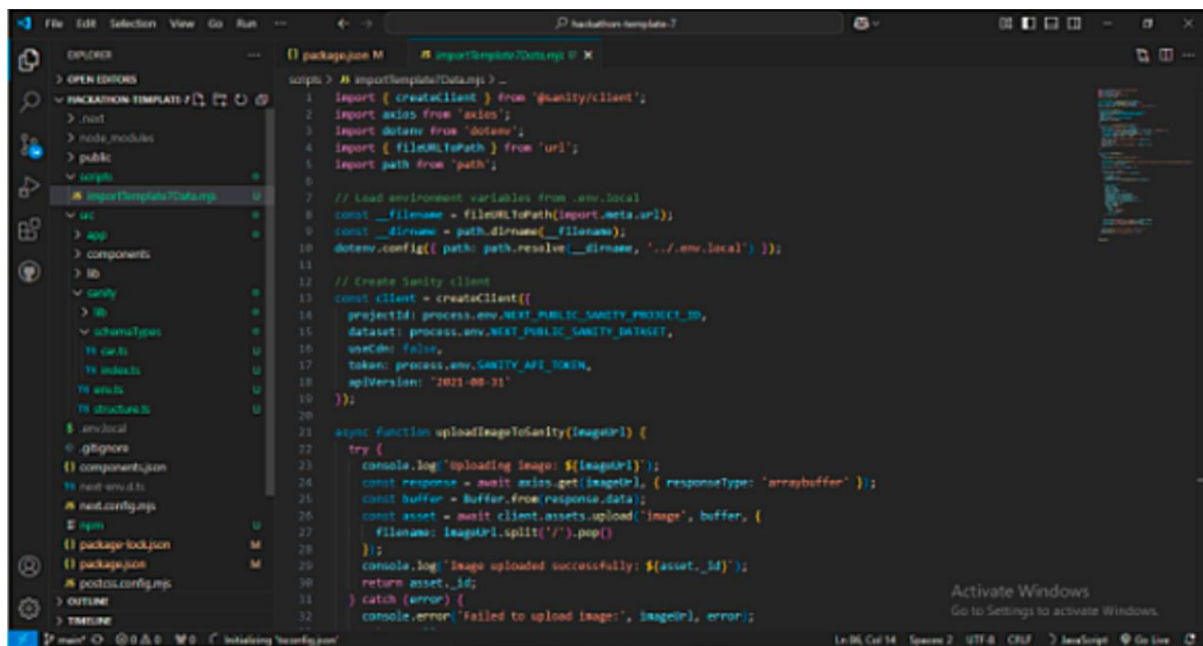


"Day 3 - API Integration Report - [Rental Car E-commerce]"

Steps For Day 3:

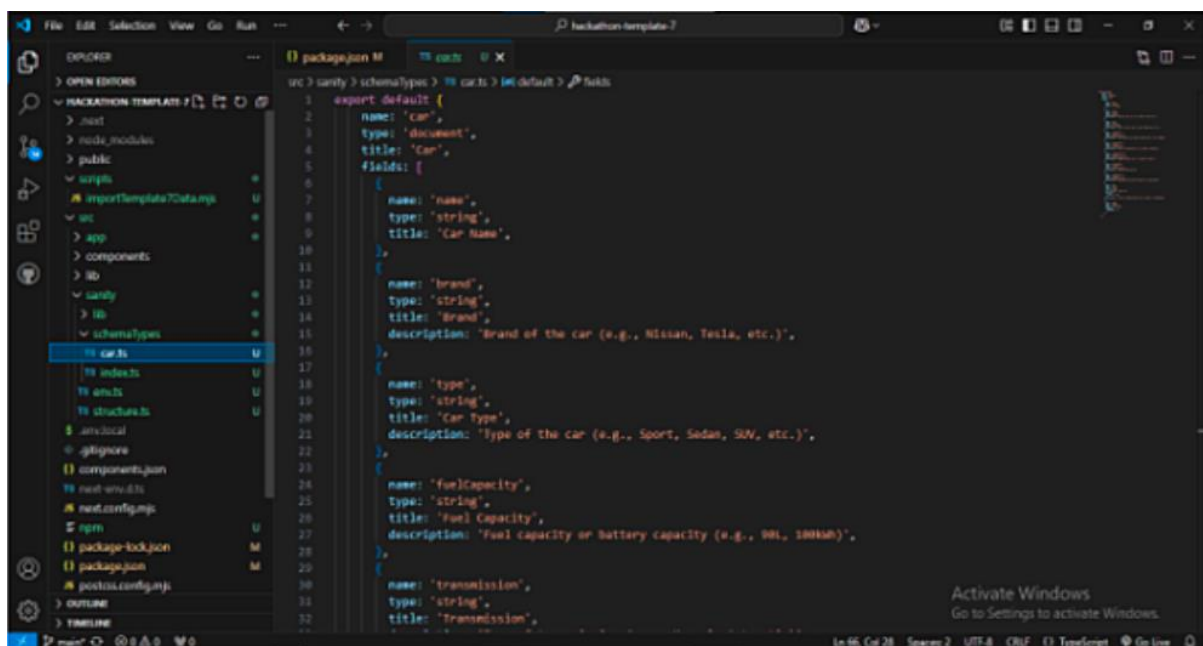
1. Understand the provided API:
2. Validate and adjust your schema:
3. API integration in Next.js:
4. API Integration in Next.js:

01. API Integration Process



```
importTemplateData.mjs
1 import { createClient } from '@sanity/client';
2 import axios from 'axios';
3 import dotenv from 'dotenv';
4 import { fileURLToPath } from 'url';
5 import path from 'path';
6
7 // Load environment variables from .env.local
8 const __filename = fileURLToPath(import.meta.url);
9 const __dirname = path.dirname(__filename);
10 dotenv.config({ path: path.resolve(__dirname, '../.env.local') });
11
12 // Create Sanity client
13 const client = createClient({
14   projectId: process.env.NEXT_PUBLIC_SANITY_PROJECT_ID,
15   dataset: process.env.NEXT_PUBLIC_SANITY_DATASET,
16   useCdn: false,
17   token: process.env.SANITY_API_TOKEN,
18   apiVersion: '2021-08-31'
19 });
20
21 async function uploadImageToSanity(imageUrl) {
22   try {
23     console.log('uploading image: ${imageUrl}');
24     const response = await axios.get(imageUrl, { responseType: 'arraybuffer' });
25     const buffer = Buffer.from(response.data);
26     const asset = await client.assets.upload('image', buffer, {
27       filename: imageUrl.split('/').pop()
28     });
29     console.log('image uploaded successfully: ${asset._id}');
30     return asset._id;
31   } catch (error) {
32     console.error('failed to upload image: ', imageUrl, error);
33   }
34 }
```

02. Adjustment made to Schemas



```
car.ts
1 export default {
2   name: 'car',
3   type: 'document',
4   title: 'Car',
5   fields: {
6     name: {
7       name: 'name',
8       type: 'string',
9       title: 'Car Name',
10     },
11     brand: {
12       name: 'brand',
13       type: 'string',
14       title: 'Brand',
15       description: 'Brand of the car (e.g., Nissan, Tesla, etc.)',
16     },
17     type: {
18       name: 'type',
19       type: 'string',
20       title: 'Car Type',
21       description: 'Type of the car (e.g., Sport, Sedan, SUV, etc.)',
22     },
23     fuelCapacity: {
24       name: 'fuelCapacity',
25       type: 'string',
26       title: 'Fuel Capacity',
27       description: 'Fuel capacity or battery capacity (e.g., 98L, 100kWh)',
28     },
29     transmission: {
30       name: 'transmission',
31       type: 'string',
32       title: 'Transmission',
33     },
34   },
35 }
```

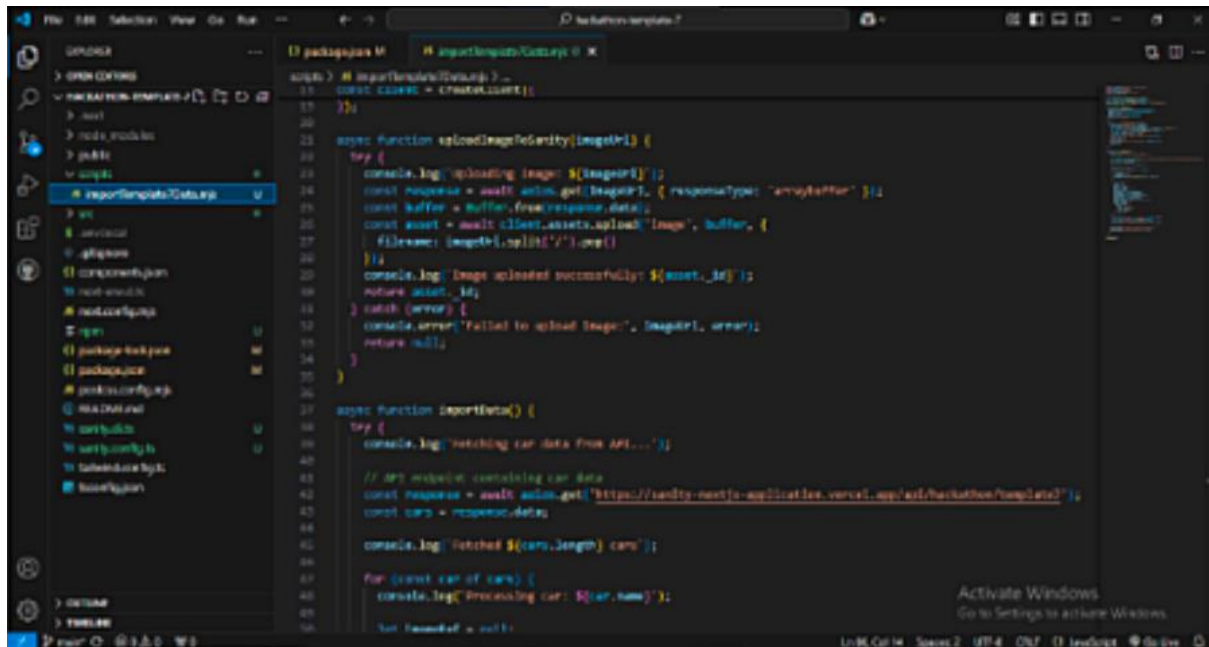
3. Data Migration Steps

Migration Process

step 1: fetch data from external API.

step 2: Transform the data to match Sanity schema.

step 3: Push the data into Sanity CMS.



```
const current = createdAt;

export async function uploadImageToSanity(imageUrl) {
  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 reportData() {
  console.log('fetching car data from API...');
  // api response containing car data
  const response = await axios.get('https://sanity-nextjs-application.vercel.app/api/hackathon/template2');
  const cars = response.data;

  console.log('Fetched ', cars.length, ' cars');

  for (const car of cars) {
    console.log('Processing car: ', car.name);
  }

  return cars;
}
```

Sanity CMS Dashboard

