LAMPIRAN

CURICULUM VITAE

NAMA LENGKAP : RAHMIKA

NIM : 061940352364

TEMPAT, TANGGAL LAHIR : LAHAT, 15 APRIL 2002

ALAMAT : DESA BANU AYU, KEC. KIKIM SELATAN, KAB LAHAT

TELEPON : 081383722182

RIWAYAT PENDIDIKAN

PENDIDIKAN	NAMA SEKOLAH	TAHUN LULUS
SD	SDN 07 KIKIM SELATAN	2013
SMP	SMPN 38 PALEMBANG	2016
SMA	SMA BINA WARGA 1 PALEMBANG	2019

PENGALAMAN INTERNSHIP:

NO	PENGALAMAN	TAHUN
1	INTERNSHIP PT. ANGKASA PURA II	2022

PENGALAMAN ORGANISASI

NO	PENGALAMAN	TAHUN
1	ANGGOTA TIM ROBOT POLSRI	2019
2	ANGGOTA LDK KHARISMA POLSRI	2019

Semua data yang tercantum dalam Curiculum Vitae ini adalah benar dan dapat dipertanggung jawabkan.

Palembang, Agustus 2023

(Rahmika)

RINCIAN BIAYA

No	Komponen	Harga Satuan	Kuantitas
1	Motor Elektrik (AC)	Rp. 3.500.000	1 Buah
2	NodeMCU	Rp. 45.000	1 Buah
3	Modul Relay 4ch	Rp. 35.000	1 Buah
4	Power Supply	Rp. 150.000	1 Buah
5	Switch	Rp. 100.000	2 Buah
6	Rel dan Baut	Rp. 120.000	2 buah
7	Plat Besi	Rp. 70.000	1 buah
8	Ubec	Rp. 45.000	1 Buah
9	Mini Breadboard	Rp. 10.000	1 Buah
10	Jumper	Rp. 10.000	1 Buah
11	Box	Rp. 25.000	1 Buah
12	Kabel	Rp. 100.000	2 Buah
13	**Total**	**Rp. 4.210.000**	

KODE PROGRAM PERANGKAT KERAS

```
#include <Arduino.h>
#if defined(ESP32) || defined(ARDUINO_RASPBERRY_PI_PICO_W)
#include <WiFi.h>
#elif defined(ESP8266)
#include <ESP8266WiFi.h>
#endif


#define PIN_RELAY_1  D2 // the Arduino pin, which connects to
the IN1 pin of relay module
#define PIN_RELAY_2  D3 // the Arduino pin, which connects to
the IN2 pin of relay module
#define PIN_RELAY_3  D4 // the Arduino pin, which connects to
the IN3 pin of relay module
#define PIN_RELAY_4  D5 // the Arduino pin, which connects to
the IN4 pin of relay module


#include <Firebase_ESP_Client.h>

// Provide the token generation process info.
#include <addons/TokenHelper.h>

// Provide the RTDB payload printing info and other helper
functions.
#include <addons/RTDBHelper.h>

/* 1. Define the WiFi credentials */
#define WIFI_SSID "OPPO A15s"
#define WIFI_PASSWORD "mikamika"

// For the following credentials, see
examples/Authentications/SignInAsUser/EmailPassword/EmailPassw
ord.ino

/* 2. Define the API Key */
#define API_KEY "AIzaSyBOynK_pfrKQzRiFEJJD5L0wG9n95CaPGc"

/* 3. Define the RTDB URL */
#define DATABASE_URL "smart-gate-m-default-
rtdb.firebaseio.com" //<databaseName>.firebaseio.com or
<databaseName>.<region>.firebasedatabase.app

/* 4. Define the user Email and password that alreadey
registerd or added in your project */
#define USER_EMAIL "rhmaprlaa1542@gmail.com"
#define USER_PASSWORD "niyuroh15"

// Define Firebase Data object
//FirebaseData fbdo;
```

```
FirebaseData stream;
FirebaseData fbdo;

FirebaseAuth auth;
FirebaseConfig config;

unsigned long sendDataPrevMillis = 0;

unsigned long count = 0;

#if defined(ARDUINO_RASPBERRY_PI_PICO_W)
WiFiMulti multi;
#endif


volatile bool dataChanged = false;
String data_didapat = "";

const int shiftAmount = 15;  // Number of positions to shift
each character

String decrypt(String message, int shift) {
  String decryptedText = "";

  for (int i = 0; i < message.length(); i++) {
    char c = message.charAt(i);

    // Check if the character is a letter
    if (isAlpha(c)) {
      // Get the ASCII value of the character
      int asciiValue = c;

      // Shift the ASCII value back by the specified amount
      int shiftedAsciiValue = (asciiValue - 'A' - shift + 26)
% 26 + 'A';

      // Convert the shifted ASCII value back to a character
      char decryptedChar = shiftedAsciiValue;

      // Append the decrypted character to the result
      decryptedText += decryptedChar;
    } else {
      // If the character is not a letter, append it as it is
      decryptedText += c;
    }
  }

  return decryptedText;
}

void streamCallback(FirebaseStream data)
{
```

```
  Serial.printf("sream path, %s\nevent path, %s\ndata type,
%s\nevent type, %s\n\n",
                data.streamPath().c_str(),
                data.dataPath().c_str(),
                data.dataType().c_str(),
                data.eventType().c_str());
  printResult(data); // see addons/RTDBHelper.h
  data_didapat = data.to<String>();
  Serial.println();

  // This is the size of stream payload received (current and
max value)
  // Max payload size is the payload size under the stream
path since the stream connected
  // and read once and will not update until stream
reconnection takes place.
  // This max value will be zero as no payload received in
case of ESP8266 which
  // BearSSL reserved Rx buffer size is less than the actual
stream payload.
  Serial.printf("Received stream payload size: %d (Max.
%d)\n\n", data.payloadLength(), data.maxPayloadLength());

  // Due to limited of stack memory, do not perform any task
that used large memory here especially starting connect to
server.
  // Just set this flag and check it status later.
  dataChanged = true;
}

void streamTimeoutCallback(bool timeout)
{
  if (timeout)
    Serial.println("stream timed out, resuming...\n");

  if (!stream.httpConnected())
    Serial.printf("error code: %d, reason: %s\n\n",
stream.httpCode(), stream.errorReason().c_str());
}

void setup()
{

  Serial.begin(115200);

#if defined(ARDUINO_RASPBERRY_PI_PICO_W)
  multi.addAP(WIFI_SSID, WIFI_PASSWORD);
  multi.run();
#else
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
#endif

  Serial.print("Connecting to Wi-Fi");
  unsigned long ms = millis();
```

```
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(300);
#if defined(ARDUINO_RASPBERRY_PI_PICO_W)
    if (millis() - ms > 10000)
      break;
#endif
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

  Serial.printf("Firebase Client v%s\n\n",
FIREBASE_CLIENT_VERSION);

  /* Assign the api key (required) */
  config.api_key = API_KEY;

  /* Assign the user sign in credentials */
  auth.user.email = USER_EMAIL;
  auth.user.password = USER_PASSWORD;

  /* Assign the RTDB URL (required) */
  config.database_url = DATABASE_URL;

  /* Assign the callback function for the long running token
generation task */
  config.token_status_callback = tokenStatusCallback; // see
addons/TokenHelper.h

  // Or use legacy authenticate method
  // config.database_url = DATABASE_URL;
  // config.signer.tokens.legacy_token = "<database secret>";

  // To connect without auth in Test Mode, see
Authentications/TestMode/TestMode.ino


////////////////////////////////////////////////////////////
////////////////////////////////
  // Please make sure the device free Heap is not lower than
80 k for ESP32 and 10 k for ESP8266,
  // otherwise the SSL connection will fail.

////////////////////////////////////////////////////////////
////////////////////////////////

#if defined(ESP8266)
  // In ESP8266 required for BearSSL rx/tx buffer for large
data handle, increase Rx size as needed.
```

```
  fbdo.setBSSLBufferSize(2048 /* Rx buffer size in bytes from
512 - 16384 */, 2048 /* Tx buffer size in bytes from 512 -
16384 */);
#endif

  // Limit the size of response payload to be collected in
FirebaseData
  fbdo.setResponseSize(2048);

  Firebase.begin(&config, &auth);

  // The WiFi credentials are required for Pico W
  // due to it does not have reconnect feature.
#if defined(ARDUINO_RASPBERRY_PI_PICO_W)
  config.wifi.clearAP();
  config.wifi.addAP(WIFI_SSID, WIFI_PASSWORD);
#endif

  // Comment or pass false value when WiFi reconnection will
control by your code or third party library
  Firebase.reconnectWiFi(true);


  // Recommend for ESP8266 stream, adjust the buffer size to
match your stream data size
#if defined(ESP8266)
  stream.setBSSLBufferSize(2048 /* Rx in bytes, 512 - 16384
*/, 512 /* Tx in bytes, 512 - 16384 */);
#endif

  if (!Firebase.RTDB.beginStream(&stream, "/message"))
    Serial.printf("sream begin error, %s\n\n",
stream.errorReason().c_str());

  Firebase.RTDB.setStreamCallback(&stream, streamCallback,
streamTimeoutCallback);

  Firebase.setDoubleDigits(5);

  config.timeout.serverResponse = 10 * 1000;

  /** Timeout options.

    //WiFi reconnect timeout (interval) in ms (10 sec - 5 min)
when WiFi disconnected.
    config.timeout.wifiReconnect = 10 * 1000;

    //Socket connection and SSL handshake timeout in ms (1 sec
- 1 min).
    config.timeout.socketConnection = 10 * 1000;

    //Server response read timeout in ms (1 sec - 1 min).
    config.timeout.serverResponse = 10 * 1000;
```

```
    //RTDB Stream keep-alive timeout in ms (20 sec - 2 min)
when no server's keep-alive event data received.
    config.timeout.rtdbKeepAlive = 45 * 1000;

    //RTDB Stream reconnect timeout (interval) in ms (1 sec -
1 min) when RTDB Stream closed and want to resume.
    config.timeout.rtdbStreamReconnect = 1 * 1000;

    //RTDB Stream error notification timeout (interval) in ms
(3 sec - 30 sec). It determines how often the readStream
    //will return false (error) when it called repeatedly in
loop.
    config.timeout.rtdbStreamError = 3 * 1000;

    Note:
    The function that starting the new TCP session i.e. first
time server connection or previous session was closed, the
function won't exit until the
    time of config.timeout.socketConnection.

    You can also set the TCP data sending retry with
    config.tcp_data_sending_retry = 1;

  */


  // initialize digital pin as an output.
  pinMode(PIN_RELAY_1, OUTPUT);
  pinMode(PIN_RELAY_2, OUTPUT);
  pinMode(PIN_RELAY_3, OUTPUT);
  pinMode(PIN_RELAY_4, OUTPUT);

  digitalWrite(PIN_RELAY_1, HIGH);
  digitalWrite(PIN_RELAY_2, HIGH);
  digitalWrite(PIN_RELAY_3, HIGH);
  digitalWrite(PIN_RELAY_4, HIGH);
}

void loop()
{


#if defined(ARDUINO_RASPBERRY_PI_PICO_W)
  Firebase.RTDB.runStream();
#endif

//  if (Firebase.ready() && (millis() - sendDataPrevMillis >
15000 || sendDataPrevMillis == 0))
//  {
//    sendDataPrevMillis = millis();
//    count++;
//    FirebaseJson json;
//    json.add("data", "hello");
//    json.add("num", count);
```

```
//    Serial.printf("Set json... %s\n\n",
Firebase.RTDB.setJSON(&fbdo, "/test/stream/data/json", &json)
? "ok" : fbdo.errorReason().c_str());
//  }

  if (dataChanged)
  {
    dataChanged = false;
    Serial.println(decrypt(data_didapat,shiftAmount));

    if (decrypt(data_didapat,shiftAmount) == "BUKA") {
      Serial.println("pintu dibuka");
      digitalWrite(PIN_RELAY_1, LOW);
      delay(1000);
      digitalWrite(PIN_RELAY_1, HIGH);
    } else {
      Serial.println("pintu ditutup");
      digitalWrite(PIN_RELAY_2, LOW);
      delay(1000);
      digitalWrite(PIN_RELAY_2, HIGH);
    }

    //    Serial.println("ada perubahan data");

    // When stream data is available, do anything here...
  }
}

// Firebase.ready() should be called repeatedly to handle
authentication tasks.

//  if (Firebase.ready() && (millis() - sendDataPrevMillis >
15000 || sendDataPrevMillis == 0))
//  {
//    sendDataPrevMillis = millis();
//
//    Serial.printf("Set bool... %s\n",
Firebase.RTDB.setBool(&fbdo, F("/test/bool"), count % 2 == 0)
? "ok" : fbdo.errorReason().c_str());
//
//    Serial.printf("Get bool... %s\n",
Firebase.RTDB.getBool(&fbdo, FPSTR("/test/bool")) ?
fbdo.to<bool>() ? "true" : "false" :
fbdo.errorReason().c_str());
//
//    bool bVal;
//    Serial.printf("Get bool ref... %s\n",
Firebase.RTDB.getBool(&fbdo, F("/test/bool"), &bVal) ? bVal ?
"true" : "false" : fbdo.errorReason().c_str());
//
//    Serial.printf("Set int... %s\n",
Firebase.RTDB.setInt(&fbdo, F("/test/int"), count) ? "ok" :
fbdo.errorReason().c_str());
//
```

```
//    Serial.printf("Get int... %s\n",
Firebase.RTDB.getInt(&fbdo, F("/test/int")) ?
String(fbdo.to<int>()).c_str() : fbdo.errorReason().c_str());
//
//    int iVal = 0;
//    Serial.printf("Get int ref... %s\n",
Firebase.RTDB.getInt(&fbdo, F("/test/int"), &iVal) ?
String(iVal).c_str() : fbdo.errorReason().c_str());
//
//    Serial.printf("Set float... %s\n",
Firebase.RTDB.setFloat(&fbdo, F("/test/float"), count + 10.2)
? "ok" : fbdo.errorReason().c_str());
//
//    Serial.printf("Get float... %s\n",
Firebase.RTDB.getFloat(&fbdo, F("/test/float")) ?
String(fbdo.to<float>()).c_str() :
fbdo.errorReason().c_str());
//
//    Serial.printf("Set double... %s\n",
Firebase.RTDB.setDouble(&fbdo, F("/test/double"), count +
35.517549723765) ? "ok" : fbdo.errorReason().c_str());
//
//    Serial.printf("Get double... %s\n",
Firebase.RTDB.getDouble(&fbdo, F("/test/double")) ?
String(fbdo.to<double>()).c_str() :
fbdo.errorReason().c_str());
//
//    Serial.printf("Set string... %s\n",
Firebase.RTDB.setString(&fbdo, F("/test/string"), F("Hello
World!")) ? "ok" : fbdo.errorReason().c_str());
//
//    Serial.printf("Get string... %s\n",
Firebase.RTDB.getString(&fbdo, F("/test/string")) ?
fbdo.to<const char *>() : fbdo.errorReason().c_str());
//
//    // For the usage of FirebaseJson, see
examples/FirebaseJson/BasicUsage/Create_Edit_Parse.ino
//    FirebaseJson json;
//
//    if (count == 0)
//    {
//      json.set("value/round/" + String(count), F("cool!"));
//      json.set(F("value/ts/.sv"), F("timestamp"));
//      Serial.printf("Set json... %s\n",
Firebase.RTDB.set(&fbdo, F("/test/json"), &json) ? "ok" :
fbdo.errorReason().c_str());
//    }
//    else
//    {
//      json.add(String(count), F("smart!"));
//      Serial.printf("Update node... %s\n",
Firebase.RTDB.updateNode(&fbdo, F("/test/json/value/round"),
&json) ? "ok" : fbdo.errorReason().c_str());
//    }
```

```
//
//    Serial.println();
//
//    // For generic set/get functions.
//
//    // For generic set, use Firebase.RTDB.set(&fbdo, <path>,
<any variable or value>)
//
//    // For generic get, use Firebase.RTDB.get(&fbdo,
<path>).
//    // And check its type with fbdo.dataType() or
fbdo.dataTypeEnum() and
//    // cast the value from it e.g. fbdo.to<int>(),
fbdo.to<std::string>().
//
//    // The function, fbdo.dataType() returns types String
e.g. string, boolean,
//    // int, float, double, json, array, blob, file and null.
//
//    // The function, fbdo.dataTypeEnum() returns type enum
(number) e.g. fb_esp_rtdb_data_type_null (1),
//    // fb_esp_rtdb_data_type_integer,
fb_esp_rtdb_data_type_float, fb_esp_rtdb_data_type_double,
//    // fb_esp_rtdb_data_type_boolean,
fb_esp_rtdb_data_type_string, fb_esp_rtdb_data_type_json,
//    // fb_esp_rtdb_data_type_array,
fb_esp_rtdb_data_type_blob, and fb_esp_rtdb_data_type_file
(10)
//
//    count++;
//  }
//}

/** NOTE:
   When you trying to get boolean, integer and floating point
number using getXXX from string, json
   and array that stored on the database, the value will not
set (unchanged) in the
   FirebaseData object because of the request and data
response type are mismatched.

   There is no error reported in this case, until you set this
option to true
   config.rtdb.data_type_stricted = true;

   In the case of unknown type of data to be retrieved, please
use generic get function and cast its value to desired type
like this

   Firebase.RTDB.get(&fbdo, "/path/to/node");

   float value = fbdo.to<float>();
   String str = fbdo.to<String>();
```

```
*/

/// PLEASE AVOID THIS ////

// Please avoid the following inappropriate and inefficient
use cases
/**

   1. Call get repeatedly inside the loop without the
appropriate timing for execution provided e.g. millis() or
conditional checking,
   where delay should be avoided.

   Everytime get was called, the request header need to be
sent to server which its size depends on the authentication
method used,
   and costs your data usage.

   Please use stream function instead for this use case.

   2. Using the single FirebaseData object to call different
type functions as above example without the appropriate
   timing for execution provided in the loop i.e., repeatedly
switching call between get and set functions.

   In addition to costs the data usage, the delay will be
involved as the session needs to be closed and opened too
often
   due to the HTTP method (GET, PUT, POST, PATCH and DELETE)
was changed in the incoming request.


   Please reduce the use of swithing calls by store the
multiple values to the JSON object and store it once on the
database.

   Or calling continuously "set" or "setAsync" functions
without "get" called in between, and calling get continuously
without set
   called in between.

   If you needed to call arbitrary "get" and "set" based on
condition or event, use another FirebaseData object to avoid
the session
   closing and reopening.

   3. Use of delay or hidden delay or blocking operation to
wait for hardware ready in the third party sensor libraries,
together with stream functions e.g. Firebase.RTDB.readStream
and fbdo.streamAvailable in the loop.

   Please use non-blocking mode of sensor libraries (if
available) or use millis instead of delay in your code.
```

```
   4. Blocking the token generation process.

   Let the authentication token generation to run without
blocking, the following code MUST BE AVOIDED.

   while (!Firebase.ready()) <---- Don't do this in while loop
   {
       delay(1000);
   }

*/
```

KODE PROGRAM PERANGKAT LUNAK (APLIKASI)

1. Tampilan Menu Pertama

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center"
    android:background="@color/teal_700"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:orientation="horizontal">


        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginBottom="20dp"
            android:fontFamily="sans-serif-black"
            android:text="SMART GATE REMOTE"
            android:textColor="#FFFFFF"
            android:textSize="30sp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="350dp"
        android:layout_height="350dp"
        android:gravity="center"
        android:orientation="horizontal">

        <ImageView
            android:id="@+id/imageView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            app:srcCompat="@drawable/gate" />
    </LinearLayout>

    <LinearLayout
```

```
        android:layout_width="120dp"
        android:layout_height="100dp"
        android:layout_marginTop="20dp"
        android:orientation="horizontal">

        <Button
            android:id="@+id/btnMasuk"
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:backgroundTint="@color/white"
            android:text="ENTER"
            android:textColor="@color/teal_700"
            android:textSize="24sp"
            app:iconPadding="6dp" />
    </LinearLayout>

</LinearLayout>
```

2. Tampilan Menu Kedua

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/teal_700"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center"
    tools:context=".button">

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:foregroundGravity="center"
        app:srcCompat="@drawable/ic_launcher_foreground"
        tools:layout_editor_absoluteX="55dp"
        tools:layout_editor_absoluteY="34dp" />

    <Button
        android:id="@+id/buttonOpen"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:foregroundGravity="center"
        android:gravity="center"
        android:text="OPEN"
        android:textColor="@color/teal_700"
        android:textSize="36sp"
```

```
        tools:layout_editor_absoluteX="136dp"
        tools:layout_editor_absoluteY="274dp" />

    <Button
        android:id="@+id/buttonClose"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:gravity="center"
        android:text="CLOSE"
        android:textColor="@color/teal_700"
        android:textSize="34sp"
        tools:layout_editor_absoluteX="131dp"
        tools:layout_editor_absoluteY="397dp" />

    <Button
        android:id="@+id/buttonStop"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:backgroundTint="@color/white"
        android:foregroundGravity="center"
        android:gravity="center"
        android:text="STOP"
        android:textColor="@color/teal_700"
        android:textSize="36sp"
        tools:layout_editor_absoluteX="136dp"
        tools:layout_editor_absoluteY="274dp" />
</LinearLayout>
```

KODE PROGRAM PENGOLAHAN DATA

```
package com.mikacyuuu.smartgatem;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class MainActivity extends AppCompatActivity {

    Button btnMasuk;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnMasuk = findViewById(R.id.btnMasuk);
        btnMasuk.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View view) {
                Intent intent = new
Intent(MainActivity.this,button.class);
                startActivity(intent);
        });
}


package com.mikacyuuu.smartgatem;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class button extends AppCompatActivity {

    Button buttonOpen,buttonClose,buttonStop;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_button);

        FirebaseDatabase database =
FirebaseDatabase.getInstance();
```

```
        DatabaseReference myRef =
database.getReference("message");



        buttonOpen = findViewById(R.id.buttonOpen);
        buttonClose = findViewById(R.id.buttonClose);
        buttonStop = findViewById(R.id.buttonStop);
        buttonOpen.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

//                Intent intent = new Intent(button.this,);
                myRef.setValue(encrypt("buka",15));
            }
        });
        buttonStop.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
//                Intent intent = new Intent(button.this,);
                myRef.setValue(encrypt("tutup",15));
            }
        });
        buttonStop.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                myRef.setValue(encrypt("stop",15));
            }
        });
    }
    public static String encrypt(String message, int shift) {
        StringBuilder encryptedText = new StringBuilder();

        for (int i = 0; i < message.length(); i++) {
            char c = message.charAt(i);

            // Check if the character is a letter
            if (Character.isLetter(c)) {
                // Convert the character to uppercase
                c = Character.toUpperCase(c);

                // Shift the character by the specified amount
                c = (char) ((c - 'A' + shift) % 26 + 'A');
            }

            encryptedText.append(c);
        }

        return encryptedText.toString();
    }
}
```

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

KESEPAKATAN BIMBINGAN TUGAS AKHIR (TA)

Kami yang bertanda tangan dibawah ini,

Pihak Pertama

Nama	: Rahmika
NIM	: 061940352364
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pihak Kedua

Nama	: Irma Salamah, S.T., M.T.I.
NIP	: 197410221998022001
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pada hari ini Senin tanggal 29 Mei 2023 telah sepakat untuk melakukan konsultasi bimbingan Tugas Akhir (TA).

Konsultasi bimbingan sekurang- kurangnya 1 (satu) kali dalam satu minggu. Pelaksanaan bimbingan pada setiap hari kerja pukul jam kerja tempat di Politeknik Negeri Sriwijaya.

Demikianlah kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas Akhir.

Palembang, 29 Mei 2023

Pihak Pertama

(Rahmika)
NIM. 061940352364

Pihak Kedua

(Irma Salamah, S.T., M.T.I.)
NIP. 197410221998022001

Mengetahui
Ketua Jurusan Teknik Elektro

(Ir. Iskandar Lutfi, M.T.)
NIP. 196501291991031002

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

ISO 9001 Registered Quality Management
UKAS MANAGEMENT SYSTEMS

KESEPAKATAN BIMBINGAN TUGAS AKHIR (TA)

ami yang bertanda tangan dibawah ini,

Pihak Pertama

Nama	: Rahmika
NIM	: 061940352364
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

Pihak Kedua

Nama	: Ir. Ali Nurdin, M.T.
NIP	: 196212071991031001
Jurusan	: Teknik Elektro
Program Studi	: Sarjana Terapan Teknik Telekomunikasi

ada hari ini Senin tanggal 29 Mei 2023 telah sepakat untuk melakukan konsultasi bimbingan ugas Akhir (TA).

onsultasi bimbingan sekurang- kurangnya 1 (satu) kali dalam satu minggu. Pelaksanaan bimbingan ada setiap hari kerja pukul 7 s.d kerja tempat di Politeknik Negeri Sriwijaya.

emikianlah kesepakatan ini dibuat dengan penuh kesadaran guna kelancaran penyelesaian Tugas khir.

Palembang, 29 Mei 2023

ihak Pertama

(Rahmika)
IM. 061940352364

Pihak Kedua

(Ir. Ali Nurdin, M.T.)
NIP. 196212071991031001

Mengetahui
Ketua Jurusan Teknik Elektro

(Ir. Iskandar Lutfi, M.T.)
NIP. 196501291991031002

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK ELEKTRO
Jalan Srijaya Negara Bukit Besar - Palembang 30139 Telepon (0711) 353414
Laman : http://polsri.ac.id, POS El: info@polsri.ac.id

LEMBAR BIMBINGAN TUGAS AKHIR

Nama : Rahmika
NIM : 061940352364
Jurusan/Program Studi : Teknik Elektro / Sarjana Terapan Teknik Telekomunikasi
Judul Tugas Akhir : Rancang Bangun Pagar Otomatis Berbasis *Internet of Things* (IoT)

Pembimbing I : Irma Salamah, S.T., M.T.I
NIP. 197410221998022001

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
1.	20/03/23	BAB I Pendahuluan	
2.	28/03/23	Revisi BAB I & BAB II	
3.	5/06/23	ACC Bab I / BAB II	
4.	17/06/23	Revisi BAB III	
5.	1/07/23	ACC BAB III BAB IV	
6.	6/07/23	Revisi BAB IV	

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
7.	9/7/23	Revisi BAB IV	
8.	14/7/23	Revisi pembahasan BAB IV Data Pengujian	
9.	17/7/23	Revisi BAB IV Pembahasan Analisis	
10.	20/7/23	ACC BAB IV Konsul Jurnal	
11.	21/7/23	Submit Jurnal	
12	29/7/23	Revisi Jurnal ACC BAB I/IV	
13	31/7/23	ACC BAB I/V	
14	3/8/'23	Acc. Siap Ujian	
15			

Palembang, Agustus 2023

Koordinator Program Studi
Sarjana Terapan Teknik
Telekomunikasi

Lindawati, S.T., M.T.I.

NIP. 197105282006042001

No. Dok. : F-PBM-17
No. Rev. : 00

Tgl. Berlaku : 13 Desember 2010

Catatan:
Ketua Jurusan/Ketua Program Studi harus memeriksa jumlah pelaksanaan bimbingan sesuai yang dipersyaratkan dalam Pedoman Laporan Akhir sebelum menandatangani lembar bimbingan ini.
Lembar pembimbingan LA ini harus dilampirkan dalam Laporan Akhir.

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
JURUSAN TEKNIK ELEKTRO
Jalan Srijaya Negara Bukit Besar - Palembang 30139 Telepon (0711) 353414
Laman : http://polsri.ac.id, POS El: info@polsri.ac.id

LEMBAR BIMBINGAN TUGAS AKHIR

Lembar : 1

Nama : Rahmika
NIM : 061940352364
Jurusan/Program Studi : Teknik Elektro / Sarjana Terapan Teknik Telekomunikasi
Judul Tugas Akhir : Rancang Bangun Pagar Otomatis Berbasis *Internet of Things* (IoT)

Pembimbing II : Ir. Ali Nurdin, M.T

NIP. 196212071991031001

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
1.	20-5-23	Bab I Pendahuluan	
2.	28-5-23	Revisi Bab I dan Bab II	
3.	5-6-23	ACC Bab I / BAB II	
4.	17-6-23	Revisi BAB III	
5.	1-7-23	ACC BAB III BAB IV	

6.	6-7-23	Revisi BAB IV	

Lembar: 2

No.	Tanggal	Uraian Bimbingan	Tanda Tangan Pembimbing
7.	9-7-23	Revisi Bab IV, Aplikasi	
8.	14-7-23	Revisi Pembahasan BAB IV Data Pengujian	
9.	17-7-23	Revisi BAB IV Pembahasan Analisis.	
10.	20-7-23	ACC BAB IV Konsul Jurnal	
11.	21-7-23	Submit Jurnal	
12	23-7-23	Revisi Jurnal	
13	29-7-23	ACC BAB I/IV	
14	31/7-23	ACC BAB I/V	
15	3-8-23	ACC Jurnal ACC Sidang TA	

Palembang, Agustus 2023

Koordinator Program Studi
Sarjana Terapan Teknik
Telekomunikasi

Lindawati, S.T., M.T.I.

NIP. 197105282006042001

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139
Telp. 0711-353414 fax. 0711-355918
Website : www.polsri.ac.id E-mail : info@polsri.ac.id

REKOMENDASI UJIAN TUGAS AKHIR

Pembimbing Tugas Akhir memberikan rekomendasi kepada,

Nama	:	Rahmika
NIM	:	061940352364
Jurusan/Program Studi	:	Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi
Judul Tugas Akhir	:	Rancang Bangun Pagar Otomatis Berbasis *Internet of Things* (IoT)

Mahasiswa tersebut telah memenuhi persyaratan dan dapat mengikuti Ujian Tugas Akhir (TA) pada Tahun Akademik 2023/2024

Palembang, Agustus 2023

Pembimbing I,

Irma Salamah, S.T., M.T.I
NIP. 197410221998022001

Pembimbing II,

Ir. Ali Nurdin, M.T
NIP. 196212071991031001

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN,
RISET DAN TEKNOLOGI
POLITEKNIK NEGERI SRIWIJAYA
Jalan Srijaya Negara, Palembang 30139 Telp. 0711-353414
Laman: http://polsri.ac.id, Pos El : info@polsri.ac.id

PELAKSANAAN REVISI TUGAS AKHIR

Mahasiswa berikut,

Nama : Rahmika

NIM : 061940352364

Jurusan/Program Studi : Teknik Elektro/Sarjana Terapan Teknik Telekomunikasi

Judul Tugas Akhir : RANCANG BANGUN PAGAR OTOMATIS BERBASIS *INTERNET OF THINGS* (IOT)

Telah melaksanakan revisi terhadap Tugas Akhir yang diujikan pada hari .... tanggal ... bulan ..... tahun ..... . Pelaksanaan revisi terhadap Tugas Akhir tersebut telah disetujui oleh Dosen Penguji yang memberikan revisi:

No.	Komentar	Nama Dosen Penguji*)	Tanggal	Tanda Tangan
1.	Acc	Ir. Ali Nurdin NIP 196212071991031001	28/8-23	
2.	Acc	Irma Salamah, S.T.,M.T.I NIP 19741021998022001	28/8	
3.	Acc	Nasron S.T., M.T NIP 196808221993011001	22/8-2023	
4.	ACC	Suzanzefi, S.T., M.Kom NIP 197709252005012003	24/8-023	

Palembang, ...
Ketua Penguji **),

(Ir. Ali Nurdin) NIP
196212071991031001

Fwd: Accepted ELKHA

Kotak Masuk

Jurnal Elkha 24 Agu

kepada saya, Irma

Yth,
Penulis Jurnal :
Irma Salamah

Memberitahukan bahwa Artikel Jurnal ID : **#67895**
Dengan Judul : " **Operate the Internet Of Things (IOT) Technology Control System on Automatic Fences**"
Diterima/Accepted, dan akan dilanjutkan proses editing (***sedang berlangsung***) selanjutnya untuk **mempublikasikan Artikel Anda pada Jurnal ELKHA Vol .15 No.2, Oktober 2023**, Segera melengkapi persyaratan berikut :

1. **Anda wajib** membayar Biaya **Publikasi** Artikel sebesar **Rp. 350.000**. dan **Rp 150.000 Cetak Buku Jurnal** (***hardcopy***) > (***Transfer ke Reg. BNI BANK 0187818727 An 'Supriono***)
*Buku Jurnal (hardcopy) akan dikirim ke alamat penulis sesuai yang tertera pada Author Statement.

2. Anda harus mengunduh, mengisi AUTHOR STATEMENT, dan kembalikan Hasil scan (***Jika belum ada***) dan ***Bukti BANK TRANSFER SLIP*** melalui email : jurnal.elkha@untan.ac.id
2.1 AUTHOR STATEMENT Link : Author Statement - ELKHA 2022.docx - Google Dokumen

Journal Teknik Elektro, Vol. xx No.yy., Month 20xx, pp. 1-2
ISSN: 1858-1463 (print), 2580-6807 (online)

Innovation in Smart Fencing with Internet of Things (IoT) Technology for Ease of Use

[1,2] Department of Electrical Engineering, Politeknik Negeri Sriwijaya Palembang, Indonesia
Corresponding Email: *) irma.salamah@yahoo.com

The fence is a structure that is deliberately ed to limit or protect the house. The fence is made high quipped with a safety lock to protect the home so only wner can enter. The fence serves to provide protection ock views so that the house becomes safer. It is usually ed manually by hand to open and close the fence. This nsidered very inconvenient and time-consuming. This em underlies the idea of creating an automatic fence ing and closing system using a smartphone by utilizing net of Things (IoT) technology in order to facilitate n work to open and close the fence, as well as reduce ffort and time required when opening and closing the when manually driven. A smartphone is used as a MCU controller contained in the ESP8266 WiFi ule via an internet connection to connect to the Android le application. To move the fence using an electric r as the driving force which is connected directly to MCU. When the fence opens and closes, the relay will ate and activate to give orders to the electric motor to the fence. The fence will stop when the switch opens the switch close responds, the motor will stop moving. ime data results are stored in Firebase and the results design are expected that all components are connected erly so that automatic fences can be used. This earch is expected to make a valuable contribution to the lopment of science and improve the quality of life of the munity by using advanced technology that responds to ed for more intuitive and efficient access to fences.

Keywords: Electric Motor, Electric Fences, Internet of Things (IoT), NodeMcu.

I. INTRODUCTION

he fence is the main part of a house in terms of rity. The fence is an element part of the design of a e that is important for its existence[1]. The fence has nction as a barrier or safety designed to limit or ent movement across the boundary it makes [2]. es are usually made of an access point by opening the by pushing it to the right or left[3]. The fence used is ding fence that moves and runs on rails with wheels ted on the bottom of the fence. To open and close fence must be done by moving the door by hand, wever, if a vehicle such as a car wants to go out and the house you have to get off the vehicle first, and if ng certain conditions such as rain opening and closing te of the house, it will be very troublesome and time-uming. This problem underlies the idea of making en open and close automatically using a remote and rtphone (android) by utilizing Internet of Things (IoT) hnology. Where the system used contains an arrangement of infrastructure that is integrated with each other so that it can make human work more effective and efficient, the Internet of Things (IoT) runs by utilizing the use of smart devices and internet networks.

The Internet of Things (IoT) is a network of physical objects. The internet is not just a network of computers but has developed into a network of devices of all types and sizes, vehicles, smartphones, household appliances, toys, cameras, medical equipment and industrial systems, animals, people, buildings, everything connected, everything communicating & information sharing based on established protocols to achieve intelligent reorganization, positioning, browsing, security, control, and even personal real-time online monitoring, online upgrade, process control, and administration[4]. The importance of the Internet of Things can be seen by its increasing application in various lines of life today. IoT gives us many ideas to participate in various aspects of development from micro to macro around the world [5]. IoT must be able to provide information about environmental conditions and control electronic devices in real time, therefore using Firebase as a database capable of sending data in real time [6].

In an increasingly connected world and as technology evolves, gates are becoming more than just physical boundaries that separate spaces [7].The need for secure, fast and convenient access is increasing, driving innovations in the form of smart fence innovations with IoT technology for user convenience. The sophistication of Internet of Things (IoT) technology offers an exciting and effective way to address these challenges. The combination of connected hardware and intelligent software allows the fence to interact with its environment and respond to user needs in real-time. This means that users can control their gate with ease, whether they are nearby or not.

By addressing existing problems, this research also brings a new concept to the world of fence access, providing a more practical, faster and safer solution. By applying this technology, the innovation relies on technical sophistication. It ensures that the user experience remains easy to access, even for those with a deep technical background.

The results of this research not only reflect a new technological solution but set our sights on a more connected future. This innovation contributes not only to our understanding of how technology can play an important role in providing real solutions but also to our

Manuscript received 20xx-xx-xx; revised 20xx-xx-xx; accepted 20xx-xx-xx

derstanding of how technology can be used to improve r lives.

Based on researchers' research of automatic gate stems [8], this system uses RF 315 wireless remote ntrol. This system has a weakness, namely the use of mote control is very limited. If the homeowner leaves remote control behind, the homeowner cannot enter house.

Researchers [9] researched automatic garage door ntrol systems. This system uses RFID indicators and crocontroller based alarms. The weakness of this stem is that the RFID tag is unreadable or has an error, the garage door cannot be opened.

Meanwhile, based on research [10] on an automatic e control system using digital image processing of hicle license plates. This control system also has rbacks, namely the vehicle license plate detection sor can only be used with a maximum distance of 20

The Smart Fence Innovation with IoT Technology for r Convenience presents several novelties and antages that distinguish it from other research in this d is an innovation designed with a focus on user nvenience, the use of Internet of Things (IoT) hnology, which shows that the fence is not just a static d, but is actively connected to the surrounding vironment and can respond to various situations, the e of remote access which allows users to control the ce remotely via smartphone which provides antages in terms of convenience. In the era of ever-olving technology, innovation continues to be the main ver to meet society's demands for convenience and ficiency. In this context, this research aims to realize a nificant step forward by developing "Smart Fence novation with IoT Technology for User Convenience". combining revolutionary Internet of Things (IoT) hnologies with the need for easy access, this research sents a solution that stands out by providing a more uitive and responsive door access experience. In this roduction, we will explore the novelty and key benefits this innovation and illustrate how this research nificantly contributes to the development of science d the welfare of society through the use of advanced hnology.

Based on these studies, the authors designed an tomatic fence control system based on the Internet of ings (IoT) with the aim of facilitating human work in ening and closing fences and protecting homeowners to main safe when opening and looking at the fence at ght or when bad weather. This study aims to design tomatic fences using IoT technology by answering the search question of whether automatic fences can be signed based on IoT. The research analysis is that nces can be designed and used using IoT technology talled on smartphones. This system uses NodeMCU as controller for the gate. NodeMCU Esp8266, is an IoT vice or wifi module that can be integrated with devices be controlled and monitored via the internet network H[12]. An electric motor as the drive for the gate, and an Android smartphone as a device that orders the fence to open or close directly connected to the internet. All devices are connected using the internet with the Android Studio application.

II. Methodology

A. Research Stages

The following section presents the architecture of the Internet of Things (IoT) Based Automatic Fence Control research. How to design the entire system can be seen in the image below.

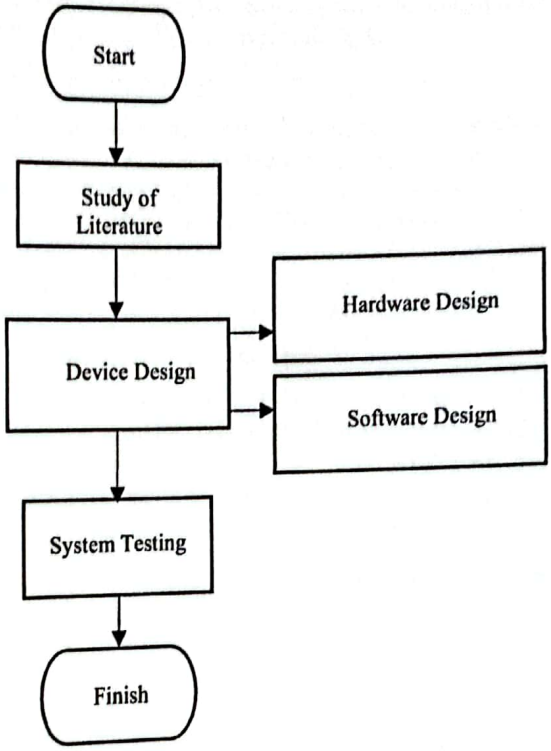

Figure 1. Research Architecture Diagram of Automatic Fence Control System

Figure 1 is the overall flowchart of the research methodology step framework for Internet of Things (IoT) Based Automatic Fence Design. In this research, a literature study was carried out by collecting and taking references from various sources such as books, theses, journals, and the internet, where these sources will become a reference for this research.

Hardware design is the process of designing all components until they are all connected to the fence. The hardware design starts with creating a schematic. The NodeMCU, which uses the ESP8266 WiFi module, is connected to a 24-volt power supply as the power source. The 4-channel relay module helps control the opening and closing of the fence when it receives commands from the NodeMCU. The relay will activate when it receives a command, so the motor driver will rotate and move the fence to open and close. The electric motor used is an AC-type electric motor because this type of motor is stronger to move the large weight of the fence. The motor driver

stop moving when the open switch and close switch ...ound, where the switch functions to disconnect the ...ement of the motor wheel, then the motor driver will ...s, and the fence will automatically be stationary. The ...ware design can be seen in the picture below in Figure

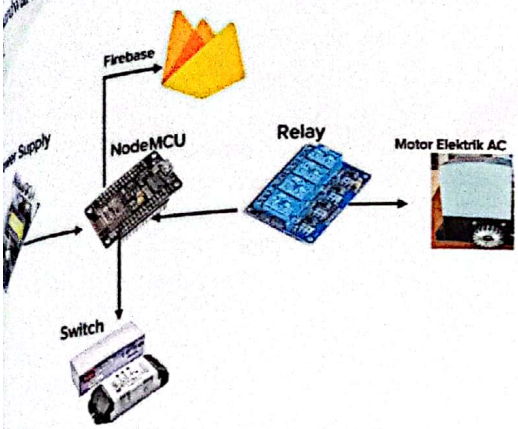

Figure 2. Hardware Design

Table 1. Hardware Component Specifications

Component Name	Description
Motor Elektrik AC	a. Power AC b. Motor Speed 1400rpm c. Max Gate Weight 1000kg d. Gate Speed 12m/min e. Temperature -20°C - 55°C f. Protecting Class IP44
NodeMCU	NodeMCU ESP8266
Power Supply	Power Supply 24V
Relay	Modul Relay 4 Channel
Switch	Switch Open & Close

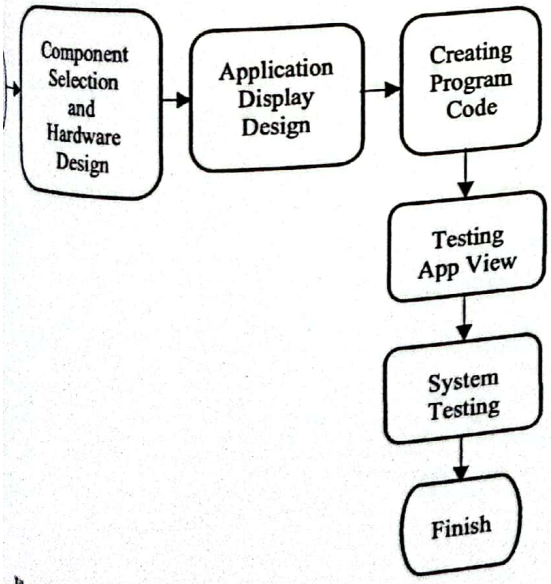

Figure 3. Automatic Fence Control Software Design

Figure 3 is a framework that will be executed in software design that aims to produce a fence control application design. The stages include:

a. Component selection and hardware design

The first step is to select the components that will be used for the hardware design. These components include a 24V power supply, NodeMCU ESP8266, 4-channel relay module, AC electric motor, open switch, and close switch.

a. Application display design

The application display design aims to make the application display design easier to make. In designing the application is designed using Android Studio.

b. Create program code

The hardware program code is created on the Arduino IDE using the C++ programming language, and the application program code is created in Android Studio using C++ and Java. The hardware program code is connected through Firebase as a data processor, and the data is sent to NodeMCU. NodeMCU reads the sent data and gives incoming commands at the same time.

c. Test the appearance of the application

Testing the application display is to test each option that is available with the features that have been designed then, the application is successful if all the features that have been designed are used successfully.

d. System testing

This section includes system testing related to tools and applications that have been designed and successfully tested.

System testing in this design includes application testing, which aims to see if application commands can operate the fence. Testing the electric motor wheel movement function to see if it works as expected. Distance and network speed testing aim to see how the fence speeds up at different distances and network speeds. Finally, see the results of power consumption used in testing the system. System testing first made a flowchart to describe the program's course against the system. The picture below is a flowchart process in testing the system to be made. The first step is to start and initialize any tools or components used. If the fence is open, the motor will move until the open switch command responds (yes), and the motor will stop automatically. The motor will move if the fence is closed until the Close switch command (yes) responds. Then, the motor will stop, and the fence will close. If the open or closed switch does not respond, the fence open/close command is repeated. The flowchart is shown in Figure 3.

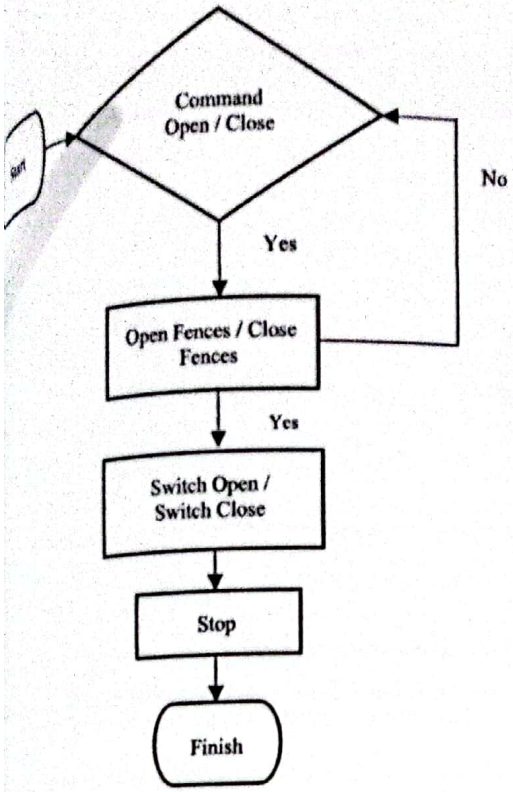

Figure 4. Complete System Experiment Flowchart

In this test, the component will be placed near the fence, ere it will be tested in its entirety by the automatic ce opening and closing test. Then several variables re measured which included voltage, current, electric or speed, and the electric power used. As well as the ation and time needed by the fence when opening and sing. Test data will be stored in real-time in Firebase ng the Arduino Caesar Chiper method (random writing hod) which is confidential so that incoming data is not wn to other people.

III. Results and Discussion

The results of this research are based on components sembled and programmed based on the Internet of ings (IoT) using smartphones with applications made h Android Studio. In this design, using a type of iron ce sliding, this fence weighs about 500 kg with a length 5.25 meters. This tool was designed as a set of mponents consisting of a 24-volt power supply as a wer source for the device, NodeMCU as a component at is connected by the application using the ESP8266 fi module and instructs the device. Nodemcu is nnected to Firebase to execute commands in the plication. The relay is turned on when it receives a mmand to move the electric motor. The type of relay ed is a 4-channel relay module. The electric motor used e AC motor that can move the fence with a maximum ight of 1000 kg at a motor speed of 1400 rpm. AC ectric motors are used because the fence to be installed ighs approximately 500 kg, so it is safe and avoids erloading the motor. When the motor is moving, the nch performs its function to disconnect or stop the vement of the motor wheel when the fence is in motion. Figure 5 and Figure 6 below are the research results when the fence opens and closes.

Figure 5. Fence in Open State

Figure 6. Fence in Closed State

A. Hardware Program Code Section

The program code section in Figure 7 is a hardware interface program that uses the Arduino IDE.

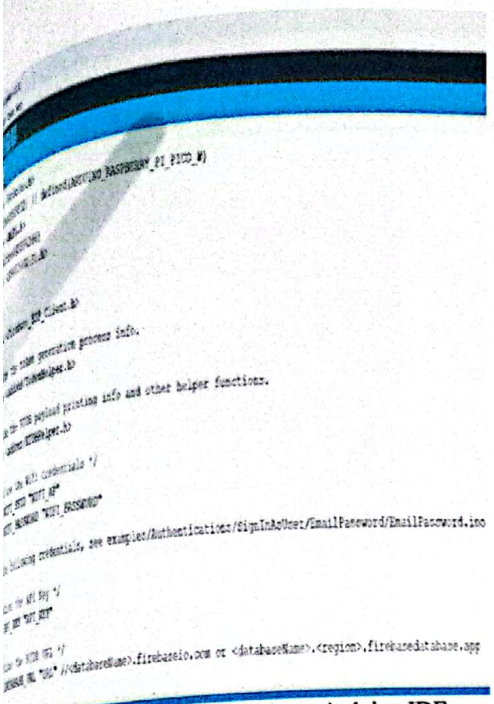

Figure 7. Hardware Program Code on Arduino IDE

igure 7 NodeMCU is programmed in C++ using WiFi a WiFi Password to connect to the internet. The data will be managed using Firebase which is grammed on the Arduino IDE, then the data will be ed in real-time.

Application Programming Code in Android Studio

his section is a program component created to design appearance of the application to make it easier to trol the tool to be tested. The application display page signed in Android Studio with a view menu whose ures have their respective functions. The application lay program code uses the C++ and Java gramming languages.

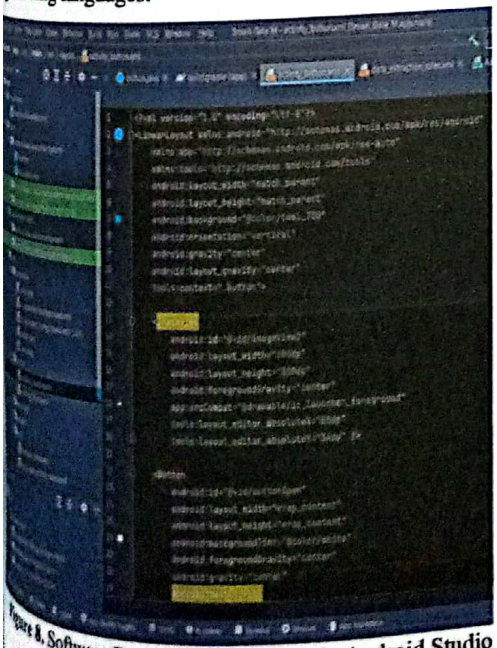

Figure 8. Software Design Program Code in Android Studio

In Figure 8 several program codes are entered for the application display design in the form of open menus and closed menus. <LinearLayout which is for creating a layout for the element to be used. <ImageView is the program code when adding image elements to the application view you created. <Button is the program code on a pushable element or button element.

C. Application Display Page Results

In this section are the results of the design program for making automatic fence control applications. The design is made with a menu page that can be used for fence controllers to open and close, data obtained in real-time will be entered in Firebase. Applications that have been designed will be transferred to smartphones that are directly connected to the internet network. The display menu on the application installed on the smartphone can be used based on the features and functions that have been made. Smartphone (android) must be connected to the internet when running the system. The commands on the fence cannot be executed if the application is not connected to the internet. Suppose the internet is experiencing bad signal interference due to weather. In that case, the possibility is to wait for the internet to reconnect or the state of the internet in a stable connection. The picture below is the result of displaying the designed application. The difference between the two is the location of the application designed in Android Studio, and then its use is moved into the smartphone as a controller. Both have the same menu because the results of displaying the application on the smartphone are the results of designing in Android Studio, which is then moved into the smartphone.

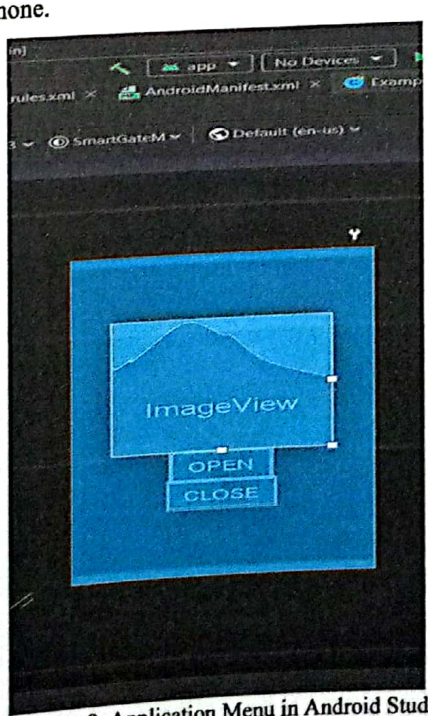

Figure 9. Application Menu in Android Studio

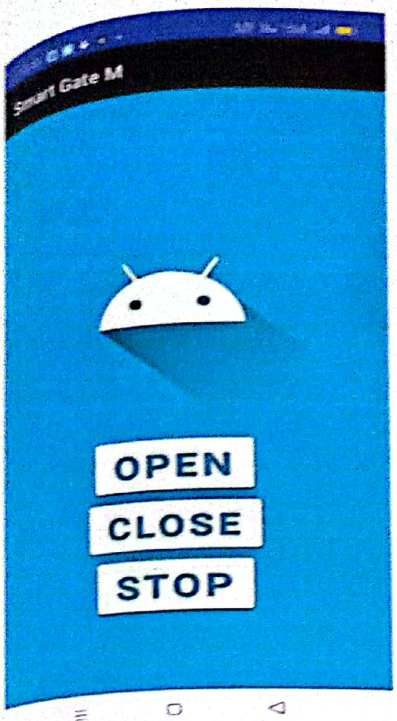

Figure 10. Application Menu on Smartphone

re 9 is the display result in Android Studio with image ams and buttons that have been designed with ams in Android Studio. Figure 10 is the display of the application on a smartphone which has been for each feature based on its function before being on the fence. The results of testing the application ion can be seen in Table 1 below.

Table 2. Application Testing Results

mmand	Required Time (Second)	Results Description
Open	23,74	Succeed
Close	23,67	Succeed
Stop	0	Succeed

ed on Table 1 the results of application testing were cessful with different time descriptions. To open, ing the fence takes 23.74 seconds until the fence stops ing. Meanwhile, to close, moving the fence requires e time than opening it, which is 23.67 seconds. The e immediately stops moving and stays still for 0 nds. The fence immediately stops moving and stays for 0 seconds. The test of the application function is cessfully executed on each command, which then n you want to control the fence, use the application alled on a smartphone (Android) using an Internet nection. At the time of testing, the Internet connection ers did not experience any interference or was stable, the test was performed as intended. If the Internet ections are interrupted or not connected, what happens e you have to wait for the Internet to reconnect to e commands to run the fence control.

D. *System Testing Results*

In this section, data is the result of testing the Internet of Things (IoT) Based Automatic Fence Control System, where the system being tested is the performance of the tool that has been designed, the results of the fence control distance test using a smartphone connected to an internet connection, as well as the results of electricity consumption used on when testing the system. Incoming data will be sent to Firebase with the letter randomization method.

Table 3. Electric Motor Test Result

Wheel Movement	Wheel Spin	Result Description
Onward	Spin Forward	Corresponding
Back Off	Spinning Back	Corresponding
Stop	Not Moving	Corresponding

Table 2 is the result of the electric motor drive function test. Based on Table 2 the movement of the electric motor is successful according to its function, where when the wheel is forward the wheel rotates forward, while when the wheel is backward the wheel rotates backward, when the wheel stops moving the wheel will stop and stop moving.

Table 4. Test Fence Controller Distance with Network Speed

Range	Network Speed	Fence Speed	Description
1 Meter	2,00 Kbps	Normal	Succeed
2 Meter	0,30 Kbps	Normal	Succeed
3 Meter	0,26 Kbps	Normal	Succeed

Table 3 shows that the fence control distance test was successfully performed at different network speeds. The relationship between distance and kbps (kilobits per second) can affect system performance and responsiveness when testing automated Internet-connected fences. While physical distance may not have a direct impact on kbps speed, there are ways in which distance can interact with Internet connection performance in this context, namely signal quality. Whereas the distance between the device and the wireless access point increases, the wireless signal quality may decrease. Poor signal quality can result in a decrease in kbps speed and even dropouts. The Auto-Fence system response may also be affected if the connection becomes unstable. The speed of the gate has no effect at short or

distances. Network speed in a fast or slow signal state not cause the gate speed to change. The network is very low because the application does not much Internet network. To see the network it was recorded on the smartphone (Android) used the system.

Table 5. Result of Electricity Consumption

Fence Condition	Voltage (Volt)	Current (Ampere)	Power (Watt)
Normal	211	0	0
Open	211	3,4	717,4
pen 1/2	211	3,0	633
pen 1/4	211	3,3	696,3
Close	211	7,8	1645,8
lose 1/2	211	7,8	1645,8
lose 1/4	211	7,4	1561,4

e 4 shows the power consumption used during the n test. The source voltage is 211 volts, measured a multimeter, and the current output differs. The nt used to open the fence is lower than the current to close the fence, in this case, because the ment of the fence is heavier when the command is to than when it is to open. The power consumption s from 633 to 1645 watts depending on the output nt and voltage of the source. Power calculation is d on the principle of Ohm's law [13], where the ula is as follows:

$$P = V \times I \quad (1)$$

ription: P = Power (watt)

V = Voltage (volt)

I = Current (ampere)

oltage results do not change, and the current results ge because the electric motor used in this design is an ype electric motor. AC type electric motor is an ic motor that is driven by alternating current or ating current (AC), where AC voltage is a voltage whose result is the same as the current, which is always changing and alternating [14].

Incoming data will be immediately sent to Firebase in real time. For data security, use the Arduino caesar chiper method (random letter method) so that if the incoming data is unknown or confidential. The results of the data sent can be seen in Figure 11 and Figure 12.

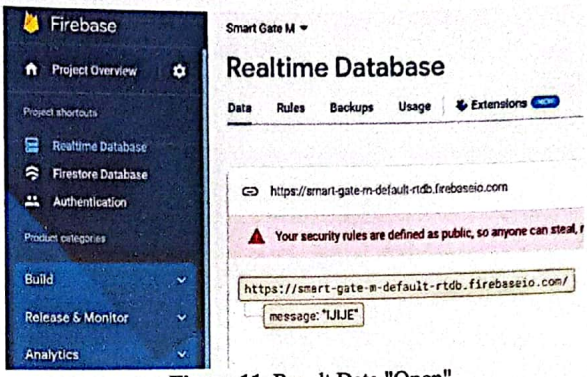

Figure 11. Result Data "Open"

Figure 11 is the result of the data when the fence moves open with the message: "IJIJE". Commands are entered through the application by pressing the open button then the motor moves the fence and real-time data results are sent to Firebase.

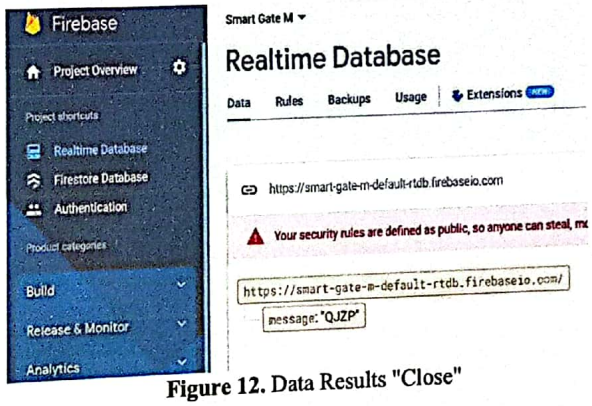

Figure 12. Data Results "Close"

Figure 12 is the result of the data when the fence moves closed with a message: "QJZP". Commands are entered through the application by pressing the close Button then an electric motor moves the fence and real-time data results are sent to firebase.

IV. CONCLUSION

In this design, it has successfully developed an automatic fence control system using Internet of Things (IoT) technology, with the advantage of being able to open the fence via a smartphone (android), which can be controlled remotely without having to push with your hands and walk towards the fence manually. Based on the results of the design and testing of an Internet of Things (IoT)-based automatic fence control system, it can be concluded that various components can be used properly and work according to their functions. The results of the data tested as a whole are appropriate and successful. The movement of the fence either opens, closes, or stops

ording to the expected test. The electric motor moves fence, the wheels rotate forward when the wheel es forward, the wheels rotate backward when the el moves backward, and the wheels will automatically when the electric motor moves stop. By using the ication, opening and closing the automatic gate can ed using an application installed on a smartphone that nnected to an internet connection. Applications that designed according to and successfully tested each re that is in the application display. Testing the nce of the controller with an internet connection does change the movement of the fence. Move the fence ing normally from a distance of 1 meter or 3 meters. m testing uses electric power consumption ranging 633-1645 watts. With a source voltage of 211 volts n opening current of 3.0-3.4 amperes, and a closing nt of 7.4-7.8 amperes. We hope that the results of our rch can be useful as well as learning. In future rch, it is expected to be able to develop an Internet of gs (IoT)-based automatic fence control system that e developed with a variety of other methods and is l according to its function.

ACKNOWLEDGEMENT

e authors thank the automated fencing research p for their support and cooperation. Special thanks to Department of Electrical Engineering, State echnic of Sriwijaya. And thanks to all those who helped both morally and materially.

REFERENCES

C. Widiasari, P. A. Sianipar, and M. Diono, "Journal of Caltex Polytechnic of Riau Automatic Control System of Home Fence Based on Internet of Things (IoT)," vol. 8, no. 2, pp. 162-174, 2022.

R. Dian Saraswati and M. Damiana Nestri Kiswari, "Fence from Territory to Pride and Identity," J. Design Commun. Vis. and New Media, vol. 1, no. 2, pp. 65-71, 2019, [online]. Available: http://journal.unika.ac.id/index.php/tuturrupa/article/view/1949

M. Massikki, A. Imran, M. Hamid, A. M. Afif, and M. Yantahin, "Development of Automatic Turnstile Driver Based on Arduino Uno Atmega 328P Microcontroller," J. Media Elektr., vol. 18, no.3,p.43,2021,doi: 10.26858/metrik.v18i3.25883

O. Vermesan and P. Friess, Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems (River Publishers Series in Communications), no. January. 2013. [Online]. Available at: https://riverpublishers.com/book_details.php?book_id=176%0Ahttp://files/12127/vermesan_et_al_2014_internet_of_things_-_converging_technologies_for_smart_environments_and.pdf

[5] F. Susanto, N. Komang Prasiani, and P. Darmawan, "Implementation of Internet of Things in Daily Life," J. IMAGINE, vol. 2, no. 1, pp. 2776-9836, 2022, [Online]. Available at: https://jurnal.std-bali.ac.id/index.php/imagine

[6] I. N. B. Hartawan and I. W. Sudiarsa, "Performance Analysis of Internet of Things Based on Firebase Real-Time Database," J. Resist. (Computer Systems Engineering), vol. 2, no. 1, pp. 6-17, 2019, doi: 10.

[7] Artikelpendidikan.id, "Internet of Things: Connecting the World through Technology," 2023. https://artikelpendidikan.id/apa-yang-dimaksud-dengan-internet-of-things/

[8] A. Hanafie, S. Suradi, S. Susilawati, and H. Hasmirawati, "Design of Automatic Fence Door System Using Wireless Remote Control Rf 315," ILTEK J. Teknol., vol. 15, no. 2, pp. 87-90, 2020, doi: 10.47398/iltek.v15i2.525.

[9] E. Efrizon, H. Herizon, and W. R. Dinata, "Design of an Automatic Garage Door Control System with RFID Indicator and Microcontroller-Based Alarm," Electron J. Ilm., vol. 9, no. 2, pp. 19-24, 2017, doi: 10.30630/eji.9.2.91.

[10] S. Alam, F. Fauzi, G. Tjahjadi, and R. Saputro, "Design of an Automatic Turnstile Control System Based on Digital Image Processing of Vehicle Number Plates Using Optical Character Recognition (OCR) Method," vol. 15, no. 2, pp. 92-100, 2022.

[11] P. D. Lestari, L. Karlitasari, and S. Maryana, "IoT (Internet of Things) Based Gate Controller," Business and Comput., vol. 1, no. 2, pp. 62-69, 2021, [Online]. Available at: http://www.jubikom.unpak.ac.id

[12] R. B. Indak, "Android and Fingerprint Based Automatic Door Opening and Closing System," J. Cosphi, vol. 2, no. 1, pp. 26-29, 2018, [Online]. Available at: https://www.cosphijournal.unisan.ac.id/index.php/cosphihome/article/view/100%0Ahttps://www.cosphijournal.unisan.ac.id/index.php/cosphihome/article/viewFile/100/55

[13] F. Savira and Y. Suharsono, "Heating elements," J. Chem. Inf. Model. vol. 01, no. 01, pp. 1689-1699, 2013.

[14] U. N. Cendana, "ELECTRIC MOTORS," no. April, 2018.