APARNA BATHULA
A20442593

VEHICLE RENTAL MANAGEMENT SYSTEM

Vehicle rental management system is designed to manage the vehicle rental and booking process for a car rental company. It stores information about vehicles, customers, bookings, and reservations. Customers can browse available cars, make reservations, and look over their booking history, while administrators can manage the collection of cars, bookings, and information about customers.

Implementation Overview:

Database Management System (DBMS): MySQL
Programming Language: PHP for server-side scripting
Front-end: HTML, CSS for user interface
Web Server: Apache (used with XAMPP)
Tools: XAMPP for local development

Entities:

Customer:
- Represents individuals who rent cars.
- Contains attributes customer_id, name, email, phone_number, and password.

Admin:
- Represents administrators who manage the car rental system.
- Contains attributes admin_id, name, and email.

Car:
- Represents the vehicles available for rent.
- Includes attributes car_id, make, model, and condition, detailing car specifications.

Reservation:
- Tracks individual bookings made by customers.
- Contains reservation details such as reservation_id, car_id, and customer_id, linking customers to rented cars.

Location:
- Represents the physical locations where cars are available for pickup and return.
- Can include attributes like location_id, address, and contact information.

Booking:
- Contains records of confirmed car reservations.
- Holds attributes such as booking_id, reservation_id, customer_id, pickup_date, return_date, and other booking-related information.

Relationships:

One-to-One Relationship:
Location and Admin have a one-to-one relationship. Each location has a single Admin responsible for it.

One-to-Many Relationships:
Admin (One) to Car (Many): An admin can manage multiple cars. Each car is managed by a single admin.
Customer (One) to Reservation (Many): A customer can make multiple reservations. Each reservation is associated with a single customer.

Many-to-Many Relationship:
Customer and Car have a many-to-many relationship: Multiple customers can rent multiple cars. Each customer can make multiple bookings (rent multiple cars). Each car can be rented by multiple customers.

Triggers:

CustomerRegistrationTrigger : This trigger is set to activate after a new customer is inserted into the customer table. It records the event in the CustomerRegistrationLog table, indicating that a customer has been registered along with a timestamp. This can be useful for tracking when new customers are added to the system.

CarReservationTrigger: This trigger is set to activate after a new reservation is inserted into the reservation table. It records the event in the CarReservationLog table, indicating that a car has been reserved, along with a timestamp. This can be useful for tracking when car reservations are made in the system.

Views:

reservationdetails view is used to retrieve specific details about each reservation, making it easier to access and display relevant data. It simplifies the process of querying reservation-related information, helping to create reports, display reservation details to customers, or support other system functionalities.

StoredProcedure:

The stored procedure (**BookCar**) checks if a car is available, and if so, creates a reservation and returns the reservation ID.

Tables and BCNF:

Customer Table:

- The primary key customer_id uniquely identifies each customer.
- There are no non-trivial functional dependencies. No attribute functionally depends on a proper subset of the primary key.

- The customer table is in BCNF as customer_id is a superkey, and there are no non-trivial functional dependencies.

Admin Table:

- The primary key admin_id uniquely identifies each admin.
- There are no non-trivial functional dependencies. No attribute functionally depends on a proper subset of the primary key.
- The admin table is in BCNF as admin_id is a superkey, and there are no non-trivial functional dependencies.

Car Table:

- The primary key car_id uniquely identifies each car.
- The attribute location_id is a foreign key that should reference a unique key in the location table. This ensures that location_id is determined by a superkey from the referenced table.
- The car table is in BCNF because car_id is a superkey, and location_id is determined by a superkey from the referenced location table.

Reservation Table:

- The primary key reservation_id uniquely identifies each reservation.
- The attributes car_id and customer_id are foreign keys that should reference unique keys in the car and customer tables, respectively. This ensures that both car_id and customer_id are determined by superkeys from their referenced tables.
- The reservation table is in BCNF because reservation_id, car_id, and customer_id are either superkeys or determined by superkeys from their referenced tables.

Booking Table:

- The primary key booking_id uniquely identifies each booking.
- The attributes reservation_id and customer_id are foreign keys that should reference unique keys in the reservation and customer tables, respectively. This ensures that both reservation_id and customer_id are determined by superkeys from their referenced tables.
- The booking table is in BCNF because booking_id, reservation_id, and customer_id are either superkeys or determined by superkeys from their referenced tables.

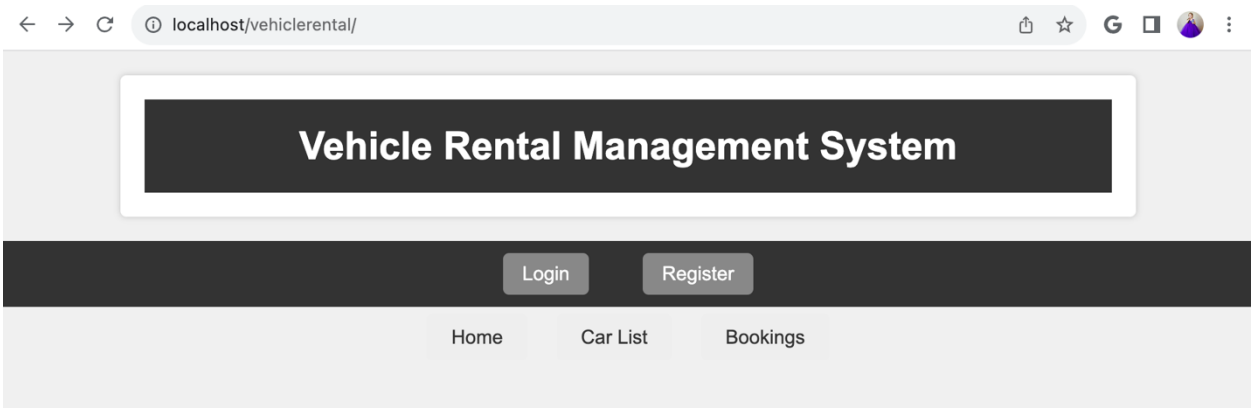
Location Table:

- The primary key location_id uniquely identifies each location.
- There are no non-trivial functional dependencies. No attribute functionally depends on a proper subset of the primary key.
- The location table is in BCNF as location_id is a superkey, and there are no non-trivial functional dependencies.

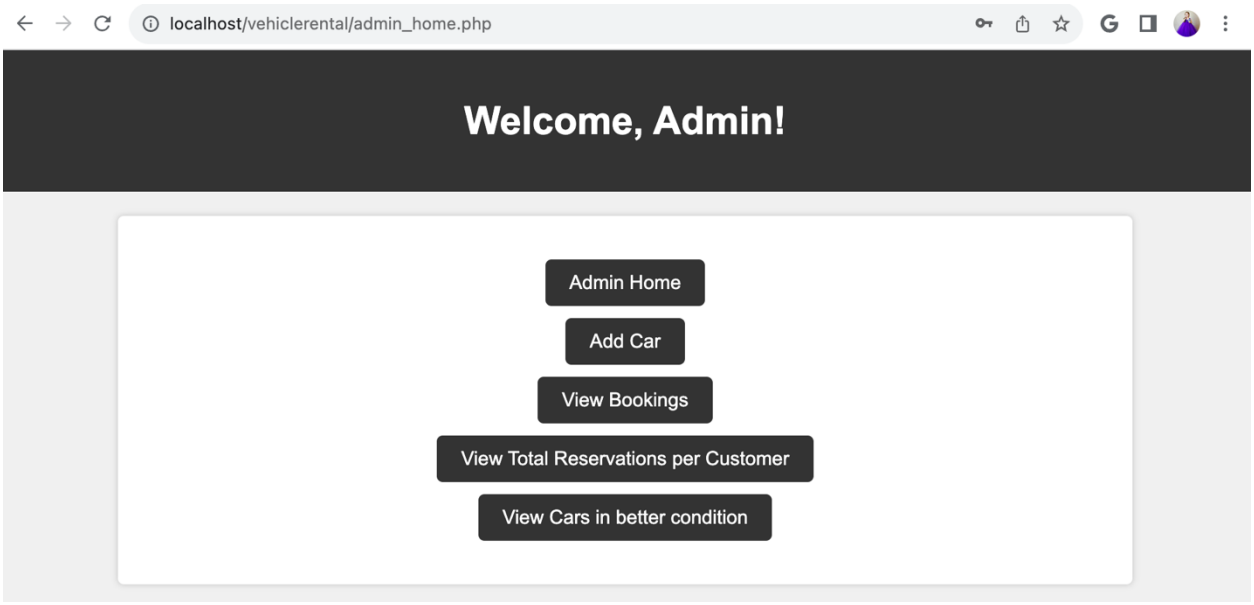
Each table in the given database schema is in BCNF as they meet the BCNF conditions, with primary keys ensuring unique identification and foreign keys referencing attributes determined by superkeys in the referenced tables.

APPENDIX:

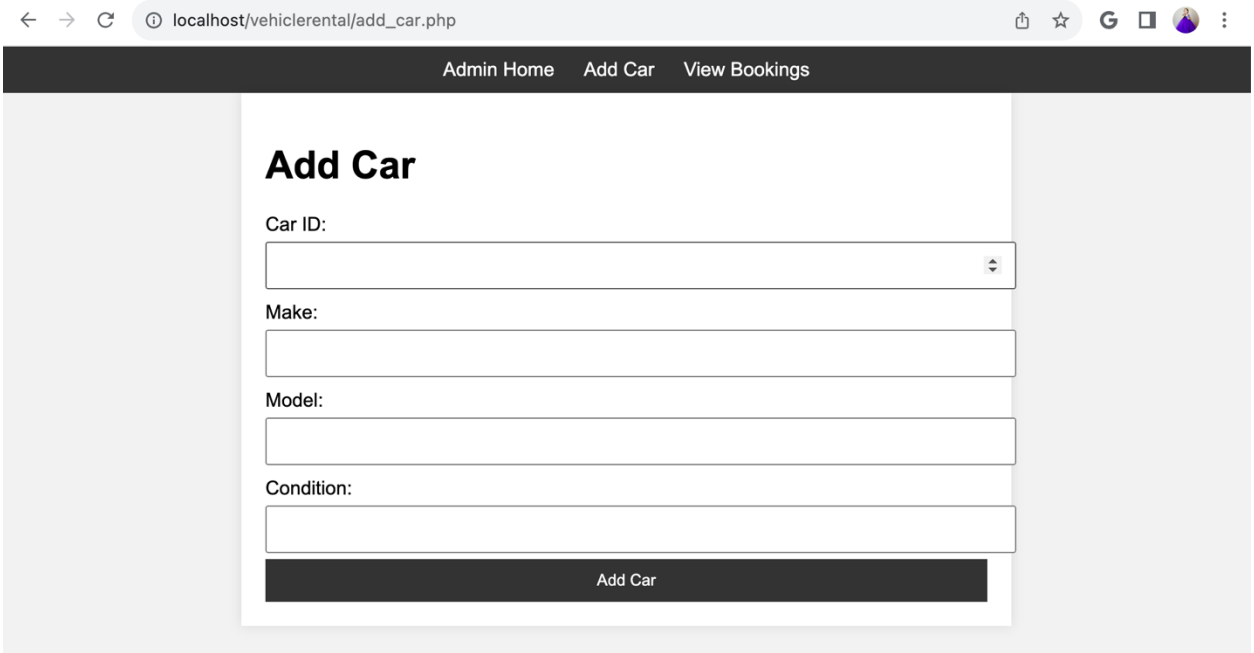
1. Home page: customer or admin can login/register and view cars and bookings.



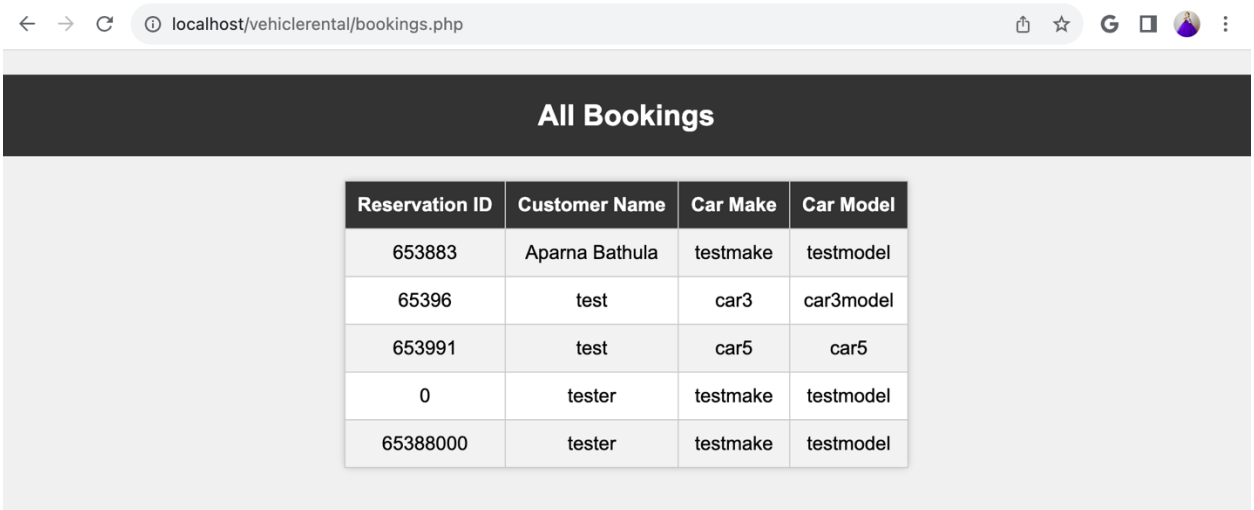
2. Admin page: When logged in with admin credentials, admin can add cars, manage bookings and customer details.



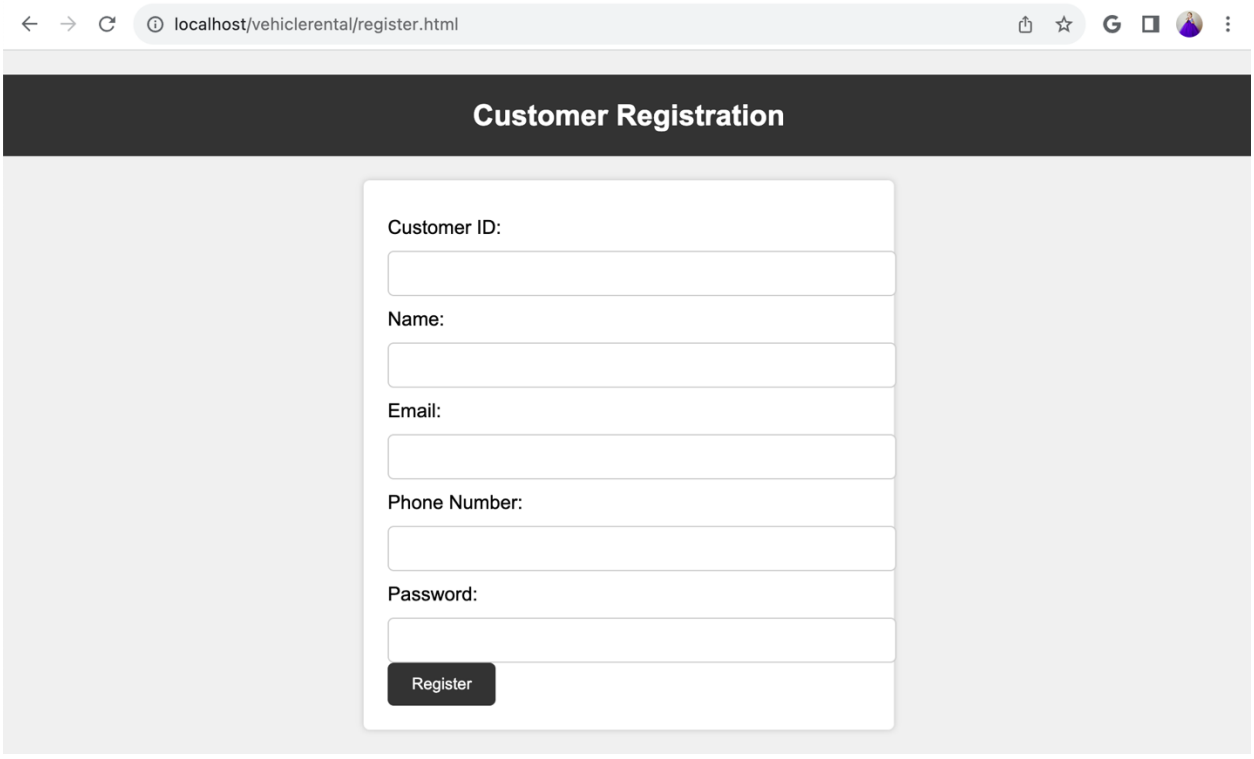
3. Add Car page: The Admin can enter details of car for listing.



4. View all bookings page where admin can see all the bookings and manage them.



5. Customer registration page: new customers can register to book cars.



6. Car listing page: Customers can see all the available cars to rent.