Day 3 - API Integration and Data Migration

Creator: Ahsan Paracha | Slot: Sunday 2-5 | Roll No: 0088100

Table of content

---

1. API Integration Process

The API integration process involved fetching product data from an external API and integrating it into our Sanity CMS and Next.js frontend. The process was designed to automatically retrieve product information, including images, names, prices, descriptions, discounts, and categories, and then upload these details into the Sanity backend. This integration ensures that our frontend always displays up-to-date product data.

---

2. Schemas in Sanity

Product.ts:

```
export default {

    name: 'product',

    type: 'document',

    title: 'Product',

    fields: [

      {

        name: 'name',

        type: 'string',

        title: 'Name',

      },

      {

        name: 'image',

        type: 'image',

        title: 'Image',

        options: {
```

```
        hotspot: true, // Enable image cropping
      },
    },
    {
      name: 'price',
      type: 'number',
      title: 'Price',
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
    },
    {
      name: 'discountPercentage',
      type: 'number',
      title: 'Discount Percentage',
    },
    {
      name: 'isFeaturedProduct',
      type: 'boolean',
      title: 'Featured Product',
    },
```

```
      {
        name: 'stockLevel',
        type: 'number',
        title: 'Stock Level',
      },
      {
        name: 'category',
        type: 'string',
        title: 'Category',
      },
    ],
  };
```

Index.ts

```
import { type SchemaTypeDefinition } from 'sanity'
import product from './product'

export const schema: { types: SchemaTypeDefinition[] } = {
  types: [product],
}
```

env.local:

```
NEXT_PUBLIC_SANITY_PROJECT_ID="587tggpl"
NEXT_PUBLIC_SANITY_DATASET="production"
```

```
SANITY_API_TOKEN="skvogBt05WwY4g1QWg8foS3MuBjqzR3lvpaaDaeEk4E4IlS1tMlNAFJs
st8WeDuV3GrJA6Rbh2wmw6jbNWnVnNvZQeAdoiNM4Qhhjy09Ke1jsbX3z8arSgsoF9iYYm01fm
byCg7mfwWYvA1FTsJ8Mf2Fkf67yEBUF1rYrCoD6AvOX4N7ojo4"
```

---

3. Code Snippets for API Integration and Migration Scripts

Migration Script: `importSanityData.mjs`

```
import { createClient } from '@sanity/client';
import fetch from 'node-fetch';

// Initialize Sanity client
const client = createClient({
  projectId: "587tggpl",
  dataset: "production",
  useCdn: false, // Set to true if you want faster reads
  apiVersion: '2025-01-13',
  token:
"skvogBt05WwY4g1QWg8foS3MuBjqzR3lvpaaDaeEk4E4IlS1tMlNAFJsst8WeDuV3GrJA6Rbh
2wmw6jbNWnVnNvZQeAdoiNM4Qhhjy09Ke1jsbX3z8arSgsoF9iYYm01fmbyCg7mfwWYvA1FTsJ
8Mf2Fkf67yEBUF1rYrCoD6AvOX4N7ojo4", // Replace with your Sanity token
});

// Function to upload an image to Sanity
async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);

    const response = await fetch(imageUrl);
    if (!response.ok) {
      throw new Error(`Failed to fetch image: ${imageUrl}`);
    }

    const buffer = await response.arrayBuffer();
    const bufferImage = Buffer.from(buffer);

    const asset = await client.assets.upload('image', bufferImage, {
```

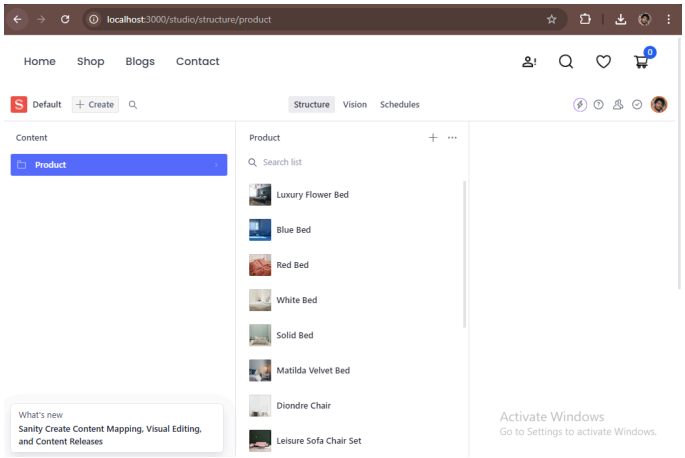
```
      filename: imageUrl.split('/').pop(),
    });

    console.log(`Image uploaded successfully: ${asset._id}`);
    return asset._id;
  } catch (error) {
    console.error('Failed to upload image:', imageUrl, error);
    return null;
  }
}

// Function to upload a single product to Sanity
async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imagePath);

    if (imageId) {
      const document = {
        _type: 'product',
        id: product.id,
        name: product.name,
        image: {
          _type: 'image',
          asset: {
            _ref: imageId,
          },
        },
        price: parseFloat(product.price), // Ensure the price is a number
        description: product.description,
        discountPercentage: product.discountPercentage,
        isFeaturedProduct: product.isFeaturedProduct,
        stockLevel: product.stockLevel,
        category: product.category,
      };

      const createdProduct = await client.create(document);
      console.log(`Product "${product.name}" uploaded successfully:`,
createdProduct);
    } else {
```

```
      console.log(`Product "${product.name}" skipped due to image upload
failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

// Function to fetch products from the provided API and upload them to
Sanity
async function migrateProducts() {
  try {
    const response = await
fetch('https://template-0-beta.vercel.app/api/product');

    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);
    }

    const products = await response.json();

    for (const product of products) {
      await uploadProduct(product);
    }
  } catch (error) {
    console.error('Error fetching products:', error);
  }
}

// Start the migration
migrateProducts();
```

Data Succssfully imported:

```
  isFeaturedProduct: true,
  name: 'Luxury Flower Bed',
  price: 2500,
  stockLevel: 2
}

E:\Day 3\project-rock>npm run dev

> project-rock@0.1.0 dev
> next dev

  ▲ Next.js 14.2.2
  - Local:        http://localhost:3000
  - Environments: .env.local

 ✓ Starting...
```

http://localhost:3000/studio/structure/product



Frontend Display Code Snippet: `app/items/page.tsx`

```
"use client"
import { useEffect, useState } from 'react';
import { createClient } from '@sanity/client';
import imageUrlBuilder from '@sanity/image-url';
import Image from 'next/image';
import Link from 'next/link';
```

```
const sanity = createClient({
  projectId: "587tggpl",
  dataset: "production",
  apiVersion: "2025-01-13",
  useCdn: true,
});

const builder = imageUrlBuilder(sanity);

interface Product {
  _id: string;
  name: string;
  price: number;
  description: string;
  discountPercentage: number;
  image: {
    asset: {
      _ref: string;
    };
  };
  category: string;
};

const ProductCards = () => {
  const [products, setProducts] = useState<Product[]>([]);
  const [loading, setLoading] = useState(true);
  const [error, setError] = useState<string | null>(null);

  const fetchProducts = async () => {
    try {
      const query = `*[_type == "product"] {
        _id,
        name,
        price,
        description,
        discountPercentage,
        image,
        category
      }`;
```

```
      const data = await sanity.fetch(query);
      setProducts(data);
      setError(null);
    } catch (err) {
      setError('Failed to load products');
      console.error("Error fetching products:", err);
    } finally {
      setLoading(false);
    }
  };

  const addToCart = (product: Product) => {
    // Implement your cart logic here
    alert(`${product.name} added to cart!`);
  };

  useEffect(() => {
    fetchProducts();
  }, []);

  if (loading) return <div className="text-center py-12">Loading
products...</div>;
  if (error) return <div className="text-center py-12
text-red-500">{error}</div>;
  if (!products.length) return <div className="text-center py-12">No
products found</div>;

  return (
    <section className="bg-gray-50 px-4 py-12 md:px-8 md:py-24 lg:px-16">
      <div className="grid grid-cols-1 gap-6 sm:grid-cols-2
lg:grid-cols-4">
        {products.map((product) => {
          const imageUrl = product.image?.asset?._ref
            ? builder.image(product.image).width(600).height(600).url()
            : null;

          const originalPrice = product.discountPercentage > 0
            ? Math.round(product.price / (1 -
product.discountPercentage/100))
```

```
            : null;

          return (
            <div
              key={product._id}
              className="group block transition-transform
hover:scale-[1.02]"
            >
              <div className="rounded-lg border border-gray-200 bg-white
p-4 transition-shadow hover:shadow-md md:p-6">
                <Link href={`/products/${product._id}`} passHref>
                  <div className="relative mb-4 aspect-square
overflow-hidden">
                    {imageUrl ? (
                      <Image
                        src={imageUrl}
                        alt={product.name}
                        fill
                        className="object-cover transition-transform
duration-300 group-hover:scale-105"
                        sizes="(max-width: 640px) 100vw, (max-width:
1024px) 50vw, 25vw"
                        priority={false}
                      />
                    ) : (
                      <div className="bg-gray-100 w-full h-full flex
items-center justify-center">
                        <span className="text-gray-400">No image
available</span>
                      </div>
                    )}
                  </div>
                </Link>

                <div className="text-center">
                  <h3 className="mb-2 line-clamp-2 text-base font-medium
text-gray-800 md:text-lg hover:text-blue-600">
                    <Link href={`/products/${product._id}`}>
                      {product.name}
                    </Link>
```

```
                  </h3>

                  <div className="flex flex-col items-center
justify-center gap-1 mb-3">
                    <p className="text-xl font-bold text-gray-900
md:text-2xl">
                      Rs {product.price.toFixed(2)}
                    </p>
                    {originalPrice && (
                      <div className="flex items-center gap-2">
                        <span className="text-sm line-through
text-gray-500">
                          Rs {originalPrice.toFixed(2)}
                        </span>
                        <span className="text-sm font-medium
text-green-600">
                          ({product.discountPercentage}% off)
                        </span>
                      </div>
                    )}
                  </div>

                  <button
                    onClick={() => addToCart(product)}
                    className="mt-2 w-full rounded-lg bg-blue-600 px-4
py-2 text-sm font-medium text-white transition-colors hover:bg-blue-700
focus:outline-none focus:ring-2 focus:ring-blue-500 focus:ring-offset-2"
                  >
                    Add to Cart
                  </button>
                </div>
              </div>
            </div>
          )
        })}
      </div>
    </section>
  );
};
```

```
export default ProductCards;
```

Frontend Output Screenshot: