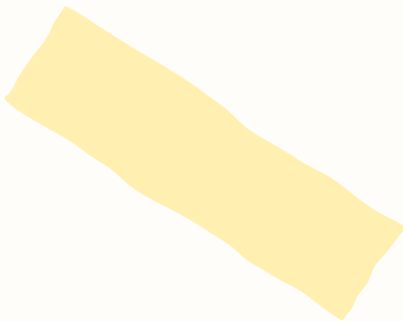
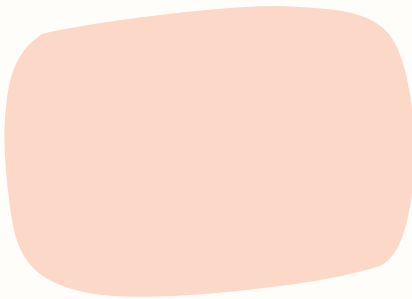


DAY 3 - API INTEGRATION REPORT - MORENT CAR RENTAL WEBSITE

Rental E-Commerce Website



How did i approch Api integration

My Approach

I checked the assigned API which was a good starting but i needed to adjust it for my needs

Assigned Api Structure

API: <https://sanity-nextjs-application.vercel.app/api/hackathon/template7>

```
{
  "id": 1,
  "name": "Koenigsegg",
  "type": "Sport",
  "fuel_capacity": "90L",
  "transmission": "Manual",
  "seating_capacity": "2 People",
  "price_per_day": "$99.00",
  "image_url": "https://car-rental-website-five.vercel.app/_next/image?url=%2F_next%2Fstatic%2Fmedia%2Fcar.11698147.jpg&w=640&q=75",
  "tags": [
    "popular"
  ]
},
```

My Api Structure

API: <https://morent-way-to-rent-car.vercel.app/api/cars>

```
{
  "heart": true,
  "reviews": 20,
  "seating_capacity": "2 People",
  "image": {
    "_type": "image",
    "asset": {
      "_ref": "image-8d4b40b0870d3054a95e666e8a9c75191612f1d3-232x72-jpg",
      "_type": "reference"
    }
  },
  "slug": {
    "current": "koenigsegg",
    "_type": "slug"
  },
  "fuel_capacity": "90L",
  "desc": "Car description",
  "icons": true,
  "available": 5,
  "name": "Koenigsegg",
  "tags": [
    "popular"
  ],
  "price_per_day": "$99.00",
  "original_price": "$100.00",
  "card_type": "mobile",
  "type": "Sport",
  "transmission": "Manual"
},
```

Data Migration

I copied assigned migration script and ran it blindly and got errors 😬 LoL, so i had to adjust it for myself

Assigned Migration Script

API: <https://github.com/AsharibAli/sanity-nextjs/blob/main/scripts/importTemplate7Data.mjs>

My Migration Script

API: <https://github.com/DanielHashmi/Morent---Way-to-Rent-Cars/blob/main/src/sanity/lib/migration.mjs>

[This Link Might Not Work Before Hackathon Deadline](#)

Changes i Made

Environment Variables

The process.env.VARIABLE_NAMES didn't worked for me for some reason and i didn't bother myself to figure it out why, i choosed to add the actual projectId and token directly since its only for one time use so...

Custom Fields

I added custom fields for my specific needs, such as **heart** for favorites, **slug** for a readable and seo friendly unique key for each car, **available** for the rental car quantity, **desc** for the description and **reviews** to show rating on the card for the car

My Preference for slug instead of using the sanity_id

Personally i'm familiar with slugs and i find it very easy to handle it inside my code, but some people might like using id's, the reason why i don't like id's to use inside my dynamic routes is because it doesn't directly fits inside the url it's fragile for that we need to use other methods to convert it from an actual id to an index but if we have a predefined slug inside the car data itself so we can directly add it inside the url and also its recommended to use slugs instead of id's

Challenges i faced

Well, the whole hackathon is a challenge for me, but adding the custom fields and then filling them with data inside sanity for each car manually 🔥

```
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: '...',
  dataset: 'production',
  useCdn: false,
  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 car data from API...');

    // API endpoint containing car data
    const response = await axios.get('https://sanity-nextjs-application.vercel.app/api/hackathon/template7');
    const cars = response.data;

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

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

      let imageUrl = null;
      if (car.image.url) {
        imageUrl = await uploadImageToSanity(car.image.url);
      }

      const sanityCar = {
        // Api Fields
        _type: 'car',
        name: car.name,
        type: car.type,
        fuel_capacity: car.fuel_capacity,
        transmission: car.transmission,
        seating_capacity: car.seating_capacity,
        price_per_day: car.price_per_day,
        original_price: car.original_price || null,
        tags: car.tags || [],
        image: imageUrl ? {
          _type: 'image',
          asset: {
            _type: 'reference',
            _ref: imageUrl,
          },
        } : undefined,
        // Custom Fields
        slug: car.name.toLowerCase().replace(' ', '-'),
        heart: false,
        available: 5,
        // Detail Page Fields
        desc: 'Car description here',
        reviews: 20,
      };

      console.log('Uploading car to Sanity', sanityCar.name);
      const result = await client.create(sanityCar);
      console.log('Car uploaded successfully: ', result._id);
    }

    console.log('Data import completed successfully!');
  } catch (error) {
    console.error('Error importing data:', error);
  }
}

importData();
```

Fetching and Displaying Data

I created `api/cars/route.ts` file inside my nextjs app folder

`route.ts`

i wrote this for `/api/cars` endpoint

```
1 import client from "@sanity/lib/client";
2 import { CardQuery } from "@sanity/lib/grok";
3 import { CAR } from "@types/types";
4 import { NextResponse } from "next/server";
5
6 export const GET = async (req: any) => {
7   const limit = req.nextUrl.searchParams.get("limit");
8   const slug = req.nextUrl.searchParams.get("slug");
9
10  try {
11    if (limit) {
12      let cars: CAR[] = await client.fetch(CardQuery);
13      return NextResponse.json(cars.slice(0, limit));
14    }
15    if (slug) {
16      let cars: CAR[] = await client.fetch(CardQuery);
17      return NextResponse.json(cars.find(car => car.slug.current === slug));
18    }
19    let cars: CAR[] = await client.fetch(CardQuery);
20    return NextResponse.json(cars);
21  } catch (error) {
22    console.error('No internet or an error occurred.', error);
23    return NextResponse.json(
24      { error: 'An error occurred while fetching data.' },
25      { status: 500 }
26    );
27  }
28 };
29
```

Endpoint: `/api/cars`

Will fetch all cars

Endpoint: `/api/cars?limit=3`

Will fetch 3 cars

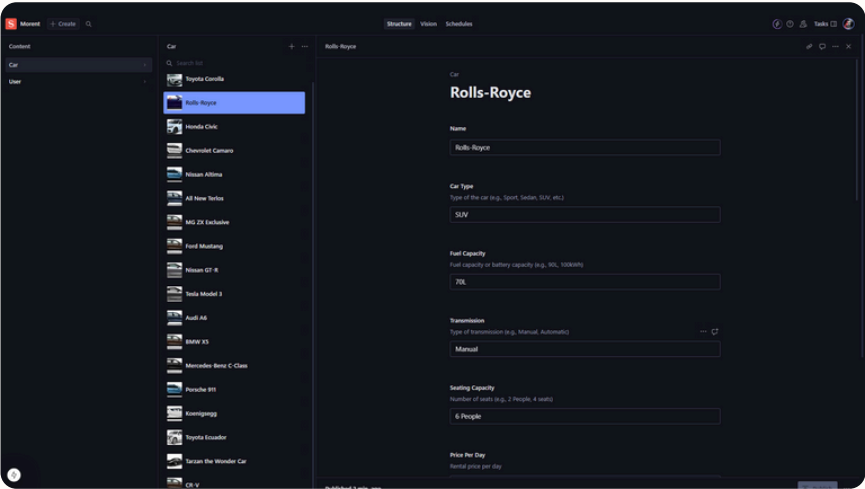
Endpoint: `/api/cars?slug=honda-civic`

Will fetch honda-civic car

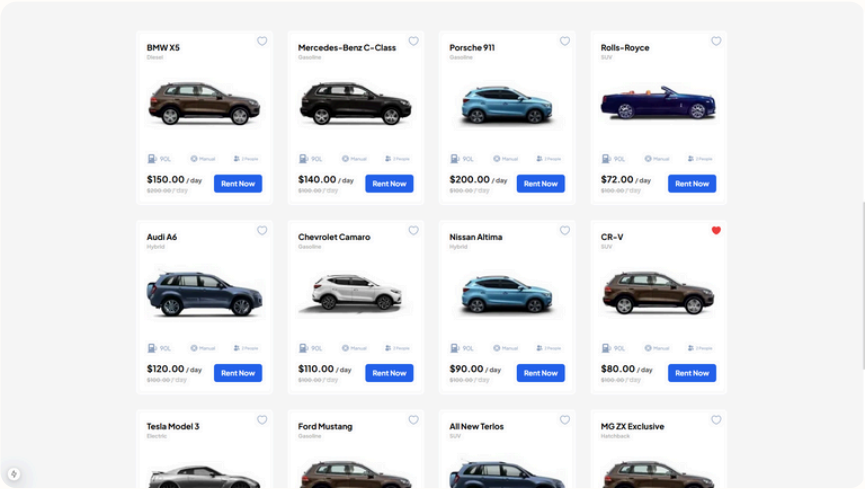
This is how i fetched data and displayed it to the frontend

```
try {
  let data = await fetch(`${process.env.NEXT_PUBLIC_BASE_URL}/api/cars`);
  carDetails = await data.json();
} catch (error) {
  console.log('No internet! or something else occurred.', error);
}
```

Sanity Populated with Data ScreenShot



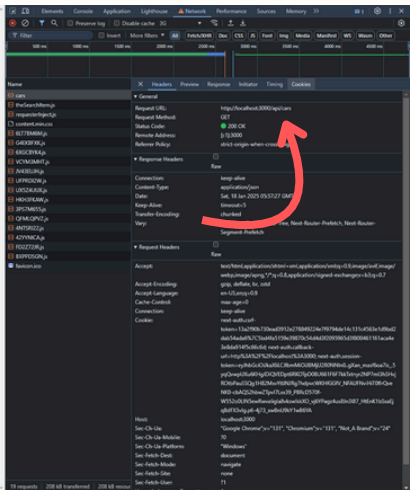
Displayed Cars ScreenShot



Api: /api/cars

```
{
  "cars": [
    {
      "id": 1,
      "make": "Toyota",
      "model": "Camry",
      "year": 2018,
      "color": "Silver",
      "status": "Available",
      "price": 25000,
      "mileage": 15000,
      "features": [
        "Automatic",
        "Air Conditioning",
        "Power Windows",
        "ABS Brakes",
        "Cruise Control"
      ],
      "description": "A reliable and comfortable sedan with a spacious interior and excellent fuel economy."
    },
    {
      "id": 2,
      "make": "Honda",
      "model": "Civic",
      "year": 2019,
      "color": "Blue",
      "status": "Available",
      "price": 22000,
      "mileage": 10000,
      "features": [
        "Automatic",
        "Air Conditioning",
        "Power Windows",
        "ABS Brakes",
        "Cruise Control"
      ],
      "description": "A sporty and efficient hatchback with a sleek design and advanced safety features."
    },
    {
      "id": 3,
      "make": "Ford",
      "model": "Mustang",
      "year": 2020,
      "color": "Red",
      "status": "Available",
      "price": 30000,
      "mileage": 5000,
      "features": [
        "Automatic",
        "Air Conditioning",
        "Power Windows",
        "ABS Brakes",
        "Cruise Control"
      ],
      "description": "A powerful and stylish sports car with a high-performance engine and sleek aerodynamics."
    }
  ]
}
```

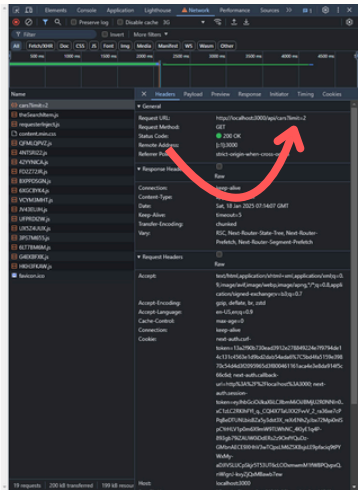
Response ScreenShot



Api: /api/cars?limit=2

```
{
  "cars": [
    {
      "id": 1,
      "make": "Toyota",
      "model": "Camry",
      "year": 2018,
      "color": "Silver",
      "status": "Available",
      "price": 25000,
      "mileage": 15000,
      "features": [
        "Automatic",
        "Air Conditioning",
        "Power Windows",
        "ABS Brakes",
        "Cruise Control"
      ],
      "description": "A reliable and comfortable sedan with a spacious interior and excellent fuel economy."
    },
    {
      "id": 2,
      "make": "Honda",
      "model": "Civic",
      "year": 2019,
      "color": "Blue",
      "status": "Available",
      "price": 22000,
      "mileage": 10000,
      "features": [
        "Automatic",
        "Air Conditioning",
        "Power Windows",
        "ABS Brakes",
        "Cruise Control"
      ],
      "description": "A sporty and efficient hatchback with a sleek design and advanced safety features."
    }
  ]
}
```

Response ScreenShot



Api: /api/cars?slug=honda-civic

Response ScreenShot

```
{
  "name": "Honda",
  "year": 2024,
  "make": "Honda Civic",
  "transmission": "Manual",
  "price": 25000,
  "available": true,
  "fuel_efficiency": "24 mpg",
  "original_price": "21000.00",
  "image": [
    "api/cars/honda-civic-2024-01-01-1000x1000.jpg"
  ],
  "slug": {
    "current": "honda-civic",
    "type": "slug"
  },
  "type": "Sedan",
  "seating_capacity": 5,
  "available": true,
  "tags": [
    "Honda Civic is a popular compact car known for its reliability, fuel efficiency, and practicality. It's available in various configurations including sedan, coupe, and hatchback, known for its smooth ride and spacious interior. The Civic appeals to a wide range of drivers from commuters to enthusiasts. It often features advanced technology and safety features, making it a top choice in its segment."
  ],
  "reviews": 3
}
```

The screenshot shows a web browser displaying the response of the API call. The response is a JSON object containing car details. A red arrow points from the 'image' field in the JSON to the 'image' field in the browser's response.

Header	Value
Content-Type	application/json
Response-URL	https://api.honda-cars.com/api/cars/honda-civic
Status-Code	200 OK
Response-Address	127.0.0.1
Response-Port	3000
Response-Header	Content-Type: application/json
Response-Body	{...}