# CS 255 Business Requirements Document

Johnny Vu

12/1/2024

## 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?*

* The goal of this project is to design a system for DriverPass that helps students prepare better for their driving tests.
* The system will give students access to online practice exams and allow them to book driving lessons.
* It will also have tools for administrators to manage lessons, track student progress, and handle payments.
* The company, DriverPass, realized that over 65% of students fail their driving tests because they aren’t fully prepared.
* This system aims to give students better tools and resources to pass their exams, both online and through in-person lessons.

### 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?*

* DriverPass wants the system to do a few key things: provide online practice exams, allow students to schedule driving lessons, track student progress, and make payments easy.
* Students should be able to book lessons at their convenience and practice driving-related questions at their own pace.
* For admins, the system should make it easy to manage student accounts, lesson schedules, and payments.
* By offering a mix of online practice and in-person lessons, DriverPass wants to increase the number of students who pass their driving tests.

### 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?*

Once the system is up and running, it should do the following:

* Let students register and create profiles, take practice tests, and book lessons.
* Track student progress and provide feedback on how they’re doing in their tests and lessons.
* Allow administrators to manage lesson schedules, track payments, and view customer data.
* The system should be easy for both students and admins to use, with everything accessible through a simple web interface.

To make this happen, I’ll need to:

* Integrate a secure payment system for students to pay for lessons.
* Create an easy-to-use lesson booking interface for students.
* Ensure data is accurate, and security is tight, especially when handling sensitive info like payment details.
* Key measurable tasks for the system include integrating a secure payment gateway for lessons, implementing an intuitive lesson scheduling interface, and ensuring data accuracy and security for all user inputs.

## 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 system should be web-based so students can use it on any device (phones, tablets, computers).
* Pages should load quickly (within 2 seconds) to make sure the user experience is smooth.
* The system also needs to be updated regularly to reflect new driving tests and lessons available.

#### 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?*

* The system needs to work on modern browsers like Chrome, Firefox, Safari, and Edge. The backend will use a database to store user profiles, lesson schedules, and payment information.

#### 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?*

* To make sure the system runs smoothly, input fields (like student details and payments) will need to be validated for accuracy.
* If there are any inconsistencies in the data entered by users, the system will automatically alert admins to check the data.

#### 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?*

* The system should be flexible enough to allow changes without needing to rewrite large amounts of code.
  + For example, admins should be able to update lesson packages or add/remove users without any issues.
  + It should also be adaptable to updates on platforms and browsers. The IT admin will need full access to the backend so they can make these changes easily.

#### 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?*

* Login details will need to be secure, requiring a username and password.
* To protect sensitive information like payments, data will be encrypted using SSL.
* If someone tries multiple failed login attempts, the system will lock the account to prevent brute-force hacking.
* Users should also be able to securely reset their password through email or other trusted methods.

### 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 should allow students to create profiles, book lessons, take practice tests, and make payments.
* Admins should be able to manage student data, lesson schedules, and payments. They should also be able to generate reports on students' progress.
* The system must send automatic notifications to students and admins about booking confirmations, cancellations, or changes.

### 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 user interface (UI) will be web-based, making it easy for students, instructors, and admins to access.
* Students need a simple, intuitive interface for booking lessons, tracking progress, and taking tests.
* Admins, on the other hand, will need a more detailed interface to manage customer profiles, lesson schedules, and payments.
* Instructors will also need a straightforward interface for viewing lesson schedules and student information.

### 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?*

* The system assumes that all users will have access to the internet and devices that are capable of using the web-based platform.
* Students will provide accurate and complete information when registering for the system.
* The system is expected to handle high volumes of traffic, especially during peak times (e.g., weekends, holidays).
* The first version of the system will focus on the core features like lesson booking and practice tests, while other features like advanced reporting may be added later.

### 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?*

* Initially, the system will only be web-based, with no support for mobile apps.
* The admin dashboard will offer basic features for managing lessons and student profiles, but more advanced tools for reporting will be added later.
* There are time, budget, and resource constraints that may impact the scope of features in the first release.

### 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.*

