

# Delhi Public School Bokaro Steel City

A Project Report on

## MOVIE BOOKING SYSTEM

For

AISSCE 2019 Examination

[As a part of the Informatics Practices Course (065)]

SUBMITTED BY

*Amal Prakash*

[Roll No. 7678054]

Under the Guidance of:

*Mr. Sanjay Pandey*

## CERTIFICATE

This is to certify that the Project / Dissertation entitled Movie Booking System is a bonafide work done by Master Amal Prakash of class XII D Session 2018-19 in partial fulfillment of CBSE's AISSCE Examination 2019 and has been carried out under my direct supervision and guidance. This report or a similar report on the topic has not been submitted for any other examination and does not form a part of any other course undergone by the candidate.

Signature of Internal Teacher

Signature of External Teacher

## ACKNOWLEDGEMENT

I undertook this Project work, as the part of my XII-Informatics Practices course. I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing software system is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help and good wishes of the people near you, who have considerable experience and idea.

I would like to extend my sincere thanks and gratitude to my teacher [Mr Sanjay Pandey](#). I am very much thankful to our Principal [Dr Hemlata S.Mohan](#) for her words of inspiration and motivation and for providing facilities and infrastructure.

I would like to take the opportunity to extend my sincere thanks and gratitude to my parents for being a source of inspiration and providing time and freedom to develop this software project.

I also feel indebted to my friends for the valuable suggestions during the project work.

**Amal Prakash**  
**Class XII D**

# CONTENTS

<b>1. Introduction</b>	5
<b>2. Objective &amp; Scope of the Project</b>	6
<b>3. Theoretical Background</b>	7
<b>4. Problem Definition &amp; Analysis</b>	10
<b>5. System Implementation</b>	12
5.1 The Hardware used:	12
5.2 The Softwares used:	12
<b>6. System Design &amp; Development</b>	13
6.1 Database Design:	13
6.2 Event Coding:	14
<b>7. User Manual</b>	41
8.1 How to install:	41
8.2 Working with Software:	42
<b>8. References</b>	43

## INTRODUCTION

The main aim of this project Movie Booking System is a best way ticket booking for Cinema Halls. Users can book tickets according to the scheduled movies. On the other hand, admin can add, update or delete movies through this application. This program mainly brings forth the usage of GUI programming in the daily usage over the network. The program when made to work over the network can prove to be an ultimate way of interaction between the user and the store.

A MIS mainly consists of a computerized database, a collection of tables for a particular subject or purpose, capable to produce different reports relevant to the user. An application program is tied with the database for easy access and interface to the database. Using Application program or front-end, we can store, retrieve and manage all information in proper way.

This software, being simple in design and working, does not require much of training to users, and can be used as a powerful tool for the automating cinema halls.

During coding and design of the software Project, Java NetBeans IDE, a powerful front-end tool is used for getting Graphical User Interface (GUI) based integrated platform and coding simplicity. As a back-end a powerful, open source RDBMS, My SQL is used as per requirement of the CBSE curriculum of Informatics Practices Course.

## OBJECTIVE & SCOPE OF THE PROJECT

The objective of the software project is to develop a computerized MIS to automate the functions of Cinema Halls. This software project is also aimed to enhance the current record keeping system, which will help managers to retrieve the up-to-date information at right time in right shape.

The proposed software system is expected to do the following functionality-

- ✓ To provide a user friendly, Graphical User Interface (GUI) based integrated and centralized environment for MIS activities.
- ✓ The proposed system should maintain all the records and transactions, and should generate the required reports and information when required.
- ✓ To provide graphical and user-friendly interface to interact with a centralized database based on client-server architecture.
- ✓ To identify the critical operation procedure and possibilities of simplification using modern IT tools and practices.

In its current scope, the software enables user to retrieve and update the information from centralized database designed with MySQL . This software does not require much training time of the users due to limited functionality and simplicity.

During the development of Movie Booking System project, Java NetBeans IDE, a powerful, open source event-driven form-based development environment is used for modular design and future expandability of the system.

Despite of the best effort of the developer, the following limitations and functional boundaries are visible, which limits the scope of this application software.

1. This software can store records and produce reports in pre-designed format in soft copy. There is no facility yet to produce customized reports. Only specified reports are covered.
2. There is no provision to calculate fine or penalty etc. for defaulter members; however it can be developed easily with the help of adding modules.
3. Some application areas like cancellation of ticket, booking confirmation mail etc. are not implemented in the project. It facilitates the buyer to go through the list of movies sheduled and choose as per his wish to book the ticket. This project is made by keeping in mind that it is to be used over the network, which can facilitate ease of booking.

So far as future scope of the project is concerned, firstly it is open to any modular expansion i.e. other modules or functions can be designed and embedded to handle the user need in future. Any part of the software and reports can be modified independently without much effort.

# THEORETICAL BACKGROUND

## *What is Database?*

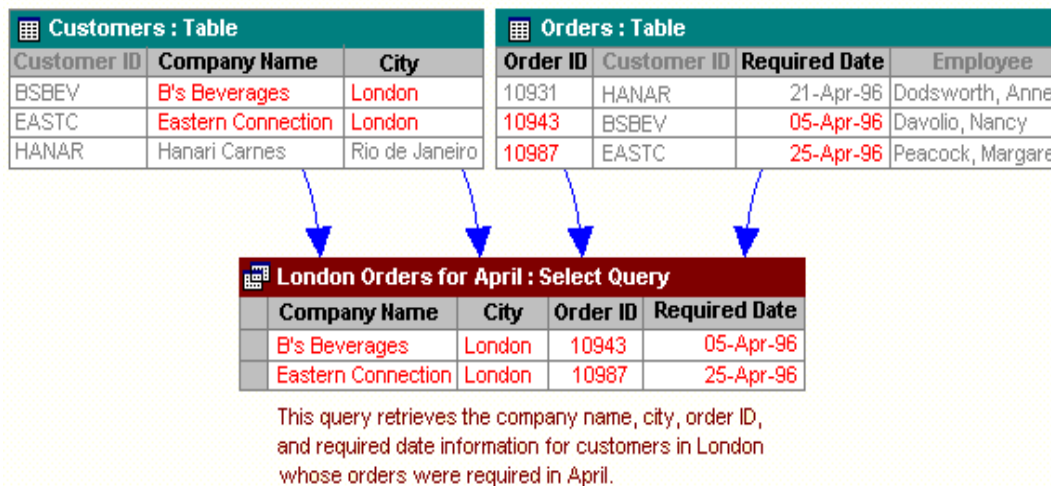
### **Introduction and Concepts:**

A database is a collection of information related to a particular subject or purpose, such as tracking customer orders or maintaining a product collection. Using any RDBMS application software like MS SQL Server, MySQL, Oracle, Sybase etc, you can manage all your information from a single database file. Within the file, divide your data into separate storage containers called tables. You may and retrieve the data using queries.

A table is a collection of data about a specific topic, such as products or suppliers. Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organises data into columns (called fields) and rows (called records).

A Primary key is one or more fields whose value or values uniquely identify each record in a table. In a relationship, a primary key is used to refer to specific record in one table from another table. A primary key is called foreign key when it is referred to from another table.

To find and retrieve just the data that meets conditions you specify, including data from multiple tables, create a query. A query can also update or delete multiple records at the same time, and perform built-in or custom calculations on your data.



### **Role of RDBMS Application Program:**

A computer database works as a electronic filing system, which has a large number of ways of cross-referencing, and this allows the user many different ways in which to re-organize and retrieve data. A database can handle business inventory, accounting and filing and use the information in its files to prepare summaries, estimates and other reports. The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available DBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase. A database management system, therefore, is a combination of hardware and software that can be used to set up and monitor a database, and can manage the updating and retrieval of database that has been stored in it. Most of the database management systems have the following capabilities:

- ◆ Creating of a table, addition, deletion, modification of records.
- ◆ Retrieving data collectively or selectively.
- ◆ The data stored can be sorted or indexed at the user's discretion and direction.
- ◆ Various reports can be produced from the system. These may be either standardized report or that may be specifically generated according to specific user definition.
- ◆ Mathematical functions can be performed and the data stored in the database can be manipulated with these functions to perform the desired calculations.
- ◆ To maintain data integrity and database use.

The DBMS interprets and processes users' requests to retrieve information from a database. In most cases, a query request will have to penetrate several layers of software in the DBMS and operating system before the physical database can be accessed. The DBMS responds to a query by invoking the appropriate subprograms, each of which performs its special function to interpret the query, or to locate the desired data in the database and present it in the desired order.

### *What is My SQL ?*



The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available RDBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase.

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. MySQL is named after co-founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (our logo) is "Sakila,".

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL is based on SQL.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL-92" refers to the standard released in 1992, "SQL:1999" refers to the standard released in 1999, and "SQL:2003" refers to the current version of the standard.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License),



- **The MySQL Database Server is very fast, reliable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server also has a practical set of features developed in close cooperation with our users. You can find a performance comparison of MySQL Server with other database managers on our benchmark page. MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

## The Main Features of MySQL

- Written in C and C++.
- Works on many different platforms.
- Uses multi-layered server design with independent modules.
- Provides transactional and nontransactional storage engines.
- Designed to make it relatively easy to add other storage engines. This is useful if you want to provide an SQL interface for an in-house database.
- Uses a very fast thread-based memory allocation system.
- Executes very fast joins using an optimized nested-loop join.
- Implements SQL functions using a highly optimized class library that should be as fast as possible. Usually there is no memory allocation at all after query initialization.
- Provides the server as a separate program for use in a client/server networked environment, and as a library that can be embedded (linked) into standalone applications. Such applications can be used in isolation or in environments where no network is available.
- Password security by encryption of all password traffic when you connect to a server.
- Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5,000,000,000 rows.
- MySQL client programs can be written in many languages. A client library written in C is available for clients written in C or C++, or for any language that provides C bindings.
- APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl are available, enabling MySQL clients to be written in many languages.
- The Connector/ODBC (MyODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections.
- The Connector/J interface provides MySQL support for Java client programs that use JDBC connections. Clients can be run on Windows or Unix. Connector/J source is available.

## What is NetBeans IDE ?

NetBeans started as a student project (originally called Xelfi) in the Czech Republic in 1996. The goal was to write a Delphi-like Java IDE in Java. Xelfi was the first Java IDE (Integrated Development Environment) written in Java, with its first pre-releases in 1997. Xelfi was a fun project to work on, especially since Java IDE space was uncharted territory at that time. The project attracted enough interest that these students, once they graduated, decided that they could market it as a commercial product. Soliciting resources from friends and relatives for a web space, they formed a company around it.

Soon after, they were contacted by Roman Stanek, an entrepreneur who had already been involved in several startups in the Czech Republic. He was looking for a good idea to invest in, and discovered Xelfi. He met with the founders; they hit it off, and a business was born.

In the spring of 1999, NetBeans DeveloperX2 was released, supporting Swing. The performance improvements that came in JDK 1.3, released in the fall of 1999, made NetBeans a viable choice for development tools. By the summer of 1999, the team was hard at work re-architecting DeveloperX2 into the more modular NetBeans that forms the basis of the software today.

Something else was afoot in the summer of 1999: Sun Microsystems wanted better Java development tools, and had become interested in NetBeans. It was a dream come true for the NetBeans team: NetBeans would become the flagship tool set of the maker of Java itself! By the Fall, with the next generation of NetBeans Developer in beta, a deal was struck. Sun Microsystems had also acquired another tools company, During the acquisition, the young developers who had been involved in open-source projects for most of their programming careers, mentioned the idea of open-sourcing NetBeans. Fast forward to less than six months later, the decision was made that NetBeans would be open sourced. While Sun had contributed considerable amounts of code to open source projects over the years, this was Sun's first *sponsored* open source project, one in which Sun would be paying for the site and handling the infrastructure.

## Features of NetBeans

A free, open-source Integrated Development Environment for software developers. You get all the tools you need to create professional desktop, enterprise, web, and mobile applications with the Java platform, as well as C/C++, PHP, JavaScript, Groovy, and Ruby.

NetBeans IDE 6.9 introduces the JavaFX Composer, support for JavaFX SDK 1.3, OSGi interoperability, support for the PHP Zend framework and Ruby on Rails 3.0, and more.

## PROBLEM DEFINITION & ANALYSIS

The hardest part of building a software system is deciding precisely what to build. No other part of the conceptual work is so difficult as establishing the detailed technical requirement. Defining and applying good, complete requirements are hard to work, and success in this endeavor has eluded many of us. Yet, we continue to make progress.

Problem definition describes the *What* of a system, not *How*. The quality of a software product is only as good as the process that creates it. Problem definition is one of the most crucial steps in this creation process. Without defining a problem, developers do not know what to build, customers do not know what to expect, and there is no way to validate that the built system satisfies the requirement.

Problem definition and Analysis is the activity that encompasses learning about the problem to be solved, understanding the needs of customer and users, trying to find out who the user really is, and understanding all the constraints on the solution. It includes all activities related to the following:

- ✓ Identification and documentation of customer's or user's needs.
- ✓ Creation of a document that describes the external behavior and the association constraints that will satisfies those needs.
- ✓ Analysis and validation of the requirements documents to ensure consistency, completeness, and feasibility
- ✓ Evolution of needs.

After the analysis of the functioning of a Movie Booking System, the proposed System is expected to do the following: -

- ✓ To provide a user friendly, Graphical User Interface (GUI) based integrated and centralized environment for booking tickets, showing movie list.
- ✓ The proposed system should maintain all the records and transactions, and should generate the required reports and information when required.
- ✓ To provide efficient and secured Information storage, flow and retrieval system, ensuring the integrity and validity of records.
- ✓ To provide graphical and user-friendly interface to interact with a centralized database based on client-server architecture.
- ✓ To identify the critical operation procedure and possibilities of simplification using modern IT tools and practices.

# SYSTEM IMPLEMENTATION

## *The Hardware used:*

While developing the system, the used hardware are:

PC with Pentium Dual Core processor having 2.00 GB RAM, SVGA and other required devices.

## *The Softwares used:*

- Microsoft Windows® 7 as Operating System.
- Java NetBeans 7.2 as Front-end Development environment.
- MySQL as Back-end Sever with Database for Testing.
- MS-Word 2007 for documentation.

# SYSTEM DESIGN & DEVELOPMENT

## **Database Design:**

An important aspect of system design is the design of data storage structure. To begin with a logical model of data structure is developed first. A database is a container object which contains tables, queries, reports and data validation policies enforcement rules or constraints etc. A logical data often represented as a records are kept in different tables after reducing anomalies and redundancies. The goodness of data base design lies in the table structure and its relationship. This software project maintains a database named **Movie** which contains the following tables.

## **Table Design:**

The database of Library System contains 2 tables. The tables are normalized to minimize the redundancies of data and enforcing the validation rules of the organization. Most of the tables are designed to store master records. The tables and their structure are given below.

**Table: movie**

<i>Column Name</i>	<i>Type</i>	<i>Size</i>
<b>Movie_name (Primary key)</b>	Varchar	50
Language	Varchar	20
Time	Varchar	10

**Table: booking**

<i>Column Name</i>	<i>Type</i>	<i>Size</i>
Movie_name	Integer	10
Class	Varchar	20
Seats	Integer	11
Date	Date	
Time	Varchar	10
Snacks	Varchar	40
Price	Decimal	10

```
MySQL 5.5 Command Line Client
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use movie;
Database changed
mysql> desc movie;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| movie_name | varchar(50)   | NO   | PRI |          |       |
| language   | varchar(20)   | NO   |     |          |       |
| time       | varchar(10)   | NO   |     |          |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.11 sec)

mysql> desc booking;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| movie_name | varchar(50)   | NO   |     | NULL    |       |
| class      | varchar(20)   | NO   |     | NULL    |       |
| SEATS      | int(11)       | NO   |     | NULL    |       |
| date       | date          | NO   |     | NULL    |       |
| time       | varchar(10)   | NO   |     | NULL    |       |
| snacks     | varchar(20)   | YES  |     | NULL    |       |
| PRICE      | decimal(10,0) | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
7 rows in set (0.09 sec)
```

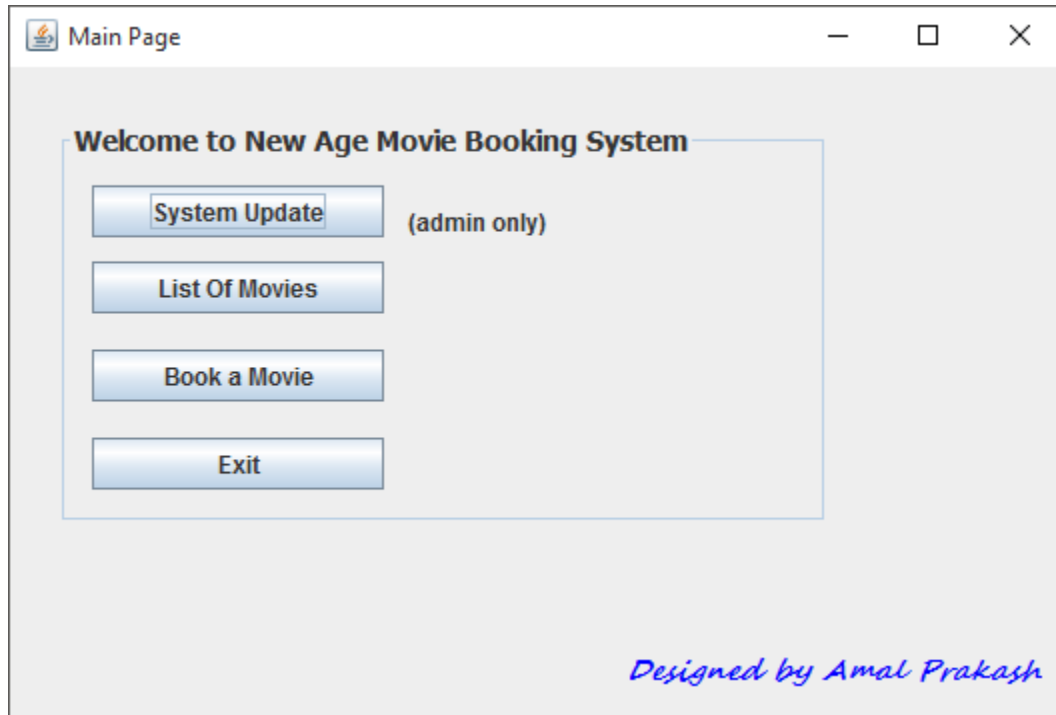
## Event Coding:

The software project for Movie Booking System contains various forms along with programming codes. Forms (JFrames) and their event coding are given below.

---

### Frame: MovieBookingSystem.java

---



### Coding for MovieBookingSystem.java

```
package MovieBookingSystem;

public class MovieBookingSystem extends javax.swing.JFrame {

    public MovieBookingSystem() {
        initComponents();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        Admin s =new Admin();
        s.setVisible(true);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

        ListofMovie s =new ListofMovie();
        s.setVisible(true);

    }
```

```

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    BookMovie bm =new BookMovie();
    bm.setVisible(true);
}
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

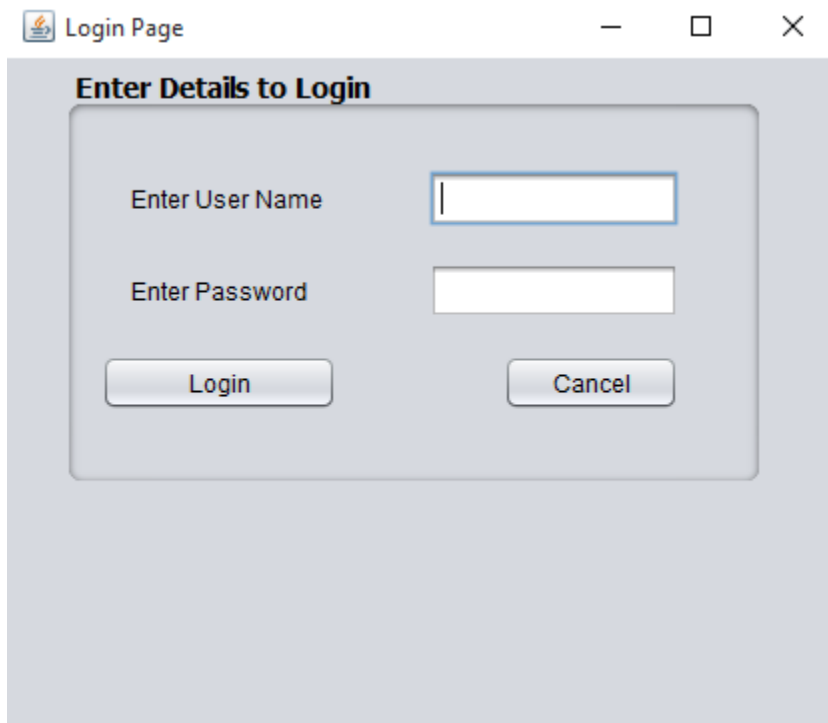
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        @Override
        public void run() {
            new MovieBookingSystem().setVisible(true);
        }
    });
}

```

---

**Frame: admin.java**

---



### **Coding for admin.java**

```

package MovieBookingSystem;

import javax.swing.JOptionPane;
public class Admin extends javax.swing.JFrame {    /**
    /**

```

```

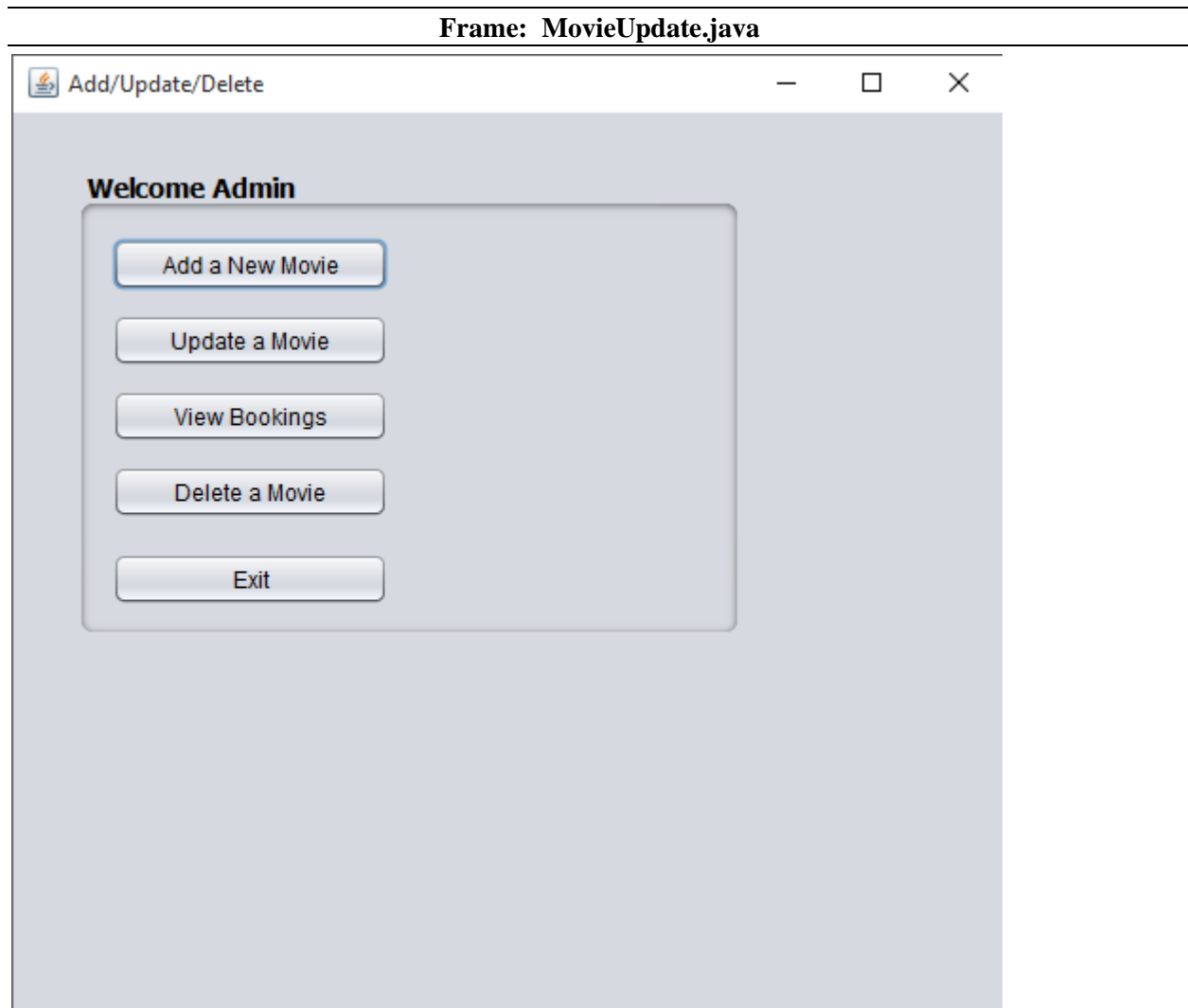
    * Creates new form Admin
    */
public Admin() {
    initComponents();
}
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    String username=user.getText();
    String pwd= new String(pass.getPassword());
    if(username.equals("amal") && pwd.equals("amal"))
    {
        // WindowEvent w= new WindowEvent(this,WindowEvent.WINDOW_CLOSING);
        // Toolkit.getDefaultToolkit().getSystemEventQueue().postEvent(w);
        MovieUpdate s =new MovieUpdate();
        s.setVisible(true);
        this.setVisible(false);
    }
    else
    {
        JOptionPane.showMessageDialog(null," Incorrect User ID or password");
        user.setText("");
        pass.setText("");
    }
}
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}

public static void main(String args[]) {
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new Admin().setVisible(true);
        }
    });
}

```





### Coding for MovieUpdate.Java

```
package MovieBookingSystem;
```

```
public class MovieUpdate extends javax.swing.JFrame {
```

```
    /**
```

```
     * Creates new form NewJFrame
```

```
    */
```

```
    public MovieUpdate() {
```

```
        initComponents();
```

```
    }
```

```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
```

```
        MovieAddition ma =new MovieAddition();
```

```
        ma.setVisible(true);
```

```
    }
```

```

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    MovieUpdation mu =new MovieUpdation();
    mu.setVisible(true);
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    MovieDeletion md =new MovieDeletion();
    md.setVisible(true);
}

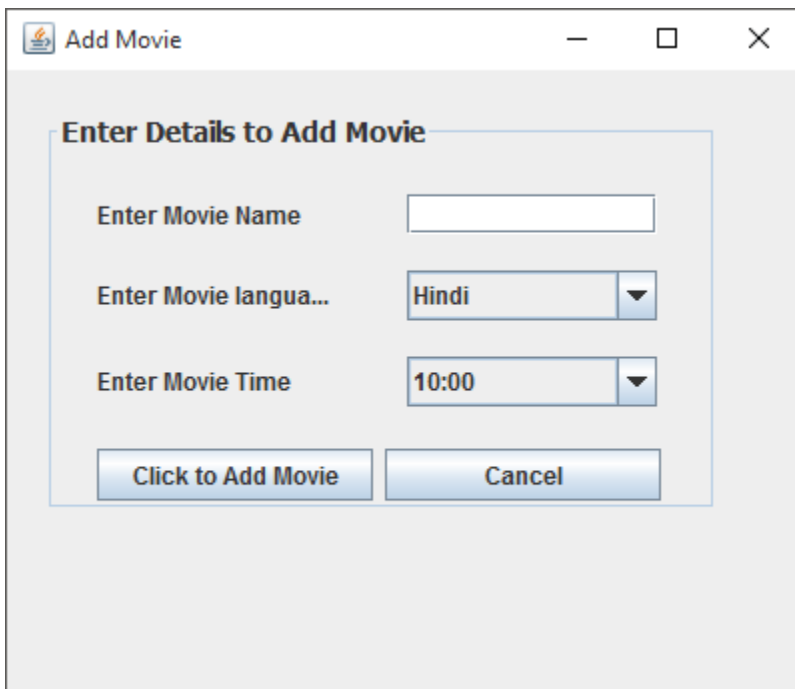
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    bookings booking=new bookings();
    booking.setVisible(true);
}

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new MovieUpdate().setVisible(true);
        }
    });
}

```

### Frame: MovieAddition.java



### Code For MovieAddition.java

```
package MovieBookingSystem;

import java.sql.*;
import javax.swing.JOptionPane;

public class MovieAddition extends JFrame {

    public MovieAddition() {
        initComponents();
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        if(moviename.getText().length()==0 )
            JOptionPane.showMessageDialog(null," Please enter movie name first.");
        else
        {
            try {
                Class.forName("com.mysql.jdbc.Driver")
                Connection
                con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

                String sql="insert into movie values(?, ?, ?)";
                PreparedStatement pst=con.prepareStatement(sql);
                pst.setString(1,moviename.getText());
                String lang=null;
```

```

        if(movieLang.getSelectedIndex()==0)
            lang="Hindi";
        if(movieLang.getSelectedIndex()==1)
            lang="English";
        if(movieLang.getSelectedIndex()==2)
            lang="Tamil";
        if(movieLang.getSelectedIndex()==3)
            lang="Malyalam";
        pst.setString(2, lang);
        String time=null;
        if(movieTime.getSelectedIndex()==0)
            time="10:00";
        if(movieTime.getSelectedIndex()==1)
            time="12:00";
        if(movieTime.getSelectedIndex()==2)
            time="14:00";
        if(movieTime.getSelectedIndex()==3)
            time="16:00";
        if(movieTime.getSelectedIndex()==4)
            time="18:00";
        if(movieTime.getSelectedIndex()==5)
            time="20:00";
        if(movieTime.getSelectedIndex()==6)
            time="22:00";
        if(movieTime.getSelectedIndex()==7)
            time="23:00";
        pst.setString(3, time);

        pst.executeUpdate();
        JOptionPane.showMessageDialog(null," Movie has been added successfully");
        con.close();

    } catch (Exception e) {
        JOptionPane.showMessageDialog(null,"error");
    }

    this.setVisible(false); //this will close frame i.e. NewJFrame

new MovieAddition().setVisible(true);
}
}

```

```

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        @Override
        public void run() {
            new MovieAddition().setVisible(true);
        }
    });
}
}

```

**Frame: M ovieUpdation.java**

### **Coding for MovieUpdation.java**

```

package MovieBookingSystem;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;
public class MovieUpdation extends javax.swing.JFrame {

```

```

    public MovieUpdation() {
        initComponents();
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Class.forName("com.mysql.jdbc.Driver");
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

            String sql="update movie set language=? , time=? where movie_name=?";
            PreparedStatement pst=con.prepareStatement(sql);
            String lang=null;
            if(movieLang.getSelectedIndex()==0)
                lang="Hindi";
            if(movieLang.getSelectedIndex()==1)
                lang="English";
            if(movieLang.getSelectedIndex()==2)
                lang="Tamil";
            if(movieLang.getSelectedIndex()==3)
                lang="Malyalam";
            pst.setString(1, lang);
            String time=null;
            if(movieTime.getSelectedIndex()==0)
                time="10:00";
            if(movieTime.getSelectedIndex()==1)
                time="12:00";
            if(movieTime.getSelectedIndex()==2)
                time="14:00";
            if(movieTime.getSelectedIndex()==3)
                time="16:00";
            if(movieTime.getSelectedIndex()==4)
                time="18:00";
            if(movieTime.getSelectedIndex()==5)
                time="20:00";
            if(movieTime.getSelectedIndex()==6)
                time="22:00";
            if(movieTime.getSelectedIndex()==7)
                time="23:00";
            pst.setString(2, time);
            String name=(String) movieName.getSelectedItem();
            pst.setString(3,name);

```

```

        pst.executeUpdate();
        JOptionPane.showMessageDialog(null," Movie has been updated successfully");
        con.close();
    }
    catch(Exception e){
        JOptionPane.showMessageDialog(null,"error");
    }

    this.setVisible(false); //this will close frame i.e. NewJFrame

    new MovieUpdation().setVisible(true);
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        this.dispose();
    }
    private void formWindowOpened(java.awt.event.WindowEvent evt) {

        try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
        con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");
        Statement st=con.createStatement();
        String selectQuery="select movie_name from movie";
        ResultSet rs=st.executeQuery(selectQuery);
        while(rs.next())
        {
            movieName.addItem(rs.getString("movie_name"));
        }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null,"error");
        }
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        ListofMovie l=new ListofMovie();
        l.setVisible(true);
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            @Override
            public void run() {

```

```

        new MovieUpdation().setVisible(true);
    }
});
}

```

**Frame: bookings.java**

movie_name	class	SEATS	date	time	snacks	PRICE
Avengers-Infinity War	Gold	1	2019-01-27	16:00		550
Avengers-Infinity War	Gold	1	2019-01-25	16:00		550
Tik Tik Tik	Gold	2	2019-01-25	20:00		1100
KGF	Premium	1	2019-01-25	14:00	Popcorns	800
Total Dhamaal	Gold	1	2019-01-29	16:00	Cold Drinks	600
Kedarnath	Gold	1	2019-01-26	14:00	Popcorns	700
Kedarnath	Platinum	4	2019-02-02	14:00		3000
Robot 2.0	Premium	6	2019-01-25	12:00	Popcorns	4050
Uri-The Surgical Strike	Gold	6	2019-01-25	10:00		3300
KGF	Premium	2	2019-01-28	14:00	Popcorns	1450
Avengers-Infinity War	Premium	3	2019-01-25	16:00	Popcorns	2100

Select Movie:

Select Date:

### Code for booking.java

```

package MovieBookingSystem;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.swing.JOptionPane;
import net.proteanit.sql.DbUtils;

public class bookings extends javax.swing.JFrame {

```



```

public void DisplayBooking()
{
    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

        String sql="select * from booking";
        PreparedStatement pst=con.prepareStatement(sql);
        ResultSet rs=pst.executeQuery();
        booking.setModel(DbUtils.resultSetToTableModel(rs));
    }
    catch (Exception e )
    {
        JOptionPane.showMessageDialog(null,"error");
    }
}

public bookings() {
    initComponents();
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
    Date date = new Date();
    MovieDate.setDate(date);
    MovieDate.setMinSelectableDate(new Date());
    SimpleDateFormat sdf = new SimpleDateFormat("E,dd-MM-yyyy");
    String md = sdf.format( MovieDate.getDate());
    DisplayBooking();

    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");
        Statement st=con.createStatement();
        String selectQuery="select movie_name from movie";
        ResultSet rs=st.executeQuery(selectQuery);
        while(rs.next())
        {
            movieName.addItem(rs.getString("movie_name"));
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null,"error");
    }
}

```

```

    }

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
dispose();
}

private void seatsbookedActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

        String sql="select sum(seats) as total_booking from booking where movie_name=? and
date=?";
        PreparedStatement psmt=con.prepareStatement(sql);
        String name=movieName.getSelectedItem().toString();
        psmt.setString(1,name);
        SimpleDateFormat d = new SimpleDateFormat("yyyy-MM-dd");
        String dt = d.format( MovieDate.getDate());
        psmt.setString(2,dt);
        ResultSet rs=psmt.executeQuery();
        if(rs.next())
        {
            seat.setText(rs.getString("total_booking"));
        }
        if(rs.getString("total_booking")==null)
        {
            JOptionPane.showMessageDialog(null,"No Bookings");
        }
        con.close();

    }
catch (Exception e) {
    JOptionPane.showMessageDialog(null,"error");
}

}
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {

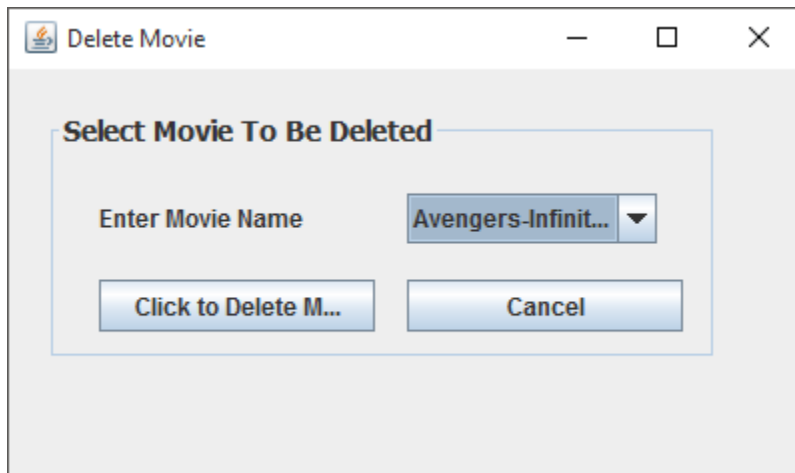
```

```

java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new bookings().setVisible(true);
    }
});
}

```

**Frame: MovieDeletion.java**



### **Coding of MovieDeletion.java**

```

package MovieBookingSystem;

```

```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;

```

```

public class MovieDeletion extends javax.swing.JFrame {

```

```

    public MovieDeletion() {
        initComponents();
    }

```

```

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {

```

```

            Class.forName("com.mysql.jdbc.Driver");
            Connection
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

```

```

String sql="delete from movie where movie_name=?";
PreparedStatement pst=con.prepareStatement(sql);
String name=(String) movieName.getSelectedItemAt();
pst.setString(1,name);
pst.executeUpdate();
JOptionPane.showMessageDialog(null," Movie has been deleted successfully");
con.close();
}

catch(Exception e)
{
JOptionPane.showMessageDialog(null,"error");
}
this.setVisible(false); //this will close frame i.e. NewJFrame

new MovieDeletion().setVisible(true);
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
this.dispose();
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {
try {
Class.forName("com.mysql.jdbc.Driver");
Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");
Statement st=con.createStatement();
String selectQuery="select movie_name from movie";
ResultSet rs=st.executeQuery(selectQuery);
while(rs.next())
{
movieName.addItem(rs.getString("movie_name"));
}
} catch (Exception e) {
JOptionPane.showMessageDialog(null,"error");
}
}

public static void main(String args[]) {
java.awt.EventQueue.invokeLater(new Runnable() {
@Override
public void run() {
new MovieDeletion().setVisible(true);
}
}
}

```

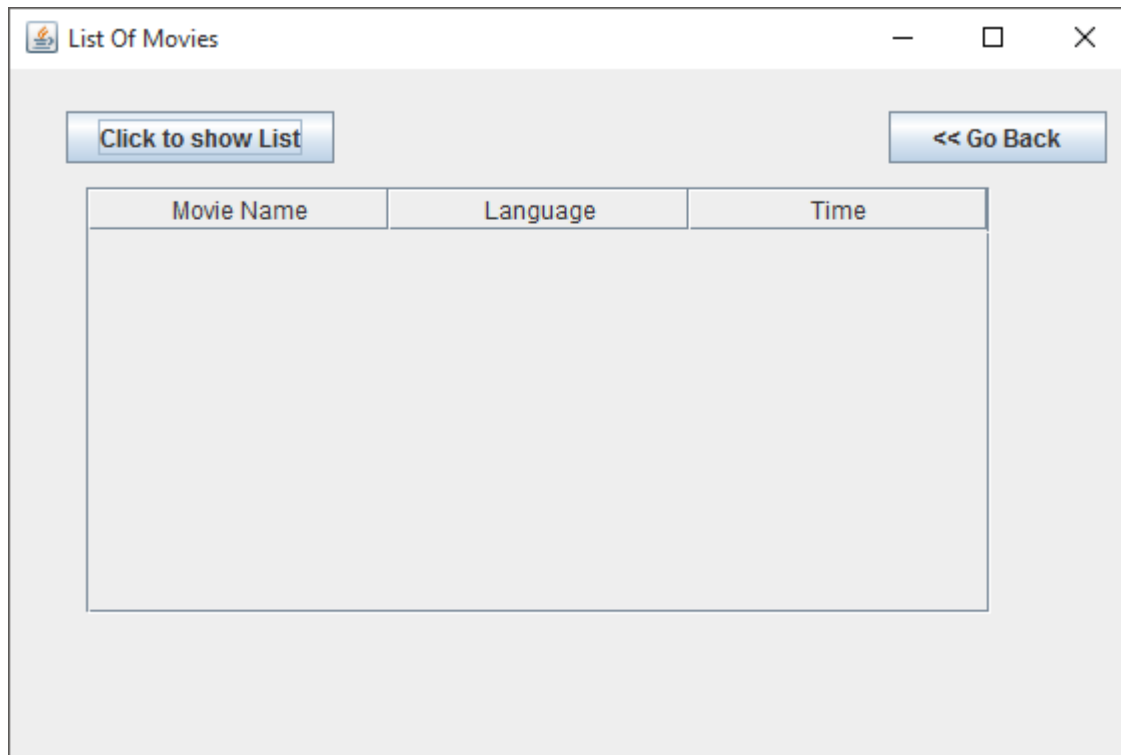
```

    }
    });

}

```

**Frame: ListofMovies.java**



### Coding for ListofMovies.java

```

package MovieBookingSystem;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import net.proteanit.sql.DbUtils;
import javax.swing.JOptionPane;

public class ListofMovie extends javax.swing.JFrame {

    public void DisplayTable()
    {

        try {
            Class.forName("com.mysql.jdbc.Driver");

```

```

        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

        String sql="select * from movie";
        PreparedStatement pst=con.prepareStatement(sql);
        ResultSet rs=pst.executeQuery();
        movie.setModel(DbUtils.resultSetToTableModel(rs));
    }
    catch (Exception e )
    {
        JOptionPane.showMessageDialog(null,"error");
    }

}

public ListofMovie() {
    initComponents();

}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    DisplayTable();
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}

public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new ListofMovie().setVisible(true);
            //ListofMovie o=new ListofMovie();
            //o.DispList();
        }
    }); }

```

### Frame: BookMovie.java

### Coding for BookMovie.java

```
package MovieBookingSystem;
import java.text.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.Statement;
import java.sql.ResultSet;
import java.util.Date;
import javax.swing.JOptionPane;
public class BookMovie extends javax.swing.JFrame {
    String mc,mn,ns,md;
    public BookMovie() {
        initComponents();
    }
    public void fillcombo(){
```

```

try {
    Class.forName("com.mysql.jdbc.Driver");
    Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");
    Statement st=con.createStatement();
    String selectQuery="select movie_name from movie";
    ResultSet rs=st.executeQuery(selectQuery);
    while(rs.next())
    {
        movieName.addItem(rs.getString("movie_name"));
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(null,"error");
}
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    ns=SeatNo.getSelectedItemAt().toString();
    SimpleDateFormat sdf = new SimpleDateFormat("E,dd-MM-yyyy");
    md = sdf.format( MovieDate.getDate());
    double total=0;
    int snacks=0;
    if (pop.isSelected())
        snacks=snacks+150;
    if (drink.isSelected())
        snacks=snacks+50;
    total=snacks+ GetCost()*Integer.parseInt((String) SeatNo.getSelectedItemAt());
    int s=0;
    try {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

        String sql="select sum(SEATS) as total_booking from booking where movie_name=?
and date=?";
        PreparedStatement psmt=con.prepareStatement(sql);
        String name=movieName.getSelectedItemAt().toString();
        psmt.setString(1,name);
        SimpleDateFormat d = new SimpleDateFormat("yyyy-MM-dd");
        String dt = d.format( MovieDate.getDate());
        psmt.setString(2,dt);
        ResultSet rs=psmt.executeQuery();

```



```

        if(rs.next())
        {

            if(rs.getString("total_booking")==null)
            {
                s=0;
            }
            else
            {
                s=Integer.parseInt(rs.getString("total_booking"));
            }
        }
        con.close();

    }
    catch (Exception e) {
        JOptionPane.showMessageDialog(null,"error1");
    }

int nos=Integer.parseInt(ns);
if ((s+nos)<=5)
{
    try
    {
        Class.forName("com.mysql.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

        String sql="insert into booking values(?, ?, ?, ?, ?, ?, ?)";
        PreparedStatement pst=con.prepareStatement(sql);
        mn=movieName.getSelectedItemAt().toString();
        pst.setString(1,mn);
        if(gold.isSelected())
            mc="Gold";
        if(premium.isSelected())
            mc="Premium";
        if(platinum.isSelected())
            mc="Platinum";
        pst.setString(2,mc);

        pst.setInt(3,Integer.parseInt(ns));
        SimpleDateFormat d = new SimpleDateFormat("yyyy-MM-dd");

```

```

String date = d.format( MovieDate.getDate());
pst.setString(4,date);
pst.setString(5,time.getText());
String snk="";
if(pop.isSelected())
    snk+=pop.getText()+" ";
if(drink.isSelected())
    snk+=drink.getText()+" ";
pst.setString(6,snk);
pst.setDouble(7,total);

pst.executeUpdate();

con.close();
JOptionPane.showMessageDialog(null,"Your Movie "+mn +" with "+mc+" class "+"with
"+ns+" seats"
    + " has been booked successfully.\nYour total cost is ="
    +GetCost()*Integer.parseInt((String) SeatNo.getSelectedItemAt())+"\nYour movie date is
="+md+
    "\nMovie will start at "+time.getText());
}
catch(Exception e)
{
JOptionPane.showMessageDialog(null,"error");
}
}
else
{
JOptionPane.showMessageDialog(null,"Sorry! Show is full");
}
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    this.dispose();
}

private void movieNameActionPerformed(java.awt.event.ActionEvent evt) {
try {
    Class.forName("com.mysql.jdbc.Driver");
    Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

```

```

String sql="select time from movie where movie_name=?";
PreparedStatement pst=con.prepareStatement(sql);
String name=(String)movieName.getSelectedItemAt();
pst.setString(1,name);
ResultSet rs=pst.executeQuery();
if(rs.next())
{
time.setText(rs.getString("time"));
}

con.close();
}
catch (Exception e) {
JOptionPane.showMessageDialog(null,"error");
}
}

private void formWindowOpened(java.awt.event.WindowEvent evt) {

gold.setSelected(true);
fillcombo();
Date date = new Date();
MovieDate.setDate(date);
MovieDate.setMinSelectableDate(new Date());
setLabelText();
jButton1.setEnabled(false);
}

private void SeatNoActionPerformed(java.awt.event.ActionEvent evt) {
setLabelText();
}

private void jCheckBox3ActionPerformed(java.awt.event.ActionEvent evt) {
jButton1.setEnabled(true);
}

private void premiumActionPerformed(java.awt.event.ActionEvent evt) {
setLabelText();
}

private void goldActionPerformed(java.awt.event.ActionEvent evt) {
setLabelText();
}

```

```

private void platinumActionPerformed(java.awt.event.ActionEvent evt) {
    setLableText();
}

private void popActionPerformed(java.awt.event.ActionEvent evt) {
    setLableText();
}

private void drinkActionPerformed(java.awt.event.ActionEvent evt) {
    setLableText();
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    terms terms=new terms();
    terms.setVisible(true);
}

private void availActionPerformed(java.awt.event.ActionEvent evt) {
    seatsavailability sa=new seatsavailability();
    sa.setVisible(true);
}

public void setLableText()
{ int snacks=0, total=0;
if (pop.isSelected())
    snacks=snacks+150;
if (drink.isSelected())
    snacks=snacks+50;
total=snacks+ GetCost()*Integer.parseInt((String) SeatNo.getSelectedItem());
    movieCost.setText("Total cost for your  "+(String) SeatNo.getSelectedItem()+

        "  seats will will be = "+total);

}

public int GetCost()
{
    if(gold.isSelected())
        return 550;
    else if(premium.isSelected())
        return 650;
    else //if(movieClass.getSelectedIndex()==2)
        return 750;
}

```

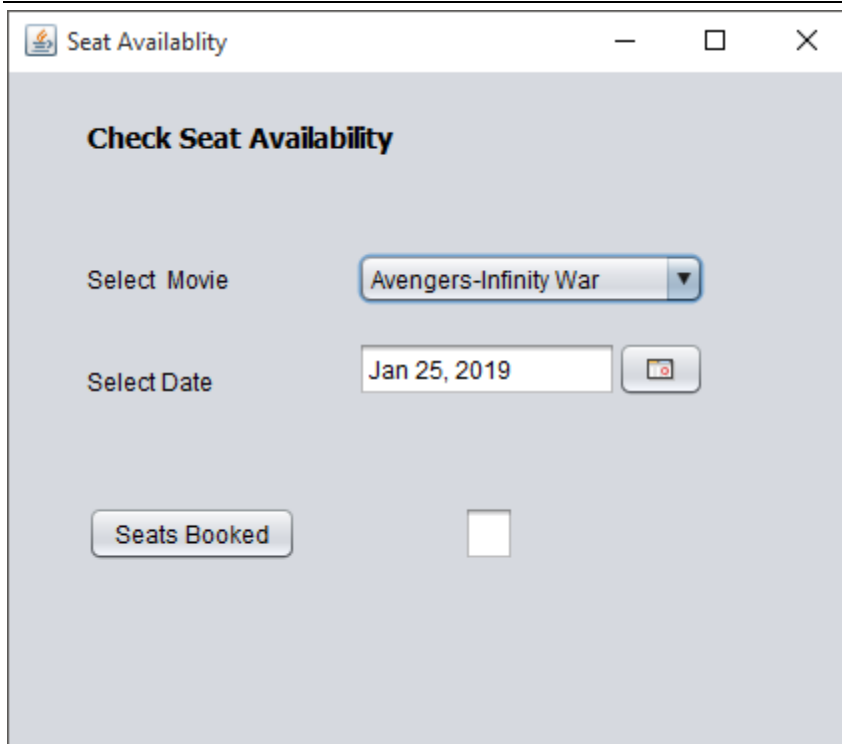
```

}

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        @Override
        public void run() {
            new BookMovie().setVisible(true);
        }
    });
}
}

```

**Frame: seatsavailability.java**



#### **Code for seatsavailability.java**

```

package MovieBookingSystem;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Date;

```

```

import javax.swing.JOptionPane;
public class seatsavailability extends javax.swing.JFrame {

    public seatsavailability() {
        initComponents();
    }
    private void seatsbookedActionPerformed(java.awt.event.ActionEvent evt) {

        try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");

String sql="select sum(seats) as total_booking from booking where movie_name=? and date=?";
            PreparedStatement psmt=con.prepareStatement(sql);
            String name=movieName.getSelectedItem().toString();
            psmt.setString(1,name);
            SimpleDateFormat d = new SimpleDateFormat("yyyy-MM-dd");
            String dt = d.format( MovieDate.getDate());
            psmt.setString(2,dt);
            ResultSet rs=psmt.executeQuery();
            if(rs.next())
            {
                seat.setText(rs.getString("total_booking"));
            }
            if(rs.getString("total_booking")==null)
            {
                JOptionPane.showMessageDialog(null,"No Bookings");
            }
            con.close();
        }
        catch (Exception e) {
            JOptionPane.showMessageDialog(null,"error");
        }

        int a=Integer.parseInt(seat.getText());
        int b=50-a;
        jLabel4.setText("Seat(s) remaining = "+b);
    }

    private void formWindowOpened(java.awt.event.WindowEvent evt) {
        Date date = new Date();
        MovieDate.setDate(date);
    }
}

```

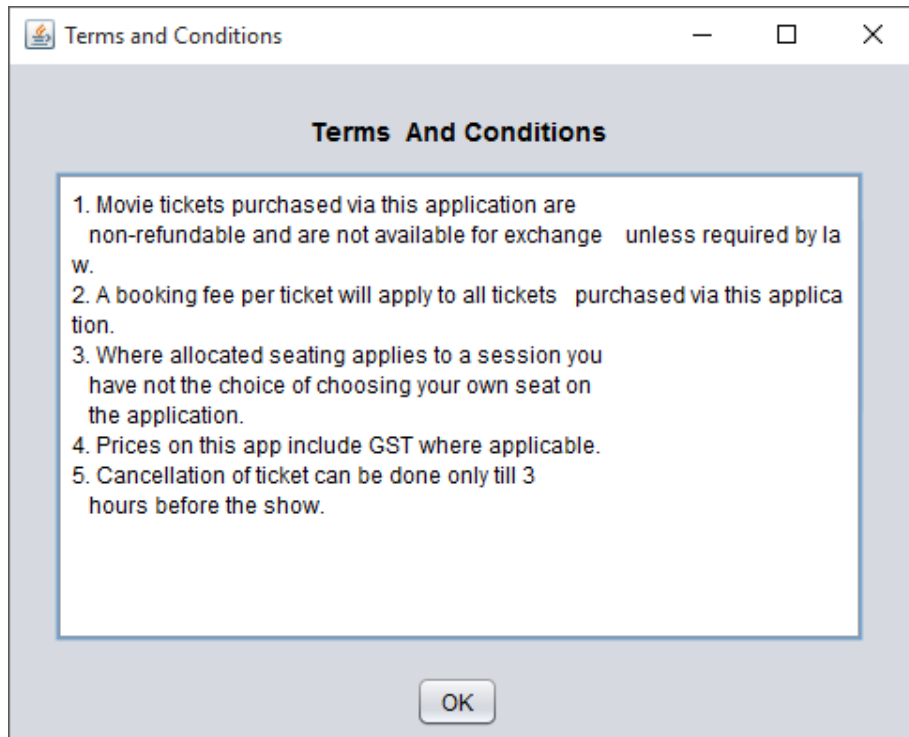
```

MovieDate.setMinSelectableDate(new Date());
SimpleDateFormat sdf = new SimpleDateFormat("E,dd-MM-yyyy");
String md = sdf.format( MovieDate.getDate());

try {
    Class.forName("com.mysql.jdbc.Driver");
    Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movie","root","dps");
    Statement st=con.createStatement();
    String selectQuery="select movie_name from movie";
    ResultSet rs=st.executeQuery(selectQuery);
    while(rs.next())
    {
        movieName.addItem(rs.getString("movie_name"));
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(null,"error");
}
}

public static void main(String args[]) {
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new seatsavailability().setVisible(true);
    }
})
}

```



### Code of terms.java

```
package MovieBookingSystem;

public class terms extends javax.swing.JFrame {
    public terms() {
        initComponents();
    }
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        BookMovie bm =new BookMovie();
        bm.setVisible(true);
        dispose();
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new terms().setVisible(true);
            }
        });
    }
}
```



# USER MANUAL

## How to install Software:

### Hardware Requirement-

- ◆ Intel Pentium/Celeron or similar processor based PC at Client/Server end.
- ◆ 128 MB RAM and 4GB HDD space (for Database) is desirable.
- ◆ Standard I/O devices like Keyboard and Mouse etc.
- ◆ Printer is needed for hard-copy reports.
- ◆ Local Area Network(LAN) is required for Client-Server Installation

### Software Requirement-

- ◆ Windows XP/2007 OS is desirable.
- ◆ NetBeans Ver 7.2 or higher should be installed with JDK and JVM.
- ◆ MySQL Ver 6.1 with Library Database must be present at machine.

### Database Installation

The software project is distributed with a backup copy of a Database named **movie** with required tables. Some dummy records are present in the tables for testing purposes, which can be deleted before inserting real data. The project is shipped with **movie.sql** file which installs a database and tables in the computer system.

Note: The PC must have MySQL server with user (**root**) and password (**dps**) . If root password is any other password, it can be changed by running MySQL Server Instance Configure Wizard.

Start ▶ Program ▶ MySQL ▶ MySQL Server ▶ MySQL Server Instance Config Wizard

Provide current password of root and new password as “dps” , this will change the root password.

To install a MySQL database from a dump file ( **movie.sql** ) , simply follow the following steps.

**Step 1:** Copy the Lib.sql file in **C:\Program files\MySql\MySql server 5.1\Bin** folder.

**Step 2:** Open MySQL and type the following command to create the database named Library.

```
mysql> create database movie;
```

**Step 3:** Open Command Window (Start ▶ Run ▶ cmd)

**Step 4:** Go to the following folder using CD command of DOS.

```
C:\Program files\MySql\MySql server 5.1\Bin>
```

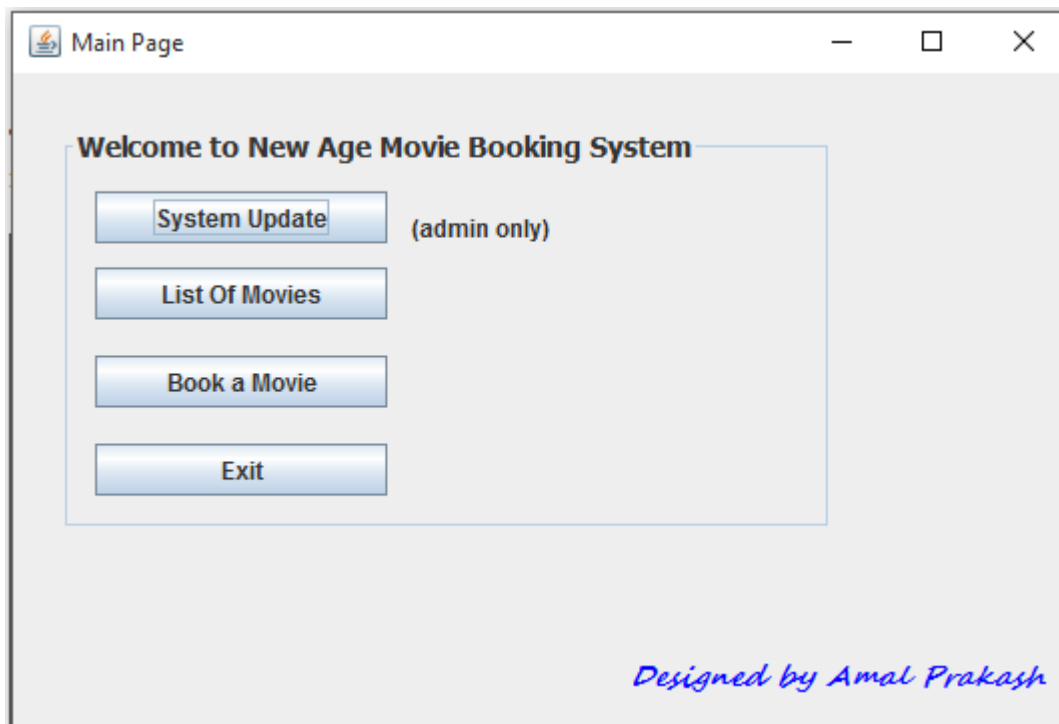
**Step 5:** type the following command on above prompt -

```
C:....\bin> mysql -u root -pdps movie < movie.sql
```

This will create a Library database with required tables.

## ***Working with SoftwareProject:***

The Movie Booking System consists of the following logically organised structure for the easy functionality. User may choose the options for corresponding works.



### **SYSTEM UPDATE :**

Through this the admin can sign in to the program with a valid username and password, so as to explore the program.

### **LIST OF MOVIES:**

Through this button, one can see the scheduled movies in the cinema hall as guest mode i.e., no username and password is required.

### **BOOK A MOVIE :**

Through this button, one can enter and book the show.

### **EXIT:**

Through this button, one can exit from the application.

## REFERENCES

In order to work on this project titled –*Movie Booking System*, the following books and literature are referred by me during the various phases of development of the project.

- (1) The Complete Reference Java 2.0  
-by Shildit
- (2) MySQL, Black Book  
-by Steven Holzner
- (2) Understanding SQL  
– Gruber
- (3) <http://www.mysql.org/>
- (4) <http://www.netbeans.org/>
- (5) On-line Help of NetBeans ®
- (6) Informatics Practices for class XII  
-by Sumita Arora
- (7) Together with Informatics Practices
- (6) Various Websites of Discussion Forum and software development activities.

Other than the above-mentioned books, the suggestions and supervision of my teacher and my class experience also helped me to develop this software project.