CS 255 Business Requirements Document
Derricko Swink

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 to take advantage of a void driver test training market and provide a better-quality service to their students.
- DP ("DriverPass") wants to develop a comprehensive system that meets the company's specific business needs.
- DP is primarily focused on improving driver training while also helping students improve on high failure rates due to inadequate training.
- DP wants to address this problem by providing a robust platform for online classes, practice tests, and on-the-road training.
- DP wants to be able to download reports and other information that can be retrieved and worked on anywhere.

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?

- DP wants a user-friendly system accessible from any device to manage appointments, track training progress, and ensure compliance with DMV regulations.
- DP wants to have full access over all accounts to be able to provide password resets and block access from terminated employees.

- DP wants to offer on-the-road training sessions with certified instructors.
- DP wants to solve the problem of the lack of resources for driver training.
- DP wants to solve the problem of people failing their driving tests at the DMV.

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?

- Develop a platform for comprehensive online classes and practice tests that allow users the ability to access, start, complete, and track progress in their online courses and practice tests.
- The system should also record scores and provide valuable feedback.
- System should enable users to schedule, modify, and cancel driving lessons online or via phone.
- Users should be able to book, change, or cancel appointments while the system updates in real-time to reflect these changes.
- System should match customers with available drivers and cars while maintaining a schedule that prevents conflicts.
- System should ensure that each user type has appropriate access permissions, and sensitive actions (such as resetting passwords and accessing customer data) are securely controlled and logged.
- system should automatically update training materials and tests to reflect the latest DMV standards. Users receive notifications of changes.
- System will determine if users find the interface easy to navigate, and usability testing will show high satisfaction rates. Key features are accessible within a few clicks.
- System should be able to add or disable packages without significant redevelopment. Future features can be integrated with minimal disruption.
- System should log reservations, cancellations, or modifications, and track test progress with clear status indicators like “not taken”, “in progress”, “passed”, and “failed”.
- System should securely store customer data, comply with privacy regulations, and allow customers to update their information as needed.
- System should generate activity reports, allowing admins to review who made changes and what actions were taken. Feedback forms allow users to rate their experience.

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?

- System needs to run off the web, preferably over the cloud.
- System should help DP focus on running the business with minimal technical problems.
- System should provide tests and ensure that the practices are current with what the DMV requires.
- System's backup and security should be handled by a third-party and not DP.
- The system should be updated and send out notifications every time the DMV has a new update.
- System should run fast, efficiently, and seamlessly.

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 should be accessible via modern web browsers on both Windows and macOS operating systems.
- The mobile app should be compatible with both iOS and Android operating systems.
- The back-end servers should run on Unix-based systems, such as Linux, due to their robustness, scalability, and security features.
- The system should be hosted on a cloud platform (AWS, Azure, Google Cloud) to ensure scalability, reliability, and ease of maintenance.
- A relational database management system (RDBMS) such as MySQL.
- Server should be used to store user data, appointment schedules, training materials, and other critical information.

- Server should be designed for scalability and high availability, with support for regular backups and data recovery.

- Back-end should be developed using a robust and scalable programming language such as Java, Python, or Node js.
- Frameworks, such as Spring Boot (for Java), Django (for Python), or Express (for Node.js) can be used to streamline development and ensure maintainability.

- System should integrate with external APIs, such as the DMV's systems, for updated on rules and policies.

- RESTful APIs should be developed to allow for seamless communication between the front-end and back-end components.

- Tools and libraries for implementing encryption, authentication, and authorization should be used to secure user data and system access.

- Regular security audits and penetration testing should be conducted to identify and mitigate potential vulnerabilities.

- Tools such as Prometheus, Grafana, or ELK Stack should be used for monitoring system performance and logging errors or unusual activities.

- Automated alerts should be set to notify admins of potential issues or system outages.

- Continuous Integration and Continuous Deployment (CI/CD) tools such as Jenkins, Travis CI, or GitHub Actions should be used to automate the testing, deployment, and release process.

- Version control systems like Git should be used for source code management and collaboration.

- Systems needs to be able to identify the driver the customer is scheduled to go out with.

- System needs to be able to track which user is matched up with a certain driver, time, and car.

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?

- Different users will be broken down into access levels.

  (1) Admin will have full access to all system functionalities, including user management, system configuration, and report generation.

  (2) IT Officer will have access to system maintenance features, user account management, and troubleshooting.

(3) Secretary will have access to scheduling, user information management, and appointment modifications.

(4) Customer will have access to their own account for booking, modifying, and canceling appointments, as well as accessing training materials.

- Unique user IDS and passwords should be used to authenticate users.
- Role-based access control (RBAC) should be implemented to ensure users can only access functionalities and data pertinent to their role.
- Each user profile should store information, permissions, and personal details.
- System should log user activities, allowing tracking of actions performed by different users.
- Each user will have a unique username or email address which will serve as their primary identifier for logging into the system and access their account.
- Usernames and email addresses should generally be treated as case-insensitive during login and account identification to avoid confusion and user inconvenience.
- Passwords should be case-sensitive to enhance security, ensuring that users must enter their passwords with the correct case for authentication.
- Immediate notification should occur for critical errors that impact system availability or functionality, such as server failures, database connectivity issues, or security breaches.
- Suspicious activities or security incidents, such as unauthorized access attempts or anomalies in user behavior, should trigger immediate alerts to the admin.
- If the system's performance degrades beyond acceptable thresholds, such as prolonged response times or high server load, the admin should be notified to investigate and resolve the issue promptly.
- Any failures or errors during scheduled maintenance tasks or updates should trigger notifications to the admin, ensuring that they are aware of potential disruptions or issues.

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?

- Users and their roles should be managed through a database table rather than hardcoded into the application code. NOTE: This allows administrators to add, remove, or modify users without requiring changes to the underlying codebase.

- Access levels and permissions for different roles should be configurable through an admin interface or configuration files. This enables administrators to adjust access rights dynamically without programming changes.

- An administrative dashboard or interface should be provided where IT admins can manage users, roles, and permissions. This interface should allow for easy addition, removal, or modification of users and their attributes.

- There are multiple ways that the system can adapt to platform updates:

    (1) Regular Maintenance & Updates – plan regular maintenance windows to apply updates, patches, and fixes to the system. Also keep track of all software components' versions, including libraries, frameworks, and dependencies, to ensure compatibility with platform updates.

    (2) Modular Architecture - Design the system using a modular architecture where different components (e.g., frontend, backend, database) can be updated independently. Use microservices architecture to isolate functionalities, making it easier to update individual services without affecting the entire system.

    (3) Automated Testing - Implement CI/CD pipelines to automate testing and deployment processes. This ensures that any updates are tested thoroughly before being applied to the live system. Regularly perform regression testing to identify and address any issues that arise due to platform updates.

    (4) Backward Compatibility - Design the system to support multiple versions of platforms and dependencies, ensuring that updates do not break existing functionality. Use feature flags to gradually roll out new features and updates, allowing for controlled testing and rollback if necessary.

    (5) Monitoring & Alerts - Implement real-time monitoring tools to track system performance and detect issues arising from platform updates. Set up alerts and notifications for critical issues, enabling prompt response and resolution.

    (6) Documentation & Training - Maintain up-to-date documentation for the system architecture, components, and update procedures. Provide ongoing training for the IT team on handling platform updates and troubleshooting potential issues.

    (7) Vendor Support & Communication - Keep in regular contact with platform and tool vendors to stay informed about upcoming updates and changes. Leverage vendor support services to address any compatibility issues and receive guidance on best practices for updates.

- The IT admin needs to have full access over all accounts so that they can reset them if someone forgets their password, or if someone gets let go the IT admin needs to be able to block their access.

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?

- Users typically need a username/email and password combination to log in. Optionally, multi-factor authentication (MFA) can enhance security.

- Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocols should encrypt data transmitted between the client and server. HTTPS should be enforced to protect sensitive information.

- If a brute force hacking attempt is detected (multiple failed login attempts from the same IP or account), the account should be temporarily locked or require additional verification steps like CAPTCHA. Continuous attempts could lead to permanent lockout or notification to administrators.

- Users should have the option to reset their password via email verification or security questions. A temporary password or reset link should be sent to their registered email to regain access.

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.

- The system shall allow users to schedule driving lessons online.

- The system shall support three different packages for driving lessons: Package One (6 hours), Package Two (8 hours with in-person lesson), and Package Three (12 hours with in-person lesson and online class).

- The system shall track and display available times for driving lessons with different instructors.

- The system shall allow users to cancel or modify their scheduled driving appointments.

- The system shall provide role-based access control (RBAC) for Admin, IT Officer, Secretary, and Customer roles.

- The system shall encrypt sensitive data such as passwords and payment information during transmission and storage.

- The system shall send notifications to admins for critical errors or security breaches.
- The system shall integrate with DMV systems to update rules, policies, and sample questions.
- The system shall generate activity logs to track user actions such as scheduling, modifying appointments, and accessing training materials.

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.)?

- Interface needs to be user-friendly, accessible, and secure.

Admin:

• Needs: Manage system configurations, user roles, and permissions.
• Interface Actions: Access to system-wide settings, user management, and generating reports.

IT Officer:

• Needs: Maintain and troubleshoot system functionalities.
• Interface Actions: Monitor system health, manage backups, and address technical issues.

Secretary:

• Needs: Schedule driving lessons and manage customer appointments
• Interface Actions: View available lesson times, book appointments, and handle customer inquiries.

Customer:

• Needs: Schedule, modify, or cancel driving lessons, access training materials, and view payment history.
• Interface Actions: View available packages, select driving instructors, manage personal details, and track lesson progress.

• Users (Admins, IT Officers, Secretaries) interact with the system through a web-based interface accessible via standard web browsers such as Chrome, Safari, and Firefox.

• Customers interact with the system via a dedicated mobile application available on iOS and Android platforms. This allows them to conveniently schedule appointments, access training materials, and receive notifications on their mobile devices.

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?

- Specific measures for data privacy, compliance with regulations (such as GDPR or CCPA), and handling of sensitive information were not detailed in the interview.

- There was no mention of performance metrics or benchmarks for system responsiveness, load handling, or scalability under varying user loads.

- DP reps did not mention any specific accessibility features or accommodations for users with disabilities, such as screen readers or alternative input methods.

- Details about how the system handles input validation, error messages, and user feedback for invalid inputs or actions were not outlined.

- The ability for the system to function or provide limited functionality when users are offline was not discussed.

- Assuming users (both customers & system administrators) are familiar with using web browsers and mobile applications for basic tasks such as scheduling appointments and managing accounts.

- Assuming customers have access to devices (smartphones, computers) and stable internet connection to interact

- Assuming that users understand basic security practices such as safeguarding their login credentials and recognizing potential phishing attempts.

- Assuming compatibility of the system with common web browsers (Chrome, Firefox) and mobile platforms (iOS, Android) without specific compatibility testing for less common configurations.

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 might face performance issues as the user base grows if not designed with scalability in mind.

- While basic security measures are discussed, advanced threats such as zero-day vulnerabilities, insider threats, and sophisticated hacking attempts might not be fully mitigated.

- The system relies heavily on an internet connection for most functionalities, limiting use in areas with poor connectivity.

- The system's ability to easily add or remove modules and packages might be limited, requiring developer intervention for significant changes.

- Integration with DMV systems and other third-party services introduces dependencies that could affect system functionality if those services change or experience outages.
- Limited availability of skilled IT personnel might constrain the development process and maintenance efforts.
- The project timeline might be too tight to thoroughly develop, test, and deploy all desired features, leading to potential delays or compromised quality.
- Budget constraints might limit the scope of features, security measures, and performance optimizations that can be implemented.
- Dependence on current technologies might limit the system's ability to adapt to future advancements without significant rework.
- Potentially could be limitations in available technology infrastructure, such as server capacity and network bandwidth.

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.