**2.1 System Architecture**

The car rental system allows a user to create an account for user to log in and select a car to rent. The user can choose when to rent and when to stop renting and confirm it and afterwards it takes user to make a payment.

**2.2 Class Diagram**

**Main Classes**

1. User

* Attributes: userId, name, email, password, role
* Methods: register(), login(), logout(), pay()

1. Car

* Attributes: carId, model, modelYear, pricePerDay, availabilityStatus
* Methods: addCar(), updateCar(), checkAvailability(),

1. Renting

* Attributes: rentingId, userId, carId, startDate, endDate
* createBooking(), cancelBooking(), viewBooking()

1. Payment

* Attributes: paymentId, bookingId, amount, paymentDate, paymentStatus
* Methods: processPayment, viewPaymentStatus()

**Subclasses**

1. User

* Customer: Customer inherits from User and has their own social security attribute.
* Admin: Inherits from User and has their own workerId and has confirmBooking() method.

1. Car

* EconomyCar: EconomyCar inherits from Car class
* LuxuryCar: LuxuryCar inherits from Car class

**2.3 Database Schema**

**Tables**

1. Users

* user\_id
* name
* email
* password
* role (“Admin” , “Customer”)

1. Cars

* car\_id
* model
* model\_year
* price\_per\_day
* availability\_status

1. Renting

* renting\_id
* user\_id
* car\_id
* start\_date
* end\_date
* total\_price

1. Payments

* payment\_id
* renting\_id
* amount
* payment\_date
* payment\_status

1. **Features and Functionality**
   1. **Key Features**

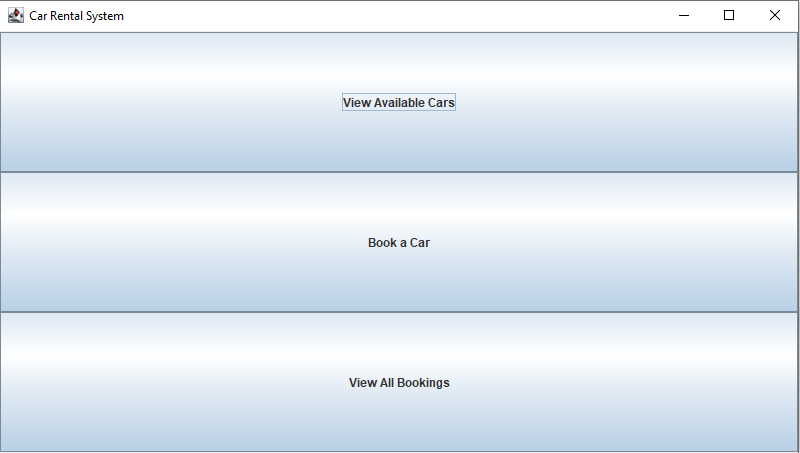
**1.** **Adding, Viewing, and Booking Cars**:

* View available cars.
* Book a car with customer details and rental duration.
* Mark cars as unavailable once booked.

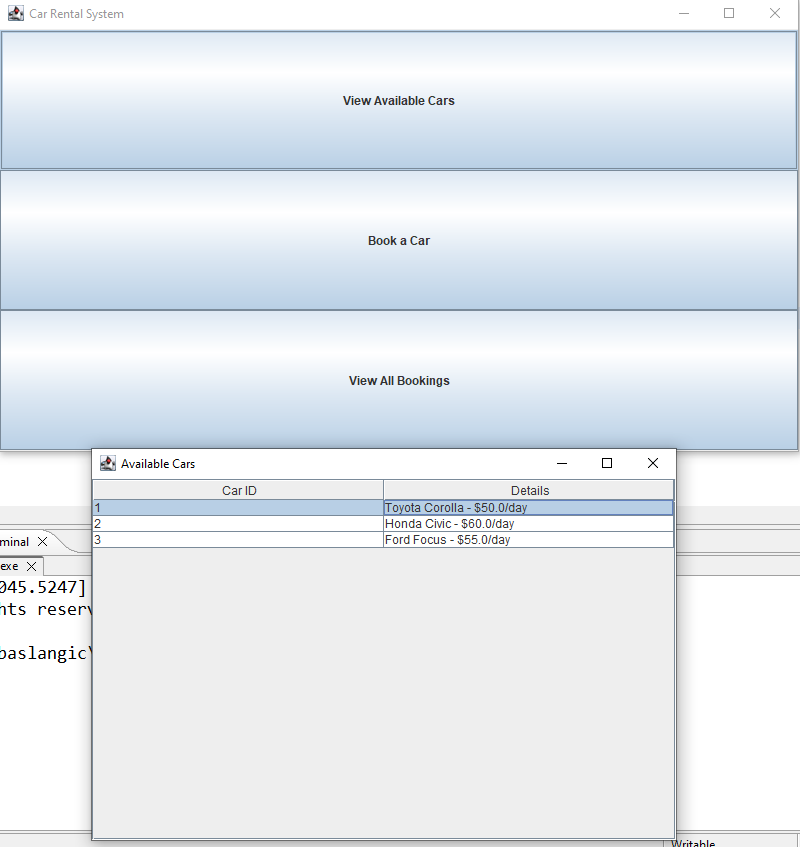
**2.** **Managing Bookings**:

* View all bookings.
* Display detailed information about each booking (car, customer, rental days, total price).
  1. **Custimization**
* A specific custimization is special handling for car tiers like luxury, economy, etc..

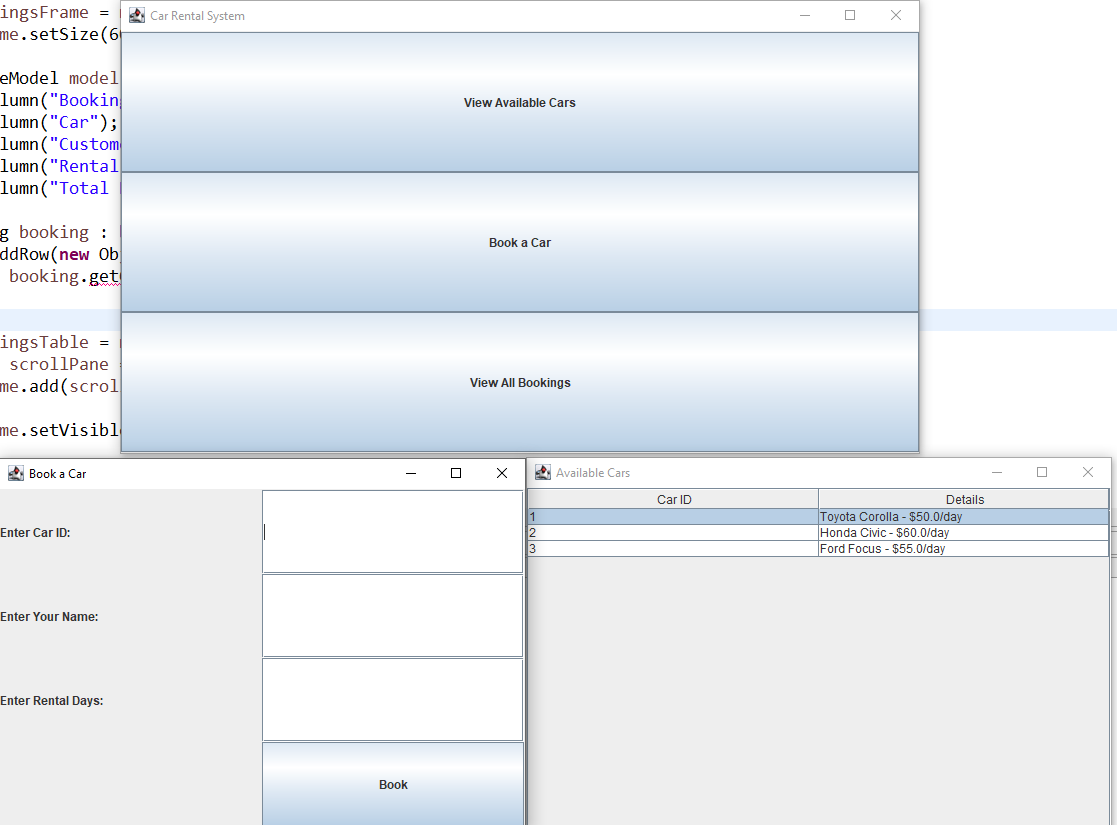
1. **Application Walkthrough**
   1. **GUI Overview**



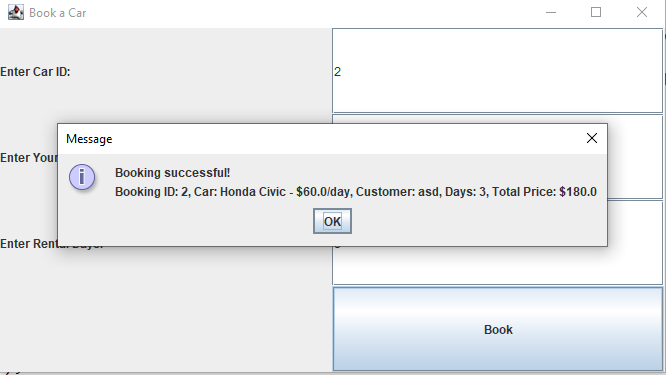
Şekil Main Menu



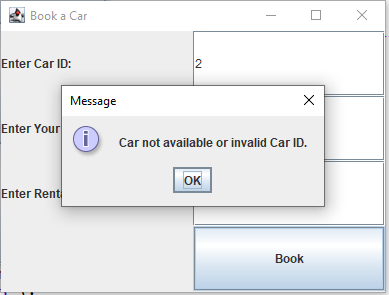
Şekil 2 When Pressed View Available Cars



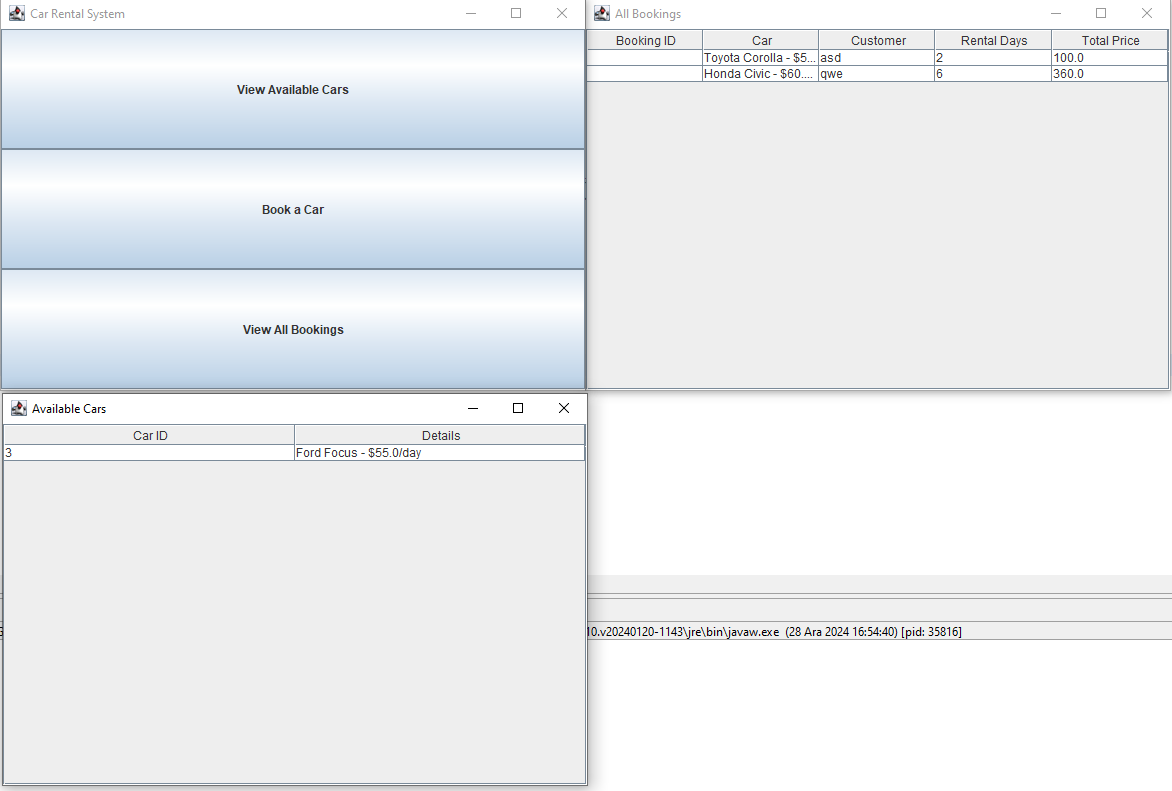
Şekil 3 After pressing Book a Car

****

Şekil 4 After booking



Şekil 5 If its already rented or invalid car ID



Şekil 6 View All Bookings

**5. Object-Oriented Principles**

**5.1. Use of Classes and Objects**

**How Classes and Objects Are Used**

1. **Classes as Blueprints:**
   * The system uses classes like Car, Booking, CarService, and BookingService as blueprints for objects.
   * Each class encapsulates data and behavior relevant to its role in the system.
2. **Examples:**
   * Car represents a single car in the inventory, encapsulating attributes like carId, model, and availability.
   * Booking represents a rental transaction, encapsulating attributes like bookingId, customerName, and the rented Car.
3. **Objects for Data Management:**
   * When a car is added or booked, an object of the Car class is created to represent it in memory.
   * Similarly, each booking is represented by a Booking object.