

LAPORAN PRAKTIKUM TEKNOLOGI CLOUD COMPUTING

WEB SERVICE REST

IF-A



Disusun oleh

Nama : Resti Ramadhani

NIM : 123220147

**PROGRAM STUDI INFORMATIKA
JURUSAN INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" YOGYAKARTA
2025**

HALAMAN PENGESAHAN

LAPORAN PRAKTIKUM

WEB SERVICE REST

IF-A

Disusun Oleh:

Resti Ramadhani 123220147

Telah diperiksa dan disetujui oleh Asisten Praktikum Teknologi Cloud Computing

Pada tanggal:

Menyetujui

Asisten Praktikum

Asisten Praktikum

Muhammad Rafli

NIM 123220078

Aditya Prayoga

NIM 123220098

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa yang senantiasa mencurahkan rahmat dan hidayah-Nya sehingga kami dapat menyelesaikan tugas “*Web-service Rest*” Praktikum Teknologi *Cloud Computing*. Adapun laporan ini berisi tentang tugas dari hasil pembelajaran RESTful API selama praktikum berlangsung.

Tidak lupa ucapan terima kasih kepada Muhammad Rafli dan Aditya Prayoga selaku asisten praktikum yang selalu membimbing dan mengajari kami dalam melaksanakan praktikum dan dalam menyusun laporan ini. Laporan ini masih sangat jauh dari kesempurnaan, oleh karena itu kritik serta saran yang membangun kami harapkan untuk menyempurnakan laporan akhir ini.

Atas perhatian dari semua pihak yang membantu penulisan ini, kami ucapkan terima kasih. Semoga laporan ini dapat dipergunakan seperlunya.

Yogyakarta, 01 Maret 2025

Penulis

DAFTAR ISI

LAPORAN PRAKTIKUM TEKNOLOGI CLOUD COMPUTING	i
HALAMAN PENGESAHAN	ii
HALAMAN PERSETUJUAN	ii
KATA PENGANTAR	iii
DAFTAR ISI	iv
DAFTAR TABEL	vi
DAFTAR GAMBAR	vii
DAFTAR LAMPIRAN	viii
BAB I PENDAHULUAN	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah	2
1.3 Tujuan.....	2
1.4 Manfaat.....	2
BAB II TINJAUAN LITERATUR	4
2.1 Konsep Pencatatan Digital.....	4
2.2 Teknologi Frontend: HTML, CSS, dan Java Script	4
2.2.1 HTML.....	4
2.2.2 CSS	4
2.2.3 Java script	5
2.3 RESTful API Sebagai Arsitektur Backend.....	5
2.4 Manajemen Data MYSQL.....	5
2.5 Object Relational Mapping (ORM).....	6
BAB III METODOLOGI.....	7
3.1 Analisis Permasalahan	7
3.2 Perancangan Solusi.....	8
3.2.1 Arsitektur sistem.....	9
BAB IV HASIL DAN PEMBAHASAN	11
4.1 Hasil.....	11
4.1.1 Backend	11
4.2 Pembahasan	28
BAB V PENUTUP	29

5.1 Kesimpulan.....	29
5.2 Saran	29
DAFTAR PUSTAKA	30
LAMPIRAN	31

DAFTAR TABEL

Tabel 4. 1	Kode Program Database.js.....	11
Tabel 4. 2	Kode Program NoteModel.js	11
Tabel 4. 3	Kode Program NoteController.js	13
Tabel 4. 4	Kode Program Route.js.....	15
Tabel 4. 5	Kode Program index.js	16
Tabel 4. 6	Kode Program Request.rest	16
Tabel 4. 7	Kode Program index.html.....	19
Tabel 4. 8	Kode Program index.js	23

DAFTAR GAMBAR

Gambar 3. 1 Arsitektur <i>Backend</i>	8
Gambar 3. 2 Database Catatan.....	9
Gambar 3. 3 Arsitektur <i>Frontend</i>	9
Gambar 3. 4 Arsitektur Sistem.....	10
Gambar 4. 1 Tabel Notes	12
Gambar 4. 2 Hasil request method POST	17
Gambar 4. 3 Hasil request method PUT	18
Gambar 4. 4 Hasil request method DELETE.....	18
Gambar 4. 5 Hasil request method GET	19
Gambar 4. 6 Tampilan UI Website	23

DAFTAR LAMPIRAN

Lampiran 1 Repository Github	31
------------------------------------	----

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam kehidupan modern, banyak individu mengalami kendala dalam mengatur dan menyimpan catatan secara efektif. Metode pencatatan konvensional dalam bentuk fisik memiliki berbagai keterbatasan, seperti risiko kehilangan, kerusakan, serta kesulitan dalam pencarian kembali informasi yang dibutuhkan. Sementara itu, sebagian besar aplikasi pencatatan digital yang tersedia sering kali kurang memberikan fleksibilitas dalam hal integrasi dengan platform lain atau kemampuan penyesuaian sesuai preferensi pengguna. Oleh karena itu, diperlukan sebuah solusi inovatif yang dapat mendukung pengguna dalam menyimpan, mengelola, dan mengakses catatan mereka dengan lebih praktis, aman, serta kompatibel di berbagai perangkat.

Untuk menjawab tantangan tersebut, pengembangan sebuah aplikasi catatan berbasis web menjadi solusi yang tepat. Aplikasi ini dirancang menggunakan HTML, CSS, dan JavaScript untuk *frontend* serta RESTful API untuk *backend*. Dengan teknologi ini, pengguna dapat mencatat, mengedit, serta menghapus informasi dengan mudah melalui antarmuka yang responsif dan interaktif. *Backend* aplikasi akan menerapkan RESTful API guna memastikan layanan yang terstruktur, modular, dan dapat diintegrasikan dengan berbagai platform lainnya. Database yang digunakan adalah MySQL, yang memungkinkan penyimpanan data secara terstruktur, aman, dan efisien.

Penggunaan RESTful API untuk *backend* memberikan keuntungan dalam hal skalabilitas serta kemudahan integrasi dengan berbagai sistem di masa depan, termasuk pengembangan aplikasi mobile. Sementara itu, penggunaan HTML, CSS, dan JavaScript pada *frontend* memungkinkan aplikasi tetap ringan, fleksibel, dan mudah dikembangkan tanpa bergantung pada framework tertentu. JavaScript akan berperan dalam meningkatkan interaktivitas serta menghubungkan *frontend* dengan *backend* melalui AJAX atau Fetch API. MySQL dipilih sebagai sistem manajemen basis data karena keandalannya dalam menangani jumlah data yang besar serta dukungannya terhadap operasi transaksi yang kompleks. Dengan kombinasi teknologi ini, aplikasi catatan yang dikembangkan tidak hanya memiliki kinerja optimal dan fleksibilitas tinggi, tetapi juga

meningkatkan produktivitas pengguna melalui sistem pencatatan yang lebih terorganisir, mudah diakses, serta dapat disesuaikan dengan kebutuhan individu maupun organisasi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, terdapat beberapa permasalahan utama yang menjadi fokus dalam pengembangan aplikasi catatan ini, yaitu:

1. Bagaimana merancang sebuah aplikasi catatan yang dapat menyimpan, mengelola, dan mengakses catatan secara efisien bagi pengguna?
2. Bagaimana mengimplementasikan arsitektur *backend* berbasis RESTful API yang mudah diintegrasikan dengan berbagai platform?
3. Bagaimana membangun *frontend* aplikasi yang fleksibel sehingga tetap ringan namun memiliki pengalaman pengguna (UX) yang baik?

1.3 Tujuan

Berdasarkan latar belakang dan rumusan masalah yang telah diuraikan, tujuan dari pengembangan aplikasi catatan berbasis web ini adalah sebagai berikut:

1. Mengembangkan aplikasi catatan berbasis web yang dapat menyimpan, mengelola, dan mengakses catatan secara efisien serta aman bagi pengguna.
2. Merancang dan mengimplementasikan *backend* berbasis RESTful API yang modular, skalabel, dan mudah diintegrasikan dengan berbagai platform.
3. Membangun *frontend* aplikasi yang fleksibel, ringan, serta memberikan pengalaman pengguna (UX) yang baik.

1.4 Manfaat

Pengembangan aplikasi catatan ini diharapkan memberikan manfaat secara teknis maupun non-teknis. Berikut manfaat yang diharapkan dengan adanya sistem ini.

a. Manfaat Teknis

1. Meningkatkan efisiensi dalam pengelolaan dan penyimpanan catatan melalui solusi digital berbasis web.
2. Memastikan keamanan data dengan implementasi arsitektur *backend* yang aman dan terstruktur.
3. Mempermudah akses catatan dari berbagai perangkat dengan dukungan teknologi berbasis web dan API yang fleksibel.

b. Manfaat Non-Teknis

1. Membantu pengguna dalam mengorganisir informasi secara lebih efektif dan sistematis.
2. Meningkatkan produktivitas individu maupun organisasi dalam pencatatan dan manajemen data.
3. Memberikan pengalaman pengguna yang lebih baik melalui antarmuka yang responsif dan mudah digunakan.

BAB II

TINJAUAN LITERATUR

2.1 Konsep Pencatatan Digital

Pencatatan digital telah berkembang pesat seiring dengan kemajuan teknologi. Menurut Ramadhan & Sari (2020), pencatatan digital lebih efisien dibandingkan metode konvensional karena memungkinkan aksesibilitas, pencarian cepat, serta integrasi dengan layanan lain. Berbeda dengan pencatatan manual yang rentan terhadap kehilangan dan kerusakan, aplikasi catatan berbasis web menawarkan penyimpanan yang lebih aman serta kemudahan dalam berbagi informasi (Susanto, 2019).

2.2 Teknologi *Frontend*: HTML, CSS, dan Java Script

2.2.1 HTML

HTML (*HyperText Markup Language*) merupakan bahasa markup yang digunakan untuk menyusun struktur dan konten halaman web. Sebagai dasar dari pengembangan situs web, HTML memungkinkan pengembang untuk menentukan berbagai elemen seperti teks, gambar, tautan, dan multimedia yang dapat diakses melalui internet. Dalam praktiknya, HTML sering dikombinasikan dengan CSS untuk memperindah tampilan serta JavaScript untuk meningkatkan interaktivitas halaman. Perannya yang fundamental menjadikan HTML sebagai teknologi utama dalam pengembangan web modern.

2.2.2 CSS

Cascading Style Sheets (CSS) adalah bahasa desain yang digunakan untuk mengatur tampilan dan tata letak halaman web. Dikembangkan oleh *World Wide Web Consortium* (W3C) pada tahun 1996, CSS memungkinkan pemisahan antara struktur konten dan elemen visual dalam dokumen berbasis markup seperti HTML. Dengan CSS, pengembang web dapat mengontrol aspek visual situs, termasuk warna, font, tata letak, serta elemen grafis seperti gambar dan latar belakang. Kehadirannya memberikan fleksibilitas dalam desain, memungkinkan web designer untuk dengan mudah mengubah tampilan sebuah situs tanpa perlu mengedit struktur HTML secara langsung.

2.2.3 Java script

JavaScript adalah bahasa pemrograman yang digunakan untuk mengembangkan website agar lebih dinamis dan interaktif. JavaScript hadir untuk meningkatkan fungsionalitas dengan memungkinkan interaksi pengguna. Dengan JavaScript, pengembang dapat membuat aplikasi web, tools, bahkan game berbasis web.

Secara teknis, JavaScript (JS) adalah bahasa pemrograman berjenis interpreter, sehingga tidak memerlukan compiler untuk dijalankan. JS memiliki berbagai fitur unggulan, seperti berorientasi objek, berjalan di sisi klien (*client-side*), bersifat *high-level programming*, serta *loosely typed*, yang memungkinkan fleksibilitas dalam penulisan kode.

2.3 RESTful API Sebagai Arsitektur *Backend*

RESTful API adalah sebuah gaya arsitektur untuk antarmuka pemrograman aplikasi (API) yang menggunakan permintaan HTTP untuk mengakses dan memanfaatkan data. Data tersebut dapat digunakan untuk melakukan operasi *GET*, *PUT*, *POST*, dan *DELETE*, yang masing-masing merujuk pada proses membaca, memperbarui, membuat, dan menghapus sumber daya yang tersedia.

API sendiri merupakan sekumpulan kode yang memungkinkan dua perangkat lunak berkomunikasi satu sama lain. Desain API menentukan cara yang tepat bagi pengembang dalam menulis sebuah program (klien) yang menggunakan API untuk meminta layanan dari aplikasi lain (server). Dalam perkembangan teknologi perangkat lunak, API telah menjadi mekanisme penting untuk memastikan interoperabilitas antar perangkat lunak.

RESTful API juga dikenal sebagai RESTful *web services* atau REST API. API ini didasarkan pada *Representational State Transfer* (REST), yaitu sebuah gaya arsitektur dan pendekatan komunikasi yang sering digunakan dalam pengembangan layanan berbasis web. Pendekatan ini juga dapat memfasilitasi komunikasi antara berbagai jenis aplikasi lainnya, sehingga meningkatkan efisiensi dalam pertukaran data dan integrasi sistem.

2.4 Manajemen Data MYSQL

MySQL adalah sistem manajemen basis data relasional (RDBMS) *open-source* yang banyak digunakan karena kecepatan, keandalan, dan kemudahan penggunaannya (Widjaja, 2020). MySQL menggunakan *Structured Query Language* (SQL) untuk

mengelola data dalam tabel-tabel yang saling berelasi, memungkinkan operasi seperti pembuatan, pembaruan, penghapusan, dan pengambilan data secara efisien (Kusuma & Prasetyo, 2021).

2.5 *Object Relational Mapping (ORM)*

Object Relational Mapping (ORM) adalah teknik yang digunakan untuk menghubungkan program berbasis *Object-Oriented Programming (OOP)* dengan basis data relasional. ORM memungkinkan pengembang untuk melakukan operasi seperti Create, Read, Update, dan Delete (CRUD) pada database menggunakan metode berbasis objek, tanpa harus menulis kode SQL secara langsung. Dengan ORM, interaksi dengan database menjadi lebih sederhana dan efisien, karena ORM menangani logika yang diperlukan untuk berkomunikasi dengan database.

Keunggulan utama ORM adalah mempercepat pengembangan, mengurangi kompleksitas kode, meningkatkan keamanan dengan mencegah serangan SQL injection, serta menghemat biaya pengembangan. Namun, ORM juga memiliki beberapa kelemahan, seperti memerlukan waktu untuk dipelajari, kurang optimal untuk *query* yang sangat kompleks, dan cenderung lebih lambat dibandingkan dengan penggunaan SQL langsung. Meskipun begitu, ORM tetap menjadi solusi populer dalam pengembangan aplikasi yang membutuhkan integrasi dengan database relasional secara lebih mudah dan efisien.

BAB III

METODOLOGI

3.1 Analisis Permasalahan

Dalam penugasan praktikum ini, permasalahan utama yang dihadapi adalah bagaimana menciptakan sebuah sistem pencatatan digital yang efektif, fleksibel, serta dapat diakses dengan mudah di berbagai perangkat. Berdasarkan identifikasi yang telah dilakukan, metode pencatatan konvensional memiliki banyak keterbatasan, seperti risiko kehilangan, kerusakan, serta kesulitan dalam mencari kembali informasi yang dibutuhkan. Sementara itu, aplikasi pencatatan digital yang telah tersedia sering kali memiliki keterbatasan dalam hal fleksibilitas integrasi dan penyesuaian oleh pengguna.

Untuk mengatasi tantangan ini, pengembangan sebuah aplikasi catatan berbasis web menjadi solusi yang diusulkan. Aplikasi ini dirancang untuk memberikan kemudahan dalam menyimpan, mengelola, serta mengakses catatan secara lebih efisien dan aman. Secara teknis, aplikasi ini akan dikembangkan menggunakan teknologi berikut:

- a. *Frontend* dibangun menggunakan HTML, CSS, dan JavaScript untuk memastikan tampilan yang responsif dan interaktif.
- b. *Backend* menggunakan RESTful API untuk menyediakan layanan yang modular, terstruktur, serta memungkinkan integrasi dengan sistem lain.
- c. MySQL dipilih sebagai sistem manajemen basis data untuk menyimpan catatan secara aman, efisien, dan dapat menangani jumlah data yang besar.

Tujuan utama dari penugasan ini adalah untuk mengembangkan aplikasi yang memiliki kinerja optimal dan fleksibilitas tinggi sehingga dapat meningkatkan produktivitas pengguna. Dengan menerapkan arsitektur RESTful API, aplikasi ini dapat dikembangkan lebih lanjut untuk integrasi dengan perangkat mobile atau sistem lain di masa depan. Selain itu, penggunaan teknologi berbasis web memungkinkan aplikasi ini tetap ringan dan mudah digunakan tanpa bergantung pada platform atau perangkat tertentu.

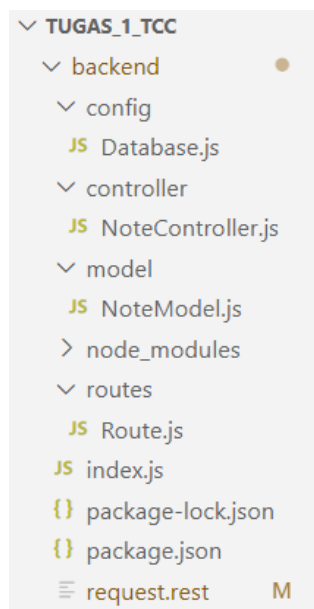
Dengan demikian, melalui pengembangan aplikasi ini, diharapkan dapat tercipta solusi pencatatan yang lebih modern, aman, serta dapat diadaptasi sesuai dengan kebutuhan individu maupun organisasi. Penugasan ini juga memberikan wawasan mengenai

bagaimana membangun sebuah sistem berbasis web yang terstruktur dan scalable dalam mendukung kebutuhan pencatatan digital masa kini.

3.2 Perancangan Solusi

Untuk mengatasi permasalahan yang telah diidentifikasi, solusi yang diterapkan dalam pengembangan aplikasi website Notes mencakup beberapa langkah berikut:

- Penggunaan Code Editor: Pengembangan kode aplikasi dilakukan menggunakan editor Visual Studio Code yang kompatibel dengan JavaScript dan Node.js.
- Pengembangan *Backend*: *Backend* dibangun dengan Node.js untuk menangani permintaan HTTP dari *frontend*, serta memanfaatkan Express.js sebagai framework guna membuat REST API yang mendukung operasi CRUD.



Gambar 3.1 Arsitektur *Backend*

- Penyimpanan Database: Data catatan (*notes*), termasuk id, judul, isi, kategori, tanggal pembuatan, dan tanggal pembaruan, disimpan dalam MySQL yang berjalan di server lokal menggunakan XAMPP.

Showing rows 0 - 6 (7 total, Query took 0.0004 seconds.)

SELECT * FROM `notes`

Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	id	judul	isi	kategori	Tanggal_dibuat	Tanggal_diupdate
<input type="checkbox"/> Edit Copy Delete	1	Rencana Kegiatan Mingguan	Minggu ini saya akan menyelesaikan laporan proyek ...	Kegiatan harian	2025-02-28 03:04:49	2025-03-01 07:17:54
<input type="checkbox"/> Edit Copy Delete	2	Ide Pengembangan Sistem Keamanan	Saya ingin menambahkan fitur notifikasi ke sistem ...	Teknologi	2025-02-28 03:11:16	2025-02-28 03:11:16
<input type="checkbox"/> Edit Copy Delete	3	Pemrograman GUI di Python	Besok saya akan mencoba membuat GUI sederhana untu...	Pemrograman	2025-02-28 03:11:20	2025-03-02 05:53:16
<input type="checkbox"/> Edit Copy Delete	5	Ide Pengembangan Sistem Keamanan Jaringan	Saya ingin menambahkan fitur notifikasi ke sistem ...	Jaringan	2025-03-01 04:53:12	2025-03-01 04:53:12
<input type="checkbox"/> Edit Copy Delete	6	mancing	aku mau mancing ke danau	kegiatan	2025-03-01 05:02:14	2025-03-01 05:02:14
<input type="checkbox"/> Edit Copy Delete	8	Evaluasi Proyek Pergudangan	Perlu meninjau kembali progres pengembangan aplika...	Manajemen Proyek	2025-03-01 06:55:30	2025-03-01 06:55:30
<input type="checkbox"/> Edit Copy Delete	12	pergi ke malang	tanggal 31 aku akan rekreasi ke malang bersama tem...	hiburan	2025-03-02 05:56:45	2025-03-02 05:57:04

Gambar 3. 2 Database Catatan

- d. Pengembangan *Frontend*: Antarmuka aplikasi dikembangkan menggunakan HTML untuk struktur halaman, CSS untuk tampilan, serta JavaScript untuk meningkatkan interaktivitas.



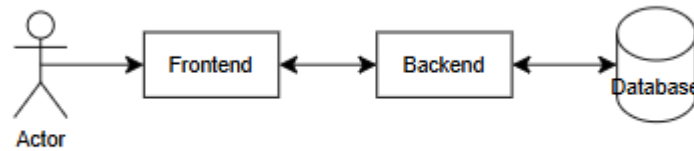
Gambar 3. 3 Arsitektur Frontend

- e. Pengujian API: Sebelum diintegrasikan dengan *frontend*, API yang telah dikembangkan diuji menggunakan *HTTP Client (REST Client)* guna memastikan fungsionalitasnya berjalan dengan baik.
- f. Validasi *Input* dan Penanganan Error: Data yang dimasukkan pengguna divalidasi sebelum dikirim ke database untuk memastikan keakuratan dan keamanan. Selain itu, mekanisme error handling diterapkan guna mencegah kesalahan dalam pemrosesan data.

3.2.1 Arsitektur sistem

Aplikasi menggunakan arsitektur berbasis klien-server dengan *frontend* sebagai antarmuka pengguna dan *Backend* sebagai pengelola data. *Database MySQL* digunakan untuk menyimpan catatan secara terstruktur. *RESTful API* dikembangkan menggunakan *Node.js* dan *Express* untuk komunikasi antara *frontend* dan *Backend*.

Berikut adalah diagram arsitektur yang menggambarkan hubungan antara komponen-komponen sistem:



Gambar 3. 4 Arsitektur Sistem

Berikut penjelasan alur dari arsitektur system yang telah dirancang:

a. *User → Frontend*

Pengguna berinteraksi dengan aplikasi melalui antarmuka pengguna (HTML, CSS, JavaScript).

b. *Frontend ↔ Backend*

Frontend mengirim permintaan (*request*) ke *backend* melalui RESTful API dan menerima respons dari *backend*.

c. *Backend ↔ Database (MySQL)*

Backend menangani logika bisnis dan berkomunikasi dengan database untuk menyimpan atau mengambil data.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil

4.1.1 Backend

Backend aplikasi ini telah berhasil dikembangkan menggunakan Node.js dan Express.js dengan MySQL sebagai basis data. API yang dibuat mampu menangani permintaan HTTP dengan metode GET, POST, PUT, dan DELETE, memungkinkan pengguna untuk mengelola catatan dengan mudah dan terstruktur.

Dalam implementasi sistem ini, koneksi ke database MySQL dilakukan menggunakan Sequelize. Dengan konfigurasi yang telah dibuat, sistem dapat berkomunikasi dengan database *catatan* yang berjalan di *localhost*.

Tabel 4. 1 Kode Program Database.js

```
import {Sequelize} from "sequelize";

const db = new Sequelize('catatan', 'root','',{
  host: 'localhost',
  dialect: 'mysql'
})

export default db;
```

Dalam pengembangan aplikasi ini, model *notes* telah berhasil dibuat menggunakan Sequelize sebagai ORM untuk mengelola database MySQL. Model ini mendefinisikan atribut judul, isi, dan kategori sebagai kolom dalam tabel. Selain itu, konfigurasi *freezeTableName* digunakan untuk memastikan nama tabel tetap *notes* tanpa perubahan. Fitur *timestamps* juga disesuaikan dengan penamaan khusus, yaitu *Tanggal_dibuat* dan *Tanggal_diupdate*, untuk mencatat waktu pembuatan dan pembaruan data.

Tabel 4. 2 Kode Program NoteModel.js

```
import {Sequelize} from "sequelize";
```

```

import db from "../config/Database.js";
import { title } from "process";
import sequelize from "sequelize";

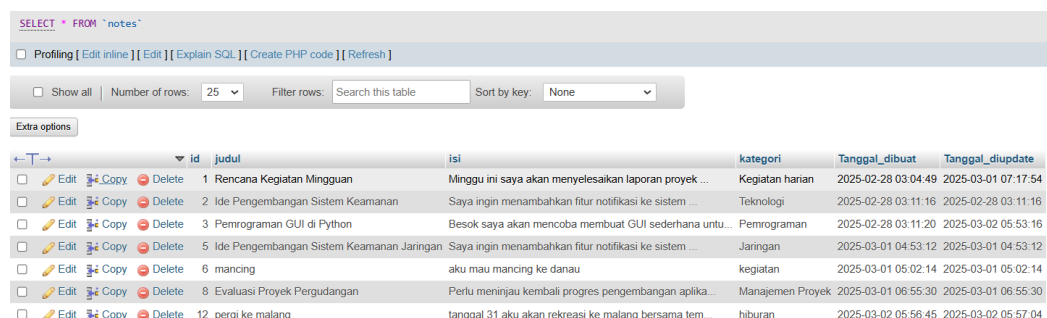
const note = db.define('notes',{
  judul:{type: Sequelize.STRING, allowNull:false},
  isi: Sequelize.STRING,
  kategori: Sequelize.STRING},{
    freezeTableName: true,
    createdAt : 'Tanggal_dibuat',
    updatedAt: 'Tanggal_diupdate'
  })
};

export default note;

(async()=>{
  await db.sync();})();

```

Dengan adanya kode tersebut, setiap kali aplikasi dijalankan, Sequelize akan memastikan tabel *notes* dibuat jika belum ada dalam *database*. Namun, data yang masuk tidak akan otomatis menyinkronkan struktur tabel yang sudah ada, melainkan hanya akan disimpan sesuai dengan skema yang telah ditentukan. Hal ini dapat dibuktikan dengan hasil tampilan berikut:



	id	judul	isi	kategori	Tanggal_dibuat	Tanggal_diupdate
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	Rencana Kegiatan Mingguan	Minggu ini saya akan menyelesaikan laporan proyek ...	Kegiatan harian	2025-02-28 03:04:49	2025-03-01 07:17:54
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	Ide Pengembangan Sistem Keamanan	Saya ingin menambahkan fitur notifikasi ke sistem ...	Teknologi	2025-02-28 03:11:16	2025-02-28 03:11:16
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	Pemrograman GUI di Python	Besok saya akan mencoba membuat GUI sederhana untu...	Pemrograman	2025-02-28 03:11:20	2025-03-02 05:53:16
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	5	Ide Pengembangan Sistem Keamanan Jaringan	Saya ingin menambahkan fitur notifikasi ke sistem ...	Jaringan	2025-03-01 04:53:12	2025-03-01 04:53:12
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	6	mancing	aku mau mancing ke danau	kegiatan	2025-03-01 05:02:14	2025-03-01 05:02:14
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	8	Evaluasi Proyek Pergudangan	Perlu meninjau kembali progres pengembangan aplika...	Manajemen Proyek	2025-03-01 06:55:30	2025-03-01 06:55:30
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	12	pergi ke malang	tanggal 31 aku akan rekreasi ke malang bersama tem...	hiburan	2025-03-02 05:56:45	2025-03-02 05:57:04

Gambar 4. 1 Tabel Notes

Dalam aplikasi *backend* ini, telah diimplementasikan berbagai fungsi untuk mengelola catatan menggunakan Sequelize sebagai ORM. API yang dikembangkan dapat menangani berbagai operasi *database*, seperti mengambil semua catatan (*getNotes*), menambahkan catatan baru (*createNote*), memperbarui catatan berdasarkan ID (*updateNote*), dan menghapus catatan (*deleteNote*). Setiap fungsi tersebut menangani permintaan HTTP dan memberikan respons sesuai dengan hasil eksekusi.

Tabel 4. 3 Kode Program NoteController.js

```
import {where} from "sequelize";
import Note from "../model/NoteModel.js";

export const getNotes = async(req, res)=>{
  try {
    const response = await Note.findAll();
    res.status(200).json(response);
  } catch (error) {
    console.log(error.message);
  }
}

//Buat Catatan
export const createNote = async(req, res)=>{
  try {
    await Note.create(req.body);
    res.status(201).json({msg: "Berhasil membuat catatan
baru....."});
  } catch (error) {
    console.log(error.message);
  }
}

//Mengedit Catatan
export const updateNote = async (req, res) => {
  try {
```

```

    const inputData = req.body
    const id = req.params.id

    await Note.update(inputData, {
      where: {
        id: id
      }
    });
    res.status(200).json({
      msg: "Berhasil mengupdate catatan",
    })
  } catch (error) {
    console.log(error.message)
  }
}

//Hapus Catatan
export const deleteNote = async (req, res) => {
  try {
    const id = req.params.id
    await Note.destroy({
      where: {
        id,
      },
    });
    res.status(200).json({
      message: "Berhasil dihapus",
    })
  } catch (error) {
    console.log(error.message)
  }
};

```

Rute (*routes*) menggunakan Express.js untuk menangani berbagai operasi terkait catatan. Rute-rute tersebut berfungsi sebagai endpoint API yang memungkinkan

pengguna berinteraksi dengan database. Rute `/notes` digunakan untuk mengambil semua catatan yang tersedia, sedangkan `/buat-catatan` digunakan untuk menambahkan catatan baru. Untuk memperbarui catatan, tersedia rute `/edit-catatan/:id`, dan rute `/hapus-catatan/:id` digunakan untuk menghapus catatan berdasarkan ID.

Tabel 4. 4 Kode Program Route.js

```
import express from "express";
import * as NoteController from
"../controller/NoteController.js";

const router = express.Router();

//endpoint get semua data user
router.get("/notes", NoteController.getNotes);

//endpoint create data user
router.post('/buat-catatan', NoteController.createNote);

//endpoint update data
router.put("/edit-catatan/:id", NoteController.updateNote)

//endpoint delete data
router.delete('/hapus-catatan/:id', NoteController.deleteNote)

export default router;
```

Pada implementasi *Backend* ini, server telah berhasil dikonfigurasi menggunakan Express.js. Server berjalan pada port 5000 dan dilengkapi dengan middleware *CORS* untuk mengizinkan akses dari berbagai sumber (*cross-origin resource sharing*). Middleware `express.json()` juga digunakan untuk menangani data dalam format JSON pada permintaan yang dikirimkan ke server. Seluruh rute API yang telah dibuat diatur melalui *Route.js*, yang di-*import* dan digunakan dalam aplikasi.

Tabel 4. 5 Kode Program index.js

```
import express from "express";
import cors from "cors";
import Route from "../routes/Route.js";

const app = express();
app.use(cors());
app.use(express.json());
app.use(Route);

app.listen(5000, ()=>console.log('Server up and running . . .
'));
```

Pengujian API dilakukan untuk memastikan bahwa fitur-fitur CRUD (*Create, Read, Update, Delete*) dapat berjalan dengan baik.

Tabel 4. 6 Kode Program Request.rest

```
GET http://localhost:5000/notes

###
POST http://localhost:5000/buat-catatan
Content-Type: application/json

{
  "judul" : "Rencana Hari Senin",
  "isi" : "Senin saya akan pergi ke rumah nenek di
Gunungkidul",
  "kategori" : "Kegiatan"
}

###
PUT http://localhost:5000/edit-catatan/3
Content-Type: application/json
```



```
{
  "judul" : "Pemrograman GUI di Python dengan vs code",
  "isi" : "Besok saya akan mencoba membuat GUI sederhana untuk
menangkap gambar dari webcam menggunakan Python",
  "kategori" : "Pemrograman"
}

###

delete http://localhost:5000/hapus-catatan/6
```

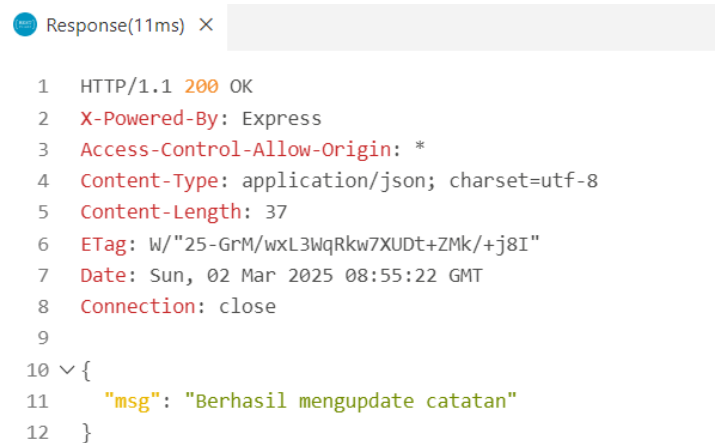
Berikut hasil pengujian API yang dilakukan dengan kode request.rest untuk pembuatan catatan baru melalui server *backend*.



```
Response(18ms) X
1 HTTP/1.1 201 Created
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 44
6 ETag: W/"2c-FKYfCcTkawyH0FHWcv6QNht3kDk"
7 Date: Sun, 02 Mar 2025 08:54:34 GMT
8 Connection: close
9
10 {
11   "msg": "Berhasil membuat catatan baru....."
12 }
```

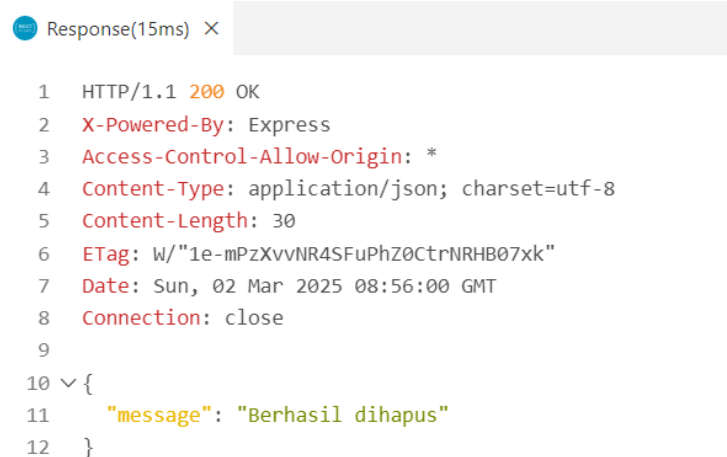
Gambar 4. 2 Hasil request method POST

Berikut hasil pengujian API yang dilakukan dengan kode request.rest untuk mengedit catatan melalui server *backend*.



Gambar 4. 3 Hasil request method PUT

Berikut hasil pengujian API yang dilakukan dengan kode request.rest untuk menghapus catatan melalui server *backend*.



Gambar 4. 4 Hasil request method DELETE

Berikut hasil pengujian API yang dilakukan dengan kode request.rest untuk mengambil data semua catatan melalui server *backend*.

```

1  HTTP/1.1 200 OK
2  X-Powered-By: Express
3  Access-Control-Allow-Origin: *
4  Content-Type: application/json; charset=utf-8
5  Content-Length: 2141
6  ETag: W/"85d-nBFa5GNX+cgDk7gfv04Xzv5TL4"
7  Date: Sun, 02 Mar 2025 09:03:46 GMT
8  Connection: close
9
10 √ [
11 √ {
12   "id": 1,
13   "judul": "Rencana Kegiatan Mingguan",
14   "isi": "Minggu ini saya akan menyelesaikan laporan proyek menghadiri
        rapat dengan tim dan mempelajari konsep baru tentang IoT Target utama ada
        lah menyelesaikan dokumentasi sebelum hari Jumat",
15   "kategori": "Kegiatan harian",
16   "Tanggal_dibuat": "2025-02-28T03:04:49.000Z",
17   "Tanggal_diupdate": "2025-03-01T07:17:54.000Z"

```

Gambar 4. 5 Hasil request method GET

4.1.2 *Frontend*

Website Note merupakan sistem pencatatan berbasis web yang dirancang untuk memudahkan pengguna dalam menambahkan, mengedit, dan menghapus catatan. Aplikasi ini memiliki tampilan yang sederhana dan responsif, dengan *frontend* yang dikembangkan menggunakan HTML, CSS, dan JavaScript, serta memanfaatkan Axios untuk berkomunikasi dengan *backend*.

Kode HTML ini merupakan tampilan antarmuka untuk aplikasi pencatatan yang memungkinkan pengguna menambahkan, mengedit, dan menghapus catatan. Bagian pertama berisi *form input* untuk mengisi judul, isi, dan kategori catatan, serta tombol simpan untuk menyimpan data. Bagian kedua menampilkan daftar catatan dalam bentuk tabel dengan opsi edit dan hapus pada setiap barisnya. Desain menggunakan Bootstrap agar tampilan lebih menarik, sementara Axios (melalui *index.js*) digunakan untuk mengelola komunikasi dengan *backend* dalam mengambil dan memodifikasi data catatan.

Tabel 4. 7 Kode Program *index.html*

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">

```

```

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhJY6hW+ALEwIH"
crossorigin="anonymous">
<title>Note</title>
<style>
    .btn-edit {
        background-color: yellow;
        color: black;
        border: none;
        border-radius: 5px;
        padding: 5px 10px;
        cursor: pointer;
    }

    .btn-hapus {
        background-color: red;
        color: white;
        border: none;
        border-radius: 5px;
        padding: 5px 10px;
        cursor: pointer;
    }
</style>
</head>
<body style="background-color: #2d6a6a; padding: 20px; color: white;">
    <div style="display: flex; gap: 2rem;">
        <div style="flex: 1;">
            <h2 style="color: white;">Buat Catatan</h2>
            <form>
                <span>
                    <label for="judul">Judul</label>
                    <textarea

```

```

        data-id
        id="judul"
        type="text"
        placeholder="Masukkan judul"
        required
        style="display: block; width: 100%;
height: 50px;"
    ></textarea>
</span>
<span style="margin-top: 1rem; display: block;">
    <label for="isi">Isi</label>
    <textarea
        data-id
        id="isi"
        type="text"
        placeholder="Masukkan catatan"
        required
        style="display: block; width: 100%;
height: 100px;"
    ></textarea>
</span>
<span style="margin-top: 1rem; display: block;">
    <label for="kategori">Kategori</label>
    <textarea
        data-id
        id="kategori"
        type="text"
        placeholder="Masukkan kategori"
        required
        style="display: block; width: 100%;
height: 50px;"
    ></textarea>
</span>
    <button type="submit" class="btn btn-primary"
style="margin-top: 1rem;">Simpan</button>

```

```

        </form>
    </div>

    <div style="flex: 2;">
        <h2>Daftar Catatan</h2>
        <main style="margin-top: 1rem">
            <table <table class="table table-success table-
bordered table-hover">
                <thead style="text-align: center;">
                    <tr>
                        <th>No</th>
                        <th>Judul</th>
                        <th>Isi</th>
                        <th>Kategori</th>
                        <th colspan="2">Ops</th>
                    </tr>
                </thead>
                <tbody id="table-catatan"></tbody>
            </table>
        </main>
        <script
src="https://cdn.jsdelivr.net/npm/axios/dist/axios.min.js"></scri
pt>
        <script src="index.js"></script>
    </div>
</div>
</body>
</html>

```

Buat Catatan

Judul
Masukkan judul

Isi
Masukkan catatan

Kategori
Masukkan kategori

Simpan

Daftar Catatan

No	Judul	Isi	Kategori	Opsi
1	Rencana Kegiatan Mingguan	Minggu ini saya akan menyelesaikan laporan proyek menghadiri rapat dengan tim dan mempelajari konsep baru tentang IoT Target utama adalah menyelesaikan dokumentasi sebelum hari Jumat	Kegiatan harian	Edit Hapus
2	Ide Pengembangan Sistem Keamanan	Saya ingin menambahkan fitur notifikasi ke sistem keamanan Arduino menggunakan modul WiFi ESP8266 agar bisa mengirim peringatan ke ponsel. Perlu mencari referensi dan melakukan uji coba.	Teknologi	Edit Hapus
3	Pemrograman GUI di Python dengan vs code	Besok saya akan mencoba membuat GUI sederhana untuk menangkap gambar dari webcam menggunakan Python	Pemrograman	Edit Hapus
4	Ide Pengembangan Sistem Keamanan Jaringan	Saya ingin menambahkan fitur notifikasi ke sistem keamanan Arduino menggunakan modul WiFi ESP8266 agar bisa mengirim peringatan ke ponsel. Perlu mencari referensi dan melakukan uji coba.	Jaringan	Edit Hapus
5	Evaluasi Proyek Pergudangan	Perlu meninjau kembali progres pengembangan aplikasi pergudangan. Fokus pada efektivitas metode Scrum, kendala yang dihadapi, dan strategi untuk meningkatkan kolaborasi tim.	Manajemen Proyek	Edit Hapus
6	pergi ke malang	tanggal 31 aku akan rekreasi ke malang bersama teman-teman	hiburan	Edit Hapus
7	Rencana Hari Senin	Senin saya akan pergi ke rumah nenek di Gunungkidul	Kegiatan	Edit Hapus

Gambar 4. 6 Tampilan UI Website

Dalam mengelola fitur pencatatan dalam aplikasi berbasis web menggunakan Axios untuk berkomunikasi dengan *backend*. Saat pengguna mengisi formulir dan menekan tombol simpan, data akan dikirim ke server menggunakan metode POST jika itu catatan baru, atau PUT jika sedang mengedit catatan. Fungsi *getNote()* mengambil daftar catatan dari *backend* dan memperbarui tampilan tabel secara dinamis. Setiap catatan memiliki tombol edit dan hapus, yang memungkinkan pengguna untuk memperbarui atau menghapus catatan menggunakan metode PUT dan DELETE. Fungsi *showNotification()* menampilkan pesan notifikasi sementara setiap kali pengguna berhasil menambah, mengedit, atau menghapus catatan.

Tabel 4. 8 Kode Program index.js

```
const formulir = document.querySelector("form");
formulir.addEventListener("submit", (e) => {
    e.preventDefault();

    const elemen_judul = document.querySelector("#judul");
    const elemen_isi = document.querySelector("#isi");
    const elemen_kategori = document.querySelector("#kategori");

    const judul = elemen_judul.value;
    const isi = elemen_isi.value;
    const kategori = elemen_kategori.value;
```

```

const id = elemen_judul.dataset.id;

if (id == ""){
  axios
    .post("http://localhost:5000/buat-catatan",{judul, isi,
kategori})
    .then(()=>{
      elemen_judul.value = "";
      elemen_isi.value = "";
      elemen_kategori.value = "";

      getNote();
      showNotification("Catatan berhasil disimpan!",
"green");
    })
    .catch((error) => console.log(error.message));
}else{
  axios
    .put(`http://localhost:5000/edit-catatan/${id}`, {judul,
isi, kategori})
    .then(()=>{
      elemen_judul.value = "";
      elemen_isi.value = "";
      elemen_kategori.value = "";

      getNote();
      showNotification("Catatan berhasil diperbarui!",
"blue");
    })
    .catch((error) => console.log(error.message));
}
});

async function getNote(){

```



```

    try {
        const {data} = await
    axios.get("http://localhost:5000/notes");

        const table = document.querySelector("#table-catatan");
        let tampilan = "";
        let no = 1;

        for(const note of await data){
            tampilan += tampilkanCatatan(no, note);
            no++;
        }
        table.innerHTML = tampilan;
        hapusNote();
        editNote();
    } catch (error) {
        console.log(error.message);
    }
}

function tampilkanCatatan(no, note){
    return `
    <tr>
        <td>${no}</td>
        <td class="judul">${note.judul}</td>
        <td class="isi">${note.isi}</td>
        <td class="kategori">${note.kategori}</td>
        <td><button data-id=${note.id} class='btn-
edit'>Edit</button></td>
        <td><button data-id=${note.id} class='btn-
hapus'>Hapus</button></td>
    </tr>
    `;
}

```

```

function hapusNote() {
    const kumpulan_tombol_hapus =
document.querySelectorAll(".btn-hapus");

    kumpulan_tombol_hapus.forEach((btn) => {
        btn.addEventListener("click", () => {
            const id = btn.dataset.id;
            axios
                .delete(`http://localhost:5000/hapus-catatan/${id}`)
                .then(() => {
                    getNote();
                    showNotification("Catatan berhasil dihapus!",
"red");
                })
                .catch((error) => console.log(error));
        });
    });
}

function editNote() {
    const kumpulan_tombol_edit = document.querySelectorAll(".btn-
edit");

    kumpulan_tombol_edit.forEach((tombol_edit) => {
        tombol_edit.addEventListener("click", () => {

            const id = tombol_edit.dataset.id;
            const judul =
                tombol_edit.parentElement.parentElement.querySelector
(
                    ".judul"
                ).innerText;
            const isi =
                tombol_edit.parentElement.parentElement.querySelector
(

```

```

        ".isi"
    ).innerText;
    const kategori =
    tombol_edit.parentElement.parentElement.querySelector
(
        ".kategori"
    ).innerText;

    const elemen_judul =
document.querySelector("#judul");
    const elemen_isi = document.querySelector("#isi");
    const elemen_kategori =
document.querySelector("#kategori");

    elemen_judul.dataset.id = id;
    elemen_judul.value = judul;
    elemen_isi.value = isi;
    elemen_kategori.value = kategori;
    });
});
}

// Fungsi Notifikasi
function showNotification(message, color) {
    const notif = document.createElement("div");
    notif.innerText = message;
    notif.style.position = "fixed";
    notif.style.bottom = "20px";
    notif.style.right = "20px";
    notif.style.background = color;
    notif.style.color = "white";
    notif.style.padding = "10px 20px";
    notif.style.borderRadius = "5px";
    notif.style.boxShadow = "0px 0px 10px rgba(0,0,0,0.3)";

```

```
notif.style.zIndex = "1000";

document.body.appendChild(notif);

// Hilangkan notifikasi setelah 3 detik
setTimeout(() => {
    notif.remove();
}, 3000);
}
getNote();
```

4.2 Pembahasan

Berdasarkan hasil implementasi, sistem *web service* Notes telah berhasil mencapai tujuannya dalam mendukung penambahan, pengeditan, dan penghapusan catatan secara dinamis. Seluruh komponen, baik *frontend* maupun *backend*, berfungsi sesuai perencanaan dengan operasi CRUD yang berjalan optimal dan antarmuka yang responsif.

Dari sisi arsitektur, sistem menerapkan pemisahan *frontend*, *backend*, dan *database* untuk memastikan skalabilitas dan kemudahan pengelolaan. Node.js dengan Express.js serta MySQL dipilih sebagai *backend* dan database yang terbukti efektif dalam menangani request. Integrasi Axios juga mempercepat komunikasi antara *frontend* dan *backend*.

Pendekatan REST API yang digunakan memungkinkan fleksibilitas pengembangan lebih lanjut, sementara validasi *input* menjaga integritas data. Pengalaman dalam pengembangan ini memberikan wawasan tentang desain REST API yang efisien, struktur database yang terorganisir, serta optimasi komunikasi *frontend-backend*. Selain itu, tantangan dalam sinkronisasi data dan error handling menjadi fokus penting dalam pengembangan selanjutnya. Secara keseluruhan, sistem ini tidak hanya berfungsi dengan baik tetapi juga menjadi studi kasus yang mendalam dalam penerapan *web service* yang efisien dan *maintainable*.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil implementasi, sistem *web service Notes* telah berhasil mencapai tujuan utamanya dalam mendukung pengelolaan catatan secara dinamis, efisien, dan responsif. Dengan penerapan arsitektur berbasis REST API serta pemisahan antara *frontend*, *backend*, dan *database*, sistem ini mampu menjalankan operasi CRUD dengan optimal. Penggunaan Node.js dengan Express.js sebagai *Backend* serta MySQL sebagai database terbukti efektif dalam menangani request dan menyimpan data secara terstruktur. Integrasi Axios juga mempercepat komunikasi antara *frontend* dan *backend*, meningkatkan pengalaman pengguna. Secara keseluruhan, sistem ini layak untuk diimplementasikan dan memiliki potensi untuk dikembangkan lebih lanjut guna meningkatkan fungsionalitas serta efisiensinya.

5.2 Saran

Berdasarkan hasil evaluasi sistem, terdapat beberapa saran untuk pengembangan lebih lanjut sistem *web service Notes*:

- a. Implementasi autentikasi dan otorisasi pengguna agar hanya pengguna yang memiliki akses yang dapat mengelola catatan.
- b. Penggunaan enkripsi untuk melindungi data pengguna, terutama pada transmisi data sensitif.
- c. Menambahkan fitur pencarian agar pengguna dapat menemukan catatan dengan lebih cepat.
- d. Pengelompokan catatan berdasarkan kategori untuk kemudahan organisasi data.
- e. Menggunakan framework *frontend* seperti React atau Vue.js untuk pengalaman pengguna yang lebih interaktif.

DAFTAR PUSTAKA

- Amazon Web Services. (2024). Apa itu RESTful API? Amazon Web Services. Diakses pada 2 Maret 2025, dari <https://aws.amazon.com/id/what-is/restful-api/>
- Biznet Gio. (2022). Apa itu CSS? Biznet Gio. Diakses pada 2 Maret 2025, dari <https://www.biznetgio.com/news/apa-itu-css>
- Dicoding. (2020). Apa itu JavaScript? Fungsi dan contohnya. Dicoding. Diakses pada 2 Maret 2025, dari <https://www.dicoding.com/blog/apa-itu-javascript-fungsi-dan-contohnya/>
- DomaiNesia. (2023). HTML adalah: Pengertian, fungsi, dan cara kerjanya. DomaiNesia. Diakses pada 2 Maret 2025, dari <https://www.domainesia.com/berita/html-adalah/>
- Kusuma, A., & Prasetyo, B. (2021). Pengelolaan basis data dengan MySQL untuk aplikasi web. Jakarta: Penerbit Informatika.
- Rahardian, T. (2019). Konsep dan implementasi database dengan MySQL. Yogyakarta: Andi Publisher.
- TechTarget. RESTful API. TechTarget. Diakses pada 2 Maret 2025, dari <https://www.techtarget.com/searchapparchitecture/definition/RESTful-API>
- Telkom University. (2024). Apa itu HTML? Telkom University. Diakses pada 2 Maret 2025, dari <https://docif.telkomuniversity.ac.id/apa-itu-html/>
- Telkom University. ReactJS: Definisi, fitur, manfaat, cara kerja, dan keunggulan. Telkom University. Diakses pada 2 Maret 2025, dari <https://jakarta.telkomuniversity.ac.id/reactjs-definisi-fitur-manfaat-cara-kerja-dan-keunggulan/>
- Widjaja, R. (2020). Pemrograman basis data dengan MySQL dan PHP. Surabaya: Graha Ilmu.

LAMPIRAN

Lampiran 1 Repository Github

https://github.com/restirama147/Resti-Ramadhani_123220147_Tugas2