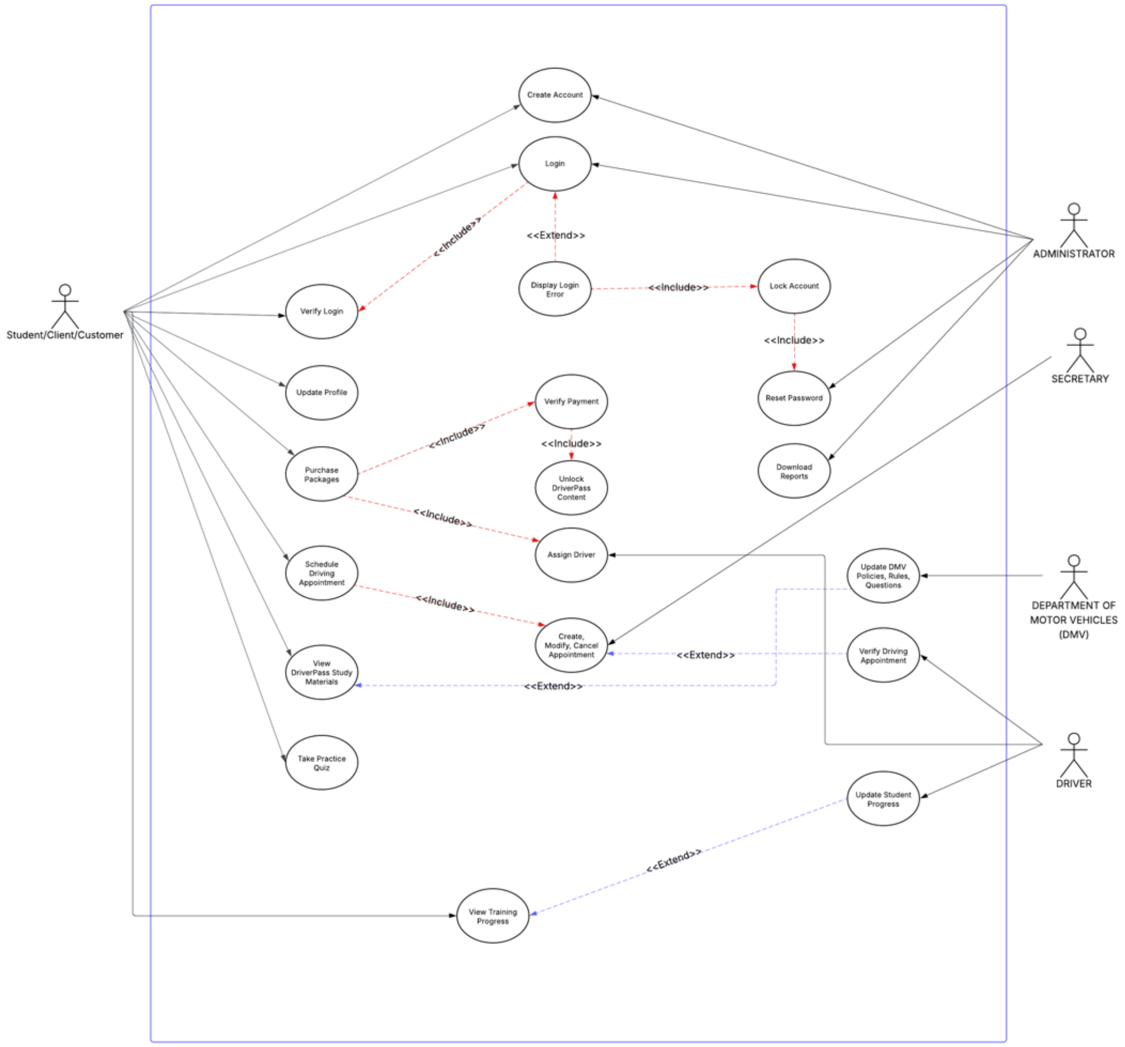
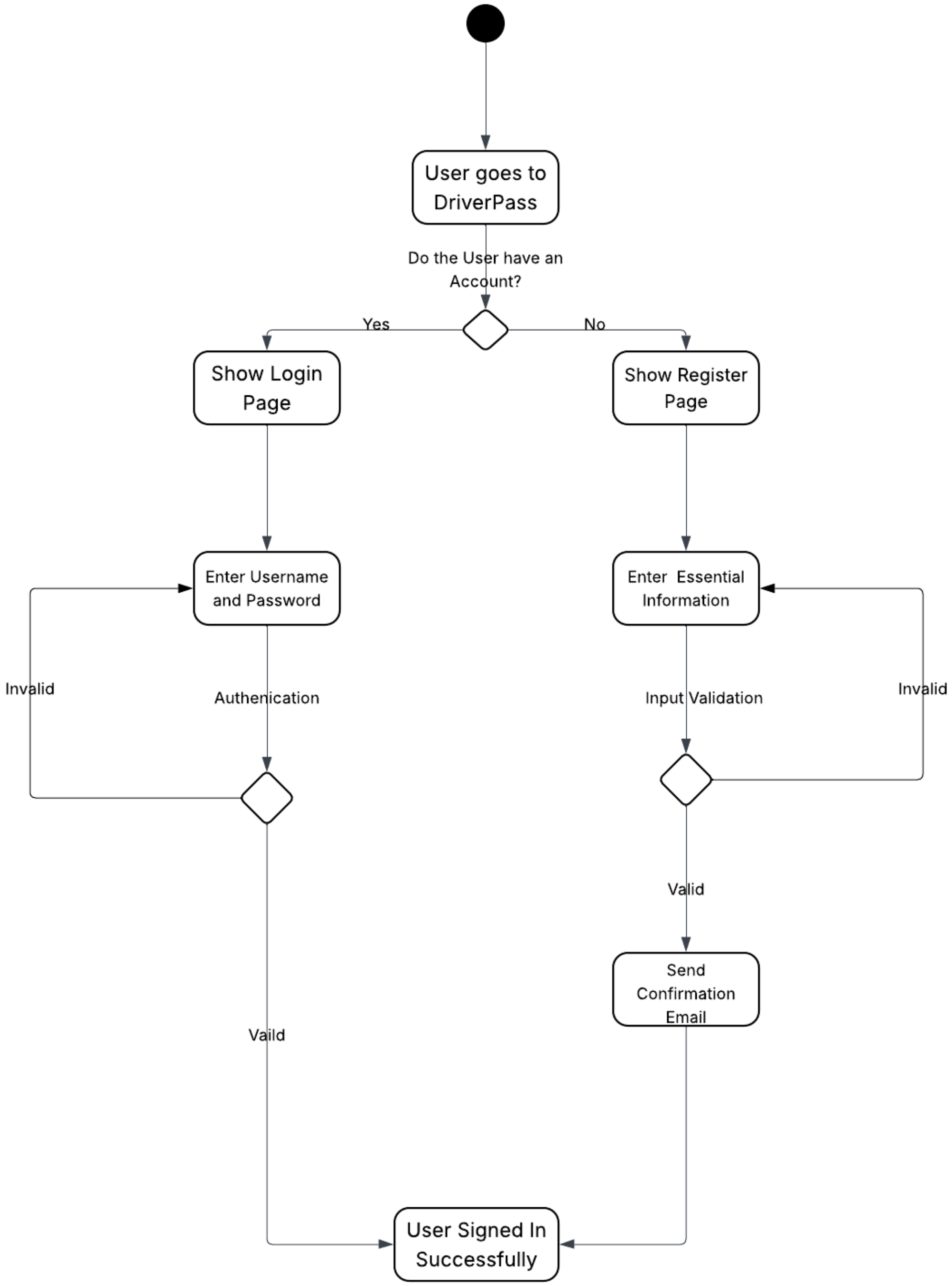
CS 255 System Design Document Template

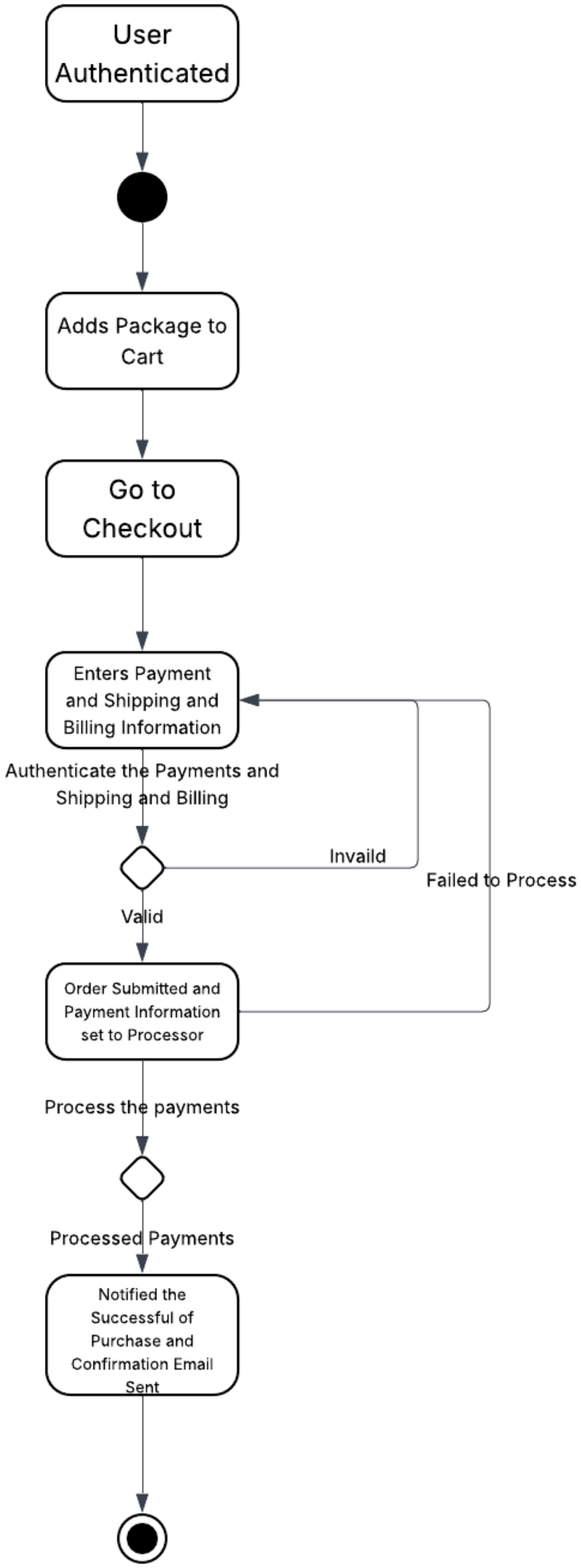
UML Diagrams

UML Use Case Diagram

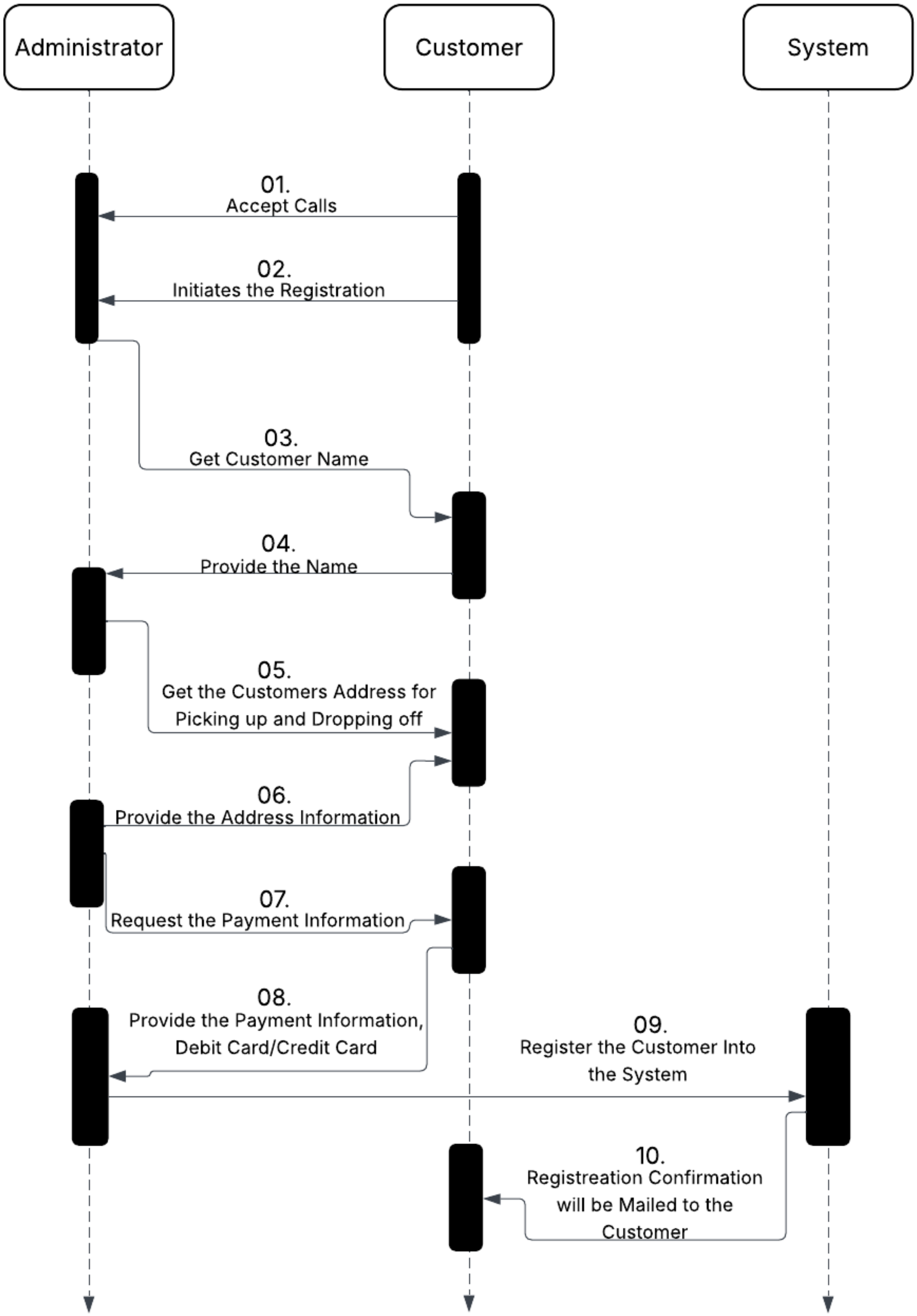


UML Activity Diagrams

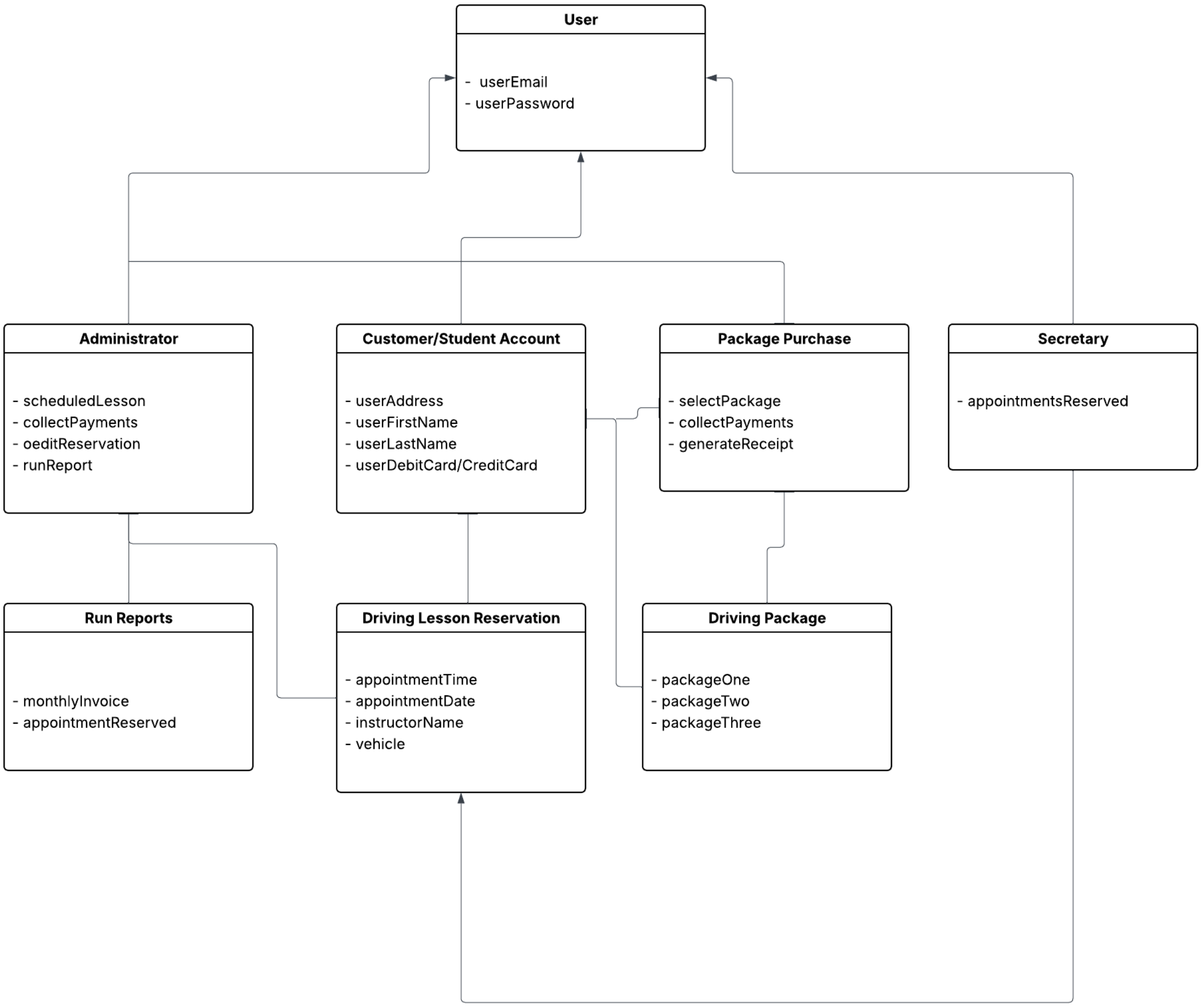

User goes to DriverPass
Do the User have an Account?
Yes
No
Show Login Page
Show Register Page
Enter Username and Password
Enter Essential Information
Invalid
Authenication
Input Validation
Invalid
Valid
Send Confirmation Email
Vaild
User Signed In Successfully


User Authenticated
Adds Package to Cart
Go to Checkout
Enters Payment and Shipping and Billing Information
Authenticate the Payments and Shipping and Billing
Invaild
Failed to Process
Valid
Order Submitted and Payment Information set to Processor
Process the payments
Processed Payments
Notified the Successful of Purchase and Confirmation Email Sent

UML Sequence Diagram


Administrator
Customer
System
01.
Accept Calls
02.
Initiates the Registration
03.
Get Customer Name
04.
Provide the Name
05.
Get the Customers Address for Picking up and Dropping off
06.
Provide the Address Information
07.
Request the Payment Information
08.
Provide the Payment Information, Debit Card/Credit Card
09.
Register the Customer Into the System
10.
Registreation Confirmation will be Mailed to the Customer

UML Class Diagram



Technical Requirements

To make DriverPass work, I spent a lot of time thinking about the behind-the-scenes requirements. Due to the fact that users are likely to use a variety of devices, I wanted to make sure the system is responsive and mobile-friendly. It would be best to have a screen resolution of at least 1280x720 in order to see everything clearly. It would be best to choose a server with at least four CPU cores, eight GB of RAM, and 100 GB of storage. And of course, regular data backups are a must in case anything crashes.

For the software, I'd recommend using a Linux server, probably Ubuntu, and something like Apache or Nginx to handle the web traffic. I'm thinking Python with Flask or Django, or even Node.js, just something clean and scalable. Because the system will be used across Windows, Mac OS, iOS, and Android, I'd want to build it using a modern framework like React or Vue.js so it looks good and runs well. It's also important to keep security in mind. If I were you, I'd use HTTPS, SSL certificates, and secure passwords. As well as making sure users have only what they need, I'd like to separate students, instructors, and admins with role-based access.

To create all the diagrams for the project, I used Lucidchart, which helped a lot with visualizing things. If this were a real project, I would use tools like Trello or Jira to manage the workflow and stay organized. DriverPass includes lessons and practice tests, so it would be smart to connect it with a learning management system like Moodle or follow SCORM standards to link it to an LMS. If users need to pay for lessons or tests, I would add a payment processor like Stripe or PayPal.

I would host the system on a flexible and reliable platform like AWS, Google Cloud, or Azure to handle more users if needed. With RESTful APIs, I'd connect the front-end and back-end of the application. Additionally, if possible, I would like to be able to track my tests automatically by integrating the DMV APIs with the project. Whenever I want to monitor and check performance, I will use tools such as New Relic or ELK for this purpose.