HACKATHON-03
DAY-03

My MarketPlace Name [Honest_Bazar]

A report documenting

- API INTEGRATION PROCESS :

1. Install Sanity Client

First, install the Sanity client library in your project.

npm install @sanity/client

2. Configure the Sanity Client

Create a sanity.js file to configure the client with your Sanity project details (project ID, dataset, and API version).

```
import { createClient } from 'next-sanity'

import { apiVersion} from '../env'

export const client = createClient({
  projectId : process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
  dataset : process.env.NEXT_PUBLIC_SANITY_DATASET,
  apiVersion,
  token: process.env.NEXT_PUBLIC_SANITY_TOKEN,
  useCdn: true,
})
```

3. Fetch Data with GROQ Query

Write a function to fetch data using a GROQ query.

```
import { client } from "@/sanity/lib/client";
import { NextResponse } from "next/server";


export async function GET() {
  try {
    const data = await client.fetch(`*[_type=="product"]{
    _id,
    title,
    "imageUrl" :productImage.asset -> url,
    price,
    tags,
    dicountPercentage,
    description,
    isNew
}`);

    return  NextResponse.json(data, { status: 200 });
  } catch (error) {
    console.error('Error fetching data from Sanity:', error);
    return new NextResponse('Error fetching data', { status: 500 });
  }
}
```

- ADJUSTMENTS MADE TO SCHEMA :

```
import { defineType } from "sanity"

export const product = defineType({
    name: "product",
    title: "Product",
    type: "document",
    fields: [
        {
            name: "title",
            title: "Title",
            validation: (rule) => rule.required(),
            type: "string"
        },
        {
            name:"description",
            type:"text",
            validation: (rule) => rule.required(),
            title:"Description",
        },
        {
            name: "productImage",
            type: "image",
            validation: (rule) => rule.required(),
            title: "Product Image"
        },
        {
            name: "price",
            type: "number",
            validation: (rule) => rule.required(),
            title: "Price",
        },
        {
            name: "tags",
            type: "array",
            title: "Tags",
            of: [{ type: "string" }]
        },
        {
            name:"dicountPercentage",
            type:"number",
            title:"Discount Percentage",
        },
        {
            name:"isNew",
            type:"boolean",
            title:"New Badge",
        }
    ]
```

```
})
```

- MIGRATION STEPS AND TOOLS USED :

1. Sanity Installation

First we have to install sanity by:

npm create sanity@latest

2. Sanity Schema

After the installation Navigate to your schema folder:
If you have a src folder, go to /src/sanity/schemaTypes.
Otherwise, go to /sanity/schemaTypes.

Than place your sanity schema there.

Don't forget to import schema's in your index.ts file

3. Data Migration Script

- Create .env file and add the following variables:

```
NEXT_PUBLIC_SANITY_PROJECT_ID="Your Project"
NEXT_PUBLIC_SANITY_DATASET="production"
NEXT_PUBLIC_SANITY_TOKEN="your Token"
```

- Create migrate.mjs inside of the script folder and put your migarting data there

- Open `package.json` file and add the following code inside of scripts:

```
    "import-data": "node ./importData.mjs"
```

- **Install**

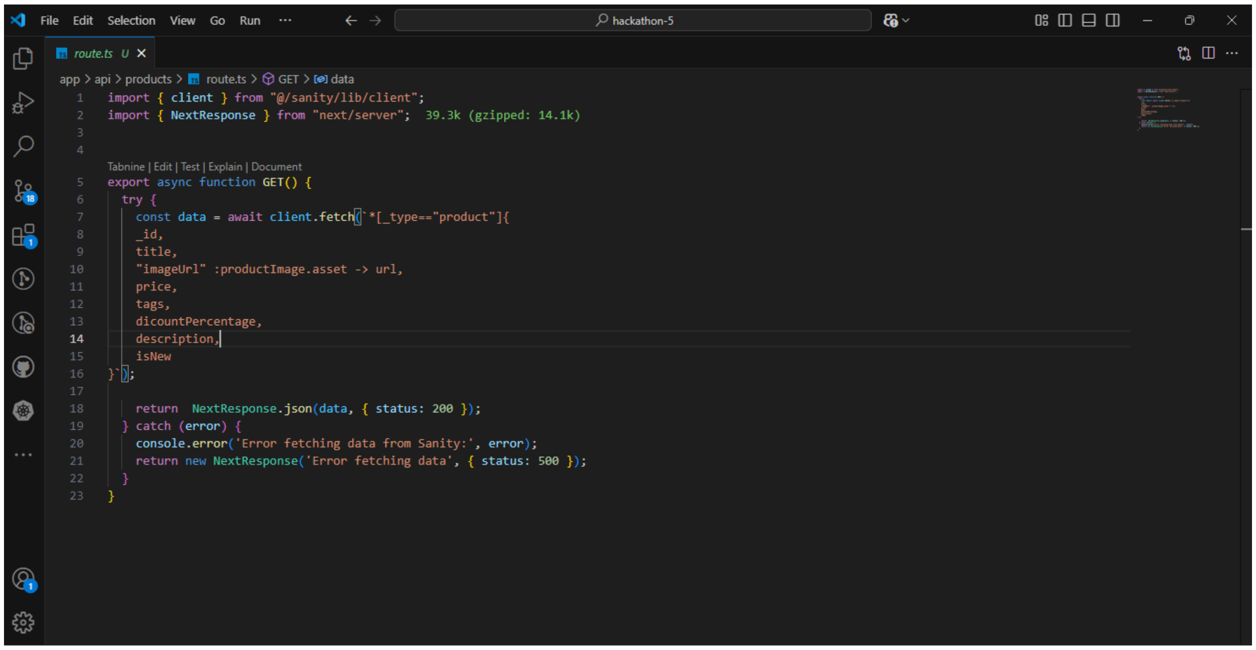
    npm install dotenv

- Now run the command

npm run import

Screenshots

API calls.

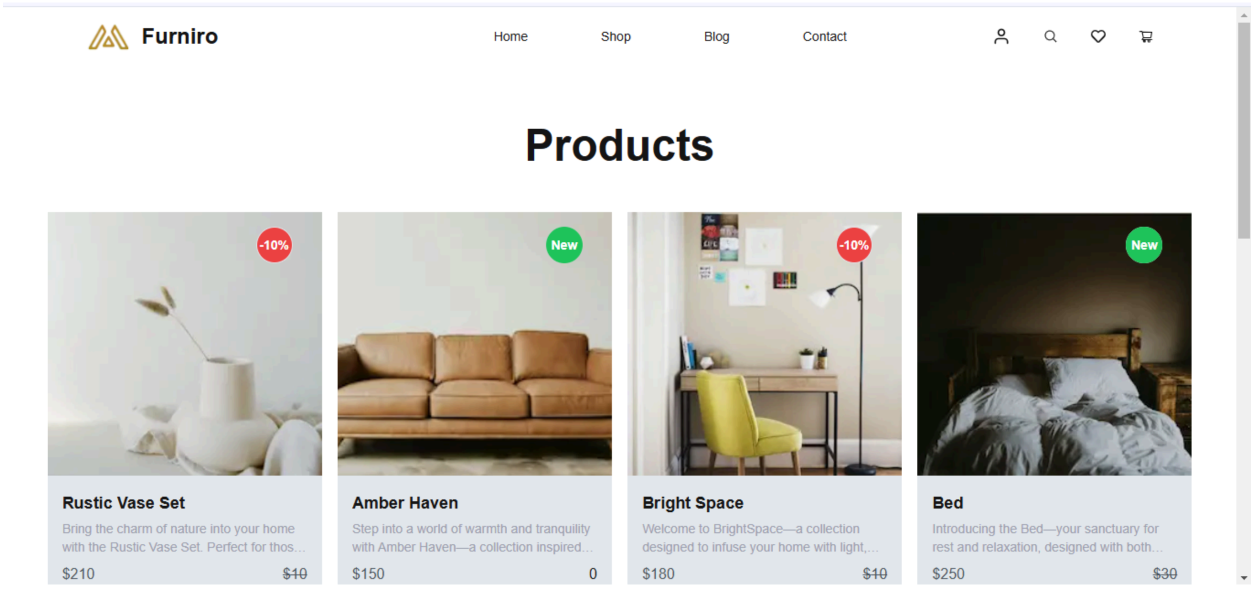

File Edit Selection View Go Run ···
hackathon-5
route.ts U
app > api > products > route.ts > GET > data
Tabnine | Edit | Test | Explain | Document

```
import { client } from "@/sanity/lib/client";
import { NextResponse } from "next/server";  39.3k (gzipped: 14.1k)


export async function GET() {
  try {
    const data = await client.fetch(`*[_type=="product"]{
    _id,
    title,
    "imageUrl" :productImage.asset -> url,
    price,
    tags,
    dicountPercentage,
    description,
    isNew
}`);

    return  NextResponse.json(data, { status: 200 });
  } catch (error) {
    console.error('Error fetching data from Sanity:', error);
    return new NextResponse('Error fetching data', { status: 500 });
  }
}
```

Data successfully displayed in the frontend.


Furniro

Home Shop Blog Contact

Products

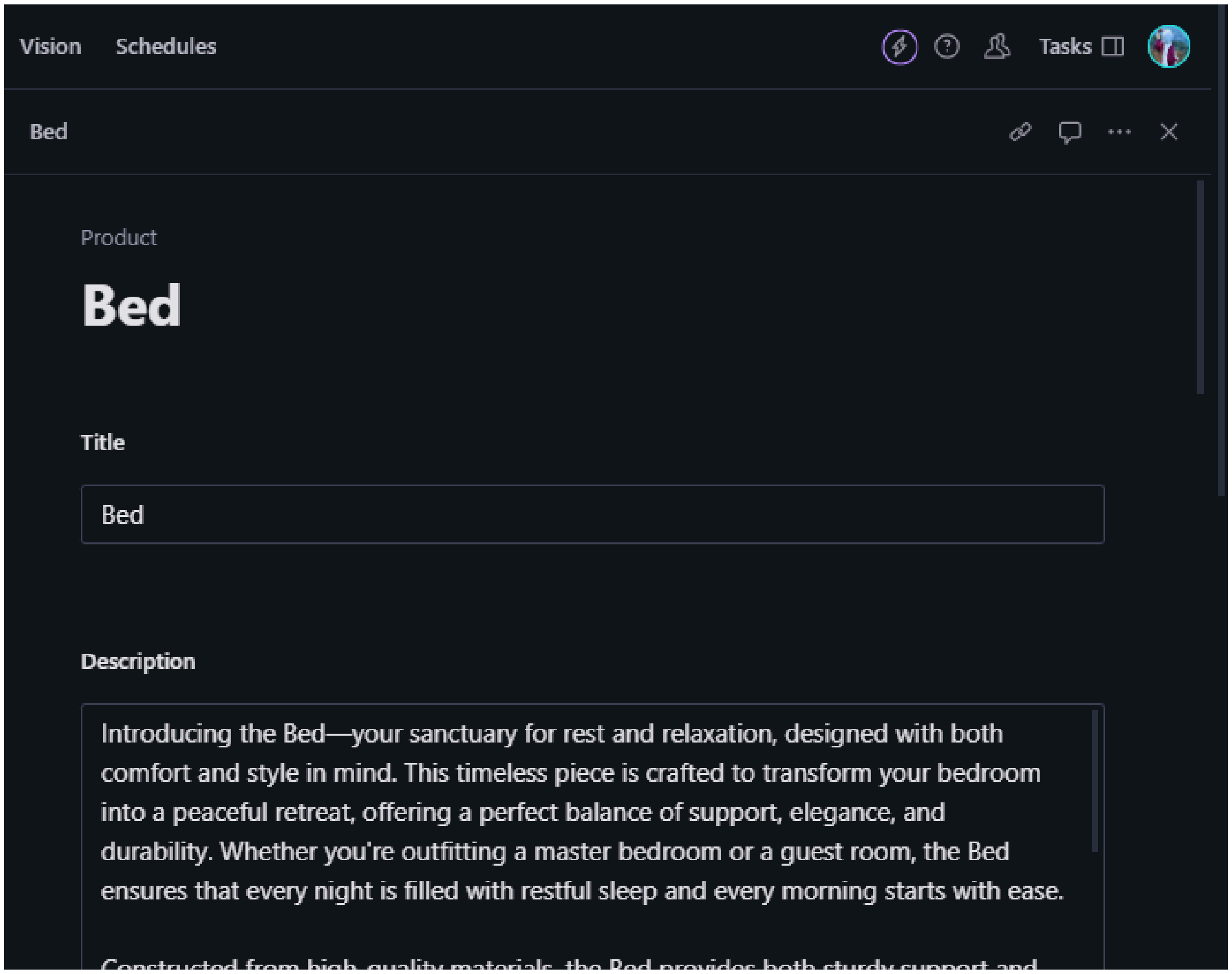
-10%
Rustic Vase Set
Bring the charm of nature into your home with the Rustic Vase Set. Perfect for thos…
$210 ~~$40~~

New
Amber Haven
Step into a world of warmth and tranquility with Amber Haven—a collection inspired…
$150 0

-10%
Bright Space
Welcome to BrightSpace—a collection designed to infuse your home with light,…
$180 ~~$40~~

New
Bed
Introducing the Bed—your sanctuary for rest and relaxation, designed with both…
$250 ~~$30~~

Populated Sanity CMS fields.



Product

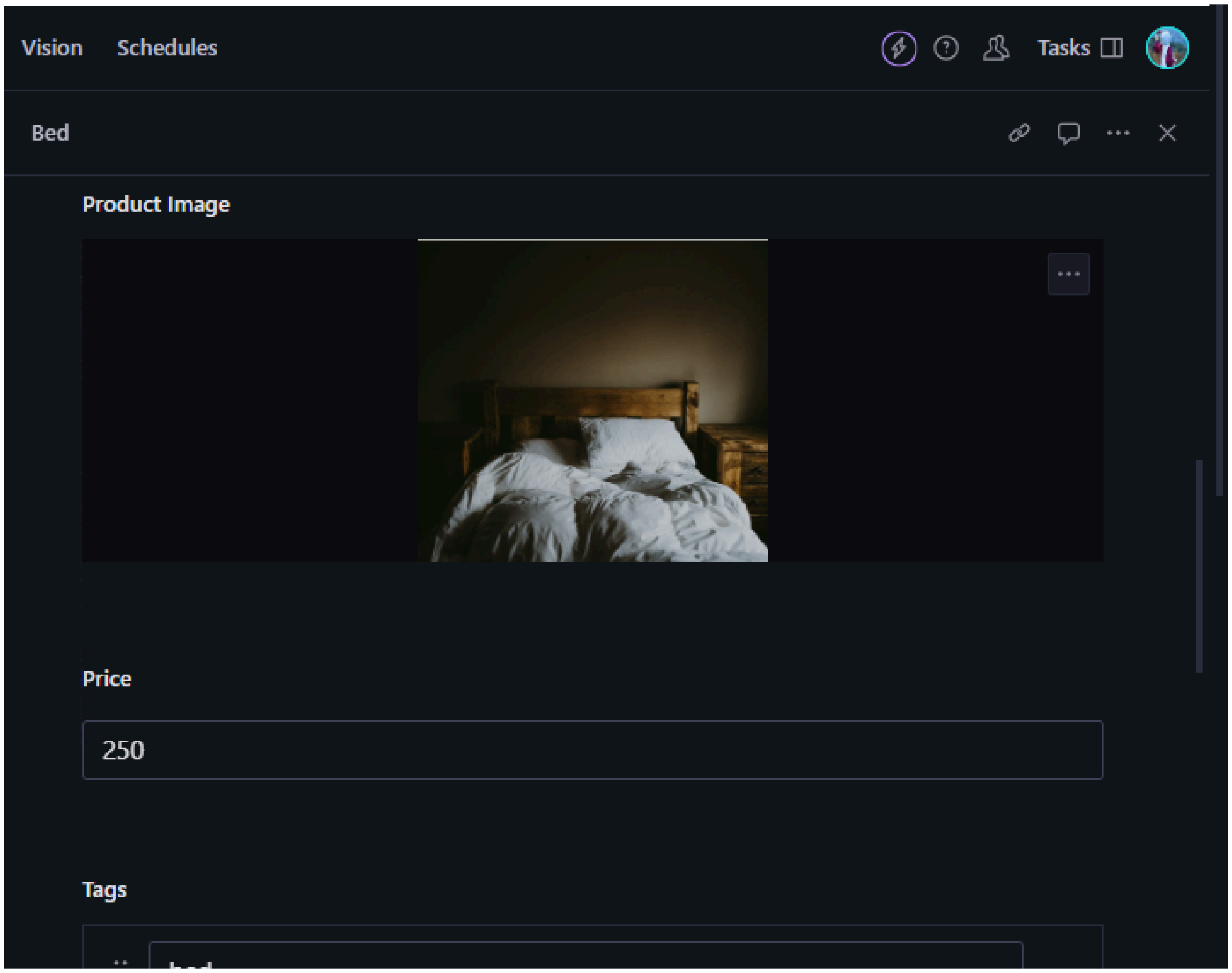
Bed

Title

Bed

Description

Introducing the Bed—your sanctuary for rest and relaxation, designed with both comfort and style in mind. This timeless piece is crafted to transform your bedroom into a peaceful retreat, offering a perfect balance of support, elegance, and durability. Whether you're outfitting a master bedroom or a guest room, the Bed ensures that every night is filled with restful sleep and every morning starts with ease.

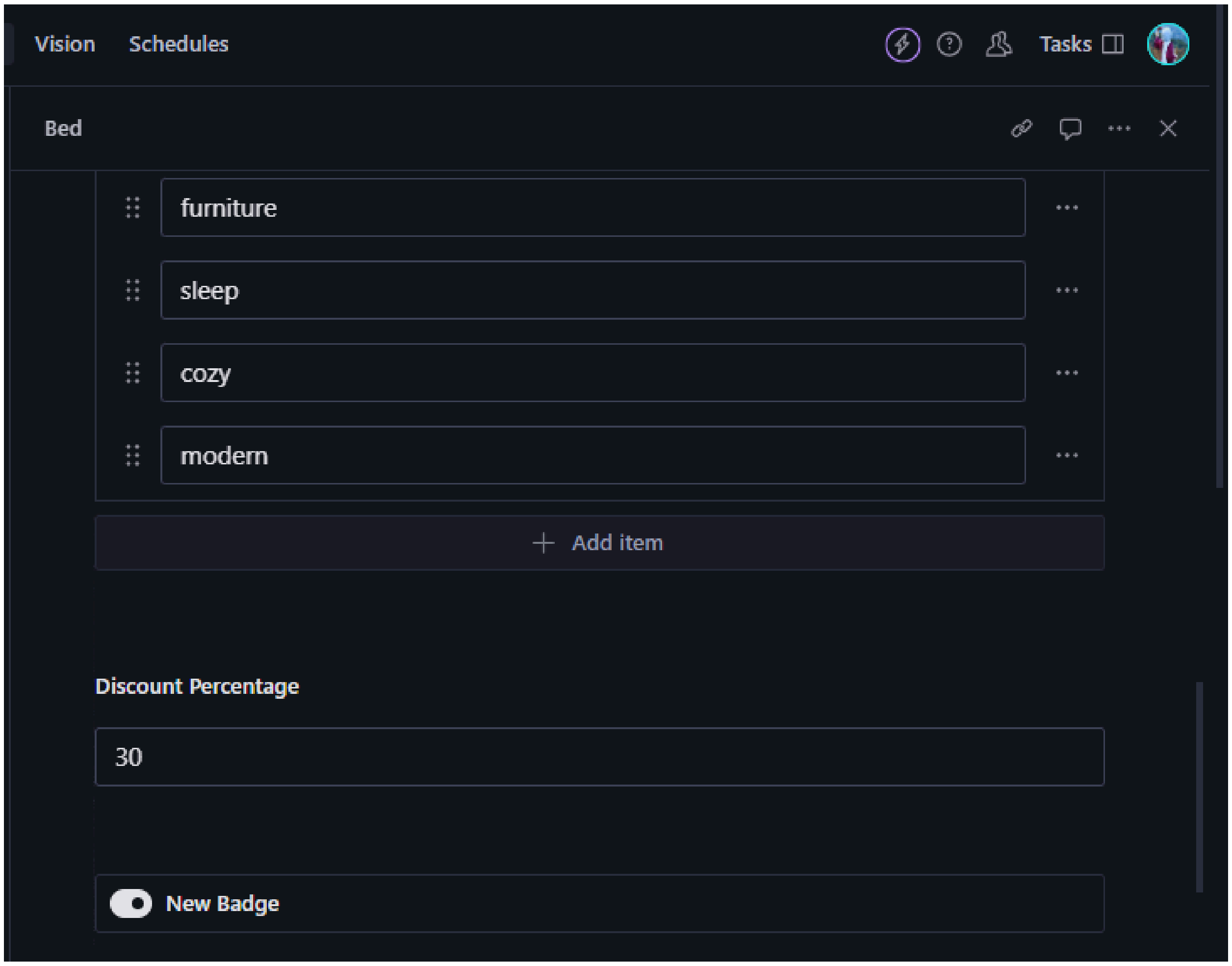


Product Image

Price

250

Tags



furniture

sleep

cozy

modern

Discount Percentage

30

New Badge

Code snippets for API integration and migration scripts.

```
import { createClient } from '@sanity/client';
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') });


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 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;
  }
}

async function uploadProduct(product) {
  try {
    const imageId = await uploadImageToSanity(product.imageUrl);
    if (imageId) {
      const document = {
        _type: 'product',
        title: product.title,
        price: product.price,
        productImage: {
          _type: 'image',
          asset: {
            _ref: imageId,
          },
        },
        tags: product.tags,
        dicountPercentage: product.dicountPercentage, // Typo in field
name: dicountPercentage -> discountPercentage
        description: product.description,
        isNew: product.isNew,
      };

      const createdProduct = await client.create(document);
      console.log(`Product ${product.title} uploaded successfully:`,
createdProduct);
    } else {
      console.log(`Product ${product.title} skipped due to image upload
failure.`);
    }
  } catch (error) {
    console.error('Error uploading product:', error);
  }
}

async function importProducts() {
  try {
    const response = await
fetch('https://template6-six.vercel.app/api/products');

    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);
  }
}

importProducts();
```