# Project Report on Bus Seat Reservation System

**Submitted by:**

*Name= PUNEET KAUR*   
*UID=24BCA10287*  
*Department Name=UIC*   
*College/University Name= CHANDIGHAR UNIVERSITY*

**Academic Year:**

2024–2025

**1. Introduction**

The **Bus Seat Reservation System** is an automated software application designed to handle the reservations of bus tickets. The system is implemented in **C programming language** and aims to simplify the process of booking and managing bus seat reservations. The manual procedure of reserving seats, checking availability, and canceling tickets is often cumbersome and prone to errors. This system automates these operations and provides an easy-to-use interface to both administrators and passengers.

The project allows the following functionalities:

* **Reserve a seat** for a passenger.
* **Cancel a reservation** and free up a seat.
* **View seat availability** and list of all buses.
* **Store and manage bus schedules** and seat arrangements.

This system serves as a practical implementation of basic **data structures** and **file handling** in C. It also introduces the concept of **modular programming** by breaking the system into functional modules.

**2. Objectives**

The main objectives of the **Bus Seat Reservation System** are as follows:

* **Automation** of seat booking, thereby reducing human error and time consumption.
* To design and implement a **user-friendly interface** that is easy to navigate and interact with.
* To maintain a **detailed record** of all reservations, seat availability, and bus schedules.
* To gain hands-on experience in **C programming** through real-world problem-solving.
* To learn the implementation of **data structures** such as arrays, structures, and functions.
* To integrate basic **file handling** techniques to store, update, and retrieve data.

**3. System Requirements**

**Hardware Requirements:**

* Computer with a minimum of **2GB RAM** and **500MB of available disk space**.
* A standard keyboard and monitor.

**Software Requirements:**

* **Language Used**: C programming
* **Compiler**: GCC (GNU Compiler Collection) or Turbo C++.
* **Operating System**: Windows/Linux
* **IDE**: Code::Blocks, Dev C++, Turbo C (for compiling and running C programs)

**4. Features of the System**

**1. Add New Bus**

The administrator can add details about a new bus, including bus number, driver’s name, arrival and departure times, origin, and destination. This data will be stored for future reference.

**2. View All Buses**

This feature allows the user to view a list of all buses with their respective details (bus number, driver, route, and seat status). The available buses are shown in a tabular format, making it easier for users to select a bus for reservation.

**3. Reserve a Seat**

Users can book a seat on a particular bus by entering the bus number, seat number, and their details. The system will check if the seat is available and reserve it for the user if so.

**4. Cancel Reservation**

The system provides an option to cancel a reservation, freeing up the seat. This requires the bus number and seat number to identify the reservation.

**5. View Available Seats**

This feature displays the current availability of seats for each bus. It shows which seats are booked and which are still open, assisting the user in selecting an available seat.

**5. System Design and Architecture**

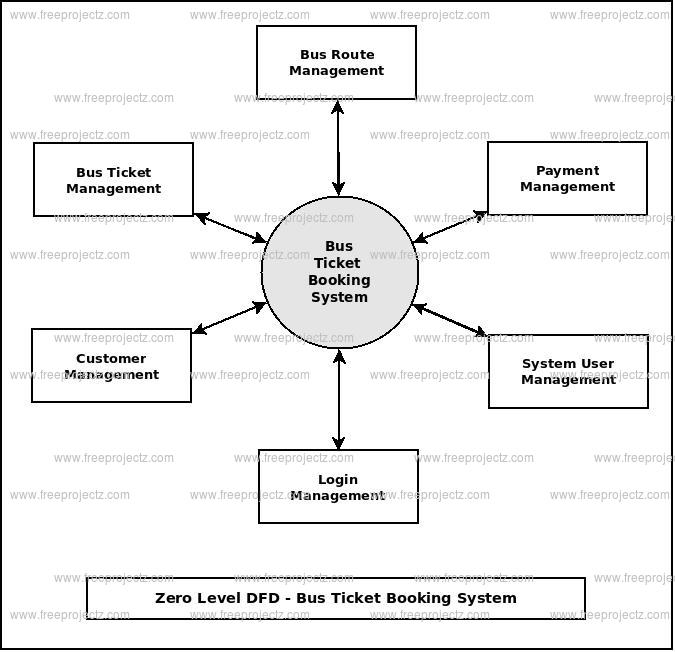
The Bus Seat Reservation System is designed using the **modular approach**, which divides the application into different functional modules. The system uses a menu-driven approach for interaction, where each option corresponds to a particular feature or functionality.

**Modules:**

1. **Bus Management Module**:
   * Allows the admin to add new buses and view bus details.
2. **Reservation Module**:
   * Handles seat reservation and cancellation requests.
3. **Availability Module**:
   * Displays available seats and helps the user select a seat.
4. **Display Module**:
   * Displays all buses and their details, including their seat reservation status.

**6. System Flowchart**

(*Insert a flowchart here that shows how the different modules interact with each other in the system. This flowchart should illustrate the sequence of operations from the main menu to booking, displaying seats, etc.*)



**7. Sample Code Snippets**

Below is a section of the code showing how bus details are entered and displayed, and how seat reservations are handled:

**Bus Structure Definition**

c

CopyEdit

#include <stdio.h>

#include <string.h>

struct Bus {

char bus\_no[10];

char driver[20];

char arrival[10];

char departure[10];

char from[20];

char to[20];

char seats[8][4][10]; // 8 rows, 4 seats per row (seating arrangement)

};

struct Bus bus[10]; // Array to hold data for 10 buses

int bus\_count = 0; // Keeps track of the number of buses

void installBus() {

printf("Enter bus number: ");

scanf("%s", bus[bus\_count].bus\_no);

printf("Enter driver's name: ");

scanf("%s", bus[bus\_count].driver);

printf("Enter arrival time: ");

scanf("%s", bus[bus\_count].arrival);

printf("Enter departure time: ");

scanf("%s", bus[bus\_count].departure);

printf("Enter starting point: ");

scanf("%s", bus[bus\_count].from);

printf("Enter destination: ");

scanf("%s", bus[bus\_count].to);

// Initializing all seats to "Available"

for(int i = 0; i < 8; i++) {

for(int j = 0; j < 4; j++) {

strcpy(bus[bus\_count].seats[i][j], "Available");

}

}

bus\_count++; // Increment bus count after adding a bus

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

}

**Seat Reservation**

c

CopyEdit

void bookSeat() {

char bus\_no[10];

int row, col;

printf("Enter bus number: ");

scanf("%s", bus\_no);

// Find the bus

int bus\_found = 0;

for(int i = 0; i < bus\_count; i++) {

if(strcmp(bus[i].bus\_no, bus\_no) == 0) {

bus\_found = 1;

printf("Bus found! Please select a seat.\n");

printf("Enter row (0-7) and column (0-3): ");

scanf("%d %d", &row, &col);

// Check if the seat is available

if(strcmp(bus[i].seats[row][col], "Available") == 0) {

strcpy(bus[i].seats[row][col], "Booked");

printf("Seat booked successfully!\n");

} else {

printf("Seat is already booked!\n");

}

break;

}

}

if(!bus\_found) {

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

}

}

**8. Output and Screenshots**

The system has various output screens that help the user interact with it. Below are some of the key outputs:

1. **Main Menu Screen**:
   * Allows the user to select an option (Add Bus, Book Seat, etc.).
2. **Bus Information Display**:
   * Lists all buses along with their schedules.
3. **Seat Availability Screen**:
   * Shows the seating arrangement for a selected bus, indicating booked and available seats.

**9. Advantages**

* **Efficiency**: Reduces the time and effort involved in managing reservations manually.
* **Accuracy**: Reduces the chances of errors in seat allocation.
* **User-Friendly**: Easy-to-navigate interface that provides quick access to information.
* **Flexibility**: Can be adapted to different types of transport services, not just buses.
* **Scalability**: The system can be extended to handle more buses and more advanced features like payment integration.

**10. Limitations**

* **Single-user system**: Currently, the system does not support multiple users accessing it at once.
* **No data persistence**: All data is lost once the program is closed. A database or file handling method could be used for permanent storage.
* **No real-time updates**: It does not consider external factors like bus delays or cancellations.

**11. Future Enhancements**

* **Database Integration**: Store bus and reservation data in a database like MySQL for persistent storage.
* **Multi-user support**: Allow multiple users to access the system simultaneously.
* **GUI Development**: Develop a graphical user interface (GUI) using libraries like GTK or Qt.
* **Web Integration**: Convert the system into a web application that can be accessed from any device.
* **Payment Integration**: Implement a payment gateway for ticket booking.

**12. Conclusion**

The **Bus Seat Reservation System** in C has provided a fundamental understanding of how to manage bus reservations, track seat availability, and cancel bookings in a programmatic way. By implementing this project, I gained practical experience in handling basic data structures, functions, file management, and user input validation in C programming.

This project can be further expanded with advanced features, making it more efficient and user-friendly. It is an excellent foundation for building larger-scale transport management systems.

**13. References**

* **“Let Us C” by Yashavant Kanetkar**
* **C Programming** tutorials from GeeksforGeeks and TutorialsPoint
* **C Language Documentation** for standard libraries and functions.