

API INTEGRATION REPORT FOR RENTAL ECOMMERCE

Introduction:

Day 3 of the **hackathon** was all about diving into the exciting world of **APIs** and exploring how to bring data to life through **integration** and **migration**. The focus was on building a functional marketplace backend by connecting external **APIs** and using **Sanity CMS** as a powerful content management tool.

This report will guide you step by step on how to seamlessly **integrate APIs into a Next.js project** and **migrate the data into Sanity CMS** for a dynamic and scalable solution. By combining the flexibility of APIs, the structured power of Sanity, and the visual appeal of Tailwind CSS, you'll see how we created a stunning front-end experience. With snapshots at every step, you'll get an in-depth look at the process, from setting up schemas to fetching data and rendering it beautifully on the front end.

Let's dive into this hands-on journey of transforming data into a marketplace-ready application!

Content:

- ❖ Introduction:
- ❖ Setting up environment variable
- ❖ Getting your sanity projectId and API Token
- ❖ Creating the Sanity Schema
- ❖ Setting up the Data import Script
- ❖ Install the PackageError! Bookmark not defined.
- ❖ Running the Migration Script
- ❖ Verifying Data Migration in SanityError! Bookmark not defined.
- ❖ Fetch Data from Sanity
- ❖ Frontend Fetch Data
- ❖ Functionality Wishlist Feature
- ❖ Category Page
- ❖ Conclusion

1. Setting Up Environment Variables

First, Create a .env.local file in your project root (if it doesn't exist).

```
NEXT_PUBLIC_SANITY_PROJECT_ID=your_project_id  
NEXT_PUBLIC_SANITY_DATASET=production  
SANITY_API_TOKEN=your_sanity_token
```

Add these variables:

- NEXT_PUBLIC_SANITY_PROJECT_ID
- NEXT_PUBLIC_SANITY_DATASET
- SANITY_API_TOKEN

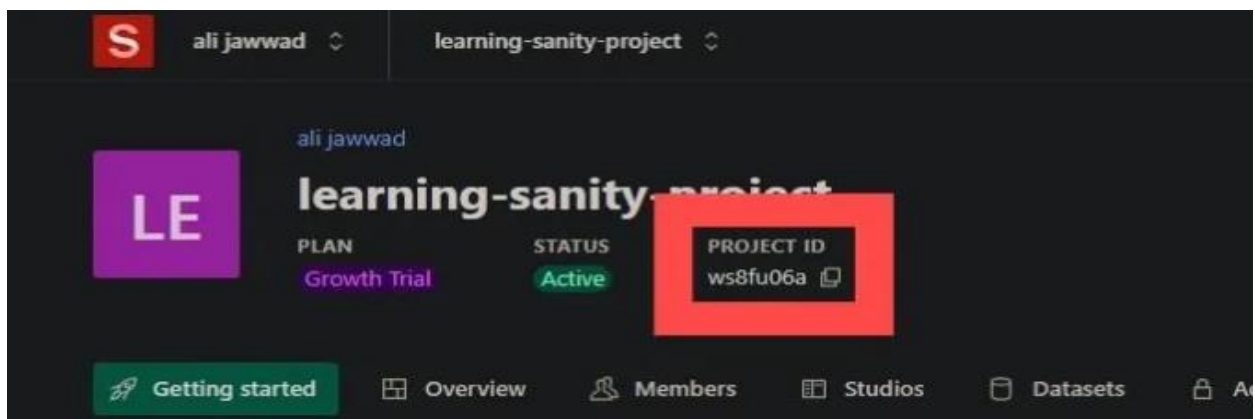
Note: Variables starting with NEXT_PUBLIC_ will be exposed to the browser.

✓ 2. Getting Your Sanity Project ID and API Token

Project ID

To find your Sanity project ID:

1. Log in to [Sanity](#).
2. Select your project.
3. Find the project ID in the dashboard.



DAY 3: MARKETPLACE BUILDER HACKATHON 2025

Use this ID for NEXT_PUBLIC_SANITY_PROJECT_ID in your .env.local file.

API Token:

To generate a Sanity API token:

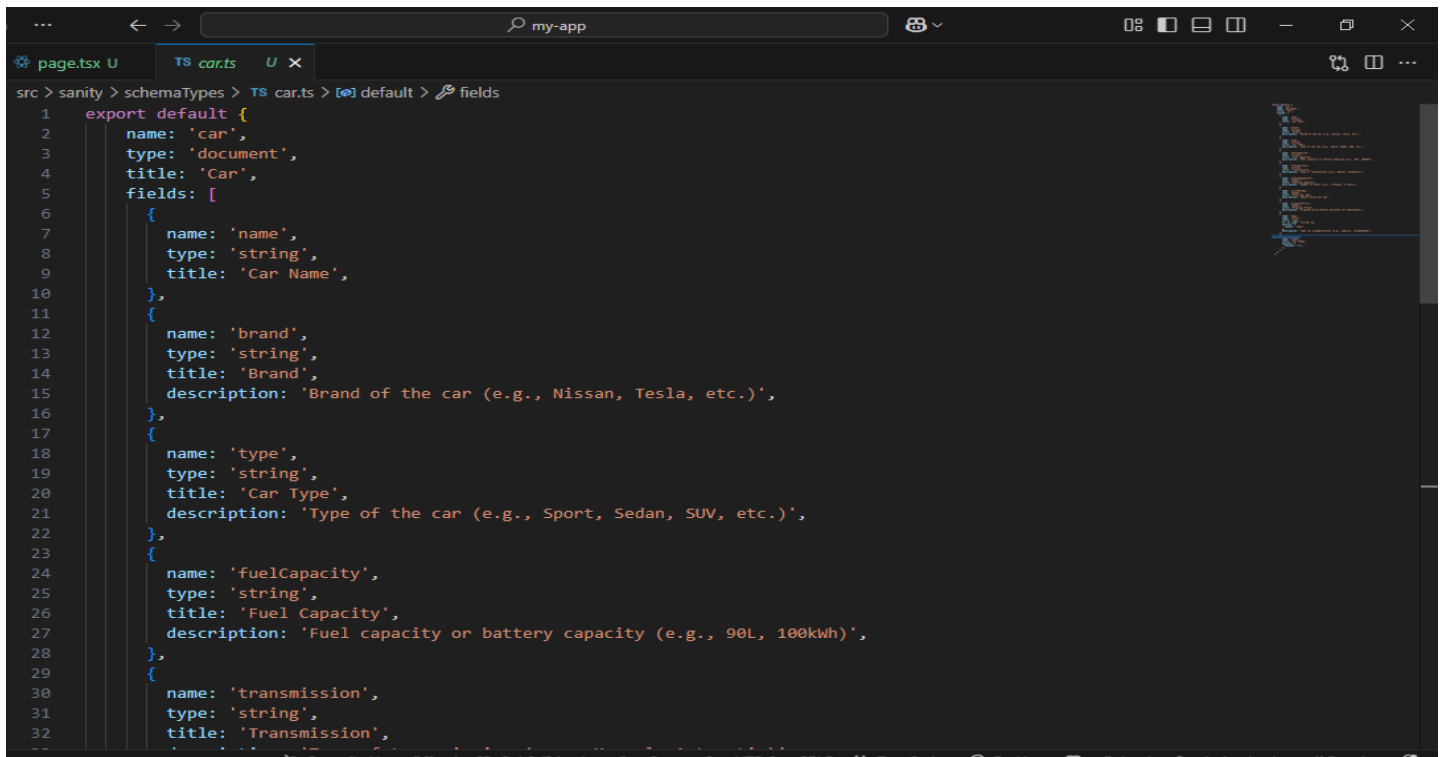
1. Go to [Sanity](#) and select your project.
2. Go to the "API" tab.
3. Click "Add API token" under "Tokens".
4. Name the token and set permissions (choose "Editor" or "Developer" for full access).
5. Copy the generated token.

Dataset:

Your **dataset** will always be **production**.

3. Creating the Sanity Schema

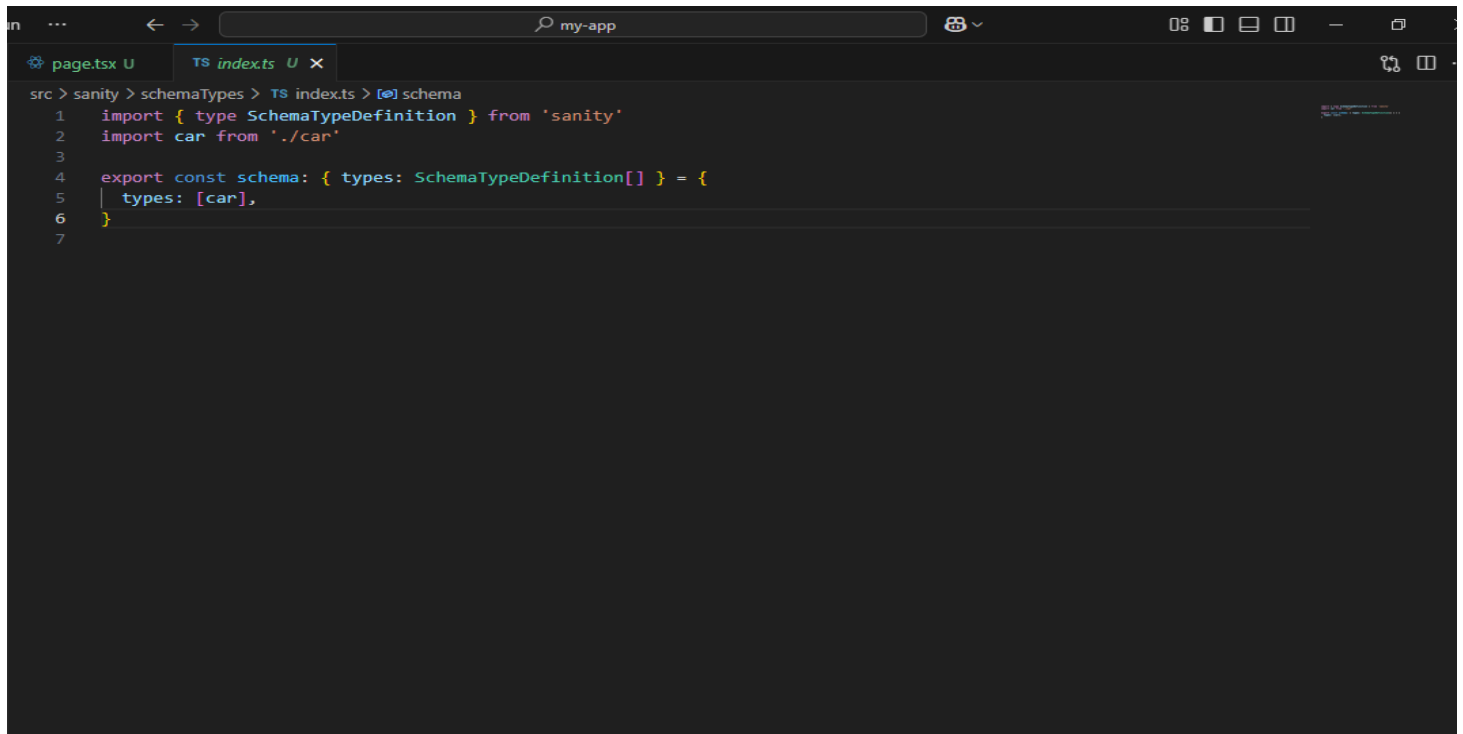
In your sanity/schemaTypes folder, create a file called car.ts.



```
src > sanity > schemaTypes > TS car.ts > default > fields
1  export default {
2    name: 'car',
3    type: 'document',
4    title: 'Car',
5    fields: [
6      {
7        name: 'name',
8        type: 'string',
9        title: 'Car Name',
10       },
11      {
12        name: 'brand',
13        type: 'string',
14        title: 'Brand',
15        description: 'Brand of the car (e.g., Nissan, Tesla, etc.)',
16       },
17      {
18        name: 'type',
19        type: 'string',
20        title: 'Car Type',
21        description: 'Type of the car (e.g., Sport, Sedan, SUV, etc.)',
22       },
23      {
24        name: 'fuelCapacity',
25        type: 'string',
26        title: 'Fuel Capacity',
27        description: 'Fuel capacity or battery capacity (e.g., 90L, 100kWh)',
28       },
29      {
30        name: 'transmission',
31        type: 'string',
32        title: 'Transmission',
33      }
```

DAY 3: MARKETPLACE BUILDER HACKATHON 2025

In your `sanity/schemaTypes/index.ts`, add the new car schema.



```
src > sanity > schemaTypes > TS index.ts > schema
1  import { type SchemaTypeDefinition } from 'sanity'
2  import car from './car'
3
4  export const schema: { types: SchemaTypeDefinition[] } = {
5    | types: [car],
6  }
7
```

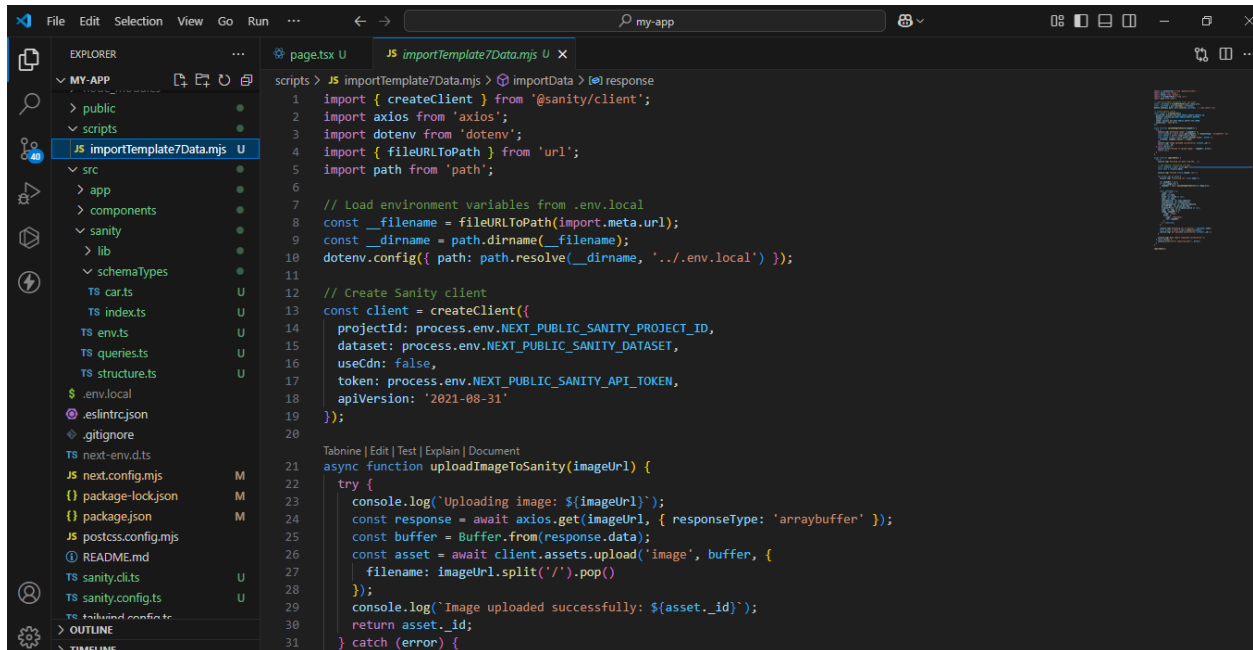
4. Setting Up the Data Import Script

1. Create a new file `scripts/importTemplate7Data.mjs` in your project root.



DAY 3: MARKETPLACE BUILDER HACKATHON 2025

Create a new file `scripts/importTemplate7Data.mjs` in your project root.



5. Install the package

let's install the necessary packages. Run the following command in your terminal:

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

6. Running the Migration Script

- ❖ After creating the `importTemplate7Data.mjs` script, add the following line to your **package.json** file to create a migration script:

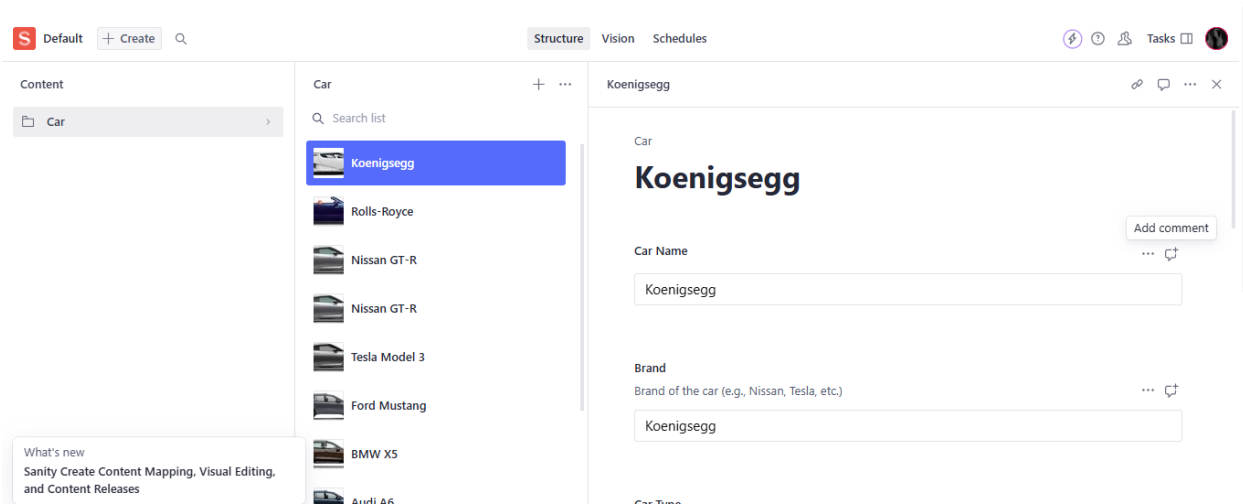
```
"scripts": {  
  "migrate": "node scripts/importTemplate7Data.mjs"  
}
```

- ❖ Alternatively, you can run the migration directly by using this command in your terminal:

```
node scripts/importTemplate7Data.mjs
```

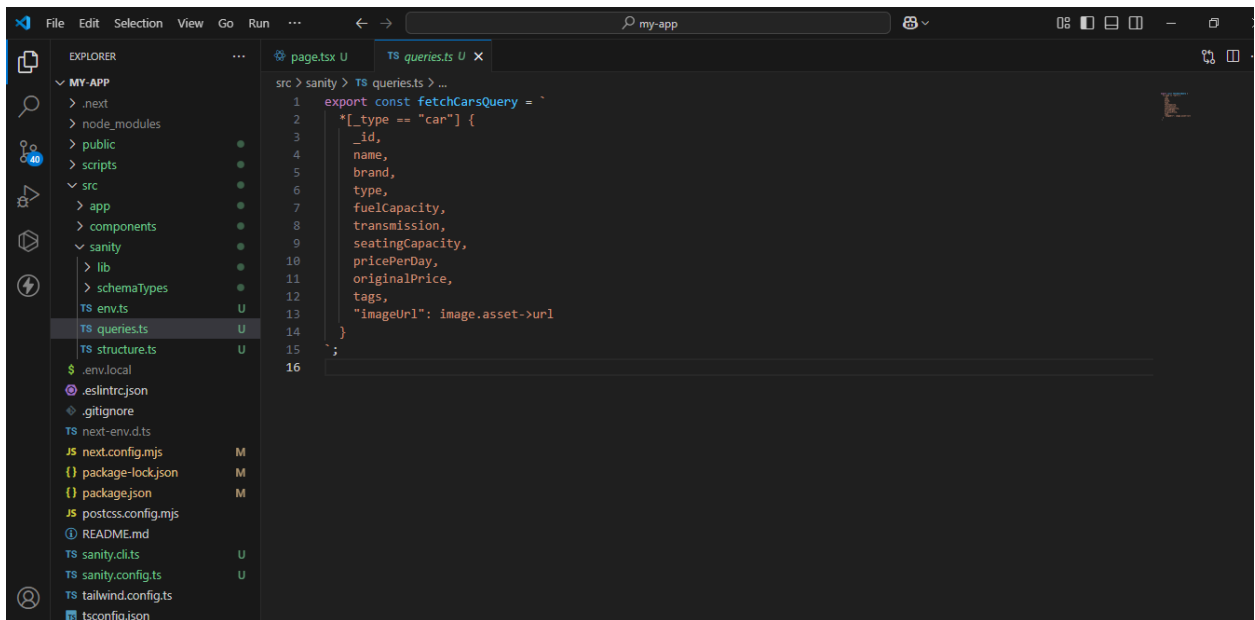
7. Verifying Data Migration in Sanity

Your data has been successfully imported into Sanity! You can check the Sanity Studio to see the newly migrated data.



8. Fetch Data from CMS:

To fetch data, create a queries.ts file in your sanity folder. Here's a sample GROQ query to fetch data:




9. Frontend: Fetch Data


Here's how the fetched data will look in the frontend, displayed based on the query you wrote in queries.ts:

Popular Cars

[View All](#)

Nissan GT-R 


Sport




80LL Manual 4 people

\$96.00 /day [Rent Now](#)

~~\$100.00~~

Rolls-Royce 


Sedan




70LL Manual 4 people

\$96.00 /day [Rent Now](#)

~~\$100.00~~

Nissan GT-R 


Sport




80LL Manual 2 people

\$80.00 /day [Rent Now](#)

~~\$100.00~~

Ford Mustang 

Gasoline




60LL Manual 7 people


\$72.00 /day [Rent Now](#)

~~\$80.00~~

Recommended Cars

Audi A6 


Hybrid




50LL Manual 6 people

\$72.00 /day [Rent Now](#)

~~\$100.00~~

Porsche 911 


Gasoline




60LL Manual 7 people

\$80.00 /day [Rent Now](#)

~~\$100.00~~

Rolls-Royce 


SUV




70LL Manual 5 people

\$80.00 /day [Rent Now](#)

~~\$100.00~~

CR-V 

SUV




80LL Manual 6 people


\$80.00 /day [Rent Now](#)

~~\$100.00~~

Recommended Cars

All New Terios 


SUV




50LL Manual 6 people

\$72.00 /day [Rent Now](#)

~~\$100.00~~

Mercedes-Benz C-Class 


Gasoline




60LL Manual 7 people

\$80.00 /day [Rent Now](#)

~~\$100.00~~

Chevrolet Camaro 


Gasoline




70LL Manual 5 people

\$80.00 /day [Rent Now](#)

~~\$100.00~~

Nissan Altima 

Hybrid



80LL Manual 6 people

\$80.00 /day [Rent Now](#)

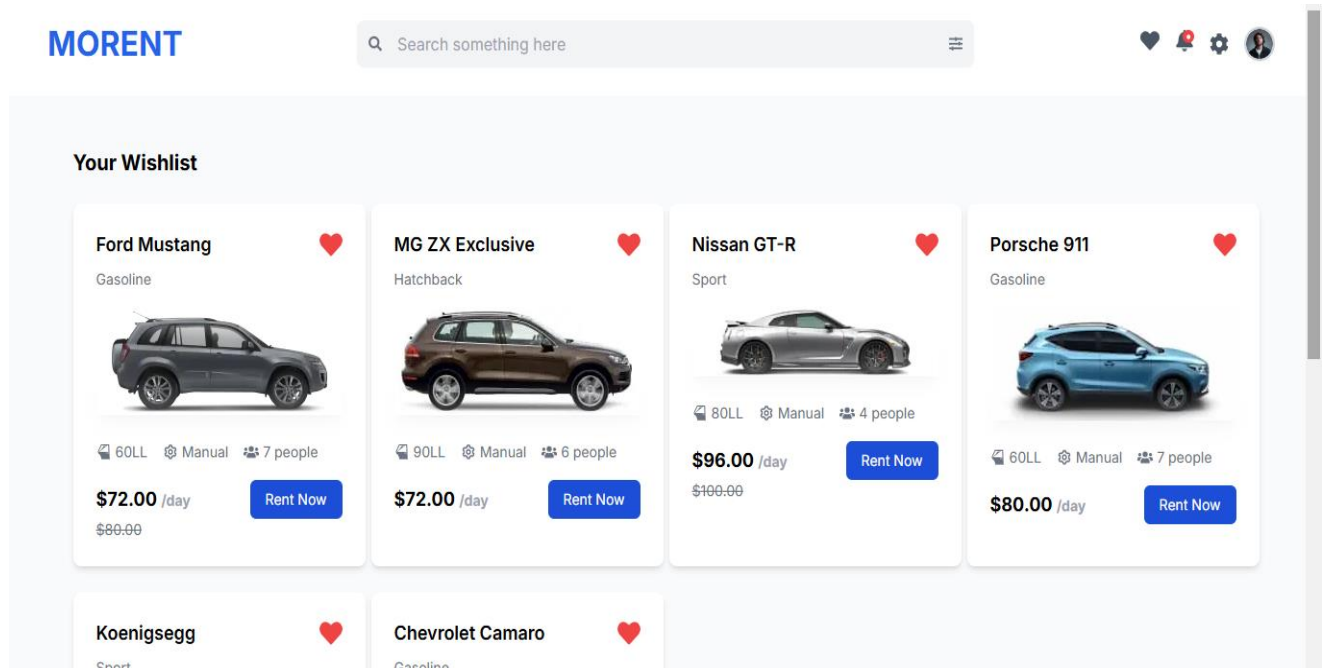
~~\$100.00~~

10. Functionality: Wishlist Feature

Since you have the freedom to add extra functionality, a great addition would be a **Wishlist** feature. This allows users to save their favorite cars for later reference. In the image, you can see how this functionality works. users can click on a "heart" icon to add a car to their wishlist. The wishlist will be saved and accessible across sessions. This feature can significantly enhance user experience by letting them easily revisit cars they're interested in.

Here's a basic outline of how you can implement it:

1. **Wishlist Icon:** Add a heart icon next to each car, which users can click to add the car to their wishlist.
2. **State Management:** Use local storage or a backend service to store the wishlist data, making it persistent.
3. **Display Wishlist:** Create a page where users can view the cars they've added to their wishlist.



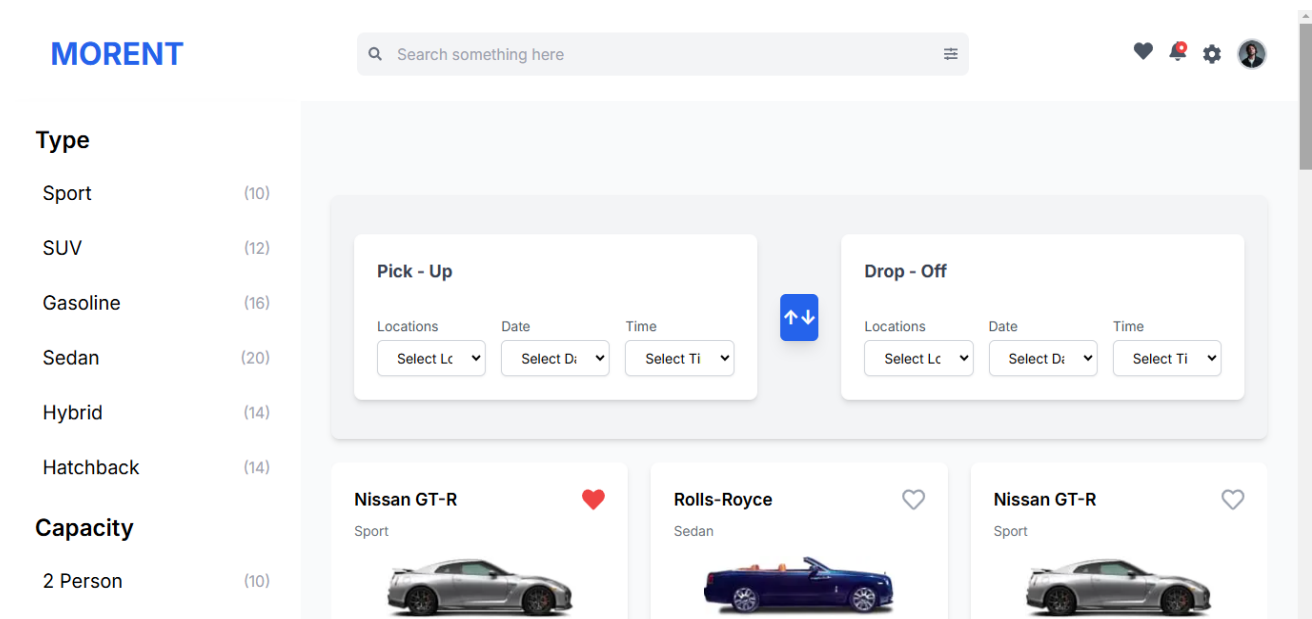
11. Category Page: Dynamic Filtering from Sanity

- The **Category Page** dynamically fetches products from Sanity, allowing users to explore different car categories.
- As shown in the image, the filtering functionality enables users to sort cars based on various attributes such as type, capacity, and price.

❖ Here's a breakdown of how the filtering works:

- **Type Filter:** Filters cars based on their type (e.g., sedan, SUV).
- **Capacity Filter:** Allows filtering cars by their seating capacity.
- **Price Filter:** Filters cars within a specific price range.

This powerful filtering system gives users more control over their search and helps them find the ideal car based on their preferences.



DAY 3: MARKETPLACE BUILDER HACKATHON 2025

Hatchback (14)

Capacity

2 Person (10)

4 Person (14)

5 Person (8)

6 Person (12)

7 or More (16)

Price

Min: \$0 Max: \$100

Nissan GT-R

Sport



\$96 / Day

80L Manual 4 Seats

Rolls-Royce

Sedan



\$96 / Day

70L Manual 4 Seats

Nissan GT-R

Sport



\$80 / Day

80L Manual 4 Seats

Ford Mustang

Gasoline



\$72 / Day

60L Manual 7 Seats

Audi A6

Hybrid



\$72 / Day

50L Manual 6 Seats

Porsche 911

Gasoline



\$80 / Day

60L Manual 7 Seats

Dynamic Filtering from Sanity:

- ❖ As shown in the image, when you apply a filter, such as the price range, the page dynamically updates to display the relevant products.
- ❖ The data is fetched from Sanity, ensuring that the product list reflects your selected filters in real-time.
- ❖ This functionality allows users to easily narrow down their choices based on type, capacity, or price range, enhancing their browsing experience.

Hybrid (14)

Hatchback (14)

Capacity

2 Person (10)

4 Person (14)

5 Person (8)

6 Person (12)

7 or More (16)

Price

Min: \$0 Max: \$72

Ford Mustang

Gasoline



\$72 / Day

60L Manual 7 Seats

Audi A6

Hybrid



\$72 / Day

50L Manual 6 Seats

Chevrolet Camaro

Gasoline



\$72 / Day

70L Manual 5 Seats

Nissan Altima

Hybrid



\$72 / Day

50L Manual 4 Seats

BMW X5

Diesel



\$72 / Day

70L Manual 6 Seats

MG ZX Exclusive

Hatchback



\$72 / Day

90L Manual 6 Seats

Conclusion:

Thank you for reviewing this report. I hope it provides you with clear and helpful instructions on how to work with the API and Sanity, making the most of its features. If you have any further questions or need assistance, feel free to reach out!