DBMS-Mini Project

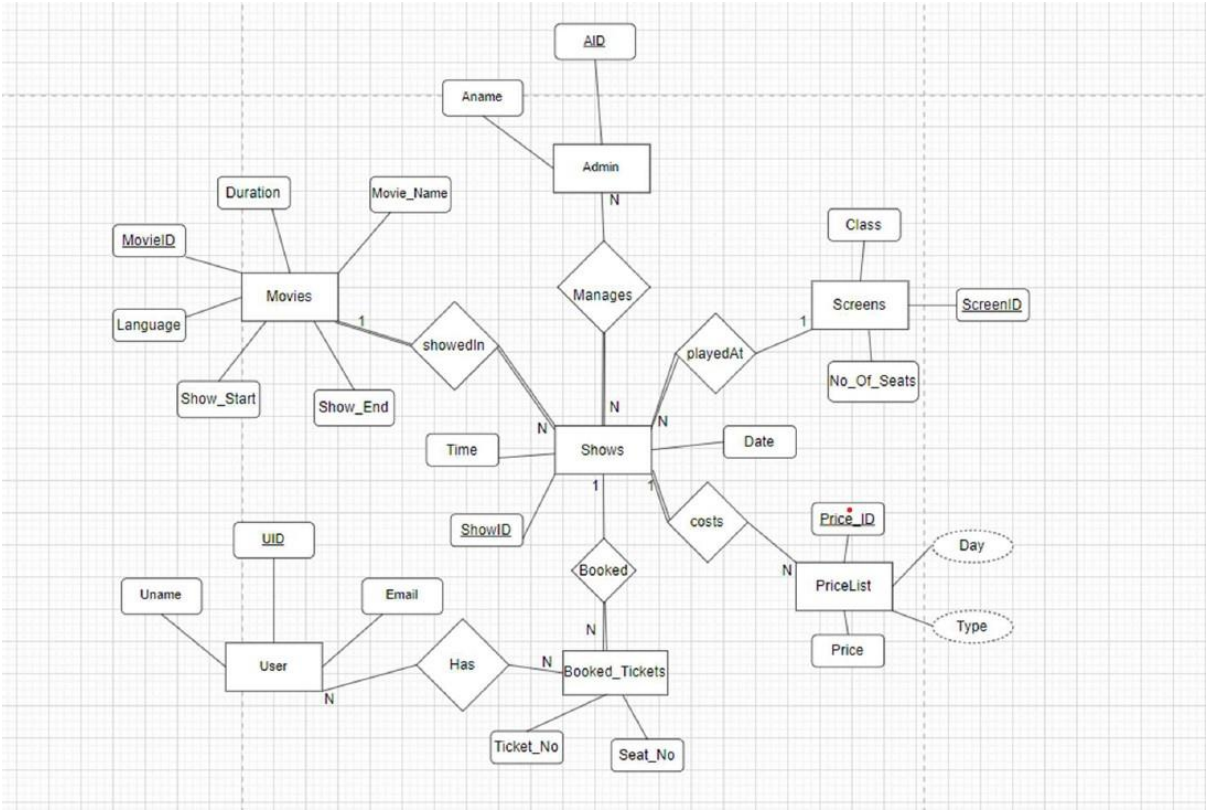
Online Movie Ticket Booking Management System

Done By:-

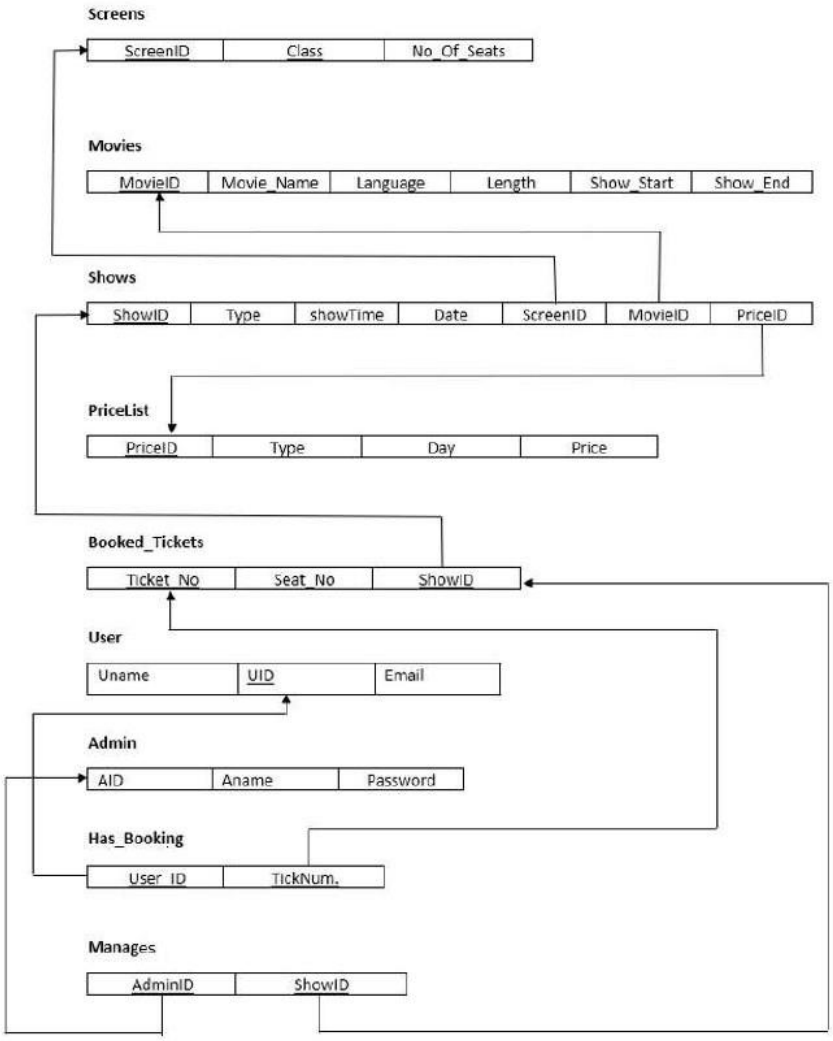
Name- Karan Mangtani
SRN- PES1UG21CS268
Sec-E

Name-Krishna Sudarshan
SRN-PES1UG21CS289
Sec-E

ER Diagram:-



Relational Schema:-



Join Query:-

1.) Return show_id for each ticket booked

```
select show_id from pes1ug21cs268_289_shows natural join
pes1ug21cs268_289_booked_tickets;
```

```
mysql> select show_id from pes1ug21cs268_289_shows natural join pes1ug21cs268_289_booked_tickets;
+---------+
| show_id |
+---------+
|       1 |
|       1 |
|       3 |
|       2 |
|       2 |
+---------+
5 rows in set (0.00 sec)
```

2.) Return show information with details of movie playing in that show

```
select * from pes1ug21cs268_289_shows left join pes1ug21cs268_289_movies on pes1ug21cs268_289_shows.movie_id = pes1ug21cs268_289_movies.movie_id;
```

```
mysql> select * from pes1ug21cs268_289_shows left join pes1ug21cs268_289_movies on pes1ug21cs268_289_shows.movie_id = pes1ug21cs268_289_movies.movie_id;
+---------+------+----------+------------+-----------+----------+----------+----------+-----------------+----------+--------+------------+------------+
| show_id | type | time     | date       | screen_id | movie_id | price_id | movie_id | movie_name      | language | length | show_start | show_end   |
+---------+------+----------+------------+-----------+----------+----------+----------+-----------------+----------+--------+------------+------------+
|       1 |    3 | 12:00:00 | 2023-11-17 |         1 |      101 |        1 |      101 | Black Panther 2 | English  |    161 | 2023-11-11 | 2023-12-11 |
|       2 |    2 | 11:30:00 | 2023-11-17 |         3 |      103 |        2 |      103 | Kantara         | Kannada  |    150 | 2023-09-30 | 2023-12-10 |
|       3 |    3 | 12:00:00 | 2023-11-18 |         1 |      101 |        1 |      101 | Black Panther 2 | English  |    161 | 2023-11-11 | 2023-12-11 |
|       4 |    2 | 11:30:00 | 2023-11-19 |         2 |      102 |        3 |      102 | Ram Setu        | Hindi    |    143 | 2023-10-25 | 2023-11-28 |
|       5 |    1 | 21:30:00 | 2023-11-20 |         4 |      101 |        5 |      101 | Black Panther 2 | English  |    161 | 2023-11-11 | 2023-12-11 |
+---------+------+----------+------------+-----------+----------+----------+----------+-----------------+----------+--------+------------+------------+
5 rows in set (0.00 sec)
```

3.) Return data about show and the screen being used

```
select * from pes1ug21cs268_289_shows natural join pes1ug21cs268_289_screens;
```

```
mysql> select * from pes1ug21cs268_289_shows natural join pes1ug21cs268_289_screens;
+-----------+---------+------+----------+------------+----------+----------+----------+-------------+
| screen_id | show_id | type | time     | date       | movie_id | price_id | class    | no_of_seats |
+-----------+---------+------+----------+------------+----------+----------+----------+-------------+
|         1 |       1 |    3 | 12:00:00 | 2023-11-17 |      101 |        1 | Platinum |         100 |
|         3 |       2 |    2 | 11:30:00 | 2023-11-17 |      103 |        2 | gold     |         120 |
|         1 |       3 |    3 | 12:00:00 | 2023-11-18 |      101 |        1 | Platinum |         100 |
|         2 |       4 |    2 | 11:30:00 | 2023-11-19 |      102 |        3 | standard |         150 |
|         4 |       5 |    1 | 21:30:00 | 2023-11-20 |      101 |        5 | platinum |          50 |
+-----------+---------+------+----------+------------+----------+----------+----------+-------------+
5 rows in set (0.00 sec)
```

AGGREGATE FUNCTIONS :-

1. Return Day and no of shows playing on that day

```
select day,count(*) from pes1ug21cs268_289_price_list group by day;
```

```
mysql> select day,count(*) from pes1ug21cs268_289_price_list group by day;
+-----------+----------+
| day       | count(*) |
+-----------+----------+
| Friday    |        2 |
| Saturday  |        2 |
| Sunday    |        1 |
+-----------+----------+
3 rows in set (0.00 sec)
```

2. Return count of movie language

```
select language,count(*) from pes1ug21cs268_289_movies group by language;
```

```
mysql> select language,count(*) from pes1ug21cs268_289_movies group by language;
+----------+----------+
| language | count(*) |
+----------+----------+
| English  |        2 |
| Hindi    |        2 |
| Kannada  |        1 |
+----------+----------+
3 rows in set (0.00 sec)
```

3. Return No of tickets booked for each movie

```
select movie_name,count(*) from pes1ug21cs268_289_movies natural join
pes1ug21cs268_289_shows group by movie_name;
```

```
mysql> select movie_name,count(*) from pes1ug21cs268_289_movies natural join pes1ug21cs268_289_shows group by movie_name;
+----------------+----------+
| movie_name     | count(*) |
+----------------+----------+
| Black Panther 2 |        3 |
| Ram Setu        |        1 |
| Kantara         |        1 |
+----------------+----------+
3 rows in set (0.00 sec)
```

4. Return No of times a type of screen was used

```
select class,count(*) from pes1ug21cs268_289_screens natural join
pes1ug21cs268_289_shows group by class;
```

```
mysql> select class,count(*) from pes1ug21cs268_289_screens natural join pes1ug21cs268_289_shows group by class;
+----------+----------+
| class    | count(*) |
+----------+----------+
| Platinum |        3 |
| standard |        1 |
| gold     |        1 |
+----------+----------+
3 rows in set (0.00 sec)
```

5. Returns max and min price for each day

```
select day,max(price),min(price) from pes1ug21cs268_289_price_list
group by day;
```

```
mysql> select day,max(price),min(price) from pes1ug21cs268_289_price_list group by day;
+----------+------------+------------+
| day      | max(price) | min(price) |
+----------+------------+------------+
| Friday   |        500 |        350 |
| Saturday |        500 |        350 |
| Sunday   |        450 |        450 |
+----------+------------+------------+
3 rows in set (0.00 sec)
```

SET OPERATORS :-

1. Returns all screen types and movie names

```
select screen_id,class from pes1ug21cs268_289_screens union select
movie_id,movie_name from pes1ug21cs268_289_movies;
```

```
mysql> select screen_id,class from pes1ug21cs268_289_screens union select movie_id,movie_name from pes1ug21cs268_289_movies;
+-----------+-----------------+
| screen_id | class           |
+-----------+-----------------+
|         1 | Platinum        |
|         2 | standard        |
|         3 | gold            |
|         4 | platinum        |
|         5 | gold            |
|       101 | Black Panther 2 |
|       102 | Ram Setu        |
|       103 | Kantara         |
|       104 | Avatar-2        |
|       105 | Bhediya         |
+-----------+-----------------+
10 rows in set (0.00 sec)
```

2. Returns ID's of movies which have shows

```
select movie_id from pes1ug21cs268_289_movies where movie_id  in
(select movie_id from pes1ug21cs268_289_shows);
```

```
mysql> select movie_id from pes1ug21cs268_289_movies where movie_id  in (select movie_id from pes1ug21cs268_289_shows);
+----------+
| movie_id |
+----------+
|      101 |
|      102 |
|      103 |
+----------+
3 rows in set (0.00 sec)
```

3. Returns screen-id which is not being used for any show

```
select screen_id from pes1ug21cs268_289_screens where screen_id not in
(select screen_id from pes1ug21cs268_289_shows);
```

```
mysql> select screen_id from pes1ug21cs268_289_screens where screen_id not in (select screen_id from pes1ug21cs268_289_shows);
+-----------+
| screen_id |
+-----------+
|         5 |
+-----------+
1 row in set (0.00 sec)
```

4. Returns show-id's for which there are no booked tickets

```
select show_id from pes1ug21cs268_289_shows where show_id not in
(select show_id from pes1ug21cs268_289_booked_tickets);
```

```
mysql> select show_id from pes1ug21cs268_289_shows where show_id not in (select show_id from pes1ug21cs268_289_booked_tickets);
+---------+
| show_id |
+---------+
|       5 |
|       4 |
+---------+
2 rows in set (0.00 sec)
```

FUNCTIONS AND PROCEDURE :-

Function

```
DELIMITER $$
CREATE FUNCTION no_of_freeseats(screen_id int,sh_date date, sh_time
varchar(10))
RETURNS int
DETERMINISTIC
BEGIN
DECLARE non_free int;
DECLARE shid int;
DECLARE total int;
set shid = (select show_id from v1 where screen_id = screen_id and date =
sh_date and time = sh_time);
```

```
set total = (select no_of_seats  from v1 where screen_id = screen_id and date
= sh_date and time = sh_time);
set non_free = (select count(*) from pes1ug21cs268_289_booked_tickets where
show_id = shid);
RETURN total-non_free;
END $$
DELIMITER;
```

```
mysql> DELIMITER $$
mysql> CREATE FUNCTION no_of_freeseats(screen_id int,sh_date date, sh_time varchar(10))
    -> RETURNS int
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE non_free int;
    -> DECLARE shid int;
    -> DECLARE total int;
    -> set shid = (select show_id from v1 where screen_id = screen_id and date = sh_date and time = sh_time);
    -> set total = (select no_of_seats  from v1 where screen_id = screen_id and date = sh_date and time = sh_time);
    -> set non_free = (select count(*) from pes1ug21cs268_289_booked_tickets where show_id = shid);
    -> RETURN total-non_free;
    -> END $$
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> Create view v1 as select * from pes1ug21cs268_289_shows natural join pes1ug21cs268_289_screens;
Query OK, 0 rows affected (0.01 sec)

mysql> select no_of_freeseats(1,'2023-11-17','11:30:00');
+-------------------------------------------+
| no_of_freeseats(1,'2023-11-17','11:30:00') |
+-------------------------------------------+
|                                       118 |
+-------------------------------------------+
1 row in set (0.01 sec)
```

Procedures

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE get_tickets()
    ->  BEGIN
    ->          INSERT INTO count_tickets SELECT uname,count(*) from pes1ug21cs268_289_user natural join pes1ug21cs268_289_has_booked group by uname;
    ->  END;$$
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> call get_tickets;
Query OK, 5 rows affected (0.00 sec)

mysql> select * from count_tickets;
+----------+---------------+
| uname    | no_of_tickets |
+----------+---------------+
| Rohit    |             1 |
| Manoj    |             1 |
| Neeraj   |             1 |
| Rahul    |             1 |
| Varun HR |             1 |
+----------+---------------+
5 rows in set (0.00 sec)
```

TRIGGERS AND CURSORS :-

Trigger: Checks the correctness of seat_no

```
DELIMITER $$
CREATE TRIGGER capacity
BEFORE INSERT ON pes1ug21cs268_289_booked_tickets
FOR EACH ROW
BEGIN
```

```
    DECLARE coun int;
    DECLARE max1 int;
    DECLARE id int;
    DECLARE err_Msg varchar(50);
    SET err_Msg="Incorrect Seat No!!";
    set id = (select screen_id from pes1ug21cs268_289_shows where show_id =
new.show_id);
    set coun = new.seat_no;
    set max1 = (select no_of_seats from pes1ug21cs268_289_screens where
screen_id = id);
    IF coun>max1 or coun<1 then
        Signal sqlstate '45000'
        SET MESSAGE_TEXT=err_Msg;
    END IF;
END $$
```

```
mysql> insert into pes1ug21cs268_289_booked_tickets values(0171,149,001);
ERROR 1644 (45000): Incorrect Seat No!!
```

Cursors: Returns list of movies which belong to a particular language

```
DELIMITER $$
CREATE PROCEDURE get_info(IN lang varchar(50))
BEGIN
DECLARE done INT DEFAULT 0;
DECLARE m_name varchar(20);
DECLARE del_movie CURSOR FOR SELECT movie_name from
pes1ug21cs268_289_movies  natural join pes1ug21cs268_289_shows where language
= lang;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
OPEN del_movie;
get_movie: LOOP
FETCH del_movie into m_name;
IF done=1 THEN LEAVE get_movie;
END IF;
insert into v2 values(m_name);
END LOOP get_movie;
CLOSE del_movie;
END$$
DELIMITER;
```

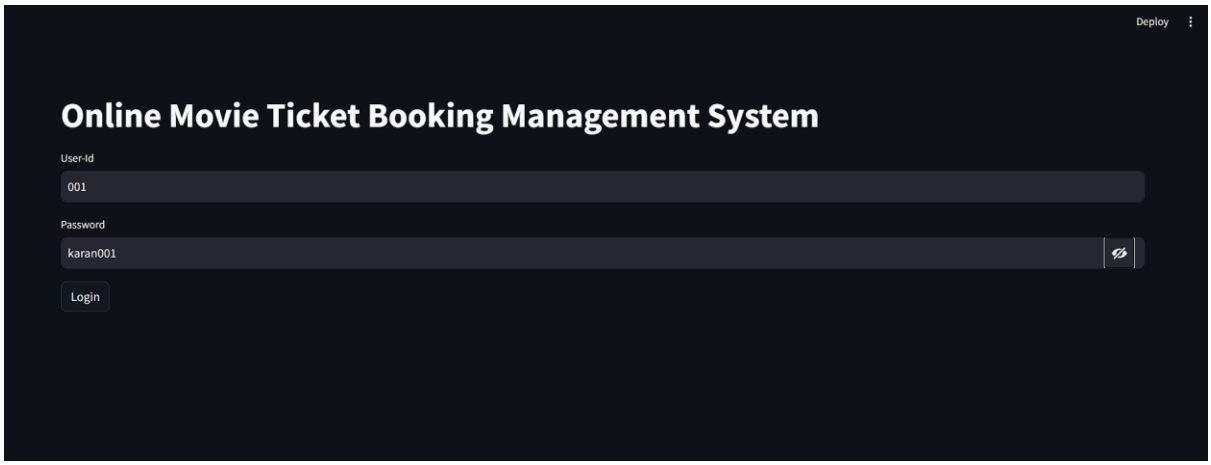
```
mysql> call get_info('English');
Query OK, 0 rows affected (0.01 sec)

mysql> select * from v2;
+-----------------+
| movie_name      |
+-----------------+
| Black Panther 2 |
| Avatar-2        |
+-----------------+
2 rows in set (0.00 sec)
```

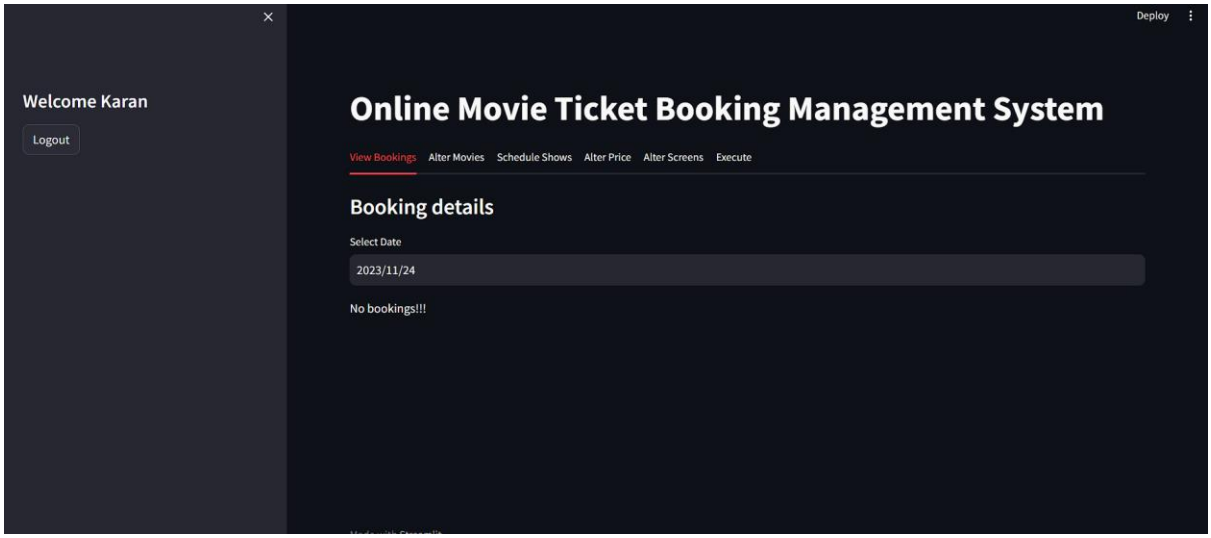
FRONTEND :-

The frontend for the database was made using the streamlit module on top of python.
It provides a generic and fluid UI that is easy to follow and use and is highly customisable.

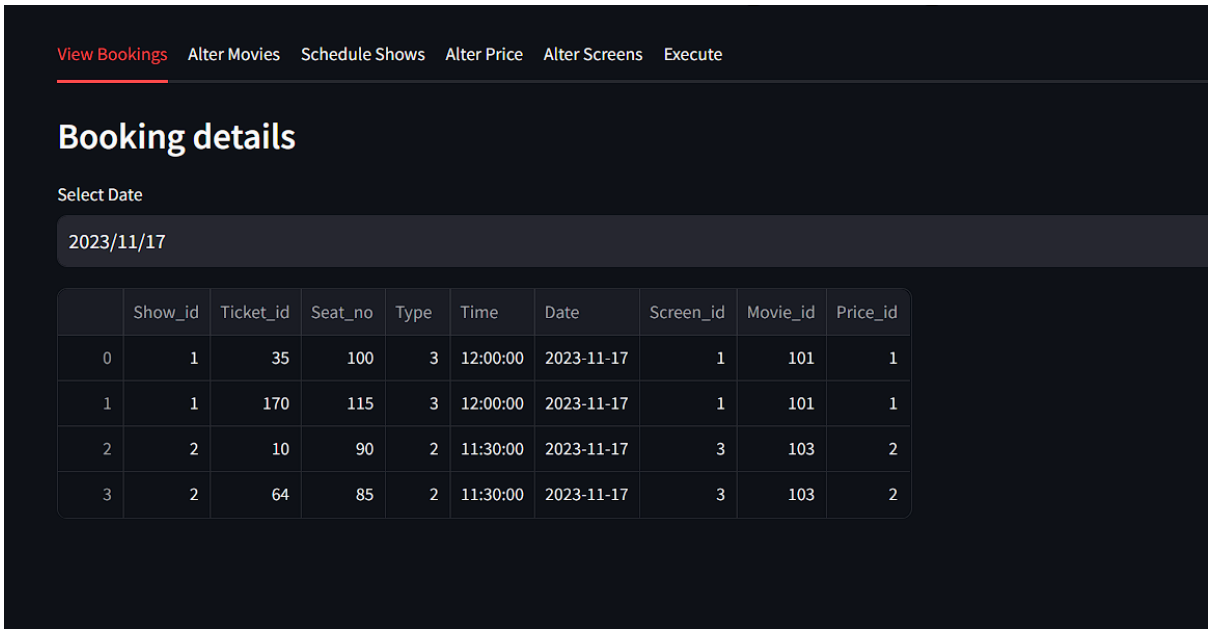
Login Page:



UI:

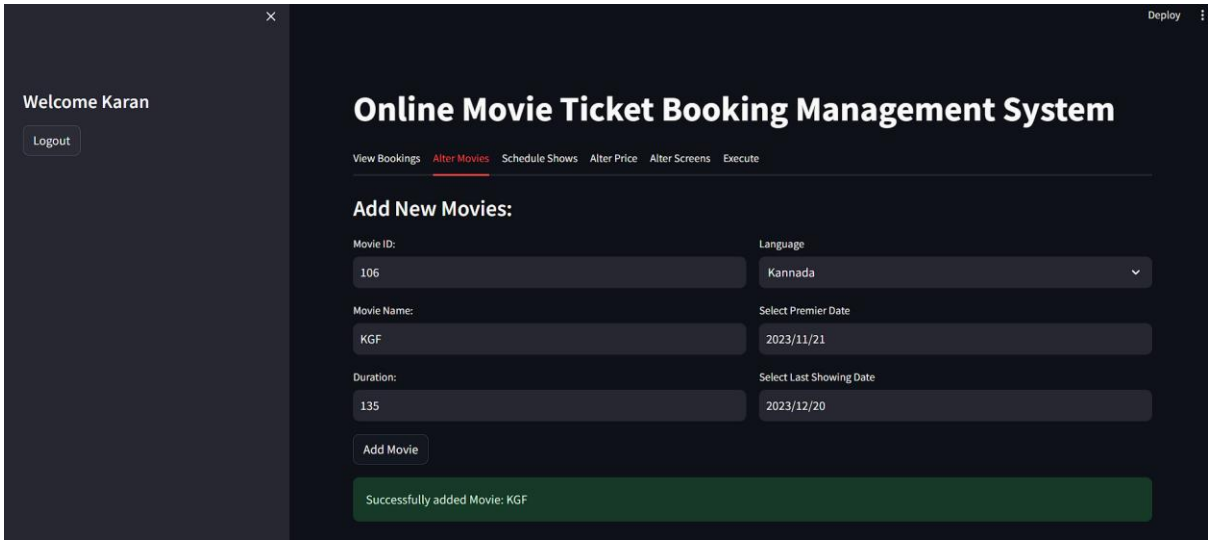


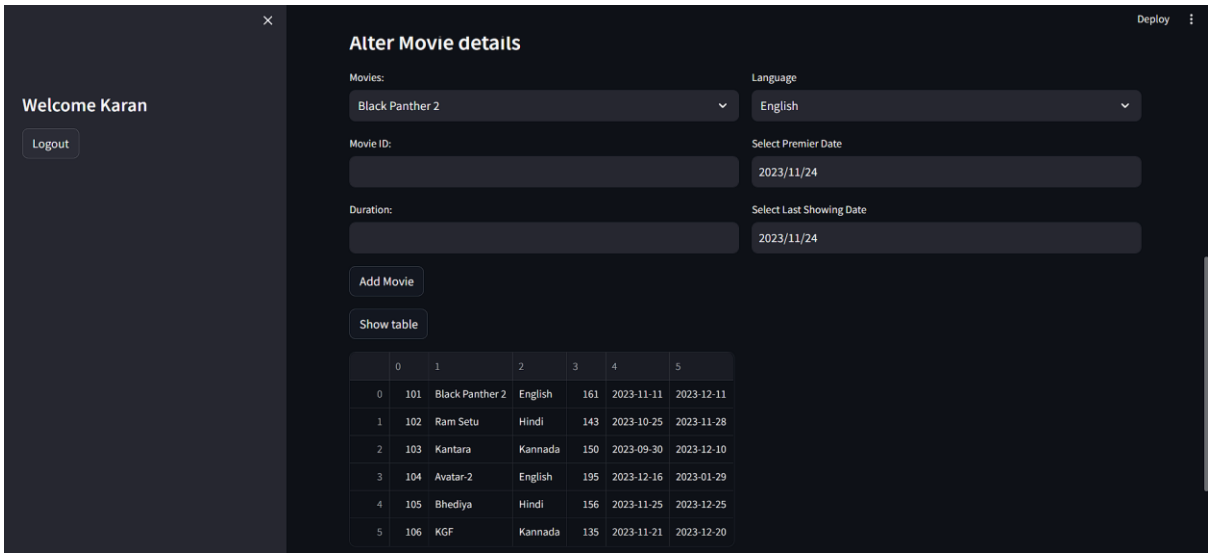
View:



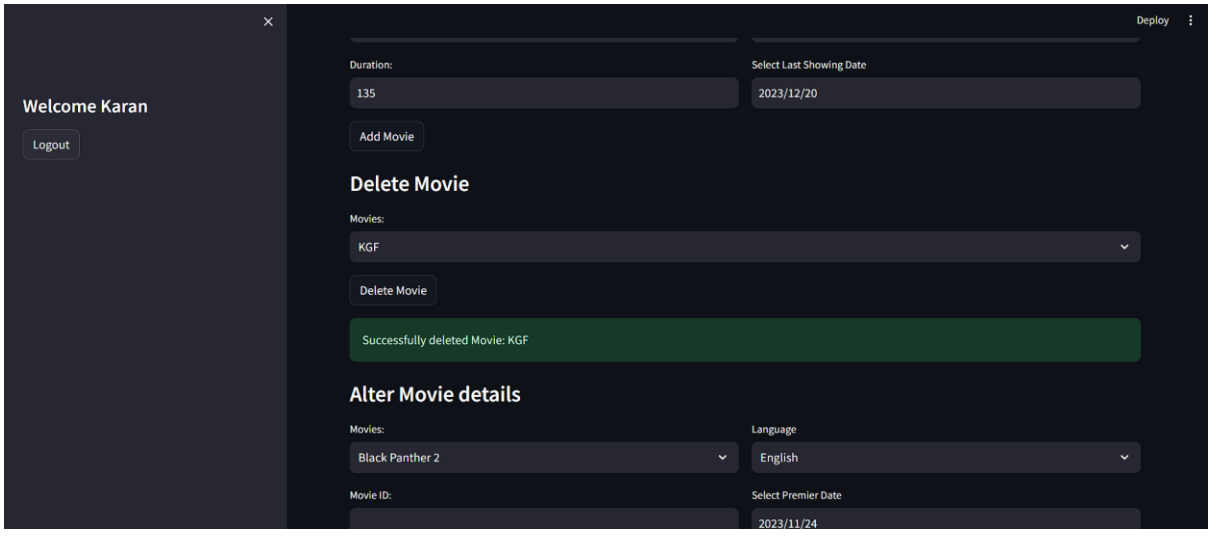
	Show_id	Ticket_id	Seat_no	Type	Time	Date	Screen_id	Movie_id	Price_id
0	1	35	100	3	12:00:00	2023-11-17	1	101	1
1	1	170	115	3	12:00:00	2023-11-17	1	101	1
2	2	10	90	2	11:30:00	2023-11-17	3	103	2
3	2	64	85	2	11:30:00	2023-11-17	3	103	2

Adding New Movie:





Deleting Movie:

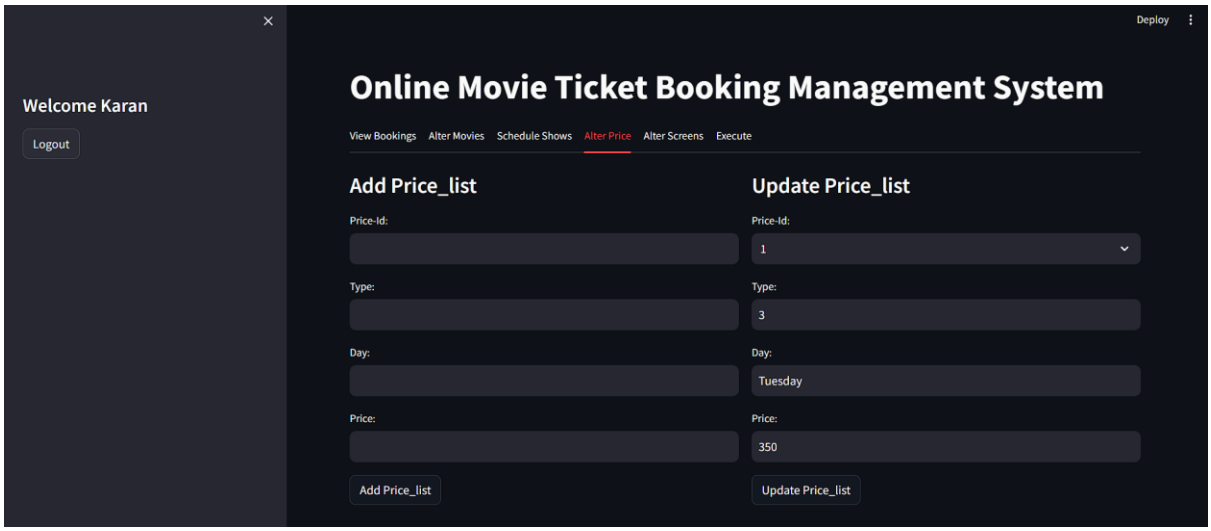


After Delete:

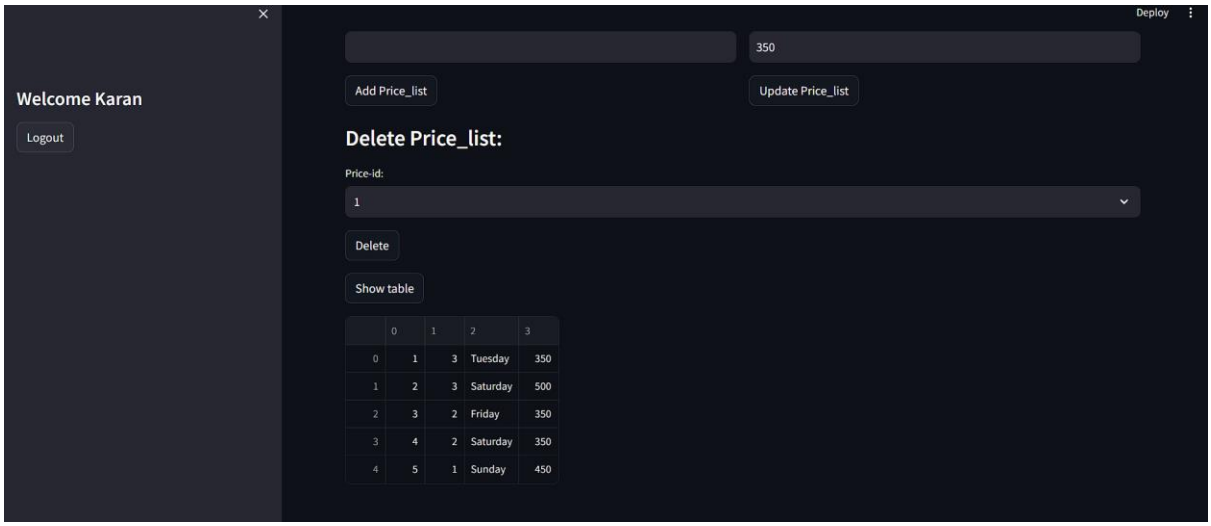
Show table

	0	1	2	3	4	5
0	101	Black Panther 2	English	161	2023-11-11	2023-12-11
1	102	Ram Setu	Hindi	143	2023-10-25	2023-11-28
2	103	Kantara	Kannada	150	2023-09-30	2023-12-10
3	104	Avatar-2	English	195	2023-12-16	2023-01-29
4	105	Bhediya	Hindi	156	2023-11-25	2023-12-25

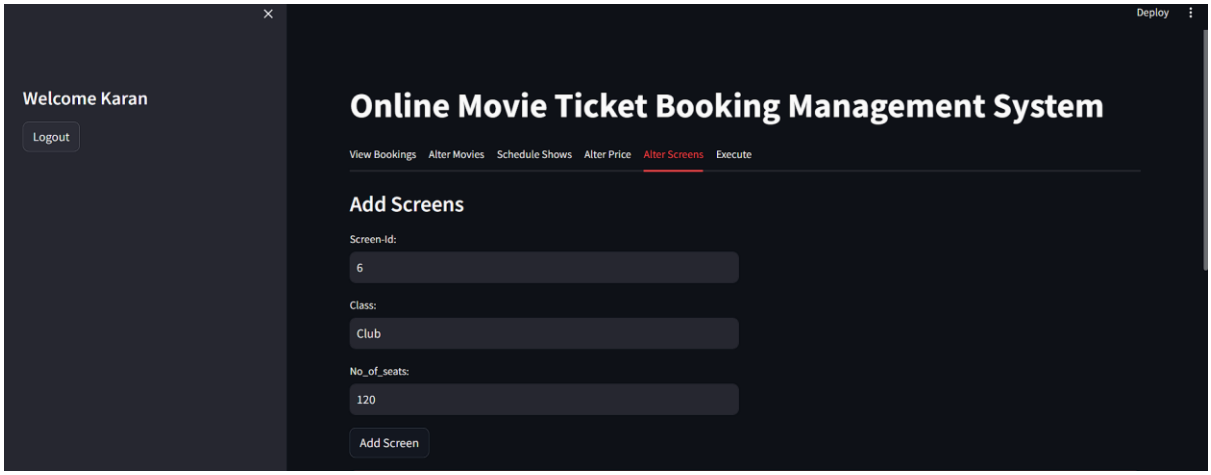
Updating Price_list:



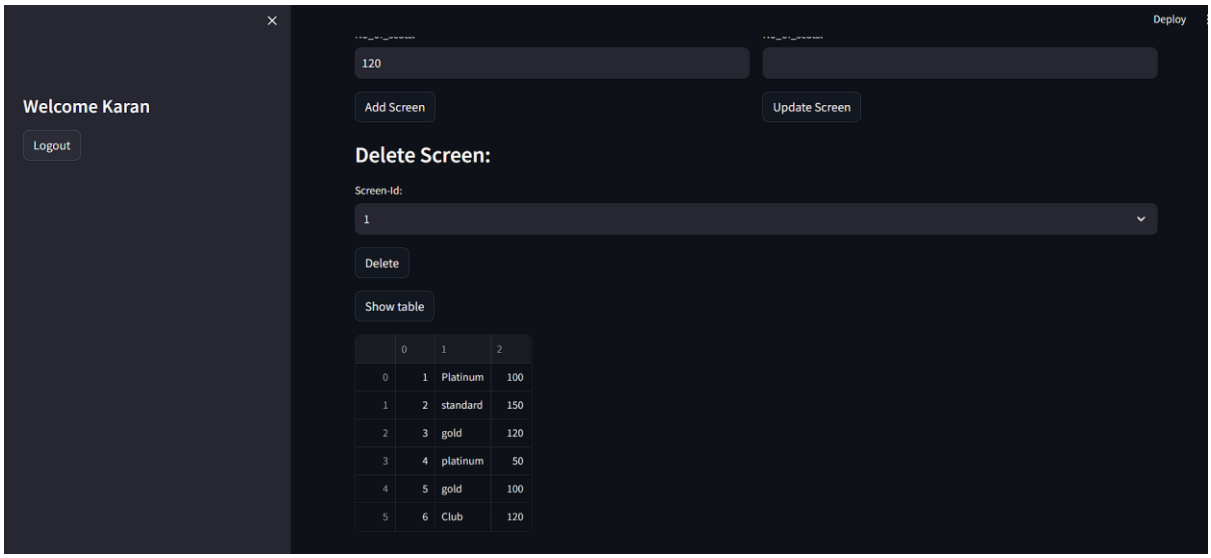
After Updating:



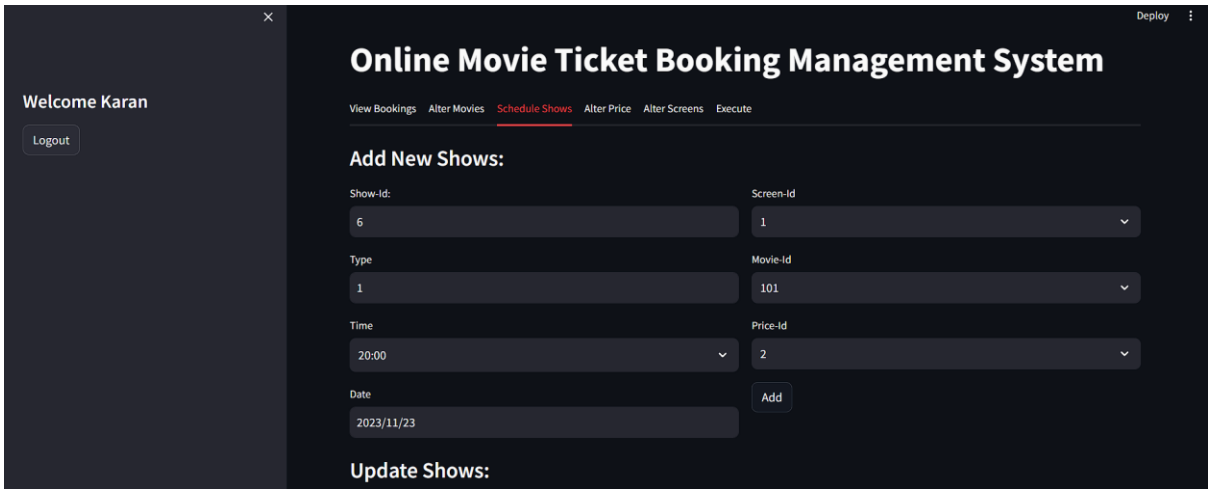
Adding Screens:



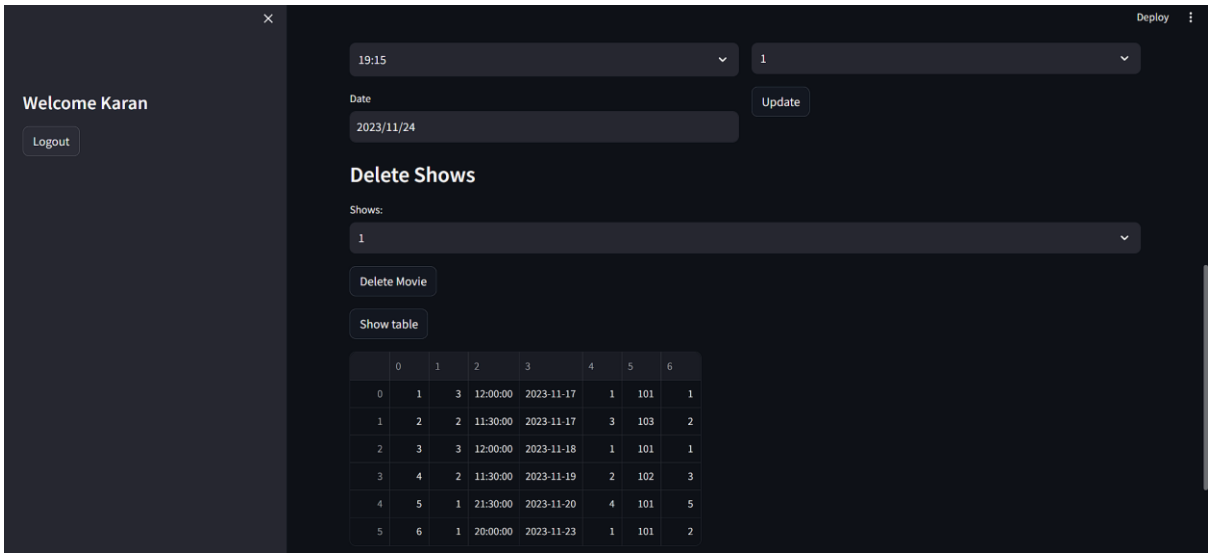
After Adding:



Adding New Show For a Screen:



After Adding:



Custom Querying: