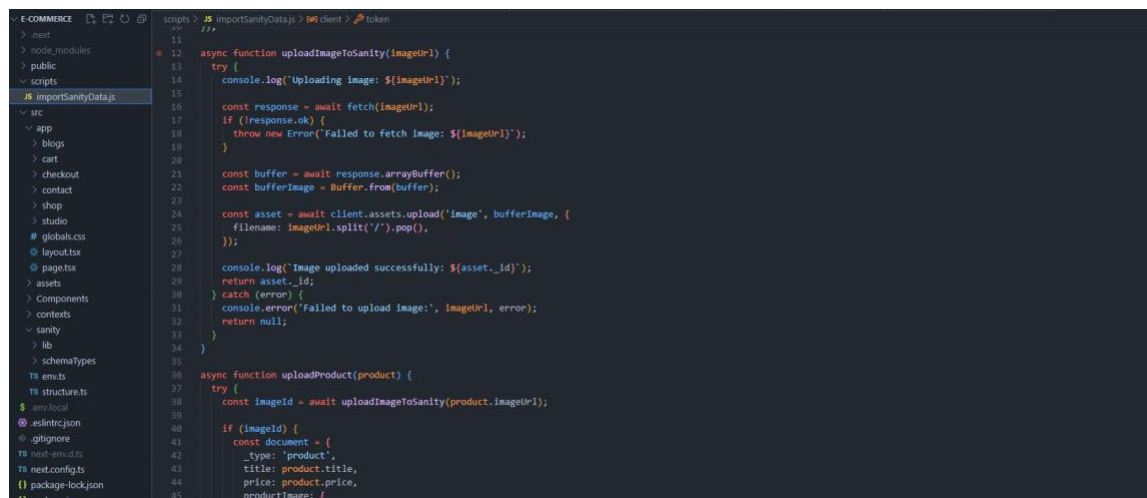


Day 3: API Integration and Data Migration for FurnitureHub Marketplace

Overview

On the third day of the FurnitureHub Marketplace hackathon, we focused on migrating data from Sanity CMS, integrating APIs, and building dynamic features in a Next.js application. Key achievements included setting up a data pipeline, designing schemas, and implementing server-side rendering for optimized performance.

Key Activities



```
11 // ...
12 async function uploadImageToSanity(imageUrl) {
13   try {
14     console.log('Uploading image: ${imageUrl}');
15
16     const response = await fetch(imageUrl);
17     if (!response.ok) {
18       throw new Error('Failed to fetch image: ${imageUrl}');
19     }
20
21     const buffer = await response.arrayBuffer();
22     const bufferImage = Buffer.from(buffer);
23
24     const asset = await client.assets.upload('image', bufferImage, {
25       filename: imageUrl.split('/').pop(),
26     });
27
28     console.log('Image uploaded successfully: ${asset._id}');
29     return asset._id;
30   } catch (error) {
31     console.error('Failed to upload image:', imageUrl, error);
32     return null;
33   }
34 }
35
36 async function uploadProduct(product) {
37   try {
38     const imageId = await uploadImageToSanity(product.imageUrl);
39
40     if (imageId) {
41       const document = {
42         _type: 'product',
43         title: product.title,
44         price: product.price,
45         productImage: {
```

1. Data Migration from Sanity

A custom script was created to migrate data from Sanity CMS to the FurnitureHub database. Here's how it works:

Sanity API Setup:

Connected to the Sanity API using environment variables for project ID, dataset, and API token.

Ensured secure authentication using process.env.

Fetching Data:

Used GROQ queries to retrieve structured data like furniture categories, descriptions, prices, and images.

Data Mapping:

Reformatted data to align with the FurnitureHub database schema.

Database Insertion:

Transferred data via bulk insertion for improved efficiency.

2. Client-Side Rendering

```
}  
⚡  
function Products() {  
  const [products, setProducts] = useState<Product[]>([]);  
  const [cart, setCart] = useState<Product[]>([]);  
  
  const fetchProducts = async () => {  
    try {  
      const query = `*[_type=="product"]{  
        _id,  
        title,  
        price,  
        description,  
        discountPercentage,  
        "imageUrl": productImage.asset->url,  
        tags  
      }`;  
      const data = await sanity.fetch(query);  
      setProducts(data);  
    } catch (error) {  
      console.error('Error fetching products:', error);  
    }  
  };  
  
  const addToCart = (product: Product) => {  
    setCart((prev) => [...prev, product]);  
    alert(`${product.title} successfully added to the cart`);  
  };  
  
  useEffect(() => {  
    fetchProducts();  
  }, []);  
}
```

Next.js components were built to display migrated data dynamically:

Data Fetching:

Utilized `getServerSideProps` to prefetch data during server-side rendering, improving performance and SEO.

Dynamic Item Cards:

Each product was displayed using reusable item card components with responsive styling.

Cards included product titles, images, prices, and links to detailed pages via dynamic routing.

3. Schema Design

The Sanity CMS schema defined the structure for storing furniture data. Key fields include:

Title: Required string field for the furniture name.

Slug: Unique, URL-friendly identifier.

Description: Detailed text about the product.

Price: Positive numeric value.

Image: High-resolution image with alt text for accessibility.

Security

Environment Variables:

API tokens and configurations were stored securely in `.env` files.

Accessed using `process.env` to prevent exposure in the frontend.

Conclusion

Day 3 established the foundation for FurnitureHub's backend and frontend integration. Key deliverables included:

Seamless data migration from Sanity CMS.

Optimized schemas ensuring consistent and scalable data storage.

Dynamic frontend components for a responsive user interface.

This work ensures a solid, extensible structure for the FurnitureHub Marketplace, facilitating further development and scalability.