BAB 3
PERANCANGAN SISTEM

3.1 ALAT DAN BAHAN

Dalam pembuatan alat ini dibutuhkan beberapa komponen yang dibutuhkan dalam perancangan sistem, alat dan bahan yang digunakan dapat dilihat pada tabel 3.1 dibawah ini :

Tabel 3.1 Daftar Alat dan Bahan

No	Alat dan Bahan	Jumlah
1	Laptop	1
2	*Smartphone*	1
3	NodeMcu ESP8266	1
4	Motor Servo	1
5	Sensor Api (*IR Flame Sensor*)	1
6	*Buzzer Active*	1
6	LED	3
7	*Power Supply*	1
8	*Software* Arduino IDE	1
9	*Google Firebase*	1
10	*Software* MIT App *Inventor*	1
11	*Software Wireshark*	1

3.1.1 Laptop

Dalam proses pengerjaan dan perancangan tugas akhir ini, laptop merupakan perangkat yang digunakan untuk melakukan riset melalui internet mengenai hal yang berkaitan dengan topik tugas akhir, penyusunan proposal tugas akhir, serta melakukan pemrograman atau *coding* pada sistem tertanam yang dibuat dalam tugas akhir ini. Spesifikasi laptop yang digunakan pada tugas akhir ini yaitu prosessor Intel(R) Celeron(R) CPU N2840, kecepatan *clock* sebesar 2.16 GHz, dan RAM *memory* sebesar 2GB.

3.1.2 *Smartphone*

Dalam proses perancangan tugas akhir ini *Smartphone* merupakan peragkat yang digunakan untuk menginstalasi *software* atau aplikasi yang nantinya akan digunakan untuk mengendalikan serta memonitoring perangkat tertanam yang dirancang pada tugas akhir ini. Spesifikasi pada *smartphone* yang digunakan yaitu prosessor *Quad-core*, *speed core* sebesar 1,2 GHz, penggunaan OS android v6.0.1 *Marsmallow*, RAM sebsar 2GB.

3.1.3 NodeMcu ESP8266

Dalam proses perancangan tugas akhir ini NodeMCU ESP8266 merupakan perangkat mikrokontroller yang dilengkapi dengan fitur GPIO, PWM, IIC, 1-*wire*, dan ADC. NodeMCU ESP8266 ini berfungsi untuk mengendalikan komponen-komponen yang ada di dalam sebuah sistem tertanam berdasarkan perintah yang tersusun dalam sebuah kode pemrograman yang telah disematkan ke dalam mikrokontroller tersebut. Pada perangkat NodeMCU ESP8266 ini telah terintegrasi deangan modul WIFI sehingga memugkinkan perangkat untuk melakukan pengiriman data melalui akses internet seperti pengiriman ke *database* pada *google firebase* ataupun ke sebuah *software* yang mendukung.

3.1.4 Motor Servo

Dalam proses perancangan sistem tertanam pada tugas akhir ini, motor servo merupakan komponen yang berfungsi untuk menggerakan membuka ataupun menutup pintu garasi berdasarkan hasil perintah yang dikirimkan melalui aplikasi *smartphone* android.

3.1.5 Sensor Api (*IR Flame Sensor*)

Dalam proses perancangan sistem tertanam pada tugas akhir ini, *ir flame sensor* merupakan sensor yang digunakan untuk mendeteksi jika terjadi nyala api di dalam garasi untuk melakukan pencegahan hal-hal yang tidak di inginkan seperti kebakaran. *Output* dari sensor ini berupa suara dari *buzzer*

dan notifikasi pada aplikasi *smartphone android* untuk memberitahu pengguna jika terdapat nyala api pada garasi.

3.1.6 *Buzzer Active*

Dalam proses perancangan sistem tertanam pada tugas akhir ini. *Buzzer* merupakan komponen yang digunakan sebagai indikator berupa bunyi. *Buzzer* mengeluarkan bunyi apabila sensor api (*ir flame sensor*) mendeteksi adanya keberadaan api di dalam garasi.

3.1.7 LED (*Light Emitting Diode*)

Dalam proses perancangan sistem tertanam pada tugas akhir ini LED digunakan sebagai pengganti fungsi lampu pada garasi yaitu sebagai penerangan. Lampu akan mati ataumpun menyala secara otomatis, LED akan menyala ketika pengguna memberikan perintah buka pintu pada aplikasi *smartphon*e, dan LED akan mati ketika pengguna memberikan perintah tutup pintu pada aplikasi *smartphon*e.

3.1.8 *Software Arduino* IDE

Dalam proses perancangan sistem tertanam pada tugas akhir ini *software* arduino IDE merupakan aplikasi yang digunakan untuk pembuatan program atau *coding* serta *uploading* dengan bahasa pemrograman bahasa C dan C++ yang akan diinputkan ke dalam mikrokontroller agar setiap komponen pada sistem dapat dikendalikan dan bekerja sesuai dengan apa yang telah diprogram sebelumnya.

3.1.9 *Google Firebase*

Dalam proses perancangan sistem tertanam pada tugas akhir ini, *Google Firebase* berfungsi sebagai *database* tempat penyimpanan data dari aplikasi maupun dari alat yang dirancang pada *cloud* dan data tersebut dapat dikirimkan ke aplikasi secara *real time.*

3.1.10 *Software* MIT App *Inventor*

Pada proses perancangan perangkat tertanam dalam tugas akhir ini, *software* MIT App *Inventor* befungsi sebagai *software* yang digunakan untuk mendesain antarmuka aplikasi *smartphone* khususnya menggunakan *platform Android* pada perancangan alat tugas akhir ini. Aplikasi untuk projek ini dirancang agar dapat melakukan kontroling membuka maupun menutup pintu garasi serta dapat memonitoring jika terdeteksi api di dalam garasi. Data yang digunakan baik dalam proses kontroling maupun monitoring diambil dari *google firebase.*

3.1.11 *Software Wireshark*

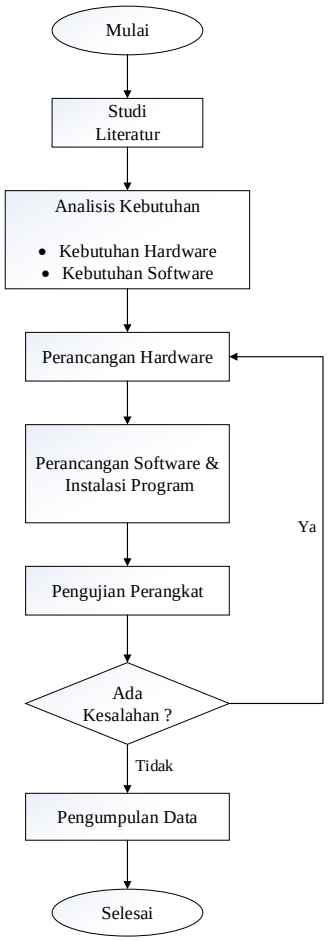
Dalam proses perancangan sistem tertanam pada tugas akhir ini, *software wireshark* versi 2.2.2 digunakan untuk memonitoring kualitas QoS yang diperoleh dari pengiriman data melalui intrnet seperti *website* ataupun aplikasi android. Untuk mendukung kinerja dari *software wireshark* perlu dilakukan penginstalan aplikasi WinCap, aplikasi ini didalamnya terdapat *driver-driver* khusus yang dipakai oleh *software wireshark*.

3.2 ALUR PENELITIAN

Pada tugas akhir ini alur penelitian yang menggunakan studi literatur, dimana pada alur penelitian ini yang pertama kali dilakukan yaitu melakukan pengumpulan data berupa referensi, referensi tersebut diperoleh dari berbagai sumber seperti buku, jurnal, serta situs website dengan sumber yang terpercaya serta berkaitan dengan judul tugas akhir yang diusung. Setelah melakukan pengumpulan data, penulis menganalisis segala kebutuhan yang diperlukan baik kebutuhan yang berbentuk *hardware* maupun *software* yang akan menunjang pembuatan projek tugas akhir ini. Setelah melakukan analisis mengenai kebutuhan projek, selanjutnya penulis melakukan perancangan perangkat yang akan dibuat dalam tugas akhir ini, perancangan alat ini meliputi perancangan perangkat keras dan perancangan perangkat lunak. Perancangan perangkat keras ini meliputi merangkai setiap komponen menjadi suatu topologi yang nantinya setiap komponen akan bekerja sesuai fungsinya masing-masing, dan pada perancangan perangkat lunak meliputi

pembuatan serta memasukan kode pemrograman pada alat yang dibuat pada projek tugas akhir ini dan pembuatan aplikasi android untuk mendukung sistem kerja alat tersebut, pada proses perancangan perangkat lunak ini dibuat sebuah *flowchart* yang didalamnya terdapat tahapan proses berjalannya perangkat.

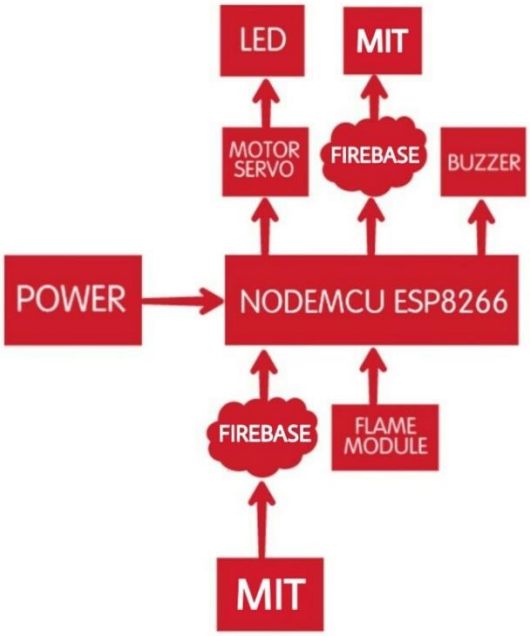
Setelah melakukan perancangan perangkat, selanjutnya penulis melakukan pengujian terhadap perangkat yang telah dibuat dalam projek tugas akhir ini. Pengujian perangkat dilakukan bertujuan untuk melihat kinerja dari perangkat yang telah dibuat apakah sudah sesuai dengan yang direncanakan atau masih terdapat problem. Pengujian dilakukan dengan meneliti kinerja dari setiap blok sistem secara menyeluruh, jika ditemukan adanya *error* pada perangkat maka dilakukanlah suatu tindakan *troubleshooting* untuk mengatasi permasalahan tersebut. Setelah dilakukan pengujian perangkat, maka proses selanjutnya yaitu dilakukannya pengambilan data hasil dari kinerja perangkat. Data yang di ambil berupa nilai yang dihasilkan dari parameter-patrameter tertentu yang terdapat pada sistem.



Gambar 3.1 *Flowchart* Alur Penelitian

3.2.1 Blok Diagram Sistem Perangkat Keras

Dalam tahapan ini terdapat penjelasan mengenai perancangan sistem yang meliputi, perancangan perangkat keras, perancangan perangkat lunak dan pemodelan struktur data. Untuk blok diagram sistem garasi pintar dapat dilihat pada gambar 3.2 di bawah ini.



Gambar 3.2 Blok Diagram Sistem Garasi Pintar

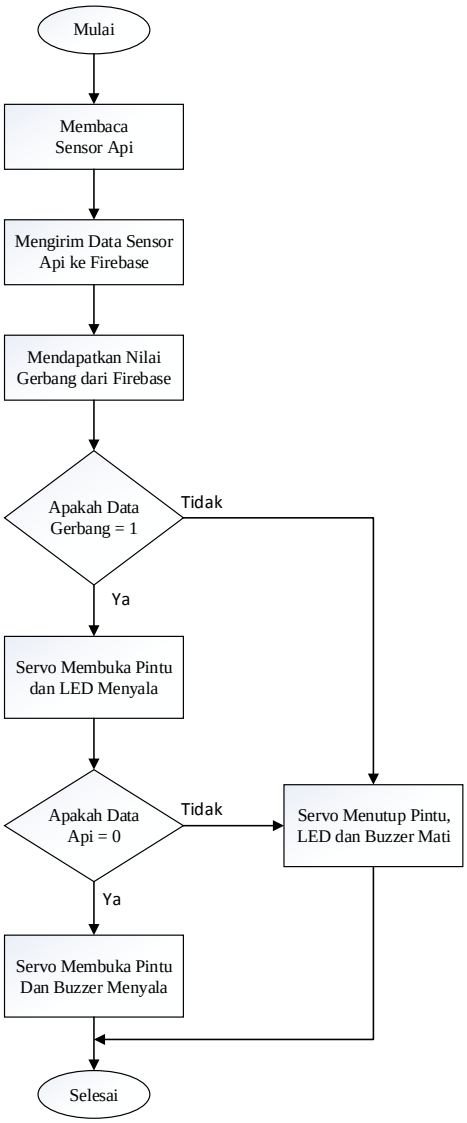
Pada gambar 1 merupakan blok diagram perancangan sistem garasi pintar, NodeMCU esp8266 diberikan sumber tegangan dari *power supply* kemudian, NodeMCU esp8266 memperoleh data dari *firebase* yang data tersebut berisi perintah dari aplikasi (MIT), data yang diperoleh tersebut merupakan perintah buka atau tutup garasi. Setelah data diolah oleh NodeMCU esp8266 lalu data menuju *output* yang berupa komponen motor servo, jika data berbentuk perintah buka maka motor servo akan bergerak membuka pintu lalu LED akan menyala, begitupun sebaliknya pada saat terdapat perintah tutup. Pada saat kondisi motor servo membuka pintu maka NodeMCU esp8266 akan mengirimkan data ke aplikasi (MIT) melalui *firebase* , lalu pada aplikasi muncul notifikasi bahwa pintu garasi terbuka. Selanjutnya pada komponen sensor api (*ir flame sensor*), sensor akan mengirimkan data ke NodeMCU esp8266 jika sensor mendeteksi keberadaan api di dalam garasi lalu NodeMCU esp8266 meneruskan data tersebut ke *output* yang berupa *buzzer* dan meneruskan ke *firebase* dan muncul notifikasi pada aplikasi.

Aplikasi *smartphone* disini berfungsi untuk mengontrol membuka dan menutup pintu garasi serta memonitoring garasi untuk mencegah terjadinya kebakaran.

3.2.2 *Flowchart* Alur Sistem

Dalam tahapan ini terdapat penjelasan mengenai perancangan perangkat lunak pada projek tugas akhir ini, perancangan perangkat lunak digambarkan melalui *flowchart* sebagai sistematika dari perancangan alat. Berikut merupakan *flowchart* alur perangkat lunak pada mikrokontroler dan *flowchart* alur perangkat lunak pada aplikasi android :

a. *Flowchart* Alur Perangkat Lunak Pada Mikrokontroler

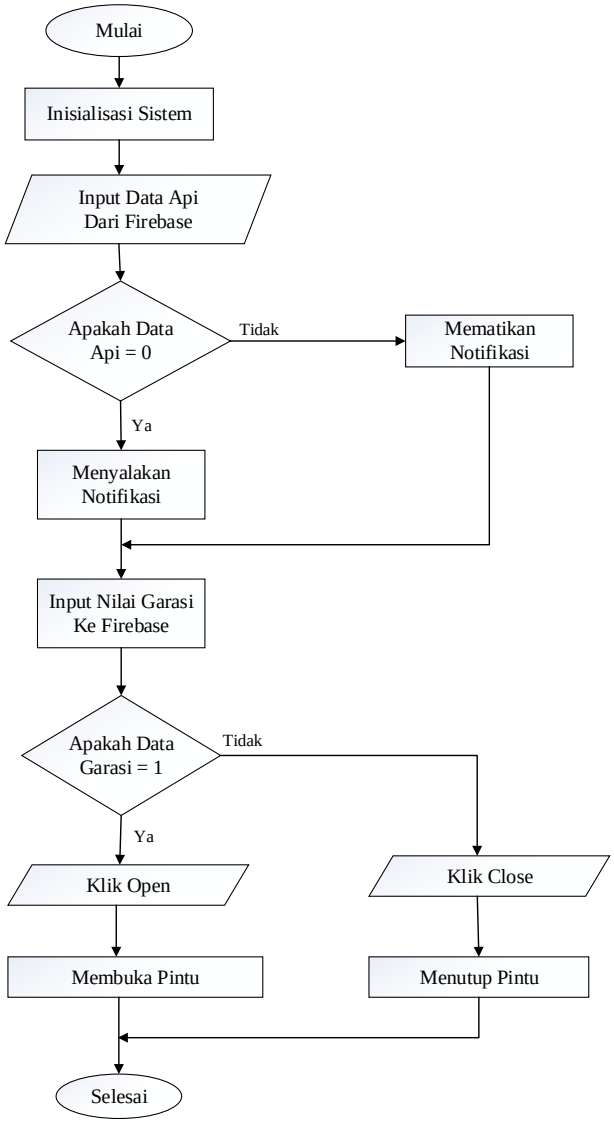


Gambar 3.3 *Flowchart* Alur Perangkat Mikrokontroler

Diagram alur pada gambar 3.3 menunjukan alur kerja mikrokontroler sesuai dengan pemrograman pada perangkat lunak yang akan dirancang, dimana mikrokontroler menjadi pusat dari pengolahan data pada sistem. Pada sistem garasi pintar ini mikrokontroler mendapat *input* dari aplikasi android berupa perintah menutup atau membuka pintu garasi. Pada saat dimasukan perintah *Open* pada aplikasi android maka mikrokontroler akan meneruskan perintah tersebut ke perangkat *output* yang berupa motor servo, lalu motor servo akan melakukan fungsi bergerak membuka pintu garasi, setelah pintu garasi terbuka maka dengan otomatis lampu LED akan menyala untuk menerangi garasi, dan dengan waktu yang sama muncul notifikasi pada aplikasi yang memberitahu bahwa pintu garasi terbuka, notifikasi ini muncul agar pengguna garasi segera menutup kembali pintu garasi.

Jika sebaliknya pada saat dimasukan perintah *Close* pada aplikasi android maka mikrokontroler akan memerintahkan motor servo untuk menutup pintu garasi, setelah pintu garasi tertutup maka dengan otomatis lampu LED akan mati karena tidak diperlukan lagi. Selain melakukan kontroling pintu garasi, pada sistem ini juga terdapat fungsi monitoring kebakaran pada garasi. *Ir flame sensor* akan mendeteksi keberadaan api di dalam garasi, jika sensor mendeteksi keberadaan api maka sensor mengirimkan data ke mikrokontroler lalu mikrokontroler memerintah *buzzer* untuk aktif dan dengan waktu yang bersamaan muncul notifikasi pada aplikasi android yang memberitahukan adanya potensi kebakaran di dalam garasi.

b. *Flowchart* Alur Perangkat Lunak Android



Gambar 3.4 *Flowchart* Alur Perangkat Lunak Android

Pada *flowchart* perangkat lunak android pada gambar 3.4 untuk memulainya langkah pertama yang dilakukan yaitu inisialisasi sistem, selanjutnya menerima data api dari *firebase*, jika data api = 0 maka muncul notifikasi pada aplikasi, selanjutnya *input* nilai garasi ke *firebase*, jika data garasi = 1 maka klik *open* dan membuka pintu, dan jika data garasi = 0 maka klik *close* dan menutup pintu.

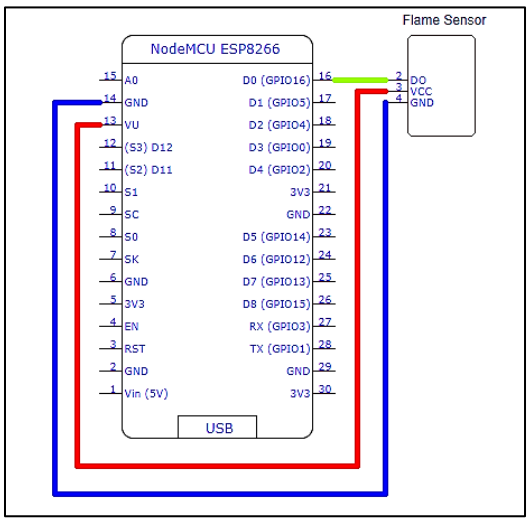
Ada dua jenis perintah yang digunakan yaitu *Open* dan *Close*, perintah *Open* untuk membuka pintu garasi sedangkan perintah *Close* untuk menutup pintu garasi. Setelah perintah *Open* dijalankan maka pintu garasi akan terbuka dan dalam waktu yang bersamaan perangkat akan mengirimkan data ke aplikasi untuk memberikan notifikasi. Pada sisi monitoring aplikasi akan menerima data dari mikrokontroler

yang berupa data *input* dari sensor api, jika sensor mendeteksi adanya api maka perangkat akan mengirimkan data ke aplikasi untuk memunculkan notifikasi.

3.2.3 Perancangan Perangkat Keras

a. Antarmuka NodeMCU dengan *IR Flame Sensor*

Dalam antarmuka ini *flame sensor* menggunkaan pin digital pada NodeMCU yaitu pada pin D0, VCC 5V, dan GND. *Flame sensor* ini berfungsi untuk mendeteksi nyala api pada daerah yang dapat dicakup, prinsip kerjanya yaitu sensor ini akan mendeteksi keberadaan api dengan membedakan spektrum cahaya *infrared* pada api yang dideteksinya dengan panjang spektrum tertentu berkisar antara 760 – 1100 nm. Sensitivitas sensor ini dapat diatur menggunkan *potensiometer*, namun sensor ini hanya dapat mengetahui adanya api atau tidak tetapi tidak dapat mengetahui dimana keberadaan api.



Gambar 3.5 Koneksi Pin NodeMCU dengan *IR Flame Sensor*

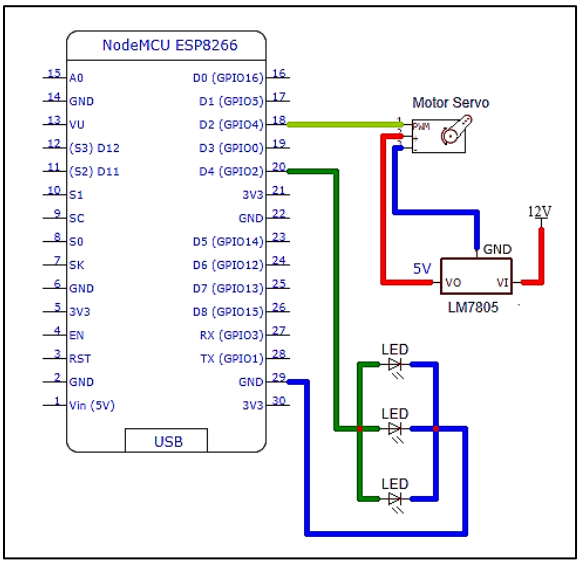
Tabel 3.2 Koneksi antara NodeMCU dengan *IR Flame Sensor*

No.	PIN	Fungsi
1	D0	Pembacaan sensor api di *port* D0
2	VCC	Catu daya 5V
3	GND	*Grounding*

b. Antarmuka NodeMCU dengan Motor Servo dan LED

Dalam antarmuka ini motor servo menggunkan pin digital pada NodeMCU yaitu pin D2, VCC 5V, dan GND serta terdapat LED yang menggunakan pin

digital D4 pada positif dan GND untuk negatif. Motor servo ini berfungsi sebagai penggerak pintu garasi apabila diberikan perintah melalui aplikasi, sedangkan LED berfungsi sebagai lampu garasi yang mana jika motor servo bergerak membuka pintu maka LED akan menyala secara otomatis dan jika motor servo bergerak menutup pintu garasi maka LED akan mati namun jika dalam kondisi motor servo membuka pintu ketika sensor api mendeteksi adanya api maka LED akan tetap mati.



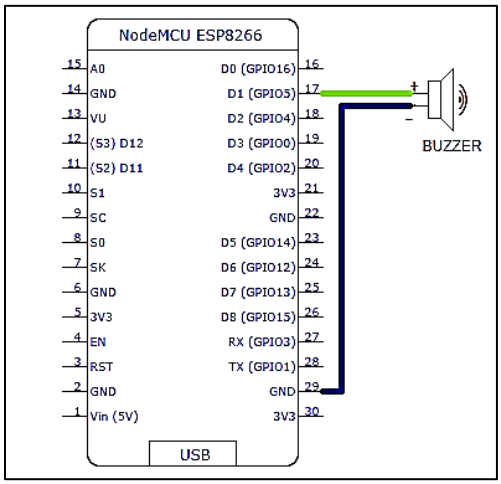
Gambar 3.6 Koneksi Pin NodeMCU dengan Motor Servo dan LED

Tabel 3.3 Koneksi antara NodeMCU dengan Motor Servo dan LED

No.	PIN	Fungsi
1	D2	Pin data motor servo di *port* D2
2	D4	Kutub positif LED di D4
3	VCC	Catu daya 5V untuk motor servo
4	GND	*Grounding*

c. Antarmuka NodeMCU dengan *Buzzer*

Dalam antarmuka ini *buzzer* menggunakan pin digital pada NodeMCU yaitu pin D1 pada positif dan GND pada negatifnya. *Buzzer* ini akan aktif dengan mengeluarkan bunyi ketika *flame sensor* mendeteksi adanya api, dan jika *flame sensor* tidak mendeteksi api maka *buzzer* berada dalam kondisi *low*.



Gambar 3.7 Koneksi Pin NodeMCU dengan *Buzzer*

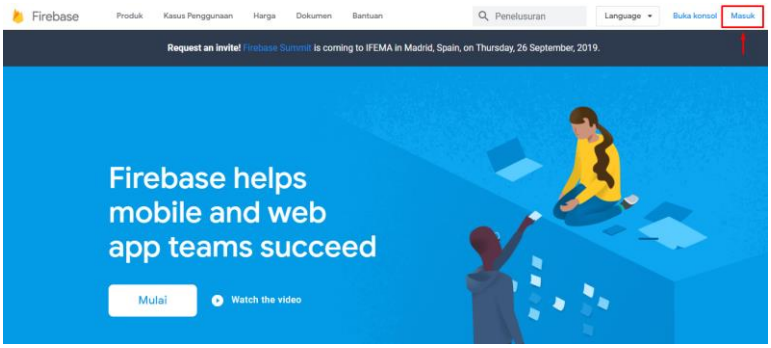
Tabel 3.4 Koneksi antara NodeMCU dengan *Buzzer*

No.	PIN	Fungsi
1	D1	Kutub positif *buzzer* di *port* D1
2	GND	*Grounding*

3.2.4 Perancangan Perangkat Lunak

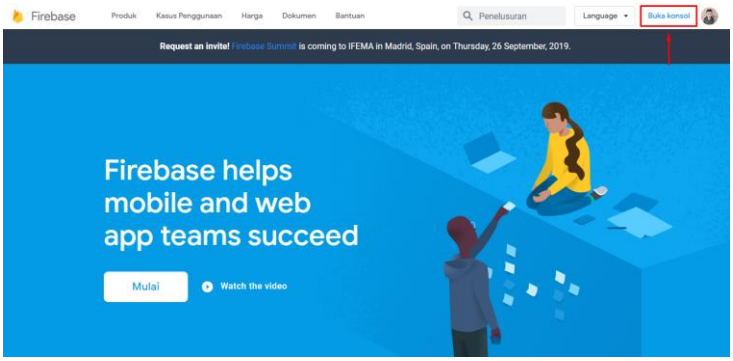
a. *Google Firebase*

Pada perancangan perangkat lunak ini *google firebase* berfungsi sebagai *database* tempat penyimpanan data yang berkaitan dengan aplikasi yang dibuat dan datanya dapat dikirim ke *client* secara *real time* sehingga aplikasi yang dibuat dapat menampilkan notifikasi kondisi terkini dari perangkat yang terhubung dengan aplikasi. Untuk memulai menggunakan *google firebase* pengguna harus mendaftar terlebih dahulu dengan memilih menu Masuk pada tampilan awal *google firebase* seperti pada gambar 3.8 yang ditandai dengan panah.



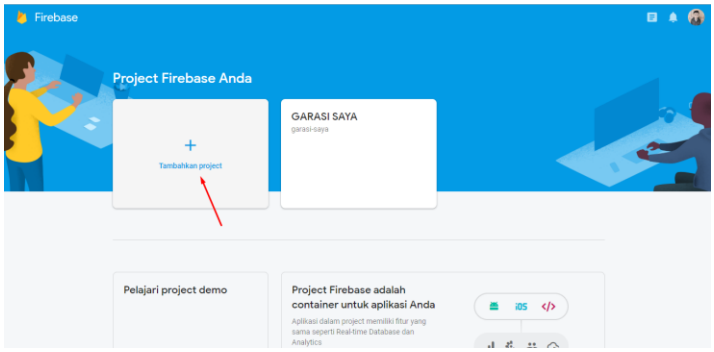
Gambar 3.8 Tampilan Awal *Firebase*

Setelah berhasil masuk ke *google firebase* lalu pilih buka konsol pada pojok kanan atas layar untuk memulai membuat projek , tampilannya dapat dilihat pada Gambar 3.9.



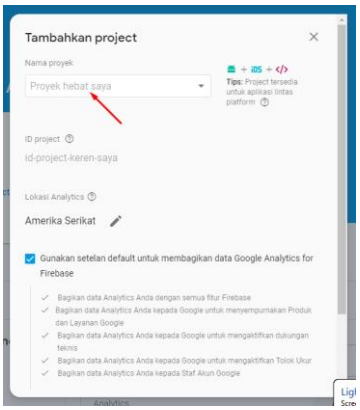
Gambar 3.9 Buka Konsol

Setelah membuka konsol langkah selanjutnya yaitu pilih tambahkan *project* seperti pada gambar 3.10 yang diberi tanda panah.



Gambar 3.10 Menu Tambahkan *Project*

Lalu beri nama *project* yang akan dibuat dengan mengisi nama project, setelah itu isi semua persetujuan dan klik *create project.*

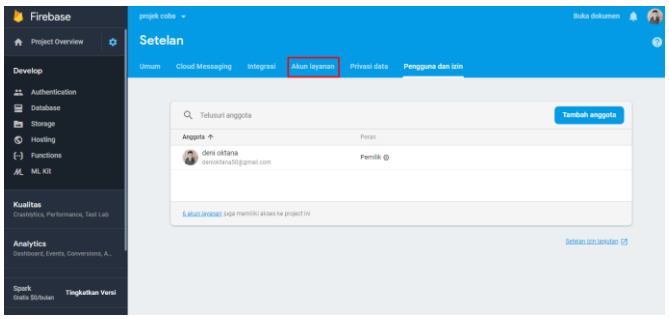


Gambar 3.11 Memasukan Nama *Project*

Setelah nama *project* dimasukan selanjutnya ditampilkan menu awal dari laman *project firebase*, setelah itu pilih *project overview* lalu pilih pengguna dan izin.

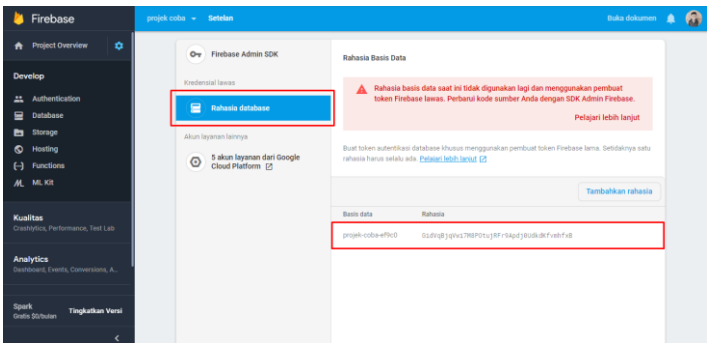
Gambar 3.12 Tampilan Awal *Project*

Setelah memilih pengguna dan izin maka akan muncul tampilan di layar seperti pada gambar 3.13, dan setelah itu pilih akun layanan.



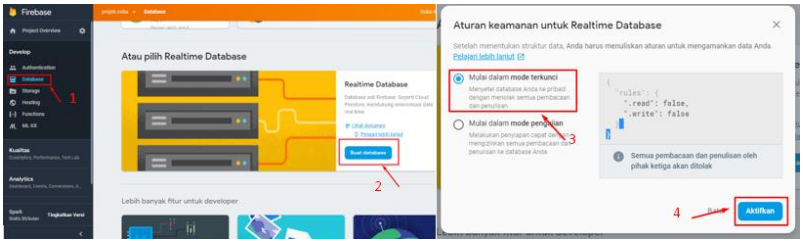
Gambar 3.13 Tampilan Setelan

Setelah memilih akun layanan lalu pilih rahasia *database* kemudian pilih tampilkan untuk menampilkan kode yang tersembunyi, kode tersebut akan digunakan untuk mengisi *firebase authentication* pada aplikasi Arduino IDE.



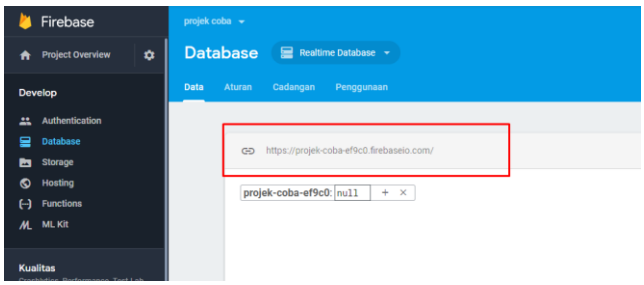
Gambar 3.14 Tampilan Akun Layanan

Setelah proses pada akun layanan selesai langkah selanjutnya yaitu pindah ke menu *database*, setelah sampai ke menu *database* selanjutnya yaitu pilih *real time database*, jika telah masuk pada *real time database* kemudian pilih mulai dari mode terkunci lalu pilih aktifkan, langkah tersebut dapat dilihat pada gambar 3.15.



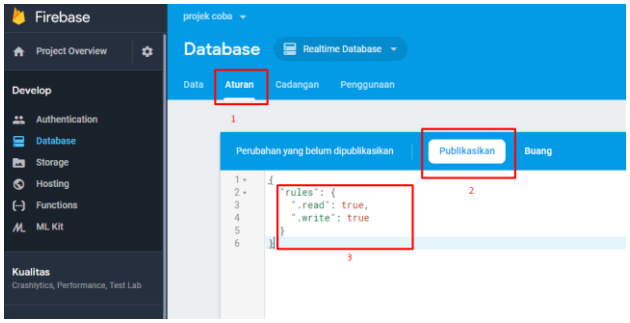
Gambar 3.15 Membuat *Realtime Database*

Pada gambar 3.16 ditampilkan menu awal *real time database*, pada menu ini terdapat link yang akan disalin untuk mengisi *firebase host* pada aplikasi Arduino IDE, dan juga untuk *firebase token* pada MIT App Inventor.



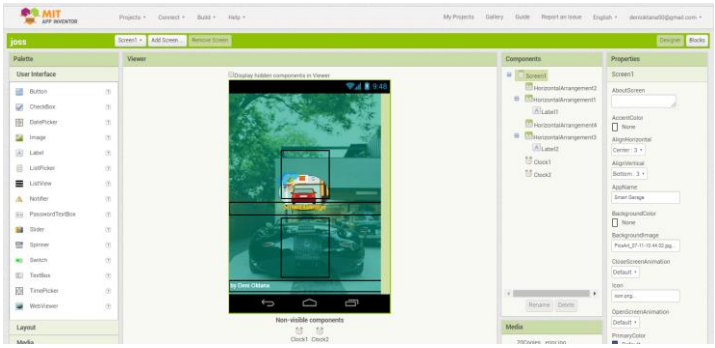
Gambar 3.16 Tampilan *Realtime Database*

Masih pada menu *realtime database* kemudian pilih aturan seperti pada gambar 3.17, pada aturan ini berisi mengenai format pembacaan nilai yang dikirim ke *database*. Lakukan perubahan pada perintah yang ada dari *false* diubah menjadi *true*, tampilannya dapat dilihat pada gambar 3.17.



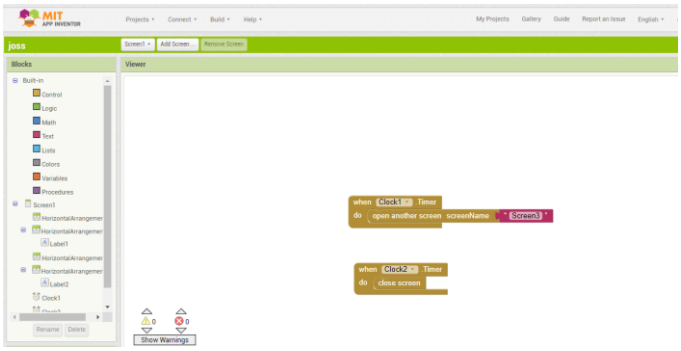
Gambar 3.17 Tampilan *Rules* Pada *Database*

b. Desain Aplikasi MIT *App Inventor*



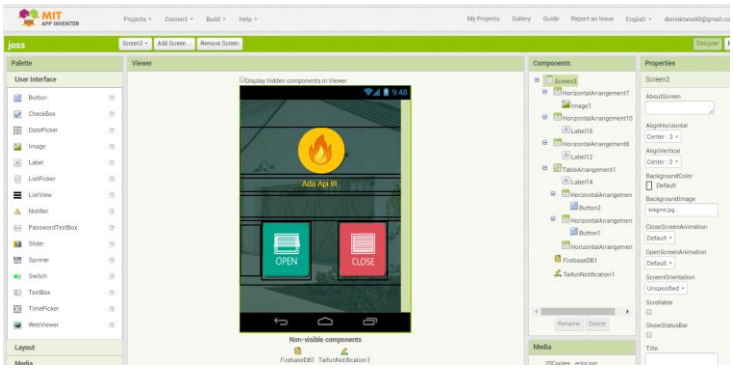
Gambar 3.18 Tampilan Awal *Screen* Aplikasi

Pada gambar 3.18 merupakan tampilan dari MIT App Inventor pada *screen* 1, tampilan ini yang nantinya akan menjadi tampilan awal pada saat aplikasi baru terbuka atau *flash screen* yang durasinya hanya 3 detik.



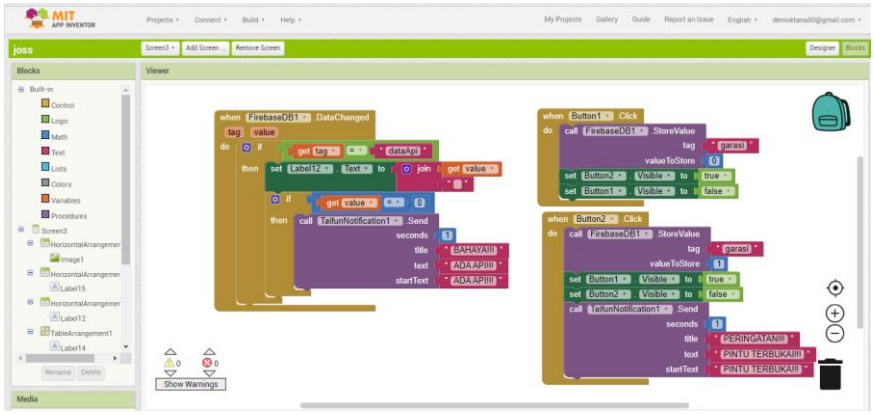
Gambar 3.19 *Block Screen* 1 Aplikasi

Tampilan pada gambar 3.19 berisi *block* pada screen 1, *block* ini berfungsi untuk memberi perintah menampilkan *flash screen* dengan durasi 3 detik setelah itu akan menampilkan *screen 2.*



Gambar 3.20 Tampilan *Screen* 2 Kontroling dan Monitoring

Tampilan pada gambar 3.20 berisi desain pada *screen* 2, pada *screen* 2 berisi *button* untuk melakukan kontroling pintu garasi dan juga terdapat notifikasi apabila *flame sensor* mendeteksi adanya api.



Gambar 3.21 *Block Screen* 2 Aplikasi

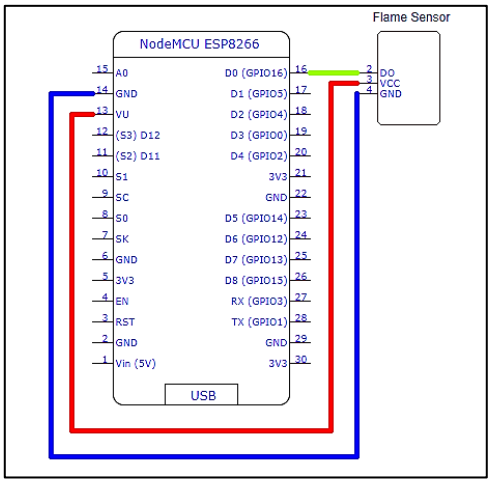
Tampilan pada gambar 3.21 berisi *block* yang digunakan untuk membangun sistem pada *screen* 2 agar dapat terhubung dengan *database* pada *google firebase* sehingga aplikasi dapat melakukan kontroling dan monitoring alat yang dibuat.

3.2.5 Prosedur Pengujian *Hardware*

Pengujian *hardware* dilakukan untuk mengetahui kualitas kinerja dari komponen yang digunakan dalam sistem apakah komponen tersebut telah sesuai dengan spesifikasi atau tidak. Dalam pengujian *hardware* ini dilakukan dengan mengambil data yang dihasilkan dari hasil kerja komponen yang telah dilakukan pengujian. Dalam pengujian *hardware* ini memerlukan peralatan yang dapat membantu proses pengujian seperti, *software* Arduino IDE, meteran, dan osiloskop.

a. Prosedur Pengujian *Flame Sensor* untuk Mendeteksi Api

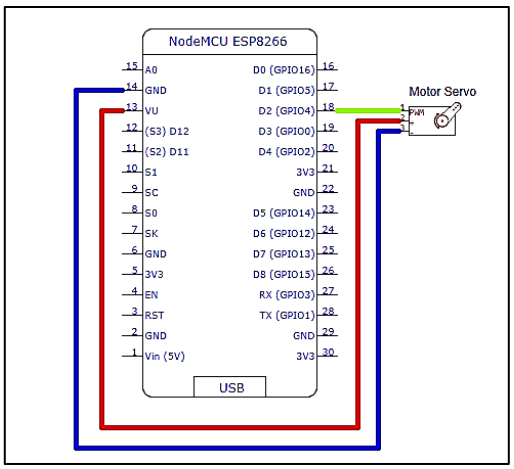
Pengujian *flame sensor* dilakukan untuk mengetahui kinerja dari sensor tersebut dalam mendeteksi keberadaan api. Pengujian dilakukan dengan menyalakan pematik api pada daerah kerja sensor lalu menguji seberapa jauh letak api serta besar sudut yang dapat dijangkau sensor dengan memindahkan api menjauh dari sensor dan diukur menggunakan meteran serta busur.



Gambar 3.22 Koneksi Pin NodeMCU dengan *IR Flame Sensor*

b. Prosedur Pengujian Perangkat Motor Servo

Pengujian motor servo dilakukan untuk mengetahui baik atau tidaknya kinerja motor servo yang digunakan, dikarenakan motor servo beroperasi berdasarkan besar sudut yang dimasukan dalam program, maka pengujiannyapun dilakukan dalam 2 kondisi, yaitu dalam kondisi 0° dan 90°. Pengujian ini dilakukan dengan menghubungkan osiloskop dengan motor servo yang telah terkoneksi dengan NodeMCU dan catu daya untuk menemukan *duty cycle* dan *interval pulse* dalam 1 gelomang.

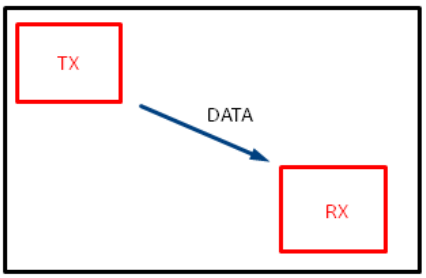


Gambar 3.23 Koneksi Pin NodeMCU dengan Motor Servo

3.2.6 Prosedur Pengujian *Quality of Service* (QOS)

Pengujian QOS dilakukan untuk menguji kelayakan konektifitas pada sistem dalam pengiriman data dari *transmitter* ke *receiver* dengan memerhatikan beberapa parameter seperti *delay*, *packet loss*, dan *throughput.*

Pengujian QOS ini dilakukan dengan cara membuat skema pada sisi Tx terdapat NodeMCU yang berfungsi sebagai *access point* dan pada sisi Rx terdapat perangkat laptop yang telah terdapat aplikasi *Wireshark* dan terhubung ke *access point* NodeMCU lalu pergerakan data dapat dilihat pada aplikasi *Wireshark* dan dilakukanlah proses *capture* untuk membuat hasil data.



Gambar 3.24 Skema Pengujian QOS

a. Prosedur Pengujian *Delay*

Pada pengujian ini betujuan untuk melihat hasil dari total waktu tunda pada saat pengiriman paket data dari Tx ke Rx. Pengujian dilakukan dengan mengamati total selisih waktu data yang sampai di sisi Rx dengan data yang diterima selanjutnya pada kondisi durasi waktu dan jarak yang berbeda.

b. Prosedur Pengujian *Packet Loss*

Pada pengujian ini bertujuan untuk mengamati jumlah total paket data yang hilang pada saat pengiriman data dari Tx menuju Rx. Pengujian dilakukan dengan percobaan pengiriman data dari Tx menuju Rx dengan durasi waktu dan jarak yang berbeda dan hasilnya dapat dilihat pada sisi Rx dengan aplikasi *Wireshark* dengan melihat parameter *loss.*

c. Prosedur Pengujian *Throughput*

Pengujian ini bertujuan untuk mengamati laju data yang dikirim dari Tx menuju Rx. Pengujian dilakukan dengan percobaan pengiriman data dari Rx ke Tx dengan durasi waktu dan jarak yang berbeda setelah itu hasilnya dihitung dengan persamaa berikut ini :

$$Throughput = \frac{\text{Jumlah Besar Data}}{Delay} \times 8$$