Name- Atharva Sanjay shevate

Roll no-2

div-E(E1)

srn-202201727

BUS RESERVATION SYSTEM

INTRODUCTION

This presentation explores a comprehensive C-based implementation of a bus reservation system.

The system emulates real-world bus reservation services and is designed to offer features like seat booking, cancellation, and status checking.

This presentation will delve into the code structure, key components, and its potential applications.

CODE

```
#include <stdio.h>

#include <stdbool.h>

#include <stdlib.h>
```

```
#include <string.h>

#define MAX_SEATS 20

typedef struct {
    int seatNo;
    int busType;  // 1 for AC, 2 for non-AC
    int seatType; // 1 for window, 2 for non-window
    bool isOccupied;
    char passengerName[50];
} SeatReservation;

void initializeSeats(SeatReservation seats[], int busType) {
    for (int i = 0; i < MAX_SEATS; i++) {
        seats[i].seatNo = i + 1;
        seats[i].busType = busType;
        seats[i].seatType = i % 2 == 0 ? 1 : 2; // Even seats are
window seats
        seats[i].isOccupied = false;
        strcpy(seats[i].passengerName, "");
    }
}
```

```
void bookSeat(SeatReservation seats[], int seatNo, char passengerName[]) {
    if (seatNo < 1 || seatNo > MAX_SEATS) {
        printf("Invalid seat number.\n");
        return;
    }

    SeatReservation *seat = &seats[seatNo - 1];
    if (seat->isOccupied) {
        printf("Seat %d is already occupied by %s.\n", seat->seatNo, seat->passengerName);
    } else {
        seat->isOccupied = true;
        strcpy(seat->passengerName, passengerName);
        printf("Seat %d has been booked successfully for %s.\n", seat->seatNo, passengerName);
    }
}

void cancelSeat(SeatReservation seats[], int busType, char passengerName[]) {
    printf("Enter the seat number to cancel: ");
```

```
    int seatNo;
    scanf("%d", &seatNo);

    for (int i = 0; i < MAX_SEATS; i++) {
        if (seats[i].busType == busType &&
strcmp(seats[i].passengerName, passengerName) == 0 &&
seats[i].seatNo == seatNo) {
            seats[i].isOccupied = false;
            strcpy(seats[i].passengerName, "");
            printf("Seat %d has been canceled successfully for
%s.\n", seatNo, passengerName);
            return;
        }
    }

    printf("Seat not found for cancellation.\n");
}

int main() {
    SeatReservation acSeats[MAX_SEATS];
    SeatReservation nonACSeats[MAX_SEATS];
    initializeSeats(acSeats, 1); // Initialize AC bus seats
    initializeSeats(nonACSeats, 2); // Initialize non-AC bus seats
```

```
    int choice;
    char passengerName[50];

    do {
        printf("Welcome to the Bus Reservation System!\n");
        printf("1. Book a seat\n");
        printf("2. Cancel a seat\n");
        printf("3. Check bus status\n");
        printf("0. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                int busType, seatType, seatNo;
                printf("Choose Bus Type (1 for AC, 2 for non-AC): ");
                scanf("%d", &busType);
                printf("Choose Seat Type (1 for window, 2 for non-
window): ");
                scanf("%d", &seatType);
                printf("Enter Passenger Name: ");
```

```
            scanf(" %[^\n]", passengerName);
            printf("Enter Seat Number: ");
            scanf("%d", &seatNo);
            bookSeat((busType == 1) ? acSeats : nonACSeats,
seatNo, passengerName);
            break;

        case 2:
            int cancelBusType;
            printf("Choose Bus Type for Cancellation (1 for AC, 2
for non-AC): ");
            scanf("%d", &cancelBusType);
            printf("Enter Passenger Name for Cancellation: ");
            scanf(" %[^\n]", passengerName);
            cancelSeat((cancelBusType == 1) ? acSeats :
nonACSeats, cancelBusType, passengerName);
            break;

        case 3:
            printf("Bus Status (AC):\n");
            for (int i = 0; i < MAX_SEATS; i++) {
                if (acSeats[i].isOccupied) {
```

```
                printf("Seat %d (Bus Type: %d, Seat Type: %d) -
Occupied by %s\n", acSeats[i].seatNo, acSeats[i].busType,
acSeats[i].seatType, acSeats[i].passengerName);
            }
        }

        printf("\nBus Status (Non-AC):\n");
        for (int i = 0; i < MAX_SEATS; i++) {
            if (nonACSeats[i].isOccupied) {
                printf("Seat %d (Bus Type: %d, Seat Type: %d) -
Occupied by %s\n", nonACSeats[i].seatNo,
nonACSeats[i].busType, nonACSeats[i].seatType,
nonACSeats[i].passengerName);
            }
        }
        break;

    case 0:
        printf("Thank you for using the Bus Reservation
System!\n");
        break;

    default:
```

```
            printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 0);

    return 0;
}
```

Code structure

- **The code is organized into functions for initialization, booking, cancellation, and status checking.**
- **It employs structured programming principles to ensure modularity and maintainability.**
- **Makes use of data structures like arrays and structs to represent seat reservations efficiently.**
- **Contains clear and concise code comments to improve code readability.**
- **Follows best practices for error handling and input validation.**

Conclusion

The Bus Reservation System code represents a robust and versatile solution for streamlining bus seat reservations and enhancing the overall passenger experience. With a well-structured codebase and a range of features, this system offers numerous advantages and possibilities.

The system simplifies the booking and cancellation of seats, providing a user-friendly interface for passengers while reducing manual workload for bus companies and travel agencies. Its modular code structure ensures maintainability and scalability, making it suitable for a wide range of applications within the transportation industry.