1. Abstraction

ShopEZ is an e-commerce platform that enables users to browse products, manage their shopping cart, and make secure purchases. The system also includes a robust admin dashboard for managing inventory, orders, and user accounts. It uses a modular design to ensure scalability, maintainability, and performance, leveraging the MERN stack.

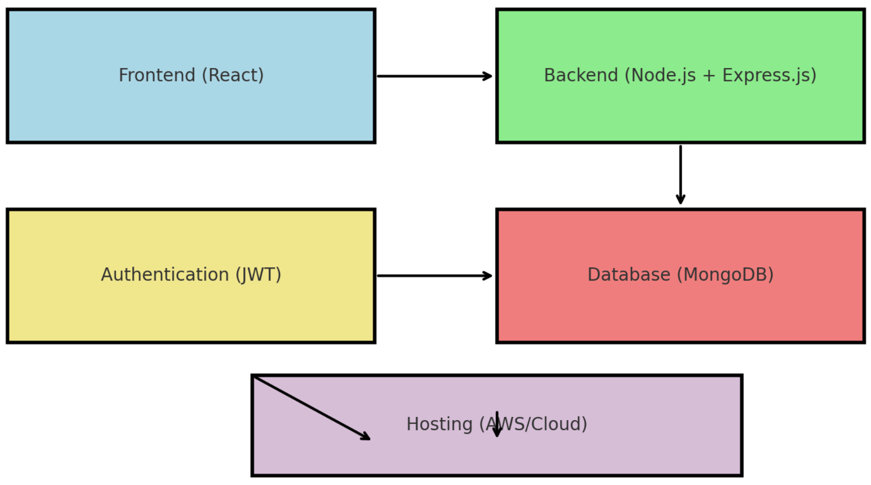
2. Block Diagram

Components to Include:

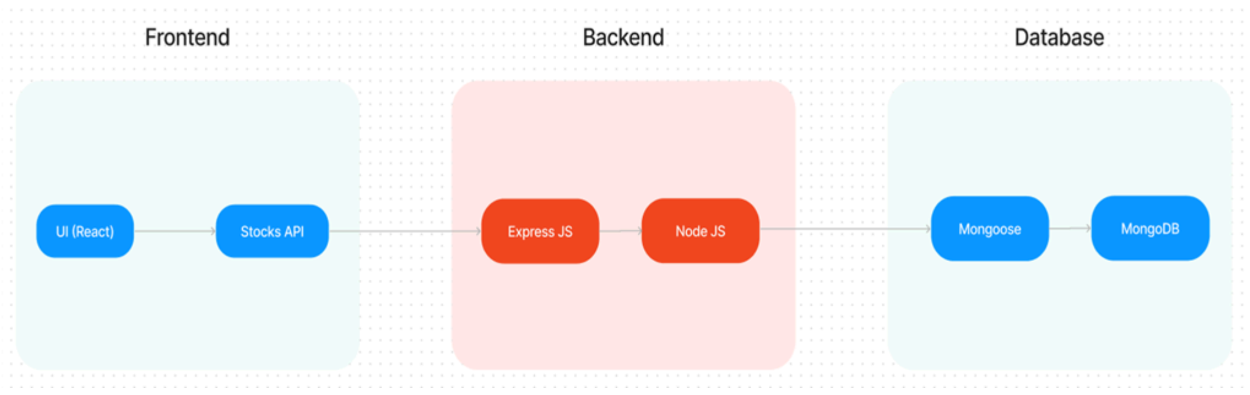
- Frontend (React): Handles the user interface.
- Backend (Node.js, Express.js): Manages API requests and server logic.
- Database (MongoDB): Stores product, user, and order data.
- Authentication (JWT): Secures user sessions.
- Hosting: Deployment on AWS or similar platforms.

1. Frontend: User Interaction → React Component → API Requests
2. Backend: API Handling → Business Logic → Database Queries
3. Database: Product, User, Order Collections
4. Deployment: Frontend and Backend are hosted on servers.

ShopEZ Block Diagram



TECHNICAL ARCHITECTURE:



In this architecture diagram:

- The frontend is represented by the "Frontend" section, including user interface components such as User Authentication, Cart, Products, Profile, Admin dashboard, etc.,
- The backend is represented by the "Backend" section, consisting of API endpoints for Users, Orders, Products, etc.,It also includes Admin Authentication and an Admin Dashboard.
- The Database section represents the database that stores collections for Users, cart, Orders and Product.

Tools Used

1. Development Tools

- **Visual Studio Code**: Code editor for writing and managing code.
- **Node.js**: JavaScript runtime for backend development.
- **React.js**: Framework for building the frontend UI.
- **MongoDB**: NoSQL database for managing data.
- **Mongoose**: ODM library for MongoDB and Node.js integration.

2. Deployment Tools

- **Netlify** or **Vercel**: For deploying the frontend.

3. Version Control

- **Git**: For version control and collaboration.
- **GitHub**: For code repository management

4. Miscellaneous

- **Chrome DevTools**: Debugging and optimizing frontend performance.

3. Implementation Steps

1. **Setup Environment**:
    - Install Node.js, React, MongoDB, and Git on your system.
    - Configure MongoDB Compass for database management.
2. **Frontend Development (React)**:
    - Design the user interface (UI) using Figma or similar tools.
    - Develop React components for pages like Home, Product Catalog, Cart, and Checkout.
    - Ensure responsiveness with CSS or frameworks like Tailwind or Bootstrap.
3. **Backend Development (Node.js + Express)**:
    - Set up RESTful APIs for product retrieval, user authentication, and order processing.
    - Integrate authentication using JWT for secure login and session handling.
4. **Database Design (MongoDB)**:
    - Define schemas for User, Product, Order, and Cart entities.
    - Connect the database to the backend using Mongoose.
5. **Integration**:
    - Link the frontend and backend via APIs using Axios or Fetch.

- Test CRUD operations (Create, Read, Update, Delete) for all features.

6. **Deployment**:
    - Host the frontend on platforms like Netlify or Vercel.
    - Deploy the backend and database on AWS, Heroku, or MongoDB Atlas.
    - Test the deployed application for performance and bugs.