Day 4 Hackathon: Car Rental Marketplace

Objective:

Enhance the Car Rental Marketplace application with dynamic routing for car detail pages, a review component for user feedback, and authentication functionality using NextAuth and Redux.

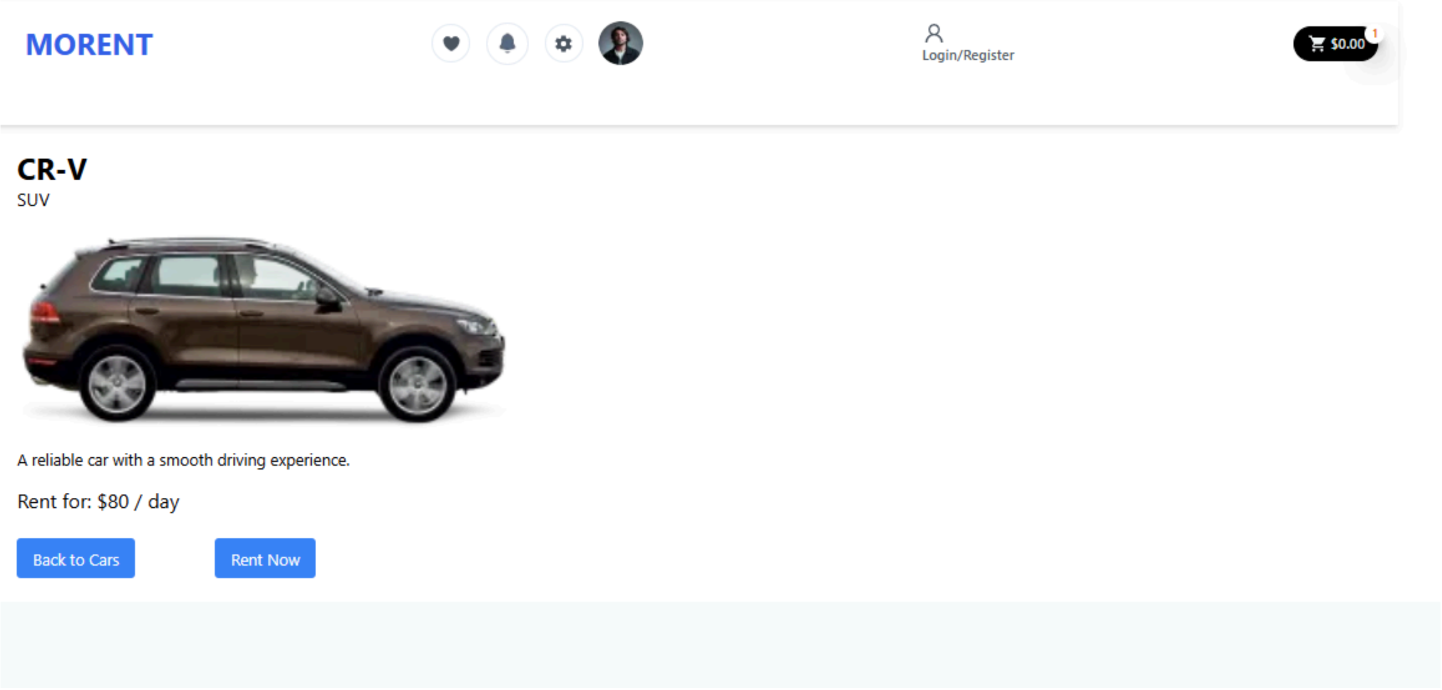
Features Implemented:

1. Dynamic Routing for Car Detail Pages

- **Overview**: Implemented dynamic routing to display detailed information for each car.
- **Approach**:
  - Utilized Next.js's file-based routing system.
  - Created a dynamic route file: [carId].js.
  - Fetched car details dynamically using getServerSideProps or getStaticProps for server-side rendering.
  - Integrated a responsive layout to display car specifications, pricing, and availability.
- **Outcome**: Users can now navigate to specific car detail pages by clicking on car listings.

High Fidelity Dashboard - Home Rent

Last Updated: **8 Aug 2022**



MORENT

Login/Register

$0.00 1

CR-V

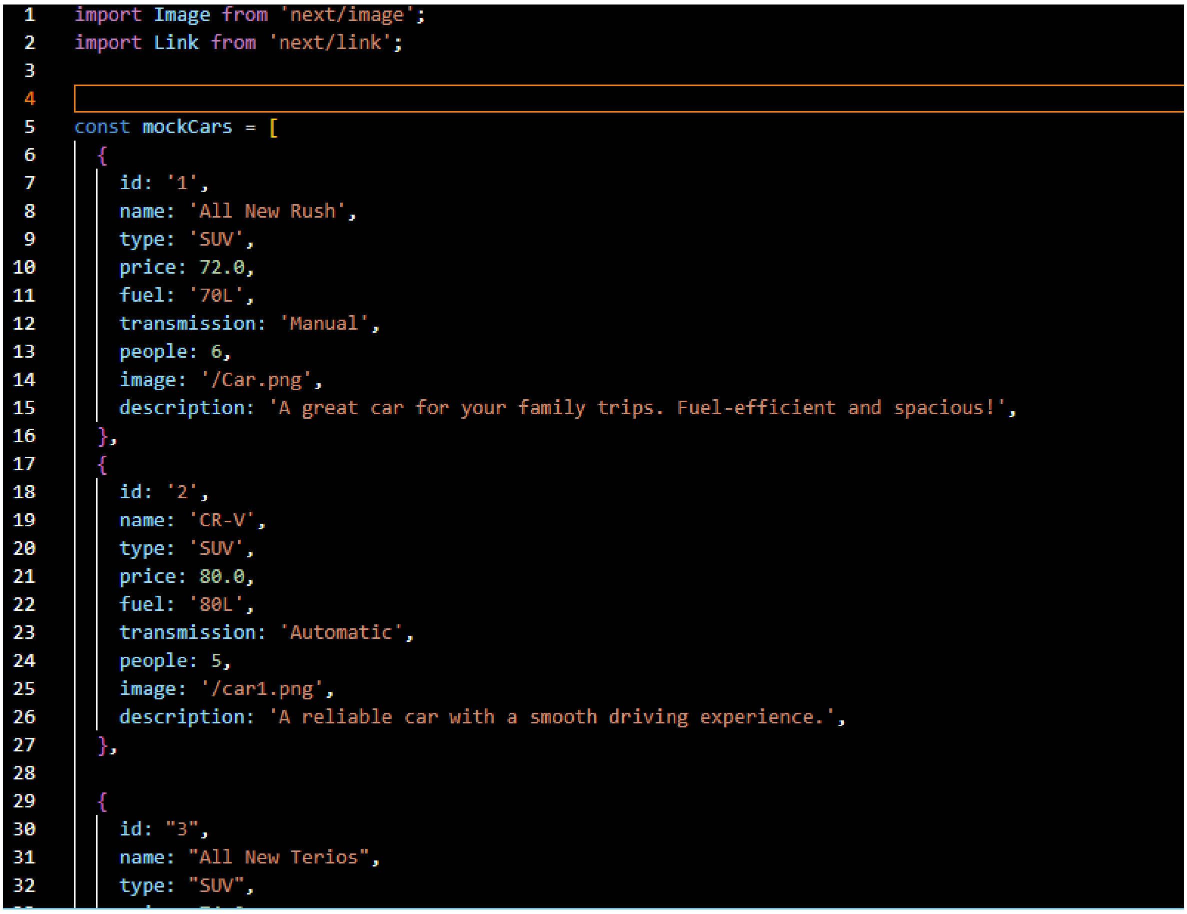
SUV

A reliable car with a smooth driving experience.

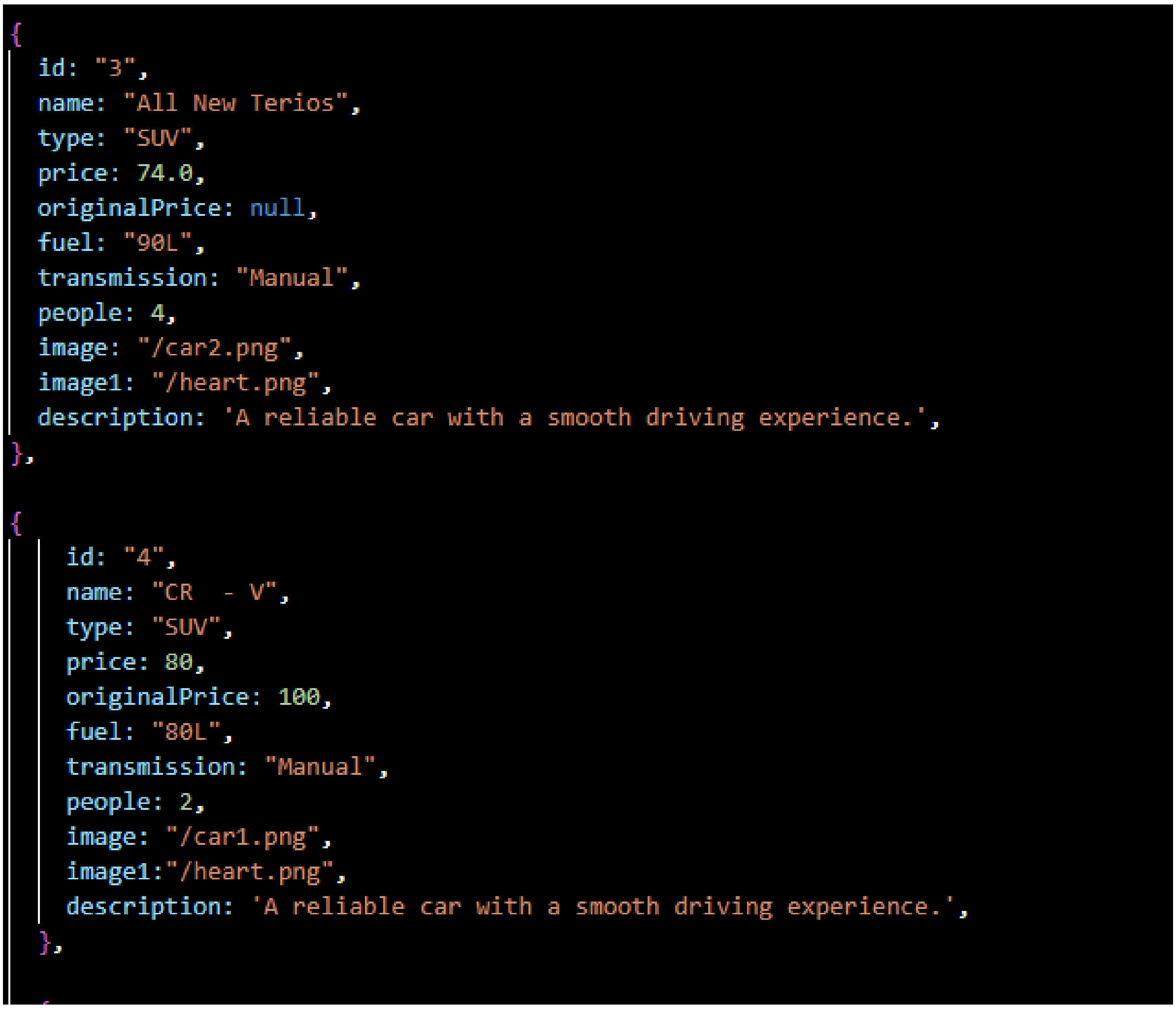
Rent for: $80 / day

Back to Cars

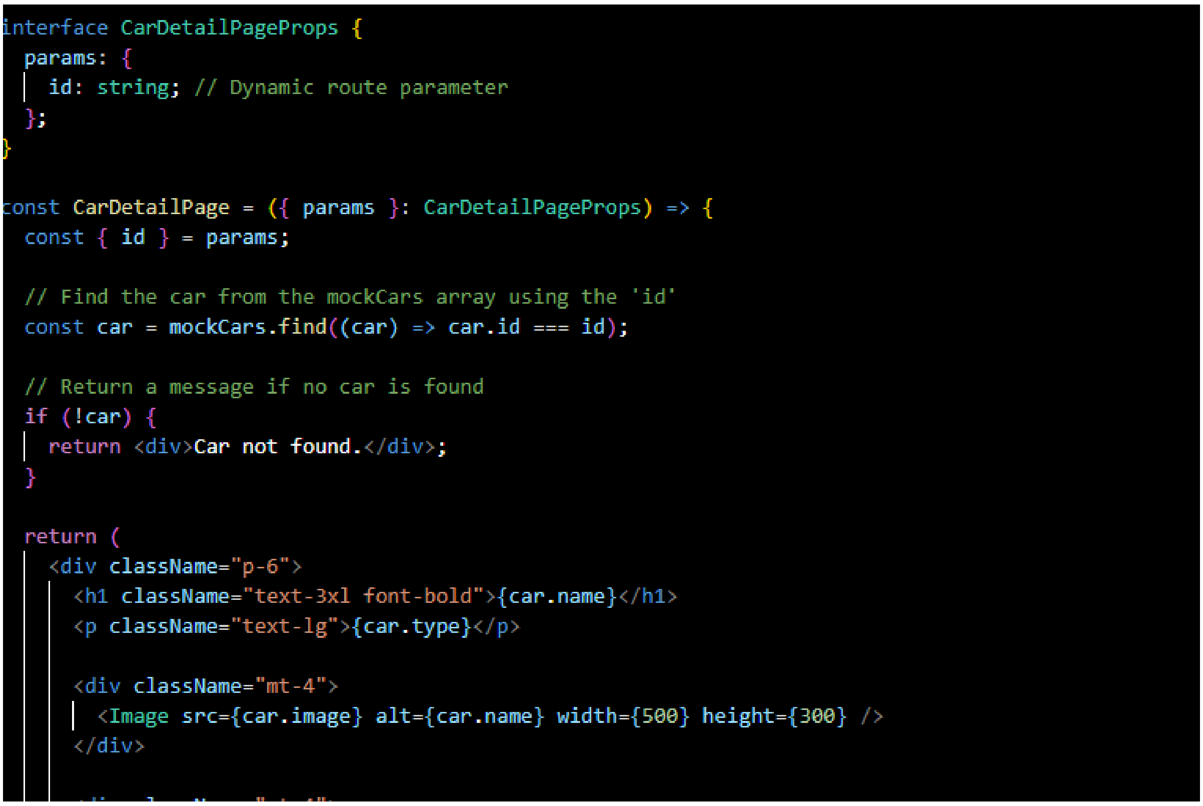
Rent Now



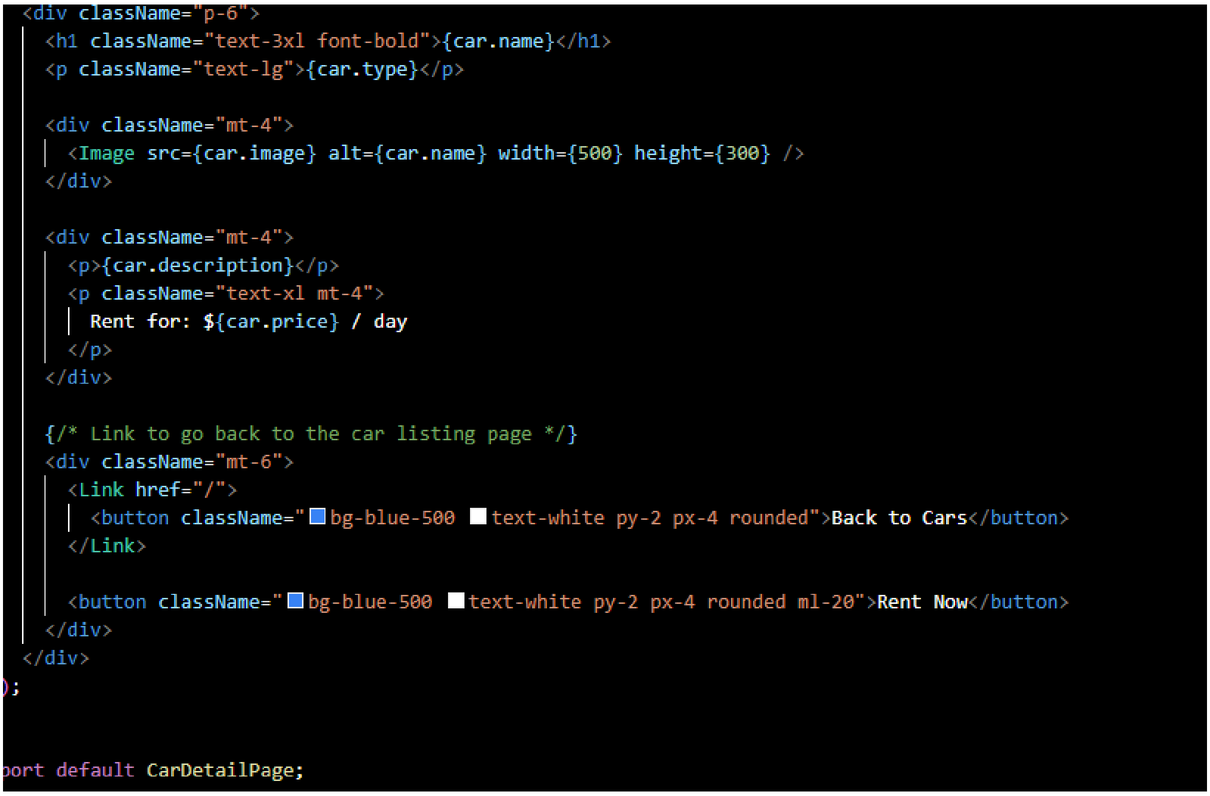
```
import Image from 'next/image';
import Link from 'next/link';


const mockCars = [
  {
    id: '1',
    name: 'All New Rush',
    type: 'SUV',
    price: 72.0,
    fuel: '70L',
    transmission: 'Manual',
    people: 6,
    image: '/Car.png',
    description: 'A great car for your family trips. Fuel-efficient and spacious!',
  },
  {
    id: '2',
    name: 'CR-V',
    type: 'SUV',
    price: 80.0,
    fuel: '80L',
    transmission: 'Automatic',
    people: 5,
    image: '/car1.png',
    description: 'A reliable car with a smooth driving experience.',
  },

  {
    id: "3",
    name: "All New Terios",
    type: "SUV",
```



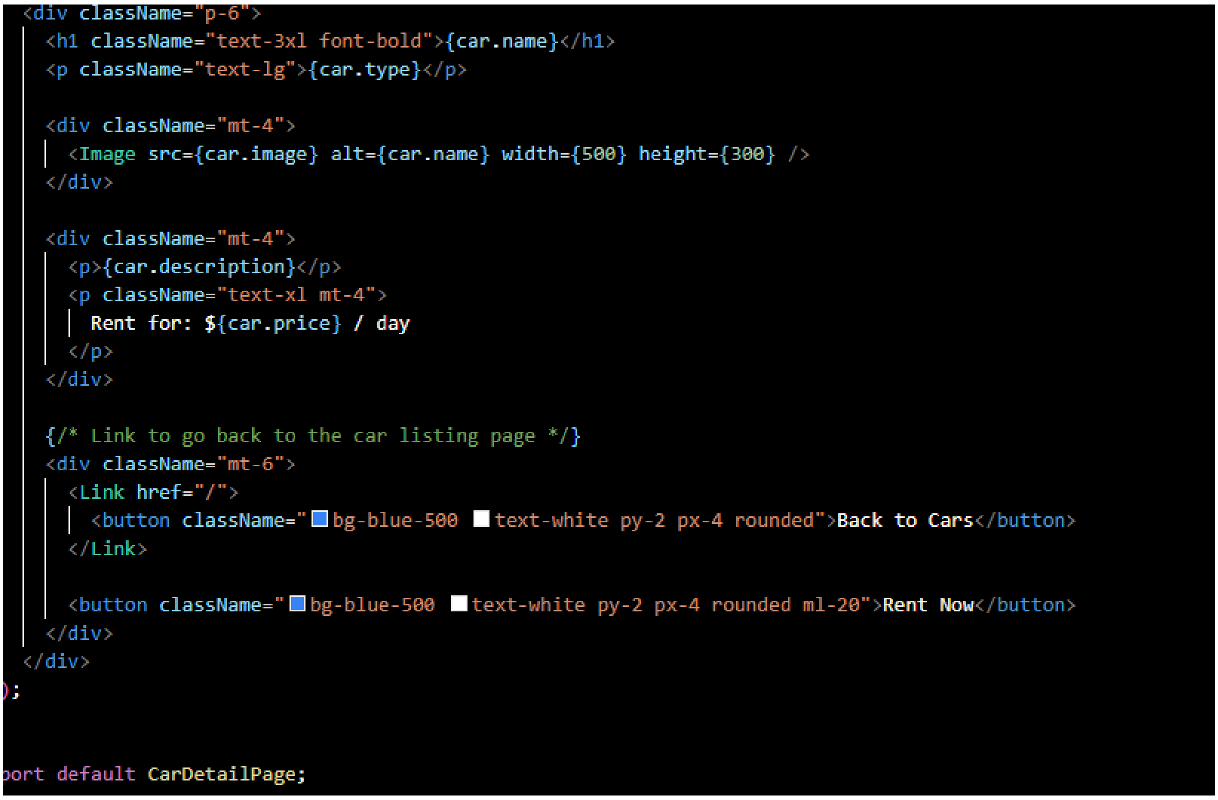
```
{
  id: "3",
  name: "All New Terios",
  type: "SUV",
  price: 74.0,
  originalPrice: null,
  fuel: "90L",
  transmission: "Manual",
  people: 4,
  image: "/car2.png",
  image1: "/heart.png",
  description: 'A reliable car with a smooth driving experience.',
},

{
    id: "4",
    name: "CR  - V",
    type: "SUV",
    price: 80,
    originalPrice: 100,
    fuel: "80L",
    transmission: "Manual",
    people: 2,
    image: "/car1.png",
    image1:"/heart.png",
    description: 'A reliable car with a smooth driving experience.',
  },
```



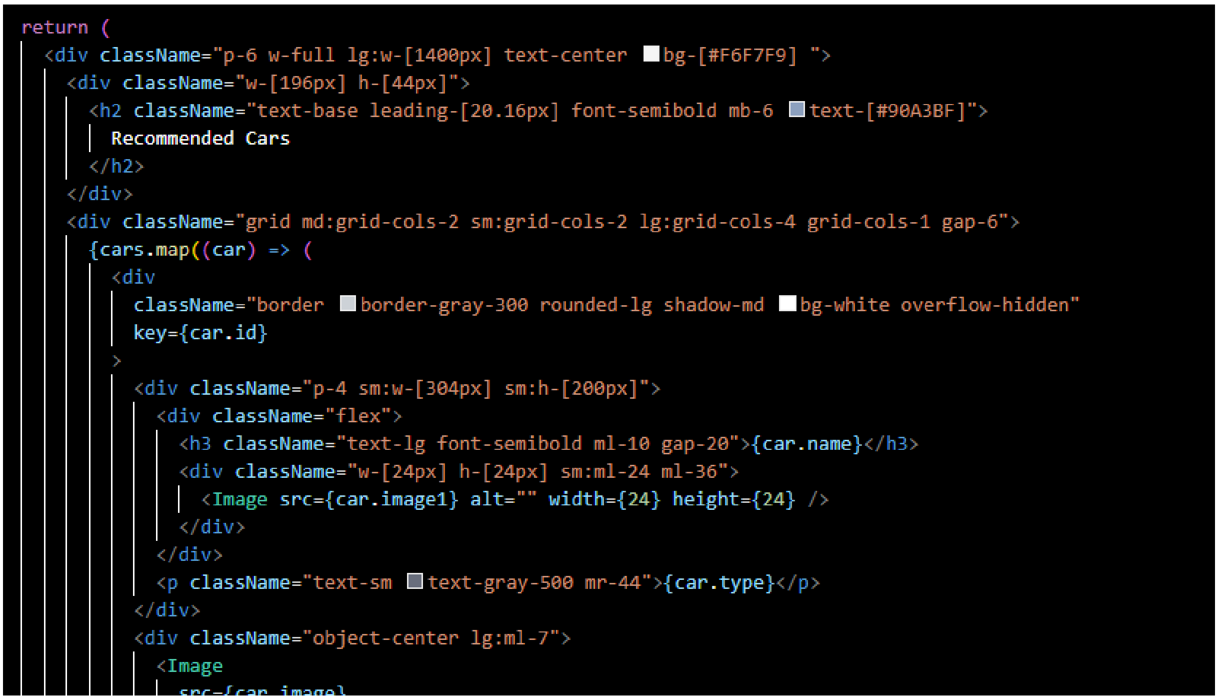
```
interface CarDetailPageProps {
  params: {
    id: string; // Dynamic route parameter
  };
}

const CarDetailPage = ({ params }: CarDetailPageProps) => {
  const { id } = params;

  // Find the car from the mockCars array using the 'id'
  const car = mockCars.find((car) => car.id === id);

  // Return a message if no car is found
  if (!car) {
    return <div>Car not found.</div>;
  }

  return (
    <div className="p-6">
      <h1 className="text-3xl font-bold">{car.name}</h1>
      <p className="text-lg">{car.type}</p>

      <div className="mt-4">
        <Image src={car.image} alt={car.name} width={500} height={300} />
      </div>
```



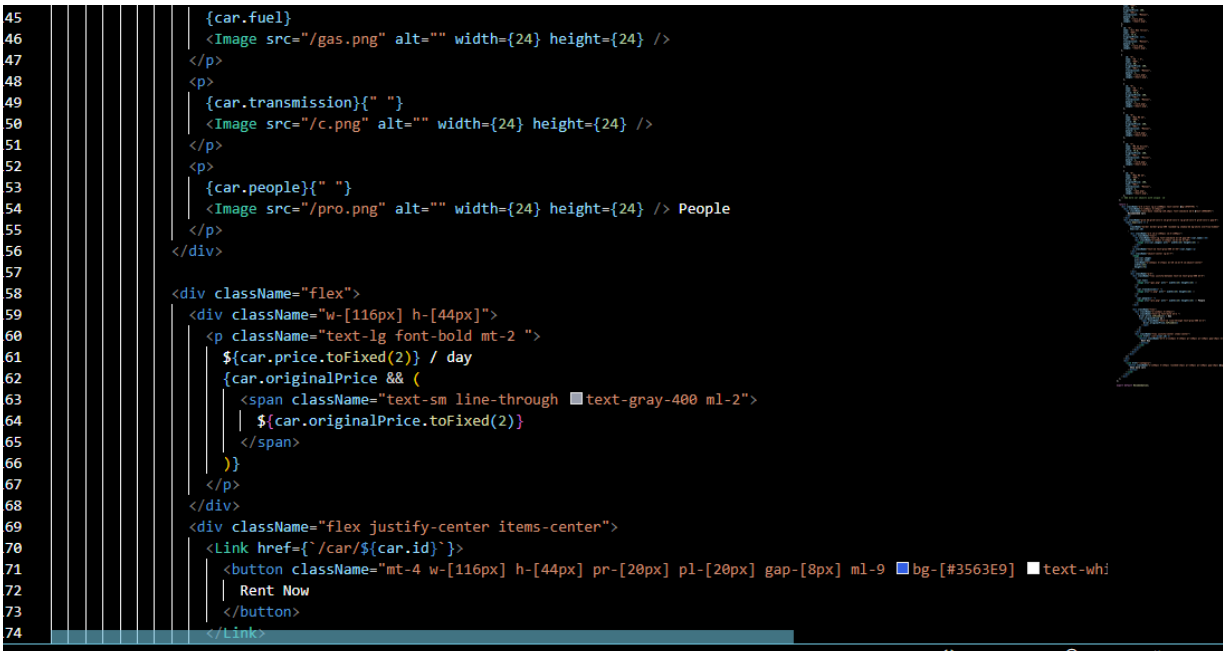
```
<div className="p-6">
  <h1 className="text-3xl font-bold">{car.name}</h1>
  <p className="text-lg">{car.type}</p>

  <div className="mt-4">
    <Image src={car.image} alt={car.name} width={500} height={300} />
  </div>

  <div className="mt-4">
    <p>{car.description}</p>
    <p className="text-xl mt-4">
      Rent for: ${car.price} / day
    </p>
  </div>

  {/* Link to go back to the car listing page */}
  <div className="mt-6">
    <Link href="/">
      <button className="bg-blue-500 text-white py-2 px-4 rounded">Back to Cars</button>
    </Link>

    <button className="bg-blue-500 text-white py-2 px-4 rounded ml-20">Rent Now</button>
  </div>
</div>
);


port default CarDetailPage;
```



```
<div className="p-6">
  <h1 className="text-3xl font-bold">{car.name}</h1>
  <p className="text-lg">{car.type}</p>

  <div className="mt-4">
    <Image src={car.image} alt={car.name} width={500} height={300} />
  </div>

  <div className="mt-4">
    <p>{car.description}</p>
    <p className="text-xl mt-4">
      Rent for: ${car.price} / day
    </p>
  </div>

  {/* Link to go back to the car listing page */}
  <div className="mt-6">
    <Link href="/">
      <button className="bg-blue-500 text-white py-2 px-4 rounded">Back to Cars</button>
    </Link>

    <button className="bg-blue-500 text-white py-2 px-4 rounded ml-20">Rent Now</button>
  </div>
</div>
);


port default CarDetailPage;
```



```
return (
  <div className="p-6 w-full lg:w-[1400px] text-center bg-[#F6F7F9] ">
    <div className="w-[196px] h-[44px]">
      <h2 className="text-base leading-[20.16px] font-semibold mb-6 text-[#90A3BF]">
        Recommended Cars
      </h2>
    </div>
    <div className="grid md:grid-cols-2 sm:grid-cols-2 lg:grid-cols-4 grid-cols-1 gap-6">
      {cars.map((car) => (
        <div
          className="border border-gray-300 rounded-lg shadow-md bg-white overflow-hidden"
          key={car.id}
        >
          <div className="p-4 sm:w-[304px] sm:h-[200px]">
            <div className="flex">
              <h3 className="text-lg font-semibold ml-10 gap-20">{car.name}</h3>
              <div className="w-[24px] h-[24px] sm:ml-24 ml-36">
                <Image src={car.image1} alt="" width={24} height={24} />
              </div>
            </div>
            <p className="text-sm text-gray-500 mr-44">{car.type}</p>
          </div>
          <div className="object-center lg:ml-7">
            <Image
              src={car.image}
```



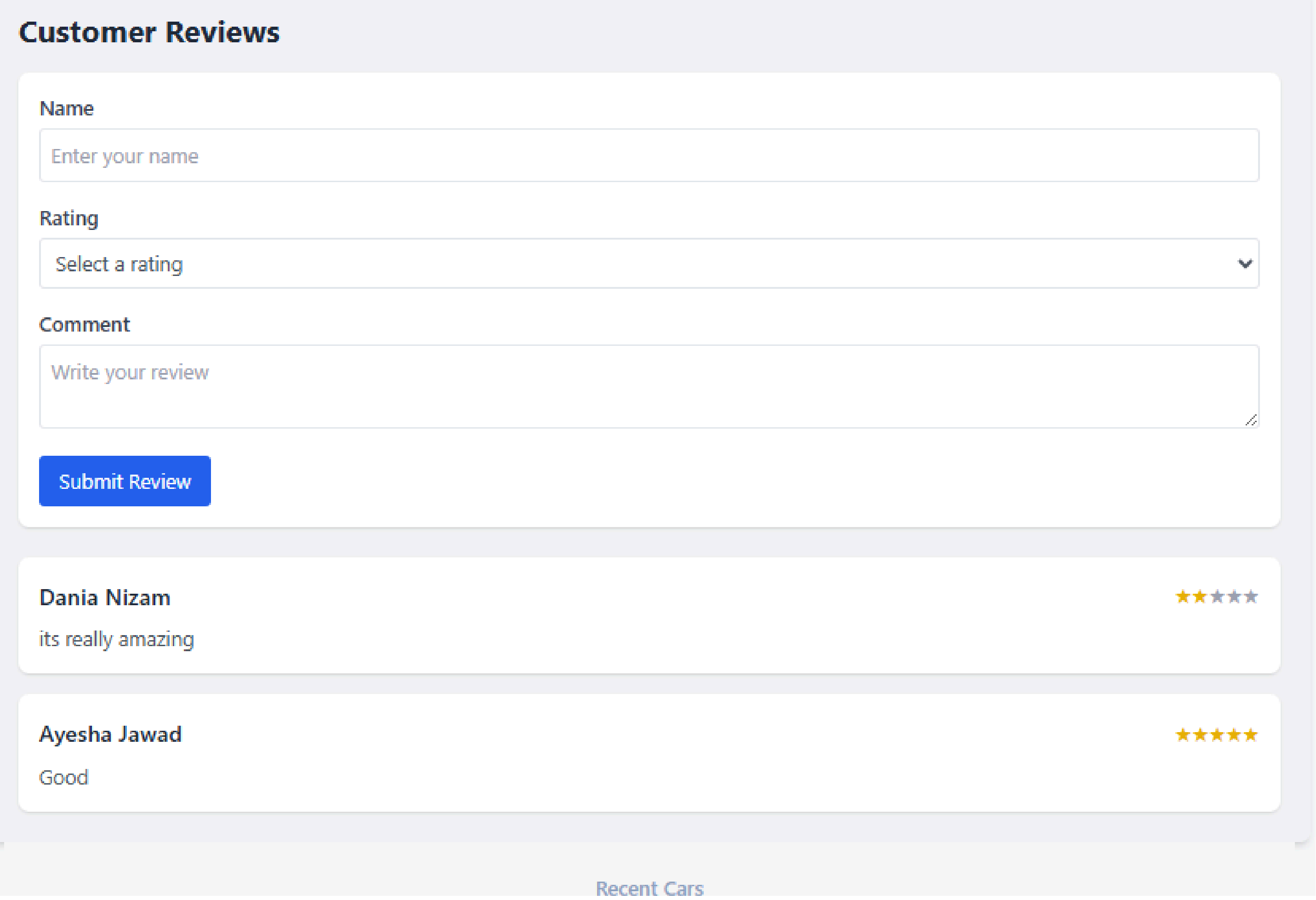
```
                {car.fuel}
                <Image src="/gas.png" alt="" width={24} height={24} />
              </p>
              <p>
                {car.transmission}{" "}
                <Image src="/c.png" alt="" width={24} height={24} />
              </p>
              <p>
                {car.people}{" "}
                <Image src="/pro.png" alt="" width={24} height={24} /> People
              </p>
            </div>

            <div className="flex">
              <div className="w-[116px] h-[44px]">
                <p className="text-lg font-bold mt-2 ">
                  ${car.price.toFixed(2)} / day
                  {car.originalPrice && (
                    <span className="text-sm line-through text-gray-400 ml-2">
                      ${car.originalPrice.toFixed(2)}
                    </span>
                  )}
                </p>
              </div>
              <div className="flex justify-center items-center">
                <Link href={`/car/${car.id}`}>
                  <button className="mt-4 w-[116px] h-[44px] pr-[20px] pl-[20px] gap-[8px] ml-9 bg-[#3563E9] text-whi
                    Rent Now
                  </button>
                </Link>
```

2. Review Component

- **Overview**: Developed a dynamic review system to allow users to leave feedback on cars.
- **Features**:
  - Display existing reviews with user ratings and comments.
  - Form to submit new reviews (rating and text input).
  - Integrated validation to ensure meaningful reviews.
- **Technical Details:**
  - Used a state management system with Redux to handle review submission and updates.

- Connected the component to a backend API to persist reviews in the database.
- Added conditional rendering to show user-specific actions (e.g., editing or deleting their reviews).

- Outcome: Users can view and submit reviews, enriching the user experience with valuable feedback.



Customer Reviews

Name

Enter your name

Rating

Select a rating

Comment

Write your review

Submit Review

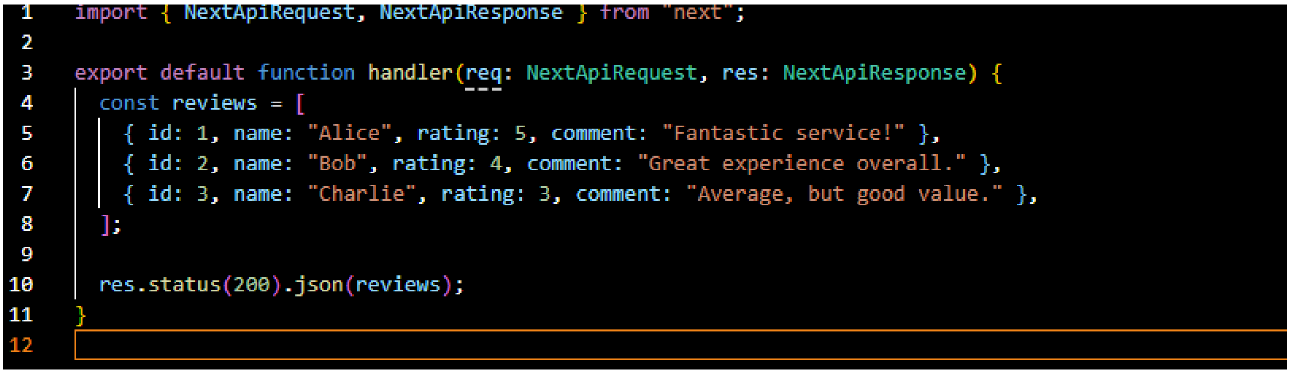
Dania Nizam ★★☆☆☆

its really amazing

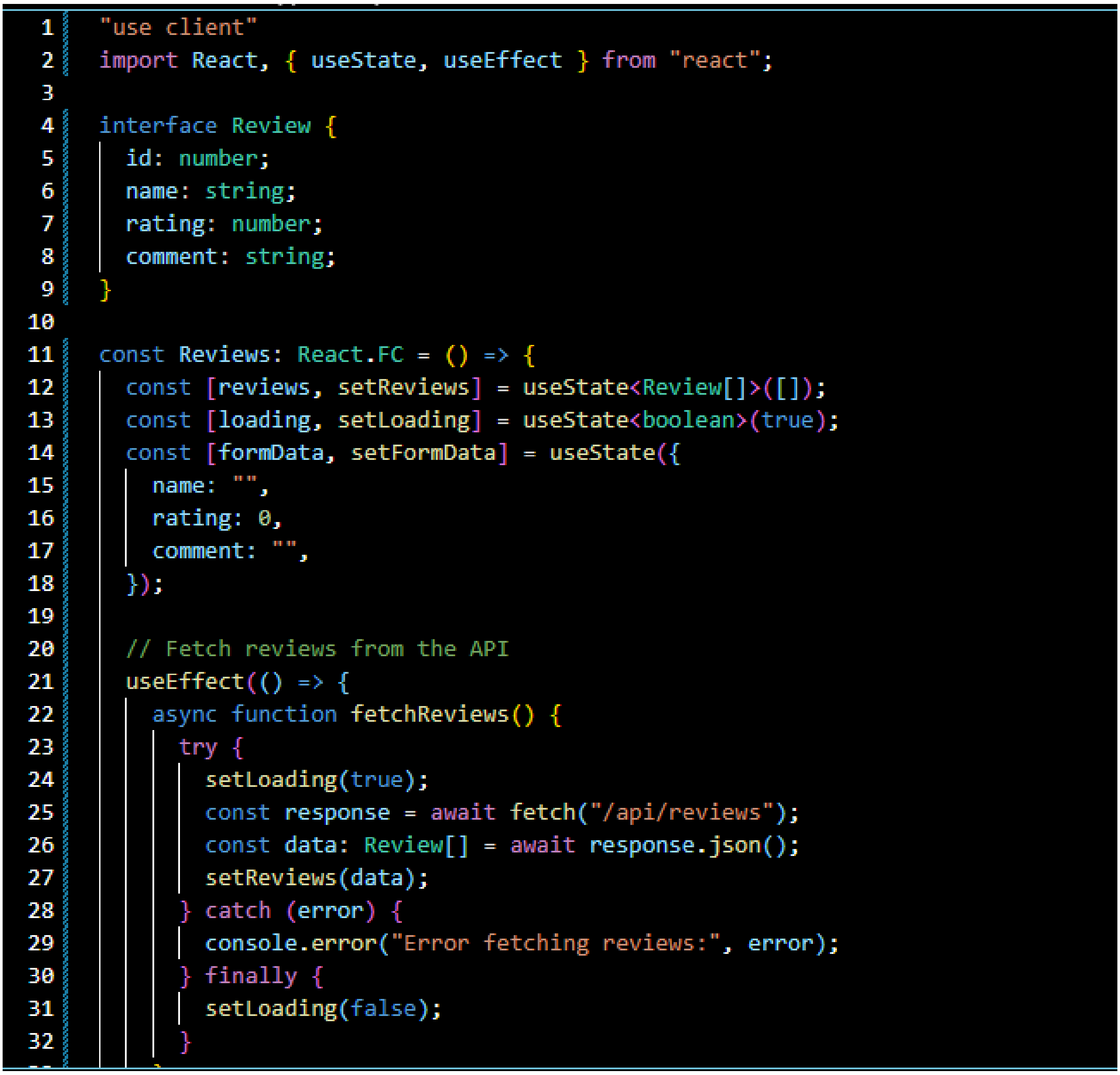
Ayesha Jawad ★★★★★

Good

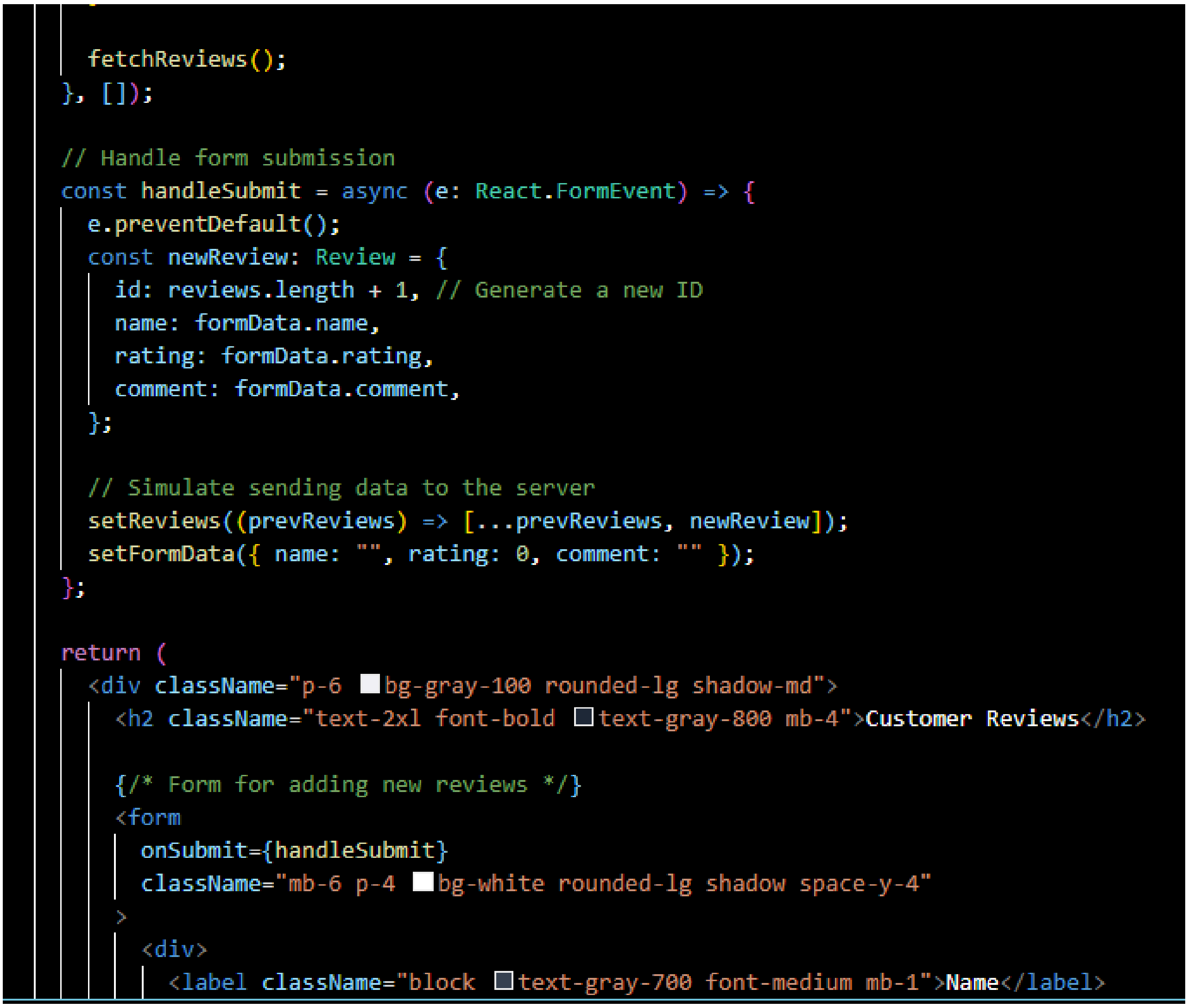
Recent Cars



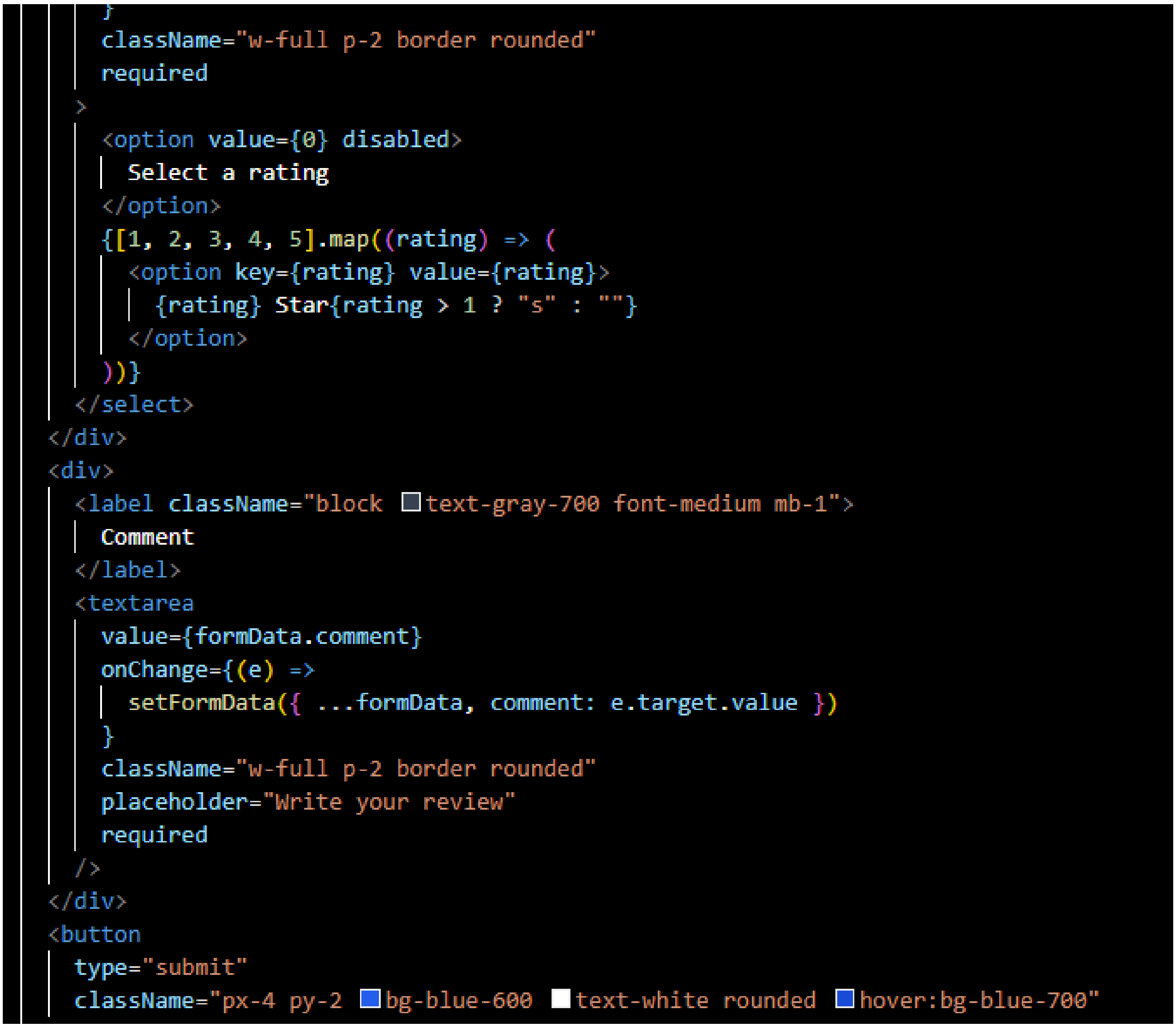
```
import { NextApiRequest, NextApiResponse } from "next";

export default function handler(req: NextApiRequest, res: NextApiResponse) {
  const reviews = [
    { id: 1, name: "Alice", rating: 5, comment: "Fantastic service!" },
    { id: 2, name: "Bob", rating: 4, comment: "Great experience overall." },
    { id: 3, name: "Charlie", rating: 3, comment: "Average, but good value." },
  ];

  res.status(200).json(reviews);
}

```



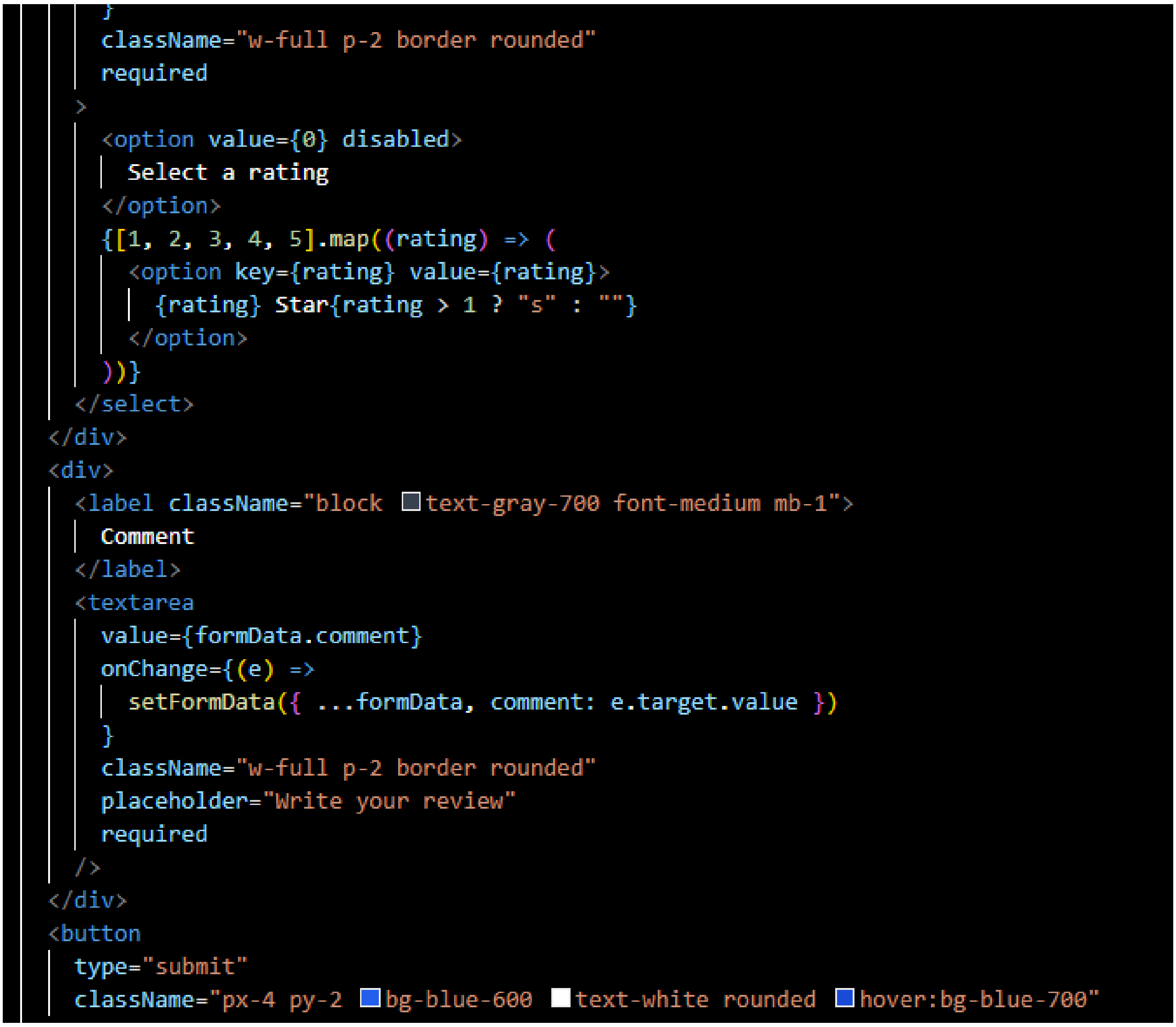
```tsx
"use client"
import React, { useState, useEffect } from "react";

interface Review {
  id: number;
  name: string;
  rating: number;
  comment: string;
}

const Reviews: React.FC = () => {
  const [reviews, setReviews] = useState<Review[]>([]);
  const [loading, setLoading] = useState<boolean>(true);
  const [formData, setFormData] = useState({
    name: "",
    rating: 0,
    comment: "",
  });

  // Fetch reviews from the API
  useEffect(() => {
    async function fetchReviews() {
      try {
        setLoading(true);
        const response = await fetch("/api/reviews");
        const data: Review[] = await response.json();
        setReviews(data);
      } catch (error) {
        console.error("Error fetching reviews:", error);
      } finally {
        setLoading(false);
      }
```



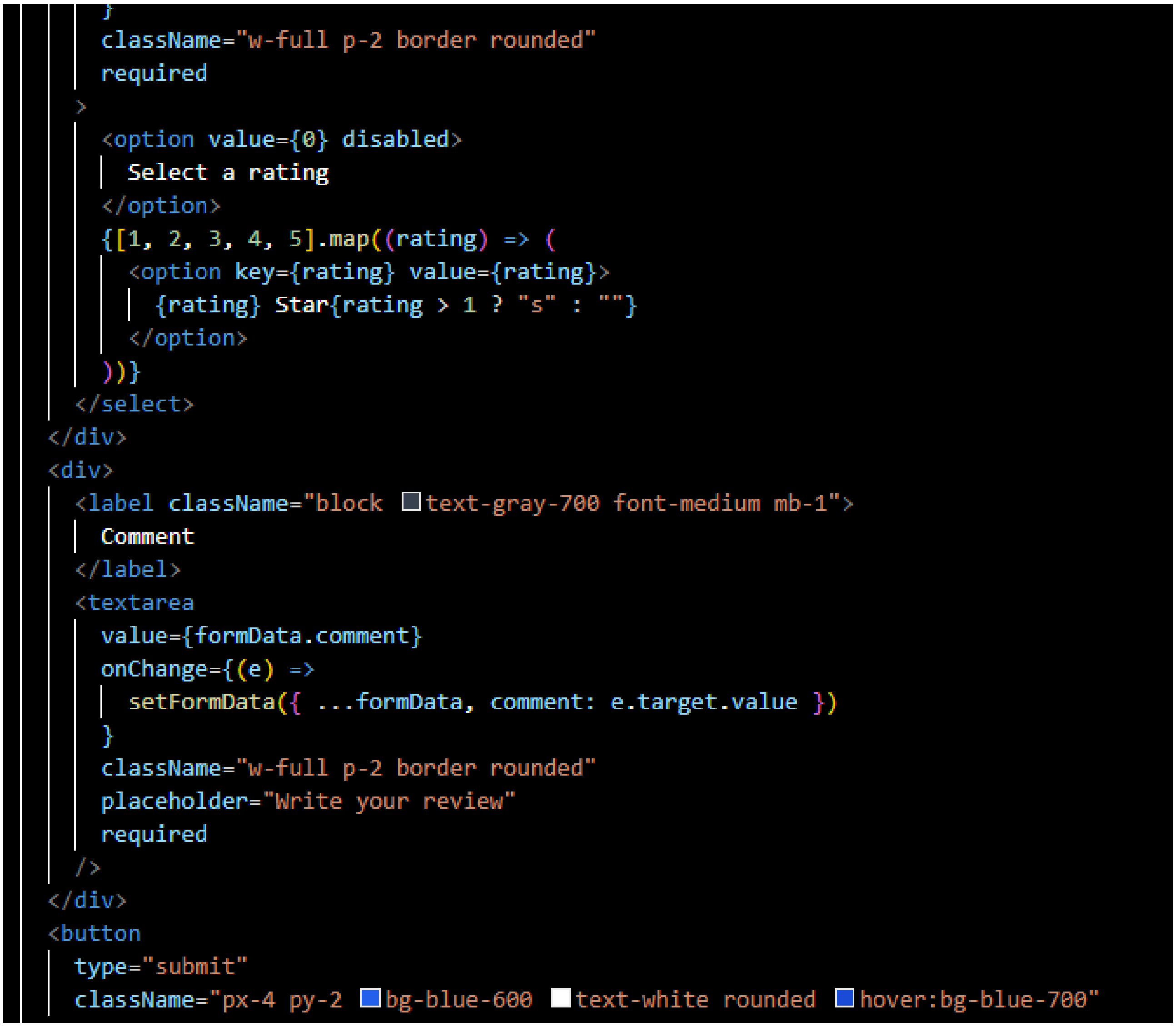
```
    fetchReviews();
  }, []);

  // Handle form submission
  const handleSubmit = async (e: React.FormEvent) => {
    e.preventDefault();
    const newReview: Review = {
      id: reviews.length + 1, // Generate a new ID
      name: formData.name,
      rating: formData.rating,
      comment: formData.comment,
    };

    // Simulate sending data to the server
    setReviews((prevReviews) => [...prevReviews, newReview]);
    setFormData({ name: "", rating: 0, comment: "" });
  };

  return (
    <div className="p-6 □bg-gray-100 rounded-lg shadow-md">
      <h2 className="text-2xl font-bold ■text-gray-800 mb-4">Customer Reviews</h2>

      {/* Form for adding new reviews */}
      <form
        onSubmit={handleSubmit}
        className="mb-6 p-4 □bg-white rounded-lg shadow space-y-4"
      >
        <div>
          <label className="block ■text-gray-700 font-medium mb-1">Name</label>
```



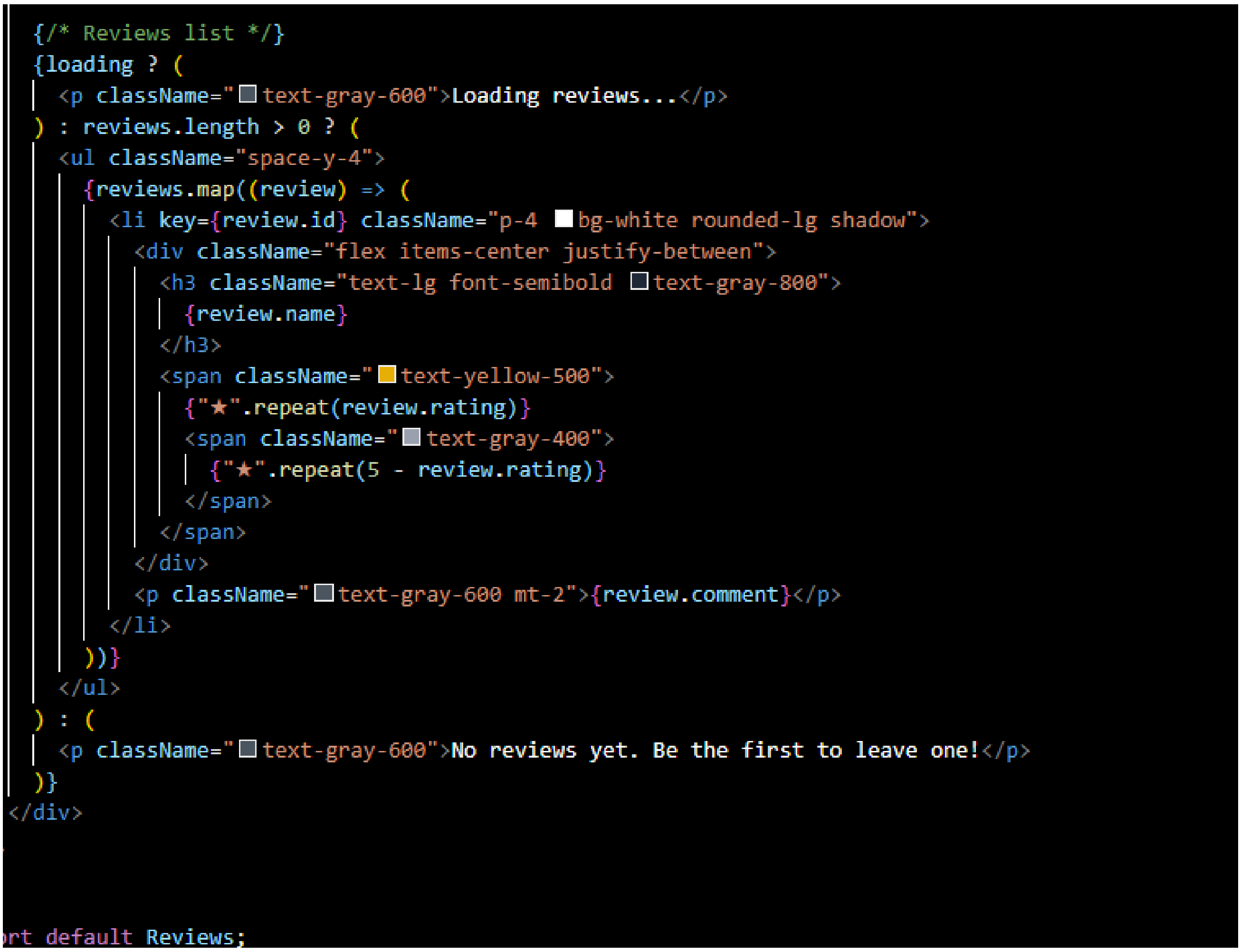
```
    }
    className="w-full p-2 border rounded"
    required
  >
    <option value={0} disabled>
      Select a rating
    </option>
    {[1, 2, 3, 4, 5].map((rating) => (
      <option key={rating} value={rating}>
        {rating} Star{rating > 1 ? "s" : ""}
      </option>
    ))}
  </select>
</div>
<div>
  <label className="block text-gray-700 font-medium mb-1">
    Comment
  </label>
  <textarea
    value={formData.comment}
    onChange={(e) =>
      setFormData({ ...formData, comment: e.target.value })
    }
    className="w-full p-2 border rounded"
    placeholder="Write your review"
    required
  />
</div>
<button
  type="submit"
  className="px-4 py-2 bg-blue-600 text-white rounded hover:bg-blue-700"
```



```
        }
        className="w-full p-2 border rounded"
        required
      >
        <option value={0} disabled>
          Select a rating
        </option>
        {[1, 2, 3, 4, 5].map((rating) => (
          <option key={rating} value={rating}>
            {rating} Star{rating > 1 ? "s" : ""}
          </option>
        ))}
      </select>
    </div>
    <div>
      <label className="block ■text-gray-700 font-medium mb-1">
        Comment
      </label>
      <textarea
        value={formData.comment}
        onChange={(e) =>
          setFormData({ ...formData, comment: e.target.value })
        }
        className="w-full p-2 border rounded"
        placeholder="Write your review"
        required
      />
    </div>
    <button
      type="submit"
      className="px-4 py-2 🟦bg-blue-600 ⬜text-white rounded 🟦hover:bg-blue-700"
```



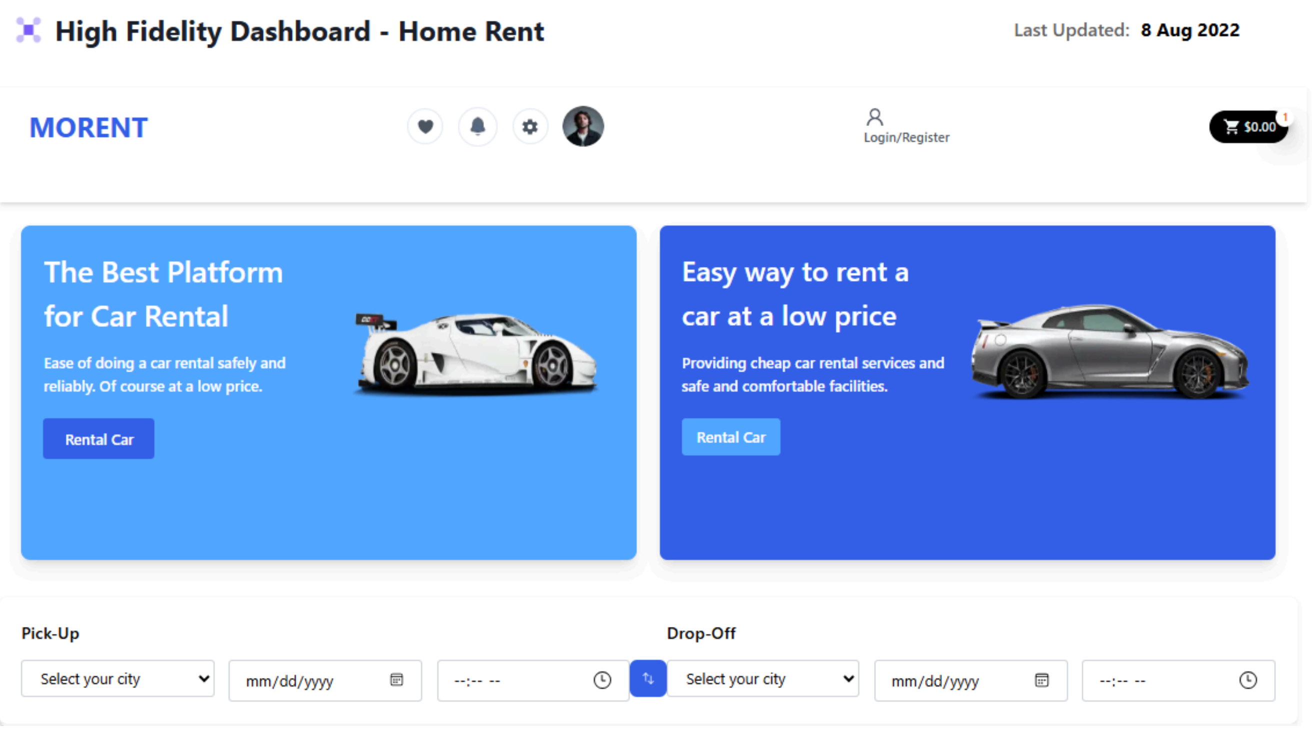
```
      }
      className="w-full p-2 border rounded"
      required
    >
      <option value={0} disabled>
        Select a rating
      </option>
      {[1, 2, 3, 4, 5].map((rating) => (
        <option key={rating} value={rating}>
          {rating} Star{rating > 1 ? "s" : ""}
        </option>
      ))}
    </select>
  </div>
  <div>
    <label className="block ■text-gray-700 font-medium mb-1">
      Comment
    </label>
    <textarea
      value={formData.comment}
      onChange={(e) =>
        setFormData({ ...formData, comment: e.target.value })
      }
      className="w-full p-2 border rounded"
      placeholder="Write your review"
      required
    />
  </div>
  <button
    type="submit"
    className="px-4 py-2 🟦bg-blue-600 □text-white rounded 🟦hover:bg-blue-700"
```



```
  {/* Reviews list */}
  {loading ? (
    <p className="text-gray-600">Loading reviews...</p>
  ) : reviews.length > 0 ? (
    <ul className="space-y-4">
      {reviews.map((review) => (
        <li key={review.id} className="p-4 bg-white rounded-lg shadow">
          <div className="flex items-center justify-between">
            <h3 className="text-lg font-semibold text-gray-800">
              {review.name}
            </h3>
            <span className="text-yellow-500">
              {"★".repeat(review.rating)}
              <span className="text-gray-400">
                {"★".repeat(5 - review.rating)}
              </span>
            </span>
          </div>
          <p className="text-gray-600 mt-2">{review.comment}</p>
        </li>
      ))}
    </ul>
  ) : (
    <p className="text-gray-600">No reviews yet. Be the first to leave one!</p>
  )}
</div>


rt default Reviews;
```

3. Authentication System

- **Overview**: Integrated a secure login/logout system using NextAuth with Google and GitHub providers.
- **Implementation**:
  - Configured NextAuth in the [...nextauth].js API route.
  - Enabled authentication providers (Google and GitHub).
  - Secured sensitive pages and actions using getSession and useSession hooks.
  - Used Redux to manage user session state across components.
- **Outcome**: Users can log in and log out seamlessly, with their session states managed dynamically.



High Fidelity Dashboard - Home Rent

Last Updated: 8 Aug 2022

The Best Platform for Car Rental

Ease of doing a car rental safely and reliably. Of course at a low price.

Easy way to rent a car at a low price

Providing cheap car rental services and safe and comfortable facilities.

Technologies UsedFrontend: React, Next.js

- **Backend**: Sanity
- **Authentication**: NextAuth (Google and GitHub providers)
- **State Management**: Redux Toolkit
- **Styling**: Tailwind CSS for responsive and modern UI

Conclusion:

Day 4 marked significant progress in building a user-friendly and functional Car Rental Marketplace. The addition of dynamic routing, a robust review system, and seamless authentication has enhanced the platform’s interactivity and usability. The application is well on its way to becoming a comprehensive solution for car rentals.