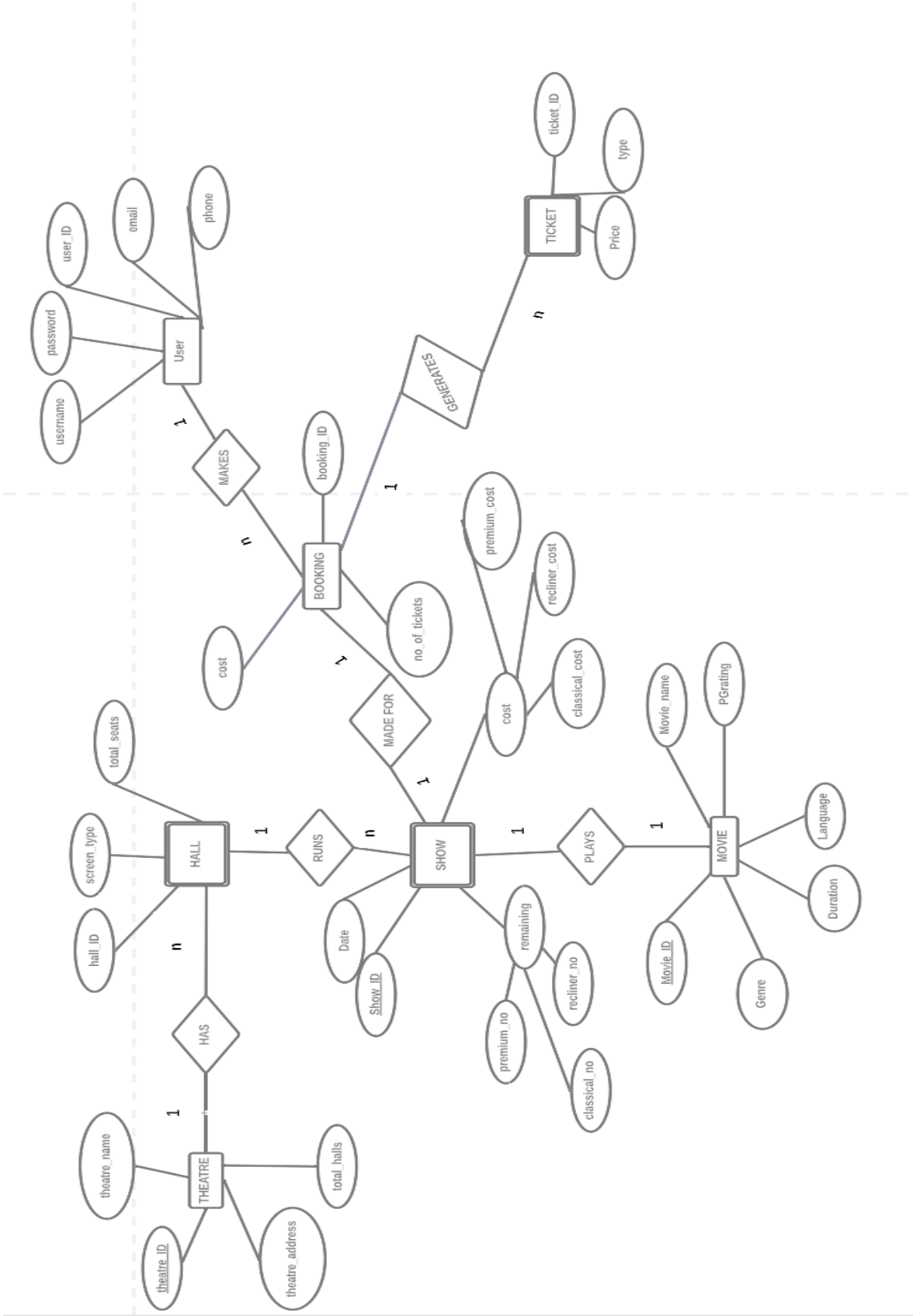
DBMS-Mini Project
MOVIE BOOKING MANAGEMENT SYSTEM

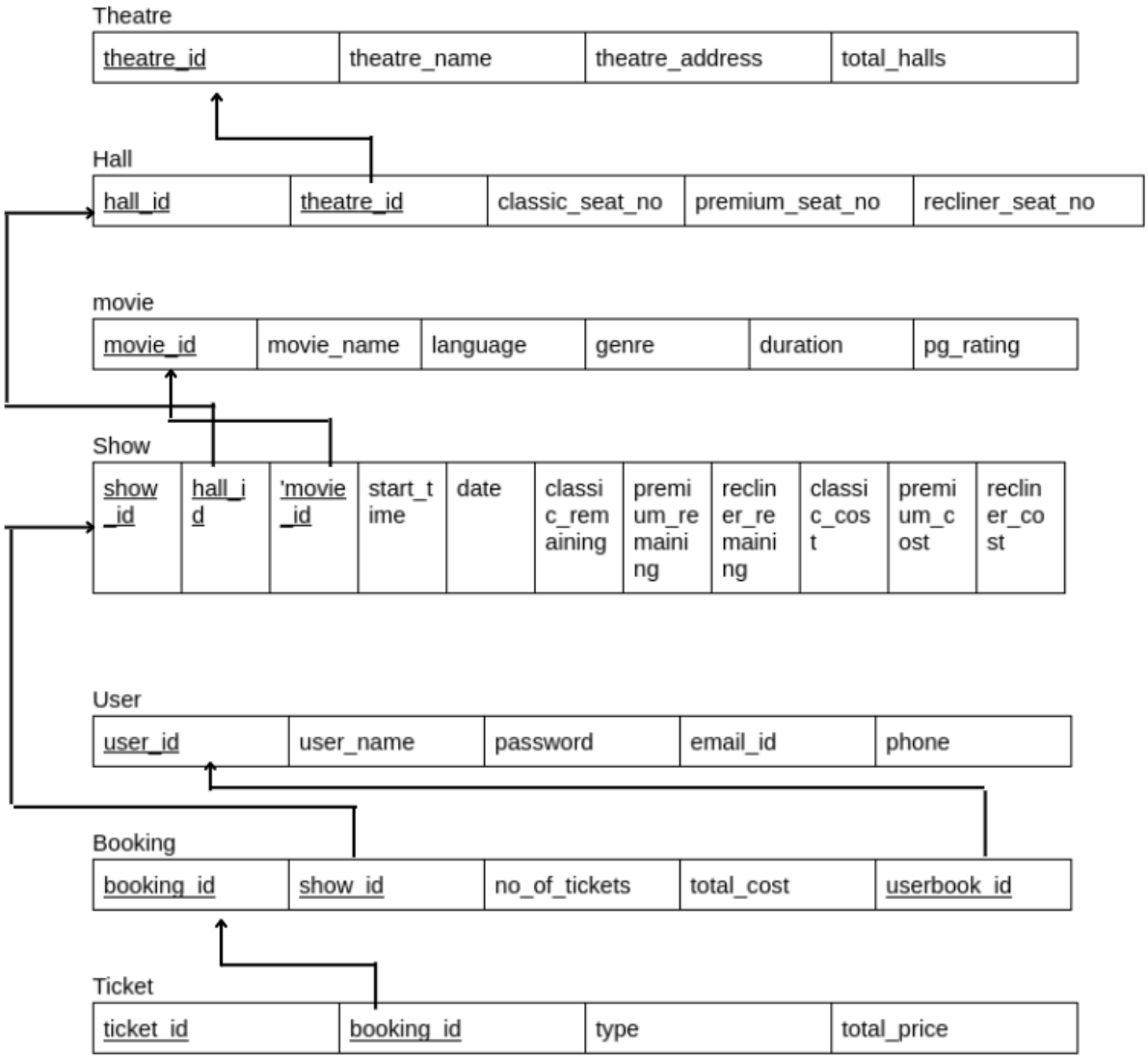
Submitted By:
ADITI SOORI
PES1UG20CS017
V Semester Section ‘A’

ABSTRACT

To 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. The system allows a user to book tickets for a given show, and for a manager/admin to add movies and shows to the database.
The Movie Booking Management System is created using Streamlit for the frontend and user interface and MySQL is used for the backend and storing of data. It allows basic CRUD operations through the
frontend which includes Adding, Viewing, Editing and Removal of tasks. It also supports login and authentication for both users and the admin.

ER DIAGRAM AND RELATIONAL SCHEMA


User
user_ID
password
username
email
phone
MAKES
1
n
BOOKING
booking_ID
cost
no of tickets
GENERATES
1
n
TICKET
ticket_ID
type
Price
MADE FOR
1
1
HALL
total_seats
screen_type
hall_ID
RUNS
1
n
SHOW
Date
Show_ID
remaining
premium_no
recliner_no
classical_no
cost
premium_cost
recliner_cost
classical_cost
PLAYS
1
1
MOVIE
Movie_name
PGrating
Language
Duration
Movie_ID
Genre
HAS
n
1
THEATRE
theatre_name
theatre_ID
theatre_address
total_halls


Theatre
theatre_id
theatre_name
theatre_address
total_halls
Hall
hall_id
theatre_id
classic_seat_no
premium_seat_no
recliner_seat_no
movie
movie_id
movie_name
language
genre
duration
pg_rating
Show
show_id
hall_id
'movie_id
start_time
date
classic_remaining
premium_remaining
recliner_remaining
classic_cost
premium_cost
recliner_cost
User
user_id
user_name
password
email_id
phone
Booking
booking_id
show_id
no_of_tickets
total_cost
userbook_id
Ticket
ticket_id
booking_id
type
total_price

DDL Statements – Building the Database
AND
Populating the Database

```
drop database movie_booking_system;
create database movie_booking_system;
use movie_booking_system;


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";
--
-- Database: `Movie_Management_System`
--

-- --------------------------------------------------------

--
-- Table structure for table `theatre`
--

CREATE TABLE `theatre` (
 `theatre_id` varchar(10) NOT NULL,
 `theatre_name` varchar(50) NOT NULL,
 `theatre_address` varchar(75) NOT NULL,
 `total_halls` int(4) NOT NULL,
  Primary Key(`theatre_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `theatre`
--

INSERT INTO `theatre` (`theatre_id`,  `theatre_name`, `theatre_address`,
`total_halls`) VALUES
('T01', 'PVR Cinemas','Koramangala, Bangalore',15),
('T02', 'INOX Movies', 'Jayanagar, Bangalore',25),
```

```
('T03', 'Cinepolis', 'Banaswadi, Bangalore',10),
('T04', 'IMAX Cinemas', 'Malleshwaram, Bangalore',20);
-- --------------------------------------------------------
--
-- Table structure for table `hall`
--

CREATE TABLE `hall` (
 `hall_id` varchar(10) NOT NULL,
 `theatre_id` varchar(10) NOT NULL,
 `classic_seat_no` int(11) NOT NULL,
 `premium_seat_no` int(11) NOT NULL,
 `recliner_seat_no` int(11) NOT NULL,
  Primary Key(`hall_id`),
  Foreign Key(`theatre_id`) REFERENCES `theatre`(`theatre_id`) ON DELETE
CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `hall`
--

INSERT INTO `hall` (`hall_id`,  `theatre_id`, `classic_seat_no`,
`premium_seat_no`,`recliner_seat_no`) VALUES
('T010','T03',80,20,10),
('T014','T01',60,10,15),
('T020','T04',75,35,5),
('T022','T02',60,30,15),
('T006','T03',69,18,12),
('T023','T02',78,20,10),
('T015','T01',92,12,8),
('T012','T04',56,30,15),
('T008','T03',43,40,16),
('T003','T04',81,20,12),
('T021','T02',72,15,20);
-- --------------------------------------------------------
--
-- Table structure for table `movie`
--

CREATE TABLE `movie` (
```

```
 `movie_id` varchar(10) NOT NULL,
 `movie_name` varchar(100) NOT NULL,
 `language` varchar(50) NOT NULL,
 `genre` varchar(20),
 `duration` varchar(45) NOT NULL,
  `pg_rating` varchar(5) NOT NULL,
    Primary Key(`movie_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `movie`
--

INSERT INTO `movie`
(`movie_id`,`movie_name`,`language`,`genre`,`duration`,`pg_rating`) VALUES
('001', 'Hichki', 'Hindi','Drama/Comedy','2 hrs 15 mins','U/A'),
('002', 'Pacific Rim Uprising',  'English','Fantasy/SciFi','1 hrs 55
mins','U/A'),
('003', 'Strangers : Prey at night','English', 'Horror','3 hrs 10 mins',
'U/A'),
('004', 'Tomb Raider', 'English','Fantasy/Action','3 hrs 15 mins', 'A'),
('005', 'Midnight Sun', 'English','Romance','2 hrs 55 mins','R'),
('006', 'Peter Rabbit','English', 'Fantasy/Adventure','2 hrs 35
mins','U/A'),
('007', 'Black Panther','English', 'Fantasy/SciFi','2 hrs 15 mins','U/A'),
('008', 'Maze Runner: The Death Cure','English', 'Fantasy/SciFi','2 hrs 45
mins','U/A'),
('009', 'Insidious: The Last Key','English', 'Horror','2 hrs 20
mins','U/A'),
('010', 'Blackmail','Hindi', 'Comedy','1 hrs 55 mins','U/A'),
('011', 'Parmanu: The Story of Pokhran','Hindi', 'Drama/Thriller','3 hrs
10 mins','U/A'),
('012', '3 Storeys','Hindi', 'Drama','1 hrs 45 mins','U/A'),
('013', 'Rajaratha','Kannada', 'Comedy','2 hrs 45 mins','U/A'),
('014', 'Yogi Duniya','Kannada', 'Drama/Thriller','3 hrs 10 mins','U/A'),
('015', 'Kurukshetra','Kannada', 'Fantasy/History','1 hrs 45 mins','U/A'),
('016', 'Kantara','Kannada','Drama/Thriller','2 hrs 30 mins','U/A');
--
-- Table structure for table `show`
--
```

```
CREATE TABLE `show` (
 `show_id` varchar(10) NOT NULL,
 `hall_id` varchar(10) NOT NULL,
 `movie_id` varchar(10) NOT NULL,
 `start_time` time DEFAULT NULL,
 `date` date DEFAULT NULL,
 `classic_remaining` int NOT NULL CHECK (`classic_remaining` >= 0),
 `premium_remaining` int NOT NULL CHECK (`premium_remaining` >= 0),
 `recliner_remaining` int NOT NULL CHECK (`recliner_remaining` >= 0),
 `classic_cost` int NOT NULL,
 `premium_cost` int NOT NULL,
 `recliner_cost` int NOT NULL,
   Primary Key(`show_id`),
  Foreign Key(`hall_id`) REFERENCES `hall`(`hall_id`) ON DELETE CASCADE ON
UPDATE CASCADE,
  Foreign Key(`movie_id`) REFERENCES `movie`(`movie_id`) ON DELETE CASCADE
ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `show`
--

INSERT INTO `show` VALUES
('SHT0140002', 'T014', '015','04:20:00 ', '2022-11-05', 4, 5,12, 300,
450,575),
('SHT0210002',  'T021', '016','04:20:00', '2022-11-08', 22, 8,10, 395,
425,500),
('SHT0220002', 'T022', '001','04:20:00', '2022-11-15', 8, 5,3, 300,
375,450),
('SHT0230002',  'T023', '002','04:20:00 ', '2022-11-25', 40,16,8,295,
355,415),
('SHT0240002',  'T020', '003','04:20:00 ', '2022-11-05', 20,10,2, 395,
325,495),
('SHT0310002', 'T008', '004', '04:20:00 ', '2022-11-21', 35,10,5, 325,
350,400),
('SHT0320002', 'T012', '005', '04:20:00', '2022-11-12', 10,8,2, 315,
375,425),
('SHT0330002', 'T003', '006','04:20:00 ', '2022-11-11', 21,15,1, 425,
450,500),
```

```
('SHT0110003', 'T010', '007','07:30:00 ', '2022-11-01', 3,12,8, 400,475,
550),
('SHT0120003', 'T006', '008', '07:30:00', '2022-11-04', 17,6,8, 275,
315,400);
--
-- Table structure for table `user`
--

CREATE TABLE `user` (
 `user_id` varchar(15) NOT NULL,
 `user_name` varchar(100) NOT NULL,
 `email_id` varchar(50) NOT NULL,
  `phone` varchar(10) NOT NULL,
  Primary Key(`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `user`
--

INSERT INTO `user` VALUES
('100', 'Amit',  'amitsinhT04@gmail.com', '9846273634'),
('101', 'Raghav', 'seth.raghav987@gmail.com','7845279834'),
('102', 'Anjali',  'anjali23g@gmail.com', '8849273345'),
('103', 'Joy', 'jmathew.123@gmail.com',  '9000567890'),
('104', 'Sudha',  'sudha_sunil07@gmail.com',  '8874323461'),
('105', 'Ajay',  'kumarajayv56@gmail.com',  '9078985643'),
('106', 'Vikram',  'jvikram.89@gmail.com', '7750912345'),
('107', 'Komal',  'komal.agarwal87@gmail.com',  '9345687654'),
('108', 'Maitri',  'maitrishahj1@gmail.com',  '9922345016'),
('109', 'Bhavya',  'bhavyashastri@gmail.com',  '8567409098'),
('110', 'Preeti',  'preeti.jain@gmail.com',  '7765433211'),
('111', 'Shreya', 'rathod_shreya@gmail.com','9800215673'),
('112', 'Aditya', 'adityarajesh2902@gmail.com','9108996762');

--
--
-- Table structure for table `booking`
--

CREATE TABLE `booking` (
```

```
 `booking_id` varchar(10) NOT NULL,
 `show_id` varchar(10) NOT NULL,
 `no_of_tickets` int(50) NOT NULL,
 `userbook_id` varchar(10) NOT NULL,
  Foreign Key(`userbook_id`) REFERENCES `user`(`user_id`) ON DELETE
CASCADE ON UPDATE CASCADE,
  Foreign Key(`show_id`) REFERENCES `show`(`show_id`) ON DELETE CASCADE ON
UPDATE CASCADE,
   Primary Key(`booking_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `booking`
--
INSERT INTO `booking` VALUES
('B001','SHT0140002',2,'100'),
('B003','SHT0220002',1,'105'),
('B004','SHT0310002',3,'102'),
('B005','SHT0330002',2,'103'),
('B006','SHT0110003',4,'101'),
('B007','SHT0220002',1,'106'),
('B008','SHT0120003',3,'107'),
('B009','SHT0110003',4,'111'),
('B010','SHT0240002',2,'112'),
('B011','SHT0140002',1,'110');
--
-- Table structure for table `ticket`
--

CREATE TABLE `ticket` (
 `ticket_id` varchar(10) NOT NULL,
  `booking_id` varchar(10) NOT NULL,
 `type` varchar(50) NOT NULL,
 `total_price` int NOT NULL,
  Primary Key(`ticket_id`),
  Foreign Key(`booking_id`) REFERENCES `booking`(`booking_id`) ON DELETE
CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
--
-- Dumping data for table `ticket`
--
```

```sql
INSERT INTO `ticket` VALUES
('T100','B001','classic',300),
('T102','B003','premium',350),
('T103','B004','recliner',315),
('T104','B005','classic',275),
('T105','B006','classic',395),
('T106','B007','premium',315),
('T107','B008','premium',325),
('T108','B009','premium',450),
('T109','B010','recliner',495),
('T110','B004','recliner',500);
```

JOIN QUERIES

1)RETRIEVE THEATRE NAMES WITH THE CORRESPONDING HALL ID'S AND THE TOTAL NUMBER OF SEATS IN THE HALL

```
SELECT
   theatre.theatre_name,
   hall.hall_id,
   (
      hall.classic_seat_no + hall.premium_seat_no + hall.recliner_seat_no
   ) AS total_seats
FROM
   theatre
JOIN hall WHERE theatre.theatre_id = hall.theatre_id;
```

Showing rows 0 - 10 (11 total, Query took 0.0004 seconds.)

```
SELECT theatre.theatre_name, hall.hall_id, ( hall.classic_seat_no +
hall.premium_seat_no + hall.recliner_seat_no ) AS total_seats FROM theatre JOIN hall
WHERE theatre.theatre_id = hall.theatre_id;
```

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

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: Nor

Extra options

theatre_name	hall_id	total_seats
PVR Cinemas	T014	85
PVR Cinemas	T015	112
INOX Movies	T021	107
INOX Movies	T022	105
INOX Movies	T023	108
Cinepolis	T006	99
Cinepolis	T008	99
Cinepolis	T010	110
IMAX Cinemas	T003	113
IMAX Cinemas	T012	101
IMAX Cinemas	T020	115

2)DISPLAY THE NAME OF THE MOVIE AND THE TIME IT STARTS

```
SELECT
   movie.movie_name,`show`.start_time
FROM
   movie
NATURAL JOIN `show`
   ;
```

Showing rows 0 - 9 (10 total, Query took 0.0005 seconds.)

```
SELECT movie.movie_name,`show`.start_time FROM movie NATURAL JOIN `show`;
```

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

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: Nor

Extra options

movie_name	start_time
Black Panther	07:30:00
Maze Runner: The Death Cure	07:30:00
Kurukshetra	04:20:00
Kantara	04:20:00
Hichki	04:20:00
Pacific Rim Uprising	04:20:00
Strangers : Prey at night	04:20:00
Tomb Raider	04:20:00
Midnight Sun	04:20:00
Peter Rabbit	04:20:00

3)RETRIEVE SHOW ID'S OF ONLY THOSE SHOWS WHERE THERE ARE SEATS REMAINING TO BE BOOKED

SELECT
*
FROM
total
NATURAL JOIN booked WHERE total_remaining < total_seats

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0028 seconds.)

```
CREATE VIEW total AS SELECT ( classic_seat_no + premium_seat_no + recliner_seat_no ) AS total_seats, show_id FROM
hall, `show` WHERE `show`.hall_id = hall.hall_id;
```

[Edit inline] [Edit] [Create PHP code]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0005 seconds.)

```
CREATE VIEW booked AS SELECT ( classic_remaining + premium_remaining + recliner_remaining ) AS total_remaining,
show_id FROM `show`;
```

[Edit inline] [Edit] [Create PHP code]

Showing rows 0 - 9 (10 total, Query took 0.0006 seconds.)

```
SELECT * FROM total NATURAL JOIN booked WHERE total_remaining < total_seats;
```

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

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

Extra options

show_id	total_seats	total_remaining
SHT0330002	113	37
SHT0120003	99	31
SHT0310002	99	50
SHT0110003	110	23
SHT0320002	101	20
SHT0140002	85	21
SHT0240002	115	32
SHT0210002	107	40
SHT0220002	105	16
SHT0230002	108	64

4)DISPLAY NAMES OF USERS WHO HAVE PURCHASED EXACTLY ONE TICKET

```
SELECT
   user_name
FROM
   `user`
JOIN booking WHERE no_of_tickets = 1 AND `user`.user_id = `booking`.`userbook_id`;
```

Showing rows 0 - 2 (3 total, Query took 0.0004 seconds.)

```
SELECT user_name FROM `user` JOIN booking WHERE no_of_tickets = 1 AND `user`.user_id =
`booking`.`userbook_id`;
```

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

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

user_name
Ajay
Vikram
Preeti

AGGREGATE FUNCTIONS

1)CALCULATE THE NUMBER OF TICKETS SOLD WHICH ARE OF THE TYPE ‘PREMIUM’

```
SELECT
   COUNT(ticket_id)
FROM
   ticket
WHERE TYPE
   = 'premium';
```

Your SQL query has been executed successfully.

```
SELECT COUNT(ticket_id) FROM ticket WHERE TYPE = 'premium';
```

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

Extra options

COUNT(ticket_id)
4

2)DISPLAY THE NAME OF THE THEATER WHICH MADE THE MOST NUMBER OF SALES AND ALSO CALCULATE THE TOTAL SALES MADE

```
CREATE VIEW most_sales_made AS(
   SELECT
      theatre.theatre_name,
      COUNT(ticket_id) AS ticket_sales
   FROM
      theatre,
      ticket,
      `show`,
      `booking`,
      hall
   WHERE
      theatre.theatre_id = hall.theatre_id AND `show`.hall_id = hall.hall_id AND `show`.show_id
= booking.show_id AND ticket.booking_id = booking.booking_id AND `show`.`date` =
'2022-11-11'
   GROUP BY
      theatre.theatre_name
);
```

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)

```
CREATE VIEW most_sales_made AS( SELECT theatre.theatre_name, COUNT(ticket_id) AS
ticket_sales FROM theatre, ticket, `show`, `booking`, hall WHERE theatre.theatre_id =
hall.theatre_id AND `show`.hall_id = hall.hall_id AND `show`.show_id = booking.show_id
AND ticket.booking_id = booking.booking_id AND `show`.`date` = '2022-11-11' GROUP BY
theatre.theatre_name );
```

[Edit inline] [Edit] [Create PHP code]

```
SELECT
   theatre_name,
   ticket_sales
FROM
   most_sales_made
WHERE
   ticket_sales =(
   SELECT
      MAX(ticket_sales)
   FROM
      most_sales_made
);
```

Showing rows 0 - 0 (1 total, Query took 0.0013 seconds.)

```
SELECT theatre_name, ticket_sales FROM most_sales_made WHERE ticket_sales =( SELECT
MAX(ticket_sales) FROM most_sales_made );
```

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

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

Extra options

theatre_name	ticket_sales
IMAX Cinemas	20

3)CALCULATE THE AVERAGE PRICE OF TICKETS

```
SELECT
   AVG(total_price)
FROM
   ticket;
```

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

```
select avg(total_price) from ticket;
```

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

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

Extra options

avg(total_price)
372.0000

4)FIND THE NUMBER OF MOVIES WHICH ARE IN ENGLISH

```
SELECT
   COUNT(*)
FROM
   movie
WHERE
   movie.language = 'English'
```

Your SQL query has been executed successfully.

```
SELECT COUNT(*) FROM movie WHERE movie.language = 'English';
```

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

Extra options

COUNT(*)
8

5)DISPLAY BLOCKBUSTER MOVIES OF THE DAY WHICH IS THE MOVIES WHICH HAVE THE HIGHEST BOOKINGS IN A DAY

```
CREATE VIEW blockbuster_movie AS(
   SELECT
      (m.movie_name) AS name_of_movie,
      COUNT(b.booking_id) AS no_of_bookings
   FROM
      booking AS b,
      `show` AS s,
      movie AS m
   WHERE
      s.show_id = b.show_id AND m.movie_id = s.movie_id
   GROUP BY
      m.movie_name
);
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)

```
CREATE VIEW blockbuster_movie AS( SELECT (m.movie_name) AS name_of_movie,
COUNT(b.booking_id) AS no_of_bookings FROM booking AS b, `show` AS s, movie AS m WHERE
s.show_id = b.show_id AND m.movie_id = s.movie_id GROUP BY m.movie_name );
```

[Edit inline] [Edit] [Create PHP code]

```
SELECT
   name_of_movie AS blockbuster_movie_of_the_day
FROM
   blockbuster_movie
WHERE
   no_of_bookings =(
   SELECT
      MAX(no_of_bookings)
   FROM
      blockbuster_movie
);
```

Showing rows 0 - 2 (3 total, Query took 0.0010 seconds.)

```
SELECT name_of_movie AS blockbuster_movie_of_the_day FROM blockbuster_movie WHERE
no_of_bookings =( SELECT MAX(no_of_bookings) FROM blockbuster_movie );
```

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

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

Extra options

blockbuster_movie_of_the_day
Black Panther
Hichki
Kurukshetra

SET OPERATIONS

1)LIST OUT ALL THE MOVIES WHICH ARE IN ENGLISH AND KANNADA

```
SELECT
   movie_name,language
FROM
   movie
WHERE
   movie.language = 'English'
UNION
SELECT
    movie_name,language
FROM
   movie
WHERE
   movie.language = 'Kannada';
```

Showing rows 0 - 11 (12 total, Query took 0.0004 seconds.)

```
SELECT movie_name,language FROM movie WHERE movie.language = 'English' UNION SELECT
movie_name,language FROM movie WHERE movie.language = 'Kannada';
```

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

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: Nor

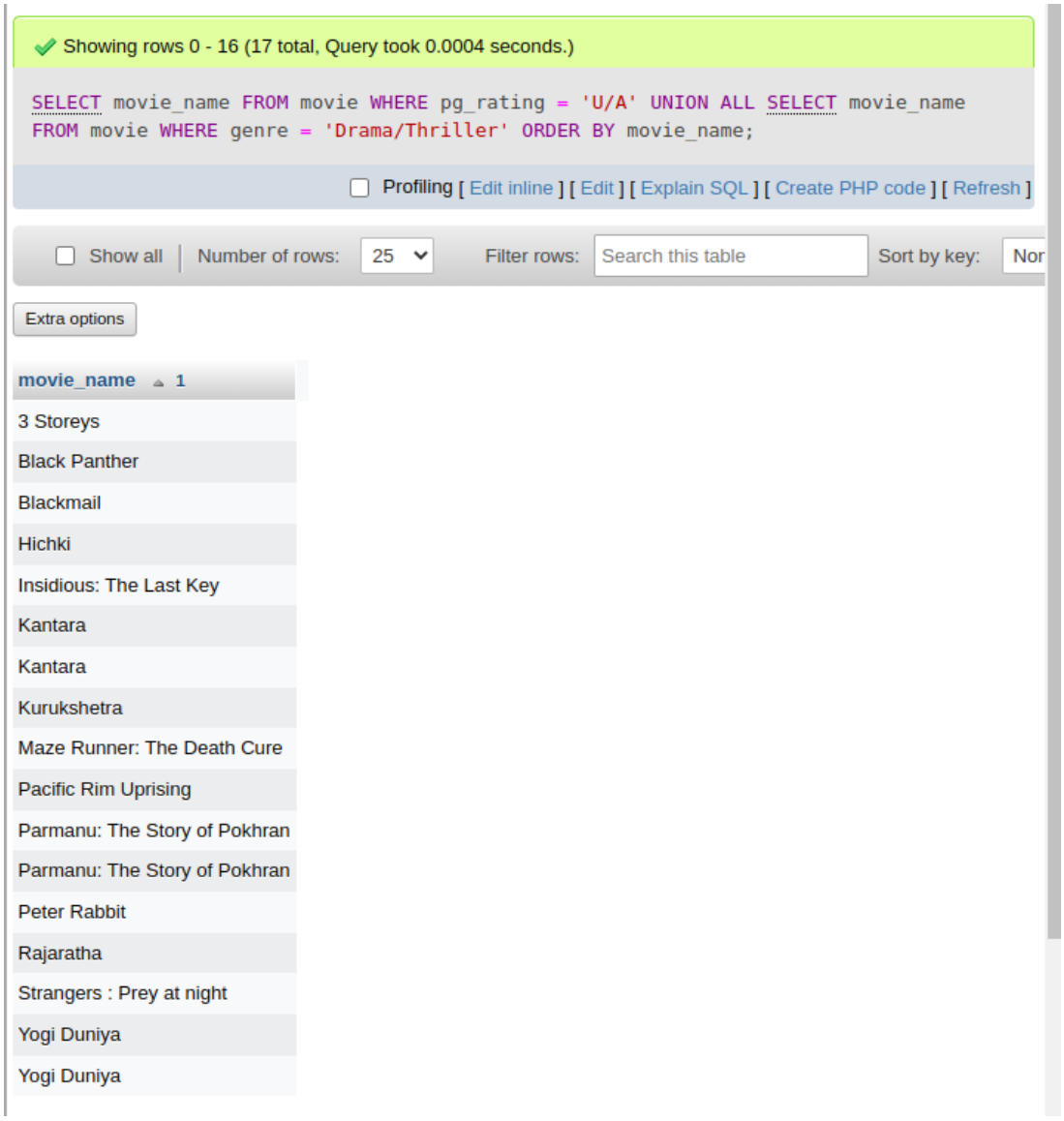
Extra options

movie_name	language
Pacific Rim Uprising	English
Strangers : Prey at night	English
Tomb Raider	English
Midnight Sun	English
Peter Rabbit	English
Black Panther	English
Maze Runner: The Death Cure	English
Insidious: The Last Key	English
Rajaratha	Kannada
Yogi Duniya	Kannada
Kurukshetra	Kannada
Kantara	Kannada

2)LIST THE MOVIES WHICH BELONG TO THE GENRE OF DRAMA/THRILLER OR HAVE A PG_RATING OF U/A

```
SELECT
   movie_name
FROM
   movie
WHERE
   pg_rating = 'U/A'
UNION ALL
SELECT
   movie_name
FROM
   movie
WHERE
   genre = 'Drama/Thriller’
```

ORDER BY movie_name;



3)DISPLAY THE NAMES OF USERS AND THEIR USER ID WHO HAVE PURCHASED MORE THAN 2 TICKETS

```
SELECT
   user_name,
   booking.userbook_id
FROM
   `user`,
   booking
WHERE
   `user`.user_id = booking.userbook_id AND EXISTS(
   SELECT
```

```
    booking.userbook_id
  FROM
    booking
  WHERE
    no_of_tickets > 2
);
```

Showing rows 0 - 9 (10 total, Query took 0.0004 seconds.)

```
SELECT user_name, booking.userbook_id FROM `user`, booking WHERE `user`.user_id =
booking.userbook_id AND EXISTS( SELECT booking.userbook_id FROM booking WHERE
no_of_tickets > 2 );
```

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

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

Extra options

user_name	userbook_id
Amit	100
Raghav	101
Anjali	102
Joy	103
Ajay	105
Vikram	106
Komal	107
Preeti	110
Shreya	111
Aditya	112

4)FIND ALL THE THEATRE NAMES WHICH HAVE NUMBER OF CLASSIC SEATS BETWEEN 85 AND 40

```
SELECT
  theatre_name,
  hall. classic_seat_no
FROM
```

```
    theatre,
    hall
WHERE
    theatre.theatre_id = hall.theatre_id OR classic_seat_no < 85 AND NOT EXISTS(
    SELECT
        hall. classic_seat_no
    FROM
        hall
    WHERE
        classic_seat_no > 40
);
```

Showing rows 0 - 10 (11 total, Query took 0.0004 seconds.)

```
SELECT theatre_name, hall. classic_seat_no FROM theatre, hall WHERE
theatre.theatre_id = hall.theatre_id OR classic_seat_no < 85 AND NOT EXISTS( SELECT
hall. classic_seat_no FROM hall WHERE classic_seat_no > 40 );
```

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

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

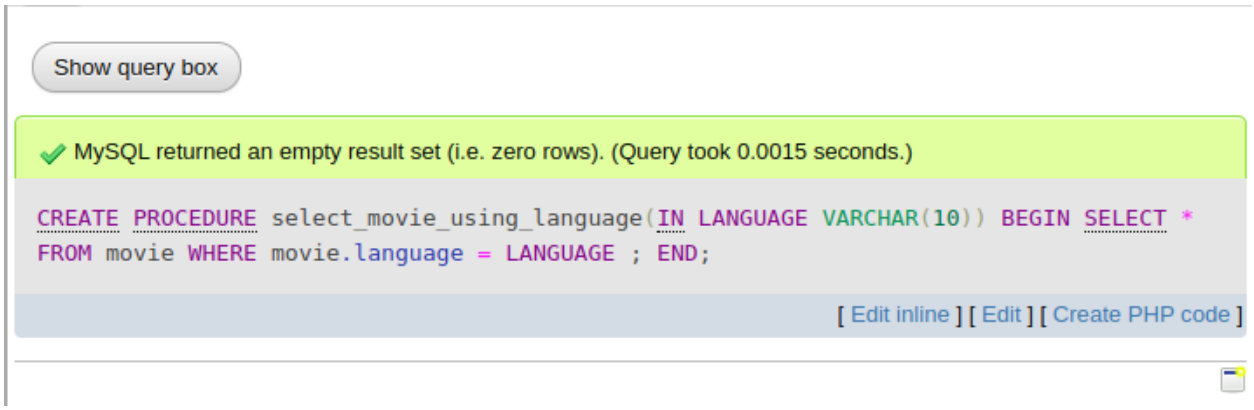
Extra options

theatre_name	classic_seat_no
PVR Cinemas	60
PVR Cinemas	92
INOX Movies	72
INOX Movies	60
INOX Movies	78
Cinepolis	69
Cinepolis	43
Cinepolis	80
IMAX Cinemas	81
IMAX Cinemas	56
IMAX Cinemas	75

FUNCTIONS AND PROCEDURES

1)PROCEDURE:
CREATE A STORED PROCEDURE TO SELECT A MOVIE USING THE USER SPECIFIED LANGUAGE

```
DELIMITER
   $$
CREATE PROCEDURE select_movie_using_language(IN LANGUAGE VARCHAR(10))
BEGIN
   SELECT
      *
   FROM
      movie
   WHERE
      movie.language = LANGUAGE ;
END $$
DELIMITER
   ;
```



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

```
call select_movie_using_language('Hindi');
```

[Edit inline] [Edit] [Create PHP code]

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

Extra options

movie_id	movie_name	language	genre	duration	pg_rating
001	Hichki	Hindi	Drama/Comedy	2 hrs 15 mins	U/A
010	Blackmail	Hindi	Comedy	1 hrs 55 mins	U/A
011	Parmanu: The Story of Pokhran	Hindi	Drama/Thriller	3 hrs 10 mins	U/A
012	3 Storeys	Hindi	Drama	1 hrs 45 mins	U/A

```
CREATE VIEW movie_fan AS(
   SELECT DISTINCT
      u.user_name,
      COUNT(b.booking_id) AS no_of_bookings
   FROM
      `user` AS u,
      booking AS b
   WHERE
      u.user_id = b.userbook_id
   GROUP BY
      u.user_name
);
```

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0012 seconds.)

```
CREATE VIEW movie_fan AS( SELECT DISTINCT u.user_name, COUNT(b.booking_id) AS
no_of_bookings FROM `user` AS u, booking AS b WHERE u.user_id = b.userbook_id GROUP BY
u.user_name );
```

[Edit inline] [Edit] [Create PHP code]

2)FUNCTION
WRITE A FUNCTION THAT DISPLAYS THE TOTAL NUMBER OF SEATS BOOKED FOR A GIVEN SHOW

```
DELIMITER $$
CREATE FUNCTION totalseatsbooked
(show_id varchar(10))
RETURNS INT(10)
BEGIN
        DECLARE seats INT(10);
   select (total.total_seats - booked.total_remaining) into seats
   from total,booked
   where show_id=show_id;
   RETURN seats;
END
$$
DELIMITER;
```

TRIGGERS AND CURSORS

1)CURSOR

CREATE A STORED PROCEDURE THAT CREATES A PHONE NUMBER LIST OF ALL USERS IN THE USER DATABASE

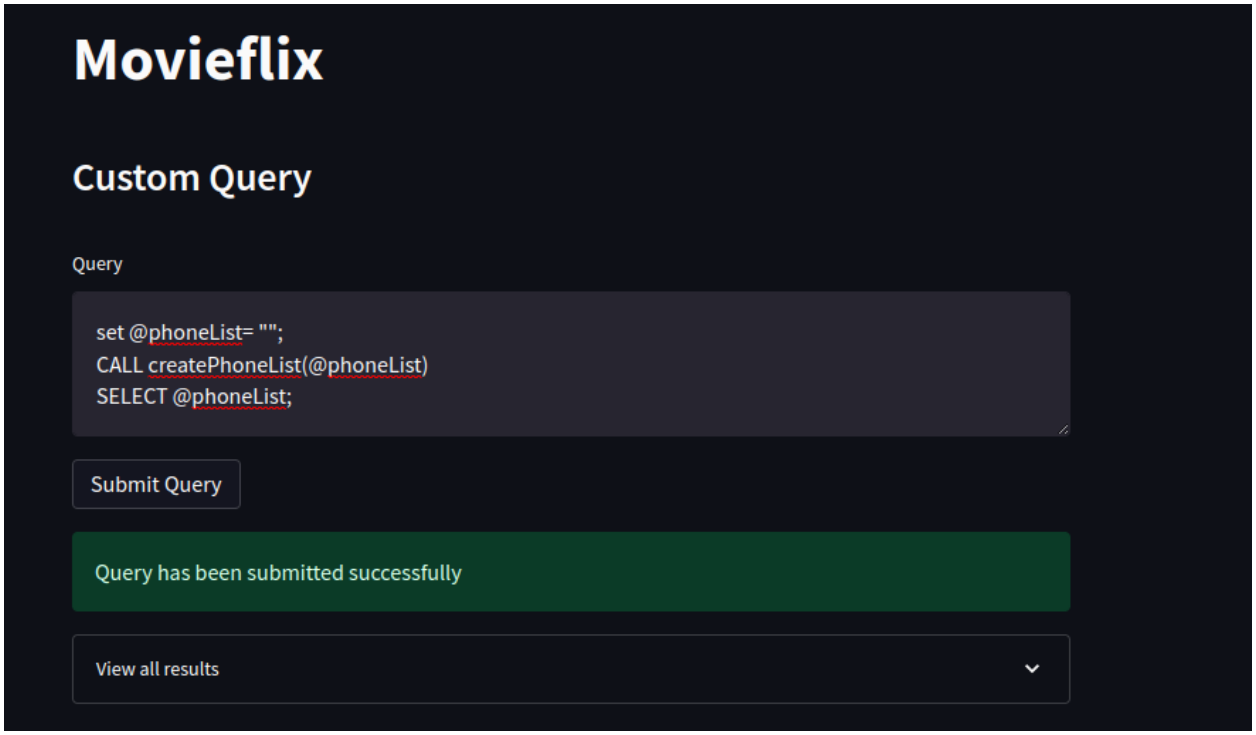
```
DELIMITER $$
CREATE PROCEDURE createPhoneList (
	INOUT phoneList varchar(4000)
)
BEGIN
	DECLARE finished INTEGER DEFAULT 0;
	DECLARE phoneno varchar(100) DEFAULT "";

	-- declare cursor for user phone
	DEClARE cur_phone
		CURSOR FOR
			SELECT phone FROM `user`;

	-- declare NOT FOUND handler
	DECLARE CONTINUE HANDLER
      FOR NOT FOUND SET finished = 1;

	OPEN cur_phone;

	getPhone: LOOP
		FETCH cur_phone INTO phoneno;
		IF finished = 1 THEN
			LEAVE getPhone;
		END IF;
		-- build phone list
		SET phoneList = CONCAT(phoneno,";",phoneList);
	END LOOP getPhone;
	CLOSE cur_phone;

END$$
DELIMITER ;
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0273 seconds.)

```
CREATE PROCEDURE createPhoneList ( INOUT phoneList varchar(4000) ) BEGIN DECLARE
finished INTEGER DEFAULT 0; DECLARE phoneno varchar(100) DEFAULT ""; -- declare
cursor for user phone DECLARE cur_phone CURSOR FOR SELECT phone FROM `user`; --
declare NOT FOUND handler DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;
OPEN cur_phone; getPhone: LOOP FETCH cur_phone INTO phoneno; IF finished = 1 THEN
LEAVE getPhone; END IF; -- build phone list SET phoneList =
```

[Edit inline] [Edit] [Create PHP code]

```
set @phoneList= "";
CALL createPhoneList(@phoneList)
SELECT @phoneList;
```



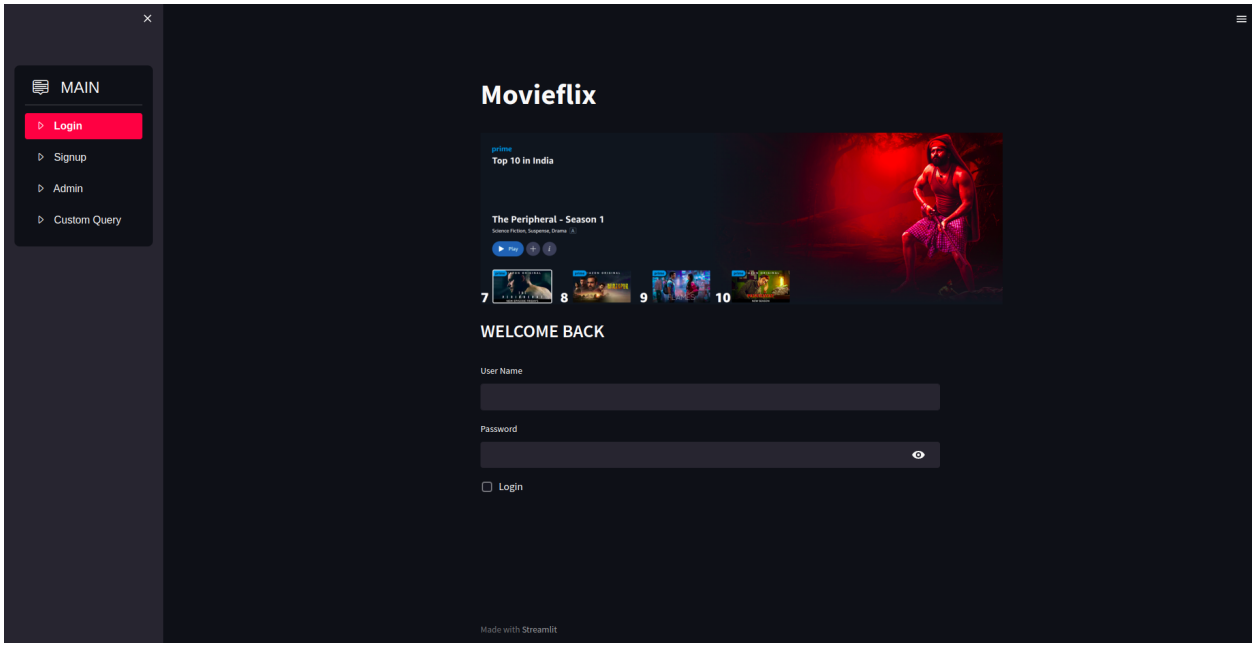
1)TRIGGER
WRITE A TRIGGER THAT ALERTS WHEN THE SEATS ARE FULL AND THERE ARE NONE REMAINING FOR A SHOW

```
create view total as select (classic_seat_no +premium_seat_no+recliner_seat_no) as
total_seats, show_id
from hall, `show `where `show`.hall_id = hall.hall_id;

create view booked as select (premium_remaining + classic_remaining+recliner_remaining) as
total_remaining, show_id from `show`;

DELIMITER $$
CREATE TRIGGER seats_full          BEFORE INSERT
ON booked
FOR EACH ROW
BEGIN
   DECLARE error_msg VARCHAR(255);
   DECLARE seat
   INT;
   SET error_msg = (" SEATS ARE FULL");
   SET seat = booked.total_remaining;
   IF seat=0
   THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = error_msg;
   END IF;
END $$
DELIMITER;
```

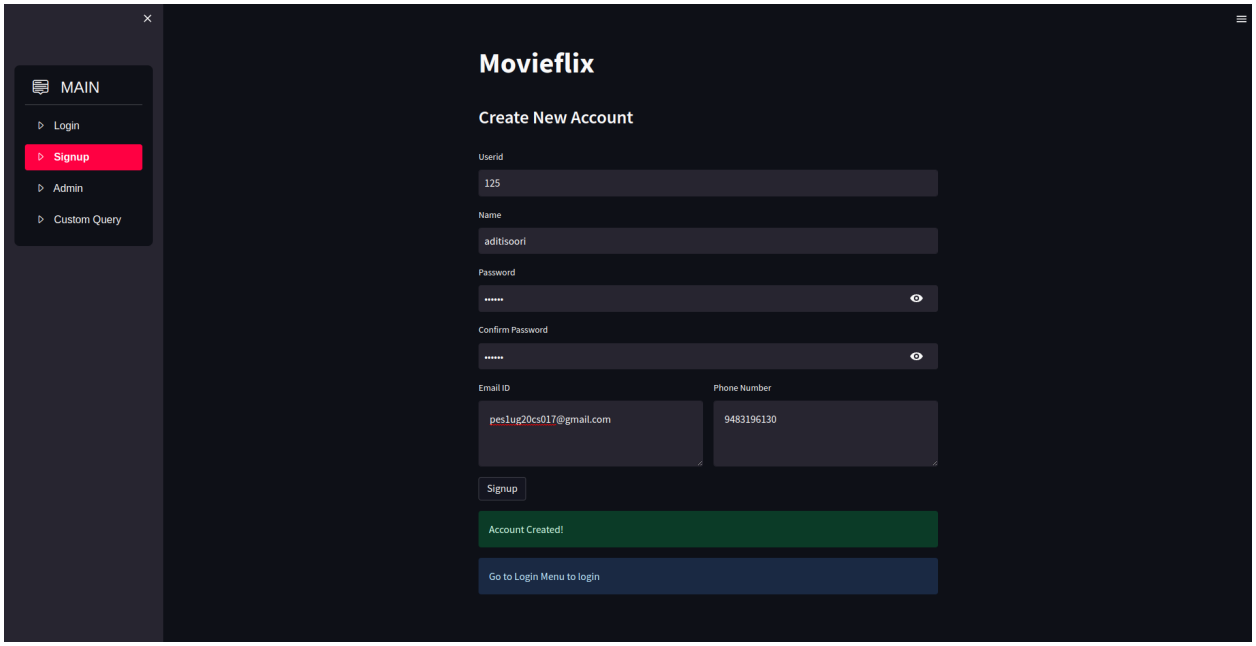
DEVELOPING THE FRONTEND

```
aditisoori@aditis-laptop:~/movie_management_system$ streamlit run /home/aditisoori
/movie_management_system/app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://172.16.21.168:8501
```

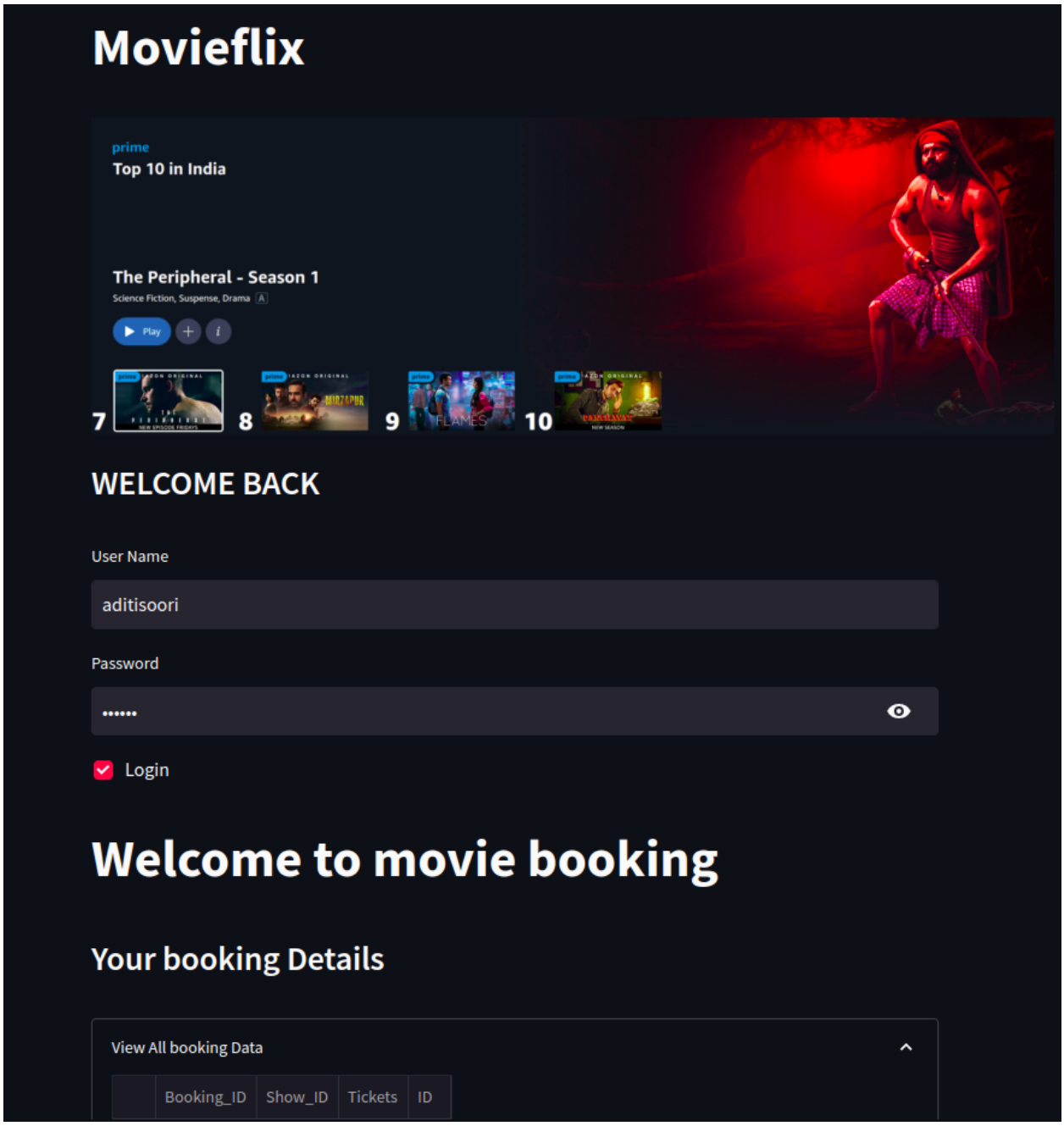
MAIN PAGE



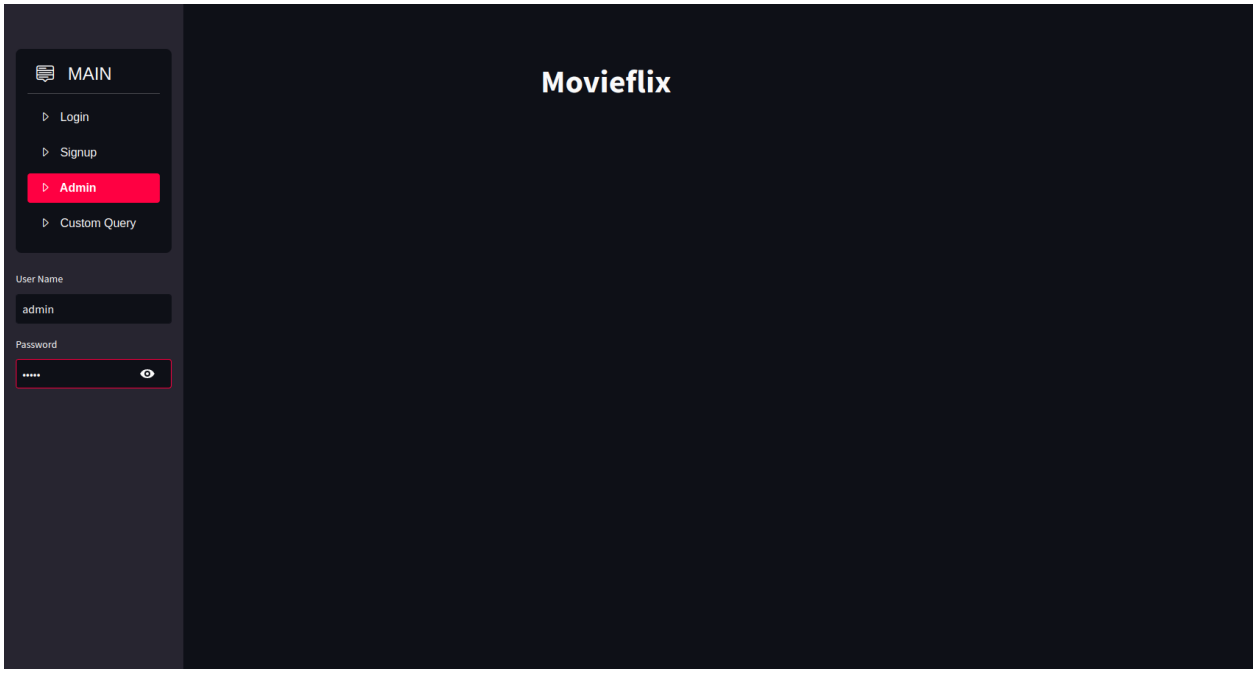
SIGNUP PAGE -CREATING A NEW USER


MAIN
Login
Signup
Admin
Custom Query
Movieflix
Create New Account
Userid
125
Name
aditisoori
Password
Confirm Password
Email ID
pes1ug20cs017@gmail.com
Phone Number
9483196130
Signup
Account Created!
Go to Login Menu to login

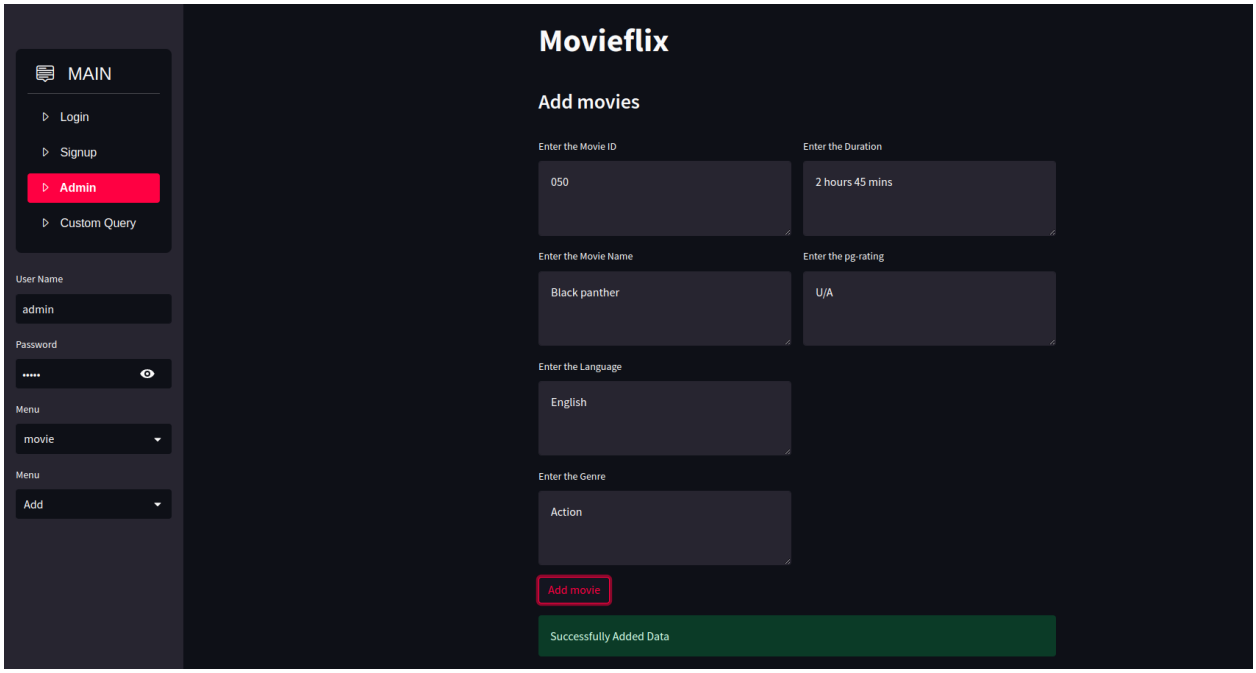
LOGIN PAGE


Movieflix
prime
Top 10 in India
The Peripheral - Season 1
Science Fiction, Suspense, Drama
Play
7
8
9
10
WELCOME BACK
User Name
aditisoori
Password
Login
Welcome to movie booking
Your booking Details
View All booking Data
Booking_ID
Show_ID
Tickets
ID

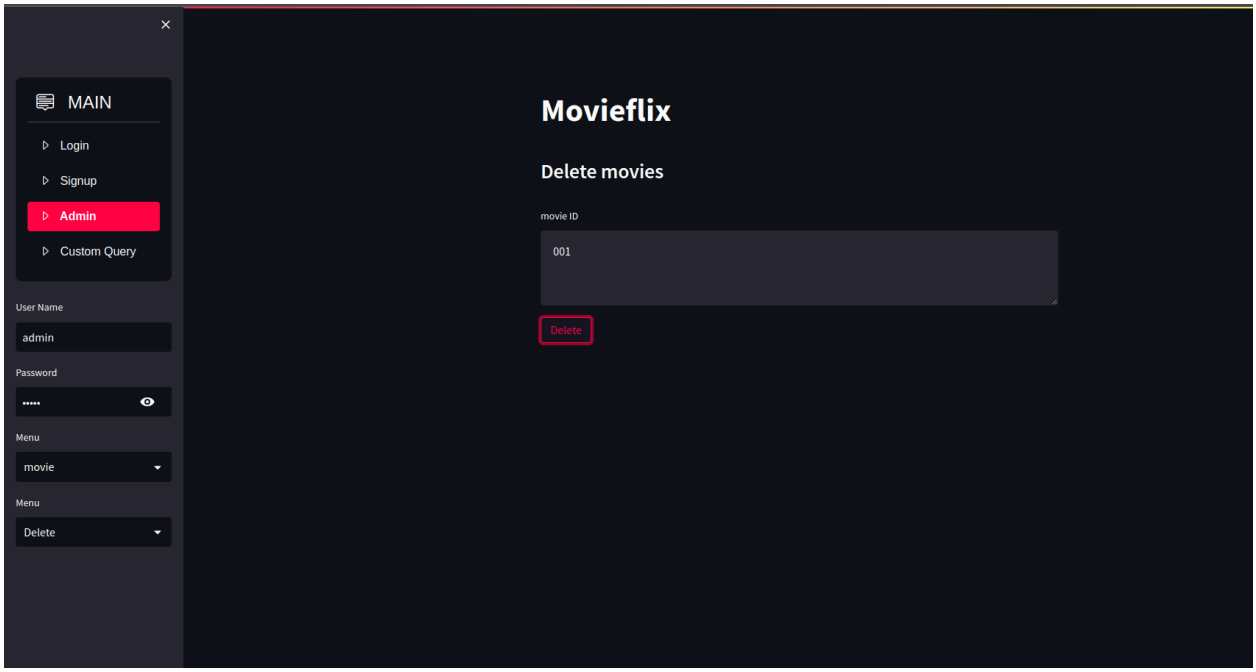
ADMIN LOGIN



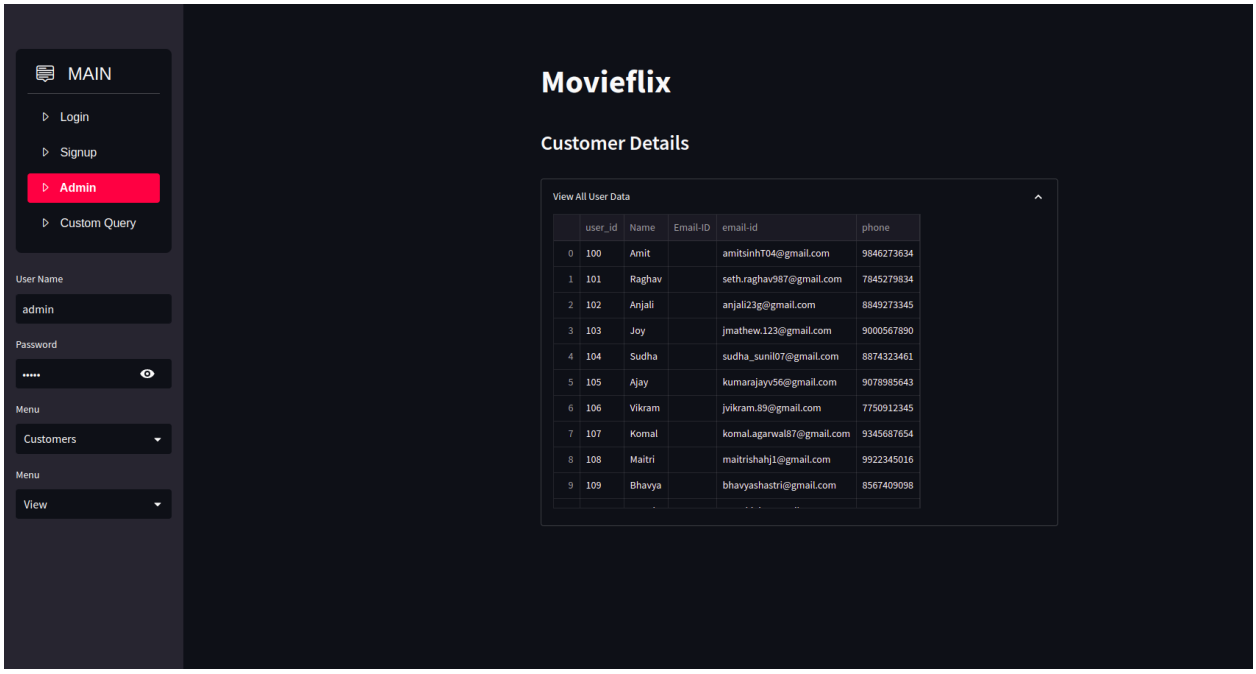
ADD MOVIE



DELETE MOVIE

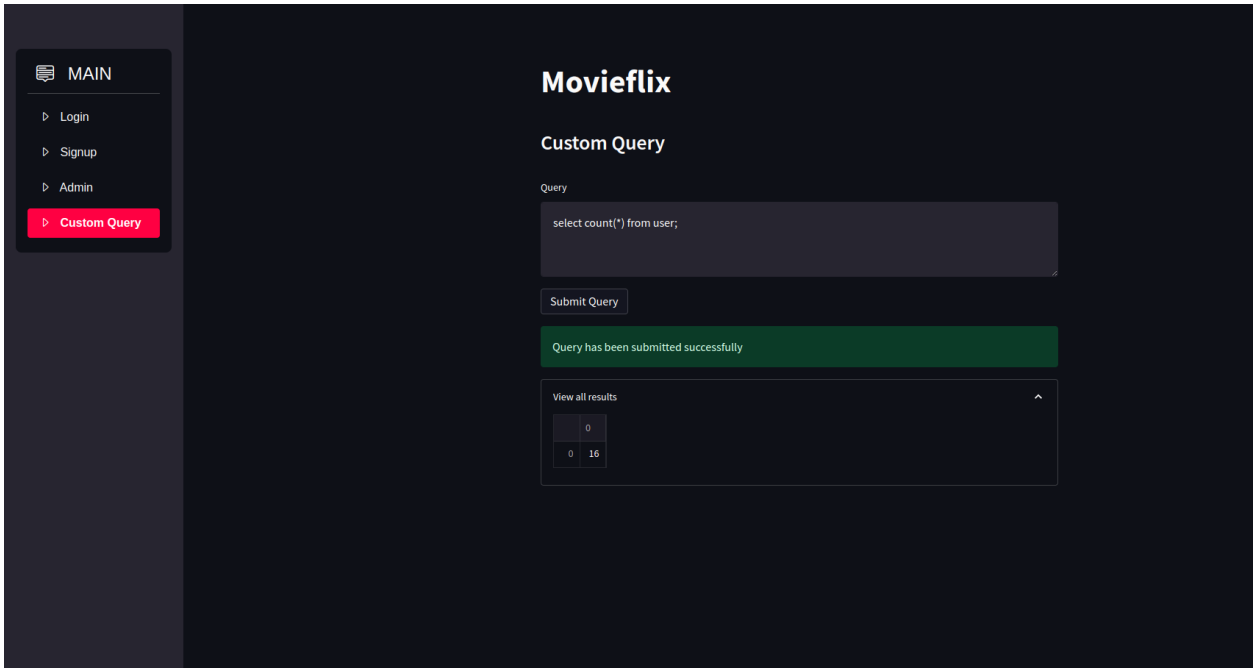


VIEW USER INFO



	user_id	Name	Email-ID	email-id	phone
0	100	Amit		amitsinhT04@gmail.com	9846273634
1	101	Raghav		seth.raghav987@gmail.com	7845279834
2	102	Anjali		anjali23g@gmail.com	8849273345
3	103	Joy		jmathew.123@gmail.com	9000567890
4	104	Sudha		sudha_sunil07@gmail.com	8874323461
5	105	Ajay		kumarajayv56@gmail.com	9078985643
6	106	Vikram		jvikram.89@gmail.com	7750912345
7	107	Komal		komal.agarwal87@gmail.com	9345687654
8	108	Maitri		maitrishahj1@gmail.com	9922345016
9	109	Bhavya		bhavyashastri@gmail.com	8567409098

CUSTOM QUERY PAGE


MAIN
Login
Signup
Admin
Custom Query
Movieflix
Custom Query
Query
select count(*) from user;
Submit Query
Query has been submitted successfully
View all results
0
0
16