Project Documentation

Movie Ticket Booking System Documentation

1-Project Overview

Description:

The Movie Ticket Booking System is a multithreaded application that allows users to book movie tickets concurrently while ensuring thread safety and synchronization. It supports admin operations like adding movies, resetting seat statuses, and deleting movies. The GUI is responsive and separate threads are used to avoid UI hanging.

Key Features

Multithreading for concurrent booking by users.
Admin operations like adding, resetting, and deleting movies.
Database integration for persistent data storage.
Thread safety and synchronization using locks.
User-friendly GUI.

2. System Architecture

Modules

admin_gui.py:

Handles admin operations like adding movies, resetting seat statuses, and deleting movies.
booking_system.py:

Core logic for managing bookings, seat statuses, and thread safety.

gui.py:

User interface for users to select movies, seats, and book tickets.
db_connection.py:

Database connection and utility functions.
test_threading.py:

Demonstrates multithreaded booking with multiple GUI threads.

performance.py:

Measures performance improvement using multithreading.

3. Code Documentation

admin_gui.py

Class: AdminGUI
Handles admin-specific operations.

init(self, root):

Initializes the admin GUI.
Parameters:
root: The root Tkinter window.
Returns: None.

show_admin_options(self):

Displays main admin options.
Returns: None.

clear_frame(self):

Clears the current frame in the GUI.
Returns: None.

add_movie_screen(self):

Displays the interface to add a new movie.
Returns: None.

add_movie_to_db(self):

Adds a new movie to the database.
Returns: None.

reset_seats_screen(self):

Displays the interface to reset seat statuses.
Returns: None.

reset_seats_in_db(self):

Resets seat statuses for a selected movie.
Returns: None.

view_bookings_screen(self):

Displays bookings for a selected movie.
Returns: None.

view_bookings_in_db(self):

Fetches and displays bookings from the database.
Returns: None.

delete_movie_screen(self):

Displays the interface to delete a movie.
Returns: None.

delete_movie_from_db(self):

Deletes a movie and its associated data from the database.
Returns: None.

switch_to_user_view(self):

Switches to the user view.
Returns: None.

booking_system.py

Class: BookingSystem
Manages core booking logic and thread safety.

init(self, rows, cols, show_id):

Initializes the booking system with seats and synchronization.
Parameters:
rows: Number of rows in the theater.
cols: Number of columns in the theater.
show_id: The ID of the movie show.
Returns: None.

fetch_seating_layout(self):

Fetches seat layout from the database.
Returns: A 2D list representing the seating layout.

display_seats(self):

Prints the seating layout.
Returns: None.

book_seat(self, user_id, row, col):

Books a seat for a user.
Parameters:
user_id: The ID of the user.
row: Row number of the seat.
col: Column number of the seat.
Returns: True if booking is successful, False otherwise.

random_booking(self, user_id):

Simulates random booking by a user.
Parameters:
user_id: The ID of the user.
Returns: None.

simulate_user(booking_system, num_users):

Simulates multiple users booking concurrently.
Parameters:
booking_system: Instance of the booking system.
num_users: Number of users.
Returns: None.

gui.py

Class: BookingSystemGUI
Handles user interface for ticket booking.

init(self, root, booking_system):

Initializes the user GUI.
Parameters:
root: The root Tkinter window.
booking_system: Instance of BookingSystem.
Returns: None.

show_movie_selection(self):

Displays available movies and showtimes.
Returns: None.
refresh_movie_list(self):

Refreshes the list of available movies and showtimes.
Returns: None.

proceed_to_seating(self):

Proceeds to seat selection for the selected movie.
Returns: None.

show_seating_layout(self):

Displays the seating layout.
Returns: None.

return_to_movie_selection(self):

Returns to the movie selection screen.
Returns: None.

book_seat(self, row, col):

Handles seat booking for a user.
Parameters:
row: Row number of the seat.
col: Column number of the seat.
Returns: None.

update_seats(self):

Updates the GUI with the latest seat statuses.
Returns: None.

switch_to_admin_view(self):

Switches to the admin view.
Returns: None.

test_threading.py

Class: MainApp
Demonstrates multithreaded booking.

init(self, root):

Initializes the main app with thread count input.
Parameters:
root: The root Tkinter window.
Returns: None.

start_threads(self):

Starts multiple threads for concurrent user GUIs.
Returns: None.

launch_gui(self):

Launches a new GUI for each thread.
Returns: None.

performance.py

single_threaded_simulation(booking_system, num_users):

Simulates booking in a single-threaded manner.
Parameters:
booking_system: Instance of BookingSystem.
num_users: Number of users.
Returns: Time taken for the simulation.

multithreaded_simulation(booking_system, num_users):

Simulates booking in a multithreaded manner.
Parameters:
booking_system: Instance of BookingSystem.
num_users: Number of users.
Returns: Time taken for the simulation.

benchmark_simulation(booking_system, num_users, num_runs=5):

Benchmarks single-threaded and multithreaded simulations.
Parameters:
booking_system: Instance of BookingSystem.
num_users: Number of users.
num_runs: Number of simulation runs.
Returns: Average times for single-threaded and multithreaded simulations.

4. Conclusion

The Movie Ticket Booking System effectively implements multithreading, thread safety, and GUI responsiveness. It achieves the project requirements, ensuring a user-friendly and scalable booking system.