# CS 255 System Design Document Template

## UML Diagrams

### UML Use Case Diagram

A diagram of a driver pass system

AI-generated content may be incorrect.

### UML Activity Diagrams

A flowchart of a program

AI-generated content may be incorrect.A diagram of a test

AI-generated content may be incorrect.

### UML Sequence Diagram

A diagram of a web application

AI-generated content may be incorrect.

### UML Class Diagram

A diagram of a computer

AI-generated content may be incorrect.

## Technical Requirements

The DriverPass system will be delivered as a cloud-hosted, browser-based web application so that customers, instructors, and administrators can reach it from any modern desktop or mobile device. The paragraphs below outline the required hardware, software, tools, and infrastructural services needed to deploy, operate, and maintain the solution.

**1. Hardware & Hosting**

**Cloud infrastructure** – The production environment will reside on a scalable IaaS/PaaS provider (AWS, Azure, or GCP). Using managed services eliminates the need for on-premises servers while giving DriverPass the elasticity to handle peak traffic around DMV-testing seasons.

**Client devices** – End-users need only an internet-connected PC, Mac, tablet, or smartphone with an up-to-date standards-compliant browser; no local installs are required.

**Peripheral needs** – Optional receipt printers or barcode scanners can be attached at training centers for in-office registrations, but they are not mandatory for the core system.

**2. Software Stack**

**Front end** – A responsive Single-Page Application built in React or Angular renders consistently across Windows, macOS, iOS, and Android browsers.

**Back end** – A RESTful API implemented in Java (Spring Boot) or Python (Django) provides business logic, session management, and integration points.

**Database** – Managed relational service (Amazon RDS / Azure PostgreSQL / Cloud SQL) stores user profiles, lesson schedules, payments, and audit logs. The schema must support ACID transactions and automatic daily backups.

**Third-party integrations** – Secure HTTPS endpoints connect to payment gateways (Stripe/PayPal) and, where available, a DMV content API for up-to-date regulations. Manual CSV import remains a fallback when the DMV feed is unavailable.

**3. Performance & Scalability**

**Response time** – Key actions (login, schedule lesson, begin test) must complete within 2 seconds for 95 % of transactions, meeting the performance requirement captured in Project One.

**Availability** – Target 99.9 % uptime through redundant application instances behind a load balancer and multi-AZ database replication.

**Elastic autoscaling** – Compute nodes scale out automatically when CPU or memory thresholds exceed 70 %, ensuring smooth operation during DMV-exam surges.

**4. Security & Compliance**

**Transport security** – All traffic is forced through TLS 1.2+; HSTS headers prevent protocol downgrade attacks.

**Authentication & RBAC** – Unique email-based logins are stored as salted hashes; role-based access control (Customer, Instructor, Employee, Admin) gates every API route.

**Account-lock policy** – After five consecutive failed logins, an account is locked and an administrator alert is triggered to mitigate brute-force attempts.

**Data-at-rest encryption** – AES-256 encryption is enabled for database volumes and S3-style object storage that hosts training videos and documents.

**Audit logging & monitoring** – Centralized logs flow to CloudWatch / Azure Monitor with 30-day retention; anomaly detection alerts the DevOps team within five minutes of suspicious activity.

**5. Maintainability & DevOps Tooling**

**CI/CD pipeline** – GitHub Actions (or Azure DevOps) builds, tests, and deploys containers to staging and production on every main-branch commit.

**Versioned infrastructure** – Terraform scripts define VPCs, subnets, security groups, and managed services, ensuring repeatable, documented environments.

**Automated testing** – Unit, integration, and end-to-end Cypress tests run on every pull request; code quality gates require 80 % minimum coverage before merge.

**Monthly updates** – Routine patching cycles address OS, framework, and dependency vulnerabilities at least once per month, satisfying the nonfunctional “regular updates” requirement.

**6. Disaster Recovery & Data Protection**

**Automated backups** – Daily point-in-time backups with a seven-day retention window; weekly full backups retained for 30 days.

**Recovery Time/Point Objectives** – RTO ≤ 4 hours; RPO ≤ 15 minutes for critical transactional data.

**Geo-replication** – Database replicas in a secondary region allow fail-over if the primary region is unavailable.

**7. System Infrastructure**

Here is a structured overview of the DriverPass system components using a bulleted architectural hierarchy:

**User Interface (Web App)**

* Built using React or Angular
* Responsive across devices

**API Gateway / Web Server**

* Java Spring Boot or Python Django backend
* Manages routes, security, session state

**Business Services**

* Handles lesson scheduling, test scoring, content access
* Integration with payment and DMV APIs

**Data Storage**

* PostgreSQL (relational DB for user/lesson data)
* S3-style object storage for training materials

**Security & Authentication**

* HTTPS/TLS encryption
* MFA and RBAC controls
* OAuth 2.0 (optional for future expansion)

**Monitoring & Logging**

* CloudWatch or Azure Monitor
* Alerts, audit logs, anomaly detection

**Deployment Pipeline**

* GitHub Actions or Azure Pipelines
* Dockerized builds, rolling deployments

**References (APA Format)**

Amazon Web Services. (n.d.). AWS architecture best practices. Retrieved from https://aws.amazon.com/architecture/

Google Cloud. (n.d.). Security best practices. Retrieved from https://cloud.google.com/security/best-practices

Microsoft Azure. (n.d.). Well-Architected Framework. Retrieved from https://learn.microsoft.com/en-us/azure/architecture/framework/

OWASP Foundation. (n.d.). Top 10 security risks. Retrieved from https://owasp.org/www-project-top-ten/

Stripe. (n.d.). Payment security and PCI compliance. Retrieved from https://stripe.com/docs/security

GitHub Docs. (n.d.). Understanding GitHub Actions. Retrieved from https://docs.github.com/en/actions