# Project Title: CarConnect – A Car Rental Platform

## 1. Purpose of the Project

The purpose of CarConnect is to develop a comprehensive, secure, and user-friendly car rental platform using Python and SQL. The system will allow customers to register, authenticate, search, and reserve vehicles in real-time, while admins can manage vehicles, customers, and view reports. The application will follow object-oriented programming principles, include exception handling, and enable seamless database interactions.

## 2. Scope of the Project

CarConnect will:  
- Enable customers to register, log in, update details, view and reserve vehicles.  
- Allow admins to manage fleet, reservations, customers, and view reports.  
- Provide a real-time reservation system with conflict resolution and notifications.  
- Be implemented in Python with a SQL backend.  
- Use object-oriented programming and follow a modular architecture.  
- Include unit testing using Python’s unittest framework.

## 3. Functional Requirements

### 3.1 User Authentication

- Login with username and password.  
- Password authentication and role-based access.  
- Throw AuthenticationException for invalid login.

### 3.2 Customer Management

- Register, retrieve, update, and delete customer records.  
- Fields: CustomerID, FirstName, LastName, Email, PhoneNumber, Address, Username, Password, RegistrationDate.

### 3.3 Vehicle Management

- Add, update, delete, and fetch vehicle details.  
- Filter by availability.  
- Fields: VehicleID, Model, Make, Year, Color, RegistrationNumber, Availability, DailyRate.

### 3.4 Reservation Management

- Create, update, cancel reservations.  
- Calculate total cost and resolve conflicts.  
- Fields: ReservationID, CustomerID, VehicleID, StartDate, EndDate, TotalCost, Status.

### 3.5 Admin Management

- Admin registration, login, update, and delete.  
- Role-based actions.  
- Fields: AdminID, FirstName, LastName, Email, PhoneNumber, Username, Password, Role, JoinDate.

### 3.6 Reporting

- Generate reports on reservations, utilization, and revenue.

## 4. Non-Functional Requirements

- Usability: CLI interface with user-friendly error handling.  
- Security: Hashed password storage and secure access.  
- Performance: Efficient queries for real-time processing.  
- Reliability: Robust exception handling.  
- Maintainability: Modular code with clean structure.

## 5. Database Schema Overview

SQL Tables:  
1. Customer  
2. Vehicle  
3. Reservation  
4. Admin  
All tables include appropriate keys and constraints.

## 6. Exception Handling

- AuthenticationException  
- ReservationException  
- VehicleNotFoundException  
- AdminNotFoundException  
- InvalidInputException  
- DatabaseConnectionException

## 7. Tools and Technologies

- Language: Python  
- Database: SQL (PostgreSQL / SQLite)  
- IDE: VSCode / PyCharm  
- Version Control: GitHub  
- Testing: unit test module in Python

## 8. Testing Requirements

- Test login with invalid credentials  
- Test customer update  
- Test vehicle addition and updates  
- Test availability queries  
- Test reservation creation and cancellation