**Component Hierarchy:**  
The application will be structured using the following component hierarchy:

* <App>: The root component. Its primary responsibility is to set up the React Router, providing the routing context for the entire application.
* <ProductList>: A direct child of <App>. This component is responsible for fetching the list of all products from the API and managing that data in its local state. It then maps over the state array to render individual product cards.
* <ProductCard>: A child component of <ProductList>. It receives individual product data (e.g., id, image, title, price) via props from its parent. It is a presentational component whose main job is to display the data and provide a clickable link to the detail view.
* <ProductDetail>: This component is not a child of <ProductList> in the component tree. Instead, it is rendered on a separate route (e.g., /product/:id). It is a sibling to <ProductList> in the routing structure defined in <App>.

**State Management and Data Flow:**

* **State Holding Component:** The <ProductList> component will hold the state for the fetched array of products using the useState hook. This state (e.g., products) is declared and managed within this component.
* **Props Receiving Components:**
  1. The <ProductCard> component receives its data via props passed down from the <ProductList> component during the .map() operation. It receives read-only data to display.
  2. The <ProductDetail> component does not receive product data via props from a parent. Instead, it receives a product ID via a **URL parameter** from React Router. It uses this ID to fetch the specific product's data directly from the API and manage it in its own local state with useState.

