Notes App Backend - Project Documentation

Internship Details

- **Intern Name:** Marnala Murali Sriram Ganapathi
- **Internship Type:** Long-Term Internship (APSCHE)
- **Organization:** SmartInternz
- **Duration:** Jan 2025 - Apr 2025
- **Domain:** Backend Development with Node.js and MongoDB

1. Project Overview

Objective

The **Notes App Backend** is a RESTful API developed using **Node.js** and **MongoDB**. It provides authentication and CRUD operations for a note-taking application, allowing users to register, log in, and manage their notes securely. This project demonstrates API development, user authentication, and database management.

---

2. Technology Stack

- **Backend Framework:** Node.js with Express.js
- **Database:** MongoDB (NoSQL)
- **Authentication:** JSON Web Tokens (JWT)
- **Middleware:** Express Middleware for authentication and error handling
- **Environment Management:** dotenv for managing environment variables

---

3. Project Structure

Folder & File Organization

/notes-app-backend

```
│ -- /models
│     ├── User.js
│     ├── Note.js
│ -- /routes
│     ├── authRoutes.js
│     ├── noteRoutes.js
│ -- /controllers
│     ├── authController.js
│     ├── noteController.js
│ -- /middleware
│     ├── authMiddleware.js
│     ├── errorMiddleware.js
│ -- /config
│     ├── db.js
│ -- /public
```

```
-- .env
-- index.js         # Main entry point of the backend
-- package.json     # Project dependencies
```

---

4. Features & Functionality

User Authentication

✅ User Registration (/api/auth/register)

✅ User Login (/api/auth/login)

✅ Secure authentication using JWT

Notes Management

✅ Create Notes (/api/notes/create)

✅ Read Notes (/api/notes)

✅ Update Notes (/api/notes/:id)

✅ Delete Notes (/api/notes/:id)

Security & Middleware

✅ JWT-based authentication for protected routes
✅ Error handling middleware for centralized error management

---

5. Installation & Setup

Prerequisites:

- Install **Node.js** (v16 or above)
- Install **MongoDB**

Step 1: Clone the Repository

```
git clone <repository-url>
cd notes-app-backend
```

Step 2: Install Dependencies

```
npm install
```

Step 3: Configure Environment Variables

Create a `.env` file and add:

```
MONGO_URI=your_mongodb_connection_string
JWT_SECRET=your_jwt_secret_key
```

Step 4: Start the Server

```
npm start
```

The API will be running at `http://localhost:5000/`.

---

6. API Endpoints

Authentication Endpoints (`/api/auth`)

Method	Endpoint	Description
POST	/register	Register a new user
POST	/login	Login user and return JWT

Notes Endpoints (`/api/notes`)

Method	Endpoint	Description
POST	/create	Create a new note
GET	/	Get all notes
PUT	/:id	Update a note
DELETE	/:id	Delete a note

---

7. Project Implementation Details

Database Models

User Model (`User.js`)

Defines the structure for user data, including authentication credentials.

Note Model (`Note.js`)

Defines the structure for storing user notes in the database.

Controllers

Authentication Controller (`authController.js`)

Handles user authentication tasks such as registration, login, and JWT token generation.

Notes Controller (noteController.js)

Implements business logic for creating, reading, updating, and deleting notes.

Middleware

Authentication Middleware (authMiddleware.js)

Verifies JWT tokens and ensures that only authenticated users access protected routes.

Error Handling Middleware (errorMiddleware.js)

Manages centralized error handling for all API routes.

---

8. Deployment Guide

To deploy the backend application on a cloud platform:

Step 1: Set Up a Cloud Server

Choose a cloud service like AWS, Heroku, or Railway to host the backend.

Step 2: Configure Environment Variables

Ensure that .env contains correct values for MONGO_URI and JWT_SECRET.

Step 3: Deploy

Use Git to push the code to the cloud server:

```
git add .
git commit -m "Deploying backend"
git push origin main
```

---

9. Conclusion

The **Notes App Backend** provides a secure and efficient solution for managing users and their notes. It demonstrates RESTful API development, authentication handling, and middleware implementation using **Node.js and MongoDB**.

This backend system can be extended with features like cloud storage integration, role-based access control, and additional security measures.

---

10. Acknowledgments

Special thanks to **SmartInternz** and **APSCHE** for the opportunity to work on this project.

Author: Marnala Murali Sriram Ganapathi
GitHub Repository: APSCHE