Project Documentation – ShoppyGlobe E-Commerce Application

# GitHub Link :-

https://github.com/noubahar123/ShoppyGlobe.git

# 1. Introduction

ShoppyGlobe is a demo e-commerce web application built using React (with Vite), Redux Toolkit, React Router v6, and Tailwind CSS. The purpose of this project is to demonstrate key concepts of front-end development such as component-based design, state management, API integration, routing, event handling, and styling.

# 2. How to Run

1. Clone or extract the project.  
2. Install dependencies: npm install, npm install -D tailwindcss postcss autoprefixer,

npx tailwindcss init -p, npm install react-router-dom@6npm install @reduxjs/toolkit react-redux     
3. Start dev server: npm run dev  
4. Open http://localhost:5173

# 3. Objectives

- Build a modular, maintainable React application.  
- Fetch and display product data from an external API (DummyJSON).  
- Implement search and category filtering features.  
- Manage cart state globally using Redux Toolkit.  
- Use React Router for navigation with lazy-loaded routes.  
- Style the entire application with Tailwind CSS.  
- Provide an interactive shopping cart and checkout flow.

# 4. Tools and Technologies

• React 18  
• Vite  
• Redux Toolkit + React-Redux  
• React Router v6  
• Tailwind CSS  
• DummyJSON API

# 5. Project Structure

shoppyglobe/  
├─ public/  
├─ src/  
│ ├─ components/  
│ ├─ hooks/  
│ ├─ pages/  
│ ├─ utils/  
│ ├─ App.jsx  
│ ├─ main.jsx  
│ └─ index.css  
├─ index.html  
├─ package.json  
├─ vite.config.js  
├─ tailwind.config.js  
└─ README.md

# 6. Features Implemented

1. Component Structure: App, Header, Footer, ProductItem, SearchBar, CategoryPills, etc.

2. Pages: Home, Categories, CategoryView, ProductDetail, Cart, Checkout, NotFound.

3. State Management: Redux Toolkit cartSlice with add, remove, increment, decrement, clear actions.

4. API Integration: useProducts.js hook, product detail fetch, category fetch.

5. Event Handling: Add to cart, increment/decrement/remove items, search input, category pills.

6. Lists: Product grid, cart items, categories list.

7. Routing: Home, Categories, CategoryView, ProductDetail, Cart, Checkout, NotFound.

8. Search: Filters products by title, brand, or category.

9. Styling: Tailwind CSS used throughout with custom utility classes.

10. Performance: Lazy loaded routes with React.lazy and Suspense.

# 7. Future Enhancements

- Cart persistence with localStorage.  
- User authentication.  
- Pagination for products.  
- Payment integration.  
- Dark mode support.

# 8. Conclusion

This project demonstrates the use of React, Redux Toolkit, Tailwind CSS, and API integration to build a functional e-commerce application. It follows best practices like component reusability, clean folder structure, lazy loading, and memoized selectors. It can be extended into a complete platform with authentication, persistent cart, and payments.