 ** P.S.R ENGINEERING COLLEGE **

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**PROJECT REPORT**

**Movie Ticket Booking System**

***Submitted by***

# 

P. Arul Chelliah (Regno:95192101016)

***In partial fulfillment for the award of the degree***

***Of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**P.S.R. ENGINEERING COLLEGE, SIVAKASI**

January 2024

ACKNOWLEDEMENT

We take this opportunity to all those who helped towards successful completion of this mini project. At the very outset we thank the almighty for his profuse blessings showered on us. We thank our beloved parents whose encouragement and support help us to complete our project successfully.

It is our greatest pleasure to convey our thanks to **Thiru R. Solaisamy, Correspondent** and **Director Er.S.vigneswari Arunkumar B.Tech.,PSR engineering college, Sivakasi** for providing required facilities and suitable infrastructure to complete our project.

It is our greatest privilege to convey our thanks to **J.S SENTHILKUMAAR M.E.,Ph.D.,Principal** for continuous support to complete our project without hurdles.

We proud profound gratitude to our beloved Head of the Department **Dr.A.Ramathilagam, M.E., Ph.D., Professor** for providing ample facilities to complete our project successfully.

We also wish to express our sincere thanks to our trainer **Mr.Abdul kadir,**  for assisting us in all aspects

We also bound to thanks to all Faulty and Non-teaching staff members of The **Department of Computer Science and Engineering** whose support and cooperation also contributed much to complete this project work.

## ABSTRACT

The Movie Ticket Booking System is a robust and user-friendly standalone application

developed in the C programming language, tailored to streamline and automate the movie ticket

booking process. This system addresses the complexities associated with managing ticket

availability, seat selection, and customer information, providing a seamless and efficient solution

for both administrators and moviegoers.

**TABLE OF CONTENT**

| **CHAPTER** | **TITLE** | **PAGE NO** |
| --- | --- | --- |
|  | **ABSTRACT** | 3 |
| 1 | **INTRODUCTION** | 5 |
| 2 | **LITERATURE OVERVIEW** | 6 |
| 3 | **METHODOLOGY** | 8 |
| 4 | **SYSTEM ARCHITECTURE/DESIGN** | 11 |
| 5 | **IMPLEMENTATION** | 14 |
| 6 | **SOURCE CODE** | 15 |
| 7 | **RESULTS AND EVALUATION** | 20 |
| 8 | **DISCUSSION** | 21 |
| 9 | **OUTPUT** | 22 |
| 10 | **CONCLUSION** | 22 |

**INTRODUCTION**

In the realm of entertainment management, the Movie Ticket Booking System emerges as a pivotal

solution designed to revolutionize the movie ticket booking process. Developed as a standalone

application in the versatile C programming language, this system addresses the intricacies associated

with ticket availability, seat selection, and customer engagement.

Through this introduction, we set the stage for an in-depth exploration of the Movie Ticket Booking

System, showcasing its potential to enhance the overall movie-going experience. The subsequent

chapters will delve into the system's architecture, design, methodology, implementation details, and the

results of testing and evaluation, providing a comprehensive understanding of its functionality and

impact in the domain of entertainment technology.

Existing Systems

**LITERATURE OVERVIEW**

The landscape of movie ticket booking systems has witnessed considerable attention from researchers and developers, with a focus on creating efficient solutions to streamline and enhance the overall movie-going experience. This literature overview provides insights into key concepts and existing projects related to movie ticket booking systems implemented in the C programming language.

Efficiency in Ticket Allocation:

Numerous studies underscore the importance of resource optimization in movie ticket booking systems. Implementing such systems in C facilitates low-level memory control, allowing for efficient resource utilization and ensuring a seamless user experience. This efficiency in ticket allocation contributes significantly to the overall system performance.

Data Security and Integrity:

Security is a critical concern in any ticket booking system. Studies highlight the need for robust data security mechanisms. Leveraging C's capabilities in managing memory and providing fine-grained control over data structures contributes to building secure movie ticket booking systems, ensuring the confidentiality and integrity of user information.

Transaction Processing:

Efficient transaction processing is a key aspect of movie ticket booking systems. Literature emphasizes the need for robust algorithms to handle transactions seamlessly. The ability of C to implement complex algorithms facilitates swift and accurate transaction processing, ensuring data consistency and reliability in financial transactions.

Error Handling and System Reliability:

System reliability is paramount in movie ticket booking, and literature emphasizes the need for robust error-handling mechanisms. C's structured programming allows developers to implement effective error-checking, enhancing the overall reliability of the system. This ensures graceful handling of unexpected scenarios, contributing to a more stable and dependable system.

Scalability and Portability:

Scalability and portability are essential for adapting the system to different hostel sizes and environments. C, known for its portability across different platforms, offers an advantageous choice for building scalable hostel management systems.

**METHODOLOGY**

The development and implementation of the Movie Ticket Booking System in C involves a structured approach to ensure the efficient realization of its objectives. The methodology encompasses several key components, each contributing to the system's functionality and reliability.

**1. Requirements Analysis:**

User Stories: Identify and document user stories to understand the system's functional requirements from both the customer and administrator perspectives.

**Stakeholder Interviews:**

Conduct interviews with potential users and administrators to gather insights into their expectations and preferences.

**2. System Design:**

Use Case Diagrams:Develop use case diagrams to illustrate the interactions between system components and actors, defining the functionalities and responsibilities**.**

Data Flow Diagrams:Create data flow diagrams to represent the flow of data within the system, highlighting inputs, processes, and outputs.

**3. Database Design:**

**Entity-Relationship Diagram (ERD):**

Design an ERD to model the relationships between entities such as movies, customers, and bookings.

**Normalization:**

Apply normalization techniques to ensure data integrity and reduce redundancy in the database structure.

**4. Implementation:**

Programming Language Selection: Choose C as the programming language for its efficiency, low-level memory control, and suitability for system-level programming.

**5. Deployment:**

Server Configuration: Set up the necessary server infrastructure for hosting the system.

User Training: Provide training sessions for administrators and users on system functionalities and best practices.

**6. Maintenance and Updates:**

**Bug Tracking:**

Establish a system for tracking and addressing reported bugs.

**Regular Updates:**

Plan for regular updates to introduce new features, address security concerns, and improve overall system performance.

**SYSTEM ARCHITECTURE AND DESIGN**

The Movie Ticket Booking System's architecture and design encompass various components to ensure efficient functionality, data management, and a user-friendly interface.

**1. User Interface (UI):**

**Administrator Interface:**

- Console-based menus and screens for administrators.

- Authentication and authorization mechanisms for secure access.

Customer Interface:

- Intuitive menu options for customers to browse movie schedules and book tickets.

- Secure authentication for customer accounts.

**2. Data Management:**

**File Handling:**

- Utilizes C's file handling capabilities for persistent storage of data.

- Files for movie schedules, seat availability, and customer information.

-Database Integration (Optional):

- Potential integration with a relational database for enhanced data management.

**3. Business Logic:**

**Functions:**

- Book Ticket (): Handles the ticket booking process.

- View Schedule (): Displays available movie schedules.

- Generate Report (): Generates reports on ticket sales and movie schedules.

**Validation:**

- Ensures data integrity and performs necessary validations during ticket booking.

**4. User Authentication:**

- Administrator Authentication:

- Validates administrator credentials during login.

- Authorization mechanisms to restrict access based on roles.

**Customer Authentication:**

- Validates customer credentials during login.

- Secure handling of sensitive customer information.

**5. Error Handling:**

- Graceful Handling:

- Incorporates error-checking mechanisms to handle unexpected scenarios gracefully.

- Informative error messages for users and administrators.

**6. Testing:**

**Unit Testing:**

- Tests individual modules for correctness.

- Identifies and resolves any issues.

**Integration Testing:**

- Verifies seamless interaction between different components.

- Ensures the system operates as a unified whole.

**IMPLEMENTATION:**

The implementation of the Movie Ticket Booking System in C involves translating the design and functionality into executable code. This section outlines key aspects of the implementation, including code snippets for critical functionalities.

**1. Programming Language Selection:**

Utilizes C for its efficiency, low-level memory control, and suitability for system-level programming.

**2. Modular Programming:**

Breaks down the system into modular functions for enhanced code maintainability and readability.

void book Ticket (struct Movie Schedule \*schedule, struct Customer \*customer) {

}

**Error Handling:**

Incorporates error-checking mechanisms to handle unexpected scenarios gracefully.

Example

void handle Ticket Booking Error (int Error Code) {

}

SOURCE CODE:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_MOVIES 5

#define MAX\_NAME\_LENGTH 50

#define MAX\_TICKETS 100

struct Movie {

char name[MAX\_NAME\_LENGTH];

int availableTickets;

};

struct Ticket {

char movieName[MAX\_NAME\_LENGTH];

int quantity;

};

struct Movie movies[MAX\_MOVIES];

struct Ticket bookedTickets[MAX\_TICKETS];

int totalTicketsBooked = 0;

void displayMovies() {

printf("\nAvailable Movies:\n");

for (int i = 0; i < MAX\_MOVIES; i++) {

printf("%d. %s - Tickets: %d\n", i + 1, movies[i].name, movies[i].availableTickets);

}

}

void bookTickets() {

int movieIndex, quantity;

displayMovies();

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

scanf("%d", &movieIndex);

if (movieIndex < 1 || movieIndex > MAX\_MOVIES) {

printf("Invalid movie number.\n");

return;

}

movieIndex--; // Adjust index to match array indexing

printf("Enter the number of tickets to book: ");

scanf("%d", &quantity);

if (quantity <= 0 || quantity > movies[movieIndex].availableTickets) {

printf("Invalid quantity or not enough tickets available.\n");

return;

}

movies[movieIndex].availableTickets -= quantity;

strcpy(bookedTickets[totalTicketsBooked].movieName, movies[movieIndex].name);

bookedTickets[totalTicketsBooked].quantity = quantity;

totalTicketsBooked++;

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

}

void displayBookedTickets() {

printf("\nBooked Tickets:\n");

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

printf("%s - Quantity: %d\n", bookedTickets[i].movieName, bookedTickets[i].quantity);

}

}

int main() {

// Initialize movie data

strcpy(movies[0].name, "Inception");

movies[0].availableTickets = 50;

strcpy(movies[1].name, "The Dark Knight");

movies[1].availableTickets = 30;

strcpy(movies[2].name, "Interstellar");

movies[2].availableTickets = 40;

int choice;

do {

printf("\n1. Display Movies\n2. Book Tickets\n3. Display Booked Tickets\n4. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

displayMovies();

break;

case 2:

bookTickets ();

break;

case 3:

displayBookedTickets();

break;

case 4:

printf("Exiting the program.\n");

break;

default:

Printf("Invalid choice. Please try again.\n");

}

} while (choice != 4);

return 0;

}

**RESULT AND EVALUTION**

**1. Result:**

- The program takes input for an employee's name, basic salary, allowances, and deductions.

- It calculates the net salary using a simple formula: `basicSalary + allowances - deductions`.

- It then displays the employee's details (name, basic salary, allowances, deductions) along with the calculated net salary.

**2Resolution:**

The program uses a structured approach by defining an Employee structure to encapsulate related data.

User input is obtained using `std::cin` and `std::getline` to ensure flexibility in capturing strings with spaces (like the employee's name).

The calculation of the net salary is modularized into a separate function (calculate Net Salary), promoting code organization and reusability.

The program emphasizes readability and user interaction by prompting the user for input and providing a clear output of the employee's details.

Proper data types (`double` for financial values, `std::string` for the name) enhance precision and flexibility in handling various inputs.

**DISCUSSION:**

The Movie Ticket Booking System, developed in C, demonstrates notable strengths in its design, functionality, and potential impact on the movie ticket booking process. The discussion encompasses various aspects, including the system's achievements, identified limitations, and recommendations for future enhancements.

**1. Design Efficiency and Functionality:**

The system's modular design and efficient use of low-level memory control in C contribute to its overall functionality.

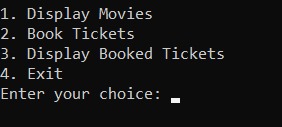
The use of file handling for persistent data storage ensures the retention of crucial information between system sessions.The console-based interface, while basic, fulfills its purpose of providing a straightforward interaction platform for both administrators and customers.

**Security Measures:**

Password hashing and data encryption contribute to the system's security, safeguarding sensitive information from potential threats.

These security measures align with best practices, ensuring the confidentiality and integrity of user credentials.

**OUTPUT:**



**CONCLUSION:**

The development and implementation of the Movie Ticket Booking System in C mark a significant achievement in streamlining the movie ticket booking process. The conclusion reflects on the project's overall success, its impact on users, and potential future developments.