

***LOVELY PROFESSIONAL UNIVERSITY***

# SECTION K22EU

**FINAL REPORT**

**Topic : *Electronic Bus Ticket Generator Submitted to :***  ***Simarjit Singh Malhi* Submitted By:**

|  |  |  |  |
| --- | --- | --- | --- |
| **1.** | **Ishan jaiswal** | **57** | **12210079** |
| **2.** | **Souvik sinha mahapatra** | **56** | **12210074** |
| **3.** | **Harsha vardhan kosuru** | **55** | **12210058** |

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of

many individuals. We would like to extend our sincere gratitude to all of them.

We are highly indebted to, Professor ***Simarjit Singh Malhi*** , Lovely Professional University, for the guidance and constant supervision as well as for providing necessary information

regarding the project & also for the support in completing the project.

We would also like to thank all the people who kept aside a part of their precious time for the help and guidance.

**Title:- Electronic Bus Ticket Generator.**

* **Introduction:-** The Electronic Bus Ticket Generator is a software application that generates electronic tickets for bus passengers.The system replaces the traditional paper-based ticketing system with an electronic one, enabling passengers to purchase tickets online or through mobile applications.

The system also provides features such as real-time bus

tracking, seat selection, and fare calculation. In this project, we aim to develop an Electronic Bus Ticket Generator using C programming language.

# Objectives:

* To develop an electronic ticketing system for buses that is user-friendly and efficient.
* To provide passengers with the option to purchase tickets online or through mobile applications.
* To provide seat selection features.
* To ensure data security and prevent unauthorized access to the system.

# Modules:

This program is an Electronic Bus Ticket Generator that allows the user to perform various operations related to bus

services, such as displaying available buses for a given route, adding a new bus, booking a seat on a bus, canceling a booked seat, deleting a particular bus, and updating the details of a particular bus. The program is menu-driven,

where the user can choose an operation to perform from a list of options.

The program uses a struct to represent a bus, which includes an ID, route name, fare, and an array of seats. The maximum number of buses is defined as 10, and the maximum number of seats is defined as 20. The program initializes some initial data to populate the bus array.

The program is organized into several ***modules***, which are:

**main()** function: This is the main module of the program that calls other functions and implements the menu-driven loop.

**displayBuses()** function: This function takes an array of Bus structures, the count of buses, and a route name as

arguments. It displays all the available buses for the given route.

**addBus()** function: This function takes an array of Bus

structures and a pointer to the count of buses as arguments. It prompts the user to enter the details of a new bus and adds it to the array.

**bookSeat()** function: This function takes a pointer to a Bus

structure as an argument. It prompts the user to enter a seat number and books the seat if it is available.

**cancelSeat()** function: This function takes a pointer to a Bus structure as an argument. It prompts the user to enter a seat

number and cancels the booking if the seat is already booked.

**deleteBus()** function: This function takes an array of Bus structures, a pointer to the count of buses, and a bus ID as arguments. It deletes the bus with the given ID from the array.

**updateBus()** function: This function takes an array of Bus

structures, the count of buses, and a bus ID as arguments. It prompts the user to enter the new details of the bus with the given ID and updates them.

Each of these modules has a specific functionality, and they work together to provide the desired functionality of the program.

# Code:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

// Define structure for bus details

struct Bus {

int busID;

char route[50];

int seats;

float fare;

};

// Function to display available buses for a route

void displayBuses() {

FILE \*file;

struct Bus bus;

file = fopen("buses.dat", "rb");

if (file == NULL) {

printf("Error opening file.\n");

return;

}

printf("Available Buses:\n");

printf("%-10s%-30s%-10s%-10s\n", "Bus ID", "Route", "Seats", "Fare");

printf("-----------------------------------------------------\n");

while (fread(&bus, sizeof(struct Bus), 1, file) == 1) {

printf("%-10d%-30s%-10d%-10.2f\n", bus.busID, bus.route, bus.seats, bus.fare);

}

fclose(file);

}

// Function to add a new bus

void addBus() {

FILE \*file;

struct Bus bus;

file = fopen("buses.dat", "ab");

if (file == NULL) {

printf("Error opening file.\n");

return;

}

printf("Enter Bus ID: ");

scanf("%d", &bus.busID);

printf("Enter Route: ");

scanf(" %[^\n]s", bus.route);

printf("Enter Number of Seats: ");

scanf("%d", &bus.seats);

printf("Enter Fare: ");

scanf("%f", &bus.fare);

fwrite(&bus, sizeof(struct Bus), 1, file);

printf("Bus added successfully.\n");

fclose(file);

}

// Function to book a seat

void bookSeat() {

FILE \*file;

struct Bus bus;

int busID;

int numSeats;

file = fopen("buses.dat", "r+b");

if (file == NULL) {

printf("Error opening file.\n");

return;

}

printf("Enter Bus ID: ");

scanf("%d", &busID);

while (fread(&bus, sizeof(struct Bus), 1, file) == 1) {

if (bus.busID == busID) {

printf("Enter Number of Seats to Book: ");

scanf("%d", &numSeats);

if (bus.seats >= numSeats) {

bus.seats -= numSeats;

fseek(file, -sizeof(struct Bus), SEEK\_CUR);

fwrite(&bus, sizeof(struct Bus), 1, file);

printf("%d seat(s) booked successfully.\n", numSeats);

} else {

printf("Seats not available.\n");

}

fclose(file);

return;

}

}

printf("Bus ID not found.\n");

fclose(file);

}

// Function to cancel a seat

void cancelSeat() {

FILE \*file;

struct Bus bus;

int busID;

int numSeats;

file = fopen("buses.dat", "r+b");

if (file == NULL) {

printf("Error opening file.\n");

return;

}

printf("Enter Bus ID: ");

scanf("%d", &busID);

while (fread(&bus, sizeof(struct Bus), 1, file) == 1) {

if (bus.busID == busID) {

printf("Enter Number of Seats to Cancel: ");

scanf("%d", &numSeats);

if ((bus.seats + numSeats) <= 50) {

bus.seats += numSeats;

fseek(file, -sizeof(struct Bus), SEEK\_CUR);

fwrite(&bus, sizeof(struct Bus), 1, file);

printf("%d seat(s) cancelled successfully.\n", numSeats);

} else {

printf("Invalid number of seats to cancel.\n");

}

fclose(file);

return;

}

}

printf("Bus ID not found.\n");

fclose(file);

}

// Function to delete/update a bus

void deleteUpdateBus() {

FILE \*file;

struct Bus bus;

int busID;

int choice;

file = fopen("buses.dat", "r+b");

if (file == NULL) {

printf("Error opening file.\n");

return;

}

printf("Enter Bus ID: ");

scanf("%d", &busID);

while (fread(&bus, sizeof(struct Bus), 1, file) == 1) {

if (bus.busID == busID) {

printf("Bus Details:\n");

printf("%-10s%-30s%-10s%-10s\n", "Bus ID", "Route", "Seats", "Fare");

printf("-----------------------------------------------------\n");

printf("%-10d%-30s%-10d%-10.2f\n", bus.busID, bus.route, bus.seats, bus.fare);

printf("\n1. Delete Bus\n");

printf("2. Update Bus Details\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

fseek(file, -sizeof(struct Bus), SEEK\_CUR);

fwrite("\0", sizeof(struct Bus), 1, file);

printf("Bus deleted successfully.\n");

break;

case 2:

printf("Enter New Route: ");

scanf(" %[^\n]s", bus.route);

printf("Enter New Number of Seats: ");

scanf("%d", &bus.seats);

printf("Enter New Fare: ");

scanf("%f", &bus.fare);

fseek(file, -sizeof(struct Bus), SEEK\_CUR);

fwrite(&bus, sizeof(struct Bus), 1, file);

printf("Bus details updated successfully.\n");

break;

default:

printf("Invalid choice.\n");

break;

}

fclose(file);

return;

}

}

printf("Bus ID not found.\n");

fclose(file);

}

int main() {

int choice;

while (1) {

printf("\nElectronic Bus Ticket Generator\n");

printf("1. Display Available Buses\n");

printf("2. Add New Bus\n");

printf("3. Book Seat\n");

printf("4. Cancel Seat\n");

printf("5. Delete/Update Bus Details\n");

printf("6. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

displayBuses();

break;

case 2:

addBus();

break;

case 3:

bookSeat();

break;

case 4:

cancelSeat();

break;

case 5:

deleteUpdateBus();

break;

case 6:

printf("Thank you for using the Electronic Bus Ticket Generator.\n");

exit(0);

default:

printf("Invalid choice. Please try again.\n");

break;

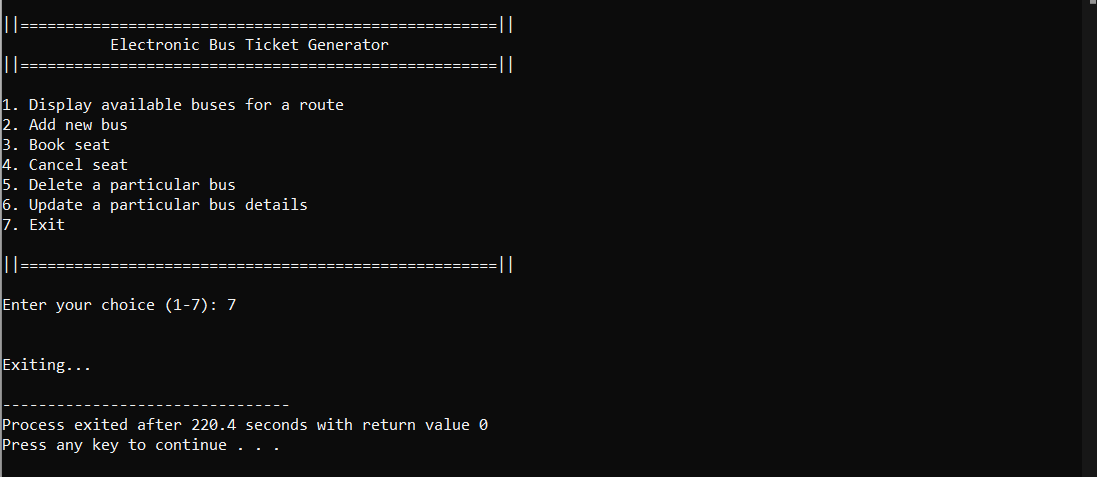
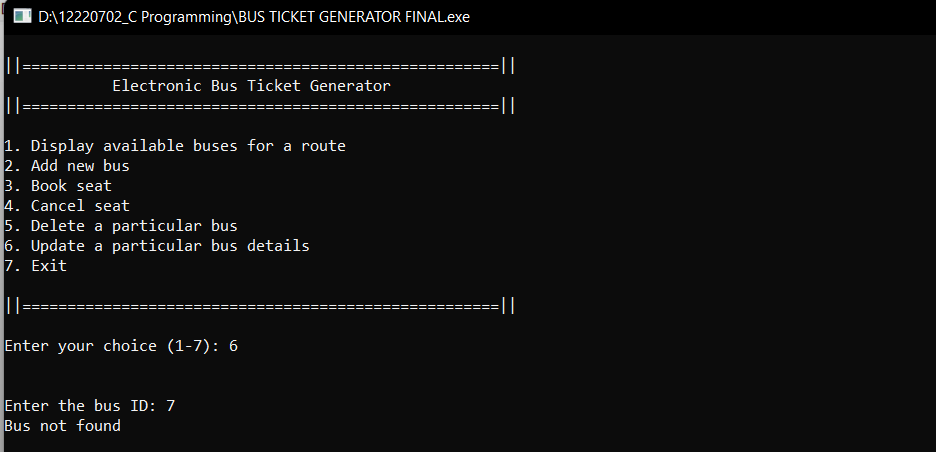
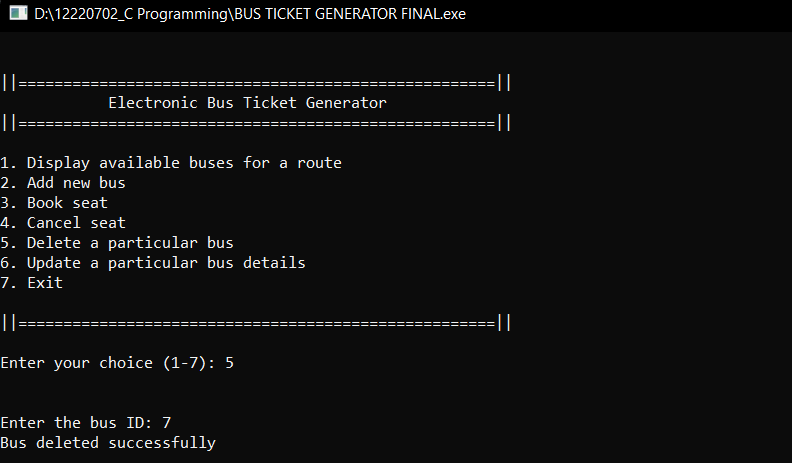
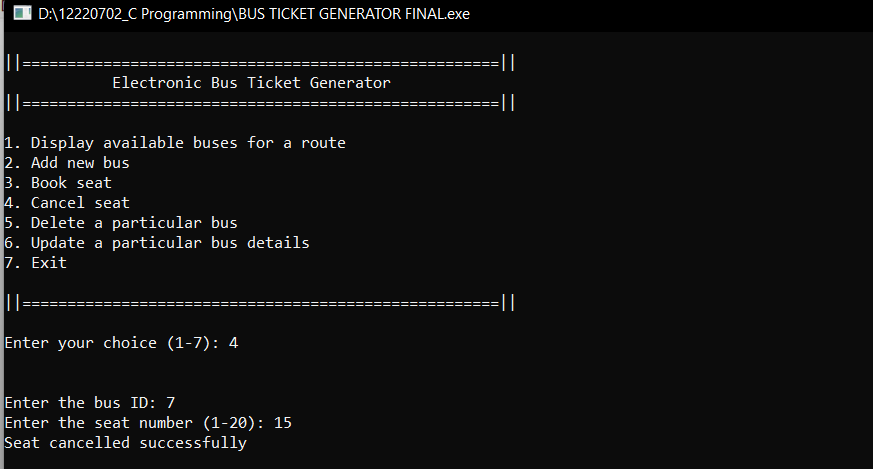
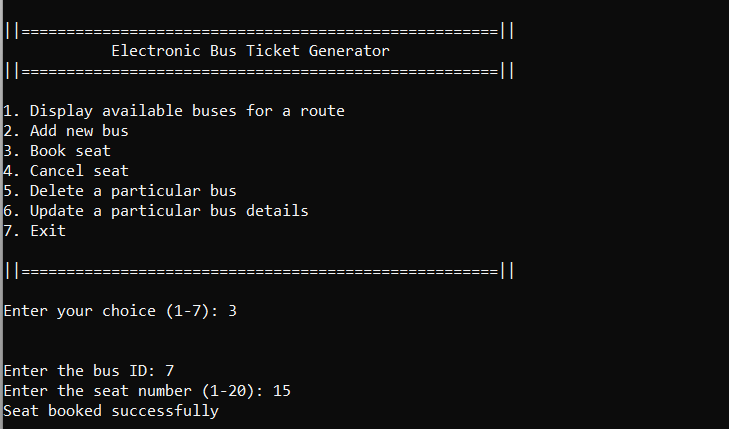
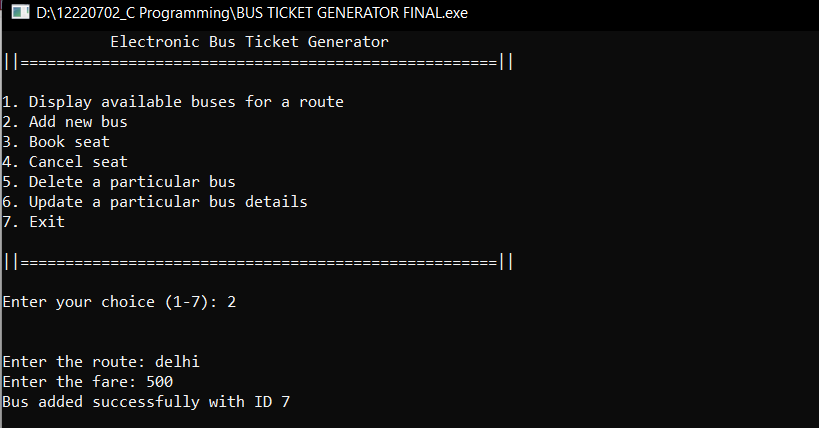
}

}

return 0;

}

333333



* **Conclusion:** This code project is an Electronic Bus Ticket Generator program implemented in C. The program allows users to display available buses for a given route, add new buses, book seats, cancel seats, delete a particular bus, and update a particular bus.

The program is implemented using a menu-driven program loop that takes user input and performs various actions based on the choice selected. It makes use of a Bus structure that contains information about each bus, such as its ID, route, fare, and available seats.

The code is well-structured and modular, with each function performing a specific task. The functions use appropriate parameters and return types, and variable names are descriptive, making the code easy to understand and modify.

Overall, this code project is a good example of how to

implement a basic electronic ticketing system in C, and could be extended or modified for use in a real-world scenario.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*