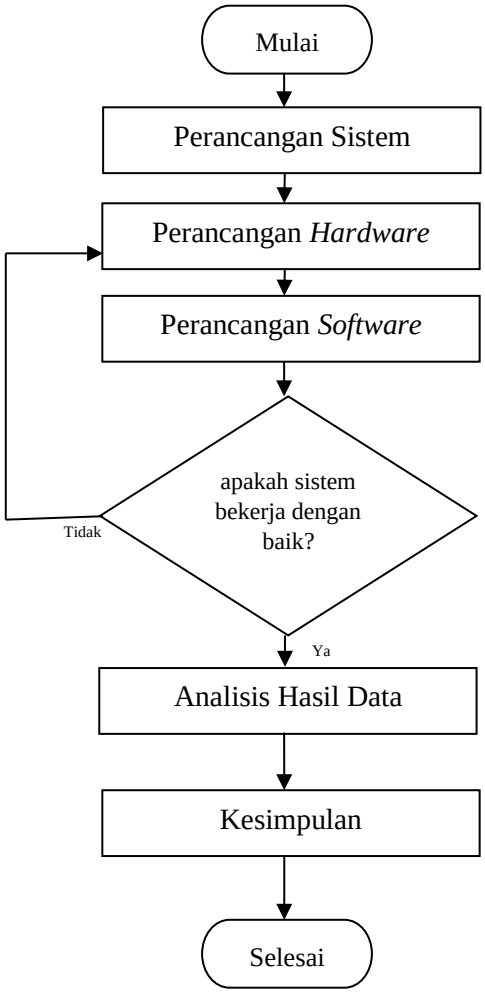
BAB 3
METODE PENELITIAN

3.1 ALUR PENELITIAN



Gambar 3.1 Flowchart Alur Penelitian

Pada Gambar 3.1 merupakan *flowchart* alur penelitian yang akan dilakukan pada penelitian ini. Pada tahap pertama yaitu perancangan sistem, perancangan sistem merupakan tahap identifikasi komponen apa saja yang akan digunakan dalam perancangan dan bagaimana cara memodelkan sebuah permasalahan yang akan diselesaikan oleh sistem. Pada tahap kedua yaitu perancangan *hardware*, pada tahap ini penulis merancang alat berdasarkan komponen apa saja yang digunakan untuk pembuatan alat prototipe. Pada tahap ketiga yaitu perancangan *software*, pada tahap ini penulis melakukan perancangan *software* dengan Menyusun *script* yang

akan dilakukan pada tiap komponen yang telah diprogram pada *software* Arduino IDE. Pada tahap keempat yaitu melakukan pengujian sistem alat hasil perancangan, jika pada saat pengujian terdapat ketidak sesuaian dengan parameter, maka akan dilakukan perbaikan perancangan *hardware* serta perancangan *software* kembali hingga pengujian tersebut berhasil. Apabila pada tahap pengujian sistem telah berhasil maka akan dilakukan pengambilan data untuk dianalisis hingga penulis dapat mengambil kesimpulan dari pengujian sistem.

3.2 ALAT DAN BAHAN

Pada perancangan penelitian ini akan membutuhkan beberapa alat dan bahan untuk menyusun perangkat keras (*hardware*) dan perangkat lunak (*software*). Pada perangkat keras (*hardware*) akan dijelaskan tentang gambar perancangan dan sistem yang akan dibuat. Sedangkan pada perancangan perangkat lunak (*software*) akan dijelaskan tentang alur diagram pemrograman pada monitoring dan alur diagram pada aplikasi android untuk menampilkan data dari *firebase*. Daftar alat dan bahan untuk Menyusun perangkat *hardware* dan perangkat *software* dapat dilihat pada Tabel 3.1.

Tabel 3.1 Daftar Alat dan Bahan

No	Alat Dan Bahan	Jumlah
1	Radiator Mobil	1
2	*Reservoir* Radiator	1
3	Laptop	1
4	*Smartphone*	1
5	NodeMCU ESP32	1
6	Sensor *Water Level*	1
7	Sensor DS18B20	1
8	*Software* Arduino IDE	1
9	*Software* MIT App Inventor	1
10	*Software Wireshark*	1
11	Google *Firebase*	1
12	*Water Pump*	1

3.2.1 Radiator mobil

Pada penelitian ini menggunakan radiator mobil sebagai objek penelitian yang akan dilakukan, baik dari segi radiator mobil maupun tangki penyimpanan air radiator mobil.

3.2.2 Water Pump

Pada penelitian ini *Water Pump* digunakan untuk mengalirkan air pada wadah agar air bisa memutar melalui radiator dan akan kembali lagi ke wadah.

3.2.3 Reservoir Radiator

Pada penelitian ini *reservoir* yang digunakan adalah reservoir motor yang berfungsi untuk menampung air cadangan radiator.

3.2.4 Laptop

Pada penelitian ini, laptop dipergunakan sebagai alat dalam mengolah seluruh bahan data yang ada. Laptop juga digunakan untuk mengkoding pada seluruh komponen serta sebagai media dalam pengambilan hasil data. Spesifikasi laptop yang digunakan meliputi prosesor Intel(R) Core (TM) i7, kecepatan clock sebesar 2.40Ghz, memiliki memori sebesar 500GB memori RAM sebesar 8192MB.

3.2.5 Smartphone

Pada penelitian ini, *smartphone* dipergunakan sebagai alat yang mempunyai aplikasi android dalam memonitoring sistem dari perancangan alat. Spesifikasi pada *smartphone* yang digunakan meliputi CPU *Quad-core* Max 1.40 GHz, versi android 7.1.2 Nougat MIUI 11, RAM sebesar 3 GB dan ROM sebesar 32 GB.

3.2.6 Sensor *Dallas* DS18B20

Pada penelitian ini sensor dalas DS18B20 digunakan untuk memonitoring suhu pada radiator mobil yang nantinya hasil akan dikirimkan ke *database* google *firebase*

3.2.7 Sensor Water Level

Pada penelitian ini sensor *water level* digunakan pada sistem sebagai pendeteksi air pada tanki cadangan radiator. Nilai ketinggian tersebut kemudian dimonitoring melalui aplikasi pada android.

3.2.8 *Software* Arduino IDE

Pada penelitian ini, *software* Arduino IDE digunakan untuk memprogram sistem pada masing-masing komponen perangkat yang digunakan.

3.2.9 *Software* MIT App Inventor

Pada penelitian ini, *software* MIT App Inventor digunakan untuk merancang dan membuat aplikasi android untuk memonitoring nilai pada sensor *Water level* dan sensor *Dallas* DS18B20.

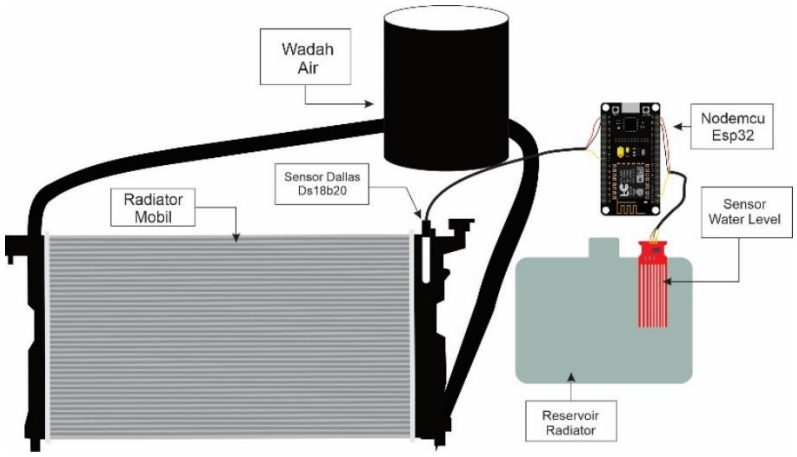
3.2.10 Software Wireshark

Pada penelitian ini, *software wireshark* digunakan untuk mendapatkan hasil dari *QoS* saat menjalankan pengiriman data menggunakan internet pada jaringan *wifi* dengan cara memfilter ip yang digunakan oleh *firebase.* Cara melakukan filter pada *wireshak* yaitu dengan printah ip.addr==34.120.160.131 pada kolom filter, sehingga nantinya akan muncul data pengirim dan penerima pada ip tersebut.

3.2.11 Google Firebase

Pada penelitian ini, Google *Firebase* digunakan untuk *database* menyimpan hasil data dari perancangan sistem secara sistematis, kemudian data akan diteruskan ke aplikasi android.

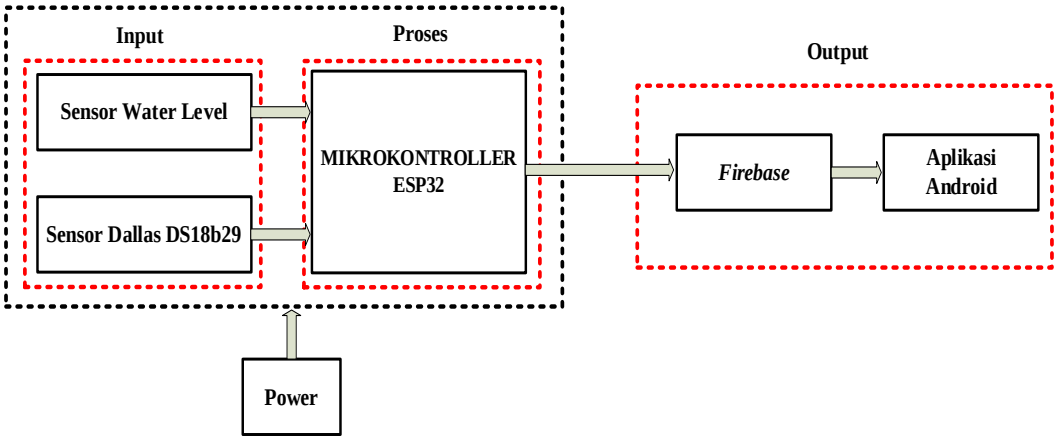
3.3 PERANCANGAN SISTEM



Gambar 3.2 Perancangan Perangkat Sistem

Pada Gambar 3.2 perancangan perangkat sistem ini dibuat untuk melakukan monitoring suhu dan ketinggian air pada prototipe radiator mobil. Perancangan perangkat sistem pada gambar 3.2 terdiri dari radiator mobil, *reservoir* radiator, mikrokontroler nodemcu ESP32, sensor dallas DS18B20 dan sensor *water level* yang nantinya akan di monitoring menggunakn aplikasi android.

3.3.1 Blok Diagram Sistem

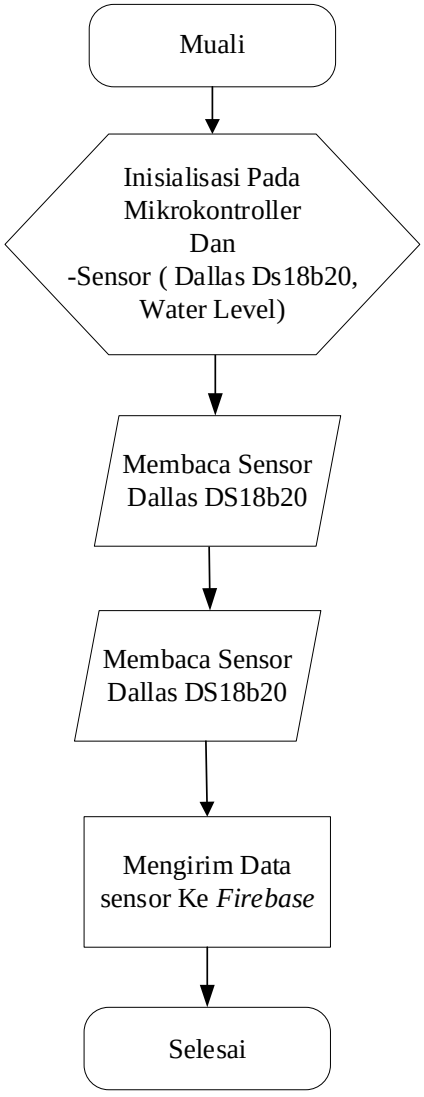


Gambar 3.3 Blok Diagram Sistem

Pada Gambar 3.3 merupakan blok diagram perancangan yang akan dibuat dimana nodeMCU ESP32 berfungsi sebagai mikrokontroller untuk menggerakan sistem. Pada penelitian ini ada beberapa *input* yang digunakan seperti sensor

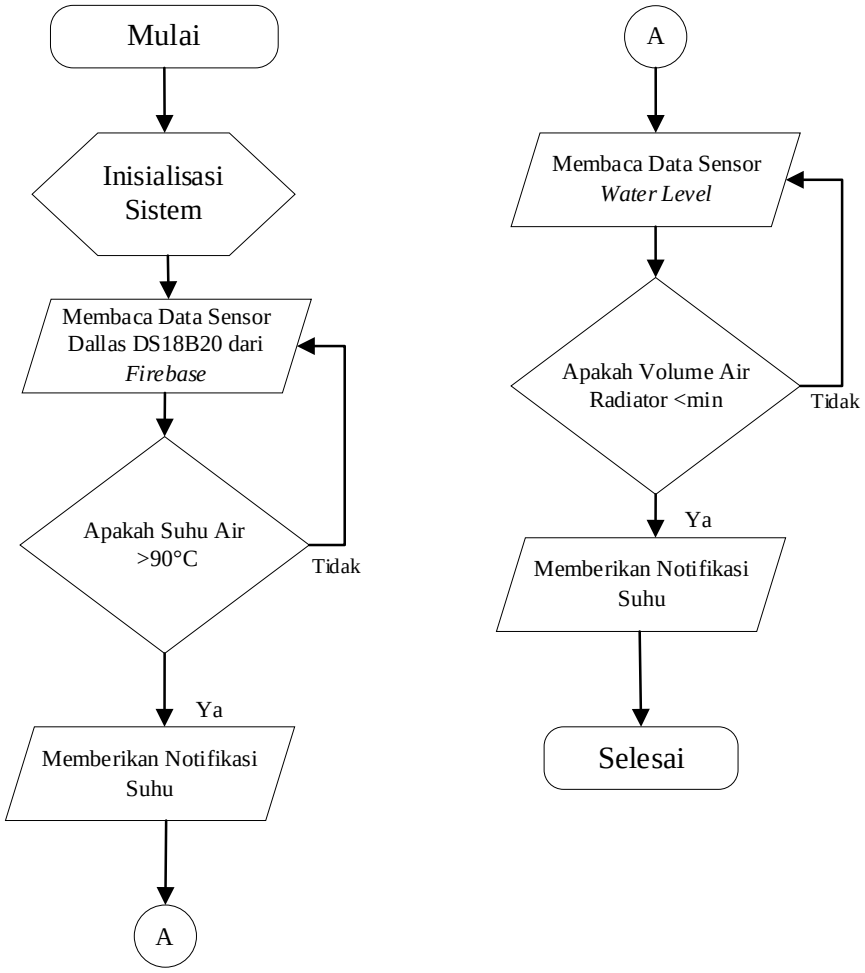
DS18B20 dan *Water Level* untuk mengukur suhu air radiator dan *level minimum* tangki air cadangan radiator. Kemudian data yang terbaca oleh sensor dikirimkan ke nodeMCU ESP32 untuk diproses. Data yang diproses oleh mikrokontroller nodeMCU ESP32 diteruskan ke output sistem. Kemudian google *firebase* sebagai *realtime database* yang akan diteruskan ke platform MIT App Inventor dan ditampilkan dalam bentuk aplikasi android. Aplikasi android akan menampilkan kondisi pembacaan sensor seperti suhu air radiator, dan *water level*. Selain itu aplikasi android berfungsi untuk memberikan notifikasi apabila nilai suhu, dan *level* minimum air pada tangki cadangan tidak pada kondisi normal.

3.3.2 Flowchart Sistem



Gambar 3.4 *Flowchart* Sistem Mikrokontroller

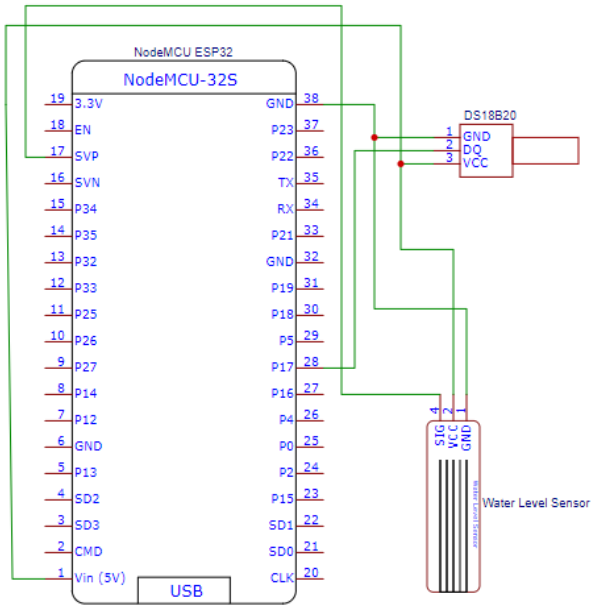
Pada Gambar 3.4 *Flowchart* sistem mikrokontrller menunjukan alur kerja sistem mikrokontroller sesuai dengan perangkat lunak yang akan dirancang. Mikrokontroller akan melakukan tugas sesuai program yang dimasukan untuk menginisialisasi variabel, sensor yang digunakan (*Dallas* DS18B20 dan *Water Level*). Proses selanjutnya adalah membaca sensor *Dallas* DS18B20 untuk mengetahui kondisi suhu air pada radiator, kemudian membaca sensor *water level* untuk mengetahui kondisi ketinggian air pada tabung *reservoir* radiator. Setelah semua data sensor terbaca kemudian oleh nodemcu ESP32 dikirim ke *Firebase* sebagai *realtime database* menggunakan komunikasi *wifi* dan jaringan *internet*. Data pada *firebase* akan ditampilkan ke aplikasi android menggunakan *platform* mit app.



Gambar 3.5 *Flowchart* Sistem Aplikasi Android

Pada Gambar 3.5 merupakan alur *flowchart* aplikasi android dimana proses pertama yaitu *inisialisasi* sistem untuk pemberian nilai awal kondisi yang dilakukan saat deklarasi variabel. Selanjutnya yaitu menerima data sensor *dallas* DS18B20 dari google *firebase*. Apabila suhu air radiator yang terbaca sensor lebih dari 90°C maka akan memunculkan notifikasi pada aplikasi android " Mobil Terlalu Panas " apabila sensor membaca suhu kurang dari 40°C maka akan muncul notifikasi " Panaskan Mobil " jika tidak maka akan kembali menerima data sensor *dallas* DS18B20 dari *google firebase*. Kemudian menerima data sensor *water level* dari *google firebase*. Apabila volume air pada tabung *reservoir* radiator kurang dari garis minimal maka akan memunculkan notifikasi pada aplikasi andorid "Air *Reservoir* Habis" jika tidak kembali menerima data sensor *water level* dari google *firebase*.

3.3.3 Perancangan Perangkat Keras



Gambar 3.6 Perancangan Skematik Perangkat Keras

Pada gambar 3.6 Perangkat Skematik perangkat keras yang akan digunakan dallam penelitian ini menggunakan beberapa komponen seperti sensor *dallas* DS18B20 dan sensor *Water level*. Dalam penelitian ini mikrokontroller yang digunakan adalah nodemcu ESP32 yang berfungsi sebagi otak dalam menjalankan

sistem. Berikut daftar table dari komponen yang terdeteksi dengan nodemcu ESP32:

Tabel 3.2 Koneksi Pin Sensor *Dallas* DS18B20 Dengan Nodemcu ESP32

No	Pin *Dallas* DS18B20	Fungsi
1	Data	Pembacaan data sensor *dallas* DS18B20 di port GPIO17 Nodemcu ESP32
2	VCC	Catu daya 5 V dari Nodemcu ESP32
3	GND	*Grounding*

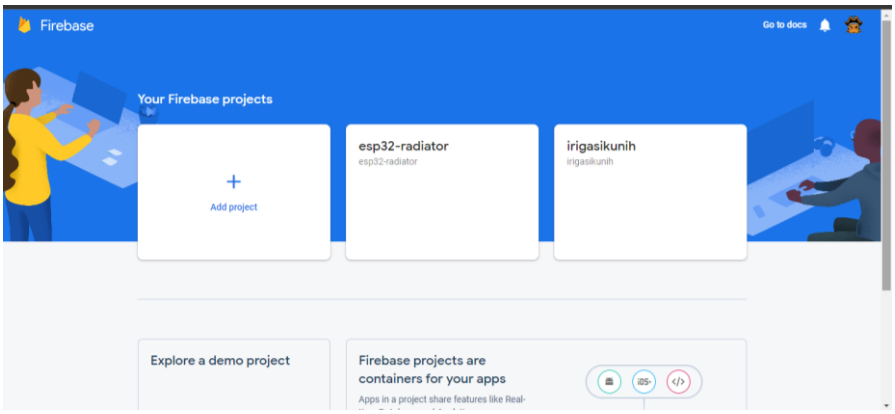
Tabel 3.3 Koneksi Pin Sensor *Water Level* Dengan NodeMCU ESP32

No	Pin *Water Level Sensor*	Fungsi
1	Data	Pembacaan data sensor *water level* di port GPIO 36 Nodemcu ESP32
2	VCC	Catu daya 5 V dari Nodemcu ESP32
3	GND	*Grounding*

3.3.4 Perancangan Perangkat Lunak

Dalam perancangan perangkat lunak dibagi menjadi 2 bagian yaitu perancangan google firebase dan MIT App Inventor.

a. Perancangan *Google Firebase*

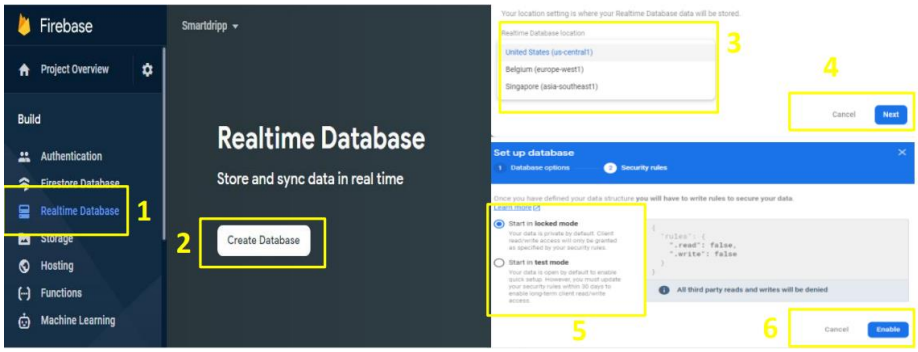


Gambar 3.7 Membuat Proyek Baru *Firebase*

Pada gambar 3.7 Merupakan tampilan awal untuk membuat proyek baru pada google *firebase*. Untuk membuat *project* baru klik pada bagian *add project*.

Gambar 3.8 Memberi Nama Proyek *Firebase*

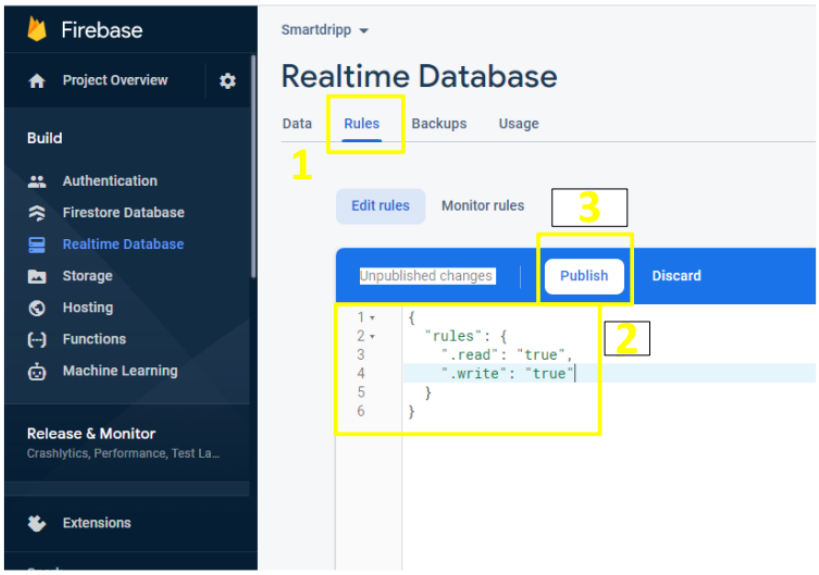
Selanjutnya pada gambar 3.8 ketikan nama proyek yang akan dibuat dan isi semua persetujuan yang ada. Pada pembuatan proyek *google firebase* ini nama yang digunakan adalah Radiator.



Gambar 3.9 Membuat *Realtime Database*

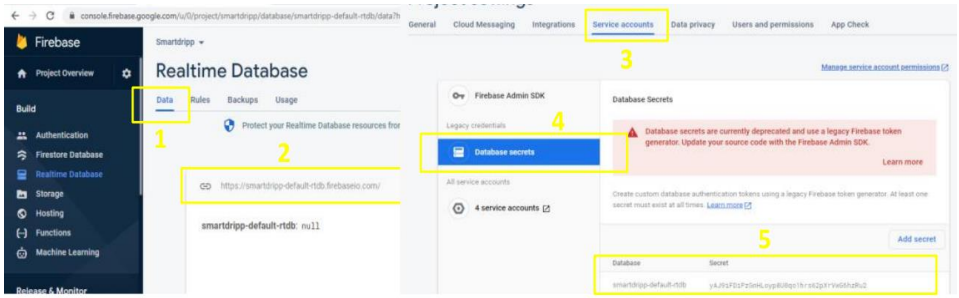
Pada gambar 3.9 merupakan langkah membuat *realtime database* di *google firebase*. Langkah pertama yang di tunjukan pada nomor satu memilih menu *realtime database* kemudian nomor dua *Create Database* untuk memulai membuat *realtime database* kemudian masuk ke langkah nomor tiga yaitu *database options* digunakan untuk mengatur lokasi penyimpanan dari *realtime database* yang digunakan pada kolom nomor empat klik *next* untuk melanjutkan ke langkah selanjutnya. Setelah memilih lokasi penyimpanan selanjutnya melakukan pengaturan keamanan yang di tunjukan pada nomor lima, di mana pada langkah ini menggunakan kondisi pada *locked* mode sehingga data yang nantinya tersimpan

dapat diatur apakah boleh diteruskan ke *platform* lain atau hanya dapat di cek di *google firebase*. Pada langkah ke enam klik *enable* untuk mengaktifkannya.



Gambar 3. 10 Mengatur *Rules Firebase*

Pada gambar 3.10 merupakan langkah untuk mengatur *rules* yang ada pada *firebase*. Langkah pertama yaitu klik pada bagian *rules*, kemudian pada bagian nomor dua di mana pengaturan *rules* yang digunakan pada kondisi *true*. Kondisi *true* berfungsi untuk membuat *database* pada *firebase* dapat ditampilkan ke platform untuk membuat aplikasi ataupun ditampilkan pada sebuah web. Kemudian pada bagian nomor tiga klik *publish* untuk menyimpan pengaturan tersebut.

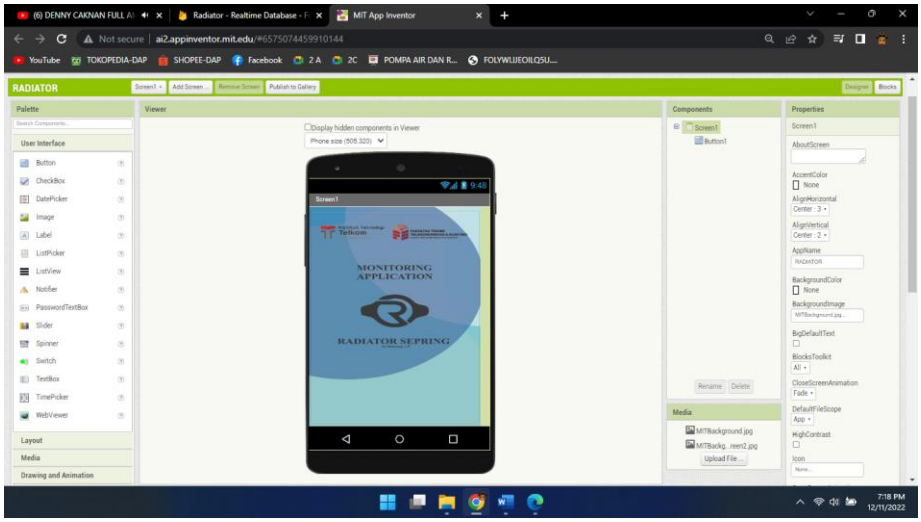


Gambar 3.11 Alamat dan *Token Firebase*

Pada gambar 3.11 merupakan langkah untuk melihat alamat dan *token* dari *firebase* yang dibuat. Di mana alamat digunakan sebagai tempat atau lokasi *database* ditampilkan yang di tunjukan pada nomor satu dan dua, sedangkan *token*

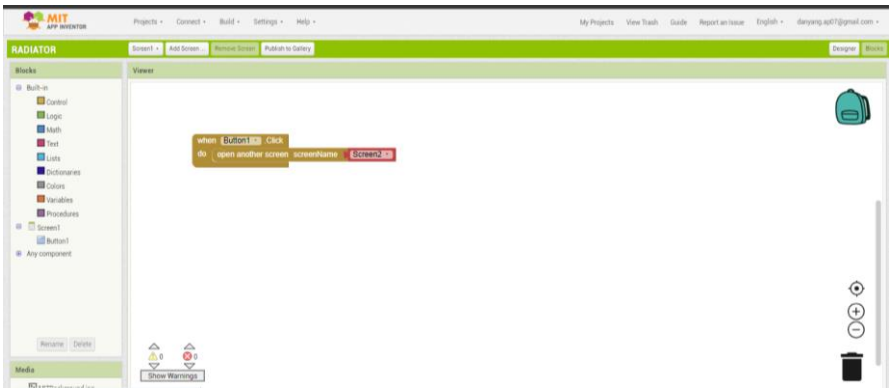
digunakan sebagai kode rahasia untuk dapat mengakses alamat *firebase* tersebut untuk kode yang digunakan di tunjukan nomor lima.

b. Perancangan MIT App Inventor



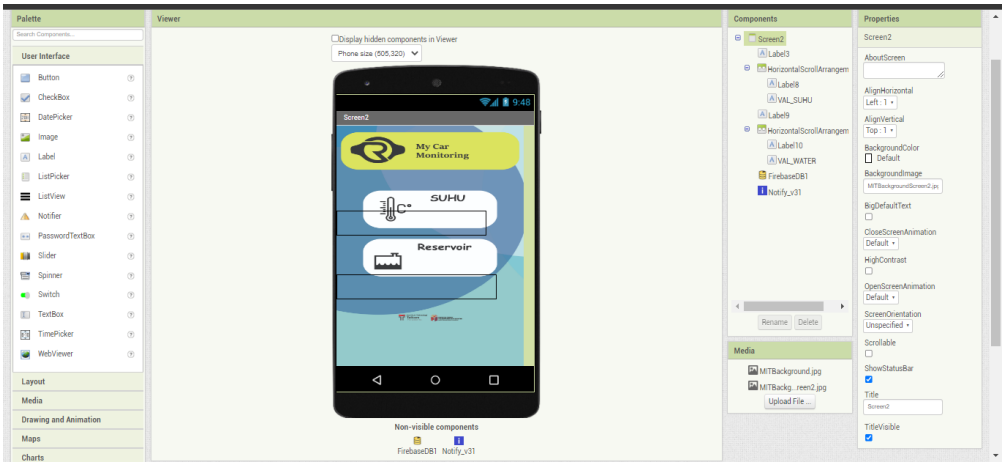
Gambar 3.12 Tampilan *Screen* 1

Pada gambar 3.12 merupakan tampilan *screen* 1 pada aplikasi android yang dibuat menggunakan platform mit app inventor. Gambar 3.12 diatas merupakan tampilan awal yang digunakan sebagai *flash screen* sebelum menuju *screen* berikutnya.



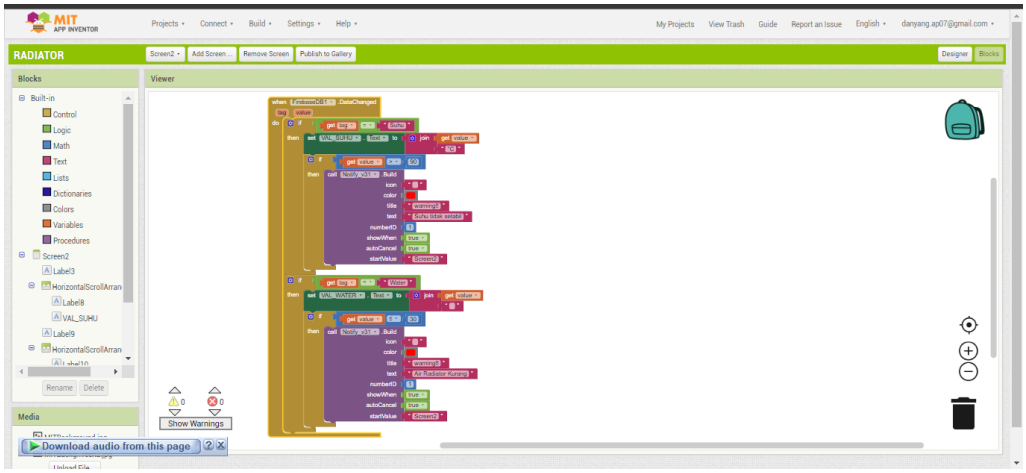
Gambar 3.13 *Block Editor Screen* 1

Gambar 3.13 merupakan tampilan *block editor* dari *screen* 1, dimana fungsi dari pembuatan *block* ini untuk membuat *screen* tap pada tampilan awal dengan mengatur *button1* yang digunakan untuk berpindah ke *screen* 2.



Gambar 3.14 Tampilan *Screen* 2

Pada gambar 3.14 merupakan tampilan *screen* 2 dari aplikasi yang dibuat menggunakan *platform* mit app inventor. *Screen* 2 berfungsi untuk memantau nilai-nilai yang didapat dari *database realtime firebase*.



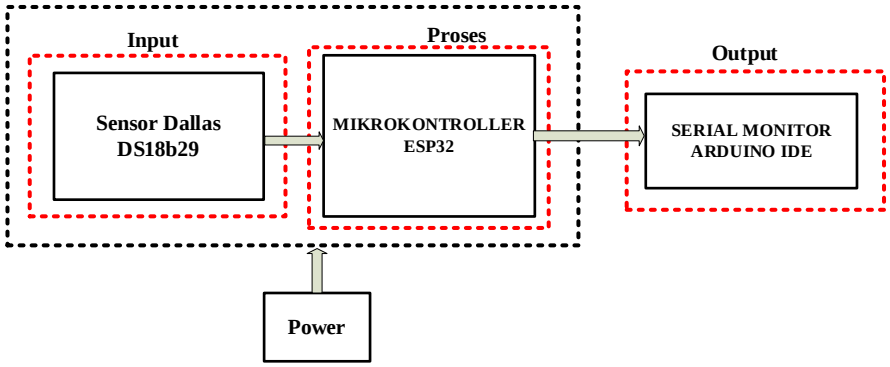
Gambar 3.15 *Block Editor Screen* 2

Pada gambar 3.15 merupakan tampilan *block* editor dari *screen* 2, dimana *block editor* ini digunakan untuk mendapatkan nilai dari *database*, memantau nilai dan memberikan notifikasi pada aplikasi yang dibuat. Jika nilai yang didapat dari *database* melebihi atau kurang dari nilai standar maka akan memunculkan sebuah notifikasi. Notifikasi yang muncul pada aplikasi berbeda-beda sesuai parameter yang digunakan.

3.4 SKENARIO PENGUJIAN

Skenario pengujian dilakukan untuk mengetahui apakah sistem yang dirancang diimplementasikan atau tidak. Sistem dapat dinyatakan beroperasi degan benar apabila semua komponen yang digunakan dapat beroperasi sesuai dengan tujuan. Beberapa proses pengujian yang akan dilakukan sebagai berikut:

3.4.1 Pengujian Sensor *Dallas* DS18B20

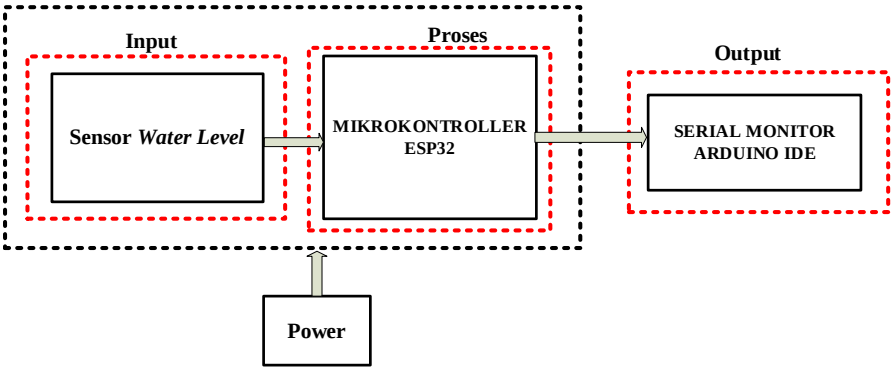


Gambar 3. 16 *Block* Diagram Pengujian Sensor *Dallas* DS18B20

Pada pengujian sensor *Dallas* DS18B20 dilakukan pengujian akurasi sensor untuk mendapatkan nilai *error* dengan membandingkan nilai pengukuran sensor *Dallas* DS18B20 yang terlihat pada *serial* monitor Arduino IDE dengan nilai aktual alat ukur termometer digital. Pengujian sensor dilakukan dengan cara merebus air pada wadah sampai suhu di atas 90°C. Besarnya nilai *error* yang diuji menggunakan rumus sebagai berikut:

$$\text{nilai } error = \left|\frac{nilai\ aktual - nilai\ pengukuran}{nilai\ aktual}\right| x100\% \qquad (3.1)$$

3.4.2 Pengujian Sensor *Water Level*



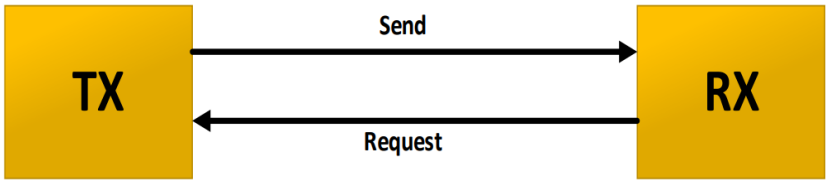
Gambar 3.17 *Block* Diagram Pengujian Sensor *Water Level*

Pada pengujian sensor *Water Level* dilakukan pengujian sensor dengan cara melakukan pengukuran dengan ketinggian air yang berbeda-beda. Pengujian dilakukan pada kondisi air posisi tidak menyentuh dan menyentuh sensor.

3.4.3 Pengujian Keseluruhan Sistem

Pada pengujian keseluruhan sistem dilakukan untuk menguji notifikasi aplikasi, pengiriman data ke *google firebase* dan pengujian prototipe sistem pendingin air radiator mobil. Pengujian pengiriman data ke *googl firebase* dengan menguji apakah data yang dikirimkan nodemcu ESP32 dapat diterima di *google firebase*, pengujian ini dilakukan dengan mengirim 2 data yang berbeda dengan tiap data dan dilakukan percobaan sebanyak 30 kali. Selanjutnya pengujian notifikasi aplikasi untuk menampilkan peringatan nilai suhu air dan tinggi air sebanyak 30 kali percobaan dengan mengatur nilai yang terbaca pada *google firebase*. Pengujian prototipe sistem pendingin air radiator dengan melakukan pemantauan suhu air yang dipanaskan hingga suhu mencapai >90℃.

3.4.4 Pengujian Quality Of Service (QoS)



Gambar 3.18 Skema Pengujian *Quality of Service* (QoS)

Pada gambar 3.18 merupakan model pengukuran yang digunakan untuk mendapatkan parameter *quality of service* (QoS). Pada TX terdapat jaringan *wifi* yang terkoneksi dengan mikrokontroler nodemcu ESP32 sebagai sisi pengirim atau transmitter. Sedangkan pada RX atau *receiver* sebagai sisi penerima sekaligus yang me*request* data dari TX merupakan *database* yang digunakan yaitu *google firebase*. Pengujian parameter dari *Quality of Service* (QOS) digunakan untuk mengukur nilai dari kualitas jaringan tertentu pada suatu layanan. Pada pengujian QoS ini ada 3 parameter yang akan diujikan pada penelitian ini yaitu *delay*, *throughput*, dan *packet loss* dengan menggunakan standar tiphon. Untuk mendapatkan data nilai dari QoS saat pengujian menggunakan aplikasi *wireshark* sebagai media untuk menampilkan paket data.

a. Pengujian *Delay*

Pengujian *delay* dilakukan untuk mengetahui semua total waktu tunda suatu paket. Pada pengujian *delay* menguji semua total selisih waktu data yang diterima di sisi Tx ke RX pada data terima selanjutnya. Setelah itu proses pengujian *delay* yang dilakukan sebanyak 7 kali percobaan dengan lama waktu pengambilan 1,5,10,15,20,25,30 menit pada jarak 3m dan penghalang berupa dinding tembok. Besarnya nilai *delay* diperoleh menggunakan persamaan :

$$Rata - rata\ delay = \frac{Total\ Delay}{Total\ Paket\ Diterima} \quad (3.2)$$

b. Pengujian *Throughput*

Pengujian *throughput* dilakukan untuk mengetahui laju pengiriman data. Pada pengujian *throughput* menguji laju pengiriman data dari sisi Tx ke RX. Setelah itu proses pengujian *throughput* yang dilakukan sebanyak 7 kali percobaan dengan lama waktu pengambilan 1,5,10,15,20,25,30 menit pada jarak 3m dan penghalang berupa dinding tembok. Besarnya nilai *throughput* diperoleh menggunakan persamaan :

$$Throughput = \frac{Paket\ Data\ Diterima}{Total\ Paket\ Diterima} \quad (3.3)$$

c. Pengujian *Packet loss*

Pengujian *packet loss* dilakukan untuk mengetahui jumlah paket yang diterima dan mengetahui jumlah paket yang hilang. Pada pengujian *packet loss* jumlah data yang hilang saat pengiriman data dari sisi Tx ke RX. Setelah itu proses pengujian *packet loss* dilakukan yang dilakukan sebanyak 7 kali percobaan dengan lama waktu pengambilan 1,5,10,15,20,25,30 menit pada jarak 3m dan penghalang berupa dinding tembok. Besarnya nilai *packet loss* diperoleh menggunakan persamaan :

$$Packet\ Loss = \frac{Paket\ Data\ Dikirim - Paket\ Data\ Diterima}{Paket\ Data\ Dikirim} x100\% \quad (3.4)$$