LOW LEVEL DESIGN

E-COMMERCE APPLICATION

Written By: SAHIL PARAG KINDO

Table of Contents

Why Architecture Design?

The "Architecture Design" phase of a project involves planning and defining the overall structure, components, interactions, and technologies that will be used to build the application. It serves as the blueprint for how the system will be organized and how its different parts will work together. Here's an overview of why architecture design is important:

- Clarity and Understanding: A well-designed architecture provides a clear structure that helps all stakeholders understand how the application's components fit together and how they interact. This understanding is crucial for effective communication among team members, project managers.
- Scalability: A proper architecture design takes into account potential future growth and scalability. It lays the foundation for the application to accommodate increased user loads, data volume, and feature additions without requiring major overhauls.
- Maintainability: A modular and organized architecture makes it easier to maintain, enhance, and extend the application over time. Changes to one part of the application should have minimal impact on other parts, reducing the risk of unintended consequences.
- Code Reusability: A well-structured architecture encourages the creation of reusable components and modules.

1. Introduction

This architecture document provides a comprehensive overview of the technical design and structure of the eCommerce MERN project. It outlines the technology stack, architectural components, and how they interact to create a functional application.

2. Technology Stack

- Front-End: React
- Back-End: Node.js with Express.js
- Database: MongoDB
- State Management: Redux or React Context
- Communication: Axios for API calls
- Deployment
- Authentication: JWT (JSON Web Tokens)
- Payment Integration: [Paytm]
- Testing: Jest, React Testing Library

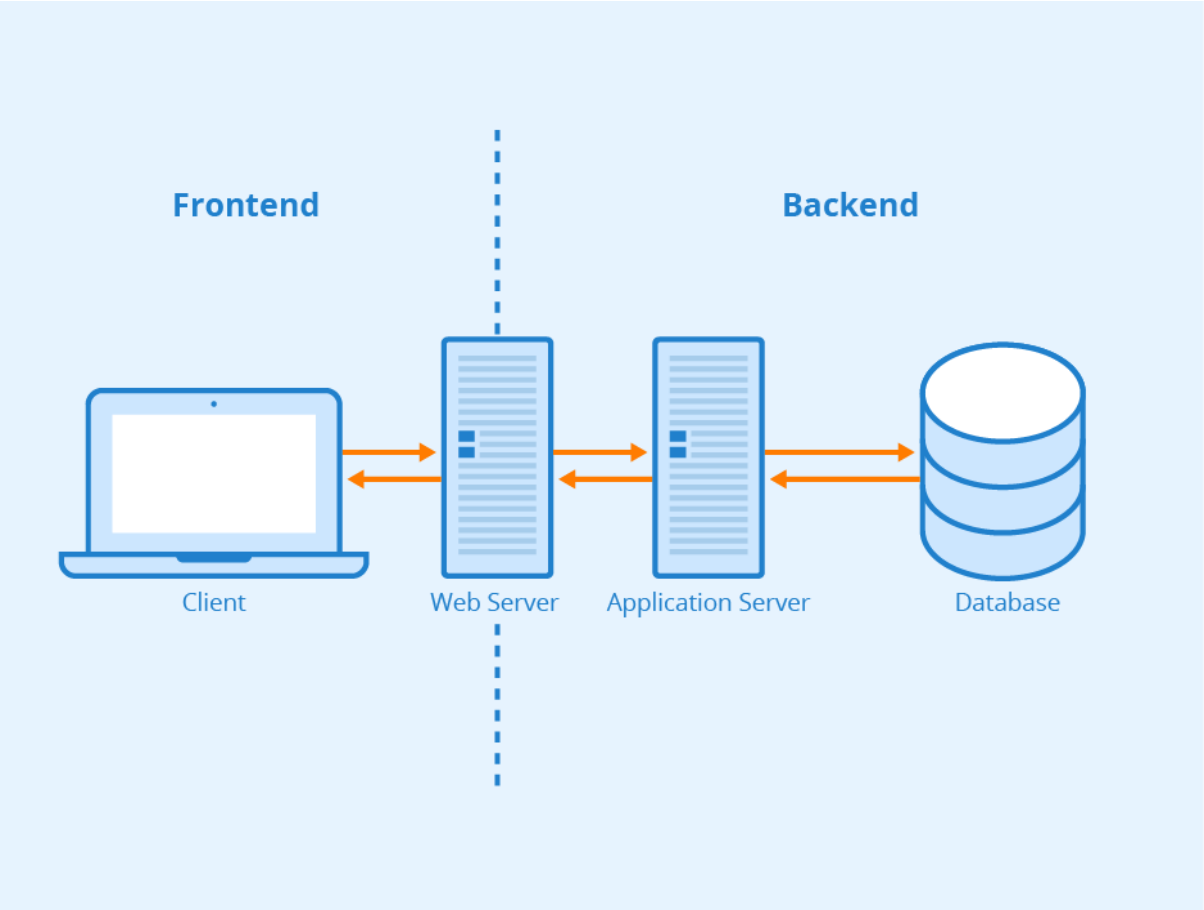
3. Architecture Overview

The eCommerce MERN project follows a three-tier architecture:

Front-End (Client): Developed using React, rendering the user interface and handling user interactions.

Back-End (Server): Implemented with Node.js and Express.js, responsible for handling API requests, business logic, and database interactions.

Database (MongoDB): Stores user information, product details, orders, and other relevant data.



4. Front-End (Client)

Components and State Management

UI components structured for reusability and maintainability.

State managed using Redux for global data or React Context for localized state.

Components include Navbar, ProductList, ProductDetail, Cart, Checkout, OrderHistory, etc.

API Communication

Axios used for making HTTP requests to server API endpoints.

Interaction with server-side API routes for fetching data and performing actions.

5. Back-End (Server)

Routes and Controllers

Express.js routes defined for user authentication, product management, cart actions, and order processing.

Controllers handle request processing, invoking services for business logic.

Middleware and Authentication

Custom middleware for route protection, authorization, and error handling.

JWT used for user authentication and securing protected routes.

Database Schema

MongoDB used for data storage.

User Model: Name, Email, Password Hash, Role

Product Model: Name, Description, Price, Image URL

Cart Model: User ID, Product IDs, Quantity

Order Model: User ID, Product IDs, Total Amount.

6. Deployment Strategy

Application deployed on a cloud platform (Heroku, AWS).

Environment variables used to manage configuration and sensitive information.

7. Security Measures

HTTPS enforced for secure communication.

Data encryption and protection of sensitive user information.

Input validation and sanitization to prevent security vulnerabilities.

8. Conclusion

This architecture document provides a detailed insight into the eCommerce MERN project's technical design, components, and interactions.

Following best practices and adhering to security standards will ensure the application's reliability, security, and scalability.