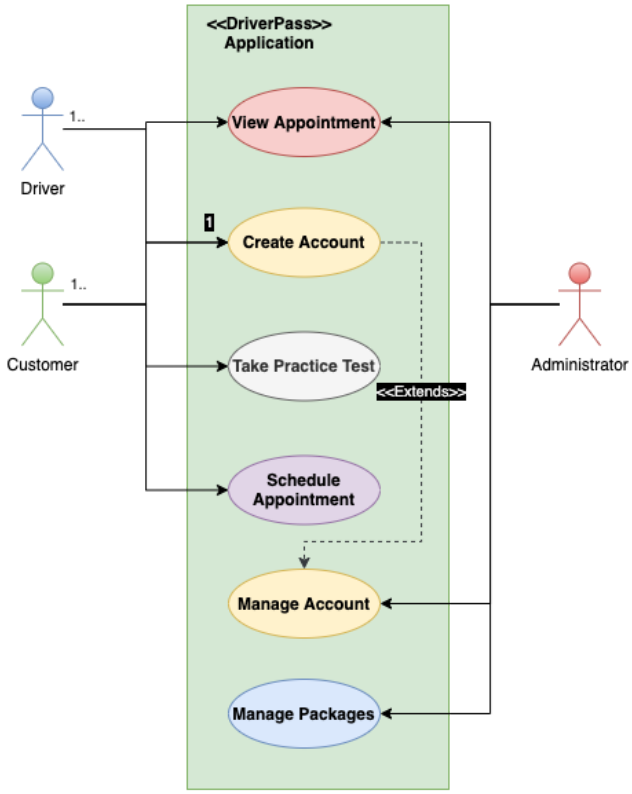
Southern New Hampshire University

CS 255 System Design Document Template

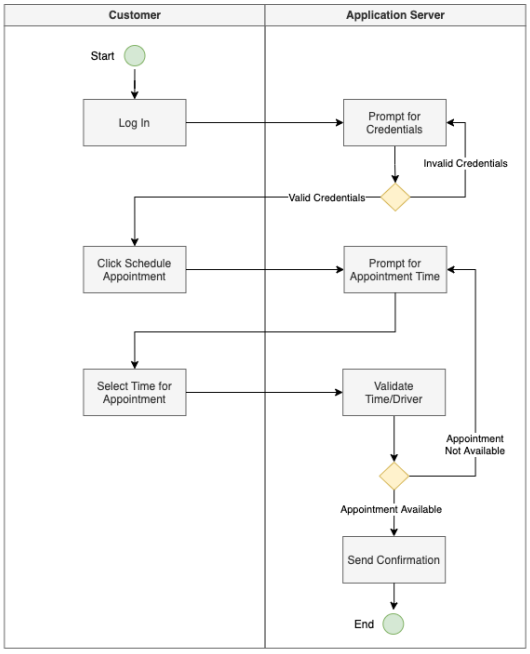
UML Diagrams

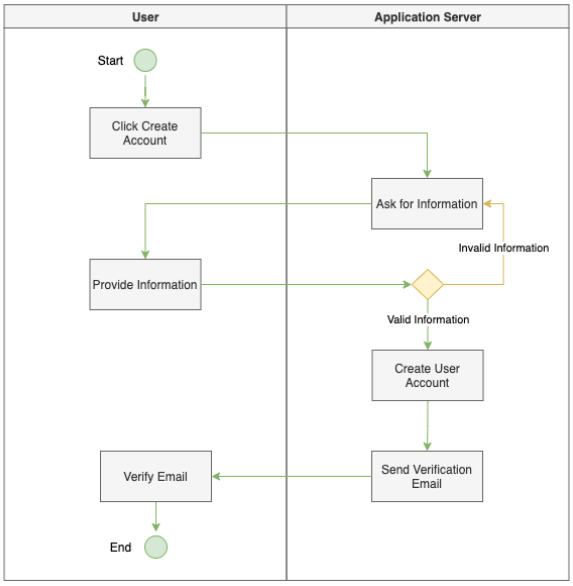
UML Use Case Diagram



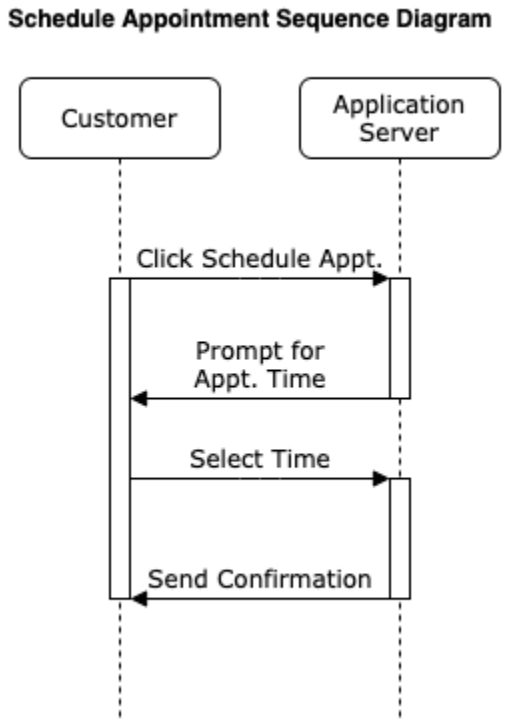
UML Activity Diagrams

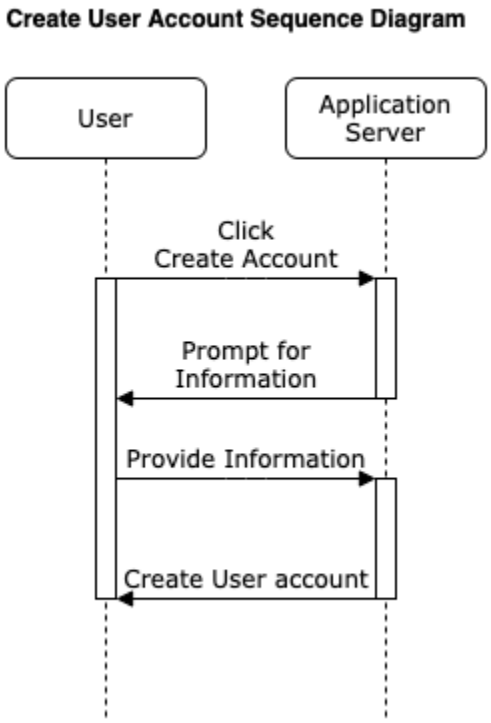
Schedule Appointment Activity Diagram



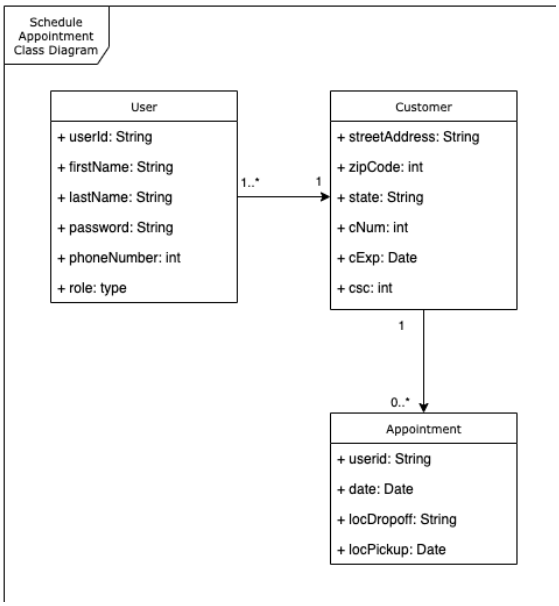


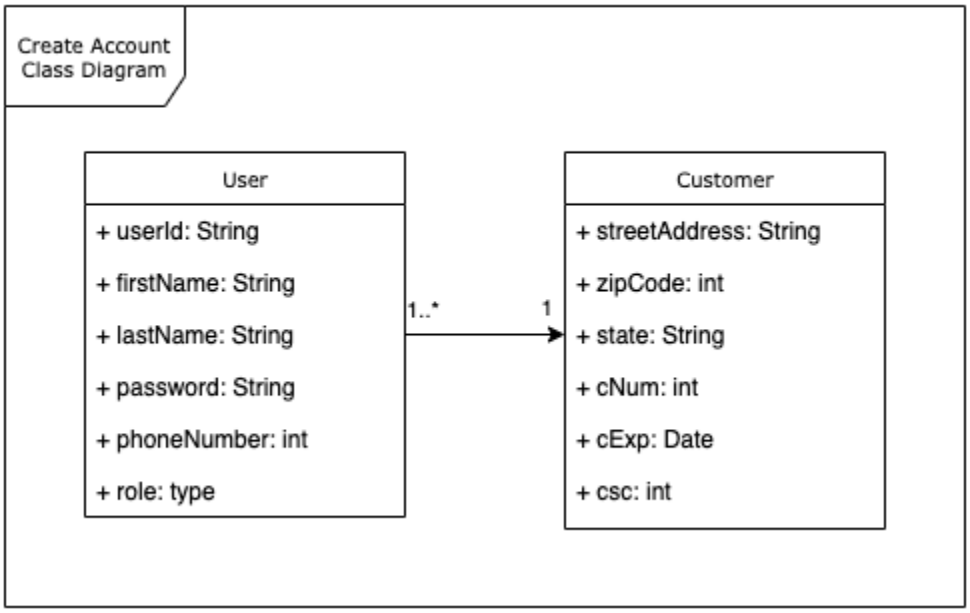
UML Sequence Diagram





UML Class Diagram





Technical Requirements

The system needs to be hosted within a cloud environment and serve browser clients via HTTP. We should aim for less than 100ms response time when clients' browsers are making requests/loading pages. With this flavor of architecture, we should be able to make updates without any outage so they could be planned around any schedule. The most cost effective method of hosting the web services would be using a lightweight flavor of linux such as Alpine; however, Ubuntu would also work. We would also require a database which could be a cloud-native service or we could provision a VM to host a Windows SQL server (this may be more expensive). Any time that we have a free form text field, we will need to sanitize the users' inputs. For instance, if the user tries to provide a userid which does not conform to our userid standards, we should throw a visible error message within their browser so they may correct it. Because we will be forming the service calls between the UI service and REST services, inputs should already be limited enough to prevent many errors. Despite this fact, we should take advantage of console logging and log alerts for our cloud platform. In this way, we could send an email to admins when a critical error occurred. Spring Boot Java would be an excellent programming language to fulfill our requirements in terms of functionality.