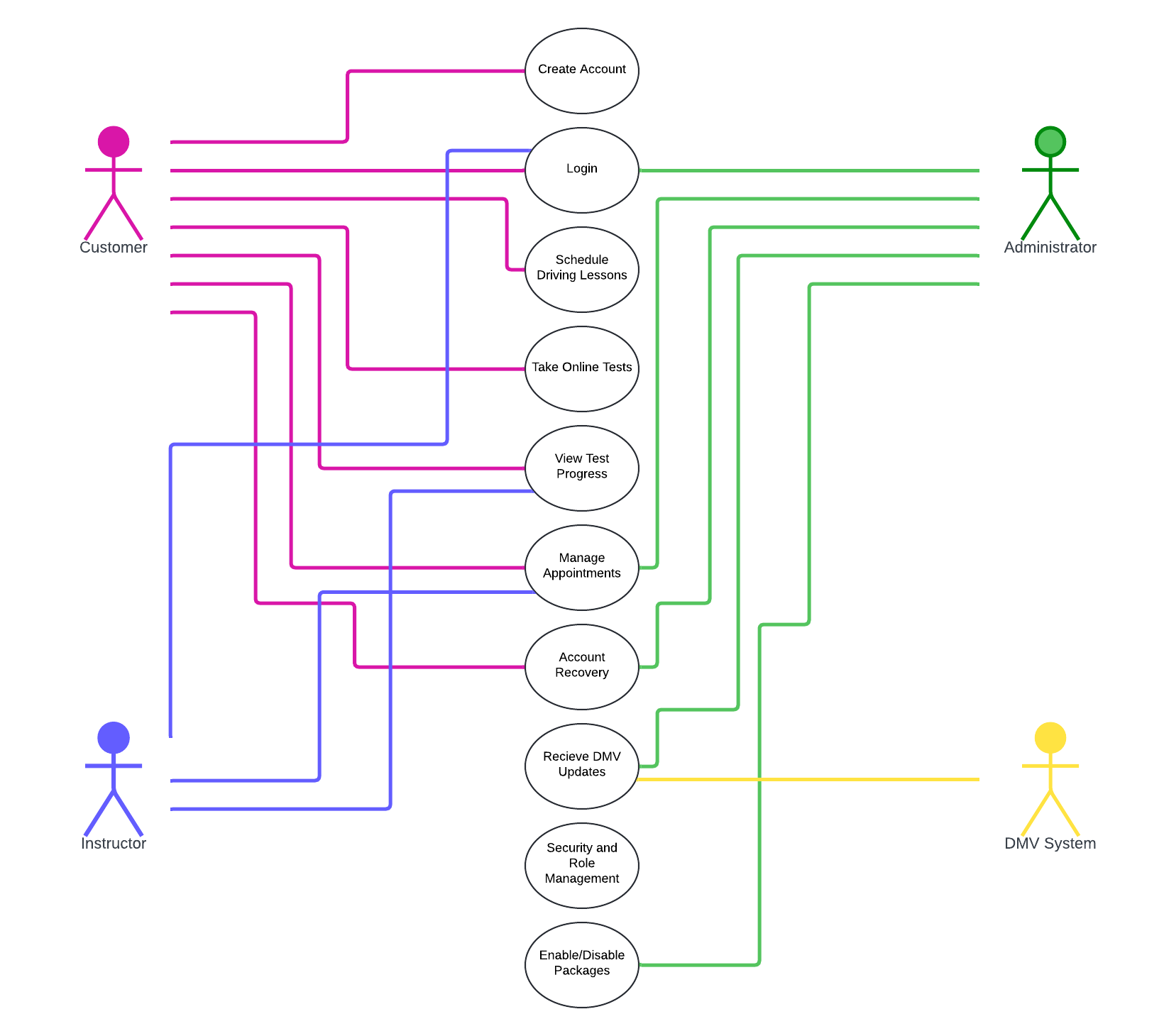
# CS 255 System Design Document Template

Created By: Andrew Torrez

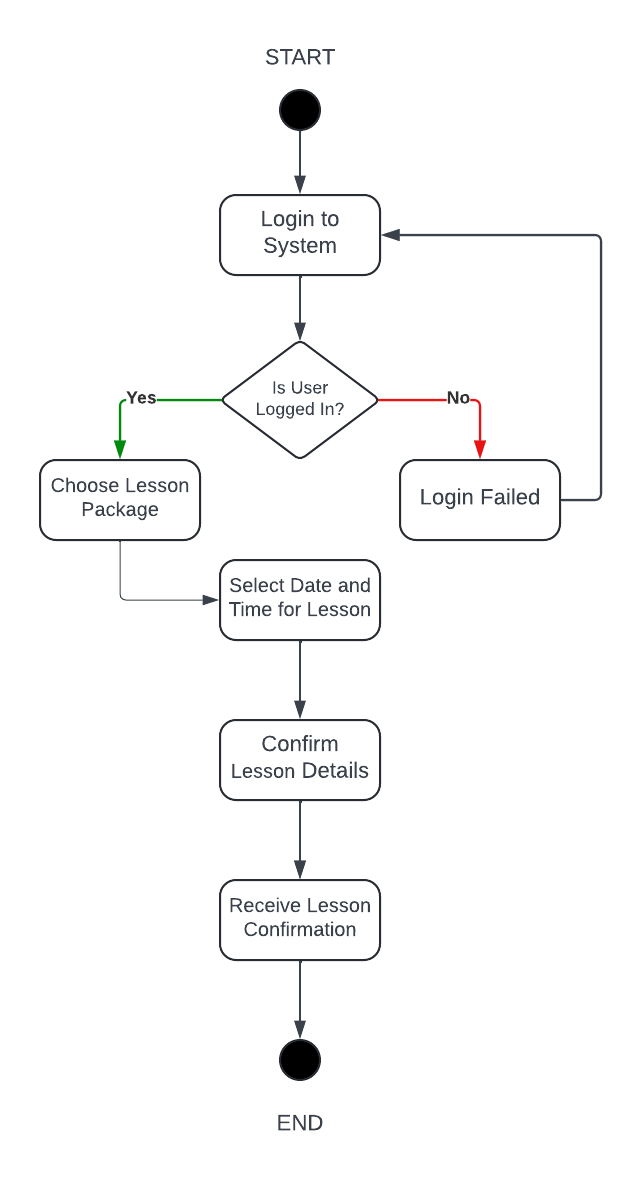
Date: August 22nd, 2024

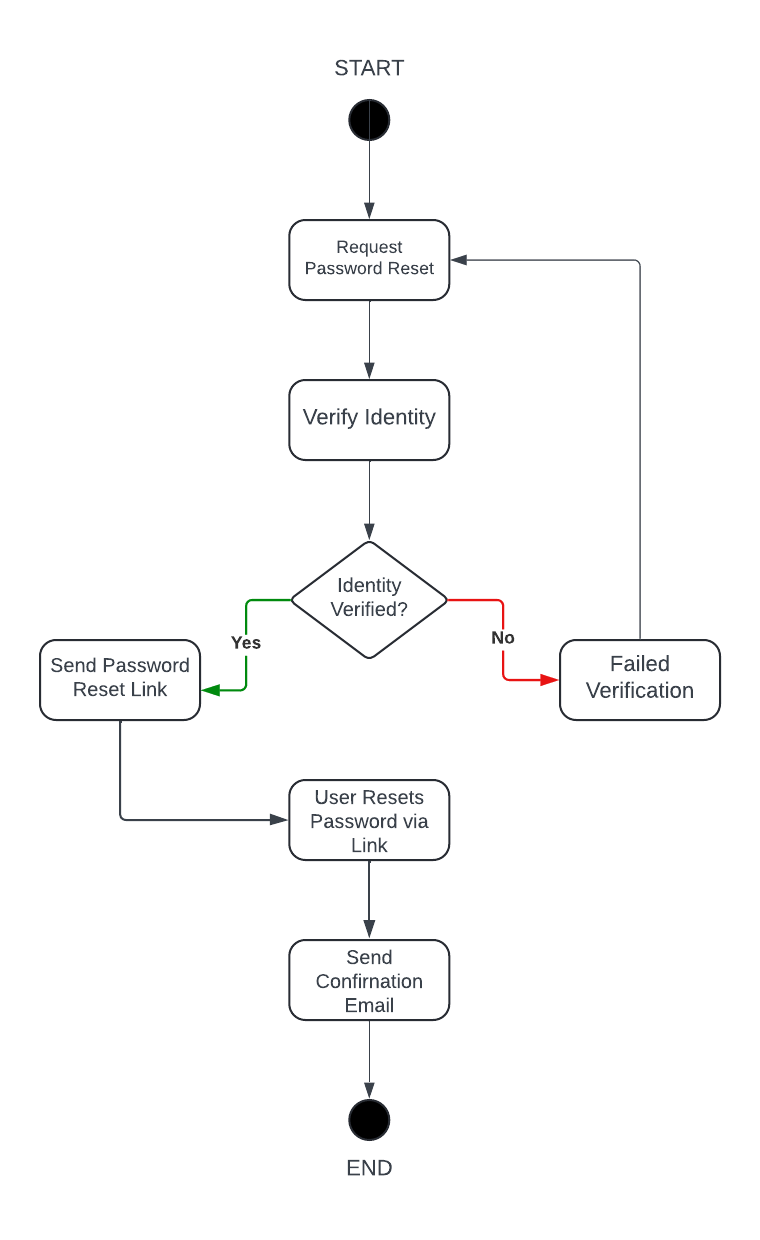
## UML Diagrams

### UML Use Case Diagram

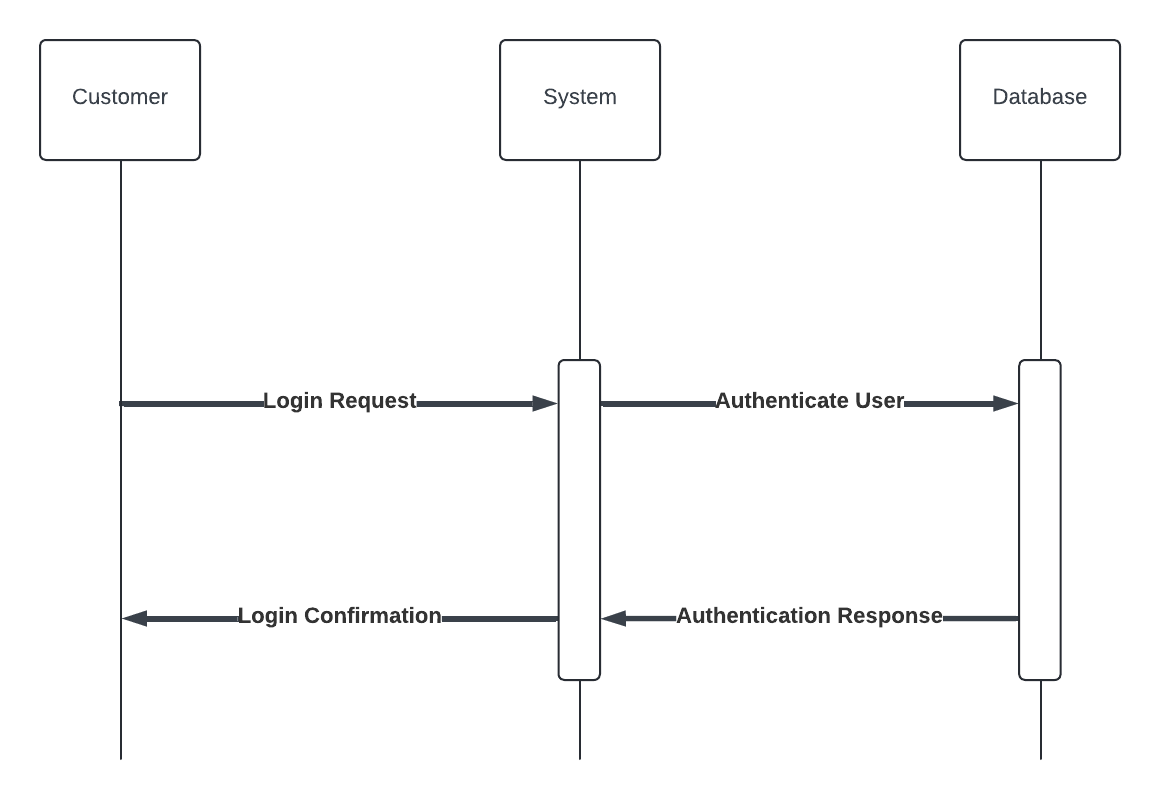


### UML Activity Diagrams

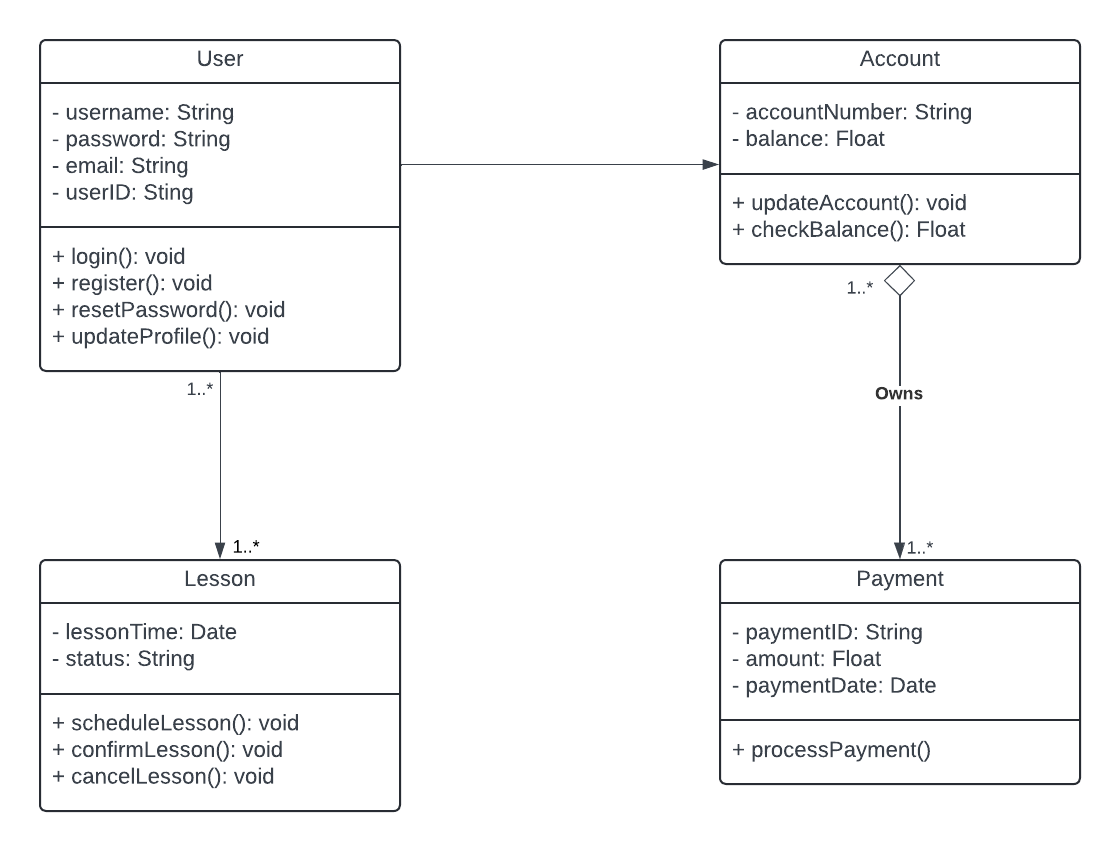




### UML Sequence Diagram



### UML Class Diagram



## Technical Requirements

1. Hardware Requirements

Server Infrastructure:

* Application Server: A high-performance application server to handle the business logic, user requests, and interactions with the database.
* Database Server: A robust database server with sufficient storage capacity to manage user data, lesson schedules, payments, and related information.
* Backup and Redundancy: A backup server for data redundancy to ensure system availability and disaster recovery.

User Devices:

* Desktops/Laptops: For users (students, instructors, and administrators) to access the system via a web-based or native desktop application.
* Mobile Devices: Support for smartphones and tablets to allow users to access the system on the go.

2. Software Requirements

Operating Systems:

* Server OS: Linux or Windows Server for hosting the application and database servers.
* Client OS: Windows, macOS, iOS, Android, and Linux for enduser devices to access the system.

Database Management System (DBMS):

* Relational Database: MySQL, PostgreSQL, or another relational database system to store user data, lesson schedules, and payment information securely.

Application Development Framework:

* Backend Development: Spring Boot (Java) or Django (Python) for handling serverside logic, authentication, and user management.
* Frontend Development: React.js or Angular.js for building the user interface, allowing users to interact with the system seamlessly.

Web Server:

* Apache or Nginx: For serving the web application and handling HTTP requests.

3. Tools

Development Tools:

* IDE: IntelliJ IDEA, Visual Studio Code, or Eclipse for developing and maintaining the codebase.
* Version Control: Git and GitHub or GitLab for source code management and collaboration among the development team.
* CI/CD Pipeline: Jenkins, CircleCI, or GitHub Actions for automating testing, deployment, and integration processes.

Design Tools:

* UML Design: Lucidchart or Visual Paradigm for creating UML diagrams, including use case, activity, sequence, and class diagrams.

4. Infrastructure Requirements

Cloud Infrastructure:

* Cloud Hosting: AWS, Azure, or Google Cloud for hosting the application and database servers. This allows for scalability, ensuring the system can handle a growing user base without performance degradation.
* Storage: Cloud-based storage solutions (e.g., AWS S3) for securely storing usergenerated content and backups.

Network Infrastructure:

* Highspeed Internet Connection: Reliable network infrastructure to ensure fast and secure access to the system for all users.
* Firewalls and Security Measures: Network security measures, including firewalls, to protect the servers from unauthorized access and attacks.

Authentication and Security:

* SSL/TLS Certificates: To ensure all data transmitted between the client and server is encrypted and secure.
* Authentication Systems: OAuth 2.0 or another robust authentication protocol to handle user login and authorization.
* Data Encryption: AES encryption for sensitive data stored in the database, such as user credentials and payment information.