**Objective**

Create a responsive and performant e-commerce product catalog page using React. The page should display a list of products, allow filtering and sorting, and implement a shopping cart functionality.

**Requirements**

**1. Product Listing**

* Fetch product data from a mock API (you can use json-server or any other mock API service).
* Display products in a grid layout with product image, name, price, and "Add to Cart" button.
* Implement infinite scrolling or pagination for loading more products.

**2. Filtering and Sorting**

* Add a sidebar with filter options (e.g., by category, price range).
* Implement sorting functionality (e.g., by price, popularity).
* Ensure filters and sorting work together correctly.

**3. Shopping Cart**

* Create a cart component that displays added items, quantities, and total price.
* Implement "Add to Cart" and "Remove from Cart" functionality.
* Use React Context or Redux for state management.

**4. Styling and Responsiveness**

* Use a CSS-in-JS solution (e.g., styled-components, Emotion) or CSS modules.
* Ensure the layout is responsive and works well on mobile, tablet, and desktop.
* Implement a coherent and visually appealing design.

**5. Performance Optimization**

* Implement code splitting and lazy loading for better initial load time.
* Use React.memo, useMemo, and useCallback where appropriate to optimize re-renders.
* Implement image lazy loading for product images.

**6. API Integration and Error Handling**

* Handle API errors gracefully and display user-friendly error messages.
* Implement loading states while fetching data.
* Use React Query or SWR for efficient data fetching and caching.

**7. Testing**

* Write unit tests for key components using Jest and React Testing Library.
* Implement at least one integration test for a critical user flow (e.g., adding a product to the cart).

**8. Documentation and Code Quality**

Write clear comments and documentation for complex parts of the code.

* Follow React best practices and maintain consistent code style.
* Use ESLint and Prettier for code formatting.

**Bonus Points**

* Implement a simple product search functionality.
* Add animations for smoother user experience (e.g., cart updates, filtering).
* Implement a basic checkout process mockup.

**Evaluation Criteria**

* Code quality and organization
* Performance and optimization techniques
* UI/UX design and responsiveness
* Error handling and edge cases
* Testing coverage and quality
* Documentation and code comments

**Submission**

* Provide a GitHub repository with your code.
* Include a README with instructions on how to run the project locally.
* Deploy the application to a free hosting service (e.g., Vercel, Netlify) and provide the live URL.