BAB 2
DASAR TEORI

2.1 TINJAUAN PUSAKA

Penelitian oleh Randis dan Sarmono pada tahun 2018 dengan judul "Aplikasi *Internet of Things* Monitoring Suhu Engine Untuk Mencegah Terjadinya *Over Heat*" membahas tentang sistem online monitoring temperatur *engine* berbasis *Internet of Things*. Penelitian ini dikembangkan untuk memantau *temperatur engine* baik melalui *website thingspeak* dan aplikasi android *virtuino*. Sistem terdiri dari beberapa komponen, komponen utama yaitu arduino uno yang akan memperoleh sinyal *input* dari sensor temperatur DS18B20, setelah itu data disimpan di *database* menggunakan jaringan internet melalui *wifi*. Peningkatan temperatur ketika *engine* beroperasi direkam oleh sensor dan akan dikirim ke mikrokontroller yang dapat diakses melalui *website* ataupun perangkat *mobile*. Hasil percobaan menunjukkan bahwa selisih nilai pembacaan sensor dengan alat ukur yang terkalibrasi cukup kecil yaitu hanya sebesar 0,14°C, sedangkan *margin error* diperoleh 0,14 %. Pengujian alat dilakukan dengan menjalankan *engine* dengan rentang waktu 1 jam, dan diperoleh *temperatur* tertinggi *engine* sebesar 69°C dan dapat diakses dengan *website* dan aplikasi pada perangkat mobile [6].

Penelitian oleh Muhammad Zaim Arief, Fiqqih, Suhanto pada tahun 2020 dengan judul "Prototipe Pengisian Air Radiator Pada Genset Secara Otomatis Berbasis Arduino Uno Via Android" membahas tentang radiator pada mesin genset yang berfungsi sebagai sistem pendingin. Dikarenakan Air pada tangki radiator genset sering mengalami keterlambatan pengisian dan pengisian masih dilakukan secara manual sehingga peneliti menggunakan sensor *Water Level* yang berbasis *Internet of Thing* (IoT) untuk mengukur ketinggian air dalam tangki radiator genset, dan menggunakan tampilan dari *Interface* Android sehingga hasil yang diperoleh dari sensor dapat di tampilkan pada tampilan monitoring aplikasi yang berupa *level* ketinggian dan jumlah volume air. Alat ini dapat dikontrol dan dimonitoring menggunakan *interface* Android. Mikrokontroler Arduino UNO digunakan untuk menerima data sensor ketinggian air, kemudian data ditampilkan pada LCD dan di kirimkan pada *interface* android melalui Wemos D1 *Mini*. Dari hasil pengujian

sistem didapatkan bahwa pompa menyala secara otomatis ketika air pada tangki ≤ 10 cm dan ketika air pada tangki ≥20 cm maka pompa akan mati secara otomatis [7].

Penelitian oleh Mohammad Hasan Fuadi, Ahmad Wahyu Purwandi dan Ridho Hendra Y.P pada tahun 2020 dengan judul "Rancang Bangun Kontrol Dan Monitoring Suhu Pada Mesin Diesel Menggunaan Web Mobile" membahas tentang memonitoring serta mengontrol suhu air pendingin supaya stabil dalam mesin diesel. Pengoperasian kontrol suhu menggunakan sistem telecontrol yang terhubung dengan jaringan internet (*Internet of Things*) sehingga pengendalian suhu diesel dapat dilakukan dari jarak jauh. Monitoring suhu dan *water level* pada tangki cadangan menggunakan *Web Mobile*. Komponen yang digunakan pada penelitian ini adalah sensor Suhu yang digunakan untuk mengukur suhu air pendingin dalam mesin diesel sehingga nantinya dapat dimonitoring suhu mesin diesel tersebut pada Web Mobile. Hasil pengujian yang didapat, Sensor Suhu memiliki kesalahan pembacaan suhu rata - rata sebesar 0,031004%. mesin diesel dengan sistem pendinginan solenoid valve terkontrol dapat menghasilkan suhu ideal dibandingkan saat solenoid *valve* terbuka (menggunakan radiator secara terus menerus) ataupun saat solenoid *valve* tertutup (tanpa menggunakan radiator), Saat Solenoid terkontrol suhu mesin dapat ideal karena sistem buka tutup solenoid valve suhu paling rendah 56,34 °C dan suhu tertinggi hanya 80,85 °C [8].

Penelitian oleh Jaim, Iskendar MS dan Sorimuda pada tahun 2018 dengan judul "Analisis variasi Jumlah Sudu Dan Cairan Pendingin (*Coolant*) Pada Kinerja Mesin Terhadap Efisiensi Bahan Bakar Dan Pencegahan Terjadinya *Overheating*" membahas tentang pengaruh dari variasi jumlah sudu dan variasi cairan pendingin terhadap efisiensi bahan bakar dan pencegahannya terjadinya *overheating*. Metode yang digunakan dalam penelitian ini adalah eksperimen, data yang di peroleh merupakan proses pengujian pada *water pump* dengan sudu *impeller* yang bervariasi juga pemakaian *water coolant* yang berbeda dan proses dilakukan secara bergantian dari setiap kali pengujian sampai mesin tersebut benar – benar mesin itu dingin. Hasil dari penelitian ini adalah Temperatur mesin yang bekerja secara optimal pada suhu yang cukup tinggi sekitar 75 °C – 90 °C. Jika mesin bekerja pada

suhu yang rendah akan membuat komponen mesin cepat mengalami kerusakan, detonasi dan membuat polusi dan boros bahan bakar [9].

Penelitian oleh Fawwaz Ramzy Darmawan, Yuri Ariyanto dan Sofyan Noor Arief pada tahun 2020 dengan judul "Pengukuran Ketinggian Air dalam Tanki Berbasis IoT menggunakan Protokol *Message Queuing Telemetry Transport* (MQTT)" membahas tentang penggunaan sensor *water level* pada tandon atau tangki air yang digunakan untuk memantau ketinggian air agar dapat meminimalisir penggunaan energi listrik. Metode yang digunakan pada penelitian ini menerapkan metode *fuzzy logic*. Alat yang digunakan adalah ESP32 sebagai mikrokontroler agar dapat terhubung ke internet dan dikirimkan ke telegram sedangkan untuk *outputanya* adalah sensor *water level* untuk mendeteksi ketinggian pada tandon atau tangka dan sensor *turbidity* untuk membaca kekeruhan air. Hasil yang didapatkan pada penelitian ini bahwa sistem dapat melakukan monitoring ketinggian air dan kekeruhan air sesuai dengan data yang telah dikirim dari ESP32 secara r*ealtime* [10].

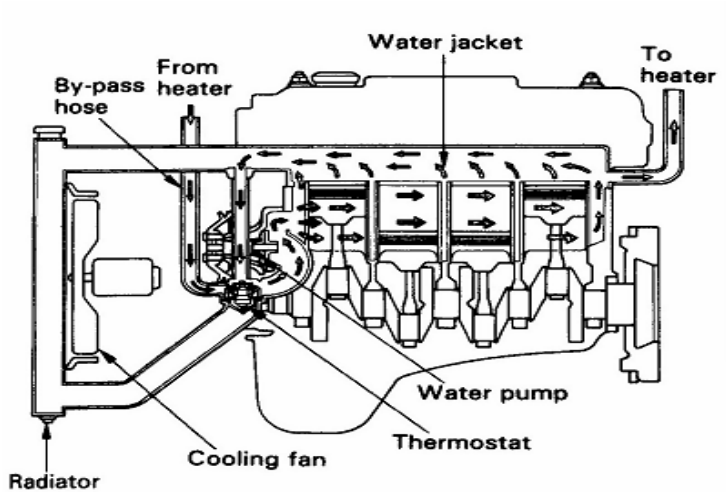
2.2 DASAR TEORI

2.2.1 Sistem Pendingin Air Radiator (Radiator *Water Cooling System*)

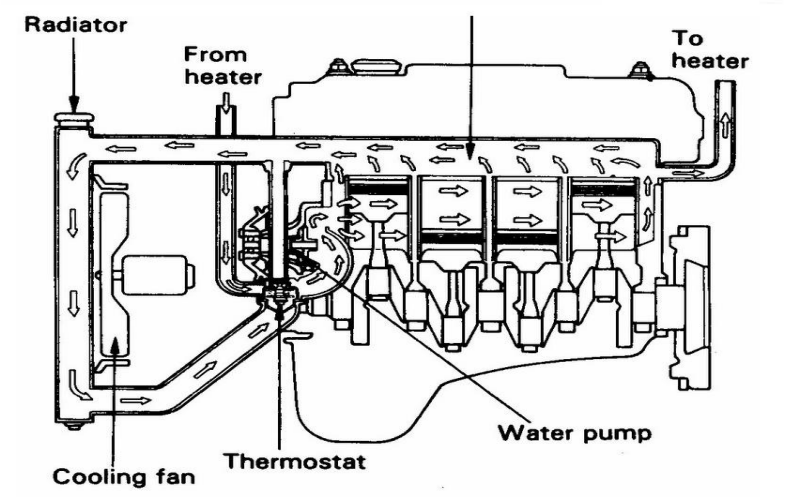
Salah satu faktor untuk menjaga umur mesin adalah terjadinya kondisi sistem pendingin (*Cooling System*). Pada kendaraan Panas dari hasil pembakaran gas dalam ruang bakar sebagian diserap oleh air pendingin pada *water jacket* yang mengelilingi dinding silinder mesin. Panas yang diserap oleh air pada radiator pendingin akan mengakibatkan naiknya temperatur air pada radiator. Apabila air pendingin tersebut tetap berada pada *water jacket* maka air cenderung akan menguap sehingga akan menyebabkan *overheat*.

Prinsip kerja pada sistem pendingin air radiator ini yaitu air akan dipompa oleh pompa air untuk bersirkulasi dari *water pump* ke *water jacket* melewati *by-pass hose* kemudian kembali lagi ke *water pump*. Ketika mesin dalam kondisi dingin dan air masih dingin maka sirkulasi air tidak berjalan karena katup *thermostat* tertutup dan *by-pass valve* terbuka, hal ini bertujuan agar air pendingin dan mesin cepat mencapai suhu kerja maksimal yaitu 80-90 °C. Apabila temperatur air pendingin sudah mencapai suhu tinggi, maka *thermostat* membuka dan *by-pass*

valve menutup, sehingga aliran air pendingin mengalir dari radiator ke *lower hose*, ke *water pump*, ke *water jacket*, ke *upper hose* dan kembali ke radiator untuk didinginkan dengan kipas dan udara yang dihasilkan dari gerakan maju kendaraan [11].



Gambar 2.1 Sirkulasi air Ketika mesin air dingin [11]**.**



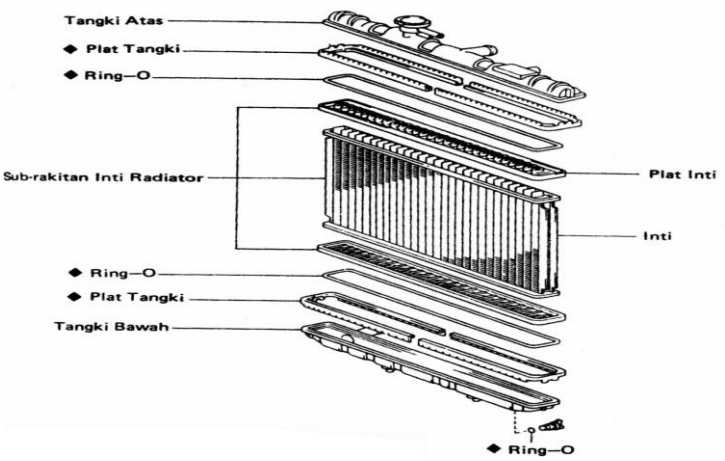
Gambar 2.2 Sirkulasi air Ketika mesin air panas [11]**.**

2.2.2 Radiator Mobil

Radiator adalah alat penukar panas yang digunakan untuk memindahkan energi panas dari satu medium ke medium lainnya yang tujuannya untuk mendinginkan maupun memanaskan. Radiator yang pada umumnya digunakan pada kendaraan bermotor (roda dua atau roda empat), namun tidak jarang radiator juga digunakan pada mesin yang memerlukan pendinginan ekstra. Pada kendaraan

baik motor atau mobil radiator pada umumnya terletak di depan dan berada didekat mesin atau pada posisi tertentu yang menguntungkan bagi sistem pendinginan. Hal ini bertujuan agar mesin mendapatkan pendinginan yang maksimal sesuai yang dibutuhkan mesin.

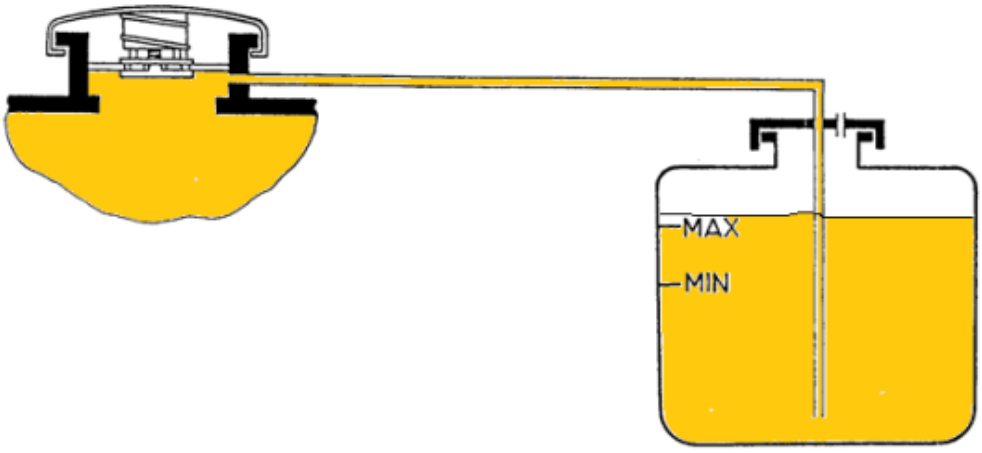
Radiator mempunyai kisi-kisi yang terdiri dari beberapa saluran air berbentuk sirip-sirip pendingin. Sirip-sirip pendingin akan memperluas jalur air yang akan didinginkan oleh kipas pendingin yang berada dibelakang radiator. Kebanyakan sirip pendingin yang digunakan berbentuk plat dan zig zag agar dapat melepas panas lebih baik. Konstruksi radiator terdiri dari Tangki atas, Inti Radiator, Tangki bawah. Tangki atas berfungsi menampung air yang telah panas dari mesin melalui *upper hose*, dan pada *upper tank* terdapat pipa pembuangan untuk mengalirkan kelebihan air menuju *reservoir* tank karena ekspansi panas. Inti Radiator berfungsi untuk membuang panas dari air ke udara agar temperature air menjadi lebih rendah dari sebelumnya. Inti radiator memiliki beberapa lubang kecil atau kisi-kisi yang berperan untuk penyerapan air yang mengalir dari atas ke bawah, mengikut perputaran sistem pendingin pada bagian mesin. Setiap saat air mengalir ke kisi-kisi tersebut akan membuang panas dari air pendingin dengan pertolongan angin saat mobil bergerak. Tangki bawah berfungsi untuk menampung air yang telah didinginkan oleh radiator yang selanjutnya akan disalurkan ke mesin[12].



Gambar 2.3 Radiator Mobil [11]**.**

2.2.3 Reservoir Radiator

Tabung *Reservoir* radiator pada penelitian ini digunakan sebagai wadah air cadangan pada radiator ketika air pada radiator mengalami kekurangan atau kelebihan air. Tangki cadangan atau *reservoir* dihubungkan ke radiator melalui selang *overflow*. Komponen ini memiliki fungsi untuk menjaga agar volume air pendingin yang ada pada radiator tetap dalam keadaan stabil. *Reservoir* tank akan menampung luapan air panas dari radiator untuk sementara waktu. Jika suhu air yang ada di dalam radiator sudah lebih dingin maka cairan yang sudah ditampung akan kembali terhisap ke *reservoir*. Sehingga air dalam radiator mobil kembali terisi lagi[12].

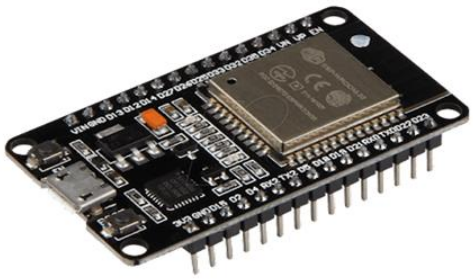


Gambar 2.4 *Reservoir* Radiator[12]

2.2.4 NodeMCU ESP32

Pada penelitian ini, NodeMCU ESP32 digunakan sebagai pengirim data hasil baca sensor dari Arduino ke google *firebase* dengan menggunakan modul *wifi* ESP32 yang terpasang. ESP32 merupakan mikrokontroller berdaya rendah pada seri *chip* (SoC) dengan *Wi*-Fi & kemampuan *Bluetooth* dua mode. Keluarga ESP32 termasuk *chip* ESP32-D0WDQ6 (dan ESP32-D0WD), ESP32-D2WD, ESP32-S0WD, dan sistem dalam paket (SiP) ESP32-PICO-D4. Pada intinya, ada mikroprosesor *Tensilica Xtensa* LX6 *dual-core* atau *single-core* dengan *clock rate* hingga 240 MHz. ESP32 sudah terintergrasi dengan *built-in antenna switches*, RF

balun, *power amplifier, low-noise receive amplifier, filters, and power management modules.* Didesain untuk perangkat seluler, perangkat elektronik yang dapat dipakai, dan aplikasi IoT, ESP32 juga bekerja dengan menggunakan daya sangat rendah melalui fitur hemat daya termasuk *fine resolution clock gatting, multiple power modes, and dynamic power scaling* [13].



Gambar 2.5 NodeMCU ESP32[13]**.**

2.2.5 Sensor Dallas DS18B20

Pada penelitian ini sensor *Dallas* DS18B20 digunakan untuk memonitoring suhu pada radiator mobil yang nantinya hasil akan dikirimkan ke *database google firebase*. Sensor *Dallas* DS18B20 adalah sensor suhu digital yang dikeluarkan oleh *Dallas Semiconductor* dan merupakan seri terbaru dari Maxim IC. Sensor suhu DS18B20 beroperasi dalam kisaran -55 ℃ sampai 125 ℃. Meskipun sensor ini dapat membaca hingga 125 ℃. Sensor ini juga dapat membaca dengan ketelitian 9-12 bit. DS18B20 memungkinkan penggunaan sensor dalam jumlah besar hanya melalui satu kabel (*single wire*) atau 1-wire *protocol*. Sensor DS18B20 memiliki beberapa fitur utama seperti antarmuka hanya menggunakan satu kabel sebagai komunikasi dan tidak memerlukan komponen tambahan[14].

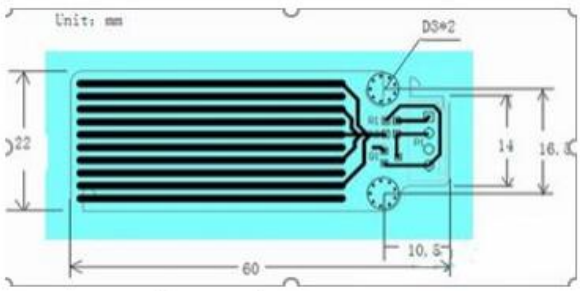
Gambar 2.6 Sensor *Dallas* DS18B20[14].

Tabel 2.1 Tabel Spesifikasi Sensor Dallas DS18B20

Nama	**Keterangan**
DC *Supply Voltage*	3-5.5 V
Tingkat Keakuratan	± 0.5
Batas Temperatur	-55°C S/d +125°C
Output	Digital 1-wire
Resolusi ADC	12bit

2.2.6 Sensor Water Level

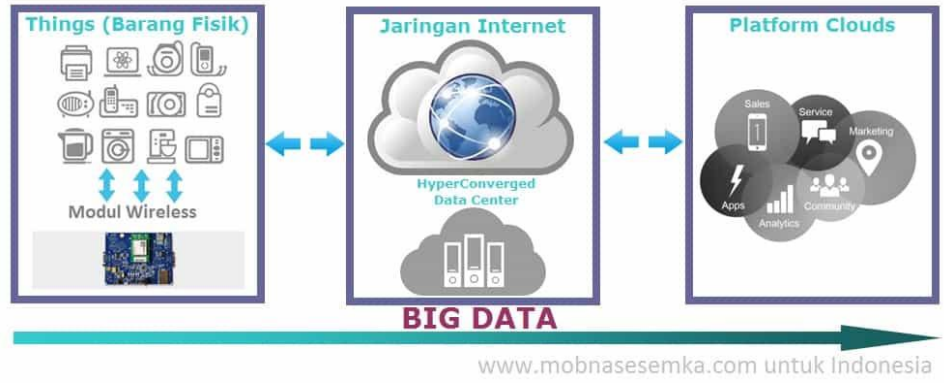
Pada penelitian ini sensor *Water Level* digunakan pada sistem sebagai pengukur tingkat level *minimun* air pada tanki cadangan radiator. Nilai ketinggian tersebut kemudian dimonitor melalui aplikasi pada android. *Water sensor* merupakan alat yang digunakan untuk mendeteksi air. *Water sensor* membaca *resistasi* yang dihasilkan oleh air yang mengenai *probe* pada sensor tersebut, semakin banyak air yang mengenai *probe* tersebut maka hambatannya semakin kecil dan ketika tidak ada air yang mengenai *probe* tersebut maka hambatanya semakin besar *Water sensor* memiliki tegangan kerja pada 5volt DC dan arus sebesar 20mA. Nilai ADC didapatkan dengan cara mengkalikan nilai ADC dengan nilai tegangan yang didapatkan dari resistansi *probe* kemudian dibagi dengan nilai tegangan sumber sehingga akan didapatkan nilai ADC dan keluaran dari *water sensor* merupakan data analog[15].



Gambar 2.7 *Sensor Water level* [16].

2.2.7 Internet of Things (IoT)

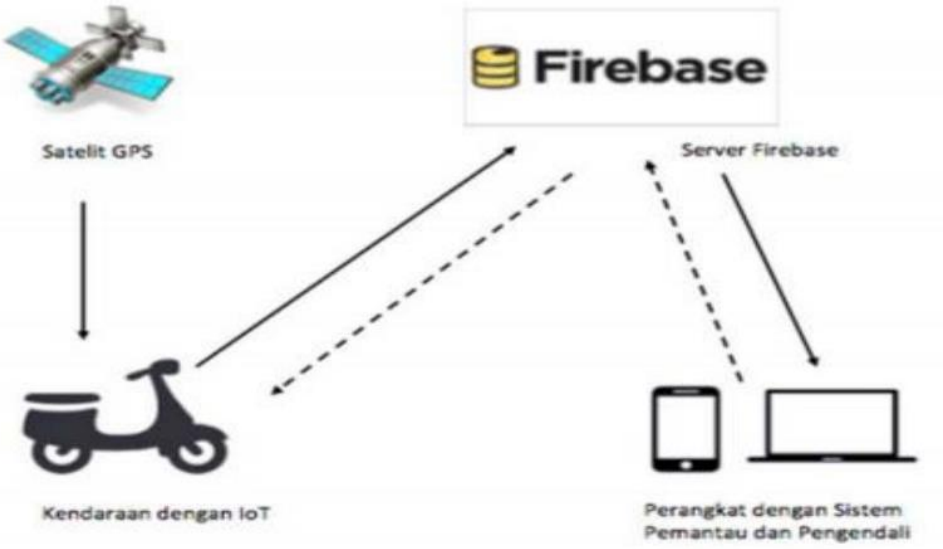
Internet of things (IoT) sebuah gagasan dimana benda dapat berkomunikasi satu dengan yang lain sebagai bagian dari satu kesatuan sistem terpadu menggunakan jaringan internet sebagai penghubung misalnya CCTV yang terpasang di sepanjang jalan dihubungkan dengan koneksi internet dan disatukan di ruang kontrol yang jaraknya mungkin puluhan kilometer. atau sebuah rumah cerdas yang dapat di manage lewat *smartphone* dengan bantuan koneksi internet. pada dasarnya perangkat IoT terdiri dari sensor sebagai media pengumpul data, sambungan internet sebagai media komunikasi dan server sebagai pengumpul informasi yang diterima sensor dan untuk analisa. Cara kerja *Internet of Things* yaitu dengan memanfaatkan sebuah argumentasi pemrograman yang dimana tiap-tiap perintah argumennya itu menghasilkan sebuah interaksi antara sesama mesin yang terhubung secara otomatis tanpa campur tangan manusia dan dalam jarak berapa pun. Internet inilah yang menjadi penghubung di antara kedua interaksi mesin tersebut, sementara manusia hanya bertugas sebagai pengatur dan pengawas bekerjanya alat tersebut secara langsung[17].



Gambar 2.8 Konsep IoT [18].

2.2.8 Firebase

Firebase merupakan salah satu *platform* untuk aplikasi *realtime* yang ketika data berubah, maka aplikasi dengan *firebase* akan meng-*update* secara langsung 20 melalui setiap *device* (perangkat) baik web atau mobile. *Firebase* sendiri mempunyai *library* (pustaka) yang lengkap untuk sebagian besar *platform* web dan *mobile* dan dapat digabungkan dengan berbagai *framework* seperti *node, java, Java Script, AngularJS*, dan lain-lain. *Application Programming Interface* (API) untuk menyimpan dan sinkronisasi data akan disimpan sebagai *bit-bit* dalam bentuk JSON pada *cloud* dan akan disinkronisasi secara *realtime* pada *firebase*. Layanan pada *firebase* yaitu meliputi autentikasi pengguna, pengaturan keamanan, dan *hosting*. Perubahan data yang terjadi pada satu *client* akan disinkronisasi pada semua *client* yang sudah terdaftar ke data tersebut. Kelebihan dari *firebase* yaitu dapat menerima banyaknya data dari 1 juta perangkat secara bersamaan. *Firebase* dapat digunakan juga untuk menyimpan data yang diterima. Kemudian *firebase* dapat digunakan untuk menyediakan layanan penyimpanan *cloud* yang memungkinkan untuk *middleware* dapat digabungkan dengan *interface* aplikasi dari pihak ketiga. *Firebase* dapat memungkinkan pengembang untuk menciptakan aplikasi *mobile* dan *web* yang digunakan untuk men-*generated* data secara *realtime* [19].



Gambar 2.9 Arsitektur *Firebase*.

2.2.9 App Inventor

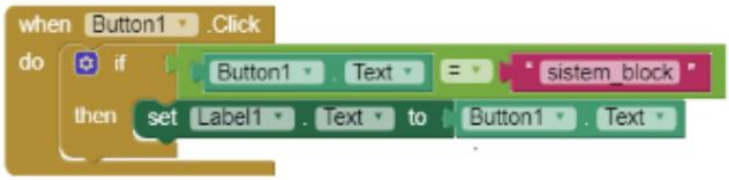
App Inventor merupakan *tool* untuk membuat aplikasi android, yang berbasis visual *block* programming. Pada app inventor dapat membuat aplikasi tanpa kode satupun App inventor juga sering disebut visual *block programming* karena penggunaannya yang dilakukan dengan menyusun dan men-*drop* & *drops* blok yang merupakan simbol-simbol perintah dan fungsi *even handler* tertentu dalam membuat aplikasi. App Inventor adalah aplikasi web sumber terbuka yang awalnya dikembangkan oleh Google, dan saat ini sudah dikelola oleh *Massachusetts Institute of Technology* (MIT). App Inventor memungkinkan pengguna baru dapat memprogram komputer untuk menciptakan aplikasi bagi sistem operasi Android. App Inventor menggunakan antarmuka grafis, serupa dengan antarmuka pengguna pada *Scratch* dan *Star* Logo TNG, yang memungkinkan pengguna dapat men-*drop-and-drop* objek visual untuk menciptakan aplikasi yang bisa dijalankan pada sistem operasi Android. Dalam menciptakan App Inventor sendiri, Google telah melakukan riset yang berhubungan dengan komputasi edukasional dan menyelesaikan lingkungan pengembangan online Google. Pada App Inventor ini terdapat beberapa komponen yang terdiri dari:

A. Komponen desainer yang berjalan pada browser berfungsi untuk memilih komponen yang diperlukan untuk mengatur propertinya. Pada komponen

desainer sendiri terdapat beberapa bagian, yaitu: *palette*, *viewer*, *component*, media dan *properties*.

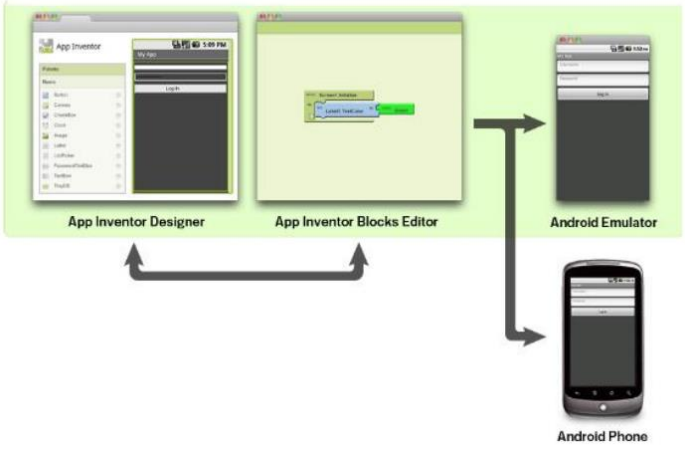
B. Block Editor berjalan di luar browser dan digunakan untuk membuat dan mengatur *behaviour* dari komponen-komponen yang akan nantinya dipilih dari komponen desainer.

C. Emulator yang digunakan untuk menjalankan dan menguji *project* yang telah dibuat.



Gambar 2.10 *Block* Editor MIT App Inventor

Block Editor merupakan beberapa kumpulan blok berisi perintah untuk fungsi percabangan, perulangan, *variable*, *array*, serta beberapa kelas yang digunakan seperti *Public Static Class*, sehingga *user* bisa langsung memakai metode tersebut tanpa perlu membuat objek terlebih dahulu [20].



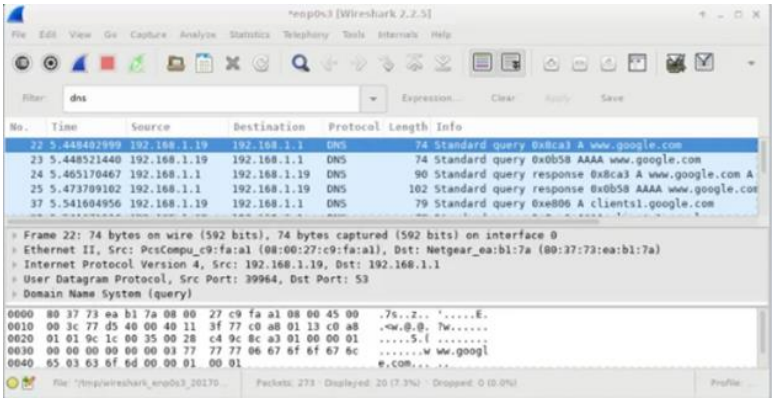
Gambar 2.11 *Flow* Proses MIT App Inventor

2.2.10 Wireshark

Wireshark merupakan sebuah aplikasi *capture* paket data berbasis *open-source* yang berguna untuk memindai dan menangkap trafik data pada jaringan internet. Aplikasi ini umum digunakan sebagai alat *troubleshoot* pada jaringan yang

bermasalah. Selain itu, juga biasa digunakan untuk pengujian *software* karena kemampuannya untuk membaca konten dari tiap paket trafik data.

Wireshark memiliki fungsi untuk pekerjaaan analisis jaringan. Cara kerjanya yaitu dengan ‘menangkap’ paket-paket data dari protokol-protokol yang berbeda dari berbagai tipe jaringan yang umum ditemukan di dalam trafik jaringan internet. Paket-paket data tersebut ‘ditangkap’ lalu ditampilkan dijendela hasil *capture* secara *real-time* [21].



Gambar 2.12 Tampilan *Software Wireshark* [21].

2.2.11 Quality of Service (QoS)

Quality of Service (QoS) atau kualitas layanan merupakan metode pengukuran yang digunakan untuk menentukan kemampuan sebuah jaringan seperti; aplikasi jaringan, *host* atau *router* dengan tujuan memberikan *network service* yang lebih baik dan terencana sehingga dapat memenuhi kebutuhan suatu layanan. Informasi yang dikumpulkan oleh *Perception* layer harus dapat dikirim dan ditampilkan dengan akurat. Jika terjadi gangguan saat pengiriman data maka data tidak dapat ditampilkan dengan sempurna. karena komunikasi dan kemampuan pengambilan keputusan yang real-time dari sensor, memprediksi *Quality of Service* (QoS) perangkat IoT sangatlah penting untuk mendeteksi kinerja maksimal dan jadwal waktu mati dan hidup perangka [22].

2.2.12 Akurasi

Akurasi merupakan keakuratan atau ketepatan yang mengacu pada seberapa dekat nilai pengukuran sensor dengan nilai sebenarnya dari variabel yang diukur. Akurasi menggambarkan seberapa akurat pengukuran sensor dalam menghasilkan nilai yang mendekati nilai sebenarnya. Akurasi pengukuran sensor dapat dinyatakan dalam satuan persen atau dalam satuan yang sama dengan variabel yang diukur. Akurasi juga dapat dievaluasi dengan membandingkan nilai pengukuran sensor dengan nilai sebenarnya yang diketahui, atau dengan membandingkan hasil pengukuran dengan nilai pengukuran dari sensor yang diketahui memiliki akurasi yang tinggi [23]. Nilai akurasi dapat diketahui dengan rumus berkut :

$$Eror = \frac{Nilai\ aktual - Nilai\ pengukuran}{Nilai\ aktual}\ x\ 100\% \quad (2.1)$$

$$Akurasi = 100\ \% - Nilai\ Eror\% \quad (2.2)$$

2.2.13 Analog To Digital Converter (ADC)

ADC (*Analog To Digital Converter*) merupakan perangkat elektronika yang berfungsi mengubah sinyal analog menjadi sinyal digital. ADC memiliki 2 karakter prinsip yaitu kecepatan sampling, dan resolusi. ADC bertindak sebagai penghubung besaran fisis yang umumnya bersifat analog seperti suhu, kelembaban, tekanan, dll. Resolusi ADC menentukan akurasi atau ketelitian nilai hasil konversi ADC". Sebagai contoh ADC 10 bit akan memiliki *output* 10 bit data digital. Ini berarti sinyal *input* dapat dinyatakan dalam 1024 atau (- 1) nilai diskrit. ADC 10 bit akan memiliki *output* 10 bit data digital. Ini berarti sinyal *input* dapat dinyatakan dalam 1024 data diskrit seperti yang digunakan pada nodemcu. Dari contoh di atas dapat disimpulkan bahwa ADC 10 bit akan memberikan resolusi yang lebih baik dari pada ADC 8 bit. Prinsip kerja ADC adalah menkonversi sinyal analog ke dalam bentuk besaran yang merupakan rasio perbandingan sinyal *input* dan tegangan referensi. Sebagai contoh, bila tegangan referensi 5 volt, tegangan *input* 3 volt, rasio *input* terhadap referensi adalah 60%. Jadi, jika menggunakan ADC 10 bit seperti pada mikrokontroller nodemcu dengan skala maksimum 1024, akan didapatkan sinyal digital sebesar 60% x 1024 = 614 (bentuk desimal) atau 110000101000

(bentuk 14 biner). Rasio perbandingan sinyal *input* dan tegangan referensi dapat dirumuskan sebagai berikut :

$$Signal = \frac{Sample}{Max\_Value}\ x\ Refrence\_vlotage \quad (2.3)$$

Sedangkan dengan asumsi bahwa nilai digital keluaran ADC merupakan nilai diskrit yang dibulatkan ke bawah, resolusi ADC tersebut dirumuskan sebagai berikut :

$$Step\ size = \frac{V\ Reff}{2^n} \quad (2.4)$$

Resolusi ADC atau yang disebut sebagai step *size* adalah perubahan terkecil pada masukan ADC yang masih bisa dideteksi, contoh ADC 10bit dengan VREF = 5V dan VIN = 2,55V maka :

$$\frac{Vin\ 2^n}{V\ Ref} = \frac{2{,}55\ V\ x\ 2^{10}}{5V} = 522{,}24 \quad (2.5)$$

Sehingga nilai digital keluaran ADC yang digunakan adalah 522,24. ADC tipe pendekatan berturut-turut merupakan ADC yang biasa digunakan di mikrokontroler. Untuk mempermudah pemahaman hubungan antara masukan dan keluaran ADC, sebagai contoh dengan melihat rumus 2.1 untuk menghitung resolusi ADC tersebut maka dengan ADC 10bit dan tegangan referensi 5V jika ADC yang terbaca adalah 254 akan mendapatkan resolusi tegangan sebesar 1,24V [13].

2.3 STANDAR THIPHON

Standar Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) adalah suatu inisiatif dari European Telecommunications Standards Institute (ETSI) Nomer TR 101 878 V5.1.1 untuk mengembangkan standar teknologi dan arsitektur jaringan yang memungkinkan integrasi antara teknologi telekomunikasi dan internet. TIPHON fokus pada standarisasi protokol jaringan, format data, dan antarmuka aplikasi yang berfungsi sebagai "jembatan"

antara jaringan telekomunikasi dan internet. Dengan demikian, TIPHON membantu memastikan *interoperabilitas* dan *interkoneksi* yang lebih baik antara jaringan telekomunikasi dan internet, sehingga memungkinkan pengembangan aplikasi dan layanan baru yang lebih inovatif dan efektif [22]. Pada penelitian ini yang digunakan dari standar TIPHON merupakan :

2.3.1 Throughput

Throughput merupakan jumlah total kedatangan paket yang diakses yang diamati pada tujuan selama interval waktu tertentu dibagi oleh durasi interval waktu tersebut [22].

Tabel 2.2 Kategori *Throughput* Menurut TIPHON

Kategori *Throughput*	**Indeks**	***Throughput***
Sangat Bagus	76-100 *bps*	4
Bagus	51-75 *bps*	3
Sedang	26-50 *bps*	2
Buruk	25 *bps*	1

2.3.2 Delay

Delay merupakan waktu yang dibutuhkan data untuk menempuh jarak dari titik asal ke titik tujuan [22].

Tabel 2.3 Kategori *Delay* Menurut TIPHON

Kategori *Latency*	***Delay***	**Indeks**
Sangat Bagus	< 150 m/s	4
Bagus	150 s/d 300 m/s	3
Sedang	300 s/d 450 m/s	2
Buruk	> 450 m/s	1

2.3.3 Packet Loss

Packet loss merupakan suatu parameter yang memberikan suatu kondisi yang menunjukan jumlah *packet* yang hilang [20].

Tabel 2.4 Kategori *Packet Loss* Menurut TIPHON

Kategori *Packet loss*	**Indeks**	***Throughput***
Sangat Bagus	0%	4
Bagus	3%	3
Sedang	15%	2
Buruk	25%	1