BAB I

PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi dan moderisasi peralatan elektronik telah menyebabkan terjadinya perubahan yang mendasar dalam kehidupan manusia. Dimana manusia membutuhkan segala sesuatunya serba otomatis, praktis, dan fleksibel. Salah satu kebutuhan manusia adalah lingkungan hidup yang nyaman. Lingkungan hidup yang nyaman tercipta apabila manusia menjaga kebersihan, sehingga sangat penting menumbuhkan kesadaran menjaga lingkungan dengan membuang sampah pada tempatnya (Ubaildillah, 2015).

Untuk mencapai kondisi masyarakat yang hidup sehat dan sejahtera di masa yang akan datang, akan sangat diperlukan adanya lingkungan permukiman yang sehat. Dari aspek persampahan, maka kata sehat akan berarti sebagai kondisi yang akan dapat dicapai bila sampah dapat dikelola secara baik sehingga bersih dari lingkungan permukiman dimana manusia beraktifitas di dalamnya (Permen PU nomor: 21/PRT/M/2006).

Berbagai upaya dilakukan oleh pemerintah dalam mengelola sampah agar tidak menjadi bencana yang merugikan masyarakat, diantaranya dengan menerbitkan Undang-Undang (UU) Nomor 18 tahun 2018 tentang pengelolaan sampah. Karakteristik dan lingkungan yang berbeda tidak jarang membuat regulasi tidak berjalan sesuai dengan yang diharapkan oleh pemerintah dalam upaya pengelolaan sampah (Wiwik Kusrini, Herpendi, Muhammad Noor, 2019).

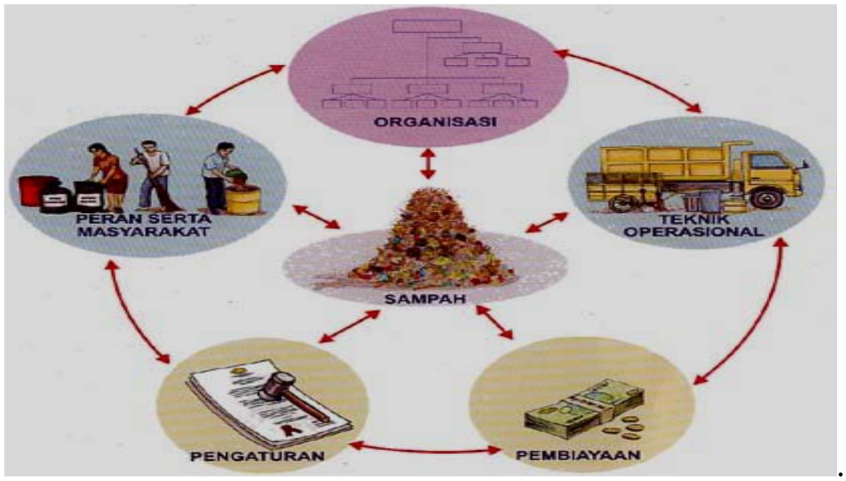
Kota-kota di dunia pada hakekatnya berkembang dengan karakteristik yang berbeda-beda, karena perkembangan kota sangat dipengaruhi oleh keadaan geografis dan sejarah/kebudayaan. Keadaan geografis kota lebih mempengaruhi fungsi dan bentuk kota, sedangkan sejarah dan kebudayaan akan mempengaruhi karakteristik dan sifat kemasyarakatan Kota (Branch, 1995: 37-38). Perkembangan kota yang cepat membawa dampak pada masalah lingkungan. Perilaku manusia terhadap lingkungan akan menentukan wajah kota, sebaliknya lingkungan juga akan mempengaruhi perilaku manusia. Lingkungan yang bersih akan meningkatkan kualitas hidup (Alkadri et al, 1999:159).

Perkembangan kota akan diikuti pertambahan jumlah penduduk, yang juga akan di ikuti oleh masalah – masalah sosial dan lingkungan. Salah satu masalah lingkungan yang muncul adalah masalah persampahan. Permasalahan lingkungan yang terjadi akan menyebabkan penurunan kualitas lingkungan (Alkadri et al, 1999:163).

Sampah akan menjadi beban bumi, artinya ada resiko-resiko yang akan ditimbulkannya (Hadi, 2000:40). Ketidakpedulian terhadap permasalahan pengelolaan sampah berakibat terjadinya degradasi kualitas lingkungan yang tidak memberikan kenyamanan untuk hidup, sehingga akan menurunkan kualitas kesehatan masyarakat. Degradasi tersebut lebih terpicu oleh pola perilaku masyarakat yang tidak ramah lingkungan, seperti membuang sampah di badan air (Alkadri et al., 1999:264) sehingga sampah akan menumpuk di saluran air yang ada dan menimbulkan berbagai masalah turunan lainnya. Kondisi ini sering terjadi di wilayah-wilayah padat penduduk di perkotaan.

Persoalan lingkungan yang selalu menjadi isu besar di hampir seluruh wilayah perkotaan adalah masalah sampah (Febrianie dalam Kompas 10 Januari 2004). Arif Rahmanullah dalam Kompas, 13 Agustus 2003 mengatakan bahwa laju pertumbuhan ekonomi di kota dimungkinkan menjadi daya tarik luar biasa bagi penduduk untuk hijrah ke kota (urbanisasi). Akibatnya jumlah penduduk semakin membengkak, konsumsi masyarakat perkotaan melonjak, yang pada akhirnya akan mengakibatkan jumlah sampah juga meningkat. Pertambahan jumlah sampah yang tidak diimbangi dengan pengelolaan yang ramah lingkungan akan menyebabkan terjadinya perusakan dan pencemaran lingkungan (Tuti Kustiah, 2005:1). Lebih jauh lagi, penanganan sampah yang tidak komprehensif akan memicu terjadinya masalah sosial, seperti amuk massa, bentrok antar warga, pemblokiran fasilitas TPA (Hadi, 2004)

Pertumbuhan jumlah sampah di kota-kota di Indonesia setiap tahun meningkat secara tajam. Sehingga diperlukan kebijakan yang tepat agar sampah yang di perkotaan khususnya, tidak menjad bom waktu di masa mendatang. Saat ini hampir seluruh pengelolaan sampah berakhir di TPA sehingga Saat ini hampir seluruh pengelolaan sampah berakhir di TPA sehingga menyebabkan beban TPA menjadi sangat berat, selain diperlukan lahan yang cukup luas, juga diperlukan fasilitas perlindungan lingkungan yang sangat mahal. Semakin banyaknya jumlah sampah yang dibuang ke TPA salah satunya disebabkan belum dilakukannya upaya pengurangan volume sampah secara sungguh-sunguh sejak dari sumber (Tuti Kustiah : 2005:3).



Gambar 1.1 Skema Manajemen Pengelolaan Sampah
Sumber : Departemen Pekerjaan Umum, (SNI 19-2454-2002)

Sampah merupakan masalah sebagian besar kota yang ada di Indonesia. Kota Parepare merupakan salah satu kota yang ada di Indonesia yang sangat serius mengurusi maasalah sampah. Usaha yang dilakukan pemerintah Kota Parepare dapat dinyatakan berhasil yang dibuktikan dari Piala Adipura yang didapatkan. Namun keberhasilan tersebut masih memiliki sedikit permasalahan yaitu sering terjadinya penumpukan penumpukan sampah di daerah tertentu yang disebabkan tidak disiplinnya masyarakat dalam membuang sampah (Siti Sauda, eka Puji Agustini, 2020). Dari kondisi tersebut tentunya pemerintah Kota Parepare melalui Dinas Kebersihan dan Lingkungan Hidup harus membuat mekanisme pengelolaan sampah yang dapat memonitoring dengan mudah tingkat penumpukan sampah. Cara yang paling memungkinkan yaitu dengan memanfaatkan teknologi informasi berupa aplikasi mobile yang dapat diakses melalui smartphone. Upaya tersebut tentunya sebagai bentuk untuk menciptakan keadaan lingkungan yang bersih dan sehat di Kota Parepare.

Dengan meningkatnya kebutuhan masyarakat dibidang teknologi yang semakin canggih kita dapat memanfaatkannya untuk membantu pengolahan sampah. Salah satunya melalui *android*. Oleh karena itu, penelitian ini akan merancang sebuah aplikasi berbasis *android*. Aplikasi tersebut diharapkan mampu membantu pemerintah di Kota Parepare agar dapat ikut andil memantau dan memberi informasi terkait penumpukan sampah sehingga dapat ditanggulangi dengan baik dan maksimal serta diharapkan pula dapat dijadikan sebagai bahan evaluasi dan penunjang pengambilan keputusan.

B. Rumusan Masalah

Dengan mengacu pada latar belakang di atas maka disusun rumusan masalah yang akan dibahas yakni, bagaimana memonitoring pengelolaan sampah Kota Parepare menuju *smart clean* dengan aplikasi *mobile*.?

C. Batasan Masalah

Dalam penyusunan tugas akhir ini perlu adanya pengertian pada pembahasan yang terfokus sehingga permasalahan tidak melebar. Adapun batasan dalam penelitian ini adalah:

1. Aplikasi ini dapat berjalan pada *smartphone* berbasis *Android* minimal versi 5.0 Lolipop.
2. Aplikasi dirancang untuk diaplikasikan di Kota Parepare.
3. Aplikasi ini berbentuk *prototype* yang terdiri dari sebuah sensor pengukur tingkat ketinggian sampah dan aplikasi yang akan menerima notifikasi dari alat tersebut.

4. Sensor berfungsi untuk mengukur ketinggian sampah dan mengirim notifikasi ketika ketinggian sampah telah mencapai batas maksimal (penuh).
5. Aplikasi pada *smartphone* berfungsi untuk memonitoring tingkat penumpukan sampah di TPS dan menerima notifikasi dari sensor untuk melakukan pengangkutan sampah.
6. Aplikasi ini hanya dapat berfungsi dengan layanan koneksi internet.

D. Tujuan

Adapun tujuan dari penelitian ini adalah menganalisis dan merancang aplikasi *mobile* monitoring pengelolaan sampah di Kota Parepare menuju *smart clean*.

E. Manfaat

1. Bagi Akademik

Dapat memberikan suatu referensi yang berguna bagi dunia akademis khususnya dalam penelitian yang akan dilaksanakan oleh para peneliti yang akan datang dalam hal perkembangan teknologi *smartphone* berbasis *Android*.

2. Bagi Dinas Lingkungan Hidup Kota Parepare

Memudahkan petugas memotoring tingkat penumpukan sampah di TPS dan melakukan pengankutan lebih efektif.

3. Bagi Penulis

Menambah pengetahuan dan wawasan serta mengembangkan daya nalar dalam pengembangan teknologi *smartphone* berbasis *Android*.

F. Aplikasiatika Penulisan

Aplikasiatika mengenai keseluruhan isi tulisan, dapat kita rinci dengan menguraikan inti tiap bab sebagai berikut :

BAB I PENDAHULUAN

Bab ini menguraikan tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat dan aplikasiatika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisikan tentang uraian teoritis mengenai cara membuat Aplikasi Manajemen Pengelolaan Sampah berbasis *Android*.

BAB III METODE PENELITIAN

Bab ini membahas tentang lokasi dan waktu penelitian, jenis penelitian, metode pengumpulan data, alat dan bahan penelitian, tahapan penelitian, metode pengujian dan desain aplikasi.

BAB IV PEMBAHASAN

Bab ini merupakan analisis aplikasi yang berjalan dan aplikasi yang akan di usulkan, serta analisis pada perangkat yang akan digunakan untuk membangun aplikasi dan implementasi hasil program aplikasi yang dirancang, yaitu berupa tampilan program.

BAB V PENUTUP

Bab ini berisikan tentang kesimpulan hasil penelitian dan saran pengembangan lebih lanjut.

BAB II

TINJAUAN PUSTAKA

A. Penelitian Sebelumnya

Dari pesatnya kemajuan teknologi di zaman ini, *smartphone* pun sebagai salah satu alat komunikasi yang sekarang multifungsi banyak membantu manusia dalam segala bidang. Dalam pengelolaan sampah, *smartphone* pun dapat digunakan melalui aplikasi yang terpasang di dalamnya untuk mengirim informasi terkait masalah sampah.

Ada banyak penelitian yang dilakukan dalam merancang aplikasi atau aplikasi untuk membantu dalam menanggulangi persoalan sampah. Akan tetapi, metode dan media yang digunakan berbeda. Hal tersebut dapat disebabkan oleh kebutuhan, perilaku masyarakat, dan juga teknologi yang digunakan. Beberapa penelitian sebelumnya yang dilakukan untuk menanggulangi persoalan sampah antara lain:

Akhmad aludi (2017), Universitas Muhammadiyah Malang jurusan Teknik Informatika melakukan penelitian "Rancang Bangun Pemetaan Unit Dan Penjadwalan Pick Up Pada Bank Sampah Malang (BSM) Berbasis Android Dan Web Responsif". Tujuan dari penelitian ini adalah menghasilkan pemetaan untuk unit BSM dan penjadwalan petugas untuk pengambilan sampah di masyarakat.

Siti Sauda, Eka Puji Agustini (2020), Universitas Bina Darma program studi Teknik Informatika dalam jurnalnya " Implementasi *Prototype* Model dalam Pengembangan Aplikasi *Smart Cleaning* Sebagai Pendukung Aplikasi *Smart City*". Hasil dari penelitian ini berupa aplikasi *smart cleaning* yang memiliki akses pengguna masyarakat, *administrator* dan petugas. Masing-masing pengguna memiliki akses dan fitur tersendiri yang dapat memproses pengaduan masyarakat terkait penumpukan sampah.

Putri Hardianti (2019), Universitas Muhammadiyah Parepare prodi Teknik Informatika dalam skripsinya "Aplikasi Informasi layanan Sampah Terjadwal Berbasis Web". Pada penelitiannya merancang sebuah aplikasi aplikasi informasi pelayanan yang dapat mempermudah masyarakat dalam menyampaikan keluahan terkait persoalan sampah di Kota Parepare.

B. Aplikasi

Aplikasi berasal dari bahasa Inggris yaitu "*application*" yang artinya adalah penggunaan maupun penerapan. Yang jika disederhanakan maka bisa disimpulkan bahwa pengertian aplikasi adalah suatu penerapan perangkat lunak atau software yang dikembangkan dengan tujuan untuk melakukan tugas-tugas tertentu.

Menurut cara pengembangannya, ada 3 jenis aplikasi, antara lain :

1. Aplikasi *Mobile*

Aplikasi *Mobile* adalah aplikasi yang dikembangkan untuk dijalankan di perangkat *mobile* atau telepon selular.

2. Aplikasi *Desktop*

Aplikasi *desktop* adalah aplikasi yang dikembangkan untuk dapat dijalankan di perangkat *PC* maupun laptop.

3. Aplikasi *Web*

Aplikasi *Web* adalah aplikasi yang dikembangkan untuk dijalankan menggunakan komputer dan koneksi internet.

Aplikasi yang dikembangkan biasanya akan dapat dijalankan oleh banyak jenis aplikasi operasi. Contohnya adalah aplikasi *game* ponsel yang bisa berjalan pada aplikasi operasi *android* maupun pada *IOS.*

Dibawah ini adalah beberapa kriteria yang menunjukkan bahwa sebuah aplikasi bisa dikatakan berkualitas, diantaranya :

a. Dapat dijalankan pada banyak *platform.*
b. Dapat memenuhi kebutuhan pengguna *gadget*.
c. Dapat merespon instruksi dengan cepat bahkan pada *gadget* dengan spesifikasi *low-end*.

Dibawah ini adalah beberapa pengertian aplikasi menurut pandangan para ahli. Diantaranya adalah sebagai berikut :

1. Sri Widianti

Pengertian aplikasi menurut Sri Widianti ialah suatu program atau perangkat lunak yang dibuat sebagai *front end* sebuah aplikasi. Kemudian digunakan dalam mengelola data untuk digunakan menjadi suatu informasi yang bermanfaat untuk penggunanya.

2. Hengky W. Pramana

Pengertian aplikasi menurut Hengky W. Pramana ialah suatu perangkat lunak yang dibuat khusus untuk memenuhi kebutuhan berbagai aktivitas maupun pekerjaan manusia. Contohnya seperti aplikasi untuk layanan masyarakat, periklanan, aktivitas niaga, permainan dan lain sebagainya.

3. Rachmad Hakim S.

Pengertian aplikasi menurut Rachmad Hakim S. ialah sebuah *software* yang dikembangakan dengan tujuan tertentu. Bisa sebagai pengolah data, menjalankan permainan dan hal-hal lainnya.

4. Harip Santoso

Pengertian aplikasi menurut Harip Santoso ialah suatu kelompok file (*report, class, form*) yang dikembangkan guna mengerjakan sebuah kegiatan tertentu yang saling berhubungan. Contohnya seperti aplikasi *payroll* dan *fixed asset*.

5. Ali Zaki dan Smitdev Community

Pengertian aplikasi menurut Ali Zaki dan Smitdev *Community* ialah suatu komponen yang digunakan sebagai media pengoperasian pengolahan dokumen atau data.

C. Pengelolaan Sampah

Menurut Undang-Undang Nomor 18 Tahun 2008 tentang Manajemen Pengelolaan Sampah, manajemen pengelolaan sampah adalah kegiatan aplikasiatis, menyeluruh, dan berkesinambungan yang meliputi pengurangan dan penanganan sampah. Kemudian menurut Direktorat PLP, Dirjen Cipta Karya Departemen PU (2003), penanganan sampah adalah upaya yang meliputi kegiatan pemilahan, pengumpulan, pemindahan, pengangkutan, pengolahan, dan pemrosesan akhir sampah.

Sedangkan menurut Hadiwiyoto (1983: 23), manajemen pengelolaan sampah ialah usaha untuk mengatur atau mengelola sampah dari proses pengumpulan, pemisahan, pemindahan, pengangkutan, sampai pengolahan dan pembuangan akhir. Sedangkan yang dimaksud dengan penanganan sampah ialah perlakuan terhadap sampah untuk memperkecil atau menghilangkan masalah-masalah yang ada kaitannya dengan lingkungan, yang dapat berbentuk membuang sampah saja atau mengembalikan (*recycling*) sampah menjadi bahan-bahan yang bermanfaat. Sehingga dari kedua pengertian tersebut dapat disimpulkan bahwa yang dimaksud dengan pengelolaan atau penanganan sampah ialah usaha untuk mengelola sampah dengan tujuan untuk menghilangkan masalah-masalah yang berkaitan dengan lingkunganuntuk mencapai tujuan yaitu kota yang bersih, sehat, dan teratur (Salipandang, 2011: 8-9).

D. Dinas Lingkungan Hidup Kota Parepare

Kantor Dinas Lingkungan Hidup Kota Parepare alamat jalan Jendral Ahmad Yani KM 6 memiliki visi misi antara lain :

Visi : Melihat potensi dan permasalahan lingkungan sejalan dengan pembangunan dan pengembangan wilayah Kota Parepare saat ini serta mendukung tercapainya visi misi Walikota Parepare, maka Dinas Lingkungan Hidup (DLH) Kota Parepare merumuskan visi di penghujung pelaksanaan RPJMD, yaitu tahun 2017-2018 adalah : "Terwujudnya Parepare Bersih, Hijau dan Peduli Lingkungan".

Rumusan ini merupakan pandangan SKPD ke depan menyangkut kemana DLH Kota Parepare harus dibawa dan diarahkan agar dapat berkarya secara konsisten, aspiratif, inovatif, dan produktif. DLH Kota Parepare melalui visi ini, memberikan gambaran menantang terkait keadaan masa depan yang berisikan harapan dan citra yang ingin diwujudkan. Dari visi tersebut di atas termuat pokok-pokok visi sebagai berikut :

- **Bersih,** merupakan cerminan perilaku seluruh *stakeholder* yang secara sadar dan aktif untuk menjaga kebersihan lingkungan dan memanfaatkan sampah dengan prinsip 3R (*Reduce, Reuse,* dan *Recycling*).
- **Hijau,** merupakan cerminan perilaku seluruh *stakeholder* untuk menjaga serta meningkatkan kuantitas dan kualitas ruang terbuka hijau sebagai paru-paru dunia pendukung resapan air.

- **Peduli lingkungan,** merupan paradigma yang menempatkan perlindungan dan pengelolaan lingkungan hidup sebagai kontrol dalam melaksanakan aktivitas dan menjadi cerminan bagi masyarakat.

Sejalan dengan visi tersebut, maka perumusan misi pula harus dilakukan oleh DLH Kota Parepare sebagai penjabaran dari visi agar dapat terwujud. Melalui misi ini diharapkan seluruh anggota organisasi dan pihak yang berkepentingan dapat mengetahui dan mengenal kebradaan dan peran DLH Kota Parepare dalam penyelenggaraan pemerintahan. Adapun misi yang diemban sebagai berikut :

- Meningkatkan kualitas kebersihan lingkunan sesuai standar yang berbasis masyarakat dan penerapan metode/teknologi ramah lingkungan.
- Meningkatkan kualitas dan kuantitas ruang terbuka hijau.
- Meningkatkan perlindungan dan pengelolaan lingkungan hidup dalam kehidupan bermasyarakat.
- Meningkatkan fungsi dan pelayanan SKPD.

E. *Android*

1. Definisi *Android*

Android merupakan aplikasi operasi untuk telepon seluler yang berbasis *Linux*. *Android* menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak (Arifianto, 2011: 1).

2. Komponen *Android*

Ada empat jenis komponen pada aplikasi *Android*, yaitu:

a. *Activities*, suatu *activities* akan menyajikan *user interface* (UI) kepada pengguna, sehingga pengguna dapat melakukan interaksi untuk menjalankan fungsi tertentu.Sebuah aplikasi *Android* bisa jadi hanya memiliki satu *activity*, tetapi umumnya aplikasi memiliki banyak *activity* tergantung pada tujuan aplikasi dan desain dari aplikasi tersebut.

b. *Service*, *service* tidak memiliki *Graphic User Interface* (*GUI*), tetapi *service* berjalan secara *background* untuk melakukan operasi-operasi yang *longrunning* (proses yang memakan waktu cukup lama) atau melakukan operasi untuk proses *remote*.

c. *Broadcast Reciever*, *broadcastreciever* berfungsi menerima dan bereaksi untuk menyiapkan notifikasi. *Broadcast Reciever* tidak memiliki *User Interface* (*UI*) tapi memiliki sebuah *Activity* untuk merespon informasi yang mereka terima ataukepada pengguna. *Broadcast receiver* hanyalah pintu gerbang menuju komponen lain dan memang dirancang untuk hanya melakukan kerja seminimal mungkin.

d. *Content Provider*, *content provider* membuat kumpulan aplikasi data secara spesifik sehingga bisa digunakan oleh aplikasi lain. *Content providers* juga berguna untuk membaca dan menulis data yang berstatus *private* dan tidak dibagikan ke suatu aplikasi (Safaat, 2014: 9-10).

3. Komponen Kebutuhan Aplikasi

a. *Android* Studio

Android Studio build system adalah *toolkit* yang digunakan untuk membangun, tes, menjalankan, dan membuat paket dari aplikasi Android. *Android*

Studio adalah lingkungan *pengembangan Android* baru berdasarkan Intelli J IDEA. Mirip dengan *Eclipse* dengan *ADT Plugin, Android Studio* menyediakan alat pengembang *Android* terintegrasi untuk pengembangan dan *debugging*. Di atas kemampuan tersebut *Android studio* menawarkan:

1. Berbasis *Gradle* membangun dukungan.
2. *Refactoring Android* - spesifik dan perbaikan yang cepat.
3. Alat *Lint* untuk menangkap kinerja, kegunaan, kompatibilitas versi dan masalah lainnya.
4. *Pro Guard* dan aplikasi-penanda tanganan kemampuan.
5. Penyihir berbasis *template* untuk membuat *desain Android* umum dan komponen.
6. Sebuah *layout editor* kaya yang memungkinkan Anda untuk *drag-and-drop*UI komponen, *layout* pratinjau pada beberapa konfigurasi layar.

b. *SDK* (*Software Development Kit*)

Android SDK adalah *tools API* (*Aplication Programming Interface*) yang diperlukan untuk memulai mengembangkan aplikasi pada *platform Android* menggunakan bahasa pemrograman *java*. *Android* merupakan subset perangkat lunak untuk *ponsel* yang meliputi aplikasi operasi, *middleware* dan aplikasi kunci yang dirilis oleh *Google*. Saat ini *SDK* disediakan

untuk alat bantu dan *API* untuk memulai mengembangkan aplikasi pada *platformAndroid* menggunakan bahasa pemrograman *java* (Safaat, 2014: 5).

c. ADT (*Android Development Tools*)

Android Development Tools (ADT) adalah *plug-in* untuk *Eclipse IDE* yang dirancang untuk memberikan lingkungan yang *powerfull* dan terpadu untuk membangun aplikasi *Android*. ADT memperluas kemampuan Eclipse untuk mempercepat dalam pembuatan *projectAndroid* baru, membuat aplikasi UI, menambahkan komponen berdasarkan *Android Framework API*, *debug* aplikasi menggunakan *Android tool SDK*, dan bahkan *ekspor unsigned* .apk file dalam rangka untuk mendistribusikan aplikasi. Mengembangkan apikasi *Android* di *Eclipse* dengan ADT sangat dianjurkan dan merupakan cara tercepat untuk memulai membuat aplikasi *Android*. Dengan dipandu *set-up project*, serta integrasi peralatan, *customXML editor*, dan *debug panel ouput*, ADT dapat memberikan dorongan luar biasa dalam mengembangkan aplikasi *Android* (Sari, 2011: 7).

F. *Google Firebase*

1. Apa itu *Firebase*

Firebase adalah suatu layanan dari *Google* untuk memberikan kemudahan bahkan mempermudah para *developer* aplikasi dalam mengembangkan aplikasinya. *Firebase* alias BaaS (*Backend as a Service*) merupakan solusi yang ditawarkan oleh *Google* untuk mempercepat pekerjaan *developer*. Dengan menggunakan *Firebase*, apps *developer* bisa fokus dalam mengembangkan aplikasi tanpa memberikan *effort* yang besar untuk urusan *backend*.

Singkat cerita mengenai sejarah dari *Firebase* didirikan pertama kali pada tahun 2011 oleh Andrew Lee dan James Tamplin. Produk *Firebase* yang pertama kali adalah Realtime Database. Realtime Database digunakan *developer* untuk menyimpan data dan *synchronize* ke banyak *user*. Kemudian ia berkembang sebagai layanan pengembang aplikasi. Pada bulan Oktober 2014, perusahaan tersebut diakuisisi oleh *Google*.

Mengenai segi layanan, dulu *Firebase* memberikan *service trial* (percobaan), namun saat ini kamu bisa memanfaatkan dan menggunakan layanan *Firebase* secara *free* (gratis). Tentu saja dengan adanya batasan-batasan tertentu.

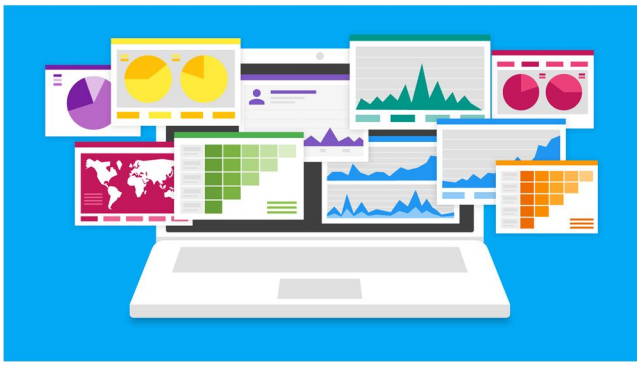
Layanan-layanan yang tersedia dari *Firebase* ada 2 pilihan, di antaranya:

a. **SPARK** : kita bisa menggunakan layanan secara gratis.
b. **BLAZE** : kita akan dikenakan biaya sesuai dengan pemakaian layanan.

2. Jenis atau Fitur *Firebase*

Selanjutnya kita akan membahas jenis atau fitur dari *Firebase* itu sendiri, berikut detailnya:

a. *Firebase Analytics*

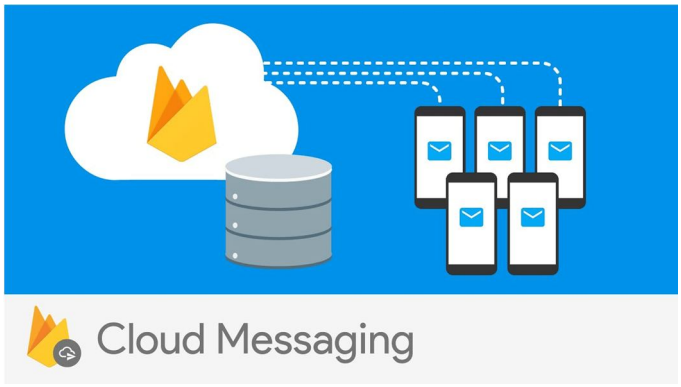


Gambar 2.1 *Firebase Analytics*

Fitur *Analytics* adalah salah satu fitur pada *Firebase* yang digunakan sebagai koleksi data dan *reporting* untuk aplikasi *Android* maupun iOS. Koleksi data pun bervariasi. Sebagai contoh, kamu dapat membuat suatu laporan atau *report* untuk pengguna aplikasi di negara Indonesia saja, atau mungkin negara lain seperti Singapura. Kamu juga bisa melihat bagian mana saja dari aplikasi yang paling sering digunakan oleh *user*.

Fitur ini mempunyai kelebihan yang memungkinkan kita untuk bisa membuat segmentasi *user* berdasarkan *user attribute. User attribute* adalah suatu parameter yang bisa kita gunakan sebagai *filter* yang bertujuan untuk *reporting* dan notifikasi. Contohnya pada aplikasi *online shop*. Dengan *user attribute*, kamu bisa tahu jumlah *user* yang membeli *handphone* merk ‘O’ atau bahkan bisa mencari tahu jam berapa transaksi yang dilakukan *user* sering terjadi.

b. *Firebase Cloud Messaging and Notifications*



Gambar 2.2 *Firebase Cloud Messaging and Notifications*

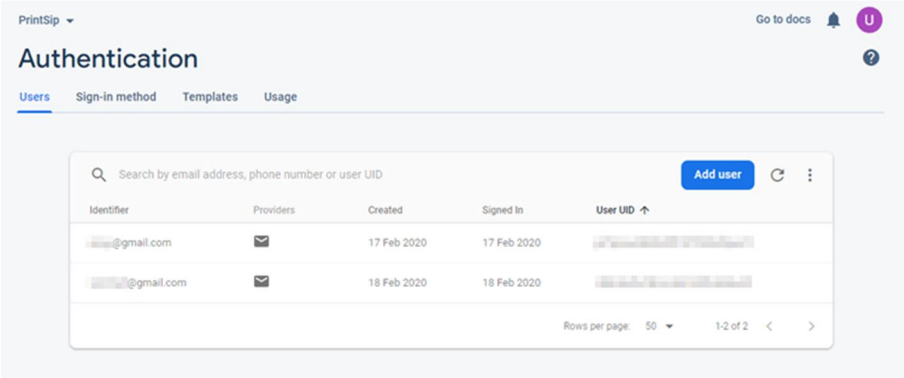
FCM (*Firebase Cloud Messaging)* yaitu menyediakan koneksi yang handal dan tentunya hemat baterai antar server maupun antar *device*. Sehingga kamu dapat mengirim dan menerima pesan serta notifikasi di *Android*, iOS, dan web tanpa perlu biaya.

Untuk menargetkan pesan lanjutan, kamu bisa targetkan pesan dengan mudah menggunakan *segment* yang telah ditentukan sebelumnya yakni menggunakan demografi dan *behavior*/perilaku. Anda dapat menargetkan pesan ke perangkat yang telah berlangganan pada topik tertentu. Selain itu, Anda bisa juga menargetkan hanya ke satu perangkat untuk mendapatkan informasi data yang terperinci. Biasanya ini dilakukan untuk proses pengujian.

Pesan notifikasi ini terintegrasi sepenuhnya dengan *Google* Analytics for *Firebase*, sehingga kamu memiliki akses pada interaksi dan tracking konversi

secara detaill. Nah, Anda dapat memantau suatu efektivitas dari satu *dashboard* tanpa perlu *coding* atau membuat program sendiri.

c. *Firebase Authentication*



Gambar 2.3 *Firebase Authentication*

Firebase Authentication adalah salah satu layanan *back-end,* fitur *Android* dan iOS, SDK yang mudah digunakan, dan tampilan *interfaces* yang siap pakai untuk mengautentikasi pengguna ke aplikasi yang kamu buat. *Firebase Authentication* mendukung autentikasi menggunakan nomor telepon, sandi, penyedia identitas gabungan populer seperti seperti *Google*, *Facebook*, dan sebagainya.

Firebase Authentication terintegrasi dengan fitur layanan *Firebase* lainnya. Aplikasi ini memanfaatkan berbagai jenis standar industri, seperti OAuth 2.0 dan *OpenID Connect*, yang memudahkan integrasi dengan *backend* khusus buatanmu.

Kamu juga dapat memudahkan pengguna untuk *login* ke aplikasi dengan menggunakan fitur *Firebase* UI (tampilan *interfaces*), sebagai alternatif *full drop-in authentication*.

d. *Firebase Cloud Firestore*

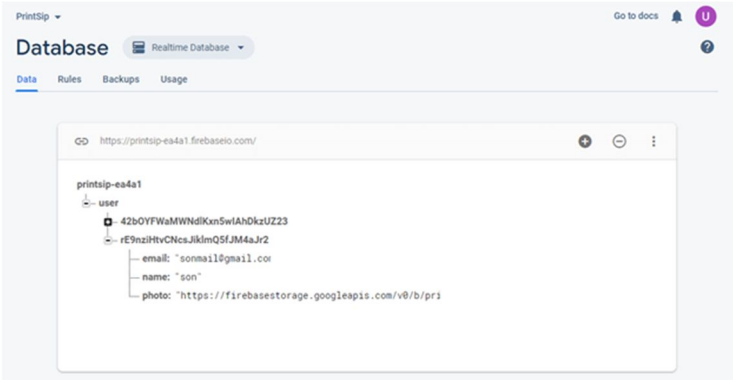


Gambar 2.4 *Firebase Cloud Firestore*

Cloud Firestore adalah *database* yang bersifat fleksibel dan terukur untuk pengembangan perangkat seperti seluler, web, dan server di *Firebase* dan *Google Cloud Platform*. Seperti halnya *Firebase Realtime Database*, *Cloud Firestore* membuat datamu tetap terkoneksi di aplikasi *user* melalui *listener realtime* dan menawarkan layanan secara *offline* untuk aplikasi seluler dan web. Dengan begitu, kamu dapat membuat aplikasi yang *powerfull*, responsif, dan mampu bekerja tanpa bergantung pada latensi koneksi internet. *Cloud Firestore* merupakan

database NoSQL yang dihosting di *cloud* dan dapat diakses melalui SDK *real* oleh aplikasi iOS, *Android* dan web.

e. *Firebase Realtime Database*



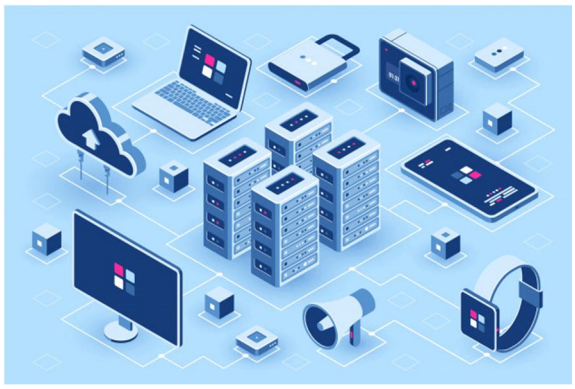
Gambar 2.5 *Firebase Realtime Database*

Firebase Realtime Database adalah *database* yang di-*host* melalui *cloud*. Data disimpan dan dieksekusi dalam bentuk *JSON* dan disinkronkan secara *realtime* ke setiap *user* yang terkoneksi. Hal ini berfungsi memudahkan kamu dalam mengelola suatu *database* dengan skala yang cukup besar. Ketika kamu membuat aplikasi lintas-platform/*multiplatform* menggunakan SDK Android, iOS, dan juga JS (*JavaScript)*, semua pengguna akan berbagi sebuah instance *Realtime Database* dan menerima *update*-an data secara serentak dan otomatis.

Kemampuan lain dari *Firebase Realtime Database* adalah tetap responsif bahkan saat *offline* karena SDK *Firebase Realtime Database* menyimpan data

langsung ke *disk device* atau memori lokal. Setelah perangkat terhubung kembali dengan internet, perangkat pengguna (*user*) akan menerima setiap perubahan yang terjadi.

f. *Firebase Hosting*



Gambar 2.6 *Firebase Hosting*

Selanjutnya ada *Firebase Hosting*, suatu layanan *hosting* konten web. Hanya dengan satu instruksi, kamu dapat mengimplementasikan aplikasi web serta menyajikan konten statis maupun dinamis ke CDN (jaringan penayangan konten) global dengan cepat.

Kegunaan dari *Firebase Hosting* itu sendiri yaitu mampu menayangkan konten melalui koneksi yang begitu aman, mengirimkan konten secara cepat, dan mendukung semua jenis konten untuk di *hosting*, mulai dari file HTML dan CSS hingga API atau layanan mikro *Express.js*.

G. *Internet Of Things*

Internet of Things (IoT) adalah suatu konsep dimana konektifitas internet dapat bertukar informasi satu sama lainnya dengan benda – benda yang ada disekelilingnya. *Internet of Things* (IoT) memungkinkan perangkat dikontrol dari jarak jauh dengan internet, maka hal tersebut menciptakan peluang untuk langsung menghubungkan dan mengitegrasikan dunia fisik ke aplikasi berbasis komputer menggunakan sensor, *database cloud,* dan internet. *Internet of Things* (IoT) merupakan istilah yang mengacu pada interkoneksi perangkat pintar, meliputi berbagai macam perangkat yang menggunakan *Internet Protocol* (IP) beserta sensor-sensor untuk keperluan apa pun, dimana pun dan kapan pun. Hal ini memungkinkan perangkat bertindak berdasarkan informasi baru yang diperoleh secara independen. Internet saat ini telah mendukung interkoneksi milyaran objek personal maupun industri, umumnya melalui aplikasi *cloud* (Purwanto, 2018).

H. *Google Maps* API

Ada banyak map editor yang dapat digunakan untuk membuat sebuah aplikasi informasi geografis. Salah satu aplikasi informasi geografis yang cukup populer adalah dengan menggunakan *Google Maps Engine*. *Google Maps Engine* merupakan sebuah layanan yang dimiliki *Google* yang dapat digunakan secara gratis. *Google Maps Engine* memiliki *Google Maps* API yang memugkinkan pengguna untuk dapat mengakses data peta tanpa perlu menyimpan seluruh data peta dalam server *Google Maps* API memiliki dua jenis API Key yaitu, Free API Key dengan jumlah *request* terbatas, dan API Key berbayar yang memiliki *unlimited requests* (F.T. Kusuma, T.N. Damayanti, D.N. Ramadan , 2017).

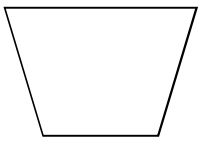
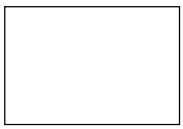
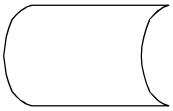
I. Daftar Simbol UML

1. Daftar Simbol *Flowmap Diagram*

Flowmap atau bagan alur adalah bagan yang menunjukan aliran di dalam program atau prosedur aplikasi secara logika. *Flowmap* ini berfungsi untuk memodelkan masukan, keluaran, proses maupun transaksi dengan menggunakan simbol-simbol tertentu. Pembuatan *flowmap* ini harus dapat memudahkan bagi pemakai dalam memahami alur dari aplikasi atau transaksi.

Tabel 2.1 ***Daftar Simbol Flowmap Diagram*** (Jogiyanto, 2001)

Simbol	Nama	Keterangan
	Terminator Awal / Akhir Program	Simbol untuk memulai dan mengakhiri suatu program
	Dokumen	Menunjukkan dokumen berupa dokumen *input* dan *output* pada proses manual dan proses berbasis computer

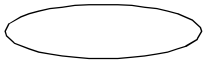
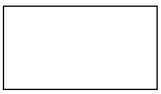
	Proses Manual	Menunjukkan kegiatan proses yang dilakukan secara manual
	Proses Komputer	Menunjukkan kegiatan proses yang dilakukan secara komputerisasi
	Arah Aliran Data	Menunjukkan arah aliran dokumen antar bagian yang terkait padasuatu System
	Penyimpanan Manual	Menunjukkan media penyimpanan data / infomasi secara manual
	Data	Simbol input/output digunakan untuk mewakili data i*nput/output*

2. Daftar Simbol *Use CaseDiagram*

Use case diagram merupakan gambaran skenario dari interaksi antara pengguna dengan aplikasi. *Use case diagram* menggambarkan hubungan antara aktor dan kegiatan yang dapat dilakukannya terhadap aplikasi.

Tabel 2.2 *Daftar Simbol Use Case Diagram* (Rosenberg, 2007)

Simbol	Nama	Keterangan
	*Actor*	Menspesifikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*

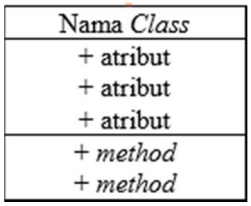
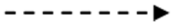
	*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan aplikasi yang menghasilkan suatu hasilterukur bagi suatu *actor*
	*System*	Menspesifikasikan paket yang menampilkan aplikasisecara terbatas
	*Unidirectionl Association*	Menggambarkan relasi antara *actor* dengan *use case* dan proses berbasis komputer
	*Dependencie or Instantitiates*	Menggambarkankebergantungan (*dependencies*) antar *item* dalam diagram
	Generalization	Menggambarkan relasi lanjut antar *use case* atau menggmabarkanstruktur pewarisan antar *actor*

3. Daftar Simbol *ClassDiagram*

Class Diagram merupakan diagram yang menggambarkan struktur aplikasi dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun aplikasi.

Tabel 2.3 *Daftar Simbol Class Diagram* (Sa‟adah, 2015)

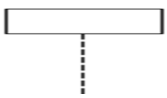
Simbol	Nama	Keterangan

	*Class*	Blok - blok pembangun pada pemrograman berorientasi obyek. Terdiri atas 3 bagian. Bagian atas adalah bagian nama dari *class.* Bagian tengah mendefinisikan properti/atribut *class*.Bagianakhirmendefinisikan*method-method* dari sebuah *class*.
	Association	Menggambarkan relasi asosiasi
	*Composition*	Menggambarkan relasi komposisi
	*Dependencies*	Menggambarkan relasi dependensi
	*Aggregation*	Menggambarkan relasi agregat

4. Daftar *Simbol SequenceDiagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar aplikasi berupa *message* terhadap waktu.

Tabel 2.4 *Daftar Simbol Sequence Diagram* (Rosenberg, 2007)

Simbol	Nama	Keterangan
	*Actor*	Orang atau divisi yang terlibat dalam suatu *system*
	*Object Lifeline*	Menyatakan kehidupan suatu objek dalam basis waktu
	*Activation*	Menyatakan objek dalam keadaan aktif dan berinteraksi

	*Message*	Menyatakan arah tujuan antara *object lifeline*
	Message (Return)	Menyatakan arah kembali antara *object lifeline*

5. Daftar Simbol *ActivityDiagram*

Activity Diagram adalah representasi grafis dari seluruh tahapan alur kerja yang mengandung aktivitas, pilihan tindakan, perulangan, dan hasil dari aktivitas tersebut. Diagram ini dapat digunakan untuk menjelaskan proses bisnis dan alur kerja operasional secara langkah demi langkah dari komponen suatu aplikasi.

Tabel 2.5 ***Daftar Simbol Activity Diagram*** (Herry, 2014)

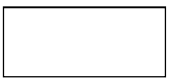
Simbol	Nama	Keterangan
	*Action*	*State* dari aplikasi yang mencerminkan eksekusi dari suatu aksi
	*Start State*	Bagaimana objek dibentukatau diawali
	*End State*	Bagaimana objek dibentukatau diakhiri
	*State Transtition*	*Statetranstition* menunjukkan kegiatan apa berikutnya setelahsuatukegiatan
	*Fork*	Percabangan yang menunjukkan aliranpada *activity diagram*

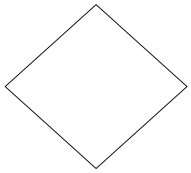
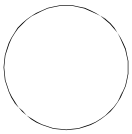
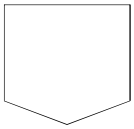
	*Join*	Percabangan yang menjadi arah aliranpada *activity diagram*
	*Decision*	Pilihan untuk mengambil Keputusan

6. Daftar Simbol *Flowchart*

Flowchart atau *Bagan alir* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur aplikasi secara logika. Bagan alir (*flowchart*) digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi.

Tabel 2.6 *Daftar Simbol Flowchart* (Kristanto, 2003)

Simbol	Nama	Keterangan
	*Terminator*	Permulaan atau akhir program
	Flow Line	Arah aliran program
	*Preparation*	Proses inisialisasi atau pemberian harga awal
	*Process*	Proses perhitungan atau proses pengolahan data
	*Input/Output Data*	Proses *input* atau *output* data, parameter, informasi
	*Predefined Process*	Permulaan sub program atau proses menjalankan sub program

	*Decision*	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	*On Page Connector*	Penghubung bagian-bagian *flowchart* yang ada pada satu halaman
	*Off Page Connector*	Penghubung bagian-bagian *flowchart* yang ada pada halaman berbeda

J. *Software Testing*

Software testing merupakan sebuah metode untuk mengetahui apakah *software* yang dibuat, berfungsi dengan baik dan benar. Tanpa adanya *software testing*, kita tidak dapat mengetahui apakah sebuah *software* sudah memenuhi semua kriteria yang dibutuhkan untuk user. *Software testing* sendiri dilakukan berdasarkan SDLC (*Software Development Life Cycle*), dengan kata lain kita melakukan eksekusi program dan selama proses yang dilakukan, memungkinkan kita untuk menemukan kesalahan atau *error*.

Pengujian *software testing* ini adalah tahap yang krusial dan cukup membutuhkan dana berkisar kurang lebih diatas 50% dari total *cost* pembuatan *software* tersebut. Beberapa tujuan dari *software testing* adalah meningkatkan

kualitas *software,* memeriksa aplikasi agar memenuhi *requirement* / kebutuhan, dan mengidentifikasi kelemahan, *error* dan kesalahan yang terjadi dalam *software* tersebut. Dalam *software testing*, ada siklus yang harus dilewati :

a. Analisis kebutuhan

   Melakukan analisa dari fase SDLC yang dilalui oleh *software*

b. Analisis Desain

   Analisis ini merupakan bentuk analisis desain dimana melakukan analisa bagian desain dan parameter yang perlu dilakukan *testing*

c. Test Desain

   Menyempurnakan strategi testing yang akan dilakukan

d. *Run test*

   Menjalankan *testing* dan mencari *error*, *bug* dan kesalahan yang ada

e. *Report Test*

   Memberikan *report* dari hasil *testing* kepada *developer* dan memberikan konklusi apakah *software* tersebut layak digunakan atau tidak.

Pengujian Aplikasi

1. White-box Testing

White Box Testing adalah salah satu cara untuk menguji suatu aplikasi atau *software* dengan melihat modul untuk memeriksa dan menganalisis kode program ada yang salah atau tidak. Jika modul ini dan telah diproduksi dalam output yang tidak memenuhi persyaratan, kode akan dikompilasi ulang dan diperiksa lagi sampai mencapai apa yang diharapkan singkatnya *White Box*

Testing ini menguji dengan cara melihat *Pure Code* dari suatu aplikai/*software* yang diuji tanpa memperdulikan Tampilan atau UI dari aplikasi tersebut.

Teknik *White-box Testing*

a. *Basis PathTesting*

Basis path testing merupakan metode yang memungkinkan perancang *testcase* untuk membuat pengukuran kompleksitas logikal dari rancangan prosedural dan menggunakan pengukuran ini sebagai panduan untuk mendefinisikan himpunan basis dari jalur eksekusi. *Test case* yang dibuat untuk menguji himpunan basis dijamin akan mengeksekusi setiap *statement* di dalam program sekurangnya sekali pada saat pengujian

b. *Flow Graph*

Flow graph merupakan notasi sederhana untuk merepresentasi *control flow.*

c. *Cyclomatic Complexity*

Cyclomatic complexity digunakan untuk mengetahui jumlah jalur yang perlu dicari. *Cyclomatic complexity* adalah *metric software* yang menyediakan ukuran kuantitatif dari kompleksitas logikal program. Nilai yang dihitung bagi *cyclomatic complexity* menentukan jumlah jalur-jalur yang independen dalam kumpulan basis suatu program dan memberikan jumlah tes minimal yang harus dilakukan untuk memastikan bahwa semua pernyataan telah dieksekusi sekurangnya satu kali.

Cyclomatic *complexity* mempunyai fondasi dalam teori *graph* dan dapat dihitung dengan satu dari tiga cara :

1) Jumlah region sama dengan *cyclomatic complexity*.
2) Cyclomatic *complexity*, V(G), untuk sebuah *flow graph*, G, didefnisikan sebagai: V(G) = E – N + 2 E adalah jumlah *edge* pada *flow graph*, dan N adalah jumlah node pada *flow graph*.
3) *Cyclomatic* complexity, V(G), untuk *flow graph*, G, juga didefinisikan sebagai: V(G) = P + 1 P adalah jumlah *predicate nodes* yang terdapat pada *flow graph* G.

d. Graph Matrix

Prosedur untuk membuat *flow graph* dan menentukan himpunan basis path dapat diterima berdasarkan mekanisme. Untuk mengembangkan *software* yang membantu pengujian basis path, sebuah struktur data yang disebut *graph matrix*, dapat sangat bermanfaat. *Graph matrix* adalah matriks kotak yang ukurannya (jumlah baris dan kolom) sama untuk jumlah *node* pada *flow graph*. Setiap baris dan kolom berhubungan dengan node yang teridentifikasi, dan data matriks berhubungan dengan koneksi (*edge*) antara

Kelemahan *White Box-Testing*:

a. Sangat mahal untuk dilakukan karena membutuhkan tester yang terampil untuk melakukan pengujian.

b. Pada perangkat lunak yang jenisnya besar, metode *white box testing* ini dianggap boros karena melibatkan banyak sumberdaya untuk melakukannya.
c. Tidak mempedulikan Tampilan UI aplikasinya.

2. *Black-box* Testing

Pada *Black Box* Testing dilakukan pengujian yang didasarkan pada detail aplikasi seperti tampilan aplikasi, fungsi-fungsi yang ada pada aplikasi, dan kesesuaian alur fungsi dengan bisnis proses yang diinginkan oleh *customer*. *Black-box Testing* ini lebih menguji ke Tampilan Luar (*Interface*) dari suatu aplikasi agar mudah digunakan oleh *Customer*. Pengujian ini tidak melihat dan menguji *souce code program*. *Black-box Testing* bekerja dengan mengabaikan struktur kontrol sehingga perhatianya hanya terfokus pada informasi *domain*.

Keuntungan dari *Black-box Testing* :

a. Penguji tidak perku memiliki pengetahuan tentang baasa pemograman tertentu
b. Pengujian yang dilakukan berdasarkan sudut pandang user agar dapat mengungkapkan inkosistensi atau ambiguitas dalam spesifikasi.
c. Programmer dan tester memiliki ketergantungan satu sama lain.

Kekurangan *Black-box Testing* :

a. Uji kasus sulit disain tanpa spesifikasi yang jelas
b. Kemungkinan memiliki pengulangan tes yang sudah dilakukan oleh programmer
c. Beberapa bagian back end tidak diuji sama sekali.

Teknik *Black-box Testing*

a. *Equivalence Partitioning*

   Cara kerja teknik ini adalah dengan melakukan partition atau pembagian menjadi beberapa partisi dari input data.

b. *Boundary Value Analysis*

   Teknik ini lebih fokus kepada *boundary*, dimana adakah *error* dari luar atau sisi dalam *software*, minimum, maupun *maximum* nilai dari *error* yang didapat.

c. *Fuzzing*

   Fuzz merupakan teknik untuk mencari *bug* / gangguan dari *software* dengan menggunakan injeksi data yang terbilang cacat ataupun sesi semi-otomatis.

d. *Cause-Effect Graph*

   Ini adalah teknik testing dimana menggunakan *graphic* sebagai pacuannya. Dimana dalam grafik ini menggambarkan relasi diantara efek dan penyebab dari *error* tersebut.

e. *Orthogonal Array Testong*

   Dapat digunakan jika *input domain* yang relatif terbilang kecil ukurannya, tetapi cukup berat untuk digunakan dalam skala besar.

f. *All Pair Testing*

   Dalam teknik ini, semua pasangan dari *test case* di desain sedemikian rupa agar dapat di eksekusi semua kemungkinan kombinasi diskrit

dari seluruh pasangan berdasar *input* parameternya. Tujuannya *testing* ini adalah memiliki pasangan *test case* yang mencakup semua pasangan tersebut.

g. *State Transition*

Testing ini berguna untuk melakukan pengetesan terhadap kondisi dari mesin dan navigasi dari UI dalam bentuk grafik.

BAB III

METODE PENELITIAN

A. Jenis Penelitian

Dalam melakukan penelitian ini, jenis penelitian yang digunakan adalah kualitatif lapangan. Dipilihnya jenis penelitian ini oleh penulis dikarenakan objek yang akan diteliti adalah suatu fenomena sosial dan dibutuhkan analisis untuk menyelesaikan persoalan sosial tersebut.

B. Lokasi dan Waktu Penelitian

1. Lokasi Penelitian

   Adapun lokasi penelitian ini dilakukan di Dinas Lingkungan Hidup Kota Parepare dan Kec. Soreang Kota Parepare.

2. Waktu Penelitian

Waktu yang dibutuhkan dalam pelaksanaan penelitian ini adalah ± 2 bulan.

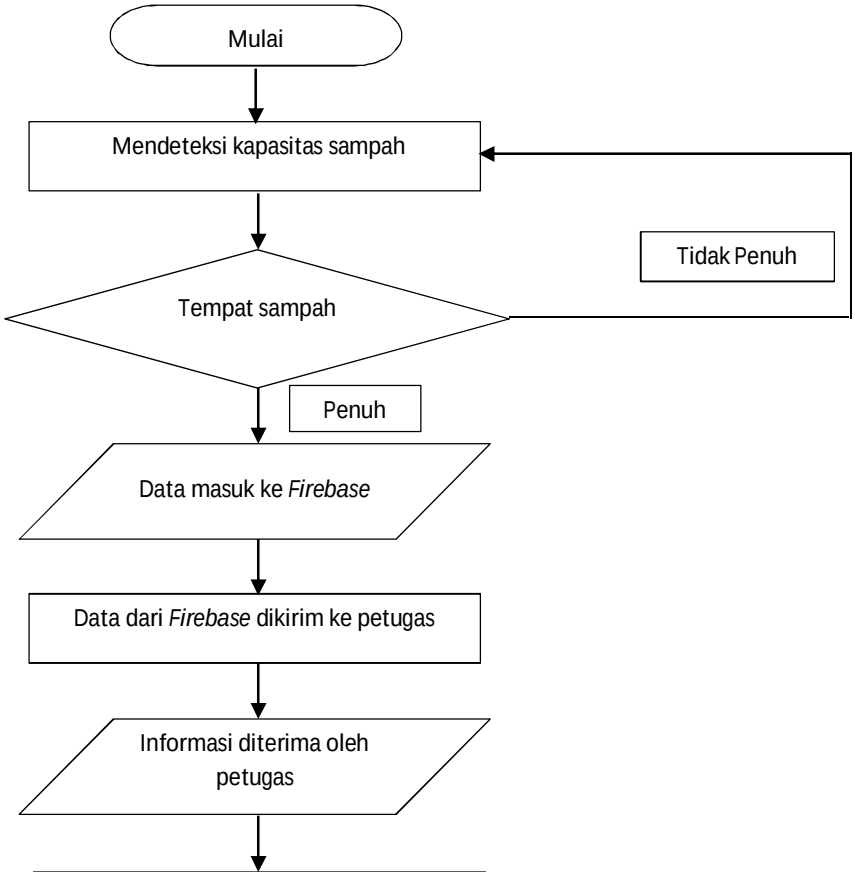
C. Alat dan Bahan Penelitian

Dalam pelaksanaan penelitian ini digunakan beberapa alat dan bahan yang terdiri dari Perangkat keras (*Hardware*) dan Perangkat Lunak (*Software*). Perangkat keras yang digunakan terdiri dari *smartphone android Oppo* A3s dengan spesifikasi *android* versi 8.1.0 Oreo, *Laptop* DELL *Latitude E7440* dengan spesifikasi *Procesor Intel Core i5*, SSD 120GB, RAM 4GB, Mikrokontroler ESP8266, Sensor *Ultra Sonic HC-SR04*, dan beberapa *Jumper Male to Female*. Sedangkan perangkat lunak yang digunakan adalah *Android Studio* untuk pembuatan aplikasi android, *Software Development Kit*, *Google Firebase*, dan *Arduino* IDE untuk memprogram ESP8266.

D. Rancangan Aplikasi

1. Aplikasi Yang Diusulkan

Adapun aplikasi yang di usulkan yaitu :



E. Teknik Pengumpulan data

Untuk mendukung keperluan pengumpulan data penelitian ini, peneliti memerlukan sejumlah data pendukung. Teknik pengumpulan data yang dilakukan disesuaikan dengan jenis data yang diambil sebagai berikut :

1. *Observasi*

   Observasi langsung terhadap objek yang diteliti yaitu aplikasi pengelolaan sampah di Kota Parepare.

2. Wawancara

   Wawancara langsung dengan Kepala Seksi Pengelolaan Kebersihan mengenai aplikasi pengelolaan sampah yang sedang berlangsung dan kendala yang dihadapi dalam pengelolaan sampah di Dinas Kebersihan Kota Parepare.

Adapun susunan wawancara ini adalah sebagai berikut :

Tema :Penanggulangan sampah di Kota Parepare.

Tujuan :Mengetahui aplikasi pengelolaan sampah dan kendala/masalah yang dihadapi di Kota Parepare.

Target Narasumber :Kepala Seksi Pengelolaan Kebersihan Kota Parepare

Waktu :Menyesuaikan waktu luang dari narasumber.

3. Studi Kepustakaan

Dalam metode ini, penulis menggunakan buku-buku referensi ataupun informasi yang merupakan pengumpulan data melalui literatur, jurnal, paper dan bacaan-bacaan yang ada kaitannya dengan judul penelitian.

4. Metode test

Metode pengujian yang digunakan pada penelitian ini adalah metode pengujian langsung yaitu dengan menggunakan pengujian *Black Box* dan *White Box*. Digunakan untuk menguji fungsi-fungsi khusus dari perangkat lunak yang dirancang. Kebenaran perangkat lunak yang diuji hanya dilihat berdasarkan keluaran yang dihasilkan dari data atau kondisi masukan yang diberikan untuk fungsi yang ada tanpa melihat bagaimana proses untuk mendapatkan keluaran tersebut. Dari keluaran yang dihasilkan, kemampuan program dalam memenuhi kebutuhan pemakai dapat diukur sekaligus dapat diketahui kesalahan-kesalahannya.

F. Tenik Analisis Data

Setelah semua data diperoleh, langkah selanjutnya adalah melakukan analisa terhadap data tersebut secara kualitatif. Karena penelitian ini bersifat kualitatif maka alat yang digunakan dalam analisis data adalah peneliti sendiri. Peneliti melakukan analisa data untuk mengidentifikasi kebutuhan, merancang aplikasi, mengimplementasikan aplikasi pada objek yang diteliti. Dalam tahap analisis data ini, dilakukan tahap – tahap sebagai berikut :

a. Pengumpulan data
b. Analisis aplikasi
c. Desain aplikasi
d. Pengujian program
e. Implementasi aplikasi

BAB IV

HASIL DAN PEMBAHASAN

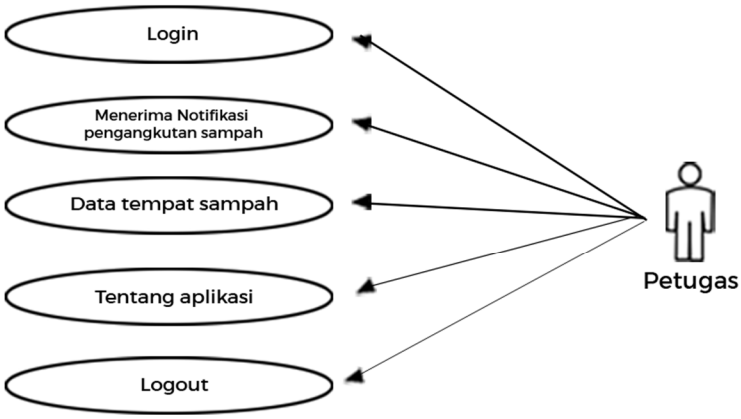
A. Analisis Kebutuhan Aplikasi

Analisis aliran data adalah analisis yang dilakukan untuk mempelajari pemanfaatan data pada setiap aktifitas dan juga bertujuan mengetahui aliran proses Informasi. Dalam analisis aplikasi ini, penulis menggunakan pegembangan orientasi objek sehingga menggunakan *Use Case Diagram, Activity Diagram, Sequence Diagram,* dan *Class Diagram* .

1. *Use Case Diagram*

Use Case Diagram adalah gambaran grafis dari beberapa atau semua *actor*, *use case*, dan interaksi diantaranya yang memperkenalkan suatu aplikasi. Analisis Aktor Aplikasi

Aktor menggambarkan segala pengguna aplikasi. Aktor dalam Aplikasi *mobile monitoring* pengelolaan sampah antara lain adalah petugas pengangkut sampah, Aktor yang memiliki hak akses memonitoring tempat sampah.



Gambar 4. 1 *Use Case Diagram* aplikasi

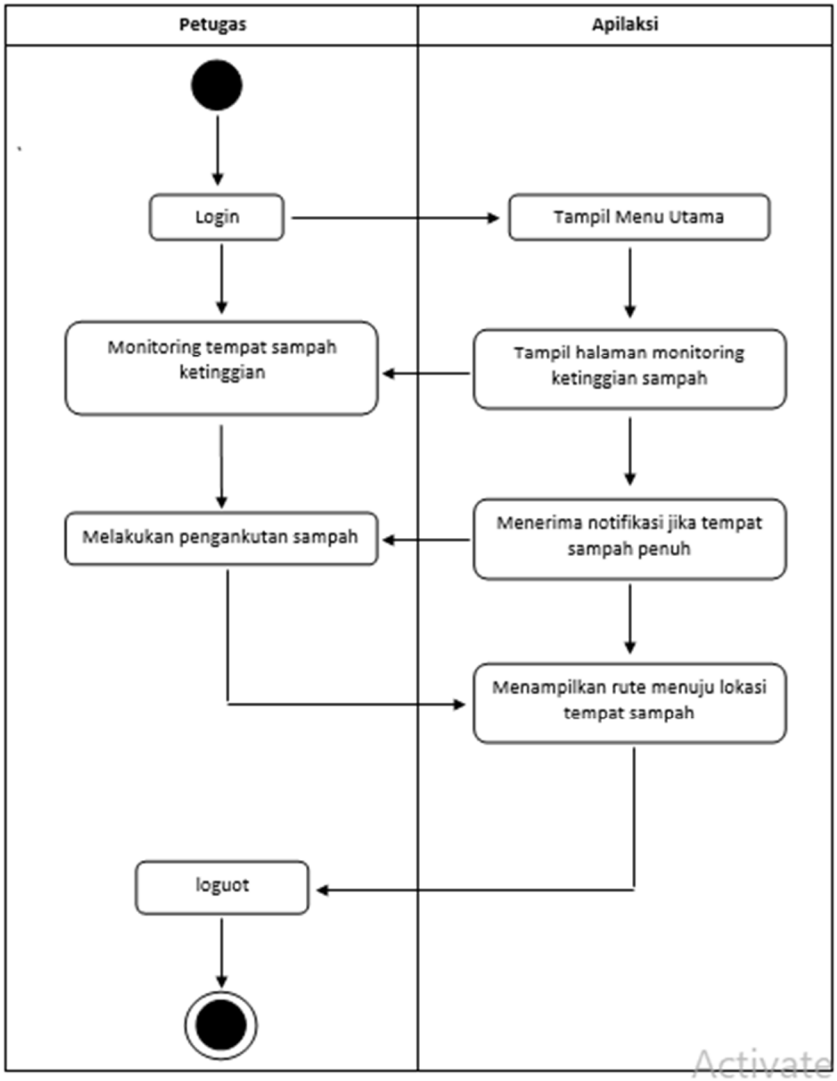
Tabel 4. 1 Penjelasan*Use Case Diagram* petugas

Nama *Use Case aplikasi*	**Deskripsi *Use Case aplikasi***
Petugas	Aktor.
Login	*Use Case* yang menjelaskan proses *Login* sebagai *Petugas*
Menerima notifikasi pengangkutan sampah	*Use Case* yang menjelaskan petugas menerima notifikasi pengangkutan sampah ketika penuh

Data tempat sampah	*Use Case* ini menjelaskan petugas dapat melihat data ketinggian dan status tempat sampah
Tentang Aplikasi	*Use Case* yang menjelaskan tentang versi aplikasi
Logout	*Use Case* ini menjelaskan proses Log out dari system

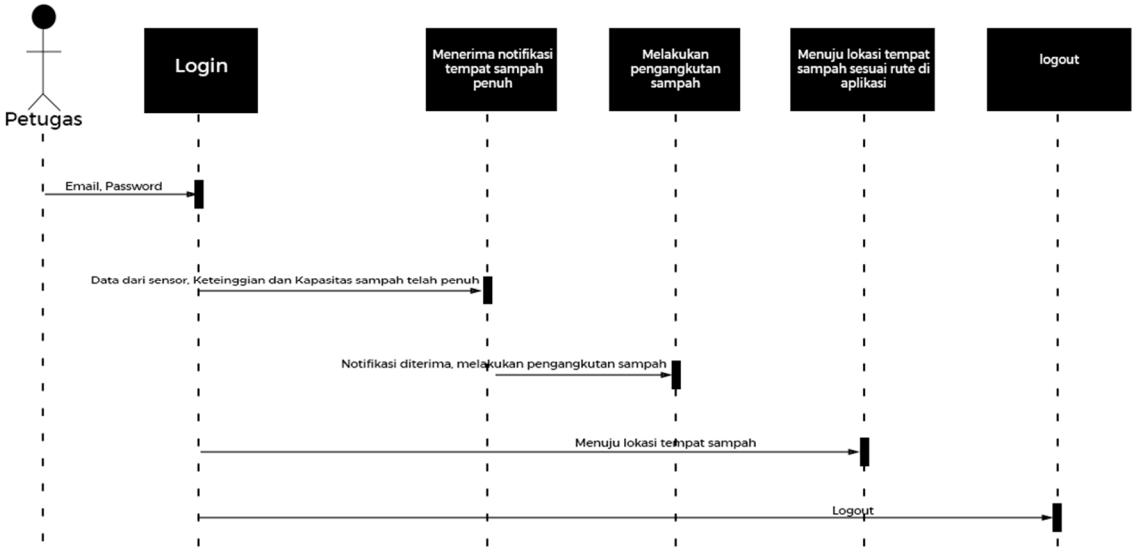
2. *ActivityDiagram*

Activity Diagram memodelkan alur kerja *(workflow)* sebuah proses bisnis dan urutan aktifitas dalam suatu proses. *Activity Diagram* yang ada di aplikasi yang penulis rancang sebagai berikut:



Gambar 4. 2 *Activity Diagram aplikasi*

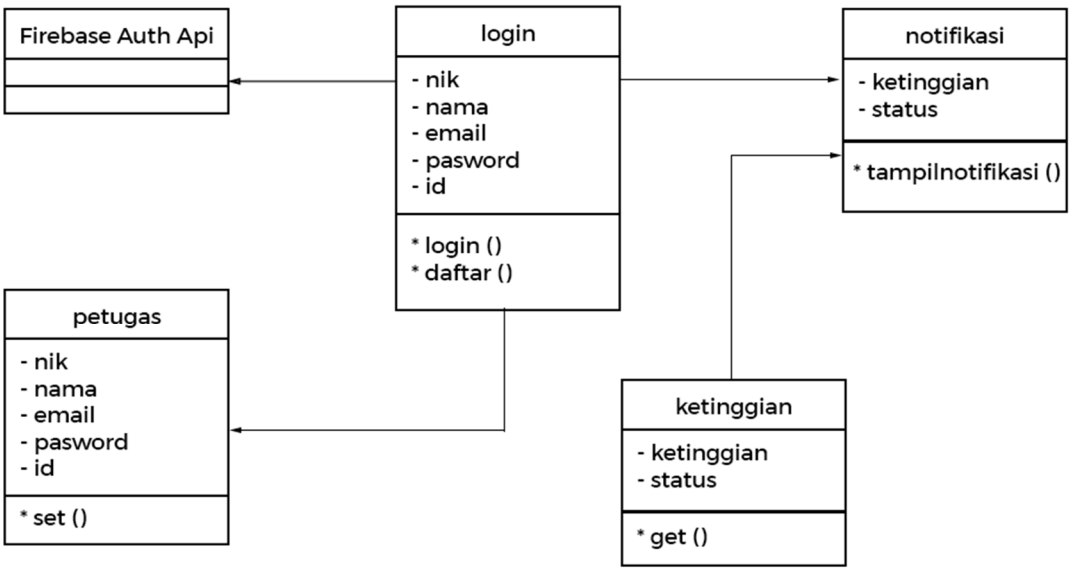
3. ***Diagram Sequence***



Gambar 4. 3 *Diagram Sequence*

4. *ClassDiagram*

Diagram Class menunjukkan rencana kelas-kelas yang akan dibentuk dalam perancangan basis data. Pada gambar 4.6 diperlihatkan bahwa aplikasi basis data terdiri dari 3 *class* yaitu *class login*, *class* notifikasi, dan *class* ketinggian. Dalam diagram terlihat aksi yang dilakukan masing-masing *class* dalam aplikasi dan hubungan antar *class* yang terjadi.



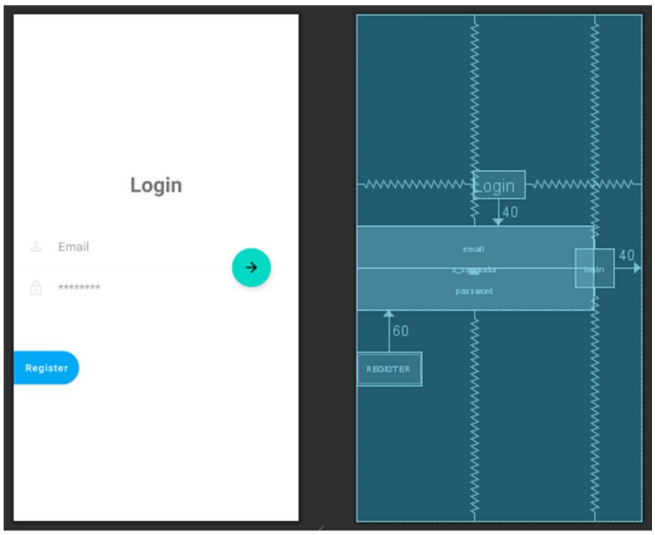
Gambar 4. 4 *Class Diagram*

Class login berhubungan dengan *Firebase Auth* API kemudian menulis data petugas ke *Firebase* Firestore Database menggunakan metode *set*. *Class* notifkasi mengambil data dari *Firebase Realtime Database* dengan metode *get* kemudian melakukan push notifikasi menggunakan metode tampilnotifikasi. *Class* Ketinggian menulis data dari sensor ke *Firebase Realtime Database* dengan metode *set*.

B. Perancangan Aplikasi

1. Halaman *Login*

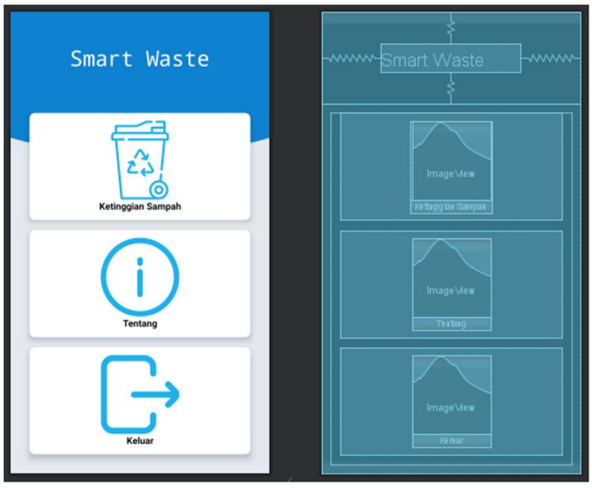
   Gambaran perancangan halaman *login* terdapat 2 *edit text* yaitu email, *password*, dan *button login* dan *register* seperti pada gambar 4.5 berikut.



Gambar 4.5 Halaman Login

2. Halaman *Dashboard*

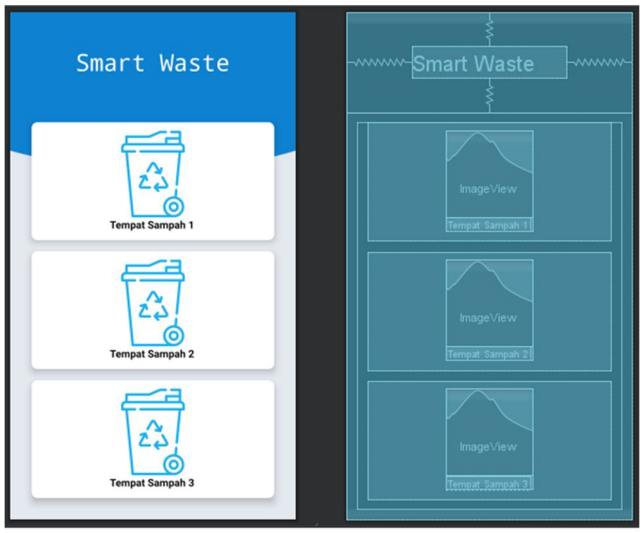
   Gambaran perancangan halaman *dashboard* terdapat *TextView* dan 4 *CardView.* Gambaran halaman *dashboard* ditunjukkan pada gambar 4.6.



Gambar 4.6 Halaman *Dashboard*

3. Halaman Ketinggian Sampah

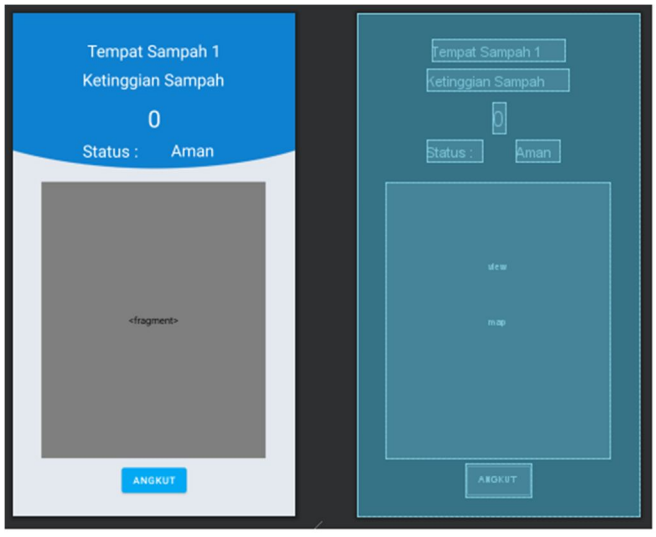
Gambaran perancangan halaman ketinggian sampah ditunjukkan pada gambar 4.7 berikut.



Gambar 4.7 Halaman Ketinggian Sampah

4. Halaman monitoring tempat sampah

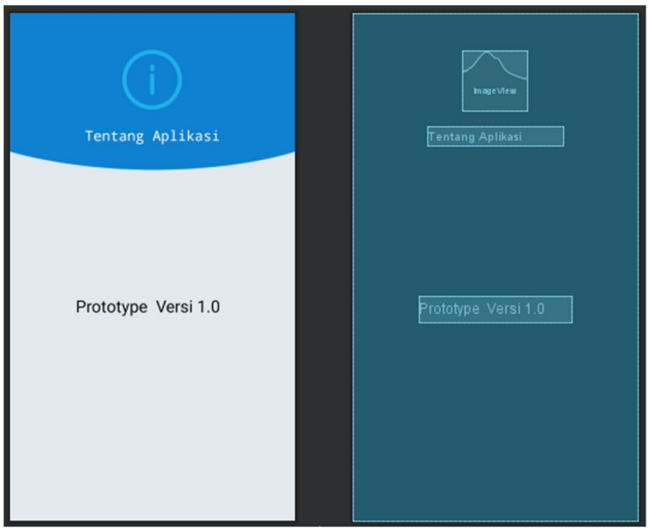
Gambaran perancangan halaman ketinggian sampah ditunjukkan pada gambar 4.8 berikut.



Gambaran 4.8 Halaman monitoring tempat sampah

5. Halaman Tentang

Gambaran perancangan halaman tentang ditunjukkan pada gambar 4.9 berikut.



Gambar 4.9 Halaman Tentang

C. Pengujian Aplikasi

1. Pengujian *Black Box*

Pengujian Aplikasi dengan Pengujian Blackbox. Pengujian ini didasarkan pada detail aplikasi seperti tampilan aplikasi dan kesesuain alur fungsi dengan proses yang diinginkan. Pengujian ini tidak menguji dan melihat source code program. Berikut ini adalah pengujian *Black Box* dari Aplikasi *Mobile* Monitoring Pengelolaan Sampah Kota Parepare Menuju *Smart Clean.*

Table 4.2 Pengujian *form login*

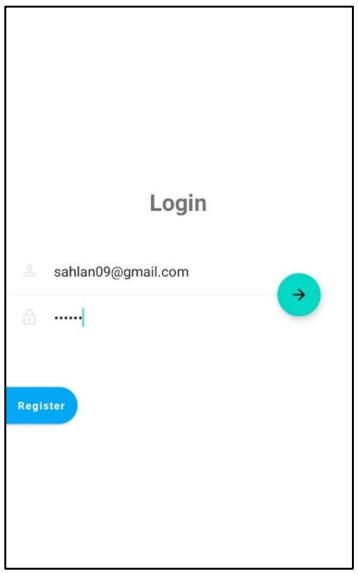
Test factor	Hasil	Keterangan
Pengujian *Login* dengan mengiput email dan *password* yang benar	✓	Berhasil *Login* dan akan masuk kehalaman *dashboard*
Screenshot Aplikasi		
		

Table 4.3 Pengujian form login password/email salah

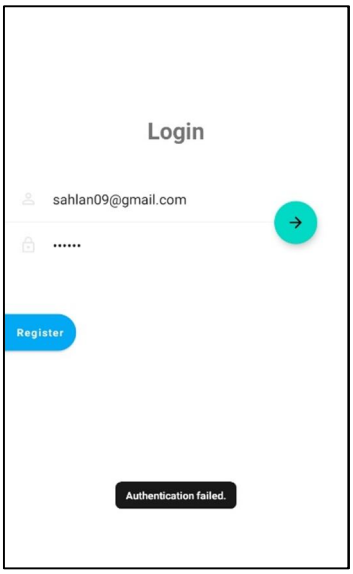
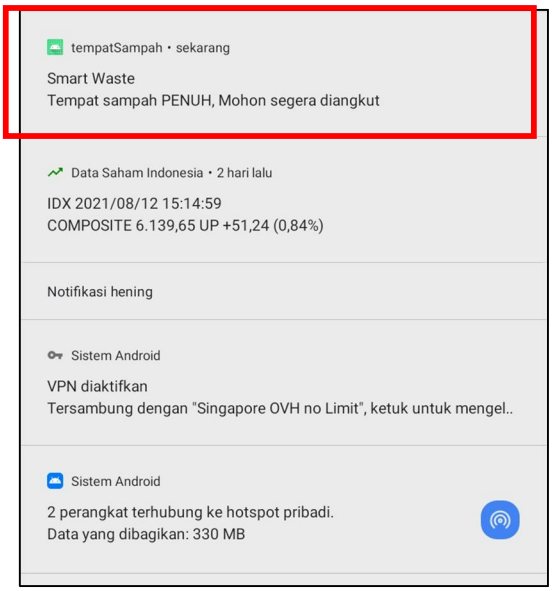
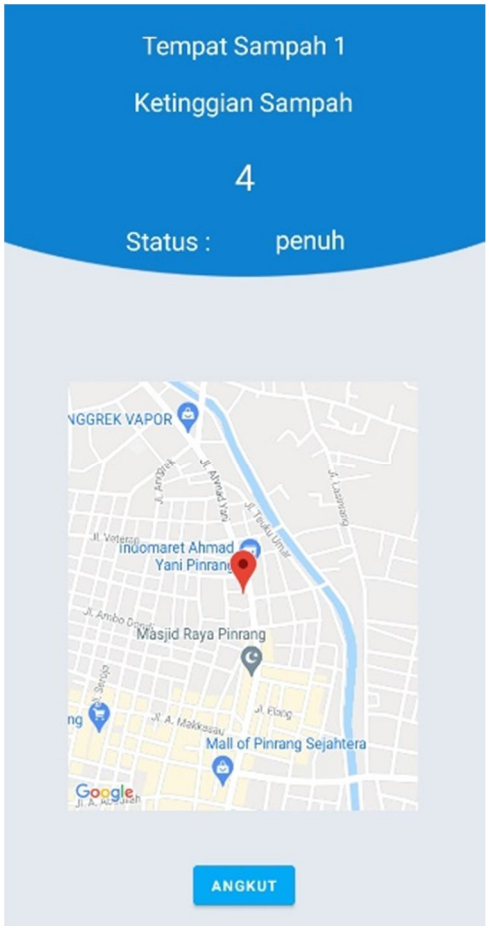
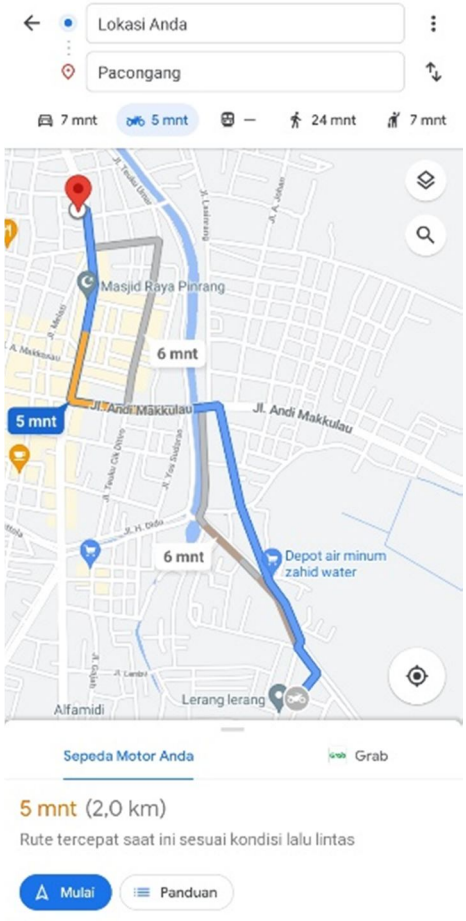
Test factor	Hasil	Keterangan
Pengujian *Login* dengan mengiput email dan *password* yang salah	✓	Pengujian berhasil dan menampilkan pesan "*Authentication faieled*"
Screenshot Aplikasi		
		

Table 4.4 Pengujian form notifikasi

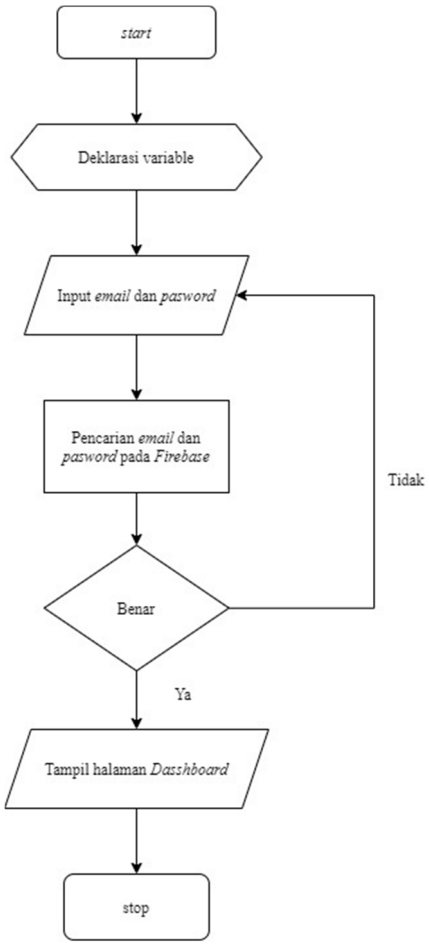
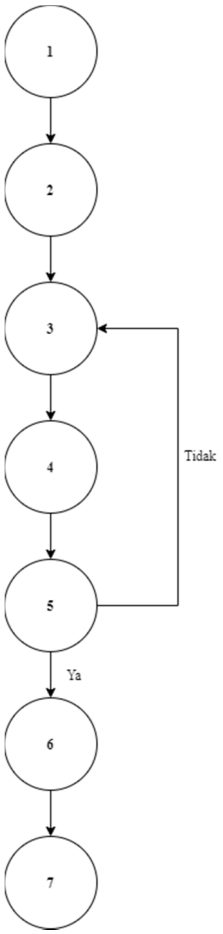
Test factor	Hasil	Keterangan
Pengujian push notifikasi ke aplikasi petugas untuk melakukan pengangkutan ke tempat sampah yang penuh	✓	Pengujian berhasil
Screenshot Aplikasi		
 		

Tabel 4.5 Pengujian Form angkut sampah

Test factor	Hasil	Keterangan
Pengujian form angkut sampah, petugas menekan tombol angkut dan akan menampilkan rute sesuai dengan lokasi tempat sampah	✓	Pengujian berhasil
Screenshot Aplikasi		
		

2. Pengujian *White Box*

a. Halaman *login*

Flowchart	*Flowgraph*
	

Gambar 4.10 *Flowchart* dan *Flowgraph* halaman *Login*

Dari gambar *flowgraph* diatas dapat dilakukan proses perhitungan sebaga berikut:

Menghitung *Cyclomatic Complexcity* V(G) dari *edge* dan *Node* dengan rumus :

V(G) = E – N + 2

N (*node*) =7

E (*edge*) = 7

P (*predikat node*) = 1

Penyelesaian : V(G) = E – N + 2

= 7 – 7 + 2

=2

Predikat = P + 1

= 1 + 1

= 2

Berdasarkan perhitungan *Cyclomatic Complexcity* dan *Flowgraph* diatas adalah :

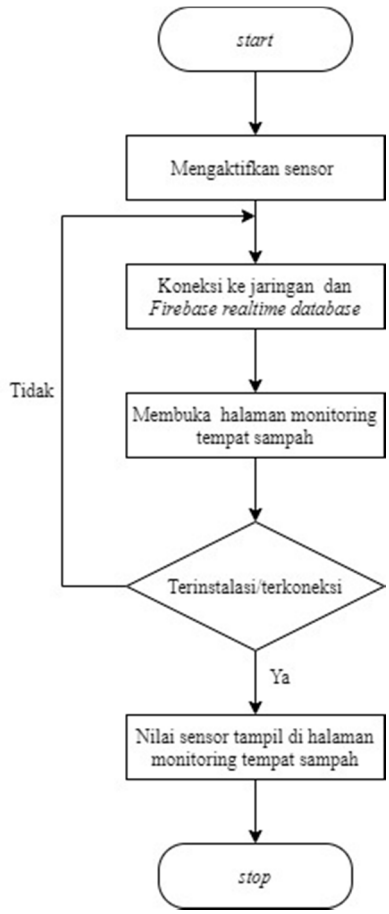
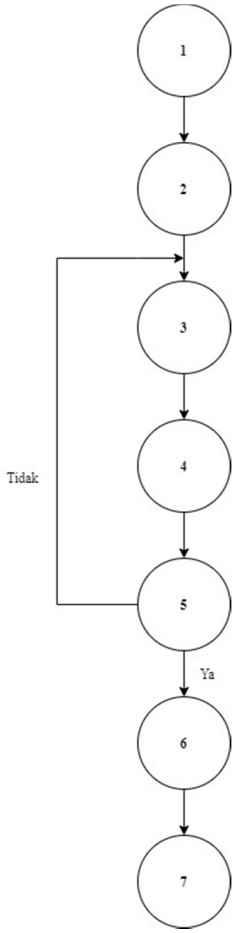
Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 7

Path 2 = 1 – 2 – 3 – 4 – 5 – 3

Grafik matriks aplikasi

No	**1**	**2**	**3**	**4**	**5**	**6**		**E – 1**
1		1						1 - 1=0
2			1					1 - 1=0
3				1				1 - 1=0
4					1			1 - 1=0
5			1			1		2 - 1=1
6							1	1 - 1=0
7								
SUM (E + 1)								2

b. Halaman Monitoring Tempat Sampah

Flowchart	*Flowgraph*
	

Gambar 4.11 *Flowchart* dan *Flowgraph* halaman Monitoring

Dari Gambar *FlowGraph* di atas dapat di lakukan proses perhitungan sebagai berikut :

Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node d*engan Rumus :

V (G) = E – N + 2

N(node) = 7

E(edge) = 7

P (predikat node) = 1

Penyelesaian : V(G) = E – N + 2

= 7 - 7 + 2

= 2

Predikat Node (N) = P + 1

= 1 + 1

= 2

Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

Independent Path pada *flowgraph* diatas adalah :

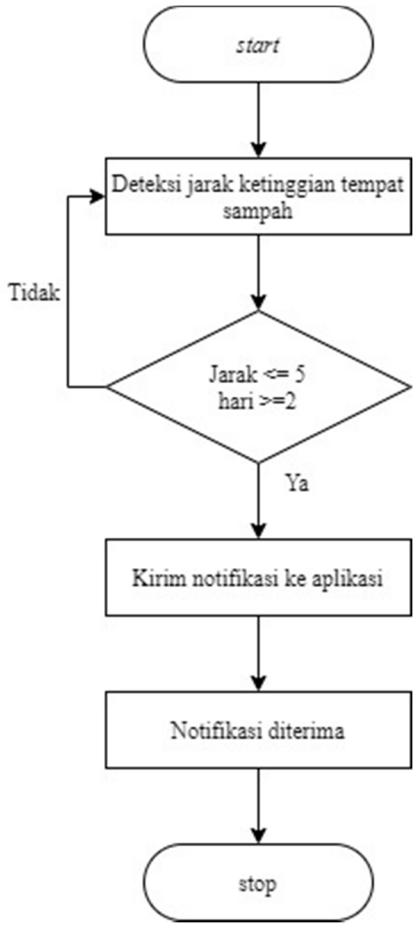
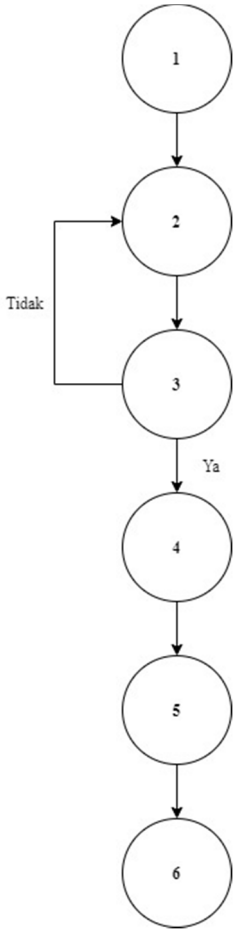
Path 1 = 1 – 2 – 3 – 4 – 5 – 6 – 7

Path 2 = 1 – 2 – 3 – 4 – 5 – 3 – 4 – 5 – 6 – 7

Grafik Matriks

	1	2	3	4	5	6	7	E – 1
1		1						1 – 1 = 0
2			1					1 – 1 = 0
3				1				1 – 1 = 0
4					1			1 – 1 = 0
5			1			1		2 – 1 = 1
6							1	1 – 1 = 0
7								0
Zum (E + 1)								1 + 1 = 2

c. Halaman Notifikasi

Flowchart	*Flowgraph*
	

Gambar 4.12 *Flowchart* dan *Flowgraph* halaman Notifikasi

Dari Gambar *FlowGraph* di atas dapat di lakukan proses perhitungan sebagai berikut :

Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node d*engan Rumus :

V (G) = E – N + 2

N(node) = 6

E(edge) = 6

P (predikat node) = 1

Penyelesaian : V(G) = E – N + 2

= 6 - 6 + 2

= 2

Predikat Node (N) = P + 1

= 1 + 1

= 2

Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

Independent Path pada *flowgraph* diatas adalah :

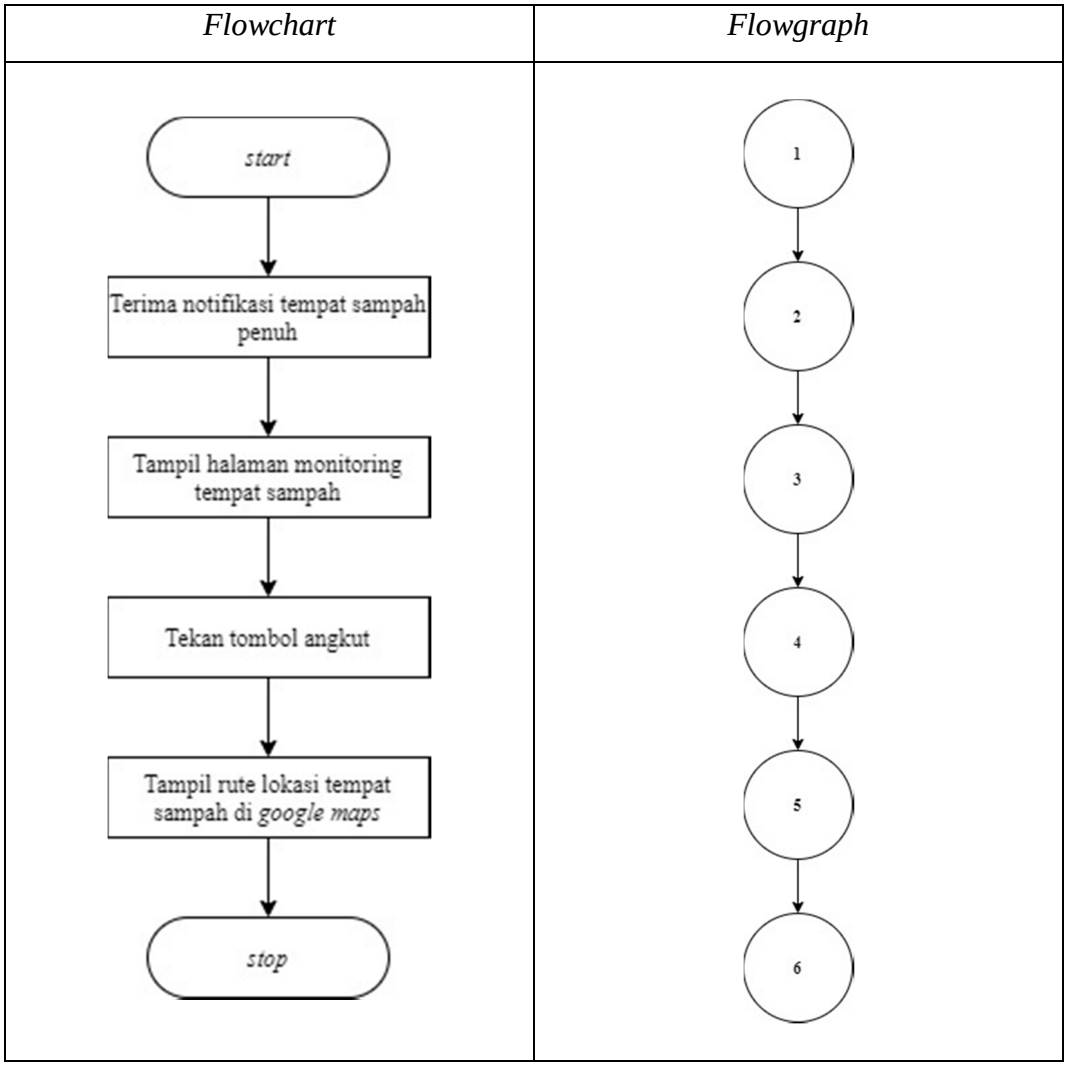
Path 1 = 1 – 2 – 3 – 4 – 5 – 6

Path 2 = 1 – 2 – 3 – 2 – 3 – 4 – 5 – 6

Grafik Matriks

	1	2	3	4	5	6	E – 1
1		1					1 – 1 = 0
2			1				1 – 1 = 0
3		1		1			2 – 1 = 1
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
Zum (E + 1)							1 + 1 = 2

d. *Flowchart* dan *Flowgraph* Angkut



Gambar 4.13 *Flowchart* dan *Flowgraph* halaman Angkut

Dari Gambar *FlowGraph* di atas dapat di lakukan proses perhitungan sebagai berikut :

Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node d*engan Rumus :

V (G) = E – N + 2

N(node) = 7

E(edge) = 6

P (predikat node) = 1

Penyelesaian : V(G) = E – N + 2

= 6 - 7 + 2

= 1

Predikat Node (N) = P + 1

= 0 + 1

= 1

Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

Independent Path pada *flowgraph* diatas adalah :

Path 1 = 1 – 2 – 3 – 4 – 5 – 6

Grafik Matriks

	1	2	3	4	5	6	E – 1
1		1					1 – 1 = 0
2			1				1 – 1 = 0
3				1			1 – 1 = 0
4					1		1 – 1 = 0
5						1	1 – 1 = 0
6							0
Zum (E + 1)							0 + 1 = 1

D. Pengujian Delay Alat ke Aplikasi

Pengujian delay ini dilakukan untuk mengetahui respon alat ke aplikasi dalam pengiriman data. Pengujian ini melakukan pengukuran dari alat tempat sampah ke aplikasi.

Tabel 4.6 Pengujian delay alat ke aplikasi lokasi Kec. Soreang

NO	KAPASITAS		DELAY	KET
	TINGGI SAMPAH	APLIKASI		
1	35 cm	35	4.24 detik	Baik
2	31 cm	31	3.72 detik	Baik
3	27 cm	27	3.85 detik	Baik
4	20 cm	20	2.40 detik	Baik
5	17 cm	17	2.74 detik	Baik
Rata – rata			3.30 detik	Baik

Tabel 4.7 Pengujian delay alat ke aplikasi lokasi Kec. Bacukiki

NO	KAPASITAS		DELAY	KET
	TINGGI SAMPAH	APLIKASI		
1	45 cm	35	3.24 detik	Baik
2	31 cm	31	3.42 detik	Baik
3	25 cm	25	3.53 detik	Baik
4	19 cm	19	2.87 detik	Baik
5	14 cm	14	3.74 detik	Baik
Rata – rata			3.36 detik	Baik

Tabel 4.8 Pengujian delay alat ke aplikasi lokasi Kec. Ujung

NO	KAPASITAS		DELAY	KET
	TINGGI SAMPAH	APLIKASI		
1	52 cm	52	3.76 detik	Baik
2	43 cm	43	3.97 detik	Baik
3	36 cm	36	2.89 detik	Baik
4	24 cm	24	2.43 detik	Baik
5	15 cm	15	3.62 detik	Baik
Rata – rata			3.34 detik	Baik

Tabel 4.9 Pengujian delay alat ke aplikasi lokasi Kec. Bacukiki barat

NO	KAPASITAS		DELAY	KET
	TINGGI SAMPAH	APLIKASI		
1	47 cm	47	4.15 detik	Baik
2	36 cm	36	4.22 detik	Baik
3	24 cm	24	3.85 detik	Baik
4	15 cm	15	3.70 detik	Baik
5	9 cm	9	4.32 detik	Baik
Rata – rata			4.05 detik	Baik

Pengujian dilakukan dengan cara memasukkan sampah ke tempat sampah. Data dari tempat sampah akan dikirim ke firebase dan diterima aplikasi, kemudian didapatkan hasil delay. Delay yang besar diakibatkan adanya proses dari sisi alat ke firebase dan dari firebase ke aplikasi yang di pengaruhi oleh jaringan yang digunakan. Dari hasil tersebut diperoleh rata-rata delay pengiriman data ke firebase dari setiap kecematan adalah 3.30 detik untuk Kec. Soreang, 3.36 detik untuk Kec. Bacukiki, 3.34 detik untuk Kec. Ujung dan 4.05 detik untuk Kec. Bacukiki barat.

Dari beberapa tahap pengujian yang dilakukan, semua berjalan dengan baik dan sesuai harapan. Proses pembuatan aplikasi menggunakan metode *prototype* dengan tahapan pengumpulan kebutuhan, desain cepat, membuat *prototype,* dan evaluasi. Pengujian pada sisi alat dan fitur aplikasi berfungsi dengan baik. Dari hasil evaluasi *prototype* aplikasi masih terdapat beberapa kekurangan dalam pembuatan aplikasi di antanya aplikasi belum dapat menerima multi notifikasi dikarenakan alat yang di buat hanya satu dan perlu beberapa alat serupa untuk dapat mengirim multi notifikasi. Pengunaan lokasi pada tempat sampah masih berdasarkan lokasi koordinat yang di input ke aplikasi belum mengunakan modul GPS, jadi update lokasi tempat sampah belum bisa otomatis jika tempat sampah berpindah lokasi.

BAB V

PENUTUP

A. Kesimpulan

Sesuai dengan dengan latar belakang dan tujuan yang telah dikemukakan sebelumnya maka dapat disimpulkan bahwa dalam membangun Aplikasi *mobile* monitoring pengelolaan sampah Kota Parepare menggunakan metode *prototype* dan melalui beberapa tahap diantarannya mengumpulkan data, perancangan aplikasi yang akan dibuat, proses pembuatan aplikasi, pengujian aplikasi dan implementasi apliksi. Aplikasi *mobile* monitoring pengelolaan sampah dibuat berbasis *android* menggunakan perangkat lunak *Android Studio* dan *Firebase* dengan bahasa pemrograman *Java*. Metode pengujian aplikasi menggunakan metode *Black Box* dan *White Box*.

Berdasarkan hasil penelitian yang telah dilakukan, dapat disimpulakan beberapa hal sebagai berikut :

1. Berdasarkan tahap-tahap pengujian yang sudah dilakukan, fitur pada aplikasi sesuia dengan harapan dan berjalan dengan baik.
2. Hasil pengujian fungsional aplikasi dan alat sesuai dengan yang diharapkan dan berhasil.

3. Berdasarkan data pengujian delay yang telah dilakukan, didapatkan nilai rata-rata delay sebesar 3.30 detik untuk Kec. Soreang, 3.36 detik untuk Kec. Bacukiki, 3.34 detik untuk Kec. Ujung dan 4.05 detik untuk Kec. Bacukiki barat.

B. Saran

Berdasarkan kesimpulan yang diuraikan diatas maka penulis dapat memberikan beberapa saran yang dapat membantu mengatasi beberapa kekurangan yang ada, diantaranya:

1. Aplikasi ini dapat dikembangkan lebih lanjut lagi dengan menambahkan pelaporan pengangkutan oleh petugas.
2. Pada sisi alat dapat ditambahkan berupa penanda di tempat sampah penuh/tidak, dan juga module GPS sebagai penanda lokasi tempat sampah.
3. Interface dari aplikasi ini masih sangat sederhana, sehingga dapat di kembangkan lebih menarik lagi.

DAFTAR PUSTAKA

Akhmad aludi. 2017. “Rancang Bangun Pemetaan Unit Dan Penjadwalan Pick Up Pada Bank Sampah Malang (BSM) Berbasis Android Dan Web Responsif”. Universitas Muhammadiyah Malang.

Arifianto, T. Membuat Interface Aplikasi Android Lebih Keren Dengan Lwuit.Yogyakarta: Andi Publiser, 2011.

Binus University, 2020. Teknik Dalam White-box dan Black-box Testing https://socs.binus.ac.id/2020/07/02/teknik-dalam-white-box-dan-black-box-testing/ 19 november 2020.

Dosen Pendidikan, 2020. Simbol Flowchart https://www.dosenpendidikan.co.id/simbol-flowchart/ 15 oktober 2020.

Dicoding Blog, 2020. Apa itu *Firebase* https://www.dicoding.com/blog/apa-itu-*Firebase*-pengertian-jenis-jenis-dan-fungsi-kegunaannya/ 15 januari 2021

F.T. Kusuma, T.N. Damayanti, D.N. Ramadan . (2017). *Aplikasi Monitoring Tempat Sampah Pintar Berbasis IoT.*

Faizah. 2008. “Pengelolaan Sampah rumah Tangga Berbasis Masyarakat (Studi Kasus di Kota Yogyakarta)”. Pasca sarjana Universitas Diponegoro Semarang Program Magister Ilmun Lingkungan.

Putri Hardianti (2019), “SIistem Informasi layanan Sampah Terjadwal Berbasis Web”. Universitas Muhammadiyah Parepare prodi Teknik Informatika.

Purwanto, R. (2018). Kesehatan Kamar Indekos Mahasiswa Berbasis Internet Of Things Design And Implementation Of Web Interface On Health Detection. 1-8.

Rio Jumardi. 2018. “Purwarupa Aplikasi Mobile Monitoring Laporan Pengaduan Sampah”. STITEK Bontang prodi Teknik Informatika.

Safaat, Nazruddin. Aplikasi Berbasis Android : Berbagai Implementasi dan Pengembangan Aplikasi Mobile Berbasis Android. Cet. I; Bandung: Informatika Bandung, 2015.

Saturadar. 2019. Aplikasi. *https://www.saturadar.com/2019/09/Pengertian-Aplikasi*15 Agustus 2020.

Unikom, 2020.Flowmap Diagram. https://elib.unikom.ac.id/files/disk1/442/jbptunikompp-gdl-hambalinim-

22097-12-daftars-l.pdf/ 15 Oktober 2020.

Siti Sauda, eka Puji Agustini. (2020). Implementasi Prototype Model Pengembangan Aplikasi Smart Cleaning Sebagai Pendukung Aplikasi Smart City. *Jurnal Manajemen, teknik Informatika & Rekayasa Komputer Vol.20 No. 1 Nov 2020*, 73-84.

Syaeful Anwar, Agus Ganda Permana, Denny Darlis. (2020). Rancang Bangun dan Implementasi Aplikasi Informasi Pengelolaan Sampah & Tempat sampah Setempat Terpadu Mandiri Dengan Kontrol Terpusat Berbasi IoT. *e-Proceeding of Applied Science Vol.1 No. 1 April 2020*, 660-673.

Ubaildillah, D. (2015). *Perancangan Aplikasi Smart Tras can Menggunakan Arduino dengan Sensor Ultrasonic.*

Wiwik Kusrini, Herpendi, Muhammad Noor. (2019). Rancang Bangun Aplikasi Informasi Antar Jemput Sampah Rumah Tangga. *Simetrik Vol. 9 No.1 Juni 2019*, 1-7.