# CS 255 Business Requirements Document

## 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, DriverPass, wants a new system to provide better driver training for their customers. The system should allow customers to take online classes and practice tests and provide on-the-road training if they want. The system should also allow the owner to access data from anywhere, online or offline, and download reports and information that can be worked on at home. Security is a concern, and different employees at DriverPass need different rights and roles. The system should have tracking functionality to monitor changes made to the system and produce an activity report.

### 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?*

* Build a system for driver training for people to help pass their driving tests at DMV
* Online classes and practice tests available
* On-the-road training if required
* Online and offline access to data
* Data access from any device
* Secure user accounts with different roles and access levels
* Ability to track changes made to records
* Ability to print an activity report of user actions
* User can make appointments, cancel, and modify appointments online
* Different packages for training (6, 8, and 12 hours in a car with trainer, in-person lesson to explain DMV rules, and access to online class with practice tests)
* Customizable packages

**System Users:**

* Liam (the owner)
* IT officer (Ian)
* Secretary (answering phone and making appointments)
* Users (for making appointments, canceling, and modifying appointments)

### 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?*

* Design and develop an online platform that allows users to access driver training content, including online classes and practice tests.
* Implement a reservation system that enables users to schedule appointments with a specific driver and car.
* Incorporate tracking and reporting features that allow the system to record and report on user activity, including reservations, cancellations, and modifications.
* Develop an authentication and authorization system that assigns different levels of access and roles to users based on their position in the company.
* Ensure that the system is secure, and that user data is protected.
* Make the system flexible so that the client can customize packages and add or remove modules in the future.

## 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?*

1. **Environments:** The system needs to run in a web-based environment. It should be accessible through web browsers on computers and mobile devices.
2. **System Speed:** The system should provide fast response times to ensure a smooth user experience. It should load pages and process requests quickly to minimize waiting times. Specific response time targets can be defined in consultation with the client based on their expectations and the complexity of the system.
3. **Concurrent Users:** The system should be able to handle a large number of concurrent users without significant delays or slowdowns. This will ensure that multiple users can access the system simultaneously without experiencing performance issues.
4. **System Updates:** The frequency of system updates will depend on the client's requirements and the nature of the system. It is essential to establish a maintenance schedule to address any bugs, security vulnerabilities, or necessary feature enhancements. Regular updates may be scheduled, such as monthly or quarterly, to ensure the system remains up-to-date and efficient.

#### 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?*

1. **Operating Systems:** The system should be designed to run on multiple platforms to ensure compatibility and accessibility for a wider range of users. The recommended platforms may include:
2. Windows
3. Unix/Linux
4. macOS
5. Mobile platforms (Android and iOS)
6. **Web Browser Compatibility:** The system should be compatible with popular web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari. It should be ensured that the system functions correctly and displays properly across different browsers and their versions.
7. **Backend Tools and Database:** The backend of the system may require specific tools and technologies to support its functionality. This may include:
8. Database Management System (DBMS): A database is necessary to store and manage data efficiently. The choice of DBMS will depend on the specific requirements and preferences of the client, such as MySQL, PostgreSQL, Oracle, or MongoDB.
9. Server Environment: The system may require a specific server environment or hosting platform, such as Apache, Nginx, or Microsoft IIS, to support its operations.

#### 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?*

1. **Distinguishing Between Users:** The system should have a user authentication mechanism in place to distinguish between different users. This can be achieved through the use of unique usernames and passwords for each user. Additionally, role-based access control (RBAC) can be implemented to assign different privileges and permissions to different user roles.
2. **Input Case-Sensitivity:** The sensitivity of user input (e.g., usernames, passwords) should be determined based on the specific requirements of the system. Case-sensitivity can be configured to match the desired behavior. For example, if usernames are case-insensitive, the system should treat "JohnDoe" and "johndoe" as the same user.
3. **Problem Notifications to Admin:** The system should have error handling and notification mechanisms in place to inform the admin about any problems encountered. This can be achieved through system logs, email notifications, or dashboard alerts. The system should notify the admin when critical errors occur, such as server failures, database connectivity issues, or security breaches. The timing of these notifications can vary depending on the severity of the problem and the importance of immediate action.

#### 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?*

1. **User Changes without Code Modification:** The system can be designed to allow for user management operations, such as adding, removing, and modifying users, without the need for changing the underlying code. This can be achieved through the implementation of an administrative interface or dashboard that provides authorized personnel (such as the IT admin) with the ability to perform these user management tasks. The admin should have the necessary privileges to manage user accounts, reset passwords, assign roles, and perform other user-related operations.
2. **Platform Updates:** To ensure adaptability to platform updates, the system should be developed using modular and scalable architecture. This allows for easier integration of new features and updates in response to changes in the underlying platform. The use of standard frameworks, libraries, and development practices can also facilitate the adaptation process when platform updates occur. It is important to stay updated with the latest versions and releases of the platforms and technologies used in the system and implement necessary updates accordingly.
3. **IT Admin Access:** The IT admin requires elevated access privileges to perform various administrative tasks and ensure the smooth operation of the system. This may include:
4. Full access to user accounts and permissions.
5. Access to system logs and error reports for troubleshooting purposes.
6. Ability to configure system settings, such as security parameters, backups, and integrations.
7. Permission to manage and configure backend tools and technologies, such as the database management system.

#### 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?*

1. **User Login:** To log in, the user will need to provide their credentials, which usually include a username or email and a password. The system should validate these credentials against the stored user data to authenticate the user. The password should be securely stored in the system using strong encryption techniques, such as hashing with a salt, to protect against unauthorized access.
2. **Secure Connection:** The connection between the client and the server should be secured using encryption protocols, such as HTTPS (HTTP over SSL/TLS), which ensures that the data transmitted between the client and the server is encrypted and protected from eavesdropping and tampering. This requires the use of SSL/TLS certificates and proper server configuration.
3. **Brute Force Protection:** To mitigate brute force hacking attempts, the system can implement measures such as account lockouts or rate limiting. For example, after a certain number of consecutive failed login attempts, the user's account can be temporarily locked for a specific period or until the user verifies their identity through additional means. This prevents automated or manual brute force attacks by limiting the number of attempts within a certain time frame.
4. **Password Recovery:** If a user forgets their password, the system should provide a password recovery mechanism. This typically involves verifying the user's identity through additional information or verification steps, such as sending a password reset link to their registered email address. The user can then follow the instructions to reset their password securely.
5. **Account Security Measures:** To enhance account security, users can be encouraged or required to use strong passwords that meet certain complexity requirements. Additionally, enabling multi-factor authentication (MFA) can add an extra layer of security by requiring users to provide additional verification factors, such as a one-time password sent to their mobile device, along with their regular login credentials.

By implementing these security measures, including strong password encryption, secure data exchange, brute force protection, and password recovery mechanisms, the system can enhance the security of user logins and protect user accounts from unauthorized access or misuse.

### 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.”*

1. The system shall allow users to create a new account.
2. The system shall validate user credentials when logging in.
3. The system shall provide a password recovery mechanism for users who forget their password.
4. The system shall allow users to update their account information, such as email address or profile details.
5. The system shall provide a secure messaging feature for users to communicate with each other.
6. The system shall allow users to upload and share documents or files.
7. The system shall provide a search functionality to allow users to search for specific information within the system.
8. The system shall generate reports based on user activity or system data.
9. The system shall provide an administrative interface for managing user accounts and system settings.
10. The system shall support role-based access control, allowing different levels of access for users based on their roles or permissions.
11. The system shall track and log user actions and system events for auditing purposes.
12. The system shall provide a notification system to inform users about important updates or events.
13. The system shall support multi-factor authentication for enhanced security.
14. The system shall integrate with external systems or APIs for data exchange or functionality, as specified in the scenario.
15. The system shall provide an intuitive and user-friendly interface for users to interact with the system.

### 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.)?*

**User Interface Needs:**

1. **Intuitive and user-friendly design:** The interface should be easy to navigate and use for all users.
2. **Responsive design:** The interface should adapt to different screen sizes and devices for a seamless experience.
3. **Clear and organized layout:** The interface should present information in a structured manner for easy comprehension.
4. **Interactive elements:** The interface should incorporate interactive elements such as buttons, forms, and menus for user interaction.
5. **Visual feedback:** The interface should provide visual cues or feedback to indicate actions, errors, or system responses.
6. **Accessibility:** The interface should be accessible to users with disabilities, adhering to accessibility standards and guidelines.
7. **Consistent branding:** The interface should reflect the branding and visual identity of the company, DriverPass.
8. **Support for multiple languages:** The interface should be able to display content in different languages, catering to a diverse user base.
9. **Support for different devices and platforms:** The interface should be compatible with various devices such as desktops, laptops, tablets, and mobile phones.

**Different Users and their Interface Needs:**

**Customers/Students:**

1. Create a new account
2. Log in to their account
3. View and update their personal information
4. Make reservations for driving lessons
5. View their driving session details and schedule
6. Access and take online classes and practice tests
7. Communicate with DriverPass through messaging or contact forms
8. View their test progress and scores

**Administrator:**

1. Manage user accounts (create, modify, deactivate)
2. Set up and manage driving lesson packages
3. Monitor and track user activity and reservations
4. Generate reports and statistics
5. Communicate with customers/students
6. Manage system settings and configurations

**IT Officer:**

1. Manage system access and security settings
2. Monitor system performance and troubleshoot issues
3. Integrate with external systems or APIs
4. Perform system maintenance and updates

**Interaction with the Interface:**

The interface should be accessible through various platforms such as web browsers (desktop and mobile), ensuring that users can access the system from their preferred devices. Mobile responsiveness is important to cater to users accessing the system on smartphones and tablets. The interface should support user interactions through input forms, buttons, menus, and other standard UI elements. It should provide clear instructions, notifications, and feedback to guide users throughout their interactions with the system.

### 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?*

1. **Payment processing:** The design does not specify the payment processing mechanism for customers/students. It assumes the existence of a payment gateway or integration with a third-party payment service to handle transactions.
2. **Instructor management:** The design does not elaborate on the management of driving instructors. It assumes that there is a separate system or process in place to manage and assign instructors to driving lessons based on availability and qualifications.
3. **User training:** The design assumes that users, both customers/students and administrators, are familiar with basic computer operations and have the necessary skills to navigate and interact with the system.
4. **Internet connectivity:** The design assumes that users have a stable internet connection to access the system. Offline functionality or limited connectivity scenarios were not considered.
5. **Localization:** Although it is mentioned that the interface should support multiple languages, the design does not provide details on how the system would handle localization and translation of content. It assumes that the necessary resources and processes are in place to manage translations.
6. **Integration with external systems:** The design assumes that there may be a need for integration with external systems or APIs for tasks such as sending notifications, accessing driving test information, or verifying user credentials. However, the specific details of these integrations are not addressed.
7. **Mobile application:** The design focuses on a web-based interface but does not explicitly mention the existence of a mobile application. It assumes that the web-based interface is responsive and accessible from mobile devices, but a dedicated mobile app is not discussed.

### 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?*

1. **Scalability:** The design does not address the scalability requirements of the system. As the number of users and driving lessons increase, the system may face challenges in handling the growing workload and concurrent user requests. Additional considerations and optimizations would be required to ensure the system can scale effectively.
2. **Resource limitations:** The design does not explicitly mention the available resources, such as server infrastructure, storage capacity, or bandwidth. Depending on the scale of the system and the number of users, the available resources may impose limitations on the system's performance, capacity, and responsiveness.
3. **Technology limitations:** The design assumes the use of standard web technologies and tools, but the specific technology stack is not mentioned. The choice of technologies may have its own limitations, such as compatibility issues, platform dependencies, or performance constraints. It is important to consider the capabilities and limitations of the chosen technologies during the implementation phase.
4. **Security considerations:** While the design mentions basic security measures, such as password encryption and role-based access control, it does not address all possible security vulnerabilities and risks. Depending on the sensitivity of the data and the regulatory requirements, additional security measures may be necessary to protect user information and ensure data privacy.
5. **User experience limitations:** The design does not provide detailed specifications for the user interface and user experience. Depending on the target users and their needs, there may be limitations in terms of accessibility, usability, and intuitiveness of the system interface. Conducting user testing and gathering feedback would be essential to identify and address any limitations in the user experience.

### 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.*

A screenshot of a calendar

Description automatically generated with low confidence