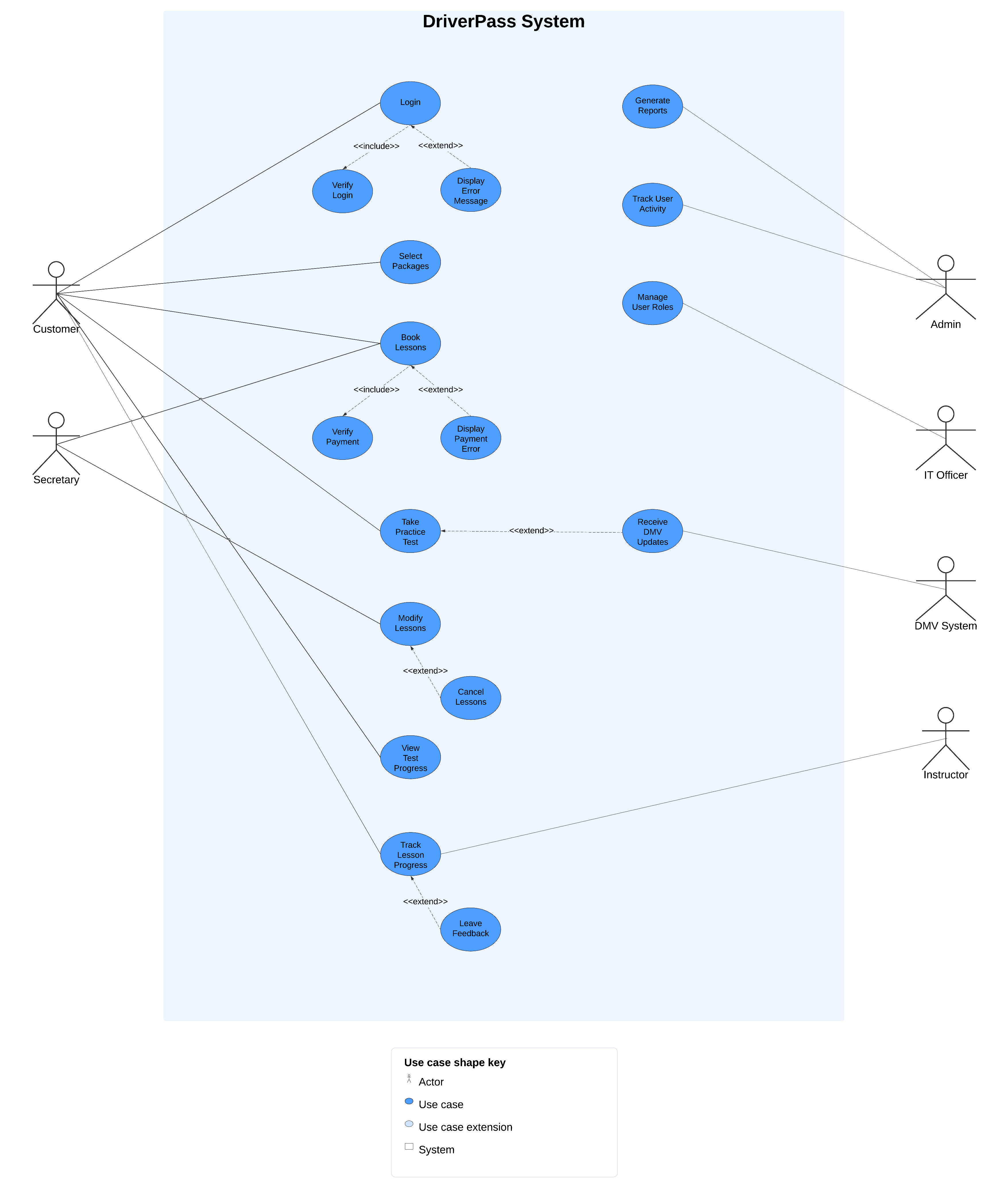
# CS 255 System Design Document Template

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

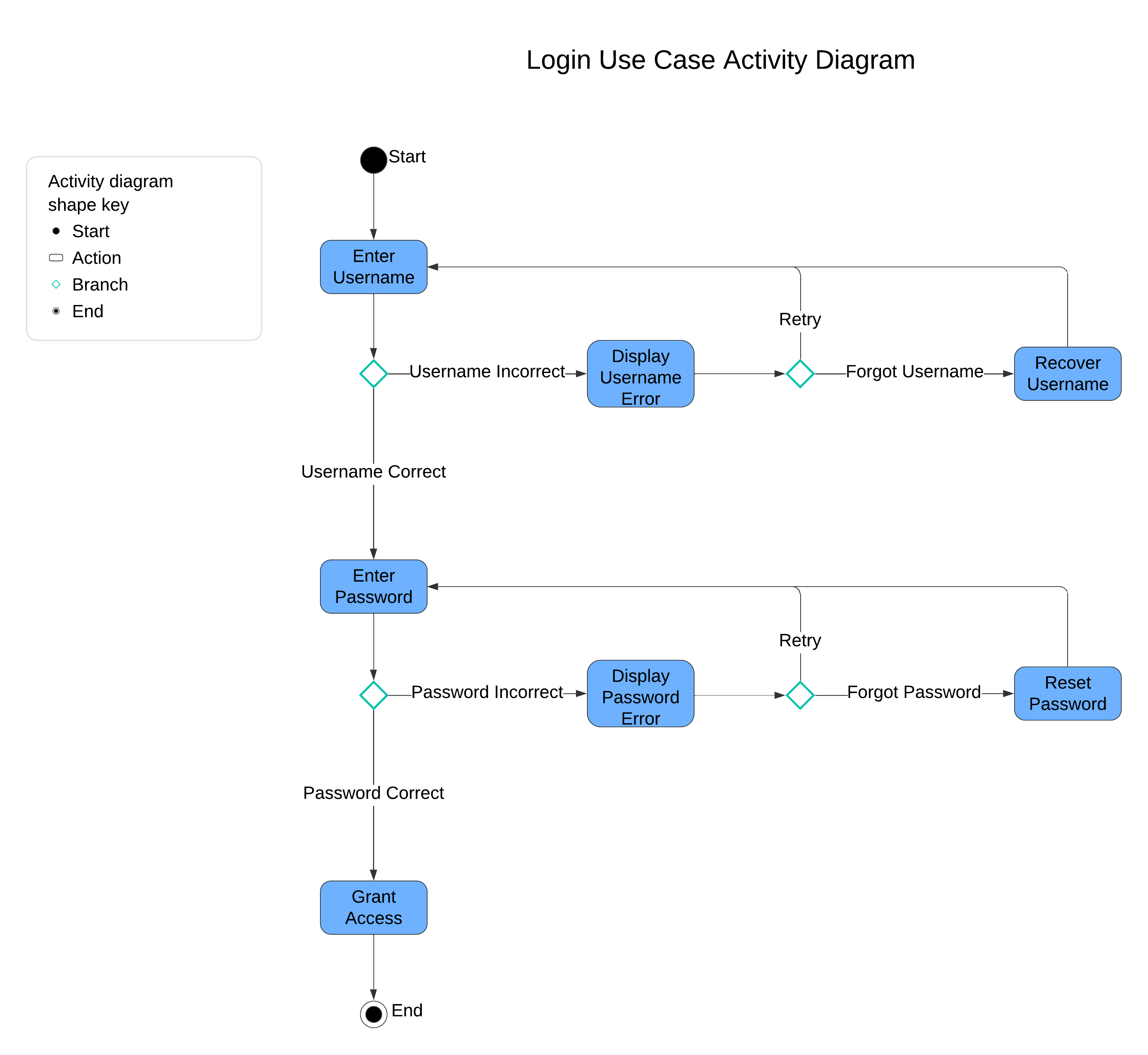
### UML Use Case Diagram

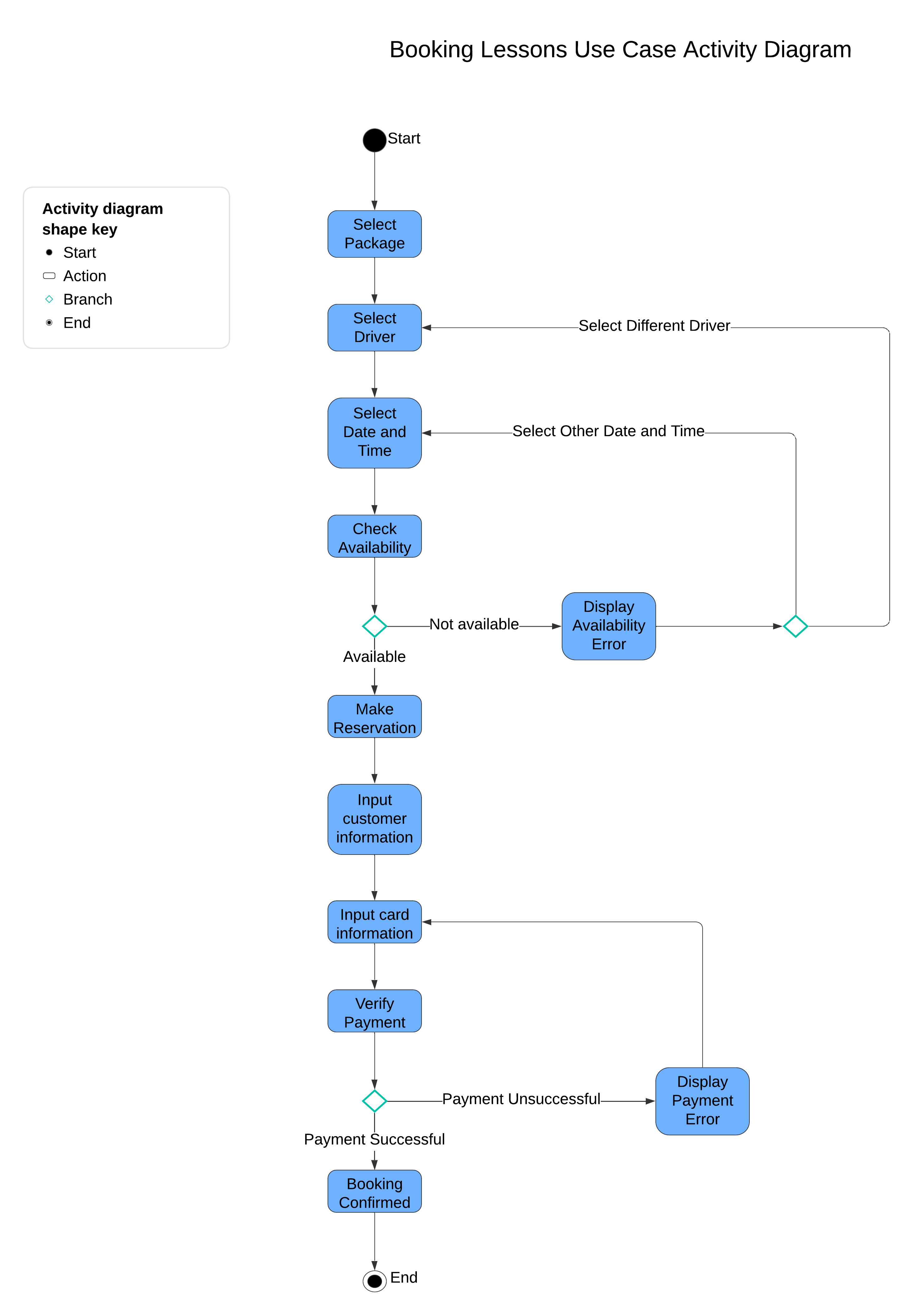
*[In Module Six, you were asked to complete a use case diagram based on your system design. If you would like to make any adjustments to your diagram, please do so. Please insert your use case diagram here. Check to make sure that you included appropriate components and symbols and that your design meets the client’s needs.]*



### UML Activity Diagrams

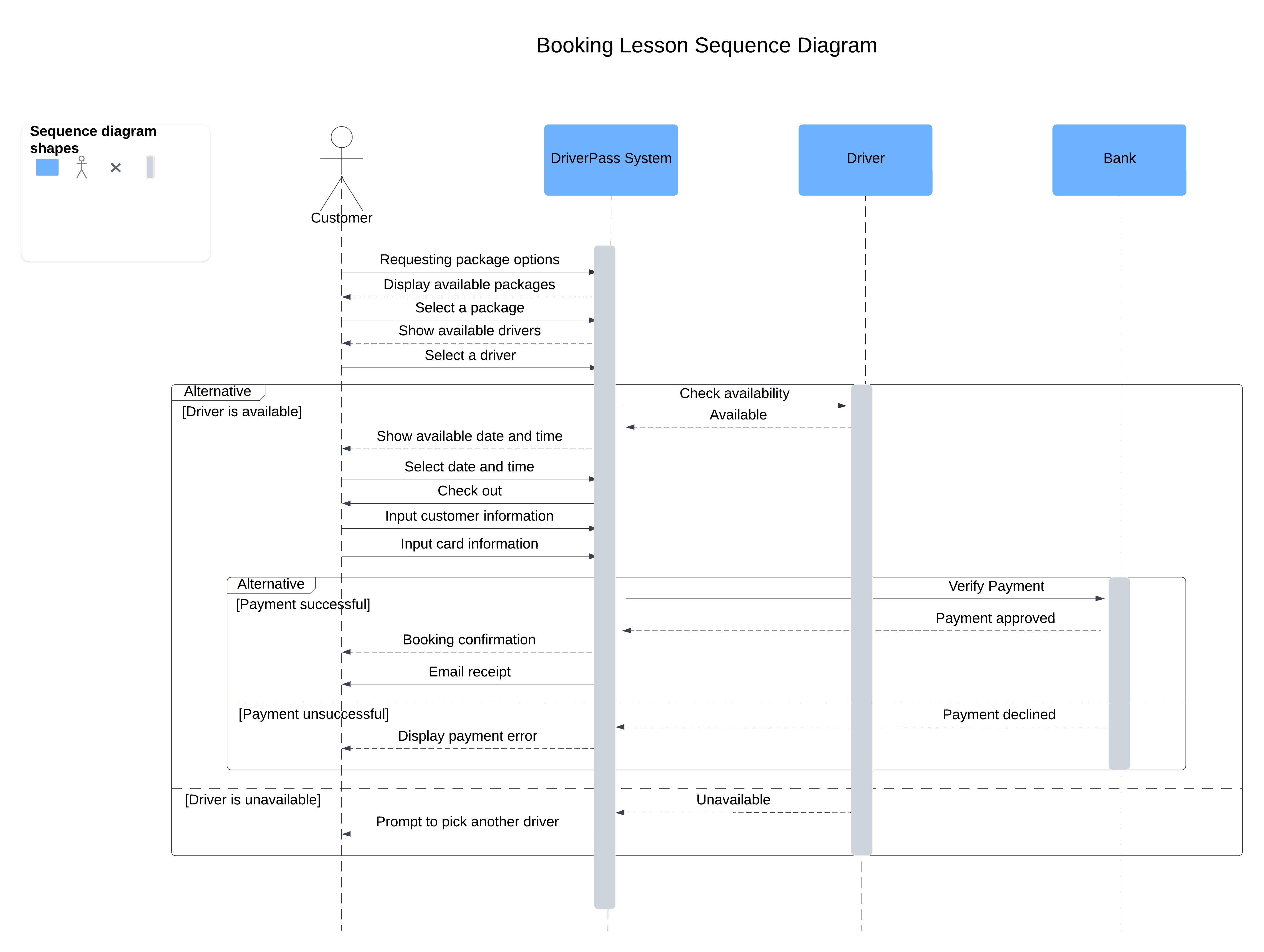
*[You were asked to choose* ***two*** *use cases and create* ***two*** *activity diagrams, one for each use case.* *Please insert* ***both*** *of your activity diagrams here. Check to make sure that you included appropriate components and symbols and that your design meets the client’s needs.]*





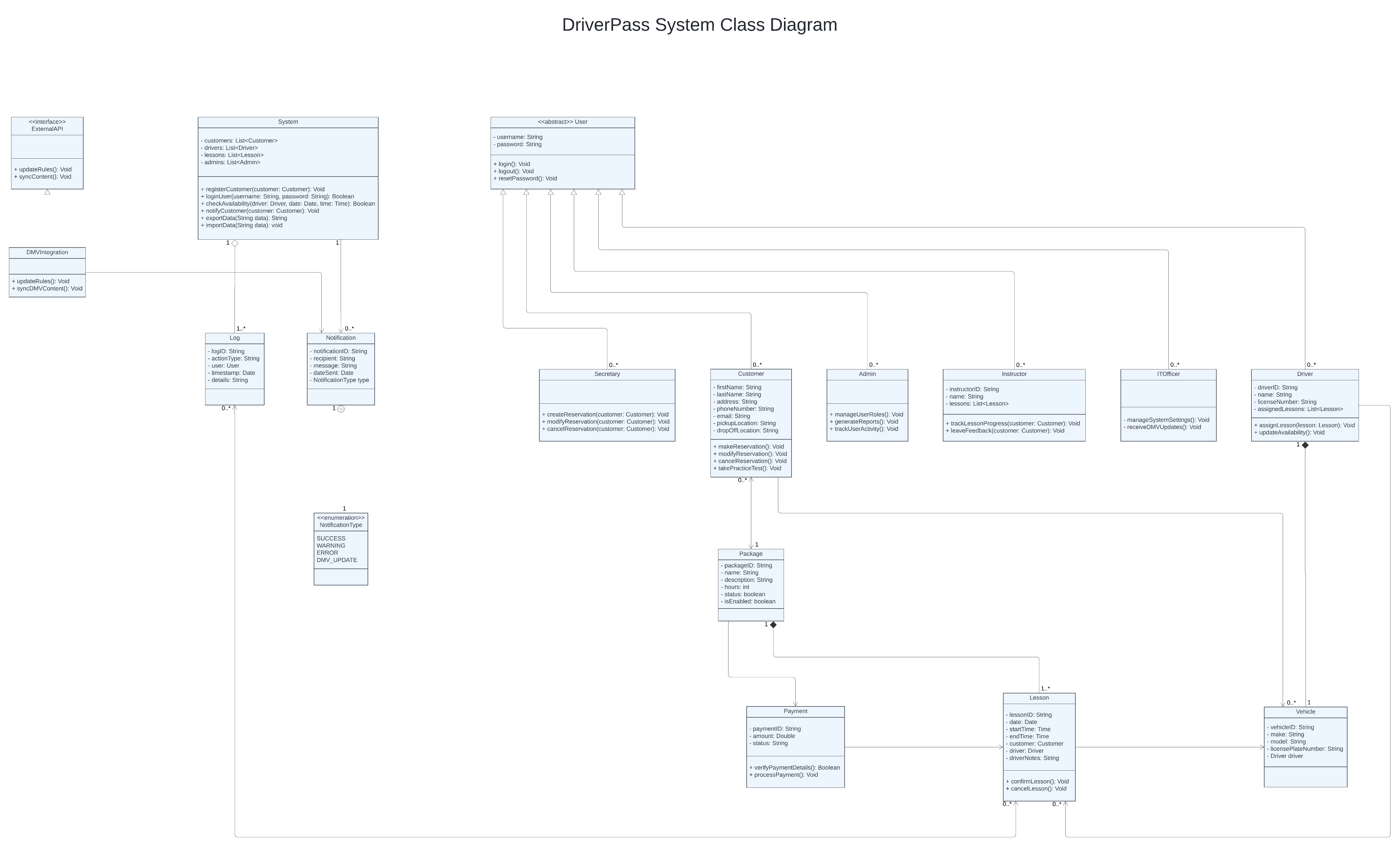
### UML Sequence Diagram

*[You were asked to create a sequence diagram based on* ***one*** *of the use cases you chose. Please insert your sequence diagram here. Check to make sure that you included appropriate components and symbols and that your design meets the client’s needs.]*



### UML Class Diagram

*[You were asked to create a class diagram based on the different classes and attributes needed for your system design. You are* ***not*** *required to include methods, but you may if you wish. Please insert your class diagram here. Check to make sure that you included appropriate components and symbols and that your design meets the client’s requirements.]*



## Technical Requirements

*[Based on the diagrams you have created, describe the technical requirements of your system. These requirements should address the required hardware, software, tools, and infrastructure necessary for your system design.]*

Hardware Requirements:

To make the DriverPass system work smoothly, we will need a cloud-based server for data storage and management. By using cloud services like AWS or Azure, we can ensure the system stays scalable and flexible while making it easy for DriverPass employees to avoid dealing with physical servers. Users can access the system from various devices, like desktops, laptops, tablets, and mobile phones. We must ensure the system is optimized for multiple devices and screen sizes, so everyone has an enjoyable experience.

Software Requirements:

DriverPass will be a web-based platform that users can access from standard web browsers like Chrome, Firefox, Safari, and Edge, whether on a desktop or mobile device. We will use responsive web design to ensure it works well on any screen. The system will also be compatible with different operating systems, including Windows, macOS, Unix/Linux, and Android/iOS, so that everyone can use it without trouble. We will use a cloud-based database like MySQL or PostgreSQL to store all the essential information, like user data, lesson schedules, reservations, and payments. The database must be reliable, support role-based access, and allow data export in formats like CSV or Excel. We will use programming languages like Java or Python on the backend, while the front end will be developed using HTML, CSS, JavaScript, and frameworks like Angular, React, or Vue.js. This will help us create a user-friendly interface that provides a nice experience.

Infrastructure Requirements:

The DriverPass system will be hosted on a cloud platform to ensure secure and reliable access for everyone, including customers, administrators, and instructors. Cloud hosting will help keep the system available, back up data, and support recovery if needed. We must also integrate with the DMV to receive updates on rules, policies, and new practice questions. This will be done using secure APIs to update the system automatically. To protect user data, we will use HTTPS for secure data transmission and implement multi-factor authentication (MFA) for secure logins. Role-based access control will also be implemented so that each user—whether a customer, instructor, IT officer, or admin—has the right level of access.

Functional Components:

The system will allow customers, instructors, IT officers, and admins to register and manage their accounts. Users can reset their passwords, and the system will use role-based permissions to protect sensitive information. Customers can book, modify, or cancel driving lessons through a simple scheduling interface. The system will also track lesson availability based on instructors and vehicles, allowing users to choose lesson packages. To manage payments, we will use a third-party payment gateway that handles transactions securely. The system will provide clear messages for successful and unsuccessful payments. Tracking user activity is another crucial feature. DriverPass will keep track of bookings, cancellations, and progress, and admins can generate valuable reports for the system used.

Non-Functional Requirements:

The DriverPass system needs to handle multiple users simultaneously without slowing down. Key actions like logging in, booking lessons, or accessing reports should take less than five seconds. The cloud infrastructure will be scalable so we can add more users, lesson packages, or even new features, like video lessons, in the future. The system will be available 24/7, with minimal downtime for maintenance, and we will use cloud redundancy strategies to ensure that. Security is a top priority, so we will use encryption, secure login protocols, and measures to prevent brute-force attacks. Admins can add, remove, or modify training packages without changing the code. This makes it easy to adjust when necessary, and we will be ready to integrate new payment options or adapt to platform updates without causing significant disruptions.

Tools and Development Environment:

Our developers will use IDEs like IntelliJ, Visual Studio Code, or PyCharm to work on the project. For version control, we will use Git along with platforms like GitHub, GitLab, or Bitbucket. To make sure everything works as expected, we will perform unit, integration, and system testing using tools like Selenium or JUnit. We will also use project management and collaboration tools like JIRA, Trello, or Microsoft Teams to maintain an agile workflow, ensuring everything moves smoothly from start to finish.