**National University of Computer & Emerging Sciences**

**Karachi Campus**



**Airline Reservation System**

**Project Report**

**Programming Fundamentals - LAB**

**Section: BCS-1G**

**Group Members**

**Faizan Basheer - (24K-0571)**

**Syed Muhammad Sufyan - (24K-0806)**

**Abdur Rehman khan - (24K-0767)**

Submission date: 4th December 2024

* Motivation

We chose the Airline Reservation System project because of its real-life relevance and practical applications. Booking flights can sometimes be a complex and time-consuming process, and we wanted to create a system that simplifies it for users while giving administrators an easy way to manage flights. The idea came from seeing how airline systems work and thinking about ways to make them more accessible and efficient**.**

* Introduction

The Airline Reservation System (ARS) is a software application created to simplify the management and booking of airline tickets. Developed in the C programming language, this system provides a straightforward and user-friendly interface tailored for both customers and airline staff. It allows users to perform a variety of essential operations, including searching for available flights based on their preferences, making reservations, modifying existing bookings, and processing cancellations. Additionally, the system incorporates administrative features that enable airline staff to manage flight schedules, update records, and oversee bookings efficiently. By combining functionality with ease of use, ARS aims to enhance the overall booking experience while streamlining operations for airline management.

* Teamwork Distribution
* **Faizan Basheer:**
* Designed the login menu, including features like registration, user login, admin login, and forgot password.
* Implemented the admin menu, focusing on adding, deleting, modifying, searching, and viewing flights.
* **Syed Muhammad Sufyan:**
* Developed the user menu features, including features like Booking flights, modifying bookings, canceling bookings, displaying flight details and user specific booking.
* Deleting expired flights automatically.
* **Abdur Rehman:**
* Handled error detection and data validation, ensuring the system operates smoothly and manages invalid inputs effectively.
* Implemented features for displaying specific user details for users
* Project Specification:
* **User Features**

1. User Registration and Authentication
2. User record
3. User Status
4. Flight Search
5. Booking Flights
6. View Bookings
7. Modify Bookings
8. Cancel Bookings

* **Admin Features**

1. Flight Management:
2. User Facilities
3. View All Bookings:

4) Search Bookings

* Methodology
* **Tools and Languages Used:**
* Programming Language: **C**
* Development Environment: **Code:** **DevC++**
* Testing Tools: Manual testing of use cases
* Explanation

# Features in the Login Menu:

The Login Menu of the Fast Airline Reservation System provides a userfriendly interface for both users and administrators to interact with the system. Here's an in-depth explanation of each feature within the login menu:

**1. Registration**

The registration feature allows new users to create an account to access the airline booking system. Once registered, users can log in with their credentials to manage their bookings.

* **Process:**

When a user selects the Register option, the program asks for:

**Username:** A unique identifier for the user (within the system).

**Password:** A secret passphrase to secure the user's account. The system enforces a rule that the password must be at least 8 characters long for security purposes.

**Date of Birth (DOB):** Required to calculate the user’s age and for agebased policies (like booking flights).

**Phone Number:** A contact number for the user, useful for communication purposes.

**Gender:** Used for demographic information and potentially for booking requirements.

**Email Address**: A unique email address to facilitate account recovery.

* **Checks and validation**

**Username Uniqueness:** The system checks if the username already exists in the system. If it does, the user is prompted to choose another one.

**Password Length:** Ensures the password entered is at least 8 characters long for enhanced security**.**

* **Data Storage:**

Upon successful registration, the system stores all the user's data in an array, and the updated user list is saved to a file (`information.txt`) for persistence. This allows the system to reload the users' information even after it is restarted**.**

**2. Login as User**

The User Login feature allows registered users to log into the system by providing their username and password. Once logged in, they can access the user menu to manage bookings and personal details.

* **Process:**

The user selects the Login as User option and is prompted to enter:

Username: The username created during the registration process.

Password: The password associated with the entered username.

* **Authentication:**

The system performs a check against the stored usernames and passwords. If both match, the user is granted access to the system, and the login status is updated (`islogin = 1`). If the login is successful, a confirmation message is displayed, and the system transitions to the User Menu.

* **Failure Handling:**

If the entered username or password is incorrect, the system informs the user and provides an opportunity to try logging in again. The program clears the input buffer (`fflush(stdin)`) and prompts the user for reentry.

**3. Login as Admin**

The Admin Login feature is for the administrative user to access the backend of the airline reservation system. Administrators have special privileges, such as managing flight data, handling bookings, and managing user accounts.

* **Process:**

The admin selects the Login as Admin option and is prompted to enter:

Username: The system uses the predefined admin username (`ADMIN\_NAME`) which is "admin".

Password: Similarly, the admin password is also predefined (`ADMIN\_PASSWORD`) and is "1234".

* **Authentication**:

If the credentials match the admin's preset username and password, the system grants access to the Admin Menu. It updates the login status (`islogin = 1, isadmin = 1`).If the login is successful, a confirmation message is shown, and the system moves to the admin menu where administrative tasks can be performed.

* **Failure Handling:**

If the credentials do not match, the system displays an error message and returns the user to the login screen. Again, the input buffer is cleared to prevent previous data from being retained.

**4. Forgot Password**

The Forgot Password option allows users to recover their forgotten passwords by providing their registered username and email address.

* **Process:**

The user is asked to enter:

Username: The username they used to register.

Email Address: The email they registered with, ensuring that only the correct user can retrieve their password**.**

* **Verification:**

The system searches through the list of users. If it finds a match for both the username and email address, it retrieves the stored password and displays it to the user.

* **Failure Handling:**

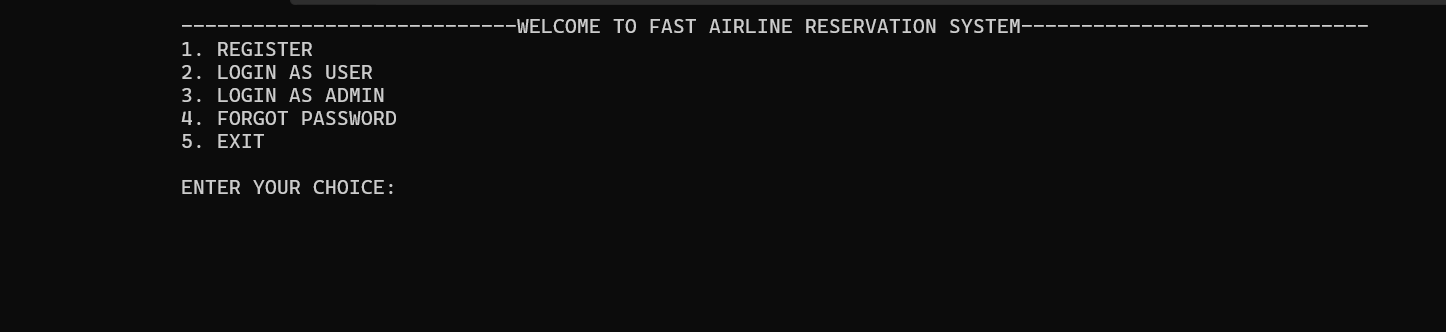
If no matching username or email address is found, the system informs the user that the account does not exist and prompts them to try again.

**5. Exit**

The Exit option allows the user to terminate the program and close the system.

* **Process:**

When selected, the program ends with a thankyou message, and the system exits using the `exit(0)` function.



# User Menu Functionality

The `usermenu()` function in the airline reservation system offers a range of options for registered users to manage their bookings and account details. This function facilitates a seamless interaction with the system by providing easy access to flight information, booking management, and account updates. The function is a crucial component of the user experience, enabling users to perform tasks such as viewing and modifying bookings, checking available flights, and logging out.

**Features of the User Menu**

**1. Displaying the User Menu**

The `usermenu()` function displays a menu with multiple choices. These options allow the user to interact with the system in various ways. The menu is simple and presents a userfriendly interface:

1. User Details

2. View Available Flights

3. Book Flights

4. View Bookings

5. Modify Bookings

6. Cancel Bookings

7. Logout

**2. User Input for Menu Selection**

The user is prompted to enter a choice, which determines the action to be performed. The system accepts an integer input from the user , and based on the input, it executes the corresponding action. The menu is displayed in a loop, and the choice is processed using a `switch` statement.Invalid choices are handled by displaying an error message and returning to the menu for reselection**.**

**3. User Details**

Functionality: The user can view their account details, including username, date of birth, phone number, email, gender, and age. The system calculates the age based on the current date and the user's birth date.

Process: The system fetches the user details from arrays and prints them on the screen. If the user’s birthdate is invalid, an error message is displayed.

**4. View Available Flights**

Functionality: Displays a list of available flights by reading flight data from a file. It shows details such as flight number, origin, destination, departure date, and time.

Process: The function `displayAvailableFlights()` reads flight information, and if flights are available, it presents them to the user. Expired flights are deleted before displaying the available ones**.**

**5. Book Flights**

Functionality: Allows the user to book a flight from the available list. The user selects a flight and chooses a class of travel (Economy, Economy Plus, Business Class, First Class). The system then asks for the number of tickets and saves the booking in a file.

Process: The system verifies the choice, updates the booking information, and saves it in the file. The `saveUser\_Flights()` function writes the user’s booking to a file, including flight number, origin, destination, travel class, and ticket count.

**6. View Bookings**

Functionality: The user can view their past bookings, including details of the flight, travel class, and number of tickets. If there are expired flights, they are deleted before displaying the bookings.

Process: The function `view\_user\_bookings()` displays the bookings of the logged in user. Any expired bookings are removed from the list.

**7. Modify Bookings**

Functionality: This option allows users to modify their existing bookings, such as changing the class of travel or the number of tickets.

Process: The system loads the user’s bookings, and the user can choose a booking to modify. They can select new travel class or update ticket count. The changes are then saved back into the file.

**8. Cancel Bookings**

Functionality: The user can cancel an existing booking. The system displays the user’s current bookings, and the user selects the booking they want to cancel.

Process: The `cancelBooking()` function allows the user to cancel a booking. It uses a temporary file to store bookings while deleting the selected one. After modifying the bookings, the temporary file is renamed to the original file.

**9. Logout**

Functionality: This option logs the user out of the system and returns to the main menu or exit point.

Process: The variable `islogin` is set to 0, which indicates that the user is logged out. The system clears the screen and returns control to the main menu**.**

* **Flow of Execution**

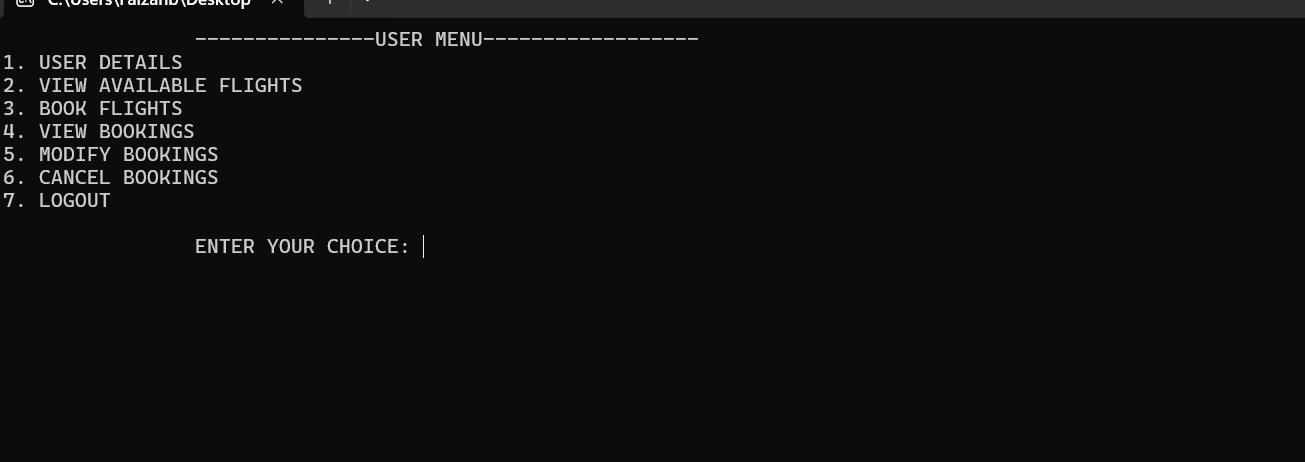
**1. User Access:** The user accesses the menu after logging in. The system first checks if the user is logged in by verifying the login status.

**2. Menu Options:** Upon entering the user menu, the system displays the available options and waits for input.

**3. Choice Processing:** The system processes the user’s choice by invoking the relevant functions for each action. For example, if the user selects "Book Flights," the system displays available flights and processes the booking.

**4. System Feedback:** After each operation, the system provides feedback, such as confirming the booking or displaying an error message for invalid input.

**5. Logout:** When the user chooses to log out, the system clears the login status and returns to the main menu or exit.

****

# Features of Admin Menu:

The provided code implements an Admin Menu system for managing flight bookings, including the ability to add, delete, modify, and search flights, as well as view and update flight bookings. The core functionality is built around flight management, where administrators can perform multiple tasks to ensure smooth operations in the airline reservation system.

**Admin Menu (`adminmenu`):**

This function displays the admin menu and prompts the user to choose one of the following options:

Add Flight: Adds a new flight to the system.

Delete Flight: Deletes a flight from the system.

Modify Flight: Modifies flight details (departure date/time).

View All Bookings: Views all flight bookings made by users.

Search Bookings: Allows the admin to search for specific bookings.

Logout: Logs out the admin and returns to the login screen.

**1. Add Flight (`addFlight`)**

This function allows the admin to add new flight details (flight number, origin, destination, travel date, and departure time).The flight information is appended to a file (`FL`) with a unique serial number for each flight.

The number of flights is tracked, and the flight details are stored in a specific format.

**2. Delete Flight (`deleteFlight`)**

The function loads available flights and displays them to the admin.

Admin can select a flight by its serial number, and the selected flight is deleted from the list.Bookings related to the deleted flight are also updated by marking them as "CANCELED" in the bookings file (`UB`).

**3. Modify Flight (`modify\_flights`)**

The admin can modify details of an existing flight (such as departure date or time).The flight file (`FL`) is updated accordingly, and user bookings are also modified with updated flight information.The system ensures that only the specific details of the flight are changed without affecting other records.

**4. View All Bookings (`view\_all\_bookings`)**

Displays all bookings currently in the system.

Ensures that no duplicate flight entries are shown for the same flight number, displaying each booking only once.

**5. Search Bookings (`searchbookings`)**

Allows the admin to search for bookings based on specific criteria (such as flight number).

If the booking exists, it is displayed; otherwise, the system notifies the admin that no matching bookings were found.

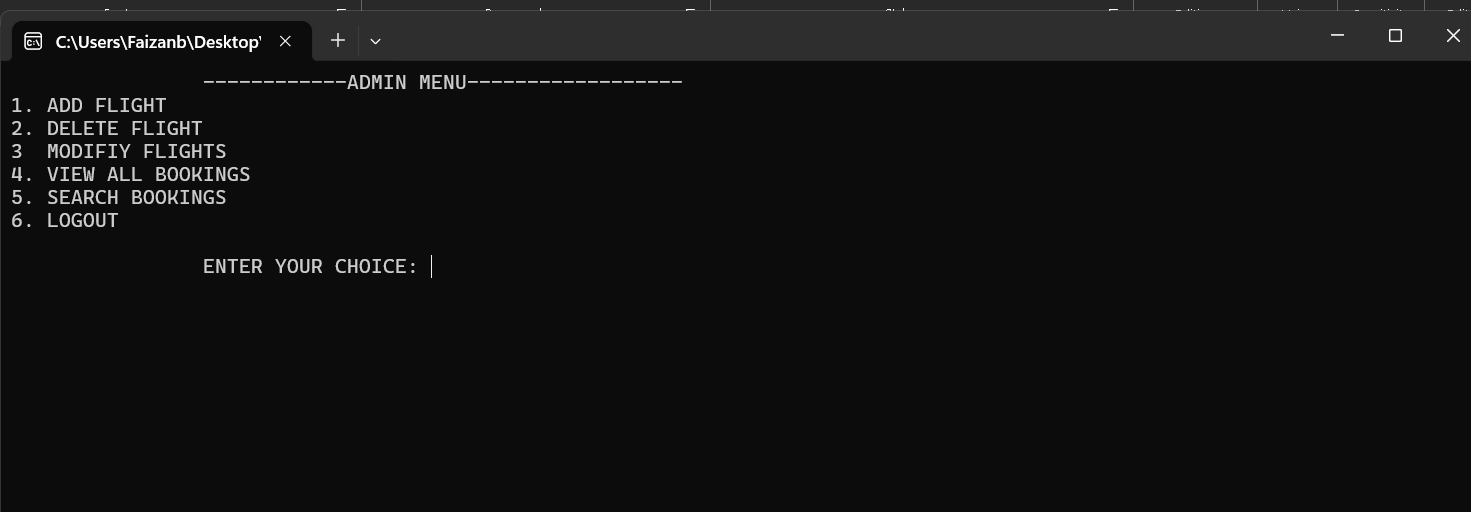
**6. Update User Bookings (`update\_user\_bookings`)**

When a flight is modified, this function ensures that user bookings associated with the flight are updated accordingly (i.e., modifying the departure date and time).

It goes through the booking file and checks for entries matching the modified flight number and updates those bookings.

**8. Handling Canceled Flights**

When a flight is deleted, the bookings associated with that flight are marked as canceled in the bookings file. This ensures the integrity of user data by keeping a clear record of canceled flights.



* File Handling

The system works with several files to manage flight data and user bookings:

Flight File (`FL`): Stores flight information, with each flight's unique identifier.

Bookings File (`UB`): Stores user bookings, with details like flight number, origin, destination, date, time, and the number of tickets.

User Information File(‘F’): Stores the user details such as username, password, email, gender, and phone number.

Temporary files are used for operations such as modifying flight details and handling bookings during deletions to avoid overwriting important data.

* Error Handling

The system includes basic error handling for file operations (e.g., when files cannot be opened or written). It also handles invalid choices in the admin menu and ensures proper validation when selecting flights for modification or deletion.

* Conclusion

In conclusion, the Airline Reservation System developed in C is a comprehensive solution designed to manage flight bookings efficiently and securely. The system focuses on several key aspects, including file management, user interaction, and error handling, ensuring smooth operation in real-world scenarios.

* **Key Features Recap:**

**1. File Handling:** The system manages flight and booking data using text files, with operations such as adding, deleting, and modifying flights being handled through safe and structured file management. Temporary files are used during modification and deletion processes to prevent data corruption.

**2. User Interaction:** The admin interface provides an easy-to-use menu to manage flights and bookings. The system ensures that only authorized users (admins) can access the flight management features, with robust validation for inputs to prevent incorrect data entry.

**3. Error Handling:** The system includes basic error handling mechanisms for common issues, such as file access errors, invalid user inputs, and flight modification failures. This ensures that the system can recover from errors gracefully without crashing or losing data.

**4. Security and Data Integrity:** The careful handling of data, combined with temporary files during modification and deletion, ensures that the integrity of flight and booking data is maintained at all times. The system also validates user inputs to avoid errors that could compromise the flight details or booking process.

**5. Scalability and Maintenance**: The system is built with modularity in mind, allowing for easy addition of new features or future improvements. The use of simple text-based files for data storage makes it easy to back up or migrate the system, although future implementations could move to a database for enhanced scalability and performance.

Overall, the Airline Reservation System is a reliable, user-friendly, and efficient tool for managing flight bookings. While the current version of the system is functional, future enhancements could include a graphical user interface (GUI) for better user experience, integration with real-time flight databases, or the implementation of more advanced error-handling and recovery mechanisms.