Vehicle Rental System - System Design Document

1. Introduction

This document provides a detailed system design for the Vehicle Rental System, outlining its architecture, database design, and component interactions.

2. System Architecture

2.1 Architectural Pattern

The system follows the **Model-View-Controller (MVC)** architecture:

- **Model:** Represents the business logic and database interaction.
- **View:** Handles the user interface with Java Swing.
- **Controller:** Manages user inputs and updates the Model and View.

2.2 High-Level Architecture

The system is divided into the following layers:

1. **Presentation Layer (UI):** Java Swing Forms
2. **Service Layer:** Business logic implementation
3. **Data Access Layer (DAO):** Database interactions via JDBC
4. **Database Layer:** Oracle DB for data storage

3. Database Design

The system uses Oracle Database with the following tables:

3.1 Tables and Schema

users

Column Name	Data Type	Constraints
user_id	INT	PRIMARY KEY, AUTO_INCREMENT
name	VARCHAR(255)	NOT NULL
email	VARCHAR(255)	UNIQUE, NOT NULL
password	VARCHAR(255)	NOT NULL
role	VARCHAR(50)	CHECK ('Admin', 'User')

vehicles

Column Name	Data Type	Constraints
vehicle_id	INT	PRIMARY KEY, AUTO_INCREMENT
make	VARCHAR(100)	NOT NULL
model	VARCHAR(100)	NOT NULL
year	INT	CHECK (year > 2000)
price_per_day	DECIMAL(10,2)	NOT NULL

rentals

Column Name	Data Type	Constraints
rental_id	INT	PRIMARY KEY, AUTO_INCREMENT
user_id	INT	FOREIGN KEY (users)
vehicle_id	INT	FOREIGN KEY (vehicles)
start_date	DATE	NOT NULL
end_date	DATE	NOT NULL
total_price	DECIMAL(10,2)	NOT NULL

4. Component Interaction

4.1 Flow of Rental Booking

1. **User selects a vehicle** from the UI.
2. The **controller processes** the request and invokes VehicleService.
3. VehicleService calls VehicleDAO to check availability.
4. If available, RentalService is triggered to process booking.
5. RentalDAO updates the rentals table.
6. Confirmation is sent back to the UI.

4.2 Payment Processing

1. User selects a payment method.
2. PaymentService processes the transaction.
3. Payment details are stored in payments table.
4. Rental is confirmed.

5. Technologies Used

- **Frontend:** Java Swing
- **Backend:** Java (JDBC, DAO, Service, Controller)
- **Database:** Oracle 21c XE
- **Build Tool:** Gradle
- **Version Control:** GitHub

6. Conclusion

This document outlines the structure and workflow of the Vehicle Rental System, ensuring a scalable and maintainable design.