

**IMPLEMENTASI PENYIMPANAN DOKUMEN MENGGUNAKAN  
GOOGLE CLOUD STORAGE PADA APLIKASI VOCAJECT**

**SKRIPSI**

Diajukan Sebagai Salah Satu Syarat Untuk Menyelesaikan  
Pendidikan Jenjang Sarjana Terapan  
Pada Politeknik Negeri Lhokseumawe



**Oleh  
ZULFAHMI**

**Nim : 2020903430056**  
**Program Studi : Teknologi Rekayasa Komputer Jaringan**  
**Jurusan : Teknologi Informasi Komputer**

**KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET DAN TEKNOLOGI  
POLITEKNIK NEGERI LHOKSEUMAWE**

**2024**

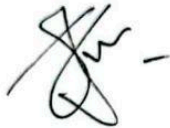
## PENGESAHAN PEMBIMBING

Skripsi yang berjudul *Implementasi Penyimpanan Dokumen Menggunakan Google Cloud Storage Pada Aplikasi Vocaject*, disusun oleh Zulfahmi, NIM 2020903430056, Program Studi Teknologi Rekayasa Komputer Jaringan, Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe telah memenuhi syarat untuk dipertanggungjawabkan di depan dewan penguji.

Buketrata 30 Mei 2024

Pembimbing I,

Pembimbing II,



**Husaini, S.Si., M.IT**  
NIP. 197310312001121001



**Fachri Ynuar Rudi F, M.T.**  
NIP. 198801062018031001

Mengetahui:

Ketua Jurusan  
Teknologi Informasi dan Komputer,

Ketua Program Studi  
Teknologi Rekayasa Komputer  
Jaringan,

**Muhammad Nasir, S.T., M.T.**  
NIP. 19750707199903100

**Nanda Saputri, S.ST., M.T.**  
NIP. 199111202022032010

## PENGESAHAN PENGUJI

Skripsi yang berjudul *Implementasi Penyimpanan Dokumen Menggunakan Google Cloud Storage Pada Aplikasi Vocaject*, Disusun oleh Zulfahmi, NIM 2020903430056, Program Studi Teknologi Rekayasa Komputer Jaringan, Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe telah dipertanggungjawabkan di hadapan dewan penguji dan dinyatakan lulus pada x Mei 2024

Dewan Penguji,

Ketua,

Sekretaris

Nama  
NIP.

Nama  
NIP.

Penguji 1,

Penguji 2,

Penguji 3,

Nama  
NIP.

Nama  
NIP.

Nama  
NIP.

Mengetahui,

Ketua Jurusan

Teknologi Informasi dan Komputer,

Ketua Program Studi

Teknologi Rekayasa Komputer  
Jaringan,

Muhammad Nasir, S.T., M.T.  
NIP. 197507071999031002

Nanda Saputri, S.ST., M.T.  
NIP. 199111202022032010

## **PERNYATAAN**

Skripsi ini saya nyatakan tidak terdapat karya yang pernah diajukan untuk memperoleh gelar Kesarjanaan pada suatu perguruan tinggi dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Buketrata 30 Mei 2024  
Penulis

Zulfahmi  
NIM. 2020903430056

## KATA PENGANTAR

Segala puji serta syukur senantiasa penulis panjatkan kepada Allah SWT yang telah memberikan rahmat dan petunjuk-Nya kepada penulis sehingga dapat menyelesaikan skripsi yang berjudul Implementasi Penyimpanan Dokumen Menggunakan Google Cloud Storage Pada Aplikasi *Vocaject*. Shalawat serta salam penulis hantarkan keharibaan junjungan kita Nabi Muhammad SAW yang telah membawa umat manusia dari alam jahiliyah ke alam Islamiyah, dari alam kebodohan ke alam yang penuh dengan ilmu pengetahuan. Skripsi ini disusun untuk memenuhi salah satu syarat dalam menyelesaikan pendidikan diploma IV Program Studi Teknologi Rekayasa Komputer Jaringan Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe.

Dalam penyusunan skripsi ini penulis banyak mendapat dukungan, dorongan, bimbingan dan bantuan dari berbagai pihak sehingga skripsi ini dapat diselesaikan. Oleh karena itu, penulis mengucapkan terima kasih kepada :

1. Teristimewa orang tua penulis yaitu Rosmiati selaku ibu penulis, keempat saudara penulis dan seluruh keluarga penulis yang telah dan selalu memberi kasih sayang kepada penulis sehingga dapat menyelesaikan skripsi ini.
2. Bapak Husaini, S.Si., M.IT dan Bapak Fahri Yanuar Rudi F, M.T. selaku pembimbing I dan II yang telah membimbing penulis, memberi petunjuk, memberi saran-saran dan arahan serta masukan sejak perlombaan KMIPN tahun 2023 sampai sekarang dapat menyelesaikan skripsi ini.
3. Bapak Ir. Rizal Syahyadi, ST., M. Eng., Sc. IPM, Asean Eng. selaku Direktur Politeknik Negeri Lhokseumawe.
4. Bapak Muhammad Nasir, S.T., M.T. selaku Ketua Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe.
5. Bapak Salahuddin, S.T., M.Cs selaku Sekretaris Jurusan Teknologi Informasi dan Komputer Politeknik Negeri Lhokseumawe.

6. Ibu Nanda Saputri, S.ST., M.T. selaku ketua Program Studi Teknologi Rekayasa Komputer Jaringan Politeknik Negeri Lhokseumawe.
7. Seluruh staf pengajar Program Studi Teknologi Rekayasa Komputer Jaringan, serta staf administrasi pada Jurusan Teknologi Informasi dan Komputer.
8. Terkhusus teman-teman dari Vocaject-Team yaitu Saiful Kamil dan Faiza Yuwafiqi, yang selalu mendampingi dan berbagi pikiran pada setiap kegiatan yang Vocaject-Team lakukan hingga bersama-sama dapat menyelesaikan skripsi ini.
9. Tak terlupakan, teman-teman angkatan 2020 Program Studi Teknologi Rekayasa Komputer Jaringan dan juga teman-teman dari Program Studi Teknik Informatika. Serta pihak yang telah berpartisipasi dalam penyelesaian skripsi ini.

Akhir kata penulis ucapkan banyak terima kasih kepada seluruh pihak yang membantu dan semoga skripsi ini dapat bermanfaat bagi pembaca, khususnya bagi penulis sendiri dan kepada peneliti-peneliti lainnya. Amin, Ya Rabbal Alamin.

Lhokseumawe 30 Mei 2024  
Penulis,

Zulfahmi  
NIM. 2020903430056

## ABSTRAK

*Vocaject* merupakan aplikasi yang digunakan dalam mendukung *Project-Based Learning* di lingkungan akademis. Saat ini belum ada fitur keamanan untuk mencegah akses oleh pihak yang tidak memiliki otoritas terhadap suatu dokumen yang disimpan pada aplikasi *Vocaject*. Dokumen yang diunggah dalam aplikasi *Vocaject* masih disimpan di sisi server sehingga beresiko kehilangan yang disebabkan oleh kerusakan atau terjadi serangan terhadap *server backend*. Untuk mengatasi hal tersebut, penelitian ini akan melakukan implementasi penyimpanan dokumen menggunakan *Google Cloud Storage* pada aplikasi *Vocaject*. Tujuan dari penelitian ini untuk menganalisis hasil setelah diimplementasikan dengan *Cloud Storage* terhadap keamanan akses dokumen dan terhadap ketersediaan data ketika terjadi serangan. Dari hasil pengujian, sistem berhasil menjalankan fungsionalnya dengan persentase 100% dan mencegah akses oleh pihak yang tidak memiliki otoritas bahkan tidak dapat dibaca melalui tangkapan menggunakan *Wireshark* dengan persentase 100%. *Vocaject* juga mampu menangani setelah diberi beban 50 sampai 100 permintaan dengan *error* 0% dan *throughput* 18 sampai 23 setiap detik. Namun memiliki batas kapasitasnya pada beban antara 250 yang terdapat *error* rata-rata sebesar 38% serta *throughput* 21.9 setiap detik dan pada beban 500 permintaan terdapat *error* rata-rata sebesar 61.7% serta *throughput* rata-rata 46.37 setiap detik.

Kata Kunci: *Google Cloud Storage, Vocaject, Project-Based Learning, Stress Testing, Laravel.*

## **ABSTRACT**

*Vocaject is an application used to support Project-Based Learning in academic environments. Currently, there are no security features to prevent unauthorized access to documents stored in the Vocaject application. The documents uploaded to the Vocaject application are still stored on the server side, which poses a risk of loss due to damage or attacks on the backend server. To address this issue, this research will implement document storage using Google Cloud Storage in the Vocaject application. The goal of this research is to analyze the results after implementing Cloud Storage in terms of document access security and data availability when an attack occurs. From the test results, the system successfully performed its functions with a 100% success rate and prevented access by unauthorized parties, even making it unreadable through captures using Wireshark with a 100% success rate. Vocaject was also able to handle loads of 50 to 100 requests with a 0% error rate and throughput of 18 to 23 requests per second. However, it reached its capacity limit under a load of 250 requests, with an average error rate of 38% and a throughput of 21.9 requests per second. Under a load of 500 requests, there was an average error rate of 61.7% and an average throughput of 46.37 requests per second.*

*Keywords: Google Cloud Storage, Vocaject, Project-Based Learning, Stress Testing, Laravel.*



## DAFTAR ISI

<b>PENGESAHAN PEMBIMBING.....</b>	<b>i</b>
<b>PENGESAHAN PENGUJI.....</b>	<b>ii</b>
<b>PERNYATAAN.....</b>	<b>iii</b>
<b>KATA PENGANTAR.....</b>	<b>iv</b>
<b>ABSTRAK.....</b>	<b>vi</b>
<b>ABSTRACT.....</b>	<b>vii</b>
<b>DAFTAR ISI.....</b>	<b>viii</b>
<b>DAFTAR TABEL.....</b>	<b>x</b>
<b>DAFTAR GAMBAR.....</b>	<b>xi</b>
<b>BAB I PENDAHULUAN.....</b>	<b>1</b>
1.1. Latar Belakang.....	1
1.2. Rumusan Masalah.....	2
1.3. Tujuan Penelitian.....	2
1.4. Batasan Masalah.....	3
1.5. Manfaat Penelitian.....	3
<b>BAB II TINJAUAN PUSTAKA.....</b>	<b>4</b>
2.1. State of the art.....	4
2.2. Tinjauan Teoritis.....	8
2.2.1. Cloud.....	8
2.2.2. Google Cloud Platform.....	10
2.2.2.1. Keunggulan Google Cloud Platform.....	11
2.2.2.2. Kekurangan Google Cloud Platform.....	12
2.2.3. Cloud Storage.....	13
2.2.4. Google Cloud Storage.....	14
2.2.5. Compute Engine.....	15
2.2.6. Google Load Balancing.....	15
2.2.7. Laravel.....	16
2.2.8. Distributed Denial of Service (DDoS).....	16
2.2.9. Stress Testing.....	17
2.2.10. Postman.....	17
2.2.11. Apache JMeter.....	18
2.2.12. Wireshark.....	18

<b>BAB III METODOLOGI PENELITIAN.....</b>	<b>20</b>
3.1. Tahapan Penelitian.....	20
3.2. Perancangan Sistem.....	21
3.2.1. Perancangan Arsitektur Deployment.....	21
3.2.2. Perancangan Sistem.....	25
3.2.3. Perancangan Diagram Use Case.....	26
3.3. Metode Pengujian.....	29
3.4. Hasil yang diharapkan.....	30
<b>BAB IV HASIL DAN PEMBAHASAN.....</b>	<b>31</b>
4.1. Tampilan Website Vocaject.....	31
4.1.1. Halaman Login.....	31
4.1.2. Halaman Jelajah Proyek.....	32
4.1.3. Halaman Detail Proyek.....	33
4.1.4. Halaman Diskusi Proyek.....	34
4.2. Tampilan Cloud Storage Buckets.....	35
4.3. Hasil Pengujian Fungsional.....	36
4.3.1. Pengujian Mengunggah Dokumen.....	36
4.3.2. Pengujian Mengunduh Dokumen.....	40
4.4. Hasil Pengujian Keamanan Akses.....	42
4.5. Hasil Pengujian Sniffing.....	45
4.5.1. Pengujian Sniffing Saat Mengunggah Dokumen.....	45
4.5.2. Pengujian Sniffing Saat Mengunduh Dokumen.....	47
4.6. Hasil Pengujian Ketersediaan.....	49
<b>BAB V PENUTUP.....</b>	<b>52</b>
<b>5.1. Simpulan.....</b>	<b>52</b>
<b>5.2. Saran.....</b>	<b>53</b>
<b>DAFTAR PUSTAKA.....</b>	<b>54</b>

## DAFTAR TABEL

Tabel 2.1 State of The Art.....	5
Tabel 4.1 Hasil pengujian mengunggah dokumen.....	39
Tabel 4.2 Hasil pengujian mengunduh dokumen.....	42
Tabel 4.3 Hasil pengujian sniffing saat dokumen diunggah.....	46
Tabel 4.5 Spesifikasi server yang digunakan pada penelitian.....	49
Tabel 4.6 Hasil pengujian stress testing.....	50

## DAFTAR GAMBAR

Gambar 2.1 Cloud.....	9
Gambar 2.2 Google Cloud Platform.....	11
Gambar 2.3 Google Cloud Storage.....	14
Gambar 2.4 Laravel.....	16
Gambar 3.1 Tahapan Penelitian.....	20
Gambar 3.2 Ilustrasi Arsitektur Deployment.....	22
Gambar 3.3 Ilustrasi rancangan sistem.....	25
Gambar 3.4 Diagram Use Case aplikasi Vocaject.....	27
Gambar 4.1 Halaman Login Website Vocaject.....	32
Gambar 4.2 Halaman jelajahi proyek.....	33
Gambar 4.3 Halaman detail proyek.....	34
Gambar 4.4 Halaman diskusi proyek.....	35
Gambar 4.5 Tampilan Cloud Storage.....	35
Gambar 4.6 Jenis dokumen yang akan diunggah.....	36
Gambar 4.7 Popup memilih dokumen yang akan diunggah.....	37
Gambar 4.8 Hasil dokumen berhasil diunggah.....	38
Gambar 4.9 Tampilan hasil unggahan dokumen ke cloud storage.....	39
Gambar 4.10 Tampilan untuk menekan tombol unduh.....	41
Gambar 4.11 Tampilan dokumen terunduh.....	41
Gambar 4.12 Tampilan inspect halaman web.....	43
Gambar 4.13 Hasil pengujian akses dokumen.....	44
Gambar 4.14 Tampilan respon yang tidak terautentikasi.....	44
Gambar 4.15 Hasil unggahan dokumen melalui Postman.....	45
Gambar 4.16 Rekaman paket unggahan dokumen menggunakan Wireshark.....	46

Gambar 4.17 Tampilan mengakses dokumen menggunakan Postman.....	47
Gambar 4.18 Rekaman paket akses dokumen menggunakan Wireshark.....	48
Gambar 4.19 Pengujian menggunakan apache jmeter.....	50

# **BAB I**

## **PENDAHULUAN**

### **1.1. Latar Belakang**

Seiring dengan kemajuan teknologi, terjadi perkembangan pesat di bidang penyimpanan data. Dulu, penyimpanan data lebih mengandalkan perangkat keras lokal seperti hard disk dan server fisik. Namun, dengan munculnya solusi-solusi penyimpanan awan (*cloud storage*), pendekatan terhadap penyimpanan data mengalami perubahan mendasar.

Penggunaan metode konvensional dalam menyimpan dokumen seringkali dihadapkan pada berbagai tantangan keamanan. Data yang disimpan di perangkat keras lokal dapat rentan terhadap risiko kehilangan akibat bencana alam atau kerusakan perangkat keras. Selain itu, akses ke dokumen konvensional juga bisa menjadi mudah diakses oleh pihak yang tidak berwenang, menyebabkan kerentanan keamanan yang signifikan.

*Cloud Storage* menjadi solusi yang semakin populer untuk mengatasi masalah keamanan dan keterbatasan penyimpanan konvensional. Dengan menyimpan data di pusat data terkemuka, seperti *Google Cloud Platform* (GCP), risiko kehilangan data karena bencana alam berkurang, dan tingkat keamanan dapat ditingkatkan melalui enkripsi dan kontrol akses yang canggih.

*Google* adalah perusahaan yang memberikan platform sebagai layanan untuk perusahaan perusahaan lain di seluruh dunia. Salah satu produk layanan google adalah *Google Cloud Platform* (GCP). Setiap produk dan layanan yang disediakan memiliki serangkaian fitur dan manfaat yang berbeda sesuai dengan kebutuhan (Fristiani & Andryani, 2022). GCP menawarkan berbagai layanan *cloud* yang dapat dipilih sesuai dengan kebutuhan pengguna. Beberapa layanan populer termasuk *Google Cloud Storage*, yang menyediakan penyimpanan objek skalabel, dan *Google Drive*, yang cocok untuk kolaborasi berkas. Pemilihan *Google Cloud Platform* sebagai penyedia cloud storage dipandang sebagai

langkah yang cerdas karena reputasi keamanan dan reliabilitasnya. Selain itu, GCP menyediakan fitur-fitur canggih seperti enkripsi data *end-to-end* untuk memastikan keamanan data pengguna.

Penelitian ini akan menggunakan aplikasi *Vocaject* sebagai objek penelitian. *Vocaject* adalah aplikasi yang memiliki tujuan utama untuk membantu kampus dalam menerapkan *Project Based-Learning* kepada mahasiswanya. Selain itu, aplikasi ini terhubung dengan berbagai industri secara global, memungkinkan kampus untuk menjalin kerja sama dengan industri melalui proyek-proyek yang diunggah oleh industri. *Vocaject* juga memiliki peran dalam membantu mengelola proyek dan pengawasan proyek yang sedang dikerjakan.

Judul penelitian yang diusulkan adalah "Implementasi Penyimpanan Dokumen Menggunakan *Google Cloud Storage* pada Aplikasi *Vocaject*". Dengan judul ini, penelitian bertujuan untuk mengeksplorasi integrasi teknologi *cloud storage* dengan aplikasi *Vocaject* guna meningkatkan efisiensi, keamanan, dan keterjangkauan penyimpanan dokumen pada aplikasi tersebut.

## **1.2. Rumusan Masalah**

Berdasarkan latar belakang yang telah diuraikan, dapat dirumuskan permasalahan penelitian sebagai berikut:

- a. Dokumen yang diunggah dalam aplikasi *Vocaject* saat ini belum ada fitur keamanan untuk mencegah akses oleh pihak yang tidak memiliki otoritas terhadap dokumen tersebut.
- b. Dokumen yang diunggah dalam aplikasi *Vocaject* masih disimpan di sisi *server* sehingga beresiko kehilangan yang disebabkan oleh kerusakan atau terjadi serangan terhadap *server backend*.

## **1.3. Tujuan Penelitian**

Guna meningkatkan pelayanan yang dapat memuaskan bagi pengguna pada aplikasi *Vocaject*. Penelitian ini memiliki tujuan sebagai berikut:

- a. Menganalisis hasil pengujian aplikasi *Vocaject* terhadap akses keamanan dokumen setelah di implementasikan dengan *Cloud Storage*.
- b. Menganalisis hasil pengujian aplikasi *Vocaject* terhadap ketersediaan dokumen setelah dilakukan serangan pada *server backend*.

#### **1.4. Batasan Masalah**

Dalam rangka mencegah penyimpangan dan mempertegas pokok masalah penelitian, penulis menetapkan beberapa batasan masalah sebagai berikut:

- a. Penelitian ini difokuskan pada perancangan dan implementasi penyimpanan dokumen menggunakan *Cloud Storage* khusus untuk aplikasi *Vocaject*.
- b. Penelitian ini akan menggunakan data *dummy* yang dibuat berdasarkan struktur dan karakteristik data nyata sebagai simulasi.
- c. Penelitian ini akan menggunakan *Cloud Storage* dari *Google Cloud Platform* dalam implementasi.

#### **1.5. Manfaat Penelitian**

Melalui implementasi penelitian ini, penulis mengharapkan pengguna dapat memiliki pengalaman yang memuaskan dalam penggunaan aplikasi *Vocaject*. Harapan ini mencakup:

- a. Penelitian ini diharapkan dapat menciptakan lingkungan aplikasi yang lebih aman dan memuaskan bagi pengguna dengan mengatasi masalah kebocoran dan kehilangan data.
- b. Dengan menangani permasalahan keamanan seperti kebocoran dan kehilangan data, penelitian ini diharapkan dapat meminimalkan potensi risiko terhadap integritas dan kerahasiaan informasi pengguna.
- c. Penelitian ini diharapkan dapat memberikan kontribusi terhadap pemahaman praktis implementasi keamanan data dalam aplikasi atau sistem lainnya.



## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1. *State of the art***

Penyusunan penelitian ini mengambil beberapa referensi yang diperoleh dari artikel-artikel yang telah dipublikasikan melalui jurnal-jurnal yang ber ISSN dan mempunyai hubungan dengan penelitian ini sebagai acuan dan perbandingan dalam melakukan penelitian ini. Adapun pemaparan dari *State of the Art* dapat dilihat pada tabel 2.1 berikut.

Tabel 2.1 *State of The Art*

No	Penulis/Tahun	Judul Artikel	Metode yang digunakan	Hasil yang diperoleh	Persamaan	Perbedaan
1	Ketut Agus Seputra, A.A. Gede Yudhi Paramartha, I Nyoman Saputra Wahyu Wijaya / (2020) (Seputra et al, 2020)	Implementasi Google Drive Cloud Storage Pada Sistem Repositori AI-Daring	Metode Agile	Berdasarkan atas kuesioner SUS yang diberikan kepada responden diperoleh rata-rata skor 84.37 dengan nilai tertinggi 97.5 dan nilai terendah 67.5. Sesuai SUS <i>Performance Chart</i> , maka skor rata-rata SUS 84.37 memperoleh nilai <i>Excellent</i> untuk tingkat kegunaan, <i>grade B</i> untuk performa aplikasi, dan <i>Acceptable</i> bagi	Melakukan penerapan pembatasan hak akses sehingga yang memiliki wewenang yang dapat mengakses.	Pengguna yang telah melalui proses autentikasi memiliki akses langsung ke penyimpanan <i>Cloud</i> . Sedangkan pada penelitian ini, <i>backend</i> yang akan mengambil dokumen dari <i>cloud</i> dan diteruskan kembali ke

				pengguna.		pengguna.
2	Nopi Ramsari, Arif Ginanjar / (2022) (Ramsari, & Ginanjar 2022)	Implementasi Infrastruktur Server Berbasis Cloud Computing Untuk Web Service Berbasis Teknologi Google Cloud Platform	Prototyping	Perancangan infrastruktur <i>server</i> untuk <i>Web Service</i> dengan tingkat reliabilitas dan <i>availability</i> tinggi menggunakan Teknologi <i>Google Cloud Platform</i> berhasil diimplementasikan sehingga dapat menjaga reliabilitas dan <i>availability</i> dari infrastruktur <i>server</i> yang dibangun.	Memanfaatkan layanan <i>cloud storage</i> dari <i>google cloud platform</i> sebagai media penyimpanan terpisah dari <i>server</i> .	Hanya dirancang untuk membangun infrastruktur web <i>service</i> saja, tidak ada integrasi dengan suatu aplikasi.
3	Moch Kholil, Syahri Mu'min / (2018) (Kholil & Mu'min, 2018)	Pengembangan Private Cloud Storage sebagai Sentralisasi Data	Metode yang digunakan melalui beberapa	<i>Private Cloud Storage</i> dirancang untuk mengumpulkan, manajemen dan berbagi	Menggunakan <i>Cloud Storage</i> untuk menyelesaikan	Memanfaat kan <i>Owncloud</i> sebagai <i>Cloud Storage</i> , sedangkan

		Universitas Nahdlatul Ulama Sidoarjo Berbasis Open Source Owncloud	tahapan, antara lain: Analisis Kebutuhan, Desain Jaringan dan Implementasi	data antar user di Universitas Nahdlatul Ulama Sidoarjo	masalah yang sama yaitu keamanan data, keterbatasan penyimpanan dan resiko kehilangan data.	penelitian ini akan menggunakan <i>Cloud Storage</i> dari <i>Google Cloud Platform</i> .
--	--	--	--	---	---	--

## 2.2. Tinjauan Teoritis

Tinjauan teoritis adalah cara penulis menjelaskan topik penelitian. Bagian ini akan menjelaskan tentang alat, platform atau teknik yang digunakan dalam penelitian ini.

### 2.2.1. *Cloud*

*Cloud* (berarti "awan" dalam bahasa Inggris) atau sering dikenal dengan *Cloud Computing* merujuk pada layanan komputasi berbasis *internet* yang menyediakan penyimpanan, pemrosesan, dan akses data dan aplikasi melalui jaringan *internet*. *Cloud Computing* atau komputasi awan adalah kombinasi penggunaan teknologi komputer dan pengembangan berbasis *Internet*. *Cloud Storage* dapat dianggap sebagai representasi *internet* dalam bentuk penyimpanan data, yang sering digambarkan dalam diagram jaringan komputer. Dalam konteks *Cloud Computing*, *Cloud Storage* menjadi suatu abstraksi dari infrastruktur kompleks yang menyembunyikan kapabilitas teknologi informasi terkait. Ini disajikan sebagai layanan yang memungkinkan pengguna mengaksesnya melalui *internet* tanpa perlu mengetahui rincian teknis yang ada di baliknya (Zulkarnain, 2020).

*Cloud Computing* adalah suatu model yang memungkinkan akses ke jaringan secara nyaman, cepat, dan sesuai dengan permintaan atau kebutuhan. Ini melibatkan penggunaan kumpulan sumber daya komputasi yang disediakan oleh penyedia layanan dengan upaya manajemen interaksi yang terkoordinasi (Manalu et al. 2021). Sebagai pengganti bergantung pada perangkat keras lokal atau infrastruktur fisik, layanan *cloud* menggunakan pusat data yang tersebar di berbagai lokasi geografis dan diakses melalui *internet*. Pendekatan ini memungkinkan organisasi dan individu untuk menyimpan data mereka, menjalankan aplikasi, serta mengakses sumber daya komputasi tanpa perlu memiliki atau mengelola infrastruktur fisik secara langsung. Layanan *cloud* memberikan fleksibilitas, skala besar, dan akses mudah ke berbagai layanan tanpa perlu memelihara perangkat keras sendiri.



Gambar 2.1 *Cloud* (Manalu et al, 2021)

Terdapat tiga model layanan utama dalam *Cloud Computing* yaitu *IaaS*, *PaaS* dan *SaaS*. Berikut penjelasan model-model layanan *Cloud Computing*:

a. *IaaS (Infrastructure as a Service)*

*IaaS* menyediakan infrastruktur dasar seperti mesin virtual, penyimpanan data, dan jaringan melalui internet. Pengguna *IaaS* memiliki kontrol tinggi atas lingkungan mereka dan dapat mengelola sistem operasi, aplikasi, dan beban kerja.

Contoh Layanan: Amazon Web Services (AWS) EC2, Microsoft Azure Virtual Machines.

b. *PaaS (Platform as a Service)*

*PaaS* menyediakan platform lengkap untuk pengembangan, pengujian, dan implementasi aplikasi. Ini termasuk lingkungan pengembangan, *database*, dan alat pengembangan lainnya. Pengguna *PaaS* fokus pada pengembangan aplikasi tanpa perlu mengelola infrastruktur di bawahnya.

Contoh Layanan: Heroku, Google App Engine, Microsoft Azure App Services.

c. SaaS (*Software as a Service*)

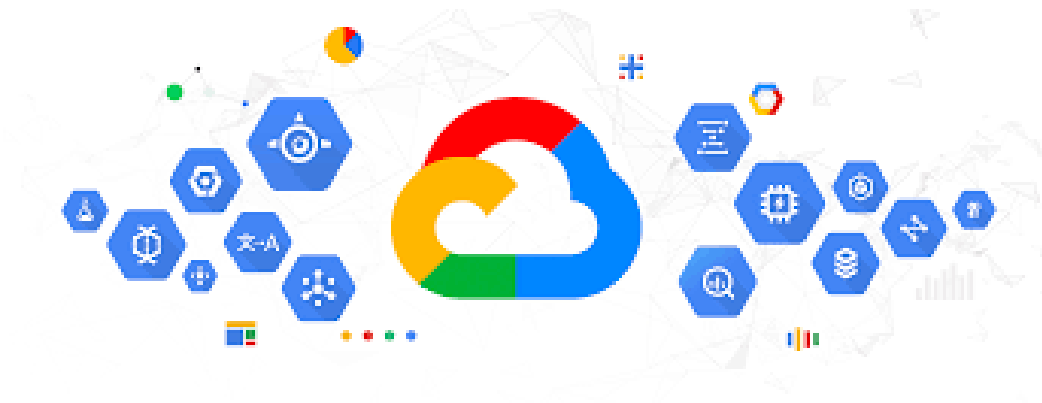
SaaS menyediakan akses ke perangkat lunak dan aplikasi melalui *internet*. Pengguna SaaS tidak perlu mengelola atau memelihara infrastruktur, karena aplikasi di hosting dan dikelola oleh penyedia layanan *cloud*.

Contoh Layanan: Google Workspace, Microsoft 365, Salesforce.

### 2.2.2. *Google Cloud Platform*

*Google Cloud Platform* (GCP) adalah salah satu dari tiga penyedia layanan cloud terbesar di dunia. GCP adalah kumpulan layanan *cloud* yang ditawarkan oleh Google untuk membantu organisasi dan individu menyimpan data, menjalankan aplikasi, dan mengakses berbagai sumber daya komputasi melalui infrastruktur *cloud* Google yang kuat dan tersebar di seluruh dunia.

Beberapa layanan yang dapat dimanfaatkan dari *Google Cloud Platform* (GCP) untuk merancang dan membangun infrastruktur *server* tersebut mencakup *Compute Engine*, *Cloud SQL*, *Cloud Storage*, dan *Container Registry* (Manalu et al, 2021). Produk-produk GCP ini menawarkan sejumlah fitur dan metode yang dapat diterapkan untuk membangun infrastruktur *server* yang memiliki ketersediaan tinggi. Beberapa diantaranya termasuk fitur *autohealing*, *multiple zones*, *load balancing*, *autoscaling*, *automatic updating*, dan *failover* (Sunaryo et al, 2019). Menerapkan layanan dari produk Teknologi GCP untuk membangun infrastruktur *server* diharapkan dapat menciptakan sistem yang memiliki tingkat keandalan dan ketersediaan tinggi, sehingga dapat mencapai tingkat *zero downtime* (Ray et al, 2019).



Gambar 2.2 *Google Cloud Platform* (Ramsari & Ginanjar, 2022)

#### 2.2.2.1. Keunggulan *Google Cloud Platform*

*Google Cloud Platform* (GCP) merupakan layanan *cloud* yang menawarkan begitu banyak keuntungan dan manfaat bagi pelanggannya. Berikut beberapa keunggulan dari GCP:

a. Infrastruktur Global

GCP memiliki pusat data global yang tersebar di berbagai wilayah, memungkinkan pelanggan untuk menjalankan aplikasi mereka di dekat pengguna akhir dan mengoptimalkan kinerja.

b. Keamanan

GCP menempatkan penekanan besar pada keamanan. Mereka menawarkan berbagai fitur keamanan, termasuk enkripsi data dalam perpindahan dan istirahat, keamanan identitas, dan alat keamanan lanjutan.

c. Fleksibilitas dan Skalabilitas

GCP menyediakan infrastruktur yang sangat fleksibel dan dapat diskalakan, memungkinkan pelanggan untuk menyesuaikan sumber daya sesuai kebutuhan mereka.



d. Integrasi dengan layanan Google

GCP terintegrasi dengan layanan Google lainnya seperti *Google Workspace*, *Google Maps*, dan YouTube, menyediakan solusi *end-to-end* untuk berbagai kebutuhan bisnis.

#### 2.2.2.2. Kekurangan *Google Cloud Platform*

*Google Cloud Platform* (GCP) juga terdapat beberapa kekurangan, berikut beberapa kekurangan GCP:

a. Keterbatasan dalam Ekosistem Pelanggan

GCP memiliki basis pengguna yang lebih kecil dibandingkan dengan beberapa pesaingnya seperti AWS dan Azure, sehingga mungkin kurangnya beberapa aplikasi dan layanan pihak ketiga.

b. Harga

Meskipun *Google Cloud* menawarkan beberapa diskon dan kebijakan harga yang fleksibel, beberapa pelanggan melaporkan bahwa biaya GCP dapat menjadi lebih tinggi dibandingkan dengan penyedia *cloud* lainnya, terutama untuk beberapa layanan.

c. Kurangnya Beberapa Fitur Enterprise

Beberapa pelanggan mencatat bahwa beberapa fitur *enterprise* yang dimiliki oleh pesaing utama mungkin kurang pada GCP. Namun, ini terus berubah karena Google terus memperbaiki dan meningkatkan layanannya.

d. Kompleksitas

Beberapa pengguna mencatat bahwa antarmuka GCP dapat terasa kompleks bagi pengguna baru, dan kurva pembelajarannya mungkin lebih curam dibandingkan dengan penyedia *cloud* lain.

### 2.2.3. *Cloud Storage*

*Cloud Storage* adalah layanan penyimpanan data yang memungkinkan pengguna untuk menyimpan, mengakses, dan mengelola data mereka di *server* jarak jauh yang dioperasikan oleh penyedia layanan *cloud*. Penyimpanan *cloud* ini memungkinkan pengguna untuk menyimpan berbagai jenis data, seperti *file*, dokumen, gambar, *video*, dan *database*, di internet. Beberapa keuntungan dari *cloud storage* meliputi:

a. Akses Mudah

Data dapat diakses kapan saja dan dari mana saja selama ada koneksi internet.

b. Skalabilitas

Pengguna dapat menambah atau mengurangi kapasitas penyimpanan sesuai kebutuhan tanpa harus membeli perangkat keras tambahan.

c. Keamanan

Penyedia layanan cloud biasanya menawarkan fitur keamanan canggih, termasuk enkripsi data, otentikasi multi-faktor, dan backup otomatis.

d. Kolaborasi

Cloud storage memungkinkan banyak pengguna untuk berbagi dan mengedit file secara bersamaan, yang meningkatkan efisiensi dan produktivitas.

e. Biaya Efektif

Mengurangi kebutuhan untuk membeli dan memelihara perangkat keras penyimpanan sendiri, karena pengguna hanya membayar untuk kapasitas yang mereka gunakan.

Contoh layanan *cloud storage* yang populer adalah Google Cloud Storage, Google Drive, Dropbox, Microsoft OneDrive, dan Amazon S3.

#### 2.2.4. Google Cloud Storage

*Cloud Storage* merupakan layanan penyimpanan dalam bentuk *object* di GCP. Penyimpanan objek atau *object storage* juga sering disebut sebagai penyimpanan berbasis objek, adalah arsitektur penyimpanan data untuk menangani data tidak terstruktur dalam jumlah besar.

*Google Cloud Storage* adalah layanan penyimpanan berbasis *cloud* yang ditawarkan oleh Google sebagai bagian dari *Google Cloud Platform* (GCP). Layanan ini memungkinkan pengguna untuk menyimpan berbagai jenis data, termasuk file, gambar, video, dan data yang besar, di infrastruktur *cloud Google* yang aman dan *scalable*.



Gambar 2.3 *Google Cloud Storage* (Ramsari & Ginanjar, 2022)

*Google Cloud Storage* memiliki beberapa kelas penyimpanan yang dirancang untuk memenuhi kebutuhan dan persyaratan spesifik, diantaranya:

a. *Standard Storage Class*:

*Standard Storage Class* merupakan kelas penyimpanan default yang cocok untuk berbagai jenis data. Ini memberikan akses tinggi, kinerja tinggi, dan durabilitas data yang tinggi. Cocok untuk data yang sering diakses.

b. *Coldline Storage Class*:

*Coldline Storage Class* adalah kelas penyimpanan yang dirancang untuk data yang jarang diakses, tetapi perlu diakses dengan cepat ketika dibutuhkan.

Harganya lebih rendah daripada *Standard Storage Class*, namun ada biaya tambahan untuk mengambil data.

c. *Nearline Storage Class*:

*Nearline Storage Class* cocok untuk data yang diakses kurang dari sekali sebulan, tetapi jika diakses, diperlukan akses cepat. Ini menawarkan biaya penyimpanan yang lebih rendah daripada *Standard Storage Class*, tetapi ada biaya tambahan untuk mengambil data.

d. *Archive Storage Class*:

*Archive Storage Class* cocok untuk data yang sangat jarang diakses, dengan waktu pengambilan yang lambat. Ini merupakan kelas penyimpanan dengan biaya penyimpanan terendah, tetapi ada biaya pengambilan dan waktu pengambilan yang lebih lama.

#### 2.2.5. *Compute Engine*

*Google Compute Engine* (GCE) adalah layanan infrastruktur *cloud* dari *Google Cloud Platform* yang menyediakan mesin virtual yang berjalan di pusat data *Google*. Layanan ini menawarkan skalabilitas tinggi, memungkinkan pengguna untuk menambah atau mengurangi jumlah instance sesuai kebutuhan. Dengan performa tinggi dan berbagai jenis mesin virtual yang dapat disesuaikan, GCE mendukung berbagai kebutuhan beban kerja. Selain itu, GCE dilengkapi dengan fitur keamanan seperti enkripsi data, *firewall*, dan manajemen identitas serta akses, memastikan data dan aplikasi pengguna tetap aman.

#### 2.2.6. *Google Load Balancing*

*Google Cloud Load Balancing* adalah layanan yang mendistribusikan lalu lintas jaringan ke berbagai *instance* di *Google Cloud Platform* untuk memastikan ketersediaan tinggi dan kinerja optimal aplikasi. Layanan ini mendukung berbagai jenis *load balancing*, termasuk HTTP(S), TCP/SSL, dan UDP, memungkinkan pengguna untuk menangani lalu lintas web, aplikasi, dan layanan lainnya secara efisien. Layanan ini juga mendukung integrasi dengan layanan keamanan seperti

Cloud Armor untuk melindungi aplikasi dari serangan DDoS. Dengan *Google Cloud Load Balancing*, pengguna dapat memastikan aplikasi mereka tetap responsif, handal, dan aman di lingkungan *cloud*.

#### 2.2.7. Laravel

Laravel adalah kerangka kerja PHP open-source yang dirancang untuk memudahkan pengembangan aplikasi web dengan sintaks yang ekspresif dan elegan. Dibuat oleh Taylor Otwell, Laravel menyederhanakan tugas-tugas umum seperti routing, otentikasi, caching, dan sesi, sehingga pengembang dapat lebih fokus pada logika bisnis aplikasi. Fitur-fitur utamanya termasuk Artisan CLI untuk otomatisasi tugas, Eloquent ORM untuk interaksi basis data yang elegan, Blade templating untuk sistem templating yang fleksibel, serta sistem routing dan middleware yang intuitif. Laravel juga menyediakan fasilitas unit testing bawaan untuk memastikan aplikasi tetap berkualitas tinggi. Dengan kemudahan penggunaan dan fleksibilitasnya, Laravel telah menjadi salah satu framework PHP paling populer dan banyak digunakan untuk membangun aplikasi web yang kuat dan scalable.



Gambar 2.4 Laravel (Baraka, 2023)

#### 2.2.8. *Distributed Denial of Service (DDoS)*

*Serangan Denial of Service (DoS)* atau *Distributed Denial of Service (DDoS)* adalah jenis serangan yang bertujuan untuk meng *overload server* dengan jumlah permintaan yang sangat besar. Hal ini dilakukan dengan cara mengirimkan

sejumlah besar permintaan sehingga menyebabkan *server* menggunakan semua sumber daya yang tersedia, sehingga *server* tersebut tidak dapat menjalankan fungsi dan tugasnya dengan efisien. Akibatnya, *server* yang terbebani tidak mampu memberikan layanan secara normal dan mengalami penolakan layanan (*denial of service*) (Hijriyanto & Ulum, 2021).

#### 2.2.9. *Stress Testing*

*Stress testing* adalah proses evaluasi kinerja dan ketahanan suatu sistem atau aplikasi web dalam menghadapi kondisi ekstrim, seperti beban kerja yang sangat tinggi. Salah satu aplikasi dari stress testing dalam bidang TI adalah simulasi serangan DDoS (*Distributed Denial of Service*), di mana sistem diuji untuk mengetahui kemampuannya bertahan dan beroperasi saat menerima volume *traffic* yang tidak biasa dan besar. Melalui *stress testing*, organisasi dapat mengidentifikasi kelemahan dalam infrastruktur mereka dan memperkuatnya agar lebih tahan terhadap serangan yang dapat mengganggu layanan atau aksesibilitas.

#### 2.2.10. *Postman*

Postman adalah platform GUI yang handal yang mempermudah dan mempercepat pengembangan API, mulai dari pembuatan, pengujian, dokumentasi, hingga berbagi API. Postman direkomendasikan untuk digunakan pada sistem operasi *Mac*, *Windows*, atau *Linux* (Arifin & Laya, 2019). Dengan Postman, pengguna dapat membuat berbagai permintaan HTTP (GET, POST, PUT, DELETE, dll.), menambahkan parameter, header, dan badan permintaan, serta memeriksa respons dari server.

Penggunaan Postman dalam proses pengembangan API membantu meningkatkan efisiensi dan akurasi pengujian, mengurangi kesalahan manusia, dan mempercepat proses debugging. Aplikasi ini juga memungkinkan kolaborasi yang lebih baik antar tim pengembang melalui fitur berbagi dan sinkronisasi koleksi permintaan.

### 2.2.11. *Apache JMeter*

Apache JMeter adalah aplikasi open-source yang digunakan untuk menguji performa dan memantau perilaku berbagai jenis aplikasi, terutama aplikasi web. JMeter adalah aplikasi pengujian kinerja yang fleksibel yang mampu mensimulasikan banyak pengguna secara bersamaan, sehingga memungkinkan evaluasi kinerja aplikasi (Ismail et al, 2023). JMeter dirancang untuk melakukan pengujian beban dengan mensimulasikan sejumlah besar pengguna atau permintaan secara bersamaan untuk menilai kinerja aplikasi di bawah beban yang tinggi.

Aplikasi ini mendukung berbagai protokol seperti HTTP, HTTPS, FTP, JDBC, dan banyak lainnya, menjadikannya alat yang serbaguna untuk pengujian aplikasi yang kompleks. Dengan antarmuka grafis yang intuitif dan kemampuan untuk membuat skrip pengujian yang kompleks, JMeter memungkinkan penguji untuk membuat skenario pengujian yang realistis dan menganalisis hasilnya dengan mudah.

### 2.2.12. *Wireshark*

*Wireshark* adalah alat yang digunakan untuk menganalisis paket data di jaringan internet. *Wireshark* termasuk dalam kategori *Network Packet Analyzer*, yang berfungsi menangkap semua informasi data selama komunikasi jaringan dan menampilkan informasi tersebut dengan sangat detail (Luthfansa & Rosiani, 2021). Aplikasi ini memungkinkan pengguna untuk melihat semua lalu lintas data yang melewati jaringan secara *real-time*, yang sangat berguna untuk mendeteksi masalah jaringan, *troubleshooting*, dan keamanan jaringan. *Wireshark* dapat menangkap paket data dari berbagai jenis protokol, seperti TCP, UDP, HTTP, dan banyak lagi, serta menampilkan data tersebut dalam format yang mudah dibaca dan dipahami. Dengan fitur filter yang kuat, pengguna dapat memfokuskan analisis pada segmen tertentu dari lalu lintas jaringan, memudahkan identifikasi isu spesifik.

*Wireshark* menyediakan berbagai alat untuk menganalisis performa jaringan dan mendeteksi anomali, seperti paket yang hilang atau keterlambatan jaringan. Aplikasi ini juga mendukung dekripsi untuk beberapa protokol, memungkinkan analisis yang lebih mendalam terhadap data yang dienkripsi.

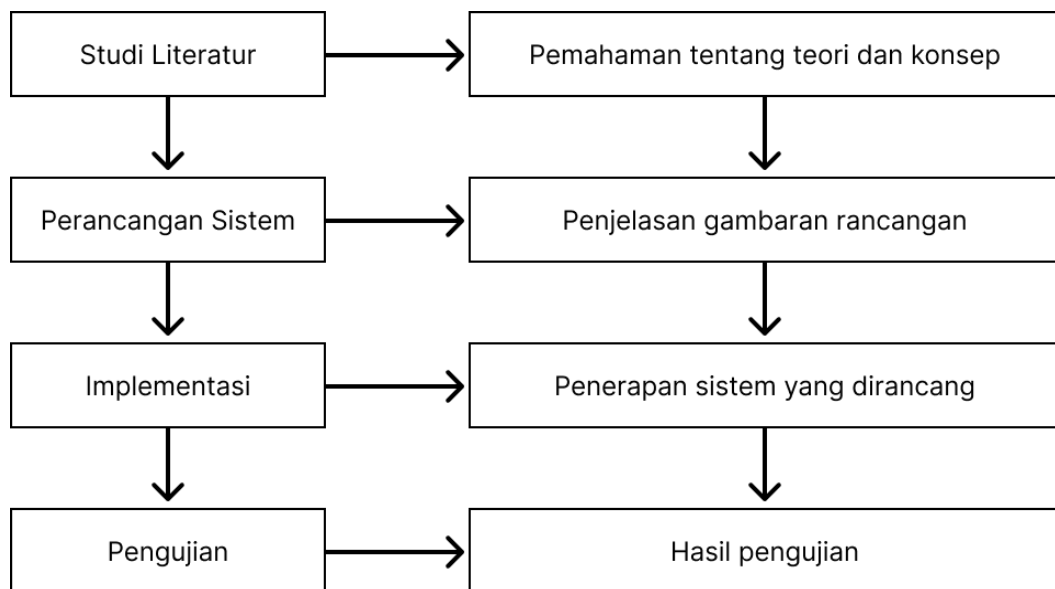


## BAB III

### METODOLOGI PENELITIAN

#### 3.1. Tahapan Penelitian

Untuk melaksanakan implementasi penyimpanan dokumen menggunakan *Cloud Storage*, diperlukan susunan kerja yang merupakan tahapan-tahapan penyelesaian yang dibahas. Kerangka kerja dalam penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1 Tahapan Penelitian

a. Studi Literatur

Untuk mendukung penelitian ini, penulis akan melakukan studi literatur dengan mengumpulkan bahan dari berbagai sumber, termasuk media internet, jurnal ilmiah, buku, dan beberapa referensi lainnya.

b. Perancangan Sistem

Perancangan sistem merupakan aktivitas menganalisis dan mendesain suatu sistem yang efektif mengikuti referensi yang didapatkan pada tahapan studi

literatur. Rancangan sistem digunakan sebagai bentuk gambaran dari sistem yang akan dikembangkan.

c. Implementasi

Implementasi merupakan tindakan untuk melaksanakan pengembangan sistem yang sudah dirancang serta mempersiapkan peralatan untuk melakukan pengujian.

d. Pengujian

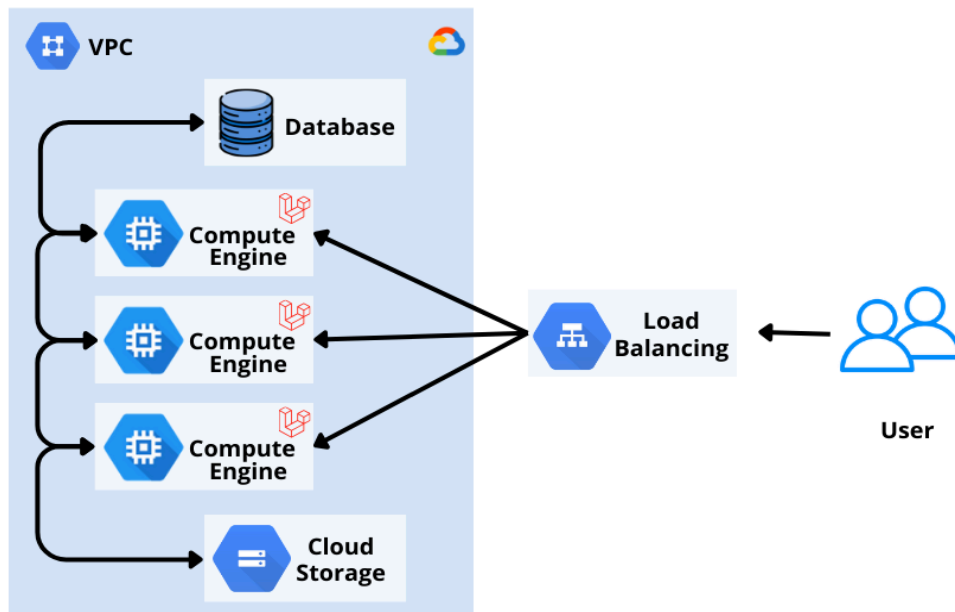
Pengujian merupakan tindakan pengetesan sistem yang telah dikembangkan untuk memastikan sistem tersebut sudah berjalan sesuai dengan rancangan serta hasil pengujian dapat dianalisa untuk dijadikan simpulan.

### 3.2. Perancangan Sistem

Perancangan ini digunakan sebagai penjelasan mengenai gambaran sistem yang akan dibangun.

#### 3.2.1. Perancangan Arsitektur *Deployment*

Rancangan arsitektur *deployment* merupakan gambaran perencanaan dan penataan komponen-komponen sumber daya untuk mendistribusikan *backend* pada lingkungan produksi termasuk *server*, jaringan, dan infrastruktur lainnya. Ilustrasi rancangan arsitektur *deployment* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Ilustrasi Arsitektur *Deployment*

Dalam ilustrasi pada Gambar 3.2, rancangan arsitektur deployment ini memanfaatkan beberapa sumber daya dari *Google Cloud Platform*. Berikut penjelasan mengenai komponen-komponen yang terdapat dalam ilustrasi tersebut:

a. *Instance Backend*

Pada ilustrasi tersebut, terdapat tiga instance mesin virtual yang digunakan untuk menjalankan *server backend*. Setiap *instance* akan diinstalasi dengan *backend* yang dikembangkan dalam penelitian ini. *Compute engine*, yang merupakan layanan infrastruktur *cloud*, memungkinkan untuk membuat dan menjalankan mesin virtual dengan spesifikasi yang dapat disesuaikan sesuai kebutuhan.

Setiap *instance* akan di-deploy dengan *backend* melalui proses mengkloning proyek dari *repository* GitHub ke masing-masing *instance*. Selain mengkloning proyek *backend*, setiap *instance* juga akan mengunduh *credentials* untuk *cloud storage* serta *environment* proyek yang akan diunggah ke *Cloud Storage*. Seperti yang telah dijelaskan pada poin 3.2.2 tentang perancangan sistem, *backend* dalam penelitian ini

dikembangkan menggunakan *framework* PHP, yaitu Laravel. *Backend* yang di-*deploy* tersebut akan digunakan oleh *frontend*, baik untuk web maupun *mobile*.

b. *Instance Database*

Selain ketiga *instance compute engine* yang digunakan sebagai *server backend*, terdapat juga satu *instance* yang digunakan sebagai *server database*. *Instance* ini diperlukan agar semua *instance backend* dapat terhubung dengan satu *server database* yang sama. *Database* yang digunakan dalam penelitian ini adalah MySQL. Dengan adanya *instance database* yang terpusat, dapat memastikan konsistensi data, memudahkan pengelolaan, serta meningkatkan kinerja dan skalabilitas sistem secara keseluruhan.

c. *Cloud Storage*

Sesuai dengan tujuan penelitian ini, pada ilustrasi tersebut terdapat *cloud storage* yang akan digunakan sebagai media penyimpanan pada aplikasi *Vocaject*. Setiap dokumen atau *file* dalam *cloud storage* disebut sebagai objek yang nantinya akan disimpan dalam suatu *bucket*. *Bucket* yang akan digunakan dalam penelitian ini akan bersifat *private*, sehingga hanya yang memiliki *credentials* saja yang dapat mengakses *bucket* tersebut.

Dalam penelitian ini, akan dibuatkan *service account* agar *credentials*-nya dapat diunduh dan disimpan di sisi *server backend*. Hal ini memungkinkan *backend* untuk memanfaatkan *credentials* tersebut guna memiliki otoritas penuh dalam mengelola *bucket*. Dengan cara ini, *backend* dapat melakukan berbagai operasi seperti menyimpan, mengambil, dan menghapus objek dalam *bucket* dengan aman dan efisien. Keamanan data juga terjamin karena akses dibatasi hanya untuk pihak yang memiliki *credentials* yang sah.

d. *Load Balancer*

*Load Balancer* merupakan sumber daya yang berguna untuk mendistribusikan lalu lintas (*traffic*) ke setiap *instance* yang terintegrasi. Dengan memanfaatkan *Load Balancer*, dapat meminimalisir risiko kegagalan terhadap ketersediaan data, termasuk dokumen, yang dapat diakibatkan oleh lonjakan lalu lintas, kerusakan pada salah satu *instance*, serangan, atau risiko lainnya.

Pada penelitian ini, ketiga *instance* backend akan diintegrasikan dengan *Load Balancer* untuk menangani masalah-masalah tersebut. Dengan demikian, *Load Balancer* akan membantu menjaga ketersediaan dan kinerja aplikasi dengan mendistribusikan lalu lintas secara merata ke setiap *instance backend*. Selain itu, penggunaan *Load Balancer* juga memungkinkan sistem untuk dapat secara otomatis menyesuaikan kapasitasnya sesuai dengan kebutuhan, sehingga dapat mengoptimalkan penggunaan sumber daya yang tersedia. *Load Balancer* juga akan dikonfigurasi *ssl* agar dapat diakses melalui protokol *HTTPS* sehingga dapat meningkatkan tingkat keamanan dengan mengenkripsi data saat proses komunikasi antara pengguna dan *server*.

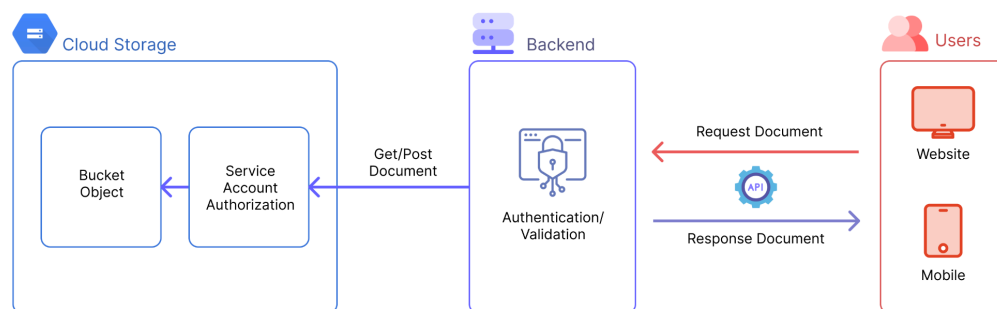
e. *Users*

*Users* adalah pengguna akhir yang mengakses aplikasi *Vocaject*. Dari ilustrasi tersebut, setiap permintaan pengguna akan melewati *Load Balancer* terlebih dahulu. *Load Balancer* kemudian akan mendistribusikan setiap permintaan pengguna ke salah satu *instance backend* yang dapat menangani permintaan tersebut. Jika permintaan dari pengguna membutuhkan data, *backend* akan berkomunikasi dengan *cloud storage* atau *database*. Setelah permintaan pengguna dioperasikan di *backend*, *backend* akan mengembalikan *respons* dalam bentuk *JSON* atau *file*. *Respons* tersebut kemudian akan diproses lebih lanjut oleh *frontend* untuk ditampilkan sebagai informasi atau sebagai *file* unduhan bagi pengguna. Proses ini memastikan bahwa aplikasi dapat menangani permintaan

pengguna dengan efisien dan responsif, serta menjaga pengalaman pengguna tetap optimal. Selain itu, dengan adanya *Load Balancer*, aplikasi dapat tetap berfungsi dengan baik meskipun terdapat lonjakan lalu lintas atau kegagalan pada salah satu instance *backend*.

### 3.2.2. Perancangan Sistem

Untuk menyelesaikan masalah keterbatasan penyimpanan, resiko akses pihak yang tidak memiliki wewenang dan kerusakan perangkat keras *server*, perlu dirancang sebuah sistem yang dapat memastikan pelaksanaan penelitian ini dilakukan dengan tepat agar tujuan penelitian dapat tercapai. Rancangan sistem yang akan digunakan dalam penelitian ini akan diilustrasikan dalam gambar 3.3.



Gambar 3.3 Ilustrasi rancangan sistem

Pada gambar tersebut terdapat tiga komponen yaitu *Users*, *Backend* dan *Cloud Storage*. Masing-masing komponen tersebut mewakili area yang harus dilalui untuk melakukan proses pengunggahan dokumen atau proses mengakses suatu dokumen.

Proses untuk mengunggah atau mengakses suatu dokumen dimulai dari interaksi pengguna yang dilakukan melalui *Website* atau perangkat *Mobile*. Interaksi tersebut dapat berbentuk mengunggah suatu dokumen, mengunduh suatu dokumen atau membuka suatu dokumen. Interaksi dari pengguna tersebut akan membentuk suatu permintaan atau *Request* ke *Server* atau pada ilustrasi tersebut

digambarkan sebagai komponen *Backend*. Request tersebut akan berjalan melalui Rest API menggunakan protokol HTTP/HTTPS yang telah dibangun di *Backend*.

Selanjutnya, saat *Request* diterima oleh *Backend*, *Backend* akan melakukan proses validasi pengguna terhadap ketersediaan akun dan otoritas dokumen yang ingin diakses. Bila *Request* yang diterima adalah permintaan untuk mengunggah suatu dokumen, *Backend* hanya akan melakukan validasi ketersediaan akun pengguna serta mengidentifikasi dokumen yang akan diunggah ke *Cloud Storage*.

Jika pengguna berhasil tervalidasi, *Backend* akan melakukan pengambilan dokumen yang minta oleh pengguna dari *Cloud Storage* atau mengunggah dokumen yang dikirimkan oleh pengguna ke *Cloud Storage*. Proses ini dibutuhkan kredensial untuk dapat mengakses *Bucket* pada *Cloud Storage*. Pada penelitian ini nantinya, *Backend* akan dibuatkan kredensial *Service Account* yang memiliki hak akses terhadap *Bucket* yang digunakan untuk penelitian ini yang akan diintegrasikan pada *Backend*. Dengan demikian, *Backend* dapat mengakses *Bucket* untuk mengambil atau mengunggah dokumen sesuai kebutuhan pengguna.

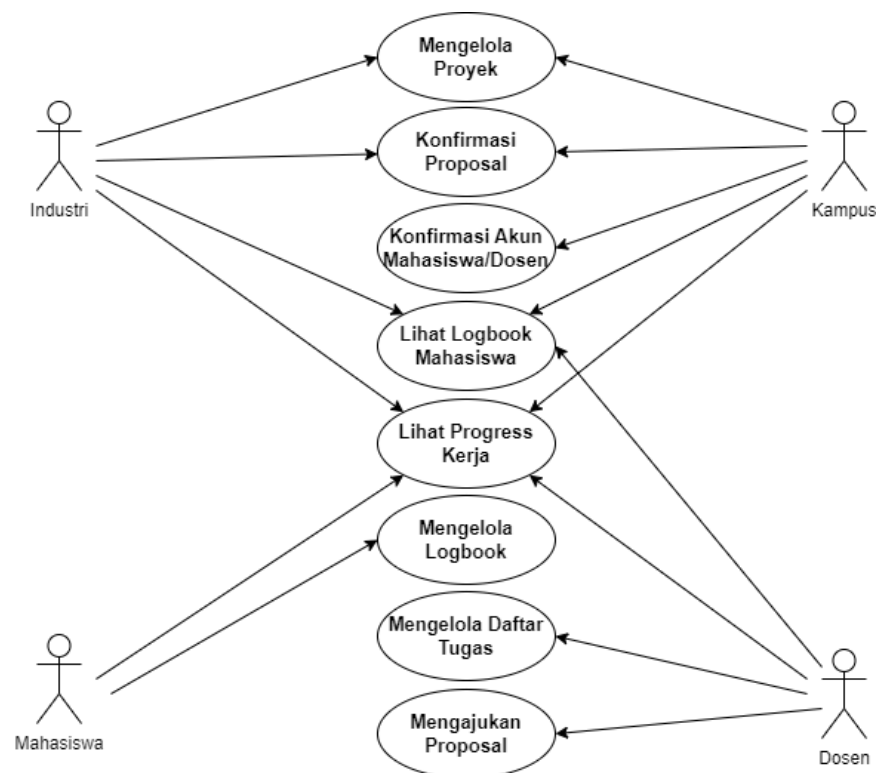
Setelah *Backend* melakukan mengambil dokumen dari *Cloud Storage* atau mengunggah dokumen ke *Cloud Storage* sesuai kebutuhan pengguna, *Backend* akan memberi tanggapan atau *Response* ke pengguna dalam bentuk pesan *Json* atau dokumen pula. *Response* tersebut nanti akan dioperasikan oleh *Frontend* baik *Website* maupun *Mobile* sesuai dengan interaksi pengguna.

### 3.2.3. Perancangan *Diagram Use Case*

Untuk menjelaskan bagaimana interaksi pengguna terhadap aplikasi *Vocaject*, akan digambarkan dalam bentuk *Diagram Use Case*. *Diagram Use Case* (*Use Case Diagram*) adalah jenis diagram yang digunakan dalam rekayasa perangkat lunak untuk mendeskripsikan fungsionalitas suatu sistem dari sudut pandang pengguna. Diagram ini membantu dalam mengidentifikasi, menggambarkan, dan mengorganisir berbagai kasus pengguna (*use case*) yang dapat terjadi dalam sistem. *Diagram Use Case* melibatkan aktor (*actor*), *use case*,

dan hubungan antara keduanya. Aktor adalah entitas eksternal yang berinteraksi dengan sistem, sementara *use case* adalah deskripsi skenario fungsionalitas sistem. Hubungan antara aktor dan *use case* menggambarkan bagaimana aktor berinteraksi dengan *use case* tertentu. Diagram ini memberikan gambaran tingkat tinggi tentang fungsionalitas sistem dan membantu dalam memahami interaksi antara pengguna dan sistem. *Diagram Use Case* sering digunakan dalam fase analisis dan perancangan sistem untuk mengidentifikasi dan merancang fungsionalitas yang dibutuhkan oleh pengguna.

Aplikasi *Vocaject* memiliki interaksi yang sangat kompleks, dengan menggunakan *Diagram Use Case*, sistem akan digambarkan secara umum, sehingga mudah untuk dipahami. Gambaran *Diagram Use Case* aplikasi *Vocaject* diilustrasikan pada gambar 3.4.



Gambar 3.4 *Diagram Use Case* aplikasi *Vocaject*



Pada Gambar 3.4, terdapat 4 aktor yang menjadi pengguna utama pada aplikasi *Vocaject* yaitu Industri, Kampus, Dosen dan Mahasiswa. Setiap aktor memiliki peran atau fungsionalitas masing-masing. Dalam aplikasi *Vocaject*, Industri dan Kampus sangat berperan penting dalam sistem ini. Industri dan Kampus memiliki peran yang sama yaitu mengelola suatu proyek. Peran tersebut akan menjadi dasar dari peran-peran lain yang akan terlibat. Proyek-proyek yang diunggah ke aplikasi juga akan menjadi dasar peran-peran lain baik dari Industri, Kampus atau aktor lainnya. Sebagai pengelola proyek, Industri dan Kampus juga memiliki peran untuk mengkonfirmasi proposal-proposal yang diajukan oleh Dosen. Walaupun Industri dan Kampus memiliki peran yang sama, terdapat perbedaan diantara keduanya, Kampus memiliki peran tersendiri terhadap validasi atau mengkonfirmasi Dosen atau Mahasiswa yang mendaftarkan akun atas nama Kampus tersebut. Selain itu, Industri dan Kampus juga dapat melihat atau mengawasi progres kerja dan *Logbook* mahasiswa. Namun, yang perlu diperhatikan bahwa Kampus tidak memiliki akses terhadap proyek yang melibatkan Dosen dan Mahasiswanya kecuali proyek tersebut berasal dari Kampus itu sendiri.

Dosen disini memiliki peran aktif dalam memulai hingga menyelesaikan proyek-proyek yang dikerjakan oleh Mahasiswa. Oleh karena itu, Dosen memiliki peran dalam proses pengajuan proposal ke proyek-proyek yang telah diunggah oleh Industri maupun Kampus. Selain itu, Dosen juga memiliki peran dalam mengelola daftar tugas. Daftar tugas yang dikelola oleh dosen akan menjadi target-target mahasiswa dalam pengerjaan suatu proyek. Sama seperti Industri dan Kampus, Dosen memiliki hak untuk melihat daftar *Logbook* mahasiswa, yang akan dijadikan sebagai bahan *review* untuk memvalidasi daftar tugas yang sudah dibuat oleh dosen tersebut.

Peran Mahasiswa dalam aplikasi ini sangatlah sederhana, Mahasiswa hanya dapat melihat progres dari proyek yang dikerjakan dan membuat laporan berupa *Logbook* baik setiap hari atau jadwal yang ditentukan oleh Dosen. *Logbook* ini yang nantinya akan menjadi dasar dari progres proyek tersebut.

### 3.3. Metode Pengujian

Dalam tahap pengujian ini, penulis akan menerapkan beberapa tahapan untuk menguji tingkat keamanan sebagai tolak ukur keberhasilan implementasi *Cloud Storage* untuk mengamankan dokumen pada aplikasi *Vocaject*. Berikut beberapa tahapan yang akan dilakukan:

a. Pengujian Fungsional

Pengujian fungsional bertujuan untuk memastikan sistem berhasil mengunggah dan mengunduh dokumen ke dan dari cloud storage. Pengujian ini mencakup frontend dan backend, melibatkan berbagai jenis file seperti gambar, video, dan file lainnya. Pengujian ini akan dilakukan dengan memeriksa antarmuka pengguna untuk memastikan kemudahan pengunggahan dan pengunduhan dokumen. Hasil pengujian akan menunjukkan keberhasilan integrasi sistem dengan cloud storage sesuai dengan rancangan sistem yang telah dibuat.

b. Pengujian Keamanan Akses

Pengujian keamanan akses bertujuan untuk memastikan bahwa pengguna tidak dapat mengakses dokumen tanpa wewenang yang sesuai. Ini mencakup pengujian terhadap lapisan keamanan, kontrol akses, dan pencegahan akses yang tidak sah. Hasil pengujian ini menunjukkan seberapa efektif sistem dalam melindungi dokumen dari akses yang tidak diizinkan.

c. Pengujian *Sniffing*

Pengujian sniffing bertujuan untuk memastikan bahwa suatu dokumen tidak dapat dibaca di jaringan selama proses pengunggahan dan pengunduhan. Dalam pengujian ini, paket-paket data yang berjalan di jaringan akan ditangkap menggunakan aplikasi *Wireshark*. Dokumen akan dinyatakan aman jika tidak dapat dideteksi atau dibaca melalui aplikasi tersebut. Pengujian sniffing juga membantu dalam mengidentifikasi apakah ada celah keamanan dalam proses pengiriman data, seperti penggunaan protokol yang rentan terhadap serangan *sniffing*.

d. Pengujian Ketersediaan

Pengujian ketersediaan bertujuan untuk memastikan bahwa dokumen tetap dapat diakses saat *server* mengalami gangguan atau kelemahan. Dalam pengujian ini, penulis akan melakukan serangan *Distributed Denial of Service* (DDoS) tepatnya *Stress Testing* menggunakan aplikasi JMeter dengan mengirimkan sejumlah permintaan dokumen untuk menguji kemampuan *server* dalam menangani permintaan pengguna yang banyak. Hasil pengujian ini memberikan gambaran sejauh mana sistem dapat memberikan layanan akses yang berkelanjutan kepada pengguna, terutama dalam menghadapi serangan seperti DDoS.

### 3.4. Hasil yang diharapkan

Dengan menerapkan penyimpanan dokumen menggunakan *Google Cloud Storage* dan implementasi sistem otentikasi pada aplikasi *Vocaject*, diharapkan dapat mengatasi beberapa masalah yang telah diidentifikasi. Ini termasuk keterbatasan penyimpanan, potensi kerusakan perangkat keras pada *server*, dan risiko akses oleh pihak yang tidak berwenang. Dengan langkah-langkah ini, diharapkan *Vocaject* menjadi lebih handal dalam mengelola data dan memberikan perlindungan terhadap kebocoran dan kehilangan data pengguna. Sebagai hasilnya, aplikasi ini diharapkan dapat digunakan dengan aman dan dapat diakses oleh seluruh pengguna dengan jaminan keamanan data yang maksimal.

## BAB IV

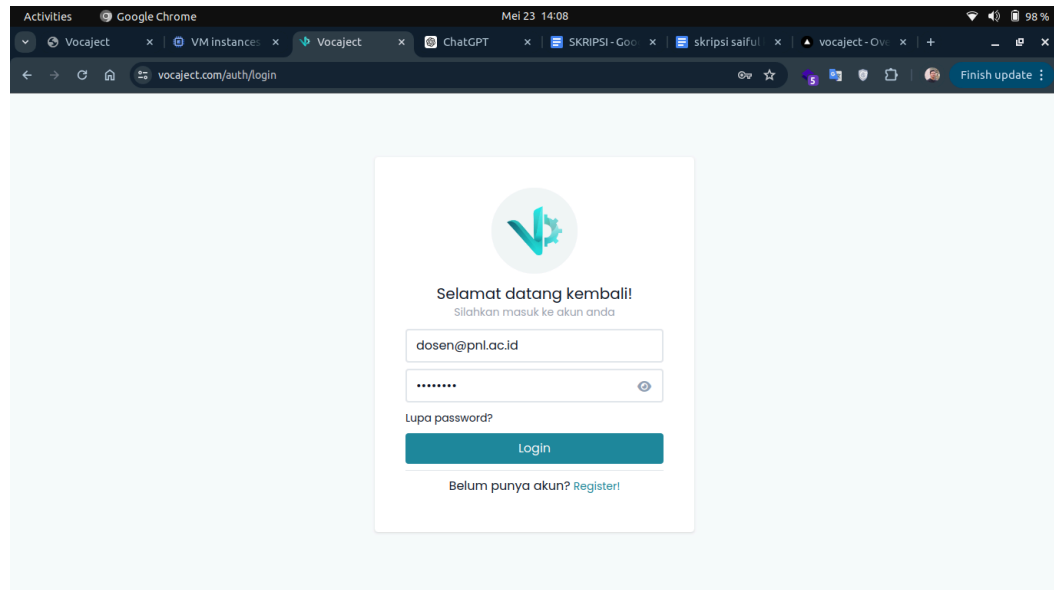
### HASIL DAN PEMBAHASAN

#### 4.1. Tampilan *Website Vocaject*

Pengembangan tampilan dari website *Vocaject* mengikuti desain yang telah dirancang. Pada bagian ini akan dijelaskan setiap halaman dalam tahapan mengunggah suatu dokumen melalui fitur *Chat* atau diskusi proyek.

##### 4.1.1. Halaman Login

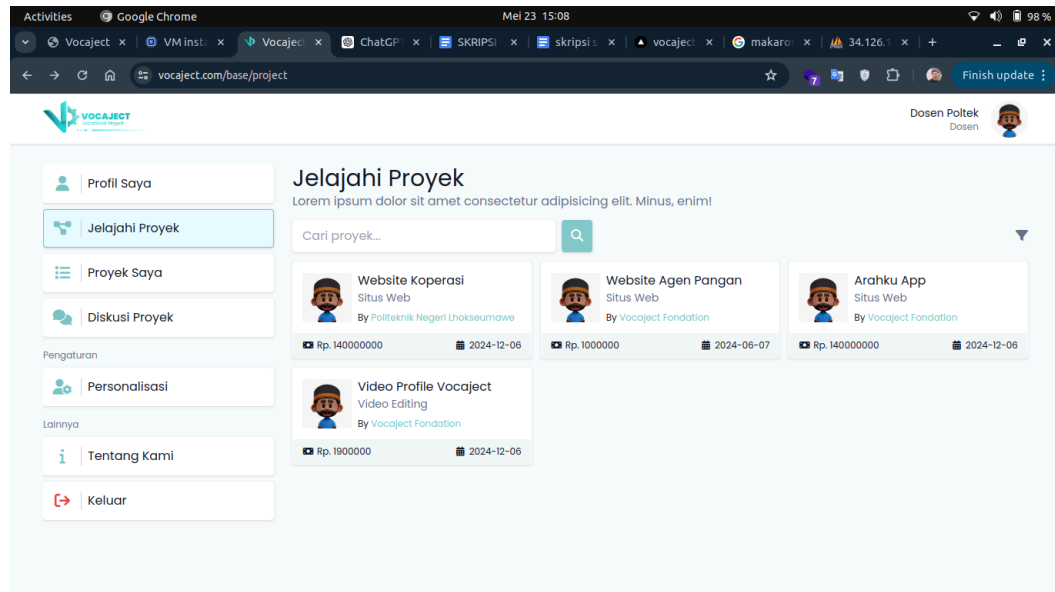
Halaman *login* digunakan oleh pengguna untuk melakukan proses otentikasi akun sebelum mereka dapat mengakses halaman utama dari aplikasi *Vocaject*. Proses otentikasi ini merupakan langkah penting untuk memastikan bahwa hanya pengguna yang memiliki izin yang dapat mengakses fitur-fitur dan data yang terdapat di dalam aplikasi. Pada halaman *login*, pengguna diharuskan memasukkan *email* dan *password* mereka ke dalam kolom yang telah disediakan. Setelah *email* dan *password* dimasukkan, pengguna dapat menekan tombol *login* untuk memulai proses otentikasi. Jika informasi yang dimasukkan benar dan sesuai dengan data yang tersimpan dalam sistem, pengguna akan berhasil masuk dan diarahkan ke halaman utama. Proses ini bertujuan untuk melindungi keamanan akun pengguna dan memastikan bahwa data pribadi mereka tetap aman. Tampilan antarmuka halaman *login* beserta kolom untuk *email* dan *password* dapat dilihat pada Gambar 4.1



Gambar 4.1 Halaman *Login Website Vocaject*

#### 4.1.2. Halaman Jelajah Proyek

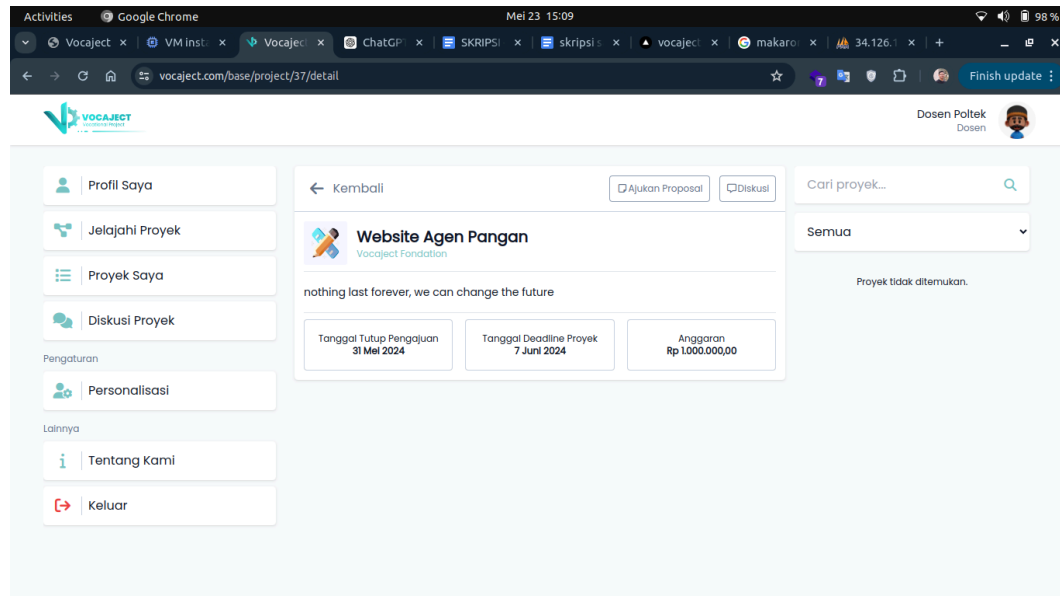
Halaman jelajah proyek digunakan oleh pengguna untuk menjelajahi berbagai proyek yang tersedia di situs web *Vocaject*. Proyek-proyek tersebut merupakan hasil unggahan yang dilakukan oleh pihak industri yang berkolaborasi dengan *platform* ini. Halaman ini dirancang untuk memberikan informasi mendetail tentang setiap proyek, termasuk tujuan, lingkup kerja, dan potensi manfaat yang bisa diperoleh. Pengguna dapat melihat daftar proyek yang telah diunggah dan mempelajari lebih lanjut mengenai setiap proyek melalui deskripsi dan dokumentasi yang disediakan. Namun, akses ke halaman jelajah proyek ini dibatasi dan hanya dapat diakses oleh dosen. Pembatasan ini bertujuan untuk memastikan bahwa informasi dan peluang yang disediakan melalui proyek-proyek ini dapat dimanfaatkan secara optimal oleh pihak yang memiliki kapabilitas dan wewenang untuk menindaklanjuti proyek tersebut dalam konteks akademis atau penelitian. Tampilan dari halaman jelajah proyek dapat dilihat pada Gambar 4.2



Gambar 4.2 Halaman jelajahi proyek

#### 4.1.3. Halaman Detail Proyek

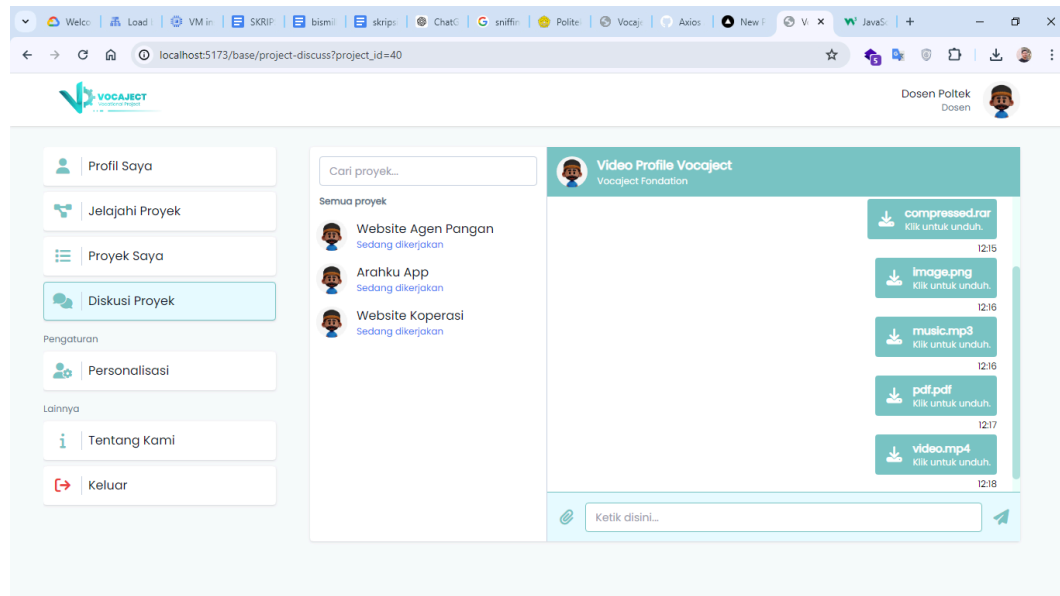
Halaman detail proyek digunakan oleh pengguna untuk melihat informasi yang lengkap dan mendalam mengenai proyek yang sedang dibuka di situs web *Vocaject*. Pada halaman ini, pengguna dapat menemukan berbagai informasi penting dan terperinci yang terkait dengan proyek tersebut. Termasuk pada tanggal tenggat penyelesaian proyek, jumlah anggaran yang tersedia, serta deskripsi proyek. Didalamnya mencakup tujuan, lingkup kerja, dan harapan dari proyek tersebut. Selain itu halaman ini juga menyediakan fitur bagi pengguna untuk mengajukan proposal mereka, fitur ini memungkinkan mereka untuk menawarkan solusi atau kontribusi yang sesuai dengan kebutuhan proyek. Pengguna juga dapat terlibat dalam diskusi dengan pihak terkait, seperti penyelenggara proyek atau anggota tim lainnya, untuk membahas rincian lebih lanjut, klarifikasi, dan kolaborasi, sehingga meningkatkan pemahaman dan koordinasi yang efektif. Semua fitur ini bertujuan untuk memberikan pengguna alat yang mereka butuhkan untuk berpartisipasi secara aktif dan produktif dalam proyek. Tampilan halaman detail proyek dapat dilihat pada Gambar 4.3



Gambar 4.3 Halaman detail proyek

#### 4.1.4. Halaman Diskusi Proyek

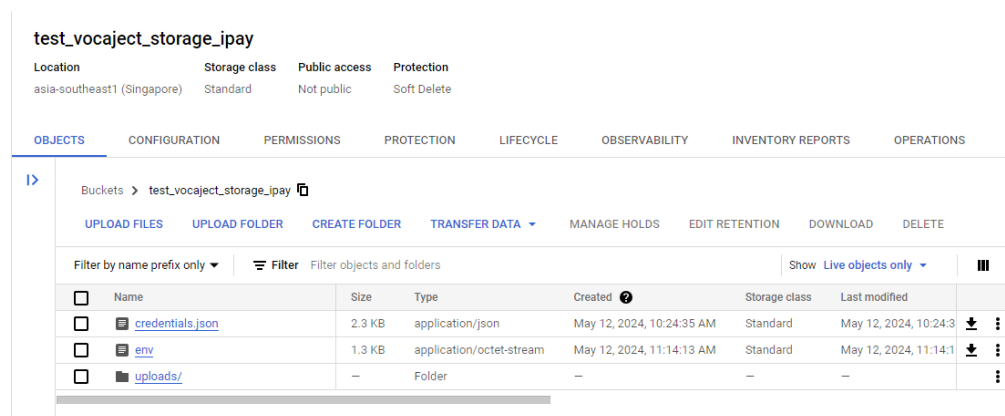
Halaman diskusi proyek merupakan *platform* yang digunakan oleh pengguna untuk berinteraksi dan berkomunikasi dengan pemilik proyek serta pihak terkait lainnya. Di halaman ini, pengguna memiliki kesempatan untuk mengadakan diskusi tentang berbagai aspek proyek, berbagi ide, memberikan masukan, dan memecahkan masalah yang muncul selama pelaksanaan proyek. Selain itu, halaman ini memungkinkan pengguna untuk berbagi dokumen terkait proyek baik itu dari pemilik proyek, dosen pembimbing, atau sumber lainnya yang relevan. Melalui fitur-fitur ini, pengguna dapat memperluas kolaborasi dan meningkatkan koordinasi dalam rangka mencapai tujuan proyek dengan lebih efektif. Tampilan halaman diskusi proyek dapat dilihat pada Gambar 4.4



Gambar 4.4 Halaman diskusi proyek

#### 4.2. Tampilan *Cloud Storage Buckets*

Sesuai dengan rancangan yang dibuat, aplikasi *Vocaject* akan menggunakan *Cloud Storage* sebagai media penyimpanan. Pada penelitian ini sudah dibuatkan satu *Bucket* yang akan menjadi tempat disimpan dokumen-dokumen yang diunggah melalui aplikasi *Vocaject* dan *file* yang dibutuhkan oleh sistem seperti *credentials* dan *environment*. Tampilan *cloud storage* dapat dilihat pada Gambar .

Gambar 4.5 Tampilan *Cloud Storage*








### 4.3. Hasil Pengujian Fungsional

Pengujian fungsional bertujuan untuk memastikan bahwa sistem dapat berhasil mengunggah dan mengunduh dokumen ke dan dari *cloud storage*. Pengujian ini melibatkan pengujian *frontend*, serta mencakup berbagai jenis *file* seperti gambar, *video*, dan *file* lainnya.

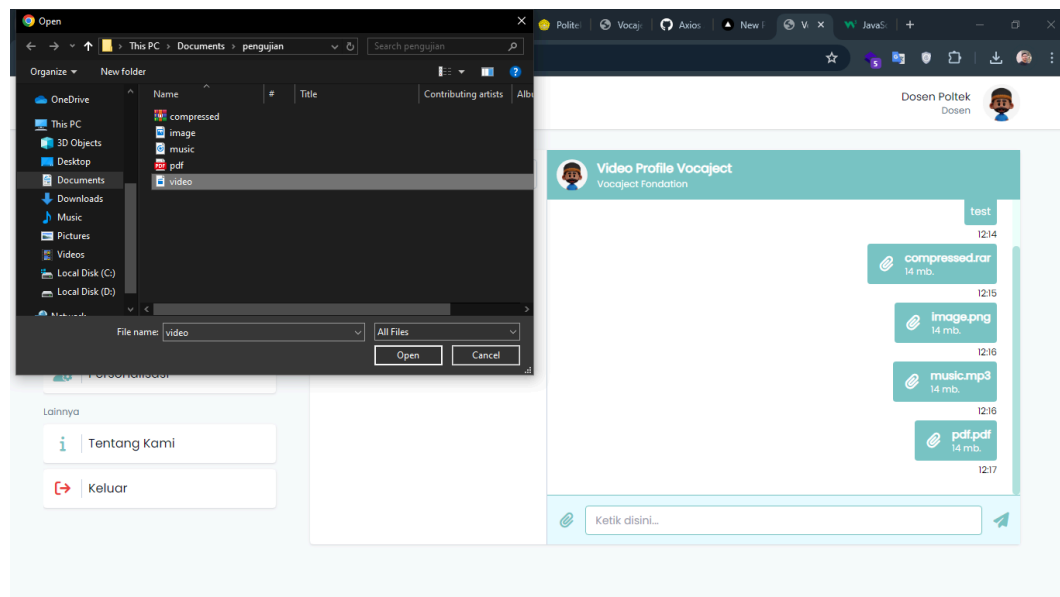
#### 4.3.1. Pengujian Mengunggah Dokumen

Pengujian ini dilakukan dengan cara mengunggah beberapa dokumen melalui halaman fitur *chat* yang tersedia pada aplikasi *Vocaject*. Dalam pengujian ini, berbagai jenis dokumen yang akan diunggah telah dipilih untuk mewakili variasi format yang umum digunakan. Jenis-jenis dokumen tersebut antara lain adalah dokumen terkompresi (*compressed*), gambar (*image*), file audio (*audio*), dokumen PDF, dan file video. Setiap jenis dokumen ini diuji untuk memastikan bahwa fitur *chat* pada aplikasi *Vocaject* mampu mengunggah berbagai format dokumen dengan benar. Jenis-jenis dokumen yang akan diunggah dapat dilihat pada Gambar 4.6.

	compressed Type: WinRAR archive	Date modified: 29/05/2024 9:49 Size: 5,91 MB
	image Type: PNG File Dimensions: 1920 x 1080	Size: 100 KB
	music	Size: 3,48 MB
	pdf	Date modified: 26/09/2021 23:15 Size: 1,32 MB
	video Length: 00:00:06 Frame height: 720 Frame width: 1280	Date modified: 13/09/2021 9:59 Size: 10,7 MB

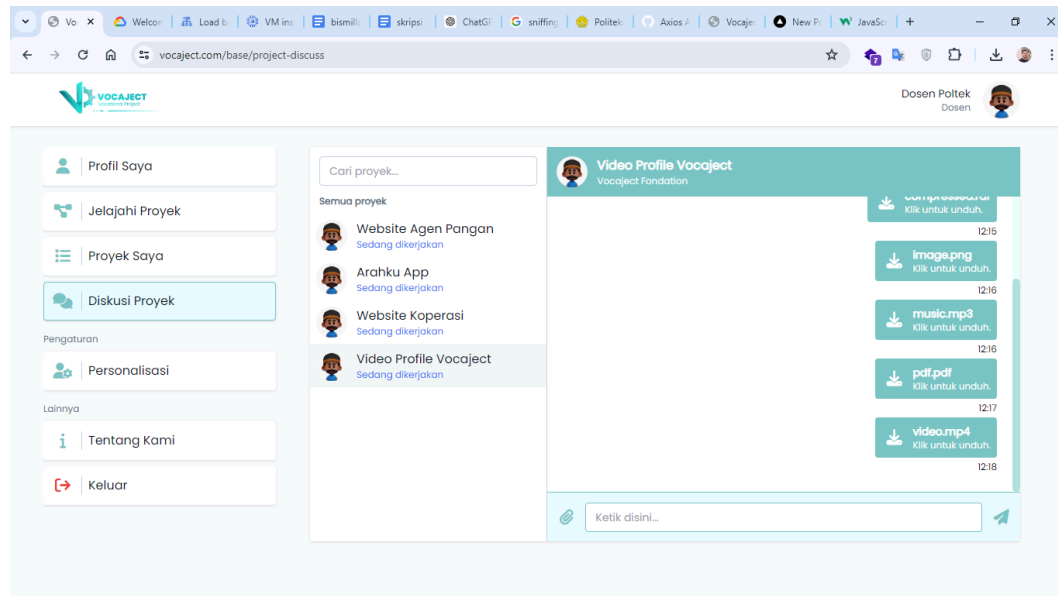
Gambar 4.6 Jenis dokumen yang akan diunggah

Dokumen-dokumen tersebut akan diunggah ke salah satu *room chat*, tepatnya pada proyek *Video Profile Vocaject*. Proses untuk mengunggah dokumen ke dalam fitur *chat* dilakukan dengan menekan ikon *paperclip* yang terdapat di dalam antarmuka *chat* tersebut. Setelah ikon *paperclip* ditekan, pengguna harus memilih dokumen yang ingin diunggah seperti pada Gambar 4.7.



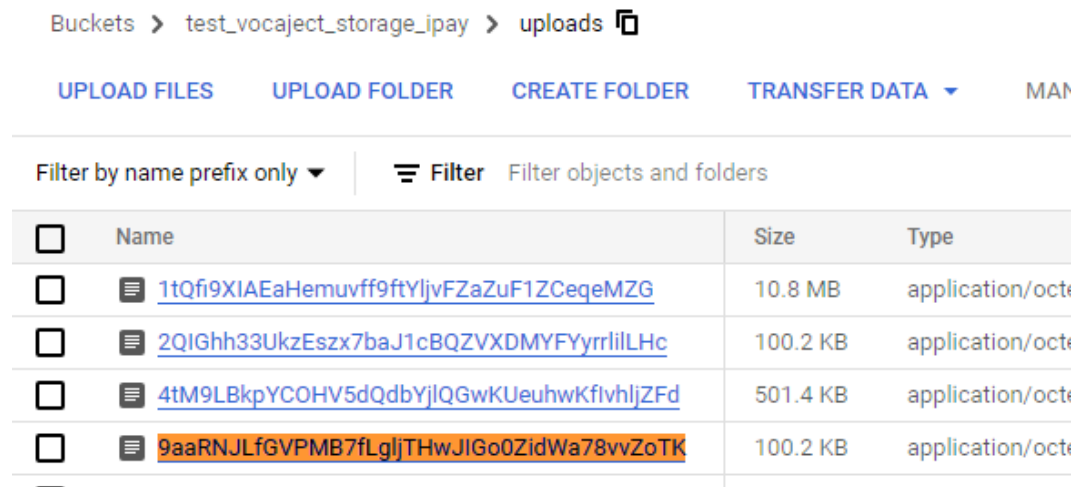
Gambar 4.7 Popup memilih dokumen yang akan diunggah

Setelah dokumen dipilih, sebuah halaman konfirmasi pengiriman akan muncul. Pada halaman ini, pengguna dapat memeriksa kembali dokumen yang akan diunggah sebelum melanjutkan proses. Jika pengguna menekan tombol kirim, maka dokumen yang telah dipilih akan mulai diunggah ke *room chat* tersebut. Setelah proses unggah selesai dan dokumen berhasil diunggah, dokumen tersebut akan ditampilkan sebagai pesan di dalam ruang obrolan. Pada Gambar 4.8, ditunjukkan bahwa semua dokumen telah berhasil diunggah dan ditampilkan dalam ruang obrolan sebagai pesan.



Gambar 4.8 Hasil dokumen berhasil diunggah

Setiap dokumen yang berhasil diunggah oleh pengguna akan diteruskan oleh *backend* untuk disimpan dalam *Cloud Storage*, seperti yang terlihat dalam Gambar 4.9. Proses ini memastikan bahwa dokumen-dokumen tersebut tersimpan secara aman dan dapat diakses dengan mudah dari berbagai perangkat. Selain itu, setiap dokumen yang diunggah akan diberi nama baru yang bersifat unik oleh sistem, menggantikan nama asli dokumen tersebut. Nama dokumen baru ini memiliki kegunaan penting, karena memfasilitasi pengguna untuk mengambil atau mengunduh dokumen yang diinginkan dengan lebih mudah dan cepat tanpa perlu menyimpan nama dokumen asli atau melakukan pencarian yang rumit.



The screenshot shows the 'uploads' folder in a cloud storage interface. It lists four files with their names, sizes, and types. The first file is highlighted in orange.

<input type="checkbox"/>	Name	Size	Type
<input type="checkbox"/>	<a href="#">1tQfi9XIAEaHemuvff9ftYlvFZaZuF1ZCeqeMZG</a>	10.8 MB	application/octet-stream
<input type="checkbox"/>	<a href="#">2QIGhh33UkzEsxz7baJ1cBQZVXDMYFYrrlilLHc</a>	100.2 KB	application/octet-stream
<input type="checkbox"/>	<a href="#">4tM9LBkpYCOHV5dQdbYjlQGwKUeuhwKflvhljZFd</a>	501.4 KB	application/octet-stream
<input type="checkbox"/>	<a href="#">9aaRNJLfGVPMB7fLgljTHwJlGo0ZidWa78vvZoTK</a>	100.2 KB	application/octet-stream

Gambar 4.9 Tampilan hasil unggahan dokumen ke *cloud storage*

Setelah dilakukan pengujian ini, hasil yang dirangkum pada Tabel 4.1 menunjukkan bahwa lima dokumen dengan variasi jenis dan ukuran telah diunggah melalui aplikasi *Vocaject* untuk menguji kemampuan *backend* dalam menangani dokumen yang beragam. Jenis dokumen yang diuji meliputi *file* terkompresi, gambar, audio, PDF, dan video. Selama proses pengunggahan, perbedaan waktu unggah yang diamati terutama disebabkan oleh ukuran dokumen dan kecepatan jaringan internet pada perangkat yang digunakan. Dokumen berukuran lebih besar memerlukan waktu unggah lebih lama, terutama jika kecepatan internet tidak optimal.. Hasil pengujian unggah dokumen dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil pengujian mengunggah dokumen

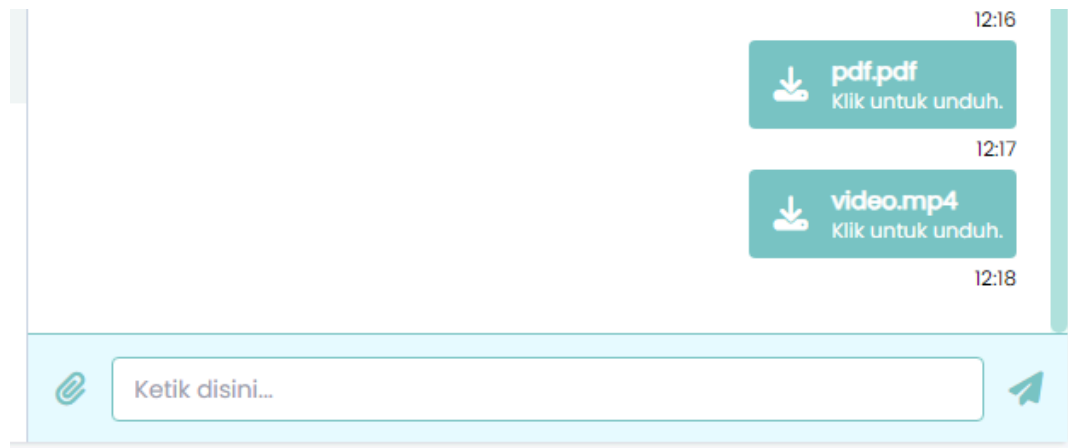
Dokumen	Jenis	Ukuran	Waktu Unggah	Status	Nama Di Cloud
image.png	Gambar	100 kb	12,1 detik	Berhasil Diunggah	9aaRNJLfGVPMB7fLgljTHwJlGo0ZidWa78vvZoTK
video.mp4	Video	10,7 mb	1,2 menit	Berhasil Diunggah	rcjoE6CEsf2NfJR1R

					USucJiaGq dfZAoiEfX 7OI2k
pdf.pdf	PDF	1,32 mb	11,40 detik	Berhasil Diunggah	WzJhkFoR nd2TfBfO WrOz8iDv gQXxMgm xw1n6RW 6Z
music.mp3	Audio	3,48 mb	26.16 detik	Berhasil Diunggah	EQC9XkaJ lgDOJ8Wj AZTx90Le yA1vZjlT RckbZoj8
compressed.rar	Compressed	5,91 mb	34,41 detik	Berhasil Diunggah	sjifci5Hj52 JK0IRJd0q solbjjfXnQ WTa7pkD qQk

Dari hasil pengujian ini, menunjukkan kelima dokumen yang diunggah berhasil tersimpan di *Cloud Storage* dengan persentase 100% berhasil. Dengan demikian, fungsional sistem dalam mengunggah telah berjalan sesuai dengan perancangan.

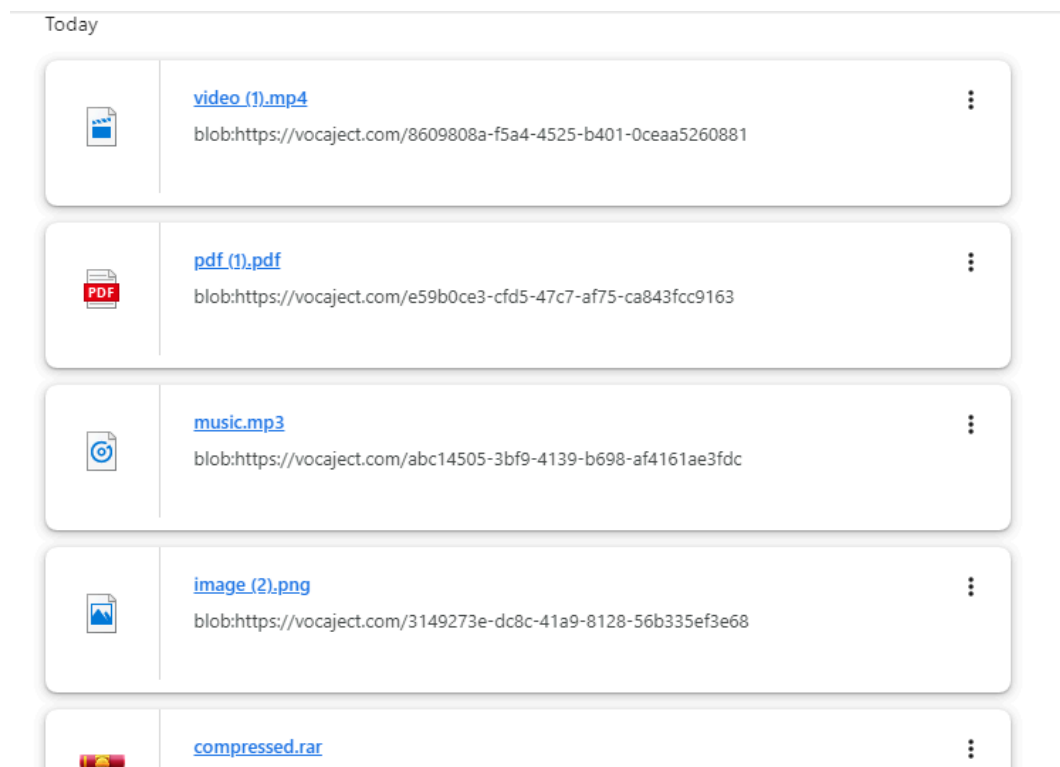
#### 4.3.2. Pengujian Mengunduh Dokumen

Pengujian ini dilakukan dengan cara menekan tombol unduh pada pesan dokumen seperti yang terlihat pada Gambar 4.10. Ketika ditekan, sistem akan memulai melakukan proses pengunduhan.



Gambar 4.10 Tampilan untuk menekan tombol unduh

Setelah proses pengunduhan berhasil, dokumen akan disimpan ke *device* pengguna dan dapat buka jika diinginkan, seperti pada Gambar 4.11.



Gambar 4.11 Tampilan dokumen terunduh

Pengujian ini melibatkan kelima dokumen yang telah diunggah sebelumnya untuk diunduh kembali. Kelima dokumen tersebut berhasil di unduh sesuai yang dijabarkan pada Tabel 4.2.

Tabel 4.2 Hasil pengujian mengunduh dokumen

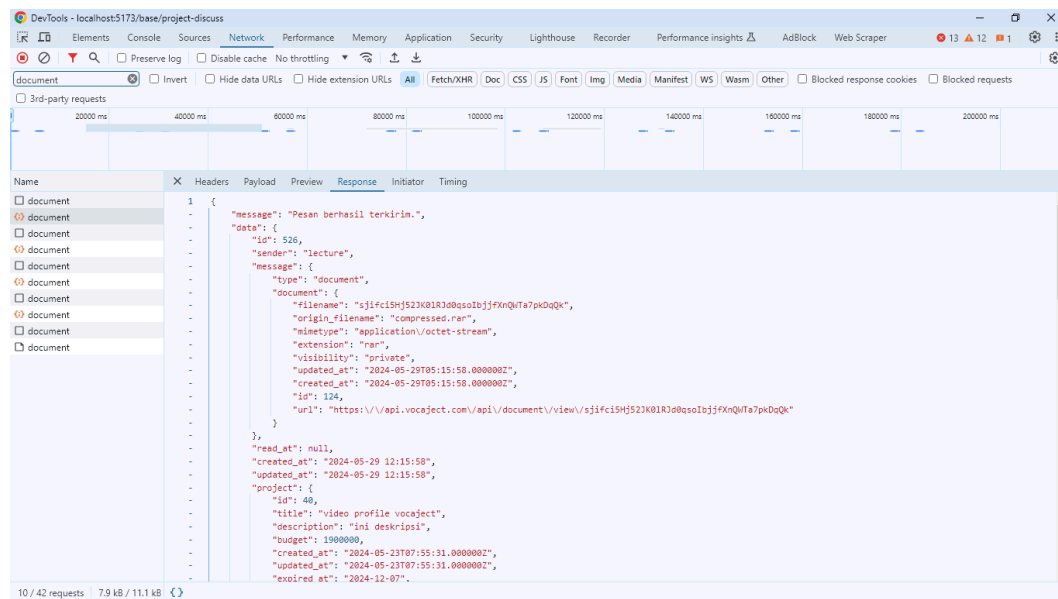
Dokumen	Jenis	Ukuran	Hasil
image.png	Gambar	100 kb	Berhasil Diunduh
video.mp4	Video	10,7 mb	Berhasil Diunduh
pdf.pdf	PDF	1,32 mb	Berhasil Diunduh
music.mp3	Audio	3,48 mb	Berhasil Diunduh
compressed.rar	Compressed	5,91 mb	Berhasil Diunduh

Dari hasil pengujian ini, dokumen yang telah diunggah sebelumnya berhasil diunduh kembali dengan persentase 100% berhasil. Dengan demikian, menunjukkan fungsional sistem dalam mengunduh dokumen sudah sesuai dengan rancangan yang dibuat.

#### 4.4. Hasil Pengujian Keamanan Akses

Pengujian keamanan akses bertujuan untuk memastikan bahwa pengguna tidak dapat mengakses dokumen tanpa wewenang yang sesuai. Ini mencakup pengujian terhadap lapisan keamanan kontrol akses dokumen dan pencegahan akses yang tidak sah.

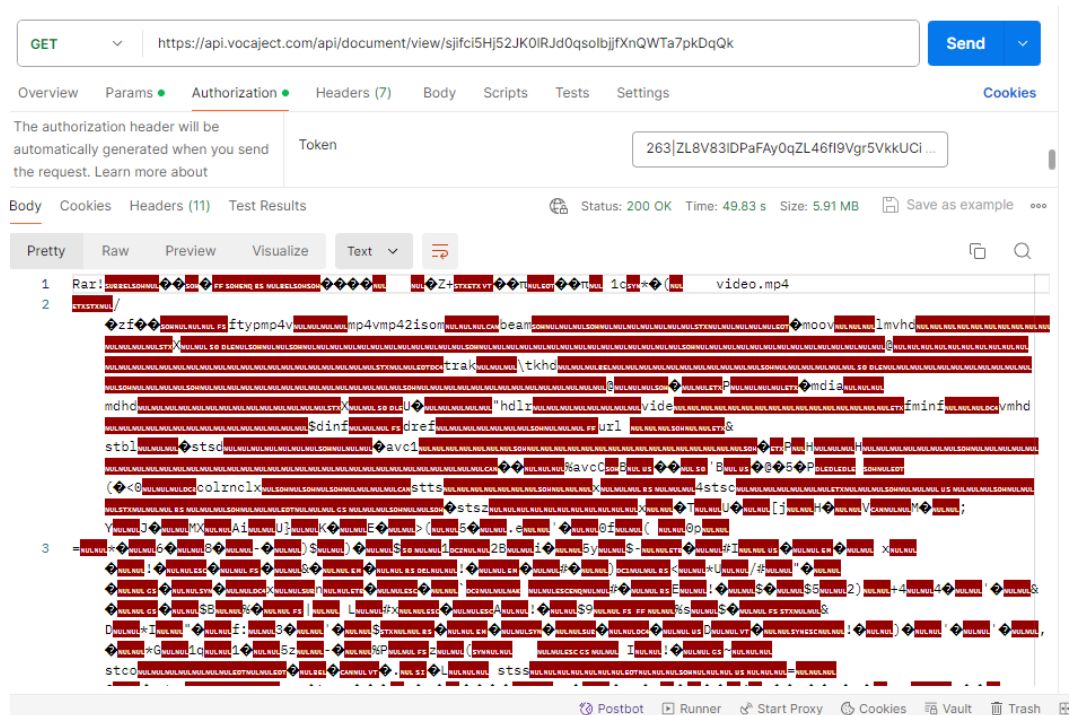
Pengujian ini dilakukan dengan cara mengakses suatu dokumen melalui *url* API menggunakan Postman. Untuk melakukan akses dokumen melalui *url* API, dibutuhkan nama dokumen unik yang diinputkan pada *url* dan *token* otentikasi yang diinputkan pada header *authorization*. Nama dokumen didapatkan dari respon hasil unggahan suatu dokumen.



Gambar 4.12 Tampilan *inspect* halaman web

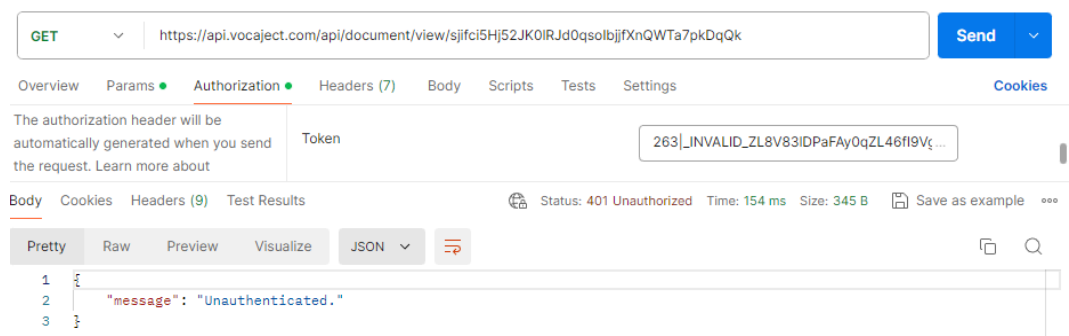
Sebagai contoh, pada Gambar 4.12 menampilkan hasil respon dari unggahan suatu dokumen dengan jenis *compressed*, yang dilihat melalui *tab Network* pada jendela inspeksi web. Pada respon tersebut, menampilkan berbagai informasi tentang dokumen tersebut seperti detail dokumen dan lainnya. Nama dokumen unik dapat diambil dari kunci *filename*. Untuk hasil pengujian akses dokumen dapat dilihat pada Gambar 4.13. Pada gambar tersebut memberikan respon berhasil dengan menampilkan konten yang terlihat aneh. Konten tersebut merupakan program dari dokumen *compressed* yang telah enkripsi.





Gambar 4.13 Hasil pengujian akses dokumen

Namun jika token otentikasi diubah dengan token yang tidak benar, konten tersebut tidak dapat ditampilkan dan mengambilkan respon yang menyatakan tidak terautentikasi seperti pada Gambar 4.14.



Gambar 4.14 Tampilan respon yang tidak terautentikasi

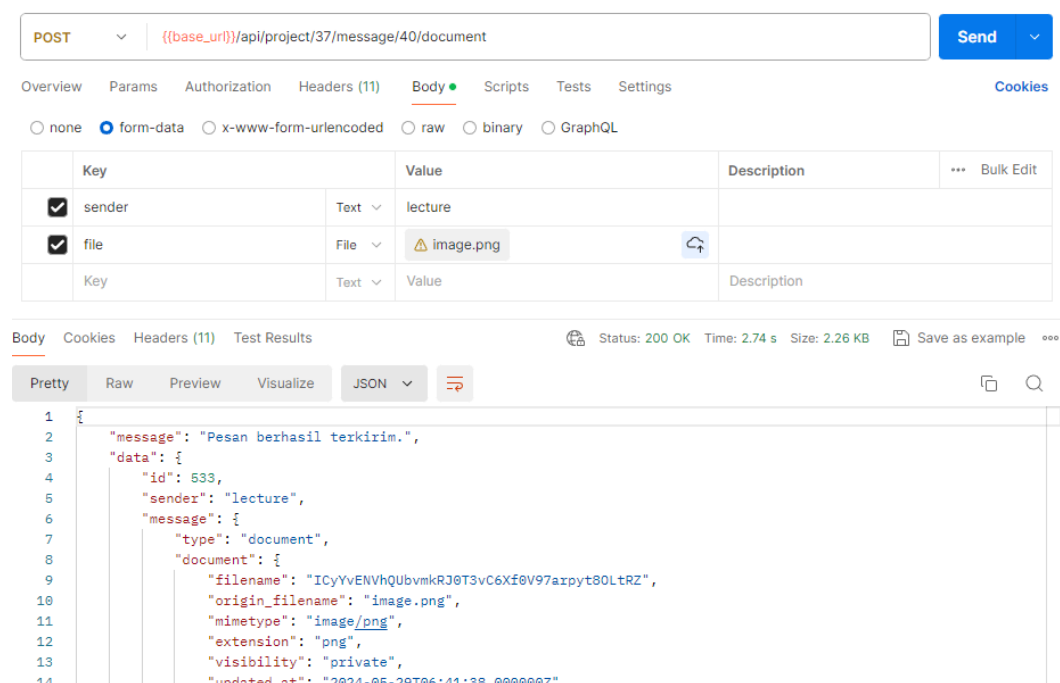
Dari hasil pengujian ini, menunjukan sistem berhasil menangani masalah pengendalian akses oleh pihak yang tidak sah. Dengan demikian, dokumen hanya dapat dikasi token otentikasi yang benar.

## 4.5. Hasil Pengujian *Sniffing*

Pengujian *sniffing* bertujuan untuk memastikan bahwa suatu dokumen tidak dapat dibaca di jaringan selama proses pengunggahan dan pengunduhan. Dalam pengujian ini, paket-paket data yang berjalan di jaringan akan ditangkap menggunakan aplikasi *Wireshark*.

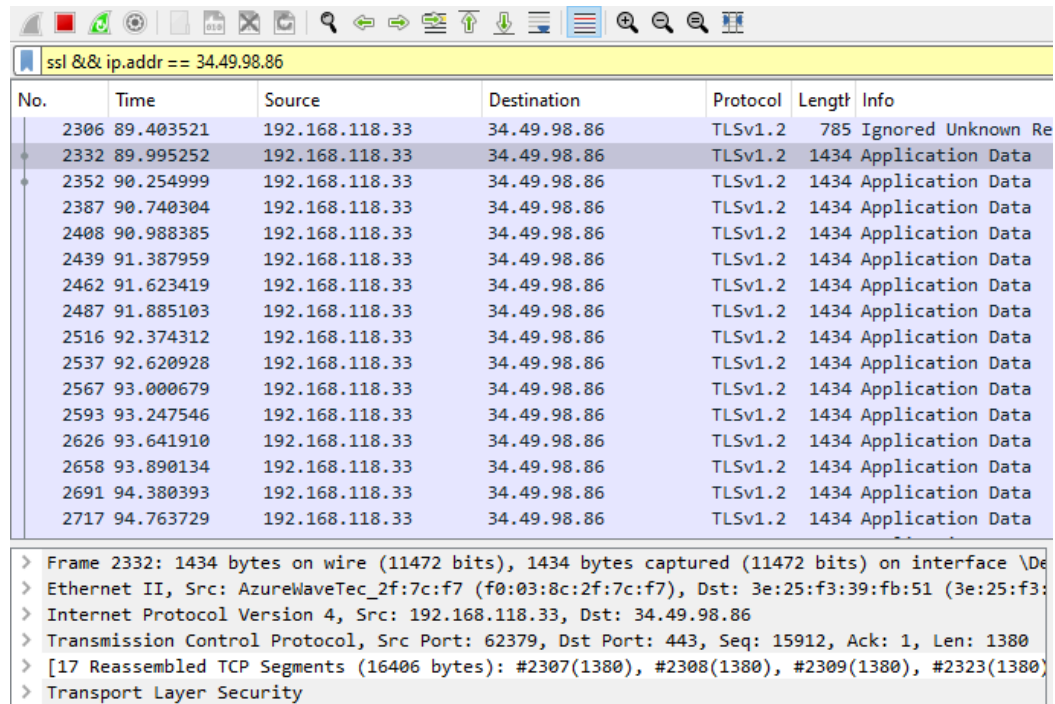
### 4.5.1. Pengujian *Sniffing* Saat Mengunggah Dokumen

Pengujian ini dilakukan dengan beberapa percobaan untuk mengunggah beberapa jenis dokumen menggunakan Postman seperti pada Gambar 4.15. Kemudian akan ditangkap menggunakan *Wireshark* untuk dianalisa.



Gambar 4.15 Hasil unggahan dokumen melalui *Postman*

Saat dokumen diunggah, yang terjadi permintaan tersebut akan dikirimkan dalam bentuk paket-paket kecil ke *server* melalui jaringan pada *device* yang digunakan. Dengan demikian, setiap paket yang melalui jaringan pada *device* yang digunakan dapat ditangkap menggunakan *Wireshark* seperti contoh rekaman paket pada Gambar 4.16..



No.	Time	Source	Destination	Protocol	Length	Info
2306	89.403521	192.168.118.33	34.49.98.86	TLSv1.2	785	Ignored Unknown Re
2332	89.995252	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2352	90.254999	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2387	90.740304	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2408	90.988385	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2439	91.387959	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2462	91.623419	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2487	91.885103	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2516	92.374312	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2537	92.620928	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2567	93.000679	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2593	93.247546	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2626	93.641910	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2658	93.890134	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2691	94.380393	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data
2717	94.763729	192.168.118.33	34.49.98.86	TLSv1.2	1434	Application Data

> Frame 2332: 1434 bytes on wire (11472 bits), 1434 bytes captured (11472 bits) on interface \De  
 > Ethernet II, Src: AzureWaveTec\_2f:7c:f7 (f0:03:8c:2f:7c:f7), Dst: 3e:25:f3:39:fb:51 (3e:25:f3:  
 > Internet Protocol Version 4, Src: 192.168.118.33, Dst: 34.49.98.86  
 > Transmission Control Protocol, Src Port: 62379, Dst Port: 443, Seq: 15912, Ack: 1, Len: 1380  
 > [17 Reassembled TCP Segments (16406 bytes): #2307(1380), #2308(1380), #2309(1380), #2323(1380)  
 > Transport Layer Security

Gambar 4.16 Rekaman paket unggahan dokumen menggunakan *Wireshark*

Seperti pada gambar tangkapan paket tersebut, terlihat paket-paket tersebut telah dilakukan penyaringan agar hanya paket yang berjalan pada protokol *ssl* dan yang menuju atau dari IP 34.49.98.86 saja yang ditangkap. IP tersebut merupakan IP yang didapatkan dari Load Balancer. Dari hasil penangkapan, *Wireshark* tidak dapat menangkap data yang dikirimkan. Hal ini dikarenakan server tempat backend di *deploy* sudah menggunakan protokol *ssl* atau *HTTPS*, sehingga setiap data yang dikirimkan melalui protokol tersebut akan terenkripsi di jaringan.

Tabel 4.3 Hasil pengujian *sniffing* saat dokumen diunggah

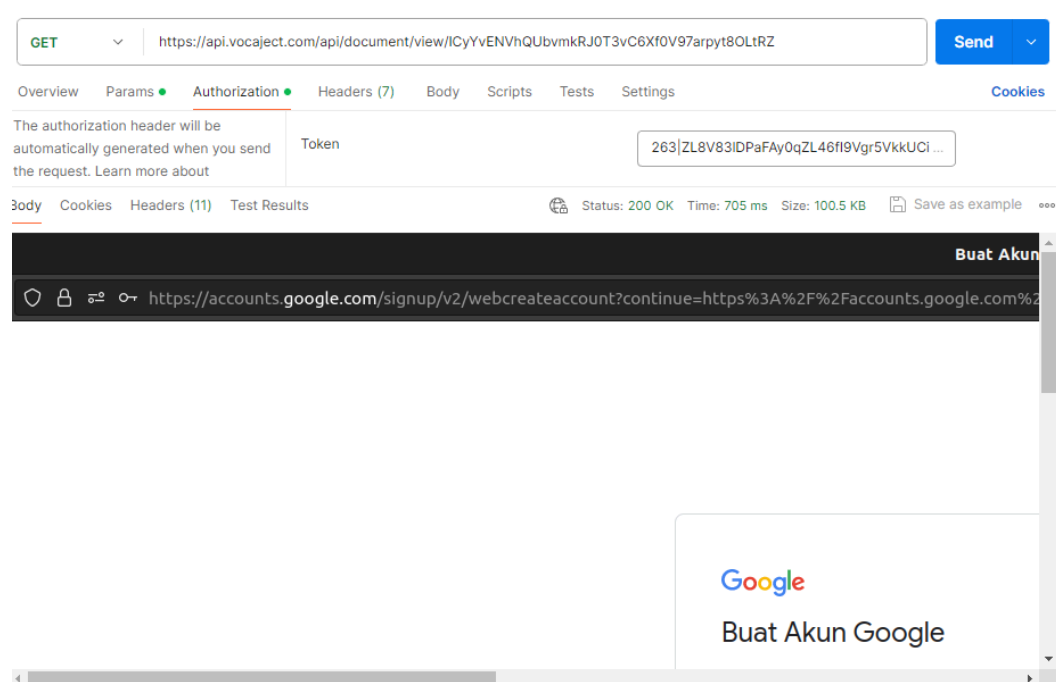
Dokumen	Jenis	Hasil
image.png	Gambar	Terdeteksi dan Tidak Dapat Dibaca
pdf.pdf	PDF	Terdeteksi dan Tidak Dapat Dibaca
music.mp3	Audio	Terdeteksi dan Tidak

		Dapat Dibaca
compressed.rar	Compressed	Terdeteksi dan Tidak Dapat Dibaca

Dari hasil pengujian ini, seperti yang dijabarkan pada Tabel 4.3 menunjukkan setiap dokumen yang diunggah dapat dideteksi namun tidak dapat dibaca oleh *wireshark* dengan persentase 100%. Dengan demikian dokumen yang diunggah sudah dijamin enkripsinya oleh protokol ssl.

#### 4.5.2. Pengujian *Sniffing* Saat Mengunduh Dokumen

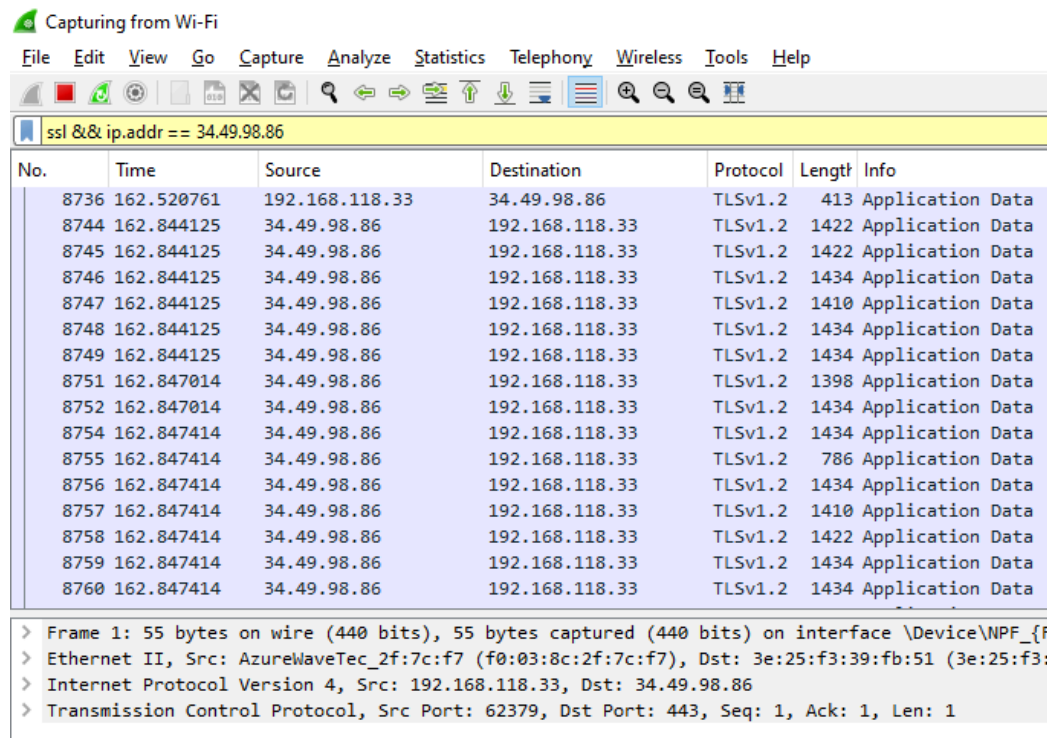
Sama seperti pengujian mengunggah dokumen, pengujian ini dilakukan dengan beberapa percobaan untuk mengakses dokumen menggunakan Postman seperti pada Gambar 4.17. Kemudian akan ditangkap menggunakan *Wireshark*.



Gambar 4.17 Tampilan mengakses dokumen menggunakan *Postman*

Saat dokumen diakses, yang terjadi permintaan tersebut akan dikirimkan ke *server backend*. Kemudian *backend* akan memberikan respon dokumen dalam

bentuk paket-paket kecil melalui jaringan pada *device* yang digunakan. Dengan demikian, setiap paket yang melalui jaringan pada *device* yang digunakan dapat ditangkap menggunakan Wireshark seperti contoh rekaman paket, seperti pada Gambar 4.18.



Capturing from Wi-Fi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ssl && ip.addr == 34.49.98.86

No.	Time	Source	Destination	Protocol	Length	Info
8736	162.520761	192.168.118.33	34.49.98.86	TLSv1.2	413	Application Data
8744	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1422	Application Data
8745	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1422	Application Data
8746	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8747	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1410	Application Data
8748	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8749	162.844125	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8751	162.847014	34.49.98.86	192.168.118.33	TLSv1.2	1398	Application Data
8752	162.847014	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8754	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8755	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	786	Application Data
8756	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8757	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1410	Application Data
8758	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1422	Application Data
8759	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data
8760	162.847414	34.49.98.86	192.168.118.33	TLSv1.2	1434	Application Data

> Frame 1: 55 bytes on wire (440 bits), 55 bytes captured (440 bits) on interface \Device\NPF\_{f0038c2f-7c-f7} (f0:03:8c:2f:7c:f7), Dst: 3e:25:f3:39:fb:51 (3e:25:f3:39:fb:51)

> Ethernet II, Src: AzureWaveTec\_2f:7c:f7 (f0:03:8c:2f:7c:f7), Dst: 3e:25:f3:39:fb:51 (3e:25:f3:39:fb:51)

> Internet Protocol Version 4, Src: 192.168.118.33, Dst: 34.49.98.86

> Transmission Control Protocol, Src Port: 62379, Dst Port: 443, Seq: 1, Ack: 1, Len: 1

Gambar 4.18 Rekaman paket akses dokumen menggunakan *Wireshark*

Seperti pada gambar tangkapan paket tersebut, terlihat paket-paket tersebut telah dilakukan penyaringan agar hanya paket yang berjalan pada protokol ssl dan yang menuju atau dari IP 34.49.98.86 saja yang ditangkap. IP tersebut merupakan IP yang didapatkan dari *Load Balancer*. Dari hasil penangkapan, *wireshark* tidak dapat menangkap data yang dikirimkan. Hal ini dikarenakan *server* tempat backend di deploy sudah menggunakan protokol ssl atau HTTPS, sehingga setiap data yang dikirimkan melalui protokol tersebut akan terenkripsi di jaringan.

Tabel 4.4 Hasil pengujian sniffing saat dokumen diakses

Dokumen	Jenis	Hasil
image.png	Gambar	Terdeteksi dan Tidak Dapat Dibaca
pdf.pdf	PDF	Terdeteksi dan Tidak Dapat Dibaca
music.mp3	Audio	Terdeteksi dan Tidak Dapat Dibaca
compressed.rar	Compressed	Terdeteksi dan Tidak Dapat Dibaca

Dari hasil pengujian ini, seperti yang dijabarkan pada Tabel 4.4 menunjukkan setiap dokumen yang diakses dapat dideteksi namun tidak dapat dibaca oleh *wireshark* dengan persentase 100%. Dengan demikian dokumen yang diunggah sudah dijamin enkripsinya oleh protokol ssl.

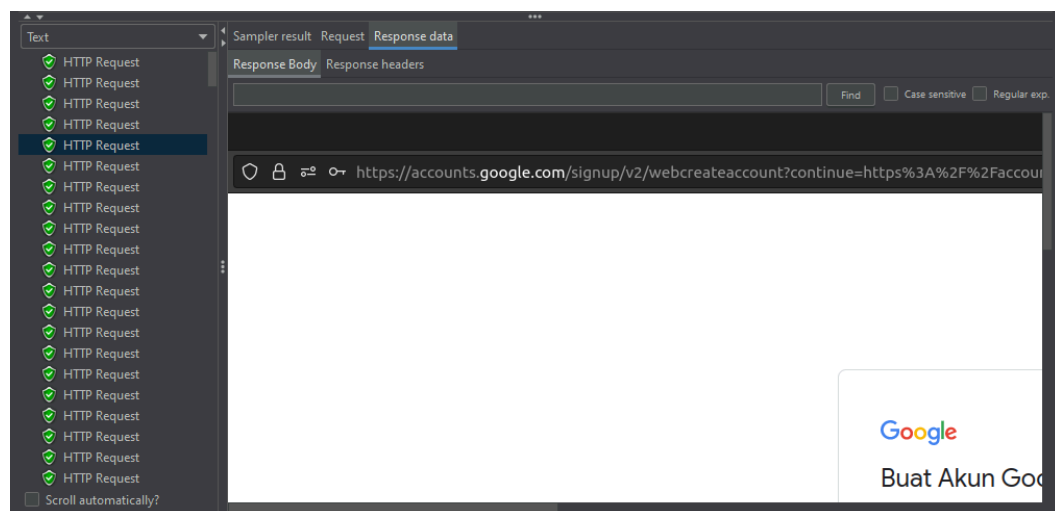
#### 4.6. Hasil Pengujian Ketersediaan

Pengujian ketersediaan bertujuan untuk memastikan bahwa dokumen tetap dapat diakses saat *server* mengalami gangguan atau kelemahan. Dalam pengujian ini, penulis akan melakukan serangan *Distributed Denial of Service* (DDoS) menggunakan aplikasi JMeter dengan mengakses sejumlah dokumen untuk menguji kemampuan *server* dalam menangani permintaan pengguna yang banyak dan dengan ukuran yang besar. Pengujian ini dilakukan pada *server backend* dengan spesifikasi pada Tabel 4.5.

Tabel 4.5 Spesifikasi server yang digunakan pada penelitian

Jenis	Volume
vCPU	1
RAM	3.75 GB

Untuk menguji performa *server backend*, simulasi dilakukan dengan mengirim data berupa *file* gambar berukuran 100 KB dan menggunakan berbagai jumlah sampel, yaitu 50, 100, 250 dan 500. Setiap jumlah sampel mewakili jumlah pengguna *virtual* yang mengirim data secara bersamaan ke *server*. Pengaturan jumlah sampel ini ditentukan oleh tiga parameter, yaitu jumlah thread, periode *ramp-up*, dan jumlah pengulangan. Periode *ramp-up* diset pada 1 detik dan jumlah pengulangan dijalankan sebanyak satu kali. Pengujian menggunakan aplikasi *apache jmeter* dapat dilihat pada gambar 4.19.



Gambar 4.19 Pengujian menggunakan *apache jmeter*

Setiap sampel akan dilakukan dalam dua kali percobaan agar dapat dibandingkan atau dilihat perbedaanya. Hasil dari pengujian ini dapat dilihat pada Tabel 4.6.

Tabel 4.6 Hasil pengujian *stress testing*

No	Looping	Beban	Error	Throughput
1	1	50	0%	18.3/detik
2	1	50	0%	17.2/detik
3	1	100	0%	23.2/detik
4	1	100	0%	23.2/detik

5	1	250	48%	21.3/detik
6	1	250	28%	22.5/detik
8	1	500	64%	44.4/detik
9	1	500	59%	48.3/detik

Berdasarkan data yang tercatat pada Tabel 4.6 menunjukkan bahwa pada percobaan dengan beban 50 sampai 100 berjalan baik dengan persentase *error* 0% dan mampu menangani 18 sampai 23 permintaan setiap detik. Namun pada percobaan dengan beban 250 terjadi *error* dengan persentase rata-rata sebesar 38% dan hanya mampu menangani permintaan dengan rata-rata 21.9 setiap detik, yang mengindikasikan adanya potensi masalah yang perlu diperhatikan. Pada percobaan dengan beban 500, *throughput* dan *error* meningkat signifikan dengan rata-rata 46.37 permintaan setiap detik dan *error* rata-rata 61.7%. Dengan demikian, *server backend* memiliki batas kapasitasnya pada beban antara 250 hingga 500 permintaan, di mana persentase kesalahan mulai meningkat drastis. Oleh karena itu, *Load Balancer* sudah mampu menangani masalah permintaan yang dinamis berdasarkan hasil *throughput* yang meningkat pada beban 500. Namun spesifikasi *server* atau jumlah *instance* yang digunakan dalam penelitian ini, masih belum mampu menangani beban permintaan yang besar. Sehingga *server backend* masih kurang mampu untuk menangani ketersediaan dokumen dengan spesifikasi dan jumlah *instance* pada penelitian ini.



## BAB V

### PENUTUP

#### 5.1. Simpulan

Berdasarkan hasil seluruh tahapan penelitian yang dilakukan pada implementasi penyimpanan dokumen menggunakan *Google Cloud Storage* pada aplikasi *Vocaject* dapat disimpulkan sebagai berikut:

1. Hasil pengujian fungsional pada penelitian ini menunjukkan bahwa unggah dan unduh dokumen berjalan dengan sukses 100% sesuai rancangan, dengan dokumen berhasil disimpan di Cloud Storage. Pengujian keamanan akses mengonfirmasi bahwa sistem efektif mengendalikan akses oleh pihak tidak sah, sehingga dokumen hanya dapat diakses dengan token otentikasi yang benar. Selain itu, pengujian sniffing menunjukkan bahwa meskipun dokumen yang diakses dapat dideteksi, namun isinya tidak dapat dibaca oleh Wireshark dengan tingkat keberhasilan 100% karena telah dienkripsi data oleh protokol SSL.
2. Berdasarkan hasil pengujian *Stress Testing* dengan *server backend* mampu menangani permintaan dengan beban 50 dan 100 dengan baik dan meraih *throughput* 18 sampai 23 permintaan setiap detik, namun memiliki batas kapasitasnya pada beban antara 250 yang terdapat *error* rata-rata sebesar 38% serta *throughput* rata-rata 21.9 setiap detik dan pada beban 500 permintaan terdapat *error* rata-rata sebesar 61.7% serta *throughput* rata-rata 46.37 setiap detik. Oleh karena itu, *Load Balancer* sudah mampu menangani masalah permintaan dokumen yang dinamis berdasarkan hasil *throughput* yang meningkat pada beban 500. Namun spesifikasi *server* atau jumlah *instance* yang digunakan dalam penelitian ini, masih belum mampu menangani beban permintaan yang besar. Sehingga *server backend* masih kurang mampu untuk menangani ketersediaan dokumen dengan spesifikasi dan jumlah *instance* pada penelitian ini.

## 5.2. Saran

Sistem yang dibangun pada penelitian masih terdapat banyak kekurangan yang dapat dikembangkan kembali lagi kedepannya, seperti:

1. Dikembangkan fitur enkripsi *end-to-end* pada data yang dikirimkan dari *Backend* agar tidak dapat dibaca seperti saat melakukan pelacakan pada protokol HTTP menggunakan *Wireshark*.
2. Dikembangkan fitur untuk memeriksa virus melalui *file* yang diunggah.
3. Dikembangkan *unit-test* pada *Backend* untuk memastikan setiap API memberikan informasi yang sesuai dengan rancangan.
4. Guna meningkatkan kinerja *server* dalam menangani beban yang besar, perlu ditingkatkan lagi spesifikasi *server* yang digunakan atau ditambahkan jumlah instance yang terintegrasi dengan *Load Balancer*.

## DAFTAR PUSTAKA

- Anissa, D. L. F., & Andryani, R. (2022). Penerapan Cloud Computing Dalam Aplikasi Panggil Teknisi Berbasis Android Menggunakan Google Cloud Platform. *Jurnal Sains Komputer & Informatika (J-SAKTI)*, 6(2), 1292–1300.
- Arifin, B. S., & Laya, M. (2019). Web Service Processor sebagai Penghubung Sistem Kiosk Medicom dengan SIM RS Kanker Dharmais. *JURNAL MULTINETICS*, 3(2).
- Baraka. (2023, October 25). *Framework PHP Modern untuk Pengembangan Aplikasi Web Modern*. <https://Baraka.Uma.Ac.Id/Laravel-Framework-Php-Modern-Untuk-Pengembangan-Aplikasi-Web-Modern/>.
- Fachry, M., Kusyanti, A., & Amron, K. (2018). Pengamanan Data pada Media Penyimpanan Cloud Menggunakan Teknik Enkripsi dan Secret Sharing. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, 2(11), 4863–4869.
- Hijriyannto, B., & Ulum, F. (2021). PERBANDINGAN PENERAPAN METODE PENGAMANAN WEB SERVER MENGGUNAKAN MOD EVASIVE DAN DDOS DEFLATE TERHADAP SERANGAN SLOW POST. *JECSIT*, 1(1), 88–92.
- Ismail, A., Ananta, A. Y., Arief, S. N., & Hamdana, E. N. (2023). PERFORMANCE TESTING SISTEM UJIAN ONLINE MENGGUNAKAN JMETER PADA LINGKUNGAN VIRTUAL. *Jurnal Informatika Polinema (JIP)*, 9(2), 159–164.
- Kholil, M., & Mu'min, S. (2018). Pengembangan Private Cloud Storage sebagai Sentralisasi Data Universitas Nahdlatul Ulama Sidoarjo Berbasis Open Source Owncloud. *Jurnal Ilmu Komputer Dan Desain Komunikasi Visual*, 3(1), 34–42.
- Lenawati, M., & Mumtahana, H. A. (2018). PENERAPAN CLOUD STORAGE DALAM PERKULIAHAN FAKULTAS TEKNIK UNIVERSITAS PGRI MADIUN . *Journal of Computer, Information System, & Technology Management* , 1(2), 55–58.
- Luthfansa, Z. M., & Rosiani, U. D. (2021). Pemanfaatan Wireshark untuk Sniffing Komunikasi Data Berprotokol HTTP pada Jaringan Internet. *Journal Information Engineering and Educational Technology (JIEET)*, 5(1).
- Manalu, A. S., Siregar, I. M., Panjaitan, N. J., & Sugara, H. (2021). RANCANG BANGUN INFRASTRUKTUR CLOUD COMPUTING DENGAN

OPENSTACK PADA JARINGAN LOKAL MENGGUNAKAN VIRTUALBOX. *Jurnal Teknik Informasi Dan Komputer (Tekinkom)*, 4(2), 303. <https://doi.org/10.37600/tekinkom.v4i2.335>

- Ramsari, N., & Ginanjar, A. (2022). Implementasi Infrastruktur Server Berbasis Cloud Computing Untuk Web Service Berbasis Teknologi Google Cloud Platform. *Conference SENATIK STT Adisutjipto Yogyakarta*, 7. <https://doi.org/10.28989/senatik.v7i0.472>
- Ray, S., & Sarkar, A. de. (2019). EXECUTION ANALYSIS OF LOAD BALANCING ALGORITHMS IN CLOUD COMPUTING ENVIRONMENT. *International Journal on Cloud Computing: Services and Architecture (IJCCSA)*, 2(5), 1–13.
- Seputra, K. A., Paramartha, A. A. G. Y., & Wijaya, I. N. S. W. (2020). IMPLEMENTASI GOOGLE DRIVE CLOUD STORAGE PADA SISTEM REPOSITORI AL-DARING. *Science and Information Technology (SINTECH)*, 3(1), 49–67.
- Sunaryo, Tedyyana, A., & Kasmawi. (2017). Rancang Bangun Server Cloud Computing Di Politeknik Negeri Bengkalis . *JURNAL INOVTEK POLBENG - SERI INFORMATIKA*, 2(1), 33–40.
- Suni, A. F., Putri, A. M., Muzaiyanah, A. M., Purbawanto, K., & Safitri, D. A. (2023). Optimalisasi Cloud Storage Guna Pengelolaan Data Administrasi Desa Pledokan. *Jurnal Bina Desa*, 5(2), 212–224.
- Wulandari, S., & Ganggi, R. I. P. (2021). Pengalaman pemanfaatan cloud storage mahasiswa Teknik Komputer Universitas Diponegoro (Undip) dalam pengelolaan arsip digital. *Informatio: Journal of Library and Information Science*, 1(1), 49. <https://doi.org/10.24198/inf.v1i1.31111>
- Yudhanto, F. H., Nugroho, H., & Suryadi, A. H. (2018). Aplikasi Pengelolaan Sistem Informasi Dinas Kesehatan Kabupaten Bandung. *E-Proceeding of Applied Science*, 4(3), 1621–1629.
- Yusmita, A. R., Anra, H., & Novriando, H. (2020). Sistem Informasi Pelatihan pada Kantor Unit Pelaksana Teknis Latihan Kerja Industri (UPT LKI) Provinsi Kalimantan Barat. *Jurnal Sistem Dan Teknologi Informasi (Justin)*, 8(2), 160. <https://doi.org/10.26418/justin.v8i2.36797>