



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

**RFID : Attendance System with Smart Access**

**GROUP 8**

<b>AJRIYA MUHAMMAD ARKAN</b>	<b>2206031826</b>
<b>IVAN YUANTAMA PRADIPTA</b>	<b>2206824243</b>
<b>RAFLI ADITHIA</b>	<b>2206026523</b>
<b>VALENTINO FARISH ADRIAN</b>	<b>2206825896</b>

## PREFACE

Puji syukur kami panjatkan kehadirat Allah SWT, Tuhan Yang Maha Esa, yang telah melimpahkan rahmat dan karunia-Nya sehingga kami dapat menyelesaikan laporan proyek akhir praktikum Sistem Waktu Nyata dan IoT ini dengan baik. Laporan ini memuat uraian mengenai pembuatan proyek akhir kami, yaitu *RFID: Attendance System with Smart Access*. Proyek ini dirancang untuk mencatat kehadiran karyawan dengan mudah sekaligus memberikan akses otomatis ke locker pribadi melalui teknologi RFID yang dipadukan dengan verifikasi PIN 4 digit.

Kami menyampaikan terima kasih yang sebesar-besarnya kepada Bapak F. Astha Ekadiyanto, ST., M.Sc. selaku dosen mata kuliah Sistem Waktu Nyata dan IoT, serta Bang Juan Jonathan selaku asisten lab pendamping yang telah memberikan bimbingan, saran, dan dukungan selama pelaksanaan proyek akhir ini. Semua masukan yang telah diberikan sangat membantu dalam menyelesaikan proyek ini dengan baik.

Kami menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, kami sangat mengharapkan saran dan kritik yang membangun dari para pembaca demi peningkatan kualitas laporan ini di masa mendatang. Semoga laporan ini dapat memberikan manfaat, baik bagi kami sebagai penyusun, maupun bagi pembaca secara umum

Depok, December 7, 2024

Group 8

## **TABLE OF CONTENTS**

<b>CHAPTER 1.....</b>	<b>4</b>
<b>INTRODUCTION.....</b>	<b>4</b>
1.1    PROBLEM STATEMENT.....	4
1.3    ACCEPTANCE CRITERIA.....	5
1.4    ROLES AND RESPONSIBILITIES.....	5
1.5    TIMELINE AND MILESTONES.....	5
<b>CHAPTER 2.....</b>	<b>7</b>
<b>IMPLEMENTATION.....</b>	<b>7</b>
2.1    HARDWARE DESIGN AND SCHEMATIC.....	7
2.2    SOFTWARE DEVELOPMENT.....	7
2.3    HARDWARE AND SOFTWARE INTEGRATION.....	8
<b>CHAPTER 3.....</b>	<b>9</b>
<b>TESTING AND EVALUATION.....</b>	<b>9</b>
3.1    TESTING.....	9
3.2    RESULT.....	9
3.3    EVALUATION.....	10
<b>CHAPTER 4.....</b>	<b>11</b>
<b>CONCLUSION.....</b>	<b>11</b>

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 PROBLEM STATEMENT**

Sistem manajemen kehadiran karyawan tradisional seringkali menghadapi berbagai kendala, seperti pencatatan data yang kurang akurat, efisiensi yang rendah, dan sulitnya integrasi dengan sistem lain. Beberapa perusahaan masih menggunakan metode manual seperti tanda tangan atau kartu absensi fisik yang rentan terhadap manipulasi atau kehilangan. Di sisi lain, keamanan fasilitas pribadi, seperti locker karyawan, juga menjadi perhatian utama, terutama jika masih menggunakan kunci fisik yang mudah hilang atau diduplikasi. Hal ini menyebabkan pengalaman kerja yang kurang nyaman bagi karyawan dan mempengaruhi efisiensi operasional perusahaan.

Selain itu, meskipun teknologi RFID telah banyak diterapkan sebagai solusi identifikasi, sistem berbasis RFID tunggal sering kali masih memiliki kelemahan dari sisi keamanan. Karyawan yang hanya menggunakan satu lapisan autentikasi, seperti scan RFID tanpa verifikasi tambahan, lebih rentan terhadap penyalahgunaan jika tag RFID hilang atau digunakan oleh pihak yang tidak berwenang. Untuk mengatasi tantangan ini, dibutuhkan solusi sistem manajemen kehadiran dan akses locker yang aman, efisien, dan terintegrasi.

Dengan meningkatnya kebutuhan akan solusi otomatis yang aman, sistem yang tidak hanya mencatat kehadiran karyawan tetapi juga memberikan akses locker secara terkendali dan aman menjadi kebutuhan yang mendesak. Dalam hal ini, kombinasi teknologi RFID dengan autentikasi tambahan seperti PIN dapat menjadi alternatif yang ideal untuk memenuhi kebutuhan tersebut, sambil memberikan kenyamanan lebih bagi pengguna.

### **1.2 PROPOSED SOLUTION**

Untuk mengatasi permasalahan di atas, kami mengembangkan RFID: Attendance System with Smart Access, sebuah sistem yang mengintegrasikan teknologi RFID untuk mencatat kehadiran karyawan dengan mudah sekaligus memberikan akses aman ke locker pribadi. Sistem ini dirancang dengan memanfaatkan RFID sebagai alat identifikasi utama, sehingga memungkinkan karyawan untuk melakukan check-in secara cepat dan akurat.

Keunggulan sistem ini terletak pada penerapan autentikasi berlapis untuk akses locker pribadi. Setelah karyawan melakukan scan RFID, sistem meminta input berupa PIN 4 digit melalui keypad. Hanya jika verifikasi RFID dan PIN berhasil, locker akan terbuka. Hal ini dirancang untuk meningkatkan keamanan sistem secara signifikan dibandingkan solusi RFID tunggal. Dengan sistem ini, karyawan tidak hanya merasakan kemudahan check-in tetapi juga mendapatkan rasa aman terhadap akses ke fasilitas pribadinya.

Proyek ini bertujuan untuk menciptakan solusi yang tidak hanya praktis dan aman, tetapi juga dapat diimplementasikan secara luas di berbagai skala perusahaan. Dengan fitur-fitur yang ada, sistem ini diharapkan dapat menjadi alternatif yang efektif untuk mengatasi masalah manajemen kehadiran karyawan dan akses fasilitas pribadi, serta mendukung peningkatan efisiensi operasional perusahaan secara keseluruhan.

### 1.3 ACCEPTANCE CRITERIA

The acceptance criteria of this project are as follows:

1. Sistem harus mampu menggunakan **RTOS dan Task Scheduling** dengan fungsi `xTaskCreate()` untuk membuat task yang menangani RFID, HTTP, dan keypad.
2. Sistem harus mampu menerapkan **Memory Management** saat proses pembuatan task untuk memastikan penggunaan memori yang efisien.
3. Sistem harus memanfaatkan **Software Timer dan Hardware Interrupts** dengan `vTaskDelay()` untuk memastikan pengaturan waktu dalam validasi RFID dan input PIN.
4. Sistem harus mampu mengimplementasikan konsep **Deadlock & Priority Inversion**, memastikan setiap task seperti `rfidHandle`, `httpHandle`, dan `keypadHandle` berjalan tanpa konflik prioritas.
5. Sistem harus mendukung penggunaan protokol **HTTP** untuk melakukan *GET API* dalam pengambilan data secara real-time.
6. Sistem harus terintegrasi dengan **Blynk** untuk menghitung dan memantau jumlah karyawan yang masuk kantor melalui dashboard Blynk.

## 1.4 ROLES AND RESPONSIBILITIES

The roles and responsibilities assigned to the group members are as follows:

Roles	Responsibilities	Person
Role 1	GET Request API and readme	Ajriya
Role 2	Software Development and Hardware Implementation	Ivan
Role 3	Software Development and Hardware Implementation	Rafli
Role 4	Software Development and Laporan Akhir	Valentino

Table 1. Roles and Responsibilities

## 1.5 TIMELINE AND MILESTONES

Task	4-Dec	5-Dec	6-Dec	7-Dec	8-Dec	9-Dec	10-Dec
Hardware Design Completion							
Software Development							
Integration and Testing							
Final Product Assembly							

The Gantt Chart consist of date interval for:

- Hardware Design completion: A milestone indicating the date when the hardware design for the embedded system is finalized, including schematic.
- Software Development: The date when the development of the user-created assembly code (software) begins, focusing on specific tasks and functionalities.

- c) Integration and Testing of Hardware and Software: A milestone indicating when the hardware and software components are integrated and tested together to ensure proper functionality.
- d) Final Product Assembly and Testing: A milestone marking when the final system product is assembled, tested, and verified to meet the acceptance criteria.

## CHAPTER 2

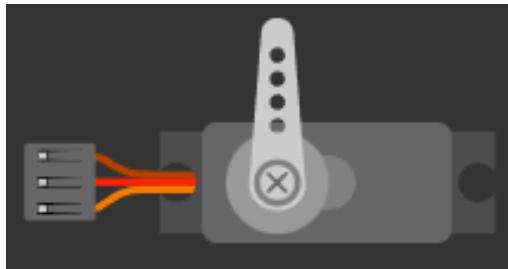
### IMPLEMENTATION

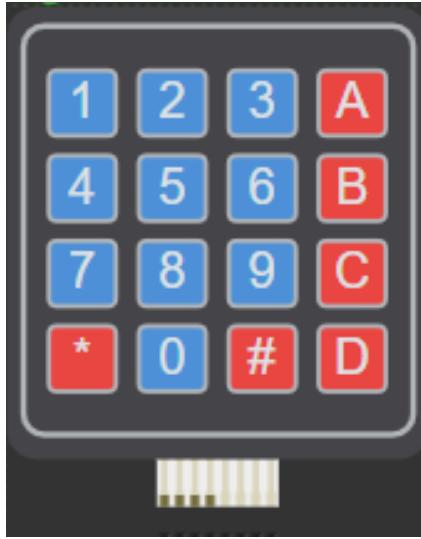
#### 2.1 HARDWARE DESIGN AND SCHEMATIC

Design dari hardware sendiri akan melibatkan beberapa komponen utama, yaitu:

- RFID Card Reader sebagai identitas kartu karyawan.
- Servo sebagai mekanisme locker.
- Buzzer untuk menandakan kartu RFID terdeteksi.
- LCD untuk menampilkan layar locker.
- Keypad untuk input pin locker dari setiap karyawannya.

#### Hardware Design

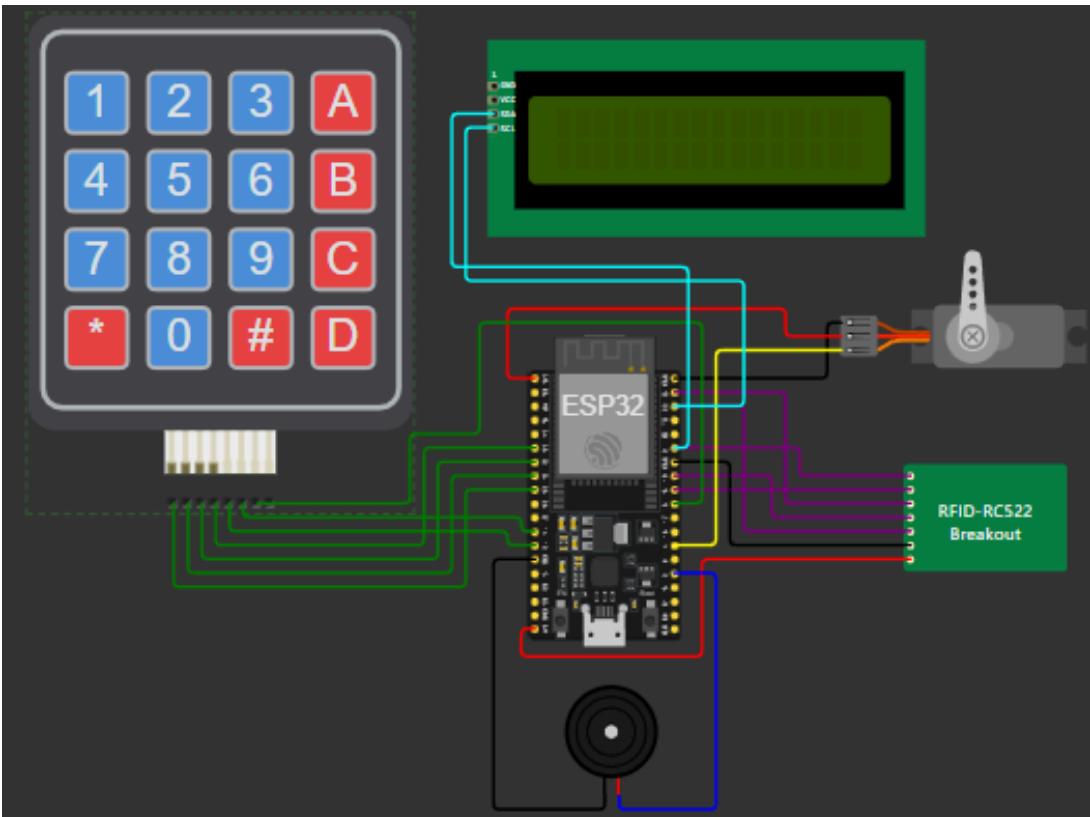
RFID Card Reader	 A green rectangular electronic component labeled "RFID-RC522 Breakout". It has a vertical row of eight yellow circular pads along its left edge.
Servo	 A servo motor assembly. It consists of a grey rectangular base with a black servo horn attached. A multi-pin ribbon cable is connected to the base.

Buzzer	
LCD (I2C)	
Keypad	

ESP32



Schematic



## 2.2 SOFTWARE DEVELOPMENT

Pada bagian ini, software yang dikembangkan menggunakan **ESP32** sebagai mikrokontroler utama untuk mengelola fungsi RFID, verifikasi PIN, dan pengendalian akses locker melalui servo. Program ini dirancang berbasis FreeRTOS untuk memanfaatkan task scheduling, dengan berbagai komponen yang diimplementasikan sebagai task yang berjalan secara paralel. Komponen-komponen ini meliputi task untuk membaca data dari RFID reader, memproses input dari keypad, dan melakukan komunikasi HTTP ke server Blynk.

### Alur Program

1. **Inisialisasi Sistem:** Program memulai dengan menginisialisasi library yang dibutuhkan, seperti **WiFi**, **MFRC522** untuk RFID, **Servo**, dan **LiquidCrystal\_I2C** untuk LCD. Pin GPIO diatur sesuai konfigurasi hardware. Sistem juga menghubungkan ESP32 ke jaringan Wi-Fi dan platform Blynk.
2. **Task RFID:** Task ini membaca UID dari kartu RFID. Ketika kartu RFID berhasil dideteksi, UID akan disimpan dan ditampilkan pada LCD. Program juga memvalidasi UID terhadap data karyawan yang diambil dari server.
3. **Task Keypad:** Setelah UID berhasil divalidasi, sistem meminta karyawan untuk memasukkan PIN melalui keypad. Input PIN diverifikasi dengan data yang disimpan untuk memastikan otentikasi lapisan kedua.
4. **Task HTTP:** Task ini mengelola komunikasi dengan server, seperti melakukan GET API untuk memvalidasi UID atau PIN karyawan dan mengirimkan status kehadiran ke dashboard Blynk.
5. **Aktuasi Servo:** Jika RFID dan PIN terverifikasi, task servo akan diaktifkan untuk membuka locker. Setelah waktu tertentu, servo akan kembali ke posisi awal untuk mengunci locker secara otomatis.
6. **Indikator Buzzer dan LCD:** Buzzer dan LCD digunakan untuk memberikan feedback kepada pengguna, seperti status validasi atau notifikasi kesalahan.

### Kode Program:

```
// #define BLYNK_TEMPLATE_ID "TMPL6HxEZ5GGi"  
// #define BLYNK_TEMPLATE_NAME "FinproIOT"  
// #define BLYNK_AUTH_TOKEN "Tg9T1j4TtzA3wprD9awXRjW3ix_Lvqai"
```

```
#define BLYNK_TEMPLATE_ID "TMPL6LA2UCLob"
#define BLYNK_TEMPLATE_NAME "FinproIOT8"
#define BLYNK_AUTH_TOKEN "TFvadeQaJIuMbuLccqYBe1A6DX9a9F_E"

#include <Arduino.h>
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <HTTPClient.h>
#include <ArduinoJson.h>
#include <ESP32Servo.h>
#include <BlynkSimpleEsp32.h>

#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#include <SPI.h>
#include <MFRC522.h>
#include <Keypad.h>

#define RST_PIN          22      // Configurable, see typical
pin layout above
#define SS_PIN           21      // Configurable, see typical
pin layout above
#define BUZZER_PIN        2      // Pin untuk buzzer
#define SERVO_PIN         4      // Pin untuk buzzer

LiquidCrystal_I2C lcd(0x27, 16, 2);

MFRC522 mfrc522(SS_PIN, RST_PIN);    // Create MFRC522 instance
String UID;

DynamicJsonDocument doc(1024);

Servo servo1;

TaskHandle_t httpTask;
TaskHandle_t rfidTask;
TaskHandle_t keypadTask;

const char ssid[] = "GJ";
```

```
const char password[] = "sumbawa8";
const char auth[] = BLYNK_AUTH_TOKEN;

// // Konfigurasi Keypad 3x4
// const byte ROWS = 4; // Jumlah baris pada keypad
// const byte COLS = 3; // Jumlah kolom pada keypad

// char keys[ROWS][COLS] = {
//     {'1', '2', '3'},
//     {'4', '5', '6'},
//     {'7', '8', '9'},
//     {'*', '0', '#'}
// };

// byte rowPins[ROWS] = {25, 33, 32, 35}; // Pin baris
// byte colPins[COLS] = {12, 14, 5}; // Pin kolom

// Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins,
// ROWS, COLS);

// Variabel untuk menangkap PIN
// String pin;
String enteredPin;
bool pinEntered;
String nama;
String uid;
String pinKaryawan;
int countKaryawan = 0;

// Sertifikat SSL
// openssl s_client -connect mocki.io:443 -showcerts
const char *server_cert = R"(
-----BEGIN CERTIFICATE-----
MIIEEdTCCA12gAwIBAgIJAKcOSkw0grd/MA0GCSqGSIb3DQEBCwUAMGgxCzAJBgNV
BAYTA1VTMSUwIwYDVQQKExxTdGFyZmllbGQgVGVjaG5vbG9naWVzLCBJbmMuMTIw
MAYDVQQLEy1TdGFyZmllbGQgQ2xhc3MgMiBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0
eTAeFw0wOTA5MDIwMDAwMDBaFw0zNDA2MjgxNzM5MTZaMIGYMQswCQYDVQQGEwJV
UzEQMA4GA1UECBMHQXJpem9uYTETMBEGA1UEBxMKU2NvdHRzzGFsZTE1MCMGA1UE
ChMcU3RhcmZpZWxkIFR1Y2hub2xvZ211cywgSW5jLjE7MDkGA1UEAxMyU3RhcmZp
ZWxkIFNlcNzpY2VzIFJvb3QgQ2VydG1maWNhdGUgQXV0aG9yaXR5IC0gRzIwggEi
MA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDVDDrEKv104vW+GZdfjohTsR8/
y8+fIBNtKTrID30892t20GPZNmCom15cAICyL11/9of5JUOG52kbUpqQ4XHj2C0N
Tm/2yEnZtvMaVq4rtnQU68/7JuMauh2WLmo7WJSJR1b/JaCTcF0D2oR0FMNnngRo
-----END CERTIFICATE-----"
)
```

```
0t+0QFodSk7PQ5E751bwAHDLUu57fa4657wx+UX2wmDPE1kCK4DMNEffud6QZW0C
zyyRpqbn3oUYSXxmTqM6bam17jQuug0DuDPfR+uxa4012Zv0gdFFRjKWcIfcAg5J
Q4W2bH07Z0phQazJ1FTfhy/HIrImzJ9ZVGif/L4qL8RVHHVAYBeFA1U5i38FAgMB
AAGjgfAwge0wDwYDVR0TAQH/BAUwAwEB/zAOBgNVHQ8BAf8EBAMCAYwHQYDVR00
BBYEFJxfAN+qAdcwKziIorhtSpzyEZGDBM8GA1UdIwQYMBaAFL9ft9HO3R+G9FtV
rNzXEMIOqYjnME8GCCsGAQUFBwEBBEmwQTAcBgggrBgfEFBQcwAYYQaHR0cDovL28u
c3MyLnVzLzAhBgggrBgfEFBQcwAoYVaHR0cDovL3guc3MyLnVzL3guY2VyMCYGA1Ud
HwQfMB0wG6AZoBeGFWh0dHA6Ly9zLnNzMi51cy9yLmNybDARBgNVHSAECjAIMAYG
BFUdIAAwDQYJKoZIhvcNAQELBQADggEBACMd44pXyn3pF31M8R5V/cxTbj5HD9/G
VfKyBDbtgB9TxF00KGu+x1X8Z+rLP3+QsjPNG1gQggL4+C/1E2DUBc7xgQjB3ad1
108YuW3e950RCLp+QCztweq7dp4zBncdDQh/U90bZKuCJ/Fp1U1ervShw3WnWEQt
8jxwmKy6abaVd38PMV4s/KCH0kdp8H1f9BRUpJVeEXgSYCfOn8J3/yNTd126/+pZ
59vPr5KW7ySaNRB6nJHGDN2Z9j8Z3/VyVOEVqQdZe40/Ui5GjLIAZHycSNPYeehu
VsyzLAQ1xk4meTKCR1b/weWsKh/NEnfVqn3sF/tM+2MR7cwA130A4w=
-----END CERTIFICATE-----");
```

```
// openssl req -newkey rsa:2048 -nodes -keyout client_key.pem
-x509 -days 365 -out client_cert.pem
const char *client_cert = R"(
-----BEGIN CERTIFICATE-----
MIIDazCCA1OgAwIBAgIULFFEDbJUaYNZHKu3TDAXImmj9j8wDQYJKoZIhvcNAQEL
BQAwRTelMAkGA1UEBhMCQVUxEzARBgNVBAgMC1NvbWUtU3RhdGUxITAfBgNVBAoM
GEIudGVybmv0IFdpZGdpdHMgUHR5IEx0ZDAeFw0yNDEyMDkwNDIwNTBaFw0yNTEy
MDkwNDIwNTBaMEUxCzAJBgNVBAYTAKFVRMwEQYDVQQIDAptb211LVN0YXR1MSEw
HwYDVQQKDBhJbnR1cm5ldCBXaWRnaXRzIFB0eSBMdGQwggEiMA0GCSqGSIb3DQEBA
AQAA4IBDwAwggEKAoIBAQCDgHOFK3fiNAVcmi2fOLuWdxshbIKo9Ib54s3Nt0ZV
MkOkoq6LmwCnbMitIfmceHNNjonFbzpeXqJneRwm+sXpLPiBvne4qUs4C+duLDBJ
cpCM/bJEe2E5uJp0VB4X9CBgfrvHtJqERQ0U8/hk4PBnz89bS30yM5a+zyoCkrWz
Hsw66pdfk0nieghJF6A2CNRRn8iKwKAtq13kXVLkGdKwzTZ0pUht2kM2BYTvVgf+
GZcKpPzfykYQEibeao1tvCakhyATvsLxgdKPVGqtdXqZfg004BkrULx04XLwKp8w
qV9PkVY9u2z/wxs0qgUTjYeRvt1ncpbzmLzEx1xOpYMhAgMBAAGjUzBRMB0GA1Ud
DgQWBQBzqfuFgHXwHH9TS71g7hm+4U76eDAPBgfGNVHRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUA
A4IBAQAD
uIsUPvvqX3Ki4JWXMRuSKhoHGjh4TwXMxNrgJM1CYnxkbpmNo7yb+DgUKSLZnj4A
KpKruSnF0A64WRtXvXs0ayXBazJx1TgXkiDC9HxdjYEHzzMO+JIhF9ce1QrtsgVI
/RK4aqrzI0K3mFRT+enZa27viXY2sysdCv0KseumSJIP4qxAOYHLraLtEtvtI8zm
x4VbmxjeNdOLixqD4K10xY/0hJbUBeT1VyRgtk6H7BpbCaVStIyKCjrpXzd2kgrr
MroVaf9SeXDQIL/YBvepu6knRj1WFORfAFBW9NEdItmnY/J7loQ2qCxpIrN4Zwk/
5vBkUYnTxy52wKYSCYnI
-----END CERTIFICATE-----");

const char *client_key = R"(
-----BEGIN PRIVATE KEY-----
```

```
MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQCDgHOFK3fiNAVc  
mi2fOLuWdxshbIKo9IB54s3Nt0ZVMk0koq6LmwCnbMitIfmceHNNjonFbzpeXqJn  
eRwm+sXpLPIbvne4qUs4C+duLDBJcpcm/bJEe2E5uJp0VB4X9CBgfrvHtJqERQ0U  
8/hk4PBNz89bS30yM5a+zyoCkrWzHsw66pdfk0nieghJF6A2CNRRn8iKwKAtql3k  
XVLkGdKwzTZ0pUht2kM2BYTvVgf+GZcKpPzfykYQEibeaoItvCakhyATvsLxgdKP  
VGqtdXqZfg004BkrULx04XLwKp8wqV9PkVY9u2z/wxs0qgUTjYeRvt1ncpbzmLzE  
x1xOpYMhAgMBAAECggEABAiDTMx3ajM/V0b81pg3E4wSxYkYErgKkdEZSJSSKEHc  
ghWJZz1Nq7vpKqfVUsix2TFwfVgTFFITkhdDnYrr8QPt4kB/I9sv+iwx3bwjc2oM  
DVtPSpon0/lqeCmGyMG/Z4EvUVaiY1M6WchCmuC4vqFc879L4jJ4gIPhFUZJAWB0  
8Wh5Tyb60vx0Y01v3r8CwVGnRDYjuvZnZq6k8WEuLBzm4lD5H1jOoLILzD0MVbwd  
o2XylclGk4YNFD9Viz/YPZZd4L6X8ExzALwqNQ1B+ZZeKzt1SxonL1UdL1kK+J8z  
QoenNwEQ7p7kMSL+0XGE0VFpgByYBAXgn40S4xN4zQKBgQDcrvSJdT3Z1KQvhIHb  
7LyJuAqUTyL4s1P8oxJ6TmIJQWq4ivq3/YseQql19DMnYRvQUxmRmNli3S8WtanB  
917Yxgink7YDh9PYyxnK2BhqdQGdh9iVBetTGzVWybrLjcPSZCom78rLZv5GvVN2  
piwcdV3eTcXfzbXti/gGErP8ZQKBgQC2tQuRCtGzBkuAOx1QjIKEheejiA02Vb7a  
NL0xlpODONclrF32BQI64L1PoL3ig9RrTiojqYDA6G2HoN4ZP8WuXGY438HdcgKX  
2JFGmuhx1G04Rh0fAzK+NPYg1GIgL6oxPw02/Q0NL2QIqyxPSPyj60dId+RL+I9P  
S7Hfp1LKDKBqFeub9yZp5x//z3JayDY5cQ2SoXIt5Vm8uzDAh10QUF6K2PtXFZ0  
N60rQUa7XQ09cqagDd2qhFziks4MGcnGdnncnR2v87aNZh+R+sp8d578pEqp6eSz  
+F8JxXSZLE2qIu7Z+2GDDukoH8mNebb/qTEZdNVZw0/71NfwL/iBv999AoGAFZeF  
u21c6NxIlenmDvfcA+HtQw0y6xxxTpq09UpHOo3AF0qsfaORhOXD4J4tcpSDyP6e  
ykIg3itZrlqbhL5dnUJ8LiJ7ZbEwj0Nfv8lUyQcCtVDKxtL47zORFr6Sbh1T7qf0  
x25WWYWN090GyoPkn3aRoQXSLAw+2281oTmltQ0CgYAHuxFvzsmBbjazP3Id5iNN  
1kNITZPExHAyWXrACUFaqF8Z5ZTpHqz13L1M9PRIBVItnIyh/RIZWTInZu86+iXg  
VkJ3eykBMU6UuJfdtfz5164Y5NHvG97FuFS6NEj6ZSWNB0D1rDy/Is1UIQe+AXcz  
JDRzfKL6LqaZvPIpoSSWgQ==  
-----END PRIVATE KEY-----");
```

```
// Mutex untuk melindungi buffer PIN  
SemaphoreHandle_t xMutex;  
  
// Fungsi untuk menangani input dari tombol angka (1 - 9)  
BLYNK_WRITE(V100) { handleKeypadInput(0, param.asInt()); }  
BLYNK_WRITE(V101) { handleKeypadInput(1, param.asInt()); }  
BLYNK_WRITE(V102) { handleKeypadInput(2, param.asInt()); }  
BLYNK_WRITE(V103) { handleKeypadInput(3, param.asInt()); }  
BLYNK_WRITE(V104) { handleKeypadInput(4, param.asInt()); }  
BLYNK_WRITE(V105) { handleKeypadInput(5, param.asInt()); }  
BLYNK_WRITE(V106) { handleKeypadInput(6, param.asInt()); }  
BLYNK_WRITE(V107) { handleKeypadInput(7, param.asInt()); }  
BLYNK_WRITE(V108) { handleKeypadInput(8, param.asInt()); }  
BLYNK_WRITE(V109) { handleKeypadInput(9, param.asInt()); }
```

```
// Fungsi untuk menangani tombol ENTER
BLYNK_WRITE(V110) {
    if (param.asInt() == 1) { // Saat tombol ENTER ditekan
        pinEntered = true;
        Blynk.virtualWrite(V111, "0");
        keypadHandle();
    }
}

BLYNK_WRITE(V1) {
    if (param.asInt() == 1) { // Saat tombol ENTER ditekan
        countKaryawan = 0;
        Blynk.virtualWrite(V0, countKaryawan);
    }
}

void buzzerBeep(int pattern) {
    if (pattern == 1) { // Irama 1 untuk kartu terdaftar
        for (int i = 0; i < 4; i++) {
            digitalWrite(BUZZER_PIN, HIGH);
            delay(100);
            digitalWrite(BUZZER_PIN, LOW);
            delay(150);
        }
    } else if (pattern == 2) { // Irama 2 untuk kartu tidak
terdaftar
        for (int i = 0; i < 2; i++) {
            digitalWrite(BUZZER_PIN, HIGH);
            delay(300);
            digitalWrite(BUZZER_PIN, LOW);
            delay(200);
        }
    }
}

// void keypadHandle(void *parameter) {
//     pin = ""; // Pastikan PIN kosong saat task dimulai
//     lcd.clear();
//     Serial.println(pinKaryawan);

//     for (;;) { // Loop tak terbatas agar terus membaca keypad
```

```
//           char key = keypad.getKey(); // Baca tombol dari keypad

//           if (key) { // Jika ada tombol yang ditekan
//               if (key == '#') { // Jika tombol '#' ditekan,
konfirmasi PIN
//                   Serial.printf("PIN Entered: %s\n",
pin.c_str());}

//           if (strcmp(pin.c_str(), pinKaryawan) == 0) { // Validasi PIN
//               Serial.println("PIN Benar!");

//               // Menampilkan pesan di LCD
//               lcd.clear();
//               lcd.setCursor(0, 0);
//               lcd.print("Welcome !!!");
//               lcd.setCursor(0, 1);
//               lcd.print(nama);

//               // Aksi servo dan buzzer
//               servo1.write(90); // Buka servo
//               buzzerBeep(1); // Bunyikan buzzer
//               vTaskDelay(3000 / portTICK_PERIOD_MS); // Delay 3 detik
//               servo1.write(180); // Tutup servo

} else { // Jika PIN salah
    Serial.println("Invalid PIN");
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Invalid PIN");
    vTaskDelay(2000 / portTICK_PERIOD_MS); // Tampilkan pesan selama 2 detik
}

pin = ""; // Reset PIN setelah validasi
else if (key == '*') { // Tombol '*' untuk reset PIN
    char abah;
}
else { // Tambahkan angka ke PIN
    pin += key; // Tambahkan angka yang ditekan ke
```

```
variabel PIN
//                                Serial.printf("Current PIN: %s\n",
pin.c_str());
```

```
//                                // Tampilkan PIN di LCD
//                                lcd.setCursor(0, 0);
//                                lcd.print("PIN : ");
//                                lcd.setCursor(0, 1);
//                                lcd.print(pin.c_str());
//                                }
//                                }
//                                vTaskDelay(50 / portTICK_PERIOD_MS); // Delay tambahan
untuk efisiensi CPU
//    }
// }
```

```
void keypadHandle() {
    if (pinEntered) {
        Serial.println("Checking PIN...");
        Serial.println(pinKaryawan);
        Serial.println(enteredPin);
        if (enteredPin == pinKaryawan) { // Bandingkan dengan
PIN yang benar
            Serial.println("PIN Correct! Access Granted.");
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("Access Granted!");

            buzzerBeep(1); // Bunyikan buzzer sebagai alarm

            // Servo membuka pintu
            servo1.write(90);
            delay(2000);
            servo1.write(0); // Kembali ke posisi awal
        } else {
            Serial.println("Incorrect PIN! Try Again.");
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("Incorrect PIN!");
            buzzerBeep(2); // Bunyikan buzzer sebagai alarm
        }
    }
}
```

```

        // Reset buffer PIN
        enteredPin = "";
        pinEntered = false;
        lcd.clear();
        lcd.print("Enter PIN: ");

    }

}

void handleKeypadInput(int value, int state) {
    if (state == 1) { // Tombol ditekan
        if (xSemaphoreTake(xMutex, portMAX_DELAY)) {
            if (enteredPin.length() < 4) {
                enteredPin += String(value); // Tambahkan angka ke
buffer PIN
                Serial.print("Entered PIN: ");
                Serial.println(enteredPin);
                Blynk.virtualWrite(V111, enteredPin);

                lcd.setCursor(enteredPin.length() - 1, 1);
                lcd.print("*"); // Gunakan '*' untuk
menyembunyikan angka

            } else {
                Serial.println("PIN sudah mencapai 4 digit.");
            }
            xSemaphoreGive(xMutex); // Lepaskan mutex
        }
    }
}

void httpHandle(void *parameter) {
    for (;;) {
        WiFiClientSecure *client = new WiFiClientSecure;
        if (client) {
            client->setCACert(server_cert);
            client->setCertificate(client_cert);
            client->setPrivateKey(client_key);
        }
    }
}

```

```

        HTTPClient https;
        if (https.begin(*client,
"https://mocki.io/v1/b600eba7-777a-4bee-86cd-57efc2f3f9af")) {
            int httpCode = https.GET();

            if (httpCode > 0) {
                if (httpCode == HTTP_CODE_OK || httpCode ==
HTTP_CODE_MOVED_PERMANENTLY) {
                    String payload = https.getString();
                    DeserializationError error =
deserializeJson(doc, payload);
                    if (error) {
                        Serial.println("Failed to parse
JSON");
                    }
                }
            } else {
                Serial.printf("[HTTPS] GET... failed, error:
%s\n", https.errorToString(httpCode).c_str());
            }
            https.end();
        } else {
            Serial.printf("[HTTPS] Unable to connect\n");
        }
        delete client;
    } else {
        Serial.println("Unable to create client");
    }
}

void rfidHandle(void *parameter) {
    for (;;) {
        if (!mfrc522.PICC_IsNewCardPresent() ||
!mfrc522.PICC_ReadCardSerial()) {
            vTaskDelay(100 / portTICK_PERIOD_MS); // Avoid busy
waiting
            continue;
        }

        Serial.println(F("**CARD DETECTED**"));
    }
}

```

```

// Bunyikan buzzer selama 500 ms
digitalWrite(BUZZER_PIN, HIGH);
delay(500);
digitalWrite(BUZZER_PIN, LOW);

// Delay tambahan sebelum pengecekan UID
vTaskDelay(500 / portTICK_PERIOD_MS);

UID = "";
for (byte i = 0; i < mfrc522.uid.size; i++) {
    UID += (mfrc522.uid.uidByte[i] < 0x10 ? "0" : "") +
        String(mfrc522.uid.uidByte[i], HEX) +
        (i != (mfrc522.uid.size - 1) ? " " : "");
}
UID.toUpperCase();
Serial.println(UID);
mfrc522.PICC_HaltA();

bool found = false;
// Mencari nama berdasarkan UID dari JSON
for (JsonObject obj : doc.as<JsonArray>()) {
    nama = obj["nama"].as<String>(); // Mengambil nilai
    "nama" dan menyimpannya sebagai String
    uid = obj["uid"].as<String>(); // Mengambil nilai
    "uid" dan menyimpannya sebagai String
    pinKaryawan = obj["pin"].as<String>(); // Mengambil
    nilai "pin" dan menyimpannya sebagai String

    if (UID == uid) {
        found = true;
        countKaryawan++;
        Blynk.virtualWrite(V0, countKaryawan);
        lcd.clear();
        lcd.print("Enter PIN: ");

        //xTaskCreate(keypadHandle, "Keypad Task", 4096,
NULL, 1, &keypadTask);
        break;
    }
}

```

```

    if (!found) {
        Serial.println("Card not registered!");

        // Menampilkan pesan di LCD
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Access Denied!");

        // Bunyikan buzzer berirama (kartu tidak terdaftar)
        buzzerBeep(2);
        continue;
    }

    vTaskDelay(1000 / portTICK_PERIOD_MS);
}

}

void blynkHandle(void *parameter) {
    for(;;){
        Blynk.run();
        vTaskDelay(300 / portTICK_PERIOD_MS); // Kurangi CPU usage
    }
}

void setup() {
    Serial.begin(115200);
    pinMode(BUZZER_PIN, OUTPUT);
    servo1.attach(SERVO_PIN);

    SPI.begin();
    mfrc522.PCD_Init();
    mfrc522.PCD_DumpVersionToSerial();

    // Inisialisasi Mutex
    xMutex = xSemaphoreCreateMutex();
    if (xMutex == NULL) {
        Serial.println("Mutex gagal dibuat!");
        while (1);
    }

    Wire.begin(26, 27); // SDA = 26, SCL = 27
}

```

```

lcd.begin();
lcd.backlight();

WiFi.begin(ssid, password);
Serial.printf("Connecting to WiFi with SSID : %s\n", ssid);
while (!WiFi.isConnected());
Serial.println("Wifi Connected");
lcd.print("Wifi Connected");

Blynk.begin(auth, ssid, password);
while (WiFi.status() != WL_CONNECTED);
Serial.println("Blynk Connected");
lcd.clear();
lcd.print("Blynk Connected");

// Create tasks
xTaskCreate(httpHandle, "HTTP Task", 8192, NULL, 15,
&httpTask);
xTaskCreate(rfidHandle, "RFID Task", 4096, NULL, 10,
&rfidTask);
xTaskCreate(blynkHandle, "Blynk Task", 4096, NULL, 5, NULL);
}

void loop() {
    // Kosong, tugas berjalan sebagai task
}

```

### 2.3 HARDWARE AND SOFTWARE INTEGRATION

Pada program yang sudah dibuat, hubungan antara hardware dan software diwujudkan melalui konfigurasi dan kontrol komponen RFID, servo motor, buzzer, dan LCD menggunakan mikrokontroler berbasis ESP32. Komponen RFID (MFRC522) terhubung melalui antarmuka SPI, menggunakan pin Slave Select (SS\_PIN) dan Reset (RST\_PIN) untuk membaca data unik (UID) kartu RFID. Data ini kemudian diverifikasi oleh software menggunakan fungsi pemrosesan UID untuk menentukan apakah akses akan diberikan atau ditolak. Komponen buzzer dan servo motor juga menjadi bagian penting dari sistem ini. Buzzer, yang terhubung ke pin digital (BUZZER\_PIN), memberikan umpan balik auditori kepada pengguna, sedangkan servo motor, yang terhubung ke pin PWM (SERVO\_PIN), mengontrol gerakan buka-tutup pada loker.

Software juga memanfaatkan LCD berbasis protokol I2C untuk memberikan informasi visual, seperti UID kartu yang terdeteksi atau status akses. LCD ini diinisialisasi menggunakan alamat I2C (LCD\_ADDRESS) dan menampilkan teks melalui fungsi lcd.print(). Semua komponen ini diinisialisasi pada fungsi setup() untuk memastikan hardware siap beroperasi, sedangkan logika utama dijalankan pada fungsi loop() yang terus memantau kartu RFID baru, memproses UID, dan memberikan kontrol ke hardware sesuai kebutuhan. Berikut adalah kode program yang mendemonstrasikan integrasi ini:

Potongan kode program dalam implementasi integrasi ke hardware:

```
#define RST_PIN          22           // Pin Reset untuk RFID
#define SS_PIN            21           // Pin Slave Select untuk
RFID
#define BUZZER_PIN        2            // Pin untuk Buzzer
#define SERVO_PIN         4            // Pin untuk Servo Motor
#define LCD_ADDRESS       0x27         // Alamat I2C untuk LCD

#include <SPI.h>
#include <MFRC522.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Servo.h>

MFRC522 mfrc522(SS_PIN, RST_PIN);    // Objek RFID Reader
Servo servo1;                      // Objek Servo
LiquidCrystal_I2C lcd(LCD_ADDRESS, 16, 2); // LCD 16x2

void setup() {
    pinMode(BUZZER_PIN, OUTPUT);      // Buzzer sebagai output
    servo1.attach(SERVO_PIN);        // Servo terhubung ke pin 4

    SPI.begin();                    // Inisialisasi SPI untuk RFID
    mfrc522.PCD_Init();            // Inisialisasi RFID Reader
    lcd.begin();
    lcd.backlight();
    lcd.print("Smart Access");     // Menampilkan teks awal
    servo1.write(0);                // Posisi awal servo
}

void loop() {
```

```
if (mfrc522.PICC_IsNewCardPresent() &&
mfrc522.PICC_ReadCardSerial()) {
    String uid = getUID();           // Membaca UID kartu
    lcd.clear();
    lcd.print("UID: ");
    lcd.print(uid);

    if (verifyUID(uid)) {          // Verifikasi UID
        lcd.setCursor(0, 1);
        lcd.print("Access Granted!");
        digitalWrite(BUZZER_PIN, HIGH);
        delay(200);
        digitalWrite(BUZZER_PIN, LOW);
        servo1.write(90);           // Membuka locker
        delay(5000);
        servo1.write(0);           // Menutup locker
    } else {
        lcd.setCursor(0, 1);
        lcd.print("Access Denied!");
        digitalWrite(BUZZER_PIN, HIGH);
        delay(1000);
        digitalWrite(BUZZER_PIN, LOW);
    }
    delay(1000);
}

String getUID() {
    String uid = "";
    for (byte i = 0; i < mfrc522.uid.size; i++) {
        uid += String(mfrc522.uid.uidByte[i], HEX);
    }
    uid.toUpperCase();
    return uid;
}

bool verifyUID(String uid) {
    return uid == "1234ABCD";      // Contoh UID valid
}
```

## CHAPTER 3

### TESTING AND EVALUATION

#### 3.1 TESTING

Pengujian dilakukan untuk memastikan fungsi dari sistem *RFID Card Attendance System* berjalan sesuai spesifikasi. Pada **Fig 1**, pengujian dilakukan dengan menggunakan kartu yang tidak terdaftar dalam database. Hasilnya, sistem berhasil mendeteksi kartu tidak valid dan menampilkan pesan "**Access Denied!**", memastikan mekanisme validasi kartu berfungsi dengan baik. Selanjutnya, pada **Fig 2**, pengujian dilakukan menggunakan kartu yang sudah terdaftar. Sistem berhasil mengenali kartu tersebut dan menampilkan informasi pemilik kartu, yang menandakan bahwa sistem dapat membedakan kartu yang valid dari yang tidak valid. Proses kemudian dilanjutkan pada **Fig 3**, di mana setelah kartu dikenali, pengguna memasukkan PIN, dan sistem memperbarui jumlah karyawan yang tercatat. Hasilnya menunjukkan bahwa sistem mampu memproses data input dengan akurat dan memperbarui informasi dengan benar.

Pengujian validasi PIN dilakukan pada **Fig 4** dan **Fig 5** untuk memastikan keamanan sistem. Pada **Fig 4**, ketika PIN yang dimasukkan benar, sistem berhasil memverifikasi input tersebut dan menampilkan pesan "**Access Granted!**", serta memberikan izin akses sesuai prosedur. Hal ini menunjukkan bahwa mekanisme verifikasi PIN bekerja dengan baik. Sebaliknya, pada **Fig 5**, ketika pengguna memasukkan PIN yang salah, sistem menolak akses dan menampilkan pesan "**Incorrect PIN!**", yang memastikan sistem mampu menangani kesalahan input secara tepat dan menjaga keamanan akses. Hasil pengujian ini menunjukkan bahwa sistem RFID telah berjalan dengan stabil dan sesuai dengan desain yang diharapkan.

#### 3.2 RESULT

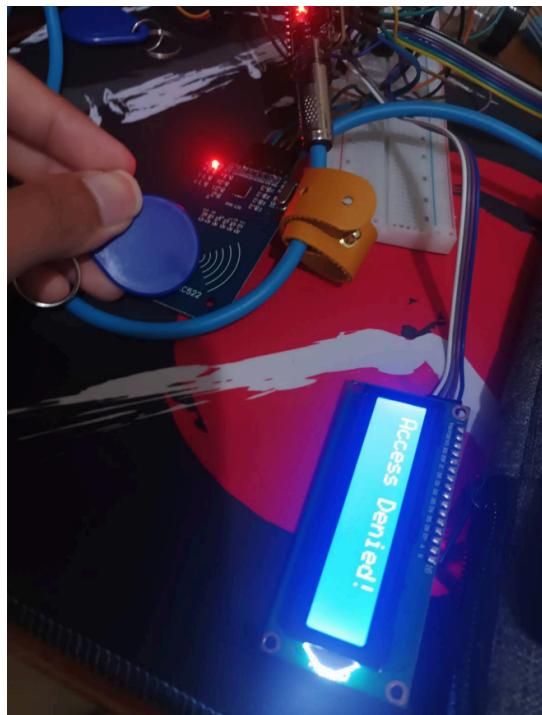


Fig 1. Test jika kartu tidak terdaftar

Pada pengujian ini, sistem berhasil mendeteksi bahwa kartu yang digunakan tidak terdaftar dalam database. Layar menampilkan pesan "**Access Denied!**" dan akses ditolak sesuai dengan prosedur yang telah ditentukan.

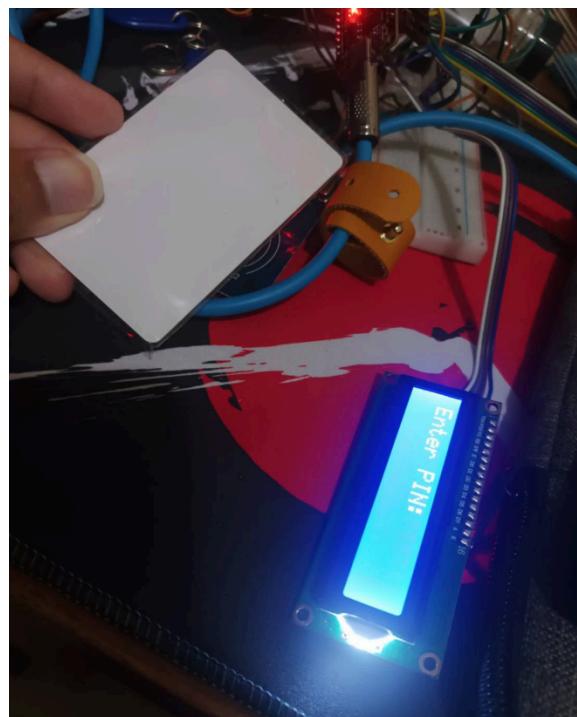


Fig 2. Test jika kartu terdaftar

Pengujian ini menunjukkan bahwa kartu yang digunakan terdeteksi sebagai kartu yang terdaftar dalam sistem. Kemudian, sistem akan melanjutkan proses dengan meminta input PIN untuk validasi lebih lanjut

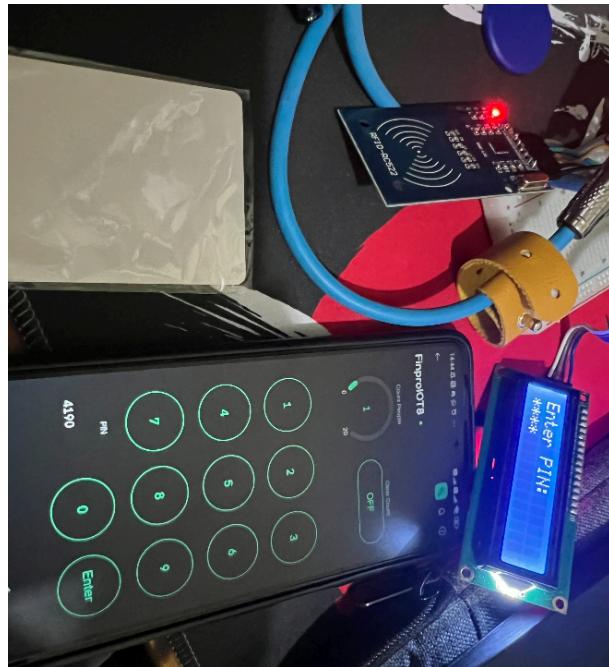


Fig 3. Memasukkan pin dan terlihat jumlah karyawan sudah bertambah.

Dapat dilihat di layar blynk bahwa jumlah karyawan sudah berambah ketika melakukan tap kartu yang id nya sudah terdaftar di database.

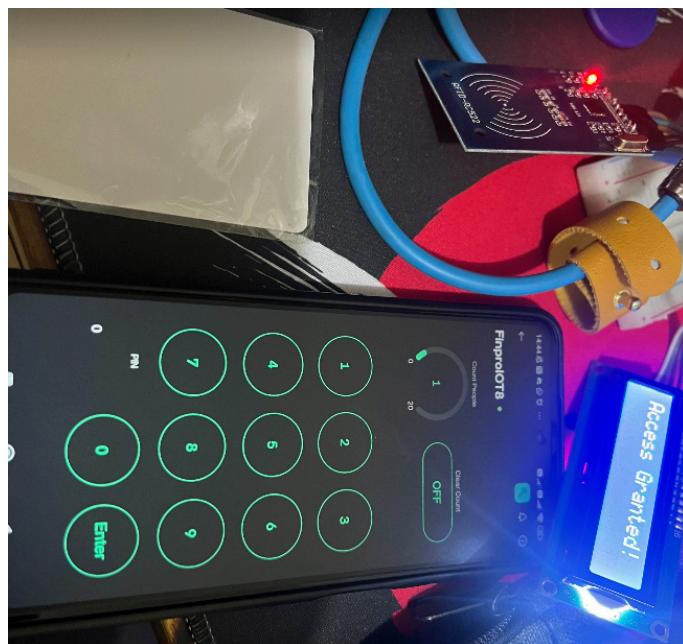


Fig 4. Ketika pin benar.

Pada pengujian ini, sistem berhasil memverifikasi PIN yang benar. Pesan "**Access Granted!**" muncul di layar, dan pengguna diberikan izin untuk mengakses loker nya.

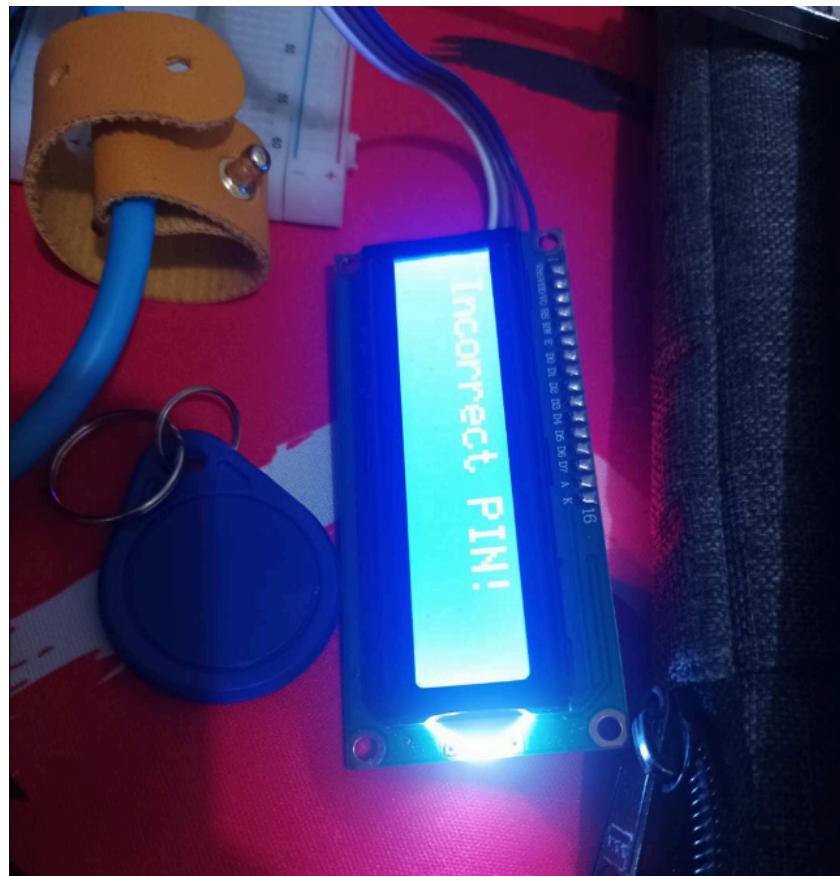


Fig 5. Ketika pin salah.

Sistem mendeteksi bahwa PIN yang dimasukkan salah dan menolak akses pengguna. Layar menampilkan pesan "**Incorrect PIN!**" yang memastikan sistem mampu membedakan input PIN yang benar dan salah untuk keamanan pengguna. Pengguna dapat langsung melakukan tap kartu baru jika menginginkan.

### 3.3 EVALUATION

Evaluasi dilakukan dengan mengukur performa dari setiap modul yang digunakan dalam sistem RFID Card Attendance System. Fokus evaluasi mencakup modul RFID, fungsi pemrosesan sistem (Function), aplikasi Blynk, LCD, serta integrasi keseluruhan antar-komponen untuk memastikan setiap elemen berjalan sesuai spesifikasi dan mendukung tujuan sistem secara keseluruhan.

## **1. Modul RFID**

Modul RFID diuji untuk memastikan kemampuannya membaca kartu dengan cepat dan akurat. Berdasarkan pengujian pada Fig 1 dan Fig 2, modul RFID berhasil membedakan kartu yang valid dan tidak valid. Respons modul terhadap kartu cukup konsisten, dengan waktu pemrosesan rata-rata kurang dari 1 detik. Hal ini menunjukkan bahwa modul RFID telah berfungsi optimal untuk mendukung sistem identifikasi pengguna.

## **2. Fungsi Pemrosesan Sistem (Function)**

Fungsi yang dirancang untuk memvalidasi kartu, memverifikasi PIN, dan memperbarui data karyawan diuji melalui berbagai skenario. Hasil pengujian pada Fig 3 hingga Fig 5 menunjukkan bahwa fungsi-fungsi tersebut bekerja secara akurat. Validasi input yang benar menghasilkan akses yang diizinkan, sementara input yang salah menghasilkan penolakan akses. Ini membuktikan bahwa alur logika pemrosesan telah dirancang dengan baik dan tahan terhadap kesalahan input pengguna.

## **3. Aplikasi Blynk**

Penggunaan aplikasi Blynk untuk memantau data karyawan diuji untuk mengevaluasi keterpaduannya dengan sistem. Pada Fig 3, hasil menunjukkan bahwa aplikasi berhasil memperbarui jumlah karyawan yang tercatat secara real-time setelah kartu yang valid digunakan. Selain itu, antarmuka aplikasi memberikan informasi yang jelas, menjadikannya alat yang efisien untuk pengelolaan data secara jarak jauh.

## **4. LCD**

Modul LCD diuji untuk memastikan kemampuannya memberikan umpan balik visual kepada pengguna. Berdasarkan pengujian, layar LCD mampu menampilkan pesan seperti "Access Denied!", "Access Granted!", dan "Incorrect PIN!" dengan cepat dan jelas. Hal ini membantu pengguna memahami status akses mereka dalam waktu singkat. Kecepatan tampilan pesan juga menunjukkan bahwa modul LCD terintegrasi dengan baik dalam sistem.

## **5. Integrasi Keseluruhan Evaluasi**

Integrasi keseluruhan bertujuan memastikan komunikasi antar-modul berjalan lancar. Hasil pengujian menunjukkan bahwa setiap modul dapat saling berkoordinasi secara real-time tanpa penundaan yang signifikan. Contohnya, setelah kartu valid dikenali, sistem segera meminta input PIN, memprosesnya, dan memberikan akses jika verifikasi berhasil. Semua proses ini berlangsung tanpa hambatan, membuktikan bahwa sistem telah dirancang dengan sinergi yang baik antar-komponen.

## CHAPTER 4

### CONCLUSION

*RFID: Attendance System with Smart Access* berhasil dikembangkan sebagai solusi yang mengatasi permasalahan dalam manajemen kehadiran karyawan dan keamanan akses fasilitas pribadi. Proyek ini tidak hanya memanfaatkan teknologi RFID sebagai alat identifikasi utama tetapi juga mengintegrasikan autentikasi tambahan berupa PIN untuk meningkatkan keamanan akses locker. Keseluruhan sistem dirancang untuk memberikan kemudahan, efisiensi, dan rasa aman kepada pengguna sekaligus meningkatkan efisiensi operasional perusahaan.

Berdasarkan hasil pengujian dan evaluasi yang telah dilakukan, sistem ini mampu memenuhi kebutuhan sebagaimana yang diidentifikasi dalam *Problem Statement* pada Chapter 1. Sistem menunjukkan performa yang stabil dalam mencatat kehadiran karyawan dengan akurasi tinggi, memberikan validasi akses yang aman, serta menawarkan integrasi yang mudah dengan protokol HTTP dan aplikasi Blynk untuk pengelolaan data secara real-time. Semua modul, seperti RFID, keypad, LCD, dan Blynk, telah berfungsi secara optimal sesuai dengan spesifikasi yang ditentukan.

Dari segi *Acceptance Criteria*, proyek ini telah memenuhi seluruh kriteria berikut:

1. **RTOS dan Task Scheduling:** Sistem menggunakan fungsi `xTaskCreate()` untuk menangani tugas RFID, HTTP, dan keypad secara terpisah. Task scheduling yang diimplementasikan memastikan eksekusi tugas berjalan secara terorganisir tanpa konflik.
2. **Memory Management:** Pengelolaan memori telah diterapkan selama pembuatan task untuk menghindari penggunaan memori yang berlebihan, menjaga performa sistem tetap efisien.
3. **Software Timer dan Hardware Interrupts:** Sistem memanfaatkan `vTaskDelay()` untuk pengaturan waktu dalam validasi RFID dan input PIN, memberikan respon yang akurat dan real-time.

4. **Deadlock & Priority Inversion:** Tidak ditemukan konflik atau deadlock selama proses pengujian. Prioritas antar-task seperti `rfidHandle`, `httpHandle`, dan `keypadHandle` diatur dengan baik, memastikan kelancaran eksekusi.
5. **Protokol HTTP:** Sistem mendukung pengambilan data menggunakan protokol HTTP melalui API *GET request* untuk integrasi data secara real-time.
6. **Blynk Integration:** Dashboard Blynk berhasil menampilkan data kehadiran karyawan secara real-time dan mencatat jumlah karyawan yang masuk dengan akurasi tinggi.

Sistem ini tidak hanya memenuhi seluruh *Acceptance Criteria* tetapi juga memberikan solusi yang inovatif dan praktis bagi perusahaan dalam meningkatkan akurasi, keamanan, dan efisiensi proses operasional. Dengan implementasi yang tepat, sistem ini dapat diadaptasi secara luas untuk kebutuhan perusahaan di berbagai skala. Sebagai langkah pengembangan lebih lanjut, peningkatan fitur seperti *lockout mechanism* untuk menghindari percobaan PIN secara berulang dan pengamanan data komunikasi melalui enkripsi dapat dipertimbangkan guna meningkatkan keandalan dan keamanan sistem.

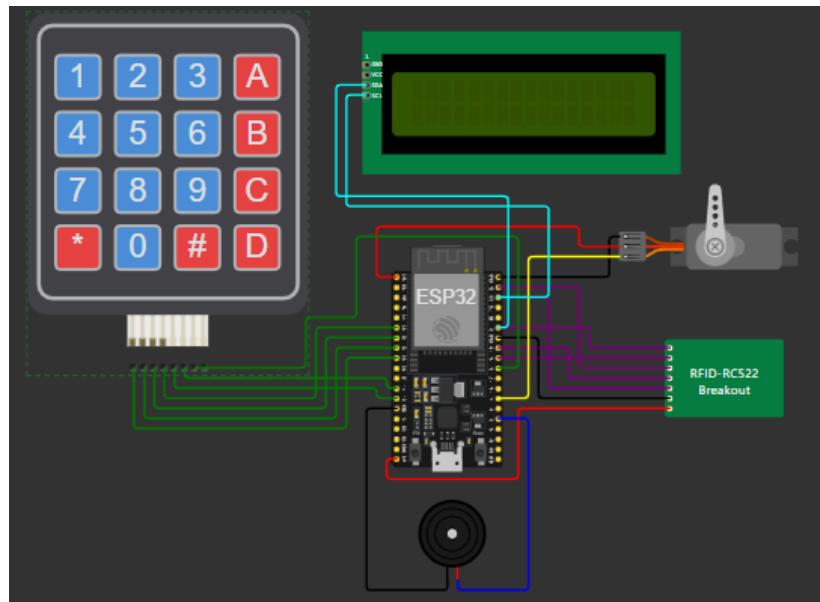
## REFERENCES

- [1] EletronicWings, “RFID RC522 Interfacing with ESP32 | ESP32,” www.electronicwings.com, 2024. Available: <https://www.electronicwings.com/esp32/rfid-rc522-interfacing-with-esp32>. [Accessed: Dec. 07, 2024]
- [2] F. Koyanagi, “ESP32 With RFID: Access Control,” Instructables. Available: <https://www.instructables.com/ESP32-With-RFID-Access-Control/>. [Accessed: Dec. 07, 2024]
- [3] Random Nerd Tutorials, “ESP32 HTTPS Requests (Arduino IDE) | Random Nerd Tutorials,” RANDOM NERD TUTORIALS, Dec. 15, 2022. Available: <https://randomnerdtutorials.com/esp32-https-requests/>. [Accessed: Dec. 07, 2024]
- [4] Random Nerd Tutorials, “ESP32 HTTP GET and HTTP POST with Arduino IDE | Random Nerd Tutorials,” RANDOM NERD TUTORIALS, Apr. 08, 2020. Available: <https://randomnerdtutorials.com/esp32-http-get-post-arduino/>. [Accessed: Dec. 07, 2024]
- [5] Random Nerd Tutorials, “Decoding and Encoding JSON Arduino | Random Nerd Tutorials,” RANDOM NERD TUTORIALS, Aug. 29, 2017. Available: <https://randomnerdtutorials.com/decoding-and-encoding-json-with-arduino-or-esp8266/>. [Accessed: Dec. 07, 2024]
- [6] ElectronicWings, “Keypad Interfacing with ESP32 | ESP32,” Electronicwings.com, 2019. Available: <https://www.electronicwings.com/esp32/keypad-interfacing-with-esp32>. [Accessed: Dec. 07, 2024]
- [7] Blynk, “Blynk Documentation,” Blynk.io, Mar. 07, 2024. Available: <https://docs.blynk.io/en>. [Accessed: Dec. 07, 2024]
- [8] Random Nerd Tutorials, “I2C LCD with ESP32 on Arduino IDE - ESP8266 compatible | Random Nerd Tutorials,” RANDOM NERD TUTORIALS, Feb. 01, 2019. Available: <https://randomnerdtutorials.com/esp32-esp8266-i2c-lcd-arduino-ide/>. [Accessed: Dec. 07, 2024]

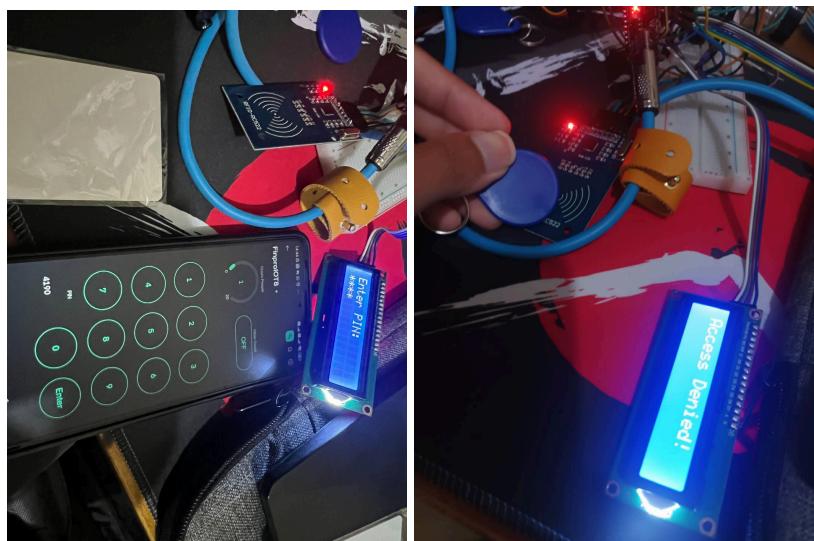
- [9] Digilab UI, “Module 5 : Deadlock & Multicore Systems,” Digilabdte.com, 2024. Available:  
<https://learn.digilabdte.com/books/realtime-system-iot/page/module-5-deadlock-multi-core-systems>. [Accessed: Dec. 07, 2024]
- [10] Digilab UI, “Module 5 : Deadlock & Multicore Systems,” Digilabdte.com, 2024. Available:  
<https://learn.digilabdte.com/books/realtime-system-iot/page/module-5-deadlock-multi-core-systems>. [Accessed: Dec. 09, 2024]
- [11] abah abah, “Muhammad Zhavier Naufal Rachman - East Jakarta, Jakarta, Indonesia | Professional Profile | LinkedIn,” LinkedIn.com, 2024. Available: [https://id.linkedin.com/in/muhammad-zhavier-naufal-rachman-8ab294300?trk=people-guest\\_people\\_search-card](https://id.linkedin.com/in/muhammad-zhavier-naufal-rachman-8ab294300?trk=people-guest_people_search-card). [Accessed: Dec. 10, 2024]

## APPENDICES

### Appendix A: Project Schematic



### Appendix B: Documentation



### Appendix C: Implementation Video

<https://youtube.com/shorts/Gdkh6c0Cn5s>