IMPLENTASI NODEMCU ESP8266 PADA APLIKASI KONTROL LAMPU JARAK JAUH BERBASIS ANDROID

Skripsi ini diajukan sebagai salah satu syarat
untuk memperoleh gelar sarjana komputer

OLEH :
NIDHO MUDDIN TAMIYAH
NIM : 0402201017

PROGRAM STUDI STRATA (S1) TEKNIK INFORMATIKA
UNIVERSITAS NAHDLATUL ULAMA CIREBON
TAHUN 2024 M/ 1446 H

UNIVERSITAS NAHDLATUL ULAMA CIREBON

PENGESAHAN STATUS SKRIPSI

JUDUL : IMPLENTASI NODEMCU ESP8266 PADA APLIKASI KONTROL LAMPU JARAK JAUH BERBASIS ANDROID

NAMA : NIDHO MUDDIN TAMIYAH

NIM : 0402201017

Mengijinkan Skripsi Sarjana Komputer ini disimpan di perpustakaan Universitas Nahdlatul Ulama dengan syarat-syarat kegunaan sebagai berikut :

1. Skripsi adalah hak milik Universitas Nahdlatul Ulama Cirebon
2. Perpustakaan Universitas Nahdlatul Ulama Cirebon dibenarkan membuat salinan untuk referensi saja.
3. Perpustakaan juga dibenarkan membuat salinan Skripsi ini sebagai bahan pertukaran antara institusi pendidikan tinggi
4. Berikan tanda √ sesuai dengan katagori Skripsi

   Sangat rahasia (Mengandung isi tentang keselamatan atau kepentingan negara Indonesia)

   Rahasia (Mengandung isi tentang kerahasiaan dari sesuatu organisasi/badan dimana penelitian Skripsi ini dikerjakan)

   ✓ Biasa

Disahkan oleh:

Rosidin, S.Kom M. Kom
Tanggal : 24 Agustus 2024

H. Sukarsa, M.Kom
Tanggal : 24 Agustus 2024

UNIVERSITAS NAHDLATUL ULAMA CIREBON

PERSETUJUAN SKRIPSI

JUDUL : IMPLENTASI NODEMCU ESP8266 PADA APLIKASI KONTROL LAMPU JARAK JAUH BERBASIS ANDROID

NAMA : NIDHO MUDDIN TAMIYAH

NIM : 0402201017

Skripsi ini telah diperiksa dan di setujui,

Cirebon,24 Agustus 2024

Pembimbing I	Pembimbing II
Rosidin, S.Kom., M.Kom	H.Sukarsa, M.Kom
NIDN.0418117605	**NIDN.0418117605**

Mengetahui:

Ketua Program Studi

Dicky Andika Sulaeman, M.Kom

NIDN.040906 69402

UNIVERSITAS NAHDLATUL ULAMA CIREBON

PENGESAHAN SKRIPSI

JUDUL : IMPLENTASI NODEMCU ESP8266 PADA APLIKASI KONTROL LAMPU JARAK JAUH BERBASIS ANDROID

NAMA : NIDHO MUDDIN TAMIYAH

NIM : 0402201017

Skripsi ini telah diujikan dan dipertahankan dihadapan Dewan pengujian pada sidang skripsi tanggal 28 Agustus 2024. Menurut pandangan kami, Skripsi ini memadai untuk penganugrahan gelar Sarjana Komputer (S.Kom)

	Tanggal	Tanda Tangan
Pembimbing I Rosidin, S.Kom., M.Kom NIDN. 0415028303	11 September 2024	
Pembimbing II H. Sukarsa, M.Kom NIDN. 0418117605	11 September 2024	
Penguji I Dicky Andika Sulaeman, M. Kom NIDN. 040906940	11 September 2024	
Penguji II Abdul Kohar, M. Kom NIDN. 0412068704	11 September 2024	

Mengetahui,
Dekan Fakultas Ilmu Komputer
Universitas Nahdlatul Ulama Cirebon

Rosidin, S.Kom., M. Kom
NIDN. 0415028303

SURAT PERNYATAAN

Yang bertanda tangan dibawah ini :

Nama	: Nidho Muddin Tamiyah
NIM	: 04022010117
Tempat, Tanggal Lahir	: Cirebon, 9 November 2001
Perguruan Tinggi	: Universitas Nahdlatul Ulama Cirebon

Menyatakan bahwa skripsi dengan judul sebagai berikut :

Judul Bahasa Indonesia	: IMPLENTASI NODEMCU ESP8266 PADA APLIKASI KONTROL LAMPU JARAK JAUH BERBASIS ANDROID
Judul Bahasa Inggris	: IMPLEMENTATION OF ESP8266 NODEMCU IN AN ANDROID-BASED REMOTE LIGHT CONTROL APPLICATION
Pembimbing I	: Rosidin, M.Kom
Pembimbing II	: H. Sukarsa, M.Kom

Adalah benar-benar **ASLI** dan **BELUM PERNAH** dibuat orang lain, kecuali yang dikembagkan dan diacu dalam daftar pustaka pada Skripsi/Tugas Akhir ini.

Demikian Pernyataan SAYA buat, apabila dikemudian hari terbukti SAYA melakukan penjiplakan karya orang lain, maka SAYA BERSEDIA menerima **SANKSI AKADEMIK**

Cirebon, 28 Agustus 2024

Yang menyatakan

Nidho Muddin Tamiyah

NIM 0402201017

ABTRACT

Warung Alfiah, as a small business, faces challenges in managing the use of electrical energy, especially for lighting. These obstacles include the difficulty of accessing light switches that are blocked by stock, wasting electrical energy because the lights are often left on, and difficulty in knowing the status of the lights when the shopkeeper is not in the shop. This study aims to design and implement a remote lighting control system using an Android application as a solution for Warung Alfiah which experiences obstacles in lighting management. The concept of this system involves the use of a control module, namely the NodeMCU ESP8266 which is connected to the shop's internet network, allowing the shopkeeper to send lighting control commands via an Android application. The NodeMCU ESP8266 control module will receive this message and execute the command to adjust the lighting according to needs. The implementation of this simple technology is expected to improve the efficiency of shop management in terms of lighting.

Keywords:

Warung Alfiah, remote lighting control, Android, NodeMCU ESP8266, energy efficiency, lighting management

ABSTRAK

Warung Alfiah, sebagai usaha kecil, menghadapi tantangan dalam mengelola penggunaan energi listrik, terutama untuk pencahayaan. Kendala tersebut meliputi sulitnya mengakses saklar lampu yang terhalangi oleh stok barang, pemborosan energi listrik karena lampu sering dibiarkan menyala, dan kesulitan mengetahui status lampu saat penjaga warung tidak berada di warung. Penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem kontrol lampu jarak jauh menggunakan aplikasi *android* sebagai solusi bagi warung alfiah yang mengalami kendala dalam manajemen pencahayaan. Konsep sistem ini melibatkan penggunaan modul kontrol yaitu *NodeMCU ESP8266* yang terhubung ke jaringan internet warung, memungkinkan penjaga warung untuk mengirimkan perintah pengontrolan lampu melalui aplikasi *adroid*. Modul kontrol *NodeMCU ESP8266* akan menerima pesan ini dan menjalankan perintah untuk mengatur penerangan lampu sesuai kebutuhan. Implementasi teknologi sederhana ini diharapkan dapat meningkatkan efisiensi pengelolaan warung dalam hal pencahayaan.

Kata Kunci:

Warung Alfiah, kontrol lampu jarak jauh, *Android*, *NodeMCU ESP8266*, efisiensi energi, manajemen pencahayaan

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Segala puji kehadirat Allah SWT, yang telah memberikan nikmat sehat wal'afiat, nikmat islam iman dan rahmat serta maghfirah-Nya sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Shalawat serta salam semoga tetap tercurahkan dan limpahkan kepada junjungan nabi kita Rasulullah Muhammad SAW. Amin.

Penulis menyadari bahwa tanpa adanya bimbingan dan dorongan dari berbagai pihak terutama dosen pembimbing yang selalu memberikan arahan dan nasehat selama proses penulisan skripsi ini sehingga dapat diselesaikan dengan baik. Izinkan penulis untuk menyampaikan ucapan terima kasih kepada :

1. Dr. Agus Sugiarto, S.H., M.H., M.M selaku Rektor Universitas Nahdlatul Ulama Cirebon
2. Bapak Rosidin, M.Kom selaku Dekan Fakultas llmu Komputer.
3. Bapak Dicky Andika Sulaeman, M.Kom selaku Ketua Program Studi Teknik Informatika
4. Bapak Rosidin, M.Kom selaku Pembimbing I
5. Bapak H. Sukarsa, M.Kom selaku Pembimbing II
6. Kedua orang tua saya Bapak Ratam dan Ibu Alfiah yang selalu mendo'akan dan memberi semangat sehingga saya berhasil dititik ini.
7. Pimpinan dan dosen beserta seluruh staf Fakultas Ilmu Komputer Universitas Nahdlatul Ulama Cirebon.
8. Rekan-rekan mahasiswa Program Studi Teknik Informatika FILKOM - UNU Cirebon

Penulis juga ingin mennguapkan terima kasih kepada semua pihak yang tidak dapat disebutkan namanya satu per satu, yang telah banyak memberikan dukungan

kepada penulis sehingga skripsi ini dapat terselesaikan. Semoga Allah SWT membalas segala kebaikan dengan pahala yang berlipat ganda. Aamiin.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna, oleh karena itu, penulis sangat mengharapkan kritik dan saran yang bersifat membangun demi perbaikan penulis dimasa yang akan datang. Akhir kata semoga skripsi ini dapat bermanfaat bagi penulis sendiri maupun bagi para pembaca yang tertarik dengan topik ini. Terima kasih.

Cirebon, 28 Agustus 2024

Penulis

Nidho Muddin Tamiyah

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

"Lampu merupakan salah satu peralatan elektronik yang digunakan untuk penerangan dengan memanfaatkan energi listrik. Sistem kontrol yang masih manual sering kali menjadi kendala menghidupkan dan mematikan lampu karena jarak antara saklar lampu satu dengan lainnya yang berjauhan sehingga membutuhkan waktu untuk menghidupkan lampu satu dengan lainnya." (Maheri & Supriyono, 2019)

Internet Of Things, atau dikenal juga dengan *IoT*, *Internet Of Things* atau *IoT* sendiri memepunyai konsep, yaitu dimana beberapa objek tertentu mempunyai kemampuan untuk mengirim data melalui jaringan tanpa perlu adanya interaksi manusia kepada manusia atau dari manusia kepada perangkat komputer. (Ibrahim A. M., 2021)

NodeMCU adalah sebuah *board* elektronik yang berbasis *chip ESP8266* dengan kemampuan menjalankan fungsi mikrokontroler dan juga koneksi internet (WiFi). Terdapat beberapa pin *I/O* sehingga dapat dikembangkan menjadi sebuah aplikasi *monitoring* maupun *controlling* pada proyek *IOT* . (Suryana, 2021)

Warung alfiah dipilih sebagai studi kasus untuk sistem kontrol lampu jarak jauh menggunakan *NodeMCU ESP8266 dan aplikasi Whatsapp*. Dalam konteks ini warung alfiah mengalami kendala dalam mengeola pencahayaan. Contohnya seperti saklar lampu yang terhalangi oleh stok barang, sehingga membuat saklar lampu menjadi sulit untuk di jangkau atau terhalangi oleh barang-barang. Hal ini juga membuat mengakibatkan pemborosan energi listrik karena lampu yang sering dibiarkan menyala. Selain itu, sulitnya mengetahui apakah lampu sedang menyala atau sudah dimatikan ketika pemilik toko tidak berada di warung.

Rancang Bangun Sistem Kontrol Lampu Jarak Jauh menggunakan *NodeMCU ESP8266* dan Aplikasi *Android* adalah sebuah proyek yang bertujuan untuk memberikan solusi bagi warung alfiah dalam mengontrol lampu secara jarak jauh. Dalam hal ini, aplikasi *Android* digunakan sebagai *platform* untuk mengontrol lampu di warung tersebut.

Konsepnya adalah dengan menggunakan modul kontrol yang terhubung ke jaringan internet di warung, pemilik warung dapat mengirimkan perintah pengontrolan lampu melalui aplikasi *android*. Pesan teks ini kemudian akan diterima oleh modul kontrol yang kemudian akan menjalankan perintah tersebut untuk mengatur penerangan lampu di warung.

Studi kasus Warung Alfiah menjadi penting karena menunjukkan implementasi teknologi sederhana namun efektif untuk membantu pemilik warung dalam mengelola penerangan warung mereka, terutama dalam situasi di mana pemilik warung tidak berada di warung.

1.2 Identifikasi Masalah

Identifikasi masalah dari latar belakang yang sudah diuraikan adalah sebagai berikut :

1. Saklar listrik yang sering kali berada di posisi yang sulit dijangkau karena terhalang oleh tumpukan stok barang. Hal ini menyebabkan kesulitan dalam menyalakan atau mematikan lampu saat diperlukan, terutama saat stok barang sedang banyak.
2. Diperlukan solusi yang mudah digunakan oleh pemilik warung tanpa perlu pengetahuan teknis yang mendalam.

1.3 Rumusan Masalah

Berdasarkan latar belakang diatas maka dapat diambil suatu perumusan masalah sebagai berikut :

1. Bagaimana sistem kontrol lampu penerangan dapat diakses menggunakan aplikasi *Android* dengan lokasi yang jauh dan bisa diakses melalui internet?
2. Bagaimana mengintegrasikan modul kontrol dengan jaringan internet yang ada di warung untuk menerima perintah dari aplikasi *Adndroid*?
3. Bagaimana menangani kendala teknis yang mungkin timbul selama pengembangan dan implementasi sistem kontrol lampu ini?

1.4 Batasan Masalah

Pembatasan masalah terhadap penelitian yang dilakukan agar lebih sesuai dan terarah. Adapun batasan-batasan dalam penelitian ini adalah sebagai berikut :

1. Ketersediaan Aplikasi Kontrol: Sistem ini hanya berfungsi apabila aplikasi kontrol tersedia dan dapat diakses oleh pengguna. Jika aplikasi tersebut tidak tersedia atau tidak dapat diakses oleh pemilik warung, sistem kontrol lampu tidak dapat dijalankan.
2. Keterbatasan Fungsionalitas: Sistem ini dirancang khusus untuk mengontrol lampu dan tidak mencakup kontrol untuk perangkat lainnya.
3. Ketergantungan pada Jaringan internet: Sistem memerlukan jaringan internet yang stabil dan dapat diakses di warung untuk beroperasi.

1.5 Tujuan dan Manfaat Penelitian

Adapun tujuan dan manfaat dari penelitian ini adalah sebagai berikut :

1.5.1 Tujuan Penelitian

Tujuan dari penelitian ini adalah :

1. Mengembangkan solusi inovatif untuk memungkinkan penjaga warung mengontrol lampu secara jarak jauh dengan menggunakan aplikasi Android dan melalui internet.
2. Memberikan kemudahan akses dan penggunaan bagi pemilik warung dalam mengontrol lampu, terutama dalam situasi di mana akses fisik ke warung terbatas.
3. Menciptakan sistem yang di mana pemilik atau penjaga warung alfiah dapat dengan mudah mengontrol lampu melalui apliasi *android*, tanpa memerlukan pengetahuan teknis yang mendalam.

1.5.2 Manfaat Penelitian

Berdasarkan uraian yang telah dijelaskan diatas maka manfaat dari penelitian diantaranya :

1. Kemudahan Pengelolaan Warung: Pemilik warung dapat dengan mudah mengontrol lampu mereka tanpa perlu berada di lokasi.
2. Efisiensi Penggunaan Energi: Sistem ini dapat membantu mengoptimalkan penggunaan energi dengan mengatur penerangan sesuai kebutuhan.
3. Implementasi Teknologi Sederhana: Studi kasus ini dapat menjadi contoh bagaimana teknologi sederhana seperti aplikasi *Android* dapat

diimplementasikan dengan efektif untuk memberikan solusi yang bermanfaat dalam kehidupan sehari-hari.

1.6 Studi Literatur

No	Penulis (tahun)	Judul Jurnal	Metode	Hasil
1.	(Maheri & Supriyono, 2019)	Pengontrol Lampu Jarak Jauh Berbasis Web	Eksperimen	Lampu dapat dihidupkan dan dimatikan dari jarak jauh, dan berdasarkan pengujian, fungsi tombol pada web berjalan dengan lancar.
2.	(Arfiansyah, Adriyanto, & Ristyawan, 2023)	Perancangan dan Implementasi Sistem Kendali Lampu Jarak Jauh Berbasis Iot di SDN 2 Mlorah	Agile	Sistem kendali lampu jarak jauh telah berhasil diimplementasikan, memungkinkan petugas sekolah untuk mengontrol lampu dari jarak jauh.
3.	(Abdaoe, Hendi Setiawan, & Perdana.S.T., 2020)	Sistem Kendali Lampu Otomatis Berbasis Iot (Internet Of Things) Menggunakan	WaterFall	Sistem kendali lampu otomatis berbasis IoT (Internet of Things) menggunakan NodeMCU ini

		NodeMCU ESP8266		dapat diimplementasikan untuk mengontrol perangkat elektronik rumah atau ruangan, khususnya pada kendali lampu, sesuai dengan yang diharapkan. Namun, agar sistem berfungsi optimal dan meminimalisir terjadinya kesalahan, diperlukan koneksi internet yang stabil.
4.	(Ibrahim & Setiyadi, 2021)	Prototype Pengendalian Lampu dan AC Jarak Jauh Dengan Jaringan Internet Menggunakan Aplikasi Telegram Berbasis	Pengembangan Prototype	Prototype Pengendalian Lampu dan AC menggunakan Telegram dapat berfungsi untuk menyalakan dan mematikan lampu, mengontrol pendingin ruangan, serta memonitoring kondisi lampu dan

		NodeMCU ESP8266		pendingin ruangan di rumah dari jarak jauh melalui aplikasi Telegram.
5.	(Nasution, Indriani, Fadhilah, Arifin, & Tamba, 2019)	Sistem Kontrol Peralatan Listrik Jarak Jauh Berbasis Arduino	Linear Predictive Coding (LPC)	Sistem Kontrol Peralatan Listrik Jarak Jauh Berbasis Arduino ini memudahkan pemilik rumah dalam mengontrol lampu dari jarak jauh.
Penelitian yang diajukan			Hasil penelitian yang diharapkan	
Implentasi nodemcu esp8266 pada aplikasi kontrol lampu jarak jauh berbasis android			Penelitian ini diharapkan menghasilkan sebuah sistem kontrol lampu jarak jauh yang efektif dan mudah digunakan, berbasis nodemcu esp8266 dan aplikasi android, yang dapat diimplementasikan di Warung Alfiah. Sistem ini akan memungkinkan pengguna untuk mengontrol lampu dari jarak jauh melalui aplikasi android.	

Tabel 1.1 Review Penelitian

1.7 Metode Penelitian

Adapun metode penelitian yang digunakan oleh penulis sebagai berikut :

1. Studi Literatur: Melakukan studi literatur untuk memahami konsep-konsep dasar yang terkait dengan sistem kontrol penerangan, penggunaan aplikasi *Android*, dan teknologi terkait lainnya.
2. Studi Pendahuluan: Melakukan studi pendahuluan untuk mengidentifikasi kebutuhan dan kendala yang mungkin dihadapi dalam pengembangan sistem kontrol penerangan ini.
3. Perancangan Sistem: Merancang sistem kontrol penerangan yang mencakup perancangan modul kontrol, integrasi dengan jaringan internet, dan mekanisme pengontrolan lampu.
4. Implementasi Sistem: Mengimplementasikan sistem kontrol penerangan yang telah dirancang ke dalam warung Alfiah dan melakukan uji coba fungsionalitas.
5. Analisis Data: Menganalisis data yang diperoleh dari uji coba untuk mengevaluasi kinerja sistem kontrol penerangan ini, termasuk efektivitas, efisiensi, dan keandalannya.
6. Evaluasi: Melakukan evaluasi terhadap hasil penelitian.

1.8 Sistematika Penulisan

Laporan skripsi ini disusun dalam lima bab dengan sistematika penulisan sebagai berikut :

BAB I PENDAHULUAN

Dalam BAB I dibahas berbagai hal yang berkaitan dengan masalah yang akan dibahas dalam penulisan ilmiah ini, diantaranya latar belakang masalah, identifikasi masalah, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, studi literatur, metode penelitian dan sistematika penulisan

BAB II LANDASAN TEORI

Dalam bab II ini akan menjelaskan kajian jurnal yang terkait, teori utama penelitian, teori pendukung penelitian yang dapat diperoleh dari berbagai sumber terpercaya seperti buku, jurnal prosiding maupun laporan.

BAB III ANALISIS MASALAH DAN PERANCANGAN PROGRAM

Dalam BAB III akan membahas tentang lingkungan pengembangan perangkat keras, alasan pemilihan interface yang digunakan, rangkaian atau diagram yang menjadi dasar dalam perancangan interface, dan garis besar alur kerja dari interface yang dibangun.

BAB IV IMPLEMENTASI DAN PENGUJIAN

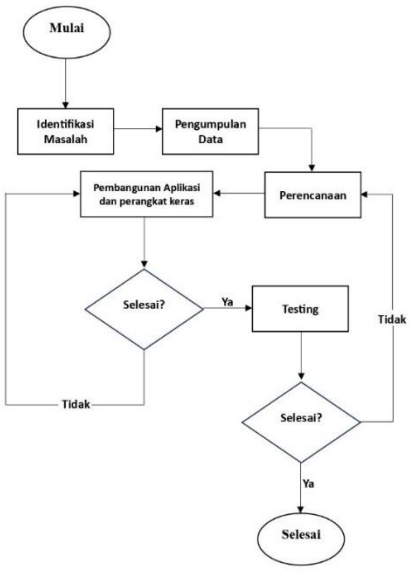
Dalam BAB IV akan dijelaskan mengenai hasil implementasi dari interface yang dikembangkan, pengujian secara simulasi, pengujian dalam situasi nyata, cara penggunaan interface yang telah dikembangkan, beserta hasil Analisa yang dicapai dari interface yang telah dibuat.

BAB V PENUTUP

Dalam BAB V ini dijelaskan tentang kesimpulan dan saran-saran dari penulis tentang penulisan ilmiah ini.

1.9 Kerangka Berpikir

Kerangka berpikir adalah gambaran atau bagan dari proses penelitian yang mendeskripsikan suatu analis pada suatu proses penelitian. Berikut adalah gambar kerangka berpikir :



Gambar 1.1Kerangka Berpikir

Gambar di atas ini adalah sebuah proses pengembangan rancang bangun sistem kontrol lampu Jarak jauh menggunakan nodemcu esp8266 dan aplikasi *android*, yang melibatkan beberapa tahapan utama dari awal hingga akhir. Berikut adalah penjelasan untuk tiap bagian proses dalam gambar di atas ini:

1. Mulai:

   Ini adalah titik awal dari seluruh proses pengembangan. Proyek dimulai dari sini setelah keputusan diambil untuk melakukan pengembangan.

2. Identifikasi Masalah:

Pada tahap ini, masalah yang ingin dipecahkan oleh proyek diidentifikasi. Ini melibatkan pemahaman mendalam tentang kebutuhan atau masalah spesifik yang ada, sehingga solusi yang tepat dapat dirancang.

3. Pengumpulan Data:

Setelah masalah diidentifikasi, data yang relevan dikumpulkan untuk memberikan informasi yang dibutuhkan selama proses pengembangan. Data ini bisa berupa persyaratan teknis, kebutuhan pengguna, atau informasi lain yang penting untuk menyelesaikan masalah.

4. Perencanaan:

   Tahap ini melibatkan penyusunan rencana untuk mengatasi masalah yang telah diidentifikasi. Rencana ini mencakup bagaimana aplikasi dan perangkat keras akan dikembangkan, sumber daya yang diperlukan, serta langkah-langkah yang harus diambil untuk mencapai tujuan proyek.

5. Pembangunan Aplikasi dan Perangkat Keras:

   Pada tahap ini, rencana yang telah dibuat diimplementasikan. Pengembangan aplikasi dan perangkat keras dilakukan sesuai dengan spesifikasi yang telah direncanakan sebelumnya. Ini adalah tahap eksekusi di mana produk atau solusi yang diinginkan mulai dibentuk.

6. Selesai? (Pertama):

   Setelah pembangunan dilakukan, ada evaluasi pertama untuk memeriksa apakah pekerjaan sudah selesai atau belum. Jika belum, proses kembali ke tahap perencanaan untuk memperbaiki atau menyempurnakan bagian-bagian yang perlu diperbaiki.

7. Testing:

Jika pembangunan dianggap selesai, proyek dilanjutkan dengan tahap pengujian. Pada tahap ini, aplikasi dan perangkat keras yang telah dikembangkan diuji untuk memastikan semuanya berfungsi sesuai dengan yang diharapkan.

8. Selesai? (Kedua):

Setelah pengujian, dilakukan evaluasi akhir untuk menentukan apakah proyek sudah benar-benar selesai dan memenuhi semua kriteria yang telah ditetapkan. Jika ya, proyek dianggap selesai dan proses berakhir. Jika tidak, maka kembali dilakukan revisi dan pengujian ulang hingga hasilnya memuaskan.

9. Selesai:

Ini adalah titik akhir dari proses pengembangan, di mana proyek dianggap berhasil diselesaikan setelah melalui semua tahapan sebelumnya. Produk atau solusi yang dikembangkan siap untuk digunakan atau diluncurkan.

BAB II
LANDASAN TEORI

2.1 Pandangan Agama Islam Tentang Perkembangan Teknologi

Perkembangan ilmu pengetahuan dan teknologi (IPTEK) dalam Islam memerlukan evaluasi mendalam dan refleksi terhadap faktor-faktor yang telah menyebabkan kemunduran di bidang ini. Langkah pertama yang harus dilakukan adalah introspeksi terhadap keterasingan kita dari nilai-nilai moral yang terkandung dalam pengetahuan dan ajaran Islam, sebagaimana yang dianjurkan oleh Al-Qur'an dan sunnah Nabi. Nilai-nilai ini merupakan modal utama yang seharusnya menjadi landasan dalam pengembangan IPTEK. Selain itu, penting untuk mengatasi pertentangan ideologis dan politik yang terjadi di antara umat Islam dari berbagai bangsa dan negara. Perselisihan ini tidak hanya menghambat kerjasama dan solidaritas, tetapi juga menghalangi terciptanya suasana kondusif untuk perkembangan ilmu pengetahuan. Dengan menghilangkan konflik-konflik tersebut, umat Islam dapat membangun fondasi yang lebih kuat untuk mengejar kemajuan di bidang IPTEK. Selanjutnya, masyarakat Islam perlu menghidupkan kembali tradisi berpikir yang bebas, kritis, dan independen. Tradisi ini akan mendorong individu untuk secara aktif mencari dan menggali informasi, serta mengembangkan pengetahuan yang diperlukan untuk kemajuan umat. Kebebasan berpikir yang sehat adalah kunci dalam menciptakan inovasi dan penemuan baru yang dapat memperkaya khazanah ilmu pengetahuan Islam.

Dengan demikian, ilmu yang ditanamkan dalam Islam bermanfaat tidak hanya bagi agama kita tetapi juga bagi umat manusia. Selain itu, Islam juga mendorong umatnya untuk melakukan penelitian yang tetap menggunakan Al-Qur'an sebagai pedoman ilmiah. Hal ini pula yang mendorong umat Islam untuk mengembangkan sifat-sifat ilmuwan yaitu kritis. كُلُّ وَٱلْفُؤَادَ وَٱلْبَصَرَ ٱلسَّمْعَ إِنَّ ۚ عِلْمٌ بِهِۦ لَكَ لَيْسَ مَا تَقْفُ وَلَا مَسْـُٔولًا عَنْهُ كَانَ أُوْلَٰٓئِكَ (QS. Al-Isra/17: 36), terbuka menerima kebenaran dari manapun datangnya ilmu tersebut اللهُ هَدٰىهُمُ الَّذِيْنَ أُولٰٓئِكَ اَحْسَنَهٗۗ فَيَتَّبِعُوْنَ الْقَوْلَ يَسْتَمِعُوْنَ الَّذِيْنَ الْأَلْبَابِ أُولُوا هُمْ وَأُولَٰٓئِكَ (QS. Az-Zumar/39: 18), dan senantiasa menggunakan akal pikirannya untuk berpikir secara kritis وَاٰخِرُ سَلٰمٌۚ فِيْهَا وَتَحِيَّتُهُمْ اللّٰهُمَّ سُبْحٰنَكَ فِيْهَا دَعْوٰىهُمْ الْعٰلَمِيْنَ رَبِّ لِلّٰهِ الْحَمْدُ اَنِ دَعْوٰىهُمْ (QS. Yunus/10: 10). Inilah yang mengantarkan pada

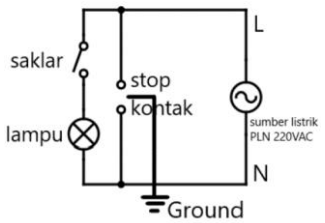
sebuah keharusan bagi setiap umat muslim agar mampu unggul dalam bidang Ilmu Pengetahuan dan Teknologi (IPTEK) sebagai sarana kehidupan yang harus diutamakan untuk mencapai kebahagiaan baik di dunia maupun di akhirat فِيْمَآ وَابْتَغِ الْاَرْضِۗ فِى الْفَسَادَ تَبْغِ وَلَا اِلَيْكَ اللّٰهُ اَحْسَنَ كَمَآ وَاَحْسِنْ الدُّنْيَا مِنَ نَصِيْبَكَ تَنْسَ وَلَا الْاٰخِرَةَ الدَّارَ اللّٰهُ اٰتٰىكَ الْمُفْسِدِيْنَ يُحِبُّ لَا اللّٰهَ اِنَّ QS. Al-Qashash/28: 77 (Mohamad Rizky Ramadhandy Budianto, 2021)

2.2 Lampu Jarak Jauh

Lampu pertama kali diciptakan oleh Thomas Alva Edison dalam bentuk lampu pijar yang sangat sederhana. Namun, seiring dengan kemajuan teknologi, jenis dan bentuk lampu yang digunakan dalam kehidupan sehari-hari telah berkembang pesat. Salah satu merek lampu, yaitu Philips, telah mengeluarkan berbagai produk, mulai dari lampu bohlam, lampu TL, hingga yang terbaru, lampu LED.

Lampu dapat didefinisikan sebagai komponen perlengkapan pencahayaan yang memancarkan cahaya. Ini adalah bagian dari perlengkapan yang menghasilkan pencahayaan yang sebenarnya. Lampu tersedia dalam berbagai jenis, termasuk lampu pijar, lampu neon, atau LED. (Dia, 2023)

Umumnya pengendalian hidup/mati lampu dilakukan dengan saklar listrik yang menyalakan dan mematikan lampu. Saat ini, dimungkinkan untuk membangun perangkat kendali pencahayaan jarak jauh menggunakan teknologi mikrokontroler NodeMCU ESP8266. Pencahayaan jarak jauh adalah sistem pencahayaan yang dapat dikontrol dari jarak jauh dari perangkat seperti ponsel pintar, tablet, dan komputer. Sistem ini memungkinkan pengguna menyalakan dan mematikan lampu tanpa menggunakan saklar listrik.



Gambar 2.1Wiring Diagram Lampu dan Saklar

(sumber gambar https://www.s-gala.com)

2.3 *Internet Of Things*

Internet of Things (IoT) adalah sebuah konsep di mana berbagai perangkat fisik terhubung ke Internet dan dapat berkomunikasi satu sama lain serta bertukar data. Perangkat ini mungkin mencakup berbagai objek sehari-hari yang dapat mengumpulkan dan mengirimkan data melalui jaringan, seperti sensor, kendaraan, dan peralatan rumah tangga.

Menurut analisa (McKinsey, 2024), *Internet of Things* adalah teknologi yang memungkinkan mesin, perangkat, dan objek fisik lainnya terhubung ke sensor jaringan untuk menangkap data dan mengelola kinerjanya sendiri. Hal ini memungkinkan mesin untuk bekerja sama dan bahkan bereaksi terhadap informasi yang baru diperoleh secara independen satu sama lain. Tujuannya untuk memudahkan manusia dalam berinteraksi dengan suatu benda sehingga benda tersebut dapat berkomunikasi dengan benda lainnya. *Internet of Things* adalah revolusi teknologi yang mewakili masa depan komputasi dan komunikasi, dan perkembangannya bergantung pada inovasi dinamis di berbagai bidang penting, mulai dari komunikasi *nirkabel* hingga sensor dan nanoteknologi.

Internet of Things (IoT) merupakan perkembangan teknologi yang menjanjikan dapat mengoptimalkan kehidupan dengan sensor sensor cerdas dan benda yang memiliki jaringan dan bekerjasama dalam internet. Internet of Things (IoT) secara sederhana adalah suatu cara untuk menghubungkan peralatan elektronik ke internet dan mengontrolnya dari seluruh dunia selama 24 jam non stop. (Ruuhwan, 2019)

2.4 NodeMCU ESP8266

NodeMCU ESP8266 adalah sebuah *platform* pengembangan yang menggabungkan mikrokontroler ESP8266 dengan modul *Wi-Fi*, dirancang untuk proyek Internet of Things (IoT). *Platform* ini memungkinkan konektivitas jaringan tanpa memerlukan modul tambahan, berkat *Wi-Fi* yang ada pada bawaannya. NodeMCU mendukung pemrograman menggunakan bahasa *Lua* atau *C/C++* melalui *Arduino IDE*, dan menyediakan berbagai pin *GPIO* untuk *input/output*, serta mendukung protokol komunikasi seperti *UART*, *SPI*, dan *I2C*. Dengan harga yang terjangkau, dukungan komunitas yang luas, dan sifat *open-source*, NodeMCU

ESP8266 menjadi pilihan populer untuk berbagai aplikasi seperti otomatisasi rumah, *monitoring* lingkungan, dan kontrol perangkat jarak jauh. Dalam penelitian kontrol lampu jarak jauh menggunakan NodeMCU ESP82666 dan aplikasi *Whatsapp*, NodeMCU ESP8266 berfungsi sebagai pengendali utama yang menerima perintah melalui internet dan mengendalikan lampu sesuai dengan perintah yang diterima.

ESP8266 adalah komponen chip terintegrasi yang dirancang untuk memenuhi kebutuhan dunia yang terhubung saat ini. Chip ini menyediakan solusi jaringan Wi-Fi terintegrasi lengkap yang dapat digunakan sebagai penyedia aplikasi atau untuk memisahkan semua fungsi jaringan Wi-Fi dari pemroses aplikasi lainnya. ESP8266 memiliki kemampuan pemrosesan dan memori internal yang memungkinkan chip diintegrasikan ke dalam sensor atau aplikasi perangkat tertentu dengan pemrograman sederhana melalui pin input/output. NodeMCU adalah platform IoT sumber terbuka. Terdiri dari perangkat keras berupa sistem-on-a-chip ESP8266 Espresif Systems dan firmware yang menggunakan bahasa pemrograman skrip Lua. Namun, NodeMCU telah mengemas ESP8266 ke dalam papan kompak dengan berbagai fitur seperti mikrokontroler, aksesibilitas Wi-Fi, dan chip komunikasi USB-ke-serial. Yang Anda perlukan untuk pemrograman hanyalah kabel ekstensi kabel data USB, yang juga berfungsi sebagai kabel data dan pengisi daya untuk ponsel pintar Android Anda. (Pratama, 2018).

2.4.1 Fungsi *NodeMCU*

1. Pegembangan IOT (Internet Of Things)

   NodeMCU dilengkapi dengan modul ESP8266 yang memungkinkan konektivitas *WiFi*, membuatnya cocok untuk bergabagai macam proyek *IoT* yang memerlukan koneksi internet. NodeMCU juga dapat digunakan untuk mengumpulkan data dari sensor dan mengirimkannya ke *server* atau layanan *cloud* untuk pemrosesan lebih lanjut.

2. Prototipe yang cepat

NodeMCU dirancang untuk dapat dihubungkan langsung ke *breadboard*, memudahkan pengembangan dan pengujian prototipe tanpa perlu *menyolder*. Mendukung berbagai macam *library* yang mempermudah *integrasi* dengan sensor, aktuator, dan layanan *cloud*. NodeMCU juga memiliki komunitas yang besar. Sehingga memiliki banyak sumber daya dan contoh proyek.

3. Pemrograman yang mudah

NodeMCU dapat diprogram menggunakan bahasa *scripting Lua* dan C++, yang mudah dipelajari dan digunakan. NodeMCU juga mendukung pemrograman melalui *Arduino IDE*, memungkinkan pengembang dapat menggunakan ekosistem yang ada pada *Arduino IDE*.

4. Fitur pada *hardware*

NodeMCU memiliki beberapa *General Purpose Input/Output* (GPIO) pin yang dapat digunakan untuk menghubungkan berbagai sensor dan aktuator.

5. Penggunaan dalam berbagai proyek

NodeMCU sangat cocok untuk proyek rumah pintar, seperti mengendalikan lampu, termostat, dan perangkat rumah tangga lainnya melalui internet. Menggunakan sensor untuk memantau kondisi lingkungan seperti suhu, kelembaban, dan kualitas udara, serta mengirimkan data tersebut secara *real-time*.

2.4.2 Versi *NodeMCU*

Bagi pengguna yang masih awal masih cukup bingung dengan beberapa *board* NodeMCU. Karena sifatnya yang *open source* tentu akan banyak produsen yang memproduksinya dan mengembangkannya. Menurut Tresna Widiyaman NodeMCU terdapat tiga versi, yaitu V1, V2 dan V3.

Generasi	Versi	Nama
1	0.9	V1

2	1.0	V2
3	1.0	V3,Lolin

Tabel 2.1 Versi NodeMCU

1. NodeMCU 0.9

NodeMCU 0.9 adalah versi pertama yang dilengkapi dengan memori *flash* sebesar 4 MB sebagai System on Chip (SoC) dan menggunakan modul ESP-12. Meskipun memiliki keunggulan dalam hal kapasitas memori, versi ini memiliki kelemahan dari segi ukuran modul yang cukup lebar. Hal ini menjadi kendala ketika digunakan pada breadboard untuk pembuatan prototipe, karena pin-pin pada *breadboard* akan sepenuhnya terpakai hanya untuk modul ini, sehingga mengurangi fleksibilitas dalam pengembangan lebih lanjut.

Gambar 2.2 NodeMCU 0.9

(Sumber gambar https://www.electronicwings.com/)

Mikrokontroller	**ESP8266**
Input Tegangan	4.5V–9V
Ukuran Board	48 mm x 25.4 mm
GPIO	17 pin
Flash Memory	4 MB
Wireless	802.11 b\g\n standard
USB to Serial converter	CP2102

Tabel 2.2 Spesifikasi NodeMCU 0.9

2. NodeMCU 1.0

NodeMCU 1.0 merupakan evolusi dari versi 0.9 dengan beberapa perbaikan penting. Pada versi 1. 0, modul ESP8266 menggunakan tipe ESP-12E yang

dinilai lebih stabil dibandingkan versi ESP-12 sebelumnya. Selain itu, ukuran papan modul telah diperkecil, sehingga lebih cocok untuk proyek pembuatan prototipe papan tempat memotong roti. Rilis ini juga memperkenalkan pin tambahan untuk komunikasi SPI (Serial Peripheral Interface) dan PWM (Pulse Wide Modulation) yang tidak tersedia di versi 0.9. Peningkatan ini membuat NodeMCU 1.0 lebih fleksibel dan mampu mengembangkan berbagai aplikasi IoT.

Gambar 2.3 NodeMCU 1.0

(Sumber gambar https://id.wikipedia.org/)

Mikrokontroller	**ESP8266**
Input Tegangan	4.5V–10V
Ukuran Board	58 mm x 31 mm
GPIO	17 pin
Flash Memory	4 MB
Wireless	802.11 b\g\n standard
USB to Serial converter	CH340G

Tabel 2.3 Spesifikasi NodeMCU 1.0

3. NodeMCU 1.0 unofficial atau V3

NodeMCU 1.0 (Unofficial Board) Produk modul ini disebut dengan unofficial board karena dibuat secara tidak resmi dengan persetujuan dari pengembang resmi NodeMCU. Setidaknya hingga artikel ini ditulis, belum ada versi resmi dari V3 NodeMCU. V3 dibuat oleh pembuat LoLin dan hanyalah versi V2 yang sedikit lebih baik. Mungkin antarmuka USB yang lebih cepat. (Suryana, 2021)

NodeMCU V3 ESP8266 adalah papan pengembangan sumber terbuka yang dapat digunakan untuk membangun aplikasi IoT hanya dengan beberapa baris skrip bahasa Lua. Otak dari board ini adalah chip ESP8266, sebuah mikrokontroler yang diproduksi di pabrik Espresif System di Shanghai, Cina. Chip ini dilengkapi dengan fungsi Wi-Fi dan TCP/IP.

Gambar 2.4 NodeMCU 1.0 *unofficia*l atau *V3*

(Sumber gambar https://elektrologi.iptek.web.id/)

Mikrokontroller	**ESP8266**
Input Tegangan	4.5V–10V
Ukuran Board	58 mm x 31 mm
GPIO	17 pin
Flash Memory	4 MB
Wireless	802.11 b\g\n standard
USB to Serial converter	CH340G

Tabel 2.4 Spesifikasi NodeMCU 1.0 unofficial atau V3

Pada penelitian ini *NodeMCU* V3 dipilih dikarenakan memiliki sejumlah keunggulan. *ESP8266* memiliki modul *Wi-Fi* terintegrasi, memungkinkan konektivitas internet yang mudah dan mendukung aplikasi *IoT*. Dengan biaya rendah, *NodeMCU* V3 cocok untuk proyek dengan anggaran terbatas, serta antarmuka yang mudah digunakan dan dukungan dari *platform* pengembangan populer seperti *Arduino IDE*, mempercepat proses pengembangan. Secara keseluruhan, *NodeMCU V3 ESP8266* adalah pilihan efisien, hemat biaya, dan mudah dikembangkan untuk penelitian ini.

2.5 *Arduino IDE*

"Arduino IDE (Integrated Development Environment) adalah *software* yang di gunakan untuk memprogram di *arduino*, dengan kata lain Arduino IDE sebagai media untuk memprogram *board Arduino*. Arduino IDE bisa di download secara gratis di website resmi Arduino IDE. Arduino IDE ini berguna sebagai *text* editor untuk membuat, mengedit, dan juga mevalidasi kode program. bisa juga digunakan untuk meng-*upload* ke *board Arduino*. Kode program yang digunakan pada *Arduino* disebut dengan istilah *Arduino "sketch"* atau disebut juga *source code arduino*, dengan ekstensi *file source code ino*. *Arduino* merupakan *software* yang telah disiapkan oleh *arduino* bagi para perancang untuk melakukan berbagai proses yang berkaitan dengan pemrograman *Arduino*. IDE ini juga sudah mendukung berbagai sistem operasi populer saat ini seperti *Windows*, *Mac*, *Linux*, dan *Android*." (Chandra, 2019)

Seperti *Arduino IDE*, editor pemrograman biasanya memiliki fungsi potong/tempel teks dan cari/ganti. Aplikasi ini memberikan umpan balik dan menampilkan pesan kesalahan di area deskripsi saat menyimpan dan mengekspor proyek. Konsol *log Arduino IDE* menampilkan teks *log* aktivitas yang dilakukan, termasuk pesan kesalahan terperinci dan informasi lainnya. Di pojok kanan bawah terdapat indikator *port* serial yang sedang digunakan. *Toolbar Arduino IDE* dilengkapi dengan ikon *shortcut keyboard* untuk meninjau dan mengunggah program, membuat, membuka, dan menyimpan sketsa. Selain itu, Anda dapat dengan mudah mengakses *monitor* serial dari *toolbar*. *Arduino IDE* merupakan aplikasi yang mencakup editor, *compiler*, dan *uploader*, serta mendukung semua rangkaian modul dalam keluarga *Arduino*, antara lain: Contoh: *Arduino Duemilanove*, *Uno*, *Bluetooth*, *Mega*. Namun ada beberapa jenis *board Arduino* yang menggunakan mikrokontroler selain seri AVR, seperti: B. Mikroprosesor ARM tidak didukung. Editor sketsa *Arduino IDE* juga menyertakan penomoran baris dan penyorotan sintaksis, pemeriksa sintaksis kode sketsa yang memudahkan pengembang untuk menulis dan menganalisis kode. Fitur-fitur ini menjadikan *Arduino IDE* alat yang efektif dan efisien ketika mengembangkan proyek berbasis mikrokontroler. Berikut adalah fungsi dari *Arduino IDE* :

1. Menulis Kode

Menyediakan area untuk menulis kode program menggunakan bahasa pemrograman *C++* yang disederhanakan dan menyorot kata kunci dan struktur kode dengan warna yang berbeda untuk memudahkan pembacaan dan debugging. Serta saran kode secara otomatis saat mengetik, sehingga mengurangi kemungkinan kesalahan pengetikan.

2. Mengkompilasi Kode

Mengubah kode yang ditulis dalam bahasa *C++* menjadi bahasa mesin yang dapat dimengerti oleh mikrokontroler *Arduino*.

3. Mengunggah Kode

Mengunggah ke Board: Mengirimkan kode yang telah dikompilasi ke mikrokontroler Arduino melalui port USB. Proses ini disebut sebagai "burning" atau "flashing".

4. Memeriksa Kesalahan

Menunjukkan kesalahan sintaks atau logika dalam kode program sehingga dapat segera diperbaiki dan membantu melacak dan memperbaiki masalah dalam kode program.

5. Mengelola Proyek:

Memungkinkan pengguna untuk membuat proyek baru dengan pengaturan yang berbeda-beda. Serta membuka kembali proyek yang telah dibuat sebelumnya untuk diedit atau dijalankan dan menyimpan kode program dan pengaturan proyek untuk digunakan di kemudian hari.

Gambar 2.5Arduino IDE

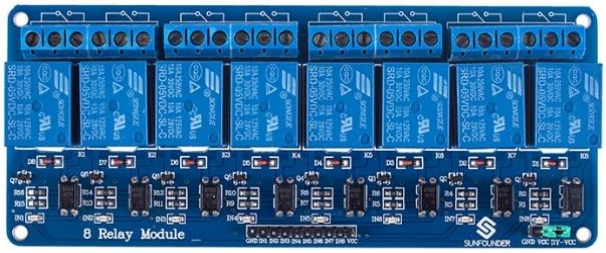
(Sumber gambar https://iconduck.com/)

2.6 *Relay*

Relay adalah saklar elektronik yang dapat membuka atau menutup suatu rangkaian dengan menggunakan kendali rangkaian elektronik lain. Relay terdiri dari kumparan, pegas, saklar (terhubung ke pegas), dan dua kontak elektronik (biasanya tertutup dan biasanya terbuka). (Indra Gunawan, 2020)

Relay dapat digunakan untuk mengendalikan motor *AC* dengan rangkaian kendali *DC* atau beban lain dengan sumber tegangan yang berbeda antara tegangan rangkaian kendali dengan tegangan beban. Kemungkinan penerapan relai antara lain: Relay sebagai pengatur hidup/mati beban dengan sumber tegangan berbeda. Relay sebagai penyeleksi atau penyeleksi relasional. Relay sebagai pelaksana rangkaian tunda. Relay sebagai proteksi atau pemutus arus pada kondisi tertentu. (Hakim, 2018)

Relay adalah komponen elektronik yang berfungsi sebagai saklar elektromekanis yang dikendalikan oleh sinyal listrik. Ketika sinyal listrik kecil diberikan ke *relay*, itu akan mengaktifkan mekanisme internal yang dapat menghidupkan atau mematikan sirkuit yang dikendalikan, biasanya dengan tegangan dan arus yang lebih besar. Relay digunakan untuk mengontrol perangkat yang memerlukan daya tinggi menggunakan sinyal daya rendah.



Gambar 2.6Relay

(Sumber gambar https://www.tokopedia.com/)

2.7 Kabel Listrik

Kabel listrik adalah jenis kabel yang digunakan untuk menghantarkan listrik dari satu titik ke titik lainnya. Kabel listrik memiliki berbagai macam bentuk dan ukuran, tergantung pada kebutuhan penggunaannya. Kabel listrik terdiri dari satu atau lebih konduktor yang diisolasi dengan bahan non-konduktif untuk melindungi konduktor dan pengguna dari bahaya listrik.

Kabel listrik berfungsi sebagai sarana untuk menghantarkan energi listrik. Secara umum, kabel listrik terdiri dari sejumlah kawat yang diisolasi, yang disatukan untuk membentuk jalur transmisi dengan banyak konduktor. Oleh karena itu, setiap kabel listrik memiliki dua komponen utama: konduktor dan isolator. Isolator, yang membungkus konduktor, biasanya terbuat dari bahan seperti thermoplastik atau thermosetting yang tidak menghantarkan listrik. Fungsi isolator ini adalah sebagai pelindung agar manusia tidak tersengat listrik saat menyentuh kabel. (Anggraini, 2022). Berikut adalah jenis kabel listrik :

1. Kabel *NYA*

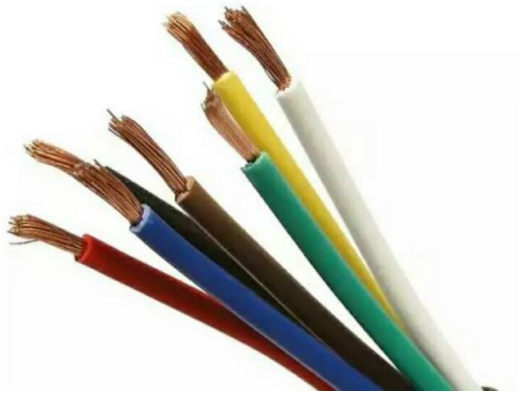
   Kabel *NYA* umumnya digunakan dalam instalasi listrik *indoor*, terutama di lingkungan perumahan, karena harganya yang relatif terjangkau. Kabel ini memiliki inti tunggal dengan lapisan isolasi berbahan *PVC* satu lapis, yang menjadikannya rentan terhadap kerusakan dan kurang tahan terhadap air. Oleh karena itu, untuk meningkatkan keamanan penggunaan kabel *NYA*, disarankan agar kabel ini dipasang di dalam pipa jenis *PVC*.

Gambar 2.7Kabel *NYA*

(Sumber Gambar https://biz.kompas.com/)

2. Kabel *NYAF*

Kabel *NYAF* adalah jenis kabel listrik yang dirancang untuk instalasi tetap dalam kondisi kering, dengan konduktor yang fleksibel dan terdiri dari banyak serabut tembaga halus. Nama *NYAF* sendiri merupakan singkatan dari beberapa kata dalam bahasa Jerman, yaitu "N" yang berarti Normenleitung atau kabel standar, "Y" yang menunjukkan isolasi PVC, "A" yang menandakan bahwa kabel ini adalah jenis tunggal, dan "F" yang mengindikasikan bahwa kabel ini fleksibel. Kabel NYAF biasanya digunakan dalam instalasi listrik yang memerlukan kabel dengan fleksibilitas tinggi, seperti dalam panel kontrol, panel distribusi, dan peralatan industri lainnya.

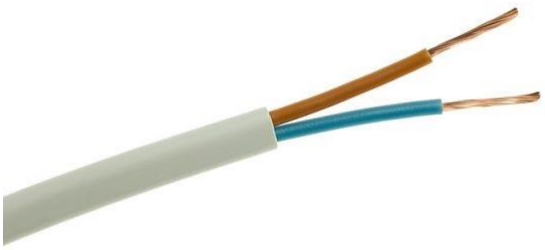


Gambar 2.8Kabel *NYAF*

(Sumber Gambar https://biz.kompas.com/)

3. Kabel *NYM*

Kabel *NYM* adalah jenis kabel listrik yang dirancang untuk instalasi tetap di dalam bangunan, baik untuk penggunaan dalam ruangan maupun luar ruangan dengan perlindungan tambahan. Kabel ini memiliki inti tembaga yang terdiri dari satu atau lebih konduktor, biasanya dilapisi dengan isolasi *PVC* yang ganda untuk memberikan perlindungan tambahan terhadap kerusakan fisik dan lingkungan. Isolasi *PVC* yang tebal pada kabel *NYM* membuatnya tahan terhadap kelembapan, panas, dan bahan kimia tertentu, sehingga cocok untuk instalasi di tempat-tempat yang memerlukan perlindungan ekstra.



Gambar 2. 9 Kabel *NYM*

(Sumber gambar https://www.tokopedia.com/)

4. Kabel *NYY*

Kabel *NYY* adalah jenis kabel listrik yang dirancang untuk instalasi tetap di dalam dan di luar ruangan, termasuk penanaman langsung di dalam tanah. Kabel ini terdiri dari satu atau lebih konduktor tembaga yang diisolasi dengan lapisan *PVC* dan dilengkapi dengan selubung luar *PVC* yang tebal dan kuat. Kabel *NYY* terkenal karena ketahanannya terhadap kelembapan, sinar *UV*, dan kondisi cuaca ekstrem, sehingga sering digunakan dalam aplikasi industri dan komersial yang memerlukan perlindungan tambahan. Selain itu, kabel *NYY* juga tahan terhadap tekanan mekanis, sehingga cocok untuk instalasi di bawah tanah tanpa memerlukan perlindungan tambahan seperti pipa atau konduit.

Gambar 2.10 Kabel *NYY*

(Sumber Gambar https://e-katalog.lkpp.go.id/)

5. Kabel *NYYHY*

Kabel *NYYHY* adalah jenis kabel listrik fleksibel yang dirancang untuk instalasi tetap dalam kondisi yang memerlukan fleksibilitas tinggi. Kabel ini terdiri dari konduktor tembaga serabut yang dilapisi dengan isolasi *PVC,* yang kemudian dilindungi oleh selubung luar *PVC* yang fleksibel dan tahan lama. Kabel *NYYHY* sering digunakan dalam instalasi listrik dalam ruangan seperti untuk menghubungkan peralatan elektronik, panel kontrol, dan instalasi penerangan di tempat yang memerlukan banyak tikungan atau pergerakan. Isolasi *PVC* yang digunakan pada kabel *NYYHY* memberikan perlindungan yang baik terhadap kelembapan, bahan kimia, dan abrasi, membuatnya cocok untuk berbagai aplikasi industri dan komersial.

Gambar 2.11 Kabel *NYYHY*

(Sumber Gambar https://inet.detik.com/)

6. Kabel *NYFGbF*

Kabel *NYFGbF* adalah jenis kabel listrik yang dirancang untuk instalasi tetap di lingkungan industri yang memerlukan perlindungan ekstra terhadap tekanan mekanis dan kondisi lingkungan yang keras. Kabel ini memiliki konduktor tembaga yang dilapisi dengan isolasi *PVC*, yang kemudian dilindungi oleh lapisan baja *galvanis* dan selubung luar *PVC*. Lapisan baja galvanis memberikan perlindungan mekanis yang sangat kuat terhadap kerusakan fisik, menjadikannya ideal untuk penanaman langsung di tanah, instalasi di area yang rawan gangguan fisik, dan penggunaan di lingkungan yang memerlukan keamanan tambahan. Kabel *NYFGbF* juga tahan terhadap kelembapan, minyak, dan bahan kimia, sehingga cocok untuk aplikasi di pabrik, tambang, dan instalasi bawah tanah.

Gambar 2.12 Kabel *NYFGbF*

(Sumber Gambar https://e-katalog.lkpp.go.id/)

Pada penelitian ini kabel *NYM* dipilih karena beberapa alasan teknis dan praktis yang membuatnya sangat sesuai untuk penggunaan dalam instalasi kontrol lampu jarak jauh menggunnakan *NodeMCU ESP8266* dan aplikasi whatsapp. Kabel *NYM* adalah jenis kabel listrik yang dirancang khusus untuk instalasi di dalam ruangan dengan isolasi yang kuat dan perlindungan terhadap gangguan mekanis dan lingkungan. Dengan lapisan isolasi *PVC* yang tebal, kabel ini memilik ketahanan yang baik terhadap suhu tinggi, kelembaban, dan bahan kimia. Selain itu, kabel *NYM* memiliki konduktivitas yang baik, yang memastikan pengaliran arus listrik yang stabil dan aman ke perangkat seperti *relay* dan *NodeMCU ESP8266*. Pilihan

kabel ini juga didukung oleh standar *SNI* yang memastikan kualitas dan keamanan penggunaan.

2.8 Kabel *Jumper*

"Kabel *jumper* adalah kabel pendek yang digunakan untuk menghubungkan komponen elektronik atau titik-titik dalam rangkaian pada *breadboard* atau prototipe. Kabel ini memungkinkan penghubungan listrik sementara tanpa perlu penyolderan. Terdapat beberapa jenis kabel *jumper*, seperti *male to male*, *male to female*, dan *female to female*, yang masing-masing memiliki ujung konektor logam atau soket untuk koneksi yang berbeda. Konektor biasanya terbuat dari logam seperti tembaga atau baja berlapis emas untuk memastikan konduktivitas yang baik, dan dilindungi oleh plastik atau bahan insulasi lainnya untuk mencegah korsleting. Kabel *jumper* juga dilapisi oleh bahan isolasi seperti *PVC* untuk mencegah kontak listrik yang tidak diinginkan dan tersedia dalam berbagai panjang untuk memenuhi kebutuhan konfigurasi rangkaian yang berbeda." (Prastyo, 2022)

Kabel j*umper* sangat berguna dalam tahap pengembangan dan pengujian prototipe, karena memungkinkan perubahan cepat dalam rangkaian tanpa memerlukan penyolderan. Dalam *breadboarding*, kabel *jumper* digunakan untuk membuat sambungan antar komponen di *breadboard*. Kabel *jumper* adalah alat yang sangat berguna bagi pelajar dan orang yang hobi dengan elektronika untuk membuat dan memodifikasi rangkaian dengan mudah dan aman. Misalnya, saat membuat rangkaian *LED* sederhana di *breadboard*, kabel jumper *male-to-male* dapat digunakan untuk menghubungkan *pin output* dari mikro kontroler seperti *Arduino* ke resistor, dan dari resistor ke *anoda LED*, serta menghubungkan *katoda LED* ke *ground* (GND) pada breadboard. Dengan menggunakan kabel *jumper*, perubahan konfigurasi rangkaian atau penggantian komponen dapat dilakukan dengan mudah tanpa memerlukan alat khusus atau penyolderan, membuat proses pengembangan dan pengujian jauh lebih efisien dan fleksibel.

2.8.1 Fungsi Kabel *Jumper*

1. Memungkinkan melakukan pengembangan dan melakukan pengujian sirkuit elektronik tanpa menyolder komponen secara

permanen. Hal ini dapat mempermudah modifikasi dan pengujian sirkuit dengan cepat.

2. Menghubungkan berbagai macam komponen seperti *resistors, capacitors*, *ICs,* dan modul lainnya dalam sebuah prototipe sirkuit. Kabel *jumper* memfasilitasi aliran listrik di antara komponen tersebut.

3. Dapat digunakan untuk eksperimen dan pembelajaran untuk membantu pengembang memahami konsep dasar elektronik dengan membangun dan menguji sirkuit sederhana.

4. Memungkinkan menghubugkan antara mikrokontroler seperti *Arduino*, *Raspberry Pi* dan perangkat lain seperti sensor, motor, atau modul komunikasi.

2.8.2 Jenis Kabel *Jumper*

1. *Male to Male* (M-M)

Kabel *jumper Male to Male* (M-M) adalah kabel pendek dengan dua konektor jantan di kedua ujungnya yang sering digunakan dalam elektronika, terutama untuk menghubungkan komponen pada *breadboard* atau *protoboard*. Kabel ini sangat fleksibel dan mudah digunakan untuk membuat rangkaian sementara atau prototipe. Fungsinya seperti jembatan yang menghubungkan dua titik pada rangkaian.

.

Gambar 2. 13 Kabel *Jumper Male to male*
(Sumber Gambar https://nyerekatech.com/)

2. *Male to Female* (M-F):

Kabel *jumper Male to Female* adalah jenis kabel pendek yang sering digunakan dalam elektronika untuk menghubungkan komponen-komponen pada rangkaian. Berbeda dengan kabel *jumper Male to Male* yang memiliki konektor jantan pada kedua ujungnya, kabel ini memiliki satu ujung konektor jantan dan satu ujungnya lagi adalah konektor betina. Fungsi utama pada kabel *jumper Male to Female* adalah untuk menghubungkan komponen yang memiliki konektor yang berbeda. Seperti menghubungkan *output* dari suatu modul yang memiliki konektor jantan ke *input* dari modul lain yang memiliki konektor *betina*.

Gambar 2.14 Kabel *Jumper Male to Female*
(Sumber Gambar https://sg.element14.com/)

3. *Female to Female* (F-F):

Kabel *jumper Female to Female* adalah jenis kabel pendek yang memiliki kedua ujungnya berupa konektor betina. Kabel ini sering digunakan dalam elektronika, terutama untuk menghubungkan komponen-komponen yang memiliki *header* jantan atau pin. Fungsi utama kabel *jumper Female to Female* adalah untuk menghubungkan komponen-komponen yang memiliki *header* jantan di kedua sisinya. Misalnya, jika ingin menghubungkan dua buah sensor yang sama-sama memiliki *header* jantan, maka bisa menggunakan kabel *jumper Female to Female* untuk menghubungkannya.

Gambar 2. 15Kabel *Jumper Female to female*

(Sumber gambar https://jogjarobotika.com/)

Pada penelitian ini kabel *jumper female to female* dipilih karena fleksibilitas dan kemudahan yang ditawarkannya dalam menghubungkan komponen elektronik seperti *NodeMCU ESP8266* dan *relay*. Kabel *jumper female to female* memungkinkan koneksi yang cepat dan aman tanpa perlu melakukan *soldering*, yang dapat menghemat waktu dan mengurangi kesalahan dalam *penyolderan*. Selain itu, kabel jumper ini sangat cocok untuk *prototyping* karena memungkinkan perubahan dan penyesuaian rangkaian dengan mudah. Dengan menggunakan kabel *jumper female to female*, kita dapat memastikan bahwa setiap pin pada *NodeMCU ESP8266* dan *relay* telah terhubung dengan benar, sehingga mendukung kestabilan dan keandalan sistem kontrol lampu jarak jauh.

2.9 Android

Android adalah sistem operasi yang digunakan pada perangkat seluler seperti *smartphone* dan tablet. Sistem operasi ini dikembangkan oleh Google dan didasarkan pada Linux. Android memiliki kode sumber yang terbuka, yang berarti siapa saja dapat melihat, memodifikasi, dan mendistribusikannya secara bebas. Hal ini memungkinkan bagi pengembang untuk membuat aplikasi atau kustomisasi yang unik untuk perangkat Android. Android juga memiliki ekosistem aplikasi yang besar dan beragam, sehingga pengguna dapat menemukan aplikasi untuk berbagai kebutuhan.

Android adalah sistem operasi yang berbasis pada *kernel Linux*, yang dirancang untuk perangkat *mobile* seperti *smartphone* dan *tablet*. Awalnya dikembangkan oleh Android *Inc*. Perusahaan ini diakuisisi oleh Google pada tahun 2005, dan Android resmi diluncurkan pada tahun 2008. Sistem operasi ini didesain dengan antarmuka yang mudah digunakan dan mendukung berbagai aplikasi yang dapat diunduh melalui *Google Play Store*, toko aplikasi resmi untuk perangkat Android. Selain untuk perangkat *mobile*, Android kini juga digunakan pada perangkat lain, seperti televisi pintar (Android TV), jam tangan pintar (Wear OS), dan kendaraan melalui Android Auto. (Mixon, 2023)

Android dipilih untuk membuat aplikasi kontrol lampu jarak jauh karena sifatnya yang *open source* memungkinkan fleksibilitas dan kustomisasi yang luas, memungkinkan pengembang untuk mengakses dan memodifikasi sistem sesuai dengan kebutuhan spesifik. Selain itu, dukungan *multi-platform* pada Android memungkinkan aplikasi untuk berjalan di berbagai jenis perangkat, seperti *smartphone*, tablet, *smartwatch*, dan *smart TV*. Android juga memiliki komunitas pengembang yang luas dan ekosistem yang memeiliki banyak *library* dan *framework* membuat pengembangan aplikasi Android lebih mudah, dengan banyak sumber daya yang dapat diakses untuk belajar dan berbagi solusi.

Gambar 2.16 Android

(Sumber gambar https://www.liputan6.com/)

2.10 Android Studio

Android Studio merupakan Integrated Development Environment (IDE) resmi yang dikembangkan oleh Google untuk pengembangan aplikasi Android. Diperkenalkan pada tahun 2013 sebagai pengganti Eclipse Android Development Tools (ADT), Android Studio dirancang untuk memberikan lingkungan pemrograman yang efisien dan terintegrasi, membantu pengembang dalam membuat, menguji, dan merilis aplikasi Android.

Android Studio adalah IDE (Integrated Development Environment), yaitu perangkat lunak yang digunakan untuk mengembangkan aplikasi *smartphone* berbasis Android. *Software* ini pertama kali diperkenalkan pada tahun 2013 di acara *Google I/O conference* dan dirilis secara resmi ke publik setahun kemudian, pada 2014. JetBrains, perusahaan perangkat lunak asal Ceko, merupakan pengembang di balik pembuatan Android Studio. Saat ini, Android Studio sudah tersedia di berbagai platform, seperti Windows, Linux, dan Mac. (Maulana, 2022)

Android Studio mendukung integrasi *IoT* dan protokol komunikasi seperti *MQTT* atau *HTTP,* yang memungkinkan aplikasi Android untuk berkomunikasi dengan perangkat seperti NodeMCU ESP8266 atau perangkat lainnya melalui jaringan internet. Android Studio menyediakan alat pengembangan yang lengkap, seperti *editor* kode, *emulator* perangkat, dan alat *debugging*, yang memungkinkan pengembang untuk mengembangkan dan menguji aplikasi dengan efisien sebelum diimplementasikan pada perangkat nyata. Selain itu, kemampuan Android Studio untuk mendukung berbagai perangkat Android, dari *smartphone* hingga tablet, memungkinkan aplikasi kontrol lampu jarak jauh diakses oleh pengguna di berbagai perangkat. Fitur *user interface builder* dalam Android Studio juga memudahkan pengembang untuk membuat antarmuka pengguna yang intuitif dan mudah digunakan, memungkinkan kontrol yang efisien terhadap perangkat lampu dari jarak jauh. Pembaruan rutin dari Google memastikan Android Studio selalu kompatibel dengan teknologi terbaru, sehingga pengembang dapat membangun aplikasi yang aman dan *up-to-date*.

Gambar 2.17 Android Studio

(Sumber gambar https://intuji.com/)

2.11 Bahasa Java

Java adalah bahasa pemrograman tingkat tinggi yang berorientasi objek, dikembangkan oleh Sun Microsystems yang sekarang telah diakuisisi oleh Oracle pada tahun 1995. Bahasa ini dirancang untuk memiliki portabilitas, yang berarti program yang ditulis dalam bahasa java dapat dijalankan di berbagai platform tanpa perlu diubah.

Menurut (Maulana, 2022) Java adalah bahasa pemrograman yang sering digunakan untuk mengembangkan bagian *back-end* dari *software*, aplikasi Android, serta *website*. Java memiliki slogan *"Write Once, Run Anywhere*," yang berarti program yang ditulis dalam Java dapat dijalankan di berbagai *platform* tanpa harus disesuaikan ulang, seperti Android, Linux, Windows, dan lainnya. Hal ini dimungkinkan karena Java menggunakan sintaks pemrograman tingkat tinggi yang dikompilasi oleh *Java Virtual Machine* (JVM) menjadi *bytecode* yang dapat berjalan di berbagai *platform*. Berkat fleksibilitasnya, Java telah digunakan di lebih dari 13 miliar perangkat. Beberapa aplikasi *mobile* terkenal yang dibangun dengan Java antara lain Twitter, Netflix, dan Spotify.

Sebagai bahasa pemrograman berorientasi objek, Java menggunakan konsep kelas dan objek untuk membangun program, yang memungkinkan pengembang untuk mengorganisir kode secara modular dan dapat digunakan kembali. Java juga dikenal dengan *garbage collection*, yang secara otomatis

mengelola memori dan menghapus objek yang tidak lagi digunakan, membantu mencegah kebocoran memori. Java digunakan secara luas dalam pengembangan aplikasi desktop, aplikasi web, aplikasi seluler (khususnya pada platform Android), serta aplikasi *server-side*. Keunggulan Java meliputi keamanan yang kuat, kinerja yang stabil, dan dukungan ekosistem yang luas, menjadikannya salah satu bahasa pemrograman yang paling populer dan diandalkan oleh pengembang di seluruh dunia.

Gambar 2.18 Java

(Sumber gambar https://diskominfo.kedirikab.go.id/)

2.12 Java Development Kit

Java Development Kit (JDK) adalah paket perangkat lunak yang disediakan oleh Oracle dan digunakan untuk mengembangkan aplikasi berbasis Java. JDK adalah alat penting bagi pengembang Java, karena berisi semua komponen yang diperlukan untuk menulis, mengompilasi, dan menjalankan program Java. JDK mencakup sejumlah alat dan pustaka yang esensial untuk siklus hidup pengembangan aplikasi Java.

JDK, singkatan dari *Java Development Kit*, adalah lingkungan pengembangan perangkat lunak yang digunakan untuk membuat *applet* dan aplikasi Java. Pengembang Java dapat memanfaatkan JDK di berbagai *platform*, seperti Windows, macOS, Solaris, dan Linux. JDK berperan dalam membantu pengembang menulis dan menjalankan program Java. Selain itu, dimungkinkan untuk menginstal beberapa versi JDK di satu komputer, karena berbagai proyek mungkin memerlukan versi JDK yang berbeda. (Maulana, 2022)

Gambar 2. 19 Java Development Kit

(Sumber Gambar https://medium.com/)

Java Development Kit (JDK) memiliki beberapa komponen utama didalamnya. Berikut adalah komponen utama Java Development Kit (JDK):

1. Java Compiler (javac): Alat ini digunakan untuk mengubah kode sumber Java (file dengan ekstensi .java) menjadi bytecode (file dengan ekstensi .class). Bytecode ini adalah format yang dapat dijalankan oleh Java Virtual Machine (JVM) di berbagai platform.
2. Java Runtime Environment (JRE): JDK mencakup JRE, yang terdiri dari JVM dan pustaka runtime yang diperlukan untuk menjalankan program Java. Ini memungkinkan aplikasi Java yang sudah dikompilasi berjalan di berbagai sistem operasi yang mendukung JVM.
3. Java Virtual Machine (JVM): Bagian dari JRE, JVM bertugas mengeksekusi bytecode Java dan mengonversinya menjadi instruksi yang dapat dimengerti oleh perangkat keras komputer. JVM adalah komponen penting yang memungkinkan Java memiliki kemampuan portabilitas antar platform.
4. Library Standar Java: JDK menyertakan berbagai pustaka standar Java yang menyediakan API (Application Programming Interface) untuk tugas-tugas umum seperti pengelolaan file, jaringan, grafik, basis data, serta fungsi input/output (I/O).
5. Debugger (jdb): JDK juga menyertakan alat debugging untuk membantu pengembang mengidentifikasi dan memperbaiki kesalahan dalam kode program mereka.

6. JAR Tool (jar): Alat ini digunakan untuk mengelola file JAR (Java Archive), yaitu format kompresi yang digunakan untuk mengemas banyak file kelas dan metadata yang terkait menjadi satu file yang dapat didistribusikan.

2.13 Bahasa *C++*

C++ adalah bahasa pemrograman yang dikembangkan oleh Bjarne Stroustrup pada tahun 1983 sebagai perpanjangan dari bahasa *C*. *C++* mendukung pemrograman berorientasi objek, yang memungkinkan pengembangan perangkat lunak yang mudah untuk dikelola. Bahasa ini memungkinkan pemrogram untuk mengontrol *hardware* secara langsung, membuatnya ideal untuk pengembangan aplikasi yang membutuhkan kinerja tinggi, seperti sistem mikrokontroler *NodeMCU ESP8266*. *C++* memiliki *library* yang tersedia di *Arduino*, memudahkan pengembangan berbagai fungsi seperti komunikasi *Wi-Fi*, pengendalian *relay*, dan integrasi dengan layanan *web*.

C++ adalah bahasa pemrograman tingkat tinggi yang digunakan untuk mengembangkan aplikasi perangkat lunak dan sistem komputer. Sebagai turunan dari bahasa C, C++ memperkenalkan fitur-fitur tambahan seperti abstraksi data, *overloading operator*, dan *polymorphism*, yang membuatnya ideal untuk membangun aplikasi yang besar dan kompleks. Tujuan utama C++ adalah menyediakan bahasa yang kuat dan fleksibel untuk mengembangkan perangkat lunak yang efisien dan cepat. Dikenal karena kecepatannya, C++ memungkinkan akses langsung ke memori dan pengoptimalan kode, yang berkontribusi pada peningkatan kinerja dan efisiensi. Selain itu, C++ dirancang untuk mendukung portabilitas antar platform, sehingga kode yang ditulis di satu sistem operasi dapat dijalankan di sistem operasi lain tanpa perlu modifikasi. (Kinasih, 2023)

2.13.1 Fungsi *C++*

1. Pemrograman Sistem

   C++ dikenal dengan kecepatan dan efisiensi dalam penggunaan memori, membuatnya ideal untuk pengembangan sistem operasi, driver perangkat keras, dan aplikasi real-time. *C++* juga memungkinkan akses langsung ke perangkat keras, memberikan tingkat kontrol yang tinggi atas sistem komputer.

2. Pengembangan Perangkat

   C++ digunakan untuk mengembangkan perangkat lunak untuk sistem *embedded*, seperti mikrokontroler dan perangkat IoT, karena efisiensi dan kemampuan untuk mengontrol *hardware*. Serta *C++* adalah pilihan populer untuk pengembangan *game* karena kemampuannya dalam menghasilkan grafis berkualitas tinggi, menangani kompleksitas logika permainan, dan mencapai kinerja optimal.

3. Pemrograman Berorientasi Objek (OOP)

   C++ mendukung *OOP*, memungkinkan pemrograman dengan menggunakan objek, kelas, dan pewarisan, meningkatkan keterbacaan dan pemeliharaan kode. Konsep *OOP* dalam *C++* memungkinkan pembuatan kode yang dapat digunakan kembali dalam proyek yang berbeda.

4. Kinerja Tinggi

   C++ dikenal sebagai bahasa yang cepat, membuatnya cocok untuk aplikasi yang membutuhkan kinerja tinggi, seperti simulasi, pemrosesan gambar, dan aplikasi ilmiah. *C++* memberikan kontrol penuh atas manajemen memori, memungkinkan pengembang untuk mengoptimalkan penggunaan memori sesuai kebutuhan.

5. Portabilitas

   C++ dapat dijalankan pada berbagai *platform*, termasuk *Windows*, *macOS*, *Linux*, dan sistem operasi lainnya.

2.13.2 Tipe Data Pada *C++*

Tipe data dalam *C++* adalah cara untuk menentukan jenis nilai yang akan disimpan dalam sebuah variabel. Tipe data ini mempengaruhi cara penggunaan variabel dan cara yang digunakan untuk mengoperasikan nilai yang disimpan di dalamnya. Berikut adalah beberapa tipe data yang umum digunakan dalam *C++*:

1. *Bolean* (*bool*) : Digunakan untuk nilai logika yang hanya memiliki dua nilai, yaitu *true* dan *false*.
2. *Character* (*char*) : Digunakan untuk karakter tunggal, seperti huruf, angka, atau simbol.
3. *Integer* (*int*) : DIgunakan untuk bilangan bulat, seperti 1, 2, 3, 4, 5, dan seterusnya.
4. *Floating Point* (*float*) : Digunaka untuk nilai bilangan decimal seperti 3.14 atau -0.5.
5. *Double* (*double*) : Digunakan untuk nilai bilangan desimal yang lebih presisi, seperti 3.14159.
6. *Valueless* (*void*) : Digunakan untuk nilai yang tidak memiliki nilai, seperti fungsi yang tidak mengembalikan nilai.
7. *String* (*string*) : Digunakan untuk nilai teks yang dapat berupa kalimat, nama, atau kata – kata.
8. *Wide Character* (*wchar_t*) : Digunakan untuk nilai karakter yang lebih luas dari karakter biasa, seperti karakter Unicode

Dalam penelitian ini, *C++* digunakan karena kompatibilitasnya dengan *Arduino IDE*, yang menyediakan lingkungan pemrograman sederhana untuk pengembangan aplikasi pada mikrokontroler *NodeMCU ESP8266*. Dengan mennggunakan *ArduinoIDE* dan *C++,* dapat memungkinkan pemrogram untuk menulis kode dengan efisiensi memori yang baik. *C++* juga memungkinkan akses langsung ke *hardware*, memberikan kontrol presisi terhadap pin *GPIO*, *timing*, dan *interrupt* yang sangat penting untuk aplikasi *real-time* seperti kontrol lampu jarak jauh menggunakan *NodeMCU ESP8266* dan aplikasi *whatsapp.*

Gambar 2.20 *C++*

(Sumber gambar https://github.com/)

2.14 MQTT

MQTT (Message Queuing Telemetry Transport) adalah protokol komunikasi ringan yang digunakan untuk mengirimkan pesan antar perangkat dalam jaringan yang memiliki keterbatasan *bandwidth*, daya, atau kemampuan komputasi. MQTT sering digunakan dalam aplikasi *Internet of Things* (IoT) karena efisiensinya dan kemampuan untuk bekerja pada jaringan dengan latensi tinggi atau koneksi yang tidak stabil.

Message Queue Telemetry Protocol (MQTT) adalah standar protokol pesan yang dirancang untuk dapat digunakan dalam komunikasi "*machine-to-machine*", dimana protokol MQTT dapat berkomunikasi dengan *device* atau perangkat yang tidak memiliki alamat khusus. Dalam ekosistem *Internet of Things* (IoT), MQTT menjadi pilihan utama bagi perangkat sensor pintar yang perlu mengirim dan menerima data melalui jaringan dengan sumber daya dan *bandwidth* yang terbatas. MQTT dikenal sebagai protokol yang ringan karena mengirim pesan dengan *header* yang kecil, hanya sebesar 2 *bytes* dengan menggunakan konsep *publish* atau *subscribe* yang dimana dapat digunakan untuk komunikasi 2 arah yaitu dapat berkomunikasi antar *server* maupun dengan *device* lain. (Hasiholan, Primananda, & Amron, 2018)

Protokol MQTT dirancang khusus untuk digunakan oleh perangkat dengan keterbatasan *bandwidth* namun dengan *latency* yang tinggi, serta jaringan yang tidak selalu dapat diandalkan. Dengan kelebihan ini, MQTT memungkinkan

komunikasi yang efisien antara perangkat dalam jaringan IoT. Keuntungan utamanya adalah bahwa perangkat hanya menerima pesan yang relevan dengan kebutuhan mereka, yang membantu mengoptimalkan penggunaan *bandwidth* dan sumber daya yang tersedia. Berikut beberapa karakteristik utama MQTT adalah:

1. Model Publisher-Subscriber: MQTT menggunakan arsitektur publish/subscribe, di mana ada dua jenis entitas utama:
   - Publisher: Perangkat yang mengirimkan pesan ke suatu topik tertentu.
   - Subscriber: Perangkat yang berlangganan pada topik tertentu untuk menerima pesan.
   - Broker: Perantara yang bertanggung jawab mengarahkan pesan dari publisher ke subscriber yang sesuai.
2. Ringan: MQTT dirancang untuk meminimalkan penggunaan bandwidth dan sumber daya, sehingga cocok untuk perangkat IoT dengan keterbatasan daya atau komputasi, seperti sensor atau mikroprosesor.
3. Koneksi Persisten: MQTT mempertahankan koneksi antara client (perangkat) dan broker selama perangkat terhubung, memungkinkan pengiriman data secara real-time dan notifikasi ketika ada pesan baru.
4. QoS (Quality of Service): MQTT mendukung beberapa level kualitas layanan (QoS) untuk menjamin pengiriman pesan:
   - QoS 0: Pesan dikirim tanpa konfirmasi.
   - QoS 1: Pesan dikirim setidaknya sekali, sehingga bisa terjadi duplikasi.
   - QoS 2: Pesan dikirim tepat sekali, memastikan tidak ada duplikasi.
5. Aplikasi dalam IoT: MQTT digunakan dalam berbagai aplikasi IoT, termasuk sistem rumah pintar (smart home), pemantauan industri, dan komunikasi antar sensor. Protokol ini memungkinkan perangkat IoT untuk berkomunikasi secara andal, meskipun koneksi internet tidak stabil atau terbatas.

2.15 Figma

Figma adalah alat desain berbasis *web* yang digunakan untuk membuat antarmuka pengguna (UI), desain *web*, prototipe, dan memungkinkan kolaborasi tim secara *real-time*. Figma memudahkan desainer dan tim untuk bekerja bersama dalam satu proyek tanpa harus mengirim *file* berulang kali, karena semua pekerjaan tersimpan di *cloud*.

Menurut (Sundego, 2023) Figma adalah salah satu *tool* berbasis *web* yang memungkinkan desainer bekerja kapan saja dan di mana saja melalui internet. Biasanya, Figma digunakan untuk merancang antarmuka aplikasi. Dalam pengembangan aplikasi baru, Figma memungkinkan tim untuk berkolaborasi dalam proses desain. Figma dapat dijalankan di sistem operasi Windows dan macOS untuk versi desktop.

Fitur utama Figma berfokus pada desain *UI* dan *UX*, membantu merancang tampilan aplikasi dan menciptakan pengalaman pengguna yang baik saat menggunakan aplikasi. Beberapa *tools* yang memiliki kemiripan seperti Figma adalah *Sketch* dan Adobe XD dan yang membedakannya hanya pada fitur. Berikut adalah fitur – fitur utama yang ada pada figma:

1. Desain *Vector-Based*: Mirip seperti Adobe XD atau Sketch, Figma menggunakan desain berbasis vektor sehingga desain yang dihasilkan bisa diubah ukurannya tanpa kehilangan kualitas.
2. Kolaborasi *Real-Time*: Figma memungkinkan banyak orang mengerjakan desain yang sama secara bersamaan, mirip dengan Google Docs. Setiap perubahan yang dibuat dapat langsung dilihat oleh anggota tim lainnya.
3. *Prototyping*: Figma juga mendukung pembuatan prototipe interaktif, di mana pengguna dapat membuat animasi sederhana untuk menunjukkan alur pengguna (user flow).
4. Integrasi dengan *Developer*: Figma menyediakan alat bagi *developer* untuk melihat dan mengambil spesifikasi desain (misalnya, ukuran, warna, *font*) untuk membantu mereka mengimplementasikan desain ke dalam kode.

Gambar 2. 21 Figma

(Sumber gambar https://blog.myskill.id)

2.16 UML (Unified Modeling Language)

Unified Modeling Language (UML) adalah metode pemodelan visual yang berfungsi sebagai alat untuk merancang dan mengembangkan perangkat lunak berbasis objek. Sebagai bahasa visual yang khusus dirancang untuk memodelkan sistem berorientasi objek, setiap elemen dan diagram dalam UML mengikuti prinsip-prinsip *object-oriented*.

UML merupakan bahasa pemodelan standar yang secara visual menggambarkan berbagai aspek dari sebuah sistem perangkat lunak. Dengan menggunakan notasi grafis yang spesifik, UML memungkinkan kita untuk memodelkan struktur, perilaku, dan interaksi objek dalam sistem tersebut. Diagram UML yang umum digunakan, seperti diagram kelas dan diagram aktivitas, memberikan representasi visual yang komprehensif dari sistem yang sedang dirancang. (Faulina, 2023).

2.16.1 Fungsi UML (Unified Modeling Language)

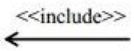
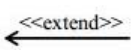
1. *Unified Modeling Language* merupakan suatu notasi standar yang digunakan untuk memvisualisasikan, merancang, dan mendokumentasikan sistem perangkat lunak. Dengan UML, berbagai aspek kompleksitas suatu sistem, mulai dari struktur internal, dinamika perilaku, hingga interaksi antar komponen, dapat direpresentasikan secara grafis. Hal ini memungkinkan para pengembang untuk memperoleh pemahaman yang lebih komprehensif dan memfasilitasi proses pengembangan perangkat lunak yang lebih efisien.

2. *Unified Modeling Language* memberikan grafis yang dapat digunakan untuk ide dan konsep yang rumit secara visual.
3. Dalam pengembangan perangkat lunak, *Unified Modeling Language* membantu para pengembang untuk mengurangi kesalahan dan meningkatkan kualitas produk akhir.
4. *Unified Modeling Language* membantu mengurangi kompleksitas dan meningkatkan pemahaman sistem secara keseluruhan, sehingga memudahkan pengembang untuk mengembangkan perangkat lunak dengan lebih cepat dan juga lebih efisien.

2.16.2 Diagram *Use Case*

Diagram *use case* menggambarkan interaksi antara seorang pengguna (user) dengan sistem yang sedang dikembangkan. Dalam konteks ini, *use case* merepresentasikan skenario atau fungsi spesifik yang dilakukan oleh pengguna saat berinteraksi dengan sistem. Pengguna, yang disebut sebagai aktor dalam diagram *use case*, memainkan peran penting sebagai entitas eksternal yang secara langsung berinteraksi dengan berbagai elemen di dalam sistem yang dirancang. Aktor ini tidak terbatas pada individu, tetapi juga bisa mencakup entitas lain seperti perangkat keras atau sistem eksternal yang berinteraksi dengan sistem

Diagram *use case* merupakan representasi visual yang digunakan untuk memodelkan interaksi antara aktor (pengguna) dengan sistem. Diagram ini secara eksplisit menggambarkan fungsionalitas sistem yang dibutuhkan oleh pengguna dalam konteks penggunaan tertentu. Dengan demikian, diagram *use case* berperan sebagai jembatan antara kebutuhan pengguna dengan implementasi sistem. (Faulina, 2023)

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 2.22 Simbol - simbol *Use Case*

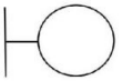
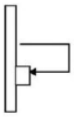
(Sumber gambar https://www.dicoding.com/)

2.16.3 Diagram Sequence

Diagram *Sequence* merupakan representasi visual yang digunakan untuk memodelkan interaksi dinamis antar objek dalam suatu sistem. Diagram ini secara eksplisit menggambarkan urutan kronologis pengiriman pesan dan pemanggilan metode antar objek, sehingga memberikan pemahaman yang mendalam mengenai alur eksekusi suatu operasi. Dengan demikian, diagram urutan menjadi alat yang efektif dalam menganalisis dan merancang interaksi antar komponen dalam sistem.

Diagram *Sequecce* merupakan representasi visual yang digunakan untuk memodelkan interaksi dinamis antar objek dalam suatu sistem. Diagram ini secara eksplisit menggambarkan urutan pengiriman dan penerimaan pesan antar objek, sehingga memberikan pemahaman yang komprehensif mengenai alur eksekusi suatu operasi. Dengan demikian,

diagram urutan menjadi alat yang efektif dalam menganalisis dan merancang interaksi antar komponen dalam sistem.. (Faulina, 2023)

Gambar	Nama	Keterangan
	Entity Class	Gambaran sistem sebagai landasan dalam menyusun basis data
	Boundary Class	Menangani komunikasi antar lingkungan sistem
	Control Class	Bertanggung jawab terhadap kelas-kelas terhadap objek yang berisi logika
	Recursive	Pesan untuk dirinya
	Activation	Mewakili proses durasi aktivasi sebuah operasi
	Life Line	Komponen yang digambarkan garis putus terhubung dengan objek

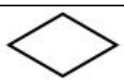
Gambar 2.23 Simbol - simbol Diagram *Sequence*

(Sumber Gambar https://blog.rumahweb.com/)

2.16.4 Diagram *Activity*

Diagram *Activity* merupakan salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang secara khusus digunakan untuk memvisualisasikan aliran kerja atau proses bisnis. Berbeda dengan diagram keadaan yang lebih menekankan pada perubahan status suatu objek dari waktu ke waktu, diagram aktivitas lebih berfokus pada urutan aktivitas yang terjadi dalam sistem. Dengan demikian, diagram aktivitas memberikan gambaran yang lebih komprehensif mengenai dinamika sistem dari perspektif aliran kerja.

Diagram *Activity* merupakan representasi visual yang digunakan untuk memodelkan dan menganalisis aliran kerja dalam suatu sistem. Diagram ini secara eksplisit menggambarkan urutan aktivitas yang terjadi dalam sistem, mulai dari inisiasi hingga terminasi proses. Dengan demikian, diagram aktivitas menjadi alat yang efektif dalam memahami, mendokumentasikan, dan mengoptimalkan proses bisnis. (Faulina, 2023)

Simbol	Nama	Keterangan
	Status awal	Sebuah diagram aktivitas memiliki sebuah status awal.
	Aktivitas	Aktivitas yang dilakukan sistem,aktivitas biasanya diawali dengan kata kerja.
	Percabangan / Decision	Percabangan dimana ada pilihan aktivitas yang lebih dari satu.
	Penggabungan / Join	Penggabungan dimana yang mana lebih dari satu aktivitas lalu digabungkan jadi satu.
	Status Akhir	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
	Swimlane	Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Gambar 2.24 Simbol - simbol Diagram Activity

(Sumber gambar https://www.dicoding.com/)

2.17 *Flowchart*

Flowchart merupakan representasi grafis yang digunakan untuk memvisualisasikan alur logika dari suatu proses atau algoritma. Dengan menggunakan simbol-simbol standar, *flowchart* menyajikan langkah-langkah yang terlibat dalam suatu proses secara berurutan dan hierarkis. Hal ini memungkinkan analisis yang lebih mendalam terhadap struktur dan efisiensi suatu proses.

Flowchart merupakan representasi grafis yang digunakan untuk memvisualisasikan alur logika dari suatu algoritma atau proses. Dengan menggunakan simbol-simbol standar, *flowchart* menyajikan langkah-langkah yang terlibat dalam suatu proses secara berurutan dan hierarkis. Alhasil, *flowchart* menjadi alat yang efektif dalam perencanaan, analisis, dan dokumentasi sistem, khususnya dalam pengembangan perangkat lunak. (Setiawan, 2021)

2.17.1 Jenis *Flowchart*

Flowchart memiliki berbagai jenis yang memiliki tujuan yang berbeda, tergantung pada kebutuhan proses atau sistem yang akan digambarkan. Berikut adalah beberapa jenis flowchart yang umum:

1. *Flowchart* dokumen

   Flowchart dokumen berfungsi sebagai representasi visual yang melacak perpindahan formulir dan laporan antar berbagai unit atau tahap dalam suatu proses. Diagram ini secara eksplisit menggambarkan bagaimana informasi didokumentasikan, diproses, dan diarsipkan.

2. *Flowchart* program

   Flowchart program terdiri dari dua macam, antara lain: *flowchart* logika program dan *flowchart* program komputer terinci

3. *Flowchart* proses

   Flowchart proses merupakan representasi visual yang digunakan untuk memodelkan dan menganalisis urutan aktivitas dalam suatu sistem. Diagram ini secara eksplisit menggambarkan langkah-langkah yang terlibat dalam suatu proses, mulai dari tahap awal hingga akhir.

4. *Flowchart* sistem

Flowchart sistem merupakan representasi visual yang digunakan untuk memodelkan dan menganalisis keseluruhan proses yang terjadi dalam suatu sistem. Diagram ini secara eksplisit menggambarkan interaksi antara berbagai komponen sistem dan urutan pelaksanaan tugas-tugas yang terlibat.

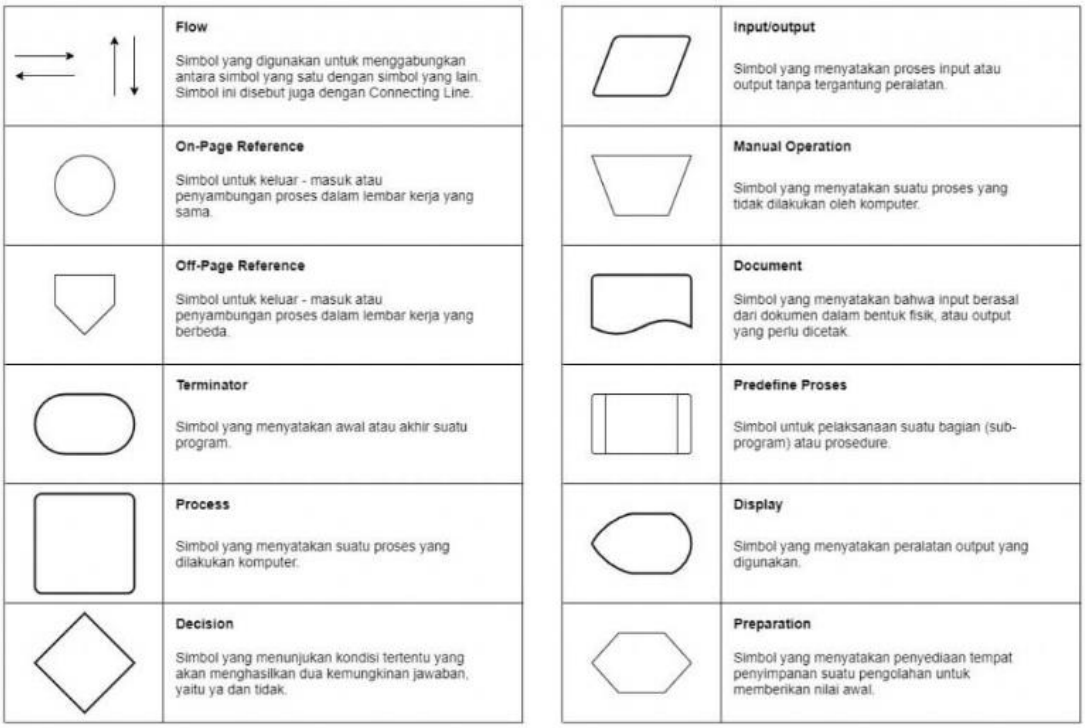
5. *Flowchart* skematik

Flowchart skematik merupakan representasi visual yang serupa dengan *flowchart* sistem, namun dengan tingkat detail yang lebih tinggi. Selain menggunakan simbol-simbol standar, *flowchart* skematik juga mengintegrasikan gambar-gambar perangkat keras dan perangkat lunak untuk memberikan gambaran yang lebih komprehensif dan mudah dipahami, terutama bagi pengguna yang tidak memiliki latar belakang teknis.

2.17.2 Fugnsi *Flowchart*

1. *Flowchart* dapat membantu dalam memvisualisasikan sebuah proses yang kompleks dengan cara yang sederhana dan mudah dipahami.
2. Membantu komunikasi antara tim atau individu dan memastikan bahwa semua pihak memahami proses yang sama.
3. Membantu dalam mengidentifikasi kelemahan dan efisiensi dalam suatu proses dan menemukan cara untuk meningkatkannya.

Dalam penelitian ini, *flowchart* sistem akan menjadi alat representasi yang paling sesuai. Hal ini dikarenakan *flowchart* sistem mampu menggambarkan secara komprehensif tahapan-tahapan dan urutan prosedur yang berjalan di dalam sistem yang sedang diteliti.



Gambar 2.25 Simbol – simbol *Flowchart*

(Sumber Gambar https://www.dicoding.com/)

BAB III
ANALISIS MASALAH DAN PERANCANGAN PROGRAM

3.1 Tempat dan Waktu Penelitian

3.1.1 Tempat Penelitian

Tempat : Warung Alfiah

Alamat : Jl. Rinjani RT 01 RW 19 Kel.Larangan Kec.Harjamukti Kota Cirebon

3.1.2 Waktu Penelitian

No.	Prosedur Penelitian	Bulan											
		Mei				Juni				Juli			
		1	2	3	4	1	2	3	4	1	2	3	4
1	Perencanaan	■	■	■	■								
2	Pengembangan				■	■	■	■					
3	Implementasi							■	■	■	■		
4	Pengujian										■	■	■

Tabel 3.1 Waktu Penelitian

3.2 Metode Pengumpulan Data

Dalam penelitian ini, pengumpulan data dilakukan melalui dua tahap. Pertama, melalui studi lapangan untuk memperoleh data primer secara langsung dari lapangan. Kedua, melalui studi pustaka untuk melengkapi data dengan referensi-referensi teoritis yang relevan.

3.2.1 Studi Lapangan atau Observasi

Melalui observasi lapangan, peneliti dapat memperoleh data empiris yang tidak dapat diperoleh melalui metode lain. Kegiatan ini memungkinkan peneliti untuk mengidentifikasi variabel-variabel laten, mengungkap dinamika sosial yang kompleks, serta memahami konteks sosial budaya yang mempengaruhi fenomena yang sedang diteliti. Sasaran dari observasi yan akan dilakukan adalah sebagai berikut :

1. Mencari tahu apakah kontrol lampu jarak jauh menggunakan nodemcu esp8266 dan aplikasi Android dibutuhkan di warung alfiah
2. Mengetahui masalah apa saja yang dialami pemilik atau penjaga warung dalam mengontrol pencahayaan warung

3.2.2 Studi Pustaka

Pengumpulan data dalam penelitian ini dilakukan melalui studi pustaka yang komprehensif. Dengan merujuk pada berbagai sumber literatur, seperti buku, jurnal ilmiah, artikel, dan sumber daring, peneliti berusaha mengumpulkan data yang relevan untuk membangun kerangka teoretis yang kuat. Sasaran dari tahapan ini adalah:

1. Penelitian ini melibatkan pengumpulan literatur yang relevan dengan sistem pengendalian lampu jarak jauh. Tujuannya adalah untuk memperoleh landasan teori yang kuat serta mengidentifikasi penelitian-penelitian terdahulu yang telah dilakukan di bidang ini.
2. Dalam penelitian ini, akan dilakukan penelusuran literatur yang berkaitan dengan sistem pengendalian lampu jarak jauh. Sumber-sumber yang akan ditelaah meliputi buku, jurnal ilmiah, artikel, dan publikasi digital lainnya yang relevan dengan topik penelitian.

3.3 Analisis Kebutuhan Perangkat Keras dan Perangkat Lunak

Proses implementasi sistem kontrol lampu jarak jauh dalam penelitian ini melibatkan penggunaan perangkat keras seperti NodeMCU ESP8266 dan perangkat lunak yang sesuai untuk pengembangan aplikasi Android. Berikut adalah perangkat keras dan perangkat lunak yang akan digunakan dalam penelitian ini:

3.3.1 Perangkat Keras atau *Hardware*

1. Laptop yang digunakan bertipe ACER NITRO 5 AN515-57 dengan spesifikasi:
   a. Processor i5 14000H
   b. RAM 16gb DDR 4

c. Memory SSD 512gb

2. NodeMCU ESP8266 lolin
3. Relay 8 channel 5v
4. Kabel Jumper female to female
5. Kabel NYM
6. Kabel Data Type C
7. Lampu
8. Smartphone

3.3.2 Perangkat Lunak atau *Software*

1. Windows 11
2. ArduinoIDE 2.3.2
3. Android Studio 2024.1.2
4. MQTTX

3.4 Analisis Perancangan Sistem Kerja

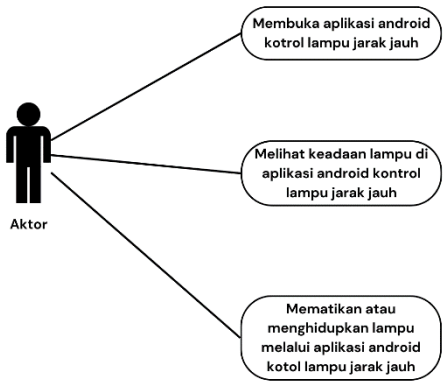
Dalam analisis perancangan sistem kerja, terdapat beberapa langkah yang perlu untuk dilakukan agar memastikan bahwa sistem yang akan dirancang dapat memenuhi kebutuhan dan berjalan dengan efektif.

3.4.1 Analisis Perancangan *UML*

Perancangan *UML* yang digunakan dalam membangun kontrol lampu jarak jauh menggunakan NodeMCU ESP8266 ini menggunakan 2 UML, yaitu *Use Case* Diagram, dan *Activity* Diagram

3.4.2 *Use Case* Diagram

Pemodelan *use case* diagram pada pengembangan aplikasi kontrol lampu jarak jauh bertujuan untuk memberikan gambaran yang jelas mengenai tujuan sistem, fitur-fitur yang akan disediakan, serta bagaimana pengguna akan berinteraksi dengan sistem. Dengan demikian, *use case* diagram menjadi alat yang penting dalam tahap perancangan sistem. Berikut adalah adalah perancangan *use case* diagram untuk kontrol lampu jarak jauh menggunakan NodeMCU ESP8266 dan aplikasi android. Terlihat seperti alur yang dibuat seperti pada gambar dibawah ini:



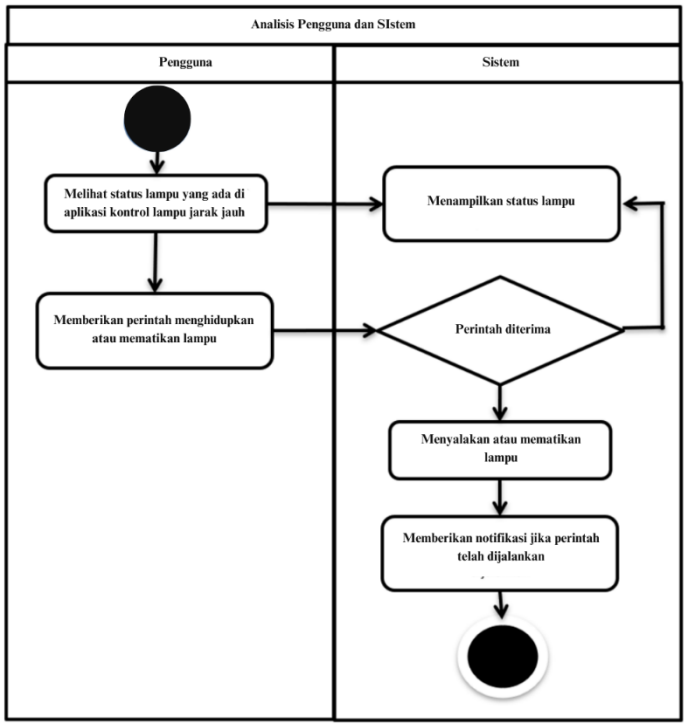
Gambar 3.1 *Use Case* Diagram

Gambar di atas menggambarkan interaksi antara seorang atau aktor dengan sebuah sistem yang dikontrol melalui WhatsApp untuk mengelola lampu. Berikut adalah penjelasan dari tiap elemen di gambar tersebut:

1. Aktor: Ini adalah pengguna atau entitas yang berinteraksi dengan sistem. Dalam konteks ini, actor adalah seseorang yang mengontrol lampu melalui aplikasi android kontrol lampu jarak jauh.
2. Membuka Aplikasi android kontrol lampu jarak jauh: Ini adalah tindakan pertama yang dilakukan oleh aktor, yaitu membuka aplikasi android kontrol lampu jarak jauh untuk memulai interaksi dengan sistem pengendalian lampu.
3. Melihat Keadaan Lampu di aplikasi android kontrol lampu jarak jauh: Setelah membuka aplikasi kontrol lampu jarak jauh, aktor bisa melihat status atau keadaan lampu, apakah lampu dalam keadaan hidup atau mati.
4. Mematikan atau Menghidupkan Lampu melalui aplikasi android kontrol lampu jarak jauh: Tindakan ini menggambarkan langkah di mana aktor mengendalikan lampu melalui aplikasi android kontrol lampu jarak jauh untuk menyalakan atau mematikan lampu. Perintah ini akan diproses oleh sistem, yang kemudian mengendalikan lampu sesuai dengan instruksi yang diberikan.

3.4.3 *Activity* Diagram

Activity Diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah *system* atau menu yang ada pada perangkat lunak. Terlihat seperti alur dari aplikasi yang dibuat seperti pada gambar dibawah ini:



Gambar 3.2 *Activity* Diagram

a) Kolom Pengguna

Kolom ini menggambarkan tindakan yang dilakukan oleh pengguna dalam proses pengendalian lampu:

1) Melihat Status Lampu:

   Pada tahap awal, pengguna melihat status lampu di aplikasi kontrol lampu jarak jauh untuk memeriksa status lampu, apakah dalam kondisi menyala atau mati.

2) Memberikan Perintah Menghidupkan atau Mematikan Lampu:

   Setelah mengetahui status lampu, pengguna kemudian memberikan perintah lebih lanjut melalui aplikasi kontrol lampu jarak jauh, yang

dapat berupa instruksi untuk menghidupkan atau mematikan lampu sesuai kebutuhan mereka. Perintah ini dikirimkan kepada sistem untuk dieksekusi.

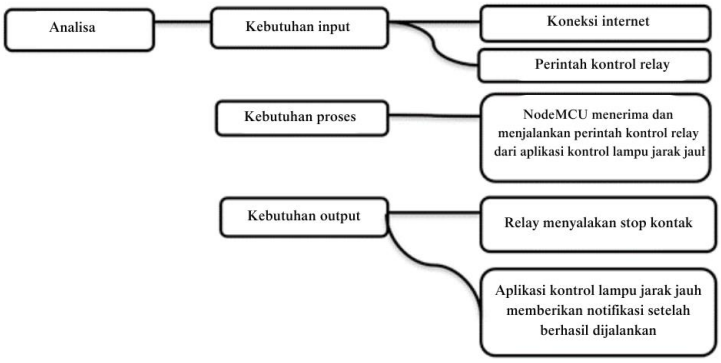
b) Kolom Sistem

Kolom ini menjelaskan respons dan operasi yang dilakukan oleh sistem setelah menerima input dari pengguna:

1) Menampilkan Status Lampu:
   Sistem memperlihatkan status lampu apakah sedang berada pada keadaan mati atau sedang hidup.
2) Perintah Diterima:
   Setelah pengguna mengirimkan perintah untuk menghidupkan atau mematikan lampu, sistem akan mengevaluasi dan memproses perintah tersebut. Diagram menunjukkan bahwa terdapat langkah keputusan di mana sistem memeriksa perintah yang diterima.
3) Menyalakan atau Mematikan Lampu:
   Berdasarkan keputusan yang dibuat pada langkah sebelumnya, sistem akan mengeksekusi perintah yang sesuai: menyalakan atau mematikan lampu.
4) Memberikan notifikasi Balasan:
   Setelah perintah pengguna dijalankan, sistem memberikan konfirmasi kepada pengguna melalui notifikasi di aplikasi kontrol lampu jarak jauh. Notifikasi ini memastikan bahwa tindakan yang diinstruksikan oleh pengguna telah berhasil dijalankan oleh sistem.

3.4.4 Analisis Gambaran Umum

Tahap ini bertujuan untuk memahami secara mendalam apa yang sebenarnya dibutuhkan oleh sistem yang akan dibangun. Kita dapat merancang sistem yang efektif, efisien, dan memenuhi ekspektasi pengguna. Salah satu pendekatan umum dalam analisis kebutuhan sistem adalah dengan membagi kebutuhan menjadi tiga bagian utama yaitu *input*, proses, dan *output*. Untuk lebih jelasnya, bisa dilihat pada gambar dibawah ini:



Gambar 3.3 Proses sistem

Gambar di atas ini menunjukkan analisis kebutuhan dalam sistem kontrol lampu jarak jauh menggunakan aplikasi android dan NodeMCU ESP8266. Gambar di atas ini membagi kebutuhan ke dalam tiga kategori utama yaitu kebutuhan input, kebutuhan proses, dan kebutuhan output.

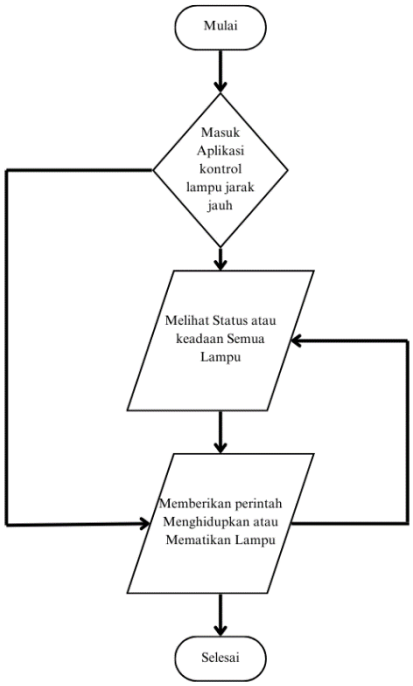
1. Kebutuhan Input:
    a. Koneksi Internet: Agar sistem dapat bekerja, diperlukan koneksi internet yang stabil. Koneksi ini penting untuk memungkinkan komunikasi antara perangkat NodeMCU dengan aplikasi kontrol lampu jarak jauh.
    b. Perintah Kontrol Relay: Perintah ini dikirim oleh pengguna melalui aplikasi kontrol lampu jarak jauh. Perintah tersebut kemudian akan diterima oleh NodeMCU untuk mengontrol relay yang terhubung.
2. Kebutuhan Proses:

    NodeMCU menerima dan menjalankan pesan perintah kontrol relay dari aplikasi kontrol lampu jarak jauh: Pada tahap ini, NodeMCU bertindak sebagai pengendali utama yang menerima perintah dari aplikasi kontrol lampu jarak jauh. Pesan perintah ini diterjemahkan menjadi sinyal kontrol untuk relay. NodeMCU memastikan bahwa perintah yang diterima sesuai dengan tujuan yang diinginkan sebelum diteruskan ke relay.

3. Kebutuhan Output:
    a. Relay menyalakan stop kontak: Setelah menerima sinyal kontrol dari NodeMCU, relay akan beraksi dengan menyalakan atau memutuskan stop kontak. Ini berarti relay berperan sebagai saklar yang dikendalikan dari jarak jauh untuk mengontrol perangkat listrik yang terhubung.
    b. Aplikasi kontrol lampu jarak jauh memberikan notifikasi setelah berhasil dijalankan: Setelah perintah berhasil dijalankan, aplikasi kontrol lampu jarak jauh akan memberikan umpan balik berupa notifikasi di aplikasi kotrol lampu jarak jauh, yang memberi tahu pengguna bahwa perintah telah dijalankan dengan sukses.

3.4.5 *Flowchart* Perancangan sistem

Flowchart perancangan sistem adalah diagram yang digunakan untuk menggambarkan alur kerja sistem atau proses sistem dalam penelitian implementasi nodemcu esp8266 pada aplikasi kontrol lampu jarak jauh berbasis android. *Flowchart* ini memberikan langkah-langkah atau urutan kegiatan yang terlibat dalam implementasi nodemcu esp8266 pada aplikasi kontrol lampu jarak jauh berbasis android, mulai dari inisialisasi sistem hingga melakukakan perintah pengendalian lampu.



Gambar 3.4 *Flowchart* Perancangan Sistem

Gambar di atas ini merupakan langkah-langkah operasional sistem kontrol lampu jarak jauh. Gambar di atas ini menggambarkan proses mulai dari inisialisasi hingga akhir pengendalian lampu, yang melibatkan pengguna dalam interaksi langsung dengan perangkat melalui aplikasi kontrol lampu jarak jauh.

1. Mulai:

   Proses dimulai ketika pengguna berinteraksi dengan sistem untuk mengontrol lampu melalui aplikasi kontrol lampu jarak jauh.

2. Masuk Aplikasi Kontrol Lampu Jarak Jauh:

   Tahap pertama dalam proses ini adalah pengguna masuk ke aplikasi kontol lampu jarak jauh.

3. Melihat Status atau keadaan Semua Lampu:

   Setelah masuk ke aplikasi kontrl lampu jarak jauh, pengguna dapat melihat status atau keadaan lampu untuk memeriksa status lampu tertentu atau semua lampu yang terhubung ke sistem.

4. Memberikan Perintah Menghidupkan atau Mematikan Lampu:

   Berdasarkan informasi status lampu yang diterima, pengguna kemudian memberikan perintah untuk menghidupkan atau mematikan lampu. Peritnah ini dikirimkan melalui aplikasi kontrol lampu jarak jauh dan diterima oleh sistem kontrol berbasis NodeMCU, yang kemudian mengeksekusi perintah tersebut.

5. Kembali ke Proses Melihat Status atau Keadaan Semua Lampu (Looping):

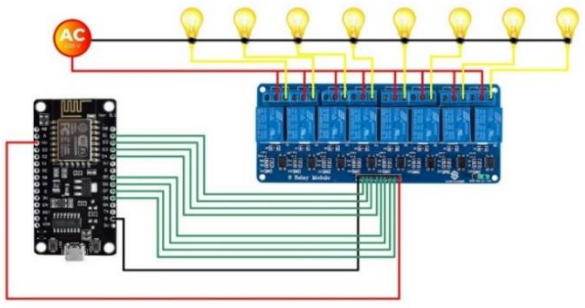
   Setelah perintah menghidupkan atau mematikan lampu dieksekusi, pengguna dapat melihat status lampu yang ada pada aplikasi kontrol lampu jarak jauh.

6. Selesai:

   Tahap ini menunjukkan bahwa pengguna telah selesai menggunakan aplikasi kontrol jarak jauh tersebut.

3.5 Percancangan Perangkat Keras

Perangkat keras yang dirancang pada penelitian ini dilengkapi dengan sistem kontrol dan *monitoring* yang terdiri dari modul NodeMCU ESP8266 dan *relay* 8 *channel*. Sistem ini memungkinkan pengendalian lampu secara *nirkabel* melalui aplikasi kontrol lampu jarak jauh. Perintah yang dikirimkan melalui aplikasi kontrol lampu jarak jauh akan diproses oleh NodeMCU ESP8266 dan diteruskan ke *relay* untuk menghidupkan lampu, mematikan lampu.atau mengetahui status lampu.



Gambar 3.5 Rangkaian Alat Kontrol Lampu

Keterangan :

a. *Port* pada *Nodemcu*

Pin yang di pakai pada NodeMcu ESP8266 untuk kontroling alat listrik 8 *port.* Meski NodeMCU bisa menggunakan 16 *port output*, ini terlihat pada pin *GPIO* 0 – 16 dengan kata lain *port output* di *NodeMCU* bisa berjumlah maksimal 16 *port*. disini penulis hanya memakai 7 *port* saja yaitu port 5, 4, 0, 2, 14, 12, 13.

Pin tegangan pada NodeMCU ada 2. Yaitu tegangan sumber yang bersumber bisa pada baterai atau pada sumber tegangan komputer yaitu ± 5V (pin VV), sedangkan ada pin lain yang di sediakan *NodeMCU* sebagai sumber tegangan yaitu tegangan dengan besaran ± 3V. Tegangan 3V biasanya di pakai untuk beberapa tegangan sensor misal (DHT11, *Ultrasonic HC-SR04*, *Infrared*, *PIR* dll)

b. Port Relay

Relay yang digunakan adalah *relay* dengan komsumsi tegangan 5V, meski di pasaran ada yang gunakan 5V, 12V dan *Relay* 220V. Disini penulis menggunakan tegangan 5V karna menghemat sambungan tegangan dari yang lain, misalnya dari adaptor 12V. Tegangan yang dipakai bersumber dari tegangan komputer atau tegangan baterai. Sehingga tegangan yang pakai oleh NodeMCU dan perangkat lainnya sama sehingga bisa mengurangi penggunaan *port* kabel.

c. Lampu

Lampu yang dipakai pada pengetesan *relay* menggunakan Lampu LED (Light Emitting Diode) dengan komsumsi tegangan kisaran 5-10V.

3.6 Perancangan Perangkat Lunak

Pada perancangan perangkat lunak ini yang perlu dilakukan, yaitu perancangan antar muka atau *user interface* pada aplikasi kontrol lampu jarak jauh yang akan digunakan oleh pengguna untuk melakukan kontrol lampu jarak jauh.

- Perancangan Halaman Utama

Gambar 3. 6 Halaman Utama

Pada gambar diatas merupakan gambar untuk tampilan halaman utama yang ada pada aplikasi kontrol lampu jarak jauh. Rancangan antar muka ini terlihat dari komponen tampilan android seperti *Text Display, Button, Card View* dan *Icon.*

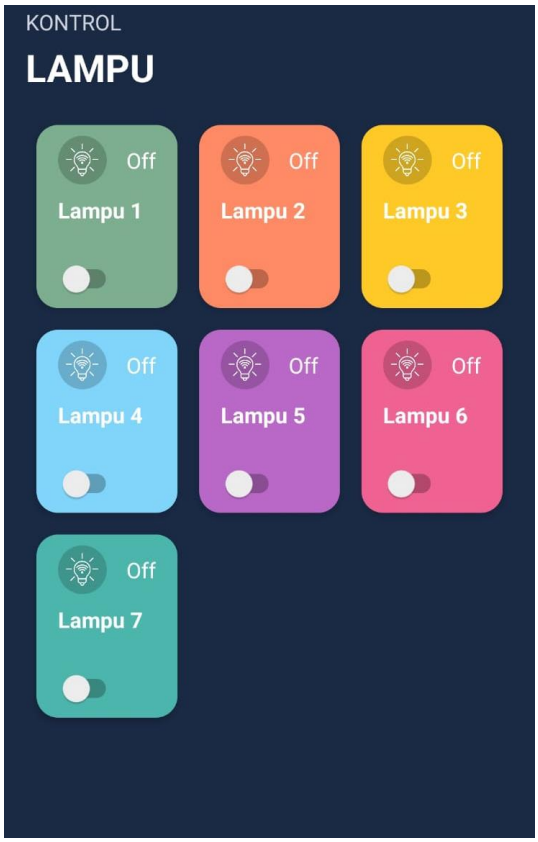
BAB IV
IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Implemetasi Sistem

Bab ini akan menyajikan hasil pengujian dan penilaian terhadap sistem yang telah dikembangkan. Tujuan utama dari pengujian adalah untuk mengukur sejauh mana sistem berhasil mencapai tujuan yang telah ditetapkan, sedangkan evaluasi bertujuan untuk menganalisis data yang diperoleh dari pengujian, serta menarik kesimpulan yang dapat mendukung pengembangan penelitian lebih lanjut.

4.1.1 Implementasi Halaman Utama

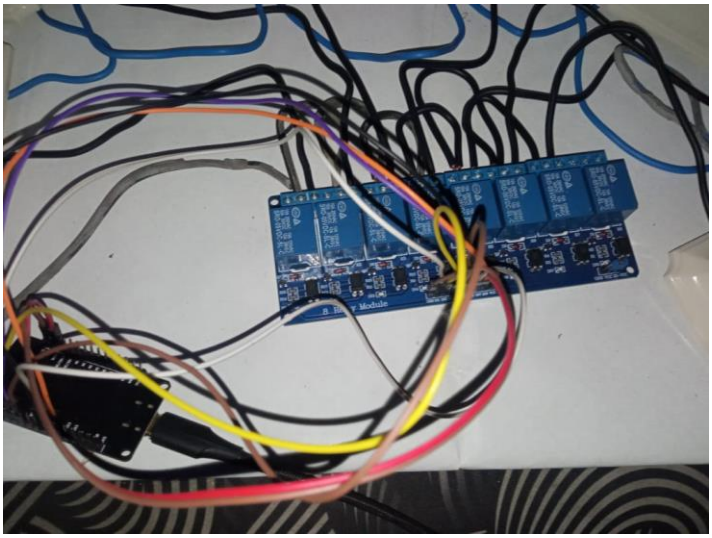
Implementasi halaman utama pada kontrol lampu jarak jauh sebagai pengontrol NodeMCU ESP8266 untuk melakukan menghidupkan atau mematikan lampu. Berikut adalah gambar tampilan halaman utama pada aplikasi kontrol lampu jarak jauh



Gambar 4.1 Implementasi Pada Halaman Utama

4.1.2 Implementasi Alat

Implementasi alat kontrol lampu jarak jauh menggunakan NodeMCU ESP8266 untuk menerima perintah pengontrolan lampu dari aplikasi kontrol lampu jarak jauh dan mengirimkan perintah ke *relay* sebagai pengontrol arus listrik.



Gambar 4.2 NodeMCU ESP8266 dan Relay

4.2 Pengujian Alat

Pengujian alat meliputi beberapa perintah yang bisa dilakakukan dan pengujian terhadap *device* atau *smartphone* lainnya.

4.2.1 Pengujian perintah

Pada pengujian perintah, ada 2 hal yang bisa dilakukan pada aplikasi kontrol lampu jarak jauh. Yaitu mematikan lampu dan menghidupkan lampu. Berikut adalah pengujian perintah pada aplikasi kontrol lampu jarak jauh:

1) Pengujian perintah pada lampu 1
   a) Menghidupkan lampu

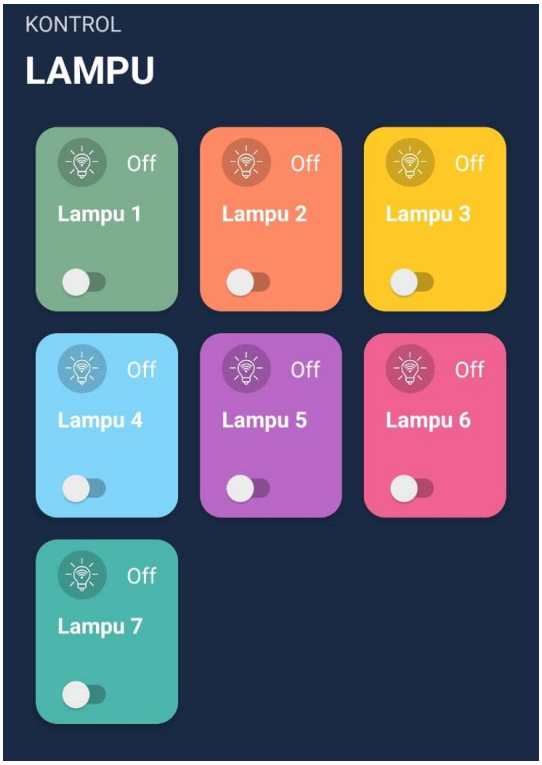
      Untuk menghidupkan lampu 1, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 1 di aplikasi kontrol lampu jarak jauh.

Gambar 4.3 Menghidupkan lampu 1 di aplikasi kontrol lampu jarak jauh

Gambar 4.4 Lampu 1 dihidupkan

b) Mematikan lampu

Untuk mematikan lampu 1, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 1 di aplikasi kontrol lampu jarak jauh.



Gambar 4.5 Mematikan lampu 1 di aplikasi kontrol lampu jarak jauh

Gambar 4.6 Lampu 1 dimatikan

2) Pengujian perintah pada lampu 2

a) Hidupkan lampu

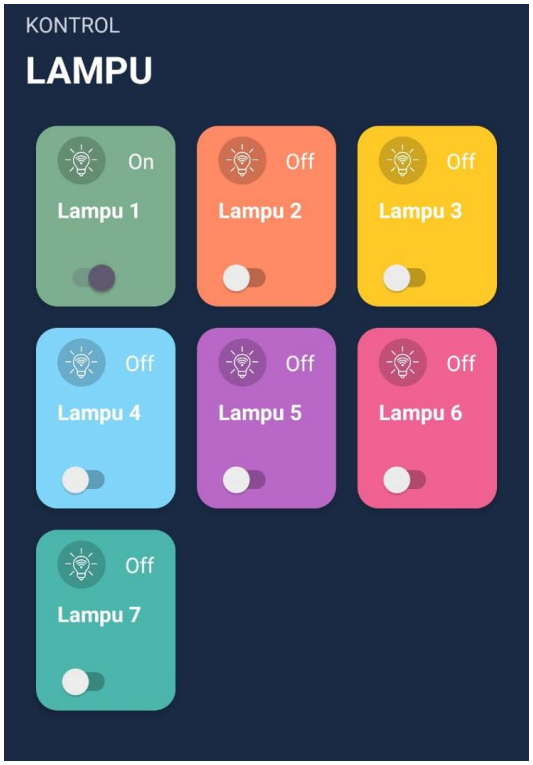
Untuk menghidupkan lampu 2, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pad lampu 2 di aplikasi kontrol lampu jarak jauh

Gambar 4. 7 Menghidupkan Lampu 2 di aplikasi kontrol lampu jarak jauh

Gambar 4.8 Lampu 2 dihidupkan

b) Matikan lampu

Untuk mematikan lampu 2, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 2 di aplikasi kontrol lampu jarak jauh



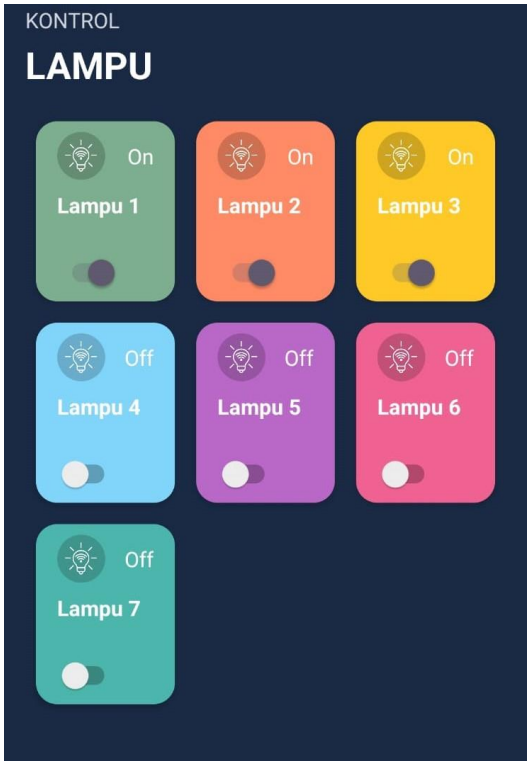
Gambar 4.9 Mematikan Lampu 2 di aplikasi kontrol lampu jarak jauh

Gambar 4.10 Lampu 2 dimatikan

3) Pengujian perintah pada lampu 3

a) Hidupkan lampu

Untuk menghidupkan lampu 3, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 3 di aplikasi kontrol lampu jarak jauh

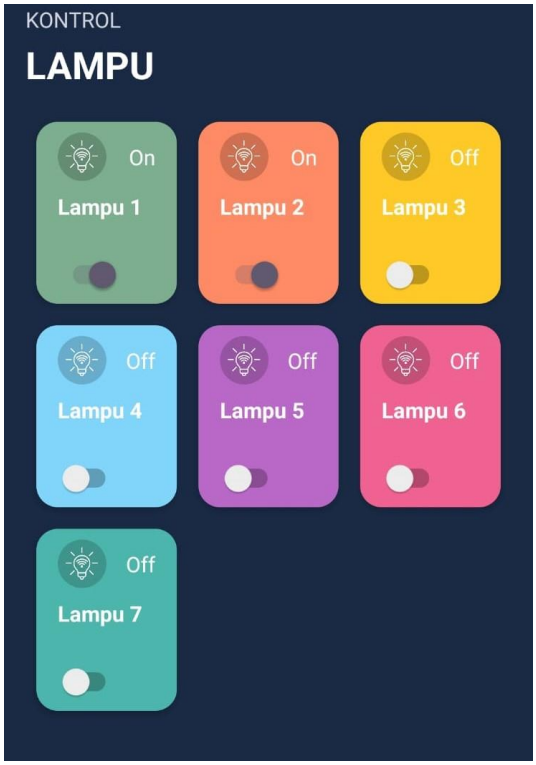


Gambar 4.11 Menghidupkan Lampu 3 di aplikasi kontrol lampu jarak jauh

Gambar 4.12 Lampu 3 dihidupkan

b) Matikan lampu

Untuk mematikan lampu 3, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 3 di aplikasi kontrol lampu jarak jauh



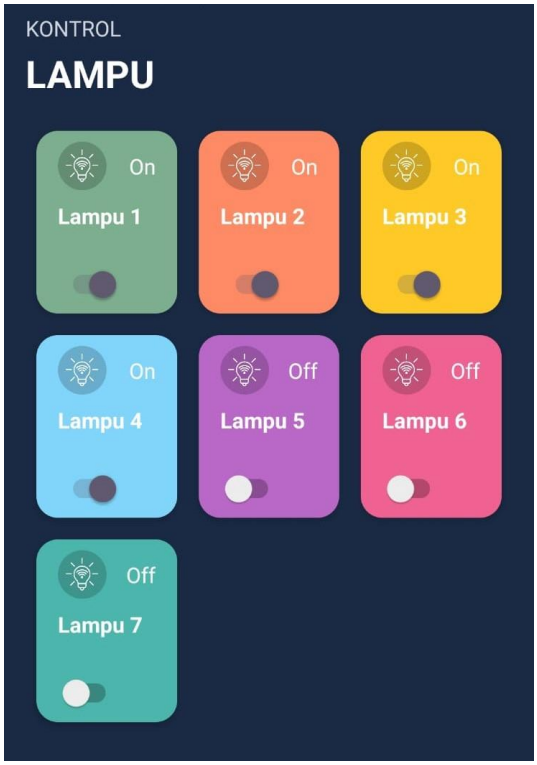
Gambar 4.13 Mematikan Lampu 3 di aplikasi kontrol lampu jarak jauh

Gambar 4.14 Lampu 3 dimatikan

4) Pengujian perintah pada lampu 4

a) Hidupkan lampu

Untuk menghidupkan lampu 4, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 4 di aplikasi kontrol lampu jarak jauh

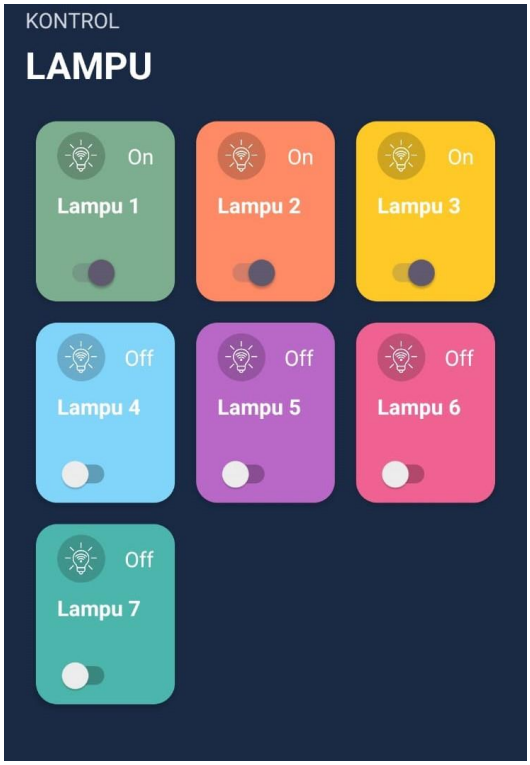


Gambar 4.15 Menghidupkan Lampu 4 di aplikasi kontrol lampu jarak jauh

Gambar 4.16 Lampu 4 dihidupkan

b) Matikan lampu

Untuk mematikan lampu 4, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 4 pada aplikasi kontrol lampu jarak jauh



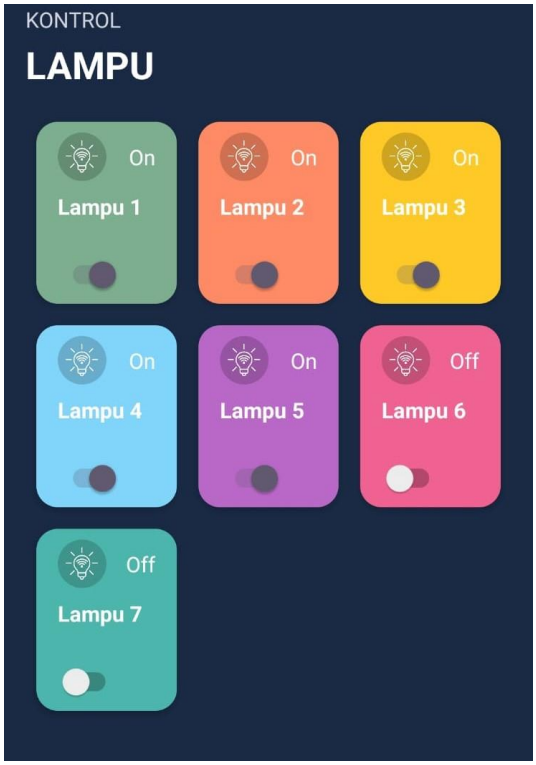
Gambar 4.17 Mematikan Lampu 4 di aplikasi kontrol lampu jarak jauh

Gambar 4.18 Lampu 4 dimatikan

5) Pengujian perintah pada lampu 5

a) Hidupkan lampu

Untuk menghidupkan lampu 5, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 5 pada aplikasi kontrol lampu jarak jauh

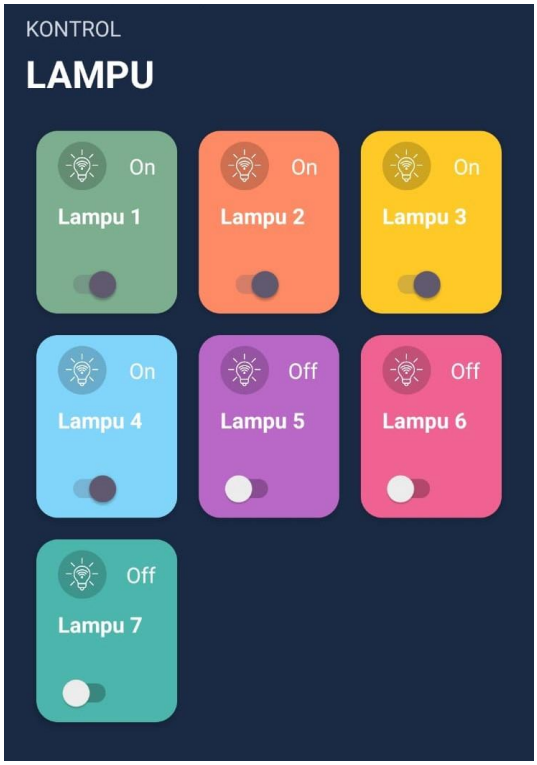


Gambar 4. 19 Menghidupkan lampu 5 di aplikasi kontrol lampu jarak jauh

Gambar 4.20 Lampu 5 dinyalakan

b) Matikan lampu

Untuk mematikan lampu 5, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 5 pada aplikasi kontrol lampu jarak jauh



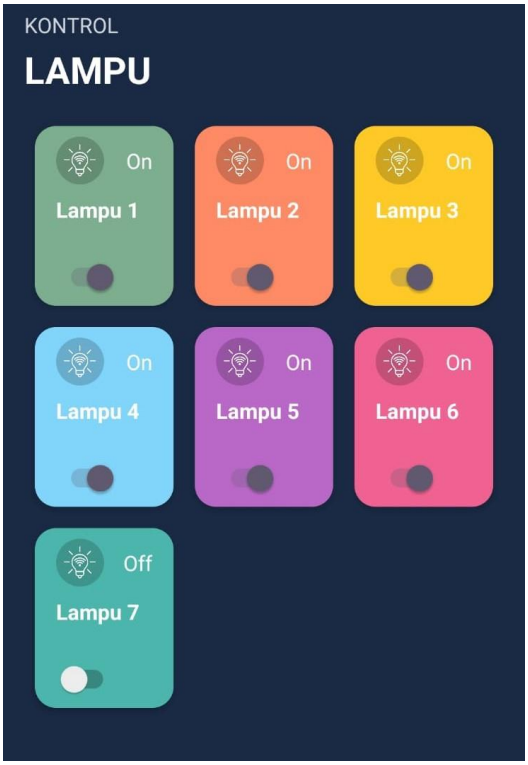
Gambar 4.21 Mematikan lampu 5 di aplikasi kontrol lampu jarak jauh

Gambar 4.22 Lampu 5 dimatikan

6) Pengujian perintah pada lampu 6

a) Hidupkan lampu

Untuk menghidupkan lampu 6, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 6 pada aplikasi kontrol lampu jarak jauh

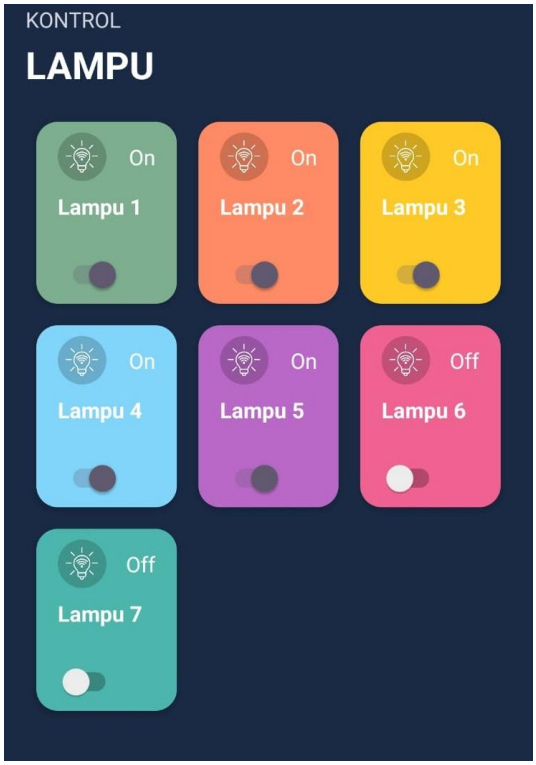


Gambar 4.23 Menghidupkann lampu 6 di aplikasi kotrol lampu jarak jauh

Gambar 4.24 Lampu 6 dihidupkan

b) Matikan lampu

Untuk mematikann lampu 6, yang perlu dilakukan adalah menekan tombol *switch* agar berubah menjadi *off* pada lampu 6 pada aplikasi kontrol lampu jarak jauh

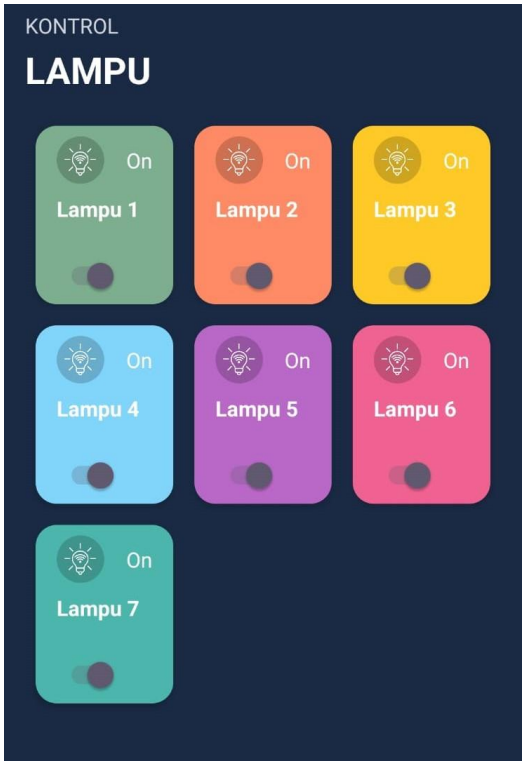


Gambar 4.25 Mematikan lampu 6 di aplikasi kotrol lampu jarak jauh

7) Pengujian perintah pada lampu 7

a) Hidupkan lampu

Untuk menghidupkan lampu 7, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *on* pada lampu 7 pada aplikasi kontrol lampu jarak jauh

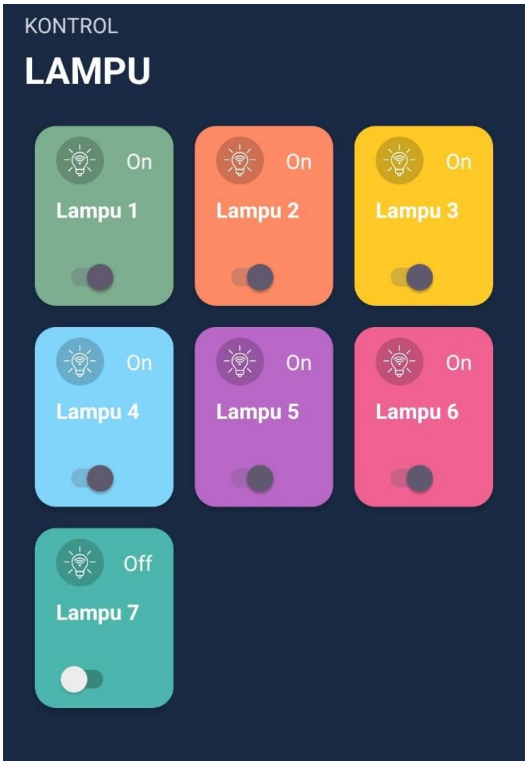


Gambar 4. 26 Menghidupkan lampu 7 di aplikasi kontrol lampu jarak jauh

Gambar 4.27 Lampu 7 dihidupkan

b) Matikan lampu

Untuk mematikan lampu 7, yang perlu dilakukan adalah dengan menekan tombol *switch* agar berubah menjadi *off* pada lampu 7 pada aplikasi kontrol lampu jarak jauh



Gambar 4.28 Mematikan lampu 7 di aplikasi kotrol lampu jarak jauh

Gambar 4.29 Lampu 7 dimatikan

4.3 Hasil Uji Coba

Hasil dari uji coba sistem kontrol lampu jarak jauh menggunakan *NodeMCU ESP8266* dan *Whatsapp* bisa dilihat pada tabel berikut ini:

No	Perintah Yang Dilakukan	Hasil
1	Menghidupkan Lampu 1	Berhasil
2	Mematikan Lampu 1	Berhasil
3	Status Lampu 1	Berhasil
4	Mennghidupkan Lampu 2	Berhasil
5	Mematikan Lampu 2	Berhasil
6	Status Lampu 2	Berhasil
7	Menghidupkan Lampu 3	Berhasil
8	Mematika Lampu 3	Berhasil
9	Status Lampu 3	Berhasil
10	Menghidupkan Lampu 4	Berhasil
11	Mematikan Lampu 4	Berhasil
12	Status Lampu 4	Berhasil
13	Menghidupkan Lampu 5	Berhasil
14	Mematikan Lampu 5	Berhasil
15	Status Lampu 5	Berhasil
16	Meghidupkan Lampu 6	Berhasil
17	Mematikan Lampu 6	Gagal
18	Status Lampu 6	Berhasil
19	Menghidupkan Lampu 7	Berhasil
20	Mematikan Lampu 7	Berhasil
21	Status Lampu 7	Berhasil

Tabel 4.1 Tabel Hasil Uji Coba

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil dari pengujian sistem yang telah dilaksanakan, maka dapat diambil kesimpulan sebagai berikut :

1. NodeMCU ESP8266 terbukti sebagai salah satu alternatif yang efektif untuk mengendalikan lampu rumah secara jarak jauh. Modul ini menunjukkan potensi besar dalam aplikasi Internet of Things (IoT) untuk rumah pintar.
2. Kinerja sistem kontrol lampu sangat bergantung pada konektivitas internet baik pada perangkat smartphone maupun NodeMCU ESP8266. Kestabilan koneksi internet secara langsung memengaruhi kecepatan dan keandalan sistem.
3. NodeMCU ESP8266 mampu memproses data yang diterima dari aplikasi kontrol lampu jarak jauh dengan baik. *Modul* ini berhasil menerjemahkan perintah dari aplikasi kontrol lampu jarak jauh menjadi sinyal kontrol yang kemudian diteruskan ke relay untuk menghidupkan dan mamatikan lampu sesuai dengan yang diprogram.

5.2 Saran

Berdasarkan hasil implementasi sistem kontrol lampu jarak jauh ini, beberapa saran pengembangan dapat diajukan untuk meningkatkan kinerja dan fungsionalitas sistem, penulis memberi beberapa saran diantaranya:

1. Untuk meningkatkan kapabilitas sistem, disarankan untuk memperluas jumlah output pada NodeMCU. Hal ini memungkinkan sistem untuk mengontrol berbagai perangkat elektronik lainnya, seperti kipas angin, AC, dan televisi
2. Penambahan fitur pemantauan visual melalui CCTV akan memberikan kenyamanan bagi pengguna dalam mengoperasikan sistem. Pengguna dapat dengan mudah memantau status sistem dari jarak jauh dan melakukan tindakan korektif jika diperlukan.

3. Penggunaan aplikasi selain membuat aplikasi untuk kotrol lampu jarak jauh, agar mengetahui apakah dengan menggunakan aplikasi yang sudah ada untuk melakukan kontrol lampu jauh bisa dilakukan, seperti aplikasi *line*, *kakaotalk*, *massanger*, *instagram*, dan lainnya.

DAFTAR PUSTAKA

Abdaoe, F., Hendi Setiawan, M., & Perdana.S.T., K. (2020). Sistem Kendali Lampu Otomatis Berbasis Iot (Internet Of Things) Menggunakan Nodencum.

Anggraini, D. (2022, September 6). *Kenali Jenis Kabel Listrik*. Retrieved from monotaro.id: https://www.monotaro.id

Arfiansyah, Z., Adriyanto, T., & Ristyawan, A. (2023). Perancangan dan Implementasi Sistem Kendali.

Chandra, M. Y. (2019). Implementasi Internet Of Things Pada Sistem Kendali Lampu Rumah Menggunakan Telegram Messenger Bot Dan NodeMcu Esp8266. *19*(1).

Dia, H. (2023, Desember 26). *Apa itu lampu*. Retrieved from rayzeek: https://www.rayzeek.com/

Faulina, A. R. (2023, Maret 23). *Pengertian, Fungsi, dan Contoh UML*. Retrieved from Sekawanmedia: https://www.sekawanmedia.co.id/

Goodwin, G. E. (2023, September 6). *What is WhatsApp? A guide to navigating the free Meta-owned communication platform*. Retrieved from businessinsider: https://www.businessinsider.com

Hakim, B. d. (2018). Sistem Monitoring Penggunaan Air PDAM Pada Rumah Tangga Menggunakan Mikrokontroler NODEMCU Berbasis Smartphone ANDROID. 9-12.

Hasiholan, Primananda, & Amron. (2018). Implementasi Konsep Internet of Things pada Sistem Monitoring Banjir menggunakan Protokol MQTT. *Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 2.

Ibrahim, A. M. (2021). Prototype Pengendalian Lampu dan AC Jarak Jauh dengan Jaringan Internet menggunakan Aplikasi Telegram berbasis NODEMCU ESP8266. *Journal of Technology Information*, 27-34.

Ibrahim, A. M., & Setiyadi, D. (2021). PROTOTYPE PENGENDALIAN LAMPU DAN AC JARAK JAUH DENGAN JARINGAN INTERNET MENGGUNAKAN APLIKASI TELEGRAM BERBASIS NODEMCU ESP8266.

Indra Gunawan, T. A. (2020). Prototipe Penerapan Internet Of Things (Iot) Pada Monitoring Level Air Tandon Menggunakan NodeMCU ESP8266 Dan Blynk. *Jurnal Informatika dan Teknologi*, 1-7.

Kinasih, T. (2023, Maret 2). *Mengenal Apa Itu Bahasa Pemrograman C++ dan Penggunaannya*. Retrieved from Kuncie: https://www.kuncie.com/

Maheri, A. S., & Supriyono, H. (2019). Pengontrol Lampu Jarak Jauh Berbasis Web.

Maulana, M. (2022, Oktober 19). *Android Studio Adalah: Pengertian, Fitur, dan Cara Install*. Retrieved from itbox: https://itbox.id/blog/android-studio-adalah/

McKinsey. (2024, Mei 28). *What is the Internet of Things (IoT)?* Retrieved from McKinsey & Company: https://www.mckinsey.com/

Mixon, E. (2023, Januari). *Adroid OS*. Retrieved from techtarget: https://www.techtarget.com/searchmobilecomputing/definition/Android-OS

Mohamad Rizky Ramadhandy Budianto, T. R. (2021). Perspektif Islam Terhadap Ilmu Pengetahuan dan Teknologi. *Jurnal Islamika*, 55-61.

Nasution, A. H., Indriani, S., Fadhilah, N., Arifin, C., & Tamba, S. P. (2019). PENGONTROLAN LAMPU JARAK JAUH DENGAN NODEMCU MENGGUNAKAN BLYNK.

Prastyo, E. A. (2022, 11 21). *Pengertian, Jenis dan Cara Kerja Kabel Jumper Arduino*. Retrieved from arduinoindonesia: https://www.arduinoindonesia.id/2022/11/pengertian-jenis-dan-cara-kerja-kabel-jumper-arduino.html

Pratama, R. (2018). APLIKASI WEBSERVER ESP8266 UNTUK PENGENDALI. *INVOTEK J. Inov. Vokasional dan Teknol*, 39-44.

Ruuhwan, R. R. (2019). Sistem Kendali dan Monitoring Pada Rumah Pintar Berbasis Internet of Things (IoT). *Innovation in Research of Informatics (INNOVATICS)*.

Setiawan, R. (2021, Agustus 4). *Flowchart Adalah: Fungsi, Jenis, Simbol, dan Contohnya*. Retrieved from Dicoding: https://www.dicoding.com/blog/flowchart-adalah/

SiddheshNan. (2024, Februari 8). *ThingESP*. Retrieved from Arduino: https://www.arduino.cc/

Sood, K. (2021, Agustus 12). *What is Twilio and How Does it Work?* Retrieved from tekkiwebsolutions: https://www.tekkiwebsolutions.com/

Sundego, J. (2023, Juni 16). *Figma Adalah: Fitur, Kegunaan, dan Manfaatnya*. Retrieved from Purwadhika: https://purwadhika.com/blog/figma-adalah-fitur-kegunaan-dan-manfaatnya

Suryana, T. (2021). Implementasi Komunikasi Web Server Nodemcu Esp8266 Dan Web Server Apache Mysql Untuk Otomatisai Dan Kontrol Peralatan Elektronik Jarak Jauh Via Internet. *Jurnal Komputa Unikom*.

LAMPIRAN – LAMPIRAN

Source Code Program

Source code pada arduino IDE

```
#include <LittleFS.h>
#include <ArduinoJson.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

// Pin yang terhubung ke relay
const int relayPins[] = {D0,
D1, D2, D5, D6, D7, D8};  //
Menggunakan 7 pin relay
const int numRelays = 7;  //
Jumlah relay yang digunakan

bool relayStatus[7] = {}; //
Inisialisasi untuk 7 relay,
default semua status relay =
false

// Inisialisasi WiFi
const char *ssid =
"JEJENHERAWAN";
const char *password =
"jh082019";

// Konfigurasi MQTT Broker
#define MQTT_USER
""
#define MQTT_PASSWORD
""
const char* mqtt_server =
"test.mosquitto.org";

// Variabel global const char*
const char* publishTopic =
"data/publish/v1";
const char* subcribeTopic =
"data/response/v1";

WiFiClient espClient;
PubSubClient
client(espClient);

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

  // Set semua pin relay
sebagai output dan dalam
kondisi OFF (aktif rendah)
  for (int i = 0; i <
numRelays; i++) {
    pinMode(relayPins[i],
OUTPUT);
    digitalWrite(relayPins[i],
HIGH);  // Set semua relay
menjadi OFF (aktif rendah)
  }

  // Inisialisasi LittleFS
untuk menyimpan data status
relay
  if (!LittleFS.begin()) {
    Serial.println("LittleFS
mount failed");
    return;
  }

  initWiFi();
  initMQTT();

  // Load status relay dari
file SPIFFS
  loadStateFromSPIFFS();
}

void loop() {
  if (!client.connected()) {
    reconnectMQTT();
  }
  client.loop();
}

void setRelay(int relayId,
bool stateRelay) {
  // Logika aktif rendah, ON
ketika LOW, OFF ketika HIGH

digitalWrite(relayPins[relayId
], stateRelay ? LOW : HIGH);
  relayStatus[relayId] =
stateRelay; // Simpan status
ke array

  // Simpan status relay ke
SPIFFS
  saveStateToSPIFFS();

  // Tentukan status relay ON
atau OFF
  String relayStatus_ =
stateRelay ? "ON" : "OFF";
  sendMQTTResponse("addrelay",
true, "Relay No " +
String(relayId) + ": " +
relayStatus_ + " Successfully
:)", JsonObject());
}
```

```
void sendListStatusRelay() {
  DynamicJsonDocument
jsonDoc(1024);
  JsonArray jsonArray =
jsonDoc.to<JsonArray>();

  // Loop untuk menambahkan
relayId dan status ke dalam
JSON array
  for (int i = 0; i <
numRelays; i++) {
    JsonObject jsonObj =
jsonArray.createNestedObject()
;  // Membuat objek relay baru
    jsonObj["relayId"] = i;
// Menambahkan relayId
    jsonObj["status"] =
relayStatus[i];  //
Menambahkan status
  }

  // Kirim JSON array ke MQTT
  sendMQTTResponse("getlist",
true, "Relay data retrieved
successfully", jsonArray);
}

void saveStateToSPIFFS() {
  String stateRelay =
getAllRelayState();
  writeFile("/relaystate.txt",
stateRelay);
}

String getAllRelayState() {
  StaticJsonDocument<512>
jsonDoc;
  JsonArray relays =
jsonDoc.to<JsonArray>();

  // Loop untuk menambahkan
relayId dan status ke dalam
JSON
  for (int i = 0; i <
numRelays; i++) {
    JsonObject relay =
relays.createNestedObject();
    relay["relayId"] = i;
    relay["status"] =
relayStatus[i];
  }

  String jsonString;
  serializeJson(jsonDoc,
jsonString);
  Serial.print("Debug
getAllRelayState: ");
  Serial.println(jsonString);

  return jsonString;
}

void loadStateFromSPIFFS() {
  String jsonRelayState =
readFile("/relaystate.txt");
// Load status dari SPIFFS
  if (jsonRelayState != "") {

updateRelayState(jsonRelayStat
e);
  }
}

void updateRelayState(String
jsonRelayState) {
  StaticJsonDocument<512> doc;
  DeserializationError error =
deserializeJson(doc,
jsonRelayState);

  if (error) {

Serial.print("Deserialization
failed: ");

Serial.println(error.c_str());
    return;
  }

  JsonArray relays =
doc.as<JsonArray>();

  // Iterasi setiap relay
dalam array JSON
  for (JsonObject relay :
relays) {
    int relayId =
relay["relayId"];
    bool status =
relay["status"];
    relayStatus[relayId] =
status;

    // Menampilkan status di
serial monitor
    Serial.print("Relay ID:
");
    Serial.print(relayId);
    Serial.print(" Status: ");
    Serial.println(status ?
"ON" : "OFF");
```

```
    // Mengatur status relay
(aktif rendah)

digitalWrite(relayPins[relayId
], status ? LOW : HIGH);
  }
}

void writeFile(const char
*path, const String &message)
{
  File file =
LittleFS.open(path, "w");
  if (!file) {
    Serial.println("Failed to
open file for writing");
    return;
  }
  if (file.print(message)) {
    Serial.println("File
written");
  } else {
    Serial.println("Write
failed");
  }
  file.close();
}

String readFile(const char
*path) {
  File file =
LittleFS.open(path, "r");
  if (!file) {
    Serial.println("Failed to
open file for reading");
    return "";
  }

  String fileContent = "";
  while (file.available()) {
    char character =
file.read();
    fileContent += character;
  }
  file.close();
  return fileContent;
}

void initWiFi() {
  WiFi.begin(ssid, password);
  while (WiFi.status() !=
WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting
to WiFi...");
  }
  Serial.println("Connected to
WiFi");
}

void initMQTT() {

client.setServer(mqtt_server,
1883);

client.setCallback(callback);
  client.setBufferSize(2048);
// Buffer untuk pesan besar

  while (!client.connected())
{
    reconnectMQTT();
  }
}

void reconnectMQTT() {
  while (!client.connected())
{
    Serial.print("Attempting
MQTT connection...");
    String clientId =
"nodeMcu-" +
String(random(0xffff), HEX);
    if
(client.connect(clientId.c_str
(), MQTT_USER, MQTT_PASSWORD))
{

Serial.println("connected");

client.subscribe(subcribeTopic
);
    } else {
      Serial.print("failed,
rc=");

Serial.print(client.state());
      Serial.println(" try
again in 5 seconds");
      delay(5000);
    }
  }
}

void callback(char* topic,
byte* payload, unsigned int
length) {
  String message;
  for (unsigned int i = 0; i <
length; i++) {
```

```
    message +=
(char)payload[i];
  }

  StaticJsonDocument<256>
jsonDoc;
  DeserializationError error =
deserializeJson(jsonDoc,
message);

  if (error) {

Serial.print("deserializeJson(
) failed: ");

Serial.println(error.c_str());
    return;
  }

  String operationType =
jsonDoc.containsKey("operation
") ?
jsonDoc["operation"].as<String
>() : "";
  int switchNumber =
jsonDoc.containsKey("switchNum
ber") ?
jsonDoc["switchNumber"].as<int
>() : -1;
  bool statusRelay =
jsonDoc.containsKey("status")
? jsonDoc["status"].as<bool>()
: false;

  if (operationType ==
"getlist") {
    sendListStatusRelay();
  } else if (operationType ==
"addrelay") {
    if (switchNumber >= 0 &&
switchNumber < numRelays) {
      setRelay(switchNumber,
statusRelay);
    }
  }
}

void sendMQTTResponse(String
operation, bool success,
String message, JsonVariant
data) {
  StaticJsonDocument<1024>
jsonDoc;
  jsonDoc["operation"] =
operation;
  jsonDoc["success"] =
success;
  jsonDoc["message"] =
message;

  if (!data.isNull()) {
    jsonDoc["data"] = data;
  }

  String response;
  serializeJson(jsonDoc,
response);

  if
(client.publish(publishTopic,
response.c_str())) {
    Serial.println("Sent
response to MQTT:");
    Serial.println(response);
  } else {
    Serial.println("Failed to
send response to MQTT");
  }
}
```

Source Code pada Android Studio

MainActivity.java

```
package
com.myproject.smartrelay;

import
android.app.ProgressDialog;
import android.graphics.Color;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import
android.widget.ProgressBar;
import android.widget.Toast;

import
androidx.activity.EdgeToEdge;
import
androidx.appcompat.app.AppComp
atActivity;
import
androidx.core.graphics.Insets;
import
androidx.core.view.ViewCompat;
import
androidx.core.view.WindowInset
sCompat;
```

```
import
androidx.recyclerview.widget.G
ridLayoutManager;
import
androidx.recyclerview.widget.R
ecyclerView;

import
com.emreesen.sntoast.SnToast;
import
com.emreesen.sntoast.Type;

import java.util.ArrayList;
import java.util.List;

public class MainActivity
extends AppCompatActivity
implements
GridAdapter.OnItemSwitchListen
er,
MQTTClient.MqttCallbackInterfa
ce {

    private MQTTClient
mqttClient;

    private
LoadingDialogFragment
loadingDialog;

    private RecyclerView
recyclerView;

    @Override
    protected void
onCreate(Bundle
savedInstanceState) {

super.onCreate(savedInstanceSt
ate);

EdgeToEdge.enable(this);

setContentView(R.layout.activi
ty_main_rev);

        recyclerView =
findViewById(R.id.recyclerView
);

recyclerView.setLayoutManager(
new GridLayoutManager(this,
3));

        mqttClient = new
MQTTClient(this, this);
        mqttClient.connect();
        isLoading(true);

    }

     @Override
    public void
onItemSwitch(int position,
boolean isChecked) {
     String status = isChecked
? "ON" : "OFF";

mqttClient.publishMessage(Topi
cManager.MAIN_PUBLISH,
position, isChecked);

    }
    @Override
    public void
onMessageReceived(String
message) {
        runOnUiThread(() -> {
         RelayStatusResponse
relayStatusResponse =
mqttClient.processPayload(mess
age);
            if
(relayStatusResponse.getOperat
ion().equals("addrelay")) {

//loadingDialog.dismiss();
                new
SnToast.Builder()

.context(MainActivity.this)

.type(Type.SUCCESS)

.message(relayStatusResponse.g
etMessage())

.duration(2000) //Optional
Default: 3000ms

.build();
            } else if
(relayStatusResponse.getOperat
ion().equals("getlist")) {

getListRelay(message);
            }
        });
    }
     private void
getListRelay(String message) {
        // 7 unique colors
```

```
        String[] colors = {
                "#7DAE8F",  //
Greenish
                "#FF8A65",  //
Orange
                "#FFCA28",  //
Yellow
                "#81D4FA",  //
Light Blue
                "#BA68C8",  //
Purple
                "#F06292",  //
Pink
                "#4DB6AC"   //
Teal
        };

        List<RelayData>
relayData =
mqttClient.processListPayload(
message);
        if (relayData != null
|| !relayData.isEmpty()) {
            List<Item>
itemList = new ArrayList<>();
            itemList.add(new
Item("Lampu 1",
Color.parseColor(colors[0]),
relayData.get(0).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 2",
Color.parseColor(colors[1]),
relayData.get(1).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 3",
Color.parseColor(colors[2]),
relayData.get(2).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 4",
Color.parseColor(colors[3]),
relayData.get(3).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 5",
Color.parseColor(colors[4]),
relayData.get(4).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 6",
Color.parseColor(colors[5]),
relayData.get(5).getStatus(),
R.drawable.smartlight));
            itemList.add(new
Item("Lampu 7",
Color.parseColor(colors[6]),
relayData.get(6).getStatus(),
R.drawable.smartlight));

             GridAdapter
adapter = new
GridAdapter(this, itemList,
this);

recyclerView.setAdapter(adapte
r);
            isLoading(false);
        }
    }
      @Override
    public void
onConnectionStatusChanged(bool
ean isConnected) {
        if (isConnected) {

mqttClient.requestList(TopicMa
nager.MAIN_PUBLISH);
        }
    }

    private void
showLoadingDialog(String
message) {
        // Membuat instance
dari LoadingDialogFragment
        loadingDialog = new
LoadingDialogFragment();

        // Tampilkan dialog
loading terlebih dahulu

loadingDialog.show(getSupportF
ragmentManager(),
"loading_dialog");
        runWithDelay(500,
message);
    }
     public void
runWithDelay(long delayMillis,
String message) {
        new
Handler().postDelayed(new
Runnable() {
            @Override
            public void run()
{
                // Update
pesan loading setelah delay
```

```
                if
(loadingDialog != null &&
loadingDialog.isVisible()) {

loadingDialog.setLoadingMessag
e(message);
                }
            }
        }, delayMillis);
    }

    public void
isLoading(boolean isLoading){
        ProgressBar
progressBar =
findViewById(R.id.progressBar)
;
        // Mengatur
visibilitas berdasarkan nilai
isLoading
        if (isLoading) {

progressBar.setVisibility(View
.VISIBLE);  // Menampilkan
ProgressBar
        } else {

progressBar.setVisibility(View
.GONE);     // Menyembunyikan
ProgressBar
        }
    }
}
```

GridAdapter.java

```
package
com.myproject.smartrelay;

import
android.content.Context;
import
android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import
android.widget.ImageView;
import android.widget.Switch;
import
android.widget.TextView;

import
androidx.annotation.NonNull;
import
androidx.cardview.widget.CardV
iew;
import
androidx.recyclerview.widget.R
ecyclerView;

import java.util.List;

public class GridAdapter
extends
RecyclerView.Adapter<GridAdapt
er.ViewHolder> {

    private List<Item>
itemList;
    private Context context;
    private
OnItemSwitchListener
onItemSwitchListener;

    // Constructor to pass the
data
    public GridAdapter(Context
context, List<Item> itemList,
OnItemSwitchListener listener)
{
        this.context =
context;
        this.itemList =
itemList;

this.onItemSwitchListener =
listener;
    }


    @NonNull
    @Override
    public ViewHolder
onCreateViewHolder(@NonNull
ViewGroup parent, int
viewType) {
        View view =
LayoutInflater.from(context).i
nflate(R.layout.itemview,
parent, false);
        return new
ViewHolder(view);
    }

    @Override
    public void
onBindViewHolder(@NonNull
ViewHolder holder, int
position) {
        // Get current item
        Item currentItem =
itemList.get(position);
```

```
        // Set title
holder.itemTitle.setText(curre
ntItem.getTitle());

        // Set card background
color
holder.cardView.setCardBackgro
undColor(currentItem.getColor(
));

        // Set switch state if
needed
holder.itemSwitch.setChecked(c
urrentItem.isSwitchOn());

        // Set initial text
status
holder.tvItemState.setText(cur
rentItem.isSwitchOn() ? "On" :
"Off");

        // Set icon if needed
holder.itemIcon.setImageResour
ce(currentItem.getIconResId())
;

        // Handle switch
toggle
holder.itemSwitch.setOnChecked
ChangeListener((buttonView,
isChecked) -> {
            // Update the
status of the item
currentItem.setSwitchOn(isChec
ked);

            // Update the text
status based on the switch
state
holder.tvItemState.setText(isC
hecked ? "On" : "Off");

            // Notify the
activity via the listener
            if
(onItemSwitchListener != null)
{
onItemSwitchListener.onItemSwi
tch(position, isChecked);
            }
        });
    }

    @Override
    public int getItemCount()
{
        return
itemList.size();
    }

    public static class
ViewHolder extends
RecyclerView.ViewHolder {

        public TextView
itemTitle, tvItemState;
        public Switch
itemSwitch;
        public ImageView
itemIcon;
        public CardView
cardView;

        public
ViewHolder(@NonNull View
itemView) {
            super(itemView);
            itemTitle =
itemView.findViewById(R.id.ite
m_title);
            itemSwitch =
itemView.findViewById(R.id.ite
m_switch);
            itemIcon =
itemView.findViewById(R.id.ite
m_icon);
            cardView =
itemView.findViewById(R.id.car
d_view);
            tvItemState =
itemView.findViewById(R.id.tvI
temState);
        }
    }

    // Define an interface for
the switch click listener
    public interface
OnItemSwitchListener {
        void onItemSwitch(int
position, boolean isChecked);
    }
```

```
}
```

Itemjava

```
package
com.myproject.smartrelay;

public class Item {

    private String title;
    private int color;
    private boolean switchOn;
    private int iconResId;

    public Item(String title,
int color, boolean switchOn,
int iconResId) {
        this.title = title;
        this.color = color;
        this.switchOn =
switchOn;
        this.iconResId =
iconResId;
    }

    public String getTitle() {
        return title;
    }

    public int getColor() {
        return color;
    }

    // Mengubah nama metode
untuk lebih mencerminkan
tujuannya
    public boolean
isSwitchOn() {
        return switchOn;
    }

    public int getIconResId()
{
        return iconResId;
    }

    // Metode setter untuk
memperbarui status switch
    public void
setSwitchOn(boolean isChecked)
{
        this.switchOn =
isChecked;
    }

}
```

LoadingDialogFragment.java

```
package
com.myproject.smartrelay;

import android.graphics.Color;
import
android.graphics.drawable.Colo
rDrawable;
import android.os.Bundle;
import
android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import
android.widget.TextView;

import
androidx.annotation.NonNull;
import
androidx.annotation.Nullable;
import
androidx.fragment.app.DialogFr
agment;

public class
LoadingDialogFragment extends
DialogFragment {

    public String
loadingMessage = "harap
tunggu...";  // Default
message
    private TextView
loadingText;

    @Nullable
    @Override
    public View
onCreateView(@NonNull
LayoutInflater inflater,
@Nullable ViewGroup container,
@Nullable Bundle
savedInstanceState) {
        // Inflate the custom
layout
        View view =
inflater.inflate(R.layout.load
ing_dialog, container, false);

        // Set the initial
loading message
        loadingText =
view.findViewById(R.id.loading
_text);
```

```
        if (loadingText !=
null) {

loadingText.setText(loadingMes
sage);
        }

        return view;
    }

    // Method to set the
loading message
    public void
setLoadingMessage(String
message) {
        // If the view is
already created, update the
text view
        if (getView() != null)
{
            TextView
loadingText =
getView().findViewById(R.id.lo
ading_text);
            if (loadingText !=
null) {

loadingText.setText(message);
            }
        }
    }

    @Override
    public void onStart() {
        super.onStart();
        // Set the dialog to
be non-cancelable

getDialog().setCancelable(fals
e);
        // Optionally: Set
dialog size if necessary
        if (getDialog() !=
null) {

getDialog().getWindow().setLay
out(ViewGroup.LayoutParams.WRA
P_CONTENT,
ViewGroup.LayoutParams.WRAP_CO
NTENT);

getDialog().getWindow().setBac
kgroundDrawable(new
ColorDrawable(Color.TRANSPAREN
T));
        }
    }
}
```

MQTTClient.java

```
package
com.myproject.smartrelay;

import
android.content.Context;
import android.os.Handler;
import android.util.Log;

import com.google.gson.Gson;
import
com.somsakelect.android.mqtt.M
qttAndroidClient;

import
org.eclipse.paho.client.mqttv3
.IMqttActionListener;
import
org.eclipse.paho.client.mqttv3
.IMqttDeliveryToken;
import
org.eclipse.paho.client.mqttv3
.IMqttToken;
import
org.eclipse.paho.client.mqttv3
.MqttCallback;
import
org.eclipse.paho.client.mqttv3
.MqttClient;
import
org.eclipse.paho.client.mqttv3
.MqttConnectOptions;
import
org.eclipse.paho.client.mqttv3
.MqttException;
import
org.eclipse.paho.client.mqttv3
.MqttMessage;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class MQTTClient {

    private static final
String TAG = "MQTTClient";
    private MqttAndroidClient
mqttAndroidClient;
    private final String
serverUri =
"tcp://test.mosquitto.org:1883
```

```
"; // Ganti dengan broker MQTT kamu
    private final String clientId = MqttClient.generateClientId();

    private MqttCallbackInterface callbackInterface;
    private Gson gson;

    // Retry interval untuk reconnect (misalnya, 5 detik)
    private static final long RECONNECT_INTERVAL_MS = 5000;
    private Handler handler = new Handler(); // Handler untuk mengatur jeda reconnect

    public MQTTClient(Context context, MqttCallbackInterface callbackInterface) {
        this.callbackInterface = callbackInterface;
        this.gson = new Gson();
        this.mqttAndroidClient = new MqttAndroidClient(context, serverUri, clientId);

initializeMqttCallback();
    }

    // Initialize MQTT callback (Handle messages, connection lost, and delivery complete)
    private void initializeMqttCallback() {

mqttAndroidClient.setCallback(new MqttCallback() {
            @Override
            public void connectionLost(Throwable cause) {
                Log.d(TAG, "Connection lost: " + cause.toString());

callbackInterface.onConnectionStatusChanged(false);

scheduleReconnect();
            }

            @Override
            public void messageArrived(String topic, MqttMessage message) {
                Log.d(TAG, "Message arrived: " + new String(message.getPayload()));

handleIncomingMessage(topic, message);
            }

            @Override
            public void deliveryComplete(IMqttDeliveryToken token) {
                Log.d(TAG, "Delivery complete: " + token.toString());
            }
        });
    }

    // Handle incoming messages
    private void handleIncomingMessage(String topic, MqttMessage message) {
        String messagePayload = new String(message.getPayload());
        Log.d(TAG, "Received message payload: " + messagePayload);

        // Menentukan aksi berdasarkan topik yang diterima
        if (topic.equals(TopicManager.MAIN_SUBSCRIBE)) {

callbackInterface.onMessageReceived(messagePayload);
        } else {
            Log.d(TAG, "Received message from unknown topic: " + topic);
        }
    }

    public void connect() {
        try {
```

```
            MqttConnectOptions
options = new
MqttConnectOptions();

            // Atur keep-alive
interval untuk memastikan
koneksi tetap hidup

options.setKeepAliveInterval(1
0); // Keep-alive interval
dalam detik


mqttAndroidClient.connect(opti
ons, new IMqttActionListener()
{
                @Override
                public void
onSuccess(IMqttToken
asyncActionToken) {
                    Log.d(TAG,
"Connected");

subscribeToTopic(TopicManager.
MAIN_SUBSCRIBE);

callbackInterface.onConnection
StatusChanged(true);
                }

                @Override
                public void
onFailure(IMqttToken
asyncActionToken, Throwable
exception) {
                    Log.d(TAG,
"Failed to connect: " +
exception.toString());

//callbackInterface.onConnecti
onStatusChanged(false);

scheduleReconnect(); //
Jadwalkan reconnect jika
koneksi hilang
                }
            });
        } catch (MqttException
e) {

e.printStackTrace();
        }
    }
```

```
    private void
subscribeToTopic(String topic)
{
        try {

mqttAndroidClient.subscribe(to
pic, 0, null, new
IMqttActionListener() {
                @Override
                public void
onSuccess(IMqttToken
asyncActionToken) {
                    Log.d(TAG,
"Subscribed to topic: " +
topic);
                }

                @Override
                public void
onFailure(IMqttToken
asyncActionToken, Throwable
exception) {
                    Log.d(TAG,
"Failed to subscribe: " +
exception.toString());
                }
            });
        } catch (MqttException
e) {

e.printStackTrace();
        }
    }

    public void disconnect() {
        try {

mqttAndroidClient.disconnect()
;
        } catch (MqttException
e) {

e.printStackTrace();
        }
    }

    // Method untuk reconnect
jika koneksi hilang
    private void reconnect() {
        if (mqttAndroidClient
!= null &&
!mqttAndroidClient.isConnected
()) {
            Log.d(TAG,
"Attempting to reconnect...");
            connect();
```

```
        }
    }

    // Menjadwalkan reconnect
dengan jeda tertentu
    private void
scheduleReconnect() {

handler.postDelayed(new
Runnable() {
            @Override
            public void run()
{
                reconnect();
// Coba reconnect setelah jeda
waktu
            }
        },
RECONNECT_INTERVAL_MS);
    }


    // Create (Publish) data
ke MQTT
    public void
publishMessage(String topic,
int switchNumber, boolean
isOn) {
        try {
            // Membuat objek
untuk data payload
            Map<String,
Object> payload = new
HashMap<>();

payload.put("operation",
"addrelay");   // Menambahkan
operasi "addrelay"

payload.put("switchNumber",
switchNumber);

payload.put("status", isOn);
// Status dikirim sebagai
boolean (true/false)

            // Mengonversi
objek payload ke JSON
menggunakan Gson
            Gson gson = new
Gson();
            String
messagePayload =
gson.toJson(payload);

            // Membuat
MqttMessage dengan payload
JSON
            MqttMessage
message = new MqttMessage();

message.setPayload(messagePayl
oad.getBytes());
            message.setQos(1);
// Set QoS (Quality of
Service)

message.setRetained(false); //
Pesan tidak disimpan sebagai
retained

            // Periksa koneksi
sebelum publikasi
            if
(mqttAndroidClient != null &&
mqttAndroidClient.isConnected(
)) {

mqttAndroidClient.publish(topi
c, message);
                Log.d(TAG,
"Message published: " +
messagePayload);
            } else {
                Log.e(TAG,
"Client not connected.
Reconnecting...");
                reconnect();
            }
        } catch (MqttException
e) {
            // Penanganan
error yang lebih baik dengan
logging yang lebih terstruktur
            Log.e(TAG, "Error
Publishing Message: " +
e.getMessage(), e);
        } catch (Exception e)
{
            Log.e(TAG,
"Unexpected error: " +
e.getMessage(), e);
        }
    }

    public void
requestList(String topic) {
        try {
            // Membuat objek
untuk data payload
```

```
            Map<String,
Object> payload = new
HashMap<>();

payload.put("operation",
"getlist");   // Menambahkan
operasi "addrelay"

            // Mengonversi
objek payload ke JSON
menggunakan Gson
            Gson gson = new
Gson();
            String
messagePayload =
gson.toJson(payload);

            // Membuat
MqttMessage dengan payload
JSON
            MqttMessage
message = new MqttMessage();

message.setPayload(messagePayl
oad.getBytes());
            message.setQos(1);
// Set QoS (Quality of
Service)

message.setRetained(false); //
Pesan tidak disimpan sebagai
retained

            // Periksa koneksi
sebelum publikasi
            if
(mqttAndroidClient != null &&
mqttAndroidClient.isConnected(
)) {

mqttAndroidClient.publish(topi
c, message);
                Log.d(TAG,
"Message published: " +
messagePayload);
            } else {
                Log.e(TAG,
"Client not connected.
Reconnecting...");
                reconnect();
            }
        } catch (MqttException
e) {
            // Penanganan
error yang lebih baik dengan
logging yang lebih terstruktur
            Log.e(TAG, "Error
Publishing Message: " +
e.getMessage(), e);
        } catch (Exception e)
{
            Log.e(TAG,
"Unexpected error: " +
e.getMessage(), e);
        }
    }


    // Method untuk mengolah
payload dari MQTT
    public RelayStatusResponse
processPayload(String
jsonPayload) {
        // Membuat instance
Gson
        Gson gson = new
Gson();

        // Mengonversi JSON
string menjadi objek
PayloadModel
        RelayStatusResponse
relayStatusResponse =
gson.fromJson(jsonPayload,
RelayStatusResponse.class);

        return
relayStatusResponse;
    }

    // Method untuk mengolah
payload dari MQTT
    public List<RelayData>
processListPayload(String
jsonPayload) {
        // Inisialisasi list
relay sebagai list kosong di
awal
        List<RelayData>
relayDataList = new
ArrayList<>();

        // Membuat instance
Gson
        Gson gson = new
Gson();

        // Mengonversi JSON
string menjadi objek
RelayStatusResponse
        RelayStatusResponse
responseModel =
```

```
gson.fromJson(jsonPayload,
RelayStatusResponse.class);

        if (responseModel !=
null &&
responseModel.getData() !=
null) {
            // Ambil
List<RelayData> dari
responseModel
            relayDataList =
responseModel.getData();

            // Iterasi atau
akses objek RelayData dalam
List
//            for (RelayData
relayData : relayDataList) {
//
System.out.println("Relay ID:
" + relayData.getRelayId() +
", Status: " +
relayData.getStatus());
//            }
        }

        return relayDataList;
    }


    public interface
MqttCallbackInterface {
        void
onMessageReceived(String
message);
        void
onConnectionStatusChanged(bool
ean isConnected);
    }


}
```

TopicManager.Java

```
package
com.myproject.smartrelay;

public class TopicManager {
    // Daftar topik MQTT yang
digunakan
    public static final String
MAIN_SUBSCRIBE =
"data/publish/v1";
    public static final String
MAIN_PUBLISH =
"data/response/v1";
    //public static final
String ID_DEVICE = "12345";

    // Jika ada topik lain,
tambahkan di sini
}
```

RelayData.java

```
package
com.myproject.smartrelay;

public class RelayData {
    private int relayId;
    private boolean status;

    // Constructor tanpa
argumen
    public RelayData() {
    }

    // Constructor penuh
    public RelayData(int
relayId, boolean status) {
        this.relayId =
relayId;
        this.status = status;
    }

    // Getters dan Setters
    public int getRelayId() {
        return relayId;
    }

    public void setRelayId(int
relayId) {
        this.relayId =
relayId;
    }

    public boolean getStatus()
{
        return status;
    }

    public void
setStatus(boolean status) {
        this.status = status;
    }
}
```

RelayStatusRespon.java

```
package
com.myproject.smartrelay;

import java.util.List;

public class
RelayStatusResponse {
```

```
    private String operation;
    private boolean success;
    private String message;
    private List<RelayData>
data;

    public String
getOperation() {
        return operation;
    }

    public void
setOperation(String operation)
{
        this.operation =
operation;
    }

    public boolean isSuccess()
{
        return success;
    }

    public void
setSuccess(boolean success) {
        this.success =
success;
    }

    public String getMessage()
{
        return message;
    }

    public void
setMessage(String message) {
        this.message =
message;
    }

    public List<RelayData>
getData() {
        return data;
    }

    public void
setData(List<RelayData> data)
{
        this.data = data;
    }
}
```