

LAPORAN PRAKTIKUM TEKNOLOGI CLOUD COMPUTING

WEB SERVICE (BACK-END AND FRONT-END)

IF - F



Disusun oleh

Nama : Windy Claudia Napitupulu

NIM : 123220029

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

**HALAMAN PENGESAHAN
HALAMAN PERSETUJUAN
LAPORAN PRAKTIKUM
TEKNOLOGI CLOUD COMPUTING**

WEB SERVICE (BACK-END DAN FRONT-END)

IF - F

Disusun Oleh :

Windy Claudia Napitupulu 123220029

Telah diperiksa dan disetujui oleh Asisten Praktikum Teknologi Cloud Computing

Pada tanggal : 26 Februari 2025

Menyetujui.

Asisten Praktikum

Asisten Praktikum

Faustina Chelloana Triatmojo

NIM.123210139

Berlyandhica Alam Febriwantoro

NIM.123210060

KATA PENGANTAR

Puji dan syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa, yang senantiasa melimpahkan rahmat dan bimbingan-Nya, sehingga saya dapat menyelesaikan Tugas Praktikum Teknologi Cloud Computing serta laporannya. Laporan ini berisi kumpulan tugas serta evaluasi hasil belajar selama praktikum berlangsung.

Kami juga mengucapkan terima kasih yang sebesar-besarnya kepada asisten praktikum yang selalu membimbing dan mengajarkan kami dalam pelaksanaan praktikum serta dalam penyusunan laporan ini. Kami menyadari bahwa laporan ini masih jauh dari sempurna, oleh karena itu, kami sangat mengharapkan kritik dan saran yang membangun untuk penyempurnaan laporan ini.

Kepada semua pihak yang telah membantu dalam penyusunan laporan ini, kami mengucapkan terima kasih. Semoga laporan ini dapat dimanfaatkan dengan sebaik-baiknya.

Yogyakarta, 26 Februari 2026

Windy Claudia Napitupulu

DAFTAR ISI

HALAMAN PENGESAHAN	ii
KATA PENGANTAR	iii
DAFTAR ISI	iv
DAFTAR TABEL	v
DAFTAR GAMBAR	vi
DAFTAR LAMPIRAN	vii
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	3
2.1 Literatur materi 1	3
2.2 Literatur materi 2	3
BAB III METODOLOGI	4
3.1 Analisis Permasalahan	4
3.2 Perancangan Solusi	4
BAB IV HASIL DAN PEMBAHASAN	5
4.1 Hasil	5
4.2 Pembahasan	7
BAB V PENUTUP	9
5.1 Kesimpulan	9
5.2 Saran	10
DAFTAR PUSTAKA	11

DAFTAR GAMBAR

Gambar 1. Pengujian API Create.....	6
Gambar 2. Pengujian API Read.....	6
Gambar 3. Pengujian API Update.....	7
Gambar 4. Pengujian API Delete.....	7

DAFTAR LAMPIRAN

Lampiran 1. Link Github.....	11
-------------------------------------	----

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dalam era digital saat ini, pencatatan informasi telah beralih dari metode konvensional ke media digital. Masyarakat semakin bergantung pada teknologi untuk mencatat, menyimpan, dan mengakses informasi dengan lebih efisien. Pencatatan manual dalam bentuk fisik sering kali menghadapi berbagai kendala seperti risiko kehilangan, keterbatasan ruang penyimpanan, serta kesulitan dalam pencarian dan pengelolaan data. Oleh karena itu, diperlukan solusi digital yang dapat membantu pengguna dalam mencatat dan mengelola informasi dengan lebih baik.

Seiring dengan perkembangan teknologi, pengembangan web service berbasis RESTful API menjadi salah satu solusi yang banyak digunakan dalam membangun aplikasi berbasis web. Dengan menggunakan web service, data dapat diakses dengan mudah dari berbagai platform, termasuk front-end yang bebas sesuai kebutuhan pengguna. Dalam konteks tugas praktikum ini, pembangunan sebuah web service untuk aplikasi catatan (notes) bertujuan untuk memberikan pengalaman pencatatan yang lebih terstruktur, efisien, serta dapat diakses secara fleksibel dari berbagai perangkat.

Teknologi yang digunakan dalam tugas ini adalah Express.js sebagai back-end framework untuk membangun REST API yang mendukung operasi CRUD (Create, Read, Update, Delete). Express.js dipilih karena sifatnya yang ringan, cepat, dan fleksibel dalam membangun server-side applications. Sementara itu, front-end dapat dikembangkan menggunakan teknologi yang bebas sesuai preferensi pengembang, yang kemudian akan dihubungkan dengan back-end melalui API. Dengan arsitektur ini, sistem dapat dikembangkan dengan pendekatan modular, sehingga memudahkan pengelolaan dan pengembangan di masa depan. Pemanfaatan teknologi ini diharapkan dapat memberikan pemahaman lebih dalam mengenai pengembangan web service berbasis RESTful API serta integrasi antara front-end dan back-end dalam sebuah aplikasi berbasis web.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, dapat dirumuskan beberapa permasalahan yang menjadi fokus dalam tugas praktikum ini:

1. Bagaimana cara membangun web service berbasis RESTful API yang dapat mengelola data catatan secara efektif?
2. Bagaimana memastikan sistem back-end yang dikembangkan memiliki fitur CRUD yang berfungsi dengan baik?
3. Bagaimana mengintegrasikan back-end dengan front-end agar dapat diakses secara fleksibel oleh pengguna?

1.3 Tujuan

1. Mengembangkan web service berbasis RESTful API menggunakan Express.js yang dapat mengelola data catatan secara efektif.
2. Mengimplementasikan operasi CRUD dalam back-end agar pengguna dapat membuat, membaca, memperbarui, dan menghapus catatan.
3. Mengintegrasikan back-end dengan agar aplikasi dapat berjalan secara optimal dan memberikan pengalaman pengguna yang baik.
4. Mempelajari konsep pengembangan web service serta penerapan arsitektur RESTful API dalam sistem informasi.

1.4 Manfaat

1. Meningkatkan keamanan akses terhadap data dalam aplikasi backend.
2. Mempermudah pengelolaan hak akses pengguna dengan penerapan RBAC.
3. Menyediakan backend dan frontend yang fleksibel dan dapat diintegrasikan dengan berbagai platform lain.

BAB II

TINJAUAN LITERATUR

2.1 Web Service dan RESTful API

Web service adalah layanan berbasis web yang memungkinkan komunikasi dan pertukaran data antar aplikasi yang berbeda melalui protokol HTTP. Salah satu pendekatan dalam pengembangan *web service* adalah dengan menggunakan *RESTful API* (*Representational State Transfer*), yang menawarkan komunikasi berbasis HTTP dengan prinsip stateless dan pemanfaatan metode standar seperti GET, POST, PUT, dan DELETE.

2.2 Express.js sebagai Back-End Framework

Express.js adalah kerangka kerja berbasis *Node.js* yang ringan dan fleksibel untuk membangun aplikasi *back-end*. Keunggulan utama dari *Express.js* adalah kemudahan dalam mengelola rute, middleware, dan integrasi dengan database serta layanan pihak ketiga.

2.3 Implementasi CRUD dalam RESTful API

Operasi CRUD (*Create, Read, Update, Delete*) adalah dasar dalam pengelolaan data dalam sistem berbasis *RESTful API*. Setiap operasi memiliki metode HTTP yang sesuai:

- *Create* → HTTP POST
- *Read* → HTTP GET
- *Update* → HTTP PUT/PATCH
- *Delete* → HTTP DELETE.

BAB III

METODOLOGI

3.1 Analisis Permasalahan

1. Buatlah sebuah web service (Back-end) dengan tema catatan (notes).
2. Tampilan Front-end dibebaskan (Wajib menggunakan Web, tech stack dibebaskan).
3. Pastikan Back-end memuat CRUD dan RESTful.
4. Pastikan Front-end dan Back-end tersambung dengan baik.

3.2 Perancangan Solusi

Pada tahap ini, pengembangan *back-end* akan menggunakan *Express.js*.

Langkah-langkah utama meliputi:

1. Menginisialisasi proyek *Node.js* dan menginstal *Express.js*.
2. Mendesain struktur database dan membuat model data.
3. Mengimplementasikan rute API untuk operasi *CRUD*.
4. Mengintegrasikan database untuk penyimpanan catatan.

Menyediakan dokumentasi API menggunakan Postman atau Swagger.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Hasil

4.1.1 Implementasi Back-End

- **Stuktur Projek**

backend/

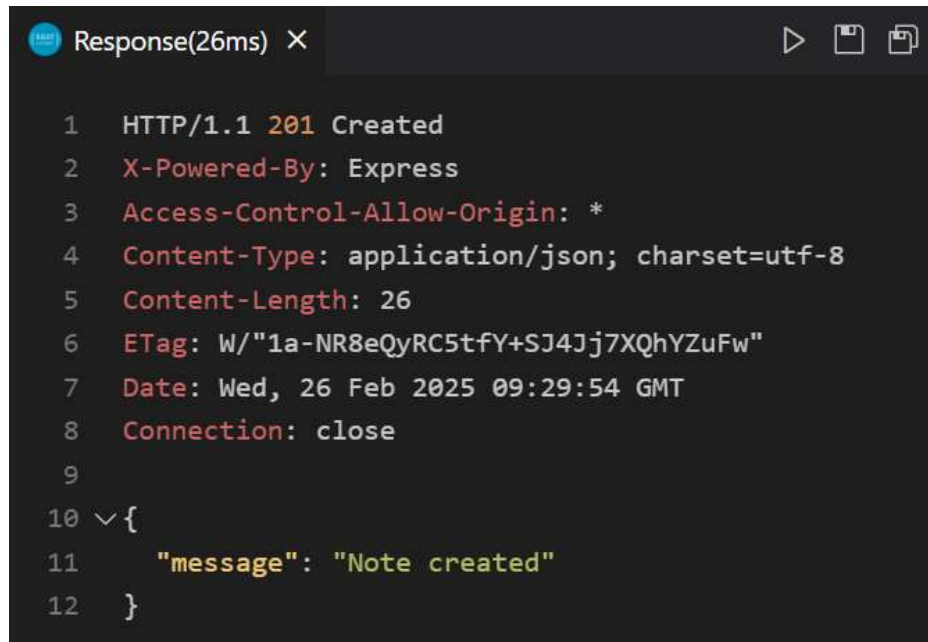
```
| — config/  
|   └─ Database.js  
| — controllers/  
|   └─ NoteController.js  
| — models/  
|   └─ NoteModel.js  
| — routes/  
|   └─ NoteRoutes.js  
| — index.js  
| — package.json  
| — package-lock.json  
| — request.rest
```

- **Pembuatan RESTful API:**

1. Implementasi server menggunakan Express.js.
2. Pembuatan rute untuk operasi CRUD (Create, Read, Update, Delete).
3. Integrasi dengan database

- Pengujian API:

1. Create



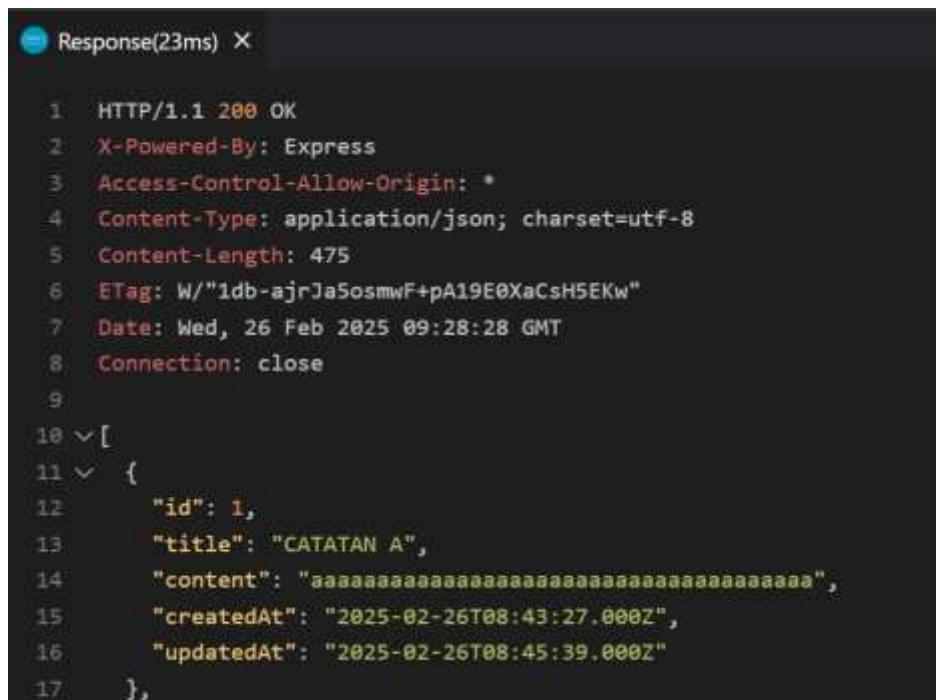
```

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: 26
6 ETag: W/"1a-NR8eQyRC5tfY+SJ4Jj7XQhYZuFw"
7 Date: Wed, 26 Feb 2025 09:29:54 GMT
8 Connection: close
9
10 {
11   "message": "Note created"
12 }

```

Gambar 1. Pengujian API Create

2. Read



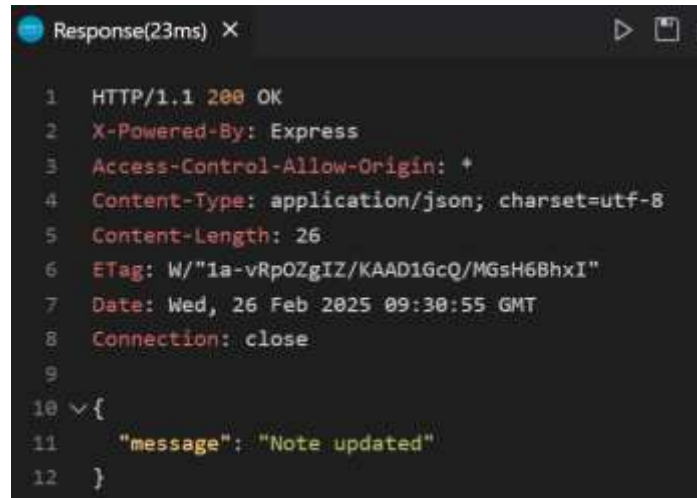
```

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: 475
6 ETag: W/"1db-ajrJaSosmwF+pA19E0XaCsH5EKw"
7 Date: Wed, 26 Feb 2025 09:28:28 GMT
8 Connection: close
9
10 [
11   {
12     "id": 1,
13     "title": "CATATAN A",
14     "content": "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa",
15     "createdAt": "2025-02-26T08:43:27.000Z",
16     "updatedAt": "2025-02-26T08:45:39.000Z"
17   },

```

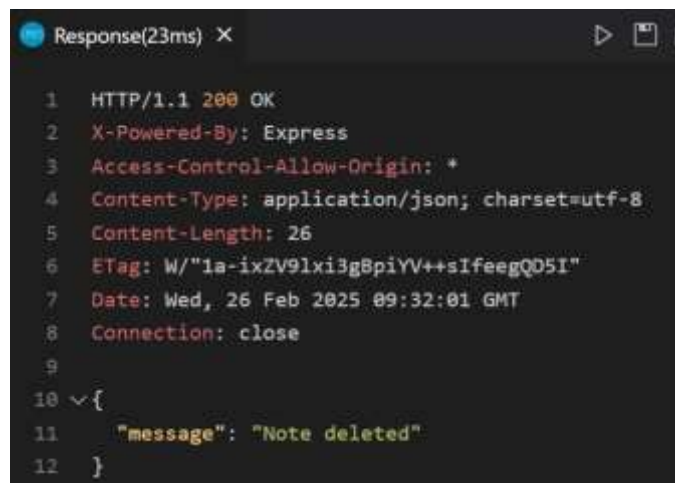
Gambar 2. Pengujian API Read

3. Update



Gambar 3. Pengujian API Update

4. Delete



Gambar 4. Pengujian API Delete

4.1.2 Implementasi Front-End

- Struktur Folder dan File

frontend/

| — scripts/

| └─ script.js

| — styles/

| └─ style.css

| — edit_note.html

| — index.html

- **Pembuatan Front-end**

1. Halaman Utama dan Menambahkan Catatan Baru

Pada halaman utama (index.html), pengguna dapat melihat daftar semua catatan yang telah disimpan. Saat halaman dimuat, sistem akan mengambil data catatan dari backend menggunakan fetch API dan menampilkannya di dalam elemen HTML. Jika tidak ada catatan yang tersedia, pengguna akan diberi informasi bahwa daftar masih kosong. Setiap catatan ditampilkan dalam bentuk kartu dengan dua tombol utama, yaitu tombol Edit untuk memperbarui catatan dan tombol Hapus untuk menghapus catatan.

Pengguna dapat menambahkan catatan baru melalui formulir yang tersedia. Formulir ini terdiri dari input untuk judul dan isi catatan, serta tombol Simpan. Setelah pengguna mengisi form dan menekan tombol simpan, data akan dikirim ke backend menggunakan metode POST melalui fetch API. Jika penyimpanan berhasil, catatan baru akan muncul di daftar catatan tanpa perlu melakukan refresh halaman.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-
scale=1" />
    <title>Notes Windy</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/boo
tstrap.min.css" rel="stylesheet" />
    <link rel="stylesheet" href="styles/style.css" />
  </head>
  <body>
    <nav class="navbar navbar-custom text-center">
      <div class="container-fluid">
        <a class="navbar-brand mx-auto fw-bold"
href="#">Notes</a>
      </div>
    </nav>

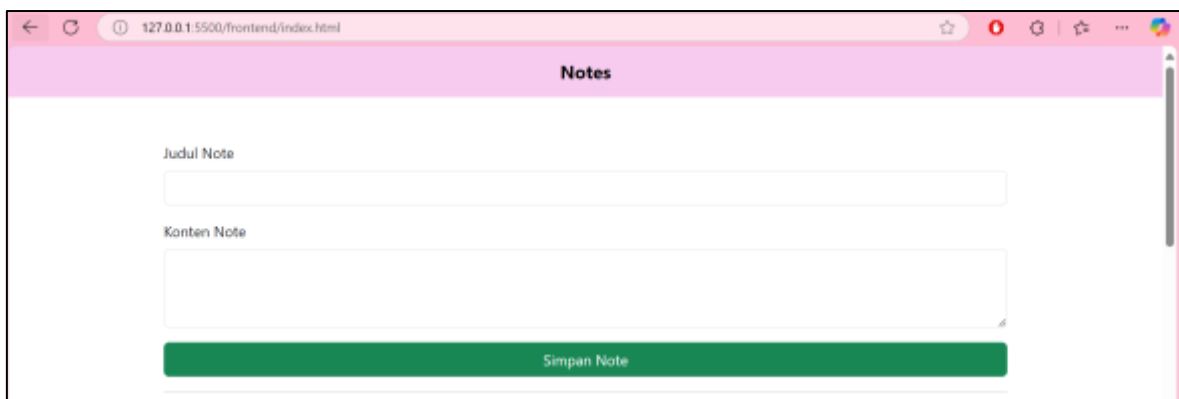
    <div class="container mt-5 w-75">
      <div class="mb-3">
        <label for="title" class="form-label">Judul Note</label>
        <input type="text" class="form-control" id="title" />
```

```

</div>
<div class="mb-3">
    <label for="konten" class="form-label">Konten
Note</label>
    <textarea class="form-control" id="konten"
rows="3"></textarea>
</div>
<div class="btn btn-success w-100"
onclick="saveNote()">Simpan Note</div>
<hr />
<div id="notesContainer"></div>
</div>

<script src="scripts/script.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootst
rap.bundle.min.js"></script>
</body>
</html>

```



Gambar 5. Halaman Utama

2. Mengedit Catatan

Ketika pengguna ingin mengedit catatan yang sudah ada, mereka dapat menekan tombol Edit pada catatan yang diinginkan. Hal ini akan mengarahkan pengguna ke halaman `edit_note.html`, di mana sistem akan mengambil data berdasarkan ID catatan yang dikirim melalui parameter URL. Setelah data ditampilkan, pengguna dapat mengubah judul dan isi catatan, lalu menekan tombol Simpan untuk memperbarui catatan. Perubahan akan dikirim

ke backend menggunakan metode PUT, dan setelah berhasil, pengguna akan diarahkan kembali ke halaman utama.

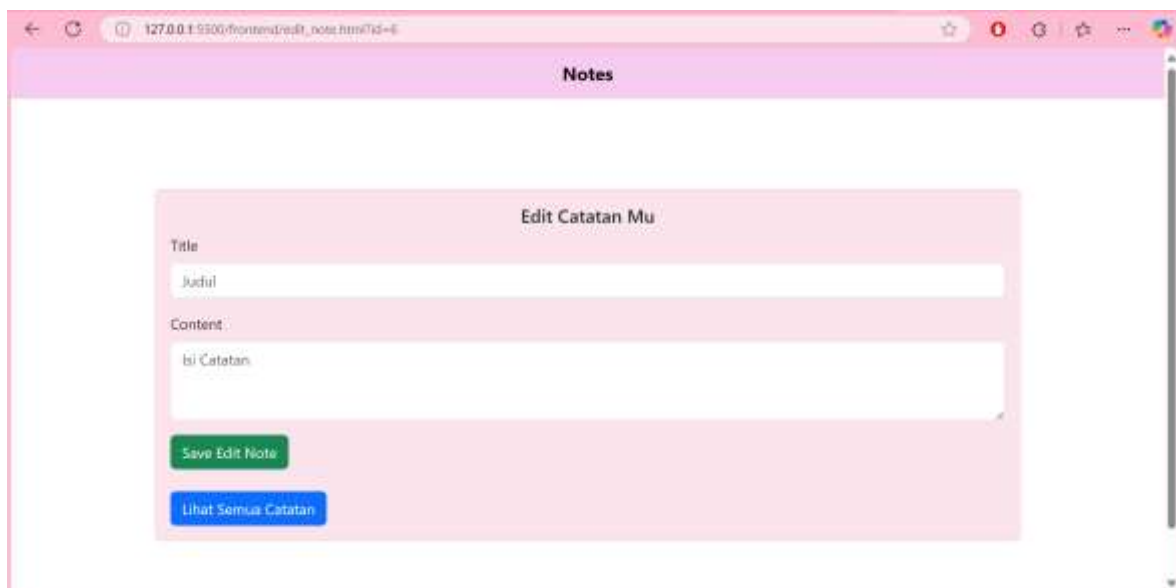
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <title>Notes Windy</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMHjY6hW+ALEwIH" crossorigin="anonymous" />
    <link rel="stylesheet" href="styles/style.css" />
  </head>
  <body>
    <nav class="navbar navbar-custom text-center">
      <div class="container-fluid">
        <a class="navbar-brand mx-auto fw-bold" href="#">Notes</a>
      </div>
    </nav>
    <div class="d-flex justify-content-center align-items-center full-height">
      <div class="card w-75 mb-3 card-custom">
        <div class="card-body">
          <center>
            <h5 class="card-title">Edit Catatan Mu</h5>
          </center>
          <form id="noteForm">
            <div class="mb-3">
              <label for="title" class="form-label">Title</label>
              <input type="text" class="form-control" id="title" name="title" placeholder="Judul" />
            </div>
            <div class="mb-3">
              <label for="content" class="form-label">Content</label>
              <textarea class="form-control" id="konten" name="konten" rows="3" placeholder="Isi Catatan"></textarea>
            </div>
          </form>
        </div>
      </div>
    </div>
  </body>
</html>
```



```

        </div>
        <button type="button" class="btn btn-success"
onclick="updateNote()">Save Edit Note</button><br /><br />
        <button type="button" class="btn btn-primary"
onclick="window.location.href='index.html'">Lihat Semua
Catatan</button>
    </form>
</div>
</div>
</div>
<script src="scripts/script.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bun
dle.min.js" integrity="sha384-
YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
crossorigin="anonymous"></script>
</body>
</html>

```



Gambar 6. Halaman Edit Notes

3. Menampilkan Daftar Catatan

Pada halaman utama (index.html), sistem akan secara otomatis mengambil dan menampilkan daftar catatan yang tersimpan di backend saat halaman dimuat. Proses ini

dilakukan dengan menggunakan fetch API yang mengirimkan permintaan GET ke endpoint backend yang menyimpan data catatan.

Setelah data diperoleh, catatan-catatan tersebut ditampilkan dalam bentuk kartu di dalam elemen `notesContainer`. Jika tidak ada catatan yang tersedia, sistem akan menampilkan pesan bahwa daftar masih kosong. Setiap catatan yang ditampilkan akan memiliki dua tombol utama, yaitu Edit dan Hapus, yang memungkinkan pengguna untuk mengelola catatan dengan mudah.

Tombol Edit akan mengarahkan pengguna ke halaman `edit_note.html`, dengan membawa ID catatan sebagai parameter di URL. Sementara itu, tombol Hapus akan memicu fungsi `deleteNote()` yang akan menghapus catatan tersebut dari backend dan memperbarui tampilan daftar catatan secara otomatis.

4.2 Pembahasan

4.2.1 Evaluasi Pencapaian Tujuan

Sistem yang dikembangkan telah berhasil mencapai tujuan awal, yaitu membangun aplikasi pencatatan yang memungkinkan pengguna untuk membuat, membaca, memperbarui, dan menghapus catatan (CRUD). API dan aplikasi catatan telah berfungsi sesuai dengan perancangan, di mana setiap operasi dilakukan secara real-time dan tersimpan dalam basis data.

Fitur utama seperti input catatan, pengeditan langsung menggunakan properti `contenteditable`, serta penghapusan catatan telah bekerja dengan baik. Interaksi antara front-end dan back-end berjalan tanpa hambatan, memastikan bahwa perubahan yang dilakukan oleh pengguna langsung tercatat dalam database tanpa perlu menyegarkan halaman.

4.2.2 Evaluasi Metode dan Teknologi

Penggunaan Express.js sebagai kerangka kerja untuk pengembangan RESTful API telah memberikan kemudahan dalam menangani permintaan HTTP serta mengelola data catatan. API yang dibangun mampu menangani permintaan CRUD dengan cepat dan efisien.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan analisis yang telah dilakukan, proyek ini merupakan aplikasi backend yang menggunakan arsitektur Model-View-Controller (MVC) untuk mengelola data catatan. Implementasi Role-Based Access Control (RBAC) telah diterapkan guna meningkatkan keamanan sistem dengan memastikan bahwa hanya pengguna yang memiliki izin tertentu yang dapat mengakses sumber daya. Teknologi yang digunakan, seperti Node.js, Express.js, dan PHPMyAdmin, dipilih karena fleksibilitas dan skalabilitasnya dalam pengelolaan data serta kemampuannya dalam mendukung integrasi dengan berbagai platform.

Dalam proses pengembangan proyek ini, beberapa aspek utama telah diimplementasikan dengan baik, seperti pengelolaan hak akses pengguna, struktur data yang jelas, serta endpoint REST API yang dapat digunakan untuk berbagai kebutuhan sistem informasi. Namun, masih terdapat beberapa peluang perbaikan dan pengembangan lebih lanjut guna meningkatkan efisiensi dan keamanan sistem.

5.2 Saran

Saran merupakan rekomendasi langkah selanjutnya untuk melanjutkan hasil yang telah dibuat.

- a) Implementasi JWT (JSON Web Token) dan enkripsi data lebih lanjut dapat digunakan untuk meningkatkan tingkat keamanan akses sistem.
- b) Penggunaan pengujian unit dan integrasi yang lebih komprehensif dapat memastikan sistem berjalan dengan lebih stabil.

DAFTAR PUSTAKA

- Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., & Chandramouli, R. (2001). Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3), 224-274. <https://doi.org/10.1145/501978.501979>
- Hunt, A., & Thomas, D. (1999). *The pragmatic programmer: Your journey to mastery*. Addison-Wesley.
- Martin, R. C. (2008). *Clean code: A handbook of agile software craftsmanship*. Prentice Hall.
- Node.js Foundation. (2023). *Node.js documentation*. Retrieved February 26, 2025, from <https://nodejs.org/en/docs>

Lampiran 1

Link Github : https://github.com/windyclaun/Notes_BackendFrontend.git