**CS 255 System Design Document Template**

This template lays out all the different sections that you need to complete for Project Two. Each section has guidance to prompt your thinking. You will need to continually reference the interview transcript as you work to make sure that you are addressing your client’s needs. There is no required length for the final document. Instead the goal is to complete each section based on what your client’s needs are. Remove this note when you are finished, and replace all bracketed text with the relevant information.

**UML Diagrams**

**UML Use Case Diagram**

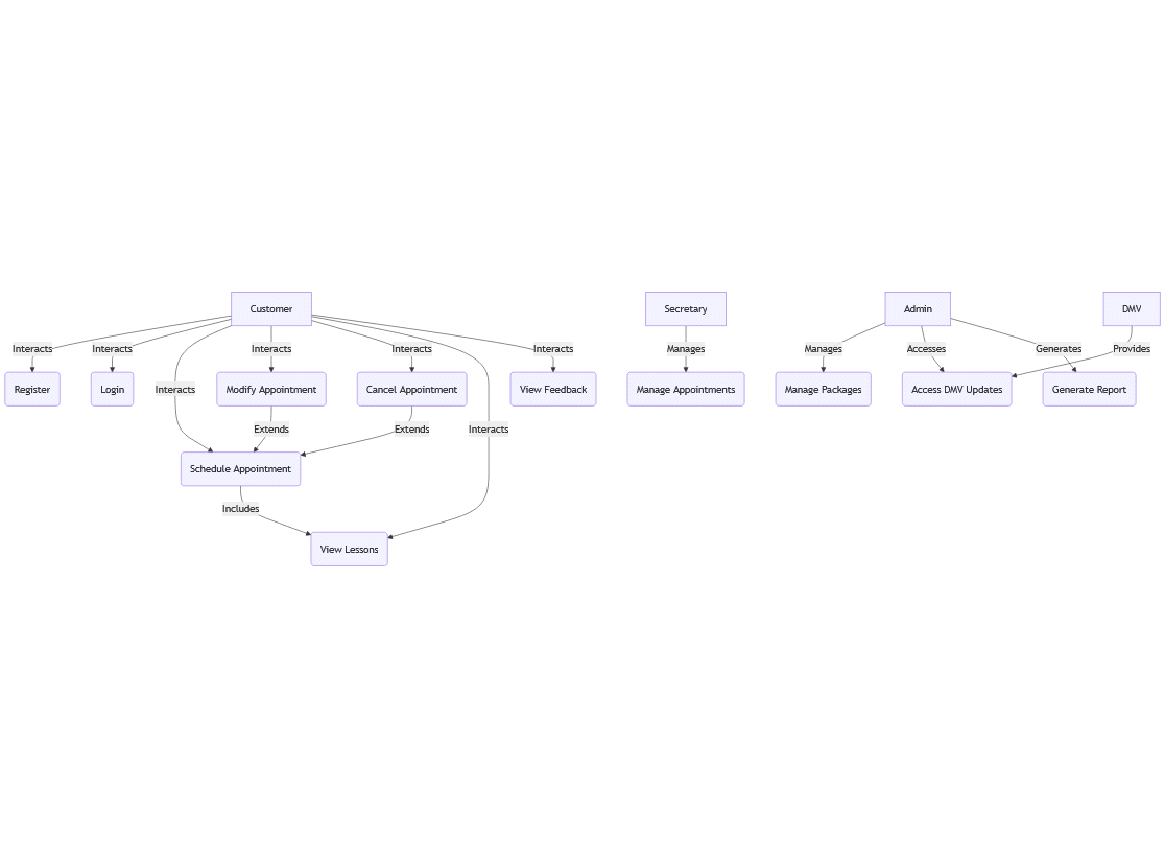
The Use Case Diagram below represents the key actors and use cases within the DriverPass system. The actors include:

Customer: The user who interacts with the system to schedule driving lessons, access lesson notes, and manage their profile.

Secretary: Manages appointments and oversees the scheduling process.

Admin: Manages system updates, lesson packages, and interacts with DMV updates.

DMV: A third-party actor that provides external data for driving tests and certifications.

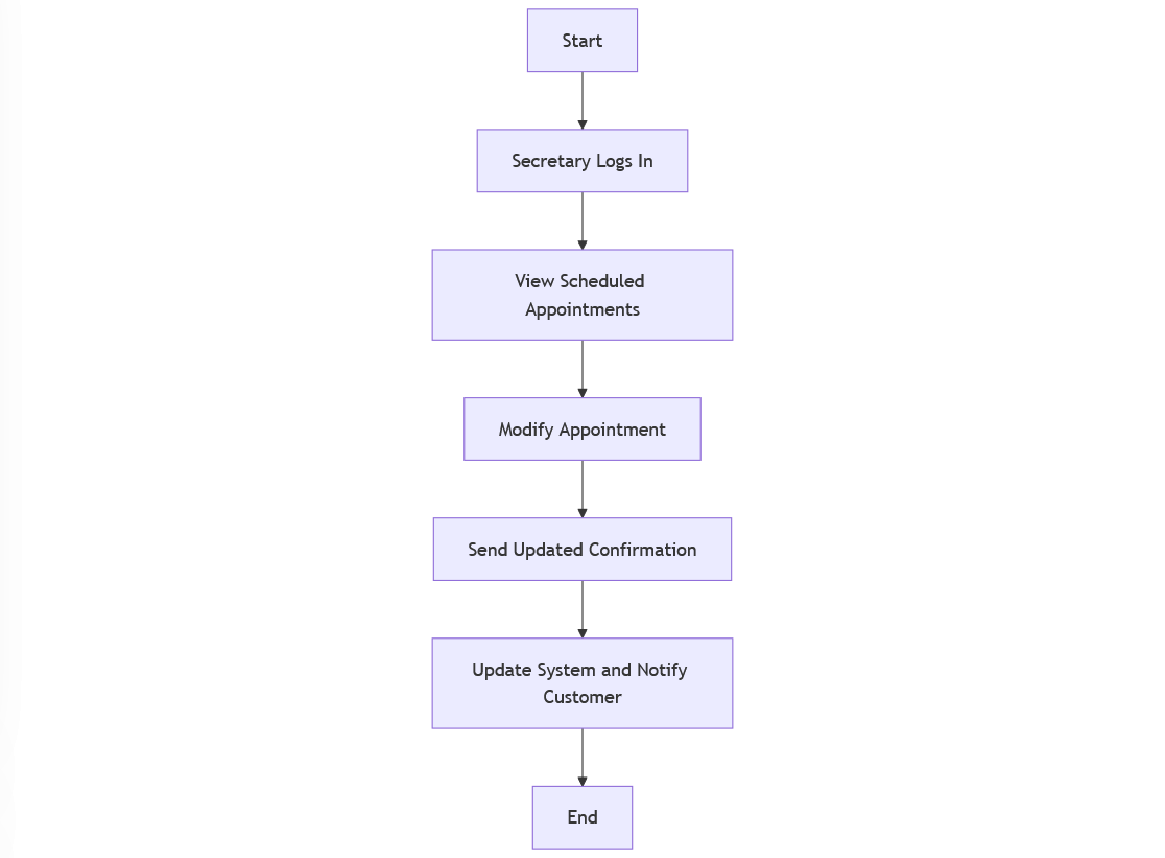


**UML Activity Diagrams**

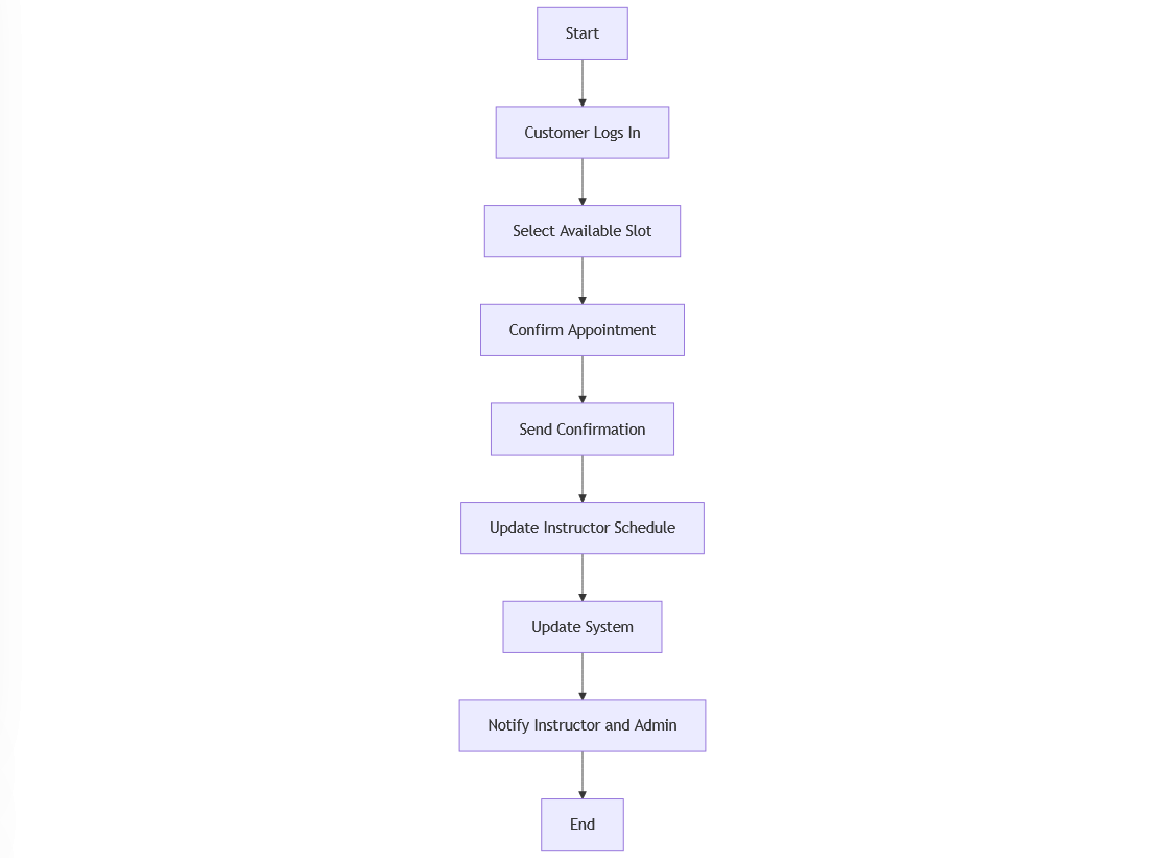
Two Activity Diagrams were created to illustrate the processes involved in two key use cases:

Schedule Appointment: This diagram outlines the steps involved in booking a driving lesson, from logging in to selecting a time slot, confirming the appointment, and receiving confirmation.

Manage Appointments: This diagram details how the Secretary manages customer appointments, including modifying and canceling bookings.

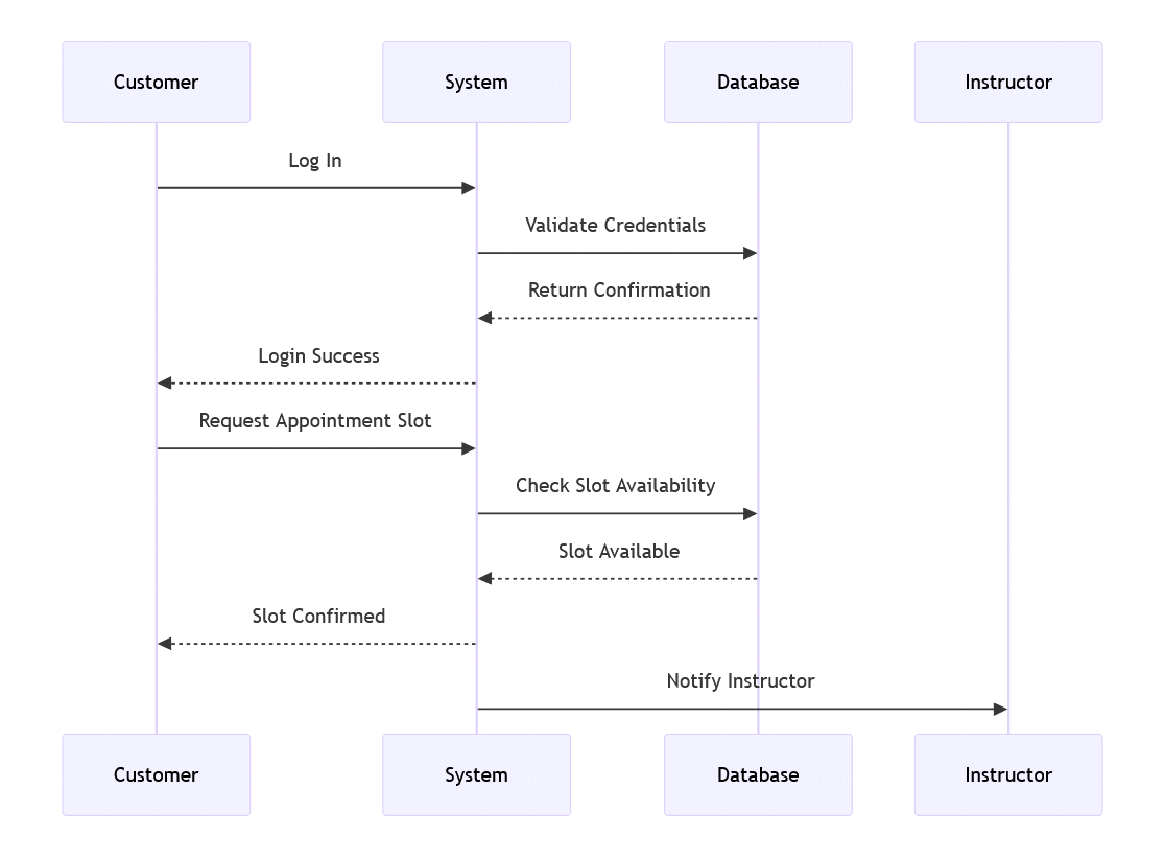


Schedule Appointment below:



**UML Sequence Diagram**

The Sequence Diagram below focuses on the Schedule Appointment use case. It shows the interaction between the Customer, System, and the Database, detailing the sequence of messages exchanged when a customer schedules a driving lesson.



**UML Class Diagram**

The Class Diagram for the DriverPass system includes the following key classes:

Customer: Attributes include name, email, password, testHistory.

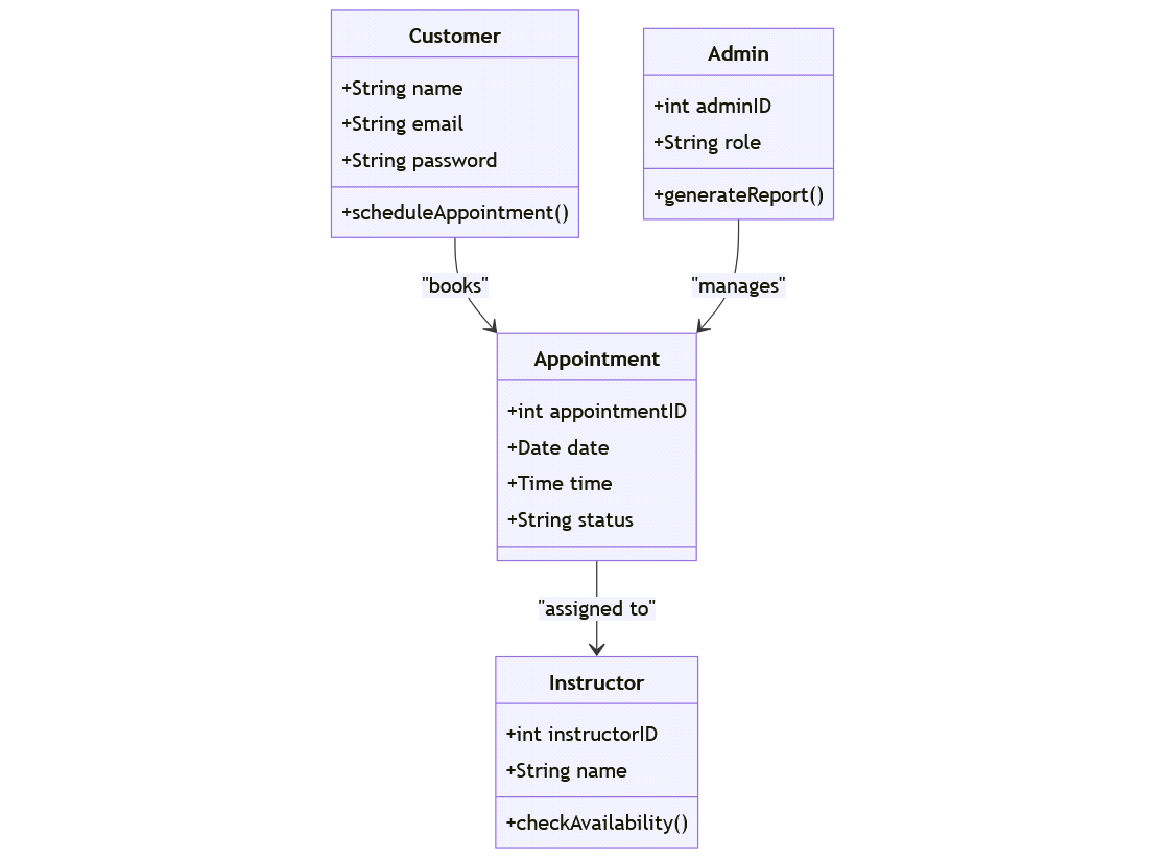
Appointment: Attributes include appointmentID, date, time, status, instructor.

Instructor: Attributes include instructorID, name, availability.

Admin: Attributes include adminID, role, permissions.

DMV: Provides updates on driving certifications and test results.

This diagram shows the relationships between these classes, highlighting how they interact within the system to manage lessons, users, and driving certifications.



**Technical Requirements**

Based on the system design and the diagrams created, the technical requirements for the DriverPass system are as follows:

Hardware:

Web servers to host the DriverPass application and manage real-time appointment scheduling.

Database servers for managing user data, lesson scheduling, and interaction with external DMV systems.

Software:

Relational Database: Use MySQL or PostgreSQL for managing user information, lesson schedules, and test results.

Web Framework: A back-end framework such as Django or Ruby on Rails to handle business logic and API integration.

Front-End: A responsive front-end using React or Angular for mobile and web access to the system.

Tools:

Version Control: Git for code management and collaboration.

Lucidchart for visual modeling and creating UML diagrams.

Integrated Development Environment (IDE): Use Visual Studio Code or PyCharm for coding.

Infrastructure:

Cloud Infrastructure: Use AWS or Azure for hosting the application, providing scalability and reliability.

Security: Secure communication using SSL/TLS for all user data exchanges. Implement two-factor authentication and password hashing for user logins.

Real-Time Scheduling: Use WebSocket technology to handle real-time appointment updates and notifications.