfigurasi mapping ini akan disesuaikan dengan kondisi di lapangan nantinya. Berikut adalah potongan kode dan flowchart dari program:

```
// Pin Definitions
#define LAMP1 4
#define LAMP2 5
#define LAMP3 18
#define LDR_PIN 34  // Pin analog untuk LDR

// Global Variables
```

```
bool automationMode = true;  // Mode otomatis


// Function to control lighting based on LDR
void controlLighting() {
  int ldrValue = analogRead(LDR_PIN); // Membaca nilai LDR (0-4095 pada ESP32)


  if (automationMode) {
    // Jika ruangan sangat terang, matikan semua lampu
    if (ldrValue < 100) { // Ambang batas terang
      analogWrite(LAMP1, 0); // Lampu 1 mati
      analogWrite(LAMP2, 0); // Lampu 2 mati
      analogWrite(LAMP3, 0); // Lampu 3 mati
      Serial.println("Lampu mati karena ruangan sangat terang.");
    }
    else {
      // Jika ruangan redup, sesuaikan kecerahan lampu
       int brightness = map(ldrValue, 0, 4095, 0, 255); // Inversi nilai LDR untuk brightness
      analogWrite(LAMP1, brightness); // Lampu 1
      analogWrite(LAMP2, brightness); // Lampu 2
      analogWrite(LAMP3, brightness); // Lampu 3
      Serial.println("Brightness lampu: " + String(brightness));
    }
  }
}


// FreeRTOS Task for Lighting
void TaskLighting(void *pvParameters) {
  while (1) {
    controlLighting();
    vTaskDelay(1000 / portTICK_PERIOD_MS);
  }
```

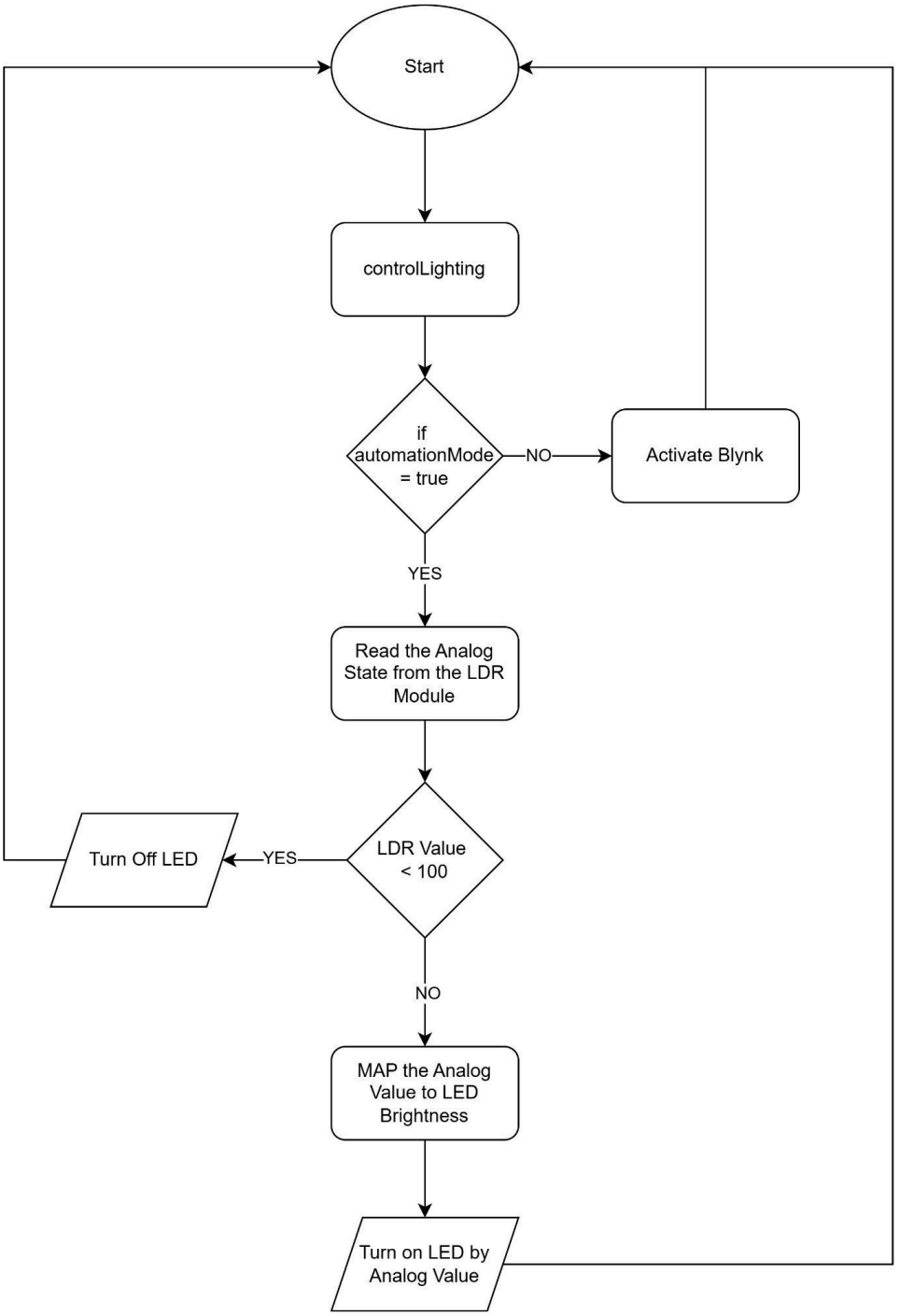
```
}

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

  // Pin Modes
  pinMode(LAMP1, OUTPUT);
  pinMode(LAMP2, OUTPUT);
  pinMode(LAMP3, OUTPUT);

  // FreeRTOS Task Creation
  xTaskCreate(TaskLighting, "Lighting Task", 1000, NULL, 1, NULL);
}

void loop() {
  // Empty loop; tasks are handled by FreeRTOS
}
```



Program IR Sensor dan L298N Motor Driver sekaligus DC Motor dirancang untuk dapat menyesuaikan kekuatan DC Motor dengan input yang diberikan oleh IR Sensor. Terdapat parameter-parameter yang akan ditentukan oleh program untuk dapat menentukan kecepatan yang akan ditetapkan pada DC Motor. Parameter ini berupa jumlah orang yang masuk kepada satu ruangan. Setiap orang yang memasuki ruangan akan menambahkan variabel kontrol yang kemudian akan dimasukkan pada perhitungan untuk menentukan kekuatan rotasi DC Motor. Jika terdapat orang yang meninggalkan ruangan, variabel kontrol

ini akan berkurang dan DC Motor pun akan menyesuaikan dengan nilai yang baru tersebut. Berikut adalah potongan kode dan flowchart dari program:

```
#define BLYNK_PRINT Serial

#define BLYNK_TEMPLATE_ID "TMPL6fSuUMh-m"
#define BLYNK_TEMPLATE_NAME "Zang Zing"
#define BLYNK_AUTH_TOKEN "iccuJhsXcjRIX20sFM75onKmGgwY_R_P"

#define IR_DC_MODE_VPIN V10
#define IR_DC_VALUE_VPIN V12

#include <Arduino.h>
#include <FreeRTOSConfig.h>
// #include <ESP32Servo.h>
// #include <BlynkSimpleEsp32.h>

#define INPUT_ENTER 23
#define INPUT_EXIT 22

#define IN1 26
#define IN2 25
#define EN1 27

#define PWM_CHN 0     // Channel 0
#define LEDC_TIMER 0  // Timer 0
#define PWM_RES 8     // 8-bit resolution (0-255)
#define PWM_FREQ 1000 // Frequency in Hz (5 kHz)

BaseType_t xSetPowerTask;

int people = 0;
bool enter_changed = false;
```

```
bool exit_changed = false;

const char *ssid = "Wokwi-GUEST";
const char *password = "";

static int IR_DC_MODE = 0;
static int IR_DC_VALUE = 0;

// BLYNK_WRITE(IR_DC_MODE_VPIN)
// {
//   IR_DC_MODE = param.asInt();
//   Serial.println("Vrombop");
// }

// BLYNK_WRITE(IR_DC_VALUE_VPIN)
// {
//   IR_DC_VALUE = param.asInt();
//   Serial.println("Zang zing");
// }

void vSetPowerTask(void *pvParameters)
{
  char buff[255];
  while (1)
  {
    if (digitalRead(INPUT_ENTER) == HIGH)
    {
      if (!enter_changed)
      {
        people = people + 1 > 5 ? 5 : people += 1;
        enter_changed = true;
      }
```

```
  }
  else
  {
    enter_changed = false;
  }

  if (digitalRead(INPUT_EXIT) == HIGH)
  {
    if (!exit_changed)
    {
      people = people - 1 < 0 ? 0 : people -= 1;
      exit_changed = true;
    }
  }
  else
  {
    exit_changed = false;
  }

  float formula = (((float)people / 5) * 200) + 50;

  formula = !IR_DC_MODE ? formula : IR_DC_VALUE;

  ledcWrite(PWM_CHN, formula);

  if (people < 5)
  {
    sprintf(buff, "People is %d (Power: %.2f)", people, formula);
    Serial.println(buff);
  }
  else
  {
```

```
       sprintf(buff, "People is more than equal to %d (Power: %.2f)",
people, formula);
      Serial.println(buff);
    }

    vTaskDelay(1000 / portTICK_PERIOD_MS);
  }
}

// put function declarations here:
void setup()
{
  // put your setup code here, to run once:
  Serial.begin(115200);

  // Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

  // Blynk.syncVirtual(IR_DC_MODE_VPIN);
  // Blynk.syncVirtual(IR_DC_VALUE_VPIN);

   xSetPowerTask = xTaskCreate(vSetPowerTask, "Set Power", 8192, NULL,
0, NULL);

  pinMode(INPUT_ENTER, INPUT);
  pinMode(INPUT_EXIT, INPUT);

  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  // pinMode(LEDC_PIN, OUTPUT);

  if (!ledcSetup(PWM_CHN, PWM_FREQ, PWM_RES))
  {
    Serial.println("LEDC setup failed!");
```

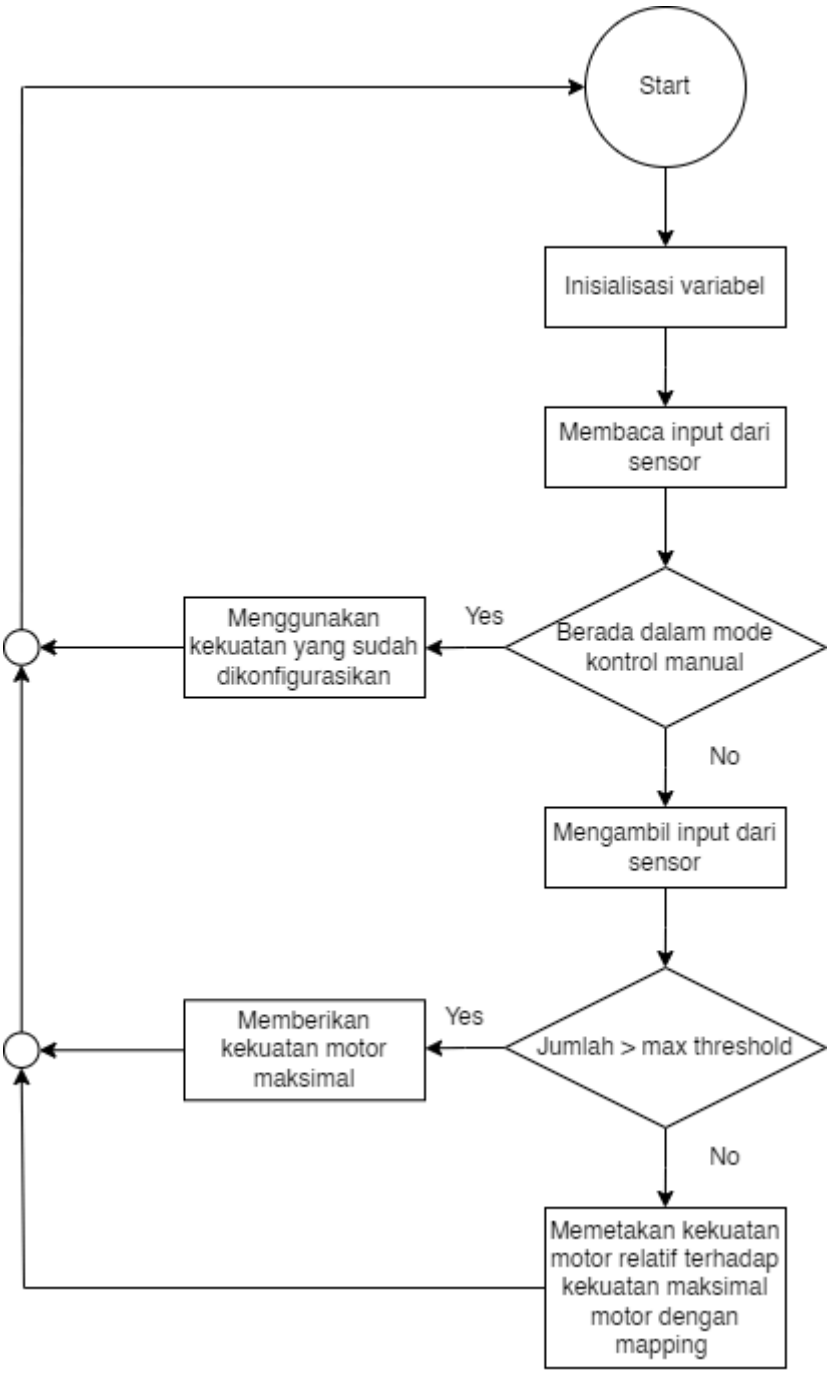
```
    while (1)
      ; // Halt if setup fails
  }

  ledcAttachPin(EN1, PWM_CHN);

  ledcWrite(PWM_CHN, 180);
}

void loop()
{
  // put your main code here, to run repeatedly:
  // Blynk.run();

  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
}
```



Program kami dapat dikontrol menggunakan Blynk. Blynk menyediakan fitur untuk mengirimkan data dari laptop ataupun smartphone kepada ESP32. Kita dapat mengatur alat secara manual dengan memasukkan nilai spesifik yang ingin diberikan kepada alat.

2.3 HARDWARE AND SOFTWARE INTEGRATION

Untuk integrasi hardware dan software, kita pertama-tama perlu membuat rangkaian asli dengan ketentuan yang ada. Kita harus menentukan pinout-pinout yang sesuai dengan kode yang diberikan. Berikut adalah kode integrasi dari kedua potongan kode:

```
#define BLYNK_PRINT Serial

#define BLYNK_TEMPLATE_ID "TMPL6fSuUMh-m"
#define BLYNK_TEMPLATE_NAME "Zang Zing"
#define BLYNK_AUTH_TOKEN "iccuJhsXcjRIX20sFM75onKmGgwY_R_P"

#define LDR_LED_MODE_VPIN V9
#define LDR_LED_VALUE_VPIN V11

#define IR_DC_MODE_VPIN V10
#define IR_DC_VALUE_VPIN V12

#include <Arduino.h>
#include <FreeRTOSConfig.h>
#include <ESP32Servo.h>
#include <BlynkSimpleEsp32.h>

#define INPUT_ENTER 23
#define INPUT_EXIT 22

#define IN1 26
#define IN2 25
#define EN1 27

#define PWM_CHN 0
#define LEDC_TIMER 0
#define PWM_RES 8
```

```
#define PWM_FREQ 1000

#define LAMP1 4
#define LAMP2 5
#define LAMP3 19
#define LDR_PIN 34

bool automationMode = true;

BaseType_t xSetPowerTask;

int people = 0;
bool enter_changed = false;
bool exit_changed = false;

const char *ssid = "Zayda 2A";
const char *password = "mautauajasih";

static int IR_DC_MODE = 0;
static int IR_DC_VALUE = 0;

static int LDR_LED_MODE = 0;
static int LDR_LED_VALUE = 0;

BLYNK_WRITE(IR_DC_MODE_VPIN)
{
  IR_DC_MODE = param.asInt();
  Serial.println("IR DC Mode read!");
}

BLYNK_WRITE(IR_DC_VALUE_VPIN)
{
```

```
  IR_DC_VALUE = param.asInt();
  Serial.println("IR DC Value read!");
}

BLYNK_WRITE(LDR_LED_MODE_VPIN)
{
  LDR_LED_MODE = param.asInt();
  Serial.println("LDR LED Mode read!");
}

BLYNK_WRITE(LDR_LED_VALUE_VPIN)
{
  LDR_LED_VALUE = param.asInt();
  Serial.println("LDR LED Value read!");
}

void vSetPowerTask(void *pvParameters)
{
  char buff[255];
  while (1)
  {
    if (digitalRead(INPUT_ENTER) == HIGH)
    {
      if (!enter_changed)
      {
        people = people + 1 > 5 ? 5 : people += 1;
        enter_changed = true;
      }
    }
    else
    {
      enter_changed = false;
```

```
          }

          if (digitalRead(INPUT_EXIT) == HIGH)
          {
            if (!exit_changed)
            {
              people = people - 1 < 0 ? 0 : people -= 1;
              exit_changed = true;
            }
          }
          else
          {
            exit_changed = false;
          }

          float formula = (((float)people / 5) * 200) + 50;

          formula = !IR_DC_MODE ? formula : IR_DC_VALUE;

          ledcWrite(PWM_CHN, formula);

          if (people < 5)
          {
                sprintf(buff, "People is %d (Power: %.2f)", people,
formula);
            Serial.println(buff);
          }
          else
          {
              sprintf(buff, "ZENKAIII! People is more than equal to %d
(Power: %.2f)", people, formula);
            Serial.println(buff);
          }
```

```
    vTaskDelay(1000 / portTICK_PERIOD_MS);
  }
}

void controlLighting()
{
  int ldrValue = analogRead(LDR_PIN);

  if (!LDR_LED_MODE)
  {
    if (ldrValue < 100)
    {
      analogWrite(LAMP1, 0);
      analogWrite(LAMP2, 0);
      analogWrite(LAMP3, 0);
      Serial.println("Lampu mati karena ruangan sangat terang.");
    }
    else
    {
      int brightness = map(ldrValue, 0, 4095, 0, 255);
      analogWrite(LAMP1, brightness);
      analogWrite(LAMP2, brightness);
      analogWrite(LAMP3, brightness);
      Serial.println("Brightness lampu: " + String(brightness));
    }
  }
  else
  {
    analogWrite(LAMP1, LDR_LED_VALUE);
    analogWrite(LAMP2, LDR_LED_VALUE);
    analogWrite(LAMP3, LDR_LED_VALUE);
```

```
        Serial.println("Brightness lampu: " + String(LDR_LED_VALUE));
      }
    }

    void TaskLighting(void *pvParameters)
    {
      while (1)
      {
        controlLighting();
        vTaskDelay(1000 / portTICK_PERIOD_MS);
      }
    }

    // put function declarations here:
    void setup()
    {
      // put your setup code here, to run once:
      Serial.begin(115200);

      Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password);

      Blynk.syncVirtual(IR_DC_MODE_VPIN);
      Blynk.syncVirtual(IR_DC_VALUE_VPIN);
      Blynk.syncVirtual(LDR_LED_MODE_VPIN);
      Blynk.syncVirtual(LDR_LED_VALUE_VPIN);

       xSetPowerTask = xTaskCreate(vSetPowerTask, "Set Power", 8192,
NULL, 0, NULL);

        xTaskCreate(TaskLighting, "Lighting Task", 1000, NULL, 1,
NULL);
```

```
  pinMode(LAMP1, OUTPUT);
  pinMode(LAMP2, OUTPUT);
  pinMode(LAMP3, OUTPUT);

  pinMode(INPUT_ENTER, INPUT);
  pinMode(INPUT_EXIT, INPUT);

  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  // pinMode(LEDC_PIN, OUTPUT);

  if (!ledcSetup(PWM_CHN, PWM_FREQ, PWM_RES))
  {
    Serial.println("LEDC setup failed!");
    while (1)
      ; // Halt if setup fails
  }

  ledcAttachPin(EN1, PWM_CHN);

  ledcWrite(PWM_CHN, 180);
}

void loop()
{
  // put your main code here, to run repeatedly:
  Blynk.run();

  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
}
```

Kemudian kita juga akan memasukkan konfigurasi untuk Blynk agar dapat dilakukan manual override dari Blynk. Kita akan membaca virtual pin dari Blynk juga pada awal setup dan selama kode berjalan untuk memastikan kita mendapatkan data paling update dari Blynk. Hanya terdapat 4 virtual pin yang akan digunakan, yaitu **LDR_LED_MODE_VPIN, LDR_LED_VALUE_VPIN, IR_DC_MODE_VPIN, dan IR_DC_VALUE_VPIN**. Virtual pin dengan kata kunci MODE akan berfungsi untuk menentukan apakah ia akan berjalan secara manual atau secara sistem, dan kata kunci VALUE untuk menentukan nilai statis yang akan digunakan jika menggunakan mode manual. Software akan di-upload ke ESP32 dan hasil dari integrasi ini akan dilakukan selanjutnya.

CHAPTER 3

TESTING AND EVALUATION

3.1 TESTING

Proyek *Green Building Comfort* dirancang menggunakan ESP32 sebagai pengontrol utama, dengan komponen seperti breadboard, LED, motor DC, dua sensor IR, driver motor L298N, dan fotoresistor. Sistem ini bertujuan meningkatkan kenyamanan dalam ruangan dengan memanfaatkan sensor untuk memantau intensitas sinar matahari yang masuk dan jumlah orang di dalam ruangan, yang kemudian secara otomatis menyesuaikan kecerahan lampu dan kecepatan kipas.

Sistem ini bekerja dengan menggunakan sensor cahaya (fotoresistor) untuk mendeteksi intensitas sinar matahari. Jika cahaya matahari tidak mencukupi, lampu akan menyala dan menyesuaikan kecerahannya sesuai dengan tingkat cahaya yang tersedia. Selain itu, dua sensor IR ditempatkan di atas pintu masuk dan keluar untuk memantau pergerakan orang. Jumlah orang di ruangan dihitung berdasarkan data dari sensor tersebut, di mana semakin banyak orang di ruangan, kipas akan berputar lebih cepat hingga mencapai lima tingkat kecepatan. Pengujian dilakukan pada berbagai tahap pengembangan menggunakan metodologi berikut:

Pengujian Unit

Pengujian unit dilakukan pada setiap komponen sistem secara terpisah:

- **Fotoresistor**: Diuji di bawah berbagai tingkat cahaya untuk memastikan pembacaan intensitas cahaya yang akurat. Skenario pengujian mencakup kondisi terang, redup, dan gelap total.
- **Sensor IR**: Diuji untuk mendeteksi pergerakan di setiap pintu. Sensor pintu masuk harus menambah jumlah orang ketika mendeteksi seseorang masuk, sedangkan sensor pintu keluar harus mengurangi jumlah ketika mendeteksi seseorang keluar.
- **LED**: Diuji untuk memastikan tingkat kecerahan berubah sesuai dengan data yang diterima dari fotoresistor.
- **Driver Motor L298N**: Diuji untuk memastikan motor DC dapat dikendalikan dengan lancar pada berbagai tingkat kecepatan.

Pengujian Integrasi

Pengujian integrasi dilakukan untuk memastikan semua komponen bekerja secara sinergis:

- **Integrasi Fotoresistor dan LED**: Memastikan bahwa perubahan intensitas sinar matahari secara otomatis menyesuaikan tingkat kecerahan lampu.

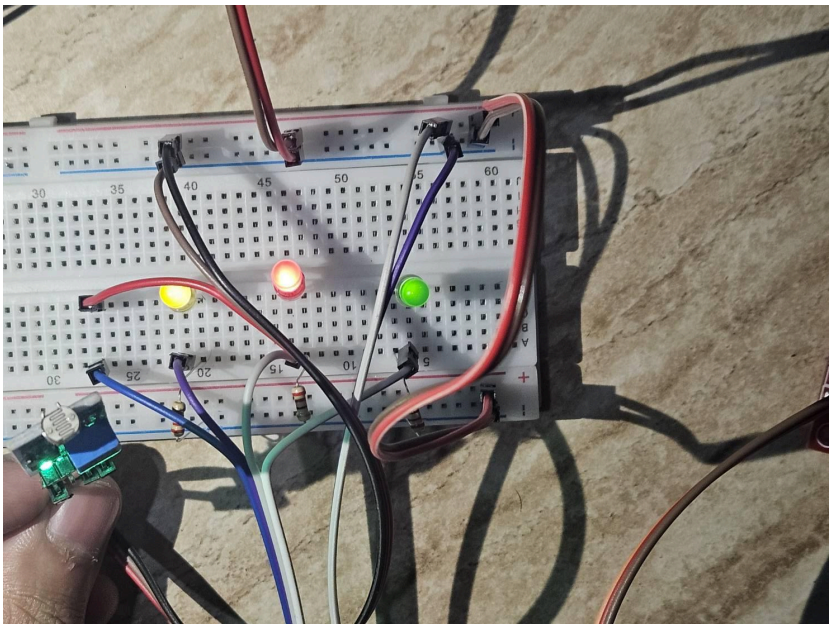


Fig 2. LED Testing

```
Brightness lampu: 46
Brightness lampu: 47
Brightness lampu: 40
```

```
Brightness lampu: 120
Brightness lampu: 122
Brightness lampu: 131
Brightness lampu: 132
```

```
Brightness lampu: 255
Brightness lampu: 255
Brightness lampu: 255
```

Fig 3. LED Result in Serial Monitor

- **Integrasi Sensor IR dan Driver Motor L298N**: Memastikan jumlah orang di ruangan dihitung secara akurat berdasarkan pergerakan di pintu masuk dan keluar,

serta kecepatan kipas menyesuaikan dengan jumlah orang di ruangan, mulai dari tingkat 1 (sedikit orang) hingga tingkat 5 (ruangan penuh).

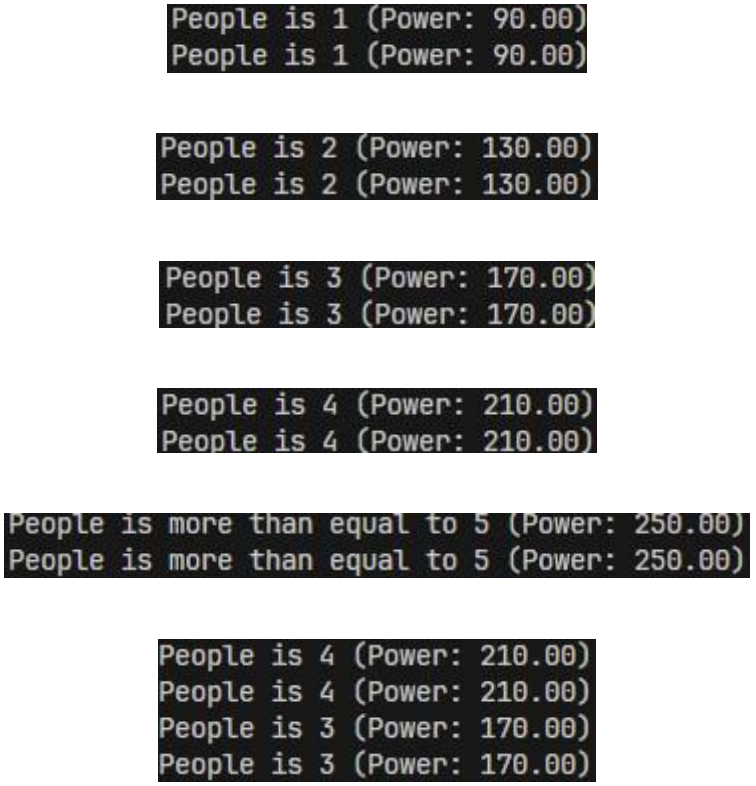


Fig 4. IR Sensor Testing

- **Blynk**: Memastikan Blynk telah terkonfigurasi

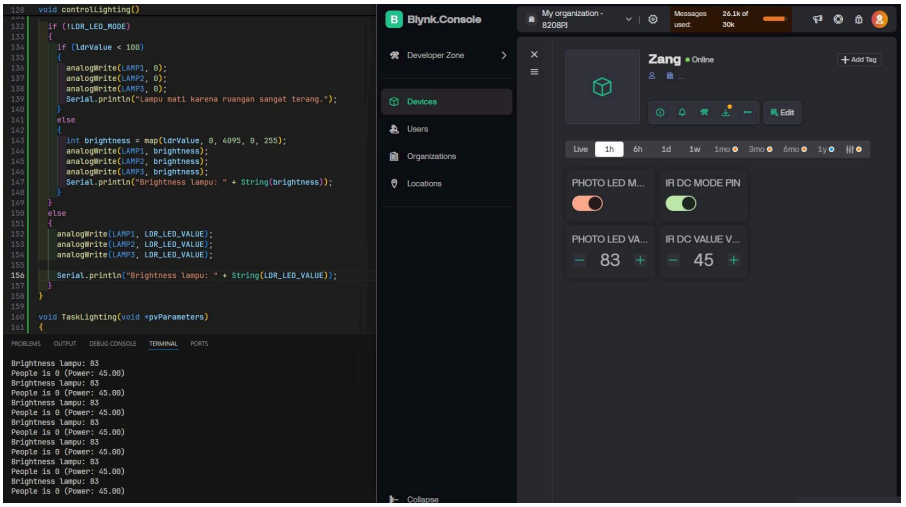


Fig 5. Blynk Testing

3.2 RESULT

Semua komponen berhasil melewati tahap pengujian unit dengan baik. Pada awalnya, terdapat masalah pada salah satu sensor IR yang gagal mendeteksi pergerakan dengan akurat, sehingga jumlah orang yang masuk atau keluar tidak terhitung dengan benar. Setelah mengganti sensor IR yang bermasalah dengan yang baru, fungsinya kembali normal. Fotoresistor berhasil mendeteksi intensitas cahaya dengan akurat dalam berbagai kondisi. LED dapat menyesuaikan tingkat kecerahan berdasarkan intensitas cahaya, dan driver motor L298N berhasil mengontrol motor DC pada semua tingkat kecepatan.

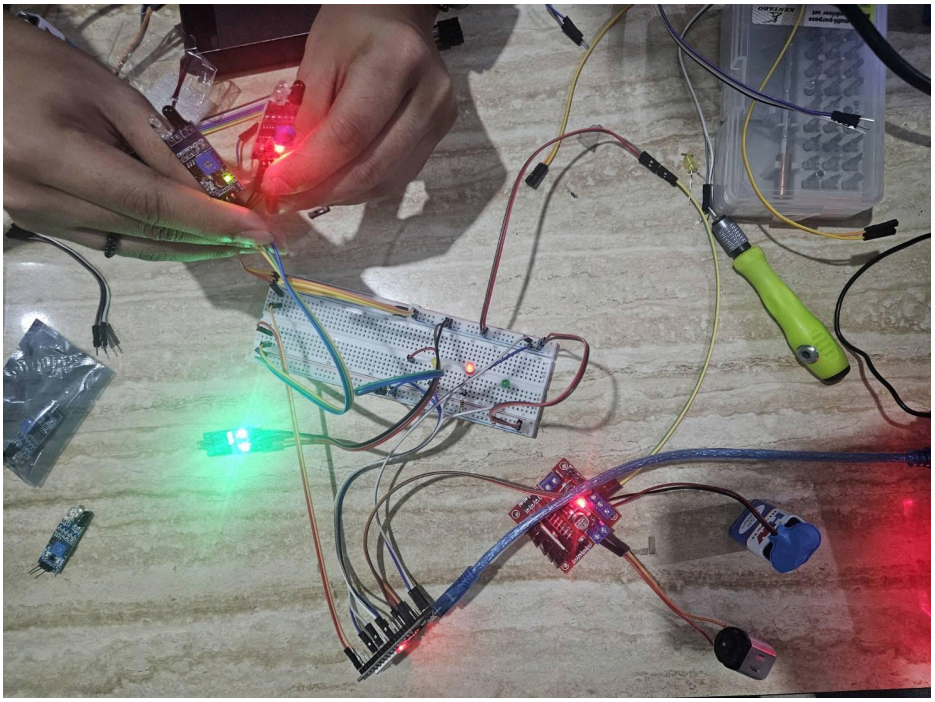


Fig 6. Testing Result

Tahap pengujian integrasi memastikan bahwa semua komponen sistem Green Building Comfort terintegrasi dengan baik. Aliran data antar modul, termasuk fotoresistor, sensor IR, LED, dan driver motor, berjalan lancar. Sistem memberikan penyesuaian waktu nyata terhadap pencahayaan ruangan dan kecepatan kipas berdasarkan intensitas cahaya dan jumlah orang di ruangan. Data dari semua sensor ditampilkan dengan akurat di aplikasi Blynk, termasuk intensitas cahaya, jumlah orang, dan tingkat kecepatan kipas. Fitur-fitur ini menunjukkan kemampuan sistem untuk menciptakan lingkungan dalam ruangan yang nyaman.

3.3 EVALUATION

Sistem *Green Building Comfort* adalah proyek IoT yang telah membuktikan keandalannya melalui pembacaan sensor yang akurat, respons cepat terhadap perubahan lingkungan, dan penggunaan energi yang efisien. Sistem ini secara efektif memantau faktor penting seperti intensitas sinar matahari dan jumlah orang di dalam ruangan, sesuai dengan tujuan utamanya untuk meningkatkan kenyamanan dalam ruangan. Pengujian pengguna mengkonfirmasi akurasi sistem dan potensinya untuk meningkatkan efisiensi energi dengan mengotomatiskan penyesuaian pencahayaan dan ventilasi berdasarkan kondisi waktu nyata.

Salah satu keunggulan utama sistem ini adalah integrasi kontrol manual melalui aplikasi Blynk. Fitur ini memungkinkan pengguna untuk mengatur tingkat kecerahan lampu dan kecepatan kipas secara langsung, memberikan fleksibilitas dan pengalaman yang lebih personal. Kemampuan ini melengkapi fungsi otomatisasi, memastikan pengguna tetap memiliki kendali atas sistem dalam situasi khusus atau tak terduga.

Jadi, *Green Building Comfort* adalah proyek IoT yang lengkap, menampilkan kinerja yang kuat dan utilitas praktis dalam memodernisasi manajemen kenyamanan dalam ruangan. Kombinasi antara otomatisasi yang akurat dan fleksibilitas kontrol manual menjadikannya solusi yang andal untuk meningkatkan efisiensi energi dan adaptabilitas dalam lingkungan bangunan pintar.

CHAPTER 4

CONCLUSION

Sistem *Green Building Comfort* telah berhasil dirancang, dikembangkan, dan diuji untuk meningkatkan kenyamanan dalam ruangan dengan memanfaatkan teknologi berbasis ESP32. Sistem ini memadukan berbagai komponen, seperti fotoresistor, sensor IR, LED, motor DC, dan driver motor L298N.

Hasil pengujian menunjukkan bahwa sistem mampu mengoptimalkan penggunaan pencahayaan dengan memanfaatkan sensor cahaya (fotoresistor) untuk mendeteksi intensitas cahaya matahari dan menyesuaikan kecerahan lampu secara otomatis. Hal ini memungkinkan energi listrik dapat digunakan secara lebih efisien. Selain itu, sistem ini juga mampu mengontrol kecepatan kipas secara otomatis berdasarkan jumlah orang di dalam ruangan yang dihitung menggunakan sensor IR, sehingga sirkulasi udara dapat diatur sesuai kebutuhan kenyamanan pengguna dan juga tetap menghemat energi.

Pengujian integrasi memastikan bahwa data dari sensor dapat diproses dengan cepat dan akurat, sehingga sistem mampu merespons perubahan lingkungan dengan baik. Keberhasilan sistem ini tidak hanya terlihat pada pengujian unit dan integrasi, tetapi juga pada pengujian penerimaan pengguna, di mana sistem menunjukkan kinerja yang andal dalam mensimulasikan kondisi lingkungan nyata dan memenuhi tujuan yang telah ditetapkan.

Sistem *Green Building Comfort* bisa diimplementasikan sebagai solusi inovatif untuk menciptakan ruang yang nyaman dan hemat energi. Dengan memanfaatkan teknologi otomatisasi, sistem ini mampu mendukung penerapan praktik bangunan ramah lingkungan (*green building*). Dengan penyempurnaan lebih lanjut, sistem ini dapat diadaptasi untuk digunakan pada skala yang lebih besar dan menjadi bagian dari sistem bangunan pintar di masa depan.

REFERENCES

[1] Digilab, In MODUL 1–9, Lab Module, Universitas Indonesia, 2024.

[2] "ESP32 - DC Motor," *ESP32 Tutorial*. https://esp32io.com/tutorials/esp32-dc-motor

[3] WALastMinuteEngineers, "In-Depth: Interface L298N DC Motor Driver Module with Arduino," Last Minute Engineers, Nov. 28, 2018. https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/.

[4] "ESP8266 NodeMCU: DC Motor and L298N Driver (Arduino IDE) | Random Nerd Tutorials," Feb. 22, 2024. https://randomnerdtutorials.com/esp8266-nodemcu-dc-motor-l298n-motor-driver-control-speed-direction/.

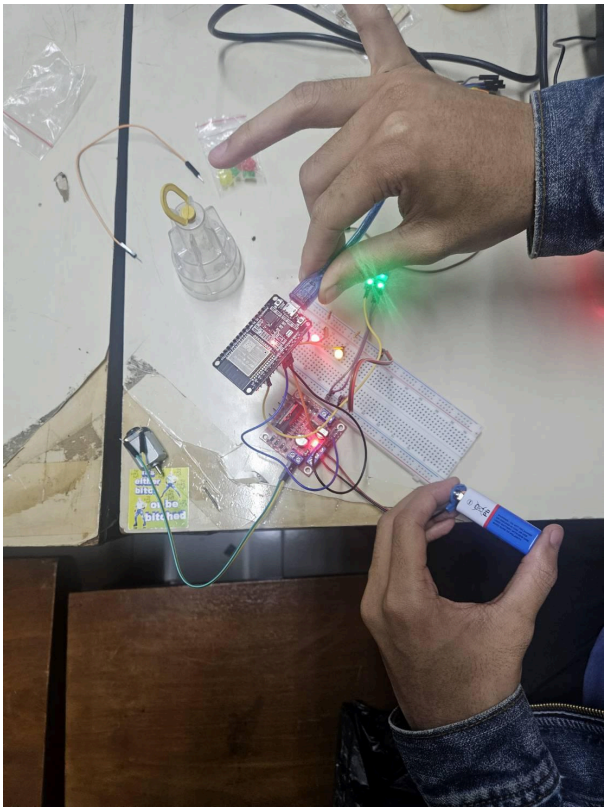
[5] F. Nawazi, "HW201 Infrared (IR) Sensor Module," Circuits DIY, Jun. 26, 2023. https://www.circuits-diy.com/hw201-infrared-ir-sensor-module/.

[6] "Arduino - LDR Module | Arduino Tutorial," Arduino Getting Started. https://arduinogetstarted.com/tutorials/arduino-ldr-module.

[7] "ESP32 PWM with Arduino IDE (Analog Output) | Random Nerd Tutorials," Oct. 30, 2018. https://randomnerdtutorials.com/esp32-pwm-arduino-ide/.

APPENDICES

Appendix A: Project Schematic



Appendix B: Documentation