RANCANG BANGUN APLIKASI ANDROID UNTUK MENGONTROL LAMPU MENGGUNAKAN *MIKROKONTROLER NODEMCU*

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Menyelesaikan Studi
Pada Program Studi D IV Teknik Informatika

Oleh :

Nama : Evan Fauzi Pranendra

NIM : 17090041

PROGRAM STUDI SARJANA TERAPAN TEKNIK INFORMATIKA

POLITEKNIK HARAPAN BERSAMA

TEGAL

2021

HALAMAN PERNYATAAN

Yang bertanda tangan dibawah ini :

Nama : Evan Fauzi Pranendra

NIM : 17090041

Adalah mahasiswa Program Studi D IV Teknik Informatika Politeknik Harapan Bersama, dengan ini saya menyatakan bahwa laporan Tugas Akhir yang berjudul :

"RANCANG BANGUN APLIKASI ANDROID UNTUK MENGONTROL LAMPU MENGGUNAKAN *MIKROKONTROLER NODEMCU*"

Merupakan hasil pemikiran sendiri secara orisinil dan saya susun secara mandiri dengan tidak melanggar kode etik hak karya cipta. Pada laporan Tugas Akhir ini juga bukan merupakan karya yang pernah diajukan untuk memperoleh gelar akademik tertentu di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila dikemudian hari ternyata Laporan Tugas Akhir ini terbukti melanggar kode etik karya cipta atau merupakan karya yang dikategorikan mengandung unsur plagiarisme, maka saya bersedia untuk melakukan penelitian baru dan menyusun laporannya sebagai laporan Tugas Akhir, sesuai dengan ketentuan yang berlaku.

Demikian pernyataan ini saya buat dengan sebenarnya dan sesungguhnya.

19 Desember 2021

Pembuat pernyataan,

Evan Fauzi Pranendra

NIM. 17090041

HALAMAN REKOMENDASI

Pembimbing Tugas Akhir memberikan rekomendasi kepada :

Nama	: Evan Fauzi Pranendra
NIM	: 17090041
Program Studi	: D IV Teknik Informatika
Judul Tugas Akhir	: "Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU* (Studi Kasus: SDN Kraton 3 Tegal)".

Mahasiswa tersebut telah dinyatakan selesai melaksanakan bimbingan dan dapat mengikuti Ujian Tugas Akhir pada tahun akademik 2021.

Tegal, Juli 2021

Pembimbing I,

Dega Surono Wibowo, S.T., M.Kom.
NIPY. 06.014.183

Pembimbing II,

Rosid Mustofa, M.Kom.

HALAMAN PENGESAHAN

Nama	: Evan Fauzi Pranendra
NIM	: 17090041
Program Studi	: D IV Teknik Informatika
Judul Tugas Akhir	: Rancang Bangun Aplikasi Android Untuk Mengontrol Lampu Menggunakan *Mikrokontroler Nodemcu*

Dinyatakan LULUS/~~TIDAK LULUS~~ setelah dipertahankan di hadapan Dewan Penguji Tugas Akhir Program Studi D IV Teknik Informatika Politeknik Harapan Bersama.

Tegal, 14 Desember 2021

Dewan Penguji :

Nama			Tandatangan
1.	Ketua	: Slamet Wiyono, S.Pd., M.Eng	1.
2.	Anggota I	: M. Nishom, M.Kom	2.
3.	Anggota II	: Rosid Mustofa, M.Kom	3.

Mengetahui,
Ketua Program Studi D IV Teknik Informatika

Slamet Wiyono, S.Pd., M.Eng
NIPY. 08.015.222

ABSTRAK

Lampu merupakan sumber penerangan pengganti sinar matahari. Pada umumnya penerangan lampu masih menggunakan sistem *on/off* pada saklar. Produk saklar yang sering digunakan masih berupa tombol yang harus kita tekan untuk mengoperasikannya. Pada malam hari perlu penerangan untuk SDN Kraton 3 Tegal agar tidak terjadi hal yang tidak diinginkan misalnya pencurian. Selain itu, untuk memudahkan karyawan dalam menyalakan lampu lewat aplikasi android. Sistem kendali lampu dengan menggunakan *Mikrokontroler NodeMCU* yang terhubung dengan internet dapat dikendalikan melalui aplikasi android di smartphone yang juga terhubung dengan internet dan menggunakan *database Firebase* yang dapat diimplementasikan dengan *mikrokontroler,* android, dan *website*. Menggunakan sensor cahaya/*LDR* yang dapat mendeteksi lampu sudah menyala atau belum, sensor arus *ACS712* dan sensor tegangan *ZMPT101B* yang dmanfaatkan untuk menghitung nilai arus, tegangan dan daya yang dikeluarkan. Dan juga menghitung biaya listrik yang dikeluarkan dari daya yang telah didapat dari sensor arus dan tegangan.

Kata Kunci: Lampu, *Internet of Things*, *Mikrokontroler, NodeMCU, Firebase,* Sensor Cahaya/*LDR*, Sensor *ACS712,* Sensor *ZMPT101B*

KATA PENGANTAR

Dengan memanjatkan puji syukur kehadiran Allah SWT, Tuhan Yang Maha Pengasih dan Maha Penyayang yang telah melimpahkan segala rahmat, hidayah dan inayah-Nya hingga terselesaikannya laporan Tugas Akhir dengan judul "RANCANG BANGUN APLIKASI ANDROID UNTUK MENGONTROL LAMPU MENGGUNAKAN *MIKROKONTROLER NODEMCU*".

Tugas Akhir merupakan suatu kewajiban yang harus dilaksanakan untuk memenuhi salah satu syarat kelulusan dalam mencapai derajat Sarjana Sain Terapan pada program Studi D IV Teknik Informatika Politeknik Harapan Bersama. Selama melaksanakan penelitian dan kemudian tersusun dalam laporan Tugas Akhir ini, banyak pihak yang telah memberikan bantuan, dukungan dan bimbingan.

Pada kesempatan ini, tak lupa penulis mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Bapak Agung Hendarto, S.E., M.A. selaku Direktur Politeknik Harapan Bersama Tegal.
2. Bapak Slamet Wiyono, S.Pd., M.Eng selaku Ketua Program Studi D IV Teknik Informatika Politeknik Harapan Bersama Tegal.
3. Bapak Dega Surono Wibowo, S.T., M.Kom selaku Dosen Pembimbing I.
4. Bapak Rosid Mustofa, M.Kom selaku Dosen Pembimbing II.
5. Kepada orang tua penulis, Bapak Fauzi dan Ibu Eva yang telah memberi dukungan dan semangat kepada penulis.
6. Semua pihak yang telah mendukung, membantu serta mendoakan penyelesaian laporan Tugas Akhir ini.

Semoga laporan Tugas Akhir ini dapat memberikan sumbangan untuk pengembangkan ilmu pengetahuan dan teknologi.

Tegal, Desember 2021
Penulis

Evan Fauzi Pranendra

DAFTAR ISI

Halaman

HALAMAN JUDUL i
HALAMAN PERNYATAAN ii
HALAMAN REKOMENDASI iii
HALAMAN PENGESAHAN iv
ABSTRAK v
KATA PENGANTAR vi
DAFTAR ISI vii
DAFTAR TABEL x
DAFTAR GAMBAR xi
DAFTAR LAMPIRAN xii
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Perumusan Masalah 2
1.3 Pembatasan Masalah 3
1.4 Tujuan Penelitian 3
1.5 Manfaat Penelitian 3
1.6 Metodologi Penelitian 4
1.6.1 Bahan Penelitian 4
1.6.2 Alat Penelitian 4
1.6.3 Alur Penelitian 5
1.7 Sistematika Penulisan 7
BAB II TINJAUAN PUSTAKA 9
BAB III LANDASAN TEORI 16
3.1 *Internet of Thinks* (*IoT*) 16
3.2 *NodeMCU* 18
3.3 *Relay* 20
3.4 *Firebase* 22

3.5 *XAMPP* 25
3.6 *Arduino IDE* 27
3.7 Bahasa *Lua* 35
3.8 *Unified Modeling Language* 37
3.8.1 *Use Case Diagram* 37
3.8.2 *Activity Diagram* 39
3.8.3 *Sequence Diagram* 39
BAB IV PERANCANGAN DAN DESAIN 41
4.1 Perancangan Sistem 41
4.2 Perancangan *Flowchart* Aplikasi 41
4.3 Perancangan Diagram Blok 43
4.4 Perancangan *UML* 43
4.4.1 *Use Case Diagram* 43
4.4.2 *Activity Diagram* 44
4.4.3 *Sequence Diagram* 46
4.5 Perancangan Desain *Mikrokontroler* 48
4.5.1 *16-channel Analog Multiplexer* 49
4.5.2 Sensor Cahaya 50
4.5.3 Sensor Arus *ACS712* 51
4.5.4 Sensor Tegangan *ZMPT101B* 51
4.5.5 Modul *Relay* 4 *Channel* 52
4.6 Perancangan Pengujian Sistem 52
4.6.1 Perancangan Pengujian Aplikasi Android 52
4.6.2 Perancangan Pengujian Aplikasi *Web* 53
4.7 Perancangan Desain Android 53
4.7.1 Desain Layout Tampilan Beranda 54
4.7.2 Desain Layout Tampilan Cek Lampu 54
4.8 Perancangan Desain *Website* 55
4.8.1 Desain *Layout* Tampilan Beranda *Web* 55

4.8.2 Desain *Layout* Tampilan Cek Lampu *Web* 56
4.8.3 Desain *Layout* Tampilan Data Laporan *Web* 56
4.8.4 Desain *Layout* Tampilan Data Lampu *Web* 57
BAB V HASIL DAN PEMBAHASAN PENELITIAN 58
5.1 Hasil Penelitian 58
5.1.1 Rangkaian *Prototype* 58
5.1.2 Rangkaian Pemasangan Sensor *ACS712* 58
5.1.3 Rangkaian Pemasangan Sensor *ZMPT101B* 59
5.1.4 Rangkaian Pemasangan Sensor Cahaya/*LDR* 60
5.1.5 Rangkaian Kabel *Jumper* 60
5.1.6 Pengujian Perangkat Keras 61
5.1.7 Pengujian Sensor 65
5.2 Pembahasan Penelitian 67
BAB VI PENUTUP 70
6.1 Kesimpulan 70
6.2 Saran 71
DAFTAR PUSTAKA 72

DAFTAR TABEL

Halaman

Tabel 2.1 Gap analisis 13
Tabel 3.1 *history* versi lama *XAMPP* 26
Tabel 3.2 Simbol-simbol pada *Use Case Diagram* 37
Tabel 3.3 Simbol *Activity Diagram* 39
Tabel 3.4 Simbol *Sequence Diagram* 40
Tabel 4.1 Koneksi pin 16-*channel Analog Multiplexer* dengan pin B*aseborad NodeMCU* 50
Tabel 4.2 Koneksi pin sensor cahaya dengan pin *Baseboard NodeMCU* dan *16-channel Analog Multiplexer* 50
Tabel 4.3 koneksi pin sensor *ACS712* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* 51
Tabel 4.4 koneksi pin sensor *ZMPT101B* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* 51
Tabel 4.5 koneksi pin *relay* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* 52
Tabel 4.6 Perancangan pengujian aplikasi android 52
Tabel 4.7 Perancangan pengujian aplikasi *web* 53
Tabel 5.1 *Test Case* Pengujian Perangkat Keras 61
Tabel 5.2 *Test Case* Pengujian Sensor *LDR/*Cahaya 65
Tabel 5.3 *Test Case* Pengujian Sensor *ACS712* dan *ZMPT101B* 67

DAFTAR GAMBAR

Halaman

Gambar 3.1 *Board NodeMCU ESP8266*..... 19
Gambar 3.2 (a) *Relay*, (b) Simbol *Relay*..... 20
Gambar 3.3 Cara kerja *relay*..... 21
Gambar 3.4 Arsitektur Sistem *Firebase*..... 23
Gambar 3.5 Metode menulis data ke *Firebase*..... 24
Gambar 3.6 *Callback* peristiwa dalam pengambilan data *Firebase*..... 24
Gambar 3.7 Tampilan *localhost XAMPP*..... 26
Gambar 4.1 *Flowchart* Alur kerja aplikasi ketika menyalakan lampu..... 42
Gambar 4.2 *Flowchart* Alur kerja aplikasi ketika mematikan lampu..... 42
Gambar 4.3 Diagram Blok..... 43
Gambar 4.4 *Use Case Diagram* Aplikasi..... 44
Gambar 4.5 *Activity Diagram* Kontrol Lampu Aplikasi..... 45
Gambar 4.6 *Activity Diagram* Cek Lampu Aplikasi..... 46
Gambar 4.7 *Sequence Diagram* Kontrol Lampu Aplikasi..... 47
Gambar 4.8 *Sequence Diagram* Cek Lampu Aplikasi..... 48
Gambar 4.9 Desain Rangkaian *Mikrokontroler*..... 49
Gambar 4.10 Tampilan Beranda Android..... 54
Gambar 4.11 Tampilan Cek Lampu Android..... 55
Gambar 4.12 Tampilan Beranda *Website*..... 56
Gambar 4.13 Tampilan Cek Lampu *Website*..... 56
Gambar 4.14 Tampilan Data Laporan *Website*..... 57
Gambar 4.15 Tampilan Data Lampu *Website*..... 57
Gambar 5.1 Rangkaian *Prototype*..... 58
Gambar 5.2 Rangkaian Pemasangan Sensor *ACS712*..... 59
Gambar 5.3 Rangkaian Pemasangan Sensor *ZMPT101B*..... 59
Gambar 5.4 Rangkaian Pemasangan Sensor Cahaya/*LDR*..... 60
Gambar 5.5 Rangkaian Kabel *Jumper*..... 61

DAFTAR LAMPIRAN

Halaman

Lampiran 1. Surat Kesepakatan Bimbingan Tugas Akhir A-1
Lampiran 2. Lembar Bimbingan Tugas Akhir..... B-1

BAB I
PENDAHULUAN

1.1 Latar Belakang

Lampu merupakan sumber cahaya yang sangat penting sebagai pengganti matahari. Lampu sebagai sumber penerangan pada malam hari yang sangat berguna bagi masyarakat. Penggunaan lampu di masyarakat sekarang ini dinilai kurang efektif dan masih sering mengabaikan penggunaannya, sering kali lampu masih tetap menyala walaupun tidak dipakai. Hal semacam ini merupakan suatu pemborosan. Disisi lain proses mematikan dan menghidupkan lampu secara manual masih dirasa banyak membuang banyak waktu. Penggunaan lampu umumnya masih menggunakan sistem *on/off* pada saklar, misalnya pada gedung-gedung perkantoran, industri maupun sekolahan. Pada gedung seperti ini biasanya memiliki beberapa lantai, jika ingin menyalakan lampu harus menekan saklar yang dinyalakan satu persatu antar ruangan. Jika gedung bertingkat maka petugas atau karyawan harus naik turun lantai untuk menyalakan lampu satu persatu.

Seperti pada gedung sekolah, di malam hari juga perlu penerangan untuk sekolah agar tidak terjadi hal yang tidak diinginkan misalnya pencurian. Ini merupakan tugas penjaga sekolah untuk selalu memastikan lampu sekolah tetap menyala pada malam hari sampai pagi hari. Penjaga sekolah harus buka tutup pintu kelas untuk menyalakan lampu pada saklar di dalam kelas untuk lampu yang ada di depan kelas setiap hari, apalagi pada SDN Kraton 3 Tegal sudah lantai tingkat 2 maka harus naik tangga lagi untuk menyalakan lampu. Dan pada pagi hari harus sudah mematikan lampu dengan cara seperti tadi untuk mematikan lampu. Hal ini sering dikeluhkan oleh penjaga sekolah SDN Kraton 3 Tegal.

Penelitian terdahulu telah dibuat tentang "Implementasi Teknologi *Internet Of Things (IoT)* Pada Rumah Pintar Berbasis *Mikrokontroler Esp8266*"

yang telah berhasil membuat sistem kontrol lampu dengan perintah suara *google assistant* dengan media koneksi internet [7]. Penelitian yang lain dengan judul "Sistem Kendali Berbasis *Mikrokontroler* Menggunakan Protokol *MQTT* pada *Smarthome*" telah berhasil membuat sistem kendali untuk melakukan *controlling* dan *monitoring* pada *smarthome* menggunakan protokol *Message Queue Telemetry Transport (MQTT)* sebagai komunikasi di dalam sistem *smarthome* [8]. Selanjutnya penelitian dengan judul "Membangun Sistem Kendali Jarak Jauh Pada Ruang Kelas Menggunakan *Thingspeak* Berbasis *Web*" telah dibuat sistem kendali ruang jarak jauh menggunakan *website* sebagai halaman kendali dengan memanfaatkan *thingspeak* sebagai *cloud server* [12]. Berikutnya penelitian dengan judul "Penerapan *Internet Of Things (IoT)* Untuk Kontrol Lampu Menggunakan Arduino Berbasis *Web*" telah berhasil membuat sistem kontrol dan monitor lampu dengan memanfaatkan *cayenne* sebagai *server* [13].

Berdasarkan permasalahan yang sedang terjadi, maka dilakukan penelitian yang berjudul "Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU*". Aplikasi ini dapat menjadi solusi dari masalah karyawan dan penjaga sekolah dalam menyalakan lampu di lorong kelas tanpa harus menekan saklar di tiap kelas. Aplikasi ini dapat menyalakan lampu dan mengecek kondisi lampu dengan jarak jauh dengan syarat harus terhubung ke koneksi internet.

1.2 Perumusan Masalah

Rumusan masalah penelitian ini adalah sebagai berikut :

1. Apa saja alat dan bahan yang dibutuhkan untuk membuat sistem kendali lampu dengan aplikasi android?
2. Bagaimana cara membuat sistem kendali lampu dengan aplikasi android?
3. Bagaimana cara kerja *NodeMCU* untuk menyalakan lampu dengan aplikasi android?

1.3 Pembatasan Masalah

Berdasarkan latar belakang dan rumusan masalah yang diatas, maka perlu dilakukan pembatasan masalah untuk membatasi pembahasan dan pemecahan masalah agar lebih terarah.

1. Aplikasi android ini hanya dapat berjalan pada Android versi minimal *Nougat.*
2. Aplikasi dan *Mikrokontroler NodeMCU* dapat berjalan jika terhubung ke koneksi internet.
3. Aplikasi android hanya dapat menyalakan/mematikan lampu dan mengecek kondisi lampu.

1.4 Tujuan Penelitian

Tujuan penelitian ini yaitu membuat aplikasi kontrol lampu yang dapat digunakan karyawan SDN Kraton 3 Tegal untuk memudahkan menyalakan lampu tanpa harus menekan saklar lampu di tiap kelas.

1.5 Manfaat Penelitian

1. **Manfaat Bagi Mahasiswa**
   a. Menambah wawasan dan pengalaman dalam pembuatan aplikasi berbasis *Mobile Android*.
   b. Dapat menambah pengetahuan dan memperluas wawasan tentang *IoT.*
2. **Manfaat Bagi Politeknik Harapan Bersama**
   a. Menjadi bahan tambahan referensi dan pembanding penelitian-penelitian yang serupa.
   b. Sebagai literatur bagi mahasiswa lain dalam membuat tugas akhir.
3. **Manfaat Bagi SDN Kraton 3 Tegal**
   a. Menyalakan lampu di SDN Kraton 3 Tegal menjadi lebih mudah menggunakan aplikasi android.
   b. Menjadi sekolah yang menggunakan teknologi *IoT*.

1.6 Metodologi Penelitian

1.6.1 Bahan Penelitian

Data yang digunakan pada penelitian yang akan dilakukan yaitu data dari buku, jurnal, internet dan penelitian terdahulu yang berkaitan dengan penelitian yang sedang dilakukan.

1.6.2 Alat Penelitian

Alat yang digunakan dalam penelitian ini dibagi dalam perangkat keras dan perangkat lunak sebagai berikut :

1. Perangkat Keras

   Adapun perangkat keras yang digunakan untuk menunjang penelitian ini sebagai berikut :

   a. Laptop spesifikasi Intel Celeron N3060, VGA Intel HD Graphics, RAM 4 GB, dan Hardisk 1 TB.
   b. *NodeMCU ESP8266*, berfungsi sebagai pengolah data masukan dan keluaran serta menjadi penghubung antara android dengan *relay*.
   c. Modul *Relay*, berfungsi menerima perintah dari *NodeMCU* untuk mengatur lampu.
   d. Adaptor 12V DC, berfungsi sebagai *Power Supply* NodeMCU.
   e. Sensor *ACS712*, berfungsi sebagai sensor arus untuk deteksi lampu sudah menyala atau mati.
   f. Sensor *ZMPT101B*, berfungsi sebagai sensor tegangan untuk mengetahui besarnya tegangan pada lampu.
   g. Sensor *LDR*/Cahaya, berfungsi untuk mendeteksi lampu sudah menyala atau belum.
   h. Lampu LED, berfungsi sebagai media pencahayaan.

2. Perangkat Lunak

   Adapun perangkat lunak yang digunakan pada penelitian ini adalah :

   a. *Star UML*, berfungsi sebagai pemodelan sistem.

b. *Arduino IDE*, sebagai media untuk memprogram *NodeMCU*.
c. *Android Studio*, untuk membuat aplikasi android.
d. *Firebase*, untuk menyimpan database secara *realtime*.
e. *XAMPP,* untuk menjalankan *web server* pada *localhost* komputer.
f. *Balsamiq,* sebagai pembuatan *mockup.*

1.6.3 Alur Penelitian

1. Identifikasi Masalah

   Bagaimana cara membuat sistem pengendalian lampu melalui aplikasi android menggunakan *mikrokontroler NodeMCU* di SDN Kraton 3 Tegal untuk mempermudah karyawan-karyawan di tempat tersebut

2. Pengumpulan Data

   Pengumpulan data pada penelitian ini dilakukan dengan cara melakukan studi literatur dan wawancara.

   a. Studi Literatur

      Pada metode Studi literatur ini dilakukan pencarian referensi teori yang relevan dengan permasalahan yang ada. Referensi ini dapat diambil dari buku, jurnal maupun *website*. Referensi tersebut berisikan tentang segala sesuatu yang berhubungan dengan sistem pengendalian lampu melalui aplikasi android menggunakan *mikrokontroler NodeMCU* berbasis *IoT*.

   b. Wawancara

      Pada metode ini secara langsung menanyakan kepada pihak objek penelitian tentang masalah yang dihadapinya dan mencari solusinya untuk menyelesaikan masalah tersebut.

3. Analisa Data

   Analisa data dilakukan berdasarkan hasil pengumpulan data untuk dijadikan acuan terhadap permasalah yang ada ketika sistem akan di buat. Data yang peniliti lakukan adalah menganalisa data tentang keluhan

penjaga sekolah SDN Kraton 3 Tegal yang telah diberi ijin oleh Kepala Sekolah SD tersebut. Ketika repot harus bolak-balik buka pintu kelas untuk menyalakan lampu di setiap halaman depan kelas, jika sedang sakit harus pergi ke sekolah untuk menyalakan lampu di sore hari dan jika sedang libur sekolah tidak bisa mematikan lampu di pagi hari karena membiarkan lampu menyala sebab sedang liburan keluarga sehingga tidak ada yang menjaga sekolah. Jika dibiarkan lampu gelap mengundang para maling datang ke sekolah tersebut, data tersebut dapat di aplikasikan ke dalam sebuah alat *IoT* yang dapat di gunakan sebagaimana mestinya.

4. Perancangan Sistem

   Proses ini digunakan dengan melakukan perancangan pemodelan untuk menjelaskan alur sistem jalannya *NodeMCU* dalam mengendalikan lampu secara lebih rinci dan detail untuk mempermudah penggunaan nantinya. Selain itu, melakukan perancangan antarmuka aplikasi android dan *website* untuk mempermudah penggunaan dalam kontrol lampu.

5. Pembuatan Sistem

   Pada tahap ini melakukan pembuatan perangkat keras yang akan digunakan kemudian melakukan *coding* pada perangkat lunak, setelah itu perangkat keras memproses perintah dari *coding* tersebut agar dapat digunakan sebagaimana mestinya.

6. Pengujian Sistem

   Pada tahap ini melakukan pengujian *blackbox* pada sistem kontrol *NodeMCU ESP8266* dengan rangkaian *prototype* yang telah dibuat terhadap objek penelitian yaitu lampu menggunakan aplikasi android yang telah ditentukan oleh peneliti agar dapat menyalakan atau mematikan lampu. Jika sistem yang dibuat berjalan sesuai yang diharapkan, maka sistem berhasil. Jika belum, maka dilakukan pengujian ulang sampai sistem berhasil.

1.7 Sistematika Penulisan

Laporan tugas akhir ini terdiri dari 6 bab, antara lain :

1. **BAB I : PENDAHULUAN**

   Menjelaskan mengenai latar belakang masalah, perumusan masalah, batasan masalah, tujuan, manfaat, metodologi penelitian dan sistematika penulisan dari tugas akhir ini yang berjudul "Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU*".

2. **BAB II : TINJAUAN PUSTAKA**

   Menguraikan dari latar belakang yang berkaitan dengan penelitian-penelitian serupa yang pernah dilakukan sebelumnya. Berisi penjelasan tentang inti sari latar belakang, tujuan, metode, dan hasil penelitian.

3. **BAB III : LANDASAN TEORI**

   Dalam bab ini tentang landasan teori yang akan digunakan untuk penyelesaian laporan penelitian yang berkaitan dengan pembuatan tugas akhir ini.

4. **BAB IV : PERANCANGAN DAN DESAIN**

   Menguraikan secara detail rancangan dan desain terhadap penulisan yang dilakukan dari sistem yang dibangun mengenai data-data yang dibuat dalam suatu perancangan alur kerja sistem dan perancangan *prototype* sistem.

5. **BAB V : HASIL DAN PEMBAHASAN PENELITIAN**

   Dalam bab ini berisi tentang dokumentasi hasil penelitian tersebut dibahas secara detail berdasarkan penelitian yang telah selesai dilakukan.

6. BAB VI : PENUTUP

Dalam bab ini diuraikan mengenai kesimpulan, yaitu menyimpulkan suatu hasil pembahsan dari suatu penelitian yang merupakan hasil akhir dan sekaligus merupakan jawaban dari permasalahan yang ada dan juga saran-saran, sebagai arahan yang mungkin dapat bermanfaat bagi penilitian yang sejenis dengan tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

Penelitian dengan judul *Smart Home* Berbasis *IoT* Menggunakan Suara Pada *Google Assistant* mengulas mengenai perangkat rumah dapat dikontrol dengan menggunakan perintah suara sehingga lebih interaktif bagi *user* daripada dengan menggunakan tombol *On*/*Off*. Untuk menggunakan *Google Assistant* harus terkoneksi dengan internet. *Google Assistant* mampu mengubah perintah suara menjadi teks. *Google Assistant* juga dapat memberi respon setelah diperintah. Untuk perintah suara maksimal tiga kalimat dalam satu *applet* pada *(If This Then That) IFTTT*. Bahasa yang didukung *IFTTT* adalah bahasa Inggris. *IFTTT* mampu mengintegrasikan antara *Google Assistant* dan *Webhooks*. *Webhooks* selanjutnya melakukan request ke *HTTP RESTful API*. Dengan *library phpMQTT* yang terdapat di *HTTP RESTful API*, perintah di *publish* ke *MQTT Broker*. *MQTT Broker* sebagai *server* meneruskan perintah ke *microcontroller* yang terhubung dengannya. *ESP32 Dev Kit* sebagai *mikrokontroler* yang terhubung dengan *MQTT Broker* mampu berkomunikasi dengan menggunakan *wireless*. *ESP32 Dev Kit* menerima perintah dari *MQTT Broker* untuk menyalakan maupun mematikan lampu yang ada di rumah melalui *relay board*. *Relay board* digunakan sebagai saklar untuk memutus atau menyambung arus listrik yang menuju ke lampu sehingga lampu dapat menyala maupun mati. Dengan dibangunnya sebuah *Smart Home* berbasis *IoT* menggunakan Suara pada *Google Assistant* ini tentunya dapat membantu bagi orang sakit yang berada di kursi roda/tempat tidur atau orang disabilitas tetapi dapat berbicara atau orang lanjut usia yang tidak dapat mencapai saklar mampu mengontrol semua perangkat yang ada di dalam rumah dengan mudah dan lebih interaktif. Selain itu, agar dapat mengontrol perangkat rumah dari jarak yang sangat jauh dan tidak terbatas dengan jarak selama terkoneksi dengan internet, kapanpun dan dimanapun dia berada [1].

Penelitian dengan judul *Taking MQTT* and *NodeMcu* to *IOT*: *Communication* in *Internet of Things* menjelaskan Protokol *MQTT* mengizinkan korespondensi antar perangkat. Mempertimbangkan kesederhanaan akses web jarak jauh Melalui *Wi-Fi*, aplikasi klien *MQTT* berbasis *ESP8266*. Jurnal ini difokuskan pada eksekusi *Arduino*, *NodeMcu* dan protokol *MQTT* dan juga menjelaskan berbagai model layanan yang mungkin untuk komunikasi di antara Perangkat *IoT*. Jurnal ini membahas komunikasi antara klien dan server menggunakan media transmisi nirkabel (*Wi-Fi*) dan kabel (*USB*). Protokol *MQTT* adalah area fokus utama dalam pekerjaan ini, kami periksa berbagai tugas menggunakan protokol ini. Kami menguji protokol ini dengan menerbitkan dan berlangganan pesan di mesin yang sama, mesin berbeda, lalu kami menyiapkan penerbit di satu perangkat dan pelanggan di perangkat lain. *LED* terhubung ke pelanggan, ini *LED ON* atau *OFF* sesuai dengan pesan yang diterbitkan oleh klien di broker dan juga memeriksa protokol *MQTT* dengan berbagai kasus seperti membuat satu penerbit dan pelanggan yang berbeda, penerbit yang berbeda dan satu pelanggan & penerbit dan pelanggan yang berbeda bersama dengan itu juga mengamati itu jika ada kehilangan informasi atau kendur saat mendistribusikan pesan dari dua pelanggan tanpa penundaan sesaat. Jadi kami menemukan bahwa protokol *MQTT* cocok untuk semua kasus yang dijelaskan di atas tetapi kecepatan transfer informasi kurang dari koneksi serial karena tergantung pada kecepatan *Wi-Fi*. Tapi protokol ini tidak pernah kehilangan data saat itu terhubung dengan *Wi-Fi* dan menyimpan data dalam antrean [2].

Penelitian dengan judul Pengendalian Lampu Rumah Berbasis *Google Asisstant* Melalui *Smartphone* Menggunakan *NodeMCU-12E ESP8266* Di Nuke Komputer Service, telah berhasil membuat *Prototype* Sistem Kontrol Lampu *Google Assistant* dengan mikrokontroler *NodeMCU-12E ESP8266*. Prototype Sistem Kontrol Lampu *Google Assistant* dibuat menggunakanm *NodeMCU-12E ESP8266* yang diaplikasikan kedalam maket sebagai *simulator*. Ruangan rumah (lampu- lampu) *Prototype* Sistem Kontrol Lampu *Google Assistant* dikontrol melalui interface aplikasi *google assistant* pada smartphone, dan bisa juga melalui *feeds* di *website*

Adafruit.io. Terbangunnya kontrol lampu ruangan rumah pada *Prototype* Sistem Kontrol Lampu *Google Assistant* yang dapat diakses dengan satu akun google yang telah terdaftar di *adafruit.io* dan *IFTTT*. *Prototype* Sistem Kontrol Lampu *Google Assistant* dapat dikontrol dimana saja dan kapan saja selama memiliki koneksi internet [5].

Penelitian dengan judul Kontrol Lampu Berbasis *Voice Command* Pada *Raspberry PI* mengulas mengenai *Alexa Voice Service* di implementasikan pada perangkat *Raspberry Pi* dengan cara melakukan *cloning* file ke *github alexa voice service* lalu dilakukan instalasi. Pada *Alexa Skills Kit* dilakukan perancangan pada *Intent Schema, Custom Slot* dan *Sample Utterances* supaya suara yang dikirim dari *Raspberry Pi* melalui *Alexa Voice Service* dapat diterjemahkan. *Intent Schema* merupakan pemodelan interaksi dalam bentuk *JSON*. *Custom Slot* merupakan pemodelan nilai-nilai penting dari sebuah kata yang diucap. Sedangkan *Sample Utterances* merupakan pemodelan sampel pengucapan. Kode program pada *AWS Lambda* dibuat dengan Bahasa pemrograman *Node.JS*. Kode program tersebut berisi *handlers* dari *intent request* yang dikirim oleh *alexa skills kit*. Ketika *intent request* termasuk dalam *handlers* tersebut maka hasil terjemahan akan di proses untuk dimodelkan dan di *publish* ke *MQTT Broker*. Perangkat *Raspberry Pi* dapat menerima hasil perintah suara yang sudah di modelkan di *AWS Lambda* dengan cara melakukan *subscribe* ke *MQTT Broker* dengan topik *lamp_control*. Ketika dilakukan pengujian tingkat akurasi dengan 3 *speaker* berbeda. Tingkat keberhasilan dari perintah mematikan dan menyalakan sebesar 95.5%, mengubah warna 100%, mengatur intensitas 93.3% dan melakukan penjadwalan 100% [6].

Penelitian dengan judul Implementasi Teknologi *IoT* (*Internet Of Think*) Pada Rumah Pintar Berbasis *Mikrokontroler Esp 8266* mengulas mengenai Teknologi *IoT* sangat bermanfaat dalam membuat rumah pintar tersebut dikarenakan teknologi internet dapat di akses di mana pun dan kapanpun jika kita ingin mengaksesnya.dengan teknologi *IoT* ini menjawab semua permasalahan yang ada dan memberikan solusi bagi seseorang yang gemar berpergian jauh keluar kota dan kita

merasa hawatir jika rumah kita belum ada yang menyalakan lampu.penelitian ini masih dapat di kembangkan dengan cara menambahkan teknologi *CCTV* pada setiap ruangan dan kita dapat memantau keadaan rumah dengan cara memonitoring dari kejauhan dengan teknologi internet [7].

Penelitian dengan judul Sistem Kendali Berbasis *Mikrokontroler* Menggunakan Protokol *MQTT* pada *Smarthome* mengulas mengenai Sistem Kendali Berbasis *Mikrokontroler* Menggunakan Protokol *MQTT* pada *Smarthome* terintegerasi dengan baik ditujukan dengan keberhasilan sistem kendali untuk melakukan *kontroling* dan *monitoring* pada *smarthome*, penerapan protokol *MQTT* untuk komunikasi di dalam sistem smarthome digunakan untuk pengiriman pesan pada sistem dengan mekanisme *publish*/*subscribe*. Dimana *publisher* akan mengirim pesan dengan topik tertentu dan *subscriber* akan menerima pesan sesuai dengan topik yang di *subscribe*. Sehingga pengiriman dan penerimaan pesan dapat sesuai berdasarkan topik yang ditentukan. Mekanisme pengiriman antar node didalam sistem menggunakan mekanisme *publish/subscribe* dari *MQTT* dimana mikrokontroler yang terintegerasi dengan sensor melakukan publish pesan dengan topik yang telah ditentukan pada *broker*. Kemudian aplikasi sistem kendali menerima data dari *broker* sesuai dengan topik yang di *subscribe* oleh aplikasi. Aplikasi kemudian akan mempublish pesan pada *broker* sesuai dengan input yang diterima oleh aplikasi, pesan yang di *publish* oleh aplikasi akan diteruskan oleh *broker* pada *mikrokontroler* yang terintegerasi dengan *LED* untuk menentukan state/kondisi dari lampu *LED* 1 dan lampu *LED* 2. Berdasarkan hasil pengujian kehandalan sistem banyaknya paket yang dikirimkan menggunakan protokol *MQTT* dalam satu waktu mempengaruhi nilai *delta time* dimana semakin singkat jeda pengiriman waktu semakin kecil nilai *delta time*-nya, dan nilai integritas data yang dikirim dan diterima melalui protokol *MQTT* adalah 100% [8].

Penelitian dengan judul *IoT* based Voice/Text Controlled Home Appliances. *IoT* diformulasikan untuk menghubungkan, mengakses, memantau, dan mengontrol entitas dunia yang ada dari jarak jauh melalui Internet. Ketika *IoT*

dikonseptualisasikan ke rumah, itu mengubah rumah sederhana menjadi rumah pintar yang lebih aman dan otomatis. Aplikasi Rumah yang dikendalikan Suara / Teks dikembangkan di mana pengguna dapat mengakses peralatan rumah tangga dari jarak jauh. Pengguna hanya dapat memberikan perintah suara atau pesan teks yang dengannya mereka dapat menghidupkan atau mematikan peralatan tergantung pada kebutuhan. Pengguna dapat menjadwalkan status peralatan saat tidak ada secara fisik di dalamnya lingkungan Hidup. Mereka juga akan diberikan informasi mengenai jadwal sebelumnya, dan mereka juga dapat mengaktifkan peralatan untuk periode waktu tertentu. Teknologi *Node-RED* digunakan untuk fungsi aplikasi yang disematkan dengan perangkat *IoT* (*NodeMCU*). Aplikasi yang dikembangkan ini menggunakan *Dialog Flow Account*. *NodeMCU* terhubung dengan peralatan rumah tangga biasa. Sesuai parameter yang diambil dari *cloud*, *NodeMCU* mengoperasikan Peralatan Rumah Tangga. Biaya implementasi aplikasi ini sangat murah karena peralatan berperforma tinggi dan berbiaya rendah digunakan. Aplikasi ini sangat konsisten dan mahir untuk orang tua dan orang dengan kemampuan berbeda yang tidak dapat menjangkau saklar,untuk menghidupkan / mematikan perangkat [10].

Adapun Gap analisis antara peneliti terdahulu dan peneliti sekarang ini di jelaskan pada tabel 2.1.

Tabel 2.1 Gap analisis.

Penulis/Judul Jurnal	Perbandingan	
	Penelitian Sebelumnya	Penelitian Sekarang
1. A. Hanani and M. A. Hariyadi 2020 [1], Smart Home Berbasis *IoT* Menggunakan Suara Pada *Google Assistant*.	1. Mengontrol lampu dengan perintah suara *Google Assistant* 2. Menggunakan *RESTapi IFTTT* dan *library phpMQTT*	1. Mengontrol lampu dengan aplikasi Android 2. Fitur cek lampu sudah menyala/belum 3. Menggunakan database *Firebase*

2. M. Kashyap, V. Sharma, and N. Gupta 2018 [2], Taking MQTT and NodeMcu to IOT: Communication in Internet of Things	1. Menggunakan layanan *server MQTT*	1. Menggunakan layanan *server* dari *Firebase*
3. A. Purwanto and S. Lutfi 2019 [5], Pengendalian Lampu Rumah Berbasis *Google Asisstant* Melalui Smartphone Menggunakan *NodeMCU-12E ESP8266* Di Nuke Komputer Service.	1. Mengontrol lampu dengan perintah suara *Google Assistant* 2. Menggunakan layanan *server database* dari *Adafruit.io* dan *IFTTT*	1. Menggunakan aplikasi Android untuk menyalakan lampu 2. Fitur cek lampu sudah menyala/belum 3. Menggunakan layanan *server database* dari *Firebase*
4. M. I. Reza, S. R. Akbar, and H. Fitriyah 2018 [6], Kontrol Lampu Berbasis *Voice Command* Pada *Raspberry PI*	1. Mengontrol lampu dengan perintah suara dari *Alexa Voice Service* 2. Menggunakan mikrokontroler *Raspberry PI*	1. Menggunakan aplikasi Android untuk menyalakan lampu 2. Menggunakan mikrokontroler *NodeMCU ESP8266*
5. R. Rizky, Z. Hakim, A. M. Yunita, and N. N. Wardah 2020 [7],	1. Mengontrol lampu dengan perintah suara *Google Assistant*	1. Menggunakan aplikasi Android untuk menyalakan

Implementasi Teknologi *IoT* (*Internet Of Think*) Pada Rumah Pintar Berbasis Mikrokontroler ESP8266.	2. Menggunakan layanan *server database* dari *Adafruit.io* dan *IFTTT*	lampu 2. Fitur cek lampu sudah menyala/belum 3. Menggunakan layanan *server database* dari *Firebase*
6. H. A. Rochman, R. Primananda, and H. Nurwasito 2017 [8], Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol *MQTT* pada *Smarthome*	1. Menggunakan layanan *server MQTT* 2. Menggunakan mikrokontroler *Wemos D1 R2*	1. Menggunakan layanan *server* dari *Firebase* 2. Menggunakan mikrokontroler *NodeMCU ESP8266*
7. S. Uma, R. Eswari, R. Bhuvanya, and G. S. Kumar 2019 [10], *IoT* based Voice/Text Controlled Home Appliances	1. Menggunakan perintah suara/teks untuk menyalakan lampu 2. Menggunakan layanan *server* dari *DialogFlow*	1. Menggunakan aplikasi Android untuk menyalakan lampu 2. Fitur cek lampu sudah menyala/belum 3. Menggunakan layanan *server database* dari *Firebase*

BAB III
LANDASAN TEORI

3.1 *Internet of Thinks* (*IoT*)

IoT saat ini merupakan teknologi yang sedang berkembang tersebar secara global. *IoT* telah melangkah di berbagai bidang terdiri dari industri, pemerintahan, akademis, dan masih berbagai penelitian dilakukan. Sekarang ini *IoT* memainkan peran penting dalam sektor bisnis, banyak bisnis yang suram dan berkembang bergantung pada *IoT* dan Industri *IoT*. *IoT* berkembang mulai dari sipil hingga sektor keamanan. Bidang seperti pertanian, *hortikultura*, perawatan kesehatan, *space*, manufaktur, konstruksi, air, dan pertambangan, sedang transit dari pengaturan infrastruktur lama ke pengaturan *IoT* modern [10].

Berinteraksi dan berkomunikasi antara manusia dengan manusia merupakan sesuatu hal yang sudah sangat umum dan biasa dilakukan, begitu pula interaksi antara manusia dengan mesin, bagaimana jika interaksi tersebut adalah mesin dengan mesin tentu semua dimulai dan berawal dari ditemukannya teknologi seperti komputer, jaringan internet, *mikroprosessor*, sensor dan juga gadget atau *devices* yang lain. Dituliskan dalam sebuah karya ilmiah dalam McKinsey Global Institute, bahwa *IoT* adalah sebuah teknologi yang memungkinkan kita untuk menghubungkan mesin, peralatan, dan benda fisik lainnya dengan sensor jaringan dan aktuator untuk memperoleh data dan mengelola kinerjanya sendiri, sehingga dimungkinkan adanya mesin untuk saling berkolaborasi dan bahkan bertindak berdasarkan informasi baru yang diperoleh secara independen. Sebuah publikasi mengenai *IoT* menjelaskan bahwa *IoT* adalah suatu keadaan ketika benda memiliki identitas, bisa beroperasi secara intelijen, dan bisa berkomunikasi dengan sosial, lingkungan, dan penggunanya. Tujuannya adalah untuk membuat manusia berinteraksi dengan benda lebih mudah, bahkan dengan tujuan supaya benda juga bisa

saling berkomunikasi dengan benda yang lainnya. Menurut McKinsey Global Institute, dalam sebuah artikelnya yang ditullis oleh Michael Hendrix, dijelaskan bahwa *IoT* akan menjadi salah satu teknologi yang akan gencar di masa depan. *IoT* diprediksi akan membawa pengaruh perkembangan dari sisi ekonomi dengan memberikan peluang mencapai besar 2.7 sampai 6.2 triliun Dolar Amerika Serikat. Pengaruh ekonomi ini dilihat dari industri *IoT* yang akan menempati posisi ketiga terbesar setelah *mobile* internet dan *automation of knowledge work*. Hal ini membuktikan bahwa teknologi *IoT* akan benar-benar berkembang dan menjadikan sebuah tren tesendiri di dunia nantinya. Beberapa tahun ini kita sudah melihat bagaimana sensor-sensor berkembang dan memengaruhi perkembangan dan kemajuan teknologi di dunia. Tentu akan banyak perusahaan yang menggunakan *IoT*, dan berusaha untuk mengembangkannya. *IoT* saat ini masih bisa dikatakan sesuatu yang sangat baru dan masih bisa dikembangkan. Contoh dalam perkembangan *IoT* yang sudah dan masih dimungkinkan untuk dapat dikembangkan lagi adalah *IoT* yang difungsikan sebagai pemantauan jarak jauh, sistem ini merupakan salah satu bentuk sistem aplikasi yang paling sering ditemukan. Salah satu caranya adalah dengan menambahkan sensor pada suatu objek benda yang ingin dipantau atau dimonitor untuk mengetahui keberadaanya atau bahkan kondisi juga tata letaknya, Sensor tersebut dikoneksikan dengan internet dengan menambahkan sebuah pemetaan atau mapping sehingga bisa diketahui letak posisinya. Dengan demikian, akan diperoleh data apa saja yang dibutuhkan dari sensor tadi, dan dapat difungsikan untuk memantaunya dari jarak jauh dengan jaringan internet, bahkan dapat langsung dipantau dengan menggunakan handphone yang sudah mendukung untuk koneksi internet. Contoh lain dalam dunia kesehatan adalah dengan menambahkan sensor yang dapat mendeteksi temperatur, suhu, kelembaban, atau bahkan tekanan udara yang dipasangkan atau ditambahkan pada alat yag dipakai langsung oleh pasien sehingga dapat memudahkan seorang dokter dalam memantau dan memonitor kondisi pasien.

Dengan adanya *IoT* seorang dokter dapat dimudahkan dalam memantau, memeriksa dan menangani pasiennya setiap hari. Inovasi-inovasi ini sangat membantu dan apabila mampu diterapkan dapat menghasilkan penghematan biaya yang cukup besar. Mudah mengatakan bahwa smartphone yang saat ini ada adalah merupakan perangkat *IoT*. Smartphone saat ini sudah memiliki kemampuan untuk terkoneksi dengan internet, dan sudah dilengkapi dengan beberapa sensor seperti layar sentuh, sensor cahaya, akselerometer, *gyroskop*, dan kompas. Akan tetapi satu hal yang terpenting dalam *IoT*, bahwa sensor-sensor ini harus dapat mendeteksi dan berkomunikasi dengan perangkat lain, dan ini yang tidak ditemukan pada smartphone terkecuali kita memasang aplikasi yang membuatnya melakukan hal tersebut [13].

3.2 *NodeMCU*

Menurut [5], *NodeMCU* merupakan sebuah *open source platform IoT* dan pengembangan *kit* yang menggunakan bahasa pemrograman *Lua* untuk membantu dalam membuat *prototype* produk *IoT* atau bisa dengan memakai *sketch* dengan *adruino IDE*. Pengembangan *kit* ini didasarkan pada modul *ESP8266*, yang mengintegrasikan *GPIO*, *PWM (Pulse Width Modulation), IIC, 1-Wire* dan *ADC (Analog to Digital Converter)* semua dalam satu *board*. *NodeMCU* berukuran panjang 4.83cm, lebar 2.54cm, dan berat 7 gram. Board ini sudah dilengkapi dengan fitur *WiFi* dan *Firmware*-nya yang bersifat *opensource*.

Penerapan *NodeMCU* seringkali digunakan untuk pembuatan *Home Automation* yang berfungsi untuk mengontrol peralatan elektronik pada rumah antara lain lampu, kipas, televisi, dan peralatan elektronik lainnya. *Home Automation* menjadi tren saat ini karena diperkirakan akan menjadi teknologi *IoT* yang banyak dijumpai pada rumah-rumah sekarang ini. *Home Automation* memudahkan penghuni rumahnya dalam beraktifitas di dalam rumah dan mudah mengontrol peralatan elektroniknya serta menghemat biaya listrik di

rumah karena bisa melihat biaya listrik yang akan dikeluarkan menggunakan sensor-sensor yang diterapkan di *NodeMCU*.

Gambar 3.1 *Board NodeMCU ESP8266*

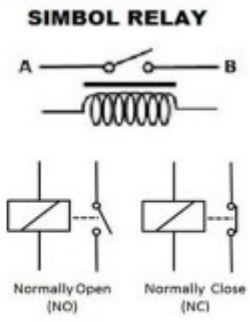
Kemudahan akses, artinya para pengguna dalam satu area dapat mengakses internet secara bersamaan tanpa perlu direpotkan dengan kabel. Pengguna yang ingin melakukan *surfing* atau *browsing* berita dan informasi di internet, cukup membawa laptop atau ponsel berkemampuan *Wi-Fi* ke tempat dimana terdapat *access point* atau *hotspot*. Perangkat ini lebih detail dan *board*-nya lebih nyaman untuk digunakan Modul *NodeMCU ESP8266-12E WiFi* dikembangkan oleh *Ai-thinker Team*. Prosesor inti *ESP8266* dalam ukuran modul yang lebih kecil *mengenkapsulasi Tensilica L106* mengintegrasikan *mikro*-daya *32-bit MCU mikro*-rendah yang sangat kuat di industri, dengan mode pendek *16-bit*, Kecepatan *clock* mendukung 80 *MHz*, 160 *MHz*, mendukung *RTOS*, terintegrasi *Wi-Fi MAC / BB / RF / PA / LNA, antena on-board.* Modul ini mendukung *IEEE802.11 b/g/n* standar, diatas protokol *TCP/IP* lengkap. Pengguna dapat menggunakan tambahkan modul ke jaringan perangkat yang ada, atau buat pengontrol jaringan terpisah. *ESP8266* adalah *SOC nirkabel* integrasi tinggi, yang dirancang untuk perancang *platform seluler* yang dibatasi ruang dan daya. Ini menyediakan kemampuan yang tak

tertandingi untuk menanamkan kemampuan *Wi-Fi* dalam sistem lain, atau berfungsi sebagai yang berdiri sendiri aplikasi, dengan biaya terendah, dan kebutuhan ruang yang minimal [5].

3.3 *Relay*

Relay merupakan komponen elektronika berupa saklar atau *switch* elektrik yang dioperasikan secara listrik dan terdiri dari 2 bagian utama yaitu *Elektromagnet* (*coil*) dan *mekanikal* (seperangkat kontak Saklar/*Switch*). Komponen elektronika ini menggunakan prinsip *elektromagnetik* untuk menggerakan saklar sehingga dengan arus listrik yang kecil (*low power*) dapat menghantarkan listrik yang bertegangan lebih tinggi [5].

Relay memiliki *switch* elektro mekanis yang mengikuti tegangan tinggi/koneksi arus yang menggunakan tegangan rendah/koneksi arus pula. Prinsip kerja *relay* pada dasarnya bekerja karena adanya medan magnet yang digunakan untuk menggerakkan saklar. Kumparan akan diberikan tegangan kerja relay yang akan menimbulkan medan magnet pada kumparan yang disebabkan oleh arus yang mengalir pada lilitan kawat [4]. Berikut adalah gambar dan juga simbol dari komponen relay.



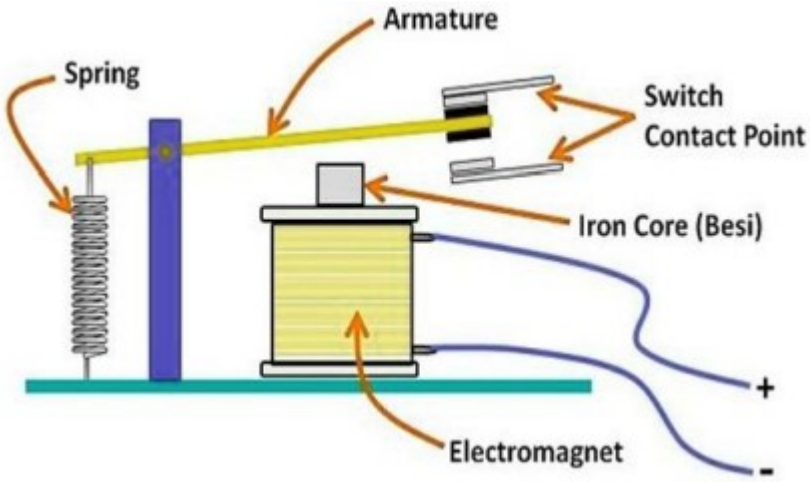
(a) (b)

Gambar 3.2 (a) *Relay*, (b) Simbol *Relay*

Relay memiliki fungsi sebagai saklar elektrik. Namun jika diaplikasikan ke dalam rangkaian elektronika, *relay* memiliki beberapa fungsi yang cukup unik. Berikut adalah beberapa fungsi komponen *relay* saat diaplikasikan ke dalam sebuah rangkaian elektronika.

1. Mengendalikan sirkuit tegangan tinggi dengan menggunakan bantuan signal tegangan rendah.
2. Menjalankan fungsi logika alias *logic function*.
3. Memberikan fungsi penundaan waktu alias *time delay function*.
4. Melindungi motor atau komponen lainnya dari kelebihan tegangan atau korsleting.

Setelah mengetahui pengertian dan fungsi *relay*, berikut adalah cara kerja atau prinsip kerja *relay* yang juga harus anda ketahui. Namun sebelumnya anda perlu tahu bahwa dalam sebuah *relay* terdapat 4 buah bagian penting yakni *Electromagnet* (*Coil*), *Armature*, *Switch Contact Point* (Saklar), dan *Spring*. Untuk info lebih jelasnya silahkan lihat gambar di bawah ini.



Gambar 3.3 Cara kerja *relay*

Dari gambar tersebut dapat diketahui bahwa sebuah Besi (*Iron Core*) yang dililit oleh kumparan *Coil*, berfungsi untuk mengendalikan Besi tersebut. Apabila kumparan *Coil* dialiri arus listrik, maka akan muncul gaya elektromagnetik yang dapat menarik *Armature* sehingga dapat berpindah dari posisi sebelumnya tertutup (*NC*) menjadi posisi baru yakni terbuka (*NO*).

Dalam posisi (*NO*) saklar dapat menghantarkan arus listrik. Pada saat tidak dialiri arus listrik, *Armature* akan kembali ke posisi awal (*NC*). Sedangkan *Coil* yang digunakan oleh *relay* untuk menarik *Contact Poin* ke posisi *close* hanya membutuhkan arus listrik yang relatif cukup kecil. Oh iya, buat anda yang belum tahu apa itu *NO* dan *NC*, berikut penjelasannya.

1. *NC* atau *Normally Close* adalah kondisi awal *relay* sebelum diaktifkan selalu berada di posisi *CLOSE* (tertutup).
2. *NO* atau *Normally Open* adalah kondisi awal *relay* sebelum diaktifkan selalu berada di posisi *OPEN* (terbuka).

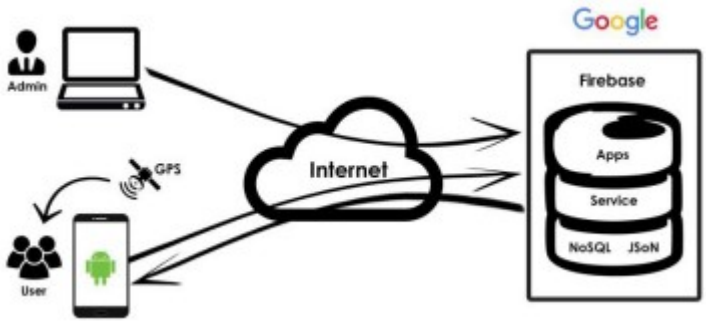
3.4 *Firebase*

Firebase Realtime Database adalah *database* yang *host* nya berada di *cloud*. Data disimpan sebagai *JSON* dan disinkronkan secara *realtime* ke setiap klien yang terhubung. *Firebase Realtime Database* menggunakan sinkronisasi data setiap kali data berubah, semua perangkat yang terhubung akan menerima *update* dalam waktu milidetik [4].

Firebase selain penyedia layanan *Realtime Database* juga menyediakan layanan *backend*. Sebuah aplikasi yang memungkinkan pengembang untuk membuat *API* yang akan disinkronkan untuk klien yang berbeda dan disimpan di *Cloud Firebase*. *Firebase* memiliki banyak *libraries* yang membuatnya mungkin untuk mengintegrasikan layanan ini dengan *Android, Ios, Javacript, Java, Objective-C* dan *Node.JS*. *Database Firebase* juga dapat diakses melalui *REST API. REST API* menggunakan *Server-Sent Event protocol* dengan membuat koneksi *HTTP* untuk menerima *notification push* dari *server*.

Pengembang menggunakan *REST API* untuk menyimpan data berikutnya *Firebase client library* yang telah diimplementasikan di aplikasi yang dibangun akan mengambil data secara *realtime*. Pengembang juga dapat menggunakan *database* ini untuk mengamankan data menggunakan *Server Firebase* dengan aturan yang ada. Untuk meng-*hosting* file, *Firebase* menyediakan *hosting* untuk file statis dengan *CND* dan fasilitas *SNL* [11].

Firebase memiliki produk utama, yaitu menyediakan *database realtime* dan *backend* sebagai layanan (*Backend as a Service*). Layanan ini menyediakan pengembang aplikasi *API* yang memungkinkan aplikasi data yang akan disinkronisasi di klien dan disimpan di *cloud Firebase* ini. *Firebase* menyediakan *library* untuk berbagai *client* platform yang memungkinkan integrasi dengan *Android, iOS, JavaScript, Java, Objective-C* dan *Node* aplikasi *Js* dan dapat juga disebut sebagai layanan *DbaaS (Database as a Service*) dengan konsep *realtime*. *Firebase* digunakan untuk mempermudah dalam penambahan fitur-fitur yang akan dibangun oleh *developer*. Dalam Gambar 3.4 ditunjukkan contoh arsitektur sistem *Firebase* dengan Android.



Gambar 3.4 Arsitektur Sistem *Firebase*

Semua data *Firebase Realtime Database* disimpan sebagai objek *JSON*. Bisa dianggap basis data sebagai *JSON tree* yang di-*host* di awan. Tidak seperti basis data *SQL*, tidak ada tabel atau rekaman. Ketika ditambahkan ke *JSON tree*, data akan menjadi simpul dalam struktur *JSON* yang ada. Meskipun basis

data menggunakan *JSON tree*, data yang tersimpan dalam basis data bisa diwakili sebagai tipe bawaan tertentu yang sesuai dengan tipe *JSON* yang tersedia untuk membantu Anda menulis lebih banyak kode yang bisa dipertahankan. Ada empat metode untuk menulis data ke *Firebase Realtime Database* pada gambar 3.5 dibawah.

Metode	Penggunaan umum
`setValue()`	Menulis atau mengedit data ke jalur yang didefinisikan, seperti **users/<user-id>/<username>**.
`push()`	Tambahkan ke daftar data. Setiap kali Anda memanggil **push( )**, Firebase akan menghasilkan ID unik, seperti **user-posts/<user-id>/<unique-post-id>**.
`updateChildren()`	Memperbarui beberapa kunci untuk jalur yang didefinisikan tanpa mengganti semua data.
`runTransaction()`	Memperbarui data kompleks yang bisa rusak karena pembaruan bersamaan.

Gambar 3.5 Metode menulis data ke *Firebase*

Untuk operasi tulis dasar, Anda bisa menggunakan *setValue()* untuk menyimpan data ke referensi yang ditetapkan, menggantikan data yang ada di jalur tersebut. Fungsi dalam pengambilan data melalui *Firebase* pada gambar 3.6 dibawah.

Pemroses	Callback peristiwa	Penggunaan standar
`ValueEventListener`	`onDataChange()`	Membaca dan memproses perubahan pada seluruh konten di sebuah jalur.

Pemroses	Callback peristiwa	Penggunaan standar
`ChildEventListener`	`onChildAdded()`	Mengambil daftar item atau memproses penambahan ke daftar item. Callback ini dipicu satu kali untuk setiap turunan yang ada, dan dipicu lagi setiap kali ada turunan baru yang ditambahkan ke jalur yang ditetapkan. `DataSnapshot` yang diteruskan ke pemroses memuat data dari turunan baru.
	`onChildChanged()`	Memproses perubahan pada item dalam daftar. Peristiwa ini dipicu setiap kali node turunan diubah, termasuk setiap perubahan pada turunan dari node turunan. `DataSnapshot` yang diteruskan ke pemroses peristiwa berisi data turunan yang diperbarui.
	`onChildRemoved()`	Memproses item yang dihapus dari daftar. `DataSnapshot` yang diteruskan ke callback peristiwa berisi data untuk turunan yang dihapus.
	`onChildMoved()`	Memproses perubahan urutan item dalam daftar yang diurutkan. Peristiwa ini dipicu setiap kali callback `onChildChanged()` dipicu oleh pembaruan yang menyebabkan pengurutan ulang turunan. Peristiwa ini digunakan dengan data yang diurutkan dengan `orderByChild` atau `orderByValue`.

Gambar 3.6 *Callback* peristiwa dalam pengambilan data *Firebase*

Untuk menambahkan *listener* peristiwa, gunakan metode *addValueEventListener()* atau *addListenerForSingleValueEvent()*. Untuk menambahkan *listener* kejadian anak, gunakan metode *addChildEventListener()*. Metode *onDataChange()* untuk membaca cuplikan statis konten pada jalur tertentu, seperti yang telah ada pada saat kejadian. Metode ini terpicu satu kali ketika *listener* terpasang dan terpicu lagi setiap kali terjadi perubahan data, termasuk anaknya. *Callback* peristiwa meneruskan cuplikan yang berisi semua data di lokasi tersebut, termasuk data anak. Jika tidak ada data, cuplikan yang dikembalikan adalah *null*. Metode *onDataChange()* dipanggil setiap kali terjadi perubahan data pada referensi *database* yang ditetapkan, termasuk perubahan ke anaknya.

3.5 *XAMPP*

XAMPP merupakan program *Apache Web Server*. Program *XAMPP* selain memiliki kemampuan sebagai penyedia layanan *web*, juga memiliki basis data terintegrasi yaitu *MySQL*. Basis data diperlukan untuk menyimpan data-data dari proses aktifitas pengguna pada sistem. Proses instalasi *web server XAMPP* begitu mudah tinggal mengeksekusi file setup *XAMPP*, dan proses instalasi pun berjalan. Setelah selesai proses instalasi, melakukan pengujian terhadap layanan *web server*. Masukkan alamat *web* dengan mengetikkan teks "*localhost*", maka muncul halaman *web XAMPP* yang menandakan bahwa layanan *web server* telah bekerja [9].

Gambar 3.7 Tampilan *localhost XAMPP*

XAMPP adalah perangkat lunak bebas, yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Fungsinya sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri atas program *Apache HTTP Server*, *MySQL database*, dan penerjemah bahasa yang dirilis dengan bahasa pemograman *PHP* dan *Perl*. Nama *XAMPP* merupakan singkatan dari *X* (empat system operasi apapun), *Apache, MySQL, PHP* dan *Perl*. Program ini tersedia dalam *GNU General Public License* dan bebas, merupakan *web server* yang mudah digunakan yang dapat melayani tampilan halaman *web* yang dinamis.

XAMPP dikembangkan dari sebuah tim proyek bernama *Apache Friends*, yang terdiri dari Tim Inti (*Core Team*), Tim Pengembang (*Development Team*) & Tim Dukungan (*Support Team*). Daftar *history* versi lama dari *XAMPP* pada tabel 3.1 dibawah.

Tabel 3.1 *history* versi lama *XAMPP*

Versi	*Apache*	*MySQL*	*PHP 5*	*PHP 4*
1.8.3	2.4.9	5.6.16	5.5.11	
1.8.2	2.4.9	5.5.36	5.4.27	

1.8.1	2.4.3	5.5.27	5.4.7	
1.8.0	2.4.2	5.5.25a	5.4.4	
1.7.7	2.2.21	5.5.16	5.3.8	
1.7.5	2.2.21	5.5.15	5.3.8	
1.7.4	2.2.17	5.5.8	5.3.5	
1.7.3	2.2.14	5.1.41	5.3.1	
1.7.2	2.2.12	5.1.37	5.3.0	
1.7.1	2.2.11	5.1.33	5.2.9	
1.7.0	2.2.11	5.1.30	5.2.8	
1.6.8	2.2.9	5.0.67	5.2.6	4.4.9
1.6.7	2.2.9	5.0.51b	5.2.6	4.4.8
1.6.6a	2.2.8	5.0.51a	5.2.5	4.4.8
1.6.6	2.2.8	5.0.51	5.2.5	4.4.8 (RC2)
1.6.5	2.2.6	5.0.51	5.2.5	4.4.7
1.6.4	2.2.6	5.0.45	5.2.4	4.4.7
1.6.3a	2.2.4	5.0.45	5.2.3	4.4.7
1.6.3	2.2.4	5.0.45	5.2.3	4.4.7
1.6.2	2.2.4	5.0.41	5.2.2	4.4.7
1.6.1	2.2.4	5.0.37	5.2.1	4.4.6
1.6.0a	2.2.4	5.0.33	5.2.1	4.4.5
1.6.0	2.2.3	5.0.33	5.2.1	4.4.5

3.6 *Arduino IDE*

Integrated Development Environment (IDE) yang diperuntukan untuk membuat perintah atau *source code*, melakukan pengecekan kesalahan, kompilasi, upload program, dan menguji hasil kerja *arduino* melalui *serial monitor*. *Arduino IDE* adalah perangkat lunak yang sangat canggih ditulis dengan menggunakan *Java*. *Arduino IDE* terdiri dari: *Editor* program, sebuah

window yang memungkinkan pengguna menulis dan pengedit program dalam bahasa *Processing* [5].

IDE merupakan lingkungan terintergrasi yang digunakan untuk melakukan pengembangan. Disebut sebagai lingkungan karena melalui *software* inilah arduino dilakukan pemrograman untuk melakukan fungsi-fungsi yang dibenamkan melalui sintaks pemrograman. *Arduino IDE* dibuat dari Bahasa pemrograman *JAVA*. *Arduino IDE* juga dilengkapi dengan *library C/C++* yang biasa disebut *Wiring* yang membuat operasi input dan output menjadi lebih mudah. *Arduino IDE* ini dikembangkan dari *software processing* yang diubah menjadi *arduino IDE* khusus untuk pemrograman dengan *arduino* [12]. *Arduino IDE* terdiri dari :

1. *Editor program*, sebuah *window* yang memungkinkan pengguna menulis dan mengedit program dalam bahasa processing.
2. *Verify/Compiler*, sebuah modul yang mengubah kode program (bahasa *processing*) menjadi kode *biner*.
3. *Uploader*, sebuah modul yang memuat kode *biner* dari komputer ke dalam memori *mikrokontroler* di dalam papan *arduino*.

Berikut adalah penjelasan menu pada software *arduino IDE*.

1. *Verify* : Mengecek apakah ada kesalahan atau tidak pada program
2. *Upload* : Memasukkan kode program kedalam modul Arduino
3. *New* : Membuat program baru
4. *Open* : Membuka program yang pernah dibuat
5. *Save* : Menyimpan program yang sudah dibuat

Arduino IDE bisa dijalankan di komputer dengan berbagai macam platform karena didukung atau berbasis *java*. *Source* program yang dibuat untuk aplikasi *mikrokontroler* adalah bahasa *C/C++* dan dapat digabungkan dengan *assembly*.

1. Struktur

Setiap program *Arduino* (biasa disebut *sketch*) mempunyai dua buah fungsi yang harus ada, antara lain:

a.) *void setup( ) { }*

Semua kode didalam kurung kurawal akan dijalankan hanya satu kali ketika program *Arduino* dijalankan untuk pertama kalinya.

b.) *void loop( ) { }*

Fungsi ini akan dijalankan setelah setup (fungsi *void setup*) selesai. Setelah dijalankan satu kali fungsi ini akan dijalankan lagi, dan lagi secara terus menerus sampai catu daya (*power*) dilepaskan.

2. *Serial*

Serial digunakan untuk komunikasi antara *arduino board*, komputer atau perangkat lainnya. *Arduino board* memiliki minimal satu *port serial* yang berkomunikasi melalui *pin 0 (RX)* dan *1 (TX)* serta dengan komputer melalui USB. Jika menggunakan fungsi-fungsi ini, pin 0 dan 1 tidak dapat digunakan untuk *input digital* atau *output digital*. Terdapat beberapa fungsi *serial* pada *arduino*, antara lain:

a.) *Serial.begin( )*

Fungsi ini digunakan untuk transmisi data *serial* dan mengatur data *rate* dalam *bits per second (baud*). Untuk berkomunikasi dengan komputer gunakan salah satu dari angka ini: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, atau 115200.

b.) *Serial.available( )*

Fungsi ini digunakan untuk mendapatkan jumlah data *byte* (*characters*) yang tersedia dan membacanya dari *port serial*. Data tersebut adalah data yang telah tiba dan disimpan dalam *buffer serial* yang menampung sampai 64 *bytes*.

c.) *Serial.read( )*

Fungsi digunakan untuk membaca data *serial* yang masuk.

d.) *Serial.print( )* dan *Serial.println( )*

Fungsi ini digunakan untuk mencetak data ke *port serial* dalam *format text ASCII*. Sedangkan fungsi *Serial.println( )* sama seperti fungsi *Serial.print( )* hanya saja ketika menggunakan fungsi ini akan mencetak data dan kemudian diikuti dengan karakter *newline* atau *enter*.

3. *Syntax*

Berikut ini adalah elemen bahasa *C* yang dibutuhkan untuk format penulisan.

a.) // (komentar satu baris)

Kadang diperlukan untuk memberi catatan pada diri sendiri apa arti dari kode-kode yang dituliskan. Cukup menuliskan dua buah garis miring dan apapun yang kita ketikkan dibelakangnya akan diabaikan oleh program.

b.) /* */ (komentar banyak baris)

Digunakan untuk menulis beberapa baris sebagai komentar. Semua hal yang terletak di antara dua simbol tersebut akan diabaikan program.

c.) { } (kurung kurawal)

Digunakan untuk mendefinisikan kapan blok program mulai dan berakhir (digunakan juga pada fungsi dan pengulangan).

d.) ; (titk koma)

Setiap baris kode harus diakhiri dengan tanda titik koma (jika ada titik koma yang hilang maka program tidak akan bisa dijalankan).

4. Tipe Data

Sebuah program secara garis besar dapat didefinisikan sebagai instruksi untuk memindahkan angka dengan cara yang cerdas. Tipe data inilah yang digunakan untuk memindahkannya.

a.) *int (integer)*

Digunakan untuk menyimpan angka dalam 2 *byte* (16 *bit*). Tidak mempunyai angka desimal dan menyimpan nilai dari -32,768 sampai 32,767.

b.) *long (long)*

Digunakan untuk menyimpan angka dalam 4 *byte* (32 *bit*). Mempunyai rentang dari -2,147,483,648 sampai 2,147,483,647.

c.) *boolean (boolean)*

Variabel sederhana yang digunakan untuk menyimpan nilai *TRUE* (benar) atau *FALSE* (salah). Sangat berguna karena hanya menggunakan 1 *bit* dari RAM.

d.) *float (float)*

Digunakan untuk angka desimal (*floating point*). Memakai 4 *byte* (32 *bit*) dari RAM dan mempunyai rentang dari -3.4028235E+38 dan 3.4028235E+38.

e.) *char (character)*

Menyimpan 1 karakter menggunakan kode *ASCII* (misalnya „A“ = 65). Hanya memakai 1 *byte* (8 *bit*) dari RAM.

5. Operator Matematika

Operator yang digunakan untuk memanipulasi angka (bekerja seperti matematika yang sederhana).

a.) = (sama dengan)

Membuat sesuatu menjadi sama dengan nilai yang lain (misalnya: x = 10 * 2, x sekarang sama dengan 20).

b.) % (persen)

Menghasilkan sisa dari hasil pembagian suatu angka dengan angka yang lain (misalnya: 12 % 10, ini akan menghasilkan angka 2).

c.) + (penjumlahan)

Menjumlahkan nilai dari suatu variabel dengan nilai variabel yang lain.

d.) – (pengurangan)

Mengurangkan nilai dari suatu variabel dengan nilai variabel yang lain.

e.) * (perkalian)

Mengalikan nilai dari suatu variabel dengan nilai variabel yang lain.

f.) / (pembagian)

Membagikan nilai dari suatu variabel dengan nilai variabel yang lain.

6. Operator Pembanding

Digunakan untuk membandingkan nilai logika.

a.) ==

Sama dengan (misalnya: 12 == 10 adalah *FALSE* (salah) atau 12 == 12 adalah *TRUE* (benar)).

b.) !=

Tidak sama dengan (misalnya: 12 != 10 adalah *TRUE* (benar) atau 12 != 12 adalah *FALSE* (salah)).

c.) <

Lebih kecil dari (misalnya: 12 < 10 adalah *FALSE* (salah) atau 12 < 12 adalah *FALSE* (salah) atau 12 < 14 adalah *TRUE* (benar)).

d.) >

Lebih besar dari (misalnya: 12 > 10 adalah *TRUE* (benar) atau 12 > 12 adalah *FALSE* (salah) atau 12 > 14 adalah *FALSE* (salah)).

7. Struktur Pengaturan

Program sangat tergantung pada pengaturan apa yang akan dijalankan berikutnya, berikut ini adalah elemen dasar pengaturan.

a.) *If else*, dengan format seperti berikut ini:

```
if (kondisi) { }
else if (kondisi) { }
else { }
```

Dengan struktur seperti diatas program akan menjalankan kode yang ada di dalam kurung kurawal jika kondisinya *TRUE*, dan jika tidak (*FALSE*) maka akan diperiksa apakah kondisi pada *else if* dan jika kondisinya *FALSE* maka kode pada *else* yang akan dijalankan.

b.) *While*, dengan format seperti berikut ini:

```
While (kondisi) {}
```

Dengan struktur ini, *while* akan melakukan pengulangan terus menurus dan tak terbatas sampai kondisi didalam kurung ( ) menjadi *false*.

c.) *for*, dengan format seperti berikut ini:

```
for (int i = 0; i < #pengulangan; i++) { }
```

Digunakan bila ingin melakukan pengulangan kode di dalam kurung kurawal beberapa kali, ganti #pengulangan dengan jumlah pengulangan yang diinginkan. Melakukan penghitungan ke atas dengan i++ atau ke bawah dengan i --.

8. Operator *Boolean*

Operator ini dapat digunakan dalam kondisi *if*, antara lain:

a.) && (logika *and*), dengan format seperti berikut ini:

```
if (digitalRead(2) == HIGH && digitalRead(3) == HIGH) {}
```

Digunakan bila ingin mendapatkan nilai yang *true* hanya jika kedua input bernilai *HIGH*.

b.) || (logika *or*), dengan format seperti berikut ini:

```
if (x > 0 || y > 0) {}
```

Digunakan bila ingin mendapatkan nilai yang *true* hanya jika nilai x atau y lebih besar dari 0.

9. *Digital*

a.) *pinMode (pin, mode)*

Digunakan untuk menetapkan mode dari suatu pin, pin adalah nomor pin yang akan digunakan dari 0-19 (pin *analog* 0-5 adalah 14-19). Mode yang bisa digunakan adalah *INPUT* atau *OUTPUT*.

b.) *digitalWrite (pin, value)*

Ketika sebuah pin ditetapkan sebagai *OUTPUT*, pin tersebut dapat dijadikan *HIGH* (5 *volt*) atau *LOW* (diturunkan menjadi *ground*).

c.) *digitalRead (pin)*

Ketika sebuah pin ditetapkan sebagai *INPUT* maka anda dapat menggunakan kode ini untuk mendapatkan nilai pin tersebut apakah *HIGH* (5 volt) atau *LOW* (diturunkan menjadi *ground*).

10. *Analog*

Arduino adalah mesin *digital* tetapi mempunyai kemampuan untuk beroperasi di dalam *analog*. Berikut ini cara untuk menghadapi hal yang bukan *digital*.

a.) *analogWrite (pin, value)*

Beberapa pin pada *Arduino* mendukung *Pulse Width Modulation (PWM)* yaitu pin 3, 5, 6, 9, 10, 11. Ini dapat merubah

pin hidup (*on*) atau mati (*off*) dengan sangat cepat sehingga membuatnya dapat berfungsi layaknya keluaran *analog*.

b.) *analogRead (pin)*

Ketika pin *analog* ditetapkan sebagai *INPUT* anda dapat membaca keluaran *voltase*-nya. Keluarannya berupa angka antara 0 (untuk 0 *volt*) dan 1024 (untuk 5 *volt*).

3.7 Bahasa *Lua*

Lua yang berarti "bulan" dalam bahasa Portugis. *Lua* merupakan bahasa pemrograman ringkas yang dirancang sebagai bahasa pemrograman dinamis berbasis skrip dengan semantik yang dapat dikembangkan atau ditambahkan. Sebagai bahasa skrip, *Lua* memiliki *API* dalam bahasa *C* yang relatif lebih sederhana dibandingkan bahasa skrip lainnya.

Lua ditulis pertama kali oleh Roberto Ierusalimschy, Luiz Henrique de Figueiredo, dan Waldemar Celes yang merupakan anggota *Computer Graphics Technology Group (Tecgraf)* pada Universitas Pontifical Catholic, Rio de Janeiro, Brasil, pada tahun 1993.

Sebelumnya, sejak tahun 1977 hingga tahun 1992, pemerintah Brazil menerapkan kebijakan perdagangan yang membatasi perdagangan termasuk pertukaran baik perangkat keras ataupun perangkat lunak komputer. Dalam atmosfer yang demikian, banyak klien *Tecgraf* tidak mampu, baik secara politik maupun finansial, untuk membeli perangkat lunak dari luar. Alasan itulah yang mendorong *Tecgraf* untuk mengimplementaskan perangkat utilitas dasar yang dibutuhkan.

Secara historis, bahasa pemrograman *Lua* berawal dari bahasa pemrograman yang digunakan untuk pendeskripsian serta entri data, serta pengkonfigurasian suatu sistem yang masing-masing dikembangkan secara terpisah oleh *tecgraf* dari tahun 1992 hingga tahun 1993. Tujuan pengembangan tersebut awalnya merupakan usaha untuk menambahkan fitur serta

meningkatkan fleksibilitas atas dua proyek pengembangan perangkat lunak yang sedang mereka kerjakan saat itu. Namun terdapat kekurangan atas alur kontrol atas bahasa tersebut sehingga menumbuhkan ide untuk mengembangkannya lebih lanjut dengan fitur-fitur dasar yang lengkap sebagai sebuah bahasa pemrograman.

Secara umum *Lua* digambarkan sebagai bahasa pemrograman multi paradigma yang menyediakan seperangkat kecil atas fitur-fitur umum yang dapat dikembangkan lebih lanjut untuk memenuhi ragam kebutuhan yang berbeda-beda, karenanya *Lua* tidak menyediakan fitur yang lengkap dan kompleks yang hanya berfokus pada satu paradigma pemrograman. Sebagai contoh, *Lua* tidak secara eksplisit mendukung inheritance yang merupakan kemampuan untuk menurunkan sifat sebuah objek dalam konsepsi pemrograman berorientasi objek namun *Lua* menyediakan fasilitas metatable yang memungkinkan hal tersebut diimplementasikan relatif cukup mudah.

Secara umum, *Lua* berusaha untuk menyediakan fitur-fitur abstrak/meta yang lebih fleksibel dan dapat dikembangkan jika dibutuhkan dibandingkan menyediakan seperangkat pustaka yang lengkap untuk memenuhi satu kebutuhan tertentu. Hal tersebut menjadikan *Lua* sebagai bahasa pemrograman yang ringkas dan dapat secara mudah diadaptasikan untuk memenuhi beragam jenis kebutuhan.

Lua merupakan bahasa pemrograman dinamis yang ditujukan untuk digunakan sebagai bahasa skrip, dan cukup ringkas untuk disisipkan dalam berbagai jenis platform utama. *Lua* hanya mendukung beberapa jenis struktur data atomik seperti; boolean, floating point, serta string. Jenis-jenis tipe/struktur data lainnya seperti; larik, set, ataupun list direpresentasikan dalam Lua melalui satu bentuk tipe data, table.

3.8 *Unified Modeling Language*

Unified Modeling Language (UML) adalah metode pemodelan secara visual sebagai sarana untuk merancang atau membuat *software* berorientasi objek. Karena *UML* ini merupakan bahasa visual untuk pemodelan bahasa berorientasi objek, maka semua elemen dan diagram berbasiskan pada paradigma *object oriented*.

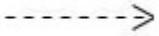
UML sendiri juga memberikan standar penulisan sebuah sistem *blueprint*, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema *database*, dan komponen-komponen yang diperlukan dalam sistem *software*. Berikut ini merupakan penjelasan dari bentuk diagram.

3.8.1 *Use Case Diagram*

Use Case Diagram adalah diagram fungsionalitas yang disediakan system sebagai unit yang saling bertukar pesan antar unit atau aktor.

Tabel 3.2 Simbol-simbol pada *Use Case Diagram*

No	Simbol	Nama	Keterangan
1		*Actor*	Menyelaraskan himpuan peran yang digunakan Ketika berinteraksi dengan *use case.*
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elmenen mandiri (*independent*) akan mempengaruhi elemen (*independent*).
3		*Generalization*	Hubungan dimana anak objek (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya induk objek

			(*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

3.8.2 *Activity Diagram*

Activity Diagram lebih memfokuskan diri pada alur sistem dari pada cara sistem itu dirakit. Diagram aktivitas menunjukan aktifitas sistem terdapat bentuk aksi. Ketika digunakan untuk pemodelan *software* dengan aktifitas merepresentasikan pemanggilan dari fungsi tertentu.

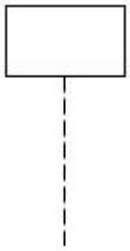
Tabel 3.3 Simbol *Activity Diagram*

No	Simbol	Nama	Keterangan
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu dan lain.
2		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
3		*Initian Node*	Merupakan objek dibentuk atau diawali.
4		*Activity Final Node*	Terdapat objek dibentuk dan dihancurkan.
5		*Fork Node*	Satu aliran yang terdapat pada tahap tertentu berubah menjadi beberapa aliran.

3.8.3 *Sequence Diagram*

Sequence Diagram digunakan untuk menggambarkan sikap objek pada *use case*, memperlihatkan objek dan *massage* yang dikirimkan dan diterima antar objek.

Tabel 3.4 Simbol *Sequence Diagram*

No	Simbol	Nama	Keterangan
1		*Life Line*	Objek *entity*, antarmuka dan saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi tentang aktifitas yang terjadi.
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi tentang aktifitas yang terjadi.

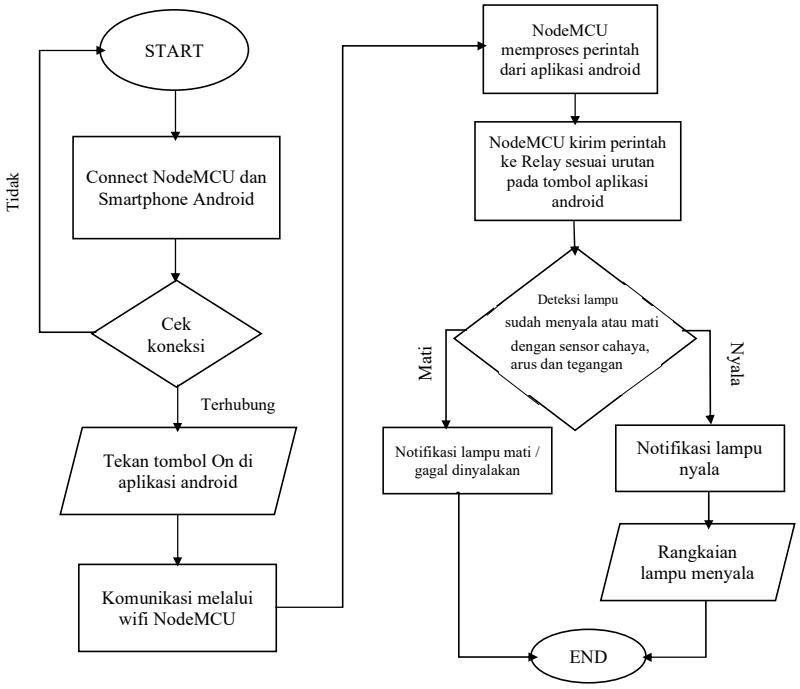
BAB IV
PERANCANGAN DAN DESAIN

4.1 Perancangan Sistem

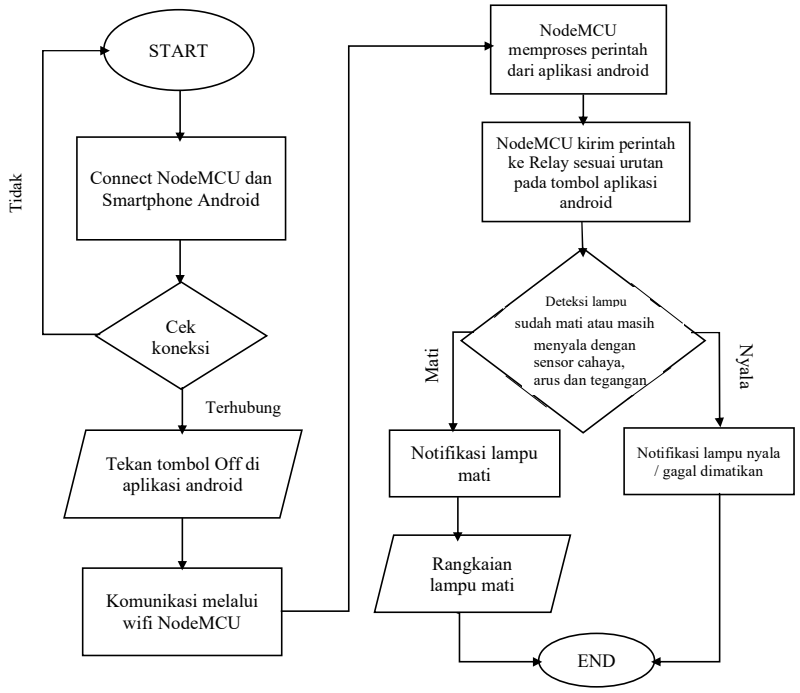
Sistem yang akan dibuat adalah Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU* pada studi kasus di SDN Kraton 3 Tegal. Menggunakan *mikrokontroler NodeMCU* yang sudah dilengkapi dengan modul *wifi ESP8266* jadi bisa di integrasikan dengan internet melalui *wifi*. Sensor yang digunakan antara lain sensor arus *ACS712* untuk membaca besarnya arus, sensor tegangan *ZMPT101B* untuk membaca besarnya tegangan, dan sensor cahaya untuk mendeteksi lampu menyala atau belum. Serta modul *relay* yang digunakan untuk mengontrol lampu menyala atau mati sesuai perintah yang dikendalikan melalui aplikasi android. Android mengirim perintah ke *database Firebase* kemudian *NodeMCU* akan membaca nilai *input relay* dari *Firebase* yang kemudian akan menyalakan atau matikan lampu dan *NodeMCU* mengirim nilai dari ketiga sensor tersebut ke *Firebase* yang akan ditampilkan di aplikasi android untuk memunculkan notifikasi apakah lampu sudah benar menyala atau mati.

4.2 Perancangan *Flowchart* Aplikasi

Alur kerja sistem dari penilitian ini digambarkan ke dalam *flowchart* berikut :



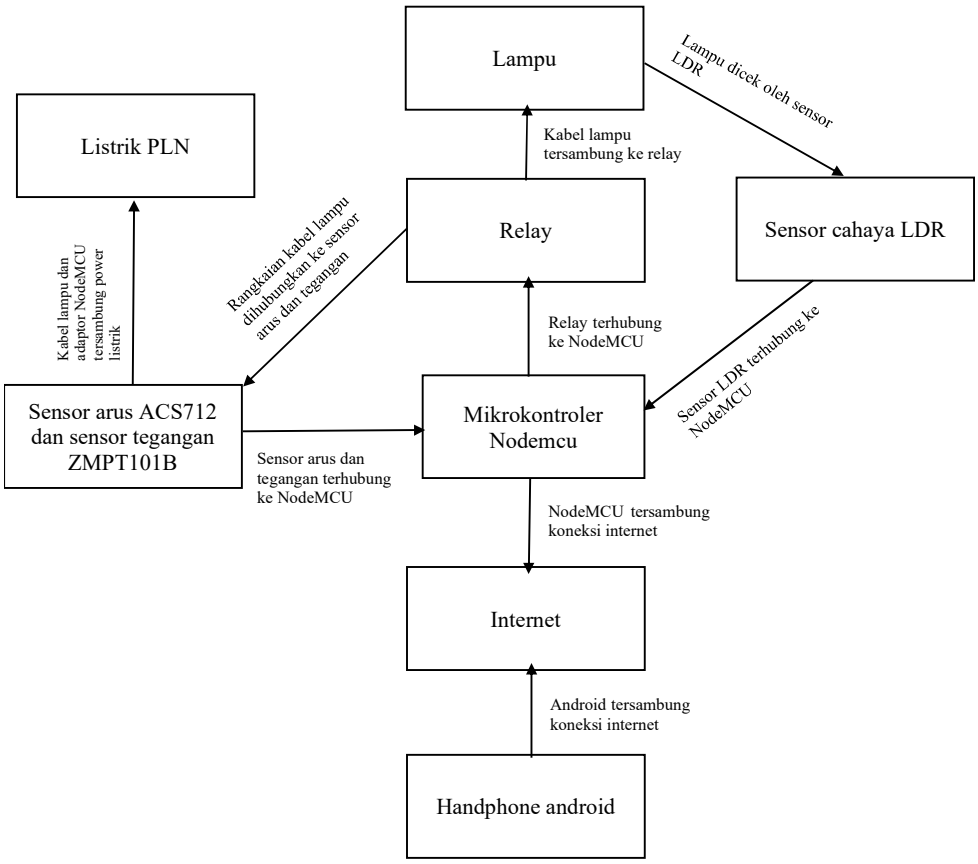
Gambar 4.1 *Flowchart* Alur kerja aplikasi ketika menyalakan lampu



Gambar 4.2 *Flowchart* Alur kerja aplikasi ketika mematikan lampu

4.3 Perancangan Diagram Blok

Pada gambar 4.3 adalah diagram blok sistem kerja alat.



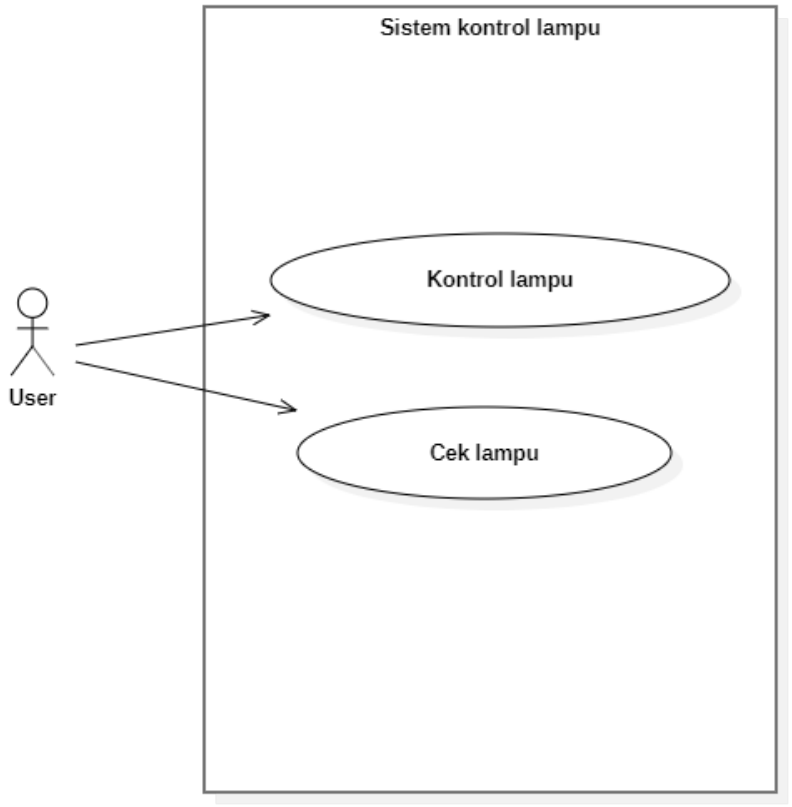
Gambar 4.3 Diagram Blok Cara Kerja Aplikasi

4.4 Perancangan *UML*

UML memiliki beberapa jenis diagram yang digambarkan untuk perancangan sistem. Berikut perancangan dan pemodelan sistem dari penelitian ini.

4.4.1 *Use Case Diagram*

Use Case dari sistem penelitian ini pada gambar 4.4 berikut :



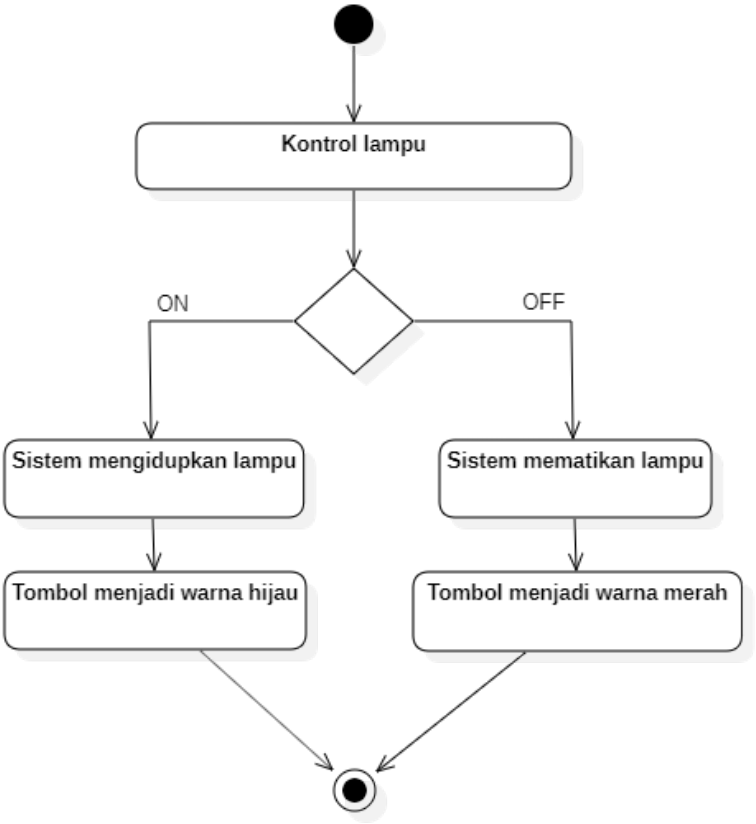
Gambar 4.4 *Use Case Diagram* Aplikasi

4.4.2 *Activity Diagram*

Pada sistem penelitian ini terdapat 2 proses *activity diagram* yang dibuat yaitu :

1. *Activity Diagram* Kontrol Lampu

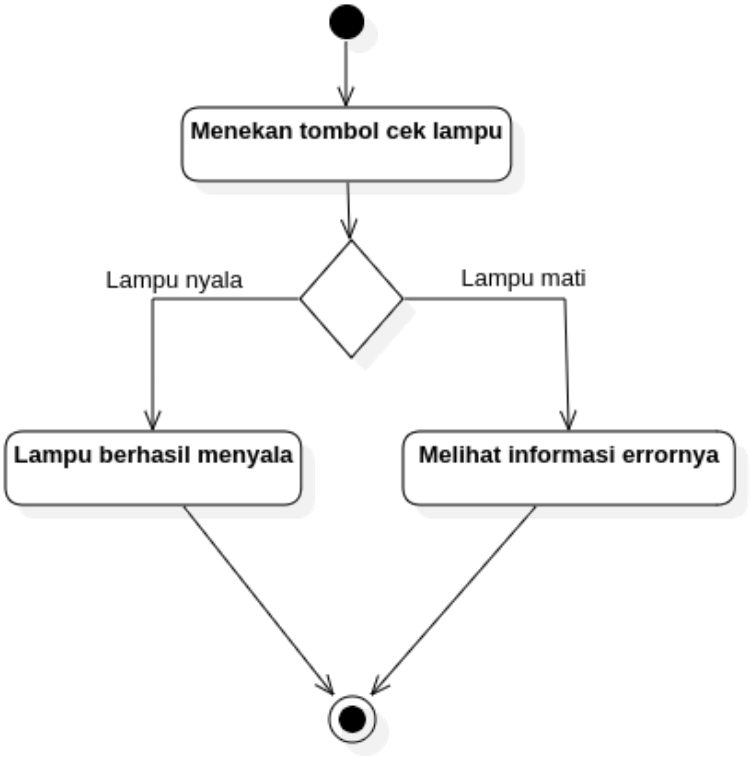
Pada proses ini menjelaskan bagaimana sistem kerja ini pada aplikasi android yang nanti akan dibaca di *NodeMCU* dalam menyalakan lampu.



Gambar 4.5 *Activity Diagram* Kontrol Lampu Aplikasi

2. *Activity Diagram* Cek Lampu

Pada proses ini menjelaskan bagaimana aplikasi android menampilkan notifikasi yang dikirim oleh sensor-sensor yang terbaca pada *NodeMCU* untuk mendeteksi lampu dan menampilkan pesan menyala atau *error*.



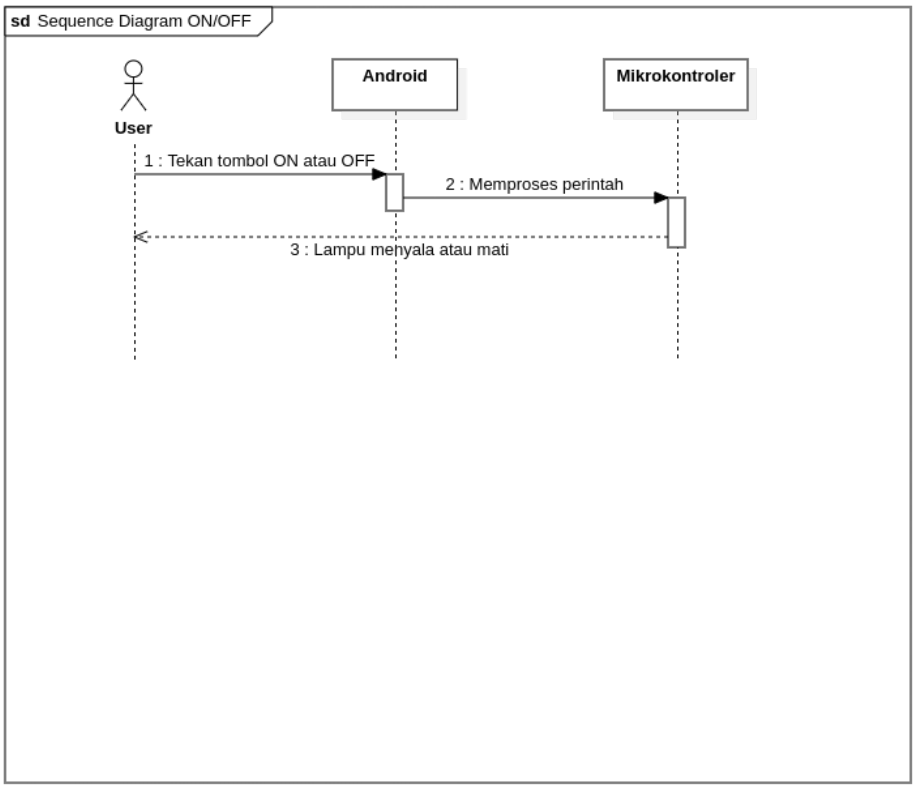
Gambar 4.6 *Activity Diagram* Cek Lampu Aplikasi

4.4.3 *Sequence Diagram*

Sequence dapat menggambarkan tahapan sistem yang harus dilakukan untuk dapat menghasilkan proses yang tertera pada *Use Case* diagram. Pada sistem penelitian ini terdapat 2 proses *sequence diagram* yang dibuat yaitu :

1. *Sequence Diagram* Kontrol Lampu

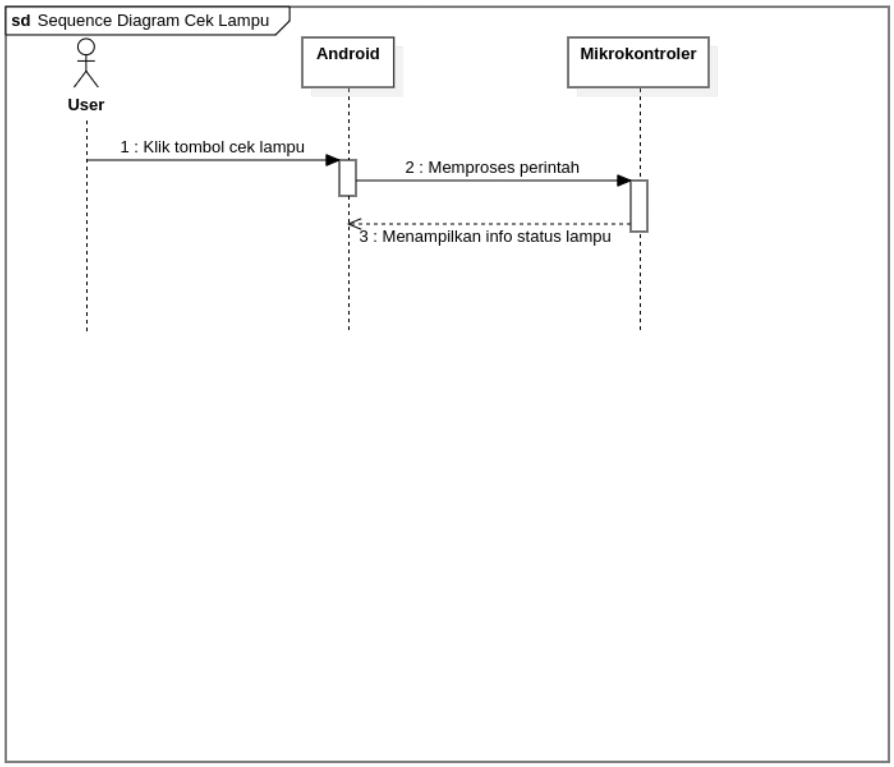
   Pada diagram ini menunjukan proses *user* melakukan perintah di aplikasi android kemudian *mikrokontroler* memproses perintah untuk menyalakan lampu kemudian di kembalikan lagi ke *user* jika lampu sudah menyala.



Gambar 4.7 *Sequence Diagram* Kontrol Lampu Aplikasi

2. *Sequence Diagram* Cek Lampu

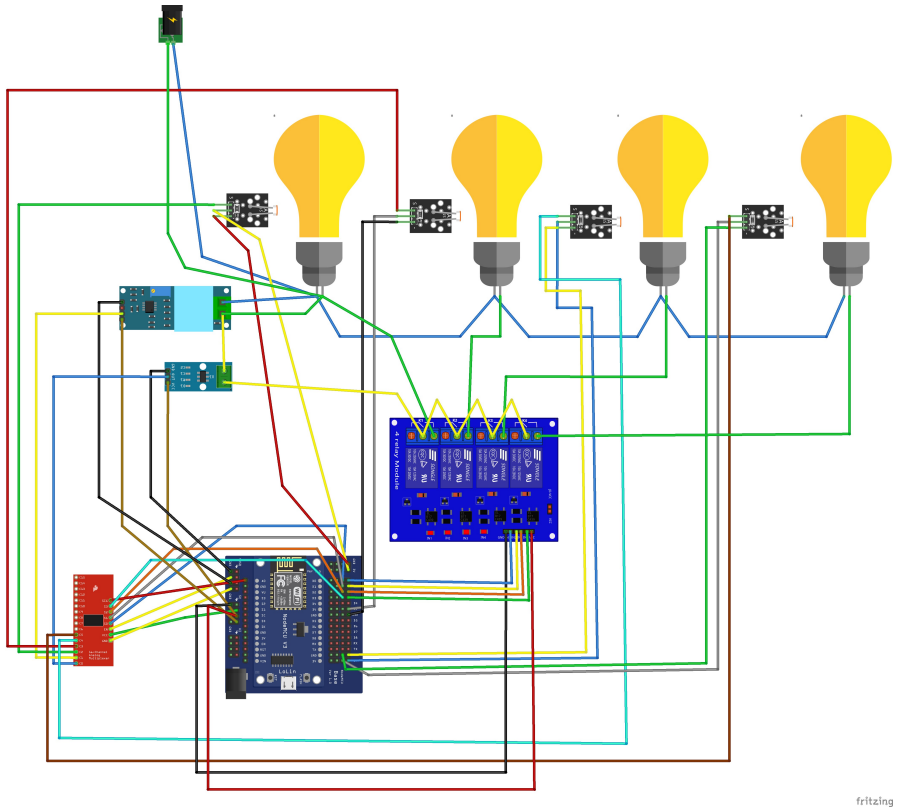
Pada diagram ini menunjukan proses *user* melakukan perintah di aplikasi android untuk mengecek status info lampu kemudian *mikrokontroler* memproses perintah tersebut dan mengirim status info lampu ke aplikasi android.



Gambar 4.8 *Sequence Diagram* Cek Lampu Aplikasi

4.5 Perancangan Desain *Mikrokontroler*

Perancangan desain *mikrokontroler* berupa desain sistem alat yang dibuat pada penelitian ini, tahap ini menentukan agar sistem tersebut bisa bekerja sesuai dengan yang dibutuhkan. Dan tahap ini menyangkut konfigurasi dari komponen yang digunakan dalam sistem tersebut, sehingga sistem yang telah dibuat dapat dimanfaatkan sesuai fungsi sistem ini.



Gambar 4.9 Desain Rangkaian *Mikrokontroler*

Gambar diatas merupakan rangkaian perangkat sistem *mikrokontroler NodeMCU* yang sudah ditambahkan *baseboard NodeMCU* untuk menambah pin-pin yang sedikit dari *NodeMCU.* Dan pin *baseboard* ini terhubung dengan sensor cahaya, sensor arus *ACS712*, sensor tegangan *ZMPT101B,* modul *relay* 4 *channel* untuk mengontrol lampu dan 16-*channel Analog Multiplexer* untuk mengubah pin digital ke analog digunakan untuk sensor-sensor tersebut. Berikut pin yang terhubung ke *mikrokontroler NodeMCU*, antara lain :

4.5.1 16-*channel Analog Multiplexer*

Koneksi pin *16-channel Analog Multiplexer* dengan pin B*aseborad NodeMCU* terdapat pada tabel 4.1 dibawah.

Tabel 4.1 Koneksi pin 16-*channel Analog Multiplexer* dengan pin B*aseborad NodeMCU.*

16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
Z	A0
S0	D0
S1	D1
S2	D2
S3	D3
EN	GND
VCC	5V
GND	GND

4.5.2 Sensor Cahaya

Koneksi pin sensor cahaya dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* terdapat pada tabel 4.2 dibawah.

Tabel 4.2 Koneksi pin sensor cahaya dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer.*

Sensor cahaya (1)	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
A0	Y2	
GND		GND
VCC		3V
Sensor cahaya (2)	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
A0	Y3	
GND		GND
VCC		3V
Sensor cahaya (3)	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
A0	Y4	
GND		GND

VCC		3V
Sensor cahaya (4)	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
A0	Y5	
GND		GND
VCC		3V

4.5.3 Sensor Arus *ACS712*

Koneksi pin sensor *ACS712* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* terdapat pada tabel 4.3 dibawah.

Tabel 4.3 koneksi pin sensor *ACS712* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer.*

Sensor *ACS712*	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
OUT	Y1	
GND		GND
VCC		5V

4.5.4 Sensor Tegangan *ZMPT101B*

Koneksi pin sensor *ZMPT101B* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* terdapat pada tabel 4.4 dibawah.

Tabel 4.4 koneksi pin sensor *ZMPT101B* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer.*

Sensor *ZMPT101B*	16-*channel Analog Multiplexer*	*Baseboard NodeMCU*
OUT	Y0	
GND		GND
VCC		5V

4.5.5 Modul *Relay 4 Channel*

Koneksi pin *relay* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer* terdapat pada tabel 4.5 dibawah.

Tabel 4.5 koneksi pin *relay* dengan pin *Baseboard NodeMCU* dan 16-*channel Analog Multiplexer.*

Relay	*Baseboard NodeMCU*
GND	GND
IN 1	D4
IN 2	D5
IN 3	D6
IN 4	D7
VCC	5V

4.6 Perancangan Pengujian Sistem

Dalam tahap ini akan menguji dua sistem antara lain pada Aplikasi Android dan Aplikasi *Website.*

4.6.1 Perancangan Pengujian Aplikasi Android

Perancangan pengujian aplikasi android terdapat pada tabel 4.6 dibawah ini.

Tabel 4.6 Perancangan pengujian aplikasi android

Kelas Uji	Skenario Pengujian	Hasil yang diharapkan
Halaman awal	Menu utama	Menampilkan list tombol yang digunakan untuk menyalakan/mematikan lampu
	Tombol *on/off* sesuai nomer	Jika tekan *on* maka lampu menyala / *off* maka lampu mati
	Cek Lampu	Pindah halaman ke cek lampu dengan *delay* 10 detik
	Panduan	Menampilkan info panduan nama lampu sesuai nomer

Halaman Cek Lampu	Arus	Menampilkan info arus yang keluar
	Tegangan	Menampilkan info tegangan yang keluar
	Daya	Menampilkan info daya yang keluar
	Notifikasi	Menampilkan info lampu jika sudah menyala / mati dan jika *error* menampilkan info masalahnya

4.6.2 Perancangan Pengujian Aplikasi *Web*

Perancangan pengujian aplikasi *website* terdapat pada tabel 4.7 dibawah ini.

Tabel 4.7 Perancangan pengujian aplikasi *web*

Kelas Uji	Skenario Pengujian	Hasil yang diharapkan
Halaman Cek Lampu	Realtime	Menampilkan info lampu yang sedang berjalan secara *realtime*
	Info	Menampilkan keterangan nama lampu
	History	Menampilkan history data lampu tertentu sesuai tanggal
Halaman Data Laporan	Tambah Data	Menambahkan data *inventory* lampu
	Data Laporan	Laporan data *inventory* lampu sesuai tanggal
Halaman Tambah lampu	Tambah Lampu	Menambahkan lampu yang akan digunakan sesuai jumlah *relay* yang terdapat pada *mikrokontroler*
	Data Lampu	Menampilkan data lampu yang sedang digunakan

4.7 Perancangan Desain Android

Perancangan desain android untuk memudahkan pengguna mengontrol dan *monitoring* alat yang berjalan. Pada tahap ini agar pengguna dapat

mengenali antarmuka aplikasi android yang nantinya akan digunakan untuk menyalakan atau mematikan lampu. Berikut antarmuka aplikasi android pada penelitian ini, antara lain:

4.7.1 Desain *Layout* Tampilan Beranda

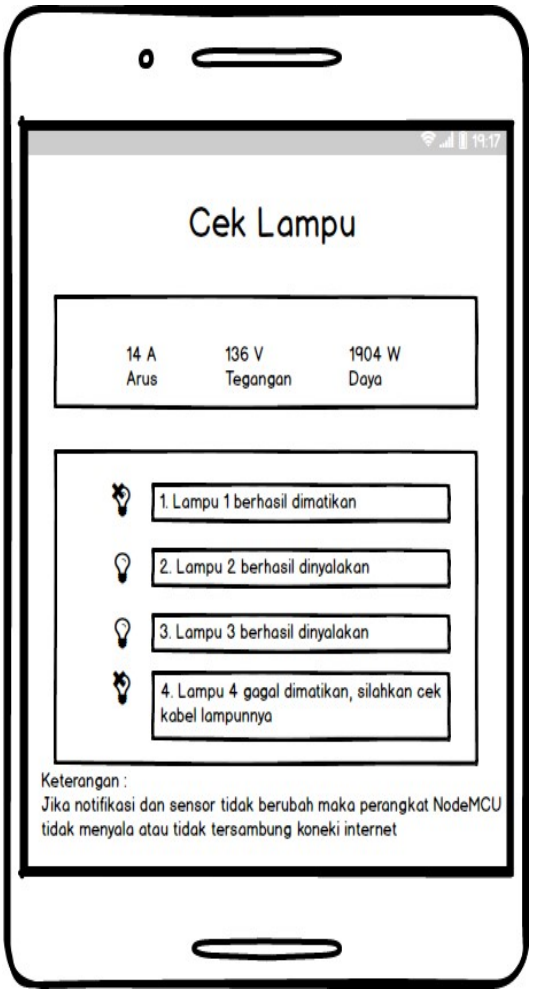
Tampilan beranda merupakan tampilan awal aplikasi berisi tombol *on/off* lampu yang digunakan untuk mengatur lampu nyala/mati.

Gambar 4.10 Tampilan Beranda Android

4.7.2 Desain *Layout* Tampilan Cek Lampu

Tampilan cek lampu merupakan notifikasi dari lampu tersebut yang telah berhasil dinyalakan atau mati dengan memunculkan pesan berhasil atau mati

beserta pesan *error*-nya. Dan juga menampilkan nilai dari sensor arus dan tegangan.



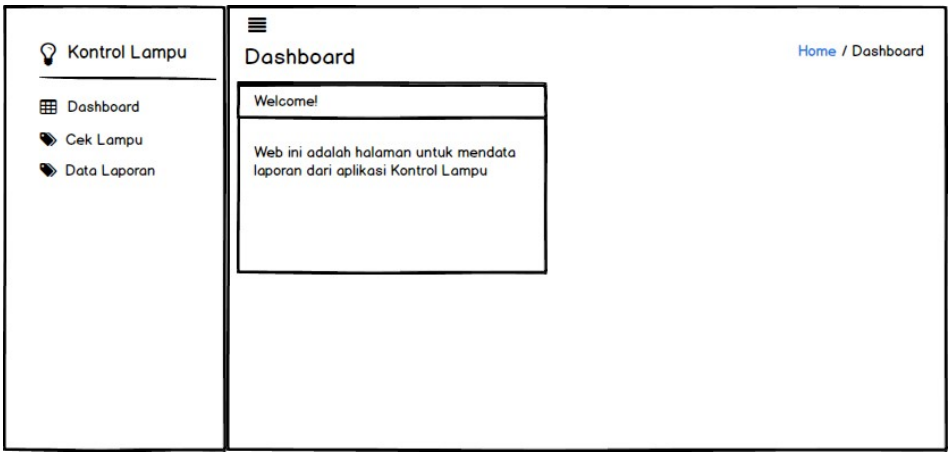
Gambar 4.11 Tampilan Cek Lampu Android

4.8 Perancangan Desain *Website*

Perancangan desain *website* ini digunakan untuk *backend* aplikasi android yang nantinya akan menyimpan data-data laporan nyala/mati lampu.

4.8.1 Desain *Layout* Tampilan Beranda *Web*

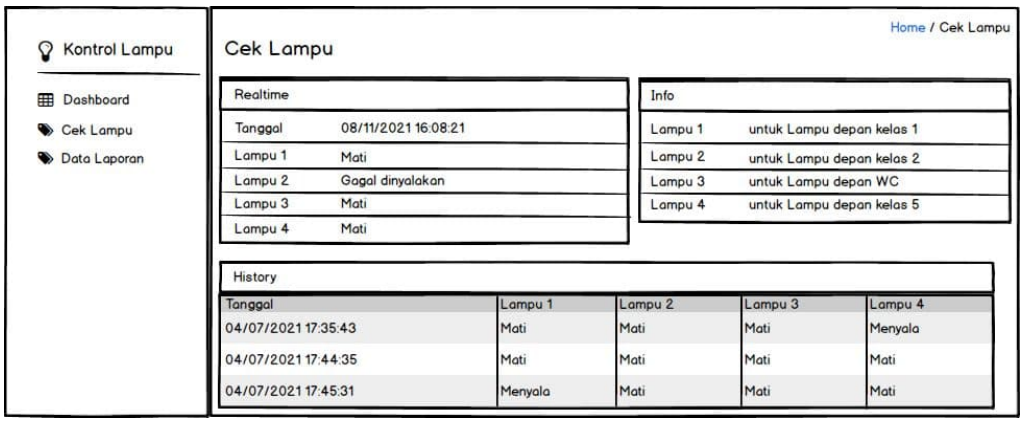
Tampilan beranda *web* menampilkan pesan selamat datang dan menu pilihan di *sidebar web*.



Gambar 4.12 Tampilan Beranda *Website*

4.8.2 Desain *Layout* Tampilan Cek Lampu *Web*

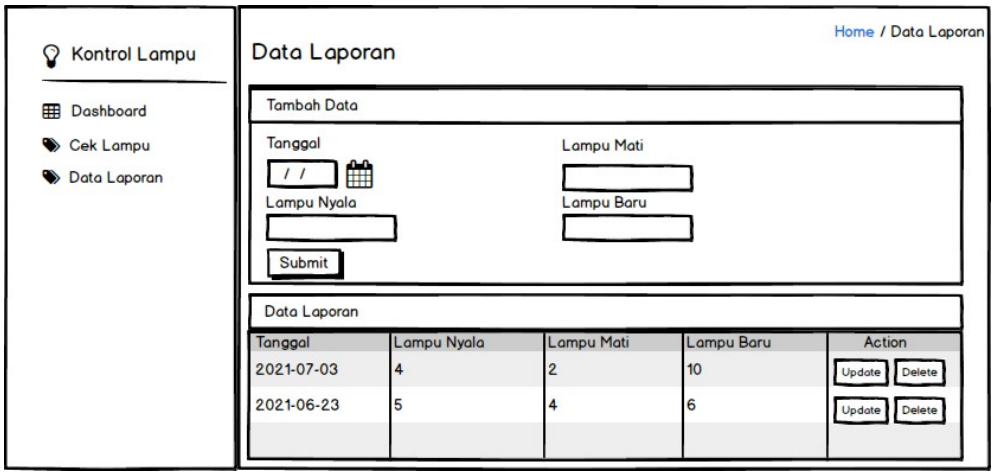
Tampilan cek lampu *web* menampilkan data laporan secara *realtime* dan *history* nyala/mati lampu dari aplikasi android yang telah disimpan di *Firebase.*



Gambar 4.13 Tampilan Cek Lampu *Website*

4.8.3 Desain *Layout* Tampilan Data Laporan *Web*

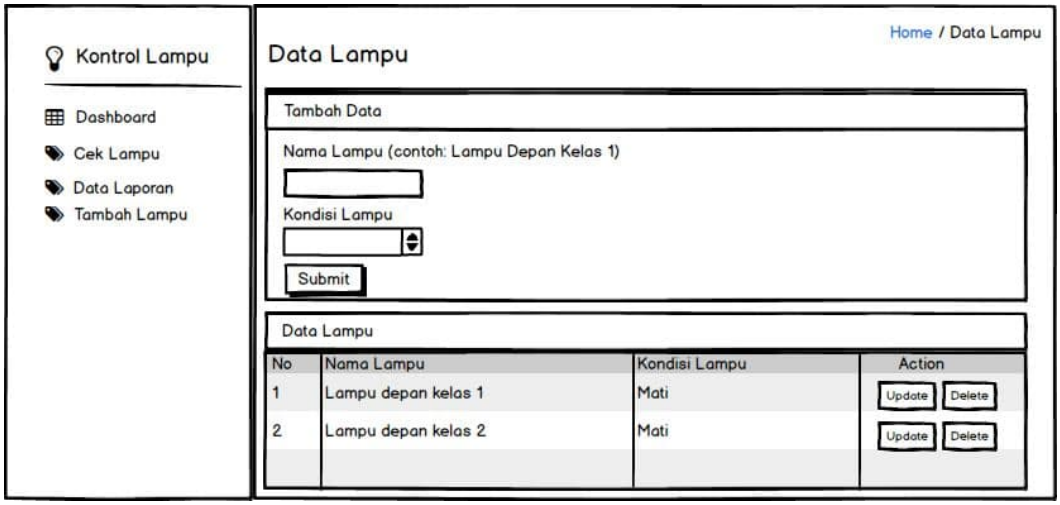
Tampilan data laporan *web* menampilkan data laporan lampu yang rusak, masih menyala, ataupun penggantian lampu baru.



Gambar 4.14 Tampilan Data Laporan *Website*

4.8.4 Desain *Layout* Tampilan Data Lampu *Web*

Tampilan data lampu *web* menampilkan data lampu yang ada pada *relay.* Pada menu ini untuk menambahkan lampu yang akan digunakan sesuai jumlah *relay* yang tersedia di *mikrokontroler.*



Gambar 4.15 Tampilan Data Lampu *Website*

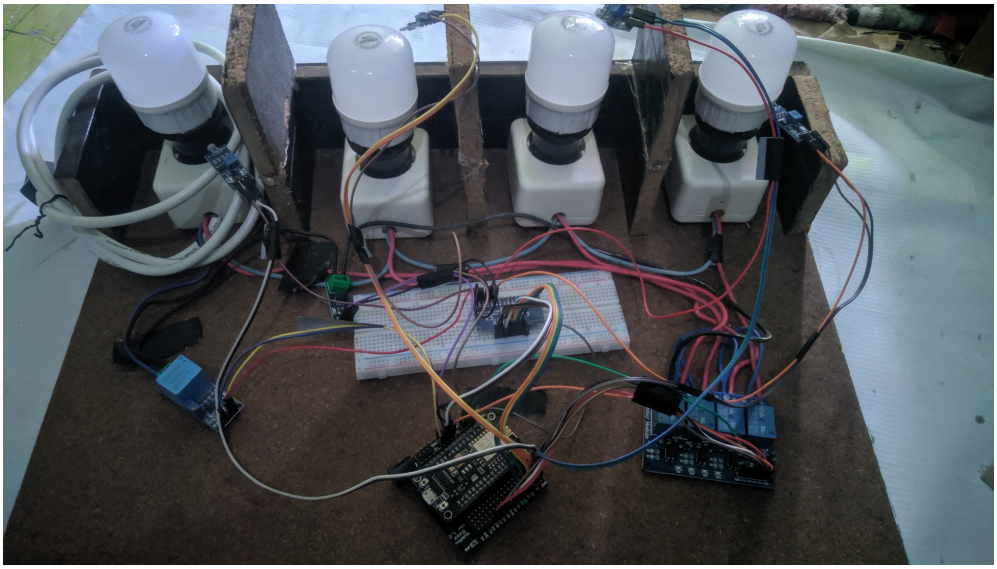
BAB V
HASIL DAN PEMBAHASAN PENELITIAN

5.1 Hasil Penelitian

Dari tahapan perancangan hingga pembuatan Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU* (Studi Kasus: SDN Kraton 3 Tegal) diperoleh hasil sebagai berikut.

5.1.1 Rangkaian *Prototype*

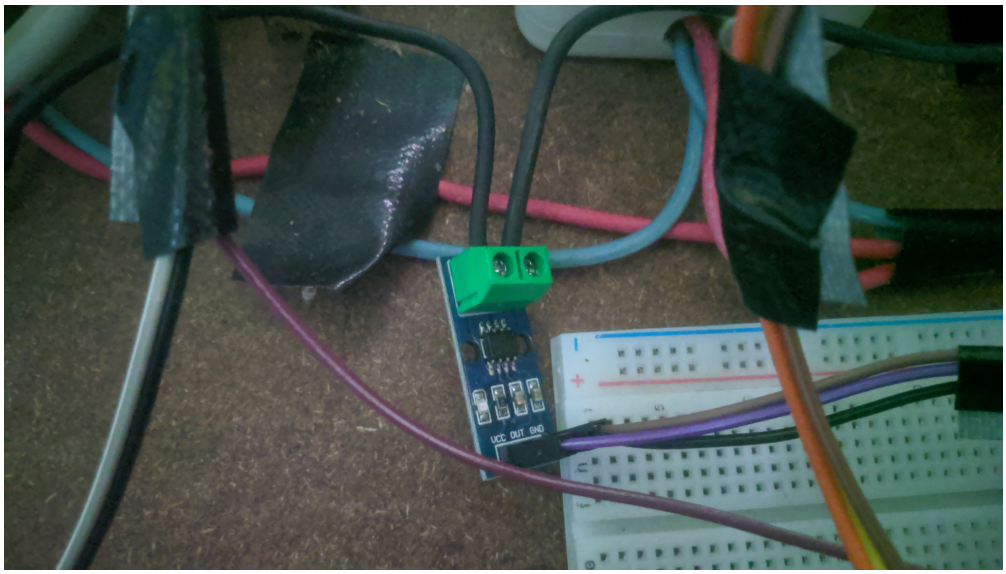
Rangkaian Rancang Bangun Aplikasi Android untuk Mengontrol Lampu Menggunakan *Mikrokontroler NodeMCU* yang telah dibuat menjadi *prototype* yang ada pada gambar 5.1 dibawah.



Gambar 5.1 Rangkaian *Prototype*

5.1.2 Rangkaian Pemasangan Sensor *ACS712*

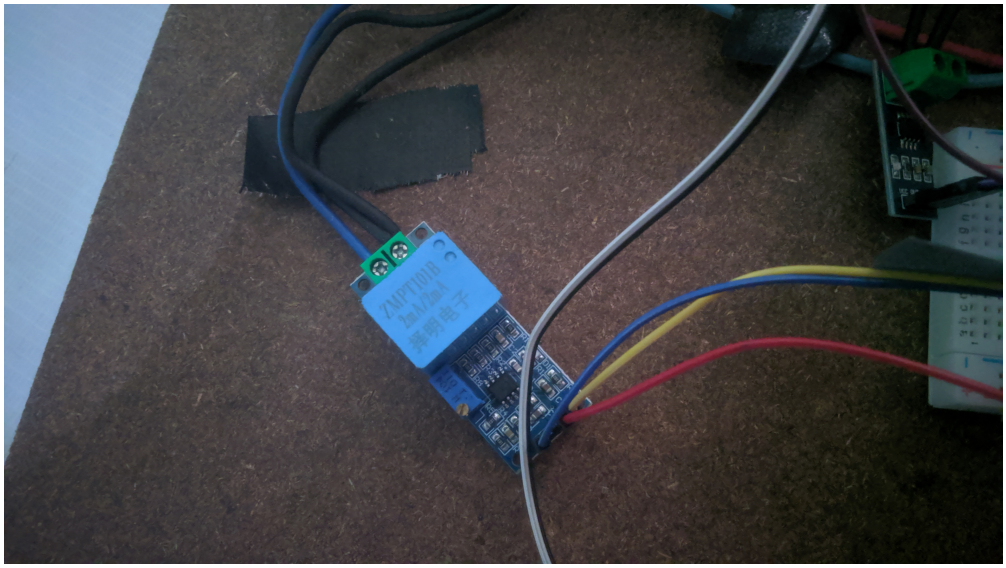
Dibawah ini adalah gambar peletakan sensor arus *ACS712* yang di sambungkan dengan salah satu kabel lampu agar dapat mendeteksi arus listrik.



Gambar 5.2 Rangkaian Pemasangan Sensor *ACS712*

5.1.3 Rangkaian Pemasangan Sensor *ZMPT101B*

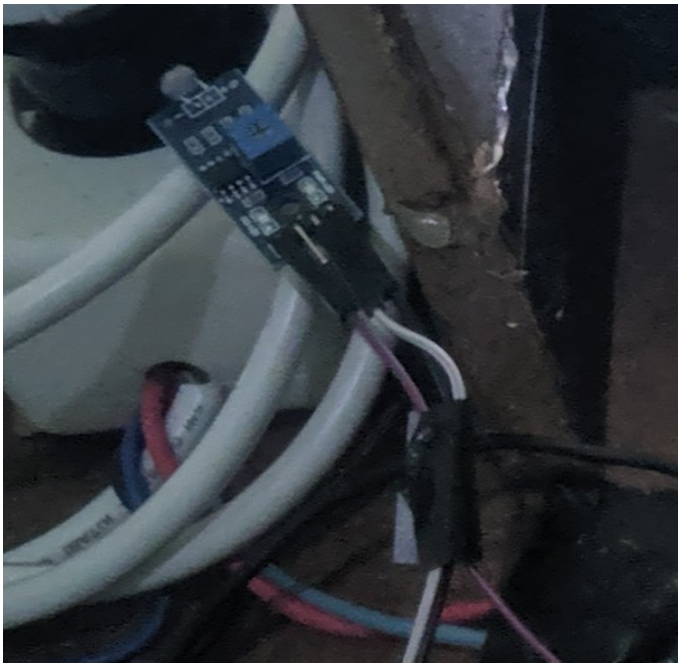
Dibawah ini adalah gambar peletakan sensor tegangan *ZMPT101B*, dari kabel lampu di *ACS712* disambungkan ke sensor ini dan kabel lampu plus (+) dan min (-) untuk di colokan ke sumber listrik.



Gambar 5.3 Rangkaian Pemasangan Sensor *ZMPT101B*

5.1.4 Rangkaian Pemasangan Sensor Cahaya/*LDR*

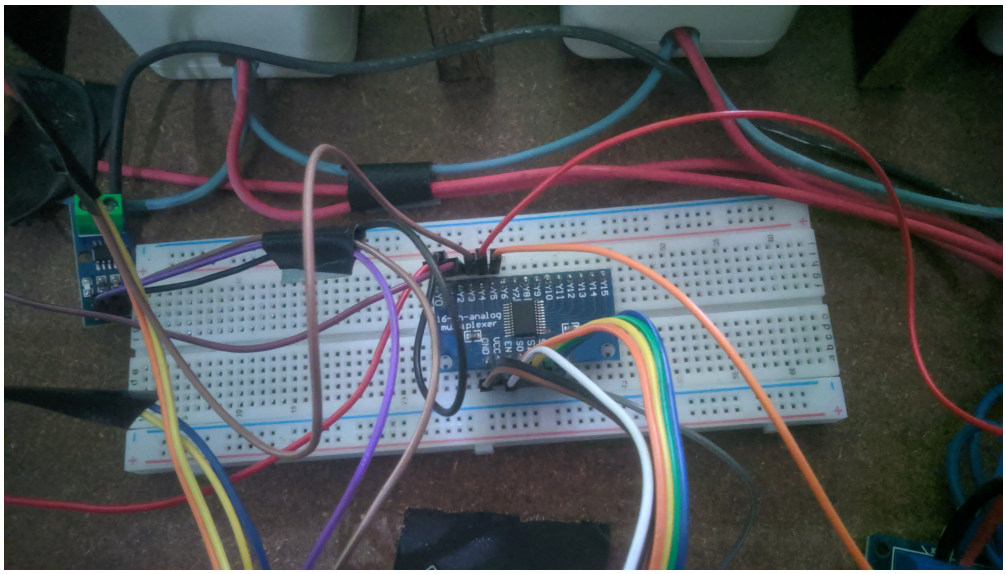
Dibawah ini adalah gambar peletakan sensor cahaya, menggunakan 4 sensor cahaya yang masing-masing didekatkan dengan setiap lampu untuk mendeteksi cahaya dari lampu.



Gambar 5.4 Rangkaian Pemasangan Sensor Cahaya/*LDR*

5.1.5 Rangkaian Kabel *Jumper*

Penggunaan kabel *jumper* sebagai penyambung antar komponen yang digunakan, dengan adanya kabel *jumper* memudahkan konektifitas antar komponen. Sehingga dapat memaksimalkan fungsi sistem, selain itu juga penggunaan kabel *jumper* tergolong mudah. Dan juga penggunaan 16-*Channel Analog Multiplexer* dengan ini sensor dapat membaca nilai *analog* dari sensor agar bisa dibaca *mikrokontroler*. Berikut ini gambar tentang rangkaian penggunaan kabel *jumper*.



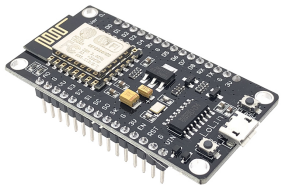
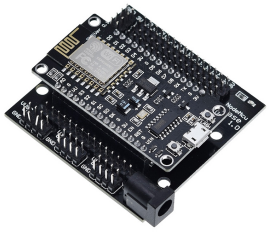
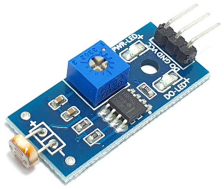
Gambar 5.5 Rangkaian Kabel *Jumper*

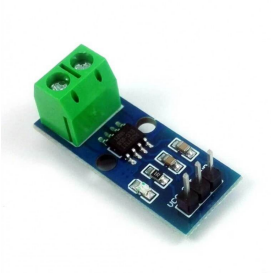
5.1.6 Pengujian Perangkat Keras

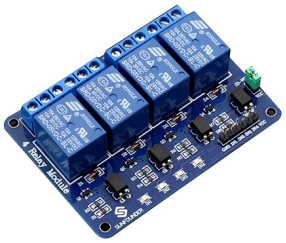
Dalam tahap pengujian perangkat keras, semua perangkat keras akan diuji terlebih dahulu. Dimana dalam pengujian ini akan diuji tingkat keberhasilan *input/output* berjalan sesuai yang diharapkan atau tidak. Hasil pengujian ini akan diterapkan menggunakan tabel, jika sistem diberi masukan kemudian hasil *output* sesuai yang di inginkan maka sistem telah lolos dari pengujian.

Tabel 5.1 *Test Case* Pengujian Perangkat Keras

No	Kondisi Awal	Proses	Keluaran	Hasil	
				Berhasil	Gagal
1	*NodeMCU ESP8266*	*NodeMCU ESP8266* akan dihubungkan dengan laptop	*Power* indikator pada *NodeMCU ESP8266* akan	Sesuai, karena *power* indikator *NodeMCU ESP8266*	*NodeMCU* gagal dinyalakan karena *power adaptor*

		dengan cara memasang kabel USB.	menyala jika sudah aktif	telah menyala.	tidak sesuai dengan yang dibutuhkan.
2	*Baseboard NodeMCU ESP8266* 	*Baseboard NodeMCU ESP8266* akan dihubungkan dengan *power supply* 12V.	*Power* indikator pada *Baseboard NodeMCU ESP8266* akan menyala jika sudah aktif	Sesuai, karena *power* indikator *Baseboard NodeMCU ESP8266* telah menyala.	*Baseboard NodeMCU* gagal dinyalakan karena *power adaptor* tidak sesuai dengan yang dibutuhkan.
3	Sensor *LDR*/Cahaya 	Sensor cahaya membutuhkan daya 3V untuk bisa aktif.	Saat sensor cahaya menerima daya yang sesuai maka indikator power pada sensor akan	Sesuai, karena saat pengujian sensor dihubungkan dengan pin daya yang sesuai	Gagal dihubungkan ketika tidak mendapat *power* yang cukup

			menyala dan keadaan sensor menjadi *stand by* untuk bekerja.	di *Baseboard NodeMCU ESP8266.*	dari daya yang ditentukan yaitu 3V.
4	Sensor Arus *ACS712* 	Sensor *ACS712* membutuhkan daya 5V untuk bisa aktif.	Saat sensor *ACS712* menerima daya yang sesuai maka indikator power pada sensor akan aktif dan posisi sensor menjadi *stand by* untuk bekerja.	Sesuai, karena saat pengujian sensor dihubungkan dengan pin daya yang sesuai di *Baseboard NodeMCU ESP8266.*	Gagal dihubungkan ketika tidak mendapat *power* yang cukup dari daya yang ditentukan yaitu 5V.
5	Sensor Tegangan *ZMPT101B* 	Sensor *ZMPT101B* membutuhkan daya 5V untuk bisa aktif menyala.	Saat sensor *ZMPT101B* menerima daya yang sesuai maka indikator power pada	Sesuai, karena saat pengujian sensor dihubungkan dengan pin daya	Gagal dihubungkan ketika tidak mendapat *power* yang

			sensor akan aktif dan posisi sensor menjadi *stand by* untuk bekerja.	yang sesuai di *Baseboard NodeMCU ESP8266.*	cukup dari daya yang ditentukan yaitu 5V.
6	Lampu *LED* 	Lampu *LED* berjalan di tegangan 220-250V maka butuh daya tambahan dari arus listrik AC.	Lampu akan menyala jika mendapat *input*-an nilai yang sudah di atur oleh *relay.*	Sesuai, lampu menyala ketika ada *input* dari *relay*.	Gagal dinyalakan ketika rangkaian kabel ada yang salah dan tidak mendapatkan sumber listrik.
7	*Relay 4 Channel* 	*Relay* membutuhkan daya 5V dari *Baseboard NodeMCU ESP8266* dan akan dihubungkan	*Relay* berfungsi untuk menyambung dan memutuskan arus listrik yang digunakan	Sesuai, karena saat *relay* mendapatkan daya yang dibutuhkan dan terhubung dengan	Gagal dihubungkan ketika tidak mendapat *power* yang cukup dari daya

		dengan kabel lampu dan kabel lampu diteruskan ke sensor arus dan tegangan untuk mendapat nilai arus dan tegangan.	untuk mengaktifkan lampu.	lampu dapat berjalan dengan sesuai.	yang ditentukan yaitu 5V

5.1.7 Pengujian Sensor

Pada tahap ini, menguji ketiga sensor apakah sudah bekerja sesuai dengan yang diinginkan, dimana data yang dihasilkan oleh sensor sama dengan data yang akan dikirim ke *database Firebase.*

Tabel 5.2 *Test Case* Pengujian Sensor *LDR/*Cahaya

Nilai *Analog*	**Lampu 1** **ON**	**Lampu 2** **ON**	**Lampu 3** **ON**	**Lampu 4** **ON**	**Keterangan**
Dibawah 300	Menyala	Menyala	Menyala	Menyala	Berhasil dinyalakan
Diatas 300	Menyala/*Error* karena seharusnya lampu mati	Menyala/*Error* karena seharusnya lampu mati	Menyala/*Error* karena seharusnya lampu mati	Menyala/*Error* karena seharusnya lampu mati	Lampu dalam keadaan menyala tetapi sensor membaca nilai *analog* diatas 300 maka

					lampu rusak / ada kabel yang putus.
Nilai Analog	**Lampu 1**	**Lampu 2**	**Lampu 3**	**Lampu 4**	**Keterangan**
	OFF	**OFF**	**OFF**	**OFF**	
Dibawah 300	Mati/*Error* karena seharusnya lampu menyala	Mati/*Error* karena seharusnya lampu menyala	Mati/*Error* karena seharusnya lampu menyala	Mati/*Error* karena seharusnya lampu menyala	Lampu dalam keadaan mati tetapi sensor membaca nilai *analog* dibawah 300 maka kabel rangkaian lampu ada yang salah / mati lampunya.
Diatas 300	Mari	Mati	Mati	Mati	Berhasil dimatikan

Dari tabel diatas dapat disimpulkan bahwa jika sensor cahaya membaca nilai sensornya dibawah 300 dalam keadaan lampu *ON* maka lampu berhasil dinyalakan, sebaliknya jika sensor membaca diatas 300 dalam keadaan lampu *OFF* maka lampu berhasil dimatikan. Dan jika sensor membaca diatas 300 tetapi lampu sedang *ON* maka ada lampu yang rusak / ada kabel yang putus. Kemudian jika sensor membaca dibawah 300 tetapi lampu sedang *OFF* maka ada kabel rangkaian lampu yang salah / mati lampunya.

Tabel 5.3 *Test Case* Pengujian Sensor *ACS712* dan *ZMPT101B*

Lampu	Sensor arus *ACS712* (A)	Sensor tegangan *ZMPT101B* (V)	Daya (W)	*Error* (%)
1	4	56	224	15
2	5	60	240	20
3	8	100	800	40
4	5	55	220	10
ON ALL	16	180	2880	50

Dari tabel diatas dapat disimpulkan bahwa keakuratan sensor arus dan tegangan ini masih belum cukup, karena arus dan tegangan yang dibaca tidak hanya lampu saja melainkan besarnya arus dari *power supply* yang diperoleh dari *mikrokontroler*. Hasil dari daya (W) diperoleh dari hasil arus dikalikan dengan tegangan.

5.2 Pembahasan Penelitian

Pada penelitian ini telah dihasilkan sebuah alat yang bisa digunakan untuk mengontrol lampu dan bisa *monitoring* besarnya arus dan tegangan yang digunakan pada lampu tersebut. Dengan menggunakan aplikasi android diharapkan pengguna bisa dengan mudah menyalakan lampu tanpa harus menekan saklar. Adapun *backend website* untuk mudah melihat data laporan lampu yang telah tersimpan pada *firebase*. Melalui *website* ini pengguna bisa melihat informasi dari lampu secara *realtime* jika sudah menyala atau mati dan melihat *history* data tanggal saat menyalakan lampu atau mematikan lampu. Selain itu, *website* ini untuk menyimpan data laporan stok lampu jika ada kerusakan pada lampu sebagai cadangan lampu yang ada. Jika kemungkinan setiap bulannya ada pergantian lampu maka dapat simpan data laporannya di *website*. Dengan sensor cahaya/*LDR,* sensor arus *ACS712,* dan sensor tegangan

ZMPT101B sebagai deteksi lampu jika ada kesalahan ketika dinyalakan bisa cukup baik di gunakan karena pengguna bisa mendapatkan informasi jika ada lampu yang rusak. Dengan tingkat akurasi dari sensor arus dan tegangan yang lumayan digunakan untuk mengetahui perangkat *mikrokontroler* sudah berjalan dengan baik atau belum. Dengan menggunakan sensor arus *ACS712* dan tegangan *ZMPT101B* ini, bisa digunakan untuk menghitung kira-kira biaya listrik yang akan dikeluarkan setiap bulannya. Untuk menghitung biaya listrik, perlu mengetahui golongan tarif listrik yang digunakan. Diketahui pada SDN Kraton 3 kira-kira menggunakan golongan 2.200 *VA (Volt Ampere)*. Jika tarif dasar listrik untuk golongan 1.301 - 2.200 *VA* adalah Rp. 1.444,70 per *kWh (Kilowatt per Hour).* Langkah selanjutnya menghitung total daya listrik yang dikeluarkan, berdasarkan hasil pengujian sensor arus dan tegangan diperoleh total daya yang digunakan untuk menghidupkan 4 buah lampu dan *mikrokontroler* yaitu 2.880 *Watt.* Jika pemakaian per hari setidaknya 12 jam, maka estimasi penggunaan dayanya menjadi 2880 x 12 = 34.560 *Watt.* Selanjutnya diubah dulu ke *kWh* dengan membagi jumlah penggunaan daya dengan 1.000, atau 34.560 : 1.000 = 34,56 *kWh*. Kemudian dikalikan dengan tarif dasar listrik, 34,56 x 1.444,70 hasilnya Rp. 49.929,00. Untuk menghitung biaya listrik dalam sebulan maka dikalikan dengan 30 yaitu 49.929 x 30 = 1.497.870 Rupiah. Maka dalam sebulan kira-kira biaya listrik yang dikeluarkan Rp. 1.497.870,00. Penggunaan sensor arus dan tegangan ini cukup mudah dan berguna untuk menghitung kira-kira biaya listrik yang harus dikeluarkan per bulannya.

Berdasarkan hasil "*Test Case*" pengujian sensor cahaya dapat disimpulkan bahwa sensor cahaya akan menampilkan notifikasi berhasil dinyalakan pada aplikasi android jika nilai sensornya dibawah 300, sebaliknya sensor cahaya akan menampilkan notifikasi berhasil dimatikan jika nilai sensor diatas 300. Dan sensor cahaya akan menampilkan notifikasi lampu yang mati / rusak / ada kabel yang putus jika nilai sensor diatas 300 tetapi kondisi lampu

sedang *ON*. Kemudian sensor cahaya akan menampilkan notifikasi *error* kabel rangkaian lampu yang salah / mati lampunya jika nilai sensor dibawah 300 tetapi kondisi lampu sedang *OFF.* Setelah sensor cahaya, pengujian keakuratan sensor arus dan tegangan ini masih belum cukup, karena arus dan tegangan yang dibaca tidak hanya lampu saja melainkan besarnya arus dari *power supply* yang diperoleh dari *mikrokontroler*. Dan tingkat *error*-nya terbilang cukup tinggi, maka dari itu dibutuhkan sensor arus dan tegangan yang lebih bagus lagi tingkat keakuratannya.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil penelitian dan pengembangan yang dilakukan, maka disimpulkan bahwa :

1. Telah dibuat aplikasi kontrol lampu untuk mengontrol lampu dengan mudah menggunakan aplikasi android untuk *turn ON/OFF*. Menggunakan *server database* dari *Firebase* yang dengan mudah di integrasikan ke 3 *service* antara lain, menyimpan *input*-an dari *relay* ataupun sensor-sensor yang dikendalikan oleh *mikrokontroler NodeMCU* ke *firebase*, aplikasi android yang terhubung *firebase* untuk mengontrol lampunya.
2. Berdasarkan pengujian "*Test Case*", penelitian ini menggunakan 3 sensor yang digunakan yaitu, sensor cahaya/*LDR,* sensor arus *ACS712,* dan sensor tegangan *ZMPT101B.* Hasil yang diperoleh dari sensor cahaya sudah baik karena bisa digunakan untuk mendeteksi nyala/mati dari lampu dengan baik, sensor arus yang dihasilkan juga cukup baik digunakan untuk mendeteksi arus yang masuk dari sumber listrik dan sensor tegangan yang memperoleh hasil yang cukup baik juga untuk mendeteksi tegangan yang masuk dari sumber listrik walaupun kedua sensor arus dan tegangan ini tingkat keakuratannya kurang tetapi bisa digunakan dengan baik sesuai penggunaannya.
3. *Website* digunakan untuk mengecek kondisi lampu dalam urutan tanggal, menyimpan data laporan penggunaan lampu, dan mengatur jumlah lampu yang akan digunakan sesuai jumlah *relay* yang terdapat pada *mikrokontroler.*

6.2 Saran

Berdasarkan pembahasan hasil penelitian, maka beberapa dapat di ajukan sebagai berikut :

1. Pada penelitian ini masih banyak kekurangan karena tingkat ke akuratan sensor masih ada yang belum cukup dan karena itu diperlukan pengembangan lagi agar bisa lebih akurat lagi dengan mengganti sensor yang lebih akurat atau dengan mengganti *mikrokontroler* yang mampu menjalankan beberapa sensor dengan cepat tanpa harus menunggu *delay* yang seperti ada di *NodeMCU.*
2. Pada aplikasi android masih banyak kekurangan pada fitur oleh karena itu untuk tahapan produksi massal di butuhkan pengembangan lagi agar lebih banyak memiliki fitur yang dimiliki.
3. Hasil penelitian ini masih berupa *prototype*, diharapkan dapat di integrasikan ke lapangan agar bisa mendapatkan manfaatnya secara langsung bagi pengguna.

DAFTAR PUSTAKA

[1] A. Hanani and M. A. Hariyadi, "Smart Home Berbasis IoT Menggunakan Suara Pada Google Assistant," *J. Ilm. Teknol. Inf. Asia*, vol. 14, no. 1, p. 49, 2020, doi: 10.32815/jitika.v14i1.456.

[2] M. Kashyap, V. Sharma, and N. Gupta, "Taking MQTT and NodeMcu to IOT: Communication in Internet of Things," *Procedia Comput. Sci.*, vol. 132, no. Iccids, pp. 1611–1618, 2018, doi: 10.1016/j.procs.2018.05.126.

[3] A. D. Pangestu, F. Ardianto, and B. Alfaresi, "Sistem Monitoring Beban Listrik Berbasis Arduino Nodemcu Esp8266," *J. Ampere*, vol. 4, no. 1, p. 187, 2019, doi: 10.31851/ampere.v4i1.2745.

[4] Pratiwi, "Rancang Bangun Sistem Telecontrolling Pada Ruangan Oven Batang Rokok Berbasis Android Secara Realtime (Studi Kasus di Pabrik Indokretek) Program Studi Jaringan Telekomunikasi Digital , Teknik Elektro ," *Pratiwi*, pp. 8–13, 2018.

[5] A. Purwanto and S. Lutfi, "Pengendalian Lampu Rumah Berbasis Google Asisstant Melalui Smartphone Menggunakan NodeMCU-12E ESP8266 di Nuke Komputer Service," *J. Himsya Tech*, vol. 20, no. 2, pp. 1–6, 2019.

[6] M. I. Reza, S. R. Akbar, and H. Fitriyah, "Kontrol Lampu Berbasis Voice Command Pada Raspberry PI," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 9, pp. 3124–3133, 2018.

[7] R. Rizky, Z. Hakim, A. M. Yunita, and N. N. Wardah, "Implementasi Teknologi Iot (Internet Of Think) Pada Rumah Pintar Berbasis Mikrokontroler ESP 8266," vol. 4, no. 2, pp. 278–281, 2020.

[8] H. A. Rochman, R. Primananda, and H. Nurwasito, "Sistem Kendali Berbasis Mikrokontroler Menggunakan Protokol MQTT pada Smarthome," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 1, no. 6, pp. 445–455, 2017, [Online]. Available: http://j-ptiik.ub.ac.id.

[9] T. F. Siallagan, “Rancang Bangun Sistem Peringatan Dini Terhadap Kebakaran Berbasis Bot Telegram Menggunakan Pendahuluan Kajian Pustaka,” vol. VI, no. 1, pp. 61–70, 2019.

[10] S. Uma, R. Eswari, R. Bhuvanya, and G. S. Kumar, “IoT based Voice/Text Controlled Home Appliances,” *Procedia Comput. Sci.*, vol. 165, no. 2019, pp. 232–238, 2019, doi: 10.1016/j.procs.2020.01.085.

[11] M. S. Novelan and M. Amin, “Monitoring System for Temperature and Humidity Measurement with DHT11 Sensor Using NodeMCU,” *Int. J. Innov. Sci. Res. Technol.*, vol. 5, no. 10, pp. 123–128, 2020.

[12] P. H. Weng, A. B. M. W, and G. C. Setyawan, “Membangun Sistem Kendali Jarak Jauh Pada Ruang Kelas Menggunakan Thingspeak Berbasis Web,” vol. 1, no. 1, pp. 1–12, 2020.

[13] B. Artono and R. G. Putra, “Penerapan Internet Of Things (IoT) Untuk Kontrol Lampu Menggunakan Arduino Berbasis Web,” *J. Teknol. Inf. dan Terap.*, vol. 5, no. 1, pp. 9–16, 2019, doi: 10.25047/jtit.v5i1.73.1, no. 1, pp. 1–12, 2020.

Lampiran 1. Surat Kesepakatan Bimbingan Tugas Akhir

SURAT KESEPAKATAN BIMBINGAN TUGAS AKHIR

Kami yang bertanda tangan di bawah ini:

Pihak Pertama

Nama	: Evan Fauzi Pranendra
NIM	: 17090041
Program Studi	: D IV Teknik Informatika

Pihak Kedua

Nama	: Dega Surono Wibowo, S.T., M.Kom
Status	: Dosen Tetap
NIDN	: 0607108202
Jabatan Fungsional	: Lektor
Pangkat/Golongan	: Penata Muda Tk. I / III/b

Pada hari ini Senin tanggal 11 Januari 2021 telah terjadi kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing I untuk Tugas Akhir Pihak Pertama. Pihak Pertama wajib melakukan bimbingan Tugas Akhir **sekurang-kurangnya 1 (satu) kali dalam 1 (satu) minggu**, adapun waktu dan tempat pelaksanaan disepakati antar pihak.

Tegal, 11 Januari 2021

Pihak Pertama	Pihak Kedua
Evan Fauzi Pranendra	Dega Surono Wibowo, S.T., M.Kom

Mengetahui,
Ketua Program Studi Teknik Informatika

Slamet Wiyono, S.Pd., M.Eng
NIPY. 08.015.222

SURAT KESEPAKATAN BIMBINGAN TUGAS AKHIR

Kami yang bertanda tangan di bawah ini:

Pihak Pertama

Nama	: Evan Fauzi Pranendra
NIM	: 17090041
Program Studi	: D IV Teknik Informatika

Pihak Kedua

Nama	: Rosid Mustofa, M.Kom
Status	: Dosen
NIDN	: -
Jabatan Fungsional	: -
Pangkat/Golongan	: -

Pada hari ini Senin tanggal 11 Januari 2021 telah terjadi kesepakatan bahwa Pihak Kedua bersedia menjadi Pembimbing II untuk Tugas Akhir Pihak Pertama. Pihak Pertama wajib melakukan bimbingan Tugas Akhir **sekurang-kurangnya 1 (satu) kali dalam 1 (satu) minggu**, adapun waktu dan tempat pelaksanaan disepakati antar pihak.

Tegal, 11 Januari 2021

Pihak Pertama	Pihak Kedua
Evan Fauzi Pranendra	Rosid Mustofa, M.Kom

Mengetahui,
Ketua Program Studi Teknik Informatika

Slamet Wiyono, S.Pd., M.Eng
NIPY. 08.015.222

Lampiran 2. Lembar Bimbingan Tugas Akhir

IK	P2	PH	04.06.G.7.b.3

D IV TEKNIK INFORMATIKA POLITEKNIK HARAPAN BERSAMA

LEMBAR BIMBINGAN TUGAS AKHIR

Nama : **Evan Fauzi Pranendra**
NIM : **17090041**
No. Ponsel : **083840007258**
Judul TA : **RANCANG BANGUN APLIKASI ANDROID UNTUK MENGONTROL LAMPU MENGGUNAKAN MIKROKONTROLER NODEMCU (STUDI KASUS : SDN KRATON 3 TEGAL)**

Dosen Pembimbing I : **Dega Surono Wibowo, S.T., M.Kom**

No	Tanggal	Pemeriksaan	Perbaikan Yang Perlu Dilakukan	Paraf Pembimbing
1.	20/1 21	① persiapkan alat ② Buat Diagram Blok Rangkaian ③ Buat Rancang prototipe alatnya ④ persiapkan UML.	Lanjutkan lagi Ukuran lampunya sensor apinya Use case activity sequence	
2.	10/2 21	Silahkan dibrat		

3	24/3 21	• Baca Arus • Baca Tegangan • Nyala lampu	Perbaikan reboot NodeMCU	
4.	31/5 21	• Lampu nyala semua • Masih reboot sendiri	Perbaikan kendala sering reboot	
5.	8/6 21	• Masalah reboot selesai	Ditambahkan notic atau pemberi-tahuan lampu mati	
6.	30/6 21	• pemberitahuan lampu mati selesai	• buat backend laporan lampu mati/nyala • membuat laporan bab 1-3	
7.	12/7 21	• Laporan Bab 1-3	Revisi bab 1-3	
8.	16/7 21	• Rev. Bab 1-3	Lanjut bab 4-6	
9.	19/7 21	• Laporan bab 4-6	Revisi bab 5	
10.	20/7 21	• Rev. bab 5	Acc	

Tegal, 21 Juli 2021
Dosen Pembimbing I

Dega Suronto Wibowo, S.T., M.Kom
NIPY. 06.014.18[illegible]

IK	P2	PH	04.06.G.7.b.3

D IV TEKNIK INFORMATIKA POLITEKNIK HARAPAN BERSAMA

LEMBAR BIMBINGAN TUGAS AKHIR

Nama : **Evan Fauzi Pranendra**
NIM : **17090041**
No. Ponsel : **083840007258**
Judul TA : **RANCANG BANGUN APLIKASI ANDROID UNTUK MENGONTROL LAMPU MENGGUNAKAN MIKROKONTROLER NODEMCU (STUDI KASUS : SDN KRATON 3 TEGAL)**
Dosen Pembimbing II : **Rosid Mustafa, M.Kom**

No	Tanggal	Pemeriksaan	Perbaikan Yang Perlu Dilakukan	Paraf Pembimbing
1	4/2 2021	konsultasi	ditingkatkan dahulu dan matangkan.	
2	19/5 2021	Desain	biar bagus buat dlu di figma nanti di konvert ke android	
3	27/5 2021	Desain	tambahkan keterangan lampu	
4	03/06 2021	Desain	informasi posisi lampu di pusatkan.	
5	21/06 2021	Desain	buatkan Backend nya yah	
6	28/6 2021	Backend	buka website yah	
7	20/7 2021	Laporan	Bab 1 - 6 lanjut Ati diselesaikan yah.	

8	21/7 2021	TKT Aplikasi	Daftar sidang Yah	

Tegal, 19 Juli 2021
Dosen Pembimbing II

Rosid Mustafa, M.Kom