Ticket Booking System

Group Name: **Crazy-Engineers**

Project NO:04

Project Name: Ticket Booking System

Group Member Name	Group Member ID
Pallab Mondol	0242220005101380
Sabbir Rahman	0242220005101244
Md.Sobuj Mia	0242220005101064
Ismail Hossain Rakib	0242220005101128

Automated Bus Ticket Booking System

1. Introduction The Automated Bus Ticket Booking System is a software application designed to facilitate the booking, editing, and cancellation of bus tickets. It provides a user-friendly interface for passengers to check seat availability, reserve seats, modify bookings, and cancel reservations. The system utilizes data structures such as linked lists, stacks, and queues to efficiently manage and organize the ticketing process.

2. Objective The primary objective of the Automated Bus Ticket Booking System is to streamline the ticket booking process and provide a convenient and efficient way for passengers to manage their reservations. The system aims to automate manual ticketing operations, reduce human errors, and enhance the overall user experience. By utilizing data structures, the system ensures effective seat management and allows for quick retrieval and modification of ticket information.

3. Description The Automated Bus Ticket Booking System is implemented using the C programming language. It utilizes a combination of linked lists, stacks, and queues to perform various operations such as seat booking, editing bookings, and deleting reservations. The system presents a menu-driven interface to users, allowing them to interact with the system through a series of options.

Upon launching the system, users are presented with a menu that includes options to display available seats, book a seat, edit a booking, delete a reservation, and exit the system. The system maintains a linked list of seat nodes, where each node represents a seat and stores information such as the seat number and passenger name. When a seat is booked, a new node is created and added to the linked list.

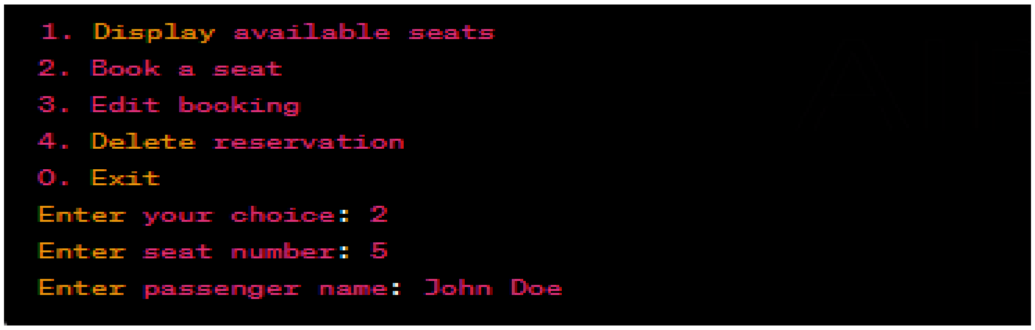
The system employs functions to perform specific tasks. The `displaySeats` function traverses the linked list and displays the current seat status, including the seat number and passenger name. The `searchSeat` function enables the system to search for a specific seat in the linked list. The `bookSeat` function allows users to reserve a seat by entering the seat number and passenger name. The `editBooking` function enables users to modify their booking by searching for a seat and updating the passenger name. The `deleteReservation` function allows users to cancel their reservation by searching for a seat and removing the corresponding node from the linked list.

4. Requirement Specification The Automated Bus Ticket Booking System has the following requirements:

- The system should provide a menu-driven interface for users to interact with.
- Users should be able to view available seats and their current status.
- Users should be able to book a seat by entering the seat number and passenger name.
- The system should prevent double booking of seats.
- Users should be able to edit their booking by searching for a seat and updating the passenger name.
- Users should be able to cancel their reservation by searching for a seat and deleting the corresponding node.
- The system should handle invalid inputs and display appropriate error messages.
- The program should be implemented in the C programming language.

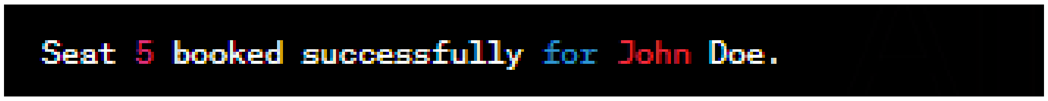
5. Sample Input and Output

Sample Input:



```
1. Display available seats
2. Book a seat
3. Edit booking
4. Delete reservation
0. Exit
Enter your choice: 2
Enter seat number: 5
Enter passenger name: John Doe
```

Sample Output:



```
Seat 5 booked successfully for John Doe.
```

Ticket Booking System

6. Challenges During the implementation of the Automated Bus Ticket Booking System, several challenges were encountered. Some of the notable challenges include:

- Efficient management of seat availability and booking information using data structures.
- Handling input validation to ensure correct and error-free user interactions.
- Implementing the edit and delete functionalities without affecting the integrity of the linked list.
- Ensuring proper memory management and avoiding memory leaks.

These challenges were addressed by carefully designing the system's data structures, implementing appropriate validation checks, and thoroughly testing the functionality to ensure reliability and stability.

7. Conclusion The Automated Bus Ticket Booking System provides a convenient and automated solution for managing bus ticket reservations. By utilizing linked lists, stacks, and queues, the system efficiently handles seat booking, editing, and cancellation. The program's user-friendly interface and robust functionality make it a valuable tool for both passengers and bus operators. With further enhancements and scalability, the system can be extended to support additional features such as payment processing and real-time seat availability updates, further improving the ticketing experience for users.

Code

```
#include <stdio.h>

#include <stdlib.h>


// Node structure for linked list

typedef struct Node {

   int seatNumber;

   char passengerName[50];

   struct Node* next;

} Node;


// Function prototypes

Node* createNode(int seatNumber, char* passengerName);
```

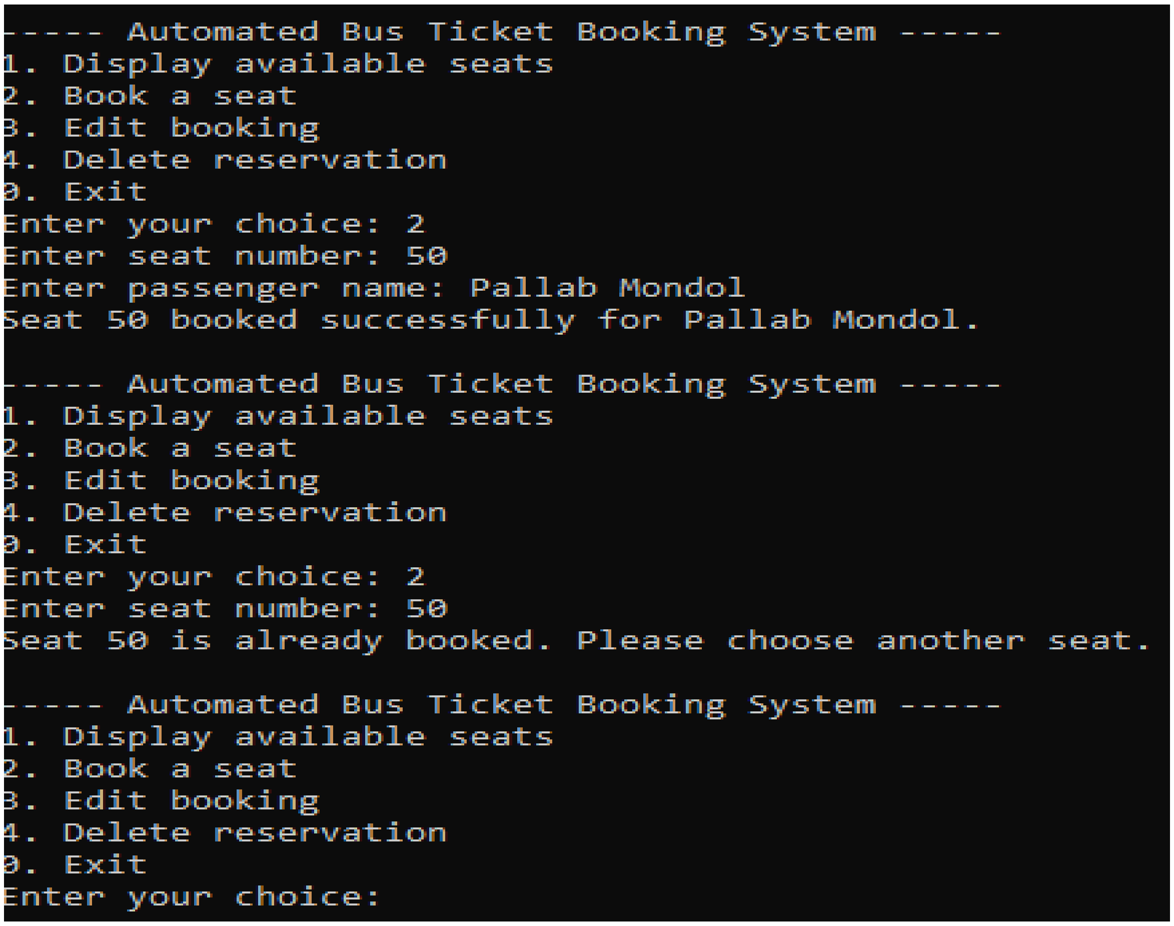
Ticket Booking System

```
void insertAtEnd(Node** head, int seatNumber, char* passengerName);

void displaySeats(Node* head);

Node* searchSeat(Node* head, int seatNumber);

void bookSeat(Node** head);

void editBooking(Node* head);

void deleteReservation(Node** head);


// Global variables

Node* seatList = NULL;


int main() {

  int choice;


  do {

    printf("\n----- Automated Bus Ticket Booking System -----\n");

    printf("1. Display available seats\n");

    printf("2. Book a seat\n");

    printf("3. Edit booking\n");

    printf("4. Delete reservation\n");

    printf("0. Exit\n");

    printf("Enter your choice: ");

    scanf("%d", &choice);


    switch (choice) {

      case 1:

        displaySeats(seatList);

        break;

      case 2:

        bookSeat(&seatList);
```

```
            break;
          case 3:
            editBooking(seatList);
            break;
          case 4:
            deleteReservation(&seatList);
            break;
          case 0:
            printf("Thank you for using the ticket booking system!\n");
            break;
          default:
            printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 0);

    return 0;
}

// Create a new node
Node* createNode(int seatNumber, char* passengerName) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->seatNumber = seatNumber;
    strcpy(newNode->passengerName, passengerName);
    newNode->next = NULL;
    return newNode;
}

// Insert a node at the end of the linked list
void insertAtEnd(Node** head, int seatNumber, char* passengerName) {
    Node* newNode = createNode(seatNumber, passengerName);

    if (*head == NULL) {
        *head = newNode;
    } else {
        Node* current = *head;
        while (current->next != NULL) {
            current = current->next;
        }
        current->next = newNode;
    }

    printf("Seat %d booked successfully for %s.\n", seatNumber, passengerName);
}

// Display all seats and their status
void displaySeats(Node* head) {
    if (head == NULL) {
        printf("No seats booked yet.\n");
        return;
    }

    printf("Seat\tPassenger Name\n");
    printf("----\t--------------\n");

    Node* current = head;
    while (current != NULL) {
        printf("%d\t%s\n", current->seatNumber, current->passengerName);
        current = current->next;
    }
}

// Search for a seat in the linked list
Node* searchSeat(Node* head, int seatNumber) {
    Node* current = head;
    while (current != NULL) {
        if (current->seatNumber == seatNumber) {
            return current;
        }
        current = current->next;
    }
    return NULL;
}

// Book a seat
void bookSeat(Node** head) {
    int seatNumber;
    char passengerName[50];

    printf("Enter seat number: ");
    scanf("%d", &seatNumber);

    if (searchSeat(*head, seatNumber) != NULL) {
        printf("Seat %d is already booked. Please choose another seat.\n", seatNumber);
    } else {
        printf("Enter passenger name: ");
        scanf(" %[^\n]s", passengerName);
        insertAtEnd(head, seatNumber, passengerName);
    }
}

// Edit booking
void editBooking(Node* head) {
    int seatNumber;
    char passengerName[50];

    printf("Enter seat number to edit booking: ");
    scanf("%d", &seatNumber);

    Node* seatNode = searchSeat(head, seatNumber);
    if (seatNode == NULL) {
        printf("Seat %d is not booked.\n", seatNumber);
    } else {
        printf("Enter new passenger name: ");
        scanf(" %[^\n]s", passengerName);
        strcpy(seatNode->passengerName, passengerName);
        printf("Booking for seat %d updated successfully.\n", seatNumber);
    }
}

// Delete reservation
void deleteReservation(Node** head) {
    int seatNumber;

    printf("Enter seat number to delete reservation: ");
    scanf("%d", &seatNumber);
    Node* current = *head;
    Node* prev = NULL;

    while (current != NULL) {
        if (current->seatNumber == seatNumber) {
            if (prev == NULL) {
                *head = current->next;
            } else {
                prev->next = current->next;
            }
            free(current);
            printf("Reservation for seat %d deleted successfully.\n", seatNumber);
            return;
        }
        prev = current;
        current = current->next;
    }

    printf("Seat %d is not booked.\n", seatNumber);
}
```

Ticket Booking System

Sample Output:



```
----- Automated Bus Ticket Booking System -----
1. Display available seats
2. Book a seat
3. Edit booking
4. Delete reservation
0. Exit
Enter your choice: 2
Enter seat number: 50
Enter passenger name: Pallab Mondol
Seat 50 booked successfully for Pallab Mondol.

----- Automated Bus Ticket Booking System -----
1. Display available seats
2. Book a seat
3. Edit booking
4. Delete reservation
0. Exit
Enter your choice: 2
Enter seat number: 50
Seat 50 is already booked. Please choose another seat.

----- Automated Bus Ticket Booking System -----
1. Display available seats
2. Book a seat
3. Edit booking
4. Delete reservation
0. Exit
Enter your choice:
```