HACAKATHONE-3

[Q-Commerce Marketplace Builder]

Abstract

A modern food delivery website that offers a seamless browsing, ordering, and checkout experience. it ensures fast performance, secure payments via Stripe, and an intuitive UI for effortless food selection.

info

Food Tuck

FARWA KANWAL

Friday 9-12 (Morning)
Day 3 - API Integration Report - [Q-Commerce]

Project Overview

Project Name: FoodTuck

FoodTuck is an online food ordering platform that allows customers to browse menus, add items to their cart, and place orders seamlessly. The platform integrates APIs for real-time data fetching, secure payments, and order tracking to enhance the user experience.

API Integration Process for FoodTuck Marketplace

Reviewed API Documentation:

- Carefully examined the API documentation to understand available endpoints (e.g., `/foods`, `/chefs`).
- Analyzed the data structure, focusing on key fields (`name`, `price`, `image`) and their data types.

Set Up API Calls:

- Used **Postman** for testing API responses and verifying data accuracy.
- Implemented **Axios** instead of `fetch` for better error handling and control.
- Created utility functions within **Next.js** to fetch API data and validate responses via console logs.

Schema Adjustments for API Data

Food Schema Modifications:

- **Added Fields:**
  - `name`: Stores food item name.
  - `category`: Defines food type (e.g., appetizer, main course).
  - `price`: Holds numeric values for cost calculations.
  - `tags`: Stores food-specific labels (e.g., vegetarian, spicy).
  - `description`: Brief details about the food item.
  - `image`: Reference to the food image asset.

Chef Schema Modifications:

- **Added Fields:**
  - `name`: Chef's full name.
  - `position`: Job role (e.g., Head Chef, Sous Chef).
  - `experience`: Number of years in the profession.
  - `specialty`: Cuisine expertise (e.g., Italian, Pastry).
  - `description`: Background summary.
  - `image`: Reference to the chef's image asset.
- **Modifications:**
  - `experience` field updated to numeric type for better representation.

Migration Steps and Tools Used

Preparation for Migration:

- Reviewed API data structure and ensured compatibility with **Sanity CMS** schemas.
- Identified relationships between **foods** and **chefs** for accurate data mapping.

Selected Migration Tools:

- **Axios:** To fetch API data efficiently.
- **Sanity Client:** For creating and managing content within Sanity CMS.
- **Node.js Environment:** Used to write migration scripts and handle API interactions.

Steps Taken for Data Migration:

1. **Fetching Data from API:**
   - Used **Axios** to make GET requests for `foods` and `chefs`.
   - Logged responses to verify data integrity before processing.
2. **Uploading Images to Sanity:**
   - Downloaded images using **Axios** as an array buffer.
   - Uploaded assets to **Sanity CMS** and linked them to respective food and chef records.
3. **Creating and Uploading Documents to Sanity:**
   - Structured API data into **Sanity-compliant** documents.
   - Included references for images, prices, and categories.
   - Uploaded data using `client.create()` method.
4. **Error Handling and Debugging:**
   - Implemented structured error handling for API failures.
   - Maintained logs for API requests, image uploads, and document creation.

Tools Used for Migration:

✓ **Axios** – Fetch API data
✓ **Sanity Client** – CMS integration
✓ **Node.js** – Script execution

Task Status

✅ **API Documentation Reviewed**
✅ **Schema Structure Validated & Adjusted**
✅ **Data Successfully Migrated**
✅ **API Integrated in Next.js**
✅ **Final Submission Prepared**

TS chefs.ts

src > sanity > schemaTypes > TS chefs.ts > ...

```ts
const chef = {
  name: 'chef',
  type: 'document',
  title: 'Chef',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Chef Name',
    },
    {
      name: 'position',
      type: 'string',
      title: 'Position',
      description: 'Role or title of the chef (e.g., Head Chef, Sous Chef)',
    },
    {
      name: 'experience',
      type: 'number',
      title: 'Years of Experience',
      description: 'Number of years the chef has worked in the culinary field',
    },
    {
      name: 'specialty',
      type: 'string',
      title: 'Specialty',
      description: 'Specialization of the chef (e.g., Italian Cuisine, Pastry)',
    },
    {
      name: 'image',
      type: 'image',
      title: 'Chef Image',
      options: {
        hotspot: true,
      },
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
      description: 'Short bio or introduction about the chef',
    },
    {
      name: 'available',
      type: 'boolean',
      title: 'Currently Active',
      description: 'Availability status of the chef',
    },
  ],
};

export default chef
```

foods.ts

src > sanity > schemaTypes > foods.ts > ...

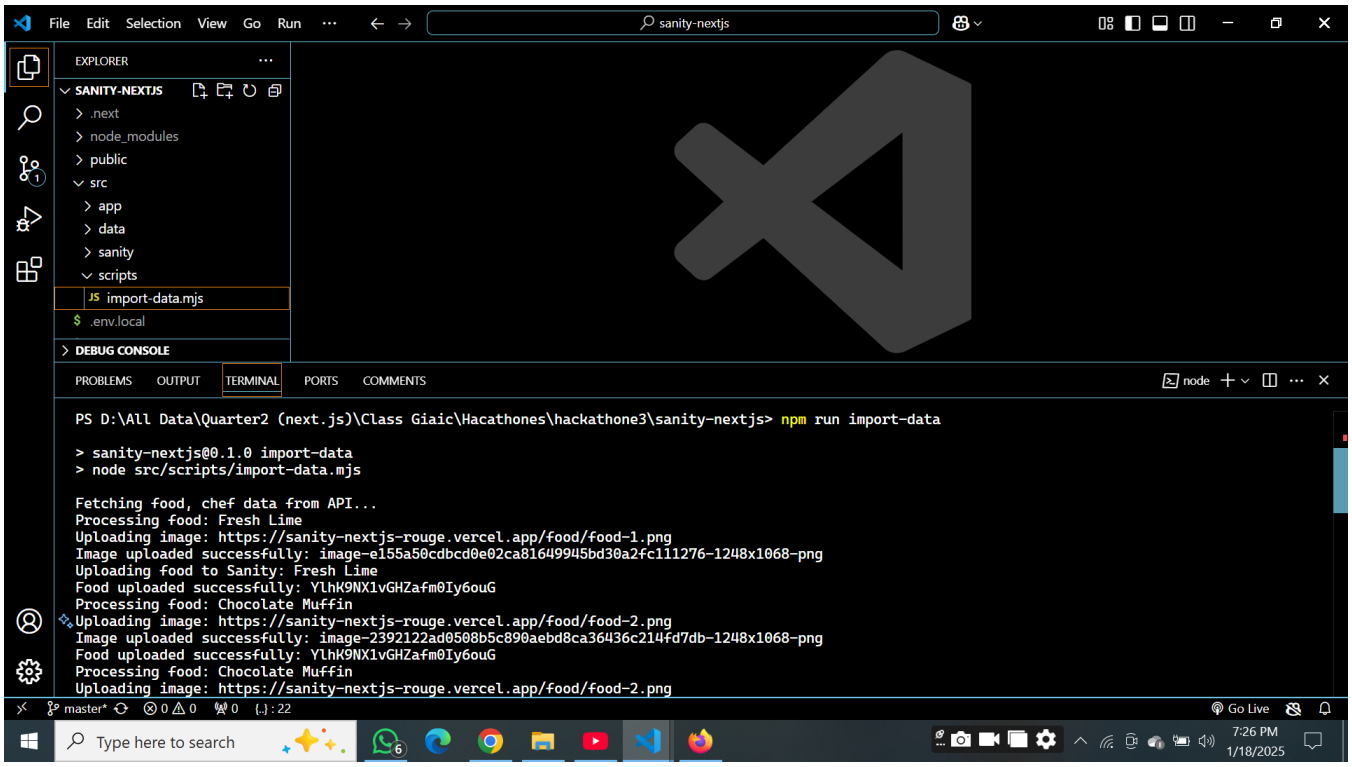
```ts
const food = {
  name: 'food',
  type: 'document',
  title: 'Food',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Food Name',
    },
    {
      name: 'category',
      type: 'string',
      title: 'Category',
      description:
        'Category of the food item (e.g., Burger, Sandwich, Drink, etc.)',
    },
    {
      name: 'price',
      type: 'number',
      title: 'Current Price',
    },
    {
      name: 'originalPrice',
      type: 'number',
      title: 'Original Price',
      description: 'Price before discount (if any)',
    },
    {
      name: 'tags',
      type: 'array',
      title: 'Tags',
      of: [{ type: 'string' }],
      options: {
        layout: 'tags',
      },
      description: 'Tags for categorization (e.g., Best Seller, Popular, New)',
    },
    {
      name: 'image',
      type: 'image',
      title: 'Food Image',
      options: {
        hotspot: true,
      },
    },
    {
      name: 'description',
      type: 'text',
      title: 'Description',
      description: 'Short description of the food item',
    },
    {
      name: 'available',
      type: 'boolean',
      title: 'Available',
      description: 'Availability status of the food item',
    },
  ],
};

export default food
```

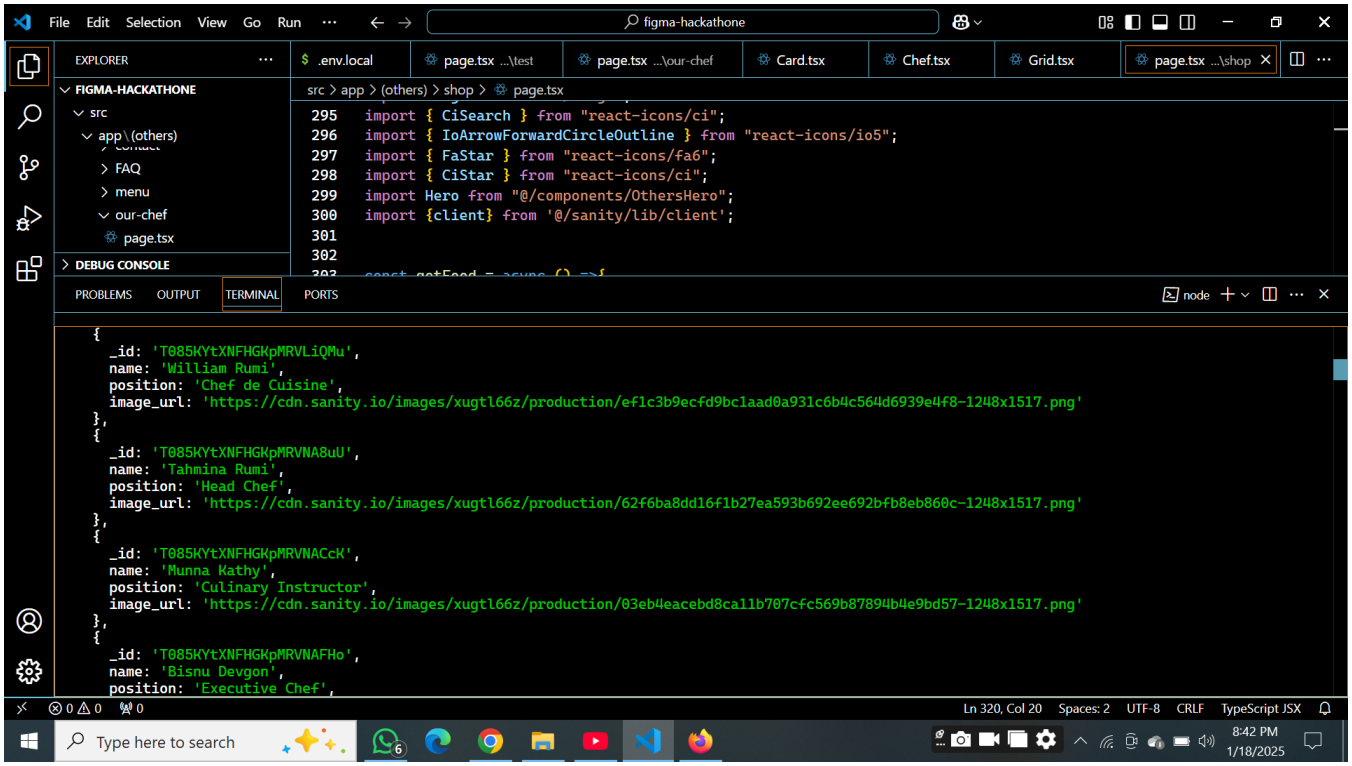
```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// 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: '2023-05-03',
});

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('Fetching food, chef data from API...');

    // API endpoint containing  data
    const promise = [];
    promise.push(
      axios.get('https://sanity-nextjs-rouge.vercel.app/api/foods')
    );
    promise.push(
      axios.get('https://sanity-nextjs-rouge.vercel.app/api/chefs')
    );

    const [foodResponse, chefsResponse] = await Promise.all(promise);
    const foods = foodResponse.data;
    const chefs = chefsResponse.data;

    for (const food of foods) {
      console.log(`Processing food: ${food.name}`);

      let imageRef = null;
      if (food.image) {
        imageRef = await uploadImageToSanity(food.image);
      }

      const sanityFood = {
        _type: 'food',
        name: food.name,
        category: food.category || null,
        price: food.price,
        originalPrice: food.originalPrice || null,
        tags: food.tags || [],
        description: food.description || '',
        available: food.available !== undefined ? food.available : true,
        image: imageRef
          ? {
              _type: 'image',
              asset: {
                _type: 'reference',
                _ref: imageRef,
              },
            }
          : undefined,
      };

      console.log('Uploading food to Sanity:', sanityFood.name);
      const result = await client.create(sanityFood);
      console.log(`Food uploaded successfully: ${result._id}`);
    }

    for (const chef of chefs) {
      console.log(`Processing chef: ${chef.name}`);

      let imageRef = null;
      if (chef.image) {
        imageRef = await uploadImageToSanity(chef.image);
      }

      const sanityChef = {
        _type: 'chef',
        name: chef.name,
        position: chef.position || null,
        experience: chef.experience || 0,
        specialty: chef.specialty || '',
        description: chef.description || '',
        available: chef.available !== undefined ? chef.available : true,
        image: imageRef
          ? {
              _type: 'image',
              asset: {
                _type: 'reference',
                _ref: imageRef,
              },
            }
          : undefined,
      };

      console.log('Uploading chef to Sanity:', sanityChef.name);
      const result = await client.create(sanityChef);
      console.log(`Chef uploaded successfully: ${result._id}`);
    }

    console.log('Data import completed successfully!');
  } catch (error) {
    console.error('Error importing data:', error);
  }
}

importData();
```

TS index.ts

src > sanity > schemaTypes > TS index.ts > ...

```ts
import chef from './chefs';
import food from './foods';
import { type SchemaTypeDefinition } from 'sanity'

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




File Edit Selection View Go Run
figma-hackathone
EXPLORER
.env.local
page.tsx ...\test
page.tsx ...\our-chef
Card.tsx
Chef.tsx
Grid.tsx
page.tsx ...\shop
FIGMA-HACKATHONE
src
app\(others)
FAQ
menu
our-chef
page.tsx
DEBUG CONSOLE
src > app > (others) > shop > page.tsx
295 import { CiSearch } from "react-icons/ci";
296 import { IoArrowForwardCircleOutline } from "react-icons/io5";
297 import { FaStar } from "react-icons/fa6";
298 import { CiStar } from "react-icons/ci";
299 import Hero from "@/components/OthersHero";
300 import {client} from '@/sanity/lib/client';
301
302
PROBLEMS OUTPUT TERMINAL PORTS
node
{
_id: 'T085KYtXNFHGKpMRVLiQMu',
name: 'William Rumi',
position: 'Chef de Cuisine',
image_url: 'https://cdn.sanity.io/images/xugtl66z/production/ef1c3b9ecfd9bc1aad0a931c6b4c564d6939e4f8-1248x1517.png'
},
{
_id: 'T085KYtXNFHGKpMRVNA8uU',
name: 'Tahmina Rumi',
position: 'Head Chef',
image_url: 'https://cdn.sanity.io/images/xugtl66z/production/62f6ba8dd16f1b27ea593b692ee692bfb8eb860c-1248x1517.png'
},
{
_id: 'T085KYtXNFHGKpMRVNACcK',
name: 'Munna Kathy',
position: 'Culinary Instructor',
image_url: 'https://cdn.sanity.io/images/xugtl66z/production/03eb4eacebd8ca11b707cfc569b87894b4e9bd57-1248x1517.png'
},
{
_id: 'T085KYtXNFHGKpMRVNAFHo',
name: 'Bisnu Devgon',
position: 'Executive Chef',
Ln 320, Col 20 Spaces: 2 UTF-8 CRLF TypeScript JSX
Type here to search
8:42 PM
1/18/2025

page.tsx

src > app > (others) > our-chef > page.tsx > ...

```tsx
import React from "react";
import Hero from "@/components/OthersHero";
import { client } from "@/sanity/lib/client";
import Image from "next/image";
import { FaArrowRight } from "react-icons/fa6";
import Link from 'next/link'

// Define the interface for Chef data
interface IChef {
  _id: string;
  name: string;
  position: string;
  image_url: string;
}

// Fetch chefs dynamically from Sanity
const getChefs = async (): Promise<IChef[]> => {
  const query = `
    *[_type == "chef"] {
      _id,
      name,
      position,
      "image_url": image.asset->url
    }
  `;
  const chefs = await client.fetch(query);
  return chefs;
};

const OurChef = async () => {
  const chefs = await getChefs();

  return (
    <div>
      {/* Render the Hero Section */}
      <Hero heading="Our Chefs" />

      {/* Render the Chef Grid */}
      <div className="wrapper grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-[30px] space-y-1 my-[50px] md:my-[120px]">
        {chefs.map((chef: IChef) => (
          <div
            className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
            key={chef._id}
          >
            {/* Chef Image */}
            <div className="w-full h-[360px] flex justify-center items-center">
              <Image
                src={chef.image_url}
                alt={chef.name}
                width={300}
                height={300}
                className="w-full h-full object-cover"
              />
            </div>

            {/* Chef Info */}
            <div className="flex flex-col items-center w-full bg-gray-50 p-2">
              <h2 className="font-bold text-lg">{chef.name}</h2>
              <p className="text-sm text-gray-600">{chef.position}</p>
              <Link href={`/our-chef/${chef._id}`}><button className="text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
};

export default OurChef;
```

page.tsx

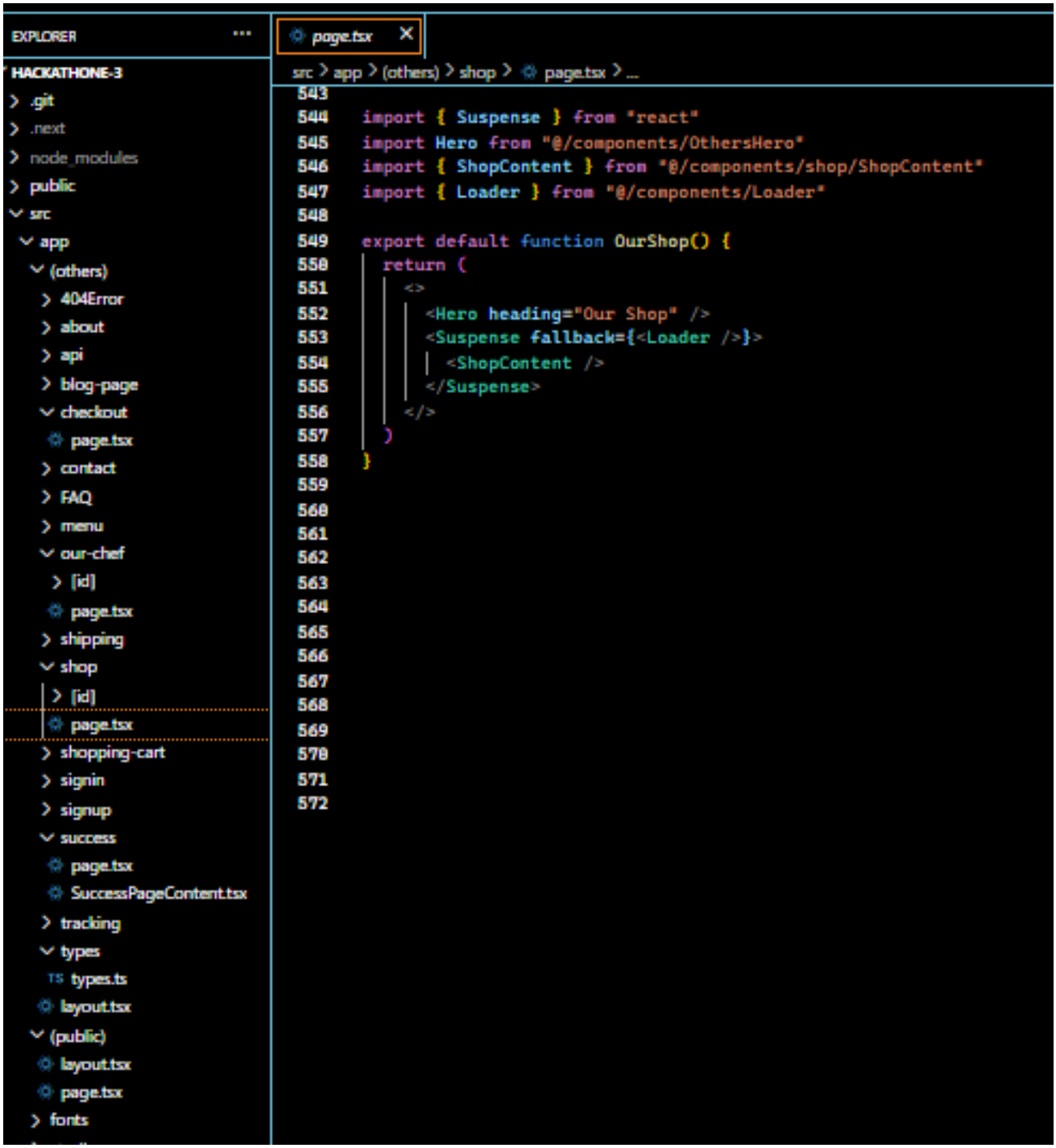
src > app > (others) > our-chef > page.tsx > ...

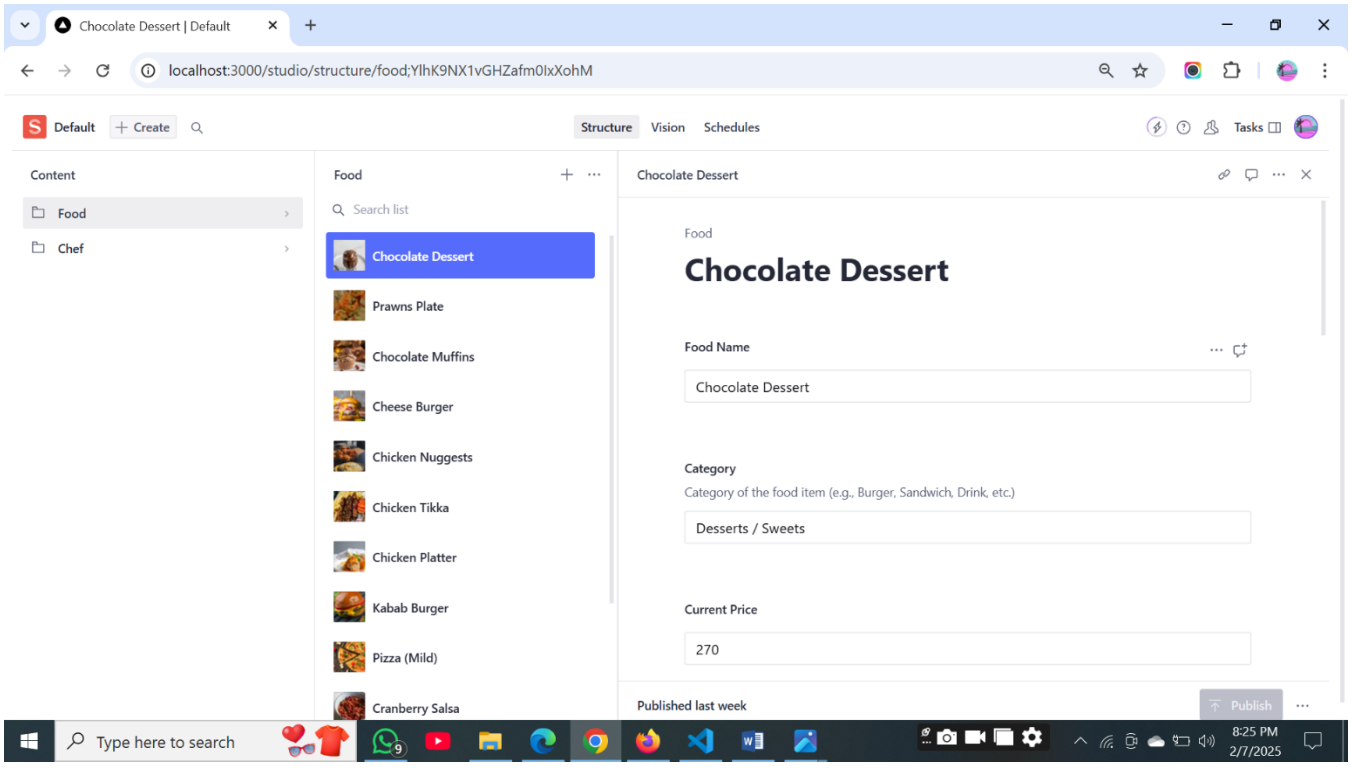
```tsx
import React from "react";
import Hero from "@/components/OthersHero";
import { client } from "@/sanity/lib/client";
import Image from "next/image";
import { FaArrowRight } from "react-icons/fa6";
import Link from 'next/link'

// Define the interface for Chef data
interface IChef {
  _id: string;
  name: string;
  position: string;
  image_url: string;
}

// Fetch chefs dynamically from Sanity
const getChefs = async (): Promise<IChef[]> => {
  const query = `
    *[_type == "chef"] {
      _id,
      name,
      position,
      "image_url": image.asset->url
    }
  `;
  const chefs = await client.fetch(query);
  return chefs;
};

const OurChef = async () => {
  const chefs = await getChefs();

  return (
    <div>
      {/* Render the Hero Section */}
      <Hero heading="Our Chefs" />

      {/* Render the Chef Grid */}
      <div className="wrapper grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-[30px] space-y-1 my-[50px] md:my-[120px]">
        {chefs.map((chef: IChef) => (
          <div
            className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
            key={chef._id}
          >
            {/* Chef Image */}
            <div className="w-full h-[360px] flex justify-center items-center">
              <Image
                src={chef.image_url}
                alt={chef.name}
                width={300}
                height={300}
                className="w-full h-full object-cover"
              />
            </div>

            {/* Chef Info */}
            <div className="flex flex-col items-center w-full bg-gray-50 p-2">
              <h2 className="font-bold text-lg">{chef.name}</h2>
              <p className="text-sm text-gray-600">{chef.position}</p>
              <Link href={`/our-chef/${chef._id}`}><button className="text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
};

export default OurChef;
```

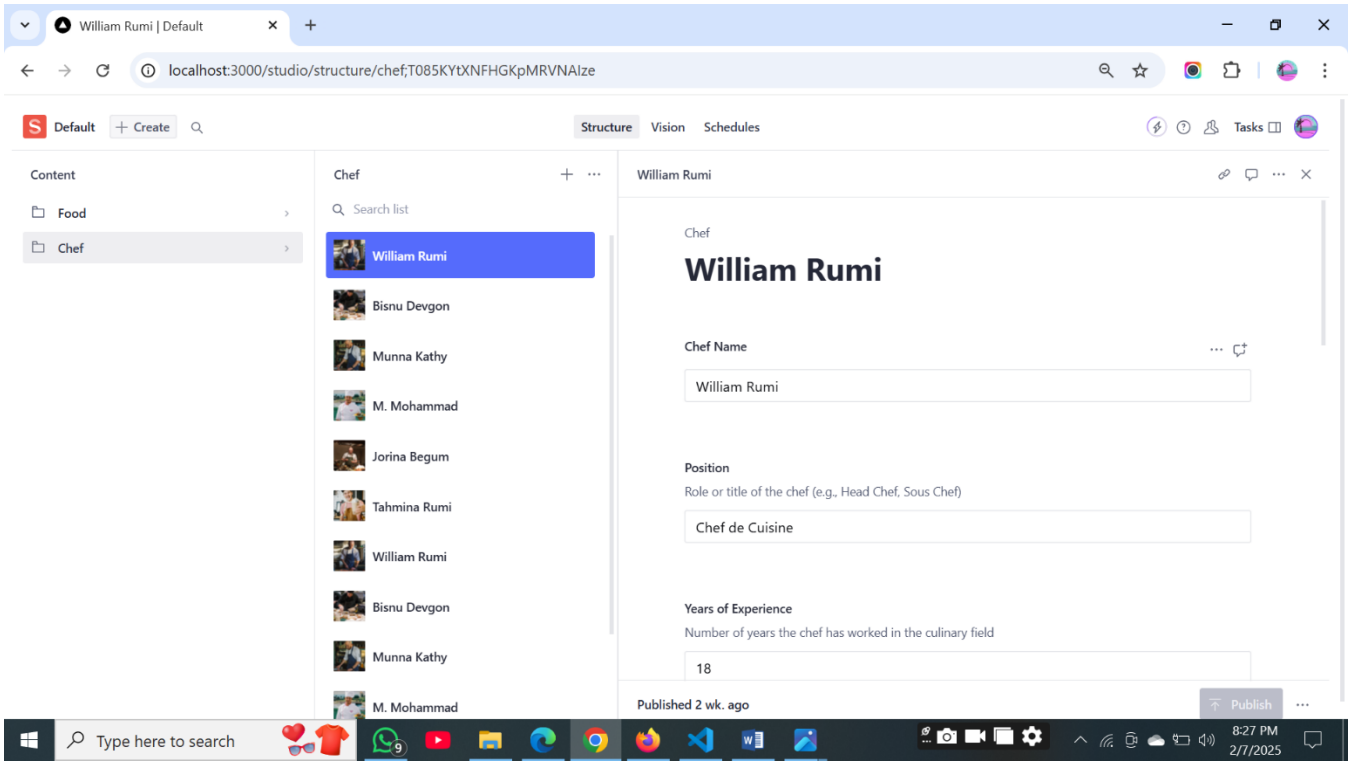
page.tsx

src > app > (others) > our-chef > page.tsx > ...

```tsx
import React from "react";
import Hero from "@/components/OthersHero";
import { client } from "@/sanity/lib/client";
import Image from "next/image";
import { FaArrowRight } from "react-icons/fa6";
import Link from 'next/link'

// Define the interface for Chef data
interface IChef {
  _id: string;
  name: string;
  position: string;
  image_url: string;
}

// Fetch chefs dynamically from Sanity
const getChefs = async (): Promise<IChef[]> => {
  const query = `
    *[_type == "chef"] {
      _id,
      name,
      position,
      "image_url": image.asset->url
    }
  `;
  const chefs = await client.fetch(query);
  return chefs;
};

const OurChef = async () => {
  const chefs = await getChefs();

  return (
    <div>
      {/* Render the Hero Section */}
      <Hero heading="Our Chefs" />

      {/* Render the Chef Grid */}
      <div className="wrapper grid grid-cols-1 sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 gap-[30px] space-y-1 my-[50px] md:my-[120px]">
        {chefs.map((chef: IChef) => (
          <div
            className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
            key={chef._id}
          >
            {/* Chef Image */}
            <div className="w-full h-[360px] flex justify-center items-center">
              <Image
                src={chef.image_url}
                alt={chef.name}
                width={300}
                height={300}
                className="w-full h-full object-cover"
              />
            </div>

            {/* Chef Info */}
            <div className="flex flex-col items-center w-full bg-gray-50 p-2">
              <h2 className="font-bold text-lg">{chef.name}</h2>
              <p className="text-sm text-gray-600">{chef.position}</p>
              <Link href={`/our-chef/${chef._id}`}><button className="text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
};

export default OurChef;
```


EXPLORER
page.tsx
HACKATHONE-3
src > app > (others) > shop > page.tsx > ...
.git
.next
node_modules
public
src
app
(others)
404Error
about
api
blog-page
checkout
page.tsx
contact
FAQ
menu
our-chef
[id]
page.tsx
shipping
shop
[id]
page.tsx
shopping-cart
signin
signup
success
page.tsx
SuccessPageContent.tsx
tracking
types
types.ts
layout.tsx
(public)
layout.tsx
page.tsx
fonts
import { Suspense } from "react"
import Hero from "@/components/OthersHero"
import { ShopContent } from "@/components/shop/ShopContent"
import { Loader } from "@/components/Loader"
export default function OurShop() {
return (
<>
<Hero heading="Our Shop" />
<Suspense fallback={<Loader />}>
<ShopContent />
</Suspense>
</>
)
}


Chocolate Dessert | Default
localhost:3000/studio/structure/food;YlhK9NX1vGHZafm0IxXohM
Default
Create
Structure
Vision
Schedules
Tasks
Content
Food
Chef
Food
Search list
Chocolate Dessert
Prawns Plate
Chocolate Muffins
Cheese Burger
Chicken Nuggests
Chicken Tikka
Chicken Platter
Kabab Burger
Pizza (Mild)
Cranberry Salsa
Chocolate Dessert
Food
Chocolate Dessert
Food Name
Chocolate Dessert
Category
Category of the food item (e.g., Burger, Sandwich, Drink, etc.)
Desserts / Sweets
Current Price
270
Published last week
Publish
Type here to search
8:25 PM
2/7/2025


William Rumi | Default
localhost:3000/studio/structure/chef;T085KYtXNFHGKpMRVNAIze
Default
Create
Structure
Vision
Schedules
Tasks
Content
Food
Chef
Chef
Search list
William Rumi
Bisnu Devgon
Munna Kathy
M. Mohammad
Jorina Begum
Tahmina Rumi
William Rumi
Bisnu Devgon
Munna Kathy
M. Mohammad
William Rumi
Chef
William Rumi
Chef Name
William Rumi
Position
Role or title of the chef (e.g., Head Chef, Sous Chef)
Chef de Cuisine
Years of Experience
Number of years the chef has worked in the culinary field
18
Published 2 wk. ago
Publish
Type here to search
8:27 PM
2/7/2025

Sort By: Price: Low to High Show: Show 12

Lime Juice
Beverages/Drinks
$120.00
~~$160.00~~
Show Details

Prawns Plate
Seafood
$200.00
~~$250.00~~
Show Details

Kabab Burger
Fast Food
$220.00
~~$270.00~~
Show Details

Chocolate Muffins
Desserts / Sweets
$230.00
~~$280.00~~
Show Details

Chocolate Dessert
Desserts / Sweets
$270.00
~~$330.00~~
Show Details

Chicken Nuggests
Fast Food
$280.00
~~$340.00~~
Show Details

Cranberry Salsa
Dips & Sauces
$300.00
~~$380.00~~
Show Details

Chicken Platter
Chicken
$340.00
~~$380.00~~
Show Details

Chicken Tikka
Chicken
$350.00
~~$410.00~~
Show Details

Cheese Burger
Fast Food
$350.00
~~$390.00~~
Show Details

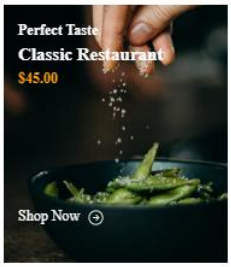
Mix Vegetables
Vegetables
$430.00
~~$490.00~~
Show Details

Pizza (Mild)
Pizza
$450.00
~~$500.00~~
Show Details

Search Product

Category

- Chicken
- Fast Food
- Pizza
- Vegetables
- Desserts / Sweets
- Seafood
- Dips & Sauces
- Beverages/Drinks
- Pizza



Filter By Price

100 - 500

$100 - $500 Filter

Latest Products

Fresh Breakfast
★★☆☆☆
14.5$

Product Tags

Services Our Menu

Tahmina Rumi
Head Chef
View Details →

William Rumi
Chef de Cuisine
View Details →

Tahmina Rumi
Head Chef
View Details →

Munna Kathy
Culinary Instructor
View Details →

Bisnu Devgon
Executive Chef
View Details →

William Rumi
Chef de Cuisine
View Details →

Bisnu Devgon
Executive Chef
View Details →

M. Mohammad
Grill Master
View Details →

Jorina Begum
Sous Chef
View Details →

M. Mohammad
Grill Master
View Details →

Munna Kathy
Culinary Instructor
View Details →

Jorina Begum
Sous Chef
View Details →