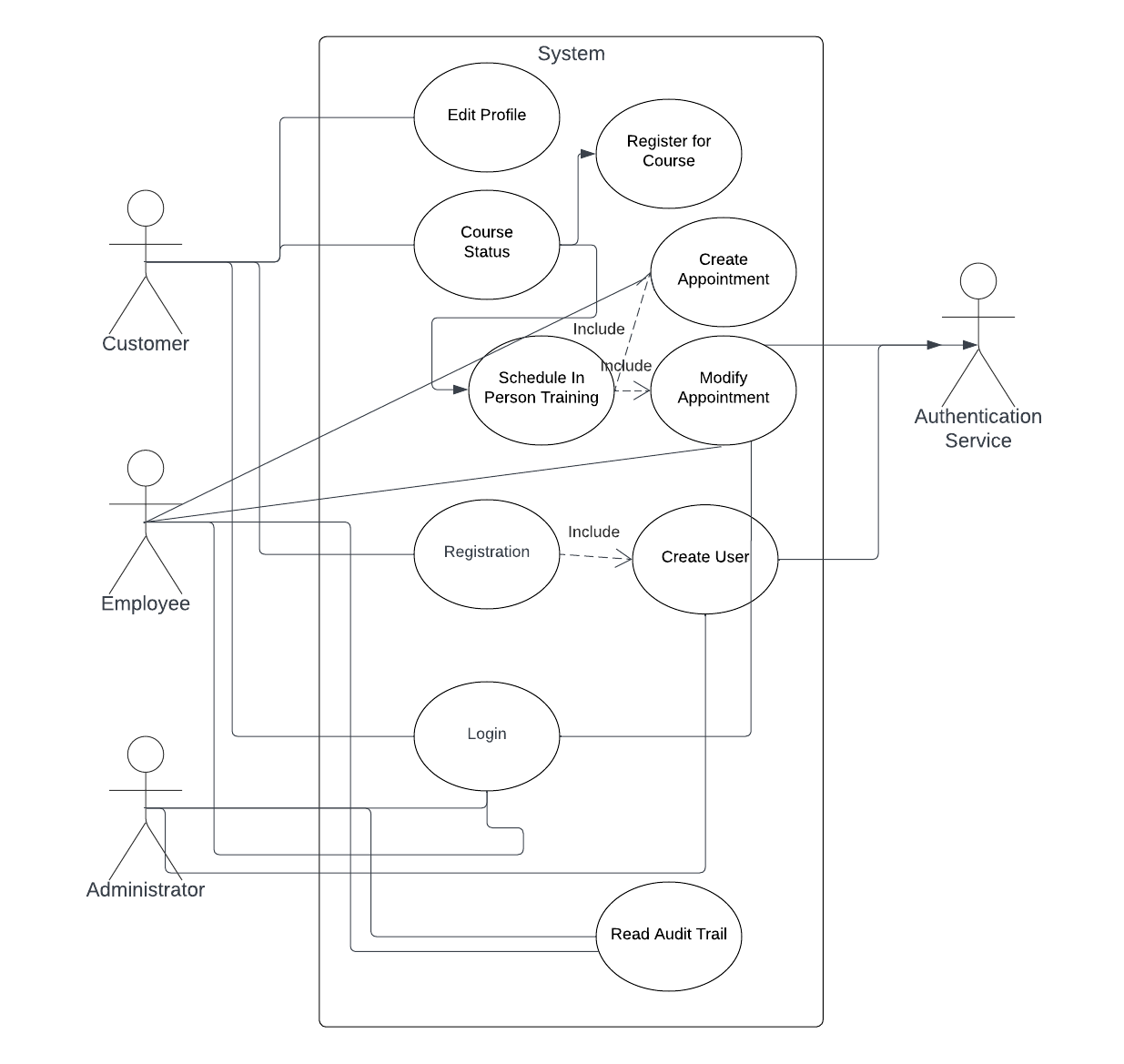
# CS 255 System Design Document Template

## UML Diagrams

### UML Use Case Diagram

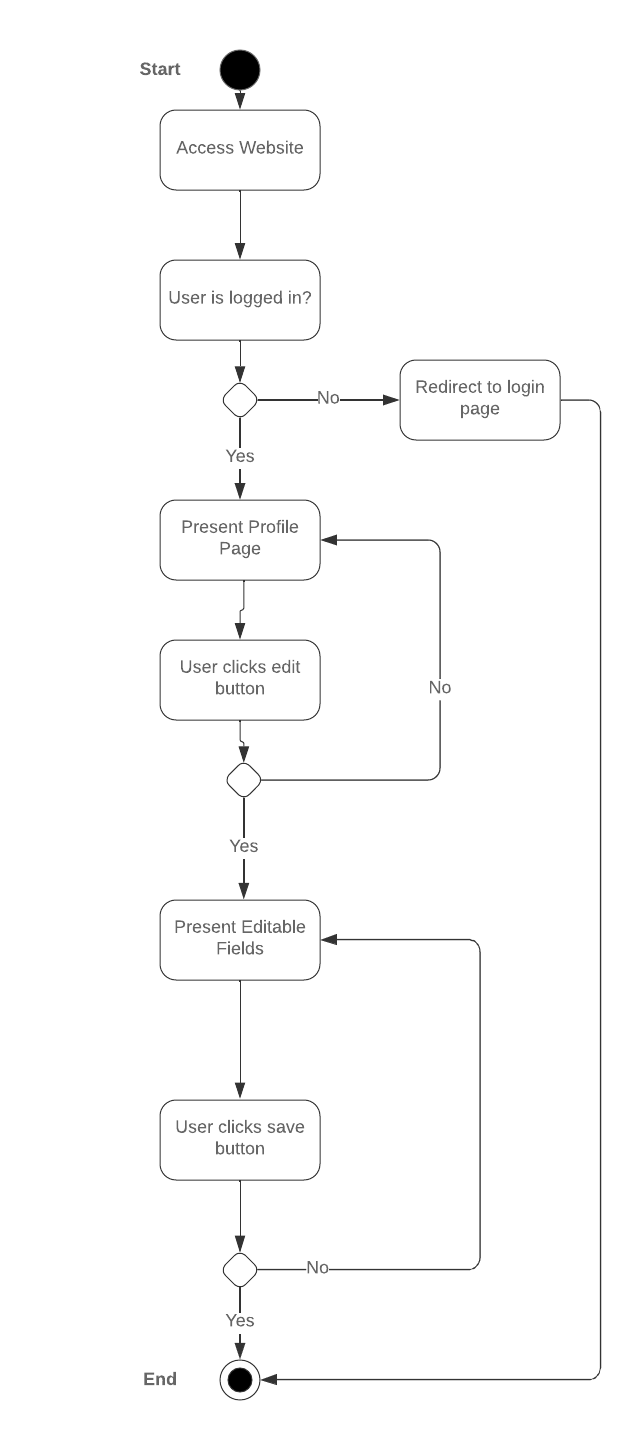
**

### UML Activity Diagrams

**Activity Diagram: User Login**

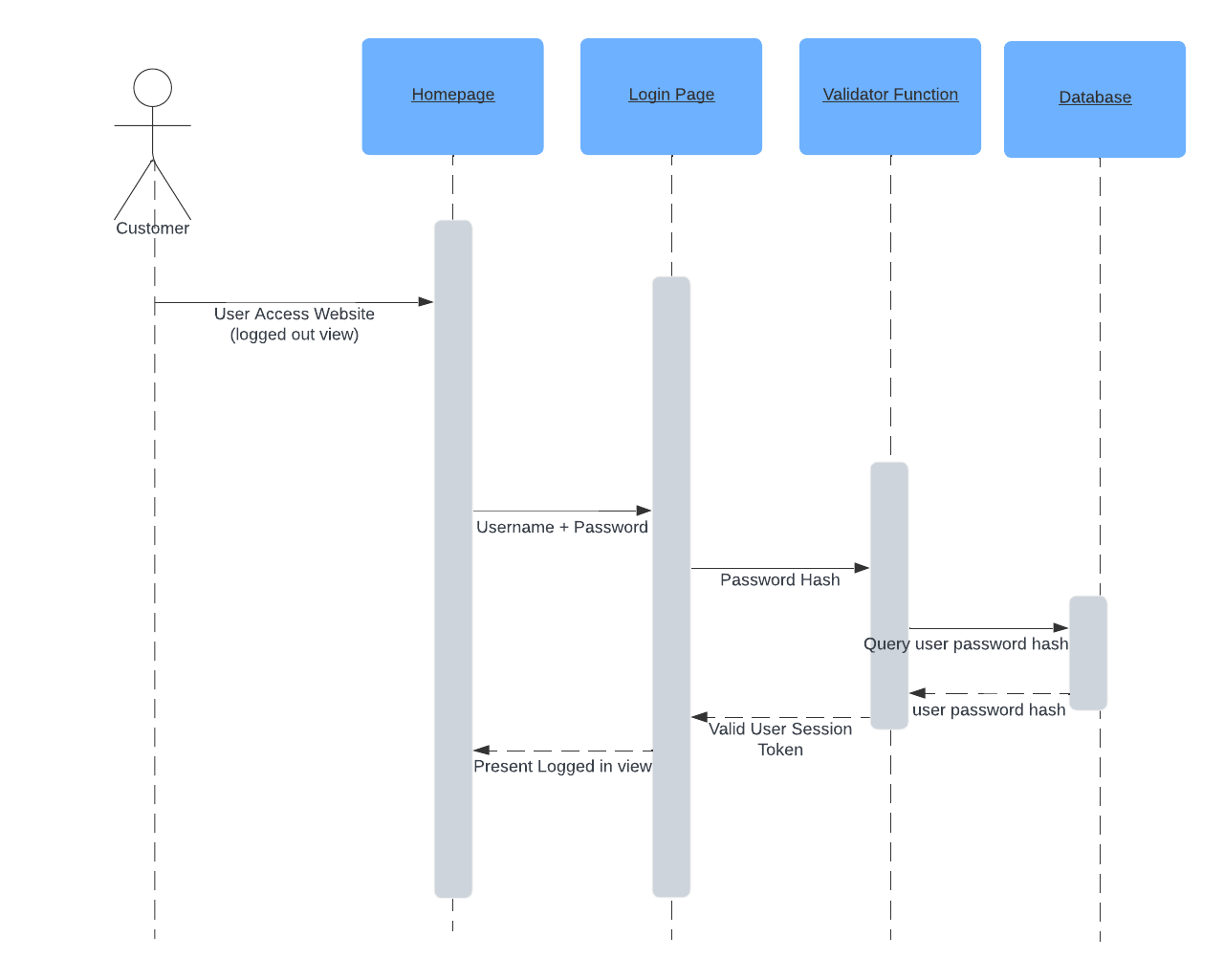
### 

**Activity Diagram: User Edit Profile**

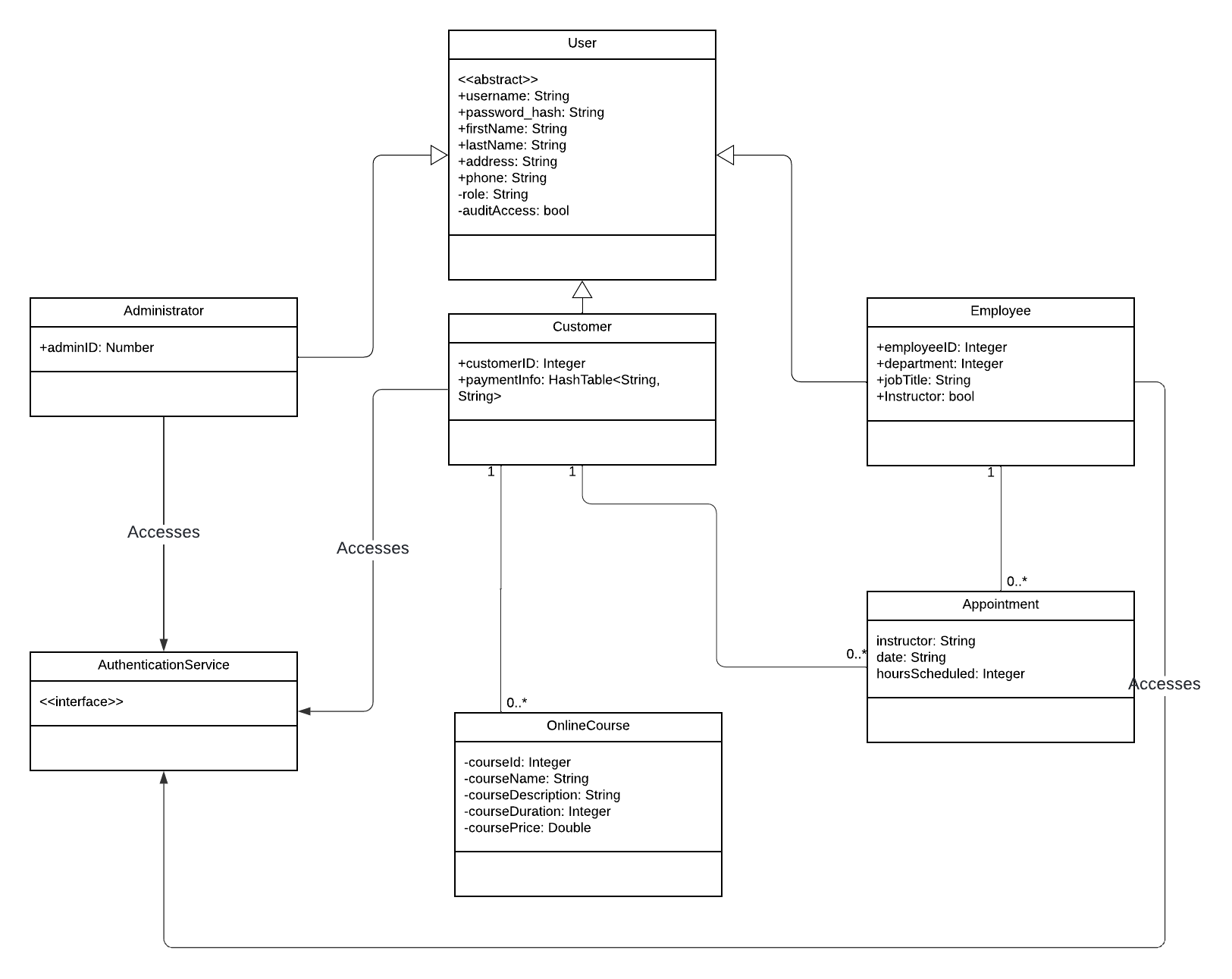


### UML Sequence Diagram

**Login Sequence**



### UML Class Diagram



## Technical Requirements

*[Based on the diagrams you have created, describe the technical requirements of your system. These requirements should address the required hardware, software, tools, and infrastructure necessary for your system design.]*

This system design will use a 3-tier web architecture (webserver frontend, application backend, database for persistence).

**Hardware and Infrastructure**

The following hardware will be used to run this system, many of the services required for networking and firewall are provided directly by the cloud providers listed with their managed virtual machine and container services and are to be considered included with them. This hardware is all virtual and can be scaled or changed at DriverPass’ convenience if necessary for scaling or other issues.

* cloud based web service running on AWS or GCP
* Linux based x86\_64 web servers (virtual machines in selected cloud)
* Managed service for application servers running as containers
* Linux based x86\_64 database servers (virtual machines in selected cloud with extra RAM)

**Software**

The following software will be used for the three tier system components:

* NGINX or other open source webserver
* Python based application server for REST API components, Flask or Django
* RDBMS(Database), PostgresQL
* External authentication service, AWS Cognito or GCP Equivalent
* Frontend JavaScript, Vue.js single page application

**Tools**

There are many tools that will be used in the creation and administration of this system including deployment tools/systems, and monitoring and security tools.

* Deployment will be handled using GitHub actions, and GitHub will be used for source control of the code
* Monitoring and alerting will use the managed service DataDog as it provides almost all possible options for monitoring and features can be chosen al-a-carte
* Security scanning will be done with SonarrQube to statically scan code, with live servers being checked by open source tools like Linux auditd, and an intrusion detection system like Suricata