Lovely Professional University

S.no	Roll no.	Name	Reg no.
1	22	Saikat Chakraborty	12016061
2	23	Rupak Debnath	12017009
3	24	Md Shahriar Shojib	12018181

Submitted To	Surbhi Ma'am

CERTIFICATE

This is to certify that Dissertation titled Multi-Face Detection using machine learning that is being submitted by Saikat Chakraborty,Md Shahriar Shojib & Rupak Debnath is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bona-fide work done under my guidance. The contents of this dissertation, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Surbhi Ma'am Assistant Professor

School of Electronics and Electrical Engineering Lovely Professional University, Phagwara Punjab.

ACKNOWLEDGEMENT

Foremost, I would like to express my sincere gratitude towards my faculties, who gave their full support in the working of this dissertation work with their stimulating suggestions and encouragement all the time. They have always been a source of ideas.

I am very thankful to my guide Surbhi for her readiness to show me the right path. Above her highly intelligent and tactful mind is her warm and inspiring heart, which has always inspired and given a thrust to do the work as a master no matter however small the task may be. Her inspiration has made me accomplish my task with ease.

I am thankful to management of Lovely Professional University for giving me an opportunity to carry out studies at the university and providing the proper infrastructure like internet facility which is pool of vast knowledge to work in.

At last but not the least my gratitude towards my parents, I would like to thank God for the strength that keep me standing and for the hope that keep me believing that this report would be possible.

DECLARATION

We Saikat chakraborty, Md Shahriar Shojib & Rupak Debnath, students of B.Tech (EEE) under the Department of Electronics and Electronic Engineering   of Lovely Professional University, Punjab, hereby declare that all the information furnished in this dissertation report is based on my intensive research and is genuine. This dissertation is done to the best of my knowledge and does not contain any part of my work that has been submitted for the award of my degree either of this university or any other university without proper citation

Saikat Chakraborty : 12016061

Md Shahriar Shojib : 12018181

Rupak Debnath : 12017009

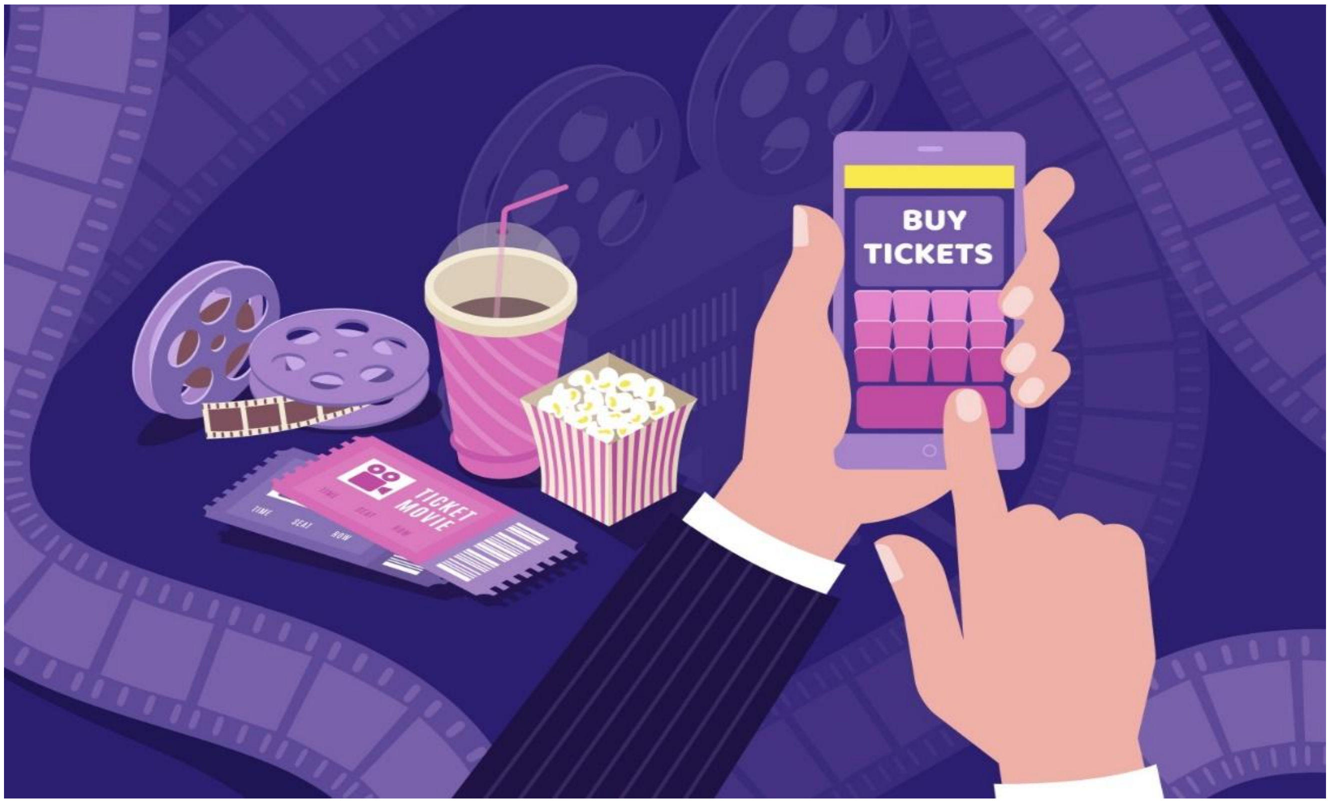
Abstract

The project objective is to book cinema tickets in online. The Ticket Reservation System is an Internet based application

that can be accessed throughout the Net and can be accessed by anyone who has a net connection. This application will reserve the tickets. This online ticket reservation system provides a website for a cinema hall where any user of internet can access it. User is required to login to the system and needs a credit card for booking the tickets. Tickets can be collected at the counter and Watching movies with family and friends in theatre is one of the best medium of entertainment after having a hectic schedule. But all this excitement vanishes after standing in hours in long queues to get tickets booked. The website provides complete information regarding currently running movies on all the screens with details of show timings, available seats. Ticket reservations are done using credit card and can be cancelled if needed. Our online tickets reservation system is one of the best opportunities for those who cannot afford enough time to get their tickets reserved standing in long queues. People can book tickets online at any time of day or night. Our reservation system also provides option to cancel the tickets which are reserved previously.

Table of Content

Movie Ticket Booking System



Introduction:

Movie ticket booking systems are designed to make the process of booking movie tickets more efficient, convenient and accessible for movie-goers. With the increasing popularity of online transactions, movie ticket booking systems have become an integral part of the movie-going experience. This report will outline the features, benefits, and potential drawbacks of a movie ticket booking system.

Summary:

In this project we book ticket using online movie Ticket reservation System. We enter into Web page by logging with User Name and Password.Then we select the Movie and later in which Theatre movie is running. Later choose Show Timings and enter no of tickets you want. Finally it displays the details of the procedure and print the form to show at respective ticket counter to get tickets.

Key Features:

User registration and login: Users should be able to create an account and log in to the system to save their preferences, view booking history, and receive updates on upcoming movies and events.

Movie information and reviews: The system should provide users with detailed information about each movie, including ratings, reviews, trailers, and cast and crew information.

Multiple payment options: The system should support various payment methods, such as credit cards, debit cards, and mobile wallets, to offer users flexibility in paying for their tickets.

Real-time availability: The system should allow users to select their seats from a seating map and see real-time availability for each showtime.

Email and SMS notifications: The system should send email or SMS notifications to users to confirm their booking, remind them of their showtime, and provide updates on any changes or cancellations.

Admin panel: An admin panel should be provided to manage the system, including adding new movies, updating showtimes, and managing user accounts and bookings.

Software used:

Compiler: Vs code

Language Used: Java

The Java programming language is widely used in the development of web-based applications, including movie ticket booking system

Function Used:

Constructor:

In Java, a constructor is a special method that has the same name as the class and is used to initialize the object's state (i.e., its instance variables or attributes) when an object of the class is created. A constructor can have parameters, and it can be overloaded (i.e., a class can have multiple constructors with different parameter lists).In a movie ticket booking system using Java, constructors are used to initialize objects of classes that represent entities in the system. Here are the use cases of constructors in our System:

File Handling:

File handling refers to the process of reading from and writing to files on a computer's file system. In programming, file handling is used to store and retrieve data in a persistent manner, meaning the data persists even after the program is closed.

In a movie ticket booking system project, file handling can be used to store and retrieve data related to movies, theaters, showtime's, and user bookings. This can help in persisting data even when the application is closed and reopened, as well as providing backup and recovery options.

Exception Handling:

Exception handling is a feature of the Java programming language that allows developers to handle errors and unexpected situations that can occur during program execution. An exception is an event that interrupts the normal flow of the program and can occur due to various reasons, such as incorrect user input, network failures, or file system errors.

Exception handling in Java involves three main steps:

Try Block: The code that may throw an exception is enclosed in a try block. If an exception occurs, the try block immediately terminates, and control is transferred to the catch block.

Catch Block: The catch block handles the exception that was thrown in the try block. It specifies the type

Modules:

- User
- Admin
- Movie

User Module:

The User Module consists of class user which contains the user's information,email id, password, username etc. It has functions to authenticate the user. After authentication, the user can make transactions for booking tickets.

Function Used:

1. File Handling

2. Exception Handling

Admin Module:

The admin module is for the administrator or the manager who has access to change the information of movies, screens, time slots etc. after authentication. This module contains the information of admin like username and functions for authentication, changing time slots, changing screen details like capacity, changing ticket prices (mentioned in transaction module).

Function Used:

1. Constructor

2. Exception Handling

Movie Module:

The "Movie module" could handle tasks related to managing movie information, such as adding new movies to the system, updating movie details, retrieving movie information, and deleting movies from the system.The "Movie module" could be responsible for managing the ticket booking process, including features such as selecting movie showtime, selecting seats, reserving seats, and processing payments for tickets. This could involve coordinating with other modules or components in the ticket booking system, such as a Payment module, Seat module, and User module, to complete the booking process.

Function Used:

1. File Handling.
2. Exception Handling.

Code Implementation:

```
import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

import java.util.Scanner;

import java.io.BufferedReader;

import java.io.FileReader;

import java.util.ArrayList;

import java.util.List;

import java.io.*;

import java.util.*;
```

```
class User {
    private List<List<String>> users;
    public User() {
        users = new ArrayList<>();
        try {
            BufferedReader reader = new BufferedReader(new FileReader("users.txt"));
            String line = reader.readLine();
            while (line != null) {
                String[] fields = line.split(",");
                if (fields.length >= 2) {
                    List<String> user = new ArrayList<>();
                    user.add(fields[0]);
                    user.add(fields[1]);
                    users.add(user);
                }
                line = reader.readLine();
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Error reading from file");
            e.printStackTrace();
        }
    }
    public void create_user(String username, String password) {
        List<String> user = new ArrayList<>();
        user.add(username);
        user.add(password);
        users.add(user);
        try {
            FileWriter writer = new FileWriter("users.txt", true);
            writer.write(username + "," + password + "\n");
            writer.close();
        } catch (IOException e) {
            System.out.println("Error writing to file");
            e.printStackTrace();
        }
    }

    public boolean authenticate_user(String username, String password) {
        for (List<String> user : users) {
            if (user.get(0).equals(username) && user.get(1).equals(password)) {
                return true;
            }
        }
        return false;
    }
}
class Movie {
    private String[][] movieDetails;
    private int count;


    public Movie() {
        movieDetails = new String[100][4]; // Initialize with space for 100 movies
        count = 0;
    }
    public void addMovie() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter movie name: ");
        String name = scanner.nextLine();
        System.out.print("Enter theater name: ");

        String theater = scanner.nextLine();

        System.out.print("Enter number of tickets: ");

        int numTickets = scanner.nextInt();

        System.out.print("Enter cost per ticket: ");

        double costPerTicket = scanner.nextDouble();



        movieDetails[count][0] = name;

        movieDetails[count][1] = theater;

        movieDetails[count][2] = Integer.toString(numTickets);

        movieDetails[count][3] = Double.toString(costPerTicket);



        count++;



        // Write to file

        try {

            BufferedWriter writer = new BufferedWriter(new FileWriter("movie_details.txt",
true));

            writer.write(name + "," + theater + "," + numTickets + "," + costPerTicket + "\n");

            writer.close();

        } catch (IOException e) {
            System.out.println("Error writing to file");

        }

    }

     public void displayMovies() {

        //System.out.println("Movie:\nTheater:\nTickets:\nCost per ticket:");

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

  System.out.println("Movie: " + movieDetails[i][0] + "\n" + "Theater: " +
movieDetails[i][1] + "\n" +"Tickets: " + movieDetails[i][2] + "\n" +"Cost per ticket: $"
+movieDetails[i][3]);

        }

    }

}

class Admin {

    private String username;

    private String password;

    private String filename;

    public Admin(String username, String password, String filename)

    {

        this.username = username;

        this.password = password;

        this.filename = filename;
    }

    public boolean login(String username, String password) {

        return (username.equals(this.username) && password.equals(this.password));

    }

    public boolean updateRecord(String movieName, String theaterName, int
numOfTickets, double ticketCost) {

        try {

            File file = new File(filename);

            Scanner scanner = new Scanner(file);

            List<String> lines = new ArrayList<String>();

            while (scanner.hasNextLine()) {

                String line = scanner.nextLine();

                String[] parts = line.split(",");

                if (parts[0].equals(movieName) && parts[1].equals(theaterName)) {

                    line = movieName + "," + theaterName + "," + numOfTickets + "," +
ticketCost;

                }

                lines.add(line);

            }

            scanner.close();

            FileWriter writer = new FileWriter(file);
            for (String line : lines) {
                writer.write(line + "\n");
            }
            writer.close();
            return true;
        } catch (IOException e) {
            e.printStackTrace();
            return false;
        }
    }
        public boolean deleteRecord(String movieName, String theaterName) {
        try {
            File file = new File(filename);
            Scanner scanner = new Scanner(file);
            List<String> lines = new ArrayList<String>();
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                String[] parts = line.split(",");
                if (!parts[0].equals(movieName) || !parts[1].equals(theaterName)) {
                    lines.add(line);
                }
            }
            scanner.close();
            FileWriter writer = new FileWriter(file);
            for (String line : lines) {
                writer.write(line + "\n");
            }
            writer.close();
            return true;
        } catch (IOException e) {
            e.printStackTrace();
            return false;
        }
    }
}
public class Finalproject {


    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        System.out.println("...Enter Your Choice...\n1.User Page\n2.Movie
booking\n3.Admin Page");
```

```
int select=sc.nextInt();

sc.nextLine();

if(select==1){

    User user = new User();

    boolean c=true;

    while(c==true){

        System.out.println("Enter 1 to Create User and 2 to Login");

        int choice=sc.nextInt();

        sc.nextLine();

        if(choice==1)

        {

            System.out.print("Enter Username: ");

            String newname=sc.nextLine();

            System.out.print("Enter Password: ");

            String newpass=sc.nextLine();

            user.create_user(newname, newpass);

        }

        else if(choice==2)

        {
```

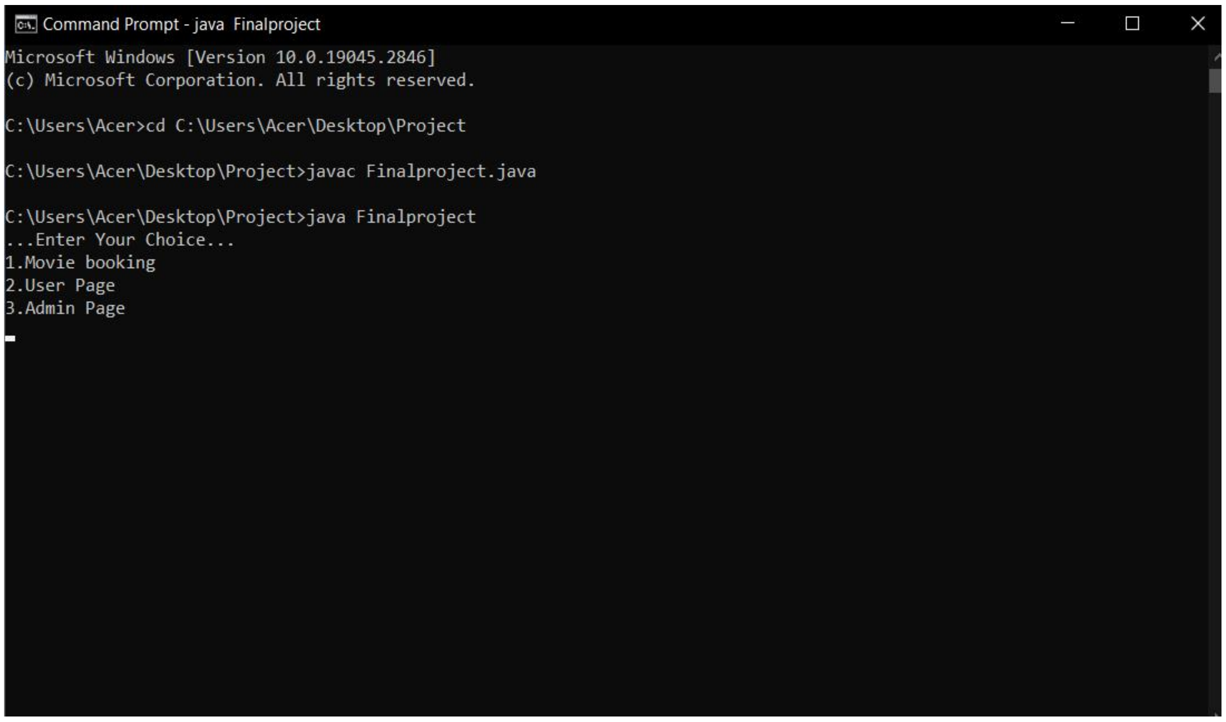
```
    System.out.print("Enter Username: ");

    String uname=sc.nextLine();

    System.out.print("Enter Password: ");

    String pass=sc.nextLine();

    // Authenticate a user

    if (user.authenticate_user(uname, pass)) {

        System.out.println("Authentication successful");

    } else {

        System.out.println("Authentication failed");

    }

}

else{

    System.out.println("wrong choice");

}

System.out.println("yould you like to continue? 1. Yes 2.No ");

int con=sc.nextInt();

if(con==1)

{

}

else
```

```
        {
            c=false;
        }
    }
}
else if(select==2){
    boolean c=true;
while(c==true)
 {
    Movie movie = new Movie();
    movie.addMovie();
    movie.displayMovies();
    System.out.println("Yould you like to continue? 1. Yes 2. No");
    int con=sc.nextInt();
    if(con==1)
    {
    }
    else{
        c=false;
    }
 }
}
else if(select==3){
    Admin admin = new Admin("admin12", "a123", "movie_details.txt");
    System.out.print("Enter Username: ");
    String Username =sc.nextLine();
    System.out.print("Enter Password: ");
    String Password = sc.nextLine();
    if (admin.login("admin12", "a123")) {
        System.out.println("....Enter Your Choice....\n1.Update \n2.Delete ");
        int x1=sc.nextInt();
        sc.nextLine();
        if(x1==1){
            System.out.print("Enter movie name: ");
            String name = sc.nextLine();
            System.out.print("Enter theater name: ");
            String theater = sc.nextLine();
            System.out.print("Enter number of tickets: ");
            int numTickets = sc.nextInt();
            System.out.print("Enter cost per ticket: ");
        double costPerTicket = sc.nextDouble();

        if (admin.updateRecord(name, theater, numTickets, costPerTicket)) {

            System.out.println("Record updated successfully");

        } else {

            System.out.println("Failed to update record");

        }

    }

    else if(x1==2){

        System.out.print("Enter movie name: ");

        String name = sc.nextLine();

        System.out.print("Enter theater name: ");

        String theater = sc.nextLine();

        if (admin.deleteRecord(name,theater )) {

            System.out.println("Record deleted successfully");

        } else {

            System.out.println("Failed to delete record");

        }

    }

   else{

        System.out.println("wrong choice");
```

```
                }

            } else {

                System.out.println("Invalid login credentials");

            }

        }

        else{

            System.out.println("wrong choice");}

    }

}
```

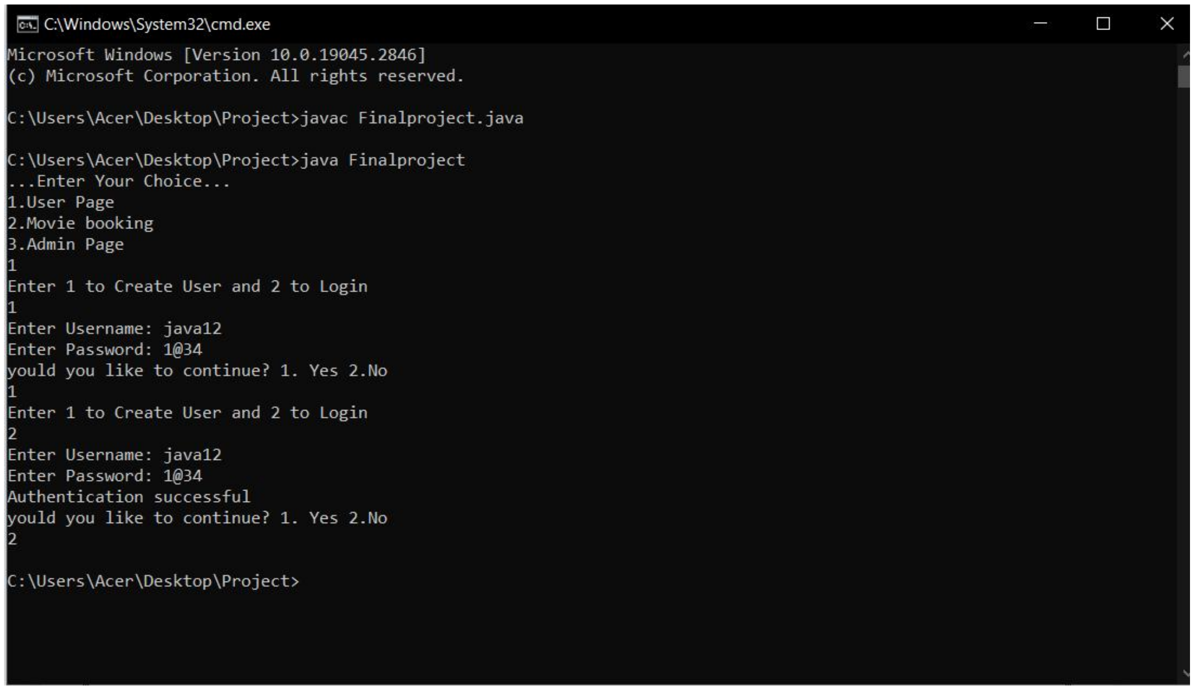
Output:

Starting interface:



```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Acer>cd C:\Users\Acer\Desktop\Project

C:\Users\Acer\Desktop\Project>javac Finalproject.java

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.Movie booking
2.User Page
3.Admin Page
```

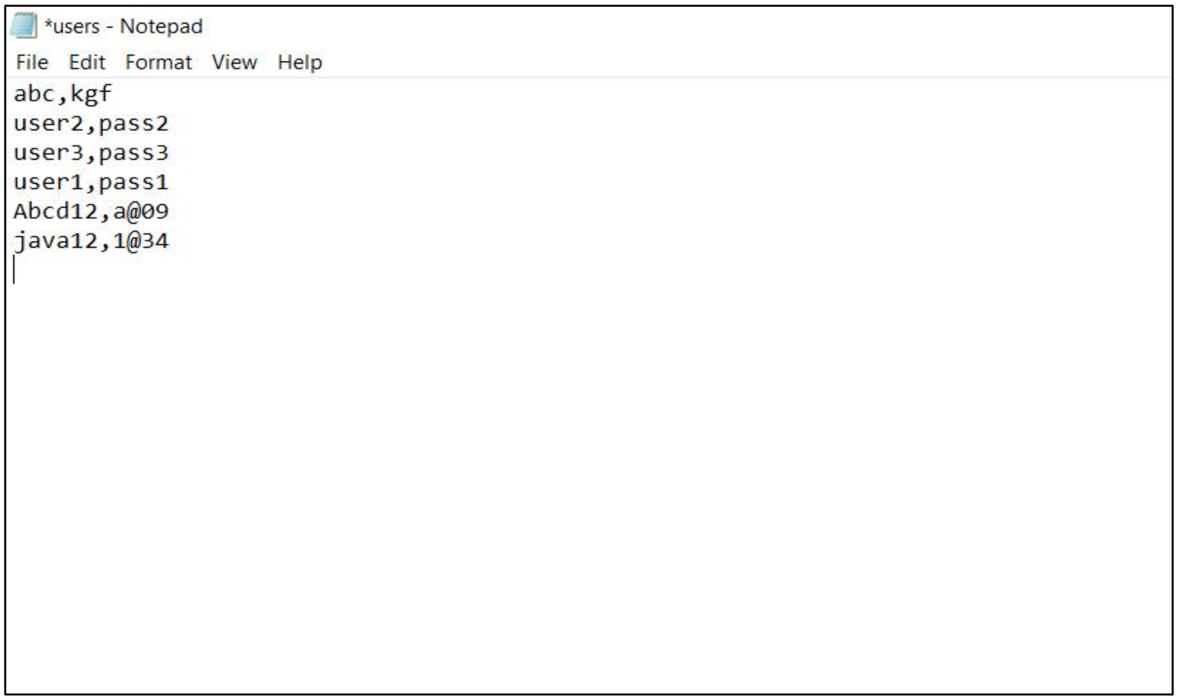
Output of User Page:



C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Acer\Desktop\Project>javac Finalproject.java

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.User Page
2.Movie booking
3.Admin Page
1
Enter 1 to Create User and 2 to Login
1
Enter Username: java12
Enter Password: 1@34
would you like to continue? 1. Yes 2.No
1
Enter 1 to Create User and 2 to Login
2
Enter Username: java12
Enter Password: 1@34
Authentication successful
would you like to continue? 1. Yes 2.No
2

C:\Users\Acer\Desktop\Project>
```

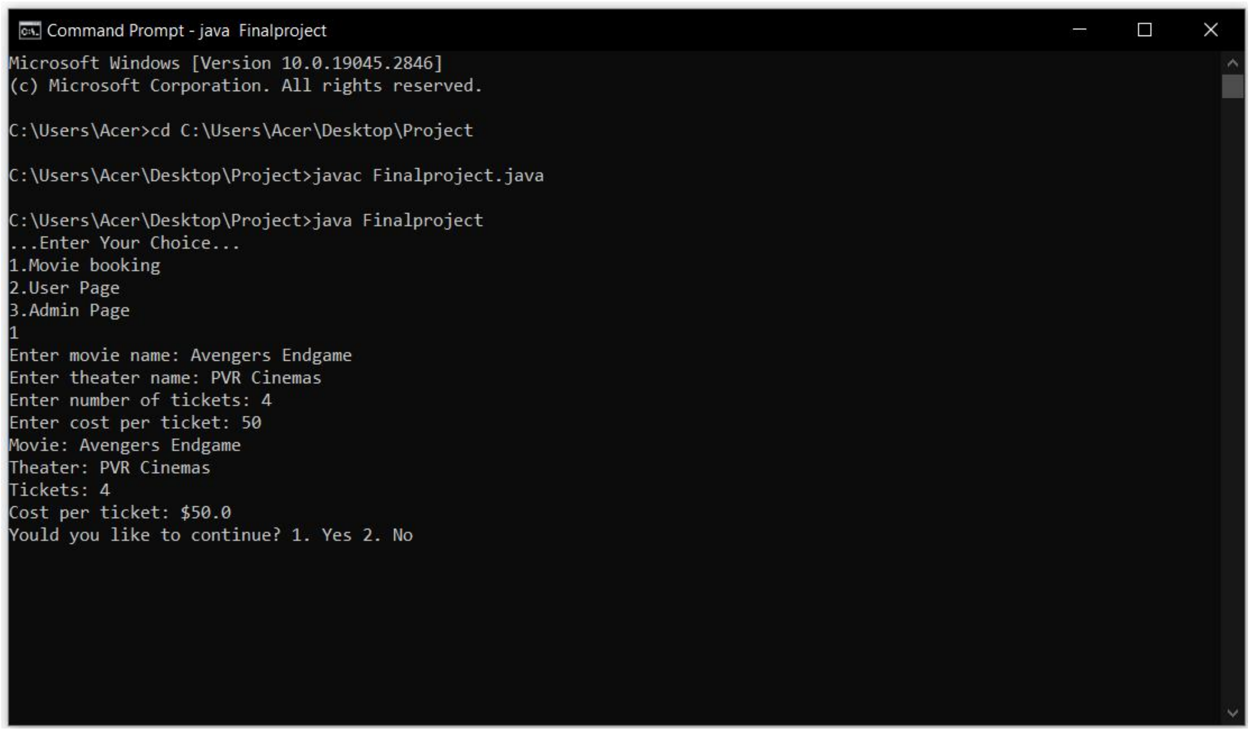
All the records of User account are stored in text file:



*users - Notepad

```
abc,kgf
user2,pass2
user3,pass3
user1,pass1
Abcd12,a@09
java12,1@34
```

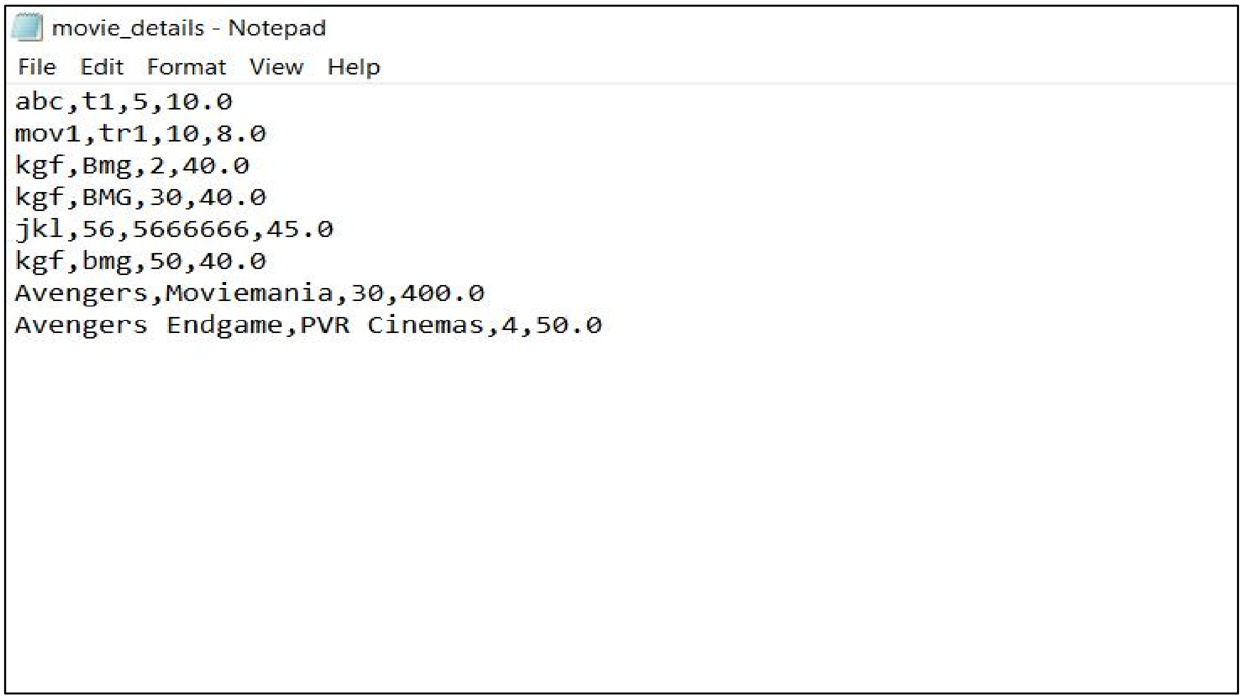
Output for Movie Details:



Command Prompt - java Finalproject

```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Acer>cd C:\Users\Acer\Desktop\Project

C:\Users\Acer\Desktop\Project>javac Finalproject.java

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.Movie booking
2.User Page
3.Admin Page
1
Enter movie name: Avengers Endgame
Enter theater name: PVR Cinemas
Enter number of tickets: 4
Enter cost per ticket: 50
Movie: Avengers Endgame
Theater: PVR Cinemas
Tickets: 4
Cost per ticket: $50.0
Would you like to continue? 1. Yes 2. No
```

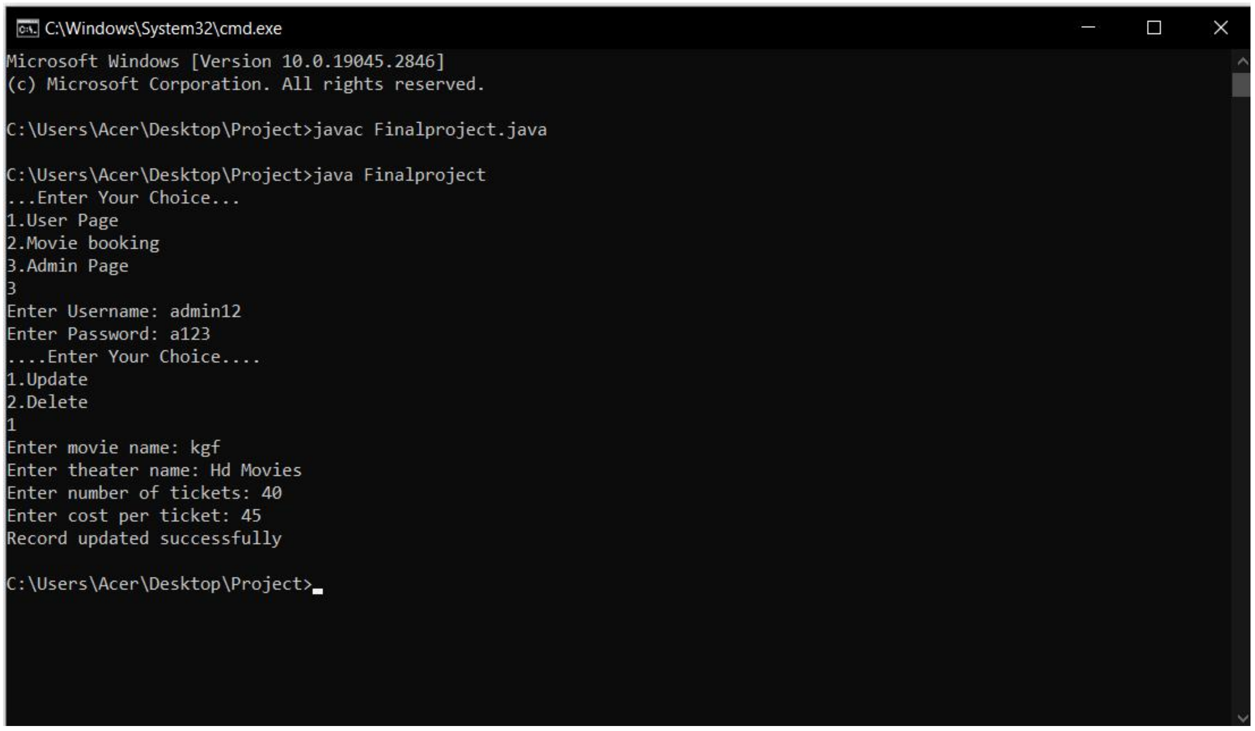
All the records of movies are stored in text file:



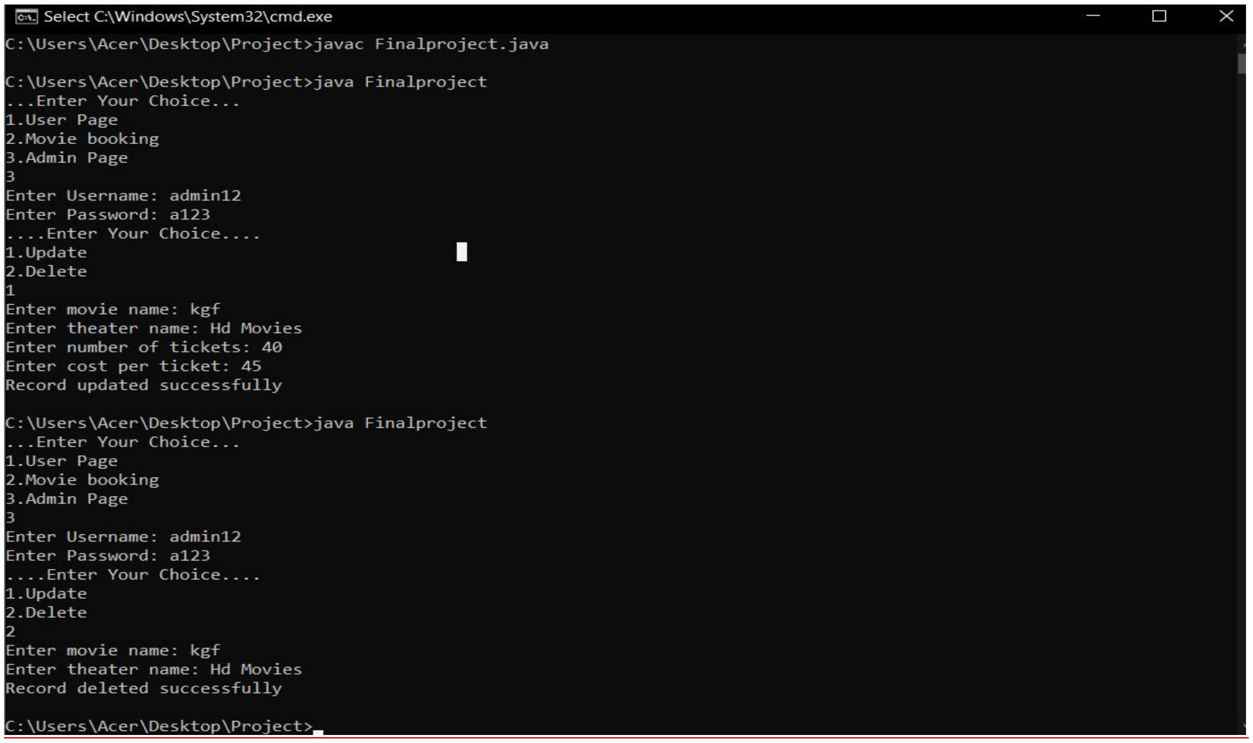
movie_details - Notepad

```
abc,t1,5,10.0
mov1,tr1,10,8.0
kgf,Bmg,2,40.0
kgf,BMG,30,40.0
jkl,56,5666666,45.0
kgf,bmg,50,40.0
Avengers,Moviemania,30,400.0
Avengers Endgame,PVR Cinemas,4,50.0
```

Output for Admin Login And Tasks:



```
Microsoft Windows [Version 10.0.19045.2846]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Acer\Desktop\Project>javac Finalproject.java

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.User Page
2.Movie booking
3.Admin Page
3
Enter Username: admin12
Enter Password: a123
....Enter Your Choice....
1.Update
2.Delete
1
Enter movie name: kgf
Enter theater name: Hd Movies
Enter number of tickets: 40
Enter cost per ticket: 45
Record updated successfully

C:\Users\Acer\Desktop\Project>
```



```
C:\Users\Acer\Desktop\Project>javac Finalproject.java

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.User Page
2.Movie booking
3.Admin Page
3
Enter Username: admin12
Enter Password: a123
....Enter Your Choice....
1.Update
2.Delete
1
Enter movie name: kgf
Enter theater name: Hd Movies
Enter number of tickets: 40
Enter cost per ticket: 45
Record updated successfully

C:\Users\Acer\Desktop\Project>java Finalproject
...Enter Your Choice...
1.User Page
2.Movie booking
3.Admin Page
3
Enter Username: admin12
Enter Password: a123
....Enter Your Choice....
1.Update
2.Delete
2
Enter movie name: kgf
Enter theater name: Hd Movies
Record deleted successfully

C:\Users\Acer\Desktop\Project>
```

Learning Outcome:

Object-oriented programming (OOP) concepts:

Building a movie ticket booking system in Java can help learners understand and apply OOP principles such as classes, objects, inheritance, encapsulation, and polymorphism. This can provide a solid foundation in OOP concepts, which are fundamental to Java and many other modern programming languages.

Java programming skills:

Developing a movie ticket booking system in Java can enhance learners' Java programming skills, including syntax, data types, variables, operators, control flow, and exception handling. It can also provide practice with Java's standard libraries and APIs for tasks such as input/output (I/O), collections, and date/time manipulation.

Project management:

Building a movie ticket booking system can provide experience in software development best practices, including requirements analysis, system design, coding standards, testing, debugging, and documentation. It can also provide opportunities for collaborative development, version control using tools like Git, and project management skills such as planning, organizing, and tracking progress.

Conclusion:

The Movie Ticket Booking Management System is an essential tool for movie theaters, as it enables them to efficiently manage their ticket sales and provide a better experience to their customers. The system has been designed to be user-friendly, efficient, and scalable, and it has been implemented using Java. The Agile methodology has been used to develop the system, which has allowed us to create a flexible and adaptive solution.

References:

1. https://www.geeksforgeeks.org/java/

2.https://www.tutorialspoint.com/java/index.htm

3.https://www.javatpoint.com/java-basics

4.https://www.codingninjas.com

5.https://www.w3schools.com/java/