CS 255 System Design Document Template

Student: Christopher Wright
Course: CS 255 – System Analysis and Design
Date: August 11, 2025

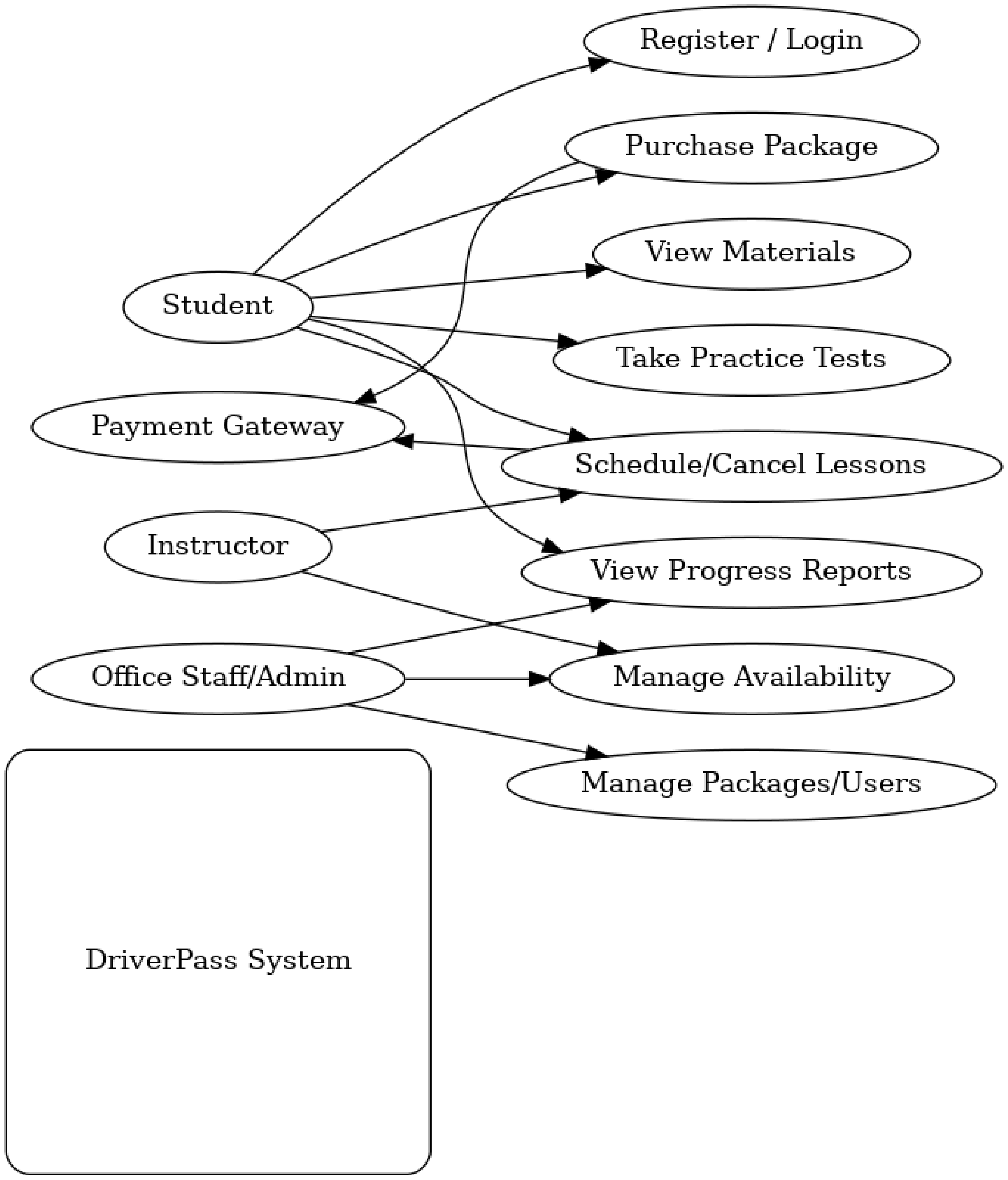
UML Diagrams

UML Use Case Diagram

Interpretation:
The use case diagram illustrates the main interactions between the DriverPass system and its users, including Students, Instructors, Office Staff/Admin, and the Payment Gateway.

Key use cases include:

- Register/Login
- Purchase Training Package
- View Materials
- Take Practice Tests
- Schedule/Cancel Lessons
- Manage Availability (Instructors)
- Manage Packages/Users (Admins)
- View Progress Reports

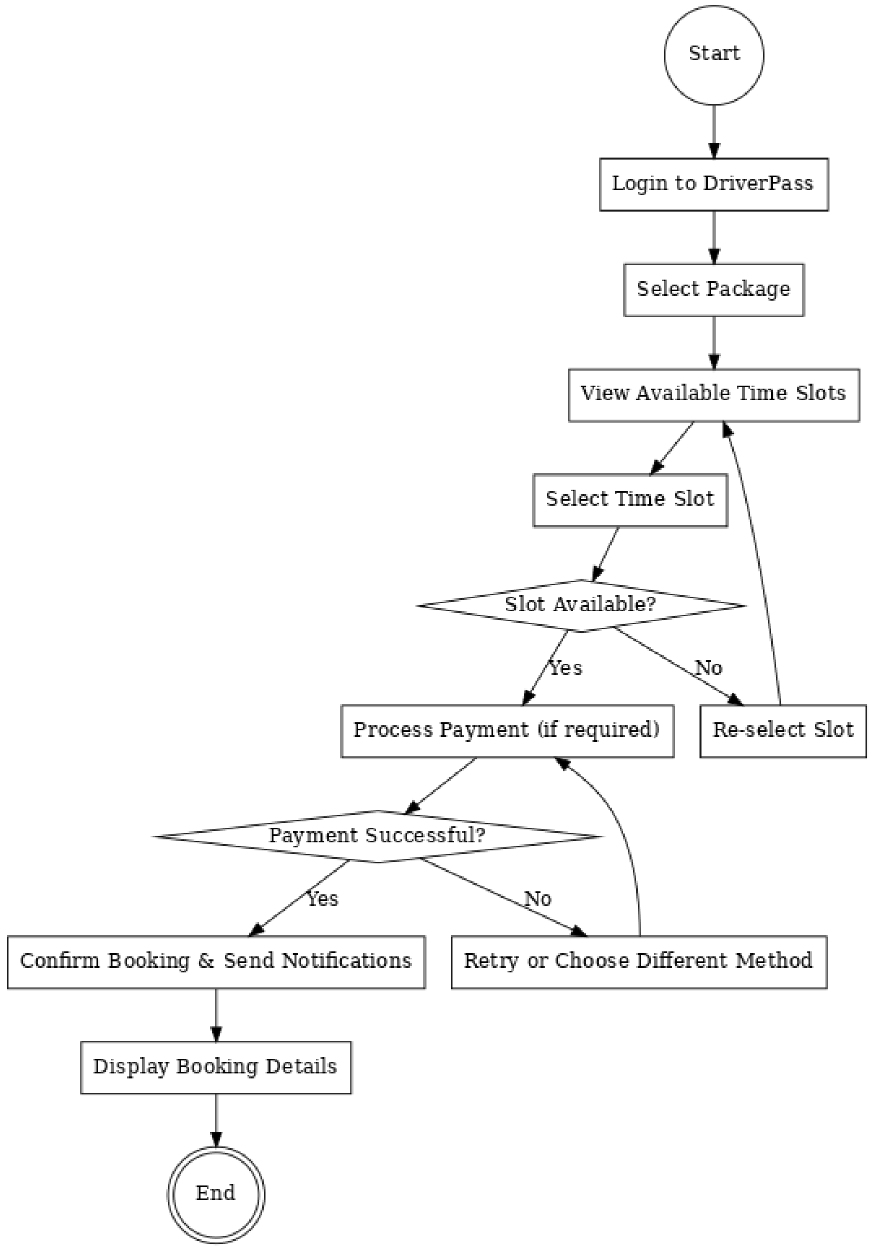


UML Activity Diagrams
Activity Diagram 1 – Schedule In-Car Lesson

Interpretation:
This activity diagram models the steps a student follows to schedule an in-car driving lesson. The process includes logging in, selecting a package, viewing available instructor time slots, booking a slot, processing payment if required, and confirming the booking with notifications sent to both student and instructor.

<u>Improvements Implemented:</u>

- Added loop to allow the student to select another slot if the chosen one becomes unavailable.
- Added branch to handle payment failures with retry or alternate payment method.
- Included detailed confirmation step with booking details.
- Incorporated session timeout handling.



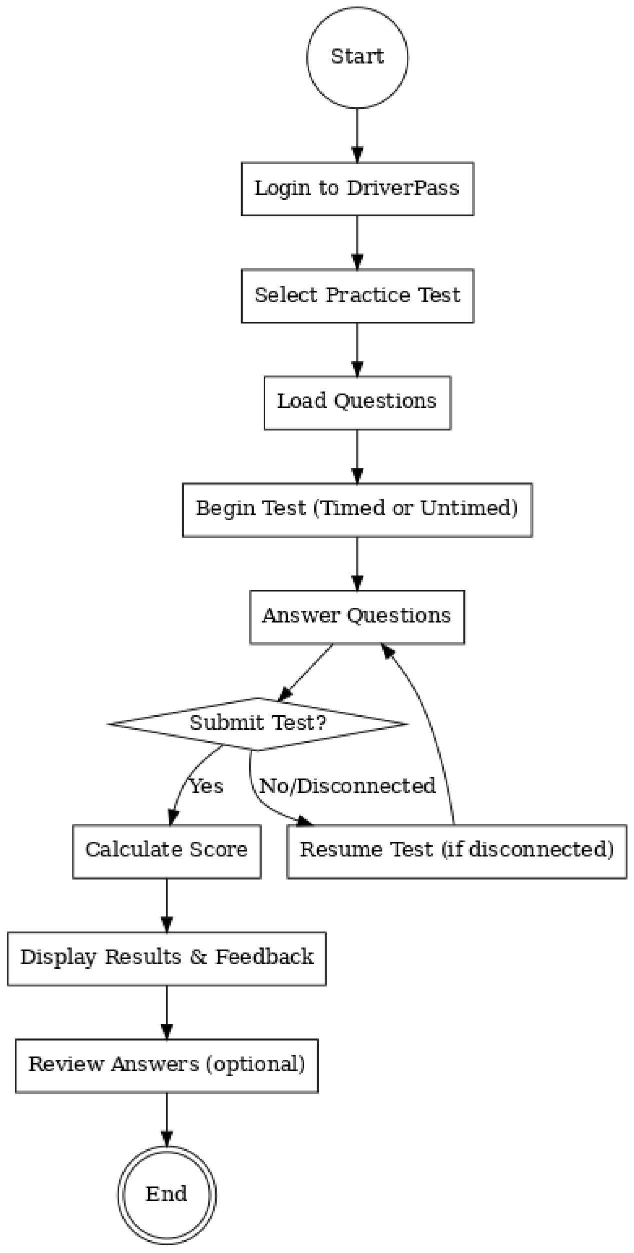
Activity Diagram 2 – Take Practice Test

Interpretation:

This diagram models a student taking an online practice test. It includes selecting a test, loading questions, answering in a timed/untimed format, submitting the test, calculating the score, and displaying results.

Improvements Implemented:

- Added handling for connectivity loss with resume option.
- Added retry loop for submitting answers if system error occurs.
- Included option to review answers after completion.

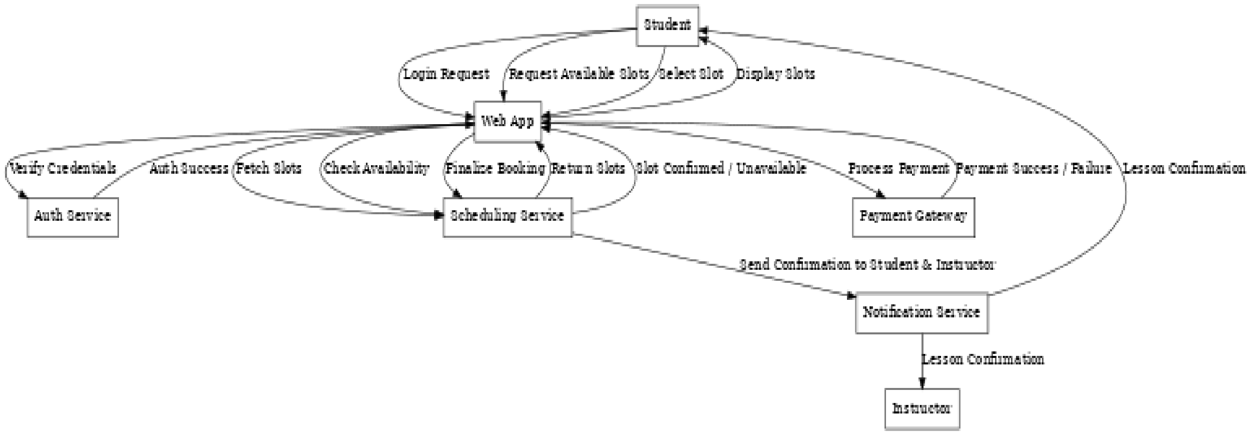


Southern New Hampshire University

UML Sequence Diagram
Interpretation:
The sequence diagram depicts the booking process for an in-car lesson. It shows interactions between the Student, Web Application, Authentication Service, Scheduling Service, Payment Gateway, and Notification Service.

<u>Key Enhancements:</u>

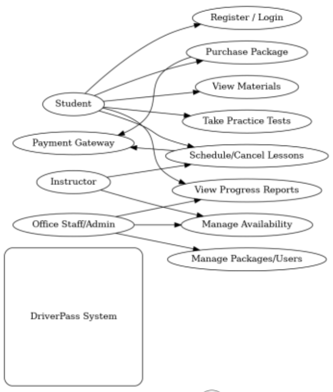
- Added alternative fragment for time slot conflicts.
- Added alternative fragment for payment failures, allowing retry or booking cancellation.
- Confirm booking details before notification is sent.

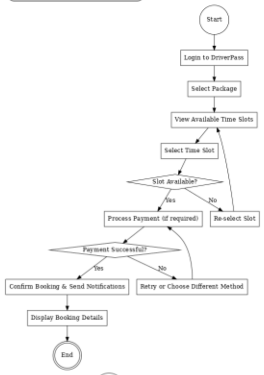


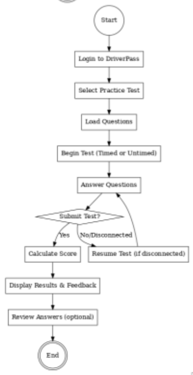
UML Class Diagram
Interpretation:
The class diagram outlines the core entities in DriverPass, including User, Student, Instructor, Staff, Package, Enrollment, Lesson, Vehicle, AvailabilitySlot, PracticeExam, Question, Attempt, ScoreReport, Payment, and Notification.

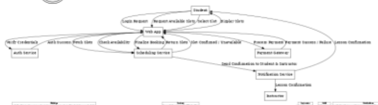
<u>Key Details:</u>

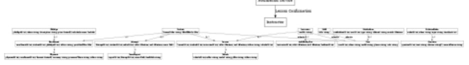
- Inheritance from User for Student, Instructor, and Staff classes.
- Association between Student and Enrollment, Enrollment and Package, Lesson and Instructor.
- Multiplicity indicators showing one-to-many relationships.
- Attributes specified for each class (e.g., name, email, status, etc.).











Technical Requirements

Hardware Requirements

- Servers: Cloud-hosted VM or container infrastructure capable of running Java/Spring Boot applications and MySQL/PostgreSQL databases.
- Client Devices: Any modern desktop, laptop, or mobile device with an internet connection and a web browser.
- Networking: Reliable internet connection, HTTPS support, load balancer for scalability.

Software Requirements

- Frontend: HTML5/JavaScript with a modern framework (e.g., React, Angular, or Vue.js).
- Backend: Java with Spring Boot framework.
- Database: MySQL or PostgreSQL.
- External Services: Payment gateway API (e.g., Stripe), email/SMS API (e.g., SendGrid/Twilio).
- Version Control: Git/GitHub or GitLab.
- CI/CD: Jenkins, GitHub Actions, or GitLab CI.

Tools

- UML diagramming software (Lucidchart, draw.io, or Visual Paradigm).
- IDE for development (IntelliJ IDEA, Eclipse, or VS Code).
- API testing tool (Postman).
- Testing frameworks (JUnit for Java, Selenium for UI tests).

Infrastructure

- Environments: Development, Testing, and Production environments.
- Security: SSL/TLS for secure connections, role-based access control, encrypted storage for sensitive data, and regular security audits.
- Monitoring: Application performance monitoring (APM) tools such as New Relic or Datadog.