# Day 3 - API Migration Report – Food Tuck

## Introduction: What is API Migration?

the process of selecting, preparing, extracting, and transforming data and permanently transferring it from one computer storage system to another

## **Steps of API Migration:**

1. Review all of JSON data:
   - Review all the data and adjust it according to schemas
   - Example

```json
[
{
    "name": "Munna Kathy",
    "position": "Culinary Instructor",
    "experience": 15,
    "specialty": "Asian Fusion",
    "image": "https://sanity-nextjs-rouge.vercel.app/chef/chef-4.png",
    "description": "Pioneer in Asian fusion dishes blending traditional
flavors with modern techniques.",
    "available": true
  },
  {
    "name": "Bisnu Devgon",
    "position": "Executive Chef",
    "experience": 20,
    "specialty": "Global Cuisine",
    "image": "https://sanity-nextjs-rouge.vercel.app/chef/chef-5.png",
    "description": "Expert in international cuisines and menu planning.",
    "available": true
  },
  {
    "name": "William Rumi",
    "position": "Chef de Cuisine",
    "experience": 18,
    "specialty": "Seafood Specialties",
    "image": "https://sanity-nextjs-rouge.vercel.app/chef/chef-6.png",
    "description": "Master of crafting exquisite seafood dishes with unique
flavors.",
    "available": true
  }
]
```

# Day 3 - API Migration Report – Food Tuck

## 2. Script Creation

- Used **Node.js** for creating the script and the **Sanity Client** library for interacting with the database.

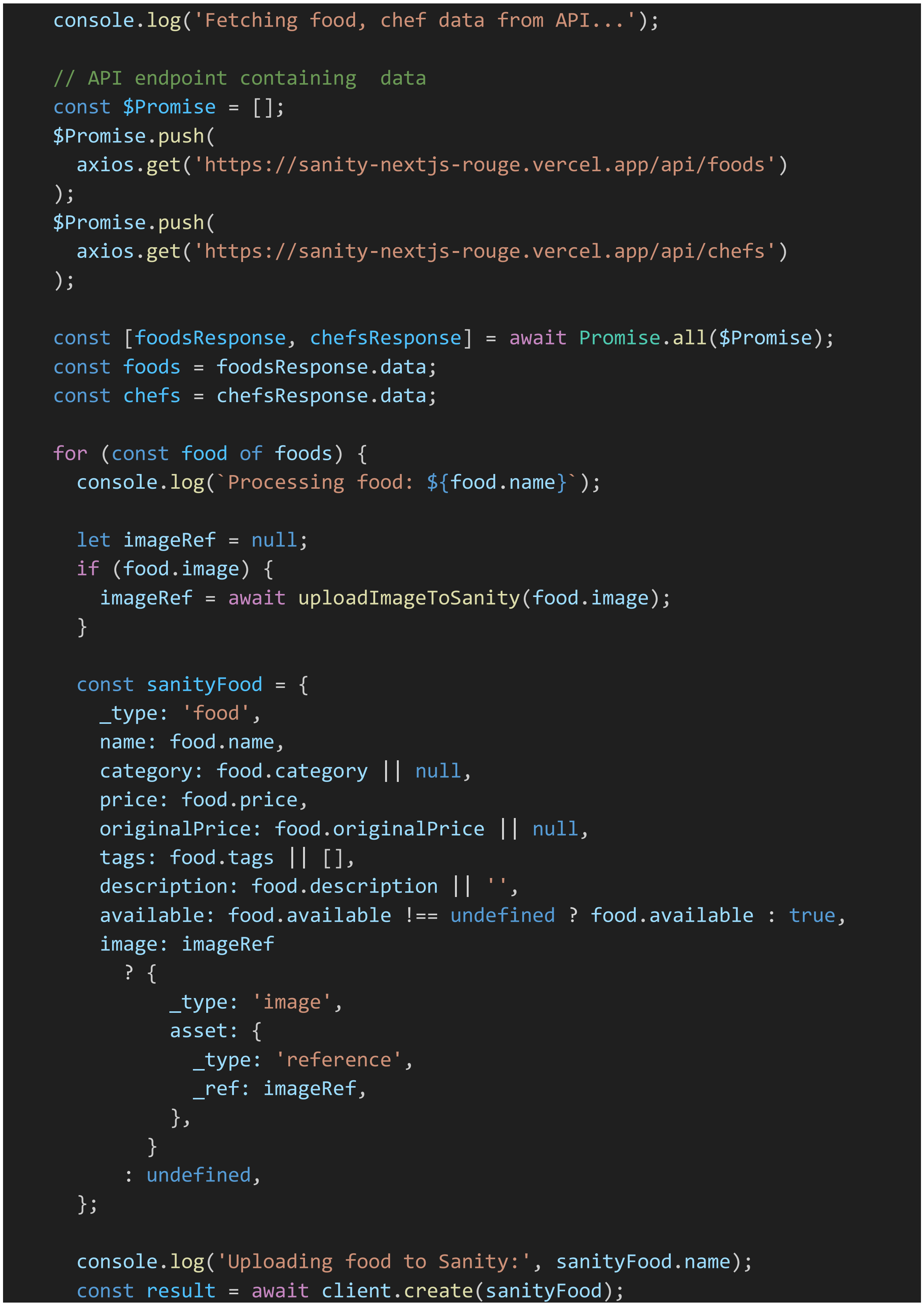
```
import { createClient } from '@sanity/client';
import axios from 'axios';
import dotenv from 'dotenv';
import { fileURLToPath } from 'url';
import path from 'path';

// 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',
});

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 [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();
```

# Day 3 - API Migration Report – Food Tuck

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

- And add `"import-data": "node scripts/importSanityData.mjs"`
- In scripts object of `package.json`
- Run it with `npm run import-data`

**Validation:**

With studio, we use vision to get queries

## Tools Used

- Sanity Client
- Node.js
- Thunder Client
- Sanity Studio
- JSON Files
- Tailwind CSS
- ShadCN