

# DBMS PROJECT REPORT

-By Tanay Jha

## **Problem Statement:**

Emulation of an online cinema movie ticket booking system using database concepts.

## **Scope:**

This project is a depiction of how movie ticket booking takes place through applications such as Bookmyshow. The user engages in a similar experience with the movies present in the respective database.

## **Software & Hardware Requirements:**

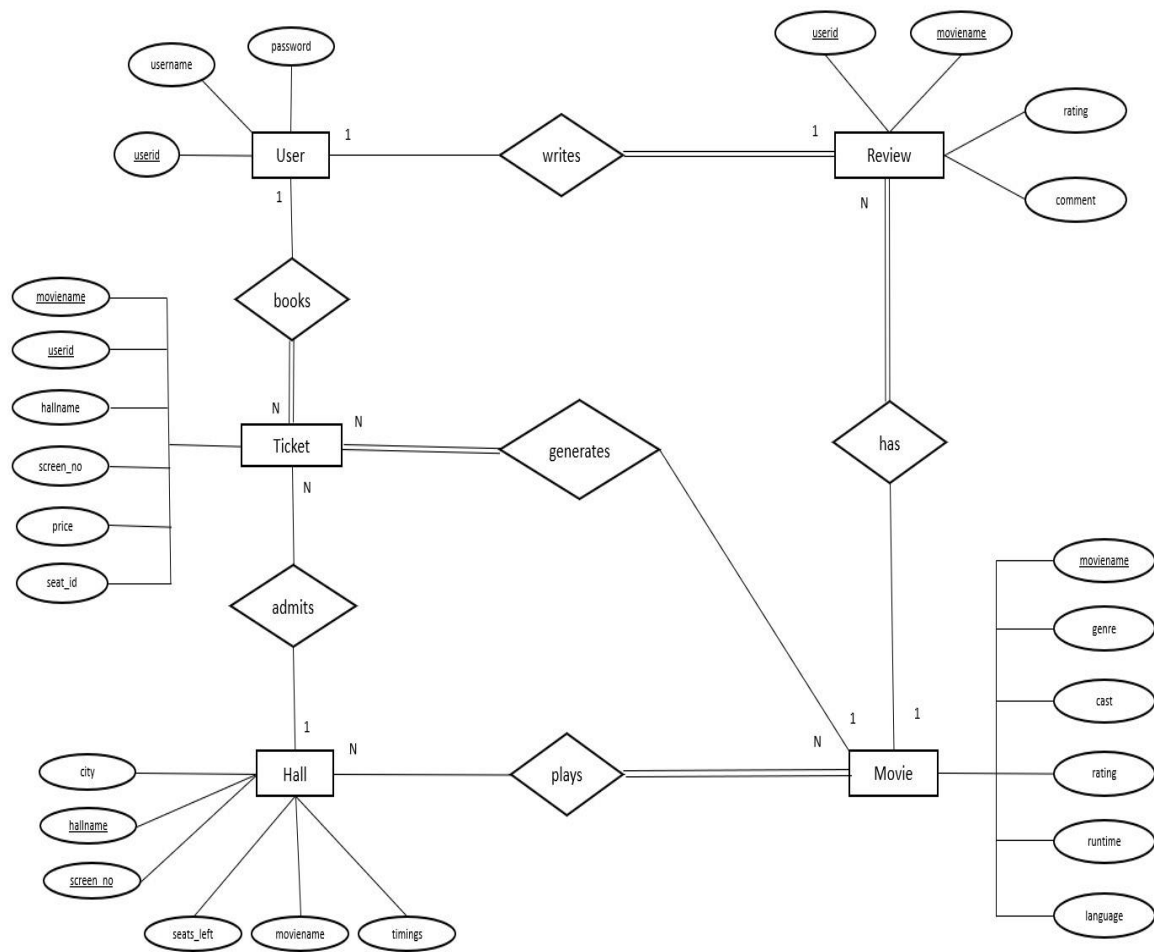
There are no external hardware requirements apart from the regular keyboard and mouse interface.

A modern Windows operating system, Oracle SQL, and Visual Studio 2012 (VC#) are required for running the program.

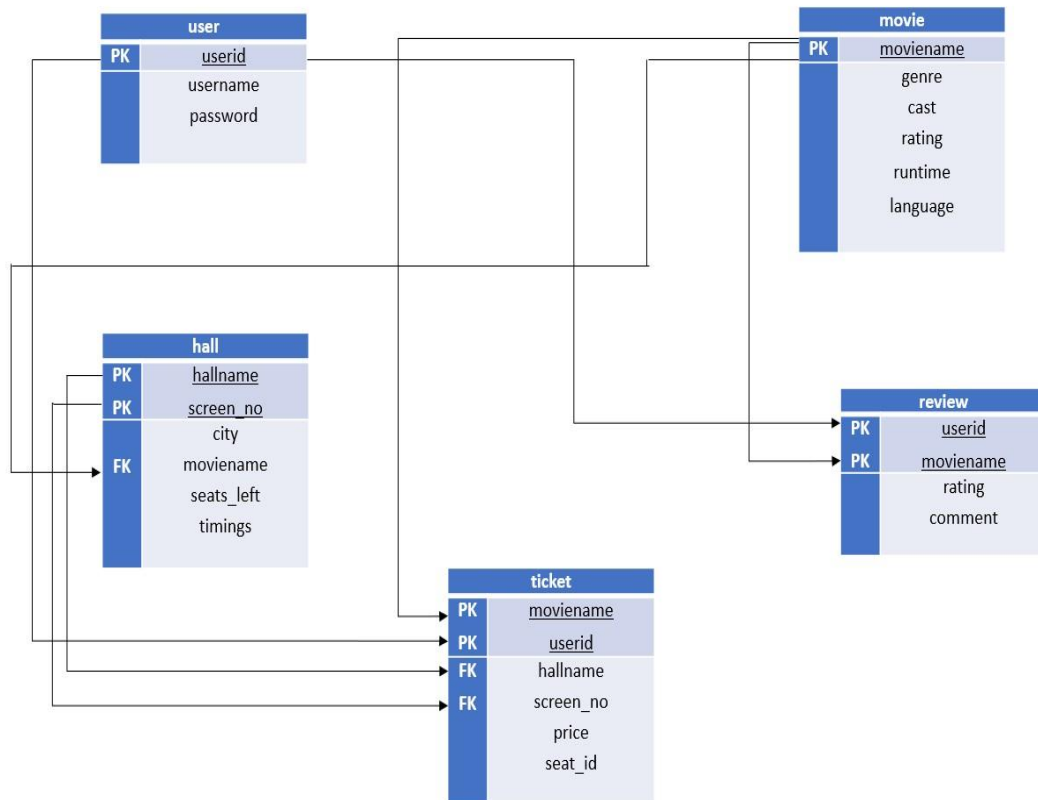
## **Design & Methodology:**

Multiple Windows Forms are linked together to seamlessly provide a smooth ticket booking experience for any number of users. This involves the provision of a quintessential sign up and log in feature, which uniquely stores ticket and user details. All attributes of users, movies, and tickets are stored in an Oracle SQL database which is connected throughout.

## ER Diagram:



## Table Relations:



## ER Schema Conversion:

User ( userid, username, password)

Hall ( hallname, screen\_no, City, ~~moviename~~, seats\_left, timings)

Ticket( moviename, userid, ~~hallname~~, ~~screen\_no~~, price, seat\_id)

Movie( moviename, genre, cast, rating, runtime, language)

Review( userid, moviename, rating, comment)

## **Steps of Conversion:**

1. Hall and Movie are in a many-to-many relationship (N to N) with Movie having total participation. Primary key 'moviename' from Movie is used as a foreign key in Hall.
2. Movie and Ticket are in a one-to-many relationship (1 to N) with Ticket having total participation. Primary key 'moviename' from Movie is used as a foreign key in Ticket.
3. User and Ticket are in a one-to-many relationship (1 to N) with Ticket having total participation. Primary key 'userid' from User is used as a foreign key in Ticket.
4. Hall and Ticket are in a one-to-many relationship (1 to N). Primary keys 'hallname' and 'screen\_no' from Hall are used as foreign keys in Ticket.
5. User and Review are in a one-to-one relationship (1 to 1) with Review having total participation. Primary key 'userid' from User is used as a foreign key in Review.
6. Movie and Review are in a one-to-many relationship (1 to N) with Review having total participation. Primary key 'moviename' from Movie is used as a foreign key in Review.

## **Normalisation:**

### **1NF:**

- Each cell is single valued.
- Entries in the column are of the same type
- Rows are uniquely identified: Each table has a primary key

### **2NF:**

- All attributes (non-key columns) are dependent on the key

### **3NF:**

- All fields (columns) can be determined only by the key in that table and no other column.

## UI and UI components:

- *Home Form:* Gives the options to either sign up or login.
- *Login Form:* User enters the userid and password to login.
- *Sign Up Form:* User enters the userid, display name and password to create an account.
- *Movie Home Page Form:* Welcomes the user from the Login page and shows the movies with posters, previous and next buttons to view all movies.
- *Change Password Form:* User can modify their existing password within the homepage.
- *Movie Details Form:* Displays all details about the movies like cast, genre and reviews given by the users.
- *Seat Selection Form:* Displays the seat layout, user can select a VIP or normal seat, the timing and hall name.
- *Ticket Form:* Once the seat is booked a ticket is generated with the details.

## Implementation:

- When a user signs up, a new insertion with his details is made in the user table.
- When a user logs in, his username and password is matched with the database.
- Home page opens with all the movies, running and their posters along with buttons to view drop down menu which provides the user options to change password, sign out and view history.
- On clicking change password, another form is loaded where the user can update his password.
- On clicking view details for any movie, another form is loaded wherein all the details are shown along with top reviews and an option to book tickets or go back.
- On clicking book tickets, a form with seat, date and theatre selection opens and the user can book their ticket.
- The trigger that we use allows only one review per user for one particular movie. On each insertion of review, it checks if the user has already added a review and if they have it updates it.
- The procedure is used to insert values into the tickets table when tickets are booked, and the number of seats left is reduced by one.

## Trigger:

```
create or replace trigger OneReviewONLY
Before insert on review
FOR EACH ROW
DECLARE Cursor c2 IS
Select userid,moviename from
review; row c2%ROWTYPE begin OPEN
c2;
LOOP
FETCH c2 INTO row;
EXIT WHEN c2%NOTFOUND;
IF row.userid=:NEW.userid AND row.moviename=:NEW.moviename THEN
Update table review set :old.comments=:new.comments where
userid=row.userid and moviename=row.moviename;
Update table review set :old.rating=:new.rating where
userid=row.userid and moviename=row.moviename;
END
LOOP;
CLOSE
c2;
end; /
```

## Procedure:

```
CREATE OR REPLACE PROCEDURE
Insert_tickets(      seatNo IN INTEGER,
mName IN VARCHAR,      hname IN VARCHAR,
screen_no IN INTEGER,      price IN
INTEGER,      userid IN INTEGER,      datex
IN Varchar
) IS      r_ticket
ticket%ROWTYPE;
seats_left INTEGER;
BEGIN
      Insert into ticket
values(mName,userid,hname,screen_no,price,seatNo);

UPDATE hall set seats_left=seats_left-1 where
hallname=hname and  moviename=mName and timings=datex;

SELECT seats_left into seats_left from hall where
hallname=hname and  moviename=mName and timings=datex;

END;
/
```



## **Result:**

A functioning interactive movie ticket booking application was successfully created with a total of seven forms, enabling the end user to create account, log in, change password, browse movies in theatres, and also book and view the tickets for the same.

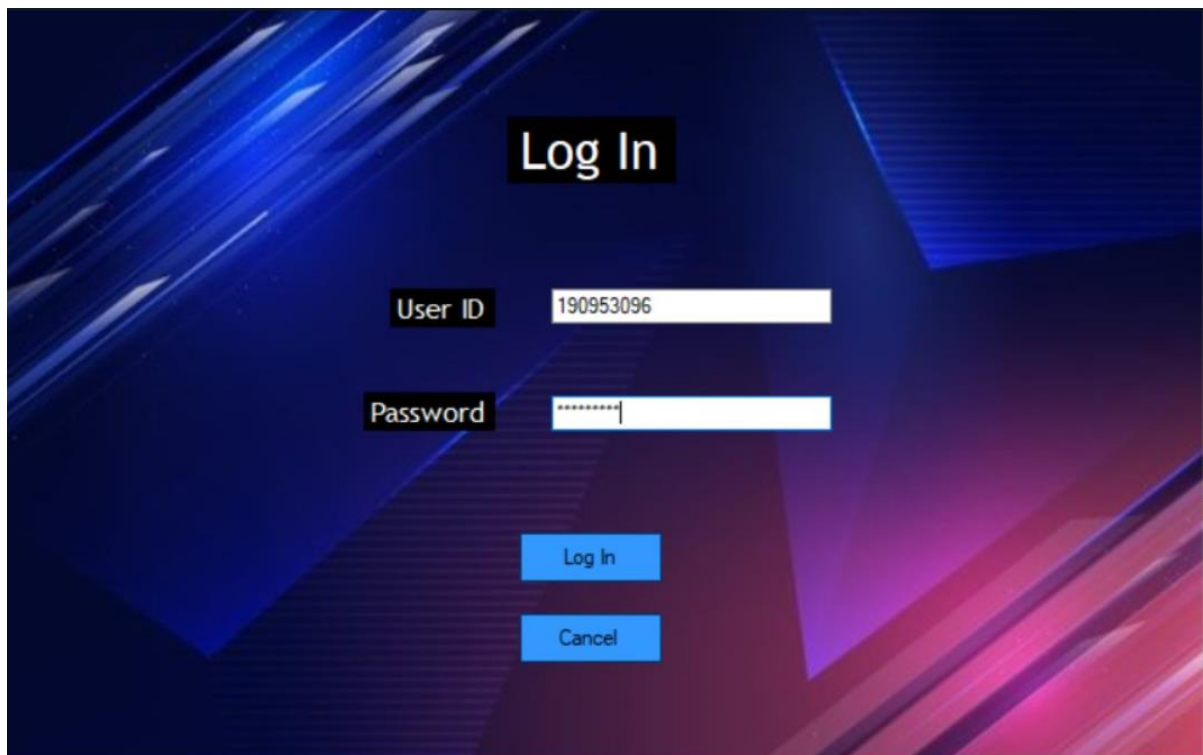
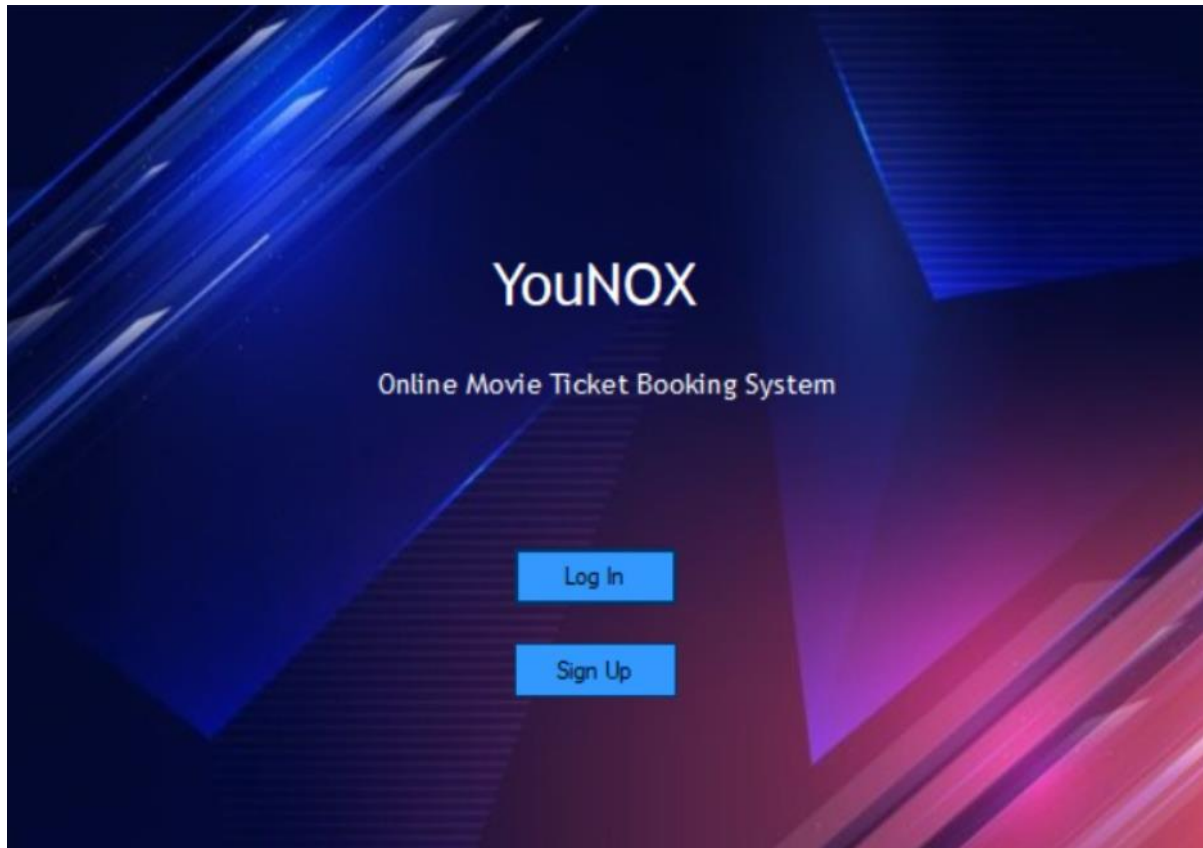
## **Discussion:**

The movie ticket booking system all in all provides the user with a very easy way to book his tickets right from the comfort of their home. In an effortless and quick process, this ensures that the user and the theatres do not face any unnecessary time wastage by standing in long queues for doing the same. The same application can be extended to not just movie ticket booking but also for events, shows, etc.

## **Conclusion:**

With the help of this application, a lot of time for the end-user is saved. The application does the job in a matter of a few minutes which would otherwise require a long time sometimes even hours. The application not only helps the end-users but also the theatres as all operations can take place online without them needing to do much manual work. The functionalities provided in the application provide the user with an immersive and personalised experience. This application has great scope, and it serves as the true meaning of convenience for an end-user.

## Screenshots of Front-End:




SETTINGS

Welcome Back, Jonathan

PREV


NEXT

Venom : Let There Be Carnage




VIEW

La La Land



VIEW

Radhe



VIEW

SETTINGS

VIEW TICKETS

CHANGE PASSWORD

SIGN OUT

PREV

NEXT

User ID: 190953096

Old Password: @john123

New Password: john@456

SUBMIT



## Venom : Let There Be Carnage

GENRE : Action, Thriller, Comedy

CAST : Tom Hardy, Woody Harrelson

RATING : 4/5

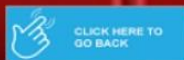
RUN-TIME : 105m

LANGUAGE : English

### Top Reviews:

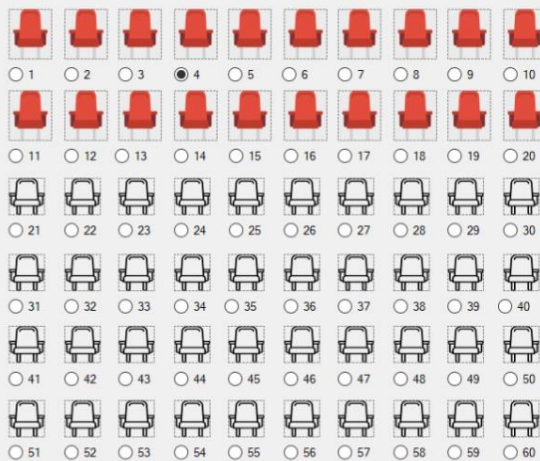
Absolutely brilliant movie. The acting by Tom Hardy is always a wonder to see. I hope Marvel makes great content like this in the future!

NEXT



BOOK NOW

## Choose Your Seat



### Select Theater

PVR

Select

### Select Date

28/11/2021

Select

Rs. 400

Cancel

Confirm Payment