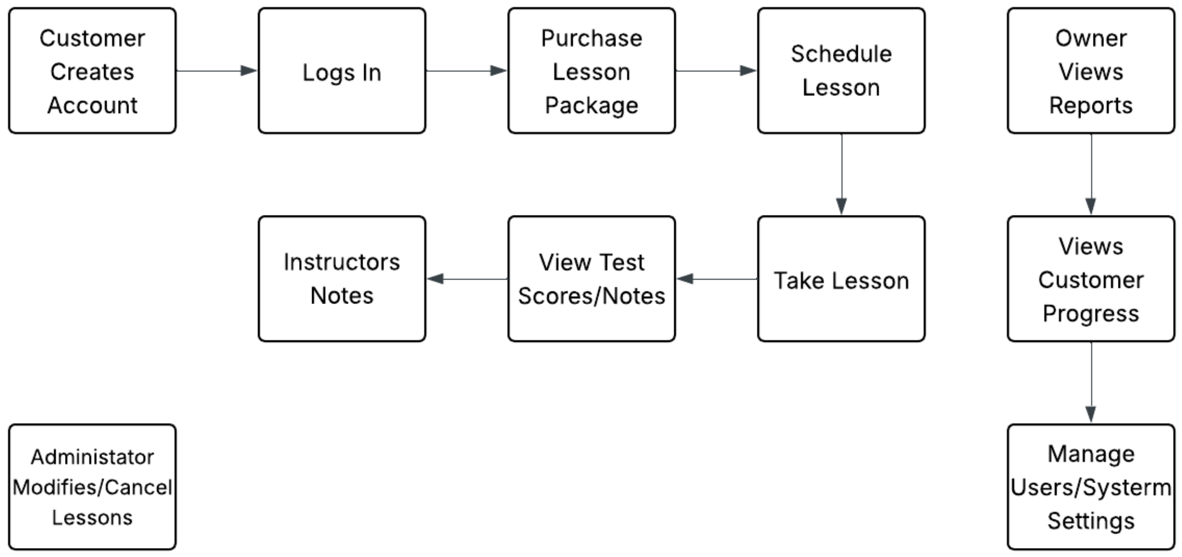
Angel Silva
December 11, 2025
CS-255 Project Two

UML Use Case Diagram



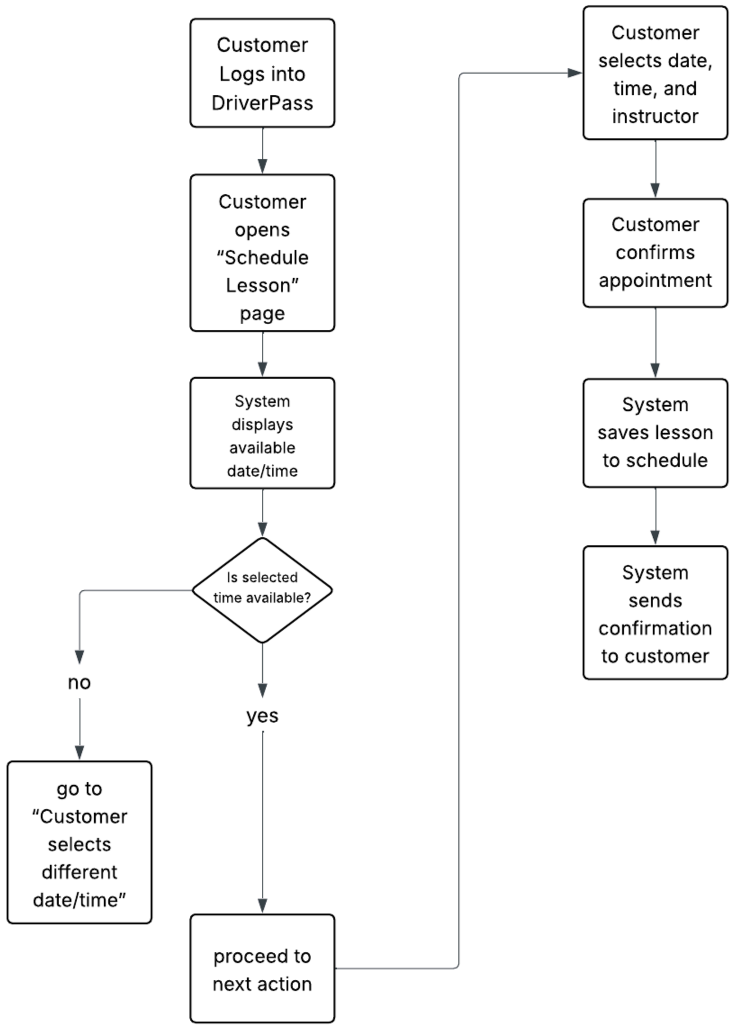
Description:
The use case diagram illustrates the major system interactions required for DriverPass. Five actors interact with the system:

- **Customer** – creates accounts, logs in, purchases lesson packages, schedules lessons, takes practice tests, and views test scores and instructor notes.
- **Secretary** – schedules and modifies/cancels lessons on behalf of customers and views student progress.
- **Administrator / IT Officer** – manages users, system settings, and reports.
- **Owner** – views high-level business reports and student progress metrics.
- **DMV System** – sends regulatory updates to the DriverPass system.

This diagram ensures all major functional requirements from the interview transcript and Project One are captured within the system boundary.

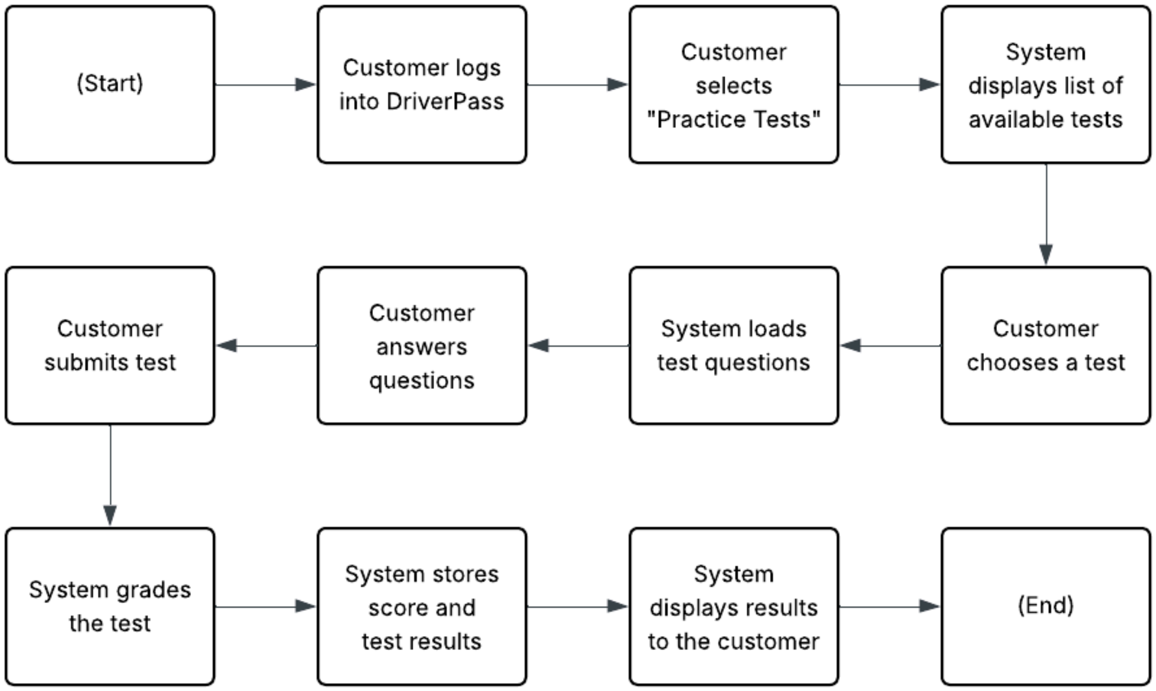
UML Activity Diagrams

Activity Diagram #1 – Scheduling a Lesson


Customer Logs into DriverPass
Customer opens "Schedule Lesson" page
System displays available date/time
Is selected time available?
no
go to "Customer selects different date/time"
yes
proceed to next action
Customer selects date, time, and instructor
Customer confirms appointment
System saves lesson to schedule
System sends confirmation to customer

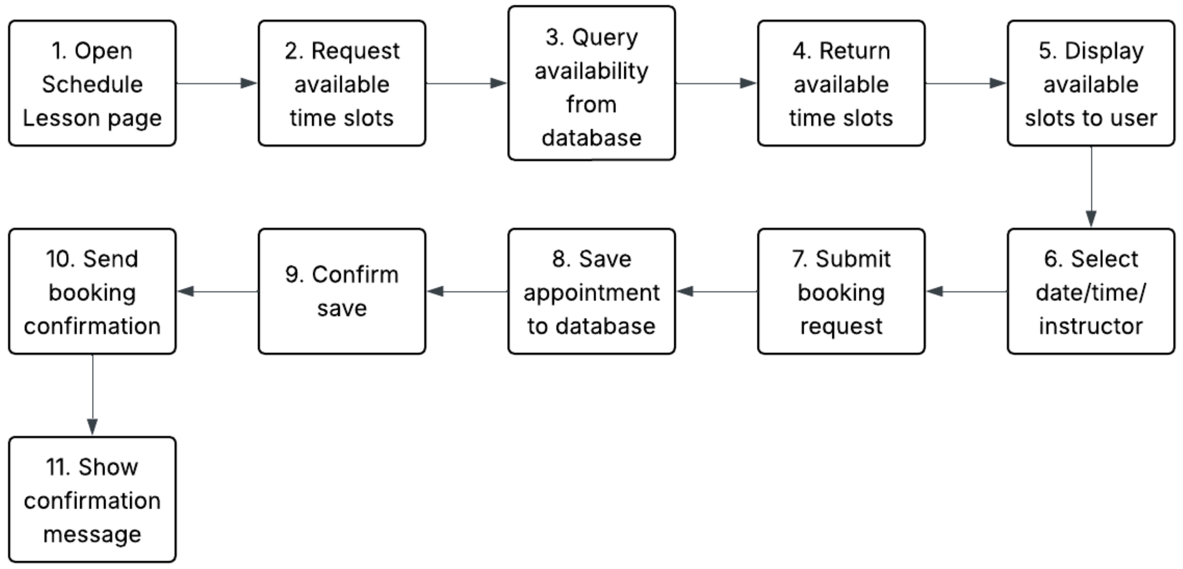
Description:
This activity diagram models the step-by-step workflow a customer follows to schedule a driving lesson. The process includes logging into DriverPass, navigating to the scheduling page, viewing available dates and times, selecting an instructor, confirming the appointment, and receiving system confirmation. A decision node handles lesson availability, ensuring conflicts are avoided.

Activity Diagram #2 – Taking an Online Practice Test


(Start)
Customer logs into DriverPass
Customer selects "Practice Tests"
System displays list of available tests
Customer chooses a test
System loads test questions
Customer answers questions
Customer submits test
System grades the test
System stores score and test results
System displays results to the customer
(End)

Description:
This diagram outlines the customer workflow for completing an online practice test. The user logs in, selects a practice test, answers questions, submits the test, and receives a graded result. This diagram demonstrates how DriverPass supports student learning outside of in-person lessons.

UML Sequence Diagram – Scheduling a Lesson



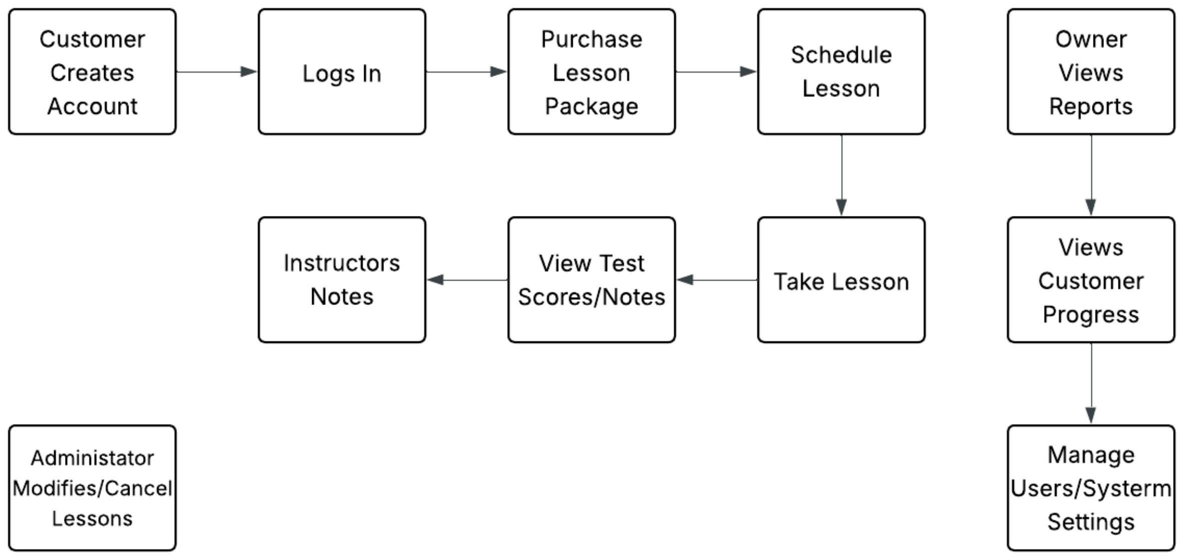
1. Open Schedule Lesson page
2. Request available time slots
3. Query availability from database
4. Return available time slots
5. Display available slots to user
6. Select date/time/ instructor
7. Submit booking request
8. Save appointment to database
9. Confirm save
10. Send booking confirmation
11. Show confirmation message

Description:
The sequence diagram shows the real-time interaction between system components during the lesson scheduling process:

- The **Customer** requests lesson availability.
- The **Web UI** forwards the request to the **Application Server**.
- The server queries the **Database** for open time slots.
- Once a slot is selected, the server saves the appointment and sends confirmation back to the customer.

This diagram illustrates system communication flow and verifies that the scheduling functionality is implemented correctly.

UML Class Diagram



Description:
The class diagram represents the data structure for DriverPass. The key classes include:

- **User** (superclass) – attributes include userID, name, email, passwordHash, and role.
- **Customer** (subclass) – adds packageType, lessonCredits, and testHistory.
- **Instructor** – stores instructorID, availability, and qualifications.
- **Lesson** – includes lessonID, date, time, status, and notes.
- **PracticeTest** – includes testID, score, and dateTaken.
- **Payment** – stores paymentID, amount, method, and package information.
- **DMVUpdate** – contains updateID, type, description, and date received.

Relationships:

- A **Customer** can have multiple lessons, payments, and practice tests.
- An **Instructor** can teach multiple lessons.
- Lessons link customers and instructors.

This structure supports all functional processes identified for DriverPass.

Technical Requirements

The following technical requirements ensure the DriverPass system is secure, scalable, and meets user needs. These requirements are derived from the business and system requirements defined in Project One.

Hardware Requirements

- Cloud-hosted virtual servers (AWS, Azure, or Google Cloud) to run application logic and store data.
- End-user devices such as laptops, desktops, tablets, and smartphones capable of running modern web browsers.
- Secure storage for system logs, backups, and DMV update files.

Software Requirements

- Web application framework such as **Node.js**, **Spring Boot**, or **.NET Core**.

- **HTML5 / CSS / JavaScript** for front-end presentation.
- Relational database system such as **MySQL** or **PostgreSQL**.
- Secure authentication library with support for role-based access control.
- API integration layer for communication with **DMV external systems**.

Tools

- **Lucidchart** for UML modeling.
- **GitHub** or similar version control system for source code tracking.
- **VS Code** or equivalent IDE for development.
- Automated testing tools for functional, security, and performance validation.

Infrastructure Requirements

- Cloud environment supporting auto-scaling, load balancing, and redundancy.
- **HTTPS encryption** using TLS/SSL certificates.
- Role-based access control enforcing permissions for Customer, Secretary, Administrator, and Owner roles.
- Daily automated data backups and redundancies to maintain uptime.
- Secure database connections using industry-standard encryption.
- Reliable integration pipeline to receive DMV updates.