**CS-255 Project One**

**Omar Raymond**

**November 30, 2024**

**David Randolph**

# CS 255 Business Requirements Document

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, DriverPass, wants the system to provide comprehensive and effective driver training to help customers better prepare for their DMV driving tests. This includes offering online courses and live instruction as optional resources to improve training outcomes.

### 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 address the high failure rate at the DMV, where more than 65% of people fail their driving tests. The current training methods are insufficient in preparing individuals for success.
* **Different Components Required include:**
  + **Online Training Module:**
* Interactive video lessons covering driving theory and practical skills.
* Quizzes and assessments to reinforce learning.
* **Live Instruction Module:**
  + Scheduling system for one on one or group live instruction sessions.
  + Video conferencing tools for remote live training.
* **Performance Tracking:**
  + A dashboard for users to monitor their progress and test readiness.
  + Analytics for identifying areas where additional training is needed.
* **DMV Test Simulation:**
  + Practice tests that mimic DMV formats to familiarize users with the testing environment.
  + Feedback and tips based on simulation results.
* **Customer Management System:**
  + User registration and profile management.
  + Payment processing for training courses and services.
* **Instructor Management Tools:**
  + Scheduling and communication tools for instructors.
  + A resource library for instructors to use during training sessions.
* **Support System:**
  + FAQs and self help resources for users.
  + Chat or email support for addressing user concerns or technical issues.

### 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 completed, the system should be able to:**
  + Provide comprehensive driver training through online courses and live instruction.
  + Offer DMV test simulations to familiarize users with test formats.
  + Track user progress and provide tailored feedback for improvement.
  + Enable scheduling and management of live training sessions.
  + Handle user registration, profile management, and payment processing efficiently.
  + Support instructors with tools for scheduling and training resources.
  + Deliver responsive customer support via chat or email.
* **Measurable Tasks needed to be completed:**
  + **Online Course Delivery:**
  + Create and integrate interactive lessons and quizzes.
  + Ensure smooth video streaming and content navigation.
* **Live Instruction Features:**
  + Implement a scheduling system for live sessions.
  + Integrate a reliable video conferencing tool.
* **Performance Analytics:**
  + Develop a user dashboard displaying progress and readiness scores.
  + Include analytics to highlight weak areas needing improvement.
* **DMV Test Simulation:**
  + Design realistic DMV style practice tests.
  + Include automated feedback on results.
  + Test usability on both iOS and Android systems.
* **Customer Management:**
  + Build user friendly registration and login features.
  + Integrate secure payment gateways.
* Instructor Tools:
  + Add scheduling, communication, and resource sharing capabilities.
* **Customer Support:**
  + Include a searchable FAQ section.
  + Integrate a ticketing or live chat system for support.

## 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.*

* **Nonfunctional Requirements include:**
  + **Security:**
  + Robust security protocols are required to protect user data, including secure authentication, authorization, encryption, and compliance with privacy regulations.
  + Security is essential to safeguard sensitive information and maintain user trust.
* **Usability:**
  + The system must feature an intuitive interface, enabling users to navigate easily.
  + This reduces onboarding time and minimizes frustration, even for those with limited technical skills.
* **Scalability:**
  + The platform should handle a growing number of users without performance degradation.
  + Scalability ensures consistent operation as user demand increases.
* **Reliability and Uptime:**
  + The system should achieve a high uptime rate, minimizing disruptions to the user experience.
  + A stable platform is critical to maintain user confidence and satisfaction.
* **Data Backup and Recovery:**
  + Regular backups should be scheduled to protect against data loss.
  + Efficient recovery processes must ensure quick restoration of course and user data in the event of issues.

#### 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?*

* **Environments:**
* Web Based: Accessible via major browsers (Chrome, Firefox, Safari, Edge).
* **Performance:**
* Pages load within 2 seconds.
  + Interactive features respond within 1 second.
  + Smooth video streaming with minimal buffering.
* **Update Frequency:**
  + Major Updates: Every 3-6 months for new features.
  + Minor Updates: As needed, critical fixes within 24-48 hours.
  + Security Patches: Deployed immediately when vulnerabilities are found.

#### 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?*

* **Platforms:**
  + Windows, macOS, and Linux/Unix for desktop users.
  + Accessible via all major web browsers (Firefox, Chrome, Safari, Edge)
* **Backend Requirements:**
  + Database: MySQL, PostgreSQL, or MongoDB for storing user data, course content, and test results.
  + Server Framework: Node.js, Django, or Spring Boot for backend operations.
  + Cloud Hosting: AWS, Azure, or Google Cloud for scalability and storage.
  + APIs: Payment (e.g., Stripe, PayPal) and video conferencing (e.g., Zoom SDK, Twilio).
  + Authentication: OAuth or JWT for secure access.
  + Analytics: Tools like Google Analytics can be used to monitor user performance.

#### 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?*

* **User Differentiation:**
  + Unique Identifiers: Each user will be distinguished by a unique identifier such as an email address or username.
  + Roles: Users will be categorized into roles (e.g., admin, instructor, student) to manage access levels and permissions.
* **Case Sensitivity:**
  + Input Handling: Usernames and passwords are case-sensitive to ensure security, while other inputs (e.g., names, email addresses) are case-insensitive for user convenience.
* **Admin Notifications:**
  + **The system should notify the admin in the following scenarios:**
    - Security Issues: Multiple failed login attempts or suspicious activity.
    - System Failures: Downtime, failed backups, or performance issues.
    - User Issues: Payment failures or unresolved support tickets.
    - Capacity Alerts: When system usage approaches resource limits.

#### 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 Code Changes**:
  + Admin Dashboard: Provide a user friendly interface for adding, removing, or modifying users without altering code.
  + Role Based Controls: Allow admins to assign or adjust user roles and permissions through the dashboard.
* **Adapting to Platform Updates:**
  + Modular Design: Use a modular system architecture to allow seamless integration of updates.
  + Backward Compatibility: Ensure updates do not disrupt existing features or data.
  + Automated Testing: Regularly test updates in staging environments to minimize risks.
* **IT Admin Access:**
  + Full Administrative Rights: Access to manage users, roles, and permissions.
  + System Monitoring Tools: View performance metrics, error logs, and system health.
  + Backup and Recovery: Ability to initiate data backups and restore processes.
  + Configuration Management: Modify system settings, integrate APIs, and oversee platform updates.

#### 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?*

* **Login Requirements:**
  + Users must provide a username and password to access the system.
* **Securing Connections and Data Exchange:**
  + All network requests will use HTTPS to ensure secure communication between client devices and the server.
  + Sign in forms will send data via HTTPS POST requests, keeping sensitive information out of URLs.
  + Encryption will protect sensitive data during transmission across networks.
* **Handling Brute Force Hacking Attempts:**
  + Accounts will be locked after 5 failed login attempts to prevent brute force attacks.
  + The system will notify the IT admin of the lockout, and the admin can guide the user on unlocking the account and updating their password.
* **Password Recovery:**
  + Users can initiate a password reset by verifying their account via an identifier, such as their email address.
  + A reset link will be sent to the verified email for users to update their password.

### 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.”*

* **Functional Requirements include:**
  + **User Account Management:**
    - The system shall validate user credentials when logging in.
    - The system shall provide customized dashboards for students, instructors, and administrators based on their roles.
    - The system shall enable administrators to create, update, and delete user accounts without changing code.
  + **Course Content Delivery:**
    - The system shall allow instructors to upload, organize, and present course materials, including lectures, readings, and multimedia content.
    - The system shall support various file formats for course materials (e.g., PDFs, videos, and slide presentations).
  + **Grade and Performance Tracking:**
    - The system shall allow instructors to input and update student grades securely.
    - The system shall display performance metrics to students via their personalized dashboards.
  + **Assignment Submission and Feedback:**
    - The system shall enable students to submit assignments securely.
    - The system shall allow instructors to provide feedback on submitted assignments.
    - The system shall integrate with plagiarism detection tools to ensure academic integrity.
  + **Discussion Forums and Messaging**
    - The system shall provide a discussion forum for students and instructors to interact.
    - The system shall support private messaging between users to facilitate direct communication.
  + **Calendar and Notification System**
    - The system shall include a calendar displaying due dates, events, and deadlines.
    - The system shall send notifications to remind users of upcoming tasks and events.

### 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 Roles and Needs**
  + **Students:** 
    - View course materials, grades, and performance metrics.
    - Submit assignments and receive feedback.
    - Participate in discussions and send/receive messages.
    - Access calendars and notifications for deadlines and events.
  + **Instructors**
    - Upload, organize, and update course content.
    - Grade assignments and provide feedback.
    - Monitor student performance and identify areas of improvement.
    - Manage discussions and send messages to students.
  + **Administrators:**
    - Create, update, and manage user accounts.
    - Monitor system usage and handle technical issues.
    - Oversee course setup and instructor assignments.
    - Review and resolve flagged issues (e.g., locked accounts).
* **Interaction Methods:**
  + Web Browsers: Full functionality on major browsers like Chrome, Firefox, Safari, and Edge for desktop and laptop users.

### 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?*

* **User Assumptions:**
  + Users have access to reliable internet connections for smooth platform operation.
  + Users have basic technical skills to navigate the system.
  + Users will use commonly available devices (smartphones, tablets, laptops, or desktops) with up to date operating systems and browsers.
* **Technology Assumptions:**
  + The system will primarily rely on cloud infrastructure for scalability and reliability.
  + Third party tools like video conferencing APIs and plagiarism checkers will be integrated without significant compatibility issues.
  + User data storage complies with security standards like GDPR or FERPA where applicable.
* **Functionality Assumptions:**
  + Notification settings will be adjustable, allowing users to manage frequency and type.
  + Calendar events and deadlines are managed uniformly across user roles.
  + Any additional custom features or integrations requested by stakeholders will be addressed in future updates.
* **Unaddressed Aspects:**
  + Detailed requirements for offline access are not explicitly covered.
  + Specific hardware requirements (e.g., device specifications) are assumed to be standard and are not explicitly detailed.
  + Support for languages other than English is assumed to be unnecessary unless explicitly requested later.

### 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?*

* **Budget Constraints:**
  + Limited funding may restrict the use of advanced features, such as AI-driven analytics or highly customizable dashboards.
  + Scalability options may be constrained, potentially impacting performance under heavy user loads.
* **Limited IT Support Resources:**
  + A small IT team may limit the ability to provide extensive real-time support.
  + This necessitates a focus on creating self-help resources like FAQs and an intuitive interface to minimize reliance on IT support.
* **Time Constraints:**
  + Tight deadlines may reduce the opportunity for extensive testing, potentially affecting initial system stability or performance.
  + Feature prioritization may delay the implementation of lower-priority functions until future updates.
* **Technology Limitations:**
  + Dependence on third-party tools (e.g., APIs for video conferencing or plagiarism detection) may introduce compatibility or reliability risks.
  + Offline access is not currently prioritized, limiting functionality for users without consistent internet connectivity.
* **User Accessibility:**
  + Assumes users have basic technical skills and modern devices; users with outdated hardware or limited digital literacy may face challenges.
* **Localization and Accessibility**
  + Initial design may lack support for multiple languages or advanced accessibility features, which might limit inclusivity.

### 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 project

Description automatically generated