Project Documentation

1. Technology Stack

- **React**: The primary UI library used to build interactive user interfaces.
- **TypeScript**: Ensures strict types, enhancing maintainability and reducing runtime errors.
- **Tailwind CSS**: For styling and making the user interface responsive across devices.
- **React Hook Form**: Used for form handling, specifically on the sign-up page.
- **Yup**: Applied for form validation to ensure correct input handling during sign-up.
- **React Router**: Manages navigation and routing throughout the application.
- **useContext and useReducer**: Used for managing state across the app instead of Redux, as Redux was determined to be unnecessary for this project.

2. Business Logic

User Interface

- **Responsive Design**: Tailwind CSS is used to ensure that the entire interface is responsive, adapting gracefully to various screen sizes and devices.

Product Display and Functionality

- **Product Fetching**: All products are fetched and displayed according to the design of the user interface.
- **Wishlist and Cart**:
    - Users can add products to a wishlist or a shopping cart.
    - Separate states managed by `useContext` and `useReducer` ensure efficient storage and retrieval.

User Account Management

- **Account Page**:
    - Displays logged-in user details, accessible via a protected route.
    - Only accessible to authenticated users.

Authentication and Authorization

- **User Authentication**:
    - Only login functionality is implemented according to the API's requirement.
    - POST request is used to simulate user creation on sign-up, with responses logged to the console.
- **Protected Routes**:
    - Key routes, such as the account management and cart pages, are protected.
    - Users must log in to view these routes, improving security.

Order Functionality

- **Restricted Order Access**: Users must be logged in to place orders, enforced through protected routing.

3. Technical Implementation Details

- **TypeScript**: Strict type checking ensures accurate data handling, reducing bugs.
- **React Hook Form and Yup**:
    - These are combined for the sign-up form, allowing form state to be managed smoothly.
    - Yup validates input fields, providing feedback to users on any incorrect entries before submission.
- **React Router**: Used to set up navigation, with conditional access for protected routes.
- **State Management with useContext and useReducer**:
    - `useContext` and `useReducer` are employed instead of Redux to manage state for simplicity.
    - This setup maintains user authentication, cart, and wishlist states separately, keeping each component decoupled.