# CS 255 Business Requirements Document

## System Components and Design

### Purpose

* DriverPass is the client, and they want their system to provide online practice exams as well as training behind the wheel for students who want their driver’s license.

### System Background

* Driverpass wants the system to provide students learning how to drive with proper online and on-hands experience for the DMV tests. The components needed are a computer, phone, and tablet.

### Objectives and Goals

* System should be able to let students take online classes and exams, as well as in-person training.
  + Use office applications (Ex. Microsoft Excel) to download reports.
  + Allow online access from any device using the system.
  + Allow online and offline data access.
  + Have students create accounts to schedule in-person lessons.
  + Have a tracking system to keep track of the drivers, students, and cars that are being used.

## Requirements

### Nonfunctional Requirements

#### Performance Requirements

*What environments (web-based, application, etc.) does this system need to run in? How fast should the system run? How often should the system be updated?*

* The system should work in both web-based and application environments.
* The load times should be around 2 seconds or less no matter the environment used.
* The system should ideally be updated monthly, with only 2 hours or less of downtime.

#### Platform Constraints

*What platforms (Windows, Unix, etc.) should the system run on? Does the back end require any tools, such as a database, to support this application?*

* Linux or Windows is a good choice for a system to run on, as they support web frontend for most desktop and mobile browsers.
* The backend would require a database for storing user info as well as system logs.

#### Accuracy and Precision

*How will you distinguish between different users?* *Is the input case-sensitive? When should the system inform the admin of a problem?*

* The frontend should handle different users by using session cookies and proper accounts.
* The input to login to accounts, such as username or password would be case-sensitive, but anything else would not need to be case-sensitive.
* The system should give inform the admin of a problem as soon as possible, otherwise give a daily diagnostic check.

#### Adaptability

*Can you make changes to the user (add/remove/modify) without changing code? How will the system adapt to platform updates? What type of access does the IT admin need?*

* User changes would be made in the backend without changing code.
* The application and the browser would have to remain up to date in tandem with each other.
* The IT admin would require access to the database as well as the server running the application.

#### Security

*What is required for the user to log in? How can you secure the connection or the data exchange between the client and the server? What should happen to the account if there is a “brute force” hacking attempt? What happens if the user forgets their password?*

* A password and username/email would be required to log in.
* For security, 2-Factor-Authentication with email or SMS would be optional.
* After enough attempts, the account would be locked in a certain amount of time/indefinitely.
* If a user forgets their password, there would be an option to send an email or SMS to the user with a temporary password, and after logging in, the user will then reset their original password.

### Functional Requirements

*Using the information from the scenario, think about the different functions the system needs to provide. Each of your bullets should start with “The system shall . . .” For example, one functional requirement might be, “The system shall validate user credentials when logging in.”*

* The system shall check the user email address and password when logging in.
* The system shall check 2-Factor-Authentication if the user has it enabled.
* The system shall lock accounts after a threshold of failed logins has been reached.
* The system shall send an email or text with a temp password in case of account lock, or the user forgets their password.
* The system shall update the user info on the backend due to any of the previous scenarios or admin commands.
* The system shall track and schedule appointments following available appointment times and user/admin input.
* The system shall notify the admins when DMV rules change.

### User Interface

*What are the needs of the interface? Who are the different users for this interface? What will each user need to be able to do through the interface? How will the user interact with the interface (mobile, browser, etc.)?*

* The interface needs to adapt to the desktop and mobile application environments, as well as be web-based.
* The users would contain the standard users and admins of the system.
* Standard users would login to find their account with appointment dates and preferences, as well as their account and order history, with the option to make payments.
* The admins would have access to manage standard user accounts and appointments to better suit the standard users’ needs.

### Assumptions

*What things were not specifically addressed in your design above? What assumptions are you making in your design about the users or the technology they have?*

* Most users have an email address or phone number.
* Most users would have proper access to the web and modern computer/phone.
* The DMV’s changes can be tracked in real time.

### Limitations

*Any system you build will naturally have limitations. What limitations do you see in your system design? What limitations do you have as far as resources, time, budget, or technology?*

* There is no way to anticipate changes to browsers.
* There is no way to anticipate the DMV’s changes.
* The front end must be compatible with all the modern-day browsers.
* 15 weeks are needed to complete the project.

### Gantt Chart

A close-up of a chart

Description automatically generated