



# Frontend Technical Assessment

## Objective:

Develop a React application that demonstrates your skills in component architecture, API integration, state management, and styling. Use the [Fake Store API](#) as the backend for this project.

---

## Requirements

### 1. Product Listing Page

#### Features:

- Fetch and display a list of products using the /products endpoint.
- Display essential product details for each product:
  - **Image**
  - **Name**
  - **Price**
  - **Category**
- Add a "View Details" button for each product that navigates to the Product Details Page.
- Include sorting options:
  - By **price** (ascending/descending).
  - By **category**.
- Paginate the product list (e.g., 10 products per page).

#### UX Considerations:

- Show a loading indicator while fetching data.
- Gracefully handle API errors with an error message.
- Display an appropriate message for empty product lists (e.g., "No products found").

## 2. Product Details Page

### Features:

- Display detailed information about the selected product, including:
    - **Image**
    - **Title**
    - **Description**
    - **Category**
    - **Price**
  - Include a "Back to Products" button to navigate back to the Product Listing Page.
- 

## 3. Create Product Page

### Features:

- Provide a form to create a new product with the following fields:
  - **Title** (text input)
  - **Description** (textarea)
  - **Price** (number input)
  - **Category** (dropdown fetched from /products/categories)
  - **Image URL** (text input)
- Validate all fields:
  - Required fields must be filled.
  - Ensure the price is a positive number.
- Submit the form via a POST request to the /products endpoint.
- Show a success message upon successful creation.

### UX Considerations:

- Handle loading and error states during form submission.
- Disable the submit button during submission.

## Bonus Features

### 1. Cart

#### Features:

- Allow users to add products to a cart from:
    - The **Product Listing Page**.
    - The **Product Details Page**.
  - Create a Cart Page to display:
    - Selected products with their **image, name, quantity, and price**.
    - Total price calculation.
  - Allow users to:
    - Update product quantities.
    - Remove products from the cart.
- 

### 2. Authentication

#### Features:

- Implement login and signup functionality using the `/auth/login` endpoint.
- Restrict access to the following pages for logged-in users only:
  - **Create Product Page**.
  - **Cart Page**.
- Display the logged-in user's name in the header with a "Logout" button.
- Persist authentication state across page reloads (e.g., using local storage or cookies).

## Technical Requirements

- **Framework:** Use React or Next.js.
- **State Management:** Use a state management solution like React Context, Redux, or Zustand.
- **Styling:** Use modern styling approaches such as TailwindCSS, CSS-in-JS, or styled-components.
- **API Calls:** Use Axios or Fetch for API requests.
- **Responsiveness:** Ensure the application is fully responsive and works seamlessly on mobile and desktop.
- **Error Handling:** Gracefully handle errors and display appropriate messages for:
  - API failures.
  - Empty states.

## Evaluation Criteria

- Code organization and clarity.
- Component reusability and modularity.
- Proper state management and data flow.
- Error handling and loading states.
- UX and responsiveness.
- Clean and consistent styling.