# Hackathon

## Day 4 - Report

Prepared By

ABDUL RAHMAN MOIN

# Day-4

## Dynamic Frontend Components

## Introduction

### Project Overview:

Day 4 focused on designing and building dynamic frontend components for the marketplace using Next.js. The primary goal was to create modular, reusable, and responsive components to display data fetched from Sanity CMS and APIs, ensuring a seamless and professional user experience.

## Key Components Built:

### Product Listing Component:

- Rendered product details dynamically in a grid layout, including car name, rent, image, and seating capacity etc.
- Used reusable Card components for modularity.

### Product Detail Component:

- Implemented dynamic routing to display detailed car information such as description, internal car images, and reviews.
- Integrated a "Related cars" section using category tags.

### Search Bar:

- Enabled real-time filtering of cars by name, car type or tags using local storage.

## Pagination:

- Added a "Show More Cars" button on the car listing page, allowing users to fetch and display more cars from the sanity dynamically alongside the existing ones.

## Category Component:

- Added a sidebar for filtering cars by type/category.
- On larger devices, the sidebar remains open by default, while on smaller devices, it can be toggled using a button.
- Each category is accompanied by a checkbox, allowing users to select one or multiple categories to view relevant cars.

# Steps for Implementation:

## Component Design:

- Created reusable components such as Card, CategoryFilter, and SearchBar.
- Ensured props were used to maintain flexibility across components.

## API Integration:

- Fetched data from Sanity CMS and rendered it dynamically on pages.
- Implemented error handling and fallback states for better user experience.

## Styling:

- Used Tailwind CSS to ensure responsive and modern designs.
- Optimized layouts for both mobile and desktop users.

## Performance Optimization:

- Implemented pagination to handle large datasets efficiently.

## Challenges and Solutions:

**Challenge:** Search functionality with dynamic page refresh.

- **Solution:** Initially faced an issue where searching on the /search page displayed old results until the page was refreshed. Solved it by programmatically implementing a hard reload using the following code:

```
if (pathname === "/search") {
    setTimeout(() => {
      window.location.reload();
    }, 200);
}
```

**Challenge:** Ensuring smooth pagination.

- **Solution:** Managed state effectively to dynamically load data on page navigation.

## Outcome:

The final result is a fully functional and responsive marketplace frontend, featuring dynamic components for product listing, details, search, filtering, and more. These components are scalable and modular, adhering to industry best practices.

## Submission Details:

### Screen Recordings:

- **Car Listing and Pagination Link:**

Screen Recording Link

- **Product Detail Page and Related Cars:**

  Screen Recording Link

- **Categorization Of Cars:**

  Screen Recording Link

- **Searching Functionality:**

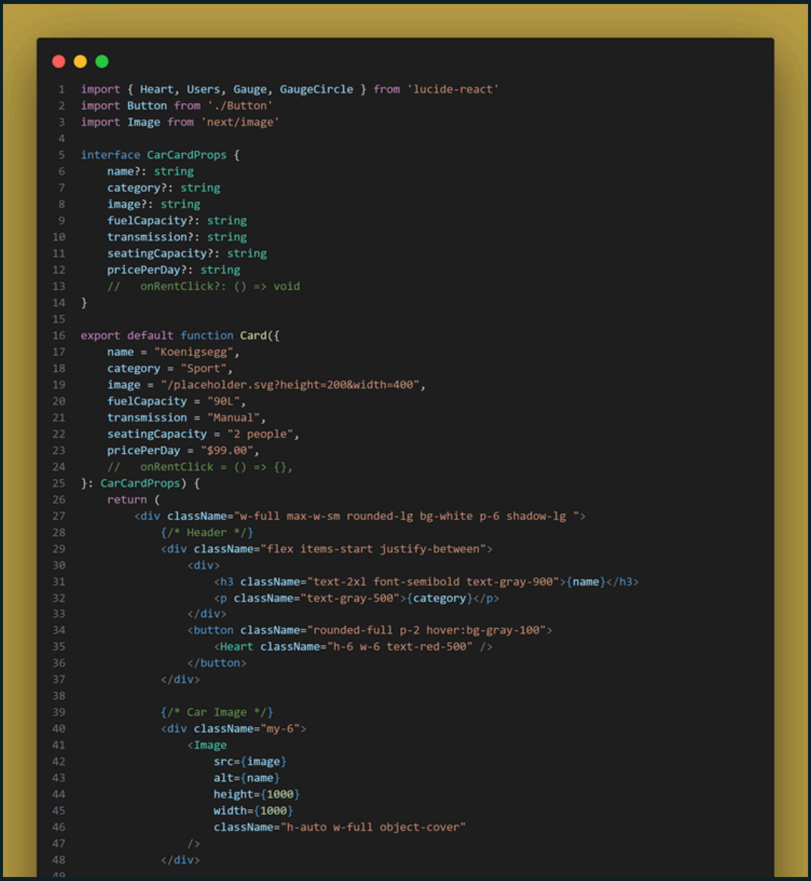
  Screen Recording Link

- **User Profile:**

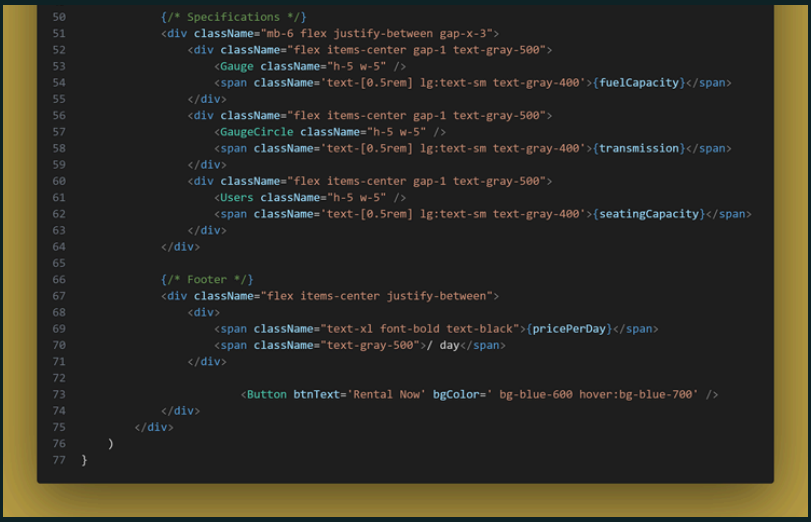
  Screen Recording Link

## Code Snippets:

### Card Component:

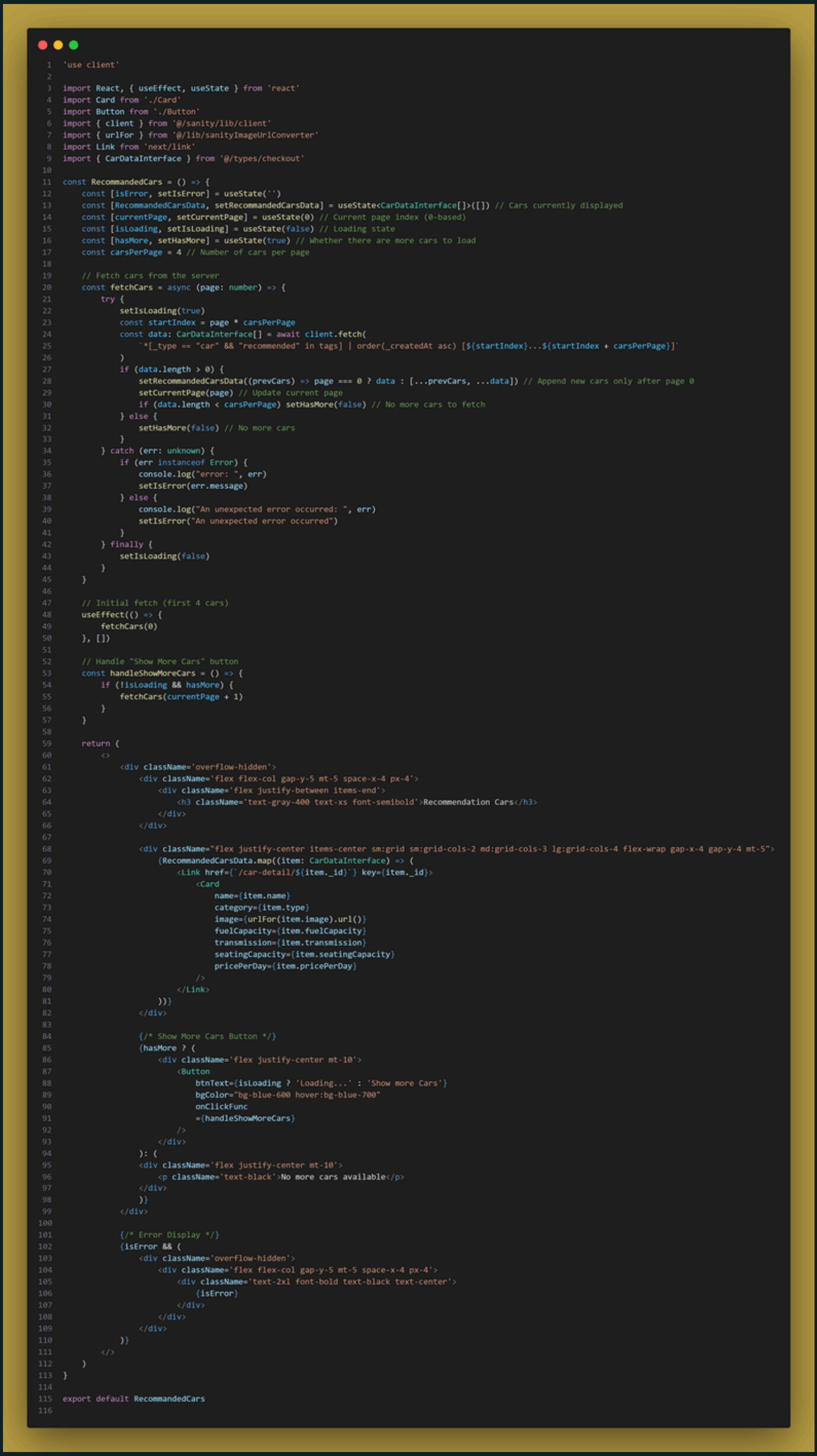


```
import { Heart, Users, Gauge, GaugeCircle } from 'lucide-react'
import Button from './Button'
import Image from 'next/image'

interface CarCardProps {
    name?: string
    category?: string
    image?: string
    fuelCapacity?: string
    transmission?: string
    seatingCapacity?: string
    pricePerDay?: string
    //   onRentClick?: () => void
}

export default function Card({
    name = "Koenigsegg",
    category = "Sport",
    image = "/placeholder.svg?height=200&width=400",
    fuelCapacity = "90L",
    transmission = "Manual",
    seatingCapacity = "2 people",
    pricePerDay = "$99.00",
    //   onRentClick = () => {},
}: CarCardProps) {
    return (
        <div className="w-full max-w-sm rounded-lg bg-white p-6 shadow-lg ">
            {/* Header */}
            <div className="flex items-start justify-between">
                <div>
                    <h3 className="text-2xl font-semibold text-gray-900">{name}</h3>
                    <p className="text-gray-500">{category}</p>
                </div>
                <button className="rounded-full p-2 hover:bg-gray-100">
                    <Heart className="h-6 w-6 text-red-500" />
                </button>
            </div>

            {/* Car Image */}
            <div className="my-6">
                <Image
                    src={image}
                    alt={name}
                    height={1000}
                    width={1000}
                    className="h-auto w-full object-cover"
                />
            </div>
```



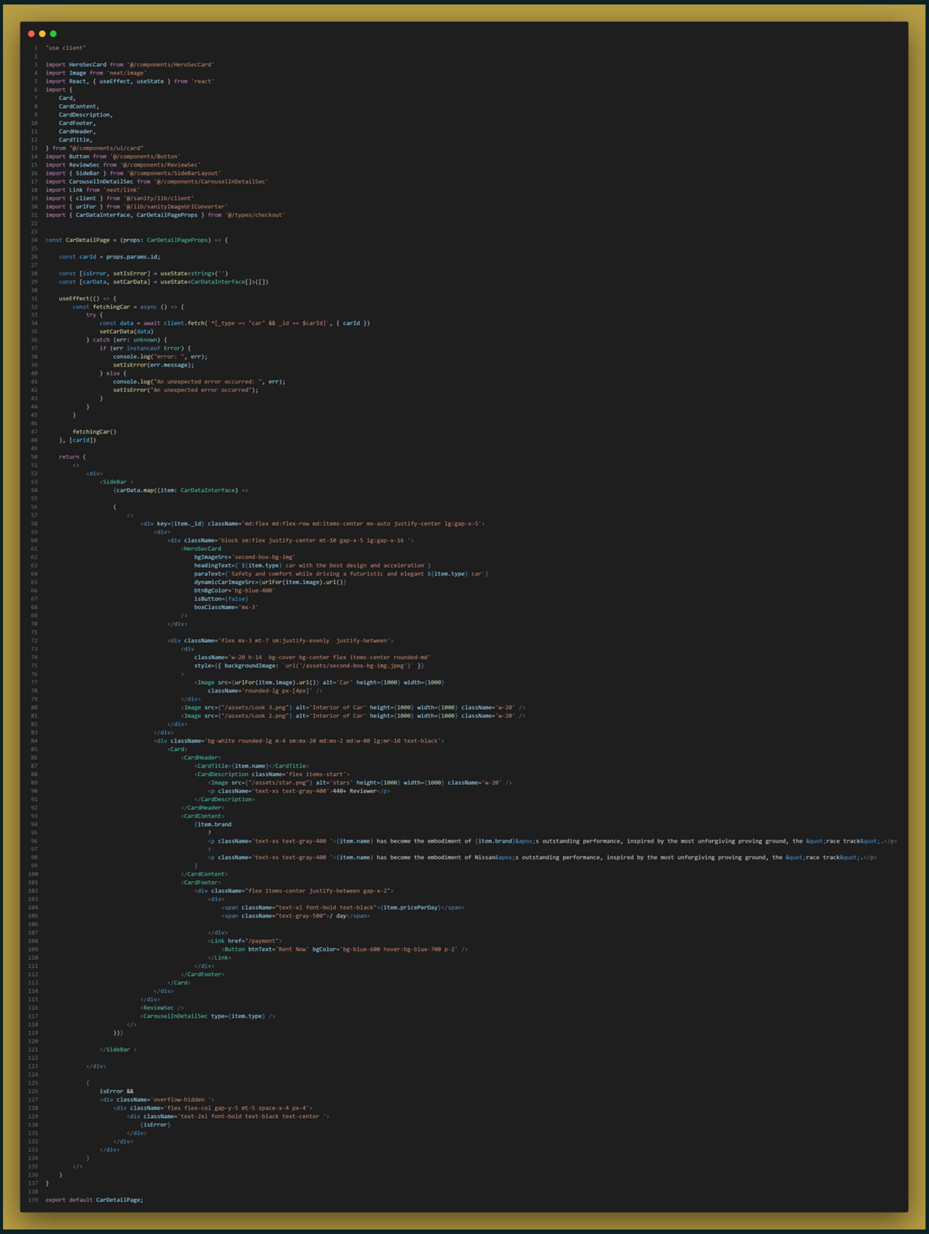
```jsx
            {/* Specifications */}
            <div className="mb-6 flex justify-between gap-x-3">
                <div className="flex items-center gap-1 text-gray-500">
                    <Gauge className="h-5 w-5" />
                    <span className='text-[0.5rem] lg:text-sm text-gray-400'>{fuelCapacity}</span>
                </div>
                <div className="flex items-center gap-1 text-gray-500">
                    <GaugeCircle className="h-5 w-5" />
                    <span className='text-[0.5rem] lg:text-sm text-gray-400'>{transmission}</span>
                </div>
                <div className="flex items-center gap-1 text-gray-500">
                    <Users className="h-5 w-5" />
                    <span className='text-[0.5rem] lg:text-sm text-gray-400'>{seatingCapacity}</span>
                </div>
            </div>

            {/* Footer */}
            <div className="flex items-center justify-between">
                <div>
                    <span className="text-xl font-bold text-black">{pricePerDay}</span>
                    <span className="text-gray-500">/ day</span>
                </div>

                        <Button btnText='Rental Now' bgColor=' bg-blue-600 hover:bg-blue-700' />
            </div>
        </div>
    )
}
```

## Car Listing Component:



```tsx
'use client'

import React, { useEffect, useState } from 'react'
import Card from './Card'
import Button from './Button'
import { client } from '@/sanity/lib/client'
import { urlFor } from '@/lib/sanityImageUrlConverter'
import Link from 'next/link'
import { CarDataInterface } from '@/types/checkout'

const RecommandedCars = () => {
    const [isError, setIsError] = useState('')
    const [RecommandedCarsData, setRecommandedCarsData] = useState<CarDataInterface[]>([]) // Cars currently displayed
    const [currentPage, setCurrentPage] = useState(0) // Current page index (0-based)
    const [isLoading, setIsLoading] = useState(false) // Loading state
    const [hasMore, setHasMore] = useState(true) // Whether there are more cars to load
    const carsPerPage = 4 // Number of cars per page

    // Fetch cars from the server
    const fetchCars = async (page: number) => {
        try {
            setIsLoading(true)
            const startIndex = page * carsPerPage
            const data: CarDataInterface[] = await client.fetch(
                `*[_type == "car" && "recommended" in tags] | order(_createdAt asc) [${startIndex}...${startIndex + carsPerPage}]`
            )
            if (data.length > 0) {
                setRecommandedCarsData((prevCars) => page === 0 ? data : [...prevCars, ...data]) // Append new cars only after page 0
                setCurrentPage(page) // Update current page
                if (data.length < carsPerPage) setHasMore(false) // No more cars to fetch
            } else {
                setHasMore(false) // No more cars
            }
        } catch (err: unknown) {
            if (err instanceof Error) {
                console.log("error: ", err)
                setIsError(err.message)
            } else {
                console.log("An unexpected error occurred: ", err)
                setIsError("An unexpected error occurred")
            }
        } finally {
            setIsLoading(false)
        }
    }

    // Initial fetch (first 4 cars)
    useEffect(() => {
        fetchCars(0)
    }, [])

    // Handle "Show More Cars" button
    const handleShowMoreCars = () => {
        if (!isLoading && hasMore) {
            fetchCars(currentPage + 1)
        }
    }

    return (
        <>
            <div className='overflow-hidden'>
                <div className='flex flex-col gap-y-5 mt-5 space-x-4 px-4'>
                    <div className='flex justify-between items-end'>
                        <h3 className='text-gray-400 text-xs font-semibold'>Recommendation Cars</h3>
                    </div>
                </div>

                <div className="flex justify-center items-center sm:grid sm:grid-cols-2 md:grid-cols-3 lg:grid-cols-4 flex-wrap gap-x-4 gap-y-4 mt-5">
                    {RecommandedCarsData.map((item: CarDataInterface) => (
                        <Link href={`/car-detail/${item._id}`} key={item._id}>
                            <Card
                                name={item.name}
                                category={item.type}
                                image={urlFor(item.image).url()}
                                fuelCapacity={item.fuelCapacity}
                                transmission={item.transmission}
                                seatingCapacity={item.seatingCapacity}
                                pricePerDay={item.pricePerDay}
                            />
                        </Link>
                    ))}
                </div>

                {/* Show More Cars Button */}
                {hasMore ? (
                    <div className='flex justify-center mt-10'>
                        <Button
                            btnText={isLoading ? 'Loading...' : 'Show more Cars'}
                            bgColor="bg-blue-600 hover:bg-blue-700"
                            onClickFunc
                            ={handleShowMoreCars}
                        />
                    </div>
                ): (
                <div className='flex justify-center mt-10'>
                    <p className='text-black'>No more cars available</p>
                </div>
                )}
            </div>

            {/* Error Display */}
            {isError && (
                <div className='overflow-hidden'>
                    <div className='flex flex-col gap-y-5 mt-5 space-x-4 px-4'>
                        <div className='text-2xl font-bold text-black text-center'>
                            {isError}
                        </div>
                    </div>
                </div>
            )}
        </>
    )
}

export default RecommandedCars

```

# Dynamic Routing Code:



```tsx
"use client"

import HeroSecCard from '@/components/HeroSecCard'
import Image from 'next/image'
import React, { useEffect, useState } from 'react'
import {
    Card,
    CardContent,
    CardDescription,
    CardFooter,
    CardHeader,
    CardTitle,
} from "@/components/ui/card"
import Button from '@/components/Button'
import ReviewSec from '@/components/ReviewSec'
import { SideBar } from '@/components/SideBarLayout'
import CarouselInDetailSec from '@/components/CarouselInDetailSec'
import Link from 'next/link'
import { client } from '@/sanity/lib/client'
import { urlFor } from '@/lib/sanityImageUrlConverter'
import { CarDataInterface, CarDetailPageProps } from '@/types/checkout'


const CarDetailPage = (props: CarDetailPageProps) => {

    const carId = props.params.id;

    const [isError, setIsError] = useState<string>('')
    const [carData, setCarData] = useState<CarDataInterface[]>([])

    useEffect(() => {
        const fetchingCar = async () => {
            try {
                const data = await client.fetch(`*[_type == "car" && _id == $carId]`, { carId })
                setCarData(data)
            } catch (err: unknown) {
                if (err instanceof Error) {
                    console.log("error: ", err);
                    setIsError(err.message);
                } else {
                    console.log("An unexpected error occurred: ", err);
                    setIsError("An unexpected error occurred");
                }
            }
        }

        fetchingCar()
    }, [carId])

    return (
        <>
            <div>
                <SideBar >
                    {carData.map((item: CarDataInterface) =>

                    (
                        <>
                            <div key={item._id} className='md:flex md:flex-row md:items-center mx-auto justify-center lg:gap-x-5'>
                                <div>
                                    <div className='block sm:flex justify-center mt-10 gap-x-5 lg:gap-x-16 '>
                                        <HeroSecCard
                                            bgImageSrc='second-box-bg-img'
                                            headingText={`${item.type} car with the best design and acceleration`}
                                            paraText={`Safety and comfort while driving a futuristic and elegant ${item.type} car`}
                                            dynamicCarImageSrc={urlFor(item.image).url()}
                                            btnBgColor='bg-blue-400'
                                            isButton={false}
                                            boxClassName='mx-3'
                                        />
                                    </div>

                                    <div className='flex mx-3 mt-7 sm:justify-evenly  justify-between'>
                                        <div
                                            className='w-20 h-14  bg-cover bg-center flex items-center rounded-md'
                                            style={{ backgroundImage: `url('/assets/second-box-bg-img.jpeg')` }}
                                        >
                                            <Image src={urlFor(item.image).url()} alt='Car' height={1000} width={1000}
                                                className='rounded-lg px-[4px]' />
                                        </div>
                                        <Image src={"/assets/Look 3.png"} alt='Interior of Car' height={1000} width={1000} className='w-20' />
                                        <Image src={"/assets/Look 2.png"} alt='Interior of Car' height={1000} width={1000} className='w-20' />
                                    </div>
                                </div>
                                <div className='bg-white rounded-lg m-4 sm:mx-20 md:mx-2 md:w-80 lg:mr-10 text-black'>
                                    <Card>
                                        <CardHeader>
                                            <CardTitle>{item.name}</CardTitle>
                                            <CardDescription className='flex items-start'>
                                                <Image src={"/assets/star.png"} alt='stars' height={1000} width={1000} className='w-20' />
                                                <p className='text-xs text-gray-400'>440+ Reviewer</p>
                                            </CardDescription>
                                        </CardHeader>
                                        <CardContent>
                                            {item.brand
                                                ?
                                                <p className='text-xs text-gray-400 '>{item.name} has become the embodiment of {item.brand}&apos;s outstanding performance, inspired by the most unforgiving proving ground, the &quot;race track&quot;.</p>
                                                :
                                                <p className='text-xs text-gray-400 '>{item.name} has become the embodiment of Nissan&apos;s outstanding performance, inspired by the most unforgiving proving ground, the &quot;race track&quot;.</p>
                                            }
                                        </CardContent>
                                        <CardFooter>
                                            <div className="flex items-center justify-between gap-x-2">
                                                <div>
                                                    <span className="text-xl font-bold text-black">{item.pricePerDay}</span>
                                                    <span className="text-gray-500">/ day</span>

                                                </div>
                                                <Link href="/payment">
                                                    <Button btnText='Rent Now' bgColor='bg-blue-600 hover:bg-blue-700 p-2' />
                                                </Link>
                                            </div>
                                        </CardFooter>
                                    </Card>
                                </div>
                            </div>
                            <ReviewSec />
                            <CarouselInDetailSec type={item.type} />
                        </>
                    ))}

                </SideBar >

            </div>

            {
                isError &&
                <div className='overflow-hidden '>
                    <div className='flex flex-col gap-y-5 mt-5 space-x-4 px-4'>
                        <div className='text-2xl font-bold text-black text-center '>
                            {isError}
                        </div>
                    </div>
                </div>
            }
        </>
    )
}

export default CarDetailPage;
```