Frontend Developer Take-Home Assignment

E-Commerce Product Dashboard with Real API Integration

Overview

Create a **Product Dashboard** application using the **DummyJSON API** that demonstrates your proficiency in Next.js, React, TypeScript, and modern frontend development practices. This assignment should take **4-5 days** to complete.

The Task

Build a product inventory dashboard that integrates with the DummyJSON Products API.

API Documentation

- **Base URL:** `https://dummyjson.com`
- **API Docs:** https://dummyjson.com/docs/products
- **Total Products:** 194 items across 24 categories
- **No authentication required**

Required Features

1. Product List View with Pagination

- Display products in a responsive grid layout
- Implement pagination using the API's `limit` and `skip` parameters
- Show 12 products per page
- Add pagination controls (Previous/Next or page numbers)
- Each product card must display:
- Product thumbnail image
- Product title
- Price with discount percentage (if applicable)
- Rating (stars or numeric)
- Stock status badge ("In Stock" / "Low Stock" / "Out of Stock")
- Category tag

API Endpoint:
GET /products?limit=12&skip=0

2. Search Functionality

- Implement real-time search using the API's search endpoint
- Search input with debouncing (300ms delay)
- Display search results count
- Show "No results found" state with a helpful message
- Clear search button

API Endpoint:

GET /products/search?q={searchQuery}

3. Category Filtering

- Fetch and display all available categories
- Filter products by selected category
- Display active filter state
- "All Categories" option to reset filter
- Show product count for filtered results

API Endpoints:
GET /products/categories
GET /products/category/{categorySlug}

4. Sorting Options

- Sort by Price (Low to High / High to Low)
- Sort by Title (A-Z / Z-A)
- Sort by Rating (Highest first)
- Use the API's `sortBy` and `order` parameters

API Endpoint:
GET /products?sortBy=price&order=asc

5. Product Detail Modal

- Click on any product card to open a detailed view
- Display full product information:
- Image gallery (multiple images)
- Full description
- Price and discount
- Rating with reviews
- Stock availability status
- Brand, SKU, dimensions, weight

- Warranty and shipping information
- Customer reviews (show at least 3)
- Close modal on backdrop click, ESC key, or close button
- Keyboard navigation support

API Endpoint:
GET /products/{id}

6. Shopping Cart (UI Only)

- Cart icon in header with count badge
- Add to cart button on product cards and detail modal
- Cart count persists on page refresh (localStorage)
- Visual feedback when an item is added (toast notification or animation)
- No need to implement full cart management

Required Tech Stack

- **Next.js 15** (App Router)
- **TypeScript** (strict mode)
- **Tailwind CSS** for styling
- **React hooks** for state management
- **SWR or TanStack Query** for data fetching

API Integration Best Practices

- Centralize API calls in a `lib/api.ts` file
- Implement proper error handling
- Show loading states (skeleton screens)
- Handle empty states gracefully
- Cache API responses appropriately
- Add error boundaries for graceful failures

Code Quality Standards

- Clean, readable, well-organized code
- Proper TypeScript typing (avoid `any`)
- Component composition (small, reusable components)
- Semantic HTML elements
- Accessibility considerations
- Consistent naming conventions

Bonus Points (Optional)

These are not required but demonstrate advanced skills:

- Unit tests (Jest + React Testing Library)
- Implement URL query parameters for filters/search (shareable links)
- Add Framer Motion animations
- Dark mode toggle with persistence

What to Include in Your Email Submission:

1. **GitHub Repository link** (public)
2. **Live Demo link** (deploy to Vercel/Netlify)
3. **Design Decisions**:[Explain your approach to layout, color choices, component structure, state management strategy, and why you chose certain libraries]
4. **challenges & solutions**:[Describe 1-2 technical challenges you faced and how you solved them]