# CS 255 Business Requirements Document - Joseph Dengler

## 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 driver training system for DriverPass.
* The client, DriverPass, aims to address the current gap in driver training, providing online classes, practice tests, and on-the-road training to enhance students' preparation for 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 System Functionality:
  + Provide online classes and practice tests.
  + Facilitate scheduling and management of on-the-road driving lessons.
  + Support different user roles: Owner, IT Officer, Secretary, and Students.
  + Ensure security measures, access controls, and tracking of user activities.
  + Offer flexible driving lesson packages with customization options.
  + Capture and manage customer information for registration and scheduling.
  + Establish a connection with the DMV for updates on rules, policies, and sample questions.
  + Implement a cloud-based system for accessibility and data security.
* Problem Statement:
  + Current driver training methods result in a high percentage of students failing their driving tests.
  + Limited availability of comprehensive online practice exams and on-the-road training.
* System Components:
  + User Management Module
  + Lesson Scheduling and Reservation Module
  + Online Classes and Practice Tests Module
  + Security and Tracking Module
  + Customer Registration Module
  + DMV Compliance Module
  + Cloud-Based System Infrastructure

### 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?*

* System Design Objectives:
  + Develop an intuitive interface reflecting Liam's vision.
  + Ensure secure and efficient access controls for different user roles.
  + Implement a flexible system architecture to accommodate future updates.
  + Provide seamless integration between online and offline data access.
  + Enable tracking and reporting features for user activities.
  + Establish a reliable connection with the DMV for real-time updates.
  + Create a cloud-based system to ensure accessibility and data security.
* Measurable Tasks:
  + Design and implement user interfaces for online classes, practice tests, and appointment scheduling.
  + Develop secure authentication and authorization mechanisms.
  + Create a flexible database schema supporting dynamic lesson packages.
  + Establish communication channels with the DMV for data updates.
  + Implement tracking mechanisms for user activities and reservations.
  + Conduct user testing to ensure system usability and reliability.
  + Provide documentation for system maintenance and future enhancements.

## 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?*

* Environments:
  + The system should run in a web-based environment, accessible from any computer or mobile device with internet connectivity.
  + Online classes and practice tests should load efficiently, ensuring a seamless user experience.
* System Speed:
  + Online classes and practice tests should load within 5 seconds to provide a responsive experience for users.
  + The scheduling and reservation system should process requests within 2 seconds.
* Update Frequency:
  + The system should have real-time updates for DMV rules, policies, and sample questions.
  + Regular system updates, including bug fixes and feature enhancements, should occur quarterly.

#### 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?*

* Supported Platforms:
  + The system should be compatible with major web browsers such as Chrome, Firefox, Safari, and Edge.
  + It should be platform-independent, running seamlessly on Windows, macOS, and Linux operating systems.
* Back-End Requirements:
  + The system's back end requires a robust and scalable database to manage user data, reservations, and lesson information.
  + Database platform: MySQL or PostgreSQL.
  + Cloud infrastructure: AWS or Azure.
* Development Tools:
  + The development environment should support common web development tools and frameworks such as React for the front end and Django for the back end.
  + Version control system: Git.
* Security Measures:
  + Implement encryption protocols (SSL/TLS) for secure data transmission.
  + Regularly update and patch the server operating system and software.
* Backup and Recovery:
  + Regularly backup the system data to prevent data loss.
  + Implement a disaster recovery plan to restore system functionality in case of failures.

#### 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 Identification:
  + Distinguish between users through unique usernames or email addresses.
  + Implement case-insensitive user identification to enhance user experience.
* Input Case-Sensitivity:
  + User inputs for usernames and passwords should be case-insensitive to prevent login issues.
* Problem Notification:
  + The system should inform the admin of a problem when:
  + Multiple consecutive login failures occur.
  + Data inconsistencies or errors are detected during reservation or lesson scheduling.
  + System updates from the DMV fail, affecting the accuracy of practice tests.

#### 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?*

* Changes Without Code Modification:
  + Allow administrators to add, remove, or modify user roles and permissions without changing the system's core code.
  + Implement a user management interface for easy administration of user accounts.
* Platform Adaptation:
  + Design the system to be modular, allowing easy adaptation to platform updates without major code changes.
  + Regularly monitor and incorporate updates from web development frameworks (React, Django) to maintain compatibility.
* Access Requirements:
  + The IT admin should have full access to user accounts, system configurations, and the database.
  + Ensure that the IT admin can modify roles, reset passwords, and manage security settings.

#### 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 valid combination of username/email and password for login.
  + Implement secure password policies (e.g., minimum length, complexity requirements).
* Data Exchange Security:
  + Use SSL/TLS encryption for securing data exchange between the client and the server.
  + Employ HTTPS protocols to protect sensitive information during transmission.
* Handling Brute Force Attacks:
  + Implement account lockout mechanisms after a predefined number of unsuccessful login attempts.
  + Notify the admin of multiple failed login attempts for further investigation.
* Forgotten Password:
  + Allow users to initiate a password reset through a secure email verification process.
  + Send a temporary password or a password reset link to the user's registered email.

### 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 during the login process.
* The system shall provide different user roles: Owner, IT Officer, Secretary, and Students.
* The system shall allow users to schedule, modify, and cancel driving appointments online.
* The system shall support the creation and customization of flexible driving lesson packages.
* The system shall capture and store customer information, including name, address, phone number, and payment details.
* The system shall track and report user activities, including reservations, modifications, and cancellations.
* The system shall establish a secure connection with the DMV for real-time updates on rules, policies, and sample questions.
* The system shall run on a cloud-based infrastructure for accessibility and data security.
* The system shall provide a user interface that reflects Liam's vision, including progress tracking and driver notes.
* The system shall load online classes, practice tests, and scheduling interfaces within 5 seconds for optimal user experience.

### 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.)?*

* Owner (Liam):
  + Access and modify all system features.
  + Customize and disable driving lesson packages.
  + View progress tracking and driver notes.
* IT Officer (Ian):
  + Full access to all accounts for maintenance purposes.
  + Reset passwords and manage user accounts.
* Secretary:
  + Answer phone calls, make appointments, and manage user schedules.
  + Schedule, modify, and cancel appointments on behalf of users.
* Students:
  + Register for lessons and access online classes/practice tests.
  + Schedule, modify, and cancel driving appointments.
* Device Compatibility:
  + The interface should be accessible from both desktop browsers and mobile devices.
  + Responsive design to accommodate various screen sizes.
* Input Forms:
  + Intuitive input forms for user registration, lesson scheduling, and modification.
  + Clearly defined progress tracking and driver notes sections.
* Navigation:
  + Easy navigation between different modules (scheduling, lessons, practice tests).
  + Clear call-to-action buttons for appointment scheduling and modification.

### 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 Connectivity:
  + Assumes users have a stable internet connection for accessing online classes and scheduling appointments.
  + Assumes users have basic knowledge of online interfaces and form submissions.
* Device Compatibility:
  + Assumes users have access to common web browsers (Chrome, Firefox, Safari, Edge) for optimal system performance.
* DMV Connection:
  + Assumes the DMV provides a reliable and secure API for real-time updates.
  + Assumes the DMV updates do not significantly impact system performance.

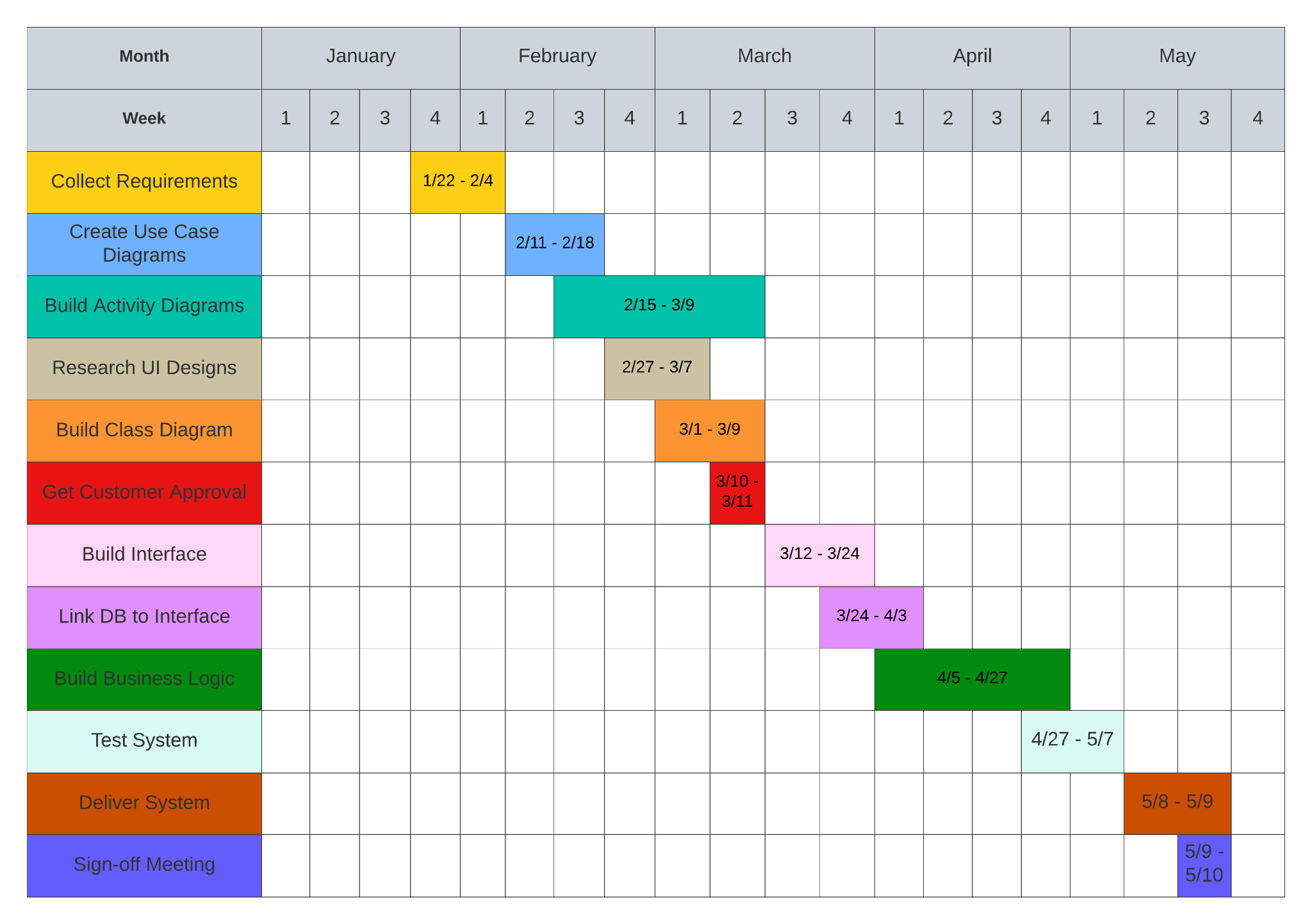
### Limitations

*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:
  + Limited budget and time may impact the depth of customization for driving lesson packages.
  + Limited resources may restrict the frequency of system updates and enhancements.
* Technology Limitations:
  + Compatibility issues with future web development frameworks could affect system adaptation.
  + Budget constraints may limit the selection of advanced security measures.
* User Training:
  + Assumes minimal user training requirements for basic system interactions.
  + Limited resources for extensive user education programs.

### 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.*

  
  
  
  
  
  
  
**CITATIONS**

* Valacich, Joseph S., and Joey F. George. Modern Systems Analysis and Design. Vol. 9, Pearson, 2020.
* Dennis, Alan. Systems Analysis and Design with UML. 4th ed., Wiley, 2012