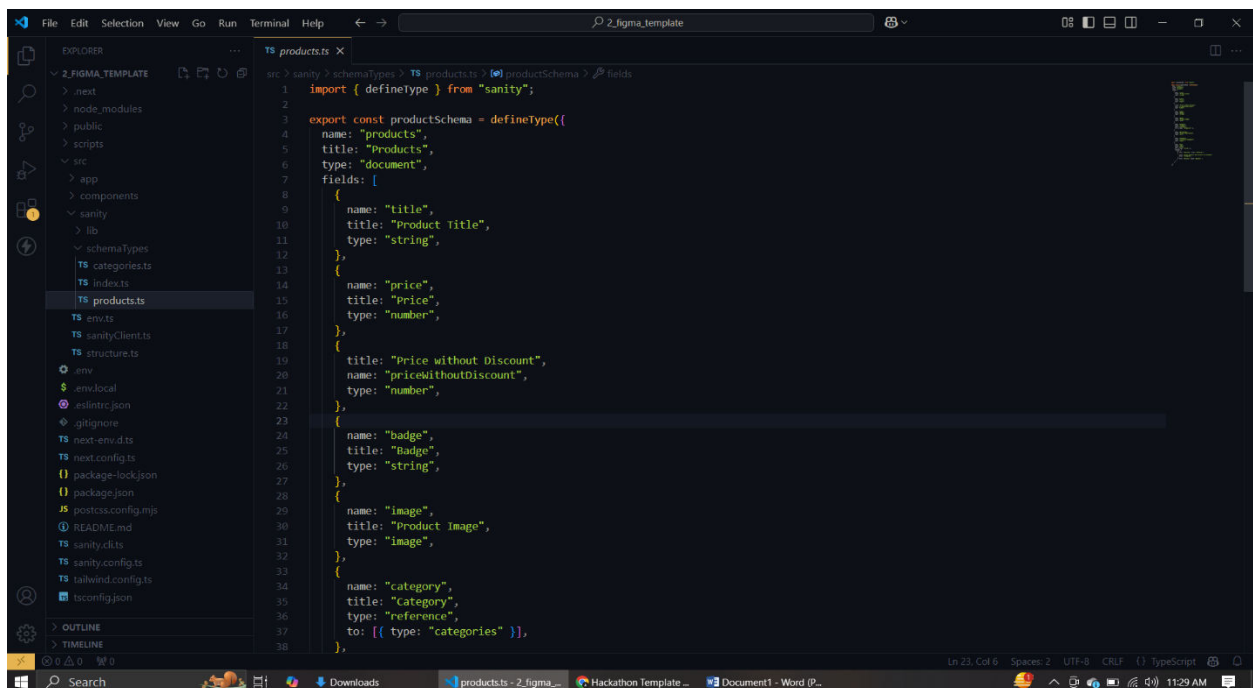


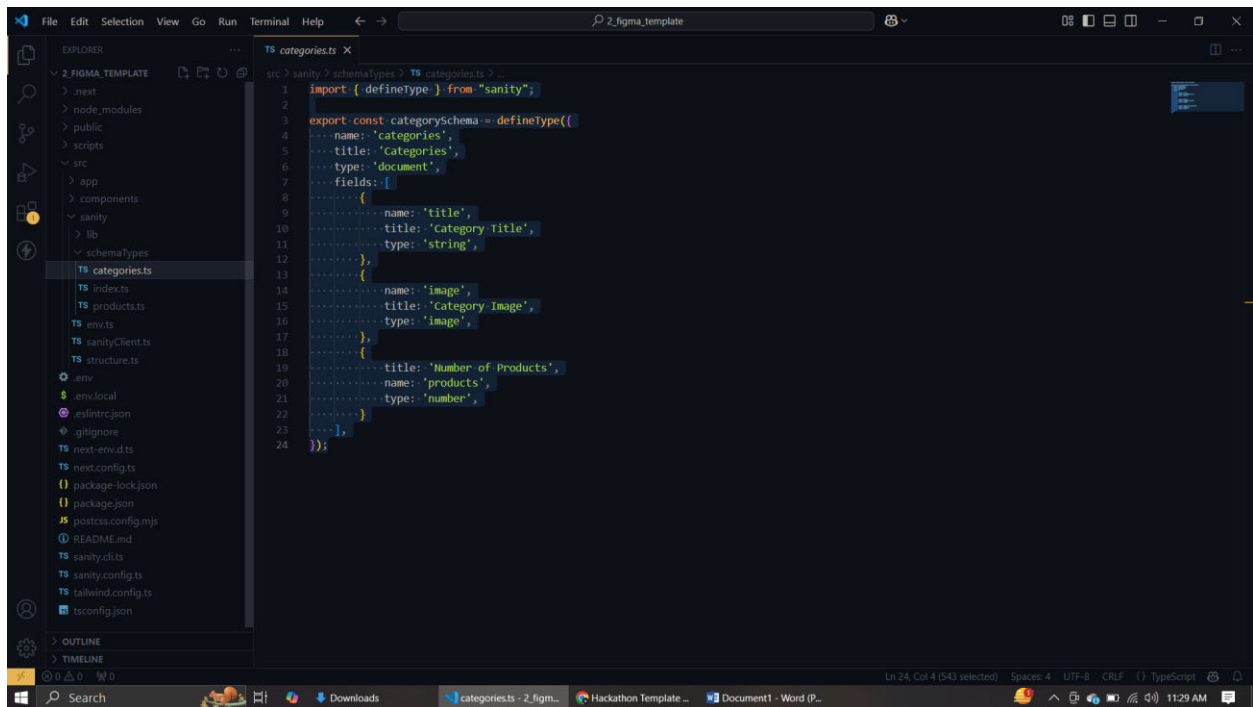
Hackathon-3 (Day-3)

API Integration & Data Migration

1. Creating Products.ts & Categories.ts files in SANITY after it's Installation and adding the schema code in both files:

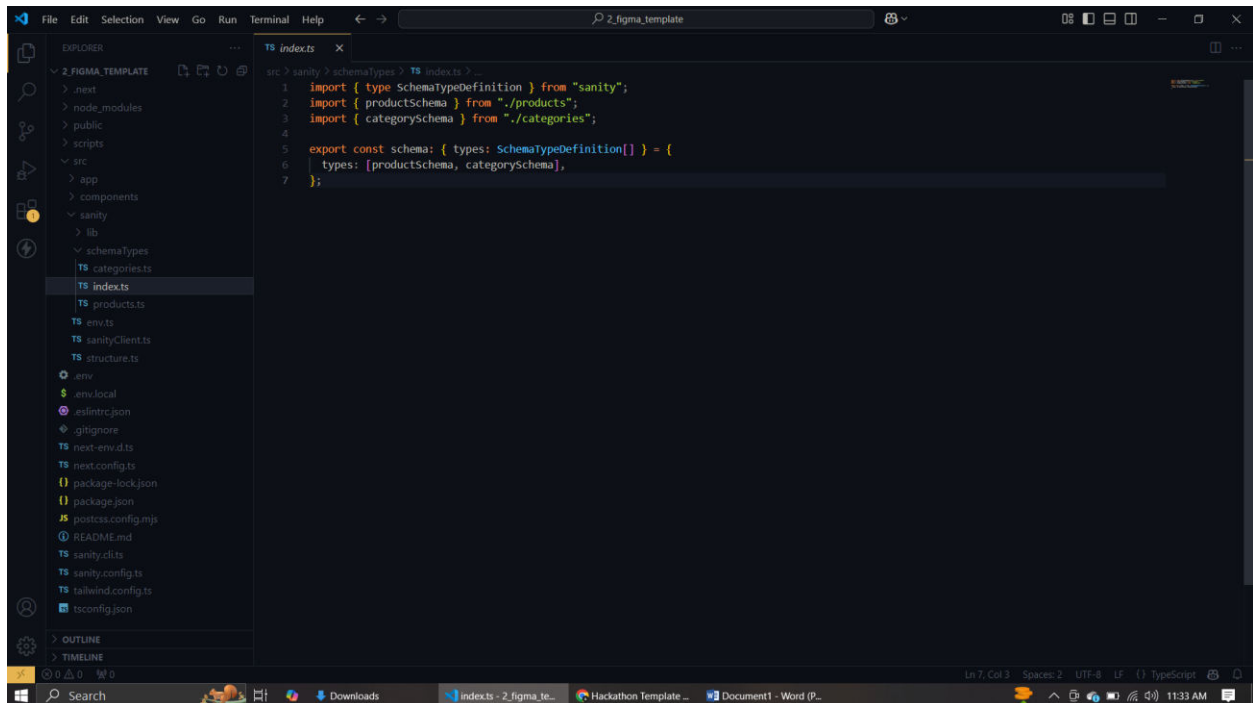


```
1 import { defineType } from "sanity";
2
3 export const productSchema = defineType({
4   name: "products",
5   title: "Products",
6   type: "document",
7   fields: [
8     {
9       name: "title",
10      title: "Product Title",
11      type: "string",
12    },
13    {
14      name: "price",
15      title: "Price",
16      type: "number",
17    },
18    {
19      title: "Price without Discount",
20      name: "priceWithoutDiscount",
21      type: "number",
22    },
23    {
24      name: "badge",
25      title: "Badge",
26      type: "string",
27    },
28    {
29      name: "image",
30      title: "Product Image",
31      type: "image",
32    },
33    {
34      name: "category",
35      title: "Category",
36      type: "reference",
37      to: [{ type: "categories" }],
38    },
39  ],
40 });
```



```
src > sanity > schemaTypes > TS categories.ts > ...
1 import { defineType } from "sanity";
2
3 export const categorySchema = defineType({
4   name: 'categories',
5   title: 'Categories',
6   type: 'document',
7   fields: [
8     {
9       name: 'title',
10      title: 'Category Title',
11      type: 'string',
12    },
13    {
14      name: 'image',
15      title: 'Category Image',
16      type: 'image',
17    },
18    {
19      name: 'Number of Products',
20      title: 'products',
21      type: 'number',
22    }
23 ],
24 });
```

2. Importing Schemas in index.ts:

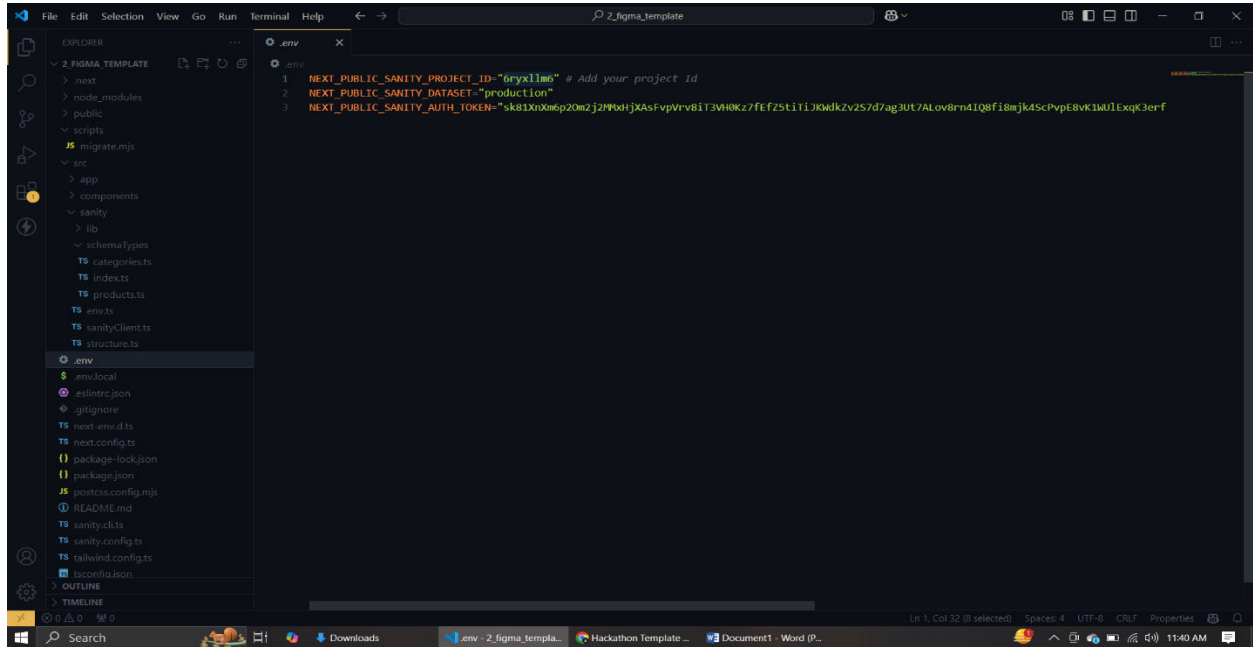


```
src > sanity > schemaTypes > TS index.ts > ...
1 import { type SchemaTypeDefinition } from "sanity";
2 import { productsSchema } from "../products";
3 import { categorySchema } from "../categories";
4
5 export const schema: { types: SchemaTypeDefinition[] } = {
6   types: [productsSchema, categorySchema],
7 };
8
```

Data Migration Script

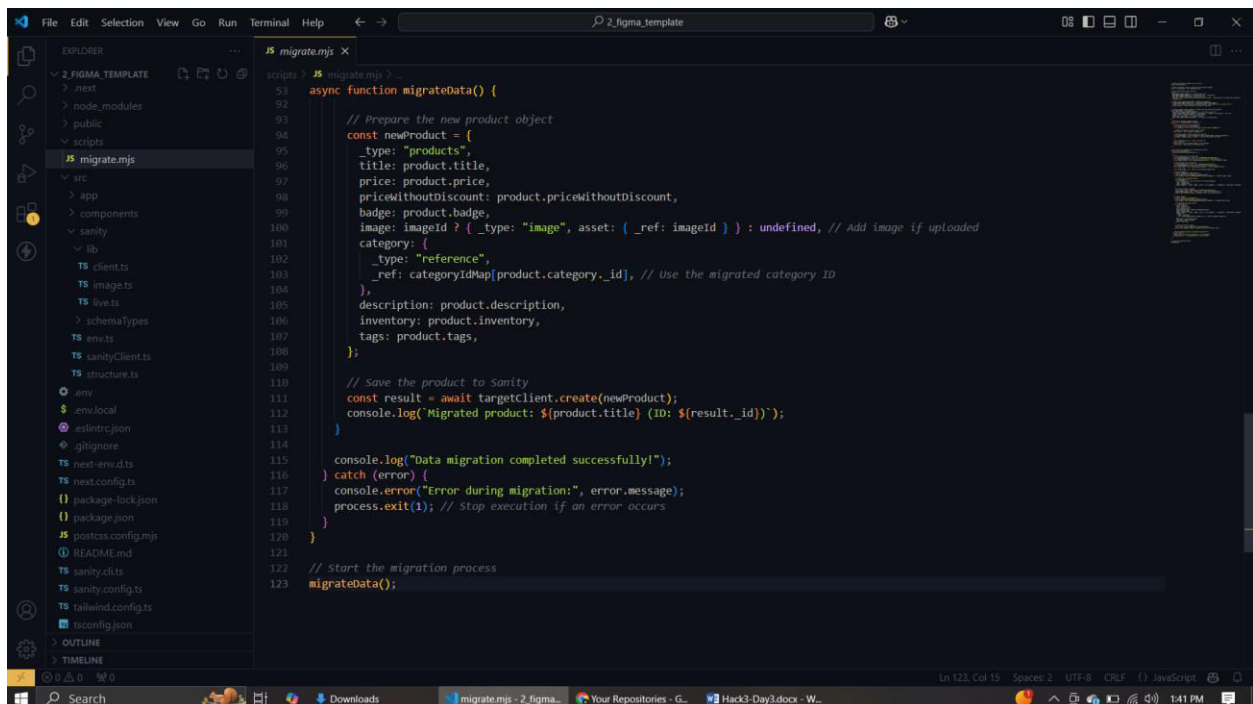
This script automates the process of transferring data from the provided REST APIs to your Sanity dataset.

3. Creating `.env` file & setting up Environment variables:



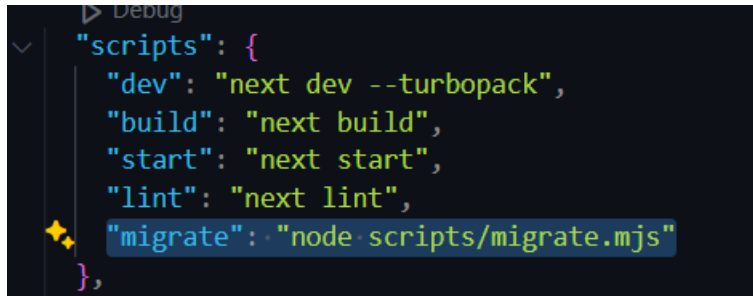
```
1 NEXT_PUBLIC_SANITY_PROJECT_ID="6ryxllm6" # Add your project id
2 NEXT_PUBLIC_SANITY_DATASET="production"
3 NEXT_PUBLIC_SANITY_AUTH_TOKEN="sk81XnXm6p20m2j2MxHjXAsFvpVrv81T3V4R0Kz7FEfZ5ti1lJKWdkzV2S/d7ag3U7ALov8rn4lQ8f18mjK4ScPvpE8vK1WU1ExqK3erf"
```

4. Creating `Migrate.mjs` inside the script folder & adding the migration code:



```
1 // Prepare the new product object
2 const newProduct = {
3   _type: "products",
4   title: product.title,
5   price: product.price,
6   priceWithoutDiscount: product.priceWithoutDiscount,
7   badge: product.badge,
8   image: imageId ? { _type: "image", asset: { _ref: imageId } } : undefined, // Add image if uploaded
9   category: {
10     _type: "reference",
11     _ref: categoryIdMap[product.category_id], // Use the migrated category ID
12   },
13   description: product.description,
14   inventory: product.inventory,
15   tags: product.tags,
16 };
17
18 // Save the product to Sanity
19 const result = await targetClient.create(newProduct);
20 console.log('Migrated product: ${product.title} (ID: ${result._id})');
21
22 console.log("Data migration completed successfully!");
23 catch (error) {
24   console.error("Error during migration:", error.message);
25   process.exit(1); // Stop execution if an error occurs
26 }
27
28 // Start the migration process
29 migrateData();
```

5. Adding the **"migrate" : "node scripts/migrate.mjs"** inside of scripts in the **"Package.json"** file:



```
Debug
{
  "scripts": {
    "dev": "next dev --turbo",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "migrate": "node scripts/migrate.mjs"
  },
}
```

6. Installing the **"dotenv"** package before running the script:

-npm install dotenv

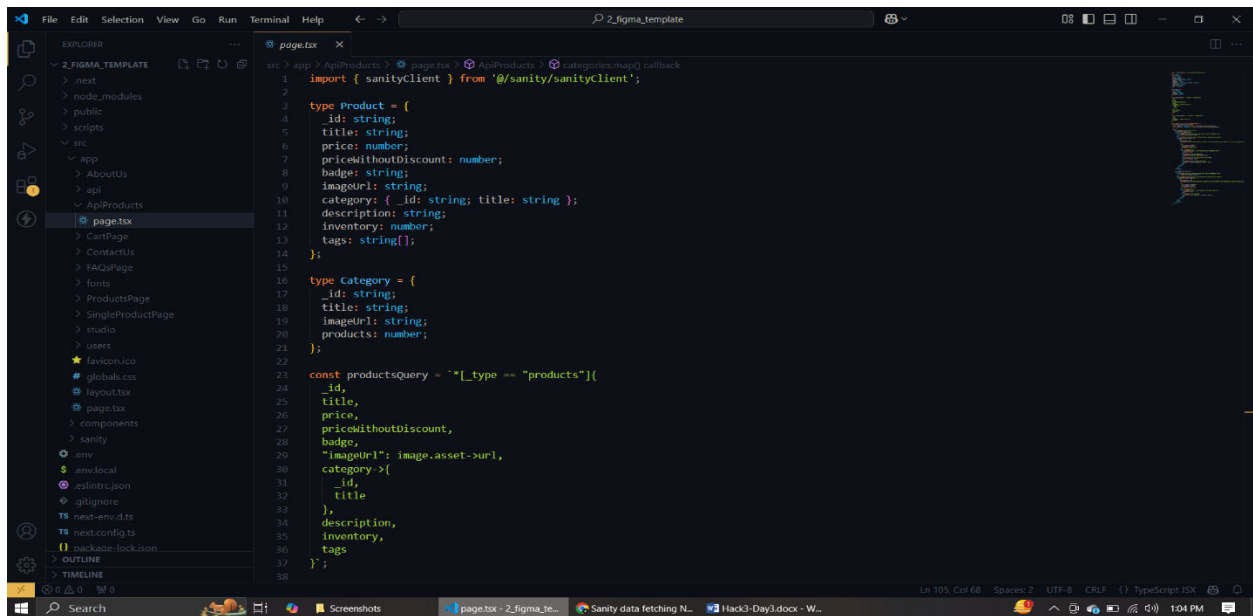
7. Running the **"npm run migrate"** command:

-This will insert the data from the Rest API to Sanity Studio.

Data Display On Front-End

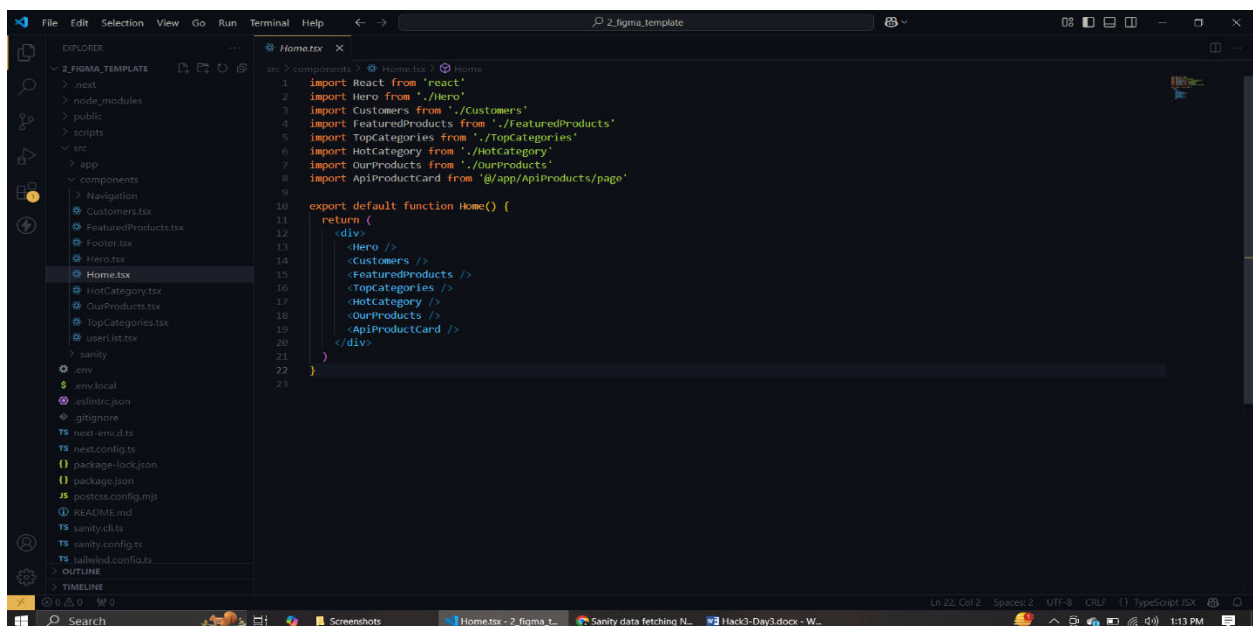
8. Creating App Router “ApiProducts” in src folder & adding code in it:

-src/app/ApiProducts/page.tsx



```
1 import { sanityClient } from '@sanity/sanityClient';
2
3 type Product = {
4   id: string;
5   title: string;
6   price: number;
7   priceWithoutDiscount: number;
8   badge: string;
9   imageUrl: string;
10  category: { id: string; title: string };
11  description: string;
12  inventory: number;
13  tags: string[];
14 };
15
16 type Category = {
17   id: string;
18   title: string;
19   imageUrl: string;
20   products: number;
21 };
22
23 const productsQuery = `*[_type == "products"]{
24   id,
25   title,
26   price,
27   priceWithoutDiscount,
28   badge,
29   "imageUrl": image.asset->url,
30   category->{
31     id,
32     title
33   },
34   description,
35   inventory,
36   tags
37 }`;
38
```

9. Importing “ApiProductCard” in Home.tsx component to show it on Frontend:



```
1 import React from 'react';
2 import Hero from './Hero';
3 import Customers from './Customers';
4 import FeaturedProducts from './FeaturedProducts';
5 import TopCategories from './TopCategories';
6 import HotCategory from './HotCategory';
7 import OurProducts from './OurProducts';
8 import ApiProductCard from '@app/ApiProducts/page';
9
10 export default function Home() {
11   return (
12     <div>
13       <Hero />
14       <Customers />
15       <FeaturedProducts />
16       <TopCategories />
17       <HotCategory />
18       <OurProducts />
19       <ApiProductCard />
20     </div>
21   );
22 }
23
```

10.Display On The Font-End:

