

K L UNIVERSITY
FRESHMAN ENGINEERING DEPARTMENT

A Project Based Lab Report

On

MOVIE TICKET BOOKING

SUBMITTED BY:

I.D NUMBER

NAME

2300030652

SRIRAM PRAVALLIKA

UNDER THE ESTEEMED GUIDANCE OF

Mr.Ch.Anil

Asst. Professor



KL UNIVERSITY
Green fields, Vaddeswaram – 522 502
Guntur Dt., AP, India.

DEPARTMENT OF BASIC ENGINEERING SCIENCES



CERTIFICATE

This is to certify that the project based laboratory report entitled “MOVIE TICKET BOOKING” submitted by Ms. **SRIRAM PRAVALLIKA** bearing Regd. No. 2300030652 to the **Department of Basic Engineering Sciences, KL University** in partial fulfillment of the requirements for the completion of a project based Laboratory in “**DATA STRUCTURES-23SC1202**” course in I B Tech II Semester, is a bonafide record of the work carried out by him/her under my supervision during the academic year 2023– 2024.

PROJECT SUPERVISOR

Mr.Ch. ANIL

HEAD OF THE DEPARTMENT

Dr. D.HARITHA

ACKNOWLEDGEMENTS

It is great pleasure for me to express my gratitude to our honorable President **Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project based laboratory report.

I express the sincere gratitude to our principal **Dr. A. Anand Kumar** for his administration towards our academic growth.

I express sincere gratitude to our Coordinator **Dr. SivaGangaPrasad** for her leadership and constant motivation provided in successful completion of our academic semester.

I record it as my privilege to deeply thank our pioneer **Dr. D.Haritha**, HOD-BES for providing us the efficient faculty and facilities to make our ideas into reality.

I express my sincere thanks to our project supervisor **Mr.Ch. Anil** for his/her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

2300030652

SRIRAM PRAVALLIKA

ABSTRACT

1.Struct Definition: The struct Movie is used to represent movie information, including the title and available seats.

2.Initialization: The initializeMovies function sets up movie data. You can fill in the details for each movie, such as title and available seats.

3.Displaying Movies: The displayMovies function is responsible for showing the available movies. You can enhance this function to display more details about each movie.

4.Booking Tickets: The bookTicket function is designed to book tickets for a specific movie. It checks if there are enough available seats and updates the seat count accordingly.

5.Main Function: The main function is the entry point of the program. It initializes movies, and then it presents a menu to the user with options to display movies, book tickets, or exit the program.

6.Menu Handling: The program uses a do-while loop to repeatedly display the menu and execute the corresponding actions until the user chooses to exit.

7.User Input: The program uses scanf to take user input for menu choices, movie index, and the number of tickets to book.

8.Compile and Run: You can compile this code using a C compiler (e.g., GCC) and run the executable. Follow the on-screen instructions to navigate through the menu and interact with the program.

INDEX

| S.NO | TITLE | PAGE NO |
|------|--------------------------------|---------|
| 1. | Introduction | 5 |
| 2. | Aim of the Project | 6 |
| 2.1 | Advantages & Disadvantages | 6-8 |
| 3. | System Requirements | 9 |
| 4. | Data Flow Diagram | 10 |
| 5. | Algorithm for each module | 11 |
| 6. | Implementation | 12-17 |
| 7. | Integration and System Testing | 18-19 |
| 8. | Conclusion | 20 |
| 9. | Future Enhancement | 21 |
| 10. | References | 22 |

INTRODUCTION

An online movie ticket booking system is a digital platform that allows customers to conveniently browse movie showtimes, select seats, and purchase tickets from any device with an internet connection. This seamless process improves efficiency for both customers and theater owners.

Since the emergence of the World Wide Web, merchants have sought to sell movie tickets directly to people who surf the internet. Gone are the days of standing in long queues at the box office or relying solely on physical ticket counters. With online booking, moviegoers can explore available showtimes, choose specific seats, and complete transactions without leaving the comfort of their homes or using dedicated kiosks at the cinema.

Online movie ticket booking systems have revolutionized the moviegoing experience. They allow customers to:

- Browse showtimes and purchase tickets anytime, anywhere: Whether it's early morning or late at night, users can access movie schedules and book tickets at their convenience.
- Select specific seats when buying tickets: No more guesswork about where you'll be sitting. Choose your preferred seats directly from the seating layout.
- Access special offers and promotions: Online platforms often provide discounts, cashback, or combo deals, enhancing the value for movie enthusiasts.
- Store payment information for faster checkout: Save time by securely storing payment details, making subsequent bookings quicker and hassle-free.

AIM :

The goal of our Movie Ticket Booking project is to create a simple and easy-to-use system for people to book movie tickets using the C programming language. We want to make sure users can easily see available seats, choose where they want to sit, and book tickets hassle-free. Our aim is also to guide users through the process, handle mistakes like wrong inputs, and keep them informed about seat availability in real-time.

ADVANTAGES:

1. Easy to use:
 - Users can easily pick a movie they want to watch.
2. Error Checking:
 - The code makes sure that users enter a valid movie choice.
3. For Individuals or Groups:
 - It works whether you're booking a ticket for yourself or a group of friends.
4. Flexible Date and Time:
 - You can choose when and at what time you want to watch the movie.
5. Clear Information:
 - It shows you all the important details like movie, date, time, and cost.
6. Understandable Code:
 - The code is written in a way that's easy to understand.
7. Learning Tool:
 - It's good for people learning how to program in C.

8. Calculates Cost:

- It figures out how much you have to pay based on the number of people.

9. Readability:

- The code is written in a way that's easy to read, like telling a story.

10. Basic and Practical:

- It covers the basics of programming and does a practical thing - booking a movie ticket.

DISADVANTAGES:-

1. Can't Handle All Mistakes:

- The code might not catch every possible mistake a user could make.

2. Runs Once and Stops:

- It doesn't keep running for multiple bookings; it finishes after one use.

3. Same Movies Every Time:

- You can only choose from the three movies listed in the code.

4. Specific Date Format:

- It expects the date in one specific way (day:month:year).

5. Limited Movie Info:

- It only tells you the name of the movie; no extra details like genre or ratings.

6. No Security Measures:

- It doesn't have protection against certain types of computer attacks.

7. Forgetful Program:

- It forgets everything after you close it; no history of past bookings.

8. No Passwords or Logins:

- It doesn't ask for a password or any personal information to confirm who you are.

9. One at a Time:

- It's designed for one person; it can't handle multiple bookings happening at the same time.

SYSTEM REQUIREMENTS

➤ SOFTWARE REQUIREMENTS:

The major software requirements of the project are as follows:

Language : Turbo-C

Operating system: Windows Xp or later.

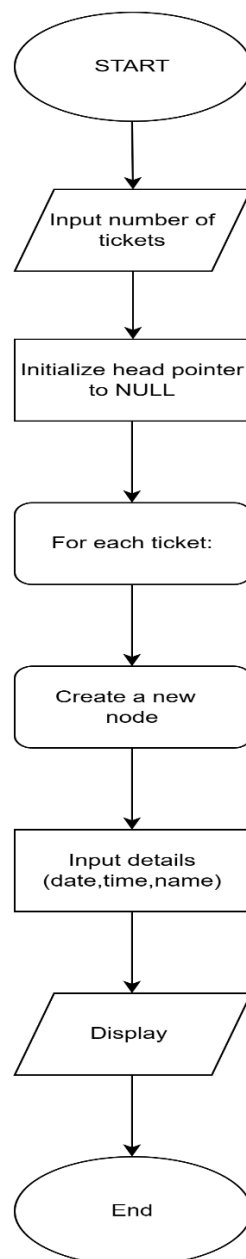
➤ HARDWARE REQUIREMENTS:

The hardware requirements that map towards the software are as follows:

RAM : Adequate RAM (e.g., 16 GB or more for servers, 8 GB or more for workstations) to ensure smooth operation of the software, including the application and database.

Processor : Multi-core processors (e.g., quad-core or higher) for servers and workstations, providing the necessary computational power to handle concurrent requests and database operations efficiently

DATA FLOW DIAGRAM



ALGORITHM

STEP1:Start

STEP2: Display "WELCOME IN OUR OFFICIAL PAGE" and movie list.

STEP3: Prompt user to "Choose the movie" and store in 'id'.

STEP4: Check 'id':

- If 'id' is 1, display "Entering into Jawan..."
- If 'id' is 2, display "Entering into Animal..."
- If 'id' is 3, display "Entering into Hi Nanna..."
- If 'id' is not 1, 2, or 3, display "Database error. Please enter correct digits" and exit.

STEP5: Prompt user for "Number of Audience" and store in 'n'.

STEP6: If 'n' is 1, prompt for and store the name of the audience in 'name'.

- If 'n' is more than 1, use a loop to input and store names in 'name' array.

STEP7: Prompt user for "In which date you want to see the movie_:
12:2023" and store in 'date'.

STEP8: Prompt user to "Choose the time" and store in 't'.

STEP9: Display selected movie, date, time, and the total amount to be paid(Rs. 177 per audience) with gst and other taxes.

STEP 10: Display "Thanking you for entering into the server" and exit.

IMPLEMENTATION

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct Ticket {

    int ticketNumber;

    char movieName[50];

    float ticketPrice;

    struct Ticket *next;

} Ticket;

Ticket* createTicket(int ticketNumber, char movieName[], float ticketPrice);

void bookTicket(Ticket **head, int ticketNumber, char movieName[], float ticketPrice);

void cancelTicket(Ticket **head, int ticketNumber);

void displayTickets(Ticket *head);

void refundAmount(Ticket **head, int ticketNumber);

int main() {

    Ticket *ticketList = NULL;

    int choice, ticketNumber;

    char movieName[50];

    float ticketPrice;
```

```
do {  
    printf("\n1. Book Ticket\n");  
    printf("2. Cancel Ticket\n");  
    printf("3. Display Tickets\n");  
    printf("4. Refund Ticket\n");  
    printf("5. Exit\n");  
    printf("Enter your choice: ");  
    scanf("%d", &choice);  
    switch(choice) {  
        case 1:  
            printf("Enter ticket number: ");  
            scanf("%d", &ticketNumber);  
            printf("Enter movie name: ");  
            scanf("%s", movieName);  
            scanf("%f", &ticketPrice);  
            bookTicket(&ticketList, ticketNumber, movieName, ticketPrice);  
            break;  
        case 2:  
            printf("Enter ticket number to cancel: ");  
            scanf("%d", &ticketNumber);  
            cancelTicket(&ticketList, ticketNumber);  
            break;
```

```

    case 3:
        displayTickets(ticketList);
        break;
    case 4:
        printf("Enter ticket number to refund: ");
        scanf("%d", &ticketNumber);

        refundAmount(&ticketList, ticketNumber); // Provide the ticketList and
ticketNumber

        break;
    case 5:
        printf("Exiting...\n");
        break;
    default:
        printf("Invalid choice. Please try again.\n");
}
} while(choice != 5);
return 0;
}

```

```

Ticket* createTicket(int ticketNumber, char movieName[], float ticketPrice) {
    Ticket *newTicket = (Ticket*)malloc(sizeof(Ticket));
    newTicket->ticketNumber = ticketNumber;
    strcpy(newTicket->movieName, movieName);
    newTicket->ticketPrice = ticketPrice;
}

```

```

    newTicket->next = NULL;

    return newTicket;
}

void bookTicket(Ticket **head, int ticketNumber, char movieName[], float
ticketPrice) {

    Ticket *newTicket = createTicket(ticketNumber, movieName, ticketPrice);

    if (*head == NULL) {

        *head = newTicket;

    } else {

        Ticket *temp = *head;

        while (temp->next != NULL) {

            temp = temp->next;

        }

        temp->next = newTicket;

    }

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

}

void cancelTicket(Ticket **head, int ticketNumber) {

    Ticket *prev = NULL;

    Ticket *temp = *head;

    while (temp != NULL && temp->ticketNumber != ticketNumber) {

        prev = temp;

        temp = temp->next;

    }

```



```

if (temp == NULL) {
    printf("Ticket not found.\n");
    return;
}

if (prev == NULL) {
    *head = temp->next;
} else
{
    prev->next = temp->next;
}

free(temp);
printf("Ticket cancelled successfully!\n");
}

void displayTickets(Ticket *head) {
    if (head == NULL) {
        printf("No tickets booked.\n");
        return;
    }

    printf("Ticket Number\tMovie Name\tTicket Price\n");

    while (head != NULL) {
        printf("%d\t%s\t%.2f\n", head->ticketNumber, head->movieName, head->ticketPrice);

        head = head->next;
    }
}

```

```

    }
}

void refundAmount(Ticket **head, int ticketNumber) {

    Ticket *temp = *head;

    Ticket *prev = NULL;

    while (temp != NULL && temp->ticketNumber != ticketNumber) {

        prev = temp;

        temp = temp->next;

    }

    if (temp == NULL) {

        printf("Ticket not found.\n");

        return;

    }

    float refundedAmount = temp->ticketPrice;

    if (prev == NULL) {

        *head = temp->next;

    } else {

        prev->next = temp->next;

    }

    free(temp);

    printf("Ticket with ticket number %d refunded successfully. Refunded amount:
%.2f\n", ticketNumber, refundedAmount);

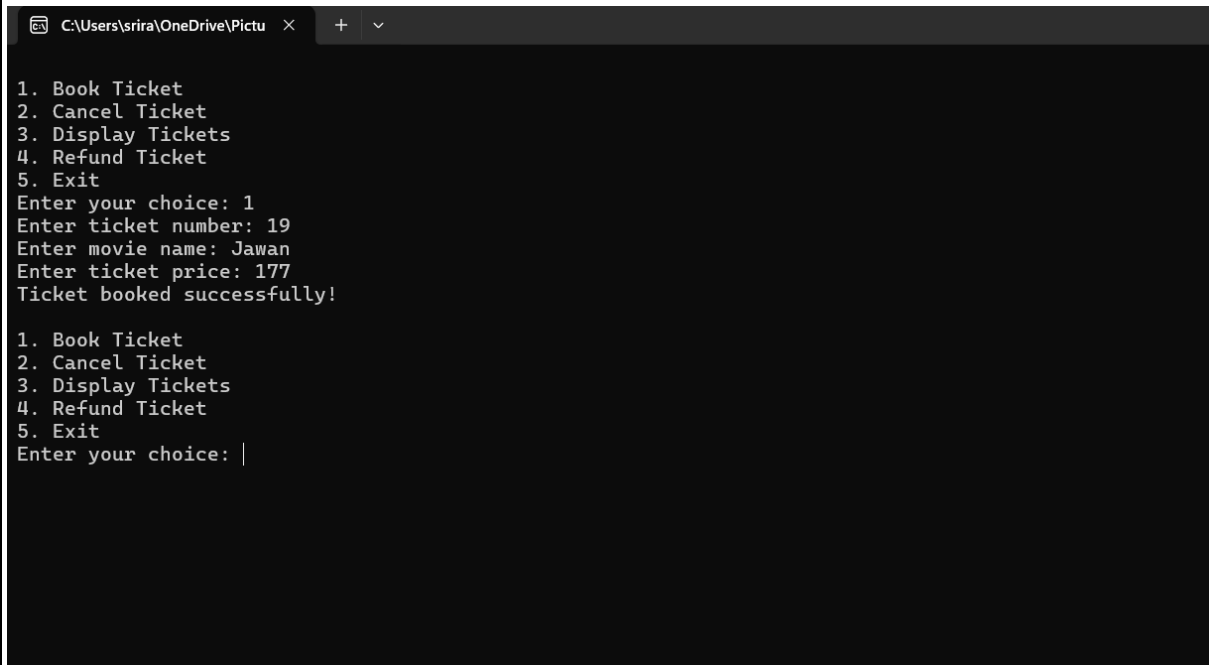
}

```

INTEGRATION AND SYSTEM TESTING

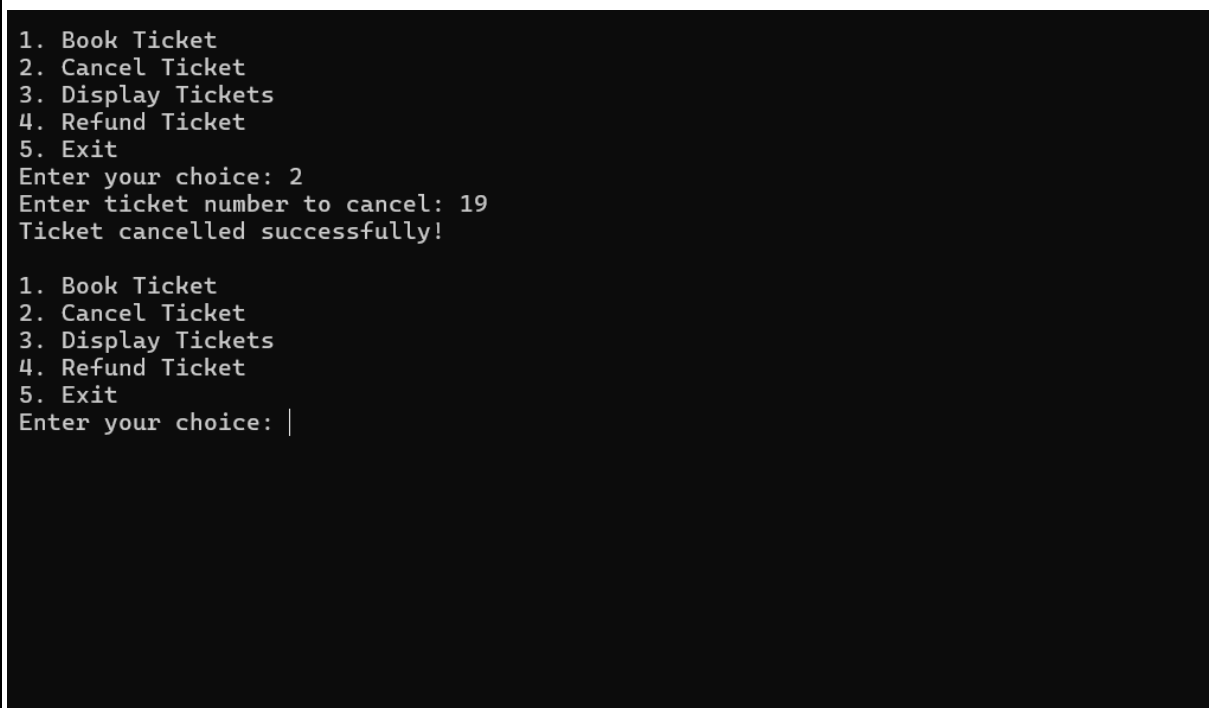
OUTPUTS

Screen Shots:



```
C:\Users\srira\OneDrive\Pictu × + v
1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 1
Enter ticket number: 19
Enter movie name: Jawan
Enter ticket price: 177
Ticket booked successfully!

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: |
```



```
1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 2
Enter ticket number to cancel: 19
Ticket cancelled successfully!

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: |
```

```
C:\Users\srira\OneDrive\Pictu × + v

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 1
Enter ticket number: 19
Enter movie name: Jawan
Enter ticket price: 177
Ticket booked successfully!

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 2
Enter ticket number to cancel: 19
Ticket cancelled successfully!

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 3
No tickets booked.

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: |
```

```
C:\Users\srira\OneDrive\Pictu × + v

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 1
Enter ticket number: 12
Enter movie name: Jawan
Enter ticket price: 177
Ticket booked successfully!

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: 4
Enter ticket number to refund: 12
Ticket with ticket number 12 refunded successfully. Refunded amount: 177.00

1. Book Ticket
2. Cancel Ticket
3. Display Tickets
4. Refund Ticket
5. Exit
Enter your choice: |
```

CONCLUSION

In conclusion, our Movie Ticket Booking project has met its goals. We've built a user-friendly application in C that lets users pick seats, book tickets, and get updates on available seats. We've made sure to catch and fix mistakes, like if someone enters the wrong information. The code is organized well and easy to understand, and our documentation helps users know what to do. Overall, our project makes booking movie tickets a smooth experience for both users and theater managers.

FUTURE ENHANCEMENTS

- **More Movies to Choose From:** Allow the program to easily add or remove movies without changing the code.
- **Make It Safer:** Add a way for users to have their own accounts and passwords, so their information is secure.
- **Remember Past Bookings:** Keep a record of all the movies someone has booked in the past.
- **Book for Friends at the Same Time:** Allow the program to handle multiple people booking tickets at once.
- **Understand Different Time Formats:** Make the program smart enough to understand different ways people might want to write the date and time.
- **Know More About the Movies:** Provide more details about the movies, like what genre they are, who the actors are, and how good people think they are.
- **Change Your Mind:** Let users cancel a booking if they change their plans and give them their money back.
- **Fix Mistakes Easily:** Make the program better at understanding when someone makes a mistake and help them fix it.

REFERENCES

- “Data Structures and Algorithms”, Pearson Education, First Edition Reprint2003.
Authors: A.V.Aho, J. E. Hopcroft, and J. D. Ullman
- “Fundamentals of datastructures in C” , Second Edition-2007.
Authors: Horowitz, Sahni, Anderson Freed.
- “Data Structures”, Second Edition, Thomson India Edition, 2005.
Authors: R. F. Gilberg, B. A. Forouzan.
- “Data Structures & Program Design in C”, FourthEdition-2007.
Authors: Robert Kruse, C.L. Tondo, Bruce Leung, Shashi Mogalla.
- https://onlinecourses.nptel.ac.in/noc19_mg30/preview