FULL-STACK NOTES APPLICATION DOCUMENTATION

Author: Group C
Date: 3/7/2025

ABSTRACT

This document provides comprehensive documentation for the full-stack Notes Application, developed using **Next.js, React, Express.js, and MongoDB**. The purpose of this system is to provide users with a secure platform to create, edit, delete, and manage notes efficiently. The documentation includes system architecture, technology stack, dependencies, API endpoints, and setup procedures.

---

1. INTRODUCTION

1.1 Background

With the increasing need for **digital note-taking solutions**, this project aims to develop a full-stack **Notes Application** that enables users to store their notes securely in a **cloud database**. The application features **user authentication**, **CRUD operations** for notes, and a **modern user interface** designed with React and TailwindCSS.

1.2 Objectives

The objectives of this project include:

- Developing a **user-friendly** and **responsive** web application for note-taking.
- Implementing **user authentication** and authorization using **JWT**.
- Enabling **CRUD (Create, Read, Update, Delete)** operations for managing notes.
- Securing user data with **MongoDB and password encryption**.

1.3 Scope

This project focuses on a **full-stack** web-based Notes Application that includes:

- **Frontend**: Developed using **Next.js** with **React**, **Redux Toolkit**, and **TailwindCSS**.
- **Backend**: Implemented using **Node.js**, **Express.js**, and **MongoDB**.
- **Authentication**: Managed using **JWT (JSON Web Tokens)** for secure user access.
- **Data Management**: Notes are stored in **MongoDB** and linked to user accounts.

---

2. SYSTEM ARCHITECTURE

The system follows a **client-server architecture**, where the frontend communicates with the backend API to perform various operations.

2.1 Architectural Diagram

```
+----------------------+       +----------------------+
|   Frontend (React)   | <-->  |   Backend (Express)  |
|   Next.js + Redux    |       |   Node.js + MongoDB  |
+----------------------+       +----------------------+
```

---

3. TECHNOLOGY STACK

The project utilizes the following technologies:

Category	Technology Used
Frontend	Next.js (React 19)
State Management	Redux Toolkit
Styling	TailwindCSS
Backend	Node.js, Express.js
Database	MongoDB, Mongoose
Authentication	JWT (JSON Web Token)
Security	Bcrypt.js (Password Hashing)
API Communication	Axios (HTTP Requests)

---

4. SYSTEM IMPLEMENTATION

4.1 Installation and Setup

4.1.1 Backend Setup

1. Navigate to the backend directory:

```
cd backend
```

2. Install dependencies:

```
npm install
```

3. Create a `.env` file and configure the following variables:

```
MONGO_URI=your_mongodb_connection_string
JWT_SECRET=your_secret_key
```

4. Start the backend server:

```
npm start
```

4.1.2 Frontend Setup

1. Navigate to the frontend directory:

```
cd frontend
```

2. Install dependencies:

```
npm install
```

3. Start the development server:

```
npm run dev
```

---

5. API ENDPOINTS

5.1 Authentication Endpoints

Method	Endpoint	Description
POST	`/api/auth/register`	Register a new user
POST	`/api/auth/login`	Authenticate user & return JWT
GET	`/api/auth/user`	Retrieve logged-in user data

5.2 Notes Management Endpoints

Method	Endpoint	Description
POST	`/api/notes`	Create a new note
GET	`/api/notes`	Retrieve all notes
GET	`/api/notes/:id`	Retrieve a specific note
PUT	`/api/notes/:id`	Update a note
DELETE	`/api/notes/:id`	Delete a note

---

6. RESULTS AND DISCUSSION

The application was successfully developed and tested. It provides a **seamless user experience** with **efficient note management** and **secure authentication mechanisms**. Future enhancements may include:

- **Real-time collaboration** on notes using WebSockets.
- **Tagging and categorization** for better organization.
- **AI-powered search functionality** for easier access to notes.

---

7. CONCLUSION

This project successfully implements a **secure and efficient Notes Application** with **full-stack capabilities**. It leverages **modern web technologies** to deliver a **responsive and user-friendly** experience.

Future work will focus on **enhancing security measures**, **optimizing database performance**, and **introducing real-time collaboration features**.

---

8. REFERENCES

The references below are formatted in **APA (7th edition) style**:

- Facebook. (2024). *React: A JavaScript library for building user interfaces*. Retrieved from https://react.dev
- Google. (2024). *Next.js: The React Framework for Production*. Retrieved from https://nextjs.org/docs
- MongoDB, Inc. (2024). *MongoDB Documentation: A NoSQL database for modern applications*. Retrieved from https://www.mongodb.com/docs
- Mozilla Developer Network. (2024). *JavaScript documentation*. Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript
- Node.js Foundation. (2024). *Node.js Documentation*. Retrieved from https://nodejs.org/en/docs