JAVA PROJECT:

# NAME: MULTI CITY MOVIE TICKET BOOKING PORTAL.

NAME:ANURAG TOMAR REG.NO.:189301052 SEC: CSE-A

INTRODUCTION:

THE FOLLOWING PROJECT HAS BEEN BUILT TO PROVIDE A USER FRIENDLY EXPERIENCE TO THE CONSUMERS WHILE BOOKING MOVIE TICKETS.

THE PROJECT HAS BEEN DEVELOPED USING MULTIPLE CONCEPTS OF JAVA TO PROVIDE SMOOTH AND FLUENT MOVIE TICKET BOOKING EXPERICNCE TO THE USER FROM DIFFERENT PLACES IN INDIA FOR THE CURRENT AVAILABLE MOVIES..

CURRENT WORK:

CURRENTLY THE PROJECT IS CAPABLE OF PROVIDING SIGN-IN OPTION TO THE EXISTING USER AND ALSO SIGN-UP FACILITY TO THE NEW USERS THAT WANT TO USE THE PORTAL. BY SAVING THEIR PASSWORDS UNDER TEXTFILES OF THEIR USERNAMES.

AFTER THIS THE PROGRAM WILL ASK FOR THEIR CITY WHERE THEY WANT TO BOOK THE TICKETS IN.AFTER THIS THE CURRENT SHOWS WILL APPEAR AND THE USER WILL SELECT THE MOVIE HE WANTS TO BUY TICKETS FOR ALONG WITH THE TIME OF THE SHOW….HE CAN ALSO CONTINUE BOOKING TICKETS FROM A DIFFERENT CITY OR THE CURRENT CITY AS PER HIS/HER WISH. THE PROGRAM WILL PROVIDE A DETAILED BILL FOR THE PURCHASE MADE BY THE USER.THE USER WILL BE SIGNED OUT AND THE PROGRAM WILL RUN AGAIN ASKING FOR THE SIGN-IN DETAILS OF THE NEXT USER.

SOFTWARE REQUIREMENTS:

TECHNOLOGIES: CORE JAVA

PLATFORM: JAVA RUNTIME ENVIRONMENT(JRE) 1.6 OR ABOVE

OPERATING SYSTEM: WINDOWS 10/8/7/XP, LINUX

OTHER TOOLS: TEXT EDITOR(NOTEPAD)

EXPLANATION:

1. THE USER WILL BE PROVIDED WITH THREE OPTIONS A) SIGN-IN B)SIGN-UP C)EXIT
2. NEXT THE EXISTING USER WILL SELECT THE SIGN IN OPTION AND THE NEW USER WILL SELECT THE SIGN UP OPTION TO CREATE THE ACCOUNT.
3. EVEN IF THE NEW USER TRIES TO SIGN-IN HE WILL BE REDIRECTED TO THE SIGN-IN OPTION AND THE ACCOUNT WILL BE CREATED.
4. AFTER THE LOGING INTO THE PORTAL THE USER WILL BE ASKED WHETHER HE WANTS TO PURCHASE TICKETS OR EXIT THE PORTAL
5. AFTER SELECTING PURCHASE THE USER WILL BE ASKED TO ENTER THE CITY IN WHICH HE/SHE WANTS TO PURCHASE THE TICKETS.
6. THE PERSON WILL ENTER THE AMOUNT OF TICKETS HE/SHE WANTS TO PURCHASE.
7. THE MOVIE OPTIONS WILL APPEAR ALONG WITH THE SHOW TIMING FOR THE PARTICULAR MOVIE.
8. THE USER WILL SELECT THE TIMING HE/SHE PREFERS AND THE BILL CORRESPONDING TO THE PURCHASE WILL BE PROVIDED AS THE OUTPUT.
9. THE PORTAL FOR SIGN-IN WILL OPEN AGAIN AND THE PROCESS IS REPEATED.

CONCEPTS USED IN THE PROJECT:

INTERFACE:

IN THIS PROJECT ABSTRACT CLASSES ARE USED TO SAVE THE USERNAME AND PASSWORD OF THE

CUSTOMER. AS IN AN ABSTARCT CLASS ALL THE VARIABLES.

ACCESS MODIFIERS :

IN JAVA THERE ARE DIFFERENT ACCESS MODIFIERS THAT DETERMINE THEIR ACCESSIBILITY TO OTHER

CLASSES OR METHODS. SUCH AS: PUBLIC, STATIC. THESE HERE ARE USED FOR THE SAME. THEY DISRUPT

THE REACHABILITY OF THE DATA MEMBERS WHICH IN THE MAKES THE CODE MORE REALISTIC. FOR

EXAMPLE FOR THE BOOKING ID IT IS SET TO STATIC SO AS IT CAN NEVER BE REINITIALISED AGAIN OR

CAN NEVER BE SAME FOR TWO PERSONS.

INHERITANCE:

INHERITANCE IS ANOTHER TOOL USED HERE TO ACCESS different METHODS AND VARIABLES SO AS TO

NOT REPEAT OUR CODE UNNECESSARILY. HERE INHERITENCE IS USED TO CONNECT SIGNIN AND SIGNUP

TO SEE WHETHER THE CUSTOMER ALREADY EXISTS WITH THE SAME USERNAME.

FILE HANDLING:

FILE HANDLING PLAYS AN IMPORTANT PART IN THIS PROJECT ALMOST EVERYTHING IS DONE

THROUGH FILE HANDLING. TO STORE CONTENT, TO CHECK FOR PASSWORD AND USERNAME, TO

DELETE FOR SOME. AS BACKEND PROGRAMMING IS AVAILABLE TO US IT IS THE BEST WAY TO THESE

TASKS WITH OUR LIMITED RESOURCES.

INPUT OUTPUT STREAM:

INPUT AND OUTPUT STREAM IS USED HEAVILY IN THIS PROJECT. IT IS USED TO READ THE USERNAME

AND PASSWORD DURING SIGNIN AND DURING PASSWORD CHECKING. IT IS ALSO USED TO ADD

FLIGHT DETAILS TO THE USERNAME FILE AND DELETE IT DURING CANCELATIONS.

EXCEPTION HANDLING:

JAVA PROJECT 6

EXCEPTIONS ARE HANDLED DURING THIS PROCESS IF ANY OCCUR DUE TO CUSTOMERS WRONG INPUT

OR THE FILE IS NOT FOUND IN THE PROCESS. EXCEPTION HANDLING IS VERY MUCH USEFUL AS LETS

THE CODE RUNNING WITHOUT DISRUPTING THE ENTIRE CODE AND PREVENTS US FROM DOING THE

WHOLE PROCESS AGAIN.

CUSTOMS EXCEPTIONS:

EXCEPTIONS HAVE BEEN ADDED TO PROVIDE FLAWLESS RUN OF CODE WITHOUT TERMINATING SUDDENLY.

CUSTOM EXCEPTIONS ADDED: CityException , InvalidMovieException.

CONTROL STATEMENTS:

CONTROL STATEMENTS ARE USED IN THE CODE TO GIVE THE COMPILER WHAT TO DO, WHEN TO DO

IT. IF-ELSE, FOR LOOP, DO WHILE, SWITCH CASES ARE USED.

CONSTRUCTOR:

CLASSES LIKE SIGNIN, SIGNUP AND FLIGHT USES CONSTRUCTOR CALLING TO DEFINE THE

PARAMETERS THAT ARE TO BE USED IN THE RESPECTIVE CLASSES. It constructs the values at the

time of object creation. It is not necessary to write a constructor for a class. It is because

java compiler creates a default constructor if your class doesn’t have any.

JAVA STRING:

STRING CLASS IS USED EXCESSIVELY. ALMOST EVERYTHING IS USED TO STORE THE CONTENTS IN THE

STRING CLASS. THE USERNAME, PASSWORD, FLIGHT BOOKING DETAILS, ETC.

WRAPPER CLASSES:

The wrapper class in Java provides the mechanism to convert primitive into object and object into

primitive. It is used to store variables like ok, etc. as objects of wrapper classes.

MULTI-THREADING:

Multithreading in java is a process of executing multiple threads simultaneously.

However, we use multithreading than multiprocessing because threads use a shared

memory area. They don’t allocate separate memory area so saves memory, and context-

switching between the threads takes less time than process.

APPENDIX:

# CLASS: PROJECT

import java.io.\*;

import thispackage.\*; //CUSTOM PACKAGE IMPORT

import java.awt.Desktop;

import java.util.Scanner;

import java.util.Random;

import java.lang.\*;

class Project extends Thread{

public static void operations() throws IOException{

Logintest loginhere = new Logintest();

for(;;){

loginhere.runlogin();

for(;;){

System.out.println("1.purchase a ticket \n2.exit");

Scanner sc25 = new Scanner(System.in);

int option = sc25.nextInt();

if(option == 1){

System.out.println("enter the name of the city you are from :");

String Surat = "Surat";String movie1 = "INTERSTELLAR";String movie2 = "INCEPTION";

String Delhi = "Delhi";String nameOfcity = "";

Theatre th = new Theatre();

Theatre.Ticket ticket = th.new Ticket();

Scanner sc = new Scanner(System.in);

try{

nameOfcity = sc.nextLine();

}

catch(Exception e){ //EXCEPTION HANDLED FOR THE WRONG INPUT OF THE CITY

System.out.println("exception caught");

}

try{

ticket.checkCity(nameOfcity);

}

catch(CityException e){

System.out.println("we only have our cinema halls in surat and delhi at the moment....sorry for the inconvinience caused.");

}

////////////////BELOW THIS IS THE CODE FOR INPUT BY THE USER REGARDING THEIR CHOICES OF THE MOVIE///////////////////////

if(nameOfcity.equalsIgnoreCase(Surat)){

System.out.println("enter the amount of tickets you want to purchase");

Scanner sc2 = new Scanner(System.in);

String number = sc2.nextLine();

System.out.println("movies currently available : \n1.INTERSTELLAR \n2.INCEPTION");

Scanner sc3 = new Scanner(System.in);

String movie = sc3.nextLine();

try {

ticket.checkMovie(movie);

} catch (InvalidMovieException e) {

System.out.println("the movie you have entered is not currently being screened in any of our theatres.");

}

if(movie.equalsIgnoreCase(movie1)){

ticket.readContent1(number,"surat");

System.out.println("please confirm the timing you prefer");

Scanner sc23 = new Scanner(System.in);

String time = sc23.nextLine();

ticket.getBill(number, time);

ticket.showChoices();

}

else if(movie.equalsIgnoreCase(movie2)){

ticket.readContent2(number,"surat");

System.out.println("please confirm the timing you prefer");

Scanner sc23 = new Scanner(System.in);

String time = sc23.nextLine();

ticket.getBill(number, time);

ticket.showChoices();

}

}

else if(nameOfcity.equalsIgnoreCase(Delhi)){

System.out.println("enter the amount of tickets you want to purchase");

Scanner sc2 = new Scanner(System.in);

String number = sc2.nextLine();

System.out.println("enter the movie choice : \n1.INTERSTELLAR \n2.INCEPTION");

Scanner sc3 = new Scanner(System.in);

String movie = sc3.nextLine();

try {

ticket.checkMovie(movie);

} catch (InvalidMovieException e) {

System.out.println("the movie you have entered is not currently being screened in any of our theatres.");

}

if(movie.equalsIgnoreCase(movie1)){

ticket.readContent1(number,"delhi");

System.out.println("please confirm the timing you prefer");

Scanner sc23 = new Scanner(System.in);

String time = sc23.nextLine();

ticket.getBill(number, time);

ticket.showChoices();

}

else if(movie.equalsIgnoreCase(movie2)){

ticket.readContent2(number,"delhi");

System.out.println("please confirm the timing you prefer");

Scanner sc23 = new Scanner(System.in);

String time = sc23.nextLine();

ticket.getBill(number, time);

ticket.showChoices();

}

}

}

else

break;

}

}

}

public void run() {

try {

operations();

} catch(Exception e) {

e.printStackTrace();

}

}

}

# CLASS: THEATRE

import java.io.\*;

import thispackage.\*; //CUSTOM PACKAGE IMPORT

import java.awt.Desktop;

import java.util.Scanner;

import java.util.Random;

import java.lang.\*;

class Theatre{ //INNER CLASS

class Ticket extends MovieTickets{ //EXTENDS USED

public void checkMovie(String name) throws InvalidMovieException{

if(!name.equalsIgnoreCase("interstellar") && !name.equalsIgnoreCase("inception"))

throw new InvalidMovieException(movie); //THROWS KEYWORD IMPLEMENTATION

else

System.out.println("name of the movie :" + name);

}

public void checkCity(String name) throws CityException{

if(!name.equalsIgnoreCase("Surat") && !name.equalsIgnoreCase("Delhi"))

throw new CityException("our services are limited to surat and delhi only"); //THROWS KEYWORD USED

else

System.out.println("name of the city :" + name);

}

public void readContent1(String number, String place){

try {

BufferedReader Buff = new BufferedReader(new FileReader(place + "\\movie1\\screen1.txt"));

this.movie = Buff.readLine();

this.time = Buff.readLine();

this.price = Buff.readLine(); //USED FILEREADER

Buff.close(); //READING FROM THE TXT FILES FROM DIFFERENT DIRECTORIES

}

catch (IOException e) {

System.out.println("the error is caught"); //EXCEPTION HANDLING FOR THE BUFFEREDREADER

}

this.number = number;

this.screen = "screen1";

System.out.println(" name: " + this.movie + " \n number of tickets: " + this.number + "\n screen: " + this.screen + "\n Show timings: " + this.time);

}

public void readContent2(String number, String place){

try {

BufferedReader Buff = new BufferedReader(new FileReader(place + "\\movie2\\screen2.txt"));

this.movie = Buff.readLine();

this.time = Buff.readLine(); //THIS. IS USED TO REFER THE CURRENT CLASS OBJECT

this.price = Buff.readLine();

Buff.close(); //CLOSING BUFFEREDREADER TO PREVENT LEAKAGE OF DATA

} catch (IOException e) {

System.out.println("the error is caught"); //EXCEPTION HANDLING FOR THE BUFFEREDREADER

}

this.number = number;

this.screen = "screen2";

System.out.println(" name: " + this.movie + " \n number of tickets: " + this.number + "\n screen: " + this.screen + "\n Show timings: " + this.time);

}

public void getBill(String number, String timedec ){

this.timedecided = timedec;

String choose = " ABCDEFGHIJ";

char[] alphabet = choose.toCharArray();

int b = Integer.parseInt(number);

int a = Integer.parseInt(this.price); //BILLING METHOD

int cost = a\*b;

int abn = (int)(Math.random()\*((17-1)+1))+1; //RANDOMLY GENERATING SEATS

int index = (int)(Math.random()\*((10-1)+1))+1;

char mn = alphabet[index];

System.out.println("\*\*\*\*\*\*\*\*\*BILL\*\*\*\*\*\*\*\*\*");

System.out.println(" name: " + this.movie + " \n number of tickets: " + this.number + "\n screen: " + this.screen + "\n Show timings: " + this.timedecided);

if(b > 1)

System.out.println("your seat numbers are in the range: "+mn+":"+abn+" "+mn+":"+(abn+b-1));

else if(b == 1)

System.out.println("your seat number is: "+mn+":"+abn);

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

System.out.println("total price to be paid: Rs. " + cost);

System.out.println("thank you for chosing our service...have a nice day!");

System.out.println("\*\*\*\*\*\*\*\*\*BILL\*\*\*\*\*\*\*\*\*");

}

public void showChoices(){

File file = new File("newHope.jpg");

try{

Desktop desktop = Desktop.getDesktop();

desktop.open(file);

}

catch(Exception e){ // FILE HANDLING (FILE OPENING)

System.out.println("the desktop was unable to open the file");

}

}

}

}

# CLASS: LOGIN

import java.io.\*;

import java.util.Scanner;

class Login{

String username;

String password;

boolean check = false;

public void setDetails(String username, String password){

this.username = username;

this.password = password;

}

public void checkCredentials(String username , String password){

File tempFile = new File(username + ".txt");

if(tempFile.exists()){

BufferedReader br;

try{

br = new BufferedReader(new FileReader(username + ".txt"));

String a = br.readLine();

br.close();

if(a.equals(password)){

System.out.println("succesfully logged in..");

}

else{

System.out.println("the password entered is incorrect please try again: ");

System.out.println("enter the password again: ");

Scanner sc1 = new Scanner(System.in);

String newPassword = sc1.nextLine();

this.checkCredentials(this.username, newPassword);

}

}

catch(FileNotFoundException e){

System.out.println(e);

}

catch(Exception e){

System.out.println(e);

}

}

else {

System.out.println("you need to create an account first: ");

System.out.println("enter a new username: ");

Scanner sc34 = new Scanner(System.in);

String username1 = sc34.nextLine();

System.out.println("enter the new password: ");

Scanner sc29 = new Scanner(System.in);

String password1 = sc29.nextLine();

this.signUp(username1, password1);

}

}

public void signUp(String username, String password){

File tempFile = new File(username + ".txt");

if(tempFile.exists()){

System.out.println("the username already exists..plz try again with a different username: ");

}

else{

try{

tempFile.createNewFile();

FileWriter fw = new FileWriter(tempFile);

fw.write(password);

fw.close();

System.out.println("sign up succesfully completed");

}

catch(Exception e){

System.out.println(e);

}

}

}

}

# CLASS: LOGINTEST

import java.io.\*;

import java.util.Scanner;

class Logintest{

public void runlogin(){

Login login = new Login();

System.out.println("1.sign in \n2.sign up(create account) \n3.press '0' to exit the program");

Scanner option = new Scanner(System.in);

int b = option.nextInt();

if(b == 0){

System.exit(0);

}

else{

System.out.println("enter your username: ");

Scanner sc = new Scanner(System.in);

String username = sc.nextLine();

Console console = System.console();

char[] password1 = console.readPassword("enter the password: ");

String password = String.valueOf(password1);

login.setDetails(username, password);

if(b == 1){

login.checkCredentials(username, password);

//login.checkPassword();

}

else if(b == 2){

login.signUp(username, password);

}

}

}

}

# CLASS: THISPACKAGE-> MOVIETICKETS

package thispackage;

public abstract class MovieTickets{

public String price;

public String number;

public String movie;

public String time;

public String screen;

public String timedecided;

}

# CLASS: TEST

import java.lang.\*;

class Test {

public static void main(String[] args) {

Project p = new Project();

p.start();

}

}

# CLASS: EXCEPTIONS

class CityException extends Exception{ //

CityException(String s){ //CUSTOM EXCEPTION FOR READING MOVIE AND CITY NAME//

super(s); //

}

}

class InvalidMovieException extends Exception{

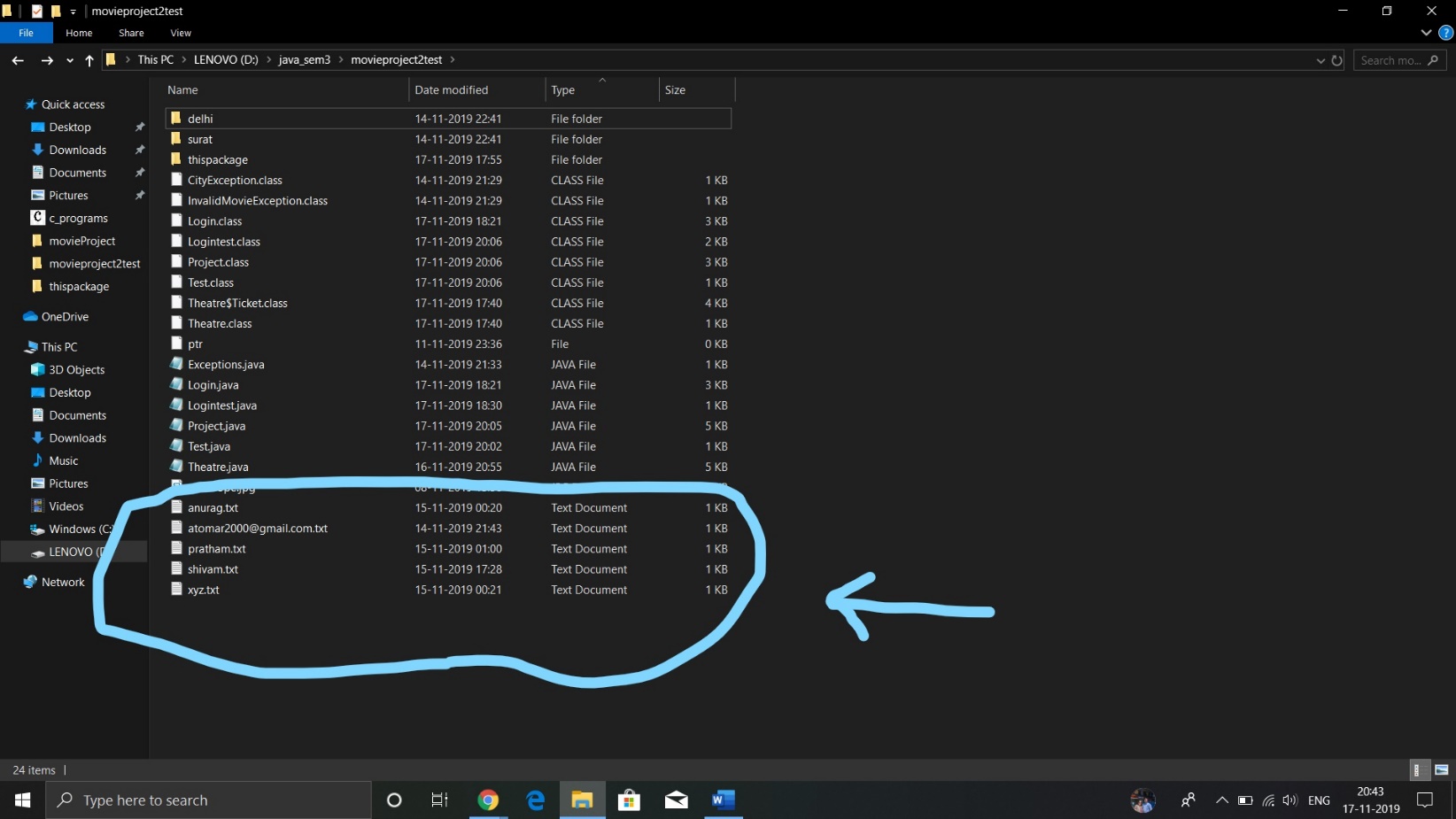
InvalidMovieException(String s){ //SUPER KEYWORD USED

super(s);

}

}

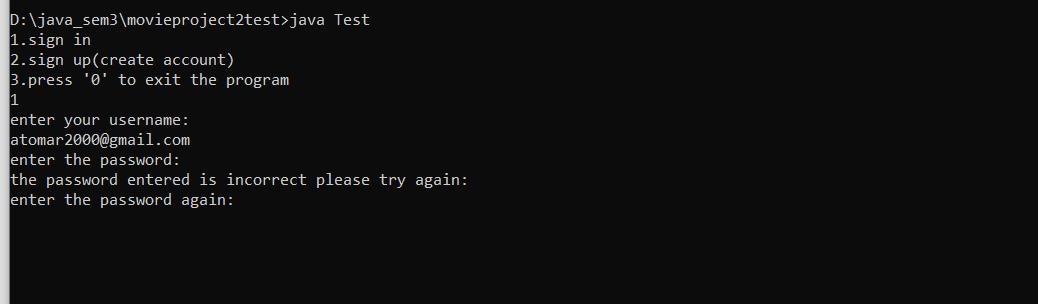
# THE USERNAME FILES GENERATED:



# OUTPUT:

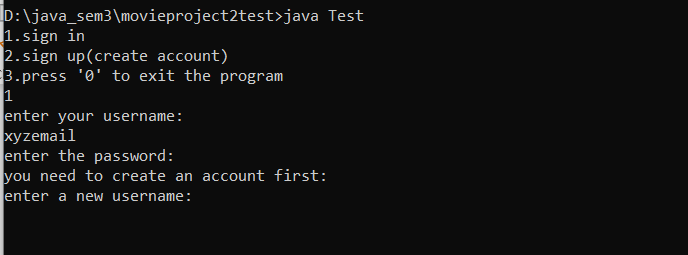
1. **When the user enters the wrong password:**

the program will generate a message saying that the password entered is incorrect and it will ask you to enter the password again .



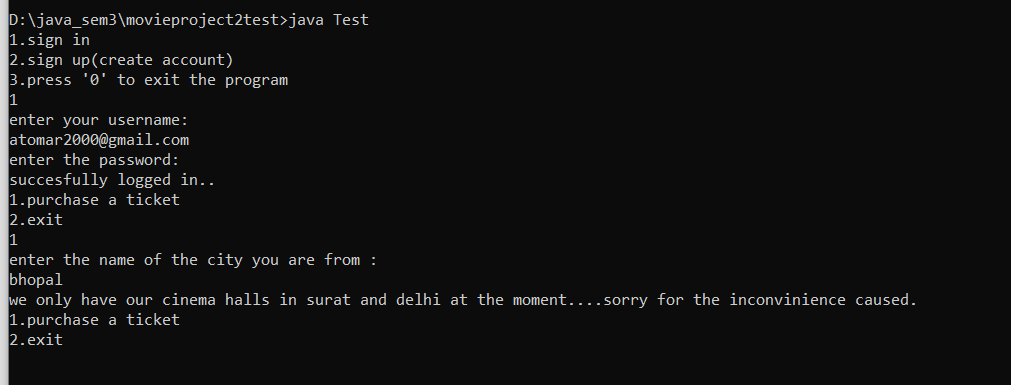
1. **When the non-existing users tries to sign-in:**

The portal will automatically switch to sign-up option and the user’s profile will be created.



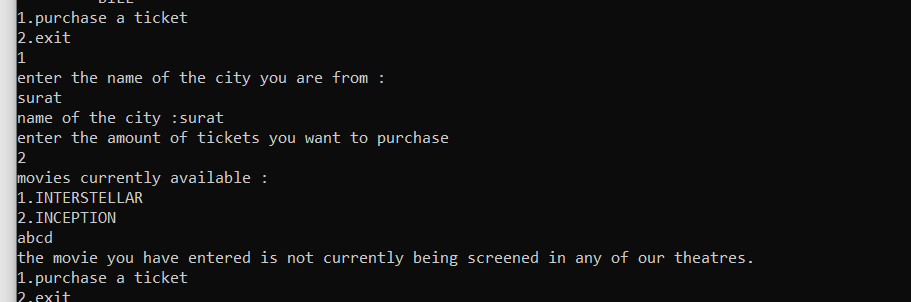
1. **When the user enters the name of the city where the facility is not available.**

The program will handle the exception by outputting a message for the users.

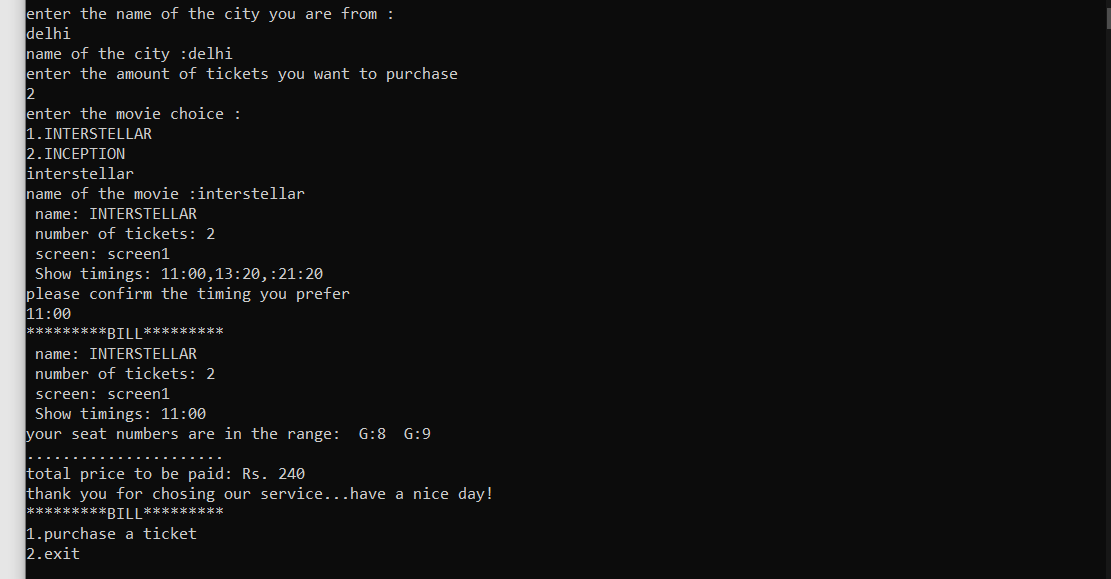


1. **When the users enters name of the movie wrong then.**

the exception is handled by providing a message for the user.

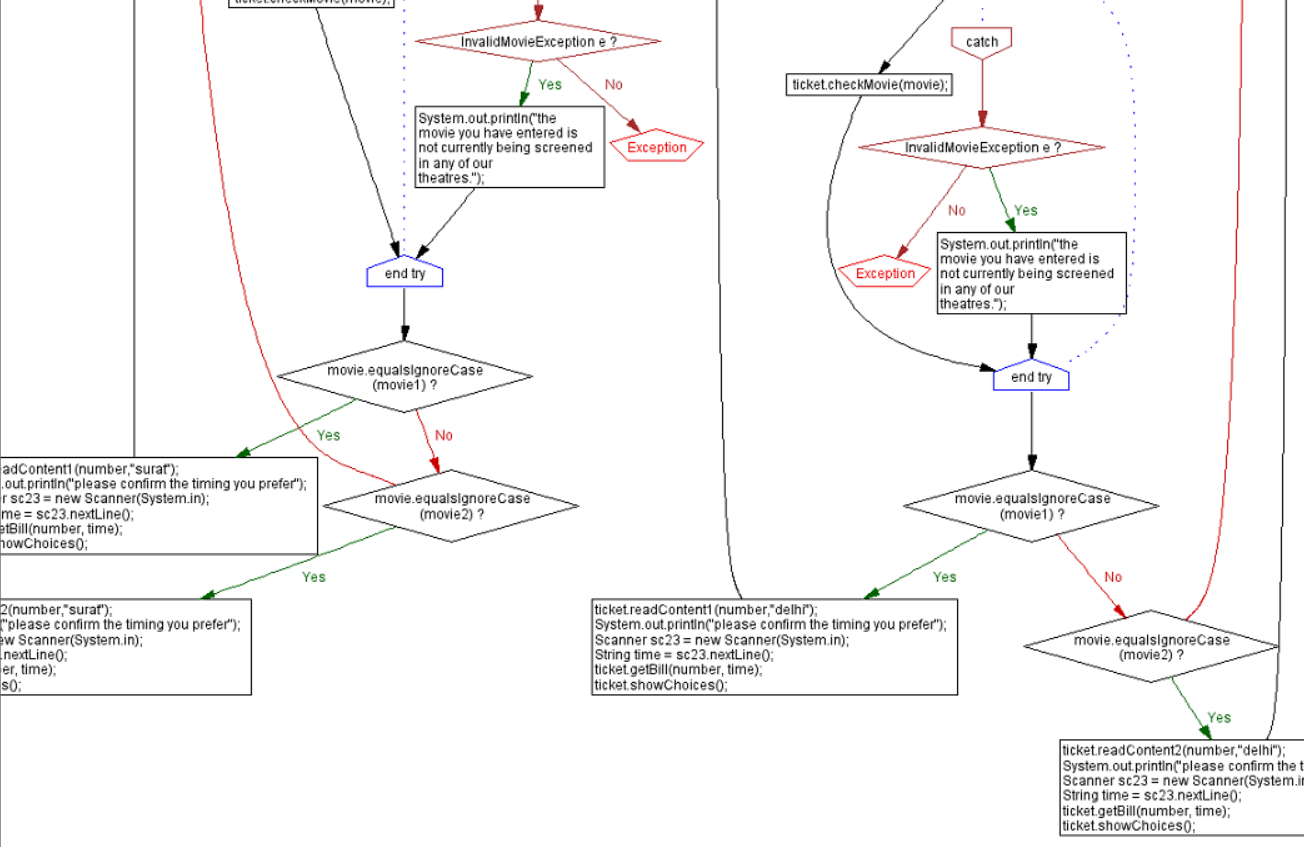


1. **When the user completes the booking the bill is given as the output.**

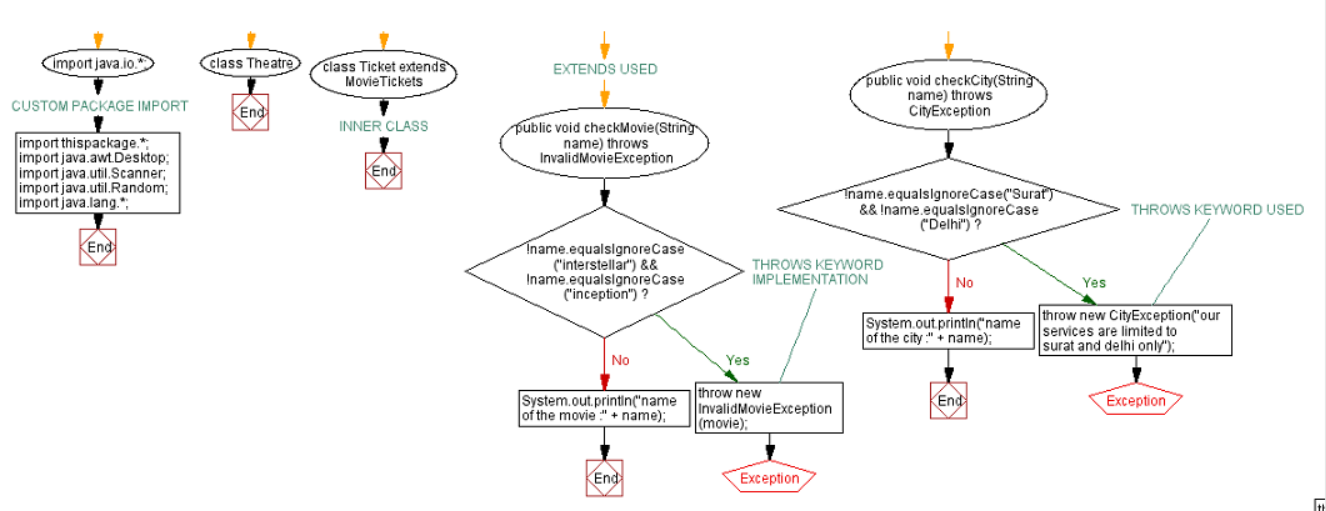


FLOWCHARTS:

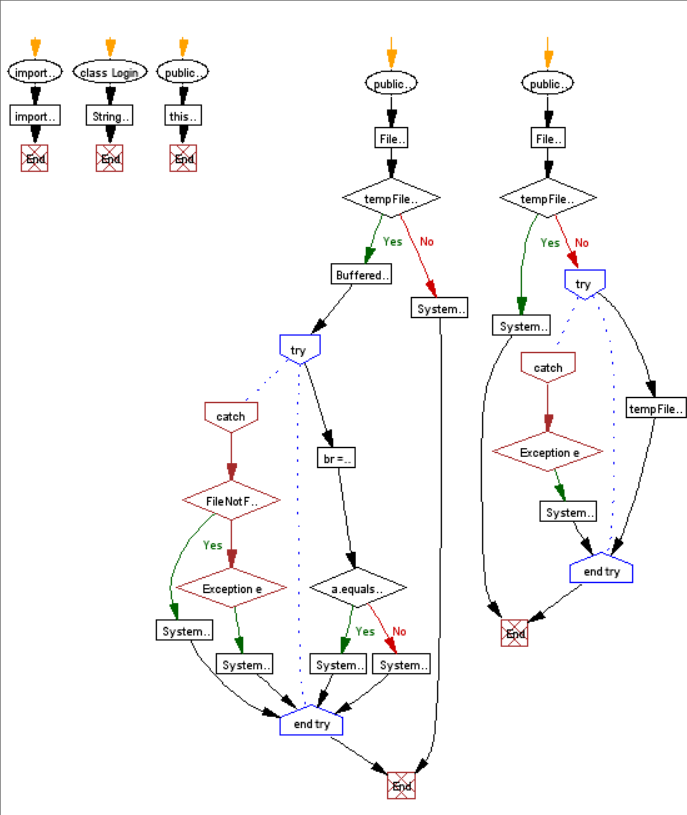
# FLOWCHART FOR PROJECT CLASS:



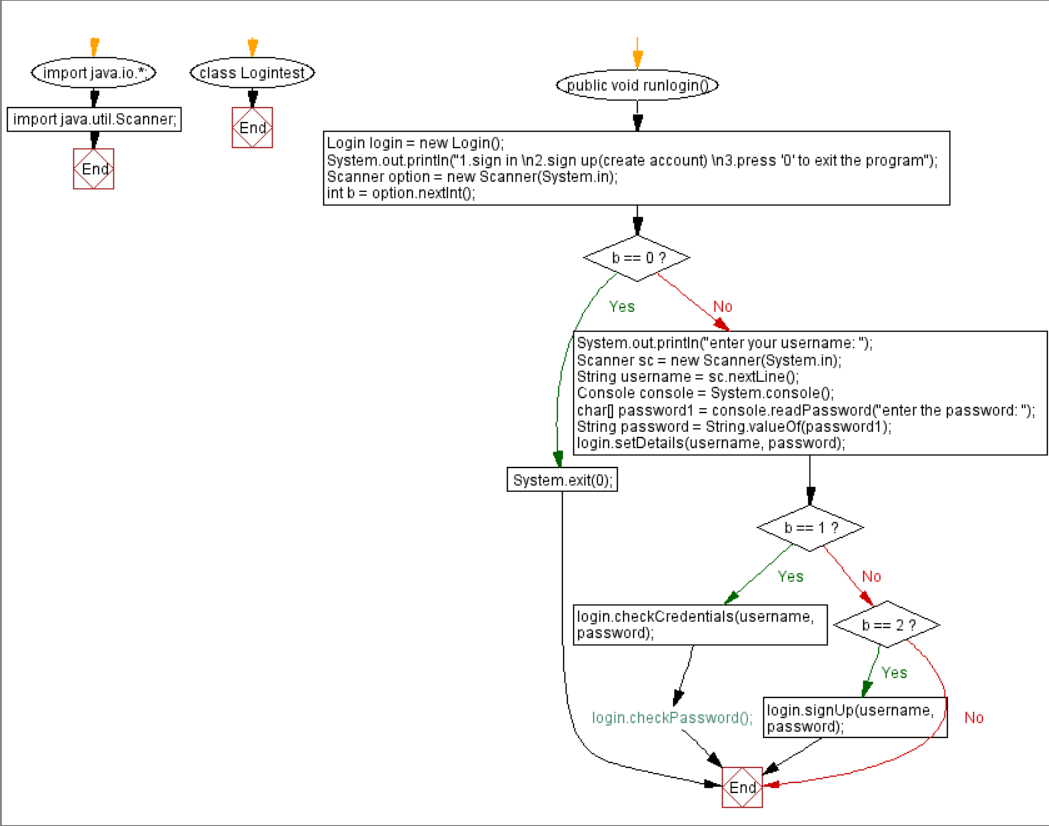
# FLOWCHART FOR THEATRE CLASS:



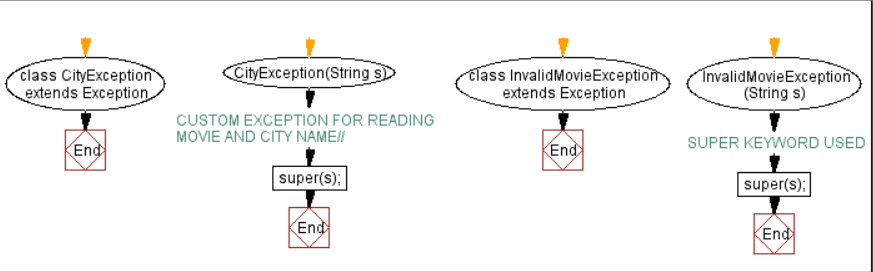
# FLOWCHART FOR LOGIN CLASS:



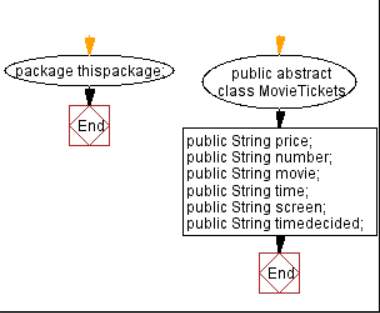
# FLOWCHART LOGINTEST:



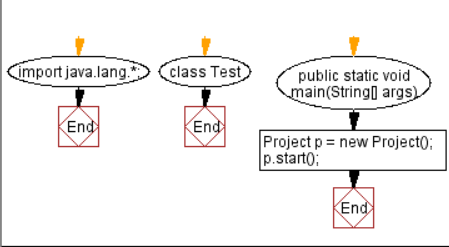
# FLOWCHART FOR THE EXCEPTION CLASS:



# FLOWCHART FOR THE MOVIETICKETS CLASS:



# FLOWCHART FOR THE TEST:



CONCLUSIONS:

Though the system still containing lot of scope of improvement in it. But its

overall look and feel gives rough picture of on existing automation system.

I have taken FileInputStream and input.txt as a substitute for database and the

project can be further improvised by using advance language like MySQL. The

Current work has limited memory of database and

requires manual input in database.

The entire project has been developed and deployed as per the requirement

stated by user, it is found to be bug free as per the testing standards that are

implemented. And by specification-untraced errors in the coming version, which

are planned to be developed in near future.

Finally, I like to conclude that I put all my effort through the development of

project and tried to fulfil most of the requirement of the user.