# CS 255 System Design Document Template

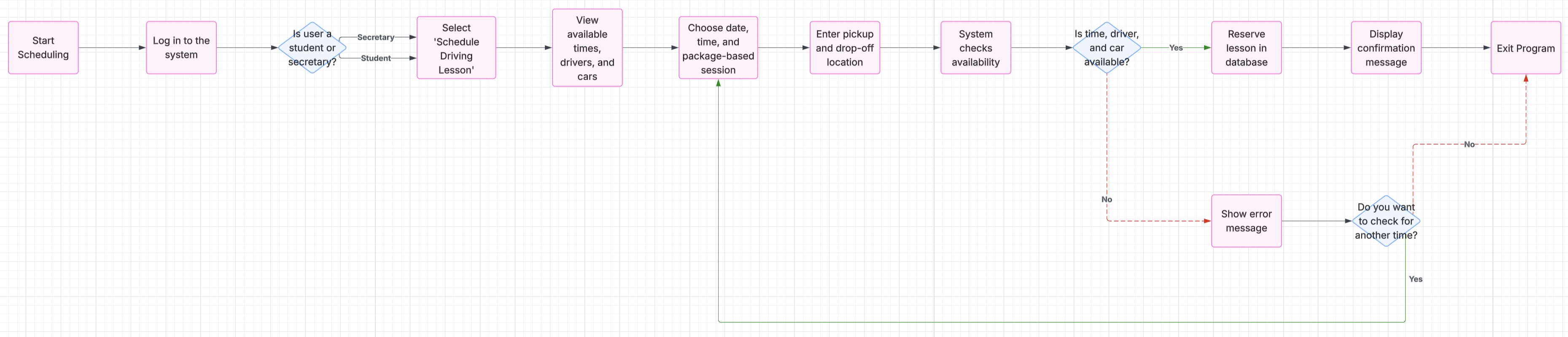
This template lays out all the different sections that you need to complete for Project Two. Each section has guidance to prompt your thinking. You will need to continually reference the interview transcript as you work to make sure that you are addressing your client’s needs. There is no required length for the final document. Instead the goal is to complete each section based on what your client’s needs are. Remove this note when you are finished, and replace all bracketed text with the relevant information.

## UML Diagrams

### UML Use Case DiagramScreenshot 2025-08-17 at 10.15.56 AM.png

### UML Activity Diagrams

Activity 1: Scheduling a Driving Lesson



*Activity 2: Manage Account Password Update and Access*

### CS255 Project 2 - Page 3.pdfCS255 Project 2 - Page 4.pdfUML Sequence Diagram

### UML Class DiagramCS255 Project 2 - Page 5.pdf

## Technical Requirements

* Hardware Requirements
  + Devices
    - Desktop or laptop computers are required for secretary use within the office environment.
    - Students can access the system via smartphones, tablets, or computers using a supported web browser.  
      Devices must support modern browsers and maintain a stable internet connection.
  + Server Infrastructure
    - The system should be hosted on centralized or cloud-based servers with adequate performance and storage capacity.
    - The infrastructure must support real-time user interactions and ensure reliable storage of user data, scheduling records, and system logs.
* Software Requirements
  + Front-End (Client-Side)
    - The system should support all modern browsers such as Chrome, Firefox, Safari, and Edge.
    - A web application should be built using a framework like React.js, Angular, or Vue.js.
    - Optional mobile access may be provided using frameworks like Flutter, React Native, or native development tools for Android and iOS.
  + Back-End (Server-Side)
    - Handles the behind-the-scenes logic and processing of the system
    - Built using reliable and well-supported server technologies
    - Ensures smooth communication between the user interface and the database
    - Protects user accounts with secure login and authentication methods
    - Makes sure only authorized users can access or update information
  + Database
    - Stores all important data such as user details, schedules, and bookings
    - Uses a structured and reliable database system to keep information organized
    - Can also use a secondary system to temporarily store frequently accessed data for faster performance
    - Ensures data is easy to access, update, and manage securely
* Tools and Utilities
  + Development Tools
    - Developers should use code editors to write and manage the system’s code efficiently
    - Tools like Git help keep track of changes and allow teams to work together smoothly
    - Development environments should be consistent to avoid errors when moving between testing and live systems
    - Automated tools can be used to test and update the system quickly and reliably
  + Testing Tools
    - Unit testing should be conducted using tools like Jest, PyTest, or JUnit.
    - End-to-end and UI testing can be performed with Cypress or Selenium.
    - API testing should be supported by tools like Postman, Swagger, or Insomnia.
* Infrastructure Requirements
  + Hosting and Deployment
    - The system should be deployed on cloud infrastructure such as AWS, Azure, Google Cloud, or DigitalOcean.
    - It should include compute resources, managed database services, and cloud storage.
    - Backup and disaster recovery mechanisms must be in place.
  + Network and Access
    - The system should use secure connections to protect data as it travels over the internet
    - A registered domain name is needed so users can easily access the system online
    - A load balancer should be in place to handle multiple users at the same time without slowing down the system
  + Security Requirements
    - Different types of users (Students, Secretaries, Admins) should have different levels of access based on their role
    - The system should check and clean all user inputs to prevent misuse or hacking
    - User sessions should be protected with secure login tokens
    - All sensitive data should be encrypted during transmission and when stored
    - Important system actions should be recorded for tracking and accountability
    - The system should follow privacy laws and regulations, depending on where it is used
  + Scalability and Performance
    - The system should be able to grow and handle more users and activity over time
    - Use temporary storage to speed up frequent data requests and reduce strain on the main database
    - Background tasks that take time should run separately without slowing down the main system
    - Regular checks and tests should be done to make sure the system runs smoothly and quickly