

**WEB SERVICE PRAKTEK**  
**LAPORAN PROJEK TUGAS BESAR WEB SERVICE**

Dosen Pengampu :

Aditya Ferdiana Arif, S.Kom., M.Kom.



Disusun Oleh:

5210311064 Febyan Putra Hermawan  
5210311071 Novendra Mustaka Fadil  
5210311075 Laurentius Andika Dwi Saputra  
5210311090 Faiz Muiz Zudin

**PROGRAM STUDI SISTEM INFORMASI SARJANA**  
**FAKULTAS SAINS DAN TEKNOLOGI**  
**UNIVERSITAS TEKNOLOGI YOGYAKARTA**  
**YOGYAKARTA**

**2023**

## Bab I

### - Persiapan Software dan System

Dalam pengembangan perangkat lunak, pemilihan teknologi yang tepat memainkan peran penting untuk mencapai keberhasilan proyek. Beberapa teknologi yang umumnya digunakan dalam pengembangan perangkat lunak modern termasuk React.js, Bootstrap, dan banyak lainnya.

- **React.js** adalah pustaka JavaScript yang sangat populer untuk membangun antarmuka pengguna (UI) yang dinamis dan responsif. Dikembangkan oleh Facebook, React memungkinkan pengembang untuk membuat komponen UI yang dapat diatur secara hierarkis dan di-refresh secara efisien ketika data berubah. Penggunaan React juga memungkinkan pengembangan aplikasi web yang mudah dipelihara dan diperbarui.
  - **Bootstrap** merupakan kerangka kerja (framework) front-end yang menyediakan berbagai komponen UI siap pakai. Dikembangkan oleh Twitter, Bootstrap memudahkan desain dan pengembangan antarmuka pengguna dengan menyediakan komponen-komponen yang dapat disesuaikan dan responsif. Penggunaan Bootstrap dapat membantu dalam memastikan bahwa aplikasi memiliki tata letak yang konsisten di berbagai perangkat.
  - **Node.js** adalah lingkungan runtime JavaScript yang dibangun di atas mesin JavaScript V8 dari Google Chrome. Ini memungkinkan penggunaan JavaScript di sisi server, memungkinkan pengembang untuk menulis kode JavaScript untuk mengelola logika server.
  - **Express.js** adalah framework server-side untuk Node.js. Ini menyediakan berbagai fitur dan alat yang memudahkan pengembangan aplikasi web dan API.
  - **MySQL** adalah sistem manajemen basis data relasional (RDBMS) yang populer. Ini menyimpan data dalam tabel terstruktur dan mendukung bahasa SQL untuk melakukan operasi database.
  - **Git** adalah sistem kontrol versi terdistribusi yang memungkinkan tim pengembang melacak perubahan dalam kode sumber proyek.
  - **VS Code** adalah editor sumber terbuka dan ringan yang dikembangkan oleh Microsoft. Ini menyediakan banyak fitur yang memudahkan penulisan dan debugging kode.
- Deskripsikan topik Project kalian dan penjelasan mengambil topik tersebut.

Topik Proyek yang kami buat yaitu tentang admin toko sepatu, kami mengambil topik ini karena Proyek yang kami pilih fokus pada pengembangan sistem administrasi untuk toko sepatu. Dalam era digital ini, efisiensi operasional menjadi kunci keberhasilan bisnis. Oleh karena itu, kami berencana untuk merancang sebuah platform administrasi yang dapat mengelola inventaris Sepatu

Salah satu aspek utama dari proyek ini adalah pengelolaan inventaris. Kami akan mengembangkan sistem untuk CRUD yaitu menambahkan, menghapus, dan mengelola stok sepatu

- Tentukan Database yang akan dibuat (SQL/No SQL)

Untuk proyek ini, telah dibuat sebuah database yang diberi nama "Pemograman\_web" dan "Sepatu\_Dev" Database ini dirancang untuk menyimpan informasi yang relevan terkait proyek yang sedang berlangsung.

- Buat desain rancangan Database yang akan dibuat :

Rancangan Database yang akan dibuat yaitu menggunakan dua database yang berbeda, yang satu untuk database Frontend dan Yang satu digunakan untuk Backend, karena memakai 2 database akan lebih terstruktur , tetapi memakai satu database juga bisa.

- Buat Desain Endpoint dengan minimal 4 Method: GET, POST, PATCH, DELETE. :

### Get All Sepatu

```

controller > JS sepatuController.js > findAllSepatus
1  const { Sepatu } = require('../models');
2
3  const findAllSepatus = async (req, res) => {
4      try {
5          const data = await Sepatu.findAll();
6          const result = {
7              status: 'ok',
8              data: data
9          };
10         res.json(result);
11     } catch (error) {
12         console.log('Error finding all Sepatus', error);
13         res.status(500).json({ status: 'error', message: 'Internal Server Error' });
14     }
15 };
16

```

### Get All Sepatu By Id

```

15 },
16
17 const getSepatuById = async (req, res) => {
18     try {
19         const { id } = req.params;
20         const data = await Sepatu.findPk(id);
21
22         if (data === null) {
23             return res.status(404).json({
24                 status: 'failed',
25                 message: `Data Sepatu with id ${id} is not found`
26             });
27         }
28
29         res.json({
30             status: 'ok',
31             data: data
32         });
33     } catch (error) {
34         console.log(error, 'Error get Sepatu by id');
35         res.status(500).json({
36             status: 'error',
37             message: 'Internal Server Error'
38         });
39     }
40 }

```

## Post New Sepatu

```
controller > JS sepatuController.js > [E] findAllSepatus
42 const createNewSepatu = async (req, res) => {
43   try {
44     // Mendapatkan request body
45     const { nama, merk, jenis, ukuran, harga, stok } = req.body;
46
47     // Membuat sepatu baru
48     const newSepatu = await Sepatu.create({ nama, merk, jenis, ukuran, harga, stok });
49
50     // Mengembalikan respons ke client
51     res.status(201).json({
52       status: 'ok',
53       data: {
54         id: newSepatu.id,
55         nama: newSepatu.nama,
56         merk: newSepatu.merk,
57         jenis: newSepatu.jenis,
58         ukuran: newSepatu.ukuran,
59         harga: newSepatu.harga,
60         stok: newSepatu.stok,
61         createdAt: newSepatu.createdAt,
62         updatedAt: newSepatu.updatedAt
63       }
64     });
65   } catch (error) {
66     console.log(error, 'Error create new Sepatu');
67     res.status(500).json({
68       status: 'error',
69       message: 'Internal Server Error'
70     });
71   }
72 };
73
```

## Put/ Patch Update Sepatu

```
73
74 const updateSepatu = async (req, res) => {
75   try {
76     // Mendapatkan req.params untuk mendapatkan data sepatu berdasarkan
77     const { id } = req.params;
78
79     // Mendapatkan req.body untuk mendapatkan data yang ingin diupdate
80     const { nama, merk, jenis, ukuran, harga, stok } = req.body;
81
82     // Mencari sepatu berdasarkan ID
83     const sepatu = await Sepatu.findByPk(id);
84
85     // Jika sepatu tidak ditemukan, kirim respons 404
86     if (!sepatu) {
87       return res.status(404).json({
88         status: 'failed',
89         message: `Data sepatu with id ${id} does not exist`
90       });
91     }
92
93     // Memperbarui data sepatu
94     sepatu.nama = nama;
95     sepatu.merk = merk;
96     sepatu.jenis = jenis;
97     sepatu.ukuran = ukuran;
98     sepatu.harga = harga;
99     sepatu.stok = stok;
100    sepatu.updatedAt = new Date();
101  }
102 }
```

```

1
2 // Menyimpan perubahan
3 await sepatu.save();
4
5 // Mengembalikan respons ke client
6 res.json({
7   status: 'ok',
8   data: {
9     id: sepatu.id,
10    nama: sepatu.nama,
11    merk: sepatu.merk,
12    jenis: sepatu.jenis,
13    ukuran: sepatu.ukuran,
14    harga: sepatu.harga,
15    stok: sepatu.stok,
16    createdAt: sepatu.createdAt,
17    updatedAt: sepatu.updatedAt
18  }
19 });
20 } catch (error) {
21   console.log(error, 'Error updating sepatu');
22   res.status(500).json({
23     status: 'error',
24     message: 'Internal Server Error'
25   });
26 }
27 };

```

## Delete Destroy Sepatu

```

controller > JS sepatusController.js > findAllSepatus
128
129 const destroySepatu = async (req, res) => {
130   try {
131     // Mendapatkan req.params untuk mendapatkan data sepatu berdasarkan
132     const { id } = req.params;
133
134     // Mencari sepatu berdasarkan ID
135     const sepatu = await Sepatu.findByPk(id);
136
137     // Jika sepatu tidak ditemukan, kirim respons 404
138     if (!sepatu) {
139       return res.status(404).json({
140         status: 'failed',
141         message: `Data sepatu with id ${id} does not exist`
142       });
143     }
144
145     // Menghapus sepatu
146     await sepatu.destroy();
147
148     // Mengembalikan respons ke client
149     res.json({
150       status: 'ok',
151       message: `Success delete sepatu with id ${id}`
152     });
153   } catch (error) {
154     console.log(error, 'Error destroy sepatu');
155     res.status(500).json({
156       status: 'error',
157       message: 'Internal Server Error'
158     });

```

- Framework dan packages yang digunakan
  - **Sequelize** adalah ORM (Object-Relational Mapping) untuk Node.js yang memungkinkan Anda berinteraksi dengan basis data relasional seperti MySQL, PostgreSQL, dan SQLite menggunakan objek JavaScript.
  - **Sequelize-cli** adalah alat baris perintah yang digunakan untuk memudahkan pengelolaan dan migrasi basis data Sequelize. Ini menyediakan perintah untuk membuat model, migrasi, dan banyak tugas lainnya terkait basis data.
  - **mysql2** adalah driver MySQL untuk Node.js. Ini memungkinkan aplikasi Node.js terhubung dan berinteraksi dengan basis data MySQL.
  - **Fastest-validator** adalah paket validasi untuk Node.js yang dirancang untuk memberikan validasi yang cepat dan efisien. Ini biasanya digunakan untuk memvalidasi data yang diterima dari pengguna atau sumber eksternal.
  - **Express generator** adalah alat yang memungkinkan Anda membuat struktur proyek Express.js secara otomatis. Ini menyederhanakan proses pembuatan aplikasi web dengan Express.js.
  - **File Config** Ini mungkin merujuk pada file konfigurasi yang digunakan dalam proyek Anda. File ini dapat berisi pengaturan seperti koneksi database, konfigurasi server, dan variabel-variabel lingkungan lainnya.
  - **Models** adalah bagian dari Sequelize yang mewakili struktur tabel dalam basis data relasional. Mereka mendefinisikan cara data disimpan dan diambil dari basis data.
  - **Node Module** Ini merujuk pada modul atau pustaka JavaScript yang digunakan dalam proyek Node.js. Modul ini dapat berupa pustaka pihak ketiga atau modul khusus yang Anda buat untuk proyek Anda sendiri.
  - **Routes** adalah bagian dari aplikasi Express.js yang menangani rute atau endpoint HTTP. Mereka menentukan cara respons ditangani ketika permintaan dikirim ke server pada rute tertentu.
  - **File .env** adalah tempat Anda dapat menyimpan variabel lingkungan proyek Anda. Ini adalah cara yang umum digunakan untuk menyimpan pengaturan sensitif seperti kunci API atau informasi koneksi basis data.
  - **App.js** Ini mungkin adalah file utama atau entry point dari aplikasi Node.js Anda. Biasanya, ini berisi konfigurasi dasar server dan pengaturan awal aplikasi.
  - **Package-lock.json** dan **package.json** Ini adalah file-file konfigurasi yang digunakan untuk mengelola dependensi proyek Node.js. **package.json** berisi daftar dependensi dan konfigurasi proyek, sementara **package-lock.json** memastikan versi dependensi yang konsisten di seluruh tim pengembang.
  - **React.js** adalah sebuah perpustakaan JavaScript yang populer digunakan untuk membangun antarmuka pengguna yang dinamis dan interaktif. memungkinkan pengembang untuk membuat komponen UI yang dapat diubah dengan efisien ketika terjadi perubahan data. Keunggulan utamanya terletak pada penggunaan Virtual DOM, yang membantu meningkatkan kinerja aplikasi dengan mengurangi manipulasi langsung terhadap DOM.
  - **Bootstrap**, di sisi lain, adalah kerangka kerja front-end yang open-source yang dikembangkan oleh Twitter. Dirancang untuk memudahkan pengembangan tata letak responsif dan desain web yang menarik, Bootstrap menyediakan

berbagai komponen dan gaya prabuatan, termasuk grid system, navigasi, formulir, dan lainnya. Hal ini memungkinkan pengembang untuk dengan cepat membangun situs web yang responsif dengan estetika yang baik.

- **Node.js** adalah lingkungan runtime JavaScript yang memungkinkan eksekusi JavaScript di sisi server. Dirancang untuk menangani aplikasi berbasis server dan berbasis jaringan, Node.js memberikan kecepatan eksekusi dan kemampuan untuk menangani banyak koneksi secara bersamaan. Ini membuatnya ideal untuk pengembangan aplikasi real-time dan berbasis kejadian.
- **Express.js**, sebagai kerangka kerja back-end untuk Node.js, menyederhanakan proses pengembangan aplikasi web dan API. Dengan desain yang minimalis dan fleksibel, Express.js menyediakan fitur dasar yang diperlukan untuk membuat aplikasi server dengan cepat. Kemampuannya untuk mendukung middleware memungkinkan integrasi fungsi tambahan ke dalam alur penanganan permintaan, meningkatkan fleksibilitas dan modularitas.

## Bab II

### 1. Langkah-langkah membuat project Backend Sepatu

Langkah pertama yang dilakukan dalam membuat project ialah membuat Package di GitBash dengan menggunakan Kode *\$mkdir PROJEK-WS-SEPATU*

Langkah selanjutnya yaitu masuk ke dalam project yang sudah dibuat dengan menggunakan kode *\$cd PROJEK-WS-SEPATU*

Langkah selanjutnya yaitu menginstal Package di dalam project yang sudah dibuat yaitu **Package.json** dan **Package-lock.json** dengan menggunakan kode *\$npm init*

Langkah selanjutnya yaitu menggunakan Perintah *\$npm install* digunakan untuk menginstal dependensi proyek yang didefinisikan dalam file package.json. Ketika Anda menjalankan perintah ini, npm akan membaca file package.json proyek Anda, mengidentifikasi paket-paket yang diperlukan, dan mengunduh serta menginstal versi paket tersebut.

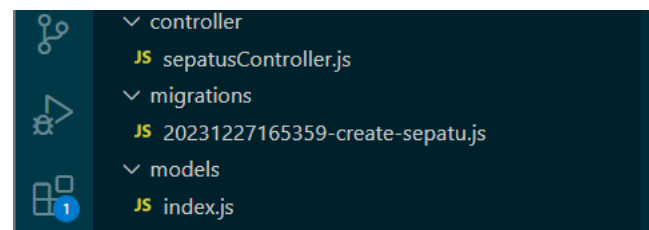
Langkah selanjutnya yaitu membuat Instalasi Library pendukung seperti

- \$npm install dotenv*
- \$npm install --save sequelize sequelize-cli*
- \$npm install sequelize init*
- \$npm install mysql2*
- \$npm install fastest-validator --save*

Di Config.json menambahkan beberapa baris kode

```
config > {} config.json > {} development > database
1  {
2    "development": {
3      "username": "root",
4      "password": null,
5      "database": "sepatudev",
6      "host": "127.0.0.1",
7      "dialect": "mysql"
8    },
9    "test": {
10     "username": "root",
11     "password": null,
12     "database": "sepatutest",
13     "host": "127.0.0.1",
14     "dialect": "mysql"
15   },
16   "production": {
17     "username": "root",
18     "password": null,
19     "database": "database_production",
20     "host": "127.0.0.1",
21     "dialect": "mysql"
22   }
23 }
24
```

Langkah selanjutnya yaitu **Migration**, untuk melakukan migrition yaitu dengan menggunakan kode `$npx sequelize migration:create --name= create-sepatu`



Untuk isi File migration Create-sepatu

```
File Edit Selection View Go ... < >
index.html katalog.html {} config.json JS 20231227165359-
migrations > JS 20231227165359-create-sepatu.js > ...
1  'use strict';
2  /** @type {import('sequelize-cli').Migration} */
3  module.exports = {
4    async up(queryInterface, Sequelize) {
5      await queryInterface.createTable('Sepatus', {
6        id: {
7          allowNull: false,
8          autoIncrement: true,
9          primaryKey: true,
10         type: Sequelize.INTEGER
11       },
12       nama: {
13         type: Sequelize.STRING
14       },
15       merk: {
16         type: Sequelize.STRING
17       },
18       jenis: {
19         type: Sequelize.STRING
20       },
21       ukuran: {
22         type: Sequelize.STRING
23       },
24     }
25   };
26
```



```

20     },
21     ukuran: {
22       type: Sequelize.STRING
23     },
24     harga: {
25       type: Sequelize.DOUBLE
26     },
27     stok: {
28       type: Sequelize.INTEGER
29     },
30     createdAt: {
31       allowNull: false,
32       type: Sequelize.DATE
33     },
34     updatedAt: {
35       allowNull: false,
36       type: Sequelize.DATE
37     }
38   });
39 },
40 async down(queryInterface, Sequelize) {
41   await queryInterface.dropTable('Sepatus');
42 }
43 };

```

Setelah dibuat, masuk kedalam migrasi dengan menggunakan `$npx sequelize db:migrate`

Setelah itu , masuk ke tahap selajutnya yaitu membuat **seeders** ini digunakan untuk memasukan data ke dalam table di database yang sudah dibuat, cara nya dengan menggunakan `$npx sequelize seed:create --name=demo-sepatu` dan `$npm install bcrypt --save`.

```

> routes
▼ seeders
JS 20231227165606-demo-sepatu.js
JS index.js
{} package-lock.json

```

Dari kode diatas akan terbentuk file seeders dan untuk isianya sebagai berikut

```

index.html  katalog.html  {} config.json  JS 20231227165606-demo-s
seeders > JS 20231227165606-demo-sepatu.js > ...
1  'use strict';
2
3  /** @type {import('sequelize-cli').Migration} */
4  module.exports = {
5    async up (queryInterface, Sequelize) {
6      return queryInterface.bulkInsert('Sepatus', [
7        {
8          nama: 'Sepatu A',
9          merk: 'Adindut',
10         jenis: 'Sneakers',
11         ukuran: '42',
12         harga: 150.0,
13         stok: 50,
14         createdAt: new Date(),
15         updatedAt: new Date()
16       },
17       {
18         nama: 'Sepatu B',
19         merk: 'nayk',
20         jenis: 'Boots',
21         ukuran: '39',
22         harga: 120.0,
23         stok: 30,
24         createdAt: new Date(),
25         updatedAt: new Date()
26       },
27       // Tambahkan data sepatu lainnya sesuai kebutuhan
28     ]);
29   },
30 },
31
32   async down (queryInterface, Sequelize) {

```

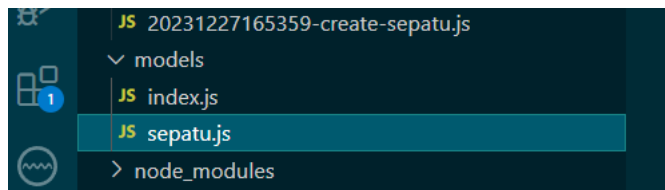
Eksekusi file seeders menggunakan `$npx sequelize db:seed:all`

## Hasil Eksekusi

The screenshot shows the phpMyAdmin interface for a database named 'sepatudev'. The 'sepatu' table is selected, and the SQL tab is active. The table contains 6 rows of data. The SQL query shown is `SELECT * FROM `sepatu``. The table structure is as follows:

id	nama	merk	jenis	ukuran	harga	stok	createdAt	updatedAt
1	Sepatu A	Adindut	Sneakers	42	150	50	2024-01-04 06:25:35	2024-01-04 06:25:35
2	Sepatu B	nayk	Boots	39	120	30	2024-01-04 06:25:35	2024-01-04 06:25:35
3	Nikw Air KW	Nike	Formal	45	12000	1	2024-01-06 03:41:35	2024-01-06 03:41:35
4	Sepatu A	Adindut	Sneakers	42	150	50	2024-01-08 06:48:46	2024-01-08 06:48:46
5	Sepatu B	nayk	Boots	39	120	30	2024-01-08 06:48:46	2024-01-08 06:48:46
6	irvan	Adidas	Olahraga	23	2323	232	2024-01-10 02:14:08	2024-01-10 02:14:08

Langkah berikutnya yaitu dengan membuat Model Database dengan cara membuat file Sepatus.js

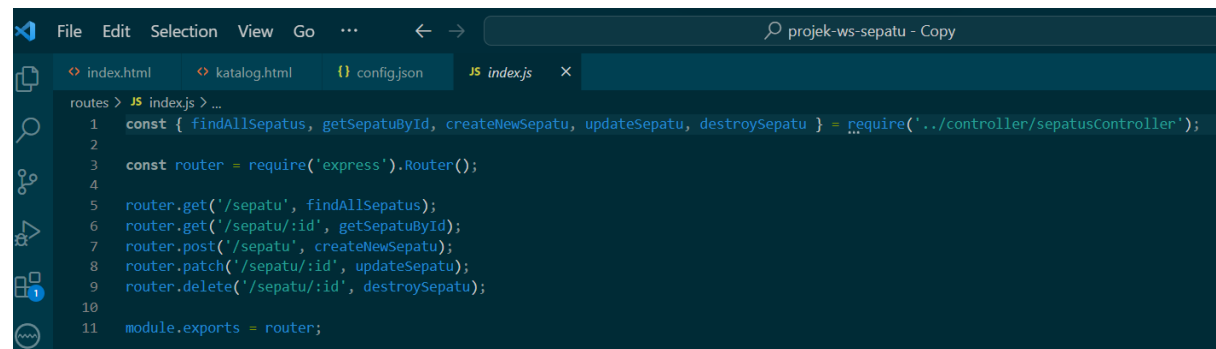


Untuk Isi dari Sepatus.js seperti ini

```
models > JS sepatu.js > ...
1  'use strict';
2  const {
3    Model
4  } = require('sequelize');
5  module.exports = (sequelize, DataTypes) => {
6    class Sepatu extends Model {
7      /**
8       * Helper method for defining associations.
9       * This method is not a part of Sequelize lifecycle.
10       * The `models/index` file will call this method automatically.
11       */
12      static associate(models) {
13        // define association here
14      }
15    }
16    Sepatu.init({
17      id: {
18        type: DataTypes.INTEGER,
19        primaryKey: true,
20        autoIncrement: true,
21        allowNull: false,
22      },
23      nama: {
24        type: DataTypes.STRING,
25      },
26      merk: {
27        type: DataTypes.STRING,
28      },
29      jenis: {
30        type: DataTypes.STRING,
31      },
32      ukuran: {
```

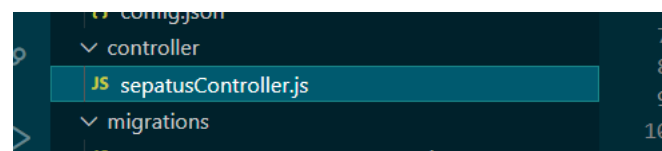
```
    },
    ukuran: {
      type: DataTypes.STRING,
    },
    harga: {
      type: DataTypes.DOUBLE,
    },
    stok: {
      type: DataTypes.INTEGER,
    },
  }, {
    sequelize,
    modelName: 'Sepatu',
  });
  return Sepatu;
};
```

Langkah selajutnta yaitu membuat File di dalam routes yaitu index.js dengan kode didalam index.js adalah sebagai berikut



```
File Edit Selection View Go ... < > projek-ws-sepatu - Copy
index.html katalog.html config.json JS index.js x
routes > JS index.js > ...
1 const { findAllSepatus, getSepatuById, createNewSepatu, updateSepatu, destroySepatu } = require('../controller/sepatusController');
2
3 const router = require('express').Router();
4
5 router.get('/sepatu', findAllSepatus);
6 router.get('/sepatu/:id', getSepatuById);
7 router.post('/sepatu', createNewSepatu);
8 router.patch('/sepatu/:id', updateSepatu);
9 router.delete('/sepatu/:id', destroySepatu);
10
11 module.exports = router;
```

Langkah selanjutnya yaitu dengan menambahkan Controller dan File sepatusController.js ini berfungsi untuk CRUD data yang ada di dalam database



### Kode Controller



```
controller > JS sepatusController.js > findAllSepatus
1 const { Sepatu } = require('../models');
2
3 const findAllSepatus = async (req, res) => {
4   try {
5     const data = await Sepatu.findAll();
6     const result = {
7       status: 'ok',
8       data: data
9     };
10    res.json(result);
11  } catch (error) {
12    console.log('Error finding all Sepatus', error);
13    res.status(500).json({ status: 'error', message: 'Internal Server Error' });
14  }
15 };
16
```

```

15 },
16
17 const getSepatuById = async (req, res) => {
18   try {
19     const { id } = req.params;
20     const data = await Sepatu.findByPk(id);
21
22     if (data === null) {
23       return res.status(404).json({
24         status: 'failed',
25         message: `Data Sepatu with id ${id} is not found`
26       });
27     }
28
29     res.json({
30       status: 'ok',
31       data: data
32     });
33   } catch (error) {
34     console.log(error, 'Error get Sepatu by id');
35     res.status(500).json({
36       status: 'error',
37       message: 'Internal Server Error'
38     });
39   }
40 }

```

controller > JS sepatuController.js > findAllSepatus

```

42 const createNewSepatu = async (req, res) => {
43   try {
44     // Mendapatkan request body
45     const { nama, merk, jenis, ukuran, harga, stok } = req.body;
46
47     // Membuat sepatu baru
48     const newSepatu = await Sepatu.create({ nama, merk, jenis, ukuran, harga, stok });
49
50     // Mengembalikan respons ke client
51     res.status(201).json({
52       status: 'ok',
53       data: {
54         id: newSepatu.id,
55         nama: newSepatu.nama,
56         merk: newSepatu.merk,
57         jenis: newSepatu.jenis,
58         ukuran: newSepatu.ukuran,
59         harga: newSepatu.harga,
60         stok: newSepatu.stok,
61         createdAt: newSepatu.createdAt,
62         updatedAt: newSepatu.updatedAt
63       }
64     });
65   } catch (error) {
66     console.log(error, 'Error create new Sepatu');
67     res.status(500).json({
68       status: 'error',
69       message: 'Internal Server Error'
70     });
71   }
72 };

```

```

1 //
2
3
4 const updateSepatu = async (req, res) => {
5   try {
6     // Mendapatkan req.params untuk mendapatkan data sepatu berdasarkan
7     const { id } = req.params;
8
9     // Mendapatkan req.body untuk mendapatkan data yang ingin diupdate
10    const { nama, merk, jenis, ukuran, harga, stok } = req.body;
11
12    // Mencari sepatu berdasarkan ID
13    const sepatu = await Sepatu.findById(id);
14
15    // Jika sepatu tidak ditemukan, kirim respons 404
16    if (!sepatu) {
17      return res.status(404).json({
18        status: 'failed',
19        message: `Data sepatu with id ${id} does not exist`
20      });
21    }
22
23    // Memperbarui data sepatu
24    sepatu.nama = nama;
25    sepatu.merk = merk;
26    sepatu.jenis = jenis;
27    sepatu.ukuran = ukuran;
28    sepatu.harga = harga;
29    sepatu.stok = stok;
30    sepatu.updatedAt = new Date();
31

```

```

32
33    // Menyimpan perubahan
34    await sepatu.save();
35
36    // Mengembalikan respons ke client
37    res.json({
38      status: 'ok',
39      data: {
40        id: sepatu.id,
41        nama: sepatu.nama,
42        merk: sepatu.merk,
43        jenis: sepatu.jenis,
44        ukuran: sepatu.ukuran,
45        harga: sepatu.harga,
46        stok: sepatu.stok,
47        createdAt: sepatu.createdAt,
48        updatedAt: sepatu.updatedAt
49      }
50    });
51  } catch (error) {
52    console.log(error, 'Error updating sepatu');
53    res.status(500).json({
54      status: 'error',
55      message: 'Internal Server Error'
56    });
57  }
58 };

```

```

controller > JS sepatuController.js > findAllSepatu
128
129 const destroySepatu = async (req, res) => {
130   try {
131     // Mendapatkan req.params untuk mendapatkan data sepatu berdasarkan
132     const { id } = req.params;
133
134     // Mencari sepatu berdasarkan ID
135     const sepatu = await Sepatu.findByPk(id);
136
137     // Jika sepatu tidak ditemukan, kirim respons 404
138     if (!sepatu) {
139       return res.status(404).json({
140         status: 'failed',
141         message: `Data sepatu with id ${id} does not exist`
142       });
143     }
144
145     // Menghapus sepatu
146     await sepatu.destroy();
147
148     // Mengembalikan respons ke client
149     res.json({
150       status: 'ok',
151       message: `Success delete sepatu with id ${id}`
152     });
153   } catch (error) {
154     console.log(error, 'Error destroy sepatu');
155     res.status(500).json({
156       status: 'error',
157       message: 'Internal Server Error'
158     });
159   }
160 }

```

Selanjutnya untuk membuat backend Tampilan web, kita instal main.js dan katalog.js

Untuk kode main.js (Ini untuk CRUD Sepatu)

```

// main.js
const baseUrl = 'http://localhost:3333'; // Sesuaikan dengan port server Anda

// Fungsi untuk mendapatkan semua sepatu
async function getAllSepatu() {
  try {
    const response = await fetch('http://localhost:3333/sepatu');
    const data = await response.json();
    console.log('Data dari server:', data);
    displaySepatuInTable(data);
  } catch (error) {
    console.error('Error:', error.message);
  }
}

```

```

// Fungsi untuk menampilkan data sepatu dalam tabel
function displaySepatuInTable(sepatuData) {
    const sepatuTableBody = $('#sepatuTableBody');
    sepatuTableBody.empty();

    console.log(typeof sepatuData);

    if (Array.isArray(sepatuData) || (sepatuData &&
Array.isArray(sepatuData.data))) {
        const sepatuList = Array.isArray(sepatuData) ? sepatuData :
sepatuData.data;

        sepatuList.forEach(sepatu => {
            const row = `
                <tr>
                    <td>${sepatu.id}</td>
                    <td>${sepatu.nama}</td>
                    <td>${sepatu.merk}</td>
                    <td>${sepatu.jenis}</td>
                    <td>${sepatu.ukuran}</td>
                    <td>Rp.${sepatu.harga}.000</td>
                    <td>${sepatu.stok}</td>
                    <td>
                        <button class="btn btn-primary"
onclick="editSepatu(${sepatu.id})" data-toggle="modal" data-
target="#editModal">Edit</button>
                        <button class="btn btn-danger"
onclick="confirmDelete(${sepatu.id})">Hapus</button>
                    </td>
                </tr>
            `;
            sepatuTableBody.append(row);
        });
    } else {
        console.error('Sepatu data is not an array.');
```



```

        success: function (response) {
            if (response.success) {
                const sepatu = response.data;

                // Mengisi formulir edit dengan data sepatu
                $('#editNama').val(sepatu.nama);
                $('#editMerk').val(sepatu.merk);
                $('#editJenis').val(sepatu.jenis);
                $('#editUkuran').val(sepatu.ukuran);
                $('#editHarga').val(sepatu.harga);
                $('#editStok').val(sepatu.stok);
                $('#editSepatuId').val(sepatu.id);

                // Munculkan modal edit
                $('#editModal').modal('show');
            } else {
                console.error('Failed to retrieve sepatu data.');
```

```

        }
    });
}

// Fungsi untuk menyimpan perubahan yang dilakukan pada sepatu
function saveEditedSepatu() {
    // Mendapatkan data dari formulir edit
    const editedSepatuData = {
        id: $('#editSepatuId').val(),
        nama: $('#editNama').val(),
        merk: $('#editMerk').val(),
        jenis: $('#editJenis').val(),
        ukuran: $('#editUkuran').val(),
        harga: $('#editHarga').val(),
        stok: $('#editStok').val(),
    };

    // Melakukan validasi data jika diperlukan

    // Mengirim data yang diubah ke server (misalnya, menggunakan Ajax)
    // dan mengupdate data di tabel setelah berhasil

    $.ajax({
        url: '/edit-sepatu',
        method: 'POST',
        data: editedSepatuData,
        success: function (response) {
```

```

        if (response.success) {
            // Menutup modal setelah berhasil menyimpan
            $('#editModal').modal('hide');

            // Menampilkan ulang data di tabel setelah perubahan
            displaySepatuInTable(response.updatedData);
        } else {
            console.error('Failed to save edited sepatu.');
```

```

        }
    },
    error: function (error) {
        console.error('Error saving edited sepatu:', error);
    }
});
}

```

*// Fungsi untuk menangani konfirmasi penghapusan sepatu*

```

function confirmDelete(sepatuId) {
    Swal.fire({
        title: 'Apakah Anda yakin?',
        text: 'Sepatu ini akan dihapus!',
        icon: 'warning',
        showCancelButton: true,
        confirmButtonText: 'Ya, hapus!',
        cancelButtonText: 'Batal'
    }).then((result) => {
        if (result.isConfirmed) {
            // Make an AJAX request to the server to delete the item
            $.ajax({
                url: '/delete-sepatu', // Replace with the actual endpoint on
your server

                method: 'POST',
                data: { sepatuId: sepatuId },
                success: function (response) {
                    // Check the response from the server
                    if (response.success) {
                        // Remove the row from the table
                        $('#sepatuTableBody
tr:has(td:contains('${sepatuId}'))').remove();
                        Swal.fire('Berhasil!', 'Sepatu berhasil dihapus.',
'success');
                    } else {
                        Swal.fire('Gagal!', 'Gagal menghapus sepatu.',
'success');
                    }
                },
                error: function () {
                    Swal.fire('Gagal!', 'Terjadi kesalahan.', 'error');
                }
            });
        }
    });
}

```

```

    });
  }
});
}

// Fungsi untuk menangani penghapusan sepatu
function hapusSepatu(sepatuId) {
  // Logika penghapusan sepatu di sini
  console.log(`Hapus sepatu dengan ID ${sepatuId}`);
}

// Fungsi untuk menambahkan sepatu dari formulir
async function tambahSepatu(sepatuData) {
  try {
    const response = await fetch(`${baseUrl}/sepatu`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(sepatuData),
    });

    const data = await response.json();
    console.log('Sepatu berhasil ditambahkan:', data);

    // Panggil fungsi untuk mendapatkan semua sepatu untuk memperbarui
    antarmuka pengguna
    await getAllSepatu();

    // Tampilkan notifikasi sukses
    alert('Sepatu berhasil ditambahkan!');

    // Mengosongkan nilai formulir setelah berhasil menyimpan
    document.getElementById('sepatuForm').reset();
  } catch (error) {
    console.error('Error:', error.message);
  }
}

document.getElementById('sepatuForm').addEventListener('submit', async (event) => {
  event.preventDefault();
  const formData = new FormData(event.target);
  const sepatuData = {};
  formData.forEach((value, key) => {
    sepatuData[key] = value;
  });
});

```

```

    // Panggil fungsi untuk menambahkan sepatu
    await tambahSepatu(sepatuData);

    // Setelah menambahkan, dapatkan kembali semua sepatu untuk memperbarui
    antarmuka pengguna
    await getAllSepatu();
  });

  // Panggil fungsi untuk mendapatkan semua sepatu saat halaman dimuat
  getAllSepatu();

```

Untuk Kode Katalog.js (Menampilkan Data Sepatu)

```

// main.js
const baseUrl = 'http://localhost:3333'; // Sesuaikan dengan port server Anda

// Fungsi untuk mendapatkan semua sepatu
async function getAllSepatu() {
  try {
    const response = await fetch('http://localhost:3333/sepatu');
    const data = await response.json();
    console.log('Data dari server:', data);
    displaySepatu(data);
  } catch (error) {
    console.error('Error:', error.message);
  }
}

// Fungsi untuk menampilkan data sepatu dalam card
function displaySepatu(sepatuData) {
  const sepatuContainer = $('#sepatuContainer');
  sepatuContainer.empty();

  console.log(typeof sepatuData);

  if (Array.isArray(sepatuData) || (sepatuData &&
Array.isArray(sepatuData.data))) {
    const sepatuList = Array.isArray(sepatuData) ? sepatuData :
sepatuData.data;

    sepatuList.forEach((sepatu, index) => {
      const card = `
        <div class="card shadow-sm col-md-3 mb-4 mx-md-2">
          <div class="card-body">
            <h5 class="card-title text-
primary">${sepatu.nama}</h5>

```

```

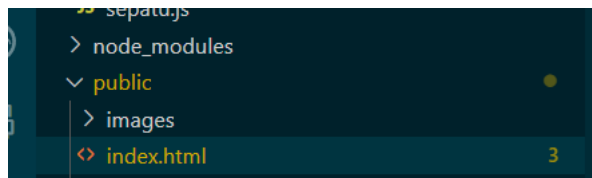
        <p class="card-text">Merk: ${sepatu.merk}</p>
        <p class="card-text">Jenis: ${sepatu.jenis}</p>
        <p class="card-text">Ukuran: ${sepatu.ukuran}</p>
        <p class="card-text">Harga: Rp${sepatu.harga}.000</p>
        <p class="card-text">Stok: ${sepatu.stok}</p>
    </div>
</div>
`;
sepatuContainer.append(card);

    if ((index + 1) % 3 === 0) {
        sepatuContainer.append('<div class="w-100 d-md-none d-lg-none d-xl-none"></div>');
    }
});
} else {
    console.error('Sepatu data is not an array.');
```

*// Panggil fungsi untuk mendapatkan semua sepatu saat halaman dimuat*  
getAllSepatu();

## 2. Langkah-langkah membuat project Frontend Sepatus

Untuk frontend, Kami menggunakan Bootstrap, adapun file yang dibuat yaitu File index.html



Adapun Kode nya

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link href="images/sepatuicon.png" rel="icon">
  <link href="images/sepatuicon.png" rel="apple-touch-icon">
  <title>Sepatu Kelompok 5</title>
  <!-- <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"

```

```

        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTWFspD3yD65VohhpUuCOMLASjC"
crossorigin="anonymous"> -->

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

    <!-- SweetAlert2 CSS -->
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/sweetalert2@10">

</head>

<body>
    <!-- Navigation Bar -->
    <nav class="navbar sticky-top navbar-expand-lg navbar-light bg-light ">
        <a class="navbar-brand ml-6 h1" href="http://localhost:3000/home">
            <h1></h1>
            <!-- Adjust the
path and dimensions as needed -->
        </a>
        <div class="navbar-collapse">
            <ul class="nav nav-pills">
                <li class="nav-item mr-2">
                    <a class="nav-link active" aria-current="page"
href="index.html">Master</a>
                </li>
                <li class="nav-item mr-2">
                    <a class="nav-link" href="katalog.html">Katalog</a>
                </li>
                <li class="nav-item dropdown mr-2">
                    <!-- <a class="nav-link dropdown-toggle" href="#"
id="navbarDropdown" role="button"
                        data-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
                        Profile
                    </a>
                    <div class="dropdown-menu mt-2 shadow" aria-
Labelledby="navbarDropdown">
                        <a class="dropdown-item"
href="http://localhost:3000/">Login</a>
                        <a class="dropdown-item"
href="http://localhost:3000/Register">Register</a>
                    </div> -->
                </li>
            </ul>

```

```

    </div>
</nav>

<div class="container mt-4">
    <h3 class="text-primary mb-4">Tambah Sepatu</h3>
    <form id="sepatuForm" enctype="multipart/form-data" action="/tambah-sepatu" method="POST">
        <div class="row">
            <div class="col-md-6 mb-3">
                <label for="nama" class="form-label">Nama:</label>
                <input type="text" class="form-control" id="nama"
name="nama" required autocomplete="off">
            </div>
            <div class="col-md-6 mb-3">
                <label for="merk" class="form-label">Merk:</label>
                <select class="form-control" id="merk" name="merk"
required autocomplete="off">
                    <option value="Nike">Nike</option>
                    <option value="Adidas">Adidas</option>
                    <option value="Converse">Converse</option>
                    <!-- Add more options as needed -->
                </select>
            </div>
        </div>
        <div class="row">
            <div class="col-md-6 mb-3">
                <label for="jenis" class="form-label">Jenis:</label>
                <select class="form-control" id="jenis" name="jenis"
required autocomplete="off">
                    <option value="Olahraga">Olahraga</option>
                    <option value="Kasual">Kasual</option>
                    <option value="Formal">Formal</option>
                    <option value="Lainnya">Lainnya</option>
                </select>
            </div>
            <div class="col-md-6 mb-3">
                <label for="ukuran" class="form-label">Ukuran:</label>
                <input type="text" class="form-control" id="ukuran"
name="ukuran" required autocomplete="off">
            </div>
        </div>
        <div class="row">
            <div class="col-md-6 mb-3">
                <label for="harga" class="form-label">Harga:</label>
                <input type="number" class="form-control" id="harga"
name="harga" required autocomplete="off">
            </div>
            <div class="col-md-6 mb-3">

```

```

        <label for="stok" class="form-label">Stok:</label>
        <input type="number" class="form-control" id="stok"
name="stok" required autocomplete="off">
    </div>
</div>
    <button type="submit" class="btn btn-primary">Tambah</button>
</form>
</div>

<div class="container mt-4">
    <h3 class="text-primary mb-4">Data Sepatu</h3>

    <table class="table table-bordered table-hover shadow-sm">
        <thead class="thead-light">
            <tr>
                <th>ID</th>
                <th>Nama</th>
                <th>Merk</th>
                <th>Jenis</th>
                <th>Ukuran</th>
                <th>Harga</th>
                <th>Stok</th>
                <th>Aksi</th>
            </tr>
        </thead>
        <tbody id="sepatuTableBody">
            <!-- Add your table rows dynamically here -->
        </tbody>
    </table>
</div>

<!-- Edit Modal -->
<div class="modal" id="editModal">
    <div class="modal-dialog">
        <div class="modal-content">

            <!-- Modal Header -->
            <div class="modal-header">
                <h4 class="modal-title">Edit Sepatu</h4>
                <button type="button" class="close" data-
dismiss="modal">&times;</button>
            </div>

            <!-- Modal Body -->
            <div class="modal-body">
                <form id="editSepatuForm" enctype="multipart/form-data"
action="/edit-sepatu" method="POST">

```



```

        <div class="mb-3">
            <label for="editNama" class="form-
label">Nama:</label>
            <input type="text" class="form-control"
id="editNama" name="editNama" required
                autocomplete="off">
        </div>
        <div class="mb-3">
            <label for="editMerk" class="form-
label">Merk:</label>
            <input type="text" class="form-control"
id="editMerk" name="editMerk" required
                autocomplete="off">
        </div>
        <div class="mb-3">
            <label for="editJenis" class="form-
label">Jenis:</label>
            <select class="form-control" id="editJenis"
name="editJenis" required autocomplete="off">
                <option value="olahraga">Olahraga</option>
                <option value="kasual">Kasual</option>
                <option value="formal">Formal</option>
                <option value="lainnya">Lainnya</option>
            </select>
        </div>
        <div class="mb-3">
            <label for="editUkuran" class="form-
label">Ukuran:</label>
            <input type="text" class="form-control"
id="editUkuran" name="editUkuran" required
                autocomplete="off">
        </div>
        <div class="mb-3">
            <label for="editHarga" class="form-
label">Harga:</label>
            <input type="number" class="form-control"
id="editHarga" name="editHarga" required
                autocomplete="off">
        </div>
        <div class="mb-3">
            <label for="editStok" class="form-
label">Stok:</label>
            <input type="number" class="form-control"
id="editStok" name="editStok" required
                autocomplete="off">
        </div>
        <!-- Add hidden input for sepatu ID -->

```

```

        <input type="hidden" id="editSepatuId"
name="editSepatuId">
    </form>
</div>

<!-- Modal Footer -->
<div class="modal-footer">
    <button type="button" class="btn btn-secondary" data-
dismiss="modal">Batal</button>
    <button type="button" class="btn btn-primary"
onclick="saveEditedSepatu()">Simpan</button>
</div>

</div>
</div>
</div>

<script src="main.js"></script>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js"
"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"><
/script>
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@10"></script>

</body>

</html>

```

Selanjutnya Yaitu membuat File katalog.html. Yang berisi Data sepatu yang sudah di input, untuk kode File katalog :

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="images/sepatuicon.png" rel="icon">
    <link href="images/sepatuicon.png" rel="apple-touch-icon">
    <title>Sepatu Kelompok 5</title>
    <!-- <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"

```

```

        integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous"> -->

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">

</head>

<body>
    <!-- Navigation Bar -->
    <nav class="navbar sticky-top navbar-expand-lg navbar-light bg-light ">
        <a class="navbar-brand ml-6 h1" href="http://localhost:3000/home">
            <h1></h1>
            <!-- Adjust the path and dimensions as needed -->
            </a>
            <div class="navbar-collapse">
                <ul class="nav nav-pills">
                    <li class="nav-item mr-2">
                        <a class="nav-link" aria-current="page"
href="index.html">Master</a>
                    </li>
                    <li class="nav-item mr-2">
                        <a class="nav-link active" href="katalog.html">Katalog</a>
                    </li>

                </ul>
            </div>
        </nav>

        <div class="container mt-4 mx-auto">
            <h3 class="text-primary mb-4">Katalog Sepatu</h3>

            <div id="sepatuContainer" class="row"></div>
        </div>

        <script src="katalog.js"></script>

        <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
        <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.min.js
"></script>
        <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"><
/script>

```

```
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@10"></script>

</body>

</html>
```

### 3. Langkah-langkah membuat project Backend Login-Register **Client**

Membuat Package, yang digunakan :

a. Package-lock.json

- Axios
- React, React-dom, React-router-dom, React-scripts
- Tailwind-react-datepicker
- Web-vitals

```
client > {} package-lock.json > {} packages > {} "" > {} dependencies
1  ∨ {
2    "name": "testing-aja",
3    "version": "0.1.0",
4    "lockfileVersion": 2,
5    "requires": true,
6    "packages": {
7      "": {
8        "name": "testing-aja",
9        "version": "0.1.0",
10       "dependencies": {
11         "@material-tailwind/react": "^1.2.4",
12         "@testing-library/jest-dom": "^5.16.5",
13         "@testing-library/react": "^13.4.0",
14         "@testing-library/user-event": "^13.5.0",
15         "axios": "^1.2.1",
16         "react": "^18.2.0",
17         "react-dom": "^18.2.0",
18         "react-router-dom": "^6.4.4",
19         "react-scripts": "5.0.1",
20         "tailwind-react-datepicker": "^1.1.8",
21         "web-vitals": "^2.1.4"
22       },
23       "devDependencies": {
24         "autoprefixer": "^10.4.13",
25         "postcss": "^8.4.19",
26         "tailwindcss": "^3.2.4"
27       }
28     }
29   }
```

b. Package.json

- Axios
- React, React-dom, React-router-dom, React-scripts
- Tailwind-react-datepicker
- Web-vitals

```

client > {} package.json > {} scripts > start
1  {
2    "name": "testing-aja",
3    "version": "0.1.0",
4    "private": true,
5    "dependencies": {
6      "@material-tailwind/react": "^1.2.4",
7      "@testing-library/jest-dom": "^5.16.5",
8      "@testing-library/react": "^13.4.0",
9      "@testing-library/user-event": "^13.5.0",
10     "axios": "^1.2.1",
11     "react": "^18.2.0",
12     "react-dom": "^18.2.0",
13     "react-router-dom": "^6.4.4",
14     "react-scripts": "5.0.1",
15     "tailwind-react-datepicker": "^1.1.8",
16     "web-vitals": "^2.1.4"
17   },

```

c. App.js  
Untuk

```

client > src > JS App.js > App
1  import './App.css';
2  import Register from './pages/Register';
3  import Login from './src/pages/Login';
4  import {BrowserRouter, Routes, Route} from 'react-router-dom';
5  import { Home } from './pages/Home';
6  import { User } from './pages/User';
7  import { Dashboard } from './pages/Dashboard'; Remove this commented out code.
8  import Update from './pages/Update';
9
10 const App = () => {
11   return (
12     <div className='App'>
13       <BrowserRouter>
14         <Routes>
15           <Route path="/" element={<Login />}/>
16           <Route path="/register" element={<Register />}/>
17           <Route path="/home" element={<Home />}/>
18           {/* <Route path="/dashboard" element={<Dashboard />}/> */}
19           <Route path="/user" element={<User />}/>
20           <Route path="/update/:id" element={<Update />}/>
21         </Routes>
22       </BrowserRouter>
23     </div>
24   );
25 }
26 export default App;

```

d. App.css

```

client > src > # App.css > ...
1  .App {
2    font-family: 'Poppins';
3  }
4

```

e. Index.js

```
client > src > JS index.js > ...
1  import React from 'react';
2  import ReactDOM from 'react-dom/client';
3  import './index.css';
4  import App from './App';
5
6  const root = ReactDOM.createRoot(document.getElementById('root'));
7  root.render(
8    <React.StrictMode>
9    |   <App />
10   </React.StrictMode>
11 );
```

f. Index.css

```
client > src > # index.css
1  @tailwind base;      Unknown at rule @tailwind
2  @tailwind components; Unknown at rule @tailwind
3  @tailwind utilities;  Unknown at rule @tailwind
4  @import url('https://fonts.googleapis.com/css2?family=Poppins&display=swap');
5
```

g. Postcss.config.js

```
client > JS postcss.config.js > ...
1  module.exports = {
2    plugins: {
3      tailwindcss: {},
4      autoprefixer: {},
5    },
6  }
7
```

h. Tailwind.config.js

```
client > JS tailwind.config.js > [?] <unknown>
1  /** @type {import('tailwindcss').Config} */
2  module.exports = {
3    content: [
4      './src/**/*.js', './src/**/*.jsx', './src/**/*.ts', './src/**/*.tsx',
5    ],
6    theme: {
7      extend: {},
8    },
9    plugins: [],
10 }
```

## Server

Membuat Package, yang digunakan :

- a. Package-lock.json menggunakan kode

```
$ npx express-generator—no-view
```

```
$ npm install
```

```
$ npm install mysql2—save
```

```
$ npm install -g nodemon
```

```
$ npm install body-parser
```

```
$ npm install cors
```

```
server > {} package-lock.json > {} packages > {} "" > {} dependencies > mysql
1  {
2    "name": "server",
3    "version": "1.0.0",
4    "lockfileVersion": 2,
5    "requires": true,
6    "packages": {
7      "": {
8        "name": "server",
9        "version": "1.0.0",
10       "license": "ISC",
11       "dependencies": {
12         "body": "^5.1.0",
13         "body-parser": "^1.20.1",
14         "cors": "^2.8.5",
15         "express": "^4.18.2",
16         "mysql": "^2.18.1",
17         "nodemon": "^2.0.20",
18         "parsen": "^0.1.1"
19       }
20     }
21   }
```

- b. Package.json

Kode yang ditambahkan yang didalam debug yaitu “start” dan “test”

```
Scripts": {
```

```
  "start": "nodemon index.js",
```

```
  "test": "echo \"Error: no test specified\" && exit 1".
```

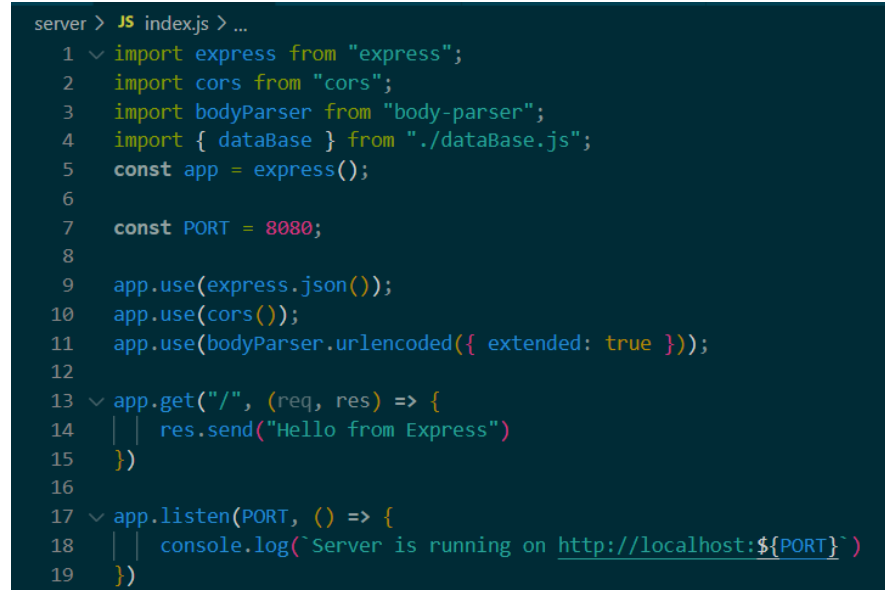
```
server > {} package.json > ...
1  {
2    "name": "server",
3    "version": "1.0.0",
4    "description": "",
5    "main": "index.js",
6    "type": "module",
7    "scripts": {
8      "start": "nodemon index.js",
9      "test": "echo \"Error: no test specified\" && exit 1"
10   },
11 }
```

c. Index.js

Menggunakan kode :

```
import express from "express";
import cors from "cors";
import bodyParser from "body-parser";
import { dataBase } from "../dataBase.js";
const app = express();
```

```
const PORT = 8080;
```



```
server > JS index.js > ...
1  import express from "express";
2  import cors from "cors";
3  import bodyParser from "body-parser";
4  import { dataBase } from "../dataBase.js";
5  const app = express();
6
7  const PORT = 8080;
8
9  app.use(express.json());
10 app.use(cors());
11 app.use(bodyParser.urlencoded({ extended: true }));
12
13 app.get("/", (req, res) => {
14   res.send("Hello from Express")
15 })
16
17 app.listen(PORT, () => {
18   console.log(`Server is running on http://localhost:${PORT}`)
19 })
```

d. database.js

Untuk menghubungkan database dengan project yang dibuat :

```
export const dataBase = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'pemograman_web'
})
```



```

server > JS dataBase.js > 📦 dataBase.connect() callback
1  import mysql from 'mysql';
2
3  export const dataBase = mysql.createConnection({
4      host: 'localhost',
5      user: 'root',
6      password: '',
7      database: 'pemograman_web'
8  })
9
10 dataBase.connect((err) => {
11     if(err) {
12         console.log(err)
13     } else {
14         console.log("connected")
15     }
16 })

```

## FrontEnd Login-Register

### - Home.jsx

```

client > src > pages > 📦 Home.jsx > ...
1  import React, {useEffect, useState} from 'react'
2  import { Link, useNavigate } from "react-router-dom";
3
4  export const Home = () => {
5      const handleLogout = () => {
6          localStorage.clear();
7          navigate("/");
8      };
9      const navigate = useNavigate();
10
11     const [username, setUsername] = useState("");
12
13     useEffect(() => {
14         // Get item from LocalStorage
15         const userInfoString = localStorage.getItem('user-info');
16
17         // Check if the item exists in LocalStorage
18         if (userInfoString) {
19             // Parse the JSON string to an object
20             const userInfo = JSON.parse(userInfoString);
21             // setUsername(userInfo.data)
22
23             // Now, you can use the userInfo object as needed
24             console.log(userInfo.data.username);
25             setUsername(userInfo.data.username);
26         }

```

```

27 |     }, []);
28 |
29 |
30 | return (
31 | <>
32 |   <nav className="fixed w-full h-1/7 z-30 top-0 ■ text-white ■ bg-white shadow" >
33 |     <div className="w-full container mx-auto flex flex-wrap items-center justify-between py-5 max-w-7xl">
34 |       <div className="p1-4 flex items-center">
35 |         <svg className="mr-1"
36 |           viewBox="-30 1 511 511.999"
37 |           xmlns="http://www.w3.org/2000/svg">
38 |           <path d="m219.992188 96.859375c-74.847657 1.046875-135.492188 62.886719-135.15625 137.742187.097656 22.058594 5.4101
39 |             fill="#ffb0"/>
40 |           <path d="m109.144531 294.160156c-14.667969 0-26.722656 11.875-26.722656 26.726563 0 25.898437 33.335937 36.613281 48
41 |             fill="#00b8ff"/>
42 |           <path d="m217.523438 204.078125c0 14.757813-11.964844 26.722656-26.722657 26.722656-14.757813 0-26.722656-11.964843-
43 |             fill="#f33954"/>
44 |           <path d="m311.175781 267.4375c0 14.757812-11.964843 26.722656-26.726562 26.722656-14.757813 0-26.722657-11.964844-26
45 |             fill="#3df399"/>
46 |           <path d="m180.988281 487.355469c0 8.089843 6.558594 14.644531 14.644531 14.644531h52.652344c8.085938 0 14.644532-6.5
47 |             fill="#00b8ff"/>
48 |           <path d="m449.699219 140.792969c-1.710938-2.199219-4.257813-3.585938-7.03125-3.828125l-39.492188-3.453125c-5.492187-
49 |             />svg>
50 |       <div className="font-bold text-2xl lg:text-2xl □ text-gray-800">SEPATU</div>
51 |     </div>
52 |     <ul className="lg:flex justify-end flex-1 items-center □ text-gray-800">
53 |       <Link to={"/home"} className="mr-7">Home</Link>
54 |       { /* <Link to="http://localhost:3333/index.html" className="mr-7">Dashboard</Link> */ }
55 |       <a href="http://localhost:3333/index.html" className="mr-7">Dashboard</a>
56 |       <Link to={"/user"} className="mr-7">User</Link>
57 |
58 |     </ul>
59 |     <div className="flex items-end justify-center flex-col gap-10">
60 |       <button onClick={handleLogout} className="flex items-center py-2 px-4 text-sm uppercase rounded ■ bg-white
61 |         | Logout
62 |       </button>
63 |     </div>
64 |   </div>
65 | </nav>
66 | <div className="flex flex-wrap md items-center h-screen mt-12">
67 |   <div className="w-full md:w-1/2 h-screen">
68 |     <div className="mx-32">
69 |       <p className="text-5xl font-bold mt-24">Welcome {username} 🍕 </p>
70 |
71 |
72 |
73 |
74 |
75 |     </div>
76 |   </div>
77 |
78 | </div>
79 |
80 | <footer className="■ bg-gray-50 shadow mt-10 fixed w-full ">
81 |   <div className="px-4 py-12 mx-auto space-y-8 overflow-hidden">
82 |     <nav className="flex flex-wrap justify-center -mx-5 -my-2">
83 |       <div class="px-5 py-2">
84 |         <Link className="text-base leading-6 ■ text-gray-500 □ hover:text-gray-900">
85 |           | About
86 |         </Link>
87 |       </div>
88 |       <div class="px-5 py-2">
89 |         <Link className="text-base leading-6 ■ text-gray-500 □ hover:text-gray-900">
90 |           | Blog
91 |         </Link>
92 |       </div>
93 |       <div class="px-5 py-2">
94 |         <Link className="text-base leading-6 ■ text-gray-500 □ hover:text-gray-900">
95 |           | Team
96 |         </Link>
97 |       </div>
98 |       <div class="px-5 py-2">
99 |         <Link className="text-base leading-6 ■ text-gray-500 □ hover:text-gray-900">
100 |           | Pricing
101 |         </Link>
102 |       </div>

```

```

103 |         <div class="px-5 py-2">
104 |             <Link className="text-base leading-6 ■ text-gray-500 ■ hover:text-gray-900">
105 |                 Contact
106 |             </Link>
107 |         </div>
108 |         <div class="px-5 py-2">
109 |             <Link className="text-base leading-6 ■ text-gray-500 ■ hover:text-gray-900">
110 |                 Terms
111 |             </Link>
112 |         </div>
113 |     </nav>
114 |     <div className="flex justify-center mt-8 space-x-6">
115 |         <div className="■ text-gray-400 ■ hover:text-gray-500">
116 |             <span className="sr-only">Facebook</span>
117 |             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
118 |                 <path fill-rule="evenodd" d="M22 12c0-5.523-4.477-10-10-10S2 6.477 2 12c0 4.991 3.657 9.128 8.438 11.688" data-bbox="117 217 131 253"/>
119 |             </svg>
120 |         </div>
121 |         <div className="■ text-gray-400 ■ hover:text-gray-500">
122 |             <span className="sr-only">Instagram</span>
123 |             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
124 |                 <path fill-rule="evenodd" d="M12.315 2c2.43 0 2.784.013 3.808.06 1.064.049 1.791.218 2.427.465" data-bbox="123 283 137 319"/>
125 |             </svg>
126 |         </div>
127 |         <div className="■ text-gray-400 ■ hover:text-gray-500">
128 |             <span className="sr-only">Twitter</span>
129 |             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
130 |                 <path d="M8.29 20.251c7.547 0 11.675-6.253 11.675-11.675 0-.178 0-.355-.012-.53A8.348 8.348 0 0 0 8.29 20.251" data-bbox="129 349 143 385"/>
131 |             </svg>
132 |         </div>
133 |         <div className="■ text-gray-400 ■ hover:text-gray-500">
134 |             <span className="sr-only">GitHub</span>
135 |             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
136 |                 <path fill-rule="evenodd" d="M12 2C6.477 2 2 6.477 2 12c0 4.991 3.657 9.128 8.438 11.688" data-bbox="135 415 149 451"/>
137 |             </svg>
138 |         </div>
139 |         <div className="■ text-gray-400 ■ hover:text-gray-500">
140 |             <span className="sr-only">Dribbble</span>
141 |             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
142 |                 <path fill-rule="evenodd" d="M12 2C6.477 2 2 6.477 2 12c0 4.991 3.657 9.128 8.438 11.688" data-bbox="141 481 155 517"/>
143 |             </svg>
144 |         </div>
145 |     </div>
146 |
147 | </div>
148 | </footer>
149 | </>
150 |
151 | )
152 | }

```

## - Login.jsx

```
client > src > pages > Login.jsx > Login > handleChange > setInputs() callback
1  import React, {useState, useEffect} from 'react'
2  import { Link, useNavigate } from 'react-router-dom';
3  import axios from 'axios';
4
5  const Login = () => {
6    const [inputs, setInputs] = useState({
7      username: "",
8      password: "",
9    });
10
11    const navigate = useNavigate();
12
13    useEffect(() => {
14      if (localStorage.getItem("user-info")) {
15        | navigate("/home");
16      }
17    })
18
19    const handleChange = (e) => {
20      | setInputs((prev) => ({
21        | ...prev,
22        | [e.target.name]: e.target.value,
23      }));
24    };
25
26    const handleSubmit = async () => {
27      | const res = await axios.post("http://localhost:8080/api/login", inputs);
```

```
28      | localStorage.setItem("user-info", JSON.stringify(res));
29      | navigate("/home");
30    }
31
32    return (
33      <body className="bg-white">
34        <div className="relative flex flex-col justify-center min-h-screen overflow-hidden">
35          <div className="w-full p-6 m-auto rounded-md shadow-md lg:max-w-xl">
36            <div className="pl-4 flex items-center mb-10" invalid:border-indigo-900 flex items-center justify-center
37
38              <div className="font-bold text-6xl lg:text-3xl text-gray-800">WELCOME TO TOKO <span className="text-6xl lg:text-3xl text-gray-800">
39            </div>
40            <form className="mt-6">
41              <div className="mb-2">
42                <label
43                  htmlFor="username"
44                  className="block text-sm font-semibold text-gray-800"
45                >
46              </label>
47              <input
48                type="username"
49                placeholder="Username"
50                className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
51                name="username"
52                onChange={handleChange}
53              />
```

```

80         <div className="flex items-center justify-center space-x-4 mt-3">
81
82
83         </div>
84     </div>
85 </form>
86
87     <p className="mt-8 text-xs font-light text-center text-gray-700">
88         Don't have an account?
89         <Link to={"/Register"}>
90             <div className="font-medium text-indigo-500 hover:underline">Register</div>
91         </Link>
92     </p>
93
94 </div>
95 </div>
96 </body>
97
98
99
100 )
101 }
102 export default Login;

```

```

54 </div>
55 <div className="mb-2">
56     <label
57         htmlFor="password"
58         className="block text-sm font-semibold text-gray-800"
59     >
60     </label>
61     <input
62
63         type="password"
64         placeholder='Password'
65         className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
66         name='password'
67         onChange={handleChange}
68     />
69 </div>
70 <div className="mt-6">
71     <button onClick={handleSubmit} className="w-full px-4 py-2 tracking-wide text-white uppercase"
72         Login
73     </button>
74 </div>
75
76 <div>
77
78 </div>
79 </div>

```

- Register.jsx

client > src > pages > Register.jsx > Register

```
1  import React, {useState} from 'react'
2  import { Link, useNavigate } from 'react-router-dom';
3  import axios from 'axios';
4
5  export const Register = () => {
6    const [inputs, setInputs] = useState({
7      username: "",
8      email: "",
9      tanggal_lahir: "",
10     password: "",
11     confirmPassword: "",
12   });
13   const [error, setError] = useState('');
14   const navigate = useNavigate();
15
16   const handleChange = (e) => {
17     setInputs((prevInputs) => ({
18       ...prevInputs,
19       [e.target.name]: e.target.value,
20     }));
21   };
22
23   const handleSubmit = (e) => {
24     e.preventDefault();
25     if (inputs.password !== inputs.confirmPassword) {
26       setError('The password and confirm password fields do not match');
27       return;
```

```
28   }
29   setError('');
30   axios.post("http://localhost:8080/api/register", inputs);
31   setTimeout(() => {
32     navigate("/");
33   }, 1000);
34 };
35
36 return (
37   <div className="relative flex flex-col justify-center min-h-screen overflow-hidden bg-white">
38     <div className="w-full p-6 m-auto rounded-md shadow-md lg:max-w-xl">
39       <div className="p-4 flex items-center mb-10 invalid:border-indigo-900 flex items-center justify-center"
40         >
41         <div className="font-bold text-5xl lg:text-4x2 text-black">Baron <span className="text-indigo-900"
42           >
43         </div>
44       </div>
45       <form className="mt-6">
46         <div className="mb-2">
47           <label
48             for="username"
49             className="block text-sm font-semibold text-gray-800"
50           >
51           </label>
52           <input
53             type="text"
54             placeholder='Username'
55             className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
```

```

54         name="username"
55         onChange={handleChange}
56     />
57 </div>
58 <div className="mb-2">
59     <label
60         for="email"
61         className="block text-sm font-semibold text-gray-800"
62     >
63     </label>
64     <input
65         type="email"
66         placeholder="Email Address"
67         className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
68         name="email"
69         onChange={handleChange}
70     />
71 </div>
72 <div className="mb-2">
73     <label
74         for="date"
75         className="block text-sm font-semibold text-gray-800"
76     >
77     </label>
78     <input
79         type="date"

```

```

80         className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
81         name="tanggal_lahir"
82         onChange={handleChange}
83     />
84 </div>
85 <div className="mb-2">
86     <label
87         for="password"
88         className="block text-sm font-semibold text-gray-800"
89     >
90     </label>
91     <input
92         type="password"
93         placeholder="Password"
94         className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
95         name="password"
96         onChange={handleChange}
97     />
98 </div>
99 <div className="mb-2">
100     <label
101         for="password"
102         className="block text-sm font-semibold text-gray-800"
103     >
104     </label>
105     <input

```

```

106         type="password"
107         placeholder='Confirm Password'
108         name='confirmPassword'
109         className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
110         onChange={handleChange}
111     />
112     {error && <div className="text-red-500 text-xs">{error}</div>}
113
114 </div>
115 <div className="mt-6">
116     <button type="submit" onClick={handleSubmit} className="w-full px-4 py-2 tracking-wide text-white"
117     Register
118     </button>
119 </div>
120 </form>
121
122 </div>
123
124 <p className="mt-8 text-xs font-light text-center text-gray-700">
125     { " " }
126     Already have an account?{ " " }
127     <Link to={"/"}>
128     <p className="font-medium text-indigo-500 hover:underline">
129     | Login
130     </p>
131     </Link>
132
133     </p>
134 </div>
135
136 )
137 }
138 export default Register;

```

## - Update.jsx

```

client > src > pages > Update.jsx > Update
1  import axios from 'axios';
2  import React, {useState, useEffect} from 'react';
3  import { Link, useNavigate, useParams } from 'react-router-dom';
4
5  const initialState = {
6      username: "",
7      email: "",
8      tanggal_lahir: "",
9      password: "",
10  };
11
12  const Update = () => {
13      const [state, setState] = useState(initialState);
14      const { username, email, tanggal_lahir, password } = state;
15      const { id } = useParams();
16
17      const navigate = useNavigate();
18
19      useEffect(() => {
20          axios.get(`http://localhost:8080/api/get/${id}`).then((response) => {
21              setState({ ...response.data[0] });
22          });
23      }, []);
24
25      const handleChange = (e) => {
26          setState((prev) => ({
27              ...prev,

```



```

28         [e.target.name]: e.target.value,
29     });
30 };
31
32 const handleSubmit = (e) => {
33     e.preventDefault();
34     axios.put(`http://localhost:8080/api/update/${id}`, {
35         username,
36         email,
37         tanggal_lahir,
38         password,
39     });
40     setTimeout(() => {
41         alert("Data berhasil diupdate")
42         navigate("/user");
43     }, 1000)
44 };
45
46 return (
47     <div className="relative flex flex-col justify-center min-h-screen overflow-hidden">
48         <div className="w-full p-6 m-auto rounded-md shadow-md lg:max-w-xl">
49             <h1 className="text-3xl font-semibold text-center text-black">
50                 Update
51             </h1>
52             <form className="mt-6">
53                 <div className="mb-2">
54                     <label

```

```

55                     for="username"
56                     className="block text-sm font-semibold text-gray-800"
57                 >
58                 </label>
59                 <input
60                     required
61                     type="text"
62                     autoComplete="off"
63                     className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
64                     name="username"
65                     onChange={handleChange}
66                     value={username || ""}
67                 />
68             </div>
69             <div className="mb-2">
70                 <label
71                     for="email"
72                     className="block text-sm font-semibold text-gray-800"
73                 >
74                 </label>
75                 <input
76                     required
77                     type="email"
78                     className="block w-full px-4 py-2 mt-2 bg-white border rounded-md"
79                     name="email"
80                     onChange={handleChange}
81                     value={email || ""}

```



## - User.jsx

```
client > src > pages > User.jsx > [0] User
1  import React, { useEffect, useState } from 'react'
2  import { Link, useNavigate } from "react-router-dom";
3  import axios from "axios";
4
5
6  export const User = () => {
7    const [data, setData] = useState([]);
8    const loadData = async () => {
9      const response = await axios.get("http://localhost:8080/api/users");
10     setData(response.data);
11   }
12
13   const navigate = useNavigate();
14
15   // Fungsi untuk mengecek apakah ada user-info di local storage, jika tidak ada maka akan di redirect ke halaman login
16   const clearLocalStorage = () => {
17     if (!localStorage.getItem("user-info")) {
18       navigate("/");
19     }
20   };
21
22   // Jalankan fungsi clearLocalStorage ketika component di tampilkan pertama kali
23   useEffect(() => {
24     clearLocalStorage();
25     loadData();
26   }, []);
27
```

```
28   // Fungsi untuk menghapus user-info di local storage dan redirect ke halaman login (logout)
29   const handleLogout = () => {
30     localStorage.clear();
31     navigate("/");
32   };
33
34   const handleDelete = (id) => {
35     if (window.confirm("Are you sure want to delete this user?")){
36       axios.delete(`http://localhost:8080/api/delete/${id}`);
37       setTimeout(() => navigate(), 100);
38       alert("Succes delete")
39     }else{
40       setTimeout(() => navigate(), 100);
41       alert("Data kamu belum terhapus")
42     }
43   }
44
45   return (
46     <>
47     <nav className="fixed w-full h-1/7 z-30 top-0 bg-white shadow" >
48     <div className="w-full container mx-auto flex flex-wrap items-center justify-between py-5 max-w-7xl">
49       <div className="pl-4 flex items-center">
50
51       </div>
52       <ul className="lg:flex justify-end flex-1 items-center text-gray-800">
53       <Link to={"/home"} className="mr-7">Home</Link>
```

```

54      { /* <Link to="http://localhost:3333/index.html" className="mr-7">Dashboard</Link> */}
55      <a href="http://localhost:3333/index.html" className="mr-7">Dashboard</a>
56      <Link to={"/user"} className="mr-7">User</Link>
57
58    </ul>
59    <div className="flex items-end justify-center flex-col gap-10">
60      <button onClick={handleLogout} className="flex items-center py-2 px-4 text-sm uppercase rounded bg-white hover:bg-gray-100">
61        Logout
62      </button>
63    </div>
64  </div>
65  </nav>
66  <div className="mt-40">
67
68    <div className="flex justify-center items-center mt-10">
69      <div className="-translate-y-6">
70        <h1 className="mb-4 text-center">Data Table Users</h1>
71        <table>
72          <thead className="text-xs font-semibold uppercase text-gray-400 bg-gray-100">
73            <tr>
74              <th className="p-2 whitespace-nowrap">
75                <div class="font-semibold text-center">No</div> Unknown property 'class' found, use 'className' instead
76              </th>
77              <th className="p-2 whitespace-nowrap">
78                <div class="font-semibold text-center">Username</div> Unknown property 'class' found, use 'className' instead
79              </th>

```

```

80              <th className="p-2 whitespace-nowrap">
81                <div class="font-semibold text-center">Email</div> Unknown property 'class' found, use 'className' instead
82              </th>
83              <th className="p-2 whitespace-nowrap">
84                <div class="font-semibold text-center">Date of Birth</div> Unknown property 'class' found, use 'className' instead
85              </th>
86              <th className="p-2 whitespace-nowrap">
87                <div class="font-semibold text-center">Password</div> Unknown property 'class' found, use 'className' instead
88              </th>
89              <th className="text-center px-10 py-2">
90                <div class="font-semibold text-center">Action</div> Unknown property 'class' found, use 'className' instead
91              </th>
92            </tr>
93          </thead>
94          <tbody className="text-sm divide-y divide-gray-100">
95            {data.map((item, index) => {
96              return (
97                <tr key={item.id}>
98                  <td className="p-2 whitespace-nowrap" scope="row">
99                    {index + 1}
100                  </td>
101                  <td className="p-2 whitespace-nowrap">
102                    {item.username}
103                  </td>
104                  <td className="p-2 whitespace-nowrap">
105                    <div class="text-left font-medium text-gray-400"> Unknown property 'class' found, use 'className' instead

```

```

106                    {item.email}
107                  </div>
108                </td>
109                <td className="p-2 whitespace-nowrap">
110                  {item.tanggal_lahir}
111                </td>
112                <td class="p-2 whitespace-nowrap text-green-500"> Unknown property 'class' found, use 'className' instead
113                  {item.password}
114                </td>
115                <td>
116                  <div class="px-6 py-4 whitespace-no-wrap text-sm leading-5"> Unknown property 'class' found, use 'className' instead
117                    <div class="flex space-x-4"> Unknown property 'class' found, use 'className' instead
118                      <Link to={`/update/${item.id}`} className="text-blue-500 hover:text-blue-600">
119                        <svg xmlns="http://www.w3.org/2000/svg"
120                          className="w-5 h-5 mr-1"
121                          fill="none" viewBox="0 0 24 24"
122                          stroke="currentColor">
123                          <path stroke-linecap="round" stroke-linejoin="round" stroke-width="2"
124                            d="M11 5H6a2 2 0 0-2 2v11a2 2 0 02 2h11a2 2 0 02 2v-5m-1.414-9.414a2 2 0 112.828 2.828L11 8.828"
125                        </path>
126                      </svg>
127                      <p>Edit</p>
128                    </div>
129                    <Link to={`/delete/${item.id}`}
130                      onClick={() => handleDelete(item.id)} className="text-red-500 hover:text-red-600">
131                      <svg xmlns="http://www.w3.org/2000/svg"

```

```

133         className="w-5 h-5 mr-1 ml-3"
134         fill="none" viewBox="0 0 24 24"
135         stroke="currentColor">
136         <path stroke-linecap="round" Unknown property 'stroke-linecap' found, use 'strokeLinecap' inst
137             stroke-linejoin="round" Unknown property 'stroke-linejoin' found, use 'strokeLinejoin' inste
138             stroke-width="2" Unknown property 'stroke-width' found, use 'strokeWidth' instead
139             d="M19 7l-.867 12.142A2 2 0 0116.138 21H7.862a2 2 0 01-1.995-1.858L5 7m5 4v6m4-6v6m1-10V4a1 1 0
140         </svg>
141         <p>Delete</p>
142         </Link>
143     </div>
144 </td>
145 </tr>
146 </tbody>
147 </table>
148 </div>
149 </div>
150 </div>
151 </div>
152 <footer className="bg-gray-50 shadow mt-10">
153     <div className="px-4 py-12 mx-auto space-y-8 overflow-hidden">
154         <nav className="flex flex-wrap justify-center -mx-5 -my-2">

```

```

160         </nav>
161     </div>
162     <div className="flex justify-center mt-8 space-x-6">
163         <div className="text-gray-400 hover:text-gray-500">
164             <span className="sr-only">Facebook</span>
165             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
166                 <path fill-rule="evenodd" d="M22 12c0-5.523-4.477-10-10-10S2 6.477 2 12c0 4.991 3.657 9.128 8.438
167             </svg>
168         </div>
169         <div className="text-gray-400 hover:text-gray-500">
170             <span className="sr-only">Instagram</span>
171             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
172                 <path fill-rule="evenodd" d="M12.315 2c2.43 0 2.784.013 3.808.06 1.064.049 1.791.218 2.427.465a4.9
173             </svg>
174         </div>
175         <div className="text-gray-400 hover:text-gray-500">
176             <span className="sr-only">Twitter</span>
177             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
178                 <path d="M8.29 20.251c7.547 0 11.675-6.253 11.675-11.675 0-.178 0-.355-.012-.53A8.348 8.348 0 022
179             </svg>
180         </div>
181         <div className="text-gray-400 hover:text-gray-500">
182             <span className="sr-only">GitHub</span>
183             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
184                 <path fill-rule="evenodd" d="M12 2C6.477 2 2 6.484 2 12.017c0 4.425 2.865 8.18 6.839 9.504.509.26
185             </svg>
186         </div>

```

```

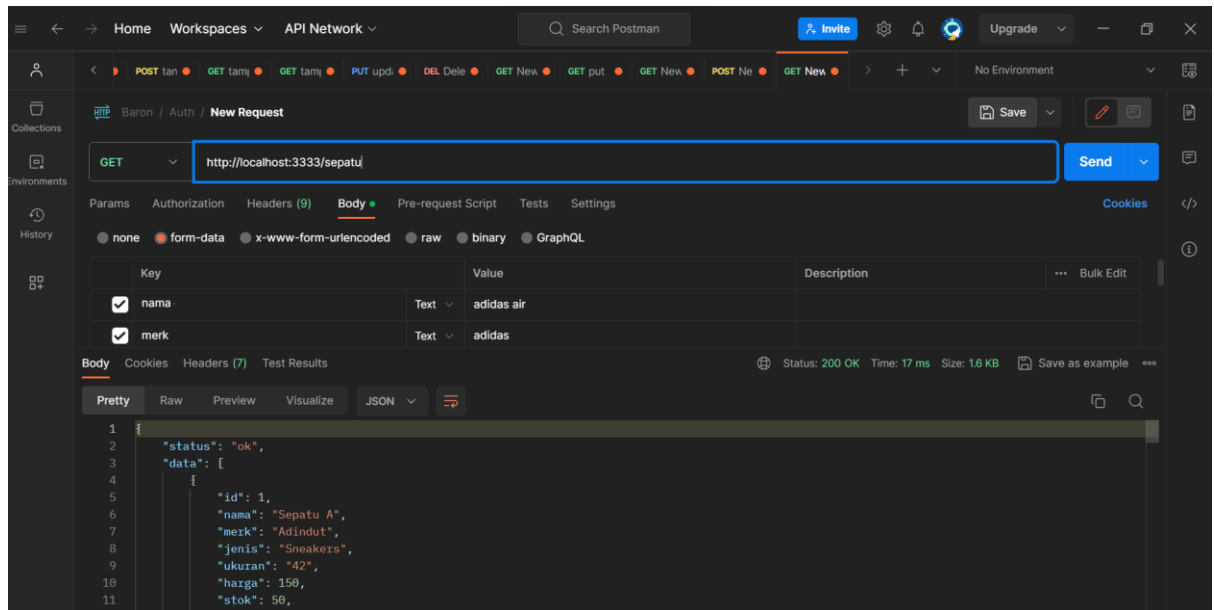
187         <div className="text-gray-400 hover:text-gray-500">
188             <span className="sr-only">Dribbble</span>
189             <svg className="w-6 h-6" aria-hidden="true" fill="currentColor" viewBox="0 0 24 24">
190                 <path fill-rule="evenodd" d="M12 2C6.48 2 2 6.48 2 12s4.48 10 10 10c5.51 0 10-4.48 10-10S17.51 2
191             </svg>
192         </div>
193     </div>
194     <p className="mt-8 text-base leading-6 text-center text-gray-400">
195         © 2023 Baron's Company, Inc. All rights reserved.
196     </p>
197 </div>
198 </div>
199 </div>
200 </div>
201 </div>
202 </div>
203 </div>

```

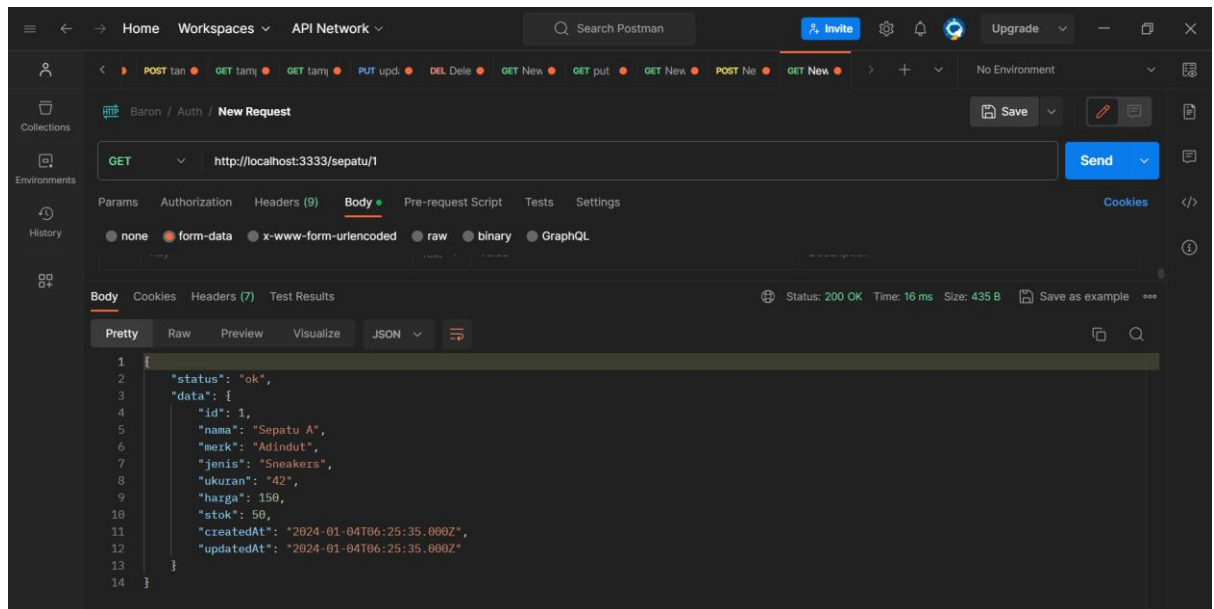
## Bab III

### Test Backend Restful API dengan Postman

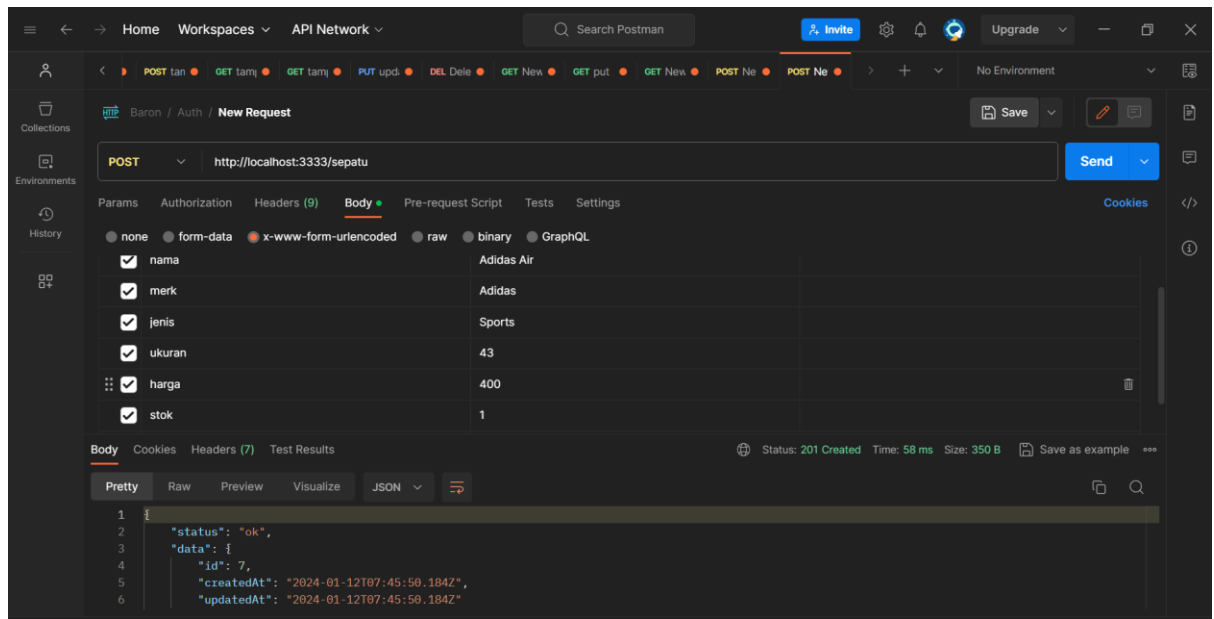
#### Untuk Menampilkan Data



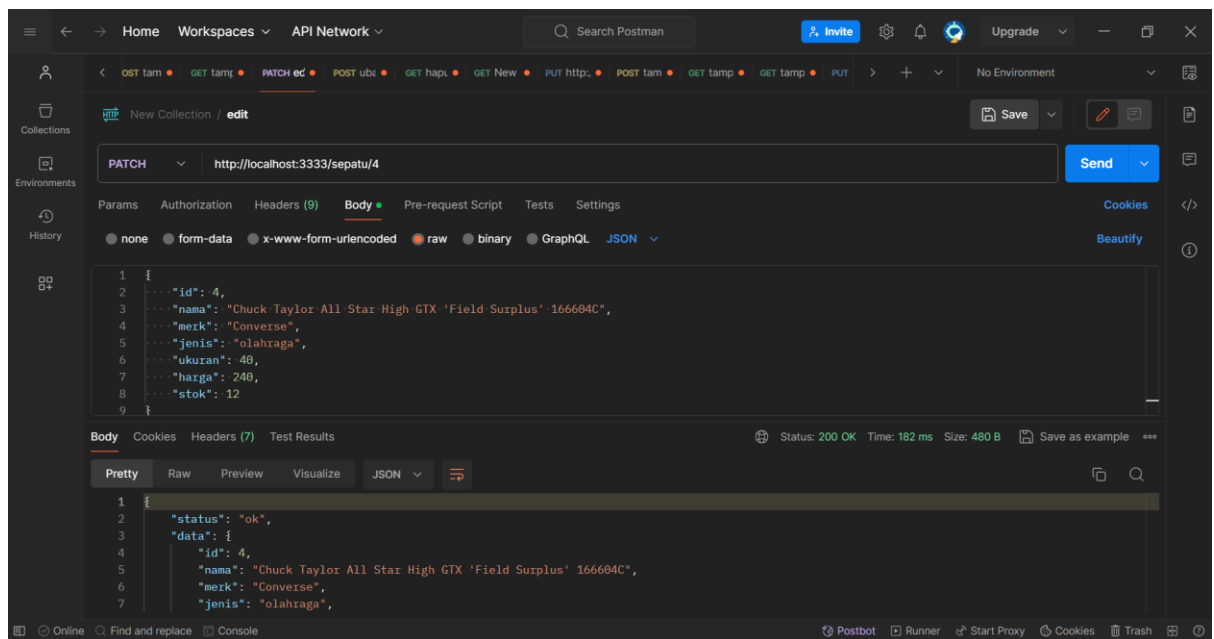
#### Untuk Menampilkan Data By id



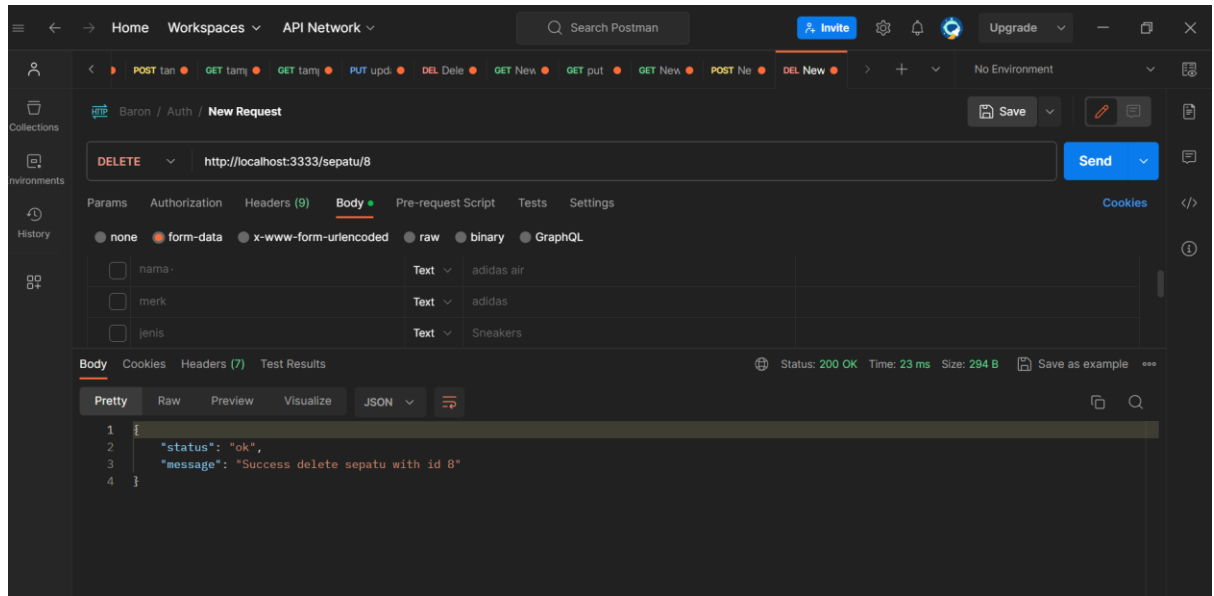
## Untuk Menambahkan Data



## Untuk Update Data

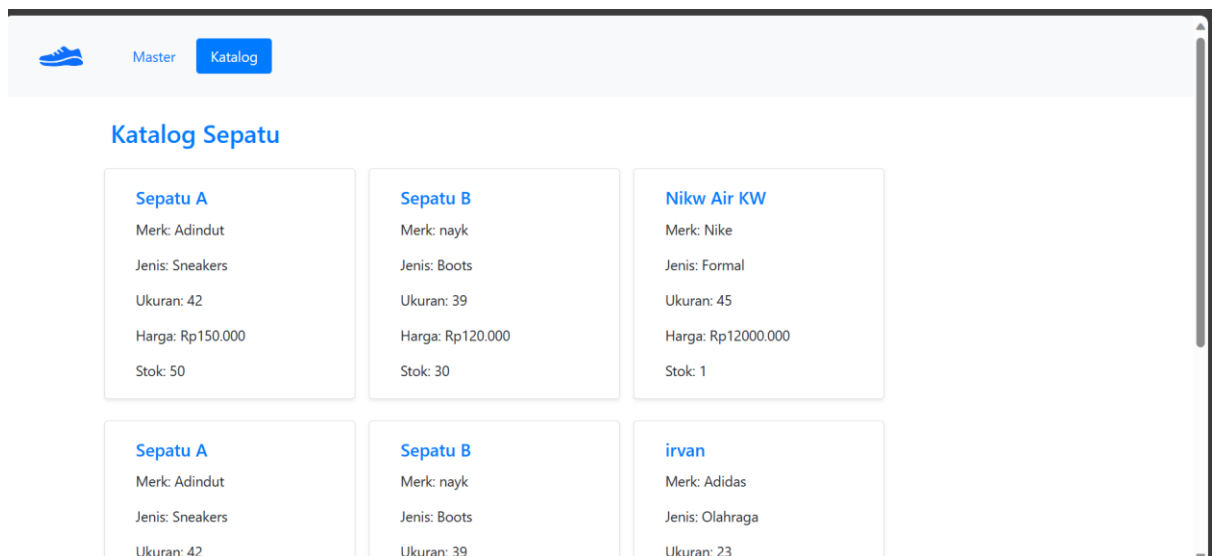


## Untuk Hapus data



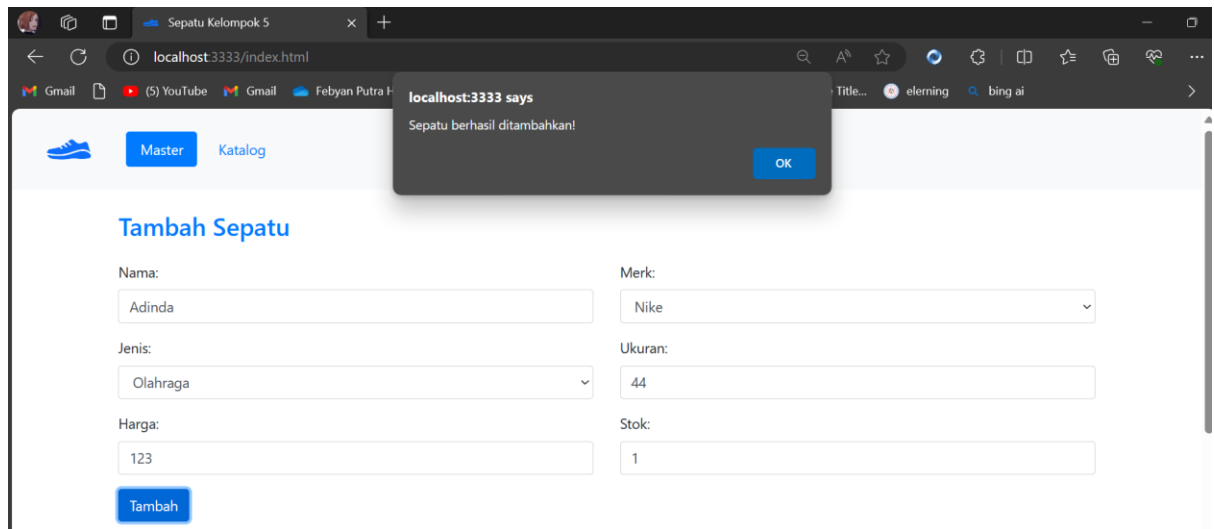
## Test Frontend Pada tampilan Web

### Untuk menampilkan Data





## Untuk Menambahkan data



The screenshot shows a web browser window with the address bar displaying 'localhost:3333/index.html'. The page has a navigation bar with a shoe icon, a 'Master' button, and a 'Katalog' link. The main content area is titled 'Tambah Sepatu' and contains a form with the following fields:

- Nama:
- Merk:
- Jenis:
- Ukuran:
- Harga:
- Stok:

A blue 'Tambah' button is located at the bottom left of the form. A dark toast notification is displayed in the center of the screen, stating 'localhost:3333 says' and 'Sepatu berhasil ditambahkan!' with an 'OK' button.