

# Full Stack Development

## 1.Introduction :

### Project Title :

ShopEZ:One-Stop Shop for Online Purchases

### Team members :

- 1.Vallepu Srilatha (22481A60A8)
- 2.Valluru Hemasri(23481A54I2 )
- 3.Valluru Pujitha(23481A54i3)
- 4.Mahesh

## 2. Project Overview

### Purpose

ShopEZ: One-Stop Shop for Online Purchases is a full-stack e-commerce platform designed to offer a seamless online shopping experience. The project aims to bridge the gap between buyers and sellers by providing a scalable and feature-rich solution. Built with React.js, Node.js, Express.js, and MongoDB, ShopEZ ensures real-time data handling, secure transactions, and smooth user interactions.

### Features

- User authentication and role-based access control
- Product catalog with search, filter, and sorting functionality
- Shopping cart with quantity management
- Secure checkout and payment integration
- Admin dashboard for managing users, products, and orders
- Order tracking and user order history
- Product reviews and ratings
- Responsive UI for mobile and desktop

## 3. Architecture

### Frontend

- Built with **React.js**
- Uses **React Router** for navigation
- State management with **Redux** or **Context API**
- Axios for API communication

- Tailwind CSS or styled-components for UI styling

## Backend

- Developed using **Node.js** and **Express.js**
- RESTful API structure
- Middleware for authentication, validation, and error handling
- Routes separated by functionality (auth, products, orders, etc.)

## Database

- **MongoDB** for storing users, products, orders, and reviews
- **Mongoose** for schema modeling
- Collections:
  - Users: username, email, password (hashed), role
  - Products: name, description, price, stock, category
  - Orders: user ID, product list, status, timestamps
  - Reviews: user ID, product ID, rating, comment

# 4. Setup Instructions

## Prerequisites

- Node.js (v16 or higher)
- MongoDB (local or cloud)
- Git

## Installation

Clone the repository:

```
git clone https://github.com/yourusername/ShopEZ.git
```

1. `cd ShopEZ`

Install backend dependencies:

```
cd backend
```

2. `npm install`

Install frontend dependencies:

```
cd ../frontend
```

3. `npm install`

4. Set environment variables:

Create a `.env` file in the backend directory with:  
MONGO\_URI=your\_mongodb\_connection\_string  
JWT\_SECRET=your\_jwt\_secret

- PORT=5000

## 5. Folder Structure

### Client (React Frontend)

```
frontend/  
├── public/  
├── src/  
│   ├── components/  
│   ├── pages/  
│   ├── redux/ or context/  
│   ├── App.js  
│   └── index.js
```

### Server (Node.js Backend)

```
backend/  
├── controllers/  
├── models/  
├── routes/  
├── config/  
├── server.js  
└── .env
```

## 6. Running the Application

### Frontend

```
cd frontend  
npm start
```

### Backend

```
cd backend  
npm start
```

## 7. API Documentation

### User Routes

- `POST /api/users/register` - Register a new user

- `POST /api/users/login` - Login and return JWT
- `GET /api/users/profile` - Get user profile (Protected)

### Product Routes

- `GET /api/products` - Get all products
- `GET /api/products/:id` - Get product details
- `POST /api/products` - Add new product (Admin only)

### Order Routes

- `POST /api/orders` - Create a new order
- `GET /api/orders/:id` - Get order by ID
- `GET /api/orders/user/:userId` - Get user's orders

## 8. Authentication

- Uses **JWT (JSON Web Tokens)** for authentication
- Tokens stored in HTTP-only cookies or local storage
- Middleware to protect routes
- Role-based access (User, Admin)

## 9. User Interface

- Homepage with featured products
- Product details page with reviews
- Shopping cart and checkout flow
- Admin dashboard with charts and management panels

*(Include screenshots or GIFs here)*

## 10. Testing

- **Jest** and **React Testing Library** for frontend
- **Mocha/Chai** or **Supertest** for backend API testing

## 11. Screenshots or Demo

- Add screenshots of:
  - Homepage
  - Product detail
  - Cart and checkout
  - Admin dashboard

- Demo link (if deployed): <https://shopez-demo.vercel.app>

## **12. Known Issues**

- Payment gateway integration is mocked
- Some features like wishlist or coupon codes are under development

## **13. Future Enhancements**

- Implement real payment gateway (Stripe/PayPal)
- Add wishlist and coupon features
- Enable user-to-user messaging
- Progressive Web App (PWA) support
- Real-time order updates using WebSockets