Ryerson University

Department of Electrical, Computer, & Biomedical Engineering
Faculty of Engineering & Architectural Science

Course Number	COE692
Course Title	Software Design and Architecture
Semester/Year	Winter Semester 2024
Instructor	Dr. Faezeh Ensan
TA Name	Bita

Lab/Tutorial Report No.	Lab 3

Report Title	Car Rental Management System

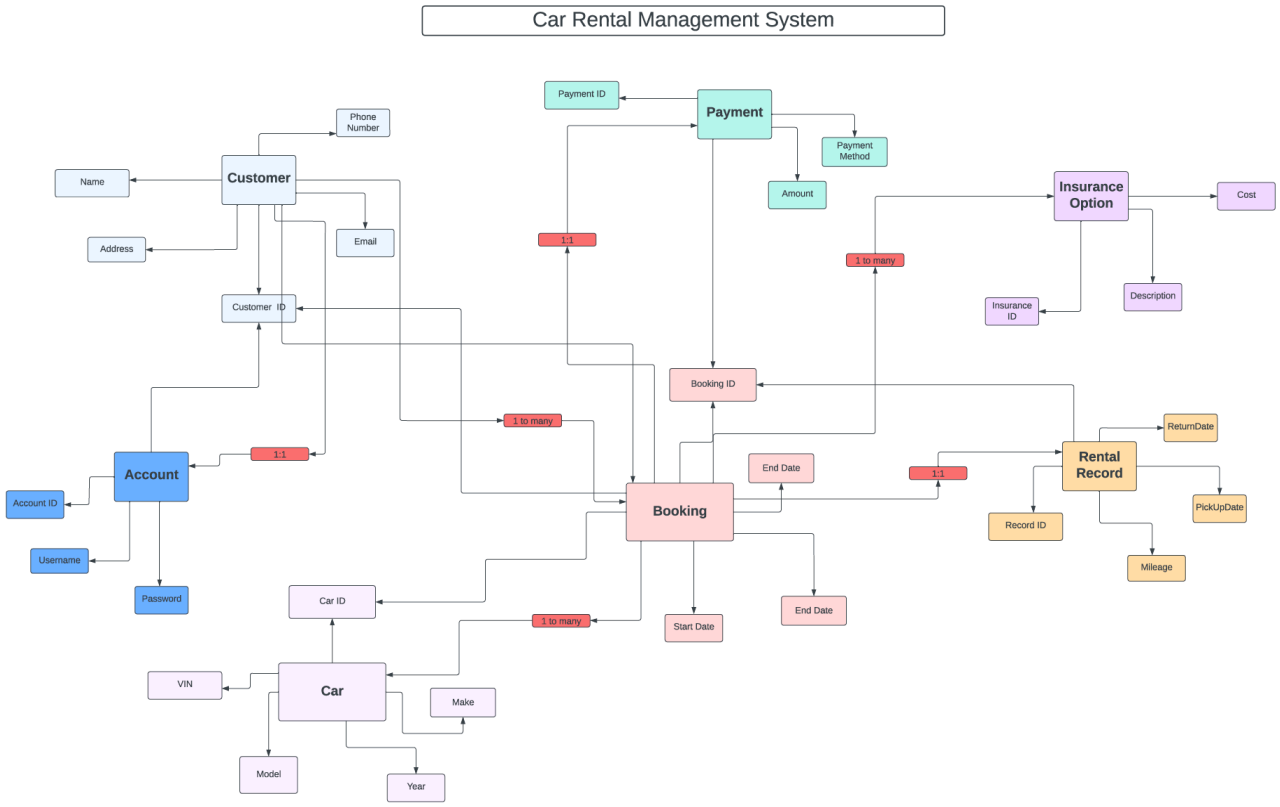
Section No.	07 & 02
Submission Date	
Due Date	

Student Name	Student ID	Signature*
Hamza Malik	501112545	HM
Kevin Bhatt	501093104	KM

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: http://www.ryerson.ca/content/dam/senate/policies/pol60.pdf*

Part 1: ER Diagram and SQL Tables

Car Rental Management System ER Diagram:



Description:

The Car Rental Management System's ER diagram is a structured representation of the data model that underpins the functionality of the entire rental service. It comprises seven primary entities, each with unique attributes and roles within the system. The entities are Customer, Account, Car, Booking, Payment, Insurance Option, and Rental Record. The ER diagram meticulously defines how the Car Rental Management System stores, processes, and interrelates data, ensuring a cohesive and efficient operational flow from customer engagement to the final return of the vehicle, all within a secured and user-centric framework.

Customer Entity:
The Customer (user) entity is central to the system, representing individuals who utilize the car rental service. It captures essential information with attributes such as Name, Address, Phone Number, and Email, uniquely identifying each customer by a Customer ID. This entity is foundational, as it ties directly to transactions, bookings, and account information, embodying the customer-centric nature of the system.

Account Entity:

Linked to the Customer entity is the Account entity through a one-to-one relationship, signifying that each customer has one account associated with their profile. The Account stores login credentials, comprised of a Username and Password, and is uniquely identified by the Account ID. This entity reflects the system's security measures and enables personalized access to the service.

Car Entity:
The Car entity represents the fleet of vehicles available for rent, detailed by attributes like VIN, Make, Model, and Year. Each car is identified by a Car ID, and this entity exhibits a one-to-many relationship with the Booking entity, indicating that a car can be associated with numerous bookings over different periods.

Booking Entity:
The Booking entity is the transactional core of the system, where rental agreements are recorded. It holds the Start and End dates of the rental and is identified by a Booking ID. Each booking is linked to one customer and one car, reflecting a many-to-one relationship in both cases, illustrating that a customer can have multiple bookings, and each car can be rented out multiple times.

Payment Entity:
Financial aspects are handled by the Payment entity, which is connected to Booking by a one-to-one relationship, denoting that each booking has a corresponding payment. This entity tracks the Payment Method and Amount, with a Payment ID serving as a unique identifier.

Insurance Option Entity:
To manage risk, the Insurance Option entity lists available insurance plans with a Cost and Description, linked to bookings by a one-to-many relationship. This setup allows customers to select multiple insurance options for each booking.

Rental Record Entity:
Lastly, the Rental Record entity documents the specifics of a car's rental usage, capturing PickUpDate, ReturnDate, and Mileage. Each record is uniquely identified by a Record ID and is associated with one booking, showing a one-to-one relationship.

SQL Code:

```
-- Create Customer Table
CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(255),
    Address VARCHAR(255),
    Phone_Number VARCHAR(20),
    Email VARCHAR(100)
);

-- Create Account Table
CREATE TABLE Account (
    AccountID INT PRIMARY KEY,
    Username VARCHAR(255),
    Password VARCHAR(255),
    CustomerID INT,
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);

-- Create Car Table
CREATE TABLE Car (
    CarID INT PRIMARY KEY,
    VIN VARCHAR(17),
    Make VARCHAR(50),
    Model VARCHAR(50),
    Year YEAR
);

-- Create Booking Table
CREATE TABLE Booking (
    BookingID INT PRIMARY KEY,
    CustomerID INT,
    CarID INT,
    StartDate DATE,
    EndDate DATE,
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID),
    FOREIGN KEY (CarID) REFERENCES Car(CarID)
);

-- Create Payment Table
CREATE TABLE Payment (
    PaymentID INT PRIMARY KEY,
    BookingID INT,
```

```
    PaymentMethod VARCHAR(50),
    Amount DECIMAL(10, 2),
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID)
);

-- Create InsuranceOption Table
CREATE TABLE InsuranceOption (
    InsuranceID INT PRIMARY KEY,
    Description TEXT,
    Cost DECIMAL(10, 2)
);

-- Create RentalRecord Table
CREATE TABLE RentalRecord (
    RecordID INT PRIMARY KEY,
    BookingID INT,
    PickUpDate DATE,
    ReturnDate DATE,
    Mileage INT,
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID)
);

-- Note: The table for InsuranceOptions does not link directly to Bookings in the provided ER diagram.
-- If there's a many-to-many relationship (which is common), you would need an associative table to handle this, like so:

-- Create BookingInsurance Table (Associative table for Booking and InsuranceOption)
CREATE TABLE BookingInsurance (
    BookingID INT,
    InsuranceID INT,
    PRIMARY KEY (BookingID, InsuranceID),
    FOREIGN KEY (BookingID) REFERENCES Booking(BookingID),
    FOREIGN KEY (InsuranceID) REFERENCES InsuranceOption(InsuranceID)
);
```

Figure 4: All SQL Tables Within The DataBase