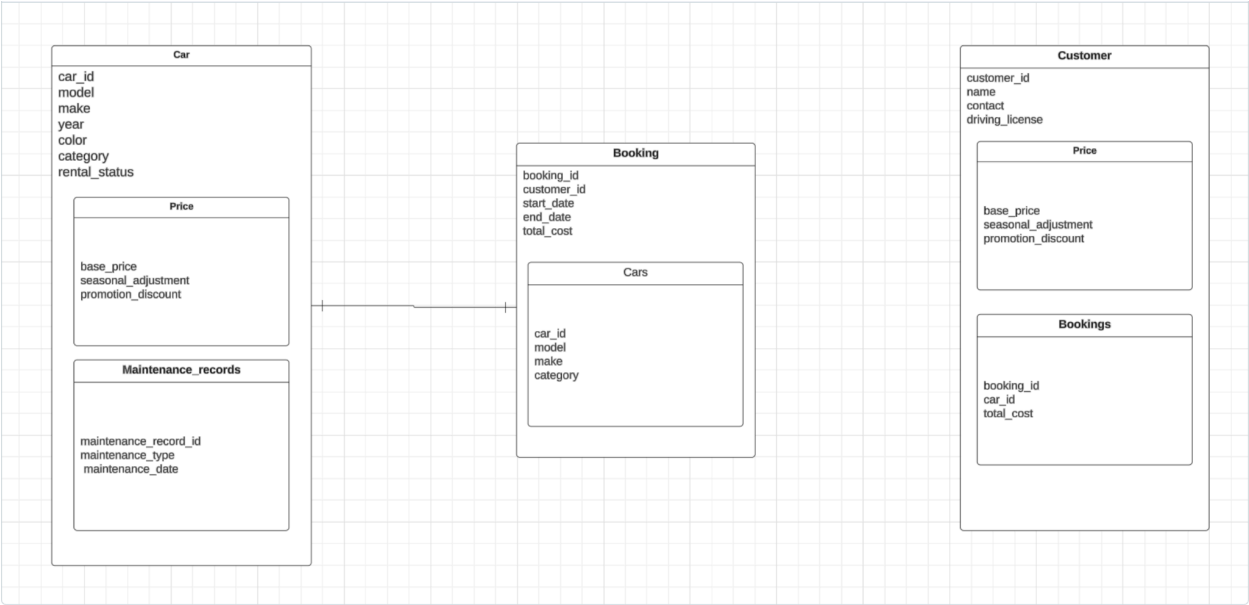
# 1. Car Rental Management System - Requirements Document

Renting a car can be one of the most frustrating parts of planning a trip. Rates are rarely standardized, meaning one company could offer a significantly different price than the brand down the street. The Car Rental Management System (CRMS) aims to automate the rental operations by managing the availability of vehicles, customer information, bookings, and vehicle maintenance. It helps ensure smooth operations through dynamic pricing based on car categories and rental duration while tracking maintenance to keep the fleet in optimal condition. The system will support multiple entities like cars, customers, bookings, and maintenance records and allow managers to generate reports to monitor operations.

# 2. Business Rules

1. Car Management
    - Each car has a unique identifier, make, model, year, category (e.g., SUV, Economy), and rental status (e.g., available, rented, maintenance).
    - A car cannot be rented when under maintenance.
    - Cars are categorized to apply distinct pricing models.
2. Customer Management
    - Customers provide their name, contact details, and license number during registration.
    - Each customer has a unique ID.
    - A customer can rent multiple cars, but each booking is tracked separately.
3. Booking Management
    - Each booking captures car ID, customer ID, rental start and end dates, and the total rental cost.
    - A car cannot be booked if it is already rented or under maintenance.
    - Bookings can be canceled, making the car available again.
4. Car Maintenance
    - Maintenance records include car ID, date, and type of service (e.g., oil change, repairs).
    - Cars under maintenance cannot be booked.
5. Dynamic Pricing
    - Rental cost is based on car category, rental duration, and seasonal demand.
    - Discounts may apply based on promotions or long-term bookings.
6. Reports and Queries
    - Reports on available cars, customer details, booking history, and maintenance status are accessible.
    - The system tracks customers with active bookings or outstanding payments.

# 3. UML Diagram




# 4. Functionalities Leveraging In-Memory Key-Value Storage

1. Most Frequently Booked Cars:
    - Track car bookings using a key-value pair: {car_id: booking_count}.
    - Useful for identifying popular cars and targeting promotions.
2. Active Sessions for Logged-In Users:
    - Store session tokens with associated customer IDs: {session_id: customer_id}.
    - Ensure quick authentication and session validation.
3. Active Bookings:
    - Maintain a key-value map of currently active bookings: {booking_id: car_id}.
    - Allows quick lookup for status updates or cancellations.
4. Dynamic Pricing Cache:
    - Cache adjusted prices for cars based on season and promotions: {car_id: adjusted_price}.
    - Speeds up booking cost calculations.

# 5. Redis Databases

### 1. Most Frequently Booked Cars

- Redis Data Structure: Sorted Set (ZSET)
- Key: mostBookedCars
- Booking count (incremented by 1 for each booking)

### 2. Active Sessions for Logged-In Users

- Redis Data Structure: Hash (HASH)
- Key: activeSessions:session_id
- Quickly validate user sessions and fetch associated customer IDs.

# 6. Redis Command

### 1. Most Frequently Booked Cars (Sorted Set: mostBookedCars)

- Create / Update
  - Increment booking count for a car:
  - ZINCRBY mostBookedCars 1 <car_id>
- Read
  - Retrieve the top N most frequently booked cars (highest scores first):
  - ZREVRANGE mostBookedCars 0 <N-1> WITHSCORES
  - Get the booking count for a specific car:
  - ZSCORE mostBookedCars <car_id>
- Delete
  - Remove a car from the most booked list:
  - ZREM mostBookedCars <car_id>

## 2. Active Sessions for Logged-In Users (Hash: activeSessions:session_id)

- Create / Update
  - Add or update a user session:
  - 

- Read
  - Retrieve a customer ID for a session:
  - 

  - Retrieve all active sessions:
  - 

- Delete
  - Remove a session (e.g., on logout):
  -