KENDALI LAMPU MENGGUNAKAN PERINTAH SUARA BERBASIS NODE MCU

Asep Nurhuda[1)], Bartolomius Harpad[2)], dan **Muhammad Sirajul Amin Mubarak[3)]**

[1,3]Teknik Informatika, STMIK Widya Cipta Dharma
[2]Program Studi Sistem Informasi, STMIK Widya Cipta Dharma
[1,2,3]Jl. M. Yamin No.25, Samarinda, 75123
E-mail : acep.noor@gmail.com[1)], arvenusharpad@gmail.com[2)], sirajulaminmubarak@gmail.com[3)]

ABSTRAK

Kendali lampu biasanya dilakukan dengan menekan sakelar untuk menyalakan dan mematikannya. pada penelitian ini, bagaimana membuat kontrol lampu menggunakan perintah suara memanfaatkan fitur speech recognition pada smarphone dari jarak jauh (internet). Prinsip kerjanya yaitu perintah suara kendali lampu dikirim dari fitur speech recognition yang ada di smartphone kemudian diterima oleh server dan NodeMCU menjalankan perintah menyalakan dan mematikan lampu. Metode pengembangan sistem yang digunakan dalam pembuatan alat kendali lampu ini adalah metode prototipe dengan tahapan mendengarkan pelanggan, merancang dan membangun prototipe, mengevaluasi prototipe apakah dapat diterima dan uji coba prototipe.Hasil akhir dari penelitian ini yaitu sebuah prototype kendali lampu dari jarak jauh melalui koneksi internet menggunakan perintah suara.

Kata kunci: Kendali Lampu, Perintah Suara, Berbasis NodeMCU.

1. PENDAHULUAN

Salah satu fitur *smartphone* android yang menarik adalah perintah suara (*Speech recognition system)* yaitu sistem yang berfungsi untuk mengubah bahasa lisan menjadi bahasa tulisan. Oleh sistem Android, ucapan manusia akan diidentifikasi menjadi kata atau kalimat berupa teks yang sesuai dengan yang diucapkan. Kata atau kaliamat tersebut dapat digunakan untuk melakukan perintah panggilan dan mengaktifkan aplikasi-aplikasi yang ada pada *smartpone*. Kemampuan fitur *speech recognition* ini akan digunakan untuk mengendalikan perangkat diluar *smartphone* dengan perangkat tambahan, seperti pada tulisan ini yaitu untuk menghidupkan dan mematikan lampu.

Lampu biasanya dihidupkan dan dimatikan menggunakan saklar, cara ini akan diubah dengan memanfaatkan fitur *speech recognition* pada *smartphone* Android untuk pengendalian lampu jarak jauh menggunakan internet dengan NodeMCU berbasis *System On Chip* ESP8266 yang merupakan salah satu bentuk penerapan dari *internet of things (IoT).*

Smartphone Android berfungsi sebagai penerima perintah suara dan pengolahan data yang hasilnya dikirim *via internet* ke server kemudian NodeMCU akan membaca masukkan tersebut dan mengendalikan modul relai sebagai saklar elektronik lampu.

2. RUANG LINGKUP

Permasalahan difokuskan pada :

1. Dalam penelitian ini pengendalian dilakukan untuk mengatur *ON* dan *OFF* lampu *AC.*
2. Pengendalian dilakukan melalui aplikasi pada *smartphone* berbasis Android versi 2.3 (Gingerbread) atau lebih tinggi.
3. Pengendalian utama dilakukan dengan perintah suara dan pegendali alternatif dengan tombol sentuh.
4. Diperlukan koneksi internet.
5. Dalam penelitian ini menggunakan software tambahan IoT *server* Thingspeak.

3. BAHAN DAN METODE

Adapun bahan dan metode yang gunakan dalam membangun penelitian ini yaitu:

3.1 Sistem Minimum

Menurut Immersa Lab (2014), Sistem Minimum Mikrokontroler adalah sebuah rangkaian paling sederhana dari sebuah mikrokontroler agar IC mikrokontroler tersebut bisa beroperasi dan diprogram. Dalam aplikasinya sistem minimum sering dihubungkan dengan rangkaian elektronik dan rangkaian lain untuk tujuan tertentu. Ada beberapa komponen yang harus diperhatikan pada sistem minimum mikrokontroler agar kita mengetahui karakteristik dan dapat menggunakan sesuai kebutuhan, antara lain:

1. *Power Supply*, Semua komponen elektronika membutuhkan *power supply* atau sering juga disebut catu daya. Mikrokontroler beroperasi pada tegangan 5 volt. Biasanya pembuatan catu daya mikrokontroler menggunakan IC regulator 7805 agar tegangannya bisa stabil.

2. *Osilator* (Pembangkit Frekuensi),kalau manusia memiliki jantung untuk bisa hidup maka mikrokontroler memiliki osilator untuk bisa beroperasi. Mikrokontroler sendiri sudah memiliki *osilator internal* yaitu sebesar 8Mhz tetapi kadang kala agar kinerja mikronkontroler lebih cepat *osilator internal* tidak bisa menangani kasus tersebut. Oleh karena itu dibutuhkan *osilator eksternal* (kristal) yang nilainya lebih dari 8Mhz. Perlu diperhatikan mikrokontroler hanya bisa beroprasi sampai 16 Mhz. jadi kalau memilih kristal untuk AVR tidak boleh lebih dari 16Mhz.
3. ISP (*In-System Programmable*), Sistem Minimum Mikrokontroler dibuat untuk di program. Prinsipnya mikrokontroler bisa diprogram secara parallel atau secara seri. Pemrograman mikrokontroler secara seri atau lebih dikenal dengan ISP tidak perlu memerlukan banyak jalur data. Tapi ISP memiliki kelemahan, jika salah *setting fuse bit* yang memiliki fungsi vital, semisal pin reset *disable* maka alamat *DEH* sudah tidak bisa digunakan lagi. Untuk mengembalikan settingan *fuse bit* tadi, harus menggunakan pemrograman tipe parallel (*high voltage programming*).
4. Rangkaian *Reset*, sama fungsinya dengan rangkaian reset pada komputer. Fungsi *reset* di mikrokontroler yaitu untuk me-*restart* program, sehingga kembali ke program awal. Penggunaan *reset* pada mikrokontroler tergantung pada pengguna, dalam hal ini opsional.

3.2 *Speech recognition*

Menurut Waldi (2015) *Speech recognition* adalah proses identifikasi suara berdasarkan kata yang diucapkan dengan melakukan konversi sebuah sinyal akustik yang ditangkap oleh audio device (perangkat *input* suara). *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.

Secara umum, *speech recognizer* memproses sinyal suara yang masuk dan menyimpannya dalam bentuk digital. Hasil proses digitalisasi tersebut kemudian dikonversi dalam bentuk spektrum suara yang akan dianalisa dengan membandingkannya dengan template suara pada *database* sistem.

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 & control*, pemasukan 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 *voice recognition*.

Algoritma yang diimplementasikan pada bahasan mengenai proses *speech recognition* ini adalah algoritma FFT *(Fast Fourier Transform*), yaitu algoritma yang cukup efisien dalam pemrosesan sinyal digital (dalam hal ini suara) dalam bentuk diskrit. Algoritma ini mengimplementasikan algoritma *Divide and Conquer* untuk pemrosesannya. Konsep utama algoritma ini adalah mengubah sinyal suara yang berbasis waktu menjadi berbasis frekuensi dengan membagi masalah menjadi beberapa upa masalah yang lebih kecil. Kemudian, setiap upa masalah diselesaikan dengan cara melakukan pencocokan pola digital suara.

Pada penelitian ini hanya menggunakan *speech recognition* karena kompleksitas algoritma yang diimplementasikan lebih sederhana dari pada *voice recognition*.

3.3 Mikrokontroler

Menurut Kadir (2014) *mikrokontroler* adalah suatu system yang mengandung masukan/keluaran, memori dan prosesor, yang digunakan pada produk seperti mesin cuci, pemutar video, mobil dan telepon. Pada prisipnya mikrokontroler adalah hal-hal yang bersifat berulang dan dapat berinteraksi dengan dengan peranti peranti eksternal seperti sensor ultrasonic untuk megukur jarak terhadap suatu objek, penerima GPS untuk menerima data posisi kebumian dari satelit, dan motor untuk mengontrol gerakan pada robot. Sebagai *computer* yang berukuran kecil, mikrokontroler cocok untuk diapliaksikan pada benda benda yang berukuran kecil misalnya sebagai pengendali pada *quadcore* ataupun robot.

3.4 ThingSpeak

Menurut situs resmi thingspeak.com (2017) "*ThingSpeak™ is an IoT analytics platform service that allows you to aggregate, visualize and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak you can perform online analysis and processing of the data as it comes in. ThingSpeak is often used for prototyping and proof of concept IoT systems that require analytics"* (ThingSpeak ™ adalah layanan platform analitik IoT yang memungkinkan Anda untuk mengumpulkan, memvisualisasikan, dan menganalisis aliran data langsung di *cloud*. ThingSpeak menyediakan visualisasi instan dari data yang dikirim oleh perangkat Anda ke ThingSpeak. Dengan kemampuan untuk mengeksekusi kode MATLAB® di ThingSpeak Anda dapat melakukan analisis online dan pemrosesan data saat masuk. ThingSpeak sering digunakan untuk membuat prototipe dan bukti konsep sistem IoT yang memerlukan analitik). Fitur Thingspeak diantaranya mengumpulkan data dalam *private channel*, mendukung Restful dan MQTT API, analisis dan visualisasi

berbasis MATLAB®, mendukung *Alert, event scheduling*, integrasi *App* dan dukungan komunitas global. Thingspeak dapat bekerja pada perangkat Arduino, Particle Photon dan Elektron, WiFi modul ESP8266 dan Raspberry Pi. Thingspeak juga mendukung integrasi pada aplikasi *mobile* dan *web*, Twitter, Twillio dan Matlab.

3.5 Arduino IDE

Menurut Kadir (2014) Arduino IDE adalah *software* yang disediakan disitus Arduino.cc yang ditujukan sebagai perangkat pengembangan *sketch* yang digunakan sebagai program dipapan Arduino. IDE (*Integrated Development Environment*) berarti bentuk alat pengembangan program yang terintegrasi sehingga berbagai keperluan disediakan dan dinyatakan dalam bentuk antar muka berbasis menu dengan menggunakan Arduino IDE, anda bisa menulis *sketch*, memeriksa ada kesalahan atau tidak pada *sketch* dan kemudahan mengunggah *sketch* yang sudah terkompilasi dengan papan Arduino

3.6 NodeMCU

Menurut situs resmi nodemcu.com (2014) *"An open-source firmware and development kit that helps you to prototype your IOT product within a few Lua script lines"* (sebuah *firmware* sumber terbuka dan set pengembangan yang membantu Anda untuk membuat protoipe produk IOT Anda dalam beberapa baris skrip Lua). NodeMCU terdiri dari perangkat keras berupa *System On Chip* ESP8266 dari ESP8266 buatan Espressif System, juga *firmware* yang digunakan, yang menggunakan bahasa pemrograman *scripting* Lua. Istilah NodeMCU secara *default* sebenarnya mengacu pada firmware yang digunakan daripada perangkat keras *development kit*. NodeMCU bisa dianalogikan sebagai board arduino-nya ESP8266. NodeMCU telah me-*package* ESP8266 ke dalam sebuah *board* yang kompak dengan berbagai fitur layaknya mikrokontroler ditambah kapabilitas akses terhadap *Wifi* juga chip komunikasi *USB to serial*. Sehingga untuk memprogramnya hanya diperlukan ekstensi kabel data *USB* persis yang digunakan sebagai kabel data dan kabel *charging* smartphone Android.

3.7 ESP8266

Menurut Kadir (2011) ESP8266 adalah keluarga modul yang berfungsi sebagai peranti *Wi-Fi*, yang dibuat oleh perusahaan China yang bernama Espressif. Salah satu jenisnya adalah ESP-01. Dengan menggunakan peranti ini, NodeMCU dapat berhubungan dengan Internet sehingga NodeMCU dapat digunakan untuk mengirim data yang berasal dari sensor ke *database server*. Kelebihan modul ini dibandingkan dengan *Wifi shield* seperti Arduino *Wifi Shield* terletak pada ukurannya yang jauh lebih kecil dan harga yang jauh lebih murah.

3.8 Modul Relai

Menurut Kadir (2011) relai mempunyai lilitan. Lilitan ini yang membuat saklar direlai dapat menutup dan membuka, modul yang mengandung relai telah dilengkapi dengan saklar elektronik berupa transistor, modul seperti itu akan memudahkan anda dalam menggunakan relai karena telah dilengkapi dengan piranti untuk mencolokan kabel masukan (untuk mengendalikan lilitan).

3.9 Wiring Diagram

Menurut Pahlevi (2011) *Wiring diagram* menggambarkan hubungan rangkaian secara detail, dari mulai simbol rangkaian sampai dengan koneksi rangkaian tersebut dengan komponen lain, sehingga akan mudah bagi kita untuk mengikuti alur sebenarnya dari sebuah rangkaian, karena digambarkan secara rinci dan lengkap.

3.10 Sistem Operasi Android

Menurut Intania dkk (2012) Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti telepon pintar dan komputer tablet. Android, Inc. didirikan di Palo Alto, California, pada bulan Oktober 2003 oleh Andy Rubin (pendiri Danger), Rich Miner (pendiri Wildfire Communications, Inc.), Nick Sears (mantan VP T-Mobile), dan Chris White (kepala desain dan pengembangan antarmuka WebTV) untuk mengembangkan "perangkat seluler pintar yang lebih sadar akan lokasi dan preferensi penggunanya". Tujuan awal pengembangan Android adalah untuk mengembangkan sebuah sistem operasi canggih yang diperuntukkan bagi kamera digital, namun kemudian disadari bahwa pasar untuk perangkat tersebut tidak cukup besar, dan pengembangan Android lalu dialihkan bagi pasar telepon pintar untuk menyaingi Symbian dan Windows Mobile (iPhone Apple belum dirilis pada saat itu). Meskipun para pengembang Android adalah pakar-pakar teknologi yang berpengalaman, Android Inc. dioperasikan secara diam-diam, hanya diungkapkan bahwa para pengembang sedang menciptakan sebuah perangkat lunak yang diperuntukkan bagi telepon seluler. Masih pada tahun yang sama, Rubin kehabisan uang.Steve Perlman, seorang teman dekat Rubin, meminjaminya $10.000 tunai dan menolak tawaran saham di perusahaan.

Google mengakuisisi Android Inc. pada tanggal 17 Agustus 2005, menjadikannya sebagai anak perusahaan yang sepenuhnya dimiliki oleh Google. Pendiri Android Inc. seperti Rubin, Miner dan White tetap bekerja di perusahaan setelah diakuisisi oleh Google. Setelah itu, tidak banyak yang diketahui tentang perkembangan Android Inc, namun banyak anggapan yang menyatakan bahwa Google berencana untuk memasuki pasar telepon seluler dengan tindakannya ini. Di Google, tim yang dipimpin oleh Rubin mulai mengembangkan *platform* perangkat seluler dengan menggunakan kernel Linux. Google memasarkan *platform* tersebut kepada produsen

perangkat seluler dan operator nirkabel, dengan janji bahwa mereka menyediakan sistem yang fleksibel dan bisa diperbarui. Google telah memilih beberapa mitra perusahaan perangkat lunak dan perangkat keras, serta mengisyaratkan kepada operator seluler bahwa kerjasama ini terbuka bagi siapapun yang ingin berpartisipasi.

3.11App Inventor

App Inventor adalah aplikasi web sumber terbuka yang awalnya dikembangkan oleh Google, dan saat ini dikelola oleh *Massachusetts Institute of Technology* (MIT). App Inventor memungkinkan pengguna baru untuk memprogram komputer untuk menciptakan aplikasi perangkat lunak bagi sistem operasi Android. App Inventor menggunakan antarmuka grafis, serupa dengan antarmuka pengguna pada *Scratch* dan *Star Logo* TNG, yang memungkinkan pengguna untuk men-*drag-and-drop obyek* visual untuk menciptakan aplikasi yang bisa dijalankan pada perangkat Android. Dalam menciptakan App Inventor, Google telah melakukan riset yang berhubungan dengan komputasi edukasional dan menyelesaikan lingkungan pengembangan *online* Google. App Inventor ini sedikit berbeda dengan app *builder* lain. Dengan App Inventor pengguna tidak pernah menemui kasus para *developer* aplikasi yang dibuat tidak jalan, dan ternyata itu hanya karena kesalahan sintak kurang tanda *semicolon* (;). App Inventor ini menggunakan teknik visual programming, berbentuk seperti susunan *puzzle-puzzle* yang memiliki logika tertentu. Pada lingkungan kerja App Inventor ini terdapat beberapa komponen yang terdiri dari :

1. Komponen Desainer

Komponen desainer berjalan pada *browser* yang digunakan untuk memilih komponen yang dibutuhkan dan mengatur *property*-nya. Pada komponen desainer sendiri terdapat 5 bagian, yaitu *palette, viewer, component, media* dan *properties*, seperti terlihat pada gambar di atas.

1) *Palette*: *list* komponen yang bisa digunakan.
2) *Viewer*: untuk menempatkan komponen dan mengaturnya sesuai tampilan yang diinginkan
3) *Component*: tempat *list* komponen yang dipakai pada *project* kita
4) *Media*: mengambil media audi atau gambar untuk *project* kita
5) *Properties*: mengatur properties komponen yang digunakan, seperti *width, height, name*, dll.

2. *Block Editor*

Block Editor berjalan di luar browser dan digunakan untuk membuat dan mengatur behaviour dari komponen-komponen yang kita pilih dari komponen desainer. Nah, berhubung *block editor* ini basisnya java, jadi lepi kalian sebelumnya harus uda ada jdk sama jre nya ya. 3. Emulator, Emulator digunakan untuk menjalankan dan mengetest project yang telah kita buat. Jadi yang blom punya android pun tetep bisa belajar karna app inventor menyediakan emulatornya juga.

3.12*Flowchart*

Menurut Barakbah, dkk (2013) dari bukunya yang berjudul Logika Dan Algoritma *Flowchart* adalah cara penulisan algoritma dengan menggunakan notasi grafis. *Flowchart* merupakan gambar atau bagan yang memperlihatkan urutan atau langkah-langkah dari suatu program dan hubungan antar proses beserta pernyataannya. Gambaran ini dinyatakan dengan simbol. Dengan demikian setiap simbol menggambarkan proses tertentu. Sedangkan antara proses digambarkan dengan garis penghubung. Dengan menggunakan *Flowchart* akan memudahkan kita untuk melakukanpengecekan bagian-bagian yang terlupakan dalam analisis masalah. Disamping itu *Flowchart* juga berguna sebagai fasilitas untuk berkomunikasi antara pemrogram yang bekerja dalam tim suatu proyek.

Flowchart menolong analis dan programmer untuk memecahkan masalah kedalam segmen segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Pada dasarnya terdapat berbagai macam *Flowchart*, diantaranya yaitu *Flowchart* Sistem (*System Flowchart*), *Flowchart Paperwork* / *Flowchart* Dokumen (*Document Flowchart*), *Flowchart* Skematik (Schematic *Flowchart*), *Flowchart* Program (*Program Flowchart*), *Flowchart* Proses (*Process Flowchart*).

Untuk keperluan pembuatan program maka digunakan *Flowchart* Program. *Flowchart* program menggambarkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program. Programmer menggunakan *Flowchart* program untuk menggambarkan urutan instruksi dari program komputer. Analis Sistem menggunakan *Flowchart* program untuk menggambarkan urutan tugas-tugas pekerjaan dalam suatu prosedur atau operasi.

Dalam pembuatan *Flowchart* program tidak ada rumus atau patokan yang bersifat mutlak. Karena *Flowchart* merupakan gambaran hasil pemikirandalam menganalisis suatu masalah yang nantinya akan diubah menjadi program komputer. Sehingga *Flowchart* yang dihasilkan dapat bervariasi antara satu pemrogram dengan yang lainnya. Namum demikian terdapat beberapa anjuran yang harus diperhatikan, yaitu ;

1. *Flowchart* digambarkan di suatu halaman dimulai dari sisi atas ke bawah dan dari sisi kiri ke kanan.
2. Aktivitas yang digambarkan harus didefinisikan dengan menggunakan bahasa dan simbol yangtepat dan definisi ini harus dapat dimengerti oleh pembacanya.
3. Kapan aktivitas dimulai dan berakhir harus ditentukan secara jelas. Hanya terdapat satu titik awal dan satu titik akhir.
4. Setiap langkah dari aktivitas harus diuraikan dengan menggunakan deskripsi kata kerja, misalkan menghitung nilai rata rata

5. Setiap langkah dari aktivitas harus berada pada urutan yang benar.
6. Lingkup dan *range* dari aktifitas yang sedang digambarkan harus ditelusuri dengan hati- hati. Percabangan-percabangan yang memotong aktivitas yang sedang digambarkan tidak perlu digambarkan pada *Flowchart* yang sama. Simbol konektor harus digunakan dan percabangannya diletakan pada halaman yang terpisah atau hilangkan seluruhnya bila percabangannya tidak berkaitan dengan sistem.
7. Gunakan simbol-simbol *Flowchart* yang standar.

3.13 *Prototype*

Menurut Darmawan (2013), "*Prototype* adalah satu versi dari sebuah sistem potensial yang memberikan ide bagi para pengembang dan calon pengguna, bagaimana sistem akan berfungsi dalam bentuk yang telah selesai."

Pendekatan *prototyping* metode digunakan jika pemakai hanya mendefinisikan objektif umum dari perangkat lunak tanpa melihat kebutuhan *input*, pemeroses dan *output*nya.

Tahap-tahap pengembangan *prototype* model menurut Pressman (2012) adalah:

1. Mendengarkan pelanggan
   Pada tahap ini dilakukan pengumpulan kebutuhan dari sistem dengan cara mendengar keluhan dari pelanggan. Untuk membuat suatu sistem yang sesuai kebutuhan, maka harus diketahui terlebih dahulu bagaimana sistem yang sedang berjalan untuk kemudian mengetahui masalah yang terjadi.
2. Merancang dan membuat *prototype*
   Pada tahap ini, dilakukan perancangan dan pembuatan *prototype* sistem. *Prototype* yang dibuat disesuaikan dengan kebutuhan sistem yang telah didefinisikan sebelumnya dari keluhan pelanggan dan pengguna.
3. Uji coba
   Pada tahap ini, *prototype* dari sistem diuji coba oleh pelanggan atau pengguna. Kemudian dilakukan evaluasi kekurangan-kekurangan dari kebutuhan pelanggan. Pengembang kemudian kembali mendengarkan keluhan dari pelanggan untuk memperbaiki *prototype* yang ada.

Kelebihan metode *prototype:*

1. Adanya Komunikasi yang baik antara pengembang dan pelanggan.
2. pengembang dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan.
3. Lebih menghemat waktu dalam pengembangan sistem.

Kekurangan metode *prototype:*

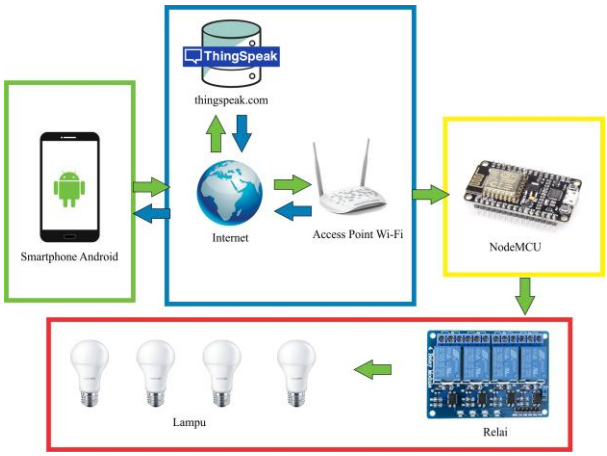
1. Resiko tinggi yaitu untuk masalah-masalah yang tidak terstruktur dengan baik, ada perubahan besar dari waktu ke waktu, dan adanya persyaratan data yang tidak menentu.
2. Interaksi pemakai penting. Sistem harus menyediakan dialog *on-line* antara pelanggan dan komputer.

Hubungan pelanggan dengan komputer yang disediakan mungkin tidak mencerminkan teknik perancangan yang baik.

4. RANCANGAN ALAT DAN SISTEM

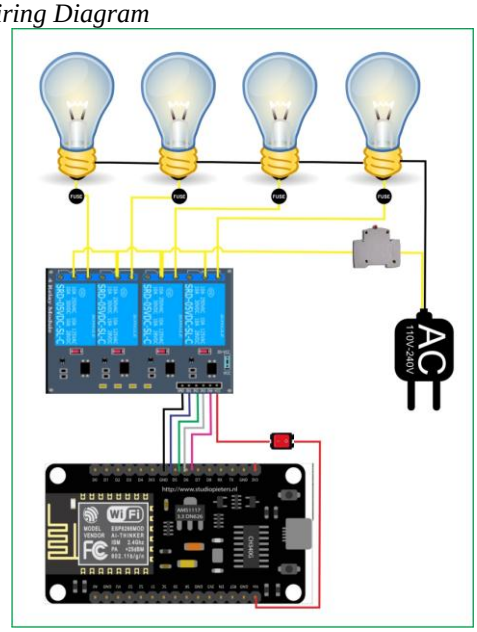
Perancangan kendali lampu menggunakan perintah suara berbasis NodeMCU ini menggunakan blok diagram, *Wiring Diagram* dan *Flowchart.*

1. Blok Diagram
   1) Blok Berwarna hijau merupakan inputan alat, yaitu s*martphone* Android untuk menginputkan perintah suara.
   2) Blok berwarna biru berfungsi menghubungkan *smartphnoe* Android ke NodeMCU melalui Internet. Thingspeak.com berfungsi menyimpan perintah yang dikirim oleh Smartphone.
   3) Blok berwarna kuning yaitu NodeMCU, NodeMCU akan membaca data yang tersimpan di server thingspeak.com
   4) Blok berwarna merah merupakan output alat, yaitu relai dan lampu. Jika NodeMCU membaca perintah untuk mematikan atau menghidupkan lampu, maka relai akan diperintahkan untuk memutuskan atau menyambungkan aliran listrik, sehingga lampu akan mati atau hidup.



Gambar 1 Desain Blok Diagram

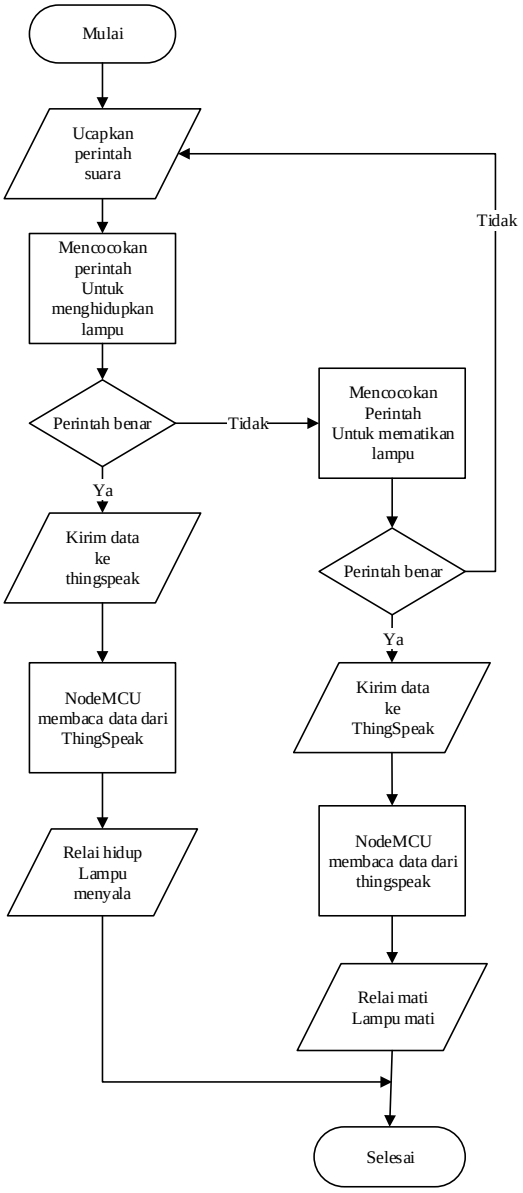
2. *Wiring Diagram*



Gambar 2 *Wiring Diagram* Alat Kendali Lampu

Pada gambar diatas, relai dihubungkan ke NodeMCU, dimana pin IN1 relai ke pin D5 NodeMCU, pin IN2 relai ke pin D6 NodeMCU, pin IN3 relai ke pin D7 NodeMCU, pin IN4 relai ke pin D8 NodeMCU, pin GND relai ke pin GND NodeMCU, pin VCC relai ke pin Vin NodeMCU dan terhubung menggunakan saklar on/off. Kemudian 1 kabel (warna kuning) masing-masing lampu dihubungkan ke konektor NO relai, konektor COM relai dipararel sampai ke MCB dan steker. Dan 1 kabel (warna hitam) lagi dipararel ke semua lampu sampai ke steker.

3. *Flowchart* sistem



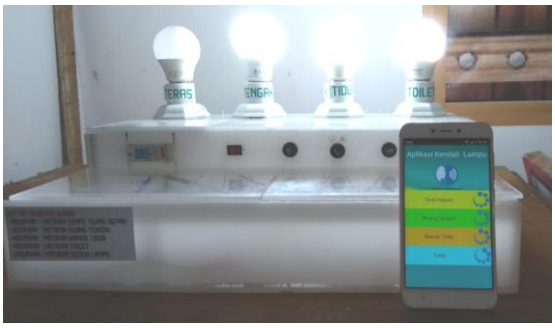
Gambar 3 *Flowchart* Sistem

Keterangaan Gambar 3

1) Mulai
2) Mengucapkan perintah suara ke aplikasi yang ada di *Smartphone.*
3) Mencocokan perintah untuk menghidupkan lampu.
4) Jika perintah benar maka akan mengirim data atau perintah ke server thingspeak
5) Jika perintah salah maka sistem akan mencocokan dengan perintah mematikakan lampu.
6) kemudian NodeMCU membaca data atau perintah terakhir yang disimpan Thingspeak untuk mengendalikan relai yaitu menghidupkan atau matikan lampu.
7) Selesai.

5. IMPLEMENTASI

Konfigurasi pin-pin pada setiap perangkat akan dihubungkan dengan pin atau port yang terdapat pada NodeMCU yang sudah terisi program. Dalam hal ini pada pin 14, 12, 13, 15 akan mengirimkan sinyal ke *relay* dan akan menggerakan saklar yang terdapat pada relay sehingga lampu dapat dinyalakan ataupun dimatikan.



Gambar 4 Aplikasi Saat *Loading*

Pada gambar 4 menampilkan kondisi aplikasi Android pada saat melakukan pengendalian, indikator lampu akan berubah menjadi animasi loading hingga berhasil mengirim data ke the Thingspeak.

Gambar 5 Aplikasi Saat Lampu Hidup

Pada gambar 5 menunjukan jika lampu hidup maka indikator lampu akan berwarna hijau dan mengeluarkan suara tentang status lampu.

Gambar 6 Aplikasi Saat Lampu Mati

Pada gambar 6 ketika lampu mati indikator lampu akan berganti warna merah dan mengeluarkan suara tentang status lampu. Sehingga tidak memerlukan banyak tombol karena status lampu dan tombol kendali menjadi satu.

6. KESIMPULAN

Telah berhasil dibuat alat yang dapat mengendalikan lampu menggunakan perintah suara menggunakan NodeMCU dengan memanfaatkan dan memaksimalkan fitur s*martphone* Android yaitu *Speech Recognition.*

Pengendalian lampu menggunakan *smartphone* dapat dilakukan dari jarak jauh melalui koneksi internet sehingga memungkinkan untuk menghidupkan dan mematikan lampu dari manapun selama koneksi internet tersedia.

7. SARAN

Diharapkan alat ini memiliki IoT *server* mandiri sehingga tidak ada ketergantungan pada sebuah vendor tertentu dan dapat mengurangi *delay* pada saat melakukan pengendalian lampu. Belum tersedia fitur penjadwalan untuk mematikan dan menghidupkan lampu. Belum memiliki sensor yang dapat membaca intensitas cahaya untuk dapat digunakan secara otomatis jika cahaya redup atau gelap maka lampu akan menyala.

8. DAFTAR PUSTAKA

Barakbah dkk 2013. *Logika Dan Algoritma*. Program Studi Teknik Informatika. Surabaya: Politeknik Elektronika Negeri Surabaya.

Darmawan, Deni. 2013. *Sistem Informasi Manajemen*. Bandung: PT Remaja Rosdakarya Offset.

Immersa Lab, 2014. *Sistem Minimum Mikrokontroler, (http://www.immersa-lab.com/sistem-minimum-mikrokontroler.htm)*, diakses 5 Oktober 2017.

Intania, dkk. 2012. *Sekali Baca Langsung Inget: Mengupas Lengkap All About Android*. Jakarta: Kuncikom.

Kadir, Abdul. 2014. *Buku Pintar Pemrograman Arduino.* Yogyakarta: Media kom.

Kadir, Abdul 2011. *Panduan Praktis Mempelajari Aplikasi Mikrokontroller Dan Pemrogramannya Menggunakan Android*. D.I Yogyakarta: Andi.

Pahlevi, Riyan Fitrian. 2011. *Menginterpresentasikan Gambar Teknik*. Yogyakarta: Modul TKR.

Pressman, RogerS. 2012. *Rekayasa Perangkat Lunak Buku I (Pendekatan Praktis)* Edisi 7 : Buku 1. Yogyakarta : Andi

ThingSpeak. 2017. *Learn More About ThingSpeak. (https://thingspeak.com/pages/learn_more).* diakses 25 April 2018.

Waldi, Vernando R. 2015. *Kontrol Penerangan Ruangan menggunakan sensor suara (Speech Recognition) Berbasis Arduino.* Skripsi tidak diterbitkan. Jurusan Teknik Elektro, Manado: Politeknik Negeri.