**Project Design Phase-II**

**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 20-06-2025 |
| Team ID | LTVIP2025TMID21133 |
| Project Name | ShopEZ : One-Stop Shop For Online Purchases |
| Maximum Marks | 4 |

**Technical Architecture:**

The technical architecture of the **ShopEZ WebApp** replicates a real-time e-commerce shopping and management system, built using the **MERN stack (MongoDB, Express.js, React.js, Node.js)**. This architecture follows a modular and scalable approach, ensuring flexibility, maintainability, and efficient handling of concurrent user interactions across different roles—customers, admins, and sellers (future scope).

The **frontend** is developed using **React.js** and hosted on **Netlify**, providing fast, secure, and continuous deployment. It offers a responsive and intuitive UI that supports dynamic rendering of components like product catalogs, shopping carts, and checkout modules, while ensuring seamless user interactions.

The **backend** is built with **Node.js and Express.js**, deployed on **Render**, and serves as the core logic hub. It handles user management, order processing, admin controls, product management, and secure API communications with external services.

**MongoDB Atlas**, a cloud-hosted NoSQL database, is used as the central data store. It ensures high availability, scalable data access, and security for managing product inventories, user

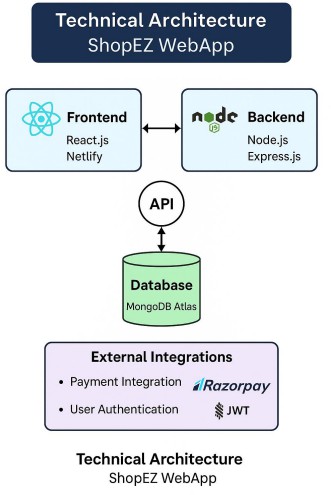
data, order details, and admin operations.

This architecture effectively simulates a real-world e-commerce ecosystem, optimized for performance, real-time interaction, and easy future integration of seller functionalities.

* **Modular architecture** separating frontend, backend, and database
* **Hosted frontend on Netlify**, **backend on Render**, **database on MongoDB Atlas** •

**Secure payment integration** using Razorpay

* **User authentication and role management** using JWT



**Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | Web-based UI for customers and admins | HTML, CSS,  Tailwind CSS, React.js |
| 2. | Application Logic - User Management | Authentication, registration, role based access | Node.js,  Express.js, JWT |
| 3. | Application Logic - Order Processing | Cart, checkout, payment processing | Node.js, Express.js |
| 4. | Application Logic - Admin Panel | Product & category  management, order tracking, user monitoring | Node.js, Express.js |

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | Database | Stores users, products, orders, admin details | MongoDB (NoSQL) |
| 6. | Cloud Database | Cloud-hosted database for scalability and availability | MongoDB Atlas |
| 7. | File Storage | Product images and user-uploaded assets | Cloudinary |
| 8. | External API -  Payment Gateway | Secure payment processing | Razorpay API |
| 9. | External API -  Location Services | Delivery optimization (future integration) | Google Maps API |

**Application Characteristics:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **S.No** | **Characteristics** | **Description** | **Technology** |  |
| 1. | Open-Source Frameworks | Modern stack for frontend and backend | React.js, Node.js, Express.js (MERN) |
| 2. | Security  Implementations | JWT-based auth, secure  APIs, encrypted payments | JWT, HTTPS, Razorpay |
| 3. | Scalable  Architecture | Modular services for handling growth and user load | Node.js + MongoDB on cloud |
| 4. | Availability | Cloud hosting for  continuous uptime and failover handling | Netlify, Render, MongoDB Atlas |
| 5. | Performance | Optimized rendering, fast APIs, local caching support | Redis (optional), IndexedDB, CDN |