

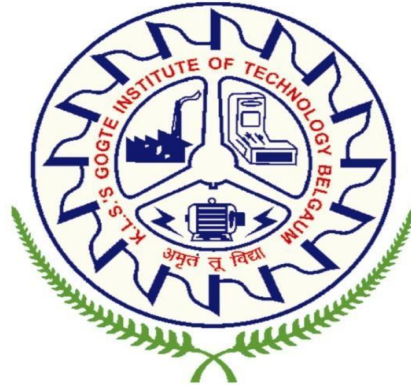
KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELAGAVI-590008

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING



Course Activity Report

THEATRE DATABASE MANAGEMENT SYSTEM(18IS43)

Submitted in the partial fulfillment for the academic requirement of

4th SEMESTER B.E

IN

DATABASE MANAGEMENT SYSTEM

Submitted by

NAME OF THE CANDIDATES

USN

1. RAHUL KULKARNI

2GI20IS053

2. OMKAR DESHPANDE

2GI20IS019

3. PRANAV PATIL

2GI20IS061

Under the guidance of: Dr. Sudhindra Deshpande

Asst. Professor, Dept Of ISE

KARNATAK LAW SOCIETY'S

GOGTE INSTITUTE OF TECHNOLOGY

UDYAMBAG, BELAGAVI-590008

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING



CERTIFICATE

This is to certify that **Rahul Kulkarni, Omkar Deshpande, Pranav Patil**, of **Fourth Semester** bearing **USN: 2GI20IS053, 2GI20IS019, 2GI20IS061** has satisfactorily completed the course in *Course Project of Database management system(18IS43)*. It can be considered as a bonafide work carried out for partial fulfilment of the academic requirement of 4th Semester B.E. (Information Science & Engineering) prescribed by KLS Gogte Institute of Technology, Belagavi during the academic year 2021-22. The report has been approved as it satisfies the academic requirements prescribed for the said degree.

Signature of the Faculty Member

Signature of the HOD

Rubrics for evaluation of Course project:**Marks allocation:**

| Sl. No | Batch No.: 03 | | | USN | |
|--------|--------------------------------------------------------------------------------------------------------|-------------|------------|------------|------------|
| 1. | Project Title: Theatre Management System | Marks Range | 2GI20IS053 | 2GI20IS019 | 2GI20IS061 |
| 2. | Problem statement (PO2) | 0-1 | | | |
| 3. | Objectives of defined problem statement (PO1,PO2) | 0-2 | | | |
| 4. | Design /Algorithm/ Flowchart/ Methodology (PO3) | 0-3 | | | |
| 5. | Implementation details/ function/ Procedures/classes and objects (Language/tools) (PO1, PO3, PO4, PO5) | 0-4 | | | |
| 6. | Working model of the final solution (PO3, PO12) | 0-5 | | | |
| 7. | Report and Oral presentation skill (PO9, PO10) | 0-5 | | | |
| | Total | 20 | | | |

*** 20 marks is converted to 10 marks for CGPA calculation**

1.Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

2.Problem Analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and Engineering sciences.

3.Design/Development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4.Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

- 5. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- 6. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
- 7. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
- 8. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
- 9. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 10. Individual and team work:** Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.
- 11. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 12. Project management finance:** Demonstrate knowledge and understanding of the engineering management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 13. Life-long learning:** Recognize the need for and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

THEATRE MANAGEMENT SYSTEM

A Database management system to store information about the movie bookings in the
theatre

INDEX

| | |
|----------------------------------------------------------------|----|
| ABSTRACT | I |
| LIST OF FIGURES | V |
| 1. Introduction | 1 |
| 1.1. Problem Statement | 1 |
| 1.2. Objectives..... | 1 |
| 1.3. Methodology..... | 1 |
| 2. Literature survey..... | 2 |
| 3. Requirement Specifications | 3 |
| 3.1.External interface requirement..... | 3 |
| 3.2.Functional Requirements..... | 3 |
| 3.3.Non-functional Requirements..... | 4 |
| 4. System Design..... | 5 |
| 4.1. ER diagram..... | 5 |
| 4.1.1. Entities..... | 5 |
| 4.1.2. Relationships..... | 5 |
| 4.1.3. Attributes of each entity, Relationship..... | 5 |
| 4.1.4. Cardinality ratios..... | 5 |
| 4.1.5. Participation Ratios..... | 6 |
| 4.1.6. ER diagram of the proposed system..... | 6 |
| 4.2.Normalization..... | 7 |
| 4.3.Schema Diagram | 8 |
| 4.3.1. Mapping of ER diagram to relational schema diagram..... | 8 |
| 4.4. Relations in Third Normal Form..... | 10 |
| 4.5. Functional Dependencies..... | 10 |
| 4.6. Justification table..... | 11 |

| | |
|-----------------------------------------------|----|
| 5. Implementation..... | 12 |
| 5.1.Database table structure..... | 12 |
| 5.2. SQL queries used..... | 14 |
| 5.3.Show Tables..... | 16 |
| 5.4.Queries..... | 20 |
| 6. Results and Discussions (Screenshots)..... | 21 |
| 7. Conclusion | 24 |
| 8. References..... | 24 |

ABSTRACT

To define, design, and implement a Database to aid in comprehensively managing all aspects of a theatre. This system deals with a single branch of a theatre. It keeps track of the various halls, currently showing movies, show timings, booked tickets, and price listings in the theatre as well as the associated attributes. It is possible for the management of the theatre to use the database to fully understand all parts of the system without use of additional software. The system allows a cashier to add tickets for a given show, and for a manager to add movies and shows to the database. The system automatically checks the validity of the new entries and disallows incorrect data from being entered. This is done automatically without user intervention.

LIST OF FIGURES

4.1. ER diagram

Diagram 4.1.D1: Entity Relation Diagram

4.2. Normalization

Diagram 4.2. D1: Normalization

4.3. Relations in Third Normal Form (3NF)

Diagram 4.3.D1: Functional Dependencies showing 3NF

4.4. Schema diagram

Diagram 4.4.D1: Schema Diagram

5 Results snaps

Diagram 5.D1: Home Page

Diagram 5.D2: Browse Movies page

Diagram 5.D3: Book tickets page

Diagram 5.D4: Contact Us page

Diagram 5.D5: Database for Contact Us table

Diagram 5.D6: Database for booking tickets table

1. INTRODUCTION

1.1. Problem statement:

Design a model describing Theatre Management System which deals with the movies available, current shows, price lists, booked tickets, halls.

1.2. Objectives:

- The main objective of this project is to issue tickets to the customer and reserve their seats.
- The other objective is to book tickets at any theatre to any show at any day.
- The other important objective is to track all the account details particularly number of tickets sold for each show in each theatre.

1.3. Methodology:

Theatre Management: This class contains the details of theatre.

Movie: It contains the movie's details along with their hall number and timings.

Hall: This class contains the movie screen in the theatre and it is the place where people sit and watch the movie.

Seats: This shows the details of seats in a particular hall and describes the type of the seat. The type of seats can be front seat, back seat, balcony seat.

2. LITERATURE SURVEY:

2.1. Existing System:

Already existing app/website which is related to our database project is BOOKMYSHOW.

Book My Show is India's biggest online movie and events ticketing brand. The website caters to ticket sales for movies, plays, concerts and sporting events via the online platform. Launched in 2007, it is headquartered in Mumbai, Maharashtra. It additionally has offices in New Delhi, Bangalore, Hyderabad, Chennai and Kolkata.

Book My Show is an online ticketing facility like Movietickets.com, Explara and Ticketmaster.com. Book My Show took the primary services provided by these two websites and consolidated it into one website for movies, plays, events and sports tickets. Apart from being an online ticketing portal, Book My Show offers information about upcoming movies and events, show timings, venue details and artist bios.

3. REQUIREMENT SPECIFICATION:

3.1. External Interface Requirements:

Various Interfaces of the Product could be:

1. Login Page
2. Registration Form
3. There will be screen displaying information about product that the User will be using.
4. If the customers select the buy button, then another screen of booking Information of their show and seats will be opened.
5. After all transaction the system makes the selling report as portable Document file and sent to the customer E-mail address.

- **HARDWARE INTERFACE:**

The system must run over the internet, all the hardware shall require to Connect the internet will be the hardware interface for the system.

For example: Modem, WAN-LAN, Ethernet Cross-Cable.

- **SOFTWARE INTERFACE:**

The System is on server so it requires the any scripting language like PHP, HTML & CSS. The system required data base also for the store of Any transaction of the system like MYSQL etc. system also require DNS (domain name space) for the naming on the internet. At last user need web browser to interact with the system. There is no need of performance requirement in this system because the Server request and response is dependent on the end user internet Connection.

3.2. Functional Requirements:

The functional requirements of this project are:

- The ability to insert new Movies. Movie name, language, length, showing start date and end date are inputted.
- Similarly, Shows can be entered by entering movie, hall, time and date. If correctly entered, a show ID is randomly generated and ask data is put into database. Then, the price ID is fetched and stored after. Shows cannot be added if an invalid type is chosen or if the hall in which it is to be shown is already occupied.

- A showing is selected for a time and date. Then a seating chart is brought up and a seat is chosen. Then a unique ticket ID is generated and the data is stored in the database.
- All data is stored on a common database. All systems in the theatre (cashier's terminals and manager's computer) access the same database for their bookings. Multiple branches of the same theatre may share the same database, but different theatres use different database.

3.3. Non-functional Requirements:

The non-functional requirements of this projects are:

- Security: The website does not allow access to any functionality by directly jumping to any particular link to that function's page. Additionally, anything that is needed to be done can only be done by first logging in. There are multiple accounts, with login by cashiers only allowing cashier functionalities and manager allowing only manager functionalities.
- Data Integrity: The project does not allow entry of data in case data is invalid. This is very important as if invalid data is added, then it can cause large problems, such as getting tickets for shows which are invalid, creating shows for invalid or incorrect movies/halls, or inserting invalid movies, each of which can have cascading effects.
- Automatic data processing: a lot of information is processed by the project instead of relying on the user to add perfect information and perform numerous functions each time. Examples include deleting old shows and movies, retrieving prices, validating inserted information in movies/shows etc. This is an important task as it can be performed much more efficiently and quickly by the system than by a human

4. SYSTEM DESIGN:

4.1. ER-Diagram:

The ER diagram for Theatre management system would comprise the following data:

4.1.1. Entities:

- Halls
- Movies
- Shows
- Booked_Tickets
- Prices

4.1.2. Relationships:

- Plays in
- Played at
- Booked
- Costs

4.1.3. Attributes of each Entity, Relationship:

HALLS (Hall_ID, Class, No_of_Seats)

MOVIES (Movie_ID, Movie_Name, Length, Language, Show_start, Show_End, Types)

SHOWS (Show_ID, Date, Time)

PRICE_LISTS (Price_ID, Type, Price, Day)

BOOKED_TICKETS (Seat_No, Ticket_No)

TYPES (Movie_ID, 4DX, 3D, 2D)

4.1.4. Cardinality Ratios:

MOVIES played at SHOWS (N:1)

SHOWS costs PRICE (1: N)

SHOWS plays in HALLS (N: 1)

SHOWS booked BOOKED_TICKETS (1: N)

4.1.5. Participation Ratios:

MOVIES played at SHOWS (T: T)

SHOWS costs PRICE (T: T)

SHOWS plays in HALLS (T: P)

SHOWS booked BOOKED_TICKETS (P: T)

4.1.6. ER Diagram of the proposed system:

The ER diagram of Proposed system is as shown in the figure

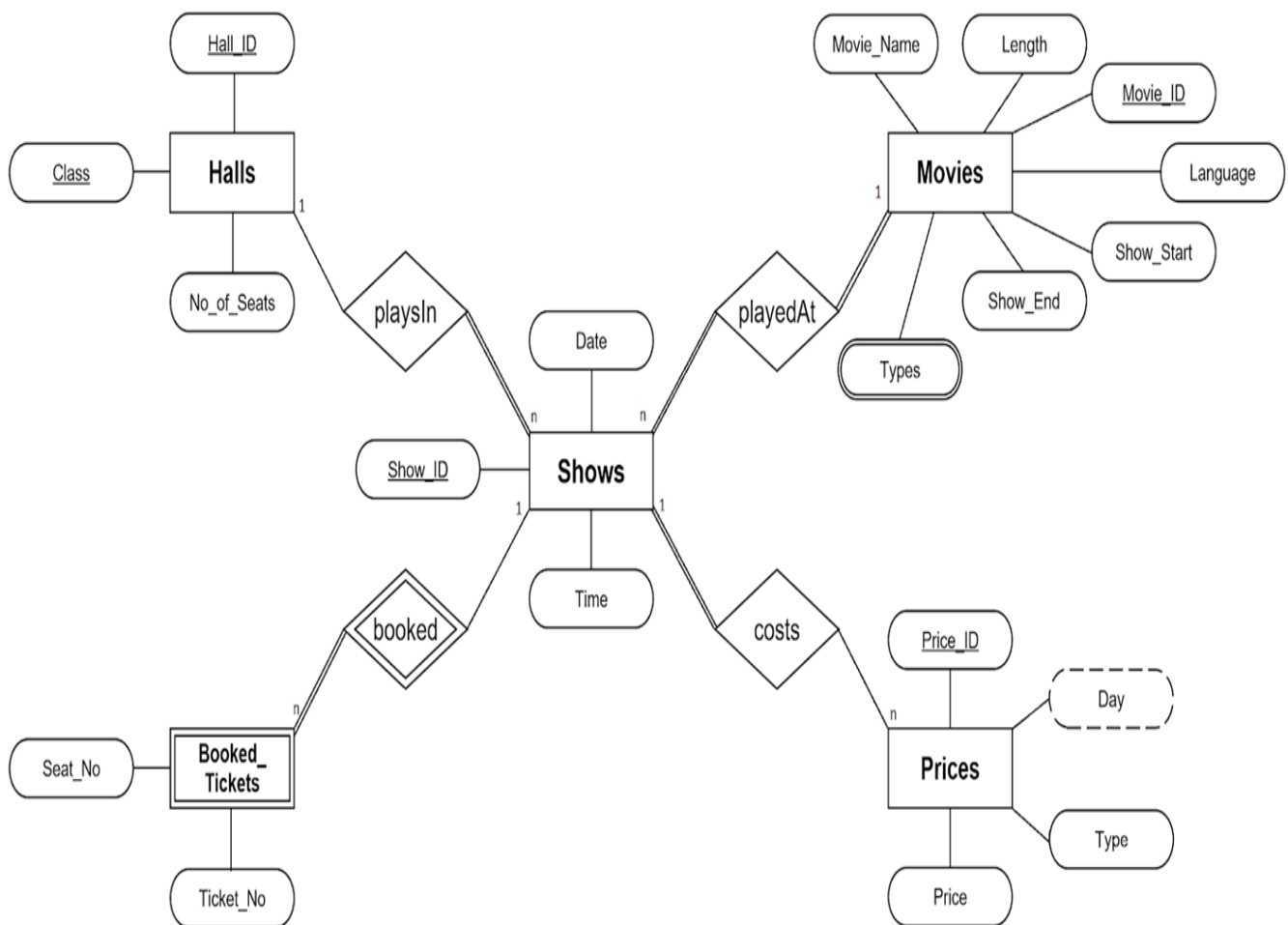


Diagram 4.1.D1: Entity Relation Diagram

4.2. Normalization:

HALLS

| <u>Hall_ID</u> | <u>Class</u> | No_Of_Seats |
|----------------|--------------|-------------|
| | | |

MOVIES

| <u>Movie_ID</u> | Movie_Name | Language | Length | Types | Show_Start | Show_End |
|-----------------|------------|----------|--------|-------|------------|----------|
| | | | | | | |

SHOWS

| <u>Show_ID</u> | Hall_ID | Movie_ID | Type | Time | Date | Price_ID |
|----------------|---------|----------|------|------|------|----------|
| | | | | | | |

PRICE LISTS

| <u>Price_ID</u> | Type | Day | Price |
|-----------------|------|-----|-------|
| | | | |

BOOKED TICKETS

| <u>Ticket_No</u> | <u>Show_ID</u> | Seat_No |
|------------------|----------------|---------|
| | | |

TYPES

| <u>Movie_ID</u> | Type 1 | Type 2 | Type 3 |
|-----------------|--------|--------|--------|
| | | | |

Diagram 4.2. D1: Normalization

First Normal Form(1NF):

As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.

The above schema is in 1NF since all the attributes are atomic and not multivalued.

Second Normal Form(2NF):

A table is said to be in 2NF if both the following conditions hold:

- Table is in 1NF (First normal form)
- No non-prime attribute is dependent on the proper subset of any candidate key of table.

Third Normal Form(3NF):

A table design is said to be in 3NF if both the following conditions hold:

- Table must be in 2NF
- Transitive functional dependency of non-prime attribute on any super key should be

removed. Our schema follows the above rules and hence is in 3NF.

Department of Information Science & Engineering

4.3. Schema Diagram:

4.3.1. Mapping of ER diagram to relational schema diagram

The mapping of ER diagram shown in the figure 4.1 to relational schema diagram is done with respect to the following guidelines

Step 1: Mapping of Regular Entity Types

For each regular (strong) entity type E in the ER schema we have created a relation R that includes all the simple attributes of E. Choose one of the key attributes of E as the primary key for R. If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

Created the relations Halls, Movies, Shows, Price_listing in the relational schema corresponding to the regular entities in the ER diagram. Hall_id, Movie_ID, Show_ID ,Price_Id and Ticket_no are the primary keys for the relations as shown in the figure 3.2.

Step 2: Mapping of Weak Entity Types

There is one weak entity booked tickets in our database.

Step 3: Mapping of Binary 1:1 Relation Types

There are no 1:1 relationship in our database.

Step 4: Mapping of Binary 1:N Relationship Types

1:N relationship types are shows and price_listing and shows and booked_tickets.

Step 5: Mapping of Binary M:N and N:1 Relationship Types

There are no M:N relationship in our database.

There exists N:1 relation between shows and movies and shows and halls.

The schema diagram of the proposed system after applying the & guidelines is as shown in the Figure 4.4

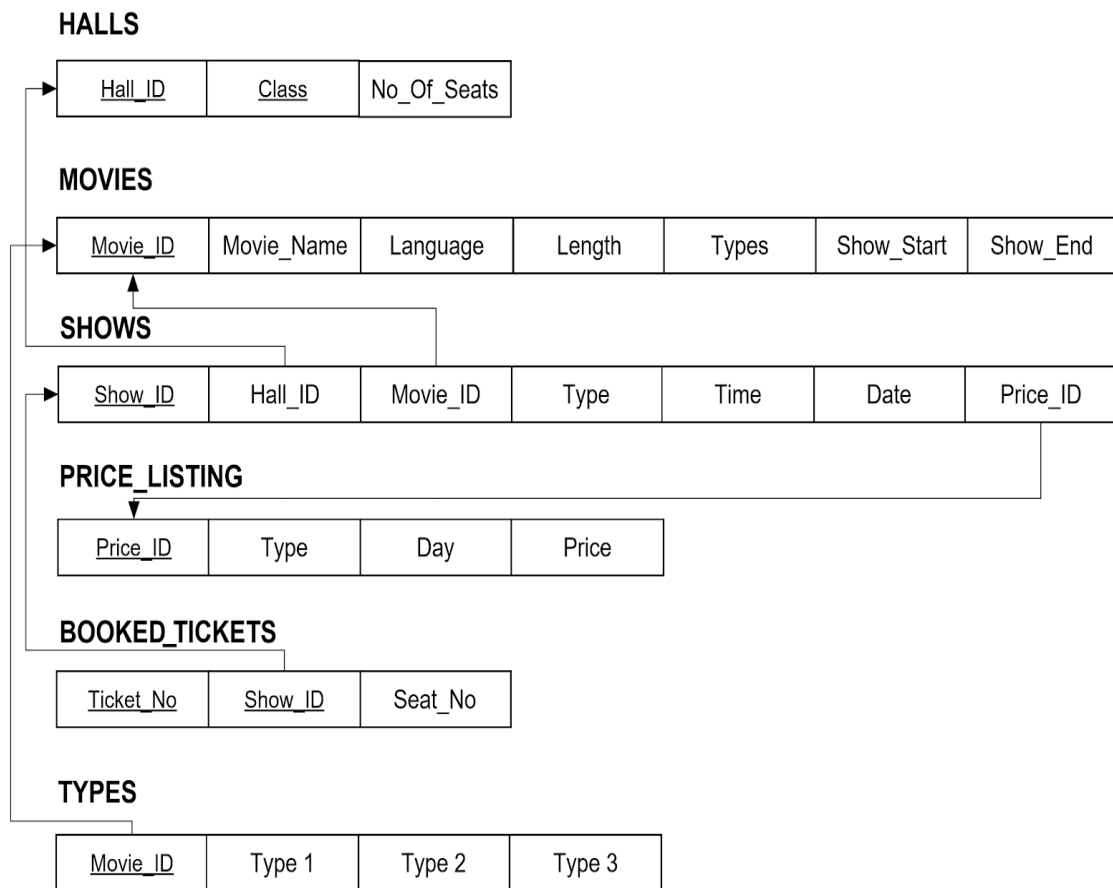
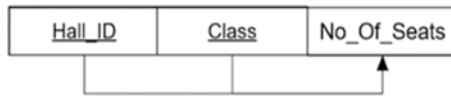


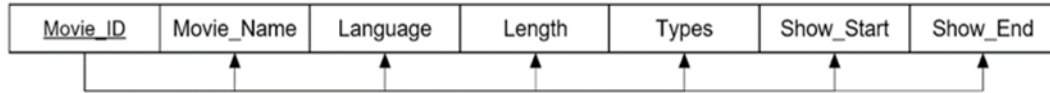
Diagram 4.4.D1: Schema Diagram

4.4. Relations in Third Normal Form(3NF):

HALLS



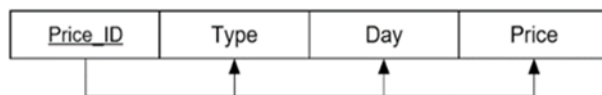
MOVIES



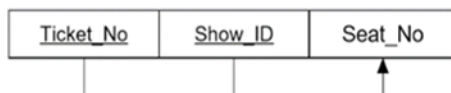
SHOWS



PRICE LISTS



BOOKED TICKETS



TYPES

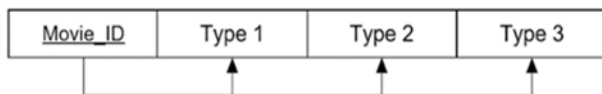


Diagram 3.3.D1: Functional Dependencies showing 3NF

4.5. Functional Dependencies:

- **HALLS** : FD: $Hall_ID, Class \rightarrow No_Of_Seats$
- **MOVIES** : FD: $Movie_ID \rightarrow Movie_Name, Language, Length, Types, Show_Start, Show_End$
- **SHOWS**: FD: $Show_ID \rightarrow Hall_ID, Movie_ID, Type, Time, Date, Price_ID$
- **PRICE LISTS**: FD: $Price_ID \rightarrow Type, Day, Price$

FD: $Type, Day \rightarrow Price$

(Type, Day forms a Super Key, hence it does not violate Third Normal Form)

- **BOOKED TICKETS**: FD: $Ticket_No, Show_ID \rightarrow Seat_No$
- **TYPES**: FD: $Movie_ID \rightarrow Type_1, Type_2, Type_3$

4.6. Justification Table:

| Entities Involved | Relationship | Description | Cardinality Ratio | Participation Ratio | Justification |
|--------------------------|---------------------|-------------------------------|--------------------------|----------------------------|----------------------------------------------------------------------------------------|
| Shows & Movies | Played at | Movies are played at shows. | N:1 | T:T | Every Show has Movies And Every movie is played at shows. |
| Shows & Price | costs | Every show costs price. | 1:N | T:T | All shows have price but all price doesn't have shows. |
| Shows & Halls | Plays in | Shows are played in halls. | N:1 | T:P | All shows are played at halls but all halls doesn't have shows. |
| Shows & Booked Tickets | booked | Tickets are booked for Shows. | 1:N | P:T | All tickets that are booked have shows but all shows need not have all tickets booked. |

5. IMPLEMENTATION:

5.1. Database table Structures:

Hall:

```
create table halls (  
    hall_id int,  
    class varchar(10),  
    no_of_seats int,  
    primary key(hall_id,class));
```

Movies:

```
create table movies (  
    movie_id int primary key,  
    movie_name varchar(40),  
    length int, language varchar(10),  
    show_start date,  
    show_end date);
```

Price listing:

```
create table price_listing (  
    price_id int primary key,  
    type varchar(3),  
    day varchar(10),  
    price int);
```

Shows:

```
create table shows (  
    show_id int primary key,  
    movie_id int,  
    hall_id int,  
    type varchar(3),  
    time int,  
    Date date,  
    price_id int,  
    foreign key(movie_id) references movies(movie_id),  
    foreign key(hall_id) references halls(hall_id),  
    foreign key(price_id) references price_listing(price_id) on update cascade);
```

Booked Tickets:

```
create table booked_tickets (  
    ticket_no int,  
    show_id int,  
    seat_no int,  
    primary key(ticket_no, show_id),  
    foreign key(show_id) references shows(show_id) on delete cascade);
```

Types:

```
create table types(  
    movie_id int primary key,  
    4DX varchar(3),  
    3D varchar(3),  
    2D varchar(3),  
    foreign key(movie_id) references movies(movie_id) on delete cascade);
```

5.2. Inserting values into the table:

Hall table:

```
insert into halls values(1, "gold", 35),  
    (1, "standard", 75),  
    (2, "gold", 27),  
    (2, "standard", 97),  
    (3, "gold", 26),  
    (3, "standard", 98);
```

Price Listing table:

```
insert into price_listing values(1, "2D", "Monday", 210),  
    (2, "3D", "Monday", 295),  
    (3, "4DX", "Monday", 380),  
    (4, "2D", "Tuesday", 210),  
    (5, "3D", "Tuesday", 295),  
    (6, "4DX", "Tuesday", 380),  
    (7, "2D", "Wednesday", 210),  
    (8, "3D", "Wednesday", 295),  
    (9, "4DX", "Wednesday", 380),  
    (10, "2D", "Thursday", 210),  
    (11, "3D", "Thursday", 295),  
    (12, "4DX", "Thursday", 380),  
    (13, "2D", "Friday", 320),  
    (14, "3D", "Friday", 335),  
    (15, "4DX", "Friday", 495),  
    (16, "2D", "Saturday", 320),
```

```
(17, "3D", "Saturday", 335),
(18, "4DX", "Saturday", 495),
(19, "2D", "Sunday", 320),
(20, "3D", "Sunday", 335),
(21, "4DX", "Sunday", 495);
```

Movies table:

insert into movies values

```
(1,"CAPTAIN AMERICA",2,"English","2020-12-20","2020-12-27"),
(2,"SHANG-CHI",2,"English","2020-01-15","2020-01-22"),
(3,"KGF-2",3,"Kannada","2020-03-05","2020-03-12"),
(4,"MAJOR",2,"Hindi","2020-05-27","2020-06-02"),
(5,"CHARLIE777",3,"Kannada","2020-07-13","2020-07-20");
```

Show table:

```
insert into shows values(11,1,2,"3D",12,"2020-01-15",2),
(12,3,3,"2D",3,"2020-03-05",4),
(13,2,1,"4DX",6,"2020-12-20",12);
```

booked_tickets table:

```
insert into booked_tickets values(1001,11,35), (1002,11,36),
(1003,11,24),
(1004,11,30),
(2001,12,10),
(2002,12,11),
(2003,12,20),
(3001,13,07),
(3002,13,19),
(3003,13,20);
```


Types table:

```
insert into types values(1,"yes","yes","no"),
    (2,"no","yes","yes"),
    (3,"no","no","yes"),
    (4,"no","no","yes"),
    (5,"no","yes","yes");
```

5.3. Show Tables:

Show tables;

```
+-----+
| Tables_in_moviebooking |
+-----+
|      booked_tickets    |
|           halls        |
|           movies       |
|      price_listing     |
|           shows        |
|           types        |
+-----+
```

Halls Table:

Select * from halls;

```
+-----+-----+-----+
| hall_id | class | no_of_seats |
+-----+-----+-----+
| 1       | gold  | 35          |
| 1       | standard | 75          |
| 2       | gold  | 27          |
| 2       | standard | 97          |
| 3       | gold  | 26          |
| 3       | standard | 98          |
+-----+-----+-----+
```

Price Listing Table:

Select * from price_listing;

| price_id | type | day | price |
|----------|------|-----------|-------|
| 1 | 2D | Monday | 210 |
| 2 | 3D | Monday | 295 |
| 3 | 4DX | Monday | 380 |
| 4 | 2D | Tuesday | 210 |
| 5 | 3D | Tuesday | 295 |
| 6 | 4DX | Tuesday | 380 |
| 7 | 2D | Wednesday | 210 |
| 8 | 3D | Wednesday | 295 |
| 9 | 4DX | Wednesday | 380 |
| 10 | 2D | Thursday | 210 |
| 11 | 3D | Thursday | 295 |
| 12 | 4DX | Thursday | 380 |
| 13 | 2D | Friday | 320 |
| 14 | 3D | Friday | 335 |
| 15 | 4DX | Friday | 495 |
| 16 | 2D | Saturday | 320 |
| 17 | 3D | Saturday | 335 |
| 18 | 4DX | Saturday | 495 |
| 19 | 2D | Sunday | 320 |
| 20 | 3D | Sunday | 335 |
| 21 | 4DX | Sunday | 495 |

Movies Table:

Select * from movies;

| movie_id | movie_name | length | language | show_start | show_end |
|----------|-----------------|--------|----------|------------|------------|
| 1 | CAPTAIN AMERICA | 2 | English | 2020-12-20 | 2020-12-27 |
| 2 | SHANG-CHI | 2 | English | 2020-01-15 | 2020-01-22 |
| 3 | KGF-2 | 3 | Kannada | 2020-03-05 | 2020-03-12 |
| 4 | MAJOR | 2 | Hindi | 2020-05-27 | 2020-06-02 |
| 5 | CHARLIE777 | 3 | Kannada | 2020-07-13 | 2020-07-20 |

Booked Tickets table:

select *from booked_tickets;

| ticket_no | show_id | seat_no |
|-----------|---------|---------|
| 1001 | 11 | 35 |
| 1002 | 11 | 36 |
| 1003 | 11 | 24 |
| 1004 | 11 | 30 |
| 2001 | 12 | 10 |
| 2002 | 12 | 11 |
| 2003 | 12 | 20 |
| 3001 | 13 | 7 |
| 3002 | 13 | 19 |
| 3003 | 13 | 20 |

Shows Table:

Select * from shows;

| show_id | movie_id | hall_id | type | time | Date | price_id |
|---------|----------|---------|------|------|------------|----------|
| 11 | 1 | 2 | 3D | 12 | 2020-01-15 | 2 |
| 12 | 3 | 3 | 2D | 3 | 2020-03-05 | 4 |
| 13 | 2 | 1 | 4DX | 6 | 2020-12-20 | 12 |

Types Table:

select *from types;

| movie_id | 4DX | 3D | 2D |
|----------|-----|-----|-----|
| 1 | yes | yes | no |
| 2 | no | yes | yes |
| 3 | no | no | yes |
| 4 | no | no | yes |
| 5 | no | yes | yes |

5.4. Queries:

1: Show the seat no for the ticket bearing the number 1004.

→select seat_no , ticket_no from booked_tickets where ticket_no=1004;

```
+-----+-----+
| seat_no | ticket_no |
+-----+-----+
|    30   |    1004   |
+-----+-----+
```

2: Show all the price details for different type of shows that are screening on Wednesday.

→select type, day, price from price_listing where day="Wednesday";

```
+-----+-----+-----+
| type |    day    | price |
+-----+-----+-----+
| 2D   | Wednesday |  210  |
| 3D   | Wednesday |  295  |
| 4DX  | Wednesday |  380  |
+-----+-----+-----+
```

3: Show the length and the schedule for the movie kgf3 and also the show id.

→Select length, show_start, show_end,show_id from movies,shows where movie_name="KGF-2" and shows.movie_id=movies.movie_id;

```
+-----+-----+-----+-----+
| length | show_start | show_end | show_id |
+-----+-----+-----+-----+
|    3   | 2020-03-05 | 2020-03-12 |    12   |
+-----+-----+-----+-----+
```

6. RESULTS AND SCREENSHOTS:

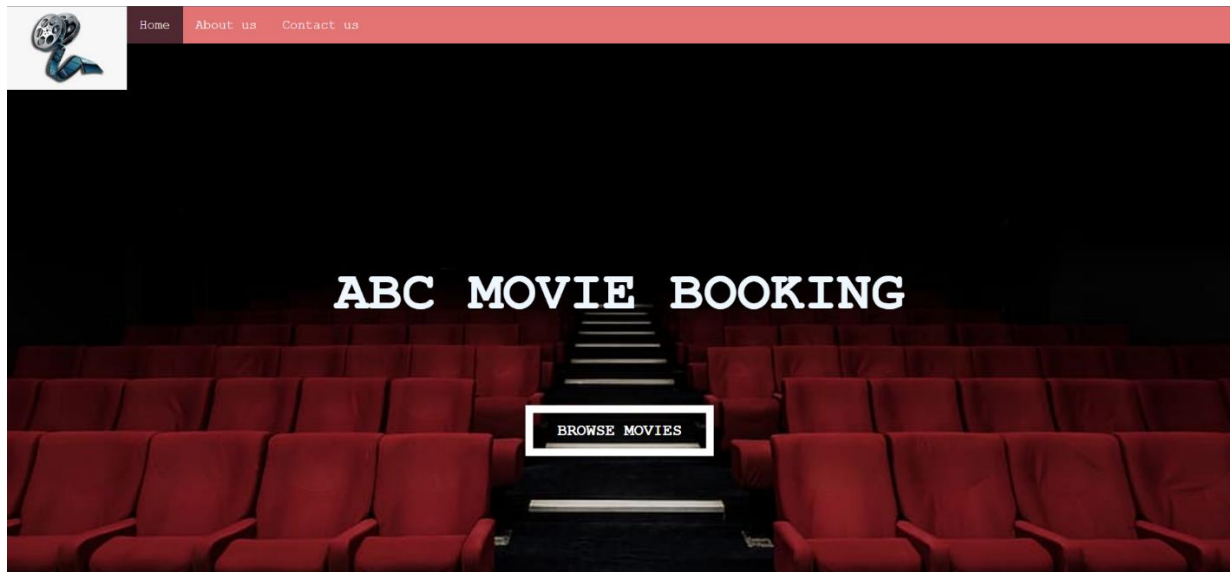


Diagram 5.D1: Home Page

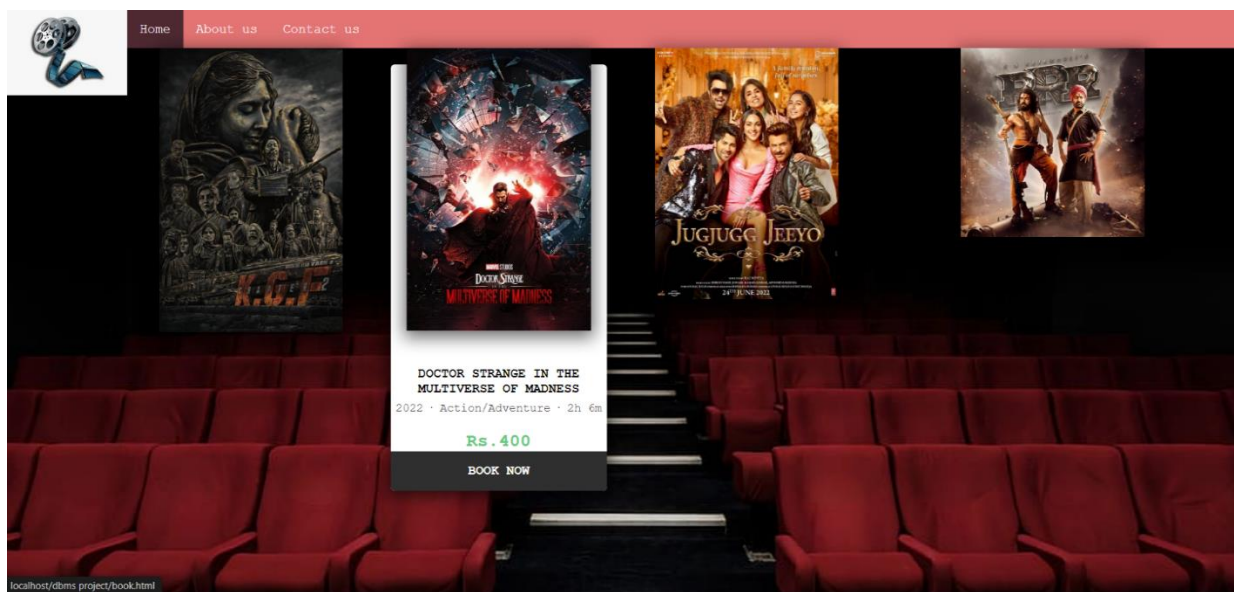
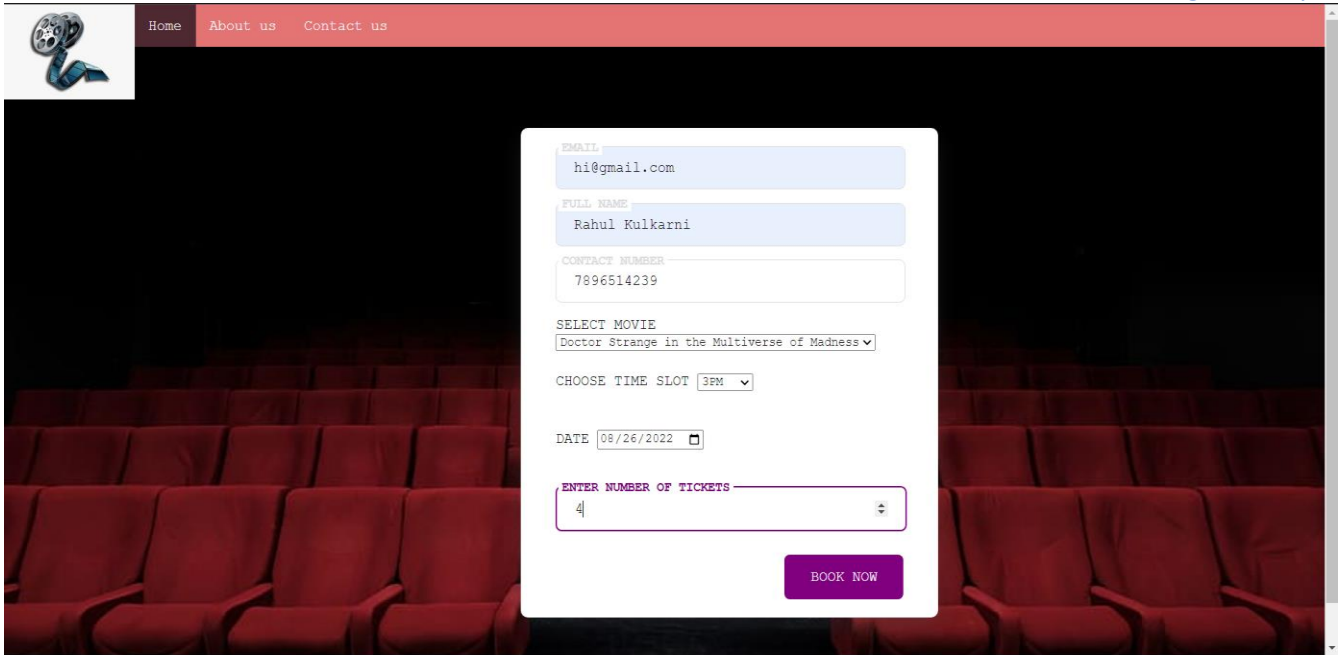


Diagram 5.D2: Browse Movies page



The screenshot shows a web application for a theatre management system. The background is a dark image of a theatre interior with rows of red seats. A white navigation bar at the top contains a film reel icon and links for 'Home', 'About us', and 'Contact us'. A central white form is used for booking tickets. It includes input fields for 'EMAIL' (hi@gmail.com), 'FULL NAME' (Rahul Kulkarni), and 'CONTACT NUMBER' (7896514239). There are dropdown menus for 'SELECT MOVIE' (Doctor Strange in the Multiverse of Madness) and 'CHOOSE TIME SLOT' (3PM). A date picker shows '08/26/2022'. A field for 'ENTER NUMBER OF TICKETS' has the value '4'. A purple 'BOOK NOW' button is at the bottom right of the form.

EMAIL
hi@gmail.com

FULL NAME
Rahul Kulkarni

CONTACT NUMBER
7896514239

SELECT MOVIE
Doctor Strange in the Multiverse of Madness

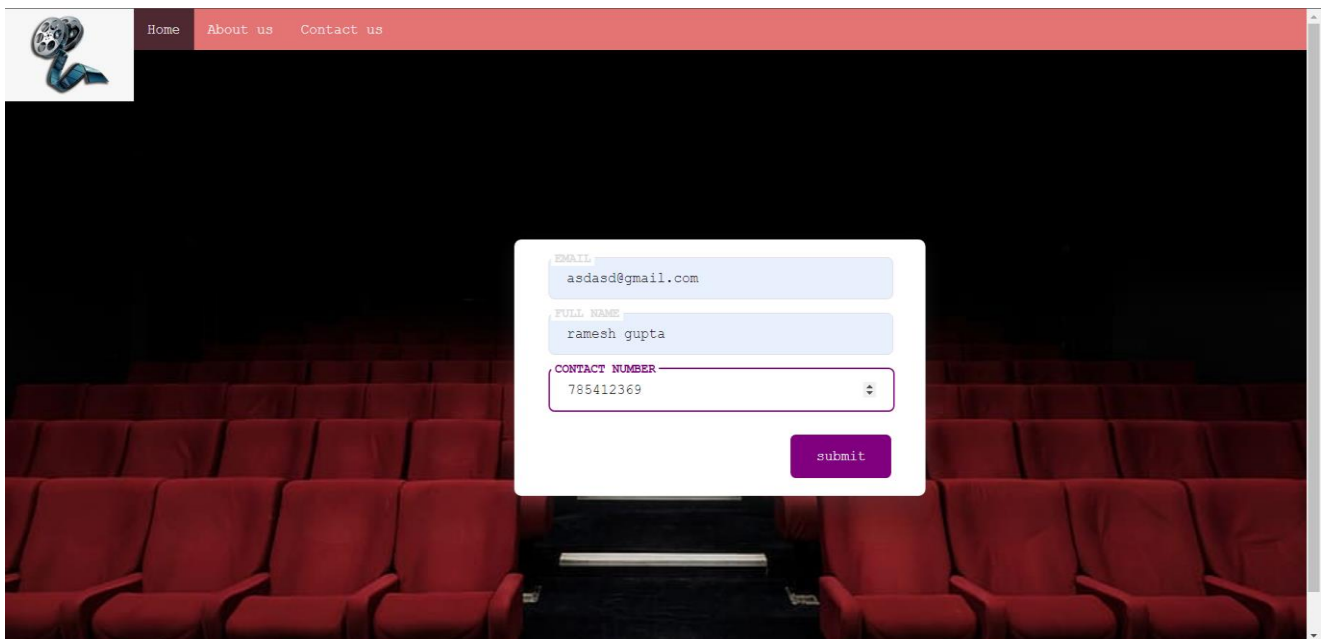
CHOOSE TIME SLOT
3PM

DATE
08/26/2022

ENTER NUMBER OF TICKETS
4

BOOK NOW

Diagram 5.D3: Book tickets page



The screenshot shows the 'Contact Us' page of the theatre management system. The background is the same theatre interior with red seats. The white navigation bar is identical. A central white form is used for contact. It includes input fields for 'EMAIL' (asdasd@gmail.com), 'FULL NAME' (ramesh gupta), and 'CONTACT NUMBER' (785412369). A purple 'submit' button is at the bottom right of the form.

EMAIL
asdasd@gmail.com

FULL NAME
ramesh gupta

CONTACT NUMBER
785412369

submit

Diagram 5.D4: Contact Us page

Server: 127.0.0.1 » Database: dbms_theatre » Table: contact

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 3 (4 total, Query took 0.0011 seconds)

```
SELECT * FROM `contact`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

| EMAIL | FNAME | PHONE |
|--------------------------|--------------------|------------|
| super@gmail.com | john | 2147483647 |
| asdasd@gmail.com | ramesh gupta | 785412369 |
| omkardeshpande@gmail.com | omkar deshpande | 789654133 |
| prathamesh123@gmail.com | prathamesh badiger | 2147483647 |

Show all | Number of rows: 25 | Filter rows: Search this table

Query results operations

Print Copy to clipboard Export Display chart Create view

Console

Diagram 5.D5: Database for Contact Us table

Server: 127.0.0.1 » Database: dbms_theatre » Table: booking

Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

Showing rows 0 - 3 (4 total, Query took 0.0005 seconds)

```
SELECT * FROM `booking`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

| EMAIL | NAME | CONTACT_NO | MOVIE_NAME | TIME | DATE | NO_OF_TICKETS |
|----------------------------|-----------------|------------|-------------------------------|------|------------|---------------|
| asdasd@gmail.com | ramesh gupta | 2147483647 | Doctor Strange in the Multive | 6PM | 2022-08-27 | 5 |
| kulkarnirahul349@gmail.com | Rahul Kulkarni | 2147483647 | Jugjugg Jeeyo | 3PM | 2022-08-25 | 4 |
| hi@gmail.com | Rahul Kulkarni | 2147483647 | Doctor Strange in the Multive | 3PM | 2022-08-26 | 4 |
| omkardeshpande@gmail.com | omkar deshpande | 47896315 | KGF chapter-2 | 6PM | 2022-08-27 | 5 |

Show all | Number of rows: 25 | Filter rows: Search this table

Query results operations

Print Copy to clipboard Export Display chart Create view

Console

Diagram 5.D6: Database for booking tickets table

7. CONCLUSION:

Our database model successfully gives information about various halls currently showing movies, show timings, booked tickets and price listings in the theatre as well as the associated attributes. It is possible for the management of the theatre to use the database to fully understand all parts of the system without using additional software.

8. REFERENCES:

- www.geeksforgeeks.com
- www.javatpoint.com
- www.tutorialspoint.com
- www.youtube.com