BAB I

PENDAHULUAN

A. Latar Belakang

Banyak masalah yang muncul berkaitan dengan sampah yang mengganggu kesehatan dan kebersihan lingkungan. Rendahnya kesadaran masyarakat dalam membuang sampah yang benar ada kaitannya dengan keadaan tempat sampah. Tempat sampah dalam keadaan bersih, unik dengan sentuhan teknologi modern akan membuat orang tertarik untuk membuang sampah dengan benar, sebaliknya tempat sampah dengan kondisi yang buruk menyebabkan orang malas membuang sampah.

Rekayasa dengan penerapan sistem kontrol memiliki peranan yang sangat penting khususnya bagi pengembangan tempat sampah yang ada pada saat ini agar tempat sampah lebih modern dan lebih efisien dalam memberitahukan kepada petugas kebersihan. perkembangan teknologi juga menyentuh aspek kepedulian terhadap lingkungan dengan adanya tempat sampah cerdas atau biasa disebut dengan tempat sampah pintar, tempat sampah yang dulunya mempunyai fungsi tunggal dengan menampung sampah saja kini telah dipadukan dengan teknologi dengan menambahkan fungsi tertentu, baik berupa otomatisasi buka tutup pada tempat sampah sampai pengiriman informasi keadaan tempat sampah kepada petugas kebersihan sehingga lebih efisien karena tidak perlu menunggu laporan secara manual dari warga sekitar.

Kebersihan merupakan salah satu faktor berlangsung dan terciptanya hidup yang bersih dan sehat, di tempat umum seperti kampus, kantor, bandara

sering terjadinya penumpukan sampah di tempat sampah karena terlalu banyak sampah yang dihasilkan. Di kota Parepare sendiri kondisi seperti itu juga terjadi dimana Sampah menumpuk tersebut mengganggu baik dari segi kebersihan maupun bau dari sampah yang mengganggu kenyamanan masyarakat. Hal ini disebabkan oleh beberapa faktor seperti petugas kebersihan yang tidak mengetahui kondisi tempat pembuangan sementara yang ada,apakah masih kosong atau sudah penuh dan faktor lain yaitu pengguna tempat sampah yang memaksa untuk membuang sampah pada tempat sampah yang penuh. Seperti yang terjadi di TPS yang ada di daerah Tegal, Kel. Lapadde, Kec. Ujung, Kota Parepare. Pada saat terjadi penumpukan sampah banyak masyarakat yang membuang sampah di sekitar TPS sehingga mengganggu kenyamanan dan juga kebersihan.

Dari permasalahan di atas maka diperlukan tempat sampah yang telah terintegrasi dengan sensor *Ultrasonic* sebagai pendeteksi yang berfungsi untuk mengetahui ketinggian sampah dan*Google Maps API S*sebagai penunjuk lokasi, dan dalam penelitian ini akan dibangun alat yaitu "*Prototype* Sistem *Monitoring* Tempat Sampah berbasis *IoT*" yang menggunakan power bank tenga surya sebagai sumber energi dari alat tersebut, dengan adanya alat ini diharapkan dapat mengurangi penumpukan sampah yang sehingga membantu dalam hal menjaga kebersihan lingkungan sekitar.

B. Rumusan Masalah

berdasarkan uraian latar belakang di atas, maka masalah yang akan dibahas antara lain, yaitu:

1. Bagaimana membuat alat untuk *monitoring* ketinggian sampah pada tempat sampah?
2. Bagaimana membuat alat untuk *monitoring* lokasi tempat sampah?

C. Tujuan Penelitian

Sesuai dengan permasalahan yang telah diuraikan di atas, maka tujuan dari penelitian ini yaitu:

1. Terbentuknya sistem identifikasi dan *monitoring* ketinggiansampah pada tempat sampah.
2. Terbentuknya sistem *monitoring* lokasi tempatsampah.

D. Batasan Masalah

Pada penelitian ini memiliki beberapa Batasan masalah dengan harapan penelitian ini nantinya akan lebih terfokus pada Batasan-batasan yang dibuat. Berikut merupakan Batasan masalah dalam penelitian ini:

1. Sistem yang dirancang hanya untuk pengidentifikasian dan monitoring ketinggian sampah di dalam tempat sampah.
2. Sistem yang dibuat tidak mendeteksi dan membedakan jenis sampah yang ada pada tempat sampah.
3. Sistem yang dibuat tidak melakukan pemilahan sampah.

E. Manfaat Penelitian

Dalam penelitian ini ada manfaat yang diharapkan bagi pembuat sistem dan juga bagi pengguna. Adapun manfaat yang diharapkan dalam pembuatan sistem ini yaitu:

1. Manfaat bagi pembuat sistem

Menambah pengetahuan mengenai *Internet of Things* (*IOT*) dan mampu mengimplementasikan ilmu pengetahuan yang di dapat dari bangku perkuliahan khususnya pada jurusan Teknik Informatika.

2. Manfaat *Pengguna*

Dapat memonitorketinggian sampah dan lokasi tempat sampah

F. Sistematika Penulisan

Adapun sistematika penulisan pada Tugas Akhir ini yaitu:

BAB I PENDAHULUAN

Berisi tentang uraian latar belakang pemilihan judul *Prototype* sistem *monitoring* tempat sampah berbasis *IOT*, rumusan masalah, batasan masalah, tujuan penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Dalam bab ini berisikan beberapa teori-teori yang mendukung dalam pembahasan penyusunan tugas akhir ini serta bahasa pemrograman yang digunakan sehingga memudahkan penulis dalam menyelesaikan masalah.

BAB III METODE PENELITIAN

Pada bab ini menjelaskan mengenai tahapan - tahapan yang dilalui dalam penyelesaian tugas akhir ini, yaitu tempat penelitian dan waktu penelitian, metode pengumpulan data, alat dan bahan penelitian, tahap penelitian, metode pengujian serta gambaran desain sistem yang akan dirancang atau dibuat.

BABIV HASIL DAN PEMBAHASAN

Pada bab ini menjelaskan mengenai hasil dari penelitian yaitu rancangan sistem dan juga pengujian dari sistem yang telah dibuat.

BABV PENUTUP

Pada bab ini menjelaskan mengenai kesimpulan dan saran.

BAB II

TINJAUAN PUSTAKA

A. Kajian Teori

1. Pengertian Sampah

Menurut Undang-Undang Nomor 18 Tahun 2018 sampah adalah sisa kegiatan sehari-hari manusia dan/atau proses alam yang berbentuk padat.

Menurut definisi *World Health Organization (WHO)* sampah adalah sesuatu yang tidak digunakan, tidak dipakai, tidak disenangi atau sesuatu yang dibuang yang berasal dari kegiatan manusia dan tidak terjadi dengan sendirinya (Candra, 2006)

Sampah merupakan bahan padat buangan dari kegiatan rumah tangga, perkantoran, rumah penginapan, hotel, rumah makan, industri, puing bahan bangunan, dan besi-besi tua bekas kendaraan bermotor. Sampah merupakan hasil samping yang sudah tidak terpakai (Sucipto, 2012).

Gambar 2.1 Sampah

2. Internet of Things

Internet of Things, atau dikenal juga dengan singkatan IoT, merupakan sebuah konsep yang bertujuan untuk memperluas manfaat dari konektivitas internet yang tersambung secara terus-menerus. Adapun kemampuan seperti berbagi data, *remote control*, dan sebagainya, termasuk juga pada benda di dunia nyata. Contohnya bahan pangan, elektronik, koleksi, peralatan apa saja, termasuk benda hidup yang semuanya tersambung ke jaringan lokal dan global melalui sensor yang tertanam dan selalu aktif.

Pada dasarnya, *Internet of Things* mengacu pada benda yang dapatdiidentifikasikan secara unik sebagai representasi virtual dalam strukturberbasis *Internet*. Istilah *Internet of Things* awalnya disarankan oleh KevinAshton pada tahun 1999 dan mulai terkenal melalui Auto-ID Center di MIT.Dan kini IoT menjadi salah satu tugas bagi seorang mahasiswa di sebuah perguruan tinggi.

3. Sistem *Monitoring*

Sistem adalah sekumpulan elemen yang terintegrasi dengan maksud yang sama untuk mencapai suatu tujuan dan merupakan sekumpulan komponen yang saling bekerja sama untuk mencapai tujuan guna memperbaiki organisasi ke arah yang lebih baik.

Banyak definisi dalam menjelaskan pengertian sistem. Namun demikian, pada umumnya definisi itu menggambarkan bahwa pengertian sistem mengandung dua konotasi, yaitu benda atau entitas, dan proses atau metode. Schrode dan Voich (1974) dalam bukunya yang berjudul

organization and management: basic systems concept misalnya, menyatakan bahwa sistem adalah "*whole compounded of several parts*" (suatu kesatuan yang tersusun dari sejumlah elemen.

Monitoring adalah proses pengumpulan informasi mengenai apa yang sebenarnya terjadi selama proses implementasi atau penerapan program. *Monitoring* adalah proses rutin pengumpulan data dan pengukuran kemajuan atas objek program/memantau perubahan yang fokus pada proses dan keluaran.

Menurut Edi Suharto *monitoring* pada dasarnya adalah pemantauan suatu kegiatan proyek atau program sosial yang dilaksanakan pada saat kegiatan tersebut sedang berlangsung.

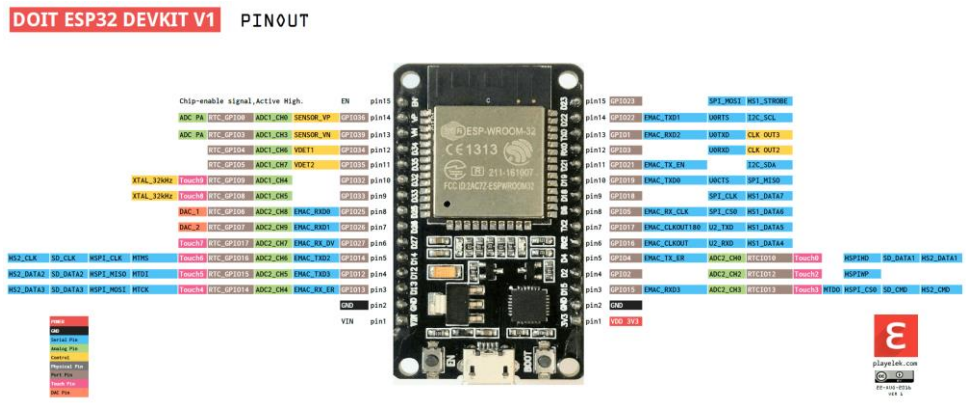
Menurut Nalahudin Muhlisin *monitoring* merupakan suatu proses pengumpulan dan menganalisis informasi dari penerapan suatu program termaksud mengecek secara reguler untuk melihat apakah kegiatan atau program itu berjalan sesuai dengan rencana sehingga masalah yang dilihat atau ditemui dapat diatasi. *Monitoring* menyediakan data dasar untuk menjawab permasalahan.

Dari beberapa pendapat di atas dapat dipahami pengertian *monitoring* adalah pemantauan suatu program yang dilaksanakan saat kegiatan tersebut sedang berlangsung, *monitoring* menyediakan data dasar untuk menjawab permasalahan.

Sedangkan yang dimaksud dengan *systemmonitoring* adalah layanan yang melakukan proses pengumpulan data dan melakukan analisis terhadap

data-data tersebut dengan tujuan untuk memaksimalkan seluruh sumber daya yang dimiliki.

4. ESP-32



Gambar 2.2*ESP-32*
Sumber: (https://www.bukalapak.com/p/elektronik/elektronik-lainnya/nn1dct-jual-esp32-nodemcu-dual-core-wifi-bluetooth-esp-32-harga-murah)

ESP32 adalah *mikrokontroler* yang dikenalkan oleh *Espressif System* merupakan penerus dari *mikrokontrolerESP8266*. Pada *mikrokontroler* ini sudah tersedia modul *WiFi* dalam *chip* sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. terlihat pada gambar di atas merupakan pin out dari ESP32. Pin tersebut dapat dijadikan input atau output untuk menyalakan LCD, lampu, bahkan untuk menggerakkan motor DC.

Berikut merupakan tabel spesifikasi dari modul ESP-32:

Tabel 2.1 Spesifikasi *ESP-32*

No	Nama Spesifikasi	Keterangan
1	Module model	*ESP32*
2	Package	DIP-16

No	Nama Spesifikasi	Keterangan
3	Size	27*40.5*4.5(±0.2)mm
4	SPI Flash	default 32Mbit
5	RAM	internal520KB+external 4M PSRAM
6	*Bluetooth*	bluetooth4.2BR/EDR and BLE standards
7	Wi-Fi	802.11 b/g/n/e/i
8	Support interface	UART, SPI, I2C, PWM
9	Support TF card	Maximum support 4G
10	IO port	9
11	Serial port rate	default 115200 bps
12	Image output format	JPEG (only supported by OV2640), BMP, GRAYSCALE
13	Spectrum range	2412 ~ 2484MHz
14	Antenna form	onboard PCB antenna , gain 2dBi
15	Transmit power	802.11b: 17±2 dBm (@11Mbps) 802.11g: 14±2 dBm (@54Mbps) 802.11n: 13±2 dBm (@MCS7)
16	Receiving sensitivity	CCK, 1 Mbps : -90dBm CCK, 11 Mbps: -85dBm 6 Mbps (1/2 BPSK): -88dBm 54 Mbps (3/4 64-QAM): -70dBm MCS7 (65 Mbps, 72.2 Mbps): -67dBm
17	Power consumption	Turn off the flash: 180mA@5V Turn on the flash and adjust the brightness to the maximum: 310mA@5V Deep-sleep: The lowest power consumption can reach 6mA@5V Moderm-sleep: up to 20mA@5V Light-sleep: up to 6.7mA@5V
18	Security	WPA/WPA2/WPA2-Enterprise/WPS
19	Power supply range	5V

No	Nama Spesifikasi	Keterangan
20	Operating temperature	-20 °C ~ 85 °C
21	Storage environment	-40 °C ~ 90 °C, < 90%RH

5. Sensor Ultrasonic *HC-SR04*

Gambar 2.3 Sensor *Ultrasonic HC-SR04*
Sumber: (https://www.andalanelektro.id/2018/09/cara-kerja-dan-karakteristik-sensor-ultrasonic-hcsr04.html)

HC-SR04 adalah sebuah modul sensor *Ultrasonic* yang biasanya digunakan untuk alat pengukur jarak. Pada *HC-SR04* terdapat sepasang *transducerUltrasonic* yang satu berfungsi sebagai *transmitter* yang bertugas untuk mengubah sinyal elektrik menjadi sinyal pulsa gelombang suara *Ultrasonic* dengan frekuensi 40KHz, dan satunya berfungsi sebagai *receiver* yang bertugas untuk menerima sinyal gelombang suara *Ultrasonic*.Sensor *HC-SR04* memiliki spesifikasi sebagai berikut:

Tabel 2.2 Spesifikasi sensor *Ultrasonic HC-SR04*

No	Nama Spesifikasi	Keterangan
1	Tegangan Operasi	5V DC
2	Arus Operasi	15Ma
3	Jangkauan Maksimal	4m

NO	Nama Spesifikasi	Keterangan
4	Jangkauan Minimal	2cm
5	Mengukur Sudut	15 Derajat
6	*input trigger signal*	*10us min. TTL pulse*
7	*output echo signal*	*TTL level signal, proportional to distance*
8	*Board Dimensions*	1-13/16" X 13/16" X 5/8"
9	*Board Connections*	4 X 0.1" *Pitch Right Angle Header Pins*

6. Google Maps API

Google Maps API merupakan serangkaian data yang adalah kumpulan library JavaScript. Untuk menggunakan fitur atau memprogram Maps Javascript API tersebut cukup mudah. Yang kita butuhkan adalah pengetahuan seputar HTML dan JavaScript, lalu koneksi Internet. Dengan menggunakan Google Maps API sendiri kita dapat menghemat waktu serta biaya. Bayangkan jika kita dapat mempergunakan atau mengaplikasikan peta digital yang handal, sehingga kita bisa focus membangun pada data-data yang diperlukan untuk kepentingan bisnis atau khalayak banyak. Google telah merangkum semua data peta-peta tersebut dengan jangkauan yang sangat luas hingga 92% di Dunia.

Google juga menyediakan Google Maps API yang memungkinkan kita membangun aplikasi dengan memanfaatkan fitur Google Maps.Google Maps API merupakan sebuah API yang disediakan oleh Google untuk menggunakan peta Google (Google Map) dalam aplikasi yang kita bangun.Ada banyak map editor yang dapat digunakan untuk membuat sebuah aplikasi informasi geografis. Salah satu aplikasi informasi

geografis yang cukup popular 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).

7. Power Bank Tenaga Surya

Gambar 2.4 Power Bank Tenaga Surya
Sumber: https://cf.shopee.co.id/file/f805379a27e1fb0e7d2f4176b425b755

SIP adalah singkatan dari Solar Integrated Powerbank yaitu suatu alat yang berfungsi untuk mengisi dayadengan memanfaatkan energi terbarukan yaitu energi surya (matahari). *Solar cell*yang berfungsi untuk mengkonversi energy matahari menjadi energy listrik.

Tabel 2.3 Spesifikasi power bank

No	Spesifikasi	Keterangan
1.	Kapasitas	30000mAh
2.	Input	DC 5 V - 2.1A(MAX)
3.	Output	DC 5V - 2.1A(MAX)
4.	Dimensi	14.5*6.8*1.9 cm

8. *Flutter*

Flutter merupakan sebuah SDK untuk pengembangan aplikasi *mobile* yang dikembangkan oleh Google untuk membangun aplikasi yang memiliki kinerja tinggi serta dapat dipublikasi ke platform Android dan IOS dari *codebase* tunggal.

Flutter dapat dengan mudah dipelajari karna menggunakan bahasa pemrograman *Dart* yang pastinya terasa familier jika sudah terbiasa menggunakan bahasa pemrograman *Java* atau *Javascript*. Selain itu Flutter juga menyertakan kerangka *reactive-functional*, mesin render 2D, *widget* siap pakai, dan *tools* untuk pengembangan.

Ada begitu banyak kerangka yang bisa digunakan untuk mengembangkan aplikasi lintas platform, seperti *React Native, Nativescript*, dan *Fuse*. Namun yang membedakannya adalah, Flutter tidak menggunakan *Webview* maupun *widget* bawaan, *Flutter* punya mesin *render* sendiri untuk menampilkan *widget*-nya, hal ini menguntungkan developer yang ingin memiliki tampilan UI unik yang konsisten pada semua perangkat karena tidak bergantung pada *widget* bawaan OEM.

Kinerja *Flutter* yang tinggi ini tentunya didukung oleh berbagai teknologi terbaik. *Flutter* dibuat dengan C, C++, Dart, Skia untuk mesin render 2D, Mojo IPC, dan Blink untuk sistem render.

Cara kerja *Flutter* pada platform *Android* yaitu, kode C/C++ dikompilasi menggunakan *Android SDK*, sebagian besar kerangka dan kode aplikasi dijalankan dalam bentuk kode *native* yang dikompilasi oleh Dart

compiler. Sedangkan pada platform iOS, kode dikompilasi dengan LLVM dan aplikasi dijalankan dengan kumpulan instruksi *native* tanpa interpreter.

Gambar 2.5 Logo*Flutter*

Flutter merupakan sebuah SDK untuk pengembangan aplikasi *mobile* yang dikembangkan oleh Google untuk membangun aplikasi yang memiliki kinerja tinggi serta dapat dipublikasi ke platform Android dan IOS dari *codebase* tunggal.

Flutter dapat dengan mudah dipelajari karna menggunakan bahasa pemrograman *Dart* yang pastinya terasa familier jika sudah terbiasa menggunakan bahasa pemrograman *Java* atau *Javascript*. Selain itu Flutter juga menyertakan kerangka *reactive-functional*, mesin render 2D, *widget* siap pakai, dan *tools* untuk pengembangan.

Ada begitu banyak kerangka yang bisa digunakan untuk mengembangkan aplikasi lintas platform, seperti *React Native, Nativescript,* dan *Fuse*. Namun yang membedakannya adalah, Flutter tidak menggunakan *Webview* maupun *widget* bawaan, *Flutter* punya mesin *render* sendiri untuk menampilkan *widget*-nya, hal ini menguntungkan developer yang ingin memiliki tampilan UI unik yang konsisten pada semua perangkat karena tidak bergantung pada *widget* bawaan OEM.

Kinerja *Flutter* yang tinggi ini tentunya didukung oleh berbagai teknologi terbaik. *Flutter* dibuat dengan C, C++, Dart, Skia untuk mesin render 2D, Mojo IPC, dan Blink untuk sistem render.

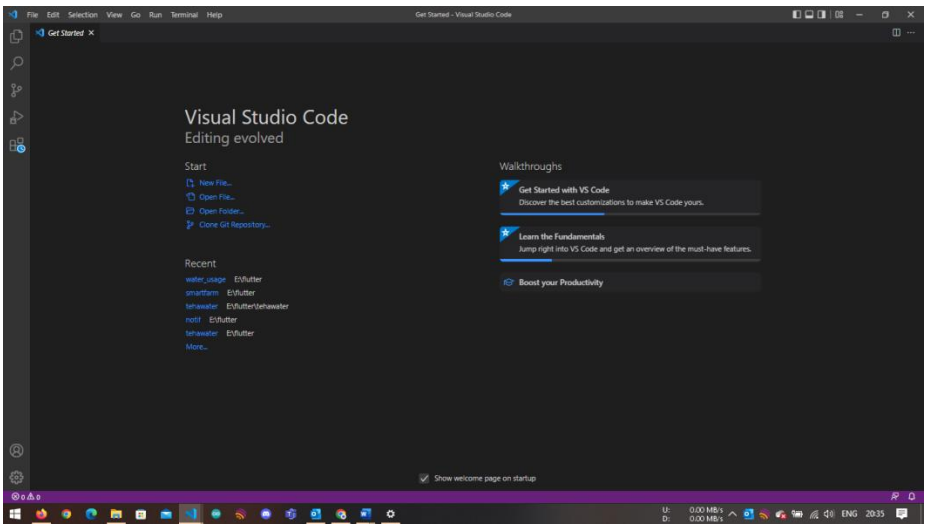
Cara kerja *Flutter* pada platform *Android* yaitu, kode C/C++ dikompilasi menggunakan *Android SDK*, sebagian besar kerangka dan kode aplikasi dijalankan dalam bentuk kode *native* yang dikompilasi oleh Dart *compiler*. Sedangkan pada platform iOS, kode dikompilasi dengan LLVM dan aplikasi dijalankan dengan kumpulan instruksi *native* tanpa interpreter.

9. Visual Studio Code

Gambar 2.6 Logo *Visual Studio Code*

Visual Studio Code adalah editor kode sumber yang dikembangkan oleh Microsoft untuk *Windows*, *Linux* dan *MacOS* . Visual Studio Code termasuk dukungan untuk *debugging* , kontrol *Git* yang tertanam dan *GitHub*, penyorotan sintaksis, penyelesaian kode cerdas, *snippet*, dan *refactoring* kode. Ini sangat dapat disesuaikan, memungkinkan pengguna untuk mengubah tema, pintasan *keyboard*, preferensi, dan menginstal ekstensi yang menambah fungsionalitas tambahan. Kode sumber adalah sumber bebas dan

terbuka dan dirilis di bawah Lisensi MIT yang permisif. Binari yang dikompilasi adalah *freeware* dan gratis untuk penggunaan pribadi atau komersial. *Visual Studio Code* didasarkan pada *Electron* , sebuah kerangka kerja yang digunakan untuk menggunakan aplikasi *Node.js* untuk desktop yang berjalan pada mesin tata letak *Blink*. Meskipun menggunakan kerangka Elektron, perangkat lunak tidak menggunakan Atom dan sebagai gantinya 12 mempekerjakan komponen editor yang sama (nama kode "*Monaco*") yang digunakan dalam *Azure DevOps* (sebelumnya disebut *Visual Studio Online* dan Layanan Tim *Visual Studio*). Dalam Survei Pengembang *Stack Overflow* 2019, *Visual Studio Code* mendapat peringkat alat lingkungan pengembang paling populer, dengan 50,7% dari 87.317 responden mengklaim menggunakannya.



Gambar 2.7 Tampilan Awal *Visual Studio Code*

10. *Firebase*

Gambar 2.8 Logo *Firebase*

Firebase adalah penyedia layanan *realtime database* dan *backend* sebagai layanan. Suatu aplikasi yang memungkinkan pengembang membuat API untuk disinkronisasikan untuk clienit yang berbeda-beda dan disimpan pada *cloud-nya Firebase*. *Firebase* memiliki banyak *library* yang memungkinkan untuk mengintegrasikan layanan ini dengan Android, IOS, *Javacript, Java, Objective-C* dan *Node.JS*. *Database Firebase* juga bersifat bisa diakses lewat *REST API*. *REST API* tersebut menggunakan protokol *Server-SentEvent* dengan membuat koneksi *HTTP* untuk menerima *push notification* dari server. Pengembang menggunakan *REST API* untuk post data yang selanjutnya *Firebaseclient library* yang sudah diterapkan pada aplikasi yang dibangun yang akan mengambil data secara *realtime*.

Jenis-jenis atau fitur *Firebase*, antara lain:

a. *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*merek 'O' atau bahkan bisa mencari tahu jam berapa transaksi yang dilakukan *user* sering terjadi.

b. *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*

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 Google, Facebook, dan sebagainya.

Firebase Authentication terintegrasi dengan fitur layanan *Firebase* lainnya. Sistem 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*

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 *SDKreal* oleh aplikasi *iOS, Android* dan *web*.

e. *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 *SDKAndroid*, *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*

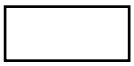
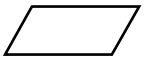
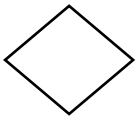
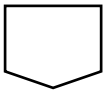
Selanjutnya ada *Firebase Hosting*, suatu layanan *hosting* konten web. Hanya dengan satu instruksi, kamu dapat mengimplementasikan aplikasi web serta menyajikan konten statis maupun dinamis ke CDN (jaringan penayangan konten) global dengan cepat.

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*.

11. *Flowchart*

"Flowchart atau bagan alir adalah representasi grafik dari sistem yang mendeskripsikan relasi fisik diantara entitas – entitas intinya. Bagan alir dapat digunakan untuk menyajikan aktivitas manual, aktivitas pemrosesan komputer, atau keduanya. Bagan alir dokumen (document flowchart) digunakan untuk menggambarkan elemen–elemen dari sistem manual, termasuk catatan akuntansi (dokumen, jurnal, buku besar, dan file), departemen organisasi yang terlibat dalam proses dan aktivitas (baik yang bersifat administratif maupun fisik) yang dilakukan Simbol – simbol yang digunakan dalam flowchart (Hasyim, 2021). " adalah sebagai berikut:

Tabel 2.4*Flowchart*

No.	Simbol	Nama	Fungsi
1		*Terminator*	Permulaan/akhir program
2		Garis Alir (*Flow Line*)	Arah aliran program
3		*Preparation*	Proses inisialisasi program / pemberian nilai awal
4		Proses	Proses perhitungan / proses pengolahan data
5		Input/Output Data	Proses input / output data, parameter, informasi
6		*Predefined process (sub program)*	Permulaan sub program/proses menjalankan sub program
7		*Decision*	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk data langkah selanjutnya
8		*On Page Connector*	Penghubung bagian-bagian *Flowchart* yang berada pada halaman berbeda
9		*Off Page Connector*	Penghubung bagian-bagian *Flowchart* yang pada halaman berbeda

12. UML (*Unified Modeling Language*)

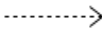
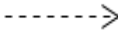
(Henderi, 2009)*UML (Unified Modeling Language)* adalah "sebuah teknik pengembangan sistem yang menggunakan bahasa grafis sebagai alat untuk pendekomentasian dan melakukan spesifikasi pada sistem". *Unified Modelling LanguageUML* adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. *UML* merupakan suatu kumpulan teknik terbaik yang telah terbukti sukses dalam memodelkan sistem yang besar dan kompleks.

(Kroenke, 2005) *UML* merupakan metodologi untuk mengembangkan sistem OOP dan sekelompok tool untuk mendukung pengembangan sistem tersebut. *UML* mulai diperkenalkan oleh *Object Management Group*, sebuah

organisasi yang telah mengembangkan model, teknologi, dan standar *OOP* sejak tahun 1980-an.

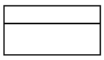
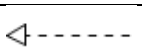
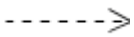
Seperti bahasa - bahasa lainnya, *UML* mendefinisikan notasi dan Syntax/semantik. Setiap bentuk memiliki makna tertentu, dan *UML Syntax* mendefinisikan bagaimana bentuk – bentuk tersebut dapat dikombinasikan. Notasi *UML* terutama diturunkan dari 3 notasi yang telah ada sebelumnya : *Grady Booch OOD (Object – Oriented Design), Jim Rumbaugh OMT (Object Modeling Technique)* dan Ivar Jacobson *OOSE (Object – Oriented Software Engineering).* Adapun daftar simbol *UML* yaitu :

Tabel 2.5*Symbol Use Case Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *Use Case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*Independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
3		*Generalization*	Hubungan dimana objek anak *(Descendent*) berbagai perilaku dan struktur data dari objek yang ada di atasnya objek induk (*Ancestor*).
4		*Include*	Menspesifikasikan bahwa Use Case sumber secara Eksplisit.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan

No	GAMBAR	NAMA	KETERANGAN
			prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

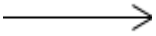
Tabel 2.6*Symbol Class Diagram*

No.	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

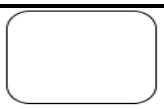
Tabel 2.7*Symbol Sequence Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
No	**GAMBAR**	**NAMA**	**KETERANGAN**
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

Tabel 2.8*Symbol State Chart Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Tabel 2.9*Symbol Activity Diagram*

No.	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

13. *White Box Testing*

Pengujian*white box* testing biasa disebut dengan *glass box*, adalah metode desain *test case* yang menggunakan struktur kontrol desain prosedural untuk memperoleh sebuah *test case* (B, 2006). Tujuan dari penggunaan pengujian *white box* yaitu menguji semua *statement* pada program. Metode pengujian *white box* dapat menjamin:

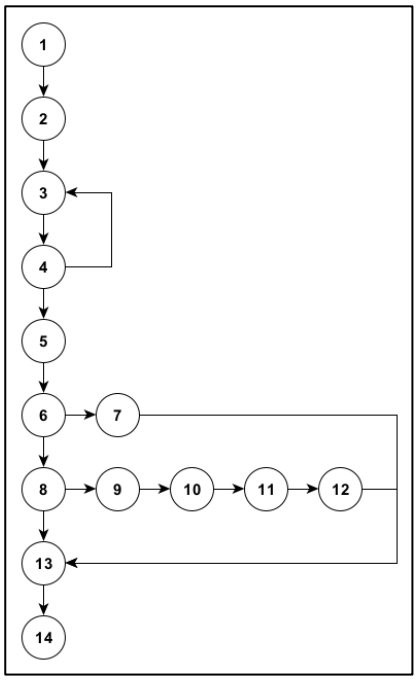
a. Semua jalur (*path*) yang independen / terpisah dapat di tes setidaknya satu kali tes.
b. Semua logika keputusan dapat dites dengan jalur yang salah dan atau jalur yang benar.
c. Semua *loop* dapat dites terhadap batasannya dan ikatan operasionalnya.
d. Semua struktur internal data dapat dites untuk memastikan validitasnya.

Metode pengujian *white box* sering kali diasosiasikan sebagai suatu teknik dalam pengukuran cakupan tes (*test coverage metrics*), yaitu sehubungan dengan menghitung persentase dari jalur-jalur yang dieksekusi berdasarkan *test case* yang dipilih (Jatnika & Irwan, 2010). Pada dasarnya sekarang ini, banyak para penguji perangkat lunak jarang melakukan pengujian dengan menggunakan *white boxtesting*, tetapi langsung melakukan pengujian pada tampilan antarmuka sistem. Namun demikian pengujian tidak bisa hanya dilakukan untuk menguji pada tampilan sistem, tetapi juga perlu dilakukan pengujian pada struktur dan kontrol logika pada kode program. Pengujian dengan metode *white boxtesting* dilakukan pada suatu perangkat lunak dengan tujuan untuk menemukan kesalahan-kesalahan yang mungkin tidak ditemukan ketika melakukan pengujian pada tampilan antarmuka sistem atau yang biasa disebut dengan metode *black box testing*.

Adapun penjelasan mengenai kesalahan-kesalahan yang mungkin dapat ditemukan seperti:

a. Kesalahan logika dan asumsi yang tidak benar kebanyakan dilakukan ketika *coding* untuk "kasus tertentu". Dibutuhkan kepastian bahwa eksekusi jalur ini telah dites (Jatnika & Irwan, 2010).
b. Asumsi bahwa adanya kemungkinan terhadap eksekusi jalur yang tidak benar. Dengan *white box testing* dapat ditemukan kesalahan ini (Jatnika & Irwan, 2010).
c. Kesalahan penulisan yang acak, seperti berada pada jalur logika yang membingungkan pada jalur normal (Jatnika & Irwan, 2010).

Contoh penggunaan *White-Box Testing* dapat dilihat pada penjelasan di bawah ini:



Gambar 2.9*Flowgraph* Aktivitas Admin

Berdasarkan *cyclomatic complexcity* V(G) pada *egde* dan *node* dapat diketahui dengan melakukan perhitungan sebagai berikut.

rumus :

V(G) = E – N + 2

E (edge) = 16

N (node) = 14

P (Predikat node) = 3

Penyelesaian :

V(G) = E – N + 2

= 16 – 14 + 2

= 4

Predikat (P) = P + 1

= 3 + 1

= 4

(1) Berdasarkan hasil perhitungan tersebut region yang didapat adalah 4

(2) *Independent Path* pada *Flowgraph* tersebut sebagai berikut:

Path 1 = 1-2-3-4-5-6-8-13-14

Path 2 = 1-2-3-4-3-4-5-6-8-13-14

Path 3 = 1-2-3-4-5-6-7-13-14

Path 4 = 1-2-3-4-5-6-8-9-10-11-12-13-14

14. *Black Box Testing*

Black box testing atau dapat disebut juga *Behavioral Testing* adalah pengujian yang dilakukan untuk mengamati hasil input dan output dari perangkat lunak tanpa mengetahui struktur kode dari perangkat lunak. Pengujian ini dilakukan di akhir pembuatan perangkat lunak untuk mengetahui apakah perangkat lunak dapat berfungsi dengan baik.

Untuk melakukan pengujian, penguji tidak harus memiliki kemampuan menulis kode program. Pengujian ini dapat dilakukan oleh siapa saja.

a. Teknik *Black Box Testing*

Ada beberapa teknik yang biasanya digunakan untuk menguji perangkat lunak. Berikut ini adalah teknik-tekniknya:

1) *All pair testing*

Teknik *all pair testing* ini dikenal juga dengan *pairwise testing*. Pengujian ini digunakan untuk menguji semua kemungkinan kombinasi dari seluruh pasangan berdasarkan input parameternya.

2) *Boundary value analysis*

Teknik ini berfokus pada pencarian *error* dari luar atau sisi dalam perangkat lunak.

3) *Cause-effect graph*

Berikutnya adalah teknik *cause-effect graph*. Teknik pengujian ini menggunakan grafik sebagai patokannya. Grafik ini menggambarkan relasi antara efek dan penyebab dari *error*.

4) *Equivalencepartitioning*

Teknik ini bekerja dengan cara membagi data input dari beberapa perangkat lunak menjadi beberapa partisi data.

5) *Fuzzing*

Fuzzing merupakan teknik pencarian *bug* dalam perangkat lunak dengan memasukkan data yang tidak sempurna.

6) *Orthogonal arraytesting*

Selanjutnya adalah *orthogonal array testing*. Teknik ini digunakan jika input berukuran kecil, akan tetapi cukup berat jika digunakan dalam skala yang besar.

7) *Statetransition*

Terakhir adalah *state transition*. Teknik ini berguna untuk melakukan pengujian terhadap mesin dan *navigasi* dari UI dalam bentuk grafik.

b. Keuntungan *Black Box Testing*

Ketika *behavioral* testing digunakan dalam pengujian perangkat lunak, ada beberapa keuntungan yang dapat diperoleh dari pengujian tersebut. Berikut ini adalah keuntungannya:

1) Penguji tidak harus memiliki pengetahuan tentang suatu bahasa pemrograman.

2) Pengujian dilakukan berdasarkan sudut pandang pengguna. Hal tersebut dilakukan agar dapat menemukan inkonsistensi dalam perangkat lunak.
3) Pengembang dan penguji memiliki ketergantungan satu dengan yang lainnya.
4) Penguji tidak perlu memeriksa kode.
5) Memungkinkan penguji dan pengembang bekerja secara independen tanpa mengganggu proses kerja satu sama lain.

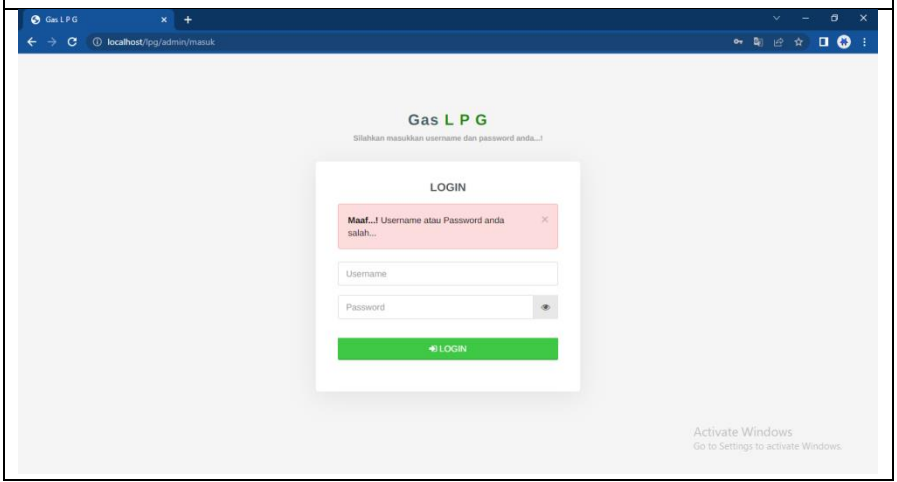
c. Kekurangan *Black Box Testing*

Selain memiliki keuntungan, *behavioraltesting* juga memiliki kekurangan. Berikut ini adalah beberapa kekurangannya:

1) Memiliki kemungkinan kesalahan tidak terdeteksi karena kurang teliti dan tidak adanya pengetahuan teknis.
2) Ada bagian *back-end* yang tidak diuji sama sekali.
3) Kemungkinan pengujian dilakukan kembali oleh *programmer*.

d. Contoh *Black Box Testing*

Tabel 2.10 Contoh Pengguanaan *Black Box Testing*

Test Faktor	Hasil	Kesimpulan
Masukkan *username* atau *password* yang tidak sesuai	✓	Berhasil, Ketika *username* atau *password* tidak sesuai maka tampil pesan bahwa *username* atau *password* tidak sesuai
Screenshot		
		

B. Kajian hasil penelitian terdahulu

1. Yohanes Bowo Widodo dan Leo Faturahman Prodi Teknik Informatika Fakultas Teknik Universitas MH. Thamrin (2019) dan juga Tata Sutabri Fakultas Teknologi Informasi Universitas Respati Indonesia berhasil membuat sebuah Sistem dengan judul "Tempat Sampah Pintar dengan Notifikasi berbasis *IoT*" menggunakan modul WeMos, sensor *Ultrasonic HC-SR04* yang berfungsi untuk mengukur ketinggian sampah yang kemudian dapat dipantau melalui aplikasi *blink* dan dapat memberi notifikasi melalui *email*.
2. Ridwan Ahmad Ma'arif, Fauziyah dan Nur Hayati Prodi Informatika Fakultas Teknologi Komunikasi dan Informatika Universitas Nasional (2019) berhasil membuat sebuah sistem dengan judul "Sistem *Monitoring* Tempat Sampah Pintar Secara *Real-Time* Menggunakan Metode *Fuzzy Logic* Berbasis *IoT*". Sistem yang dibuat menggunakan *NodeMCUesp8266,* 2 buah sensor *Ultrasonic HC-SR04* yang masing-masing berfungsi untuk mendeteksi ketinggian sampah dan juga jarak manusia ke tempat sampah, kemudian pada sistem ini juga menggunakan *Servo SG90* untuk membuka penutup tempat sampah pada saat ada objek yang mendekati tempat sampah. Selanjtunta pada sistem ini menggunakan aplikasi android untuk memantau keadaan tempat sampah.
3. Reiza Yahya Prodi Teknik Elektro Fakultas Teknologi Informasi dan Elektro Universitas Teknologi Yogyakarta (2018) berhasil membuat sebuah sistem yang berjudul "Purwarupa Kotak Sampah Pintar berbasis *Internet of Things (IoT)*". Sistem tersebut menggunakan Arduino Uno sebagai mikrokontroller

dan Modul *WIFI ESP8266* sebagai perangkat penghubung ke internet. Pada sistem ini juga menggunakan sensor *Ultrasonic HC-SR04* untuk mendeteksi ketinggian sampah serta juga menggunakan sensor tambahan *Flame Sensor* sebagai pendeteksi nyala api dan juga *Buzzer* sebagai alarm. Untuk monitoring kondisi tempat sampah menggunakan WEB.

Berdasarkan penelitian terdahulu yang di uraikan di atas, maka peneliti bermaksud membuat penelitian membuat suatu sistem yang bisa memonitoring ketinggian sampah pada Tempat Pembuangan Sementara (TPS) menggunakan sensor *Ultrasonic HC-SR04* serta Lokasi penempatan TPS menggunakan Google Maps APIyang akan dipantau menggunakan aplikasi Android. Sehingga nantinya diharapkan sistem tersebut dapat meminimalisir terjadinya penumpukan sampah akibat terlambatnya petugas kebersihan mengangkut sampah pada tempat sampah.

BAB III

METODE PENELITIAN

A. Jenis Penelitian

Dalam pembuatan proposal ini digunakan metode *deskripsif* yang menggambarkan fakta-fakta dan informasi secara sistematis, faktual dan akurat. Penelitian ini dilakukan melalui internet yang dapat memberikan sumber data dan pengetahuan mengenai sistem yang diteliti, kemudian mencocokkan dengan kemungkinan yang terjadi dalam usaha penyelesaian masalah.

B. Lokasi dan Waktu

1. Lokasi penelitian ini dilakukan pada BTN. Graha D'naila, Kelurahan Galuang Maloang, Kecamatan Bacukiki, Kota Parepare.
2. Waktu yang digunakan untuk penelitian ini ± 2 bulan.

C. Alat dan Bahan

Dalam proses penelitian dan pembuatan sistem ini maka dibutuhkan adanya perangkat keras dan perangkat lunak. Berikut dibawah ini merupakan perangkat lunak dan perangkat keras yang digunakan dalam penelitian ini.

1. Perangkat Keras

Perangkat keras yang akan digunakan pada penelitian ini dapat dilihat pada tabel dibawah ini:

Tabel 3.1 Spesifikasi Perangkat Keras

No.	Spesifikasi	
1	Merk Laptop	Asus X441UVK
2	*Processor* Laptop	Intel(R) Core(TM) i5-7200U CPU @ 2,50GHz (4 CPUs), ~2,7GHz
3	*Memory* Laptop	4GB

No.	Spesifikasi	
4	*Mikrokontroller*	ESP32
5	Jenis Sensor	*Ultrasonic HC-SR04,*
6	Perangkat Tambahan	Kabel Jumper, Power Bank Tenaga Surya

2. Perangkat Lunak

Perangkat lunak yang digunakan untuk membuataplikasi*Prototype* Sistem *Monitoring* Tempat Sampah berbasis *IoT* dapat dilihat sebagai berikut:

Tabel 3.2 Spesifikasi Perangkat Lunak

No.	Spesifikasi	
1	Sistem operasi	*Windows 10*
2	Tool Pemrograman	*ArduinoIDE, Visual Studio Code*
3	Bahasa Pemrograman	*C* dan *Dart*
4	*Database*	*Firebase*
5	Tool Pemrograman	*ArduinoIDE, Visual Studio Code*

3. Perangkat Mobile

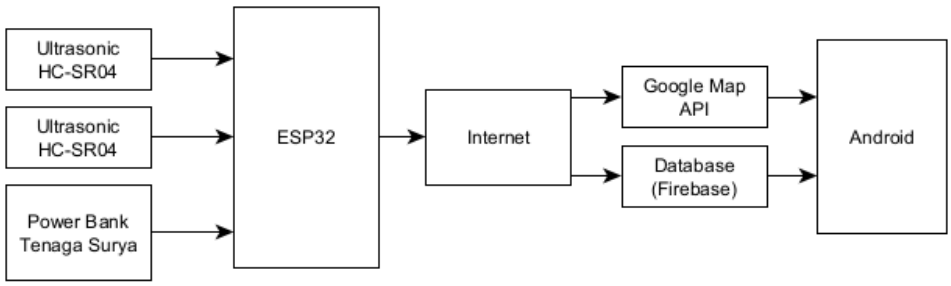
Perangkat *Android* mobile yang digunakan untuk menginstall dan menjalankan aplikasiaplikasi *Prototype* Peternakan Ayam Broiler Berbasis *Internet of Things* dapat dilihat pada tabel berikut:

Tabel 3.3 Spesifikasi Perangkat *Mobile*

No.	Spesifikasi	
1	Model	*Oppo A5s*
2	OS	Android 8.1
3	RAM	*Firebase*
4	Resolution	*ArduinoIDE, Visual Studio Code*
5	CPU	*Mediatek MT6765 Helio P35 Octa-core*
6	LCD	6.22 inci, 720 x 1520 pixels IPS

D. Rancangan Sistem

Perancangan sistem ini meliputi perancangan perangkat keras dan perancangan perangkat lunak. Adapun blok diagram dari sistem yang akan dibuat bisa dilihat di bawah ini:



Gambar 3.1 Diagram Blok

1. ***Power bank tenaga surya***

   Berfungsi untuk menyimpan energidan menyalurkan energi ke perangkat keras lainnya seperti *ESP32*

2. **Modul Sensor *Ultrasonic HC-SR04***

   Berfungsi untuk mengidentifikasi ketinggian sampah pada Tempat sampah.

3. **Google Maps API**

   Berfungsi untuk menentukan titiklokasi dari Tempat sampah

4. ***ESP32***

   Berfungsi sebagai mikrokontroller untuk proses input sehingga dapat menghasilkan output yang diinginkan.

5. ***Google Firebase***

berfungsi sebagai *database* untuk menyimpan nilai ketinggian sampah dan lokasinya sehingga nantinya dapat diakses menggunakan *android*

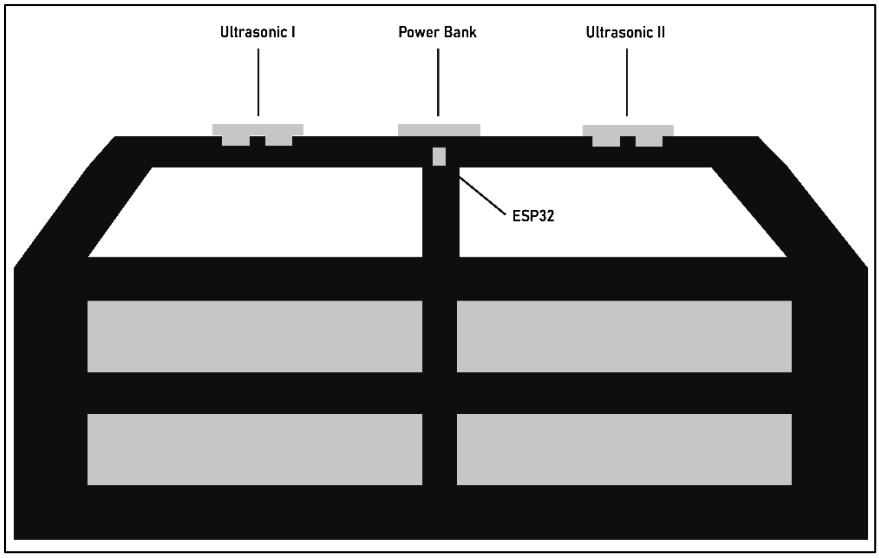
6. ***Android***

Berfungsi sebagai perangkat untuk memantau lokasi dan kapasitas dari tempat sampah

Adapun untuk perancangan perangkat keras dan perangkat lunak akan dijelaskan dibawah ini:

1. **Perancangan Perangkat Keras**

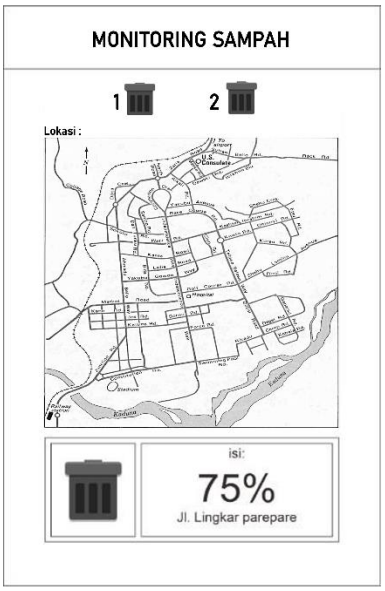
Perancangan perangkat keras ini merupakan tahapan yang sangat penting dalam pembuatan sebuah sistem *prototype* khususnya pada penelitian ini yaitu pembuatan *Prototype* Sistem *Monitoring* Tempat Sampah berbasis *IoT*. Perancangan perangkat keras penting untuk menentukan komponen-komponen yang akan digunakan dalam pembuatan sistem agar komponen yang dipilih dapat sesuai dengan kebutuhan. Berikut merupakan rancangan perangkat keras yang akan digunakan pada pembuatan sistem ini:



Gambar 3.2 Rancangan Perangkat Keras

2. Perancangan Perangkat Lunak

Pada perancangan perangkat lunak ini akan diperlihatkan desain tampilan aplikasi. Berikut dibawah ini merupakan desain tampilan aplikasi tersebut.



Gambar 3.3 Desain tampilan aplikasi

E. Teknik Pengumpulan data

Metode pengumpulan data yang tepat yaitu dengan mempertimbangkan penggunaannya berdasarkan jenis data dan sumbernya. Data yang *objektif* dan relevan dengan pokok permasalahan penelitian merupakanindikator keberhasilan suatu penelitian. Pengumpulan data penelitian ini dilakukan dengan cara sebagai berikut:

1. **Observasi**

   Merupakan metode pengumpulan data dengan cara mengadakan pengamatan langsung kepada objek penelitian yaitu dengan mengunjungi dan mengamati secara langsung kondisi Tempat Pembuangan Sementara yang akan dijadikan objek penelitian.

2. **Wawancara**

   Merupakan teknik pengumpulan data dengan cara mengadakan tanya jawab atau wawancara langsung kepada narasumber. Dalam penelitian ini, peneliti melakukan pengumpulan data dengan mewawancarai langsung warga yang berada di sekitar Tempat Pembuangan Sementara.

3. **Studi Pustaka**

   Mengumpulkan data dengan mempelajari masalah yang berhubungan dengan objek yang diteliti, bersumber dari buku-buku pedoman, literatur yang disusun oleh para ahli untuk melengkapi data yang diperlukan dalam penelitian baik secara *offline* maupun *online*. Adapun beberapa buku yang digunakan dalam mengumpulkan data-data

yang dibutuhkan yaitu buku – buku tentang *mikrokontroler*.

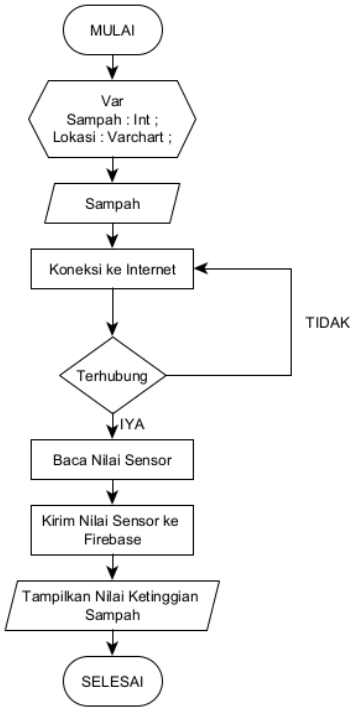
F. Teknik Analisis Data

Teknik analisis data yang digunakan penulis adalah menggunakan data statistik, yaitu menggunakan pembanding untuk hasil akhir keakurasian dan kestabilan antara sensor yang digunakan yaitu *Ultrasonic HC-SR04* dengan menggunakan alat ukur jarak seperti penggaris atau juga meteran. Dan juga apakah yang tampil pada aplikasi sudah sesuai dengan yang diidentifikasi oleh sensor.

G. Diagram Alir

a. Diagram Alir Perancangan Perangkat Keras

Berikut di bawah ini merupakan diagram alir dari perangkat keras yang akan dibuat.

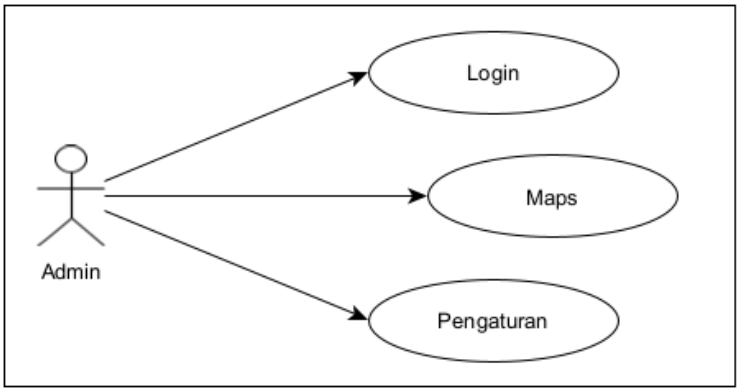


Gambar 3.4 Flowchart rancangan perangkat keras

Berdasarkan gambar di atas sistem perangkat keras dimulai dari Inisialisasi variabel ketinggian tempat sampah dan juga lokasi. Kemudian alat akan dikoneksikan ke internet melalui *wifi*. Setelah mendapat akses internet alat akan membaca nilai dari sensor *Ultrasonic*. Kemudian nilai tersebut akan dikirim ke *firebase*.

b. Diagram Alir Perancangan Perangkat Lunak

Berikut di bawah ini merupakan diagram alir dari perangkat lunak yang akan dibuat.



Gambar 3.5 Use case rancangan perangkat lunak

Tabel 3.4 Penjelasan *use case*

No	Nama Use Case	Deskripsi *Use Case*
1	Login	*Use case* ini menggambarkan proses login petugas kebersihan untuk masuk ke halaman *monitoring*
2	Monitoring	*Use case* ini memberikan informasi ketinggian sampah pada TPS 1 dan TPS 2
3	Maps	Use case ini memberikan informasi letak TPS 1dan TPS2
4	Pengaturan	Use case ini mengatur batas maksimal ketinggian sampah

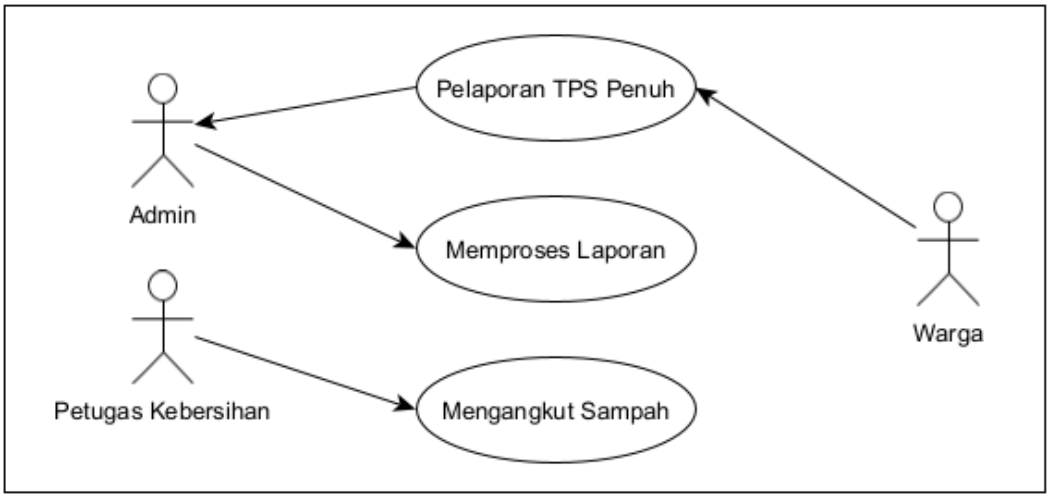
BAB IV

HASIL DAN PEMBAHASAN

A. HASIL

1. Analisis sistem yang berjalan

Saat ini petugas kebersihan pada saat melakukan pengangkutan sampah masih kurang efisien dikarenakan masih sangat bergantung terhadap laporan masyarakat sehingga menyebabkan terjadinya penumpukan sampah.



Gambar 4.1 Use case Sistem yang Berjalan

Tabel 4.1 Penjelasan *use case* sistem yang sedang berjalan

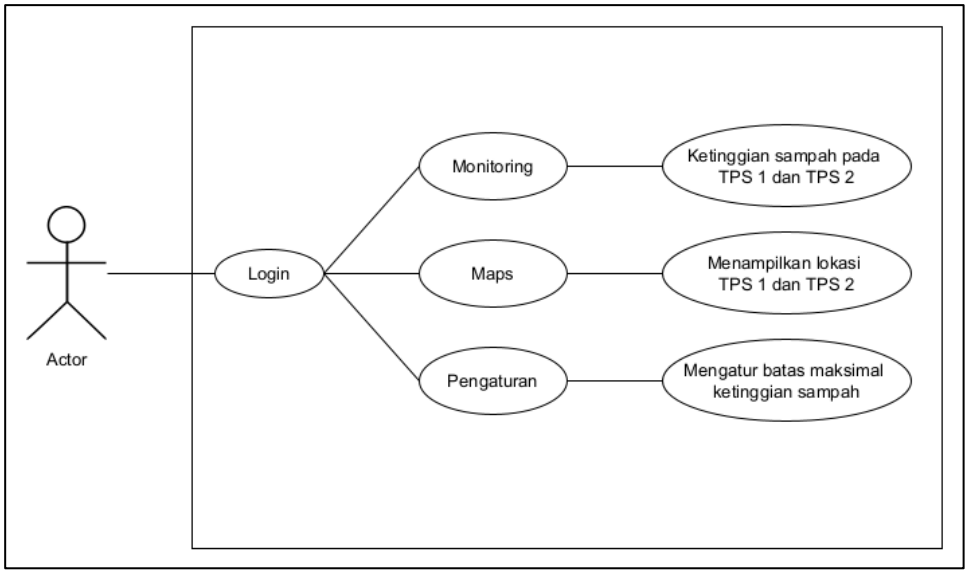
No	Nama Use Case	Deskripsi *Use Case*
1	Pelaporan TPS Penuh	Warga melakukan pelaporan kepada admin bahwa sampah pada TPS penuh
2	Memproses Laporan	Admin akan memproses laporan untuk selanjutnya memerintahkan petugas kebersihan mengangkut sampah
3	Mengangkut Sampah	Petugas kebersihan melaukan pengangkutan sampah pada TPS

2. Analisis sistem yang diusulkan

Dari permasalahan yang dialami diatas makan penulis mengusulkan sebuah sistem yang di harapkan mampu membantu petugas kebersihan untuk memonitoring keadaan dan lokasi TPS (Tempat Pembuangan Sampah) sehingga petugas kebersihan tidak perlu lagi menunggu laporan dari warga untuk mengangkut sampah.

sistem perangkat keras dimulai dari Inisialisasi variabel ketinggian tempat sampah dan juga lokasi. Kemudian alat akan dikoneksikan ke internet melalui *wifi*. Setelah mendapat akses internet ESP32 akan membaca nilai dari sensor *Ultrasonic.* Kemudian nilai tersebut akan dikirim ke *firebase*.

a. *Use case* Diagram aplikasi



Gambar 4.2 *Use Case* diagram aplikasi

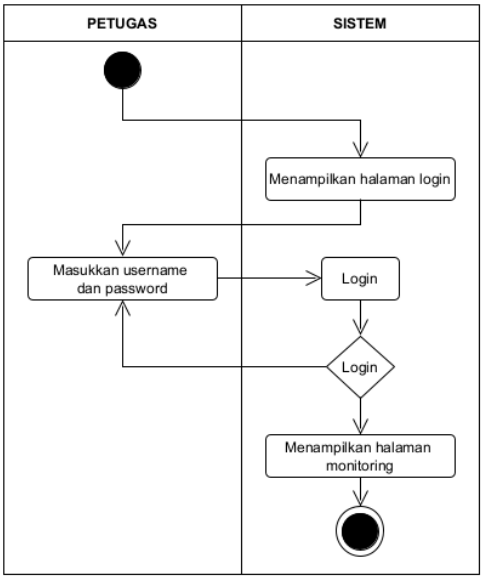
Tabel 4.2 Penjelasan Use case diagram

No	Nama Use Case	Deskripsi *Use Case*
1	Login	*Use case* ini menggambarkan proses login petugas kebersihan untuk masuk ke halaman *monitoring*
2	Monitoring	*Use case* ini memberikan informasi ketinggian sampah pada TPS 1 dan TPS 2
3	Maps	Use case ini memberikan informasi letak TPS 1dan TPS2
4	Pengaturan	Use case ini mengatur batas maksimal ketinggian sampah

b. Activity Diagram

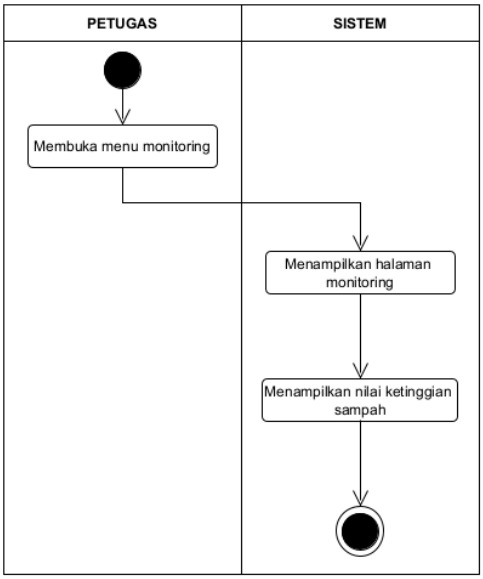
Activity ini menjelaskan proses urutan yang dilalui oleh petugas dalam melakukan pengolahan data pada aplikasi, data yang diolah berupa *login*, *monitoring*, *maps*, pengaturan.

1) *Activity* Diagram *Login*



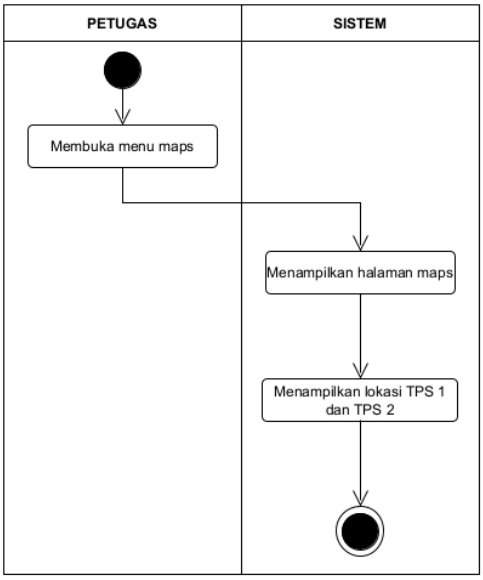
Gambar 4.3 *Activity* Diagram *Login*

2) *Activity* Diagram *Monitoring*



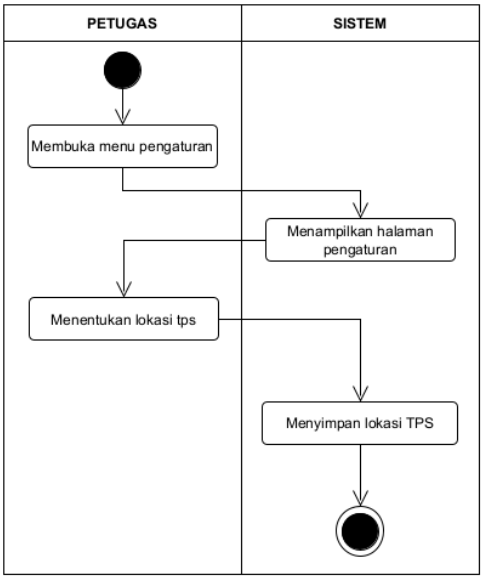
Gambar 4.4 *Activity* Diagram *Monitoring*

3) *Activity* Diagram *Maps*



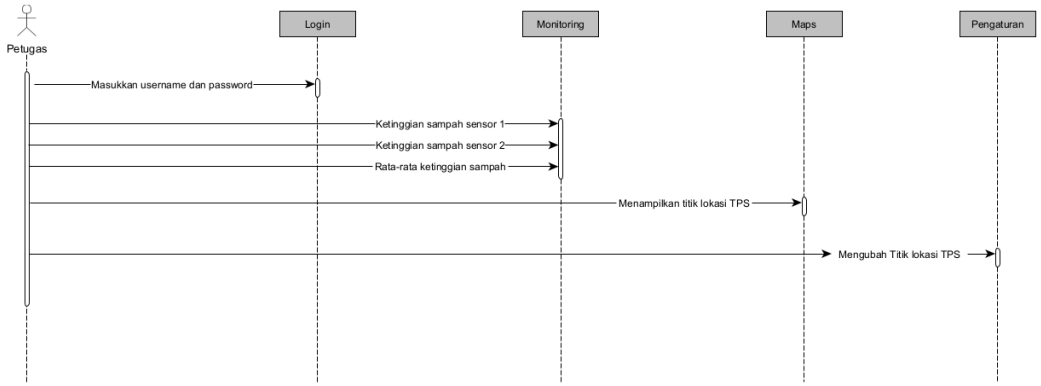
Gambar 4.5 *Activity* Diagram *Maps*

4) *Activity* Diagram Pengaturan



Gambar 4.6 Activity Diagram Pengaturan

c. *Sequence* Diagram



Gambar 4.7 *Sequence* Diagram

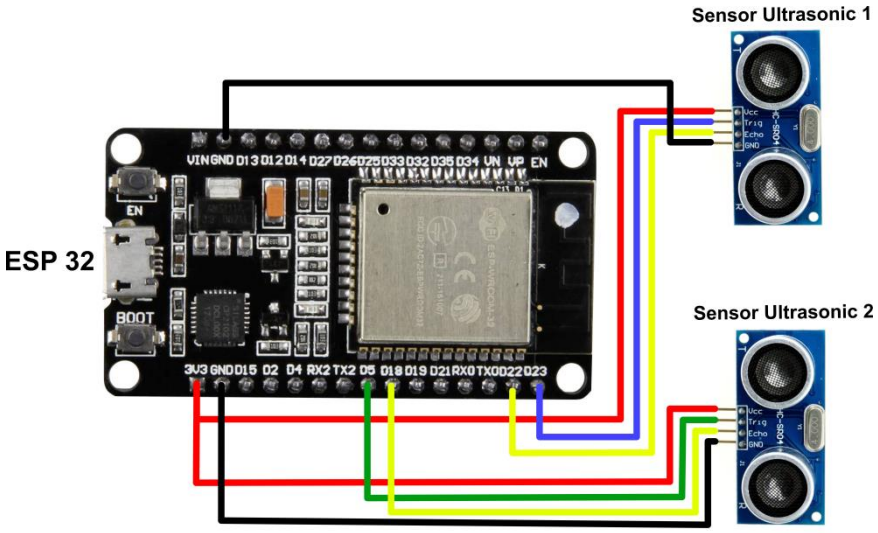
B. Pembahasan

pada bagian ini akan dibahas mengenai hasil rancangan perangkat lunak dan perangkat keras serta pengujian perangkat.

1. Hasil Rancangan perangkat Keras (*Hardware)*

Pada perancangan perangkat keras (*Hardware*) menjelaskan beberapa alat yang digunakan pada penelitian ini. Adapun beberapa rangkaian perangkat keras yang digunakan sebagai berikut:

a) Rangkaian Sensor Ultrasonic *HC-SR04*



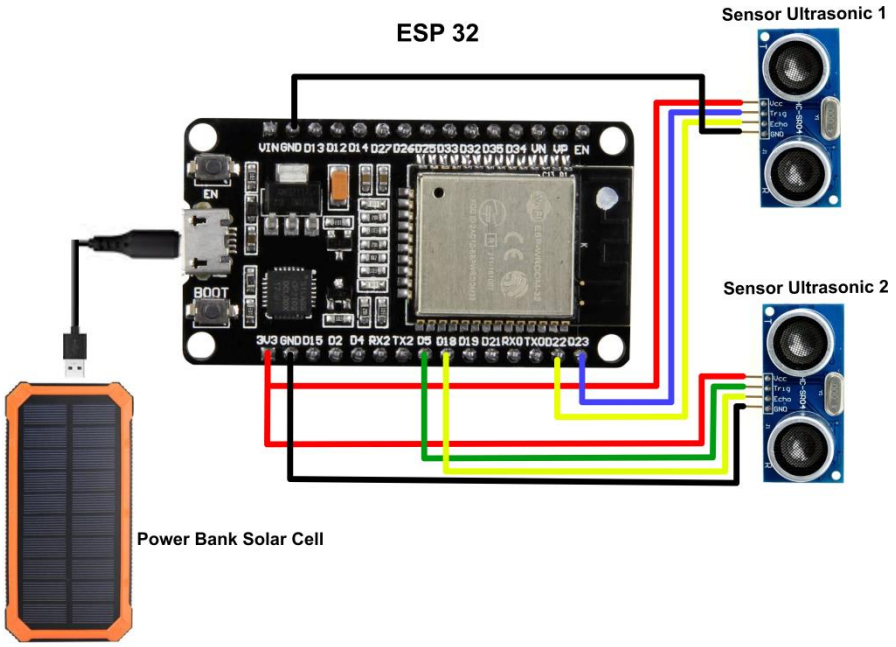
Gambar 4.8 Rangkaian Ultrasonic HC-SR04

pada rangkain ini penulis menggunakan sensor *Ultrasonic* yang digunakan untuk mengukur ketinggian sampah, pada rangkaian ini dapat dilihat pada Gambar 4.8 menggunakan 4 (empat) buah pin pada setiap sensor, pin (Trig) pada sensor 1 (satu) dihubungkan ke pin (23) *ESP32*, kemudian pin (Echo) pada sensor 1 (satu) dihubungkan ke pin (22), kemudian pin (VCC) pada kedua sensor dihubungkan ke pin (3v3) pada *ESP32* dan pin (GND) pada kedua sensor dihubungkan ke pin (GND) pada *ESP32*. Berikut tabel rangkaian tersebut.

Tabel 4.3 Rangkaian sensor Ultrasonic

No	***ESP32***	***Ultrasonic HC-SR04***
1	3v3	VCC
2	GND	GND
3	D23	Trig (sensor 1)
4	D5	Trig (sensor 2)
5	D22	Echo (sensor 1)
6	D18	Echo (sensor 2)

b) Rangkaian keseluruhan



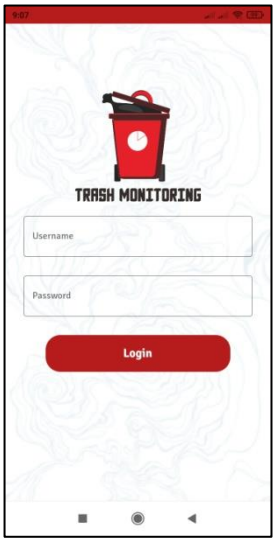
Gambar 4.9 Rangkaian keseluruhan

Dapat dilihat dari Gambar 4.9 merupakan rangkaian keseluruhan perangkat keras (*Hardware*) sistem *monitoring* tempat sampah yang terdiri dari sensor *Ultrasonic* sebagai alat pengukur ketinggian sampah, *ESP32* sebagai pemroses data dan power bank sebagai sumber daya.

2. Hasil Rancangan Perangkat Lunak (*Software*)

Penulis membuat perangkat lunak yang digunakan untuk memonitoring kerja perangkat keras yang dapat menampilkan nilai ketinggian sampah pada TPS secara *real time*.

a. Tampilan aplikasi *Login*



Gambar 4.10 Tampilan *login*

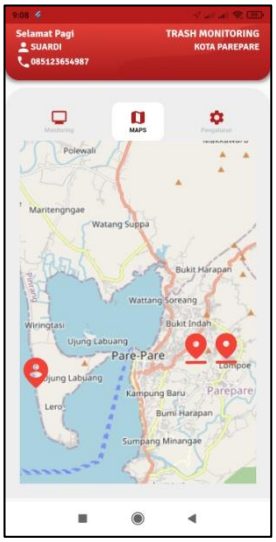
Pada Gambar 4.10 adalah tampilan *login* yang bertujuan sebagai halaman pembuka sebelum mengakses aplikasi *monitoring* tempat sampah, pada halaman ini petugas harus mengisi terlebih dahulu *username* dan *password* yang benar untuk selanjutnya menuju ke halaman *monitoring*.

b. Tampilan *monitoring*

Gambar 4.11 Tampilan *monitoring*

Pada Gambar 4.11 adalah tampilan *monitoring* yang bertujuan agar petugas dapat memantau secara langsung dan *realtime* ketinggian sampah pada TPS, pada setiap TPS terdapat 2 sensor yang mengukur ketinggian sampah dan akan ditampilkan nilai ketinggian sampahnya beserta rata-rata ketinggian sampah pada TPS.

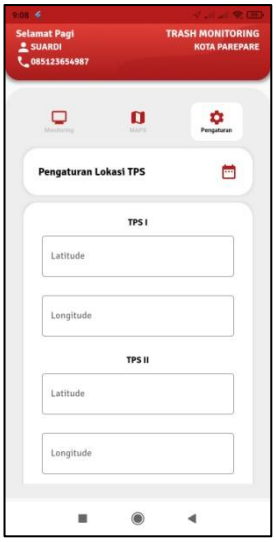
c. Tampilan *maps*



Gambar 4.12 Tampilan *maps*

Pada Gambar 4.12 adalah tampilan *maps* yang menampilkan titik lokasi TPS atau tempat pembuangan sampah, *maps* ini bertujuan agar petugas dapat melihat lokasi TPS dengan mudah melalui aplikasi.

d. Tampilan pengaturan



Gambar 4.13 Tampilan pengaturan

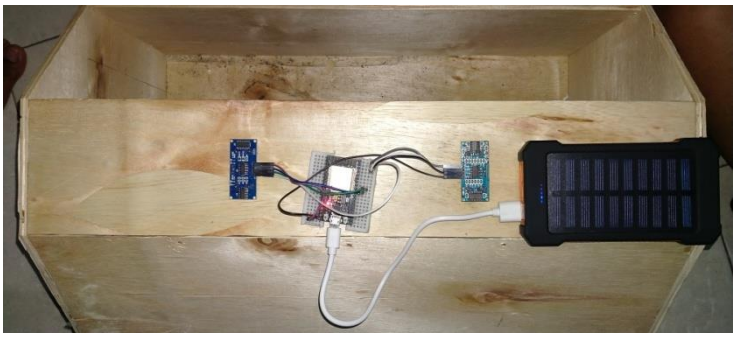
Pada Gambar 4.13 merupakan tampilan pengaturan aplikasi yang terdiri dari menentukan *latitude* dan *longitude* sebagai titik lokasi TPS.

3. Rancangan *Prototype*

Berikut adalah hasil rancangan *prototype* secara keseluruhan dari sistem *montoring* tempat sampah berbasis *Internet of Things.*

Gambar 4.14 Rancangan *prototype* keseluruhan

Dari Gambar 4.14 terlihat bentuk fisik rancangan dari sistem yang terbuat dari bahan utama kayu dibentuk menyerupai bentuk tempat pembuangan sampah dengan ukuran panjang 50 cm, lebar 15 cm dan tinggi 30 cm.



Gambar 4.15 Rancangan komponen sistem

Pada Gambar 4.15 terdapat beberapa komponen yang digunakan dalam perancangan *prototype* ini, diantaranya 2 (Satu) buah *ESP32*, kemudian 4 (Empat) sensor *Ultrasonic*, kemudian 2 (Dua) Power Bank Solar Cell dan 1(Satu) breadboard. Pada penelitian ini rangkaian *ESP32*

dihubungkan menggunakan kabel *jumper* dengan sensor *Ultrasonic* dan sumber daya dari *ESP32* diperoleh dari *Power Bank Solar*

4. Pengujian alat

a. Pengujian nilai sensor *Ultrasonic HC-SR04*

Pada pengujian sensor *ultrasonic* dilakukan dengan membandingkan nilai yang didapatkan oleh sensor *ultrasonic* dengan nilai yang didapatkan dari pengukuran menggunakan penggaris untuk mengetahui persentase perbedaan nilai dari kedua alat tersebut. Berikut ini hasil pengujian yang dilakukan:

Tabel 4.4 Hasil pengujian sensor *Ultrasonic* 1

No	***Ultrasonic HC-SR04 (Centi meter)***	**Penggaris *(centi meter)***	**Selisih**	**Error (%)**
1	8,06	8	0,06	0,008
2	19,98	20	0,02	0,001
3	15,14	15	0,14	0,009
4	25,14	25	0,14	0,006
5	4,97	5	0,03	0,006
Rata-rata *error*				0,006

Data tabel diatas merupakan hasil pengujian sensor *ultrasonic*. Untuk mendapatkan nilai *error* seperti yang didapat pada tabel diatas dilakukan perhitungan dengan rumus sebagai berikut:

$$\%error = \frac{Nilai\ Sensor - Nilai\ Acuan}{Nilai\ Acuan} \text{ x } 100\% \quad \text{.....(1)}$$

Tabel 4.5 Hasil pengujian sensor *Ultrasonic* 2

No	***Ultrasonic HC-SR04 (Centi meter)***	**Penggaris *(centi meter)***	**Selisih**	**Error (%)**
1	7,89	8	0,11	0,014
2	19,81	19	0,19	0,01
3	14,66	14	0,34	0,024
4	24,70	24	0,70	0,029
5	5,23	5	0,23	0,046

Rata-rata *error*	0,025

Data tabel diatas merupakan hasil pengujian sensor *Ultrasonic*. Untuk mendapatkan nilai *error* seperti yang didapatkan pada tabel diatas dilakukan perhitungan dengan rumus sebagai berikut:

$$\%error = \frac{Nilai\ Sensor - Nilai\ Acuan}{Nilai\ Acuan} \text{ x } 100\% \quad \text{.....(2)}$$

Tabel 4.6 Hasil pengujian nilai rata-rata sensor ultrasonic 1 dan 2

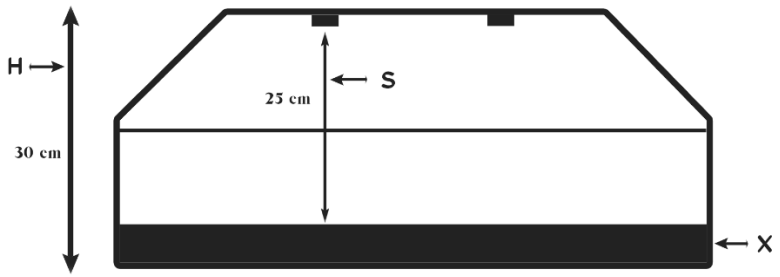
No	***Ultrasonic HC-SR04* (Sensor 1)**	***Ultrasonic HC-SR04* (Sensor 2)**	**Rata-rata (*Centimeter*)**
1	8,06	7,89	7,97
2	19,98	19,81	19,89
3	15,14	14,66	14,9
4	25,14	24,70	24,92
5	4,97	5,23	5,1

Data tabel diatas merupakan hasil perhitungan nilai rata-rata dari sensor *Ultrasonic* 1 *dan* sensor *Ultrasonic* 2. Untuk mendapatkan nilai rata-rata seperti yang didapatkan pada tabel diatas dilakukan perhitungan dengan rumus sebagai berikut:

$$\text{X} = \frac{XI + X2 + \cdots + Xn}{n}$$

Keterangan :

n = jumlah data



H = Tinggi tempat sampah
S = Jarak sensor dengan sampah
X = Ketinggian sampah

X = H – S
= 30 cm – 25 cm
= 5 cm

Gambar 4.16 Rumus mengukur ketinggian sampah

b. Pengujian ketahanan Power Bank Solar Cell

Pada pengujian power bank solar cell dilakukan dengan menyambungkan power bank solar cell ke alat monitoring sampah tanpa sinar matahari untuk mengetahui berapa lama power bank solar cell dapat bertahan pada malam hari. Berikut hasil pengujian yang dilakukan.

Tabel 4.7 Pengujian ketahanan Power Bank Solar Cell

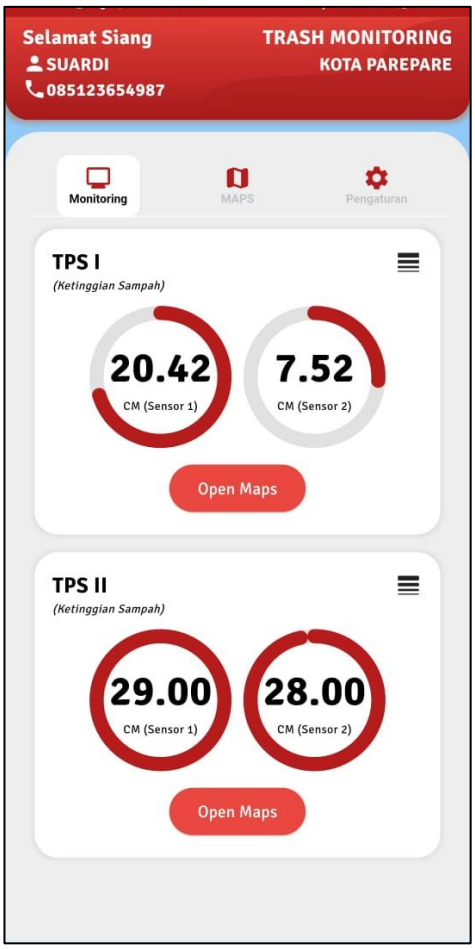
No	Waktu Menyala	Waktu Mati	Selisih
1	20.34 WITA	08.59 WITA	12 Jam 25 Menit
2	18.30 WITA	06.00 WITA	12 Jam 30 Menit
3	18.06 WITA	06.27 WITA	12 Jam 21 Menit

5. Pengujian *Black Box*

Berikut merupakan pengujian *Black Box* dari aplikasi *prototype* sistem *monitoring* tempat sampah berbasis *internet of things.*

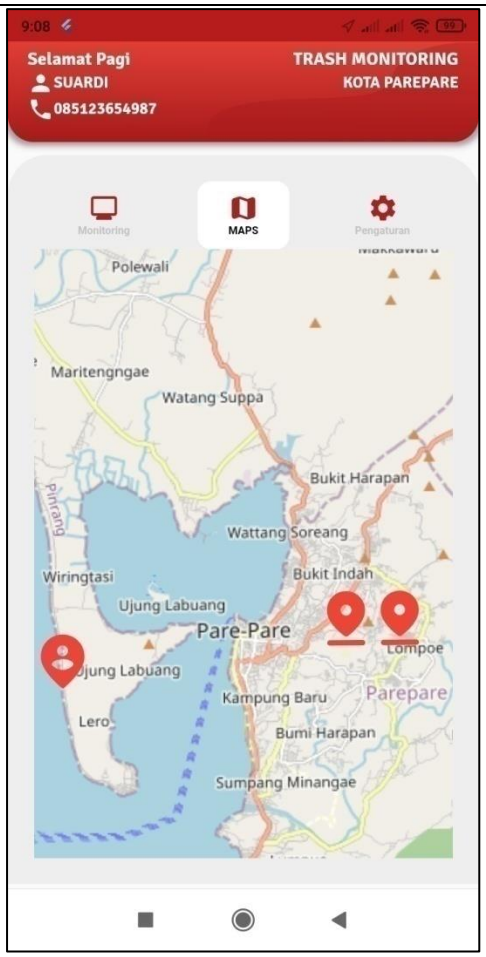
a. Pengujian monitoring pada aplikasi

Tabel 4.8 Pengujian halaman monitoring sensor pada aplikasi

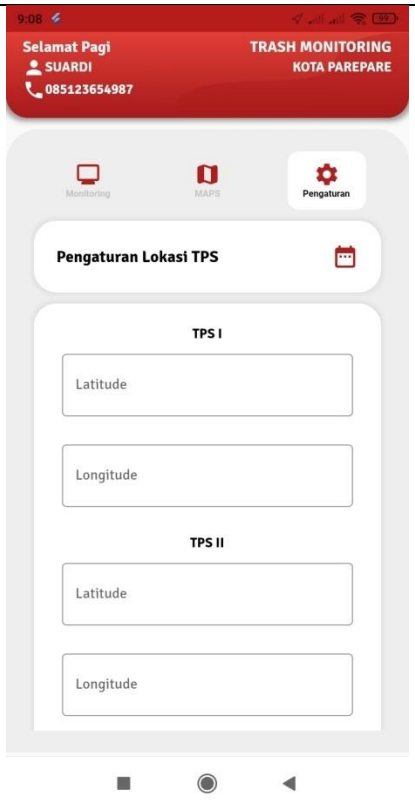
Uji coba	Hasil yang diharapkan	Hasil	Keterangan
Membuka halaman *monitoring* sensor	Nilai pada halaman *monitoring* sesuai dengan nilai sensor yang didapatkan melalui *serial monitor*	□	[x] diterima [ ]ditolak
Foto / *Screenshot*			
			

b. Pengujian lokasi

Uji coba	Hasil yang diharapkan	Hasil	Keterangan
Membuka halaman *Maps*	Halaman *Maps* menampilkan titik lokasi tempat sampah	□	[x] diterima [ ]ditolak

Foto / *Screenshot*


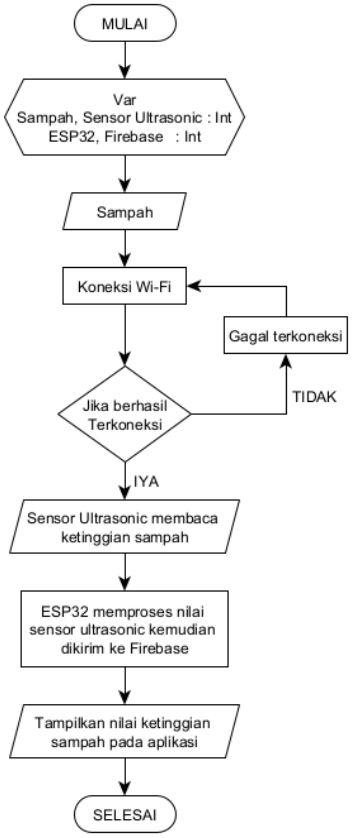
c. Pengujian pengaturan lokasi

Uji coba	Hasil yang diharapkan	Hasil	Keterangan
Membuka halaman Pengaturan	Halaman pengaturan dapat menyimpan titik lokasi dengan baik		[x] diterima [ ]ditolak
Foto / *Screenshot*			
			

6. Pengujia *White Box*

a. Halaman monitoring sensor

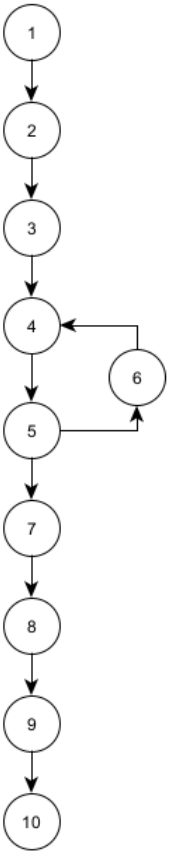
1) Flowchart



Gambar 4.1 *Flowchart monitoring* sensor

Berdasarkan gambar 4.16 di atas, monitoring sensor dimulai dengan sistem yang akan melakukan inisialisasi kemudian sistem akan mencoba terkoneksi ke internet. Apabila sistem telah terkoneksi dengan internet maka sistem akan mengambil nilai sensor dari database yaitu nilai ketinggian sampah. Setelah nilai telah didapatkan maka sistem akan menampilkannya ke halaman monitoring sensor.

2) *Flowgraph*



Berdasarkan gambar 4.37 yang disajikan diatas maka dapat dilakukan proses perhitungan sebagai berikut:

a) Menghitung *cyclomatic Complexcity* V(G) dari *Edge* dan *Node*

Dengan rumus V(G) = E – N + 2

Dengan *Edge* = 10

Dengan *Node* = 10

Dengan Predikat *Node* = 1

Penyelesaian:

V(G) = E – N + 2

= 10 – 10 + 2

= 2

Predikat *Node* = P + 1

= 1 + 1

= 2

b) Berdasarkan perhitungan *Cyclomatic Complexcity* dari *flowgraph* diatas memiliki *Region* = 2

c) *Independent path* pada *flowgraph* diatas adalah:

Path 1 = 1 – 2 – 3 – 4 – 5 – 7 – 8 – 9 – 10

Path 2 = 1 – 2 – 3 – 4 – 5 – 6 – 4 – 5 – 7 – 8 – 9 – 10

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan rumusan masalah, tujuan penelitian dan pembahasan yang sudah diuraikan, maka dapat ditarik kesimpulan bahwa alat monitoring tempat sampah berbasis IoT dapat dibuat dengan menggunakan ESP32, sensor ultrasonic dan power bank solar cell yang terhubung ke aplikasi android untuk memudahkan pengguna atau petugas kebersihan untuk memonitoring tempat sampah dan memantau ketinggian tempat sampah setiap saat. Sensor *ultrasonic* yang digunakan pada alat tersebut cukup baik karena nilai yang didapatkan dari sensor *ultrasonic* tidak jauh berbeda dari nilai yang didapat dari penggaris sebagai alat ukur. Dari pengujian yang dilakukan didapatkan nilai presentase *error* sebesar 0,006% (*Ultrasonic1)* dan 0,025% (*Ultrasonic2*) masing-masing setelah 5 kali pengujian.

B. Saran

Setelah dilakukan pembuatan Perancangan Sistem Monitoring Tempat Sampah Berbasis IoT menggunakan ESP32. Terdapat beberapa saran untuk pembaca dan pengembang selanjutnya. Berikut adalah saran dari penulis:

1. Dalam pengembangan selanjutnya bisa mengganti jenis sensor/ultrasonic yang mampu mendeteksi dengan tingkat ketepatan yang lebih tinggi.

2. Untuk peneliti berikutnya diharapkan menggunakan module GPS untuk mengetahui lokasi tempat sampah yang telah penuh.
3. Penulis menyarankan untuk penelitian berikutnya menambahkan sensor MQ-135 untuk memonitoring supaya dapat mendeteksi kualitas udara dan mendeteksi gas ammonia.

DAFTAR PUSTAKA

Cecep, Dani Sucipto. 2012. Teknologi Pengolahan Daur Ulang Sampah. Semarang: Gosyen Publishing.

FAIZAL, E. (2017). *Sistem Monitoring Tempat Sampah Berbasis Internet Of Things* (Doctoral dissertation, Universitas Gadjah Mada).

Hasyim, A. H. (2021). *Dasar Pemrograman.* CV. Bangun Bumitama.

Henderi. 2018. Unified Modelling Language. Tangerang : Raharja Enrichment Centre (REC).

Jatnika, H., & Irwan, S. (2019). "Testing dan Implementasi Sistem". Jakarta: hendrajatnika.web.id.

Ma'arif, R. A., Fauziah, F., & Hayati, N. (2019). Sistem Monitoring Tempat Sampah Pintar Secara Real-time Menggunakan Metode Fuzzy Logic Berbasis IOT. Jurnal Infomedia: Teknik Informatika, Multimedia, dan Jaringan, 4(2), 69-74.

Mufti and Indra, "Rancang bangun tempat sampah pintar menimbang dan mengenali jenis sampah pada bank sampah budi luhur 1,2," J. Teknol. Inf. . Univ. Budi Luhur, pp. 1–6, 2012.

R. Ganda, "Rancang Bangun Tempat Sampah Pintar Menggunakan Sensor Jarak Berbasis Mikrokontroler Atmega 328," Dept.Fisika, Univ. Sumartra Utara, p. 20, 2018.

Reiza Yahya Prodi (2018). "Purwarupa Kotak Sampah Pintar berbasis Internet of Things (IoT)". Yogyakarta: Universitas Teknologi Yogyakarta.

Ridwan Ahmad Ma'arif, Fauziyah dan Nur Hayati (2019). "Sistem Monitoring Tempat Sampah Pintar Secara Real-Time Menggunakan Metode Fuzzy Logic Berbasis IoT". Jakarta: Universitas Nasioanl.

Sari, R. L. (2021). *Rancang bangun tempat sampah pintar berbasis ESP32* (Doctoral dissertation, Politeknik Harapan Bersama Tegal).

Vandenberg, S. L., Yoder, R. C., Kroenke, D. M., & Auer, D. J. (2018). Database Processing Fundamentals, Design, and Implementation (15th ed.). Hoboken: Pearson.

Wuryanto, A., Hidayatun, N., Rosmiati, M., & Maysaroh, Y. (2019). Perancangan Sistem Tempat Sampah Pintar Dengan Sensor HCRSF04 Berbasis Arduino UNO R3. *Jurnal Khatulistiwa Informatika*, *21*(1), 55-60.

Yohanes Bowo Widodo dan Leo Faturahman (2019). "Tempat Sampah Pintar dengan Notifikasi berbasis IoT". Jakarta: Universitas Respati Indonesia.

Z. A. Zhafira, Z. Doni and Herryawan P, "Analisis dan Rancang Bangun Sistem Monitoring Tempat Sampah Berbasis IOT menggunakan Protokol MQTT," J. Proc. Conf. Electr. Eng. Telemat. Ind. Technol. Creat. Media, pp. 302–307, 2018.