# CS 255 Business Requirements Document Template

## System Components and Design

### Purpose

*What is the purpose of this project? Who is the client and what do they want their system to be able to do?*

* This project's purpose is to create a solution to the driving instructing industry. DriverPass aims to do this by making a more modern solution to instructing new drivers onto passing the DMV test through online classes and practice exams paired with optional on-the-road experience.
* DriverPass’ Owner, Liam, wants to have a system where students can easily access their websites to enroll and register into their programs through a package/tier system.

### System Background

*What does DriverPass want the system to do? What is the problem they want to fix? What are the different components needed for this system?*

* They want the system to be not only easy to maintain but have a low maintenance and failure rate. They want their system to easily handle user requests by having them fill out forms for registration and contacting them directly after. The system should be able to keep tabs on everyone through analytics, logging who (and when) created, edited, and canceled reservations; They should be able to this online and offline (through phone or in-person appointments).
* Multiple components are required such as a database that has constraints on reservations (making sure it's possible to see when a user’s reservation is, who is driving with them, and what car they are driving with.) An online web application is needed to handle registration by both students and the secretary. An analytic system is needed to track reservations and other data. A view system is required to export the data into various views such as PDF and Excel (.xcl files). A notification system is needed to notify the staff about any reservation changes and any changes to policy, rules, or regulations from the DMV. A package management system is needed to manage and customize packages; most important is being able to disable packages.

### Objectives and Goals

*What should this system be able to do when it is completed? What measurable tasks need to be included in the system design to achieve this?*

* This system should be able to let users create, modify, and cancel reservations and notify DriverPass of when the last change was and by who. The system should be able to log data and be able to be viewed through many mediums such as Excel/PDF. Students should be able to register from a list of packages (which some may be disabled due to any reason) and then pick the time and date they are available. Users should be able to access a few pages such as registration, contacting DriverPass, and a contact student page for employees at DriverPass. The system should notify DriverPass when an update comes from the DMV such as a policy or regulation change or example test/questions come in.
* The measurable tasks to achieve this is:
  + A User Authentication/Interface System
  + An Online Reservation System
  + A Reservation Management System
  + A Database System Aligned with a Model
  + Package Mangement System
  + A Reporting/Analytic System
  + Data Access System
  + A Payment System
  + Cloud-Based Deployment System
  + User Information Input System
  + DMV Observer System
  + Notification System

## Requirements

### Nonfunctional Requirements

*In this section, you will detail the different nonfunctional requirements for the DriverPass system. You will need to think about the different things that the system needs to function properly.*

#### 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 environments that the system needs to be ran in is a web-based application.
* The system should run at a moderate speed, the speed demand will not be a big issue considering that many users will only be taking tests (which are loaded in once). However it is important to know that users may be streaming content from the web-application (and the server) so it is a good idea to keep a bandwidth constraint on our mind.
* The system should be updated at least once per day, somewhere in the morning (depending on where the server’s timezone is). The reason the system needs to update every day is because it is actively listening for new data and rules from the DMV about taking tests and what is actively going to be on the exams.

#### 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?*

* I believe the operating system can be universal but for simplicity and maximum performance and tunability, the system will run on a Linux based operating system (such as Debian, Ubuntu, or Fedora.) This decision is heavily affected by the fact that DriverPass wants the systems to be running on the cloud
* The back end will require various tools in-order to function properly. This set of tools include such as a SQL server (or any database software), a small but robust Load Balancing to distribute network traffic to mitigate bandwidth from video streaming, and various encryption libraries to both authenticate and literally encrypt users information during registration and authentication.

#### 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?*

* I will distinguish users by two values; their ID and their username – both of which will be unique. This will allow for both an easy composite key, and a stable ecosystem to where people can easily be identified without searching an arbitrary number. Each user will have their basic data stored along with their ID such as their name, address, and status.
* The system should inform the admin of a problem when unexpected behaviors arise. Such is the fact when a transaction failed due to some server-side issue, such as Stripe not trusting the back end server. Admins will also be notified when a bad actor attempts to manipulate various components, and unexpected behavior caused from this should be logged and admins should be notified. Finally, Admins should be know about various unauthenticated requests anywhere meaningful, such as local database access, shell access, or to their own account.

#### 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?*

* Depending on how the User class is setup, it is possible to modify User without changing code. If we follow the open-closed principle we may be able to code User to where that it can be open for extension but closed for modification.
* The system will adapt to platform updates using a simple system that either gets notified or checks periodically for an update; when the system is scheduled to be down or a rolling-update, the system will update properly; with backups strictly imposed.
* The IT admin may need DevOPs, Shell, and Database access to the servers to make sure that all operations are running smoothly. The IT admin will need to check all faucets necessary for operation, network diagnostics, bandwidth usage, database logs (both access and usage), web-server information. The IT admin may need to check on the Load Balancing proxy server.) The IT admin will most likely interface with the server(s) and system through a Cloud-Based platform such as AWS or Azure.

#### 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?*

* All that is required for the user to log in is their username(could be email) and password.
* We can secure the connection using various methods, and the most commonly used and industry standard is using simple SSL Encryption throughout the entire web-application. SSL can easily be enabled using the web-server application and forcing all connections to go through port 443 and use a secure, verified certificate.
* If there is a brute force hacking attempt, the user’s account should be immediately locked; both a notification should go to an admin and to the user. The user should get an email letting them know their account is locked and that they need to click the link (or change their password) to unlock it. The bruteforcer (mid-attack) should be subject to slower response times (Rate limited) and Human Reaction times being more persistent in validation.
* If the user forgets their password, a link should be send to their email that allows them to reset their password. At the same time, the IT Admin should also be able to reset passwords easily for employees who forget their passwords.

### 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 allow users to take online classes
* The system shall allow users to take practice tests.
* The system shall allow DriverPass to track reservations
* The system shall notify admins when a user updates a reservations
* The system shall allow file upload and file download for offline work
* The system shall allow Users to create their own reservations
* The system shall allow Users to update their own reservations.
* The system shall differentiate between various Packages.
* The system shall be able to have customizable packages.
* The system shall have the ability to disable packages.
* The system shall allow users to schedule appointments with information about pick up and drop off location.

### 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 needs of the interface is that it needs to show what tests the customer took, it should show what’s in progress and the ones that the customer completed. It should show test name, time taken, score, and status for each test. Other interfaces should be input forms where the user (or assigned secretary) fills out information.
* Users should be able to view their previous tests, take new ones, and review feedback left from the user (driver notes). The assigned Secretary should be able to create and modify appointments and reservations. The other staff may need a page to contact students.
* The user will interact with the interface from a browser using a web app, so things must be catered for all major devices such as mobile, tablet, laptop, and desktop.

### 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?*

* Well somethings that aren’t mentioned is responsive design and the ideas that the design may not be able to translate into another medium. Another idea is user flow, how easy will it be for the user to navigate throughout the application.
* An assumption I am making is that the majority of the users will either be on a laptop or their phone. These two assumptions cause me to focus on responsive design and making sure features aren’t lost to device size

### 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?*

* The system will naturally have limitations, first and foremost its on the cloud which may bring in issues and instability due to factors out of DriverPass’ control. Another limitation I see is that users may struggle accessing certain features due to device limitations.
* We have a technology limit due to the fact that it must be on the cloud, this is not entirely bad because security is covered but things aren’t up to our control. Budget isn’t a limit for us, this project will not require much money to operate. However time may be an issue because many of these features can take weeks at a time for a team. We only have 8 days to design a Class Diagram and less then 2 weeks to build an interface.

### Gantt Chart

*Please include a screenshot of the GANTT chart that you created with Lucidchart. Be sure to check that it meets the plan described by the characters in the interview.*

