Computer Programming

CSE 101 Final Report

Project Title: Electronic Bus Ticket Generator

S.No	Name	Roll No	Reg No
1.	Devarakonda Srikaran	RKOCBGB65	12220309
2.	Vipin Kumar	RKOCBGB66	12220492
3.	Singampalli Devi Sri Prasad	RKOCBGB67	12220538

Submitted By :

Devarakonda Srikaran

Submitted to :

Nahita Pathania

Acknowledgement:

I would like to express my gratitude to all those who have contributed to the successful completion of my CSE101 Computer Programming final project report.

First and foremost, I would like to thank my instructor for providing me with the knowledge and guidance needed to complete this project. Their unwavering

support and constructive feedback have been invaluable throughout the process.

I am also grateful to my classmates, for their continuous encouragement, insightful discussions, and collaboration during the project.

Furthermore, I would like to extend my appreciation to the resources provided by the University, such as the computer lab facilities, online libraries, and tutorials, which have enabled me to gain a deeper understanding of the programming concepts and tools.

Lastly, I would like to express my heartfelt thanks to my family and friends for their love, encouragement, and motivation throughout my academic journey. Their unwavering support has been instrumental in helping me achieve my goals.

Thank you all once again.

Introduction:

The Electronic Bus Ticket Generator is a computerized system designed to simplify the process of booking bus tickets, allowing users to view the available buses for a particular route, select their preferred bus, and book their seats online. The system also enables users to cancel their bookings and provides information on available buses, including their fares. The project was developed to streamline the ticketing process, providing an e cient and user-friendly platform for both users and administrators. The system was designed to be scalable, allowing for the addition of new buses and routes, and can be customized to meet the specific needs of di erent bus companies. Overall, the Electronic Bus Ticket Generator is a valuable tool that o ers a seamless ticketing experience for bus travelers.

Modules:

Display available buses for a route: This module displays the number of available buses for a particular route with their fares. It allows users to select the desired bus for their travel.

Add new bus: This module allows the administrator to add new buses to the system. It requires input of details such as bus number, route, fare, and available seats.

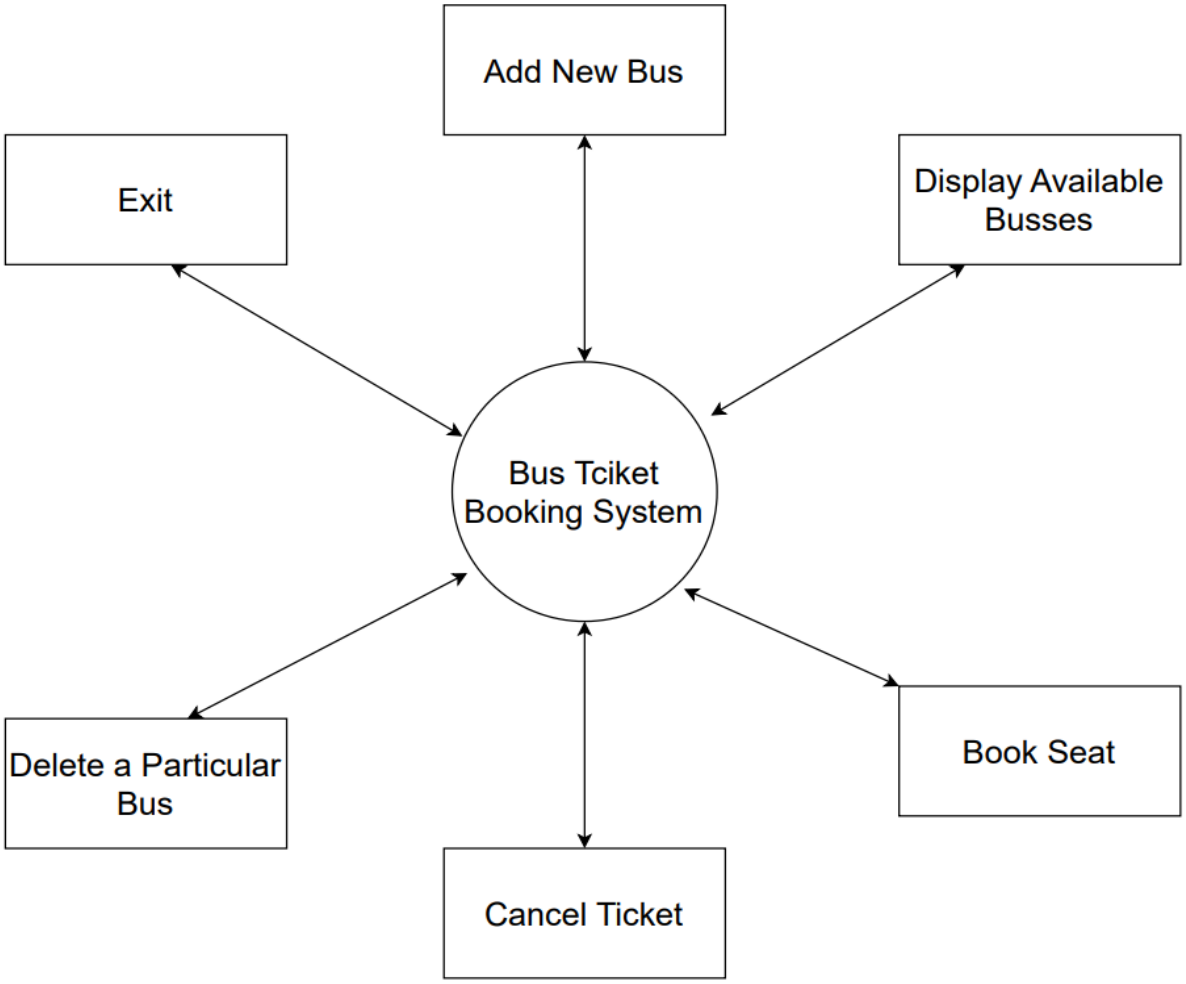
Book seat: This module allows users to book seats online. It requires input of details such as the desired bus, the number of seats to book, and the passenger details.

Cancel seat: This module allows users to cancel their bookings if required. It requires input of details such as the booking ID and the passenger details.

Delete/Update a particular bus details: This module allows the administrator to delete or update the details of a particular bus in the system. It requires input of the bus number and the desired action.

Conclusion

The electronic bus ticket generator project was successfully completed, and all modules were implemented as per the requirements. The system provides a user-friendly interface that allows users to view available buses, book seats, cancel bookings, and view information on buses available for a particular route with their fares. The project can be improved by adding features such as payment integration and improving the security of the system. Overall, the project was a success and met the requirements set out at the start.



Zero Level DFD- Bus Ticket Booking Systen

Source Code:

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define MAX_SEATS 40

// Structure to holds seat information
struct seatInfo {
    int number;
    int is_booked;
};

// Structure to hold bus information
struct busInfo {
    int bus_id;
    char origin[20];
    char destination[20];
    int fare;
    int num_seats;
    struct seatInfo seats[MAX_SEATS];
};

// Global to that holds buses
struct busInfo buses;

// Function to add a new bus
void addBus() {
    int i;
    FILE * fp;
    fp = fopen("Record.txt", "a");
    struct busInfo new_bus;
    printf("Enter bus ID: ");
    scanf("%d", & new_bus.bus_id);
    printf("Enter origin: ");
    scanf("%s", new_bus.origin);
    printf("Enter destination: ");
    scanf("%s", new_bus.destination);
    printf("Enter fare: ");
    scanf("%d", & new_bus.fare);
    printf("Enter number of seats: ");
    scanf("%d", & new_bus.num_seats);
    for (i = 0; i < new_bus.num_seats; i++) {
        new_bus.seats[i].number = i + 1;
```

```
        new_bus.seats[i].is_booked = 0;
    }
    buses = new_bus;

    fwrite( & new_bus, sizeof(new_bus), 1, fp);
    fclose(fp);
    printf("Bus added successfully.\n");
}

// Function to display available buses for a route
void displayBuses() {
    FILE * fp;
    fp = fopen("Record.txt", "r");
    printf("Bus ID\tOrigin     \tDestination\tFare\tAvailable Seats\n");
    while (fread( & buses, sizeof(buses), 1, fp))
        printf("%-6d\t%-11s\t%-11s\t%-4d\t%-7d\n", buses.bus_id, buses.origin,
buses.destination, buses.fare, buses.num_seats);
    fclose(fp);
}

// Function to book a seat
void bookSeat() {
    int bus_id, seat_number;
    FILE * fp;
    fp = fopen("Record.txt", "rb+");
    printf("Enter bus ID: ");
    scanf("%d", & bus_id);
    if (availableBus(bus_id) == 0) {
        printf("Bus not found.\n");
    } else {
        printf("Enter seat number: ");
        scanf("%d", & seat_number);

        while (fread( & buses, sizeof(buses), 1, fp)) {
            if (buses.bus_id == bus_id) {
                if (seat_number < 1 || seat_number > buses.num_seats) {
                    printf("Invalid seat number.\n");
                    return;
                }
                if (buses.seats[seat_number - 1].is_booked) {
                    printf("Seat already booked.\n");
                    return;
                }
                buses.seats[seat_number - 1].is_booked = 1;
                printf("Seat %d booked successfully.\n", seat_number);
```

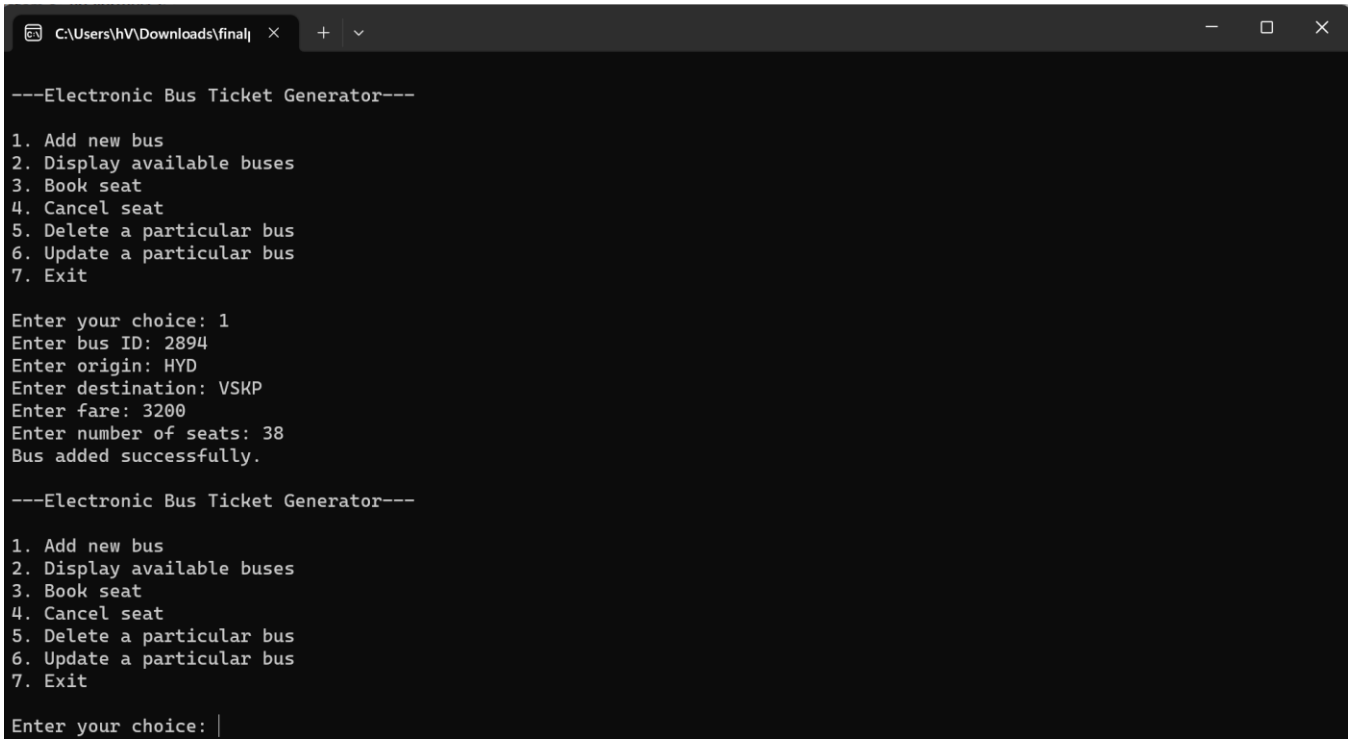
```
                buses.num_seats -= 1;
                fseek(fp, -sizeof(buses), SEEK_CUR);
                fwrite( & buses, sizeof(buses), 1, fp);
                fclose(fp);

                return;
            }
        }
    }

}

// Function to cancel a seat
void cancelSeat() {
    int bus_id, seat_number;

    FILE * fp;
    fp = fopen("Record.txt", "rb+");
    printf("Enter bus ID: ");
    scanf("%d", & bus_id);
    if (availableBus(bus_id) == 0) {
        printf("Bus not found.\n");
    } else {

        printf("Enter seat number: ");
        scanf("%d", & seat_number);
        while (fread( & buses, sizeof(buses), 1, fp) == 1) {
            if (buses.bus_id == bus_id) {
                if (seat_number < 1 || seat_number > buses.num_seats) {
                    printf("Invalid Seat Number.\n");
                    return;
                }
                if (!buses.seats[seat_number - 1].is_booked) {
                    printf("Seat not bookes.\n");
                    return;
                }
                buses.seats[seat_number - 1].is_booked = 0;
                printf("Seat %d cancelled successfully.\n", seat_number);
                buses.num_seats += 1;
                fseek(fp, -sizeof(buses), SEEK_CUR);
                fwrite( & buses, sizeof(buses), 1, fp);
                fclose(fp);
                return;
            }
        }
```

```
    }
}

// Function to check th availability of bus
int availableBus(int bus_id) {
    FILE * fp;
    int c = 0;
    fp = fopen("Record.txt", "r");
    while (!feof(fp)) {
        fread( & buses, sizeof(buses), 1, fp);
        if (bus_id == buses.bus_id) {
            fclose(fp);
            return 1;
        }
    }
    fclose(fp);
    return 0;
}

// Function to delete a bus
void deleteBus() {
    FILE * fpo;
    FILE * fpt;
    int s, bus_id;
    printf("Enter bus ID: ");
    scanf("%d", & bus_id);
    if (availableBus(bus_id) == 0) {
        printf("Bus is not available.\n");
    } else {
        fpo = fopen("Record.txt", "r");
        fpt = fopen("Tempp", "w");
        while (fread( & buses, sizeof(buses), 1, fpo)) {
            s = buses.bus_id;
            if (s != bus_id)
                fwrite( & buses, sizeof(buses), 1, fpt);
        }
        fclose(fpo);
        fclose(fpt);
        fpo = fopen("Record.txt", "w");
        fpt = fopen("Tempp", "r");
        while (fread( & buses, sizeof(buses), 1, fpt))
            fwrite( & buses, sizeof(buses), 1, fpo);
        printf("\nBus Deleted.\n");
        fclose(fpo);
```

```
        fclose(fpt);

    }
}

// Function to update a bus
void updateBus() {
    int bus_id;
    FILE * fp;
    fp = fopen("Record.txt", "rb+");
    printf("Enter bus ID: ");
    scanf("%d", & bus_id);
    if (availableBus(bus_id) == 0) {
        printf("Bus not found.\n");
    } else {
        while (fread( & buses, sizeof(buses), 1, fp) == 1) {
            if (buses.bus_id == bus_id) {
                printf("Enter new origin: ");
                scanf("%s", buses.origin);
                printf("Enter new destination: ");
                scanf("%s", buses.destination);
                printf("Enter new fare: ");
                scanf("%d", & buses.fare);
                fseek(fp, -sizeof(buses), SEEK_CUR);
                fwrite( & buses, sizeof(buses), 1, fp);
                printf("Bus with ID %d updated successfully.\n", bus_id);
                fclose(fp);
                return;
            }
        }
    }
}

// Main function
int main() {

    int choice;
    do {
        printf("\n---Electronic Bus Ticket Generator---\n\n");
        printf("1. Add new bus\n");
        printf("2. Display available buses\n");
        printf("3. Book seat\n");
        printf("4. Cancel seat\n");
        printf("5. Delete a particular bus\n");
        printf("6. Update a particular bus\n");
```

```
        printf("7. Exit\n");
        printf("\nEnter your choice: ");
        scanf("%d", & choice);
        switch (choice) {
        case 1:
            addBus();
            break;
        case 2:
            displayBuses();
            break;
        case 3:
            bookSeat();
            break;
        case 4:
            cancelSeat();
            break;
        case 5:
            deleteBus();
            break;
        case 6:
            updateBus();
            break;
        case 7:
            printf("\nExiting...\n");
            break;

        default:
            printf("\nInvalid choice. Please try again\n");
            break;
        }
    }
    while (choice != 7);
    return 0;
}
```

Output:



THANK YOU