# 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 purpose of this project is to develop a system for the DriverPass program, which aims to assist users in passing their driving test through practice tests, tutorials, and scheduling real driving exams. The system should streamline the process of booking lessons, tracking student progress, and providing feedback.
* The client is DriverPass, a company focused on helping individuals prepare for and pass their driving tests. They want their system to enable students, instructors, and administrators to interact seamlessly, manage schedules, track progress, and maintain course content effectively.

### 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 help users effectively prepare for and pass their driving tests by providing access to practice tests, tutorials, and the ability to schedule driving lessons. The system should also enable tracking of user progress and allow instructors to manage schedules and provide feedback.

### The problem DriverPass aims to solve is the inefficient and disjointed process of preparing for driving tests, which includes booking lessons, accessing learning materials, and tracking progress. They want to centralize and streamline these activities.

* Components need for the system:
  + **User Management**: Registration, authentication, and account management for students, instructors, and administrators.
  + **Scheduling System**: A calendar for booking and managing lessons.
  + **Learning Modules**: Access to practice tests, tutorials, and study materials.
  + **Progress Tracking**: Features to track student progress and provide instructor feedback.
  + **Content Management**: Tools for administrators to update and manage course content.
  + **Notification System**: Alerts and reminders for scheduled lessons and deadlines.

### 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?*

* **System Capabilities When Completed:**
  + **User Interaction**: Enable students to register, access learning materials, schedule lessons, and track progress.
  + **Instructor Tools**: Allow instructors to manage schedules, provide feedback, and monitor student progress.
  + **Administration**: Support administrators in managing users, updating content, and overseeing system operations.
  + **Notifications**: Provide automated reminders and alerts for lessons and deadlines.
* **Measurable Tasks for System Design:**
  + **User Registration & Authentication**: Implement secure sign-up and login processes.
  + **Scheduling Module**: Develop a booking system for driving lessons with calendar integration.
  + **Progress Tracking**: Create features to monitor and display student learning progress.
  + **Feedback System**: Design a mechanism for instructors to provide and students to receive feedback.
  + **Content Management**: Build tools for updating and maintaining tutorials, practice tests, and other learning materials.
  + **Notification Service**: Set up automated email or SMS alerts for upcoming lessons and important dates.

## 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.*

* The system must be user-friendly and accessible on various devices, including smartphones and tablets.
* It should ensure data security and protect personal information of all users.
* The system must provide a responsive experience, loading practice exams quickly and efficiently.

#### 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 must allow users to take practice exams with an average load time of under 3 seconds.
* Users should receive immediate feedback on their practice exam results after completion.
* Training session schedules should be updated in real-time to avoid conflicts.

The system should be primarily web-based, accessible through a browser to ensure students and instructors can access the platform from any location. This would allow for cross-platform compatibility, so users can access it via desktops, laptops, tablets, and smartphones. A mobile application could also be developed to improve user experience, especially for scheduling on-the-road training sessions and taking practice exams on the go. This could increase convenience for users who rely on mobile devices. The backend should be cloud-based for scalability, data storage, and accessibility. This ensures that users can access the system anytime, from anywhere, without requiring local server infrastructure.

The system should have a load time of 3 seconds or less for the main pages, such as practice exams and scheduling tools. Users expect a fast and responsive system, especially when taking exams or booking training sessions.

For features like booking training sessions or checking exam results, the system should update in real-time to avoid scheduling conflicts and to ensure that students receive immediate feedback. Security updates should be implemented as needed, typically monthly or whenever critical vulnerabilities are discovered, to ensure data protection and regulatory compliance. Minor updates and bug fixes should be rolled out as soon as issues are identified. This could be on a continuous basis, using automated deployments to fix performance issues or errors quickly.

#### 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?*

* **Windows:**
  + The system should be compatible with Windows as it is one of the most widely used operating systems, especially in educational and personal settings.
* **MacOS:**
  + Support for MacOS is important to ensure compatibility with users who use Apple devices, especially in educational institutions.
* **Linux/Unix:**
  + For flexibility and broader support, the system should also be accessible via Linux-based systems, particularly since Linux is popular for web servers and backend development.
* **Mobile Platforms (iOS and Android):**
  + Since many users will likely access the platform on their mobile devices, support for iOS and Android is essential, either via a mobile-optimized web interface or a dedicated mobile app.
* **Browsers:** 
  + The system should be compatible with major browsers such as Chrome, Safari, Firefox, and Edge, ensuring wide accessibility.

Yes, the back end will require a robust database to support the application. Key options include:

* MySQL, PostgreSQL, or Microsoft SQL Server could be used to store structured data, such as student information, exam results, training schedules, and instructor availability.
* Relational databases are ideal for managing structured queries and ensuring data integrity.

#### 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 differentiate between various types of users (students, instructors, and administrators), the system will assign specific roles to each user during account creation. These roles will define the level of access and the actions each user can perform:

* **Students:**
  + Can take online practice exams, schedule training sessions, and view progress.
  + Limited access to personal data and exam/training history.
* **Instructors:**
  + Can view and manage their schedules, access student performance data, and provide feedback.
  + Access restricted to managing their own schedule and students assigned to them.
* **Administrators:**
  + Have full access to all parts of the system, including managing users, generating reports, and overseeing the system’s functionality.
  + Can modify practice exam content and monitor system usage.
* **IT Support:**
  + Limited to technical tasks such as system updates, user support, and troubleshooting.

Role-Based Access Control (RBAC) will be implemented to ensure that each user is assigned the correct role, restricting access to only what is necessary for their tasks. This ensures that sensitive information and features are only accessible to the right users.

Typically, usernames are not case-sensitive to prevent confusion and login issues (treating "JohnDoe" and "johndoe" as the same username). Passwords should be case-sensitive to ensure stronger security, allowing a wider range of character combinations and increasing password complexity. For form inputs like names, addresses, and email addresses, case-sensitivity is generally not required, but email addresses should be treated as case-insensitive for consistency.

Regular notifications for non-critical issues (minor errors, warnings) can be logged in a dashboard where admins can review and decide if further action is needed. Critical issues such as security breaches, system resource issues, and payment failures, should generate instant alerts to allow for immediate resolution.

#### 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, changes to users (adding, removing, or modifying user details) should be possible without changing the code. This can be achieved through:

* **Admin Dashboard/Control Panel:** Administrators should have access to a web-based dashboard where they can:
  + Add new users (students, instructors, or other admins) by entering details and assigning roles.
  + Modify user profiles, such as updating personal details or changing roles (e.g., promoting a student to instructor).
  + Remove or deactivate users if they no longer need access to the system.

#### To ensure the system remains functional and compatible with platform updates (e.g., operating system, database, or application platform updates), the following strategies can be used:

* **Regular Monitoring and Testing**:
  + IT admins should regularly monitor platform updates and test the system in a staging environment before applying updates to the live system. This ensures that any potential compatibility issues can be identified and resolved without affecting users.
* **Modular Architecture:**
  + Designing the system with a modular architecture allows individual components (e.g., user interface, database, payment system) to be updated independently. This reduces the risk of a single update breaking the entire system.
* **Automatic Backups and Rollbacks**:
  + Before applying platform updates, the system should perform automatic backups. If an update causes issues, the system can roll back to the previous stable state.
* **Cloud-Based Solutions:**
  + If hosted on a cloud platform, the system can take advantage of managed services that handle platform updates while ensuring minimal disruption to the system.
* **Automated CI/CD Pipeline:**
  + Implementing a continuous integration/continuous deployment (CI/CD) pipeline ensures that platform updates and code changes are automatically tested, deployed, and rolled back in case of failure, keeping the system compatible and functional.

The IT Admin needs comprehensive access to manage and maintain the system. Here’s a breakdown of the required access levels:

* **User Management:**
  + IT admins should be able to create, delete, or modify user accounts.
  + They should also have access to modify roles and permissions for different types of users (students, instructors, other admins).
* **System Monitoring and Logs:**
  + IT admins need access to system monitoring tools to track system performance (e.g., CPU, memory usage, disk space) and ensure the system is running smoothly.
  + Access to system logs for error tracking, login attempts, and usage patterns is essential for troubleshooting and identifying security risks.
* **Security and Permissions:**
  + IT admins should have control over the security settings, including setting password policies, limiting login attempts, and managing two-factor authentication (2FA) settings.
  + They should be able to manage access control lists (ACLs) to restrict or grant access to different parts of the system.
* **Database Access:**
  + IT admins need read/write access to the database to perform maintenance tasks such as backing up data, restoring backups, and optimizing database performance.
  + This access allows them to resolve issues like corrupted data or failed transactions quickly.
* **System Updates and Maintenance:**
  + IT admins should have the ability to apply software patches, update system components, and deploy new versions of the system when necessary.
  + Access to manage scheduled maintenance windows for tasks such as server updates, data migrations, or system optimizations is also essential.
* **Backup and Recovery:**
  + IT admins must have access to automated backup tools and the ability to perform manual backups or initiate a recovery in case of system failure.
* **Security Alerts and Incident Response:**
  + IT admins should receive notifications of security issues (e.g., failed login attempts, suspicious activity) and have access to tools that allow them to quickly lock user accounts or block access in response to an incident.

#### 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?*

For users to log in, the system requires the following:

* **Username or Email Address:** Users will enter a unique identifier such as a username or registered email address.
* **Password:** Users must enter a password that meets security criteria (e.g., minimum length, special characters, etc.).
* **Two-Factor Authentication (Optional):** For enhanced security, users may be required to complete a two-factor authentication (2FA) step by entering a code sent to their mobile device or email.

To secure the connection and protect data exchange between the client (user) and server, the following measures should be implemented:

* **HTTPS (SSL/TLS Encryption):**
  + Ensure all communication between the client and server is conducted over HTTPS. This encrypts the data during transmission using SSL/TLS, preventing attackers from intercepting sensitive information such as login credentials.
* **Data Encryption:**
  + Sensitive user data (passwords, personal information) should be encrypted before being stored in the database using hashing algorithms such as bcrypt or SHA-256. Passwords should never be stored in plaintext.
* **Secure Session Management:**
  + Use secure cookies to manage user sessions, and ensure that session tokens are random, unique, and HTTP-only to prevent cross-site scripting (XSS) attacks.
* **Cross-Site Request Forgery (CSRF) Protection:**
  + Implement CSRF tokens to ensure that requests made on behalf of the user are legitimate and not part of a malicious attack.
* **Firewall and Intrusion Detection:**
  + Employ firewalls and intrusion detection systems (IDS) to monitor incoming traffic and detect suspicious activity or unauthorized attempts to access the server.

In the event of a brute-force attack (where an attacker tries multiple username-password combinations repeatedly), the system should implement several defenses:

* **Account Lockout After Failed Attempts:**
  + After a certain number of consecutive failed login attempts (e.g., 5 attempts), the account should be temporarily locked (e.g., for 15 minutes) to prevent further attempts. The user will not be able to log in until the lockout period has passed.
  + Alternatively, the account could be locked indefinitely until the user manually unlocks it by following a password reset process or verifying their identity.
* **Captcha:**
  + After a few failed login attempts, prompt the user with a Captcha to ensure that login attempts are coming from a human and not an automated script.
* **Two-Factor Authentication (2FA):**
  + Implement 2FA for logins. Even if the attacker guesses the password, they would also need access to the user’s secondary authentication method (such as a mobile phone) to successfully log in.
* **Alerts and Notifications:**
  + Notify the user via email or SMS if there have been multiple failed login attempts, allowing them to take immediate action, such as resetting their password or contacting support.

If a user forgets their password, the system should implement a secure and user-friendly password recovery process:

* **Password Reset Request:**
  + The user can request a password reset by entering their registered email address or username on the login page.
* **Verification of Identity:**
  + To verify the user's identity, the system sends an email with a password reset link or a one-time password (OTP) to the registered email or phone number.
* **Secure Reset Link:**
  + The password reset link should be time-sensitive (e.g., valid for 15-30 minutes) and contain a unique token that allows the user to securely reset their password.
  + The link should expire after a single use, and if not used, it should be invalidated.
* **Password Criteria:**
  + The user is prompted to create a new password that meets the system’s security requirements (minimum length, special characters).
* **Notification:**
  + After successfully resetting the password, the system should notify the user via email, informing them that their password has been changed. This helps the user detect any unauthorized changes to their account.
* **Optional Security Questions:**
  + For added security, some systems ask users to answer predefined security questions before allowing password resets.

### 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 validate user credentials when logging in to ensure that only authorized users can access the platform.
* The system shall allow students to register for an account, providing necessary details such as name, email, and password.
* The system shall allow students to take online practice exams, presenting questions, tracking responses, and providing feedback.
* The system shall store and track student exam results to enable progress tracking and performance analysis.
* The system shall allow students to schedule on-the-road training sessions with available instructors.
* The system shall allow instructors to view and manage their training schedules, showing upcoming sessions and available slots.
* The system shall allow instructors to access student performance data from practice exams to tailor on-the-road training.
* The system shall notify students of upcoming training sessions via email or SMS.
* The system shall allow administrators to add, modify, or remove users (students, instructors, and other admins) through a control panel.
* The system shall allow administrators to update and manage practice exam content, including adding new questions or modifying existing ones.
* The system shall restrict access based on user roles, ensuring that students, instructors, and administrators only access features relevant to their roles.
* The system shall allow students to view their training and exam history, including completed practice exams and scheduled sessions.
* The system shall integrate with a third-party payment processor for any fees associated with on-the-road training.
* The system shall generate reports for administrators on overall system usage, student performance, and instructor activity.
* The system shall log all failed login attempts and notify the administrator after a predefined number of failed attempts.
* The system shall lock student accounts after multiple failed login attempts to prevent unauthorized access.
* The system shall allow users to reset their passwords by sending a password reset link to their registered email.
* The system shall encrypt all sensitive data, such as user credentials and personal information, during storage and transmission.

### 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 for the DriverPass system must be user-friendly, intuitive, and accessible across multiple devices. It should provide clear navigation for students, instructors, and administrators, ensuring that users can perform tasks efficiently without confusion.

### Students: Easy access to exams, scheduling, and feedback. Mobile-friendly design for on-the-go usage.

### Instructors: Schedule management and student performance tracking. Responsive on both mobile and desktop.

### Administrators: Full access to user management, content updates, and reporting tools. A comprehensive dashboard interface is essential for administrative tasks.

### 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?*

* **Users:**
  + Will have access to the internet to utilize the online practice exams and scheduling features.
  + Will have valid credentials to access their accounts.
* **Instructors:** 
  + will have the necessary qualifications to provide on-the-road training.
* **Error Handling and User Support:**
  + The design does not explicitly cover how the system will handle errors such as incorrect inputs, failed transactions, or system downtimes.
  + It also does not describe whether there will be a support/help section for users to resolve common issues or contact support.
* **Localization and Multi-Language Support:**
  + The design does not address whether the system will support multiple languages for users who may not speak the default language of the system. This could be relevant, especially if DriverPass expands to non-English speaking markets.
* **Instructor Availability Management:**
  + The design does not provide specific information about how instructors will manage their availability. Will they have predefined time slots or dynamic scheduling capabilities?

### 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 design has limitations in scalability, offline functionality, instructor customization, security, and mobile optimization. Accessibility features and support for multiple languages or regions are also limited.

### Resource, time, and budget constraints may impact the system's ability to implement advanced security features, extensive testing, and third-party integrations. There could be challenges with cloud infrastructure costs, cross-platform compatibility, and optimization for both mobile and desktop devices.

### Overall, the system will function well but may require incremental updates to address these limitations and enhance performance, scalability, and security over time.

### 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.*

