BUG BUSTERS

Shankramma .K-3BR23EC150

Vandana.K-3BR23EC179

U. Preeti dharani-3BR23EC173

T.Sahana-3BR23EC163

V.Hema Nandini-3BR23EC180


STADIUM
SEATING
MANAGEMENT

Introduction:-

Stadium seating management systems are used to handle the allocation, booking, and management of seats in a stadium for various events like sports games, concerts, or performances. These systems must consider factors like seat availability, different seating sections (VIP, regular, student sections), pricing tiers, and sometimes the unique layout of the stadium. In code, this can be achieved by modeling the stadium as a grid or list of sections, rows, and seats, tracking each seat's status (e.g., available, reserved, sold)

Problem statement:-

1. Inefficient seat allocation leading to underutilization of stadium capacity.
2. Difficulty in balancing customer preferences, such as group seating and proximity, with maximizing revenue.
3. Inability to handle last-minute seat changes, cancellations, or upgrades smoothly.
4. Compliance with safety regulations, including accessibility and social distancing protocols.
5. Revenue loss due to suboptimal pricing strategies or unsold seats.
6. The risk of booking errors, such as double-booking or assigning unavailable seats.

OBJECTIVES:-

1. View the current seating arrangement:

a . A seat marked with 0 is available.

b. A seat marked with 1 is reserved.

2. Reserve a seat by specifying the row and column number, provided the seat is available.

3. Cancel a seat reservation by specifying the row and column number, if the seat is currently reserved.

4. Check the availability of a specific seat by entering the row and column number.

Algorithm:-

1. Initialize Seating : Take rows and cols as input.

2. Display Menu : Continuously show the following options until the user exits:

1. Display seating arrangement.
2. Reserve a seat
3. Cancel a reservation.
4. Check seat availability.
5. Exit.

3. Perform Actions Based on User Choice:

- Display seating: Print the current seats grid.
- Reserve seat : Input row and col.
- Check if seat is within valid range.
- If seat is available (0), set it to 1 (reserved).
- Else, inform the user that the seat is already reserved.
- Cancel reservation : Input row and col . Check if seat is within valid range . If seat is reserved (1), set it to 0 (available).Else, inform the user that the seat is not reserved.
- Check availability : Input row and col . If seat is within valid range, display whether it is available or reserved.

4. Exit : End the program when the user selects the exit option.

CRUD operation:-

1.Read:

display_seating() reads and displays the entire seating arrangement.check_availability(row, col) reads the status of a specific seat.

2.Update:

- reserve_seat(row, col) updates the seat status from available to reserved.
- cancel_reservation(row, col) updates the seat status from reserved to available.

3.Delete:

- cancel_reservation(row, col) removes a reservation, marking the seat as available.
- These CRUD operations allow the system to manage the stadium seating dynamically.

```
Enter the number of rows: 100
Enter the number of columns: 150

Menu:
1. Display seating arrangement
2. Reserve a seat
3. Cancel a reservation
4. Check seat availability
5. Exit
Choose an option: 2
Enter row number to reserve: 5
Enter column number to reserve: 6
Seat (5, 6) reserved successfully.

Menu:
1. Display seating arrangement
2. Reserve a seat
3. Cancel a reservation
4. Check seat availability
5. Exit
Choose an option: 5
Exiting...
>>> |
```

Conclusion:-

- This implementation serves as a solid foundation for a seating management system. It could be further enhanced with additional features such as tracking total reserved seats, summarizing all reservations, and improving input validation for a more robust user experience. Overall, it effectively meets the basic requirements for managing stadium seating.

Future enhancement:-

Seat Block Reservation: - Allow users to reserve a block of seats in a rectangular pattern by specifying a start seat and an end seat. - Automatically ensure that all seats in the block are available before proceeding with the reservation.

Dynamic Pricing: - Assign different prices to seats based on their location (e.g., front rows are more expensive). Add a method to calculate the total price for reserved seats.


THANK
YOU