DriverPass

# **CS 499 Project Software Design Document Enhancement**

Version 1.0

## Table of Contents

[**CS 499 Project Software Design Document Enhancement** 1](#_Toc196069308)

[Table of Contents 2](#_Toc196069309)

[Document Revision History 2](#_Toc196069310)

[Executive Summary 3](#_Toc196069311)

[Design Constraints 3](#_Toc196069312)

[Technical Requirements 4](#_Toc196069313)

[System Architecture View 5](#_Toc196069314)

[Component Diagram 5](#_Toc196069315)

[Class Diagram 6](#_Toc196069316)

[Use Case Diagram 7](#_Toc196069317)

[Activity Diagrams 8](#_Toc196069318)

[Sequence Diagram 10](#_Toc196069319)

[API Endpoints 11](#_Toc196069320)

## [Document Revision History](#_heading=h.lnxbz9)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 0.1 | 3/20/2025 | Dorrian Robinson | Imported existing documents from original design document. Added executive summary & design constraints. |
| 0.2 | 4/2/2025 | Dorrian Robinson | Added endpoints. Updated design constraints. |
| 0.3 | 4/12/2025 | Dorrian Robinson | Added authentication and payment diagrams. |
| 1.0 | 4/20/2025 | Dorrian Robinson | Cleanup & revision. Minor update to technical requirements. |

## [Executive Summary](#_heading=h.35nkun2)

The Driver Pass system is designed to provide driver training services to help customers prepare for and pass their DMV driving tests. The system will offer online classes, practice tests, and on-the-road training capabilities. Driver Pass aims to provide quality training through various packages tailored to customer needs.

The platform will be cloud-based, allowing access from anywhere via web browsers and mobile devices. Key functionalities include user account management, training package selection, online learning materials, practice tests, driving lesson scheduling, and comprehensive reporting. The system will incorporate role-based access control to ensure appropriate system usage by administrators, IT officers, secretaries, drivers, and customers.

## [Design Constraints](#_heading=h.1ksv4uv)

1. **Cloud-Based Deployment** - The system must be deployed in the cloud to reduce the technical burden and ensure accessibility from anywhere.
2. **Security & Authentication -** Role-based access control required for different user types (admin, IT officer, instructors, customers).
3. **Data Integrity** - System must maintain accurate records of all transactions, changes, and user activities with audit trails.
4. **DMV Compliance** - System must be able to incorporate DMV updates on rules, policies, and test questions when notified.
5. **Scalability** – The system must support business growth and the addition of new features or training packages.
6. **Offline Access** - Limited functionality for accessing (but not modifying) downloaded reports and information offline.
7. **Mobile Compatibility** - Interface must be accessible via both computers and mobile devices.
8. **Data Privacy** - System must securely handle and store customer information including payment details.

## [Technical](#_heading=h.44sinio) Requirements

1. Hardware Requirements

* Cloud / Web servers to reduce technical burden on company and employees
* Database server(s) for storing data

1. Software Requirements

* Hosting: Kubernetes pods running nodes on Linux
* Database System: SQL / Relational Database (i.e., MySQL)
* Security: Common website certificates. HTTPS, SSL/TLS

1. Tools

* Cloud Hosting: AWS or Azure
* IDE: Visual studio and VSCode
* Database Design: SSMS, MySQL Workbench, PGAdmin
* Version Control: Github, sourcetree, bitbucket
* Authentication: Auth0
* Payment: Stripe
* Testing: Postman for API tests and Selenium for UI test automation

1. Security Requirements

* RBAC authentication
* PCI Compliance for payments
* Secure password policies and validation
* Data encryption during transfer (data held on cloud has options for encryption)
* Data backups can be handled by cloud hosting services. Redundancy across regions or zones. Central 1, Central 2 backup vs US West, US East backups

1. Other

* Scalability as product grows
* Modular design for adding features as the product grows
* Automated recovery systems
* Performance Monitoring

## [System Architecture View](#_heading=h.44sinio)

### Component Diagram

A screenshot of a computer screen

AI-generated content may be incorrect.

The system architecture is comprised of components: Client, server, and database.

1. The Client Component
   1. Browser: Used to interact with the system
   2. Client Session: Manages session data for a user (i.e., cart management)
   3. Dashboard: Displays training progress, reservations, and user information
   4. Learning Portal: Provides access to classes and practice tests
2. Server Component
   1. Reservation System: Manages driving appointments
   2. DMV Integration: Receives DMV updates from an admin
   3. Report Generator: Creates system and user reports
3. Services
   1. Auth0 Service: Manages user authentication and assigns tokens
   2. Stripe Service: Handles payment transactions
4. Database Component
   1. Cloud Database: Stores all application data
   2. Database Backups: Zone or regionally redundant backups

### Class Diagram

A computer screen shot of a computer flow chart

AI-generated content may be incorrect.

### Use Case Diagram

*A screenshot of a computer

Description automatically generated*

### Activity Diagrams

*A diagram of a company

Description automatically generated*

A diagram of a process

Description automatically generated

### Sequence Diagram

A screenshot of a computer

Description automatically generated

## [API](#_heading=h.2jxsxqh) Endpoints

| **Method** | **Purpose** | **URL** | **Notes** |
| --- | --- | --- | --- |
| **GET** | Retrieves user profile | /api/users | Requires auth token |
| **PUT** | Update user information | /api/users | Requires auth token |
| **GET** | Retrieves available packages for purchase | /api/packages |  |
| **GET** | Retrieves details for a specific package | /api/packages/{packageId} | Requires packageId as a query parameter |
| **POST** | Creates a reservation with an instructor | /api/reservations | Requires auth token |
| **GET** | Retrieves all reservations for the current user | /api/users/reservations | Requires auth token |
| **PUT** | Updates an existing reservation | /api/reservations | Requires auth token |
| **DELETE** | Cancel reservation | /api/reservations | Requires auth token |
| **GET** | Retrieves all available timeslots | /api/reservation/timeslots/{date} | The date query parameter is optional for filtering slots |
| **GET** | Retrieves the user’s available classes. | /api/classes/ | Requires auth token |
| **GET** | Retrieves the user’s available practice tests | /api/tests | Requires auth token |
| **POST** | Submits the user’s test | /api/tests/submit | Requires auth token |
| **POST** | Processes the purchase of a package | /api/payments | Requires auth token. Uses stripe. |
| **GET** | Generates a user’s activity report | /api/reports/activity | Admin access only |
| **GET** | Generates a user’s reservation report | /api/reports/reservation | Admin access only |
| **POST** | Authenticates a login | /api/auth/login | Uses auth0 |
| **POST** | Registering a new user | /api/auth/register | Uses auth0 |
| **POST** | Send an email to reset the user’s password | /api/auth/reset-password | Uses auth0 |

## 