# 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

A black background with white text

Description automatically generated

### UML Activity Diagrams

A diagram of a lesson

Description automatically generated A diagram of a payment process

Description automatically generated

### UML Sequence Diagram

A screenshot of a diagram

Description automatically generated

### UML Class Diagram

A diagram of a system

Description automatically generated

## Technical Requirements

The DriverPass system is designed to be a cloud-based web application that supports online and offline access to user data. Below are the key technical requirements for the system, covering hardware, software, tools, and infrastructure.

**1. Hardware Requirements:**

* **Servers:**  
  The system will be hosted on cloud servers with high availability. The servers should be scalable to handle varying loads, especially during peak times such as during registration periods or test dates.
* **Client Devices:**  
  The system should be accessible from various client devices, including desktops, laptops, tablets, and smartphones. These devices should have internet connectivity for online access. No specific hardware requirements are needed beyond the basic capabilities of these devices.

**2. Software Requirements:**

* **Operating System:**  
  The server-side of the system should run on a Linux-based operating system due to its stability, security, and cost-effectiveness. The client-side should be compatible with all major operating systems, including Windows, macOS, iOS, and Android.
* **Web Server:**  
  Apache or Nginx should be used as the web server to manage HTTP requests and serve the application to users.
* **Database Management System (DBMS):**  
  A relational database management system such as MySQL or PostgreSQL should be used to store user data, reservations, payments, and other critical information.
* **Programming Languages:**  
  The system’s backend should be developed using a robust server-side language like Python or Java. The frontend should be developed using HTML, CSS, and JavaScript, with a framework like React or Angular for enhanced user interface (UI) interactions.
* **Security Software:**  
  SSL/TLS certificates must be implemented to secure data transmission. Additionally, firewall and intrusion detection/prevention systems should be used to protect against unauthorized access.

**3. Tools:**

* **Development Tools:**  
  Integrated Development Environments (IDEs) such as Visual Studio Code for frontend development and PyCharm for backend development should be used. Version control should be managed using Git and GitHub.
* **CASE Tools:**  
  Lucidchart or similar tools should be used for creating and managing UML diagrams to document the system design.
* **CI/CD Tools:**  
  Jenkins or GitHub Actions should be implemented for continuous integration and continuous deployment to ensure automated testing and deployment of updates.

**4. Infrastructure:**

* **Cloud Hosting Platform:**  
  Amazon Web Services (AWS) or Microsoft Azure should be used for hosting the application, providing scalable computing resources, databases, and networking capabilities. The cloud platform should also manage backups, load balancing, and disaster recovery.
* **Backup and Recovery:**  
  Daily automated backups should be scheduled to secure user data and system configurations. Recovery procedures must be in place to ensure minimal downtime in case of system failure.
* **Network Infrastructure:**  
  A robust and secure network architecture should be established, including VPNs for remote access, secure API gateways for interfacing with external systems like the DMV, and load balancers to manage traffic.

**5. Security and Compliance:**

* **User Authentication:**  
  The system should use multi-factor authentication (MFA) to verify user identity. Role-based access control (RBAC) should be implemented to ensure that only authorized personnel can access sensitive data.
* **Data Encryption:**  
  All user data, particularly payment information, should be encrypted at rest and in transit using industry-standard encryption algorithms.
* **Compliance:**  
  The system should comply with relevant data protection regulations, such as GDPR or CCPA, ensuring user privacy and data security. Additionally, the system must stay updated with DMV regulations to ensure the training and testing materials are current.