



**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE  
OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to J.N.T.U, Hyderabad)**

**Bachupally(v), Hyderabad, Telangana, India.**

**MOVIE TICKET BOOKING SYSTEM**

A course project submitted in complete requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY IN DATA SCIENCE**

Submitted by

**A.ROHITH REDDY      21071A6767**

**B.BHANU PRASAD      21071A6775**

**B.GURU KOUSHIK      21071A6779**

**G.AVINASH      21071A6788**

Under the guidance of

**Dr. N.Sunanda**

**Assistant Professor**

**Department of Computer Science & Engineering- (CyS, DS) and AI&DS**



**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE  
OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to J.N.T.U, Hyderabad)**

**Bachupally(v), Hyderabad, Telangana, India.**

**CERTIFICATE**

This is to certify that **A.ROHITH (21071A6767),B.BHANU PRASAD(21071A6775),B.GURU KOUSHIK(21071A6775) , G.AVINASH(21071A6788)** completed their course project work at Department of Computer Science & Engineering (CYS,DS,AI&DS) of VNR VJIET, Hyderabad entitled “**MOVIE TICKET BOOKING SYSTEM**” in complete fulfillment of the requirements for the award of B.Tech degree during the academic year 2022-2023. This work is carried out under my supervision and has not been submitted to any other University/Institute for award of any degree/diploma.

**Dr. N.Sunanda**

Assistant Professor

Department of CSE-(CyS, DS) and AI&DS

**VNRVJIET**

**Dr. M. Raja Sekar**

Professor and HOD

Department of CSE-(CyS, DS) and AI&DS

**VNRVJIET**



**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE  
OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to J.N.T.U, Hyderabad)**

**Bachupally(v), Hyderabad, Telangana, India.**

**DECLARATION**

This is to certify that our project titled “**MOVIE TICKET BOOKING SYSTEM**” submitted to Vallurupalli Nageswara Rao Institute of Engineering and Technology in fulfilling the requirement for the award of Bachelor of Technology in Computer Science and Engineering (Data Science) is a bonafide report to the work carried out by us under the guidance and supervision of Dr. N.Sunanda, Assistant Professor, Department of CSE- (CyS, DS) and AI&DS. To the best of our knowledge, this has not been submitted in any form to other universities or institutions for the award of any degree or diploma

<b>A.ROHITH REDDY</b>	<b>21071A6767</b>
<b>B.BHANU PRASAD</b>	<b>21071A6775</b>
<b>B.GURU KOUSHIK</b>	<b>21071A6779</b>
<b>G.AVINASH</b>	<b>21071A6788</b>



## **VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY**

**(Affiliated to J.N.T.U, Hyderabad)**

**Bachupally(v), Hyderabad, Telangana, India.**

### **ACKNOWLEDGEMENT**

Over a span of one and a half years, VNR VJIET has helped us transform ourselves from mere amateurs in the field of Computer Science into skilled engineers capable of handling any given situation in real time. We are highly indebted to the institute for everything that it has given us. We would like to express our gratitude towards the principal of our institute, **Dr. Challa Dhanunjaya Naidu** and the Head of the Department, Computer Science & Engineering- (CyS, DS) and AI&DS, **Dr. M. Raja Sekar**, for their kind cooperation and encouragement which helped us complete the project in the stipulated time. Although we have spent a lot of time and put in a lot of effort into this project, it would not have been possible without the motivating support and help of our course coordinator of Software Engineering, **Dr. N.Sunanda**, Assistant Professor, Department of CSE- (CyS, DS) and AI&DS. We thank her for the guidance, constant supervision and for providing necessary information to complete this project. Our thanks and appreciations also go to all the faculty members, staff members of VNRVJIET, and all our friends who have helped us put this project together.

# TABLE OF CONTENTS

<b>Details of Contents</b>	<b>Page. No.</b>
Aim	06
Problem Statement	06
Chapter-1	
1.0 Introduction	07
1.1 Purpose	07
1.2 Scope	08
1.3 Acronyms	08
Chapter-2	
2.0 Overall Description	09
2.1 Software Interfaces	09
2.2 Hardware Interfaces	09-10
Chapter-3	
3.0 Functional Requirements	11
3.1 Schema	12-13
3.2 Data	14-15
Chapter-4	
4.1 Use-Case Diagram	16
4.2 E-R Diagram	17
4.3 Class Diagram	18
4.4 Sequence Diagram	18
4.5 Deployment Diagram	19
4.6 Component Diagram	19
Chapter-5	
5.0 DDL and DML Queries	20-23
5.1 Implementation	24-31
Chapter-6	
6.0 Conclusion	32

## AIM

The aim of the movie management system project is to create a comprehensive and efficient software application that facilitates the management and organization of movie-related information. This system is designed to assist movie producers, distributors, theater owners, and other stakeholders in the film industry to handle various aspects of movie management seamlessly. To create a centralized and well-structured database that stores information about movies, including titles, release dates, genres, actors, directors, producers, and other relevant details.

## PROBLEM STATEMENT

The movie industry faces various challenges in managing and organizing movie-related information, which can lead to inefficiencies and hinder optimal performance. The existing manual processes and disparate systems often result in data inconsistencies, delays in distribution, and difficulties in accessing relevant information. Therefore, the problem statement for the movie management system project is as follows:

- **Inefficient Data Management:** The movie industry deals with vast amounts of data, including movie details, cast and crew information, release schedules, and distribution channels. The lack of a centralized and well-organized database leads to data redundancy, inconsistency, and difficulty in updating and maintaining accurate information.
- **Complex Movie Distribution:** Coordinating the distribution of movies to theaters, streaming platforms, and other channels is a complex task. Current distribution methods may be time-consuming and error-prone, resulting in delays in movie releases and lost revenue opportunities.
- **Tedious Theater Screening Management:** Theater owners often face challenges in managing screen schedules and allocating movies to different screens. This can lead to underutilization of screens, inefficient screening arrangements, and conflicts in scheduling.
- **Limited Box Office Insights:** The absence of real-time box office data and revenue tracking hampers informed decision-making by movie producers and distributors. Without timely insights, it becomes challenging to evaluate movie performance and adjust marketing strategies.
- **Lack of Integration:** The movie industry relies on various platforms, including online ticket booking services and streaming platforms. The lack of seamless integration between these platforms and internal movie management systems hinders the overall user experience and revenue generation.
- **Security and Access Control:** With sensitive movie-related data, it is crucial to ensure robust security measures to prevent unauthorized access, data breaches, and potential leaks of unreleased movie information.

In light of these challenges, the movie management system project seeks to develop a comprehensive and user-friendly software application that streamlines movie management processes, optimizes movie distribution, provides real-time insights, and enhances data security. The system aims to facilitate better collaboration among stakeholders in the movie industry, improving the overall efficiency and success of movie production, distribution, and screening.

# CHAPTER-1

## **INTRODUCTION**

The movie industry has always been a significant cultural and entertainment force, captivating audiences with its storytelling and visual artistry. Over the years, the industry has witnessed exponential growth, with a plethora of movies being produced and distributed worldwide. However, managing the complexities of movie production, distribution, and screening efficiently has become a challenging task. The primary objective of the Movie Management System is to provide a centralized platform where all movie-related information can be stored, organized, and accessed with ease. This includes details about movies, such as titles, genres, release dates, cast, crew, and production-related data.

### **1.1 PURPOSE**

The purpose of the Movie Management System is to provide a comprehensive and efficient software solution that serves multiple key objectives and benefits for the movie industry. The system is designed to enhance and streamline various aspects of movie management, benefiting movie producers, distributors, theater owners, and other stakeholders in the following ways:

- **Efficient Data Management:** The primary purpose of the Movie Management System is to establish a centralized and well-organized database to manage vast amounts of movie-related information. This includes details about movies, cast, crew, production, distribution, and screening schedules. By centralizing data, the system ensures data consistency, reduces redundancy, and simplifies data maintenance and updates.
- **Streamlined Movie Distribution:** The system aims to optimize and automate movie distribution workflows. By providing efficient tools to manage the distribution process, it enables seamless coordination between producers and distributors, ensuring timely releases across different distribution channels, such as theaters, streaming platforms, and home media.
- **Effective Theater Screening Management:** The Movie Management System facilitates theater owners in managing their screening schedules and movie allocations efficiently. This purpose is achieved by offering an intuitive interface where theater managers can schedule screenings, allocate movies to different screens, and avoid conflicts, leading to optimized screen utilization and revenue generation.

## **1.2 SCOPE**

- ★ The scope of the Movie Management System encompasses a wide range of functionalities and features designed to meet the diverse needs of the movie industry. It aims to provide a comprehensive solution for managing various aspects of movie production, distribution, and screening.
- ★ The primary scope of the system includes: Movie Database Management: The system will have a centralized and well-organized database to store comprehensive information about movies. This includes details such as movie titles, genres, release dates, cast, crew, production companies, and other relevant data.

## **1.3 ACRONYMS:**

- ★ DVD - Digital Versatile Disc
- ★ Blu-ray - Blu-ray Disc
- ★ VOD - Video on Demand
- ★ OTT - Over-the-Top
- ★ CGI - Computer-Generated Imagery
- ★ IMAX - Image Maximum
- ★ DCP - Digital Cinema Package
- ★ DRM - Digital Rights Management
- ★ API - Application Programming Interface
- ★ UI - User Interface
- ★ UX - User Experience
- ★ CRM - Customer Relationship Management
- ★ ROI - Return on Investment
- ★ EOD - End of Day



## **CHAPTER – 2**

### **2. OVERALL DESCRIPTION:**

The Movie Management System is a comprehensive software application designed to revolutionize the way movies are managed, produced, distributed, and screened within the movie industry. It serves as a centralized platform that caters to the diverse needs of movie producers, distributors, theater owners, and other stakeholders involved in the filmmaking process.

The system's core objective is to streamline and optimize movie management workflows, enhancing efficiency, productivity, and profitability across the industry. It achieves this by offering a range of essential features and functionalities, which can be customized to suit the unique requirements of different movie production houses and distribution companies.

### **2.1 SOFTWARE INTERFACES:**

The Movie Management System interacts with various software interfaces to fulfill its functionalities and cater to the needs of different stakeholders. These interfaces include:

- **User Interface (UI):** The User Interface is the front-end of the Movie Management System that allows users to interact with the system. It provides a visually appealing and intuitive interface through which stakeholders can access and manage movie-related information. The UI is designed to be user-friendly, enabling users to perform tasks such as adding new movies, updating movie details, scheduling screenings, generating reports, and accessing real-time insights.
- **Admin Interface:** The Admin Interface is a specialized user interface designed for administrators or superusers with elevated privileges. It allows administrators to configure system settings, manage user access rights, and perform system maintenance tasks. Admins have control over user management, data backups, security configurations, and other essential system-level functions.

### **2.2 PRODUCT FUNCTIONS**

Hardware interfaces in the context of the Movie Management System refer to the physical connections and devices that enable the system to interact with external hardware components. These interfaces facilitate the smooth functioning and integration of the software with various hardware

devices. Some common hardware interfaces for the Movie Management System include:

Input Devices:

- ★ Keyboard: Users can input data and commands using a standard keyboard for tasks such as data entry and system navigation.
- ★ Mouse or Touchpad: The system can be operated using a mouse or touchpad to select options, click buttons, and interact with graphical elements in the user interface.
- ★ Barcode/QR Code Scanners: The system may support barcode or QR code scanners for quick and accurate entry of movie details, ticket information, or inventory management.

## CHAPTER-3

### 3.0 FUNCTIONAL REQUIREMENTS:

Functional requirements specify the specific tasks, behaviors, and capabilities that the Movie Management System should perform. These requirements outline what the system should do to meet the needs and expectations of its users. Here are some functional requirements for the Movie Management System:

- **Movie Database Management:** a. The system shall allow administrators to add, update, and delete movie records with relevant details such as title, genre, cast, crew, and release date. b. The system shall provide a search and filter functionality to enable users to find specific movies based on various criteria. c. The system shall support the storage and retrieval of movie posters and promotional images.
- **Movie Production Management:** a. The system shall facilitate the tracking of movie production schedules, including pre-production, filming, and post-production activities. b. The system shall allow producers to allocate resources and manage the crew for each movie production. c. The system shall provide a status update feature to monitor the progress of movie production.
- **Movie Distribution Management:** a. The system shall enable movie distributors to manage the distribution process, including assigning movies to theaters, streaming platforms, and other distribution channels. b. The system shall support automated scheduling of movie releases based on release dates and target markets.
- **Theater Screening Management:** a. The system shall allow theater managers to schedule movie screenings on specific dates and times for each screen in the theater. b. The system shall provide conflict resolution features to avoid overlapping movie screenings or double bookings. c. The system shall support the allocation of different movies to multiple screens for multiple showtimes.
- **Box Office and Revenue Tracking:** a. The system shall capture real-time ticket sales data for each movie screening in theaters. b. The system shall calculate and display box office revenue for each movie based on ticket sales. c. The system shall generate reports and analytics on movie performance, including revenue trends and audience attendance.

### 3.1 SCHEMA:

#### CUSTOMER TABLE:

customer(cid, cname, email\_id, phone\_no, tid)

colname	datatype	Data width	Constraints
cid	number	2	Primary key
Cname	Varchar	20	
Email_id	varchar	30	
Phone_no	number	10	
tid	number	2	

#### THEATRE TABLE:

theatre(tname, theatreid, location)

colname	datatpe	Data width	constraints
Tname	varchar	20	
theatreid	Number	2	Primary key
Location	varchar	30	

#### TICKETS TABLE:

tickets(tid, price, seat\_no, show\_date, show\_time, movie\_name, seat\_category)

colname	datatype	Data width	Constraints
Tid	number	2	Foreign key
Price	number	3	
Seat_no	varchar	3	
Show_date	date		
Show_time	varchar	8	
Movie_name	Varchar	50	
Seat_category	varchar	20	

**MOVIE TABLE:**

movie(m\_id, movie\_name, genre, rating)

colname	datatype	Data width	constraints
Mid	number	3	Primary key
Movie_name	Varchar	50	
Genre	varchar	20	
Rating	number	2	

**SHOW TABLE:**

show(st\_time, end\_time, show\_id, language, movie\_name)

colname	datatype	Data width	Constraints
St_time	varchar	8	
End_time	varchar	8	
Show_id	number	2	Primary key
Language	varchar	10	
Movie_name	varchar	50	

**BOOKING TABLE:**

booking(cname, tid, tname, show\_date, show\_time, show\_id)

colname	datatype	Data width	Constraints
cname	Varchar	20	
tid	number	3	Primary key
tname	varchar	20	
Show_date	Date		
Show_time	varchar	8	
Show_id	number	2	Foreign key

## DATA:

### CUSTOMER TABLE:

cid	cname	Email_id	Phone_no	Tid
1	Ramesh	<a href="mailto:ramesh@gmail.com">ramesh@gmail.com</a>	9391249726	53
2	Suresh	<a href="mailto:suresh@gmail.com">suresh@gmail.com</a>	9030925895	1
3	Leo	<a href="mailto:mesiileo@gmail.com">mesiileo@gmail.com</a>	7382028600	29
4	Ronaldo	<a href="mailto:ronald@gmail.com">ronald@gmail.com</a>	8790610794	3
5	Virat	<a href="mailto:vk1825@gmail.com">vk1825@gmail.com</a>	6302206762	40
6	Kiran	<a href="mailto:kiran@gmail.com">kiran@gmail.com</a>	7981015171	59
7	Ramesh	<a href="mailto:ramesh@gmail.com">ramesh@gmail.com</a>	9398572700	68
8	Ganesh	<a href="mailto:ganesh@gmail.com">ganesh@gmail.com</a>	9849496697	78
9	Vijay	<a href="mailto:vijay@gmail.com">vijay@gmail.com</a>	6302636827	80
10	Nikitha	<a href="mailto:nikitha@gmail.com">nikitha@gmail.com</a>	9502291707	90

### THEATRE TABLE:

tname	Theatre_id	Location
Sai Ranga theatre	40	Miyapur
Sri bhrmaramba hall	60	Jntu
Miraj theater	90	Miyapur
Viswanath 70MM AC	70	KPHB main road
Prasads multiplex	80	IMAX road, NTR Marg

### TICKETS TABLE:

tid	price	Seat_no	Show_date	Show_time	Movie_name	Seat_category
5	250	A1	15-SEP-22	2:00PM	Vikram	VIP SEATING
15	170	B6	02-JUL-20	6:00PM	Major	GOLD
25	120	C7	09-MAR-13	10:00PM	RRR	SILVER
35	150	D3	30-AUG-08	2:00PM	Ante sundaraniki	PLATINUM
45	170	B1	24-DEC-15	7:00AM	Virata parvam	GOLD
55	250	A6	08-SEP-12	6:00PM	RRR	VIP SEATING
65	120	C8	12-JUN-22	10:00PM	Vikram	SILVER
35	150	D5	30-AUG-20	2:00PM	Major	PLATINUM
75	170	B9	08-SEP-17	7:00AM	Sammatame	GOLD
85	250	A6	08-AUG-16	10:00PM	RRR	VIP SEATING

**MOVIE TABLE:**

M_id	Movie_name	genre	rating
13	Vikram	action	8.8
23	Major	drama	8.7
33	RRR	action	8
43	Ante sundariniki	comedy	8.2
53	Virata parvam	thriller	8.5
63	Sammataame	Romance	6.8

**SHOW TABLE:**

St_time	End_time	Show_id	language	Movie_name
7:00AM	10:00AM	6	Telugu	RRR
2:00PM	5:00PM	12	Tamil	Vikram
6:00PM	9:00PM	18	Telugu	Major
10:00PM	1:00PM	24	Telugu	Ante sundariniki

**BOOKING TABLE:**

cname	tid	tname	Show_date	Show_time	Show_id
Sandeep	25	Mallikarjuna theatre	09-MAR-13	10:00PM	6
Keerthi	15	Viswanath 70MM AC	02-JUL-20	6:00PM	18
Sahithi	5	Prasads multiplex	15-SEP-22	2:00PM	12
Anjali	35	Mallikarjuna theatre	30-AUG-20	2:00PM	24
Roshini	45	Sri bhramaramba cinema hall	24-DEC-15	7:00AM	6
Vijay	85	Viswanath 70MM AC	08-AUG-16	10:00PM	24

## CHAPTER-4

### 4.0 UML DIAGRAMS:

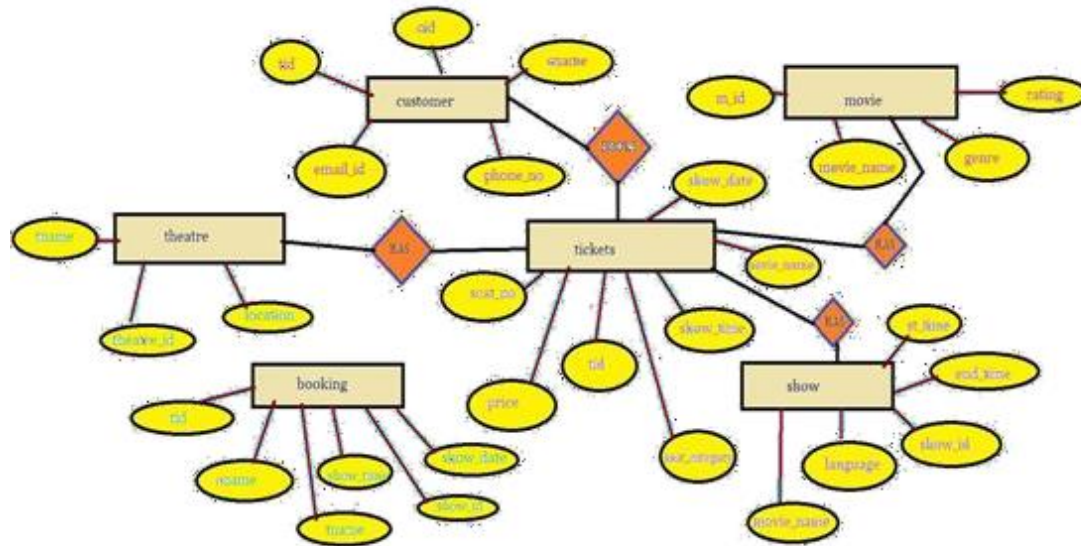
A UML diagram is a diagram based on the UML (Unified Modeling Language) with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or classes, in order to better understand, alter, maintain, or document information about the system.

### 4.1 USE-CASE DIAGRAM:

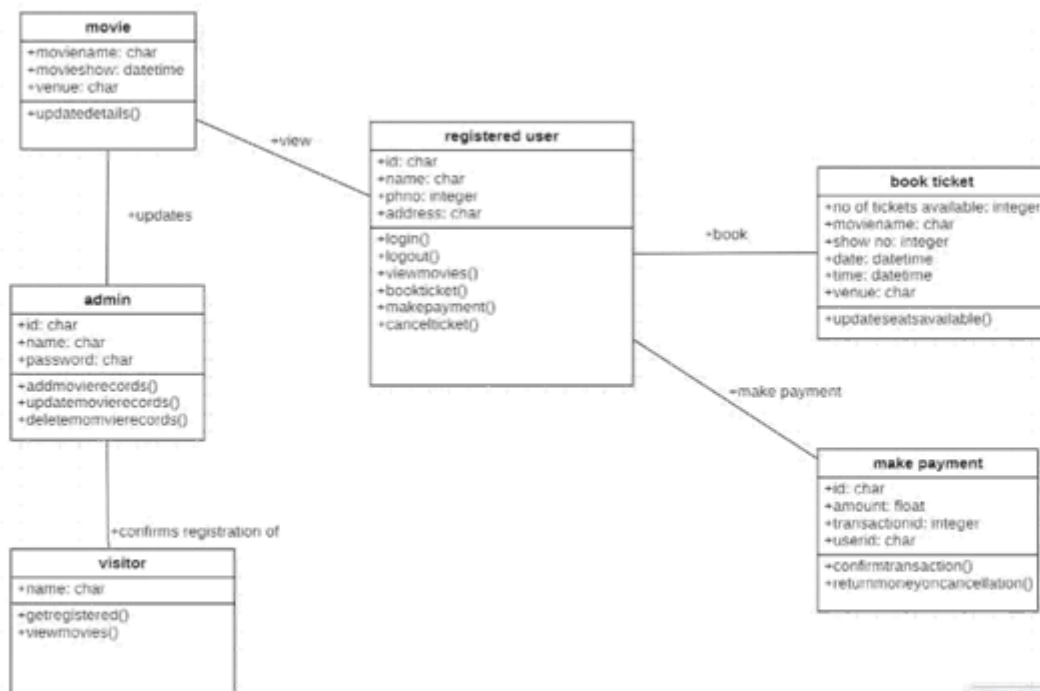




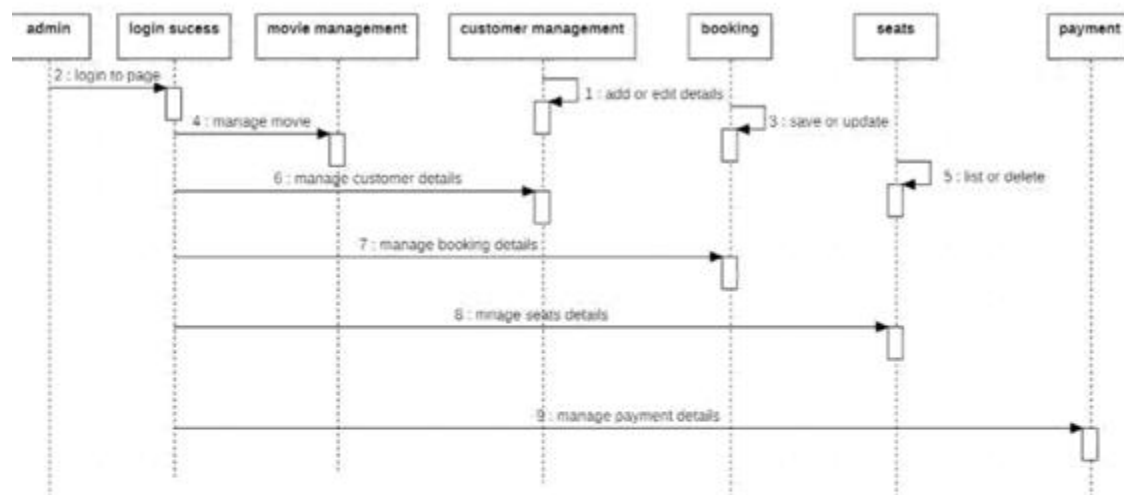
## 4.2 ER DIAGRAM:



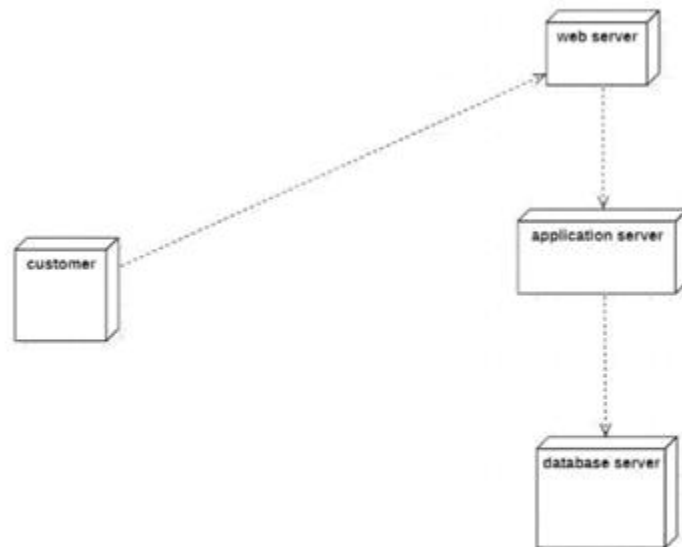
### 4.3 CLASS DIAGRAM:



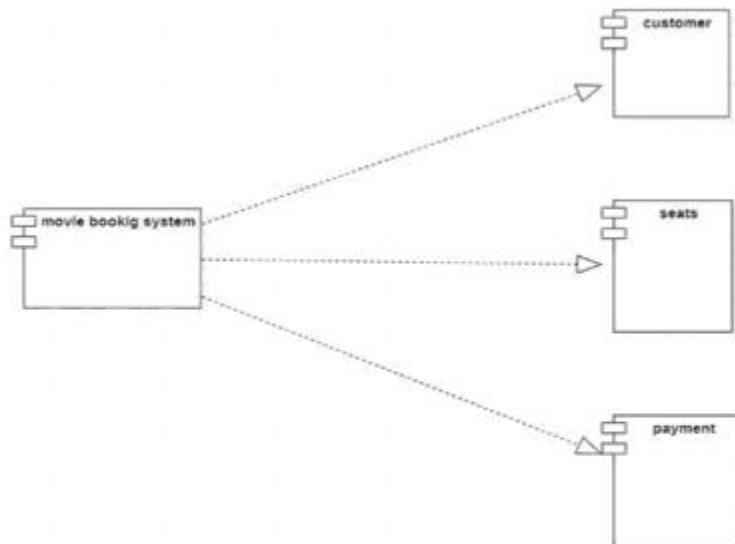
### 4.4 SEQUENCE DIAGRAM:



## 4.5 DEPLOYMENT DIAGRAM



## 4.6 COMPONENT DIAGRAM



## CHAPTER-5

### 5.0 DDL AND DML COMMANDS:

#### CUSTOMER TABLE:

```
create table customers(cid number(2) primary key, cname varchar(20),  
email_id varchar(30), phone_no number(10), tid number(2));
```

```
insert into customers values(1, 'sahithi', 'sahithi@gmail.com', 9391249726, 5);  
insert into customers values(2, 'keerthi', 'keerthi@gmail.com', 9030925895, 15);  
insert into customers values(3, 'sandeep', 'sandeep@gmail.com', 7382028600, 25);  
insert into customers values(4, 'anjali', 'anjali@gmail.com', 8790610794, 35);  
insert into customers values(5, 'roshini', 'roshini@gmail.com', 6302206762, 45);  
insert into customers values(6, 'kiran', 'kiran@gmail.com', 7981015171, 55);  
insert into customers values(7, 'ramesh', 'ramesh@gmail.com', 9398572700, 65);  
insert into customers values(8, 'ganesh', 'ganesh@gmail.com', 9849496697, 75);  
insert into customers values(9, 'vijay', 'vijay@gmail.com', 6302636827, 85);  
insert into customers values(10, 'nikitha', 'nikitha@gmail.com', 9502291707, 95);  
select * from customers;
```

CID	CNAME	EMAIL_ID	PHONE_NO	TID
1	sahithi	sahithi@gmail.com	9391249726	5
2	keerthi	keerthi@gmail.com	9030925895	15
3	sandeep	sandeep@gmail.com	7382028600	25
4	anjali	anjali@gmail.com	8790610794	35
5	roshini	roshini@gmail.com	6302206762	45
6	kiran	kiran@gmail.com	7981015171	55
7	ramesh	ramesh@gmail.com	9398572700	65
8	ganesh	ganesh@gmail.com	9849496697	75
9	vijay	vijay@gmail.com	6302636827	85
10	nikitha	nikitha@gmail.com	9502291707	95

[Download CSV](#)

10 rows selected.

## THEATRE TABLE:

```
create table theatre(tname varchar(30), theatreid number(2), location varchar(30), constraint theatre_pk primary key (theatreid));
```

```
insert into theatre values('Mallikarjuna theatre', 30, 'Kukatpally');
insert into theatre values('Sri bharamaramba cinema hall', 50, 'Vasant Nagar');
insert into theatre values('Shiva Parvathi theater', 60, 'Kukatpally');
insert into theatre values('Viswanath 70MM AC', 70, 'KPHB main road');
insert into theatre values('Prasads multiplex', 80, 'IMAX road, NTR Marg');
select * from theatre;
```

TNAME	THEATREID	LOCATION
Mallikarjuna theatre	30	Kukatpally
Sri bharamaramba cinema hall	50	Vasant Nagar
Shiva Parvathi theater	60	Kukatpally
Viswanath 70MM AC	70	KPHB main road
Prasads multiplex	80	IMAX road, NTR Marg

[Download CSV](#)

5 rows selected.

## TICKETS TABLE:

```
create table tickets(tid number(2), price number(3), seat_no varchar(3), show_date date, show_time varchar(8), movie_name varchar(50), seat_category varchar(10), foreign key(tid) references booking(tid));
```

```
insert into tickets values(5, 250, 'A1', '15-SEP-22', '2:00PM', 'Vikram', 'VIP SEATING');
insert into tickets values(15, 170, 'B6', '02-JUL-20', '6:00PM', 'Major', 'GOLD');
insert into tickets values(25, 120, 'C7', '09-MAR-13', '10:00PM', 'RRR', 'SILVER');
insert into tickets values(35, 150, 'D3', '30-AUG-08', '2:00PM', 'Ante sundaraniki', 'PLATINUM');
insert into tickets values(45, 170, 'B1', '24-DEC-15', '7:00AM', 'Virata parvam', 'GOLD');
insert into tickets values(55, 250, 'A6', '08-SEP-12', '6:00PM', 'RRR', 'VIP SEATING');
insert into tickets values(65, 120, 'C8', '12-JUN-22', '10:00PM', 'Vikram', 'SILVER');
insert into tickets values(35, 150, 'D5', '30-AUG-20', '2:00PM', 'Major', 'PLATINUM');
insert into tickets values(75, 170, 'B9', '08-SEP-17', '7:00AM', 'Sammataame', 'GOLD');
insert into tickets values(85, 250, 'A6', '08-AUG-16', '10:00PM', 'RRR', 'VIP SEATING');
select * from tickets;
```

## MOVIE TABLE:

```
create table movie(mid number(3), movie_name varchar(50), genre varchar(20), rating number(2), constraint movie_pk primary key (mid))
```

```
insert into movie values(13, 'Vikram', 'Action', 8.8);
insert into movie values(23, 'Major', 'Drama', 8.7);
insert into movie values(33, 'RRR', 'Action', 8);
insert into movie values(43, 'Ante Sundaraniki', 'Comedy', 8.2);
insert into movie values(53, 'Virata Parvam', 'Thriller', 8.5);
insert into movie values(63, 'Sammataame', 'Romance', 6.8);
select * from movie;
```

MID	MOVIE_NAME	GENRE	RATING
13	Vikram	Action	9
23	Major	Drama	9
33	RRR	Action	8
43	Ante Sundaraniki	Comedy	8
53	Virata Parvam	Thriller	9
63	Sammataame	Romance	7

[Download CSV](#)

6 rows selected.

## SHOW TABLE:

```
create table show(st_time varchar(8), end_time varchar(8), show_id number(2) primary key, language varchar(10), movie_name varchar(50));
```

```
insert into show values('7:00AM', '10:00AM', 6, 'Telugu', 'RRR');
insert into show values('2:00PM', '5:00PM', 12, 'Tamil', 'Vikram');
insert into show values('6:00AM', '9:00AM', 18, 'Telugu', 'Major');
insert into show values('10:00PM', '1:00PM', 24, 'Telugu', 'Ante sundariniki');
select * from show;
```



ST_TIME	END_TIME	SHOW_ID	LANGUAGE	MOVIE_NAME
7:00AM	10:00AM	6	Telugu	RRR
2:00PM	5:00PM	12	Tamil	Vikram
6:00AM	9:00AM	18	Telugu	Major
10:00PM	1:00PM	24	Telugu	Ante sundariniki

[Download CSV](#)

4 rows selected.

## BOOKING TABLE:

```
create table booking(cname varchar(20), tid number(3), tname varchar(20), show_date date, show_time varchar(10), show_id number(2), primary key(tid), foreign key(show_id)
references show(show_id));
```

```
insert into booking values('sandeep', 25, 'Mallikarjuna theatre', '09-MAR-13', '10:00PM', 6);
insert into booking values('keerthi', 15, 'Viswanath 70MM AC', '02-JUL-20', '6:00PM', 18);
insert into booking values('sahithi', 5, 'Prasads multiplex', '15-SEP-22', '2:00PM', 12);
insert into booking values('anjali', 35, 'Mallikarjuna theatre', '30-AUG-20', '2:00PM', 24);
insert into booking values('roshini', 45, 'Sri bhramaramba cinema hall', '24-DEC-15', '7:00PM', 6);
insert into booking values('vijay', 85, 'Viswanath 70MM AC', '08-AUG-16', '10:00PM', 24);
select * from bookings;
```

CNAME	TID	TNAME	SHOW_DATE	SHOW_TIME	SHOW_ID
sandeep	25	Mallikarjuna theatre	09-MAR-13	10:00PM	6
keerthi	15	Viswanath 70MM AC	02-JUL-20	6:00PM	18
sahithi	5	Prasads multiplex	15-SEP-22	2:00PM	12
anjali	35	Mallikarjuna theatre	30-AUG-20	2:00PM	24
roshini	45	Sri bhramaramba cinema hall	24-DEC-15	7:00PM	6
vijay	85	Viswanath 70MM AC	08-AUG-16	10:00PM	24

[Download CSV](#)

6 rows selected.

## 5.1 Executing Queries with JDBC:

1. Print the list of customers who booked the tickets for the show\_id > 10

### Code:

```
package program79;

import java.sql.*;

public class fileinputstream {

    public static void main(String args[]){

        try{

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

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select booking.cname,shows.show_id from
booking inner join shows on booking.show_id=shows.show_id where shows.show_id>10");

            while(rs.next())

            {

                System.out.println(rs.getString(1)+" "+rs.getInt(2));

            }

        }

        catch(Exception e)

        {

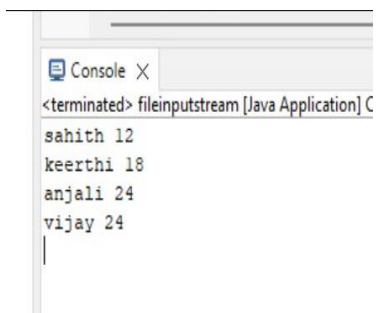
            System.out.println(e);

        }

    }

}
```

### output:





## 2. Print the list of movies watched by a specific customer

### Code:

```
package program79;

import java.sql.*;

public class fileinputstream {

    public static void main(String args[]) {

        try

        {

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

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select booking.cname, shows.movie_name from
booking inner join shows on booking.show_id = shows.show_id;");

            while(rs.next())

            {

                System.out.println(rs.getString(1)+" "+rs.getString(2));

            }

        }

        catch(Exception e)

        {

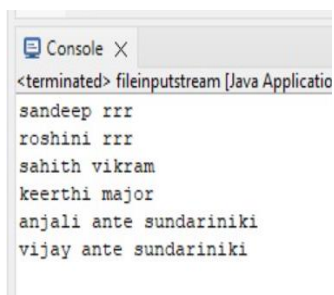
            System.out.println(e);

        }

    }

}
```

### output:



```
<terminated> fileinputstream [Java Applicatio
sandeep rrr
roshini rrr
sahith vikram
keerthi major
anjali ante sundariniki
vijay ante sundariniki
```

3. print the movies present whose rating is above 8

**code:**

```
package program79;

import java.sql.*;

public class fileinputstream {

    public static void main(String args[]){

        try

        {

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

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select movie_name from movie where
rating>=8");

            while(rs.next())

            {

                System.out.println(rs.getString(1));

            }

        }

        catch(Exception e)

        {

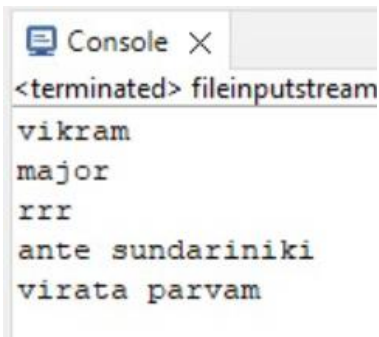
            System.out.println(e);

        }

    }

}
```

**output:**



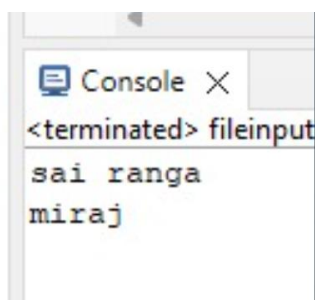
```
<terminated> fileinputstream
vikram
major
rrr
ante sundariniki
virata parvam
```

4. Print the list of theatres present in 'miyapur'.

**Code:**

```
package program79;
import java.sql.*;
public class fileinputstream {
    public static void main(String args[]){
        try
        {
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movies","root","root");
            Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select tname from theatre where
location='miyapur'");
            while(rs.next())
            {
                System.out.println(rs.getString(1));
            }
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

**output:**



5. Print the list of customers who booked seats of 'gold' category

**Code:**

```
package program79;

import java.sql.*;

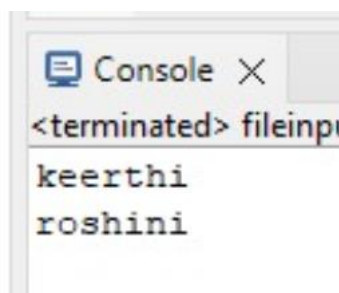
public class fileinputstream {
    public static void main(String args[]){
        try
        {
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movies","root","root");
            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select booking.cname from ticket
inner join booking on ticket.tid = booking.tid inner join shows on booking.show_id =
shows.show_id where ticket.seat_category like 'gold'");

            while(rs.next())
            {
                System.out.println(rs.getString(1));

            }
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

**output:**



6. Print the no of customers for each seat category

**Code:**

```
package program79;

import java.sql.*;

public class fileinputstream {

    public static void main(String args[]){

        try

        {

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

            Statement stmt=con.createStatement();

            ResultSet rs=stmt.executeQuery("select ticket.seat_category,
count(booking.cname) from ticket inner join booking on ticket.tid = booking.tid inner join
shows on booking.show_id = shows.show_id group by ticket.seat_category;");

            while(rs.next())

            {

                System.out.println(rs.getString(1)+ " "+rs.getInt(2));

            }

        }

        catch(Exception e)

        {

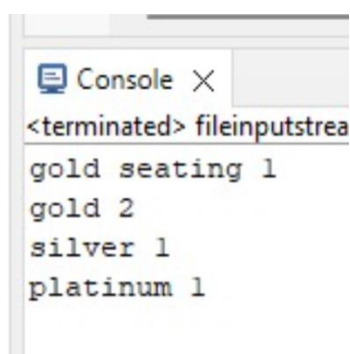
            System.out.println(e);

        }

    }

}
```

**output:**



7. Print the list of theatres and the movies being shown in 'jntu'.

**Code:**

```
package program79;

import java.sql.*;

public class fileinputstream {

    public static void main(String args[]){

        try

            {

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

                Statement stmt=con.createStatement();

                ResultSet rs=stmt.executeQuery("select theatre.tname,
shows.movie_name from theatre inner join booking on theatre.tname = booking.tname inner
join shows on booking.show_id =shows.show_id where theatre.location = 'jntu'");

                while(rs.next()){

                    System.out.println(rs.getString(1)+" "+rs.getString(2));

                }

            }

        catch(Exception e) {

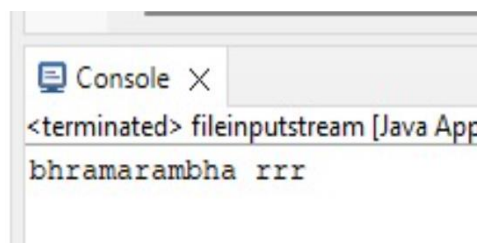
            System.out.println(e);

        }

    }

}
```

**Output:**



## 8. Print the average rating of movies based on languages

### Code:

```
package program79;

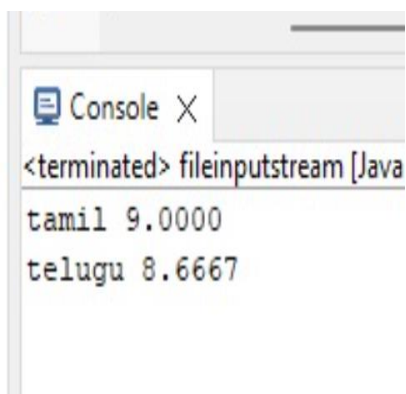
import java.sql.*;

public class fileinputstream {
    public static void main(String args[]){
        try
        {
            Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/movies","root","root");
            Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select shows.language,
avg(movie.rating) from movie inner join shows on movie.movie_name = shows.movie_name group by
shows.language;");

            while(rs.next())
            {
                System.out.println(rs.getString(1)+" "+rs.getString(2));

            }
        }
        catch(Exception e)
        {
            System.out.println(e);
        }
    }
}
```

### output:



## **6.0 CONCLUSION:**

Movie ticket booking system is an efficient as well as flexible way of storing and maintaining the data related to various movies that are being screened in multiple theatres across the cities.

With this project we are able to store, manipulate and retrieve the data from the database and view them in various tabular combinations.

Our primary concern in this project is to implement various concepts that are involved in real time applications to demonstrate the production activities such as viewing and booking tickets. We have tried to incorporate the concepts like DDL, DML, etc into the project to correlate it with the real time scenario.