**StudyNotion**

**Project Description**

StudyNotion is a fully functional ed-tech platform that enables users to create, consume, and rate educational content. The platform is built using the MERN stack, which includes ReactJS, NodeJS, MongoDB, and ExpressJS. StudyNotion aims to provide:

1. A better learning experience for students, making education more accessible and engaging.
2. A platform for instructors to showcase their expertise and connect with learners across the globe.

**Requirement Document:**

1. System Architecture: The StudyNotion ed-tech platform consists of three main components:

* front end
* back end
* database

The platform follows a client-server architecture, with the front end serving as the client and the back end and database serving as the server.

**Front-end**

To build the front end, we use following frameworks and libraries:

* ReactJS, which is a popular JavaScript library for building user interfaces.
* We also use CSS and Tailwind, which are styling frameworks that help make the user interface look good and responsive.
* Additionally, we use some npm packages (react-hot-toast, react-icons, react-otp-input, react-router-dom, react-redux etc) to add extra functionality to the front end.
* To manage the state of the application, we use Redux, which is a popular state management library for React.
* Finally, we use a development environment called VSCode, which is a popular code editor, to develop the front end.

The front end of StudyNotion has all the necessary pages that an ed-tech platform should have. Some of these pages are:

**1. For Students:**

* Homepage: This page will have a brief introduction to the platform, as well as links to the course list and user details.
* Course List: This page will have a list of all the courses available on the platform, along with their descriptions and ratings.
* Wishlist: This page will display all the courses that a student has added to their Wishlist.
* Cart Checkout: This page will allow the user to complete the course purchase.
* Course Content: This page will have the course content for a particular course.
* User Details: This page will have details about the student's account, including their name, email, and other relevant information.
* User Edit Details: This page will allow the student to edit their account details.

**2. For Instructors:**

* Dashboard: This page will have an overview of the instructor's courses, as well as the ratings and feedback for each course.
* Course Management Pages: These pages will allow the instructor to create, update, and delete courses, as well as manage the course content and pricing.
* View and Edit Profile Details: These pages will allow the instructor to view and edit their account details.

**Back-end**

1. Description of the Back-end Architecture: StudyNotion uses a monolithic architecture, with the backend built using Node.js and Express.js, and MongoDB as the primary database. Monolithic architecture refers to a design approach where all the modules of the application are combined into a single large program, with a single codebase, to enable better control, security, and performance.
2. Features and Functionalities of the Back-end: The back end of StudyNotion provides a range of features and functionalities, including:

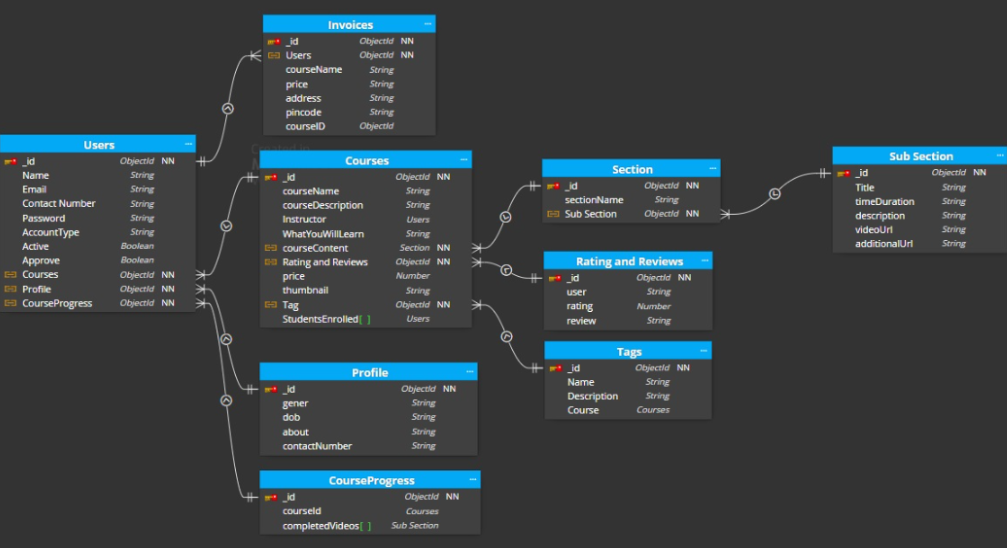
* User authentication and authorization: Students and instructors can sign up and log in to the platform using their email addresses and password.
* The platform also supports OTP (One-Time Password) verification and forgot password functionality for added security.
* Course management: Instructors can create, read, update, and delete courses, as well as manage course content and media. Students can view and rate courses.

**Design Document:**

**Data Models and Database Schema:**

The back end of StudyNotion uses a range of data models and database schemas to manage data, including:

* Student schema: Includes fields such as name, email, password, and course details for each student.
* Instructor schema: Includes fields such as name, email, password, and course details for each instructor.
* Course schema: Includes fields such as course name, description, instructor details, and media content.



**Frameworks, Libraries, and Tools used:**

The back end of StudyNotion uses a range of frameworks, libraries, and tools to ensure its functionality and performance, including:

1. Node.js: Node.js is used as the primary framework for the back end.
2. MongoDB: MongoDB is used as the primary database, providing a flexible and scalable data storage solution.
3. Express.js: Express.js is used as a web application framework, providing a range of features and tools for building web applications.
4. JWT: JWT (JSON Web Tokens) are used for authentication and authorization, providing a secure and reliable way to manage user credentials.
5. Bcrypt: Bcrypt is used for password hashing, adding an extra layer of security to user data.
6. Mongoose: Mongoose is used as an Object Data Modelling (ODM) library, providing a way to interact with MongoDB using JavaScript.

**API Design:**

The StudyNotion platform's API is designed following the REST architectural style. The API is implemented using Node.js and Express.js. It uses JSON for data exchange and follows standard HTTP request methods such as GET, POST, PUT, and DELETE. Sample API requests and responses:

1. GET /api/courses: Get all courses Response: A list of all courses in the database
2. GET /api/courses/:id: Get a single course by ID Response: The course with the specified ID
3. POST /api/courses: Create a new course Request: The course details in the request body Response: The newly created course
4. PUT /api/courses/:id: Update an existing course by ID Request: The updated course details in the request body Response: The updated course
5. DELETE /api/courses/:id: Delete a course by ID Response: A success message indicating that the course has been deleted.

**Deployment:**

The deployment process for the StudyNotion ed-tech platform will involve hosting the application on various cloud-based services. The front end will be deployed using Vercel, a popular hosting service for static sites built with React. The back-end will be hosted on Render, two cloud-based hosting services for applications built with Node.js and MongoDB. The database will be hosted on MongoDB Atlas, a fully managed cloud database service. Overall, the deployment process for StudyNotion will ensure a stable and scalable hosting environment for the application, allowing users to access the platform seamlessly from anywhere in the world.

**Future Enhancements:**

This section discusses potential future improvements to the StudyNotion platform. These enhancements are listed along with an explanation of how they would improve the platform and priority for implementation.

1. Gamification features: Adding gamification features such as badges, points, and leaderboards can increase user engagement and motivation. This would be a medium-priority enhancement.
2. Personalized learning paths: Creating personalized learning paths for each student based on their interests and learning style can increase student satisfaction and success. This would be a high-priority enhancement.
3. Mobile app: Creating a mobile app for the platform would allow for more convenient access to course content and features, and would increase the platform's reach. This would be a high-priority enhancement.
4. Machine learning-powered recommendations: Using machine learning algorithms to provide personalized course recommendations can improve student engagement and satisfaction. This would be a medium to high-priority enhancement. Overall, these enhancements would significantly improve the StudyNotion platform and its offerings to students, instructors, and administrators.