Day 3 - API Integration and Data Migration Report

Template 3

Prepared By: Muhammad Nasrullah

Date: February 18, 2025

1. Introduction

The focus of Day 3 was on seamlessly integrating APIs into our Next.js project and migrating data to Sanity CMS. This involved fetching product data from Template 3 API, mapping it to the CMS schema, and rendering it dynamically on the frontend. Through this process, we gained hands-on experience in API handling, schema adjustments, and data transformation.

2. API Integration

API Used: https://template-03-api.vercel.app/api/products

Steps Followed:

1. Created a utility function to fetch API data:

```
export async function fetchProducts() {
    const res = await fetch('https://template-03-api.vercel.app/api/products');
    return await res.json();
}
```

2. Tested API using Postman/Browser.

3. Ensured error handling during API calls:

```
try {
    const data = await fetchProducts();
} catch (error) {
    console.error('Error fetching products:', error);
}
```

3. Data Migration

Schema Used:

Imported the provided schema from Template 3 repository.

Migration Script:

```
import sanityClient from './sanityClient';
```

```
const migrateData = async () => {
    const products = await fetchProducts();
    products.forEach(async (product) => {
        await sanityClient.create({
            _type: 'product',
            title: product.title,
            price: product.price,
            description: product.description,
        });
    });
};
migrateData();
```

4. Frontend Integration

Components Created:

1. Created a ProductList component to display fetched data:

```
const ProductList = () => {
    const [products, setProducts] = useState([]);
    useEffect(() => {
        fetchProducts().then(setProducts);
    }, []);
    return (
        <div>
            {products.map((product) => (
                <div key={product.id}>
                    <h3>{product.title}</h3>
                    <p>{product.description}</p>
```

```
                <span>{product.price}</span>
            </div>
        ))}
    </div>
  );
};
export default ProductList;
```

5. Error Handling and Conclusion

Challenges:

- Schema mismatches between API and CMS.

Solution:

Adjusted schema to match API fields and added error handling in API calls.

Conclusion: Successfully integrated Template 3 API, migrated data to Sanity CMS, and displayed it on the frontend. Learned how to handle schema validation, API errors, and data transformation for real-world applications.