API Integration & Data Migration

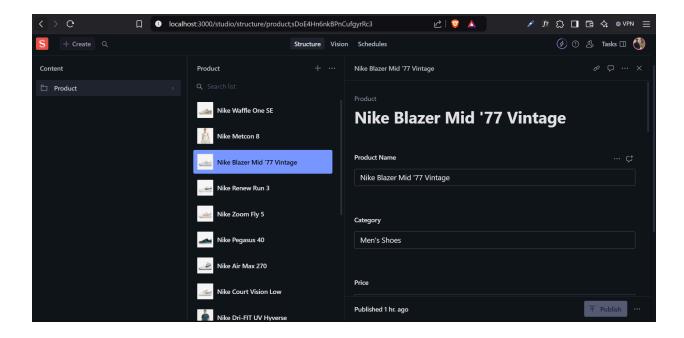
Products API Integration:

- I reviewed APIs given in the document and also explored the whole document to know the process.
- Then I made a migration file to fetch data from API and insert that data into Sanity CMS.

```
us migration.mjs U X
                                                                                                                          EXPLORER
                               scripts > Js migration.mjs >
FIGMADESIGN
                                     async function uploadImageToSanity(imageUrl) {
> 🧰 public
> a sanity
                                36

✓ Im scripts

   us migration.mjs
                                     async function importData() {
> 🐂 types
                                         console.log('migrating data please wait...');
  eslintrc.json
                                        // API endpoint containing car data
  .gitignore
                                        const response = await axios.get('https://template-03-api.vercel.app/api/
  next-env.d.ts
                                         products');
                                         const products = response.data.data;
  N next.config.mjs
                                         console.log("products ==>> ", products);
  postcss.config.mjs
  README.md
                                           let imageRef = null;
                                           if (product.image) {
  sanity.cli.ts
                                             imageRef = await uploadImageToSanity(product.image);
  sanity.config.ts
  tailwind.config.ts
  tsconfig.json
                                            const sanityProduct = {
                                              _type: 'product',
                                              productName: product.productName
```



Schema Adjustments:

 I adjusted my sanity schema because the products that are provided in API are different from my products schema, so i made some changes.

```
EXPLORER
                                                                                                                                        % □ ...
                                        sanity > schemaTypes > rs products.ts > [2] productSchema > \cancel{P} fields
                                               export const productSchema = {
       > 📭 .next
                                                   name: 'product',
       > ||| app
                                                    type: 'document',
        > 📑 data
                                                    fields: [
        > node_modules
                                                             name: 'productName',
        > 🧓 public
                                                            title: 'Product Name',
type: 'string',
        B
        > 📴 lib

✓ □ schemaTypes

           index.ts
                                                             name: 'category',
                                                             title: 'Category',
           rs env.ts
                                                             type: 'string',
           structure.ts
        > In scripts
                                                             name: 'price',
        > 📑 types
                                                             title: 'Price',
                                                             type: 'number',
          eslintrc.json
          .gitignore
                                                             name: 'inventory',
       OUTLINE
                                                             title: 'Inventory',
```

```
EXPLORER
                                                                                                                                                                                 rs products.ts U X
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    ţη II ···
                                                                                                                                                                                    sanity > schemaTypes > 
substituting products.ts > 
substituting productSchema > 
productS
                                                                                                                                                                                                                     export const productSchema = {
                                   > 📭 .next
                                   > |||| app
                                                                                                                                                                                                                                        fields: [
                                   > III components
                                                                                                                                                                                                                                                                               name: 'status',
                                   🗦 📑 data
                                                                                                                                                                                                                                                                               title: 'Status',
                             Run and Debug (Ctrl+Shift+D)
                                                                                                                                                                                                                                                                                type: 'string',
品
                                      > 🏴 lib
                                                                                                                                                                                                                                                                               name: 'image',
                                        title: 'Image',
 type: 'image', // Using Sanity's image type for image field
                                                                                                                                                                                                                                                                               options: {
                                                                                                                                                                                                                                                                                                 hotspot: true,
                                                 structure.ts
                                  > In scripts
                                   > 📑 types
                                                                                                                                                                                                                                                                                name: 'description',
                                           뷰 .env
                                                                                                                                                                                                                                                                               title: 'Description',
                                                                                                                                                                                                                                                                               type: 'text',
                                           eslintrc.json
                                            .gitignore
                                 OUTLINE
                                  TIMELINE
```

API Creation for products fetching:

• Now, I created one endpoint called /products where I fetched all the products that are stored in my Sanity CMS.

```
route.ts U X
       EXPLORER
                                                                                                                                   % □ ··
      ✓ FIGMADESIGN
                                      app > api > products > ₁ѕ route.ts > � GET > ❷ query
                                             import { client } from "@/sanity/lib/client";
       > 📭 .next
       ∨ 📠 арр
                                             export async function GET(request: Request) {

✓  api \ products

        > (productName)
                                                          *[_type == "product"]{
                                                          id.
        > checkout
                                                         productName,
B
                                                         category,
        > contact
        > login
                                                         inventory,
> products
                                       12
                                                         colors,
        > register
                                                          status.
        > 🖿 studio
                                                          "imageUrl": image.asset->url, // Fetch the URL for the image
          ★ favicon.ico
                                                          description
             g l o b a l s .css
          layout.tsx
          🎡 page.tsx
                                                     const data = await client.fetch(query);
       > 🔛 components
(8)
       > 📙 data
       OUTLINE
                                                  } catch (error) {
       TIMELINE
```

Fetching Products to Frontend:

• Then, I fetched that API I created on the frontend to display all the products. Below is the image showcasing how I did.

```
FIGMADESIGN
Q
                                        import Image from 'next/image'

✓ I components

                                        import React from 'react'
       widgets
                                         import Filter from '@/public/images/Filters.png'
          Best tsx
                                        import Cateogories from '../ui/Cateogories'
                                         import FilterController from '../ui/FilterController'
import { IoIosArrowDown } from 'react-icons/io'
          ∰ Cart.tsx
          Checkout.tsx
$
                                         import Product from '../ui/Product'
          Dont.tsx
                                         import { ProductType } from '@/types/product'
          Essentials.tsx
          Featured.tsx
          Gear.tsx
                                         const Products = async () => {
Help.tsx
                                             const res = await fetch('http://localhost:3000/api/products');
          Links.tsx
                                             const data: ProductType[] = await res.json();
          Login.tsx
         Might.tsx
                                                 <div className='w-full flex flex-col gap-4 md:px-10 px-4 py-10 over'>
          Register.tsx
                                                     📑 data
                                                         <div className='text-[24px] '>New (500)</div>
       > node modules
                                                         <div className='md:flex hidden items-center gap-6 text-[16px]'>
                                                             <div className='flex items-center gap-1'>
       > 🧓 public
                                                                <div>Hide Filters</div>
                                                                 <div>
      TIMELINE
```

Individually Fetching Products With Dynamic API Route:

• I made a dynamic API route to fetch a product that matches the name of the product requested by the user.

```
EXPLORER
                                     route.ts U X
                                                                                                                                 ţე Ш ...
       FIGMADESIGN
                                      app > api > products > [productName] > ▼ route.ts > ❤ GET
                                            import { client } from "@/sanity/lib/client";
       > 📭 .next

✓  api\products

                                                request: Request.

✓ IproductName

                                                 { params }: { params: Promise<{ productName: string }> }
           route.ts
                                                 const productName = (await params).productName
        > 📭 cart
                                                    console.log("API Id: ", productName);
        > m checkout
                                                     if (!productName) {
        > contact
                                                         throw new Error("Id Not Found")
> 🖿 login
        > products
        > register
                                                     const query = `
                                                         *[_type == "product" && productName == $productName][0] {
        > 🖿 studio
          ★ favicon.ico
                                                             id.
                                                             productName,

∃ globals.css

                                                             category,
          layout.tsx
                                                             price,
          🏶 page.tsx
                                                             inventory,
(2)
       > Figure components
                                                             colors,
                                                             status.
                                                             "imageUrl": image.asset->url,
```

```
<sub>ເນື</sub> 🖽 ...
Explorer (Ctrl+Shift+E)
✓ FIGMADESIGN
                                     export async function GET(
 > next
 ∨ 📠 арр
                                               const data = await client.fetch(query, { productName });
  29

✓ ☐ [productName]

                                              if (!data) {
                                                   throw new Error("Product Not Found")
     route.ts
  > 📭 cart
  > checkout
                                              return new Response(
  > contact
                                                  JSON.stringify(data),
  > 🖿 login
                                                   { status: 200, headers: { "Content-Type": "application/json" } }
  > products
  > register
                                           } catch (error) {
  > studio
                                              console.error(error);
    🛨 favicon.ico
    ∃ globals.css
                                              return new Response(
    layout.tsx
                                                  JSON.stringify({ message: "Failed to fetch product" }),
                                                   { status: 500, headers: { "Content-Type": "application/json" } }
    page.tsx
 OUTLINE
 TIMELINE
                                                                             Ln 29, Col 1 Spaces: 4 UTF-8 CRLF {} TypeScript @ Go Live
```

Individually Displaying Products on Dynamic Page:

 After creating an API for an individual product I made an route for a dynamic page and fetched that API there to display desired product.

```
ξ'n II ...
       EXPLORER

₱ page.tsx U X

ф
      ✓ FIGMADESIGN
                                      app > products > [productName] > ∰ page.tsx > [] ProductPage
                                       1 import Detail from '@/components/ui/Detail';
                                             import { ProductType } from '@/types/product';
       V 📠 арр
                                            import React from 'react';
        > api\products
        > 📭 cart
                                            const ProductPage = async ({ params }: { params: { productName: string } }) => {
        > checkout
                                                console.log(params.productName)
        > contact
                                                 const res = await fetch(`http://localhost:3000/api/products/${params.
        > login
                                                productName}`);
品

✓ I products

                                                 if (!res.ok) {

✓ IproductName

                                                     console.error(errorData.message);
                                                     return <div>Error: {errorData.message}</div>;
        > register
        > 🖿 studio
                                                 const data: ProductType = await res.json();
          ★ favicon.ico
                                                 console.log(data);
          ∃ globals.css
          🔅 layout.tsx
          page.tsx
                                                     <div className='md:px-10 px-4 py-10 w-full'>
       > Expression components
                                       20
                                                         ∢Detail
                                                             title={data.productName}
       OUTLINE
                                                             description={data.description}
```