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 design and develop a comprehensive system for DriverPass, a company dedicated to improving the way students prepare for their driving tests. The client, DriverPass, wants to provide both online learning tools and real-world driving instruction. The system will help students access online practice exams, schedule driving lessons, and communicate with staff. It will also give employees the ability to manage appointments, track student progress, and maintain vehicle schedules. DriverPass’s goal is to make the learning experience smoother, more efficient, and more accessible for all users. The system must be available online across different types of devices and should include secure, role-based access so that students, administrators, IT personnel, and secretaries can each perform their specific tasks without interfering with each other’s responsibilities.

### 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 has identified a significant challenge in the way people currently prepare for DMV driving tests. Many students fail because existing training options are limited, disconnected, or of poor quality. To solve this issue, DriverPass wants to combine digital resources with hands-on instruction. The company envisions a centralized online platform where students can practice for their written tests, book lessons with qualified instructors, and monitor their own progress. The system will also help staff handle scheduling, vehicle assignments, student communication, and other administrative tasks. By integrating these services into one platform, DriverPass hopes to offer a better learning experience, while also improving operational control behind the scenes. The main components of this system will include user accounts, scheduling functionality, lesson tracking, vehicle management, reporting tools, and secure access control.

### 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?*

The objectives below provide a specific and measurable breakdown of the system’s required functionality:

* **User Management**
  + Enable account creation and login functionality for all user roles: students, administrators, IT staff, and secretaries.
  + Implement secure authentication and role-based access permissions.
* **Scheduling System**
  + Allow students to schedule, cancel, and modify driving lesson appointments online via their personal accounts.
  + Allow secretaries to create and manage appointments for customers who prefer to schedule via phone or in-person.
  + Track driving lesson information, including student name, instructor, vehicle, date, time, and location.
* **Role-Based Access Control**
  + Grant administrators full control, including password resets, account management, and access termination.
  + Allow secretaries to view, edit, and manage appointments.
  + Allow students to view and manage their own lessons and personal profile.
* **Reporting and Logs**
  + Enable users to download appointment schedules or data summaries in common formats such as Excel.
  + Log all user activities, such as scheduling, cancellations, and modifications, for transparency and auditing.
  + Generate activity reports for administrators and IT staff to monitor system usage.
  + Implement automated data backups and provide data recovery options in case of system failure.
* **Accessibility and Usability**
  + Ensure the system is accessible on all major platforms, including desktops, laptops, tablets, and smartphones.
  + Design a user-friendly interface for both tech-savvy and less technical users.

## 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?*

Nonfunctional requirements define how the system should operate rather than what it should do. They focus on system performance, compatibility, precision, flexibility, and safety.

* The system will be a web-based application accessible through all modern browsers on both desktop and mobile devices.
* It should load key pages such as the login page, scheduling module, and profile dashboard in under two seconds under normal conditions.
* The system should support concurrent access by multiple users without performance degradation.
* Regular system updates should occur at least once per month to ensure security patches, feature enhancements, and bug fixes are implemented.
* Maintenance windows will be scheduled during low-usage hours to minimize disruption for users.
* Logging and monitoring systems shall be in place to detect and respond to slowdowns or system failures within minutes.

#### 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 application must run on both Windows and Unix-based server environments to allow flexibility in hosting choices.
* The backend will require a relational database management system such as MySQL or PostgreSQL to manage user data, appointments, and system logs.
* The frontend must be compatible with HTML5, CSS3, and JavaScript frameworks (e.g., React or Angular).
* APIs must be RESTful and designed to integrate with potential third-party services like Google Calendar or Twilio for notifications.
* RESTful APIs must be designed to enable modular integration with third-party services such as Google Calendar (for scheduling) and Twilio (for SMS/email notifications).
* The system must support mobile responsiveness using standard frameworks like Bootstrap or Tailwind CSS.

#### 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?*

* Each user will be identified by a unique username or email, and role-based identifiers will determine access levels within the system.
* Inputs will be validated for format, length, and type, and will be case-sensitive where appropriate (e.g., passwords).
* The system will send an alert to the admin in the case of data input anomalies, scheduling conflicts, or suspected security violations.
* All date/time entries will follow a standard format (e.g., ISO 8601) and include time zone information to avoid discrepancies.
* System logs will track data changes (who, what, when) for audit and troubleshooting purposes.
* The system will display user-friendly error messages to guide users toward correct input formats and prevent common input mistakes.

#### 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?*

* The admin should be able to manage user accounts (add, remove, or modify users) through the dashboard without requiring code-level changes.
* The system should be designed using modular components to support future platform updates or enhancements with minimal disruption. For example, configurations (e.g., feature toggles, thresholds) will be stored externally, allowing dynamic adjustments without code change.
* The IT admin will have full control over backend access and user role management.

#### 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 will log in using a username/email and a secure password, with an option to enable two-factor authentication.
* All communications between client and server will be encrypted using HTTPS and SSL protocols.
* In case of multiple failed login attempts (suggesting brute-force attacks), the system will lock the account and notify the user and the admin.
* Users who forget their password will be guided through a secure password reset process, involving email verification and temporary one-time tokens.
* Sessions will automatically expire after a defined period of inactivity, requiring re-authentication.

### 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.”*

This section outlines the core capabilities that the DriverPass system must provide to meet the needs of its users. These functions are essential to ensure the platform delivers a seamless, accessible, and efficient experience for students, instructors, and administrative staff.

* The system shall validate user credentials during login.
* The system shall allow students to register and create a profile.
* The system shall allow students to schedule and manage practice driving sessions.
* The system shall allow instructors to set their availability and view scheduled lessons.
* The system shall provide access to digital learning materials and practice exams.
* The system shall record and display progress metrics to both students and instructors.
* The system shall allow administrators to manage user roles and monitor system performance.
* The system shall send automated email reminders for upcoming sessions.
* The system shall log and store user activity for reporting and troubleshooting.
* The system shall allow secretaries to manage lesson schedules on behalf of students via the backend.

### 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 design must ensure ease of use, clarity, and accessibility for all user types. Each role will require a tailored interface experience that reflects their unique needs and workflows, whether accessed from a computer, tablet, or smartphone.

* The user interface should be simple and intuitive, with separate views for students, instructors, and administrators.
* Students will need access to course content, booking tools, and performance tracking.
* Instructors will require features to manage their schedule and view student history.
* The interface will support interaction via both web browsers and mobile-responsive layouts.
* Accessibility standards must be followed to accommodate users with disabilities.
* Administrators and IT staff should have access to a dashboard with system analytics and control panels.
* UI components should support real-time feedback and validations (e.g., warnings for overlapping bookings).

### 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?*

* It is assumed that users will have access to a modern internet browser and a stable internet connection.
* It is assumed that users will have basic digital literacy to navigate the interface.
* The system assumes instructors and admins will manage their schedules and data responsibly.
* It is assumed that third-party services (e.g., email providers or calendar APIs) used for notifications and integrations will be reliable and available.
* The application assumes that user feedback will be provided post-launch for future refinement.

### 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 project is constrained by budget limitations, which may impact advanced features like video tutorials or live support chat.
* The initial launch will be web-based only; a mobile app might be developed later if budget permits.
* Time constraints may limit thorough user testing before the MVP release.
* The system will initially support only English, which might limit non-English-speaking users.
* System integration with state-specific DMV APIs for automated test result syncing may not be available in the first phase.
* The system may not include AI-driven recommendations or adaptive learning until a future release.

### 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 computer

AI-generated content may be incorrect.

Note: The full-size view of this chart is available on this link:

<https://lucid.app/lucidchart/be717432-a8cd-4dda-a390-4f4ecbe67e2c/edit?viewport_loc=-1743%2C1064%2C1972%2C1140%2CuDe-dIt-NWfS&invitationId=inv_cc1db0f6-8bcf-4374-9fd7-802dd8173e94>

### Technical Requirements

The following technical requirements support the functional and nonfunctional needs of the DriverPass system as identified during the business requirements phase.

#### ****Hardware and Software Environment****

* The system shall be hosted on cloud infrastructure (e.g., AWS, Azure) with scalable resources.
* The backend must be compatible with both Windows Server and Unix-based hosting environments.
* Minimum recommended server specs: 4-core CPU, 8GB RAM, 100GB SSD for production.
* Client devices must support modern browsers (Chrome, Firefox, Safari, Edge) on desktop and mobile.

#### ****Database and Backend Tools****

* A relational database such as **PostgreSQL** or **MySQL** will be used to manage users, lessons, vehicles, and logs.
* Backend services shall be implemented using RESTful APIs developed in **Node.js**, **Python (Django/Flask)**, or **Java (Spring Boot)**.
* Data access and business logic will be separated using an MVC or layered architecture.

#### ****Frontend Tools and Frameworks****

* The frontend will be web-based using **React.js** or **Angular**, with a responsive design framework (e.g., **Bootstrap** or **Tailwind CSS**) to ensure mobile accessibility.
* Interfaces will follow **WCAG 2.1** accessibility standards and support dynamic validation and real-time user feedback.

#### ****Performance and Uptime****

* Core user-facing pages (e.g., login, scheduling, profile) must load in under 2 seconds under normal server load.
* The system must support at least 200 concurrent users without degradation.
* 99.9% system uptime is required, excluding scheduled maintenance.
* Scheduled maintenance and system updates will occur during off-peak hours (e.g., weekends at night).

#### ****Security Requirements****

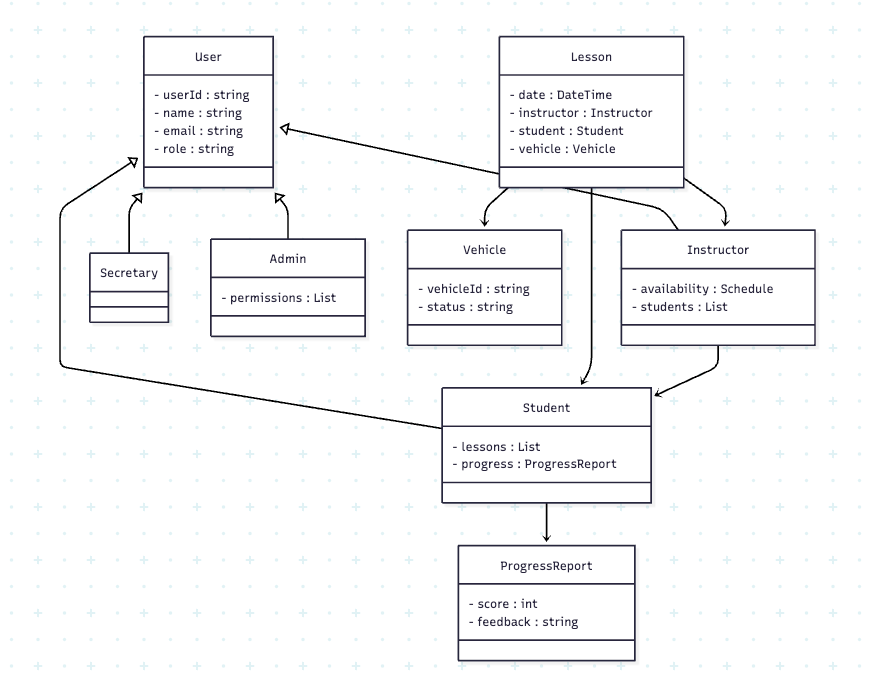
* All data in transit shall be encrypted using **SSL/TLS** (HTTPS protocol).
* Passwords must be hashed using secure algorithms (e.g., bcrypt or Argon2).
* Support for **two-factor authentication (2FA)** for admins and users who opt in.
* Login systems must implement account lockout policies after 5 failed login attempts, with admin notification.
* Password recovery will use secure, token-based reset links with time-limited validity.

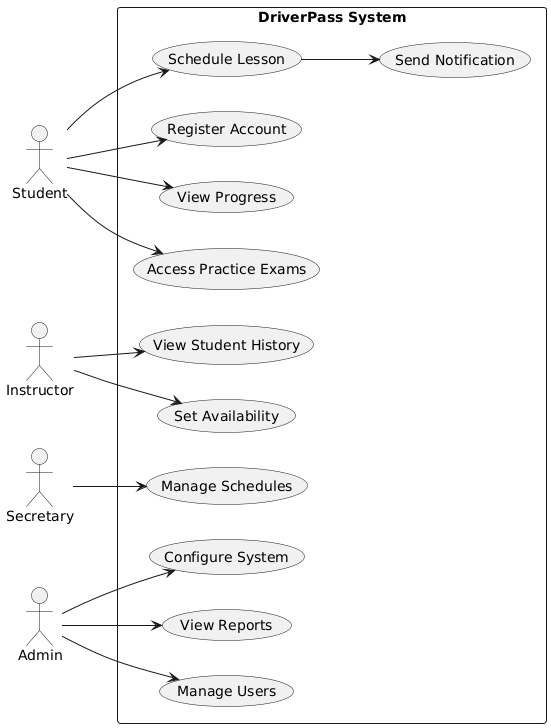
#### ****Monitoring and Logging****

* All API calls and user actions shall be logged with timestamp, user ID, and action details.
* System logs will be stored for at least 6 months and accessible by admins.
* Automated monitoring tools (e.g., AWS CloudWatch, New Relic) will be used to monitor performance and detect errors.

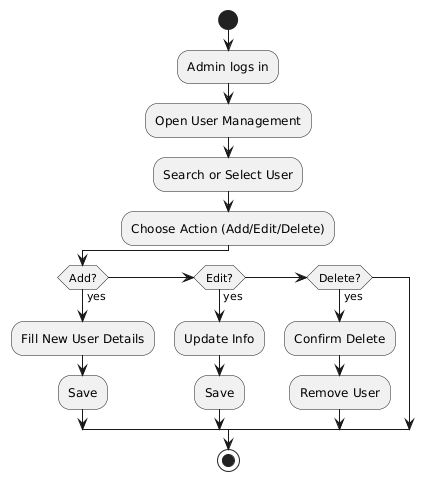
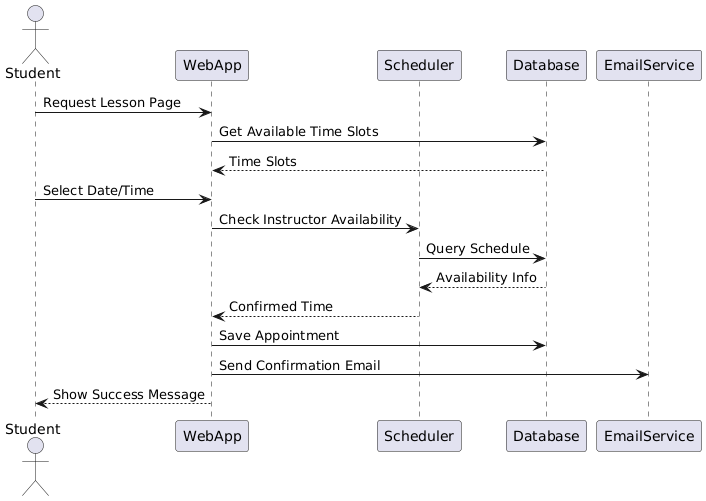
#### ****Adaptability and Maintenance****

* Admin users will be able to add, update, or remove users via an admin dashboard without changing code.
* Feature toggles and configuration files will allow changes to system settings without redeployment.
* RESTful APIs and microservices will allow future integration with external tools such as DMV systems, calendar sync, or SMS/email alerts.

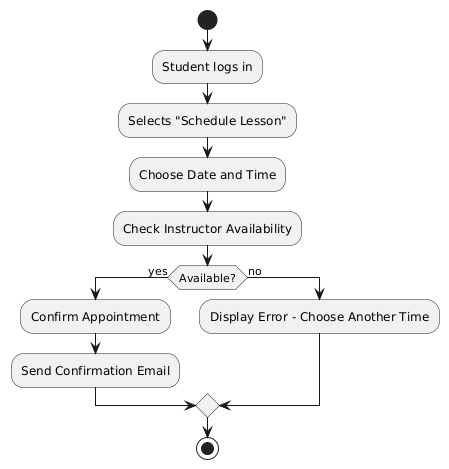
**UML Class Diagram**

**UML Use Case Diagram**

**UML Activity Diagram**

**Use Case: Manage User Roles**

**UML Activity Diagram**

**Use Case: Schedule Driving Lesson**

**UML Sequence Diagram**

**Use Case: Schedule Driving Lesson**