# 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?*

* Client: Liam, the current owner of DriverPass corporation.
* Vision:
  + Offer training to improve a prospective driver’s chance of passing their driver’s test.
  + Fill a market void caused by poor the current poor driver training offerings.
  + Offer practice tests online.
  + Offer in-vehicle training services.

### 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?*

* Problem:
  + High fail rate at the DMV.
  + Limited preparation materials currently available.
  + No current systems in place to facilitate student success.
* Components:
  + Offer online class material with tests for competency assessment.
  + Driver-student in-vehicle on-on-one training.
  + Data needs to be accessible anywhere – even offline – and synced once internet connectivity restored.
  + Implementation of security; management of user roles and permissions.; log data collection.
  + Manage system reservations. Customer-capable scheduling/updating/canceling reservations.
  + Initial Package offerings with option to scale offerings for future business needs.
  + Database setup to store customer profile, registrations, payment transactions and history.
  + Pull DMV updates and information to keep training materials current.
  + Implement friendly user interface that allows customers to track their progress and bolster their confidence for the test.
  + Build Cloud-based infrastructure to facilitate dynamic growth with minimal disruptions.

### 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?*

Completed System Capabilities

* Provide/facilitate online classes and practice tests aligned with DMV test material.
* Facilitate driver registration/lesson sessions to include
  + Time slots
  + Instructor/student pairings
  + Cars used in instruction
* Track lesson completion and overall progress to include:
  + Tests taken
  + Time of tests
  + Scores from tests
  + Boolean completion status [True/False]
* Communicate with customers:
  + Helpdesk for struggling customers
  + Contact form for Business-to-Customer [B2C] interaction.
* Access Control:
  + Implement least privileges based on role of system user or administrator.
  + Give Liam full system access and privileges for accessing/offloading customer data.
* Notify Staff of New DMV Material
  + Implement new-DMV-material notification system.
  + Allow for test materials to be modified to input relevant course material updates.

Measurable Tasks Included in System Design

* Develop user roles, authentication protocols, authorization protocols.
* Design online course material, modules, and tests
* Create Reservation Management System with availability notifications and tracking protocols.
* Create functionality to track student progress and performance.
* Develop Customer Profile System.
* Develop Database management system – preferably Cloud-based.
* Create DMV update management system for use in new course/test material.
* Ensure data compliance and privacy for student and faculty records.
* Establish secure connection to/from [likely] Cloud-based Server.
* Create friendly User Interface for desktop, tablet, and phone devices.

## 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?*

* Environment:
  + Recommend running in cloud environment.
  + Cloud hosting offers rapid scalability, outsourced reliability, and easy access from anywhere.
  + Cloud hosting negates up-front expenses of vertically integrated self-funded server hardware.
* System Speed
  + Webpages should load within 1 – 5 seconds (Camarena, 2023) to avoid de-ranking risk by search engines,
  + Users expect rapid page loads and may canceling their transactions due to impatience with excessive system latency.
* System Update Cadence
  + The system update schedule should sync with the DMV cadence pace.
  + If DMV has no release schedule, system should check for updates no later than biweekly.

#### 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?*

* System platform
  + Linux: Ubuntu Server
  + Platform independence in terms of accessibility to allow iPhone, Android, Windows, MacOS, and other system to connect.
* Backend tools
  + SQL database such as MySQL RDBMS
  + RESTful API to connect to database and GET/POST/PUT/DELETE traffic from the client [front end] to the server [back end] (Sisheshy2ey, n.d.).
  + Would recommend Java CXF as build choice for API spec due to cost savings, flexibility of implementation, and security.
  + Maven package manager for backend build.
  + Apache Tomcat for deploying the app.

#### 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?*

* Distinguishing Between Users
  + Establish role based access controls:
    - Administrator [admin privileges] Liam
    - Super Users [elevated privileges] Ian
    - Developers [developer privileges]
    - Regular Users [general system use]
    - Guests [preview privilieges]
* Input Case Sensitivity
  + Passwords must be case sensitive for login
  + Usernames must be case sensitive for login
* When Should System Notify Admin of Problem
  + System should log problems to console for admin review.
  + System should immediately notify Liam if client/server connection is broken.
  + System should notify Liam of reservations, cancellations, and modifications.

#### 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?*

* User Management without Changing Code
  + Short answer: Yes
    - System should allow Role Based Access Control [RBAC]
    - Admins [Liam] assign roles and permissions that are associated with those roles.
    - Liam can assign equal role to himself if he deems necessary to distribute administrative workload.
    - User Interface handles add/remove/modify users without affecting entire code base.
* System Adaptation to Updates
  + Cloud offers standardized updates
    - Updates may be scheduled
    - Server will be split into separate instances to ensure constant uptime, regardless of update size. Only one active instance will host the production system.
* Admin IT Access
  + As mentioned above, admins require full system access with all permissions enabled.
  + Admins may create, read, update, and delete system users.
  + Admins may assign role-based access permissions.
  + Admins receive all system logs.

#### 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?*

* User login requirements
  + All users required to login for system access
  + All users required to submit unique username/password combo
  + Multifactor authentication [MFA] setup required upon account creation. MFA links cellphone number [something you have] with username/password [something you know].
  + User shall provide email address for account recovery during account creation.
* Securing Connection and Data Transfer
  + Encryption via HTTPS protocol shall be used.
* Handling Brute Force Attacks
  + Account lockout after 3 consecutive incorrect username/password attempts
  + User must create new username/password via email.
  + CAPTCHA challenges shall be implemented to deter brute force attacks.
  + If all the above measures fail, IP blocking shall be enabled to prevent specific addresses from attacking the system.
* User Password Recovery
  + Password recovery for forgotten or compromised password handled with user-provided email address given during account creation.
  + Email shall be sent to user with a link to reset their password.
  + Password reset link that is sent to user shall expire after 24 hours.

### 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:
  + Require account creation to log in
  + Require all users to submit unique username/password combo for account creation.
  + Require all users to submit their account recovery email address during account creation.
  + Send email notification to users to activate a newly created account.
  + Require users to click the link in the account activation email to begin using their accounts.
  + Allow users to access data and perform actions based upon their assigned role.
  + Provide online services to facilitate driving classes and practice tests.
  + Give users the ability to schedule tests and appointments with the secretary.
  + Track and log all user activities including but not limited to:
    - Making:
      * Cancellations
      * Reservations
      * Appointment modifications
  + Support multiple packages for driving lessons with varying time durations for in-person lessons.
  + Enable customization of lesson plans by admins without requiring extensive codebase updates.
  + Record and display all driver notes and lesson times for each lesson.
  + Store all user information for user profile to include:
    - First name
    - Last name
    - Address
    - Phone number
    - State
    - Payment [Credit card] details
  + Exchange data over secure HTTPS connection.
  + Use CAPTCHA for account login.
  + Institute account lockout policy after 3 consecutive failed login attempts.
  + Allow users to reset password via recovery email.
  + Keep passwords securely encrypted to prevent unauthorized access.
  + Establish and maintain connection to DMV to receive updates regarding policies, rules, and sample questions for tests.
  + Auto-update system administrators when new DMV driver data is available.
  + Provide friendly and easy-to-use user interface to system users.
  + Be platform independent, so users of all types of devices have the chance to interact with the system.
  + Be consistent with page layout [same color theme, font, etc.] to ensure consistent experience regardless of device screen size.
  + Have high uptime >= 98%.
  + Run off the web, preferably via cloud topology.
  + Have automated security and update schedule.

### 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.)?*

* Needs of the Interface
  + Shall be user friendly and uncomplicated.
  + Shall support different roles with various access and permissions.
  + Shall present information to users in a clear and organized way to facilitate driving lesson packages, scheduling actions, and appointment creation.
  + Shall require and facilitate secure login and account recovery protocols.
  + Shall allow users to track their progress.
  + Shall allow users to check and update their profile.
  + Shall allow users to schedule and manage their lessons.
* Different Users for Interface
  + Shall present different views for the following roles:
    - Administrators
    - Customers/Users
    - Secretary
* User Capabilities
  + Admins:
    - Log in to their account.
    - Manage driver lessons by adding, modifying, or disabling them.
    - Track user activities such as cancellations or reservations.
    - Handle account-related issues.
    - Receive notifications from DMV and manage system configurations.
  + Secretary:
    - Log in to their account.
    - Access appointments on behalf of customers
    - Book appointments on behalf of customers
    - View and manage customer rescheduling or cancellation needs
    - Provide assistance to customers with their inquiries
  + Customer/Users
    - Log in to their account.
    - View available lessons, lesson details, and prices.
    - Schedule appointments for driving lessons with dates and times available for view.
    - Access their current progress with lessons.
    - Access their test scores.
    - Access study materials, practice tests, and driving lesson resources.
* How Users Interact with Interface:
  + Via Web Browser:
    - Admins, secretaries, and customer/users can access the user interface via any device capable of connecting to the internet via a web browser.
  + Mobile Devices:
    - User interface should be responsive to handle tablet and cellphone screen sizes.

### 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?*

* Things not Explicitly Addressed Above:
  + User Interface and Design
    - Specific visual layout [Wireframe] was not provided.
    - User interface was not described in detail above.
    - Color scheme was not decided.
    - Typography, icons, and logo were not provided.
  + Error Handling and Message Logging
    - Specifics were not provided for:
      * Error handling
      * Input validation errors
      * User feedback for runtime bugs
  + Testing and Quality Assurance
    - Testing methodologies not established. Methodologies might include:
      * Unit Testing
      * User Acceptance Testing
      * Integrated Testing
    - Quality Assurance measures were not discussed.
  + Data Storage and Backup
    - Data storage was not addressed in detail.
    - Backup protocols were not addressed in detail.
  + Language and Localization:
    - The design did not address multiple language needs for users.
* Assumptions Made:
  + User Technical Proficiency:
    - Design assumes basic user technical proficiency with online learning platforms.
  + Availability of DMV Updates:
    - System design assumes DMV will make updates, rules, and other materials easily available.
  + Internet Connectivity:
    - The designs assume that users will have access to the internet.
  + Standard Security Measures
    - The system assumes that encryption measures like CAPTCHA and other security practices will be incorporated and followed during system design and use.

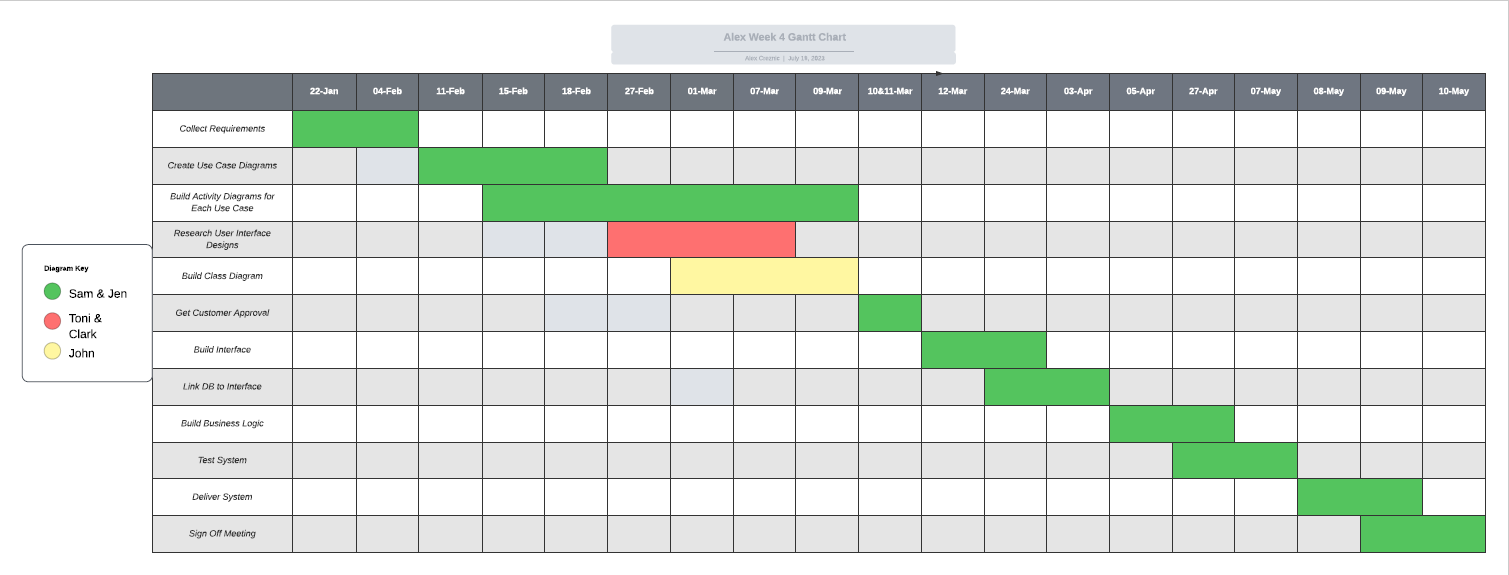
### Limitations

*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
  + System lacks detailed user interface design.
  + System lacks third-party integration for hook-up with DMV.
  + System did not specify scalability plans or detail how it will handle rapid and dynamic user increases/decreases over time.
  + System did not provide specifics for testing approach for online test takers.
  + System did not give security audit details, such as how it will be implemented or who shall implement it.
* Budget, Time, Technology Limitations:
  + Cloud topologies may be too expensive over time.
  + Vertically integrated topologies may be too expensive up front.
  + Time requirements are greater with vertically integrated system. Developer requirements may necessitate hiring professionals, which increases the cost of development further.
  + Vendor-run cloud systems offer less granular control over the system, possibly negating certain features that Liam wants included in the application.
  + Adding too many features to the launch version of the application will bump out the time estimate, delaying earnings for the system owner.

### 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.*

**

**References**

Camarena, A. (March 07, 2023). *What is page speed and how to improve it.* Semrush SEO. https://www.semrush.com/blog/page-speed/.

Visheshy2ey. (n.d.). *RESTful web services.* Geeks for Geeks. https://www.geeksforgeeks.org/restful-web-services/.