CISC 332 Final Report

Luis Rivera-Wong - 10142361
Quentin Petraroia - 10145835
Braedan Robinson - 10188414

Assumptions

1. There will be ONE address per user/theatre complex/supplier

2. There will be ONE phone number per user/theatre complex/supplier

3. Only a registered user can leave a movie review even if they haven't seen the movie they can still leave a review because they may have seen it before/using a different website

4. You need to input a credit card to register for an account

Problems Encountered During Development

After we received the go ahead from Professor Powley, to use the laravel framework, our group quickly got to work understanding it. One of the first problems we encountered was figuring out where to start the project. As a group we instantly felt overwhelmed with how much we had to do and how little time we would have to get it done. After hours of brainstorming, we proposed an idea of how we can cut the work into chunks and work on parts at a time, and then connect them at the end. This was a good idea as we were able to knock out the front end design of the website fairly quickly.

When exploring the use of an API to build an adoptable user interface, we encountered a problem with querying the database. The error didn’t give what we expected which was to obtain a value from the database through a form-specified query about an attribute. Trying to solve the problem, it consumed hours of progress so eventually we decided to scrap a lot of the dependencies and form use through Vue.js. We realized that it was hard to learn multiple technologies as well as databases at the same time. As a group we resided with using Laravel and a minimal amount Vue components as it was not making our product any better and was just wasting production time.

Another problem we ran into was the lack of organization in terms of meeting, and focusing on development of the project. Due to each of our busy schedules we found it hard to meet at decent hours during the day, which resulted in each of us remotely working on components. This resulted in confusion when bringing the individual components together near the final deliverable deadline which caused a plethora of errors, and headaches. In order to complete the project we had to individually run-through our contributions, what the code was accomplishing, how to access and utilize the functions, and how to handle the output. The process took more time away from development, but in the end we were able to fix the errors, complete the project and gain a new set of knowledge about a new powerful framework.

Important Design and Implementation Decisions

While thinking about our project. Our group decided on trying to implement as close to a production level website as we could. We wanted users of our website to make sure they felt safe and secure. By implementing front end and back end validations we knew that our users would always be inputting the correct input while signing up for our website. The reason why we included both was because that front end validations can easily be tricked. Also, because front end validations require javascript we knew that if a user had javascript deactivated that none of our validations would work. By implementing back end validations as well, we for sure know that the users input could be checked by a validator and rejected if it is not the proper form submission.

Another important design discussion we had with our group was that of user authentication. How could we tell that users are logged in? How could we make sure that someone wouldn't buy tickets if they didn't have an account? The answer to this question was the use of middleware. By implementing custom middleware we were able to verify that the user was a authenticated user. An example of this would be if someone who is not logged in typed the specific url of the purchase page, they would be instantly directed to the login page. Using our custom middleware we were able to apply it directly to the admin side of the website. By creating a special PHP function is_admin in our user model. We were able to check the roles of the users. If a user was equal to the role “1”, then that user was an admin. We used this custom function in our Routes file to check against the HTTP requests (GET, POST) to make sure we would display the proper views for the admin or the regular user.

We decided to keep the design scheme of the website as simple as possible in order to create a user-friendly environment, that is easy accessible. In terms of styling, we decided to use bootstrap to create a responsive website, which makes it accessible for multi-platform use and creates appropriately sized blocks according to sections of the webpage. Another styling

feature, we went with basic identifiable colours that is easy on the eyes, which created visually appealing button styling.

The Technologies and Tools Used In Developing the Application, Why and Experience Using Them

Laravel

We decided to challenge ourselves to create a enterprise-level project purposely to learn more about the model-view-controller architecture and enterprise-level database management. After receiving permission from Professor Powley to use the PHP framework laravel, we began teaching ourselves. Laravel is a PHP framework intended for web development which follows the model-view-controller (MVC) architecture. It comes with a dedicated dependency manager, and alternate ways of accessing a database. Trying to learn a framework at the same time as developing a project was quite a headache at the start due to the learning curve. After the basic functionalities were mastered, it made development a lot faster, with a lot less errors.

By using Laravel we were able to follow industry standards by using the MVC architecture. The model (our database) responds to requests and can change the information it stores. The view displays everything that the user sees. It displays the data from the model to the web page. Finally, the controller takes in the user input and properly calls the model and view to properly display the data.

Bootstrap

Another web development tool we used to accomplish the styling and responsiveness of the HTML blocks for our website was Bootstrap. Bootstrap is a HTML, CSS, JS framework that is fully responsive, allowing the website to be accessible from any platform. Our group has some web development background, and we are all familiar with the framework so the utility was optimized which allowed us to create a visually appealing, yet minimalistic website. There was minimal time wasted on errors as the solution was always easy to find.

Docker

Vue.js

If We Could Go Back

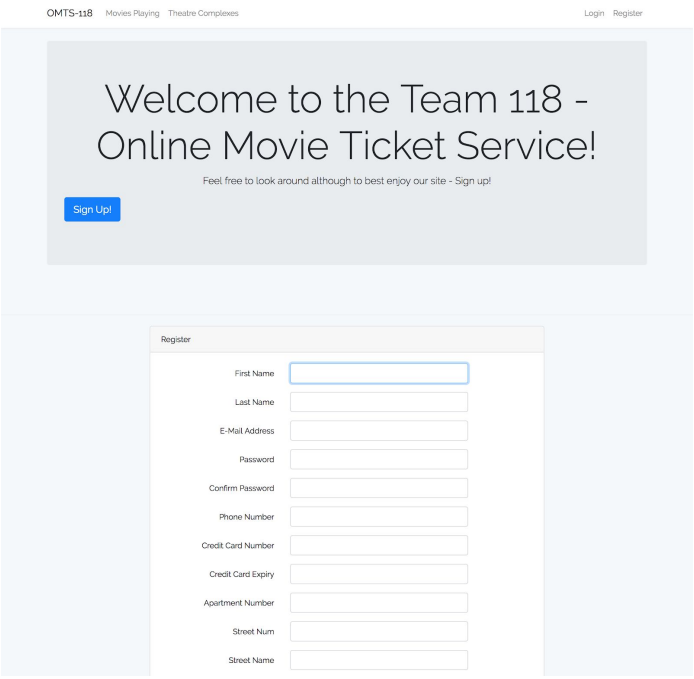
If we had the chance to go back and change things our group would have made the implementation process simpler, leaving the Laravel framework and using technologies learned in class. Looking back at the reason we decided to head down our development path, we wanted to create a product that uses similar technology to that of a large enterprise businesses, and to use the project as another piece of deliverables we could put on our resume. Although we finished with a solid product, it created a lot of headaches trying to learn, build the system and balance our workloads amongst other courses as well. Using HTML, CSS, PHP and JS to create the online movie ticket system would have been an easier job as our group has used each of the languages, and had some experience creating websites.

Another change we would have made is organizing set meeting times throughout the semester, and have milestones to be accomplished by each meeting. Throughout the semester a majority of our time on campus was used to focus on other courses since there were only two big deadlines for the project, so sometimes we felt that we were rushed. By meeting every week, and slowly building the project piece by piece would have prevented a lot of stress, and potentially allowed us to create a more polished product.

User Guide

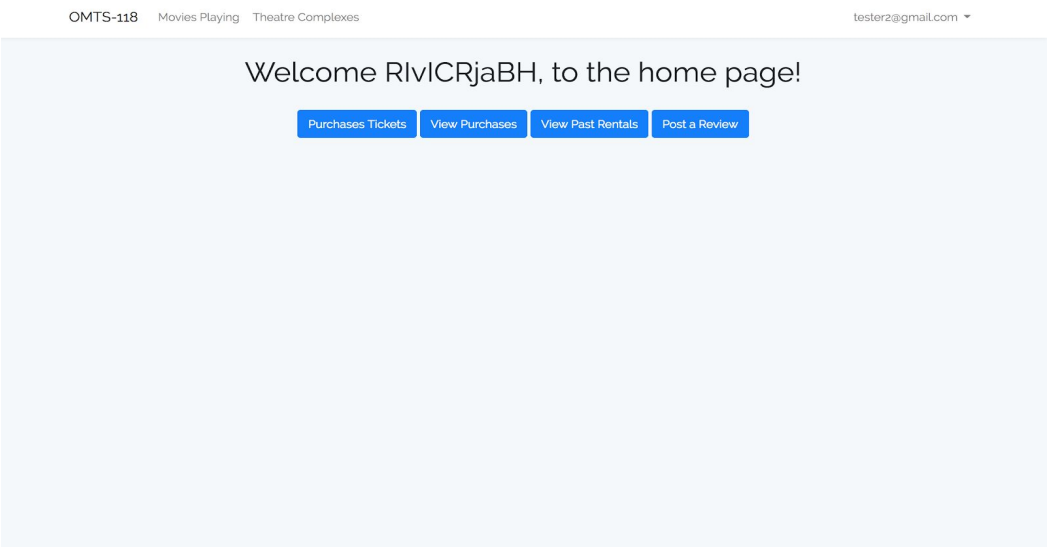
Regular User:

1. When a regular user goes to our website. They are greeted with the homepage. Here they can choose to log in if they are already a user or register.



2. If a user chooses to register they fill in the following fields. Not shown in the picture is the City, Province, Country and Postal Code fields. A user must either fill in the apartment number field or the street number field.

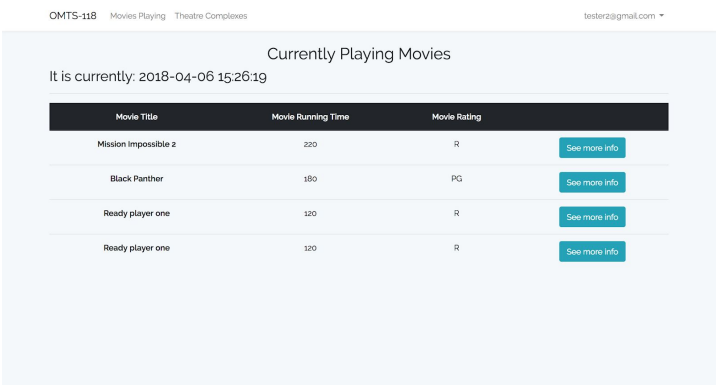
3. After registering you are greeted with the user home page. Here users go through the various functionalities of the OMTS. These are shown in the buttons below as well as the items in the nav bar.



These functions are:

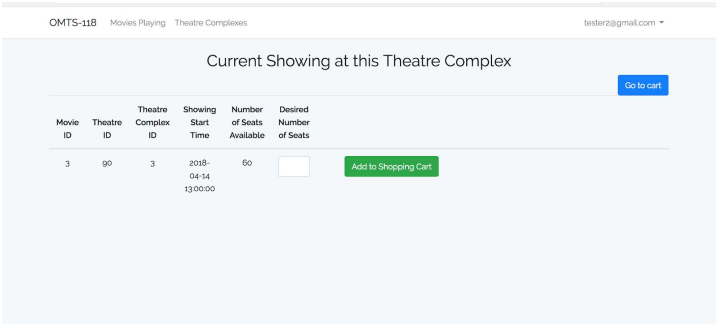
Browse movies playing at the various theatre complexes.

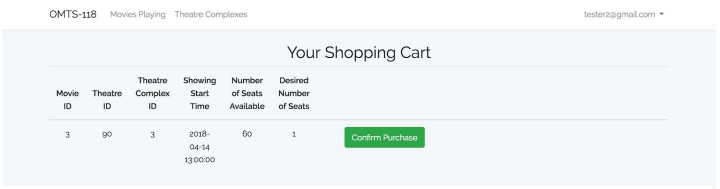
- Click on “Movies Playing” located at the top left of the webpage, in the Nav bar
- Movies from each theatre complex are gathered and displayed



Purchase tickets movies showing theatres

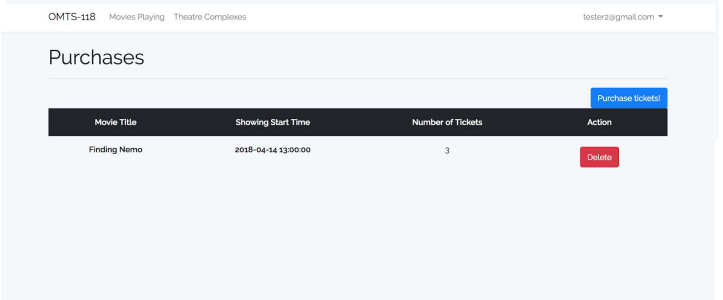
- Click on “Theatre Complexes” located at the top left of the homepage, in the Nav bar, or “Purchase Tickets” in the body of the webpage
- Select a theatre complex of choice
- Once a theatre complex is chosen you can see what movies are playing in each theatre and at what time they are showing.
- Users can also buy tickets and purchase tickets buy choosing the amount of tickets they want and then going to the add to cart page and confirming their purchase





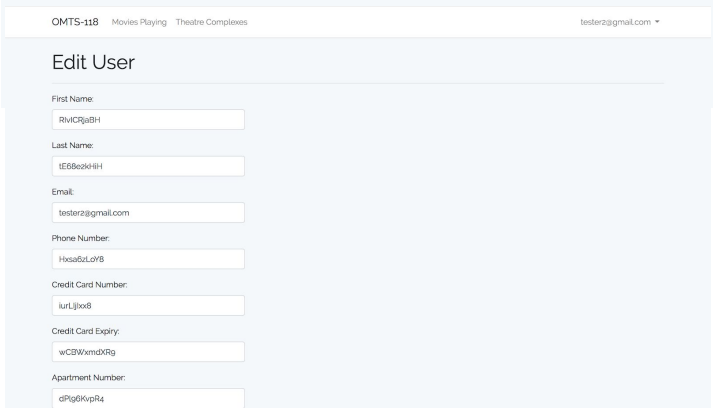
View ticket purchases/cancel a purchase

- Click on “View Purchases” located in the body of the homepage
- All purchased tickets for movies with future showtimes are displayed
- To cancel a ticket, the click “Delete” which will remove the purchased ticket(s) and confirm



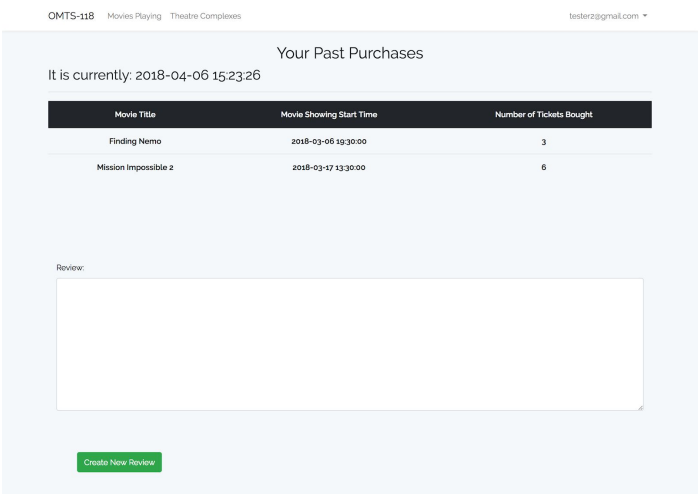
Update User personal details

- Click the user’s email for a drop down menu located in the top right of the page in the Nav bar
- Select “Edit User”
- Modify necessary user details in the corresponding text fields
- When finished, click “Update” to save the updated info



Browse past rentals

- Click on “View Past Rentals” located in the body of the homepage
- All previously purchased tickets with showtimes that have past are displayed to the user

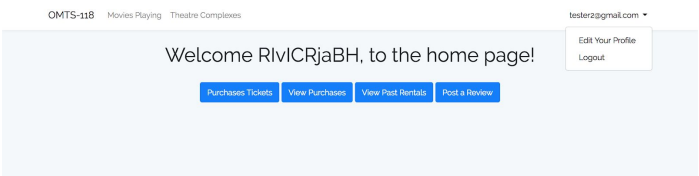


Add a review for a movie

- Click on the “Post a Review” button on the body of the home page
- Click on the movie you would like to see more details/add a review to
- Add a review, and click “Create New Review” when finished

When the user is finished with the movie ticket system (log out)

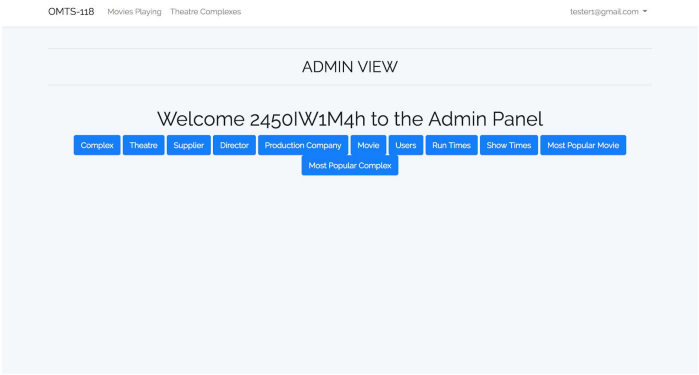
- Click the user’s email for a drop down menu located in the top right of the page in the Nav bar
- Select “Log out”
- The user has now been logged out of their account!

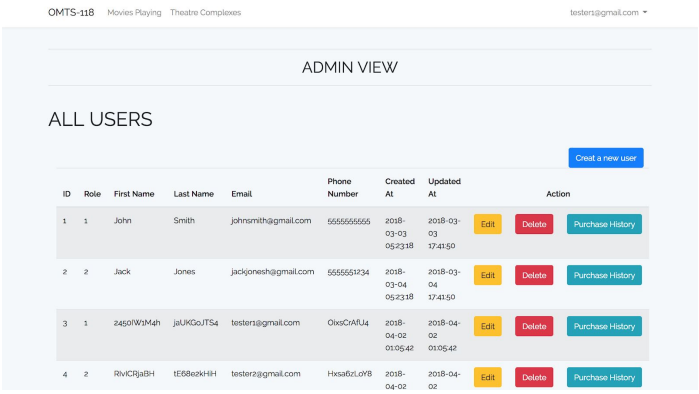


Administrators:

List all the members/remove a member/ view purchases

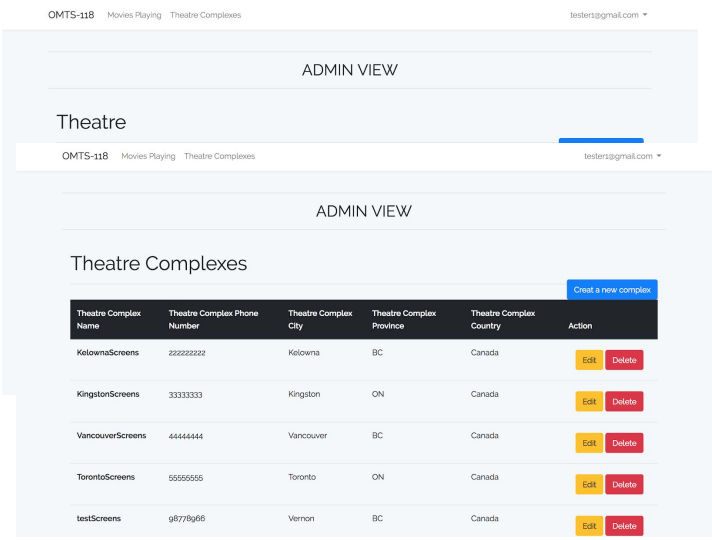
- Click “Users” located in the body of the homepage
- All users who registered, are displayed to the admin.
- To remove a member
    - click on “Delete” and confirm
- To view the purchase history of a particular user
    - Click on “Purchase history”
    - All tickets for future showtimes, and past ticket purchases are displayed





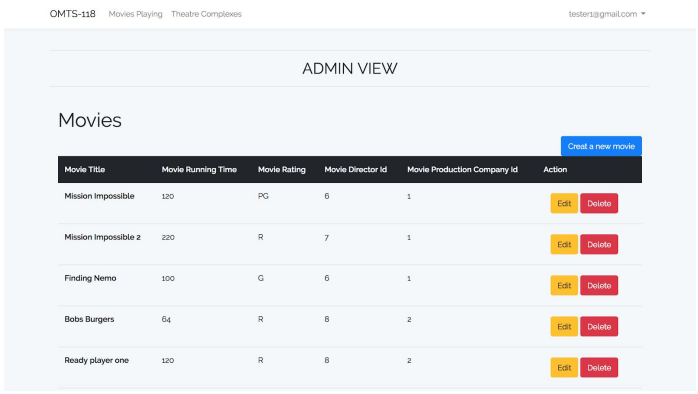
Add or update the information for a theatre complex/theatre

- Click on “Complex” or “Theatre”
- To update the information, click on “Edit”
    - Update necessary text fields
    - Click “Update” when finished
- To create a new complex or theatre
    - Click “Create New (Complex/Theatre)”
    - Enter information about the theatre or complex
    - When finished click “Create New ”
- To delete a complex or theatre
    - Click “Delete” and confirm



Add movies to the database

- Click on “Movie”
- Click “Create a new movie”
- Enter the details of the movie, the location and showtime
- Click “Create New” to confirm the new movie

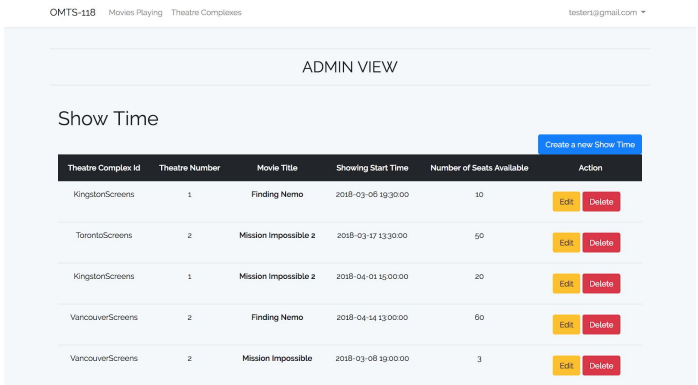


Update where/when movies are showing

- Click on “Show Times”
- Select the movie to update and click “Edit” to modify information
- When finished click “Update” to save the information

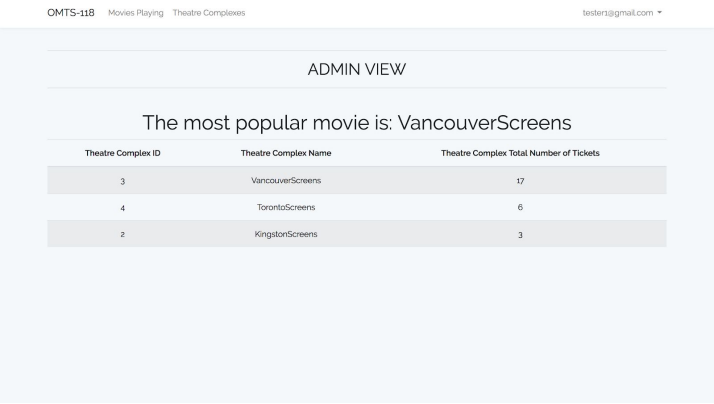
Find the most popular movie

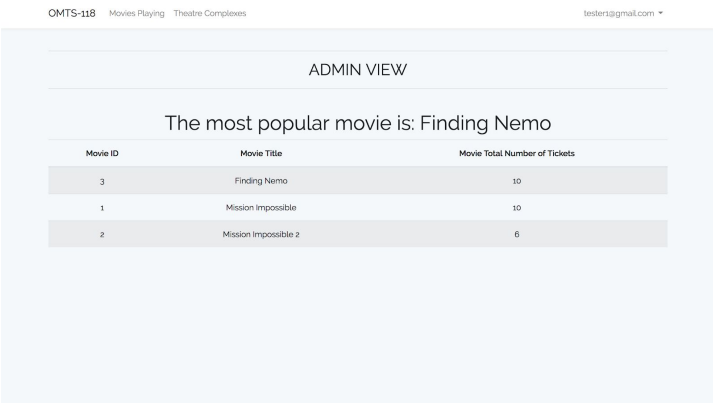
- Click on the “Most Popular Movie” button on the home page
- The most popular movies are displayed in descending order



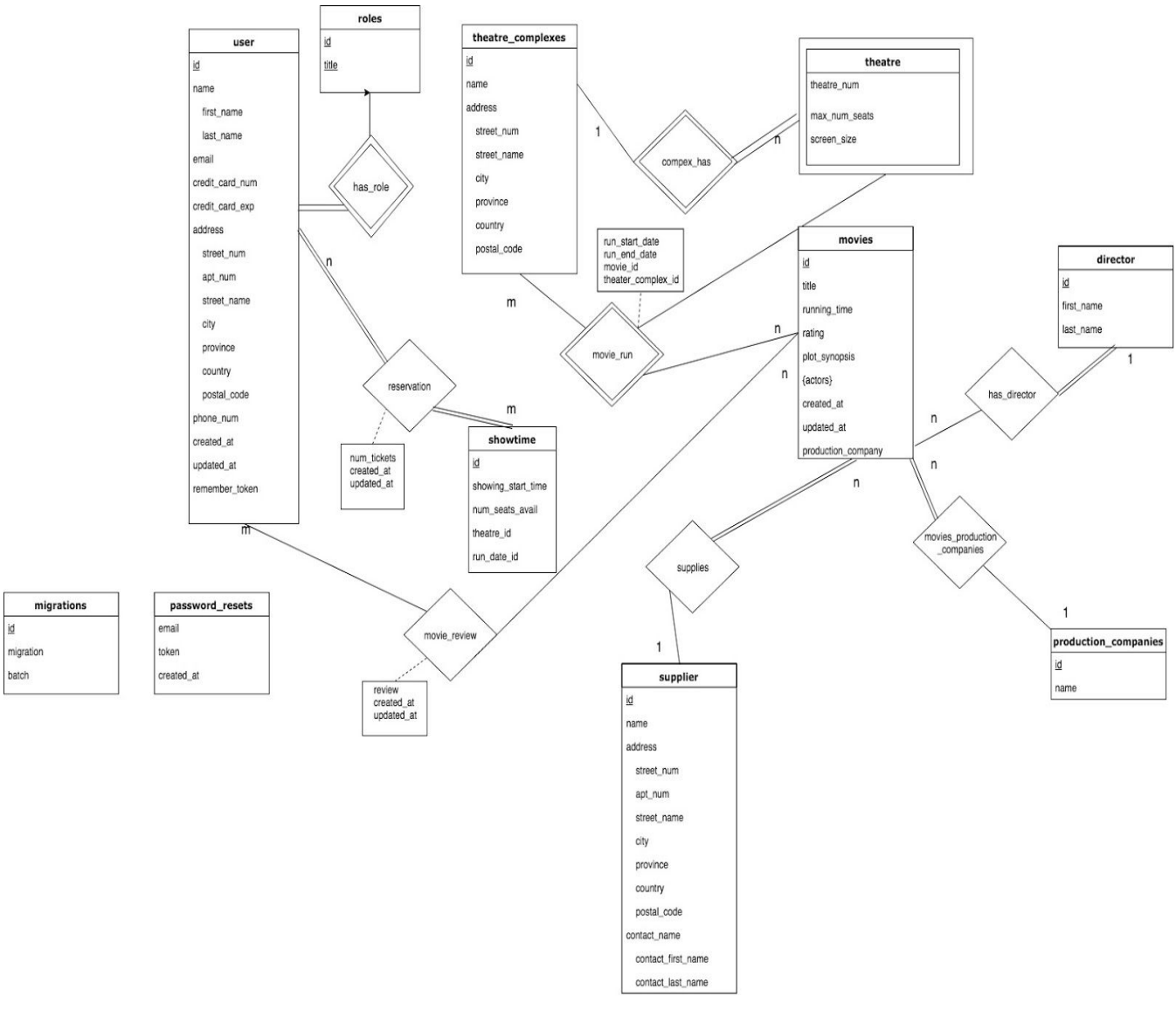
Find the most popular theatre complex

- Click on the “Most Popular Complex” button on the home page
- The most popular theatre complexes are displayed in descending order





ER Diagram


roles
id
title
user
id
name
first_name
last_name
email
credit_card_num
credit_card_exp
address
street_num
apt_num
street_name
city
province
country
postal_code
phone_num
created_at
updated_at
remember_token
has_role
theatre_complexes
id
name
address
street_num
street_name
city
province
country
postal_code
compex_has
theatre
theatre_num
max_num_seats
screen_size
run_start_date
run_end_date
movie_id
theater_complex_id
movie_run
movies
id
title
running_time
rating
plot_synopsis
{actors}
created_at
updated_at
production_company
director
id
first_name
last_name
has_director
reservation
num_tickets
created_at
updated_at
showtime
id
showing_start_time
num_seats_avail
theatre_id
run_date_id
supplies
movies_production
_companies
production_companies
id
name
supplier
id
name
address
street_num
apt_num
street_name
city
province
country
postal_code
contact_name
contact_first_name
contact_last_name
movie_review
review
created_at
updated_at
migrations
id
migration
batch
password_resets
email
token
created_at
1
n
m

Relational Schema

```
create database team3;
use team3;



CREATE TABLE `actors` (
   `id` int(10) UNSIGNED NOT NULL,
   `first_name` varchar(255) NOT NULL,
   `last_name` varchar(255) NOT NULL,
   PRIMARY KEY (`id`)
);

CREATE TABLE `actors_movies` (
```

```
   `actor_id` int(10) UNSIGNED NOT NULL,
   `movie_id` int(10) UNSIGNED NOT NULL,
   PRIMARY KEY (`actor_id`,`movie_id`),
   KEY `actors_movies_movie_id_foreign` (`movie_id`)
);

CREATE TABLE `carts` (
   `id` int(10) UNSIGNED NOT NULL,
   `user_id` int(10) UNSIGNED NOT NULL,
   `showing_id` int(10) UNSIGNED NOT NULL,
   `run_date_id` int(10) UNSIGNED NOT NULL,
   `number_of_tickets` int(11) NOT NULL,
   `created_at` timestamp NULL DEFAULT NULL,
   `updated_at` timestamp NULL DEFAULT NULL,
   PRIMARY KEY (`id`),
   KEY `carts_user_id_foreign` (`user_id`),
   KEY `carts_showing_id_foreign` (`showing_id`),
   KEY `carts_run_date_id_foreign` (`run_date_id`)
);



CREATE TABLE `directors` (
   `id` int(10) UNSIGNED NOT NULL,
   `first_name` varchar(255) NOT NULL,
   `last_name` varchar(255) NOT NULL,
   PRIMARY KEY (`id`)
);


CREATE TABLE `migrations` (
   `id` int(10) UNSIGNED NOT NULL,
   `migration` varchar(255) NOT NULL,
   `batch` int(11) NOT NULL,
   PRIMARY KEY (`id`)
);

CREATE TABLE `movies` (
```

```
   `id` int(10) UNSIGNED NOT NULL,
   `title` varchar(255) NOT NULL,
   `running_time` int(11) NOT NULL,
   `rating` varchar(255) NOT NULL,
   `plot_synopsis` text NOT NULL,
   `director_id` int(10) UNSIGNED DEFAULT NULL,
   `prod_comp_id` int(10) UNSIGNED DEFAULT NULL,
   `supplier_id` int(10) UNSIGNED DEFAULT NULL,
   PRIMARY KEY (`id`),
   KEY `movies_director_id_foreign` (`director_id`),
   KEY `movies_prod_comp_id_foreign` (`prod_comp_id`),
   KEY `movies_supplier_id_foreign` (`supplier_id`)
);

CREATE TABLE `password_resets` (
   `email` varchar(255) NOT NULL,
   `token` varchar(255) NOT NULL,
   `created_at` timestamp NULL DEFAULT NULL,
   KEY `password_resets_email_index` (`email`)
);


CREATE TABLE `production_companies` (
   `id` int(10) UNSIGNED NOT NULL,
   `name` varchar(255) NOT NULL,
   PRIMARY KEY (`id`)
);


CREATE TABLE `reservations` (
   `id` int(10) UNSIGNED NOT NULL,
   `user_id` int(10) UNSIGNED DEFAULT NULL,
   `showing_id` int(10) UNSIGNED DEFAULT NULL,
   `number_of_tickets` int(10) UNSIGNED NOT NULL,
   PRIMARY KEY (`id`),
   KEY `reservations_user_id_foreign` (`user_id`),
   KEY `reservations_showing_id_foreign` (`showing_id`)
);
```

```
CREATE TABLE `reviews` (
   `user_id` int(10) UNSIGNED NOT NULL,
   `movie_id` int(10) UNSIGNED NOT NULL,
   `review` text NOT NULL,
   `created_at` timestamp NULL DEFAULT NULL,
   `updated_at` timestamp NULL DEFAULT NULL,
   PRIMARY KEY (`user_id`,`movie_id`),
   KEY `reviews_movie_id_foreign` (`movie_id`)
);


CREATE TABLE `roles` (
   `id` int(10) UNSIGNED NOT NULL,
   `title` varchar(255) NOT NULL,
   PRIMARY KEY (`id`)
);


CREATE TABLE `run_dates` (
   `id` int(10) UNSIGNED NOT NULL,
   `movie_id` int(10) UNSIGNED NOT NULL,
   `theatre_complex_id` int(10) UNSIGNED NOT NULL,
   `run_start_date` date NOT NULL,
   `run_end_date` date NOT NULL,
   PRIMARY KEY (`id`),
   UNIQUE KEY `run_dates_movie_id_theatre_complex_id_unique`
(`movie_id`,`theatre_complex_id`),
   KEY `run_dates_theatre_complex_id_foreign` (`theatre_complex_id`)
);


CREATE TABLE `show_times` (
   `id` int(10) UNSIGNED NOT NULL,
   `theatre_id` int(10) UNSIGNED NOT NULL,
   `showing_start_time` datetime NOT NULL,
   `num_seats_avail` int(10) UNSIGNED NOT NULL,
   `run_date_id` int(10) UNSIGNED NOT NULL,
```

```
   PRIMARY KEY (`id`),
   UNIQUE KEY `showing_id` (`theatre_id`,`showing_start_time`,`run_date_id`),
   KEY `show_times_run_date_id_foreign` (`run_date_id`)
);

CREATE TABLE `suppliers` (
   `id` int(10) UNSIGNED NOT NULL,
   `name` varchar(255) NOT NULL,
   `phone_num` varchar(255) NOT NULL,
   `contact_first_name` varchar(255) NOT NULL,
   `contact_last_name` varchar(255) NOT NULL,
   `apt_num` varchar(255) NOT NULL,
   `street_num` varchar(255) NOT NULL,
   `street_name` varchar(255) NOT NULL,
   `city` varchar(255) NOT NULL,
   `province` varchar(255) NOT NULL,
   `country` varchar(255) NOT NULL,
   `postal_code` varchar(255) NOT NULL,
   PRIMARY KEY (`id`)
);

CREATE TABLE `theatres` (
   `id` int(10) UNSIGNED NOT NULL,
   `theatre_num` varchar(255) NOT NULL,
   `max_num_seats` int(11) NOT NULL,
   `screen_size` int(11) NOT NULL,
   `theatre_complex_id` int(10) UNSIGNED NOT NULL,
   PRIMARY KEY (`id`),
   KEY `theatres_theatre_complex_id_foreign` (`theatre_complex_id`)
);


CREATE TABLE `theatre_complexes` (
   `id` int(10) UNSIGNED NOT NULL,
   `name` varchar(255) NOT NULL,
   `phone_num` varchar(255) NOT NULL,
   `street_num` varchar(255) NOT NULL,
   `street_name` varchar(255) NOT NULL,
```

```
    `city` varchar(255) NOT NULL,
    `province` varchar(255) NOT NULL,
    `country` varchar(255) NOT NULL,
    `postal_code` varchar(255) NOT NULL,
    PRIMARY KEY (`id`)
);



CREATE TABLE `users` (
    `id` int(10) UNSIGNED NOT NULL,
    `role_id` int(10) UNSIGNED NOT NULL DEFAULT '2',
    `first_name` varchar(255) NOT NULL,
    `last_name` varchar(255) NOT NULL,
    `email` varchar(255) NOT NULL,
    `password` varchar(255) NOT NULL,
    `phone_num` varchar(255) NOT NULL,
    `credit_card_num` varchar(255) NOT NULL DEFAULT '',
    `credit_card_exp` varchar(255) NOT NULL DEFAULT '',
    `apt_num` varchar(255) NOT NULL,
    `street_num` varchar(255) NOT NULL,
    `street_name` varchar(255) NOT NULL,
    `city` varchar(255) NOT NULL,
    `province` varchar(255) NOT NULL,
    `country` varchar(255) NOT NULL,
    `postal_code` varchar(255) NOT NULL,
    `remember_token` varchar(100) DEFAULT NULL,
    `created_at` timestamp NULL DEFAULT NULL,
    `updated_at` timestamp NULL DEFAULT NULL,
    PRIMARY KEY (`id`),
    UNIQUE KEY `users_email_unique` (`email`),
    KEY `users_role_id_foreign` (`role_id`)
);

-- ############################################################ --

INSERT INTO `actors` (`id`, `first_name`, `last_name`) VALUES
(3, 'Dory', 'Fish'),
```

```
(6, 'Stacy', 'Jones');



INSERT INTO `actors_movies` (`actor_id`, `movie_id`) VALUES
(3, 1),
(6, 1),
(6, 2),
(3, 3);


INSERT INTO `carts` (`id`, `user_id`, `showing_id`, `run_date_id`,
`number_of_tickets`, `created_at`, `updated_at`) VALUES
(1, 4, 4, 4, 1, '2018-04-06 14:37:26', '2018-04-06 14:37:26');


INSERT INTO `directors` (`id`, `first_name`, `last_name`) VALUES
(6, 'Jason', 'Bourne'),
(7, 'John', 'Wick'),
(8, 'Tony', 'Stark');



INSERT INTO `migrations` (`id`, `migration`, `batch`) VALUES
(1, '2014_10_11_000000_create_roles_table', 1),
(2, '2014_10_12_000000_create_users_table', 1),
(3, '2014_10_12_100000_create_password_resets_table', 1),
(4, '2018_02_24_031625_create_production_companies_table', 1),
(5, '2018_02_24_031627_create_movies_table', 1),
(6, '2018_02_24_031628_create_directors_table', 1),
(7, '2018_02_24_183737_create_actors_table', 1),
(8, '2018_02_24_183738_create_actors_movies_table', 1),
(9, '2018_02_24_184631_create_suppliers_table', 1),
(10, '2018_02_24_202726_create_theatre_complexes_table', 1),
(11, '2018_02_24_204701_create_theatres_table', 1),
(12, '2018_02_24_212421_create_run_dates_table', 1),
(13, '2018_02_24_212529_create_show_times_table', 1),
(14, '2018_02_24_225510_create_reservations_table', 1),
```

```
(15, '2018_02_24_230741_create_reviews_table', 1),
(16, '2018_03_29_170630_add_foreign_keys_to_movies', 1),
(17, '2018_04_01_155604_create_carts_table', 1);



INSERT INTO `movies` (`id`, `title`, `running_time`, `rating`, `plot_synopsis`,
`director_id`, `prod_comp_id`, `supplier_id`) VALUES
(1, 'Mission Impossible', 120, 'PG', 'Tom Cruise is on a mission yeehaw', 6, 1, 20),
(2, 'Mission Impossible 2', 220, 'R', 'Tom Cruise is back on a mission', 7, 1, 60),
(3, 'Finding Nemo', 100, 'G', 'A dad fish looses his son and must find him', 6, 1,
20),
(4, 'Bobs Burgers', 64, 'R', 'its burger time', 8, 2, 61),
(5, 'Ready player one', 120, 'R', 'You are in a video game! WOW!', 8, 2, 61),
(7, 'Black Panther', 180, 'PG', 'Black panther is a super hero movie', 7, 1, 60);



INSERT INTO `production_companies` (`id`, `name`) VALUES
(1, 'MGM'),
(2, 'TMG');



INSERT INTO `reservations` (`id`, `user_id`, `showing_id`, `number_of_tickets`)
VALUES
(30, 4, 1, 3),
(99, 4, 2, 6),
(100, 4, 4, 3),
(101, 3, 5, 3),
(102, 3, 5, 1),
(103, 1, 4, 2),
(104, 1, 5, 2),
(105, 6, 5, 1),
(106, 6, 5, 2),
(108, 7, 5, 1),
(109, 7, 4, 2);
```

```
INSERT INTO `reviews` (`user_id`, `movie_id`, `review`, `created_at`, `updated_at`)
VALUES
(1, 1, 'bang bang bang pow', '2018-03-02 00:00:00', '2018-03-02 00:00:00'),
(1, 3, 'sad movie...', '2018-03-17 00:00:00', '2018-03-18 00:00:00'),
(7, 3, 'Great movie!', '2018-04-04 00:00:00', '2018-04-04 00:00:00');



INSERT INTO `roles` (`id`, `title`) VALUES
(1, 'admin'),
(2, 'user');


INSERT INTO `run_dates` (`id`, `movie_id`, `theatre_complex_id`, `run_start_date`,
`run_end_date`) VALUES
(1, 3, 2, '2018-03-01', '2018-03-16'),
(2, 2, 4, '2018-03-16', '2018-03-22'),
(3, 2, 2, '2018-03-31', '2018-04-14'),
(4, 3, 3, '2018-04-14', '2018-04-21'),
(5, 1, 3, '2018-04-07', '2018-04-15'),
(10, 7, 7, '2018-03-02', '2018-04-18'),
(11, 5, 5, '2018-03-19', '2018-04-19'),
(13, 5, 7, '2018-03-28', '2018-04-20');



INSERT INTO `show_times` (`id`, `theatre_id`, `showing_start_time`,
`num_seats_avail`, `run_date_id`) VALUES
(1, 55, '2018-03-06 19:30:00', 10, 1),
(2, 90, '2018-03-17 13:30:00', 50, 2),
(3, 55, '2018-04-01 15:00:00', 20, 3),
(4, 90, '2018-04-14 13:00:00', 60, 4),
(5, 90, '2018-03-08 19:00:00', 3, 5),
(7, 95, '2018-04-04 00:00:00', 55, 4),
(8, 97, '2018-03-27 00:00:00', 32, 11);
```

```
INSERT INTO `suppliers` (`id`, `name`, `phone_num`, `contact_first_name`,
`contact_last_name`, `apt_num`, `street_num`, `street_name`, `city`, `province`,
`country`, `postal_code`) VALUES
(20, 'Some Supplier', '1234567788', 'Mike', 'Will Make Money', '', '55', 'Some
Street', 'Some City Name', 'Quebec', 'Canada', 'H8G1J0'),
(60, 'BestSupplier', '7894561122', 'Micheal', 'Li', '89', '88', 'JayZ Drive',
'Ottawa', 'Ontario', 'Canada', 'K9F1H6'),
(61, 'QSupplier', '89867655', 'Quentin', 'Jones', '', '23', 'Big srt', 'Edmonton',
'AB', 'Canada', 'K9K4S');


INSERT INTO `theatres` (`id`, `theatre_num`, `max_num_seats`, `screen_size`,
`theatre_complex_id`) VALUES
(55, '1', 200, 20, 1),
(90, '2', 150, 15, 2),
(91, '3A', 400, 500, 5),
(93, '3B', 3, 4, 5),
(95, '4A', 90, 88, 7),
(97, '4B', 799, 800, 7),
(99, '5A', 999, 8000, 4);



INSERT INTO `theatre_complexes` (`id`, `name`, `phone_num`, `street_num`,
`street_name`, `city`, `province`, `country`, `postal_code`) VALUES
(1, 'KelownaScreens', '222222222', '123', 'main street', 'Kelowna', 'BC', 'Canada',
'V1B4Z1'),
(2, 'KingstonScreens', '33333333', '543', 'cherry street', 'Kingston', 'ON',
'Canada', 'K4X8H2'),
(3, 'VancouverScreens', '44444444', '723', 'bloop street', 'Vancouver', 'BC',
'Canada', 'M2R7Z9'),
(4, 'TorontoScreens', '55555555', '742', 'bleep street', 'Toronto', 'ON', 'Canada',
'M9K8J4'),
(5, 'testScreens', '98778966', '159', 'Pinegrove', 'Vernon', 'BC', 'Canada',
'V3B9C2'),
(7, 'PentictonScreens', '65465465', '982', 'Blank', 'Street', 'AB', 'Canada',
'B8Y3D');
```

```
INSERT INTO `users` (`id`, `role_id`, `first_name`, `last_name`, `email`,
`password`, `phone_num`, `credit_card_num`, `credit_card_exp`, `apt_num`,
`street_num`, `street_name`, `city`, `province`, `country`, `postal_code`,
`remember_token`, `created_at`, `updated_at`) VALUES
(1, 1, 'John', 'Smith', 'johnsmith@gmail.com', '12345678', '5555555555',
'1234567890123456', '0421', '43', '644', 'Johnson St.', 'Kingston', 'Ontario',
'Canada', 'K7K4S1', NULL, '2018-03-03 05:23:18', '2018-03-03 17:41:50'),
(2, 2, 'Jack', 'Jones', 'jackjonesh@gmail.com', '12345678', '5555551234',
'9999999999999999', '0522', '', '633', 'Princess St.', 'Kingston', 'Ontario',
'Canada', 'K7K4S2', NULL, '2018-03-04 05:23:18', '2018-03-04 17:41:50'),
(3, 1, '2450IW1M4h', 'jaUKGoJTS4', 'tester1@gmail.com',
'$2y$10$2w3.yi/igxXxiTHUO7nIjuU1X6GJrI7gIFtMTRl2SZnbUSdhYUKfm', 'OixsCrAfU4',
'Diq5MMpXyx', 'bryLPrakPk', '2h12Tw8YnF', 'rXD7A67WpV', 'Q8dl8tWwre', 'fvIh8821T1',
'Ia3rf2H3BP', 'jl6r4gKEeY', 'aC9d0YHryz',
'yjlgRtVKQYoRfeKsBkmqpsIlqy3ngmzUhZ1EemJ4OTuMdMA6FhLZTJw6gGfA', '2018-04-02
01:05:42', '2018-04-02 01:05:42'),
(4, 2, 'RIvICRjaBH', 'tE68e2kHiH', 'tester2@gmail.com',
'$2y$10$t8/DUtaiN3chww8dvBS0net0.ybVna3RX8eO6ZH6DshH7SHyDdQyG', 'Hxsa6zLoY8',
'iurLIjIxx8', 'wCBWxmdXR9', 'dPl96KvpR4', 'jK5IzhBAFW', '7b6bGsYSQm', 'Qq7XfrNzf3',
'M996Gox4GJ', 'rvbrxOC6ee', 'OiFDQ96AJ1',
'2dpkm1dqu73F7dRZlhIL2B4RIpH7GJNiT0C2l9IzUwMbrVH0z7WDptSmVXwy', '2018-04-02
01:05:42', '2018-04-02 01:05:42'),
(6, 2, 'Bob', 'Billy', 'bob@gmail.com',
'$2y$10$C8CJ.yF8Q2XOnO.Jvu4hqOepeH6KdO5PUCwwWeiVnyvHgp5obxWxC', '2508643851',
'9999999999999', '1212', '', '24', 'br street', 'Toronto', 'ON', 'Canada', 'K7L
4A6', 'dQGhd1wufkaUdx0DtpCGGzclGxhKVa5YiyIYetnHQBl6VWtdy66O1qZ702sJ', '2018-04-02
10:46:46', '2018-04-02 10:47:38'),
(7, 2, 'Curtis', 'Miller', 'Kdog@gmail.com',
'$2y$10$kHBek7GCm6jby/ma2tyIZe8tBieFW1T.Tkxd2dkpl4S.pBQlNJMl.', '9999999',
'12345678', '1221', '', '22', 'borg street', 'Vernon', 'BC', 'Canada', 'v9xb5w',
'9PX4a28xeFX2HIqb9MN4HuSQkcGL7zGwPsEgNgMyY0hWAL9OFFSeDN7pYFls', '2018-04-02
11:45:41', '2018-04-02 11:45:54'),
(8, 2, 'Bobby', 'Jones', 'bobby@gmail.com',
'$2y$10$hNAuvf02lklAqS9KMLK/R.V1WHvcvrAd8HUUpf6h9ntc4PemF/YLe', '6565676',
'65543234', '543342', '', '11', 'Pinegrove', 'Kelown', 'BC', 'Canada', 'V8WI2K',
'jW0FjdbJBFKnYKKxXcO9wGcfR9y8rLjrOOOBIDrgdHTEgtZp3l2fG82yeIXW', '2018-04-02
```

```
13:09:20', '2018-04-02 13:09:20');
```

SQL Queries