CINEMA BOOKING SYSTEM DOCUMENTATION

Contents  
1. Developer............................................................................................1

2. Introduction.........................................................................................2

3. Method and Implementation.................................................................3

4. Function Table......................................................................................4

1. Developer

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Email** |
| Full-stack Developer | Anna Rikova | [AARikova22@codingburgas.bg](mailto:AARikova22@codingburgas.bg) |

1. Introduction

|  |  |  |
| --- | --- | --- |
| **№** | **Question** | **Answer** |
| 1 | What is the product? | A simple console-based C++ application for managing a cinema: movies, shows, and ticket bookings. Designed as a course project for System Programming or Object-Oriented Programming classes. |
| 2 | Where is it available? | The project took place in GitHub and in order for the files to be accessible to everyone they were uploaded in the GitHub Repository of project.  Link - <https://github.com/codingburgas/movie-ticket-booking-system-10th-grade-AARikova22> |
| 3 | How will the system make sure that multiple users do not book the same seat? | The system checks the isReserved status of each seat at the moment of booking. If the seat is already marked as reserved, it cannot be selected again. |
| 4 | Will there be a timeout session that reserves seats temporarily? | No, timeout sessions are not implemented. Seats are only reserved after booking is confirmed. |
| 5 | Will the system use a first come, first serve algorithm? | Yes. The first user to confirm a booking gets the seat. Others will see it as unavailable. |
| 6 | Will there be transaction locks involved in the system? | No. Since this is a console-based single-user system, transaction locking is not required. |
| 7 | What payment methods can the customer use (for example, credit card or cash)? | The customer can use both payment methods. |
| 8 | How is the payment performed? Does the customer pay themselves online or through a ticket agent on the location? | First, the user selects a specific movie showing in a particular hall. Then, they choose how many seats they want to book, select the seats from the list, and choose a payment method. After the user completes the payment, they receive the corresponding tickets, and the selected seats are marked as occupied. |
| 9 | How will the price of the booking be calculated? Will it vary based on the popularity of the show? | Price does not change based on popularity. All pricing is static and seat-type-based. |
| 10 | Does the seat type affect the pricing? | Yes. There are three seat categories with different prices: silver, gold, platinum. |
| 11 | Will there be discount coupon codes? | No, the prices are static. |
| 12 | How are we handling these instances, such as the same cinema having multiple cinema halls showing different movies simultaneously? | The Cinema class contains vector halls. Each Hall object has its own shows. |
| 13 | Is the same movie being shown at different times in the same cinema/hall? | Yes. The system supports multiple shows of the same movie at different times, in the same and different halls. |

1. Method and Implementation

|  |  |
| --- | --- |
| **№** | **Description** |
| 1 | Development: Implemented using object-oriented programming principles in C++. Used classes such as Admin, Cinema, Movie, Hall, Show, and Seat. Each class manages a specific part of the system. |
| 2 | Task Management: Tasks were broken down into features: movie management, show scheduling, search filtering, ticket generation, and admin interface. |
| 3 | Workflow: The application was developed and tested in Visual Studio using MSVC. Each major function was tested through console input/output during development. |
| 4 | Testing: Manual testing was used by simulating user input in console scenarios (e.g. adding a show, searching for a movie, booking a ticket). |

1. Function Table

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| Admin::login() | Method | Authenticates admin by verifying username and password. |
| getIntegerInput() | Utility Function | Validated integer input handler for safe cin >>. |
| Cinema::printallshowsInHalls() | Method | Prints all shows scheduled in all cinema halls. |
| Cinema::SearchShow(...) | Method | Searches shows by movie title (case-insensitive). |
| Cinema::SearchShowbylanguage(...) | Method | Searches shows by movie language. |
| Cinema::SearchShowbygenre(...) | Method | Searches shows by movie genre. |
| Cinema::SearchShowbyreleasedate(...) | Method | Searches shows by release date. |
| Cinema::addMovie() | Method | Adds a new movie to the cinema’s movie list. |
| Cinema::deleteMovie() | Method | Deletes a movie and removes all related shows. |
| Cinema::addShow() | Method | Adds a show to a selected hall with seat setup. |
| Cinema::deleteShow() | Method | Deletes a specific show by index. |
| Cinema::updateShow() | Method | Updates the showtime for an existing show. |
| Hall::printallshows() | Method | Prints all shows scheduled in the current hall. |
| Seat::isreserved() | Method | Marks the seat as reserved if not already, returns success. |
| Seat::printseatinfo() | Method | Prints seat type, price, and reservation status. |
| Show::printMovieInfo() | Method | Prints the title and time of the show. |
| Show::printSeatsInfo() | Method | Lists all seats with their current status. |
| Show::seatReservation(int) | Method | Reserves a seat by index; returns true if successful. |
| Show::unreserveSeat(int) | Method | Manually unreserves a seat (used when booking fails). |
| adminMenu(Cinema&) | Menu Function | Displays and handles the admin menu options. |
| searchMenu(Cinema&) | Menu Function | Provides options to search shows by various filters. |
| displayResultsAndBook(...) | Controller | Displays search results and handles show selection for booking. |
| processBooking(...) | Controller | Manages seat selection, payment, validation, and confirmation. |
| printTicket(...) | Output Function | Prints a formatted ticket for a reserved seat. |