Rangsit House ticket booking system

(ระบบการจองบัตรรับชมภาพยนตร์ออนไลน์ของโรงภาพยนตร์รังสิต เฮ้าส์)

Rathsiri Chintaworn,[1] Arnon Sennual,[2] Pokpong Songmuang[3]

Abstract

This paper is to develop a Rangsit House ticket booking database system under the topic of the cinema booking system, which provides a website for a cinema hall that any internet user can access. This system is required users to log in before using the system and is needed a smartphone is needed for paying the tickets. The paper also describes the functional business requirements, sample data, software code, and user interface. Here we report an ER diagram including entities and attributes, table details in SQL language through MySQL server on the XAMPP, a control panel context diagram, and a data flow diagram to guide the functional design. For implementing the function of online cinema ticket booking management, the requirement function, overall structure, and designed software codes in detail following the diagrams. In addition, we constructed the user interface appearances through Figma and used HTML, CSS, PHP and JavaScript for coding the front end. Thus, the online booking ticket system is one of the alternative choices for those who cannot afford enough time to get their tickets before watching a movie. People can book tickets online at any time of day or night.

Keyword: Cinema booking system, Database system, SQL, User interface

1. Introduction

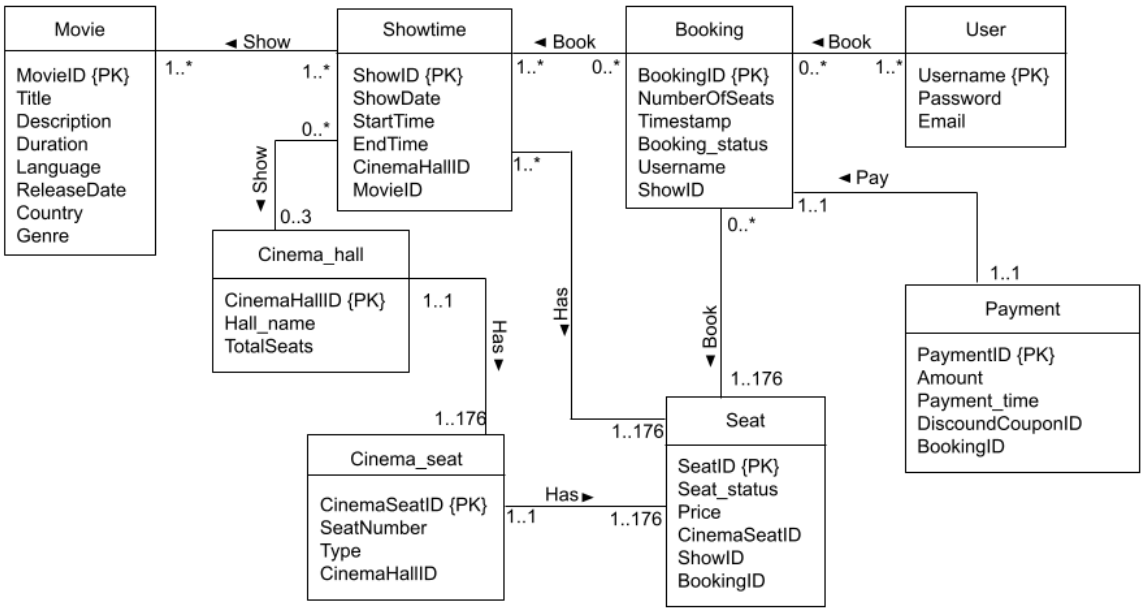
Nowadays, technology is a part of human life, and internet usage has become one of the most predominant things in daily life. People have changed their behavior to become more dependent on the internet since the pandemic of COVID-19 situation until these days. Likewise, movie theaters have to adapt themselves to rival changing consumer behavior. Due to the expanding market monopoly and increased activity of the online streaming system during the Covid-19 pandemic, Thailand's movie theater industry suffered greatly. Additionally, a few small cinema enterprises have shut down because of a lack of customers. The movie theater industry in Thailand is also highly competitive.

Thus, online platforms for purchasing movie tickets have emerged as another way to serve customers. When massive business owners might benefit from investing in internet systems for booking movie tickets, although small business owners can afford this system, this is because it is not worth considering the long-term system maintenance costs.

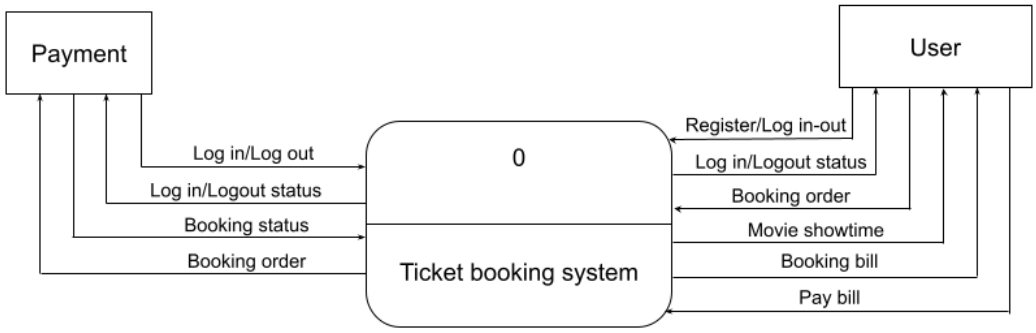
Therefore, we decided to develop an online movie ticket booking system for small cinema businesses from our knowledge to create the database system with SQL language and construct a sample user interface page with HTML, CSS, PHP, and JavaScript

2. Diagram and Design

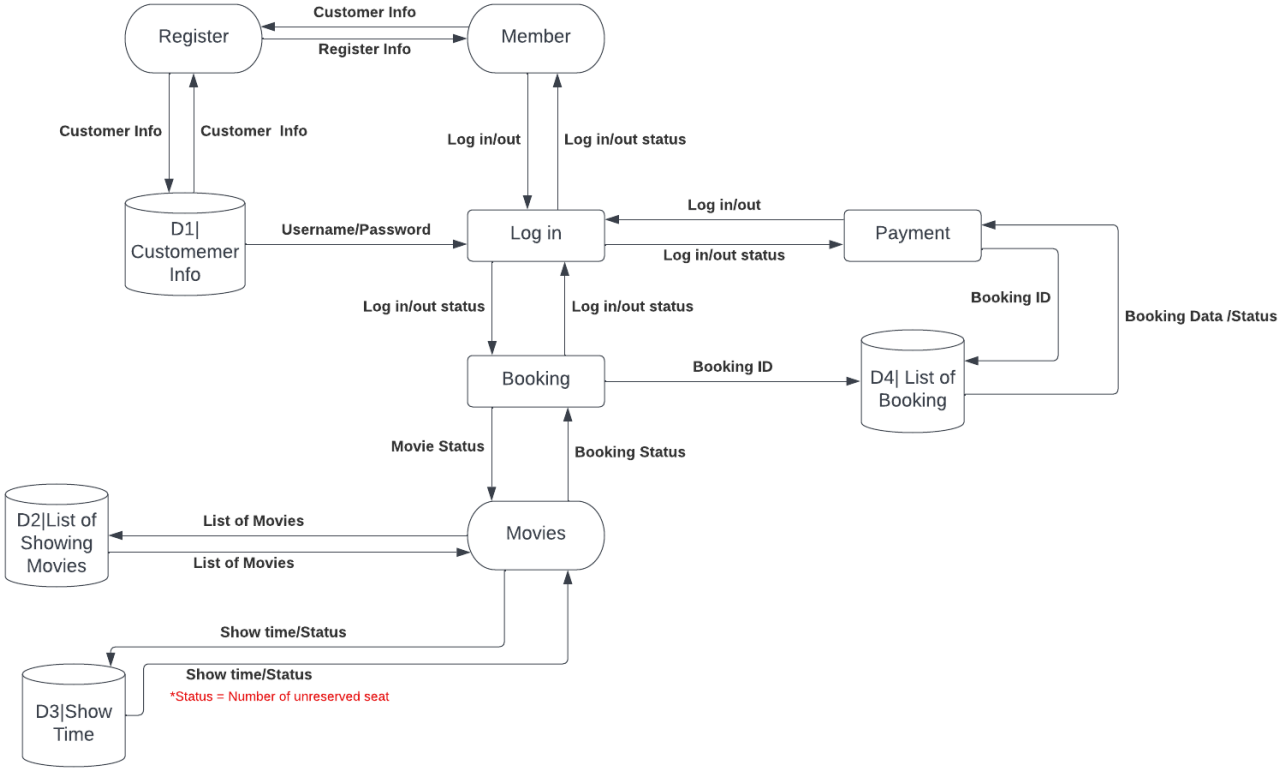
2.1 ER diagram



2.2 Context diagram

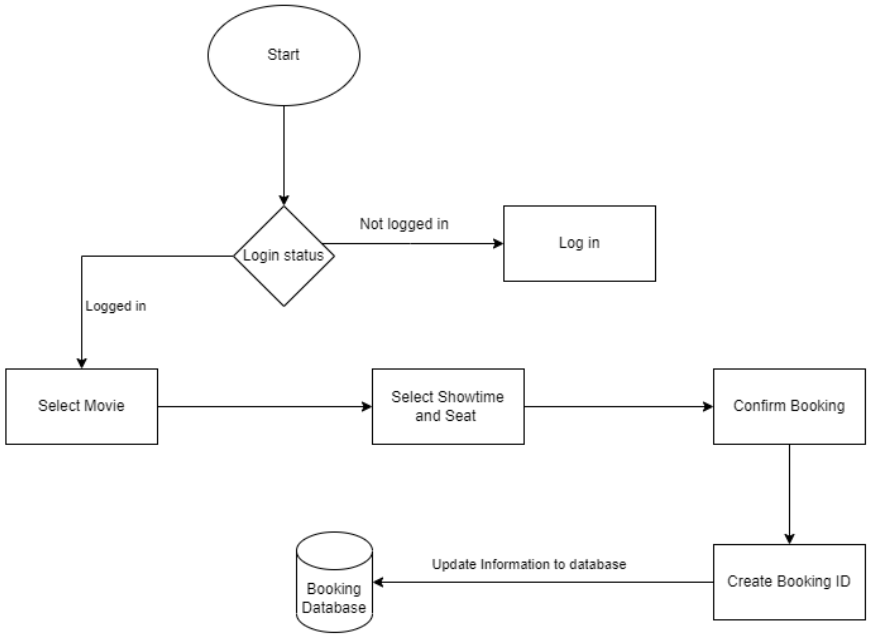


2.3 Data Flow diagram

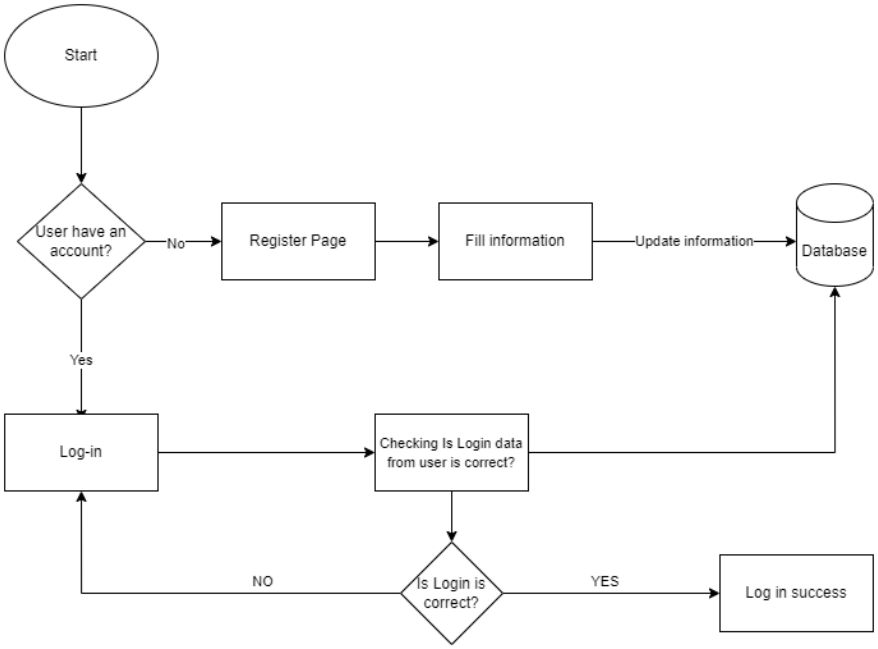


2.4 Functional design

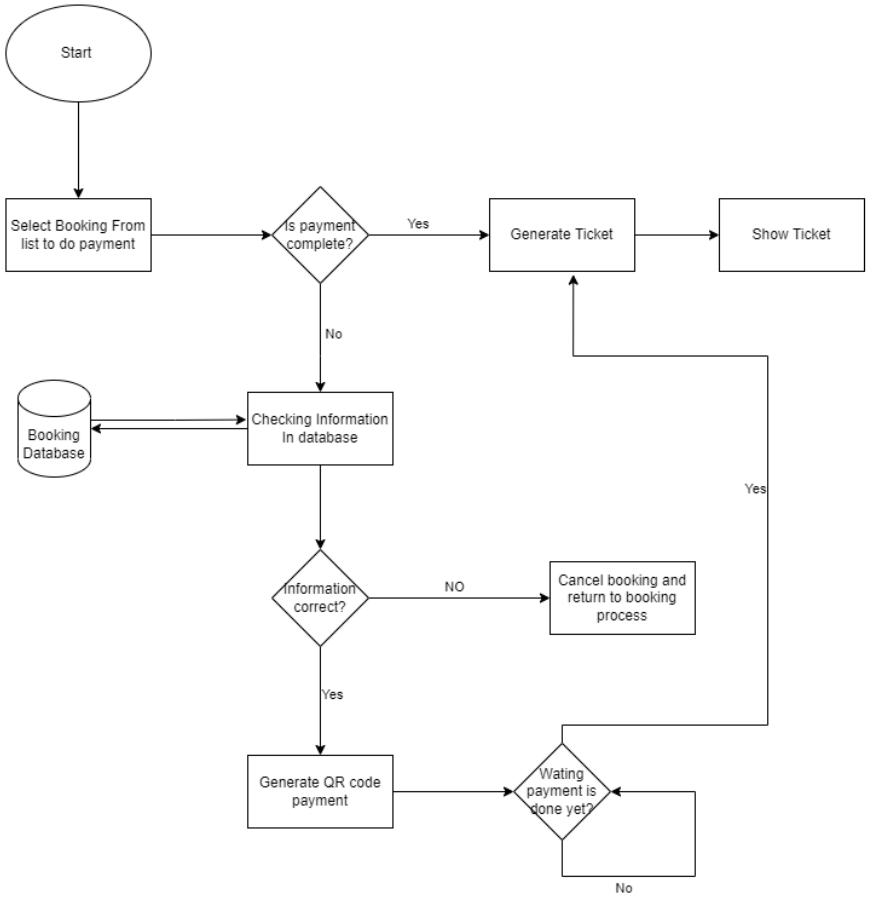
1. *Booking Flow chart*



2. *Login Flow chart*

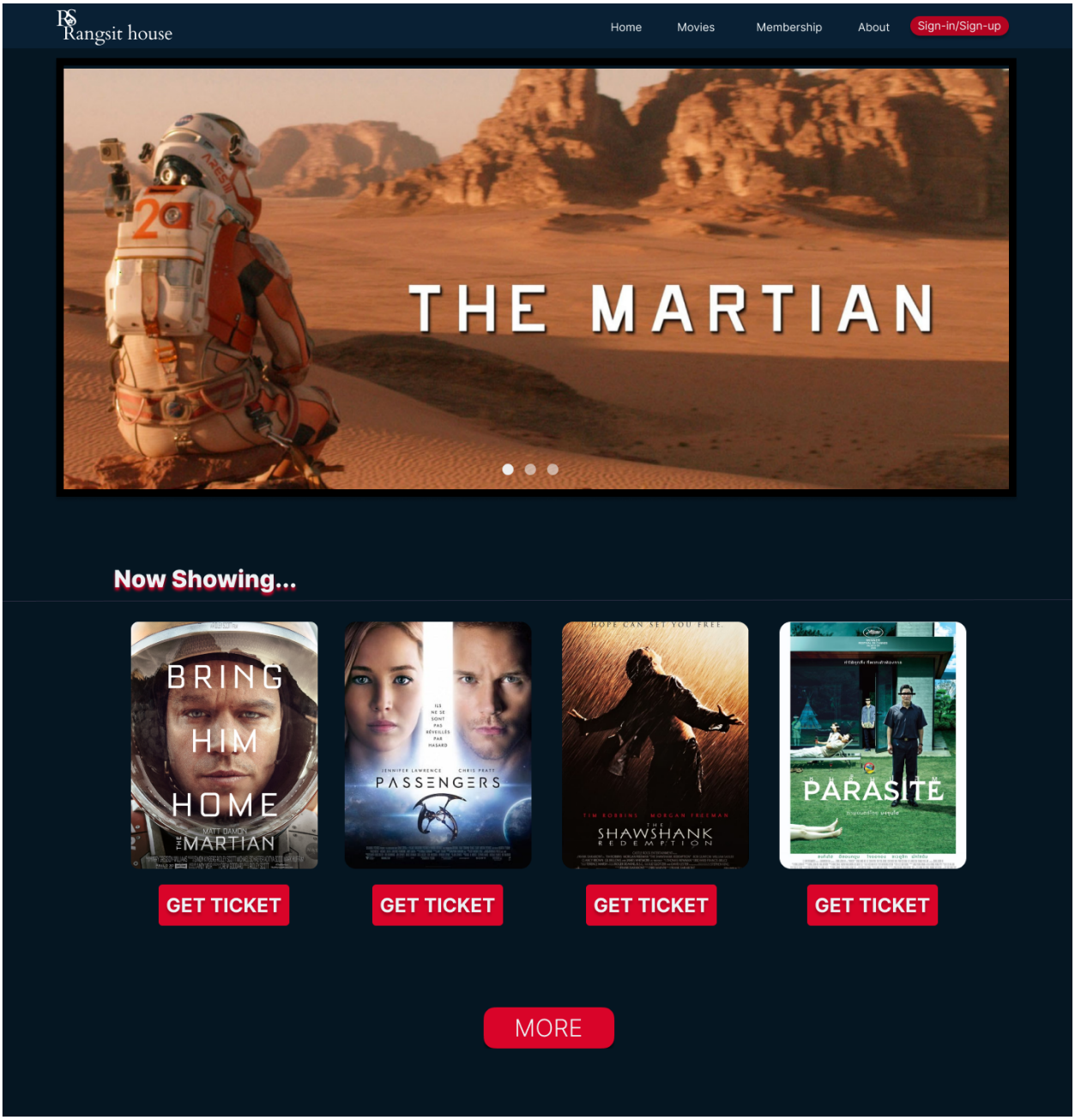


3. Payment Flow chart

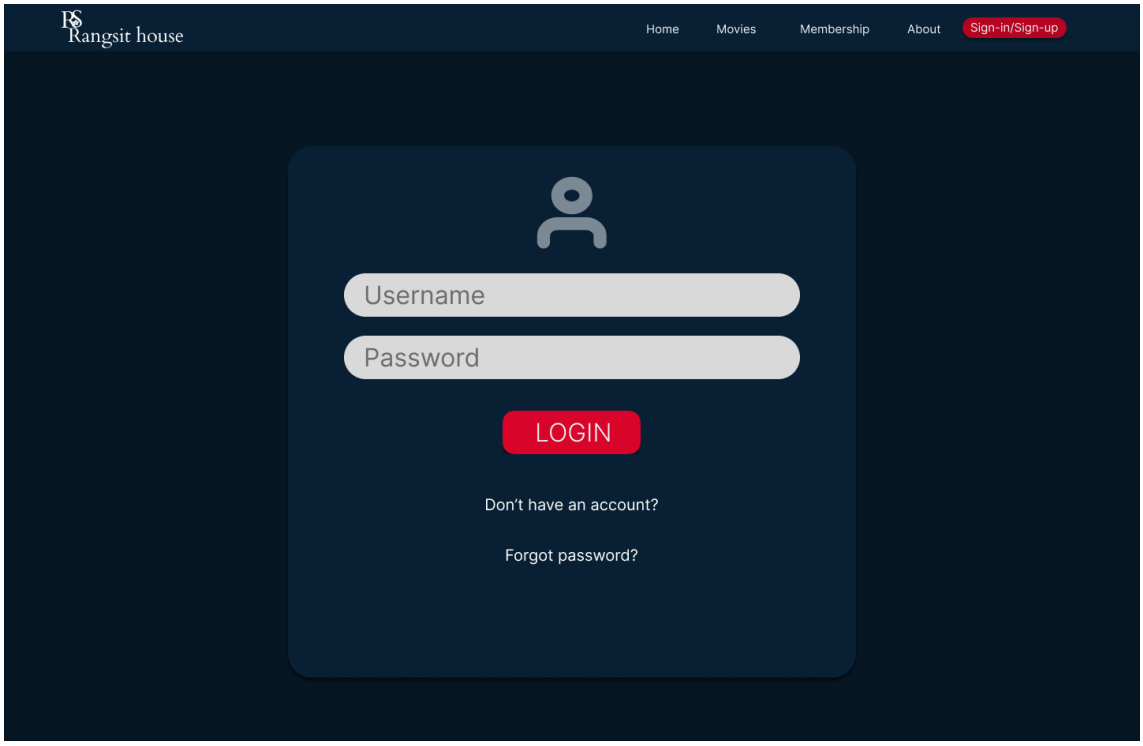


2.5 User interface

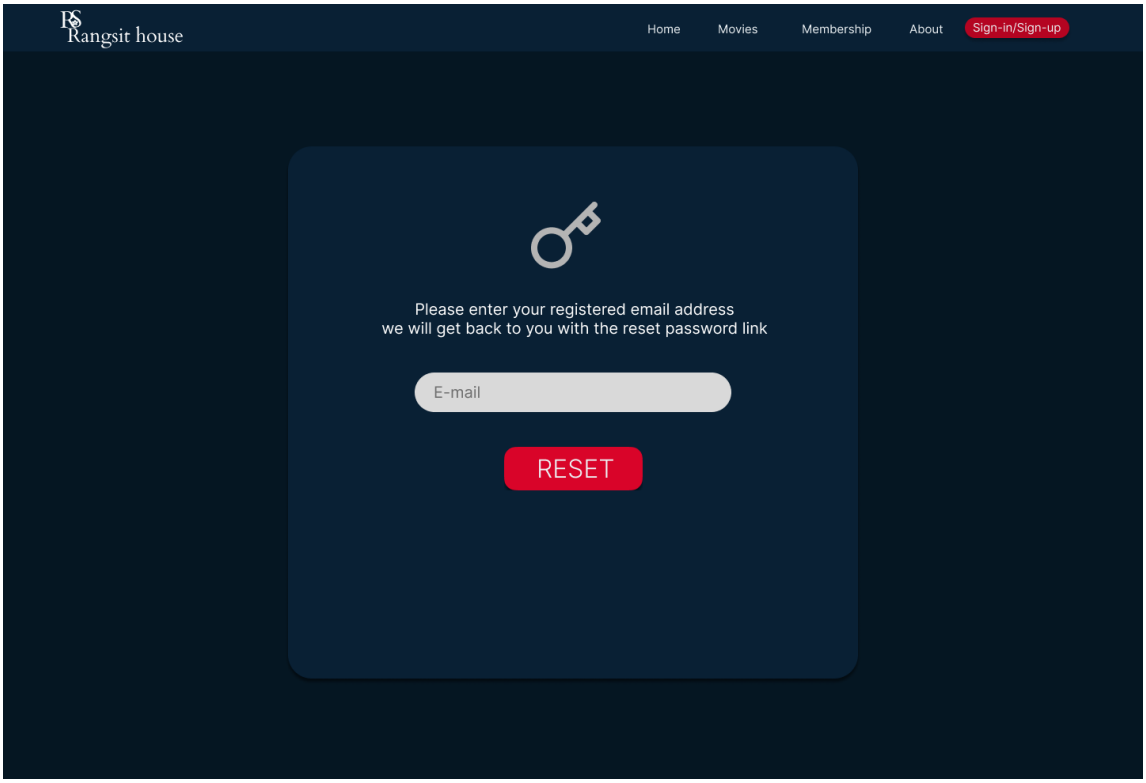
1. *Home page*



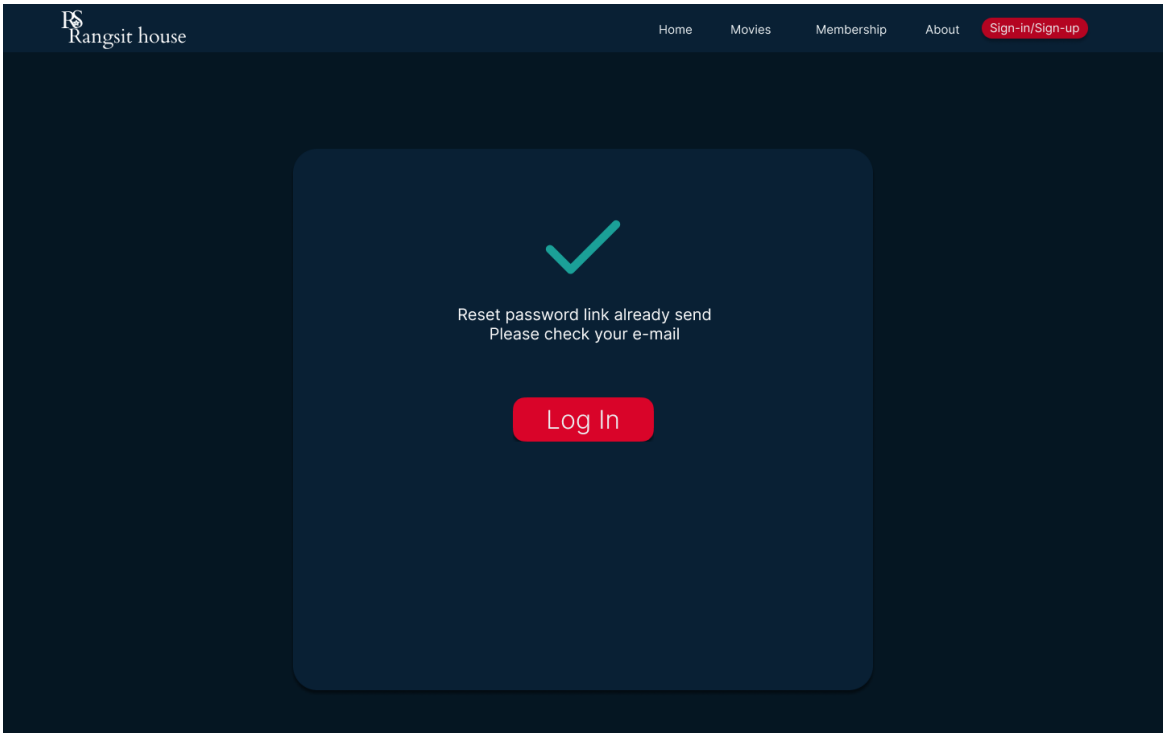
2. *Login page*



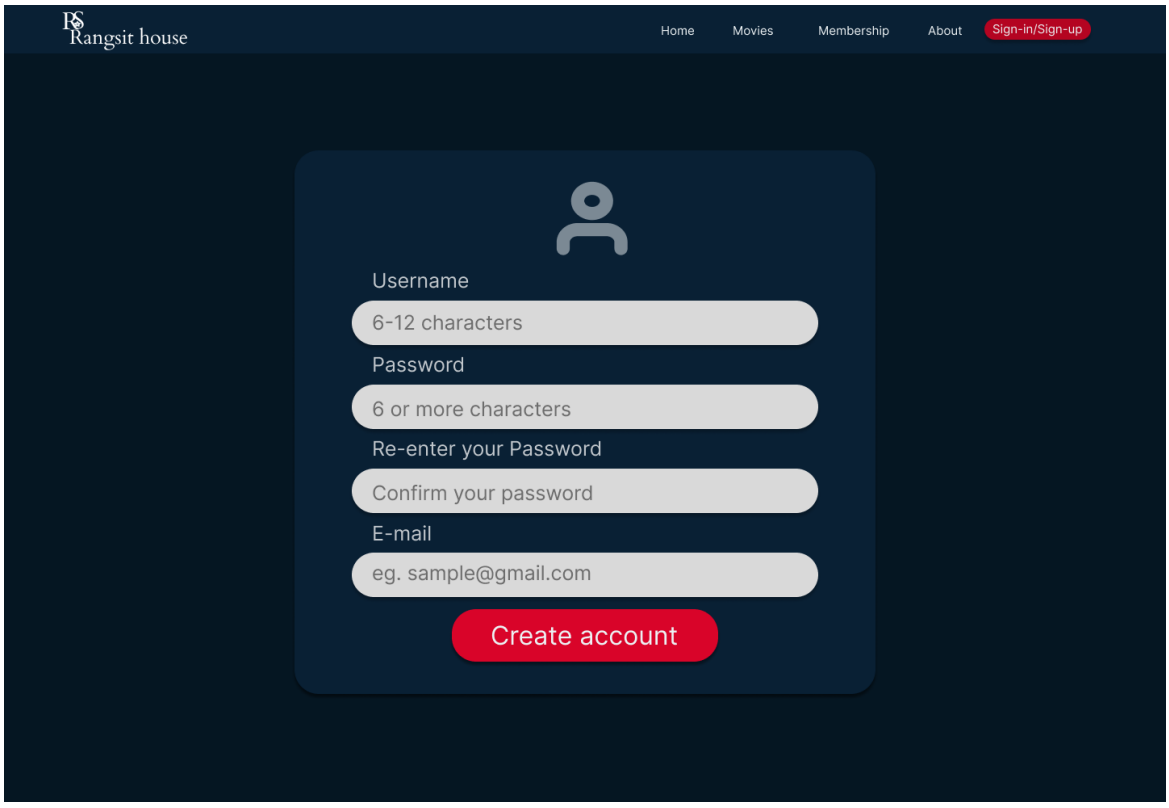
3. *Forget password page*



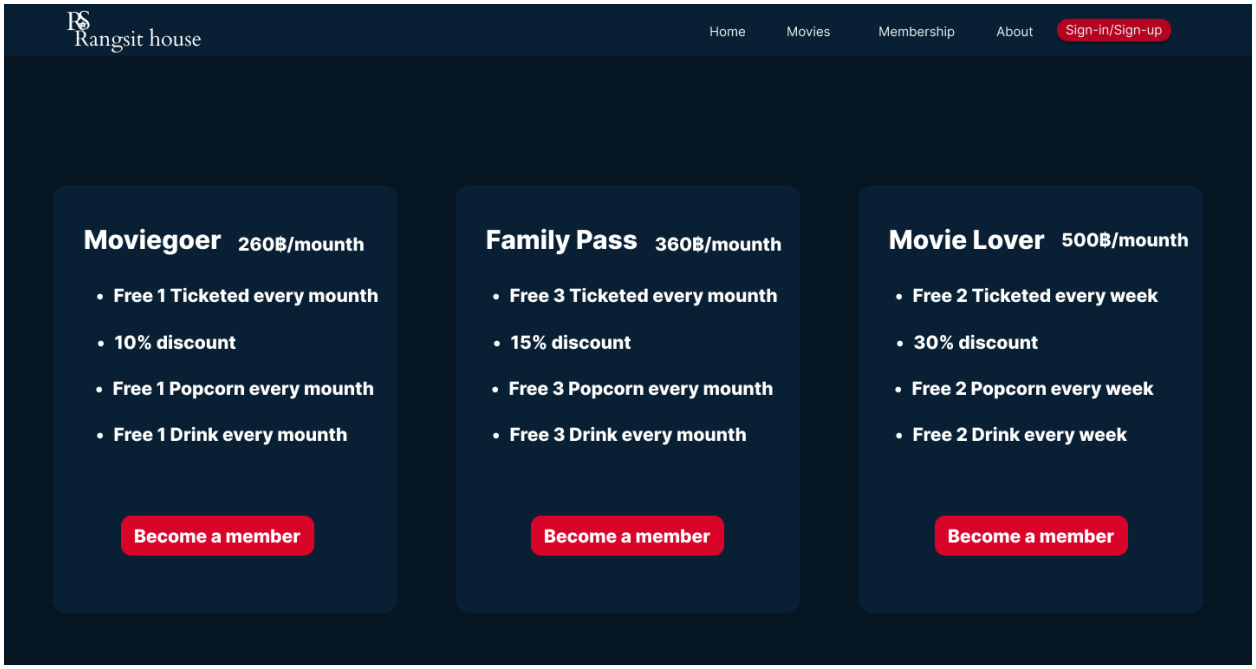
4. *Reset password submitted page*



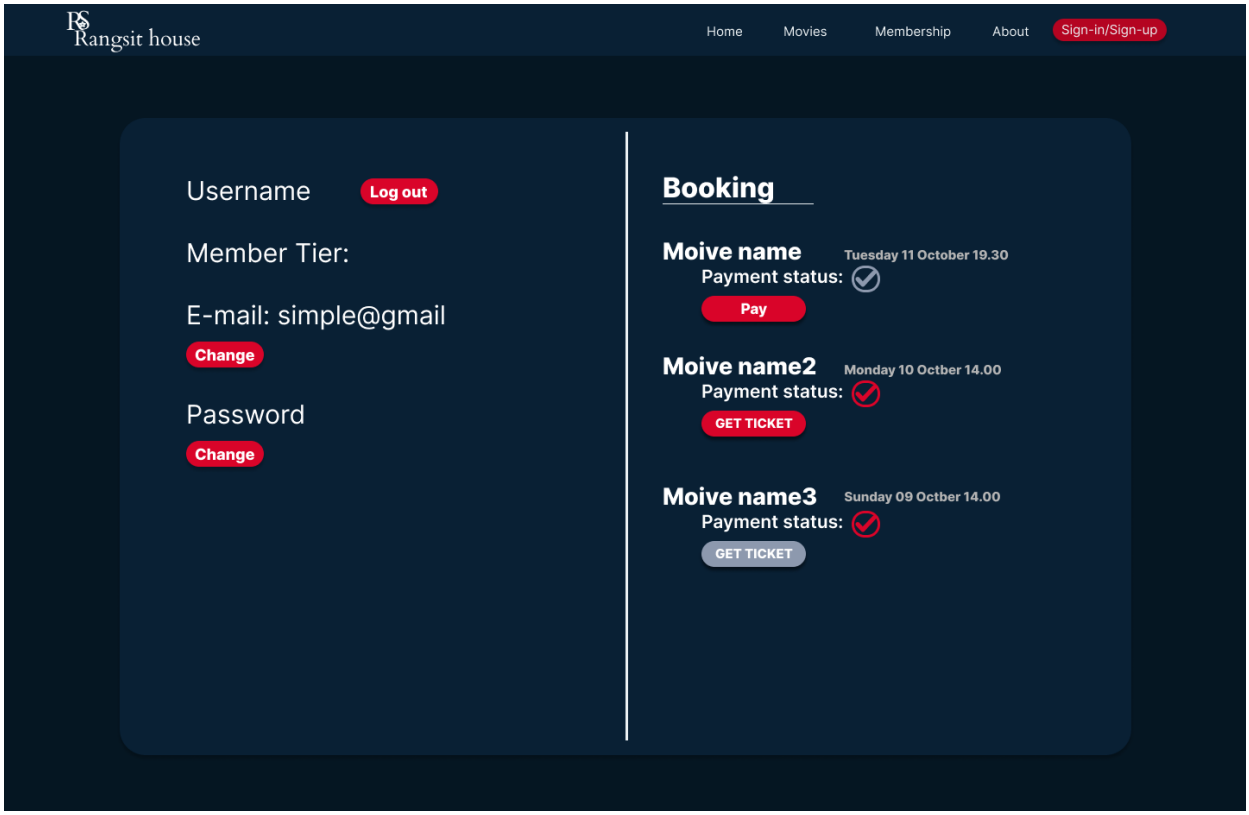
5. *Register page*



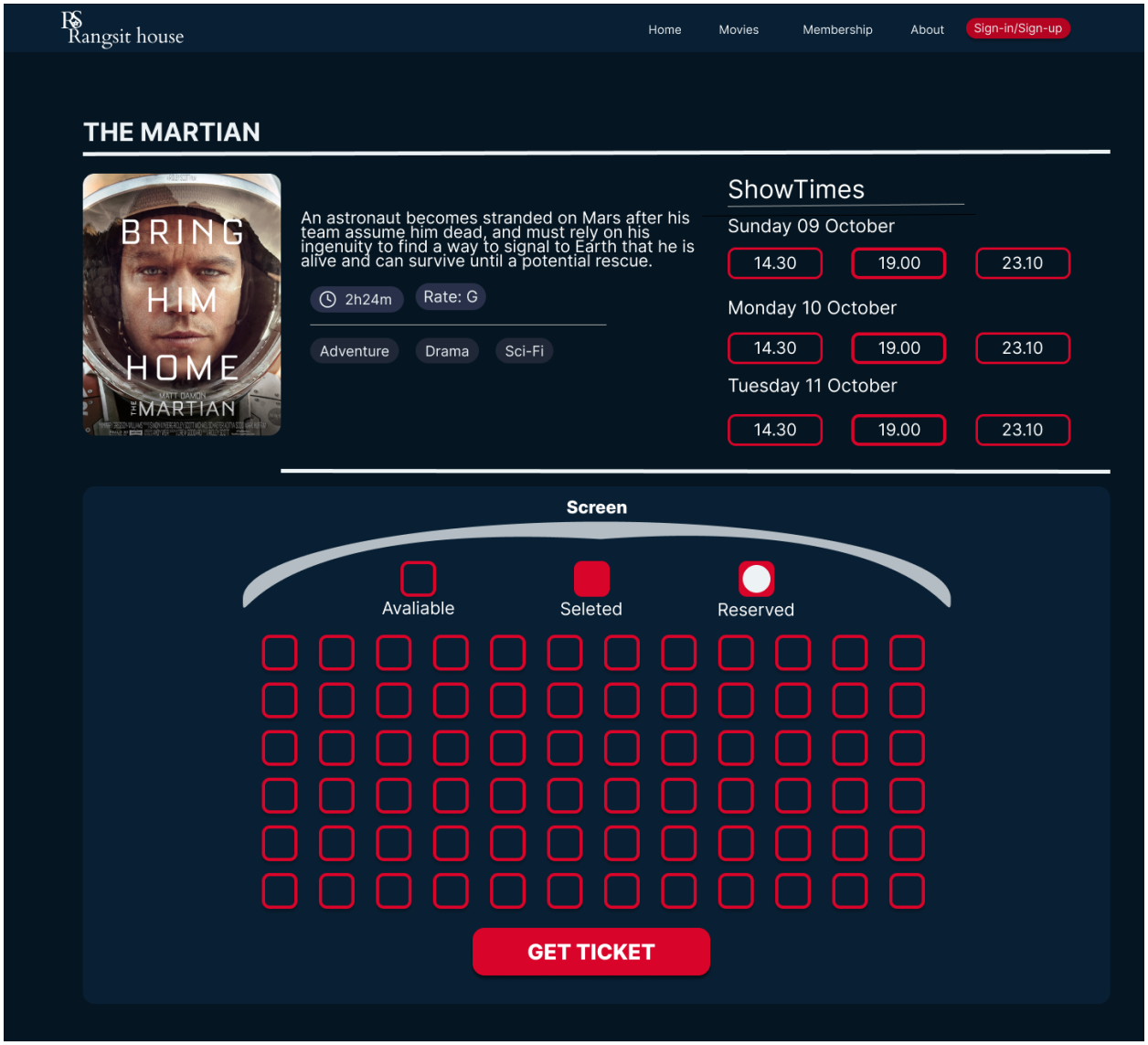
6. *Membership page*



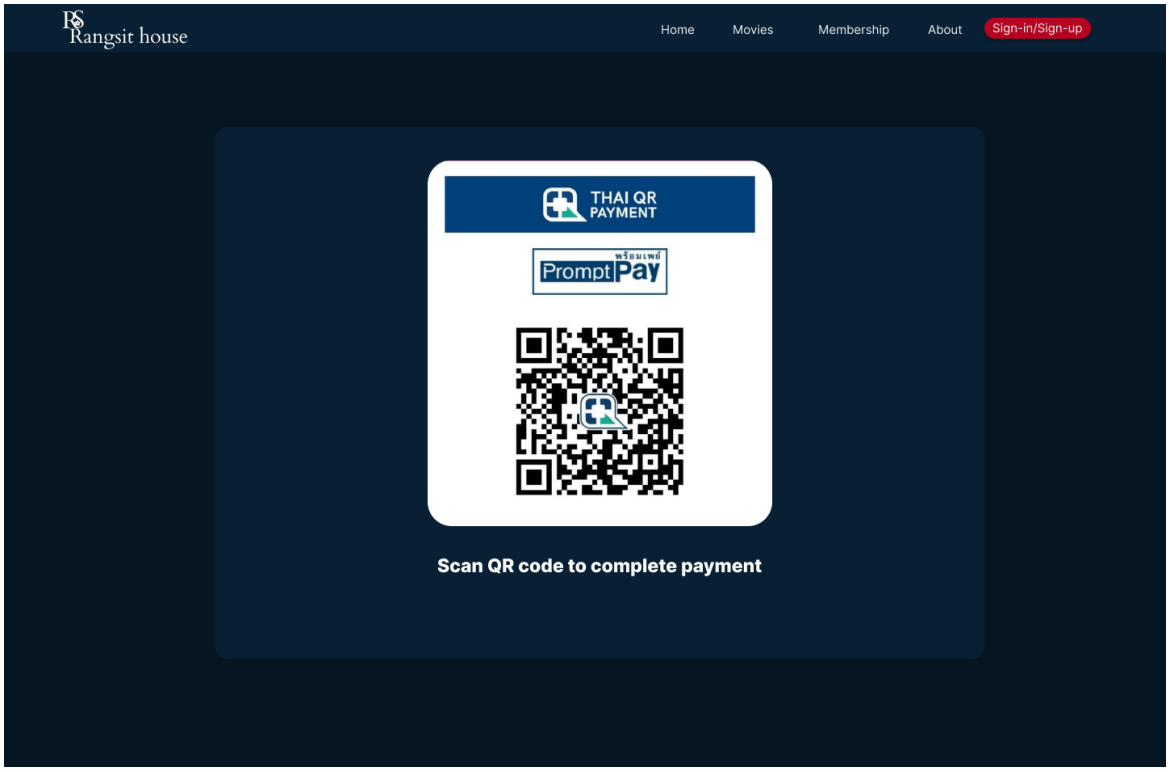
7. *Profile page*



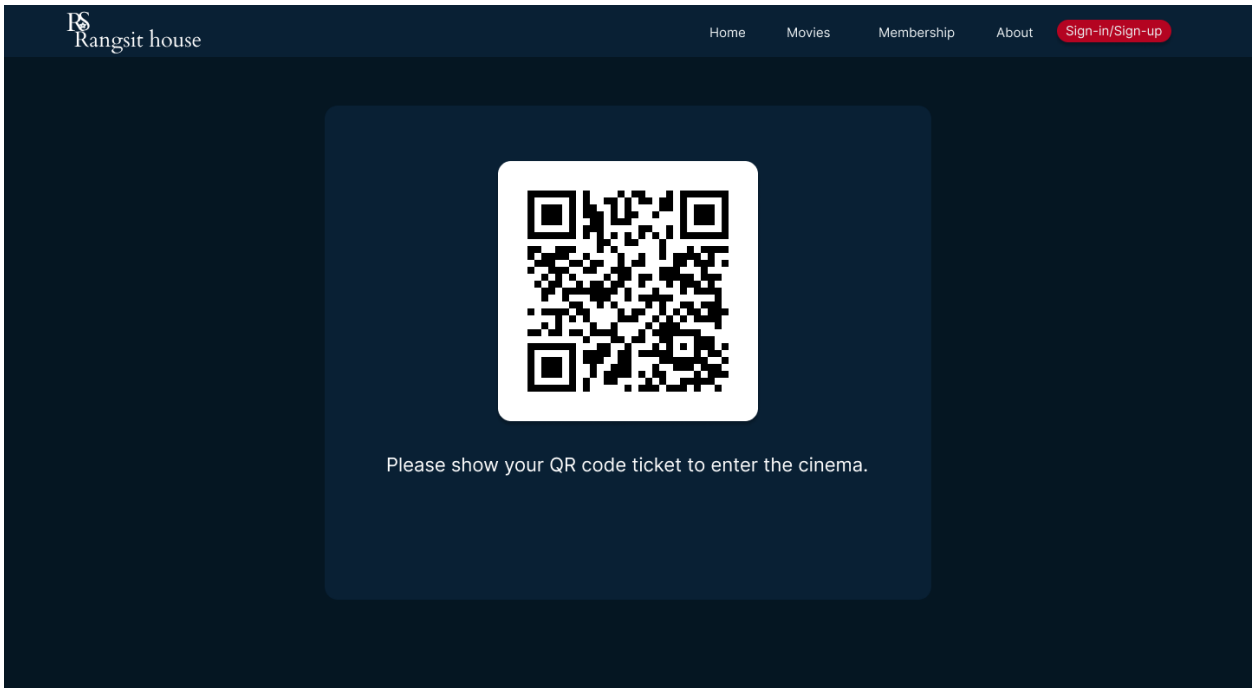
8. *Booking page*



9. Payment page



10. Get ticket page



3. Implementation on Database Management Systems

3.1 Implementation and Discussion

3.1.1 Home page

This page is used to query movie data and show it to the Home page as details.

```
$q = "SELECT * FROM movie LIMIT 5";
$result = $mysqli -> query($q);
while ($row = $result->fetch_array()) { ?>

    <div class="movie-card">
        <div class="movie-img">
            <img src="material/<?= $row['title'] ?>.jpg">
        </div>
        <div class="movie-detail-card">
            <div class="movie-detail-container">
                <div class="detail-container">
                    <a id="title"><span><?= $row['title'] ?></span></a>
                    <a>Rate: <?= $row['rate'] ?></a>
                    <div class="genre-detail">
                        <div class="genre-detail-1">
                            <?php
                            $genre = $row['genre'];
                            $genre_arr = explode(",", $genre);
                            foreach ($genre_arr as $genre_value) { ?>
                                <a><?= $genre_value ?></a>
                            <?php }
                            ?>
                        </div>
                    </div>
                    <a>Time: <?= ($row['duration']) ?></a>
                </div>
                <a href="buyticket_page.php?movie_id=<?= $row['movie_id'] ?>" style="align-self: center;"><button class="red-
            </div>
        </div>
    </div>
<?php }
?>
```

3.1.2 Profile page

First section of the profile page is used to check tickets in the database and show it. Also, use DATEDIFF function to check if it is passed or not and update it.

```
<?php
if(isset($_SESSION['user_id'])){
    require_once('connect.php');
    $user_id = $_SESSION['user_id'];
    $user_q = "SELECT *  FROM user WHERE user_id = $user_id;";
    $user_result = $mysqli -> query($user_q);
    $user_row = $user_result -> fetch_array();

    $ticket_q = "SELECT ticket_id,title,showdate,date_format(Start_time,'%H:%i') AS Start_time,status FROM ticket t
        INNER JOIN showtime st ON t.show_id = st.show_id
        INNER JOIN movie m ON st.movie_id = m.movie_id
        WHERE user_id = $user_id ORDER BY showdate DESC,ticket_id ASC;";

    $ticket_result = $mysqli -> query($ticket_q);


    #every time this page is loaded check to update show status first
    $movie_date = "UPDATE showtime SET show_status = 'ended' WHERE DATEDIFF(showdate,NOW()) < 1; ";
    $movie_date_result = $mysqli -> query($movie_date);

}

?>
```

3.1.3 Register page

For registration page We use to check is email that user input is in database or not then insert new account or return to let user know the error

```php
<?php

if ($_POST['email'] and $_POST['password'] and $_POST['c_password'] and $_POST['fname'] and $_POST['lname']) {
    $email = $_POST['email'];
    $password = $_POST['password'];
    $c_password = $_POST['c_password'];
    $fname = $_POST['fname'];
    $lname = $_POST['lname'];
    $fullname = $fname . " " . $lname;



    #Check confirm password is correct
    if (($password != $c_password)) {
        header("location:register_page.php?passwordnotmatch");
    } else {
        require_once('connect.php');
        #Check duplicate email in database
        $q = "SELECT * FROM user WHERE email = '$email'";
        $result = $mysqli->query($q);
        $count = $result->num_rows;
        #$password = md5($password); #hashing password
        if ($count >= 1) { #Return if Value exist in database;
            header("location:register_page.php?duplicate");
        } else { #INSERT INTO user table and redirect to login page
            $q = "INSERT INTO user (email,password,name) VALUE ('$email','$password','$fullname');";
            $result = $mysqli->query($q);
            header("location:login_page.php?createaccountcomplete");
        }

        $mysqli->close();
    }
} else { #Input is NULL return;
    header("location:register_page.php?inputnull");
}
```

3.1.4 Login page

For the login page SQL command is simple use to check is data in database or not and return the result

```php
<?php
if(isset($_POST['email']) AND isset($_POST['password'])){
    $email = $_POST['email'];
    $password = $_POST['password'];

    require_once('connect.php');
    $q = "SELECT * FROM user WHERE email = '$email' AND password = '$password';";
    $result = $mysqli-> query($q);

    if($result->num_rows >= 1){
        $row = $result -> fetch_array();
        session_start();
        $_SESSION['user_id'] = $row['user_id'];
        header("location:userprofile_page.php");
        exit();
    }
    else{
        header("location:login_page.php?usernotfound");
        exit();
    }

}
else{
    header("location:login_page.php");
    exit();
}
?>
```

3.1.5 Reset password page

For the reset password page user to update new password if user email is matched in database.

```php
<?php

$email = $_POST['email'];
if(isset($_POST['password']) AND isset($_POST['password'])){

    $pass = $_POST['password'];
    $c_pass = $_POST['c_password'];


    if($pass != $c_pass){
        header('location:resetpass_page.php?passwordnotmacthwithemail='.$email);
    }
    else{
        require_once('connect.php');
        $q = "UPDATE user SET password = '$pass' WHERE email = '$email';";
        if(!$mysqli -> query($q)){
            header('location:forgotpass_page.php?updatepasserror');
        }
        else{
            header('location:forgotpass_page.php?resetcomplete');
        }

    }

}
else{
    header('location:resetpass_page.php?inputerrorwithemail='.$email);
}
```

3.1.6 Booking page

This page is used to show more detail of movies, so we just query all columns with matched movies that the user selected.

```php
<?php
if (isset($_GET['movie_id'])) {
    $movie_id = $_GET['movie_id'];
    require_once('connect.php');
    $movie = "SELECT * FROM movie WHERE movie_id = $movie_id";
    $movie_result = $mysqli->query($movie);
    $movie_row = $movie_result->fetch_array();
} else {
}


?>
```

3.1.7 Confirmation ticket page

At first this page will get input from the html form and user, inserted Into to the ticket table to create the new ticket and also update seats remaining in the showtime table.

```php
<?php
session_start();?>

<?php
if(isset($_POST['show_id'])){

    $ticket_num = $_POST['ticket_num'];
    $user_id = $_SESSION['user_id'];
    $show_id = $_POST['show_id'];
    require_once('connect.php');

    $create_ticket = "INSERT INTO `ticket` (`ticket_id`, `show_id`, `user_id`, `status`) VALUES (NULL, '$show_id', '$user_id', 'Pay');";
    $create_ticket_q = $mysqli -> query($create_ticket);

    $seat_update = "UPDATE showtime SET avaliable_seat = avaliable_seat-$ticket_num WHERE show_id = $show_id;";
    $seat_update_q = $mysqli -> query($seat_update);

    header('location:userprofile_page.php');


}

else{
    header('location:home_page.php');
}
```

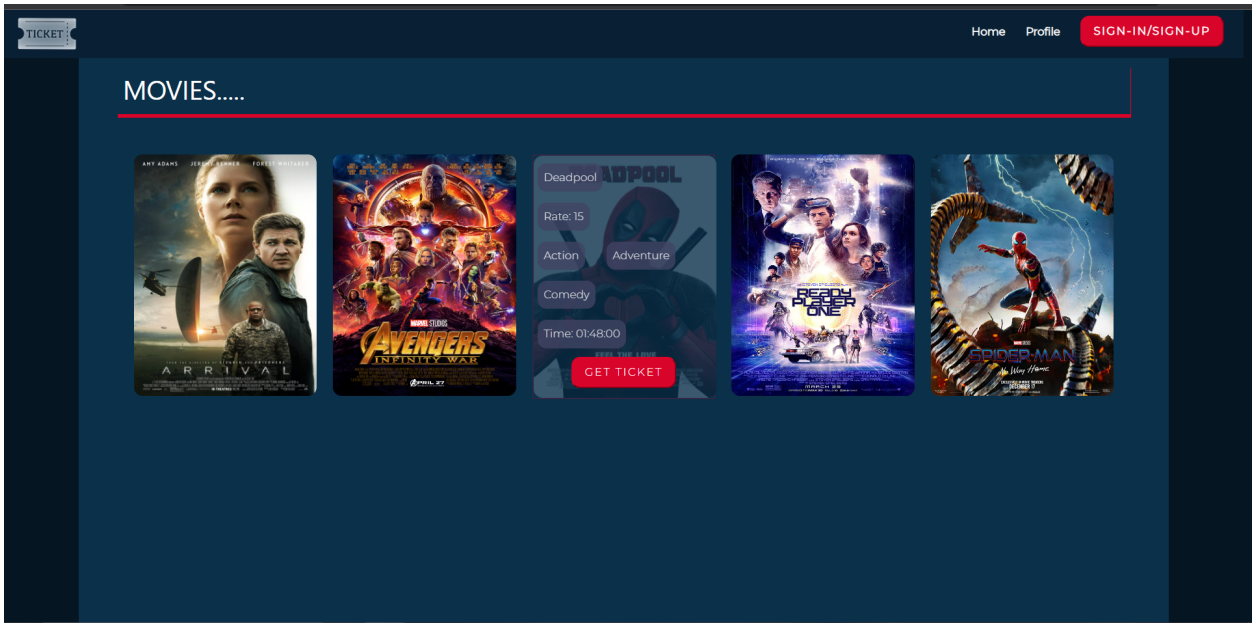
3.1.8 Payment page

The payment page is very simple with a ticket id that the user selects and query for show in the web page.

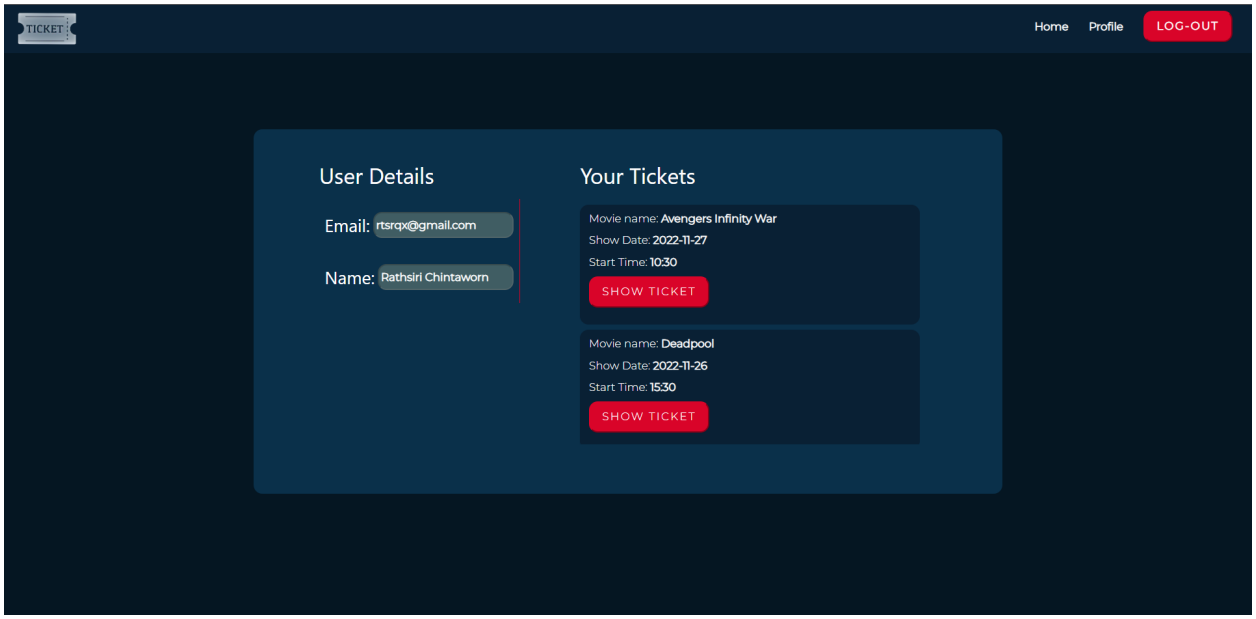
```php
<?php
if(isset($_GET['ticket_id'])){
    $ticket_id = $_GET['ticket_id'];
    require_once('connect.php');
    $q = "UPDATE ticket SET status = 'Show Ticket' WHERE ticket_id = $ticket_id;";
    $q_result = $mysqli -> query($q);
    header('location:userprofile_page.php');
}
else{
    header('location:userprofile_page.php');
}
?>
```

3.2 Screenshots

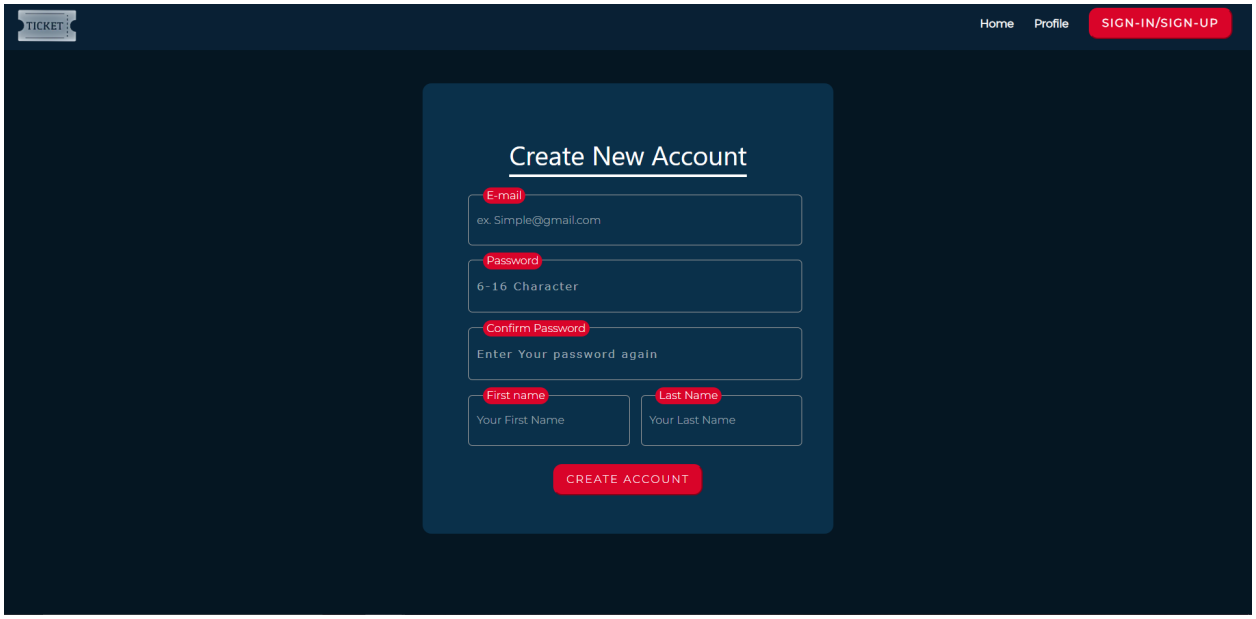
3.2.1 Home page



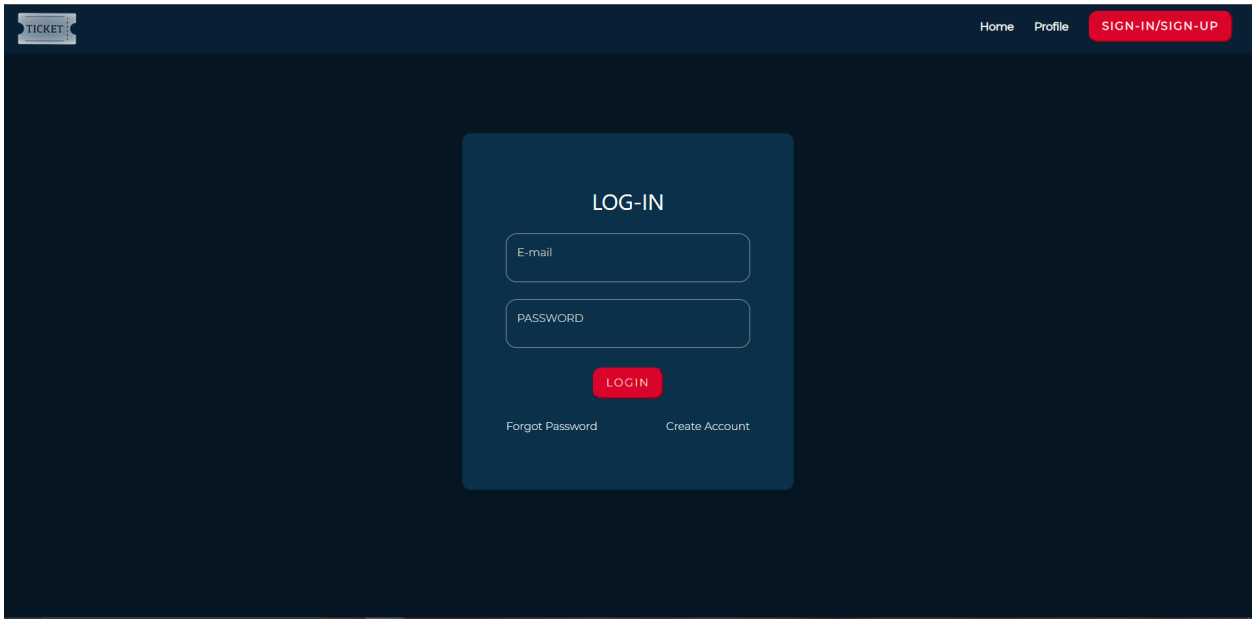
3.2.2 Profile page



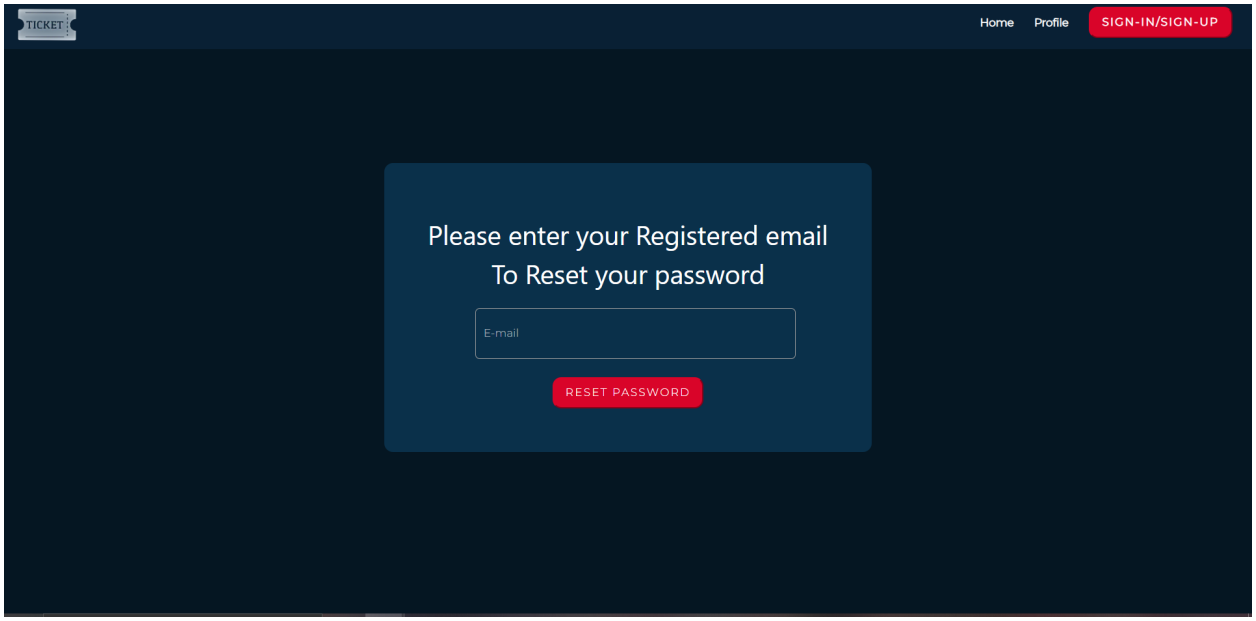
3.2.3 Register page

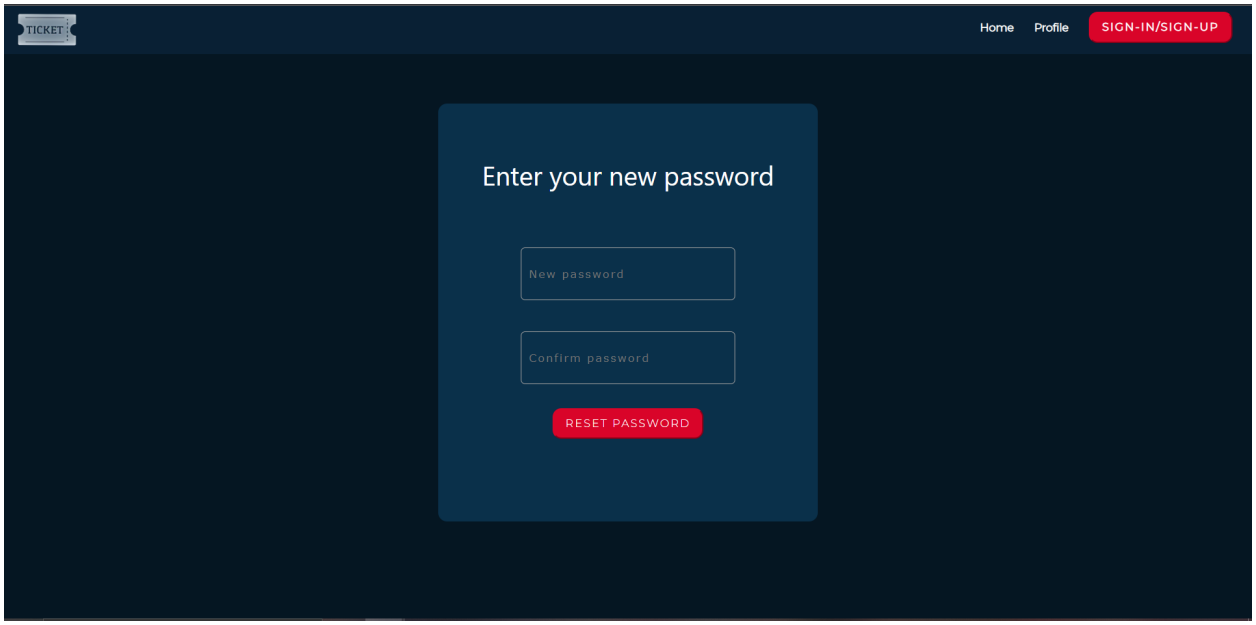


3.2.4 Login page

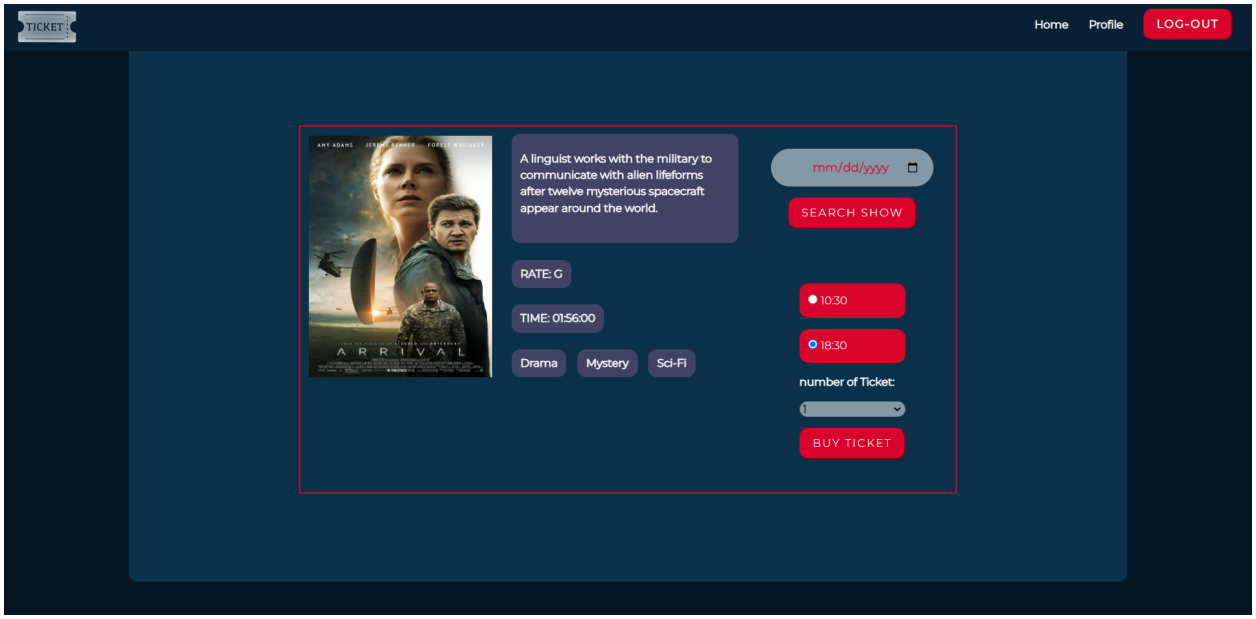


3.2.5 Reset password page

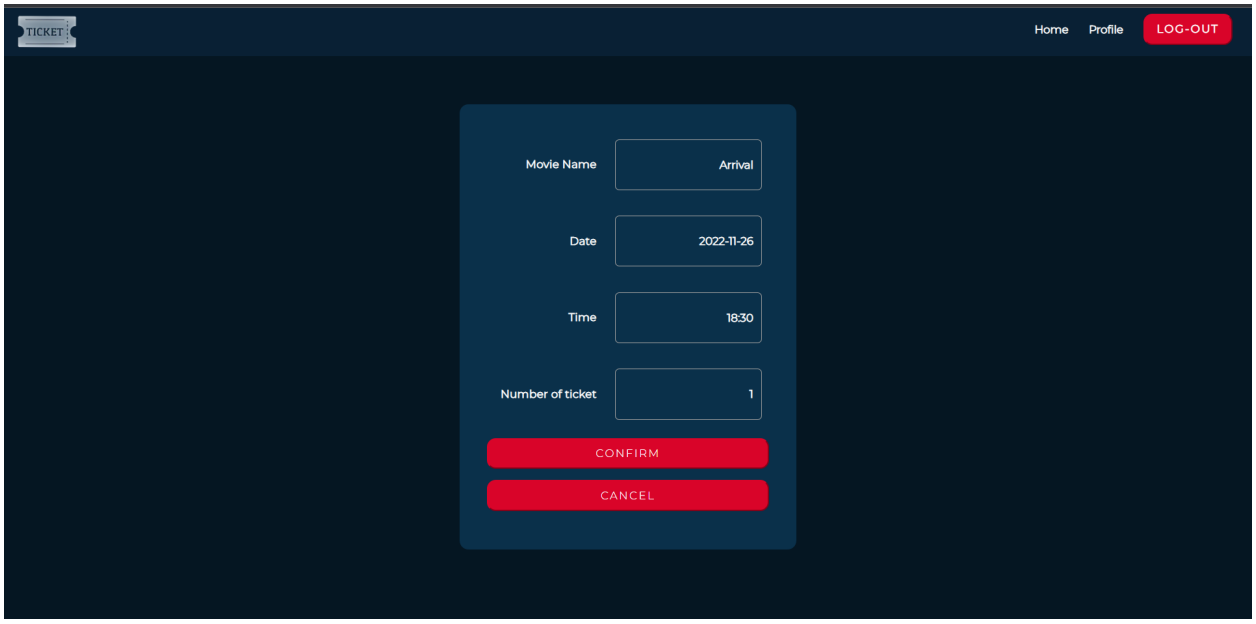




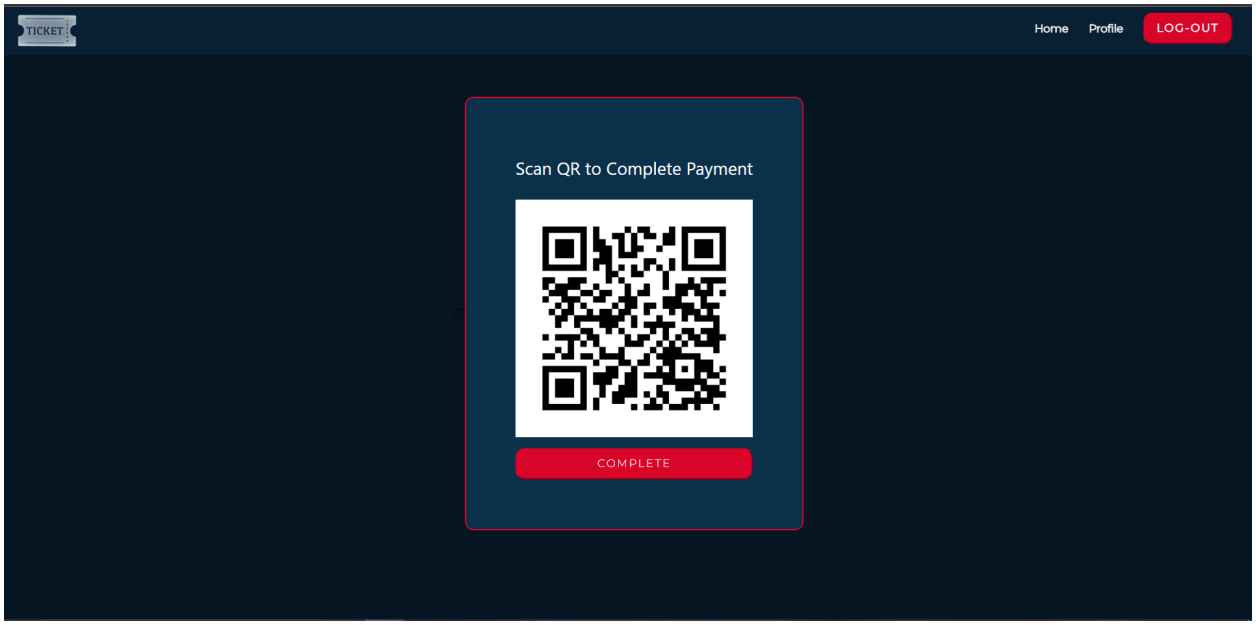
3.2.6 Booking page



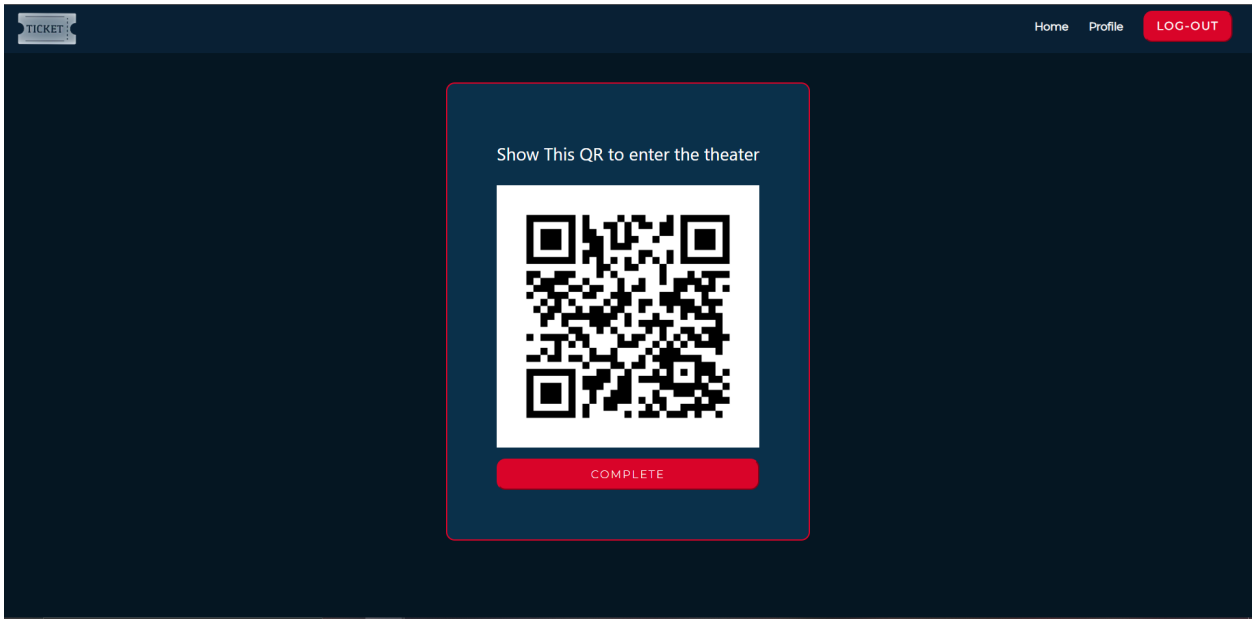
3.2.7 Confirmation ticket page



3.2.8 Payment page



3.2.9 Show ticket page



4. Explanation of Stored Procedures

Trigger command will update ticket status in ticket table if showtime status updated
Trigger Command In Table showtime

```
CREATE TRIGGER `update_ticketstatus` AFTER UPDATE ON `showtime` FOR EACH ROW
    UPDATE ticket SET ticket.status = 'ended'
        WHERE ticket.show_id = show_id
```

Before Update show_status column

	show_id	showdate	Start_time	end_time	movie_id	avaliable_seat	show_status
Edit Copy Delete	1	2022-11-22	10:30:01	12:20:00	1	47	ended
Edit Copy Delete	2	2022-11-22	15:30:02	17:20:00	1	0	ended
Edit Copy Delete	3	2022-11-22	18:30:03	20:20:00	1	47	ended
Edit Copy Delete	4	2022-11-23	10:30:04	12:20:00	1	47	ended
Edit Copy Delete	5	2022-11-23	10:30:05	12:20:00	2	47	ended
Edit Copy Delete	6	2022-11-24	18:30:03	20:20:00	1	47	ended
Edit Copy Delete	7	2022-11-24	10:30:01	12:20:00	1	47	ended
Edit Copy Delete	8	2022-11-24	15:30:02	17:20:00	1	47	ended
Edit Copy Delete	9	2022-11-27	10:30:00	13:10:00	2	47	coming

	ticket_id	show_id	user_id	status
Edit Copy Delete	2	1	1	ended
Edit Copy Delete	3	1	1	ended
Edit Copy Delete	19	1	1	ended
Edit Copy Delete	22	9	1	ended
Edit Copy Delete	23	9	1	Show Ticket

```
UPDATE `showtime` SET `show_status` = 'ended' WHERE `showtime`.`show_id` = 9;
```

Then after updating show_status in the showtime table so ticket status will change too.

← T →	ticket_id	show_id	user_id	status
Edit Copy Delete	2	1	1	ended
Edit Copy Delete	3	1	1	ended
Edit Copy Delete	19	1	1	ended
Edit Copy Delete	22	9	1	ended
Edit Copy Delete	23	9	1	ended

5. Conclusion

In conclusion, this project is developed successfully and the performance is found to be satisfactory. This project is designed to meet the requirements of assigning jobs. It has been developed in HTML, CSS, PHP, and JavaScript for front-end and the database has been built in MySQL server keeping in mind the specifications of the system.

The proposed system will allow efficient small business cinemas operations, encourage the development of a database of online ticket booking systems, resulting in market competition that is fair. The system is designed to be user-friendly, where users can easily understand the operation of the website with an easy-to-understand user interface. It also provides convenience to users because the website can be accessed, reserved and paid for anywhere, anytime.