Laporan Praktikum

Mata Kuliah PBO

Tugas Frontend dan Backend

Dosen Pengampu :

Willdan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh :

(Nur’aini Dwi Anra)

(2301148)

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

1. Pendahuluan

Pada praktikum ini, kami mengembangkan aplikasi web menggunakan arsitektur yang terorganisir dengan modularisasi untuk backend, frontend, dan integrasi database. Backend dibangun menggunakan Node.js dan Express dengan arsitektur berbasis controller dan middleware untuk memastikan skalabilitas dan pemisahan tanggung jawab. Frontend dibangun menggunakan React.js dengan Redux untuk manajemen state. Database dikelola menggunakan PhpMyAdmin dengan skema berbasis tabel MySQL.

2. Struktur Program

Aplikasi ini dibangun dengan menggunakan beberapa t

1) **Frontend**
   - Framework: React.js
   - State Management: Redux
   - Tools: Axios untuk komunikasi API
2) **Backend**
   - Framework: Express.js
   - Struktur Modular:
     - **config**: Konfigurasi database menggunakan Sequelize dan variabel .env.
     - **controllers**: Menangani logika endpoint.
     - **middleware**: Middleware untuk otentikasi dan validasi.
     - **models**: ORM untuk tabel database menggunakan Sequelize.
     - **routes**: Routing untuk endpoint API.
3) **Database**
   - Tabel:
     - users (ID, nama, email, password, role)
     - sessions (sid, expires, data)
4) **Proses Kerja**
   - Backend API: CRUD (Create, Read, Update, Delete) untuk tabel users.
   - Middleware: Menyediakan lapisan validasi dan otentikasi.
   - Frontend: Menampilkan data dari database melalui API.
   - Testing: Dilakukan dengan request.rest untuk menguji endpoint secara manual.

3. Penjelasan Code

1) Backend (Express.js)

- **File index.js (Main Server):**

javascript

Copy code

```
const express = require('express');
const cors = require('cors');
const dotenv = require('dotenv');
const bodyParser = require('body-parser');
const db = require('./config/database'); // Sequelize config
const userRoutes = require('./routes/users');

dotenv.config();
const app = express();
app.use(cors());
app.use(bodyParser.json());

// Routes
app.use('/users', userRoutes);

db.authenticate()
   .then(() => console.log('Database connected...'))
   .catch(err => console.log('Error: ' + err));

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

- **File .env (Environment Variables):**

plaintext

Copy code

```
DB_HOST=localhost
DB_USER=root
DB_PASS=yourpassword
```

DB_NAME=yourdatabase

- **File routes/users.js (Routing untuk Users):**

javascript
Copy code

```
const express = require('express');
const router = express.Router();
const { getUsers, updateUser } = require('../controllers/userController');

router.get('/', getUsers);
router.patch('/:id', updateUser);

module.exports = router;
```

- **File controllers/userController.js (Logika API):**

javascript
Copy code

```
const { User } = require('../models');

exports.getUsers = async (req, res) => {
  try {
    const users = await User.findAll();
    res.json(users);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
};

exports.updateUser = async (req, res) => {
  try {
    const { id } = req.params;
```

```
    const { name, email } = req.body;
    await User.update({ name, email }, { where: { id } });
    res.json({ message: 'User updated successfully' });
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
};
```

2. Frontend (React + Redux)

- Redux Store: Tidak berubah dari laporan sebelumnya.
- React Component (UserList.js): Tidak berubah dari laporan sebelumnya.

3. Database (PhpMyAdmin)

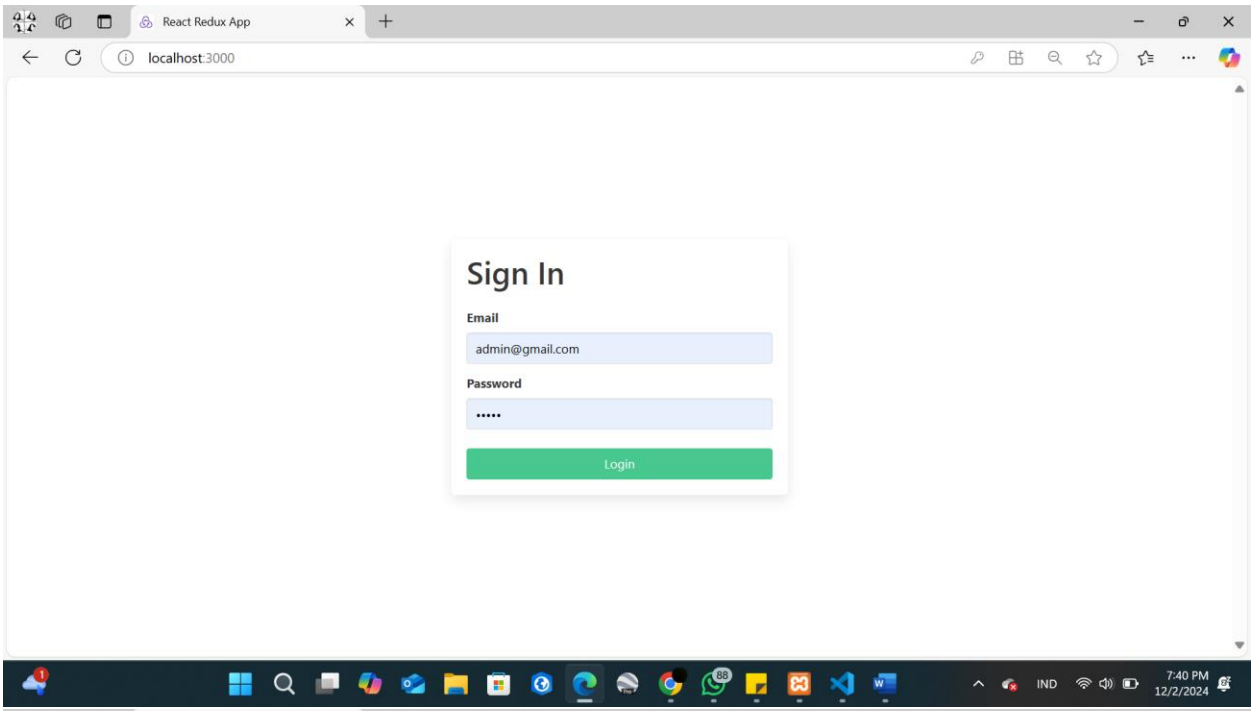
- Tabel Users:

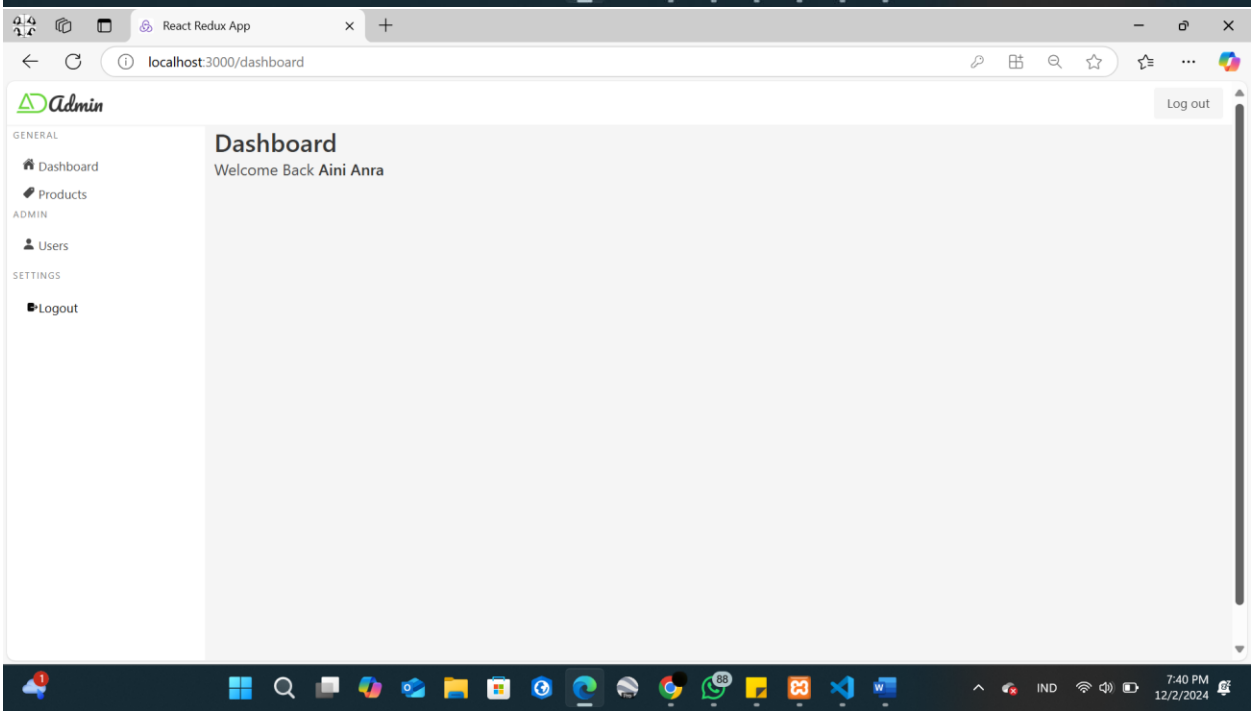
sql

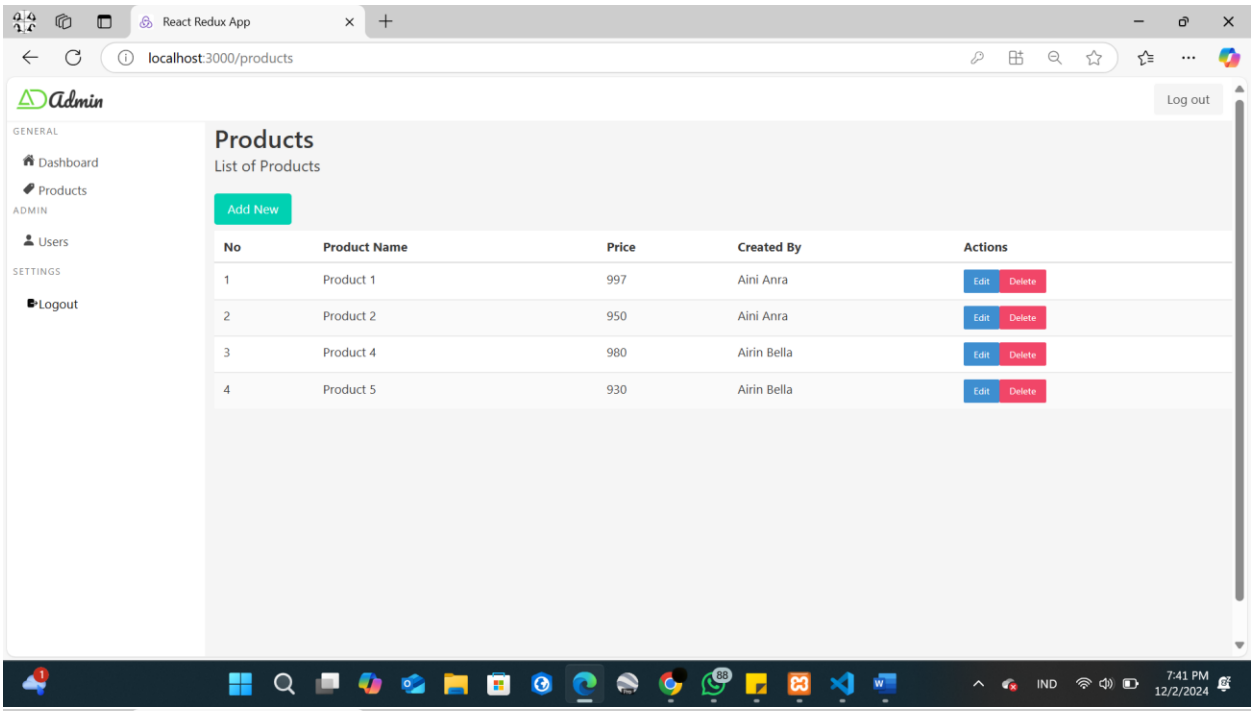
Copy code

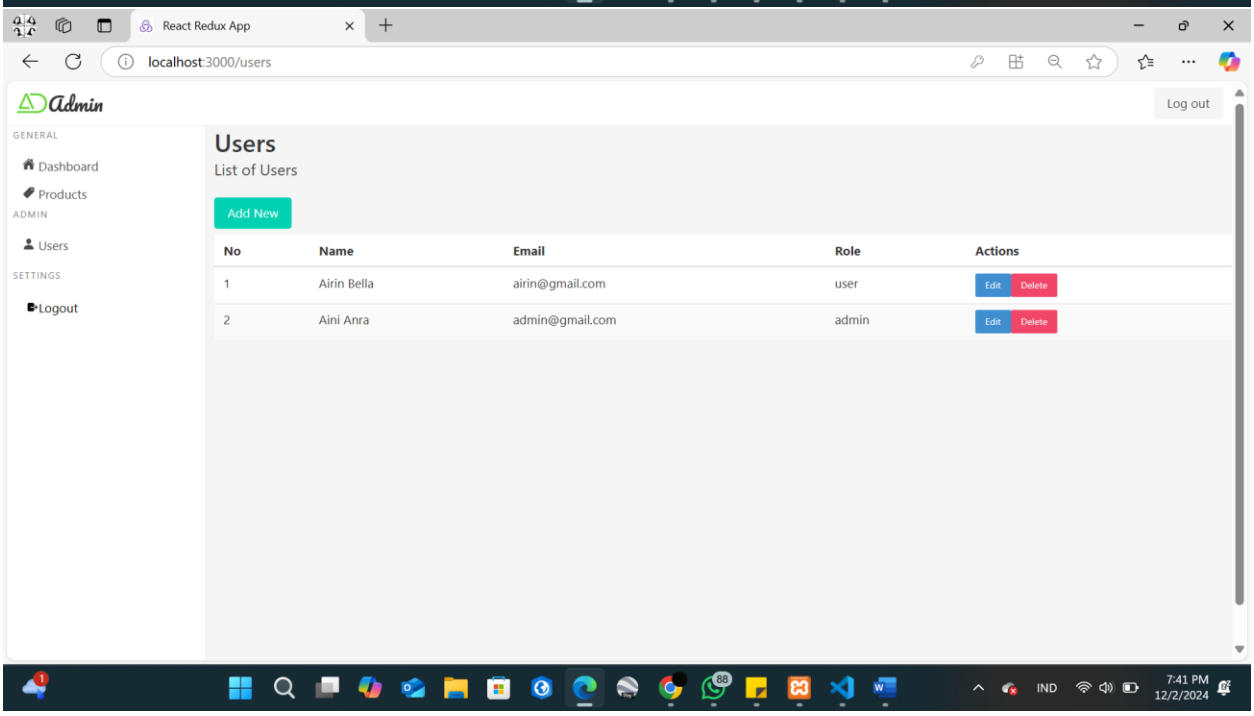
```
CREATE TABLE `users` (
  `id` INT AUTO_INCREMENT PRIMARY KEY,
  `name` VARCHAR(50),
  `email` VARCHAR(100),
  `password` VARCHAR(255),
  `role` ENUM('user', 'admin')
);
```

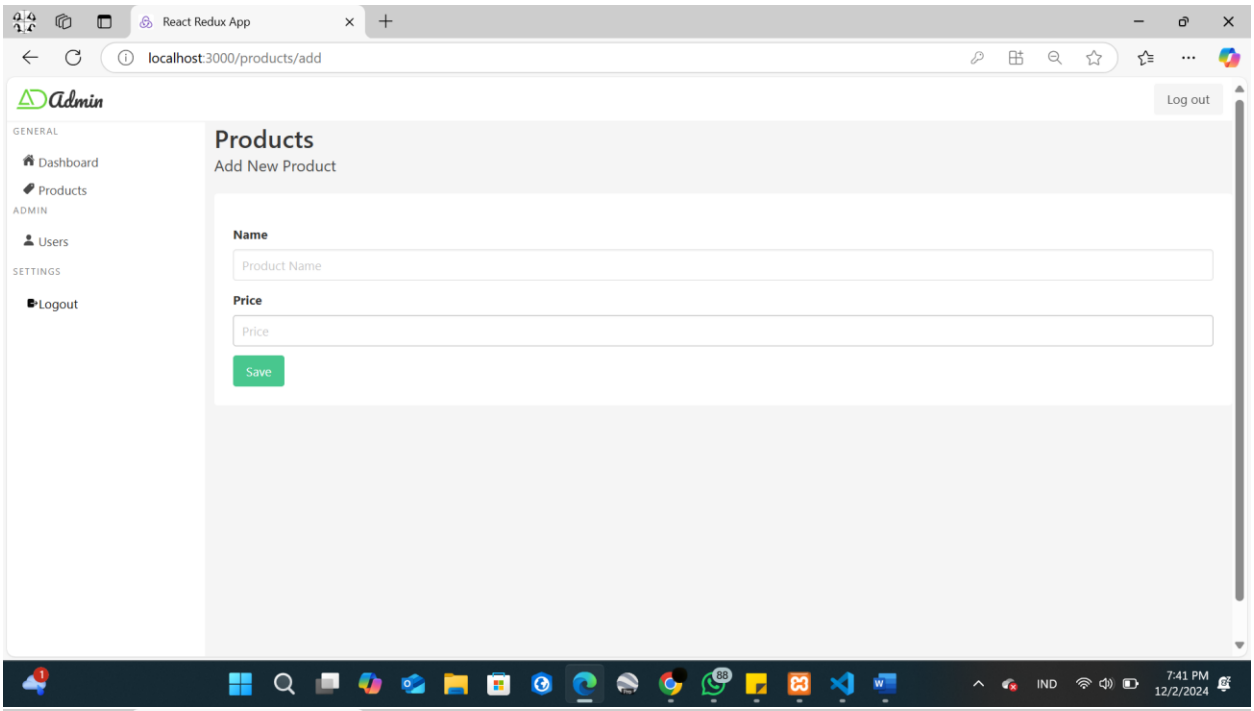
4. Hasil

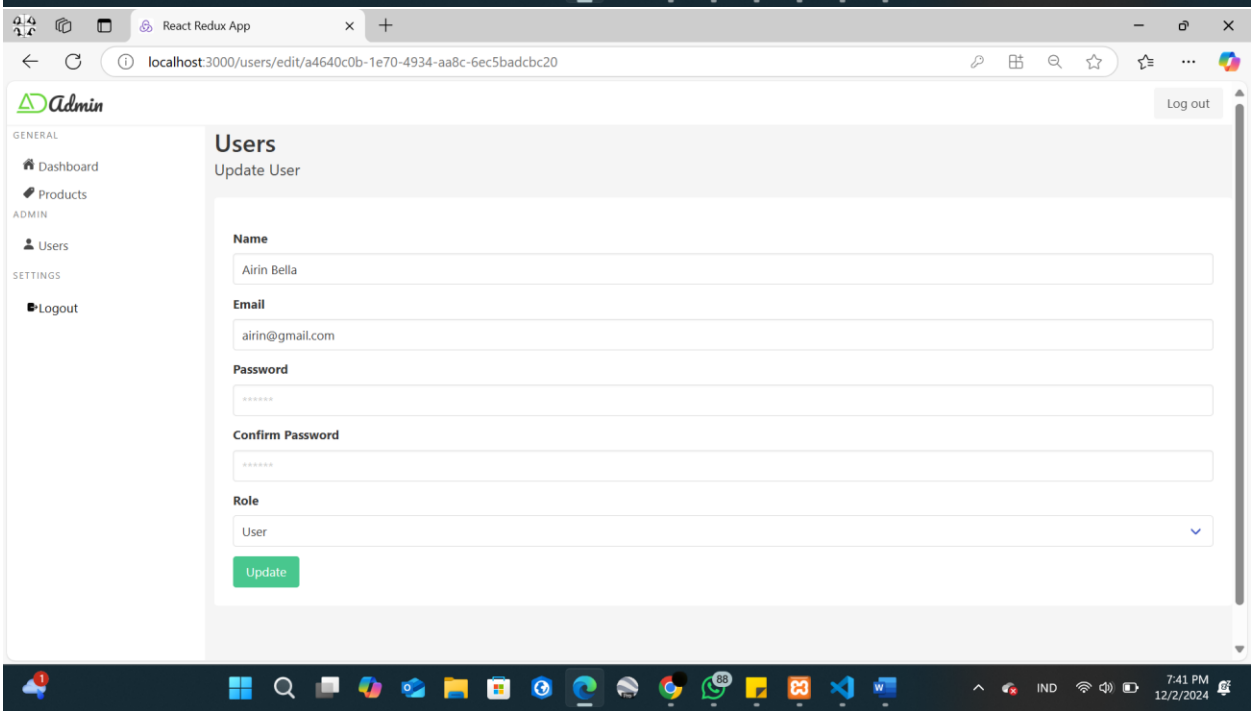

React Redux App
localhost:3000
Sign In
Email
admin@gmail.com
Password
.....
Login
IND
7:40 PM
12/2/2024

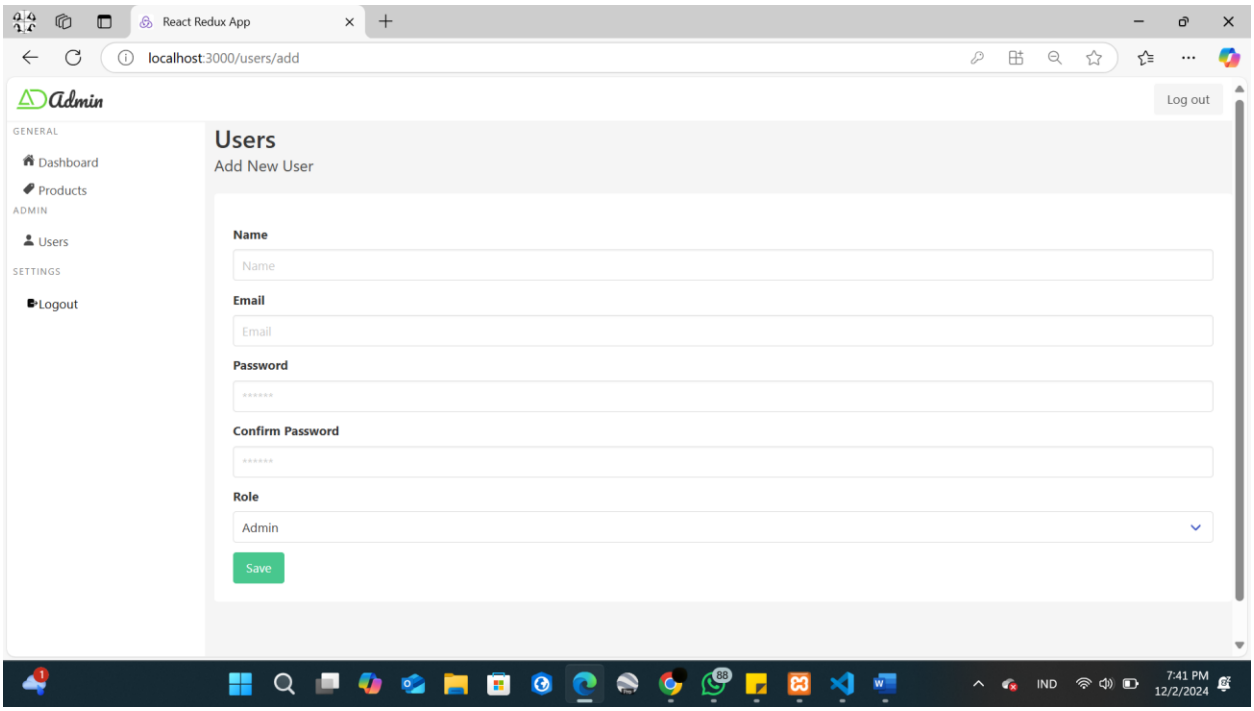

React Redux App
localhost:3000/dashboard
Admin
Log out
GENERAL
Dashboard
Products
ADMIN
Users
SETTINGS
Logout
Dashboard
Welcome Back Aini Anra
IND
7:40 PM
12/2/2024


React Redux App
localhost:3000/products
Admin
Log out
GENERAL
Dashboard
Products
ADMIN
Users
SETTINGS
Logout
Products
List of Products
Add New
No
Product Name
Price
Created By
Actions
1
Product 1
997
Aini Anra
Edit
Delete
2
Product 2
950
Aini Anra
Edit
Delete
3
Product 4
980
Airin Bella
Edit
Delete
4
Product 5
930
Airin Bella
Edit
Delete
IND
7:41 PM
12/2/2024


React Redux App
localhost:3000/users
Admin
Log out
GENERAL
Dashboard
Products
ADMIN
Users
SETTINGS
Logout
Users
List of Users
Add New
No
Name
Email
Role
Actions
1
Airin Bella
airin@gmail.com
user
Edit
Delete
2
Aini Anra
admin@gmail.com
admin
Edit
Delete
IND
7:41 PM
12/2/2024


React Redux App
localhost:3000/products/add
Admin
Log out
GENERAL
Dashboard
Products
ADMIN
Users
SETTINGS
Logout
Products
Add New Product
Name
Product Name
Price
Price
Save
IND
7:41 PM
12/2/2024


React Redux App
localhost:3000/users/edit/a4640c0b-1e70-4934-aa8c-6ec5badcbc20
Admin
Log out
GENERAL
Dashboard
Products
ADMIN
Users
SETTINGS
Logout
Users
Update User
Name
Airin Bella
Email
airin@gmail.com
Password
Confirm Password
Role
User
Update
IND
7:41 PM
12/2/2024



5. Kesimpulan

Praktikum ini menunjukkan pentingnya modularisasi dalam pengembangan aplikasi web. Dengan memisahkan logika ke dalam controllers, middleware, dan routes, pengelolaan kode menjadi lebih mudah dan terstruktur. Penggunaan .env meningkatkan keamanan aplikasi, sementara pengujian menggunakan file request.rest mempermudah validasi API sebelum integrasi dengan frontend. Kombinasi React Redux di frontend dan Express.js di backend berhasil menciptakan sistem aplikasi web yang responsif dan efisien.