Project planning phase

Assignment of work to team members:

1. V Pavithra– Frontend development
2. N Kavya Sree– Backend development
3. K Muneswari– Testing & deployment
4. M Navya Sree– Final documentation

Frontend Development

The frontend serves acts as the client and the backend acts as the server. The frontend encompasses the user interface, presentation, and integrates the Axios library to facilitate easy communication with the backend through RESTful APIs.

To enhance the user experience, the frontend leverages the Bootstrap and Material UI libraries, creating a real-time and visually appealing interface for users.

Backend Development

On the backend, we utilize the Express js framework to manage server-side logic and communication. Express.js provides a robust foundation for handling requests and responses efficiently.

Express.js is a lightweight and flexible web application framework that runs on Node.js, designed to handle server-side logic and communication with ease. It simplifies the process of managing HTTP requests and responses by providing an intuitive routing system, which allows developers to define how their application should respond to various client requests like GET, POST, PUT, and DELETE. Express also supports the use of middleware—functions that execute during the request-response cycle—which can be used for tasks such as authentication, logging, and error handling. One of its key strengths is its modularity; developers can structure their applications in a clean and maintainable way, separating routes, controllers, and database logic. It integrates seamlessly with tools like Mongoose to connect with MongoDB, making it ideal for building RESTful APIs and full-stack applications.

Testing & deployment

When it comes to testing in modern JavaScript development, React and npm offer a powerful ecosystem of libraries tailored to different layers of your application. React testing libraries, such as React Testing Library and Jest, are designed specifically for testing React components in a way that mirrors real user interactions. These tools help ensure that your UI behaves as expected by simulating clicks, form inputs, and rendering logic. On the other hand, npm testing libraries cover a broader range of use cases, including backend testing. Libraries like Mocha, Chai, and Super test are commonly used to test Node.js applications, especially APIs built with Express.js

Final Documentation

This documentation typically includes:

- **Project Summary**: A clear overview of what was delivered, including goals, features, and outcomes.
- **Technical Documentation**: Details about the codebase, APIs, database schema, or deployment steps—especially useful if the client or another developer will maintain the project.
- **User Guide or Manual**: Instructions for using the product or system, tailored to the client's level of technical knowledge.
- **Testing Reports**: Evidence of quality assurance, such as test cases, bug fixes, and performance benchmarks.
- **Invoice and Payment Records**: Final billing, including hours worked, rates, and payment terms.
- **Handover Checklist**: A list of all deliverables, credentials (like admin logins), and access rights transferred to the client.
- **Feedback or Testimonial Request**: A polite prompt for the client to share their experience, which can help you build credibility.