# CS 255 System Design Document

## UML Diagrams

### UML Use Case Diagram

Figure 1

*DriverPass System Use Case Diagram*

*A diagram of a driver pass system

Description automatically generated*

*Note.* This use case diagram depicts a simplified reference to how each user will interact with the system.

### UML Activity Diagrams

Figure 2

*DriverPass System Activity Diagram: View Lesson Details*

*A diagram of a process

Description automatically generated*

*Note.* This activity diagram depicts the flow for a user to view lesson details.

Figure 3

*DriverPass System Activity Diagram: Booking Process*

A diagram of a process

Description automatically generated

*Note.* This activity diagram depicts the process for a student booking a lesson.

### UML Sequence Diagram

Figure 4

*DriverPass System Sequence Diagram*

*A screenshot of a computer screen

Description automatically generated*

*Note.* This sequence diagram shows the booking lesson process between each core element of the system.

### UML Class Diagram

Figure 5

*DriverPass System Class Diagram*

*A diagram of a program

Description automatically generated*

*Note.* This Class diagram depicts the relationship between each element of the system for the booking process.

## Technical Requirements

## The DriverPass system relies on key hardware, software, and infrastructure to address its technical requirements. Starting with the hardware, the system's back end consists of cloud-based Azure servers, databases, and applications. The cloud platform enables scalability, allowing the system to grow as demand increases. High-speed internet with low latency is another key component, ensuring that users experience seamless performance, which is important for a system designed to handle real-time interactions, like scheduling and online testing.

## Furthermore, a secure and scalable storage solution is required to manage the data, including user information and transaction records. Finally, the system needs to be accessible via desktop computers and mobile devices that run on Windows, Unix, Android, or iOS, ensuring compatibility across access points.

On the software side, the system must be flexible, supporting Windows and Unix operating systems to cater to a broad user base. It will also need to use Azure’s cloud-based database, CosmosDB, which is critical for managing the data that DriverPass will handle, such as user profiles, lesson schedules, and transaction histories. The system's user interface is built using web framework Django, while the mobile framework Flutter ensures that mobile users have a responsive experience. For secure and efficient communication between the system’s frontend and backend, RESTful APIs are implemented, supported by robust authentication tools like OAuth 2.0, which manage user access and security. OAuth 2.0 will be abstracted as we implement Microsoft Authentication Library (MSAL) through the cloud back end to authenticate user requests. To maintain the system’s health and performance, monitoring tools through Azure will be used, providing real-time insights and logging capabilities. Security is further bolstered by encryption libraries that protect data during transmission and storage, and DevOps tools such as Jenkins or GitLab CI automate testing, deployment, and system updates, ensuring that the system remains resilient and up to date.

The infrastructure for DriverPass is designed for flexibility and reliability. Cloud computing platforms are at the heart of this infrastructure, offering the scalability and flexibility needed to handle varying loads and ensure consistent availability. The Azure Load balancer will be employed to distribute incoming traffic evenly across servers, which is crucial for maintaining consistent performance under high traffic times. To enhance content delivery, the Azure Content Delivery Network (CDN) will be incorporated, speeding up access to resources regardless of the user's location. The system’s security will be reinforced by a comprehensive security infrastructure, including firewalls, intrusion detection systems, and VPNs, which protect both the network and the data being exchanged. Scaling infrastructure ensures that the system can automatically adjust resources based on real-time demand, preventing bottlenecks and maintaining performance.

By culminating all these hardware, software, and infrastructure components, the DriverPass system is prepared to securely and efficiently manage user data, perform reliably under varying conditions, and adapt to future changes; all while providing an intuitive and responsive user experience across multiple platforms.