Day 3 - API Integration Report - [Ann Fashion Store]

Hackathon Task #3

Product Detail Page for Sanity CMS:

This documentation explains the structure and functionality of the **Product Detail Page**, which displays detailed information about a single product fetched from **Sanity CMS**.

The ProductDetailPage is a **server-side component** in Next.js that:

1. Fetches product data from **Sanity CMS** using a unique id.
2. Displays the product details using a client-side UI component called ProductDetailUI

- **Query:**

  - Fetches a single product from the post document where _id matches the provided id.
  - Retrieves fields:
    - ProductName: Name of the product.
    - ProductPrice: Price of the product.
    - ProductDescription: Description of the product.
    - imageUrl: URL of the product image.

- **Parameters:**

  - The id parameter is passed dynamically to filter the specific product.

Key Features

1. **Server-Side Rendering (SSR):**
   - The product data is fetched on the server before rendering the page, ensuring up-to-date content.
2. **Dynamic Routing:**

    - Utilizes Next.js dynamic routing ([id].tsx) to generate pages for each product based on its unique id.
3. **Error Handling:**
    - Displays a user-friendly message if the product is not found.

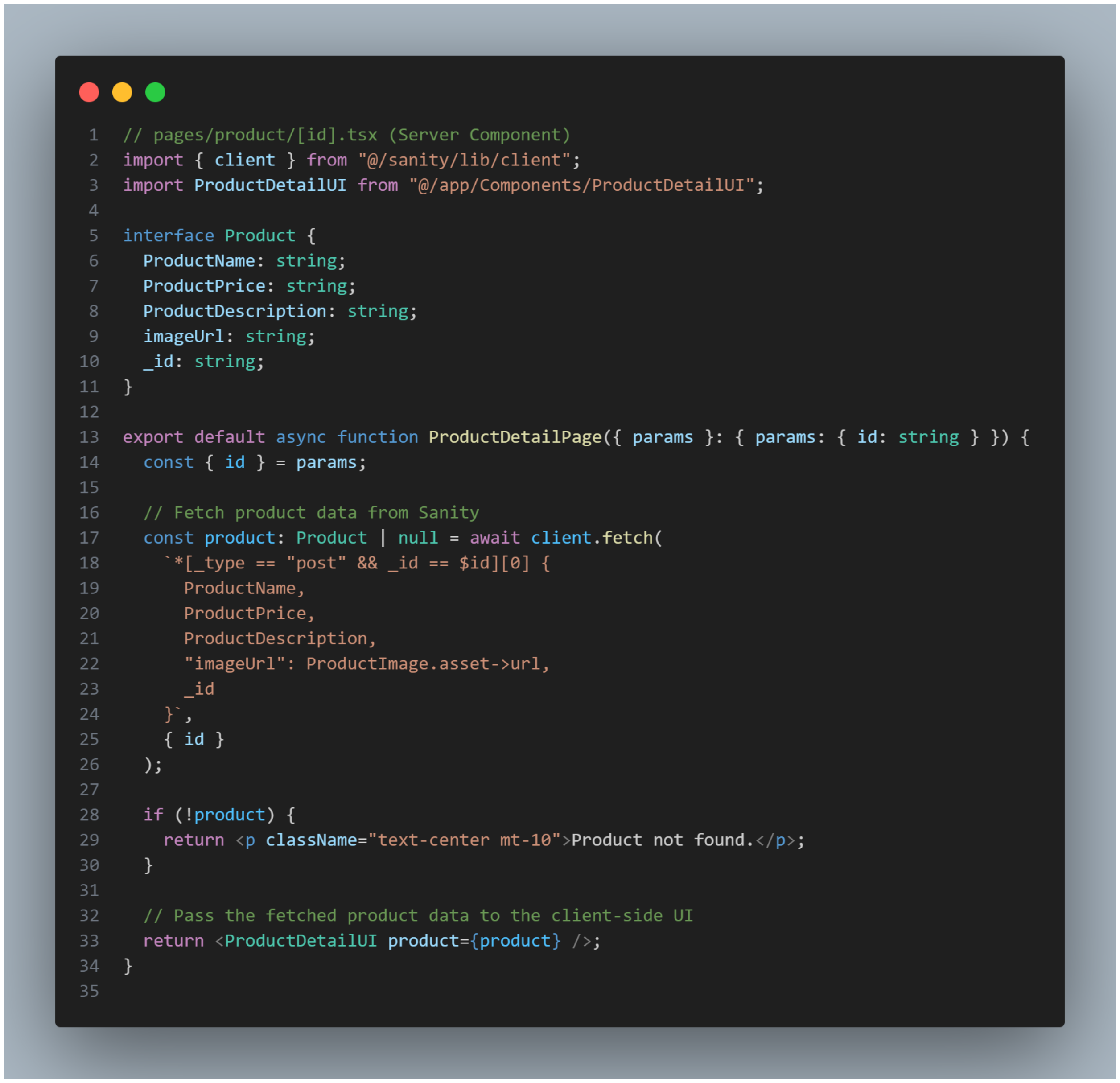
Usage

1. **URL Structure:**
    - Access a product's detail page via /product/{id}, where {id} is the product's unique identifier.
2. **Real-Time Updates:**
    - The use of server-side fetching ensures the product details are always up-to-date with the latest data in Sanity CMS.

Benefits

- **Performance:** Server-side rendering ensures faster initial page loads and better SEO.
- **Scalability:** Easily handles dynamic content for multiple products by using unique id parameters.
- **Modularity:** Separates data fetching and UI rendering for better maintainability.

This component is a core part of building an e-commerce website or marketplace, providing dynamic and detailed product pages for users.



```tsx
// pages/product/[id].tsx (Server Component)
import { client } from "@/sanity/lib/client";
import ProductDetailUI from "@/app/Components/ProductDetailUI";

interface Product {
  ProductName: string;
  ProductPrice: string;
  ProductDescription: string;
  imageUrl: string;
  _id: string;
}

export default async function ProductDetailPage({ params }: { params: { id: string } }) {
  const { id } = params;

  // Fetch product data from Sanity
  const product: Product | null = await client.fetch(
    `*[_type == "post" && _id == $id][0] {
      ProductName,
      ProductPrice,
      ProductDescription,
      "imageUrl": ProductImage.asset->url,
      _id
    }`,
    { id }
  );

  if (!product) {
    return <p className="text-center mt-10">Product not found.</p>;
  }

  // Pass the fetched product data to the client-side UI
  return <ProductDetailUI product={product} />;
}
```