# CS 255 Business Requirements Document

## System Components and Design

### Purpose

DriverPass seeks to provide a way to provide drivers with better means of refining their driving skills to prepare for their driving test at the Department of Motor Vehicles (DMV) as too many students are failing their driving tests.

### System Background

DriverPass aims to solve the issues of too many students failing the driving exam on their first attempt. The system should provide users with the tools to better prepare for their driving tests including written tests, scheduled driving sessions, and feedback. This system aims to fill a void in the marker for providing student drivers resources to prepare for their test.

### Objectives and Goals

The system should be able to meet the needs of the end users to meet the goals set by Driver. In addition to serving the end users, the internal users should be provided with the tools to maintain the status of entities within the system. The IT maintenance should be minimal to allow client focus on business processes.

* Online access to data by End Users and Clients
  + Desktop Website
  + Mobile Website
* Role-Based Access Control – Employees and users should have certain limitations placed on their access to the system
* Tracking – user actions should be traceable to determine errors or steps taken
* Reporting – Employees should be capable of generating reports in an exportable format
* Reservation System – End users and certain employees should be able to book reservations for a customer user
* Customizable Packages – The employees should be able to at least disable packages, consideration should be handled in the design for the prospect of package maintenance by a user.
* DMV Integration – The application must retrieve information from the DMV to notify employee users action is required on site content
* Cloud-based Infrastructure – The application must be low maintenance and available on the cloud

## Requirements

### Nonfunctional Requirements

#### Performance Requirements

* The system should be web-based with access from desktop and mobile through a web browser
* The response time of the system should be as follows:
  + For interacting with objects and components in the UI the response time is not to exceed 100ms
  + Data create, read, update, and delete operations should load fully for the page in UI sections not to exceed 1 second
  + Report generation for download should not exceed 10 seconds
* The system should be updated as follows:
  + For Critical vulnerabilities that require code changes, a plan to address should be determined within 1 week of discovery, with the plan to implement immediately following
  + For critical vulnerabilities that do not require code changes, resolution should be addressed within 1 week of discovery, unless otherwise impossible
  + For installed dependencies, updates should be planned within two weeks of discovery and plan to implement within 1 month of conclusion of planning
  + System updates should be audited on the cloud infrastructure for user required action on a bi-weekly basis, with plans to implement based on the severity of the required action at client discretion
  + Packages within the system that require code changes should be updated quarterly at client cost and discretion
  + Content updates requiring code changes should be updated quarterly unless otherwise noted at client cost and discretion
  + Content updates that do not require code changes should be made entirely at client discretion

#### Platform Constraints

* The server side of the application should be run on a server focused Linux distribution
* The client side of the application should be capable of running on Microsoft Edge, Mozilla Firefox, Google Chrome, and Safari with versions not to exceed 6 months of age from project start on Windows, macOS, Debian-based Linux distributions, Android, iOS operating systems
* A Postgres relational database management system to store application and place user restrictions will be required
* A GraphQL API to interact between the RDBMS and clients will be implemented as part of the server

#### Accuracy and Precision

* Users will be distinguished through a UUID assigned by an OIDC provider with association to the user email
* User roles will be distinguished as follows:
  + Student User – Able to create
  + Admin User – Able to manage users allowing credential resets and report generation
  + Internal User – Able to conduct student user actions on behalf of the student user
* Passwords should be 8 character in length at a minimum to include a lowercase alphabetic character, uppercase alphabetic character, numerical character, and a special non-alph/non-numeric character
* Usernames for login will match the user email
* Admins should receive real-time email notifications for the following:
  + Failed login attempts for a single user exceeding 5 tries

#### Adaptability

* Student users, Internal Users, and Admin should be capable of being added, removed, and modified without code modifications
* Platform updates should require minimal code changes and Admin user interaction
* IT admin will need the following access:
  + Generate exportable reports of user actions
  + Update access of Internal and Student Users
  + Reset passwords of users
  + Updating package availability

#### Security

* User logins should require a username and password, which is verified through an OIDC provider
* Users will be required to utilize 2FA through SMS, phone call, email, and authenticator apps
* Communication between the client and server should be handled via HTTPS and disallowed via HTTP
* User payment information should be hashed prior to storage through ASVS level 2 standards
* Once a user has reached 5 login attempts, with no successful attempts, the user will be required to utilize a CAPTCHA prior to additional login attempt submissions
* User request of self-password resets should be an unattended process
* Admin users should be capable of handling password resets for other types of users

### Functional Requirements

* The system will allow Student users to book, modify, or cancel driving lessons.
* The system will allow Internal users and Admin users to book, modify, or cancel driving lessons on behalf of a Student User
* The system will allow Student users to view test results, study material, and lesson feedback
* The system will offer multiple types of packages to users
* The system will accept user payments for services
* The system will be capable of providing Admin users with updates based on DMV changes automatically
* The system will be capable of generating exportable reports with support for filtering for Admin User access
* The system will track Student user information as follows:
  + Scheduled Lessons
  + Subscribed lessons
  + Tests taken
  + Lesson notes
  + Student identification information

### User Interface

* The user interface will display the following information to a student user and internal user:
  + Online test progress
  + User information:
    - First Name
    - Last Name
    - Address
    - City
    - State
    - Zip
    - Phone
    - Email
  + Driver notes viewable by driving lesson
  + Special needs
  + Driver photo for selected lesson notes
  + Student photo
* The user interface should support Student and internal modification of the Special needs and user information
* The user interface should support Driver Notes updates for the assigned driver of a lesson
* The user interface will be responsive for consistent presentation of information on both a Desktop and Mobile display
* Desktop users will utilize a mouse pointer and keyboard to interact with elements
* Mobile users will utilize tap and a mobile keyboard to interact with elements
* Admin users should have a dedicated interface for managing users, resetting user credentials, viewing user actions, and generating reports

### Assumptions

* The system assumes the user will have internet or mobile data access to utilize the system
* The system assumes the user will have a device with a supported web browser for application access
* The DMV system provides an interactable API or Web service for requesting updated information
* Client delivery expectation and budget allow time for requirements shifts

### Limitations

* Customization of packages may require developer intervention with initial system iteration
* Time limitations assume no shifts in requirements put in place by the client
* Client budgeting for allowing scalability of a Cloud infrastructure is reasonable for project budget
* Integration with DMV is heavily dependent on DMV system exposing appropriate data for consumption

### Gantt Chart

A screenshot of a project management

AI-generated content may be incorrect.