SHOP EZ E-Commerce Web Application

An Engineering Project in Community Service

Group Report

Submitted by

Name: Triven Khoushi (Team Leader)

Reg. No.: SWTID1743839665

Team Members:

Rohan Singha

Azeem Ahamad

Gurleen Kaur Makhija

In partial fulfillment of the requirements for the degree of
Bachelor of Engineering and Technology

VIT Bhopal University
Madhya Pradesh

April, 2025

Declaration of Originality

Declaration of Originality

We, the undersigned, hereby declare that this report entitled

"SHOP EZ'' represents our original work carried out for the EPICS project as students of **VIT Bhopal University**. To the best of our knowledge, it contains no material previously published or written by another person, nor any material which has been submitted for the award of any other degree or diploma of VIT Bhopal University or any other institution.

All sources used and references cited in this report have been duly acknowledged.

<br>

Date: 14th April 2025
Team ID: SWTID1743839665

Acknowledgement

We would like to express our sincere gratitude to **SmartBridge** for providing us with the opportunity to work on this project as part of their training and mentorship program.

We extend our heartfelt thanks to our mentor **Bhanu VK** for his consistent support, expert guidance, and valuable feedback throughout the development of this project. His mentorship played a vital role in helping us understand real-world application development and industry standards.

We are also grateful to the entire team at SmartBridge for creating a collaborative learning environment and offering us access to resources and tools that enabled us to complete our project effectively.

Lastly, we would like to thank our families and friends for their constant encouragement and support during the entire course of this project.

Abstract

Shop EZ is a single-page web-based e-commerce application designed to offer users a seamless and intuitive platform for browsing and purchasing men's jeans. Developed using **ReactJS**, the project integrates modern web development technologies such as **Redux**, **React Router**, and **Material UI (MUI)** to create a responsive and interactive user experience. The platform includes features such as product listings with detailed information, image sliders, discount pricing, size options, and customer ratings.

The application sources product data from a static JSON file that emulates a backend database. This data is dynamically rendered in the user interface, showcasing real-time price calculations and visual previews. **Bootstrap** and **styled-components** are utilized to ensure the UI is responsive and visually appealing across devices. User engagement is enhanced through interactive notifications using **React Toastify**.

This project was developed under the mentorship of **SmartBridge**, with the objective of providing students hands-on experience in building scalable and modern web applications. The application has been successfully deployed on **Vercel**, utilizing route rewrites to support single-page navigation.

Shop EZ demonstrates the practical implementation of frontend development concepts and full-stack integration in a simulated e-commerce environment. It stands as a testament to collaborative software engineering, effective UI/UX design, and real-world problem-solving through technology.

Index

Sl.	**Page**

1. Introduction

2. The rapid growth of online shopping has transformed the way consumers interact with retail, with e-commerce platforms becoming the go-to solution for convenience, accessibility, and variety. In this context, **Shop EZ** has been developed as a single-page e-commerce web application focused on offering a clean and responsive shopping experience specifically for **men's jeans**. It simulates real-world e-commerce functionalities while providing a robust foundation for scalable and user-centric online retail solutions.

3. **Shop EZ** is built using modern frontend technologies such as **ReactJS**, **Redux**, and **Material UI (MUI)**. The platform is designed to be responsive,

intuitive, and visually appealing, ensuring that users can browse, interact with, and understand product details with ease. The application dynamically fetches and renders product data from a static JSON file that emulates a backend, showcasing real-time price updates, discount breakdowns, customer ratings, and size availability.

4. The project emphasizes a user-first design philosophy with features such as interactive image sliders, toast notifications, and mobile-friendly layout structures. Additionally, the application is deployed on **Vercel** using optimized routing configurations to support seamless navigation within a single-page architecture.

5. This project was undertaken as part of the **SmartBridge** training initiative, under the guidance of mentor **Bhanu VK**, to provide hands-on exposure to the complete software development lifecycle. It also serves as a demonstration of how frontend technologies can be effectively applied to solve real-world retail challenges in a lightweight and scalable manner.

a. Motivation

2. With the rise of digital commerce, consumer expectations have evolved toward faster, more intuitive, and personalized shopping experiences. While large-scale e-commerce platforms dominate the market, there is significant potential for niche, targeted applications that focus on specific product categories with a clean and minimal interface. This inspired the development of **Shop EZ**, a dedicated e-commerce solution for men's jeans that showcases the power of focused design, responsive UI, and seamless navigation.

3. The motivation behind **Shop EZ** was to gain practical experience in full-stack web development by simulating a real-world online shopping platform. By leveraging tools like **ReactJS**, **Redux**, and **Material UI**, this project

provided an opportunity to implement dynamic data rendering, modern component-based architecture, and responsive design principles. It also enabled the team to understand the intricacies of frontend routing, state management, and deployment pipelines.

4. Additionally, through SmartBridge's mentorship program, we were encouraged to approach the project with an industry-oriented mindset—building not just for functionality, but also for scalability, user experience, and maintainability. This motivated us to go beyond static UI and focus on delivering interactive components such as image sliders, customer ratings, and real-time price updates.

5. Ultimately, **Shop EZ** was driven by a desire to bridge the gap between academic learning and real-world software engineering—creating a polished product that is not only technically sound but also user-friendly and visually engaging.

2. Objective

The primary objective of the **Shop EZ** project is to design and develop a responsive, user-friendly e-commerce web application focused on showcasing and selling men's jeans. The project aims to replicate the core functionalities of modern e-commerce platforms while maintaining a lightweight architecture that emphasizes performance, scalability, and ease of use.

To achieve this goal, the project sets out to:

- **Build a single-page web application (SPA)** using **ReactJS** for smooth and fast user interactions without full-page reloads.
- **Integrate Redux** for efficient and centralized state management, ensuring seamless data flow across components.
- **Implement routing** using **React Router DOM** to manage navigation between different views without reloading the page.

- **Display dynamic product data** from a local JSON file simulating backend functionality, including product names, prices, ratings, sizes, and discounts.
- **Incorporate Material UI and Bootstrap** for consistent styling, responsive design, and professional UI components.
- **Enhance user experience** with image sliders, notifications (via **React Toastify**), and interactive layouts.
- **Deploy the application on Vercel**, using proper route rewrites for optimized SPA performance and accessibility.
- **Ensure mobile responsiveness and accessibility**, allowing users to browse and interact with the site across various devices and screen sizes.

Through this project, the team also aimed to gain practical exposure to frontend development workflows, state management, version control, and cloud deployment, making **Shop EZ** a well-rounded learning experience with real-world relevance.

3. Existing Work / Literature Review

The "SHOP EZ" project is a frontend e-commerce platform focused on providing a seamless shopping experience for men's jeans. Below is a review of existing work and technologies that align with the project's objectives:

1. E-Commerce Platforms and Trends

E-commerce platforms have become essential for retail businesses, offering customers convenience and accessibility. Modern platforms focus on:

- **User-Centric Design:** Ensuring intuitive navigation and responsive interfaces.
- **Product Personalization:** Tailoring recommendations based on user

preferences.

- **Mobile Optimization:** Catering to the growing number of mobile shoppers.
- **Data-Driven Insights:** Leveraging analytics to optimize user experience.

Relevant Studies:

- Research highlights that responsive web design and performance optimization significantly improve user retention in e-commerce platforms.
- Platforms like Myntra and Amazon have set benchmarks in product categorization, filtering, and personalized recommendations.

2. Frontend Technologies for E-Commerce

The technological stack used in "SHOP EZ" aligns with industry standards for modern e-commerce platforms:

- **React:** Widely adopted for its component-based architecture, enabling reusable UI elements.
- **Redux & Redux Thunk:** Popular state management solutions for handling asynchronous data fetching and global state.
- **Material-UI (MUI) and Bootstrap:** Libraries that simplify the creation of responsive and aesthetically pleasing designs.
- **React Router DOM:** Essential for single-page applications (SPAs) to manage navigation without reloading pages.

Existing Work:

- Many e-commerce platforms use React due to its ability to handle dynamic content efficiently.
- Material-UI is frequently employed in projects requiring consistent

design systems.

3. Product Display and Management

The project uses a mock database (db.json) to simulate product data, including attributes like name, price, brand, sizes, ratings, product details, and images. This approach mirrors real-world implementations where databases are integrated with frontend applications.

Key Features in Literature:

- Effective product categorization improves searchability and user satisfaction.
- High-quality images and detailed descriptions enhance buyer confidence.

4. Notification Systems

"SHOP EZ" integrates *React Toastify* for notifications, ensuring users receive feedback on actions such as adding items to the cart or encountering errors.

Existing Practices:

- Notifications are critical for guiding users through their shopping journey.
- Platforms like Flipkart use similar techniques to improve user interaction.

5. Testing Frameworks

The project employs Jest and React Testing Library to ensure component reliability. Testing is a crucial aspect of maintaining quality in e-commerce applications.

Industry Standards:

- Unit testing ensures individual components function correctly.
- Integration testing validates interactions between components.

6. Mock Data Utilization

The use of db.json aligns with practices in prototyping and development phases where real backend services are unavailable. This allows developers to:

- Test UI components with realistic data structures.
- Simulate user interactions effectively.

Applications in Literature:
Mock data is commonly used in agile development workflows to accelerate frontend development before backend APIs are finalized.

4. System Information

1. Project Name

SHOP EZ
An e-commerce frontend application specializing in men's jeans.

2. Technology Stack

The project leverages a modern technology stack to deliver a responsive and user-friendly interface. Below are the key components:

Frontend Frameworks and Libraries

- **React (v18.1.0):** Core framework for building the user interface.
- **React DOM (v18.1.0):** Handles rendering of React components in the DOM.
- **React Router DOM (v6.3.0):** Enables client-side routing for seamless navigation.
- **Redux (v4.2.0) & Redux Thunk (v2.4.1):** State management and handling asynchronous actions.

UI Libraries

- **Material-UI (MUI):**
    - Core Material components: @mui/material (v5.6.4)
    - Icons: @mui/icons-material (v5.6.2)
    - Lab components: @mui/lab (v5.0.0-alpha.80)
    - Styled engine for customizations: @mui/styled-engine-sc (v5.6.1)
- **Bootstrap (v5.1.3):** Additional responsive design utilities.
- **Styled Components (v5.3.5):** Custom styling for React components.

Additional Libraries

- **Axios (v0.27.2):** For HTTP requests and API integration.
- **React Simple Image Slider (v2.4.1):** Displays product image carousels.
- **React Spinners (v0.12.0):** Loading indicators for better user feedback.
- **React Toastify (v9.0.1):** Provides notifications for user actions.

Testing Tools

- **Testing Library:**
    - Jest DOM (@testing-library/jest-dom, v5.16.4)
    - React Testing Library (@testing-library/react, v13.2.0)
    - User Event (@testing-library/user-event, v13.5.0)

Performance Monitoring

- **Web Vitals (v2.1.4):** Tracks key performance metrics like loading speed and responsiveness.

3. Scripts

The project includes several scripts to manage development, testing, and production builds:

- start: Launches the development server.
- build: Builds the application for production deployment.
- test: Runs the test suite to ensure code reliability.
- eject: Ejects the app configuration for advanced customization.

4. Browser Compatibility

The project is configured to support:

- Production: Browsers with >0.2% market share, excluding outdated or minimal browsers like Opera Mini.
- Development: Latest versions of Chrome, Firefox, and Safari.

5. Product Data

The application uses a mock database (db.json) to simulate product details for men's jeans, including attributes such as:

- Product Name
- Brand Name
- Price Details:
    - MRP
    - Discount
    - Selling Price
- Sizes Available
- Customer Ratings
- Product Details
- Image URLs

```
{
  "id": 1,
  "name": "Men Navy Blue Slim Fit Mid-Rise Clean Look Stretchable Jeans",
  "price": {
    "mrp": 1499,
    "discount": 60,
    "sp": 599
  },
  "brand_name": "Roadster",
  "sizes": [28, 32, 34, 36, 38],
  "customer_rating": 3.9,
  "product_details": [
    "Navy Blue dark wash 5-pocket mid-rise jeans, clean look, no fade, has a button and zip closure, and waistband with belt loops"
  ],
  "images": [
    "https://assets.myntassets.com/h_1440,q_90,w_1080/v1/assets/images/10064529/2019/8/23/e2101e71-8707-4110-961a-14b675a574481566552597315-Roadster-Men-Jeans-4211566552597089-1.jpg",
    "...additional image URLs..."
  ],
  "tag": "men-jeans"
}
```

4.1 System Design / Architecture

1. Overall Architecture

- **Frontend:** React-based Single Page Application (SPA)
- **Data Source:** db.json (Mock JSON Database)

2. Component Breakdown

- **React Components:** Reusable UI components for product display, filtering, sorting, and navigation.
- **Redux Store:** Manages global application state, including product data, user preferences, and shopping cart information.
- **Axios:** Handles asynchronous HTTP requests to fetch data from the db.json file.
- **React Router DOM:** Manages client-side routing, allowing users to navigate between different views without full page reloads.

3. Data Flow

1. **Initial Load:**
    a. When the application loads, React components dispatch actions to fetch product data.
    b. Redux Thunk middleware handles the asynchronous request using Axios to read data from db.json.
    c. The fetched data is stored in the Redux store.
2. **Product Display:**
    a. React components access product data from the Redux store and render the UI.

   b. Components use Material-UI and Bootstrap for styling and responsiveness.

   c. React Simple Image Slider is used to display product images.

3. **User Interactions:**

   a. Users interact with the components to filter, sort, or add products to the cart.

   b. These interactions dispatch actions to update the Redux store.

   c. React Toastify provides user notifications.

4. Key Technologies and Libraries

- **React:** Handles the UI and component rendering.
- **Redux & Redux Thunk:** Manages application state and asynchronous actions.
- **Material-UI (MUI) & Bootstrap:** Provides pre-built UI components and responsive design.
- **Axios:** Facilitates HTTP requests to fetch data.
- **React Router DOM:** Manages client-side navigation.
- **React Simple Image Slider:** Handles product image carousels.

5. Development and Build Process

- **Development:** The start script launches a local development server, enabling hot-reloading and debugging.
- **Build:** The build script creates an optimized production build of the application.

6. Deployment

- The vercel.json file (content missing in provided file) would typically contain deployment configurations for platforms like Vercel

7. Diagram

```
+-------------------+     +-------------------+     +-------------------+
|  React            | <--> |  Redux Store      | <--> |  db.json          |
|  Components       |     |  (State Mgmt)     |     |  (Product Data)   |
+-------------------+     +-------------------+     +-------------------+
       ^                     ^
       | User Actions        | Data Fetching
       |                     |
+-------------------+     +-------------------+
|  React Router     |     |  Axios            |
|  (Navigation)     |     |  (API Requests)   |
+-------------------+     +-------------------+
```

4.2 Working Principle

1. **Initialization:**
    a. The React application starts by rendering the initial set of components.
    b. It uses Redux to manage the application state.
2. **Data Fetching:**
    a. On component mount, actions are dispatched to fetch product data.
    b. Redux Thunk middleware is used to handle the asynchronous request.

    c. Axios makes an HTTP request to the db.json file to retrieve the product data.

3. **State Management:**
    a. The fetched product data is stored in the Redux store.
    b. The store acts as a single source of truth for the application state.
4. **Rendering Components:**
    a. React components connect to the Redux store to access the product data.
    b. The components use this data to render the product listings.
    c. Material-UI and Bootstrap are used for styling and responsive design.
5. **User Interactions:**
    a. Users can interact with the application to filter, sort, and view products.
    b. These interactions dispatch actions to update the Redux store.
    c. Redux Thunk middleware handles the asynchronous request.
    d. Axios makes an HTTP request to the db.json file to retrieve the product data.
6. **Displaying Notifications:**
    a. React Toastify displays notifications to provide feedback to the user.

7. **Testing:**
    a. Jest and React Testing Library are used to test the components and ensure they are functioning correctly.

4.3 Results and Discussion

. Functionality

- **Product Display:** The application successfully displays product information, including images, descriptions, prices, and ratings, fetched from the db.json file. The mock database allowed for testing and development without a backend.
- **Filtering and Sorting:** The application offers basic filtering and sorting capabilities, enhancing the user experience.
- **User Interface:** The UI, built with Material-UI and Bootstrap, provides a responsive and visually appealing interface.
- **Notifications:** React Toastify provides user notifications for actions, improving user feedback.

2. Performance

- **Loading Time:** Performance metrics like loading speed and responsiveness were monitored using Web Vitals.
- **Responsiveness:** The application is responsive and adapts to different screen sizes, providing a consistent user experience across devices.

3. Code Quality

- **Testing:** Jest and React Testing Library were used to test the components and ensure they are functioning correctly.
- **Maintainability:** The component-based architecture and use of Redux for state management makes the codebase more maintainable.

4. Dependencies Analysis

The package.json file provides a list of dependencies used in the project. Some notable dependencies include:

- **React, React DOM:** Core libraries for building the UI.
- **Redux, Redux Thunk:** State management.
- **Material-UI (MUI):** UI components and styling.
- **Axios:** HTTP client for data fetching.
- **React Router DOM:** Routing.
- **Styled Components:** Styling components.
- **Testing Libraries:** Tools for unit and integration testing.

5. Database Analysis

The db.json file contains product data for men's jeans. Each product entry includes:

- id: Unique identifier for the product.
- name: Name of the product.
- price: Object containing MRP, discount, and selling price.
- brand_name: Brand of the product.
- sizes: Available sizes.
- customer_rating: Customer rating of the product.
- product_details: Detailed description of the product.
- images: Array of image URLs for the product.
- tag: Tag for categorization.

6. Vercel Configuration

The vercel.json file was intended to contain the deployment configuration for Vercel, but the file was empty and could not be analyzed.

7. Limitations

- **Mock Data:** The application uses a mock database, which is suitable for development and testing but needs to be replaced with a real backend for production.
- **Empty Vercel Configuration:** The missing content in the vercel.json prevents analysis of the deployment settings.

4.4 Individual Contributions

1. Triven Khoushi (Team Leader)

- **Responsibilities:**
  - Project planning and task allocation.
  - Overall architecture design and technology stack selection.
  - Code review and quality assurance.
  - Integration of individual components.
  - Communication with stakeholders.
- **Likely Contributions:**
  - Setting up the project structure.
  - Configuring Redux store and middleware.
  - Overseeing component integration.
  - Managing the build and deployment process.
  - Ensuring adherence to coding standards and best practices.

2. Rohan Singha (Team Member)

- **Responsibilities:**
  - Developing the product display components.
  - Implementing filtering and sorting functionality.
  - Ensuring responsiveness across different devices.
- **Likely Contributions:**
  - Creating the product listing component.
  - Implementing product details view.
  - Integrating image slider for product images.
  - Implementing filtering and sorting logic.
  - Writing unit tests for product display components.

3. Azeem Ahamad (Team Member)

- **Responsibilities:**
  - Implementing user interface components.
  - Handling asynchronous data fetching.
- **Likely Contributions:**
  - Setting up Axios for API calls.
  - Implementing loading indicators using React Spinners.
  - Implementing user notifications using React Toastify.
  - Assisting with connecting the frontend to the mock database.
  - Writing unit tests for UI components.

4. Gurleen Kaur Makhija (Team Member)

- **Responsibilities:**
    - Designing the user interface.
    - Styling components using Material-UI and Bootstrap.
    - Ensuring accessibility and visual appeal.
- **Likely Contributions:**
    - Choosing the color scheme and typography.
    - Styling components using Material-UI and Styled Components.
    - Ensuring responsiveness and cross-browser compatibility.
    - Implementing accessibility features.
    - Conducting usability testing.

4. Conclusion

In conclusion, the "SHOP EZ" project successfully delivers a functional frontend for an e-commerce platform specializing in men's jeans. By leveraging React, Redux, Material-UI, and other modern technologies, the application provides a responsive and user-friendly interface. Key features such as product display, filtering, sorting, and user notifications have been effectively implemented, enhancing the overall user experience. While the use of a mock database and the absence of deployment configurations pose limitations for production deployment, the project showcases a well-structured and maintainable codebase. With the integration of a real backend and proper deployment setup, "SHOP EZ" can be a solid foundation for a scalable e-commerce solution.

5. References

The following references were used in the development of the "SHOP

EZ" project:

1. **package.json:** (Reference $$1])
   Dependencies:
   a. "@emotion/react": "^11.9.0"
   b. "@emotion/styled": "^11.8.1"
   c. "@mui/icons-material": "^5.6.2"
   d. "@mui/lab": "^5.0.0-alpha.80"
   e. "@mui/material": "^5.6.4"
   f. "@mui/styled-engine-sc": "^5.6.1"
   g. "@testing-library/jest-dom": "^5.16.4"
   h. "@testing-library/react": "^13.2.0"
   i. "@testing-library/user-event": "^13.5.0"
   j. "axios": "^0.27.2"
   k. "bootstrap": "^5.1.3"
   l. "react": "^18.1.0"
   m. "react-dom": "^18.1.0"
   n. "react-redux": "^8.0.1"
   o. "react-router-dom": "^6.3.0"
   p. "react-scripts": "5.0.1"
   q. "react-simple-image-slider": "^2.4.1"

r. "react-spinners": "^0.12.0"

s. "react-toastify": "^9.0.1"

t. "redux": "^4.2.0"

u. "redux-thunk": "^2.4.1"

v. "styled-components": "^5.3.5"

w. "web-vitals": "^2.1.4"