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**

```typescript
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
```
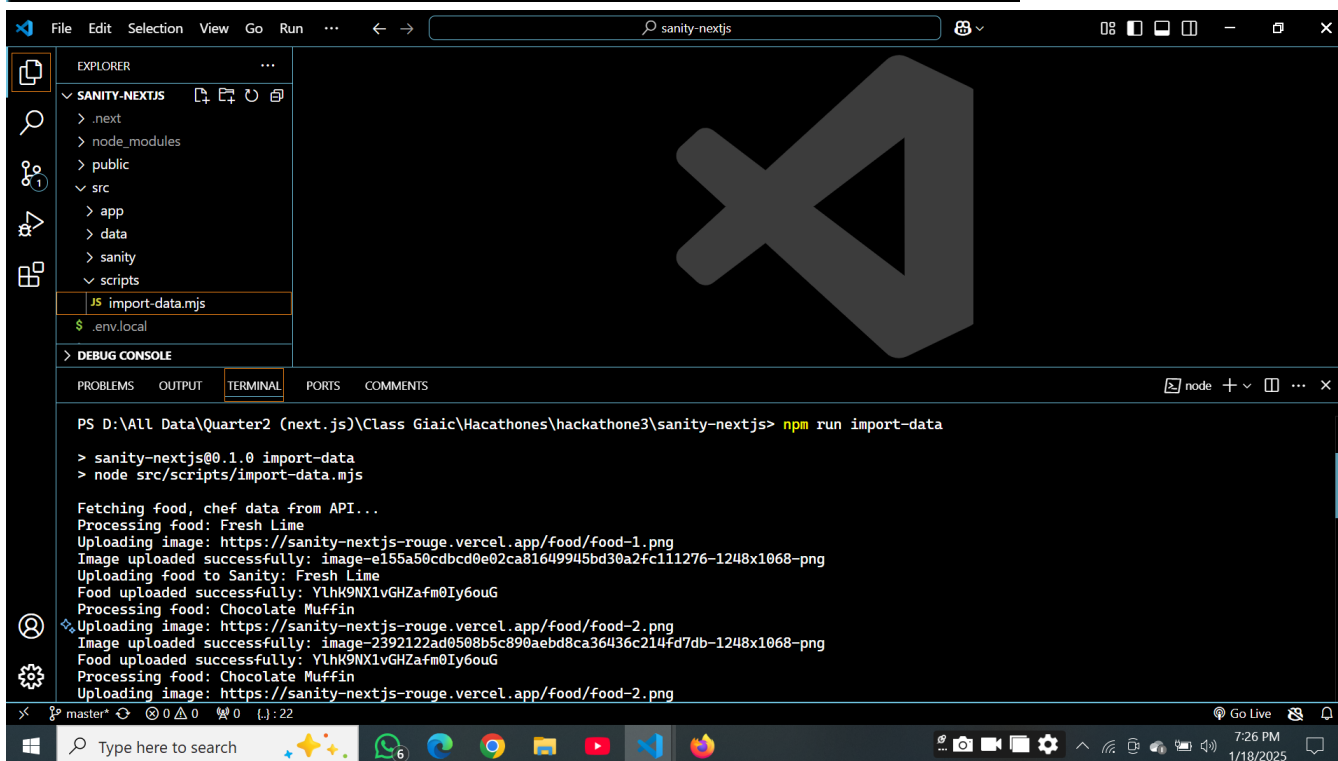
```ts
const food = {
  name: 'food',
  type: 'document',
  title: 'Food',
  fields: [
    {
      name: 'name',
      type: 'string',
      title: 'Food Name',
    },
    {
      name: 'category',
      type: 'string',
      title: 'Category',
      description:
        'Category of the food item (e.g., 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
```
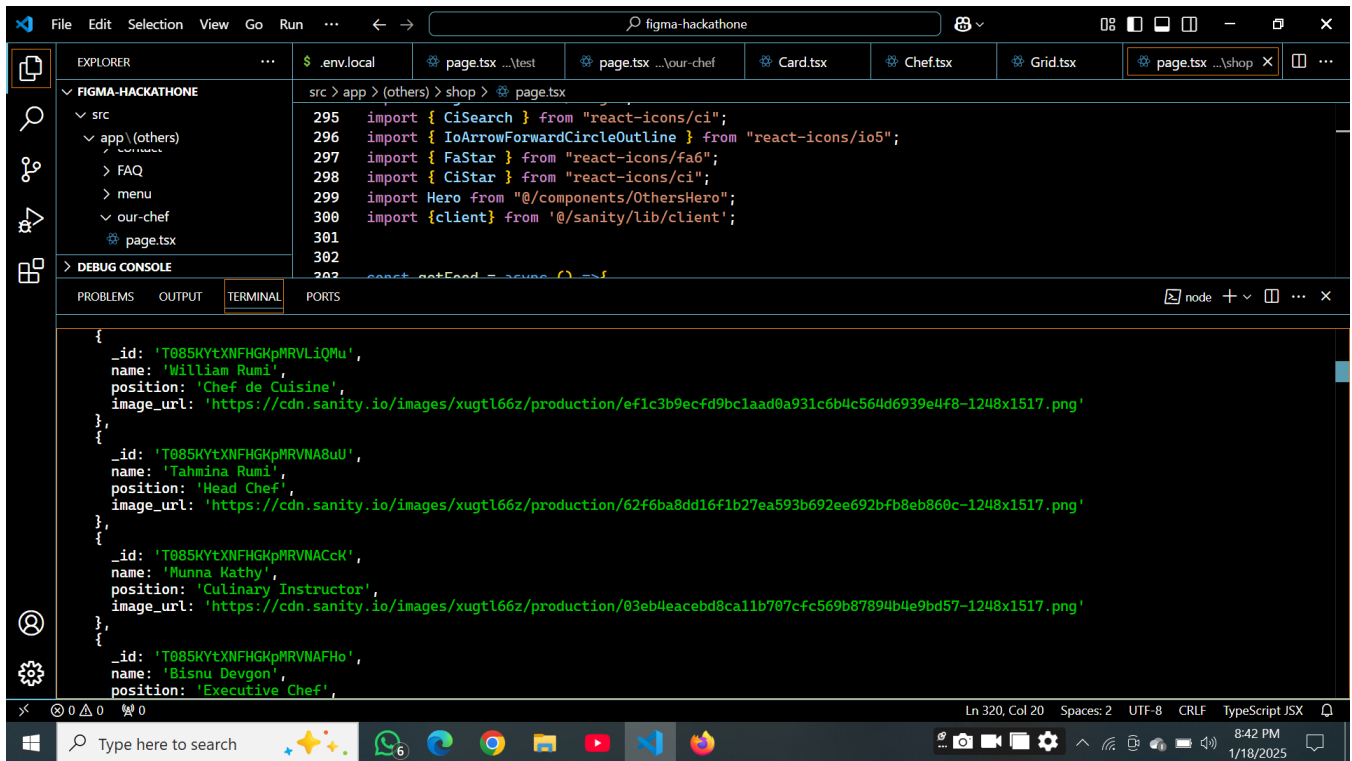
```javascript
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// 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: '2021-08-31',
});

export 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;
  }
}

export 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 [foodsResponse, chefsResponse] = await Promise.all($Promise);
    const foods = foodsResponse.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();
```

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

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


PS D:\All Data\Quarter2 (next.js)\Class Giaic\Hacathones\hackathone3\sanity-nextjs> npm run import-data

> sanity-nextjs@0.1.0 import-data
> node src/scripts/import-data.mjs

Fetching food, chef data from API...
Processing food: Fresh Lime
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-1.png
Image uploaded successfully: image-e155a50cdbcd0e02ca81649945bd30a2fc111276-1248x1068-png
Uploading food to Sanity: Fresh Lime
Food uploaded successfully: YlhK9NX1vGHZafm0Iy6ouG
Processing food: Chocolate Muffin
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-2.png
Image uploaded successfully: image-2392122ad0508b5c890aebd8ca36436c214fd7db-1248x1068-png
Food uploaded successfully: YlhK9NX1vGHZafm0Iy6ouG
Processing food: Chocolate Muffin
Uploading image: https://sanity-nextjs-rouge.vercel.app/food/food-2.png

EXPLORER ···

.env.local   page.tsx ...\test   page.tsx ...\our-chef   Card.tsx   Chef.tsx   Grid.tsx   page.tsx ...\shop

src > app > (others) > shop > page.tsx

FIGMA-HACKATHONE
- src
  - app\(others)
    - contact
    - FAQ
    - menu
    - our-chef
      - page.tsx

DEBUG CONSOLE

```
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
303    const getFeed = async () =>{
```

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

```tsx
21  import React from "react";
22  import Hero from "@/components/OthersHero";
23  import { client } from "@/sanity/lib/client";
24  import Image from "next/image";
25  import { FaArrowRight } from "react-icons/fa6";
26  import Link from 'next/link'
27
28  // Define the interface for Chef data
29  interface IChef {
30    _id: string;
31    name: string;
32    position: string;
33    image_url: string;
34  }
35
36  // Fetch chefs dynamically from Sanity
37  const getChefs = async (): Promise<IChef[]> => {
38    const query = `
39      *[_type == "chef"] {
40        _id,
41        name,
42        position,
43        "image_url": image.asset->url
44      }
45    `;
46    const chefs = await client.fetch(query);
47    return chefs;
48  };
49
50  const OurChef = async () => {
51    const chefs = await getChefs();
52
53    return (
54      <div>
55        {/* Render the Hero Section */}
56        <Hero heading="Our Chefs" />
57
58        {/* Render the Chef Grid */}
59        <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]">
60          {chefs.map((chef: IChef) => (
61            <div
62              className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
63              key={chef._id}
64            >
65              {/* Chef Image */}
66              <div className="w-full h-[360px] flex justify-center items-center">
67                <Image
68                  src={chef.image_url}
69                  alt={chef.name}
70                  width={300}
71                  height={300}
72                  className="w-full h-full object-cover"
73                />
74              </div>
75
76              {/* Chef Info */}
77              <div className="flex flex-col items-center w-full bg-gray-50 p-2">
78                <h2 className="font-bold text-lg">{chef.name}</h2>
79                <p className="text-sm text-gray-600">{chef.position}</p>
80                <Link href={`/our-chef/${chef._id}`}><button className="text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
81              </div>
82            </div>
83          ))}
84        </div>
85      </div>
86    );
87  };
88
89  export default OurChef;
90
```

```tsx
21  import React from "react";
22  import Hero from "@/components/OthersHero";
23  import { client } from "@/sanity/lib/client";
24  import Image from "next/image";
25  import { FaArrowRight } from "react-icons/fa6";
26  import Link from 'next/link'
27
28  // Define the interface for Chef data
29  interface IChef {
30    _id: string;
31    name: string;
32    position: string;
33    image_url: string;
34  }
35
36  // Fetch chefs dynamically from Sanity
37  const getChefs = async (): Promise<IChef[]> => {
38    const query = `
39      *[_type == "chef"] {
40        _id,
41        name,
42        position,
43        "image_url": image.asset->url
44      }
45    `;
46    const chefs = await client.fetch(query);
47    return chefs;
48  };
49
50  const OurChef = async () => {
51    const chefs = await getChefs();
52
53    return (
54      <div>
55        {/* Render the Hero Section */}
56        <Hero heading="Our Chefs" />
57
58        {/* Render the Chef Grid */}
59        <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]">
60          {chefs.map((chef: IChef) => (
61            <div
62              className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
63              key={chef._id}
64            >
65              {/* Chef Image */}
66              <div className="w-full h-[360px] flex justify-center items-center">
67                <Image
68                  src={chef.image_url}
69                  alt={chef.name}
70                  width={300}
71                  height={300}
72                  className="w-full h-full object-cover"
73                />
74              </div>
75
76              {/* Chef Info */}
77              <div className="flex flex-col items-center w-full bg-gray-50 p-2">
78                <h2 className="font-bold text-lg">{chef.name}</h2>
79                <p className="text-sm text-gray-600">{chef.position}</p>
80                <Link href={`/our-chef/${chef._id}`}><button className="text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
81              </div>
82            </div>
83          ))}
84        </div>
85      </div>
86    );
87  };
88
89  export default OurChef;
90
```

```tsx
21  import React from "react";
22  import Hero from "@/components/OthersHero";
23  import { client } from "@/sanity/lib/client";
24  import Image from "next/image";
25  import { FaArrowRight } from "react-icons/fa6";
26  import Link from 'next/link'
27
28  // Define the interface for Chef data
29  interface IChef {
30    _id: string;
31    name: string;
32    position: string;
33    image_url: string;
34  }
35
36  // Fetch chefs dynamically from Sanity
37  const getChefs = async (): Promise<IChef[]> => {
38    const query = `
39      *[_type == "chef"] {
40        _id,
41        name,
42        position,
43        "image_url": image.asset->url
44      }
45    `;
46    const chefs = await client.fetch(query);
47    return chefs;
48  };
49
50  const OurChef = async () => {
51    const chefs = await getChefs();
52
53    return (
54      <div>
55        {/* Render the Hero Section */}
56        <Hero heading="Our Chefs" />
57
58        {/* Render the Chef Grid */}
59        <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]">
60          {chefs.map((chef: IChef) => (
61            <div
62              className="w-full lg:w-[260px] h-[450px] shadow-lg rounded-lg overflow-hidden"
63              key={chef._id}
64            >
65              {/* Chef Image */}
66              <div className="w-full h-[360px] flex justify-center items-center">
67                <Image
68                  src={chef.image_url}
69                  alt={chef.name}
70                  width={300}
71                  height={300}
72                  className="w-full h-full object-cover"
73                />
74              </div>
75
76              {/* Chef Info */}
77              <div className="flex flex-col items-center w-full bg-gray-50 p-2">
78                <h2 className="font-bold text-lg">{chef.name}</h2>
79                <p className="text-sm text-gray-600">{chef.position}</p>
80                <Link href={`/our-chef/${chef._id}`}><button className=" text-[#FF9F0D] inline-block font-semibold">View Details <FaArrowRight className="inline-block text-sm"/></button></Link>
81              </div>
82            </div>
83          ))}
84        </div>
85      </div>
86    );
87  };
88
89  export default OurChef;
90
```

page.tsx ✕

HACKATHONE-3
> .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
        TS types.ts
      ⚙ layout.tsx
    ∨ (public)
      ⚙ layout.tsx
      ⚙ page.tsx
    > fonts

```tsx
543
544    import { Suspense } from "react"
545    import Hero from "@/components/OthersHero"
546    import { ShopContent } from "@/components/shop/ShopContent"
547    import { Loader } from "@/components/Loader"
548
549    export default function OurShop() {
550      return (
551        <>
552          <Hero heading="Our Shop" />
553          <Suspense fallback={<Loader />}>
554            <ShopContent />
555          </Suspense>
556        </>
557      )
558    }
559
560
561
562
563
564
565
566
567
568
569
570
571
572
```

```tsx
"use client"

import { useState, useEffect, useRef } from "react"
import Image from "next/image"
import { Button } from "@/components/ui/button"
import { Input } from "@/components/ui/input"
import { Star, Facebook, Twitter, Instagram, PinIcon as Pinterest } from "lucide-react"
import Hero from "@/components/OthersHero"
import { Loader } from "@/components/Loader"
import { useCart } from "@/contexts/CartContext"
import { client } from "@/sanity/lib/client"
import { toast } from "react-hot-toast"
import Link from "next/link"

interface ProductData {
  _id: string
  name: string
  imageUrl: string
  description: string
  price: number
}

interface FeaturedImage {
  _id: string
  imageUrl: string
}

export default function ProductPage({ params }: { params: { id: string } }) {
  const [quantity, setQuantity] = useState(1)
  const [selectedImage, setSelectedImage] = useState(0)
  const [product, setProduct] = useState<ProductData | null>(null)
  const [featuredImages, setFeaturedImages] = useState<FeaturedImage[]>([])
  const [translateX, setTranslateX] = useState(0)
  const [isHovering, setIsHovering] = useState(false)
  const { addToCart } = useCart()
  const sliderRef = useRef<HTMLDivElement>(null)

  useEffect(() => {
    const fetchProduct = async () => {
      try {
        const query = `*[_type == "food" && _id == $id][0] {
          _id,
          name,
          description,
          "imageUrl": image.asset->url,
          price
        }`
        const data = await client.fetch(query, { id: params.id })
        setProduct(data)
      } catch (error) {
        console.error("Error fetching product:", error)
      }
    }

    const fetchFeaturedImages = async () => {
      try {
        const query = `*[_type == "food"] {
          _id,
          "imageUrl": image.asset->url
        }`
        const data = await client.fetch(query)
        setFeaturedImages(data)
      } catch (error) {
        console.error("Error fetching featured images:", error)
      }
    }

    fetchProduct()
    fetchFeaturedImages()
  }, [params.id])

  useEffect(() => {
    const slider = sliderRef.current
    if (!slider || featuredImages.length === 0) return

    const slideImages = () => {
      if (!isHovering) {
        setTranslateX((prevTranslateX) => prevTranslateX - 0.5)
      }
    }

    const intervalId = setInterval(slideImages, 100)

    return () => clearInterval(intervalId)
  }, [featuredImages, isHovering])

  const handleAddToCart = () => {
    if (product) {
      addToCart({
        id: product._id,
        name: product.name,
        price: product.price,
        quantity: quantity,
        image: product.imageUrl,
      })
      toast.success(`${product.name} added to cart!`)
    }
  }

  if (!product) {
    return (
      <div>
        <Loader />
      </div>
    )
  }

  return (
    <div>
      <Hero heading="Shop Details" />
      <div className="min-h-screen bg-white py-8">
```

Browser Window 1 — Chocolate Dessert | Default

S Default   + Create   🔍   Structure   Vision   Schedules   Tasks

Content
- 📁 Food
- 📁 Chef

Food
🔍 Search list

- Chocolate Dessert
- Prawns Plate
- Chocolate Muffins
- Cheese Burger
- Chicken Nuggets
- 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

8:25 PM 2/7/2025

Browser Window 2 — William Rumi | Default

S 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

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

Perfect Taste
Classic Restaurant
$45.00

Shop Now ⊕
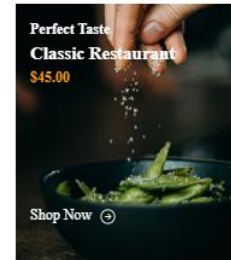
Filter By Price
100 — 500

$100 - $500   Filter

Latest Products
Fresh Breakfast
★★☆☆☆
14.5$

Product Tags

Services Our Menu

Kabab Burger

★★★★★ (5 reviews)

$220.00

Experience the taste in every bite! Our fresh and delicious food items make your special moments even more memorable. Prepared with the finest ingredients and authentic flavors, each dish brings a new delight to your table!

-	1	+	**Add to Cart**

Share: 🇫 🐦 📷 📌

Description

Spice up your meal with our Spicy Beef Kabab Burger, a perfect blend of flavors and textures. Made with succulent beef kababs, seasoned with aromatic spices, and grilled to perfection, this burger is a treat for meat lovers. Topped with fresh tomato slices and a hint of spicy sauce, every bite offers a burst of bold and zesty flavors. Served in a soft, toasted bun, this burger is ideal for those who crave a little heat in their food. Whether it's a quick snack or a hearty meal, our Spicy Beef Kabab Burger is sure to satisfy your cravings. Pair it with fries or a cold drink for the ultimate dining experience!

Featured Products

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 →
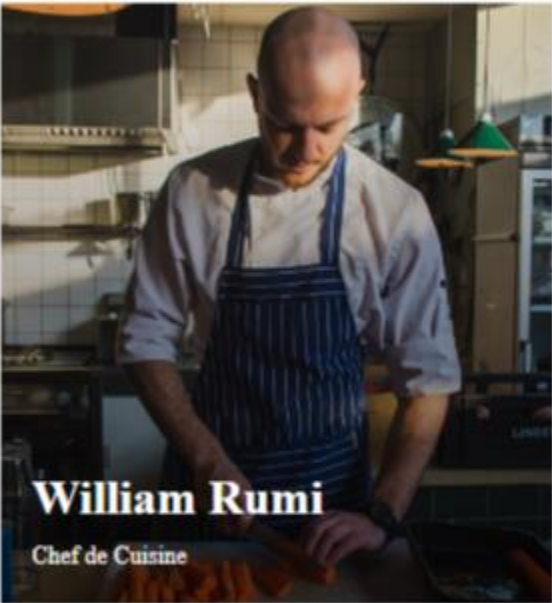

Munna Kathy
Culinary Instructor
View Details →

Jorina Begum
Sous Chef
View Details →

William Rumi

Chef de Cuisine

Chef Details

Specialty
Seafood Specialties

Experience
18 years

Rating
Master Chef

Availability
Available for Events

About William Rumi

Master of crafting exquisite seafood dishes with unique flavors.

Book William Rumi for Your Event