# Anna Fuentes

CS255-10309

# CS 255 System Design Document Template

## UML Diagrams

This system design document presents the proposed design for the DriverPass system. DriverPass is a driver training service provider that seeks to enhance its scheduling, lesson-management, and user-account functionality through an integrated online and mobile platform. The design supports the client’s business goal of providing online and on-road driver training services in a streamlined, secure manner. It integrates both process and object modeling through various UML diagrams and a detailed description of the system’s technical requirements.

### UML Use Case Diagram

The DriverPass use case diagram illustrates the interaction between users and the system.  
Primary actors include the **Student, Secretary, Instructor/Driver, Owner, IT Officer,** and the **DMV System.**Each actor has distinct responsibilities and access privileges, ensuring a secure and efficient workflow.

* **Students** can register, schedule, modify, and cancel driving lessons. They can also access online learning materials, take practice tests, and view their progress.
* The **Secretary** manages student accounts, schedules, and cancellations, especially for in-office or phone requests.
* The **Instructor** records lesson notes and updates lesson completion statuses.
* The **Owner** oversees reports and system performance metrics.
* The **IT Officer** manages system maintenance, account resets, and access control.
* The **DMV System** provides updates to rules and questions to ensure test materials remain current.

This diagram also includes key relationships such as <<include>> for shared processes (e.g., Validate Login, Manage Profile) and <<extend>> for optional behaviors (e.g., Download Reports).

A diagram of a driver pass system

AI-generated content may be incorrect.

### UML Activity Diagrams

**Use Case #1: Schedule Lesson**

* Insert Activity Diagram 1 here.
* This diagram models the workflow for a customer scheduling a driving lesson: e.g., “Log In → Choose Package → Select Date/Time → Confirm Reservation → System Saves Lesson → Send Confirmation”.
* The diagram helps identify decision points (for example: if the selected time is unavailable, redirect to “Choose different time”) and enforces clarity around flow.

**Use Case #2: Manage User Accounts**

* Insert Activity Diagram 2 here.
* This diagram models the workflow for administrative account management: e.g., “Administrator logs in → Select User → Assign Role / Reset Password → Save Changes → Notify User”.
* Provides clarity on internal processes and helps developers understand system behavior for administrative functions.

*A diagram of a flowchart

AI-generated content may be incorrect.*

### A diagram of a system AI-generated content may be incorrect.

### UML Sequence Diagram

* The sequence diagram should include lifelines for **Customer, System, Database,** and **Instructor** (or whichever actors relevant). It should show the message flow such as:
  1. Customer: “Request reservation”
  2. System: “Validate data”
  3. System → Database: “Check availability”
  4. Database: “Return availability status”
  5. System: “Save reservation”
  6. System → Instructor: “Notify of new lesson”
  7. System → Customer: “Send confirmation”
* Sequence diagrams help communicate timing and interaction between components, which is key for the development team.

A diagram of a program

AI-generated content may be incorrect.

### UML Class Diagram

The DriverPass Class Diagram illustrates the main structural components of the system and how they relate to one another. The central **User** class defines shared attributes such as userID, username, password, and role. Specialized roles: **Student, Instructor, Secretary, Owner**, and **ITOfficer**, extend from this class to represent different user permissions. Supporting classes include **Lesson, Vehicle, Package, Account**, and **DMVSystem**, which handle scheduling, resources, and data management. Relationships between these classes show how users interact with lessons, vehicles, and system data, forming the foundation of DriverPass’s functionality.

A diagram of a driver pass

AI-generated content may be incorrect.

## Technical Requirements

**Hardware Requirements**

* Cloud-based infrastructure (for example virtual servers in a public cloud provider) to host the web application, database, and file storage.
* End-user devices: desktops, laptops, tablets, and mobile phones (iOS/Android) so that customers and instructors can access the system from various locations.
* Network connectivity: reliable broadband/internet access with reasonable latency for real-time scheduling and notifications.

**Software Requirements**

* Web server software (Apache HTTP Server or NGINX) or a PaaS offering to host the front-end web application components.
* Database management system:MySQL, PostgreSQL, or a cloud-native managed database service for storing user, lesson, and schedule data.
* Web application framework: Java Spring Boot, Python Flask/Django, or equivalent modern stack for implementing business logic.
* Front-end technologies: HTML5, CSS3, JavaScript (and possibly a framework like React or Angular) to provide a user-friendly customer/instructor interface.
* Browser support: Chrome, Firefox, Edge, Safari — ensuring cross-platform compatibility.

**Tools and Infrastructure**

* Modeling tool: Lucidchart for creating UML diagrams (use cases, activity, sequence, class diagrams).
* Secure cloud hosting (Amazon Web Services AWS or Microsoft Azure) with encrypted connections (HTTPS/TLS).
* Logging and monitoring tools: to track user activity, system performance, and security events.
* Role-based access control (RBAC) system: to manage distinct user roles (Owner, IT Officer, Secretary, Instructor, Customer) with permissions.

**Security Considerations**

* Encrypt data in transit (using TLS) and at rest (using Provider-managed encryption) to maintain confidentiality and integrity of data. Microsoft Learn+2Rippling+2
* Implement multi-factor authentication (MFA) especially for administrative or high-privilege users to reduce risk of unauthorized access. [SentinelOne+1](https://www.sentinelone.com/platform/small-business/cloud-security-for-small-businesses/?utm_source=chatgpt.com)
* Regular security audits and configuration reviews to identify misconfigurations and vulnerabilities within cloud infrastructure. [SentinelOne+1](https://www.sentinelone.com/cybersecurity-101/cloud-security/small-business-cloud-security/?utm_source=chatgpt.com)
* Role-based access control (RBAC) and least-privilege principle to ensure users have only the permissions necessary for their tasks. [Ricoh USA+1](https://www.ricoh-usa.com/en/insights/articles/data-security-best-practices-every-small-business-should-follow?utm_source=chatgpt.com)
* Backup and disaster recovery strategy: regular backups of database and configuration files, encrypted and stored off-site to ensure recovery in event of a breach or system failure.