High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components for **ShopEase - Product Listing Page**. The purpose of this project is to deliver a simple, responsive e-commerce product listing page with filtering, sorting, and a grid layout for browsing products, using modern frontend technologies.

---

1. System Architecture Overview

Architecture Description:
The system is a single-page React application structured into modular components. It fetches product data (mocked or via API), manages UI state for filtering and sorting, and displays products in a responsive grid.

Module/Component	Role/Responsibility
App Root	Initializes app, manages global state
ProductList	Displays products in a grid, receives filtered/sorted data
FilterBar	UI for category, price, and other filters
SortBar	UI for sorting options (e.g., price, popularity)
ProductCard	Renders individual product details
Data Source	Provides product data (mocked JSON or API)

---

2. Component Interactions

Source Component	Target Component	Interaction Description
App Root	Data Source	Fetches product data on load
FilterBar	App Root	Sends filter criteria updates
SortBar	App Root	Sends sorting option updates
App Root	ProductList	Passes filtered and sorted product data
ProductList	ProductCard	Renders each product as a card

Sequence Flow:

1. App Root fetches product data.
2. User interacts with FilterBar/SortBar.
3. App Root updates state and passes filtered/sorted data to ProductList.
4. ProductList renders ProductCards.

---

3. Data Flow Overview

Data Source	Consumed By	Data/Action Description
Product Data	App Root	Initial data fetch (mocked or API)
Filter Criteria	App Root	Updates state on user input
Sort Option	App Root	Updates state on user input
Filtered/Sorted Data	ProductList	Receives processed product list for display

---

4. Technology Stack

Layer/Area	Technology/Framework
UI Framework	React
Styling	Tailwind CSS, CSS
Language	JavaScript, HTML
Data Handling	Local state, (optional) fetch API
Build Tooling	Vite or Create React App

---

5. Scalability & Reliability

- **Scalability:**
  - Component-based design enables easy extension (e.g., adding pagination or more filters).
  - Stateless UI components facilitate reuse and testing.
- **Reliability:**
  - Data fetching and state management are isolated in the App Root.
  - Responsive design ensures usability across devices.
- **Security:**
  - No sensitive data handled; standard React security practices apply.

End of Document