COE125 – Software Engineering

Group Members:  
Balilo, Mayrien Joy

de Pano, Claudette Jean R.

Gadia, Daniel Christian

Airline Ticket Reservation

1. Introduction

In today’s airline industry users or customers can reserve seat or book flight as long as they are connected to internet. The innovation of technology has made travelling in the air easier for customers with airline reservation system. Thousands of people flock the airline industry these days so that they can arrive their destination within a short period of time. Through this airline reservation system, customers can reserve seat, book flight and set the date if their departure and arrival using the internet. Customers can choose the place they want to go here in the Philippines. This system contributed to fundamental changes in the structure of the industry. The places available in this system are USA, Philippines, Japan, Indonesia, Argentina, Poland, Belgium, Chile, and Saudi Arabia. In the database, the flight schedule on different routes and aircraft seating capacity is stored, the username of the user, the city or country for the destination, and the payment along with their deposits and their accounts. The system ensures the security of the customers. The airline reservation system take pride in monitoring, administration and maintenance of the database environment using MySQLite Browser.

1. Objectives

The objectives of this system are:

1. To help the customers in reserving air tickets through online system
2. To implement the airline reservation system to the best satisfaction of the customer
3. Use database MySQLite to facilitate this process

The normal process of this system is to fill the data and then process the data which used to cause a lot of inconvenience to both the administrator and customers. In using databases, the reservation of airline tickets will be facilitated. The airline reservation system can be presented in views of the different involved with it. These are the administrator and customer.

1. Functions
2. Main Page

* The user can login using the correct username and password
* If the user has no account yet, choose the signup button to create an account

1. Signup Page

* The user must fill up the fields create an account. These fields are the first name, last name, email address, username, password, address, date of birth, gender, and contact number.
* Cancel button is created to exit the program without saving it

1. Administrator Login

* The default username and password are fixed

1. Managing Flights

* The admin will be able to manage the flights by first choosing the date on the calendar widget.
* After choosing the date, the flights that have been stored in the database will show on the drop-down menu and the admin may delete the flight chosen.
* The details that would show up on the system are the location, departure, time of arrival, price, and available seats.
* The admin can put remarks and updates it in the system.

1. Managing Passengers

* In managing the passengers, first is to choose the desired date to the see the available flights that would show on the drop-down menu.
* The first name, last name, gender, and age will be displayed after choosing the passenger on the Flights drop-down list.
* The admin can delete it.

1. Managing Users

* In managing the users, the admin chooses the username in the system and it will tell the details of that user.
* The admin may delete the user

1. Customer Login

* When the user already signed up, the assigned username and password are entered in the login page

1. Booking a Flight

* The customer will choose either round trip or one-way.
* The origin place must be fill up with a selected country and city. Same with the destination.
* Schedule the date of flight and the return flight.
* Count the number of passengers (adult, children, infant, senior citizen).
* The flight details like the location, departure, arrival, price, and available seats will be displayed on the Flights section.

1. Flight Status

-The customer would be allow to view his/her scheduled flights and  
check the remarks if something comes up with the flight.

1. Payment Method

* The customer would enter his/her credit card details
* The payment details would be available for viewing at the top  
  portion.

1. Logout

* Allows the user or admin to logout the account.

1. Database (MySQLite Browser)

* The information or data is stored in SQLite Browser. Every details is seen in the database.

1. Architectural Design

The architecture of Airline Ticket Reservation is based on the three-tier architecture. This mainly consists of three layers namely:

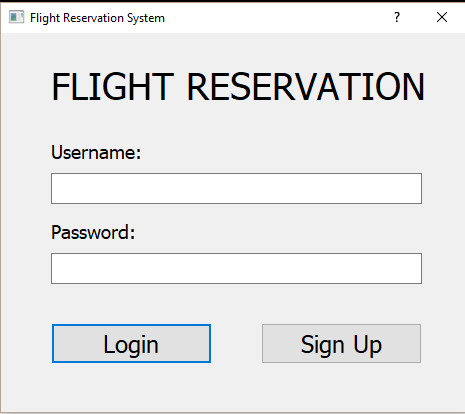
* Presentation Tier
* Business Tier
* Data Access Tier

A picture containing object

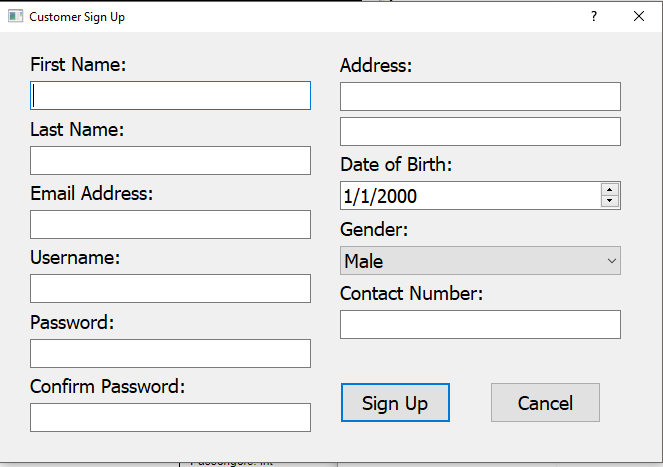
Description generated with very high confidence

The presentation layer is the topmost layer of the application and it is responsible for the user interface of the application which deals with the presentation of data to the user. The business logic layer is the middle layer of the application and the business logic for the Airline Ticket Reservation would be present here. It is responsible for the information exchange between the user interface and the database. The database layer is the final layer of the application. All the data related to the system is stored and retrieved from here.

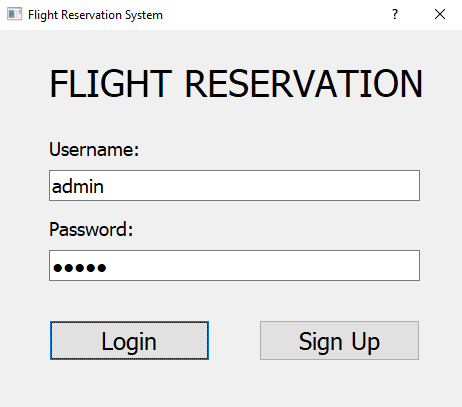
1. Graphical User Interface
2. Main Page



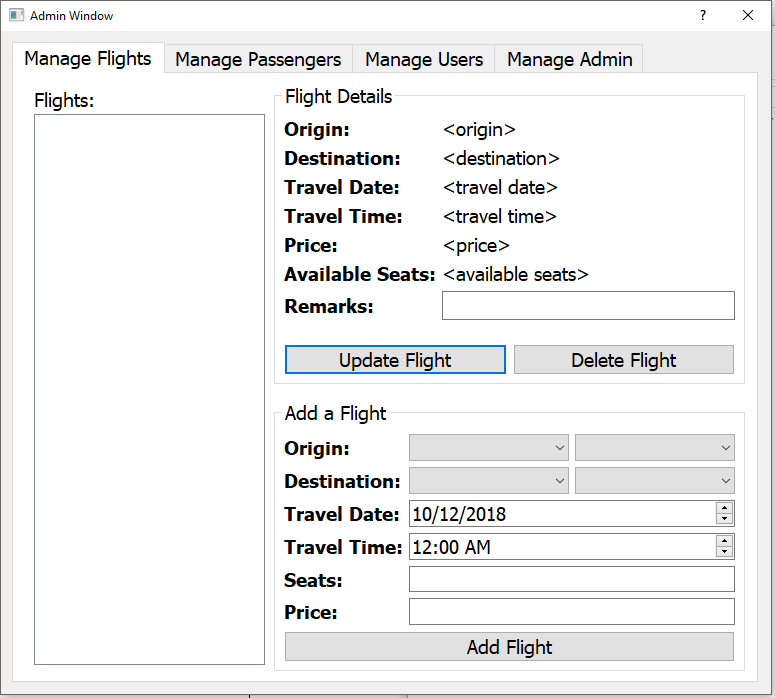
1. Sign Up Page



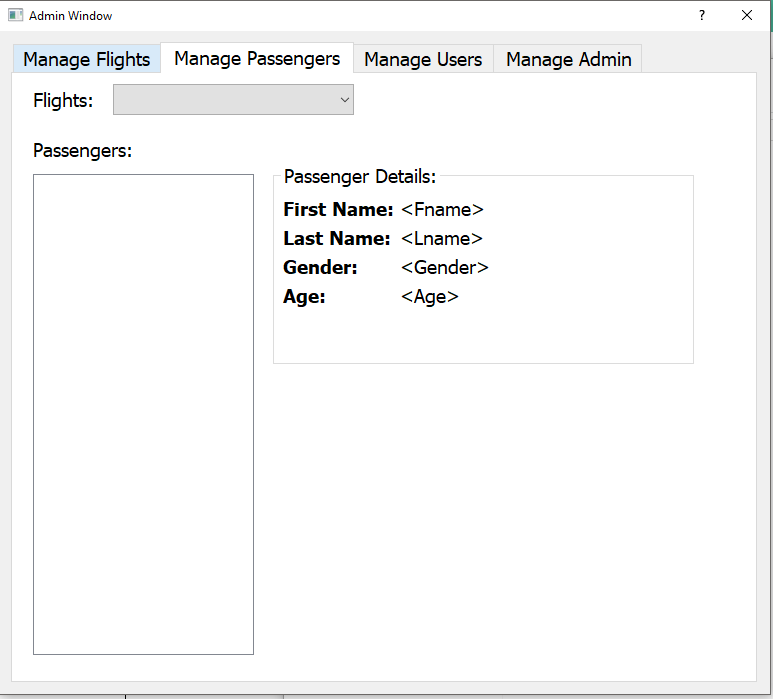
1. Administrator Login



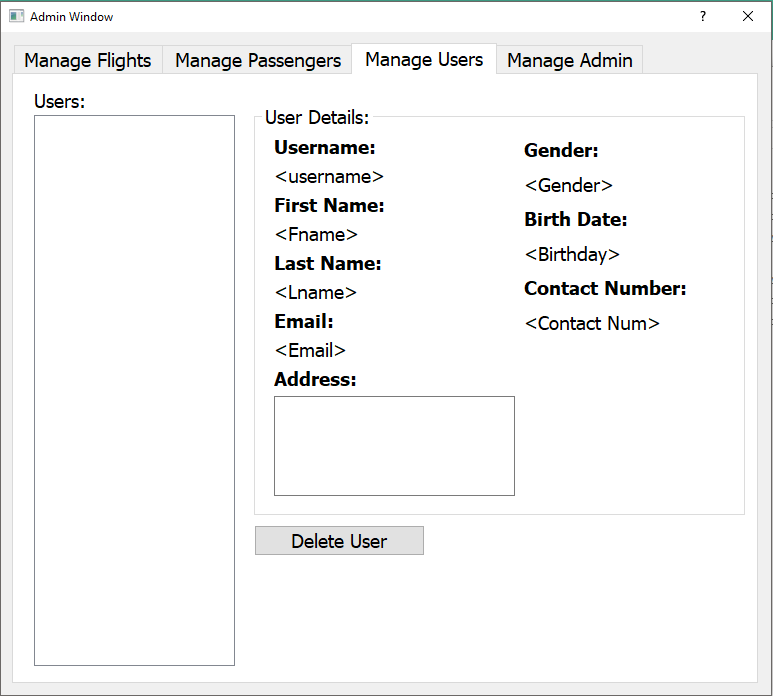
1. Managing Flights

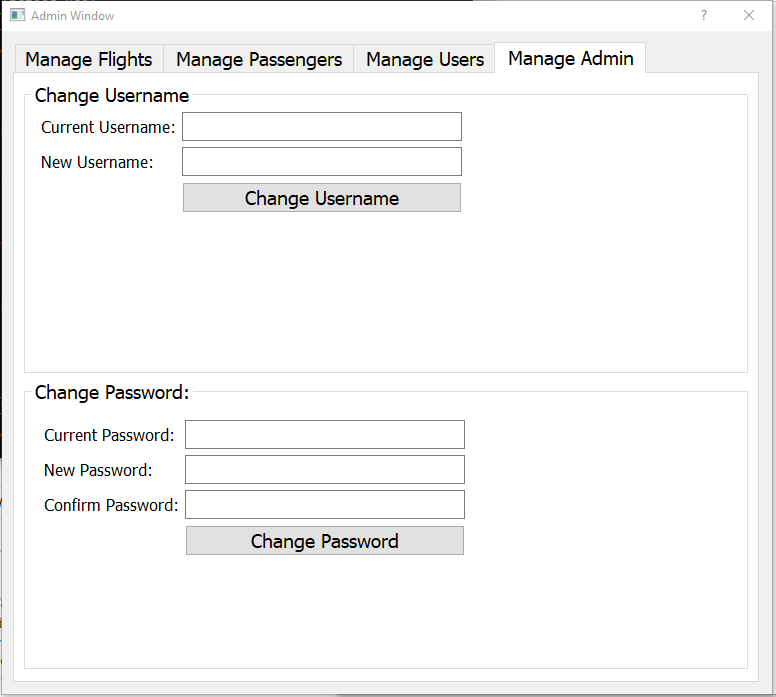
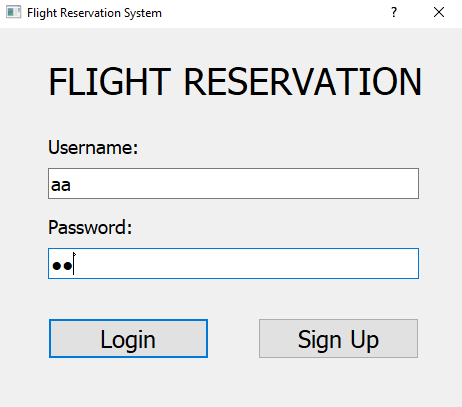
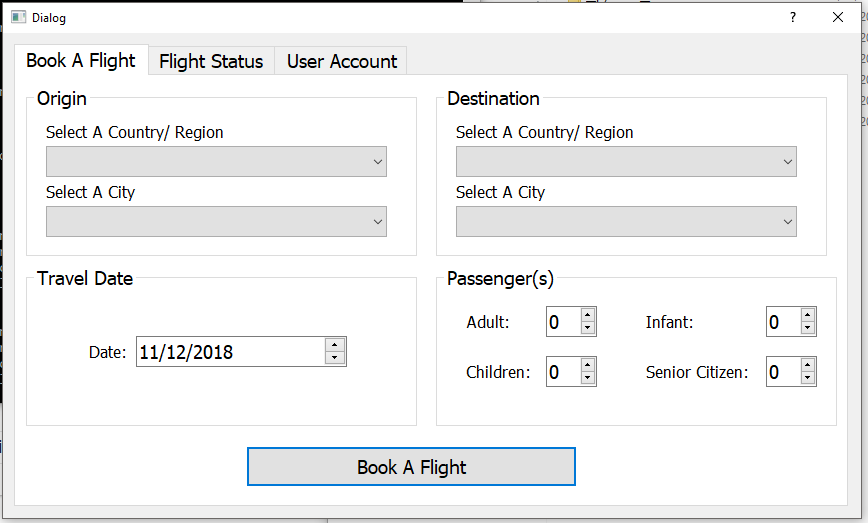
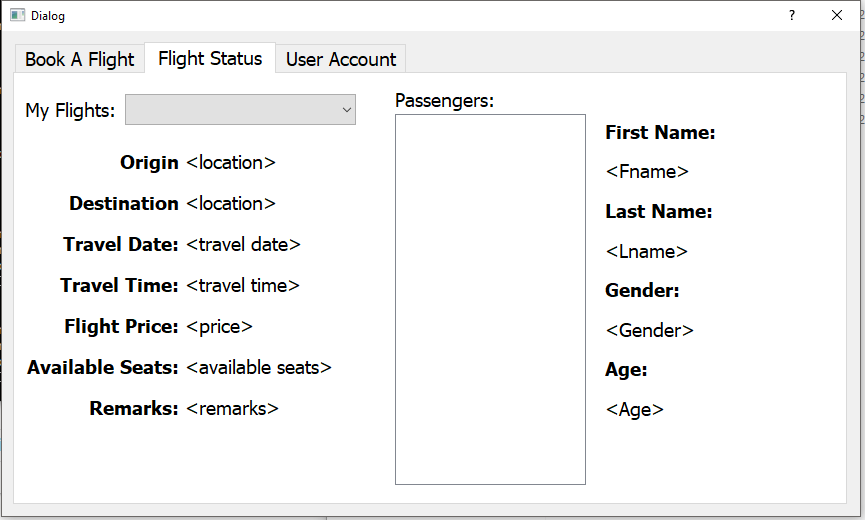
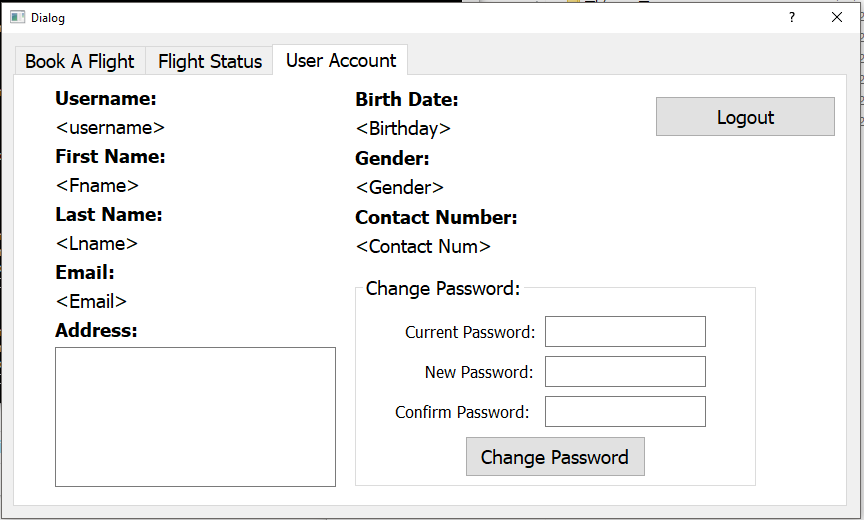
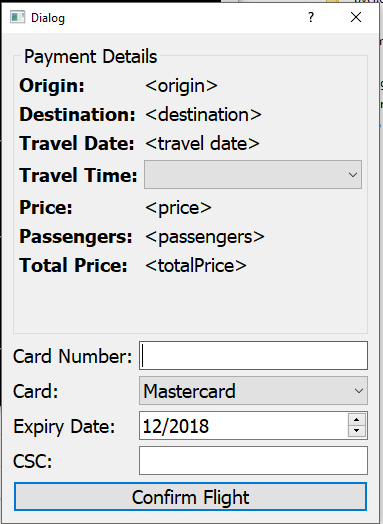


1. Manage Passengers



1. Manage Users



1. Manage Admin  
   
2. Customer login  
   
3. Booking a Flight  
   
4. Flight Status  
   
5. User Account  
   
6. Payment  
   
7. Use Case Diagram

**A picture containing text, map

Description generated with very high confidence**

For the Use Case Diagram of the system, the whole system is seen. It contains use cases and actors outside the box. The system name is Airline Ticket Reservation. As you can see, the ovals represent the system’s function. These functions are search for the available flight, find matching flight, reserve ticket, and login/sign up site. The actor of the system is the customer. The customer is the one who uses the functions except for finding the matching flight. Search engine is the one responsible to find a matching flight.

1. Use Case Specification

|  |  |
| --- | --- |
| **Use Case Name** | Customer Login Page |
| **Description** | Allows the user to login and access their accounts |
| **Actor** | Customer/Passenger |
| **Pre-Condition** | None |
| **Main Sequence** | 1. User enters “username” and “password” 2. Click “Login” 3. Checks missing or wrong username or password 4. Login |
| **Alternative Sequence** | System prompts the user that the information is missing or incorrect |
| **Post Condition** | The user enters the book a flight page. |

|  |  |
| --- | --- |
| **Use Case Name** | Admin Login Page |
| **Description** | Allows the admin to login and access the account |
| **Actor** | Administrator |
| **Pre-Condition** | Only the admins are required to login |
| **Main Sequence** | 1. Admin enters “username” and “password” 2. Click “Login” 3. Checks missing or wrong username or password 4. Login |
| **Alternative Sequence** | System prompts the admin that the information is missing or incorrect |
| **Post Condition** | The admin enters the managing flights, managing passengers, and managing users. |

|  |  |
| --- | --- |
| **Use Case Name** | Sign Up Page |
| **Description** | Allows the user to create an account |
| **Actor** | Customer/Passenger |
| **Pre-Condition** | None |
| **Main Sequence** | 1. Enter all required fields for information. 2. After filling up the required fields, click sign up. |
| **Alternative Sequence** | None |
| **Post Condition** | The user will be directed to the log in page. |

|  |  |
| --- | --- |
| **Use Case Name** | Managing Flights |
| **Description** | Allows the admin to manage the flights of the user. |
| **Actor** | Administrator |
| **Pre-Condition** | The admin must login the admin account |
| **Main Sequence** | 1. Choose the date on the calendar widget. 2. Select the passenger to see the details. 3. Put remarks whenever needed. |
| **Alternative Sequence** | None |
| **Post Condition** | The user database is updated. |

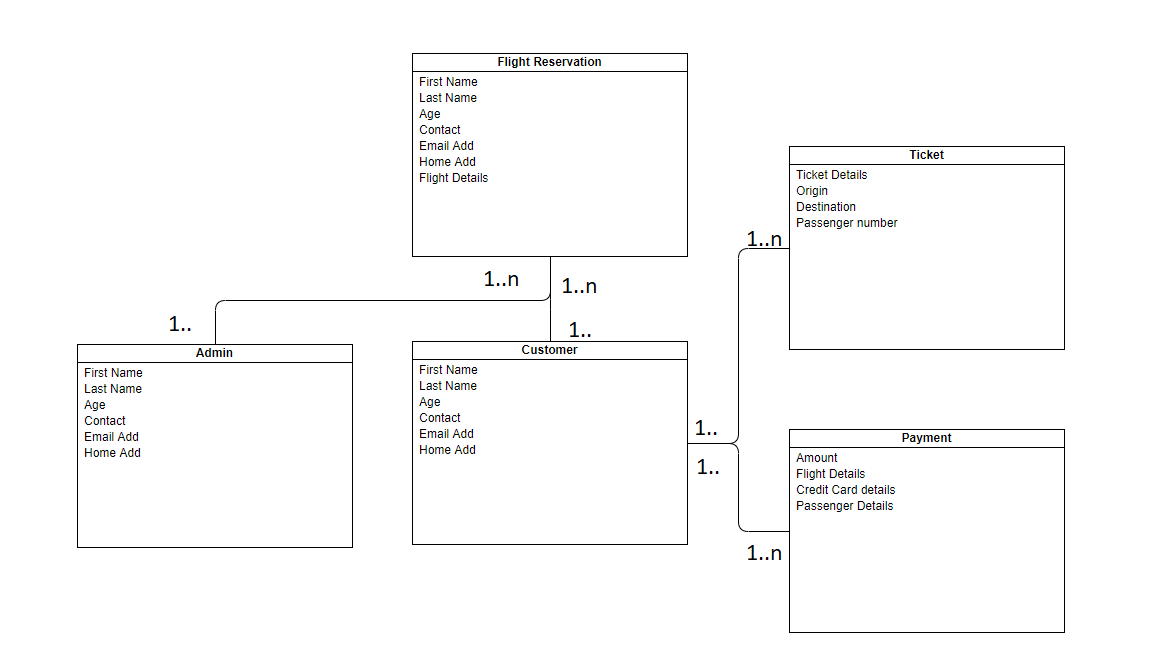
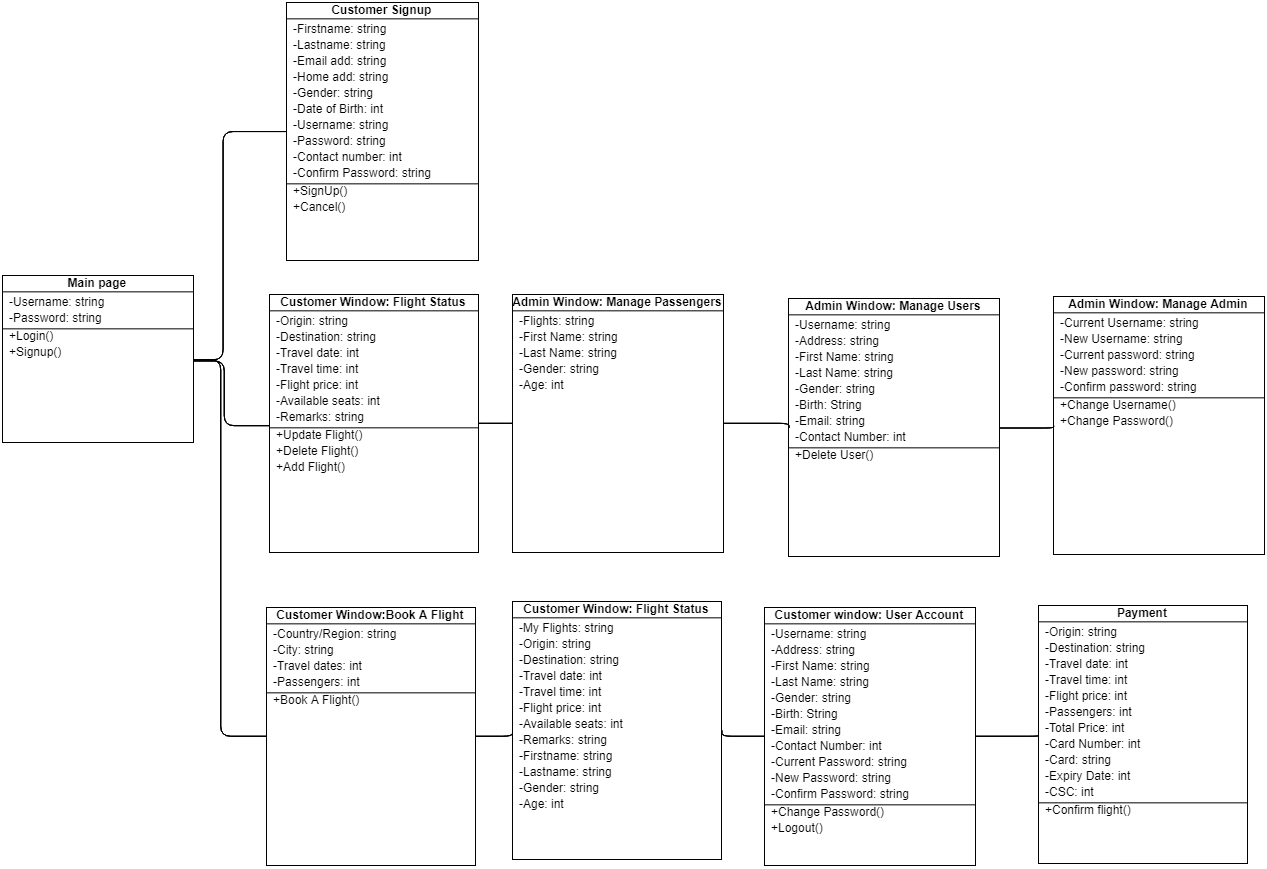
|  |  |
| --- | --- |
| **Use Case Name** | Managing Passenger |
| **Description** | Allows the admin to manage the flights of the user. |
| **Actor** | Administrator |
| **Pre-Condition** | The admin must login the admin account |
| **Main Sequence** | 1. Choose the desired date on the calendar widget. 2. Select the passenger to see the details. 3. The admin can delete the selected passenger. |
| **Alternative Sequence** | None |
| **Post Condition** | The user is deleted in the database. |

|  |  |
| --- | --- |
| **Use Case Name** | Managing Users |
| **Description** | Allows the admin to manage the users that are registered in the system. |
| **Actor** | Administrator |
| **Pre-Condition** | The admin must login the admin account |
| **Main Sequence** | 1. Select the username listed in the program. 2. Delete the user if needed. |
| **Alternative Sequence** | None |
| **Post Condition** | The user is deleted in the database. |

|  |  |
| --- | --- |
| **Use Case Name** | Booking a Flight |
| **Description** | Allows the user to book a flight |
| **Actor** | Customer/Passenger |
| **Pre-Condition** | The user must login the user account |
| **Main Sequence** | 1. Choose if one-way or round trip. 2. Source / Destination 3. Travel Date 4. Passengers |
| **Alternative Sequence** | None |
| **Post Condition** | The user booked a flight |

|  |  |
| --- | --- |
| **Use Case Name** | Flight Status |
| **Description** | Allows the user to see the flight status |
| **Actor** | Customer/Passenger |
| **Pre-Condition** | The admin must login the admin account |
| **Main Sequence** | 1. Choose the desired date on the calendar widget. 2. Select the passenger to see the details. 3. The admin can delete the selected passenger. |
| **Alternative Sequence** | None |
| **Post Condition** | The user is deleted in the database. |

|  |  |
| --- | --- |
| **Use Case Name** | Logout Page |
| **Description** | Allows the user to logout from their account |
| **Actor** | Customer/Passenger |
| **Pre-Condition** | The user must login the user account |
| **Main Sequence** | 1. Click “Logout” |
| **Alternative Sequence** | None |
| **Post Condition** | The user will be directed to the login page |

1. Class Diagram  
     
   Conceptual Class Diagram  
     
   The Figure above is a business level type of representation. The model represents the relationship of each class. [1..] being atleast one and [1..n] being one or more than one. The following attributes and class names are of pseudo types that are near identical to the implementation of the code.   
     
     
     
     
     
     
     
     
     
     
     
     
   Detailed Class Diagram  
   

The figure above is the detailed class diagrams. It shows all the class in a sequence that would most likely happen when the program has began to start. The diagram shows all the attributes and functions that were needed to complete the project.

1. Test Case Diagram

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE #** | **FUNCTION** | **EVENT** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** |
| TC # 1 | User Login | Enter the correct username and password then click Login and press Enter. | Redirected to booking a flight page. | OK |
|  |  | Enter a mismatching username and password then click Login then press Enter. | Reload login page, clear username and password fields. | OK |
|  |  | Enter admin as username and password then click Login then press Enter. | Redirect to managing a flight, managing a passenger, and managing a user’s page. | OK |
| TC # 2 | User Registration | Fill up the necessary information needed then click Sign Up. | Redirect to Login Page. | OK |
|  |  | Incorrect input or lack of input to the required fields. | Shows the message box that it is required to fill up all the required fields. | OK |
| TC # 3 | Managing Flights | Select the desired date and choose the desired passenger in the drop-down list. Click Delete Flight. | The flight of the selected passenger is now deleted in the database. | OK |
|  |  | Select the desired date and choose the desired passenger in the drop-down list. Put Remarks then click Update. | The flight of the selected passenger is now updated in the database. | OK |
| TC # 4 | Managing Passengers | Select the desired date and choose the desired passenger in the drop-down list. Click delete passenger. | The chosen passenger is now deleted in the database. | OKS |
| TC # 5 | Managing Users | Select a username. | The information like the name, email, and address are shown in the screen. | OK |
|  |  | Select a username then click Delete User | The user is now deleted in the database. | OK |
| TC # 6 | Booking a Flight | Select the source or destination, travel dates, and passengers. | The details are displayed on the program. | OK |
| TC # 7 | Flight Status |  |  |  |
| TC # 8 | Payment Method | Select the kind of method when paying. | The system then tells the confirmation. | OK |
| TC # 9 | Logout Page | Click Logout | The user or admin is directed to the login page. | OK |

1. Errors Encountered

|  |  |
| --- | --- |
| Date | Error |
| 9/20/18 | * Failed to upload the document (Gantt Chart) under the Documentation folder in the Github. The file was already reuploaded inside the folder. |
| 9/21/18 | * The title of our project did not match with the system name in the use case diagram. |
| 9/22/18 | * At first, we failed to download the pyqt5 on the path environment because the command prompt is not in the administrator. |
| 9/23/18 | * No errors |
| 9/24/18 | * No errors |
| 9/25/18 | * At first, we were unable to convert the ui to python because of the wrong command. |
| 9/26/18 | * No errors |
| 9/27/18 | * No errors |
| 9/28/18 | * No errors |
| 9/29/18 | * No errors |
| 9/30/18 | * No errors |
| 10/1/18 | * No errors |
| 10/2/18 | * No errors |
| 10/3/18 | * No errors |
| 10/4/18 | * This is the start of Spring3. We were unable to assign correctly the task to each of us. |
| 10/5/18 | * No errors |
| 10/6/18 | * No errors |
| 10/7/18 | * No errors |
| 10/8/18 | * No errors |
| 10/9/18 | * The group unable to finish the tasks at that moment. |
| 10/10/18 | * No errors |
| 10/11/18 | * The Cancel button is not yet working. |
| 10/12/18 | * No errors |

1. Recommendation

The group recommend to assess public reactions to prototype systems and get a mobile-ready application in minutes.

1. Conclusion  
   For the conclusion of this project, we can say that it was a success. Having a finished program up and running, the group achieved the goal to create a Flight reservation system using python environment. The group managed to implement the airline reservation system to the best satisfaction of the customer according to what the group can offer. And also the group managed to use database MySQlite to facilitate this process.