

# MOVIE BUZZ

## PROJECT DOCUMENTATION

### Table of Contents

<b>1. ABSTRACT .....</b>	<b>2</b>
1.1. TITLE OF THE PROJECT .....	2
1.2. BRIEF DESCRIPTION OF PROJECT .....	2
1.3. NAME AND HALL TICKET OF TEAM NUMBERS.....	2
<b>2. ER DIAGRAM.....</b>	<b>3</b>
<b>3. DATA BASE TABLES.....</b>	<b>4</b>
<b>4. CODE .....</b>	<b>21</b>
• HomePage.java .....	21
• CustomerPage.java .....	25
• Register.java.....	27
• Login.java .....	33
• MoviePage.java.....	36
• MovieSelection.java.....	38
• AdminLogin.java.....	42
• AdminView.java .....	45
<b>5. Output Screenshots.....</b>	<b>52</b>
<b>6. Conclusion .....</b>	<b>56</b>

# MOVIEBUZZ

A PROJECT Done By

Name	Roll number
Rishik	1602-19-737-156
Rithik	1602-19-737-157
Shailendhar	1602-19-737-167

## ABSTRACT

The project objective is to book cinema tickets online. **MovieBuzz** is a **Desktop-based application** that can be accessed throughout the Net and can be accessed by anyone who has a net connection. This application will automate the reservation of tickets and inquiries about the availability of the tickets.

Watching movies with family and friends in the theatre is one of the best mediums of entertainment after having a hectic schedule. But all this excitement vanishes after standing in hours in long queues to get tickets booked.

The application provides complete information regarding currently running movies on all the screens with details of show timings, available seats, and fare charges of different classes. Seats can be reserved for different classes as well for the same show and class also.

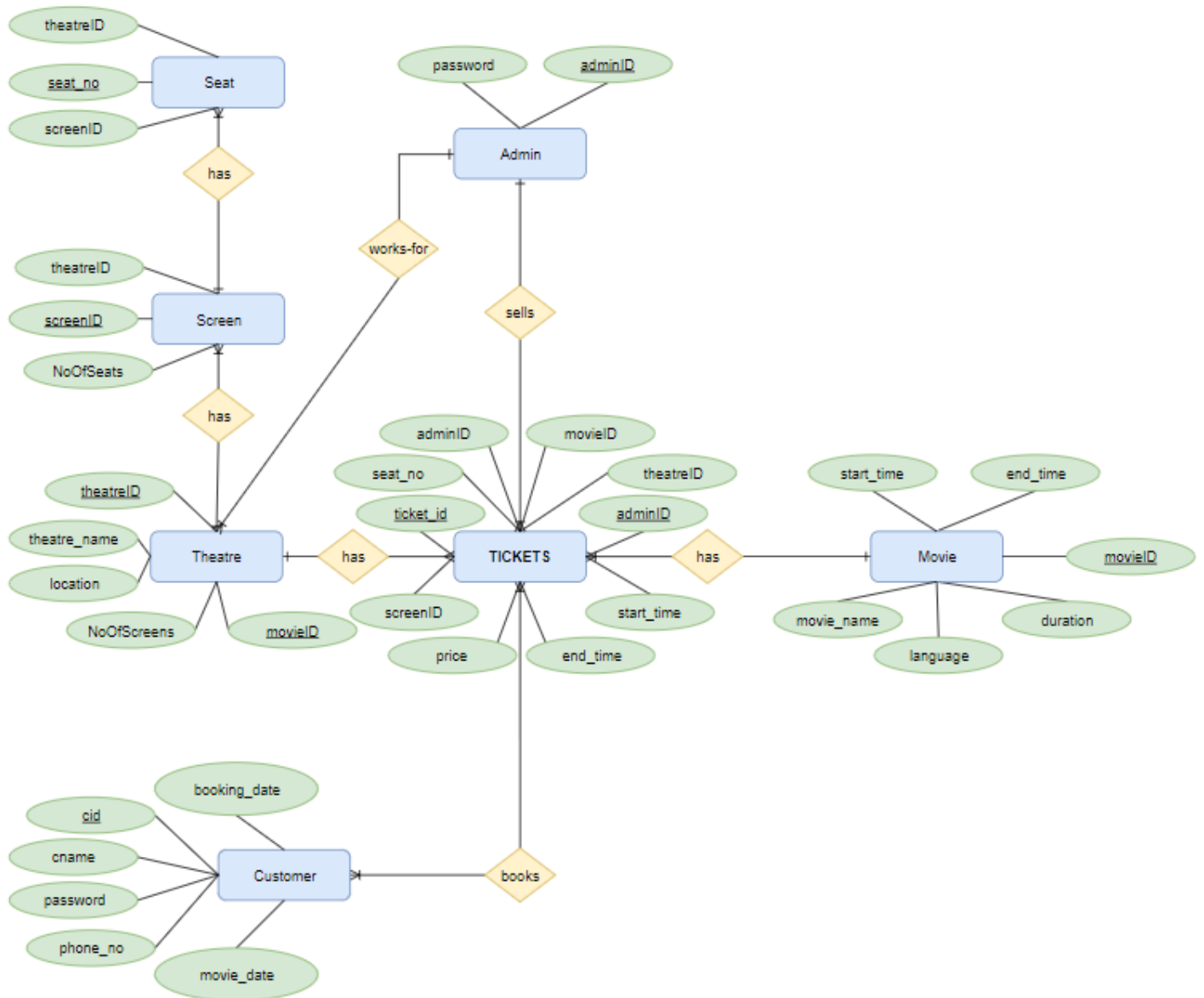
By using **MovieBuzz**, Users can be able to reserve seats in nearby theatres by looking at information regarding currently running movies with details of show timings and available seats. Whereas, Admin maintains and manages the information related to movie tickets and theatres.

Our application, **MovieBuzz** 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. It also provides the option to cancel the tickets which are reserved previously.

## PROJECT DESCRIPTION:

**MovieBuzz** is a Desktop application for booking cinema tickets. Customers can interact with the **MovieBuzz** application to know about currently running movies and their schedules and also service information provided by the theatres. The objective of **MovieBuzz** is to create an application for Movie ticket booking that allows users to know about new movies, cinema locations, available theatres, their schedules, class, and ticket price. In the booking process when a customer selects his city then cinema theatres of that location are filtered. In the next step, he/she selects the theatre where he/she wishes to see a movie, then selects the movie and other details like show date, show time, screen, class, price, and no. of tickets. Based on given parameters a layout of seat status is visible to the customer. Now the customer can select desired seat locations as per the number of seats required and then tickets get booked. The Administrator will be able to see booked and canceled tickets.

## ER DIAGRAM:



## **LIST OF TABLES:**

- ADMIN
- CUSTOMER
- MOVIE\_DETAILS
- MOVIE\_TIMINGS
- THEATRE
- SHOWS
- SCREEN
- SEAT
- TICKETS
- BOOKINGS

## **NORMALIZATION:**

**Normalization** is the process of **minimizing redundancy** from a relation or set of relations. Redundancy in relation may cause insertion, deletion and updation anomalies. So, it helps to minimize the redundancy in relations. **Normal forms** are used to eliminate or **reduce redundancy** in database tables.

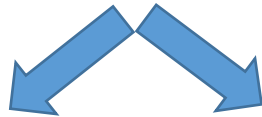
There are different types of normal forms:

- a. 1<sup>st</sup> Normal Form
- b. 2<sup>nd</sup> Normal Form
- c. 3<sup>rd</sup> Normal Form
- d. BCNF Normal Form
- e. 4<sup>th</sup> Normal Form
- f. 5<sup>th</sup> Normal Form

The above tables are created to avoid repetition of values in the main table and manage mainly many – to – many relationships. All the above tables are in **3<sup>rd</sup> Normal Form** and are free from Insertion, Deletion and Updating Anomalies.

## CUSTOMER DETAILS TABLE

1. CID
2. CNAME
3. PASSWORD
4. PHONE_NO
5. BOOKING_ID
6. BOOKING_DATE
7. MOVIE_DATE
8. TICKET_ID



1. CID
2. CNAME
3. PASSWORD
4. PHONE_NO

1. BOOKING_ID
2. CUSTOMER_ID
3. BOOKING_DATE
4. MOVIE_DATE
5. TICKET_ID

As one customer can have multiple bookings the customer details table is divided into 2 partitioned tables. The above 2 isolated tables are **CUSTOMER TABLE** AND **BOOKINGS TABLE**.

### CUSTOMER TABLE:

#### LIST OF ATTRIBUTES:

- ✓ customer\_id VARCHAR2(10)
- ✓ cname VARCHAR2(10)
- ✓ password VARCHAR2(10)
- ✓ phone\_no NUMBER(10)

**PRIMAY KEY:** customer\_id

```
SQL> create table CUSTOMER(  
 2 customer_id varchar2(10) PRIMARY KEY,  
 3 cname varchar2(20) NOT NULL,  
 4 password varchar2(10) NOT NULL,  
 5 phone_no number(10) UNIQUE);
```

Table created.

```
SQL> desc CUSTOMER;
```

Name	Null?	Type
CUSTOMER_ID	NOT NULL	VARCHAR2(10)
CNAME	NOT NULL	VARCHAR2(20)
PASSWORD	NOT NULL	VARCHAR2(10)
PHONE_NO		NUMBER(10)

```

SQL> insert into CUSTOMER values('&customer_id', '&cname', '&password', &phone_no);
Enter value for customer_id: C-01
Enter value for cname: Michael
Enter value for password: mike101
Enter value for phone_no: 9887662233
old 1: insert into CUSTOMER values('&customer_id', '&cname', '&password', &phone_no)
new 1: insert into CUSTOMER values('C-01', 'Michael', 'mike101', 9887662233)

1 row created.

SQL> insert into CUSTOMER values('&customer_id', '&cname', '&password', &phone_no);
Enter value for customer_id: C-02
Enter value for cname: Rishik
Enter value for password: rishik2002
Enter value for phone_no: 9493019049
old 1: insert into CUSTOMER values('&customer_id', '&cname', '&password', &phone_no)
new 1: insert into CUSTOMER values('C-02', 'Rishik', 'rishik2002', 9493019049)

1 row created.

SQL> insert into CUSTOMER values('C-03', 'Rithik', 'rithik2001', 9100107899);

1 row created.

SQL> insert into CUSTOMER values('C-04', 'Shailendhar', 'shail2000', 6302755514);

1 row created.

```

```

SQL> select * from CUSTOMER;

```

CUSTOMER_I	CNAME	PASSWORD	PHONE_NO
C-01	Michael	mike101	9887662233
C-02	Rishik	rishik2002	9493019049
C-03	Rithik	rithik2001	9100107899
C-04	Shailendhar	shail2000	6302755514

## **BOOKINGS TABLE:**

### **LIST OF ATTRIBUTES:**

- ✓ booking\_id VARCHAR2(10)
- ✓ customer\_id VARCHAR2(10)
- ✓ booking\_date DATE
- ✓ movie\_date DATE
- ✓ ticket\_id VARCHAR2(10)

**PRIMARY KEY:** booking\_id

**FOREIGN KEY:** customer\_id, ticket\_id

```
SQL> create table BOOKINGS(  
 2  booking_id varchar2(10) primary key,  
 3  customer_id varchar2(10),  
 4  booking_date DATE,  
 5  movie_date DATE,  
 6  ticket_id varchar2(10),  
 7  foreign key(customer_id) references CUSTOMER(customer_id),  
 8  foreign key(ticket_id) references TICKETS(ticket_id));
```

Table created.

```
SQL> desc BOOKINGS;
```

Name	Null?	Type
BOOKING_ID	NOT NULL	VARCHAR2(10)
CUSTOMER_ID		VARCHAR2(10)
BOOKING_DATE		DATE
MOVIE_DATE		DATE
TICKET_ID		VARCHAR2(10)

```
SQL> insert into BOOKINGS values('BK-01', 'C-02', '05-JUN-21', '01-JUL-21', 'TIK-01');
```

1 row created.

```
SQL> insert into BOOKINGS values('BK-02', 'C-03', '05-JUN-21', '05-JUL-21', 'TIK-02');
```

1 row created.

```
SQL> insert into BOOKINGS values('BK-03', 'C-04', '05-JUN-21', '15-JUL-21', 'TIK-03');
```

1 row created.

```
SQL> select * from BOOKINGS;
```

BOOKING_ID	CUSTOMER_I	BOOKING_D	MOVIE_DAT	TICKET_ID
BK-01	C-02	05-JUN-21	01-JUL-21	TIK-01
BK-02	C-03	05-JUN-21	05-JUL-21	TIK-02
BK-03	C-04	05-JUN-21	15-JUL-21	TIK-03

## ADMIN TABLE

1. ADMIN_ID
2. PASSWORD

The above **ADMIN** table is already in 3<sup>rd</sup> Normal Form

### ADMIN TABLE:

#### LIST OF ATTRIBUTES:

- ✓ admin\_id VARCHAR2(10)
- ✓ password VARCHAR2(15)

**PRIMARY KEY:** admin\_id

```
SQL> create table Admin(  
2  admin_id varchar2(10),  
3  password varchar2(15) NOT NULL,  
4  primary key(admin_id));
```

Table created.

```
SQL> desc Admin;
```

Name	Null?	Type
ADMIN_ID	NOT NULL	VARCHAR2(10)
PASSWORD	NOT NULL	VARCHAR2(15)

```
SQL> insert into Admin values('System', 'IT-156-157-167');
```

1 row created.

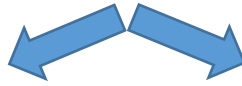
```
SQL> select * from Admin;
```

ADMIN_ID	PASSWORD
System	IT-156-157-167



## MOVIE TABLE

1. MOVIE_ID
2. MOVIE_NAME
3. LANGUAGE
4. DURATION
5. START_TIME
6. END_TIME



1. MOVIE_ID
2. MOVIE_NAME
3. LANGUAGE
4. DURATION

1. MOVIE_ID
2. START_TIME
3. END_TIME

As one movie can have multiple show timings the movie table is divided into 2 partitioned tables. The above 2 isolated tables are **MOVIE DETAILS TABLE** AND **MOVIE TIMINGS TABLE**.

### MOVIE DETAILS TABLE:

#### LIST OF ATTRIBUTES:

- ✓ movie\_id VARCHAR2(10)
- ✓ movie\_name VARCHAR2(10)
- ✓ language VARCHAR2(10)
- ✓ duration VARCHAR2(10)

**PRIMARY KEY:** movie\_id

```
SQL> create table MOVIE_DETAILS(  
2 movie_id varchar2(10) primary key,  
3 movie_name varchar2(25) NOT NULL,  
4 language varchar2(10) NOT NULL,  
5 duration varchar2(10) NOT NULL);
```

Table created.

```
SQL> desc MOVIE_DETAILS;
```

Name	Null?	Type
MOVIE_ID	NOT NULL	VARCHAR2(10)
MOVIE_NAME	NOT NULL	VARCHAR2(25)
LANGUAGE	NOT NULL	VARCHAR2(10)
DURATION	NOT NULL	VARCHAR2(10)

```

SQL> insert into MOVIE_DETAILS values('&movie_id', '&movie_name', '&language', '&duration');
Enter value for movie_id: M-01
Enter value for movie_name: BLACK WIDOW
Enter value for language: English
Enter value for duration: 2hr 14min
old 1: insert into MOVIE_DETAILS values('&movie_id', '&movie_name', '&language', '&duration')
new 1: insert into MOVIE_DETAILS values('M-01', 'BLACK WIDOW', 'English', '2hr 14min')

1 row created.

SQL> insert into MOVIE_DETAILS values('&movie_id', '&movie_name', '&language', '&duration');
Enter value for movie_id: M-02
Enter value for movie_name: Spider Man: NO WAY HOME
Enter value for language: English
Enter value for duration: 2hr 25min
old 1: insert into MOVIE_DETAILS values('&movie_id', '&movie_name', '&language', '&duration')
new 1: insert into MOVIE_DETAILS values('M-02', 'Spider Man: NO WAY HOME', 'English', '2hr 25min')

1 row created.

```

```

SQL> insert into MOVIE_DETAILS values('M-03', 'K.G.F Chapter 2', 'Hindi', '2hr 50min');

1 row created.

SQL> insert into MOVIE_DETAILS values('M-04', 'Jersey', 'Hindi', '2hr 30min');

1 row created.

SQL> insert into MOVIE_DETAILS values('M-05', 'RRR', 'Telugu', '3hr 00min');

1 row created.

```

```
SQL> select * from MOVIE_DETAILS;
```

MOVIE_ID	MOVIE_NAME	LANGUAGE	DURATION
M-01	BLACK WIDOW	English	2hr 14min
M-02	Spider Man: NO WAY HOME	English	2hr 25min
M-03	K.G.F Chapter 2	Hindi	2hr 50min
M-04	Jersey	Hindi	2hr 30min
M-05	RRR	Telugu	3hr 00min

## MOVIE TIMINGS TABLE:

### LIST OF ATTRIBUTES:

- ✓ movie\_id VARCHAR2(10)
- ✓ start\_time VARCHAR2(10)
- ✓ end\_time VARCHAR2(10)

**PRIMARY KEY:** (movie\_id, start\_time, end\_time)

**FOREIGN KEY:** movie\_id

```
SQL> create table MOVIE_TIMINGS(  
  2  movie_id varchar2(10),  
  3  start_time varchar2(10),  
  4  end_time varchar2(10),  
  5  foreign key(movie_id) references MOVIE_DETAILS(movie_id),  
  6  primary key(movie_id, start_time, end_time));
```

Table created.

```
SQL> desc MOVIE_TIMINGS;
```

Name	Null?	Type
MOVIE_ID	NOT NULL	VARCHAR2(10)
START_TIME	NOT NULL	VARCHAR2(10)
END_TIME	NOT NULL	VARCHAR2(10)

```
SQL> insert into MOVIE_TIMINGS values('M-01', '10:30 am', '12:44 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-01', '4:00 pm', '6:14 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-02', '10:30 am', '12:55 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-02', '4:00 pm', '6:25 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-03', '10:30 am', '1:20 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-03', '4:00 pm', '6:50 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-04', '10:30 am', '1:00 pm');
```

1 row created.

```
SQL> insert into MOVIE_TIMINGS values('M-04', '10:30 am', '1:00 pm');
1 row created.

SQL> insert into MOVIE_TIMINGS values('M-04', '4:00 pm', '6:30 pm');
1 row created.

SQL> insert into MOVIE_TIMINGS values('M-05', '10:30 am', '1:30 pm');
1 row created.

SQL> insert into MOVIE_TIMINGS values('M-05', '4:00 pm', '7:00 pm');
1 row created.
```

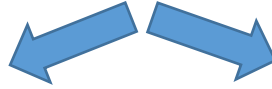
```
SQL> select * from MOVIE_TIMINGS;
```

MOVIE_ID	START_TIME	END_TIME
M-01	10:30 am	12:44 pm
M-01	4:00 pm	6:14 pm
M-02	10:30 am	12:55 pm
M-02	4:00 pm	6:25 pm
M-03	10:30 am	1:20 pm
M-03	4:00 pm	6:50 pm
M-04	10:30 am	1:00 pm
M-04	4:00 pm	6:30 pm
M-05	10:30 am	1:30 pm
M-05	4:00 pm	7:00 pm

```
10 rows selected.
```

## THEATRE DETAILS TABLE

1. MOVIE_ID
2. THEATRE_ID
3. THEATRE_NAME
4. LOCATION
5. NO OF SCREENS



1. THEATRE_ID
2. THEATRE_NAME
3. LOCATION
4. NO OF SCREENS

1. MOVIE_ID
2. THEATRE_ID

As one theatre can have multiple movies the theatre details table is divided into 2 partitioned tables. The above 2 isolated tables are **THEATRE TABLE** AND **SHOWS TABLE**.

### THEATRE TABLE:

#### LIST OF ATTRIBUTES:

- ✓ theatre\_id VARCHAR2(10)
- ✓ theatre\_name VARCHAR2(10)
- ✓ location VARCHAR2(15)
- ✓ NoOfScreen NUMBER(1)

**PRIMARY KEY:** theatre\_id

```
SQL> create table THEATRE(  
2  theatre_id varchar2(10) primary key,  
3  theatre_name varchar2(10) NOT NULL,  
4  location varchar2(25) NOT NULL,  
5  NoOfScreens number(1) NOT NULL);
```

Table created.

```
SQL> desc THEATRE;
```

Name	Null?	Type
THEATRE_ID	NOT NULL	VARCHAR2(10)
THEATRE_NAME	NOT NULL	VARCHAR2(10)
LOCATION	NOT NULL	VARCHAR2(25)
NOOFScreens	NOT NULL	NUMBER(1)

```
SQL> insert into THEATRE values('T-01', 'IMAX', 'Khairtabad', 2);
```

1 row created.

```
SQL> insert into THEATRE values('T-02', 'INOX', 'Banjara Hills', 2);  
1 row created.  
SQL> insert into THEATRE values('T-03', 'AMBCinemas', 'Gachibowli', 2);  
1 row created.
```

```
SQL> select * from THEATRE;
```

THEATRE_ID	THEATRE_NA	LOCATION	NOOFSCREENS
T-01	IMAX	Khairtabad	2
T-02	INOX	Banjara Hills	2
T-03	AMBCinemas	Gachibowli	2

## SHOWS TABLE:

### LIST OF ATTRIBUTES:

- ✓ theatre\_id VARCHAR2(10)
- ✓ movie\_id VARCHAR2(10)

**PRIMARY KEY:** (theatre\_id, movie\_id)

**FOREIGN KEY:** theatre\_id, movie\_id

```
SQL> create table SHOWS(  
2  theatre_id varchar2(10),  
3  movie_id varchar2(10),  
4  foreign key(theatre_id) references THEATRE(theatre_id),  
5  foreign key(movie_id) references MOVIE_DETAILS(movie_id),  
6  primary key(theatre_id, movie_id));
```

Table created.

```
SQL> desc SHOWS
```

Name	Null?	Type
-----	-----	-----
THEATRE_ID	NOT NULL	VARCHAR2(10)
MOVIE_ID	NOT NULL	VARCHAR2(10)

```
SQL> insert into SHOWS values('T-01', 'M-02');
```

1 row created.

```
SQL> insert into SHOWS values('T-01', 'M-03');
```

1 row created.

```
SQL> insert into SHOWS values('T-01', 'M-04');
```

1 row created.

```
SQL> insert into SHOWS values('T-02', 'M-01');
```

1 row created.

```
SQL> insert into SHOWS values('T-02', 'M-03');
```

1 row created.

```
SQL> insert into SHOWS values('T-02', 'M-05');
```

1 row created.

```
SQL> insert into SHOWS values('T-03', 'M-01');
```

1 row created.

```
SQL> insert into SHOWS values('T-03', 'M-02');
```

1 row created.

```
SQL> insert into SHOWS values('T-03', 'M-04');
```

1 row created.

```
SQL> select * from SHOWS;
```

THEATRE_ID	MOVIE_ID
-----	-----
T-01	M-02
T-01	M-03
T-01	M-04
T-02	M-01
T-02	M-03
T-02	M-05
T-03	M-01
T-03	M-02
T-03	M-04

9 rows selected.



## SCREEN TABLE

1. SCREEN_ID
2. THEATRE_ID
3. NO OF SEATS

The above **SCREEN** table is already in 3<sup>rd</sup> Normal Form.

### SCREEN TABLE:

#### LIST OF ATTRIBUTES:

- ✓ screen\_id VARCHAR2(10)
- ✓ theatre\_id VARCHAR2(10)
- ✓ NoOfSeats NUMBER(3)

**PRIMARY KEY:** (screen\_id, theatre\_id)

**FOREIGN KEY:** theatre\_id

```
SQL> create table SCREEN(  
2 screen_id varchar2(10),  
3 theatre_id varchar2(10),  
4 NoOfSeats number(3) NOT NULL,  
5 foreign key(theatre_id) references THEATRE(theatre_id),  
6 primary key(screen_id, theatre_id));
```

Table created.

```
SQL> desc SCREEN;
```

Name	Null?	Type
SCREEN_ID	NOT NULL	VARCHAR2(10)
THEATRE_ID	NOT NULL	VARCHAR2(10)
NOOFSEATS	NOT NULL	NUMBER(3)

```
SQL> insert into SCREEN values('SC-01', 'T-01', 15);  
1 row created.
```

```
SQL> insert into SCREEN values('SC-02', 'T-01', 15);  
1 row created.
```

```
SQL> insert into SCREEN values('SC-01', 'T-02', 15);  
1 row created.
```

```
SQL> insert into SCREEN values('SC-02', 'T-02', 15);  
1 row created.
```

```
SQL> insert into SCREEN values('SC-01', 'T-03', 15);  
1 row created.
```

```
SQL> insert into SCREEN values('SC-02', 'T-03', 15);  
1 row created.
```

```
SQL> select * from SCREEN;
```

SCREEN_ID	THEATRE_ID	NOOFSEATS
SC-01	T-01	15
SC-02	T-01	15
SC-01	T-02	15
SC-02	T-02	15
SC-01	T-03	15
SC-02	T-03	15

6 rows selected.



## SEAT TABLE

1. SEAT_NO
2. SCREEN_ID
3. THEATRE_ID

The above **SEAT** table is already in 3<sup>rd</sup> Normal Form.

### SEAT TABLE:

#### LIST OF ATTRIBUTES:

- ✓ seat\_no VARCHAR2(10)
- ✓ screen\_id VARCHAR2(10)
- ✓ theatre\_id VARCHAR2(10)

**PRIMARY KEY:** (seat\_no, screen\_id, theatre\_id)

**FOREIGN KEY:** (screen\_id, theatre\_id)

```
SQL> create table SEAT(  
  2 seat_no varchar2(10),  
  3 screen_id varchar2(10),  
  4 theatre_id varchar2(10),  
  5 foreign key(screen_id, theatre_id) references SCREEN(screen_id, theatre_id),  
  6 primary key(seat_no, screen_id, theatre_id));
```

Table created.

```
SQL> desc SEAT;
```

Name	Null?	Type
SEAT_NO	NOT NULL	VARCHAR2(10)
SCREEN_ID	NOT NULL	VARCHAR2(10)
THEATRE_ID	NOT NULL	VARCHAR2(10)

```
SQL> insert into SEAT values('VIP-01', 'SC-01', 'T-01');
```

1 row created.

```
SQL> insert into SEAT values('VIP-02', 'SC-01', 'T-01');
```

1 row created.

```
SQL> insert into SEAT values('GOLD-01', 'SC-01', 'T-01');
```

1 row created.

```
SQL> insert into SEAT values('GOLD-02', 'SC-01', 'T-01');
```

1 row created.

```
SQL> insert into SEAT values('REC-01', 'SC-01', 'T-01');
```

1 row created.

```
SQL> insert into SEAT values('REC-02', 'SC-01', 'T-01');
```

1 row created.

```
SQL> select * from SEAT;
```

SEAT_NO	SCREEN_ID	THEATRE_ID
VIP-01	SC-01	T-01
VIP-02	SC-01	T-01
GOLD-01	SC-01	T-01
GOLD-02	SC-01	T-01
REC-01	SC-01	T-01
REC-02	SC-01	T-01
VIP-01	SC-02	T-01
VIP-02	SC-02	T-01
GOLD-01	SC-02	T-01
GOLD-02	SC-02	T-01
REC-01	SC-02	T-01

SEAT_NO	SCREEN_ID	THEATRE_ID
REC-02	SC-02	T-01
VIP-01	SC-01	T-02
VIP-02	SC-01	T-02
GOLD-01	SC-01	T-02
GOLD-02	SC-01	T-02
REC-01	SC-01	T-02
REC-02	SC-01	T-02
VIP-01	SC-02	T-02
VIP-02	SC-02	T-02
GOLD-01	SC-02	T-02
GOLD-02	SC-02	T-02

SEAT_NO	SCREEN_ID	THEATRE_ID
REC-01	SC-02	T-02
REC-02	SC-02	T-02
VIP-01	SC-01	T-03
VIP-02	SC-01	T-03
GOLD-01	SC-01	T-03
GOLD-02	SC-01	T-03
REC-01	SC-01	T-03
REC-02	SC-01	T-03
VIP-01	SC-02	T-03
VIP-02	SC-02	T-03
GOLD-01	SC-02	T-03

SEAT_NO	SCREEN_ID	THEATRE_ID
GOLD-02	SC-02	T-03
REC-01	SC-02	T-03
REC-02	SC-02	T-03

```
36 rows selected.
```

## TICKET TABLE

1. TICKET_ID
2. CUSTOMER_ID
3. ADMIN_ID
4. MOVIE_ID
5. START_TIME
6. END_TIME
7. THEATRE_ID
8. SCREEN_ID
9. SEAT_NO
10. PRICE

### TICKETS TABLE:

#### LIST OF ATTRIBUTES:

- ✓ ticket\_id VARCHAR2(10)
- ✓ customer\_id VARCHAR2(10)
- ✓ admin\_id VARCHAR2(10)
- ✓ movie\_id VARCHAR2(10)
- ✓ start\_time VARCHAR2(10)
- ✓ end\_time VARCHAR2(10)
- ✓ theatre\_id VARCHAR2(10)
- ✓ screen\_id VARCHAR2(10)
- ✓ seat\_no VARCHAR2(10)
- ✓ price NUMBER

**PRIMARY KEY:** ticket\_id

**FOREIGN KEYS:** customer\_id, admin\_id, (movie\_id, start\_time, end\_time), (seat\_no, screen\_id, theatre\_id)

```
SQL> create table TICKETS(  
 2  ticket_id varchar2(10) primary key,  
 3  customer_id varchar2(10),  
 4  admin_id varchar2(10),  
 5  movie_id varchar2(10),  
 6  start_time varchar2(10),  
 7  end_time varchar2(10),  
 8  theatre_id varchar2(10),  
 9  screen_id varchar2(10),  
10  seat_no varchar2(10),  
11  price number NOT NULL,  
12  foreign key(customer_id) references CUSTOMER(customer_id),  
13  foreign key(admin_id) references ADMIN(admin_id),  
14  foreign key(movie_id, start_time, end_time) references MOVIE_TIMINGS(movie_id, start_time, end_time),  
15  foreign key(seat_no, screen_id, theatre_id) references SEAT(seat_no, screen_id, theatre_id));
```

Table created.

```
SQL> desc TICKETS;
```

Name	Null?	Type
TICKET_ID	NOT NULL	VARCHAR2(10)
CUSTOMER_ID		VARCHAR2(10)
ADMIN_ID		VARCHAR2(10)
MOVIE_ID		VARCHAR2(10)
START_TIME		VARCHAR2(10)
END_TIME		VARCHAR2(10)
THEATRE_ID		VARCHAR2(10)
SCREEN_ID		VARCHAR2(10)
SEAT_NO		VARCHAR2(10)
PRICE	NOT NULL	NUMBER

```
SQL> insert into TICKETS values('TIK-01', 'C-02', 'System', 'M-02', '10:30 am', '12:55 pm', 'T-03', 'SC-01', 'REC-02', 750.00);
```

```
1 row created.
```

```
SQL> insert into TICKETS values('TIK-02', 'C-03', 'System', 'M-03', '4:00 pm', '6:50 pm', 'T-01', 'SC-01', 'VIP-01', 350.00);
```

```
1 row created.
```

```
SQL> insert into TICKETS values('TIK-03', 'C-04', 'System', 'M-01', '10:30 am', '12:44 pm', 'T-02', 'SC-02', 'GOLD-02', 500.00);
```

```
1 row created.
```

```
SQL> select * from TICKETS;
```

TICKET_ID	CUSTOMER_I	ADMIN_ID	MOVIE_ID	START_TIME	END_TIME	THEATRE_ID
SCREEN_ID	SEAT_NO	PRICE				
TIK-01	C-02	System	M-02	10:30 am	12:55 pm	T-03
SC-01	REC-02	750				
TIK-02	C-03	System	M-03	4:00 pm	6:50 pm	T-01
SC-01	VIP-01	350				
TIK-03	C-04	System	M-01	10:30 am	12:44 pm	T-02
SC-02	GOLD-02	500				

## **CODE:**

### **HomePage.java**

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.UIManager;

public class HomePage extends JFrame implements ActionListener {
    private JLabel background;
    private ImageIcon img;

    private JButton customer;
    private JButton admin;

    private JMenuBar mbr;
    private JMenu file;
    private JMenuItem about;
    private JMenuItem credentials;
    private JMenuItem quit;

    private Container cp;

    public HomePage() {
        cp = getContentPane();

        img = new ImageIcon("HomePage.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 654, 469);

        customer = new JButton("CUSTOMER");
        customer.setBounds(180, 270, 100, 25);
        admin = new JButton("ADMIN");
        admin.setBounds(359, 270, 100, 25);

        mbr = new JMenuBar();
        setJMenuBar(mbr);

        file = new JMenu("FILE");
        mbr.add(file);
        file.add(about = new JMenuItem("About"));
        file.add(credentials = new JMenuItem("Credentials"));
        file.add("-----");
        file.add(quit = new JMenuItem("Quit"));

        customer.addActionListener(this);
        admin.addActionListener(this);
        about.addActionListener(this);
        credentials.addActionListener(this);
        quit.addActionListener(this);
    }
}
```

```

        cp.add(background);
        background.add(customer);
        background.add(admin);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(654, 469);
        setVisible(true);
        setTitle("HOME PAGE");
    }

    public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        if (arg.equals("About")) {
            String projectInfo = ("MOVIEBUZZ is a Desktop application for booking cinema
tickets." + "\n" +
                                "The objective of MOVIEBUZZ is to create an application for Movie ticket
" + "\n" +
                                "booking that allows users to know about new
movies, cinema locations, " + "\n" +
                                "available theatres, their schedules, class, and
ticket price." + "\n" + "\n" +
                                "
                                THANK YOU!!");
            JOptionPane.showMessageDialog(this, projectInfo, "INFORMATION",
JOptionPane.INFORMATION_MESSAGE);
        }
        else if (arg.equals("Credentials")) {
            String credentialsInfo = ("Name: V. Rishik, ROLL NO: 1602-19-737-156, Section: IT-C"
+ "\n" +
                                "Name: K. Rithik, ROLL NO: 1602-19-
737-157, Section: IT-C" + "\n" +
                                "Name: B. Shailendhar, ROLL NO: 1602-
19-737-167, Section: IT-C" + "\n");
            JOptionPane.showMessageDialog(this, credentialsInfo, "INFORMATION",
JOptionPane.INFORMATION_MESSAGE);
        }
        else if (arg.equals("Quit")) {
            System.exit(0);
        }
        else if (arg.equals("CUSTOMER")) {
            dispose();
            new CustomerPage();
        }
        else if (ae.getActionCommand().equals("ADMIN")) {
            dispose();
            new AdminLogin();
        }
    }

    public static void main(String[] args) {
        try {

```

```
        UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
    }
    catch(Exception e) {
        System.out.println("Look and Feel not set");
    }

        new HomePage();
    }
}
```

### CustomerPage.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.UIManager;

public class CustomerPage extends JFrame implements ActionListener {
    private JLabel background;
    private ImageIcon img;

    private JButton register;
    private JButton login;
    private JButton goback;

    private Container cp;

    public CustomerPage() {
        cp = getContentPane();

        img = new ImageIcon("CustomerPage.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 738, 416);

        register = new JButton("REGISTER");
        register.setBounds(500, 160, 100, 25);
        login = new JButton("LOGIN");
        login.setBounds(500, 190, 100, 25);
        goback = new JButton("GOBACK");
        goback.setBounds(500, 220, 100, 25);

        register.addActionListener(this);
        login.addActionListener(this);
        goback.addActionListener(this);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }

        cp.add(background);
        background.add(register);
        background.add(login);
        background.add(goback);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(738, 416);
        setVisible(true);
        setTitle("CUSTOMER PAGE");
    }
}
```



```
public void actionPerformed(ActionEvent ae) {  
    String arg = ae.getActionCommand();  
    if (arg.equals("REGISTER")) {  
        dispose();  
        new Register();  
    }  
    else if (arg.equals("LOGIN")) {  
        dispose();  
        new Login();  
    }  
    else if (arg.equals("GOBACK")) {  
        dispose();  
        new HomePage();  
    }  
}
```

## **Register.java**

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.UIManager;
import java.util.*;
import java.util.regex.*;
import java.math.*;
import java.sql.*;

public class Register extends JFrame implements ActionListener {
    private JLabel heading;
    private JLabel fmand;
    private JLabel background;
    private ImageIcon img;

    private JButton submit;
    private JButton reset;
    private JButton goback;

    private JTextField firstName;
    private JTextField lastName;
    private JPasswordField password;
    private JPasswordField confirmPassword;
    private JTextField phoneNumber;

    private JLabel fnLabel;
    private JLabel ast1;
    private JLabel lnLabel;
    private JLabel ast2;
    private JLabel passLabel;
    private JLabel ast3;
    private JLabel cpassLabel;
    private JLabel ast4;
    private JLabel pnLabel;
    private JLabel ast5;

    private Container cp;

    public Register() {
        initializeComponents();
        registerListeners();
        addComponentsToFrame();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(1100, 732);
        setVisible(true);
        setTitle("REGISTRATION PAGE");
    }

    private void initializeComponents() {
        cp = getContentPane();
```

```
img = new ImageIcon("Register.jpg");

background = new JLabel("", img, JLabel.CENTER);
background.setBounds(0, 0, 1100, 732);

heading = new JLabel("SIGN UP");
heading.setForeground(Color.BLUE);
heading.setFont(new Font("Serif", Font.BOLD, 35));
heading.setBounds(455, 15, 150, 25);

fmand = new JLabel("(*) - ALL FIELDS ARE MANDATORY");
fmand.setForeground(Color.RED);
fmand.setFont(new Font("Serif", Font.BOLD, 18));
fmand.setBounds(375, 45, 325, 25);

fnLabel = new JLabel("Enter your first Name:");
ast1 = new JLabel("*");
fnLabel.setForeground(Color.BLACK);
ast1.setForeground(Color.RED);
fnLabel.setFont(new Font("Serif", Font.BOLD, 18));
ast1.setFont(new Font("Serif", Font.BOLD, 18));
fnLabel.setBounds(300, 90, 300, 25);
ast1.setBounds(475, 93, 10, 25);

firstName = new JTextField(15);
firstName.setBounds(550, 90, 150, 25);

lnLabel = new JLabel("Enter your last Name:");
ast2 = new JLabel("*");
lnLabel.setForeground(Color.BLACK);
ast2.setForeground(Color.RED);
lnLabel.setFont(new Font("Serif", Font.BOLD, 18));
ast2.setFont(new Font("Serif", Font.BOLD, 18));
lnLabel.setBounds(300, 140, 300, 25);
ast2.setBounds(475, 143, 10, 25);

lastName = new JTextField(15);
lastName.setBounds(550, 140, 150, 25);

passLabel = new JLabel("Enter your password:");
ast3 = new JLabel("*");
passLabel.setForeground(Color.BLACK);
ast3.setForeground(Color.RED);
passLabel.setFont(new Font("Serif", Font.BOLD, 18));
ast3.setFont(new Font("Serif", Font.BOLD, 18));
passLabel.setBounds(300, 190, 300, 25);
ast3.setBounds(475, 193, 10, 25);

password = new JPasswordField(15);
password.setBounds(550, 190, 150, 25);
```

```
cpassLabel = new JLabel("Re-enter passowrd:");
ast4 = new JLabel("*");
cpassLabel.setForeground(Color.BLACK);
ast4.setForeground(Color.RED);
cpassLabel.setFont(new Font("Serif", Font.BOLD, 18));
ast4.setFont(new Font("Serif", Font.BOLD, 18));
cpassLabel.setBounds(300, 240, 300, 25);
ast4.setBounds(475, 243, 10, 25);
```

```
confirmPassword = new JPasswordField(15);
confirmPassword.setBounds(550, 240, 150, 25);
```

```
pnLabel = new JLabel("Enter phone number:");
ast5 = new JLabel("*");
pnLabel.setForeground(Color.BLACK);
ast5.setForeground(Color.RED);
pnLabel.setFont(new Font("Serif", Font.BOLD, 18));
ast5.setFont(new Font("Serif", Font.BOLD, 18));
pnLabel.setBounds(300, 290, 300, 25);
ast5.setBounds(475, 293, 10, 25);
```

```
phoneNumber = new JTextField(15);
phoneNumber.setBounds(550, 290, 150, 25);
```

```
goback = new JButton("GOBACK");
goback.setBounds(300, 390, 100, 25);
submit = new JButton("SUBMIT");
submit.setBounds(450, 390, 100, 25);
reset = new JButton("RESET");
reset.setBounds(600, 390, 100, 25);
```

```
}
```

```
private void registerListeners() {
    submit.addActionListener(this);
    reset.addActionListener(this);
    goback.addActionListener(this);
```

```
    try {
        UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
```

```
    }
```

```
catch(Exception e) {
    System.out.println("Look and Feel not set");
}
}
```

```
private void addComponentsToFrame() {
    cp.add(background);
```

```
    background.add(heading);
    background.add(fmand);
    background.add(fnLabel);
    background.add(ast1);
```

```

background.add(firstName);
background.add(lnLabel);
background.add(ast2);
background.add(lastName);
background.add(passLabel);
background.add(ast3);
background.add(password);
background.add(cpassLabel);
background.add(ast4);
background.add(confirmPassword);
background.add(pnLabel);
background.add(ast5);
background.add(phoneNumber);
background.add(goback);
background.add(submit);
background.add(reset);
}

```

```

public void actionPerformed(ActionEvent ae) {
    if (ae.getActionCommand().equals("GOBACK")) {
        dispose();
        new CustomerPage();
    }
    else if (ae.getActionCommand().equals("RESET")) {
        firstName.setText("");
        lastName.setText("");
        password.setText("");
        confirmPassword.setText("");
        phoneNumber.setText("");
    }
    else if (ae.getActionCommand().equals("SUBMIT")) {
        String name, pass, cpass;
        name = "";
        BigDecimal pn = null;

        try {
            pass = String.valueOf(password.getPassword());
            cpass = String.valueOf(confirmPassword.getPassword());
            if (!phoneNumber.getText().equals(""))
                pn = new BigDecimal(phoneNumber.getText());

            String fname = firstName.getText();
            fname = fname.replaceAll("( )+", " ");
            fname = fname.trim();

            if ((name.equals(" ")) || (pass.equals("")) || (cpass.equals("")) ||
(phoneNumber.getText().equals(""))) {
                JOptionPane.showMessageDialog(this, "ENTER COMPLETE
DETAILS", "ERROR", JOptionPane.ERROR_MESSAGE);
            }
            else if ((fname.equals("")) || (fname.equals(null)) || (!fname.matches("[a-zA-Z]+(
[a-zA-z]+)*$"))) {

```

```

JOptionPane.showMessageDialog(this, "FIRST NAME SHOULD BE
ENTERED PROPERLY" + "\n" + "ALLOWED CHARACTERS ARE 'a-z' (or) 'A-Z'", "WARNING",
JOptionPane.WARNING_MESSAGE);
    }
    else if ((lastName.getText().equals("")) || (lastName.getText() == null) ||
(!lastName.getText().matches("^([a-zA-Z]*$)")) {
JOptionPane.showMessageDialog(this, "LAST NAME SHOULD BE
ENTERED PROPERLY" + "\n" + "ALLOWED CHARACTERS ARE 'a-z' (or) 'A-Z'", "WARNING",
JOptionPane.WARNING_MESSAGE);
    }
    else if (String.valueOf(password.getPassword()).length() < 8) {
JOptionPane.showMessageDialog(this, "PASSWORD SHOULD BE
ATLEAST 8 CHARACTERS", "WARNING", JOptionPane.WARNING_MESSAGE);
    }
    else if (!pass.equals(cpass)) {
JOptionPane.showMessageDialog(this, "ENTERED PASSWORD AND
CONFIRMATION PASSWORD DIDN'T MATCHED", "WARNING",
JOptionPane.WARNING_MESSAGE);
    }
    else if (phoneNumber.getText().length() != 10) {
JOptionPane.showMessageDialog(this, "PHONE NUMBER MUST BE
10 DIGITS", "WARNING", JOptionPane.WARNING_MESSAGE);
    }
    else if (phoneNumber.getText().charAt(0) <= 53) {
JOptionPane.showMessageDialog(this, "STARTING DIGIT OF THE
PHONE NUMBER SHOULD BE 6, 7, 8 (or) 9", "WARNING", JOptionPane.WARNING_MESSAGE);
    }
    else {
        name = fname + " " + lastName.getText();
        String customerID = null;

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "rishik", "rishikv");

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("select count(*) from
CUSTOMER");

            rs.next();

            customerID = "C-";
            int idNum = rs.getInt(1) + 1;
            if (idNum < 10)
                customerID = customerID + "0" + idNum;
            else
                customerID += idNum;

            PreparedStatement psmt = con.prepareStatement("insert into
CUSTOMER values(?, ?, ?, ?)");

            psmt.setString(1, customerID);
            psmt.setString(2, name);

```

```

        psmt.setString(3, pass);
        psmt.setBigDecimal(4, pn);

        psmt.executeUpdate();

        String msg = ("DETAILS OF THE CUSTOMER" + "\n" +
            "Customer's Name: " + name + "\n" +
            "Customer's Password: " + pass + "\n" +
            "Customer's Phone Number: " + pn + "\n"
+
            "CUSTOMER ID: " + customerID);
        JOptionPane.showMessageDialog(this, msg, "INFORMATION",
JOptionPane.INFORMATION_MESSAGE);
        con.close();
    }
    catch (Exception e) {
        e.printStackTrace();
    }
}
}
catch (NumberFormatException e) {
    JOptionPane.showMessageDialog(this, "PHONE NUMBER SHOULD NOT
CONTAIN ANY ALPHABETS", "WARNING", JOptionPane.WARNING_MESSAGE);
}
catch (NullPointerException e) {
    JOptionPane.showMessageDialog(this, "FIRST NAME SHOULD NOT BE
EMPTY" + "\n" + "ALLOWED CHARACTERS ARE 'a-z' (or) 'A-Z'", "WARNING",
JOptionPane.WARNING_MESSAGE);
}
}
}
}

```

## Login.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.UIManager;
import java.util.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener {
    private JLabel heading;
    private JLabel background;
    private ImageIcon img;

    private JButton submit;
    private JButton goback;

    private JTextField userName;
    private JPasswordField password;

    private JLabel userLabel;
    private JLabel passLabel;

    private Container cp;

    public Login() {
        initComponents();
        registerListeners();
        addComponentsToFrame();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(540, 360);
        setVisible(true);
        setTitle("LOGIN PAGE");
    }

    public void initComponents() {
        cp = getContentPane();

        img = new ImageIcon("Login.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 540, 360);

        heading = new JLabel("SIGN IN");
        heading.setFont(new Font("Serif", Font.BOLD, 18));
        heading.setForeground(Color.RED);
        heading.setBounds(300, 15, 75, 30);

        userLabel = new JLabel("Enter your userID: ");
        userLabel.setFont(new Font("Times New Roman", Font.BOLD, 18));
        userLabel.setForeground(Color.WHITE);
        userLabel.setBounds(200, 45, 200, 75);
```



```

        passLabel = new JLabel("Enter your password: ");
        passLabel.setFont(new Font("Times New Roman", Font.BOLD, 18));
        passLabel.setForeground(Color.WHITE);
        passLabel.setBounds(200, 90, 200, 75);

        userName = new JTextField(15);
        userName.setBounds(400, 68, 100, 25);

        password = new JPasswordField(15);
        password.setBounds(400, 113, 100, 25);

        submit = new JButton("SUBMIT");
        submit.setBounds(225, 175, 100, 30);
        goback = new JButton("GOBACK");
        goback.setBounds(375, 175, 100, 30);
    }

    private void registerListeners() {
        submit.addActionListener(this);
        goback.addActionListener(this);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }
    }

    private void addComponentsToFrame() {
        cp.add(background);
        background.add(heading);
        background.add(userLabel);
        background.add(userName);
        background.add(passLabel);
        background.add(password);
        background.add(submit);
        background.add(goback);
    }

    public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        if (arg.equals("GOBACK")) {
            dispose();
            new CustomerPage();
        }
        else if (arg.equals("SUBMIT")) {
            int flag = 0;
            String uname, pass;
            uname = userName.getText();
            pass = String.valueOf(password.getPassword());

```

```

        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "rishik", "rishikv");

            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery("select customer_id, password from
CUSTOMER");

            while(rs.next()) {
                if ((uname.equals(rs.getString(1))) && (pass.equals(rs.getString(2)))) {
                    JOptionPane.showMessageDialog(this, "LOGIN SUCCESSFUL",
"INFORMATION", JOptionPane.INFORMATION_MESSAGE);
                    flag = 1;
                    dispose();
                    new MoviePage();
                }
            }
            con.close();

            if (flag == 0)
                JOptionPane.showMessageDialog(this, "INCORRECT CREDENTIALS,
TRY AGAIN!!!", "WARNING", JOptionPane.WARNING_MESSAGE);
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

### **MoviePage.java**

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.UIManager;

public class MoviePage extends JFrame implements ActionListener {
    private JLabel background;
    private ImageIcon img;

    private JLabel heading;

    private JButton next;
    private JButton goback;

    private Container cp;

    public MoviePage() {
        cp = getContentPane();

        img = new ImageIcon("MoviePage.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 609, 405);

        next = new JButton("CONTINUE");
        next.setBounds(255, 330, 100, 25);

        goback = new JButton("GOBACK");
        goback.setBounds(260, 300, 85, 25);

        next.addActionListener(this);
        goback.addActionListener(this);

        cp.add(background);
        background.add(next);
        background.add(goback);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(609, 405);
        setVisible(true);
        setTitle("MOVIE PAGE");
    }

    public void actionPerformed(ActionEvent ae) {
```

```
String arg = ae.getActionCommand();
if (arg.equals("CONTINUE")) {
    dispose();
}
else if (arg.equals("GOBACK")) {
    dispose();
    new CustomerPage();
}
}
```

### MovieSelection.java

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import javax.swing.UIManager;
import java.sql.*;

public class MovieSelection extends JFrame implements ActionListener {
    private JLabel background;
    private ImageIcon img;

    private JLabel heading;

    private Choice cmovie;

    private JButton show;
    private JButton goback;

    private JTable table;

    private JScrollPane scrollPane;

    private Container cp;

    public MovieSelection() {
        initComponents();
        registerListeners();
        addComponentsToFrame();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(960, 350);
        setVisible(true);
        setTitle("MOVIE SELECTION PAGE");
    }

    public void initComponents() {
        cp = getContentPane();

        img = new ImageIcon("MovieSelection.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 960, 350);

        heading = new JLabel("MOVIE SELECTION");
        heading.setFont(new Font("Serif", Font.BOLD, 22));
        heading.setForeground(Color.RED);
        heading.setBounds(350, 50, 225, 30);

        cmovie = new Choice();
        cmovie.add("BLACK WIDOW");
        cmovie.add("Spider Man: NO WAY HOME");
        cmovie.add("K.G.F Chapter 2");
```

```

        cmovie.add("Jersey");
        cmovie.add("RRR");
        cmovie.setBounds(375, 90, 175, 25);

        show = new JButton("SHOW");
        show.setBounds(500, 200, 75, 25);
        goback = new JButton("GOBACK");
        goback.setBounds(300, 200, 100, 25);
    }

    private void registerListeners() {
        show.addActionListener(this);
        goback.addActionListener(this);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }
    }

    private void addComponentsToFrame() {
        cp.add(background);
        background.add(heading);
        background.add(cmovie);
        background.add(show);
        background.add(goback);
    }

    public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        if (arg.equals("GOBACK")) {
            dispose();
            new MoviePage();
        }
        else if (arg.equals("SHOW")) {
            String mname = cmovie.getItem(cmovie.getSelectedIndex());

            Object[][] rows = null;
            Object[] columns = {"TheatreID", "TheatreName", "Location", "NoOfScreens"};

            JFrame vTabel;
            JLabel mLabel;
            JTextField movie;

            int totalRows = 0, index = 0;

            try {
                Class.forName("oracle.jdbc.driver.OracleDriver");
            }

```

```

        Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "rishik", "rishikv");

        Statement stmt = con.createStatement();
        ResultSet rs = null;

        if (mname.equals("BLACK WIDOW")) {
            rs = stmt.executeQuery("select count(*) from THEATRE T,
MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'BLACK WIDOW' and S.movie_id =
MD.movie_id and T.theatre_id = S.theatre_id");
            rs.next();
            totalRows = rs.getInt(1);

            rs = stmt.executeQuery("select T.theatre_id, T.theatre_name, T.location,
T.NoOfScreens from THEATRE T, MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'BLACK
WIDOW' and S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");
            rows = new Object[totalRows][columns.length];
        }
        else if (mname.equals("Spider Man: NO WAY HOME")) {
            rs = stmt.executeQuery("select count(*) from THEATRE T,
MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'Spider Man: NO WAY HOME' and
S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");
            rs.next();
            totalRows = rs.getInt(1);

            rs = stmt.executeQuery("select T.theatre_id, T.theatre_name, T.location,
T.NoOfScreens from THEATRE T, MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'Spider
Man: NO WAY HOME' and S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");
            rows = new Object[totalRows][columns.length];
        }
        else if (mname.equals("K.G.F Chapter 2")) {
            rs = stmt.executeQuery("select count(*) from THEATRE T,
MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'K.G.F Chapter 2' and S.movie_id =
MD.movie_id and T.theatre_id = S.theatre_id");
            rs.next();
            totalRows = rs.getInt(1);

            rs = stmt.executeQuery("select T.theatre_id, T.theatre_name, T.location,
T.NoOfScreens from THEATRE T, MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'K.G.F
Chapter 2' and S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");
            rows = new Object[totalRows][columns.length];
        }
        else if (mname.equals("Jersey")) {
            rs = stmt.executeQuery("select count(*) from THEATRE T,
MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'Jersey' and S.movie_id = MD.movie_id and
T.theatre_id = S.theatre_id");
            rs.next();
            totalRows = rs.getInt(1);

            rs = stmt.executeQuery("select T.theatre_id, T.theatre_name, T.location,
T.NoOfScreens from THEATRE T, MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'Jersey' and
S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");

```

```

        rows = new Object[totalRows][columns.length];
    }
    if (mname.equals("RRR")) {
        rs = stmt.executeQuery("select count(*) from THEATRE T,
MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'RRR' and S.movie_id = MD.movie_id and
T.theatre_id = S.theatre_id");

        rs.next();
        totalRows = rs.getInt(1);

        rs = stmt.executeQuery("select T.theatre_id, T.theatre_name, T.location,
T.NoOfScreens from THEATRE T, MOVIE_DETAILS MD, SHOWS S where MD.movie_name = 'RRR' and
S.movie_id = MD.movie_id and T.theatre_id = S.theatre_id");
        rows = new Object[totalRows][columns.length];
    }

    while (index < totalRows) {
        rs.next();
        rows[index][0] = rs.getString(1);
        rows[index][1] = rs.getString(2);
        rows[index][2] = rs.getString(3);
        rows[index][3] = rs.getInt(4);

        index += 1;
    }

    table = new JTable(rows, columns);
    scrollPane = new JScrollPane(table);

    vTabel = new JFrame("LIST OF THEATRES");
    vTabel.add(scrollPane, "Center");
    vTabel.setVisible(true);
    vTabel.setSize(500, 250);

    con.close();
}
catch (Exception e) {
    e.printStackTrace();
}
}

}

public static void main(String[] args) {
    new MovieSelection();
}
}

```



### AdminLogin.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.UIManager;
import java.util.*;
import java.sql.*;

public class AdminLogin extends JFrame implements ActionListener {
    private JLabel heading;
    private JLabel background;
    private ImageIcon img;

    private JButton submit;
    private JButton goback;

    private JTextField userName;
    private JPasswordField password;

    private JLabel userLabel;
    private JLabel passLabel;

    private JMenuBar mbr;
    private JMenu file;
    private JMenuItem about;
    private JMenuItem credentials;

    private Container cp;

    public AdminLogin() {
        initComponents();
        registerListeners();
        addComponentsToFrame();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(612, 408);
        setVisible(true);
        setTitle("ADMIN PAGE");
    }

    public void initComponents() {
        cp = getContentPane();

        img = new ImageIcon("AdminLogin.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 612, 408);

        mbr = new JMenuBar();
        setJMenuBar(mbr);

        file = new JMenu("FILE");
```

```

        mbr.add(file);
        file.add(about = new JMenuItem("About"));
        file.add(credentials = new JMenuItem("Credentials"));

        heading = new JLabel("ADMIN SIGN IN");
        heading.setFont(new Font("Serif", Font.BOLD, 25));
        heading.setForeground(Color.RED);
        heading.setBounds(185, 15, 225, 30);

        userLabel = new JLabel("Enter your userID: ");
        userLabel.setFont(new Font("Times New Roman", Font.BOLD, 18));
        userLabel.setForeground(Color.BLACK);
        userLabel.setBounds(125, 45, 200, 75);

        passLabel = new JLabel("Enter your password: ");
        passLabel.setFont(new Font("Times New Roman", Font.BOLD, 18));
        passLabel.setForeground(Color.BLACK);
        passLabel.setBounds(105, 90, 200, 75);

        userName = new JTextField(15);
        userName.setBounds(300, 70, 125, 25);

        password = new JPasswordField(15);
        password.setBounds(300, 115, 125, 25);

        submit = new JButton("SUBMIT");
        submit.setBounds(125, 200, 100, 30);
        goback = new JButton("GOBACK");
        goback.setBounds(300, 200, 100, 30);
    }

    private void registerListeners() {
        submit.addActionListener(this);
        goback.addActionListener(this);
        about.addActionListener(this);
        credentials.addActionListener(this);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }
    }

    private void addComponentsToFrame() {
        cp.add(background);
        background.add(heading);
        background.add(userLabel);
        background.add(userName);
        background.add(passLabel);
        background.add(password);
    }

```

```

        background.add(submit);
        background.add(goback);
    }

    public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        if (arg.equals("About")) {
            String msg = ("Admin is the special USER in MOVIEBUZZ." + "\n" +
                "Admin can view all the details about theatres, " + "\n" +
                "movies and majorly can view all the customers " + "\n" +
                "and their respective bookings.");
            JOptionPane.showMessageDialog(this, msg, "INFORMATION",
JOptionPane.INFORMATION_MESSAGE);
        }
        else if (arg.equals("Credentials")) {
            String msg = ("Username: System" + "\n" +
                "Password: IT-156-157-167");
            JOptionPane.showMessageDialog(this, msg, "INFORMATION",
JOptionPane.INFORMATION_MESSAGE);
        }
        else if (arg.equals("SUBMIT")) {
            int flag = 0;
            String uname, pass;
            uname = userName.getText();
            pass = String.valueOf(password.getPassword());

            if ((uname.equalsIgnoreCase("System")) && (pass.equals("IT-156-157-167"))) {
                JOptionPane.showMessageDialog(this, "LOGIN SUCCESSFUL",
"INFORMATION", JOptionPane.INFORMATION_MESSAGE);
                dispose();
                new AdminView();
            }
            else {
                JOptionPane.showMessageDialog(this, "INCORRECT CREDENTIALS, TRY
AGAIN!!", "WARNING", JOptionPane.WARNING_MESSAGE);
            }
        }
        else if (arg.equals("GOBACK")) {
            dispose();
            new HomePage();
        }
    }
}

```

### AdminView.java

```
import java.awt.*;
import java.awt.event.*;
import java.awt.print.*;
import javax.swing.*;
import javax.swing.UIManager;
import java.sql.*;

public class AdminView extends JFrame implements ActionListener {
    private JLabel heading;
    private JLabel background;
    private ImageIcon img;

    private JLabel tList;

    private Choice cmovie;

    private JButton viewTable;
    private JButton goback;

    private JTable table;

    private JScrollPane scrollPane;

    private Container cp;

    public AdminView() {
        initializeComponents();
        registerListeners();
        addComponentsToFrame();

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(612, 382);
        setVisible(true);
        setTitle("VIEW TABLES");
    }

    public void initializeComponents() {
        cp = getContentPane();

        img = new ImageIcon("AdminView.jpg");

        background = new JLabel("", img, JLabel.CENTER);
        background.setBounds(0, 0, 612, 382);

        heading = new JLabel("VIEW TABLES");
        heading.setFont(new Font("Serif", Font.BOLD, 25));
        heading.setForeground(Color.RED);
        heading.setBounds(185, 15, 225, 30);

        tList = new JLabel("TABLES LIST");
        tList.setFont(new Font("Serif", Font.BOLD, 20));
```

```

        tList.setForeground(Color.BLUE);
        tList.setBounds(415, 15, 225, 30);

        cmovie = new Choice();
        cmovie.add("CUSTOMER");
        cmovie.add("BOOKINGS");
        cmovie.add("MOVIE_DETAILS");
        cmovie.add("MOVIE_TIMINGS");
        cmovie.add("THEATRE");
        cmovie.add("SHOWS");
        cmovie.add("SCREEN");
        cmovie.add("TICKETS");
        cmovie.setBounds(400, 50, 150, 25);

        viewTable = new JButton("VIEW TABLE");
        viewTable.setBounds(400, 200, 125, 25);

        goback = new JButton("GOBACK");
        goback.setBounds(415, 250, 100, 25);
    }

    public void registerListeners() {
        viewTable.addActionListener(this);
        goback.addActionListener(this);

        try {
            UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
        }
        catch(Exception e) {
            System.out.println("Look and Feel not set");
        }
    }

    public void addComponentsToFrame() {
        cp.add(background);
        background.add(heading);
        background.add(cmovie);
        background.add(tList);
        background.add(viewTable);
        background.add(goback);
    }

    public void actionPerformed(ActionEvent ae) {
        String arg = ae.getActionCommand();
        if (arg.equals("GOBACK")) {
            dispose();
            new AdminLogin();
        }
        else if (arg.equals("VIEW TABLE")) {
            String s = cmovie.getItem(cmovie.getSelectedIndex());
            Object[][] rows;
            JFrame vTabel;

```

```

int index = 0, totalRows;
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "rishik", "rishikv");

    Statement stmt = con.createStatement();

    if (s.equals("CUSTOMER")) {
        ResultSet rs = stmt.executeQuery("select count(*) from CUSTOMER");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"CustomerID", "CustomerName", "Password",
"PhoneNumber"};

        rs = stmt.executeQuery("select * from CUSTOMER");
        rows = new Object[totalRows][columns.length];

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);
            rows[index][2] = rs.getString(3);
            rows[index][3] = rs.getString(4);

            index += 1;
        }

        table = new JTable(rows, columns);
        scrollPane = new JScrollPane(table);

        vTabel = new JFrame("CUSTOMER TABLE");
        vTabel.add(scrollPane, "Center");
        vTabel.setVisible(true);
        vTabel.setSize(650, 250);
    }
    else if (s.equals("BOOKINGS")) {
        ResultSet rs = stmt.executeQuery("select count(*) from BOOKINGS");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"BookingID", "CustomerID", "BookingDate",
"MovieDate", "TicketID"};

        rs = stmt.executeQuery("select * from BOOKINGS");
        rows = new Object[totalRows][columns.length];

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);
            rows[index][2] = rs.getDate(3);
            rows[index][3] = rs.getDate(4);

```

```

        rows[index][4] = rs.getString(5);

        index += 1;
    }

    table = new JTable(rows, columns);
    scrollPane = new JScrollPane(table);

    vTabel = new JFrame("BOOKINGS TABLE");
    vTabel.add(scrollPane, "Center");
    vTabel.setVisible(true);
    vTabel.setSize(650, 250);
}
else if (s.equals("MOVIE_DETAILS")) {
    ResultSet rs = stmt.executeQuery("select count(*) from
MOVIE_DETAILS");

    rs.next();
    totalRows = rs.getInt(1);

    Object[] columns = {"MovieID", "MovieName", "Language",
"Duration"};

    rs = stmt.executeQuery("select * from MOVIE_DETAILS");
    rows = new Object[totalRows][columns.length];

    while (index < totalRows) {
        rs.next();
        rows[index][0] = rs.getString(1);
        rows[index][1] = rs.getString(2);
        rows[index][2] = rs.getString(3);
        rows[index][3] = rs.getString(4);

        index += 1;
    }

    table = new JTable(rows, columns);
    scrollPane = new JScrollPane(table);

    vTabel = new JFrame("MOVIE_DETAILS TABLE");
    vTabel.add(scrollPane, "Center");
    vTabel.setVisible(true);
    vTabel.setSize(650, 250);
}
else if (s.equals("MOVIE_TIMINGS")) {
    ResultSet rs = stmt.executeQuery("select count(*) from
MOVIE_TIMINGS");

    rs.next();
    totalRows = rs.getInt(1);

    Object[] columns = {"MovieID", "StartTime", "EndTime"};
    rs = stmt.executeQuery("select * from MOVIE_TIMINGS");
    rows = new Object[totalRows][columns.length];

```

```

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);
            rows[index][2] = rs.getString(3);

            index += 1;
        }

        table = new JTable(rows, columns);
        scrollPane = new JScrollPane(table);

        vTabel = new JFrame("MOVIE_TIMINGS TABLE");
        vTabel.add(scrollPane, "Center");
        vTabel.setVisible(true);
        vTabel.setSize(500, 250);
    }
    else if (s.equals("THEATRE")) {
        ResultSet rs = stmt.executeQuery("select count(*) from THEATRE");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"TheatreID", "TheatreName", "Location",
"NoOfScreens"};

        rs = stmt.executeQuery("select * from THEATRE");
        rows = new Object[totalRows][columns.length];

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);
            rows[index][2] = rs.getString(3);
            rows[index][3] = rs.getString(4);

            index += 1;
        }

        table = new JTable(rows, columns);
        scrollPane = new JScrollPane(table);

        vTabel = new JFrame("THEATRE TABLE");
        vTabel.add(scrollPane, "Center");
        vTabel.setVisible(true);
        vTabel.setSize(500, 250);
    }
    else if (s.equals("SHOWS")) {
        ResultSet rs = stmt.executeQuery("select count(*) from SHOWS");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"TheatreID", "MovieID"};
        rs = stmt.executeQuery("select * from SHOWS");

```



```

        rows = new Object[totalRows][columns.length];

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);

            index += 1;
        }

        table = new JTable(rows, columns);
        scrollPane = new JScrollPane(table);

        vTabel = new JFrame("SHOWS TABLE");
        vTabel.add(scrollPane, "Center");
        vTabel.setVisible(true);
        vTabel.setSize(500, 250);
    }
    else if (s.equals("SCREEN")) {
        ResultSet rs = stmt.executeQuery("select count(*) from SCREEN");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"ScreenID", "TheatreID", "NoOfSeats"};
        rs = stmt.executeQuery("select * from SCREEN");
        rows = new Object[totalRows][columns.length];

        while (index < totalRows) {
            rs.next();
            rows[index][0] = rs.getString(1);
            rows[index][1] = rs.getString(2);
            rows[index][2] = rs.getString(3);

            index += 1;
        }

        table = new JTable(rows, columns);
        scrollPane = new JScrollPane(table);

        vTabel = new JFrame("SCREEN TABLE");
        vTabel.add(scrollPane, "Center");
        vTabel.setVisible(true);
        vTabel.setSize(650, 250);
    }
    else if (s.equals("TICKETS")) {
        ResultSet rs = stmt.executeQuery("select count(*) from TICKETS");
        rs.next();
        totalRows = rs.getInt(1);

        Object[] columns = {"TicketID", "CustomerID", "AdminID", "MovieID",
"StartTime", "EndTime", "TheatreID", "ScreenID", "SeatNo", "Price"};

```

```
rs = stmt.executeQuery("select * from TICKETS");
rows = new Object[totalRows][columns.length];
```

```
while (index < totalRows) {
    rs.next();
    rows[index][0] = rs.getString(1);
    rows[index][1] = rs.getString(2);
    rows[index][2] = rs.getString(3);
    rows[index][3] = rs.getString(4);
    rows[index][4] = rs.getString(5);
    rows[index][5] = rs.getString(6);
    rows[index][6] = rs.getString(7);
    rows[index][7] = rs.getString(8);
    rows[index][8] = rs.getString(9);
    rows[index][9] = rs.getString(10);
```

```
    index += 1;
```

```
}
```

```
table = new JTable(rows, columns);
scrollPane = new JScrollPane(table);
```

```
vTabel = new JFrame("TICKETS TABLE");
vTabel.add(scrollPane, "Center");
vTabel.setVisible(true);
vTabel.setSize(1000, 500);
```

```
}
```

```
}
```

```
catch (Exception e) {
    e.printStackTrace();
```

```
}
```

```
}
```

```
}
```

```
}
```

## OUTPUT SCREENSHOTS:

### CUSTOMER - SCREENSHOTS:



## SIGN UP

(\*) - ALL FIELDS ARE MANDATORY

Enter your first Name: \*

Enter your last Name: \*

Enter your password: \*

Re-enter passowrd: \*

Enter phone number: \*

GOBACK

SUBMIT

RESET

## SIGN IN

Enter your userID:

Enter your password:

SUBMIT

GOBACK



LOGIN SUCCESSFUL

OK





## ADMIN - SCREENSHOTS:

ADMIN PAGE


FILE

### ADMIN SIGN IN

Enter your userID:

Enter your password:

INFORMATION

 LOGIN SUCCESSFUL

ADMIN PAGE

### VIEW TABLES

### TABLES LIST

## VIEW TABLES



## TABLES LIST

CUSTOMER

CUSTOMER

BOOKINGS

MOVIE\_DETAILS

MOVIE\_TIMINGS

**THEATRE**

SHOWS

SCREEN

TICKETS

VIEW TABLE

GOBACK

## VIEW TABLES



## TABLES LIST

THEATRE

VIEW TABLE

GOBACK

TheatreID	TheatreName	Location	NoOfScreens
T-01	IMAX	Khairtabad	2
T-02	INOX	Banjara Hills	2
T-03	AMBCinemas	Gachibowli	2

## **CONCLUSION:**

In closing, **MovieBuzz** is a Desktop application and the main intention is for booking cinema tickets. Using this application, Customers can book their cinema tickets in effortless manner rather than standing in a queue for booking the tickets.

MovieBuzz is presently a Desktop application which uses ORACLE 11g as the DataBase.

Additional Changes to this application in the future version will be as follows:

- ✓ This Desktop application will be converted to a web application that is stored on a remote server and delivered over through Internet so that it will be helpful as a real word application.
- ✓ The transactions will become more secure to gain the trust of the customers.
- ✓ The data in the data base will be more secure and maintains integrity with other systems.