# CS 255 System Design Document

## UML Diagrams

### UML Use Case Diagram

A diagram of a diagram

Description automatically generated

### UML Activity Diagrams

**Use Case 1:**

A blue diamond with black text

Description automatically generated

**Use Case 2:**

A diagram of a customer relationship

Description automatically generated

### UML Sequence Diagram

A diagram of a system

Description automatically generated

**UML Class Diagram**

A diagram of a computer server

Description automatically generated

## Technical Requirements

1. **Hardware Requirements**:
   * **Servers**: The system requires robust servers to host the application and store the database. These servers should have sufficient processing power, memory, and storage capacity to handle concurrent user interactions and data storage needs.
   * **Networking Equipment**: Reliable networking equipment, including routers, switches, and firewalls, is essential to ensure seamless communication between client devices and the server infrastructure.
   * **Mobile Devices and Computers**: Users should be able to access the system from various devices such as computers, smartphones, and tablets. Therefore, the system should be compatible with different operating systems and device specifications.
2. **Software Requirements**:
   * **Operating System**: The server infrastructure should run on a stable and secure operating system such as Linux or Windows Server.
   * **Database Management System**: A robust DBMS is required to manage the system's data efficiently. Options include relational database systems like MySQL, PostgreSQL, or SQL Server, considering the relational nature of the data such as user accounts and reservations.
   * **Web Server**: A web server software like Apache or Nginx is necessary to serve web pages and handle HTTP requests from client devices.
   * **Programming Languages**: The system may require multiple programming languages for server-side and client-side development. Common choices include Java, Python, JavaScript (Node.js), HTML, and CSS.
   * **Development Tools**: Integrated Development Environments (IDEs) such as Eclipse, Visual Studio Code, or IntelliJ IDEA are essential for developers to write, debug, and test code efficiently.
   * **Version Control**: Version control systems like Git should be used to manage source code changes, facilitate collaboration among developers, and track project history.
3. **Infrastructure Requirements**:
   * **Cloud Platform**: Leveraging a cloud platform like Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) offers scalability, reliability, and flexibility. Cloud services such as virtual machines, databases, storage, and networking can be provisioned as needed.
   * **Backup and Disaster Recovery**: Implementing backup and disaster recovery solutions is crucial to protect against data loss and system downtime. Regular backups should be scheduled, and disaster recovery plans should be in place to ensure business continuity in case of unforeseen events.
   * **Security Measures**: The infrastructure should incorporate robust security measures, including encryption protocols, firewalls, intrusion detection/prevention systems, and access control mechanisms. Regular security audits and updates are essential to mitigate cybersecurity risks.
4. **Integration and APIs**:
   * **DMV Integration**: The system should integrate with the DMV's systems via APIs to receive updates on driving test requirements, rules, and policies. API endpoints should be securely authenticated and documented for seamless data exchange.
   * **Payment Gateway Integration**: Integration with a secure payment gateway is necessary to process online payments for reservations. This requires adherence to industry-standard security protocols such as PCI DSS compliance to safeguard sensitive financial information.
5. **Monitoring and Performance Optimization**:
   * **Monitoring Tools**: Implementing monitoring tools such as Prometheus or AWS CloudWatch allows real-time monitoring of system performance, resource utilization, and application health. Alerts should be configured to notify administrators of any anomalies or issues.
   * **Performance Optimization**: Regular performance testing and optimization are essential to ensure that the system meets performance requirements under varying loads. Techniques such as caching, load balancing, and code optimization should be employed to enhance system responsiveness and scalability.

By addressing these hardware, software, tools, and infrastructure requirements, the DriverPass system can be designed to meet the client's needs for a robust, secure, and scalable solution for driver training and reservation management.