SHOPEZ – MERN FULL STACK WEB APPLICATION DOCUMENTATION

INTRODUCTION

- Project Title: Shopez
- Team Members: Individual project (single member)

PROJECT OVERVIEW

Purpose

Shopez is an e-commerce web application designed to offer users a seamless online shopping experience. It allows customers to browse products, view detailed descriptions, place orders, and manage their shopping activities efficiently.

Features

1. **Comprehensive Product Catalog:**
   ShopEZ offers an extensive range of products with detailed descriptions, customer reviews, pricing, and available discounts.

2. **Shop Now Button:**
   Every product includes a "Shop Now" button to initiate the order process quickly.

3. **Order Details Page:**
   After clicking "Shop Now," users are redirected to an order details page to enter their shipping address, payment information, and specific product requirements.

4. **Secure & Efficient Checkout:**
   The platform ensures safe handling of personal data and provides a fast, reliable checkout experience.

5. **Order Confirmation & Review:**
   Users receive confirmation after placing orders and can view complete order details, including shipping and payment information.

ARCHITECTURE

- Frontend (React JS)

  The frontend is built using React and includes UI components for user authentication, product listings, cart, user profile, and the admin dashboard. It uses React components, Context API for state management, and React Router for navigation.

- Backend (Node.js + Express.js)

  The backend provides RESTful API endpoints for users, products, cart, and orders. It handles admin authentication, dashboard operations, and all business logic using Express.js.

- Database (MongoDB)

  MongoDB stores collections such as Users, Products, Cart, and Orders. Mongoose is used for defining schemas and managing database interactions.

SETUP INSTRUCTIONS

Prerequisites

- Node.js
- MongoDB (Local or MongoDB Atlas)
- Git

Installation

1. Clone the repository

```
git clone [your_repo_link]
```

2. Install backend dependencies

```
cd server
npm install
```

3. Install frontend dependencies

```
cd ../client
npm install
```

4. Create .env file in the /server folder

```
PORT=4000
MONGO_URI=your_connection_string
JWT_SECRET=your_secret_key
```

FOLDER STRUCTURE

Client (React Frontend)

```
client/
├── public/              # Static assets
├── src/                 # Main source code
│   ├── components/      # Reusable UI components
│   ├── context/         # Global state management
│   ├── image/           # Images and icons
│   ├── pages/           # Application pages/screens
│   ├── styles/          # CSS files (App.css, etc.)
│   ├── App.js           # Root component
│   ├── index.js         # Entry point
│   ├── index.css        # Global styles
│   ├── App.test.js      # App tests
│   ├── reportWebVitals.js  # Performance metrics
│  └── setupTests.js     # Test configuration
│
├── .gitignore
├── package-lock.json
└── README.md
```

Server (node.js + express.js)

```
  server/
│  package.json
│  package-lock.json
│  index.js
│  server.js
│  .env
│
 ├– controllers/
 ├– models/
 ├– routes/
 ├– middleware/
└─ config/
```

RUNNING THE APPLICATION

Frontend

- cd client
- npm start

Backend

- cd server
- npm start

API DOCUMENTATION

User APIs

- **POST /api/auth/register** – Register a new user
- **POST /api/auth/login** – User login
- **GET /api/auth/profile** – Get logged-in user profile *(protected)*

Product APIs

- **GET /api/products** – Get all products
- **GET /api/products/:id** – Get a single product by ID
- **POST /api/products** – Add a product *(admin only)*
- **PUT /api/products/:id** – Update a product *(admin only)*
- **DELETE /api/products/:id** – Delete a product *(admin only)*

Cart APIs

- **POST /api/cart/add** – Add item to cart
- **GET /api/cart/:userId** – Get user cart
- **DELETE /api/cart/:itemId** – Remove item from cart

Order APIs

- **POST /api/orders** – Place a new order
- **GET /api/orders/:userId** – Get orders of a user
- **GET /api/orders/details/:orderId** – Get order details

Admin APIs

- **POST /api/admin/login** – Admin login
- **GET /api/admin/users** – Get all users
- **GET /api/admin/orders** – Get all orders

AUTHENTICATION

- ## JWT Generation:

  When a user logs in, the backend verifies the credentials and generates a JWT token.

- ## Token Storage:

  The JWT token is stored on the frontend using **localStorage** or **sessionStorage**.

- ## Protected Routes:

  For every protected API request, the token is included in the Authorization header as:

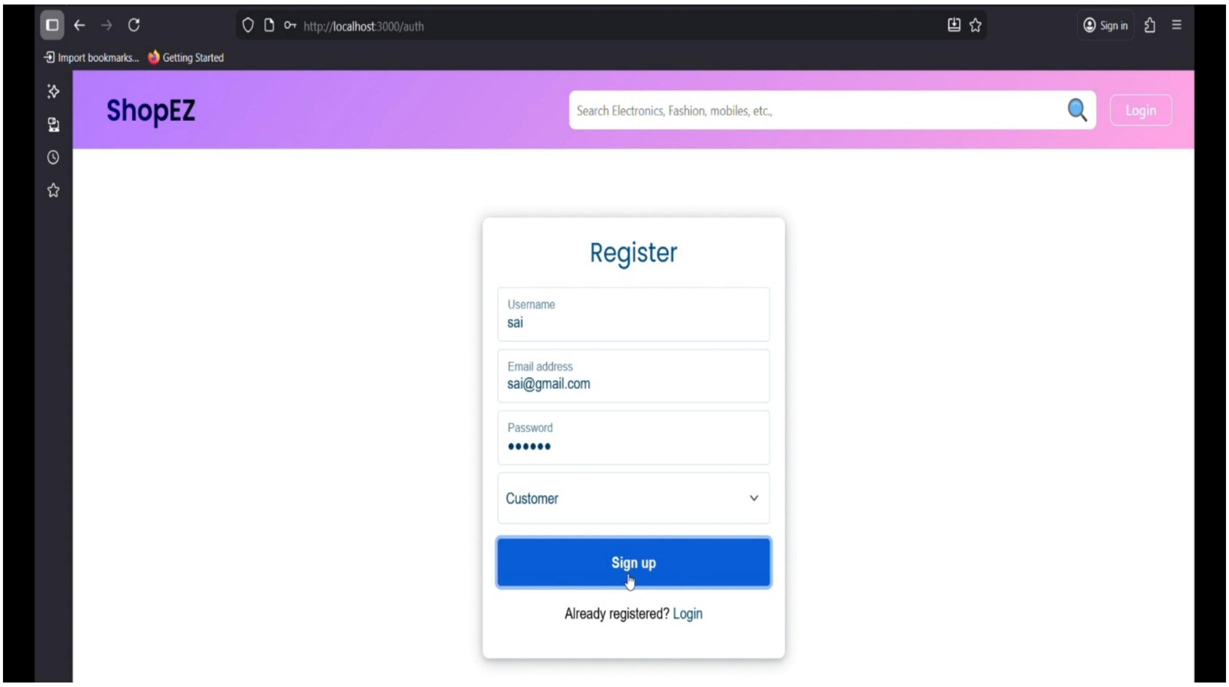
  ```
  Authorization: Bearer <token>
  ```

- ## Authorization:

  The backend validates the token before granting access to restricted routes such as user profile, cart, orders, and admin operations.

USER INTERFACE

Signup Page


http://localhost:3000/auth
Import bookmarks...
Getting Started
Sign in
ShopEZ
Search Electronics, Fashion, mobiles, etc.
Login

Register

Username
sai

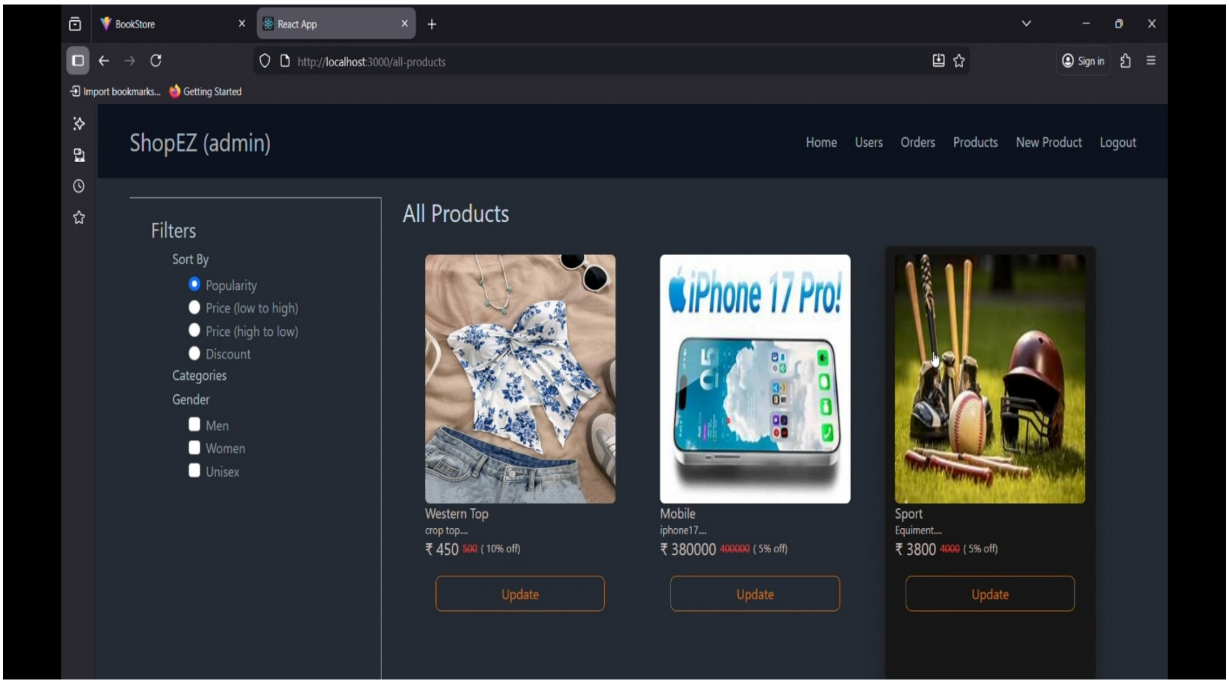
Email address
sai@gmail.com

Password
••••••

Customer

Already registered? Login

Product Listing Page



Filters

Sort By

● Popularity
○ Price (low to high)
○ Price (high to low)
○ Discount

Categories

Gender

□ Men
□ Women
□ Unisex

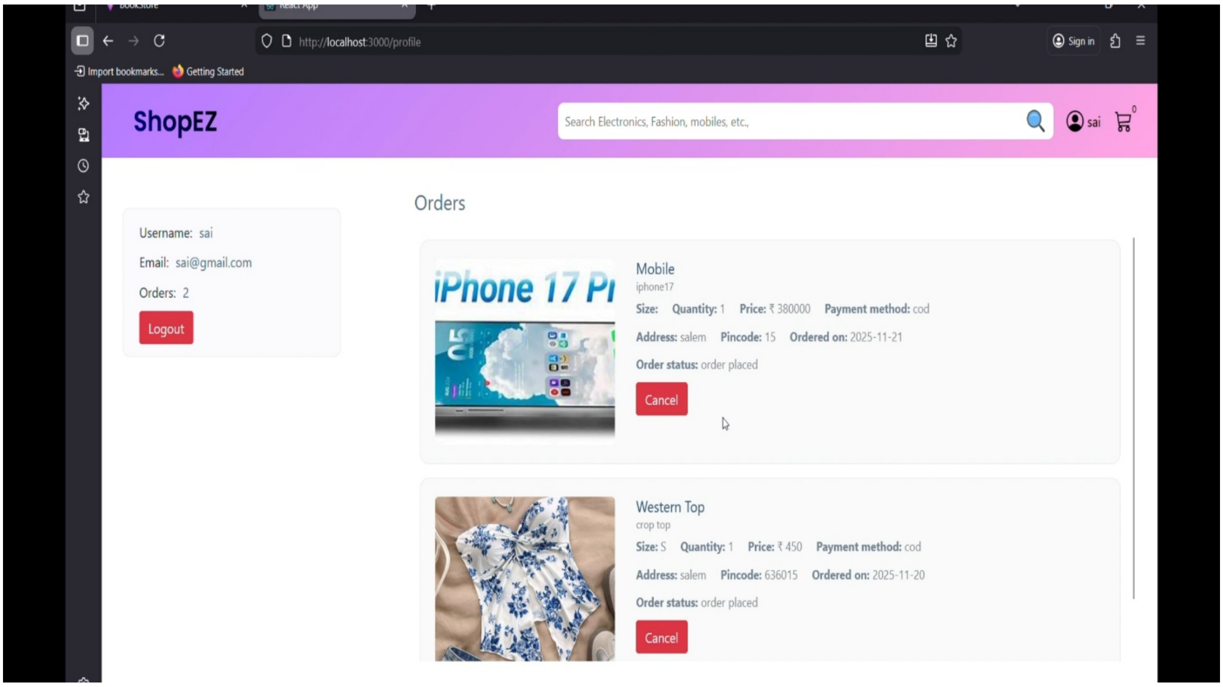
All Products

Western Top
crop top...
₹ 450 500 (10% off)

Mobile
iphone17...
₹ 380000 400000 (5% off)

Sport
Equiment...
₹ 3800 4000 (5% off)

Order Details Page



http://localhost:3000/profile
Import bookmarks... Getting Started
Sign in
ShopEZ
Search Electronics, Fashion, mobiles, etc.,
sai
0
iPhone 17 Pr

Username: sai

Email: sai@gmail.com

Orders: 2

Logout

Orders

Mobile
iphone17

Size: **Quantity:** 1 **Price:** ₹ 380000 **Payment method:** cod

Address: salem **Pincode:** 15 **Ordered on:** 2025-11-21

Order status: order placed

Cancel

Western Top
crop top

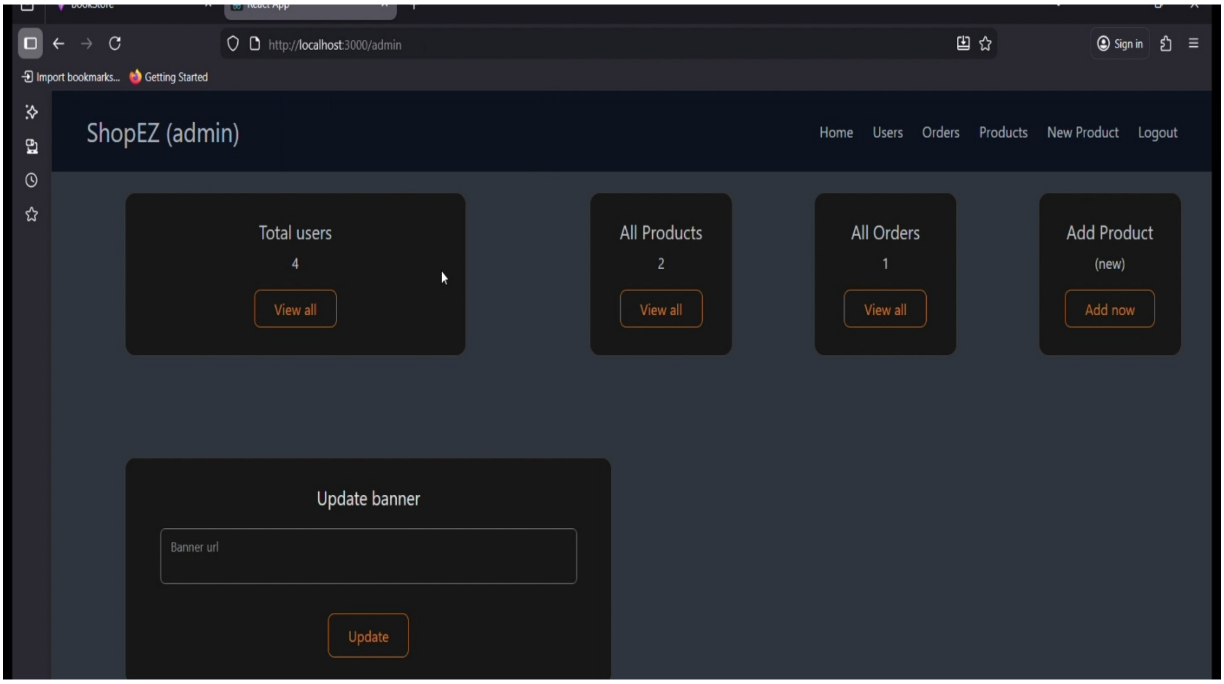
Size: S **Quantity:** 1 **Price:** ₹ 450 **Payment method:** cod

Address: salem **Pincode:** 636015 **Ordered on:** 2025-11-20

Order status: order placed

Cancel

Admin Dashboard



http://localhost:3000/admin
Import bookmarks... Getting Started
Sign in
ShopEZ (admin)
Home
Users
Orders
Products
New Product
Logout

Total users
4
View all

All Products
2
View all

All Orders
1
View all

Add Product
(new)
Add now

Update banner

Banner url

Update

TESTING

Testing Strategy

- **UI Testing:**
  Each page (Home, Products, Cart, Profile, Admin Dashboard) was manually tested to verify layout, navigation, and user interactions.

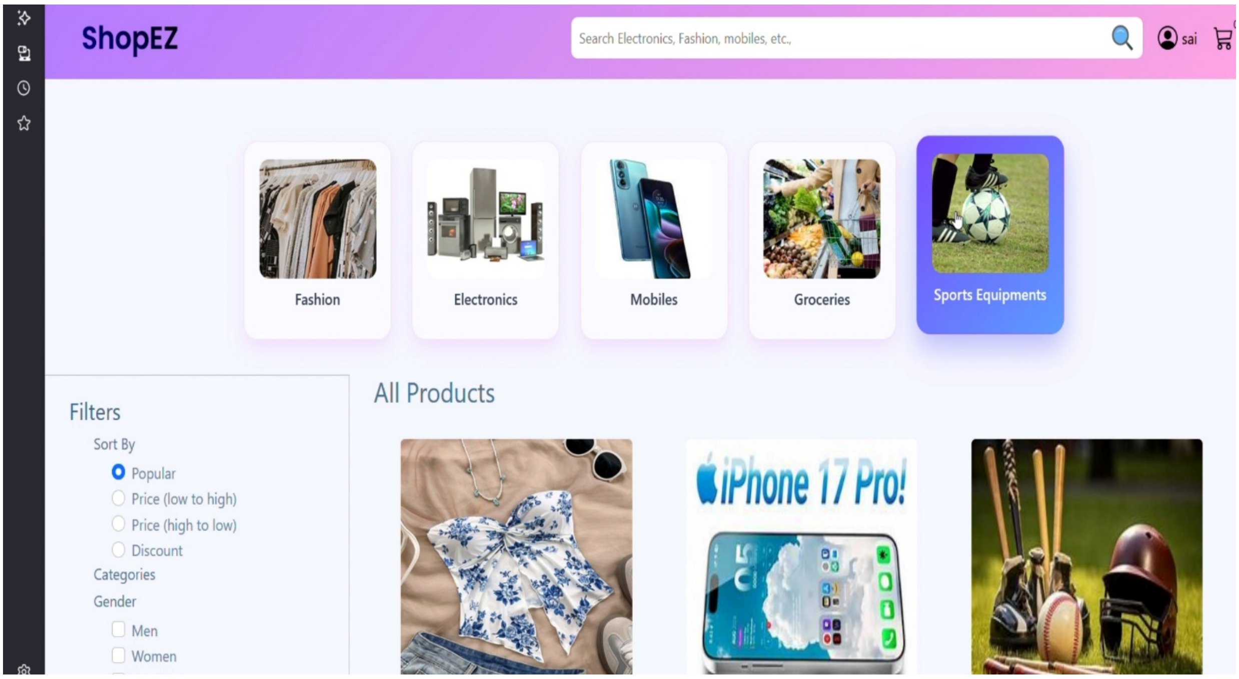
- **API Testing:**
  All backend API endpoints were tested using **Postman** to validate request methods, parameters, responses, and error handling.

Tools Used

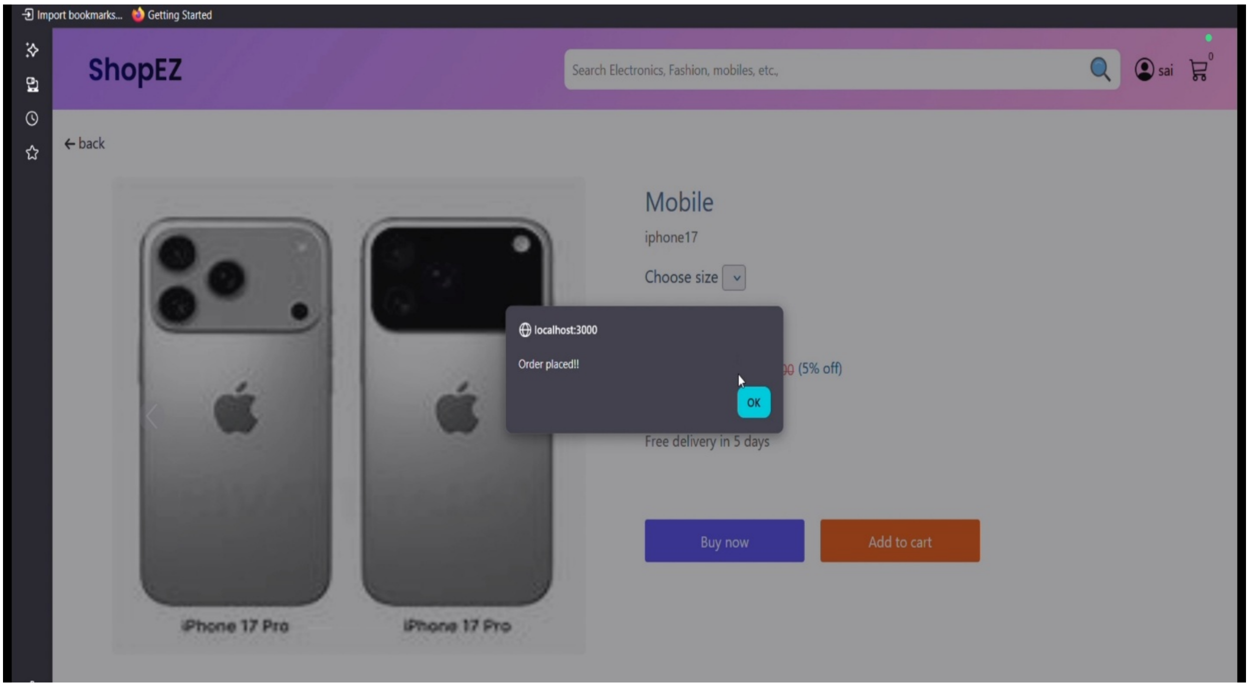
- **Manual UI Testing** (Browser-based testing)
- **Postman** for API endpoint testing
- **Console Logs & Debugging Tools** for resolving errors

SCREENSHOTS OR DEMO

Home Page



Order Confirmation Page


localhost:3000

Order placed!!

KNOWN ISSUES

- Some pages are not fully mobile responsive.
- API responses may be slow at times.
- Error messages are not always displayed clearly.
- Duplicate entries can occur if buttons are clicked repeatedly.
- Images may load slowly on weak internet.

FUTURE ENHANCEMENTS

- Add advanced search and filters for better user experience.
- Implement payment gateway integration for online payments.
- Add an admin analytics dashboard with charts and reports.
- Improve mobile responsiveness and overall UI design.
- Enable email/SMS notifications for orders and updates.
- Add wishlist, product reviews, and ratings.
- Implement role-based access control for enhanced security.