Jaypee Institute of Information Technology University, Noida

SOFTWARE DEVELOPMENT FUNDAMENTALS LAB-1(24B15CS111)

A Project Based Learning (PBL)

On

MOVEMATE - TRANSPORT MANAGEMENT CONSOLE APPLICATION

Session ODD

Semester (2025-

2026)

SUBMITTED BY:

Arzoo (992501030311)

Shourya Porwal (992501030305)

Sanyam Goyal (992501030310)

Vanshika Soni (992501030487)

SUMITTED TO:

Mrs. Akanksha Mehndiratta

[2] TABLE OF CONTENTS

[3] SUMMARY

Brief overview of the project :

This project is a Bus Management System made in C language. It stores bus details like routes, stops, payment and timings The system also checks the crowd level and gives a final score to each bus. This project helps in understanding structures, arrays, functions, pointers and user input in C.

Objectives :

- To design a basic Bus Management System using C language
- To store essential bus details such as route and stops
- To record occupancy percentage of buses
- To present all bus information in a clear and organized format
- To apply fundamental C programming concepts like structures and functions

[4] INTRODUCTION

Transportation is used daily, and buses play a major role in it. This project presents a management system that stores bus-related and passenger data and displays results in a clear manner. The system provides a basic example of applying C programming to a real-life-style scenario.

Background & Context

Public buses vary in terms of travel time, crowd levels, number of stops, and overall route conditions. A simple computerized system makes it easier to store details of many buses in one place and check important information quickly. This project uses basic C programming ideas such as structures, arrays, functions, and user input to create a straightforward bus information system

Problem Statement

Keeping track of several buses manually can become confusing, especially when each bus has different routes, stops, and crowd levels. There is a need for a basic system that can store bus information and calculate useful values like occupancy percentage .The main problem addressed is to organize bus details in a simple C program and display them in a clear manner

Passengers and operators face issues such as:

- No real-time seat availability.
- No properly displayed route & break details.
- Manual fare calculation errors.
- Lack of central record for bookings.
- No consolidated system for multiple passengers booking at once.

A system was needed that could:

- Show seat availability dynamically
- Store booking records persistently
- Handle multiple passengers
- Calculate fare automatically
- Display stops and break times

Project Objectives

- Develop a complete, functional C-based transport booking system.
- Implement dynamic memory allocation for multiple passengers.
- Use file handling to store persistent bus state & booking history.
- Apply modular programming using functions and structures.
- Demonstrate clean user interface and logical flow.
- Provide seat availability, fare calculation, route details, and via stops.

Scope of the Project

- ➢ Five fixed routes: Delhi → Agra, Jaipur, Kanpur, Varanasi, Chandigarh
- ➢ Persistent seat tracking
- ➢ Via stop display
- ➢ Break time based on distance
- ➢ Multi-passenger booking
- ➢ Payment simulation
- ➢ Ticket-like confirmation output
- ➢ Saving booking records

[5] SYSTEM REQUIREMENTS

- Windows / Linux PC
- GCC compiler
- Text editor (VS Code, Code Blocks, Notepad++)
- Standard C libraries (stdio.h, stdlib.h, math.h, string.h)

Functional Requirements

- Ability to enter bus details
- Display of final bus report

Non-Functional Requirements

- Easy to operate
- Quick responses
- Clean, understandable output
- Error-free basic operations

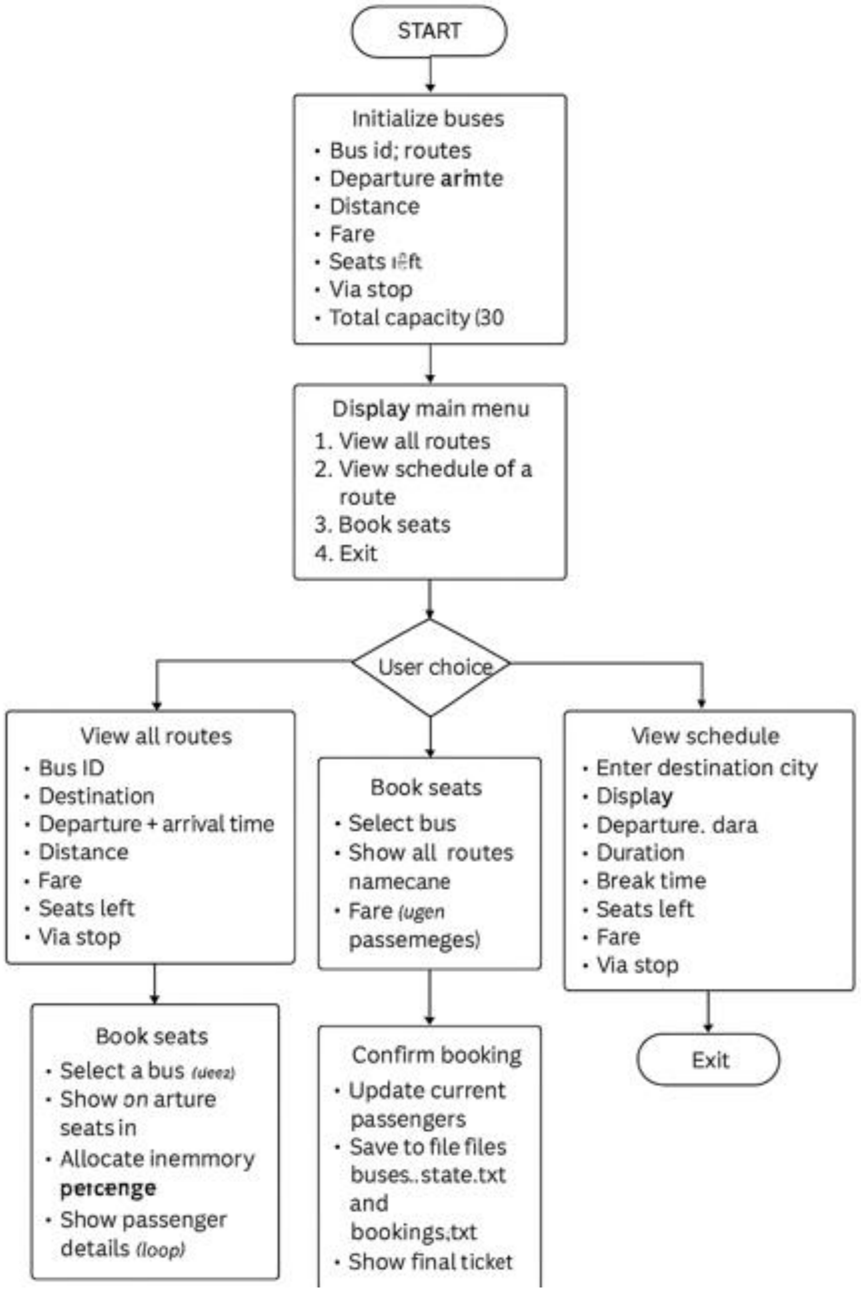
[6] Design and Implementation

SRS Document

Software Requirements Specification

- Users: Students, teachers, small transport operators
- Inputs: Bus number, route, stops, payment modes, ratings
- Processes:
  - Store bus data
  - Compute occupancy percentage
- Outputs:
  - Bus details
  - Final score

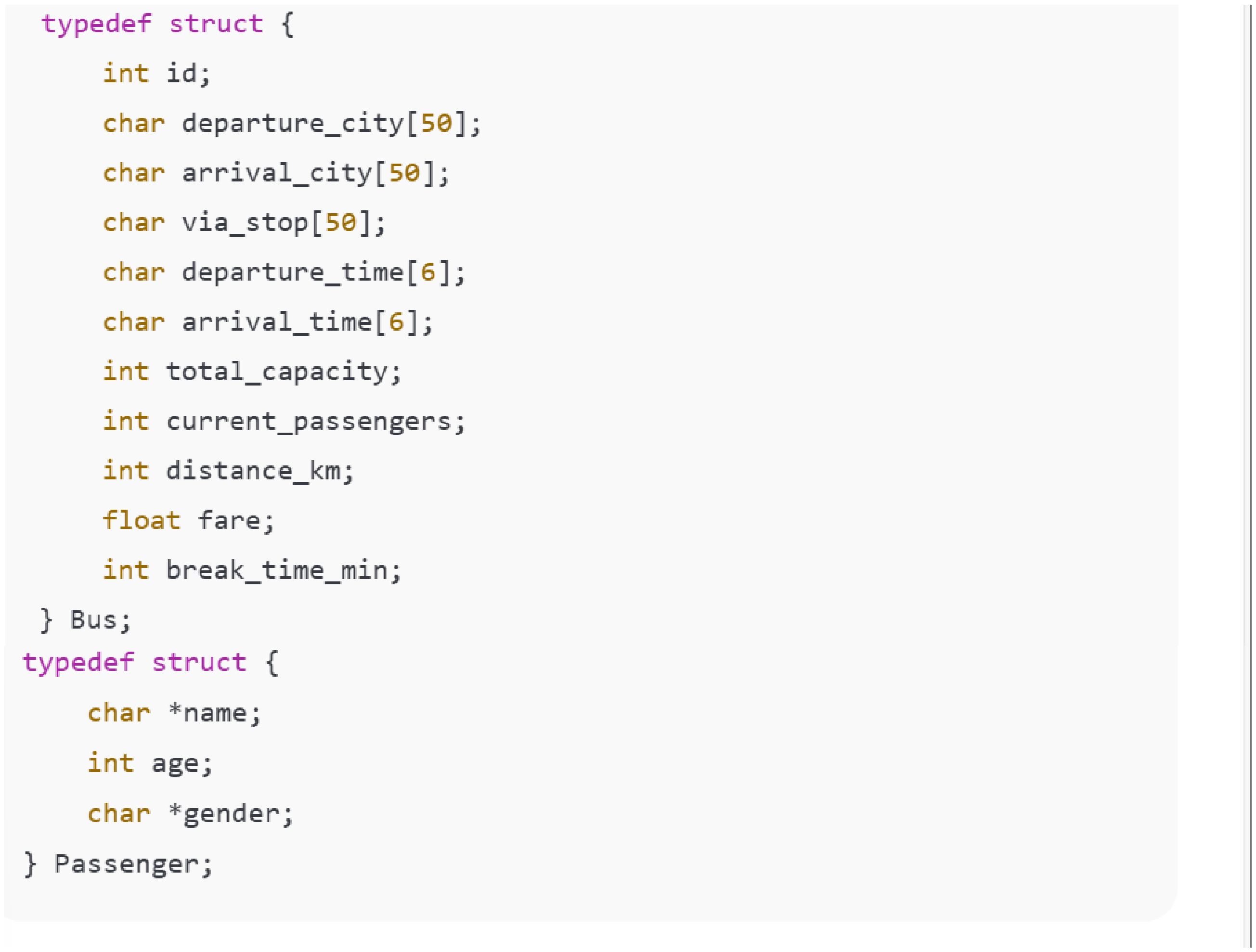
Flowchart



(fig. 1)

Code Snippets Showing Use of C Concepts

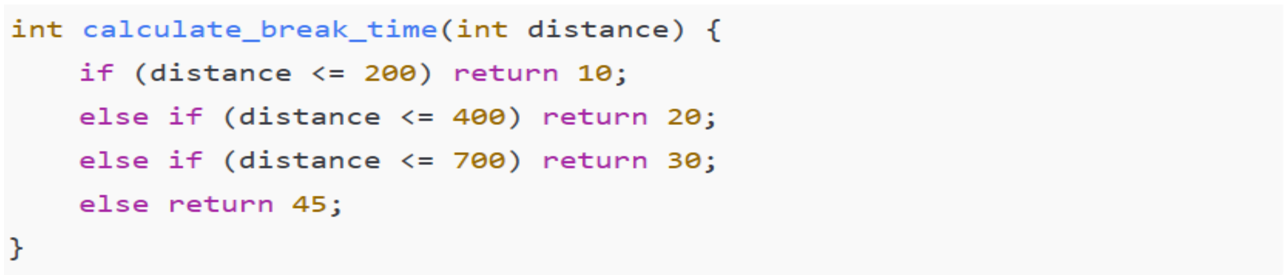
1. Structure (Struct) –



```c
typedef struct {
    int id;
    char departure_city[50];
    char arrival_city[50];
    char via_stop[50];
    char departure_time[6];
    char arrival_time[6];
    int total_capacity;
    int current_passengers;
    int distance_km;
    float fare;
    int break_time_min;
} Bus;
typedef struct {
    char *name;
    int age;
    char *gender;
} Passenger;
```

(fig. 2)

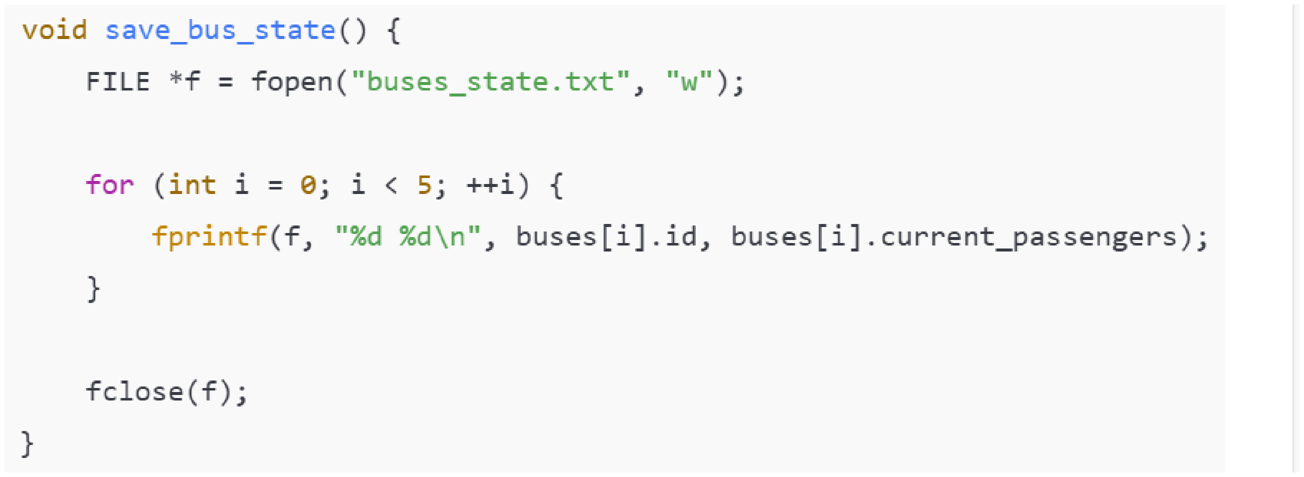
2. Functions –



```c
int calculate_break_time(int distance) {
    if (distance <= 200) return 10;
    else if (distance <= 400) return 20;
    else if (distance <= 700) return 30;
    else return 45;
}
```

(fig. 3)

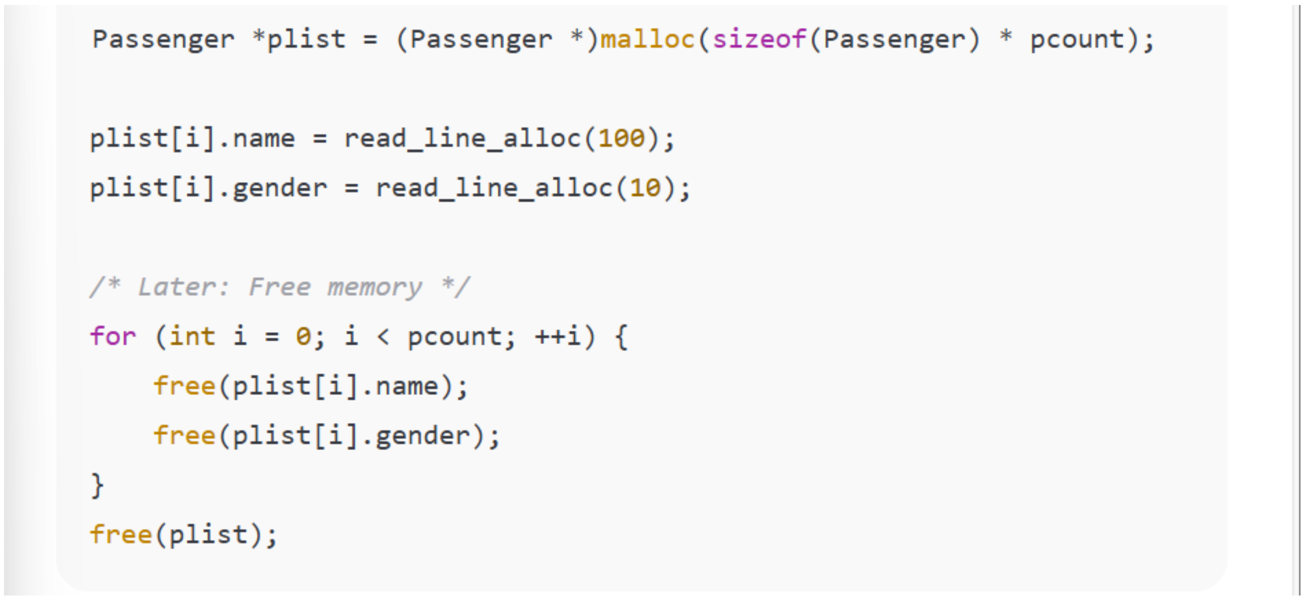
3. File Handling –



```c
void save_bus_state() {
    FILE *f = fopen("buses_state.txt", "w");

    for (int i = 0; i < 5; ++i) {
        fprintf(f, "%d %d\n", buses[i].id, buses[i].current_passengers);
    }

    fclose(f);
}
```

(fig.4)

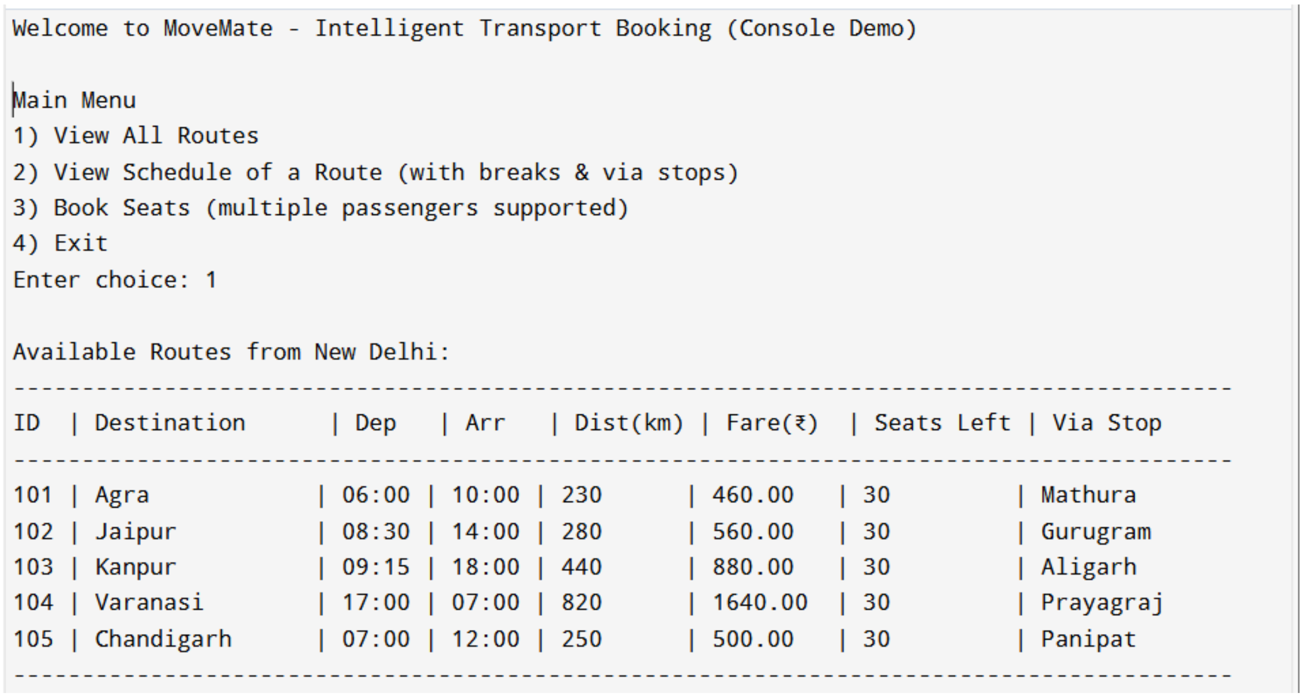
4. Dynamic Memory Allocation (malloc + free) –



```c
Passenger *plist = (Passenger *)malloc(sizeof(Passenger) * pcount);

plist[i].name = read_line_alloc(100);
plist[i].gender = read_line_alloc(10);

/* Later: Free memory */
for (int i = 0; i < pcount; ++i) {
    free(plist[i].name);
    free(plist[i].gender);
}
free(plist);
```

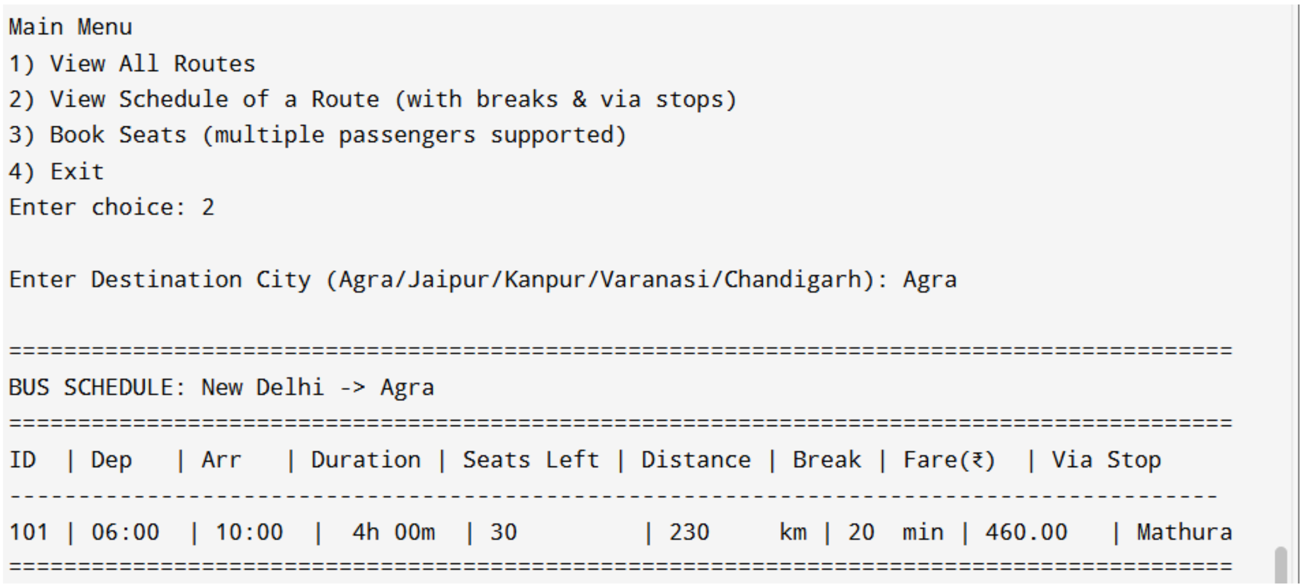
(fig. 5)

Output Screenshots with description



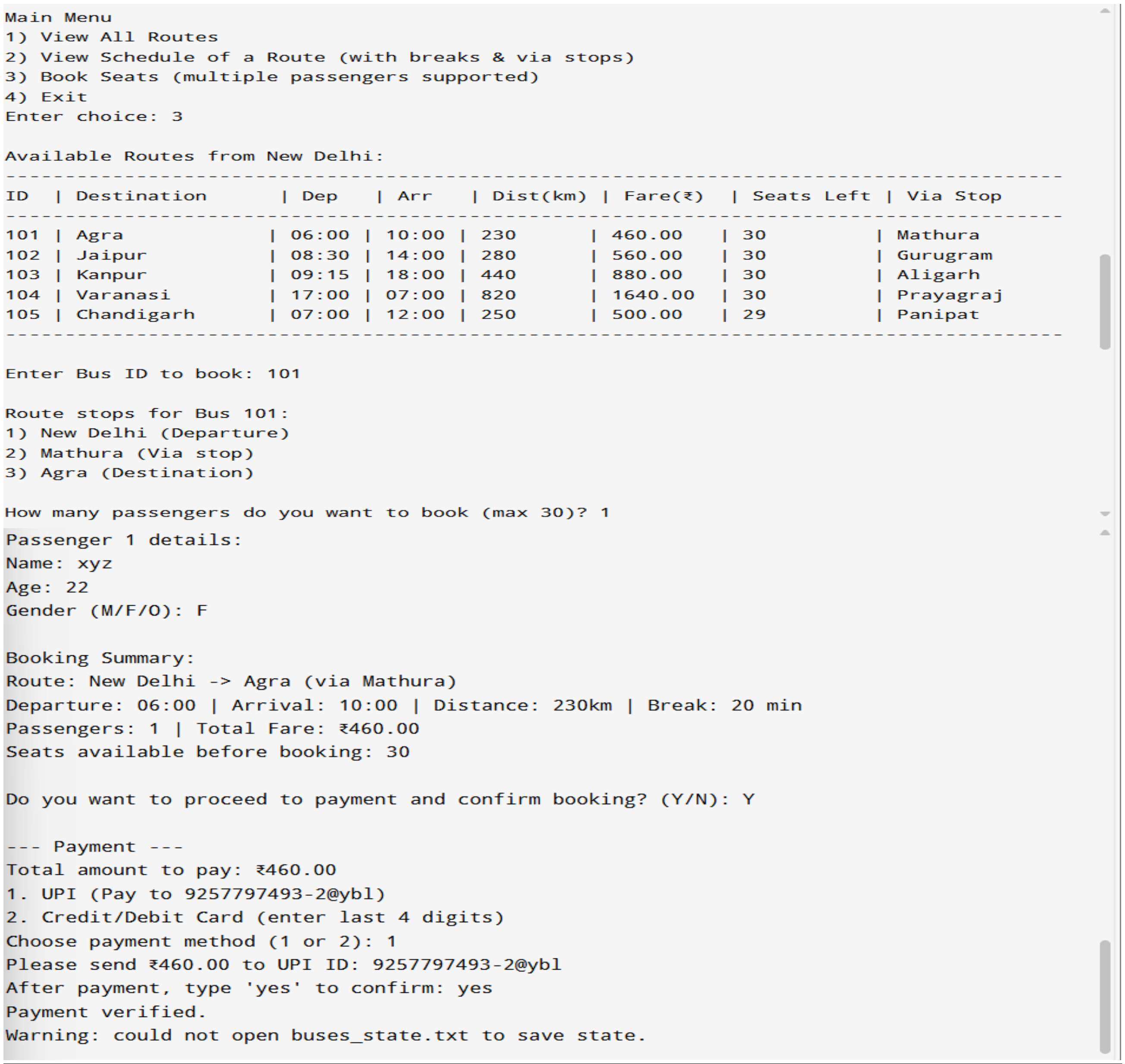
```
Welcome to MoveMate - Intelligent Transport Booking (Console Demo)

Main Menu
1) View All Routes
2) View Schedule of a Route (with breaks & via stops)
3) Book Seats (multiple passengers supported)
4) Exit
Enter choice: 1

Available Routes from New Delhi:
------------------------------------------------------------------------------------------
ID  | Destination      | Dep   | Arr   | Dist(km) | Fare(₹)  | Seats Left | Via Stop
------------------------------------------------------------------------------------------
101 | Agra            | 06:00 | 10:00 | 230      | 460.00   | 30         | Mathura
102 | Jaipur          | 08:30 | 14:00 | 280      | 560.00   | 30         | Gurugram
103 | Kanpur          | 09:15 | 18:00 | 440      | 880.00   | 30         | Aligarh
104 | Varanasi        | 17:00 | 07:00 | 820      | 1640.00  | 30         | Prayagraj
105 | Chandigarh      | 07:00 | 12:00 | 250      | 500.00   | 30         | Panipat
------------------------------------------------------------------------------------------
```

(fig. 6 - Output on choosing 1)

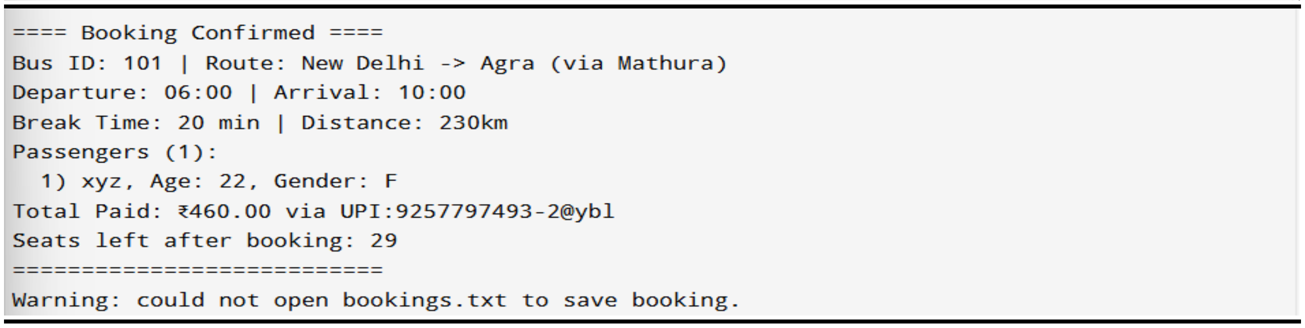


```
Main Menu
1) View All Routes
2) View Schedule of a Route (with breaks & via stops)
3) Book Seats (multiple passengers supported)
4) Exit
Enter choice: 2

Enter Destination City (Agra/Jaipur/Kanpur/Varanasi/Chandigarh): Agra

==========================================================================================
BUS SCHEDULE: New Delhi -> Agra
==========================================================================================
ID  | Dep   | Arr   | Duration | Seats Left | Distance | Break | Fare(₹)  | Via Stop
-----------------------------------------------------------------------------------------
101 | 06:00  | 10:00  |  4h 00m  | 30         | 230     km | 20  min | 460.00   | Mathura
==========================================================================================
```

(fig. 7 - Output on choosing 2)

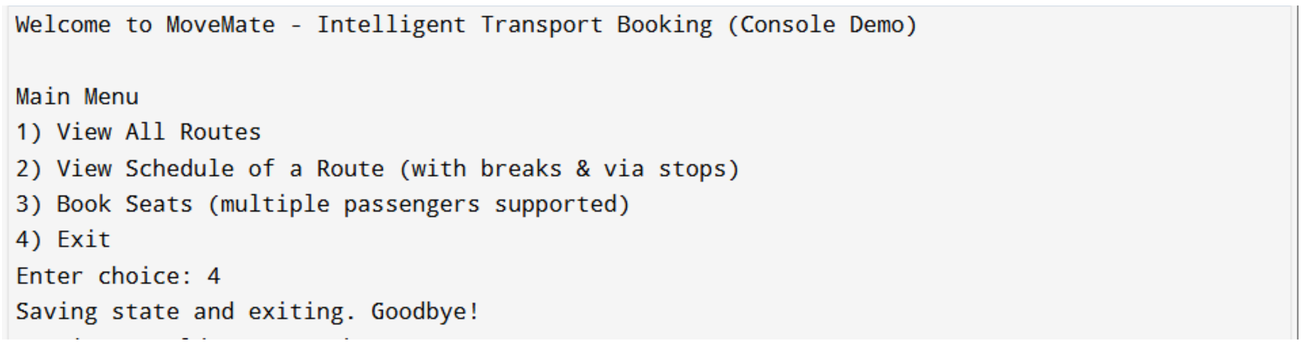


```
Main Menu
1) View All Routes
2) View Schedule of a Route (with breaks & via stops)
3) Book Seats (multiple passengers supported)
4) Exit
Enter choice: 3

Available Routes from New Delhi:
------------------------------------------------------------------------------------------
ID  | Destination       | Dep   | Arr   | Dist(km) | Fare(₹)  | Seats Left | Via Stop
------------------------------------------------------------------------------------------
101 | Agra              | 06:00 | 10:00 | 230      | 460.00   | 30         | Mathura
102 | Jaipur            | 08:30 | 14:00 | 280      | 560.00   | 30         | Gurugram
103 | Kanpur            | 09:15 | 18:00 | 440      | 880.00   | 30         | Aligarh
104 | Varanasi          | 17:00 | 07:00 | 820      | 1640.00  | 30         | Prayagraj
105 | Chandigarh        | 07:00 | 12:00 | 250      | 500.00   | 29         | Panipat
------------------------------------------------------------------------------------------

Enter Bus ID to book: 101

Route stops for Bus 101:
1) New Delhi (Departure)
2) Mathura (Via stop)
3) Agra (Destination)

How many passengers do you want to book (max 30)? 1
Passenger 1 details:
Name: xyz
Age: 22
Gender (M/F/O): F

Booking Summary:
Route: New Delhi -> Agra (via Mathura)
Departure: 06:00 | Arrival: 10:00 | Distance: 230km | Break: 20 min
Passengers: 1 | Total Fare: ₹460.00
Seats available before booking: 30

Do you want to proceed to payment and confirm booking? (Y/N): Y

--- Payment ---
Total amount to pay: ₹460.00
1. UPI (Pay to 9257797493-2@ybl)
2. Credit/Debit Card (enter last 4 digits)
Choose payment method (1 or 2): 1
Please send ₹460.00 to UPI ID: 9257797493-2@ybl
After payment, type 'yes' to confirm: yes
Payment verified.
Warning: could not open buses_state.txt to save state.
```



```
==== Booking Confirmed ====
Bus ID: 101 | Route: New Delhi -> Agra (via Mathura)
Departure: 06:00 | Arrival: 10:00
Break Time: 20 min | Distance: 230km
Passengers (1):
  1) xyz, Age: 22, Gender: F
Total Paid: ₹460.00 via UPI:9257797493-2@ybl
Seats left after booking: 29
===========================
Warning: could not open bookings.txt to save booking.
```

(fig. 8 - Output on choosing 3)



```
Welcome to MoveMate - Intelligent Transport Booking (Console Demo)

Main Menu
1) View All Routes
2) View Schedule of a Route (with breaks & via stops)
3) Book Seats (multiple passengers supported)
4) Exit
Enter choice: 4
Saving state and exiting. Goodbye!
```

(fig. 9 - Output on choosing 4)

[7] Conclusion

Summary & Achievement of Objectives

The system successfully stores bus details, calculates crowd levels, and shows all information on the screen in a simple and readable manner. All project objectives were achieved effectively.

Future Work & Recommendations

Possible improvements include:

- Adding bus timing information
- Adding fare details
- Including live traffic or delay status
- Adding a search or filter option
- Linking the project with a database for long-term storage

[8] References

- School computer science notes
- NCERT / textbook concepts
- Online tutorials for basic C programming