Southern New Hampshire University

CS 255 System Design Document
Joshua Sevy
February 20, 2026

DriverPass System Summary

The DriverPass System provides learners with online learning and scheduled driving lessons to help prepare for their driver's test. Customers can create an account, purchase training packages, schedule driving lessons, complete practice tests, and track their progress through the DriverPass system. Instructors, administrators, and support staff will have access to the same platform through role-based access to schedule customers, access content, and operate the system.

The DriverPass system is designed using both process and object-oriented modeling. Process modeling makes sure that workflows like lesson creation and payment processing follow standard business practices. Object modeling shows how the different parts of the system are organized.

UML Diagrams

UML Use Case Diagram

The UML use case diagram identifies the primary actors interacting with the DriverPass system:

- Customer
- Instructor
- Secretary
- Administrator/Owner
- IT Officer
- External Payment Processor
- DMV Content Provider

The use case diagram below (Figure 1) displays the core functionality of the DriverPass System, including account management, lesson scheduling, package purchasing, practice testing, administration reporting, and system maintenance. External systems support payment authorization and DMV content synchronization.

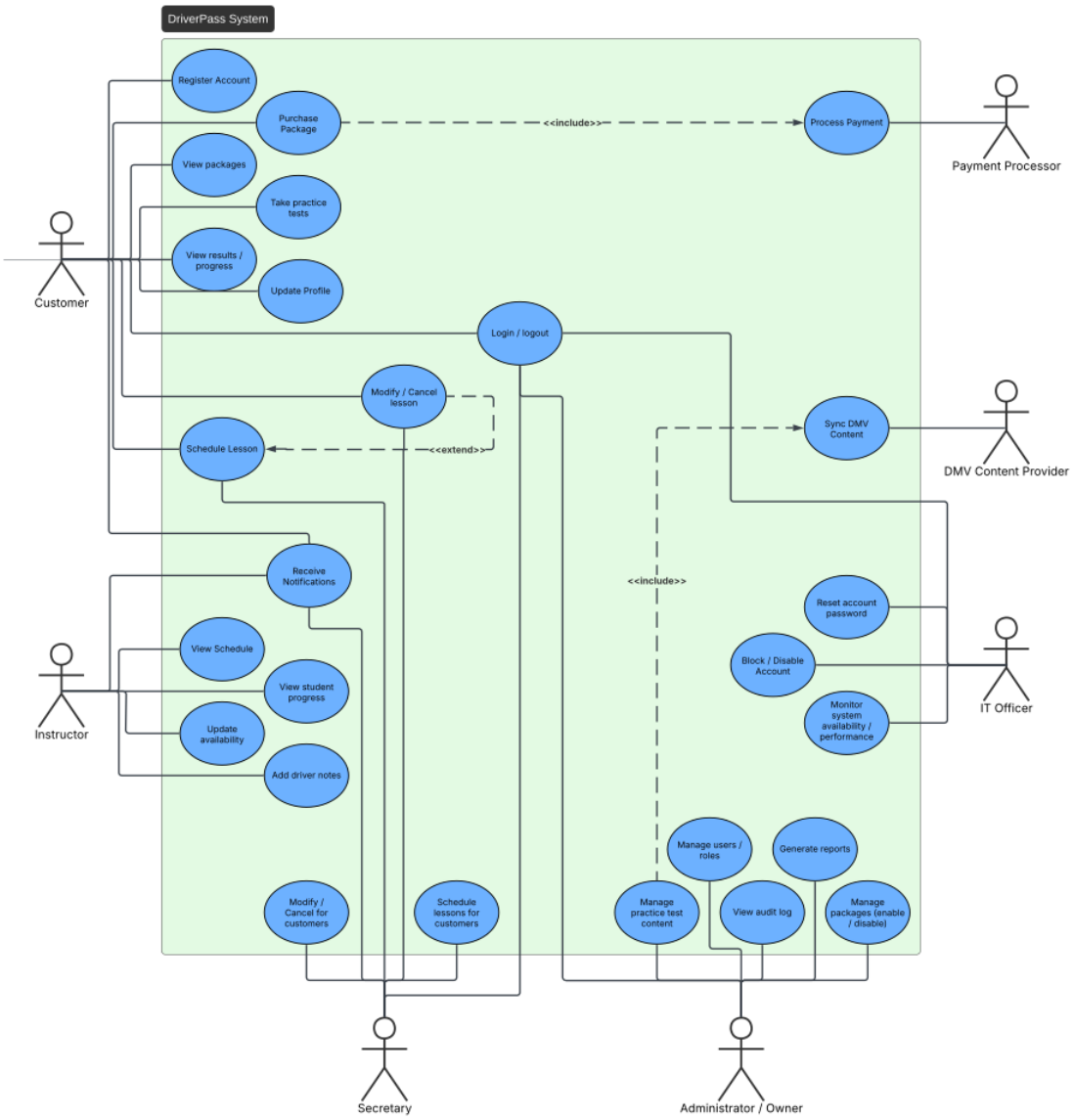

Figure 1. DriverPass use case diagram

UML Activity Diagrams

Two activity diagrams were created to showcase major workflows within the DriverPass system:

Activity Diagram 1: Schedule Lesson

The activity diagram in Figure 2 shows how a customer books a driving lesson using the DriverPass system. First, the customer logs in and chooses a lesson. The system then checks if an instructor is available. If the chosen time is open, the system schedules the lesson and confirms the booking. If not, the customer is asked to pick a different time.

This diagram uses swimlanes to clearly separate each role and highlights the system's validation steps before confirming a booking.

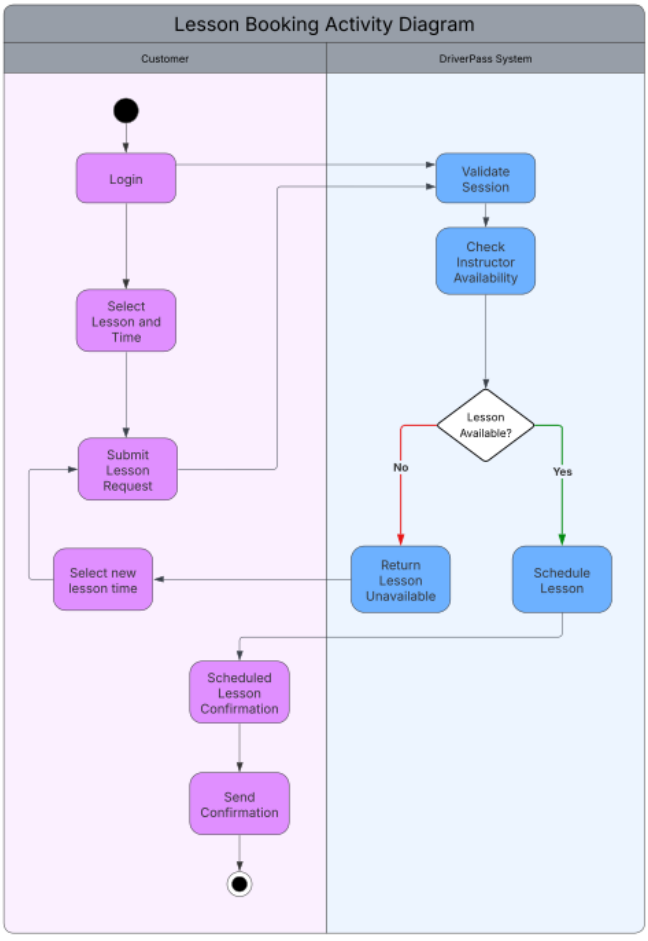

Figure 2. Activity diagram lesson booking

Activity Diagram 2: Purchase Package / Payment Processing

The activity diagram (Figure 3) models the purchasing workflow after a user selects a lesson package. The DriverPass system creates a purchase order and sends a payment processing request to an external payment processing system. Based on the payment authorization result, the DriverPass system will either activate the package to the customer and record the transaction or will prompt the customer to retry the payment process.

The diagram steps through the interaction between the internal system processes and external services while maintaining clear decision logic.

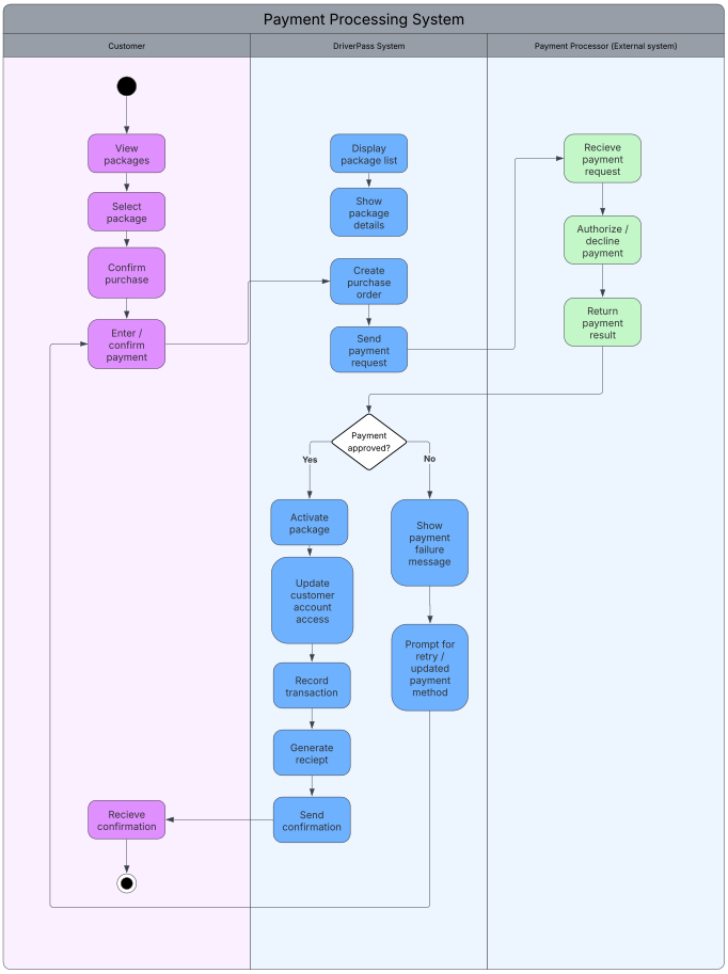

Figure 3. Activity Diagram payment processing system

UML Sequence Diagram

The sequence diagram (Figure 4) models the Purchase Package use case by showing the sequence of interactions between the Customer, DriverPass System, and the external payment processing system.

The sequence diagram shows the message flow starting with package selection and purchase confirmation. The system creates a purchase order, sends a payment authorization request, receives approval, records the transaction, activates package access, and sends confirmation to the customer.

This diagram is useful for developers, as it shows the timing and communication between components. This helps show how system responsibilities are distributed across the DriverPass system.

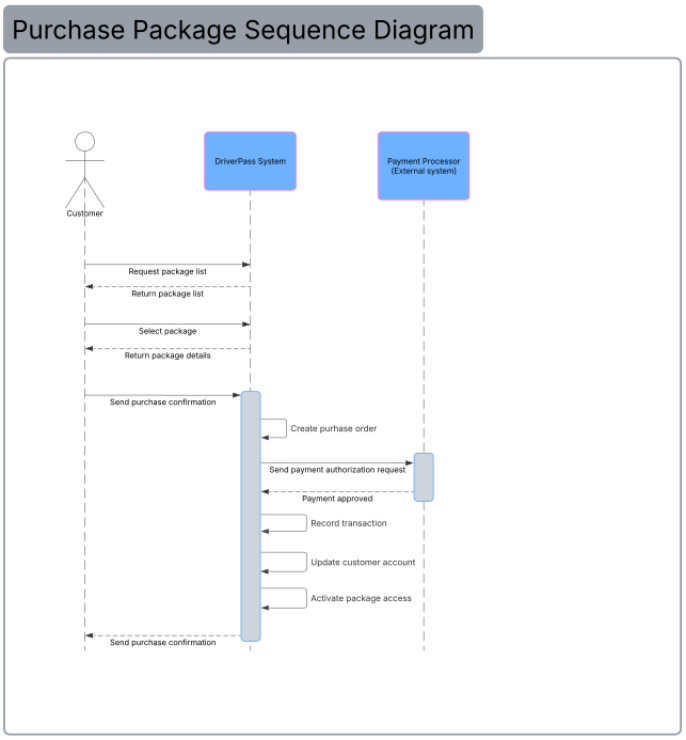

Figure 4. Purchase package sequence diagram

UML Class Diagram

The UML class diagram below (Figure 5) defines the structural design of the DriverPass system, showcasing the key classes.

Connections between classes model real-world interactions, for instance, a customer scheduling lessons, instructors recording progress, and administrators managing the system's content. The class diagram supports scalability by organizing system behaviors around modular objects.

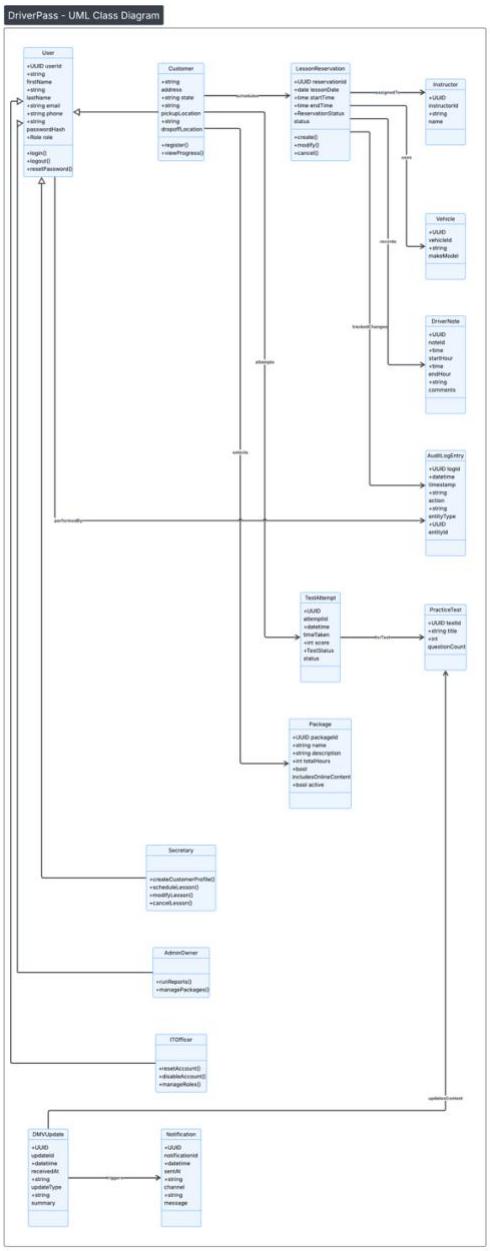

Figure 5. UML class diagram

Southern New Hampshire University

Technical Requirements

Summary:

The DriverPass Technical Design is driven by functional and non-functional requirements from the Business Requirements Document created in Project One. DriverPass is a cloud-based web application that allows individuals (Customers, Instructors, and Administrators) to access training materials, schedule lessons, and track individual progress on any Internet-enabled device. Browser-based access provides users with ease of use and minimal installation.

Scalable cloud computing technology delivers DriverPass with continuous uptime, secure data storage, and dependable integrations with third-party services while processing payments and providing DMV content updates. Role-based access control and authentication protect all sensitive user and payment data while ensuring that each User type has only the access to functions required for their roles. Performance and reliability features include rapid lesson scheduling, accurate data synchronization, and system reliability during peak operational hours.

The technical requirements for DriverPass will ultimately help fulfill their primary business goal of providing a reliable, secure, scalable online learning & scheduling platform for students to prepare for employment with DriverPass. Additionally, the Technical Requirements allow DriverPass to expand as an organization.

Hardware Requirements:

- Cloud-based server infrastructure to support scalable application deployment.
- Storage system for lesson data, user accounts, and training material.
- End-user devices should include web-based devices and native devices capable of running modern web browsers.

Software Requirement:

- A modern web application framework capable of supporting responsive, browser-based access.
- The application must provide security for users through a role-based authentication and authorization service.
- Relational database management system for storing user data, lesson schedules, and training records.
- API integration for an external payment processor and DMV content provider updates.
- Monitoring and logging software to track system activity and performance.

Infrastructure Requirements:

- Cloud hosting platform providing:
    - 24-hour a day, 7 days a week availability.
    - Automatic scaling during peak usage.

    - Backup and disaster recovery services.
    - Usage and security response notifications.
- Secure HTTPS communication for all user interactions.
- API gateway for external service integrations.
- CI/CD (Continuous integration/continuous deployment) pipelines to enable system adaptability.
- Internal analytics system for tracking user journey and experience (user interaction tracking only).

Security Requirements:

- Encrypted credentials to authenticate users under the OAuth 2.0 protocol.
- Access to the system is limited through role-based permissions.
- Payment processing via third-party processors to provide secure handling of payment information with PCI compliance.
- Audit log for monitoring administrative changes and scheduling.
- Encryption will be used for data both in transit and at rest.

Performance Requirements

- Support for concurrent users without any performance degradation
- Quick response times for scheduling and practice testing operations
- Reliable synchronization of DMV content updates (potential CMS system)
- System monitoring to detect availability or performance issues.