

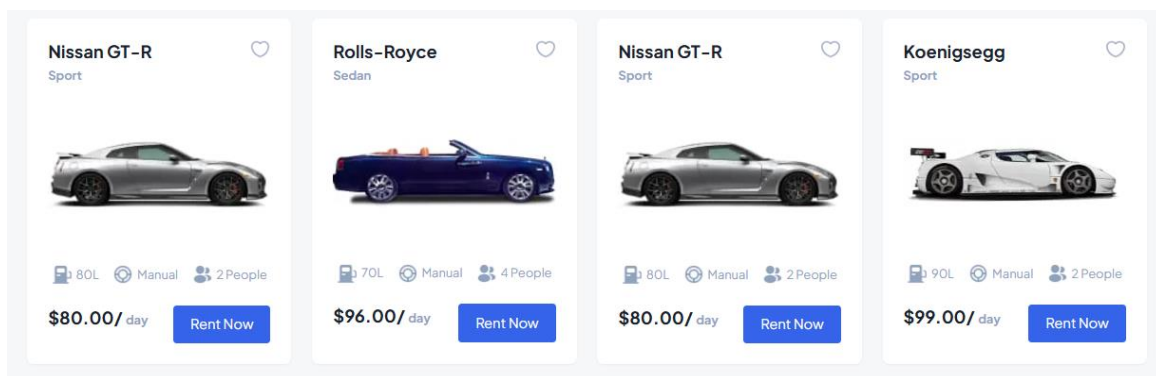
# BUILDING DYNAMIC MARKETPLACE: HACKATHON DAY-4

Prepared by: Fatima Faisal

## 1. Functional Deliverables:

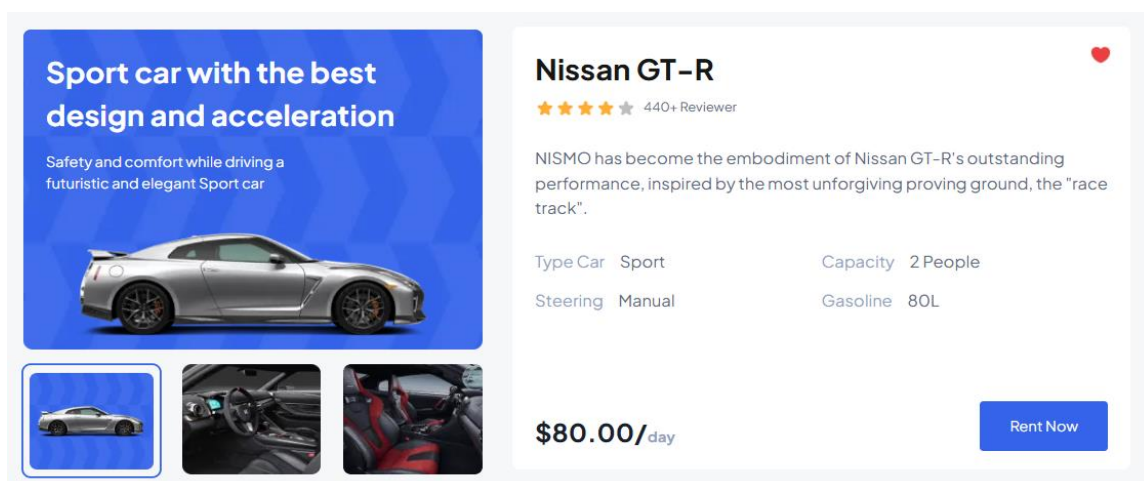
- **The product listing page with dynamic data:**

Grid of rental products displaying dynamic data from Sanity CMS:



- **The product listing page with dynamic data:**

Individual product detail page with accurate routing and data rendering. Containing a **Rent Now** button that proceeds to **Checkout** and Payment form.



- **Working Category Filters & Search bar:**

- **Category Filters:**

TYPE

☒ Sport (10)

☒ SUV (12)

☐ MPV (16)

☐ Sedan (20)

☐ Cope (14)

☐ Hatchback (14)


CAPACITY

☒ 2 Person (10)

localhost:3000/detail-car-rent/4-nissan-gt-r?category=SUV&category=Sport

Nissan GT-R

Sport



80L

Manual


2 People

\$80.00/ day

Rent Now

Nissan GT-R

Sport



80L

Manual


2 People

\$80.00/ day

Rent Now

Koenigsegg

Sport



90L

Manual

2 People

\$99.00/ day


Rent Now

Recommendation Car

View All

All New Terios

SUV



90L

Manual


6 People

\$74.00/ day

Rent Now

CR-V

SUV



80L

Manual


6 People

\$80.00/ day

Rent Now

Rolls-Royce

SUV



70L

Manual

6 People

\$72.00/ day

Rent Now

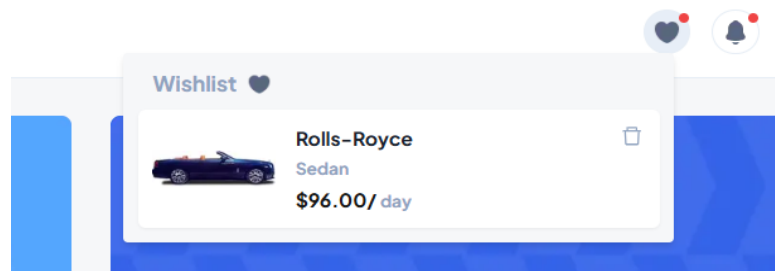
➤ **Search Functionality:**



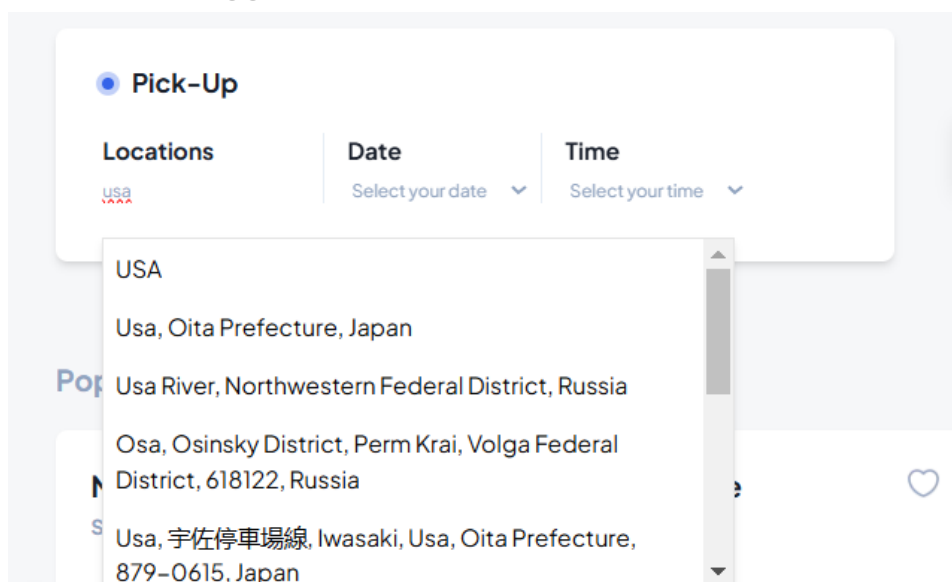
• **Additional Features:**

➤ **Wishlist Functionality:**

Allowing users to save products for future reference, using local storage to persist data.



➤ **Location Suggestion Functionality:**



## 2. Code Deliverables:

- Code Snippets For Car Cards:

```
{cars.map((car: any) => (  
  <div  
    className="bg-white rounded-lg flex flex-col p-6 relative flex-shrink-0 lg:flex-shrink  
    w-[304px] md:w-full lg:w-auto shadow-sm"  
    key={car._id}  
  >  
    <div className="flex items-start justify-between">  
      <div className="flex flex-col text-left space-y-1">  
        <p className="font-bold text-xl text-[#1A202C]">{car.name}</p>  
        <p className="font-bold text-sm text-[#90A3BF]">  
          {car.category}  
        </p>  
      </div>  
      <FavIcon car={car} />  
    </div>  
  
    <Link href={` /detail-car-rent/${car.slug.current}`}>  
      <Image  
        src={car.image.asset.url}  
        width={300}  
        height={100}  
        alt="car-image"  
        className="mt-16 w-64 h-auto object-cover mx-auto"  
      />  
    </Link>  
  )
```

```
</Link>  
  
<div className="flex space-x-4 sm:flex-row mt-16 items-center justify-center -mx-2">  
  <div className="flex space-x-1 items-center justify-center">  
    <Image  
      height={24}  
      width={24}  
      src="/images/gas-station.png"  
      alt="Gas-Station"  
    />  
    <p className="text-[#90A3BF] text-sm font-medium">{car.fuel}</p>  
  </div>  
  <div className="flex space-x-1 items-center justify-center">  
    <Image  
      height={24}  
      width={24}  
      src="/images/manual.png"  
      alt="Mode"  
    />  
    <p className="text-[#90A3BF] text-sm font-medium">{car.mode}</p>  
  </div>  
  <div className="flex space-x-1 items-center justify-center">  
    <Image
```

- **Dynamic Routing:**

```
<Link href={` /detail-car-rent/${car.slug.current}`}>
```

### 3. Technical Report

#### ***1. Steps Taken to Build and Integrate Components:***

- Fetched data from Sanity CMS using queries and integrated it into the components.
- Implemented required functionality based on the fetched data, ensuring that the components correctly utilized the data.
- Created separate components for UI elements and logic, adhering to the separation of concerns for easier maintenance.

#### ***2. Challenges Faced and Solutions Implemented:***

- Encountered errors such as missing closing curly braces '}' in a query, which led to server error. This was identified and fixed by reviewing the query syntax carefully.
- Encountered other errors related to data handling and component rendering, but these were resolved through debugging and checking for any misconfigurations.
- Some server-client mismatches were faced, but they were mitigated by ensuring proper data handling between client and server components and keeping the code separated.

#### ***3. Best Practices Followed During Development:***

- Organized components in a dedicated **app/components** folder to maintain clarity and modularity.
- Kept server-side and client-side code separate to avoid potential errors related to SSR (Server-Side Rendering) and client-side execution.
- Ensured the UI components were responsive, adapting to different screen sizes for a better user experience.