**CS 255 Business Requirements Document Template**

Complete this template by replacing the bracketed text with the relevant information.

This template lays out all the different sections that you need to complete for Project One. Each section has guiding questions to prompt your thinking. These questions are meant to guide your initial responses to each area. You are encouraged to go beyond these questions using what you have learned in your readings. You will need to continually reference the interview transcript as you work to make sure that you are addressing your client’s needs. There is no required length for the final document. Instead, the goal is to complete each section based on your client’s needs.

**Tip:** You should respond in a bulleted list for each section. This will make your thoughts easier to reference when you move into the design phase for Project Two. One starter bullet has been provided for you in each section, but you will need to add more.

**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 client wants a new system built to provide driving lessons for customers. Essentially, the idea is that students can create an online account and then create, cancel, and schedule appointments and driving reservations by themselves. There will be different packages that the customer can choose from. However, they may also call the office if they wish, and their secretary would have the ability to do this for them. Each reservation should also have a driver, student, and time and day associated with it, which would also be stored in the database and be viewable online to the student.
* The client wants employees to be able to download the data so that they can work with it in offline mode.
* Overall, the main purpose of the project is just a system that helps provide students with driving lessons and general stuff related to learning driving.

**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?*

* As for the problem DriverPass wants to fix, I think they just want a simple digital solution for providing driving lessons. Doing everything in person or manually is tedious, and they can't really increase the number of the clients they have this way either. So I think all in all, they just want an effective solution to fix the burdens and drawbacks of the extra work it takes to do things without mdoern technology. Moreover, they want more students to do good at driving tests, which helps solve multiple problems.
* This system will have several different components. First, there will need to be a database for the storage of all data, such as driver, student, and time data for reservations. However, the client also wants basic website features, such as support for resetting passwords, and general administration abilities like rights and roles, revoking users' access, and downloading general tracking and activity data. All of this will have to be stored in a database.
* As such, a large component of the system will include features specific to administrators and other staff members. There will be different roles for different staff members. Not every staff member will have access to the same aspects of the system. An IT administrator, for instance, may have the most access to the system as they need to debug and troubleshoot issues that other employees may have. We may also want a type of secretary role, since some users may want to schedule appointments physically when they walk into the store--or when they call the company.
* There will also be a large student component to the system. Students need to be able to register and create a profile. There may be profile customization options, password reset options, and maybe data export features. The student must also be able to schedule and cancel appointments by themselves. There may also be more luxury features that aren't necessarily required but would be helpful to the system, such as printing out appointment dates as well as a study plan, so that the student can reference this data offline.
* There will also be a frontend component because this is a web-based application with a user interface. This frontend component will allow users to interact with the application.
* Finally, the offline component should be considered, even though it will be limited. For example, data should be downloadable in formats such as Excel in case users or administrators want to work with it in offline mode--without being connected to the website. The client was explicit about some portions of the data working fine in offline mode, so support for this must be considered.

**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?*

* When the system is completed, users should be able to register an account, login, and schedule driving reservations. They should also be able to manage their account data on here, and if they lose their password, their should be recovery options to help them recover it.
* Admistrators should have access to special data such as tracking data and user data, and this data should work just fine when they want to export it. This includes the ability to track every user's status, such as how far they are into the program, and their last login date to ensure that everything is working fine. We would also want to track which drivers are associated with which cars and students. We need to know when a car or driver is unavailable, and have a system that handles all of this data. We also don't want to give one driver too many students, and ideally, the system should be as self-sustaining as possible with checks that balance the workload.
* We will also want a system for online classes. The students need to read the class material somewhere, and there should be quizzes, tests, and other assignments that they can complete. This will require implementing a type of LMS (learning management system) to manage all of this. There should also be grading, and instructors should be able to view students' grades and progress.
* We will also want general logging and tracking abilities. For example, if a user claims that their account got hacked, they should be able to open a support ticket. The IT administrator on the other end should be able to look at the database logs and see IP addresses, user agents, and other specific information. There should be timestamps for any changes and operations in the system. If a staff member deletes something from the website, this should also be logged for security and safety reasons. Overall, the more tracking abilities there are, the more effective our support system can be, and it will be easier to troubleshoot and debug potential issues that may occur later on.
* There must also be a class registration system. Students must be able to register for classes by themselves online. However, they may also want to call the company and talk to a secretary that does it for them, so the secretary must have options to schedule appointments for users and customers.
* As for measurable tasks, I think we would first want to start with the groundwork and blueprint of the website. This includes starting with the database and database models themselves. We should figure out everything that will be stored, and create database tables for everything. Then we can begin incorporating core functionality while the design of the website is worked on simultaneously. Core functionality might include starting with an account authentication system so that users can login, gradually building up each time. This would be "measurable" because you start smal land starttart with the foundations, before gradually building up.
* The system should also have support for various packages that the student can choose from, which should ideally be customizable.

**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 client specifically said the system should be web-based and preferably be run over the cloud. So we will want to look into web-based technologies and cloud computing.
* The system should also be updated frequently, as well as receive alerts whenever the DMV has an update, so that they can update their rules, policies, and questions too.
* The required speed wasn't specifically mentioned, but nonetheless, it sounds like the client still wants something very efficient and fast-performing. The website should be performant so it can provide decent user experience for their consumers. It should also be scalable for these reasons, so that it can handle a lot of users online at the same time that are also using the service.

**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?*

* Since the application is web-based, the system should run on Windows, Unix, Mac, mobile, and on browsers from different operating systems.
* The backend requires a database to store all of the different data, such as user data, tracking data, exam data, reservation data, and so on.
* There will also be background tasks that connect to the DMV and make requests often to check for new updates.

**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 distinguish between different users, we'll want to give each user their own username, user ID, and restrict their email to that account. All of this information will be provided by the user when they first register on the website. Every user will also have their own role and status. For instance, there may be a administrator role and then a "current student" role for students currently studying, and then an "ex student" role for students who are finished with the program.
* Input won't be case sensitive in order to provide a seamless experience when users enter their username and scuh. However, it will definitely be case sensitive for passwords for security reasons.
* The system should inform the admin if any of the systems go offline or need updating. This may also happen when the DMV gets an update and the current system needs to be updated.

**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?*

* Yes, you should be able to manage user accounts without changing code. There would be a simple web admin panel interface for this so that administrator accounts can use it easily. Accessing the database itself directly should rarely occur unless absolutely necessary.
* The system will have background tasks that frequently check for updates, such as for when the DMV is updated.
* The IT admin should have full access to user data. This includes resetting passwords, changing user information, and seeing user tracking and activity data, which should be downloadable in an offline format.

**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?*

* The user will just require a username and password to login. 2FA will be optional for security reasons.
* The connection will be secured using basic encryption protocols, so the website will use HTTPS rather than HTTP, and it will have a digitally signed and verified SSL certificate. This would help secure the website against MITM (man in the middle) attacks so other people can't eavesdrop into the netwrok.
* When a brute forcing attempt is detected, the attacker's network should simply be ratelimited so that their requests are rejected. This can be done by incorporating global ratelimiting on all endpoints.
* If a user forgets their password, they can either contact an IT admin to fix it, or they can reset their password manually by sending a password reset link to their email.

**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 sanitize all user input at all times to help prevent against exploits.
* The system shall make sure that usernames and passwords are correct and exist in the database when a user logs in.
* The system shall generate and send a password reset link to a user's email if they click the "Forgot Password?" option.
* The system shall always track the DMV for regular updates in the background and send the IT admin an alert when there is one.
* The system shall have an admin panel that can only be accessed if an account has the Admin role; otherwise, the user's request shall be rejected.
* The system shall be responsive even if you resize your web browser.
* The system should thwart brutefor attacks if an attacker sends too many requests.
* The system shall send the user a 2FA alert and ask for their code if 2FA is enabled on the user's account.

**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 has some basic needs. It should be responsive at all times and be intuitive and simple to use for everyone. This applies to both mobile and browser, and the user will be able to use either one to access the interface.
* The main different users will be visitors, registered users, and admin accounts. Different user accounts will have different roles that let them access different parts of the websites. Admins will be able to access everything.
* Admin accounts will have ability to manage accounts and all of the data, as well as view it. Users will have the ability to create driving reservations and schedule appointments, as well as view general DMV information. Visitors without accounts will just be able to browse the website normally until they register an account to unlock access to more.

**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 main assumption is that every user has access to the internet and either a PC or a mobile phone. The main thing that was not addressed was if someone has no internet and just wants to access their account locally at the DMV. An internal offline version of the network might be worth considering, but it will be tricky to implement.
* In either case, I think it's safe to assume that every user will have access to the internet via some kind of device.
* It is also assumed that every user has an email address in order to be able to register for the website.

**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 will be a couple of limitations. First, we cannot spend too much time making the website work well across every mobile device and operating system. We should start out by adding general, broad support for all systems, before adding more specific support for popular operating systems such as Windows. Otherwise, we would be exhausting resources. Browsers may receive random updates too which we cannot always account for.
* Resources should be distributed equally, starting with the foundations of the website and all core systems required for it to function.
* Specific time estimations were also mentioned, such as 9 days for the database tables and linking. 12 days for starting general work. 22 days for business logic. We should adhere to this and see these as general deadlines.
* Overall, we need 15 weeks to complete the entire project, as shown in the charts below.
* The client's budget wasn't specified, but it's fair to assume that we should keep everything as cheap as possible unless it impacts performance and business logic. As long as the application meets all of the requirements, then that is the main goal, without making the client spend too much money.

**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.*

*Closeup image of the first part (First image)*



*Second part (Second image)*



*Full chart*

