DriveMaster: Technical Report

Process, Progress, and Results of the System Design for a Subscription-based Management System for Colombian Driving Academies

David Gerardo Diaz Gomez (20201020087)
Cristian Arturo Parra Gonzales (20201578102)
Daniel Mateo Montoya González (20202020098)

Universidad Distrital Francisco José de Caldas
Faculty of Engineering

October 24, 2025

Contents

List of Tables

1 Abstract

This technical report details the planning, design, and architecture of "DriveMaster," a Software-as-a-Service (SaaS) platform. The system is designed to provide Colombian driving academies with a comprehensive tool for managing students, instructors, schedules, and regulatory compliance for various license categories. The methodology employed includes agile and object-oriented planning artifacts such as the Business Model Canvas, User Stories in Gherkin format, User Story Mapping, and CRC Cards. The architectural design is defined using the C4 model, detailed Class and Deployment Diagrams, and Business Process Model and Notation (BPMN). The primary outcome is a robust and scalable system blueprint that addresses the core business needs of B2B clients while ensuring adherence to national regulations (RUNT/MINTRA).

2 Introduction

Driving academies in Colombia face significant administrative challenges in managing student progress, scheduling resources, and ensuring compliance with evolving national regulations. The lack of integrated digital tools often leads to inefficiencies, errors in tracking mandatory hours, and difficulties in generating legally required certificates.

The "DriveMaster" project addresses this problem by proposing a centralized, subscription-based SaaS platform. The system aims to automate and streamline the core operations of driving academies, from student registration and payment tracking to final certification. The objective of this report is to formally document the foundational artifacts that define the project's scope, requirements, and technical architecture, serving as a definitive guide for the development, testing, and deployment phases.

3 Literature Review

The primary driver for this project is not academic literature but the specific regulatory framework governing driving education in Colombia. The key "literature" consists of norms and technical resolutions issued by the Ministry of Transport (*Ministerio de Transporte*) and the Single National Transit Registry (*Registro Único Nacional de Tránsito - RUNT*). These regulations dictate:

- The mandatory number of theoretical and practical hours for each license category (e.g., A1, B1, C1).
- The required format and data for official course completion certificates.
- The criteria for instructors and vehicles to be considered valid for training.

The DriveMaster system is designed to encapsulate these rules within its business logic to ensure continuous compliance for its client academies.

4 Background

To fully understand the project's context, the following concepts are crucial:

- **Software-as-a-Service (SaaS):** DriveMaster operates on a B2B SaaS model. Academies do not own the software but subscribe to it, paying a recurring fee. This model allows for centralized updates (e.g., regulatory changes) and reduces the IT burden on the client.
- **Multi-tenancy:** The architecture must support multiple academies (tenants) using the same application instance, with their data securely segregated.

- **Microservices-based Architecture:** The project mentions distinct backends (Java for authentication, Python for business logic), suggesting a microservices approach. This allows for independent development, scaling, and technology choices for different parts of the system.
- **Testing-Driven Approach:** The project explicitly plans for unit (JUnit, pytest), acceptance (Cucumber), and performance (JMeter) testing, indicating a focus on quality assurance from the initial stages.

5 Objectives

The primary objectives of the DriveMaster project are:

1. To develop a system that automates the registration and progress tracking of students according to Colombian regulations.
2. To provide a robust scheduling module that manages the availability of instructors and vehicles.
3. To generate legally compliant course completion certificates automatically when a student meets all requirements.
4. To build a scalable and secure multi-tenant SaaS platform capable of managing multiple driving academies.
5. To offer distinct user interfaces and functionalities tailored to four key roles: Academy Secretary, Instructor, Student, and System Administrator.
6. To ensure system reliability and correctness through a comprehensive, multi-layered testing strategy.

6 Scope

In Scope:

- Management of the student lifecycle: registration, progress tracking, payment logging, and certification.
- Management of instructors and vehicles, including assignments and availability.
- A self-service portal for students to view progress and schedule classes.
- A SaaS administration module for managing academy subscriptions and global regulatory rules.
- Generation of reports for billing and student progress.
- Secure authentication and role-based access control.

Out of Scope:

- Direct integration with government financial or payment gateway systems (payments are logged, not processed).
- Direct, real-time integration with RUNT/MINTRA systems. The interaction is simulated based on their required data formats.
- Human Resources (HR) management features for the academy (e.g., payroll).
- Advanced accounting features beyond basic payment logging and usage-based billing reports.

7 Assumptions

- The regulatory requirements for hours and certification, while subject to change, are well-defined and can be modeled in the system.
- Academies have reliable internet access to use the cloud-based SaaS platform.
- The user roles defined (Secretary, Instructor, Student, Admin) accurately represent the primary actors in a typical driving academy's workflow.
- Simulated services for RUNT/MINTRA are sufficient for the academic and functional purposes of this project.

8 Limitations

- The system has been validated in a simulated pre-production environment. Full-scale load and performance testing with real-world data from multiple concurrent academies is pending.
- The system's security measures (e.g., protection against specific cyber-attacks) are designed based on best practices but have not undergone a formal, external security audit.
- The current design is based on the requirements gathered during the initial analysis phase. Significant changes to these requirements may necessitate a design review.

9 Methodology

The project follows a structured methodology combining agile planning with formal design artifacts. The process is divided into two main phases: Planning and Design.

9.1 Phase 1: Project Definition and Planning

This phase established the project's foundation using the following artifacts:

- **Business Model Canvas (BMC):** Used to define the value proposition, customer segments (B2B), revenue streams (subscriptions), and key activities.
- **User Stories:** Captured functional requirements from the perspective of the end-users. Each story was written in the standard format (*As a..., I want to..., so that...*) and paired with acceptance criteria in Gherkin syntax (**Given/When/Then**) to facilitate automated acceptance testing with Cucumber.
- **User Story Mapping:** Organized user stories visually to prioritize features and define the Minimum Viable Product (MVP) and subsequent development iterations.
- **CRC (Class-Responsibility-Collaborator) Cards:** An object-oriented design technique used to perform an initial identification of the core classes of the system, their primary responsibilities, and their interactions.

9.2 Phase 2: System Architecture and Design

This phase translated the requirements into a concrete technical blueprint. The following models and diagrams were created:

- **C4 Model:** Used to describe the system's architecture at different levels of abstraction. The project includes the System Context diagram (showing how the system fits into its environment) and the Container diagram (showing the high-level technology choices, e.g., Web App, API, Databases).

- **Class Diagram:** A detailed UML diagram illustrating the static structure of the system, including classes, attributes, methods, and the relationships between them (e.g., 'Student', 'CourseRegulation', 'Schedule').
- **Deployment Diagram:** Shows the physical deployment of the system's containers, illustrating how software components are mapped to hardware or cloud infrastructure.
- **Business Process Model and Notation (BPMN):** Detailed flowcharts modeling the key business processes for the Instructor, Student, and Subscription management workflows.
- **UI Mockups:** Visual prototypes of the user interface for each role, providing a clear vision of the final product's look and feel.

10 Results

The primary result of the design process is a comprehensive specification for the DriveMaster system, ready for the implementation phase.

10.1 Key Planning Artifacts

The User Story Map (Table 1) defined the MVP, focusing on critical features for authentication, student management, scheduling, and SaaS licensing.

Table 1: DriveMaster User Story Mapping Summary

Priority	Activities / Themes
MVP / Critical	Authentication & Access, Student Lifecycle Management, Class Management, SaaS License Management.
Iteration 2 / Medium	Advanced Compliance Updates, Financial Reporting, Vehicle Maintenance Tracking.

The CRC cards identified key domain objects. For example, the **SubscriptionLicense** class was identified as a central component for the SaaS model, responsible for managing an academy's access and collaborating with billing services.

10.2 Architectural Blueprint

The resulting architecture is a modern, distributed system designed for scalability and maintainability.

- **Frontend:** A Single-Page Application (SPA) built with **Angular**, providing a dynamic and responsive user experience.
- **Backend:** A microservices-based approach with:
  - A **Java** service for handling authentication and security-critical operations.
  - A **Python** service for implementing the core business logic, such as regulatory calculations and scheduling.
- **Deployment:** The deployment diagram outlines a cloud-based deployment, likely using containers (e.g., Docker) for each service to ensure scalability and separation of concerns.

The UI mockups and BPMN diagrams provide an unambiguous guide for both frontend developers and backend engineers on the expected user flow and system behavior.

11 Discussion

The chosen methodology and resulting artifacts directly address the project's objectives. The use of Gherkin for user stories (Objective 6) not only clarifies requirements but also creates a direct link to automated acceptance tests, reducing ambiguity and improving quality.

The architectural decision to use separate Java and Python services (Objective 4) is a sound engineering choice. It leverages the strengths of each language—Java for robust, enterprise-grade security and Python for rapid development of complex business rules. This separation supports scalability and maintainability.

The detailed planning artifacts, particularly the User Story Map, ensure that the development effort can be focused on delivering a valuable MVP first, which is crucial for a startup or a new product line. The CRC cards provided an early validation of the object-oriented design, ensuring that responsibilities are well-defined, which is a prerequisite for a clean class structure as seen in the later Class Diagram.

A key implication is that the system is designed for change. The ‘CourseRegulation‘ class and the "Update Global Regulation" user story show that the system is built with the expectation that legal requirements will evolve, a critical feature for long-term viability in a regulated industry.

12 Conclusion

The definition and design phases of the DriveMaster project have been successfully completed. A comprehensive set of artifacts has been produced, providing a clear and detailed blueprint for the system's implementation. The project is well-defined, with clear objectives, scope, and a robust, modern architecture designed for scalability, maintainability, and regulatory compliance.

This technical report establishes a solid foundation for the development phase. The alignment between requirements (User Stories), design (CRC, Class Diagrams), and architecture (C4, Deployment Diagrams) gives confidence that the final product will meet the specified needs of Colombian driving academies.

13 References

The following technologies and standards were referenced in the creation of this design:

- Gherkin Syntax for Behavior-Driven Development. `https://cucumber.io/docs/gherkin/reference/`
- Business Process Model and Notation (BPMN). `https://www.omg.org/bpmn/`
- The C4 model for software architecture. `https://c4model.com/`

A Glossary

BPMN Business Process Model and Notation. A graphical representation for specifying business processes in a business process model.

C4 Model A lean graphical notation technique for modelling the architecture of software systems at different levels of abstraction.

CRC Class-Responsibility-Collaborator. A brainstorming technique for designing object-oriented software.

Gherkin A business-readable, domain-specific language that lets you describe software's behavior without detailing how that behavior is implemented.

MVP Minimum Viable Product. A version of a product with just enough features to be usable by early customers who can then provide feedback for future product development.

RUNT Registro Único Nacional de Tránsito. Colombia's Single National Transit Registry.

SaaS Software-as-a-Service. A software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted.