**CASE STUDY**

****

**C.V. RAMAN GLOBAL UNIVERSITY,**

**BHUBANESWAR, ODISHA, INDIA.**

**CASE STUDY ON:**

**Bus Reservation System**

**GROUP - 7**

**UNDER THE SUPERVISION OF**

**ASST. PROFESSOR**

**MR. SANDEEP PADESUR**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SUBMITTED BY:-**

|  |  |
| --- | --- |
| **NAME** | **REGD.NO** |
| **SUBHAM TARASIA** | **CL20250106019293124** |
| **OMMKAR SAI MISHRA** | **CL2025010601891092** |
| **SOMNATH MAHAPATRA** | **CL2025010601926495** |
| **SATYAJIT ROUTRAY** | **CL2025010601923263** |

**DECLARATION**

We hereby declare that the project titled “**Bus Reservation System**” submitted by us in partial fulfilment of the requirements for the 6th semester of Bachelor of Technology, was carried out under the supervision and guidance of **Mr. Sandeep Padesur Sir**.

**ACKNOWLEDGEMENT**

I would like to express my heartfelt gratitude and appreciation to my faculty teacher, **Mr. Sandeep Padesur**, for providing me with the invaluable opportunity to work on this case study on **Bus Reservation System** using **C** . His constant guidance, support, and encouragement have been instrumental in the completion of this project, helping me refine my understanding of machine learning concepts and techniques related to medical prediction models.

This project has significantly enhanced my knowledge of algorithms and methodologies applied in predictive healthcare, especially in the field of breast cancer diagnosis. I would also like to extend my sincere thanks to my teammates, whose cooperation and support were crucial in finalizing this project within the given timeframe.

|  |  |
| --- | --- |
| TOPIC | PAGE NO |
| INTRODUCTION | 6 |
| PROBLEM STATEMENT | 7 |
| METHEDOLOGY/ALGORITHMS USED | 7-15 |
| OUTPUT | 16-18 |
| CONCLUSION | 18-19 |
| REFERENCES | 20 |

**CONTENTS**

1. **Introduction -:**

### ****Bus Reservation System****

A console-based application designed to manage bus ticket bookings efficiently. Built using **C** with a text-based graphical interface, this system simplifies the process of reserving and managing bus seats for both passengers and administrators.

### ****Key Features****

1. **User-Friendly Interface**
   * ASCII-based GUI with headers, grids, and menus
   * Clear navigation using numbered options
2. **Core Functionalities**
   * View available bus routes
   * Book/Cancel tickets
   * Check real-time seat availability
   * Payment calculation (₹200 per seat)
3. **Data Persistence**
   * Stores reservations in text files (reservations1.txt, etc.)
   * Maintains seat counts in separate files (tr1.txt, etc.)
4. **Security**
   * Password-protected login system (user/pass)
5. **Visualization**
   * Seat grid layout (32 seats/bus)

**2. Problem Statement -:**

The Bus Reservation System addresses the inefficiencies of manual ticketing processes in public and private bus services. Traditional systems often suffer from:

- Human errors in seat allocation and record-keeping.

- No real-time visibility of seat availability.

- Cumbersome cancellation processes without refund tracking.

- Lack of centralized data storage for bookings.

This system automates seat reservations, cancellations, and status tracking to streamline operations for both passengers and bus operators.

Key Features to Solve the Problem

1. Real-Time Seat Management

- Display seat availability via a grid-based UI.

- Prevent double-booking using file-based data persistence.

2. User Authentication

- Secure login to restrict unauthorized access.

3. Cancellation with Error Handling

- Validate seat/bus numbers and ensure refund transparency.

- Guard against invalid inputs (e.g., non-existent seats).

4. Cost-Effective Design

- Terminal-based UI for low-resource environments.

- File storage instead of databases to reduce infrastructure costs.

Challenges Addressed

- Input Validation

- Ensure users cannot book/cancel invalid seats (e.g., seat 0 or 33).

- Data Corruption

- File operations handle incomplete writes/corruption checks.

- Concurrency

- Limitation: The current system does not handle simultaneous bookings (no file locking).

Technical Constraints

- Platform-dependent (relies on `conio.h` for Windows).

- Limited to 32 seats per bus.

- No encryption for stored passenger data.

Target Audience

- Small-scale bus operators.

- Local transport services in rural/urban areas.

- Educational institutions for internal transportation management.

Limitations

- Scalability: Adding buses requires manual code changes.

- No network support for remote bookings.

- Basic error handling (e.g., no retry mechanisms for file errors).

**3. Code/Algorithm Used -:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <conio.h>

#define HLINE "================================================================"

#define VLINE "||                                                              ||"

#define CLEAR system("cls")

char ch[10][130] = {"Mo Bus route number 19","OSRTC","op travels","Mo Bus route number 18","Shiv Shakti Transport","Mo Bus route number 17 ac"};

char name[32][100] = {'\0'};

char number[32][2] = {'\0'};

int num1[32] = {0};

int trno;

// Function prototypes

void printHeader();

void bus();

void name\_number(int booking, char numstr[100], int booked\_seats);

void booking();

int read\_number(int trno);

void read\_name(int trno);

void status();

void status\_1(int trno);

void cancle();

void login();

void printBoxedText(char text);

void seatGrid();

void printMainMenu();

void printExitScreen();

void printBusStatusHeader();

// Graphical elements

void printHeader() {

    CLEAR;

    printf("\n\n%s\n", HLINE);

    printf("||                BUS RESERVATION SYSTEM                      ||\n");

    printf("%s\n\n", HLINE);

}

void printBoxedText(char text) {

    printf("||  %-60s  ||\n", text);

}

void seatGrid() {

    printf("\n");

    printf("       =======================     =======================\n");

    printf("       | 1 | 2 | 3 | 4 |         | 5 | 6 | 7 | 8 |       \n");

    printf("       =======================     =======================\n");

    printf("       | 9 |10 |11 |12 |         |13 |14 |15 |16 |       \n");

    printf("       =======================     =======================\n");

    printf("       |17 |18 |19 |20 |         |21 |22 |23 |24 |       \n");

    printf("       =======================     =======================\n");

    printf("       |25 |26 |27 |28 |         |29 |30 |31 |32 |       \n");

    printf("       =======================     =======================\n\n");

}

void login() {

    char uname[10];

    char pword[10];

    char c;

    int i = 0;

    do {

        printHeader();

        printf("||                   LOGIN PAGE                            ||\n");

        printf("%s\n", HLINE);

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

        printf("||                  ENTER USERNAME:                        ||\n");

        printf("||                  > ");

        scanf("%9s", uname);

        printf("||                  ENTER PASSWORD:                        ||\n");

        printf("||                  > ");

        i = 0;

        while(i < 9) {

            c = getch();

            if(c == '\r') break;

            pword[i++] = c;

            printf(" ");

        }

        pword[i] = '\0';

        if(strcmp(uname,"user") == 0 && strcmp(pword,"pass") == 0) {

            printf("\n%s\n", HLINE);

            printf("||                LOGIN SUCCESSFUL!                         ||\n");

            printf("%s\n", HLINE);

            printf("\nPress any key to continue...");

            getch();

            break;

        } else {

            printf("\n%s\n", HLINE);

            printf("||              INVALID CREDENTIALS!                        ||\n");

            printf("%s\n", HLINE);

            printf("Press any key to try again...");

            getch();

        }

    } while(1);

}

void printMainMenu() {

    printHeader();

    printf("||                     MAIN MENU                            ||\n");

    printf("%s\n", HLINE);

    printBoxedText("1. View Bus List");

    printBoxedText("2. Book Tickets");

    printBoxedText("3. Cancel Booking");

    printBoxedText("4. Bus Status Board");

    printBoxedText("5. Exit");

    printf("%s\n", HLINE);

    printf("\nEnter your choice: ");

}

void bus() {

    printHeader();

    printf("||                    AVAILABLE BUSES                       ||\n");

    printf("%s\n", HLINE);

    for(int i = 0; i < 5; i++) {

        printf("||  %d. %-55s||\n", i+1, ch[i]);

    }

    printf("%s\n", HLINE);

    printf("\nPress any key to return to main menu...");

    getch();

}

void booking() {

    int i=0;

    char numstr[100];

    printHeader();

    printf("||                    TICKET BOOKING                         ||\n");

    printf("%s\n\n", HLINE);

    bus();

    printf("\nEnter the Bus number: ");

    scanf("%d",&trno);

    printHeader();

    printf("||                  SELECT YOUR SEATS                        ||\n");

    printf("%s\n\n", HLINE);

    status\_1(trno);

    int booked\_seats = read\_number(trno);

    int available\_seats = 32 - booked\_seats;

    if(available\_seats <= 0) {

        printHeader();

        printf("||              NO SEATS AVAILABLE!                          ||\n");

        printf("%s\n", HLINE);

        getch();

        return;

    }

    printf("\n\n\tAvailable Seats: %d\n", available\_seats);

    printf("\tNumber of Tickets: ");

    int booking;

    scanf("%d",&booking);

    if(booking > available\_seats) {

        printHeader();

        printf("||          NOT ENOUGH SEATS AVAILABLE!                    ||\n");

        printf("%s\n", HLINE);

        getch();

        return;

    }

    itoa(trno,numstr,10);

    name\_number(booking, numstr, booked\_seats);

    printHeader();

    printf("||                    PAYMENT DETAILS                         ||\n");

    printf("%s\n", HLINE);

    printf("\tTotal Amount: Rs.%d\n", 200 booking);

    printf("%s\n", HLINE);

    printf("\nPress any key to confirm payment...");

    getch();

}

void name\_number(int booking, char numstr[100], int booked\_seats) {

    char tempstr[100], tempstr1[20] = "status", tempstr2[20] = "number";

    FILE a, b;

    int current\_seats[32] = {0};

    int current\_count = 0;

    strcat(numstr, ".txt");

    strcat(tempstr1, numstr);

    strcat(tempstr2, numstr);

    a = fopen(tempstr1, "a");

    b = fopen(tempstr2, "a");

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

        int seat\_number;

        while(1) {

            printHeader();

            printf("||                  PASSENGER DETAILS %02d                      ||\n", i+1);

            printf("%s\n", HLINE);

            seatGrid();

            printf("\tSeat Number: ");

            scanf("%d", &seat\_number);

            if(seat\_number < 1 || seat\_number > 32) {

                printf("\nInvalid seat number. Please choose between 1 and 32.\n");

                printf("Press any key to continue...");

                getch();

                continue;

            }

            int is\_booked = 0;

            // Check existing bookings

            for(int j = 0; j < booked\_seats; j++) {

                if(num1[j] == seat\_number) {

                    is\_booked = 1;

                    break;

                }

            }

            // Check current session

            for(int j = 0; j < current\_count; j++) {

                if(current\_seats[j] == seat\_number) {

                    is\_booked = 1;

                    break;

                }

            }

            if(is\_booked) {

                printf("\nSeat %d is already booked. Please choose another seat.\n", seat\_number);

                printf("Press any key to continue...");

                getch();

            } else {

                current\_seats[current\_count++] = seat\_number;

                break;

            }

        }

        printf("\tPassenger Name: ");

        scanf("%s", name[seat\_number - 1]);

        fprintf(a, "%s ", name[seat\_number - 1]);

        fprintf(b, "%d ", seat\_number);

    }

    fclose(a);

    fclose(b);

}

int read\_number(int trno) {

    char tempstr2[20] = "number";

    FILE a;

    char numstr[100];

    int i = 0, j = 0;

    itoa(trno, numstr, 10);

    strcat(numstr, ".txt");

    strcat(tempstr2, numstr);

    a = fopen(tempstr2, "r");

    if (a == NULL) {

        return 0;

    }

    while(fscanf(a, "%d", &num1[i]) != EOF) {

        i++;

    }

    fclose(a);

    return i;

}

void read\_name(int trno) {

    char tempstr1[20] = "status";

    FILE b;

    char numstr[100];

    int i = 0, j = 0;

    itoa(trno, numstr, 10);

    strcat(numstr, ".txt");

    strcat(tempstr1, numstr);

    b = fopen(tempstr1, "r");

    if (b == NULL) {

        return;

    }

    while(fscanf(b, "%s", name[i]) != EOF) {

        i++;

    }

    fclose(b);

}

void status() {

    printHeader();

    printf("||                  BUS STATUS BOARD                         ||\n");

    printf("%s\n", HLINE);

    int trno;

    printf("\tEnter Bus Number: ");

    scanf("%d",&trno);

    status\_1(trno);

    printf("\nPress any key to continue...");

    getch();

}

void status\_1(int trno) {

    int j = read\_number(trno);

    read\_name(trno);

    char tempname[33][10] = {"Empty"};

    printHeader();

    printf("||                  SEAT AVAILABILITY                        ||\n");

    printf("%s\n", HLINE);

    printf("\tBus Number: %d (%s)\n\n", trno, ch[trno-1]);

    // Create tempname array with passenger names

    for(int i = 0; i < j; i++) {

        if(num1[i] > 0 && num1[i] <= 32) {

            strcpy(tempname[num1[i]], name[i]);

        }

    }

    // Display seats in grid format

    for(int i = 0; i < 32; i += 8) {

        // First section

        for(int s = i; s < i+4; s++) {

            printf(" %2d.%-8s", s+1, tempname[s+1]);

        }

        printf("     |     ");

        // Second section

        for(int s = i+4; s < i+8; s++) {

            printf(" %2d.%-8s", s+1, tempname[s+1]);

        }

        printf("\n");

    }

    int available = 0;

    for(int i = 1; i <= 32; i++) {

        if(strcmp(tempname[i], "Empty") == 0) available++;

    }

    printf("\n%s\n", HLINE);

    printf("||               Available Seats: %-2d                     ||\n", available);

    printf("%s\n", HLINE);

    printf("\nPress any key to continue...");

    getch();

}

void cancle() {

    printHeader();

    printf("||                  CANCEL BOOKING                           ||\n");

    printf("%s\n", HLINE);

    int seat\_no;

    printf("\tEnter Bus Number: ");

    scanf("%d",&trno);

    printf("\tEnter Seat Number: ");

    scanf("%d",&seat\_no);

    char numstr[100], tempstr2[20] = "number", tempstr1[20] = "status";

    itoa(trno, numstr, 10);

    strcat(numstr, ".txt");

    strcat(tempstr1, numstr);

    strcat(tempstr2, numstr);

    int booked\_seats = read\_number(trno);

    read\_name(trno);

    FILE a, b;

    a = fopen(tempstr1, "w");

    b = fopen(tempstr2, "w");

    for(int i = 0; i < booked\_seats; i++) {

        if(num1[i] != seat\_no) {

            fprintf(b, "%d ", num1[i]);

            fprintf(a, "%s ", name[i]);

        }

    }

    fclose(a);

    fclose(b);

    printHeader();

    printf("||               CANCELLATION SUCCESSFUL                     ||\n");

    printf("%s\n", HLINE);

    printf("\tRefund Amount: Rs.200\n");

    printf("%s\n", HLINE);

    printf("\nPress any key to continue...");

    getch();

}

int main() {

    login();

    int choice;

    do {

        printMainMenu();

        scanf("%d", &choice);

        switch(choice) {

            case 1: bus(); break;

            case 2: booking(); break;

            case 3: cancle(); break;

            case 4: status(); break;

            case 5:

                printHeader();

                printf("||         THANK YOU FOR USING OUR SYSTEM!               ||\n");

                printf("%s\n", HLINE);

                exit(0);

            default:

                printHeader();

                printf("||              INVALID CHOICE!                         ||\n");

                printf("%s\n", HLINE);

                printf("Press any key to continue...");

                getch();

        }

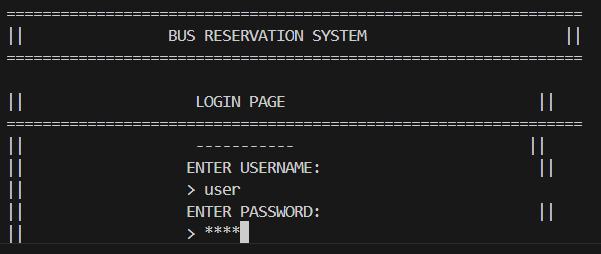
    } while(1);

    return 0;

}

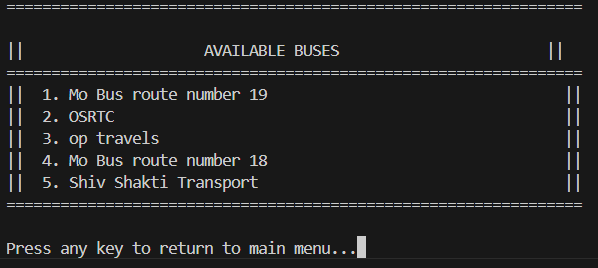
**4. Output:**

1. **Login page :**

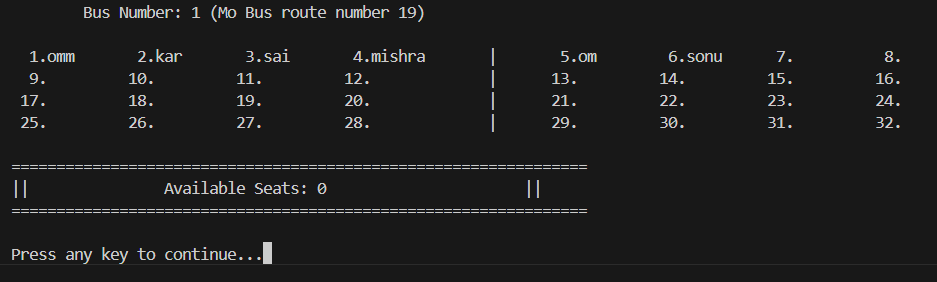




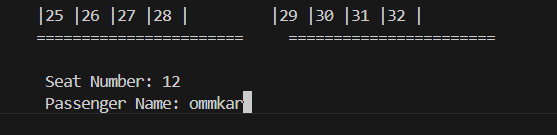
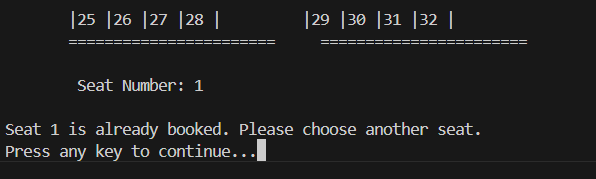
1. **View Bus List :**

****

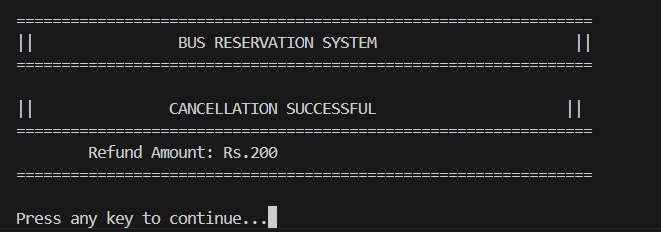
1. **Book Tickets:**

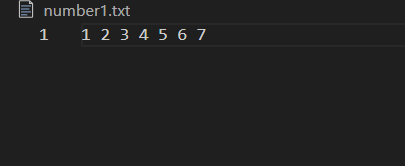
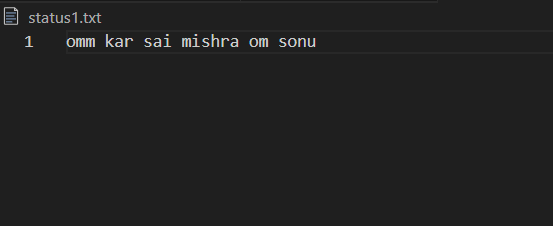


****

****

1. **Cancel booking:**

****

**4. Data base:**  **-:**

**5. Conclusion -:**

The Bus Reservation System developed in C demonstrates a practical application of core programming concepts such as file handling, user authentication, and terminal-based UI design. Key features include:

- User-Friendly Interface: Utilizes bordered menus, seat grids, and real-time status updates for seamless navigation.

- Data Persistence: Leverages file operations (`status[trno].txt`, `number[trno].txt`) to store booking details securely.

- Modular Design: Functions like `booking()`, `cancle()`, and `status\_1()` ensure code reusability and maintainability.

Technical Insights

- The project relies on non-standard libraries like `conio.h` for terminal I/O, limiting cross-platform compatibility. Future iterations could adopt libraries like NCurses for broader OS support.

- Tools like GCC and MinGW were instrumental in compiling and testing the code, while AI assistants like DeepSeek aided in debugging and logic refinement.

Learning Outcomes

- Strengthened understanding of C programming fundamentals , including arrays, loops, and file operations.

- Gained experience in designing interactive console applications with structured menus and input validation.

Future Enhancements

- Integrate aGUI (Graphical User Interface) using frameworks like GTK or Qt.

- Implement a database system (e.g., SQLite) for efficient data management.

- Add features like online payments, seat maps for multiple buses, and admin panels.

This project serves as a foundational step toward building more complex reservation systems while emphasizing the importance of clean code practices and user-centric design.

Thank you for using the Bus Reservation System

**6. References -:**

### ****6. References****

#### ****C Compilers and Development Environments****

* **GCC (GNU Compiler Collection)**  
  Used for compiling and testing the C code on Linux/Unix systems.  
  Reference: [GCC Official Documentation](https://gcc.gnu.org/onlinedocs/)
* **MinGW (Minimalist GNU for Windows)**  
  Used for compiling the code on Windows systems (supports conio.h for terminal I/O).  
  Reference: [MinGW Wiki](https://osdn.net/projects/mingw/" \t "_blank)

#### ****AI Tools for Code Assistance****

* **DeepSeek-R1-Lite-Preview**  
  Used for code logic suggestions, debugging, and documentation formatting.  
  Reference: [DeepSeek](https://www.deepseek.com/" \t "_blank)