CS 255 System Design Document Template

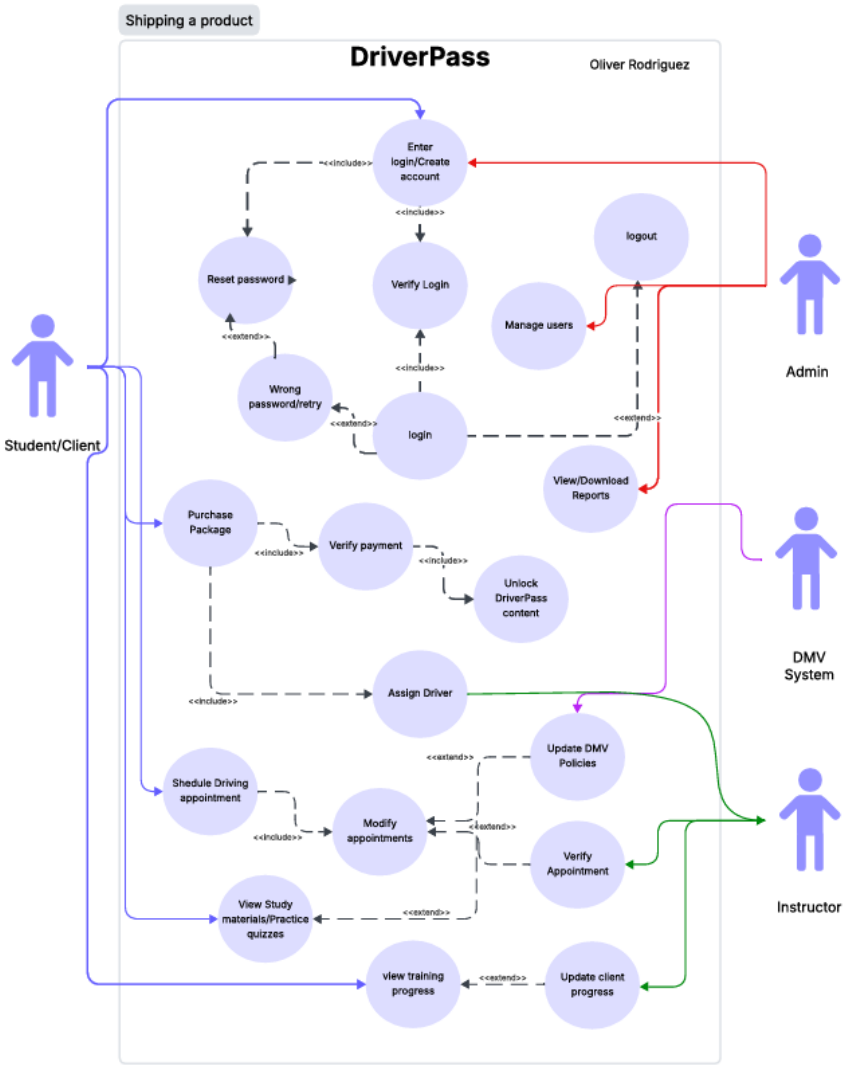
Oliver Rodriguez

CS 255

UML Diagrams

UML Use Case Diagram

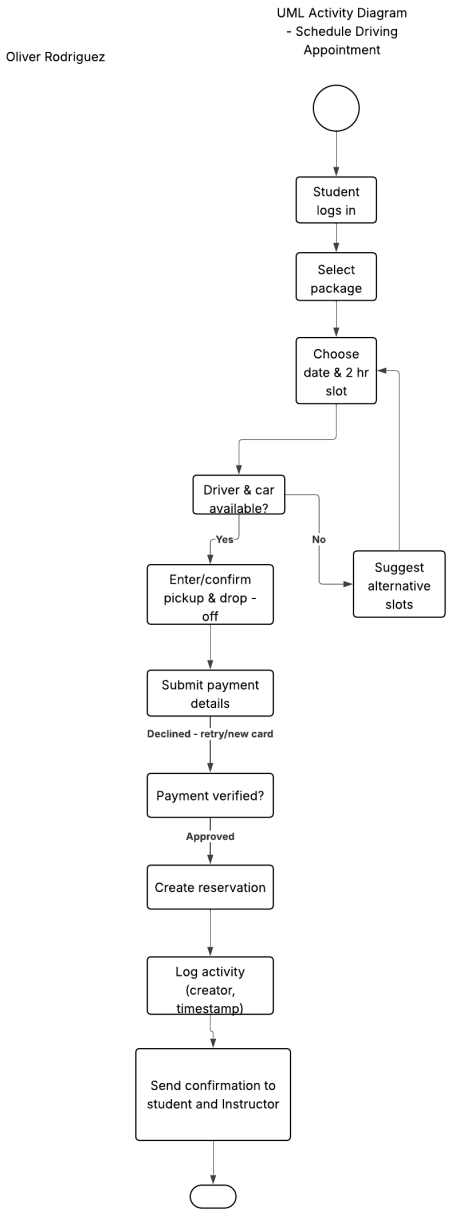
[

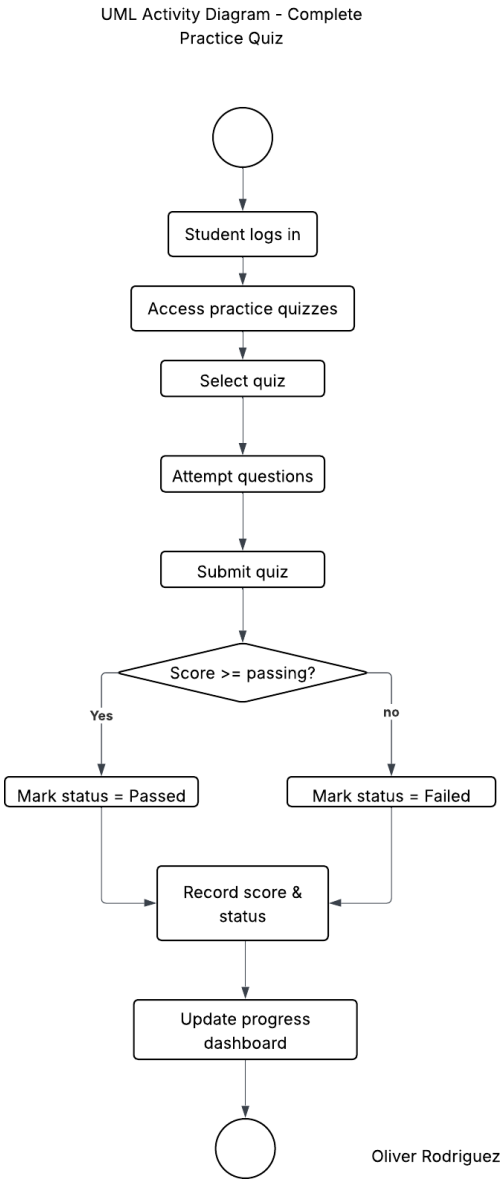


]

UML Activity Diagrams

[

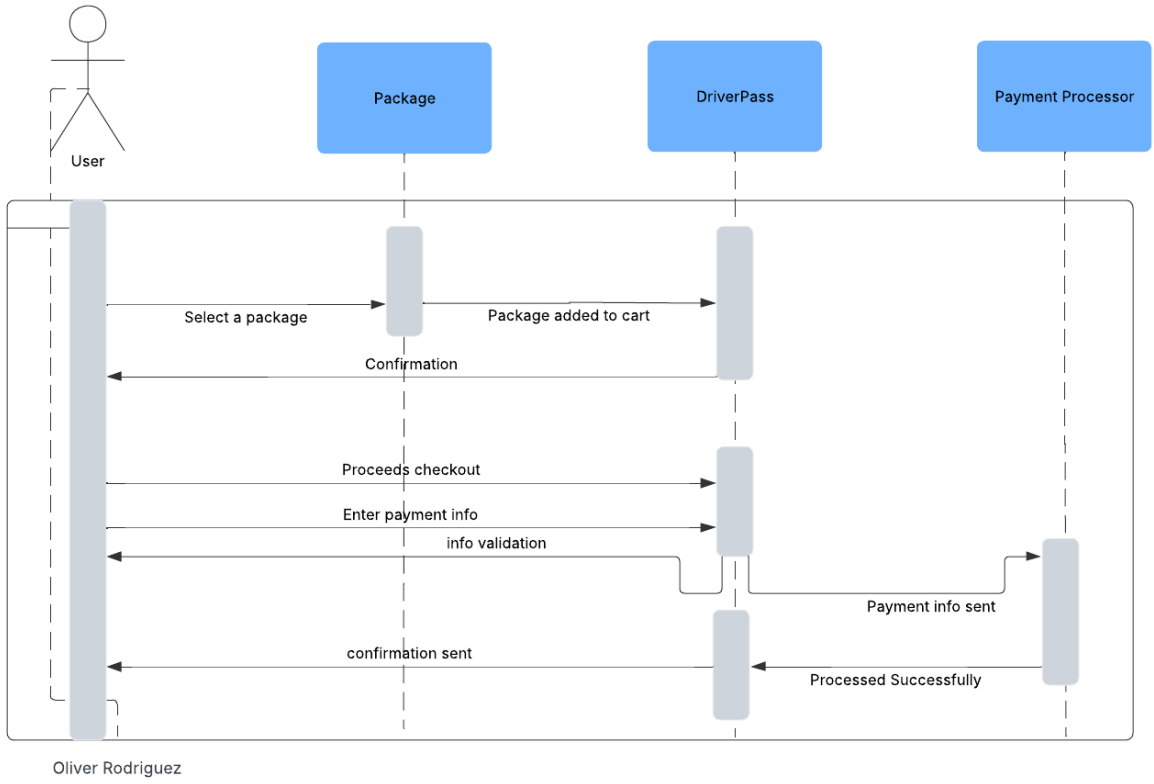




]

UML Sequence Diagram

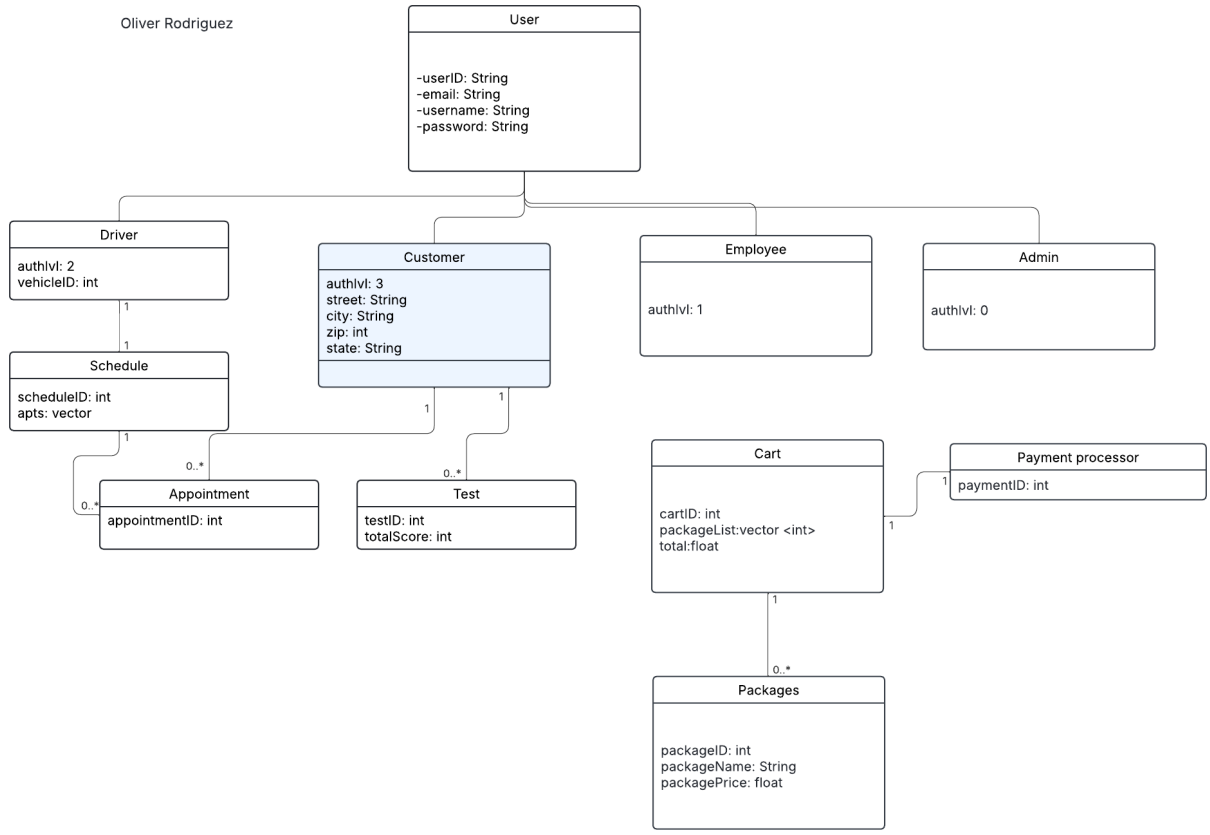
[



]

UML Class Diagram

[



]

Technical Requirements

Hardware Requirements

- **Client Devices**:
  - Students, instructors, and administrators will access the system through standard desktops, laptops, or mobile devices.
  - Mobile devices must support modern browsers (Safari, Chrome, Edge, Firefox).
- **Server Hardware (Cloud-hosted)**:
  - Cloud provider (e.g., AWS, Azure, or Google Cloud) with load-balanced web servers.
  - Minimum specs: 8 vCPUs, 16 GB RAM, and 250 GB SSD storage to handle simultaneous reservations, tests, and report generation.
  - Auto-scaling enabled to handle peak usage (e.g., before DMV test dates).

Software Requirements

-

    - Browser-based interface developed with **HTML5, CSS, JavaScript**, and a modern framework (e.g., React or Angular).
    - Responsive design to support both mobile and desktop access.
      Handles appointment scheduling, package management, practice quiz logic, and reporting.
    - Middleware to process secure payment transactions via a third-party Payment Gateway API.
- **Database**:
    - **Relational Database (MySQL or PostgreSQL)** to store structured data such as user accounts, appointments, test results, and packages.
    - Enforced relationships as per UML class diagram (e.g., Customer ↔ Appointment, Cart ↔ Packages).
    - Daily backups configured automatically.
- **Operating Systems**:
    - Server OS: **Linux (Ubuntu LTS)** for scalability and security.
    - Client OS: Cross-platform (Windows, macOS, iOS, Android) via browser support.

Tools

- **CASE Tool**: Lucidchart (for UML diagram creation and ongoing system design documentation).
- **IDE/Development Tools**: Eclipse, IntelliJ, or VS Code for development.
- **Version Control**: GitHub or GitLab for source code management.
- **Testing Tools**: Selenium (UI testing), JUnit/PyTest (unit testing).

Infrastructure

- **Hosting**:
    - Cloud-based deployment (AWS Elastic Beanstalk or Azure App Service) to minimize local infrastructure needs.
    - Integrated CI/CD pipeline to streamline updates.
- **Security Infrastructure**:
    - All data exchanged via **HTTPS/TLS encryption**.
    - **Role-based access control** as defined in the class diagram (Admin, Driver, Customer, Employee, Manager).
    - Automatic account lockout after multiple failed login attempts.
    - Encrypted storage for sensitive information such as payment details (PCI-DSS compliance).
- **Scalability and Maintenance**:
    - Cloud-native design ensures scaling to support growth (e.g., adding new training packages).
    - IT Admin can add/remove users, reset passwords, and disable packages without code changes.