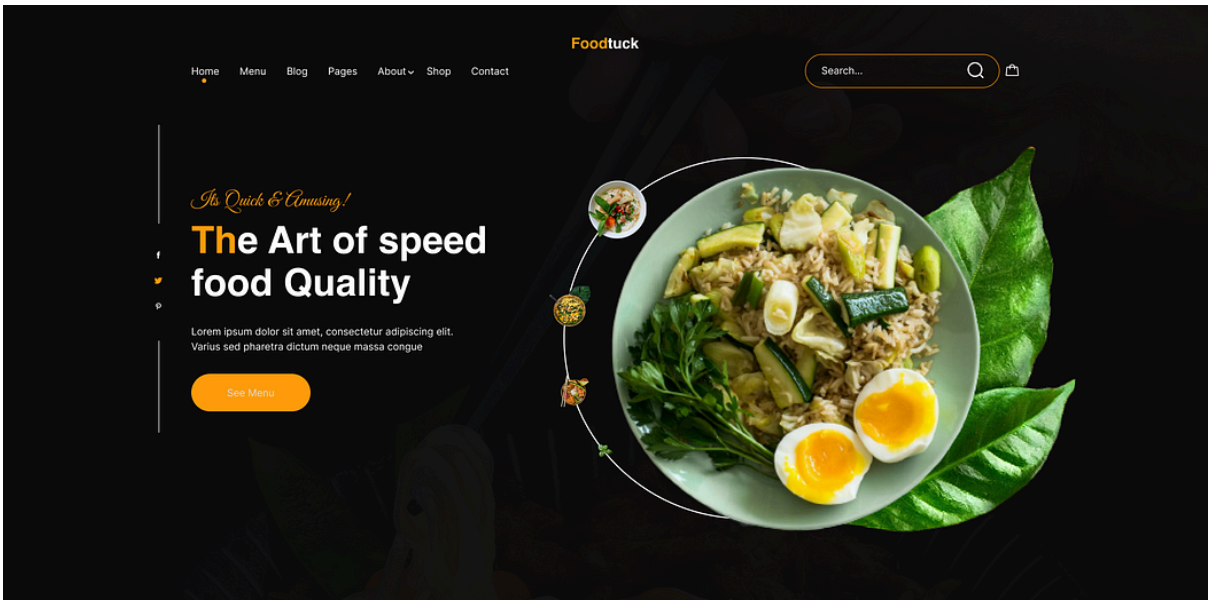
Integrating Sanity with Next.js:



Importing Food and Chef Data from an External API

In this blog post, we will explore how to integrate Sanity into a Next.js project for a restaurant website. Our focus will be on setting up environment variables, creating schemas for food and chef entries, and importing data from an external API.

This guide assumes you have an existing Next.js project and Sanity installed.

Table of Contents

1. Setting Up Environment Variables

Begin by configuring your environment variables. If a `.env.local` file doesn't exist in your project's root directory, create one. Add the following variables:

```
NEXT_PUBLIC_SANITY_PROJECT_ID=your_project_id
```

```
NEXT_PUBLIC_SANITY_DATASET=production
```

```
SANITY_API_TOKEN=your_sanity_token
```

Ensure sensitive tokens are securely stored and not exposed to the browser unless necessary.

2. Fetching Sanity Project ID and API Token

Project ID

1. Log in to Sanity.
2. Select your project.
3. Locate the Project ID in the dashboard and add it to your `.env.local` file.

API Token

1. Navigate to the “API” tab in your project.
2. Click “Add API Token” under “Tokens.”
3. Name the token, set permissions, and copy the generated token for use in `.env.local`.

3. Creating the Sanity Schemas

Food Schema (`food.ts`)

```
export default {

 name: 'food',

 type: 'document',

 title: 'Food',

 fields: [

   { name: 'name', type: 'string', title: 'Food Name' },

   { name: 'description', type: 'string', title: 'Description' },

   { name: 'price', type: 'number', title: 'Price' },

   { name: 'rating', type: 'number', title: 'Rating' },

   { name: 'ratingCount', type: 'number', title: 'Rating Count'
},
```

```
    { name: 'tags', type: 'array', title: 'Tags', of: [{ type: 'string' }] },

    { name: 'image', type: 'image', title: 'Image', options: { hotspot: true } },

  ],

};
```

Chef Schema (`chef.ts`)

```
export default {

  name: 'chef',

  type: 'document',

  title: 'Chef',

  fields: [

    { name: 'name', type: 'string', title: 'Chef Name' },

    { name: 'position', type: 'string', title: 'Position' },
```

```
    { name: 'experience', type: 'string', title: 'Experience' },

    { name: 'specialty', type: 'string', title: 'Specialty' },

    { name: 'description', type: 'string', title: 'Description' },

    { name: 'available', type: 'boolean', title: 'Available' },

    { name: 'image', type: 'image', title: 'Image', options: { hotspot: true } },

  ],

};
```

Update your `schemaTypes/index.ts` to include these schemas.

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

import food from './food';

import chef from './chef';

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

```
 types: [food, chef],

};
```

4. Writing the Data Import Script

Create a script to fetch and import data from an external API into Sanity.

scripts/importSanityData.mjs

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

import axios from 'axios';

import dotenv from 'dotenv';

import { fileURLToPath } from 'url';

import path from 'path';

// Load environment variables

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 {

   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(),

    });

    return asset._id;

  } catch (error) {

    console.error('Image upload failed:', error);

    return null;

  }

}

async function importData() {

  try {

    const [foodsResponse, chefsResponse] = await
Promise.all([
```

```
axios.get('https://sanity-nextjs-rouge.vercel.app/api/foods'),

axios.get('https://sanity-nextjs-rouge.vercel.app/api/chefs'),

  ]);

  const foods = foodsResponse.data;

  const chefs = chefsResponse.data;

  for (const food of foods) {

    let imageRef = null;

    if (food.image) {

      imageRef = await uploadImageToSanity(food.image);

    }

    const sanityFood = {

      _type: 'food',
```

```
        name: food.title,

        description: food.description,

        price: food.price,

        rating: food.rating?.rate || 0,

        ratingCount: food.rating?.count || 0,

        tags: food.category ? [food.category] : [],

        image: imageRef ? { _type: 'image', asset: {
_type: 'reference', _ref: imageRef } } : undefined,

      };

      await client.create(sanityFood);

    }

    for (const chef of chefs) {

      let imageRef = null;

      if (chef.image) {
```

```
        imageRef = await uploadImageToSanity(chef.image);

      }

      const sanityChef = {

        _type: 'chef',

        name: chef.name,

        position: chef.position,

        experience: chef.experience,

        specialty: chef.specialty,

        description: chef.description,

        available: chef.available,

        image: imageRef ? { _type: 'image', asset: {
_type: 'reference', _ref: imageRef } } : undefined,

      };

      await client.create(sanityChef);
```

```
    }

  } catch (error) {

    console.error('Data import failed:', error);

  }

}

importData();
```

5. Running the Import Script

Install dependencies:

```
npm install @sanity/client axios dotenv
```

Add a script to your `package.json`:

```
"scripts": {

  "import-data": "node scripts/importSanityData.mjs"
```

```
}
```

Run the script:

```
npm run import-data
```

Conclusion

By following this guide, you've learned how to set up Sanity integration in a Next.js project, create custom schemas for food and chef entries, and import data from an external API. This approach allows you to manage and display dynamic content seamlessly in your Next.js application.