DriverPass — System Design Document

Shadab Chowdhury

CS 255

October 19, 2025

Table of Contents

1. Executive Summary

2. System Overview

3. UML Diagrams

   3.1 Use Case Diagram

   3.2 Activity Diagrams

   3.3 Sequence Diagram

   3.4 Class Diagram

4. Technical Requirements

   4.1 Software & Tools

   4.2 Infrastructure & Deployment

   4.3 Security Model

   4.4 Nonfunctional Requirements

5. RACI & Roles/Permissions

6. Assumptions, Risks, and Mitigations

7. Glossary

8. References

1. Executive Summary

DriverPass needs a secure, easy-to-use system that allows students to learn online, schedule in-car lessons, and track progress, while staff and administrators manage accounts, packages, and reporting. This document presents a blended object/process design with UML diagrams and detailed technical requirements suitable for implementation.

2. System Overview

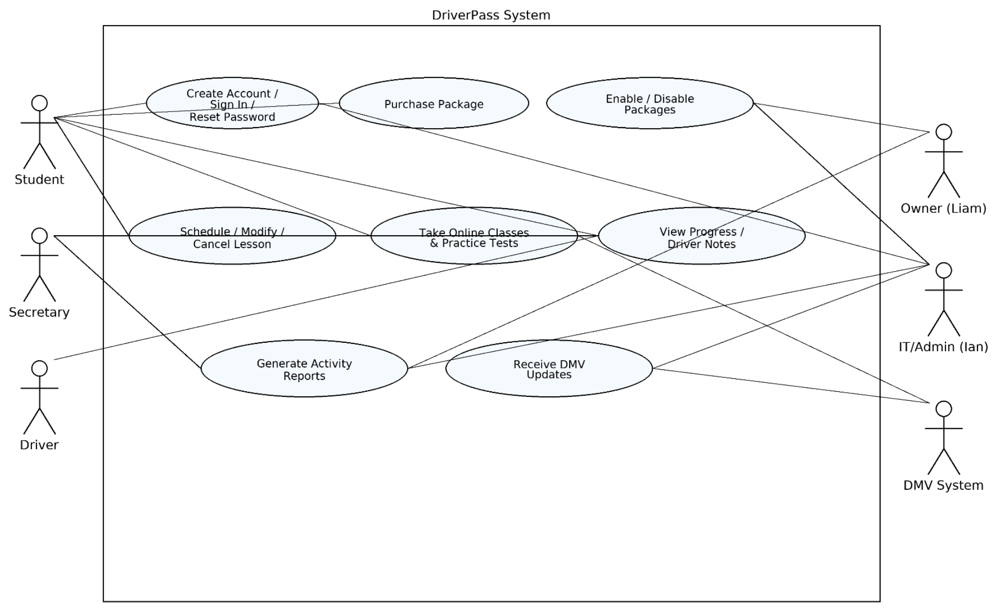
The system provides four pillars of functionality: (1) student learning & assessments, (2) reservation-based lesson scheduling, (3) package catalog and purchase management, and (4) administration & reporting. The design supports web and mobile clients, with a modular backend and a secure data layer.

3. UML Diagrams

3.1 Use Case Diagram

Actors: Student, Secretary, Driver, Owner (Liam), IT/Admin (Ian), DMV System. Key use cases include: account management, package purchase, lesson scheduling, online coursework & practice tests, progress/notes, reports, and DMV updates.

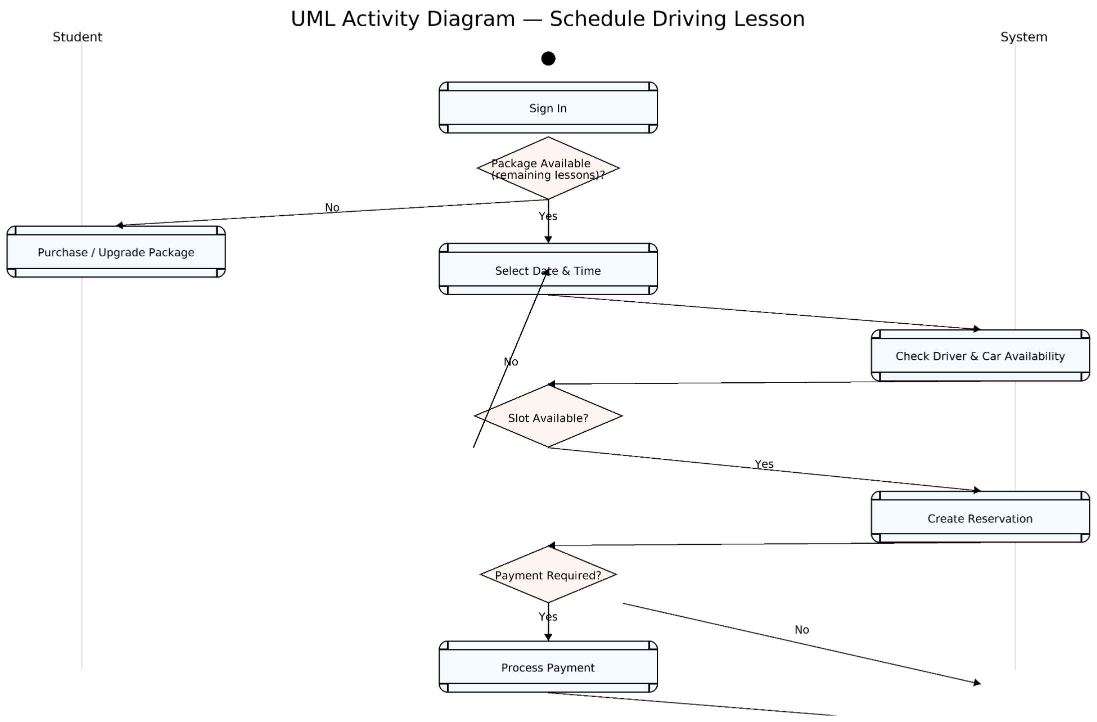
DriverPass — UML Use Case Diagram


DriverPass System
Create Account / Sign In / Reset Password
Purchase Package
Enable / Disable Packages
Student
Owner (Liam)
Schedule / Modify / Cancel Lesson
Take Online Classes & Practice Tests
View Progress / Driver Notes
Secretary
IT/Admin (Ian)
Generate Activity Reports
Receive DMV Updates
Driver
DMV System

3.2 Activity Diagrams

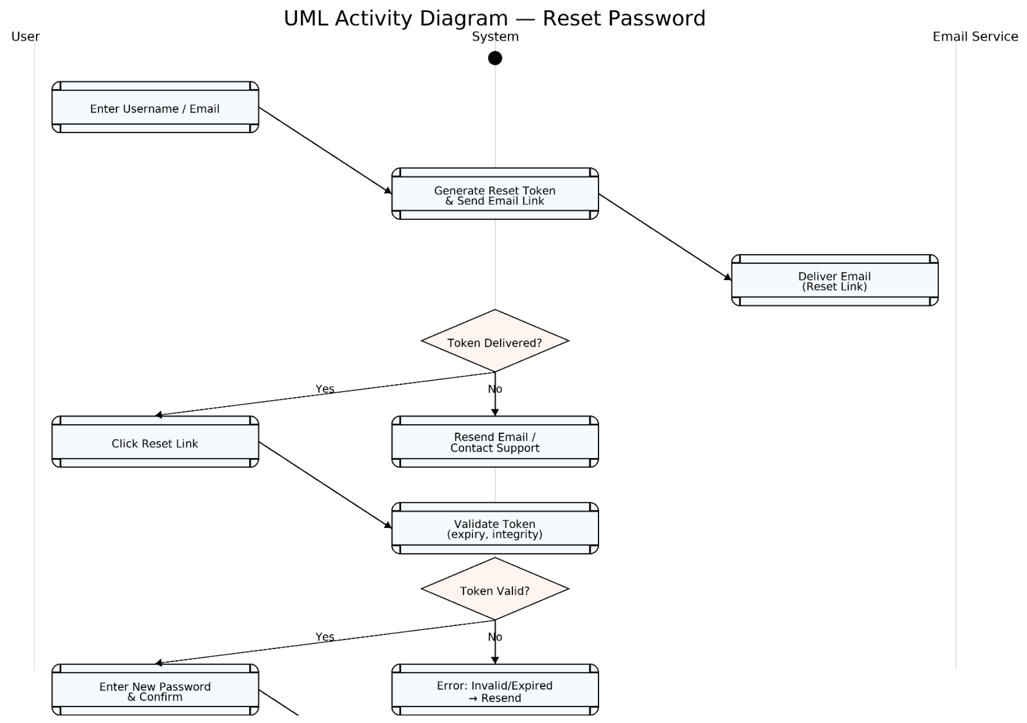
Activity #1 — Schedule Driving Lesson

Start → Sign in → Choose package/remaining sessions → Pick date/time → System checks driver/car availability → Confirm reservation → Send confirmation notification → End.


UML Activity Diagram — Schedule Driving Lesson
Student
System
Sign In
Package Available (remaining lessons)?
No
Yes
Purchase / Upgrade Package
Select Date & Time
Check Driver & Car Availability
No
Slot Available?
Yes
Create Reservation
Payment Required?
Yes
No
Process Payment

Activity #2 — Reset Password

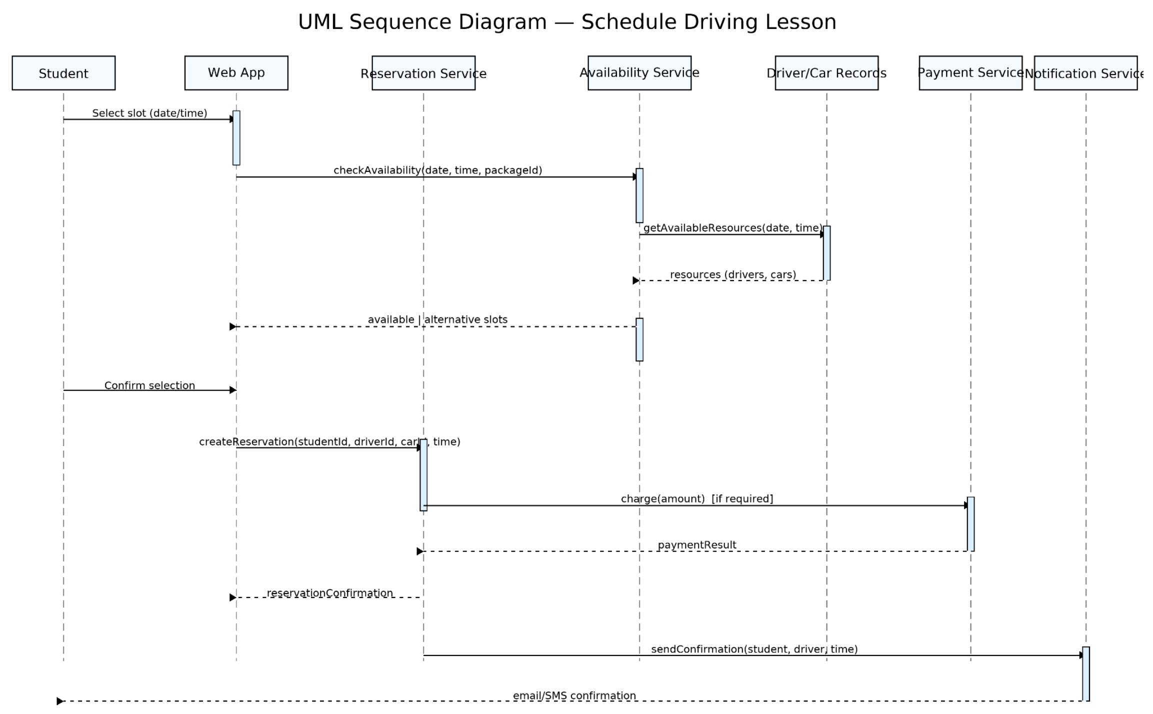
Start → Enter username/email → System sends secure token → User opens link → Enter new password → Validate policy → Update credentials → Confirmation → End.


UML Activity Diagram — Reset Password
User
System
Email Service
Enter Username / Email
Generate Reset Token & Send Email Link
Deliver Email (Reset Link)
Token Delivered?
Yes
No
Click Reset Link
Resend Email / Contact Support
Validate Token (expiry, integrity)
Token Valid?
Yes
No
Enter New Password & Confirm
Error: Invalid/Expired → Resend

3.3 Sequence Diagram (Schedule Driving Lesson)

Lifelines: Student → Web App → Reservation Service → Availability Service → Driver/Car Records → Payment Service → Notification Service.

Happy Path: Student selects slot → Web App requests availability → Availability Service queries driver/car → Reservation Service creates booking → (Optional) Payment → Notification dispatched.

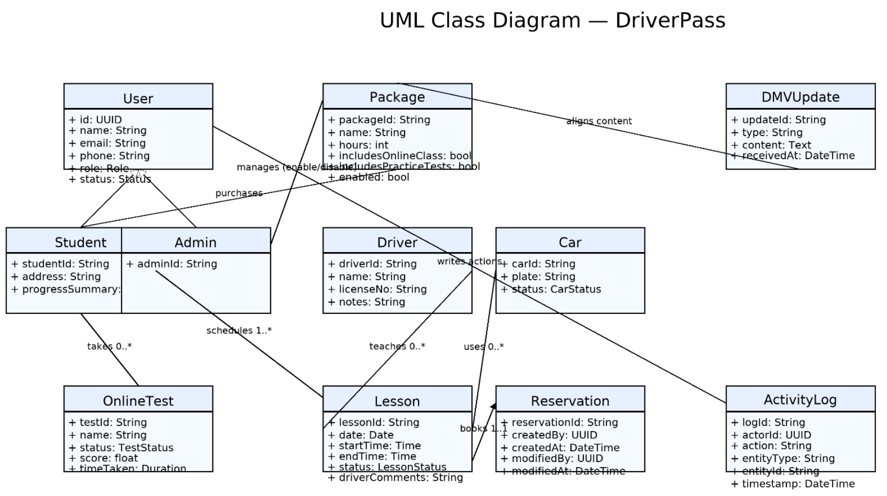

UML Sequence Diagram — Schedule Driving Lesson
Student
Web App
Reservation Service
Availability Service
Driver/Car Records
Payment Service
Notification Service
Select slot (date/time)
checkAvailability(date, time, packageId)
getAvailableResources(date, time)
resources (drivers, cars)
available | alternative slots
Confirm selection
createReservation(studentId, driverId, carId, time)
charge(amount) [if required]
paymentResult
reservationConfirmation
sendConfirmation(student, driver, time)
email/SMS confirmation

3.4 Class Diagram

Core classes and attributes:

- User(id, name, email, phone, role, status)
- Student extends User(studentId, address, progressSummary)
- Admin extends User(adminId)
- Driver(driverId, name, licenseNo, notes)
- Car(carId, plate, status)
- Package(packageId, name, hours, includesOnlineClass, includesPracticeTests, enabled)
- Lesson(lessonId, studentId, driverId, carId, date, startTime, endTime, status, driverComments)
- Reservation(reservationId, lessonId, createdBy, createdAt, modifiedBy, modifiedAt)
- OnlineTest(testId, name, status, score, timeTaken)
- DMVUpdate(updateId, type, content, receivedAt)

• ActivityLog(logId, actorId, action, entityType, entityId, timestamp)



Notes:
— Generalization arrows: Student/Admin inherit from User.
— Lesson links Student, Driver, and Car; Reservation confirms a Lesson.
— Admin manages Packages; Students purchase Packages and take OnlineTests.
— DMVUpdate aligns learning content; ActivityLog records user actions.

4. Technical Requirements

4.1 Software & Tools

• Backend: Java/Spring Boot or C#/.NET 8 (REST), MVC/MVVM
• Frontend: React or Angular (responsive)
• Database: MySQL/PostgreSQL/Azure SQL; ORM (JPA/EF Core)
• Auth: RBAC (Student, Secretary, Driver, Admin/IT, Owner), password policy, optional MFA
• Integrations: DMV updates via webhook/API; email/SMS notifications
• Dev tooling: Git, CI/CD, Lucidchart for UML exports (PNG)

4.2 Infrastructure & Deployment

• Cloud (AWS/Azure/GCP) with managed DB, automatic backups, monitoring
• Environments: Dev, Test, Prod; blue/green deployments; CDN for assets
• Scalability: autoscaling group and load balancer for peak booking windows

4.3 Security Model

• Transport: HTTPS/TLS; Data at rest: encrypted volumes & backups
• Identity: salted password hashing, lockout, optional MFA; secure, tokenized password reset

• Access: RBAC with least privilege; audit trail for create/modify/cancel reservations and account changes
• AppSec: input validation, parameterized queries, CSRF/XSS protections, OWASP Top 10 review

4.4 Nonfunctional Requirements

• Availability: ≥ 99.5% uptime; RPO ≤ 24h, RTO ≤ 4h
• Performance: typical page ≤ 2s; schedule lookup ≤ 1s under normal load
• Scalability: horizontal web tier; read replicas for reporting
• Usability: mobile-friendly UI; clear forms/confirmations
• Maintainability: modular services, code reviews, automated tests

5. RACI & Roles/Permissions

Role	Key Capabilities	R	A	C/I
Student	Enroll, online classes/tests, schedule/cancel lessons, view progress	R		I
Secretary	Assist scheduling, update contact info	R		C/I
Driver	View calendar, add lesson notes	R		I
Admin/IT	Manage users/roles, enable/disable packages, reports	R	A	C/I
Owner	View KPIs/reports,		A	C/I

approve
changes

6. Assumptions, Risks, and Mitigations

• Assumptions: DMV API/webhook access is available; initial packages are defined by IT; internet required for edits.
• Risks: DMV schema changes; peak-time load spikes; account takeover attempts.
• Mitigations: versioned integration, autoscaling and rate limiting, MFA and lockout, continuous monitoring.

7. Glossary

• RBAC: Role-Based Access Control
• MFA: Multi-Factor Authentication
• RPO/RTO: Recovery Point/Time Objective

8. References

DriverPass Interview Transcript; Course templates; Lucidchart exports.