Rayna Best

02/23/24

Professor Bermudez

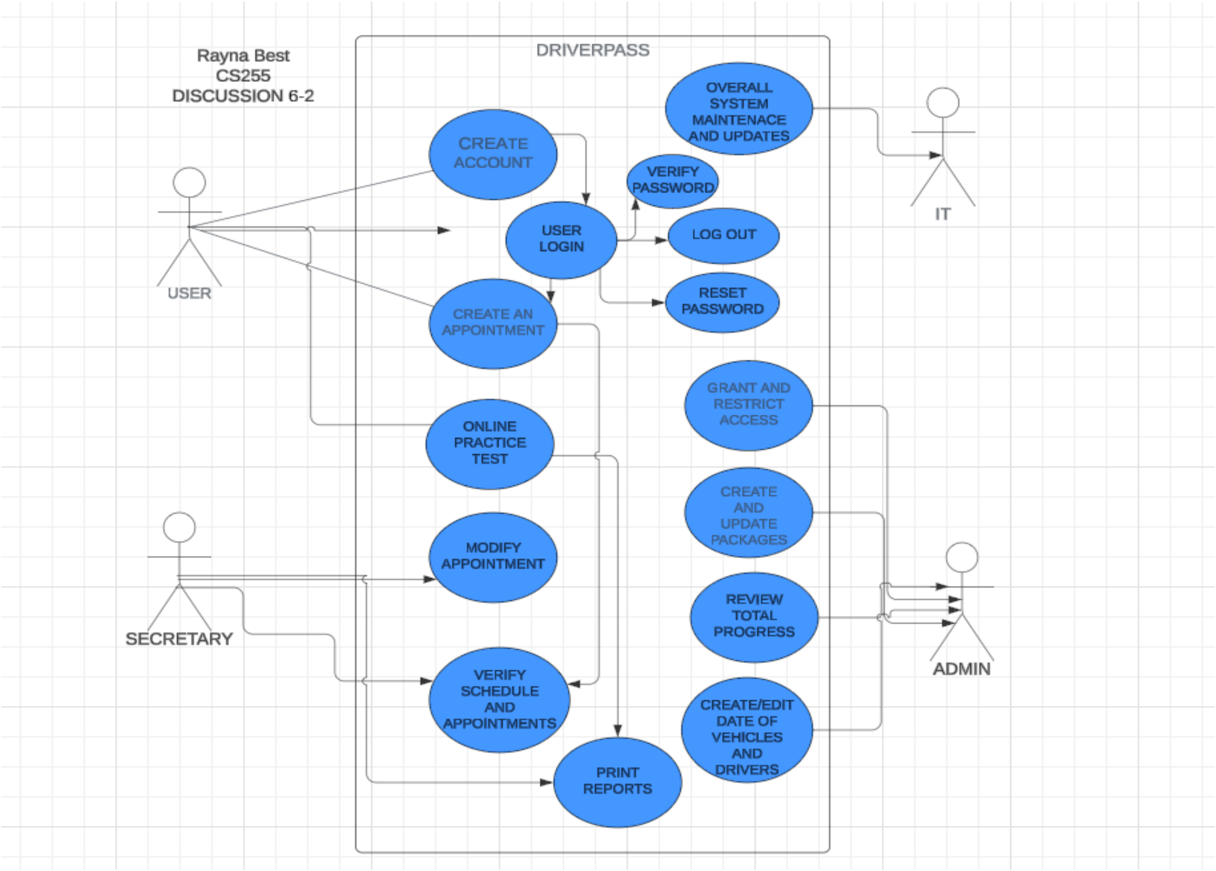
CS 255

System Analysis and Design

Project Two

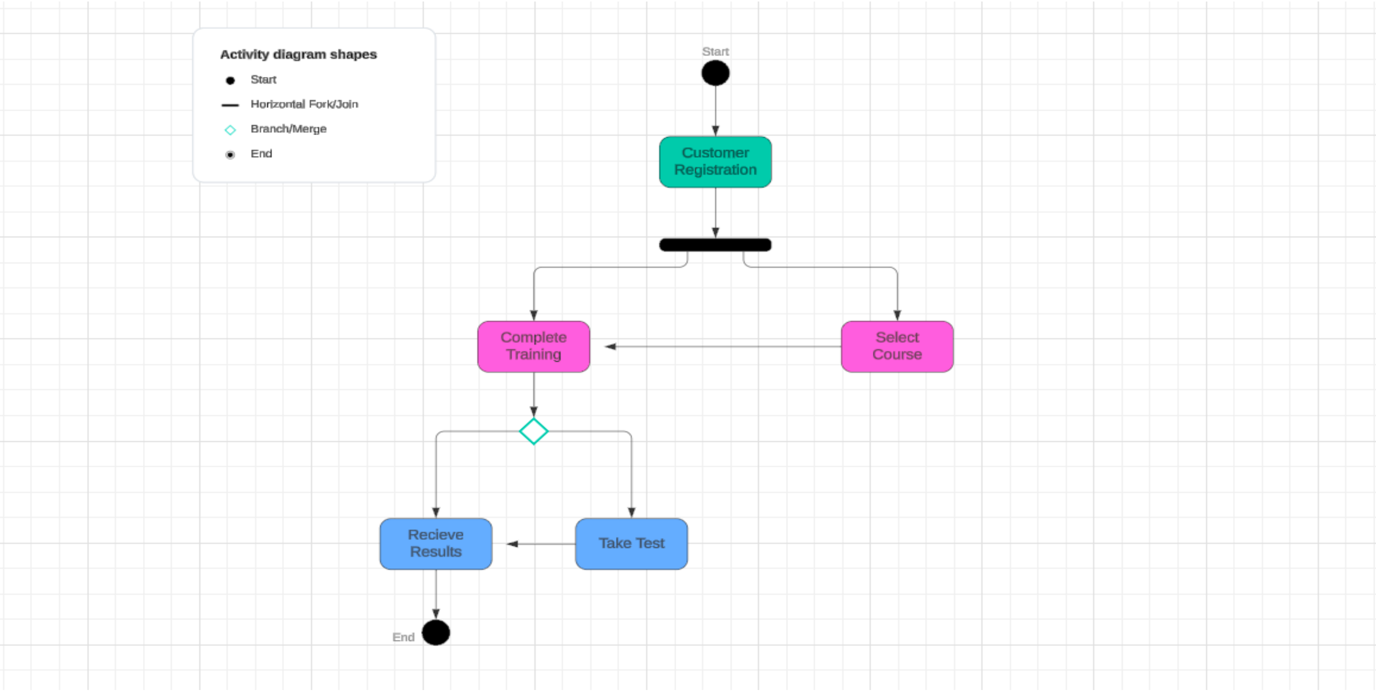
UML Diagrams

UML Use Case Diagram

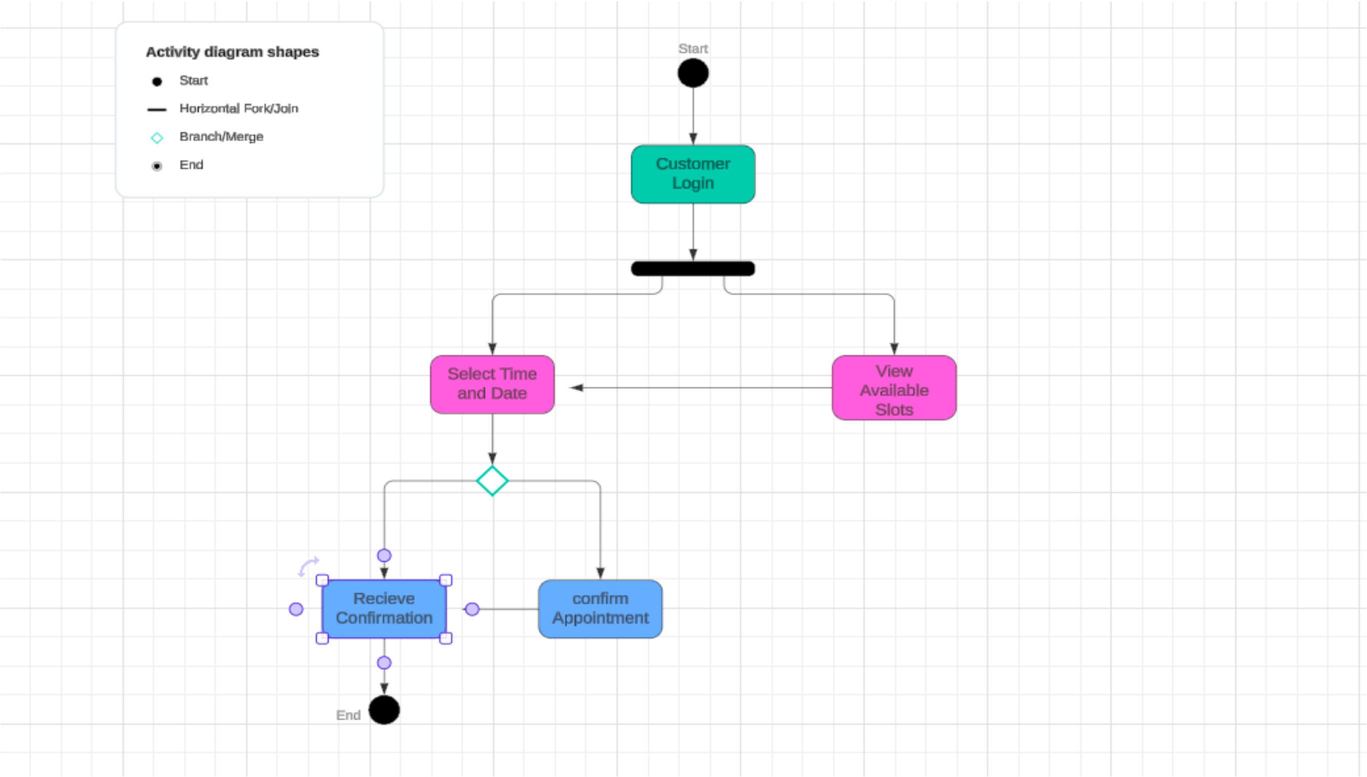


UML Activity Diagrams

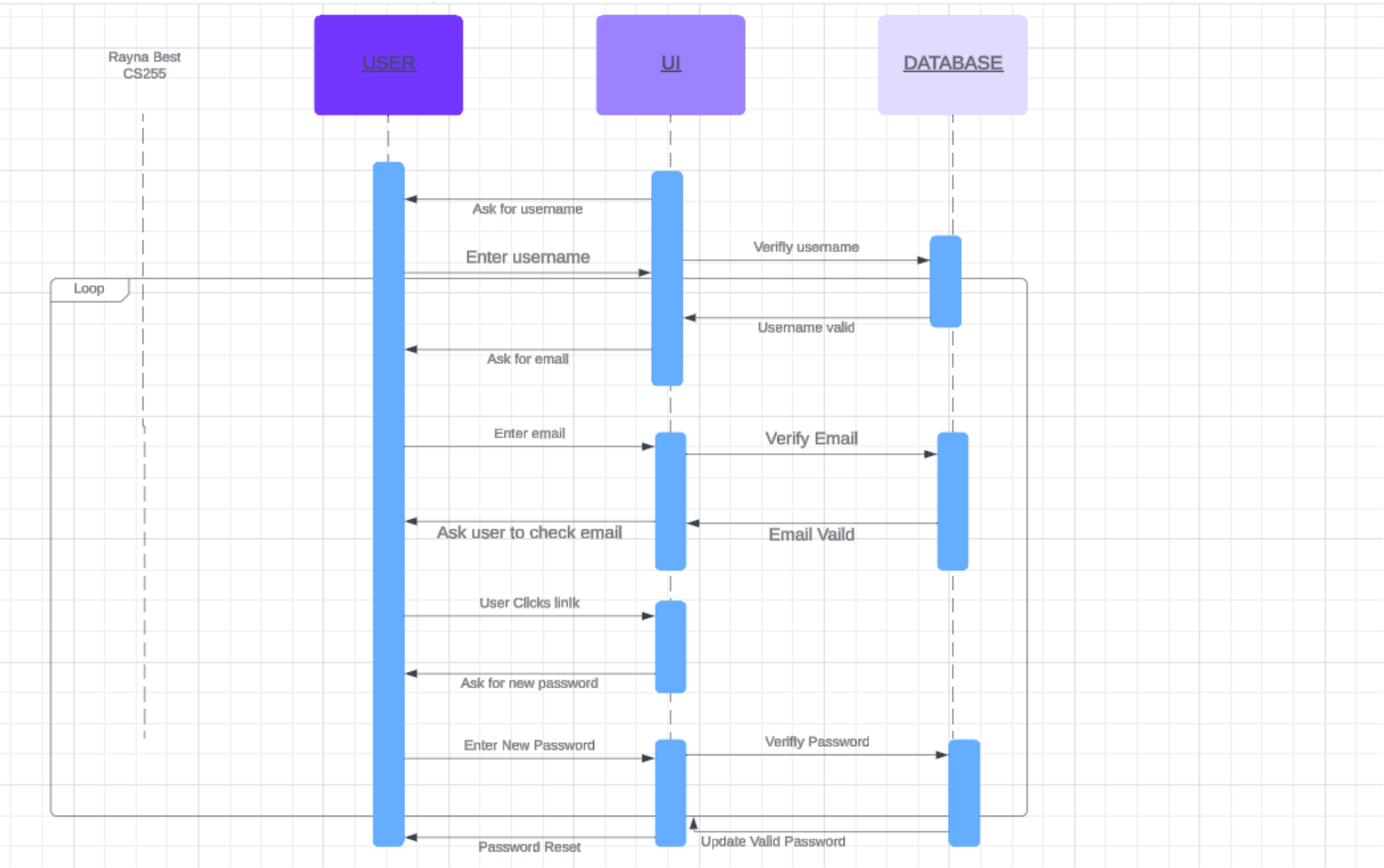
Activity Diagram for Online Training for Written Tests:



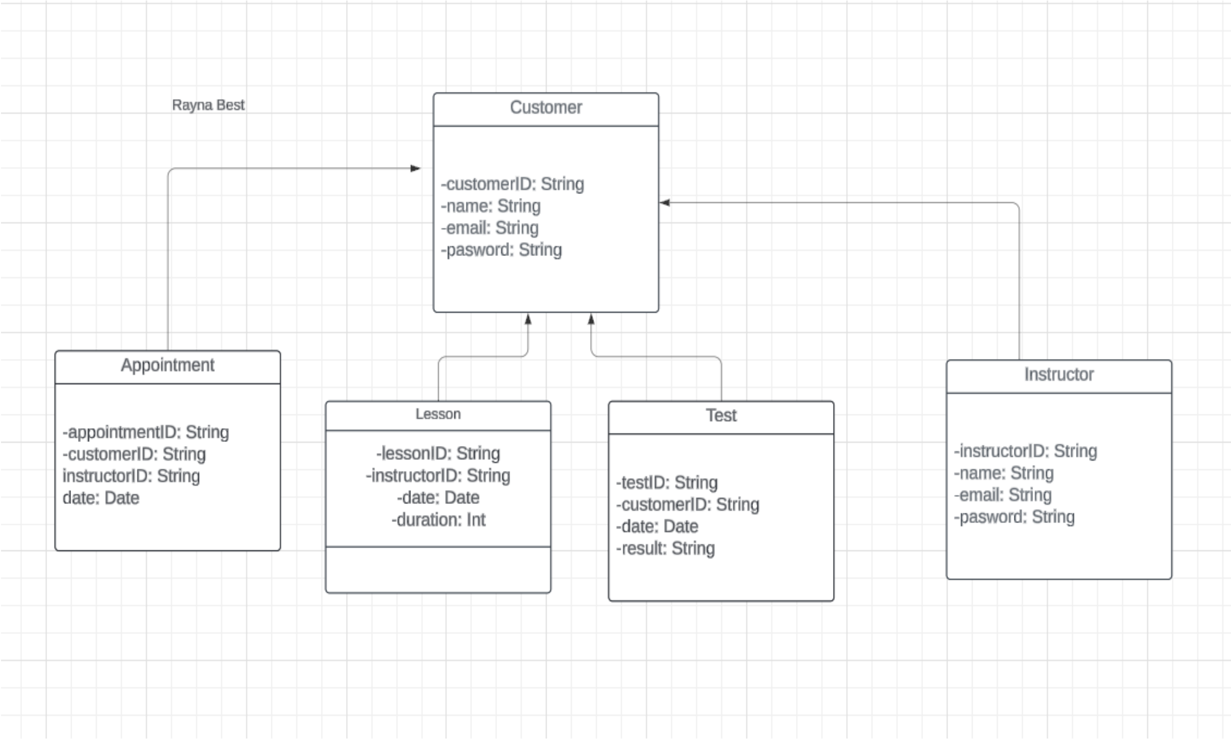
Activity Diagram for Scheduling Driving Lessons:



UML Sequence Diagram



UML Class Diagram



Technical Requirements

The successful implementation of the DriverPass system needs a robust technical infrastructure comprising various hardware, software, tools, and network components. Hardware requirements include servers capable of hosting the application, ample storage for data management, and a reliable network infrastructure to ensure seamless connectivity. On the software front, developers will require a comprehensive suite of tools, including integrated development environments (IDEs), database management systems, and web server software for application deployment. BPMN modeling tools will facilitate the creation of activity diagrams, while UML modeling tools will aid in crafting sequence and class diagrams. A secure infrastructure featuring reliable internet connectivity, power backup systems, and stringent security measures such as firewalls and encryption protocols is essential to safeguard sensitive data. A robust database system capable of securely storing customer information, appointment details, lesson schedules, and test results is imperative for system functionality. Programming languages such as HTML, CSS, JavaScript, Java, and Python will be utilized for web and backend development, while SQL will handle database management tasks. User-friendly interfaces tailored to the specific needs of customers, secretaries, and administrators, accessible via web browsers or mobile applications, will enhance user experience. Integration with third-party services for additional functionalities such as payment gateways and communication APIs will enrich the system's capabilities. Testing frameworks for unit testing, integration testing, and user acceptance testing will ensure system reliability and performance. The system architecture should be designed with scalability in mind, capable of accommodating increasing user loads and data volumes as the system expands.

Regular maintenance procedures, updates, and troubleshooting protocols will be implemented to uphold system integrity and efficiency over time.