

Day 3 - API Integration Report - Q-Commerce Web

-----Food Tuck -----

Prepared by: Kashaf Noor

API Integration Process

API Integration Process

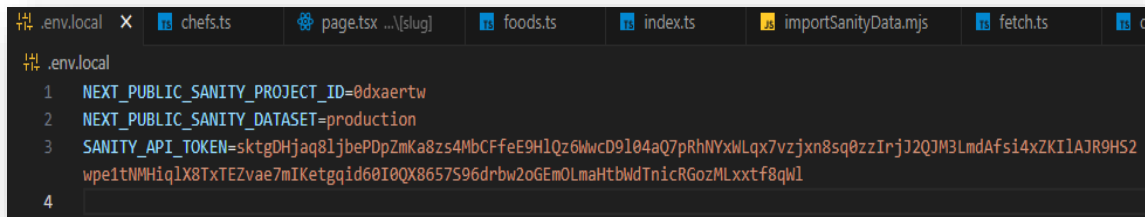
The API integration for the Q-Commerce web application was completed in several steps:

1. Understanding API Documentation:

- I thoroughly reviewed the API documentation of the third-party services used for food and chef data.
- Focused on identifying the key endpoints required for integration, such as food items, chef details, and real-time updates.

2. Setting Up Environment Variables:

- I set up environment variables in .env.local to securely store the necessary API keys and tokens for accessing the API:

A screenshot of a code editor with a dark theme. The top bar shows several open files: .env.local, chefs.ts, page.tsx ...[slug], foods.ts, index.ts, importSanityData.mjs, fetch.ts, and another .env.local. The active file is the first .env.local, which contains four lines of environment variables: 1. NEXT_PUBLIC_SANITY_PROJECT_ID=0dxaertw, 2. NEXT_PUBLIC_SANITY_DATASET=production, 3. SANITY_API_TOKEN=sktgDHjaq8ljbePDpZmKa8zs4MbCFfeE9HlQz6Wwcd9l04aQ7pRhNYxwLqx7vzjxn8sq0zzIrjJ2QJM3LmdAfsi4xZKI1AJR9HS2wpe1tNMHiqIX8TxTEZvae7mIKetgqid60IQX8657S96drbw2oGEmOLmaHtbWdTnicRGozMLxxtf8qwl, and 4. (empty line).

```
.env.local
1  NEXT_PUBLIC_SANITY_PROJECT_ID=0dxaertw
2  NEXT_PUBLIC_SANITY_DATASET=production
3  SANITY_API_TOKEN=sktgDHjaq8ljbePDpZmKa8zs4MbCFfeE9HlQz6Wwcd9l04aQ7pRhNYxwLqx7vzjxn8sq0zzIrjJ2QJM3LmdAfsi4xZKI1AJR9HS2wpe1tNMHiqIX8TxTEZvae7mIKetgqid60IQX8657S96drbw2oGEmOLmaHtbWdTnicRGozMLxxtf8qwl
4
```

3. Endpoint Integration:

- I integrated various endpoints to fetch food item data and chef details.
- Added error handling and response validation to ensure the integration is reliable and consistent.

Adjustments Made to Schemas

1. Sanity Schema Updates:

- I updated the food schema to accommodate additional fields such as price, ingredients, and availability, ensuring it aligns with the API responses:

FOOD SCHEMA

```
2.
3. export default {
4.   name: 'food',
5.   type: 'document',
6.   title: 'Food',
7.   fields: [
8.     {
9.       name: 'name',
10.      type: 'string',
11.      title: 'Food Name',
12.    },
13.    {
14.      name: 'slug',
15.      type: 'slug',
16.      title: 'Slug',
17.      options: {
18.        source: 'name'
19.      }
20.    },
21.    {
22.      name: 'category',
23.      type: 'string',
24.      title: 'Category',
25.      description:
26.        'Category of the food item (e.g., Burger, Sandwich, Drink,
27.        etc.)',
28.    },
29.    {
30.      name: 'price',
31.      type: 'number',
32.      title: 'Current Price',
33.    },
34.    {
35.      name: 'originalPrice',
36.      type: 'number',
37.      title: 'Original Price',
38.      description: 'Price before discount (if any)',
39.    },
40.    {
41.      name: 'tags',
42.      type: 'array',
43.      title: 'Tags',
44.      of: [{ type: 'string' }],
45.      options: {
46.        layout: 'tags',
```

```

47.     },
48.     description: 'Tags for categorization (e.g., Best Seller,
Popular, New)',
49.   },
50.   {
51.     name: 'image',
52.     type: 'image',
53.     title: 'Food Image',
54.     options: {
55.       hotspot: true,
56.     },
57.   },
58.   {
59.     name: 'description',
60.     type: 'text',
61.     title: 'Description',
62.     description: 'Short description of the food item',
63.   },
64.   {
65.     name: 'available',
66.     type: 'boolean',
67.     title: 'Available',
68.     description: 'Availability status of the food item',
69.   },
70. ],
71. };
72.

```

I also modified the chef schema, adding fields for expertise and experience to enrich the chef data:

CHEF SCHEMA

```

export default {
  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',
    },
  ],
};

```

Migration Steps and Tools Used

1. Tools Utilized:

- I used `@sanity/client` to migrate the data from the API into the Sanity CMS.

2. Migration Steps:

- I fetched food and chef data from the external API.
- Transformed the API data to fit the updated Sanity schemas.

- Uploaded the transformed data to Sanity using the following migration script:

```
3. import { createClient } from 'next-sanity'
4.
5. import { apiVersion, dataset, projectId } from '../env'
6.
7. export const client = createClient({
8.   projectId,
9.   dataset,
10.  apiVersion,
11.  useCdn: true, // Set to false if statically generating pages, using
    ISR or tag-based revalidation
12.})
13.
```

```
import { defineQuery, groq } from "next-sanity";

export const allchefs = defineQuery(`
  *[_type == "chef"]{
    _id,
    name,
    position,
    experience,
    specialty,
    image {
      asset -> {
        _id,
        url
      }
    },
    description,
  }`);

export const fourchefs = groq`*[_type == "chef"][0..3]`;

export const allfoods = groq`*[_type == "food"]`;

export const fourfoods = groq`*[_type == "food"][0..3]`
```

```
export interface Food {
  _id: string;           // Unique identifier for the food item
```

```

    name: string;           // Name of the food item
    _type : "food";
    category?: string;      // Category of the food item (optional)
    price: number;          // Current price of the food item
    originalPrice?: number; // Original price before discount (optional)
    tags?: string[];        // Tags for categorization (optional)
    image: { asset: { _ref: string; _type: "image" } }; // Image associated
with the food item (optional)
    description?: string;   // Description of the food item (optional)
    available: boolean;
    slug :{
      _type : "slug";
      current :string
    }
  }
}

```

For chef

```

type Chef = {
  _id: string;           // Unique identifier for the chef document
  name: string;          // Chef's name
  position: string;      // Position or role of the chef
  experience: number;    // Years of experience the chef has
  specialty: string;     // Chef's area of expertise (e.g., Italian, Pastry)
  image: { asset: { _id: string; url: string } }; // Image
associated with the chef, with optional asset ID and URL
  description: string;   // A short bio or description of the chef
};

export default async function Page() {
  const chefs : Chef[] = await sanityFetch({query: allchefs})
}

```

for food

```

interface FoodPageProps{
  params: Promise<{slug:string}>
}

async function getProduct(slug:string): Promise<Food> {
  return client.fetch(
    groq`*[_type == "food" && slug.current == $slug][0]{
      _id,
      name,
      category,
    }
  `
  );
}

```

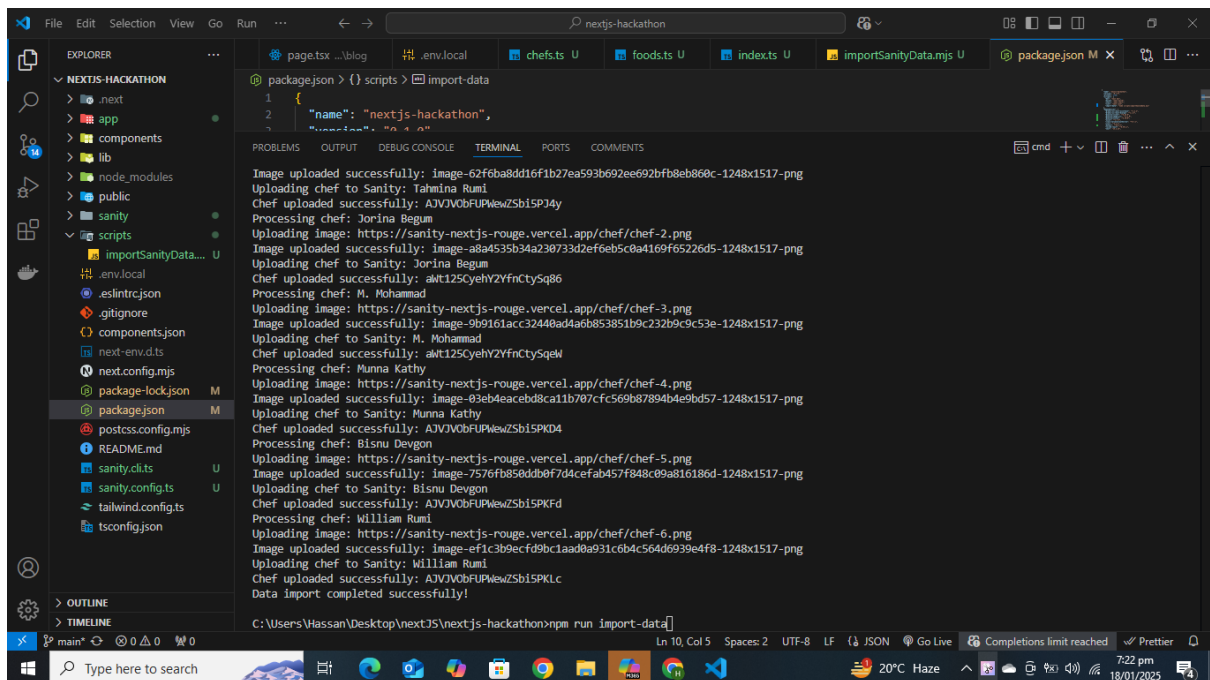
```

    price,
    originalPrice,
    tags,
    image {
      asset -> {
        _id,
        url
      }
    },
    description,
    available,
  }, {slug}
)
}

export default async function FoodPage({params}:FoodPageProps){
  const {slug} = await params
  const food =await getProduct(slug)

```

1. API Calls:



2. Data Displayed on Frontend:

CHEF



Tahmina Rumi

Position: Head Chef
Experience: 12 years
Specialty: Italian Cuisine
Description: Expert in crafting authentic Italian dishes and pastries.



Munna Kathy

Position: Culinary Instructor
Experience: 15 years
Specialty: Asian Fusion
Description: Pioneer in Asian fusion dishes blending traditional flavors with modern techniques.



Bisnu Devgon

Position: Executive Chef
Experience: 20 years
Specialty: Global Cuisine
Description: Expert in international cuisines and menu planning.




FOOD

hackat! x Create x Elite - F x Create x ChatGPT x GIAIC x Create x Micros x Sign up x New To x +


localhost:3000/shop

Home Menu Blog Pages About Shop Contact


search ...



Fresh Lime
Drink
Refreshing fresh lime drink made with natural ingredients.
\$38 \$45
Available
Healthy Popular

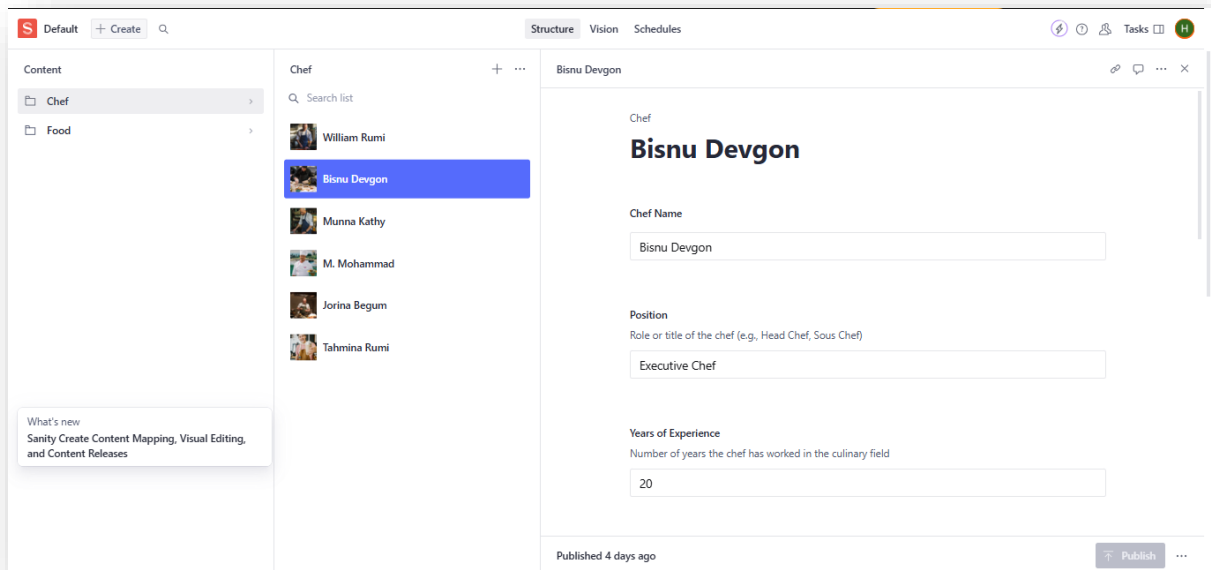


Country Burger
Sandwich
Classic country-style burger served with fries.
\$45 \$50
Available
Recommended



Pizza
Main Course
Delicious vegetarian pizza topped with fresh vegetables and cheese.
\$43 \$50
Available
Cheesy Vegetarian

3. Populated Sanity CMS Fields:



This report outlines the steps I took to integrate APIs into the Q-commerce website, update schemas, and migrate the data into Sanity CMS.