JustGo Technologies Limited

https://www.justgo.com

Frontend Engineer Recruitment: React Coding Challenge

Overview

You are required to build a project based on React. Please follow the details below and comply accordingly. This React assignment will assess your thought process, comprehension and decision-making.

Timeline:
9.00 pm, Wednesday, 28 January,2026 (Submitting before the deadline is encouraged)

Project: Product Explorer Dashboard

You will build a small React application that displays and explores product data.

Use the public API: **https://dummyjson.com/**

Application Routes:

- /products
- /products/{id}
- /products/categories
- /products/search?q=
- /settings

Functional Requirements:

Note: *Design artifacts, such as Figma files or wireframes, will not be supplied. The use of professional judgment and best practices is required.*

Core Features (Required)

- Product list page
- Product details page
- Search products by title
- Sort by product’s Unit price
- Filter products by category

- The search results page is a standalone, navigable entity. Direct access (e.g., via URL or bookmark) must yield results identical to an in-app search. The URL or persistent state must fully capture the query, filters, sorting, and pagination to ensure consistent data regardless of the access path.
- Implement infinite scrolling, fetching 20 items per load.
- Use Table (<table>) to display Product list. You must not use any third-part library instead a re-usable component is encouraged .
- Use Loading skeleton and loading placeholder where it requires
- Implement a loading skeleton or a loading placeholder in areas where dynamic content requires it.
- The settings page must include a single configurable option: Currency. This should be implemented as a radio button group with the choices: USD, Pound, and Euro. The selected currency value must be utilized wherever price data is displayed throughout the application. To manage and access this configuration value globally, the React Context API should be employed.
- Implement Error Boundary
- Responsive layout (desktop + mobile)

Technical Requirements:

You must use:

- React + TypeScript
- Vite
- React Router
- React Query (server state)
- Zustand (client/global state)
- Tailwind CSS

You do not need to:

- Implement authentication
- Create a backend

State Management Expectations

We expect a clear separation of concerns:

Concern	**Suggested Tool**
API / server data	React Query
UI state (filters, search etc.)	Zustand
Routing	React Router

README.md (Very Important)

Your submission must include a README.md answering the following:

1. What trade-offs did you consciously make due to time constraints?
2. If this app needed to scale (more data, more features), what would you refactor first?
3. Did you use AI tools? If yes, which parts and how did you verify correctness?

Note: *Submissions without this README will not be considered.*

AI Usage Policy

- AI tools are allowed
- You are expected to understand and explain all submitted code
- During the interview, you may be asked to:
    - Walk through your code
    - Modify a feature
    - Fix a small bug

Note: *Using AI without understanding the output will be immediately obvious.*

Submission Guidelines

Please submit:

- A public GitHub repository link
- Clear setup instructions (npm install, npm run dev)
- README.md (mandatory)

Optional but appreciated:

- Sensible commit history
- Basic linting setup