**Project Title**

**CRUD API**

**Project Overview**

**Introduction:**

The CRUD API project is designed to handle Create, Read, Update and Delete (CRUD) operations in a structured and efficient manner. The primary purpose of this API is to provide back-end service that allows applications to interact with a database, facilitating data management for users.

The main purpose of this project is to build a robust and scalable API that performs CRUD operations for managing users within an application. It allows clients, web applications, to interact with the back-end securely and efficiently.

The Objective of CRUD API is to efficiently manage the data, it helps in maintain the consistency and validation of the data in the database

Many modern applications require a way to interact with the databases for dynamic data operations, like managing user profiles, storing user data, handling other resources. The CRUD API streamlines this process by providing well-structured endpoints that allow seamless data manipulation without direct database interaction from the front end.

It abstracts the complexities of database operations, offering a user-friendly interface for managing data.

**Technical Details**

**Technical Stack:**

Backend: Node.js

Framework: Express.js

Database: MongoDB

Other Technologies: BcryptJS, View Engine, EJS

**Architecture:**

There are basically 5 layers of the CRUD API Architecture:

1. **Client Layer:** In thisthe client sends the HTTP requests (GET, POST, PUT, DELETE) to interact with the API.
2. **Router Layer:** This layer is responsible to handle the client requests to the appropriate controller functions.
3. **Controller Layer:** Handles the core logic for CRUD operations, interacting with the database through the models and sending responses back to the client.
4. **Model Layer:** Defines the schema and methods to interact with the database.
5. **Database Layer:** Stores and manages data.

**Installation**

**Node.js Installation:** Ensure that Node.js is installed. (Check using **node -v** command).

**Project Setup:** Create a directory using command **mkdir crud-api** and go change the directory using **cd crud-api.**

**Initializing Project:** Initialize node.js project using command **npm init -y** which creates a file package.json.

**Packages Installation:** Install necessary packages (bcryptjs, body-parser, dotenv, ejs, express, method-override, mongoose, nodemon) using command **npm i bcryptjs body-parser dotenv ejs express method-override mongoose -D nodemon.**

**bcryptjs:** For the encryption of password while storing.

**body-parser:** To parse the data received from the frontend forms, etc.

**dotenv:** To create and use the **.env** (environment variable file) in the project.

**ejs:** To perform the server side rendering of frontend using ejs view engine.

**Express:** A web framework for handling HTTP requests.

**Method-override:** To override the form GET and POST method to any HTTP request method.

**Mongoose:** ODM library for MongoDB.

**Nodemon:** For restart the server automatically when the code has changes to save.

**Configuration:** Create **app.js** to configure the Express server.

**Running the Application:** Start the application using the command **npm start**.

**Conclusion**

The CRUD API project using NodeJS successfully implemented essential create, read, update and delete operations. By utilizing Express for routing client’s HTTP requests and Mongoose for data management, a scalable and maintainable architecture was achieved.

**Key Objectives:**

* Build a working API that handles data operations.
* Organized the code by separating different tasks.
* Used MongoDB to save and manage the data efficiently.

**Lessons Learned:**

* Breaking the code into small parts makes it easier to manage.
* Handling errors properly is important for smoother operations.
* Improved skills in working with databases like MongoDB.

**Future Enhancements:**

* Adding security features like user login and permissions.
* Validating data more thoroughly before saving.

**Contact Information**

**Project Lead:** Nikunj Bansal

**Email:**nikunj.banssal@gmail.com

**GitHub:** bnikunj31