Name:Muhammad Hamza Roll#00310223

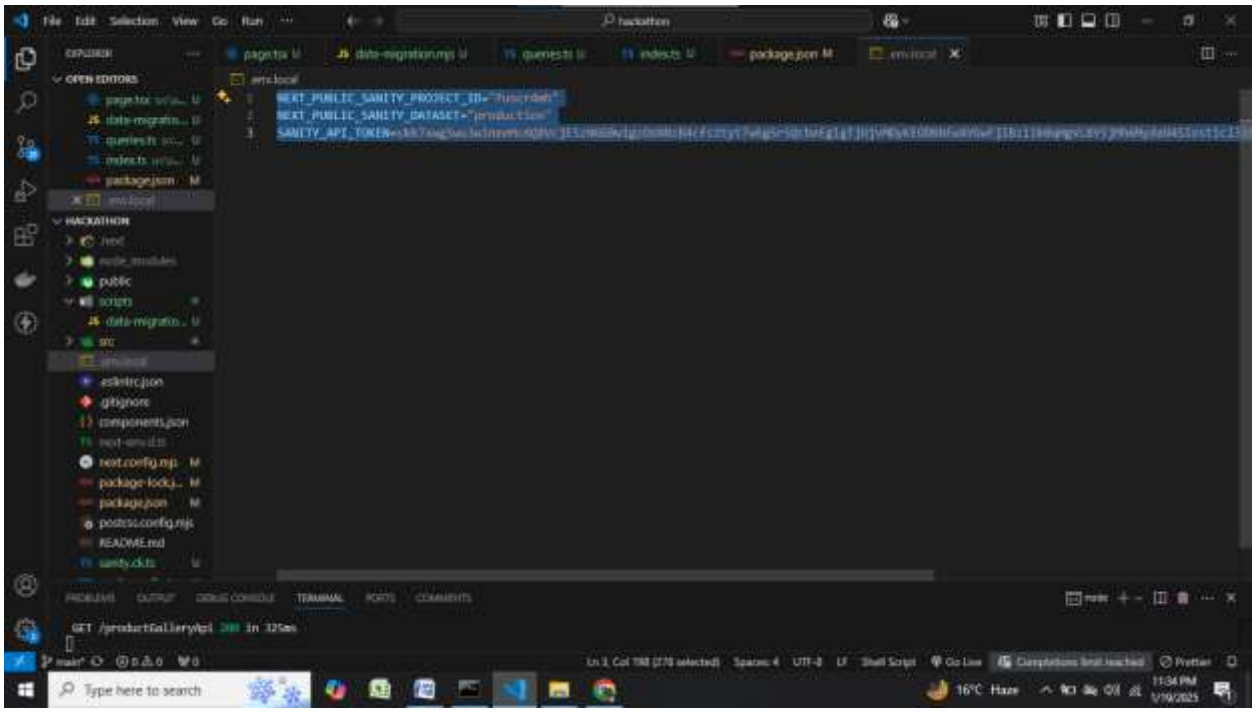
Document Title: Day 3-Api Intergration Report - [Template 3 –Nike website]

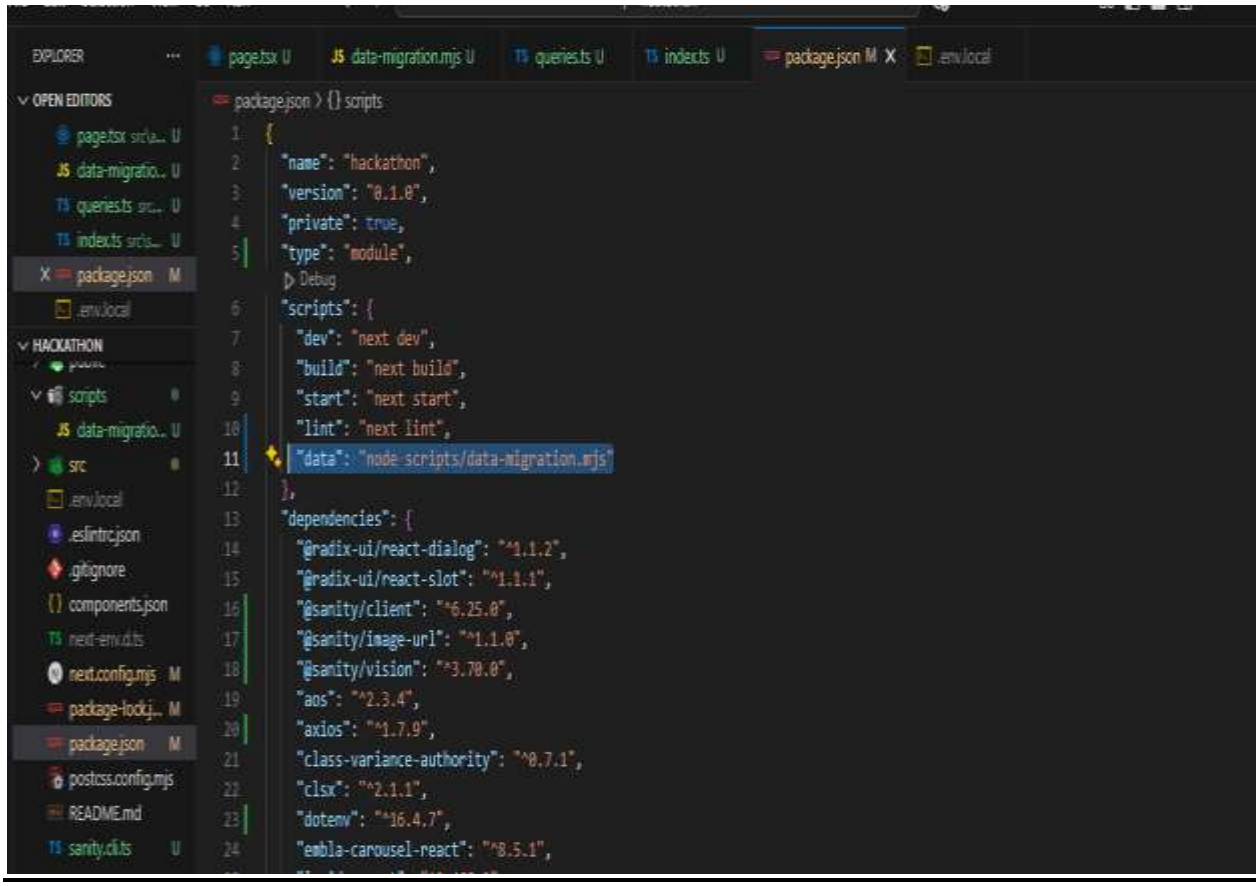
♣ API integration process.

♣ Adjustments made to schemas.

♣ Migration steps and tools used.

Api Token Generated:




EXPLORER
page.tsx U
data-migration.mjs U
queries.ts U
index.ts U
package.json M
.env.local
OPEN EDITORS
HACKATHON
scripts
src
components.json
next.config.mjs M
package-lock.j... M
postcss.config.mjs
README.md
sanity.cli.ts U
package.json > {} scripts
Debug
{
"name": "hackathon",
"version": "0.1.0",
"private": true,
"type": "module",
"scripts": {
"dev": "next dev",
"build": "next build",
"start": "next start",
"lint": "next lint",
"data": "node scripts/data-migration.mjs"
},
"dependencies": {
"@radix-ui/react-dialog": "^1.1.2",
"@radix-ui/react-slot": "^1.1.1",
"@sanity/client": "^6.25.0",
"@sanity/image-url": "^1.1.0",
"@sanity/vision": "^3.70.0",
"aos": "^2.3.4",
"axios": "^1.7.9",
"class-variance-authority": "^0.7.1",
"clsx": "^2.1.1",
"dotenv": "^16.4.7",
"embla-carousel-react": "^8.5.1",

```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// Load environment variables from .env.local
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);
dotenv.config({ path: path.resolve(__dirname, '../.env.local') });

// Create Sanity client
const client = createClient({
  projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
  useCdn: false,
  token: process.env.SANITY_API_TOKEN,
  apiVersion: '2021-08-31'
});


async function uploadImageToSanity(imageUrl) {
  try {
    console.log(`Uploading image: ${imageUrl}`);
    const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
    const buffer = Buffer.from(response.data);
    const asset = await client.assets.upload('image', buffer, {
      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;
  }
}

async function importData() {
  try {
    console.log('migrating data please wait...');

    // API endpoint containing car data
    const response = await axios.get('https://template-03-api.vercel.app/api/products');
    const products = response.data.data;
    console.log("products ==>> ", products);


    for (const product of products) {
      let imageRef = null;
      if (product.image) {
        imageRef = await uploadImageToSanity(product.image);
      }

      const sanityProduct = {
        _type: 'product',
        productName: product.productName,
        category: product.category,
        price: product.price,
        inventory: product.inventory,
        colors: product.colors || [], // Optional, as per your schema
        status: product.status,
        description: product.description,
        image: imageRef ? {
          _type: 'image',
          asset: {
            _type: 'reference',
            _ref: imageRef,
          },
        } : undefined,
      };

      await client.create(sanityProduct);
    }

    console.log('Data migrated successfully!');
  } catch (error) {
    console.error('Error in migrating data ==>> ', error);
  }
}

importData();
```

When you show all product in page you import all product and when you want to get some product you assign[0..3] its means 3 4 product in one page

```
import { defineQuery } from "next-sanity";

 export const allproducts = defineQuery (`

       *[_type == "product"] {
       _id,
       name,
       description,
       price,
      discountPercentage,
      priceWithoutDiscount,
      rating,
      ratingCount,
      tags,
      sizes,
       "imageUrl": image.asset->url
       }
     `)

     export const fourPro = defineQuery (`

         *[_type== "product"] [0..3]{
         _id,
         name,
         description,
         price,
        discountPercentage,
        priceWithoutDiscount,
        rating,
        ratingCount,
        tags,
        sizes,
         "imageUrl": image.asset->url
         }
       `)
```

Project Id assign , dataset , usedCdn and api Version

```
import { createClient } from "next-sanity";

const client = createClient({

    projectId:"fuocr4mh",
    dataset:"production",
    useCdn:true,
    apiVersion: "2023-01-01",
})


export async function sanityFetch({query,params = {}}: {query:string, params?: any }){
    return await client.fetch(query,params)
 }
```

Types assign productSchema and import productschema from ./product

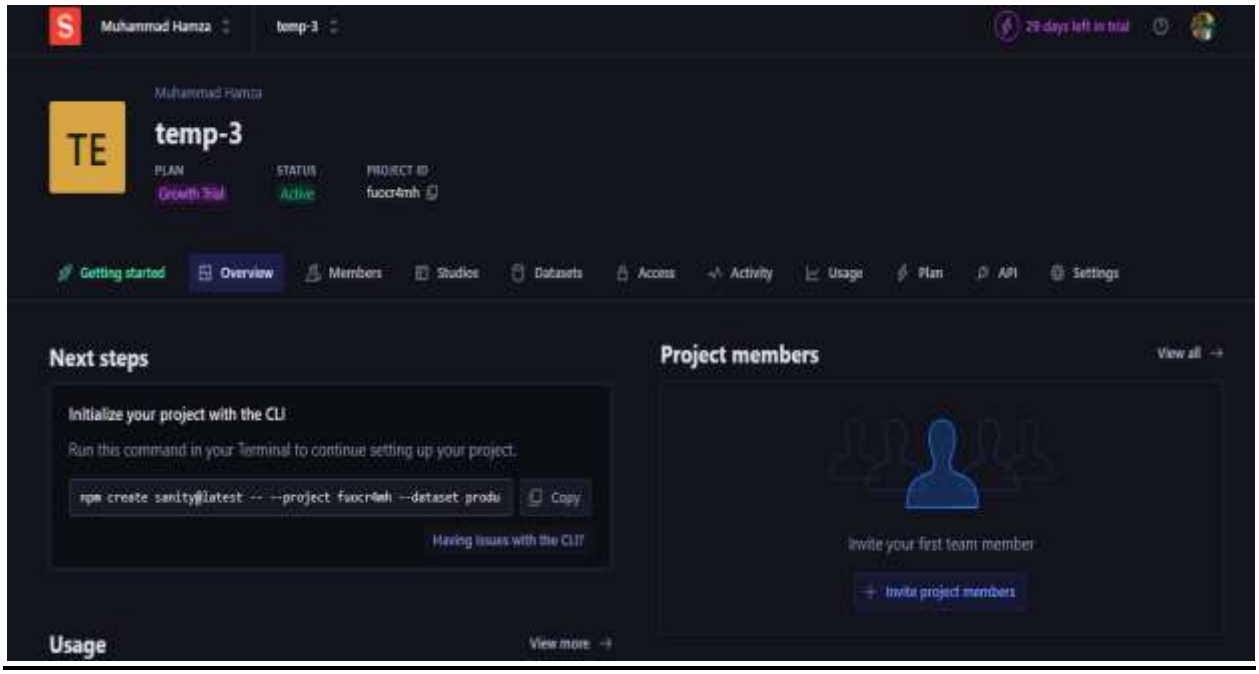
```
import { type SchemaTypeDefinition } from 'sanity'
import { productSchema } from './product'

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

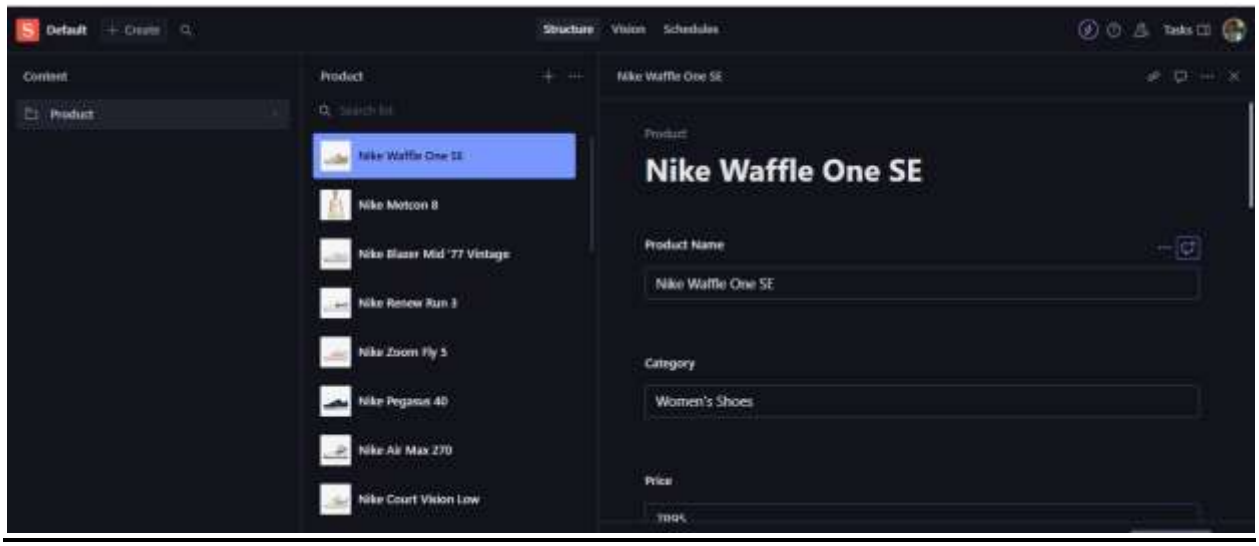
copy product Schema from link:

```
export const productSchema = {
    name: 'product',
    title: 'Product',
    type: 'document',
    fields: [
      {
        name: 'productName',
        title: 'Product Name',
        type: 'string',
      },
      {
        name: 'category',
        title: 'Category',
        type: 'string',
      },
      {
        name: 'price',
        title: 'Price',
        type: 'number',
      },
      {
        name: 'inventory',
        title: 'Inventory',
        type: 'number',
      },
      {
        name: 'colors',
        title: 'Colors',
        type: 'array',
        of: [{ type: 'string' }],
      },
      {
        name: 'status',
        title: 'Status',
        type: 'string',
      },
      {
        name: 'image',
        title: 'Image',
        type: 'image', // Using Sanity's image type for image field
        options: {
          hotspot: true,
        },
      },
      {
        name: 'description',
        title: 'Description',
        type: 'text',
      },
    ],
  }
```

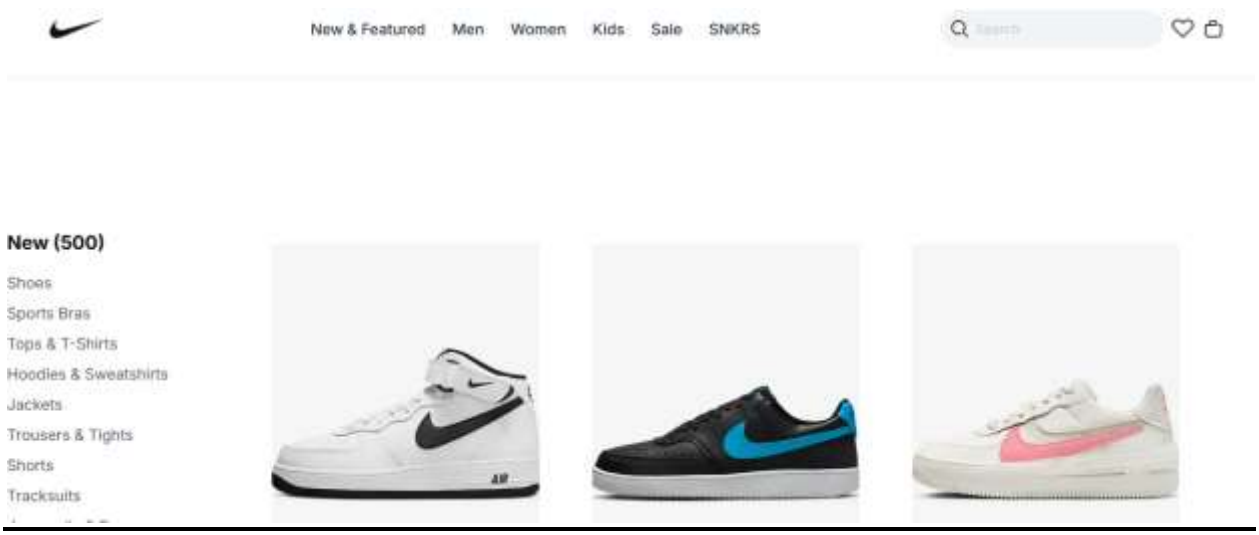
Get link From Overview:



Sanity data integration :



Fetch Result:



Day 3 Checklist:

Self-Validation Checklist:

API Understanding: ✔

Schema Validation: ✔

Data Migration: ✔

API Integration in Next.js: ✔

Submission Preparation: ✔