Day 3 - API Integration Report - G. Ecommerce [Heckto]

API Integration Process:

First of all get the Api from sanity project in which i wanna insert or migrate data

1. Configuring Environment Variables

Save the copied API token in your `.env.local` file

2 Add in client folder

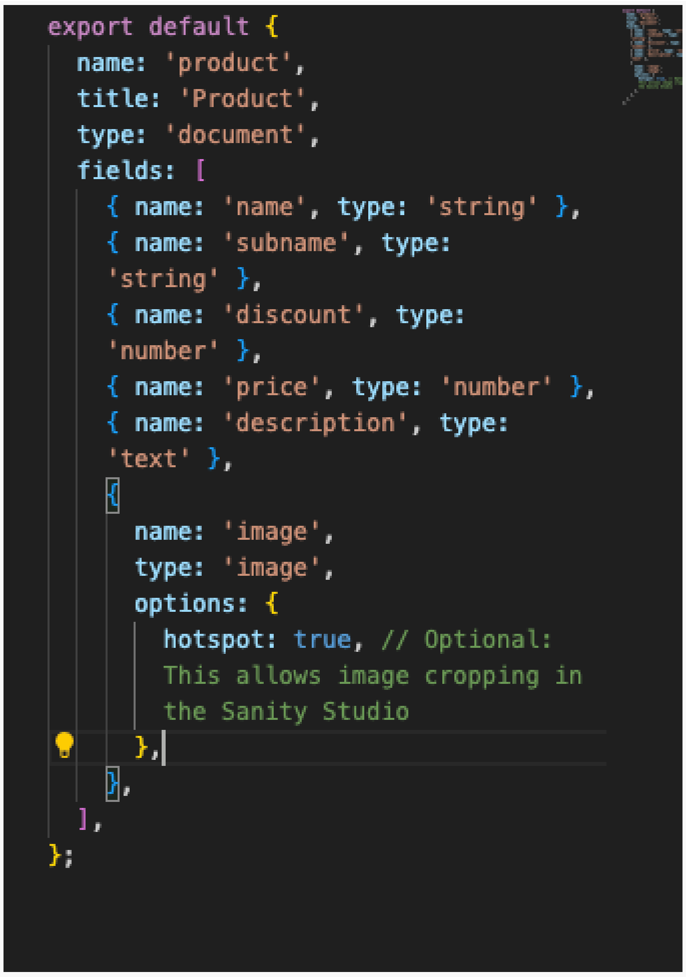
Add the api in your client folder or simply copy and paste this in

code in `sanity/lib/client.ts`

```
//sanity/lib/client.ts

import { createClient } from 'next-sanity'

import { apiVersion, dataset, projectId } from '../env'


export const client = createClient({

 projectId,Dataset,

apiVersion,token:process.env.SANITY_API,useCdn:true,

})
```

3. Defining the Product Schema

Create a file named `product.ts` in your schema folder and paste the following code or make your own: `sanity/schemaTypes//product.ts`



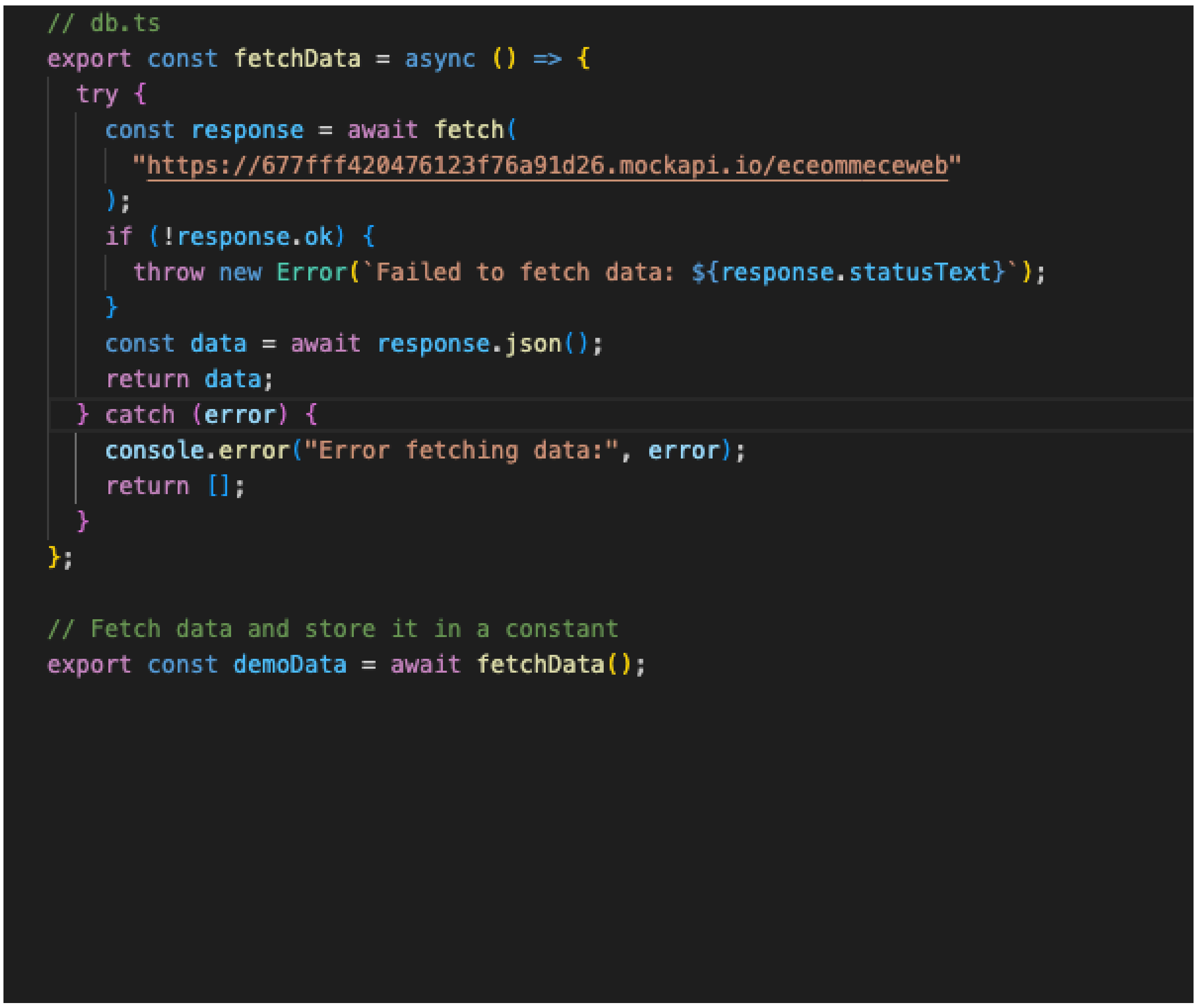
```
export default {
  name: 'product',
  title: 'Product',
  type: 'document',
  fields: [
    { name: 'name', type: 'string' },
    { name: 'subname', type:
    'string' },
    { name: 'discount', type:
    'number' },
    { name: 'price', type: 'number' },
    { name: 'description', type:
    'text' },
    {
      name: 'image',
      type: 'image',
      options: {
        hotspot: true, // Optional:
        This allows image cropping in
        the Sanity Studio
      },
    },
  ],
};
```

Then, update your `sanity/schemaTypes/index.ts` file to include the new product schema:

```
import { type SchemaTypeDefinition } from 'sanity'


import product from './product'

export const schema: { types: SchemaTypeDefinition[] } = {

 types: [product],

}
```

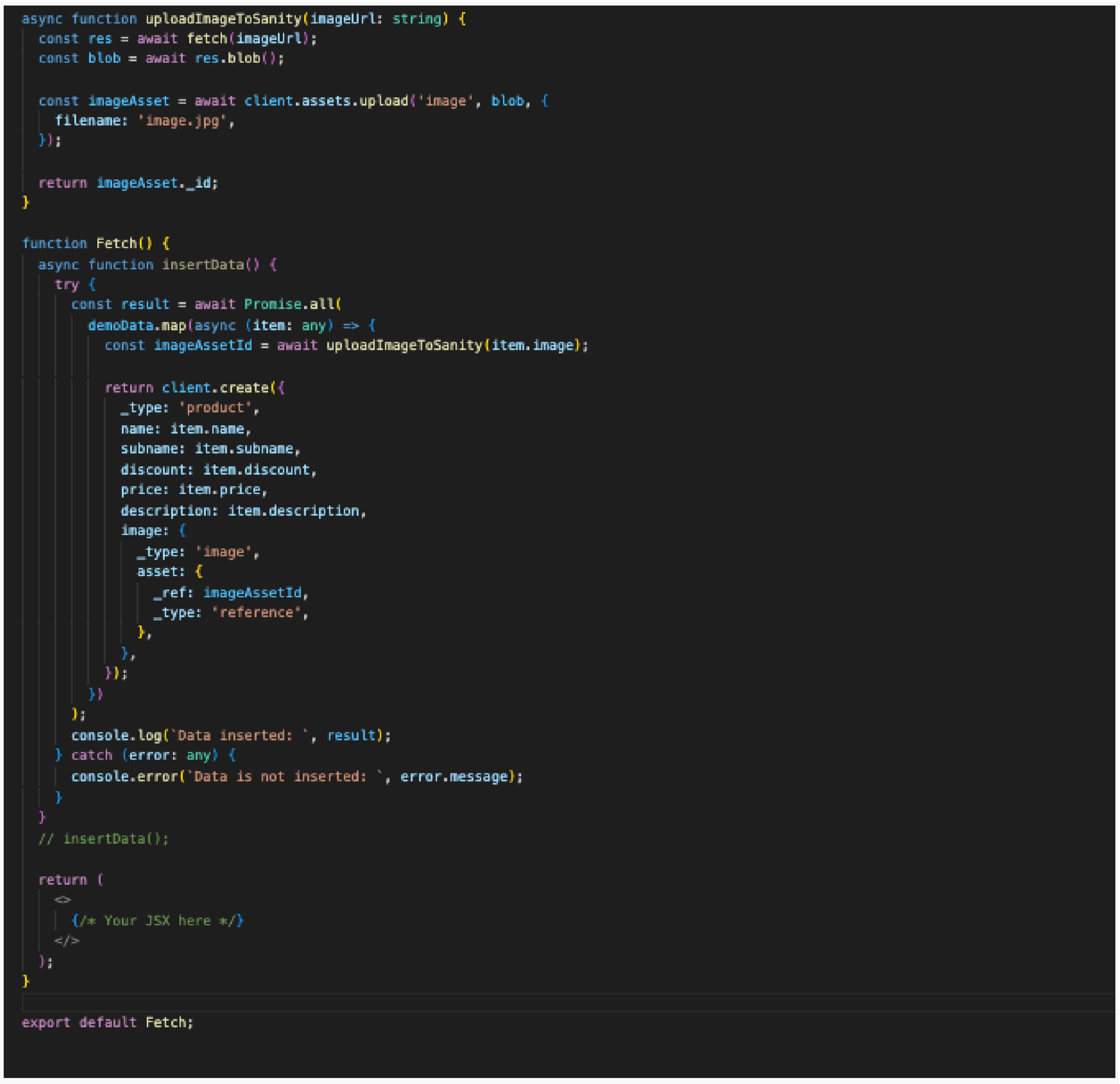
Creating a Database | Mock Data

Next, create a `db.ts` file to create or fetch mock data:



```
// db.ts
export const fetchData = async () => {
  try {
    const response = await fetch(
      "https://677fff420476123f76a91d26.mockapi.io/eceommeceweb"
    );
    if (!response.ok) {
      throw new Error(`Failed to fetch data: ${response.statusText}`);
    }
    const data = await response.json();
    return data;
  } catch (error) {
    console.error("Error fetching data:", error);
    return [];
  }
};

// Fetch data and store it in a constant
export const demoData = await fetchData();
```

4. Building the Insert Page

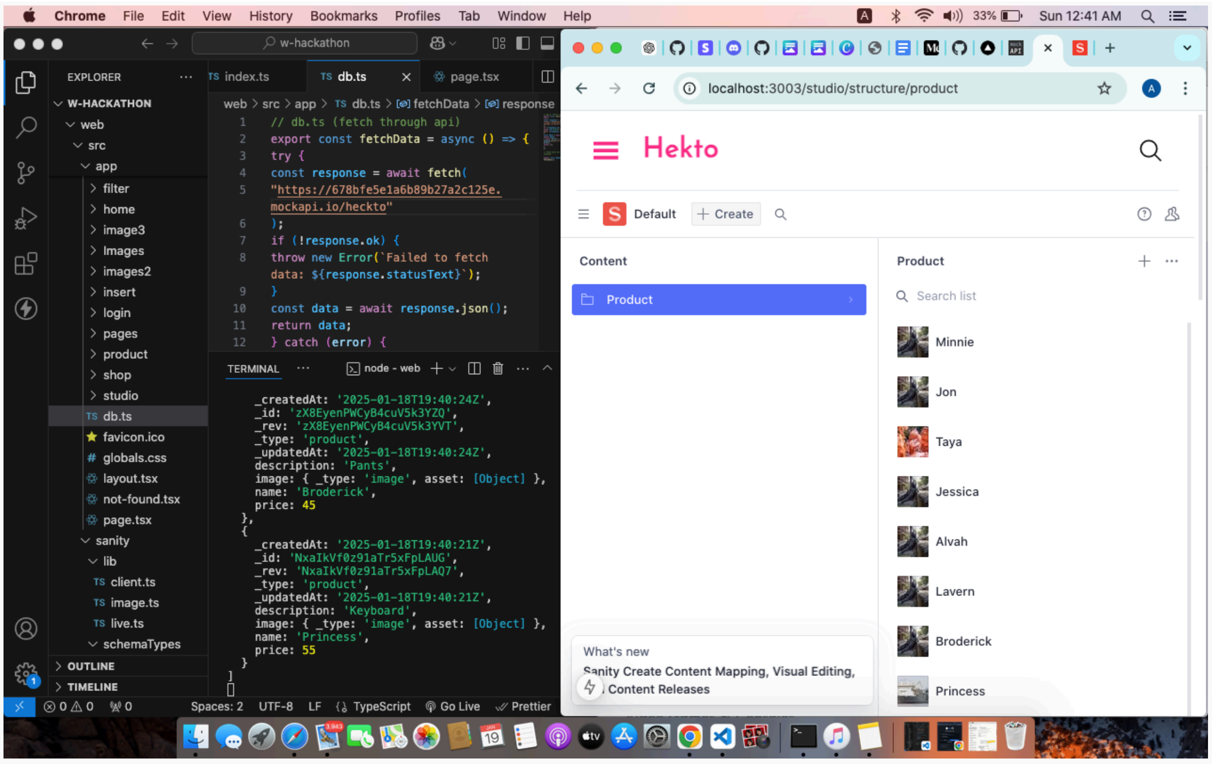


```
async function uploadImageToSanity(imageUrl: string) {
  const res = await fetch(imageUrl);
  const blob = await res.blob();

  const imageAsset = await client.assets.upload('image', blob, {
    filename: 'image.jpg',
  });

  return imageAsset._id;
}

function Fetch() {
  async function insertData() {
    try {
      const result = await Promise.all(
        demoData.map(async (item: any) => {
          const imageAssetId = await uploadImageToSanity(item.image);

          return client.create({
            _type: 'product',
            name: item.name,
            subname: item.subname,
            discount: item.discount,
            price: item.price,
            description: item.description,
            image: {
              _type: 'image',
              asset: {
                _ref: imageAssetId,
                _type: 'reference',
              },
            },
          });
        })
      );
      console.log(`Data inserted: `, result);
    } catch (error: any) {
      console.error(`Data is not inserted: `, error.message);
    }
  }
  // insertData();

  return (
    <>
      {/* Your JSX here */}
    </>
  );
}

export default Fetch;
```

5. Testing and Verifying the Data

1. Run the development server using the terminal

```
Npm run dev
```



```
// db.ts (fetch through api)
export const fetchData = async () => {
try {
const response = await fetch(
"https://678bfe5e1a6b89b27a2c125e.mockapi.io/heckto"
);
if (!response.ok) {
throw new Error(`Failed to fetch data: ${response.statusText}`);
}
const data = await response.json();
return data;
} catch (error) {
```

```
    _createdAt: '2025-01-18T19:40:24Z',
    _id: 'zX8EyenPWCyB4cuV5k3YZQ',
    _rev: 'zX8EyenPWCyB4cuV5k3YVT',
    _type: 'product',
    _updatedAt: '2025-01-18T19:40:24Z',
    description: 'Pants',
    image: { _type: 'image', asset: [Object] },
    name: 'Broderick',
    price: 45
  },
  {
    _createdAt: '2025-01-18T19:40:21Z',
    _id: 'NxaIkVf0z91aTr5xFpLAUG',
    _rev: 'NxaIkVf0z91aTr5xFpLAQ7',
    _type: 'product',
    _updatedAt: '2025-01-18T19:40:21Z',
    description: 'Keyboard',
    image: { _type: 'image', asset: [Object] },
    name: 'Princess',
    price: 55
  }
]
```

Data Migrated Succesfully!