SYSTEM DESCRIPTION

Overall Purpose and Functionality
This system is a modern, multi-page e-commerce web application designed to provide users with a sea

Main Components and Their Responsibilities
- App.js: The root component orchestrates routing between pages, integrates global UI elements (NavB
- NavBar.js: Provides site-wide navigation, displays cart status, and links to key sections.
- Body.js: Handles the main product listing, category filtering, and product selection logic.
- Cart.js: Manages the shopping cart, displaying selected items, handling quantity changes, and initiatin
- Product Detail Page: Displays detailed product information, reviews, and includes a search/filter for re
- Specialized Pages (Under $20, Under $40, For Him, For Her): Filter and display products based on pr
- JournalSection.js: Presents blog or journal content related to products or lifestyle.
- Redux State (CartState.js): Centralized state management for cart operations (add, remove, update it
- Product Store (Store.js): Static product catalog used for rendering product lists and details.

Key Technologies and Frameworks Used
- React: Core UI library for building interactive components and managing application state.
- Redux Toolkit: State management, primarily for cart functionality.
- React Router: Client-side routing for multi-page navigation.
- Chakra UI, Material UI, MUI: Component libraries for consistent, responsive design and UI elements.
- @google-pay/button-react, @paypal/react-paypal-js: Integrations for Google Pay and PayPal paymen
- Emotion: CSS-in-JS styling for custom component styles.
- React Icons: Iconography for UI elements.
- React Hover Image: Enhanced product image interactions.

Data Flow Between Components
- Product data is sourced from a static store (Store.js) and passed to listing components (Body, speciali
- User interactions (add to cart, filter products, search reviews) trigger state updates via Redux actions.
- Cart state is accessed and manipulated by Cart.js and NavBar.js, ensuring cart status is reflected acro
- Checkout flow collects cart data and passes it to integrated payment components (Google Pay, PayPa
- Routing (via React Router) ensures seamless navigation between pages, with relevant data passed a

External Dependencies and Integrations
- Payment Gateways: Google Pay and PayPal integrations via official React components; support for V
- UI Libraries: Chakra UI, Material UI, MUI for design consistency and accessibility.
- Static Assets: Product images and icons loaded from external URLs and local assets.
- No backend/API integration is evident; product data and cart state are managed client-side.

Security Considerations
- Payment Security: Reliance on official payment gateway components (Google Pay, PayPal) ensures P
- Data Privacy: As the system appears to be client-side only, sensitive user data (e.g., payment details)
- State Management: Cart and product data are managed in Redux; no evidence of persistent storage o
- External Resources: Images and assets loaded from external sources should be validated to prevent i
- No explicit authentication or authorization mechanisms are present; if user accounts or order history a

In summary, this is a React-based, client-side e-commerce platform with robust UI, dynamic product filt

PLANTUML DIAGRAM

```
@startuml
!include https://raw.githubusercontent.com/plantuml-stdlib/C4-PlantUML/master/C4_Container.puml

' System Context
```

```
Person(user, "User", "A shopper browsing and purchasing products.")

System_Boundary(ecommerce, "E-Commerce Web Application") {

  Container(appjs, "App.js", "React Component", "Root component orchestrating routing, layout, and glo
  Container(navbar, "NavBar.js", "React Component", "Site-wide navigation, cart status, and links to key
  Container(body, "Body.js", "React Component", "Main product listing, category filtering, and product se
  Container(cart, "Cart.js", "React Component", "Manages shopping cart, item quantities, and initiates ch
  Container(productdetail, "Product Detail Page", "React Component", "Displays product details, reviews
  Container(under20, "Under $20 Page", "React Component", "Displays products filtered by price (<$20
  Container(under40, "Under $40 Page", "React Component", "Displays products filtered by price (<$40
  Container(forhim, "For Him Page", "React Component", "Displays products filtered for male shoppers.
  Container(forher, "For Her Page", "React Component", "Displays products filtered for female shoppers
  Container(journal, "JournalSection.js", "React Component", "Displays blog/journal content related to p
  ContainerRedux(cartstate, "CartState.js", "Redux State", "Centralized cart state management (add, re
  Container(store, "Store.js", "Static Product Store", "Static product catalog for product lists and details."
  Container(payment, "Payment Components", "React Integrations", "Google Pay, PayPal, Visa, Amex,
}

' External Dependencies
Container_Ext(ui_libs, "UI Libraries", "Chakra UI, Material UI, MUI", "Component libraries for consistent
Container_Ext(icons, "Icon Libraries", "React Icons", "Iconography for UI elements.")
Container_Ext(assets, "Static Assets", "External URLs & Local", "Product images and icons loaded fron

' Relationships (Data Flow)
Rel(user, appjs, "Browses and interacts with")
Rel(appjs, navbar, "Routes navigation and cart status")
Rel(appjs, body, "Routes to product listing")
Rel(appjs, cart, "Routes to cart and checkout")
Rel(appjs, productdetail, "Routes to product detail page")
Rel(appjs, under20, "Routes to Under $20 page")
Rel(appjs, under40, "Routes to Under $40 page")
Rel(appjs, forhim, "Routes to For Him page")
Rel(appjs, forher, "Routes to For Her page")
Rel(appjs, journal, "Routes to Journal section")

Rel(body, store, "Reads product data from")
Rel(under20, store, "Reads filtered product data from")
Rel(under40, store, "Reads filtered product data from")
Rel(forhim, store, "Reads filtered product data from")
Rel(forher, store, "Reads filtered product data from")
Rel(productdetail, store, "Reads product details from")

Rel(cart, cartstate, "Reads and updates cart state via Redux")
Rel(navbar, cartstate, "Reads cart status from Redux")
Rel(cart, payment, "Passes cart data for checkout")
Rel(payment, assets, "Loads payment icons/assets from")
Rel(navbar, icons, "Loads icons from")
Rel(appjs, ui_libs, "Uses UI components from")
Rel(body, ui_libs, "Uses UI components from")
Rel(cart, ui_libs, "Uses UI components from")
Rel(productdetail, ui_libs, "Uses UI components from")
Rel(journal, ui_libs, "Uses UI components from")
Rel(appjs, assets, "Loads images and assets from")
Rel(body, assets, "Loads product images from")
Rel(productdetail, assets, "Loads product images from")
Rel(journal, assets, "Loads images from")

@enduml
```

THREAT MODELING

```
[
  {
    "asset": "Payment Data (Cart contents, payment details)",
    "threat": "Spoofing - Attacker impersonates user to initiate fraudulent payments",
    "severity": "High",
    "countermeasure": "Enforce secure payment gateway integrations, use HTTPS everywhere, and con
  },
  {
    "asset": "Cart State (Redux)",
    "threat": "Tampering - Manipulation of cart data in client-side state (e.g., price, quantity)",
    "severity": "Medium",
    "countermeasure": "Validate cart contents and pricing server-side during payment processing; use si
  },
  {
    "asset": "Product Store (Store.js)",
    "threat": "Tampering - Modification of product data (e.g., price, description) via client-side attacks",
    "severity": "Medium",
    "countermeasure": "Move product data to a backend API with server-side validation; restrict client-sid
  },
  {
    "asset": "User Session (Browser)",
    "threat": "Repudiation - User denies actions (e.g., purchases) due to lack of authentication or logging
    "severity": "Medium",
    "countermeasure": "Implement user authentication and server-side logging for transactions if order hi
  },
  {
    "asset": "Payment Components (Google Pay, PayPal, etc.)",
    "threat": "Information Disclosure - Leakage of sensitive payment information via insecure integrations
    "severity": "High",
    "countermeasure": "Use only official, up-to-date payment components; ensure all payment data is ha
  },
  {
    "asset": "Static Assets (Images, Icons from external URLs)",
    "threat": "Tampering - Malicious images or assets injected via compromised external sources",
    "severity": "High",
    "countermeasure": "Use trusted asset sources, validate and sanitize loaded assets, and implement C
  },
  {
    "asset": "UI Components (Chakra UI, Material UI, MUI)",
    "threat": "Denial of Service - Exploitation of UI libraries to crash or degrade site performance",
    "severity": "Low",
    "countermeasure": "Keep dependencies updated, monitor for vulnerabilities, and use error boundarie
  },
  {
    "asset": "Product Reviews (Search/filter functionality)",
    "threat": "Information Disclosure - Injection of malicious content in review search/filter (e.g., XSS)",
    "severity": "Medium",
    "countermeasure": "Sanitize all user-generated content and filter input; use React's built-in protection
  },
  {
    "asset": "Routing (React Router)",
    "threat": "Elevation of Privilege - Direct navigation to restricted pages (if authentication is added later)
    "severity": "Medium",
    "countermeasure": "Implement route guards and authentication checks for sensitive pages if user ac
  },
  {
    "asset": "Cart State (Redux)",
    "threat": "Denial of Service - Large or malformed cart data causing UI or payment component failures
    "severity": "Low",
    "countermeasure": "Validate cart data size and format before processing; implement limits on cart siz
  },
  {
    "asset": "Static Product Store (Store.js)",
    "threat": "Information Disclosure - Exposure of sensitive product data if store contains unpublished or
    "severity": "Low",
    "countermeasure": "Ensure only public product data is included in client-side store; move sensitive da
  },
  {
    "asset": "Payment Data",
    "threat": "Denial of Service - Abuse of payment components to flood payment gateway or block legiti
    "severity": "Medium",
    "countermeasure": "Implement rate limiting and CAPTCHA on checkout initiation; monitor for abnorm
  }
]
```