SOFE 3650U - Software Design and Architectures
Final Project: Airline Reservation System Case Study
Group 31

NAME	STUDENT ID
Charles Olagunju	100749818
Lorenzo Caguimbal	100555976
Osasogie Osuki	100748837
Roniel Casaclang	100755336

Iteration 1: Establishing an Overall System Structure

- Step 1: Review inputs

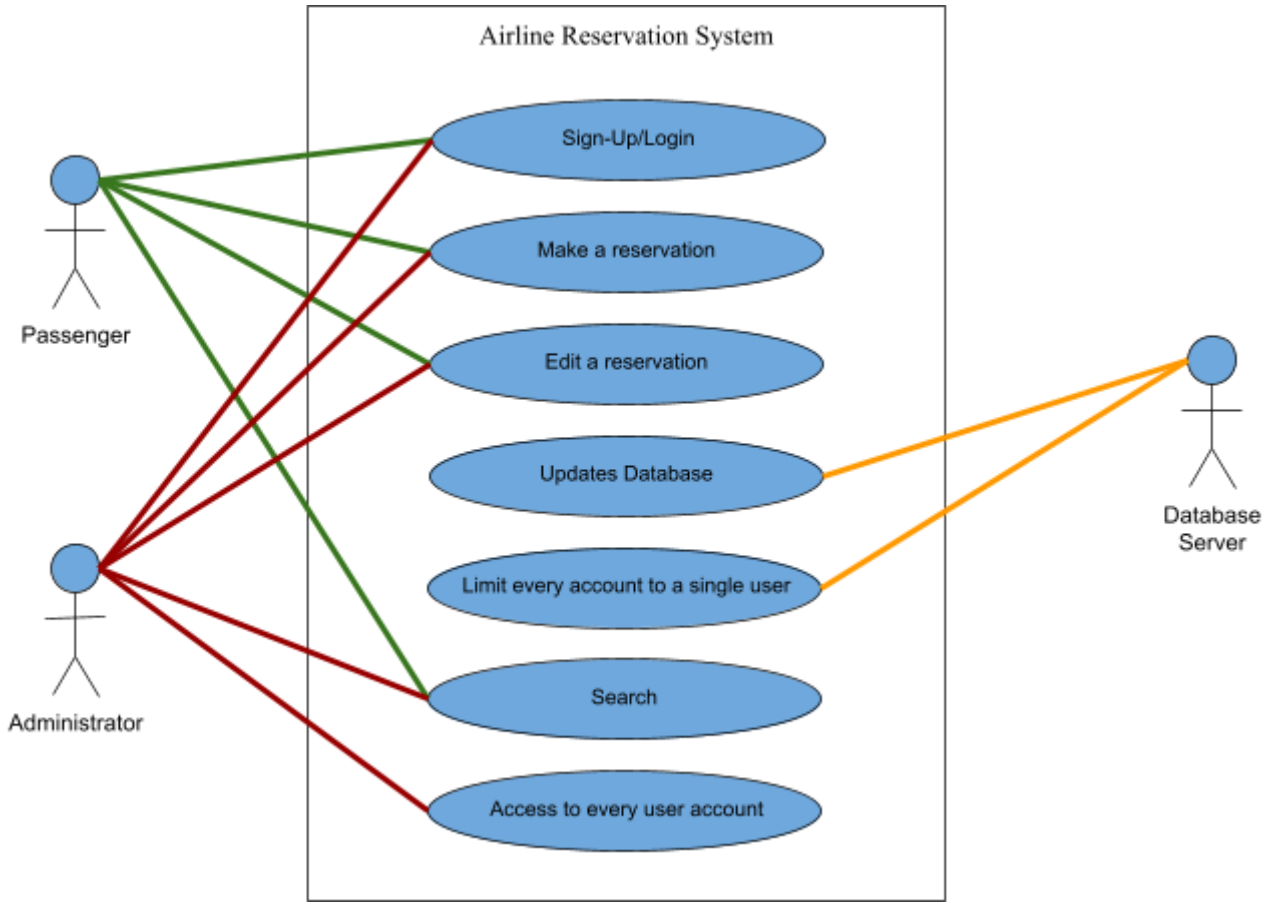

Figure 2: Use Case Diagram

Use Case:

Use Case	Description
UC-1: Sign-Up / Login	A user or administrator registers/logs into the system through the sign-up/login screen.
UC-2: Make a reservation	A user makes a flight reservation, selecting the times, dates, passenger seats, travel class, destination and type of flight ticket.
UC-3: Edit a reservation	A user edits a reservation under their account. An admin edits a user's reservation.
UC-4: Updates database	Airline server updates the database for every user activity and every information that is entered.
UC-5: Limit every account to a single user	Airline server limits the user from registering the same email.
UC-6: Search	A user or admin uses the search bar with a filter to search for destinations or airlines.
UC-7: Access to every user account	An admin opens a user's account.

Quality Attributes:

ID	Quality Attribute	Scenario	Associated Use Case
QA-1	Accessibility	The website is user-friendly and accessible. The UI allows any user to navigate the website easily and quickly make reservations.	UC-1, UC-3, UC-4, UC-6 & UC-7
QA-2	Security	User logs into the system and makes changes to their account. The system has all changes documented, timed and also displays to the user.	UC-5
QA-3	Compatibility	User decides to use an Android device to make flight reservations and it is still functional as it is compatible with the device.	ALL
QA-4	Availability	There is downtime in the regular operation of the system. The system is backed up and running in less than a minute.	ALL
QA-5	Performance	All services are 100% functional at all times.	ALL

System Constraints:

ID	Constraint
CON-1	Accommodate for multiple users must be able to book a reservation / use the system simultaneously, automatically storing each sessions in the database
CON-2	Compatibility with desktop operating system (Windows, MacOS, Linux)
CON-3	Future support for mobile operating system (iOS, Android)
CON-4	System must respond quickly to request of information from database / servers. (transitions between forms, filtering of available flights, etc.)
CON-5	Network connections between user and servers should be consistent and reliable
CON-6	System must always be up to date with flight details (available flights, # of available seats, time of arrival/departure, etc.)

Architectural Concerns:

ID	Concern
CRN-1	Establishing an overall system structure.
CRN-2	Leverage the team's knowledge about JavaScript Frameworks and Databases.
CRN-3	Allocate work to members of the development team.

- **Step 2: Establish Iteration Goals by Selecting Drivers**

  This iteration will mainly focus on the architectural concern about the compatibility of the system between each platform. A suitable reference architecture is needed to satisfy the system requirements. The drivers for this iteration will be:
  - UC-4: Updates database
  - CON-2: Compatibility with desktop operating system (Windows, MacOS, Linux)
  - CON-3: Future support for mobile operating system (iOS, Android)

  Elements to decompose:

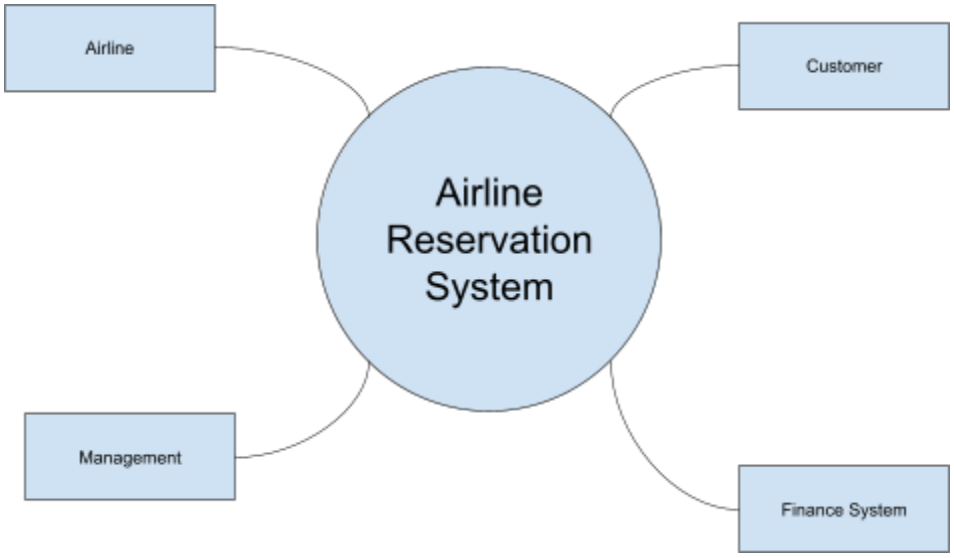

Figure 2: Context Diagram

- **Step 3: Choose One or More Elements of the System to Refine**

  The element to refine is the entire airline system (see Figure 2). The refinement is performed through decomposition.

- **Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers**

  Choosing design concepts:
  - Reference architectures
  - Patterns (Design patterns, deployment patterns)
  - Tactics
  - Externally developed components

Design Decisions and Location	**Rationale**
Logically structure the client/server part of the system using the **Rich Internet Application** reference architecture	The web applications reference architecture communicates with a server using the HTTP protocol. This architecture is oriented toward the development of applications that are accessed from a web browser which the Airline Reservation System is intended to be. It provides clients / customers to easily access the Airline Reservation to book available flights.
Physically structure the application using the **three-tier deployment pattern**	A three-tier deployment pattern is most common when using web applications.

Table 1: Design Decisions

- **Step 5: Instantiate Architectural Elements, Allocate Responsibilities**

Design Decisions and Locations	**Rationale**
Add Data Sources	Need to store data locally for later use.
Presentation	Responsible for managing user interaction
Rich UI Engine	Responsible for rendering user interface elements inside the plug-in execution container
Business processing	Responsible for managing business logic on the client side
Service interfaces	Responsible for exposing services that are consumed by the components that run in the browser
Message types	Responsible for managing the types of messages that are exchanged between the client part and the server part of the application.

- **Step 6: Sketch Views and Review Design Decisions**

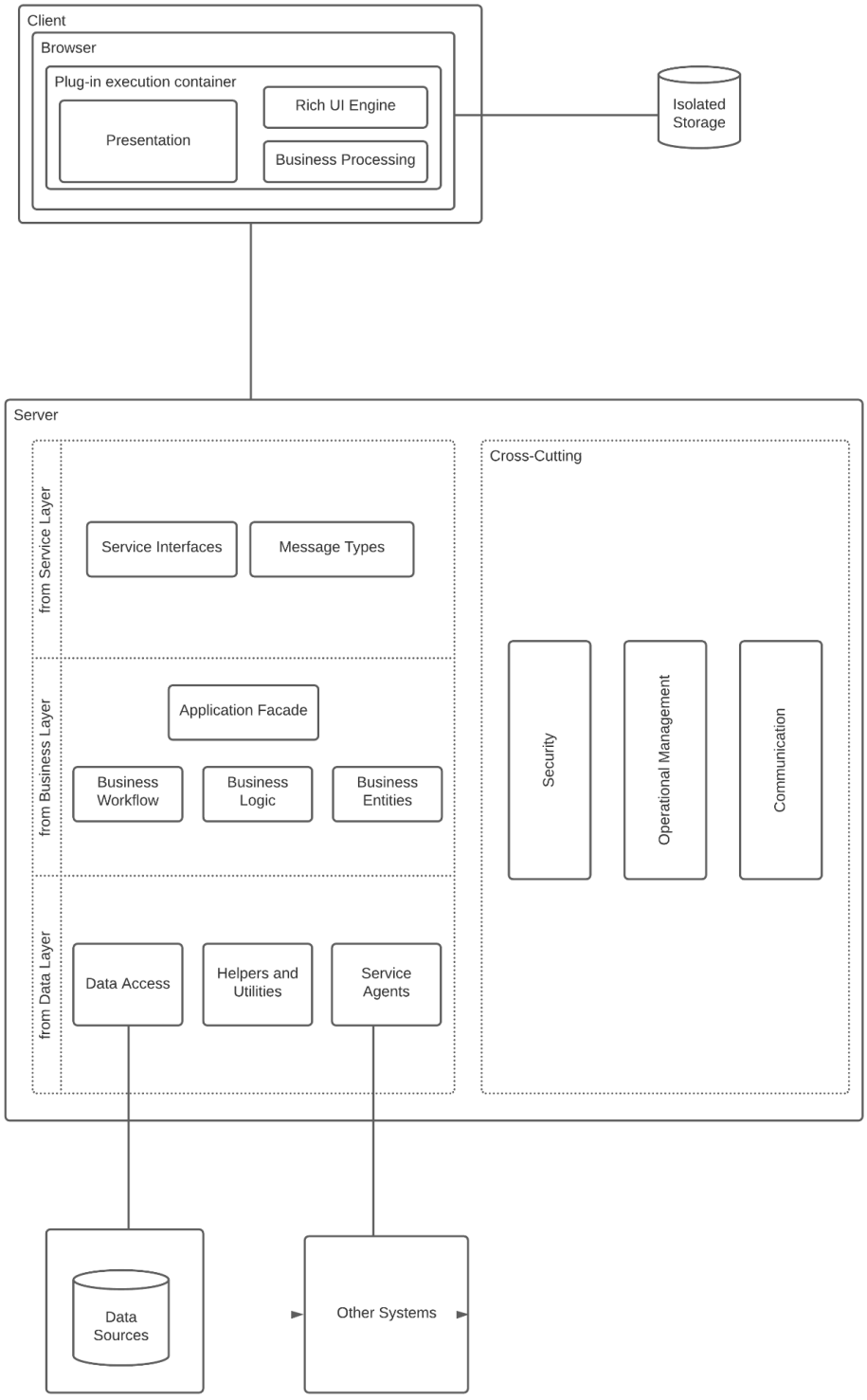

Figure 1: Rich Internet Application layer

- **Step 7: Perform Analysis of Current Design and Review Iteration**

<table>
<tr><th>Not Addressed</th><th>Partially Addressed</th><th>Completely Addressed</th><th>Design Decisions Made During the Iteration</th></tr>
<tr><td></td><td>UC-4</td><td></td><td>The selected reference architecture (Rich Internet Application) established the modules that will support this functionality.</td></tr>
<tr><td></td><td>QA-3</td><td></td><td>The selected reference architecture will allow access and be compatible with Windows, MacOS, Linux, iOS, Android, etc. providing they have access to supporting web browsers.</td></tr>
<tr><td></td><td></td><td>CON-2</td><td rowspan="2">Uses Rich Internet Application. This architecture is a web application designed compatible with desktop applications. It runs in a web browser and does not require software installation on the client side meaning it will have no trouble running on mobile devices (iOS & Android).</td></tr>
<tr><td></td><td></td><td>CON-3</td></tr>
<tr><td></td><td>CRN-2</td><td></td><td rowspan="2">Technologies considered take into account the knowledge of the developers and how comfortable they are with said technologies.</td></tr>
<tr><td></td><td>CRN-3</td><td></td></tr>
</table>

Iteration 2: Identifying Structures to Support Primary Functionality

- **Step 2: Establish Iteration Goal by Selecting Drivers**

  This iteration will mainly focus on the overall structure of our system. This being how the website will look and function. The drivers for this iteration will be:
  - UC-1: Sign-Up / Login
  - UC-2: Make a reservation
  - UC-3: Edit a reservation
  - UC-6: Search
  - CRN-1: Establishing an overall system structure.

- **Step 3: Choose One or More Elements of the System to Refine**

  The elements that will be refined in this iteration would be the use cases but mainly, UC-1, UC-2 and UC-6. Refining these elements will help support the functionality of the system and also address CRN-1.

- **Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers**

Design Decisions and Location	Rationale and Assumptions
Build the user interface of the application using HTML, CSS and the React JavaScript framework.	The react framework for building Web Applications ensures usability and functionality. The framework gives the website more structure, a rich user experience and leverages the team's knowledge of the React framework.

- **Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces**

Design Decisions and Location	Rationale
Map the system use cases to domain objects	An initial identification of domain objects can be made by analyzing the system's use cases. Mapping the system will help and address CRN-1, which was to have an overall structure of our system.

- **Step 6: Sketch Views and Record Design Decisions**

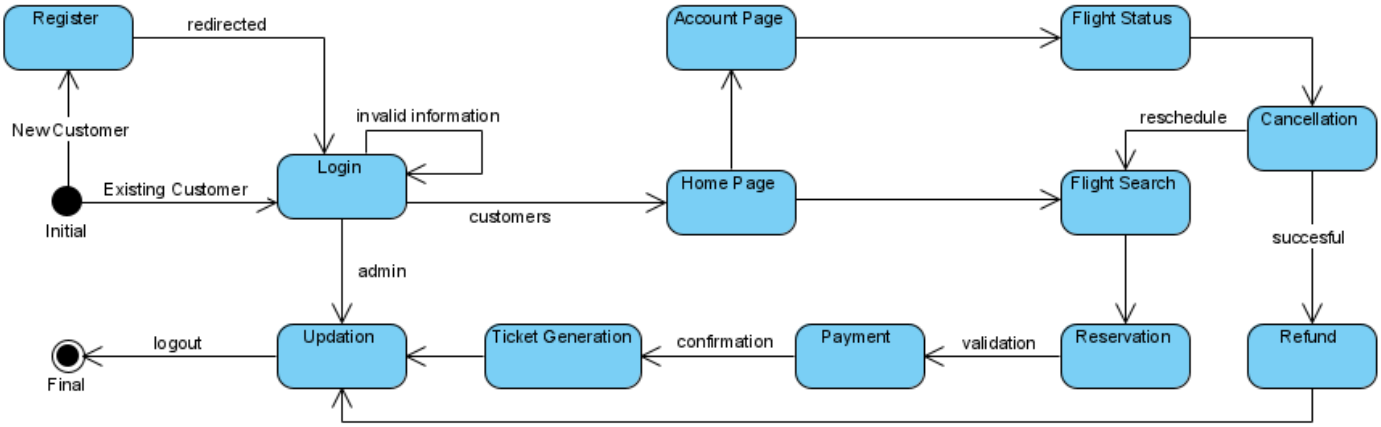

Figure 2: Activity Diagram

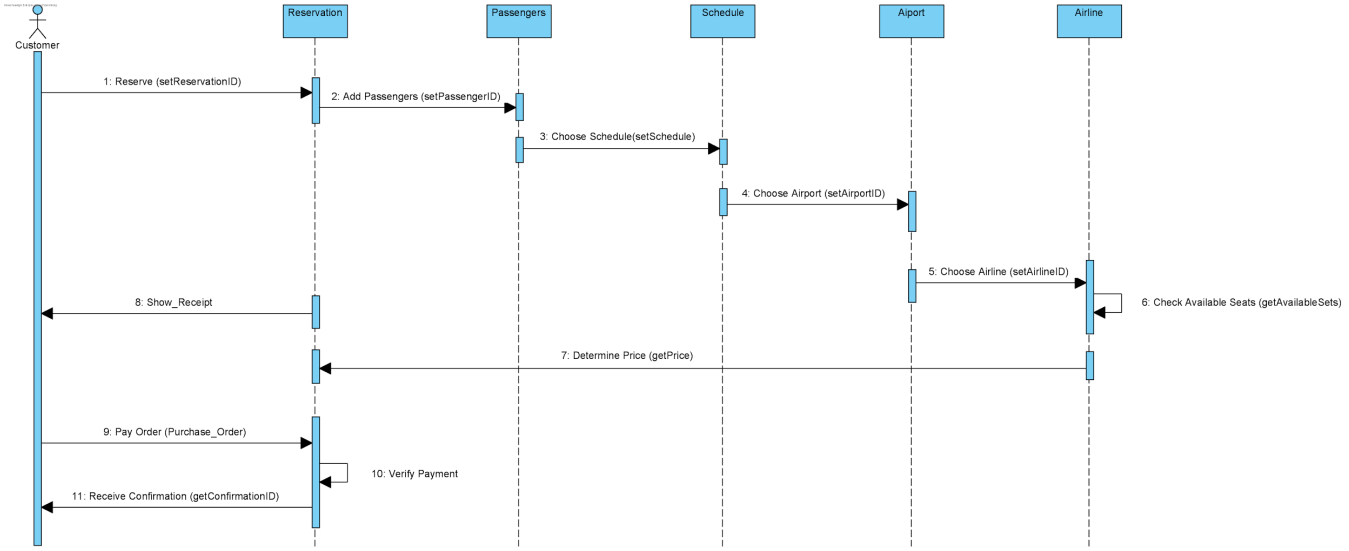

Figure 3: Sequence diagram for use case UC-2

Method Name	**Description**
Element: Customer	
1: Reserve (setReservationID)	Initialize reservation with the specified customer
8: Show_Receipt	Shows all flight details (total price, destination, etc.)
9: Pay Order (Pruchase_Order)	Prompts user for payment method to be filled out

11: Receive Confirmation (getConfirmationID)	Finalize flight order with confirmation
Element: Reservation	
2: Add Passengers (setPassangerID)	Add customer details to the following steps
10: Verify Payment	Verify payment from step (9)
Element: Passengers	
3: Choose Schedule(setSchedule)	Allows passenger to select flight schedule (time of arrival/departure)
Element: Schedule	
4: Choose Airport(setAirportID)	Lets passenger choose departure location (airport)
Element: Airport	
5: Choose Airline(setAirlineID)	Lets passenger choose which airline to travel in
Element: Airline	
6: Check_Available Seats(getAvailableSets)	Check the available seats in the selected airline
7: Determine Price(getPrice)	Determine the total of the specified flight

- **Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose**

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		UC-1	Modules across the layers and preliminary interfaces to support this use case have been identified. Developers have an idea of the structure of the system due to the activity and sequence diagram which will aid in the construction of the website.
		UC-2	
		UC-3	
		UC-6	
		QA-1	The elements that are associated with the use case (UC-2, 3, 6) have been identified.
		QA-2	The elements that are associated with the use case (UC-1) have been identified.

		CRN-1	Modules associated with all of the use cases have been identified. Structure of the system has been defined from the diagrams (activity & sequence).
	CRN-2		Technologies considered take into account the knowledge of the developers and how comfortable they are with said technologies.

Iteration 3: Addressing Quality Attribute Scenario Driver

- **Step 2: Establish Iteration Goal by Selecting Drivers**

  For this iteration the main focus will be on QA-4: Availability and QA-5: Performance. The selected drivers are:
  - UC-7: An admin open's a user's account
  - UC-5: Limit every account to a single user
  - UC-4: Updates Database

- **Step 3: Choose One or More Elements of the System to Refine**

  For this step, the elements to be refined are UC-7, UC-5 and UC-4 from iteration 1. This will address QA-4: Availability and QA-5: Performance.

- **Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers**

Design Decision and Location	Rationale and Assumption
Using PHP and MySQL for the Database.	Refining the Database, ensures that every user has a unique ID (including the admin) and records user activities such as flight bookings under their account. The unique ID will differentiate certain users (such as the admin) from other users, who will have more user actions available to them (such as being able to access every client's account). This addresses QA-4: Availability, in which having a database ensures that the system is backed up and is running in less than a minute. Also addresses QA-5.

- **Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces**

Design Decisions and Location	Rationale
Using JavaScript and leveraging the React	Leveraging the React framework to

framework for the reservation and registration system.	restrict users from making certain types of user actions such as creating multiple accounts with the same email address. This addresses QA-5: Performance, ensuring that all services such as the buttons, user registration and reservations forms, are 100% functional at all times and works as intended.

- **Step 6: Sketch Views and Record Design Decisions**

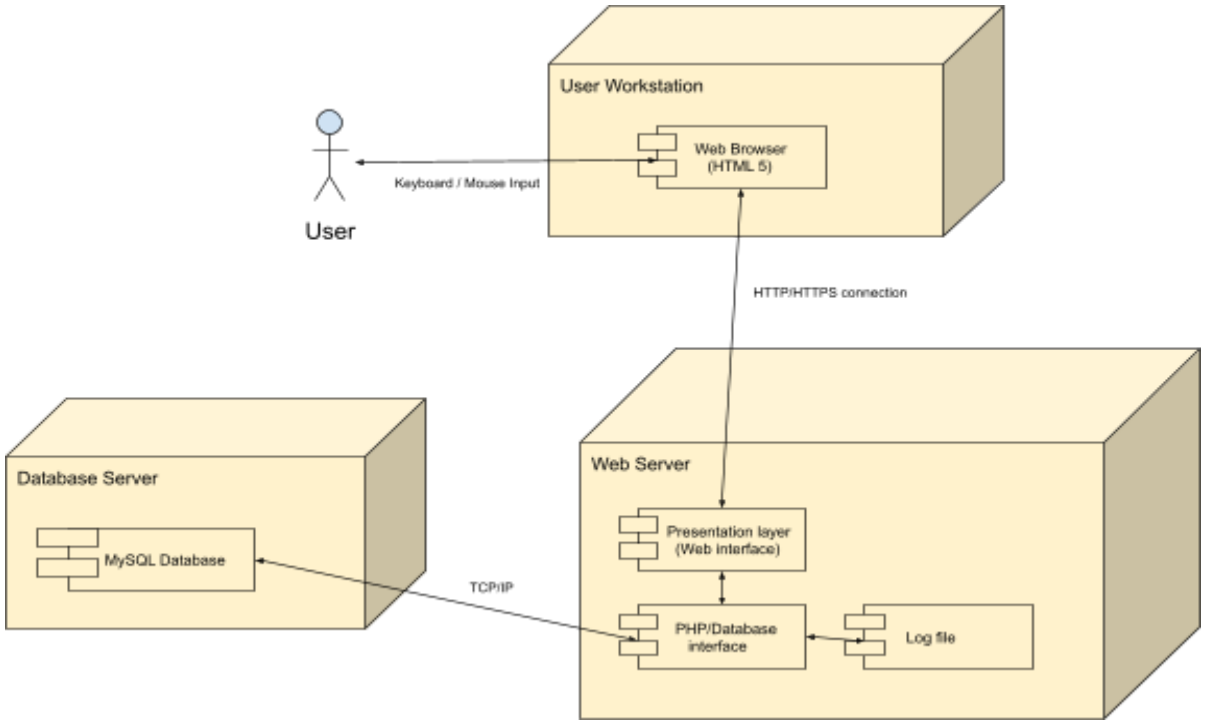

Figure 4: UML Deployment Diagram

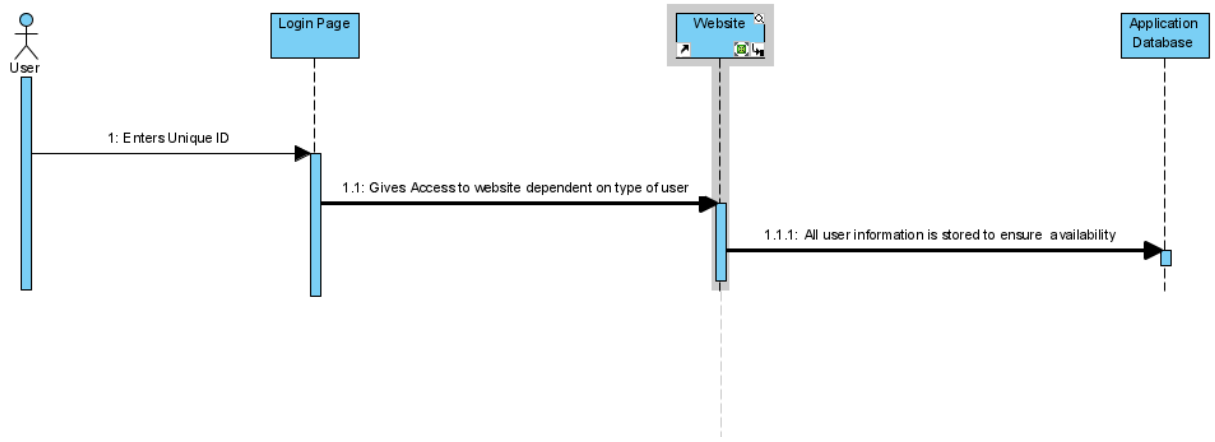

Figure 5: Sequence Diagram Illustrating Login Interaction with Database to Determine the Type of User

- **Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose**

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During the Iteration
		UC-4	The use of PHP and MySQL supports these use cases to be able to access information that are stored in the database. This will also allow updates to database when information is modified/added/deleted.
		UC-5	
		UC-7	
	QA-4		Database ensures that all important user and system information is backed up and run in less than a minute if any error occurs.
	QA-5		Creating a script that will allow only certain actions to be available based on whether a user is an admin or a customer ensure everything will work as intended.
		CON-4	The design decision of incorporating PHP and MySQL for the database addresses these concerns (CON-4, 5, 6) of requesting information from the database and having a stable connection between the user and the server.
		CON-5	
		CON-6	