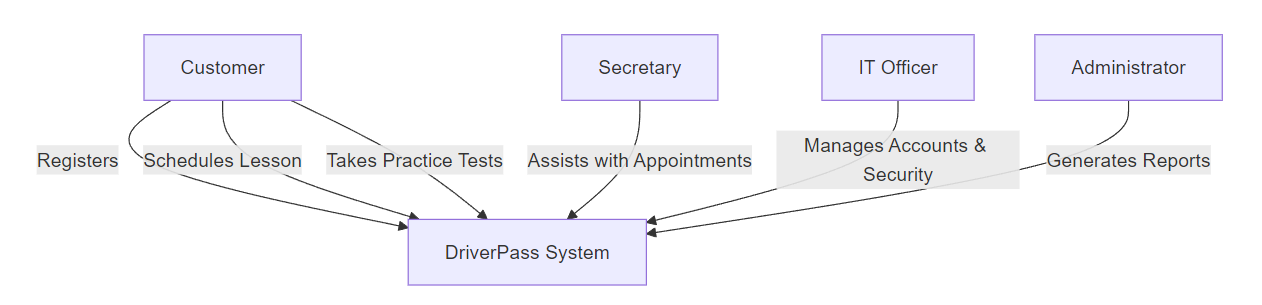
# CS 255 DriverPass System Design Document

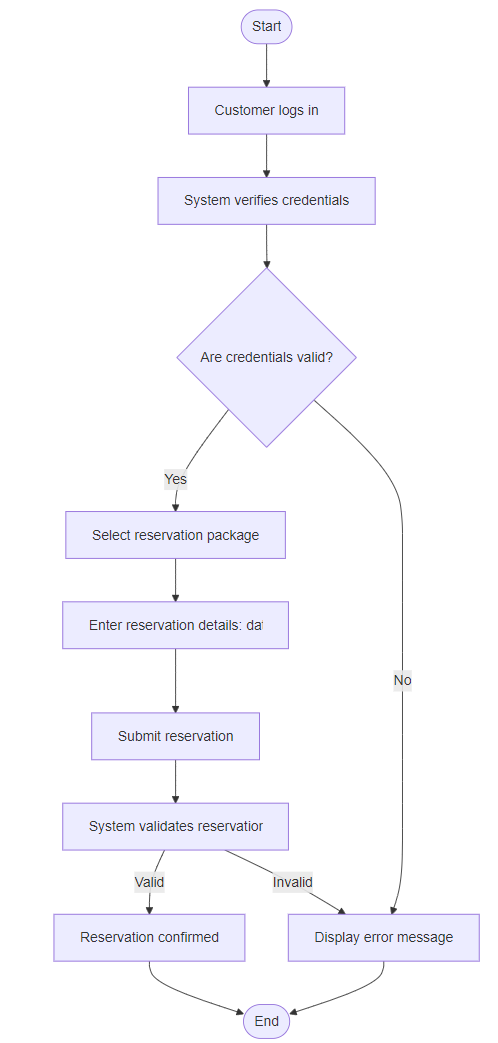
**Project Purpose:**  
Design and develop a web-based system for DriverPass that provides online driver training, practice tests, and scheduling for in-person driving lessons. The system will also offer role-based access for administrators, track user activity, and generate performance and scheduling reports.

## UML Diagrams

### UML Use Case Diagram

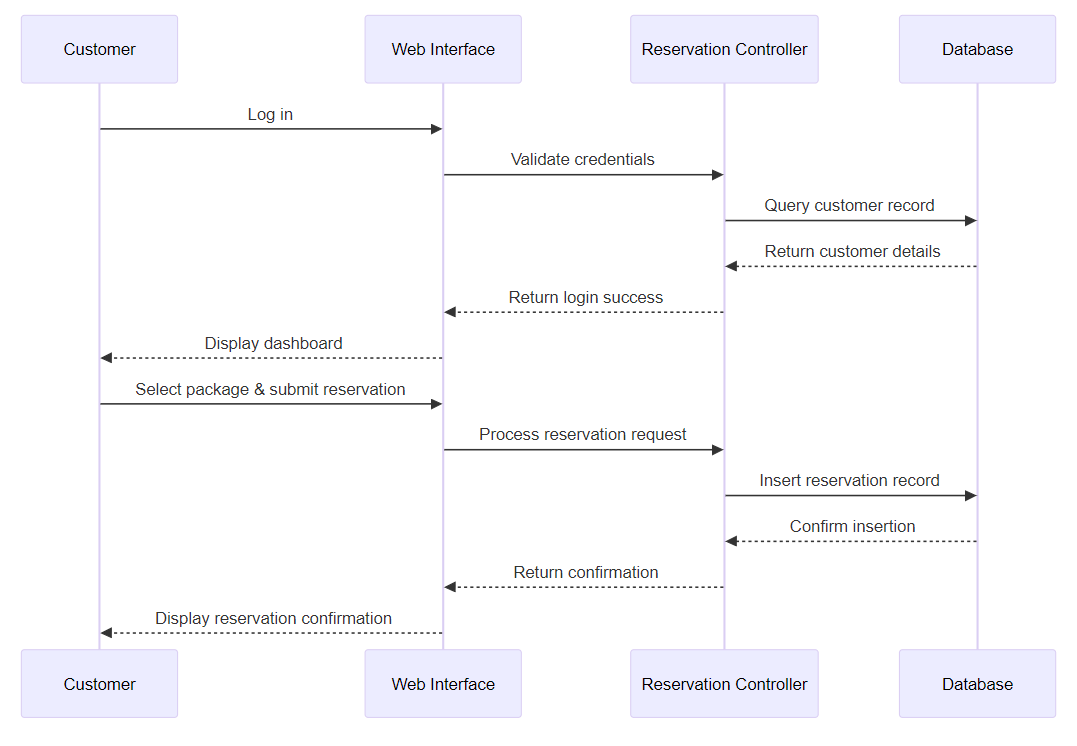


### UML Activity Diagrams

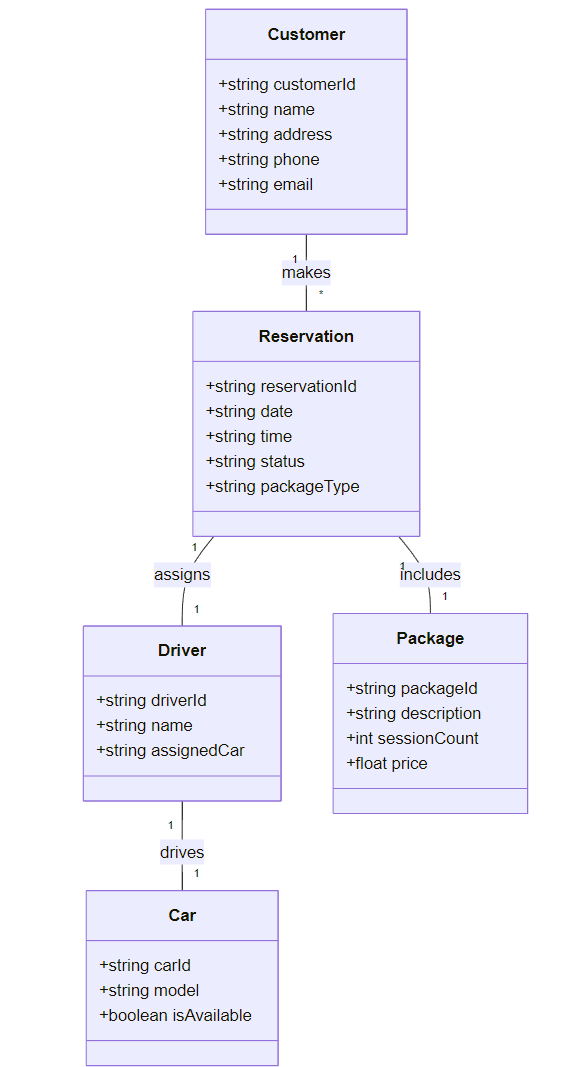
Activity Diagram 1: Make a Reservation

### Activity Diagram 2: Modify/Cancel Reservation

### UML Sequence Diagram



### UML Class Diagram



## Technical Requirements

Based on the UML diagrams and the client’s business requirements, the system must meet the following technical specifications:

**Hardware & Infrastructure**

* **Cloud-Based Hosting:**  
  – Deploy the system on a cloud platform (e.g., AWS, Azure, or Google Cloud) to ensure high availability and scalability.  
  – Support auto-scaling during peak usage periods.
* **Servers & Storage:**  
  – Utilize robust servers with sufficient CPU and memory resources to handle concurrent user access.  
  – Implement secure, redundant storage solutions with regular backups and disaster recovery mechanisms.
* **Client Devices:**  
  – Ensure compatibility across desktops, laptops, and mobile devices via modern web browsers.

**Software**

* **Web Application Framework:**  
  – Develop the system as a web-based application using a server-side language (e.g., Node.js, Python, or Java) and a modern front-end framework (e.g., React, Angular, or Vue.js) to provide a responsive user interface.
* **Database Management:**  
  – Employ a relational database system (such as MySQL or PostgreSQL) to store user data, reservations, and activity logs.  
  – Ensure real-time data synchronization to prevent redundancy and maintain data integrity.
* **Reporting & Data Export:**  
  – Provide functionality to export reports in formats like CSV or XLSX, facilitating offline analysis in tools such as Excel.

**Tools & Integration**

* **Diagramming & Documentation:**  
  – Create UML diagrams using Typora with embedded Mermaid.js code for a lightweight, code-driven approach.
* **Security Tools:**  
  – Integrate SSL/TLS encryption for secure data transmission.  
  – Use monitoring tools (e.g., ELK Stack, Splunk) to log user activity and track system performance.
* **Version Control & CI/CD:**  
  – Utilize Git for source code management and establish CI/CD pipelines for streamlined development and deployment.

**Security & Access**

* **Authentication & Authorization:**  
  – Implement role-based access control, ensuring that users (customers, secretaries, IT officers, and administrators) have the appropriate permissions.  
  – Enforce strong password policies and consider multi-factor authentication for sensitive operations.
* **Data Protection:**  
  – Encrypt sensitive data at rest and in transit.  
  – Maintain comprehensive logs of system activity to support audits and troubleshooting.
* **User Activity Tracking:**  
  – Record all changes to reservations and user data to quickly identify and resolve any issues.

**Scalability & Adaptability**

* **Modular Architecture:**  
  – Design the system so that features—such as training packages—can be easily enabled, disabled, or modified without extensive code changes.
* **Performance Optimization:**  
  – Use load balancing to distribute traffic evenly across servers.  
  – Ensure the system can scale horizontally to meet future growth demands.