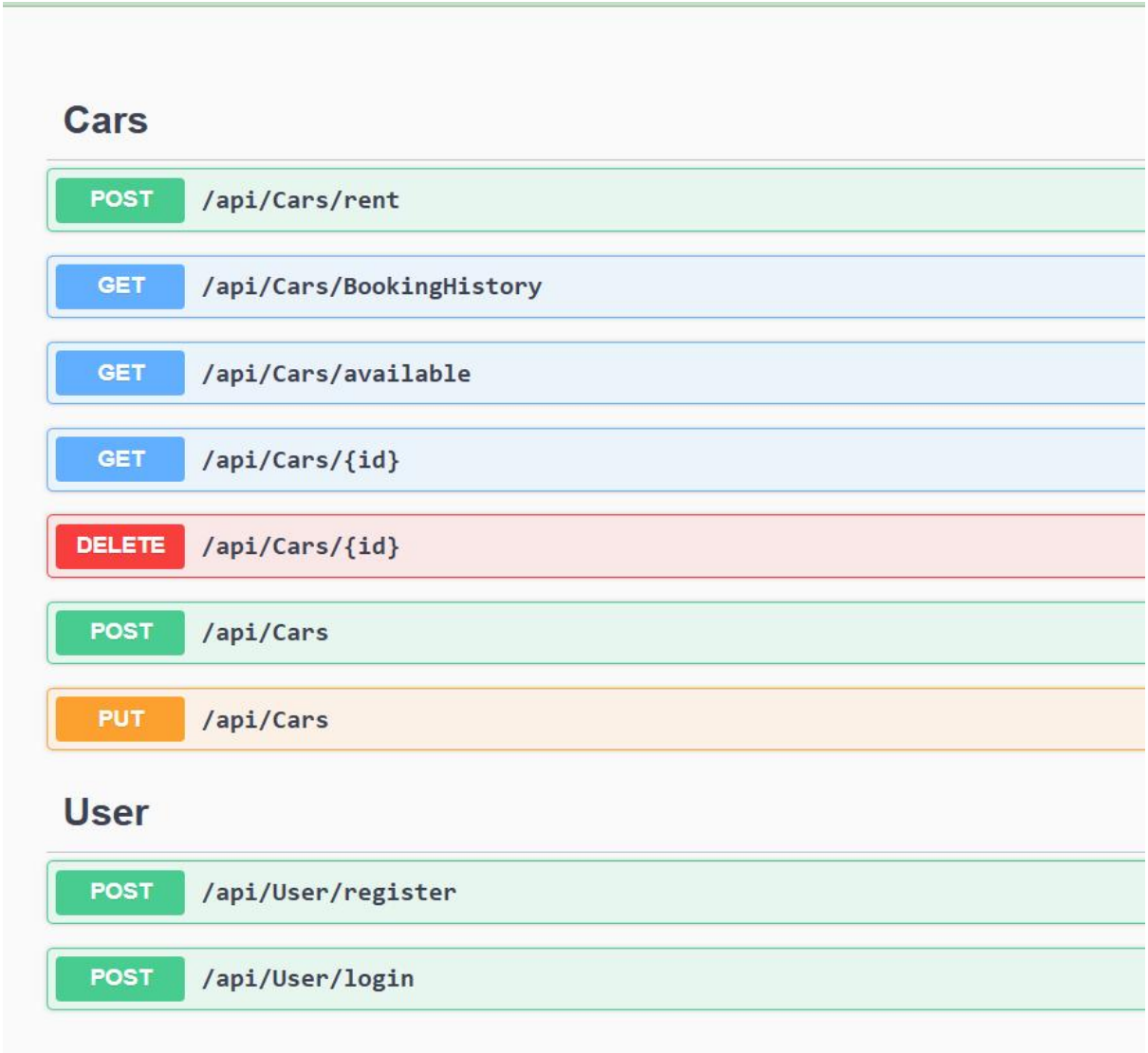
Car Rental System API using C#



Car Model

The Car class represents a car in the car rental system and includes the following properties:

- **Id:** A unique identifier for the car.
- **Make:** The make of the car (e.g., Toyota, Ford). It is a required field, and the length must be between 1 and 50 characters.
- **Model:** The model of the car. It is also a required field, with the length constrained between 1 and 50 characters.

- **Year:** The manufacturing year of the car. The custom validation ValidYear ensures that this field contains a valid year value starting from 1990 to current year.
- **PricePerDay:** The rental price per day for the car. It must be a positive value greater than 0.
- **isAvailable:** A boolean flag indicating if the car is currently available for rent. It defaults to true (i.e., available).

User Model

The User class represents a user in the system (either user or an admin) and includes the following properties:

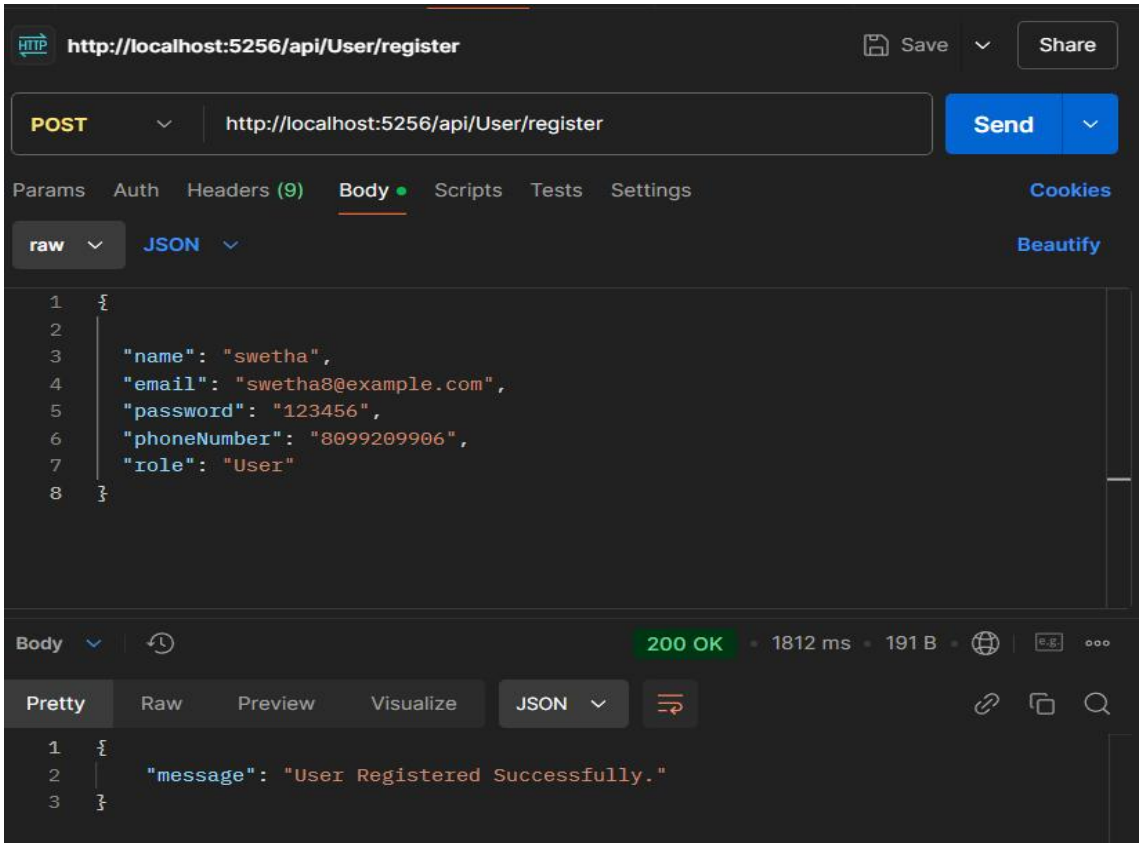
- **Id:** A unique identifier for the user.
- **Name:** The name of the user. It is required and must have a length between 1 and 100 characters.
- **Email:** The user's email address. It is required and must be a valid email format.
- **Password:** The password for user authentication. It is required, and the length must be at least 6 characters.
- **PhoneNumber:** The user's phone number. It is required and must be exactly 10 digits.
- **Role:** The role of the user (either "Admin" or "User"). It is required and validated to ensure that only these two values are allowed.

1.User

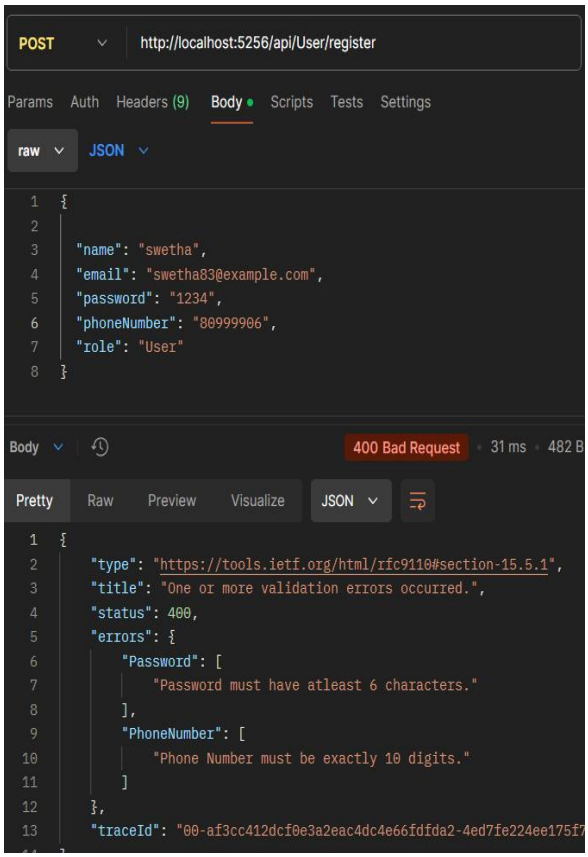
1. **POST /api/User/register:**

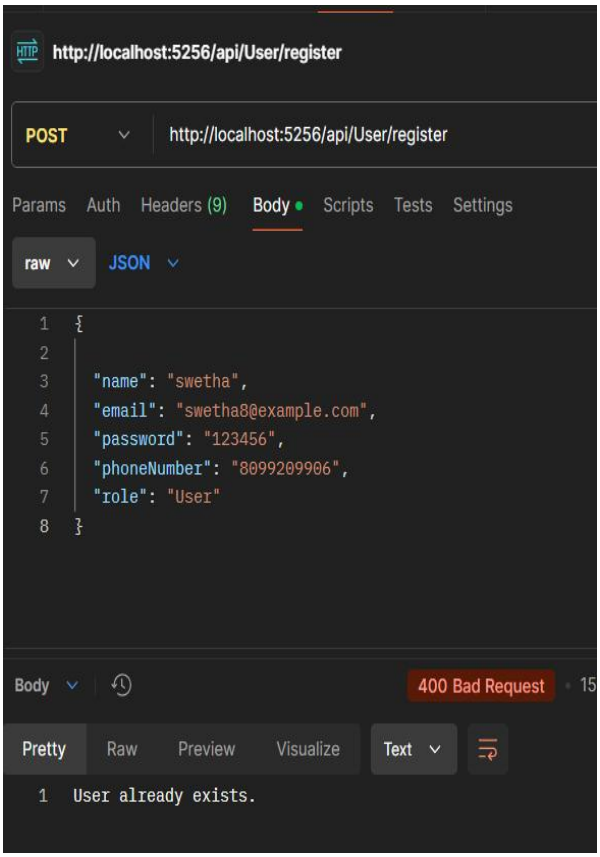
- Registers a new user in the system.It takes user details as input and create user by storing in database.Outputs user created message and successful status code.
- Password is hashed by using bcrypt

Successful user Creation:



Invalid user

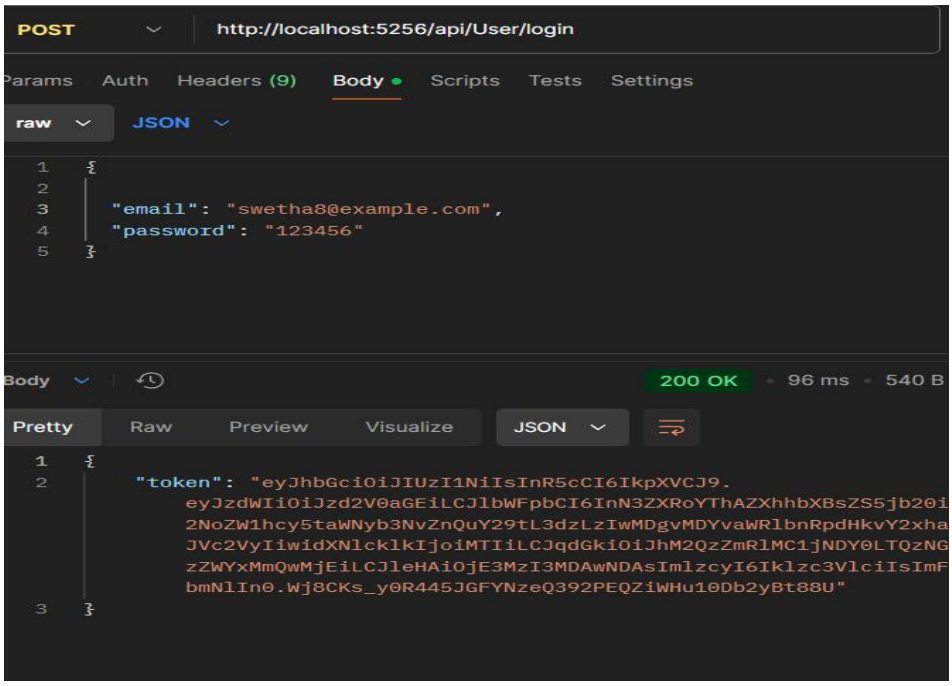




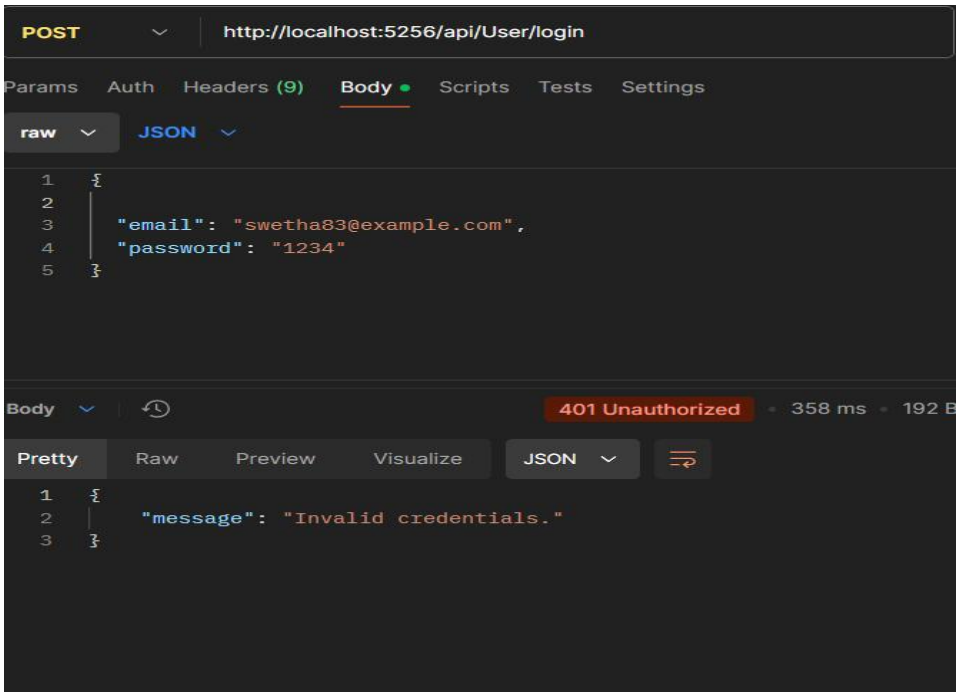
2.POST /api/User/login

- Authenticates a user and generates a JWT token.Takes Valid email and password as input.
- Provides Authentication success with a JWT token.

Valid Login:



Invalid Login:

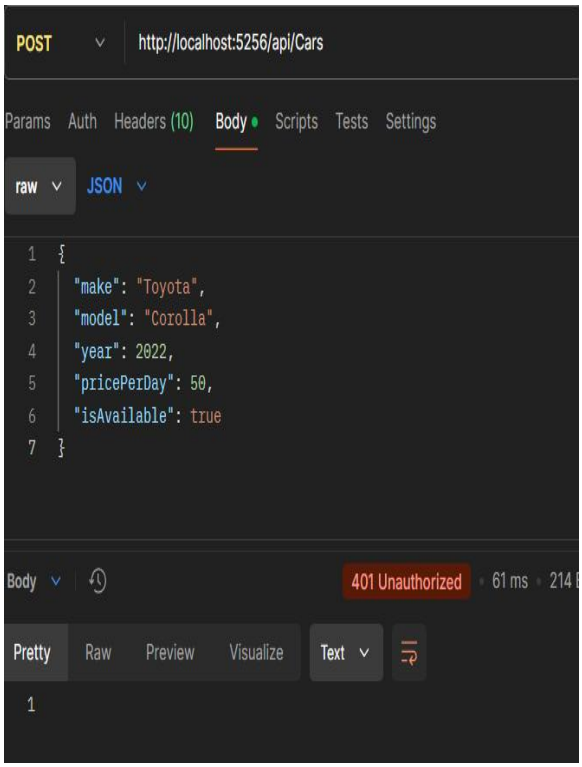


Car Management (Admin)

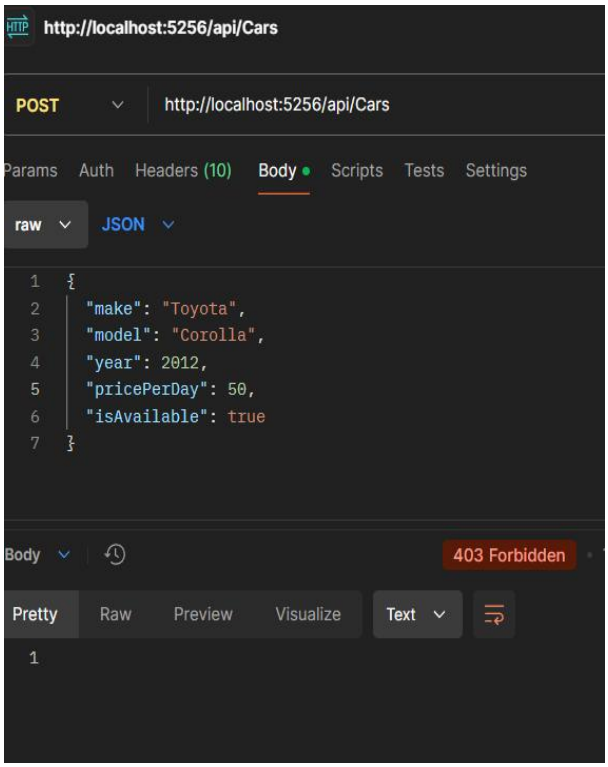
1. **POST /api/Cars**

- Adds a new car to the system (admin-only operation).Car details such as Make, Model, Year, and PricePerDay as input
- Gives Confirmation message with the added car details as output.

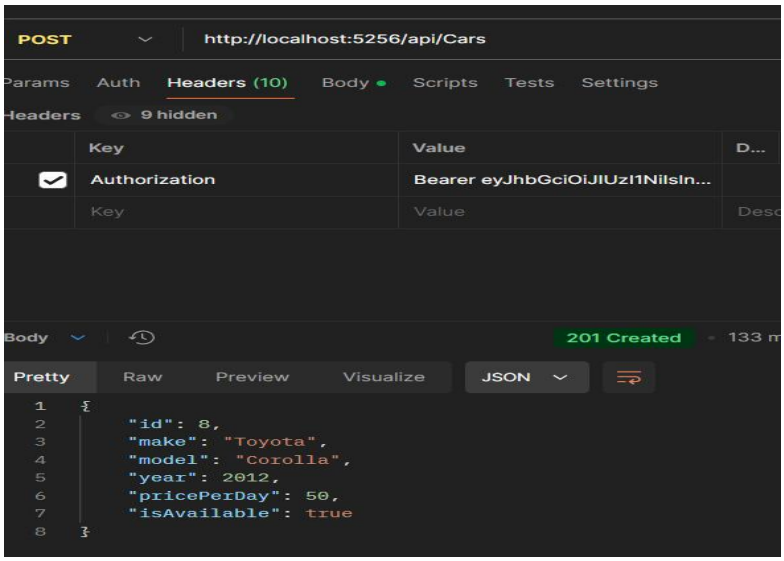
UnAuthorized User:



Authrozied but Role:User



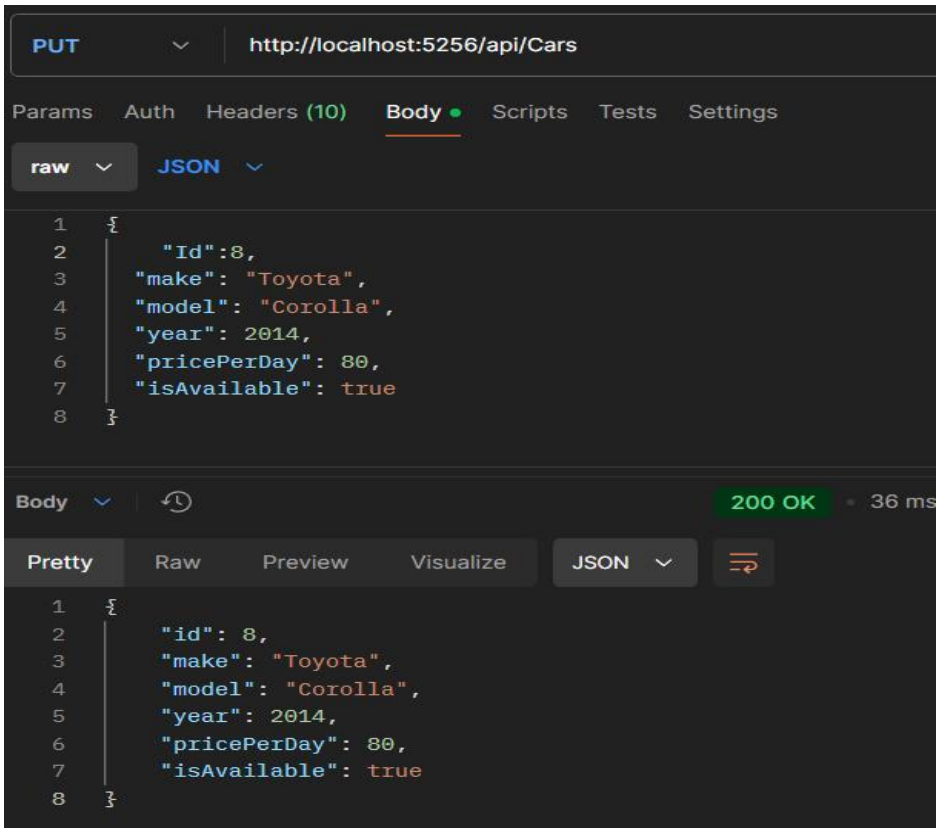
Valid Creation:



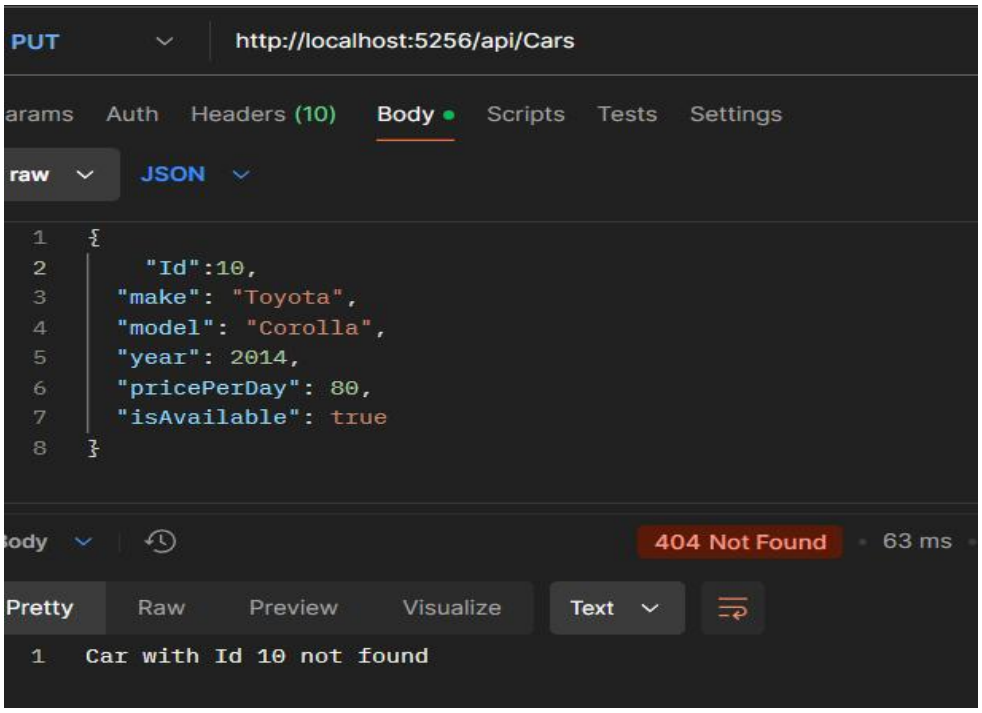
2. **PUT** /api/Cars

- **Purpose:** Updates details of an existing car .Car id and fields to update.
- Updated car details gives as output.

Updated car details price from 50 to 80 and year to 2014.

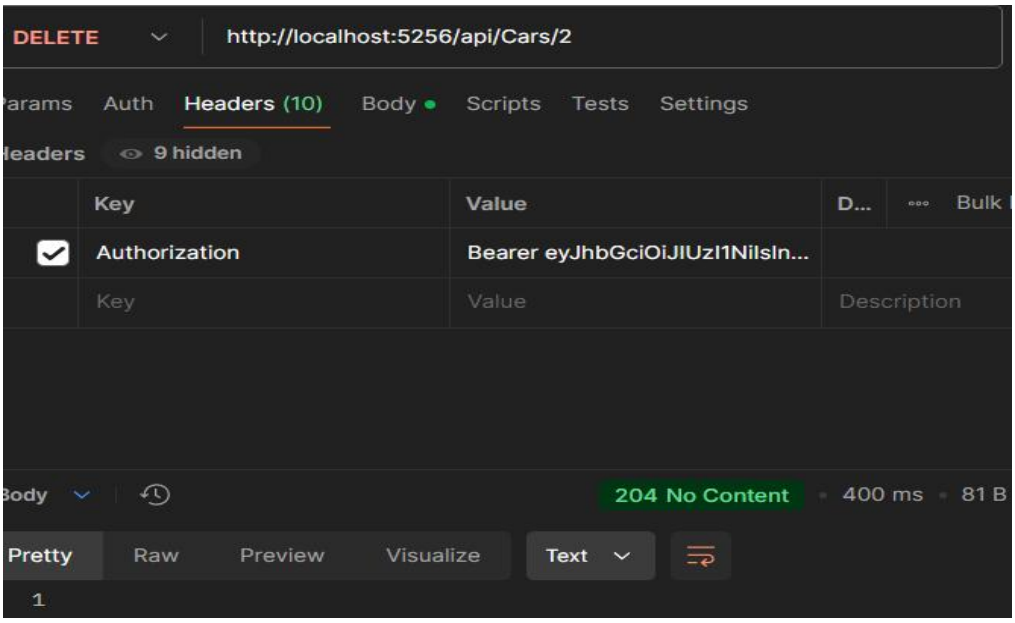


Car Id not found:



3. **DELETE** /api/Cars/{id}

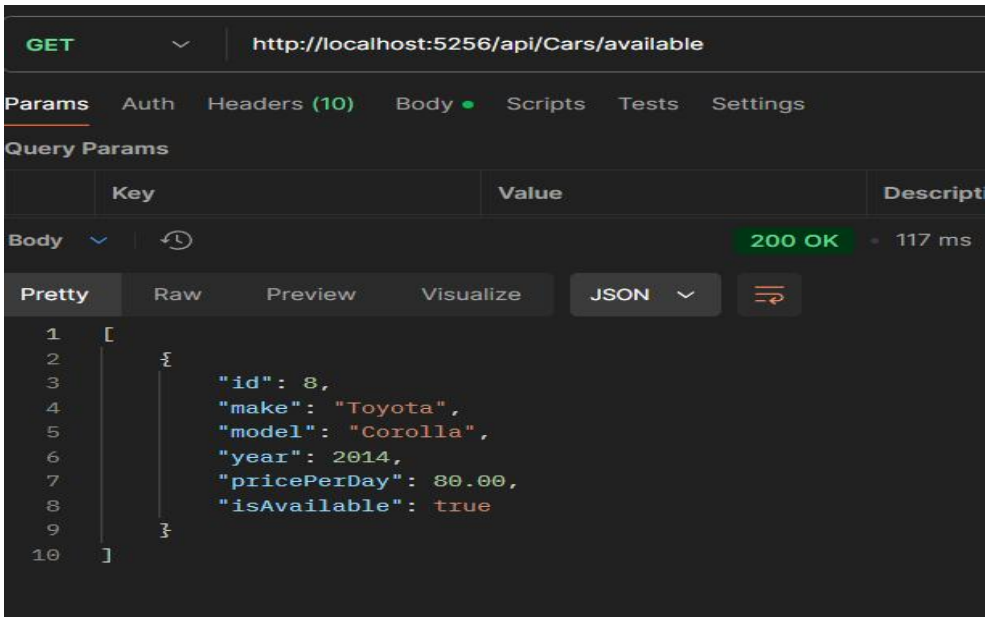
- Deletes a car record from the system (admin-only operation) by taking id as input.
- Expected Output: No Content upon successful deletion.



2. Car Management (allow Aynonymous users)

1. GET /api/Cars/available

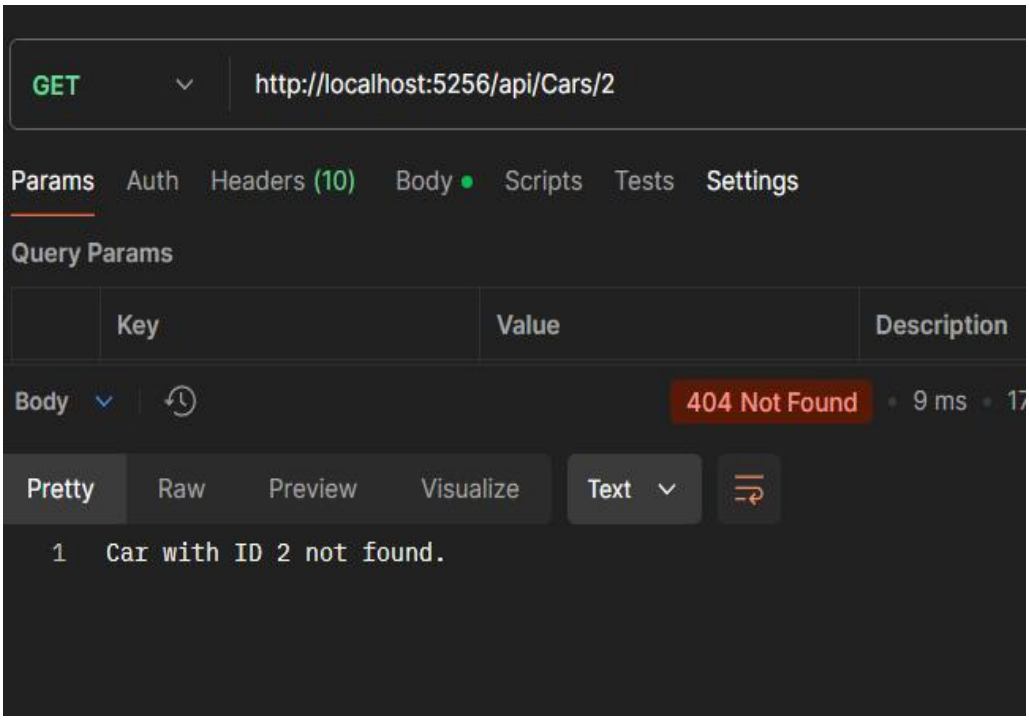
- Fetches a list of cars currently available for rent.
- Expected Output: Array of car objects marked as IsAvailable: true.



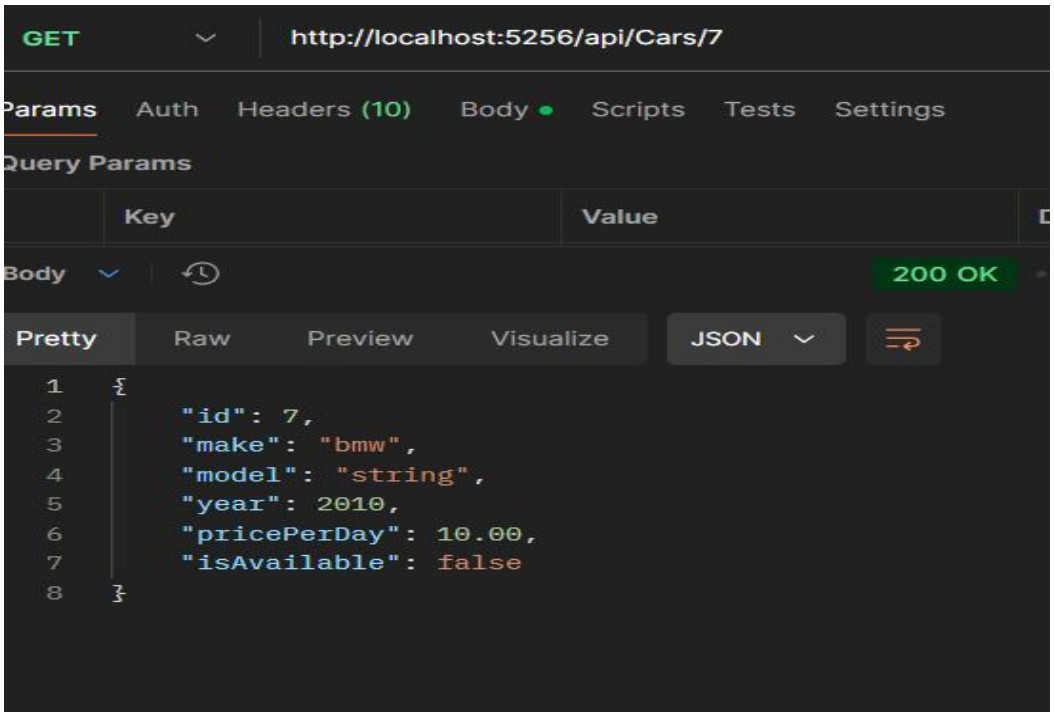
2. GET /api/Cars/{id}

- Retrieves detailed information for a specific car based on the car's id.
- Expected Output: Car details including make, model, year, price, and availability.

When carId is not present:



Valid id:

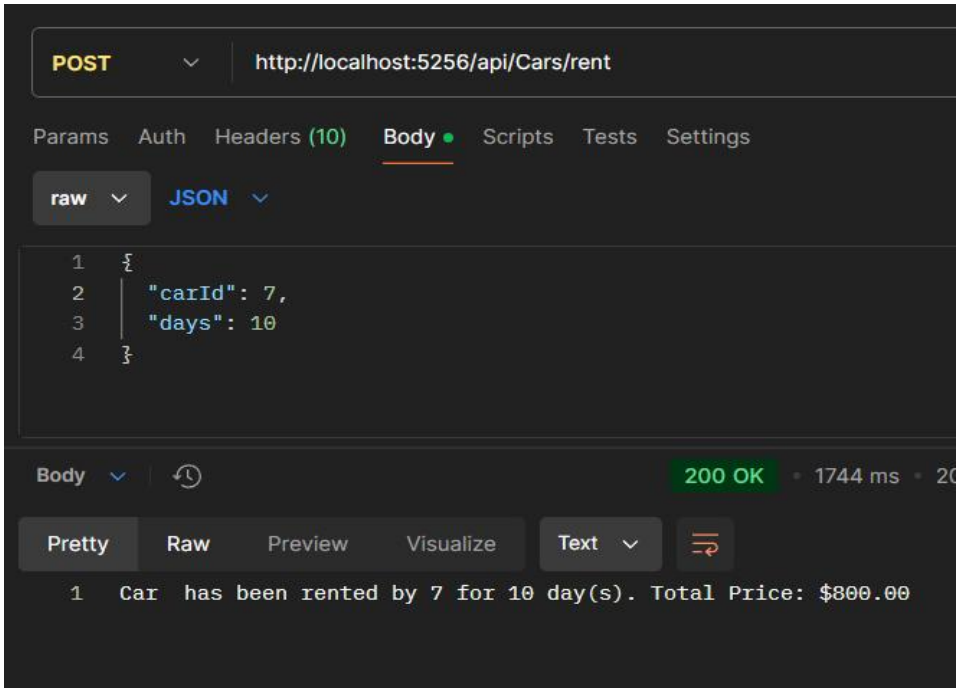


3. Car Rental (Only Authenticated Users)

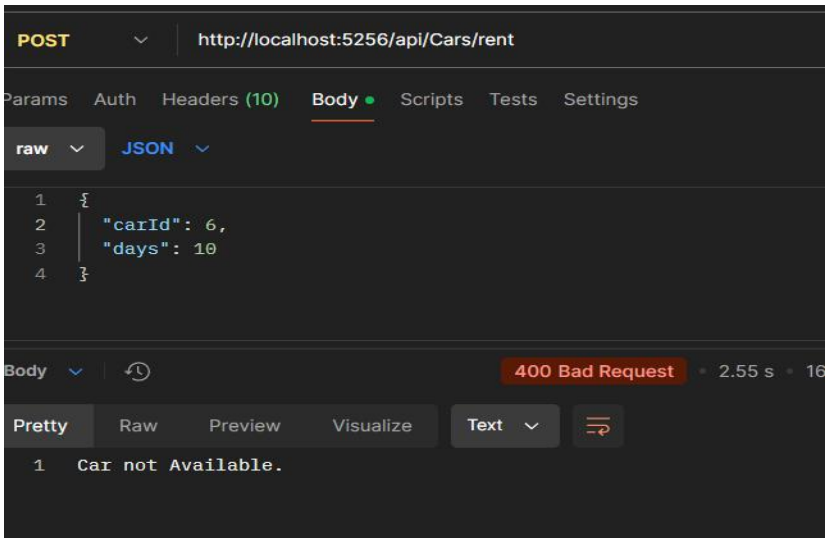
1. POST /api/Cars/rent

- Allows users to rent a car by providing required details like CarId, rental dates.User info is retrived from the token.
- Expected Output: Rental confirmation message with calculated total cost.
- Sents SMS to user phoneNumber on sucessful booking and stores the info in tranaction table.

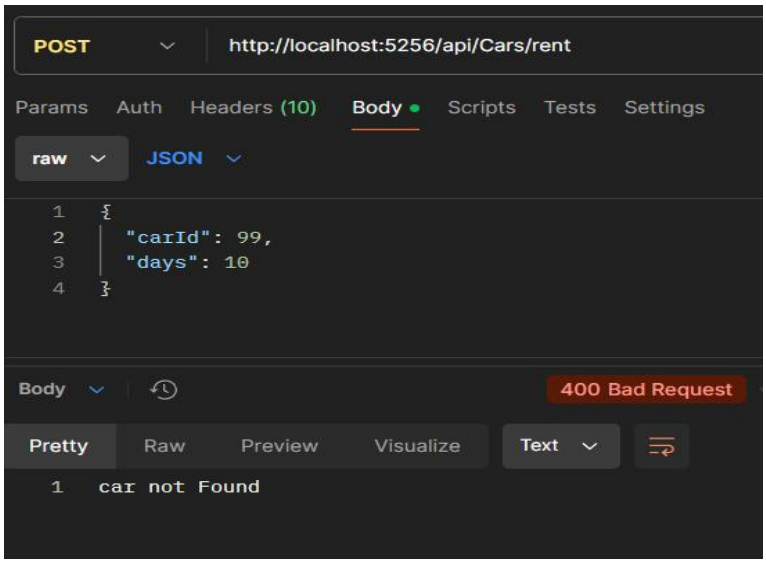
Case1: Success



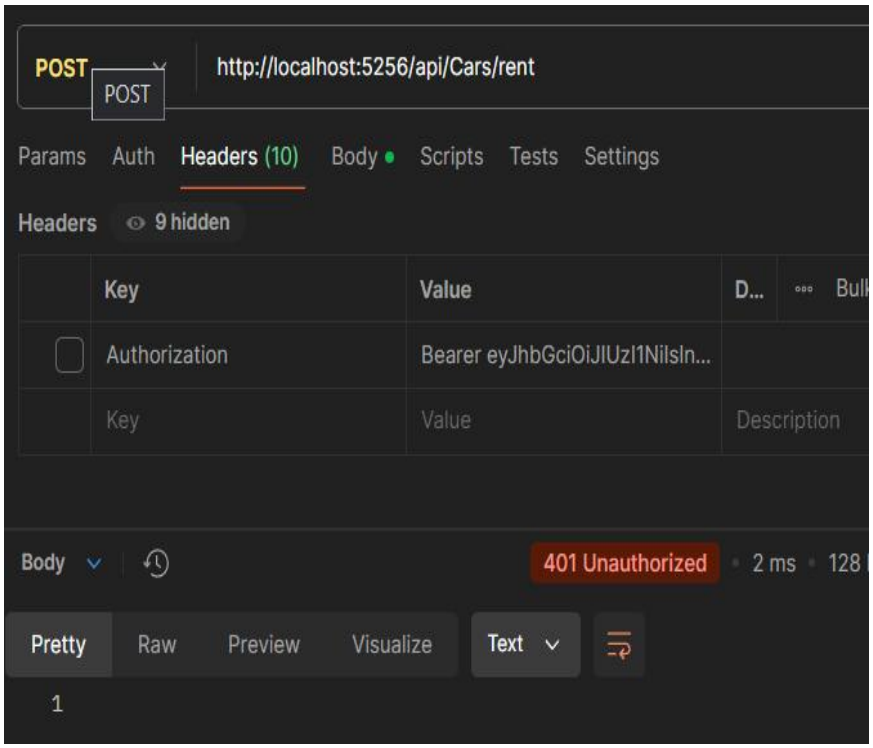
Case2: Requested Car not Available.



Case3: Requested CarId does not exist



4.Case4: Unauthorized Access:



2. GET /api/Cars/BookingHistory

- Retrieves the booking history for the authenticated user.
- Expected Output: List of all past and current bookings for the user.