

**U23CS301- Problem Solving Using C**

**Project Report**

MOVIE TICKET BOOKING

Submitted By

CSE268 - VISHNU SURYA

CSE267 - VISHAL

CSE214 - SARAVANA GUHAN

CSE219 - SELVA PRIYAN

CSE220 – SENTAMIL SELVAN

PROBLEM STATEMENT

**MOVIE TICKET BOOKING**

**Problem Statement: Movie Ticket Booking System Development in C**

The problem is to design and implement a movie ticket booking system where users can view available movies, book tickets for their preferred movie, cancel bookings, and check the availability of tickets. The system should display a list of movies with the number of tickets available for each. Users can select a movie, specify the number of tickets they wish to book, and provide their name to complete the booking. The system should ensure that tickets are only booked if the desired number of tickets are available. If the user needs to cancel a booking, they should provide the booking reference, and the system should return the tickets to the movie’s available pool. Additionally, the system should handle incorrect inputs and display appropriate error messages. The movie ticket booking system should function as a menu-driven application where users can choose between viewing available movies, booking tickets, canceling bookings, or exiting the program. The booking information (movie name, number of tickets, customer name) should be stored and displayed as needed, with a focus on maintaining accurate ticket availability for each movie.

**Introduction**

**C programming Language**

C is an [imperative](https://en.wikipedia.org/wiki/Imperative_programming) [procedural](https://en.wikipedia.org/wiki/Procedural_programming) language, supporting [structured programming](https://en.wikipedia.org/wiki/Structured_programming), [lexical variable scope](https://en.wikipedia.org/wiki/Lexical_variable_scope), and [recursion](https://en.wikipedia.org/wiki/Recursion_(computer_science)), with a [static type system](https://en.wikipedia.org/wiki/Static_type_system). It was designed to be [compiled](https://en.wikipedia.org/wiki/Compiled) to provide [low-level](https://en.wikipedia.org/wiki/Low-level_programming_language) access to [memory](https://en.wikipedia.org/wiki/Computer_memory) and language constructs that map efficiently to [machine instructions](https://en.wikipedia.org/wiki/Machine_instructions), all with minimal [runtime support](https://en.wikipedia.org/wiki/Runtime_system). Despite its low-level capabilities, the language was designed to encourage cross-platform programming. A [standards](https://en.wikipedia.org/wiki/Specification_(technical_standard))-compliant C program written with [portability](https://en.wikipedia.org/wiki/Software_portability) in mind can be compiled for a wide variety of computer platforms and operating systems with few changes to its source code.

Since 2000, C has consistently ranked among the top four languages in the [TIOBE index](https://en.wikipedia.org/wiki/TIOBE_index), a measure of the popularity of programming languages

**Software Requirement**

|  |  |
| --- | --- |
| **Category** | **Requirement** |
| Development Tools | Code::Blocks, Dev-C++, Eclipse CDT, or Visual Studio Code |
| Compiler | GCC, MinGW, or Clang |
| Debugger | GDB (GNU Debugger) |
| Operating System | Windows 7 or later, Linux (Ubuntu/Fedora), macOS |
| Libraries | Standard C libraries: stdio.h, stdlib.h, string.h |
| Storage | Disk space for program files and data files |
| Memory | Minimum 4 GB RAM |
| Optional Tools | Git, GitHub/GitLab, Doxygen, Testing frameworks (Check, CUnit) |

**Design**

**Use case Diagram**

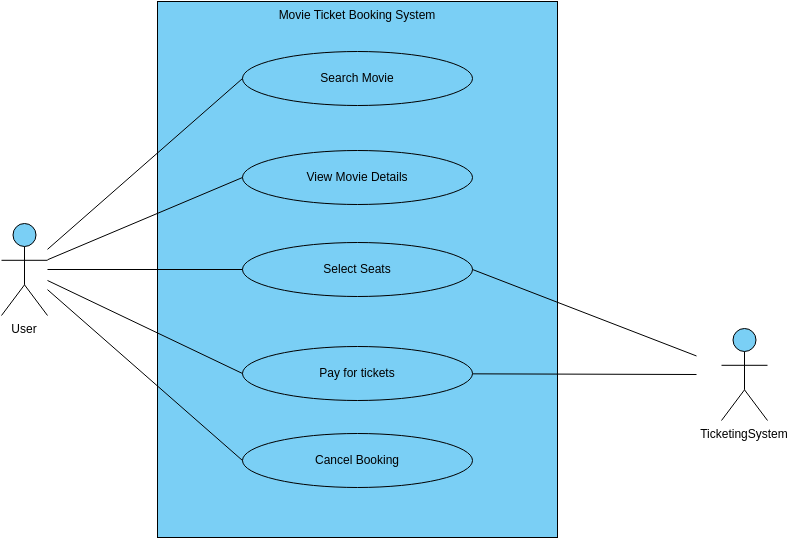


Fig : **use case diagram for movie ticket booking system**

**Design**

**Data Flow Diagram**

1. Data Flow Diagram (DFD)

Below is a simple Data Flow Diagram for your cinema ticket booking system:

Entities:

User: The person who interacts with the system to book a movie ticket.

Processes:

View Available Movies: Displays available movies and their details.

Select Movie and Show Timing: User selects a movie and show timing.

Select Seats and Book: User selects seats and books them.

Payment: User makes the payment for the selected seats.

Generate Ticket: Final step to generate the movie ticket.

Data Stores:

Movie Data: Stores information about movies (title, price, details, seats available).

Booking Details: Stores the selected movie, seats, and payment status.

Data Flows:

Movie Data: Flows from the Movie Data store to the View Available Movies process.

Movie Selection: Flows from the User to the Select Movie and Show Timing process.

Seat Booking: Flows from the User to the Select Seats and Book process.

Payment: Flows from the User to the Payment process.

Ticket: Flows from the Generate Ticket process to the User.**Design**

**Flow Chart**

**| Start Booking Process |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Display Available Movies |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| User Chooses Movie |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Display Movie Details |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Select Show Timing |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Select Number of Seats |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Are Seats Available? | ---> No ---> Display Error**

**|**

**Yes**

**|**

**V**

**+----------------------------+**

**| Calculate Total Cost |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Input Seat Numbers |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Proceed to Payment |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| Payment Successful? | ---> No ---> Display Error**

**|**

**Yes**

**|**

**V**

**+----------------------------+**

**| Display Ticket |**

**+----------------------------+**

**|**

**V**

**+----------------------------+**

**| End Booking Process |**

**+----------------------------+**

**Design**

*pseudocode*

**STEP 1: START**

**STEP 2: DISPLAY welcome message**

**STEP 3: WHILE user does not exit**

**STEP 4: DISPLAY available movies**

**STEP 5: INPUT movie choice**

**STEP 6: IF movie choice is valid**

**STEP 7: DISPLAY movie details**

**STEP 8: SELECT show timing**

**STEP 9: INPUT number of seats**

**STEP 10: IF seats are available**

**STEP 11: DECREASE available seats**

**STEP 12: CALCULATE total cost**

**STEP 13: INPUT seat numbers**

**STEP 14: DISPLAY booking confirmation**

**STEP 15: IF payment successful**

**STEP 16: DISPLAY ticket details**

**STEP 17: ELSE**

**STEP 18: DISPLAY payment failure**

**STEP 19: ELSE**

**STEP20: DISPLAY seat availability error**

**STEP 21: ELSE**

**STEP 22: DISPLAY invalid movie choice**

**STEP 23: END WHILE**

**STEP 24: DISPLAY thank you message**

**STEP 25: END**

**Module**

A movie ticket booking project involves several essential modules to ensure smooth functionality. The **User Management Module** handles user registration, login, profile management, and role-based access for customers and admins. The **Movie Management Module** enables admins to manage movies, including adding, updating, and deleting showtimes, and storing movie details like title, genre, and duration. The **Seat Selection Module** provides an interactive interface for users to view seat layouts, check availability, and select seats dynamically. The **Ticket Booking Module** allows users to browse available showtimes, reserve and confirm seats, and complete the booking process. To handle financial transactions, the **Payment Gateway Integration Module** facilitates secure payment processing with support for multiple payment methods. Additionally, the **Notification and Email Module** ensures users receive confirmations, reminders, and updates via email or SMS. A **Reporting and Analytics Module** provides admins with insights into booking trends and revenue. Technologies like React.js or Angular for the frontend, Django, Flask, or Express.js for the backend, and databases like MySQL or MongoDB can be used to build this system.

**Implementation**

#include <stdio.h>

#include <string.h>

// Structure for a Movie

typedef struct {

char title[50];

char details[200];

int price;

int seatsAvailable;

int screenNumber;

} Movie;

// Functions

void displaySeparator() {

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

}

void displayWelcomeMessage() {

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

printf(" WELCOME TO THE GRAND VETRI CINEMAS BOOKING\n");

printf(" ADDRESS: [1, Melur Main Rd, TTC Nagar, Industrial Estate, Uthangudi, Madurai, Tamil Nadu 625020]\n");

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

}

void displayAvailableMovies(Movie movies[], int size) {

displaySeparator();

printf("Movies available today:\n");

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

printf("%d. %s (Rs. %d)\n", i + 1, movies[i].title, movies[i].price);

}

displaySeparator();

}

void viewMovieDetails(Movie movie) {

displaySeparator();

printf("Movie Details:\n");

printf("Title : %s\n", movie.title);

printf("Description : %s\n", movie.details);

printf("Price per ticket: Rs. %d\n", movie.price);

printf("Screen Number : %d\n", movie.screenNumber);

printf("Show Timings :\n");

printf(" 1. 9:00 AM - 12:30 PM\n");

printf(" 2. 2:00 PM - 5:30 PM\n");

printf(" 3. 7:00 PM - 10:30 PM\n");

printf("Seats Available: %d\n", movie.seatsAvailable);

displaySeparator();

}

int selectShowTiming() {

int timing;

printf("Select your preferred show timing:\n");

printf(" 1. 9:00 AM - 12:30 PM\n");

printf(" 2. 2:00 PM - 5:30 PM\n");

printf(" 3. 7:00 PM - 10:30 PM\n");

printf("Enter your choice (1, 2, or 3): ");

scanf("%d", &timing);

if (timing < 1 || timing > 3) {

printf("Invalid choice! Defaulting to Show 1 (9:00 AM - 12:30 PM).\n");

timing = 1;

}

return timing;

}

void selectSeats(Movie \*movie, int \*totalCost, int \*tickets, int \*seatNumbers) {

int seats;

displaySeparator();

printf("Enter the number of seats you want to book: ");

scanf("%d", &seats);

if (movie->seatsAvailable >= seats) {

movie->seatsAvailable -= seats;

\*totalCost = seats \* movie->price;

\*tickets = seats;

printf("Enter the seat numbers (space-separated): ");

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

scanf("%d", &seatNumbers[i]);

}

printf("\nBooking successful! You have booked %d seats.\n", seats);

printf("Total cost: Rs. %d\n", \*totalCost);

} else {

printf("Sorry, only %d seats are available!\n", movie->seatsAvailable);

\*totalCost = 0;

\*tickets = 0;

}

displaySeparator();

}

void payForTickets(int totalCost, int \*isPaid) {

displaySeparator();

if (totalCost > 0) {

printf("Payment successful! Amount paid: Rs. %d\n", totalCost);

\*isPaid = 1;

} else {

printf("No tickets to pay for!\n");

\*isPaid = 0;

}

displaySeparator();

}

void displayTicket(const char \*theaterName, const char \*address, Movie movie, int tickets, int \*seatNumbers, int showTiming) {

const char \*showTimings[] = {"9:00 AM - 12:30 PM", "2:00 PM - 5:30 PM", "7:00 PM - 10:30 PM"};

displaySeparator();

printf(" \*\*\* YOUR TICKET \*\*\*\n");

printf("Theater Name : %s\n", theaterName);

printf("Address : %s\n", address);

printf("Movie : %s\n", movie.title);

printf("Screen : %d\n", movie.screenNumber);

printf("Show Timing : %s\n", showTimings[showTiming - 1]);

printf("Number of Tickets: %d\n", tickets);

printf("Seat Numbers : ");

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

printf("%d ", seatNumbers[i]);

}

printf("\n\nThank you for choosing Grand Vetri Cinemas!\n");

displaySeparator();

}

int main() {

// Movie data

Movie movies[] = {

{"Vadachennai", "An action drama about a gangster's life.", 200, 50, 1},

{"Maharaja", "A period drama with rich visuals.", 300, 30, 2},

{"The Conjuring", "A horror movie based on real events.", 150, 40, 3}

};

int numMovies = sizeof(movies) / sizeof(Movie);

// Booking details

int movieIndex = -1, totalCost = 0, tickets = 0, showTiming = 1;

int seatNumbers[10]; // Can hold up to 10 seat numbers for simplicity

int isPaid = 0;

// Theater details

const char theaterName[] = " The Grand Vetri Cinemas";

const char address[] = "1, Melur Main Rd, TTC Nagar, Industrial Estate, Uthangudi, Madurai, Tamil Nadu 625020";

// Display welcome message

displayWelcomeMessage();

// Main flow

while (1) {

displayAvailableMovies(movies, numMovies);

printf("Enter the number of the movie you'd like to book (or 0 to exit): ");

scanf("%d", &movieIndex);

if (movieIndex == 0) {

printf("Thank you for visiting Grand Vetri Cinemas! Goodbye!\n");

break;

} else if (movieIndex < 1 || movieIndex > numMovies) {

printf("Invalid choice! Please select a valid movie.\n");

continue;

}

Movie \*selectedMovie = &movies[movieIndex - 1];

viewMovieDetails(\*selectedMovie);

// Select show timing

showTiming = selectShowTiming();

printf("\nLet's proceed with your booking.\n");

// Select seats

selectSeats(selectedMovie, &totalCost, &tickets, seatNumbers);

// Pay for tickets

if (tickets > 0) {

payForTickets(totalCost, &isPaid);

}

// Display ticket

if (isPaid) {

char confirmation[10];

printf("Enter 'OK' to print your ticket: ");

scanf("%s", confirmation);

if (strcmp(confirmation, "OK") == 0 || strcmp(confirmation, "ok") == 0) {

displayTicket(theaterName, address, \*selectedMovie, tickets, seatNumbers, showTiming);

} else {

printf("Invalid input. Exiting...\n");

}

}

printf("\nThank you for using Grand Vetri Cinemas Booking System!\n");

break;

}

return 0;

}

Output

WELCOME TO LATHANGI CINEMAS BOOKING   
 ADDRESS: [1, venkatakrishna road,main street,udumalpet, Tamil Nadu 625020]  
 ---------------------------------------------------------------------------------------------------  
  
  
 ---------------------------------------------

Movies available today:

1. Vadachennai (Rs. 200)
2. Maharaja (Rs. 300)
3. The Conjuring (Rs. 150)

---------------------------------------------

Enter the number of the movie you'd like to book (or 0 to exit): 1

---------------------------------------------

Movie Details: Title : Vadachennai Description : An action drama about a gangster's life. Price per ticket: Rs. 200 Screen Number : 1 Show Timings :

1. 9:00 AM - 12:30 PM
2. 2:00 PM - 5:30 PM
3. 7:00 PM - 10:30 PM Seats Available: 50

Select your preferred show timing:

1. 9:00 AM - 12:30 PM
2. 2:00 PM - 5:30 PM
3. 7:00 PM - 10:30 PM Enter your choice (1, 2, or 3): 3

Let's proceed with your booking.

---------------------------------------------

Enter the number of seats you want to book: 2 Enter the seat numbers (space-separated): 1 2

Booking successful! You have booked 2 seats. Total cost: Rs. 400

---------------------------------------------  
  
 ---------------------------------------------

Payment successful! Amount paid: Rs. 400

---------------------------------------------

Enter 'OK' to print your ticket:

OK

---------------------------------------------  
 \*\*\* YOUR TICKET \*\*\*

Theater Name : The lathagi Cinemas Address : 1, venkatakrishna road,main street,Udumalpet, Tamil Nadu 625020 Movie : Vadachennai Screen : 1 Show Timing : 7:00 PM - 10:30 PM Number of Tickets: 2 Seat Numbers : 1 2

Thank you for choosing Grand Vetri Cinemas!

---------------------------------------------

Thank you for using Grand Vetri Cinemas Booking System!

**Conclusion**

The **Movie Ticket Booking System** simplifies the process of reserving tickets by providing an efficient, user-friendly platform that caters to both users and administrators. It streamlines the entire ticketing process, from selecting a movie and viewing showtimes to confirming bookings and making payments. By leveraging modern technology, the system minimizes errors, enhances user experience, and reduces operational overhead.