# 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 section of a Business Requirements Document should include the client's name or organization, project purpose, client's goals, and key problems to solve. The project aims to automate inventory management, reduce manual errors, and improve efficiency. The client wants the system to track inventory levels in real-time, generate automatic restock alerts, and provide detailed reports. The key problems to address include issues with manual inventory tracking, leading to frequent stockouts and overstock situations. This template helps in defining the system components and design.]

### 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 aims to streamline and enhance the process of scheduling and managing driving lessons by automating the process, providing an easy-to-use interface for students and instructors, tracking student progress and performance, and generating reports and analytics on lesson attendance and performance. The system aims to address current issues such as time-consuming manual scheduling, difficulty in tracking student progress, lack of centralized data management, and inefficient communication between students and instructors.

The system consists of a user interface, scheduling module, progress tracking tools, communication tools, reporting and analytics, and data visualization tools. The system aims to address the current issues and improve the overall experience for students and instructors.]

### 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 aims to automate scheduling, track progress, generate reports, facilitate communication, and centralize data management. It should integrate calendars for real-time scheduling and develop algorithms to match student availability with instructor schedules. Progress tracking tools should be created for instructors, and a student portal should be developed for viewing and receiving feedback. Report generation modules should include data visualization tools for easy interpretation.

Communication tools should include a messaging system for students and instructors, and notifications and reminders for upcoming lessons. A centralized database for storing lesson, student, and instructor information should be developed, ensuring data security and privacy measures.]

## 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?*

* [ This section outlines the nonfunctional requirements for the DriverPass system, including its performance requirements, operating environments, response time, load time, and scalability. The system should be accessible via web browsers on both desktop and mobile devices, with a dedicated mobile app for iOS and Android platforms. It should have a response time of less than 2 seconds for most user interactions, load time of 3 seconds, and be able to handle many concurrent users without significant performance degradation. Regular updates should be rolled out monthly, with emergency patches and user feedback being driven to continuously improve the user experience.]

#### 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 should run on Windows, Unix/Linux, macOS, and iOS and Android for desktop users, Unix/Linux for server-side operations, macOS for Apple devices, and both iOS and Android for mobile users. The back end requires a robust Relational Database Management System (RDBMS) or a NoSQL database for flexibility and scalability. The server should be a reliable web server like Apache or Nginx, and an application server like Node.js, Tomcat, or Django for back-end logic. Development tools include version control tools like Git, Continuous Integration and Deployment pipelines like Jenkins or GitHub Actions, and security tools like encryption and authentication mechanisms like OAuth or JWT for secure user access.]

#### 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?*

* [ System identification is crucial for tracking user-specific data and preferences. Unique identifiers like usernames, email addresses, or user IDs are stored in a database. Input case sensitivity depends on the system's design, with usernames and passwords often case-sensitive for security. System administrators should be informed of problems like repeated failed login attempts, system performance metrics exceeding predefined thresholds, critical errors or exceptions impacting system functionality, or unusual user activity detected.]

#### 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?*

* [ An admin interface or dashboard allows administrators to make changes to users without changing code, allowing them to add, remove, or modify user information without requiring code changes. Systems can adapt to platform updates by implementing modular architecture and continuous integration/continuous deployment pipelines, ensuring seamless updates and integration without disrupting existing functionality. IT admins need elevated access to manage user data, monitor system performance, troubleshoot issues, and manage user permissions, roles, and security settings.]

#### 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?*

* [ To log in to a system, users need a username or email address and password, with some systems requiring two-factor authentication. To secure data exchange, use HTTPS and end-to-end encryption. Regular system updates and patching are crucial. To prevent brute force hacking, implement account lockout mechanisms after failed login attempts, use CAPTCHAs to distinguish between human users and automated scripts, and monitor suspicious activities. Provide a password reset option for users, with security questions or verification steps to ensure identity. Encourage strong, unique passwords and consider password managers.]

### 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 aims to provide user management, security, and platform adaptability. It allows administrators to add, remove, and modify user accounts, validate user credentials, and provide password reset options. It also locks accounts after a predefined number of failed login attempts to prevent brute force attacks. The system uses HTTPS, two-factor authentication, and strong encryption algorithms to encrypt sensitive user data. It logs and monitors user activities for security purposes. The system supports modular architecture, allowing seamless updates and integration of new features. IT admins have elevated access to manage user data, backups, and system performance. The system offers a user-friendly interface and clear error messages for troubleshooting.]

### 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 should be user-friendly, responsive, accessible, and secure. It should allow users to log in and out securely, manage profiles, and manage two-factor authentication. Administrators should have access to database, server, and application management tools. The interface should be optimized for mobile devices, compatible with major web browsers, and available as a desktop application. ]

### 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's scalability, data backup, user training, compliance with data protection laws, and integration with other systems are not specifically addressed. It will be hosted on a reliable, scalable cloud infrastructure, using modern web technologies and robust frameworks.]

### 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?*

* [ System design limitations include scalability, security, integration, user experience, and data management. Limited resources, time, budget, and technology can impact project progress. Mitigation strategies include prioritization, scalability planning, security best practices, user feedback, and resource allocation. To address these challenges, prioritize critical features, plan for iterative improvements, and stay updated with security trends and patches. ]

### 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 graph with green rectangles

Description automatically generated]