Movie Management System

Description

This system will provide movie details about any movie that the user wants to know about. These details include Cast details, Director, Producer details, Genre of the movie, etc. Our database system will allow the user to search movie based on filter like movie name, producer, director name, cast name. It also allows the user to check the movies available in nearby theatres based on user's zip code. Following this, the user can also check show timings for the movie on a particular day (from a drop-down menu).

Entity Sets

1. Movie
2. Casts
3. Director
4. Producer
5. Viewer
6. Award
7. Genre
8. Theater
9. Show
10. StreamingPlatform
11. Revenue

Attributes

1. Movie (Id (primary key), Name (not null), Language, Country, Release_date (not null), Description, Length (not null))
2. Casts (Id (primary key), Name (not null), Age, Gender)
3. Director (Id (primary key), Name (not null), Age, Gender)
4. Producer (Id (primary key), Name (not null), Age, Gender)
5. Viewer (Id (primary key), Name (not null), Age, Gender)
6. Award (Id (primary key), Type)
7. Genre (Id (primary key), type)
8. Theater (Id (primary key), Name (not null), ZIP_Code)
9. Show (Id (partial key), S_Date(not null), S_Time(not null))
10. Revenue (Id (primary key), TotalExpenses, TotalEarnings, CollectedRevenue(derived))
11. StreamingPlatform (Id (primary key), Name)

Relationship Sets

1. viewed_rated_by (Movies are viewed and rated by Viewers)
2. produced_by (Movies are produced by Producers)
3. directed_by (Movies are directed by Directors)
4. starred_by (Movies are starred by casts)
5. receive (Movies receive Awards)
6. belong_to (Movies belong to Genres)

Movie Management System

7. played_at (Movies are played in Theaters)
8. display(Theater display shows)
9. generate(Movies generate revenue)
10. streamed_on(Movies are streamed on streaming platforms)

Business Rules-

MOVIES

- Movies are directed by Directors. Each Movie is directed by at least one Director. (participation constraint)
- Movies are rated by viewers. Each movie may or may not be rated by any viewer.
- Movies are viewed by viewers. Each movie may or may not be viewed by any viewer.
- Movies are played in Theaters. Each Movie may or may not be played in any Theater.
- Movies are streamed on streaming platforms. A movie may or may not be streamed on any platform.
- Movies are produced by Producers. Each Movie is produced by at least one Producer. (participation constraint)
- Movies are starred by casts. Each Movie should have at least one cast member. (participation constraint)
- Movies belong to Genre. Each Movie belongs to at least one Genre. (participation constraint)
- Movies receive Awards. A Movie may or may not receive an award.

VIEWERS

- A viewer may or may not view or rate any movies.

DIRECTORS

- Directors direct movies.
- A director may or may not have directed any movie.

PRODUCERS

- Producers produce movies.
- A producer may or may not have produced any movie.

THEATERS

- Theaters display shows. A theater has one or more shows. (participation constraint)

STREAMING PLATFORM

- A streaming platform streams at least one movie.

Movie Management System

CASTS

- Casts are starred in movies.
- Each cast has been starred in at least one movie.

AWARD

- Awards can be awarded to Movies.
- Each award can be awarded to at least one movie. (participation constraint)

GENRE

- There should be at least one movie for a particular genre. (participation constraint)

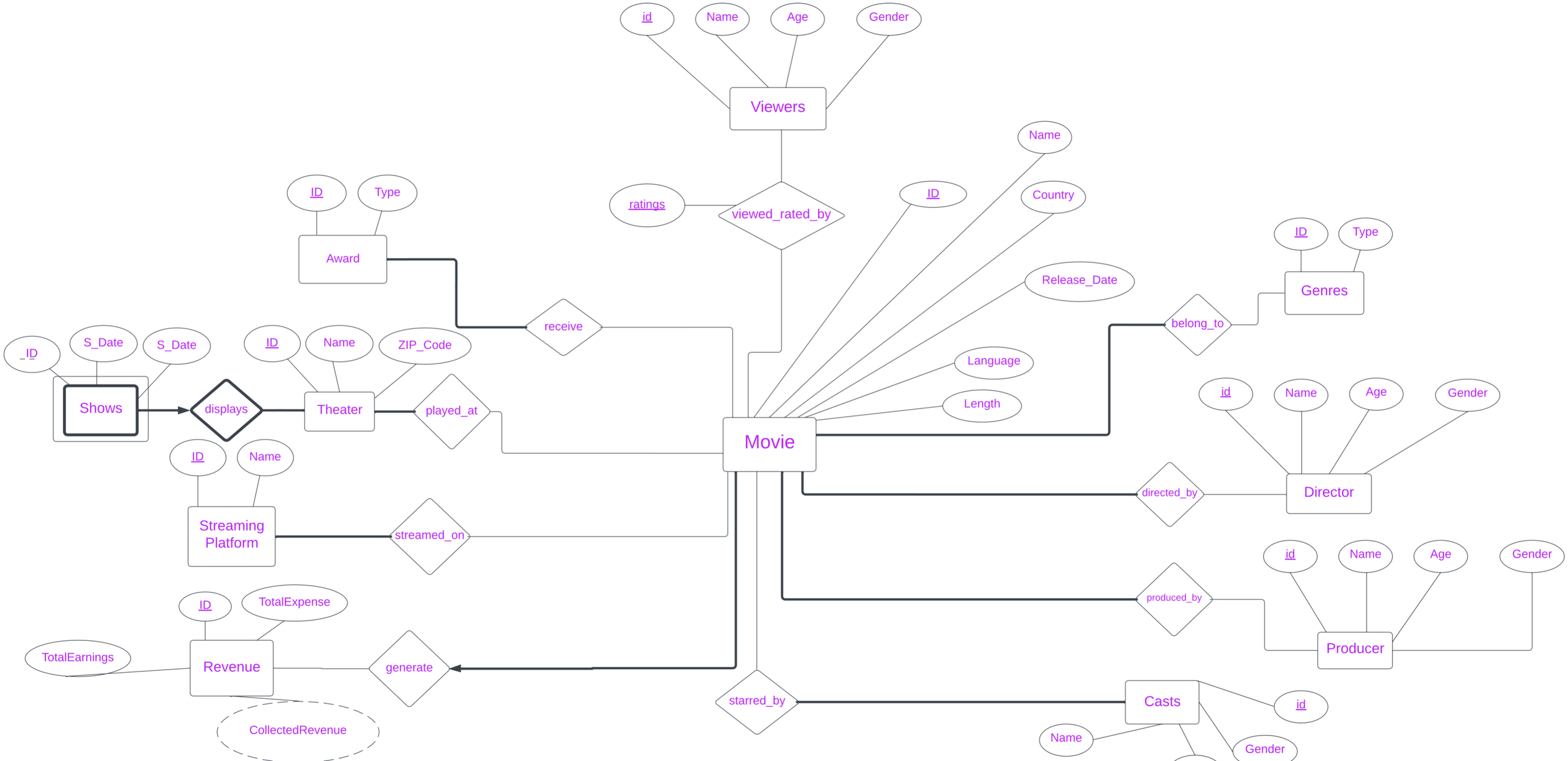
SHOW

- A show is displayed in the theater. (Weak Entity)

Data for our Application

- We will populate our database by manual entry of tuples in each table.

ER Diagram


id
Name
Age
Gender
Viewers
ratings
viewed_rated_by
ID
Type
Award
receive
Name
ID
Country
Release_Date
ID
Type
Genres
belong_to
Language
Length
_ID
S_Date
S_Date
Shows
displays
ID
Name
ZIP_Code
Theater
played_at
Movie
id
Name
Age
Gender
Director
directed_by
ID
Name
Streaming Platform
streamed_on
id
Name
Age
Gender
Producer
produced_by
ID
TotalExpense
TotalEarnings
Revenue
generate
CollectedRevenue
starred_by
Casts
id
Name
Gender

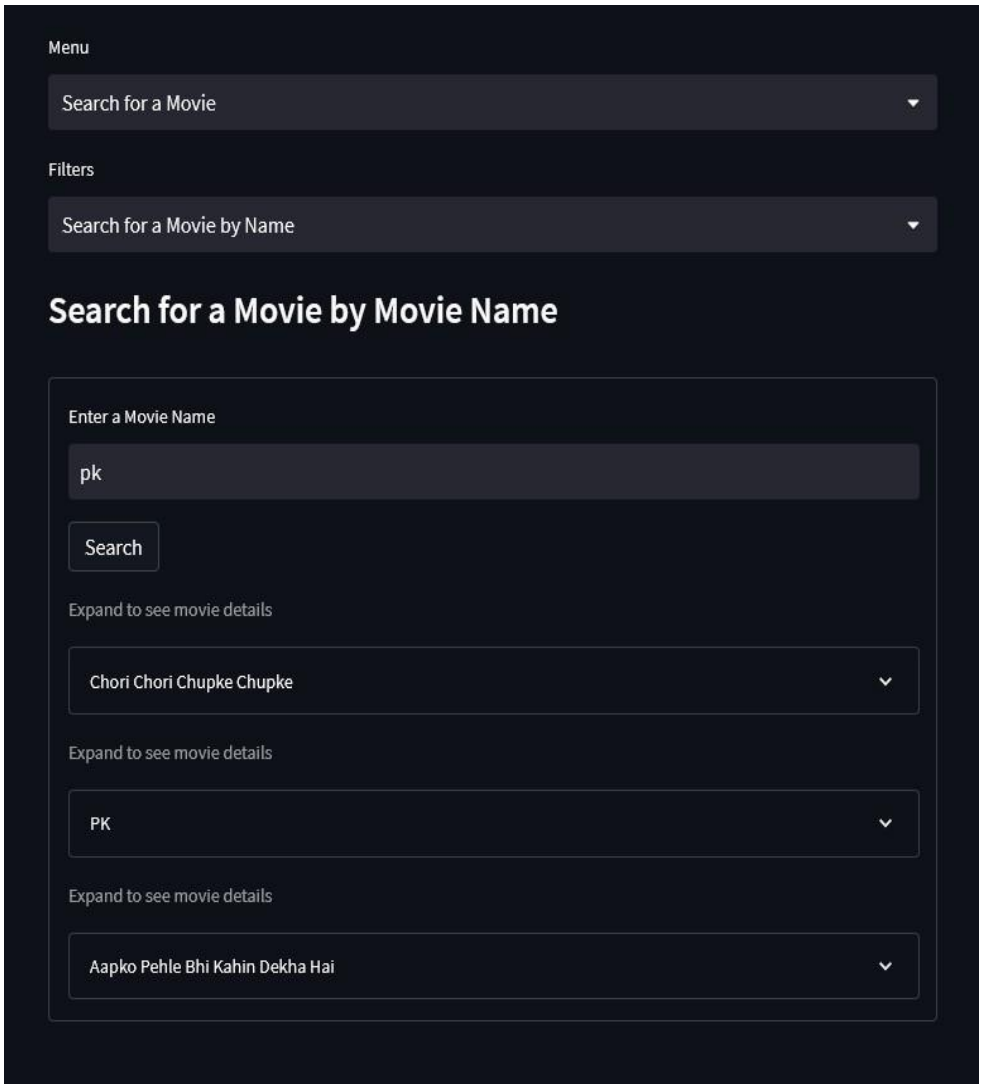
Movie Management System

How will the user interact with our database?

Our primary focus is to build a user-friendly database system that will enable users to easily utilize various functionalities that we are going to incorporate in the project. There are various ways in which a user could interact with our system, some of which are listed as follows:

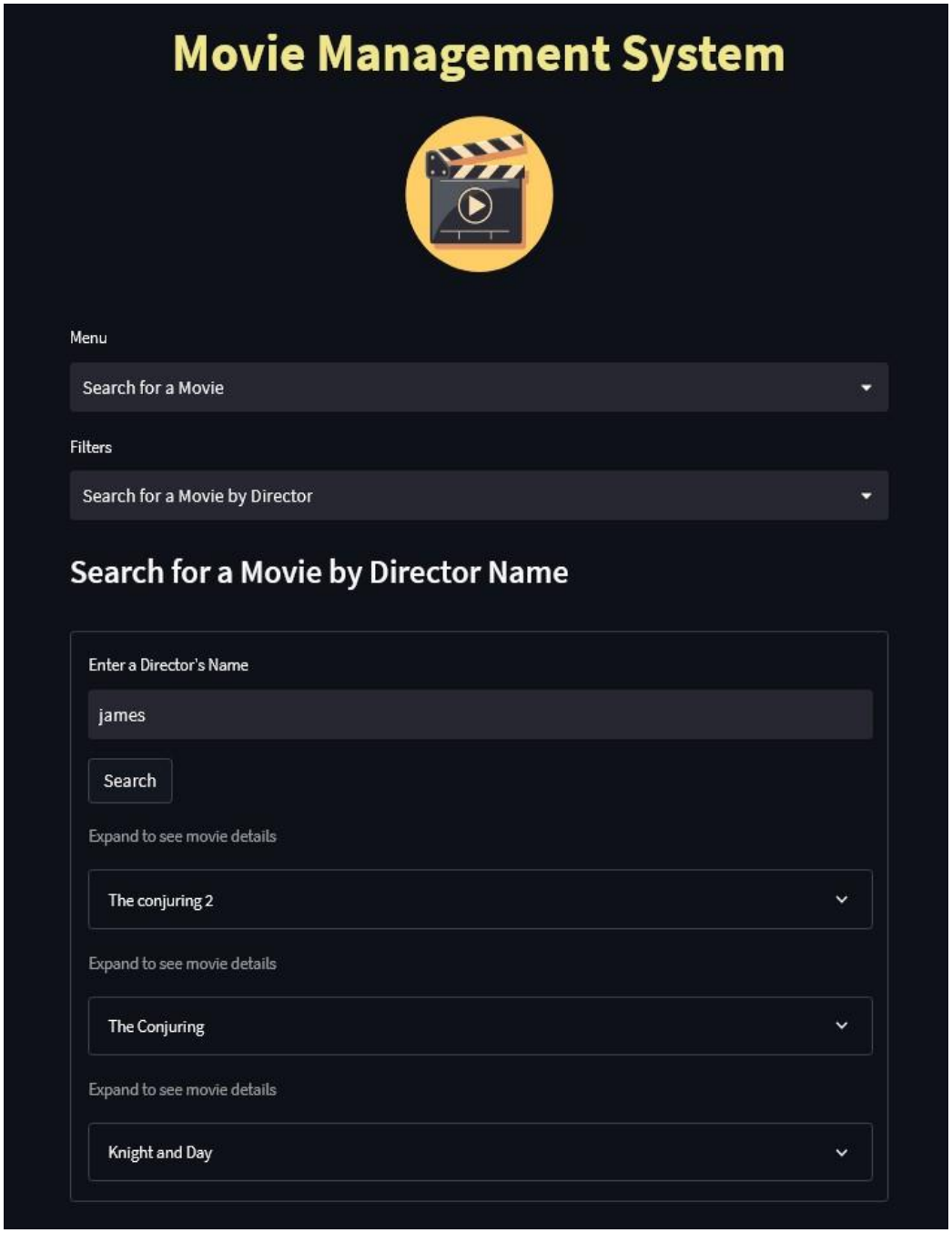
1) **<u>Search for a Movie:</u>**

   a) **Search by Movie name**: A user can find details about the movie by searching for it in the search box as shown in the screenshot below. The user need not put the full name of the movie, instead just a subpart of the name will also work here. For example, if we search pk, the output contains Chori Chori Chu**pk**e Chu**pk**e, **PK**, Aa**pk**o Pehle Bhi Kahin Dekha Hai.



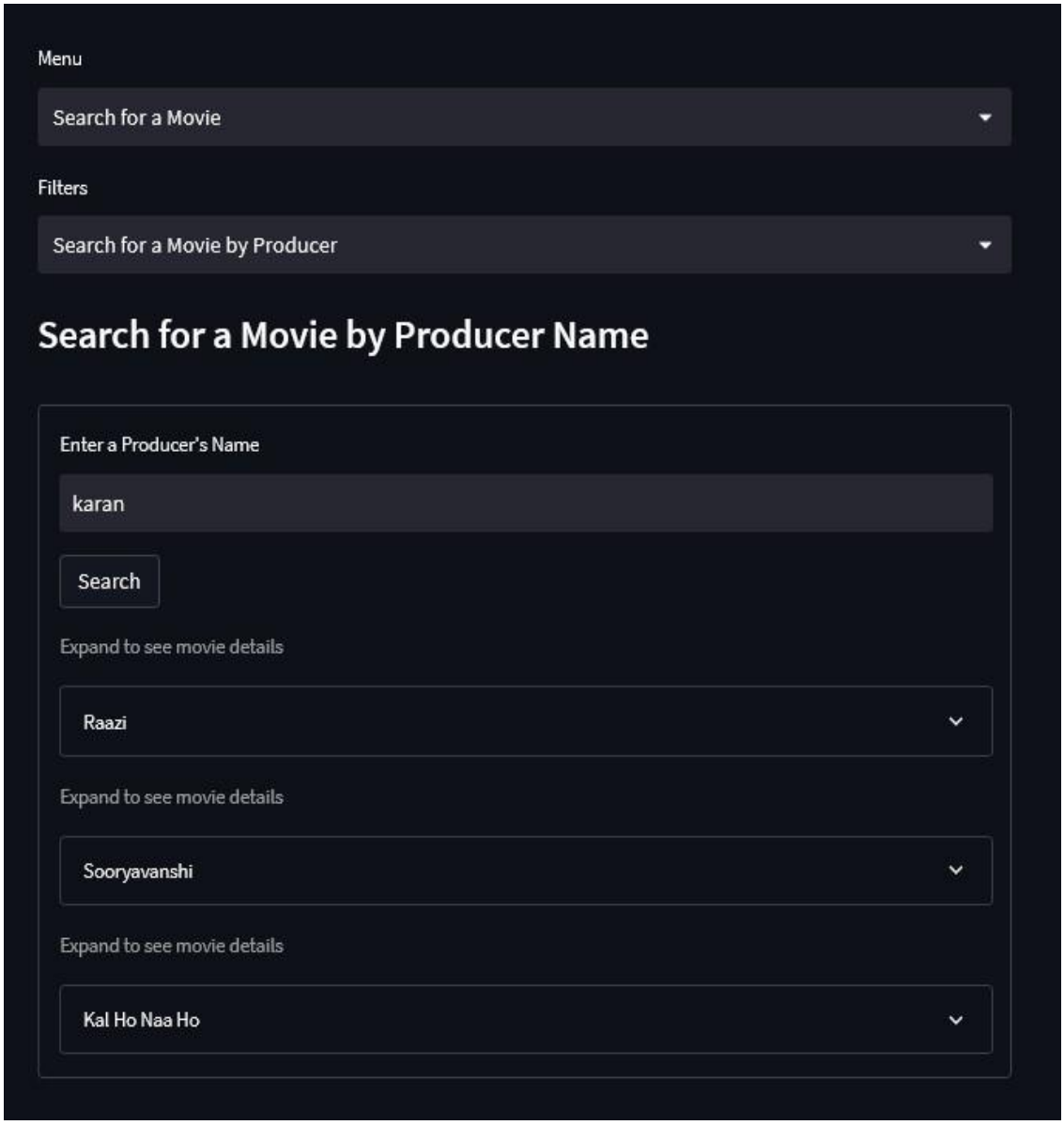
Movie Management System

b) **Search by Director name:**
The user can also find out details about the movie by searching for a particular director name. In the screenshot below, all 3 movies have director name as **James** Wan, **James** Mangold.



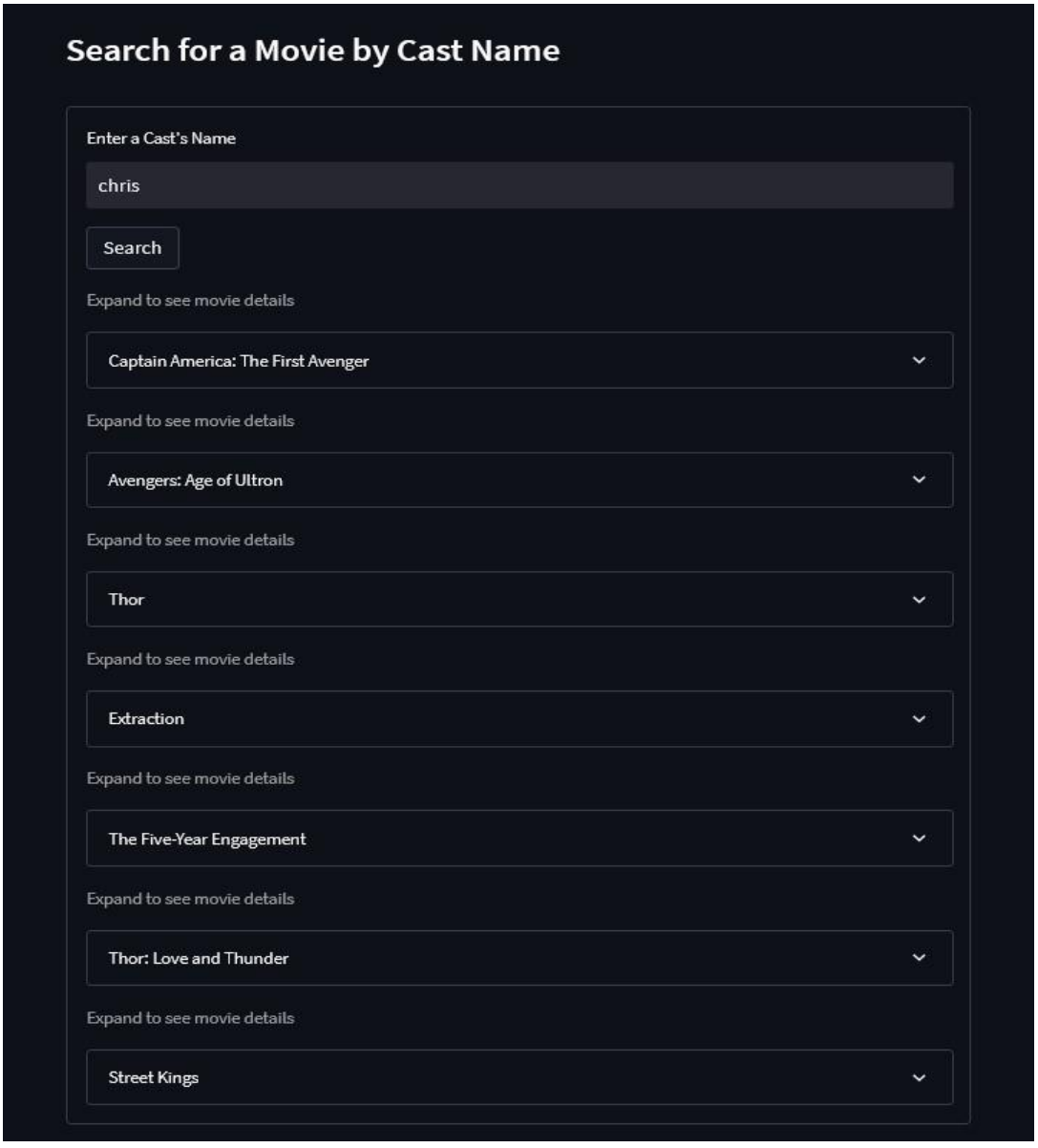
Movie Management System

c) **Search by Producer name:**
The user can also find out details about the movie by searching for a particular producer name. In the screenshot below, all movies have producer name as **Karan** Johar.



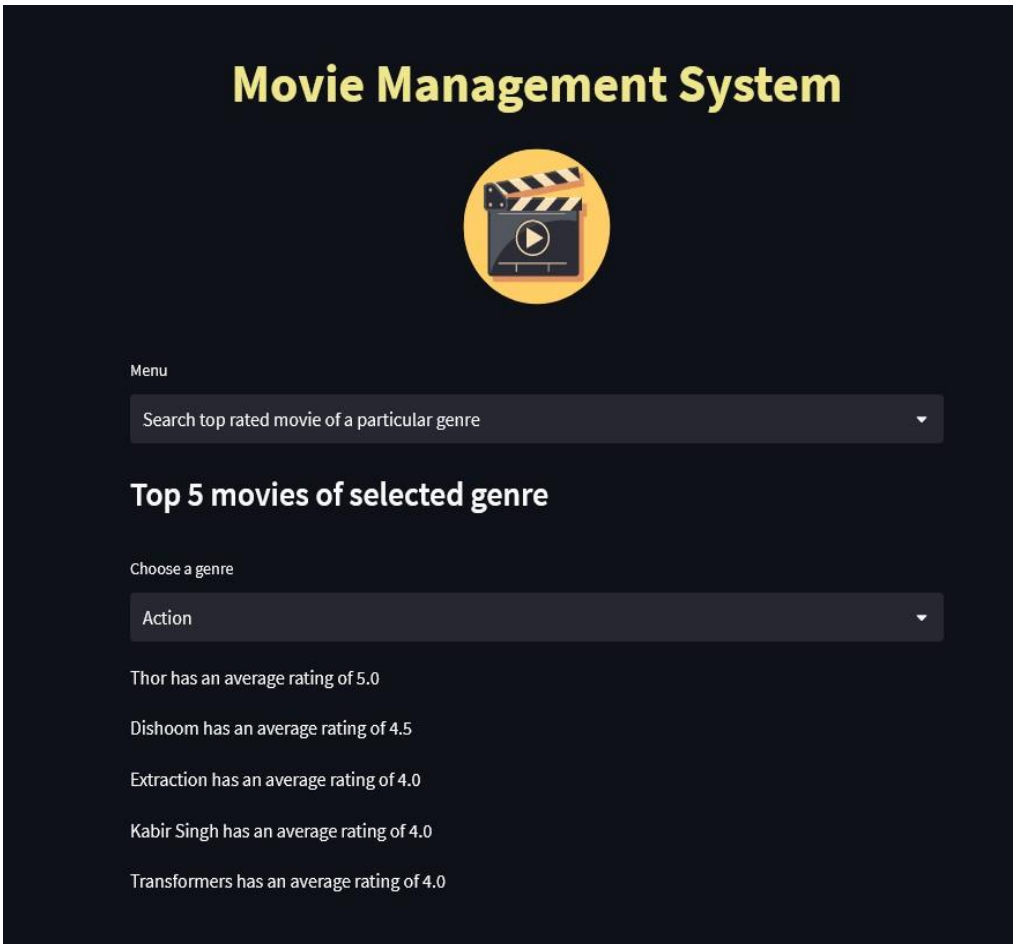
Movie Management System

d) **Search by Cast name:**
The user can also find out details about the movie by searching for a particular cast name. In the screenshot below, all movies have cast name as **Chris** Evans, **Chris** Hemsworth.



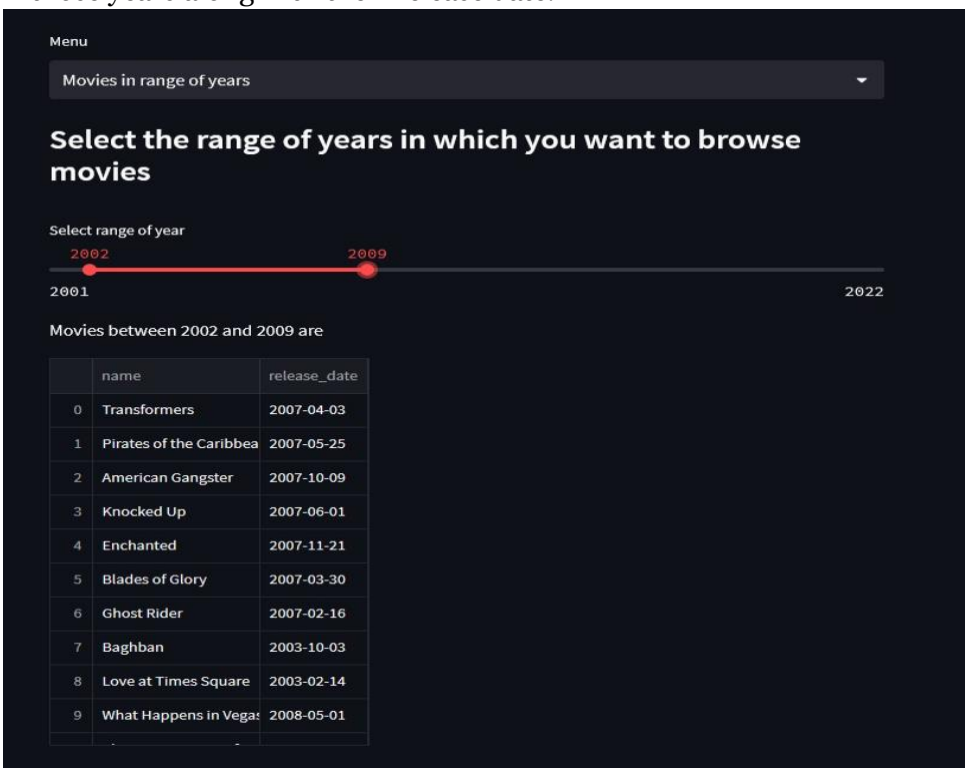
2) Search top rated movie of a particular genre:
A user can simply select a genre from the list and it will display top rated movies of that particular genre. This way, the user can get interesting movie recommendations based on genre.

Movie Management System



3) Movies in range of years:

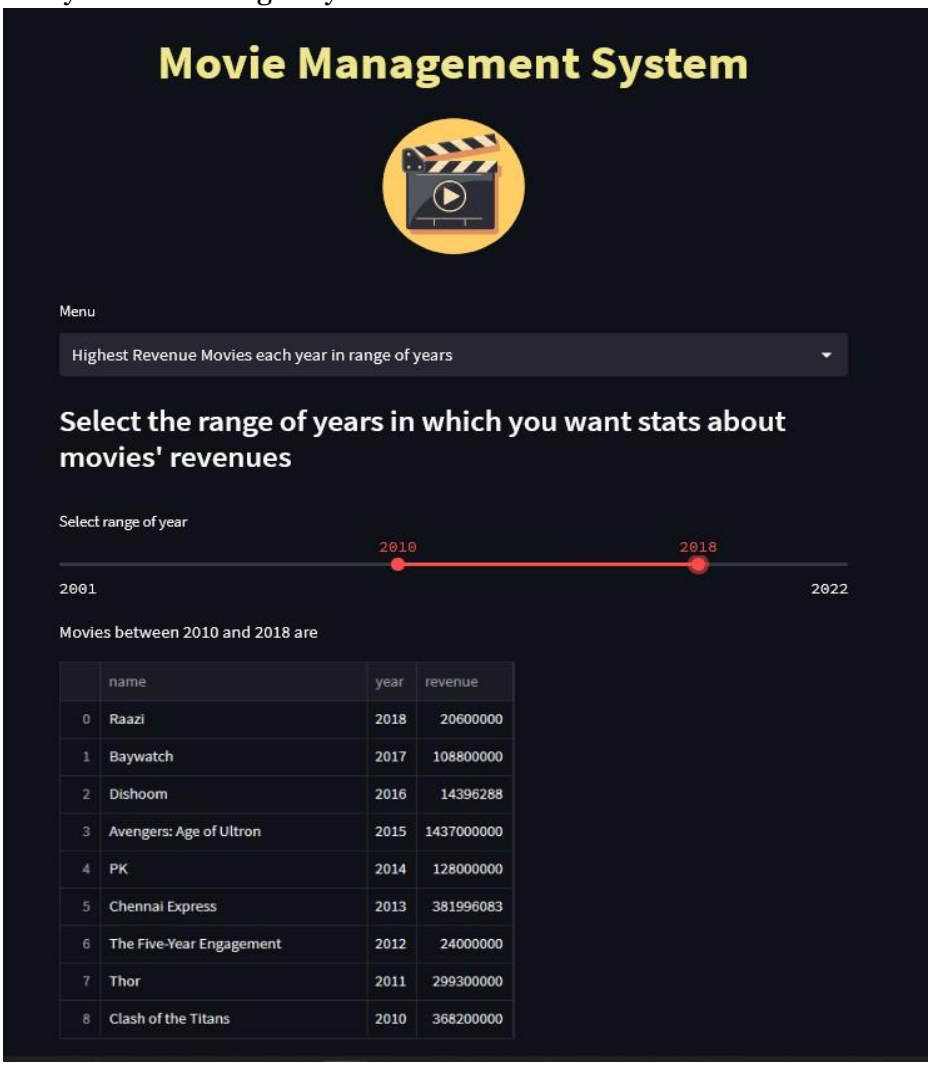
A user can select range of years using a slider. The output will be list of names movies in those years along with their release date.



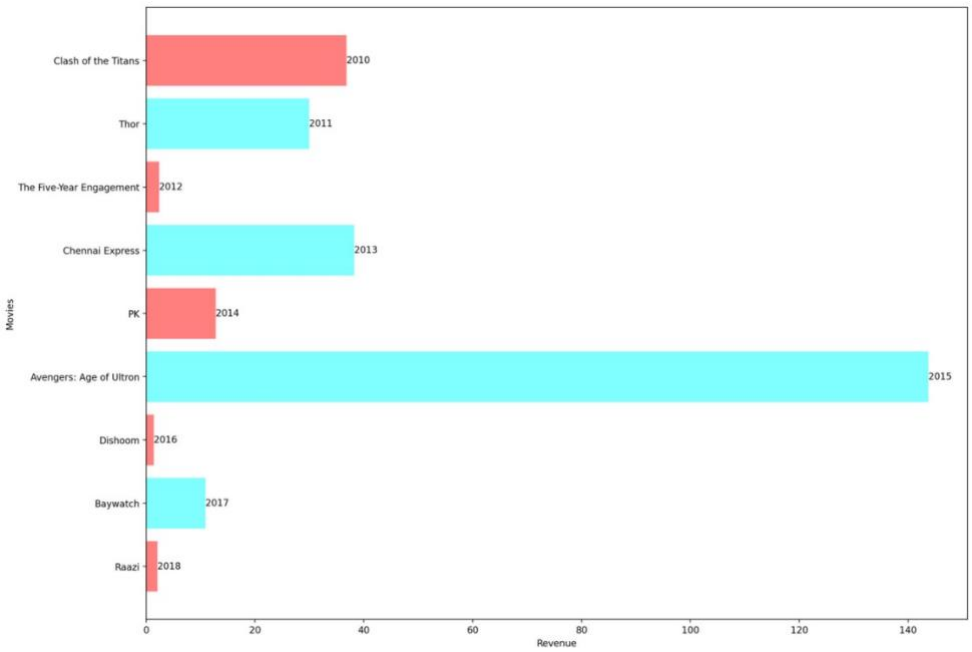
	name	release_date
0	Transformers	2007-04-03
1	Pirates of the Caribbea	2007-05-25
2	American Gangster	2007-10-09
3	Knocked Up	2007-06-01
4	Enchanted	2007-11-21
5	Blades of Glory	2007-03-30
6	Ghost Rider	2007-02-16
7	Baghban	2003-10-03
8	Love at Times Square	2003-02-14
9	What Happens in Vegas	2008-05-01

Movie Management System

4) **Highest Revenue Movies each year in range of years:**
A user can select a range of years and can compare highest grossing movies in each year over a range of years.



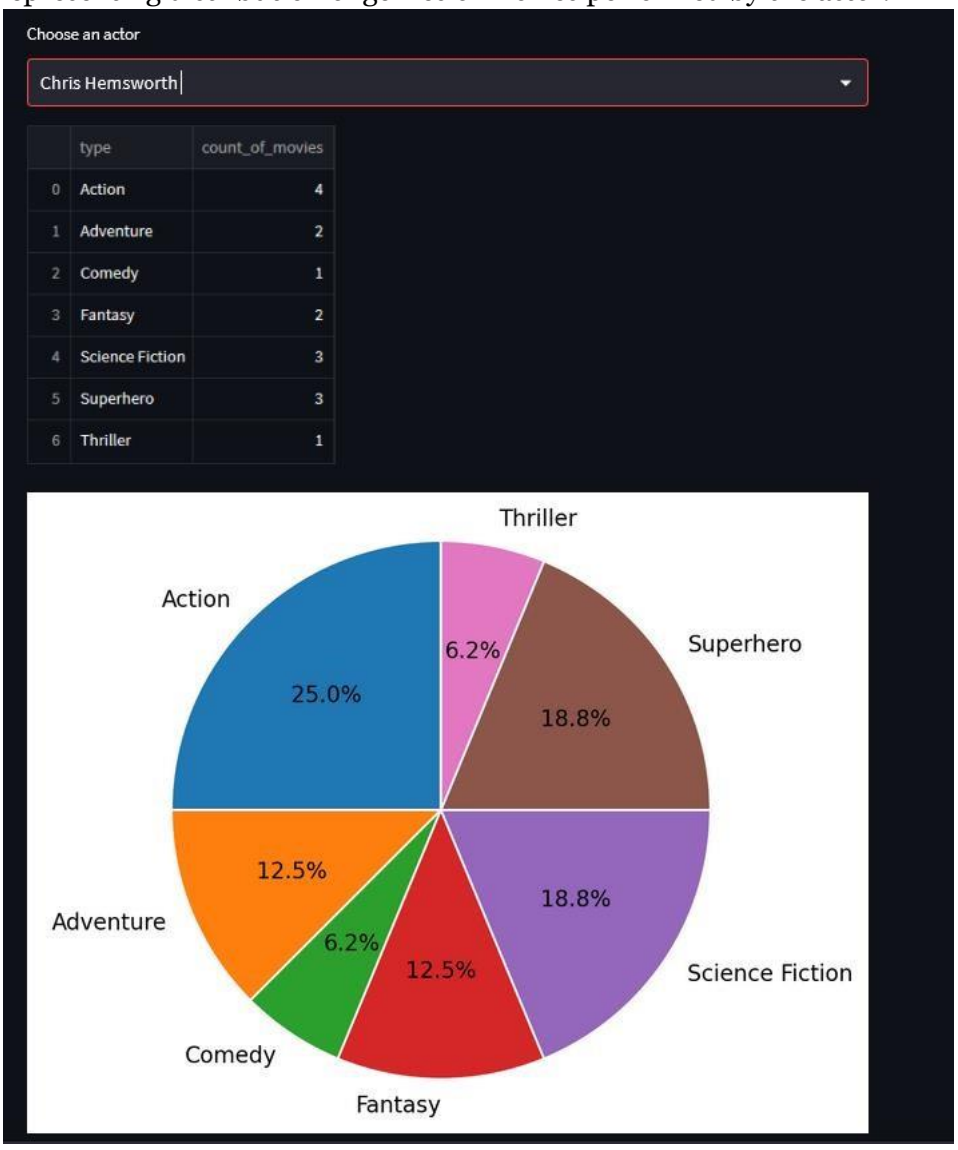
	name	year	revenue
0	Raazi	2018	20600000
1	Baywatch	2017	108800000
2	Dishoom	2016	14396288
3	Avengers: Age of Ultron	2015	1437000000
4	PK	2014	128000000
5	Chennai Express	2013	381996083
6	The Five-Year Engagement	2012	24000000
7	Thor	2011	299300000
8	Clash of the Titans	2010	368200000



Movie Management System

5) **Distribution of Genres by Actor:**

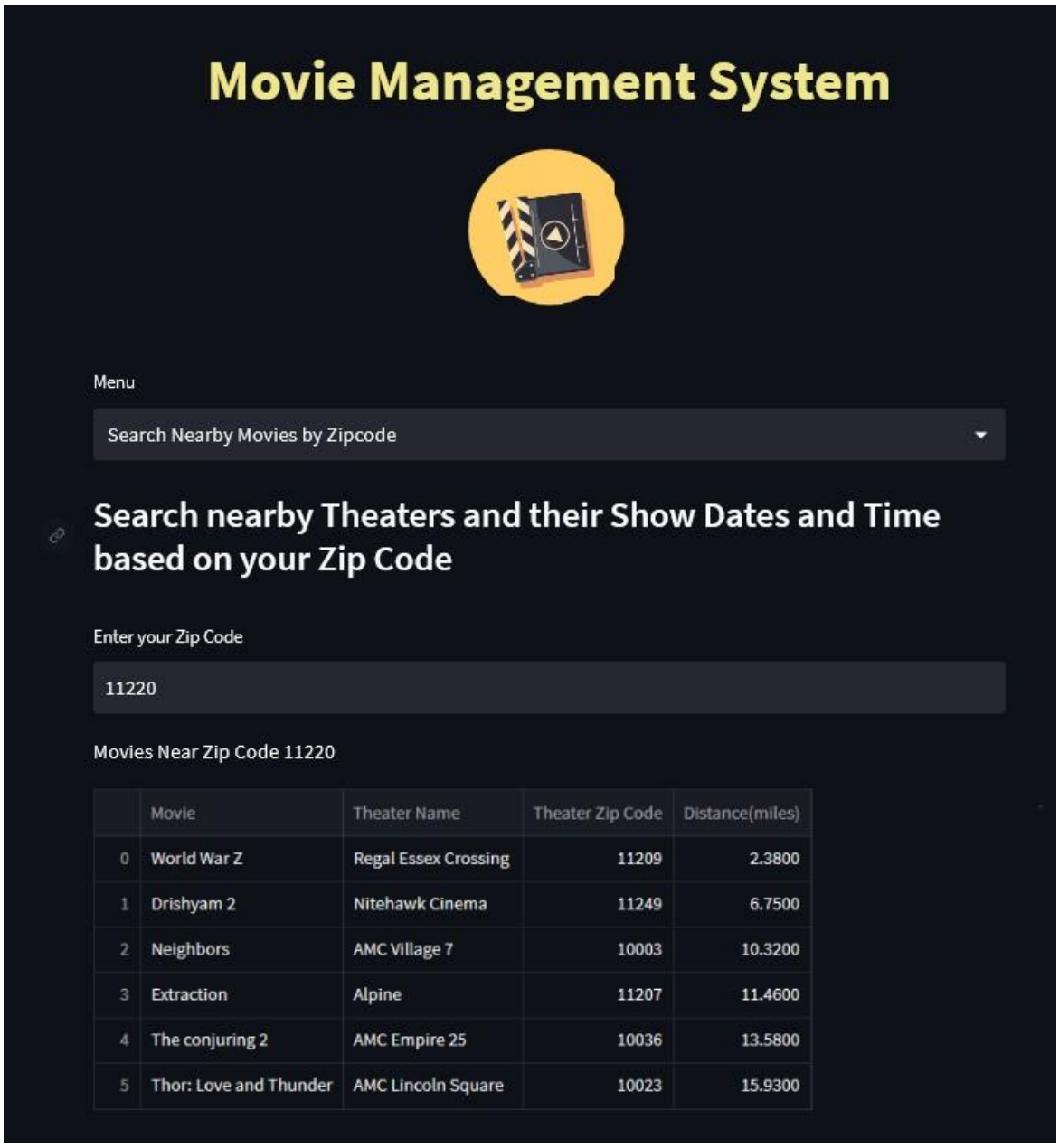
A user can select an actor from the dropdown menu. The output is a pie chart representing distribution of genres of movies performed by the actor.



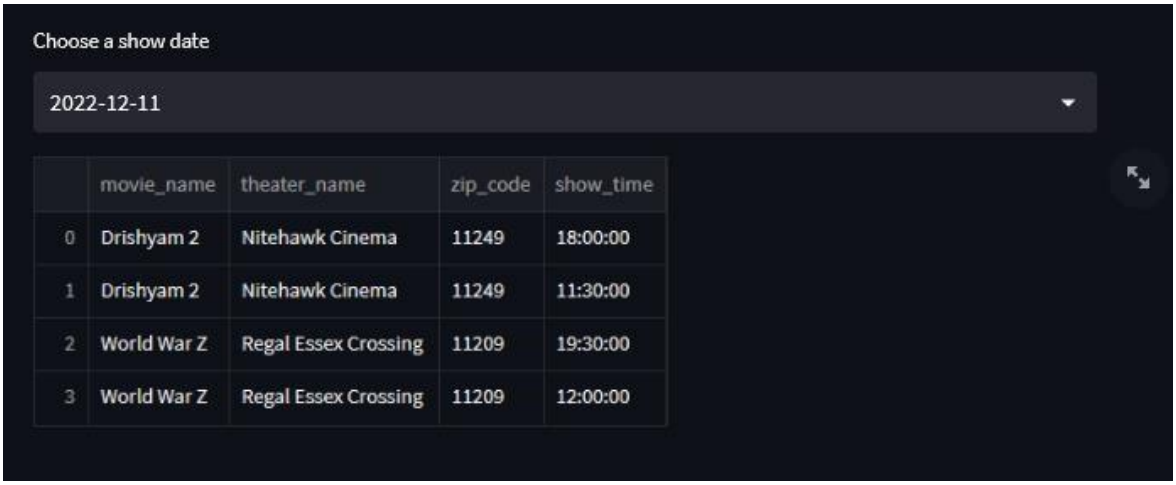
6) **Search Nearby Movies by Zipcode:**

A user can input a zip code and based on this zip code the user will be shown distance to all the theatres in our database along with the movie displayed in the theatres. Following this, a user can select a day for which he wants to see the show timings. Once a date is selected, a table will be displayed showing movie names along with show times being displayed in the theatres.

Movie Management System



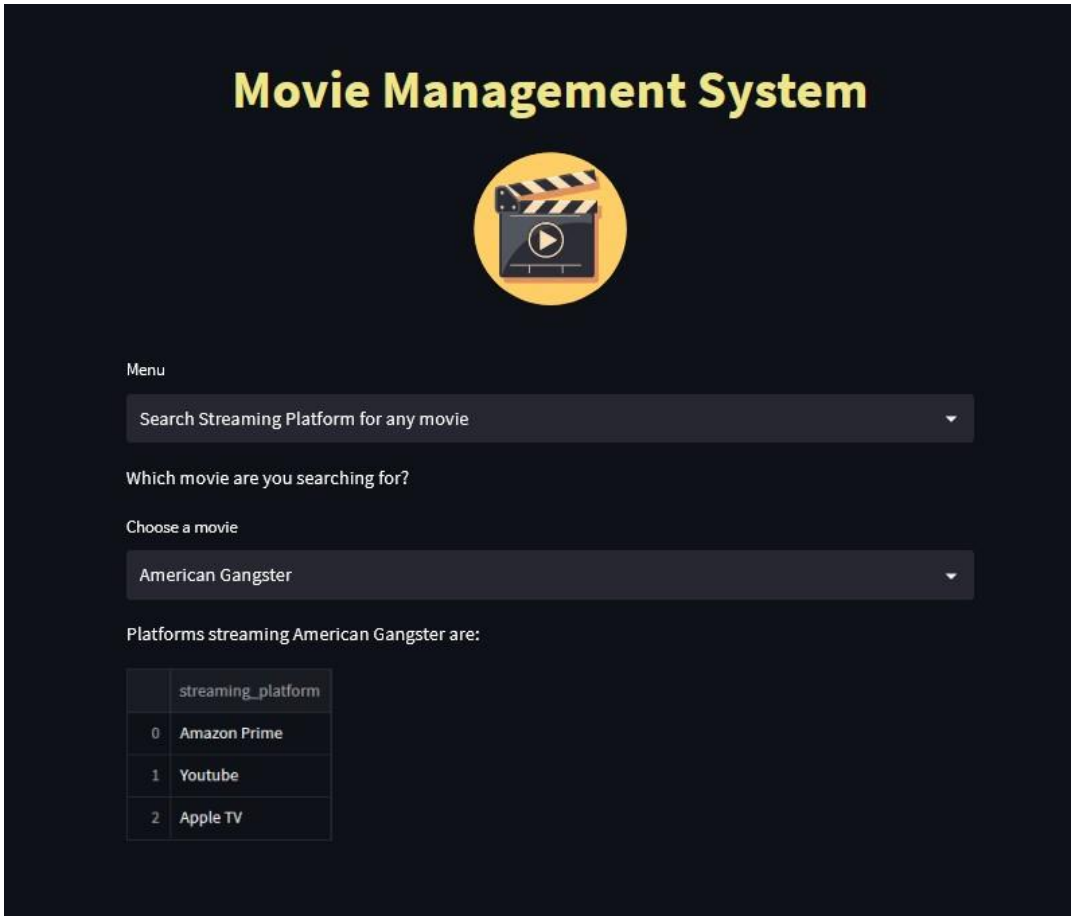
	Movie	Theater Name	Theater Zip Code	Distance(miles)
0	World War Z	Regal Essex Crossing	11209	2.3800
1	Drishyam 2	Nitehawk Cinema	11249	6.7500
2	Neighbors	AMC Village 7	10003	10.3200
3	Extraction	Alpine	11207	11.4600
4	The conjuring 2	AMC Empire 25	10036	13.5800
5	Thor: Love and Thunder	AMC Lincoln Square	10023	15.9300



	movie_name	theater_name	zip_code	show_time
0	Drishyam 2	Nitehawk Cinema	11249	18:00:00
1	Drishyam 2	Nitehawk Cinema	11249	11:30:00
2	World War Z	Regal Essex Crossing	11209	19:30:00
3	World War Z	Regal Essex Crossing	11209	12:00:00

Movie Management System

7) **Search Streaming Platform for any movie:**
The user can select a movie from the drop-down menu and search for its streaming platform.



Movie Management System

Schema.sql

```
DROP TABLE IF EXISTS Revenue CASCADE;
DROP TABLE IF EXISTS Movie CASCADE;
DROP TABLE IF EXISTS Casts CASCADE;
DROP TABLE IF EXISTS Director CASCADE;
DROP TABLE IF EXISTS Producer CASCADE;
DROP TABLE IF EXISTS Genre CASCADE;
DROP TABLE IF EXISTS Award CASCADE;
DROP TABLE IF EXISTS Theater CASCADE;
DROP TABLE IF EXISTS Show CASCADE;
DROP TABLE IF EXISTS StreamingPlatform CASCADE;
DROP TABLE IF EXISTS Viewer CASCADE;
DROP TABLE IF EXISTS Viewed_Rated_by CASCADE;
DROP TABLE IF EXISTS Movie_belong_to CASCADE;
DROP TABLE IF EXISTS Directed_by CASCADE;
DROP TABLE IF EXISTS Produced_by CASCADE;
DROP TABLE IF EXISTS Starred_by CASCADE;
DROP TABLE IF EXISTS Streamed_On CASCADE;
DROP TABLE IF EXISTS Played_At CASCADE;
DROP TABLE IF EXISTS Receive_Awards CASCADE;


CREATE TABLE Revenue (
  Id Integer Primary Key,
  TotalExpense bigint not null,
  TotalEarnings bigint not null,
  CollectedRevenue bigint GENERATED ALWAYS AS (TotalEarningsTotalExpense)STORED
);

CREATE TABLE Movie (
  Id integer Primary Key,
  Name varchar(100) not null,
  Language varchar(100),
  Country varchar(100),
  Release_date Date not null,
  Length Integer not null,
  Revenue_Id Integer not null,
  FOREIGN KEY(Revenue_Id) references Revenue(Id)
 );

CREATE TABLE Casts (
  Id integer Primary Key,
  Name varchar(100) not null,
  Age Integer,
  Gender varchar(10)
 );

CREATE TABLE Director (
```

Movie Management System

```
  Id integer Primary Key,
  Name varchar(100) not null,
  Age Integer,
  Gender varchar(10)
 );

CREATE TABLE Producer (
  Id integer Primary Key,
  Name varchar(100) not null,
  Age Integer,
  Gender varchar(10)
 );

CREATE TABLE Genre (
  Id integer Primary Key,
  Type varchar(128)
);

 CREATE TABLE Award (
  Id integer Primary Key,
  Type varchar(500)
);

CREATE TABLE Theater (
  Id Integer Primary Key,
  Name varchar(100) not null,
  ZIP_Code Integer
);

CREATE TABLE Show (
  Id Integer,
  Theater_Id Integer,
  S_Date Date not null,
  S_Time time not null,
  Primary Key(Theater_Id, Id),
  FOREIGN KEY(Theater_Id) references Theater(Id) on delete cascade
);

CREATE TABLE StreamingPlatform(
  Id Integer Primary Key,
  Name varchar(100) not null
 );


CREATE TABLE Viewer(
  Id Integer Primary Key,
  Name varchar(100) not null,
  Age Integer,
  Gender varchar(10)
 );
```

Movie Management System

```
CREATE TABLE Viewed_Rated_by(
  Viewer_Id Integer,
  Movie_Id Integer,
  Rating Float,
  Primary Key(Viewer_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Viewer_Id) references Viewer(Id)
 );


CREATE TABLE Movie_belong_to(
  Genre_Id Integer,
  Movie_Id Integer,
  Primary Key(Genre_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Genre_Id) references Genre(Id)
 );


CREATE TABLE Directed_by(
  Director_Id Integer,
  Movie_Id Integer,
  Primary Key(Director_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Director_Id) references Director(Id)
 );

 CREATE TABLE Produced_by(
  Producer_Id Integer,
  Movie_Id Integer,
  Primary Key(Producer_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Producer_Id) references Producer(Id)
 );

CREATE TABLE Starred_by(
  Cast_Id Integer,
  Movie_Id Integer,
  Primary Key(Cast_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Cast_Id) references Casts(Id)
 );

 CREATE TABLE Streamed_On(
  Streaming_Platform_Id Integer,
  Movie_Id Integer,
  Primary Key(Streaming_Platform_Id, Movie_Id),
  Foreign Key(Movie_Id) references Movie(Id),
  Foreign Key(Streaming_Platform_Id) references StreamingPlatform(Id)
 );
```

Movie Management System

```
CREATE TABLE Played_At(
 Theater_Id Integer,
 Movie_Id Integer,
 Primary Key(Theater_Id, Movie_Id),
 Foreign Key(Movie_Id) references Movie(Id),
 Foreign Key(Theater_Id) references Theater(Id)
);

CREATE TABLE Receive_Awards(
 Award_Id Integer,
 Year Integer,
 Movie_Id Integer,
 Primary Key(Award_Id, Movie_Id),
 Foreign Key(Award_Id) references Award(Id),
 Foreign Key(Movie_Id) references Movie(Id)
);
```

Data loading commands

Command to transfer all our csv data files from local system onto jedi.poly.edu

This is an optional step as all our data is already uploaded and is present in folder as15098_mp5578_project/data.

scp award.csv Casts.csv Directed_by.csv Director.csv Genre.csv Movie_belong_to.csv Movie.csv Played_At.csv Produced_by.csv Producer.csv Receive_Awards.csv Revenue.csv Show.csv Starred_by.csv Streamed_On.csv StreamingPlatform.csv Theater.csv Viewed_Rated_by.csv Viewer.csv as15098@jedi.poly.edu:~

Command to copy data from csv files on jedi uploaded via the above command to the tables created by running schema.sql file.

Go inside project folder cd as15098_mp5578_project/data

cat Revenue.csv | psql -U as15098 -d as15098_db -c "COPY Revenue from STDIN CSV HEADER"

cat Movie.csv | psql -U as15098 -d as15098_db -c "COPY Movie from STDIN CSV HEADER"

cat Casts.csv | psql -U as15098 -d as15098_db -c "COPY Casts from STDIN CSV HEADER"

cat Director.csv | psql -U as15098 -d as15098_db -c "COPY Director from STDIN CSV HEADER"

cat Producer.csv | psql -U as15098 -d as15098_db -c "COPY Producer from STDIN CSV HEADER"

Movie Management System

cat Genre.csv | psql -U as15098 -d as15098_db -c "COPY Genre from STDIN CSV HEADER"

cat award.csv | psql -U as15098 -d as15098_db -c "COPY award from STDIN CSV HEADER"

cat Theater.csv | psql -U as15098 -d as15098_db -c "COPY Theater from STDIN CSV HEADER"

cat Show.csv | psql -U as15098 -d as15098_db -c "COPY Show from STDIN CSV HEADER"

cat StreamingPlatform.csv | psql -U as15098 -d as15098_db -c "COPY StreamingPlatform from STDIN CSV HEADER"

cat Viewer.csv | psql -U as15098 -d as15098_db -c "COPY Viewer from STDIN CSV HEADER"

cat Viewed_Rated_by.csv | psql -U as15098 -d as15098_db -c "COPY Viewed_Rated_by from STDIN CSV HEADER"

cat Movie_belong_to.csv | psql -U as15098 -d as15098_db -c "COPY Movie_belong_to from STDIN CSV HEADER"

cat Directed_by.csv | psql -U as15098 -d as15098_db -c "COPY Directed_by from STDIN CSV HEADER"

cat Produced_by.csv | psql -U as15098 -d as15098_db -c "COPY Produced_by from STDIN CSV HEADER"

cat Starred_by.csv | psql -U as15098 -d as15098_db -c "COPY Starred_by from STDIN CSV HEADER"

cat Streamed_On.csv | psql -U as15098 -d as15098_db -c "COPY Streamed_On from STDIN CSV HEADER"

cat Played_At.csv | psql -U as15098 -d as15098_db -c "COPY Played_At from STDIN CSV HEADER"

cat Receive_Awards.csv | psql -U as15098 -d as15098_db -c "COPY Receive_Awards from STDIN CSV HEADER"