Volume 5, No. 2 2024

JISE

Journal of Internet and Software Engineering

ISSN 2797-9016

https://ugm.id/jise

The journal published by
Department of Electrical Engineering and Informatics
Vocational College, Universitas Gadjah Mada

JISE
Journal of Internet and Software Engineering
Volume 5, No. 2 2024

EDITORIAL TEAM

Journal of Internet and Software Engineering (JISE)

https://ugm.id/jise

The journal published by
Department of Electrical Engineering and Informatics
Vocational College, Universitas Gadjah Mada
Sekip unit III, Caturtunggal, Terban,
Kec. Gondokusuman, Kab. Sleman, D.I. Yogyakarta 55281

JISE
Journal of Internet and Software Engineering
Volume 5, No. 2 2024

https://ugm.id/jise

The journal published by
Department of Electrical Engineering and Informatics
Vocational College, Universitas Gadjah Mada

Studi Komparasi Teknik Pengujian *End-to-End* pada *E-payment*: Studi Kasus Travelink (Sistem *Web E-ticketing*)

Azza Ulil Afidah[1], Divi Galih Prasetyo Putri[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
azzaulil99@mail.ugm.ac.id
[*]Korespondensi: divi.galih@ugm.ac.id;

Abstract – Financial technology is an important agenda in the financial services industry and the economy in the future. One of the technological innovations in the banking, finance, and trade sectors is electronic payment or commonly called e-payment. The e-payment system is very influential in the ticket purchase transaction process on the web, so it is necessary to ensure the success of transactions in a software through end-to-end testing. Regarding the system development period, currently, software development companies want fast and effective testing times, therefore appropriate test execution methods are needed for end-to-end testing of the system. This study uses a case study of the Travelink e-payment system. The uniqueness of this system is that there are payment methods that may require special testing methods. This study was carried out with a comparison between three testing methods, including manual, automated and hybrid testing (a combination of manual and automated testing). Comparison of the three testing methods is carried out using test metrics, requirement coverage and test execution time. The comparison results with the two of test metrics show that the automated testing method covers 100% of the requirements and is the fastest to use for end-to-end testing of e-payment Credit Card and Virtual Account. The hybrid testing method provides 100% test coverage and is the fastest to use in QRIS e-payment end-to-end testing. It is hoped that this can be a reference for determining the test method that will be used in the e-payment system.

Keywords – End-to-end testing, E-payment, Robot Framework, Automated, Hybrid

Intisari – Teknologi finansial menjadi agenda penting dalam industri jasa layanan keuangan dan perekonomian di masa depan. Salah satu inovasi teknologi di bidang perbankan, keuangan dan perdagangan adalah *electronic payment* atau biasa disebut *epayment*. Sistem *e-payment* sangat berpengaruh dalam proses transaksi pembelian tiket pada *web*, sehingga perlu untuk memastikan keberhasilan transaksi dalam suatu perangkat lunak melalui pengujian *end-to-end*. Berkaitan dengan masa pengembangan sistem, perusahaan pengembang perangkat lunak saat ini menginginkan proses pengujian yang cepat dan efektif, oleh karena itu metode eksekusi pengujian yang tepat diperlukan untuk pengujian *end-to-end* pada sistem. Penelitian ini menggunakan studi kasus sistem *e-payment* Travelink. Keunikan dari sistem ini adalah adanya metode pembayaran yang memerlukan metode pengujian khusus. Penelitian ini dilakukan dengan perbandingan antara ketiga metode pengujian, di antaranya pengujian secara manual, otomatis, dan *hybrid* (gabungan manual dan otomatis). Komparasi ketiga metode pengujian dilakukan dengan menggunakan matriks pengujian *requirement coverage* dan waktu eksekusi pengujian. Hasil perbandingan dengan kedua matriks pengujian menunjukkan bahwa metode pengujian otomatis mencakup 100% *requirement* dan yang paling cepat digunakan untuk pengujian *end-to-end epayment Credit Card* dan *Virtual Account*. Metode pengujian *hybrid* memberikan cakupan pengujian 100% dan paling cepat digunakan pada pengujian *end-to-end e-payment QRIS*. Hal ini diharapkan dapat menjadi referensi untuk menentukan metode pengujian yang akan digunakan pada sistem *e-payment*.

Kata kunci – Pengujian *end-to-end, E-payment,* Robot Framework, Otomatis, *Hybrid*

I. PENDAHULUAN

Teknologi finansial menjadi agenda penting pada industri jasa layanan keuangan dan perekonomian di masa depan. Salah satu inovasi teknologi di bidang perbankan, keuangan dan perdagangan adalah pembayaran elektronik atau *Electronic Payment* [1]. Sistem pembayaran elektronik atau *epayment* digambarkan sebagai proses pembiayaan yang dilakukan secara *online* dalam suatu *e-commerce* [2]. Manfaat sistem *e-payment* bagi individu adalah pelanggan dapat melakukan transaksi kapanpun dan dimanapun. Keunggulan dari segi biaya operasional yang gratis untuk semua bisnis, klien, dan pemasok serta manfaatnya dalam menurunkan aktivitas kriminal dan kasus penipuan, merupakan manfaat yang besar untuk bisnis [3].

Pengaruh besar *web* pada kehidupan keseharian adalah individu merasa terbiasa dengan transaksi *online* di *ecommerce* untuk penjualan dan pembelian produk dan usaha [4]. Berdasarkan beberapa aspek di atas, sistem *e-payment* terbukti sangat berpengaruh dalam proses transaksi pembelian secara *online*, sehingga perlu untuk memastikan keberhasilan proses transaksi dalam suatu *software* melalui proses pengujian.

Teknik pengujian *end-to-end* merupakan pengujian yang dilakukan untuk melakukan verifikasi fungsionalitas terhadap keseluruhan alur sistem dan sering digunakan perusahaan dalam pengembangan aplikasi *web* [5], [6], [7]. Pengujian *end-to-end* adalah teknik pengujian yang memiliki cakupan besar dan membutuhkan waktu yang lama dalam pelaksanaannya. Saat ini perusahaan pengembang perangkat lunak menginginkan masa pengembangan sistem yang lebih singkat [8], sehingga upaya pemilihan teknik eksekusi pengujian *end-to-end* (manual atau automasi) diperlukan agar proses pengujian menjadi lebih singkat.

Adapun sistem *e-payment* yang memiliki berbagai macam metode pembayaran memerlukan metode pengujian yang bervariasi juga. Contohnya *e-payment* pada *e-wallet* seperti QRIS (*Quick Response Code Indonesian Standard*) yang mana dalam proses pengujiannya membutuhkan *device* terpisah untuk melakukan *scan* dan membayar [9], [10], [11], sehingga penelitian ini dilakukan untuk membuktikan bahwa adanya teknik pengujian *end-to-end* secara *hybrid* (gabungan manual dan otomatis) menjadi lebih cepat dan efisien. Metode eksekusi yang memungkinkan untuk pengujian *end-to-end* adalah pengujian manual, otomatis dan gabungan keduanya (*hybrid*). Penelitian ini dilakukan untuk menentukan metode eksekusi yang paling sesuai digunakan untuk pengujian *end-to-end* pada studi kasus sistem *e-payment* Travelink.

II. DASAR TEORI

Beberapa penelitian terdahulu telah membahas tentang perbandingan pengujian manual dan otomatis dengan menggunakan beberapa *framework* atau *automation tools* [12]. Adapun penelitian terdahulu yang membahas terkait *requirement coverage* sebagai matriks pengujian dan juga penelitian yang berkaitan dengan pengujian sistem *e-payment*. Berikut penelitian-penelitian terdahulu dengan topik terkait dengan penelitian ini.

A. Perbandingan Pengujian Manual dan Otomatis

Penelitian terkait perbandingan antara pengujian secara manual dan otomatis pernah dilakukan oleh [8] yang mengembangkan automasi pada pengujian *end-to-end web-based application*. Penelitian ini menggunakan Protractor sebagai *automation tool*. Penelitian ini menyimpulkan bahwa pengujian *end-to-end* secara otomatis membutuhkan pengerjaan yang lama pada iterasi pertama. Namun pada iterasi kedua, lama pengerjaan menjadi lebih pendek karena hanya menjalankan kembali skrip pengujian yang sudah dibuat pada iterasi pertama. Oleh karena itu, jumlah iterasi pelaksanaan pengujian *end-to-end* sangat berpengaruh dalam pemilihan teknik pengujian yang digunakan.

B. Matriks *Requirement Coverage*

Penelitian terdahulu yang menggunakan *requirement coverage* sebagai matriks pengujian pernah dilakukan oleh [13]. Penelitian ini menyediakan analisis perbandingan antara MBT (*Model-based testing*), CT (*Combinatorial Testing*) dan pengujian manual dalam hal MC/DC (*Modified Condition/Decision Coverage*) dan *Requirement Coverage*. Evaluasi efisiensi teknik pengujian dilakukan untuk menentukan perbedaan *test suite* yang dihasilkan oleh masing-masing teknik pengujian. Efisiensi diukur dengan penggunaan waktu eksekusi pengujian dan *requirement coverage*. Penelitian ini menyimpulkan bahwa CT adalah teknik pengujian yang paling efisien jika dibandingkan dengan MBT dan pengujian manual di mana CT ditempuh dalam waktu yang singkat dan mencapai 100% *requirement coverage*.

C. Pengujian Sistem *E-payment*

Penelitian yang dilakukan oleh [14] membahas tentang pengembangan dan pengujian pada *website e-commerce* dengan sistem pembayaran QR *code*. Pengujiannya dilakukan dengan pengujian fungsionalitas dan *system testing*. Adapun pengujian *front-end* dilakukan dengan memesan produk yang kehabisan stok dan verifikasi proses pembayaran berfungsi dan dapat diandalkan tanpa eror atau notifikasi yang menandakan transaksi gagal. Hasil pengujian fungsionalitas adalah pengguna berhasil melakukan pemesanan produk dan menyelesaikan pembayaran dengan sukses.

Berdasarkan dari penelitian-penelitian terdahulu yang serupa dengan penelitian ini, dapat diketahui bahwa perbandingan pengujian terfokus pada metode pengujian manual dan otomatis. Perbedaan penelitian pada studi kasus sistem *e-payment* Travelink ini adalah selain membandingkan metode pengujian manual dan otomatis, penelitian ini juga membandingkan metode pengujian gabungan dari manual dan otomatis (*hybrid*). Perbandingan metode pengujian pada penelitian ini menggunakan dua matriks yaitu *requirement coverage* dan waktu eksekusi pengujian.

III. METODOLOGI

A. Bahan

Data atau informasi utama yang digunakan pada penelitian ini berasal dari sistem *web e-ticketing* Travelink sebagai *System Under Test* (SUT). Sistem Travelink ini diambil dari perusahaan *payment gateway* yaitu PT. Aino Indonesia sebagai studi kasus untuk penelitian ini. Adapun pada studi kasus *website* Travelink yang memiliki sistem *e-payment* terintegrasi dan sistem *e-payment* tersebut yang akan digunakan sebagai bahan utama pada penelitian ini.

Adapun beberapa metode *e-payment* yang tersedia pada Travelink dan digunakan dalam pengujian ini adalah *e-payment credit / debit card, virtual account*, dan QRIS (*Quick Response Code Indonesian Standard*). Ketiga metode tersebut memiliki proses pembayaran yang berbeda, sehingga dipilih sebagai bahan perbandingan teknik pengujian pada penelitian ini.

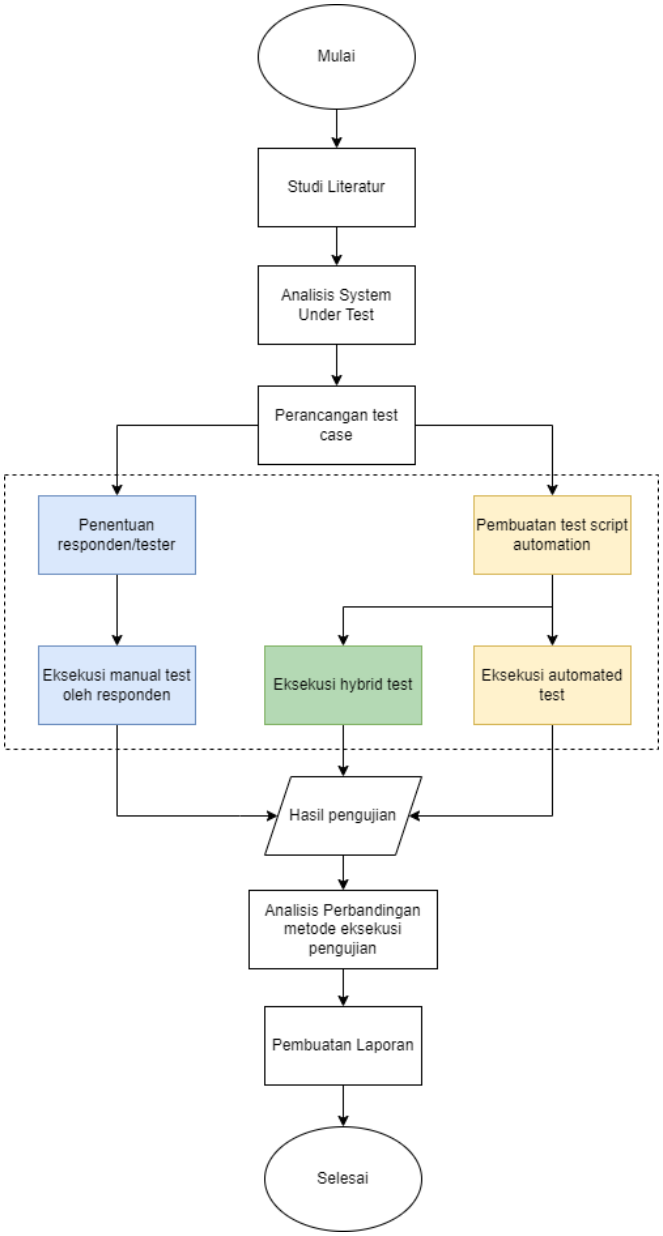
B. Peralatan

Penelitian dilakukan dengan menggunakan perangkat keras dan perangkat lunak. Spesifikasi dan peralatan yang digunakan dalam penelitian ini ditampilkan pada rincian berikut.

1. Notebook Intel® CoreTM i5-7200U, CPU @2.50GHz, RAM 8 GB dengan sistem operasi Windows 10 Pro
2. Chrome browser v118.0 dan chromedriver v118.0
3. Python v3.11.2 dan pip v22.3.1 untuk instalasi Robot Framework
4. Robot Framework v6.0.2
5. Visual Studio Code sebagai *text editor*
6. Mobile app yang digunakan untuk pembayaran *e-payment* QRIS

C. Tahapan Penelitian

Penelitian ini terdiri dari beberapa tahapan yang dapat dilihat pada Gambar 1.

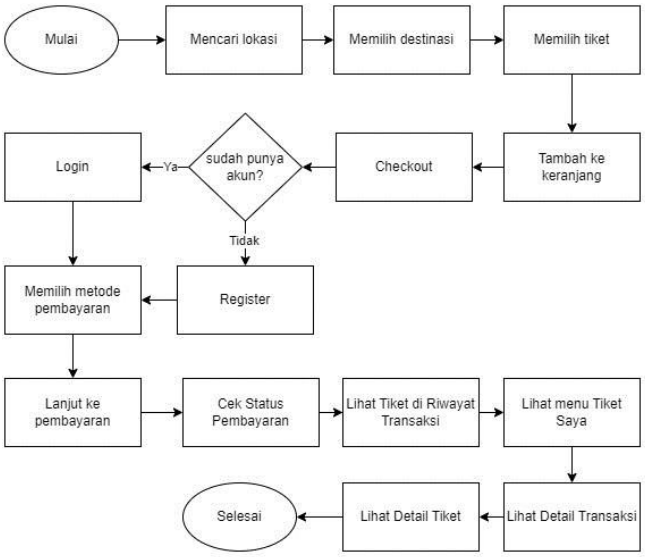


Gambar 1. Alur penelitian

Penelitian dapat dilihat pada Gambar 1 diawali dengan proses analisis *system under test* untuk proses memahami sistem yang diuji terkait alur kerja dan pengenalan fitur-fitur yang akan diuji. Kemudian dilakukan proses studi literatur terhadap penelitian-penelitian terdahulu yang membahas tentang bagaimana membuat pengujian otomatis dan perbandingannya dengan pengujian manual. Proses selanjutnya adalah perancangan *test case* atau skenario pengujian yang akan digunakan sebagai panduan dalam melakukan pengujian serta untuk memenuhi persyaratan kebutuhan dari sistem yang diuji. Setelah dilakukan perancangan *test case*, dilanjutkan proses eksekusi pengujian yang terbagi menjadi tiga metode, yaitu manual, otomatis, dan *hybrid*. Setelah didapatkan *output data* untuk perbandingan dari masing-masing metode pengujian, proses selanjutnya yaitu analisis perbandingan metode eksekusi pengujian dengan menggunakan waktu eksekusi dan *requirement coverage*. Proses yang terakhir adalah pembuatan laporan untuk menyajikan penelitian ini sesuai sistematika penulisan yang telah ditentukan. Alur penelitian terdapat tiga proses yang dibedakan dengan warna biru, hijau dan kuning. Proses dengan warna biru merupakan proses pengujian secara manual. Sedangkan proses dengan warna kuning adalah proses pengujian otomatis dan warna hijau untuk proses pengujian secara *hybrid*.

D. Analisis Sistem *Under Test*

Travelink merupakan sistem berbasis web yang digunakan untuk pembelian tiket masuk wisata secara *online* dan memiliki sistem pembayaran terintegrasi dengan aplikasi. Alur utama dalam sistem ini adalah pembeli dapat memilih tiket dan menambahkan ke keranjang, kemudian *checkout* keranjang dan melakukan pembayaran dengan metode pembayaran yang telah tersedia. Lalu sistem akan *generate*/membuat tiket *online* yang dapat digunakan untuk validasi tiket untuk masuk ke dalam destinasi. Gambaran alur utama pada sistem Travelink dapat dilihat pada Gambar 2.



Gambar 2. *User flow* Travelink

E. Perancangan Pengujian

1. Perancangan *Test Case*

Tahap perancangan *test case* adalah proses perancangan langkah pengujian yang akan digunakan sebagai dasar dari penyusunan skrip pengujian. Perancangan *test case* perlu dilakukan agar langkah-langkah yang diterapkan pada *framework* yang dipakai sudah benar, tidak cacat, dan dapat

diimplementasikan. Tahapan ini berguna untuk memastikan cakupan pengujian lengkap dan sesuai dengan melakukan verifikasi dan validasi terhadap kriteria dari fitur yang akan diuji. Tahapan perancangan *test cast* dapat ditunjukkan pada Tabel 1.

Tabel 1. *Test case*

Test ID	*Test Type*	*Pre-condition*	*Test Case*	*Test Step*	*Expected Result*
REG_01	Positif	User berada di halaman Registrasi	Registrasi dengan data valid	1. Isi *form register* dengan data valid 2. Klik tombol Lanjutkan	*Register* berhasil
REG_02	Negatif	User berada di halaman Registrasi	Registrasi dengan data kosong	1. Klik kolom email 2. Biarkan kosong 3. Klik tombol Lanjutkan	Menampilkan pesan *error* pada *text field* bahwa kolom ini harus diisi
REG_03	Negatif	User berada di halaman Registrasi	Registrasi dengan email terdaftar	1. Isi *form register* dengan email yang sudah didaftarkan 2. Klik tombol Lanjutkan	*Register* gagal dan muncul pesan *error* pada *text field* email bahwa email sudah digunakan
REG_04	Negatif	User berada di halaman Registrasi	Format email tidak valid	1. Isi kolom email dengan format tidak valid	Menampilkan pesan *error* pada *text field* bahwa email ini tidak valid
REG_05	Negatif	User berada di halaman Registrasi	Nomor hp diisi selain angka	1. Isi kolom nomor hp dengan huruf/simbol	Menampilkan pesan *error* pada *text field* bahwa kolom ini hanya bisa diisi dengan angka

Perancangan *test case* mempertimbangkan data *input* dan *output* dari setiap fitur yang menyebabkan *test case* dibagi menjadi dua jenis yaitu *test case* positif dan negatif. *Test case* positif dirancang untuk memeriksa apakah aplikasi melakukan apa yang diharapkan dengan memberikan data input yang valid. Sedangkan *test case* negatif dibuat untuk memeriksa apakah aplikasi berperilaku seperti yang diharapkan dengan *input* negatif dan memvalidasi aplikasi terhadap data yang tidak valid. *Test case* negatif bertujuan agar aplikasi tidak melakukan apa yang seharusnya tidak dilakukan dan untuk memeriksa apakah kesalahan ditampilkan pada kondisi yang seharusnya. Hal ini diperlukan untuk teknik pengujian *end-to-end* dalam menguji fungsionalitas sistem secara keseluruhan. Salah satu *test case* yang telah dibuat dapat dilihat pada Tabel 1.

2. Pengaturan Robot Framework

Tahap ini mencakup aktivitas persiapan *environment* yang akan digunakan untuk pengujian, pengidentifikasian elemen halaman sistem, pembuatan *keywords* serta pembenahan pada setiap *error* yang mungkin muncul ketika proses pembuatan. *Environment* yang digunakan mencakup *framework* pengujian, *library–library* yang terkait dengan *framework* pengujian yang dipilih, serta perangkat lunak lain seperti *extension browser* untuk pengambilan elemen halaman sistem.

Instalasi Robot framework dapat menggunakan pip, program untuk manajemen paket di Python. Instalasi Python pada komputer biasanya menginstal pip secara otomatis dan dokumentasi lebih lengkapnya terdapat pada situs resmi Robot framework. Setelah semua terpasang, maka selanjutnya adalah pembuatan *file* baru dengan ekstensi .robot dalam *folder project* Robot. Selanjutnya proses *coding* menggunakan Selenium Library dan Selenium *webdriver* untuk menjalankan langkah pengujian pada elemen halaman.

Perihal pengaturan *Robot Framework* dijelaskan pada beberapa poin penting di bawah ini:

a. *File Directory*

Project Robot framework disimpan dalam satu *folder* yang berisi file-file .robot dengan berbagai jenis skrip pengujian di dalamnya. Semua *folder* dibuat dengan nama *folder* sesuai dengan yang diinginkan.

b. Pengumpulan *Variables*

Variables pada Robot framework ini didefinisikan dengan bagian *** Variables *** yang tersedia di semua *test cases* dan *keywords* dalam *file* yang sama. Pengumpulan *variable* yang menjadi satu *file* berguna untuk penulisan yang berulang dan mudah dipahami.

c. Membuat *Keywords*

Penggunaan *keywords* yang berasal dari *Selenium Library* untuk membuat skrip detail Langkah pengujian dalam setiap *test case*. *Robot framework* memiliki *default keywords* berupa variasi interaksi pengguna pada suatu *website* seperti halnya *click, hover, hold, drag,* dan *scroll*. Pada proyek ini, *keywords* dikumpulkan menjadi beberapa *file* .resource sesuai dengan fitur yang diuji agar pembacaan dokumentasi yang lebih mudah.

d. Menjalankan Skrip Pengujian

Pembuatan skrip pengujian perlu adanya validasi untuk memastikan bahwa interaksi yang dilakukan berhasil. Pengecekan elemen yang dihasilkan oleh *robot framework*

apakah sudah sesuai dengan yang seharusnya muncul atau tidak. Hal ini dilakukan dengan menjalankan skrip pengujian yang telah dibuat. *Tools* yang digunakan untuk menjalankan skrip pengujian sama seperti yang digunakan untuk pembuatan skrip pengujian yaitu *Visual Studio Code* dengan fitur *Terminal*. Cara menjalankan atau *running test script* dapat dilihat pada Gambar 3.

Gambar 3. *Command* untuk menjalankan skrip

e. Mengakses *Test Report* dan *Test Log*

Test report dihasilkan dalam bentuk *file HTML* dan berisi tentang laporan pengujian yang telah dilakukan pada setiap langkah pengujian, skenario dan keseluruhan pengujian. Informasi pada *test report* terdapat data seperti *outcome status* dan lama eksekusi pengujian. Selain *test report*, terdapat *test log* yang berisi laporan pengujian secara detail seperti gambar tangkapan layar (*screenshot)* apabila terdapat skrip pengujian yang gagal.

3. Penentuan Responden (*Manual Tester*)

Beberapa responden digunakan sebagai penguji dalam pengujian manual untuk mendapatkan lama eksekusi yang dibutuhkan secara *general*. Responden diambil menggunakan teknik *non probability Convenience sampling*. Teknik tersebut menggunakan populasi target yang memenuhi kriteria tertentu seperti kedekatan geografis, kemudahan untuk komunikasi, ketersediaan pada waktu tertentu, atau kesediaan untuk berpartisipasi dalam penelitian. Hal ini yang menjadikan beberapa responden yang terlibat hanya anggota tim Travelink pada PT. Aino Indonesia dengan tujuan menjaga hal yang *confidential* pada *project* ini.

Responden berjumlah tiga orang dan masing-masing menguji sistem secara manual dengan tiga *end-to-end test case* yang sama yaitu *test case e-payment Credit Card, Virtual Account* dan QRIS. Responden dijelaskan beberapa hal sebelum dilakukan eksekusi pengujian, terkait tata cara pelaksanaan pengujian *end-to-end*. Tujuannya adalah untuk menyetarakan kemampuan responden sebagai penguji dan dapat memahami cara melakukan pengujian *end-to-end* secara manual.

IV. HASIL DAN PEMBAHASAN

A. Implementasi *Test Case* Menggunakan Robot Framework

Test case yang telah dirancang kemudian diimplementasikan pada *Robot Framework* dengan penulisan skrip pengujian dengan menggunakan SeleniumLibrary. Implementasi *test case* ini terbagi menjadi eksekusi pengujian *end-to-end* secara otomatis dan *hybrid*, sehingga akan ada 2 *test suite* yaitu *automated testing* dan *hybrid testing*. Salah satu skrip pengujian pada *automated testing* yaitu *test case* pembayaran sukses dengan metode *Virtual Account* (VA) dapat dilihat pada Gambar 4.

```
Pay with VA
    Copy VA number
    Open ATM Simulator in new tab
    Select bank code    013    # VA Permata
    Input VA number
    Execute VA payment
    Take a screenshot    success_VApayment
    Switch Window    title=payment_front

Copy VA number
    Wait Until Element Is Visible
    ...    xpath=//*[@id="card-summary-wrap"]/span[2]/span/div[2]/img    timeout=10s
    Click Element    xpath=//*[@id="card-summary-wrap"]/span[2]/span/div[2]/img
    ...    # copy VA number

Open ATM Simulator in new tab
    Execute Javascript    window.open('')    # open new tab
    Get Window Titles    # get all tab titles
    Switch Window    title=undefined    # switch to new tab
    Go To    ${atm_simulator_url}
    Wait Until Page Contains    Virtual Account Simulator

Select bank code    [Arguments]    ${bank_code}
    Click Element    bankcode
    Click Element    xpath=//option[@value='${bank_code}']

Input VA number
    Click Element    //input[@id='rekening']
    Press Keys    //input[@id='rekening']    CTRL+V    # paste VA number

Execute VA payment
    Click Button    inq-btn
    Wait Until Element Is Visible    //button[@id='pay']
    Click Button    //button[@id='pay']
    Wait Until Element Is Visible    //span[@id='pay-status']
```

Gambar 4. Skrip pengujian pembayaran dengan *virtual account*

Test case ini dilakukan saat berada di halaman pembayaran. Langkah pengujian diawali dengan menyalin nomor *virtual account*. Penyalinan *virtual account* dilakukan dengan klik tombol *copy*. Selanjutnya, ATM *Simulator* dibuka pada *tab* baru dan diverifikasi elemen halamannya. Kode bank dipilih sesuai metode pembayaran yang dipilih sebelumnya dan tempel *virtual account* pada *text field* nomor rekening. Tombol *inquire* diklik dan tunggu hingga tombol Bayar muncul. Saat tombol Bayar muncul, kemudian diklik dan ditunggu status pembayaran muncul. Terakhir, sistem diarahkan untuk kembali ke halaman pembayaran. Skrip pengujian pembayaran dengan QRIS dapat dilihat pada Gambar 5.

```
Pay with QRIS
    Wait Until Element Is Visible
    ...    xpath=//span[contains(text(),'Status Pembayaran')]    timeout=10s
    Page Should Contain    Kode QR
    Page Should Contain Element    //*[@id="card-summary-wrap"]/div[2]/p/canvas
    ...    # qris image
```

Gambar 5. Skrip pengujian pembayaran dengan QRIS

Skrip pengujian pada Gambar 5 ini hanya dilakukan untuk verifikasi halaman pembayaran. Hal ini dikarenakan pembayaran QRIS tidak dapat diuji secara otomatis, sehingga pada skrip pengujian ini akan dilakukan pengujian secara manual yang termasuk dalam proses *hybrid testing*. Langkah pengujian pada *test case* ini diawali dengan verifikasi *element text* Status Pembayaran yang mengacu pada halaman pembayaran QRIS sudah muncul atau belum. Halaman pembayaran tersebut juga diverifikasi apakah terdapat Kode QR atau tidak. Hasil pengujian dari beberapa *test case* ditunjukkan pada Tabel 2.

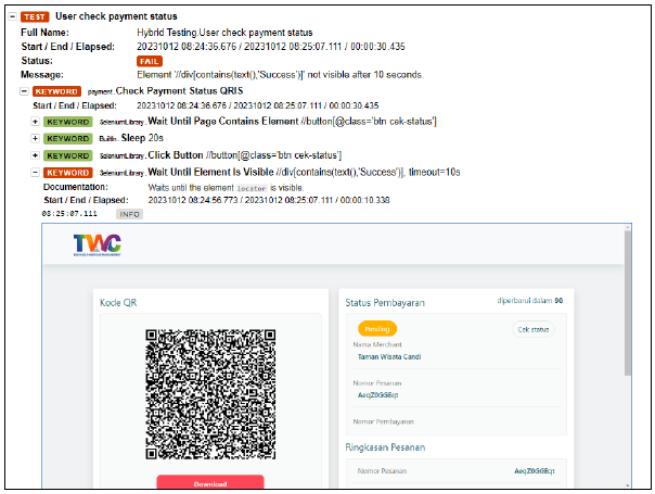
Tabel 2. Hasil Pengujian

Test case	*Manual*		*Automated*		*Hybrid*	
	Status	*Defects*	*Status*	*Defects*	*Status*	*Defects*
Pembayaran dengan QRIS	*PASS*	-	*CAN'T BE TESTED*	*Scan QRIS can't be automated*	*PASS*	-
Periksa status pembayaran setelah pembayaran sukses	*PASS*	-	*FAIL*	*The status does not change to success because there is no QRIS payment*	*PASS*	-
Lihat Tiket yang berhasil dibeli	*PASS*	-	*FAIL*	*No tickets were purchased*	*PASS*	-

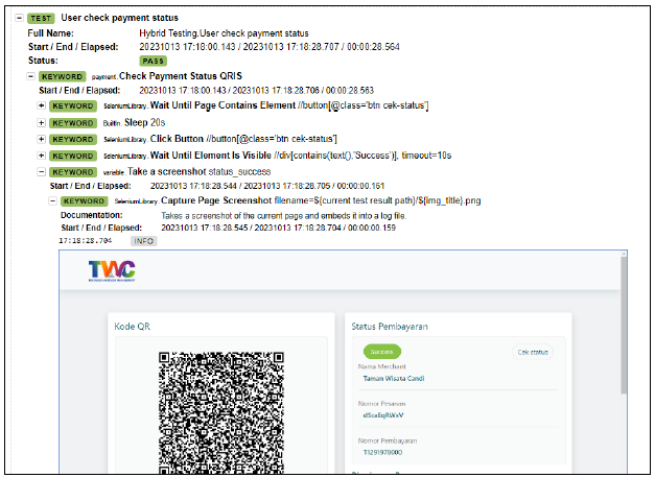
B. Hasil Pengujian

Pengujian *end-to-end* pada sistem *e-payment* Travelink dilakukan pada 57 *test cases* dengan 3 metode yaitu pengujian manual, pengujian otomatis dan pengujian *hybrid*. Tabel 2 terdiri dari data *Test ID* yang diuji, kolom Status dan *Defects* pada setiap metode eksekusi. Kolom Status memiliki 3 jenis status untuk menandai dari hasil pengujian. Status *PASS* dengan warna hijau untuk *test case* yang lolos uji, sedangkan status *FAIL* yang berwarna merah untuk *test case* yang gagal uji. Adapun status *CAN'T BE TESTED* dengan warna abu-abu yang artinya *test case* tidak dapat dieksekusi dan gagal. Kolom *Defects* digunakan untuk menuliskan keterangan penyebab *test case* tidak lolos uji.

Tabel 2 merupakan hasil pengujian dari beberapa *test case* yang menunjukkan hasil yang signifikan. Berdasarkan tabel tersebut, dapat dilihat bahwa pada pengujian secara manual dan *hybrid*, semua *test case* berhasil dieksekusi dan lolos uji. Hasil pengujian secara *automated* terdapat 54 *test cases PASS* atau lolos uji, dua *test cases FAIL* atau gagal uji dan satu *test case* tidak dapat dieksekusi. Satu *test case* yang tidak dapat dieksekusi terjadi pada *test case* dengan id PYM_04 yaitu pengujian pembayaran QRIS. Pengujian pada pembayaran QRIS membutuhkan sistem terpisah yaitu *mobile app* untuk melakukan pembayaran, sehingga tidak bisa diuji secara otomatis dan berujung gagal uji. Kemudian *test case* dengan id PYM_05 yang menguji status pembayaran setelah pembayaran, tidak lolos uji karena saling berkaitan dengan *test id* PYM_04. Status pembayaran tidak berubah menjadi sukses karena tidak ada proses pembayaran. *Test id* PYM_06 juga berkaitan dengan *test case* dengan id PYM_05, pengujian terhadap tiket yang berhasil dibeli, gagal uji karena tidak ada tiket yang dibeli. Hasil pengujian *test case* yang gagal uji dapat dilihat pada Gambar 6, sedangkan untuk hasil pengujian *test case* yang lolos uji dapat dilihat Gambar 7.



Gambar 6. *Failed test report*



Gambar 7. *Success test report*

Tabel 3. *Requirement Traceability Matrix*

No.	*Requirements*	*Test ID*	*Manual Testing Result*	*Automated Testing Result*	*Hybrid Testing Result*
1.	*Payment Page* – Pengguna dapat melihat tata cara pembayaran sesuai metode yang dipilih. Sistem akan mengarahkan pengguna untuk mengisi akun kartu kredit jika memilih metode tersebut. Selain kartu kredit, pengguna melakukan pembayaran di luar sistem.	PYM_02 (VA) PYM_03 (CC) PYM_04 (QRIS)	*Covered*	*Not Covered*	*Covered*
2.	Cek Status Pembayaran – Pengguna dapat melihat dan memperbarui status pembayaran. Pengguna dapat melihat tiket yang berhasil dibeli.	PYM_05 PYM_06 HST_17 HST_18 HST_19	*Covered*	*Not Covered*	*Covered*

C. Analisis Perbandingan Metode Eksekusi Pengujian

1. *Requirement Coverage*

Pengujian yang telah dilakukan menghasilkan data yang dapat diolah menjadi bahan perbandingan metode eksekusi pengujian. Perbandingan pertama menggunakan data *Requirement Coverage* yang berasal dari *Requirement Traceability Matrix* atau RTM. Tabel 3 menampilkan beberapa *requirements* dari total 15 *requirements* pada sistem Travelink yang telah dilakukan pengujian *end-to-end* secara manual, otomatis dan *hybrid*. RTM yang dibuat pada penelitian ini terdiri dari data *requirements system*, *test case* yang mencakup *requirements system*, dan status *requirement* dari hasil eksekusi tiap metode pengujian. *Requirement* yang dituliskan merupakan dokumen *user acceptance criteria* atau UAC. UAC berasal dari kebutuhan sistem secara bisnis yang digunakan sebagai acuan dalam pengembangan dan pengujian sistem. Status *requirement* terdiri dari *Covered* dan *Not Covered*. Status *Covered* diberikan pada *requirement* yang berhasil dalam pengujian *test case* yang mencakup *requirement* tersebut. Sedangkan t*est cases* yang terdapat kegagalan atau *bug* setelah pengujian, maka *requirement* terkait diberi status *Not covered*.

Berdasarkan Tabel 3 diperoleh informasi bahwa terdapat 2 *requirements* tidak ter-*cover* pada pengujian otomatis. Dua *requirements* yang tidak ter-*cover* adalah pembayaran QRIS dan cek status pembayaran QRIS. Hal ini dikarenakan *test case* pembayaran QRIS tidak dapat diuji secara otomatis. *Test case* cek status pembayaran mengalami gagal uji karena tidak ada pembayaran yang berhasil, sehingga pengujian otomatis tidak dapat mencakup 2 *requirements* tersebut. Pengujian *end-to-end* secara manual dan *hybrid* semua *requirement* tercakup oleh *test case* yang dieksekusi, baik itu *test case Credit Card, Virtual Account*, maupun QRIS. *Test case Credit Card* dan *Virtual Account* yang dilakukan dengan pengujian otomatis mampu mencakup seluruh *requirement*. Data dari masing-masing metode pengujian tersebut kemudian dihitung menggunakan Persamaan (1) seperti berikut.

$$\text{Requirement Coverage (RC)}\% = \frac{NRC}{TNR} \times 100\% \quad (1)$$

a. RC *Manual testing:* $\frac{15}{15}$ x 100% = 100%

b. RC Automated testing: $\frac{13}{15}$ x 100% = 87%

c. RC *Hybrid testing*: $\frac{15}{15}$ x 100% = 100%

4. Waktu Eksekusi

Waktu eksekusi dihitung dengan Persamaan (2), dan waktu eksekusi pengujian manual ditampilkan pada Tabel 4.

$$\text{Rata waktu eksekusi manual test} = \frac{total\ waktu\ eksekusi\ manual\ test}{total\ responden} \quad (2)$$

Tabel 4. Waktu eksekusi pengujian manual (detik)

	End-to-end test case		
	TC 1 (CC)	**TC 2 (VA)**	**TC 3 (QRIS)**
Responden 1	1.075	817	925
Responden 2	661	727	802
Responden 3	850	906	681
Rata-rata	862	816,677	802,677

Perbandingan metode eksekusi pengujian pada penelitian ini juga menggunakan waktu eksekusi pengujian. Tabel 4

menampilkan data rata-rata waktu eksekusi pengujian manual dengan menggunakan Persamaan (2). Metode eksekusi pengujian manual dilakukan oleh responden berjumlah tiga orang. Masing-masing responden menguji tiga *end-to-end test case* (CC, VA, dan QRIS). Berdasarkan data dari tabel tersebut, waktu eksekusi setiap *end-to-end test case* dijumlah dan dibagi dengan total responden, sehingga didapatkan rata-rata waktu eksekusi. Rata-rata waktu eksekusi pengujian manual kemudian digunakan untuk bahan perbandingan dengan metode eksekusi yang lain.

Berdasarkan Tabel 5, pengujian *end-to-end test case* dengan pembayaran *Credit Card* dan *Virtual Account* paling cepat jika dilakukan dengan metode pengujian otomatis. Perbedaan waktu eksekusi antara otomatis dan *hybrid* tidak jauh berbeda dibandingkan dengan manual. Pengujian pada *end-to-end test case* dengan pembayaran QRIS paling cepat jika dilakukan dengan metode pengujian otomatis. Hal ini disebabkan oleh jumlah *test case* yang dapat dieksekusi lebih sedikit dan terdapat *test case* yang gagal, sehingga pengujian secara otomatis berjalan lebih cepat namun tidak lolos uji. Semua *test case* QRIS lolos uji pada metode *hybrid* dan manual. Metode *hybrid* lebih cepat 599,626 detik dibandingkan dengan manual.

5. Analisis Berdasarkan Matriks *Requirement Coverage* dan Waktu Eksekusi

Berdasarkan dari perbandingan dengan matriks pengujian *requirement coverage*, metode pengujian manual dan *hybrid* memberikan cakupan pengujian 100% pada semua *end-to-end test case*, sedangkan pengujian otomatis pada *test case* QRIS menghasilkan cakupan pengujian 87%. Adapun dua *requirement* yang tidak ter-*cover* terjadi pada pembayaran QRIS dan cek status pembayaran QRIS. Hal ini menunjukkan bahwa pengujian *end-to-end test case Credit Card, Virtual Account,* dan QRIS yang dilakukan secara manual dan *hybrid* mampu mencakup semua *requirements*. Pengujian otomatis dapat mencakup semua *requirements* dengan kasus pengujian *end-to-end Credit Card* dan *Virtual Account*.

Berdasarkan dari perbandingan dengan matriks pengujian waktu eksekusi pengujian, pengujian *end-to-end test case* pembayaran *Credit Card* dan *Virtual Account* paling cepat dilakukan dengan metode pengujian secara otomatis. Pengujian secara otomatis juga terlihat paling cepat pada *endto-end test case* pembayaran QRIS, namun hal itu disebabkan oleh beberapa *test case* gagal uji. Berdasarkan perbandingan antara manual dan *hybrid*, metode pengujian *hybrid* lebih cepat dan efektif dilakukan untuk *end-to-end test case* pembayaran QRIS.

Analisis dari perbandingan di atas hanya menggunakan waktu saat eksekusi pengujian saja, sedangkan pada pengujian otomatis dan *hybrid* memerlukan waktu yang lama untuk pembuatan kode program. Pembuatan kode program yang dilakukan juga membutuhkan wawasan terkait *automation tools* yang digunakan. Hal ini menunjukkan bahwa eksekusi pengujian otomatis dan *hybrid* dapat berjalan dalam waktu yang singkat, namun membutuhkan waktu yang relatif lama saat pembuatan kode program. Hasil perbandingan dari kedua matriks pengujian dapat dilihat pada Tabel 5.

Tabel 5. Perbandingan berdasarkan matriks *Requirement Coverage* dan Waktu Eksekusi

Metode dan matriks pengujian		***E2E Test case***		
		TC 1 (CC)	**TC 2 (VA)**	**TC 3 (QRIS)**
Manual	*Requirement Coverage*	100%	100%	100%
	Lama Eksekusi (detik)	1.075	817	925
Otomatis	*Requirement Coverage*	100%	100%	87%
	Lama Eksekusi (detik)	661	727	802
Hybrid	*Requirement Coverage*	100%	100%	100%
	Lama Eksekusi (detik)	850	906	854

V. SIMPULAN

Berdasarkan sejumlah *defects* yang ditemukan pada pengujian *end-to-end* QRIS yang dilakukan secara otomatis menggunakan Robot Framework, menghasilkan persentase *test coverage*, yaitu *Requirement Coverage* sebesar 87%. Selain pengujian tersebut, metode pengujian manual dan hybrid menghasilkan *Requirement Coverage* sebesar 100%.

Berdasarkan analisis perbandingan dari waktu eksekusi pengujian dihasilkan bahwa metode pengujian otomatis adalah metode yang paling cepat untuk semua *test case*. Namun, metode pengujian otomatis pada *test case end-to-end* QRIS tidak mencakup 100% *test coverage*, maka dapat disimpulkan bahwa metode pengujian hybrid adalah metode yang paling cepat dan sesuai untuk *test case end-to-end* QRIS. Sedangkan metode pengujian otomatis adalah metode yang paling cepat dan sesuai untuk *test case end-to-end Credit Card* dan *Virtual Account*.

REFERENSI

[1] T. P. Nagarhalli, V. Vaze, and N. K. Rana, "A Review of Current Trends in the Development of Chatbot Systems," in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India: IEEE, Mar. 2020, pp. 706–710. doi: 10.1109/ICACCS48705.2020.9074420.

[2] K. Kaur and A. Pathak, "E-Payment System on E-Commerce in India," 2015. [Online]. Available: www.ijera.com

[3] M. Y. M. Al-Sabaawi, A. A. Alshaher, and M. A. Alsalem, "User trends of electronic payment systems adoption in developing countries: an empirical analysis," *Journal of Science and Technology Policy Management*, vol. 14, no. 2, pp. 246–270, Mar. 2023, doi: 10.1108/JSTPM-11-2020-0162.

[4] M. Masihuddin, B. Ul, I. Khan, M. Mueen, U. I. Mattoo, and R. F. Olanrewaju, "A Survey on E-Payment Systems: Elements, Adoption, Architecture, Challenges and Security Concepts," *Indian J Sci Technol*, vol. 10, no. 20, 2017, doi: 10.17485/ijst/2017/v10i20/113930.

[5] D. Manova, D. Petrova-Antonova, and S. Ilieva, *TASSA Methodolgy: End-to-End Testing of Web Service Compositions*. 2018.

[6] M. Leotta, D. Clerissi, F. Ricca, and P. Tonella, "Approaches and Tools for Automated End-to-End Web Testing," in *Advances in Computers*, vol. 101, Elsevier, 2016, pp. 193–237. doi: 10.1016/bs.adcom.2015.11.007.

[7] S. Di Meglio and L. L. L. Starace, "Towards Predicting Fragility in End-to-End Web Tests," in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering*, Salerno Italy: ACM, Jun. 2024, pp. 387–392. doi: 10.1145/3661167.3661179.

[8] J. Lian Min, A. Istiqomah, A. Rahmani, P. Negeri Bandung, and P. P. Tester Padepokan Tujuh Sembilan-Bandung, "EVALUASI PENGGUNAAN MANUAL DAN AUTOMATED SOFTWARE TESTING PADA PELAKSANAAN END-TO-END TESTING," *Jurnal Teknologi Terapan)* |, vol. 6, no. 1, 2020.

[9] A. Gunawan, A. F. Fatikasari, and S. A. Putri, "The Effect of Using Cashless (QRIS) on Daily Payment Transactions Using the Technology Acceptance Model," *Procedia Comput. Sci.*, vol. 227, pp. 548–556, 2023, doi: 10.1016/j.procs.2023.10.557.

[10] A. Rachman, N. Julianti, and S. Arkoyah, "Challenges and Opportunities for QRIS Implementation as a Digital Payment System in Indonesia," *EkBis J. Ekon. Dan Bisnis*, vol. 8, no. 1, pp. 1–13, Jun. 2024, doi: 10.14421/EkBis.2024.8.1.2134.

[11] M. Susanti and H. Kresnha Reza, "Added Value and Ease of Using Quick Responses Qris Indonesian Standard (QRIS)," *Int. J. Sci. Technol. Manag.*, vol. 3, no. 3, pp. 715–723, May 2022, doi: 10.46729/ijstm.v3i3.518.

[12] G. A. Jasmin and D. G. P. Putri, "Komparasi Metode Automasi dan Hybrid pada Pengujian Aplikasi Mobile WebRTC Menggunakan Appium," *J. Internet Softw. Eng.*, vol. 5, no. 1, pp. 29–36, May 2024, doi: 10.22146/jise.v5i1.9034.

[13] M. N. Zafar, W. Afzal, and E. Enoiu, "Evaluating System-Level Test Generation for Industrial Software: A Comparison between Manual, Combinatorial and Model-Based Testing," in *Proceedings - 3rd ACM/IEEE International Conference on Automation of Software Test, AST 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 148–159. doi: 10.1145/3524481.3527235.

[14] J. Islam, "Undergraduate Thesis Subject: Smart E-commerce Website with Digital payment," 2023, doi: 10.13140/RG.2.2.14218.24004.

Rancang Bangun Aplikasi Monitor Kadar Gula Darah Berbasis *Mobile*

Arief Zuhri[1], Margareta Hardiyanti[1], Norma Latif Fitriyani[2], Ganjar Alfian[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
ariefzuhri@mail.ugm.ac.id
margareta.hardiyanti@ugm.ac.id
[2]Department of Data Science, Sejong University;
norma@sejong.ac.kr
[*]Korespondensi: ganjar.alfian@ugm.ac.id;

Abstract – Diabetes is a serious public health condition and one of the leading causes of mortality worldwide. The goal of diabetes treatment is to prevent or delay complications and optimize quality of life. The American Diabetes Association (ADA) outlines the multifactorial approach to reducing the risk of diabetes complications through lifestyle changes and education. Mobile-based diabetes management applications have been proven to assist individuals successfully manage diabetes self-care. Although existing apps offer significant benefits, there is still scope for further advancement in terms of the efficiency of diabetes management and the sustainability of system development. This study designs a mobile-based blood glucose monitoring system based on the multifactorial approach. The technology stack was based on Google's Modern Android Development (MAD) approach. The development stages include analysis, design, implementation, and testing. The implementation applies six components: targeting the latest Android version, using Android Studio, Kotlin, Jetpack Compose, Android Jetpack, and adopting best practices for architecture and testing. The application was successfully built with two main features: blood glucose monitoring and diabetes treatment recommendations. The monitoring feature presents blood glucose charts, activity logs that may affect glucose levels, and blood sugar alarms. The diabetes treatment recommendation feature provides education and encourages patients to adopt a healthy lifestyle. The use of MAD resulted in a scalable and maintainable system. Testing showed the app functions as designed, with no detected malfunctions. This study is expected to assist patients achieve the diabetes treatment goals and ensure the system remains relevant and impactful for the long term.

Keywords – Diabetes Management, Monitoring System, Mobile Application, Multifactorial Approach, Modern Android Development

Intisari – Diabetes merupakan kondisi serius pada kesehatan masyarakat serta menjadi salah satu penyebab mortalitas tertinggi secara global. Tujuan perawatan diabetes adalah mencegah atau menunda komplikasi dan mengoptimalkan kualitas hidup. American Diabetes Association (ADA) menjabarkan pendekatan multifaktorial untuk mengurangi risiko komplikasi diabetes yang diterapkan melalui perubahan gaya hidup dan edukasi. Manajemen diabetes melalui aplikasi berbasis *mobile* terbukti membantu individu dalam keberhasilan melakukan manajemen diabetes secara mandiri. Meskipun aplikasi yang ada telah memberikan manfaat signifikan, masih terdapat ruang untuk pengembangan lebih lanjut dalam hal efektivitas manajemen diabetes dan keberlanjutan pengembangan sistem. Pada penelitian ini, sistem monitor kadar gula darah berbasis *mobile* dirancang berdasarkan pendekatan multifaktorial. Teknologi pengembangan mengadopsi pendekatan *Modern Android development* (MAD) dari Google. Tahapan pengembangan mencakup analisis, perancangan, implementasi, dan pengujian. Implementasi menerapkan enam komponen MAD: penargetan versi Android terbaru, penggunaan Android Studio, Kotlin, Jetpack Compose, Android Jetpack, dan penerapan praktik terbaik untuk arsitektur dan pengujian. Aplikasi berhasil dibangun dengan dua fitur utama: monitor kadar gula darah dan rekomendasi perawatan diabetes. Fitur monitor menyajikan grafik gula darah, pencatatan aktivitas yang dapat memengaruhi gula darah, dan alarm gula darah. Adapun rekomendasi perawatan diabetes memberikan edukasi dan dorongan kepada pasien untuk menerapkan gaya hidup sehat. Penerapan MAD menghasilkan sistem yang skalabel dan mudah dipelihara. Dari hasil pengujian, fungsi aplikasi berjalan sesuai dengan hasil analisis dan perancangan serta tidak ditemukan kerusakan. Hasil penelitian ini diharapkan agar sistem dapat membantu pasien dalam mencapai tujuan perawatan diabetes. Selain itu, sistem diharapkan dapat terus dikembangkan untuk memastikan relevansinya dan dampak yang berkelanjutan.

Kata kunci – Manajemen Diabetes, Sistem Monitor, Aplikasi *Mobile*, Pendekatan Multifaktorial, *Modern Android Development*

I. PENDAHULUAN

Diabetes melitus (diabetes) merupakan kondisi serius pada kesehatan masyarakat serta menjadi salah satu penyebab mortalitas tertinggi secara global. Tercatat 536,6 juta orang dewasa (usia 20-79 tahun) di seluruh dunia hidup dengan diabetes. Kondisi ini menyebabkan orang yang hidup dengan diabetes menjadi rentan terhadap risiko yang membahayakan hidup. Lebih-lebih, data menunjukkan bahwa diabetes telah menyebabkan 6,7 juta kematian orang dewasa di seluruh dunia pada tahun 2021 [1].

Tujuan perawatan diabetes adalah menunda atau mencegah komplikasi dan mengoptimalkan kualitas hidup [2]. Beberapa komplikasi diabetes adalah penyakit kardiovaskular aterosklerosis (ASCVD) [3], penyakit ginjal kronis (CKD) [4], retinopati diabetik, neuropati diabetik, serta ulserasi dan amputasi kaki [5]. Untuk mencegah komplikasi, penting untuk memantau gula darah secara teratur [3], [4], [5], [6]. Teknologi pemantauan termutakhir dapat menggunakan sensor CGM (*continuous glucose monitoring*) yang terbukti efektif dan aman dalam manajemen diabetes [6]. Sensor CGM ditempatkan pada bawah kulit, dapat di perut atau lengan,

kemudian sensor akan mengukur gula darah secara kontinu dan menampilkan hasilnya setiap 5 menit [7].

American Diabetes Association (ADA) menjabarkan pendekatan multifaktorial untuk mengurangi risiko komplikasi diabetes yang terdiri atas komponen-komponen dasar dalam manajemen diabetes dan diterapkan melalui perubahan gaya hidup dan edukasi diabetes. Komponen-komponen dasar dalam manajemen diabetes mencakup manajemen gula darah, manajemen tekanan darah, manajemen kolesterol, dan penggunaan obat penurun kadar gula darah dengan manfaat jantung dan ginjal. Komponen-komponen tersebut merupakan pilar dalam mengontrol risiko komplikasi diabetes yang diterapkan melalui perubahan gaya hidup serta edukasi dan dukungan manajemen mandiri [3].

Implementasi sistem monitor kadar gula darah ke dalam aplikasi berbasis *mobile* telah dilakukan pada penelitian-penelitian sebelumnya [8], [9], [10], [11], [12], [13], [14], [15], [16]. Hasil penelitian tersebut menunjukkan bahwa sistem dapat mengedukasi pasien diabetes menuju keberhasilan dalam melakukan manajemen diabetes secara mandiri [9], [10], [11], [12], [13], [15], [16]. Pasien diabetes menjadi lebih sadar terhadap kondisi gula darah mereka dan termotivasi untuk menerapkan gaya hidup sehat demi menjaga kadar gula darah [10], [11], [12].

Akan tetapi, meskipun aplikasi-aplikasi yang ada telah memberikan manfaat signifikan, masih terdapat ruang untuk pengembangan lebih lanjut dalam hal efektivitas manajemen diabetes dan keberlanjutan pengembangan sistem. Belum ada sistem monitor kadar gula darah yang dirancang dengan menerapkan pendekatan multifaktorial. Di sisi teknis, aplikasi-aplikasi tersebut masih dikembangkan tanpa mengikuti pedoman Google.

Oleh karena itu, penelitian ini bertujuan untuk merancang dan membangun sistem monitor kadar gula darah berbasis *mobile* yang pengembangan fiturnya berdasarkan rekomendasi ADA untuk mengurangi risiko komplikasi diabetes, yakni pendekatan multifaktorial. Sistem juga akan menerapkan pendekatan *Modern Android development* (MAD) dari Google. MAD merupakan alat, pustaka, dan praktik terbaik yang direkomendasikan oleh pakar Android untuk membangun aplikasi secara efisien dan menghasilkan aplikasi dengan pengalaman pengguna dan skalabilitas yang tinggi [17].

Dengan demikian, penerapan pendekatan multifaktorial pada sistem monitor kadar gula darah berbasis *mobile* diharapkan dapat membantu pasien diabetes dalam manajemen diabetes yang lebih efektif sehingga dapat mencapai tujuan perawatan. Sementara itu, penerapan MAD diharapkan dapat meningkatkan keberlanjutan pengembangan sistem untuk perluasan fungsi dan pemeliharaan sistem.

Perancangan sistem ini akan memanfaatkan keunggulan dari sensor CGM. Namun, penelitian ini tidak membahas terkait integrasi aplikasi dengan sensor CGM. Selain itu, penelitian ini juga tidak untuk membuktikan keunggulan dari pendekatan MAD. Penelitian ini hanya akan fokus pada penerapan gagasan pendekatan multifaktorial dan MAD.

II. DASAR TEORI

Implementasi sistem monitor kadar gula darah berbasis *mobile* telah dilakukan pada penelitian sebelumnya. Hasil penelitian tersebut menunjukkan bahwa aplikasi *mobile* untuk monitor kadar gula darah dapat bermanfaat bagi pasien diabetes.

Penelitian [8], misalnya, merancang sistem monitor kadar gula darah kontinu menggunakan sensor dan aplikasi Android yang menampilkan data secara *real-time* serta alarm gula darah. Sistem ini mempermudah pemantauan gula darah secara *real-time* sehingga membantu pasien dalam melakukan perawatan.

Penelitian yang lebih baru, penelitian [9], merancang aplikasi Android untuk manajemen diabetes dengan fitur yang lebih komprehensif, seperti pengingat (pengukuran gula darah, minum obat, makan tepat waktu, dan berolahraga), pencatatan dan grafik gula darah, kalkulator dosis insulin, dan cetak laporan. Aplikasi ini dirancang berdasarkan data protokol Sri Lanka, Application-Based Care in Diabetes. Berdasarkan pengujian, sistem terbukti dapat membantu pasien dalam mengontrol gula darah dengan lebih baik.

Lebih jauh lagi, penelitian [10] merancang aplikasi Android yang difokuskan untuk remaja dengan diabetes tipe 1. Aplikasi dilengkapi dengan pencatatan manual gula darah, pengingat pengukuran gula darah, mengatur target gula darah, serta video dan materi edukasi. Aplikasi ini dirancang melalui proses kelompok fokus dan tinjauan ahli. Meskipun ada beberapa masalah teknis, aplikasi tersebut berhasil memotivasi remaja untuk mencatat gula darah secara rutin.

Sementara itu, penelitian [11] merancang aplikasi Android untuk manajemen mandiri bagi pasien diabetes tipe 1 dan 2 di Australia. Aplikasi memiliki fitur pencatatan gula darah, aktivitas, dan berat badan beserta grafik-grafiknya, pesan motivasi dan saran, dan materi edukasi makanan. Perancangan aplikasi melalui tahapan identifikasi kebutuhan pengguna dan masukan pakar, kemudian dibangun menggunakan Java. Hasil uji juga menunjukkan kepuasan pengguna serta peningkatan kesadaran pasien dalam manajemen mandiri.

Di sisi lain, penelitian [12] merancang aplikasi Android untuk manajemen diabetes yang dapat membagikan statistik medis melalui *web service* dan terhubung dengan dokter serta memiliki fitur manajemen asupan nilai gizi. Fitur lain yang tersedia adalah pemantauan gula darah, visualisasi data, kalkulator insulin, dan pengingat pengukuran gula darah, makan, dan insulin. Aplikasi ini dirancang melalui analisis aplikasi serupa. Hasil uji menunjukkan hasil yang sama, yakni

peningkatan kontrol gula darah pada pasien dan mendorong anak-anak melakukan manajemen mandiri.

Seperti penelitian sebelumnya, penelitian [13] juga merancang aplikasi Android untuk remaja dengan diabetes tipe 1 yang mendukung manajemen mandiri. Aplikasi memiliki fitur, di antaranya, pencatatan dan grafik gula darah, pencatatan dan pengingat pengobatan, dan edukasi diet, olahraga, dan komplikasi. Aplikasi ini dirancang berdasarkan tujuh perilaku perawatan diri AADE (American Association of Diabetes Educators) dan studi pustaka. Aplikasi dibangun menggunakan Ionic dan ChartJS. Pengujian juga menunjukkan bahwa aplikasi dapat membantu remaja meningkatkan manajemen mandiri.

Serupa juga dengan penelitian yang telah dilakukan, penelitian [14] merancang aplikasi Android dan *cloud* untuk pasien diabetes tipe 2 yang memungkinkan pasien mendapatkan layanan pemantauan jarak jauh oleh dokter. Fitur lainnya mencakup pemantauan gula darah beserta grafik, pencatatan gula darah, pengobatan, diet, aktivitas fisik, tekanan darah, dan berat badan, cetak laporan, artikel edukasi secara umum, dan pengingat. Aplikasi dirancang berdasarkan analisis aplikasi sejenis, pendapat pakar, dan tinjauan sumber daya daring dari Iranian Ministry of Health and Medical Education, Association of Diabetes Care & Education Specialists, International Diabetes Federation, dan American Diabetes Association. Aplikasi ditargetkan untuk Android 2.3 sampai 4.4. Hasil uji kegunaan menunjukkan kepuasan pengguna, menandakan bahwa sistem dapat membantu pasien mengubah gaya hidup sehat.

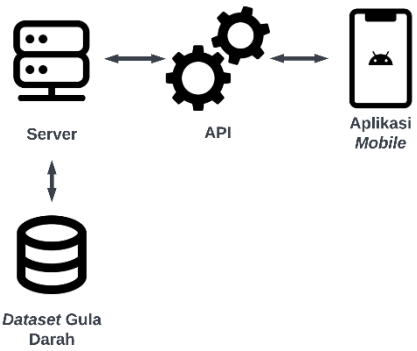
Berbeda dengan penelitian sebelumnya, penelitian [15] merancang aplikasi monitor mandiri berbasis Android bagi pasien diabetes di Malaysia yang berfokus pada manajemen pola makan. Aplikasi memiliki fitur pemantauan kadar gula darah berdasarkan pencatatan manual, pencatatan makanan, pengingat jadwal janji temu, dan kalkulator dosis insulin berdasarkan jumlah karbohidrat. Manfaat aplikasi bagi pasien, di antaranya, dapat meningkatkan kontrol gula darah dan kepatuhan terhadap pola makan.

Adapun penelitian terbaru, penelitian [16], merancang aplikasi Android untuk perawatan mandiri bagi pasien diabetes tipe 2 dengan fitur umpan balik, pemantauan gula darah beserta grafik, pemantauan berat badan, pengecekan tekanan darah, saran dosis insulin, edukasi, evaluasi makanan oleh ahli gizi, penghitung langkah, penyiapan laporan, dan pengingat. Aplikasi ini dirancang berdasarkan analisis aplikasi serupa. Aplikasi dibangun menggunakan Android Studio 2.3.3, Java, SQLite, dan Android View serta ditargetkan untuk Android 4.4. Aplikasi ini diharapkan dapat meningkatkan gaya hidup dan memotivasi pasien diabetes dalam melakukan perawatan diabetes.

Penelitian-penelitian sebelumnya telah menghasilkan aplikasi monitor kadar gula darah yang terbukti bermanfaat dalam membantu pasien mengontrol gula darah dan meningkatkan manajemen mandiri. Sistem-sistem yang telah diusulkan memiliki fitur-fitur dasar, seperti pemantauan dan pencatatan, visualisasi data, pengingat, serta edukasi dan motivasi. Namun, tidak ada penelitian yang secara eksplisit menerapkan pendekatan multifaktorial dari ADA yang mencakup manajemen komprehensif gula darah, tekanan darah, kolesterol, serta perubahan gaya hidup. Selain itu, dari segi teknologi pengembangan, penelitian-penelitian tersebut seluruhnya menggunakan metode pengembangan konvensional, seperti Java dan *framework* lain, yang belum mengadopsi praktik-praktik terbaik dari MAD. Oleh karena itu, penelitian ini berfokus pada penerapan pendekatan multifaktorial serta MAD yang diharapkan dapat meningkatkan kualitas aplikasi dari segi pengalaman pengguna dan kemudahan pengembangan.

III. METODOLOGI

Arsitektur sistem dalam penelitian ini ditunjukkan pada Gambar 1. *Dataset* gula darah dihimpun di sebuah basis data pada suatu server. Untuk mengambil data gula darah dari aplikasi *mobile*, API berperan sebagai proksinya.



Gambar 1. Arsitektur sistem monitor kadar gula darah

Penelitian ini menggunakan sumber data gula darah yang diperoleh dari *dataset* publik hasil penelitian sebelumnya [18]. *Dataset* tersebut dipublikasikan oleh Jaeb Center for Health Research (JCHR) [19]. *Dataset* digunakan untuk mendemonstrasikan fungsionalitas aplikasi

Adapun rancang bangun aplikasi melalui empat tahapan, yakni analisis, perancangan, implementasi, dan pengujian. Masing-masing tahapan dijabarkan sebagai berikut.

A. Analisis

Pendekatan multifaktorial dianalisis dan hasilnya diekstraksi menjadi fitur-fitur aplikasi. Bagaimana detail dan batasan dari suatu fitur ditentukan berdasarkan tinjauan pedoman perawatan diabetes yang dipublikasikan oleh ADA [20], [21], [22], [23], [24], [25], [26], [27]. Beberapa pedoman pendukung yang dipaparkan oleh American Heart Association (AHA) [28], [29] dan World Health Organization (WHO) [30] dalam kaitannya dengan komplikasi diabetes juga dipergunakan sebagai pedoman pemberian rekomendasi.

Adapun fitur utama aplikasi ditunjukkan pada Gambar 2, yakni terdiri atas **monitor gula darah** yang mencakup grafik

gula darah, pencatatan aktivitas, dan alarm gula darah serta **rekomendasi perawatan diabetes** yang mencakup rekomendasi berdasarkan gula darah, tekanan darah, dan BMI.

Gambar 2. Fitur utama aplikasi

Pada fitur grafik gula darah, akan menampilkan data gula darah dari waktu ke waktu dalam diagram garis. Terdapat juga statistik yang menampilkan nilai terakhir, rata-rata, nilai tertinggi, dan nilai terendah dari data pada grafik. Grafik juga menampilkan diagram TIR (*time in range*) menggunakan diagram lingkar. Definisi TIR ditunjukkan pada persamaan (1) [6].

$$TIR = \frac{\Sigma Kadar\ gula\ darah\ dlm.jangkauan}{\Sigma Kadar\ gula\ darah\ semua\ kategori} \times 100\% \quad (1)$$

Grafik gula darah juga akan diintegrasikan dengan data aktivitas yang dicatat oleh pasien. Aktivitas yang dapat dicatat oleh pengguna adalah aktivitas yang dapat memengaruhi kenaikan atau penurunan gula darah, seperti makan, olahraga, dan konsumsi obat [23]. Pada aktivitas makan, pasien dapat melihat angka perubahan kadar gula darah dari makanan yang dikonsumsi berdasarkan *2-hour postprandial glucose test* [6].

Kemudian, pada fitur alarm gula darah, akan menampilkan peringatan hipo/hiperglikemia dalam bentuk notifikasi. Alarm gula darah dapat membantu pasien untuk mengambil tindakan pencegahan secara cepat sebelum kadar gula darah menjadi terlalu tinggi atau terlalu rendah [26], [31], [32]. Alarm pada dasarnya memonitor hasil pengukuran gula darah secara *real-time* dari sensor CGM. Alarm akan memiliki level peringatan berdasarkan klasifikasi kategori gula darah oleh ADA [6]. Selain itu, batas bawah dan batas atas alarm diatur pada angka 75 mg/dL dan 170 mg/dL berdasarkan penelitian [33] yang merupakan batas optimal untuk meminimalkan kejadian hipo/hiperglikemia.

Berikutnya, pada fitur perawatan diabetes, terdapat rekomendasi berdasarkan gula darah, tekanan darah, dan BMI. Adapun faktor pemicu meningkatnya risiko komplikasi diabetes adalah gula darah, tekanan darah, dan kolesterol yang tidak terkontrol [3], [4], [5]. Pada kontrol kolesterol, gaya hidup yang diterapkan pada kontrol tekanan darah juga dapat bermanfaat untuk mengontrol kolesterol [3], [28]. Selain itu, kontrol kolesterol juga dapat dilakukan dengan mengontrol gula darah [3]. Kemudian terkait BMI, menjaga berat badan ini juga penting untuk dilakukan karena berat badan berkaitan dengan penyebab diabetes dan meningkatnya risiko komplikasi ASCVD [34]. Selain itu, BMI juga bisa memberikan gambaran secara keseluruhan dari kondisi kesehatan seseorang. Untuk mencari nilai BMI, ditunjukkan pada persamaan (2) [30].

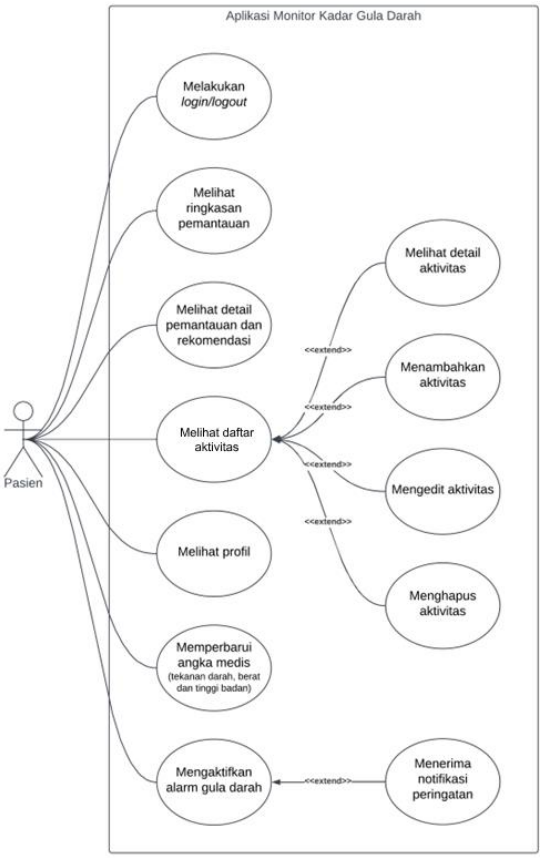
$$BMI = \frac{Berat\ badan\ (kg)}{Tinggi\ badan \times tinggi\ badan\ (m^2)} \quad (2)$$

Adapun di sisi teknis, perancangan aplikasi didasarkan pada pendekatan *Modern Android development* (MAD). Pendekatan MAD dianalisis berdasarkan tinjauan dokumentasi yang dipublikasikan oleh Google [35]. Pendekatan ini terdiri atas enam komponen, yakni penggunaan Kotlin sebagai bahasa pemrograman, penggunaan Android Studio sebagai IDE, penargetan versi Android terbaru, pembangunan UI menggunakan Jetpack Compose, penerapan *modern app architecture* (arsitektur berlapis) dan pengujian otomatis, serta penggunaan pustaka Android Jetpack [17].

B. Perancangan

Hasil dari tahapan analisis selanjutnya diolah untuk divisualisasikan ke dalam diagram sehingga dapat tergambar bagaimana nantinya aplikasi akan diimplementasikan.

Pada Gambar 3 menunjukkan rancangan *use case diagram* yang menggambarkan fungsionalitas yang tersedia pada aplikasi.



Gambar 3. Rancangan *use case diagram*

Adapun Gambar 4 menunjukkan rancangan arsitektur aplikasi *mobile* yang terdiri atas 3 lapisan. Pertama, lapisan UI yang bertanggung jawab untuk menyajikan data kepada pengguna. Kedua, lapisan data yang bertanggung jawab untuk menghubungkan aplikasi dengan layanan penyedia data. Ketiga atau terakhir, lapisan domain yang berisi logika bisnis dan menghubungkan antara lapisan UI dengan lapisan data.

Gambar 4. Arsitektur aplikasi *mobile*

C. Implementasi

Tahapan implementasi mengonversikan hasil analisis dan perancangan ke dalam bahasa pemrograman. Sumber kode bersifat terbuka dan tersedia di repositori [36]. Hasil dari tahapan ini adalah aplikasi yang dapat dioperasikan.

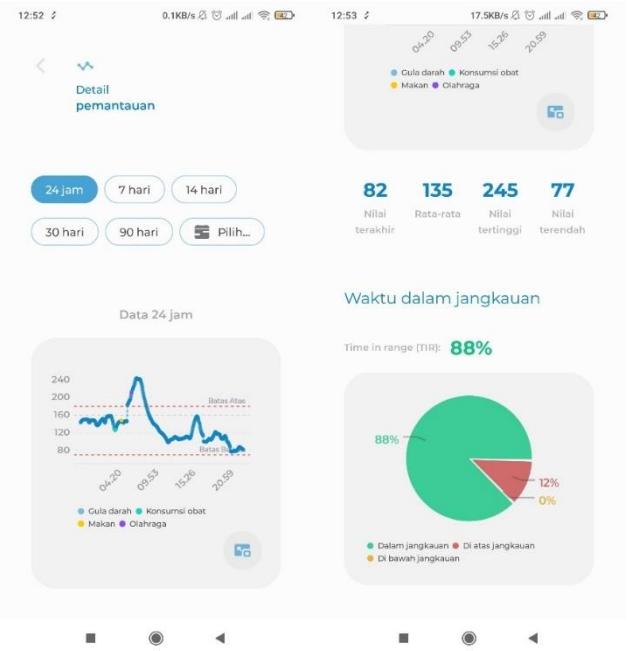
D. Pengujian

Tahapan pengujian dilakukan untuk memastikan logika bisnis yang telah diimplementasikan berjalan sesuai dengan spesifikasi kebutuhan pengguna. Adapun pengujian yang dilakukan mencakup pengujian unit dan pengujian integrasi. Pengujian unit akan menguji logika bisnis, sementara pengujian integrasi akan menguji interaksi pengguna saat menggunakan aplikasi.

IV. HASIL DAN PEMBAHASAN

A. Fitur Monitor Gula Darah

Gambar 5 menunjukkan tangkapan layar dari fitur grafik gula darah. Grafik gula darah menampilkan visualisasi data gula darah dari periode atau rentang waktu yang dipilih. Pada grafik tersebut, terdapat garis bantu batas atas dan batas bawah dari kadar gula darah yang direkomendasikan. Garis bantu ini dapat membantu pasien agar selalu memastikan kadar gula darah tidak keluar dari jangkauan aman (<70 mg/dL atau >180 mg/dL). Grafik menyertakan informasi aktivitas yang dicatat pengguna agar memudahkan pasien dan dokter dalam melihat secara langsung bagaimana pengaruh aktivitas yang dilakukan terhadap kenaikan atau penurunan kadar gula darah. Grafik juga dilengkapi dengan statistik yang memudahkan pengguna mengidentifikasi nilai terakhir, rata-rata, nilai tertinggi, dan nilai terendah dari data gula darah yang disajikan. Selain itu, grafik juga dilengkapi dengan diagram TIR yang membantu pengguna melihat capaian target TIR—melihat persentase kadar gula darah yang berada pada kategori dalam jangkauan (70-180 mg/dL), di atas jangkauan/hiperglikemia (>180 mg/dL), dan di bawah jangkauan/hipoglikemia (<70 mg/dL).

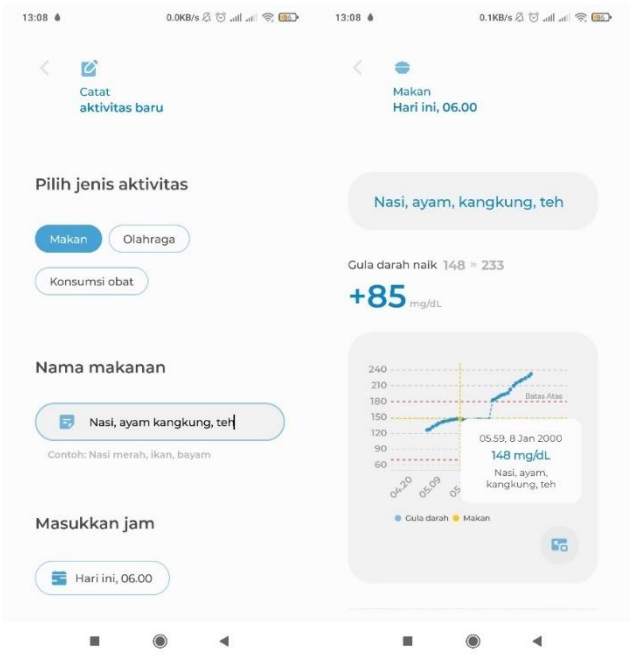


Gambar 5. Tangkapan layar grafik gula darah

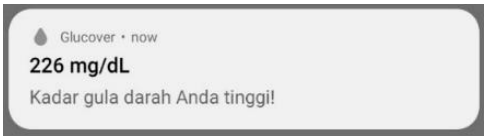
Berikutnya, Gambar 6 menunjukkan tangkapan layar dari fitur pencatatan aktivitas. Pencatatan aktivitas dapat membantu pengguna untuk melacak bagaimana gula darah mengalami kenaikan atau penurunan setelah melakukan suatu aktivitas, seperti makan, olahraga, maupun konsumsi obat atau insulin. Pengguna dapat melihat angka perubahan kadar gula darah dan grafik yang menggambarkan korelasi antara aktivitas yang dilakukan dengan perkembangan kadar gula darah sejak aktivitas mulai dilakukan hingga efek aktivitas terhadap gula darah berakhir. Pada aktivitas makan, pengguna dapat melihat bagaimana pengaruh makanan yang dikonsumsi terhadap kenaikan gula darah hingga 2 jam setelah makan.

Selanjutnya, Gambar 7 menunjukkan tangkapan layar dari notifikasi fitur alarm gula darah. Alarm gula darah dapat memberikan peringatan kejadian hipo/hiperglikemia secara *real-time* kepada pengguna dalam bentuk notifikasi. Notifikasi ditampilkan setiap 5 menit, sebagaimana CGM melakukan pembacaan gula darah setiap 5 menit. Informasi yang ditampilkan pada notifikasi berupa kadar gula darah dan pesan peringatan dari kondisi gula darah terkini. Selain itu, notifikasi dapat memberikan suara dan getaran, bergantung pada level peringatannya. Pada kadar gula darah dalam jangkauan, tidak mendekati batas atas dan batas bawah, notifikasi tidak ditampilkan menggunakan suara dan getaran. Notifikasi hanya memberikan informasi bahwa kondisi gula darah saat ini adalah aman. Notifikasi baru ditampilkan bersama suara jika kadar gula darah mendekati batas atas atau bawah sebagai bentuk peringatan awal. Dengan peringatan awal ini, diharapkan dapat membantu pasien agar segera melakukan tindakan preventif untuk mencegah kadar gula darah semakin naik atau semakin turun. Sementara itu, notifikasi dengan suara sekaligus getaran diberikan jika kadar

gula darah berada di atas dan bawah jangkauan sebagai bentuk peringatan penting bahwa kondisi gula darah terkini sudah memasuki kategori hiperglikemia atau hipoglikemia.



Gambar 6. Tangkapan layar pencatatan aktivitas



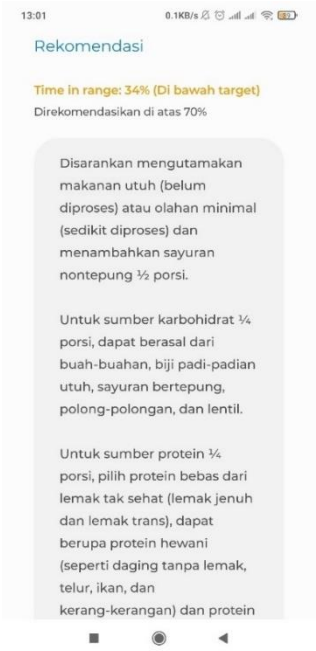
Gambar 7. Tangkapan layar notifikasi alarm gula darah

B. Fitur Rekomendasi Perawatan Diabetes

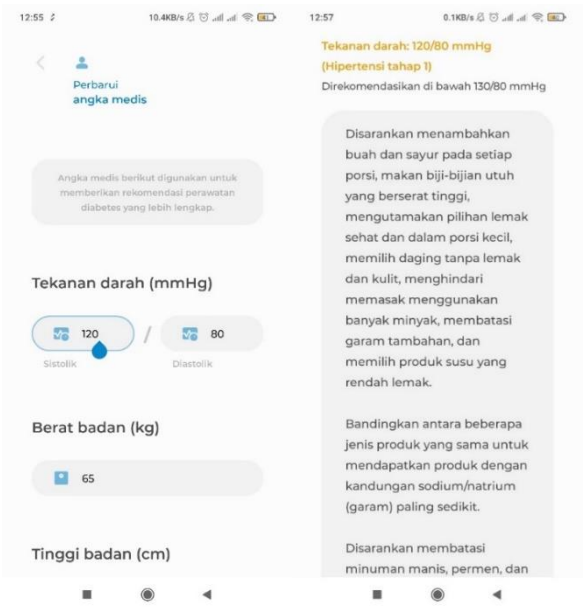
Gambar 8 menunjukkan tangkapan layar dari fitur rekomendasi perawatan diabetes berdasarkan gula darah. Rekomendasi diberikan berdasarkan nilai TIR. Karena TIR menghitung persentase kadar gula darah yang berada pada kategori dalam jangkauan, TIR dapat memberikan gambaran secara umum apakah rencana manajemen diabetes yang sedang dilakukan pasien sudah tepat. Oleh karena itu, nilai TIR dapat dijadikan dasar untuk memberikan rekomendasi gula darah. Jika pasien memiliki persentase TIR di atas 70%, pasien telah memenuhi target yang direkomendasikan dalam melakukan manajemen gula darah [6]. Adapun rekomendasi gula darah yang diberikan secara garis besar adalah menerapkan pola makan *diabetes plate method* (DPM) yang mengutamakan makanan olahan minimal dan sayuran nontepung (rendah karbohidrat) serta melakukan olahraga 150 menit/minggu. Melalui rekomendasi yang diberikan tersebut, diharapkan dapat membantu pasien untuk menjaga kadar gula darah agar berada pada kategori dalam jangkauan.

Berikutnya, Gambar 9 menunjukkan tangkapan layar dari fitur rekomendasi perawatan diabetes berdasarkan tekanan darah. Untuk rekomendasi tekanan darah dan BMI, pengguna perlu mengisi data angka medis dahulu karena rekomendasi yang diberikan berdasarkan kondisi pasien saat ini. Adapun tekanan darah yang direkomendasikan bagi pasien diabetes adalah di bawah 130/80 mmHg [3]. Pembagian kategori tekanan darah yang diberikan berdasarkan klasifikasi AHA [29]. Untuk rekomendasi tekanan darah yang diberikan secara garis besar adalah menerapkan pola makan DASH (*dietary approaches to stop hypertension*) yang mengutamakan makanan tinggi serat, rendah lemak, dan rendah garam serta melakukan olahraga 150 menit/minggu. Rekomendasi yang diberikan tersebut diharapkan dapat membantu pasien untuk menjaga tekanan darah dari hipertensi. Adapun rekomendasi gaya hidup yang diberikan juga dapat bermanfaat untuk menjaga kolesterol pasien diabetes.

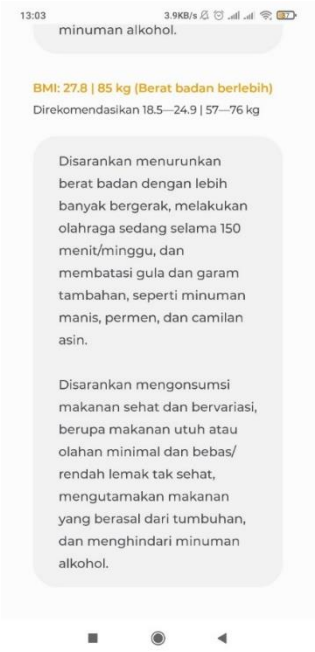
Selanjutnya, Gambar 10 menunjukkan tangkapan layar dari fitur rekomendasi perawatan diabetes berdasarkan BMI. Nilai BMI yang direkomendasikan adalah pada rentang 18,5-24,9 [30], [34]. Sementara itu, berat badan yang direkomendasikan untuk pasien dapat berbeda-beda, bergantung pada tinggi badan masing-masing individu. Pembagian kategori berat badan didasarkan pada klasifikasi WHO [30]. Adapun rekomendasi BMI yang diberikan secara garis besar adalah pasien direkomendasikan untuk lebih banyak bergerak dan menerapkan pola makan yang mengutamakan makanan olahan minimal, makanan bervariasi, sayur-sayuran, dan makanan rendah lemak. Rekomendasi yang diberikan bertujuan agar pasien dapat menjaga kesehatan tubuh dan berat badan.



Gambar 8. Tangkapan layar rekomendasi gula darah



Gambar 9. Tangkapan layar rekomendasi tekanan darah



Gambar 10. Tangkapan layar rekomendasi BMI

C. Penerapan MAD

MAD terdiri atas enam komponen [17] dan diterapkan dalam pengembangan aplikasi dalam penelitian ini. Komponen-komponen tersebut dijabarkan sebagai berikut.

Pertama, pengembangan aplikasi ditargetkan untuk versi Android terbaru, yakni Android 13 dengan level API 33 [17]. Versi Android terbaru menawarkan keamanan, privasi, dan performa yang ditingkatkan [37]. Penargetan versi terbaru ini juga memungkinkan penggunaan teknologi mutakhir dan memastikan aplikasi tersedia pada perangkat-perangkat terbaru [17], [37].

Kedua, IDE yang digunakan adalah Android Studio Electric Eel dan Flamingo. Android Studio menyediakan fasilitas pengembangan perangkat lunak yang dibutuhkan, seperti *source-code editor*, *build automation tools*, *debugger*, *lint tools*, *code optimizer and obfuscator*, dan emulator [38]. Fasilitas disediakan dalam satu tempat yang memungkinkan pengembangan dilakukan secara efisien.

Ketiga, bahasa pemrograman yang digunakan adalah Kotlin. Karakteristik dari sintaks Kotlin adalah ekspresif dan ringkas, artinya kode dapat dipahami secara intuitif dan lebih sedikit *boilerplate* [39]. Oleh karena itu, penggunaan Kotlin memungkinkan waktu pengembangan yang lebih singkat dan pemeliharaan kode yang lebih mudah.

Keempat, untuk membangun UI, digunakan Jetpack Compose. Pustaka ini didasarkan pada paradigma deklaratif sehingga setiap kali *event* terjadi, UI akan dibangun ulang berdasarkan *state* saat ini [40]. Dengan demikian, pemeliharaan UI dapat dilakukan secara lebih sederhana dan sedikit eror.

Kelima, aplikasi diperkuat oleh Android Jetpack sebagai sekumpulan pustaka dengan fungsi-fungsi yang spesifik untuk menunjang pembangunan aplikasi. Android Jetpack membungkus kode-kode yang rumit ke dalam pustaka sehingga pengembangan menjadi lebih berfokus pada fungsionalitas sistem untuk pengguna alih-alih infrastruktur sistem. Jetpack Compose juga memberikan dukungan kompatibilitas penggunaan teknologi baru pada Android versi lama [41].

Terakhir, arsitektur aplikasi yang digunakan adalah modern app architecture. Dalam perancangan ini, aplikasi dibagi menjadi 3 lapisan dengan tanggung jawab yang berbeda, yakni (1) lapisan UI yang menyajikan data ke layar dan menangani interaksi pengguna [42], (2) lapisan data yang menyediakan data [43], dan (3) lapisan domain yang berisi logika bisnis dan menghubungkan lapisan UI dengan data [44]. Pemisahan aplikasi ke dalam 3 lapisan yang saling independen ini dapat meningkatkan skalabilitas, memudahkan pemeliharaan, dan memudahkan pengujian.

Adapun pengujian dilakukan untuk memverifikasi kebenaran, perilaku fungsional, dan kegunaan aplikasi sebelum rilis ke publik [45]. Pengujian sistem dalam penelitian ini dilakukan secara otomatis oleh mesin. Kasus-kasus uji ditulis ke dalam kode, kemudian dijalankan oleh mesin sehingga pengujian dapat dilakukan dengan efisien.

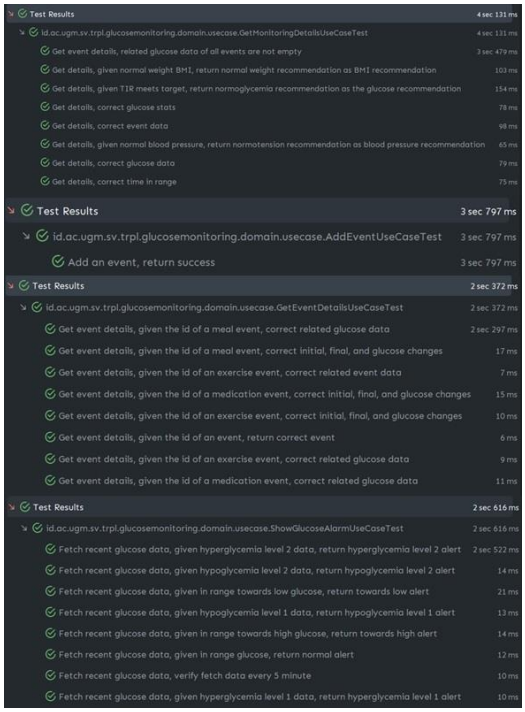
D. Hasil Pengujian

Pengujian aplikasi dilakukan melalui Android Studio dan menggunakan pustaka JUnit, Espresso, Truth, dan MockK.

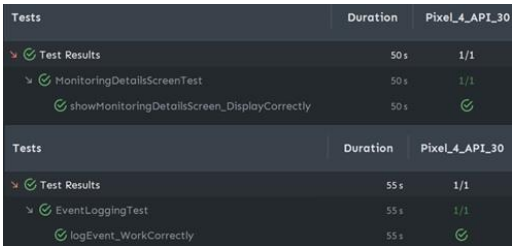
Pengujian unit diuji dengan 26 kasus uji yang ditunjukkan pada Gambar 11. Semua kasus uji mengeluarkan hasil sesuai yang diharapkan sehingga hal ini menunjukkan bahwa logika

bisnis yang diterapkan dalam aplikasi telah sesuai dengan spesifikasi kebutuhan pengguna dan tidak terdapat kerusakan.

Sementara itu, pengujian integrasi menguji interaksi pengguna di 2 alur penggunaan yang ditunjukkan pada Gambar 12. Hasil pengujian yang dilakukan mengembalikan hasil yang diharapkan sehingga hal ini menunjukkan bahwa komponen-komponen pada sistem berhasil diintegrasikan dengan baik.



Gambar 11. Hasil pengujian unit



Gambar 12. Hasil pengujian integrasi

V. SIMPULAN

Sistem monitor kadar gula darah berbasis *mobile* berhasil dirancang berdasarkan pendekatan multifaktorial. Aplikasi terdiri atas fitur monitor gula darah dan rekomendasi perawatan diabetes. Fitur monitor gula darah mencakup grafik gula darah, pencatatan aktivitas, dan alarm yang dapat membantu pasien dalam mengontrol gula darah sebagai faktor penting dalam manajemen diabetes. Sementara itu, fitur perawatan diabetes berisi rekomendasi-rekomendasi yang bersumber dari para pakar untuk mengedukasi dan mendorong pasien agar menerapkan gaya hidup yang sehat yang disesuaikan dengan kondisi terkini dari pasien diabetes.

Penerapan pendekatan multifaktorial pada aplikasi berbasis *mobile* ini diharapkan dapat membantu pasien diabetes dalam mengelola kondisi mereka secara mandiri dan efektif. Hal ini akan mengarah pada meningkatnya kesadaran diri tentang perawatan diabetes. Pada akhirnya, kemampuan manajemen diri yang lebih baik tersebut dapat membantu mereka mencapai tujuan perawatan diabetes, yakni menunda bahkan mencegah komplikasi yang membahayakan serta meningkatkan kualitas hidup.

Adapun dari sisi teknis, hasil rancang bangun dalam penelitian ini adalah aplikasi dengan aspek keberlanjutan pengembangan sistem. Enam komponen yang tercakup dalam MAD telah berhasil diterapkan di dalam implementasi aplikasi. Di samping performa yang baik, penerapan MAD ini menghasilkan aplikasi yang skalabel dan mudah dipelihara. Selain itu, aplikasi bersifat sumber terbuka dengan kode sumber yang telah tersedia di repositori. Oleh karena itu, diharapkan sistem dapat terus dikembangkan oleh siapa saja untuk memastikan relevansinya dan dampaknya yang berkelanjutan.

REFERENSI

[1] D. Magliano dan E. J. Boyko, *IDF diabetes atlas*, 10th edition. Brussels: International Diabetes Federation, 2021.

[2] N. A. ElSayed *dkk.*, “4. Comprehensive Medical Evaluation and Assessment of Comorbidities: *Standards of Care in Diabetes—2023*,” *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S49–S67, Jan 2023, doi: 10.2337/dc23-S004.

[3] N. A. ElSayed *dkk.*, “10. Cardiovascular Disease and Risk Management: *Standards of Care in Diabetes—2023*,” *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S158–S190, Jan 2023, doi: 10.2337/dc23-S010.

[4] N. A. ElSayed *dkk.*, “11. Chronic Kidney Disease and Risk Management: *Standards of Care in Diabetes—2023*,” *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S191–S202, Jan 2023, doi: 10.2337/dc23-S011.

[5] N. A. ElSayed *dkk.*, “12. Retinopathy, Neuropathy, and Foot Care: *Standards of Care in Diabetes—2023*,” *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S203–S215, Jan 2023, doi: 10.2337/dc23-S012.

[6] N. A. ElSayed *dkk.*, “6. Glycemic Targets: *Standards of Care in Diabetes—2023*,” *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S97–S110, Jan 2023, doi: 10.2337/dc23-S006.

[7] S. R. Patton, Associate Professor of Pediatrics, Department of Pediatrics, University of Kansas Medical Center, Kansas City, Kansas, M. A. Clements, dan Assistant Professor of Pediatrics, University of Missouri-Kansas City School of Medicine, Children’ s Mercy Hospital, Kansas City, Missouri, US, “Continuous Glucose Monitoring Versus Self-monitoring of Blood Glucose in Children with Type 1 DiabetesThe Pros and Cons,” *US Endocrinology*, vol. 08, no. 01, hlm. 27, 2012, doi: 10.17925/USE.2012.08.01.27.

[8] Y. Y. Cai, D. Cao, X. H. He, dan Q. X. Wang, “Continuous Glucose Monitoring System Based on Smart Phone,” *Procedia Engineering*, vol. 29, hlm. 3894–3898, 2012, doi: 10.1016/j.proeng.2012.01.590.

[9] K. C. Gunawardena *dkk.*, “The Influence of the Smart Glucose Manager Mobile Application on Diabetes Management,” *J Diabetes Sci Technol*, vol. 13, no. 1, hlm. 75–81, Jan 2019, doi: 10.1177/1932296818804522.

[10] B. E. Holtz *dkk.*, “The design and development of MyT1DHero: A mobile app for adolescents with type 1 diabetes and their parents,” *J Telemed Telecare*, vol. 25, no. 3, hlm. 172–180, Apr 2019, doi: 10.1177/1357633X17745470.

[11] M. D. Adu, U. H. Malabu, A. E. O. Malau-Aduli, dan B. S. Malau-Aduli, "The development of My Care Hub Mobile-Phone App to Support Self-Management in Australians with Type 1 or Type 2 Diabetes," *Sci Rep*, vol. 10, no. 1, hlm. 7, Jan 2020, doi: 10.1038/s41598-019-56411-0.

[12] I. Volkov dan G. Radchenko, "DiaMeter: a Mobile Application and Web Service for Monitoring Diabetes Mellitus," dalam *2020 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, Yekaterinburg, Russia: IEEE, Mei 2020, hlm. 0384–0387. doi: 10.1109/USBEREIT48449.2020.9117654.

[13] L. D. F. P. A. Alves, M. M. Maia, M. F. M. D. Araújo, M. M. C. Damasceno, dan R. W. J. F. D. Freitas, "Desenvolvimento e validação de uma tecnologia MHEALTH para a promoção do autocuidado de adolescentes com diabetes," *Ciênc. saúde coletiva*, vol. 26, no. 5, hlm. 1691–1700, Mei 2021, doi: 10.1590/1413-81232021265.04602021.

[14] R. Salari, S. R Niakan Kalhori, M. GhaziSaeedi, M. Jeddi, M. Nazari, dan F. Fatehi, "Mobile-Based and Cloud-Based System for Self-management of People With Type 2 Diabetes: Development and Usability Evaluation," *J Med Internet Res*, vol. 23, no. 6, hlm. e18167, Jun 2021, doi: 10.2196/18167.

[15] M. M. Din, N. R. Ali, T. Tajuddin, S. R. M. Dawam, dan S. M. Murad, "Mobile Application for Malaysian Diabetes Dietary Monitoring System (My-DDMS)," dalam *2021 6th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, Kedah, Malaysia: IEEE, Des 2021, hlm. 1–6. doi: 10.1109/ICRAIE52900.2021.9704006.

[16] E. Mehraeen *dkk.*, "Design and Development of a Mobile-Based Self-Care Application for Patients with Type 2 Diabetes," *J Diabetes Sci Technol*, vol. 16, no. 4, hlm. 1008–1015, Jul 2022, doi: 10.1177/19322968211007124.

[17] Google, "Modern Android development," Android Developers. Diakses: 24 Mei 2023. [Daring]. Tersedia pada: https://developer.android.com/modern-android-development

[18] G. Alfian *dkk.*, "Blood glucose prediction model for type 1 diabetes based on artificial neural network with time-domain features," *Biocybernetics and Biomedical Engineering*, vol. 40, no. 4, hlm. 1586–1599, Okt 2020, doi: 10.1016/j.bbe.2020.10.004.

[19] DirecNet, "Evaluation of Counter-regulatory Hormone Responses during Hypoglycemia and the Accuracy of Continuous Glucose Monitors in Children with T1DM," Diabetes Research in Children Network. Diakses: 28 Februari 2023. [Daring]. Tersedia pada: https://public.jaeb.org/direcnet/stdy/167

[20] American Diabetes Association, "*Standards of Medical Care in Diabetes—2022* Abridged for Primary Care Providers," *Clinical Diabetes*, vol. 40, no. 1, hlm. 10–38, Jan 2022, doi: 10.2337/cd22-as01.

[21] A. P. Campbell, "DASH Eating Plan: An Eating Pattern for Diabetes Management," *Diabetes Spectrum*, vol. 30, no. 2, hlm. 76–81, Mei 2017, doi: 10.2337/ds16-0084.

[22] A. B. Evert *dkk.*, "Nutrition Therapy for Adults With Diabetes or Prediabetes: A Consensus Report," *Diabetes Care*, vol. 42, no. 5, hlm. 731–754, Mei 2019, doi: 10.2337/dci19-0014.

[23] American Diabetes Association, "Good to Know: Factors Affecting Blood Glucose," *Clinical Diabetes*, vol. 36, no. 2, hlm. 202–202, Apr 2018, doi: 10.2337/cd18-0012.

[24] American Diabetes Association, "Blood Glucose & Exercise," ADA. Diakses: 28 Mei 2023. [Daring]. Tersedia pada: https://diabetes.org/healthy-living/fitness/getting-started-safely/blood-glucose-and-exercise

[25] American Diabetes Association, "What is the Diabetes Plate Method?," Diabetes Food Hub. Diakses: 28 Mei 2023. [Daring]. Tersedia pada: https://www.diabetesfoodhub.org/articles/what-is-the-diabetes-plate-method.html

[26] American Diabetes Association, "Choosing a CGM," ADA. Diakses: 25 Juni 2023. [Daring]. Tersedia pada: https://diabetes.org/tools-support/devices-technology/choosing-cgm

[27] American Diabetes Association, "Types of Insulin for People with Diabetes," *American Diabetes Association*.

[28] D. K. Arnett *dkk.*, "2019 ACC/AHA Guideline on the Primary Prevention of Cardiovascular Disease: Executive Summary: A Report of the American College of Cardiology/American Heart Association Task Force on Clinical Practice Guidelines," *Circulation*, vol. 140, no. 11, Sep 2019, doi: 10.1161/CIR.0000000000000677.

[29] P. K. Whelton *dkk.*, "2017 ACC/AHA/AAPA/ABC/ACPM/AGS/APhA/ASH/ASPC/NMA/PCNA Guideline for the Prevention, Detection, Evaluation, and Management of High Blood Pressure in Adults: A Report of the American College of Cardiology/American Heart Association Task Force on Clinical Practice Guidelines," *Hypertension*, vol. 71, no. 6, Jun 2018, doi: 10.1161/HYP.0000000000000065.

[30] World Health Organization, "A healthy lifestyle," WHO recommendations. Diakses: 29 Mei 2023. [Daring]. Tersedia pada: https://www.who.int/europe/news-room/fact-sheets/item/a-healthy-lifestyle---who-recommendations

[31] D. Howsmon dan B. W. Bequette, "Hypo- and Hyperglycemic Alarms: Devices and Algorithms," *J Diabetes Sci Technol*, vol. 9, no. 5, hlm. 1126–1137, Sep 2015, doi: 10.1177/1932296815583507.

[32] S. A. Little *dkk.*, "Recovery of Hypoglycemia Awareness in Long-standing Type 1 Diabetes: A Multicenter 2 × 2 Factorial Randomized Controlled Trial Comparing Insulin Pump With Multiple Daily Injections and Continuous With Conventional Glucose Self-monitoring (HypoCOMPaSS)," *Diabetes Care*, vol. 37, no. 8, hlm. 2114–2122, Agu 2014, doi: 10.2337/dc14-0030.

[33] Y. K. Lin *dkk.*, "Alarm Settings of Continuous Glucose Monitoring Systems and Associations to Glucose Outcomes in Type 1 Diabetes," *Journal of the Endocrine Society*, vol. 4, no. 1, hlm. bvz005, Jan 2020, doi: 10.1210/jendso/bvz005.

[34] N. A. ElSayed *dkk.*, "8. Obesity and Weight Management for the Prevention and Treatment of Type 2 Diabetes: *Standards of Care in Diabetes—2023*," *Diabetes Care*, vol. 46, no. Supplement_1, hlm. S128–S139, Jan 2023, doi: 10.2337/dc23-S008.

[35] Google, "Developer guides," Android Developers. Diakses: 8 Maret 2023. [Daring]. Tersedia pada: https://developer.android.com/docs

[36] A. Zuhri, "GlucoseMonitoring," GitHub. [Daring]. Tersedia pada: https://github.com/ariefzuhri/GlucoseMonitoring

[37] Google, "Target API level requirements for Google Play apps," Play Console Help. Diakses: 10 Oktober 2024. [Daring]. Tersedia pada: https://support.google.com/googleplay/android-developer/answer/11926878

[38] Google, "Meet Android Studio," Android Developers. Diakses: 30 Mei 2023. [Daring]. Tersedia pada: https://developer.android.com/studio/intro

[39] Kotlin, "Kotlin for Android," Kotlin Documentation. Diakses: 30 Mei 2023. [Daring]. Tersedia pada: https://kotlinlang.org/docs/android-overview.html

[40] A.-C. Bellini, "Jetpack Compose is now 1.0: announcing Android's modern toolkit for building native UI," Android Developers Blog. Diakses: 31 Mei 2023. [Daring]. Tersedia pada: https://android-developers.googleblog.com/2021/07/jetpack-compose-announcement.html

[41] Google, "Getting started with Android Jetpack," Android Developers. Diakses: 31 Mei 2023. [Daring]. Tersedia pada: https://developer.android.com/jetpack/getting-started

[42] Google, "UI layer," Android Developers. Diakses: 5 Juni 2023. [Daring]. Tersedia pada: https://developer.android.com/topic/architecture/ui-layer

[43] Google, "Data layer," Android Developers. Diakses: 5 Juni 2023. [Daring]. Tersedia pada: https://developer.android.com/topic/architecture/data-layer

[44] Google, "Domain layer," Android Developers. Diakses: 5 Juni 2023. [Daring]. Tersedia pada: https://developer.android.com/topic/architecture/domain-layer

[45] Google, "Fundamentals of testing Android apps," Android Developers. Diakses: 31 Mei 2023. [Daring]. Tersedia pada: https://developer.android.com/training/testing/fundamentals

Implementasi *Chatbot* pada Telegram sebagai *Monitoring Assistant* dengan Analisis Teks Klasifikasi Menggunakan Metode *Support Vector Machine*

Dwi Lestari[1], Lukman Subekti[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
lestdwi1003@mail.ugm.ac.id
[*]Korespondensi: lukmans@ugm.ac.id;

Abstract – In the digital era of Industry 4.0, there is a strong push for the integration of the world with internet-based processes as the main platform. A chatbot can be defined as a smart program designed to understand the meaning of human language. To provide answers that match user inquiries, a chatbot needs to simulate and classify the intent behind users' conversations. The objective of this research is to develop intent classification on a chatbot using Natural Language Processing (NLP) and the Support Vector Machine (SVM) algorithm. This research developed a chatbot on the Telegram platform with the Support Vector Machine (SVM) algorithm to measure accuracy levels, making the chatbot more precise. This research project successfully implemented the proposed design. The result of this research is a chatbot named "MONA," which can be used as a Monitoring Assistant for Netmonk customers registered in the system. Additionally, this research successfully conducted training on intent labels and texts, enabling the chatbot to provide interactive responses to user questions. Testing was performed using FastAPI integration to transmit data. The results of this testing showed an accuracy of 98.92% with support vector machine, 93.54% with naïve Bayes, and 95.8% with neural networks, making the support vector machine the preferred method in this application.

Keywords – Chatbot, Intent, Text Classification, Support Vector Machine, NLP

Intisari – Dalam dunia industri 4.0 era digital ini sangat mendorong integrasi dunia dengan proses internet sebagai dudukan utama. *Chatbot* sendiri dapat diartikan sebagai sebuah program ponsel pintar yang didesain untuk memahami maksud dari bahasa manusia, untuk mendapatkan jawaban yang sesuai dengan yang ditanyakan pengguna, *chatbot* perlu melakukan simulasi dan klasifikasi percakapan maksud atau keinginan (*intent*) dari pengguna. Tujuan dari penelitian ini adalah membuat klasifikasi *intent* pada *chatbot* dengan menggunakan teknologi *Natural Language Processing* dan algoritma *Support Vector Machine* (SVM). Penelitian ini membuat pengembangan *chatbot* pada platform Telegram dengan algoritma *Support Vector Machine* (SVM) untuk mengukur tingkat akurasi, sehingga *chatbot* menjadi lebih akurat. Penelitian ini sudah berhasil mengimplementasikan rancangan tersebut. Hasil dari penelitian yang dibuat ini adalah *chatbot* yang bernama "MONA" dapat digunakan sebagai *Monitoring Assistant* pelanggan *Netmonk* yang sudah terdaftar dalam sistem. Selain itu, penelitian ini juga berhasil melakukan *training* pada label dan teks *intent*, sehingga *chatbot* dapat memberikan jawaban yang interaktif dari pertanyaan pengguna, kemudian pengujian ini menggunakan integrasi *FastAPI* dalam mengirimkan data. Pada pengujian ini diperoleh hasil akurasi 98,92% dengan *support vector machine*, 93,54% dengan *naïve bayes*, dan 95,8% dengan *neural network*, sehingga penelitian ini menggunakan metode *support vector machine* dalam pengaplikasiannya.

Kata kunci – *Chatbot, Intent,* Teks Klasifikasi, *Support Vector Machine, NLP*

I. PENDAHULUAN

Monitoring Network merupakan suatu metode yang digunakan untuk melakukan pengamatan dan *monitoring* pada sistem jaringan sebuah komputer yang berjalan. *Monitoring network* dapat diterapkan pada sebuah *device* atau server yang terhubung ke *network* [1]. Dalam *monitoring* ini dapat memberikan sebuah informasi mengenai kondisi atau keadaan dari *device* atau *server* yang dimonitor. Informasi dari monitoring tersebut dapat dilakukan analisa kondisi server atau *device*. Tetapi yang menjadi permasalahan umum dalam sebuah perusahaan atau industri adalah fleksibilitas dalam proses *monitoring* apabila terjadi permasalahan human eror.

Dunia industri era digital sekarang semakin meningkatkan kecerdasan buatan dalam mengintegrasi ke dalam kehidupan sehari-hari. Salah satu yang dimanfaatkan di era digital ini adalah layanan tanya jawab secara *online* melalui *chat* pribadi, dengan adanya layanan *chat* secara *real-time* akan sangat memudahkan kebutuhan para pengguna dalam beraktivitas. *Chatbot* merupakan salah satu contoh dari sistem kecerdasan buatan dan menjadi salah satu contoh interaksi manusia dengan komputer yang sering dikenal [2].

Chatbot dapat diartikan juga sebagai program komputer yang diatur atau dirancang untuk dapat memahami satu atau lebih dari bahasa manusia dengan menggunakan pemrosesan bahasa alami untuk memudahkan pekerjaan manusia. *Chatbot* juga diartikan sebagai bot pintar, agen interaktif, asisten digital, atau entitas percakapan buatan [3]. Dilihat dari perkembangannya, *chatbot* akan sangat menjanjikan dalam dunia industri terutama dalam menyediakan kebutuhan pengguna dengan cepat dan dapat menanggapi secara khusus pertanyaan pengguna. Ada empat pendekatan kecerdasan buatan (*artifial intelligence*/ AI), yaitu berpikir layaknya manusia (*thinking humanly*), berpikir secara rasional (*thinking rationally*), bertindak layaknya manusia *(acting humanly*) dan bertindak secara rasional (*acting rationally*) [4]. Seiring berjalannya waktu kecerdasan buatan ini akan muncul mesin yang benar-benar mampu berpikir, bertindak, mendengar, melihat, dan berbicara layaknya manusia.

Pemrosesan bahasa alami atau yang disebut *Natural Language Processing* (NLP) merupakan kecerdasan buatan yang bertujuan dalam melakukan proses atau memahami bahasa alami manusia pada komputer [5]. Teknologi N*atural Language Processing* (NLP) ini menjadikan *chatbot* bisa menjadi seperti manusia saat melakukan dialog dengan pengguna. *Chatbot* menjadi salah satu dari model yang di implementasikan oleh teknologi *Natural Language Processing* (NLP) yang banyak dilakukan nya penelitian pada beberapa tahun terakhir. Banyak sekali industri yang mengintegrasikan *chatbot* ke dalam proses bisnis untuk mendapatkan suatu manfaat. Penerapan *chatbot* pada proses bisnis ini menjadikan fleksibilitas yang baik, karena dapat digunakan pada berbagai aplikasi. *Chatbot* juga dapat memberikan layanan yang interaktif dan menarik bagi pengguna. Tujuan adanya *chatbot* ini diterapkan yaitu untuk memberikan jawaban yang akurat terhadap pertanyaan-pertanyaan yang di berikan oleh pengguna. Selain itu juga, perusahaan mendapatkan manfaat lain dengan penggunaan *chatbot* ini bisa melihat data yang terkumpul melalui *history* percakapan *chatbot* untuk memahami pengguna dan mengetahui pendapat atau hal menarik yang ditanyakan oleh pengguna.

II. DASAR TEORI

A. *Chatbot*

Berbagai macam penelitian yang sudah dilakukan untuk *chatbot*, sehingga *chatbot* sendiri memiliki beragam definisi yang berbeda-beda. *Chatbot* adalah sebuah perangkat lunak dengan teknologi bahasa alami yang akan berinteraksi dengan pengguna dalam mencari informasi atau dialog yang berisikan perintah [2]. *Chatbot* di ciptakan bukan semata-mata tidak memiliki fungsi dan tujuan, dengan kata lain fungsi dan tujuan dari adanya *chatbot* untuk menyimulasikan percakapan cerdas dengan seorang manusia. Dialog yang dilakukan untuk mencari informasi akan menyediakan pengguna dengan informasi yang relevan dengan yang di *query* mereka.

Dialog yang berorientasi pada perintah ini dirancang untuk melakukan percakapan dengan hasil yang akan memberikan perintah yang dijalankan. Sehingga dapat disimpulkan bahwa sistem pencarian informasi hanya memberikan informasi mengenai suatu hal sedangkan sistem yang berorientasi perintah akan lebih interaktif dan memperbolehkan pengguna melakukan sebuah kegiatan perintah seperti pemesanan. *Chatbot* akan menganalisis teks yang diterimanya dan memahami apa yang dimaksud dan dibutuhkan oleh pengguna. Jawaban dari *chatbot* sudah tersedia berdasarkan maksud yang telah dipahami oleh sistem.

B. *Machine Learning*

Machine learning atau biasa dikenal dengan sebutan pembelajaran mesin ini merupakan salah satu ilmu komputer yang dapat bekerja tidak dengan di program secara eksplisit. *Machine learning* adalah kecerdasan buatan (AI) yang dapat mempelajari cara dalam membuat data [6]. *Machine learning* juga dapat diartikan sebagai ilmu yang mempelajari algoritma dan model statistik yang digunakan pada sistem komputer untuk bisa melakukan pekerjaan tertentu tanpa adanya instruksi yang eksplisit. Cara kerja dari *machine learning* ini sangat bergantung pada pola dan kesimpulan sehingga untuk mendapatkan pola dan kesimpulan tersebut dibutuhkannya algoritma *machine learning* yang menghasilkan model matematika yang didasarkan pada data sampel yang disebut dengan *training data*.

C. *Natural Language Processing*

Natural Language Processing (NLP) merupakan salah satu metode dari cabang ilmu *Artificial Intelligence* (AI) yang digunakan untuk memproses bahasa alami menjadi bahasa yang dapat dipahami oleh komputer [5]. Bahasa alami sendiri adalah bahasa yang digunakan oleh manusia dalam sehari-hari untuk bertukar informasi satu sama lain. Tujuan dari NLP ini yaitu melakukan proses atau memahami bahasa alami manusia yang dilakukan oleh komputer yang berhubungan pada kemampuan semantik dan sintaksis [7]. Pada dasarnya komputer tidak dapat mengerti bahasa alami yang digunakan oleh manusia, oleh karena itu diperlukannya algoritma yang dapat mengubah teks supaya dapat dimengerti. Pada dasarnya teknik yang digunakan dalam penerapan NLP ini adalah menggunakan pendekatan analisis sintaksis dan analisis semantik. Analisis sintaksis sendiri merupakan analisis bahasa alami dengan memperhatikan aturan dalam penulisan tata bahasa. Kemudian, untuk analisis semantik merupakan analisis bahasa alami dengan memperhatikan makna atau arti dari setiap kata, penggunaan tanda sampai struktur dari teks tersebut. Penelitian dari NLP sendiri memiliki tujuan utama yaitu membuat mesin komputer mampu memahami atau mengerti makna bahasa alami yang kemudian memberikan respons yang sesuai [8].

D. *Support Vector Machine*

Support Vector Machine (SVM) adalah suatu metode yang handal dalam menyelesaikan masalah klasifikasi data. Dalam penggunaannya model SVM ini dapat mengolah data menjadi data latih dan data uji. Data latih sendiri dapat digunakan dalam membentuk suatu model *Support Vector Machine* (SVM) sedangkan nilai dari parameter bebasnya dipilih dari data awal [9]. Setelah itu, model dari SVM yang dihasilkan ini akan digunakan untuk mengklasifikasi data uji. Algoritma *Support Vector Machine (SVM)* ini juga dapat diartikan sebagai algoritma yang memiliki tujuan untuk menemukan *hyperplane* maksimal, dimana *hyperplane* sendiri merupakan suatu fungsi yang dapat memisahkan antara dua kelas. Selama proses berlangsung, SVM akan memaksimalkan margin atau jarak antar pola pelatihan dan batas keputusan. Margin sendiri dapat diartikan sebagai jarak terpendek dari sebuah *hyperplane* terhadap satu sisi dari margin itu sama dengan jarak *hyperplane* dengan sisi margin lainnya, dengan catatan pada kedua margin tersebut berada pada posisi yang paralel dengan *hyperplane*.

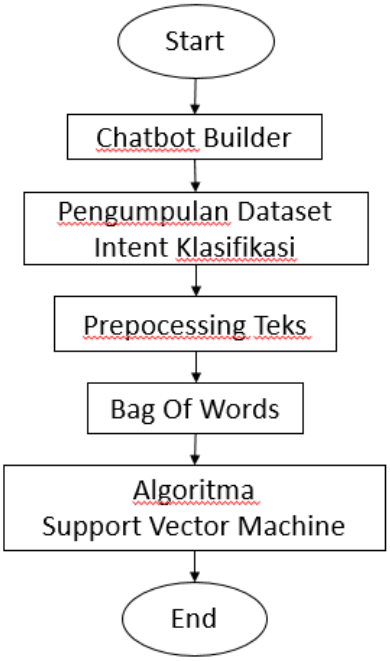
E. Telegram

Telegram merupakan salah satu jenis media untuk komunikasi yang di mana media komunikasi diartikan sebagai sarana dalam mendistribusikan, menyebarkan, dan

menyampaikan informasi [10]. Seiring dengan perkembangan teknologi dan komunikasi maka memunculkan perkembangan terhadap media komunikasi yang digunakan manusia dalam penyampaian pesan dan informasi. Salah satu bentuk perkembangan teknologi komunikasi yaitu media komunikasi *online*. Media *online* dapat diartikan sebagai sarana komunikasi secara *online* atau diakses melalui internet yang terdapat teks, foto, video, dan juga suara. Contoh dari sarana komunikasi *online* seperti email, *mailing list* (*website*, *blog*, whatsapp, line, dan telegram termaksud salah satu kategori media *online)*.

III. METODOLOGI

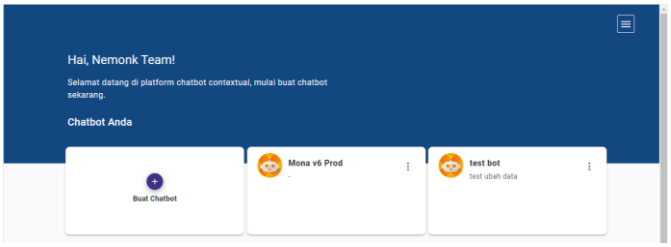
Tahapan penelitian ini dimulai dari membangun *chatbot* pada *chatbot builder* untuk membuat skema dari *chatbot*nya sendiri, kemudian dilakukannya pengumpulan *dataset* pada *intent* klasifikasi sebelum dilakukannya *preprocessing* teks. Setelah data tersebut diolah dan menghasilkan angka perhitungan dari data *preprocessing* tersebut masuklah ke dalam proses *Bag of Words* di mana kalimat atau dokumen dipresentasikan sebagai kantung dari kata-katanya hanya mempertimbangkan pada setiap kata saja dan menghiraukan tata bahasa dari urutan kata. Kemudian baru masuk ke dalam algoritma *support vector machine*. Pada Gambar 1 merupakan alur dari penelitian.



Gambar 1. *Flow* alur penelitian

A. *Chatbot Builder*

Pada proses membuat *chatbot* dibutuhkan sebuah *chatbot builder* untuk memudahkan dalam melakukan *development* pada *chatbot*. *Chatbot builder* ini bertujuan untuk mengetahui seberapa efektif dan interaktif *chatbot* dalam memberikan *respond* kepada pengguna. Hasil pengembangan *chatbot builder* ditampilkan pada Gambar 2.



Gambar 2. Tampilan *chatbot builder*

Pada Gambar 2 di atas merupakan tampilan dari *chatbot builder* yang merupakan hasil dari pengembangan secara mandiri yang digunakan sebagai platform untuk membuat, mengelola, dan juga mengintegrasikan *chatbot.* Pada *chatbot builder* ini mendukung integrasi dengan platform sosial media berupa telegram yang diterapkan sebagai sarana komunikasi *chatbot.*

B. Pengumpulan *Dataset Intent* Klasifikasi

Pada tahapan pengumpulan *dataset* ini bertujuan sebagai pengujian pada algoritma yang digunakan untuk mengetahui nilai akurasi terbaik untuk mengukur seberapa baik model dapat memprediksi variasi dalam data. Selain itu juga dapat melatih model di mana *dataset* yang sudah diberi label seperti *greeting*, *server*, dan *network* ini untuk melatih model pembelajaran mesin agar dapat mengenali *intent* dengan akurasi yang tinggi. *Dataset intent* klasifikasi disajikan pada Tabel 1.

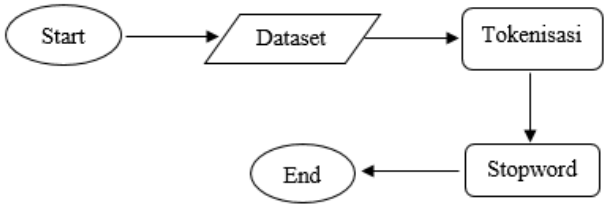
Tabel 1. Contoh *dataset intent* klasifikasi

Label	Text
Greeting	Selamat pagi
Greeting	Halo *chatbot*
Greeting	Apakah Anda hadir, Saya butuh bantuan dalam beberapa hal.
Greeting	Selamat sore, Bagaimana keadaan Anda hari ini?
Server	Bagaimana status server saat ini ?
Server	Bagaimana perkembangan terkini dari status server ?
Server	Apakah server sedang beroperasi dengan baik
Server	Adakah masalah pada server ?
Network	Status jaringan saat ini
Network	Kondisi perangkat jaringan
Network	Bagaimana ketersediaan perangkat jaringan saat ini

Pada Tabel 1 di atas berisikan mengenai teks dari *intent* label *greeting*, *server*, dan *network* atau jaringan. Pada tabel ini juga berisikan mengenai contoh-contoh dari pertanyaan atau pernyataan yang berkaitan dengan *greeting*, *server* dan juga perangkat jaringan. Dengan adanya *dataset* ini, memudahkan *chatbot* dalam memberikan informasi yang dibutuhkan oleh pengguna.

C. *Preprocessing* Teks

Tahap *preprocessing* ini merupakan tahapan di mana data *input* akan diolah kembali yang kemudian diproses lebih lanjut. Pada tahapan *preprocessing* ini kalimat sebagai masukan yang sudah dikumpulkan kemudian akan diekstrak teksnya untuk melalui beberapa tahapan proses seperti tokenisasi dan *stopwords*. Gambar 3 di bawah ini merupakan alur dari *preprocessing*



Gambar 3. *Flow preprocessing*

Pada proses tokenisasi ini merupakan salah satu prosedur dalam pemecahan teks menjadi kata, frasa, atau bagian lainnya yang memiliki makna, yaitu token. Dapat diartikan bahwa tokenisasi merupakan bentuk segmentasi dari teks. Gambar 4 menunjukkan teks sebelum ditokenisasi, sedangkan Gambar 5 menunjukkan teks setelah ditokenisasi.

```
Corpus
```

	text
0	bagaimana status server saat ini?
1	bagaimana perkembangan terkini dari status ser...
2	apakah server sedang beroperasi dengan baik?
3	bisakah anda memberikan pembaruan terbaru sepu...
4	mohon informasi mengenai kondisi server saat ini.

Gambar 4. Teks sebelum ditokenisasi

	text
0	[bagaimana, status, server, saat, ini, ?]
1	[bagaimana, perkembangan, terkini, dari, statu...
2	[apakah, server, sedang, beroperasi, dengan, b...
3	[bisakah, anda, memberikan, pembaruan, terbaru...
4	[mohon, informasi, mengenai, kondisi, server, ...

Gambar 5. Teks setelah ditokenisasi

Setelah ditokenisasi, data akan melewati proses *Stopwords*. *Stopwords* merupakan kata yang biasa muncul dalam jumlah besar dan juga tidak mempunyai makna yang berarti pada sebuah kalimat. *Stopwords* juga dapat diartikan sebagai kata-kata umum (*commond words*) yang biasa muncul dan juga tidak memberikan sebuah informasi penting (diacuhkan). Kata-kata *Stopwords* disajikan pada Tabel 2.

Tabel 2. Contoh kata *stopwords*

Kata-kata *stopwords*			
adalah	adapun	akan	atau
bahwa	banyak	bisa	bagaimana
cuma	cara	cukup	dan
datang	dari	di	hendak

Pada Tabel 2 merupakan contoh beberapa kata *stopwords* yang kemudian akan dilakukan proses pembuangan *stopword*s dengan tujuan untuk mengetahui suatu kata tersebut merupakan bagian dari *stopwords* atau bukan. Proses pembuangan *stopword*s disajikan pada Tabel 3

Tabel 3. Proses *stopwords removal*

Jenis		**Teks**
Sebelum *Removal*	*Stopword*	"bagaimana", "status", "server", "saat", "ini"
Setelah *Removal*	*Stopword*	"bagaimana", "status", "server"

Dapat dilihat pada Tabel 3 terdapat contoh kalimat sebelum di lakukan *stopword removal* dan juga setelah dilakukan *stopword removal.* Beberapa kata yang terdaftar dalam *stopword* tersebut akan hilang secara otomatis setelah dilakukannya *stopword removal*.

D. *Bag of Word*

Pada tahapan *Bag of Word* (BoW) ini teks, kalimat atau dokumen dipresentasikan sebagai kantung dari kata-katanya dimana hanya mempertimbangkan pada setiap kata saja dan menghiraukan tata bahasa dari urutan kata. Contoh token kata sebelum BoW disajikan pada Tabel 4, sedangkan setelah BoW pada Tabel 5.

Tabel 4. Contoh token kata sebelum BoW

	Kalimat
Token Kata	Bagaimana status server baik ini saat

Tabel 5. Contoh token kata setelah BoW

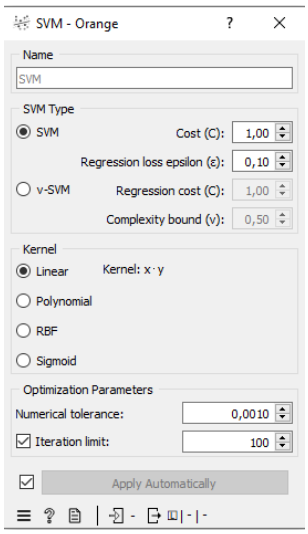
Token Kata	**bagaimana**	**status**	**server**	**baik**	**ini**	**saat**
BoW	1	1	1	0	1	1

Pada Tabel 4 di atas ditunjukkan representasi dari kalimat "bagaimana status server saat ini". Setelah semua kata dalam kalimat di *dataset* sudah diubah ke dalam bentuk *Bag of Words* seperti pada Tabel 5, yaitu berpola angka 0 dan 1 atau bisa juga dengan pola angka 1 dan 2 pada kalimat tersebut akan dijadikan sebagai masukan atau *input* dalam menganalisis dengan algoritma model *support vector machine*.

E. Analisis *Support Vector Machine*

Pada *software* Orange ini dapat menggunakan model *Support Vector Machine* (SVM) dalam melakukan pengujian model data. Pada Gambar 6 di bawah merupakan bentuk dari karakteristik pada model *Support Vector Machine* (SVM)

yang akan digunakan dalam mengetahui nilai parameter terbaik. Pada penelitian ini, dapat memilih tipe dari *Support Vector Machine* nya sendiri yaitu ada SVM dan v-SVM (nu). Pada tipe SVM terdapat *cost* dan *regression loss epsilon* di mana fungsi dari *cost* sendiri sebagai evaluasi seberapa baik model *Support Vector Machine* dalam menangani data *training* dan juga seberapa baik model ini dapat memisahkan kelas yang berbeda.



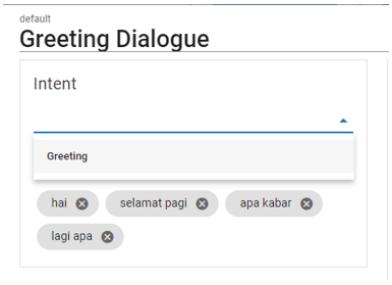
Gambar 6. Karakteristik SVM

IV. HASIL DAN PEMBAHASAN

Pada penelitian ini dilakukan dua macam metode pengujian, yang pertama pengujian pada *chatbot builder* dan yang kedua pengujian akurasi pada algoritma yang akan digunakan. Pengujian pada *chatbot builder* ini bertujuan untuk mengetahui seberapa efektif dan interaktif *chatbot* dalam memberikan jawaban kepada pengguna. Sedangkan pada pengujian kedua yaitu pengujian pada algoritma yang digunakan ini bertujuan untuk mengetahui nilai akurasi terbaik untuk mengukur seberapa baik model dapat memprediksi variasi dalam data.

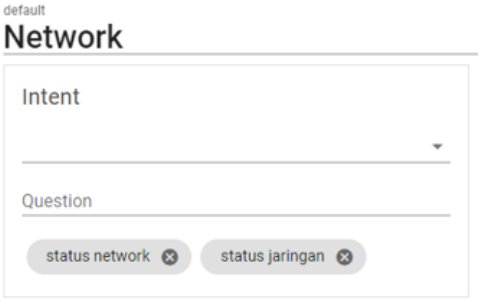
A. Pengujian *Intent* pada *Chatbot Builder*

Pengujian *intent* pada *chatbot builder* terdapat *greeting dialogue*, *intent greeting*, dan *intent network Greeting dialogue* disajikan pada Gambar 7, *Intent server* disajikan pada Gambar 8, sedangkan *Intent network* disajikan pada Gambar 9.



Gambar 7. *Intent greeting*

Gambar 8. *Intent server*

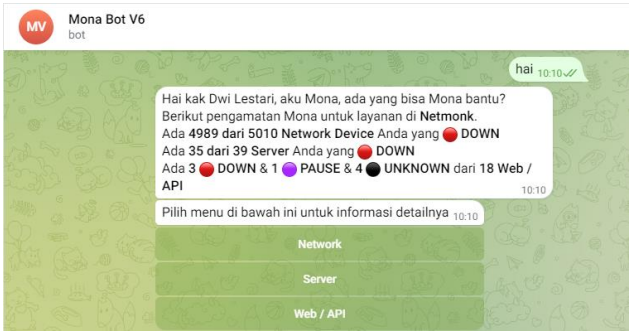


Gambar 9. *Intent Network*

Setelah dilakukan masukan satu per satu *question* yang akan digunakan, kemudian *intent* tersebut akan secara otomatis tersimpan dalam sistem *builder*. Seperti pada Gambar 7, Gambar 8, dan Gambar 9 setelah dilakukan penyimpanan, label *intent* dan juga *question* sudah secara langsung tersimpan dalam sistem. Setelah itu akan lebih mudah dalam mengimplementasikan *intent* pada *greeting* dialog pada saat melakukan *builder* di *chatbot*. Selain *question* itu, masih bisa menambahkan *question* lainnya untuk mewakili dari kalimat *intent* lainnya. Hasil sebelum dilakukan *training intent* disajikan pada Gambar 10.

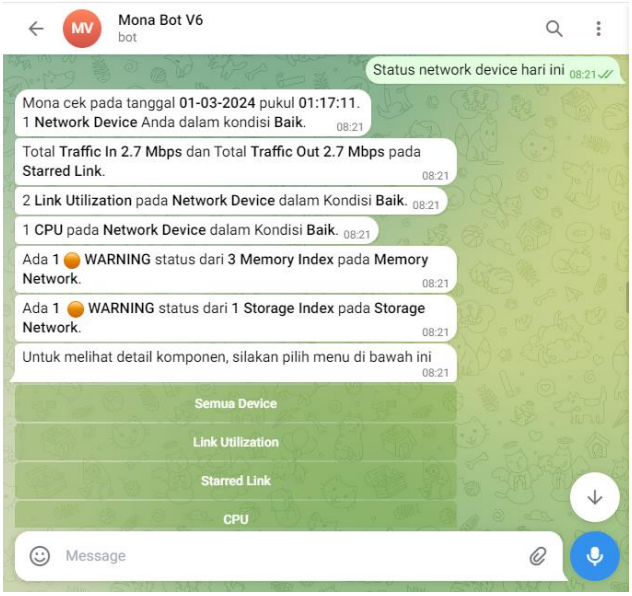
Gambar 10. Hasil sebelum dilakukan *training intent*

Pada gambar 10 terlihat hasil *chatbot* sebelum dilakukan *training*, *chatbot* tidak dapat memahami maksud dari pengguna. Hasil setelah dilakukan *training intent greeting* disajikan pada Gambar 11. Hasil setelah dilakukan *training intent server* disajikan pada Gambar 12. Hasil setelah dilakukan *training intent network* disajikan pada Gambar 13.



Gambar 11. Hasil setelah dilakukan *training intent greeting*

Gambar 12. Hasil setelah dilakukan *training intent server*

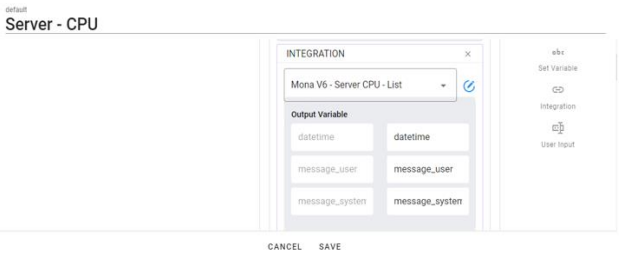


Gambar 13. Hasil setelah dilakukan *training intent network*

Setelah dilakukan *training intent greeting*, *server*, dan *network* seperti pada Gambar 11, Gambar 12, Gambar 13 dapat dilihat bahwa *chatbot* sudah bisa memahami kata-kata dari pengguna sehingga *chatbot* dapat memberikan jawaban yang sesuai.

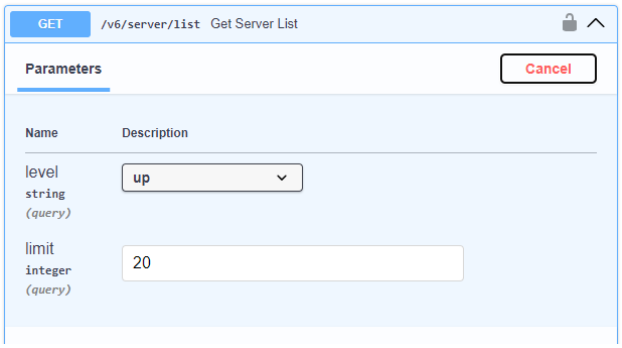
B. Pengujian Integrasi FastAPI

Gambar 14 merupakan integrasi *FastAPI* pada *chatbot.*



Gambar 14. Integrasi *FastAPI* pada *chatbot*

Pada saat melakukan *chatbot builder*, *chatbot* tidak akan bisa memberikan jawaban yang informatif apabila datanya tidak ada. Oleh karena itu, dilakukannya *integration* pada *chatbot*. Pada Gambar 14 merupakan fitur dalam *chatbot builder* dalam melakukan integrasi data dengan menggunakan Integrasi *FastAPI*. Pada *chatbot builder* hanya perlu menentukan id yang digunakan pada integrasi *FastAPI*, sehingga data akan terhubung pada *chatbot builder* sebagai *output*. Parameter *FastAPI* disajikan pada Gambar 15.



Gambar 15. Parameter *FastAPI*

Pada Gambar 15 di atas merupakan bentuk dari *FastAPI* dari Mona V6 – Server - List, terdapat parameter yang menjadi salah satu peran penting dari definisi *endpoint* API, selain itu juga dapat membuat API yang lebih fleksibel dan dapat menerima berbagai jenis masukan dari pengguna dengan cara yang aman dan terstruktur pastinya. Pada parameter ini terdapat dua parameter yaitu level dan limit. Parameter level dapat digunakan dalam menentukan tingkat atau level detail dari informasi yang diperoleh. Nilai yang terdapat pada parameter level ini yaitu "*up*" dan "*down*". Level "*up*" berarti mengembalikan informasi tentang server yang sedang aktif atau *up*. Kemudian level "*down*" berarti mengembalikan informasi tentang server yang sedang tidak aktif atau *down*.

C. Pengujian Model *Dataset*

1. Pengujian Nilai Akurasi pada *naïve bayes*

Pada pengujian *dataset* dengan model algoritma *naïve bayes* ini di dapatkan hasil perhitungan akurasi sebesar 93,5 % ini menunjukkan bahwa tingkat keberhasilan model dalam melakukan prediksi kelas atau label pada *dataset* sebesar 93,5%. Pada hasil nilai tersebut, model

naïve bayes berhasil melakukan prediksi dengan tepat sekitar 93,5% dari semua sampel yang ada dalam *dataset*. Hasil nilai akurasi dengan *naive bayes* disajikan pada Gambar 16.

```
Naive Bayes Accuracy Score ->  93.54838709677419
```

Gambar 16. Hasil nilai akurasi dengan *naive bayes*

Pada Gambar 16 terlihat proses dalam melatih model klasifikasi *naïve bayes* dengan menggunakan data latih dan kemudian dilakukan pengujian performa dengan menggunakan data uji.

2. Pengujian nilai akurasi pada *support vector machine*

Hasil nilai akurasi *support vector machine* disajikan pada Gambar 17.

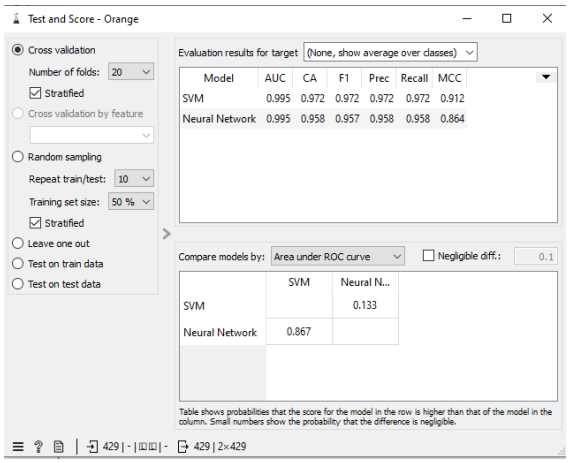
```
SVM Accuracy Score ->  98.9247311827957
```

Gambar 17. Hasil nilai akurasi SVM

Pada pengujian nilai akurasi seperti pada Gambar 17 dengan menggunakan klasifikasi algoritma *support vector machine* ini di dapatkan hasil 98,92%. Hasil tersebut menunjukkan tingkat keberhasilan model dalam memprediksi kelas atau label pada sebuah *dataset*. Dengan hasil nilai akurasi sebesar ini, model *support vector machine* berhasil memprediksi dengan tepat sekitar 98,92% dari semua sampel yang ada dalam *dataset*. Akurasi yang tinggi ini menunjukkan kinerja yang baik dari model dalam memahami pola-pola yang ada dalam *dataset* dan juga dalam membuat prediksi yang akurat.

3. Pengujian Model dengan Menggunakan Orange

Pengujian model dengan menggunakan Orange dapat diketahui hasil *test* dan *score* dengan *cross validation*. Hasil *test* dan *score* dengan *cross validation* disajikan pada Gambar 18.



Gambar 18. Hasil *test* dan *score* dengan c*ross validation*

Pada pengujian kali ini menggunakan dua buah model algoritma yaitu *support vector machine* dan *neural network*. Kemudian, pada pengujian ini juga menggunakan *dataset* yang berbeda untuk membuktikan model dan *dataset* mana yang paling baik untuk digunakan. Pada Gambar 18 merupakan hasil dari pengujian dengan menggunakan *cross validation* di Orange. Melalui pengujian ini di dapatkan hasil akurasi pada *neural network* sebesar 0.958.

V. SIMPULAN

Setelah dilakukannya perancangan, pengimplementasian, dan pengujian, penelitian ini dapat diambil kesimpulan, bahwa berhasil melakukan *build* pada *chatbot* dengan menggunakan *chatbot builder* dan berhasil melakukan *training* pada label dan teks *intent*, sehingga *chatbot* dapat memberikan jawaban yang akurat sesuai dengan pertanyaan dari pengguna. Kemudian, berhasil melakukan pengujian terhadap integrasi dengan menggunakan *FastAPI*, sehingga *chatbot* mampu memberikan (*message*) berupa *summary information* kepada pengguna.

Pengujian model dalam mengukur akurasi pada *chatbot* ini di dapatkan hasil yang terbaik adalah dengan menggunakan metode *support vector machine* memperoleh hasil nilai akurasi 98,92%, sedangkan dengan metode *naïve bayes* didapatkan hasil akurasi sebesar 93,54%, dan terakhir dengan menggunakan metode *Neural Network* didapatkan hasil akurasi sebesar 95,8 %.

DAFTAR PUSTAKA

[1] S. Lee, K. Levanti, and H. S. Kim, “Network monitoring: Present and future,” *Comput. Netw.*, vol. 65, pp. 84–98, Jun. 2014, doi: 10.1016/j.comnet.2014.03.007.

[2] T. P. Nagarhalli, V. Vaze, and N. K. Rana, “A Review of Current Trends in the Development of Chatbot Systems,” in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India: IEEE, Mar. 2020, pp. 706–710. doi: 10.1109/ICACCS48705.2020.9074420.

[3] E. Adamopoulou and L. Moussiades, “An overview of chatbot technology,” IFIP Advances in Information and Communication Technology, pp. 373–383, Jun. 2020. doi:10.1007/978-3-030-49186-4_31

[4] S. J. Russell and P. Norvig, Artificial Intelligence. Upper Saddle River, N.J, New Jersey: Pearson Education, 2009.

[5] M. Zhou, N. Duan, S. Liu, and H.-Y. Shum, “Progress in Neural NLP: Modeling, Learning, and Reasoning,” *Engineering*, vol. 6, no. 3, pp. 275–290, Mar. 2020, doi: 10.1016/j.eng.2019.12.014.

[6] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/science.aaa8415.

[7] C. O. Bilah, “Deteksi Intent pada Teks Bahasa Indonesia Menggunakan Bidirectional Long Short-Term Memory,” thesis, 2023

[8] Y. Kang, Z. Cai, C.-W. Tan, Q. Huang, and H. Liu, “Natural language processing (NLP) in management research: A literature review,” *J. Manag. Anal.*, vol. 7, no. 2, pp. 139–172, Apr. 2020, doi: 10.1080/23270012.2020.1756939.

[9] S. Yue, P. Li, and P. Hao, “SVM classification:Its contents and challenges,” *Appl. Math.- J. Chin. Univ.*, vol. 18, no. 3, pp. 332–342, Sep. 2003, doi: 10.1007/s11766-003-0059-5.

[10] F. Fitriansyah and Aryadillah, “Penggunaan Telegram Sebagai Media Komunikasi Dalam Pembelajaran Online,” Cakrawala, vol. 20, Sep. 2020. doi:10.31294/jc.v20i2.8935

Implementasi *Web Application Firewall* (WAF) pada Aplikasi Fishku Berbasis *Google Cloud Armor*

Nabila Apriliana Widiyono[1], Unan Yusmaniar Oktiawati[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
nabilaapriliana@mail.ugm.ac.id
[*]Korespondensi: unan_yusmaniar@ugm.ac.id;

Abstract – The security of Website Applications has become a pressing issue in an increasingly complex digital era. This research aims to implement Google Cloud Armor security services on the Google Cloud Platform to protect the "Fishku" Website Application, specifically against three types of attacks: Local File Inclusion (LFI), Vulnerability Scanner, and Protocol Attack. Website Application security plays a crucial role, given the rising threats of cyberattacks that can jeopardize the integrity and confidentiality of user data. This study employs the "Fishku" Website Application as the test subject. Testing is conducted before and after the implementation of Google Cloud Armor, using a laptop as the tool and the Kali Linux operating system installed on VirtualBox to evaluate the effectiveness of the protection provided. Furthermore, this research involves configuring a Load Balancer and utilizing Alerts features to detect potential attacks. Comprehensive system performance monitoring is conducted through data metric analysis. The research results indicate that Google Cloud Armor successfully shields the "Fishku" Website Application from these attacks, highlighting its effectiveness in enhancing security and system performance. The implications of this research are significant in the context of developing secure Website Applications, with Google Cloud Armor as a viable solution. In conclusion, this study offers valuable insights into the necessity of safeguarding Website Applications and demonstrates how Google Cloud Armor can be a solution to combat cyber threats. The broader implications of these research findings can serve as a foundation for future developments in Website Application security.

Keywords – Fishku Application, Security System, Google Cloud Armor, Google Cloud Platform

Intisari – Keamanan Aplikasi *Website* menjadi isu mendesak dalam era digital yang semakin kompleks. Penelitian ini bertujuan untuk menerapkan layanan keamanan *Google Cloud Armor* pada *Google Cloud Platform* dalam melindungi Aplikasi *Website* Fishku, khususnya terhadap 3 (tiga) jenis serangan *Local File Inclusion (LFI)*, *Vulnerability Scanner*, dan *Protocol Attack*. Keamanan Aplikasi *Website* memiliki peran yang sangat penting mengingat meningkatnya serangan siber yang dapat mengancam integritas dan kerahasiaan data pengguna. Penelitian ini menggunakan Aplikasi *Website* "Fishku" sebagai subjek uji coba. Pengujian dilakukan sebelum dan setelah penerapan *Google Cloud Armor*, dengan menggunakan laptop sebagai alat dan sistem operasi *Kali Linux* yang telah dilakukan instalasi pada *VirtualBox* untuk menguji keberhasilan perlindungan yang diberikan. Selain itu, penelitian ini juga melibatkan konfigurasi *Load Balancer* dan pemanfaatan fitur *Alerts* untuk mendeteksi serangan potensial. Analisis data *metric* juga dilakukan untuk memantau kinerja sistem secara lebih komprehensif. Hasil penelitian menunjukkan bahwa *Google Cloud Armor* berhasil melindungi Aplikasi *Website* "Fishku" dari serangan tersebut dan mengungkapkan keberhasilan perlindungan dalam keamanan dan kinerja sistem. Dampak dari penelitian ini penting dalam konteks pengembangan Aplikasi *Website* yang aman, dengan *Google Cloud Armor* sebagai solusi yang layak dipertimbangkan. Dalam kesimpulannya, penelitian ini memberikan pandangan penting mengenai perlunya perlindungan pada Aplikasi *Website* dan memberikan pandangan bagaimana *Google Cloud Armor* dapat menjadi solusi dalam mengatasi ancaman siber. Implikasi lebih luas dari hasil penelitian ini dapat membentuk landasan untuk perkembangan keamanan Aplikasi *Website* di masa depan.

Kata kunci – Aplikasi Fishku, Sistem Keamanan, *Google Cloud Armor, Google Cloud Platform*

I. PENDAHULUAN

Di era disrupsi dan era revolusi industri 4.0 perkembangan teknologi aplikasi dalam dunia bisnis menuntut manusia selalu untuk berkembang dan berinovasi. Teknologi internet dan dunia digital mengalami disrupsi yang begitu cepat dan canggih yang semua itu membantu dalam kehidupan manusia [1]. Sistem yang dikembangkan para ilmuwan pasti mengalami keterbatasan atau kelemahan. Hal inilah yang menjadikan manusia untuk selalu berpikir dalam mengatasi masalah yang dihadapi. Setiap teknologi yang diciptakan pasti memiliki kelemahan. Kelemahan atau kekurangan dalam produk teknologi mendorong manusia untuk dapat memperbaiki dengan inovasi sehingga meminimalisir kelemahan tersebut. Kelemahan atau kekurangan dalam produk teknologi mendorong manusia untuk dapat memperbaiki dengan inovasi, sehingga meminimalisir kelemahan tersebut.

Keamanan atau *security* sebuah aplikasi harus diperhatikan oleh setiap pemilik, instansi atau perusahaan agar tehindar dari kejahatan. Masalah keamanan atau gangguan sudah terjadi dan bertebaran di internet seperti serangan *Malware*, eksploitasi, injeksi *database* dan sebagainya. Menurut [2] pada Tahun 2016, terdapat sekitar 90% kejahatan internet yang menyerang aplikasi *web* dengan serangan yang paling terkenal yaitu dengan cara menginjeksi *database* dengan capaian 47,06%. Kerusakan maupun kebocoran data ini akan mengancam setiap saat dengan meningkatnya kemahiran atau kepandaian sumber daya manusia. Dalam mengatasi masalah pengamanan *web* atau aplikasi dari serangan *hacker* dapat dilakukan dengan cara *self-test*. *Self-test* yaitu suatu pengujian yang dilakukan terhadap *web* server secara legal dengan aktivitas menyerupai *hacker*. [2] menyatakan *self test* dapat dilakukan dengan beberapa metode *penetrastion testing* seperti *Information System Security Assesment Framework (ISSAF)*, *Open Web Aplication Security Project (OWASP)* versi4 dan *Open Source Security Testing Methodologi Manual (OSSTMM)*. Dalam Penelitian ini akan dilakukan uji keamanan terhadap ancaman aplikasi *web* Fishku dengan metode *Web Application Firewall (WAF)* [3].

Dalam konteks inovasi di dunia aplikasi, aplikasi "Fishku" hadir sebagai sebuah *startup e-commerce* di bidang perikanan. Aplikasi ini bertujuan untuk mempermudah kegiatan jual beli ikan serta mendeteksi kesegaran ikan menggunakan teknologi *machine learning*. Fishku merupakan hasil kerja tim dari peneliti yang juga merupakan bagian dari tim pengembang aplikasi ini. Aplikasi Fishku diciptakan melalui Program Bangkit Academy 2022 (Google, GoTo, Traveloka) yang merupakan bagian dari Studi Independen Kampus Merdeka Kemendikbudristek. Fishku tersebut berhasil mendapatkan pendanaan dari Dikti dan Google serta penghargaan dalam *Top 15 Best Capstone Project-Bangkit Academy*.[4].

Latar belakang ini mendorong peneliti untuk mengembangkan sistem keamanan yang berjudul "Implementasi *Web Application Firewall (WAF)* pada aplikasi Fishku berbasis *Google Cloud Armor*". Dengan kata lain, tujuan utama untuk memastikan bahwa layanan keamanan yakni *Google Cloud Armor* yang digunakan mampu memberikan perlindungan yang lebih baik terhadap serangan dan potensi ancaman yang mungkin terjadi pada aplikasi *web* tersebut.

II. DASAR TEORI

A. Keamanan Aplikasi *Web*

Keamanan dalam aplikasi *web* merupakan suatu isu yang memerlukan perhatian serius. Terdapat beberapa solusi yang dapat dilakukan untuk menerapkan layanan keamanan pada aplikasi *web*. Meskipun hal ini tidak sepenuhnya sempurna, tetapi tindakan preventif telah diambil untuk menghindari hal- hal yang tidak diinginkan. Salah satu solusi yang dapat diterapkan adalah dengan mengimplementasikan *Web Application Firewall* (WAF). WAF berfungsi untuk memeriksa lalu lintas data paket dari aplikasi *web* dan juga dapat berfungsi untuk memblokir beberapa serangan pada aplikasi *web* [5]. Berdasarkan daftar OWASP Top Ten 2021, tiga serangan teratas adalah *Broken Access Control*, *Cryptographic Failures*, dan *Injection* [6]. Ketiga serangan ini, seperti yang sudah umum diketahui, merupakan serangan terhadap target pada server *web*.

B. *Web Application Firewall (WAF)*

Web Application Firewall (WAF) adalah perangkat lunak yang bertugas memantau aliran lalu lintas antara aplikasi *web* dan internet. *Firewall* diletakkan di sisi server yang menganalisis setiap paket data yang masuk. Hanya paket- paket data yang dianggap aman yang diteruskan ke server, sedangkan semua paket data yang diidentifikasi sebagai berpotensi membahayakan akan ditolak oleh *firewall*. Meskipun diklaim sebagai metode perlindungan keamanan, WAF tidak selalu mampu memberikan perlindungan total terhadap segala bentuk serangan yang mungkin terjadi pada aplikasi [7].

C. *Cloud Armor*

Cloud Armor adalah layanan keamanan yang disediakan oleh *Google Cloud Platform* untuk melindungi aplikasi dan layanan *cloud* dari serangan DDoS dan serangan *web* lainnya. Layanan ini menggunakan infrastruktur global Google untuk memberikan pertahanan secara besar-besaran terhadap serangan DDoS. Cloud Armor terintegrasi dengan *Load Balancer* HTTP(S) global dan memblokir lalu lintas berdasarkan alamat IP atau rentangnya. Layanan ini dapat digunakan untuk mencegah pengguna atau lalu lintas yang berbahaya agar tidak mengakses sumber daya, atau yang lebih buruk lagi, mengambil alih kendali dari VPC berdasarkan aturan yang telah ditetapkan. Mode pratinjau memungkinkan pengguna menganalisis pola serangan tanpa mengganggu pengguna reguler [8].

D. *Local File Inclusion*

Local File Inclusion (LFI) adalah sebuah kerentanan yang memungkinkan seorang penyerang untuk menyisipkan atau memasukkan berkas menggunakan karakter khusus (seperti "../") ke dalam server melalui peramban *web*. Kerentanan ini terjadi ketika sebuah aplikasi *web* memasukkan berkas ke dalam proses tanpa melakukan proses pembersihan atau penyaringan input dengan benar. Akibatnya, hal ini memberikan peluang bagi penyerang untuk memanipulasi input yang dimasukkan dan menyisipkan muatan yang

memungkinkan akses ke direktori lain *(Path Ttraversal)* serta menyertakan berkas lain dari server *web* itu sendiri [9].

E. *Vulnerability Scanner*

Vulnerability scanner adalah alat atau perangkat lunak yang digunakan untuk menemukan dan menganalisis celah keamanan dalam sistem atau aplikasi. Tujuannya adalah untuk mengidentifikasi potensi kerentanan yang bisa dimanfaatkan oleh penyerang, sehingga langkah-langkah pencegahan dapat diambil sebelum kerusakan terjadi. *Scanner* ini biasanya digunakan untuk mencari kerentanan keamanan seperti *Cross-site scripting, SQL Injection, Command Injection, Path Traversal,* dan konfigurasi server yang tidak aman [10].

F. *Protocol Attack*

Protocol Attack adalah serangan yang memanfaatkan celah dalam protokol komunikasi. Salah satu jenis serangannya adalah HTTP *splitting*. Dalam serangan ini, penyerang memisahkan permintaan HTTP menjadi dua bagian dengan karakter CR *(carriage return)* dan LF *(line feed)* yang dimasukkan ke dalam *header* HTTP yang tidak divalidasi. Ini memungkinkan penyerang untuk mengendalikan *header* dan *body* dari *respons* HTTP yang diterima dari aplikasi *web*. Akibatnya, server atau aplikasi bisa memberikan *respons* yang tidak diharapkan, dan ini dapat memfasilitasi serangan seperti XSS atau pencurian datasensitif oleh penyerang [11].

G. *Log-Based Alerts*

Membuat kebijakan *alerting* berbasis metrik digunakan untuk melacak data metrik yang dikumpulkan *Cloud Monitoring* untuk memberikan notifikasi ketika ambang batas peristiwa dan metrik tercapai. Kebijakan ini dapat memiliki satu atau lebih kondisi untuk memicu *alerting* dan akan membuat insiden yang terlihat di konsol *Cloud Monitoring*. Ada beberapa tindakan yang dapat dipilih untuk menangani insiden terbuka, seperti mengakui insiden sebagai masalah yang diketahui, menyaring kondisi yang terkait, melihat kebijakan *alerting*, dan mengedit kebijakan *alerting*. Dalam hal notifikasi, konfigurasi dapat dilakukan untuk mengirimkan email, SMS, dan berbagai bentuk pemberitahuan lainnya. Untuk memberikan panduan mengenai langkah-langkah yang dapat diambil untuk menangani situasi, tautan menuju dokumentasi juga dapat disertakan dalam peringatan tersebut [8].

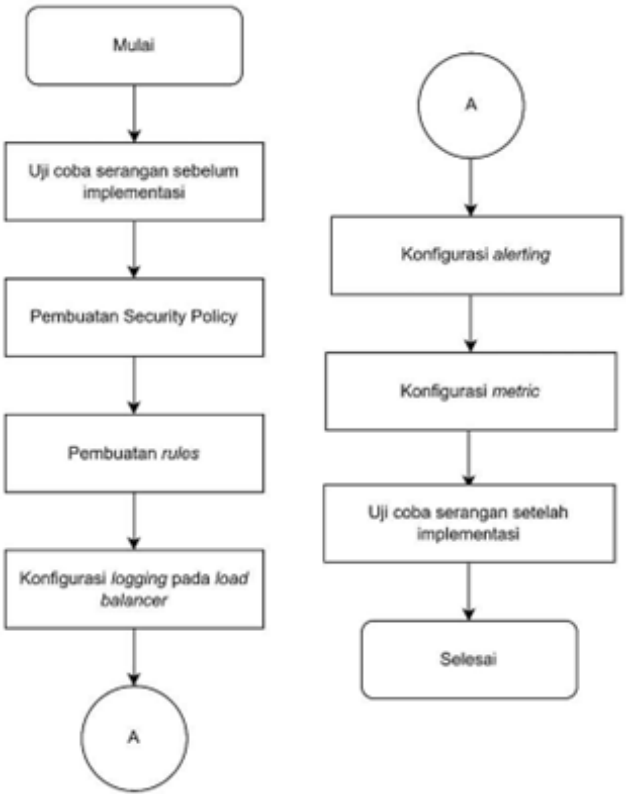
H. *Log-Based Metric*

Log memiliki potensi untuk digunakan sebagai dasar pembuatan metrik. Dalam *Cloud Logging*, *log* dapat dikumpulkan sesuai dengan filter yang ditetapkan setiap kali ada kecocokan pola tertentu. Informasi yang dihasilkan dari *log* ini kemudian akan diungkapkan dalam Monitoring dan bisa dimanfaatkan lebih lanjut untuk membuat tampilan informasi dan kebijakan peringatan. Sebagai ilustrasi, *log* yang berisi pesan kesalahan 404 tertentu dapat dihitung selama satu menit dan diperlihatkan sebagai ukuran atau metrik. Metrik berbasis *log* ini bisa termasuk metrik sistem, yang telah ditetapkan sebelumnya oleh *Cloud Logging*, atau metrik yang dibuat oleh pengguna berdasarkan proyek tertentu dengan mengatur kriteria penyaringan tertentu [8].

III. METODOLOGI

A. Tahapan Teknik

Tahapan teknik merupakan alur teknis dari penelitian ini. Tahapan teknik dalam penelitian ini disajikan pada Gambar 1.



Gambar 1.Tahapan teknik penelitian

Tahapan ini dimulai dengan melakukan uji coba serangan sebelum implementasi, yang melibatkan simulasi serangan untuk mengidentifikasi potensi kerentanan pada aplikasi *web* fishku. Selanjutnya, langkah pembuatan *security policy* dilakukan, di mana aturan- aturan keamanan yang mengatur lalu lintas aplikasi diformulasikan. Proses pembuatan *rules* dilanjutkan untuk mengidentifikasi pola serangan yang harus diantisipasi oleh WAF. Pada tahap konfigurasi *logging* pada *load balancer*, parameter pengumpulan data *log* diatur untuk mengawasi lalu lintas dan aktivitas aplikasi secara mendalam. Langkah selanjutnya adalah mengonfigurasi *alerting*, di mana mekanisme pemberitahuan otomatis diaktifkan saat terdeteksi aktivitas yang mencurigakan.

Pengaturan *metric* juga dilakukan untuk mengukur dan menganalisis performa serta keamanan aplikasi. Tahapan akhir melibatkan uji coba serangan setelah implementasi, di mana pengujian ulang dilakukan untuk menguji efektivitas *Web Application Firewall* dalam menanggapi serangan yang sebelumnya telah diidentifikasi. Selama seluruh tahapan ini, langkah-langkah teknis ini mengarah pada implementasi yang kuat dan lebih aman untuk aplikasi Fishku, dengan menggunakan *Web Application Firewall* berbasis *Google Cloud Armor*.

B. Aplikasi Fishku

Aplikasi Fishku merupakan *platform e-commerce* perikanan yang menonjolkan fitur pendeteksian kesegaran ikan melalui teknologi *machine learning*. Interaksi antarmuka aplikasi dirancang untuk mempermudah koneksi antara nelayan dan pembeli, menciptakan pengalaman yang lancar dan efisien. Aplikasi ini juga menghadirkan layanan deteksi kesegaran ikan sebagai solusi untuk memastikan kualitas produk yang dijual. Aplikasi Fishku beroperasi di lingkungan *Google Cloud Platform* (GCP) dengan pendekatan *serverless* yang mengoptimalkan efisiensi dan skalabilitas. Saat ini, Aplikasi Fishku terakreditasi status SLSA 3 yang menandakan *platform* sumber dan lingkungan memenuhi standar khusus untuk menjamin kemampuan perlindungan yang jauh lebih kuat terhadap gangguan dibandingkan level sebelumnya dengan mencegah kelompok ancaman tertentu, seperti kontaminasi *cross-build*. menggunakan layanan *Cloud Run* untuk menjalankan aplikasi sesuai permintaan tanpa mengurus infrastruktur fisik. Pengelolaan basis data dan informasi penting dikelola oleh *Cloud SQL* dan *Cloud Storage*, menawarkan keandalan dan skalabilitas dalam manajemen data. Keamanan aplikasi dijaga melalui pemantauan aktif dan konfigurasi tepat menggunakan *GCP Console*, memastikan integritas data. Kolaborasi teknologi dan inovasi mendasari lahirnya Aplikasi Fishku, sebuah solusi *e-commerce* aman dan handal dalam industri perikanan

C. Implementasi Pembuatan *Security Policy*

Dalam lingkungan *Google Cloud Platform* (GCP), terutama dengan menggunakan fitur *Cloud Armor*, langkah-langkah untuk membuat kebijakan keamanan adalah sebagaiberikut

1. Masuk ke panel *Cloud Armor* dan pilih opsi 'Buat Kebijakan Keamanan.'
2. Beri nama kebijakan yang akan dibuat dan tambahkan deskripsi kebijakan tersebut. Pilih jenis kebijakan sebagai 'Kebijakan keamanan *backend*' sesuai dengan kebutuhan. Selanjutnya, atur tindakan *default* ketika permintaan tidak sesuai dengan aturan yang telah ditetapkan. Atur tindakan *default* menjadi 'deny' dan kode *respons* menjadi '403 *Forbidden*.' Klik 'Lanjut' untuk melanjutkan konfigurasi kebijakan keamanan.
3. Buat aturan *default* pertama untuk mengizinkan akses dari semua alamat IP. Pilih 'Mode Dasar', masukkan karakter '*' pada kolom 'Cocok', dan pilih 'Izinkan'. Klik 'Selesai' untuk menyelesaikan konfigurasi aturan.
4. Klik 'Selesai' untuk meninjau ringkasan dan melanjutkan ke langkah selanjutnya.
5. Tambahkan target, klik 'SELESAI' dan 'BUAT KEBIJAKAN.'
6. Kebijakan Keamanan berhasil dibuat.

Untuk informasi lebih lanjut, klik nama kebijakan,. Tindakan ini akan membawa ke rincian lebih lanjut mengenaikebijakan yang telah dibuat. Selain itu, Kebijakan Keamanan ini akan menerapkan aturan-aturan keamanan pada *web load balancer*, sehingga melindungi situs *web* dari lalu lintas berbahaya.

D. Pembuatan *Rules Block Local File Inclusion*

Proses konfigurasi untuk mengatasi serangan LFI dapat dilakukan melalui serangkaian langkah sebagai berikut. Langkah pertama, menuliskan deskripsi jenis *rules* yang akan diterapkan di sini penulis menambahkan deskripsi "*Block Local File Inclusion*". Kemudian, menggunakan fitur *Advanced Mode* dalam konfigurasi. Pada bagian ini, menambahkan aturan dengan menggunakan *syntax* khusus yang ditunjukkan pada Gambar 2.

```
evaluatedPreconfiguredExpr('lfi-stable')
```

Gambar 2. *Syntax* pembuatan *rules block local file inclusion*

Setelah menambahkan *syntax*, langkah berikutnya adalah memilih tindakan yang akan diambil jika serangan terdeteksi. Dalam hal ini, memilih tindakan "Deny" untuk menolak permintaan yang sesuai dengan kondisi serangan LFI juga akan mengatur kode *respons* menjadi "403 *Forbidden*" untuk menunjukkan bahwa akses ditolak. Terakhir, perlu menentukan prioritas aturan. Prioritas ini menentukan urutan eksekusi aturan dalam daftar. Dalam contoh ini, aturan LFI akan diberi prioritas 9000 untuk memastikan bahwa ia diterapkan dengan benar.

E. Pembuatan *Rules Block Vulnerability Scanner*

Proses implementasi membuat *rules* adalah menuliskan deskripsi yakni "*Block Vulnerabiliy Scanner*". Selanjutnya, pilih *mode advanced* dalam pengaturan, dan gunakan *syntax* spesifik yaitu ditunjukkan pada Gambar 3.

```
evaluatedPreconfiguredExpr('scannerdetectio
n-stable')
```

Gambar 3. *Syntax* pembuatan *rules block vulnerability scanner*

Syntax ini akan mengaktifkan kondisi yang telah dipersiapkan sebelumnya oleh *platform* keamanan untuk mendeteksi serangan dari *vulnerability scanner*. Setelah kondisi ditambahkan, langkah berikutnya adalah memilih tindakan yang akan diambil jika serangan terdeteksi. Pilih tindakan "*Deny*" untuk menolak permintaan yang sesuai dengan kondisi serangan dari *vulnerability scanner*. Tidak hanya itu, perlu dilakukan pengaturan kode *respons* menjadi "403 *Forbidden*" untuk menunjukkan bahwa akses ditolak. Terakhir, tentukan prioritas aturan. Prioritas ini menentukan urutan eksekusi aturan dalam daftar. Dalam contoh ini, aturan untuk menghadapi serangan dari *vulnerability scanner* akan diberi prioritas 9001 untuk memastikan bahwa aturan ini diterapkan dengan benar setelah aturan lainnya.

F. Pembuatan *Rule Block Protocol Attack*

Di dalam *Google Cloud Armor*, prosedur konfigurasi adalah sebagai berikut. Pertama, mengisikan deskripsi yaitu "Block *Protocol Attack*". Dalam *mode advanced*, gunakan *syntax* spesifik yang ditunjukkan pada Gambar 4.

```
evaluatedPreconfiguredExpr('protocolattack-
stable')
```

Gambar 4. *Syntax* pembuatan *rule block protocol attack*

Setelah kondisi ditambahkan, langkah selanjutnya adalah memilih tindakan saat serangan terdeteksi. Pilih opsi "Deny" untuk menolak permintaan sesuai kondisi serangan *protocol attack*. Tetapkan juga kode *respons* menjadi "403 *Forbidden*" agar terlihat bahwa akses ditolak. Lalu, atur prioritas urutan dalam daftar. Pada contoh ini, berikan prioritas 9003 pada aturan yang ditetapkan untuk menghadapi serangan *protocol attack*, agar aturan ini diterapkan dengan benar setelah aturan lainnya.

G. Konfigurasi *Load Balancer*

Mengaktifkan pada *load balancer* adalah langkah pertama untuk mengumpulkan *log*, dan kemudian modifikasi *log query* di *Google Cloud Armor* adalah langkah berikutnya untuk menganalisis dan memfilter *log* yang telah dikumpulkan. Kombinasi kedua langkah ini membantu mengamankan lingkungan sistem dengan mendapatkan. Pada bagian *log query* modifikasi *query* tersebut seperti pada Gambar 5.

```
ANDjsonPayload.enforcedSecurityPolicy.
configuredAction:(DENY)
```

Gambar 5. Konfigurasi *load balancer*

Dengan melakukan konfigurasi ini, *load balancer* akan mulai mencatat dan mencatat aktivitas serta trafik yang melintasi *backend*. Langkah ini memungkinkan untuk memantau dan menganalisis data *logging* guna mendapatkan wawasan yang lebih baik tentang penggunaan layanan, serta potensi ancaman atau masalah keamanan. Pada tahap selanjutnya, data *logging* yang telah terkumpul akan dapat digunakan untuk analisis lebih lanjut dan pengambilan tindakan yang sesuai

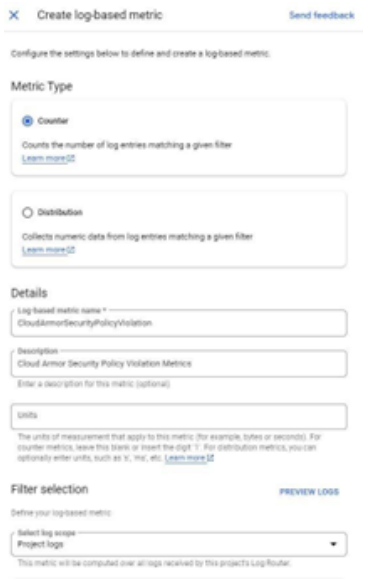
H. Pembuatan *Rules Log-Based Alerts*

Konfigurasi *alerting* pada *Google Cloud Armor* bertujuan untuk menerima notifikasi atau pemberitahuan ketika kebijakan keamanan yang telah dibuat dalam Cloud Armor mendeteksi serangan atau aktivitas yang mencurigakan. Pada langkah ini, perlu memastikan bahwa kueri *log* yang telah dimasukkan benar dan sesuai dengan kebutuhan. Dalam konteks pengaturan keamanan *Cloud Armor*, langkah pertama adalah mengonfigurasi notifikasi *log*. Dapat menentukan frekuensi notifikasi dan opsi penutupan otomatis untuk memudahkan manajemen notifikasi. Setelah konfigurasi selesai, simpan kebijakan notifikasi yang telah dibuat. Selanjutnya, konfigurasi notifikasi ke alamat email dengan mengedit saluran pemberitahuan. Masukkan alamat email dan nama yang sesuai. Terakhir, kembali ke konfigurasi kebijakan dan pilih alamat email yang telah ditambahkan pada saluran pemberitahuan. Dengan mengatur *alerting* ini, dapat memantau aktivitas pada layanan HTTP *Load Balancer*. Notifikasi akan dikirimkan melalui email saat ada aktivitas mencurigakan atau potensi serangan pada layanan aplikasi *web*. Ini memungkinkan *respons* yang cepat dan perlindungan sistem dari ancaman yang mungkin terjadi.

I. Konfigurasi *Load-Based Metric*

Pada konfigurasi *metric* dalam *Google Cloud Armor*, langkah-langkahnya melibatkan pengaturan metrik untuk memonitor aktivitas keamanan pada layanan *Load Balancer*.Dalam hal ini, dapat memanfaatkan *Log Explorer* dengan membuat kueri *log* spesifik yang menganalisis aktivitas terkait dengan kebijakan keamanan yang memiliki tindakan penolakan akses (*DENY*). Berikut adalah konfigurasi yang diperlukan disajikan pada Gambar 6. Bagian ini untuk verifikasi *kueri log*. Setelah semua selesai, maka Langkah terakhir adalah klik "*Create Metric*".

Dengan mengonfigurasi metrik ini, memiliki pemantauan aktif terhadap aktivitas yang berkaitan dengan kebijakan keamanan yang diterapkan pada *Load Balancer*. Ini membantu mengidentifikasi potensi ancaman atau insiden keamanan dengan lebih cepat, sehingga, dapat me*respons* dengan tindakan yang tepat untuk melindungi layanan aplikasi *web*.



Gambar 6. Konfigurasi *log-based metric*

J. Pengujian Penelitian

1. Uji Coba Serangan *Local File Inclusion*

Serangan *Local File Inclusion* (LFI) adalah teknik serangan di mana penyerang mencoba memanipulasi URL dengan mengirimkan parameter yang mengarahkan ke file sistem yang seharusnya tidak dapat diakses oleh pengguna. Langkah-langkah pengujian sebagai berikut :

a. Menggunakan perintah 'curl' di *Kali Linux* dengan perintah yang ditunjukkan pada Gambar 7.

```
$ curl -Ii (INPUT BASE
URL)/loadImage?filename=../../../
etc/passw d
```

Gambar 7. Perintah 'curl' di *Kali Linux* untuk uji coba serangan *local file inclusion*

b. Tujuan dari perintah ini adalah untuk menguji apakah sistem atau server memiliki kerentanan terhadap serangan yang melibatkan manipulasi direktori atau *path traversal*. *Path traversal* adalah serangan di mana penyerang mencoba mendapatkan akses ke direktori atau file di luar jalur normal yang diizinkan.

c. Aplikasi Fishku akan memproses permintaan dan memberikan *respons* sesuai dengan kebijakan keamanan yang diimplementasikan oleh *Google Cloud Armor*.

Dengan melaksanakan uji serangan ini nantinya akan mengevaluasi *respons* dan perlindungan yang diberikan oleh *Google Cloud Armor* terhadap serangan *Local File Inclusion* (LFI). Apakah *Google Cloud Armor* dapat mencegah akses yang tidak sah ke direktori sistem yang sensitif melalui serangan LFI. Dari hasil uji coba ini, dapat menilai keberhasilan perlindungan yang diberikan oleh*Google Cloud Armor* terhadap serangan LFI.

2. Uji Coba Serangan *Vulnerability Scanner*

Serangan *Vulnerability Scanner* adalah upaya untuk mengidentifikasi kerentanan dalam sebuah aplikasi atau sistem. Sebelumnya aplikasi Fishku telah diimplementasikan perlindungan Google Cloud Armor. Langkah-Langkah Uji Coba :

a. Menjalankan perintah di *Kali Linux* menggunakan 'curl' seperti yang ditunjukkan pada Gambar 8.

```
$ curl -Ii (INPUT BASE
URL) -H "User- Agent:
```

Gambar 8. Perintah 'curl' di *Kali Linux* untuk uji coba serangan *vulnerability scanner*

b. Dalam perintah ini, menggunakan utilitas cURL untuk membuat permintaan HTTP ke URL. Opsi"-I" memerintahkan cURL hanya untuk mengambil *header respons* HTTP dan tidak mengunduh isi asli dari halaman. Opsi "-i" memungkinkan untuk melihat tampilan lengkap *header respons* dalam hasil, yang mencakup informasi tentang status *respons*, jenis konten, dan lainnya. Menggunakan opsi "-H" untuk mengatur *header* khusus dalam permintaan. *Header* yang diatur adalah "User-Agent: nikto", yang mengidentifikasi bahwa permintaan berasal dari alat pengujian keamanan Nikto. Tujuan dari uji coba ini adalah untuk melihat bagaimana sistem me*respons* serangan dari alat pengujian keamanan yang dikenal dan apakah ada kerentanan yang dapat dieksploitasi.

c. Permintaan ini akan diarahkan ke aplikasi Fishku yang telah diimplementasikan perlindungan oleh *Google Cloud Armor*.

Melalui pengujian penelitian ini, bertujuan untuk mengidentifikasi apakah *Google Cloud Armor* dapat mendeteksi dan menghalangi serangan dari alat *scanning* seperti Nikto. Dengan melakukan ini, dapat mengukur keberhasilan perlindungan *Google Cloud Armor* dalam menghadapi serangan yang mencoba untuk mengidentifikasi kerentanan pada aplikasi.

3. Uji Coba Serangan *Protocol Attack*

Pada bagian ini, menjelaskan uji coba yang akan dilakukan terhadap aplikasi Fishku dengan menggunakan teknik serangan *Protocol Attack* setelah menerapkan *Google Cloud Armor* sebagai lapisan keamanan tambahan. Serangan *protocol attack* melibatkan upaya untuk memanipulasi protokol komunikasi antara klien dan server. Dalam skenario ini, menyimulasikan serangan dengan menggunakan perintah *curl* yang dimodifikasi untuk mengganti karakteristik protokol HTTP. Langkah – Langkah uji coba serangan sebagai berikut:

a. Menjalankan perintah di *Kali Linux* menggunakan *curl* seperti yang ditunjukkan pada Gambar 9.

```
$ curl -Ii "(INPUT BASE URL)/index.html?foo=advanced%0d%0aContent-
Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-
Type:%20text/html%0d%0aContent-
Length:%2035%0d%0a%0d%0a<html>Sorry,%20System%20Down</html>"
```

Gambar 9. Perintah di *Kali Linux* untuk uji coba serangan *protocol attack*

b. Dalam perintah *curl* tersebut, karakter *newline* (%0d%0a) adalah representasi URL:-encoded dari karakter *newline* (CR-LF) yang disisipkan dalam URL. Ini bisa digunakan untuk mencoba memanipulasi bagaimana server-server yang berinteraksi dengan permintaan tersebut akan me*respons* dengan cara yang tidak diharapkan atau bahkan berpotensi terkena kerentanan keamanan. Potensi hasil dari manipulasi semacam ini adalah mungkin terjadi perbedaan dalam interpretasi protokol antara server *web* dan server *frontend* atau *proxy*, yang dapat dimanfaatkan oleh penyerang.

c. Permintaan ini akan diarahkan ke aplikasi Fishku yang telah diimplementasikan perlindungan oleh *Google Cloud Armor*.

Melalui pengujian ini bertujuan untuk menguji apakah *Google Cloud Armor* dapat mendeteksi dan menghalangi serangan yang berusaha memanipulasiprotokol komunikasi antara klien dan server. Serta mengetahui apakah perlindungan ini berhasil dalam mencegah serangan yang mencoba memanipulasi protokol untuk mendapatkan akses yang tidak sah ke server.

IV. HASIL DAN PEMBAHASAN

Hasil dalam bab ini, akan dibahas secara mendalam mengenai implementasi *Web Application Firewall* (WAF) pada aplikasi Fishku berbasis *Google Cloud Armor*. Penelitian ini mengarah pada tujuan utama yaitu melindungi aplikasi Fishku dari serangan-serangan berbahaya seperti *Local File Inclusion* (LFI), *Vulnerability Scanner*, dan *Protocol Attack*. Selain itu, telah dilakukan konfigurasi yang melibatkan penggunaan *load balancers*, penerapan *log-based alerts*, dan pengukuran metrik untuk mendukung upaya keamanan:

A. Hasil Uji Coba Serangan tanpa *Google Cloud Armor*

Dalam bab ini, akan dibahas secara mendalam mengenai implementasi *Web Application Firewall* (WAF) pada aplikasi Fishku berbasis *Google Cloud Armor*. Penelitian ini mengarah pada tujuan utama yaitu melindungi aplikasi Fishku dari serangan-serangan berbahaya seperti *Local File Inclusion* (LFI), *Vulnerability Scanner*, dan *Protocol Attack*. Berikut adalah Tabel 1 hasil uji serangan sebelum implementasi *security policy*.

Tabel 1. Hasil uji serangan sebelum implementasi *Google Cloud Armor*

Jenis Serangan	***Web Application Firewal***		**Keterangan**
	Pertama (tanpa *Google Cloud Armor*)	**Kedua (dengan *Google Cloud Armor*)**	
Local File Inclusion (LFI)	Berhasil	-	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 200 OK
Vulnerability Scanner	Berhasil	-	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 200 OK
Protocol Attack	Berhasil	-	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 200 OK

1. Hasil Serangan *Local File Inclusion* tanpa *Google Cloud Armor*

   Serangan LFI menjadi fokus penelitian untuk mengidentifikasi potensi kerentanan dalam aplikasi web Fishku. Berikut ini adalah hasil yang diperoleh dari pengujian serangan LFI sebelum adanya penerapan kebijakan keamanan.

   Pada pengujian serangan LFI, dilakukan pemanfaatan celah pada mekanisme keamanan aplikasi web Fishku guna mencoba mengakses file sensitif di dalam server. *Payload* digunakan untuk mencoba mengakses file *password* yang terletak di direktori /etc pada server Fishku.

   Proses ini bertujuan untuk mengidentifikasi kemampuan aplikasi dalam mencegah akses tidak sah ke file sensitif. Jika respons HTTP yang diterima adalah 200 OK, maka ini mengindikasikan keberhasilan *payload* dalam melakukan *traversal* direktori dan mengakses file/*etc/password*. Temuan ini mengungkap potensi kelemahan dalam keamanan yang bisa dieksploitasi oleh penyerang untuk memperoleh informasi sensitif [9]. Hasil serangan LFI sebelum penerapan *Google Cloud Armor* memiliki dampak serius terhadap keamanan, karena penyerang bisa mendapatkan akses ke file-file sensitif seperti *password*. Hal ini membuka risiko potensial terhadap serangan lebih lanjut pada sistem. Temuan respons HTTP 200 OK juga menunjukkan bahwa aplikasi belum memiliki mekanisme keamanan yang memadai untuk melindungi dari serangan LFI. Dengan adopsi *Google Cloud Armor*, diharapkan bahwa kelemahan ini dapat diminimalkan, dengan sistem mendeteksi dan mencegah serangan LFI sebelum mencapai aplikasi.

2. Hasil Serangan *Vulnerability Scanner* tanpa *Google Cloud Armor*

   Hasil uji serangan menggunakan *Vulnerability Scanner* sebelum penerapan kebijakan keamanan menunjukkan adanya potensi celah dalam mekanisme keamanan aplikasi. Pada pengujian ini, pengiriman permintaan dengan *header "User-Agent:* nikto" dilakukan untuk mengevaluasi kerentanan aplikasi terhadap serangan tertentu. Respons HTTP 200 OK yang diterima menunjukkan bahwa kebijakan keamanan pada saat itu belum mampu mendeteksi dan mencegah akses dari alat *Vulnerability Scanner*. Temuan ini mengindikasikan celah dalam pertahanan sistem yang memungkinkan alat seperti Nikto mendapatkan respons yang diinginkan [12].

   Respons HTTP 200 OK dalam pengujian ini tidak sesuai dengan harapan, mengindikasikan bahwa mekanisme keamanan pada sistem belum cukup kuat untuk melawan serangan otomatis atau dari alat seperti Nikto. Hal ini menandakan bahwa kebijakan keamanan yang ada tidak dapat dalam mendeteksi dan mencegah serangan dari alat tersebut. Dalam pengujian ini menunjukkan urgensi untuk mengambil langkah-langkah perlindungan yang lebih kuat dan memadai, termasuk implementasi *Google Cloud Armor*, untuk melindungi aplikasi dari ancaman serangan yang memanfaatkan celah keamanan.

3. Hasil Serangan *Protocol Attack* tanpa *Google Cloud Armor*

   Hasil uji serangan *Protocol Attack* sebelum penerapan kebijakan keamanan menunjukkan bahwa protokol keamanan server tidak mampu mendeteksi dan mencegah upaya manipulasi protokol HTTP dengan karakteristik tertentu.

   Dalam pengujian ini, serangan dilakukan dengan mengirimkan permintaan manipulatif yang berisi karakteristik yang dapat memicu respons yang tidak semestinya. karakteristik manipulatif seperti %0d%0a digunakan untuk memanipulasi *header* permintaan HTTP [11]. Respons yang diharapkan seharusnya adalah kode status HTTP 200 OK, yang menunjukkan bahwa manipulasi karakteristik protokol berhasil.

   Namun, hasil uji serangan ini mengindikasikan bahwa protokol keamanan pada server tidak mampu mendeteksi dan mencegah tindakan manipulasi karakteristik protokol tersebut. Respons yang diterima adalah kode status HTTP 200 OK, yang seharusnya tidak seharusnya terjadi dalam kondisi normal. Temuan ini memberikan catatan penting tentang potensi celah dalam mekanisme keamanan server terhadap manipulasi protokol HTTP.

B. Hasil Uji Coba Serangan dengan *Google Cloud Armor*

Pada sub bab ini, akan diuraikan hasil dari uji coba serangan setelah dilakukan implementasi kebijakan keamanan menggunakan *Google Cloud Armor* pada aplikasi Fishku. Hasil yang diperoleh akan dianalisis untuk mengevaluasi perlindungan yang diberikan oleh *Google Cloud Armor* terhadap serangan-serangan yang telah direncanakan. Berikut ini Tabel 2 merupakan hasil uji serangan sesudah implementasi *security policy*.

Tabel 2. Hasil uji serangan sesudah implementasi *Google Cloud Armor*

Jenis Serangan	***Web Application Firewal***		**Keterangan**
	Pertama (tanpa *Google Cloud Armor*)	**Kedua (dengan *Google Cloud Armor*)**	
Local File Inclusion (LFI)	-	Berhasil	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 403
Vulnerability Scanner	-	Berhasil	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 403
Protocol Attack	-	Berhasil	*Syntax* dikirimkan kepada server dan hasil respons HTTP/2 403

1. Hasil Serangan *Local File Inclusion* dengan *Google Cloud Armor*

Pengujian dilakukan dengan menjalankan perintah pada lingkungan Kali Linux yang memanfaatkan celah LFI pada aplikasi Fishku. Dalam implementasi *Google Cloud Armor*, terdapat mekanisme yang memungkinkan penilaian terhadap ekspresi keamanan tertentu. Pada kasus ini, terdapat suatu aturan (*rules*) atau kondisi yang menggambarkan potensi serangan LFI dengan nama atau kode *lfi-stable* dalam konfigurasi *Google Cloud Armor* [13]. Evaluasi ekspresi tersebut kemungkinan memeriksa apakah ada karakteristik dari URL yang mencoba melakukan *traversal* direktori seperti ../../../, dan jika ada, sistem akan menolak akses dengan respons HTTP 403 *Forbidden*. Respons HTTP 403 *Forbidden* pada serangan LFI setelah implementasi *Google Cloud Armor* mengindikasikan bahwa aturan-aturan keamanan yang diatur dalam konfigurasi *Google Cloud Armor* telah berhasil mengidentifikasi dan mencegah upaya serangan LFI. Implikasinya, kebijakan keamanan yang diimplementasikan melalui *Google Cloud Armor* telah memberikan perlindungan yang kuat terhadap potensi celah LFI dan mampu mendeteksi serta mencegah serangan sebelum mencapai aplikasi *web* Fishku. Hal ini berdampak positif pada peningkatan tingkat keamanan keseluruhan dari aplikasi web Fishku [8].

2. Hasil Serangan *Vulnerability Scanner* dengan *Google Cloud Armor*

Pada sub bab ini, akan diuraikan hasil dan pembahasan dari uji coba serangan *Vulnerability Scanner* setelah implementasi kebijakan keamanan menggunakan *Google Cloud Armor*. Uji coba ini bertujuan untuk mengidentifikasi kemampuan kebijakan keamanan dalam menghadapi serangan yang dilakukan oleh alat otomatis seperti *Vulnerability Scanner,* yang dalam kasus ini akan diwakili oleh alat "Nikto".Web.

Pengujian dilakukan dengan menjalankan perintah pada lingkungan Kali Linux yang memanfaatkan alat "*curl*" untuk mengirimkan permintaan HTTP dengan *header "User-Agent*: Nikto". Dalam perintah ini, flag-1 digunakan untuk hanya mengambil *header* respons dari server, sedangkan flag-H digunakan untuk menambahkan *header* khusus ke dalam permintaan HTTP. *Header* yang ditambahkan dalam contoh ini adalah "*User-Agent*: Nikto", yang mengubah informasi pengenal alat yang melakukan permintaan menjadi "Nikto".

Alat "Nikto" adalah sebuah alat *open-source* yang digunakan untuk mengidentifikasi kerentanan potensial pada aplikasi web [12]. Hasil respons dari server menunjukkan status kode 403 (*Forbidden*), yang mengindikasikan bahwa sistem telah berhasil mencegah upaya akses oleh alat "Nikto". Penerapan mekanisme keamanan yang tepat dalam konfigurasi Gogle Cloud Armor mampu mendeteksi karakteristik dari alat "Nikto" dan mencegah upaya pemindaian atau penetrasi yang tidak sah [8].

3. Hasil *Protocol Attack* dengan *Google Cloud Armor*

Pada uji coba ini, menggunakan serangan *Protocol Attack* yang berfokus pada memanipulasi protokol komunikasi dengan memasukkan karakter khusus seperti %0d%0a yang merepresentasikan karakter *Carriage Return* (CR) dan *Line Feed* (LF). Serangan ini mencoba memanipulasi protokol untuk menciptakan respons palsu dari server. Uji coba dilakukan dengan menjalankan perintah pada lingkungan Kali Linux yang memanfaatkan alat "*curl*" untuk mengirimkan permintaan HTTP yang memasukkan karakter %0d%0a ke dalam parameter "*foo*". kode status HTTP 403 menunjukkan bahwa akses ditolak (*Forbidden*). Dalam konteks ini, serangan *Protocol Attack* yang menggunakan teknik HTTP *Splitting* mencoba memanipulasi karakteristik protokol HTTP dengan menyisipkan karakter %0d%0a untuk menciptakan respons palsu yang diinginkan [14]. Namun, mekanisme keamanan yang diimplementasikan dalam *Google Cloud Armor* berhasil mendeteksi anomali ini dan menolak akses dengan respons HTTP 403

Forbidden. Dengan demikian, hasil pengujian ini memberikan bukti bahwa implementasi kebijakan keamanan dalam *Google Cloud Armor* mampu melindungi aplikasi web Fishku dari serangan *Protocol Attack* yang berupaya memanipulasi protokol komunikasi [8]. Perlindungan ini memiliki dampak positif dalam meminimalkan risiko potensial dari serangan semacam itu, sehingga integritas dan keamanan aplikasi web tetap terjaga dengan baik.

C. Hasil Perbandingan Sebelum dan Sesudah Implementasi *Google Cloud Armor*

Pada pengujian ini telah melakukan serangkaian uji coba serangan *Local File Inclusion* (LFI), *Vulnerability Scanner*, dan *Protocol Attack* sebelum dan sesudah implementasi *Google Cloud Armor*. Hasil pengujian ini menunjukkan bahwa sebelum implementasi *Google Cloud Armor* pengujian serangan memberikan respons HTTP 200 OK yang menandakan bahwa ketiga jenis serangan tersebut diterima oleh server. Hal ini menunjukkan bahwa kemampuan keamanan Aplikasi Web Fishku yang masih rentan terhadap ancaman serangan. Sebaliknya, setelah implementasi *Google Cloud Armor* dan melakukan konfigurasi aturan *security policy*. Serangan LFI, *Vulnerability Scanner*, dan *Protocol Attack* telah berhasil diatasi. Respons HTTP 403 *Forbidden* menandakan bahwa kebijakan keamanan *Google Cloud Armor* berhasil mendeteksi dan mencegah akses dari serangan-serangan tersebut. Implementasi *Google Cloud Armor* memberikan perlindungan yang efektif terhadap potensi ancaman keamanan dan meningkatkan tingkat keamanan aplikasi web Fishku secara keseluruhan. Berikut adalah Tabel 3 hasil perbandingan dari pengujian.

Tabel 3. Hasil perbandingan sebelum dan sesudah implementasi *Google Cloud Armor*

Pengujian Serangan	**Tanpa *Google Cloud Armor***		**Dengan *Google Cloud Armor***	
	Diterima	**Ditolak**	**Diterima**	**Ditolak**
LFI (1)	200	0	0	403
Vulnera (1)	200	0	0	403
Protocol Attact (1)	200	0	0	403
LFI (2)	200	0	0	403
Vulnera (2)	200	0	0	403
Protocol Attact (2)	200	0	0	403

D. Hasil *Log-Base Alerts*

Pada sub-bab ini, akan dijelaskan hasil dari analisis implementasi *alerting* yang telah dilakukan setelah penerapan *security policy* pada *Google Cloud Armor*. Tujuan dari analisis ini adalah untuk mengidentifikasi bagaimana sistem *alerting* dapat mendeteksi dan memberikan pemberitahuan terkait aktivitas mencurigakan atau serangan yang terjadi pada aplikasi web Fishku. Sebelum membahas hasil analisis, berikut adalah konfigurasi *alerting* yang telah diterapkan:

1. *Log Query*: Digunakan untuk mencari pola aktivitas yang mencurigakan pada log.
2. *Alert Policy*: Kebijakan yang mendefinisikan kriteria yang harus terpenuhi agar sebuah pemberitahuan atau notifikasi dapat dikirimkan.

Alerting melalui email memiliki peran krusial dalam menjaga keamanan dan ketersediaan sistem. Notifikasi *real-time* memungkinkan deteksi dini terhadap masalah atau ancaman, memungkinkan tindakan respons segera sebelum situasi memburuk. Dalam aspek keamanan, notifikasi seketika melalui email membantu mengidentifikasi dan menangani serangan atau upaya penetrasi dengan cepat. Dengan demikian, *alerting* melalui email adalah alat penting dalam menjaga kontrol penuh terhadap lingkungan teknologi dan mengambil langkah proaktif untuk menjaga kinerja dan keamanan yang optimal.

F. Hasil *Log-Based Metric*

Melalui penggunaan *Metric Explorer*, grafik dan visualisasi dapat digunakan untuk memahami pola dan tren yang muncul saat serangan terjadi. Dalam skripsi ini, hasil dari pengujian serangan LFI, *Vulnerability Scanner*, dan *Protocol Attack* akan dianalisis melalui metrik yang telah dihasilkan pada Gambar 10 sebagai berikut.

Gambar 10. Hasil grafik *metric monitoring*

Dalam gambar grafik metrik yang telah disajikan. Sebagai tambahan, bahwa dalam *Metric Explorer* ini memiliki opsi untuk mengunduh file CSV yang memuat data yang diambil dalam grafik ini. Dengan memanfaatkan fitur ini, dapat mengakses data yang digunakan dalam penghitungan metrik tersebut. Berikutnya, akan menyajikan hasil dari data respons serangan yang tercatat dalam *log metric* yang dimuat pada Gambar 11 berikut ini.

Waktu	Hasil	Keterangan
Wed Aug 23 2023 09:58:00	0	Tidak ada aktivitas yang terdeteksi
Wed Aug 23 2023 09:59:00	0.05	**Terdeteksi aktivitas dengan nilai respons 0.05 (5% dari skala respons)**
Wed Aug 23 2023 10:00:00	0.05	**Terdeteksi aktivitas dengan nilai respons 0.05 (5% dari skala respons)**
Wed Aug 23 2023 10:01:00	0	Tidak ada aktivitas yang terdeteksi
Wed Aug 23 2023 10:02:00	0	Tidak ada aktivitas yang terdeteksi
Wed Aug 23 2023 10:14:00	0	Tidak ada aktivitas yang terdeteksi
Wed Aug 23 2023 10:15:00	0.016666666666666666	**Terdeteksi aktivitas dengan nilai respons 0.0167 (1.67% dari skala respons)**
Wed Aug 23 2023 10:16:00	0.016666666666666666	**Terdeteksi aktivitas dengan nilai respons 0.0167 (1.67% dari skala respons)**
Wed Aug 23 2023 10:17:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:18:00	0.05	**Terdeteksi aktivitas dengan nilai respons 0.05 (5% dari skala respons)**
Wed Aug 23 2023 10:19:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:20:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:29:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:30:00	0.016666666666666666	**Terdeteksi aktivitas dengan nilai respons 0.0167 (1.67% dari skala respons)**
Wed Aug 23 2023 10:31:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:41:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:42:00	0.03333333333333333	**Terdeteksi aktivitas dengan nilai respons 0.0333 (3.33% dari skala respons)**
Wed Aug 23 2023 10:43:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 10:44:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 11:00:00	0	Tidak ada aktivitas yang terdeteksi pada interval ini.
Wed Aug 23 2023 11:01:00	0.016666666666666666	**Terdeteksi aktivitas dengan nilai respons 0.0167 (1.67% dari skala respons)**
Wed Aug 23 2023 11:02:00	0	Tidak ada aktivitas yang terdeteksi pada akhir periode.

Gambar 11. Detail aktivitas *log metric*

Analisis keseluruhan dari data menunjukkan pola aktivitas yang bervariasi selama rentang waktu yang diamati. Terlihat bahwa pada beberapa interval waktu, hasil respons mencapai nilai tertentu yang mengindikasikan adanya aktivitas pelanggaran dalam sistem. Secara keseluruhan, analisis data memberikan wawasan tentang pola, respons data, waktu aktivitas, dan intensitas aktivitas pelanggaran dalam sistem.

V. SIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan dengan judul "Implementasi WAF pada Aplikasi Fishku Berbasis *Google Cloud Armor*", beberapa kesimpulan dapat diambil, bahwa implementasi *Web Application Firewall* (WAF) menggunakan layanan *Google Cloud Armor* berhasil meningkatkan tingkat keamanan aplikasi web Fishku terhadap jenis serangan, termasuk *Local File Inclusion* (LFI), *Vulnerability Scanner*, dan *Protocol Attack*. Penerapan WAF ini menghasilkan perubahan signifikan pada respons server dari sebelumnya yang cenderung positif (respons kode 200) menjadi lebih aman dengan penolakan akses (respons kode 403), menunjukkan efektivitas perlindungan yang lebih baik. Pengujian juga melibatkan konfigurasi *Load Balancer*, yang memungkinkan distribusi lalu lintas aplikasi secara efisien di antara beberapa sumber daya server. Hal ini dapat meningkatkan ketersediaan dan kinerja aplikasi, serta mengoptimalkan tata letak data untuk mengurangi risiko *overloading* pada satu server. Penggunaan fitur *Alerting* juga telah memberikan dampak positif dalam menginformasikan secara cepat jika terjadi serangan atau aktivitas mencurigakan. Notifikasi melalui email memungkinkan respons cepat dari tim keamanan untuk mengidentifikasi dan merespons ancaman segera setelah terdeteksi. Analisis data metrik dan visualisasi grafik menjadi alat yang penting dalam pemantauan kinerja aplikasi dan deteksi dini ancaman. Metrik performa yang diambil dari *Google Cloud Platform* memberikan wawasan yang lebih dalam tentang bagaimana aplikasi beroperasi.

REFERENSI

[1] A. Aprilia and S. Subiyantoro, "Peluang dan Tantangan : Bisnis di era disrupsi industri," J. Eduscience, vol. 9, no. 2, pp. 377–387, 2022, doi: 10.36987/jes.v9i2.2820.

[2] M. Yunus, "Analisis Kerentanan Aplikasi Berbasis Web Menggunakan Kombinasi Security Tools Project Berdasarkan Framework Owasp Versi 4," J. Ilm. Inform. Komput., vol. 24, no. 1, pp. 37–48, 2019, doi: 10.35760/ik.2019.v24i1.1988.

[3] I. Barokah and A. Asriyanik, "Analisis Perbandingan Serverless Computing Pada Google Cloud Platform," J. Teknol. Inform. dan Komput., vol. 7, no. 2, pp. 169– 187, 2021, doi: 10.37012/jtik.v7i2.662.

[4] Ika, "Student-Made E-Commerce Platform Fi$hku Enables Easy Trading of Fishery Products," Universitas Gadjah Mada, 2022. https://www.ugm.ac.id/en/news/22980-student- made-e-commerce-platform-fi-hku-enables-easy- trading-of-fishery-products (accessed Apr. 13, 2023).

[5] R. A. Muzaki, O. C. Briliyant, M. A. Hasditama, and H. Ritchi, "Improving Security of Web-Based Application Using ModSecurity and Reverse Proxy in Web Application Firewall," 2020 Int. Work. Big Data Inf. Secur. IWBIS 2020, pp. 85–90, 2020, doi: 10.1109/IWBIS50925.2020.9255601.

[6] OWASP, "Top 10 Web Application Security Risks," OWASP, 2021. https://owasp.org/www-project-top- ten/ (accessed Aug. 29, 2023).

[7] B. Reddy Bhimireddy, A. Nimmagadda, H. Kurapati, L. Reddy Gogula, R. Rani Chintala, and V. Chandra Jadala, "Web Security and Web Application Security: Attacks and Prevention," 2023 9th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2023, vol. 1, pp. 2095–2099, 2023, doi: 10.1109/ICACCS57279.2023.10112741.

[8] B. Gerrad and K. Clapa, Professional Cloud Architect- Google Cloud Certification Guide. Packt Publishing Ltd., 2019.

[9] M. Chawda, D. P. Sharma, and M. J. Patel, "Deep Dive into Directory Traversal and File Inclusion Attacks leads to Privilege Escalation," Int. J. Sci. Res. Sci. Eng. Technol., vol. 4099, pp. 115–120, 2021, doi: 10.32628/ijsrset218384.

[10] B. Zukran and M. M. Siraj, "Performance Comparison on SQL Injection and XSS Detection using Open Source Vulnerability Scanners," 2021 Int. Conf. Data Sci. Its Appl. ICoDSA 2021, pp. 61–65, 2021, doi: 10.1109/ICoDSA53588.2021.9617484.

[11] K. Bremberg, "HTTP response splitting exploitations and mitigations," Detectify, 2019, [Online]. Available: https://blog.detectify.com/2019/06/14/http-response- splitting-exploitations-and-mitigations/

[12] M. I. Irawan, U. Y. K. Hediyanto, and , "Implementasi Keamanan Jaringan Pada Cloudfri Dengan Metode Hardening," eProceedings …, vol. 9, no. 2, pp. 644–649, 2022, [Online]. Available: https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/17632

[13] D. P. Iskandar, "Implementasi Web Application Firewall (WAF) Mod Security dan Pengujian Menggunakan SQL Injection l," no. December, 2020.

[14] S. Chavan, P. Jadhav, R. Mahajan, and K. Thopate, "Web Application Security Threats : SQL Injection Attack," vol. 12, no. 7, pp. 401–416.

Implementasi *Ansible* pada Otomasi *Honeypot Deployment* Berbasis Web

Dharma Kurniawan[1], Yuris Mulya Saputra[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
dharma.k@mail.ugm.ac.id
*Korespondensi: ym.saputra@ugm.ac.id;

Abstract – In this increasingly complex digital era, information system security has become a crucial issue for organizations across various industries. The rising number of cyberattacks and their evolving patterns pose threats to the confidentiality and integrity of an organization's data. In Indonesia, the National Cyber and Crypto Agency (BSSN) reported that as of April 2022, cyberattacks in Indonesia had reached 100 million cases, dominated by ransomware and malware attacks. One solution to address this problem is the implementation of a honeypot system. Honeypots have evolved into a valuable tool for identifying attacks and studying attacker strategies, enabling organizations to strengthen their defenses. However, manually implementing and managing honeypots can be time-consuming and resource-intensive. Therefore, this research aims to automate the honeypot deployment process using a popular configuration management tool known as Ansible. Additionally, to streamline Ansible's operation in honeypot deployment, a user-friendly web-based application has been created. This application not only deploys honeypots but also monitors the process. Based on the test results, the application successfully deployed honeypots to all sensors running on Google Cloud Platform across three different regions.

Keywords – Automation, Ansible, Web Application, Honeypot, Flask, Ansible AWX, Google Cloud Platform

Intisari – Dalam era digital yang semakin kompleks ini, keamanan sistem informasi menjadi masalah penting bagi organisasi di berbagai industri. Peningkatan serangan siber serta polanya yang semakin bervariatif bisa menimbulkan ancaman bagi kerahasiaan data atau integritas dari suatu organisasi. Di Indonesia, Badan Siber dan Sandi Negara (BSSN) mencatat bahwa hingga sepanjang bulan April 2022, serangan siber di Indonesia telah mencapai angka 100 juta kasus dengan jenis serangan yang didominasi oleh serangan *ransomware* dan *malware*. Salah satu solusi untuk mengatasi permasalahan ini adalah dengan mengimplementasikan sistem *honeypot*. *Honeypot* telah berkembang menjadi salah satu alat yang berguna untuk mengidentifikasi serangan dan mempelajari strategi penyerang, yang memungkinkan organisasi untuk memperkuat pertahanan mereka. Namun, penerapan dan pengelolaan *honeypot* secara manual dapat memakan banyak waktu dan sumber daya. Oleh karena itu, penelitian ini bertujuan untuk melakukan otomasi proses *honeypot deployment* dengan menggunakan alat manajemen konfigurasi yang populer yang dikenal sebagai *Ansible*. Selain itu, untuk mempermudah pengoperasian *Ansible* dalam melakukan *honeypot deployment*, maka dibuat sebuah aplikasi *user friendly* berbasis web. Aplikasi tidak hanya sekedar melakukan *honeypot deployment*, namun juga memantau prosesnya. Berdasarkan hasil pengujian, aplikasi berhasil melakukan *honeypot deployment* ke semua sensor yang berjalan di *Google Cloud Platform* dengan tiga *region* berbeda.

Kata kunci – Otomasi, *Ansible*, Aplikasi Web, *Honeypot, Flask, Ansible AWX, Google Cloud Platform*

I. PENDAHULUAN

Dalam era digital saat ini yang semakin berkembang pesat, keamanan sistem informasi merupakan salah satu isu yang menjadi perhatian penuh di berbagai sektor. Keamanan sistem informasi merupakan bidang yang mengacu pada perlindungan sistem informasi dari ancaman yang tidak disengaja atau disengaja selama proses normal operasinya, serta dari upaya untuk mencuri, mengubah, atau menghancurkan bagian-bagiannya [1]. Peningkatan serangan siber serta polanya yang semakin kompleks bisa menimbulkan ancaman bagi kerahasiaan data atau integritas dari suatu organisasi. Di Indonesia, Badan Siber dan Sandi Negara (BSSN) mencatat bahwa hingga sepanjang bulan April 2022, serangan siber di Indonesia telah mencapai angka 100 juta kasus dengan jenis serangan yang didominasi oleh serangan *ransomware* dan *malware* [2]. Menurut Naik, bentuk serangan siber beragam jenisnya, di antaranya *malware, phising, password attack, man-in-the-middle attack, SQL injection, DoS, crypto-jacking,* dan ancaman dari dalam organisasi itu sendiri [3]. Faktor dari terjadinya serangan siber seperti mengambil keuntungan dari data yang dieksploitasi dan yang paling utama adalah lemahnya sistem keamanan pada *server* [4]. Efek yang ditimbulkan bukanlah sesuatu yang bisa diremehkan, seperti kerugian finansial, kerugian reputasi, bahkan kehilangan data berharga tentu sangatlah mengerikan. Maka dari itu, untuk menciptakan sistem keamanan terbaik, organisasi harus memiliki mekanisme pengamanan yang efektif dalam mendeteksi, mengidentifikasi, serta mampu merespons serangan tersebut. *Honeypot* merupakan salah satu inovasi dalam dunia keamanan siber yang diciptakan untuk memancing penyerang serta merekam semua aktivitas mereka, kemudian data tersebut bisa diteliti dan dipelajari, sehingga strategi baru bisa diterapkan untuk memperkuat pertahanan sistem. Menurut Verma & Dubey, *honeypot* adalah sistem berwujud *virtual machine* yang menyimulasikan *service* atau *port* yang terbuka secara disengaja agar memiliki potensi untuk diserang dan diretas [5]. *Honeypot* memiliki cara kerja dengan menyerupai sistem atau layanan, sehingga mengalihkan perhatian penyerang agar lebih tertarik berinteraksi dengan *honeypot* tersebut. Di saat yang sama, *honeypot* akan mengambil data penyerang seperti aktivitas, teknik, serta alat yang digunakan. Namun, instalasi *honeypot* serta manajemen yang masih dilakukan secara manual adalah tindakan yang kurang efisien dan memakan cukup banyak waktu. Terlebih, sebagian besar organisasi menerapkan infrastruktur yang kompleks.

Banyaknya mesin dan sistem untuk menjalankan *honeypot* ditambah konfigurasi yang beragam serta pengelolaan *honeypot* secara individu di setiap mesinnya menjadi tugas yang cukup merepotkan.

Berbagai permasalahan yang muncul ini kemudian disimpulkan menjadi beberapa poin, antara lain konfigurasi yang mulai kompleks, konsumsi waktu dan daya yang besar, konsistensi yang sulit dipertahankan, skalabilitas terbatas, dan kurang fleksibel dalam merespons perubahan. Dari banyaknya poin yang disampaikan, proses *honeypot deployment* yang masih dilakukan secara manual secara tidak langsung dapat memberikan efek yang cukup merugikan bagi pihak organisasi dari segi waktu hingga finansial. Maka dari itu, diperlukan setidaknya sistem atau alat yang mampu untuk mengeliminasi beberapa permasalahan akibat dari proses *honeypot deployment* yang masih manual.

Ansible merupakan salah satu alat pengelolaan konfigurasi yang populer dan efisien dalam menjalankan tugas-tugas manajemen sistem secara otomatis. Menurut Lovdianchel, *Ansible* adalah *tools open source* yang menerapkan konsep *infrastructure as a code* untuk melakukan berbagai tugas seperti *provisioning server*, *configuration management*, dan *deployment* aplikasi secara terotomasi [6]. *Ansible* menggunakan bahasa yang mudah untuk dipahami manusia, sehingga admin sistem dapat membuat konfigurasi sesuai dengan kebutuhan. Setelah itu, *Ansible* akan menjalankan konfigurasi tersebut di seluruh infrastruktur secara konsisten. Dengan memanfaatkan *Ansible*, admin sistem akan lebih mudah dalam memasang *honeypot* di berbagai mesin dan konfigurasi yang telah didefinisikan seragam dan konsisten.

Tujuan dari penelitian ini adalah untuk mengisi ketimpangan pengetahuan mengenai *honeypot deployment* secara efektif dan efisien menggunakan *Ansible*. Selain itu, dengan penambahan fitur *dashboard* dapat memudahkan admin dalam mengelola dan juga memantau proses *honeypot deployment*. Dengan penelitian ini, diharapkan hasil yang didapat mampu berperan penting dalam pengembangan praktik terbaik dan menjadi pedoman bagi para praktisi keamanan untuk mengotomasi *honeypot deployment* dan meningkatkan pertahanan sistem dari serangan siber.

II. DASAR TEORI

A. Penelitian Sebelumnya

Penelitian dengan judul *"Securing a Small Network by using Raspberry Pi Honeypot"* membahas tentang penggunaan *honeypot* dengan memanfaatkan perangkat *Raspberry Pi* untuk memantau serta mengamankan jaringan. Dari penelitian tersebut menghasilkan kesimpulan bahwa penggunaan *honeypot* yang dikombinasikan dengan *Raspberry Pi* merupakan cara efisien serta mampu menekan biaya untuk membangun sistem pemantauan dan pengamanan jaringan rumah atau bisnis level bawah [7]. Selain itu, penelitian sejenis dengan judul *"Dynamic Honeypot Deployment in the Cloud"* membahas tentang konsep baru *honeypot deployment* di lingkungan *cloud* untuk mengatasi keterbatasan dari sistem yang ada saat ini. Dari penelitian tersebut menghasilkan kesimpulan bahwa sistem yang diusulkan mencapai otomasi dalam menyediakan serta menjalankan *honeypot* dengan menggunakan salah satu *service* pada *cloud AWS* yaitu *AWS Lambda* [8].

Penelitian dengan judul *"Infrastructure as Code for Security Automation and Network Infrastructure Monitoring"* membahas tentang pemanfaatan konsep *infrastructure as code* untuk mempermudah dan membuat pekerjaan pada pengelolaan infrastruktur jaringan menjadi lebih cepat dan efisien. Selain itu, penelitian ini mengusulkan penggunaan *Ansible* sebagai *tools* otomasi untuk melakukan berbagai pekerjaan seperti membuat *virtual machine*, *intrusion detection system*, *honeypot*, dan pengelolaan *security information* dan *event* agar dapat terhubung pada infrastruktur jaringan secara efektif [9]. Selain itu, terdapat penelitian sejenis dengan judul *"Implementation of Devops Method for Automation of Server Management Using Ansible"* membahas tentang pemanfaatan metode *DevOps* yaitu suatu konsep pengembangan dan penyampaian perangkat lunak ke infrastruktur dengan mengambil pendekatan secara kolaboratif dan integratif antara *developer* dan *software operation*. Selain itu, penelitian ini juga mengusulkan penggunaan *Ansible* dengan metode *DevOps* untuk melakukan otomasi pengelolaan banyak *server* [10].

Penelitian dengan judul *"Creation of a High-Interaction Honeypot System based-on Docker Containers"* membahas tentang pembuatan sistem *high-interaction honeypot* dengan memanfaatkan *Docker* sebagai wadah untuk menampung *honeypot* serta mendeteksi serangan pada *network-level* dan *host-level*. Hasil dari penelitian ini adalah sistem *honeypot* berbasis *Docker* yang lebih sulit untuk dideteksi dan diterapkan pada sistem *Linux* [11]. Selain itu, terdapat penelitian sejenis dengan judul *"Containerized cloud-based honeypot deception for tracking attackers"* membahas tentang pengenalan konsep *honeypot* serta *honeypot deployment* secara *container-based* di *cloud* untuk mencapai sistem yang portabel, kuat, dan mudah dalam instalasi dan pengelolaannya. Penelitian ini memberikan kesimpulan bahwa sistem yang dibuat secara efektif mampu mengumpulkan lebih banyak informasi serangan, namun masih terbatas pada tipe ancaman yang bisa dideteksi [12].

Penelitian dengan judul "Implementasi *Flask Framework* pada Pembangunan Aplikasi Sistem Informasi *Helpdesk (SIH)"* membahas tentang pengembangan aplikasi berbasis web dengan memanfaatkan *framework Flask* untuk pembuatan *API*, *Bootstrap* untuk *framework* pembuatan *UI*, dan *PostgreSQL* untuk penyimpanan *database*. Penelitian ini memberikan kesimpulan bahwa implementasi *framework Flask* pada pengembangan aplikasi berbasis web dapat meningkatkan efisiensi dan efektivitas pada proses bisnis [13]. Selain itu, terdapat penelitian sejenis dengan judul "Implementasi *Flask Framework* pada Pembangunan Aplikasi *Purchasing Approval Request"* membahas tentang pengembangan aplikasi dengan memanfaatkan *framework Flask*. Dari penelitian tersebut menghasilkan kesimpulan

bahwa penggunaan *framework Flask* pada aplikasi mampu meminimalisasi *error* pada aplikasi, menyederhanakan proses pembuatan aplikasi, dan mempercepat komunikasi antara *backend* dan *frontend* aplikasi [14].

Penelitian dengan judul *"*Penerapan *React JS* pada Pengembangan *FrontEnd* Aplikasi *Startup Ubaform"* membahas tentang pengembangan aplikasi berbasis dengan memanfaatkan *React JS* untuk pengembangan tampilan *UI* atau *frontend* pada web. Dari penelitian tersebut menghasilkan kesimpulan bahwa pengembangan *frontend* dengan memanfaatkan *React JS* mampu menghemat waktu serta mempermudah pengelolaan karena setiap bagian pada tampilan web dikemas menjadi *component,* sehingga mempermudah dalam proses *debugging* [15].

Berdasarkan penelitian sebelumnya, otomasi *honeypot deployment* sudah pernah dilakukan dengan alat dan metode yang beragam. Namun, penelitian tentang pemanfaatan *Ansible* yang difokuskan untuk *honeypot deployment* masih terbatas. Selain itu, implementasi *Ansible* masih berbasis *Command Line Interface (CLI)*, belum menerapkan penggunaan berbasis *API* atau web seperti salah satunya menggunakan *AWX Ansible*. Penelitian ini dimaksudkan untuk mengimplementasikan *Ansible* untuk melakukan otomasi *honeypot deployment* yang diintegrasikan dengan aplikasi web. Selain itu, beberapa pengujian juga dilakukan untuk memastikan aplikasi web berfungsi dengan baik serta mengetahui *honeypot* berhasil menerima *traffic* dari *port* yang diekspos.

B. *Honeypot*

Honeypot merupakan sistem yang terhubung ke jaringan berfungsi sebagai umpan untuk memikat penyerang dunia maya dan membantu mereka mendeteksi, mengalihkan, dan mempelajari upaya peretasan untuk mendapatkan akses tidak sah ke sistem informasi. Menurut Titarmare dkk., menyatakan bahwa *honeypot* adalah sistem tiruan yang dirancang untuk mengalihkan penyerang dari sistem yang dilindungi dan mendapatkan informasi tentang aktivitas mencurigakan dari mereka, serta dilengkapi dengan kemampuan untuk memantau dan menyimpan semua *logs* aktivitas yang terjadi pada *honeypot* [16]. *Honeypot* menampilkan dirinya sebagai target potensial penyerang di internet, biasanya *server* atau data berharga lainnya, dan mengumpulkan informasi dan memberi tahu korban tentang segala upaya untuk mendapatkan akses tidak sah ke sistem informasi. Sistem *honeypot* sering bekerja dengan sistem operasi yang diperkuat, yang memiliki langkah-langkah keamanan tambahan untuk meminimalkan potensi bahaya. Biasanya, mereka dikonfigurasi dengan kerentanan yang disengaja sehingga dapat dimanfaatkan oleh penyerang. Misalnya, seperti membuka dan mengekspos *port* pada protokol yang sering diincar oleh penyerang contohnya *SMB*, *FTP*, *SSH*, *RDP*, dsb.

C. *DevOps*

DevOps adalah praktik manajemen infrastruktur dan pengembangan perangkat lunak yang bertujuan untuk mengintegrasikan tim pengembangan dan tim operasi dalam siklus pengembangan perangkat lunak. Tujuan utama *DevOps* adalah untuk meningkatkan kualitas perangkat lunak, mengurangi risiko dan biaya dalam pengembangan dan pengoperasian sistem, dan mempercepat pengiriman perangkat lunak ke produksi.

DevOps mengintegrasikan budaya, praktik, dan sumber daya untuk mencapai tujuan ini. Beberapa konsep dan praktik utama *DevOps* termasuk:

1. Otomasi: Mengurangi kesalahan manusia dan mempercepat rilis perangkat lunak dengan mengotomasikan proses pengujian, integrasi, pengiriman, dan pengoperasian.
2. Kolaborasi: Mendorong tim operasi, pengembangan, dan tim lain yang terlibat dalam siklus pengembangan, seperti *QA (Quality Assurance)*, untuk bekerja sama lebih erat.
3. Pengelolaan Konfigurasi: Menggunakan alat otomatis untuk mengelola konfigurasi sistem dan aplikasi untuk memastikan konsistensi dan skalabilitas.
4. Orkestrasi: Menggunakan alat orkestrasi untuk mengelola dan mengotomasikan proses yang kompleks seperti penyebaran perangkat lunak.
5. Kontainerisasi: Menggunakan teknologi seperti *Docker* untuk menggabungkan aplikasi dan komponennya ke dalam wadah yang dapat digunakan di berbagai tempat.
6. *Infrastructure as Code* (IaC): Mendefinisikan infrastruktur dengan kode yang memungkinkan untuk membuat, mengelola, dan merespons perubahan infrastruktur dengan cara yang seragam dan otomatis.

Beberapa alat dan teknologi yang digunakan oleh *DevOps* adalah sebagai berikut:

1. *Docker* adalah *platform* kontainer yang memungkinkan aplikasi berjalan secara konsisten.
2. *Kubernetes* adalah *orchestrator* kontainer yang digunakan untuk mengelola dan merencanakan aplikasi berbasis kontainer dalam proses produksi.
3. *Ansible* adalah alat untuk otomatisasi konfigurasi dan manajemen yang membantu menjalankan infrastruktur dengan kode.
4. *Git* adalah sistem kontrol versi yang bertanggung jawab atas kode sumber aplikasi.

D. *Ansible*

Ansible adalah alat *deployment* aplikasi, manajemen konfigurasi, dan penyediaan perangkat lunak berbasis *open source*. *Ansible* berjalan pada banyak sistem *Unix* dan memiliki kemampuan untuk mengkonfigurasi sistem *Unix* dan *Microsoft Windows*. Untuk mengonfigurasi *server* jarak jauh, *Ansible* menggunakan protokol *SSH* dan mengirimkan *command* melalui koneksi *SSH* [17]. Berdasarkan penelitian yang dilakukan oleh Elradi, menyatakan bahwa pekerjaan yang diotomasi menggunakan *Ansible* mampu mempercepat penyelesaian tugas hingga 70% lebih cepat jika dibandingkan tanpa menggunakan otomasi [18]. Selain itu, *Ansible* menggunakan bahasa yang mudah untuk dibaca manusia, yaitu *YAML* untuk mendefinisikan tugas yang akan dieksekusi.

Menurut salah satu artikel oleh Maha, *Ansible* memiliki beberapa komponen di antaranya [19]:

1. *Inventory*, berisi alamat *IP* dari setiap perangkat yang dikelola.
2. *Modules*, sebuah *package* yang berisi unit kode untuk dieksekusi serta memiliki fungsi yang spesifik.
3. *Ad-hoc*, baris perintah untuk menjalankan *task* melalui *Command Line Interface* atau *CLI*.
4. *Playbooks*, sebuah *file* dengan format *YAML* yang berisi serangkaian *tasks* yang dieksekusi secara urut.

Selain itu, terdapat komponen lain seperti *roles* yaitu suatu metode untuk memecah kompleksitas *playbook* menjadi lebih kecil yang direpresentasikan ke dalam bentuk komponen modular yang di dalamnya memuat gabungan dari beberapa konten dengan fungsi spesifik.

E. *Ansible AWX*

Menurut Freeman dan Keating, *Ansible AWX* merupakan proyek *upstream open source* dari versi komersial *Ansible Tower* yang dikembangkan oleh *Red Hat* dan memiliki fitur yang sama dengan *Ansible Tower*, namun tanpa *support* atau versi *stable* dari *Red Hat* [20]. *Ansible AWX* mengadopsi teknologi *Ansible* dengan fitur tambahan berbasis web dan *API* untuk mengelola *playbook, inventory, credential*, dan *resources* lainnya. *Ansible AWX* menyediakan *API* yang bisa dikombinasikan dengan aplikasi lain, sehingga berbagai macam aplikasi bisa dikembangkan dengan memanfaatkan alat ini.

F. *Flask*

Flask merupakan *framework Python* yang dimanfaatkan dalam pembuatan web. *Flask* seringkali disebut sebagai *micro framework* karena tidak membutuhkan alat atau *library* tertentu. Menurut Steffi, *micro framework* merupakan versi minimal dari sebuah *framework* yang di dalamnya berisi kode dengan fungsi tertentu dan dibangun pada *platform* tertentu juga [21]. Sehingga jika ingin menambahkan fungsi-fungsi lainnya seperti integrasi ke *database*, validasi *form*, dan sebagainya membutuhkan modul pihak ketiga. Namun, sekarang sudah banyak modul yang memang dikembangkan untuk menunjang fungsionalitas *Flask* seolah-olah modul tersebut diimplementasikan oleh *Flask* sendiri.

G. *React*

Menurut Boduch dan Derks, *React* atau *React JS* merupakan *library* yang dimanfaatkan dalam pembuatan tampilan pengguna atau *user interface* secara interaktif pada sebuah aplikasi [22]. Dengan menggunakan *React*, pengembang dapat mengelompokkan *UI* menjadi beberapa potongan kode yang disebut sebagai *component*. *Component* ini bersifat independen dan dapat digunakan kembali sesuai kebutuhan. *React* ini bukanlah *framework* karena fungsi utamanya hanya untuk memuat *component* pada tampilan aplikasi.

H. *Docker*

Docker merupakan *platform* layanan yang menggunakan teknologi virtualisasi pada level *OS* untuk menghasilkan perangkat lunak yang dikemas dalam paket dengan *environment* terisolasi yang disebut sebagai *container* [23]. *Container* ini bersifat isolatif antara satu dengan yang lain serta menggabungkan perangkat lunak mereka sendiri, *library*, beserta berkas konfigurasi. *Container* juga mampu berkomunikasi satu sama lain dan bahkan bisa juga ke luar mesin *host* dengan konfigurasi tertentu.

I. *Kubernetes*

Kubernetes atau biasa disebut *K8s* merupakan *platform* orkestrasi *open source* yang ditujukan untuk otomasi *deployment*, pengelolaan beban kerja, penyediaan konfigurasi, dan pengelolaan pada aplikasi *container* [24]. Pada level produksi, *Kubernetes* sangat dibutuhkan untuk memastikan *container* tetap berjalan dan meminimalisasi waktu *downtime*. Selain itu, *Kubernetes* juga menyediakan fitur-fitur lain seperti *load balancing*, pembaruan dan *scaling* aplikasi, dan otomasi *rollback* dan *rollout container* [25].

J. *Google Cloud Platform*

Google Cloud Platform atau biasa disingkat *GCP* merupakan salah satu penyedia terkemuka layanan *cloud computing* yang ditawarkan oleh *Google*. *Cloud computing*, penyimpanan, *cloud database*, analitik, pembelajaran mesin, kecerdasan buatan, *Internet of Things (IoT)*, dan lainnya adalah beberapa layanan infrastruktur *cloud* yang disediakan oleh *Google Cloud Platform*.

Beberapa layanan yang tersedia di *GCP* di antaranya sebagai berikut:

1. *Google Compute Engine* adalah layanan *cloud computing* yang memungkinkan *user* membuat dan mengelola mesin virtual di infrastruktur *Google*, yang memungkinkan *user* menjalankan berbagai jenis tugas.
2. *Google Cloud Storage* adalah layanan penyimpanan *cloud* yang memungkinkan *user* menyimpan dan mengelola data secara skalabilitas dengan berbagai jenis penyimpanan,

seperti penyimpanan objek dan penyimpanan berbasis blok, dan lainnya.

3. *Google BigQuery* adalah layanan analitik data yang memungkinkan analisis data secara cepat dan skalabel.
4. *Google Cloud SQL* adalah layanan sistem *database* yang mendukung *database SQL Server*, *MySQL*, dan *PostgreSQL*.

III. METODOLOGI

A. Bahan

Beberapa perangkat lunak dan *framework* yang dibutuhkan dalam penelitian ini adalah sebagai berikut:

1. Sistem Operasi *Windows 10*, yang akan digunakan pada perangkat admin.
2. Sistem Operasi *Ubuntu Server 20.04 LTS*, yang akan digunakan pada *VPS*.
3. *Ansible*, sebagai alat untuk otomasi *honeypot deployment*.
4. *Ansible AWX*, sebagai alat untuk menyediakan *API* dari *Ansible*.
5. *Minikube*, sebagai alat untuk menjalankan *Kubernetes single-node*.
6. *Flask*, sebagai *framework* untuk pembuatan *API*.
7. *React JS*, sebagai *library* untuk pembuatan *UI dashboard*.
8. Beberapa *honeypot* yang sudah di-*build* menggunakan *Docker* (*Dionaea*, *Cowrie*, *Gridpot*, *Honeytrap*, *Elasticpot*, *RDPY*).
9. *Python*, sebagai bahasa pemrograman untuk pembuatan *API*.
10. *PostgreSQL*, sebagai sistem penyimpanan *database*.
11. *Google Cloud Platform (GCP)*, sebagai *cloud environment* untuk pengujian *honeypot deployment*.
12. *Google Compute Engine*, sebagai sensor berwujud *VM* yang berjalan pada *cloud GCP*.

B. Peralatan

Beberapa peralatan yang digunakan pada penelitian ini di antaranya sebagai berikut:

1. Laptop

   Laptop yang digunakan dalam penelitian ini memiliki spesifikasi disajikan pada Tabel 1.

Tabel 1. Spesifikasi laptop

Sistem Operasi	Memori	CPU	*Storage*
Windows 10	8 GB	AMD Ryzen 5 4500U 2,38 GHz	500 GB

2. *Virtual Machine GCE*

   Virtual Machine GCE yang digunakan dalam penelitian ini memiliki spesifikasi disajikan pada Tabel 2.

Tabel 2. Spesifikasi *GCE*

Sistem Operasi	Memori	CPU	Storage
Ubuntu 20.04 LTS	8 GB	single core 2 vCPU 2,20 GHz	20 GB

3. *VPS*

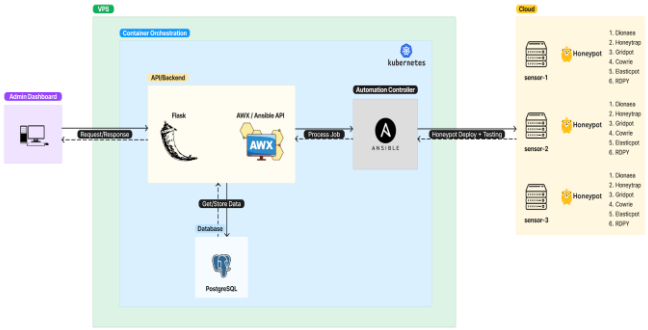
   VPS yang digunakan dalam penelitian ini memiliki spesifikasi disajikan pada Tabel 3.

Tabel 3. Spesifikasi *VPS*

Sistem Operasi	Memori	CPU	*Storage*
Windows 10	16 GB	Dual core 3,50 GHz	1,5 TB

C. Arsitektur Sistem *Honeypot Deployment*

Sistem terbagi menjadi beberapa komponen di antaranya adalah web *dashboard*, *API backend* dan *Ansible*, dan sensor di *cloud environment*. Proses *honeypot deployment* diawali dengan admin menambahkan sensor yang ingin diinstal melalui *form* di web *dashboard*. Pada *form* terdapat beberapa parameter yang wajib diisi seperti *IP address* sensor, nama sensor, dan *honeypot* yang ingin diinstal. Setelah *submit form*, data akan dikirimkan ke *backend* atau *Flask API* untuk diproses. Beberapa data akan disimpan ke *database* untuk kebutuhan web *dashboard* dan sisanya akan diteruskan ke *API Ansible* yaitu *AWX*. *AWX* memproses data tersebut untuk mempersiapkan *job template honeypot deployment*. Setelah *job template* selesai disiapkan, *Ansible* mulai mengeksekusi proses *honeypot deployment* ke sensor yang berjalan di *cloud GCP* sekaligus menjalankan *testing* secara terotomasi untuk memastikan *honeypot* telah berhasil terpasang. Arsitektur sistem *honeypot deployment* secara lengkap disajikan pada Gambar 1.



Gambar 1. Arsitektur sistem *honeypot deployment*

D. Topologi Jaringan Sistem

Topologi jaringan pada penelitian ini terbagi menjadi 2 *network*, yaitu *home network* dan *VPC network*. *Home network* merupakan jaringan *private* yang di dalamnya terdapat perangkat *PC* admin, *VPS*, dan *router*. *PC* admin dan *VPS* terhubung ke *router* untuk mendapatkan *private IP* agar mampu berkomunikasi di jaringan lokal. Selain itu, *router* juga mendapatkan *NAT IP* dari *ISP* agar perangkat yang terhubung ke *router* memiliki akses ke internet. *VPC network* merupakan jaringan *cloud* yang di dalamnya terdapat perangkat *VM GCE*, *VPC Router*, dan *VPC firewall*. *GCE* tersebar di beberapa *region* sehingga masing-masing terhubung pada *network address* yang berbeda. Selain itu, fungsinya juga dibagi menjadi sensor dan *main server*. Sensor merupakan *VM* yang memiliki fungsi khusus sebagai *honeypot server*. *Server* ini menjadi target utama untuk proses *honeypot deployment*. Semua *port* yang terdapat pada sensor diekspos untuk menerima semua *traffic*. Namun, terdapat *port* khusus yaitu *port* 22888 untuk koneksi *SSH* dari *PC* admin dan tidak bisa diakses dari *IP* lain. Sementara itu, *main server* berfungsi sebagai *server* yang dilindungi dari sebagian besar *traffic*. *Server* tersebut hanya bisa diakses oleh *traffic* yang berasal dari *PC* admin. Untuk mendapatkan hasil tersebut diperlukan *VPC firewall* yang mengatur setiap *traffic* yang masuk ke *VPC network*. Spesifikasi pembagian *IP address* dan *rules* pada *firewall* disajikan pada Tabel 4 dan Tabel 5.

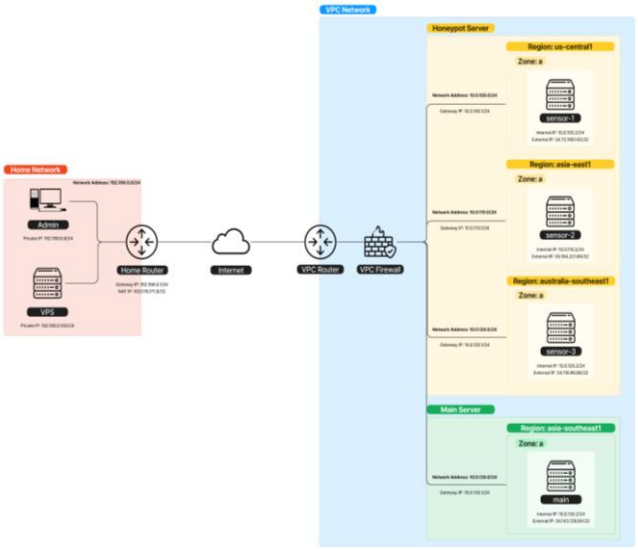
Tabel 4. Pembagian *IP address GCE*

No	Perangkat	*IP Address*	*Subnet*	Jenis *IP*
1	Admin *PC*	192.168.0.9	/24	*Private*
2	*VPS*	192.168.0.100	/24	*Private*
3	*Home Router*	192.168.0.1	/24	*Gateway*
		103.178.171.9	/32	*NAT*
4	*VPC Router*	10.0.100.1	/24	*Gateway*
		10.0.110.1	/24	*Gateway*
		10.0.120.1	/24	*Gateway*
		10.0.130.1	/24	*Gateway*
5	sensor-1	10.0.100.2	/24	*Internal*
		34.72.199.142	/32	*External*
6	sensor-2	10.0.110.2	/24	*Internal*
		35.194.221.89	/32	*External*
7	sensor-3	10.0.120.2	/24	*Internal*
		34.116.96.86	/32	*External*
8	main	10.0.130.2	/24	*Internal*
		34.143.139.84	/32	*External*

Tabel 5. Daftar *rules firewall*

No	Nama	*Port*	*Source IP*	Target	*Priority*	*Action*
1	*hp-allow-custom-ssh-admin*	tcp:22888	103.178.171.9/32	*hp-server*	1	*allow*
2	*hp-deny-custom-ssh*	tcp:22888	0.0.0.0/0	*hp-server*	2	*deny*
3	*hp-allow-all-inbound*	all	0.0.0.0/0	*hp-server*	1000	*allow*
4	*main-allow-all-inbound-admin*	all	103.178.171.9/32	*main-server*	1	*allow*
5	*main-deny-all-inbound*	all	0.0.0.0/0	*main-server*	1000	*deny*

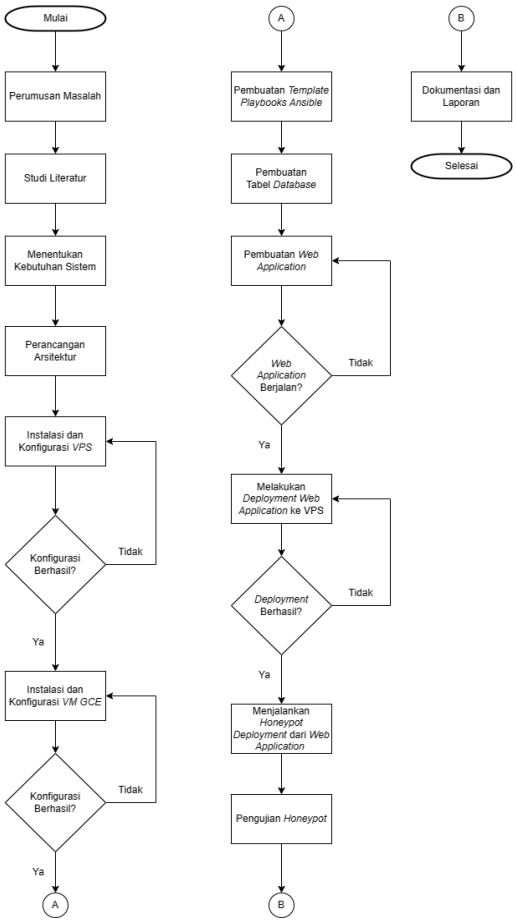
Topologi jaringan sistem pada *home network* dan *VPC network* secara keseluruhan disajikan pada Gambar 2.



Gambar 2. Topologi jaringan sistem

E. Diagram Alir Penelitian

Diagram alir dari penelitian yang dilakukan disajikan pada Gambar 3.



Gambar 3. Diagram alir penelitian

F. Instalasi dan Konfigurasi *VPS*

Secara umum langkah-langkah instalasi dan konfigurasi *VPS* adalah sebagai berikut.

1. Konfigurasi *User*

Konfigurasi *user* dilakukan dengan cara membuat *user* baru bernama *ansiadmin*. *User* ini akan menjadi *user* utama untuk mengeksekusi berbagai macam perintah di *server*. Selanjutnya menambahkan *user* tersebut ke grup *root* dan menambahkan akses *sudo*.

2. Konfigurasi *SSH*

Konfigurasi *SSH* dilakukan dengan menjalankan *generate SSH keygen* pada *user "ansiadmin"*. Penggunaan sistem *key* ini bertujuan agar koneksi *SSH* dari *server* ke sensor dilakukan secara *password-less* atau tanpa *password*, sehingga proses *honeypot deployment* oleh *Ansible* dapat berfungsi dengan baik.

3. Instalasi *Package*

Instalasi *package* dilakukan dengan menginstal beberapa *package* seperti *net-tools, lynx, acl, python3-pip,* dan *ansible*.

4. Instalasi dan Konfigurasi *Docker*

Melakukan instalasi *Docker* dengan nama *package docker.io* dan menambahkan *user "ansiadmin"* ke dalam grup *Docker* agar bisa memiliki akses untuk mengeksekusi perintah *Docker*.

5. Instalasi *Minikube*

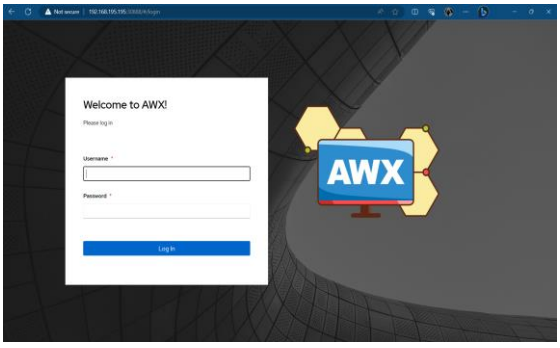
Melakukan instalasi *Minikube* sebagai *single cluster Kubernetes* dan instalasi *kubectl* sebagai *command-line tool* untuk pengelolaan *cluster Kubernetes*.

6. Instalasi *PostgreSQL*

Melakukan instalasi *PostgreSQL* di dalam *Minikube* atau *cluster Kubernetes*. *PostgreSQL* berperan sebagai sistem *database* utama pada penelitian ini.

7. Instalasi *AWX Ansible*.

Melakukan instalasi *AWX* di dalam *Minikube* dan menjalankan *port forwarding* agar memiliki akses ke *dashboard* web *AWX*. Tampilan awal instalasi *AWX Ansible* disajikan pada Gambar 4.



Gambar 4. Halaman *login AWX*

G. Konfigurasi *AWX Ansible*

Secara umum konfigurasi yang dilakukan pada *AWX Ansible* adalah sebagai berikut.

1. Menambahkan *Organizations*

Pada *AWX Ansible*, *Organizations* merupakan reprensentasi dari pengelompokan *resources*, seperti *host, inventory, project*, dan *credential* di dalam *environment AWX Ansible* dalam rangka pengelolaan *resources* dan akses kontrol.

2. Menambahkan *Execution Environment*

Pada *AWX Ansible*, *Execution Environment* merupakan *runtime environment* yang telah ditentukan sebelumnya untuk mengeksekusi *playbooks* dan *jobs* pada *Ansible*. *Execution Environment* dibangun pada kontainer *Docker* yang di dalamnya sudah terinstal *Ansible*, *Python*, dan *dependency* lainnya sehingga mampu mengontrol serta mengelola jalannya *playbooks Ansible*.

3. Menambahkan *Projects*

Pada *AWX Ansible*, *Projects* merupakan komponen mendasar yang digunakan untuk mengatur dan mengelola *playbooks*, *templates*, dan *resources* lainnya serta mendapatkan berkas tersebut melalui *version control system* seperti *Git*.

4. Menambahkan *Inventories*

Pada *AWX Ansible*, *Inventories* merupakan kumpulan *hosts, groups*, dan *variables* yang menjadi target bagi *Ansible* untuk mengeksekusi *playbooks* dan *jobs*.

5. Menambahkan *Credential Types*

Pada *AWX Ansible*, *Credential Types* merupakan konfigurasi tambahan berupa tipe kredensial yang telah didefinisikan terlebih dahulu sebelumnya untuk mengakses berbagai sistem, layanan, dan *resources* saat *playbooks* dan *jobs Ansible* dieksekusi. *Credential Types* berisi informasi dan protokol yang digunakan saat membangun koneksi ke sistem target. *"Sensor Credential"* merupakan jenis kredensial yang digunakan untuk autentikasi ke sensor. *"Postgres Credential"* merupakan jenis kredensial yang digunakan untuk autentikasi ke *database*.

6. Menambahkan *Credentials*

Pada *AWX Ansible*, *Credentials* merupakan informasi yang dibutuhkan untuk mengakses dan autentikasi pada berbagai sistem, layanan, dan *resources* saat mengeksekusi *playbooks* dan *jobs Ansible*. *Credentials* memanfaatkan *Credential Types* sebagai *template* untuk mendefinisikan informasi seperti *username, password, SSH Keys*, dan sebagainya yang dibutuhkan ketika membangun koneksi ke sistem.

Kredensial *"ansible"* merupakan kredensial yang digunakan oleh *server* untuk mendapatkan akses *SSH* ke sensor. Kredensial *"sensor_init"* merupakan kredensial *user* yang digunakan oleh *server* untuk melakukan konfigurasi awal pada sensor sebelum *user "ansigent"* dibuat. Kredensial *"ansigent"* merupakan kredensial *user* yang digunakan oleh *server* untuk melakukan *deployment package* dan *honeypot* pada sensor. Kredensial *"postgres"* merupakan kredensial yang digunakan oleh *server* untuk mengakses *database* penelitian.

7. Menambahkan *Applications*

Pada *AWX Ansible*, *Applications* merupakan fitur yang berfungsi untuk memudahkan integrasi *API Ansible* dengan aplikasi lain. *Applications* menyediakan *token* yang bisa digunakan oleh aplikasi lain agar bisa memiliki akses ke *resources Ansible*. Setelah *Applications* ditambahkan *token* tidak langsung didapatkan karena *token* tersebut harus diasosiasikan dengan *user* terlebih dahulu.

8. Menambahkan *Templates*

Pada *AWX Ansible*, *Templates* merupakan serangkaian konfigurasi dan parameter yang digunakan untuk mendefinisikan serta mengeksekusi *playbooks* atau *jobs Ansible*. *Templates* menyajikan konfigurasi yang dapat digunakan berulang kali dengan aturan dan masukan yang sudah ditentukan, yang disebut sebagai *Job Templates*. *Workflow Job Templates* merupakan serangkaian *Job Templates* yang disusun dan dieksekusi secara berurutan sesuai dengan aturan yang telah didefinisikan.

a. *Job template: 1. Initial Configuration Honeypot*

Template ini mendefinisikan konfigurasi awal pada sensor sebelum melakukan *honeypot deployment*. Di dalamnya terdapat *playbook* dengan berbagai macam *tasks* seperti melakukan membuat *user "ansigent"*, menambahkan akses *sudo* ke *user "ansigent"*, menambahkan *SSH public key* ke *user "ansigent"*, dan memperbarui *port SSH* menjadi 22888.

b. *Job Template: 2. Package Installation Honeypot*

Template ini mendefinisikan instalasi *package* yang akan diinstal pada sensor. Di dalamnya terdapat *playbook* dengan berbagai macam *tasks* seperti instalasi *base-tools* seperti *curl, git, nano,* dsb., instalasi *QEMU* yaitu *emulator* untuk menjalankan *image Docker* di berbagai arsitektur *CPU*, instalasi *Docker*, dan instalasi modul *Python PostgreSQL*.

c. *Job Template*: *3. Deploy Selected Honeypot*

Template ini mendefinisikan instalasi jenis *honeypot* yang akan diinstal pada sensor. Di dalamnya terdapat *playbook* dengan berbagai macam *tasks* seperti menjalankan *task honeypot deployment*, mengecek status *deployment* secara berkala, menghentikan *service honeypot* yang tidak dipilih.

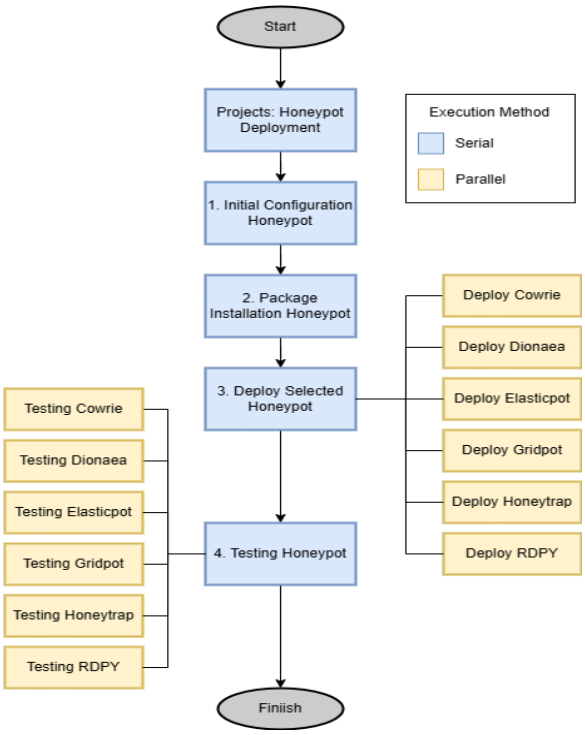
d. *Job Template*: *4. Testing Honeypot*

Template ini mendefinisikan pengecekan status *honeypot* pasca proses *deployment*. Di dalamnya

terdapat *playbook* dengan berbagai macam *tasks* seperti mengecek status *honeypot* berdasarkan *port* yang dibuka, menunggu proses pengecekan selesai, menampilkan *output* dari hasil pengecekan.

e. *Workflow Job Template*: *Initial Deployment Workflow*

Template ini mendefinisikan alur eksekusi *Job Templates* yang telah dibuat sebelumnya saat proses *honeypot deployment* pertama kali dijalankan. *Initial Deployment Workflow* disajikan pada Gambar 5.



Gambar 5. Visualisasi *Initial Deployment Workflow*

f. *Workflow Job Template*: *Initial Deployment Workflow*

Template ini mendefinisikan alur eksekusi *Job Templates* apabila terdapat perubahan pada *honeypot* yang sudah diinstal pada sensor.

H. Konfigurasi Sensor

Pada sensor hanya perlu melakukan konfigurasi *SSH*. Konfigurasi dilakukan dengan cara menyalin *public key SSH* yang telah dibuat di *VPS* ke sensor. *VPS* membutuhkan akses *SSH* ke sensor untuk bisa mengeksekusi *jobs honeypot deployment* pada sensor.

I. Pembuatan Tabel *Database*

Pembuatan tabel *database* pada penelitian ini tidak dilakukan secara manual, melainkan menggunakan fitur migrasi. Migrasi ini merupakan sebuah *version control system* untuk *database*. Beberapa *framework* web sudah mendukung fitur ini, salah satunya *Flask*. Fitur ini sangat membantu dalam pengelolaan tabel sehingga tabel selalu tersinkronisasi dengan struktur model data yang didefinisikan pada *source code* aplikasi web. Caranya cukup sederhana, yaitu diawali dengan membuat komponen model yang isinya terdapat nama tabel, nama kolom, tipe data, dan *constraint*. Lalu, komponen tersebut diinisialisasi ke dalam proyek *Flask*. Terakhir, menjalankan *command* migrasi dari *Flask*.

J. Pembuatan *API*

Hal pertama yang perlu dilakukan sebelum membuat *API* adalah menentukan spesifikasi dari *API* tersebut. Berikut spesifikasi *API* yang dibuat.

1. *Users*: *API* ini berfungsi untuk pengelolaan *users* yang terdaftar pada aplikasi web.
2. *Authentications*: *API* ini berfungsi untuk pengelolaan *token* milik *user* yang terdaftar pada aplikasi web. *Token* ini berfungsi agar *user* memiliki akses ke *resources* pada web.
3. *Sensors*: *API* ini berfungsi untuk pengelolaan sensor yang terdaftar pada aplikasi web.
4. *Honeypots*: *API* ini berfungsi untuk pengelolaan *honeypot* yang terdaftar pada aplikasi web.
5. *Jobs*: *API* ini terhubung langsung dengan *API AWX Ansible*. *API* ini berfungsi untuk mengeksekusi *Workflow Job Templates* yang terdaftar pada *AWX Ansible*.
6. *HoneypotSensor*: Setiap sensor memiliki *honeypot* sendiri. *API* ini berfungsi sebagai relasi antara sensor dengan *honeypot* yang terinstal di dalamnya. *Honeypot* yang terinstal di satu sensor tidak bergantung dengan *honeypot* di sensor lainnya. Sehingga *API* ini mengatasi ketergantungan tersebut.

K. Pembuatan *Dashboard*

Pada tahapan ini adalah melakukan pembuatan *dashboard* mengacu desain *mockup* yeng telah dibuat. Berikut uraian tampilan *mockup* halaman pada *dashboard*.

1. *Register*: Pada halaman *register* digunakan untuk mendaftarkan *user* baru ke dalam aplikasi.
2. *Login*: Pada halaman *login* digunakan untuk melakukan proses autentikasi *user*.
3. *Dashboard*: Halaman *dashboard* merupakan halaman utama yang digunakan untuk menampilkan beberapa informasi umum seperti jumlah sensor, jumlah *honeypot*, jumlah *deployment* yang sudah dijalankan, daftar riwayat *honeypot deployment* di semua sensor, dan navigasi ke beberapa halaman lain.
4. *Sensors*: Pada halaman *sensors* menampilkan daftar sensor yang terdaftar pada aplikasi.
5. *Add New Sensor*: Halaman *add new sensor* merupakan halaman untuk menambahkan sensor baru ke dalam aplikasi.
6. *Sensor Details*: Pada halaman *sensors details* menampilkan informasi sensor yang terdaftar secara detil.
7. *Sensor Logs*: Pada halaman *sensors logs* menampilkan informasi *logs* saat proses *honeypot deployment* dijalankan.
8. *Honeypots*: Pada halaman *honeypots* menampilkan informasi daftar *honeypot* yang tersedia pada aplikasi.

9. *Add New Honeypot*: Halaman *add new honeypot* merupakan halaman untuk menambahkan *honeypot* baru ke dalam aplikasi.
10. *Honeypot Details*: Pada halaman *honeypots details* menampilkan informasi *honeypot* yang terdaftar secara detil.
11. *Users*: Pada halaman *users* menampilkan informasi daftar *user* yang terdaftar pada aplikasi.
12. *Users Details*: Pada halaman *users details* menampilkan informasi *user* yang terdaftar secara detil.
13. *Profile*: Pada halaman *profile* menampilkan informasi *user* yang digunakan saat mengakses aplikasi web.

L. *Deployment* Aplikasi Web

Setelah *API* dan *dashboard* selesai dibuat, maka langkah berikutnya adalah melakukan *deployment* aplikasi ke *VPS*. Aplikasi tersebut dijalankan di dalam *Minikube*. Namun, sebelum dijalankan aplikasi tersebut harus melewati proses *building docker image* terlebih dahulu.

M. Metode Pengujian

Pada penelitian ini terdapat dua pengujian yaitu pengujian *dashboard* dan pengujian *honeypot*. Pengujian *dashboard* dilakukan dengan menguji berbagai menu dan fungsi yang terdapat pada *dashboard*. Pengujian ini bertujuan untuk mengetahui apakah aplikasi sudah berfungsi sesuai dengan yang dibutuhkan. Pengujian *honeypot* dilakukan dengan melakukan *scanning port* pada sensor menggunakan *tools netcat*. Setiap *honeypot* memiliki beragam *service* yang diekspos. Namun, untuk pengujian ini hanya diambil satu *service* dari setiap *honeypot*. *Service* atau *port honeypot* yang diuji disajikan pada Tabel 6 berikut.

Tabel 6. Daftar *service/port honeypot* yang diuji

No	*Honeypot*	*Port*
1	*Cowrie*	tcp:22
2	*Dionaea*	tcp:21
3	*Elasticpot*	tcp:9200
4	*Gridpot*	tcp:8000
5	*Honeytrap*	tcp:631
6	*RDPY*	tcp:3389

IV. HASIL DAN PEMBAHASAN

Hasil yang didapatkan dari penelitian ini adalah sebuah sistem atau aplikasi yang mampu melakukan *honeypot deployment* pada sensor secara otomatis dengan memanfaatkan *tools* seperti *Ansible* dan *dashboard* yang membantu *user* untuk mengeksekusi dan *monitoring* proses *deployment*. Untuk memastikan apakah aplikasi dapat berjalan dengan baik maka diperlukan pengujian. Ada dua jenis pengujian yang dilakukan pada aplikasi, yaitu pengujian *dashboard* dan pengujian *honeypot*.

A. Pengujian *Dashboard*

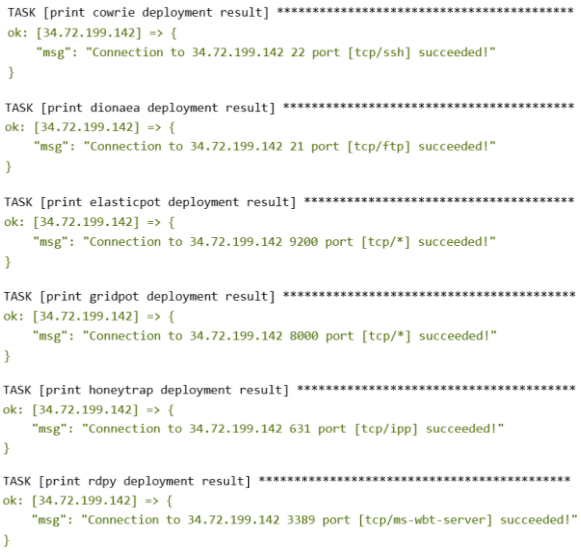
Pengujian *dashboard* berfungsi untuk mengetahui apakah aplikasi sudah berfungsi sesuai yang diharapkan. Pengujian ini dimulai dari penambahan *user* baru atau *register* hingga proses *honeypot deployment* dijalankan oleh *user*. Rincian hasil pengujian *dashboard* disajikan pada Tabel 7.

Tabel 7. Rincian hasil pengujian *dashboard*

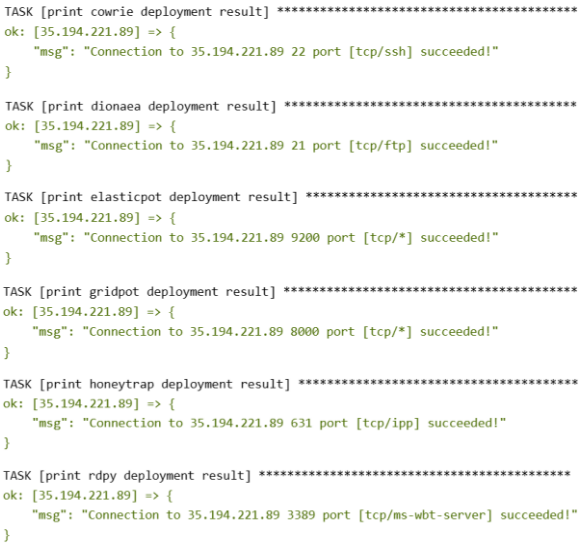
Komponen Pengujian	Skenario yang diuji	Hasil Pengujian	Kesimpulan
Fitur *Register*	Melakukan pendaftaran *user* dengan memasukkan *username*, *password*, dan nama lengkap.	*User* berhasil terdaftar dan memiliki akses ke dalam *dashboard*.	☒ Berhasil ☐ Tidak
Fitur *Login*	Melakukan *login* dengan menggunakan *username* dan *password*	*User* berhasil *login* serta mendapatkan *token*.	☒ Berhasil ☐ Tidak
Menu *Dashboard*	Menampilkan jumlah total sensor dan *honeypot* yang terdaftar pada aplikasi dan *history honeypot deployment* dari semua sensor.	Pada menu *Dashboard* sudah bisa menampilkan data sesuai dengan skenario.	☒ Berhasil ☐ Tidak
Menu *Sensors*	Menampilkan daftar sensor yang terdaftar pada aplikasi.	Pada menu *Sensors* sudah bisa menampilkan daftar sensor yang terdaftar pada aplikasi.	☒ Berhasil ☐ Tidak
Fitur *Add Sensor*	Mendaftarkan sensor baru dengan memasukkan nama sensor, *IP address*, *description*, dan *honeypot* yang ingin diinstal.	Sensor berhasil ditambahkan dan ditampilkan di menu *Sensors*.	☒ Berhasil ☐ Tidak
Menu *Sensor Details*	Menampilkan data informasi sensor secara detil.	Pada menu *Sensor Details* sudah bisa menampilkan informasi sensor secara detil.	☒ Berhasil ☐ Tidak
Menu *Sensor Logs*	Menampilkan informasi *logs* proses *honeypot deployment*.	Pada menu *Sensor Logs* sudah bisa menampilkan *logs* dari proses	☒ Berhasil ☐ Tidak

Komponen Pengujian	Skenario yang diuji	Hasil Pengujian	Kesimpulan
		honeypot deployment.	
Fitur *Relaunch Job*	Menjalankan ulang proses *honeypot deployment.*	Proses *honeypot deployment* berhasil dijalankan ulang menggunakan fitur *Relauch Job*.	☒ Berhasil ☐ Tidak
Fitur *Update Sensor*	Memperbarui data sensor.	Data sensor berhasil diperbarui dengan fitur *Update Sensor*.	☒ Berhasil ☐ Tidak
Fitur *Delete Sensor*	Menghapus data sensor dari aplikasi.	Data sensor berhasil dihapus dari aplikasi.	☒ Berhasil ☐ Tidak
Menu *Honeypots*	Menampilkan daftar *honeypot* yang terdaftar pada aplikasi.	Pada menu *Honeypots* sudah bisa menampilkan daftar *honeypot* yang terdaftar pada aplikasi. Menu ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Fitur *Add Honeypot*	Mendaftarkan *honeypot* baru dengan memasukkan nama *honeypot* dan *description*.	*Honeypot* berhasil ditambahkan dan ditampilkan di menu *Honeypots*. Fitur ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Menu *Honeypot Details*	Menampilkan data informasi *honeypot* secara detil.	Pada menu *Honeypot Details* sudah bisa menampilkan informasi *honeypot* secara detil. Menu ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Fitur *Update Honeypot*	Memperbarui data *honeypot.*	Data *honeypot* berhasil diperbarui dengan fitur *Update Honeypot*. Fitur ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Fitur *Delete Honeypot*	Menghapus data *honeypot* dari aplikasi.	Data *honeypot* berhasil dihapus dari aplikasi. Fitur ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Menu *Users*	Menampilkan daftar *user* yang terdaftar pada aplikasi.	Pada menu *Users* sudah bisa menampilkan daftar *user* yang terdaftar pada aplikasi. Menu ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Menu *User Details*	Menampilkan data informasi *user* secara detil.	Pada menu *User Details* sudah bisa menampilkan informasi *user* secara detil. Menu ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Fitur *Update User*	Memperbarui *roles* pada *user*.	*Roles user* berhasil diperbarui dengan fitur *Update User*. Menu ini juga hanya bisa diakses oleh *roles* admin.	☒ Berhasil ☐ Tidak
Menu *Profile*	Menampilkan data informasi pemilik akun.	Pada menu *Profile* sudah bisa menampilkan data informasi pemilik akun.	☒ Berhasil ☐ Tidak
Fitur *Update Profile*	Memperbarui data informasi pemilik akun.	Data informasi pemilik akun berhasil diperbarui menggunakan fitur *Update Profile*.	☒ Berhasil ☐ Tidak

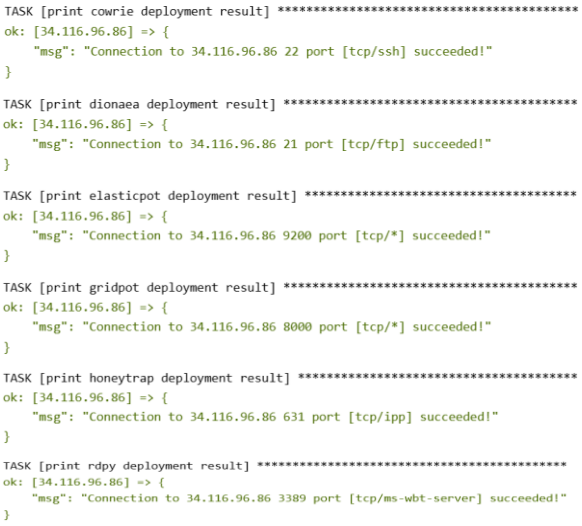
Pengujian *honeypot* berfungsi untuk mengetahui apakah *honeypot* berhasil terpasang dan *service*-nya berjalan di sensor. Pengujian ini dilakukan pada 3 sensor yang tersebar di beberapa *region*. Dari hasil pengujian didapatkan bahwa semua *honeypot* yang telah selesai diinstal pada setiap sensor dapat menerima *traffic* dari *port* atau *service* yang dijalankan. Pengujian ini dilakukan pada 3 sensor yang tersebar di beberapa *region*. Dari hasil pengujian didapatkan bahwa semua *honeypot* yang telah selesai diinstal pada setiap sensor dapat menerima *traffic* dari *port* atau *service* yang dijalankan. Hasil pengujian *honeypot* sensor-1 disajikan pada Gambar 6. Hasil pengujian *honeypot* sensor-2 disajikan pada Gambar 7. Hasil pengujian *honeypot* sensor-3 disajikan pada Gambar 8. Berikut ini hasil dari *honeypot deployment* di semua sensor.



Gambar 6. Hasil pengujian *honeypot* sensor-1

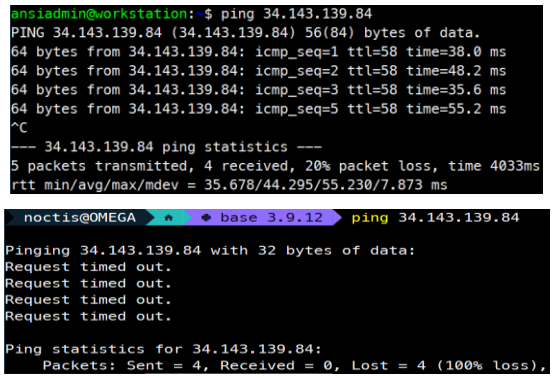


Gambar 7. Hasil pengujian *honeypot* sensor-2



Gambar 8. Hasil pengujian *honeypot* sensor-3

Pengujian untuk *main server* juga dilakukan untuk memastikan bahwa *server* hanya bisa diakses oleh *PC* admin. Pengujian dijalankan dengan melakukan *ping* ke *main server*. Pengujian pertama dilakukan dari *PC* admin, berikutnya dari *PC* lain. Pengujian pertama sudah dipastikan berhasil melakukan *ping*, sedangkan berikutnya gagal melakukan *ping* karena *traffic* diblok oleh *firewall* pada *VPC*. Hasil pengujian disajikan pada Gambar 9 berikut.



Gambar 9. Hasil pengujian *ping* ke *main server*

V. SIMPULAN

Berdasarkan penelitian yang sudah dilaksanakan dapat dianalisis dan menghasilkan beberapa kesimpulan bahwa, penelitian ini menghasilkan aplikasi web yang difungsikan khusus untuk *honeypot deployment* dengan memanfaatkan *tools Ansible*. *API Flask* dan *Ansible AWX* terintegrasi dengan baik sehingga proses *honeypot deployment* dapat dieksekusi melalui *dashboard* aplikasi web. Aplikasi sudah bisa menampilkan *logs* sehingga proses *honeypot deployment* bisa terpantau dengan baik.

Berdasarkan kesimpulan yang sudah dijelaskan, peneliti menambahkan beberapa saran yang dapat dilakukan untuk mengembangkan penelitian selanjutnya yaitu proses instalasi dan konfigurasi pada *VPS* dan sensor masih dilakukan secara manual, sehingga diperlukan otomasi. Otomasi juga diperlukan pada tahap pengkodingan sampai *deployment* aplikasi ke *VPS* dengan menerapkan konsep *DevOps*. Jumlah *honeypot* yang tersedia pada aplikasi masih terbatas dan sangat memungkinkan jumlahnya bertambah, sehingga dibutuhkan fitur untuk menambahkan *honeypot* baru pada aplikasi dan diotomasi prosesnya.

REFERENSI

[1] W. N. M. Sukma, A. Reynata, D. A. Yasmine and D. M. P. Pratama, "SYSTEMATIC LITERATURE REVIEW (SLR) : KEAMANAN DALAM SISTEM INFORMASI," *Journal of Comprehensive Science*, vol. II, no. 6, 2023.

[2] Kominfo, "Antisipasi Bersama Tingkatkan Sistem dan Cegah Serangan Siber," 22 September 2022. [Online]. Available: https://aptika.kominfo.go.id/2022/09/antisipasi-bersama-tingkatkan-sistem-dan-cegah-serangan-siber/.

[3] N. Naik, "IMPLEMENTATION OF TECHNIQUES TO AVOID CYBER ATTACKS," *The Online Journal of Distance Education and e-Learning*, vol. XI, no. 2, 2023.

[4] S. Parulian, D. A. Pratiwi and M. C. Yustina, "Ancaman dan Solusi Serangan Siber di Indonesia," *Telecommunications, Networks,*

Electronics, and Computer Technologies, vol. I, no. 2, pp. 85-92, 2021.

[5] A. S. Verma and A. Dubey, "A Review on Honeypot Deployment," *London Journal of Research in Computer Science and Technology*, vol. XX, no. 1, 2020.

[6] J. Lovdianchel, "Mengenal Ansible Automation," 2020. [Online]. Available: https://medium.com/dot-intern/configuration-management-dengan-ansible-case-study-dcd0fe925064. [Accessed 8 Agustus 2023].

[7] M. Catherine and A. Kumawat, "Securing a Small Network by using Raspberry Pi Honeypot," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. IX, no. 7, pp. 773-778, 2020.

[8] I. Beres and D. Hurley-Smith, "Dynamic honeypot deployment in the cloud," 2022.

[9] M. Hasbi, A. R. A. Nurwa, D. F. Priambodo and W. R. A. Putra, "Infrastructure as Code for Security Automation and Network Infrastructure Monitoring," *Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer*, 2022.

[10] A. Khumaidi, "IMPLEMENTATION OF DEVOPS METHOD FOR AUTOMATION OF SERVER MANAGEMENT USING ANSIBLE," *TRANSFORMTIKA*, 2021.

[11] J. Buzzio-Garcia, "Creation of a High-Interaction Honeypot System based-on Docker containers," in *World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 2021.

[12] V. S. D. Priya and S. S. Chakkaravarthy, "Containerized cloud-based honeypot deception for tracking attackers," Scientific Reports, 2023.

[13] C. Wijayanto and Y. A. Susetyo, "IMPLEMENTASI FLASK FRAMEWORK PADA PEMBANGUNAN APLIKASI SISTEM INFORMASI HELPDESK (SIH)," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, 2022.

[14] D. F. Ningtyas and N. Setiyawati, "Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request," *Jurnal Janitra Informatika dan Sistem Informasi*, 2021.

[15] Nasution and L. Iswari, "Penerapan React JS Pada Pengembangan FrontEnd Aplikasi Startup Ubaform," *AUTOMATA*, 2021.

[16] N. Titarmare, N. Hargule and A. Gupta, "An Overview of Honeypot Systems," *International Journal of Computer Sciences and Engineering*, 2019.

[17] M. Zadka, DevOps in Python, 2nd ed., Apress, Berkeley, CA, 2022.

[18] M. D. Elradi, "Ansible: A Reliable Tool for Automation," *Electrical and Computer Engineering Studies*, vol. II, no. 1, 2023.

[19] P. Maha, "Konsep Dasar Ansible," 2020. [Online]. Available: https://medium.com/prastamaha/konsep-dasar-ansible-6e7451cb0502. [Accessed 7 Agustus 2023].

[20] J. Freeman and J. Keating, Mastering Ansible. Effectively automate configuration management and deployment challenges with Ansible 2.7, 3rd ed., Packt Publishing Ltd, 2019.

[21] Steffi, "What is Flask?," 2022. [Online]. Available: https://medium.com/data-science-ai-learning-journey/what-is-flask-cab5eb6e74f0. [Accessed 8 Agustus 2023].

[22] A. Boduch and R. Derks, React and React Native: A complete hands-on guide to modern web and mobile development with React.js, 3rd ed., Packt Publishing Ltd., 2020.

[23] R. Setiawan, "Apa Itu Docker? Apa Kegunaan dan Kelebihannya?," 2021. [Online]. Available: https://www.dicoding.com/blog/apa-itu-docker/. [Accessed 8 Agustus 2023].

[24] Kubernetes, "Apa itu Kubernetes?," 2020. [Online]. Available: https://kubernetes.io/id/docs/concepts/overview/what-is-kubernetes/. [Accessed 8 Agustus 2023].

[25] S. K. Mondal, R. Pan, H. M. D. Kabir, T. Tian and H.-N. Dai, "Kubernetes in IT administration and serverless computing: An empirical study and research challenges," *The Journal of Supercomputing*, 2021.

Analisis Perbandingan Kinerja *Container Network Interface Flannel* dan *Cilium* sebagai *Interface* Utama pada Multus CNI dalam Jaringan Klaster *Kubernetes*

Bayu Agung Prakoso[1], Unan Yusmaniar Oktiawati[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
bayu.a@mail.ugm.ac.id
*Korespondensi: unan_yusmaniar@ugm.ac.id;

Abstract – Container technology has become an alternative for virtualizing internet service infrastructure due to resource efficiency. An IT infrastructure can consist of numerous containers, with Kubernetes acting as Container Orchestration. The Container Network Interface (CNI) is used in Kubernetes service scenarios to manage networks, thus facilitating interconnection of services. However, issues such as limited network capability, lack of flexibility, and scalability, as well as security, arise in the use of CNI plugins. The solution to these problems is the Multus CNI, which allows multiple network interfaces on a single pod. This study evaluates the performance of Flannel and Cilium as CNI plugins in a Kubernetes Cluster environment involving Multus CNI. The metrics analyzed include latency, packet loss, throughput, and CPU usage. The research results will provide a better understanding of the trade-offs that must be made when choosing between Flannel and Cilium as CNI plugins in a Kubernetes Cluster environment.

Keywords – Kubernetes Cluster, CNI, Flannel, Cilium, Multus CNI

Intisari – *Container* menjadi alternatif virtualisasi infrastruktur layanan internet berkat efisiensi penggunaan sumber daya. Infrastruktur IT dapat terdiri dari beragam *container*, dengan *Kubernetes* berperan sebagai *Container Orchestration*. *Container* Network *Interface* (CNI) dipergunakan dalam skenario layanan pada *Kubernetes* untuk mengatur jaringan sehingga memudahkan terhubungnya layanan. Namun, masalah seperti kemampuan jaringan terbatas, kurangnya fleksibilitas, dan terbatasnya skalabilitas serta keamanan menjadi isu dalam penggunaan CNI *plugin*. Solusi atas persoalan tersebut adalah Multus CNI yang memungkinkan beragam antarmuka jaringan pada satu *pod*. Studi ini melakukan evaluasi kinerja antara *Flannel* dan *Cilium* sebagai *plugin* CNI dalam lingkungan *Kubernetes Cluster* dengan melibatkan Multus CNI. Metrik yang dianalisis mencakup *latency, packet loss, throughput,* dan CPU *usage*. Hasil penelitian akan menghasilkan pemahaman lebih baik mengenai kompromi yang harus dilakukan saat memilih antara *Flannel* dan *Cilium* sebagai *plugin* CNI dalam lingkungan *Kubernetes Cluster*.

Kata kunci – *Kubernetes Cluster*, CNI, *Flannel, Cilium,* Multus CNI

I. PENDAHULUAN

Dalam dunia yang terus bertransformasi menuju digitalisasi, kebutuhan akan layanan internet yang efisien menjadi prioritas. *Cloud computing* dan teknologi pendukungnya telah mendorong munculnya teknologi virtualisasi, di mana satu perangkat dapat menjalankan lebih dari satu sistem operasi secara bersamaan. Dari teknologi virtualisasi, kemudian berkembanglah teknologi *container* yang memungkinkan penggunaan *kernel* yang sama atau yang dikenal sebagai *shared kernel*, menambah efisiensi penggunaan sumber daya.

Dalam konteks layanan seperti *microservices*, konektivitas antar *container* menjadi vital, sehingga diperlukan aplikasi untuk mengatur dan menghubungkan berbagai *container* menjadi suatu *cluster* dengan *container orchestration*. Di antara berbagai jenis *orchestration* yang tersedia, *Kubernetes* muncul sebagai pilihan utama karena kemampuannya dalam mengotomatisasi penyebaran, skala, dan manajemen aplikasi *container* [1].

Kubernetes merupakan sistem orkestrasi kontainer *open-source* yang otomatis melakukan penyebaran, penskalaan, dan manajemen aplikasi yang dikontainerisasi [2]. Sistem ini menyediakan cara untuk mengorganisir dan mengelola beban kerja yang dikontainerisasi dan layanan di dalam klaster mesin [2]. *Kubernetes* telah digunakan dalam berbagai bidang, seperti penyebaran aplikasi yang dikontainerisasi pada penyedia *cloud* [3], implementasi sistem manajemen perangkat lunak non-kontainer [4], pengamanan klaster *Kubernetes* [5], mempercepat proses migrasi antar *cloud* [6], serta mengevaluasi performa dan skalabilitas dalam penyebaran kontainer [7].

Namun, dalam konfigurasi jaringan *container*, *Kubernetes* bergantung pada *plugin Container* Network *Interface* (CNI). Dua implementasi CNI yang populer adalah *Cilium*, yang telah melihat pertumbuhan besar dalam kontribusi dan adopsi [8], dan *Flannel*, implementasi CNI pertama untuk *Kubernetes* yang masih menjadi pilihan yang baik untuk penggunaan awal [9]. Untuk menambah fleksibilitas dan memungkinkan penggunaan berbagai *plugin* CNI dalam satu *cluster Kubernetes*, Multus CNI menjadi solusi.

Tujuan dari penelitian ini adalah untuk membandingkan kinerja dari *Flannel* dan *Cilium* ketika digunakan di atas Multus CNI, termasuk evaluasi *latency*, *packet loss*, *throughput*, dan CPU *usage*. Kontribusi utama dari penelitian ini adalah memberikan wawasan tentang bagaimana performa kedua *plugin* CNI dalam kondisi yang sama, yang dapat membantu pengguna *Kubernetes* dalam memilih strategi

jaringan yang paling sesuai dengan kebutuhan mereka. Penelitian ini menjadi penting karena, meski *Flannel* dan *Cilium* telah banyak digunakan dalam lingkungan produksi, penelitian yang membandingkan kinerja keduanya ketika digunakan di atas Multus CNI masih terbatas.

II. DASAR TEORI

Dalam evolusi teknologi informasi dan komunikasi, teknologi kontainerisasi telah menjadi titik balik penting dalam proses pengembangan dan penyebaran aplikasi. Kemampuan kontainerisasi untuk mengemas aplikasi dan dependensinya dalam format yang dapat ditransfer antar lingkungan berbeda menunjukkan efisiensi yang signifikan dalam meningkatkan proses *provisioning*, *delivering*, dan *maintaining* infrastruktur. *Kubernetes*, menjadi pilar dalam orkestrasi kontainer, memegang peran kunci dalam menerapkan keuntungan ini pada skala produksi yang besar. Salah satu elemen vital dalam struktur *Kubernetes* adalah *Container* Network *Interface* (CNI), yang merupakan spesifikasi dan serangkaian perangkat lunak yang merancang bagaimana kontainer berkomunikasi dengan jaringan. Terdapat penelitian yang telah melibatkan analisis perbandingan kinerja dan aplikasi antara *Flannel*, *Cilium*, dan Multus CNI, yang relevan dan mendukung penelitian ini dalam memperluas pemahaman tentang cara kerja dan kinerja mereka dalam lingkungan *Kubernetes*.

Penelitian yang berjudul "*Assessing Container* Network *Interface Plugins*: *Functionality*, *Performance*, *and Scalability*" membahas dua metode utama yang digunakan untuk pengiriman paket yaitu eBPF dan solusi berbasis *bridge*. Dalam penelitian itu eBPF digunakan untuk pengiriman paket antar-*host* dan memiliki *overhead* yang lebih tinggi dibandingkan dengan pengiriman paket intra-*host* karena titik tambahan di *host* untuk mendukung mode *overlay* dan proses *tunneling* VxLAN. Sebaliknya, *Flannel*, *Weave*, dan *Kube-router* menggunakan solusi berbasis *bridge* untuk pengiriman paket intra-*host*, di mana paket melewati Linux *bridge* dan mengeksekusi panggilan fungsi yang terkait dengan *bridge* [10]. Penelitian tersebut juga melakukan evaluasi kinerja berbagai *plugin* CNI, termasuk *Flannel*, *Calico*, *Weave*, *Cilium*, dan Kube-router. Hasil evaluasi menunjukkan bahwa *Flannel* dan *Weave* memiliki *overhead* terendah untuk komunikasi intra-*host*, sementara *Calico* dan *Cilium* memiliki *overhead* tertinggi [10]. Untuk komunikasi antar-*host*, *Calico* dan *Cilium* memiliki *overhead* terendah, sementara *Flannel* dan *Weave* memiliki *overhead* tertinggi [10]. Kesimpulannya, penelitian tersebut menyatakan bahwa pilihan *plugin* CNI tergantung pada kasus penggunaan dan kebutuhan khusus, dan tidak ada solusi yang cocok untuk semua skenario [10].

Dalam penelitian yang berjudul "*Performance Studies of Kubernetes Network Solutions*", [11] melakukan penilaian terhadap empat *plugin* CNI yang direkomendasikan oleh CNCF dalam lingkungan *datacentre* fisik. Penilaian ini bertujuan untuk mengevaluasi kinerja mereka dalam hal latensi dan rata-rata *throughput* TCP untuk berbagai ukuran *Maximum Transmission* Unit (MTU), jumlah berbeda dari antarmuka jaringan yang digabungkan, dan kondisi *offloading* segmen antarmuka yang berbeda. Selama pengujian, kinerja empat *plugin* CNI (*Flannel*, *Calico*, Kube-router, dan *Weave*) diukur dalam hal latensi dan rata-rata *throughput* TCP [11]. Dari segi kinerja, *Flannel* menunjukkan kinerja terbaik di antara *plugin overlay* yang diuji saat menggunakan MSS (*Maximum Segment Size*) yang kecil. Di sisi lain, *Calico* dan Kube-router, yang merupakan *plugin* non-*overlay*, menunjukkan *throughput* yang hampir sama dengan *throughput* baremetal [11]. Namun, penggunaan *Calico* dalam mode VXLAN memberikan hasil yang kurang memuaskan saat digunakan dengan ukuran MSS yang kecil [11]. Untuk *plugin Weave*, meskipun menggunakan teknologi *overlay* yang sama, performanya kurang dibandingkan *Flannel* [11]. Kesimpulannya, penelitian tersebut menyarankan bahwa kinerja *plugin* CNI sangat bergantung pada berbagai faktor seperti ukuran MTU, jumlah antarmuka jaringan yang digabungkan, dan kondisi *offloading* segmen antarmuka [11]. Pengguna disarankan untuk memilih *plugin* CNI berdasarkan kasus penggunaan dan persyaratan spesifik mereka [11].

Dalam penelitian mereka yang berjudul "*Network Policies in Kubernetes*: *Performance Evaluation and Security Analysis*", [12] mengevaluasi performa dan keamanan dari kebijakan jaringan dalam *Kubernetes*. Penelitian ini secara khusus menyoroti *overhead* performa dari solusi berbasis eBPF seperti *Calico* dan *Cilium*. Pengukuran *throughput* dan latensi *end-to-end* dilakukan menggunakan TCP *stream* mode dan *request-response* (RR) mode dari netperf, dengan tujuan untuk mendapatkan hasil pengukuran yang akurat [12]. Menurut hasil penelitian ini, kebijakan jaringan adalah solusi keamanan yang memiliki *overhead* rendah dan cocok untuk komunikasi *inter-container* yang membutuhkan latensi rendah. Penelitian ini juga menunjukkan bahwa *overhead* performa dari kebijakan jaringan hampir dapat diabaikan, dengan hanya sedikit variasi tergantung pada jumlah kebijakan dan berbagai resep kebijakan yang berbeda [12].

Penelitian yang berjudul "Multi-level Network *Software Defined Gateway Forwarding* System *Based on* Multus", mempresentasikan implementasi dan proposal sistem *gateway* penerusan yang ditentukan oleh perangkat lunak jaringan *multi*-level berdasarkan Multus. Metodologi yang diadopsi dalam penelitian ini mencakup penggunaan *Kubernetes* untuk manajemen terpusat dan kontrol *cluster* layanan, pemanggilan dinamis *plugin* CNI jenis berbeda untuk konfigurasi antarmuka, serta pendukung jaringan *multi*-level mode *kernel* dan mode pengguna [13]. Mereka melaksanakan tiga kelompok eksperimen komparatif untuk mengevaluasi sistem yang diusulkan. Hasilnya menunjukkan bahwa sistem yang diusulkan menunjukkan kinerja penerusan yang lebih tinggi dibandingkan perangkat *gateway* tradisional ketika dibatasi hingga 1GB memori yang tersedia [13]. Selain itu, penelitian ini menunjukkan bahwa sistem yang diusulkan memiliki skalabilitas yang baik, toleransi kesalahan, dan kemampuan penyeimbangan beban setelah pengenalan skema manajemen klaster *Kubernetes* [13].

Dalam penelitian yang berjudul "*Enhancement* in Multus CNI *for* DPDK *Applications* in *the Cloud Native Environment*", [14] mengusulkan penggunaan *plugin* Multus CNI dalam lingkungan *Kubernetes* untuk menyediakan berbagai antarmuka ke *pod*, dengan menggunakan konfigurasi jaringan dari *plugin* CNI sekunder. Penelitian ini memfokuskan pada implementasi kebijakan jaringan menggunakan *plugin* CNI sekunder untuk memproses paket data dalam aplikasi yang didukung DPDK yang berjalan di dalam kontainer [14]. Selain itu, penelitian ini juga mengusulkan integrasi CNI ruang pengguna dengan Multus CNI untuk memfasilitasi aplikasi yang didukung DPDK dengan menyediakan jaringan untuk pesawat kontrol. Hasil dari penelitian ini menunjukkan bahwa melalui penggunaan Multus CNI, *vhost-user* dapat digunakan oleh aplikasi DPDK untuk mengakses *stack* jaringan *kernel*. Pendekatan ini memberikan fleksibilitas dan skalabilitas pada penyebaran CNF dalam lingkup *cloud native* [14].

III. METODOLOGI

A. Alat dan Bahan

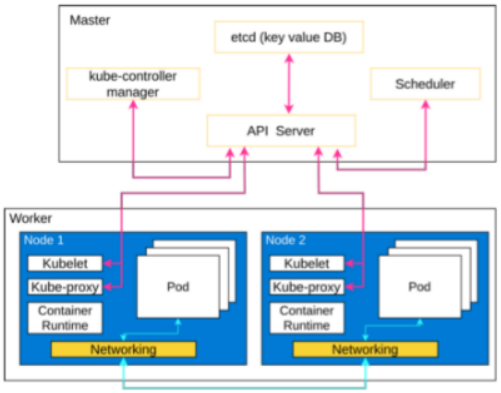
Dalam penelitian ini, digunakan beberapa perangkat keras dan perangkat lunak penting yang mendukung operasi dan hasil penelitian. Untuk perangkat keras, penelitian ini memanfaatkan klaster *Kubernetes* yang terdiri dari dua *node* pekerja dan satu *node* master. Spesifikasi untuk setiap *node* adalah 2 *vCPU*, 4 GB *memory*, dan *hard disk* 80 GB. Selain itu, penelitian ini juga menggunakan sebuah laptop sebagai titik akses dan kontrol utama, yang digunakan untuk mengakses layanan *cloud* dan menjalankan perangkat lunak yang dibutuhkan dalam penelitian. Spesifikasi laptop yang digunakan adalah sistem operasi Windows 10, CPU AMD *Ryzen* 5 5600H, 16 GB *memory*, dan *hard disk* 512 GB.

Untuk perangkat lunak, penelitian ini menggunakan beberapa perangkat berikut:

1. Ubuntu 20.04 x64: Sistem operasi berbasis Linux yang digunakan pada semua *node* dalam klaster.
2. *Kubernetes* versi 1.21: Platform open-*source* untuk orkestrasi kontainer yang digunakan untuk mengelola dan menjalankan aplikasi dalam kontainer.
3. Kubeadm: Alat yang membantu *mem-bootstrapping* klaster *Kubernetes*, memudahkan penyiapan dan konfigurasi komponen kunci *Kubernetes*.
4. *Python* versi 3.9.6: Bahasa pemrograman yang digunakan untuk menulis skrip otomatisasi pengaturan klaster dan menjalankan pengujian.
5. *Fabric*: Pustaka *Python* yang memudahkan otomatisasi perintah *shell* melalui SSH.
6. *Flannel*, *Cilium*, dan Multus CNI: *Plugin* CNI yang digunakan untuk memungkinkan komunikasi antar *pod* di klaster *Kubernetes*.
7. Iperf3: Alat untuk melakukan pengujian kinerja jaringan yang digunakan untuk mengevaluasi kinerja dari *plugin* CNI.

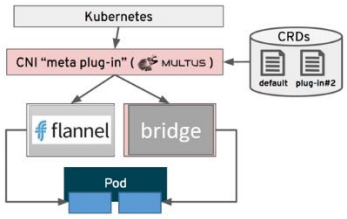
B. Perancangan Arsitektur

Penelitian ini merancang arsitektur dengan komponen utama *Kubernetes*: Master, *Worker*, dan *Networking* via CNI (*Container* Network *Interface*) seperti yang ditunjukkan oleh Gambar 1. Master, sebagai pengendali kluster, memiliki komponen ectd, *Scheduler*, *kube-controller-manager*, dan API Server. *Worker*, tempat menjalankan *pod*, terdiri dari *Kubelet*, *Container Runtime*, dan *Kube-proxy*.

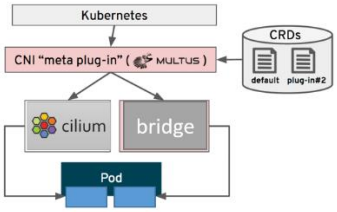


Gambar 1. Arsitektur *Kubernetes Cluster*

Model jaringan CNI diadopsi untuk *networking*, dengan menggunakan *Flannel*, *Cilium*, dan Multus CNI. *Flannel* dan *Cilium* digunakan secara bergantian di atas Multus CNI sesuai kebutuhan pengujian, sementara Multus CNI memungkinkan konfigurasi *multiple network interfaces* pada *pods* seperti yang ditunjukkan oleh Gambar 2 dan Gambar 3.

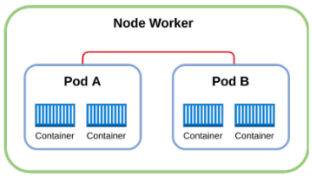


Gambar 2. Arsitektur *Flannel* di atas Multus CNI



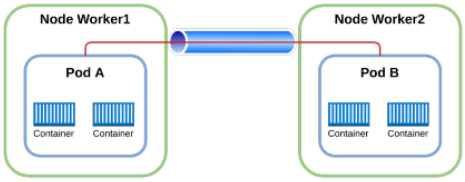
Gambar 3. Arsitektur *Cilium* di atas Multus CNI

Penelitian ini mengadopsi dua skenario pengujian: pengujian komunikasi antar *Pod* dalam satu *Node* seperti yang ditunjukkan oleh Gambar 4 dan antar *Pod* pada *Node* berbeda seperti yang ditunjukkan dalam Gambar 5. Skenario pertama relevan untuk kasus dengan ketergantungan sumber daya lokal atau spesifik, sementara skenario kedua membantu memaksimalkan ketersediaan dan ketahanan.



Gambar 4. Pengujian antar *Pod* dalam Satu *Node*

Tujuan pengujian ini adalah untuk mengevaluasi kinerja *Flannel* dan *Cilium* dalam penggunaan nyata, memberikan gambaran tentang performa mereka dalam berbagai situasi dan membantu dalam pemilihan solusi CNI terbaik untuk lingkungan produksi.



Gambar 5. Pengujian antar *Pod* berbeda *Node*

C. Pembuatan *Kubernetes Cluster*

Dalam penelitian ini, proses pembuatan kluster *Kubernetes* dilakukan dengan menggunakan *Python script* yang disokong oleh *library Fabric*. Berikut adalah langkah-langkahnya:

1. Pengunduhan Repositori: Kode sumber yang diperlukan dapat ditemukan di repositori *GitHub* pada *link* berikut: https://github.com/testdrivenio/kubernetes-fabric. Repositori ini dapat diunduh menggunakan perintah *clone Git*.
2. Penyiapan Lingkungan Virtual *Python*: Sebuah lingkungan virtual *Python* dibuat dan diaktifkan. Lingkungan ini menyediakan ruang kerja terisolasi untuk mengelola dependensi *Python*.
3. Pemasangan Dependensi: Semua dependensi yang diperlukan untuk menjalankan *script* dalam repositori diinstal menggunakan perintah *pip*.
4. Pendaftaran Akun Digital Ocean: Untuk menampung kluster, dibutuhkan server yang dalam konteks Digital Ocean disebut "*droplet*". Maka, perlu membuat akun di Digital Ocean.
5. Generasi *Token* Akses Digital Ocean: *Token* ini akan digunakan untuk otentikasi saat berinteraksi dengan API Digital Ocean.
6. Penambahan *Token* Akses ke Variabel Lingkungan: *Token* akses yang telah dibuat ditambahkan ke variabel lingkungan.
7. Penambahan Kunci SSH Publik ke Akun Digital Ocean: Kunci ini digunakan untuk otentikasi saat melakukan koneksi SSH ke *droplet*.
8. Pembuatan Kluster: Proses ini dimulai dengan menjalankan *script* create.sh. *Script* ini mengandung serangkaian instruksi untuk membuat kluster, termasuk pembuatan *droplets*, penentuan alamat IP, pemasangan *Docker*, pengaturan *Kubernetes*, dan lainnya.

Langkah-langkah selanjutnya adalah pemasangan dan konfigurasi Multus CNI dan *Flannel*.

D. Konfigurasi *Flannel* pada Multus CNI

Dalam tahap ini, *Flannel* dikonfigurasikan sebagai solusi CNI pada Multus untuk menciptakan lingkungan yang diuji. Hal ini dilakukan dengan menginstal *Flannel* ke dalam kluster *Kubernetes* dan kemudian membuat Network *Attachment Definition* yang merujuk ke *Flannel* sebagai CNI *plugin*. Proses konfigurasi ini melibatkan beberapa langkah teknis termasuk penyetelan jaringan virtual, penjadwalan *Pod*, dan penentuan peraturan lalu lintas jaringan.

E. Uji *Performance* dan Pengambilan Data pada *Flannel*

Setelah *Flannel* berhasil dikonfigurasikan pada Multus CNI, tahap pengujian *performance* dan pengambilan data dimulai. Pengujian dilakukan dengan skenario komunikasi antar *Pod* dalam satu *Node* dan antar *Node* yang berbeda. Pengukuran utama yang diambil adalah *latency*, *packet loss*, *throughput*, dan penggunaan CPU. Untuk menjamin validitas dan reliabilitas data, setiap pengujian dijalankan beberapa kali dan hasil rata-rata digunakan sebagai representasi kinerja.

F. Migrasi *Default* CNI dari *Flannel* ke *Cilium*

Tahap ini melibatkan migrasi CNI *default* dari *Flannel* ke *Cilium*. Langkah pertama adalah membuat *snapshot* cadangan untuk semua *node* sebelum migrasi, dilanjutkan dengan proses migrasi yang melibatkan penghapusan Network *Attachment Definition* yang ada untuk *Flannel*, penghapusan *Flannel* dari kluster, dan instalasi *Cilium*. Setelah migrasi selesai, semua *node* diaktifkan kembali dan status *node* diverifikasi.

G. Konfigurasi *Cilium* pada Multus CNI

Konfigurasi *Cilium* pada Multus CNI dilakukan setelah proses migrasi selesai. Hal ini melibatkan penyesuaian berbagai parameter dan konfigurasi untuk mengoptimalkan kinerja *Cilium*. Dalam tahap ini, Network *Attachment Definition* baru dibuat untuk *Cilium* dan penyesuaian konfigurasi Multus CNI dilakukan untuk merujuk ke *Cilium* sebagai *plugin* CNI *default*.

H. Uji *Performance* dan Pengambilan Data pada *Cilium*

Seperti pada tahap pengujian *Flannel*, tahap pengujian untuk *Cilium* melibatkan evaluasi kinerja dalam dua skenario yaitu antar *Pod* dalam satu *Node* dan antar *Node* yang berbeda. Pengukuran yang sama - *latency*, *packet loss*, *throughput*, dan penggunaan CPU - juga digunakan dalam tahap ini. Seperti sebelumnya, setiap pengujian dijalankan beberapa kali dan hasil rata-rata digunakan sebagai representasi kinerja.

I. Analisa Hasil

Tahap akhir dari metodologi penelitian ini adalah analisis hasil pengujian. Analisis ini difokuskan pada evaluasi kinerja *Flannel* dan *Cilium* dalam berbagai skenario pengujian berdasarkan hasil pengukuran *latency*, *packet loss*, *throughput*, dan penggunaan CPU. Selain itu, analisis juga melibatkan perbandingan hasil pengujian kedua solusi

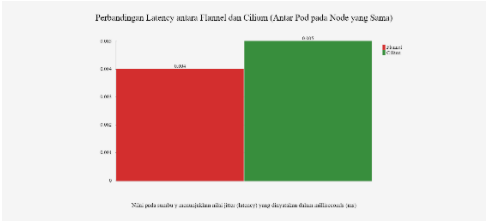
tersebut untuk memberikan gambaran yang jelas tentang kinerja dan efisiensi masing-masing dalam konteks penggunaan nyata. Dengan demikian, hasil analisis ini dapat digunakan untuk membuat keputusan yang lebih berinformasi tentang pemilihan solusi CNI terbaik untuk implementasi dalam lingkungan produksi.

IV. HASIL DAN PEMBAHASAN

Analisis berfokus pada penjelasan dan evaluasi dari hasil pengujian yang telah diimplementasikan. Fokus pengujian adalah tiga jenis *Container* Network *Interface* (CNI) yang berbeda, yakni *Flannel*, *Cilium*, dan Multus CNI. Dalam penelitian ini digunakan kombinasi *Flannel* dan Multus CNI, serta kombinasi *Cilium* dan Multus CNI secara bergantian untuk mengamati dampaknya terhadap efisiensi komunikasi antar *Pod* dalam kluster *Kubernetes*. Diadopsi dua skenario pengujian utama yaitu pengujian komunikasi antar *Pod* dalam satu *Node* yang sama dan pengujian komunikasi antar *Pod* yang berada pada *Node* yang berbeda. Penampilan hasil pengujian akan disajikan dalam format grafik untuk memfasilitasi proses analisis hasil yang lebih efisien.

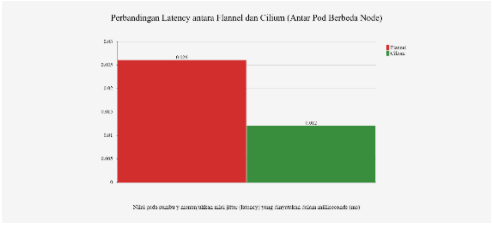
A. Perbandingan Kinerja dalam Aspek *Latency*

Dalam pengujian *performance* dalam aspek *latency*, dilakukan perbandingan antara *Flannel* dan *Cilium* dalam dua skenario yang berbeda: komunikasi antar *pod* dalam satu *node* yang sama dan komunikasi antar *pod* dalam *node* yang berbeda. *Performance latency* antar *pod* dalam satu *node* ditunjukkan pada Gambar 6, sedangkan *performance latency* antar *pod* berbeda *node* ditunjukkan pada Gambar 7.



Gambar 6. *Performance Latency* antar *Pod* dalam Satu *Node*

Berdasarkan hasil pengujian *latency* pada skenario antar *pod* dalam *node* yang sama, *Flannel* mencatat *jitter* sebesar 0.004 ms, sedangkan *Cilium* menunjukkan nilai sedikit lebih tinggi, yaitu 0.005 ms seperti yang ditunjukkan pada Gambar 6.

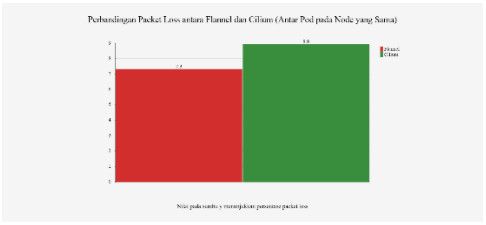


Gambar 7. *Performance Latency* antar *Pod* Berbeda *Node*

Pada skenario komunikasi antar *pod* di antara *node* yang berbeda, *Flannel* mencatat *jitter* sebesar 0.026 ms, sementara *Cilium* menunjukkan nilai yang jauh lebih rendah, yaitu 0.012 ms seperti yang ditunjukkan pada Gambar 7.

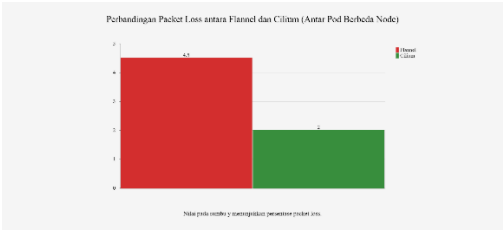
B. Perbandingan Kinerja dalam Aspek *Packet Loss*

Pada skenario antar *pod* dalam *node* yang sama, ditemukan bahwa *Flannel* mengalami *packet loss* sebesar 7,3%, sedangkan *Cilium* mengalami *packet loss* sebesar 8,9% seperti yang ditunjukkan pada Gambar 8.



Gambar 8. *Performance Latency* antar *Pod* dalam Satu *Node*

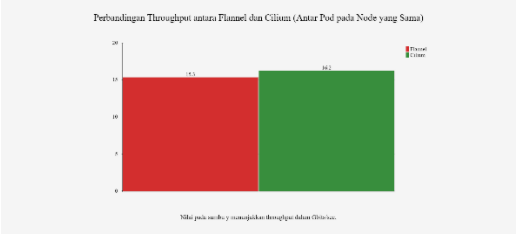
Sedangkan dalam skenario antar *pod* di antara *node* yang berbeda, *Flannel* diperoleh *packet loss* sebesar 4.5%, sedangkan pada *Cilium*, *packet loss* hanya sebesar 2% seperti yang ditunjukkan pada Gambar 9.



Gambar 9. *Performance Latency* antar *Pod* Berbeda *Node*

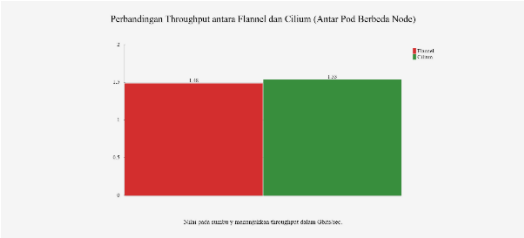
C. Perbandingan Kinerja dalam Aspek *Throughput*

Untuk skenario antar *pod* dalam *node* yang sama, *Flannel* menghasilkan *throughput* sebesar 15.3 Gbits/*sec*, sementara *Cilium* menghasilkan *throughput* yang lebih tinggi, yaitu sebesar 16.2 Gbits/*sec* seperti yang ditunjukkan pada Gambar 10.



Gambar 10. *Performance Latency* antar *Pod* dalam Satu *Node*

Sedangkan dalam skenario antar *pod* di antara *node* yang berbeda, *Flannel* mencapai *throughput* sekitar 1.48 Gbits/*sec*, sementara *Cilium* mencapai 1.53 Gbits/*sec* seperti yang ditunjukkan pada Gambar 11.

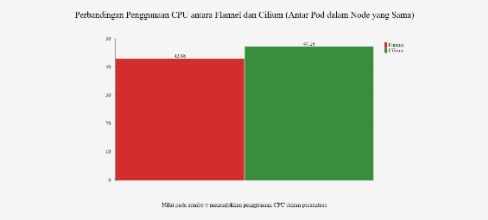


Gambar 11. *Performance Latency* antar *Pod* Berbeda *Node*

Mengenai peningkatan *throughput*, penting untuk mempertimbangkan peningkatan ini dalam konteks fungsi dan fitur tambahan yang disediakan oleh *Cilium*. Salah satu keunggulan penting *Cilium* dibanding *Flannel* adalah dukungan terhadap BPF (*Berkeley Packet* Filter), yang menyediakan kecepatan dan fleksibilitas yang lebih baik dalam pengolahan paket serta penerapan kebijakan jaringan.

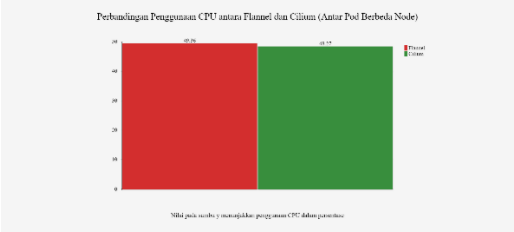
D. Perbandingan Kinerja dalam Aspek CPU *Usage*

Untuk skenario antar *pod* dalam *node* yang sama, *Flannel* mencapai penggunaan CPU sebesar 42.88%, sedangkan *Cilium* mencapai penggunaan CPU sebesar 47.16% seperti yang ditunjukkan pada Gambar 12.



Gambar 12. *Performance Latency* antar *Pod* dalam Satu *Node*

Sedangkan dalam skenario antar *pod* di antara *node* yang berbeda, *Flannel* mencapai penggunaan CPU sebesar 49.36%, sedangkan *Cilium* mencapai penggunaan CPU sebesar 48.27% seperti yang ditunjukkan pada Gambar 13.



Gambar 13. *Performance Latency* antar *Pod* Berbeda *Node*

Penggunaan CPU ini sebanding dengan efisiensi yang ditawarkan oleh setiap CNI. *Flannel*, sebagai solusi jaringan *overlay* yang sederhana, dapat mengonsumsi lebih banyak CPU karena *overhead* yang terkait dengan enkapsulasi paket. Di sisi lain, *Cilium*, yang menggunakan teknologi BPF (*Berkeley Packet* Filter) modern, mungkin lebih efisien dalam penggunaan CPU, yang dapat menjelaskan penggunaan CPU yang sedikit lebih rendah.

Namun, juga penting untuk diperhatikan bahwa hasil ini mungkin dipengaruhi oleh spesifikasi *node*, termasuk jumlah *vCPU*. Dalam penelitian ini, *node* yang digunakan memiliki dua *vCPU*, dan perbedaan dalam penggunaan CPU mungkin sebagian dihasilkan dari sejauh mana masing-masing CNI dapat memanfaatkan sumber daya tersebut.

V. SIMPULAN

Melalui pengujian dan analisis data yang telah dilakukan, beberapa penemuan penting dapat disimpulkan. Dalam konteks latensi, ketika komunikasi terjadi antar *pod* dalam satu *node*, *Flannel* menunjukkan kinerja yang sedikit lebih baik dengan latensi yang lebih rendah dibandingkan dengan *Cilium*. Namun, ketika skenarionya melibatkan komunikasi antar *node* yang berbeda, *Cilium* mengungguli *Flannel* dengan latensi yang lebih rendah. Perbedaan ini bisa memiliki dampak signifikan pada performa aplikasi, terutama bagi aplikasi yang sangat sensitif terhadap latensi.

Kemudian, dalam aspek *packet loss*, *Flannel* menunjukkan efisiensi yang lebih baik dalam komunikasi antar *pod* dalam satu *node*, dengan *packet loss* sebesar 7,3%, dibandingkan dengan *Cilium* yang memiliki *packet loss* sebesar 8,9%. Namun, dalam skenario komunikasi antar *node*, *Cilium* unggul dengan *packet loss* sebesar 2% dibandingkan dengan *Flannel* yang mencapai 4,5%. Kinerja kedua solusi CNI ini tampaknya dapat bervariasi tergantung pada skenario penggunaan.

Selanjutnya, dalam hal *throughput*, *Cilium* menunjukkan performa yang sedikit lebih baik dibandingkan dengan *Flannel* dalam kedua skenario tersebut. Keunggulan lain dari *Cilium* terletak pada dukungannya terhadap BPF, yang memberikan kecepatan dan fleksibilitas yang lebih tinggi dalam pengolahan paket dan implementasi kebijakan jaringan.

Akhirnya, dalam pengukuran penggunaan CPU, *Cilium* umumnya memerlukan sedikit lebih banyak sumber daya CPU dibandingkan dengan *Flannel*, baik dalam skenario komunikasi di dalam *node* maupun antar *node*. Namun, mempertimbangkan bahwa *Cilium* memanfaatkan BPF, ini bisa menandakan bahwa *Cilium* lebih efisien dalam penggunaan CPU, meski perbedaan ini tampaknya kecil.

REFERENSI

[1] Datadog, "9 insights on real world container use," November 2022. [Online].

[2] C. Carrión, "Kubernetes scheduling: Taxonomy, ongoing issues and challenges," *ACM Comput. Surv.*, vol. 55, p. 1–37, July 2022.

[3] N. M. P. G. Sindhu G, "Deploying a Kubernetes Cluster With Kubernetes Operation Kops On Aws Cloud Experiments And Lessons Learned," *IJEAT*, vol. IJEAT, p. 984–989, 2020.

[4] G. Shi, W. Cai, J. Zhang, C. Gao, S. Sun, Y. Zhang and Y. Jiang, "KubeCOM: an implementation of a non-containerized software management system based on Kubernetes," in *Third International Conference on Computer Science and Communication Technology (ICCSCT 2022)*, Beijing, 2022.

[5] A. Rahman, S. I. Shamim, D. B. Bose and R. Pandita, "Security Misconfigurations In Open Source Kubernetes Manifests: An Empirical Study," *ACM Trans. Softw. Eng. Methodol*, vol. 32, p. 1–36, 2023.

[6] F. R. Poetra, S. Prabowo, S. A. Karimah and R. D. Prayogo, "Performance analysis of video streaming service migration using container orchestration," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 830, p. 022100, April 2020.

[7] G. Avolio, M. Cadeddu and R. Hauser, "Evaluating Kubernetes as an orchestrator of the Event Filter computing farm of the Trigger and Data Acquisition system of the ATLAS experiment at the Large Hadron Collider," *EPJ Web Conf.*, vol. 214, p. 07024, 2019.

[8] C. N. C. Foundation, "Announcing the Cilium annual report," January 2023. [Online].

[9] M. Mackrory, "The ultimate guide to using calico, flannel, weave and cilium," June 2021. [Online].

[10] S. Qi, S. G. Kulkarni and K. K. Ramakrishnan, "Assessing container network interface plugins: Functionality, performance, and scalability," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, p. 656–671, March 2021.

[11] N. Kapocius, "Performance studies of kubernetes network solutions," in *2020 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream)*, Vilnius, 2020.

[12] G. Budigiri, C. Baumann, J. T. Muhlberg, E. Truyen and W. Joosen, "Network policies in kubernetes: Performance evaluation and security analysis," in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, Porto, 2021.

[13] Z. Wang, Y. Ji, W. Zheng and M. Li, "Multi-level Network Software Defined Gateway Forwarding System Based On Multus," in *Proceeding of 2021 International Conference on Wireless Communications, Networking and Applications*, 2022, p. 166–176.

[14] A. Ayub, M. Ishaq and M. Munir, "Enhancement in multus CNI for DPDK applications in the cloud native environment," in *2023 26th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, Paris, 2023.

Pengembangan Aplikasi AR Cinurawa untuk Agen Pengembang Properti Perumahan

Moh Herlambang Akasyah[1], Dinar Nugroho Pratomo[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
begeniro@mail.ugm.ac.id
[*]Korespondensi: dinar.nugroho.p@ugm.ac.id;

Abstract – The housing industry has experienced significant growth in recent years. However, the progress in the real estate industry's performance has not been balanced with the implementation of technology, resulting in home purchases still causing confusion and burden for many consumers. The activities involved in real estate transactions often require substantial resources, making them inefficient. The current advancement in hardware technology allows for easy computation and display of 3D objects. The ease of creating 3D models on computers can be utilized to provide a more detailed representation of houses to be sold, without the need to wait for the construction of sample houses by real estate agents. The created 3D house models can be presented to prospective buyers through various means, one of which is using Android-based augmented reality (AR) technology. The AR application utilizes the Marker-Based method and is developed using Unity and Vuforia. The results of this research indicate that the use of augmented reality applications can assist in the marketing of housing properties. The application offers a more interactive experience and facilitates a clearer understanding of the properties being offered for potential buyers. AR enables prospective buyers to view realistic 3D house models through their mobile devices. This AR application has the potential to enhance the effectiveness of housing property marketing.

Keywords – Augmented Reality, Unity, Vuforia, Marker-Based, Residential Property

Intisari – Industri perumahan telah mengalami pertumbuhan yang signifikan dalam beberapa tahun terakhir. Peningkatan dalam performa industri properti tidak berimbang dengan penerapan teknologi yang mengakibatkan pembelian rumah masih dapat menimbulkan kebingungan dan beban bagi banyak konsumen. Aktivitas transaksi properti perumahan yang memerlukan banyak sumber daya seringkali tidak efisien. Perkembangan teknologi perangkat keras saat ini memungkinkan komputasi untuk menampilkan objek 3D dapat dilakukan dengan mudah. Kemudahan dalam pembuatan model 3D pada komputer dapat dimanfaatkan untuk memberikan gambaran yang lebih detail tentang rumah yang akan ditransaksikan, tanpa harus menunggu pembangunan rumah sampel dari agen industri perumahan. Model 3D rumah yang telah dibuat dapat disampaikan kepada calon pembeli melalui berbagai cara, salah satunya menggunakan teknologi *augmented reality* (AR) berbasis Android. Aplikasi AR ini menggunakan metode *Marker-Based* dan dikembangkan menggunakan *Unity* dan *Vuforia*. Hasil dari penelitian ini menunjukkan bahwa penggunaan aplikasi *augmented reality* dapat membantu dalam pemasaran properti perumahan. Aplikasi ini memberikan pengalaman yang lebih interaktif dan memudahkan calon pembeli untuk memahami dengan lebih jelas tentang properti yang ditawarkan. AR memungkinkan calon pembeli melihat model 3D rumah secara nyata melalui perangkat *mobile* mereka. Aplikasi AR ini memiliki potensi untuk meningkatkan efektivitas dalam pemasaran properti perumahan.

Kata kunci – *Augmented Reality, Unity, Vuforia, Marker-Based,* Properti Perumahan

I. PENDAHULUAN

Industri perumahan telah mengalami pertumbuhan yang signifikan dalam beberapa tahun terakhir. Sampai dengan kuartal 3 tahun 2022, kinerja tahunan (*year on year*) sektor industri properti (*real estate*) bertumbuh sebesar 2,17% melanjutkan pertumbuhan tahun sebelumnya yang juga positif. Kinerja sektor industri properti (*real estate*) secara per kuartal (*q on q*) pada kuartal 3 tahun 2022, juga mengalami pertumbuhan sebesar 0,02% dari kuartal sebelumnya [1].

Peningkatan performa industri properti tidak sejalan dengan penerapan teknologi yang memudahkan transaksi, menyebabkan kebingungan dan beban bagi konsumen saat membeli rumah. Agen perumahan masih menggunakan gambar "rumah sampel" pada brosur untuk menarik calon pembeli, tetapi metode tradisional ini memiliki keterbatasan dalam memberikan gambaran yang menyeluruh dan menarik. Calon pembeli harus mengunjungi rumah sampel untuk mendapatkan gambaran yang detail, yang juga merepotkan bagi agen dan memakan waktu dan biaya.

Kegiatan transaksi properti perumahan yang membutuhkan banyak sumber daya juga tidak efisien. Dalam upaya meningkatkan efisiensi, pengembangan teknologi perangkat keras memungkinkan komputasi yang mudah untuk menampilkan objek 3D. Model 3D rumah dapat digunakan sebagai medium untuk memberikan gambaran mendetail tanpa harus menunggu pembangunan rumah sampel.

Penggunaan teknologi *augmented reality* (AR) berbasis Android menjadi salah satu cara untuk menyampaikan model 3D rumah kepada calon pembeli dengan pengalaman yang interaktif dan mendetail. Teknologi ini dapat menampilkan model 3D dari rumah yang ada di buku katalog dari berbagai sudut dan membantu dalam pemasaran properti perumahan [2]. Penerapan teknologi AR mampu meningkatkan kinerja dengan memungkinkan banyak calon pembeli untuk dapat memvisualisasikan bagaimana gambaran detail properti yang akan dibeli terlihat dan meminimalkan kegiatan mereka untuk mengunjungi rumah sampel ketika akan membuat keputusan untuk membeli [3]. Teknologi *Augmented Reality* (AR) merupakan aplikasi yang menggabungkan dunia nyata dengan dunia digital berbentuk 2D maupun 3D yang secara bersamaan diproyeksikan di lingkungan dunia nyata [4]. AR dapat mengubah brosur perumahan menjadi lebih hidup dan memungkinkan pengguna melihat representasi virtual dari

properti yang akan dibeli. Teknologi AR juga telah digunakan dalam berbagai sektor seperti pendidikan, industri, dan hiburan. Penelitian ini fokus pada penerapan AR dalam industri properti perumahan.

II. DASAR TEORI

A. *C#*

C# adalah bahasa pemrograman tingkat tinggi yang dikembangkan oleh Microsoft. Bahasa ini dirancang untuk membangun aplikasi yang berjalan di platform Microsoft.NET [5]. C# memiliki sintak yang mirip dengan bahasa C++ dan Java, tetapi dengan fitur-fitur tambahan. C# digunakan secara luas dalam pengembangan aplikasi *desktop*, *web*, dan *mobile*, serta dalam pengembangan permainan. Bahasa pemrograman C# menawarkan dukungan yang kuat untuk pemrograman berorientasi objek, pemrograman generik, dan pemrograman berbasis komponen.

B. *Object Oriented Programing*

Object Oriented Programming (OOP) adalah pendekatan dalam pengembangan perangkat lunak yang mengatur perangkat lunak sebagai kumpulan objek yang memiliki data dan fungsi-fungsi terkait. OOP merupakan teknik pemrograman yang berfokus pada pengorganisasian dan pengelompokan konsep ke dalam objek yang memiliki atribut dan perilaku tertentu. OOP dapat membantu memecah kompleksitas sistem menjadi bagian-bagian yang lebih kecil dan terorganisir, sehingga mempermudah pemeliharaan, pengembangan, dan reusabilitas kode [6].

C. *Model 3D*

Model 3D merupakan objek tiga dimensi yang memiliki bentuk, volume, dan ruang yang ditentukan oleh koordinat X, Y, dan Z. Model 3D terdiri dari elemen-elemen pembentuk seperti *Vertex*, *Edge*, dan *Face*. *Vertex* merupakan titik yang dinyatakan dalam koordinat X, Y, Z. Ketika dua *Vertex* digabungkan, mereka membentuk *Edge*. *Vertex* dan *Edge* yang membentuk bidang permukaan akan membentuk *Face*. Kombinasi dari *Vertex*, *Edge*, dan *Face* ini membentuk sebuah objek utuh yang disebut dengan *Mesh* [7].

D. *Augmented Reality*

Augmented Reality (AR) merupakan teknologi yang menggabungkan objek maya dalam bentuk dua dimensi atau tiga dimensi ke dalam lingkungan nyata, kemudian memproyeksikan objek maya tersebut secara berkelanjutan dalam waktu nyata [4].

E. *Unity Game Engine*

Unity adalah sebuah *Game Engine* yang dikembangkan oleh Unity Technologies. *Unity* pertama kali diumumkan dan diluncurkan pada bulan Juni 2005 di *Apple Worldwide Developers Conference* sebagai *game engine* eksklusif untuk platform Mac OS X [8]. Hingga Tahun 2023, *Unity* telah dikembangkan untuk mendukung lebih dari 25 platform [8].

F. *Vuforia*

Vuforia adalah sebuah *Software Development Kit* (SDK) untuk membuat aplikasi *Augmented Reality* pada berbagai perangkat, seperti *smartphone*, tablet, dan perangkat *wearable*. *Vuforia* dikembangkan oleh perusahaan *software* Amerika Serikat bernama PTC dan dapat diintegrasikan dengan *Unity Game Engine* [9].

G. *Multimedia Development Life Cycle*

Multimedia Development Life Cycle (MDLC) merupakan suatu siklus pengembangan produk multimedia yang dimulai dengan tahap analisis produk, pengembangan produk, dan peluncuran [10]. Meskipun memiliki akar pengembangan yang sama dengan *Software Development Life Cycle* (SDLC), MDLC memiliki karakteristik yang unik terkait dengan pengembangan dan penggunaan elemen multimedia. Aplikasi permainan memiliki kompleksitas dalam pengembangan aplikasi yang menyebabkan model pengembangan yang terpisah dan lebih spesifik menggunakan *Game Development Life Cycle* (GDLC) meski merupakan bagian dari produk multimedia non-linear. Secara umum, MDLC digunakan untuk membangun produk multimedia linear dan non-linear.

H. *Unified Modeling Language*

Unified Modeling Language (UML) merupakan standar bahasa yang digunakan di industri untuk mendefinisikan *requirement*, melakukan analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek [11]. UML menyediakan notasi yang jelas dan konsisten untuk mengkomunikasikan ide, konsep, dan struktur dalam pengembangan perangkat lunak. Melalui penggunaan diagram-diagram yang terdefinisi dalam UML seperti *Use Case*, *Class*, *Sequence*, dan lainnya.

I. *Use Case Diagram*

Use case diagram adalah salah satu bentuk pemodelan yang digunakan untuk menggambarkan penggunaan dari sistem yang akan dibangun. Diagram ini berfokus pada interaksi antara pengguna sistem (aktor) dengan sistem itu sendiri melalui sebuah cerita atau skenario penggunaan. *Use case diagram* membantu dalam mengidentifikasi fungsi-fungsi utama yang akan diimplementasikan dalam sistem.

J. *Black Box Testing*

Black box testing merupakan metode pengujian yang dilakukan untuk memeriksa apakah semua fungsi perangkat lunak berjalan dengan benar sesuai dengan kebutuhan fungsional yang telah ditentukan, tanpa memperhatikan struktur atau implementasi internal dari perangkat lunak tersebut [12].

III. METODOLOGI

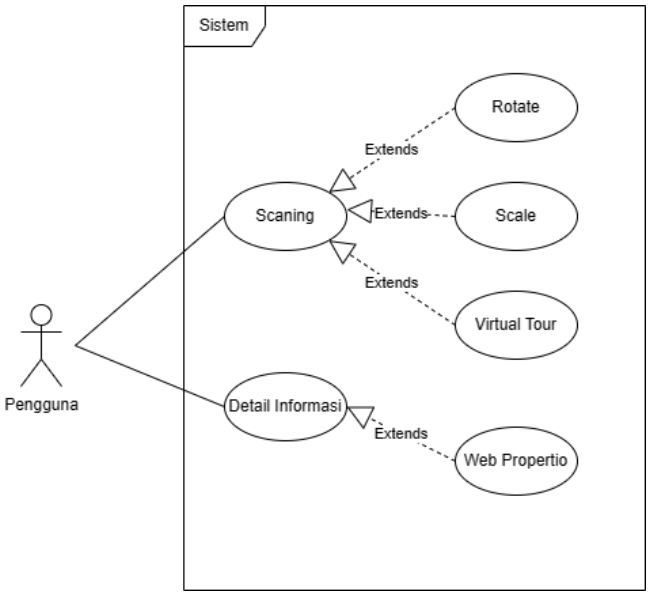
Metode yang akan digunakan pada pengembangan aplikasi adalah *Multimedia Development Life* Cycle (MDLC) versi Luther-Sutopo [10]. MDLC memiliki 6 tahap yang perlu dilakukan dalam pengembang di antaranya sebagai berikut.

A. Konsep *(Concept)*

Proses transaksi jual beli properti perumahan masih menggunakan metode "rumah sampel" sebagai penggambaran rumah yang akan ditransaksikan. Pihak agen perumahan diharuskan untuk membangun rumah terlebih dahulu untuk dapat memberikan gambaran detail rumah kepada calon pembeli, sedangkan calon pembeli diharuskan untuk datang ke lokasi "rumah sampel" supaya dapat mengetahui detail rumah yang akan dibeli. Tujuan dikembangkannya aplikasi AR perumahan merupakan sebagai medium untuk menjembatani pihak agen perumahan dan calon pembeli dalam memperlihatkan dan melihat detail properti yang akan ditransaksikan tanpa harus menghabiskan banyak sumber daya.

B. Perancangan *(Design)*

Use case diagram merupakan diagram yang dibuat untuk menggambarkan skenario sistem dari sisi penggunanya. *Use case diagram* pada aplikasi AR memiliki 1 aktor yang melambangkan pengguna aplikasi yang akan dikembangkan. *Use case diagram* aplikasi AR dapat dilihat pada Gambar 1.



Gambar 1. *Use case diagram* Aplikasi AR

C. Pengumpulan bahan *(Material collecting)*

Pengumpulan bahan merupakan tahap dalam pengembangan aplikasi AR untuk mengumpulkan bahan yang dibutuhkan dalam proses pengembangan aplikasi AR. Bahan yang dibutuhkan dalam pengembangan aplikasi AR adalah sebagai berikut:

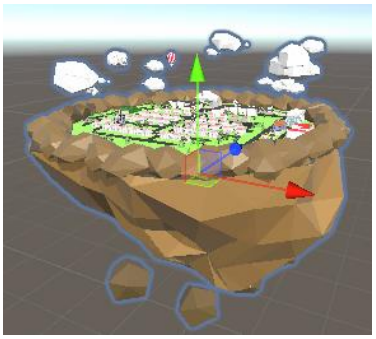
1. Model 3D Rumah

Model 3D rumah adalah objek yang akan muncul ketika kamera berhasil mengidentifikasi markah yang telah ditentukan. Model ini akan dipakai untuk dua markah, yaitu markah eksterior dan markah interior dengan menghilangkan atap dari model rumah. Model 3D rumah dapat dilihat pada Gambar 2.

Gambar 2. Model rumah

2. Model 3D *Siteplan*

Model 3D *siteplan* adalah objek yang muncul ketika kamera berhasil mengidentifikasi markah untuk *siteplan*. Model 3D *siteplan* dapat dilihat pada Gambar 3.



Gambar 3. Model *siteplan*

3. Gambar Markah

Gambar markah adalah gambar yang digunakan untuk mengidentifikasi di mana dan objek apa yang akan dimunculkan ketika kamera berhasil mengidentifikasi gambar. Aplikasi AR yang akan dikembangkan akan menggunakan tiga gambar markah yang terintegrasi dengan brosur seperti yang dapat dilihat pada Gambar 4.

Gambar 4. Brosur Cinurawa

D. Perakitan *(Assembly)*

Aplikasi AR dikembangkan menggunakan *Unity Game Engine* 2021.3.16f1 dan *Vuforia Engine* 10.12.3. Antarmuka (UI) pada aplikasi AR menggunakan objek *canvas* bertipe "*Screen space – overlay*" supaya elemen yang ada pada *ui* selalu ditampilkan pada layar dan tidak tertutup objek lain. Aplikasi memiliki menu utama yang berisi dua tombol, di antaranya tombol untuk membuka *AR Camera* dan tombol untuk membuka halaman informasi perumahan. Halaman *AR Camera* yang belum menemukan gambar markah yang telah ditentukan dapat dilihat pada Gambar 5.

Gambar 5. Halaman *AR Camera*

1. Eksterior Rumah

AR Camera akan menampilkan model rumah eksterior ketika kamera berhasil memindai gambar markah yang telah ditentukan. Tampilan *AR Camera* ketika berhasil memindai gambar markah rumah eksterior dapat dilihat pada Gambar 6.

Gambar 6. *Scan* rumah eksterior

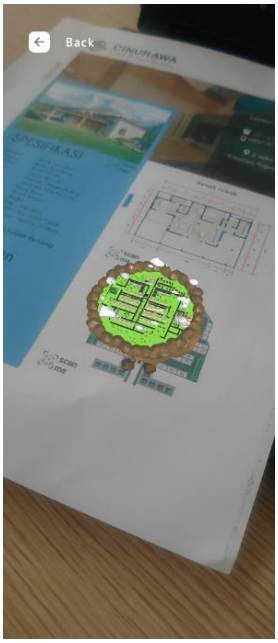
2. Interior Rumah

AR Camera akan menampilkan model rumah interior ketika kamera berhasil memindai gambar markah yang telah ditentukan. Tampilan *AR Camera* ketika berhasil memindai gambar markah rumah interior dapat dilihat pada Gambar 7.

Gambar 7. *Scan* rumah interior

3. Lingkungan *Siteplan*

AR Camera akan menampilkan model *siteplan* ketika kamera berhasil memindai gambar markah yang telah ditentukan. Tampilan *AR Camera* ketika berhasil memindai gambar markah *siteplan* dapat dilihat pada Gambar 8.



Gambar 8. *Scan Siteplan*

E. Pengujian *(Testing)*

Tahap pengujian bertujuan untuk memastikan bahwa aplikasi yang telah dikembangkan berjalan sesuai dengan yang telah disepakati. Pengujian akan dilakukan dengan menggunakan metode *black box* ke seluruh fungsi yang dikembangkan untuk mengetahui apakah fungsi sudah berjalan dengan baik atau masih terdapat permasalahan.

F. Distribusi *(Distribution)*

Tahap distribusi merupakan tahap terakhir dalam MDLC yang berfungsi sebagai tahap dalam merilis aplikasi ke publik. Tahap ini dilaksanakan apabila aplikasi sudah benar-benar siap dan telah melewati tahap testing.

IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian Black Box Aplikasi AR

Pengujian aplikasi juga dilakukan dengan mengajukan kuesioner ke pihak terkait untuk mengevaluasi fungsionalitas aplikasi. Pengisi kuesioner merupakan tim pengembang dan pemasaran properti perumahan. Hasil dari pengujian yang dilakukan oleh perwakilan dari empat pengembang properti perumahan dapat dilihat pada Tabel 1.

Tabel 1. Pengujian *black box*

No	Elemen	Deskripsi	Kondisi	Hasil yang diharapkan	Hasil Aktual
1	Tombol "Info Perumahan "	Pengguna menekan tombol		Membuka halaman “Info Perumahan”	OK
2	Tombol "*Back*"	Pengguna menekan tombol	Pengguna berada di halaman “Info Perumahan ”	Membuka halaman “Menu Utama”	OK
3	Tombol "*Scan Now*"	Pengguna menekan tombol		Membuka halaman *AR Camera* “*Scan Window*”	OK
4	Tombol "*Back*"	Pengguna menekan tombol	Pengguna berada di halaman *AR Camera* “*Scan Window*”	Membuka halaman “Menu Utama”	OK
5	Memindai gambar markah rumah eksterior	Pengguna mengarahkan kamera AR ke gambar markah rumah eksterior	Pengguna berada di halaman *AR Camera* “*Scan Window*”	Menampilkan objek 3D rumah eksterior	OK
6	Memutar objek 3D rumah eksterior	Pengguna melakukan *gestur twist* pada layar	Objek 3D rumah eksterior tampil di layar	Objek 3D rumah eksterior berputar menurut sumbu Y	OK
7	Mengubah ukuran objek 3D rumah eksterior	Pengguna melakukan *gestur pinch* pada layar	Objek 3D rumah eksterior tampil di layar	Objek 3D rumah eksterior berubah ukuran	OK
8	Memindai gambar markah rumah interior	Pengguna mengarahkan kamera AR ke gambar markah rumah interior	Pengguna berada di halaman *AR Camera* “*Scan Window*”	Menampilkan objek 3D rumah interior	OK
9	Memutar objek 3D rumah interior	Pengguna melakukan *gestur twist* pada layar	Objek 3D rumah interior tampil di layar	Objek 3D rumah interior berputar menurut sumbu Y	OK
10	Mengubah ukuran objek 3D rumah interior	Pengguna melakukan *gestur pinch* pada layar	Objek 3D rumah interior tampil di layar	Objek 3D rumah interior berubah ukuran	OK
11	Memindai gambar markah rumah *siteplan*	Pengguna mengarahkan kamera AR ke gambar markah rumah *siteplan*	Pengguna berada di halaman *AR Camera* “*Scan Window*”	Menampilkan objek 3D rumah *siteplan*	OK
12	Memutar objek 3D rumah *siteplan*	Pengguna melakukan *gestur twist* pada layar	Objek 3D *siteplan* tampil di layar	Objek 3D rumah *siteplan* berputar menurut sumbu Y	OK
13	Mengubah ukuran objek 3D rumah *siteplan*	Pengguna melakukan *gestur pinch* pada layar	Objek 3D *siteplan* tampil di layar	Objek 3D rumah *siteplan* berubah ukuran	OK
14	Arah hadap anotasi *siteplan*	Pengguna menggerakkan kamera memutari *siteplan*	Objek 3D *siteplan* tampil di layar	Anotasi selalu menghadap kamera	OK
15	Membuka anotasi *sipeplan*	Pengguna menekan anotasi *siteplan*	Objek 3D *siteplan* tampil di layar	Anotasi menampilkan informasi yang tersedia	OK

No	Elemen	Deskripsi	Kondisi	Hasil yang diharapkan	Hasil Aktual
16	Menutup anotasi *sipeplan*	Pengguna menekan anotasi *siteplan*	Objek 3D *siteplan* tampil di layar dan anotasi *siteplan* terbuka	Anotasi menyembunyikan informasi tersedia	OK
17	Audio rekaman cinurawa	Pengguna melakukan pemindaian gambar markah	Pengguna berada di halaman *AR Camera* "*Scan Window*"	Audio rekaman cinurawa diputar	OK
18	Menampilkan tombol "*Virtual Tour*"	Pengguna melakukan pemindaian gambar markah	Pengguna berada di halaman *AR Camera* "*Scan Window*"	Tombol "*Virtual Tour*" ditampilkan	OK
19	Tombol "*Virtual Tour*"	Pengguna menekan tombol	Objek 3D rumah eksterior atau interior tampil di layar	Membuka situs web *virtual tour*	OK

B. Hasil Pengujian Pengaruh Aplikasi AR

Pengujian pengaruh dilakukan dengan mengajukan kuesioner ke pihak terkait. Pengisi kuesioner merupakan perwakilan dari 4 tim pengembang dan pemasaran properti perumahan yang beroperasi di daerah Tegal, Jawa Tengah. Penguji akan mencoba aplikasi AR yang telah dikembangkan dan mengisi sembilan pertanyaan pada kuesioner dengan menilai berdasarkan skala 1, yang berarti tidak membantu, sampai dengan 10, yang berarti sangat membantu. Hasil dari pengujian yang dilakukan oleh perwakilan dari empat pengembang properti perumahan dapat dilihat pada Tabel 2.

Tabel 2. Nilai rata-rata oleh pengembang perumahan

No. Pertanyaan	Fitur Yang Diuji	Nilai Rata - Rata
1	Halaman "Info Perumahan"	8.25
2	Fitur "Scan Rumah Eksterior"	7.75
3	Fitur "Scan Rumah Interior"	8.75
4	Fitur "Scan Peta Masterplan"	7
5	Fitur "*Scale*" model 3D	7.5
6	Fitur "*Rotate*" model 3D	7.25
7	Fitur "Anotasi" Masterplan	7.5
8	Fitur "*Virtual Tour* Rumah"	8.75
9	Teknologi *Augmented Reality* dalam meningkatkan pemasaran	8.5
	Total	7.92

Berdasarkan pengujian yang telah dilakukan oleh tim pengembang properti perumahan, didapatkan hasil rata - rata dari seluruh pertanyaan yang ada yaitu 7.92.

V. SIMPULAN

Aplikasi AR untuk membantu pemasaran properti perumahan telah berhasil dikembangkan dan telah dilakukan uji coba oleh beberapa pihak agen properti perumahan. Hasil yang dapat disimpulkan dari penelitian ini bahwa, aplikasi AR berhasil dikembangkan untuk membantu calon pembeli dan agen properti perumahan dalam melakukan transaksi properti. Berdasarkan survei yang telah dilakukan nilai rata-rata keseluruhan yang diberikan agen properti perumahan adalah 7.92. Fitur dari aplikasi AR yang paling penting dalam membantu transaksi properti perumahan merupakan fitur yang berhubungan dengan interior rumah. Berdasarkan survei, didapati pada fitur *scan* interior dan fitur *virtual tour* mendapatkan nilai rata-rata masing-masing 8.75 dari pihak agen perumahan. Fitur dari aplikasi AR yang perlu ditinjau ulang adalah fitur *scan* masterplan. Berdasarkan survei, 100% pihak agen perumahan memberikan nilai 7.

REFERENSI

[1] "Data Pertumbuhan Industri Properti di Indonesia, 2011 - 2022." https://www.dataindustri.com/produk/tren-data-pertumbuhan-industri-real-estate-properti/ (accessed Feb. 14, 2023).

[2] Fernando, Y. *et al.*, 'Penerapan Teknologi Augmented Reality Katalog Perumahan Sebagai Media Pemasaran Pada PT. San Esha Arthamas', *Jurnal Sains Komputer & Informatika (J-SAKTI*, 5(1), pp. 62–71, 2021.

[3] Ambre, T. P. *et al.*, 'Implementation of the 3d Digitalized Brochure using Marker-based Augmented Reality for Real Estates', *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*, pp. 483–487. doi: 10.1109/ICIRCA48905.2020.9183196, 2020.

[4] B. Afifah, T. Widiyaningtyas, and U. Pujianto, "Pengembangan bahan ajar perakitan komputer bermuatan augmented reality untuk menumbuhkan keaktifan belajar siswa," *Tekno*, vol. 29, no. 2, p. 97, 2019, doi: 10.17977/um034v29i2p97-115.

[5] L. Yeremia, D. Pangau, S. Tangkawarouw, G. Kaunang, and A. S. M. Lumenta, "Game Based Education : Pengenalan Peristiwa Sejarah Permesta di Minahasa," *J. Tek. Inform.*, vol. 14, no. 2, pp. 203–208, 2019, [Online]. Available: https://ejournal.unsrat.ac.id/v3/index.php/informatika/article/view/23995.

[6] M. Rais, "Penerapan Konsep Object Oriented Programming Untuk Aplikasi Pembuat Surat," *PROtek J. Ilm. Tek. Elektro*, vol. 6, no. 2, pp. 96–101, 2019, doi: 10.33387/protk.v6i2.1242.

[7] M. Fadya and I. P. Sari, "Modelling 3D dan Animating Karakter pada Game Edukasi 'World War D' Berbasis Android," *Multinetics*, vol. 4, no. 2, pp. 43–48, 2018, doi: 10.32722/multinetics.vol4.no.2.2018.pp.43-48.

[8] F. Jerga, "What Is The Unity Game Engine- All You Need To Know," *Eincode*, 2021. https://medium.com/eincode/what-is-the-unity-game-engine-all-you-need-to-know-d4ce77a1b7d2 (accessed Jun. 05, 2023).

[9] P. Inc, "Getting Started with Vuforia Engine in Unity," *Inc, PTC*. 2018, Accessed: Jun. 05, 2023. [Online]. Available: https://library.vuforia.com/getting-started/getting-started-vuforia-engine-unity#about-vuforia-engine.

[10] R. Roedavan, B. Pudjoatmodjo, and A. P. Sujana, "Multimedia Development Life Cycle (MDLC)," *Teknol. dan Inf.*, no. Multimedia, p. 7, 2022, doi: 10.13140/RG.2.2.16273.92006.

[11] D. W. T. Putra and R. Andriani, "Unified Modelling Language (UML) dalam Perancangan Sistem Informasi Permohonan Pembayaran Restitusi SPPD," *J. TeknoIf*, vol. 7, no. 1, p. 32, 2019, doi: 10.21063/jtif.2019.v7.1.32-39.

[12] A. Fahrezi, F. N. Salam, G. M. Ibrahim, and R. Rahman, "Pengujian Black Box Testing pada Aplikasi Inventori Barang Berbasis Web di PT . AINO Indonesia," *J. Ilmu Komput. dan Pendidik.*, vol. 1, no. 1, pp. 1–5, 2022.

Rancangan Pengalaman dan Antarmuka Pengguna Aplikasi Media Pembelajaran Siswa Tunagrahita dengan Pendekatan *Lean UX*

Taufik Kemal Thaha[1], Nanang Arifuddin[1], Margareta Hardiyanti[1,*]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
taufik.k.t@mail.ugm.ac.id
nanangarifudin@mail.ugm.ac.id
[*]Korespondensi: margareta.hardiyanti@ugm.ac.id;

Abstract – Reading skills play a pivotal role in advancing education, shaping the vision, character, and abilities of students, including those with intellectual disabilities who attend Special Schools. Several challenges are encountered by students in the fundamental reading process, such as reading letters, words, and sentences. Hence, leveraging mobile learning media, there is a need to design the user experience and interface of a learning media application to assist teachers in providing enjoyable reading material for students with intellectual disabilities throughout the learning process. This research was conducted with six teachers at SLB B-C YBA Surakarta, SLB B-C Panca Bakti Mulia Surakarta, and SLB Agca Center Surakarta. The Lean User Experience (Lean UX) method was employed with two iterations to maximize design solutions. The design process was executed using Figma, and usability testing was performed, quantifying success rates to gauge user design success. To evaluate satisfaction, the User Experience Questionnaire (UEQ) and System Usability Scale (SUS) were employed. The results from the success rate metric showed an 88% achievement in the first iteration, which increased to 94% in the second iteration. The SUS questionnaire results achieved a score of 80 in the first iteration and 86.25 in the second iteration, falling into the "excellent" category. Consequently, this design furnishes a positive user experience and interface that aligns with the requirements of teachers in supporting the fundamental reading learning process for students with intellectual disabilities.

Keywords – Intellectual Disabilities, Learning Media, Lean Ux, User Interface, Usability Testing

Intisari – Keterampilan membaca mendukung kemajuan pendidikan dalam membentuk visi, karakter, dan keterampilan bagi siswa, termasuk siswa tunagrahita yang menempuh pendidikan di Sekolah Luar Biasa (SLB). Terdapat beberapa kendala yang dihadapi siswa dalam proses dasar membaca seperti membaca huruf, kata, dan kalimat. Oleh karena itu, dengan memanfaatkan media pembelajaran *handphone*, diperlukan rancangan pengalaman dan antarmuka pengguna aplikasi media pembelajaran untuk membantu guru dalam memberikan materi membaca yang menyenangkan bagi siswa tunagrahita selama proses pembelajaran. Penelitian ini dilakukan kepada enam guru di SLB B-C YBA Surakarta, SLB B-C Panca Bakti Mulia Surakarta, dan SLB Agca Center Surakarta. Metode yang digunakan adalah Lean UX dengan dua kali proses iterasi agar dapat meningkatkan solusi desain dengan maksimal. Proses perancangan menggunakan *tools* Figma dan pengujian menggunakan *usability testing* dengan menghitung *success rate* untuk mengukur keberhasilan pengguna menggunakan rancangan. Sedangkan untuk mengukur kepuasan (*satisfication*) dengan menggunakan *User Experience Questionnaire* (UEQ) dan *System Usability Scale* (SUS). Hasil *success rate* pada iterasi pertama menunjukkan mencapai 88% dan iterasi kedua mencapai 94%. Kuesioner SUS mendapatkan hasil skor akhir 80 pada iterasi pertama dan skor 86,25 pada iterasi kedua, yang masuk ke dalam kriteria "*excellent.*" Sehingga rancangan ini memberikan pengalaman dan antarmuka pengguna yang baik dan sesuai dengan kebutuhan guru dalam mendukung proses pembelajaran membaca dasar siswa tunagrahita.

Kata kunci – Tunagrahita, Media Pembelajaran, *Lean Ux*, Antarmuka Pengguna, *Usability Testing*

I. PENDAHULUAN

Pendidikan memegang peran utama dalam membentuk visi, karakter, keterampilan, untuk mempersiapkan generasi mendatang dalam menghadapi tantangan yang terus berkembang. Salah satu upaya dalam mendukung kemajuan pendidikan adalah pengembangan keterampilan membaca siswa, termasuk siswa tunagrahita yang menempuh pendidikan di Sekolah Luar Biasa (SLB). Meningkatkan keterampilan membaca menjadi kunci untuk menguasai berbagai macam bidang studi dan menjadi langkah awal memasuki proses pembelajaran yang lebih kompleks bagi siswa tunagrahita [1].

Tunagrahita adalah retardasi mental atau indikasi adanya keterbelakangan mental dan hambatan intelektual [2, 3]. Situasi tersebut membuat siswa tunagrahita terkendala dalam meningkatkan keterampilan membaca.[4] menjelaskan bahwa dampak dari keadaan yang dialami siswa tunagrahita adalah belum mampu berpikir secara abstrak, sulit berkonsentrasi, terhambatnya kapasitas memori, dan terkadang mengalami hambatan dalam persepsi dibandingkan siswa pada umumnya.

Kemajuan teknologi di era modern ini melahirkan alternatif-alternatif yang dapat memberikan manfaat jika digunakan dengan tepat. Berbagai upaya telah dilakukan agar dapat mendukung keterampilan membaca siswa tunagrahita, salah satunya adalah dengan penggunaan aplikasi media pembelajaran *handphone*.

Namun, tidak semua aplikasi dapat dijadikan media pembelajaran untuk siswa tunagrahita. Sedangkan apabila proses pembelajaran tidak didukung dengan media yang memadai dapat menyebabkan siswa tunagrahita merasa jenuh dan penurunan perhatian terhadap proses pembelajaran [5]. Keterbatasan fokus dan konsentrasi siswa tunagrahita menjadi

tantangan yang dihadapi dalam merancang aplikasi media pembelajaran.

Untuk mengatasi permasalahan tersebut, diperlukan sebuah strategi yang tepat sasaran sehingga dapat membantu proses pembelajaran menjadi lebih menarik dan membuat siswa tunagrahita antusias [6]. Sehingga guru memegang peran yang krusial sebagai fasilitator dalam mendukung keterampilan membaca untuk siswa tunagrahita.

Oleh karena itu, diperlukan rancangan pengalaman dan antarmuka pengguna aplikasi media pembelajaran untuk membantu guru dalam memberikan materi membaca yang menyenangkan bagi siswa tunagrahita selama proses pembelajaran. Melalui implementasi desain pengalaman pengguna yang sesuai kebutuhan, pengguna dapat menggunakan rancangan tersebut untuk mendukung aktivitas mereka dengan optimal [7]. Sehingga rancangan ini bisa menjadi solusi yang intuitif dan interaktif dalam membantu guru untuk mendukung proses pembelajaran membaca dasar siswa tunagrahita di sekolah luar biasa.

II. DASAR TEORI

A. Tunagrahita

Definisi tunagrahita adalah kondisi di mana seseorang mengalami hambatan dalam aspek kecerdasan dan kemampuan intelektual yang lebih rendah daripada umumnya [8]. Kelompok tunagrahita dikelompokkan pada Tabel 1.

Tabel 1. Rentang IQ tunagrahita [9].

Kategori	Rentang IQ
Tunagrahita ringan	70-55
Tunagrahita sedang	55-40
Tunagrahita berat	40-25
Tunagrahita sangat berat	< 25

Berdasarkan hasil wawancara yang telah dilakukan kepada guru SLB, perancangan aplikasi media pembelajaran difokuskan untuk siswa tunagrahita ringan.

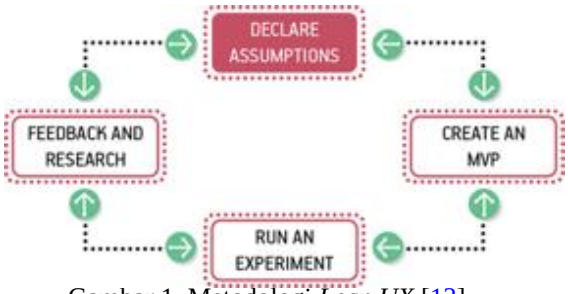
B. *User Interface/User Experience* (*UI/UX*)

Tampilan yang berfungsi sebagai jembatan dan tempat berinteraksinya pengguna dan sistem disebut juga antarmuka pengguna atau *user interface* [10]. Sedangkan pengalaman pengguna yang mencakup sisi emosional, lingkungan, dan psikologi disebut juga pengalaman pengguna atau *user experience* [11].

C. *Lean UX*

Sebuah metode untuk memberikan konteks aktual dari sebuah sistem untuk mencapai tingkat keakuratan yang baik dalam waktu singkat disebut juga *Lean UX* [12]. Metodologi *Lean UX* pada Gambar 1 terdiri dari beberapa tahapan di antaranya:

1. *Declare Assumption*, dalam tahap ini tim membuat rincian masalah yang dihadapi pengguna, membuat hipotesis tentang solusi yang mungkin, membuat proto persona sebagai representasi pengguna, melakukan analisis tugas, dan merancang secara kolaboratif dalam bentuk prototipe beresolusi rendah.
2. *Create a Minimum Viable Product* (MVP), yang merupakan hasil yang menggambarkan hipotesis mereka dalam tingkat resolusi yang lebih tinggi.
3. *Run an Experiment*, menjalankan eksperimen pada MVP yang telah dibuat. Eksperimen ini dilakukan secara independen dan melibatkan *stakeholders*.
4. *Feedback and Research*, pada tahap ini, MVP divalidasi dengan melibatkan pengguna untuk pengujian usabilitas dan dilakukan analisis perilaku.



Gambar 1. Metodologi *Lean UX* [13]

D. *Affinity Diagram*

Pengelompokan ide-ide yang sudah terorganisir disajikan dalam metode visual disebut juga *Affinity Diagram* [14].

E. *Userflow*

Userflow merupakan Alur dari sekumpulan langkah atau tugas yang dilakukan pengguna secara bertahap dari awal hingga akhir saat berinteraksi dengan sebuah aplikasi atau situs web [15].

F. *User Persona*

Representasi karakter atau profil target pengguna yang akan menggunakan suatu produk disebut juga *User Persona* [13].

G. *Sitemap*

Sitemap adalah struktur navigasi yang menggambarkan hierarki halaman secara keseluruhan pada suatu situs web atau aplikasi.

H. Prototipe

Mockup visual yang dapat digunakan pengguna untuk menjadi representasi dari solusi desain yang ditawarkan secara menyeluruh disebut prototipe [16]. Prototipe sendiri terdiri dari *Low-Fidelity Prototype* dan *High-Fidelity Prototype*. *Low-Fidelity Prototype* disebut juga sebagai kerangka sketsa dasar dari sebuah aplikasi [17, 13]. Sedangkan *High-Fidelity Prototype* adalah visualisasi elemen desain yang lebih lengkap dari *Low-Fidelity Prototype*.

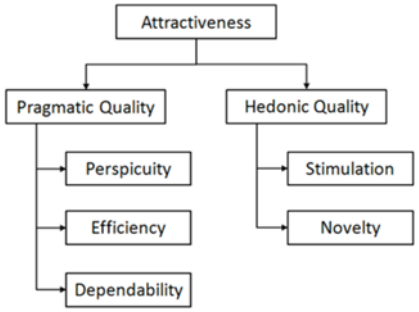
I. *Usability Testing*

Usability Testing adalah pengujian langsung kepada pengguna untuk menilai bagaimana pengguna berinteraksi

dengan produk yang telah dibuat [11]. Salah satu metode paling sederhana untuk mengukur keberhasilan pengguna adalah dengan pengukuran *success rate* [18].

J. *User Experience Questionnaire* (UEQ)

UEQ merupakan analisis objektif menggunakan kuesioner yang bertujuan untuk mengukur secara cepat pengalaman pengguna dalam menggunakan sebuah produk. Aspek yang dikur dalam UEQ dijelaskan pada Gambar 2.



Gambar 2. Skala Pengukuran UEQ [19]

K. *System Usability Scale* (SUS)

SUS merupakan pengukuran *usability* suatu sistem yang berdasarkan perspektif pengguna dalam menggunakan rancangan secara keseluruhan [20]. Pertanyaan SUS disajikan melalui Tabel 2.

Tabel 2. Kuesioner SUS Indonesia

Kode	Pertanyaan Kuesioner
P1	Saya berpikir akan menggunakan sistem ini lagi
P2	Saya merasa sistem ini rumit untuk digunakan
P3	Saya merasa sistem ini mudah untuk digunakan
P4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini
P5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya
P6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi) pada sistem ini
P7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat
P8	Saya merasa sistem ini membingungkan
P9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini
P10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini

III. METODOLOGI

Alur penelitian seperti pada Gambar 3 dimulai dari melakukan studi literatur untuk mengumpulkan referensi yang relevan sesuai cakupan penelitian yang dilanjutkan dengan pengisian kuesioner dan wawancara kepada perwakilan guru. Iterasi pertama dimulai dari metode *Lean UX* yang terdiri dari:

A. *Declare Assumptions*

1. *Problem Statement*: Menggambarkan perspektif pengguna (guru SLB) terhadap permasalahan, dipetakan menggunakan *Affinity Diagram*.
2. *Hypothesis*: Perumusan fitur sebagai hipotesis dari asumsi yang dipetakan untuk pengembangan antarmuka.
3. *User Persona*: Representasi fiktif dari target pengguna untuk memahami karakteristik dan kebutuhan mereka.
4. *Task Analysis*: Pembuatan alur sistem dan navigasi menggunakan *userflow* dan *sitemap* untuk mendefinisikan aktivitas pengguna.
5. *Collaborative Design*: Sketsa prototipe *low-fidelity* dari alur dan navigasi yang dirancang.

B. *Create a Minimum Viable Product (MVP)*

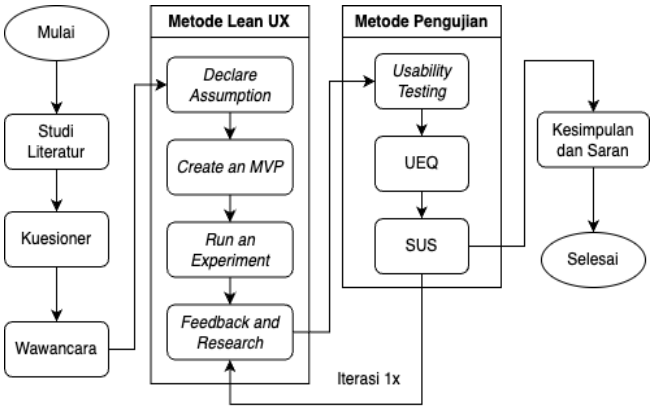
1. *Design System*: Pembuatan komponen desain untuk konsistensi antarmuka.
2. *High-Fidelity Design*: Desain antarmuka beresolusi tinggi menggunakan komponen dari *Design System*.
3. Prototipe: Fase pembuatan antarmuka interaktif yang dapat diuji langsung oleh pengguna.

C. *Run an Experiment*

Melakukan uji coba pada prototipe dengan *usability testing* untuk menghitung *success rate* dari tugas pengguna.

D. *Feedback and Research*

Mengumpulkan kritik, saran, dan evaluasi alur desain antarmuka menggunakan SUS dan UEQ setelah uji coba prototipe seperti yang ditunjukkan pada Gambar 3 sebagai berikut.



Gambar 3. Alur penelitian

Setelah itu, dilakukan pengujian melalui *Usability Testing* dan mengukur keberhasilan desain menggunakan *System Usability Scale* (SUS) dan *User Experience Questionnaire* (UEQ) dan dilanjutkan dengan proses iterasi kedua dengan langkah yang serupa untuk meningkatkan solusi desain yang digunakan.

IV. HASIL DAN PEMBAHASAN

A. Hasil Iterasi Pertama

1. Lean UX

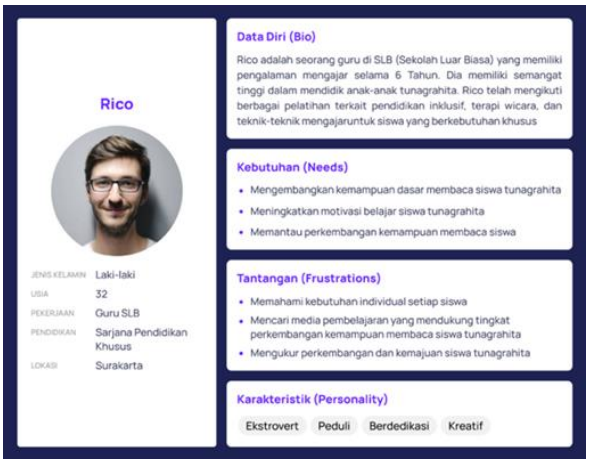
a. *Declare Assumption*

Pada tahap ini, seluruh data hasil jawaban saat kuesioner wawancara akan dikelompokkan dan disortir menjadi beberapa bagian sehingga menghasilkan solusi desain yang mendukung proses pembelajaran berdasarkan *Affinity Diagram* pada Tabel 3.

Tabel 3. *Affinity diagram* Iterasi pertama

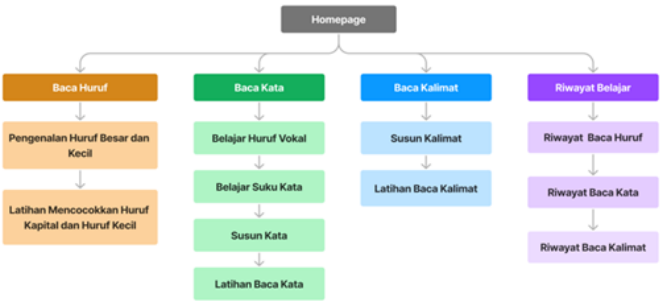
Aspek	Temuan
Introduction questions	Jumlah siswa per kelas: Maksimal: 8 Orang dan rata-rata: 5 Orang
	Siswa SLB SD, SMP, dan SMA masih sesuai dengan usia siswa di sekolah umum
Looking for pain points	Daya ingat siswa tunagrahita yang terbatas
	Guru perlu mengulangi materi yang diberikan
	Libur membuat siswa tunagrahita lebih sulit mengingat materi
	Guru kesulitan memantau proses belajar
	Aplikasi yang tersedia belum mencakup kebutuhan materi siswa tunagrahita
Understanding user's behavior	Terdapat beberapa kegiatan ekstrakurikuler yang bertujuan untuk mendukung kemampuan aspek sosial siswa tunagrahita
	Siswa sudah terbiasa menggunakan *handphone*
	Siswa tunagrahita terbiasa dengan fitur *voice note* dibanding mengetik
To Solve the solution	Fitur utama yang terdiri dari baca huruf, baca kata, dan baca kalimat
	Terdapat fitur untuk memantau progres belajar siswa
Get new insights from user	Penyusunan materi memperhatikan kurikulum dan kompetensi dasar
	Rancangan aplikasi menggunakan navigasi suara dan ilustrasi yang menarik

Selanjutnya adalah perancangan *user persona* guru yang dijelaskan melalui Gambar 4.

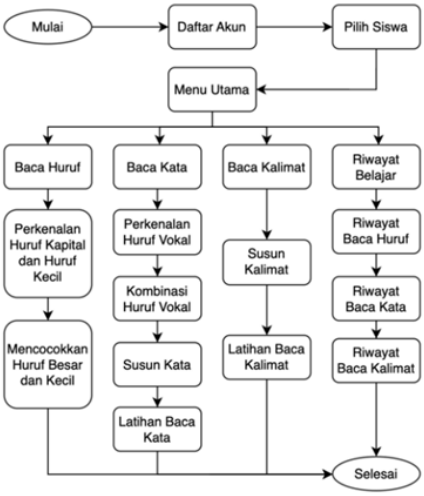


Gambar 4. *User Persona* Guru

Setelah proses pembuatan *user persona*, selanjutnya adalah perancangan *sitemap* pada Gambar 5 dan *userflow* pada Gambar 6.

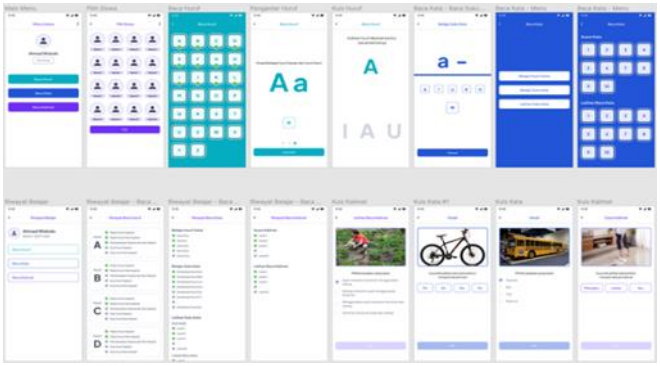


Gambar 5. *Sitemap* rancangan aplikasi



Gambar 6. *Useflow* rancangan aplikasi

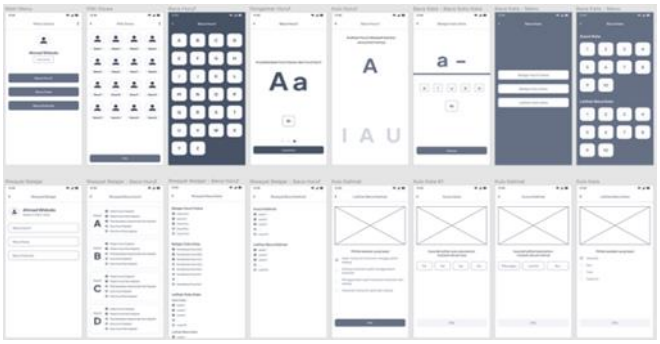
Selanjutnya pada Gambar 7 yaitu perancangan *low-fidelity prototype* menjadi yang sudah dijabarkan pada *sitemap* dan *userflow*.



Gambar 7. *Lo-fi prototype* fitur utama

b. *Create a Minimum Viable Product (MVP)*

Langkah selanjutnya adalah membuat *hi-fidelity prototype* fitur utama pada Gambar 8.



Gambar 8. *Hi-fi prototype* fitur utama

c. *Run an Experiment.*

Tahapan adalah pengujian *usability* dengan mengukur *success rate* dari prototipe. Daftar tugas pengguna disajikan pada Tabel 4.

Tabel 4. Daftar tugas *usability testing*

Kode	Kategori	Tugas
T1	Akun	Mendaftarkan akun
T2		Masuk akun
T3		Mengubah kata sandi
T4		Keluar akun
T5	Profil Siswa	Menambah profil siswa
T6		Mengedit profil siswa
T7		Menghapus profil siswa
T8	Baca Huruf	Membaca huruf kapital
T9		Membaca huruf kecil
T10		Membedakan huruf kapital dan kecil
T11		Mencocokkan huruf kapital
T12		Mencocokkan huruf kecil
T13	Baca Kata	Membaca huruf vokal
T14		Membaca suku kata
T15		Menyusun suku kata menjadi kata
T16		Mencocokkan kata dari ilustrasi
T17	Baca Kalimat	Menyusun kalimat dari kata
T18		Mencocokkan kalimat dari ilustrasi
T19	Riwayat Belajar	Melihat riwayat baca huruf
T20		Melihat riwayat baca kata
T21		Melihat riwayat baca kalimat

Jumlah responden dari pengujian iterasi pertama sebanyak enam orang guru SLB. Hasil dari pengujian *success rate* pada iterasi pertama disajikan pada Tabel 5.

Tabel 5. Hasil *success rate* Iterasi pertama

Kode	R1	R2	R3	R4	R5	R6
T1	SB	SB	SB	SB	SB	SB
T2	S	S	S	S	S	S
T3	S	S	S	S	S	S
T4	S	S	S	S	S	S
T5	S	S	S	G	S	S
T6	S	SB	S	SB	S	S
T7	S	S	S	S	S	S
T8	S	S	S	S	S	S
T9	S	S	S	S	S	S
T10	S	S	S	S	S	S
T11	G	G	G	G	S	S
T12	S	S	S	S	S	S
T13	S	S	S	S	S	S
T14	SB	S	SB	G	S	SB
T15	SB	S	S	S	S	S
T16	S	S	S	S	SB	S
T17	S	S	S	S	S	SB
T18	SB	S	S	S	S	S
T19	S	S	S	S	S	S
T20	S	S	S	SB	S	S
T21	S	S	S	S	S	S

S = Sukses, SB = Sebagian Berhasil, G = Gagal

Perhitungan *success rate* disajikan pada Perhitungan 1.

$$Success\ Rate = \frac{S + (SB \times 0{,}5)}{21\ x\ 6\ responden}\ x\ 100\% \quad (1)$$

$$Success\ Rate = \frac{104 + (16 \times 0{,}5)}{126}\ x\ 100\%$$

$$Success\ Rate = 88\%$$

Hasil tersebut menandakan bahwa rancangan secara keseluruhan tugas sudah bisa diselesaikan oleh pengguna.

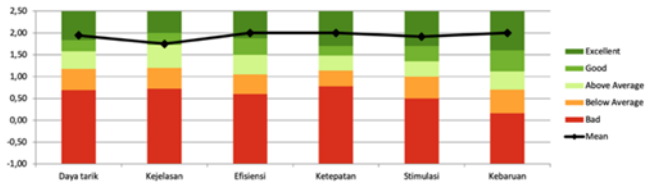
d. *Feedback and Research.*

Setelah mengukur *success rate*, dilanjutkan dengan mengumpulkan kritik dan saran dari responden pada Tabel 6.

Tabel 6. Kritik dan saran Iterasi pertama

No	Kritik dan Saran
1	Tidak ada perbedaan yang jelas antara daftar akun dan masuk akun saat membuka aplikasi
2	Tombol *speaker* yang sulit diidentifikasi apakah dalam keadaan aktif atau nonaktif
3	Fitur mencocokkan huruf tidak ada arahan untuk menggeser huruf sesuai bentuk pada soal
4	Tidak membuat huruf vokal masing-masing dikombinasikan dengan huruf yang dipilih, melainkan langsung menggabungkan huruf vokal. Contoh, apabila pengguna memilih huruf B, maka langsung diberikan contoh pengucapan "Ba", "Bi", "Bu", "Be", dan "Bo"
5	Terdapat beberapa ilustrasi dan soal yang sulit dimengerti untuk siswa tunagrahita pada alur latihan baca kata dan baca kalimat
6	Semua soal pilihan ganda diubah hanya menjadi tiga opsi (a, b, dan c), hal ini dikarenakan siswa tunagrahita tidak terbiasa menggunakan empat opsi
7	Masih terdapat banyak ruang kosong yang belum dimaksimalkan pada rancangan antarmuka
8	Semua opsi jawaban pada alur Latihan Baca Kalimat memiliki struktur kalimat teracak yang membuat siswa tunagrahita bingung mendapatkan konteks pertanyaan yang dimaksud

2. Setelah memberikan kritik dan saran, selanjutnya adalah pengisian kuesioner UEQ. Hasil dari 26 butir kuesioner dikelompokkan menjadi enam skala UX yang disajikan Gambar 9.



Gambar 9. Hasil UEQ Iterasi pertama

Pada gambar tersebut dapat dilihat bahwa hanya aspek Kejelasan mendapatkan nilai *mean* 1,75 dan kriteria hasil "*good*" sedangkan aspek Daya Tarik (1,94), Efisiensi (2,00), Ketepatan (2,00), Stimulasi (1,92) dan Kebaruan (2,00) yang masing-masing mendapatkan kriteria "*excellent*".

3. Kemudian dilanjutkan dengan pengukuran menggunakan SUS yang disajikan melalui Tabel 7.

Tabel 7. Skor SUS Iterasi pertama

P/R	R1	R2	R3	R4	R5	R6
P1	5	4	4	5	5	4
P2	1	1	1	1	1	2
P3	5	5	5	5	5	4
P4	1	4	1	3	3	4
P5	5	4	5	4	4	4
P6	1	1	1	2	2	3
P7	5	5	5	5	5	5
P8	1	1	1	1	1	2
P9	5	4	4	4	4	4
P10	5	5	4	4	4	4
Skor SUS	90	75	87,5	82,5	80	65

Untuk menghitung skor akhir SUS, dapat dilakukan dengan menghitung rata-rata skor sesuai Perhitungan 2 di bawah ini.

$$\text{Skor SUS} = \frac{\text{Jumlah Skor SUS}}{\text{Jumlah Responden}} = \frac{480}{6} \quad (2)$$

$$\text{Skor SUS} = 80$$

Skor 80 dalam SUS masuk kriteria "*excellent*" [5].

B. Hasil Iterasi Kedua

1. Lean UX

a. *Declare Assumption*

Pada tahap ini, dilakukan peninjauan kembali iterasi pertama beserta solusi rancangan dengan menggunakan *Affinity Diagram* pada Tabel 8

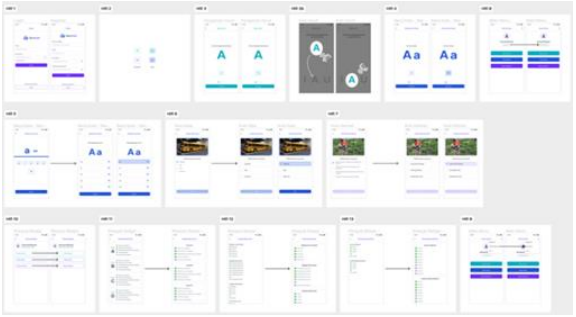
Tabel 8. *Affinity Diagram* Iterasi Kedua

Masalah Utama	Solusi Rancangan
Responden sulit membedakan antara masuk akun dan daftar akun. Hal ini dikarenakan alur pertama aplikasi saat dibuka adalah Masuk Akun.	Menukar alur antara masuk akun dan daftar akun, dan membuat desain tombol primer dan tombol sekunder sehingga lebih mudah diidentifikasi oleh pengguna.

Masalah Utama	Solusi Rancangan
Interaksi tombol *speaker* memiliki status yang kurang jelas apakah dalam keadaan aktif atau nonaktif	Pembuatan variasi *component* tombol apabila dalam keadaan aktif dan keadaan nonaktif.
Pengguna tidak mengetahui apa yang harus dilakukan saat antarmuka mencocokkan huruf yang ditampilkan	Perlu navigasi baik video demo ataupun petunjuk untuk mengarahkan pengguna mencocokkan huruf yang ditampilkan
Pengguna tidak mengetahui apa yang harus dilakukan saat melihat antarmuka kombinasi huruf vokal pada baca materi belajar suku kata di menu baca kata	Menggabungkan langsung huruf vokal dan huruf yang sudah dipilih. Contoh, apabila pengguna memilih huruf B, maka langsung diberikan contoh pengucapan "Ba", "Bi", "Bu", "Be", dan "Bo"
Ilustrasi yang kompleks dan soal yang tidak familiar dengan siswa tunagrahita	Mengganti dengan ilustrasi yang lebih mudah dikenal dan dekat dengan kehidupan sehari-hari siswa tunagrahita
Siswa tunagrahita tidak terbiasa dengan opsi jawaban terlalu banyak membuat siswa tunagrahita sulit mencerna soal yang ditanyakan	Semua soal pilihan ganda disesuaikan menjadi hanya tiga opsi saja (a, b, dan c)
Elemen dan komponen desain yang masih kurang memanfaatkan ruang kosong	Melakukan penyesuaian ukuran *font*, dan aset-aset media untuk setiap elemen desain yang ada pada rancangan
Kemampuan membaca yang terbatas pada siswa tunagrahita mempersulit mereka dalam mengerjakan soal latihan baca kata yang terlalu panjang	Seluruh opsi jawaban latihan baca kalimat, dibuat lebih ringkas yang hanya terdiri dari Subjek + Predikat + Objek. Contoh: "Ayah siram bunga"
Ilustrasi yang kompleks dan soal yang tidak familiar dengan siswa tunagrahita	Mengganti dengan ilustrasi yang lebih mudah dikenal dan dekat dengan kehidupan sehari-hari siswa tunagrahita
Siswa tunagrahita tidak terbiasa dengan opsi jawaban terlalu banyak membuat siswa tunagrahita sulit mencerna soal yang ditanyakan	Semua soal pilihan ganda disesuaikan menjadi hanya tiga opsi saja (a, b, dan c)
Elemen dan komponen desain yang masih kurang memanfaatkan ruang kosong	Melakukan penyesuaian ukuran *font*, dan aset-aset media untuk setiap elemen desain yang ada pada rancangan
Kemampuan membaca yang terbatas pada siswa tunagrahita mempersulit mereka dalam mengerjakan soal latihan baca kata yang terlalu panjang	Seluruh opsi jawaban latihan baca kalimat, dibuat lebih ringkas yang hanya terdiri dari Subjek + Predikat + Objek. Contoh: "Ayah siram bunga"

b. *Create a Minimum Viable Product (MVP)*

Pada tahap ini merupakan tahap membuat rancangan desain berdasarkan solusi rancangan yang ditampilkan pada Gambar 10.



Gambar 10. Perbaikan desain Iterasi kedua

c. *Run an Experiment.*

Selanjutnya adalah dengan melakukan pengujian *usability testing* dengan menghitung *success rate* pada Tabel 9

Tabel 9. Hasil *success rate* Iterasi kedua

Kode	R1	R2	R3	R4	R5	R6
T1	S	S	S	S	S	S
T2	S	S	S	S	S	S
T3	S	S	S	S	S	S
T4	S	S	S	S	S	S
T5	S	S	S	S	SB	S
T6	S	S	S	S	S	S
T7	S	S	S	S	S	SB
T8	S	S	S	S	S	S
T9	S	S	S	S	S	S
T10	S	S	S	S	S	S
T11	SB	S	SB	S	SB	S
T12	S	SB	S	S	S	S
T13	S	S	S	S	S	S
T14	S	S	S	S	S	S
T15	G	G	SB	S	SB	S
T16	S	S	S	S	S	S
T17	S	S	S	S	S	S
T18	S	S	S	S	S	S
T19	S	S	S	S	S	S
T20	S	S	S	SB	S	S
T21	S	S	S	S	S	S

S = Sukses, SB = Sebagian Berhasil, G = Gagal

Perhitungan *success rate* disajikan pada Perhitungan 3.

$$Success\ Rate = \frac{S + (SB \times 0{,}5)}{21\ x\ 6\ responden}\ x\ 100\% \quad (3)$$

$$Success\ Rate = \frac{115 + (9 \times 0{,}5)}{126}\ x\ 100\%$$

$$Success\ Rate = 94\%$$

Hasil akhir *success rate* sebesar 94% yang artinya seluruh pengguna hampir menyelesaikan semua tugas yang diberikan.

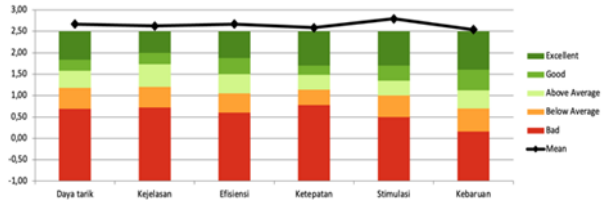
d. *Feedback and Research.*

Setelah mengukur *success rate*, dilanjutkan dengan menampung kritik dan saran dari responden pada Tabel 10.

Tabel 10. Kritik dan saran Iterasi kedua

No	Kritik dan Saran
1	Penambahan bendahara kata dan variasi soal
2	Penambahan materi membaca tingkat lanjutan

2. Setelah memberikan kritik dan saran, selanjutnya adalah pengisian kuesioner UEQ. Hasil dari 26 butir kuesioner dikelompokkan menjadi enam skala UX yang disajikan Gambar 11.



Gambar 11. Hasil UEQ Iterasi kedua

Hasil pengukuran UEQ pada iterasi kedua berdasarkan skala UX di atas menghasilkan skor Daya Tarik (2,67), Kejelasan (2,63), Efisiensi (2,67), Ketepatan (2,58), Stimulasi (2,79), dan Kebaruan (2,64). Gambar di atas juga menjelaskan bahwa keenam skala UX berwarna hijau gelap yang menandakan kriteria "*excellent*".

3. Kemudian dilanjutkan dengan pengukuran keseluruhan rancangan antarmuka dan pengalaman pengguna dengan menggunakan SUS yang disajikan melalui Tabel 11.

Tabel 11. Skor SUS Iterasi Kedua

P/R	R1	R2	R3	R4	R5	R6
P1	5	4	4	5	4	5
P2	1	1	1	1	1	1
P3	5	5	5	5	5	5
P4	1	2	2	4	1	2
P5	5	4	4	5	6	5
P6	1	1	2	1	1	1
P7	5	3	4	5	3	4
P8	1	1	1	1	1	1
P9	5	5	4	5	5	4
P10	4	4	4	2	3	2
Skor SUS	92,5	80	77,5	90	87,5	90

Untuk menghitung skor akhir SUS, dapat dilakukan dengan menghitung rata-rata skor seperti yang ditunjukkan pada Perhitungan 4.

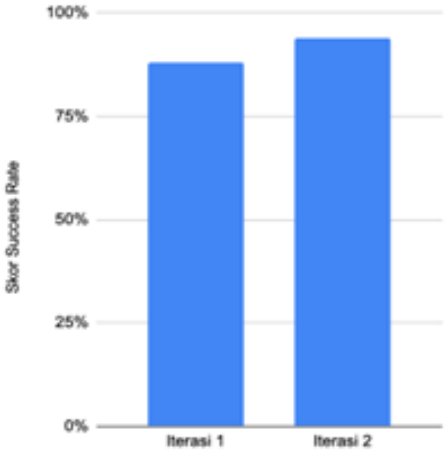
$$\text{Skor SUS} = \frac{\text{Jumlah Skor SUS}}{\text{Jumlah Responden}} = \frac{517{,}5}{6} \quad (4)$$

$$\text{Skor SUS} = 86{,}25$$

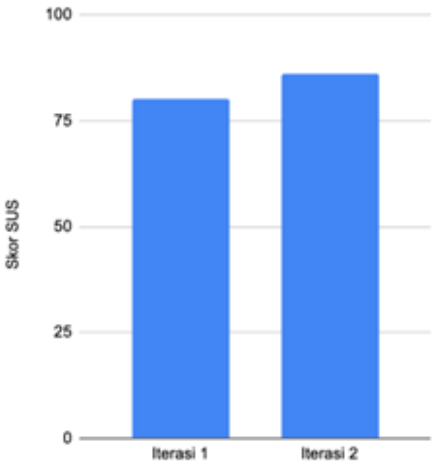
Skor 86,25 dalam SUS masuk kriteria "*excellent*" [5].

C. Perbandingan Antar Iterasi

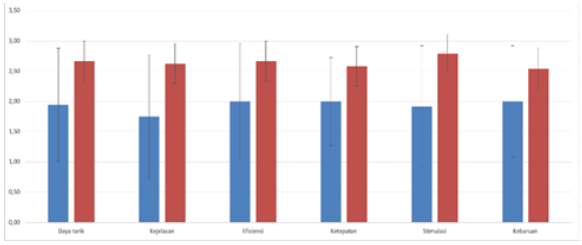
Berdasarkan hasil dari kedua iterasi, maka proses selanjutnya adalah membandingkan kedua hasil terasi yang disajikan melalui Gambar 12, Gambar 13, dan Gambar 14.



Gambar 12. Perbandingan *success rate*



Gambar 13. Perbandingan SUS



Gambar 14. Perbandingan UEQ

Berdasarkan keterangan pada Gambar 12, Gambar 13, dan Gambar 14 menunjukkan bahwa seluruh rancangan pada iterasi kedua memiliki hasil yang lebih baik diukur dari keenam aspek skala UX. Aspek *perspicuity* (Kejelasan) menjadi aspek yang mengalami peningkatan paling signifikan dibanding aspek lainnya dengan selisih 0,88 dan aspek Stimulasi memiliki nilai tertinggi berdasarkan pengukuran UEQ.

V. SIMPULAN

Berdasarkan hasil dan pembahasan pada perancangan aplikasi media pembelajaran tunagrahita, dapat disimpulkan bahwa, perancangan antarmuka menggunakan metode *Lean UX* yang terdiri dari *Declare Assumption*, *create an MVP*, *Run an Experiment*, dan *Feedback and Research*. Proses perancangan dilakukan dengan dua kali iterasi yang menghasilkan rancangan prototipe desain aplikasi media pembelajaran. Pengukuran *success rate* pada iterasi pertama adalah 88% dan iterasi kedua sebesar 94% menunjukkan bahwa antarmuka dan pengalaman pengguna yang telah dirancang, efisien dalam membantu proses pembelajaran guru dan siswa tunagrahita. Hasil kuesioner Hasil kuesioner UEQ menunjukkan kenaikan nilai *mean* antar iterasi dari enam skala UX yaitu *attractiveness* (daya tarik) meningkat 0,73, *perspicuity* (kejelasan) meningkat 0,88, *efficiency* (efisiensi) meningkat 0,67, *dependability* (ketepatan) meningkat 0,58, *stimulation* (stimulasi) meningkat 0,87, dan *novelty* (kebaruan) meningkat 0,54. Berdasarkan *benchmark* yang digunakan, semua aspek tersebut termasuk dalam kriteria "*excellent*" dan aspek *stimulation* memiliki nilai tertinggi. Hasil kuesioner SUS pada iterasi pertama mendapatkan skor 80 dan terjadi kenaikan pada iterasi kedua dengan skor akhir 86,25. Kedua skor tersebut masuk ke dalam kriteria *excellent*. Sehingga dapat dikatakan bahwa rancangan prototipe meningkatkan kepuasan pengguna dan memiliki pengalaman pengguna yang baik dalam membantu guru memberikan materi membaca dasar kepada siswa tunagrahita.

REFERENSI

[1] P. Fauziah, "Penggunaan Multimedia Interaktif Cerdas Belajar Baca dalam Meningkatkan Kemampuan Membaca Permulaan pada Anak Tunagrahita Ringan (Studi Eksperimen Dengan Desain Single Subject Research Terhadap Siswa Tunagrahita Ringan Kelas III SDLB C Cinta Asih)," *Jurnal UNIK: Pendidikan Luar Biasa*, vol. 1, no. 1, 2016.

[2] A. Meria, "Model Pembelajaran Agama Islam Bagi Anak Tunagrahita di SDLB YPPLB Padang Sumatera Barat," *Tsaqafah: Jurnal Peradaban Islam*, vol. 11, no. 2, 2015.

[3] A. Yanni, I. Kamala, M. S. Assingkily, and R. Rahmawati, "Analisis kemampuan intelektual anak tunagrahita ringan di SD Negeri Demakijo 2," *Jurnal Pendidikan*, vol. 21, no. 1, pp. 64–75, 2020.

[4] P. Puput and S. Tjutju, "Metode VAKT untuk pembelajaran membaca permulaan anak tunagrahita ringan," *JASSI ANAKKU*, vol. 18, no. 1, pp. 25–31, 2018.

[5] A. Rahmana and F. Y. Al Irsyadi, "Perancangan Alat Permainan Edukatif Berbasis Aplikasi Sebagai Media Pengenalan Alat Musik Untuk Anak Tunagrahita di SLB Negeri Sukoharjo," Doctoral dissertation, Universitas Muhammadiyah Surakarta, 2023.

[6] V. H. Saputra, D. Darwis, and E. Febrianto, "Rancang bangun aplikasi game matematika untuk penyandang tunagrahita berbasis mobile," *Jurnal Komputer Dan Informatika*, vol. 15, no. 1, pp. 171–181, 2020.

[7] Y. H. Wijaya, L. Liliana, and L. E. Sari, "Penerapan Desain User Experience Pada Aplikasi Penghitungan Matematika Bagi Anak Penyandang Tunagrahita di Quali International Surabaya," *Jurnal Infra*, vol. 9, no. 2, pp. 192–198, 2021.

[8] P. Puput and S. Tjutju, "Metode VAKT untuk pembelajaran membaca permulaan anak tunagrahita ringan," *JASSI ANAKKU*, vol. 18, no. 1, pp. 25–31, 2018.

[9] D. R. Desiningrum, *Psikologi anak berkebutuhan khusus*, 2017.

[10] G. Goel, P. Tanwar, and S. Sharma, "UI-UX Design Using User Centred Design (UCD) Method," in *2022 International Conference on Computer Communication and Informatics (ICCCI)*, 2022, pp. 1–8.

[11] A. Cooper, R. Reimann, D. Cronin, and C. Noessel, *About Face: The Essentials of Interaction Design*, 2014.

[12] M. A. Insani, M. A. Gustalika, and I. Kresna, "Prototype Desain User Interface Aplikasi My School Menggunakan Metode Lean UX," *Journal of Information System Research (JOSH)*, vol. 3, no. 4, pp. 626–635, 2022.

[13] N. Hidayah, Z. Rafiiuddin, A. Durachman, Y. Rustamaji, and E. Rustamaji, "User Experience Design Analysis Using Lean UX Method," in *2021 9th International Conference on Cyber and IT Service Management (CITSM)*, 2021.

[14] M. A. D. Pratama, Y. R. Ramadhan, and T. I. Hermanto, "Rancangan UI/UX Design Aplikasi Pembelajaran Bahasa Jepang Pada Sekolah Menengah Atas Menggunakan Metode Design Thinking," *JURIKOM (Jurnal Riset Komputer)*, vol. 9, no. 4, pp. 980–987, 2022.

[15] M. F. Aziz, Harlili, and D. P. Satya, "Designing Human-Computer Interaction for E-Learning using ISO 9241-210:2010 and Google Design Sprint," in *2020 7th International Conference on Advance Informatics: Concepts, Theory and Applications (ICAICTA)*, 2020, pp. 1–6.

[16] M. F. Ardiansyah and P. Rosyani, "Perancangan UI/UX Aplikasi Pengolahan Limbah Anorganik Menggunakan Metode Design Thinking," *LOGIC: Jurnal Ilmu Komputer dan Pendidikan*, vol. 1, no. 4, pp. 839–853, 2023.

[17] T. Zhang, P. L. P. Rau, G. Salvendy, and J. Zhou, "Comparing Low and High-Fidelity Prototypes in Mobile Phone Evaluation," *International Journal of Technology Diffusion (IJTD)*, vol. 3, no. 4, pp. 1–19, 2012.

[18] M. A. Mayasari and M. A. Kresna, "Penerapan Metode LEAN UX Pada Perancangan UI/UX Aplikasi Digilib Unsika Versi Windows," *INTECOMS: Journal of Information Technology and Computer Science*, vol. 4, no. 2, pp. 392–405, 2021.

[19] M. Schrepp, J. Thomaschewski, and A. Hinderks, "Construction of a benchmark for the user experience questionnaire (UEQ)," 2017.

[20] J. Brooke, "SUS: a 'quick and dirty' usability," *Usability Evaluation in Industry*, vol. 189, no. 3, pp. 189–194, 1995.

Rancang Bangun Aplikasi Media Pembelajaran Berbasis Android untuk Siswa Tunagrahita Sekolah Luar Biasa

Nanang Arifudin[1], Taufik Kemal Thaha [1], Margareta Hardiyanti[1,*], Sinung Suakanto[2]
[1]Departemen Teknik Elektro dan Informatika, Sekolah Vokasi, Universitas Gadjah Mada;
nanangarifudin@mail.ugm.ac.id
taufik.k.t@mail.ugm.ac.id
[2]Program Studi Sistem Informasi, Fakultas Rekayasa Industri, Universitas Telkom;
sinung@telkomuniversity.ac.id
[*]Korespondensi: margareta.hardiyanti@ugm.ac.id;

Abstract – Equal access to education is a fundamental right for all citizens of Indonesia, including children with special needs. Special Schools (SLB) in Indonesia provide educational services for students with various special needs, including those with intellectual disabilities, who exhibit diverse IQ levels that affect their learning abilities, particularly in reading. Conventional reading teaching methods are still widely used in SLBs, requiring a high level of patience from teachers and heavily relying on their capabilities. To address these challenges, an Android-based reading learning media application has been developed as an alternative teaching tool. The application includes reading materials, exercises, audio, images, student management features, and learning history records. The development process involves user needs analysis, design, implementation, and testing phases. Kotlin programming language was utilized for application development, with Firebase serving as the database solution. Testing results show that the application meets quality standards through beta testing, responsive testing, instrumentation testing, and user acceptance testing. This application is expected to serve as an effective tool for teachers in enhancing students' reading skills, especially for those with intellectual disabilities, and to provide meaningful support in the context of special needs education.

Keywords – Android, Application, Learning Media, Intellectual Disabilities

***Intisari* – Semua warga negara Indonesia berhak memperoleh pendidikan yang setara, termasuk anak-anak dengan kebutuhan khusus. Sekolah Luar Biasa (SLB) di Indonesia memberikan layanan pendidikan bagi siswa berkebutuhan khusus, salah satunya tunagrahita, yang memiliki tingkat IQ bervariasi dan memengaruhi kemampuan belajar, terutama dalam membaca. Metode pembelajaran membaca konvensional masih sering digunakan di SLB, membutuhkan kesabaran tinggi dan bergantung pada kemampuan guru. Untuk mengatasi kendala ini, dikembangkan aplikasi media pembelajaran membaca berbasis Android sebagai alternatif. Aplikasi ini memuat materi membaca, latihan soal, audio, gambar, manajemen siswa, dan catatan riwayat belajar. Proses pengembangan meliputi analisis kebutuhan pengguna, perancangan, implementasi, dan pengujian. Bahasa pemrograman *Kotlin* digunakan untuk pengembangan aplikasi, sementara *Firebase* digunakan sebagai basis data. Pengujian menunjukkan aplikasi ini telah memenuhi standar kualitas melalui *beta testing*, *responsive testing*, *instrumentation testing*, dan *user acceptance testing*. Aplikasi ini diharapkan menjadi alat bantu efektif bagi guru dalam meningkatkan keterampilan membaca siswa tunagrahita.**

Kata kunci – Android, Aplikasi, Media Pembelajaran, Tunagrahita

I. PENDAHULUAN

Semua warga negara Indonesia berhak mendapatkan pendidikan yang sejajar, termasuk anak-anak dengan kebutuhan khusus [1]. Hak ini mencakup pendidikan dasar hingga menengah atas. Sekolah Luar Biasa (SLB), sekolah khusus yang menyesuaikan pelayanan, alat-alat, bahan, layanan, dan strategi pembelajaran dengan karakteristik unik dari individu siswanya, hadir untuk memfasilitasi siswa yang memiliki kebutuhan khusus [2]. SLB dikategorikan menjadi beberapa jenis, salah satunya adalah jenis SLB-C tunagrahita, yaitu siswa yang memiliki intelegensi di bawah rata-rata.

Siswa tunagrahita memiliki variasi kemampuan kognitif dan intelektual yang berbeda-beda, tergantung pada tingkat keparahan gangguan mereka. Tunagrahita menunjukkan variasi dalam kategori IQ, terbagi menjadi tunagrahita ringan (IQ 50-70), sedang (IQ 30-50), berat, dan sangat berat (dengan IQ kurang dari 30), yang secara signifikan memengaruhi proses belajar mereka [3]. Salah satu aspek penting dalam proses belajar anak-anak ini adalah kemampuan membaca, yang memiliki dampak signifikan dalam memperoleh pengetahuan, memfasilitasi studi, dan memungkinkan komunikasi efektif. Beberapa siswa mungkin sudah mampu membaca kalimat, sementara yang lain belum mampu membedakan jenis huruf abjad. Oleh karena itu, setiap siswa diberikan arahan pembelajaran yang disesuaikan dengan kebutuhan masing-masing siswa.

Penulis melakukan survei di beberapa SLB di Kota Surakarta dan menemukan bahwa dalam pengajaran membaca siswa tunagrahita masih menggunakan metode konvensional seperti papan tulis dan huruf-huruf abjad yang dicetak fisik. Implementasi metode konvensional menuntut guru untuk memiliki tingkat kesabaran yang tinggi, mengingat perbedaan kemampuan yang bervariasi di antara siswa, sehingga memerlukan waktu yang lebih lama. Kondisi tersebut membuat siswa sangat bergantung pada pengetahuan dan pengalaman guru yang mengajar, suatu keterbatasan yang menunjukkan kebutuhan akan pendekatan yang lebih inovatif dan adaptif.

Perkembangan teknologi saat ini telah mencapai tahap yang pesat. Hampir semua lapisan masyarakat di Indonesia,

termasuk siswa-siswi tunagrahita di SLB, telah mengadopsi penggunaan *handphone*. Tidak hanya siswa, para guru dan orang tua juga memiliki *handphone* dengan sistem operasi Android. Penggunaan media pembelajaran digital membuat siswa tunagrahita memberikan peningkatan terhadap minat belajar mereka [4]. Penerapan pembelajaran melalui aplikasi berbasis Android memberikan manfaat signifikan bagi para guru dan orang tua. Hal ini menjadikan pembelajaran lebih mudah dan menarik bagi siswa [5].

Siswa tunagrahita memiliki beberapa kendala saat diajar oleh guru, termasuk kesulitan mengenal huruf kapital dan non-kapital, mengeja kata, membaca kata, serta membaca kalimat. Oleh karena itu, muncul ide untuk menciptakan media pembelajaran berbasis aplikasi Android sebagai solusi untuk membantu pembelajaran membaca pada siswa tunagrahita. Media pembelajaran ini memiliki fokus pada pengenalan huruf kapital dan non-kapital, pembelajaran vokal, pengembangan keterampilan merangkai kata, dan kemampuan membaca kalimat. Penggunaan aplikasi ini akan sepenuhnya diarahkan dan didampingi oleh guru. Harapannya, aplikasi pembelajaran ini tidak hanya berfungsi sebagai alat pengajaran, tetapi juga sebagai alat pemantauan kemampuan membaca siswa oleh para guru atau orang tua.

II. DASAR TEORI

Bab ini mengulas berbagai literatur yang terkait dengan aplikasi media pembelajaran berbasis Android untuk tunagrahita. Tinjauan pustaka ini menjadi fondasi dalam proses pembuatan aplikasi untuk memastikan hasil yang tepat guna dan bermanfaat bagi penggunanya. Berikut akan disajikan beberapa sumber literatur yang menjadi acuan dalam penelitian ini.

Penelitian yang dilakukan oleh [4] bertujuan untuk mengembangkan media pembelajaran berbasis multimedia yang khusus ditujukan untuk siswa tunagrahita kelas V di SD SLBIT Baitul Jannah Bandar Lampung. Dalam proses rancang bangun, penelitian ini menggunakan Metode Pengembangan *Multimedia Development Life Cycle* (MDLC), yang terdiri dari tahap-tahap seperti konseptualisasi, desain, pengumpulan materi, perakitan, pengujian, dan distribusi. Hasil dari penelitian ini adalah sebuah media pembelajaran berbasis android yang dikembangkan dengan menggunakan perangkat lunak Construct 2. Media pembelajaran ini berfokus pada pengenalan angka dari 1 hingga 100 serta perhitungan angka dari 1 hingga 20, yang disesuaikan dengan kebutuhan dan tingkat pemahaman anak tunagrahita kelas V. Aplikasi media pembelajaran yang dikembangkan ini memiliki beragam fitur seperti menu, game, video, materi pembelajaran, simulasi, latihan soal, dan evaluasi. Tujuan utama dari pengembangan media ini adalah memberikan pendekatan pembelajaran yang lebih interaktif dan efektif, yang dapat membantu anak tunagrahita dalam memahami konsep angka dan perhitungan.

Penelitian yang dilakukan oleh [6] telah menghasilkan aplikasi "Gredio," berbasis Android, yang khusus dirancang untuk anak tunagrahita di SLB B-C YPLAB Wartawan Turangga Lengkong. Aplikasi ini mengadopsi metode fonetis dalam proses pembelajaran membaca dengan pendekatan hafalan bunyi setiap rangkaian huruf dari suku kata. Gredio menyediakan beragam media pembelajaran, termasuk suara, video, objek gambar, dan teks, yang bertujuan untuk memaksimalkan efektivitas pembelajaran anak tunagrahita. Selain itu, aplikasi ini juga menyertakan latihan-latihan khusus yang membantu anak tunagrahita mengingat materi pembelajaran dengan lebih baik, sementara penggunaan video dimaksudkan untuk menjaga minat mereka sehingga pembelajaran menjadi lebih menarik dan tidak membosankan. Diharapkan bahwa aplikasi Gredio dapat menjadi alat yang efektif dalam mendukung anak tunagrahita dalam memahami konsep membaca dengan metode fonetis.

Penelitian yang dilakukan oleh [7] aplikasi Edugra diaplikasikan pada *handphone* android dengan fokus penggunaan pada siswa tunagrahita di SLB B-C YPLAB. Hasil pengujian aplikasi menunjukkan ketertarikan siswa tunagrahita terhadap aplikasi tersebut, walaupun informasi yang tersedia belum memberikan rincian mendalam mengenai hasil evaluasi. Harapan dari pengembangan aplikasi Edugra adalah agar siswa tunagrahita dapat lebih mudah memahami kegiatan bina diri serta mempelajari dasar-dasar membaca, berhitung, dan mengenal benda. Aplikasi ini diharapkan mampu menjadi alat pembelajaran yang efektif dan menarik bagi siswa tunagrahita, membantu mereka mengembangkan keterampilan dasar, serta meningkatkan pemahaman mereka tentang dunia sekitar.

Penelitian yang dilakukan oleh [5] telah menghasilkan sebuah aplikasi media pembelajaran berbasis Android yang bertujuan untuk memfasilitasi guru dan orang tua dalam membantu anak tunagrahita kelas 1-4 di SDLB Negeri Bekasi Jaya dalam pengembangan kemampuan bina diri. Metode pengembangan yang digunakan adalah *Model Rapid Application Development* (RAD). Aplikasi ini menawarkan beragam materi pembelajaran, termasuk video, suara, dan deskripsi yang menggambarkan kegiatan sehari-hari anak-anak dalam merawat dan mengurus diri. Keunggulan utamanya adalah aksesibilitas melalui perangkat android yang memungkinkan pengguna untuk mengaksesnya di mana saja dan kapan saja. Aplikasi ini diharapkan dapat menjadi alat bantu yang efektif bagi orang tua dan guru dalam mendukung upaya pembinaan diri anak-anak tunagrahita, memungkinkan mereka untuk mengembangkan keterampilan bina diri dengan lebih baik.

Penelitian yang dilakukan oleh [4] berfokus pada pengembangan media pembelajaran dalam bentuk game edukasi yang ditujukan untuk pengenalan angka 1-100 dan perhitungan 1-20, khususnya dirancang untuk anak tunagrahita kelas V di SD SLB. Pendekatan pengembangan yang digunakan dalam penelitian ini adalah *Multimedia Development Life Cycle* (MDLC). Hasil penelitian menunjukkan bahwa *game* edukasi yang dikembangkan berhasil mendapatkan respons positif dalam uji coba. Media

pembelajaran ini terbukti efektif sebagai alat bantu pendidikan bagi anak tunagrahita kelas V. Penggunaan teknologi dalam konteks pendidikan, terutama dalam era revolusi industri 4.0, menekankan pentingnya penguasaan perangkat lunak (*software*) sebagai keterampilan yang diperlukan oleh pendidik untuk menciptakan media pembelajaran yang efektif dan sesuai dengan kebutuhan pembelajaran anak tunagrahita.

Penelitian yang dilakukan oleh [8] mengangkat tema implementasi *Multimedia Development Life Cycle* (MDLC) dalam pengembangan aplikasi media pembelajaran untuk anak tunagrahita. Hasil penelitian ini berfokus pada penciptaan aplikasi bernama "Edinata," yang bertujuan untuk memfasilitasi anak tunagrahita dalam pengenalan huruf, angka, benda-benda sekitar, serta menyajikan elemen permainan mewarnai. Dalam konteks pendidikan anak tunagrahita, pendekatan multimedia interaktif ini diharapkan dapat memberikan solusi efektif dalam proses pembelajaran mereka. Aplikasi "Edinata" menjadi sebuah alat bantu yang berpotensi untuk mendukung anak tunagrahita dalam mengembangkan pemahaman mereka terhadap huruf, angka, dan lingkungan sekitar, dengan memanfaatkan teknologi multimedia sebagai sarana pembelajaran yang menarik dan interaktif.

Penelitian yang dilakukan oleh [9] berjudul "Efektivitas Media Pembelajaran Berbagai Aplikasi Dalam Meningkatkan Pemahaman Materi Sains Untuk Siswa Tunagrahita: *Literature Review*" bertujuan untuk menganalisis efektivitas penggunaan media pembelajaran dari berbagai aplikasi dalam meningkatkan pemahaman materi sains bagi siswa tunagrahita. Metode yang digunakan dalam penelitian ini adalah *Systematic Literature Review* (SLR). Hasil dari *review* literatur ini menunjukkan bahwa implementasi kurikulum 2013 yang sejalan dengan penggunaan media pembelajaran yang disesuaikan dengan kebutuhan siswa tunagrahita mampu memperbaiki pemahaman mereka terhadap materi sains. Hal ini menunjukkan bahwa pendekatan ini berpotensi untuk memberikan kontribusi positif dalam meningkatkan efektivitas proses pembelajaran siswa tunagrahita, khususnya dalam pemahaman materi sains yang mereka pelajari.

Penelitian yang dilakukan oleh [10] dalam judul "Perancangan Media Pembelajaran Aritmatika Berbasis Web Untuk Anak Tunagrahita Ringan" bertujuan untuk mengembangkan media pembelajaran berbasis web yang khusus ditujukan untuk anak tunagrahita ringan. Dalam proses pengembangan, penelitian ini mengadopsi model *Linear Sequential/Waterfall Model*, yang merupakan pendekatan klasik yang sistematis dan berurutan dalam pembuatan perangkat lunak. Aplikasi media pembelajaran aritmetika yang dihasilkan telah diimplementasikan di SLB Negeri Pohuwato dan telah melalui serangkaian pengujian *whitebox* dan *blackbox* untuk memastikan fungsionalitasnya.

Harapan dari penggunaan aplikasi ini adalah dapat meningkatkan motivasi belajar anak tunagrahita di SLB Negeri Pohuwato. Dengan demikian, hasil penelitian ini mendukung penyediaan alat pembelajaran yang sesuai dengan kebutuhan anak tunagrahita ringan dan mampu meningkatkan kualitas pembelajaran mereka.

Penelitian yang dilakukan oleh [11] mengenai "Multimedia Interaktif untuk Meningkatkan Kemampuan Membaca Permulaan Siswa Tunagrahita" telah menghasilkan multimedia interaktif yang disampaikan dalam bentuk CD. Metode penelitian yang digunakan diadaptasi dari Borg and Gall. Hasil penelitian tersebut menunjukkan bahwa multimedia interaktif tersebut efektif dalam meningkatkan kemampuan membaca permulaan siswa tunagrahita di SLB Dr. Idayu 1 Malang. Pencapaian ini didukung oleh hasil validasi oleh ahli media sebesar 97% (sangat layak), validasi ahli materi sebesar 76% (layak), dan validasi ahli pembelajaran ABK sebesar 88% (sangat layak). Dengan demikian, penelitian ini memberikan kontribusi penting dalam pengembangan media pembelajaran yang efektif untuk meningkatkan kemampuan membaca permulaan siswa tunagrahita, sekaligus membuktikan bahwa multimedia interaktif dapat menjadi alat yang layak dan efektif dalam konteks pendidikan bagi siswa dengan kebutuhan khusus.

Dalam penelitian yang dilakukan oleh [12] berjudul "Rancang Bangun Aplikasi Pembelajaran Huruf Hijaiyah dan Tajwid Berbasis Android (Studi Kasus SLB Negeri 1 Jakarta)," diterapkan metode ADDIE (*Analysis, Design, Development, Implementation, Evaluation*) dalam pengembangan aplikasi pembelajaran. Aplikasi ini ditujukan untuk kasus SLB Negeri 1 Jakarta dan hasilnya adalah produk multimedia interaktif bernama "Aku Gemar Membaca," yang telah disesuaikan untuk siswa tunagrahita. Aplikasi ini dirancang dengan antarmuka yang sederhana, mudah digunakan, dan disertai dengan petunjuk bagi guru. Media ini kaya akan gambar, audio, video, serta animasi menarik, dan menerapkan metode *Sequential Auditory System* (SAS) dengan penggunaan audio yang signifikan. Dengan demikian, aplikasi ini mendorong siswa untuk belajar secara mandiri. Aplikasi "Aku Gemar Membaca" tersedia dalam format CD, memastikan kemudahan penggunaan dan penyimpanan, sehingga menjadi sebuah alat pembelajaran yang sesuai dengan kebutuhan siswa tunagrahita dan mendukung proses pembelajaran mereka secara efektif.

Penelitian-penelitian terkait pengembangan aplikasi media pembelajaran membaca siswa tunagrahita berbasis android yang dijabarkan di atas menunjukkan bahwa aplikasi dapat bermanfaat bagi siswa tunagrahita.

III. METODOLOGI

Metodologi yang digunakan dalam penelitian ini yaitu analisis, perancangan, implementasi rancangan, dan pengujian.

1. Analisis

Analisis adalah tahapan awal dan penting dalam rancang bangun aplikasi media pembelajaran. Tahapan ini dilakukan analisis kebutuhan pengguna sehingga bisa ditentukan fitur-fitur yang perlu dikembangkan dalam aplikasi.

2. Perancangan

Tahap perancangan adalah proses pengembangan rincian teknis dari aplikasi berdasarkan kebutuhan pengguna yang telah diidentifikasi pada tahap analisis. Rancangan *use case* berdasarkan fitur yang telah ditentukan dapat dilihat pada Gambar 1.

Gambar 1. *Use case* aplikasi media pembelajaran

3. Implementasi

Tahap implementasi merupakan langkah dalam pengembangan perangkat lunak di mana kode program dibangun sesuai dengan rancangan yang telah disusun pada tahap perancangan sebelumnya. Ini adalah tahap penting dalam mewujudkan konsep dan desain menjadi produk perangkat lunak yang fungsional.

4. Pengujian

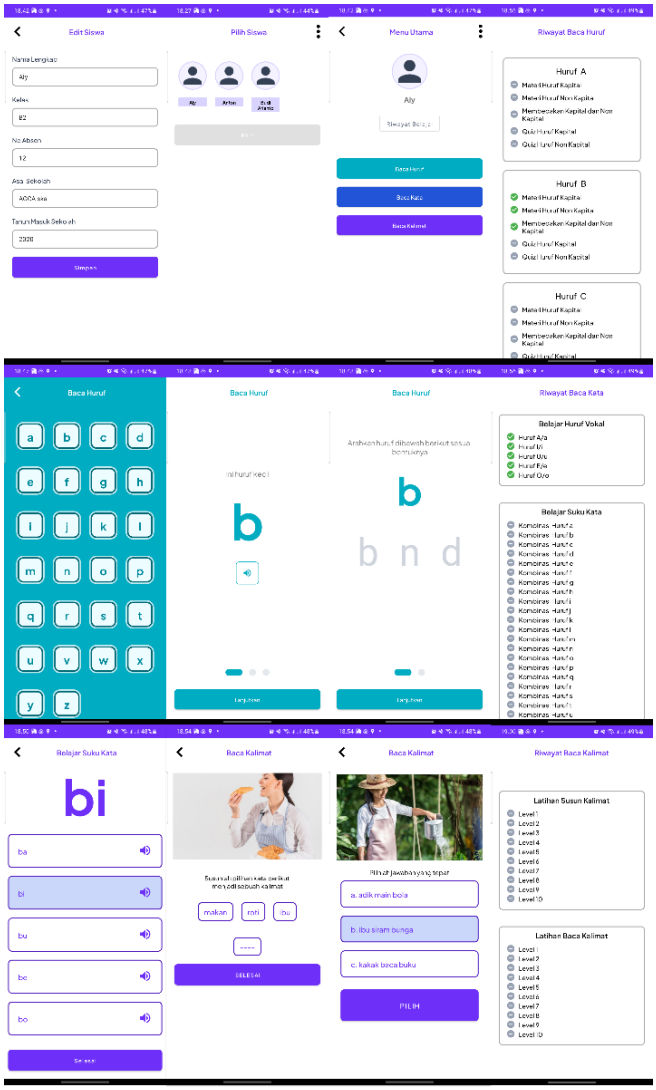
Pengujian adalah tahap akhir dalam proses rancang bangun aplikasi, di mana aplikasi yang telah dikembangkan akan diuji untuk memastikan bahwa semua fungsionalitas berjalan dengan benar dan sesuai dengan ekspektasi.

IV. HASIL DAN PEMBAHASAN

Implementasi hasil rancangan adalah hasil pengubahan rancangan aplikasi menjadi sebuah produk aplikasi media pembelajaran membaca tunagrahita. Hasil pengujian dilakukan melalui empat tahap, yaitu *beta testing*, *instrumentation testing*, *responsive testing*, dan *User Acceptance Testing (UAT)*.

A. Implementasi Hasil Rancangan

Implementasi aplikasi media pembelajaran berbasis android menggunakan bahasa *kotlin* dan *firebase*. Hasil dari implementasi dapat dilihat pada Gambar 2.



Gambar 2. Hasil implementasi aplikasi

B. Hasil *Beta Testing*

Hasil *beta testing* melibatkan perwakilan dari 3 SLB menunjukkan bahwa semua skenario pengujian berjalan lancar dan sesuai dengan ekspektasi. Meskipun demikian, beberapa aspek teknis memerlukan perbaikan dan optimalisasi, seperti yang tercatat dalam kolom saran pada Tabel 1. Setelah menerima saran dan umpan balik, tim pengembang telah melakukan perbaikan teknis yang diperlukan. Dengan penyelesaian perbaikan ini, aplikasi siap memasuki tahap *testing* selanjutnya, diharapkan mampu memberikan pengalaman pengguna yang lebih optimal dan berkualitas saat menghadapi tahap uji coba selanjutnya.

Tabel 1. Hasil *beta testing*

No	Aksi	Hasil	Saran
1	Authentifikasi	Pass	Menambah *scroll* pada halaman *register*.
2	Manajemen Siswa	Pass	Menambah *scroll* pada halaman tambah dan edit siswa.
3	Fitur Baca Huruf	Pass	Memperbaiki *loading* suara terlalu lama.
4	Fitur Baca Kata	Pass	Memperbaiki *loading* suara terlalu lama, ukuran *font* dan gambar latihan soal.
5	Fitur Baca Kalimat	Pass	Memperbaiki *loading* suara terlalu lama, ukuran *font* dan gambar latihan soal.
6	Riwayat	Pass	-

C. Hasil *Responsive Testing*

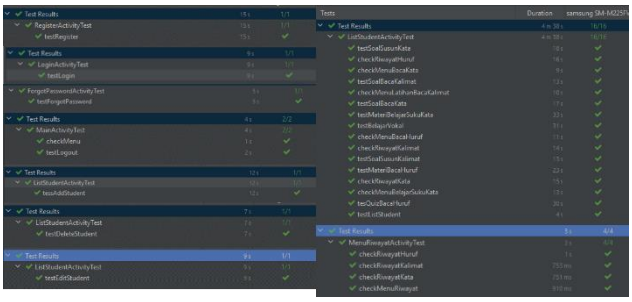
Setelah melewati *responsive testing* pada berbagai perangkat seperti Samsung M22, emulator Samsung Galaxy Note 2, dan Tablet Samsung Galaxy SM-X200, aplikasi menunjukkan performa optimal. Semua fitur aplikasi berfungsi baik dan sesuai di semua ukuran layar perangkat yang diuji, tanpa masalah tampilan, sebagaimana tercatat dalam Tabel 2. Ini menegaskan bahwa aplikasi responsif dan dapat digunakan pada berbagai jenis perangkat, memberikan pengalaman pengguna yang konsisten dan memuaskan.

Tabel 2. Hasil *responsive testing*

No	Aksi	Samsung M22(6,4")	Samsung Galaxy Note 2(7")	Samsung SM-X200 (10,5")
1	Authentifikasi	Pass	Pass	Pass
2	Manajemen Siswa	Pass	Pass	Pass
3	Fitur Baca Huruf	Pass	Pass	Pass
4	Fitur Baca Kata	Pass	Pass	Pass
5	Fitur Baca Kalimat	Pass	Pass	Pass
6	Riwayat	Pass	Pass	Pass

D. Hasil *Instrumentation Testing*

Setelah menjalankan serangkaian pengujian otomatis menggunakan Espresso sebagai alat *instrumentation testing*, dapat dilihat bahwa semua skenario pengujian telah berhasil dilaksanakan, sebagaimana terlihat pada Gambar 3. Espresso, yang terkenal dengan kemampuannya untuk melakukan pengujian UI dan integrasi di aplikasi Android, telah menunjukkan bahwa setiap fitur dan interaksi di dalam aplikasi bekerja dengan sesuai dan tanpa hambatan. Hal ini menegaskan bahwa aplikasi telah dikembangkan dengan stabil dan siap untuk diuji tahapan selanjutnya.



Gambar 3. Hasil *instrumentation testing*

E. Hasil *User Acceptance Testing*(UAT)

Hasil dari *User Acceptance Testing* (UAT) menunjukkan bahwa semua fitur aplikasi telah melewati uji coba dengan sukses, sebagaimana terdokumentasi dalam Tabel 3. Setiap skenario pengujian menghasilkan hasil "Pass", menegaskan bahwa aplikasi memenuhi ekspektasi dan kebutuhan pengguna akhir. Kesuksesan ini mencerminkan komitmen pengembangan aplikasi untuk memperhatikan kebutuhan dan ekspektasi pengguna serta memenuhi standar kualitas yang diharapkan. Dengan hasil positif dari UAT ini, aplikasi telah mencapai tingkat kesiapan untuk diluncurkan dan digunakan oleh pengguna akhir dalam lingkungan produksi. Diharapkan bahwa aplikasi ini akan memberikan pengalaman pengguna yang efektif dan memuaskan dalam mencapai tujuan mereka.

Tabel 3. Hasil UAT

No	Aksi	Hasil	Saran
1	Authentifikasi	Pass	-
2	Manajemen Siswa	Pass	-
3	Fitur Baca Huruf	Pass	-
4	Fitur Baca Kata	Pass	-
5	Fitur Baca Kalimat	Pass	-
6	Riwayat	Pass	-

V. SIMPULAN

Berdasarkan hasil penelitian mengenai rancang bangun aplikasi pembelajaran berbasis android untuk siswa tunagrahita diperoleh kesimpulan bahwa, aplikasi media pembelajaran yang berfokus pada materi membaca yaitu baca huruf, baca kata, dan baca kalimat telah berhasil dikembangkan menggunakan bahasa *kotlin* dengan menggunakan basis data *firebase*. Dalam memastikan semua fitur yang dirancang dapat digunakan dengan baik telah dilakukan empat tahap pengujian yaitu *beta testing*, *instrumentation testing*, *responsive testing*, dan *user acceptance testing*. Hasil pengujian aplikasi diperoleh hasil yang baik yang dapat disimpulkan bahwa aplikasi sudah siap digunakan oleh pengguna yang lebih luas.

REFERENSI

[1] F. Husna, N. R. Yunus, and A. Gunawan, "Hak Mendapatkan Pendidikan Bagi Anak Berkebutuhan Khusus Dalam Dimensi Politik Hukum Pendidikan," *SALAM: Jurnal Sosial dan Budaya Syar-i*, vol. 6, no. 2, pp. 207–222, Mar. 2019, doi: 10.15408/sjsbs.v6i1.10454.

[2] F. Nasution, Y. L. Anggraini, and K. Putri, "Pengertian Pendidikan, Sistem Pendidikan Sekolah Luar Biasa, dan Jenis-Jenis Sekolah," *Jurnal Edukasi Nonformal*, vol. 3, no. 2, 2022.

[3] N. L. G. K. Widiastuti and I. M. A. Winaya, "Prinsip Khusus dan Jenis Layanan Pendidikan Bagi Anak Tunagrahita," 2019.

[4] H. Saputra and E. Febriyanto, "Media Pembelajaran Berbasis Multimedia Untuk Anak Tuna Grahita," *Mathema: Jurnal Pendidikan Matematika*, vol. 1, no. 1, 2019.

[5] A. Fitriyani, H. Lubis, and A. Achmad, "Media Pembelajaran Bina Diri Anak Tunagrahita SDLB Negeri Bekasi Jaya Berbasis Android," *JSI (Jurnal Sistem Informasi) Universitas Suryadarma*, 2023.

[6] K. Sundari, B. Yudhi, and M. R. Mutaqin, "Gredio Aplikasi Belajar Membaca Untuk Anak Tunagrahita Dengan Metode Fonetis," 2015.

[7] D. K. Wardani, "Aplikasi Edukasi untuk Anak Tunagrahita Ringan Berbasis Android," 2014.

[8] P. Ambarwati and P. Syifa Darmawel, "Implementasi Multimedia Development Life Cycle pada Aplikasi Media Pembelajaran untuk Anak Tunagrahita," *Majalah Ilmiah UNIKOM*, vol. 18, no. 2, 2020.

[9] D. A. Rokhim, J. A. Nenohai, N. I. Agustina, and M. Munzil, "Efektivitas Media Pembelajaran Berbagai Aplikasi dalam Meningkatkan Pemahaman Materi Sains untuk Siswa Tunagrahita," *UNESA Journal of Chemical Education*, vol. 12, no. 1, pp. 37–43, Jan. 2023, doi: 10.26740/ujced.v12n1.p37-43.

[10] Nursyaida, Anas, A. S. Anwar, A. Y. Labolo, and Azwar, "Perancangan Media Pembelajaran Aritmatika Berbasis Web untuk Anak Tunagrahita Ringan," *Simtek: jurnal sistem informasi dan teknik komputer*, vol. 7, no. 2, 2022.

[11] A. Fauzia and U. Kustiawan, "Multimedia Interaktif untuk Meningkatkan Kemampuan Membaca Permulaan Siswa Tunagrahita," 2017.

[12] A. Muzhaffar, "Rancang Bangun Aplikasi Pembelajaran Huruf Hijaiyah dan Tajwid Berbasis Android(Studi Kasus SLB Negeri 1 Jakarta)," 2022.