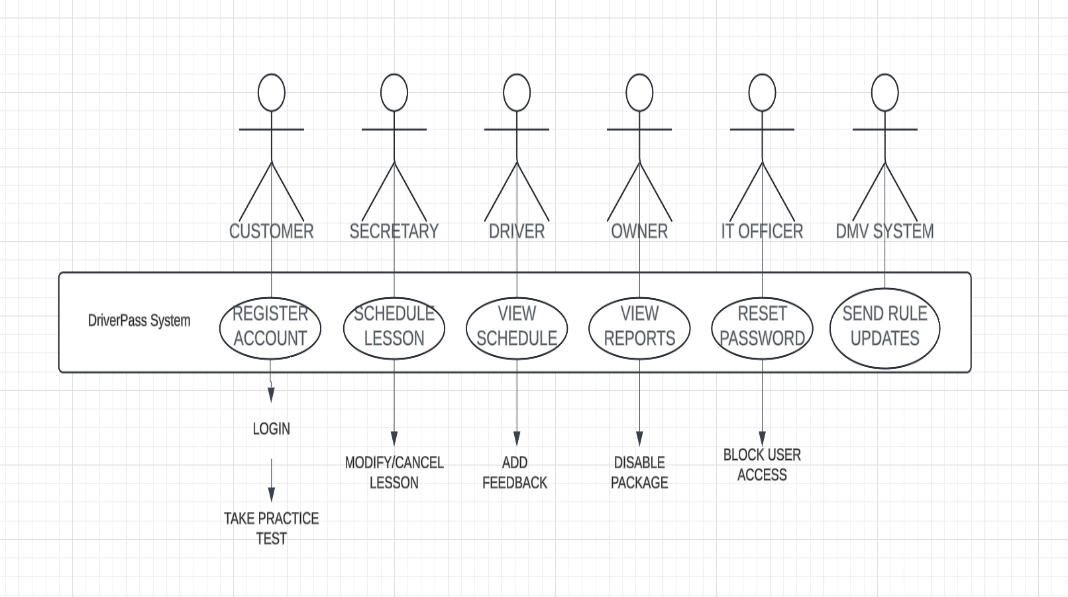
# CS 255 System Design Document Template

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.

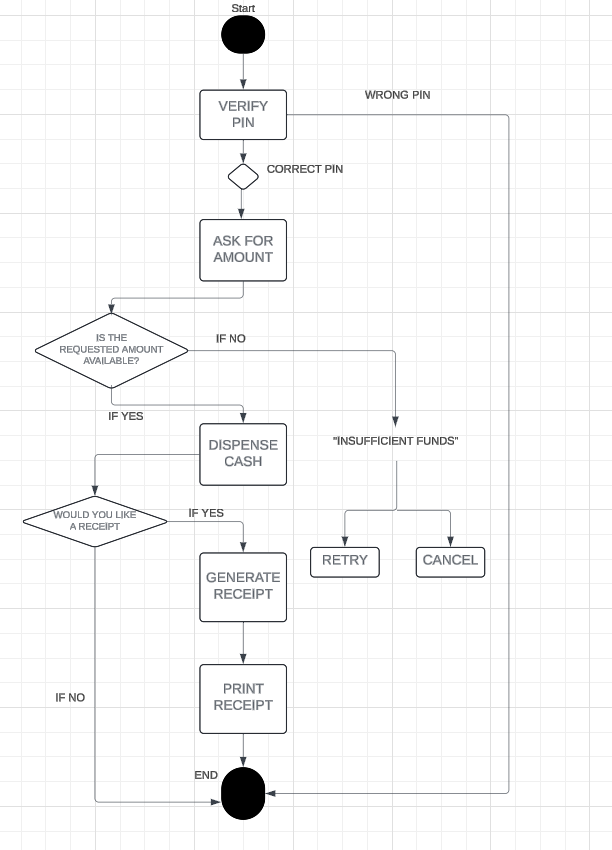
## UML Diagrams

### UML Use Case Diagram

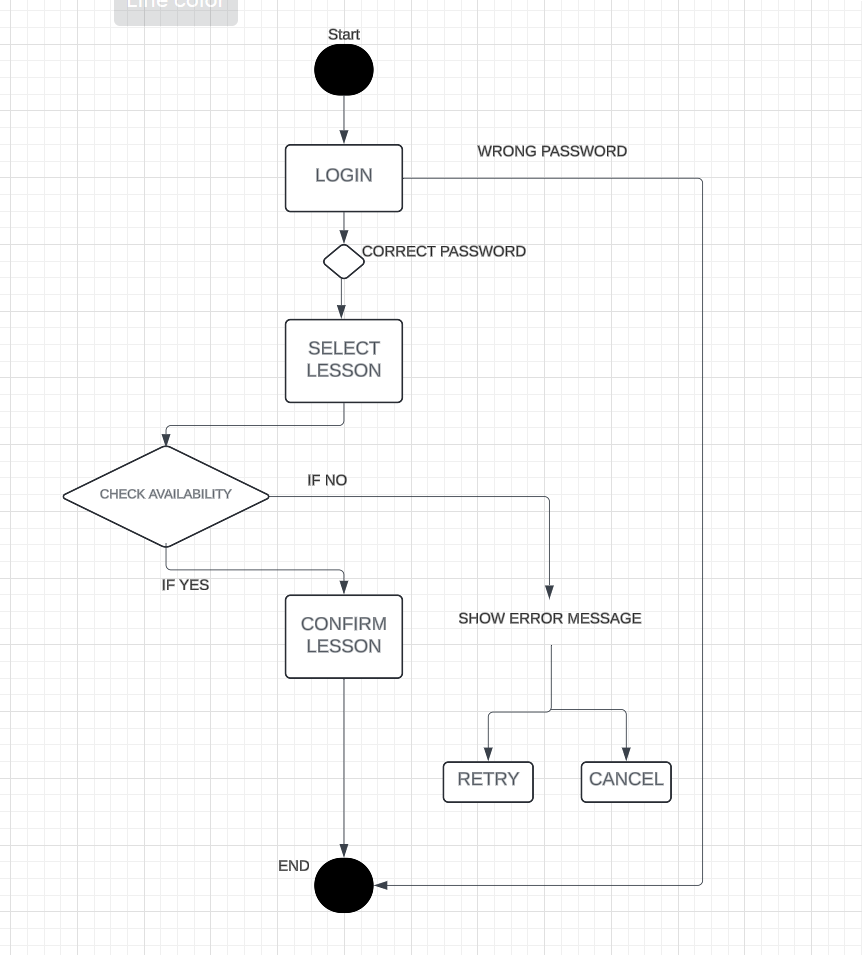
**

**Explanation**: This diagram shows the different users (actors) like Customers, Admins, and Secretaries, and their interactions with the DriverPass system. Each actor can perform specific actions, such as scheduling lessons or managing accounts.

### UML Activity Diagrams

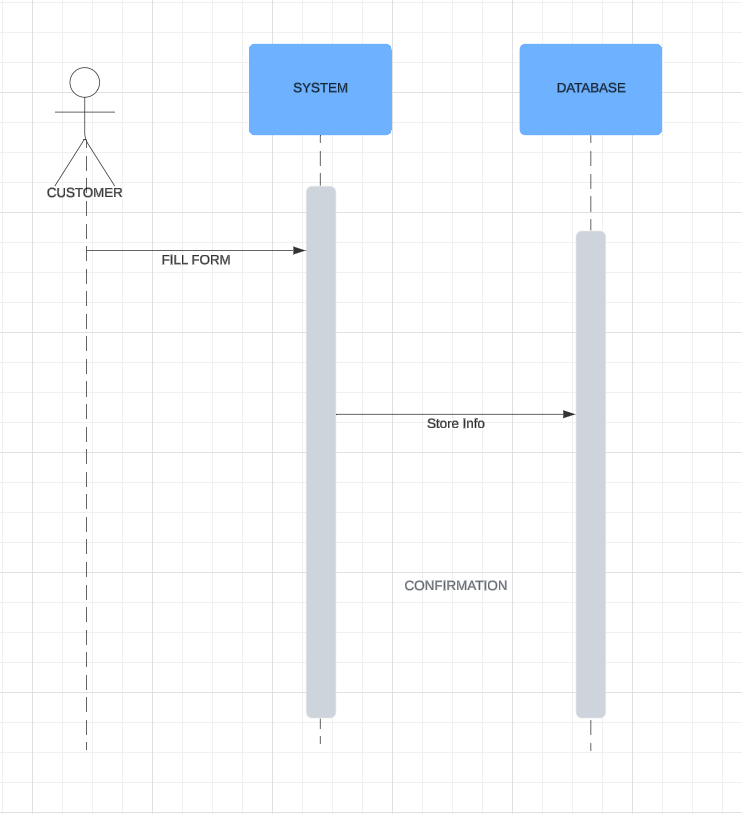
**

**Explanation**: This diagram represents the flow of an ATM transaction, including PIN verification, amount selection, cash dispensing, and receipt generation.



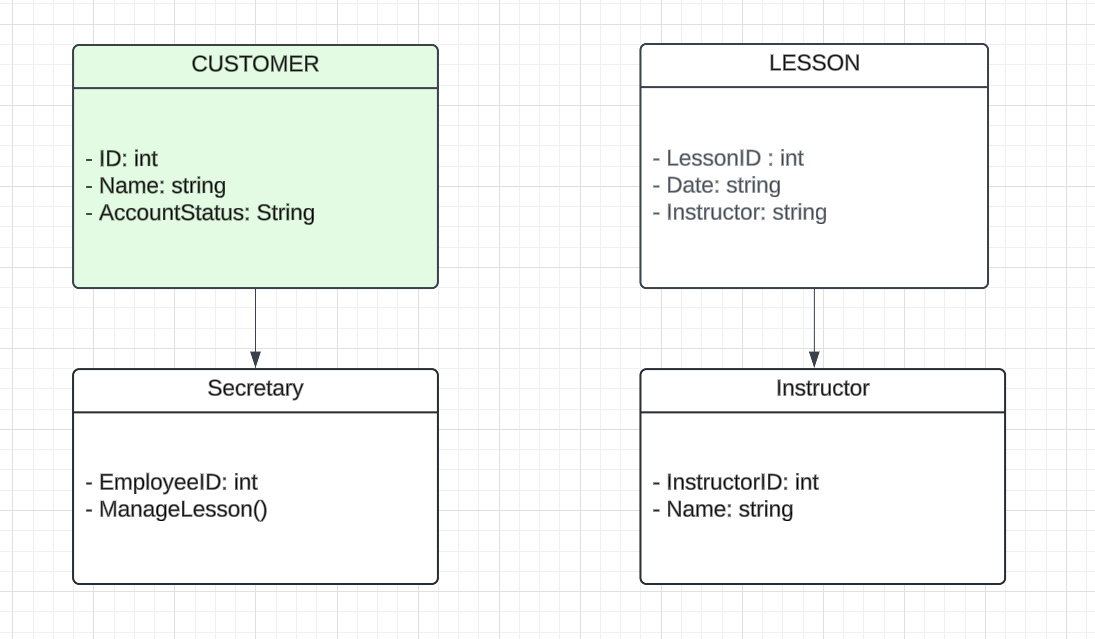
### Explanation: This diagram shows the steps for scheduling a lesson, from logging in to confirming availability and booking a slot.

### UML Sequence Diagram

**

**Explanation**: This diagram demonstrates the interaction between the Customer, System, and Database when registering an account. It tracks the process from submitting the form to storing the information and confirming the registration.

### UML Class Diagram

**

**Explanation**: This class diagram illustrates the relationships between key entities like Customers, Lessons, Secretaries, and Instructors. Each class includes attributes relevant to the DriverPass system, such as ID, name, and lesson details.

## Technical Requirements

*The DriverPass system requires a robust infrastructure to ensure smooth operation and scalability. For hardware, cloud-based servers are essential to store customer data and manage scheduling, while customers and staff will access the system via desktops and mobile devices. The system will run on a Linux or Windows server, utilizing MongoDB or MySQL for secure data storage. Development will be conducted using Python with the Flask framework for backend services, along with Visual Studio Code for coding and GitHub for version control. Cloud hosting through AWS or Azure will provide scalability, and HTTPS will be used for secure communication, with encrypted database connections to protect sensitive user information.*