ICS1010 MAJOR PROJECT

BUS RESERVATION SYSTEM

The Bus Reservation System is an application designed to facilitate the management of bus bookings, seat reservations, and customer information for a bus service provider.

Key components and functionalities of the system include:

•**User Authentication**: Users can log in with a username and password or create a new account.
•**Bus Information**: Displays a list of available buses with details like bus number, destination, charges, and departure time.
•**Seat Reservation**: Allows users to book seats on buses, specifying the number of seats and passenger details.
•**Cancellation**: Provides the ability to cancel booked tickets by entering the reservation number and seat details.
•**Seat Status Check**: Enables users to check the availability and status of seats in a chosen bus.
•**Data Management**: Uses file handling to store and retrieve passenger data, reservation details, and user credentials.

Functions used and their Functionality

1. . **main()**
 - The main function of the program.
 - Controls the flow of execution by presenting a menu of options to the user and calling corresponding functions based on user input.

Each function serves a specific purpose in managing different aspects of the bus reservation system, including data handling, user interaction, and system logic.

2. **loadDataFromFile()**
 - Loads passenger data from a file (`x`) into the program.
 - Uses the `loadData()` function to read data from the file and construct a binary search tree (`BST`).

3. **saveDataToFile()**
 - Saves passenger data stored in the program (in the binary search tree) to a file (`x`).

4. **savePassengerData(BST *r, FILE *f)**
 - Saves passenger data recursively from a binary search tree node `r` to a file `f`.
 - Uses preorder traversal to save data in the file.

5**. loadData(FILE *f)**
 - Reads passenger data from a file `f` and constructs a binary search tree.
 - Parses each line in the file to extract passenger ID and name, then inserts them into the binary search tree.
 - Updates the `busSeat` array to mark booked seats.

6**. DisplaySeat(int choice)**
 - Displays the seating arrangement for a specific bus `choice`.
 - Reads data from file `x` to check seat status (empty or booked) for each seat in the chosen bus.

7**. insert(BST **r, int custId, char name[])**
 - Inserts a new node with passenger data (`custId` and `name`) into the binary search tree `r`.
 - Uses recursive insertion based on passenger ID (`custId`).

8. **reservationInfo(BST *r, int s, int *custIDmatched)`**
   - Searches for reservation information based on the customer ID `s` in the binary search tree `r`.
   - If a matching reservation is found, prints detailed information about the reservation.
   - Updates the `custIDmatched` flag to indicate if a matching reservation was found.

9. **cost(BST *r)**
   - Calculates and returns the ticket cost based on the bus number stored in the passenger data.
   - Uses a switch-case statement to determine the cost based on the bus number.

10. **bookTicket(int randomNum)**
    - Guides the user through the process of booking tickets.
    - Prompts the user to choose a bus, select seats, enter passenger details, and confirms the booking with a reservation number (`randomNum`).

11. **cancel(int randomNum)**
    - Guides the user through the process of canceling booked tickets.
    - Asks for the reservation number, bus number, and seat numbers to cancel the booking.
    - Updates the `busSeat` array to mark canceled seats as empty.

12. **delete(int lineNumberToRemove)**
    - Deletes a specific line (reservation) from the file (`x`) based on the reservation number (`lineNumberToRemove`).
    - Uses file handling to read the file, copy non-matching lines to a temporary file, and replace the original file with the updated data.

13. **busLists()**
    - Displays a list of available buses with details like bus number, name, destination, charges, and departure time.
    - Helps users choose a bus for booking or checking seat availability.

14. **status()**

- Guides the user to check the status of seats in a chosen bus.
- Prompts the user to enter the bus number and displays the seating arrangement for that bus.

15. **login(int *passcount)**

- Handles user authentication by checking username and password against stored credentials in the file (`file.txt`).
- Tracks the number of login attempts (`passcount`) and exits the program after three failed attempts.

16. **reg()**

- Guides the user through the process of creating a new account.
- Asks for a username and password, checks if the username is available, and stores the new credentials in the file (`file.txt`).

17. **redColor()`, `magnetaColor()`, `green()`, `resetColor()**

- These functions are used to format console output with different colors.
- `redColor()`: Sets the text color to red.
- `magnetaColor()`: Sets the text color to magenta.
- `green()`: Sets the text color to green.
- `resetColor()`: Resets the text color to the default color.

Each function serves a specific purpose in managing different aspects of the bus reservation system, including data handling, user interaction, and system logic.

Contribution :

Pawar Yuvraj Promod (B23CS1051)

•Handled cancellation of booked tickets and implementing measures to protect user privacy and sensitive information.

•Implemented the functionality for user authentication, including login and signup processes.

• Managed user data, ensuring efficient storage, retrieval, ensuring security and privacy of user information.

Kurai Saksham Vinod (B23ME1027)

•Developed the part of the program responsible for displaying seats and buses in an organized manner.

•Designed the user interface (UI) with colors to enhance the visual appeal and user experience.

•Ensured that the program meets the requirements and provides a user-friendly experience for booking bus tickets.

Nitin Rajendra Koronde (B23CH1028)

•Implemented features for checking seat availability, viewing bus details, and navigating through the system's UI.

•Managed planning, coordination, and testing of the system's functionalities

•Made Readme file.

Conclusion:

We have learnt about designing, File handling, project management, problem-solving, and adaptability while creating the Bus Reservation System program. Collaboration with team members enhanced skills in diverse areas, leading to a comprehensive and functional system that prioritizes experience and efficient data handling.