React/Next.js Frontend Developer Assignment: Mini Product Catalog

Goal: Build a small, responsive product catalog application using Next.js and React.

Introduction:
This assignment assesses your ability to build a functional frontend application using React and Next.js. We want to see how you structure your code, handle state, fetch data, implement routing, and apply styling. Focus on code quality, clarity, and best practices.

Project Description:
You will create a simple application that displays a list of products fetched from a provided API endpoint. Users should be able to view the list of products, click on a product to see its details, and potentially filter or sort the products.

Core Requirements:

1. **Technology Stack:**
    - **Framework:** Next.js (latest stable version)
    - **Language:** JavaScript or TypeScript (TypeScript is preferred but not mandatory unless specified otherwise).
    - **Styling:** Choose **one** of the following and use it consistently:
        - Tailwind CSS
        - CSS Modules
        - Styled Components / Emotion
    -
    - **API:** Use the FakeStore API (https://fakestoreapi.com/) - specifically the /products endpoint. No API key is required.
2. **Pages & Routing:**
    - **Product Listing Page (/ or /products):**
        - Fetch and display a grid or list of products from https://fakestoreapi.com/products.
        - Each product item should display at least: Image, Title, Price, Category.
        - Implement basic loading state UI while fetching data.
        - Implement basic error handling UI if the API call fails.
        - Clicking on a product item should navigate the user to the Product Detail page.
    -
    - **Product Detail Page (/products/[id]):**
        - Fetch and display detailed information for a single product using its ID (e.g., https://fakestoreapi.com/products/1).
        - Display: Image, Title, Price, Category, Description.
        - Handle cases where the product ID is invalid or the API request fails.
        - Provide a way to navigate back to the Product Listing page (e.g., a "Back" button or breadcrumbs).

    -
3. **Data Fetching:**
    - Use appropriate Next.js data fetching methods (e.g., getStaticProps, getServerSideProps, client-side fetching with SWR/React Query or useEffect).
    - **Justify your choice** of data fetching strategy (or strategies) in the README.md.
4. **Responsiveness:**
    - The application layout should be responsive and usable on different screen sizes (mobile, tablet, desktop).
5. **Code Quality & Structure:**
    - Organize your code into logical components and folders.
    - Write clean, readable, and maintainable code.
    - Use functional components and React Hooks.
    - Use next/link for internal navigation and next/image for image optimization.

Bonus Features (Optional - Choose any you like):

- **Filtering:** Add functionality to filter products by category on the Product Listing page.
- **Sorting:** Add functionality to sort products by price (ascending/descending) or title on the Product Listing page.
- **Search:** Implement a basic search bar to filter products by title on the Product Listing page.
- **State Management:** If implementing features like filtering/sorting/search, consider using useReducer or useContext for managing shared state if useState becomes cumbersome. (Avoid complex libraries like Redux unless you feel it's strongly justified).
- **Loading Skeletons:** Implement skeleton loaders for a better loading state experience instead of just a simple "Loading..." message.
- **Pagination:** If the API supported it (or you mock it), implement pagination on the Product Listing page. (FakeStoreAPI doesn't directly support pagination on the main /products endpoint, so you might need to fetch all and paginate client-side or use limit and offset if available on other endpoints).
- **Testing:** Write basic unit tests for a few key components or utility functions using Jest and React Testing Library.
- **TypeScript:** If you initially chose JavaScript, implement the project using TypeScript.
- **Deployment:** Deploy the application to Vercel or Netlify and provide the live URL.

Deliverables:

1. A link to a public Git repository (e.g., GitHub, GitLab) containing your source code.
2. Ensure your commit history is meaningful.
3. A README.md file in the repository root that includes:
    - Clear instructions on how to install dependencies and run the project locally (npm install && npm run dev or yarn install && yarn dev).
    - A brief explanation of your project structure.

    - Justification for your chosen data fetching strategy/strategies.
    - Mention which styling solution you used.
    - List any bonus features you implemented.
    - (Optional) Link to the live deployment if you completed that bonus feature.
4.

Evaluation Criteria:

- **Functionality:** Does the application meet all core requirements? Do features work correctly?
- **Code Quality:** Is the code clean, readable, maintainable, and well-structured? Are React/Next.js best practices followed?
- **Next.js Concepts:** Correct use of file-based routing, data fetching methods, next/link, next/image.
- **React Concepts:** Proper use of hooks, component composition, state management, event handling.
- **Styling:** Is the chosen styling method used effectively and consistently? Is the UI clean and functional?
- **Responsiveness:** Does the application adapt well to different screen sizes?
- **Error Handling:** How gracefully does the application handle API errors or missing data?
- **Git History:** Is the commit history clear and logical?
- **README:** Is the README clear and informative?
- **Bonus Features:** Quality of implementation for any bonus features attempted.

Time Estimate:
We estimate this assignment should take approximately **4-8 hours** of focused work. However, this is just a guideline. Focus on quality over quantity. If you run short on time, prioritize completing the core requirements well.

Questions:
If you have any questions about the assignment, please reach out to wasil@serri.club.

Good luck! We look forward to seeing your submission.