repository.ub.ac.id

KONTROL LAMPU BERBASIS *VOICE COMMAND* PADA RASPBERRY PI

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Muhammad Irfan Reza
NIM. 145150300111121

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

UNIVERSITAS BRAWIJAYA

PENGESAHAN

KONTROL LAMPU BERBASIS *VOICE COMMAND* PADA RASPBERRY PI

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun Oleh :
Muhammad Irfan Reza
NIM: 145150300111121

Skripsi ini telah diuji dan dinyatakan lulus pada
19 Januari 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T, M.Eng	Hurriyatul Fitriyah, S.T, M.Sc
NIP. 19820809 201212 1 004	NIP. 19851001 201504 2 003

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D
NIP: 19710518 200312 1 001

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 19 Januari 2018

Muhammad Irfan Reza

NIM: 145150300111121

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

KATA PENGANTAR

Puji syukur kehadirat Tuhan YME yang telah melimpahkan rahmat, taufik serta hidayah-Nya sehingga laporan skripsi yang berjudul "KONTROL LAMPU BERBASIS *VOICE COMMAND* PADA RASPBERRY PI" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. dan Ibu Hurriyatul Fitriyah, S.T, M.Sc. selaku dosen pembimbing skripsi penulis yang dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika.
3. Kedua orangtua dan seluruh keluarga besar atas segala nasehat, kasih sayang dan kesabaran dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
4. Teman-teman komunitas Robotiik yang telah memberikan banyak pengalaman dan ilmu sehingga penulis dapat menyelesaikan skripsi ini.
5. Teman-teman Teknik Komputer atas dukungan dan semangatnya sehingga penulis dapat menyelesaikan skripsi ini.
6. Seluruh civitas akademika Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi Informatika di Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 19 Januari 2018

Penulis

muhammad.irfan.reza@gmail.com

repository.ub.ac.id

ABSTRAK

Smart home merupakan sebuah rumah yang menyediakan kenyamanan, keamanan dan kemudahan kepada penghuninya dalam pengoperasian setiap peralatan di setiap waktunya. *Smart home* biasanya terdiri dari beberapa peralatan elektronik seperti lampu, kipas, *air-conditioner* dsb. Salah satu metode pengontrolan *smart home* yang sedang berkembang saat ini adalah pengontrolan menggunakan suara. Saat ini terdapat terdapat beberapa teknologi *voice control* yang sedang dikembangkan salah satunya adalah *voice control system* yang dikembangkan oleh Amazon yaitu, Alexa. Pada penelitian ini dilakukan pengimplementasian *Alexa Voice Service* pada Raspberry Pi supaya perangkat tersebut dapat melakukan kontrol pada lampu yaitu mematikan, menyalakan, mengubah warna, mengatur intensitas dan melakukan penjadwalan.

Perintah suara yang diterima pada perangkat Raspberry akan dikirimkan ke *Alexa Skills Kit* melalui *Alexa Voice Service* yang sudah terpasang di Raspberry, selanjutnya perintah suara yang sudah diterima di *Alexa Skills Kit* akan dikirim ke *Amazon Web Service Lamba* untuk di modelkan dengan format JSON sehingga dapat dikenali oleh sistem. Lalu data yang sudah diformat tersebut akan di *publish* ke MQTT Broker. Sistem akan men-*subscribe* data tersebut sebagai acuan tindakan apa yang akan dilakukan. Tingkat akurasi dari keberhasilan sistem dalam menerjemahkan perintah suara mematikan dan menyalakan lampu adalah 95,8%, menguah warna lampu 100%, mengatur intensitas 93,3%, melakukan penjadwalan 100%.

Kata kunci : *smart home*, *voice command*, *speech recognition*, *alexa*

UNIVERSITAS BRAWIJAYA

repository.ub.ac.id

ABSTRACT

Smart home is a home that provides comfort, safety and convenience to its inhabitants in the operation of every device in every time. Smart home usually consists of some electronic device such as lamps, fans, air-conditioner, etc. One of the smart home control methods currently developing is voice control. Currently there are some voice control technology that is being developed one of them is the voice control system developed by Amazon. In this research will be implementing Alexa Voice Service on Raspberry so that the device can control the lamp to turn off, turn on, change color, set intensity and do scheduling.

Voice commands received on the raspberry will be sent to alexa skills kit via alexa voice service that is installed in Raspberry, then voice command that has been received in alexa skills kit will be sent to amazon web service lamba to modeled with JSON format so it can be recognized by system. The formatted data will be published to MQTT Broker. System will subscribe the data as a reference action what will to do. The accuracy of the system's success in translating voice commands to turn off and turn on is 95.8%, to change color is 100%, to set the intensity is 93.3%, to do scheduling is 100%.

Keywords : smart home, voice command, speech recognition, alexa

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Lampu merupakan sumber penerangan yang memanfaatkan energi listrik. Tarif dari energi listrik di dunia terutama di Indonesia semakin naik dari tahun ke tahunnya. Hal ini menyebabkan warga harus melakukan penghematan energi listrik guna menghemat pengeluaran dana untuk pembayaran tarif energi listrik. Masalah yang saat ini sering terjadi adalah sistem kontrol manual khususnya untuk menghidupkan dan mematikan lampu dari saklar. Terkadang, warga lupa atau malas untuk mematikan lampu dikarenakan jauhnya posisi saklar.

Pada saat ini perkembanan teknologi sangatlah pesat. Salah satu contohnya adalah teknologi *Smart home* atau biasa disebut Rumah Pintar. *Smart home* merupakan sebuah rumah yang menyediakan kenyamanan dan kemudahan kepada penghuninya dalam pengontrolan setiap peralatan di setiap waktunya (Tri Fajar, dkk., 2009). Salah satu cara pengontrolan *smart home* yang sedang berkembang saat ini adalah pengontrolan menggunakan suara (Kjetil, 2017). Pengontrolan dengan suara memberikan kemudahan dan kenyamanan pada penggunanya(Vincent , dkk., 2006).

Saat ini terdapat beberapa teknologi *voice service* yang sedang dikembangkan oleh beberapa perusahaan besar salah satunya *voice service* dari Amazon yaitu, Alexa (Kjetil, 2017). Salah satu fitur dari Alexa yang berbeda dari teknologi *voice service* lainnya adalah *Alexa Skills Kit*. Fitur tersebut memudahkan pengembang untuk merancang dan memanajemen sebuah kemampuan interaksi dari perintah suara yang diinginkan.

Penelitian ini bertujuan untuk mengimplementasikan Alexa pada Raspberry Pi Sehingga perangkat tersebut dapat memiliki kemampuan sebagai *modul speech recognition*. Raspberry Pi dipilih karena dapat diimplementasikan *Alexa Voice Service*, selain itu pada perangkat Raspberry Pi terdapat pin GPIO yang dapat dimanipulasi untuk kebutuhan sistem ini. Nantinya Raspberry Pi akan dijadikan pusat pengontrolan lampu menggunakan perintah suara. Fungsi kontrol yang dapat dilakukan diantaranya mematikan, menyalakan, mengatur intensitas, mengubah warna dan melakukan penjadwalan pada lampu.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dikemukakan, rumusan masalah pada penelitian ini antara lain:

1. Bagaimana cara mengimplementasikan *Alexa Voice Service* pada perangkat Raspberry Pi ?
2. Bagaimana cara merancang *Alexa Skills Kit* supaya perintah suara yang dikirim dari Raspberry Pi melalui *Alexa Voice Service* dapat diterjemahkan ?

3. Bagaimana cara merancang kode program pada *Amazon Web Service Lambda* supaya dapat memodelkan hasil terjemahan dari *Alexa Skills Kit* sehingga sesuai dengan format yang dibutuhkan Raspberry Pi ?
4. Bagaimana perangkat Raspberry Pi dapat menerima hasil terjemahan perintah suara yang sudah di modelkan di *Amazon Web Service Lambda* melalui protokol pengiriman MQTT ?
5. Bagaimana akurasi sistem dalam menerjemahkan perintah suara ?

1.3 Tujuan

Adapun maksud dan tujuan dari penelitian ini adalah sebagai berikut :

1. Dapat mengimplementasikan *Alexa Voice Service* pada perangkat Raspberry Pi.
2. Dapat merancang *Alexa Skills Kit* supaya perintah suara yang dikirim dari Raspberry Pi melalui *Alexa Voice Service* dapat diterjemahkan.
3. Dapat merancang kode program untuk *Amazon Web Service Lambda* supaya dapat memodelkan hasil terjemahan dari *Alexa Skills Kit* sehingga sesuai dengan format yang dibutuhkan Raspberry Pi.
4. Perangkat Raspberry Pi dapat menerima hasil terjemahan perintah suara yang sudah di modelkan di *Amazon Web Service Lambda* melalui protokol pengiriman MQTT

1.4 Manfaat

Memberikan gambaran implementasi Alexa pada Raspberry Pi sehingga dapat memudahkan setiap orang dalam melakukan kontrol *device* pada *smart home* berbasis perintah suara khususnya mengontrol lampu.

1.5 Batasan masalah

Agar pembahasan dalam penelitian ini dapat dilakukan secara terarah dan mendapatkan hasil sesuai dengan yang diharapkan, maka perlu diterapkan batasan permasalahan yaitu implementasi Alexa pada Raspberry Pi untuk kontrol terhadap lampu. Batasan-batasan permasalahan lain, antara lain :

1. Implementasi memanfaatkan mini komputer Raspberry Pi.
2. Impementasi memanfaatkan *microphone* untuk media pemberian perintah suara.
3. Implementasi pada LED RGB fungsi kontrol yang dapat dilakukan adalah menyalakan, mematikan, pengaturan intensitas dan melakukan penjadwalan
4. Implementasi pada lampu AC fungsi kontrol yang dapat dilakukan adalah mematikan, menyalakan dan melakukan penjadwalan.
5. Terdapat 4 lokasi lampu yang dapat diatur yaitu *garden*, *living room*, *bedroom* dan *kitchen*.
6. Terdapat 7 jenis warna yang mampu diubah oleh LED RGB yaitu *red*, *green*, *blue*, *white*, *yellow*, *cyan* dan *magenta*.

1.6 Sistematika pembahasan

Uraian singkat mengenai metodologi penelitian pada masing-masing bab adalah sebagai berikut :

BAB I : **Pendahuluan**

Menjelaskan tentang latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika pembahasan dari "Kontrol Lampu Berbasis *Voice Command* Pada Raspberry Pi".

BAB II : **Landasan Kepustakaan**

Pada bab ini akan menjelaskan tentang landasan teori yang terkait dengan penelitian. Pada bab ini juga dijelaskan tentang penelitian serupa yang pernah dilakukan.

BAB III : **Metode Penelitian**

Membahas tentang langkah kerja yang dilakukan dalam penulisan diantaranya studi literatur, analisis kebutuhan sistem, perancangan sistem, dan implementasi *Alexa* pada Raspberry Pi.

BAB IV : **Perancangan dan Implementasi**

Bab ini menjelaskan perancangan sistem penelitian serta implementasi sistem baik dari internal sistem itu sendiri hinggal eksternal sistem.

BAB V : Pengujian dan Analisis

Membahas pengujian dan analisis berdasarkan perancangan dan implementasi. Pengujian ini dilakukan dengan menggunakan parameter keberhasilan perintah suara dan tingkat akurasinya.

BAB VI : Kesimpulan dan Saran

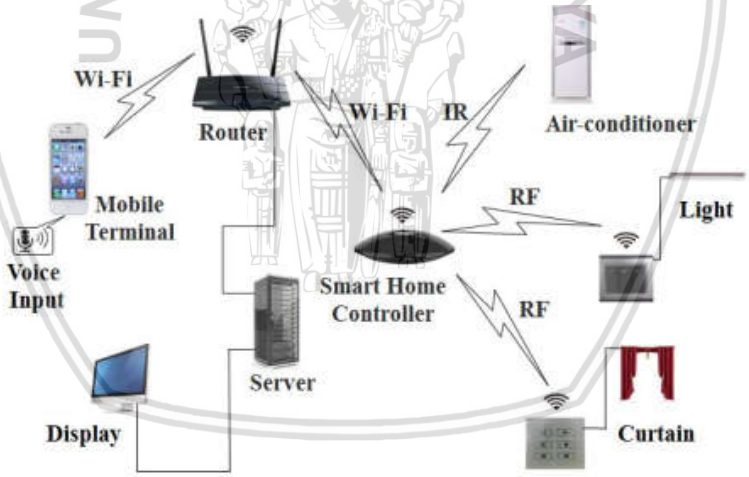
Membuat kesimpulan yang diperoleh dari hasil pengujian dan analisis berdasarkan perancangan dan implementasi serta saran-saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

Penelitian ini mengkaji beberapa jurnal penelitian terkait yang telah dipublikasikan diantaranya penelitian dari Yunliang dan Peng yang berjudul "*The design and implementation of the voice control system of smart home based on iOS*" yaitu penelitan yang memanfaatkan perangkat iOS untuk mengontrol peralatan pada *smart home* menggunakan *smartphone*. Selanjutnya penelitian yang berjudul "*Scheduler and Voice Recognition on Home*" oleh Syarif dan Syahrial yang memanfaatkan Raspberry Pi sebagai *gateway* pengontrol peralatan-peralatan dan yang terakhir adalah "*Home Automation Using Raspberry Pi through Siri Enabled Mobile Devices*" Ana dan kawan-kawan dimana penelitian mereka berfokus pada pengintegrasian Raspberry Pi dengan Siri.

2.1 Tinjauan Pustaka

Yunliang dan Peng (2016) menggunakan perangkat iOS untuk melakukan otomatis rumah dengan perintah suara pada *smart home*. Setelah perangkat iOS menerima perintah suara, perintah tersebut akan dikirimkan ke *database* pada *server* melalui jaringan internet untuk di kenali, lalu perintah tersebut akan dikembalikan ke perangkat terkait. Gambar 2.1 menampilkan struktur dari *voice control system* berbasis iOS.

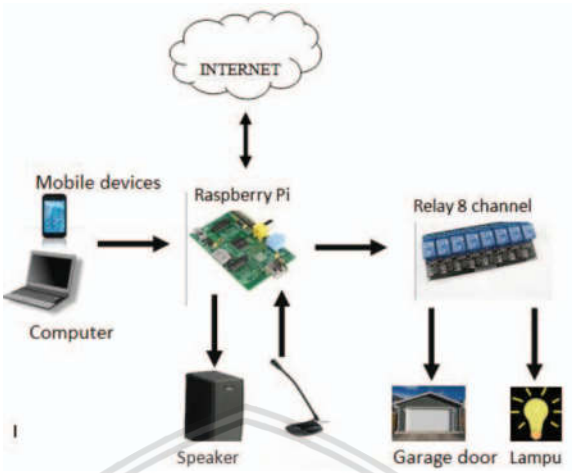


Gambar 2. 1 Struktur *Voice Control System* Berbasis iOS

Sumber : (Yunliang dan Peng)

Syarif dan Syahrial (2015) menggunakan perangkat Raspberry Pi sebagai pusat pengendali otomasi rumah. Otomasi rumah yang diajukan memiliki 3 cara berbeda untuk mengontrol peralatan elektronik. Pertama secara manual dengan tombol, kedua melalui *web interface* dan menggunakan *voice command*. *Voice command* pada sistem tersebut hanya digunakan untuk sekedar mengontrol keadaan dari peralatan elektronik yaitu kondisi *on* atau *off*, tidak sampai mengontrol suatu kondisi tertentu contohnya mengatur intensitas cahaya lampu atau suhu dari *air-*

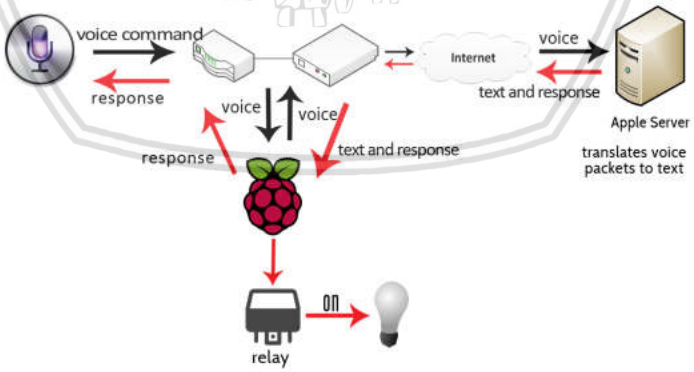
conditioner. Gambar 2.2 menampilkan arsitektur dari *Scheduler and Voice Recognition on Home*.



Gambar 2. 2 Arsitektur *Scheduler and Voice Recognition on Home*

Sumber : (Syarif dan Syahrial)

Penelitan selanjutnya adalah dari Ana dan kawan-kawan (2015) yaitu pegintegrasian teknologi Siri dengan Raspberry Pi. Agar sistem otomasi rumah dapat menggunakan kemampuan *speech recognition* dari aplikasi siri, sebuah *proxy server* dibutuhkan untuk melakukan translasi perintah suara yang nantinya hasil translasi akan dikirim ke *home device*. SiriProxy adalah sebuah *proxy server* yang digunakan di sistem untuk menerima teks hasil terjemahan dan respon dari *server apple* dan memprosesnya melalui SiriProxy plug-in. SiriProxy plug-in dibuat dengan tujuan sebagai perantara supaya GPIO port dapat menerjemahkan perintah yang sebelumnya dikirim dari *apple server.* Gambar 2.3 menampilkan arsitektur dari *Home Automation Using Raspberry Pi with Siri*.



Gambar 2. 3 Arsitektur *Home Automation Using* Raspberry Pi *with* Siri

Sumber : (Ana dan kawan-kawan)

2.2 Dasar Teori

Bab ini membahas beberapa teori terkait Kontrol Lampu Berbasis *Voice Command* Pada Raspberry Pi. Beberapa diantaranya yaitu *Smart Home, Speech Recognition, Alexa* dan *Raspberry Pi*.

2.2.1 *Smart Home*

Smart home, atau juga dikenal sebagai rumah otomatis, bangunan cerdas adalah sebuah konsep yang saat ini sedang berkembang. *Smart home* menggabungkan perangkat umum yang mengendalikan fitur rumah. Awalnya, teknologi *smart home* digunakan untuk mengendalikan sistem lingkungan seperti pencahayaan dan pemanasan, namun baru-baru ini penggunaan teknologi rumah pintar telah berkembang sehingga hampir semua komponen listrik di dalam rumah dapat di integrasikan dalam sistem. Saat ini teknologi *smart home* tidak hanya terbatas untuk menghidupkan dan mematikan perangkat, tapi juga bisa memantau lingkngan internal dan aktivitas yang sedang dilakukan di rumah tersebut. (Vincent dkk., 2006).

Pada awalnya teknologi *smart home* menggunakan remote untuk melakukan fungsi kontrol terhadap perangkat-perangkat yang terdapat pada rumah tersebut. Seiring berkembangnya teknologi, saat ini untuk melakukan kontrol perangkat-perangkat yang terdapat dirumah pintar dapat dilakukan melalui *smartphone*, *website*, bahkan melalui perintah suara.

2.2.2 *Speech Recognition*

Speech recognition adalah proses mengubah sinyal ucapan ke urutan kata, dengan menggunakan algoritma yang diimplementasikan sebagai program komputer(Anusuya, dkk., 2009). Pola kerja *speech recognition* adalah mencocokan sinyal akustik yang diterima dengan data yang tersimpan dalam *template* ataupun *database*. Proses pencocokan memiliki dua model utama yaitu Model Akustik yang terdiri dari fonem yang memiliki nilai tertentu yang diambil dari sinyal akustik dan Model bahasa berupa metode yang mengestimasikan satu kata diikuti oleh serangkaian kata lainnya.

Speech Recognition juga merupakan sistem yang digunakan untuk mengenali perintah kata dari suara manusia dan kemudian diterjemahkan menjadi suatu data yang dimengerti oleh komputer. Pada saat ini, sistem ini digunakan untuk menggantikan peranan input dari keyboard dan mouse. Keuntungan dari sistem ini adalah pada kecepatan dan kemudahan dalam penggunaannya. Kata – kata yang ditangkap dan dikenali bisa jadi sebagai hasil akhir, untuk sebuah aplikasi seperti *command* & kontrol, penginputan data, dan persiapan dokumen. Parameter yang dibandingkan ialah tingkat penekanan suara yang kemudian akan dicocokkan dengan *template database* yang tersedia. Sedangkan sistem pengenalan suara berdasarkan orang yang berbicara dinamakan speaker *recognition*

2.2.3 Alexa

Alexa merupakan sebuah *intelligent personal assistant* yang dikembangan oleh Amazon dan populer berkat Amazon Echo dan perangkat Amazon Echo Dot yang dikembangkan oleh Amazon Lab126. Perangkat tersebut mampu berinteraksi menggunakan suara, memainkan musik, membuat list hal yang ingin dilakukan,

pengaturan *alarm*, memaikan *audiobooks*, menyediakan informasi cuara, lalu lintas dan informasi *real time* lainnya (Grant Clauser, 2017).

Alexa pertama kali dikenalkan Amazon pada November 2014 bersama dengan Echo, Alexa terinspirasi dari *computer voice* dan sistem percakapan pada film fiksi sains Stark Trek. Pada Januari 2017, konferensi Alexa pertama diadakan di Nashville yang mengumpulkan komunitas pengembang Alexa. Hingga saat ini Alexa telah dikembangkan hingga ke ranah *home automation*.

Salah satu fitur Alexa yang dapat digunakan pihak pengembangnya adalah *Alexa Skills Kit*. Fitur tersebut memungkinkan para pengembang untuk membuat dan mem-*publish* kemampuan untuk Alexa menggunakan *Alexa Skills Kit*. Dengan fitur tersebut, setiap pengembang dapat menambahkan kemampuan Alexa untuk menerjemahkan perintah-perintah baru khususnya di bidang *home automation*.

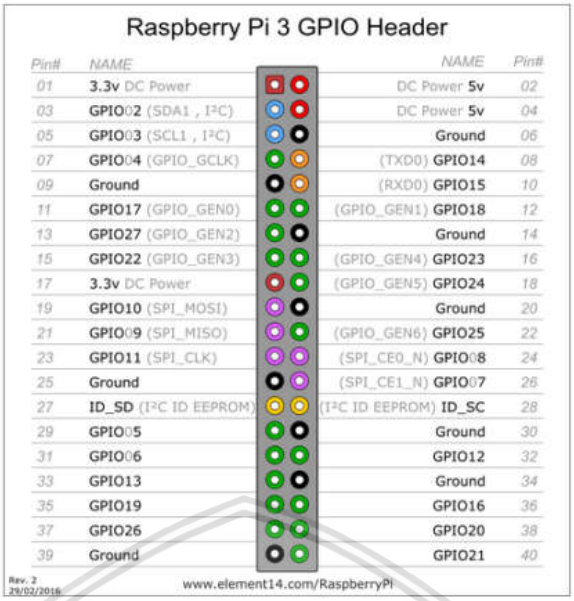
2.2.4 Raspberry Pi

Raspberry Pi adalah sebuah komputer papan tunggal (*single-board computer*) atau SBC berukuran kartu kredit. Raspberry Pi telah dilengkapi dengan semua fungsi layaknya sebuah komputer lengkap, menggunakan SoC (*System-on-chip*) ARM yang dikemas dan diintegrasikan diatas PCB. Perangkat ini menggunakan kartu SD untuk *booting* dan penyimpanan jangka panjang

Raspberry Pi memiliki dua model yaitu model A dan model B. Secara umum Raspberry Pi Model B, 512MB RAM. Perbedaan model A dan B terletak pada *memory* yang digunakan, Model A menggunakan *memory* 256 MB dan model B 512 MB. Selain itu model B juga sudah dilengkapai dengan *ethernet port* (kartu jaringan) yang tidak terdapat di model A. Desain Raspberry Pi didasarkan seputar SoC (*System-on-a-chip*) Broadcom BCM2835, yang telah menanamkan prosesor ARM1176JZF-S dengan 700 MHz, VideoCore IV GPU, dan 256 Megabyte RAM (model B). Penyimpanan data didesain tidak untuk menggunakan *hard disk* atau *solid-state drive*, melainkan mengandalkan kartu SD (SD *memory card*) untuk *booting* dan penyimpanan jangka panjang.

Raspberry Pi bersifat *open source* (berbasis Linux), Raspberry Pi bisa dimodifikasi sesuai kebutuhan penggunanya. Sistem operasi utama Raspberry Pi menggunakan Debian GNU/Linux dan bahasa pemrograman Python. Salah satu pengembang OS untuk Raspberry Pi telah meluncurkan sistem operasi yang dinamai Raspbian, Raspbian diklaim mampu memaksimalkan perangkat Raspberry Pi. Sistem operasi tersebut dibuat berbasis Debian yang merupakan salah satu distribusi Linux OS.

Salah satu fitur *powerful* dari Raspberry Pi adalah sederetan GPIO (general purpose input/output) Pinnya. Pada Raspberry Pi 3 terdapat 40 pin, setiap pin-pin tersebut mempunya fungsinya masing-masing. Input pada pin GPIO bukan hanya sekedar dari *switch* melainkan dapat dari sensor atau sinyal dari komputer lainnya, begitu juga dengan *output* yang dapat melakukan banyak hal. Gambar 2.4 merupakan gambar GPIO Header dari Raspberry Pi 3

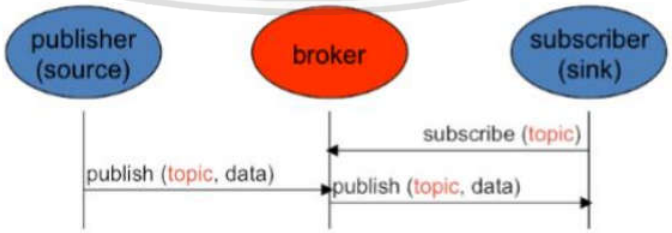


Gambar 2. 4 Raspberry Pi 3 Model B GPIO 40 Pin Block Pinout

Sumber : (www.element14.com/RaspberryPi)

2.2.5 MQTT

Message Queuing Telemetry Transport (MQTT) adalah sebuah protokol konektivitas *machine to machine* (M2M) atau *Internet of Things* (IoT) yang berbasis *open source* (Eclipse) dengan standard terbuka (Oasis) yang digunakan untuk perangkat terbatas dan *bandwidth* yang rendah, dengan adanya *latency* tinggi atau berjalan pada jaringan yang tidak sesuai. Protokol MQTT menggunakan prinsip *publish-subscribe*. *Publisher* menghasilkan informasi tertentu dan menerbitkan informasi tersebut, sedangkan *subsciber* merupakan client yang tertarik untuk mendapatkan informasi tertentu dengan mendaftarkan minat dari informasi tersebut. Dan terdapat broker yang menjamin *subscriber* mendapatkan informasi yang disajikan oleh *publisher*. (Weiss, dkk., 2013). Gambar 2.5 merupakan gambar mekanisme komunikasi *publish-subscribe*.



Gambar 2. 5 Mekanisme Komunikasi Publish/Subscribe

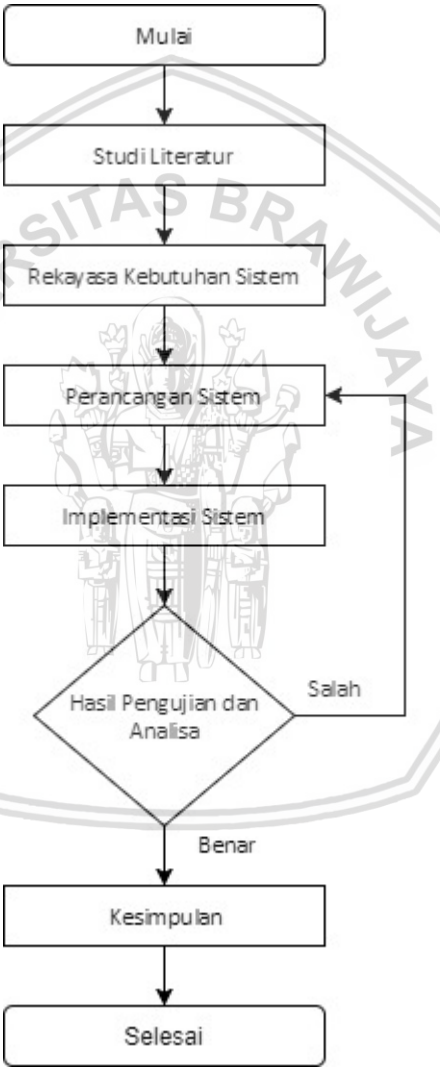
Sumber : (Weiss, Beat. Hunkeler, Urs. Munari, Andrea. Schott, Wolfgang. Truong, Hong Linh, 2013)

BAB 3 METODOLOGI

Pada Bab 3 Metodologi ini akan menjelaskan bagaimana pengimplementasian sistem yang terdiri dari alur metodologi penelitian, studi literatur, rekayasa kebutuhan sistem, perancangan sistem, pengujian dan analisa serta penutup.

3.1 Metode Penelitan

Metode Penelitian pada Tugas Akhir ini merupakan langkah-langkah pengerjaan dan cara penyelesaian Tugas Akhir, berikut ini merupakan alur metode penelitian yang dilakukan :



Gambar 3. 1 Metodologi Penelitian

3.2 Studi Literatur

Pada Penelitian tugas akhir ini, studi literatur dilakukan untuk pemahaman terhadap tinjauan pustaka dan dasar teori, berikut ini merupakan studi literatur yang dilakukan:

1. *Smart Home*

   Studi literatur mengenai *Smart Home* dilakukan dengan memahami buku dan paper terkait *Smart Home*. Bagaimana mengimplementasikan *smart home*. Pada tahap ini, peneliti mempelajari apa saja komponen yang terdapat pada *smart home*

2. *Speech Recognition*

   Studi literatur mengenai *Speech Recognition* dilakukan kajian dari beberapa literatur terkait baik dari jurnal, buku maupun publikasi di internet. Penelitian ini mempelajari bagaimana cara *Speech Recognition* bekerja.

3. *Alexa*

   Pada studi literatur mengenai *Alexa*, peneliti mempelajari bagaimana cara mengimplementasikan *Alexa* pada Raspberry Pi dan mempelajari bagaimana menambahkan fitur atau kemampuan dari Alexa untuk dapat menerjemahkan perintah yang di kirim melalui perangkat Raspberry Pi

4. MQTT

   Pada studi literatur mengenai MQTT, peneliti akan melakukan kajian pada jurnal yang menjelaskan mekanisme sistem *publish-subscribe* yang merupakan cara komunikasi pesan secara tidak langsung atau *indirect communication* pada suatu jaringan. Pada penelitian ini dipelajari bagaimana cara kerja sistem *publish-subscribe* secara umum dan protokol yang bekerja dalam mendukung sistem tersebut

3.3 Rekayasa Kebutuhan Sistem

Rekayasa kebutuhan sistem dilakukan untuk mengetahui apa saja yang dibutuhkan oleh sistem agar sistem dapat dikatakan bekerja sesuai dengan tujuan sistem. Kebutuhan dari sistem pada penelitian ini terdiri dari gambaran umum sistem yang didalamnya terdapat pembahasan perspektif sistem, ruang lingkup, karakteristik pengguna, serta asumsi dan ketergantungan sistem. Selain itu terdapat pula pembahasan kebutuhan fungsional yang nantinya akan dijelaskan secara rinci pada bab selanjutnya mengenai rekayasa kebutuhan.

3.4 Perancangan Sistem

Tahap perancangan sistem ini bertujuan agar perancangan sistem penelitian yang dilakukan menjadi terstruktur. Perancangan sistem akan dimulai dengan merancang masing-masing sub sistem yang diperlukan diantaranya Sub Sistem Mikrofon, Sub Sistem LED dan Sub Sistem Relay. Selain perancangan sistem itu sendiri terdapat pula perancangan *eksternal* sistem dari *Amazon Web Service* yaitu perancangan *Alexa Skills Kit* dan kode program pada *Amazon Web Service Lamda* yang nantinya akan di jelaskan secara rinci pada bab mengenai perancangan sistem.

3.5 Implementasi Sistem

Tahap implementasi sistem ini akan menjelaskan arsitektur sistem dan bagaimana alur perintah suara tersebut di proses. Implementasi sistem ini dilakukan ketika semua proses perancangan sistem telah dipenuhi. Pembahasan tentang implementasi sistem disini antara lain menjelaskan tentang implementasi pada Raspberry Pi sebagai perangkat penerima perintah suara dan kode program untuk menerima hasil terjemahan perintah suara. Selanjutnya implementasi pada *Amazon Web Service* yaitu *Alexa Skills Kit* dan kode program pada *Amazon Web Service Lambda* juga akan dibahas pada bab ini.

3.6 Pengujian dan Analisa

Pada tahap pengujian sistem ini menggunakan parameter yang disesuaikan pada perancangan sistem. Tahap ini akan menjelaskan bagaimana penelitian dilakukan dari perancangan dan implementasi lalu dilakukan pengujian dan analisa yang akan dijelaskan pada bab 6.

3.7 Kesimpulan

Pada bab kesimpulan ini akan menjelaskan bagaimana hasil dari penelitian yang telah dilakukan dan saran yang nantinya akan membantu penelitian selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

Pada Bab 4 Rekayasa Kebutuhan ini akan dijelaskan mengenai kebutuhan-kebutuhan dari sistem yang dibangun agar bisa memenuhi tujuan yang telah dibuat sebelumnya. Kebutuhan tersebut meliputi gambaran umum sistem dan kebutuhan fungsional. Dengan adanya rekayasa kebutuhan diharapkan sistem yang akan dibuat dapat bekerja dengan baik.

4.1 Gambaran Umum Sistem

Sistem ini bekerja untuk mengontrol LED RGB dan lampu AC menggunakan Relay yang dihubungkan dengan mini komputer Raspberry menggunakan *voice command*. Perangkat Raspberry berfungsi sebagai modul *speech recognition* sekaligus perangkat untuk melakukan kontrol pada LED seperti mematikan, menyalakan, mengubah warna, mengatur intentsitas dan melakukan penjadwalan kapan LED tersebut harus mati atau hidup. Sedangkan untuk Relay hanya sebatas mematikan, menyalakan dan melakukan penjadwalan.

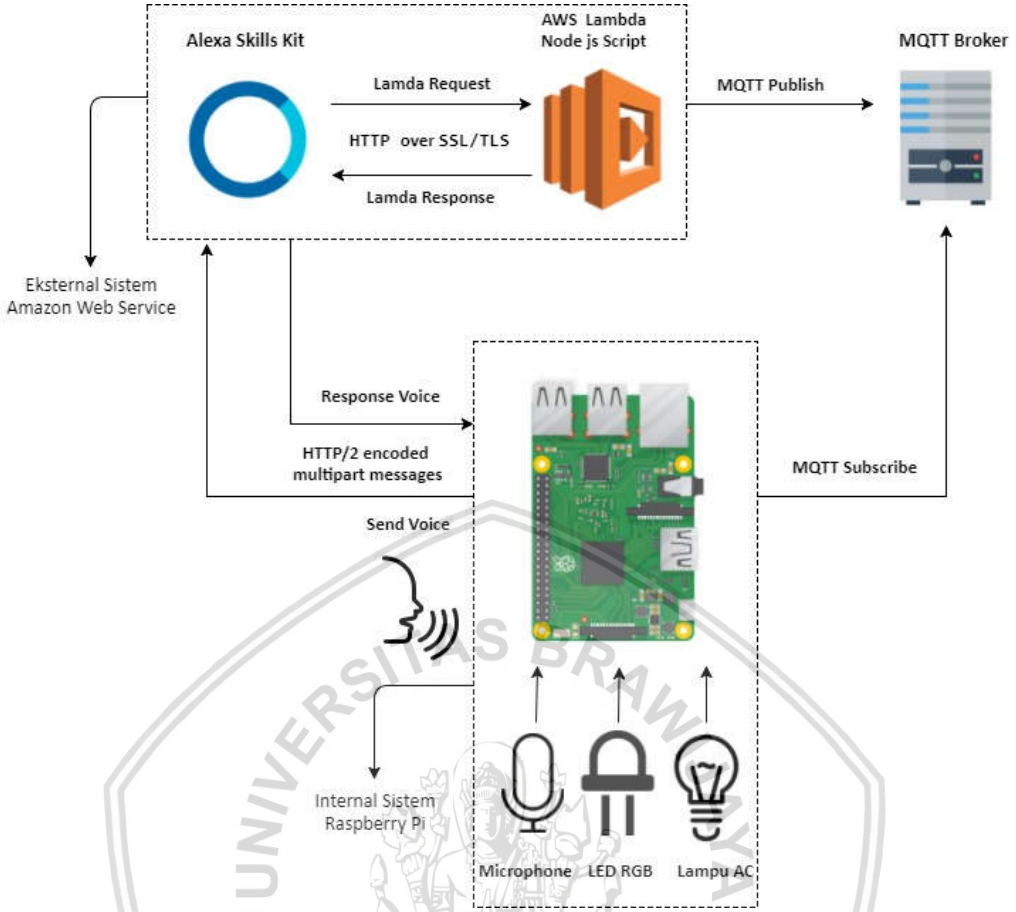
Raspberry tersebut nantinya akan di *install* sebuah *service* yang bernama *Alexa Voice Service* sehingga perangkat tersebut memiliki kemampuan menjadi modul *speech recognition*. Pada sistem ini, suara yang berhasil diterima oleh perangkat Raspberry melalui mikrofon akan diteruskan ke *Amazon Web Service* dan diolah sehingga menghasilkan teks yang nantinnya akan digunakan untuk penentuan pengambilan keputusan.

Sistem ini menggunakan metode *publish-subscribe* MQTT untuk melakukan pengiriman data teks yang telah dimodelkan sedemikian rupa di *Amazon Web Service Lambda*. Data akan di *publish* ke *broker* MQTT dan di *subscribe* oleh perangkat Raspberry sebagai acuan untuk melakukan kontrol terhadap LED dan Relay.

Pada sistem ini terdapat empat lokasi lampu yang dapat di kontrol diantaranya ***garden***, ***living room***, ***bedroom*** untuk LED RGB dan ***kitchen*** lampu yang terpasang pada relay. Untuk fungsi kontrol mengubah warna pada LED, warna yang dapat diubah diantaranya ***red***, ***green***, ***blue***, ***white***, ***yellow***, ***cyan*** dan ***magenta***. Sedangkan untuk mengatur intensitas, nilai presentasi yang dapat diterima oleh sistem adalah **0 – 100 %**. Untuk fungsi penjadwalan parameter ucapan waktu yang dapat diterima oleh sistem adalah format penyebutan jam **am**-**pm** yaitu format waktu 12 jam.

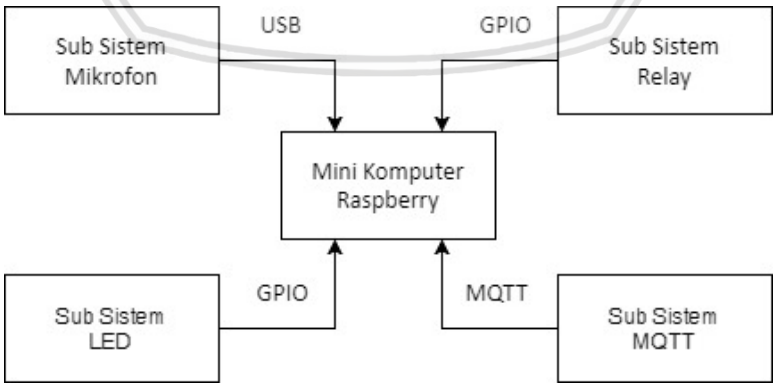
4.1.1 Perpektif Sistem

Sistem ini pada akhirnya ditujukan untuk setiap orang yang menginginkan teknologi *smart home* pada rumah mereka. Sistem ini akan akan dipasang di salah satu ruangan yang strategis untuk melakukan kontrol lampu. Sistem ini merupakan sebuah sistem besar yang dapat dilihat pada Gambar 4.1.



Gambar 4.1. Sistem Secara Keseluruhan

Seperti terlihat pada Gambar 4.1, sistem akan dibagi menjadi dua bagian yaitu eksternal sistem dan internal sistem. Pada eksternal sistem *Amazon Web Service* dilakukan dua percangan dan implementasi yaitu pada ***Alexa Skills Kit*** dan **AWS Lambda**.



Gambar 4.2 Sub Sistem Dari Internal Sistem Raspberry Pi

Pada Gambar 4.2 menunujukan bagian-bagian dari Internal Sistem Raspberry Pi ditunjukan pada Gambar 4.1 yang telah dibagi menjadi beberapa sub sistem yang nantinya akan dirancang dan diimplementasikan.

4.1.2 Ruang Lingkup

Sistem ini dibuat untuk digunakan oleh setiap orang yang menginginkan adanya fungsi kontrol pada rumahnya untuk mengontrol lampu. Saat ini sudah banyak cara yang dipakai untuk mengontrol lampu pada *smart home* mulai dari menggunakan *remote*, melalui *website* hingga melalui perintah suara. Maka dari itu sistem ini dibuat dengan memanfaatkan layanan amazon bernama *Alexa Voice Service* supaya sistem dapat melakukan kontrol lampu berbasis perintah suara.

4.1.3 Karakteristik Pengguna

Karakteristik pengguna alat ini diperuntukan untuk pengguna yang menginginkan kemudahan dalam pengontrolan alat-alat pada *Smart Home* contohnya lampu (LED). Selain itu pengguna harus memiliki kemampuan berbahasa inggris dikarenakan sistem ini menggunakan Bahasa Inggris.

4.1.4 Asumsi dan Ketergantungan

Asumsi ketergantungan yang terdapat pada sistem ini antara lain :

1. Alat ini menggunakan mini USB Mikrofon sehingga jarak ideal pengguna melakukan kontrol melalui *voice command* adalah radius 0-4 meter dari alat ditempatkan.
2. Alat ini akan mengirimkan data suara ke *Amazon Web Service* dan melakukan *subscribe* ke broker MQTT sehingga membutuhkan koneksi internet.
3. Alat ini membutuhkan sumber daya listrik dengan tegangan 5v untuk menyalakan perangkat Raspberry.

4.2 Kebutuhan Fungsional

Adapun kebutuhan fungsional yang harus dipenuhi dalam penelitian ini antara lain :

1. **Minikomputer Raspberry dapat menjadi modul *speech recognition*.**

   Minikomputer Raspberry akan dipasang sebuah layanan berupa *Alexa Voice Service* sehingga mini komputer tersebut dapat menerima masukan suara melalui Mini USB Microphone. Nantinya suara tersebut akan dikirim ke *Alexa Skills Kit* dimana perinta suara tersebut akan di proses disana dan selanjutnya diteruskan ke AWS Lambda.

2. **Minikomputer Raspberry dapat melakukan *subscribe* ke *broker* MQTT**

   Minikomputer Raspberry akan melakukan *subscribe* ke broker MQTT dengan topik **lamp_control** untuk mendapatkan hasil perintah suara yang telah diterjemahkan Alexa. Sebelum terjemahan perintah suara tersebut dikirim ke *broker* MQTT, perintah suara tersebut akan di modelkan di *AWS Lamda* supaya menjadi format JSON dan di *publish* ke MQTT *broker*.

3. **Sistem dapat mematikan dan menyalakan LED atau lampu pada relay dengan perintah suara berbahasa inggris**.

   Sistem akan mematikan atau menyalakan LED atau lampu pada relay berdasarkan data hasil *subscribe* ke MQTT *broker*. Ketika terdapat parameter status *on* atau *off* serta parameter lokasi dari LED atau lampu pada relay tersebut, maka sistem akan melakukan manipulasi pada pin GPIO raspberry untuk menyalakan atau mematikan LED atau lampu pada relay tersebut.

4. **Sistem dapat mengubah warna LED dengan perintah suara berbahasa inggris**.

   Sistem akan mengubah warna LED berdasarkan data hasil *subscribe* ke MQTT *broker*. Ketika terdapat parameter *color* serta parameter lokasi dari LED tersebut, maka sistem akan melakukan manipulasi pada pin GPIO raspberry untuk mematikan LED.

5. **Sistem dapat mengubah intensitas cahaya LED dengan perintah suara berbahasa inggris**.

   Sistem akan mengubah intensitas cahaya dari LED berdasarkan data hasil *subscribe* ke MQTT broker. Ketika terdapat parameter *percentage* serta parameter lokasi dari LED tersebut, maka sistem akan melakukan manipulasi pada pin GPIO raspberry untuk mematikan LED.

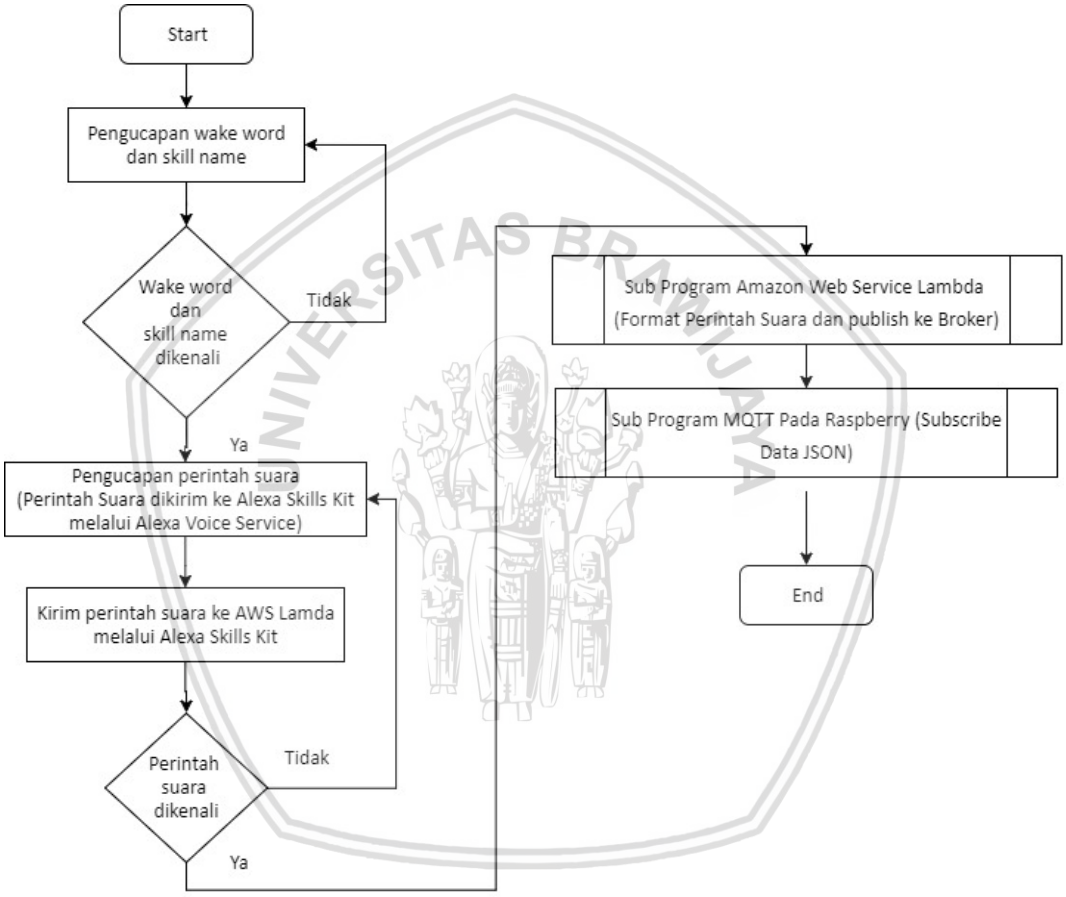
6. **Sistem dapat melakukan penjadwalan untuk mematikan dan menyalakan LED atau lampu pada relay dengan perintah suara berbahasa inggris**.

   Sistem akan mematikan atau menyalakan LED atau lampu pada relay berdasarkan data hasil *subscribe* ke MQTT broker. Ketika terdapat parameter status *on* atau *off*, parameter *time*, serta parameter lokasi dari LED atau lampu pada relay tersebut, maka sistem akan melakukan manipulasi pin GPIO raspberry untuk menyalakan atau mematikan LED atau lampu pada relay berdasarkan waktu yang telah ditentukan.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada bab ini akan membahas tentang perancangan dan implementasi diantaranya adalah perancangan dan implementasi dari internal sistem yaitu Sub Sistem Mikrofon, Sub Sistem LED, Sub Sistem Relay dan Sub Sistem MQTT. Selain itu dilakukan juga perancangan dan implementasi dari eksternal sistem yaitu *Alexa Voice Service*, *Alexa Skills Kit* dan *Amazon Web Service Lambda*.

Adapun cara kerja sistem secara garis besar ditunjukan oleh *flowchart* dibawah ini pada Gambar 5.1.



Gambar 5.1 Alur Sistem Secara Keseluruhan

5.1 Perancangan

5.1.1 Perancangan Alexa Skills Kit

Pada tahap perancangan *Alexa Skills Kit*, dilakukan beberapa hal diantaranya mendefinisikan ***invocation name***, ***interaction model*** dan ***service end point type***. *Incovation name* merupakan pendefinisian nama *skill* yang nantinya akan disebutkan diawal penggunaan sistem. Pada sistem ini *invocation name* akan diberi nama ***lamp control mode*** dimana nantinya ketika *invocation name* disebut, maka sistem akan diarahkan ke *skill* tersebut sehingga sistem memiliki

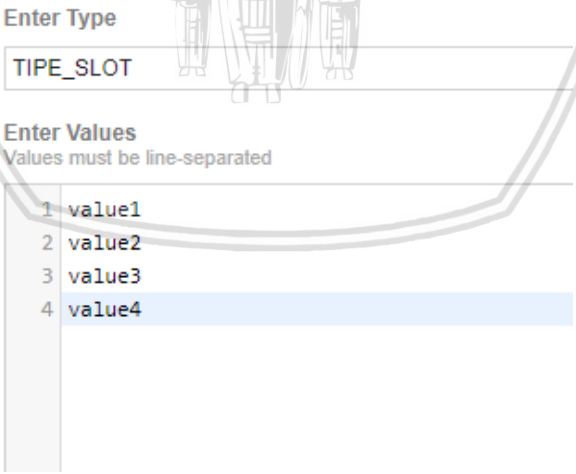
kemampuan untuk menerjemahkan beberapa perintah yaitu mematikan, menyalakan, merubah warna, mengatur intentsitas dan menjadwalkan lampu.

Selanjutnya adalah *interaction model*, perancangan pada *interaction model* adalah merancang ***intent schema*** (representasi tindakan yang dapat memenuhi permintaan pengucap), ***custom slot*** (argument atau kata kunci dari kalimat pengucap) dan ***sample utterances*** (contoh pengucapan). *Intent schema* di desain dengan bentuk JSON seperti terlihat pada Gambar 5.2.

```
{
  "slots": [
    {
      "name": "nama_slot_1",
      "type": "TIPE_SLOT"
    },
    {
      "name": "nama_slot_2",
      "type": "TIPE_SLOT"
    }
  ],
  "intent": "NamaIntent"
},
```

Gambar 5.2 Perancangan Intent Schema

Pada perancangan *custom slot*, didefinisikan *slot type* dan *value* yang merepresentasikan dari *slot* tersebut. *Slot type* merupakan sebuah nama yang mewakili *value* yang terdapat di dalamnya. Sedangkan *value* merupakan nilai yang merepresentasikan *slot type* tersebut. Model pendefinisiannya dapat dilihat pada Gambar 5.3.



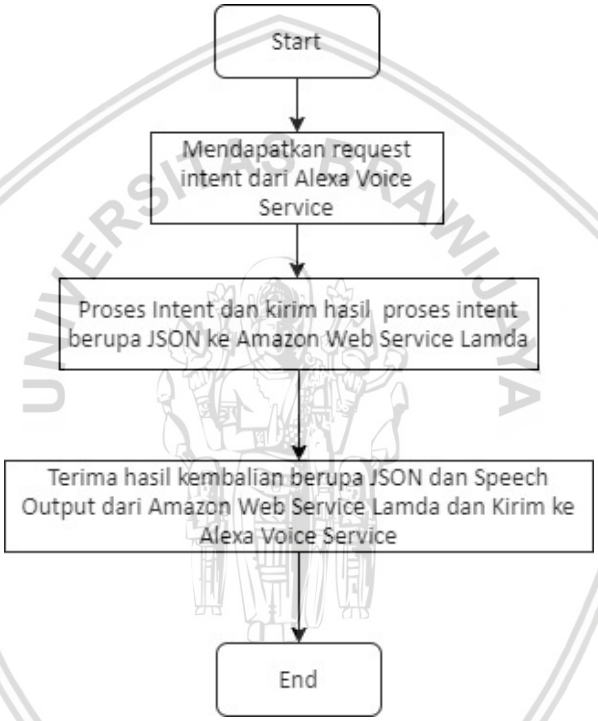
Gambar 5.3 Perancangan Custom Slot

Sedangkan untuk *sample utterance*, didefinisikan beberapa contoh pengucapain yang mendukung tujuan sistem sehingga dapat mematikan, menyalakan, mengubah warna, mengatur intensitas dan melakukan penjadwalan. Cara pendefinisiannya dapat dilihat pada Gambar 5.4.

```
NamaIntent kalimat bebas {nama_slot_1} kalimat bebas {nama_slot_2}
NamaIntent {nama_slot_1} kalimat bebas {nama_slot_2}
NamaIntent kalimat bebas {nama_slot_1} {nama_slot_2}
```

Gambar 5.4 Perancangan Sample Utterances

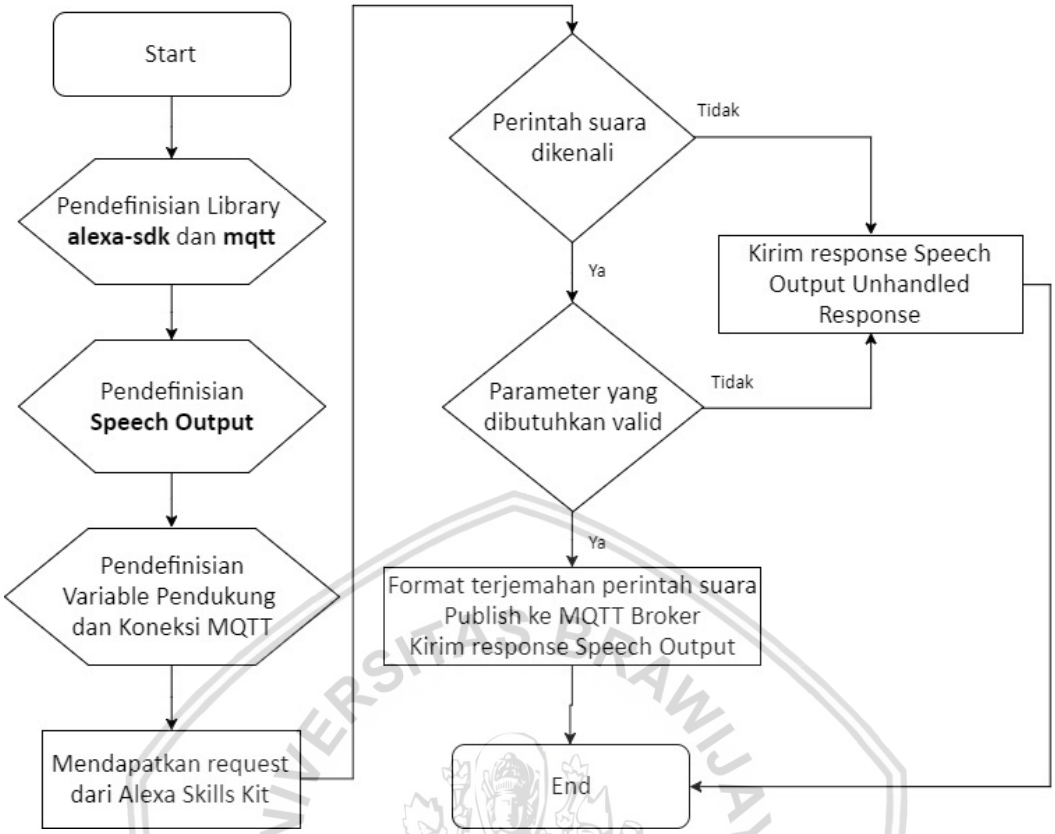
Selanjutnya adalah *service end point type*. *Service end point type* pada sistem ini merupakan alamat dari *Amazon Web Service Lambda*. Nantinya hasil terjemahan suara pada *Alexa Skills Kit* yang berbentuk JSON akan dikirimkan ke alamat *Amazon Web Service Lambda* tersebut dan diolah sehingga data yang di publish ke MQTT Broker suda menjadi bentuk lain yang dikenali oleh sistem yang nantinya akan men-*subscribe* data tersebut. Alur yang terdapat pada *Alexa Skills Kit* ini dapat dilihat di *flowchat* pada Gambar 5.5.



Gambar 5.5 Alur Kerja Pada Alexa Skills Kit

5.1.2 Perancangan Amazon Web Service Lamda

Pada tahap perancangan *Amazon Web Service Lambda* langkah yang dilakukan adalah membuat sebuah kode program yang nantinya akan menangani terjemahan *intent* dari *Alexa Skills Kit*. Kode program akan dibuat dengan Bahasa pemrograman Node.js. Nantinya program tersebut akan di upload ke *Amazon Web Service Lambda* dan di-*compile* disana. Program tersebut memiliki kemampuan menerjemahkan *intent request*, mengambil nilai slot, memodelkan data yang di dapat menjadi bentuk yang dikenali oleh sistem. Nantinya data yang sudah di format tersebut akan di publish ke MQTT Broker. Alur kerja pada *Amazon Web Service Lamda* dapat dilihat pada Gambar 5.6.



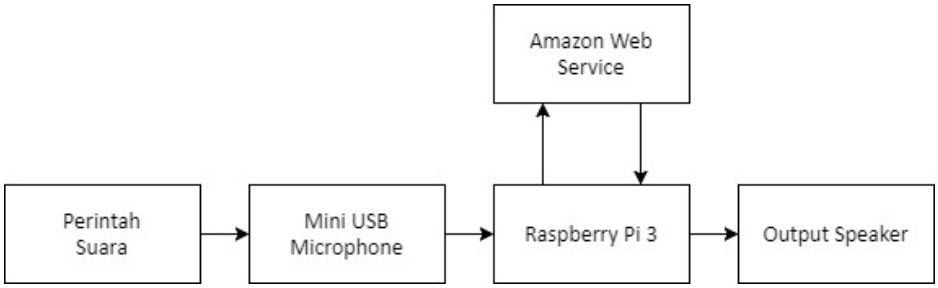
Gambar 5.6 Alur Kerja Kode Program Pada Amazon Web Service Lambda

5.1.3 Perancangan Sub Sistem Mikrofon (Modul Speech Recognition)

Pada tahap perancangan sub sistem mikrofon (modul *speech recognition*) ditentukan penggunaan perangkat-perangkat yang dibutuhkan seperti mini komputer yang digunakan adalah raspberry pi 3 dan untuk mini mikrofonnya adalah Mini USB Microphone. Spesifikasi dari mini usb microphone tersebut adalah sebagai berikut :

1. Sensitivity: -47dB±4dB
2. Sensitivity Loss: -3dB at 1.5V
3. Work Voltage: 4.5V
4. Frequency Response: 100Hz-16kHz
5. SNR: > -67dB

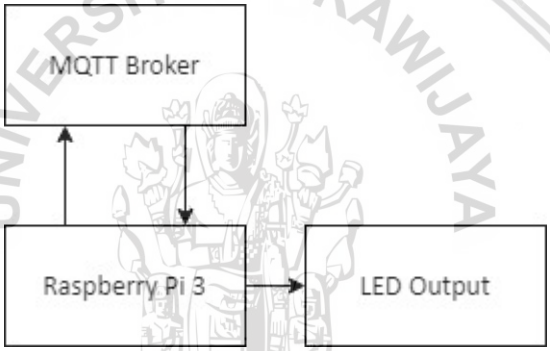
Selain Mini USB Microphone, sub sistem ini juga akan dihubungkan dengan sebuah output speaker untuk mengeluarkan suara *response* dari perintah yang telah diucapkan. Blok diagram dari sub sistem ini dapat dilihat pada Gambar 5.7.



Gambar 5.7 Blok Diagram Sub Sistem Mikrofon

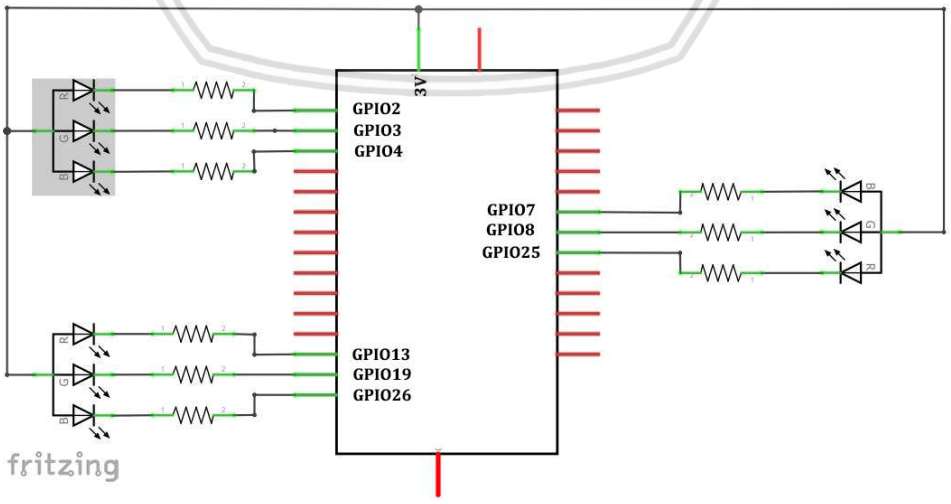
5.1.4 Perancangan Sub Sistem LED

Pada tahap perancangan Sub Sistem LED akan dilakukan penentuan penempatan LED ke pin GPIO raspberry pi. LED nantinya akan menjadi output dari hasil perintah suara yang diucapkan. Sub sistem ini nantinya akan dipengaruhi oleh data hasil subscribe ke MQTT Broker yaitu berupa perintah mematikan, menyalakan, mengubah warna, mengatr rintensitas dan menjadwalkan LED. Blok diagram sub sistem ini dapat dilihat pada Gambar 5.8.



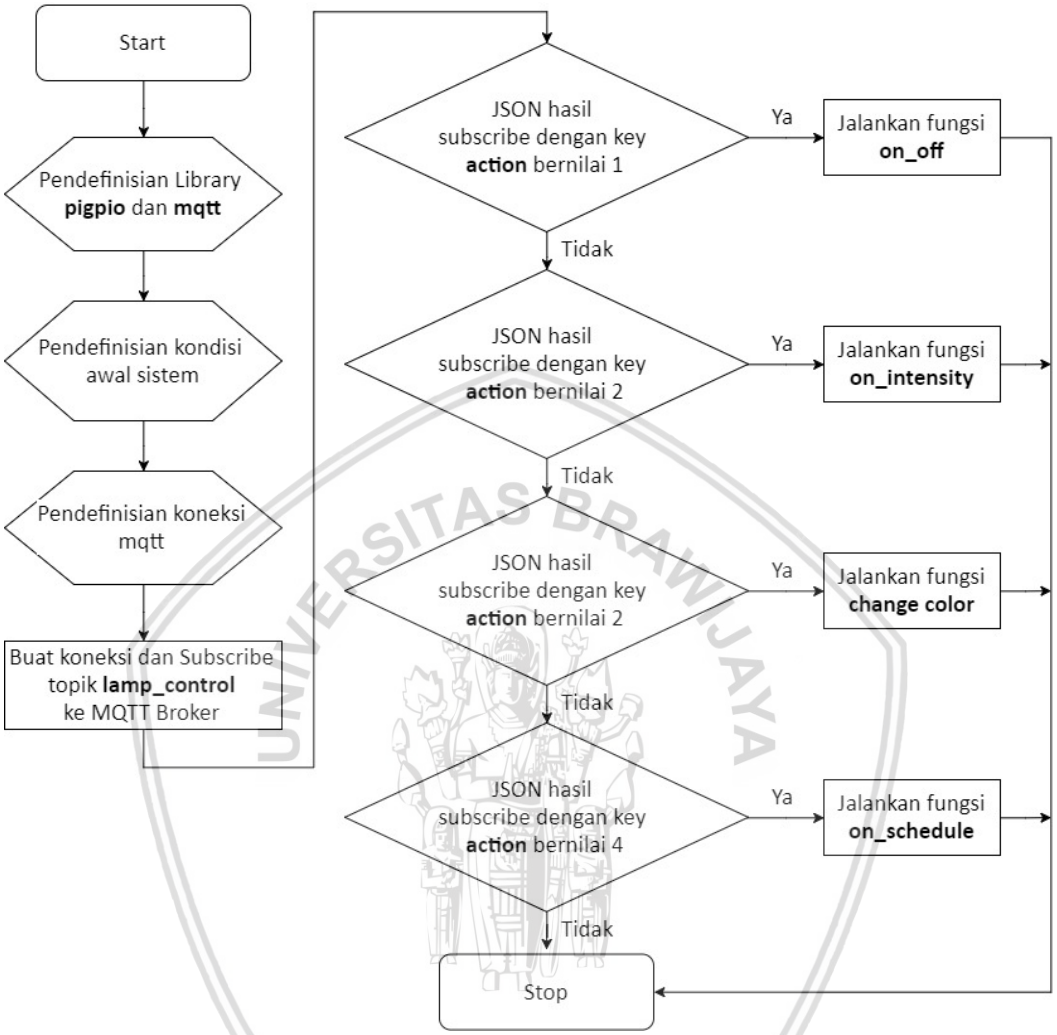
Gambar 5.8 Blok Diagram Sub Sistem LED

Sedangkan untuk skematik koneksi antara pin LED ke GPIO pin raspberry pi dapat dilihat pada Gambar 5.9 berikut.



Gambar 5.9 Skematik Sub Sistem LED

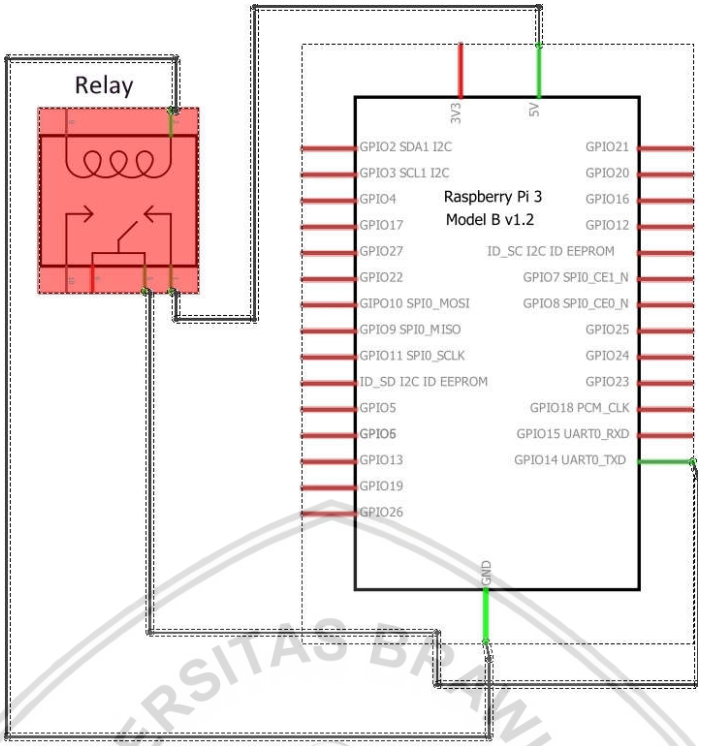
Selain penentuan penempatan LED ke pin GPIO, dilakukan juga perancangan kode program pada sub sistem LED. Gambar 5.10 memperlihatkan alur dari program pada sub sistem LED.



Gambar 5.10 Alur Kerja Kode Program Pada Sub Sistem LED

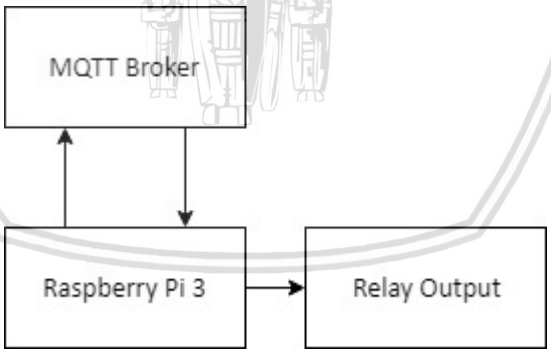
5.1.5 Perancangan Sub Sistem Relay

Pada tahap perancangan Sub Sistem Relay akan dilakukan penentuan penempatan Relay ke pin GPIO raspberry pi. Relay nantinya akan menjadi output dari hasil perintah suara yang diucapkan. Sub sistem ini nantinya akan dipengaruhi oleh data hasil subscribe ke MQTT Broker yaitu berupa perintah mematikan, menyalakan saja dikarenakan lampu dengan arus AC hanya mampu untuk dimatikan dan dinyalakan. Skematik untuk koneksi antara relay dan raspberry pi dapat dilihat pada Gambar 5.11, sedangkan untuk blok diagramnya dapat dilihat pada Gambar 5.12.



Gambar 5.11 Skematik Sub Sistem Relay

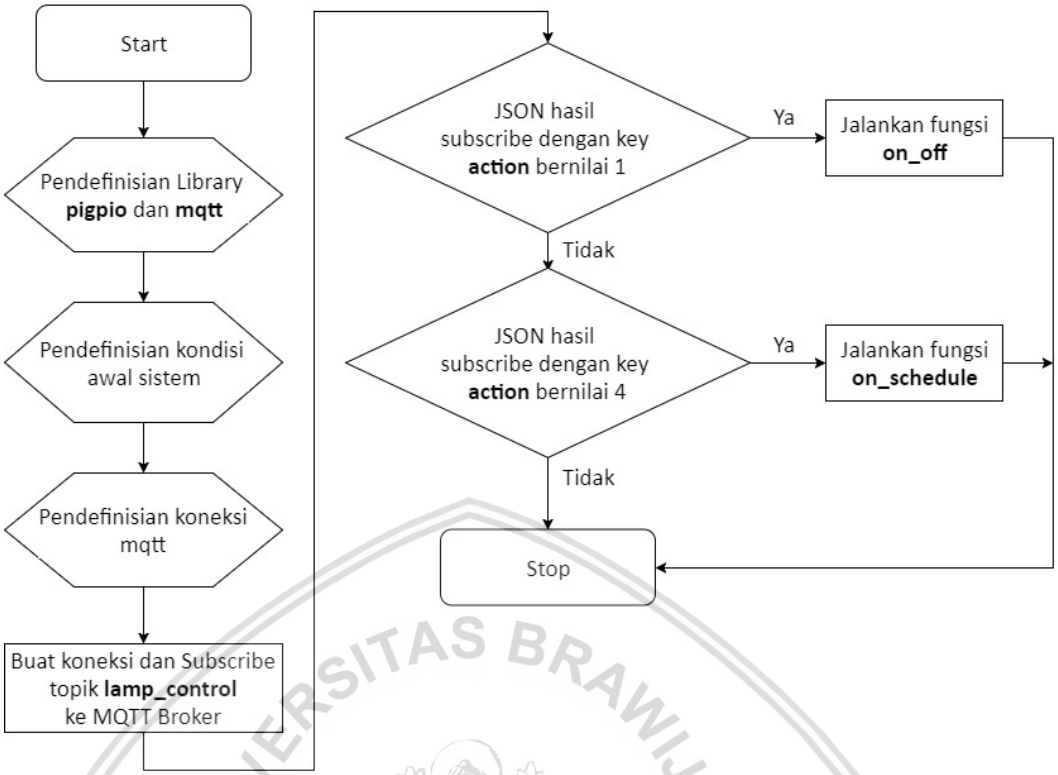
Gambar 5.11 menunjukan hubungan antara pin GPIO dengan pin pada relay. Pin Vin pada relay dihubungkan dengan pin 5v pada raspberry, pin Gnd dihubungkan dengan pin ground dan pin IN1 dihubungkan ke pin GPIO 14.



Gambar 5.12 Blok Diagram Sub Sistem Relay

Gambar 5.12 merupakan gambar blok diagram dari sub sistem relay. Sub sistem tersebut akan mendapat input dari hasil subscribe ke MQTT Broker.

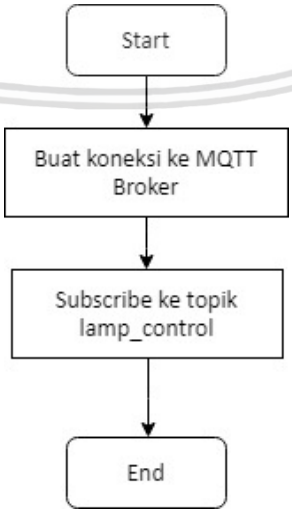
Selain penentuan penempatan relay ke pin GPIO, dilakukan juga perancangan kode program pada sub sistem relay. Gambar 5.13 memperlihatkan alur dari program pada sub sistem relay.



Gambar 5.13 Alur Kerja Kode Program Pada Sub Sistem Relay

5.1.6 Perancangan Sub Sistem MQTT

Pada tahap perancangan Sub Sistem MQTT dilakukan penentuan library MQTT yang nantinya akan dipakai. Untuk sub sistem ini library yang akan dipakai adalah MQTT.js pada Bahasa pemrograman nodejs. MQTT ini nantinya akan digunakan untuk mekansime komunikasi antara sistem dengan MQTT Broker dimana sistem akan men-subscribe suatu topik yang telah ditentukan yaitu **lamp_control**. Alur kerja dari sub sistem ini dapat dilihat pada Gambar 5.14 berikut.



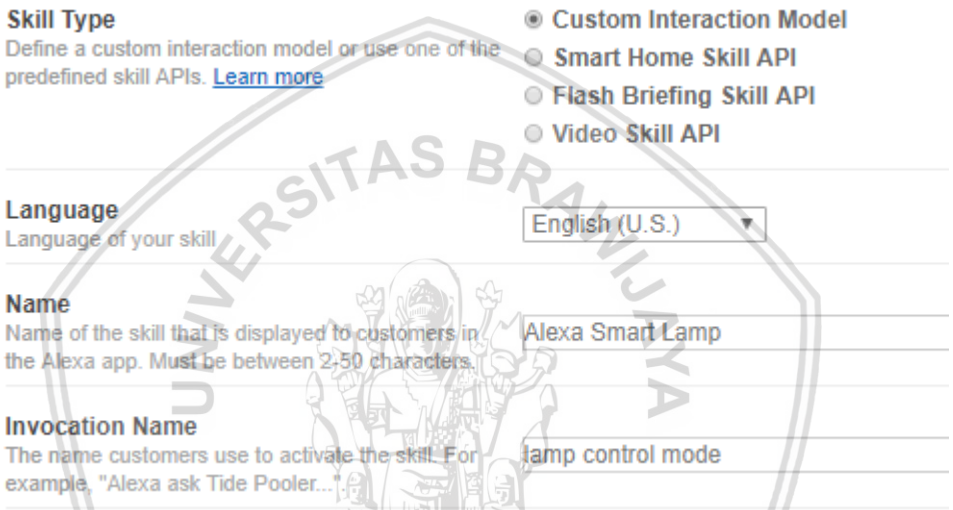
Gambar 5.14 Alur Kerja Sub Sistem MQTT

5.2 Implementasi

5.2.1 Implementasi Alexa Skills Kit

Pada tahap implementasi alexa skills kit akan dijelaskan tahap-tahap pengaturan yang dilakukan supaya alexa skills kit dapat bekerja sesuai dengan yang diharapkan. Implementasi alexa skills kit dilakukan dengan tahapan-tahapan sebagai berikut :

Lakukan login pada halaman developer amazon, lalu arahkan halaman website ke dashboard alexa lalu klik getting started pada alexa skills kit. Lalu halaman website akan diarahkan halaman alexa skills kit. Pilih add a new skill selanjutnya isikan informasi seperti pada Gambar 5.15.



Gambar 5.15 Skill Information Alexa Skills Kit

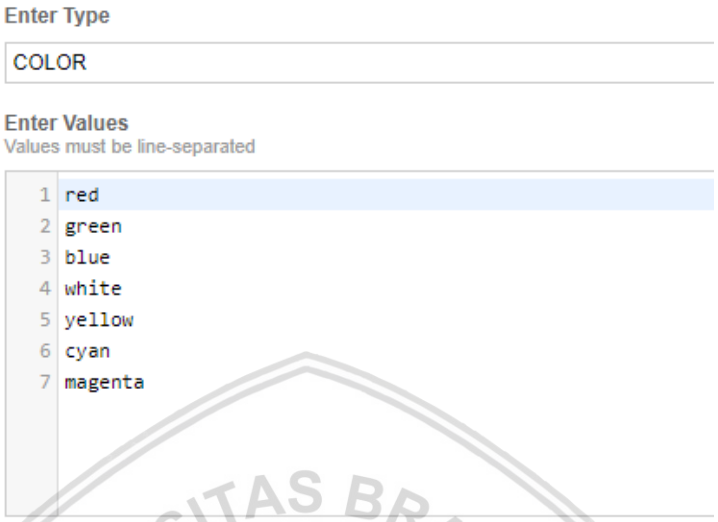
Selanjutnya adalah pendefinisian pada Intent Schema yang telah dibahas pada bagian perancangan. Intent schema merupakan format standar dari alexa skills kit dalam bentuk JSON dimana format tersebut befungsi untuk menerjemahkan request yang dikirim dari alexa voice service. Table 5.1 menunjukan intent schema yang telah di desain untuk sistem.

Tabel 5.1 *Intent Schema*

```
{
  "intents": [
    {
      "intent": "AMAZON.CancelIntent"
    },
    {
      "intent": "AMAZON.HelpIntent"
    },
    {
      "intent": "AMAZON.StopIntent"
    },
    {
      "slots": [
        {
```

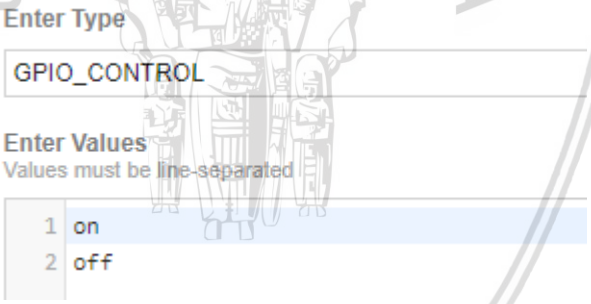
```
            "name": "color",
            "type": "COLOR"
          },
          {
            "name": "location",
            "type": "LOCATION"
          }
        ],
        "intent": "ChangeColorIntent"
      },
      {
        "slots": [
          {
            "name": "percentage",
            "type": "AMAZON.NUMBER"
          },
          {
            "name": "location",
            "type": "LOCATION"
          }
        ],
        "intent": "DimIntent"
      },
      {
        "slots": [
          {
            "name": "status",
            "type": "GPIO_CONTROL"
          },
          {
            "name": "location",
            "type": "LOCATION"
          }
        ],
        "intent": "OnOffIntent"
      },
      {
        "slots": [
          {
            "name": "status",
            "type": "GPIO_CONTROL"
          },
          {
            "name": "time",
            "type": "AMAZON.TIME"
          },
          {
            "name": "location",
            "type": "LOCATION"
          }
        ],
        "intent": "ShceduleIntent"
      }
    ]
}
```

Selanjutnya adalah pendefinisian custom slot sebagai penunjang agar sistem dapat lebih mudah menerjemahkan perintah suara yang berisikan beberapa kata penting.



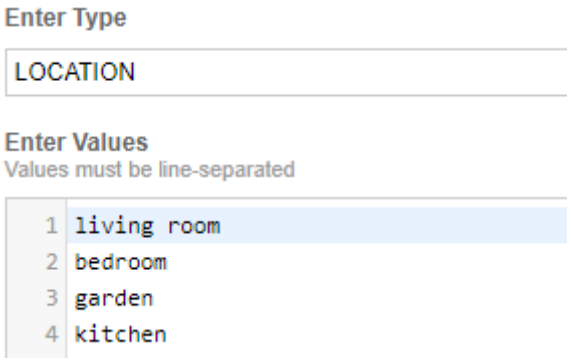
Gambar 5.16 Custom Slot Color

Gambar 5.16 merupakan perancangan untuk custom slot color. Pada sistem ini warna yang dapat diubah LED diantaranya warna *red*, *green*, *blue*, *white*, *yellow*, *cyan* dan *magenta*.



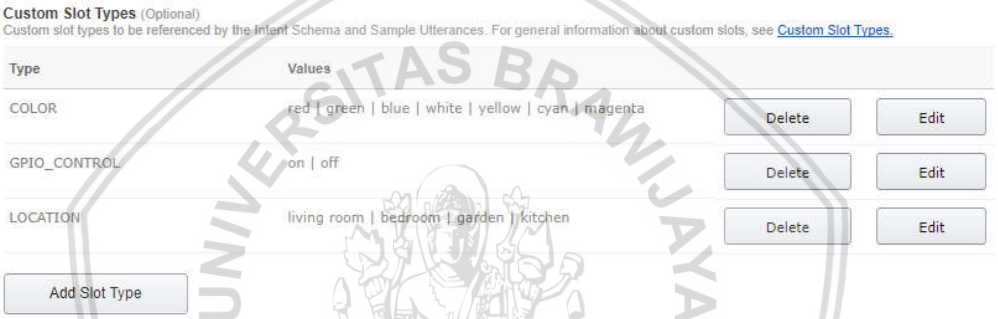
Gambar 5.17 Custom Slot GPIO Control

Gambar 5.17 merupakan perancangan untuk custom slot control. Pada sistem ini kontrol pada lampu terbagi menjadi dua yaitu on untuk menyalakan dan off untuk mematikan.



Gambar 5.18 Custom Slot Location

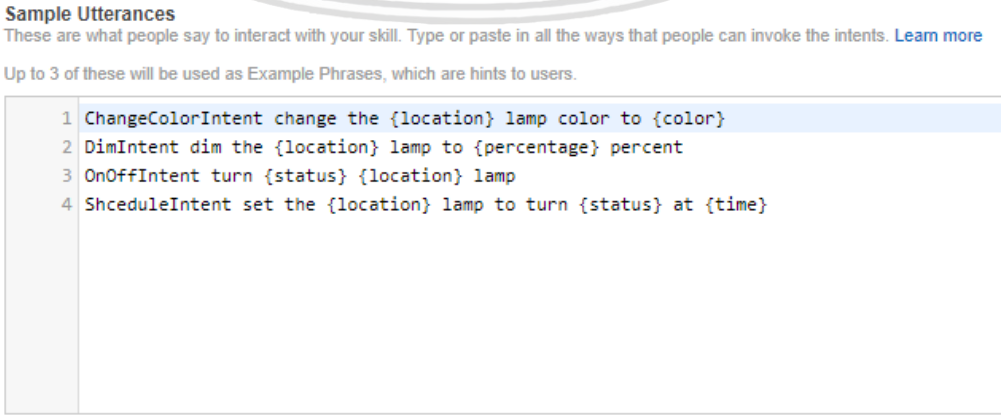
Gambar 5.18 merupakan perancangan untuk custom slot location. Pada sistem ini lokasi dari lampu diantaranya *living room*, *bedroom*, *garden* dan *kitchen*.



Gambar 5.19 Hasil Akhir Setelah Custom Slot

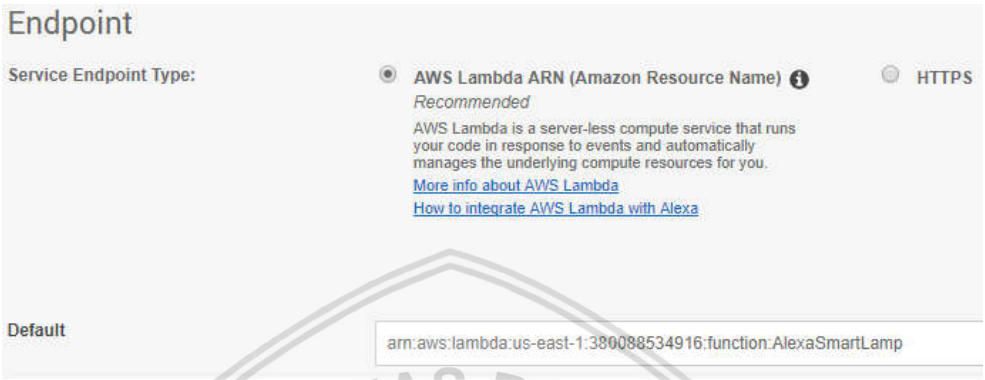
Gambar 5.19 merupakan tampilan hasil ketika seluruh custom slot telah di definisikan.

Selanjutnya adalah pada bagian sample utterances. Bagian tersebut adalah pendefinisian contoh pengucapan yang dikaitkan dengan intent schema yang sebelumnya dibuat. Contoh-contoh pengucapan yang didefinisikan dapat dilihat pada Gambar 5.20 berikut.



Gambar 5.20 Sample Utterances

Selanjutnya adalah pada bagian configuration. Pada bagian ini didefinisikan service end point dari alexa skills kit ini. Service end point merupakan tempat dimana output berupa JSON dari alexa skills kit di proses, dalam hal ini output tersebut nantinya akan di proses di Amazon Web Service Lambda maka dari itu alamat dari service end point diarahkan ke alamat Amazon Web Service Lambda yang dapatkan ketika pengimplementasian Amazon Web Service Lambda.

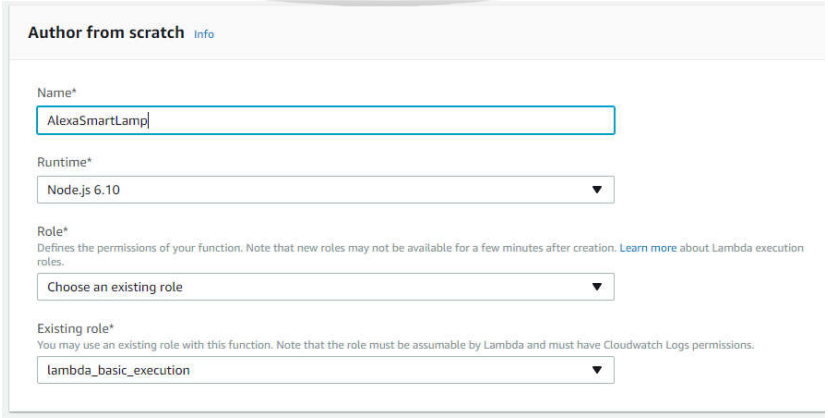


Gambar 5.21 Pendefinisian Service End Point ke Alamat AWS Lambda

5.2.2 Implementasi Amazon Web Service Lamda

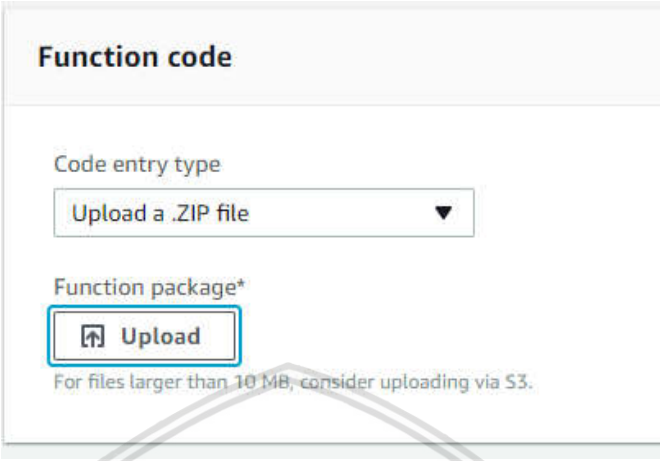
Pada pengimplementasian Amazon Web Service Lambda dilakukan pembuatan kode program berbasis Node.js. Kode program tersebut digunakan untuk menangani request perintah suara. Nantinya request tersebut akan di format ke bentuk yang dikenali sub sistem LED dan Relay sehingga ketika sub sistem MQTT men-subscribe data yang telah di publish melalui Amazon Web Service Lambda sub sistem LED dan Relay dapat menentukan aksi apa yang harus dilakukan.

Ketika kode program telah selesai dibuat kompres kode program tersebut menjadi zip dan upload kode program tersebut ke Amazon Web Service Lambda dengan cara buka alamat https://console.aws.amazon.com pilih Lambda, lalu pilih create new function. Maka akan ditampilkan halaman seperti Gambar 5.45. Isikan data seperti gambar tersebut.



Gambar 5.22 Konfigurasi Amazon Web Service Lambda 1

Selanjutnya klik create function pada pojok kanan bawah halaman website. Lalu halaman website akan diarahkan ke halaman selanjutnya. Pada halaman tersebut lakukan upload program dengan pilihan seperti Gambar 5.24.

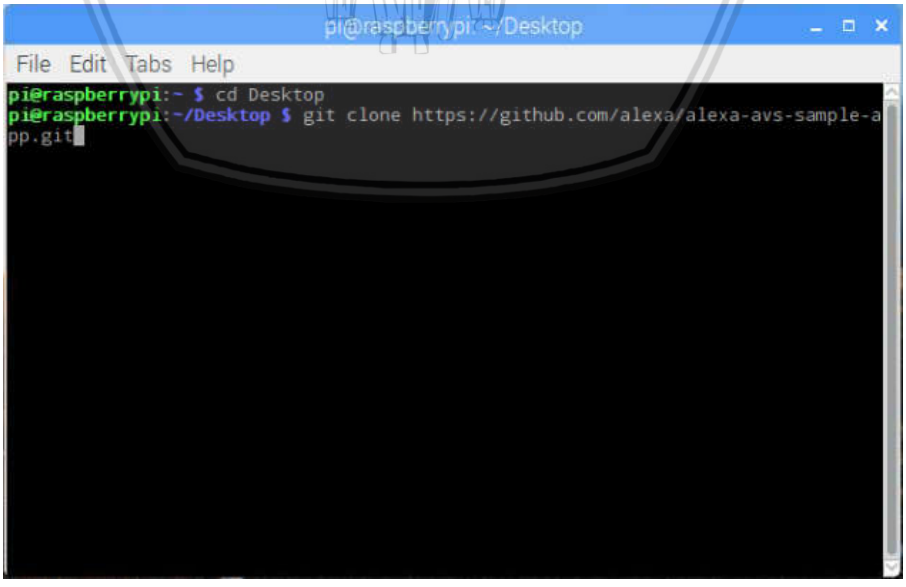


Gambar 5.23 Upload Kode Program

5.2.3 Implementasi Sub Sistem Mikrofon (Modul *Speech Recognition*)

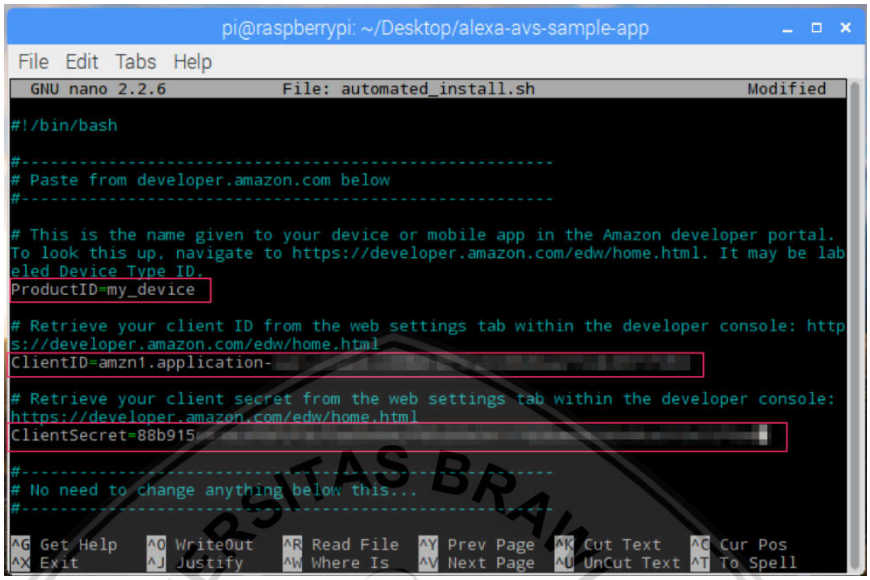
Pada bagian implementasi sub sistem mikrofon dilakukan dua hal yaitu pemasangan mikrofon dan pemasangan service alexa paada perangkat raspberry pi. Pemasangan mikrofon cukup dengan menghubungkan mini usb mikrofon ke port usb yang terdapat pada perangkat raspberry pi. Untuk pemasang service alexa dapat dilihat melalui tahap-tahap berikut.

Tahap awal pemasangan service alexa pada perangkat raspberry pi adalah dengan membuka terminal pada perangkat raspberry dan mengetikan perintah seperti Gambar 5.25.



Gambar 5.24 Cloning Alexa AVS dari Github

Setelah proses cloning selesai lakukan konfigurasi pada script instalasi. Bagian yang dikonfigurasi antara lain **Product ID**, **Client ID** dan **ClientSecret.** Tetap pada terminal yang sama, jalankan perintah **cd ~/Desktop/alexa-avs-sample-app** lalu **nano automated_install.sh**. Akan muncul tampilan seperti Gambar 5.26.



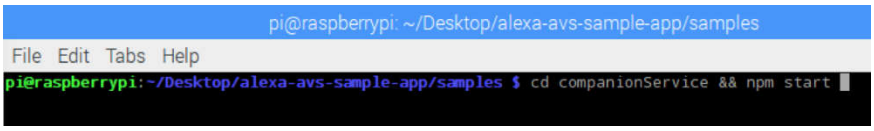
Gambar 5.25 Konfigurasi Script Instalasi

Selanjutnya setelah script telah selesai di konfigurasi jalankan script dengan perintah **. automated_install.sh**. Tunggu hingga proses instalasi selesai. Setelah proses instalasi selesai hal selanjutnya yang harus dilakukan adalah menjalankan web service, sample app dan wake word engine. Fungsi dari web service adalah untuk melakukan autorisasi dengan cloud alexa, sample app untuk berkomunikasi ke alexa voice service dan wake word engine supaya sistem dapat memulai interaksi dengan menggunakan nama sebutan tertentu untuk sistem ini adalah alexa.

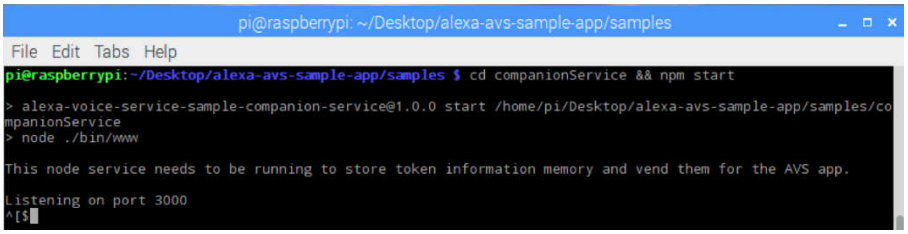
Untuk menjalankan web service buka satu terminal baru lalu jalankan perintah berikut :

cd ~/Desktop/alexa-avs-sample-app/samples
cd companionService && npm start

Maka tampilan terminal tersebut akan terlihat seperti Gambar 5.49 dan Gambar 5.27.



Gambar 5.26 Web Service Bagian 1

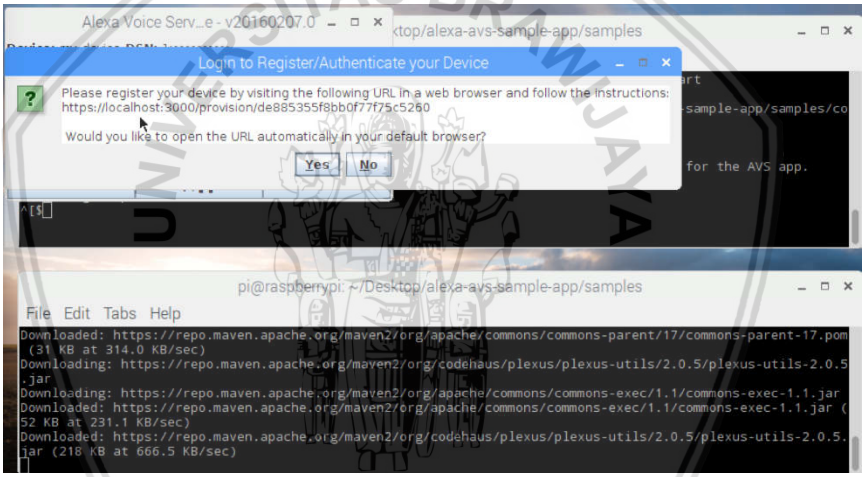


Gambar 5.27 Web Service Bagian 2

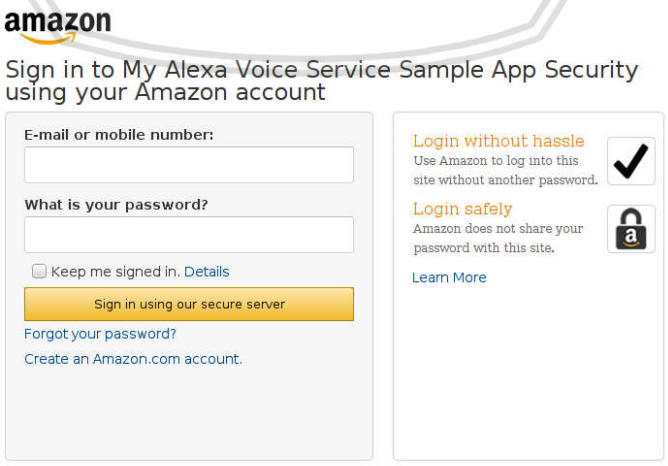
Selanjutnya buka terminal kedua, pada terminal ini akan dijalankan sample app dengan perintah sebagai berikut :

cd ~/Desktop/alexa-avs-sample-app/samples
cd javaclient && mvn exec:exec

Maka tampilan terminal tersebut akan terlihat seperti Gambar 5.29. Selanjutnya ketika ditekan tombol yes maka halaman sebuah halaman web browser untuk login ke akun amazon akan terbuka. Lakukan login supaya sistem dapat terhubung ke alexa voice service seperti pada Gambar 5.30.

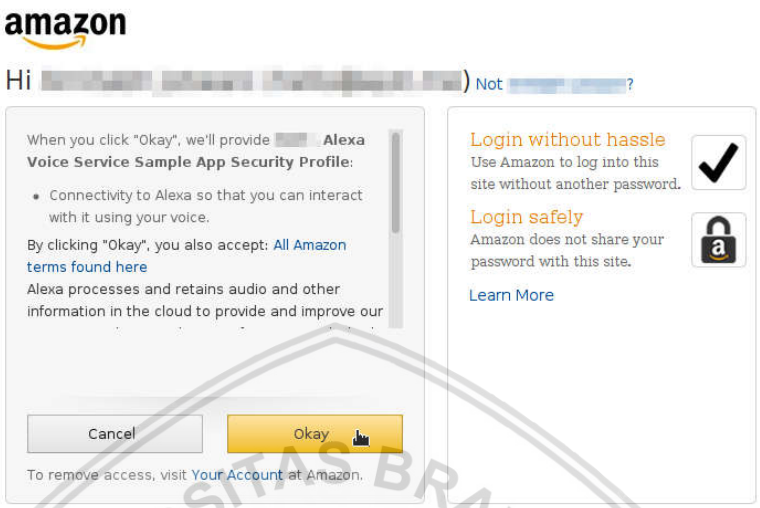


Gambar 5.28 Sample App Bagian 1

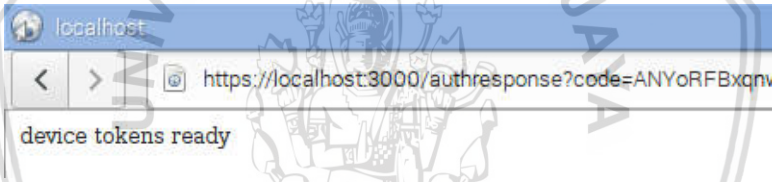


Gambar 5.29 Login Akun Amazon

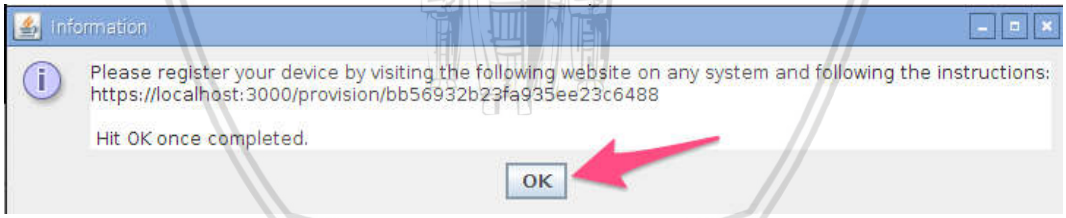
Setelah login akan muncul halaman konfirmasi autentikasi pilih Okay seperti Gambar 5.31, lalu tampilan browser akan terlihat seperti Gambar 5.32. Selanjutnya tekan OK seperti Gambar 5.33, maka sistem akan siap menerima input suara yang nantinya akan diteruskan ke Alexa Voice Service.



Gambar 5.30 Konfirmasi Autentikasi



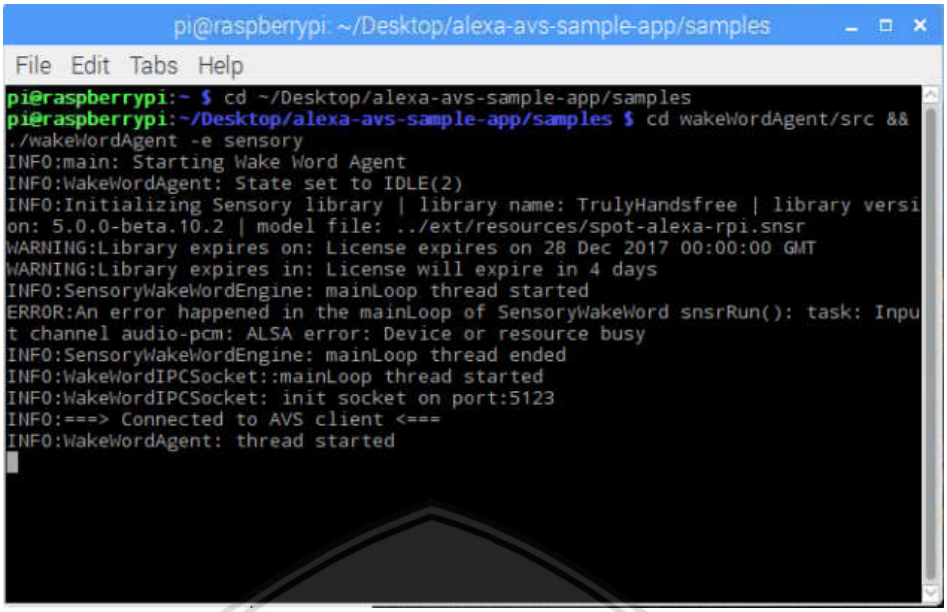
Gambar 5.31 Tampilan Browser Setelah Konfirmasi Autentikasi



Gambar 5.32 Konfirmasi Registrasi Sukses

Selanjutnya adalah bagian wake word engine, seperti tahap sebelumnya, buka satu terminal baru lalu jalankan perintah berikut seperti Gambar 5.34

```
cd ~/Desktop/alexa-avs-sample-app/samples
cd wakeWordAgent/src && ./wakeWordAgent -e sensory
```



Gambar 5.33 Wake Word Engine

Setelah ketiga hal tersebut dijalankan maka sistem akan siap menerima perintah suara yang akan diteruskan ke alexa voice service.

5.2.4 Implementasi Sub Sistem LED

Pada implementasi sub sistem LED dilakukan dua hal yaitu melakukan pengkoneksian LED ke GPIO pin raspberry melalui project board dan membuat kode program supaya LED dapat berfungsi sebagaimana mestinya. Kode program dibuat menggunakan Node.js. Pada tahap implementasi ini akan diperlihatkan potongan potongan kode program pada tabel berdasarkan fungsinya sebagai berikut :

Tabel 5.2 Pendefinisian Library dan Variable Pendukung Sub Sistem

```
1   var Gpio = require('pigpio').Gpio;
2   var mqtt = require('mqtt');
3
4   var r = {};
5   var g = {};
6   var b = {};
7
8   r['garden'] = new Gpio(2, {mode: Gpio.OUTPUT});
9   g['garden'] = new Gpio(3, {mode: Gpio.OUTPUT});
10  b['garden'] = new Gpio(4, {mode: Gpio.OUTPUT});
11
12  r['living room'] = new Gpio(25, {mode: Gpio.OUTPUT});
13  g['living room'] = new Gpio(8, {mode: Gpio.OUTPUT});
14  b['living room'] = new Gpio(7, {mode: Gpio.OUTPUT});
15
16  r['bedroom'] = new Gpio(13, {mode: Gpio.OUTPUT});
17  g['bedroom'] = new Gpio(19, {mode: Gpio.OUTPUT});
18  b['bedroom'] = new Gpio(26, {mode: Gpio.OUTPUT});
19
20  var lamp_color = {};
21  var lamp_state = {};
```

Pada Table 5.2 merupakan potongan program pendefinisian library dan variable pendukung sub sistem. Library yang digunakan adalah pigpio dan mqtt. Library pigpio berfungsi supaya pin GPIO pada raspberry pi dapat diatur sedemikian rupa seperti menatur nilai digitalnya dan pengaturan PWM. Sedangkan library mqtt berfungsi sebagai penghubung sub sistem ke MQTT broker.

Tabel 5.3 Pendefinisian Kondisi Awal Sub Sistem

```
1   lamp_color['garden'] = 'white';
2   lamp_state['garden'] = 'of';
3
4   lamp_color['living room'] = 'white';
5   lamp_state['living room'] = 'of';
6
7   lamp_color['bedroom'] = 'white';
8   lamp_state['bedroom'] = 'of';
9
10  r['garden'].digitalWrite(1);
11  g['garden'].digitalWrite(1);
12  b['garden'].digitalWrite(1);
13
14  r['living room'].digitalWrite(1);
15  g['living room'].digitalWrite(1);
16  b['living room'].digitalWrite(1);
17
18  r['bedroom'].digitalWrite(1);
19  g['bedroom'].digitalWrite(1);
20  b['bedroom'].digitalWrite(1);
21
22
23  var timeInterval = new Array();
24  var timeOuts = new Array();
25
26  var start_dutyCycle = 0;
```

Pada Table 5.3 merupakan potongan program dimana kondisi awal sistem di definisikan, yaitu seluruh LED akan berada pada kondisi mati dan ketika lampu dinyalakan pertama kali warna yang akan dihasilkan adalah warna putih.

Tabel 5.4 Pendefinisian Koneksi MQTT dan Fungsi Mapping

```
1   var options = {
2         port: '1883',
3         clientId:                'mqttjs_'                +
4   Math.random().toString(16).substr(2, 8),
5         username: 'username',
6         password: 'password',
7   };
8
9   var    client    =    mqtt.connect('mqtt://irfanreza.tk',
10  options);
11  client.on('connect', function() {
12        client.subscribe('lamp_control');
13  });
14
```

```
client.on('message', function(topic, message) {
      msg_toStr = message.toString('utf8');
      msg_JSON = JSON.parse(msg_toStr);
      console.log(msg_JSON);
      console.log(msg_JSON.action);

      if (msg_JSON.action == 1) {
            on_off(msg_JSON.status, msg_JSON.location);
      } else if (msg_JSON.action == 2) {
            on_intensity(lamp_color[msg_JSON.location],
msg_JSON.percentage, msg_JSON.location)
      } else if (msg_JSON.action == 3) {
            change_color(msg_JSON.color,
msg_JSON.location)
      } else if (msg_JSON.action == 4) {
            on_schedule(msg_JSON.time,  msg_JSON.status,
msg_JSON.location);
      }

});

function map_range(value, low1, high1, low2, high2) {
    return low2 + (high2 - low2) * (value - low1) /
(high1 - low1);
}
```

Pada Table 5.4 merupakan potongan program dimana dilakukan pendefinisian koneksi MQTT berupa alamat broker tujuan subscribe dan username serta password. Selanjutnya terdapat sebuah fungsi event berupa on message dimana ketika subscribe terjadi maka akan dilakukan pengecekan message action dimana ketika action bernilai 1 maka fungsi mematikan atau menyalakan lampu akan dipanggil, ketika bernilai 2 maka fungsi pengaturan intensitas akan dipanggil, ketika bernilai 3 maka fungsi merubah warna akan dipanggil dan ketika bernilai 4 maka fungsi penjadwalan akan dipanggil. Selain itu terdapat fungsi mapping dimana hasil nilainya akan digunakan untuk pengaturan intensitas berbasis PWM. Perintah suara akan mengirimkan nilai persentase nya maka dari itu dibutuhkan mapping nilai supaya menjadi nilai yang dapat dikenali PWM yaitu 0 hingga 255.

Tabel 5.5 Fungsi On Off

```
function on_off(status, location) {
      clearInterval(timeInterval[location]);
      if (status == 'on') {
            r[location].digitalWrite(0);
            g[location].digitalWrite(0);
            b[location].digitalWrite(0);

            lamp_color[location] = 'white';
            lamp_state[location] = 'on';
            console.log("Lampu Nyala");
      } else {
            r[location].digitalWrite(1);
            g[location].digitalWrite(1);
            b[location].digitalWrite(1);

```

```
16              lamp_color[location] = 'white';
17              lamp_state[location] = 'off';
18              console.log("Mati");
19        }
20  }
```

Pada Table 5.5 didefinisikan fungsi bernama on_off. Fungsi tersebut berfungsi menangani perintah suara dengan tujuan mematikan atau menyalakan LED. Parameter yang digunakan adalah lokasi dan status yang diinginkan yaitu on atau off.

Tabel 5.6 Fungsi On Intensity

```
1   function on_intensity(last_color, percentage, location)
2   {
3         clearInterval(timeInterval[location]);
4         var r_duty = 0;
5         var g_duty = 0;
6         var b_duty = 0;
7         var r_dutyLimit = 0;
8         var g_dutyLimit = 0;
9         var b_dutyLimit = 0;
10        var dutyValue = Math.floor(map_range(percentage,
11  0, 100, 255, 0));
12        console.log("Ganti Intensitas");
13        console.log("Duty      Value      =      ",
14  Math.floor(map_range(percentage, 0, 100, 255, 0)));
15        if (last_color == 'white') {
16              r_dutyLimit = dutyValue;
17              g_dutyLimit = dutyValue;
18              b_dutyLimit = dutyValue;
19              console.log('Last color = ', last_color);
20              console.log('Percentage = ', percentage);
21              console.log('location = ', location);
22        } else if (last_color == 'red') {
23              r_dutyLimit = dutyValue;
24              g_dutyLimit = 255;
25              b_dutyLimit = 255;
26              console.log('Last color = ', last_color);
27              console.log('Percentage = ', percentage);
28              console.log('location = ', location);
29        } else if (last_color == 'green') {
30              r_dutyLimit = 255;
31              g_dutyLimit = dutyValue;
32              b_dutyLimit = 255;
33              console.log('Last color = ', last_color);
34              console.log('Percentage = ', percentage);
35              console.log('location = ', location);
36        } else if (last_color == 'blue') {
37              r_dutyLimit = 255;
38              g_dutyLimit = 255;
39              b_dutyLimit = dutyValue;
40              console.log('Last color = ', last_color);
41              console.log('Percentage = ', percentage);
42              console.log('location = ', location);
43        } else if (last_color == 'yellow') {
44              r_dutyLimit = dutyValue;
```

```
45              g_dutyLimit = dutyValue;
46              b_dutyLimit = 255;
47              console.log('Last color = ', last_color);
48              console.log('Percentage = ', percentage);
49              console.log('location = ', location);
50        } else if (last_color == 'cyan') {
51              r_dutyLimit = 255;
52              g_dutyLimit = dutyValue;
53              b_dutyLimit = dutyValue;
54              console.log('Last color = ', last_color);
55              console.log('Percentage = ', percentage);
56              console.log('location = ', location);
57        } else if (last_color == 'magenta') {
58              r_dutyLimit = dutyValue;
59              g_dutyLimit = 255;
60              b_dutyLimit = dutyValue;
61              console.log('Last color = ', last_color);
62              console.log('Percentage = ', percentage);
63              console.log('location = ', location);
64        }
65        timeInterval[location] = setInterval(function () {
66
67              if (r_duty < r_dutyLimit && r_dutyLimit !=
68    255) {
69                    r_duty += 5;
70              } else {
71                    r_duty = r_dutyLimit;
72              }
73
74              if (g_duty < g_dutyLimit && g_dutyLimit !=
75    255) {
76                    g_duty +=5;
77              } else {
78                    g_duty = g_dutyLimit;
79              }
80
81              if (b_duty < b_dutyLimit && b_dutyLimit !=
82    255) {
83                    b_duty += 5;
84              } else {
85                    b_duty = b_dutyLimit;
86              }
87
88              r[location].pwmWrite(r_duty);
89              g[location].pwmWrite(g_duty);
90              b[location].pwmWrite(b_duty);
91        }, 20);
92  }
```

Pada Table 5.6 didefinisikan fungsi bernama on_intensity. Fungsi tersebut berfungsi menangani perintah suara dengan tujuan mengatur intensitasi LED. Parameter yang digunakan adalah lokasi dan persentase yang diinginkan. Pada fungsi ini dilakukan pengecekan kondisi lampu terakhir sebagai parameter acuan mengubah nilai dari setiap pin GPIO yang terhubung dengan LED.

Tabel 5.7 Fungsi Change Color

```
1   function change_color(color, location) {
2         clearInterval(timeInterval[location]);
3         if (color == 'white') {
4               console.log('Color   before   change   :   ',
5   lamp_color[location]);
6               r[location].digitalWrite(0);
7               g[location].digitalWrite(0);
8               b[location].digitalWrite(0);
9               lamp_color[location] = color;
10              console.log('Color   after   change   :   ',
11  lamp_color[location]);
12        } else if (color == 'red') {
13              console.log('Color   before   change   :   ',
14  lamp_color[location]);
15              r[location].digitalWrite(0);
16              g[location].digitalWrite(1);
17              b[location].digitalWrite(1);
18              lamp_color[location] = color;
19              console.log('Color   after   change   :   ',
20  lamp_color[location]);
21        } else if (color == 'green') {
22              console.log('Color   before   change   :   ',
23  lamp_color[location]);
24              r[location].digitalWrite(1);
25              g[location].digitalWrite(0);
26              b[location].digitalWrite(1);
27              lamp_color[location] = color;
28              console.log('Color   after   change   :   ',
29  lamp_color[location]);
30        } else if (color == 'blue') {
31              console.log('Color   before   change   :   ',
32  lamp_color[location]);
33              r[location].digitalWrite(1);
34              g[location].digitalWrite(1);
35              b[location].digitalWrite(0);
36              lamp_color[location] = color;
37              console.log('Color   after   change   :   ',
38  lamp_color[location]);
39        } else if (color == 'yellow') {
40              console.log('Color   before   change   :   ',
41  lamp_color[location]);
42              r[location].digitalWrite(0);
43              g[location].digitalWrite(0);
44              b[location].digitalWrite(1);
45              lamp_color[location] = color;
46              console.log('Color   after   change   :   ',
47  lamp_color[location]);
48        } else if (color == 'cyan') {
49              console.log('Color   before   change   :   ',
50  lamp_color[location]);
51              r[location].digitalWrite(1);
52              g[location].digitalWrite(0);
53              b[location].digitalWrite(0);
54              lamp_color[location] = color;
55              console.log('Color   after   change   :   ',
56  lamp_color[location]);
57        } else if (color == 'magenta') {
```


repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

```
58              console.log('Color before change : ',
59      lamp_color[location]);
60              r[location].digitalWrite(0);
61              g[location].digitalWrite(1);
62              b[location].digitalWrite(0);
63              lamp_color[location] = color;
64              console.log('Color after change : ',
65      lamp_color[location]);
66          }
67      }
```

Pada Table 5.7 didefinisikan fungsi bernama change color. Fungsi tersebut berfungsi menangani perintah suara dengan tujuan mengubah warna dari LED. Parameter yang digunakan adalah lokasi dan warna yang diinginkan. Pada fungsi ini dilakukan manipulasi nilai digital dari setiap pin LED yang terhubung ke GPIO pin sehingga dapat menghasilkan warna sesuai dengan perintah suara.

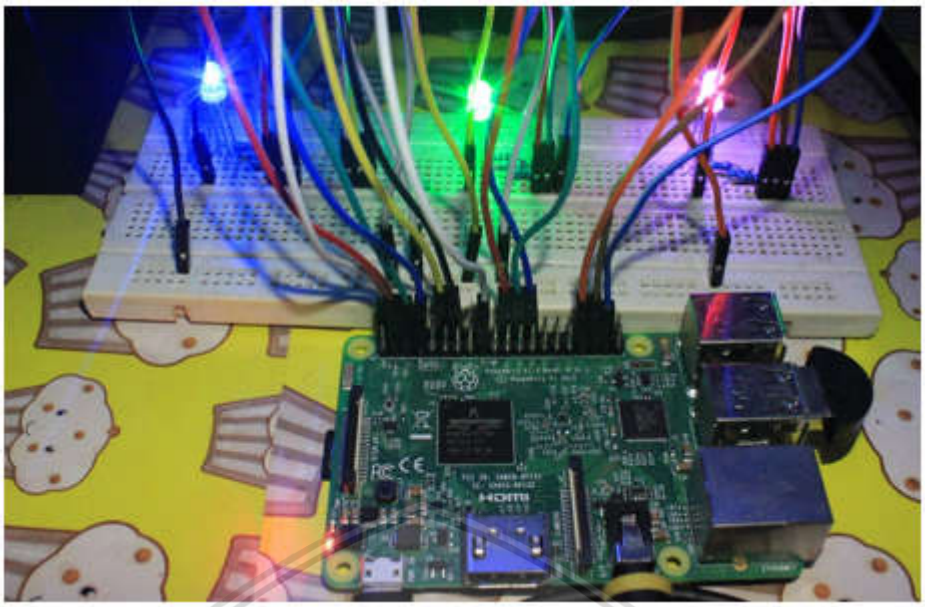
Tabel 5.8 Fungsi On Schedule

```
1       function on_schedule(time, status, location) {
2           clearTimeout(timeOuts[location]);
3           var timeSplit = time.toString().split(":");
4           console.log(timeSplit[0]);
5           console.log(timeSplit[1]);
6           var now = new Date();
7           var millisTimeOut = new Date(now.getFullYear(),
8       now.getMonth(),     now.getDate(),     timeSplit[0],
9       timeSplit[1], 0, 0) - now;
10
11          if (millisTimeOut < 0) {
12              millisTimeOut += 86400000;
13          }
14
15          console.log("Scheduling ", location, " lamp to
16      turn ", status, " at ", time);
17          console.log("Milis dibutuhkan : ", millisTimeOut);
18
19          timeOuts[location] = setTimeout(function() {
20              if (status == 'on') {
21                  r[location].digitalWrite(0);
22                  g[location].digitalWrite(0);
23                  b[location].digitalWrite(0);
24
25                  lamp_color[location] = 'white';
26                  lamp_state[location] = 'on';
27              } else if (status == 'off') {
28                  r[location].digitalWrite(1);
29                  g[location].digitalWrite(1);
30                  b[location].digitalWrite(1);
31
32                  lamp_color[location] = 'white';
33                  lamp_state[location] = 'off';
34              }
35          }, millisTimeOut);
36      }
```

Pada Table 5.8 didefinisikan fungsi bernama on_schedule. Fungsi tersebut bertugas menangani perintah suara dengan tujuan melaukan penjadwalan pada LED. Parameter yang digunakan adalah lokasi, status, dan waktu. Pada fungsi nantinya kode di atas akan membuat timer dengan fasilitas timeout dari Node.js sehingga LED dapat dimatikan atau dinyalakan pada waktu tertentu.

Selanjutnya setelah semua fungsi berhasil di definisikan maka dilakukan instalasi library pendukung supaya kode program tersebut dapat berjalan. Pada implementasi sub sistem ini dilakukan pemasangan library pigpio untuk melakukan manipulasi pada GPIO pin raspberrypi. Caranya adalah dengan mengetikan perintah **sudo apt-get install pigpio** seperti pada Gambar 5.35 berikut.

Gambar 5.34 Instalasi Library pigpio



Gambar 5.35 Pengkoneksian LED dengan GPIO pin

5.2.5 Implementasi Sub Sistem Relay

Pada implementasi sub sistem relay dilakukan dua hal yaitu melakukan pengkoneksian relay ke GPIO pin raspberry melalui project board dan membuat kode program supaya relay dapat berfungsi sebagaimana mestinya.

Tabel 5.9 Penyisipan Kode Pada Pendefinisian Variable

```
1   var Gpio = require('pigpio').Gpio;
2   var mqtt = require('mqtt');
3
4   var r = {};
5   var g = {};
6   var b = {};
7
8   var kitchen = new Gpio(14, {mode: Gpio.OUTPUT});
9
10  r['garden'] = new Gpio(2, {mode: Gpio.OUTPUT});
11  g['garden'] = new Gpio(3, {mode: Gpio.OUTPUT});
12  b['garden'] = new Gpio(4, {mode: Gpio.OUTPUT});
13
14  r['living room'] = new Gpio(25, {mode: Gpio.OUTPUT});
15  g['living room'] = new Gpio(8, {mode: Gpio.OUTPUT});
16  b['living room'] = new Gpio(7, {mode: Gpio.OUTPUT});
17
18  r['bedroom'] = new Gpio(13, {mode: Gpio.OUTPUT});
19  g['bedroom'] = new Gpio(19, {mode: Gpio.OUTPUT});
20  b['bedroom'] = new Gpio(26, {mode: Gpio.OUTPUT});
21
22  var lamp_color = {};
23  var lamp_state = {};
```

Pada Table 5.9 merupakan potongan program pendefinisian library dan variable pendukung sub sistem. Library yang digunakan adalah pigpio dan mqtt. Library pigpio berfungsi supaya pin GPIO pada raspberry pi dapat diatur sedemikian rupa seperti menatur nilai digitalnya dan pengaturan PWM.

Sedangkan library mqtt berfungsi sebagai penghubung sub sistem ke MQTT broker.

Tabel 5.10 Penyisipan Kode Pada Kondisi Awal Sub Sistem

```
1   lamp_state['kitchen'] = 'off';
2
3   lamp_color['garden'] = 'white';
4   lamp_state['garden'] = 'of';
5
6   lamp_color['living room'] = 'white';
7   lamp_state['living room'] = 'of';
8
9   lamp_color['bedroom'] = 'white';
10  lamp_state['bedroom'] = 'of';
11
12  r['garden'].digitalWrite(1);
13  g['garden'].digitalWrite(1);
14  b['garden'].digitalWrite(1);
15
16  r['living room'].digitalWrite(1);
17  g['living room'].digitalWrite(1);
18  b['living room'].digitalWrite(1);
19
20  r['bedroom'].digitalWrite(1);
21  g['bedroom'].digitalWrite(1);
22  b['bedroom'].digitalWrite(1);
23
24
25  var timeInterval = new Array();
26  var timeOuts = new Array();
27
28  var start_dutyCycle = 0;
```

Pada Table 5.10 merupakan potongan program dimana kondisi awal sistem di definisikan, yaitu seluruh LED dan lampu relay akan berada pada kondisi mati dan ketika lampu dinyalakan pertama kali warna yang akan dihasilkan adalah warna putih.

Tabel 5.11 Penambahan Kode Program Pada Fungsi On Off

```
1   function on_off(status, location) {
2         clearInterval(timeInterval[location]);
3
4         if (location != 'kitchen') {
5               if (status == 'on') {
6                     r[location].digitalWrite(0);
7                     g[location].digitalWrite(0);
8                     b[location].digitalWrite(0);
9
10                    lamp_color[location] = 'white';
11                    lamp_state[location] = 'on';
12                    console.log("Lampu Nyala");
13              } else {
14                    r[location].digitalWrite(1);
15                    g[location].digitalWrite(1);
16                    b[location].digitalWrite(1);
```


repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

```
17
18                      lamp_color[location] = 'white';
19                      lamp_state[location] = 'off';
20                      console.log("Mati");
21                }
22          }
23
24          if (location == 'kitchen') {
25                if (status == 'on') {
26                      kitchen.digitalWrite(0);
27                } else {
28                      kitchen.digitalWrite(1);
29                }
30          }
31
32    }
```

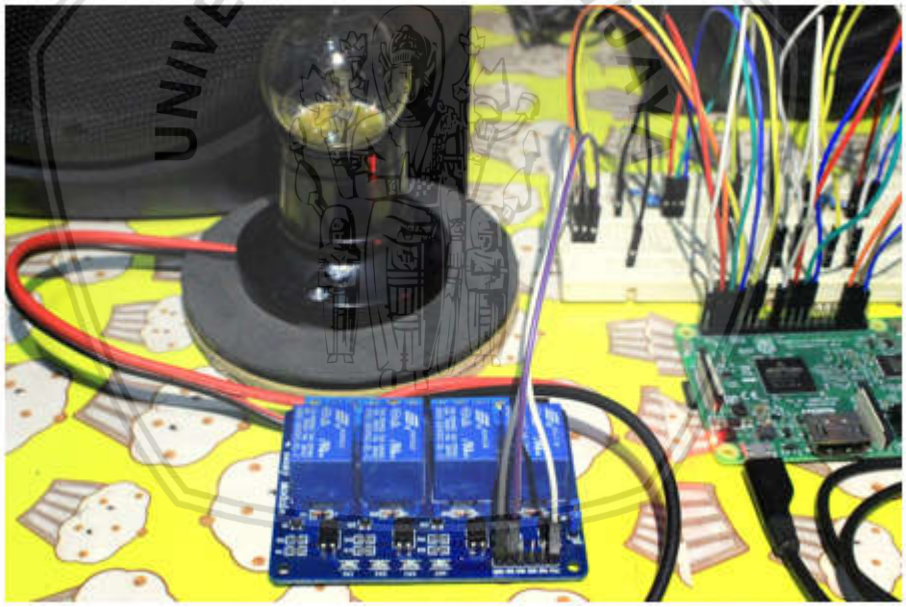
Pada Table 5.11 didefinisikan fungsi bernama on_off. Fungsi tersebut berfungsi menangani perintah suara dengan tujuan mematikan atau menyalakan LED dan Relay. Parameter yang digunakan adalah lokasi dan status yang diinginkan yaitu on atau off.

Tabel 5.12 Penambahan Kode Program Pada Fungsi Schedule

```
1     function on_schedule(time, status, location) {
2           clearTimeout(timeOuts[location]);
3           var timeSplit = time.toString().split(":");
4           console.log(timeSplit[0]);
5           console.log(timeSplit[1]);
6           var now = new Date();
7           var millisTimeOut = new Date(now.getFullYear(),
8     now.getMonth(),      now.getDate(),      timeSplit[0],
9     timeSplit[1], 0, 0) - now;
10
11          if (millisTimeOut < 0) {
12                millisTimeOut += 86400000;
13          }
14
15          console.log("Scheduling ", location, " lamp to
16    turn ", status, " at ", time);
17          console.log("Milis dibutuhkan : ", millisTimeOut);
18
19          timeOuts[location] = setTimeout(function() {
20                if(location != 'kitchen') {
21                      if (status == 'on') {
22                            r[location].digitalWrite(0);
23                            g[location].digitalWrite(0);
24                            b[location].digitalWrite(0);
25
26                            lamp_color[location] = 'white';
27                            lamp_state[location] = 'on';
28                      } else if (status == 'off') {
29                            r[location].digitalWrite(1);
30                            g[location].digitalWrite(1);
31                            b[location].digitalWrite(1);
32
33                            lamp_color[location] = 'white';
```

```
34                                  lamp_state[location] = 'off';
35                          }
36                  }
37
38                  if (location == 'kitchen') {
39                          if (status == 'on') {
40                                  kitchen.digitalWrite(0);
41                          } else {
42                                  kitchen.digitalWrite(1);
43                          }
44                  }
45
46          }, millisTimeOut);
47  }
```

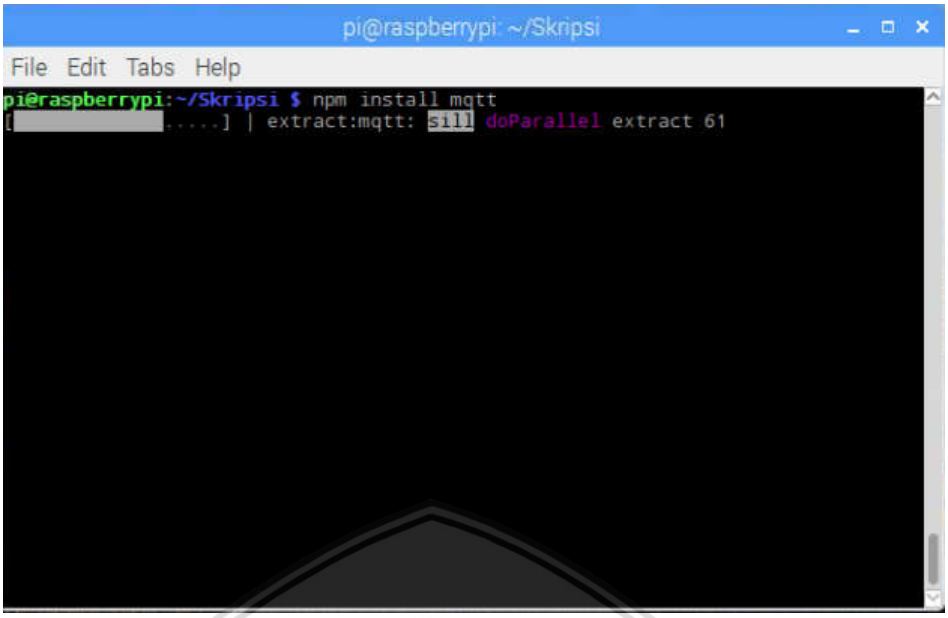
Pada Table 5.12 didefinisikan fungsi bernama on_schedule. Fungsi tersebut bertugas menangani perintah suara dengan tujuan melaukan penjadwalan pada LED dan Relay. Parameter yang digunakan adalah lokasi, status, dan waktu. Pada fungsi nantinya kode di atas akan membuat timer dengan fasilitas timeout dari Node.js sehingga LED atau Relay dapat dimatikan atau dinyalakan pada waktu tertentu.



Gambar 5.36 Sub Sistem Relay

5.2.6 Implementasi Sub Sistem MQTT

Pada implementasi sub sistem MQTT dilakukan penginstallan library mqtt.js yang nantinya akan digunakan pada kode program sub sistem LED dan relay untuk melakukan subscribe data ke MQTT Broker. Tahapan penginstallannya adalah buka terminal raspberry lalu arahkan di direktori tempat kode program disimpan. Selanjutnya jalankan perintah **npm install mqtt**, lalu tunggu beberapa saat seperti Gambar 5.38.



Gambar 5.37 Proses Instalasi Library MQTT.js

Setelah proses instalasi selesai library mqtt.js siap dipakai. Potongan kode program pengoneksian dan subscribe datanya dapat dilihat pada Tabel 5.13 berikut.

Tabel 5.13 Kode Program Sub Sistem MQTT

```
1   var options = {
2         port: '1883',
3         clientId:                    'mqttjs_'                    +
4   Math.random().toString(16).substr(2, 8),
5         username: 'username',
6         password: 'password',
7   };
8
9   var    client    =    mqtt.connect('mqtt://irfanreza.tk',
10  options);
11  client.on('connect', function() {
12        client.subscribe('lamp_control');
13  });
14
15  client.on('message', function(topic, message) {
16
17  });
```

Tabel 5.13 menunjukan potongan kode program untuk melakukan pendefinisian koneksi MQTT. Topik yang akan di subscribe adalah **lamp_control**.

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini membahas mengenai pengujian dan analisis hasil dari implementasi sistem yang telah diterapkan melalui tahapan perancangan dan implementasi. Pengujian dilakukan untuk mengetahui apakah kebutuhan sistem telah terpenuhi. Pengujian dibagi menjadi beberapa tahap sesuai dengan tujuannya agar lebih mudah dalam proses analisa.

6.1 Pengujian Raspberry Sebagai Modul Speech Recognition

6.1.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan perangkat raspberry pi sebagai modul speech recognition. Tingkat keberhasilannya adalah ketika perangkat raspberry dapat merespon ketika diucapkan wake word **alexa** dan invocation name dari skill yang telah di implementasikan pada alexa skills kit yaitu **lamp control mode**.

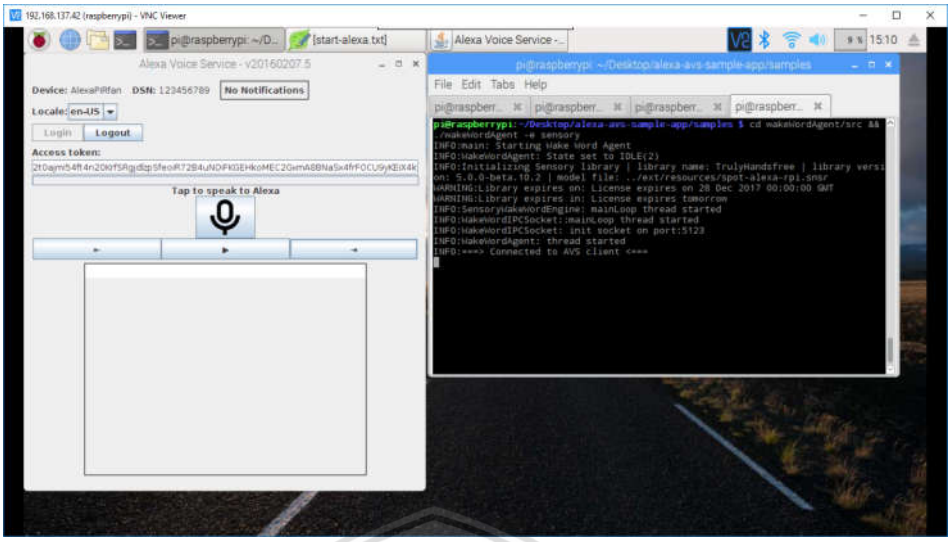
6.1.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.

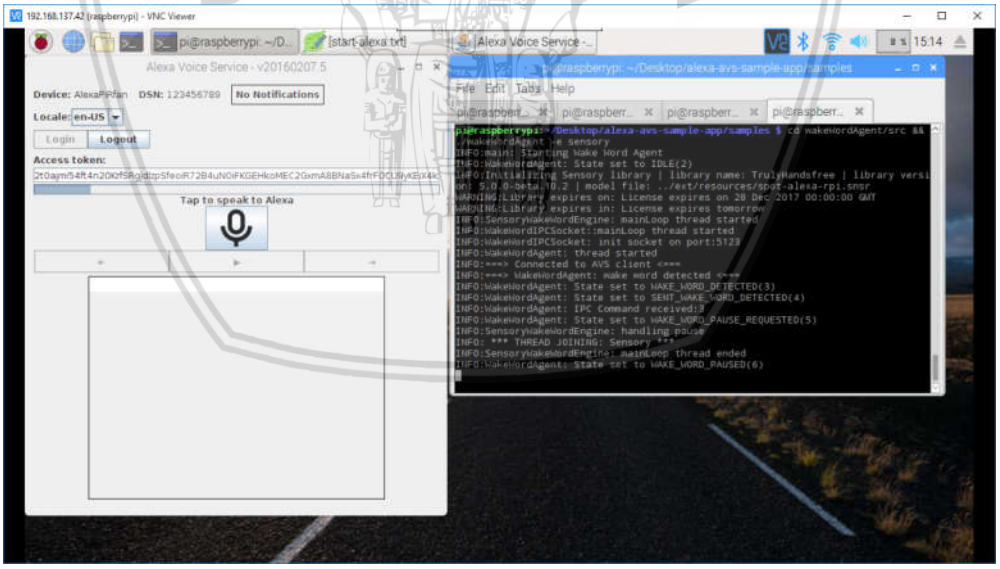
6.1.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan wake word engine dan alexa voice service sample app pada raspberry. Gambar 6.1 merupakan tampilan terminal dan sample app ketika pertama kali dijalankan.



Gambar 6.1 Tampilan Terminal Raspberry dan Sample App

Selanjutnya adalah ketika wake word dan invocation name diucapkan respon dari sistem dapat dilihat pada Gambar 6.2 berikut. Dapat dilihat perubahan pada terminal dan sample app. Pada sample app terlihat indikator di atas gambar speaker bergerak. Hal tersebut menandakan perangkat sedang dalam kondisi listen. Selain itu speaker output akan mengeluarkan suara **Welcome to the lamp control mode, if you new please say help**. Hal tersebut menandapakn bahwa sistem telah terhubung ke alexa skills kit.



Gambar 6.2 Wake Word dan Invocation Name Dikenali

Pada tahap pengujian ini dilakukan pengucapan wake word dan invocation name sebanyak 10 kali dengan hasil yang dapat dilihat pada Tabel 6.1 berikut ini.

Tabel 6.1 Hasil Percobaan Modul Speech Recognition

Percobaan Ke-	Wake Word Terdeteksi	Invocation Name Terdeteksi	Berhasil Terhubung Ke Skill
1	Ya	Ya	Ya
2	Ya	Tidak	Tidak
3	Ya	Ya	Ya
4	Ya	Ya	Ya
5	Ya	Ya	Ya
6	Ya	Ya	Ya
7	Ya	Ya	Ya
8	Ya	Ya	Ya
9	Ya	Ya	Ya
10	Ya	Ya	Ya

6.1.4 Analisa Pengujian

Data dari hasil percobaan menunjukan bahwa sub sistem mikrofon yang bertindak sebagai modul speech recognition dapat berfungsi sebagaimana mestinya yaitu menerima perintah suara tertentu dengan tingkat keberhasilan 90%. Sub sistem tersebut berhasil terhubung ke alexa skills kit melalui alexa voice service sehingga siap menerima perintah selanjutnya.

6.2 Pengujian Menyalakan Dan Mematikan LED

6.2.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem LED dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk mematikan dan menyalakan LED berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai acuan untuk mematikan dan menyalakan LED.

6.2.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.

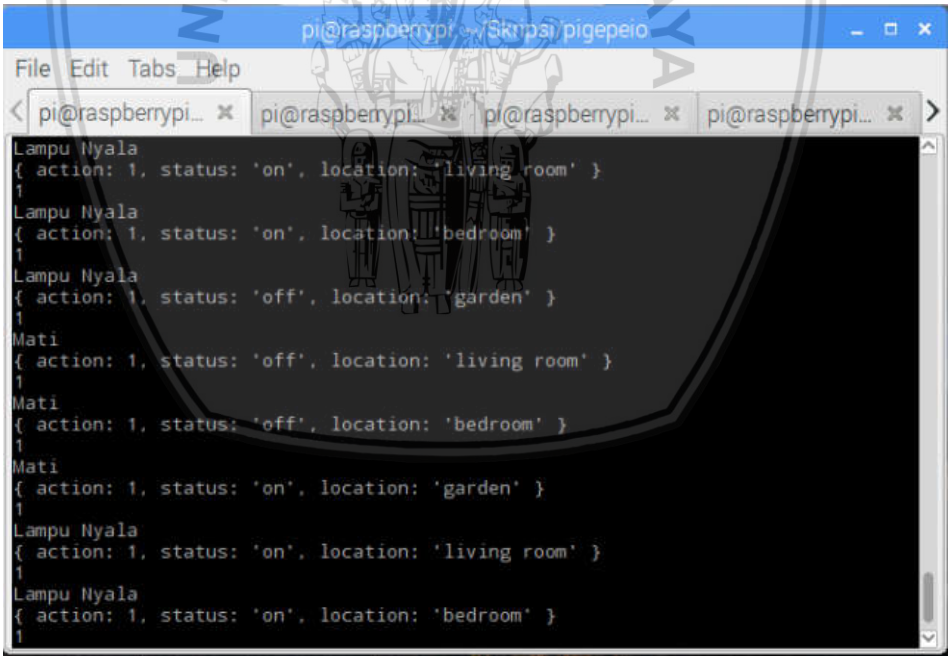
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk mematikan atau menyalakan LED pada lokasi tertentu.

Beberapa perintah suara yang dapat dikenali oleh sistem untuk mematikan atau menyalakan LED adalah sebagai berikut:

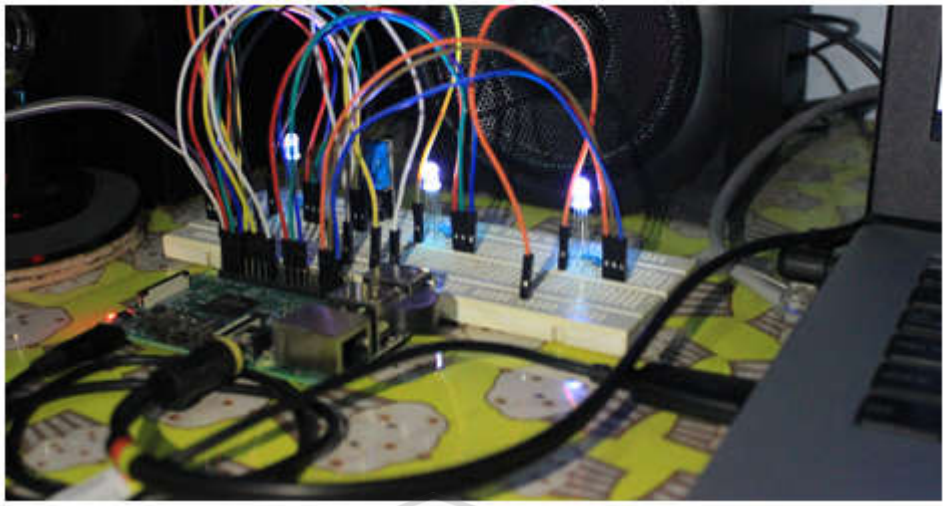
1. Turn on garden lamp.
2. Turn on living room lamp.
3. Turn on bedroom lamp.
4. Turn off garden lamp.
5. Turn off living room lamp.
6. Turn off garden lamp.

6.2.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem LED. Gambar 6.3 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.4 menunjukan LED pada saat menyala.



Gambar 6.3 Hasil Subscribe Untuk Mematikan dan Menyalakan LED



Gambar 6.4 Hasil Ketika Semua LED Dalam Keadaan On

Pada tahap pengujian ini dilakukan percobaan untuk menyalakan dan mematikan LED masing masing sebanyak 5 kali untuk setiap lokasi, maka akan ada total 15 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.2

Tabel 6.2 Hasil Percobaan Menyalakan dan Mematikan LED

Percobaan Ke-	Action	Location	LED On/Off
1	On	Garden	On
2	On	Living room	On
3	On	Bedroom	On
4	Off	Garden	Off
5	Off	Living room	Off
6	Off	Bedroom	Off
7	On	Garden	On
8	On	Living room	On
9	On	Bedroom	On
10	Off	Garden	Off
11	Off	Living room	Off
12	Off	Bedroom	Off
13	On	Garden	On
14	On	Living room	On
15	On	Bedroom	On

6.2.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem led dapat berfungsi mematikan dan menyalakan LED dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.2 yatu kesesuaian antara action yang diucapkan dan keadaan LED setelahnya.

6.3 Pengujian Menyalakan Dan Mematikan Lampu Pada Relay

6.3.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem relay dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk mematikan dan menyalakan lampu pada relay berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai acuan untuk mematikan dan menyalakan lampu pada relay.

6.3.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

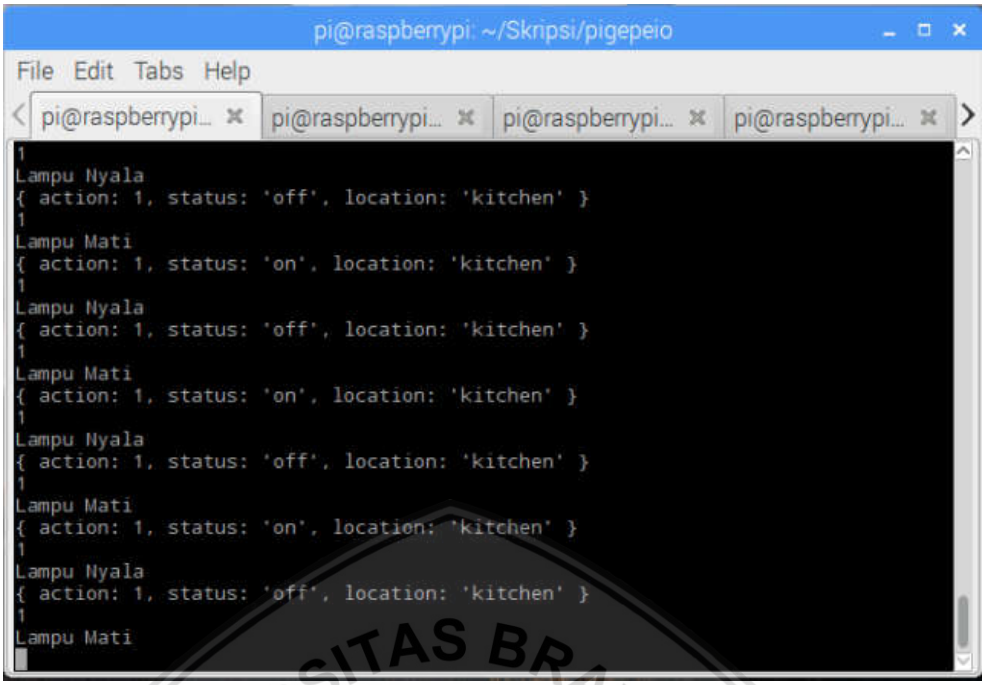
1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk mematikan atau menyalakan lampu di relay. Untuk saat ini lokasi lampu yang terhubung ke relay adalah kitchen.

Perintah suara yang dapat dikenali oleh sistem untuk mematikan atau menyalakan lampu pada relay adalah sebagai berikut:

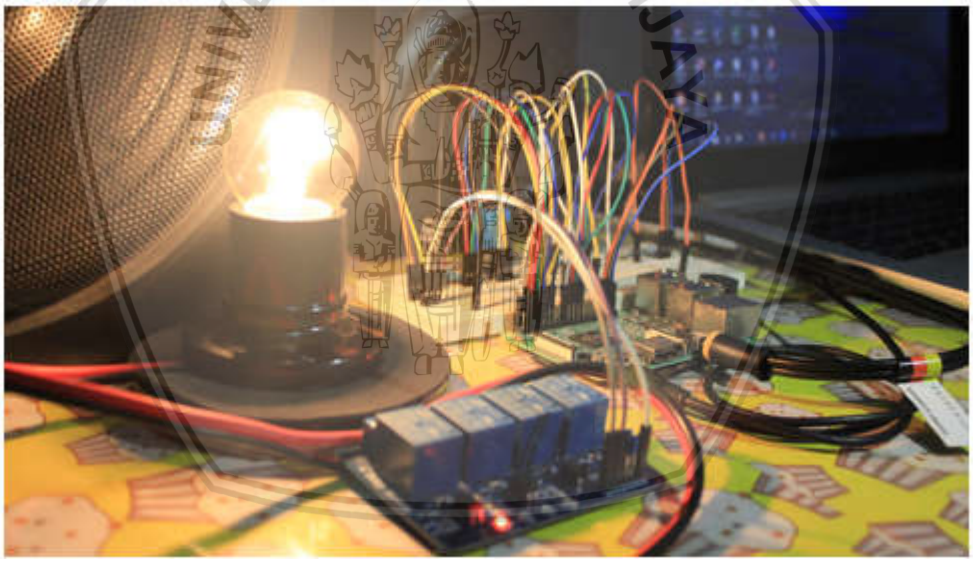
1. Turn on kitchen lamp.
2. Turn off kitchen lamp.

6.3.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem relay. Gambar 6.5 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.6 menunjukan relay pada saat menyala.



Gambar 6.5 Hasil Subscribe Mematikan dan Menyalakan Lampu di Relay



Gambar 6.6 Hasil Ketika Lampu Relay Kondisi On

Pada tahap pengujian ini dilakukan percobaan untuk menyalakan dan mematikan lampu pada relay masing masing sebanyak 5 kali, maka akan ada total 10 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.3

Tabel 6.3 Hasil Percobaan Menyalakan dan Mematikan Relay

Percobaan Ke-	Action	Location	Lampu On/Off
1	On	Kitchen	On
2	Off	Kitchen	Off

3	On	Kitchen	On
4	Off	Kitchen	Off
5	On	Kitchen	On
6	Off	Kitchen	Off
7	On	Kitchen	On
8	Off	Kitchen	Off
9	On	Kitchen	On
10	Off	Kitchen	Off

6.3.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem relay dapat berfungsi mematikan dan menyalakan lampi di relay dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.3 yatu kesesuaian antara action yang diucapkan dan keadaan lampi di relay setelahnya.

6.4 Pengujian Mengubah Warna LED

6.4.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem LED dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk mengubah warna LED berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai acuan untuk mengubah warna LED.

6.4.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk mengubah warna LED pada lokasi tertentu.

Beberapa lokasi yang dapat diubah warna LEDnya adalah sebagai berikut:

1. Garden
2. Livingroom

3. Bedroom

Sedangkan untuk warna yang dapat ditampilkan oleh sub sistem adalah sebagai berikut:

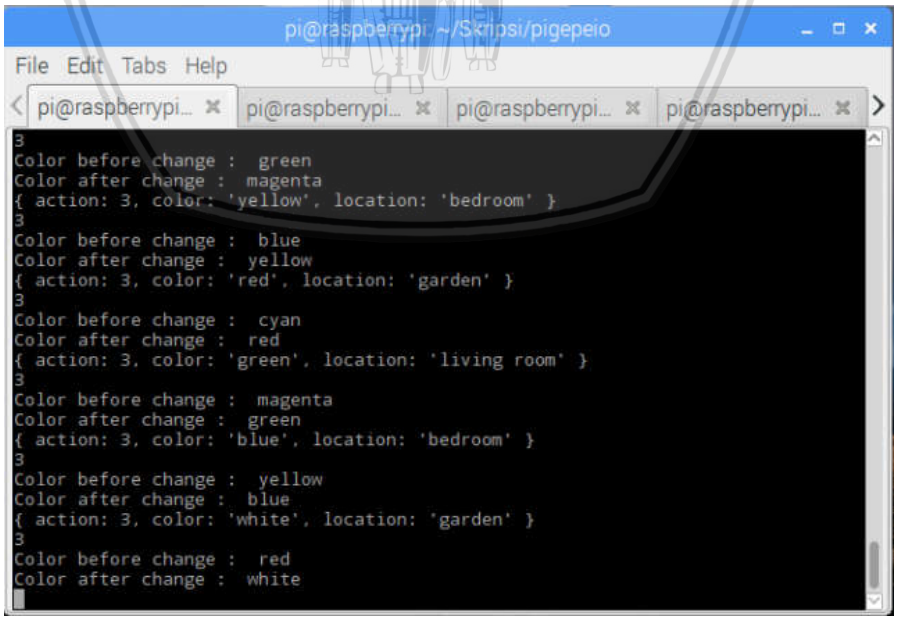
1. Red
2. Green
3. Blue
4. White
5. Cyan
6. Magenta
7. Yellow

Selanjutnya adalah beberpa contoh perintah untuk mengubah warna LED adalah sebagai berikut:

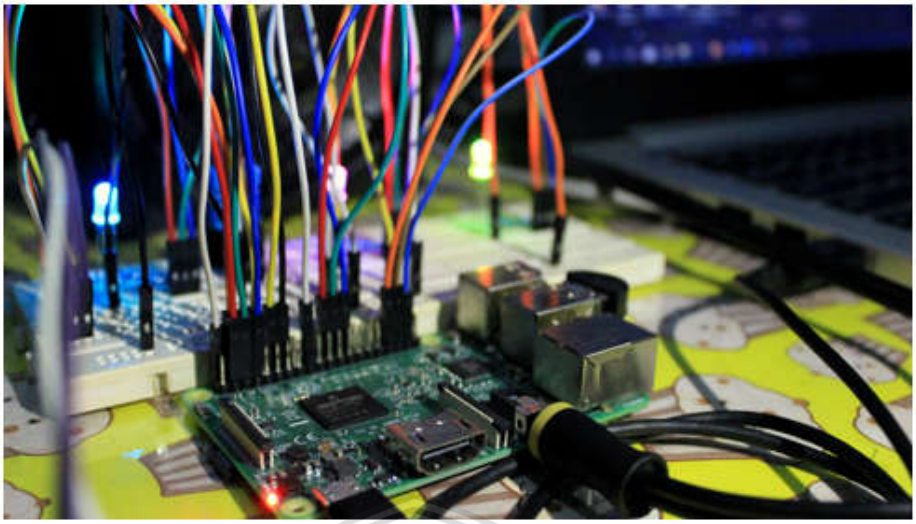
1. Change garden lamp color to blue
2. Change living room lamp color yo yellow
3. Change bedroom lamp color to magenta

6.4.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem LED. Gambar 6.7 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.8 menunjukan LED pada terjadi perubahan warna.



Gambar 6.7 Hasil Subscribe Mengubah Warna LED



Gambar 6.8 Hasil Perubahan Warna Pada LED

Pada tahap pengujian ini dilakukan percobaan untuk mengubah warna LED pada lokasi tertentu sebanyak 10 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.4

Tabel 6.4 Hasil Pengujian Mengubah Warna LED

Percobaan Ke-	Color	Location	Warna LED
1	Red	Garden	Red
2	Green	Living room	Green
3	Blue	Bedroom	Blue
4	Cyan	Garden	Cyan
5	Magenta	Living room	Magenta
6	Yellow	Bedroom	Yellow
7	Red	Garden	Red
8	Green	Living room	Green
9	Blue	Bedroom	Blue
10	White	Garden	White

6.4.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem led dapat berfungsi mengubah warna LED dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.4 yatu kesesuaian antara warna dan lokasi yang diucapkan serta keadaan LED setelahnya.

6.5 Pengujian Mengubah Intensitas Cahaya LED

6.5.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem LED dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk mengubah intensitas LED berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai acuan untuk mengubah intensitas LED.

6.5.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk mengubah intensitas LED pada lokasi tertentu.

Beberapa lokasi yang dapat diatur intensitas LEDnya adalah sebagai berikut:

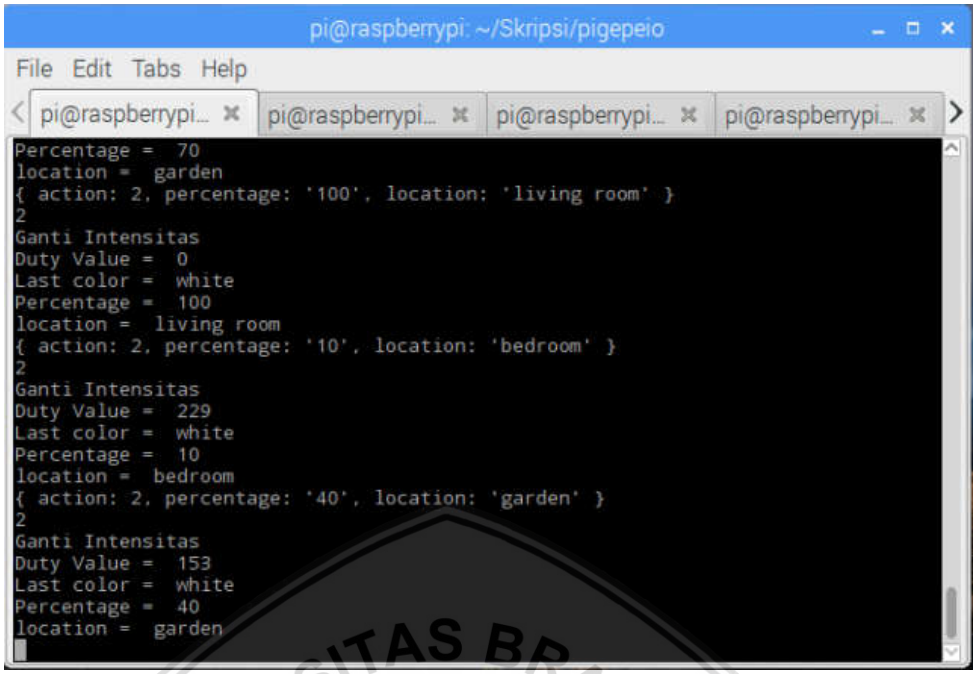
1. Garden
2. Livingroom
3. Bedroom

Sedangkan untuk rentang intensitas yang dapat di atur oleh LED adalah mulai 0 persen hingga 100 persen dan untuk beberapa contoh perintahnya adalah sebagai berikut:

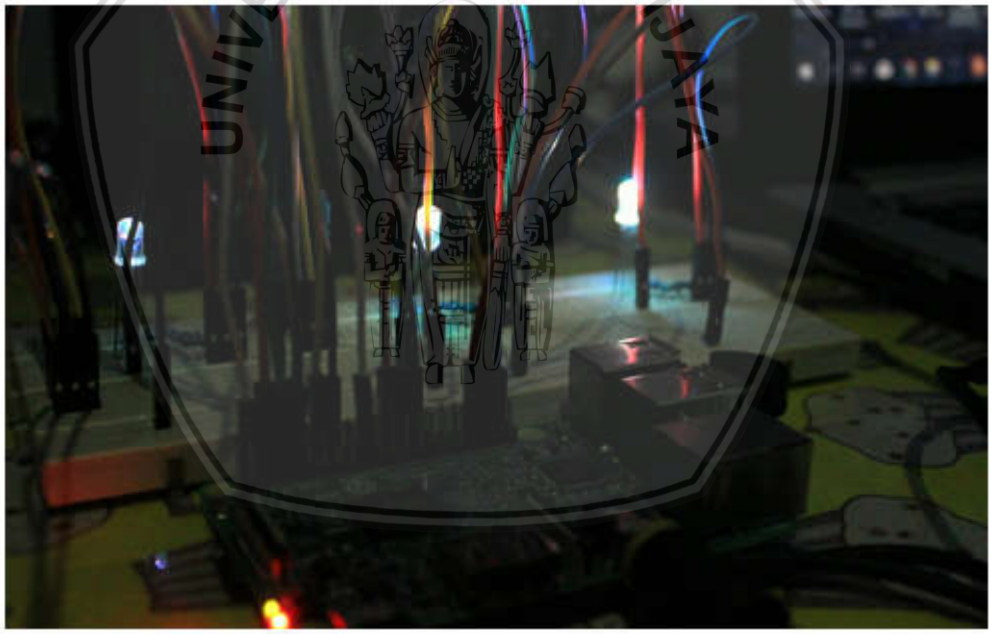
1. Dim garden lamp to 10 percent
2. Dim living room lamp to 40 percent
3. Dim bedroom lamp to 70 percent

6.5.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem LED. Gambar 6.9 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.10 menunjukan LED pada terjadi perubahan intensitas.



Gambar 6.9 Hasil Subscribe Mengubah Intensitas LED



Gambar 6.10 Hasil Perubahan Intensitas LED

Pada tahap pengujian ini dilakukan percobaan untuk mengubah intensitas LED pada lokasi tertentu sebanyak 10 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.5

Tabel 6.5 Hasil Pengujian Mengatur Intensitas LED

Percobaan Ke-	Persentase Diucapkan	Location	Persentase Hasil Subscribe Pada Terminal
1	10 %	Garden	10 %

2	40 %	Living room	40 %
3	70 %	Bedroom	70 %
4	100 %	Garden	100 %
5	10 %	Living room	10 %
6	40 %	Bedroom	40 %
7	70 %	Garden	70 %
8	100 %	Living room	100 %
9	10 %	Bedroom	10 %
10	40 %	Garden	40 %

6.5.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem led dapat berfungsi mengubah intensitas LED dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.5 yatu kesesuaian antara persentase dan lokasi yang diucapkan serta keadaan LED setelahnya.

6.6 Pengujian Penjadwalan Pada LED

6.6.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem LED dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk melakukan penjadwalan pada LED berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai mematikan atau menyalakan LED pada waktu tertentu.

6.6.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk melakukan penjadwalan untuk mematikan atau menyalakan LED pada lokasi dan waktu tertentu.

Beberapa lokasi LED yang dapat dilakukan penjadwalan adalah sebagai berikut:

1. Garden

2. Livingroom
3. Bedroom

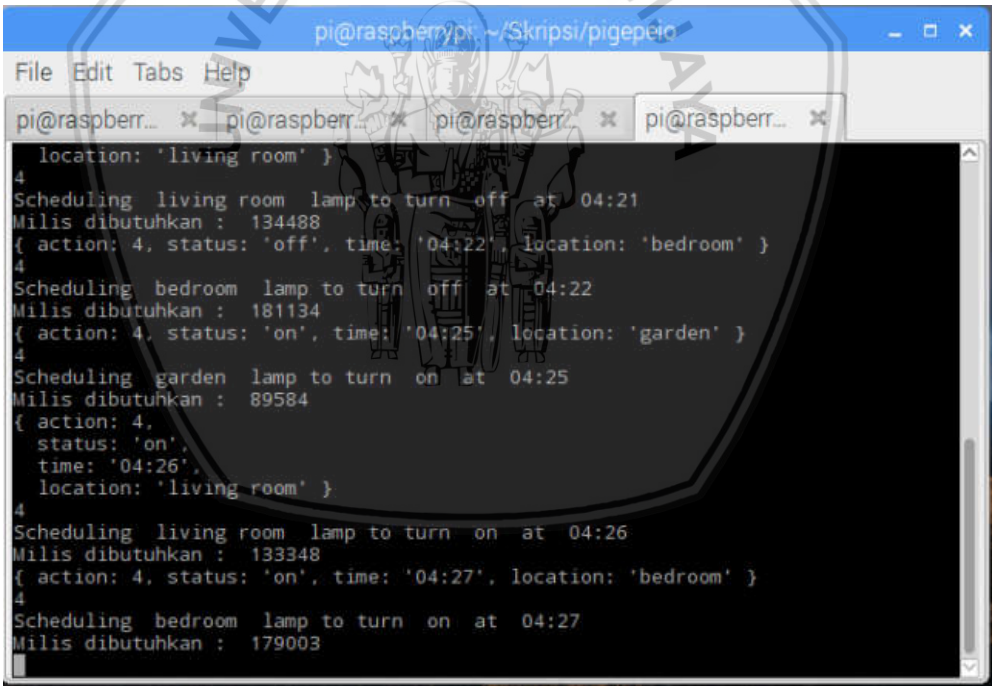
Sedangkan untuk beberapa contoh perintahnya adalah sebagai berikut:

1. Set garden lamp to turn on at 10:10 pm
2. Set living room lamp to turn on at 10:12 pm
3. Set garden lamp to turn off at 10:15 pm

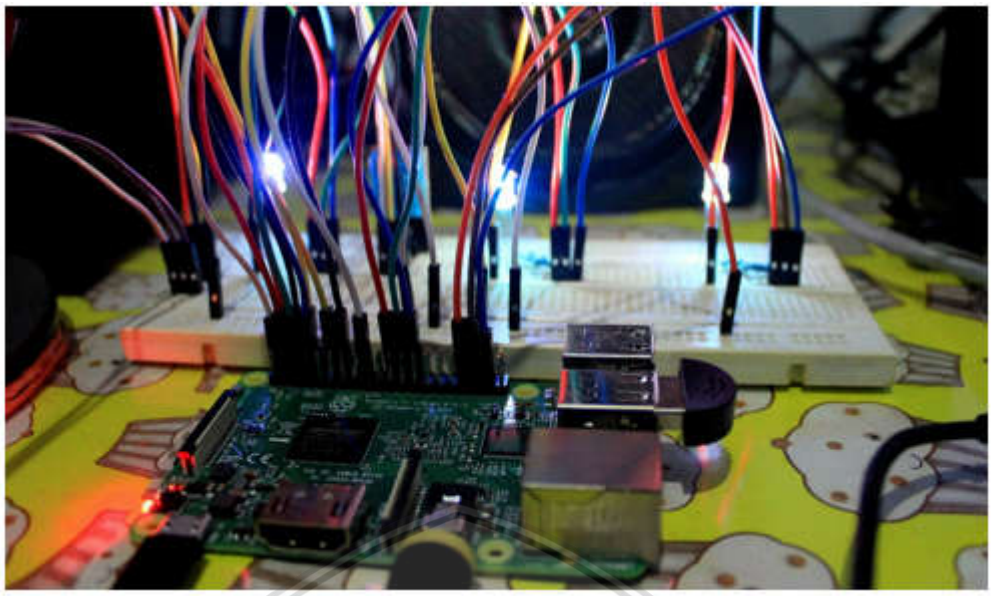
Pada fitur penjadwalan ini untuk setiap lokasi hanya dapat menyimpan satu perintah yaitu untuk mematikan atau menyalakan saja. Perintah terakhir akan menjadi perintah yang akan di eksekusi nantinya.

6.6.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem LED. Gambar 6.11 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.12 menunjukan LED pada saat berhasil dinyalakan dan dimatikan.



Gambar 6.11 Hasil Subscribe Penjadwalan LED



Gambar 6.12 Hasil Penjadwalan Ketika Semua LED Menyala

Pada tahap pengujian ini dilakukan percobaan untuk melakukan penjadwalan LED pada lokasi tertentu sebanyak 10 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.6

Tabel 6.6 Hasil Pengujian Penjadwalan Pada LED

Percobaan Ke-	Action	Time	Location	LED On/Off
1	On	4:16 am	Garden	On
2	On	4:15 am	Living room	On
3	On	4:17 am	Bedroom	On
4	Off	4:20 am	Garden	Off
5	Off	4:21 am	Living room	Off
6	Off	4:22 am	Bedroom	Off
7	On	4:25 am	Garden	On
8	On	4:26 am	Living room	On
9	On	4:27 am	Bedroom	On
10	Off	4:30 am	Garden	Off

6.6.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem led dapat berfungsi melakukan penjadwalan untuk mematikan dan menyalakan LED dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.6 yatu

kesesuaian antara action, waktu dan lokasi yang diucapkan serta keadaan LED setelahnya.

6.7 Pengujian Penjadwalan Lampu Pada Relay

6.7.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan sub sistem relay dalam menjalankan fungsinya. Tingkat keberhasilannya adalah ketika perintah suara yang diucapkan untuk melakukan penjadwalan pada relay berhasil di subscribe oleh sub sistem. Nantinya data hasil subscribe tersebut digunakan sebagai mematikan atau menyalakan lampu di relay pada waktu tertentu.

6.7.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi Sub sistem mikrofon telah dilakukan yaitu mengaktifkan web service, sample app dan wake word engine.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk melakukan penjadwalan untuk mematikan atau menyalakan lampu pada relay. Untuk saat ini lokasi lampu yang terhubung ke relay adalah kitchen.

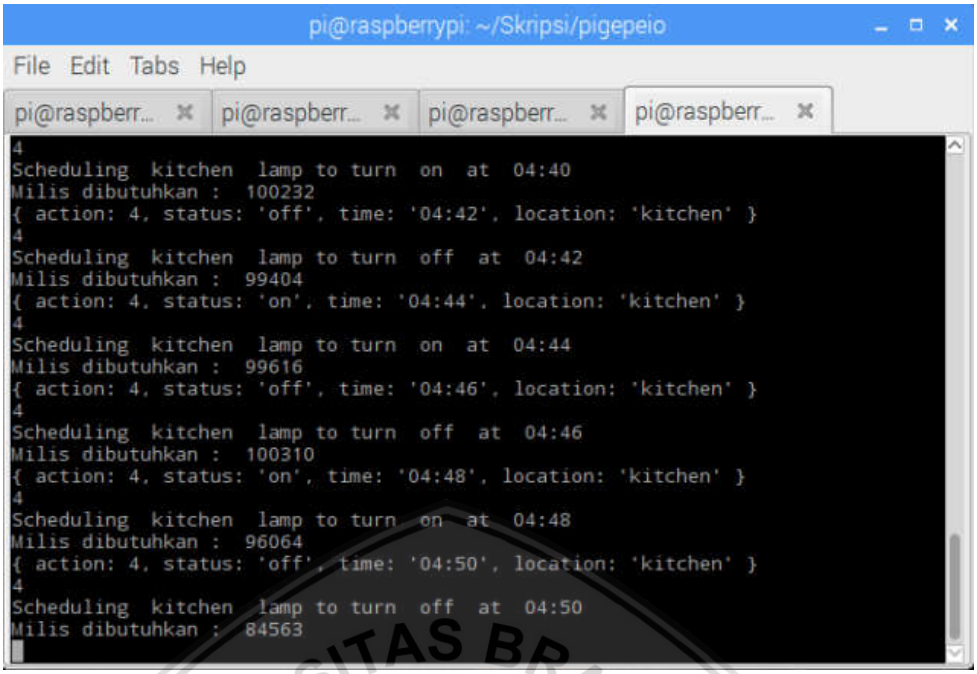
Perintah suara yang dapat dikenali oleh sistem untuk melakukan penjadwalah mematikan atau menyalakan lampu pada relay adalah sebagai berikut:

1. Set kitchen lamp to turn on at 10:10 pm
2. Set kitchen lamp to turn off at 10:15 pm

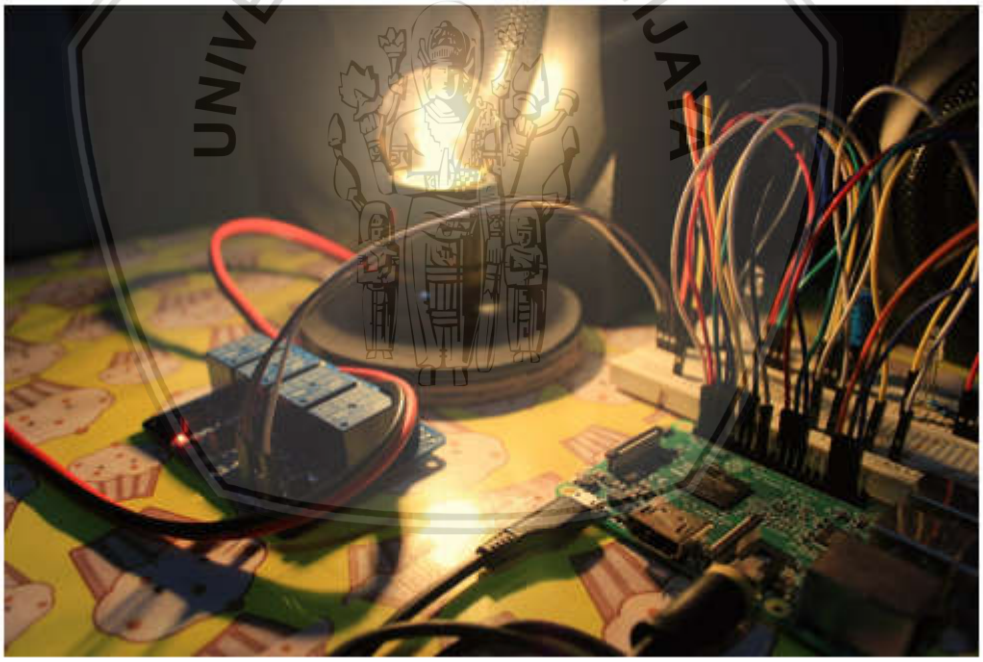
Pada fitur penjadwalan ini untuk setiap lokasi hanya dapat menyimpan satu perintah yaitu untuk mematikan atau menyalakan saja. Perintah terakhir akan menjadi perintah yang akan di eksekusi nantinya.

6.7.3 Hasil Pengujian

Hasil pengujian ini akan ditampilkan pada terminal raspberry pi yang bertindak menjalankan kode program nodejs untuk sub sistem relay. Gambar 6.13 merupakan tampilan terminal raspberry pi ketika menerima data subscribe dari MQTT broker. Sedangkan untuk Gambar 6.14 menunjukan lampu di relay pada saat berhasil dinyalakan dan dimatikan.



Gambar 6.13 Hasil Subscribe Penjadwalan Relay



Gambar 6.14 Hasil Penjadwalan Ketika Lampu di Relay Menyala

Pada tahap pengujian ini dilakukan percobaan untuk melakukan penjadwalan lampu di relay pada lokasi kitchen sebanyak 10 kali pengujian yang dapat dilihat hasilnya pada Tabel 6.7

Tabel 6.7 Hasil Pengujian Penjadwalan Pada Relay

Percobaan Ke-	Action	Time	Location	LED On/Off
1	On	4:32 am	Kitchen	On

2	Off	4:34 am	Kitchen	Off
3	On	4:36 am	Kitchen	On
4	Off	4:38 am	Kitchen	Off
5	On	4:40 am	Kitchen	On
6	Off	4:42 am	Kitchen	Off
7	On	4:44 am	Kitchen	On
8	Off	4:46 am	Kitchen	Off
9	On	4:48 am	Kitchen	On
10	Off	4:50 am	Kitchen	Off

6.7.4 Analisa Pengujian

Data dari hasil percobaan mematikan menunjukan bahwa sub sistem relay dapat berfungsi melakukan penjadwalan untuk mematikan dan menyalakan lampu di relay dengan tingkat keberhasilan 100%. Hal tersebut dapat dilihat pada Tabel 6.7 yatu kesesuaian antara action, waktu dan lokasi yang diucapkan serta keadaan lampu di relay setelahnya.

6.8 Pengujian Tingkat Akurasi

6.8.1 Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui persentase akurasi dari sistem dalam menjalankan fungsinya. Tingkat akurasinya adalah ketika perintah suara yang diucapkan oleh 3 orang yang berbeda dapat dikenali dan dijalankan oleh sistem. Pengujian ini meliputi seluruh pengujian fungsionalitas yang sudah diuji sebelumnya yaitu mematikan, menyalakan, mengubah warna, mengatur intensitas dan melakukan penjadwalan.

6.8.2 Prosedur Pengujian

Prosedur yang dilakukan pada pengujian ini adalah sebagai berikut:

1. Perangkat raspberry telah terhubung ke internet.
2. Tahap-tahap pada implementasi dari semua Sub sistem telah dilakukan.
3. Selanjutnya adalah ucapkan kata **alexa,** tunggu hingga terdengar suara yang menandakan sistem sudah dalam kondisi listen.
4. Lalu ucapkan invocation name yaitu lamp control mode.
5. Selanjutnya ucapkan perintah untuk melakukan pengujian mematikan, menyalakan, mengubah warna, mengatur intensitas dan melakukan penjadwalan seperti pengujan pengujian sebelumnya

6.8.3 Hasil Pengujian

Hasil pengujian tingkat akurasi ini dapat dilihat pada Tabel 6.8 – Tabel 6.11 dimana Tabel 6.8 menunjukan hasil pengujian untuk perintah mematikan dan menyalakan lampu, Tabel 6.9 untuk perintah mengubah warna lampu, Tabel 6.10 untuk perintah mengatur intensitas lampu dan Tabel 6.11 untuk perintah penjadwalan.

Tabel 6.8 Hasil Pengujian Akurasi Perintah Mematikan dan Menyalakan

Percobaan Ke-	Action	Location	Hasil Pada LED On/Off		
			Speaker 1	Speaker 2	Speaker 3
1	On	Garden	On	On	On
2	On	Living room	On	Off	On
3	On	Bedroom	On	On	On
4	On	Kitchen	On	On	On
5	Off	Garden	Off	Off	Off
6	Off	Living room	Off	Off	Off
7	Off	Bedroom	Off	Off	Off
8	Off	Kitchen	Off	Off	Off

Tabel 6.9 Hasil Pengujian Akurasi Perintah Mengubah Warna

Percobaan Ke-	Color	Location	Hasil Warna LED		
			Speaker 1	Speaker 2	Speaker 3
1	Red	Garden	Red	Red	Red
2	Green	Living room	Green	Green	Green
3	Blue	Bedroom	Blue	Blue	Blue
4	Cyan	Garden	Yellow	Yellow	Yellow
5	Magenta	Living room	Magenta	Magenta	Magenta

Tabel 6.10 Hasil Pengujian Akurasi Perintah Mengatur Intensitas

Percobaan Ke-	Persentase	Location	Persentase Hasil Subscribe Pada Terminal		
			Speaker 1	Speaker 2	Speaker 3
1	10 %	Garden	10 %	1010 %	10 %
2	40 %	Living room	40 %	40 %	40 %

3	70 %	Bedroom	70 %	70 %	70 %
4	100 %	Garden	100 %	100 %	100 %
5	10 %	Living room	10 %	10 %	10 %

Tabel 6.11 Hasil Pengujian Akurasi Perintah Penjadwalan

Percobaan Ke-	Action	Time	Location	LED On/Off		
				Speaker 1	Speaker 2	Speaker 3
1	On	1:02 pm	Garden	On	On	On
2	On	1:04 pm	Living room	On	On	On
3	On	1:06 pm	Bedroom	On	On	On
4	On	1:08 pm	Kitchen	On	On	On
5	Off	1:10 pm	Garden	Off	Off	Off
6	Off	1:12 pm	Living room	Off	Off	Off
7	Off	1:14 pm	Bedroom	Off	Off	Off
8	Off	1:18 pm	Kitchen	Off	Off	Off

6.8.4 Analisa Pengujian

Data dari hasil percobaan uji tingkat akurasi menunjukan bahwa untuk perintah mematikan dan menyalakan lampu dapat bekerja dengan tingkat keberhasilan 95,8%, untuk perintah mengubah warna led tingkat keberhasilannya 100%, untuk perintah mengatur intensitas tingkat keberhasilannya 93,3% dan untuk perintah penjadwalan tingkat keberhasilannya 100% . Hal tersebut dapat dilihat pada Tabel 6.8 sampai Tabel 6.11 yatu kesesuaian antara action hasil yang terjadi pada tabel bagian speaker 1 hingga 2.

BAB 7 PENUTUP

Pada bab ini penulis memberikan kesimpulan berdasarkan perancangan, implementasi, pengujian, dan analisis terhadap sistem yang telah dilakukan. Selain itu penulis juga memberikan saran agar penelitian ini dapat dilanjutkan agar penelitian ini menjadi lebih baik dan berguna bagi pembaca. Berikut ini merupakan kesimpulan dan saran yang akan diberikan:

7.1 Kesimpulan

Berdasarkan perancangan, implementasi, pengujian, dan analisis yang telah dilakukan, maka penulis menyimpulkan:

1. Alexa Voice Service diimplementasikan pada perangkat Raspberry Pi dengan cara melakukan cloning file ke github alexa voice service lalu dilakukan instalasi.
2. 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.
3. 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.
4. 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.
5. 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%

7.2 Saran

Berikut beberapa saran yang penulis harapkan dalam pengembangan sistem pada penelitian ini:

1. Menggunakan *voice service* yang berbeda seperti *google assistant*, Mycroft dari Microsoft atau Siri dari Apple agar dapat dibandingkan dan ditemukan yang lebih baik.

2. Membuat sistem menjadi modular dimana ketika terdapat penambahan lampu tidak perlu dilakukan penambahan kode program lagi.
3. Melakukan penelitian lebih lanjut tentang analisis performansi pada kondisi lainnya seperti ketika kondisi koneksi internet yang terbatas atau pada bagaimana respon sistem ketika terdapat noise suara dilingkungan sekitarnya.

DAFTAR PUSTAKA

Ana, M., Ian. B., Alec, Z., Adrian, N., Reggie, C., 2015. *Home Automation Using Raspberry Pi through Siri Enabled Mobile Devices*. *8th IEEE International Conference Humanoid, Nanotechnology, Information Technology Communication and Control, Environment and Management (HNICEM)*.

Anusuya, M.A., Katti, S.K., 2009. *Speech Recognition by Machine: A Review*. *International Journal of Computer Science and Information Security,Maret*.

Grant, C., 2017. *What Is Alexa? What Is the Amazon Echo, and Should You Get One?*. [online]. Tersedia di: < https://thewirecutter.com/reviews/what-is-alexa-what-is-the-amazon-echo-and-should-you-get-one/> [Diakses 7 Desember 2017]

Hidayat, S. dan Farid, S., 2015. *Scheduler and Voice Recognition on Home Automation Control System*. *3rd International Conference on Information and Communication Technology (ICoICT)*.

Kjetil, H., 2017. *Corporate social responsibility*. [online]. Tersedia di: < https://blog.nordicsemi.com/getconnected/voice-control-in-the-smart-home> [Diakses 7 September 2017]

Piyare, R. dan Tazil, M., 2011. *Bluetooth based Home Automation System using Cell Phone*. *IEEE 15th International Symposium on Consumer Electronics*.

Prapto, P., 2015. *Speech Recognition*. [online]. Tersedia di: < https://praptoprasojo.wordpress.com/2015/11/13/speech-recognition/> [Diakses 7 September 2017]

Raspberry Pi Org, 2017. *GPIO Raspberry Pi Models A and B*. [online]. Tersedia di: < https://www.raspberrypi.org/documentation/usage/gpio/> [Diakses 7 September 2017]

Tri, F,Y.S. dan Novi, A., 2009. PERANCANGAN *SOFTWARE APLIKASI PERVASIVE SMART HOME*. *Seminar Nasional Aplikasi Teknologi Informasi*.

Vincent, R., David, M., David, D., Bruno, M., Laurent, D., Christopher, L., 2016. *The Smart Home Concept : our immediate future*. *IEEE E-Learning in Industrial Electronics*.

Wang, Y. dan Dong, P., 2016. *The design and implementation of the voice control system of smart home based on iOS*. *Proceedings of 2016 IEEE International Conference on Mechatronics and Automation*.

Zhu, J., Gao, X., Yang, Y., Li, H., Ai, Z., Cui, X., 2010. *Developing A Voice Control System for ZigBee-based Home Automation*. *2nd IEEE International Conference onNetwork Infrastructure and Digital Content*.