Data Structure

ASSIGNMENT

Group Members

Aakash Goswami	24CC1001
Anshuman Dash	24CC1005
Kaustubh Bhavsar	24CC1010
Kiran Chinthakindi	24CC1023

OUR PROJECT

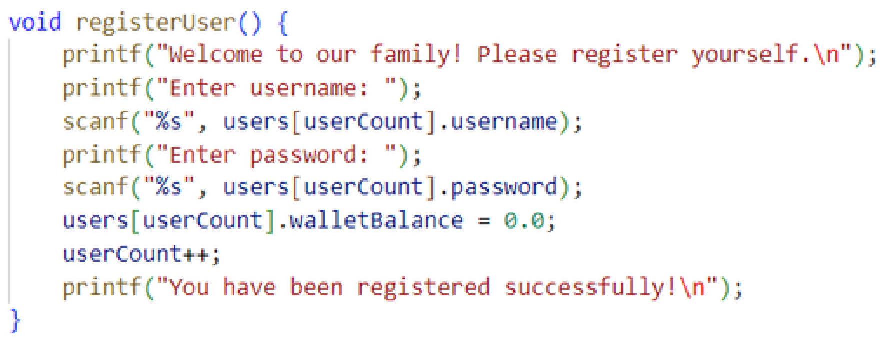
We create a comprehensive train booking system using the C programming language, leveraging linked lists as the core data structure to enable efficient and dynamic management of reservations, cancellations, and user information.

Key Features:

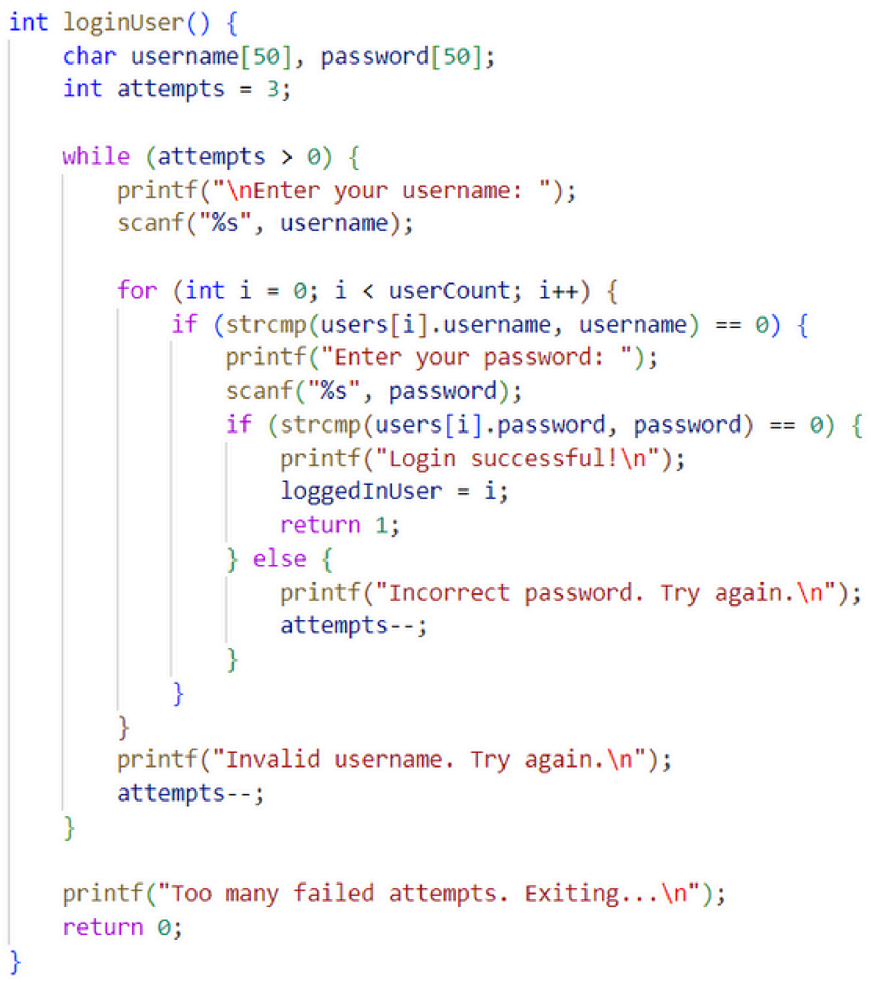
1. User Registration and Login

Users can register with a username and password. They start with a wallet balance of zero.

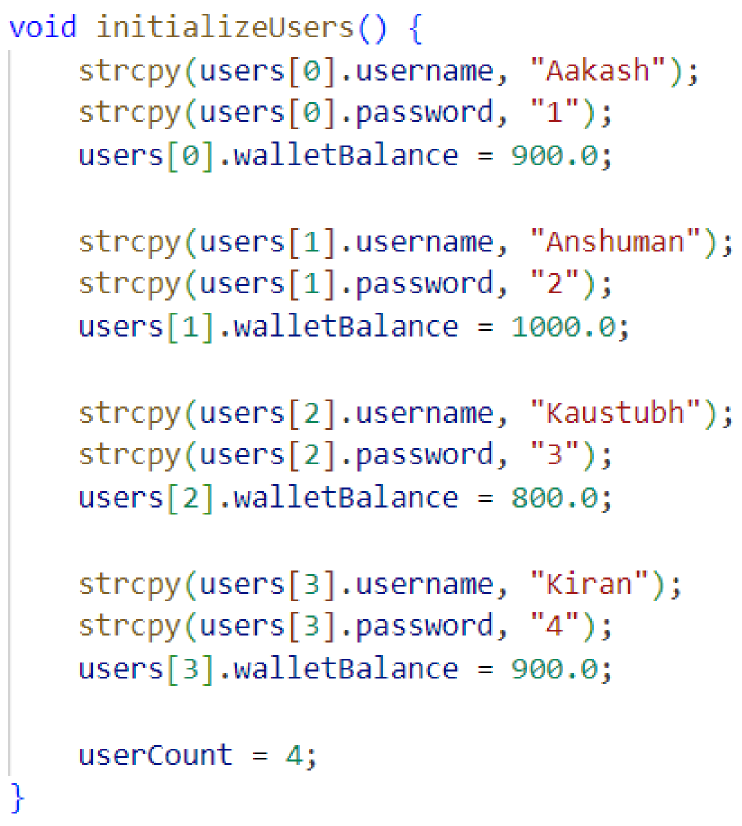
Registered users can log in with a maximum of three attempts allowed.



```c
void registerUser() {
    printf("Welcome to our family! Please register yourself.\n");
    printf("Enter username: ");
    scanf("%s", users[userCount].username);
    printf("Enter password: ");
    scanf("%s", users[userCount].password);
    users[userCount].walletBalance = 0.0;
    userCount++;
    printf("You have been registered successfully!\n");
}
```



```c
int loginUser() {
    char username[50], password[50];
    int attempts = 3;

    while (attempts > 0) {
        printf("\nEnter your username: ");
        scanf("%s", username);

        for (int i = 0; i < userCount; i++) {
            if (strcmp(users[i].username, username) == 0) {
                printf("Enter your password: ");
                scanf("%s", password);
                if (strcmp(users[i].password, password) == 0) {
                    printf("Login successful!\n");
                    loggedInUser = i;
                    return 1;
                } else {
                    printf("Incorrect password. Try again.\n");
                    attempts--;
                }
            }
        }
        printf("Invalid username. Try again.\n");
        attempts--;
    }

    printf("Too many failed attempts. Exiting...\n");
    return 0;
}
```

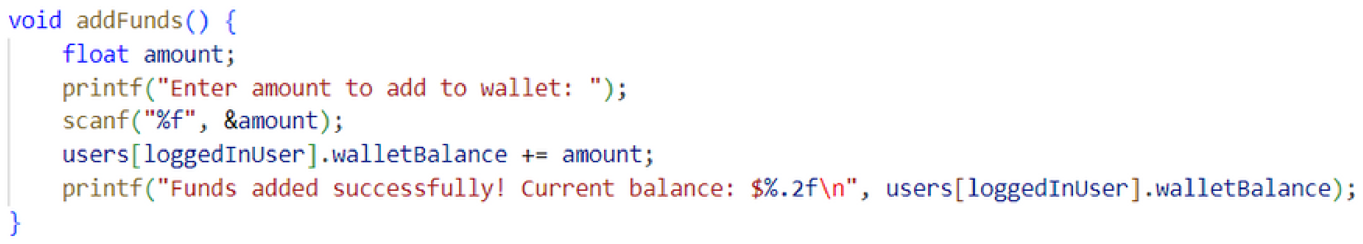


```c
void initializeUsers() {
    strcpy(users[0].username, "Aakash");
    strcpy(users[0].password, "1");
    users[0].walletBalance = 900.0;

    strcpy(users[1].username, "Anshuman");
    strcpy(users[1].password, "2");
    users[1].walletBalance = 1000.0;

    strcpy(users[2].username, "Kaustubh");
    strcpy(users[2].password, "3");
    users[2].walletBalance = 800.0;

    strcpy(users[3].username, "Kiran");
    strcpy(users[3].password, "4");
    users[3].walletBalance = 900.0;

    userCount = 4;
}
```

2. Wallet System:

Each user has a wallet, which they can top up using the **addfunds()** function.

Wallet balance is used to pay for train reservations.



```c
void addFunds() {
    float amount;
    printf("Enter amount to add to wallet: ");
    scanf("%f", &amount);
    users[loggedInUser].walletBalance += amount;
    printf("Funds added successfully! Current balance: $%.2f\n", users[loggedInUser].walletBalance);
}
```

3. Train Database:

The train structure stores information about available trains (train number, source, destination, time, and fare).

A predefined database of trains is initialized in the train array.

```c
// Structure for train database
typedef struct Train {
    int trainNumber;
    char from[50];
    char to[50];
    char time[10];
    float fare;
} Train;
```

```c
// Train database
Train trains[] = {
    {1, "Mumbai", "Allahabad", "10:00 AM", 50.0},
    {2, "Chennai", "Delhi", "2:00 PM", 60.0},
    {3, "Kerala", "Kashmir", "6:00 PM", 70.0},
    {4, "Odisha", "Gujarat", "9:00 PM", 80.0}
};
int trainCount = 4;
```

4. Seat Management with Linked Lists:

Seats are represented using a linked list (seat structure). The **insertseat()** function add seats to the list, and shows available seats.

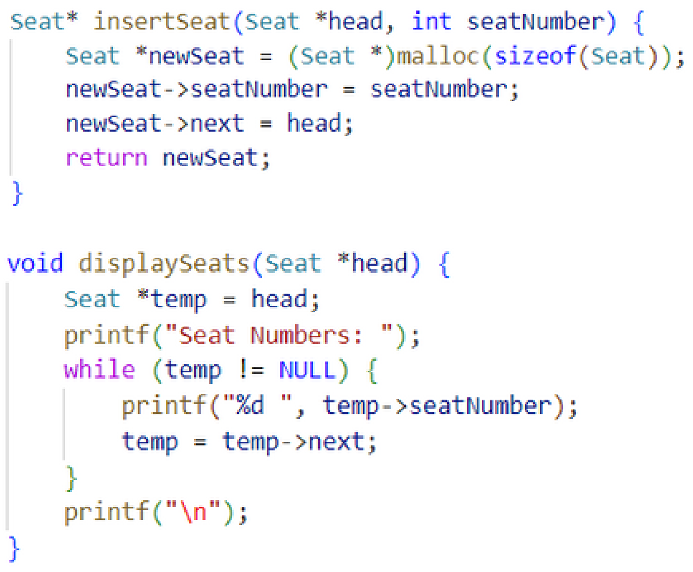
Booked seats are tracked using a global array (bookedSeats).

```c
// Global variables for booked seats and previous stats
int bookedSeats[100] = {0}; // 1 indicates booked, 0 indicates available
typedef struct {
    char username[50];
    int trainNumber;
    int seatNumber;
    float amountPaid;
} PreviousStat;
PreviousStat previousStats[10];
int previousStatsCount = 0;
```

```c
// Structure for linked list to manage train seats
typedef struct Seat {
    int seatNumber;
    struct Seat *next;
} Seat;
```

```c
void displayAvailableTrains() {
    printf("Available trains:\n");
    for (int i = 0; i < trainCount; i++) {
        printf("%d. Train %d - From %s to %s - %s - Rs.%.2f\n",
               i + 1, trains[i].trainNumber, trains[i].from, trains[i].to, trains[i].time, trains[i].fare);
    }
}
```

//Insert_seat function.

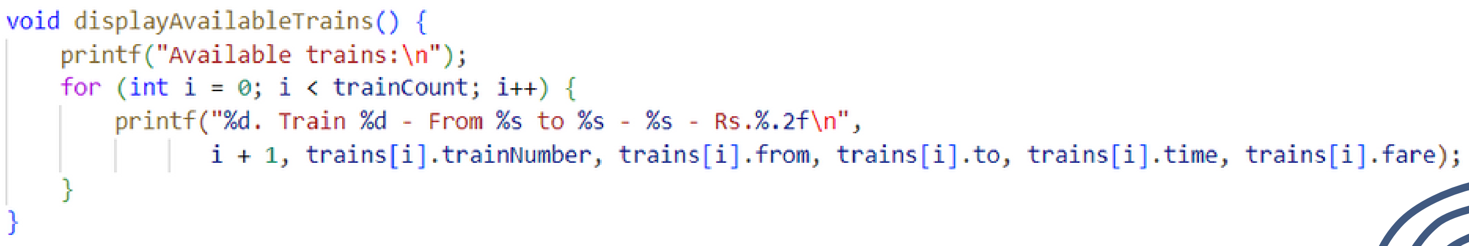


```c
Seat* insertSeat(Seat *head, int seatNumber) {
    Seat *newSeat = (Seat *)malloc(sizeof(Seat));
    newSeat->seatNumber = seatNumber;
    newSeat->next = head;
    return newSeat;
}

void displaySeats(Seat *head) {
    Seat *temp = head;
    printf("Seat Numbers: ");
    while (temp != NULL) {
        printf("%d ", temp->seatNumber);
        temp = temp->next;
    }
    printf("\n");
}
```

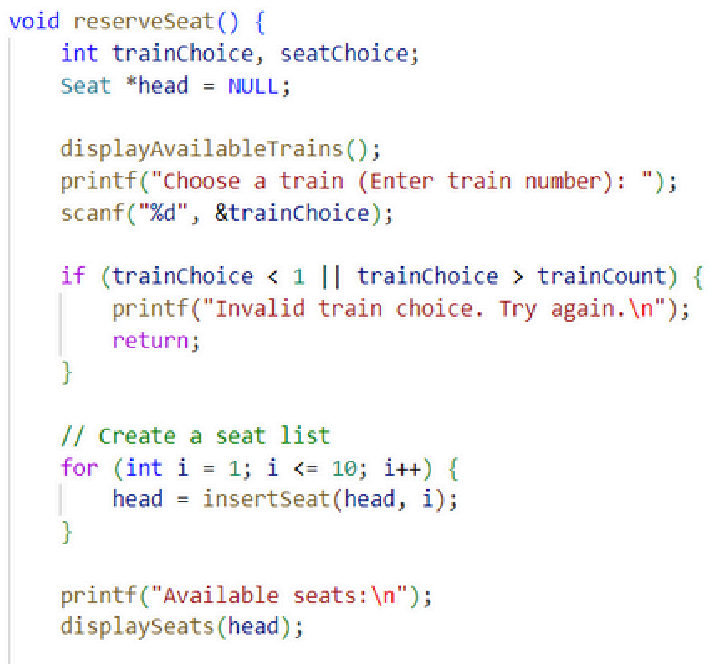
5. Reservation System:

Users can view available trains and seats, and book a seat ().

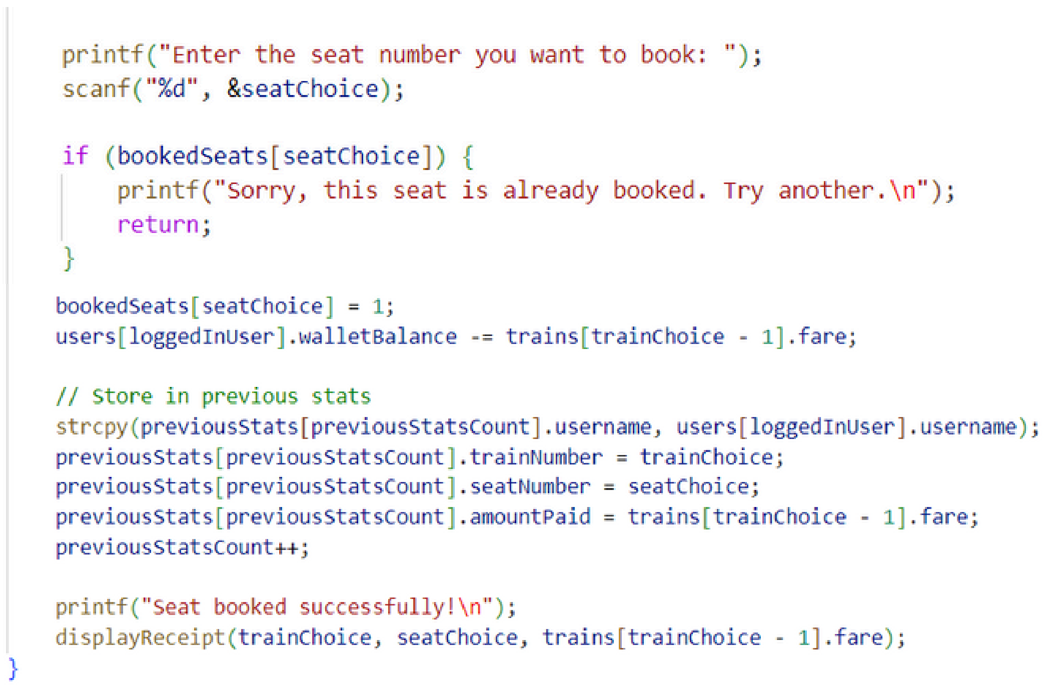
Booked seats are marked as reserved, and the wallet balance is updated. Booking details are saved in previousstats for reference.



```c
void displayAvailableTrains() {
    printf("Available trains:\n");
    for (int i = 0; i < trainCount; i++) {
        printf("%d. Train %d - From %s to %s - %s - Rs.%.2f\n",
                i + 1, trains[i].trainNumber, trains[i].from, trains[i].to, trains[i].time, trains[i].fare);
    }
}
```



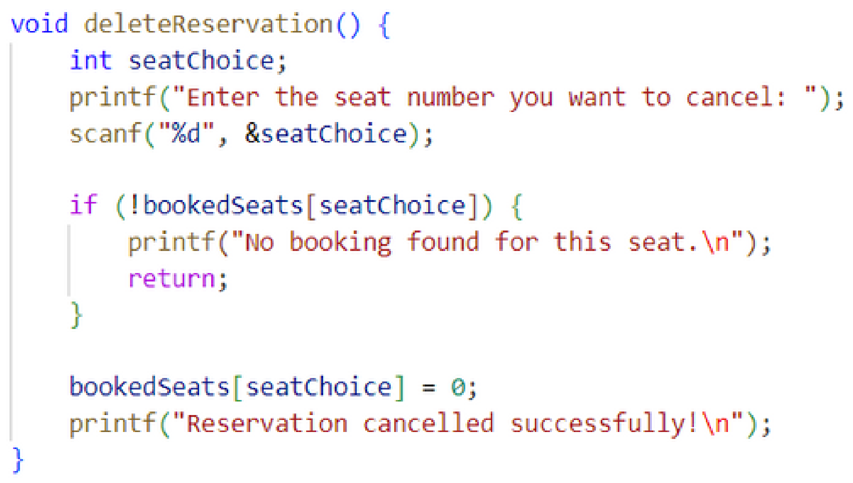
```c
void reserveSeat() {
    int trainChoice, seatChoice;
    Seat *head = NULL;

    displayAvailableTrains();
    printf("Choose a train (Enter train number): ");
    scanf("%d", &trainChoice);

    if (trainChoice < 1 || trainChoice > trainCount) {
        printf("Invalid train choice. Try again.\n");
        return;
    }

    // Create a seat list
    for (int i = 1; i <= 10; i++) {
        head = insertSeat(head, i);
    }

    printf("Available seats:\n");
    displaySeats(head);
```



```c
    printf("Enter the seat number you want to book: ");
    scanf("%d", &seatChoice);

    if (bookedSeats[seatChoice]) {
        printf("Sorry, this seat is already booked. Try another.\n");
        return;
    }
    bookedSeats[seatChoice] = 1;
    users[loggedInUser].walletBalance -= trains[trainChoice - 1].fare;

    // Store in previous stats
    strcpy(previousStats[previousStatsCount].username, users[loggedInUser].username);
    previousStats[previousStatsCount].trainNumber = trainChoice;
    previousStats[previousStatsCount].seatNumber = seatChoice;
    previousStats[previousStatsCount].amountPaid = trains[trainChoice - 1].fare;
    previousStatsCount++;

    printf("Seat booked successfully!\n");
    displayReceipt(trainChoice, seatChoice, trains[trainChoice - 1].fare);
}
```

6. Cancellation System:

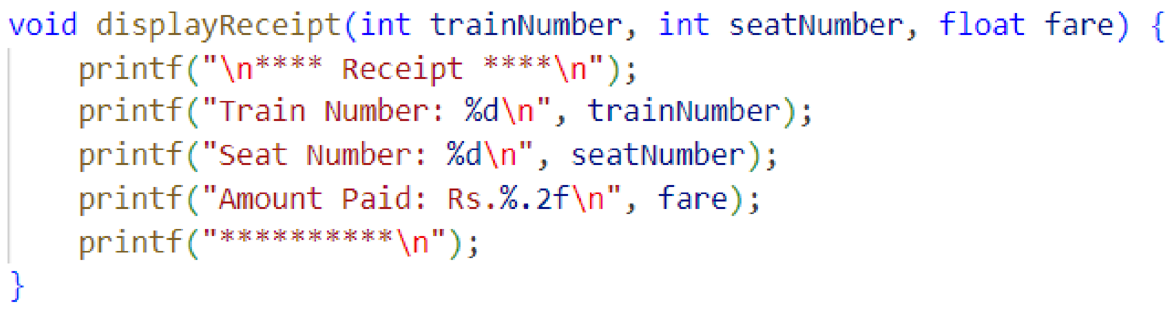
Users can cancel their seat reservations, making seats available again.



```c
void deleteReservation() {
    int seatChoice;
    printf("Enter the seat number you want to cancel: ");
    scanf("%d", &seatChoice);

    if (!bookedSeats[seatChoice]) {
        printf("No booking found for this seat.\n");
        return;
    }

    bookedSeats[seatChoice] = 0;
    printf("Reservation cancelled successfully!\n");
}
```

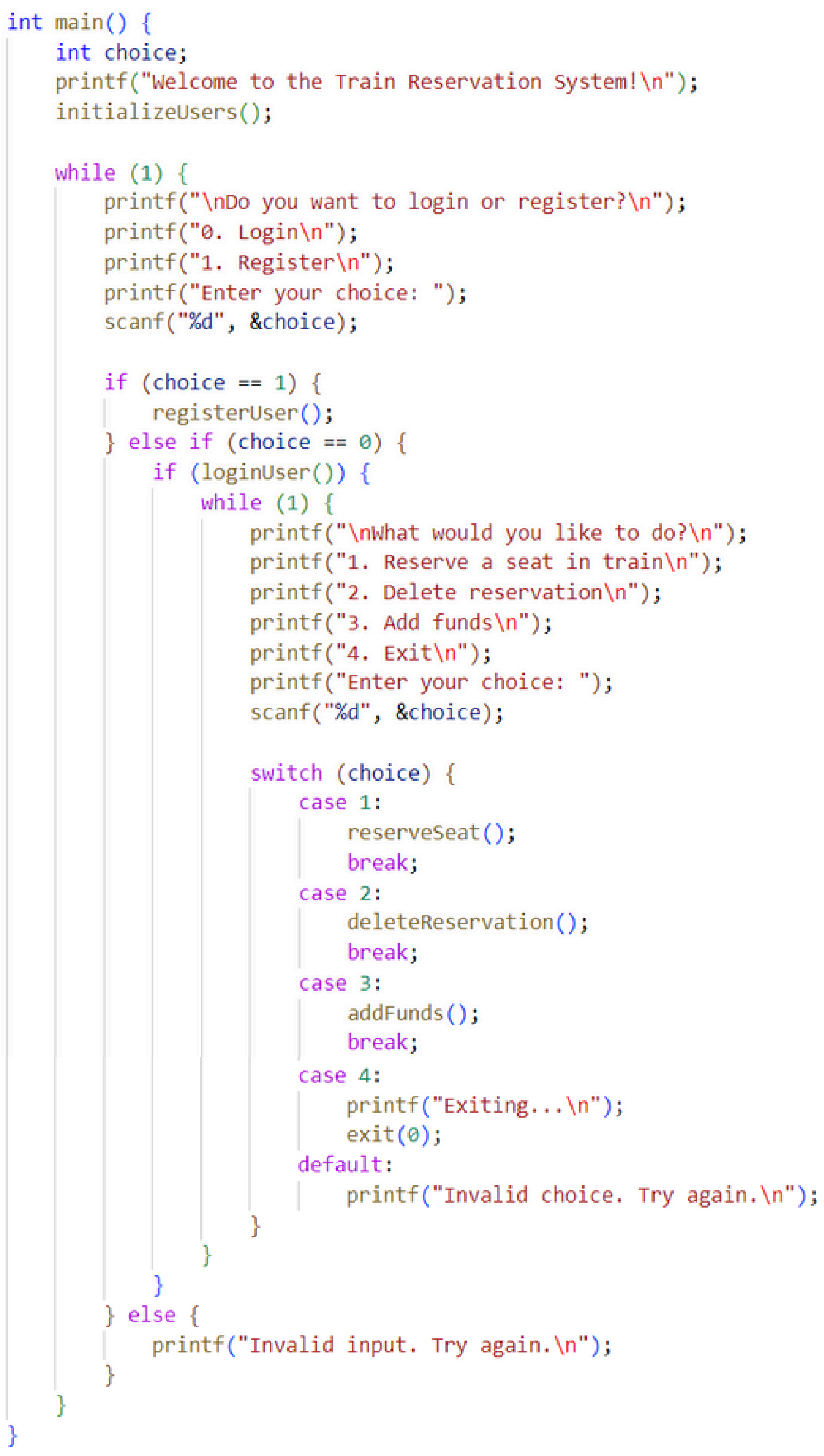
7. Receipt Display:

After booking a seat, a receipt is generated and displayed using **displayreceipt()** function



```c
void displayReceipt(int trainNumber, int seatNumber, float fare) {
    printf("\n**** Receipt ****\n");
    printf("Train Number: %d\n", trainNumber);
    printf("Seat Number: %d\n", seatNumber);
    printf("Amount Paid: Rs.%.2f\n", fare);
    printf("**********\n");
}
```

Main() Function.



```c
int main() {
    int choice;
    printf("Welcome to the Train Reservation System!\n");
    initializeUsers();

    while (1) {
        printf("\nDo you want to login or register?\n");
        printf("0. Login\n");
        printf("1. Register\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        if (choice == 1) {
            registerUser();
        } else if (choice == 0) {
            if (loginUser()) {
                while (1) {
                    printf("\nWhat would you like to do?\n");
                    printf("1. Reserve a seat in train\n");
                    printf("2. Delete reservation\n");
                    printf("3. Add funds\n");
                    printf("4. Exit\n");
                    printf("Enter your choice: ");
                    scanf("%d", &choice);

                    switch (choice) {
                        case 1:
                            reserveSeat();
                            break;
                        case 2:
                            deleteReservation();
                            break;
                        case 3:
                            addFunds();
                            break;
                        case 4:
                            printf("Exiting...\n");
                            exit(0);
                        default:
                            printf("Invalid choice. Try again.\n");
                    }
                }
            }
        } else {
            printf("Invalid input. Try again.\n");
        }
    }
}
```

Program Workflow:

1. Users choose to register or log in.

2. After logging in, they can:

Reserve a seat (choose train and seat, pay fare).

Cancel a reservation.

Add funds to their wallet.

Exit the program.

3. Seats and trains are displayed for the user's reference, and updates occur dynamically.

Example Usage...

A user logs in and chooses a train.

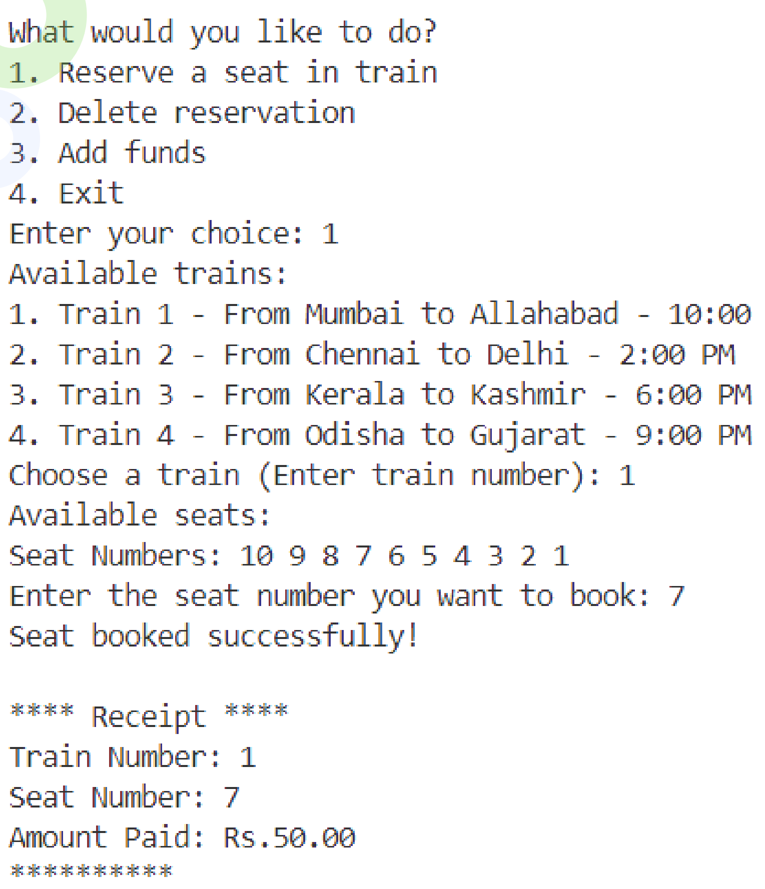
They view and select an available seat.

The seat is booked, and their wallet is charged the train fare.

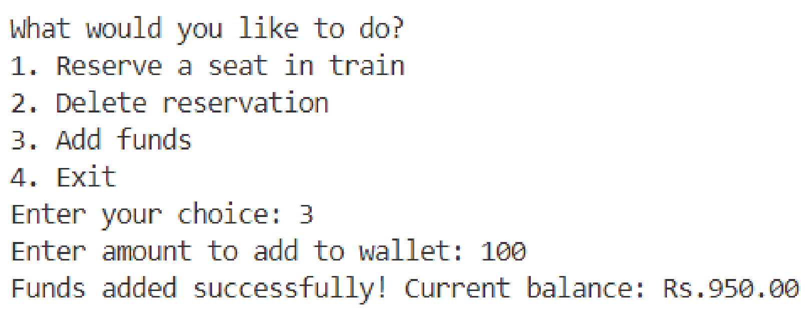
A receipt is displayed, and the booking details are stored for future reference.

Output

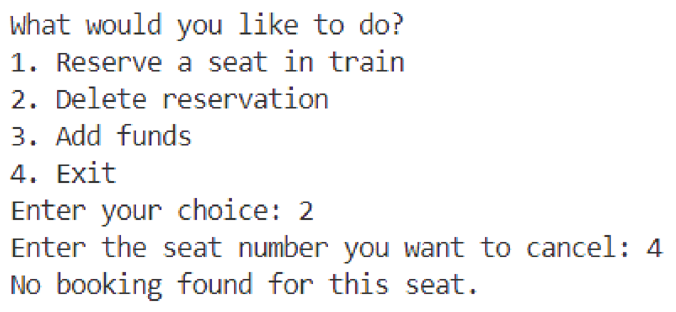
```
Welcome to the Train Reservation System!

Do you want to login or register?
0. Login
1. Register
Enter your choice: 0

Enter your username: Aakash
Enter your password: 1
Login successful!
```



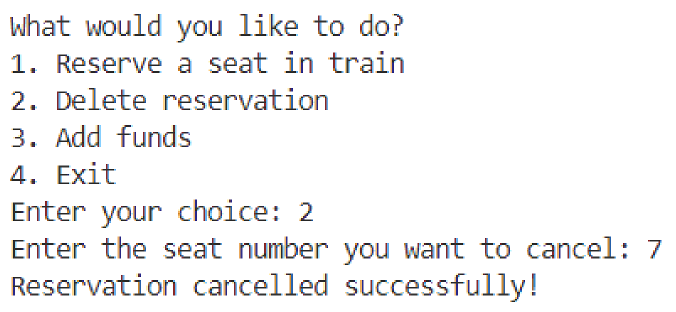
```
What would you like to do?
1. Reserve a seat in train
2. Delete reservation
3. Add funds
4. Exit
Enter your choice: 1
Available trains:
1. Train 1 - From Mumbai to Allahabad - 10:00
2. Train 2 - From Chennai to Delhi - 2:00 PM
3. Train 3 - From Kerala to Kashmir - 6:00 PM
4. Train 4 - From Odisha to Gujarat - 9:00 PM
Choose a train (Enter train number): 1
Available seats:
Seat Numbers: 10 9 8 7 6 5 4 3 2 1
Enter the seat number you want to book: 7
Seat booked successfully!

**** Receipt ****
Train Number: 1
Seat Number: 7
Amount Paid: Rs.50.00
**********
```



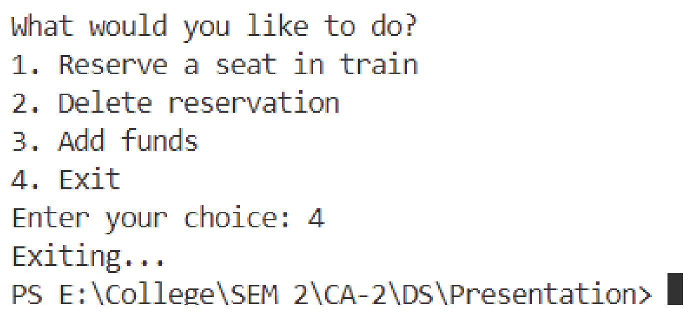
```
What would you like to do?
1. Reserve a seat in train
2. Delete reservation
3. Add funds
4. Exit
Enter your choice: 3
Enter amount to add to wallet: 100
Funds added successfully! Current balance: Rs.950.00
```



```
What would you like to do?
1. Reserve a seat in train
2. Delete reservation
3. Add funds
4. Exit
Enter your choice: 2
Enter the seat number you want to cancel: 4
No booking found for this seat.
```



```
What would you like to do?
1. Reserve a seat in train
2. Delete reservation
3. Add funds
4. Exit
Enter your choice: 2
Enter the seat number you want to cancel: 7
Reservation cancelled successfully!
```



```
What would you like to do?
1. Reserve a seat in train
2. Delete reservation
3. Add funds
4. Exit
Enter your choice: 4
Exiting...
PS E:\College\SEM 2\CA-2\DS\Presentation>
```