CS 255 DriverPass Business Requirements Document

**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 to address the existing gap in driver training tools. The system aims to provide students with access to online practice exams and on-the-road training to better prepare them for their driving tests.

**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 intends to create a system that offers online practice exams and on-the-road training to improve the success rate of students in passing their driving tests. The system needs to facilitate user registration, scheduling of driving lessons, matching students with instructors and vehicles, providing access to online study materials, tracking user progress, and generating reports.

DriverPass envisions a platform where students can conveniently access study materials, schedule driving lessons, track their progress, and receive personalized guidance from experienced instructors. The system should cater to the needs of various stakeholders including students, instructors, and administrators, providing a seamless and efficient user experience.

**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 system should:

* Improve the success rate of students in passing driving tests by providing comprehensive training resources.
* Enhance user convenience by offering online practice exams and study materials accessible anytime, anywhere.
* Streamline the process of scheduling driving lessons and matching students with instructors and vehicles.
* Facilitate progress tracking and performance analysis to identify areas for improvement and optimize training effectiveness.

**Requirements**

**Nonfunctional Requirements**

**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?

* The system should be accessible via web browsers and mobile devices to ensure widespread availability and accessibility. Devices such as laptops/desktops, smartphones, and tablets.
* Response times for system interactions should be fast to provide a seamless user experience, with page loads and data retrieval occurring within seconds.
* Regular updates and maintenance should be performed to keep the system running efficiently and to address any performance issues promptly.

**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 be compatible with multiple platforms including Windows, macOS, and Linux to accommodate a diverse user base. It should also be able to handle iOS and Android for mobile versions.
* Multiple different back-end resources would be needed to support the application, such as a database, web server, application server, programming frameworks, security tools, monitoring and logging, and backup/recovery options.

**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?

* Users should be distinguished by using user authentication. This would require creating an account with unique credentials like username and password. This would be verified against stored user data ensuring no duplication and that when logging in, it is the correct user info. For further security, it can be required that usernames and passwords are a certain character length, and have uppercase/lowercase letters, numbers, and special characters. There can also be different roles and permissions. Student vs Teacher will give different access and usability based on the role of the user.   
  User inputs should be accurately processed to ensure correct scheduling and tracking of driving lessons, with validation checks implemented to prevent errors and inconsistencies.
* The system should promptly notify administrators of any errors or issues encountered during operation, allowing for timely resolution and minimal disruption to user activities.

**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 system should allow for easy modification of user roles and permissions to accommodate changes in the organizational structure and staffing. This can be done by having a user management interface that allows admins to add, remove, or modify user roles and permissions. There should not be a need to dig into the code or database itself to make these changes.
* IT admins may need additional access such as access to define and assign roles and permissions to users based on responsibilities. Ability to configure the system settings, parameters, and preferences. And the ability to perform maintenance such as monitoring, diagnosing and resolving issues, updating software, and backing up data.
* Updates to the platform should be seamlessly integrated to ensure continuous operation, with backward compatibility maintained to minimize disruptions for existing users.

**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 authentication should be secure to protect sensitive information, with multi-factor authentication and encryption protocols implemented to safeguard user credentials and data. They should be required to log in with a username and password and a 6-digit one-time passcode.
* Data exchange between the client and server should be encrypted using industry-standard encryption algorithms (e.g., SSL/TLS) to prevent unauthorized access and data breaches.
* Measures should be implemented to mitigate potential security threats such as brute force attacks. The system should lock the account out from further attempts to log in when a brute force attempt is detected. An alert should be emailed to the users account notifying them of this.
* If a user forgets their password, they should be presented with an option to reset the password. The account should lock due to too many failed attempts and provide the option to reset the password. If selected a password reset email would be sent to the user with a secure link to reset 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.”

* The system shall allow users to register for accounts and log in securely, with password hashing and salting techniques employed to protect user credentials.
* The system shall allow users to be able to schedule driving lessons, specifying preferred dates, times, and instructors, with real-time availability updates to make decisions.
* The system shall match students with available instructors and vehicles based on scheduling preferences, considering factors such as proximity, availability, and instructor qualifications.
* The system shall offer online practice exams and study materials should be accessible to registered users, with progress tracking features to monitor completion status and performance.
* The system shall show student progress and reporting features to instructors and administrators, allowing for performance analysis and feedback.

**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 intuitive and user-friendly, with clear navigation menus, interactive elements, and responsive design to accommodate users with varying levels of technical expertise. It should also include accessibility features for a range of possible disabilities.
* Different user roles (students, instructors, administrators) should have access to tailored interfaces with relevant functionalities, with role-based permissions enforced to restrict access to sensitive features and data.
* The interface should be accessible via web browsers on desktop and mobile devices, with cross-browser compatibility and support for popular screen sizes and resolutions.

**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?

* Users have basic internet connectivity and access to modern web browsers (e.g., Google Chrome, Mozilla Firefox, Safari).
* Third-party services may be integrated for payment processing (e.g., Stripe, PayPal) and compliance updates (e.g., DMV regulations, driving test requirements).
* Users have access to a range of different devices, from a smartphone to a tablet, laptop, or desktop.

**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?

* Resource constraints may impact the speed of development and deployment, with project timelines subject to available manpower and budget allocations.
* Budget limitations may restrict the implementation of certain features or integrations, with prioritization required to ensure alignment with project goals and stakeholder expectations.
* Time constraints and project timelines may be constrained by deadlines or external factors.
* Scalability and expansion of the system are reliant on the above limitations.

**Gantt Chart:**

****

