

C PROJECT

BUS TICKET RESERVATIONSYSTEM

BATCH – 68

B.TECH CSE

TEAM MEMBERS –

DEVANSH JOSHI -590028189

KUSHAL GUPTA - 590028146

1. Problem definition :-

Create a console-based Bus Reservation System in C that allows users to:

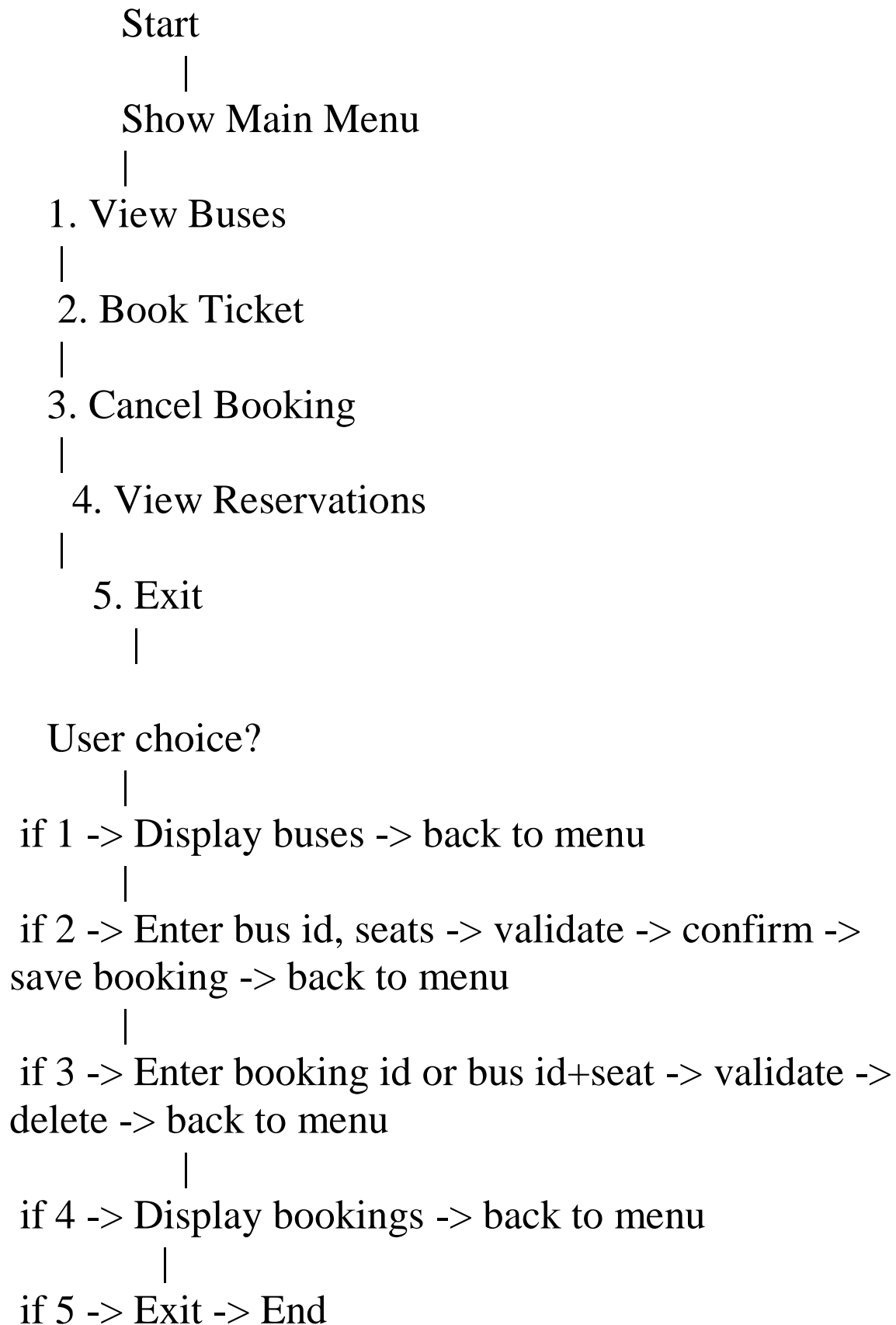
- View available buses and their details (bus id, route, date, time, total seats).
- Book a ticket for a given bus (reserve seat(s)).
- Cancel a booking.
- View all current reservations.

Requirements & constraints:

- Use C and simple file I/O to persist bookings between runs.
- Use structures to represent buses and bookings.

- Provide a simple menu-driven UI in the terminal.
- Basic validation (seat availability, valid bus id).

2. Flow chart :-



3. Algorithm :-

- Start program.
- Load buses and bookings from files (if files exist); else initialize default buses.
- Show menu and read user choice.
- For View Buses: print list of buses and seats available.
- For Book Ticket:
 - * Prompt for bus id.
 - * Check existence and seats available.
 - * Collect passenger name and number of seats.
 - * Assign booking id, update bus available seats and write booking to file.
- For Cancel Booking:
 - * Prompt for booking id.
 - * Search booking, remove it, update bus seats, update files.
- For View Reservations: list bookings.
- Loop until Exit.
- Save data and exit.

4. Problem faced by group :-

- * Handling concurrent changes (concurrency not implemented in this simple project).
- * Designing a simple persistence format (we used text files with simple parsing).
- * Ensuring IDs remain unique across runs — solved using an incremental counter saved to file.
- * Input validation and defensive programming to prevent crashes on bad input.

5. Code :-

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
```

```
#define MAX_BUSES 100
```

```
#define MAX_SEATS 250
#define MAX_NAME 50
#define RESERVATION_FILE
"reservations.dat"
```

```
typedef struct {
    int bus_id;
    char source[30];
    char destination[30];
    int total_seats;
    int available_seats;
    float fare;
} Bus;
```

```
typedef struct {
    char
passenger_name[MAX_NAME];
    int bus_id;
    int seat_number;
```

```
    int reservation_id;  
} Reservation;
```

```
Bus buses[MAX_BUSES];  
Reservation  
reservations[MAX_SEATS *  
MAX_BUSES];  
int reservation_count = 0;  
int next_reservation_id =  
1001;
```

```
void initialize_buses() {  
    int bus_data[][6] = {  
        {101, 30, 500},  
        {102, 35, 50},  
        {103, 40, 60},  
        {104, 32, 55},  
        {105, 38, 80}  
    };  
};
```



```
    char* sources[] = {"New  
Delhi", "Bhiwani",  
"Chandigarh", "Lucknow",  
"Manali"};  
    char* destinations[] =  
{"lahore", "Pune",  
"Dehradun", "Sonipat",  
"Amritsar"};  
  
    for(int i = 0; i < 5; i++) {  
        buses[i].bus_id =  
bus_data[i][0];  
        buses[i].total_seats =  
bus_data[i][1];  
        buses[i].available_seats  
= bus_data[i][1];  
        buses[i].fare =  
bus_data[i][2];  
        strcpy(buses[i].source,  
sources[i]);
```

```
strcpy(buses[i].destination,  
destinations[i]);  
    }  
}
```

```
void display_buses() {  
    printf("\n AVAILABLE  
BUSES \n");  
    printf("Bus  
ID\nSource\t\nDestination\nA  
vailable Seats\nFare\n");  
    printf("\n");
```

```
    for(int i = 0; i < 5; i++) {  
        if(buses[i].bus_id != 0) {  
            printf("%d\t%-12s\t%-  
12s\t%d\t\t\t$%.2f\n",  
                    buses[i].bus_id,  
                    buses[i].source,
```

```
buses[i].destination,  
  
buses[i].available_seats,  
        buses[i].fare);  
    }  
}  
printf("\n");  
}
```

```
int find_bus(int bus_id) {  
    for(int i = 0; i <  
MAX_BUSES; i++) {  
        if(buses[i].bus_id ==  
bus_id) {  
            return i;  
        }  
    }  
    return -1;  
}
```

```
void book_ticket() {  
    int bus_id, bus_index;  
    char name[MAX_NAME];  
  
    display_buses();  
  
    printf("Enter Bus ID: ");  
    scanf("%d", &bus_id);  
  
    bus_index =  
    find_bus(bus_id);  
    if(bus_index == -1) {  
        printf("Invalid Bus  
ID!\n");  
        return;  
    }  
}
```

```
if(buses[bus_index].available  
_seats <= 0) {  
    printf("Sorry, no seats  
available on this bus!\n");  
    return;  
}
```

```
    printf("Enter Passenger  
Name: ");  
    getchar();  
    fgets(name, MAX_NAME,  
stdin);  
    name[strcspn(name, "\n")]  
= 0;
```

```
Reservation new_res;
```

```
strcpy(new_res.passenger_n  
ame, name);
```

```
        new_res.bus_id = bus_id;
        new_res.seat_number =
buses[bus_index].total_seats
-
buses[bus_index].available_s
eats + 1;
        new_res.reservation_id =
next_reservation_id++;
```

```
reservations[reservation_cou
nt++] = new_res;
```

```
buses[bus_index].available_s
eats--;
```

```
        printf("\n TICKET BOOKED
SUCCESSFULLY \n");
```

```
    printf("Reservation ID:
%d\n",
new_res.reservation_id);
    printf("Passenger: %s\n",
new_res.passenger_name);
    printf("Bus ID: %d\n",
new_res.bus_id);
    printf("Seat Number:
%d\n",
new_res.seat_number);
    printf("Total Fare: $%.2f\n",
buses[bus_index].fare);
}
```

```
void cancel_ticket() {
    int reservation_id, found =
0;
```

```
    printf("Enter Reservation
ID to cancel: ");
```

```
scanf("%d",  
&reservation_id);
```

```
for(int i = 0; i <  
reservation_count; i++) {
```

```
if(reservations[i].reservation_i  
d == reservation_id) {  
    found = 1;  
    int bus_index =  
find_bus(reservations[i].bus_i  
d);
```

```
    printf("\n  
RESERVATION FOUND \n");  
    printf("Reservation ID:  
%d\n",  
reservations[i].reservation_id)  
;
```



```
        printf("Passenger:
%s\n",
reservations[i].passenger_name);
```

```
        printf("Bus ID: %d\n",
reservations[i].bus_id);
```

```
        printf("Seat Number:
%d\n",
reservations[i].seat_number);
```

```
        char confirm;
        printf("\nAre you sure
you want to cancel? (y/n): ");
        scanf(" %c",
&confirm);
```

```
        if(tolower(confirm) ==
'y') {
```

```
buses[bus_index].available_seats++;
```

```
        for(int j = i; j < reservation_count - 1; j++) {  
            reservations[j] = reservations[j + 1];  
        }  
        reservation_count--  
;
```

```
        printf("Reservation  
cancelled successfully!\n");  
    } else {  
        printf("Cancellation  
aborted.\n");  
    }  
    break;  
}
```

```

    }

    if(!found) {
        printf("Reservation ID
not found!\n");
    }
}

void view_all_reservations() {
    if(reservation_count == 0) {
        printf("No reservations
found!\n");
        return;
    }

    printf("\n=== ALL
RESERVATIONS ===\n");
    printf("Res ID\tPassenger
Name\t\tBus ID\tSeat No\n");

```

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

```
        for(int i = 0; i <  
reservation_count; i++) {  
            printf("%d\t%-  
20s\t%d\t%d\n",  
  
reservations[i].reservation_id,  
  
reservations[i].passenger_na  
me,  
  
reservations[i].bus_id,  
  
reservations[i].seat_number);  
        }  
    }
```

```
void save_reservations() {
```

```
FILE *file =  
fopen(RESERVATION_FILE,  
"wb");  
if(file == NULL) {  
    printf("Error saving  
reservations!\n");  
    return;  
}
```

```
    fwrite(&reservation_count,  
sizeof(int), 1, file);
```

```
    fwrite(&next_reservation_id,  
sizeof(int), 1, file);
```

```
    fwrite(reservations,  
sizeof(Reservation),  
reservation_count, file);
```

```
    fclose(file);  
}
```

```
void load_reservations() {  
    FILE *file =  
    fopen(RESERVATION_FILE,  
    "rb");  
    if(file == NULL) {  
        return;  
    }  
  
    fread(&reservation_count,  
    sizeof(int), 1, file);  
  
    fread(&next_reservation_id,  
    sizeof(int), 1, file);  
    fread(reservations,  
    sizeof(Reservation),  
    reservation_count, file);  
  
    fclose(file);  
}
```

```
int main() {
    int choice;

    initialize_buses();
    load_reservations();

    printf(" BUS
RESERVATION SYSTEM
\n");

    do {
        printf("\nMain Menu:\n");
        printf("1. View Available
Buses\n");
        printf("2. Book Ticket\n");
        printf("3. Cancel
Ticket\n");
        printf("4. View All
Reservations\n");
```

```
printf("5. Exit\n");  
printf("Enter your choice:  
");  
scanf("%d", &choice);
```

```
switch(choice) {  
    case 1:  
        display_buses();  
        break;  
    case 2:  
        book_ticket();  
        break;  
    case 3:  
        cancel_ticket();  
        break;  
    case 4:
```

```
view_all_reservations();  
        break;  
    case 5:
```



```
save_reservations();  
    printf("Thank you  
for using Bus Reservation  
System!\n");  
    break;  
    default:  
        printf("Invalid  
choice! Please try again.\n");  
    }  
} while(choice != 5);  
  
return 0;  
}
```

6. SNAP OF CODE :-

```
git > lab > C maincode.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <ctype.h>
5
6  #define MAX_BUSES 100
7  #define MAX_SEATS 250
8  #define MAX_NAME 50
9  #define RESERVATION_FILE "reservations.dat"
10
11 typedef struct {
12     int bus_id;
13     char source[30];
14     char destination[30];
15     int total_seats;
16     int available_seats;
17     float fare;
18 } Bus;
19
20 typedef struct {
21     char passenger_name[MAX_NAME];
22     int bus_id;
23     int seat_number;
24     int reservation_id;
25 } Reservation;
26
27 Bus buses[MAX_BUSES];
28 Reservation reservations[MAX_SEATS * MAX_BUSES];
29 int reservation_count = 0;
30 int next_reservation_id = 1001;
31
32 void initialize_buses() {
33     int bus_data[][6] = {
34         {101, 30, 500},
35         {102, 35, 50},
36         {103, 40, 60},
37         {104, 32, 55},
```

```

38         {105, 38, 80}
39     };
40
41     char* sources[] = {"New Delhi", "Bhiwani", "Chandigarh", "Lucknow", "Manali"};
42     char* destinations[] = {"lahore", "Pune", "Dehradun", "Sonipat", "Amritsar"};
43
44     for(int i = 0; i < 5; i++) {
45         buses[i].bus_id = bus_data[i][0];
46         buses[i].total_seats = bus_data[i][1];
47         buses[i].available_seats = bus_data[i][1];
48         buses[i].fare = bus_data[i][2];
49         strcpy(buses[i].source, sources[i]);
50         strcpy(buses[i].destination, destinations[i]);
51     }
52 }
53
54 void display_buses() {
55     printf("\n AVAILABLE BUSES \n");
56     printf("Bus ID\nSource\t\nDestination\nAvailable Seats\nFare\n");
57     printf("\n");
58
59     for(int i = 0; i < 5; i++) {
60         if(buses[i].bus_id != 0) {
61             printf("%d\t%-12s\t%-12s\t%d\t\t\t%.2f\n",
62                 buses[i].bus_id,
63                 buses[i].source,
64                 buses[i].destination,
65                 buses[i].available_seats,
66                 buses[i].fare);
67         }
68     }
69     printf("\n");
70 }

```

```

71
72 int find_bus(int bus_id) {
73     for(int i = 0; i < MAX_BUSES; i++) {
74         if(buses[i].bus_id == bus_id) {
75             return i;
76         }
77     }
78     return -1;
79 }
80
81 void book_ticket() {
82     int bus_id, bus_index;
83     char name[MAX_NAME];
84
85     display_buses();
86
87     printf("Enter Bus ID: ");
88     scanf("%d", &bus_id);
89
90     bus_index = find_bus(bus_id);
91     if(bus_index == -1) {
92         printf("Invalid Bus ID!\n");
93         return;
94     }
95
96     if(buses[bus_index].available_seats <= 0) {
97         printf("Sorry, no seats available on this bus!\n");
98         return;
99     }
100
101     printf("Enter Passenger Name: ");
102     getchar();
103     fgets(name, MAX_NAME, stdin);
104     name[strcspn(name, "\n")] = 0;
105

```

```

105
106     Reservation new_res;
107     strcpy(new_res.passenger_name, name);
108     new_res.bus_id = bus_id;
109     new_res.seat_number = buses[bus_index].total_seats - buses[bus_index].available_seats + 1;
110     new_res.reservation_id = next_reservation_id++;
111
112     reservations[reservation_count++] = new_res;
113     buses[bus_index].available_seats--;
114
115     printf("\n TICKET BOOKED SUCCESSFULLY \n");
116     printf("Reservation ID: %d\n", new_res.reservation_id);
117     printf("Passenger: %s\n", new_res.passenger_name);
118     printf("Bus ID: %d\n", new_res.bus_id);
119     printf("Seat Number: %d\n", new_res.seat_number);
120     printf("Total Fare: $%.2f\n", buses[bus_index].fare);
121 }
122
123 void cancel_ticket() {
124     int reservation_id, found = 0;
125
126     printf("Enter Reservation ID to cancel: ");
127     scanf("%d", &reservation_id);
128
129     for(int i = 0; i < reservation_count; i++) {
130         if(reservations[i].reservation_id == reservation_id) {
131             found = 1;
132             int bus_index = find_bus(reservations[i].bus_id);
133
134             printf("\n RESERVATION FOUND \n");
135             printf("Reservation ID: %d\n", reservations[i].reservation_id);
136             printf("Passenger: %s\n", reservations[i].passenger_name);
137             printf("Bus ID: %d\n", reservations[i].bus_id);
138             printf("Seat Number: %d\n", reservations[i].seat_number);
139

```

```

69
70     char confirm;
71     printf("\nAre you sure you want to cancel? (y/n): ");
72     scanf(" %c", &confirm);
73
74     if(tolower(confirm) == 'y') {
75
76         buses[bus_index].available_seats++;
77
78         for(int j = i; j < reservation_count - 1; j++) {
79             reservations[j] = reservations[j + 1];
80         }
81         reservation_count--;
82
83         printf("Reservation cancelled successfully!\n");
84     } else {
85         printf("Cancellation aborted.\n");
86     }
87     break;
88 }
89
90 if(!found) {
91     printf("Reservation ID not found!\n");
92 }
93 }
94
95 void view_all_reservations() {
96     if(reservation_count == 0) {
97         printf("No reservations found!\n");
98         return;
99     }
100 }

```

```

printf("\n=== ALL RESERVATIONS ===\n");
printf("Res ID\tPassenger Name\tBus ID\tSeat No\n");
printf("-----\n");

for(int i = 0; i < reservation_count; i++) {
    printf("%d\t%-20s\t%d\t%d\n",
           reservations[i].reservation_id,
           reservations[i].passenger_name,
           reservations[i].bus_id,
           reservations[i].seat_number);
}

}

void save_reservations() {
    FILE *file = fopen(RESERVATION_FILE, "wb");
    if(file == NULL) {
        printf("Error saving reservations!\n");
        return;
    }

    fwrite(&reservation_count, sizeof(int), 1, file);
    fwrite(&next_reservation_id, sizeof(int), 1, file);
    fwrite(reservations, sizeof(Reservation), reservation_count, file);

    fclose(file);
}

void load_reservations() {
    FILE *file = fopen(RESERVATION_FILE, "rb");
    if(file == NULL) {
        return;
    }
}

```

```
204     fread(&reservation_count, sizeof(int), 1, file);
205     fread(&next_reservation_id, sizeof(int), 1, file);
206     fread(reservations, sizeof(Reservation), reservation_count, file);
207
208
209     fclose(file);
210 }
211
212 int main() {
213     int choice;
214
215     initialize_buses();
216     load_reservations();
217
218     printf(" BUS RESERVATION SYSTEM \n");
219
220     do {
221         printf("\nMain Menu:\n");
222         printf("1. View Available Buses\n");
223         printf("2. Book Ticket\n");
224         printf("3. Cancel Ticket\n");
225         printf("4. View All Reservations\n");
226         printf("5. Exit\n");
227         printf("Enter your choice: ");
228         scanf("%d", &choice);
229
230         switch(choice) {
231             case 1:
232                 display_buses();
233                 break;
234             case 2:
235                 book_ticket();
236                 break;
```



```

220     do {
221         printf("\nMain Menu:\n");
222         printf("1. View Available Buses\n");
223         printf("2. Book Ticket\n");
224         printf("3. Cancel Ticket\n");
225         printf("4. View All Reservations\n");
226         printf("5. Exit\n");
227         printf("Enter your choice: ");
228         scanf("%d", &choice);
229
230         switch(choice) {
231             case 1:
232                 display_buses();
233                 break;
234             case 2:
235                 book_ticket();
236                 break;
237             case 3:
238                 cancel_ticket();
239                 break;
240             case 4:
241                 view_all_reservations();
242                 break;
243             case 5:
244                 save_reservations();
245                 printf("Thank you for using Bus Reservation System!\n");
246                 break;
247             default:
248                 printf("Invalid choice! Please try again.\n");
249         }
250     } while(choice != 5);
251
252     return 0;
253 }

```

OUTPUT :-

```
File Edit Selection View Go Run Terminal Help python
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS c:\Users\VICTUS\Desktop\python> cd "c:\Users\VICTUS\Desktop\python\" ; if ($?) { gcc report.c -o report } ; if ($?) { .\report }
BUS RESERVATION SYSTEM

Main Menu:
1. View Available Buses
2. Book Ticket
3. Cancel Ticket
4. View All Reservations
5. Exit
Enter your choice: 2

AVAILABLE BUSES
Bus ID
Source
Destination
Available Seats
Fare
101 New Delhi Lahore 30 $500.00
102 Bhubani Pune 25 $50.00
103 Chandigarh Dehradun 40 $60.00
104 Lucknow Sonapat 32 $75.00
105 Manali Amritsar 38 $80.00

Enter Bus ID: 101
Enter Passenger Name: devansh

TICKET BOOKED SUCCESSFULLY
Reservation ID: 1001
Passenger: devansh
Bus ID: 101
Seat Number: 1
Total Fare: $500.00

Main Menu:
1. View Available Buses
2. Book Ticket
3. Cancel Ticket
4. View All Reservations
5. Exit
Enter your choice: 
```

REPORT.H :-

```
git > lab > C REPORT.H > reservations
1  #ifndef BUS_RESERVATION_H
2  #define BUS_RESERVATION_H
3
4  #include <stdio.h>
5  #include <stdlib.h>
6  #include <string.h>
7  #include <ctype.h>
8
9  #define MAX_BUSES 100
10 #define MAX_SEATS 250
11 #define MAX_NAME 50
12 #define RESERVATION_FILE "reservations.dat"
13
14 typedef struct {
15     int bus_id;
16     char source[30];
17     char destination[30];
18     int total_seats;
19     int available_seats;
20     float fare;
21 } Bus;
22
23 typedef struct {
24     char passenger_name[MAX_NAME];
25     int bus_id;
26     int seat_number;
27     int reservation_id;
28 } Reservation;
29
30 extern Bus buses[MAX_BUSES];
31 extern Reservation reservations[MAX_SEATS * MAX_BUSES];
32 extern int reservation_count;
33 extern int next_reservation_id;
34
35 void initialize_buses();
36 void display_buses();
37 int find_bus(int bus_id);
38 void book_ticket();
39 void cancel_ticket();
40 void view_all_reservations();
41 void save_reservations();
42 void load_reservations();
43
44 #endif
45
```