

BAB III METODE PENELITIAN

3.1 Perangkat Lunak

Dalam melakukan penelitian (Jembatani) Pada Pengembangan Model Aplikasi *Smart Farming* Berbasis *Internet Of Things* ada beberapa perangkat lunak yang dibutuhkan. Daftar perangkat lunak yang dibutuhkan pada penelitian ini dituliskan pada tabel 3.1 berikut.

Tabel 3. 1 Perangkat Lunak yang dibutuhkan

No.	Nama Software	Spesifikasi	Fungsi
1	Figma	Desktop	Sebagai alat dalam pembuatan desain antarmuka dan prototipe aplikasi
2	Arduino IDE	Desktop	Alat dalam menjalankan kode mikrokontroler dan mengintegrasikan pada aplikasi
3	Android Studio	Desktop	Alat yang digunakan dalam pengembangan antarmuka yang kompatibel dan berjalan pada sistem operasi Android

3.2 Perangkat Keras

Perangkat keras adalah peranti-peranti secara fisik yang digunakan dalam penelitian dan pengembang. Pada penelitian ini perangkat keras yang digunakan untuk menghubungkan dengan aplikasi yang diberi nama Jembatani berupa perangkat elektronika dan sensor yang dapat mengirimkan data.

3.2.1 Alat

Alat adalah instrumen yang digunakan peneliti dalam melakukan penelitian dan pengembangan untuk mengintegrasikan perangkat keras dengan aplikasi Jembatan pada pengembangan model aplikasi *smart farming* berbasis *Internet Of Things*. Berikut penjelasan tabel 3.2 terkait alat yang dibutuhkan pada penelitian.

Tabel 3. 2 Alat Yang Dibutuhkan

No.	Nama Alat	Spesifikasi	Fungsi	Jumlah
1	Komputer/ laptop	Windows 10	Untuk menjalankan kode dan integrasi perangkat keras dengan aplikasi	1 Unit
2	Solder	Pemanas komponen	Untuk merangkai komponen elektronika menjadi satu rangkaian	1 Unit
3	<i>Multitester</i>	Alat ukur daya	Digunakan untuk mengukur parameter listrik pada komponen elektronika, seperti tegangan, arus dan sebagainya	1 Unit
4	Tang potong	–	Digunakan untuk memotong kabel	1 Unit
5	<i>Breadboard</i>	Sirkuit elektronika	Digunakan dalam membangun prototipe elektronika sementara.	1 Unit

3.2.2 Komponen

Komponen perangkat elektronik fisik yang merupakan unsur penting dalam membuat rangkaian monitoring dan kendali pada pengembangan *smart farming*. Berikut adalah penjelasan komponen yang dibutuhkan dalam pengembangan sistem monitoring dan kendali pada tabel 3.3.

Tabel 3. 3 Komponen Yang Dibutuhkan

No	Nama Alat	Spesifikasi	Fungsi	Jumlah
1	Nodemcu	ESP32	Perangkat untuk memproses perintah yang akan dijalankan	1 Unit
2	Sensor kelembaban	Soil moisture sensor	Sebagai pendeteksi kadar air yang terkandung pada tanah	1 Unit
3	Sensor suhu	DHT 11	Sebagai pendeteksi suhu dan kelembaban pada lingkungan sekitar	1 Unit
4	Smartphone	Android	Digunakan untuk mengetes dan menjalankan prototipe desain aplikasi	1 Unit
5	Sensor Temperatur Tanah	Sensor DS18B20	Sebagai pendeteksi temperatur pada tanah	1 Unit
6	Relay	–	Mengendalikan aliran arus listrik dalam on/off	1 Unit
7	PCB matrik	Sirkuit elektronika	Sebagai papan sirkuit	1 Unit
8	Pompa Air	DC 12V	Digunakan untuk mengalirkan air pada pengembangan model smart farming	1 Unit
9	Box Kasing	–	Sebagai kasing untuk menyatukan dan melindungi rangkaian	1 Unit
10	Lcd 16x2 i2c	–	Menampilkan data dari sensor	1 Unit

3.3 Alur Penelitian

Pembahasan yang terdapat pada bab ini adalah mengenai langkah-langkah yang dilakukan dalam melakukan penelitian (Jembatan) Pengembangan Model Aplikasi Smart Farming Berbasis *Internet Of Things* digambarkan dengan blok diagram alur. Berikut adalah blok alur penelitian seperti pada gambar 3.1.



Gambar 3. 1 Alur Penelitian

3.4 Studi Literatur

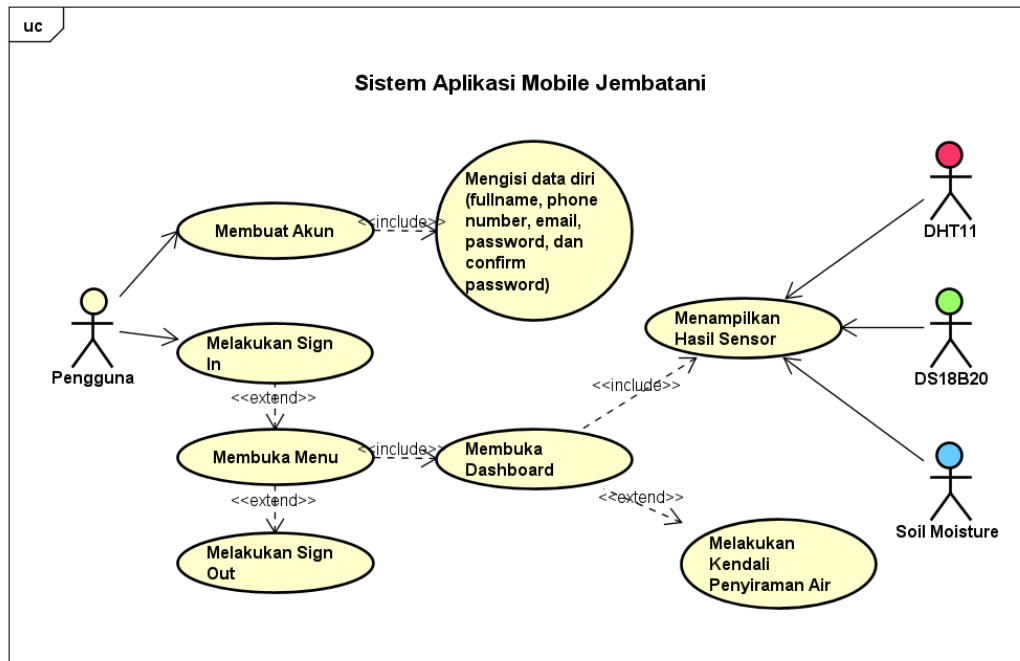
Pada tahap ini penulis mengumpulkan data atau referensi yang relevan pada penelitian tentang (Jembatan) Pengembangan Model Aplikasi *Smart Farming* Berbasis *Internet Of Things* dengan melibatkan pengumpulan jurnal, buku dan *website*.

3.5 Perancangan Sistem

Perancangan sistem merupakan proses pengembangan model aplikasi smart farming berbasis IoT yang akan menjelaskan bagaimana sistem ini dirancang secara keseluruhan. Bagian ini mencakup arsitektur sistem, yang menggambarkan hubungan antara komponen perangkat keras dan perangkat lunak, seperti sensor dan aktuator. Perancangan perangkat keras menjelaskan pemilihan skema dan integrasi sensor untuk pengumpulan data lingkungan serta aktuator untuk otomatisasi proses pertanian. Perancangan perangkat lunak akan dibahas dalam bentuk pengembangan desain ui/ux dan pemrograman aplikasi. Diagram *use case* dan *activity* juga akan disertakan untuk memvisualisasikan alur kerja sistem secara keseluruhan, dari pengumpulan data hingga pengambilan keputusan.

3.5.1 Use Case Diagram

Diagram *use case* dari Sistem Aplikasi Mobile Jembatani, yang merupakan bagian dari pengembangan aplikasi *smart farming* berbasis IoT. Diagram ini menunjukkan interaksi antara pengguna dan sistem dalam beberapa fungsi utama. Pengguna dapat membuat akun dengan mengisi data pribadi, melakukan sign in, membuka menu, melihat hasil sensor dari perangkat IoT seperti DHT11 (sensor suhu dan kelembaban), DS18B20 (sensor suhu), dan *Soil Moisture* (sensor kelembaban tanah). Setelah itu, pengguna dapat mengendalikan penyiraman air melalui aplikasi. Diagram ini menggambarkan bagaimana aplikasi memungkinkan pengguna untuk memantau kondisi lingkungan pertanian dan mengontrol proses irigasi secara efisien.



Gambar 3. 2 Use Case Diagram Sistem

Berikut tabel 3.4 merupakan deskripsi use case berdasarkan gambar diagram sistem yang dimulai dari pengguna (aktor), melakukan proses daftar akun dan masukan, melihat data sensor dan mengendalikan sistem akuator alat *smart farming*, dan sampai dengan sistem berakhir.

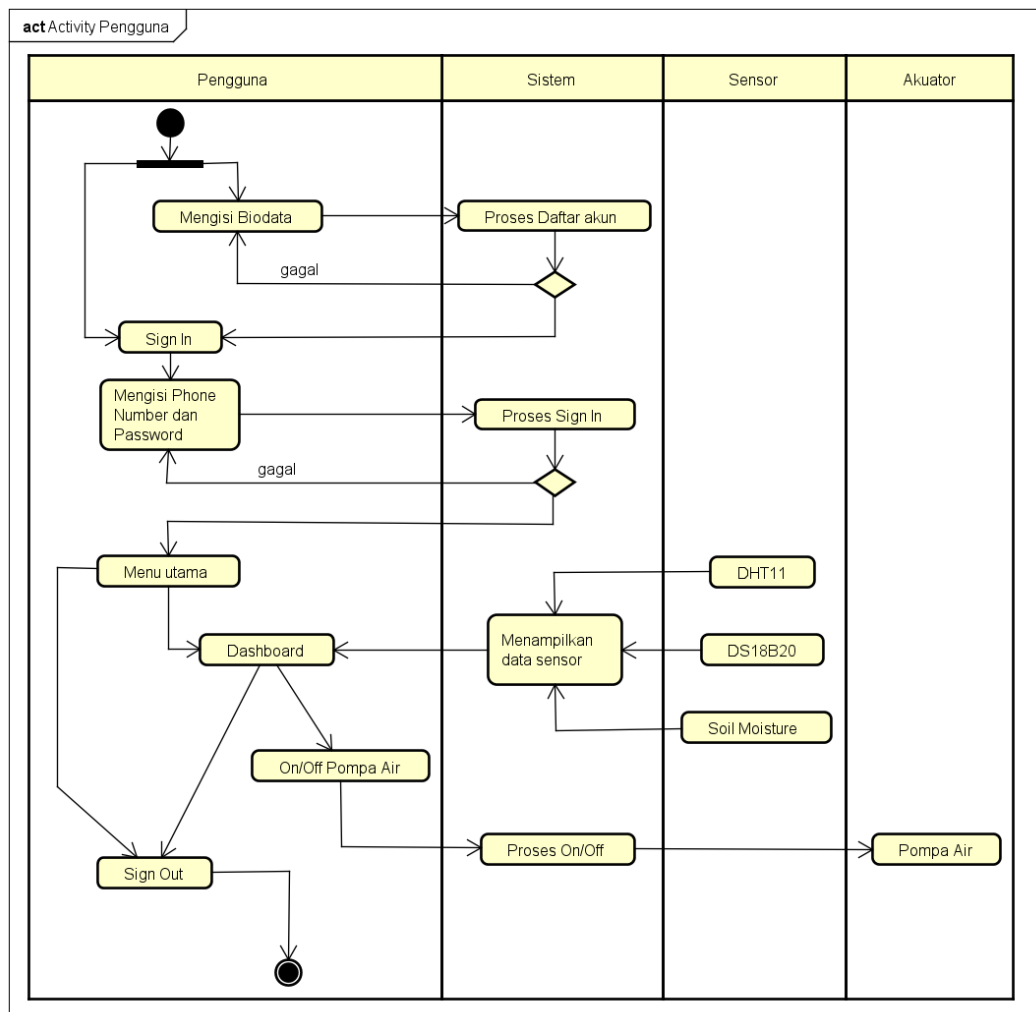
Tabel 3. 4 Deskripsi Use Case

Pengguna	Pengguna merupakan aktor yang menjalankan aplikasi Jembatani dari melakukan registrasi/masukan akun, melihat data sensor dan kendali sitem akuator secara realtime, dan keluar dari akun yang dibuat atau aplikasi.
Melakukan Sign in	Login merupakan bagian yang dibuat untuk melakukan masukan ke akun yang sudah diregistrasi sebelumnya.
Membuat Daftar Akun	Registrasi merupakan bagian yang dibuat untuk pengguna yang tidak memiliki akun, agar dapat masuk ke dalam aplikasi.
Membuka Menu	Merupakan bagian yang menampilkan berapa banyak kebun yang kita integrasi dengan perangkat keras IoT.

Membuka Dashboard	Tampilan pada aplikasi Jembatani yang memungkinkan untuk melihat data sensor dan kendali sistem akuator.
DHT11	Mengirimkan input sensor berupa kelembaban dan suhu yang ditampilkan pada aplikasi realtime, berdasarkan input dari sensor.
DS18B20	Mengirimkan input sensor berupa suhu tanah yang ditampilkan pada aplikasi realtime, berdasarkan input dari sensor.
Soil Moisture	Mengirimkan input sensor berupa kelembaban tanah yang ditampilkan pada aplikasi realtime, berdasarkan input dari sensor.
Melakukan Kendali Penyiraman Air	Merupakan bagian yang berfungsi untuk menghidupkan pompa air dari aplikasi secara realtime.

3.5.2 Activity Diagram

Diagram *activity* mendeskripsikan alur kerja atau langkah-langkah aktivitas yang dilakukan oleh pengguna dalam menjalankan suatu proses bisnis atau alur kerja. Diagram ini memvisualisasikan urutan kegiatan, pengambilan keputusan, serta interaksi antara berbagai komponen seperti pengguna, sistem, sensor, dan aktuator.

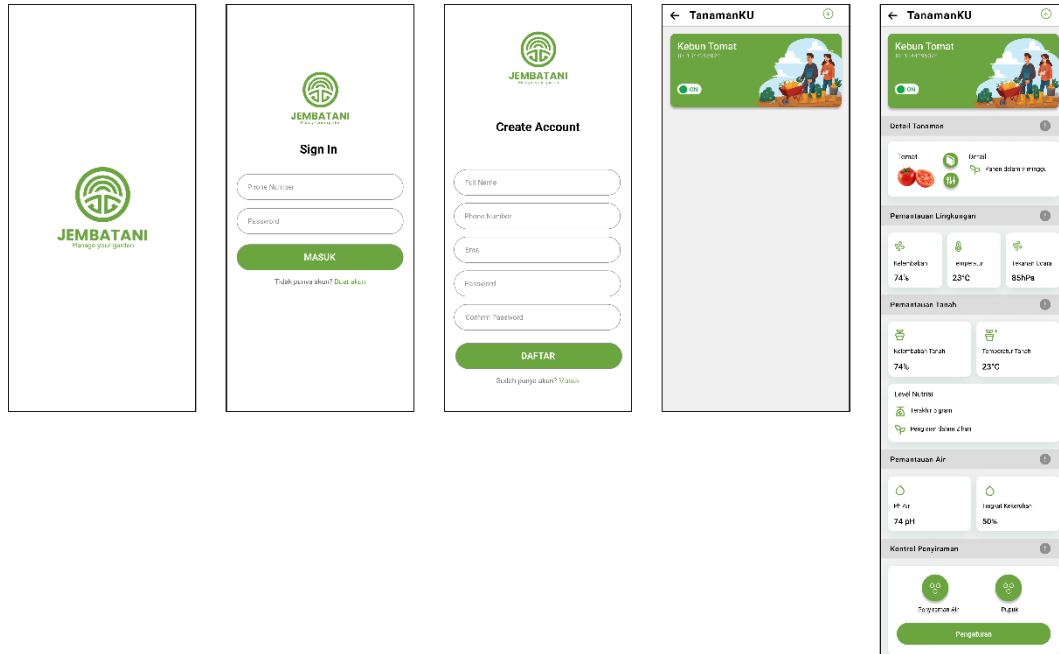


Gambar 3. 3 Activity Diagram

3.5.3 Rancangan Perangkat Lunak

Pada tahapan ini perancangan perangkat lunak dilakukan untuk menciptakan aplikasi mobile Jembatan sebagai antarmuka untuk menjalankan fungsi *use case* penelitian ini. Dalam tahapan ini penulis membaginya menjadi beberapa aspek.

3.5.3.1 Desain UI/UX Aplikasi Jembatani



Gambar 3. 4 Desain Aplikasi Jembatani

Rancangan antarmuka pengguna dan pengalaman pengguna (*User Interface* dan *User Experience*) merupakan proses untuk menciptakan komponen atau fitur yang disesuaikan dengan kebutuhan pengguna. Proses penelitian ini menggunakan metode *design thinking* dalam mengetahui kebutuhan pengguna yang dikeluarkan dalam bentuk visual. Berikut ini merupakan penjelasan fungsi dari desain aplikasi Jembatani.

a. SplashScreen

Tampilan halaman awal aplikasi, merupakan desain yang menampilkan logo/merek dari aplikasi sebagai pembuka.



Gambar 3. 5 Desain SplashScreen

b. Sign In dan Create Account User

Tampilan pada Gambar 3.6, merupakan halaman *sign in* ini terdapat fungsi yang digunakan untuk dapat mengakses aplikasi Jembatani, dengan mengisi biodata diri

1. *phone number* <diisi dengan akun pengguna>
2. *password* < diisi dengan akun pengguna>

Tampilan pada Gambar 3.6, merupakan halaman *create account* ini terdapat fungsi yang digunakan pengguna untuk membuat akun agar dapat mengakses aplikasi Jembatani, dengan mengisi biodata diri yang didaftarkan:

1. *full name* <diisi secara kustom oleh calon pengguna, untuk membuat akun)

2. *phone number* <diisi secara kustom oleh calon pengguna, untuk membuat akun>
3. *email* < diisi secara kustom oleh calon pengguna, untuk membuat akun >
4. *password* < diisi secara kustom oleh calon pengguna, untuk membuat akun >
5. *confirm password* < diisi secara kustom oleh calon pengguna, untuk membuat akun >

The image shows two side-by-side mobile app screens for 'JEMBATANI'. Both screens feature the JEMBATANI logo at the top, which consists of a green circular icon with a stylized bridge and the text 'JEMBATANI Manage your garden'.

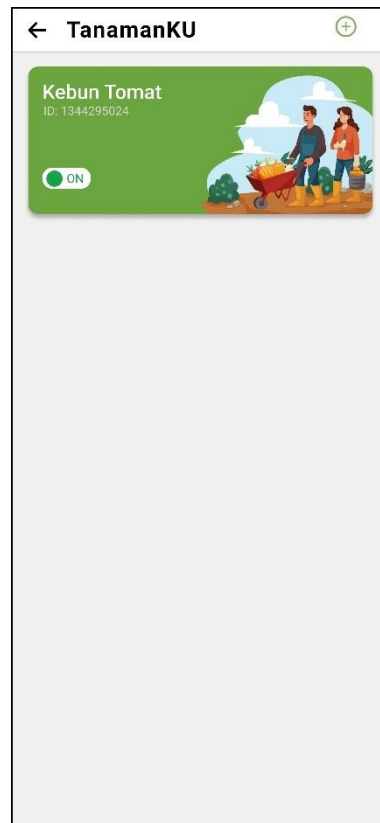
The left screen is titled 'Sign In'. It has two input fields: 'Phone Number' and 'Password'. Below these is a green button labeled 'MASUK'. At the bottom, there is a link that says 'Tidak punya akun? Buat akun'.

The right screen is titled 'Create Account'. It has five input fields: 'Full Name', 'Phone Number', 'Email', 'Password', and 'Confirm Password'. Below these is a green button labeled 'DAFTAR'. At the bottom, there is a link that says 'Sudah punya akun? Masuk'.

Gambar 3. 6 Desain Sign In dan Create Account

c. Menu

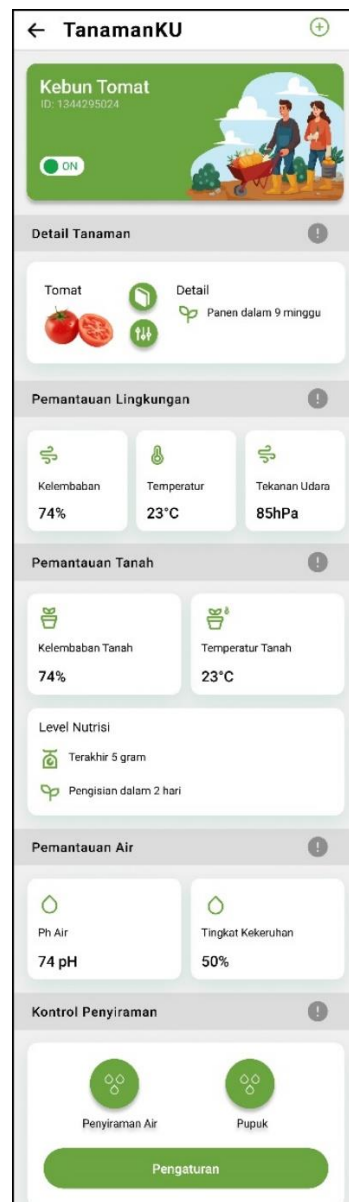
Halaman ini merupakan tampilan utama pada aplikasi Jembatanani. Akun pengguna digunakan untuk mendapatkan akses tampilan ini. Pengguna dapat melakukan aktivitas menambahkan kebun atau masuk ke dalam *dashboard monitoring* dan sistem *akuator* pompa air. Tampilan menu halaman pengguna dapat dilihat pada Gambar 3.7.



Gambar 3. 7 Desain Menu

d. Dashboard

Halaman ini merupakan tampilan *dashboard* kebun pada aplikasi Jembatani. Pengguna dapat mengetahui detail tanaman yang mereka tanam, melakukan aktivitas *monitoring* berdasarkan sensor yang aktif pada kebunnya dan sistem *akuator* pompa air. Tampilan menu halaman pengguna dapat dilihat pada Gambar 3.8.



Gambar 3. 8 Desain Dashboard

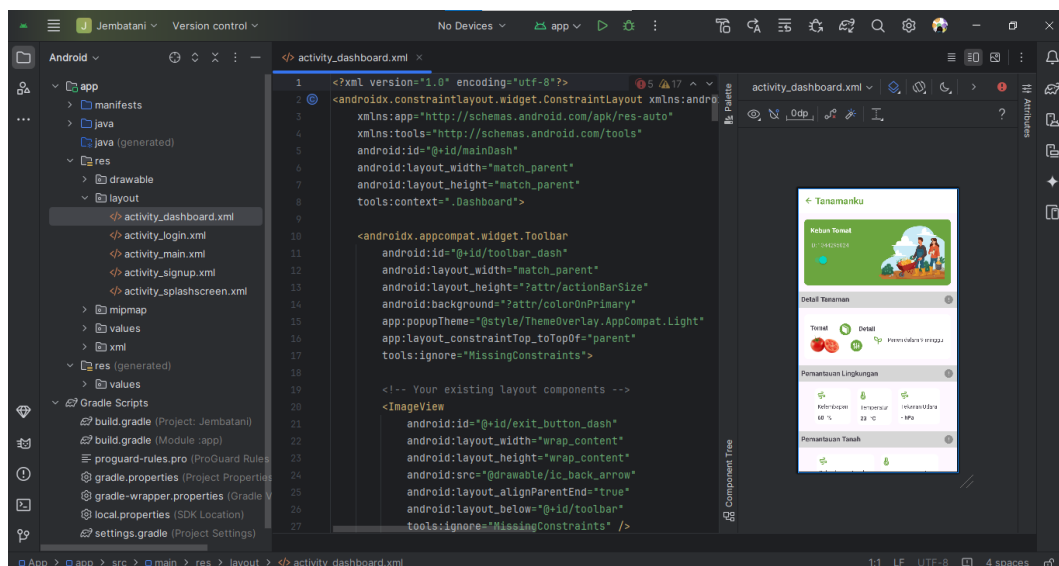
3.5.3.2 Pemrograman Aplikasi Jembatanani

Pemrograman aplikasi Jembatanani meliputi beberapa tahapan yang dilakukan untuk menghasilkan aplikasi efektif dan efisien. Tahap awal yang dilakukan adalah dengan mempersiapkan *setup tools* aplikasi *Android Studio IDE* dari *Google*. *Android Studio IDE* ini menyediakan fitur yang memudahkan dalam pengembangan aplikasi Jembatanani. Tahap selanjutnya pengembang melakukan *layouting* berdasarkan desain yang sebelumnya dibuat dengan bahasa pemrogram

yang kompatibel dengan *Android Studio IDE* seperti *Java*, *Kotlin* dan *Flutter*. Pada tahap ini pemrograman dibagi 2 tahapan. Berikut merupakan tahapan yang dilakukan dalam membuat aplikasi Jembatanani.

a. Layouting Aplikasi Jembatanani

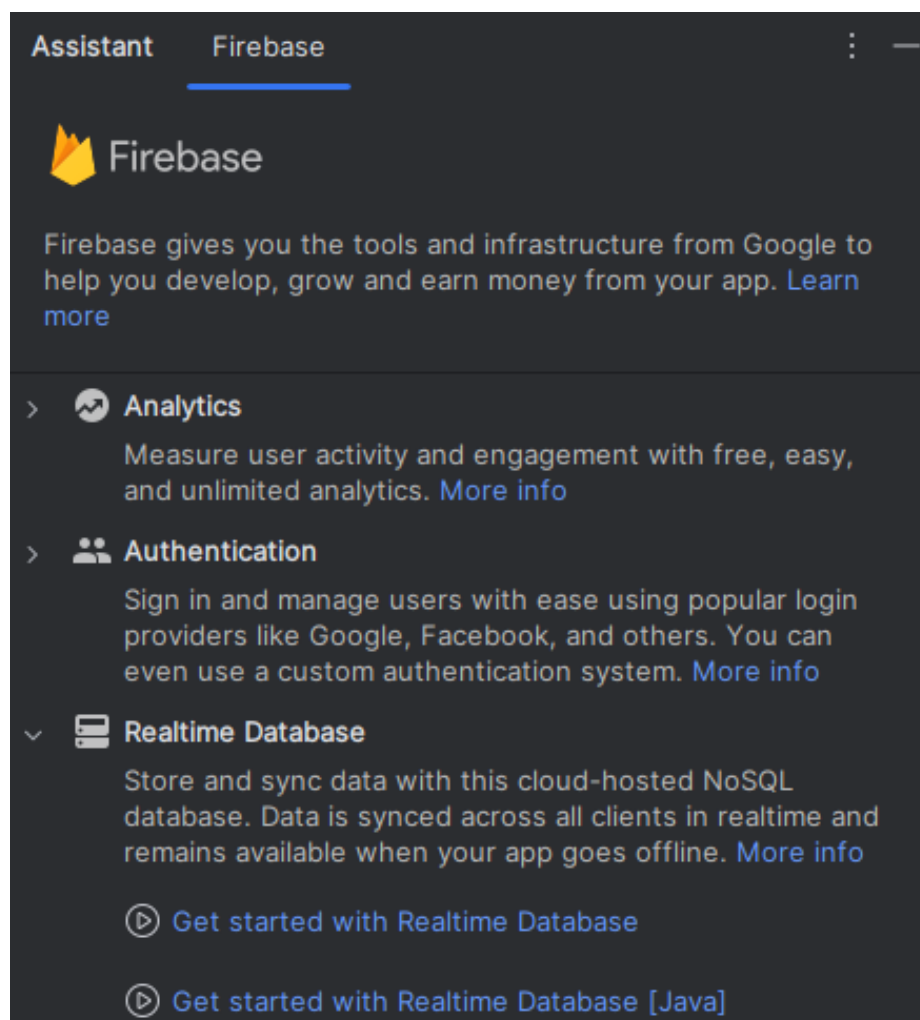
Layouting aplikasi Jembatanani pada *Android Studio* melibatkan pembuatan desain antarmuka pengguna yang intuitif dan responsif dengan memanfaatkan komponen-komponen dasar seperti *View* dan *ViewGroup*. Proses dimulai dengan menentukan struktur tata letak berdasarkan desain yang sebelumnya dibuat, dengan menggunakan tipe layout seperti *LinearLayout* untuk susunan vertikal atau horizontal, *RelativeLayout* untuk posisi relatif satu sama lain, dan *ConstraintLayout* untuk fleksibilitas yang lebih kompleks dalam mengatur elemen. Setiap komponen layout antarmuka memiliki komponen, seperti tombol, teks, dan gambar, didefinisikan dalam file XML yang memungkinkan dalam mengatur properti visual dan interaktifnya. Selain itu, *Android Studio* memiliki fitur yang memungkinkan pengembang dalam melakukan *drag-and-drop* di *Design Editor* untuk memudahkan pengaturan layout secara visual, serta alat-alat untuk memastikan tampilan tetap konsisten.



Gambar 3. 9 Layouting Aplikasi Jembatanani

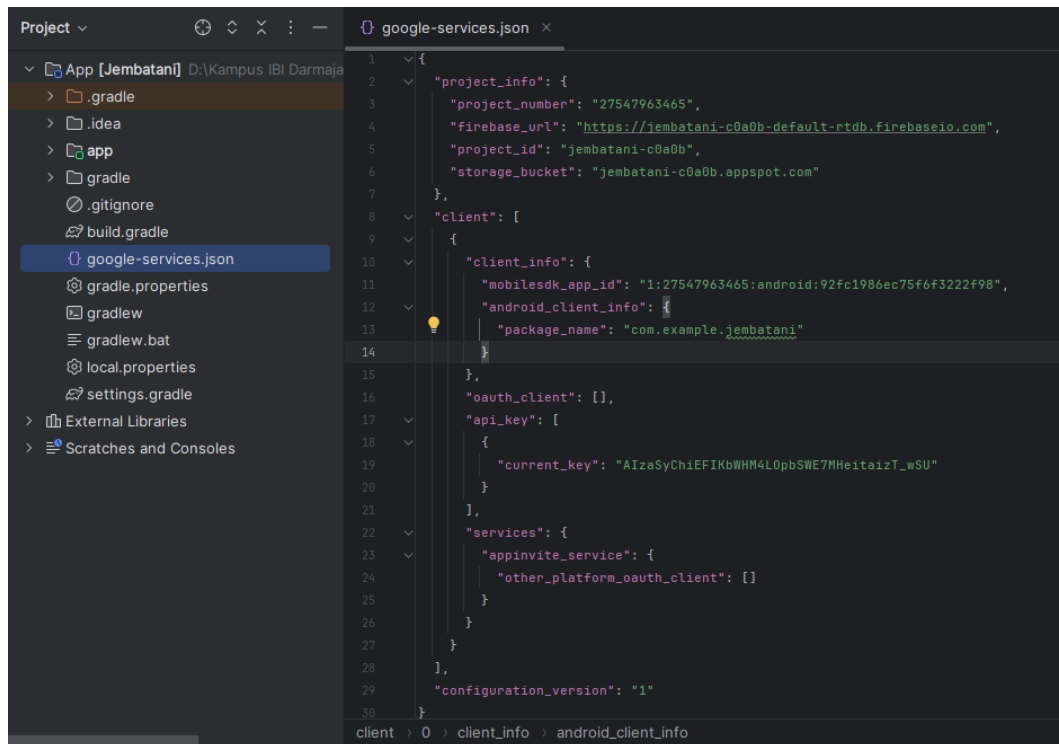
b. Integrated Firebase REST API

Integrated Firebase REST memungkinkan pengembangan aplikasi Jembatan dan perangkat IoT berinteraksi berbagai layanan melalui permintaan *HTTP* dengan efisien dan fleksibel dalam menampilkan data sensor dan operasi perangkat. Dengan metode *HTTP* dengan standar *GET*, *POST*, *PUT* dan *DELETE*, perangkat IoT memungkinkan dapat menyimpan, membaca, mengupdate dan menghapus data di *Firebase Realtime Database*, serta autentikasi perangkat. Hal ini memungkinkan perangkat IoT dapat mengirimkan data sensor secara *realtime*, dan menampilkannya pada aplikasi Jembatan.



Gambar 3. 10 Integrated Firebase REST API

Konfigurasi layanan *firebase* dilakukan agar aplikasi *Android* mendapatkan layanan *firebase* adalah dengan menambahkan file “*google-services.json*” pada proyek *Android Studio*. Dengan melakukan konfigurasi, aplikasi *Android* dapat secara otomatis terhubung dengan proyek *firebase*, sehingga memudahkan dalam menambahkan metode *HTTP*.



```

1  {
2  }
3  "project_info": {
4    "project_number": "27547963465",
5    "firebase_url": "https://jembatanani-c0a0b-default-rtdb.firebaseio.com",
6    "project_id": "jembatanani-c0a0b",
7    "storage_bucket": "jembatanani-c0a0b.appspot.com"
8  },
9  "client": [
10   {
11     "client_info": {
12       "mobilesdk_app_id": "1:27547963465:android:92fc1986ec75f6f3222f98",
13       "android_client_info": {
14         "package_name": "com.example.jembatanani"
15       }
16     },
17     "oauth_client": [],
18     "api_key": [
19       {
20         "current_key": "AIzaSyCh1EFIKbWHH4L0pbSWE7MHHeitaizT_wSU"
21       }
22     ],
23     "services": {
24       "appinvite_service": {
25         "other_platform_oauth_client": []
26       }
27     }
28   },
29   "configuration_version": "1"
30 }

```

Gambar 3. 11 Konfigurasi Layanan Firebase

Memberikan koneksi pada pengembangan aplikasi *Android*, dengan membuat file *FireApp.java* dan menambahkan fungsi *void onCreate()*. Penambahan fungsi ini dilakukan agar ketika aplikasi *Jembatanani* menerima atau mengirim perintah, akan langsung terkoneksi secara *realtime* dengan layanan *firebase* untuk diteruskan kepada perangkat *IoT*.


```

© FireApp.java x
1 package com.example.jembatani;
2
3 > import ...
6
7 public class FireApp extends Application{
8     @Override
9     public void onCreate() {
10         super.onCreate();
11
12         Firebase.setAndroidContext(this);
13     }
14
15 }

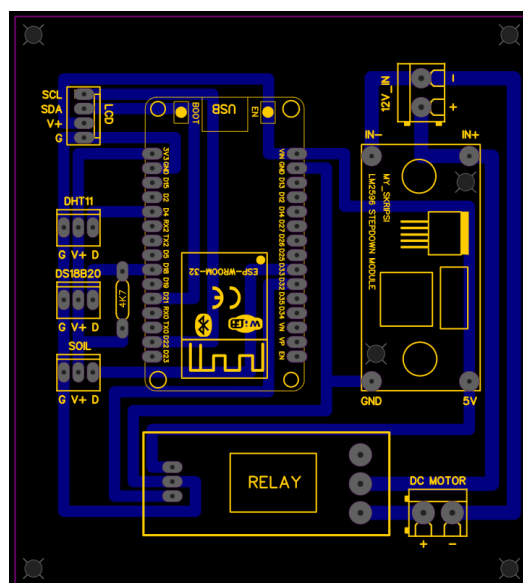
```

Gambar 3. 12 Koneksi Layanan Firebase

3.5.4 Perancangan Pada Perangkat Keras

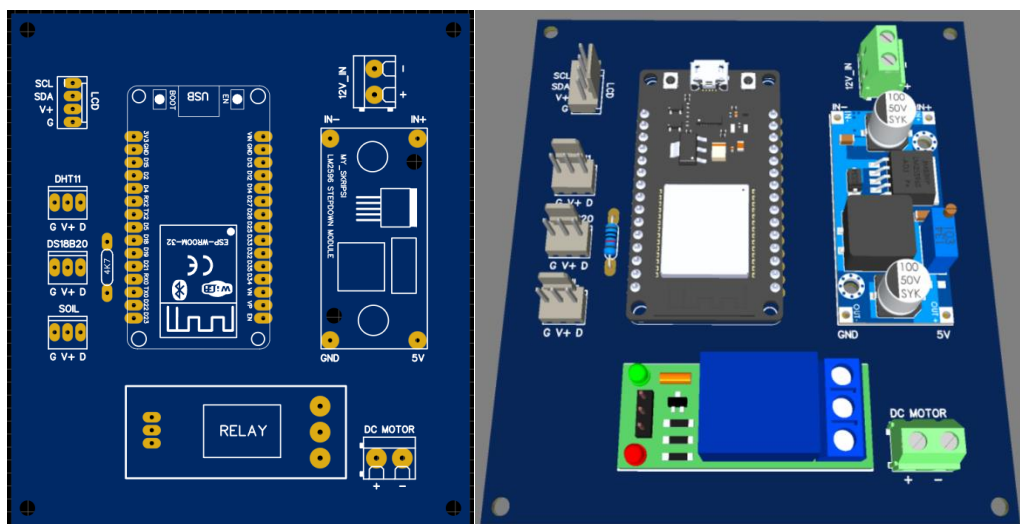
Perancangan perangkat keras merupakan proses dalam menggabungkan komponen elektronika yang dilakukan dalam penelitian ini, agar dapat membentuk sistem *smart farming* efektif dan efisien untuk pengguna.

3.5.4.1 Rangkaian Sirkuit PCB



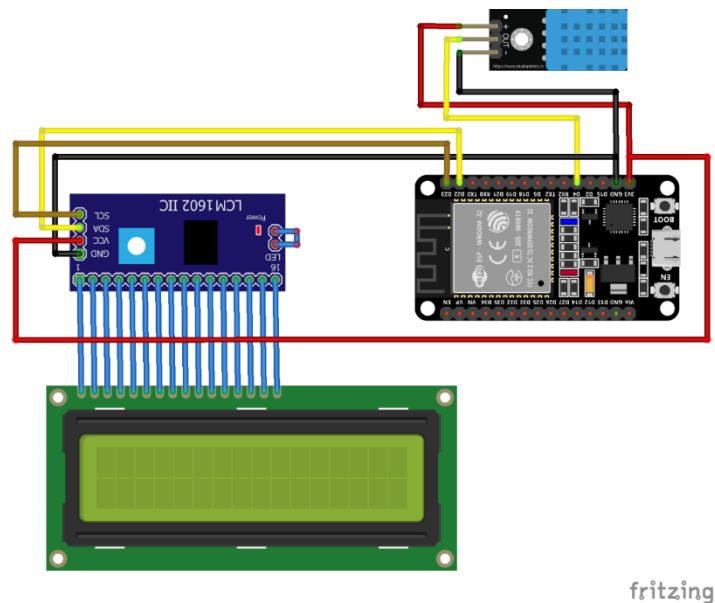
Gambar 3. 13 Layout Sirkuit PCB

Rangkaian sirkuit PCB pada gambar tersebut menunjukkan tata letak beberapa komponen yang saling terhubung. Pada rangkaian terdapat banyak komponen yang saling terkoneksi seperti ESP-WROOM-32 (ESP32), *header pin* (SCL, SDA, V+, G) dan (DHT11, DS18B20, SOIL) digunakan untuk menghubungkan berbagai sensor), resistor (4.7k Ohm) digunakan sebagai *pull-up* untuk sensor DS18B20, modul *Step Down* (LM2596) modul ini mengkonversi tegangan dari 12V menjadi 5V untuk menyuplai daya ke ESP32 dan komponen lainnya, relay digunakan untuk mengontrol pompa air, dan konektor daya (IN+, IN) digunakan untuk menghubungkan Power Supply sebagai sumber tegangan eksternal AC. Berikut rangkaian sirkuit PCB dalam tampilan 2 dimensi dan 3 dimensi.



Gambar 3. 14 Rangkaian Sirkuit PCB 2D dan 3D

3.5.4.2 Rangkaian Sensor DHT 11



Gambar 3. 15 Rangkaian DHT 11

Berdasarkan gambar diagram rangkaian di atas dapat dijelaskan sensor DHT 11 digunakan untuk memantau suhu dan kelembaban lingkungan sekitar. Komponen yang digunakan pada rangkaian ini antara lain:

1. Mikrokontroler Nodemcu ESP32 berfungsi sebagai alat berpikir dan mengontrol seluruh komponen.
2. Sensor DHT 11 digunakan untuk memantau suhu dan kelembaban lingkungan sekitar.
3. Display LCD 16x2 i2c digunakan untuk menampilkan data suhu dan kelembaban.

Dalam diagram rangkaian dapat diketahui bahwa sensor DHT 11 mendapatkan catu daya dari pin 3V3, terhubung ke pin GND dan pin data D4 pada papan mikrokontroler. Display LCD 16x2 i2c juga dihubungkan dengan papan mikrokontroler. Dengan konfigurasi yang dilakukan ini dapat memungkinkan data suhu dan kelembaban dari sensor DHT11, yang selanjutnya dapat ditampilkan pada display LCD 16x2 i2c.

```

void sensor_dht11() {
    get_dht11 = read_dht(temperature, humidity, 4, DHT11);

    firebase.setInt("sensor/dht11_humidity", humidity); // Kirim Data Ke Firebase
    firebase.setInt("sensor/dht11_suhu", temperature); // Kirim Data Ke Firebase

    lcd.setCursor(12, 0);
    lcd.print(" ");
    lcd.setCursor(12, 0);
    lcd.print(temperature,0);
    lcd.setCursor(15,0);
    lcd.print("C");
    lcd.setCursor(12, 1);
    lcd.print(" ");
    lcd.setCursor(12, 1);
    lcd.print(humidity,0);
    lcd.setCursor(15, 1);
    lcd.print("%");
}

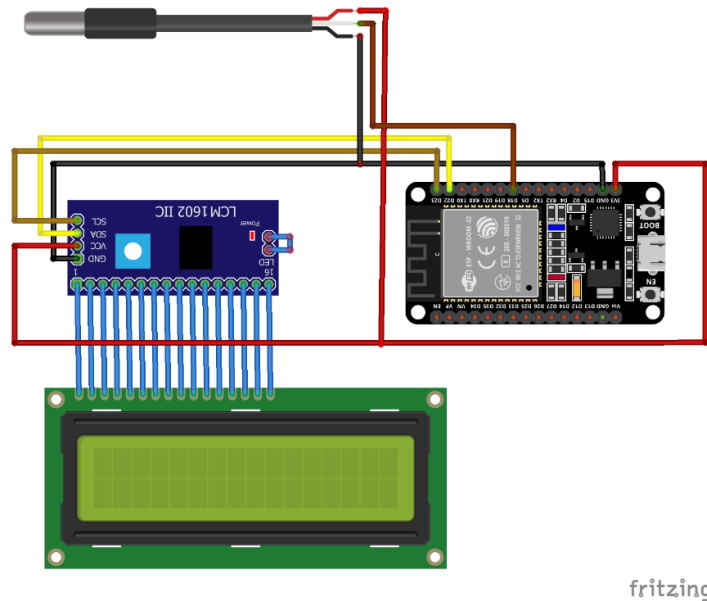
```

Gambar 3. 16 Potongan Program DHT 11 dan LCD 16x2 i2c

Program ini terdiri dari beberapa fungsi utama yang bekerja bersama untuk membaca data dari sensor DHT11, mengirimkan data tersebut ke *Firestore*, dan menampilkan data pada layar LCD. Fungsi *read_dht* digunakan untuk membaca suhu dan kelembaban dari sensor DHT11, yang kemudian disimpan dalam variabel *temperature* dan *humidity*. Setelah data dibaca, fungsi *firebase.setInt* digunakan untuk mengirimkan data suhu dan kelembaban ke *Firestore*, masing-masing pada path *sensor/dht11_humidity* dan *sensor/dht11_suhu*. Hal ini memungkinkan data yang diperoleh dari sensor untuk disimpan secara online dan diakses dari jarak jauh.

Selain itu, program ini menggunakan beberapa fungsi lcd untuk menampilkan data suhu dan kelembaban pada layar LCD. Fungsi *lcd.setCursor* digunakan untuk mengatur posisi kursor di layar LCD sebelum mencetak data. Fungsi *lcd.print* kemudian digunakan untuk mencetak data suhu dan kelembaban pada posisi yang telah ditentukan. Nilai suhu dicetak bersama dengan simbol “C” untuk menunjukkan derajat Celcius, sedangkan nilai kelembaban dicetak bersama dengan simbol “%” untuk menunjukkan persentase kelembaban. Dengan kombinasi fungsi-fungsi ini, program ini tidak hanya mengumpulkan dan menyimpan data lingkungan tetapi juga memberikan visualisasi real-time kepada pengguna.

3.5.4.3 Rangkaian Sensor DS18B20



Gambar 3. 17 Rangkaian Sensor DS18B20

Rangkaian tersebut tersusun dengan sensor suhu DS18B20, modul LCD I2C, dan modul mikrokontroler ESP32. Sensor DS18B20 terhubung ke mikrokontroler melalui pin data (D5) untuk mengukur suhu, dengan pin VCC dan GND terhubung ke sumber daya dan ground. Modul LCD I2C memudahkan kontrol layar LCD menggunakan protokol I2C, dengan pin SCL dan SDA terhubung ke pin D1 dan D2 mikrokontroler. Mikrokontroler mengumpulkan data suhu dari sensor DS18B20 dan menampilkannya pada layar LCD melalui modul I2C.

```
void sensor_DS18B20() {
    DS18B20.requestTemperatures(); // Send the command to get temperatures
    suhu = DS18B20.getTempCByIndex(0);

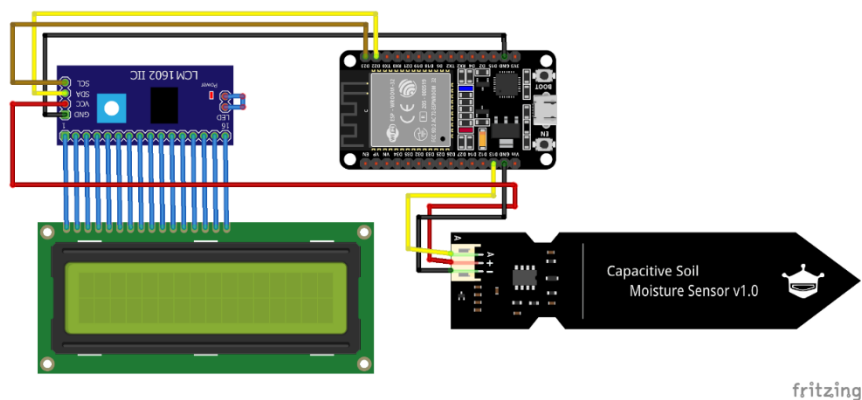
    firebase.setInt("sensor/ds18b20", suhu); // Kirim Data Ke Firebase

    lcd.setCursor(3,0);
    lcd.print(" ");
    lcd.setCursor(3,0);
    lcd.print(suhu,0);
    lcd.setCursor(6, 0);
    lcd.print("C");
}
```

Gambar 3. 18 Potongan Program DS18B20 dan LCD 16x2 i2c

Fungsi `DS18B20.requestTemperatures()` mengirimkan perintah untuk mengambil data suhu, sedangkan `suhu = DS18B20.getTempCByIndex(0)`; mengambil suhu dalam derajat Celcius dari sensor pertama (indeks 0). Data suhu ini kemudian dikirim ke Firebase menggunakan `firebase.setInt("sensor/ds18b20", suhu)`. Selanjutnya, program mengatur tampilan pada layar LCD. `Lcd.setCursor(3,0)`; mengatur posisi kursor pada baris 0, kolom 3, diikuti dengan `lcd.print(" ");` untuk menghapus tampilan sebelumnya. Lalu, `lcd.setCursor(3,0)`; mengatur kembali kursor pada posisi awal, dan `lcd.print(suhu,0)`; menampilkan nilai suhu tanpa angka desimal. Akhirnya, `lcd.setCursor(6,0)`; mengatur kursor di posisi setelah angka suhu, dan `lcd.print("C")`; menambahkan simbol derajat Celsius.

3.5.4.4 Rangkaian Sensor Soil Moisture



Gambar 3. 19 Rangkaian Soil Moisture

Rangkaian tersebut tersusun dari beberapa komponen elektronika dengan mikrokontroler NodeMCU ESP32, display LCD 16x2 i2c dan Capacitive Soil Moisture Sensor yang berfungsi untuk mengetahui kandungan air yang terkandung pada tanah. Dalam penyusunan rangkaian pin yang dihubungkan memungkinkan sensor untuk mendeteksi data yang kemudian akan ditampilkan display LCD 16x2 i2c. Display LCD 16x2 i2c digunakan untuk menampilkan status data yang diinput oleh sensor.

```

void sensor_soil(){
    int data_adc = analogRead(33);
    soil_moist = map(data_adc, 1700, 3200, 100, 0);
    firebase.setInt("sensor/soil", soil_moist); // Kirim Data Ke Firebase
    Serial.println(soil_moist);
    lcd.setCursor(3, 1);
    lcd.print("  ");
    lcd.setCursor(3, 1);
    lcd.print(soil_moist);
    lcd.setCursor(6, 1);
    lcd.print("%");
}

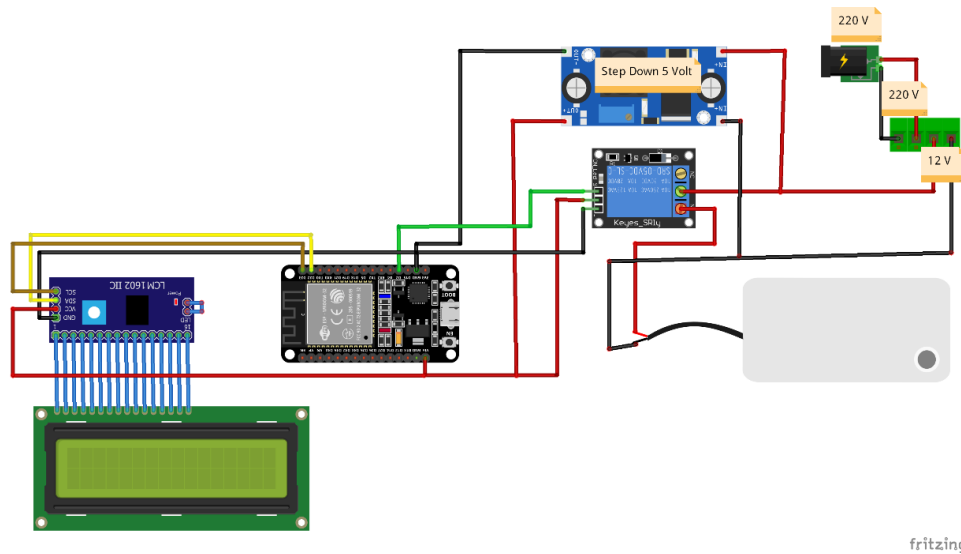
```

Gambar 3. 20 Potongan Program Soil Moisture dan LCD 16x2 i2c

Fungsi `sensor_soil()` dimulai dengan membaca data dari pin analog 33 menggunakan `analogRead()`, kemudian menyimpan nilai tersebut dalam variabel `data_adc`. Nilai ini kemudian dipetakan ke dalam skala yang lebih bermakna untuk kelembaban tanah menggunakan fungsi `map()`. Data kelembaban tanah yang telah dipetakan (`soil_moist`) kemudian dikirimkan ke Firebase dengan menggunakan `firebase.setInt()`, yang menyimpan nilai tersebut pada path "sensor/soil". Nilai kelembaban juga dicetak ke serial monitor menggunakan `Serial.println()` untuk tujuan debugging atau monitor.

Setelah itu, nilai kelembaban ditampilkan pada layar LCD. Posisi kursor LCD diatur terlebih dahulu menggunakan `lcd.setCursor()`, dan layar dibersihkan dengan mencetak spasi kosong (" "). Nilai kelembaban kemudian dicetak pada posisi yang diinginkan menggunakan `lcd.print()`, diikuti dengan mencetak simbol persentase (%) untuk menunjukkan bahwa nilai tersebut merupakan persentase kelembaban tanah. Fungsi ini membantu dalam memonitor kondisi kelembaban tanah secara real-time dan menyimpannya dalam *database firebase* untuk analisis lebih lanjut.

3.5.4.5 Rangkaian Sistem Kendali Pompa Air



Gambar 3. 21 Rangkaian Sistem Kendali

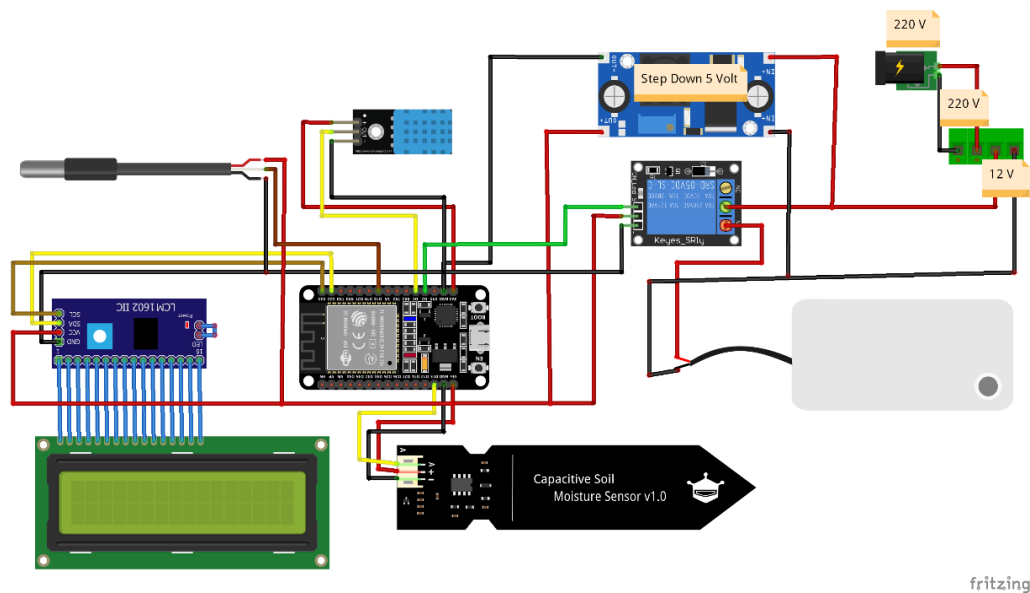
Rangkaian ini merupakan skema dari perakitan pada sistem kendali pompa air yang dapat dioperasikan untuk pengelolaan air dalam jarak jauh oleh pengguna menggunakan *smart farming*. Komponen penyusun skeman ini adalah mikrokontroler Nodemcu ESP32 sebagai alat berpikir yang memproses sinyal input dan mengendalikan output. Juga terdapat Relay yang berguna mengaktifkan dan menonaktifkan pompa air berdasarkan perintah dari mikrokontroler.

```
void kontrol_relay (){
    data1 = firebase.getString("skripsi/relay");
    Serial.print("Received String:\t");
    if(data1 == "1"){
        digitalWrite(relay_pin, HIGH);
        satu();
    }
    else{
        digitalWrite(relay_pin, LOW);
        nol();
    }
    Serial.print("Relay : ");
    Serial.println(data1);
}
```

Gambar 3. 22 Potongan Program Sistem Kendali Pompa Air

Fungsi kontrol_relay() dimulai dengan mengambil data string dari Firebase pada path "sensor/relay" menggunakan firebase.getString(), dan menyimpannya dalam variabel data1. Data yang diterima kemudian dicetak ke serial monitor untuk tujuan debugging atau monitor. Setelah data diterima, program mengecek apakah data1 bernilai "1". Jika benar, relay diaktifkan dengan mengatur pin relay (relay_pin) ke HIGH menggunakan digitalWrite(), dan memanggil fungsi satu(). Jika data1 tidak bernilai "1", relay dimatikan dengan mengatur pin relay ke LOW dan memanggil fungsi nol(). Status relay dicetak ke serial monitor untuk memastikan status terbaru dari relay. Fungsi ini memungkinkan sistem untuk mengontrol perangkat yang terhubung dengan relay secara jarak jauh melalui Firebase, memberikan fleksibilitas dalam pengoperasian perangkat.

3.5.4.6 Seluruh Rangkaian Smart Farming



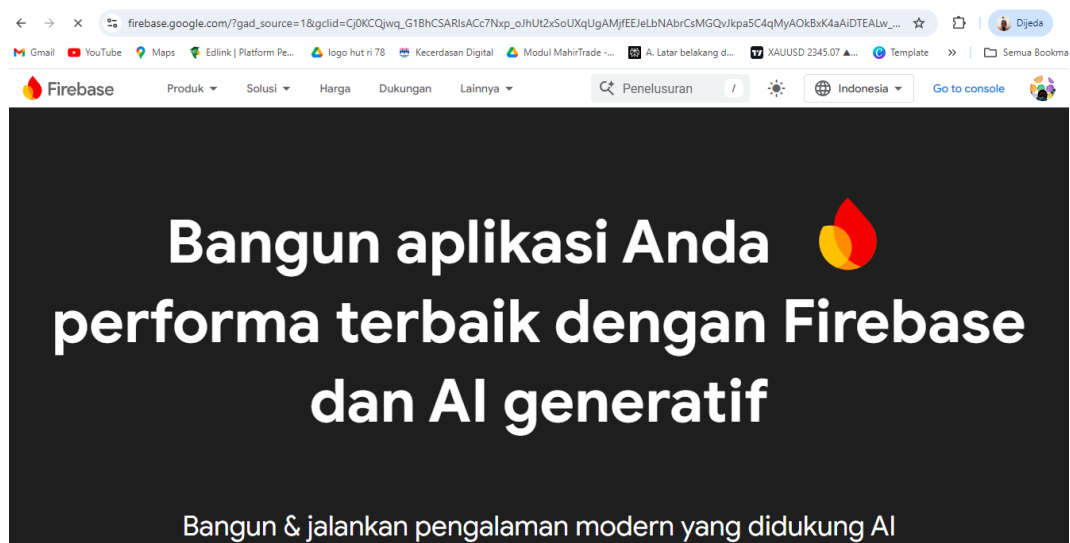
Gambar 3. 23 Rangkaian Smart Farming

Rangkaian atau skema keseluruhan merupakan komponen yang sudah disusun menjadi satu kesatuan. Dalam hal ini penulis membuat skema *smart farming* yang dapat dimanfaatkan untuk memonitoring dan melakukan penyiraman secara *real-time*. Adapun skema keseluruhan ditunjuk pada gambar 3.21.

Skema di atas merupakan seluruh rangkaian elektronika yang menjadi satu kesatuan pada *smart farming*. Yang meliputi beberapa skema yang digabungkan menjadi satu, antara lain:

- Mikrokontroler Nodemcu ESP 32 yang berfungsi sebagai otak sistem untuk memproses data.
- Sensor kelembaban tanah yang terhubung ke mikrokontroler untuk mengukur tingkat kelembaban tanah.
- Relay yang dapat dikendalikan oleh mikrokontroler untuk mengaktifkan atau menonaktifkan perangkat lain, seperti pompa air.
- Komponen-komponen pendukung lainnya seperti kabel, resistor, dan modul komunikasi serial.

3.5.5 Perancangan Firebase Realtime Database

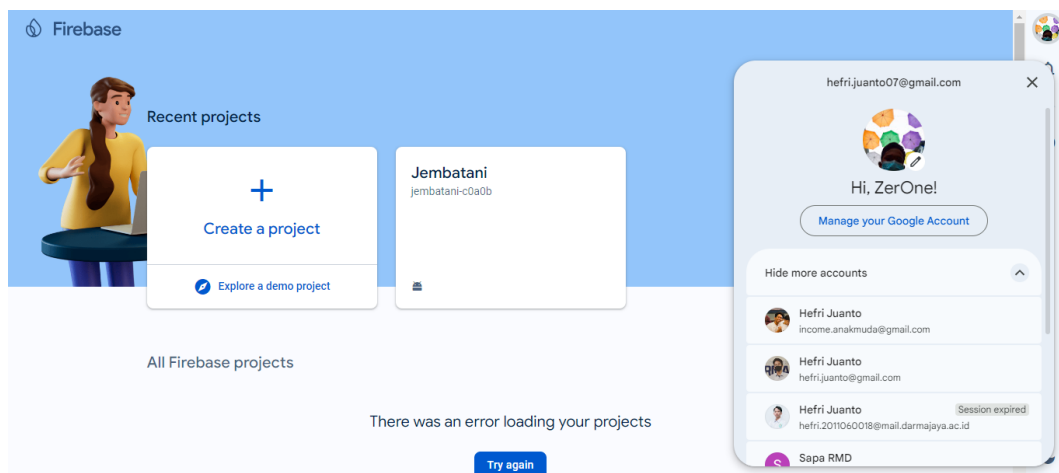


Gambar 3. 24 Firebase

Perancangan Firebase Realtime Database pada aplikasi ini bertujuan untuk mengelola dan menyimpan data secara terpusat dan real-time untuk beberapa fungsi penting, seperti autentikasi pengguna, monitoring sensor, dan kontrol perangkat. Struktur database ini terdiri dari beberapa node utama, termasuk “users” yang menyimpan informasi pengguna seperti nama, email, nomor telepon, dan password. Node “skripsi” menyimpan data sensor yang dikumpulkan dari perangkat IoT, seperti kelembaban, suhu, kelembaban tanah, dan suhu tanah. Selain itu, terdapat

node “watering” yang digunakan untuk mengirimkan perintah kontrol, seperti menghidupkan atau mematikan penyiraman, dengan menggunakan nilai 0 atau 1. Pendekatan ini memastikan bahwa semua data dapat diakses dan diupdate secara real-time oleh aplikasi, memungkinkan integrasi yang mulus antara pengguna dan perangkat IoT. Berikut ini merupakan alur pembuatan *firebase realtime database*:

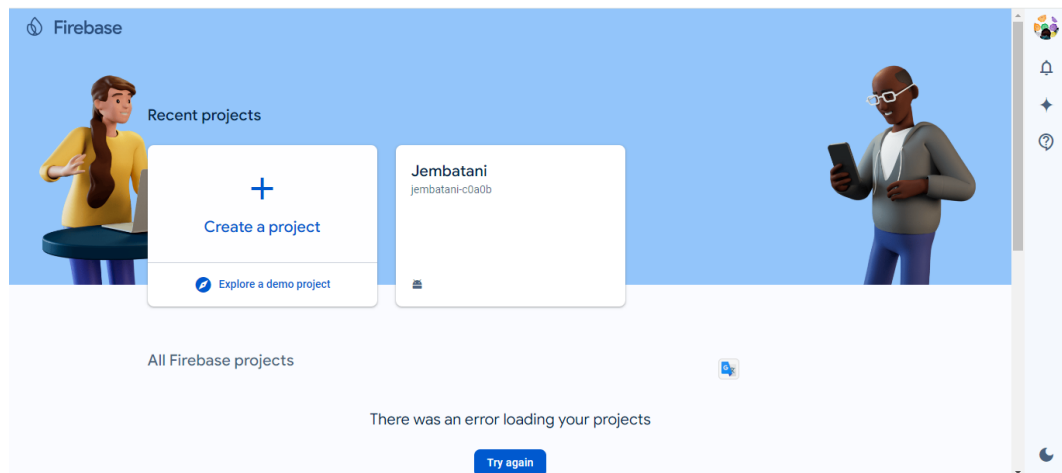
3.5.5.1 Membuat Akun dan Login ke Firebase Console



Gambar 3. 25 Login Firebase Console

Langkah pertama untuk menggunakan *Firebase Realtime Database* adalah dengan membuat akun *Google*. Setelah memiliki akun, buka *Firebase Console* (<https://console.firebase.google.com/>) dan login menggunakan akun *Google* tersebut. *Firebase Console* adalah platform utama untuk mengelola semua proyek *firebase*, termasuk *database*, *hosting*, otentikasi, dan lain-lain. Setelah *login*, selanjutnya membuat proyek *firebase* dengan menekan *create a project*.

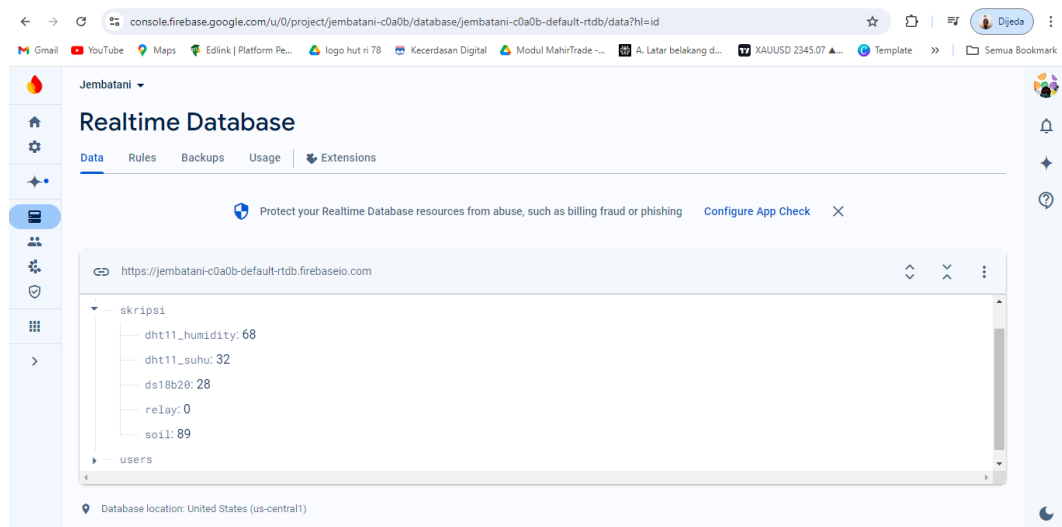
3.5.5.2 Membuat Proyek di Firebase Console



Gambar 3. 26 Membuat Proyek Firebase Console

Setelah melakukan *login*, langkah selanjutnya adalah membuat proyek baru di *Firebase Console*. Klik tombol “*Add Project*” dan masukkan nama proyek. Dengan mengaktifkan atau menonaktifkan *Google Analytics* sesuai kebutuhan. Setelah proyek dibuat, *Firebase* akan mengatur *backend* dan infrastruktur yang diperlukan untuk aplikasi *android*. Proyek ini akan menjadi wadah untuk mengelola semua layanan *firebase* yang akan dibuat dalam pengembangan aplikasi *Jembatani*, termasuk *realtime database*.

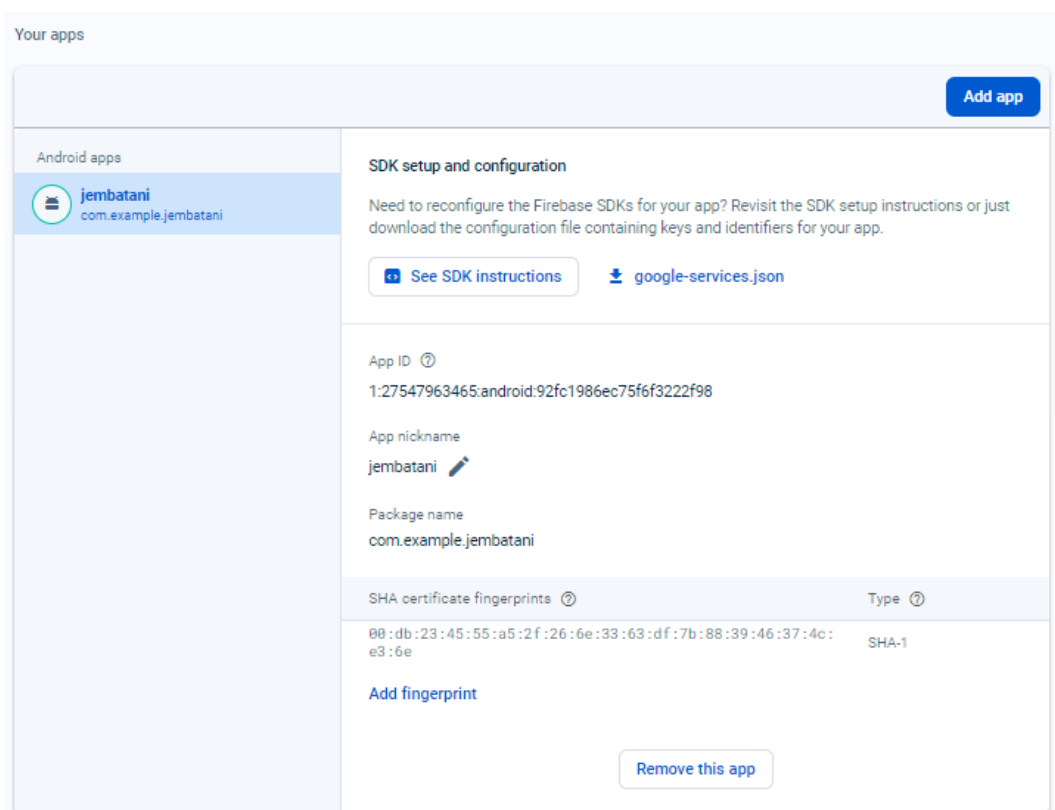
3.5.5.3 Mengatur Firebase Realtime Database



Gambar 3. 27 Firebase Realtime Database

Dengan proyek sudah dibuat, navigasikan ke bagian "Database" di Firebase Console dan pilih "Create Database" di bawah tab Realtime Database. Saat pembuatan firebase akan meminta untuk memilih lokasi penyimpanan data dan mengatur aturan akses (*security rules*). Aturan ini dapat disesuaikan untuk menentukan siapa yang dapat membaca atau menulis data ke *database* aplikasi Jembatanani. Setelah itu, *firebase* akan membuat *instance realtime database* yang dapat diakses dari aplikasi.

3.5.5.4 Integrasi Firebase ke Android Studio



Gambar 3. 28 SDK Firebase

Untuk mengintegrasikan *firebase realtime database* dengan android studio atau aplikasi yang dibuat, buka tab "*project settings*" di *firebase console*, lalu pilih opsi "*general*" untuk menambahkan data terkait proyek yang dibuat dengan android studio. Firebase akan memberikan konfigurasi SDK Firebase berupa cuplikan kode *JavaScript* yaitu *google-services.json* yang perlu ditambahkan ke dalam proyek

android studio. Dengan mengimpor SDK tersebut, aplikasi akan dapat memulai menggunakan layanan *firebase*, termasuk *realtime database* dalam aplikasi web Anda.

3.6 Implementasi

Setelah mengumpulkan alat dan bahan serta melakukan perancangan, ditahap ini dilakukan implementasi perangkat keras, implementasi perangkat lunak dan implementasi *firebase realtime database*. Implementasi pada *firebase* dilakukan untuk mengintegrasikan perangkat keras dan perangkat lunak, agar dapat mengirim atau menerima data satu sama lain.

3.7 Pengujian Keseluruhan Sistem

Dalam pengujian sistem, meliputi uji aplikasi Jembatanani dan perangkat keras *smart farming*, serta uji *firebase realtime database* untuk melihat data yang divalidasi. Pertama, integrasi sistem harus diverifikasi, memastikan semua komponen dapat saling berkomunikasi dan bertukar data dengan lancar. Validasi fungsional terhadap setiap fitur, seperti pembuatan akun pengguna, masuk ke dalam aplikasi dengan akun yang sudah dibuat, monitoring data pertanian dan penyiraman air harus diuji untuk memastikan sistem berjalan sesuai harapan peneliti.

3.8 Analisa

Analisa dilakukan bersama saat melakukan pengujian keseluruhan sistem yang bertujuan untuk mengetahui kerja alat tersebut. Selain ini analisa dilakukan untuk mengetahui kecepatan dan responsivitas aplikasi menerima data sensor, serta perintah untuk menghidupkan penyiraman air pada perangkat keras dari aplikasi Jembatanani. Berdasarkan hasil pengujian keseluruhan sistem yang telah didapat akan dianalisis untuk memastikan bahwa sistem telah dibuat sesuai rencana penelitian.