Department of CSE

Course Name: Operating Systems
Course Code: CSE325
Section No: 01
Group No: 09
Semester: Fall2022
Name of the Project Report: Online Ticket Booking System
Date of submission: 03/01/2023

Submitted By

Name	Id
Md. Abdul Ahad Rifat	2020-1-60-215
Md. Bayzid Hassan	2020-2-60-171
Sadia Alam Choity	2020-1-60-162

Submitted To

Dr. Md. Nawab Yousuf Ali
Professor
Department of Computer Science and Engineering
East West University

ONLINE TICKET BOOKING SYSTEM

ABSTRACT

This problem involves designing a solution that uses POSIX threads, mutex locks, and semaphores to synchronize the activities of users booking tickets for shows at a multiplex cinema theatre. The goal is to ensure that the total number of available seats is accurately maintained and there is no data loss during the booking process.

ITRODUCTION

Online Ticket Booking means the sale of tickets to events via online. Where customer get chance to show the number of tickets are available as well as they are capable to purchase a ticket by their own. On the other hand it helps the seller to sell more tickets and it makes works more efficient as well as it provides various kinds of features, functionalities, promotion, marketing and advertising. In this project, we are tasked with implementing a solution using POSIX threads, mutex locks, and semaphores to synchronize the activities of users booking tickets for shows at a multiplex cinema. The program must ensure that the total number of available seats is accurately reflected and that there is no data loss during the booking process.

RELATED WORKS

We try to build an online ticket booking system for Star Cineplex. Star Cineplex is the first multiplex cinema theatre in Bangladesh. It also gives its viewers the facility to check available tickets and book shows via online ticket booking system. Here it create certain number of threads to sell all available tickets.

Here we are use some technique to solve our program

- POSIX threads
- Mutex locks
- Semaphores

Threads

A thread is a unit of execution on concurrent programming. Multithreading is a technique which allows a CPU to execute many tasks of one process at the same time. These threads can execute individually while sharing their resources.

Mutex Locks

Mutex lock in OS is essentially a variable that is binary nature that provides code wise functionality for mutual exclusion. At times, there may be multiple threads that may be trying to access same resource like memory or I/O etc. To make sure that there is no overriding. Mutex provides a locking mechanism.

Semaphores

A semaphore is an integer variable, shared among multiple processes. The main aim of using a semaphore is process synchronization and access control for a common resource in a concurrent environment.

PROPOSED WORKS

The program first initializes the mutex and semaphore, then creates the user threads and waits for them to finish. Finally, it cleans up and exits.

The user threads execute the user define function, which attempts to book a ticket by first acquiring the mutex lock and then checking if there are any tickets available. If there are, it books a ticket and decrements the number of available tickets. Otherwise, it prints a message saying that it was unable to book a ticket. The thread then releases the mutex lock and increments the semaphore before exiting.

The main thread waits for all of the user threads to finish by calling sem wait for each thread. This ensures that all of the threads have completed their work before the main thread cleans up and exits.

EXPERIMENTAL RESULT

```
rifat@rifat-ubuntu-linux:~/Downloads$ ./a.out
Input for Number of Users : 12
Input for Number of Tickets : 13
Input for Number of Shows : 2
Thread 4: Booking ticket...
Thread 1: Booking ticket...
Thread 2: Booking ticket...
Thread 3: Booking ticket...
Thread 12: Booking ticket...
Thread 5: Booking ticket...
Thread 6: Booking ticket...
Thread 7: Booking ticket...
Thread 11: Booking ticket...
Thread 10: Booking ticket...
Thread 8: Booking ticket...
Thread 9: Booking ticket...

------------------------------------------------------------
Do you want to exit the System ?
 1.Yes
 2.No
Enter the value : 2

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

Input for Number of Users : 5
Input for Number of Tickets : 2
Input for Number of Shows : 5
Thread 3: Booking ticket...
Thread 2: Booking ticket...
Thread 1: Not Booking ticket...
Thread 4: Not Booking ticket...
Thread 5: Not Booking ticket...

------------------------------------------------------------
Do you want to exit the System ?
 1.Yes
 2.No
Enter the value : 1

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

!!!Thank You!!!
rifat@rifat-ubuntu-linux:~/Downloads$
```

FUTURE WORK

1. Allow the user to cancel a booking
2. Implement a system for purchasing tickets online
3. Allow the user to choose their seats
4. Implement a loyalty program for repeat customers

CONCLUSION

Now a days everyone likes to do anything using online and online ticket booking system is one of them. Here we try to focus a build program to check number of seats are available or not for the viewers using threads. Also we use some other feature to avoiding a deadlock and those features are mutex locks and semaphores. At the end we try to build program where user can easily visualize the ticket is available or not.

CODE

C Language

```c
#include <pthread.h>
#include <semaphore.h>
#include <stdio.h>
#include <stdlib.h>

int num_users;
int num_tickets;
int num_shows;

pthread_mutex_t mutex;
sem_t semaphore;

void *book_show(void *arg)
{
    int thread_id = (int) arg;

    pthread_mutex_lock(&mutex);
    if (num_tickets > 0)
    {
        printf("Thread %d: Booking ticket...\n",(thread_id+1));
        num_tickets--;
    }
    else
    {
        printf("Thread %d: Not Booking ticket...\n",(thread_id+1));
    }
    pthread_mutex_unlock(&mutex);
    sem_post(&semaphore);

    pthread_exit(NULL);  // Terminate thread
}

int main()
{
    int value=0;
    do
    {

        printf("Input for Number of Users : ");
        scanf("%d",&num_users);
        printf("Input for Number of Tickets : ");
        scanf("%d",&num_tickets);
        printf("Input for Number of Shows : ");
        scanf("%d",&num_shows);


        // Initialize mutex and semaphore
        pthread_mutex_init(&mutex, NULL);
        sem_init(&semaphore, 0, 0);

        // Create user threads
        pthread_t threads[num_users];
        for (int i = 0; i < num_users; i++)
        {
            pthread_create(&threads[i], NULL, book_show, (void *)i);
        }

        // Wait for user threads to finish booking shows
        for (int i = 0; i < num_users; i++)
        {
            sem_wait(&semaphore);
        }

        // Clean up and exit
        pthread_mutex_destroy(&mutex);
        sem_destroy(&semaphore);
        printf("\n---------------------------------------------------------------\n");
        printf("Do you want to exit the System ?\n 1.Yes\n 2.No\nEnter the value : ");
        scanf("%d",&value);
        printf("\n---------------------------------------------------------------\n\n");

    }
    while(value!=1);
    printf("!!!Thank You!!!\n");
    return 0;
}

```