

National College of Ireland
BSc in Computing
2016/2017

Eoin Sutton
X13116053
eoin.sutton@live.com

Movie-io

Technical Report



Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: Eoin Sutton
Student ID: X13116053
Supervisor: Eugene McLaughlin

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: _____

Date: _____

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

Table of Contents

Executive Summary	6
1 Introduction	7
1.1 Background.....	7
1.2 Aims.....	7
1.3 Technologies	7
2 System.....	8
2.1 Requirements	8
2.1.1 Purpose.....	8
2.1.2 Project Scope	8
2.1.3 Definitions, Acronyms, and Abbreviations	9
2.1.4 User Requirements Definition	9
2.1.5 Requirements Specification.....	9
2.2 Functional requirements	10
2.2.1 Use Case Diagram.....	10
2.2.2 Requirement 1: User Registration	10
2.2.3 Requirement 2: User Login	12
2.2.4 Requirement 3: Movie Search.....	14
2.2.5 Requirement 4: Rate Movie.....	16
2.2.6 Requirement 5: Share Movie List	18
2.2.7 Requirement 6: Review Users.....	20
2.2.8 Requirement 7: View Cinemas.....	22
2.3 Non-Functional Requirements	24
2.3.1 Performance/Response time requirement.....	24
2.3.2 Security requirement	24
2.3.3 Reliability requirement.....	24
2.3.4 Usability requirement.....	24
2.4 Interface requirements	25
2.5 GUI	25
2.6 Application Programming Interfaces (API)	34
2.7 System Architecture.....	34
2.8 System Evolution	36

3	Design and Architecture	37
3.1	Implementation	37
3.1.1	User Login & Registration	37
3.1.2	Web Scraper API.....	41
3.1.3	Movie Reviews	44
3.1.4	User Reviews	49
3.1.5	What's Hot - Location specific movie data.....	52
3.1.6	Facebook Sharing and My List Page.....	55
3.1.7	Cinemas/Maps page	58
3.1.8	Theme Switcher	60
3.2	Graphical User Interface (GUI) Layout.....	63
3.3	Testing.....	73
3.4	Customer testing.....	81
3.5	Heuristic Evaluation	92
4	Conclusions	94
5	References	95
6	Appendix A	98
6.1	Project Proposal	100
	Movie Rating App.....	100
6.1.1	Objectives	100
6.1.2	Background	101
6.1.3	Technical Approach.....	102
6.1.4	Project Plan.....	103
6.1.5	Technical Details	104
6.1.6	Evaluation	104
6.2	Monthly Journals.....	105
	My Achievements	105
6.3	Supervisor Meetings	111
6.4	Links	114

Executive Summary

The purpose of this document is to set out the technical details for the development of Movie-io, a mobile web based application that allows users to rate and review movies. The key aspects of this application are that users can share lists of their favourite movies with friends and review other reviewers of the site. It also contains other functionality such as geo-location to report the nearest cinemas to the user.

It is being developed in jQuery mobile, HTML and CSS for the front end and PHP with MySQL in the backend. This will allow for a modern, fast and responsive application.

The intended audience for this application are movie aficionados and people who would like a modern application to help them track the movies they have watched, record their thoughts on them and share those thoughts with their friends.

1 Introduction

This project is intended to create an application to allow users to rate and review the movies they have watched. It is also intended to allow users to share these reviews with friends and other users of the site. It will have additional features such as the ability for users to “review the reviewer” and geo-location functionality to show the nearest cinemas. It is intended to be a one-stop shop for keeping track of movies you’ve loved and loathed and sharing your discoveries with friends.

1.1 Background

I chose this project (one of the pre-approved projects in NCI) as I have a love of movies and felt there was a gap in the market for a cross-platform, easy to use application that will quickly allow me to review a movie I’ve just seen, keep those in a handy list and share that list with my friends. There are several sites that will allow you to rate movies but none with all the functionality together or the ease of use I would like to build.

1.2 Aims

The main aim of this application is to create a simple, easy and quick to use mobile web based application to rate and review movies. It will include a modern interface and modern functionality through libraries such as jQuery mobile. It will be able to gather information on movies from the web quickly and store user reviews and ratings.

1.3 Technologies

The client side will be based on jQuery mobile, HTML5 and CSS3 to create a modern responsive look and feel. In the backend PHP will be used to manage connections to the database, while the database itself will be MySQL.

2 System

2.1 Requirements

2.1.1 Purpose

The purpose of this document is to set out the requirements for the development of Movie-io, a responsive web application that will allow users to rate and review movies and TV shows.

2.1.2 Project Scope

The scope of the project is to develop a mobile responsive web application that will fulfil the requirements set out in this requirements document. The app will be created using HTML5, with CSS for styling, and Javascript for complex functions. PHP on the server will handle interaction with the database, which shall be MySQL. Various libraries including, but not limited to, jQuery and jQuery Mobile will also be used.

The app will be optimised for use on mobile devices and will enable users to create an account, search for movies, review movies, share movie lists with friends, and review other users of the application. Geo-location will be used for listing nearby cinemas to book tickets. Searching for movies will involve the use of APIs to scrape data from the web, and, where possible, the use of freely available APIs such as the OMDb.

The intended customers are movie lovers who wish to keep track of, and review, the films and shows they have watched, as well as the ability to share those movies with friends. It is intended as the “one stop shop” for your movie needs, giving users only the information they need to view, review, track and share movies.

As part of defining project scope I elicited user requirements from my project mentor, Eugene McLaughlin, who mentioned the ability to “review the reviewer” would be a good feature. I also elicited user requirements from a sample of 5 friends who I asked for views on features they would most like to see in an app of this type. This was done via face to face interviews, asking each user a set number

of questions (see Appendix A). Section 2.1.4, User Requirements Definition, defines this in more detail.

2.1.3 Definitions, Acronyms, and Abbreviations

API – Application Programming Interface

GPS – Global Positioning System

CSS – Cascading Style Sheets

OMDB – Open Movie Database

2.1.4 User Requirements Definition

The user requirements were set out by contacting 5 different users from a sample of people who were regular movie goers.

The following requirements were gathered via user interviews (Appendix A).

- The app shall allow users to rate and review movies and TV shows
- The app shall be responsive, with a wait of no more than 1-2 seconds to complete a task
- The app shall include geo-location functionality to list nearest cinemas with appropriate URL to their websites
- The app shall allow users to share their favourite movie lists with friends
- The app shall allow users to review other users in a “review the reviewer” fashion

2.1.5 Requirements Specification

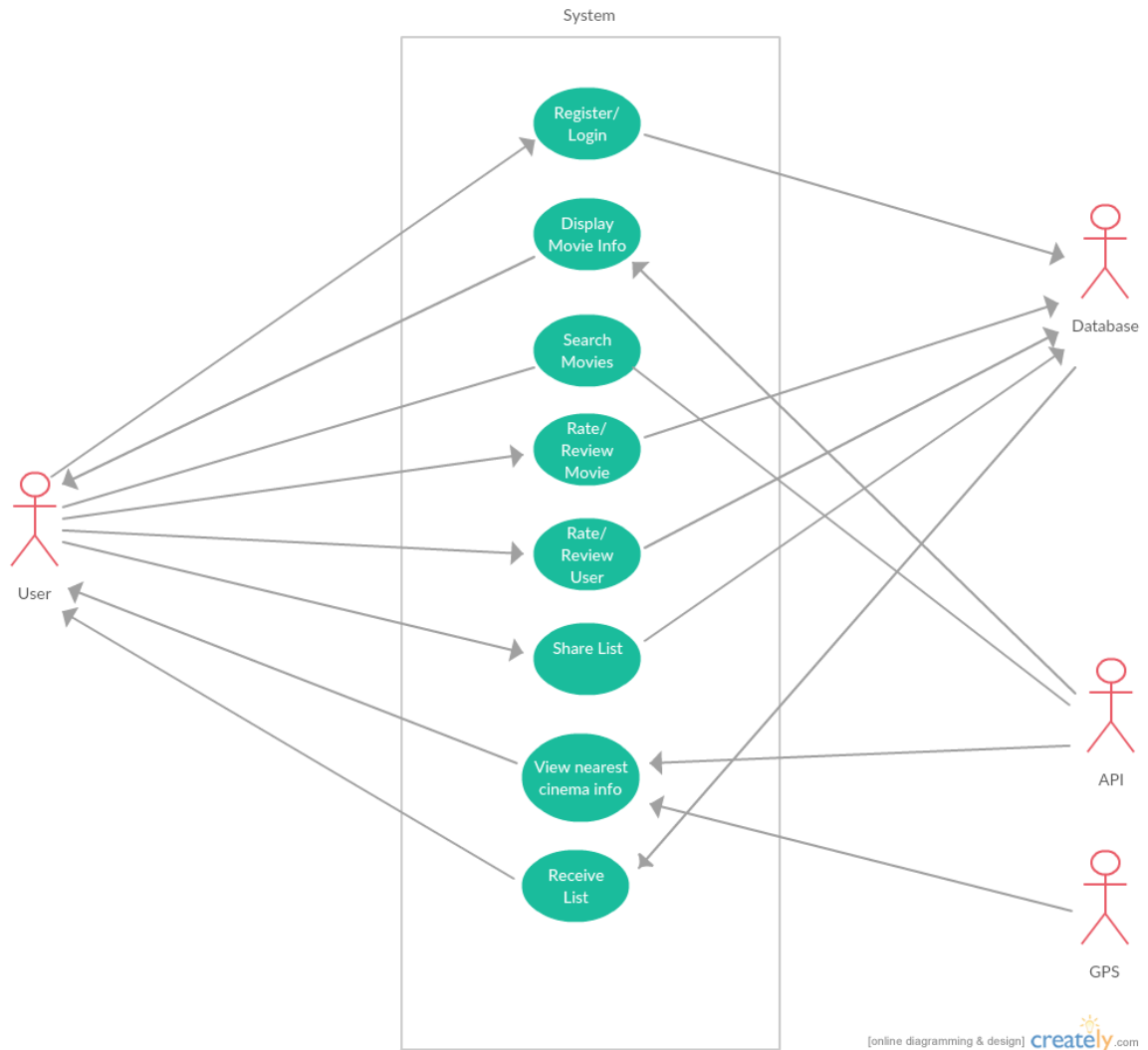
The app will be a mobile responsive web app based on HTML5, CSS and Javascript. It will:

- Allow a user to register an account and login
- Allow the user to search for movies and TV shows and return info about that movie or TV show including synopsis, cast, director, writer, runtime, date of release
- Allow a user to select a movie and rate that movie a score out of 10 and also review that movie in plain text
- Allow a user to send and receive lists of rated movies from other users
- Allow a user to review other registered users of the application
- Display a list of movie theatres nearby to the users location via GPS, with links to their websites

The database will be MySQL and PHP will be used as the interface to that database.

2.2 Functional requirements

2.2.1 Use Case Diagram



2.2.2 Requirement 1: User Registration

2.2.2.1 Description & Priority

The user shall be able to register with a username and password. This is essential in order to save lists and share these lists with other users.

2.2.2.2 Use Case Priority 1

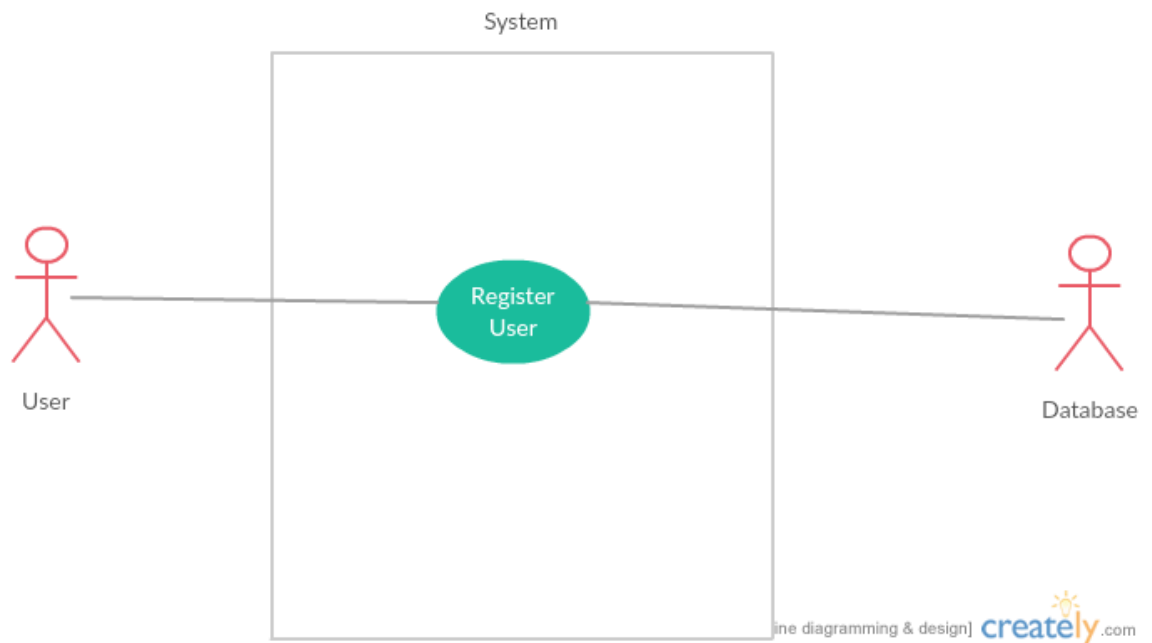
Scope

The scope of this use case is to allow the user to register in the application.

Description

The user registration use case consists of methods to allow a user to login. The user shall select a password and a username and this will be saved to the database.

Use Case Diagram



Flow Description

Precondition

The registration page has been loaded in the browser and an internet connection is active.

Activation

This use case begins when the user enters his/her details and clicks the register button.

Main flow

1. When the user clicks the register button, the details are sent to the system
2. The system will save the details in the database
3. The system will return a success message to the user

Alternate flow

1. None identified

Exceptional flow

1. If the system sees a username that already exists, the system will return a failure message to the user

Termination

The use case terminates once the user has successfully created an account.

Post condition

The system returns to the login page.

2.2.3 Requirement 2: User Login

2.2.3.1 Description & Priority

The user shall be able to login to the application with a username and password. This is essential in order to be able to create and share lists.

2.2.3.2 Use Case Priority 2

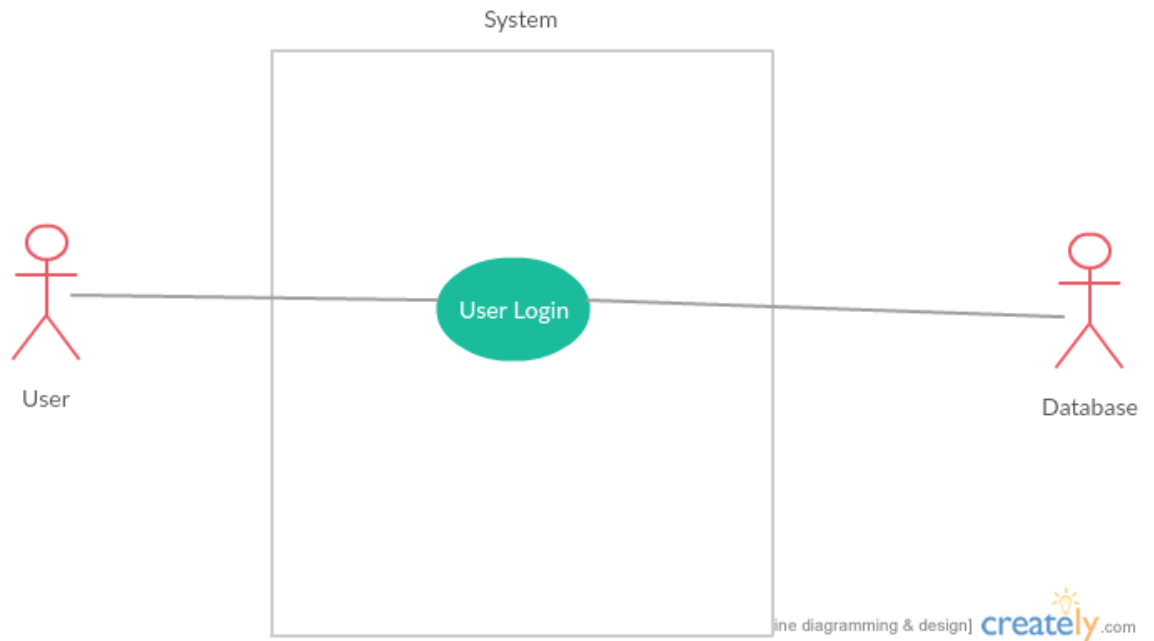
Scope

The scope of this use case is to allow a user to login to access the functions of the application.

Description

This use case describes the login procedure for the application. A user shall be able to enter his/her username and password and this will log the user in to the application.

Use Case Diagram



Flow Description

Precondition

The login page has been loaded in the browser and an internet connection is active.

Activation

This use case begins when the user enters his/her details and clicks the login button.

Main flow

1. When the user clicks the login button, the details are sent to the system
2. The system checks credentials against the database
3. The system returns a success message to the user

Alternate flow

1. None Identified

Exceptional flow

E1 :

1. If the user is already logged in, the system will return that message
2. If the credentials of the user are incorrect, the system will return a fail message
3. The use case restarts at Main Flow 1

Termination

The user is logged in.

Post condition

The system redirects the user to the main page.

2.2.4 Requirement 3: Movie Search

2.2.4.1 Description & Priority

The user shall be able to search for movies and TV shows and have relevant data for same returned to the user. Important for getting movie info to work with.

2.2.4.2 Use Case Priority 3

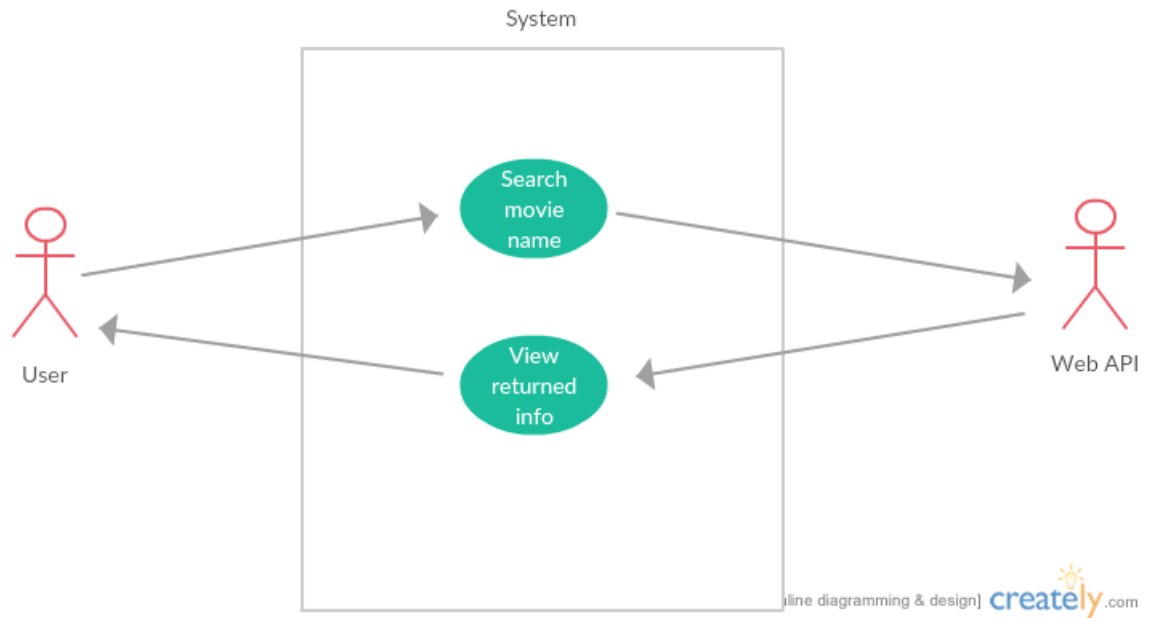
Scope

The scope of this use case is for a user to search for movie and TV show names and have info returned for display.

Description

This use case describes the search procedure for a user. The user will enter a movie name in the application and click enter. The application will route via an API to retrieve info for that movie including cast, director, writer, runtime, date of release and synopsis, which will be displayed on screen.

Use Case Diagram



Flow Description

Precondition

The user is logged in and an internet connection is active.

Activation

This use case begins when the user clicks the search link and enters a movie or TV show title in the application search field.

Main flow

1. The system will send the movie name to the API
2. The API will return the details of said movie to system
3. The system will display details on screen

Alternate flow

1. None Identified

Exceptional flow

E1 :

1. If a movie is not found, no data will be returned to the user

Termination

The user receives movie data.

Post condition

The application has movie data displayed on screen.

2.2.5 Requirement 4: Rate Movie

2.2.5.1 Description & Priority

The user shall be able to rate a movie (the user may also add a review). Important as this is the main functionality of the application.

2.2.5.2 Use Case Priority 4

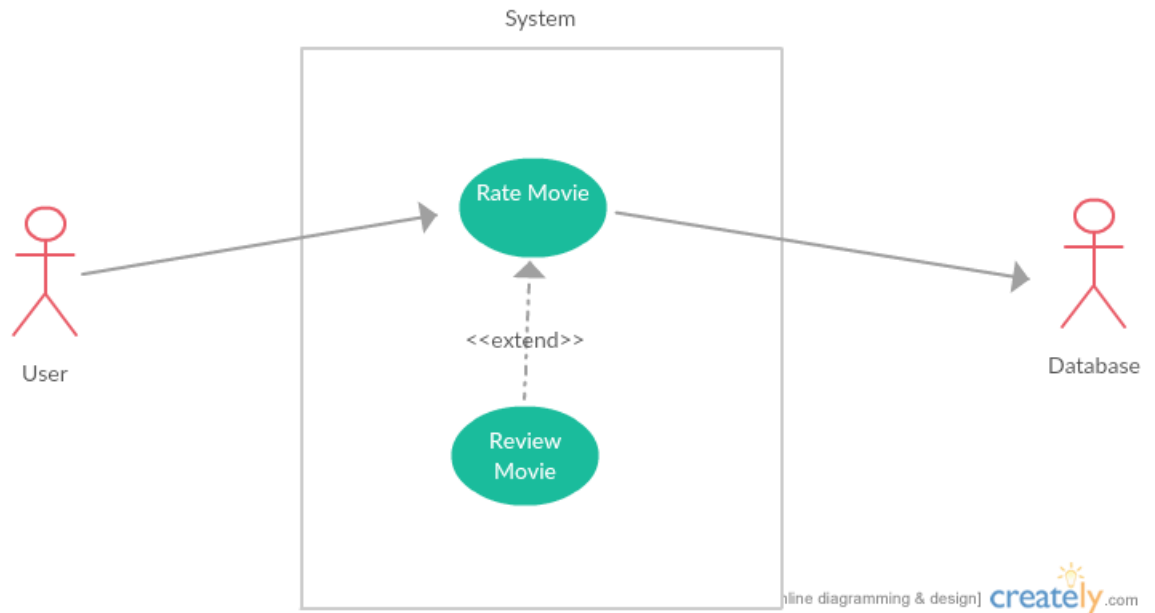
Scope

The scope of this use case is for a user to be able to rate/review a movie and have that saved to the database.

Description

This use case describes the procedure for rating a movie. The user may add a rating to a movie that he/she has searched for or clicked on. The user may also add a review in plain text along with the rating. The rating will be a score out of 10.

Use Case Diagram



Flow Description

Precondition

The user is logged in and an internet connection is active.

Activation

This use case begins when the user clicks the search button and enters a movie or TV show title in the application search field.

Main flow

1. The system will send the movie name to the API
2. The API will return the details of said movie to system
3. The system will display details on screen
4. The user may enter a rating out of 10
5. The user may enter a plaintext review of the movie
6. The user may then click the “Submit” button
7. The system will save the details to the database

Alternate flow

1. The user clicks on a movie name under the “Box Office” page
2. The system continues from the third point on the main flow

Exceptional flow

E1 :

1. If the database is not accessible, the data will not get saved and the user will return to the Search page
2. If the API is not accessible, the user will not move past the Search page

Termination

The user clicks the Submit button and the data is saved.

Post condition

The system saves the rating and/or review to the database.

2.2.6 Requirement 5: Share Movie List

2.2.6.1 Description & Priority

The user shall be able to share the list of movies he/she has currently rated with friends. This is an important feature and a unique selling point of the application.

2.2.6.2 Use Case Priority 5

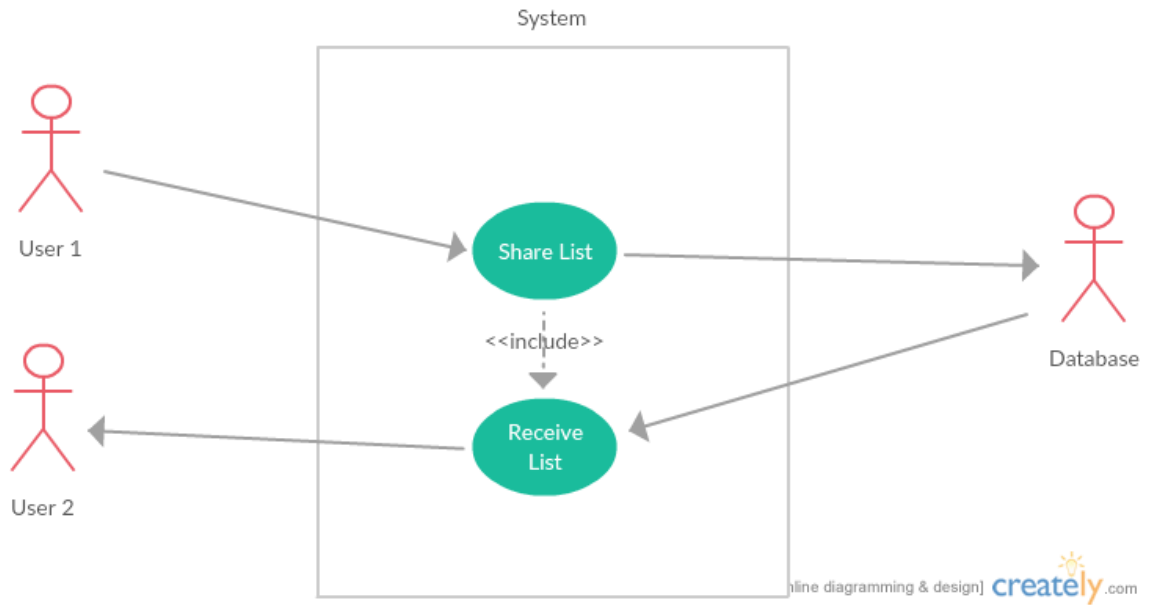
Scope

The scope of this use case is for a user to be able to share movie lists he/she has saved to the database with another user of the application.

Description

In the “My List” section of the application, this will list the movies that the logged in user has rated or reviewed. This list can then be shared by clicking the “Share” button on the page. The user can select from a list of other users he/she has flagged as friends. The other user can then accept or reject the share request.

Use Case Diagram



Flow Description

Precondition

The user is logged in and an internet connection is active.

Activation

This use case begins when the user selects the “Share” link on the “My List” page.

Main flow

1. The system will list the friends the logged in user has
2. The user may select the friend to send the list to
3. The system will get the friend details from the database
4. The system will send the friend a notification on their account
5. The friend may view the list via their account

Alternate flow

1. The user may send the list to a friend from the friend’s user profile
2. The user continues from the third step of the main flow

Exceptional flow

E1 :

1. No friends are defined so the user cannot send the list

Termination

The users friend receives the list sent by the logged in user.

Post condition

The users friend can view the rated movie list from his/her friend.

2.2.7 Requirement 6: Review Users

2.2.7.1 Description & Priority

The user shall be able to review other users. This feature is important to differentiate the application from others.

2.2.7.2 Use Case Priority 6

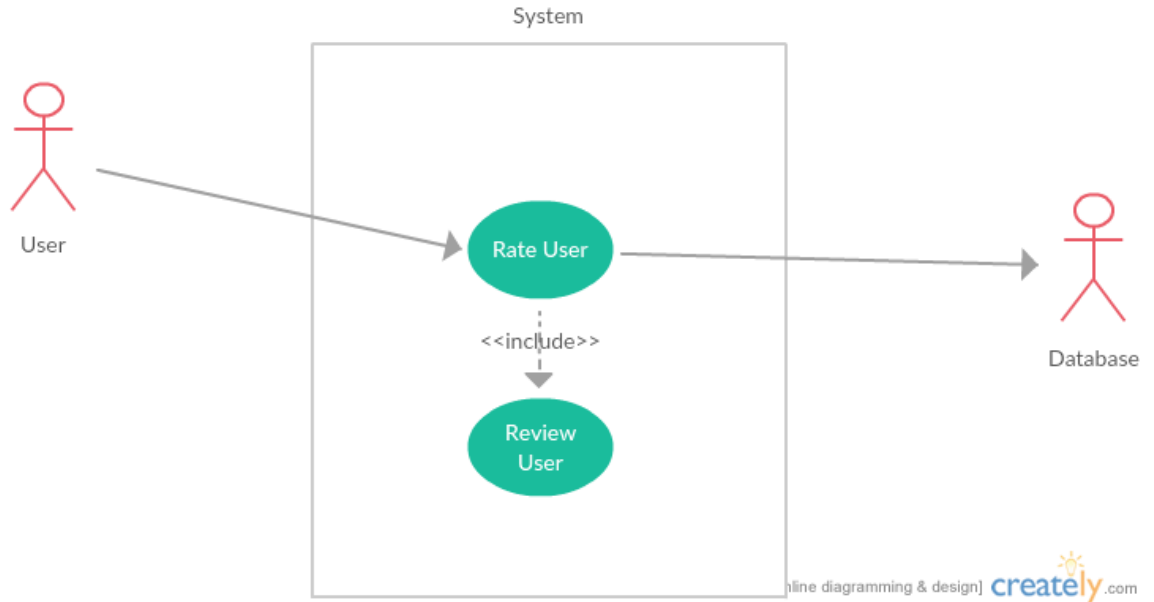
Scope

The scope of this use case is to enable a logged in user to review other users of the application.

Description

When a logged in user selects another users profile, he/she has shall have the option to review this particular reviewer in text format and add a rating (out of 10).

Use Case Diagram



Flow Description

Precondition

The user is logged in and an internet connection is active.

Activation

This use case begins when a logged in user selects the profile of another user.

Main flow

1. The user may rate the user a score out of 10
2. The user may review the user in plain text
3. The user clicks the Review button
4. The system will save the details to the database against that user's profile

Alternate flow

1. The user may select another user via the Share page
2. The user continues from the first step of the main flow

Exceptional flow

E1 :

1. The user loses connection to the internet and the transaction to the database cannot commit. The data is not saved.

Termination

The logged in user is returned to the main page.

Post condition

The other user now has a rating or review against them.

2.2.8 Requirement 7: View Cinemas***2.2.8.1 Description & Priority***

The user shall be able to view nearby cinemas and link to them directly within the app. This is important to be able to round off the features in the application.

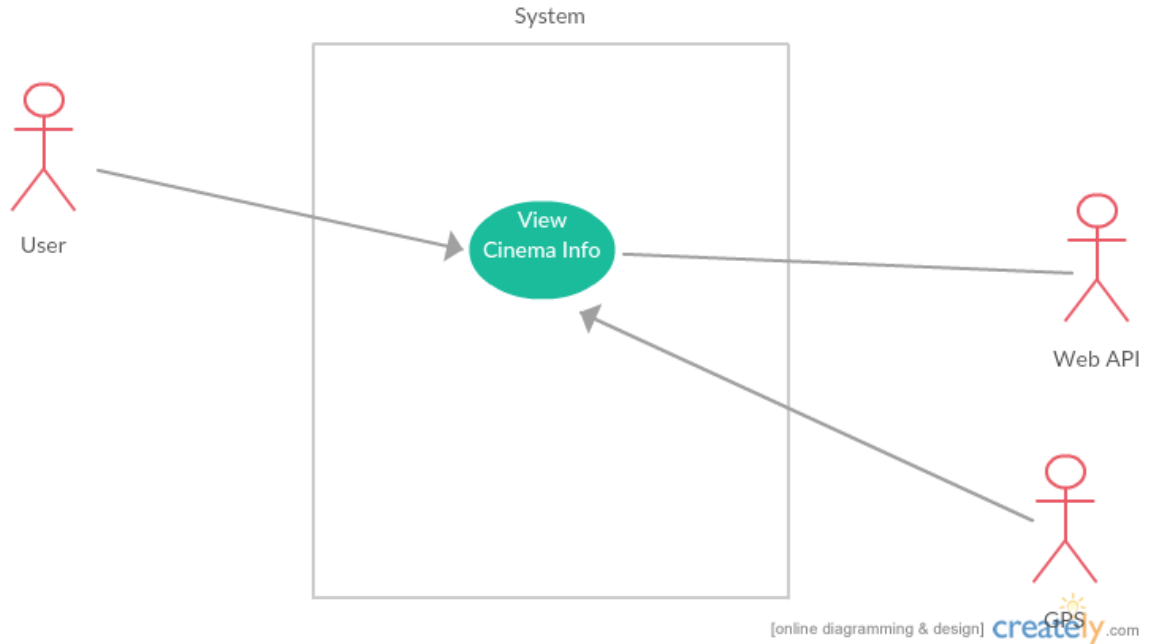
2.2.8.2 Use Case Priority 7**Scope**

The scope of this use case is to allow a user to see nearby cinemas to his/her GPS location with a link to the website of each.

Description

When a user clicks the “Cinemas” link, the application will display a list of nearby cinemas. The user can then click on these to be taken to the relevant website.

Use Case Diagram



Flow Description

Precondition

The user is logged in and an internet connection is active.

Activation

This use case begins when a user is logged in.

Main flow

1. The user may select the “Cinemas” link from the homepage
2. The system will display query the web APIs
3. The device GPS will return the current co-ordinates
4. The system will find nearby cinemas to the current co-ordinates
5. The system will display a list of nearby cinemas

Alternate flow

1. None Identified

Exceptional flow

E1 :

1. The device GPS is not given permission to activate.
2. The cinemas list is blank

Termination

The system returns cinema lists via the APIs.

Post condition

The system displays the cinemas to the logged in user.

2.3 Non-Functional Requirements

2.3.1 Performance/Response time requirement

The system shall be responsive and a user shall not have to wait more than 2 seconds for a registration or login be completed. Any longer and the app will make the user feel that there is a delay. Additionally, the user shall not wait longer than 2 seconds for database Creates, Reads, Updates or Deletes to take place. The application should also support an unlimited number of concurrent users. The database shall be sufficiently big to accommodate the relevant data.

2.3.2 Security requirement

The system shall be sufficiently secure so as to only allow registered users to login. On registration, user passwords shall be hashed and given salt values to improve security. Pages in the system will not display unless a user has logged in.

2.3.3 Reliability requirement

The system shall be entirely self-sufficient and not require maintenance from outside parties, except in emergency circumstances. Availability shall be greater than 99.9%. The system should not lose data in the event of circumstances which might trigger this.

2.3.4 Usability requirement

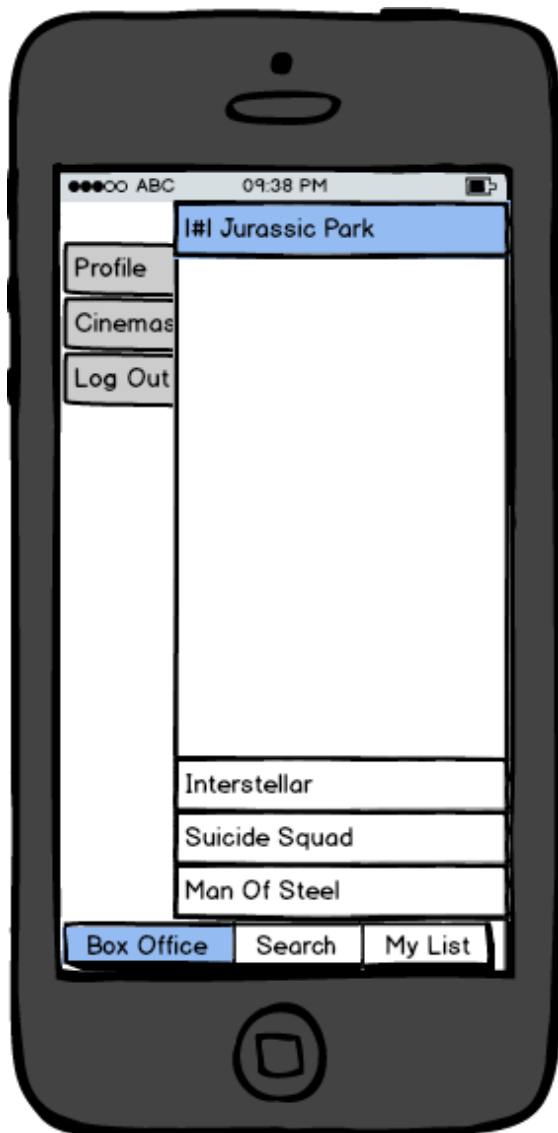
The system shall be usable by novice users of smartphones and internet websites. No specific level of expertise shall be required for any functions. Mobile responsive interfaces shall be used throughout with ease of use paramount for the user experience. Documentation shall be available by way of project technical document.

2.4 Interface requirements

The system shall interface with a MySQL database and its various tables through a language such as PHP. The system shall correctly interact with web based APIs including, but not limited to, freely available APIs such as the OMDb & Google Maps, and custom web scraping APIs where necessary. Web standards such as geo-location functionality shall be supported.

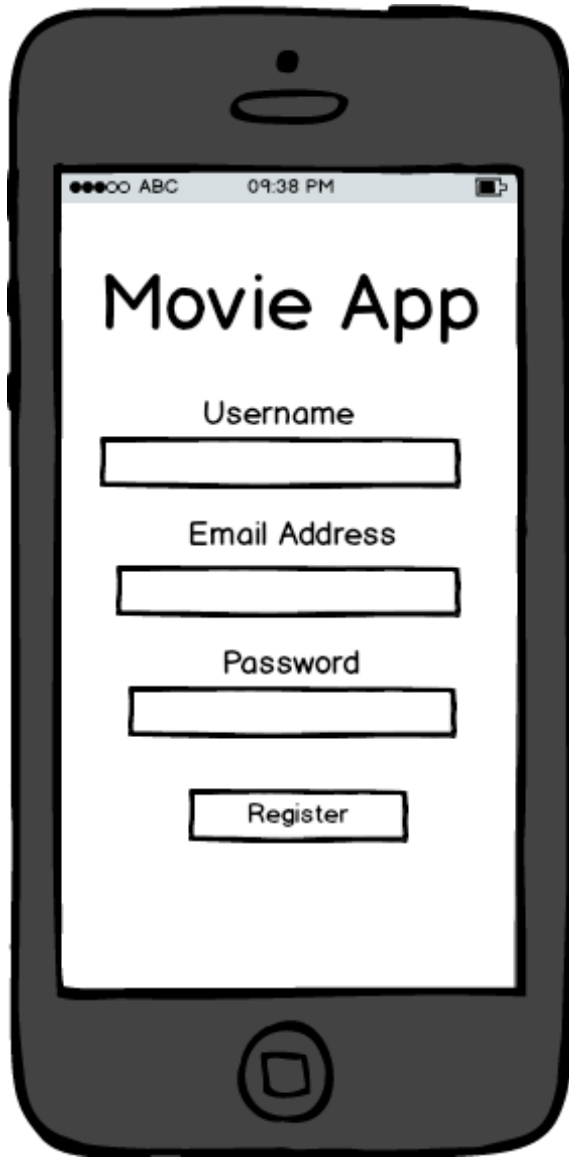
2.5 GUI

The below mockups are a look at the menu items and pages of the proposed design, there will be a side menu, accessible when clicking a link icon at the top left and a menu along the bottom:

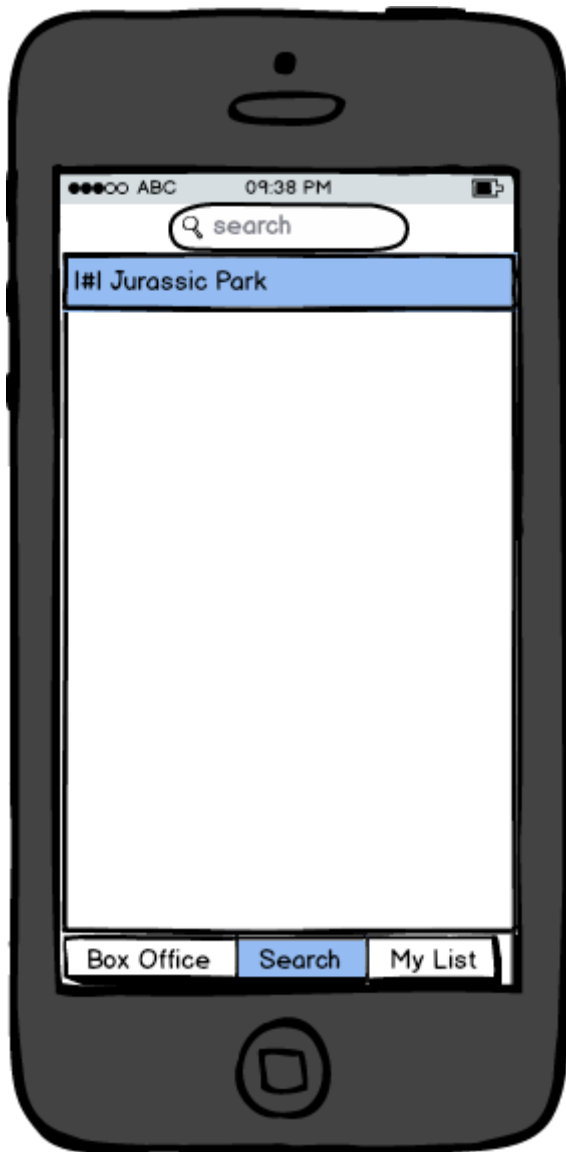


The user login & registration:





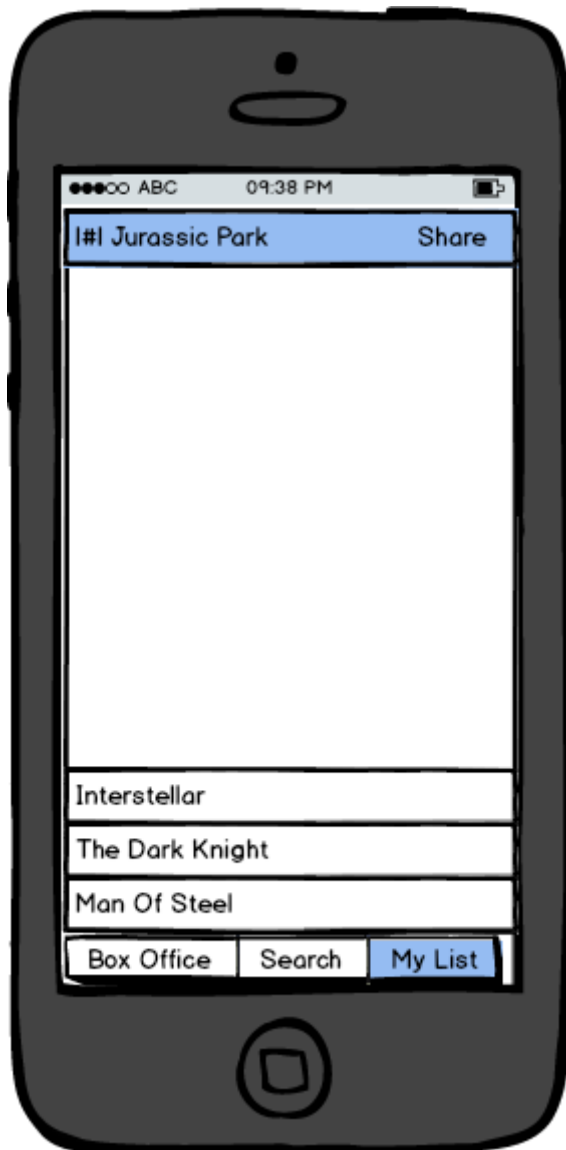
Search function, accessed by clicking the search link:



The rate movie function screen, accessed by clicking on a searched movie:



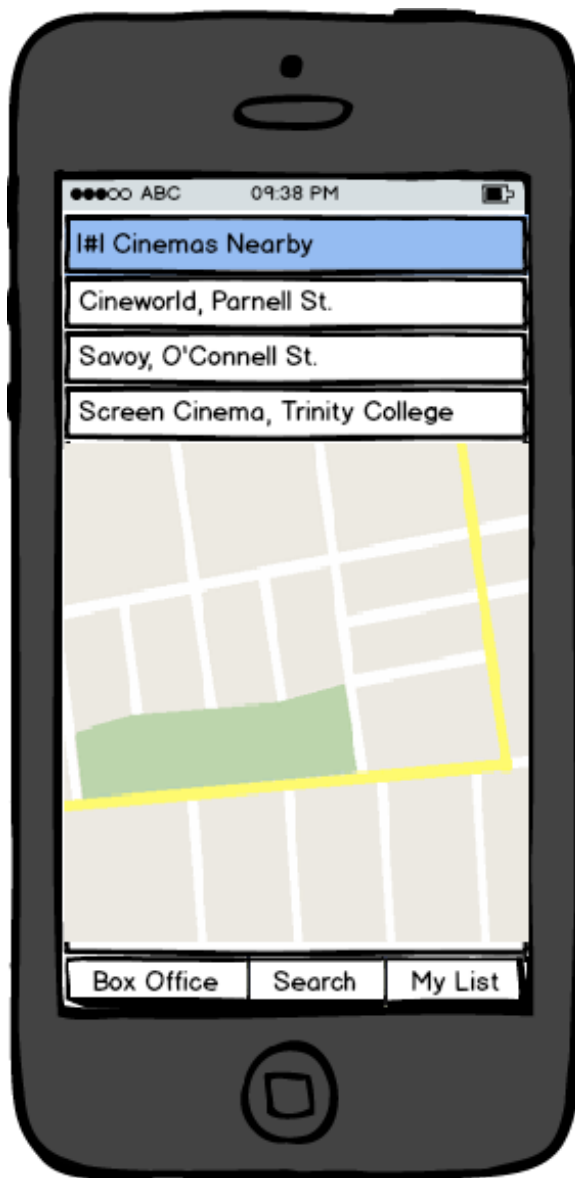
The share button on the My List page:



The user review page, accessed by clicking on a user's profile:



The nearby cinemas page:



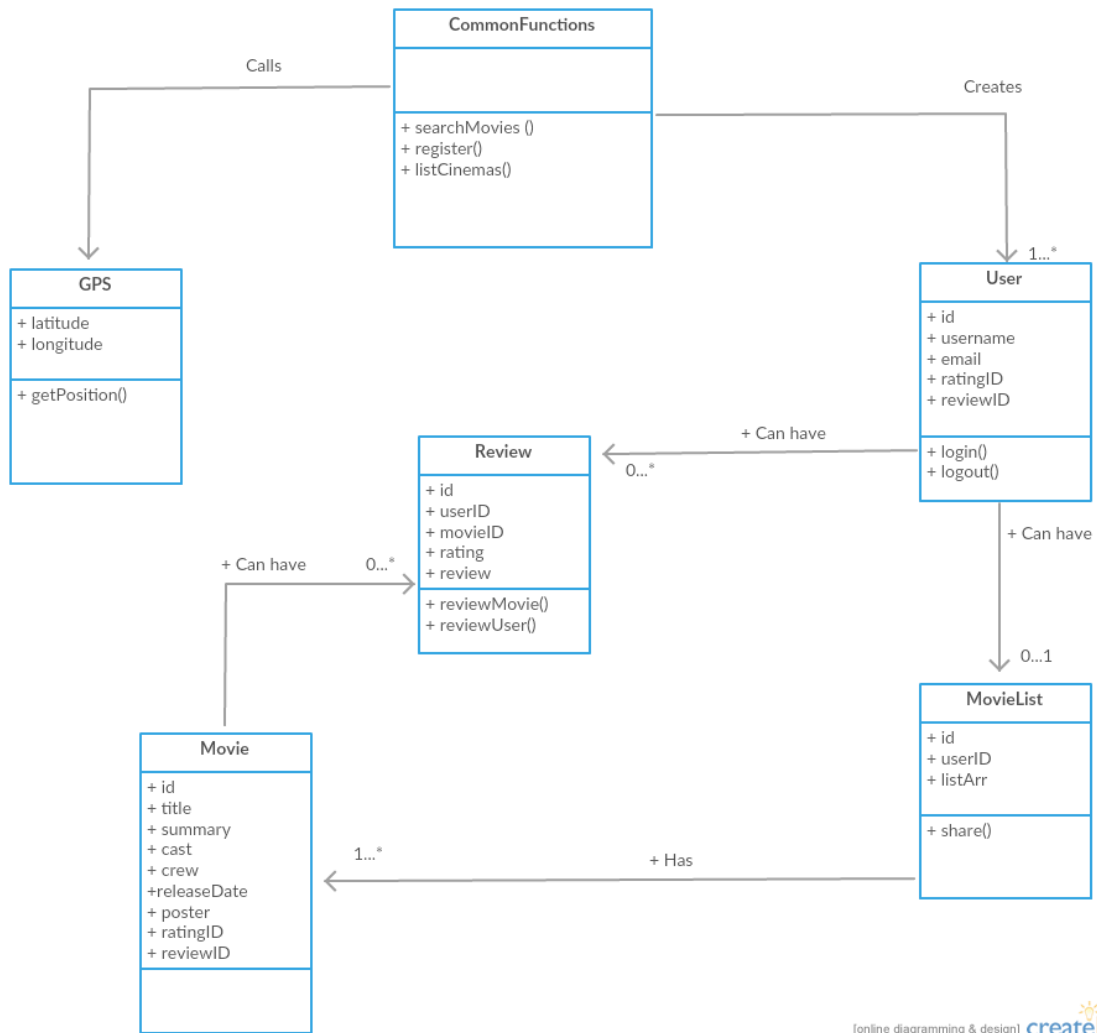
2.6 Application Programming Interfaces (API)

The system shall interact with the Open Movie Database freely available API to enable the system gather information about movies and TV shows. An API to scrape extra information from various sources may also be used. The user shall only need to enter the title of the piece, and the API shall return information such as, synopsis, cast, director, writer, release date, picture and so on. Google maps may also be used as an API for location-specific information such as cinema listings.

jQuery mobile may be used in site design to develop mobile ready, pleasing page design.

2.7 System Architecture

The following class diagram outlines the architecture behind the application. It is split out into several functions such as User, Review, Movie, MovieList etc. This enables easy relationships to be drawn between each of the functions. Many of these shall also have their own tables in the database, i.e. User table, Review table, where the ids can be matched.



2.8 System Evolution

Over time, the system could grow add more features such as having operating system specific design standards, like iOS or Android specific layouts and dedicated native applications. It could also grow to include community specific features such as having discussion forums for movie fans. As popularity grows, targeted ads could be added to the site, allowing for revenue generation. The dedicated native applications could also offer upgraded versions for a few euro each on the respective app stores with ads removed and some upgraded features, like custom avatars for users and additional content.

3 Design and Architecture

3.1 Implementation

Movie-io was implemented from the ground up using jQuery Mobile, PHP, MySQL and HTML5/CSS3. Programming began in late 2016 for the mid-point presentation and continued into 2017 when the bulk of the development work took place. This section will focus on the implementation of the main functionalities of the app and the challenges they presented.

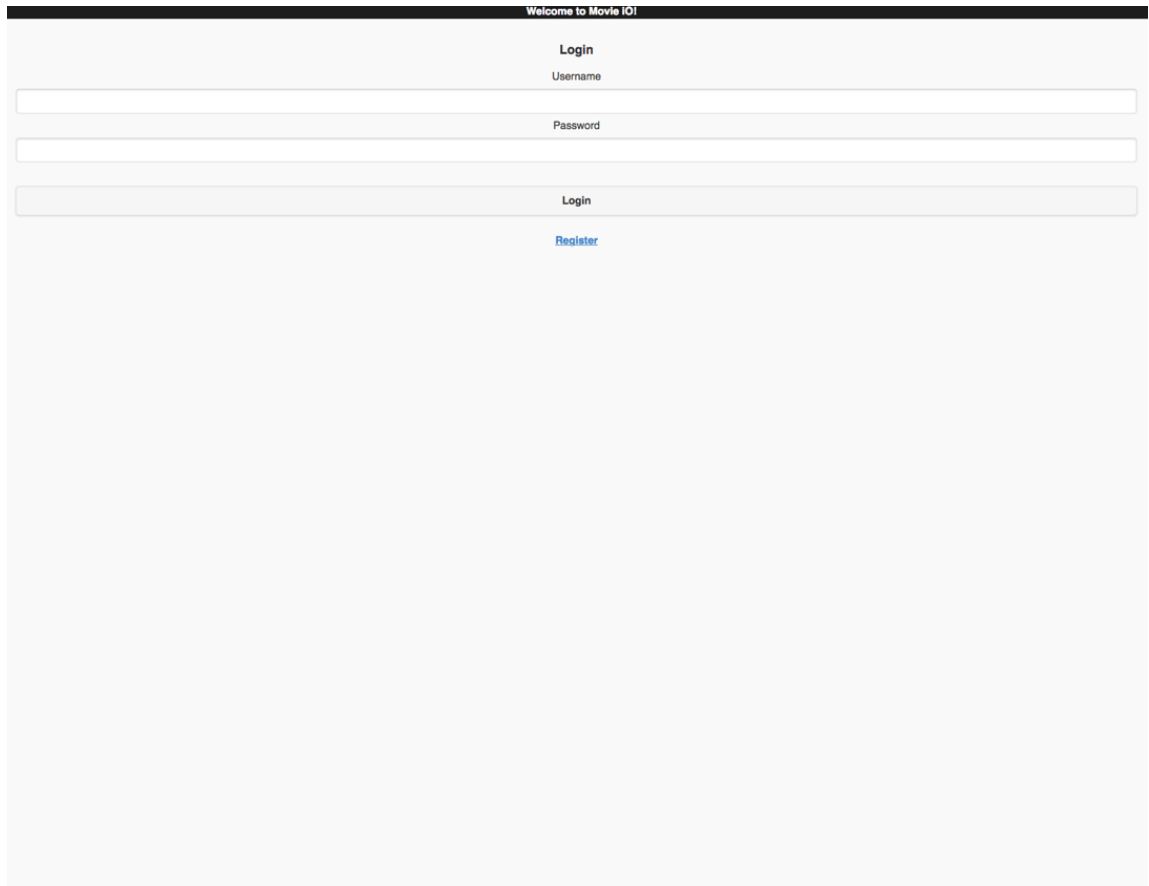
The application uses ajax extensively to pass data between the client and server. There are two javascript files implementing various functions – main.js and geolocation.js. For separation of concerns and ease of development and bug fixing, the php files are separated into one file per function i.e. login.php, load_movie_reviews.php etc. The HTML markup is contained within the index.php file and I used jQuery Mobile for layout. This meant I could use the one HTML file for most pages (excluding specifically the public movie review page which is shared via facebook, explained further below). Each jQuery Mobile page is identified by a data-role and associated id, preceded by a hashtag i.e. #main-page. This is what will appear on the URL. Additionally, I implemented a function in the main.js called checkLogin() which is run each time a page is loaded – this checks to see whether the user is logged in, and if not, redirects them to the login page.

The first function that is called when the login page is loaded is the Geo Location function. This calls getPosition() in the geolocation.js file to request the users location – once the user allows this, the latitude and longitude co-ordinates are saved to session variables for use later on in the app.

3.1.1 User Login & Registration

User Login and Registration was developed with security in mind. A user can register a username, email and password. The HTML form passes these details

via ajax to register.php. This checks if both the username and email are already currently in use, and if not, inserts the data to the login table in the database. For added security, a salt value is generated each time and the password is hashed with the salt value. Additionally, the hashed password is hashed again 65536 times in order to protect against brute force attacks.



The image shows a web form for logging in. At the top, there is a black header bar with the text "Welcome to Movie IQ!". Below this, the form is centered and contains the following elements:

- A heading "Login" in bold.
- A label "Username" above a text input field.
- A label "Password" above a text input field.
- A "Login" button.
- A blue link labeled "Register" below the login button.

Sign Up

Username

Email Address

Password

Submit

[Login](#)

```

if($row)
{
    echo("This email address is already registered");
}

// Insert details into DB
$query = "
    INSERT INTO login (
        username,
        password,
        salt,
        email
    ) VALUES (
        :username,
        :password,
        :salt,
        :email
    )
";

// generate salt to help with hashing password
$salt = dechex(mt_rand(0, 2147483647)) . dechex(mt_rand(0, 2147483647));

// This hashes the password with the salt for security
$password = hash('sha256', $_POST['password'] . $salt);

// Hash the hash value 65536 more times to protect against brute force attacks
for($round = 0; $round < 65536; $round++)
{
    $password = hash('sha256', $password . $salt);
}

// Here we prepare our tokens for insertion into the SQL query. We do not
// store the original password; only the hashed version of it.
$query_params = array(
    ':username' => $_POST['username'],
    ':password' => $password,
    ':salt' => $salt,
    ':email' => $_POST['email']
);

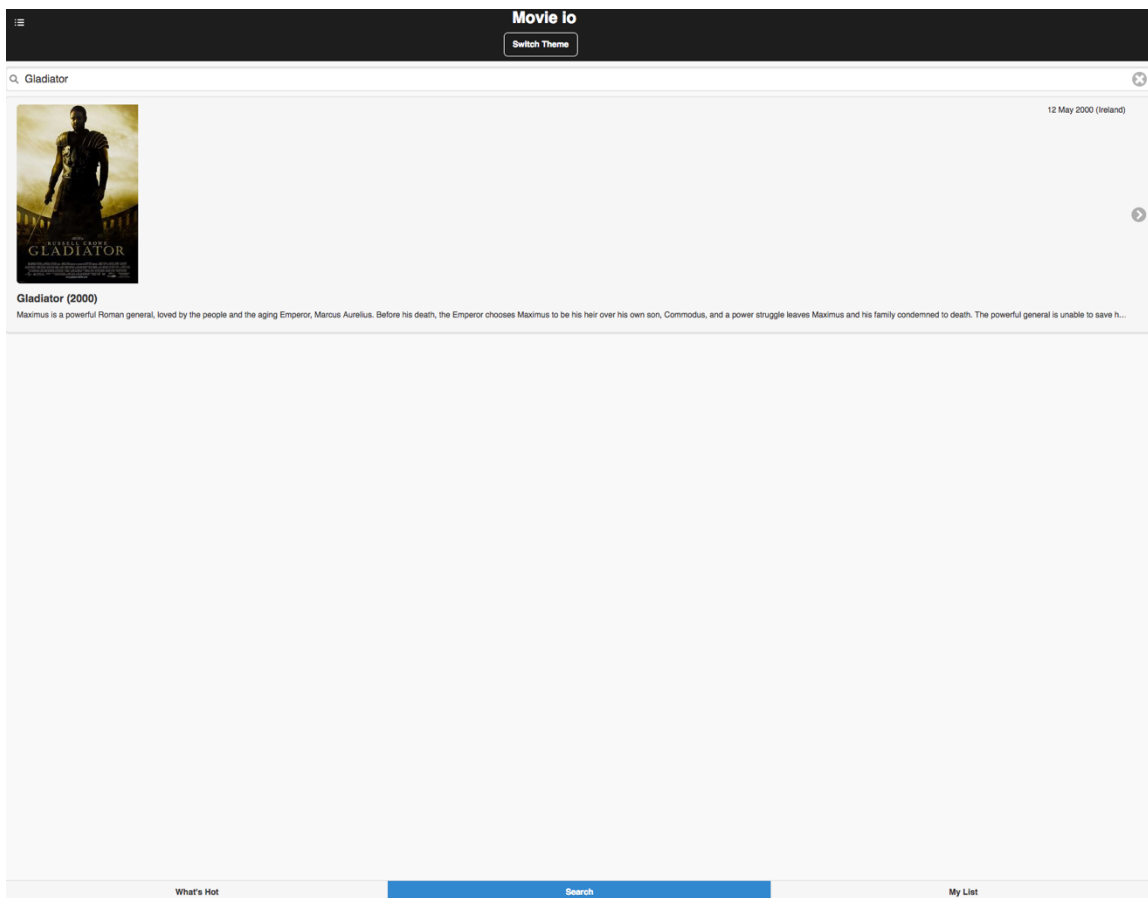
```

Login is similar in that the form passes the login name and password via ajax to login.php. The php script checks if the username exists, and if so, uses the salt value for that username and the submitted password to see if the passwords match, and if so, the use is logged in. The login and registration process were assisted via a post on the devshed.com forums and this is referenced in the files. The users profile values (if any) are then saved to PHP session variables for use later on in the application. The user is then redirected to the #main-page.

3.1.2 Web Scraper API

Upon login, the user is presented with the #main-page, which is a jQuery Mobile page. As explained earlier, each page in the application is a jQuery Mobile page, denoted by a hash in front of the page name in the URL. In the application itself, the #main-page is called 'What's Hot'. To the right of the 'What's Hot' link is the 'Search' link. This takes the user to the search page.

Here is one of the core features of the app and something that is required in order for the app to work. This lets the user search for a movie title and will return details for that movie, that the user can read and then review and rate.



This functionality is based around a web scraper which I wrote in PHP to grab movie details from the web. When a user enters a movie title (they can also enter

the year it was released to narrow the search) and presses the carriage return key, an ajax request is made to scraperv2.php with the movie title.

```
//search page function
$( document ).on( "pagecreate", "#search-page", function() {
    //alert("pageinit function loaded");
    checkLogin();
    $('#autocomplete-input').keypress(function (e) {
        if (e.which == 13) {

            var myMovie = $('#autocomplete-input').val();

            $.ajax({
                type: "POST",
                url: "scraperv2.php",
                data: "moviename="+myMovie,
                dataType : 'json',
                success: function(result){
                    //$("#message")[0].value = "Success";
                    //alert("Success! " + JSON.stringify(result.Title));
                    loadSearchData(result);

                },
                error: function(xhr, status, error){
                    //$("#message")[0].value = "Ajax error!" + result;
                    alert("Scraper JS error" + xhr.responseText);
                }
            });

            return false;
        }
    });
});
```

Scraperv2.php automatically queries www.imdb.com for the movie title it just received. It returns the URL for that title, and this is then used again to query the site page of that specific movie. Once found, the entire HTML markup content of that page is returned to the script and I then use the find() function to query parts of the HTML structure to find the title, release date, plot information, actors, director, awards, metascore and poster details for the movie. For example to find the title I use "find("h1[itemprop=name]", 0)" where I know that it will be located in

the first instance of h1 where itemprop is equal to name. I include a freeware php DOM parser called Simple HTML DOM in the code to enable these functions. The results are saved to variables and returned in the ajax response via JSON encoded array.

```
<?php
/*
 * @reference: DOM Parser http://simplehtmldom.sourceforge.net/
 */
include('simple_html_dom.php');
//ini_set('display_errors', 1);

// parameter
$query_params = $_POST['moviename'];
//replace spaces with +
$query_params = str_replace(' ', '+', $query_params);

// Load IMDB page
$url = 'http://www.imdb.com/find?ref=mv_sr_fn&q=' . $query_params . '&s=all';
$html = str_get_html(file_get_contents($url));

//get movie title href
$title_url = $html->find("td[class=result_text] a", 0)->href;

//save movie title url
$url2 = 'http://www.imdb.com' . $title_url;
$act_page = str_get_html(file_get_contents($url2));

//start getting movie
$title = $act_page->find("h1[itemprop=name]", 0)->plaintext;
$release_date = $act_page->find("a[title='See more release dates']", 0)->plaintext;
$plot = $act_page->find("div[class='inline canwrap'] p", 0)->plaintext;
$actors = "";
foreach ($act_page->find("span[itemprop=actors] a") as $key) {
    $actors = $actors . $key->plaintext . ', ';
};
$director = $act_page->find("span[itemprop=director] a", 0)->plaintext;
$awards = "";
foreach ($act_page->find("span[itemprop=awards]") as $key) {
    $awards = $awards . $key->plaintext . ', ';
};
$metascore = $act_page->find("div[class=metacriticScore]", 0)->plaintext;
$poster = html_entity_decode(($act_page->find("div[class=poster] a", 0)->children(0)->src), ENT_COMPAT, 'UTF-8');

//return values
echo json_encode(array(
    "Title" => $title,
    "Released" => $release_date,
    "Plot" => $plot,
    "Actors" => $actors,
    "Director" => $director,
    "Awards" => $awards,
    "Metascore" => $metascore,
    "Poster" => $poster
));

?>
```

This in turn is then passed to two other javascript functions, loadSearchData() and loadMovieData(), which populate the preview on the search page, and the dedicated movie page itself within the app respectively.

```
function loadSearchData(e){
  //add movie title to global variable
  movieTitle = e.Title;
  $('#autocomplete').html(
    '<li class="ui-first-child ui-last-child"><a href="#movie-page" class="ui-btn ui-btn-icon-right ui-icon-carat-r"> '+
    '<h2 id="searchTitle">' + e.Title.replace("&nbsp;"," ") + '</h2>'+
    '<div><p>' + e.Plot + '</p></div>'+
    '<p class="ui-ll-aside">' + e.Released+ '</p></a></li>'
  );
  loadMovieData(e);
};

//movie page function
function loadMovieData(e){
  //check for &nbsp; in title
  //alert("title is " +e.movie_title);
  var title = "";
  if((e.Title.indexOf("&nbsp;")) != -1){
    title = e.Title.replace("&nbsp;"," ");
  }
  else{
    title = e.Title;
  }
  $('#title').html('<h2>' + title + '</h2>');
  $('#releasedate').html('Release Date: ' +e.Released);
  $('#synopsis').html(e.Plot);
  $('#starring').html(e.actors);
  $('#awards').html(e.Awards);
  $('#metascore').html(e.Metascore);
  $('#posterdiv').html('');
  //load star ratings
  $('#ratingdiv').raty({
    starOff: 'images/star-off.png',
    starOn: 'images/star-on.png',
    size: 24
  });
  //now load reviews
  loadReviews(e);
}
```

Getting the scraper working was initially quite challenging. I originally created the scraper in Python using BeautifulSoup (see scrape.py). Although it worked, I could not get the movie title to be passed to the python script from the application. Additionally I had difficulty returning data in the correct format to the application via ajax. I also found it impossible to integrate the python script with the rest of the application on both localhost and a server on the internet – there were simply too many differences. I spent many hours trying to resolve it before finally deciding to completely re-write it in PHP for the sake of simplicity and keeping with the rest of the code architecture.

3.1.3 Movie Reviews

Once a movie is loaded from the web scraper, a user can see it's details including release date, plot, cast and the poster, among others.

Movie io

Switch Theme

Gladiator (2000)

Release Date: 12 May 2000 (Ireland)

Plot: Maximus is a powerful Roman general, loved by the people and the aging Emperor, Marcus Aurelius. Before his death, the Emperor chooses Maximus to be his heir over his own son, Commodus, and a power struggle leaves Maximus and his family condemned to death. The powerful general is unable to save his family, and his loss of will allows him to get captured and put into the Gladiator games until he dies. The only desire that fuels him now is the chance to rise to the top so that he will be able to look into the eyes of the man who will feel his revenge. Written by Chris "Morph" Terry

Starring: Russell Crowe , Joaquin Phoenix , Connie Nielsen

Awards: Won 5 Oscars. , Another 53 wins & 101 nominations.


Metascore: 64

☆☆☆☆☆

What did you think?

Submit

What's Hot Search My List



Also on this page a user can submit his/her rating and review of the movie. The rating is out of 5 (rating stars are part of the free 'Raty' plugin, referenced in index.php) and the user can write their thoughts in the text area. On submit, the saveMovie() function is called which passes the movie details into a javascript FormData object and passes it to save_movie.php. This checks first to see if a movie with that title already exists in the database, and if not, saves the data to the movies table. It returns the id of that insert to the saveMovie() function if successful.

```

//insert movie data
function saveMovie(){
  var title = $('#title').text().replace("&nbsp;","").trim();
  var release_Date = $('#releasedate').text().replace("Release Date:", "").trim();
  var synopsis = $('#synopsis').text().trim();
  var starring = $('#starring').text().trim();
  var awards = $('#awards').text().trim();
  var metascore = $('#metascore').text().trim();
  var poster = $('#posterimg').prop('src');

  var form_data = new FormData();
  form_data.append("poster", poster);
  form_data.append("title", title);
  form_data.append("release_date", release_Date);
  form_data.append("synopsis", synopsis);
  form_data.append("starring", starring);
  form_data.append("awards", awards);
  form_data.append("metascore", metascore);

  $.ajax({
    type: "POST",
    url: "save_movie.php",
    data: form_data,
    contentType: false,
    processData: false,
    //$("#title="+title+"&release_date="+release_Date+"&synopsis="+synopsis+"&starring="+starring+"&awards="+awards+"&metascore="+metascore,
    success: function(result){
      //$("#message")[0].value = "Success";
      //alert("Movie save Success! " + result);
      sessionStorage.setItem("movie_id", result);
      //now also insert review
      insertReview();
    },
    error: function(xhr, status, error){
      //$("#message")[0].value = "Ajax error!" + result;
      alert("Movie save JS error!" + xhr.responseText);
    }
  });
}

```

The insertReview() function is then called to save the users review and score to the database in the table movie_reviews. It saves the aforementioned movie id along with the score, movie details, review text, date and user id to the database, via the review_movie.php script. It also saves the GPS co-ordinates of the user. These were first gathered when the app was loaded if the user agreed to their location being known. This is in order to show location specific movie data in the What's Hot page – described further below.

```

//insert movie review
function insertReview(){
  var lat = sessionStorage.getItem("userLat");
  var lng = sessionStorage.getItem("userLong");
  console.log("lat: " + lat + " lng: " + lng);
  var review = $('#txtAreaReview').val().trim();
  var score = $('#ratingdiv').raty('score');
  var title = $('#title').text().replace("&nbsp;","").trim();
  var release_Date = $('#releasedate').text().replace("Release Date:", "").trim();
  var user_id = $('#sessionuserid').text();
  var movie_id = +sessionStorage.getItem("movie_id");
  //console.log("movie id=" + movie_id);
  //var movie_id = movie_id.replace("'", '').trim();
  //console.log("movie id=" + movie_id);
  //alert("review text: " + review + ", user id: " + user_id + ", title: " + title + ", release: " + release_Date + ", score " + score + " LatLng: " + lat + " + lng + " movie
  $.ajax({
    type: "POST",
    url: "review_movie.php",
    data: "review="+review+"&title="+title+"&release_date="+release_Date+"&user_id="+user_id+"&score="+score+"&lat="+lat+"&lng="+lng+"&movie_id="+movie_id,
    success: function(result){
      //alert("Review save Success! " + JSON.stringify(result.Title));
      //reload reviews
      loadReviews(null, title);
    },
    error: function(xhr, status, error){
      //$("#message")[0].value = "Ajax error!" + result;
      alert("Review JS error!" + xhr.responseText);
    }
  });
}

```

Once the review is saved, the movie page will automatically update to add the users review below the submit button, in the reviews section. This is done via the loadReviews() function which is called once the review is successfully saved. This passes the title to load_reviews.php, which queries the database to find all reviews for that specific movie and returns them to the function in a JSON encoded array.

```
//start
if(!empty($_POST))
{
    // parameters
    $query_params = array(
        ':movie_title' => $_POST['title']
    );

    //Insert
    $query = "
SELECT a.review, a.score, b.username, b.location, b.avatar, b.id, a.date from movie_reviews a
JOIN login b on (a.user_id = b.id)
WHERE
    a.movie_title = :movie_title
";

    try
    {
        // run query
        $stmt = $db->prepare($query);
        $result = $stmt->execute($query_params);
    }
    catch(PDOException $ex)
    {
        echo("Failed to run query: " . $ex->getMessage());
    }

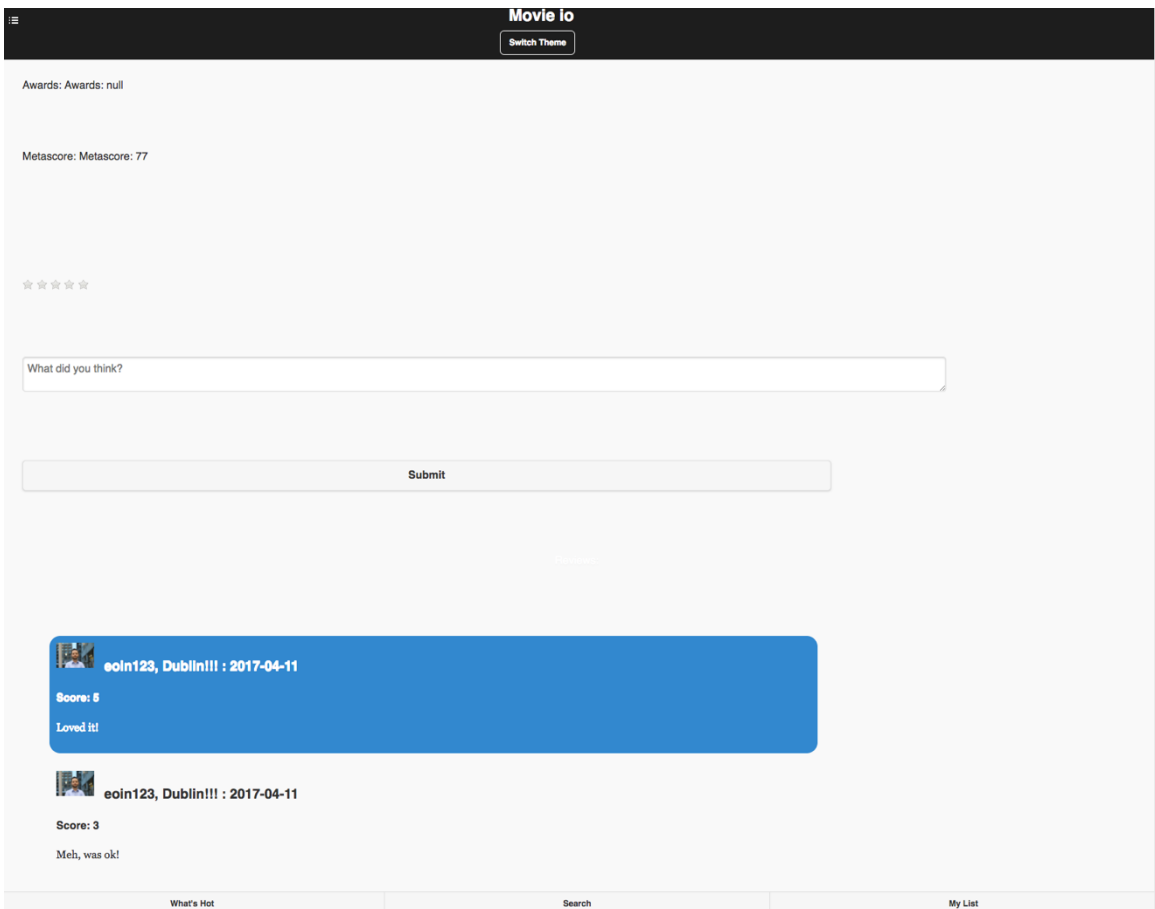
    $encode = array();
    while ($row = $stmt->fetch()) {
        $encode[] = $row;
    };

    echo json_encode($encode);
}
```

The function then loops through each element of the array and appends a HTML unordered list (ul) to the reviews section of the movie page, with a list item for each review. This will contain the username of the person who submitted the review, the date of the review, the location, their score/review and their profile picture from the database.

```
//find movie reviews
function loadReviews(e, title){
  if(e){
    var title = e.Title.replace("&nbsp;"," ").trim();
  }
  else{
    var title = title.replace("&nbsp;"," ").trim();
  }
  console.log("title is " + title);
  $.ajax({
    type: "POST",
    url: "load_reviews.php",
    data: "&title="+title,
    success: function(result){
      //$("#message")[0].value = "Success";
      //alert("Movie fetch success! " + result);
      result = $.parseJSON(result);
      console.log(result);
      $('#reviews').empty();
      $.each(result, function (key, value) {
        var list = $('<ul></ul>');
        $('#reviews').append(list);
        console.log("Key: " + key + " value: " +value.review);
        list.append("<li onclick='goToPublicProfile(\x22" + value.id + "\x22)'>" + "<div style='width:5%; height:5%;><img src='" + value.avatar +
          + "<p>" + value.review + "</p>" + "</li>");
      });
    },
    error: function(xhr, status, error){
      //$("#message")[0].value = "Ajax error!" +result;
      alert("Movie fetch error " + xhr.responseText);
    }
  });
};
```

The loadReviews() function is also called when the movie data has loaded from the web scraper, so all reviews are visible when a movie page loads.



Another important occurrence here is that `loadReviews()` creates each list item with an `onclick` function. This contains the returned user id from the `load_reviews.php` query. I did this so that `onclick` of a user review, the `goToPublicProfile(user_id)` function will be called that will redirect the user to the public profile page of the other user who wrote the review. This is explained in more detail below.

The challenges here included ensuring the SQL joins were correct for the `load_reviews.php` query so that both user and review data was returned, and ensuring the HTML I was writing for the user reviews would fit and size correctly on the movie page.

3.1.4 User Reviews

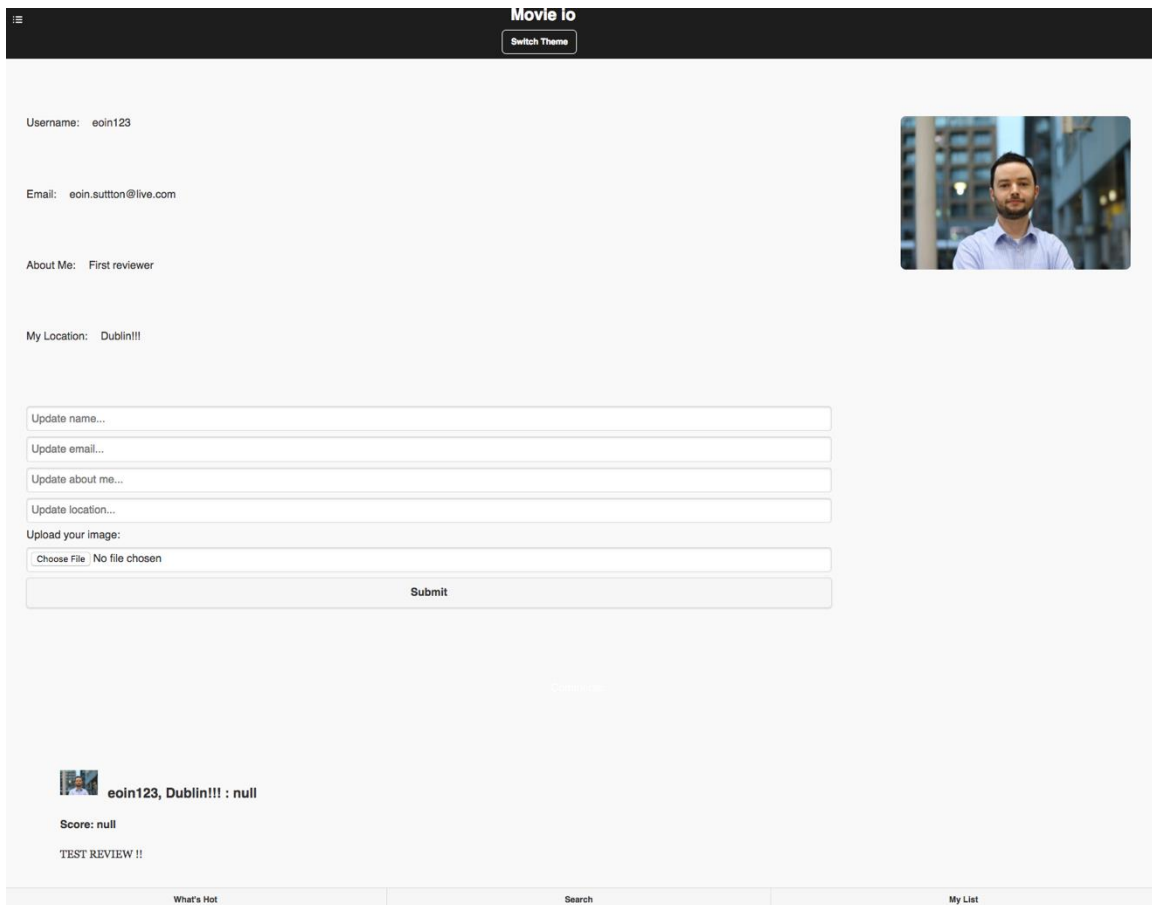
Part of the functionality of the app is that users can 'review the reviewer'. To enable this, I had to create a private profile page and a public profile page for users. The private one is where the user is logged in and they can update their avatar (image), email, username, location and say a little about themselves. They could also view reviews from other users of their contributions to the site. The public page is where other users could view information about you, and add their comments to your profile.

On click of the bullet icon on the top left of the app, a menu drawer opens with some further links. One of these is for the Profile page. The user data here is populated in two ways. On initial login, a users details are saved to PHP session variables which I call in the HTML for the profile page.

```
<div id="profilediv" class="ui-panel-wrapper">
<div id="namediv" class="mainDivs">Username: <span id="namespan"><?php echo $_SESSION['username']; ?></span></div>
<div id="profileavatar" style="width:20%; height:20%; margin-top:5%; margin-right:2%; float:right;">Email: <span id="emailspan"><?php echo $_SESSION['email']; ?></span></div>
<div id="aboutmediv" class="synopsis">About Me: <span id="aboutmespan"><?php echo $_SESSION['about_me']; ?></span></div>
<div id="locationdiv" class="mainDivs">My Location: <span id="locationspan"><?php echo $_SESSION['location']; ?></span></div>
<p>
<form id="formdata" method="post" enctype="multipart/form-data" class="mainDivs">
<!--div id="updatenametextarea" style="width:40%; height:80%; margin-top:5%; margin-left:2%; float:left">
<textarea id="updatenametextarea" placeholder="Update Name..."></textarea>-->
<input type="text" name="username" placeholder="Update name...">
<input type="text" name="email" placeholder="Update email...">
<input type="text" name="about_me" placeholder="Update about me...">
<input type="text" name="location" placeholder="Update location...">
Upload your image: <input type="file" name="myfile" id="filetoupload">
<button onclick="updateProfile2()">Submit</button>
```

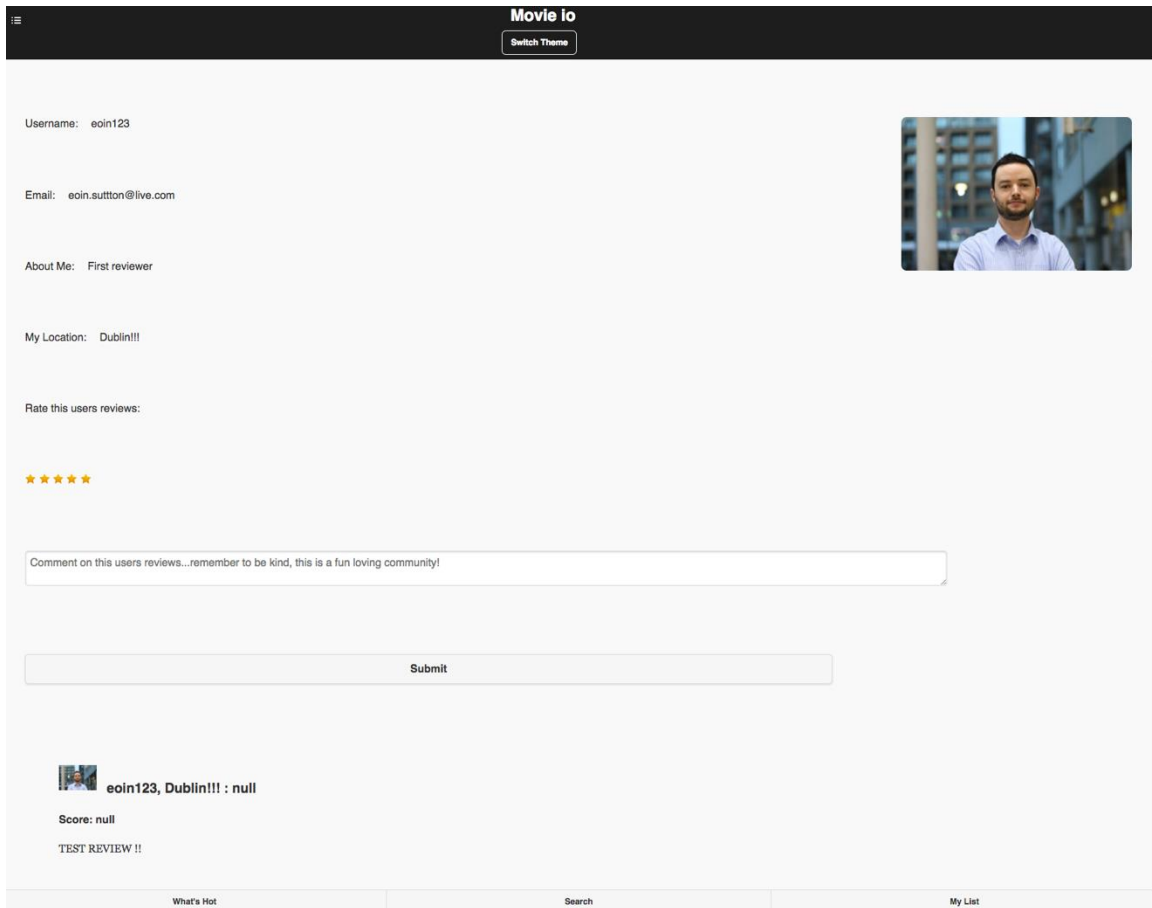
However I had a challenge here where the session data is not available until after a refresh of the page. I didn't want to have to force a refresh so inside the `login()` function I also populate these same divs at login time.

Another challenge was saving the profile picture. A user has the option to upload a picture for their profile. Initially I had set this up to convert the chosen picture to base64 encoded text and saved that in the database. However calling this was difficult and using the HTML Data URIs (e.g. `src="data:image/png;base64"`) resulted in issues like only half of the image being rendered, and resulted in a less snappy experience for the user (reloads were slower). I decided to save the images directly to the `/images` folder on the server and simply reference their location for display.



Loading the profile page also calls the `loadUserReviews()` function which, like the `loadReviews()` function on the movie page, loads the comments users have left for you. This passes a GET request to `load_user_reviews.php` which grabs the data from the `user_reviews` table and uses the user id that was stored in the PHP `$_SESSION['user']['id']` variable on login.

The public profile page is available by clicking on any of the reviews/comments left on either the movie pages or your user profile. These use the `goToPublicProfile(user_id)` function mentioned earlier, which is activated onclick. The user is then redirected to the public page which is a carbon copy of the private profile page, but with the score and review/comments section in place of the text fields for updating usernames etc.



Users can submit comments in a similar manner to submitting reviews for movies. `insertReview2()` is called when submitting here, passing the values to `review_user.php`. `loadPublicUserReviews()` is called on page load, which in turn loads the reviews from the `user_reviews` table for that person id, via `load_public_user_reviews.php`.

3.1.5 What's Hot - Location specific movie data

The 'What's Hot' page (the #main-page) contains a list of movies which are popular near the user. This functionality is location specific and only lists movie reviews where the score is greater than or equal to 3 and where the location of the reviews is within 100km of the users current location. This was designed so that users can get an idea of what is popular around them no matter their location. So what's popular around Paris will be different to what's popular around Dublin for example.

I implemented this by calling the loadGeoData() function from the geolocation.js file on page-create of the #main-page. This was developed to send a GET request to the get_location_data.php script which returns all movie reviews from the movie_reviews table where there are available latitude and longitude co-ordinates, and where the review score is ≥ 3 . The results are returned in a JSON encoded array.

```
$query = "
SELECT * from movie_reviews
WHERE lat is not null
AND lng is not null
AND score >= 3
";

try
{
    // run query
    $stmt = $db->prepare($query);
    $result = $stmt->execute($query_params);
}
catch(PDOException $ex)
{
    echo("Failed to run query: " . $ex->getMessage());
}

$encode = array();
while ($row = $stmt->fetch()) {
    $encode[] = $row;
};

echo json_encode($encode);
```

In the javascript function, I then get the current users latitude and longitude coordinates from the sessionStorage variables (saved when the app first loads). I then loop through each element of the returned array and pass the user coordinates and the co-ordinates of the review into a Haversine function.

A Haversine function is a function which calculates the great circle distance between two points.

```
//haversine function to return great circle distance between two points
LatLon.prototype.distanceTo = function(point, radius) {
  if (!(point instanceof LatLon)) throw new TypeError('point is not LatLon object');
  radius = (radius === undefined) ? 6371e3 : Number(radius);

  var R = radius;
  var v1 = toRad(this.lat), a1 = toRad(this.lon);
  var v2 = toRad(point.lat), a2 = toRad(point.lon);
  var v_total = v2 - v1;
  var a_total = a2 - a1;

  var a = Math.sin(v_total/2) * Math.sin(v_total/2)
    + Math.cos(v1) * Math.cos(v2)
    * Math.sin(a_total/2) * Math.sin(a_total/2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c;

  return d;
};

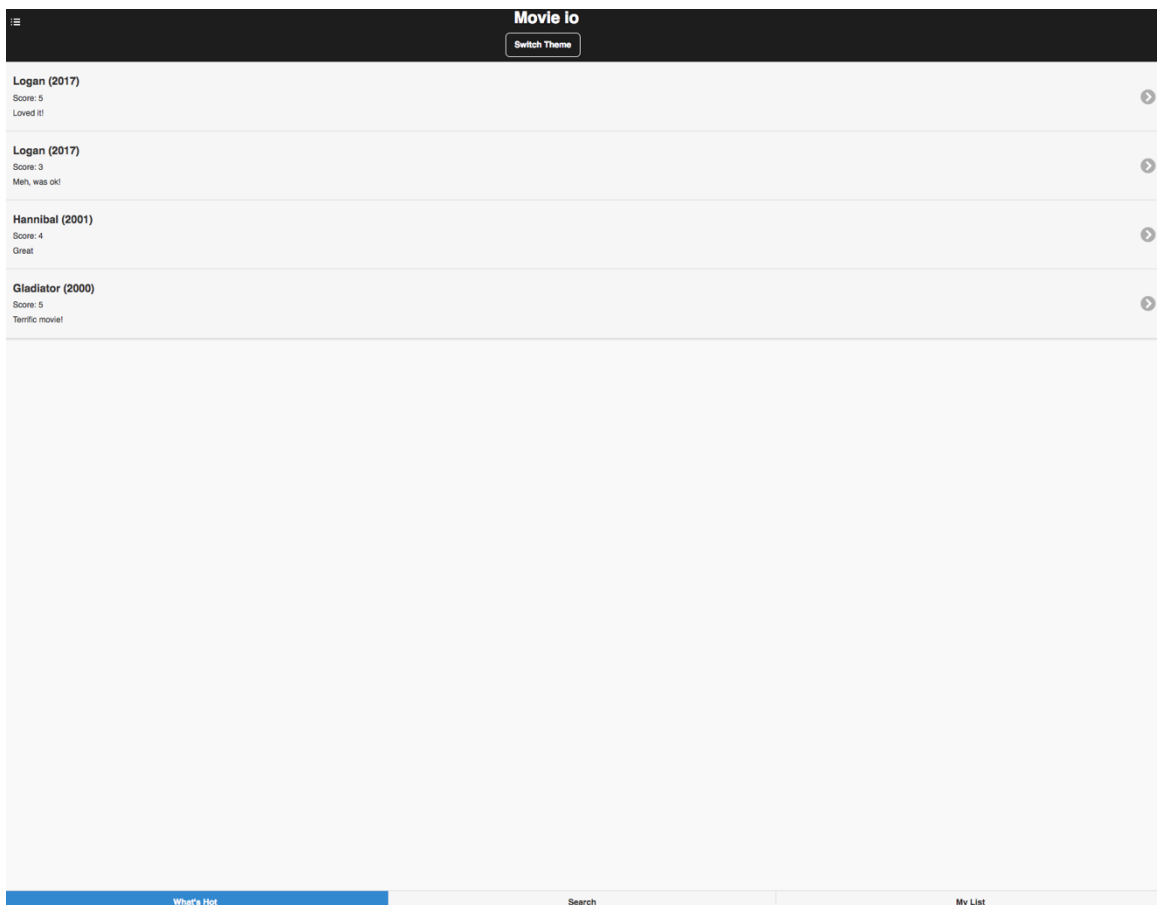
function LatLon(lat, lon) {
  // allow instantiation without 'new'
  if (!(this instanceof LatLon)) return new LatLon(lat, lon);
  this.lat = Number(lat);
  this.lon = Number(lon);
}

//calculate radians
function toRad(x) {
  return x * Math.PI / 180;
}
```

If the array element (thus, the review) was created at a location less than or equal to 100 kilometres from the users current location, a list item (li) is appended to an unordered list (ul) on the page with details of the movie title, score and review text.

```
//get location data from reviews
function loadGeoData(){
  //array of movie review ids where location is within 100km of user
  var arr = [];
  $.ajax({
    type: "GET",
    url: "get_location_data.php",
    success: function(result){
      result = $.parseJSON(result);
      console.log(result);
      //create user lat long object
      var userLoc = new LatLon(sessionStorage.getItem("userLat"), sessionStorage.getItem("userLong"));

      $.each(result, function (key, value) {
        var reviewLoc = new LatLon(Number(value.lat), Number(value.lng));
        //get distance
        var dist = (((userLoc.distanceTo(reviewLoc)) /1000).toFixed(2));
        if(dist <= 100){
          $('#listviewpage2').append("<li onclick='loadMovieDataFromList(\x22 + value.movie_id + "\x22)'><a href='#'>"
            + "<h2>" + value.movie_title + "</h2>"
            + "<p>Score: " + value.score + "</p>" + "<p>" + value.review + "</p>" + "</a></li>").listview('refresh');
        }
      });
    },
    error: function(xhr, status, error){
      alert("Geo location data fetch error " + xhr.responseText);
    }
  });
}
```



Additionally, when the list item is appended, an onclick function is referenced – loadMovieDataFromList() to which is passed the movie’s id.

This function, which is also used on the list items on the My List page, sends a POST request to `load_movies_from_list.php` with the id. This returns the result set of a select query on the movies table, joined with the movie_reviews table. It's returned back to the javascript function as a JSON encoded array and it then passes this to the `loadMovieData()` function which I mentioned earlier which in turn loads the data for the movie page. The user is then re-directed there. Being able to re-use code like this was really helpful.

Challenges here included getting the Haversine function working. It required creating custom LatLong objects and ensuring the latitude/longitude co-ordinates were being converted to 'radians' which are specific units of measure. I also needed to revise the query in the PHP file a number of times as it is used to gather the movie and some review data for the page.

3.1.6 Facebook Sharing and My List Page

The My List page contains a list of all of the movies the user has reviewed or rated, sorted by the top rated movies descending to the lowest rated. This is populated at pagecreate by a function that sends an ajax GET request to `load_mylist.php`.

```
//load MyList page movies
//on page load
$( document ).on( "pagecreate", "#list-page", function() {
  $.ajax({
    type: "GET",
    url: "load_mylist.php",
    success: function(result){
      //$("#message")[0].value = "Success";
      //alert("My List fetch success! " + result);
      result = $.parseJSON(result);
      console.log(result);
      $.each(result, function (key, value) {
        //$("#listviewpage").append(list);
        console.log("key: " + key + " value: " +value.review);
        $('#listviewpage').append("<li onclick='loadMovieDataFromList(\x22" + value.id + "\x22)'><a href='#'>"
          + "<h2>" + value.movie_title + "</h2>"
          + "<p> Your Score: " + value.score + "</p>" + "<p>" + value.review + "</p>" + "</a></li>").listview('refresh');
      });
    },
    error: function(xhr, status, error){
      //$("#message")[0].value = "Ajax error!" + result;
      alert("User review fetch error " + xhr.responseText);
    }
  });
  //toggleBtn();
});
```

This in turn returns a JSON encoded array of the results of a select query on the database – movie review details of movies this user id has submitted, ordered by score, descending. The same PHP file serves the `public-list-page.php` which

forwards and id parameter in a POST request, so the PHP file is split into two functions – one where the POST is not null, and the other where it is (i.e. it's a GET request). The GET request used for the My List page gathers the users id from the PHP session variables.

```
}
else{

// parameters
$query_params = array(
    'id' => $_SESSION['user']['id']
);

//query
$query = "
SELECT a.movie_title, a.release_date, a.review, a.score, b.id
FROM movie_reviews a join movies b on (a.movie_title = b.title)
WHERE user_id = :id|
ORDER BY a.score desc
";

try
{
    // run query
    $stmt = $db->prepare($query);
    $result = $stmt->execute($query_params);
}
catch(PDOException $ex)
{
    echo("Failed to run query: " . $ex->getMessage());
}

$encode = array();
while ($row = $stmt->fetch()) {
    $encode[] = $row;
};

echo json_encode($encode);
```


This gives the user a view of his movie reviews sorted by rating. At the top right of the page is the Facebook sharing icon. Clicking this will send a link to Facebook that the user can post on his/her page to share the movie list with their friends.

Movie IO Switch Theme


- Logan (2017)**
Your Score: 5
Loved it!
- Gladiator (2000)**
Your Score: 5
Terrific movie!
- The Matrix (1999)**
Your Score: 5
Best Sci-Fi I've ever seen
- Hannibal (2001)**
Your Score: 4
Great
- The Matrix Reloaded (2003)**
Your Score: 4
It was enjoyable for a while
- Logan (2017)**
Your Score: 3
Meh, was ok!
- Hannibal (2001)**
Your Score: 1
Meh!
- It (2017)**
Your Score: 1
Awful

What's Hot Search **My List**

This uses the Facebook sharer.php API and it contains a link to the site. When posted, it presents some text and the link on your facebook page.

 **Share on Facebook**

Share on your own Timeline ▼

 **Eoin Sutton**
Say something about this...

Heres a list of my favourite movies on Movie-io! ✕

movie-io.byethost7.com
MOVIE-IO

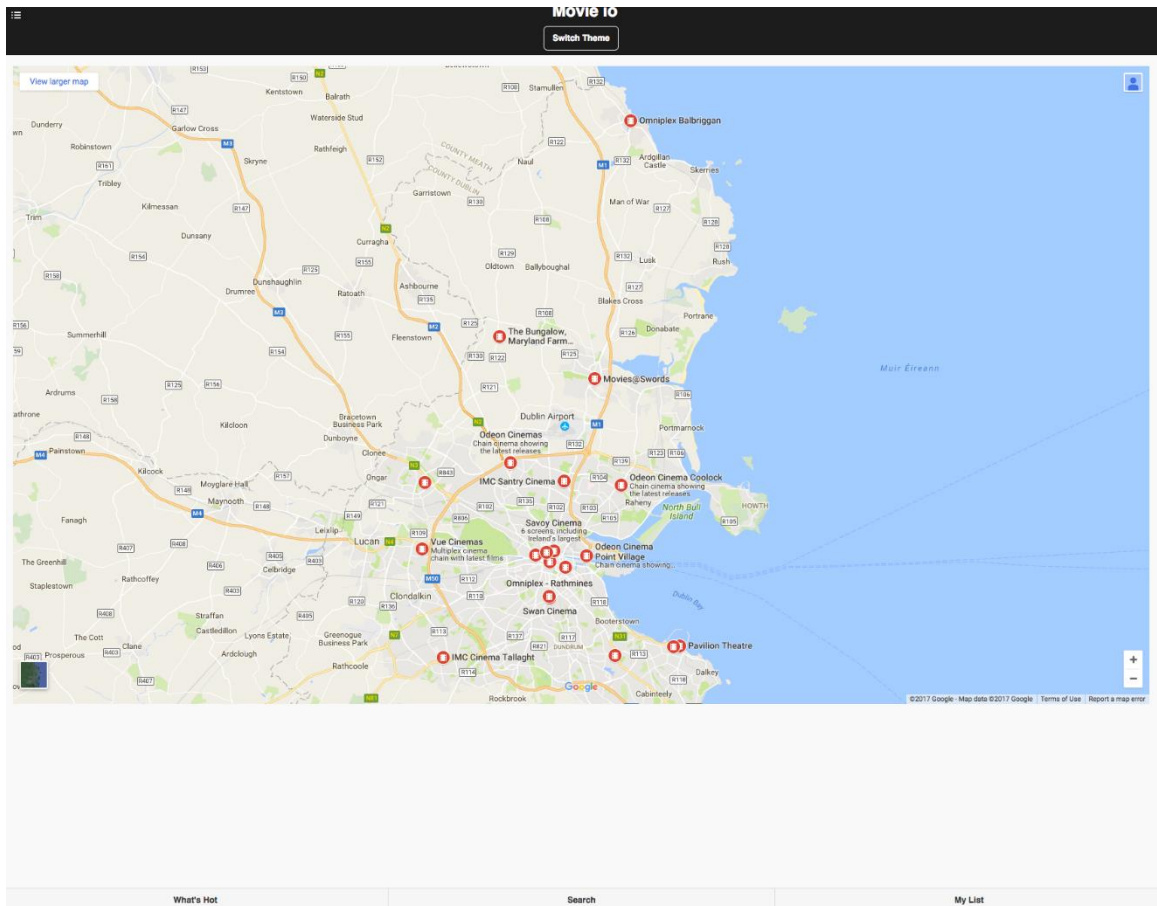
The icon contains the link itself as a href. The link is for the online version of the app on ByetHost. On page load of the #main-page (What's Hot) I gather the user_id and concatenate that to the link and then attach that as the href attribute of the icon.

```
//check user logged in on load of main-page data role, also grab gps position
$( document ).on( "pagecreate", "#main-page", function() {
    checkLogin();
    loadGeoData();
    toggleStyles();
    //set facebook Link
    var user_id = $('#sessionuserid').text();
    var link = "https://www.facebook.com/sharer/sharer.php?u=https://movie-io.byethost7.com/public-list-page.php?id="+ user_id
    $("#facebookLink").attr("href", link);
});
```

The link brings the user to the public-list-page.php. This is a separate page of the app that does not require login and brings the user to the My List page. On document load, it sends the POST request mentioned earlier to load_mylist.php, with the user id that was send via the URL and gets the same movie data, sorted by rating. The user can see their friends' movie list and when they click on any of the links, it directs them to the login page to login or register to gain access to the site.

3.1.7 Cinemas/Maps page

When a user opens the menu drawer on the left of the app, they are shown a link to the Cinemas page. This takes them to a page in the app where a Google map will load and show the nearest cinemas around them which they can click on for directions and website links.



This was implemented by using the Google Maps Embed API. This allowed me to load a Google map via an iframe and use a standard query in the URL to gather the data I needed. I first registered with Google via developers.google.com/maps and generated an API key to be allowed to use the API. Once I had this, it was simply a case of adding the API into the URL for the iframe, with the search query, to call the data I needed.

However it was not quite so easy. The query I used was 'cinemas near Dublin' which was fine, as I was in Dublin. However this wouldn't work for anywhere else in the world. I also had to input the latitude and longitude co-ordinates into the query to center the map. While this was doable, the query did not allow co-ordinates as parameters, only place names. So I needed to develop a way to input the current place name of the user into the query so Google would return cinemas near their location.

I did this via a process known as ‘reverse geocoding’. On map page load, I gather the latitude and longitude via sessionStorage. I then instantiate a Geocoder object and input the lat/long values into the geocode method. This returns the full address of this particular point on the map, up to several levels (house name, street, suburb, city etc.). After some testing and trial and error, I eventually found the 6th level contains the city area (or equivalent) around the user. I save this to a variable and include it in the query with the lat/long co-ordinates. This ensures that no matter where the user is using the app, the maps page will always show cinemas in their vicinity.

```
var latlng = {lat: parseFloat(latlngStr[0]), lng: parseFloat(latlngStr[1])};
//use geocoder to find current place name and use that to grab results of map search query
geocoder.geocode({'location': latlng}, function(results, status) {
  if (status == "OK"){
    if (results[0]){
      var currentLoc = results[0].address_components[5].long_name;
      //alert("location name = " + results[0].address_components[5].long_name);
      //load map
      $('#map-canvas').html('<iframe width="100%" height="70%" frameborder="0" style="border:0" +
      'src="https://www.google.com/maps/embed/v1/search?key=AIzaSyCoSolGmCfXef_f3hK7qtxWdUFGwCfF0qE&center='+ latlng
      +'&zoom=11&q=cinemas+near+' + currentLoc +' allowfullscreen></iframe>');
      $('#map-canvas').css('background-color', 'red');
      $('#map-canvas').attr('style', 'background-color: red !important');
```

3.1.8 Theme Switcher

To give users a choice of look and feel for the app, I decided to include two CSS themes. A light theme and a dark theme. The dark theme uses a mix of black backgrounds and red content and gives an alternative look to the site, while also helping users avoid eye strain at night. The light theme is the default, but users can switch at any time, without reloading by clicking the ‘Switch Theme’ button on the header of each page.

The Matrix Reloaded (2003)



Release Date: 21 May 2003 (Ireland)

Plot: Six months after the events depicted in The Matrix, Neo has proved to be a good omen for the free humans, as more and more humans are being freed from the matrix and brought to Zion, the one and only stronghold of the Resistance. Neo himself has discovered his superpowers including super speed, ability to see the codes of the things inside the matrix, and a certain degree of precognition. But a nasty piece of news hits the human resistance: 250,000 machine sentinels are digging to Zion and would reach them in 72 hours. As Zion prepares for the ultimate war, Neo, Morpheus and Trinity are advised by the Oracle to find the Keymaker who would help them reach the Source. Meanwhile Neo's recurrent dreams depicting Trinity's death have got him worried and as if it was not enough, Agent Smith has somehow escaped deletion, has become more powerful than before and has chosen Neo as his next target. Written by Gountra

Starring: Keanu Reeves , Laurence Fishburne , Carrie-Anne Moss

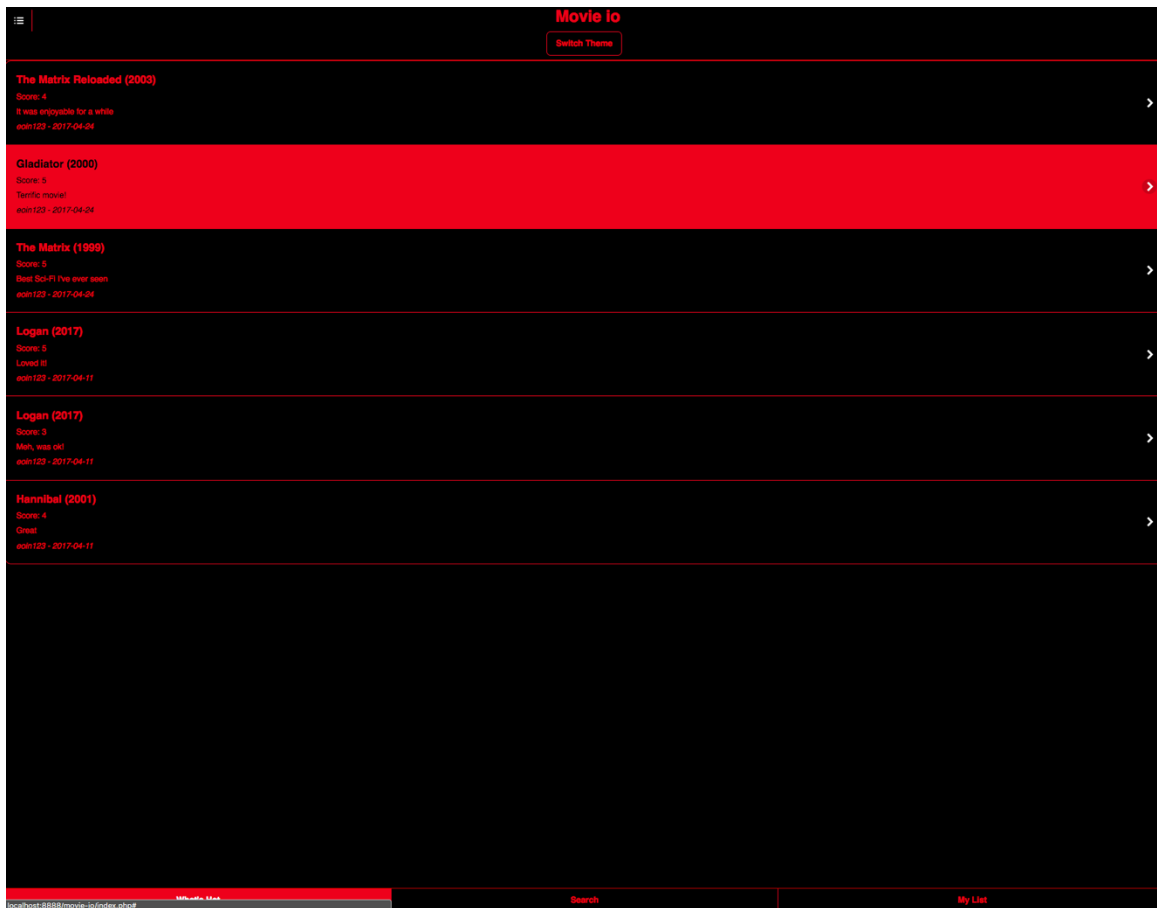
Awards: 7 wins & 31 nominations.

Metascore: 62

Score:

★★★★★

What did you think?



This was implemented by importing both stylesheets on app load, designating one as 'stylesheet alternate':

```
<link rel="stylesheet" href="jquery/themes/Movie0_Light.css" id="light" />  
<link rel="stylesheet alternate" href="jquery/themes/Movie0_Dark.css" id="dark" />
```

The button is given a class called 'toggleBtn' and on click via the function toggleStyles(), the href of the main stylesheet is changed dynamically at each click, switching between the two.

```
//toggle stylesheets
function toggleStyles(){
$('.toggleBtn').on('click', function() {
    var href = $('#light').attr('href');
    if(href == 'jquery/themes/Movie0_Light.css'){
        $('#light').attr('href','jquery/themes/Movie0_Dark.css');
    }
    else{
        $('#light').attr('href','jquery/themes/Movie0_Light.css');
    }
});
}
```

Originally I had a toggle slider using the jQuery Mobile 'flipswitch' data-role, but it proved inconsistent on each page where the page would be dark but the slider text would say light. It turned out to be a nightmare trying to ensure the text displayed the correct one for each theme. I spent many hours on it and finally decided to use the current setup. The button, is I think, the most elegant way to switch themes.

3.2 Graphical User Interface (GUI) Layout

Below are the mobile GUI layouts of the various pages of the app. Desktop scale layouts are available in the implementation section.

Welcome to Movie IO!

Login

Username

Password

Login

[Register](#)

Welcome to Movie IO!

Sign Up

Username

Email Address

Password

Submit

[Login](#)



Switch Theme

Guardians of the Galaxy Vol. 2 (2017)

Score: 5

Loved it!

eoin123 - 2017-04-29



Forrest Gump (1994)

Score: 3

I liked it

eoin123 - 2017-04-27



Aliens (1986)

Score: 5

Terrific!

eoin123 - 2017-04-27



The Matrix Reloaded (2003)

Score: 4

It was enjoyable for a while

eoin123 - 2017-04-24



Gladiator (2000)

Score: 5

Terrific movie!

eoin123 - 2017-04-24



What's Hot


Search

My List

Movie io

Switch Theme

Q Fellowship of the ring



19 December 2001 (Ireland)

The Lord of the Rings: The Fellowship of the...
An ancient Ring thought lost for centuries has been found, and th...

What's Hot Search My List



Switch Theme

The Lord of the Rings: The Fellowship of the Ring (2001)




Release Date: 19 December 2001 (Ireland)

Plot: An ancient Ring thought lost for centuries has been found, and through a strange twist in fate has been given to a small Hobbit named Frodo. When Gandalf discovers the Ring is in fact the One Ring of the Dark Lord Sauron, Frodo must make an epic quest to the Cracks of Doom in order to destroy it! However he does not go alone. He is joined by Gandalf, Legolas the elf, Gimli the Dwarf, Aragorn, Boromir and his three Hobbit friends Merry, Pippin and Samwise. Through mountains, snow, darkness, forests, rivers and plains, facing evil and danger at every corner the Fellowship of the Ring must go. Their quest to destroy the One Ring is the only hope for the end of the Dark Lords reign! Written by Paul Twomey







What's Hot

Search

My List

☰ **Movie io** 

Switch Theme

- Logan (2017)**
Your Score: 5
Loved it! 
- Aliens (1986)**
Your Score: 5
Terrific! 
- Guardians of the Galaxy Vol. 2 (2017)**
Your Score: 5
Loved it! 
- Gladiator (2000)**
Your Score: 5
Terrific movie! 
- The Matrix (1999)**
Your Score: 5
Best Sci-Fi I've ever seen 
- Hannibal (2001)**
Your Score: 4
Great 

What's Hot Search **My List**



Movie io

Switch Theme

Username: eoin123



Email: eoin.suttton@live.com

About Me: First reviewer

My Location: Dublin!!!

Update name...

Update email...

Update about me...

Update location...

Upload your image:

Choose File No file chosen

Submit

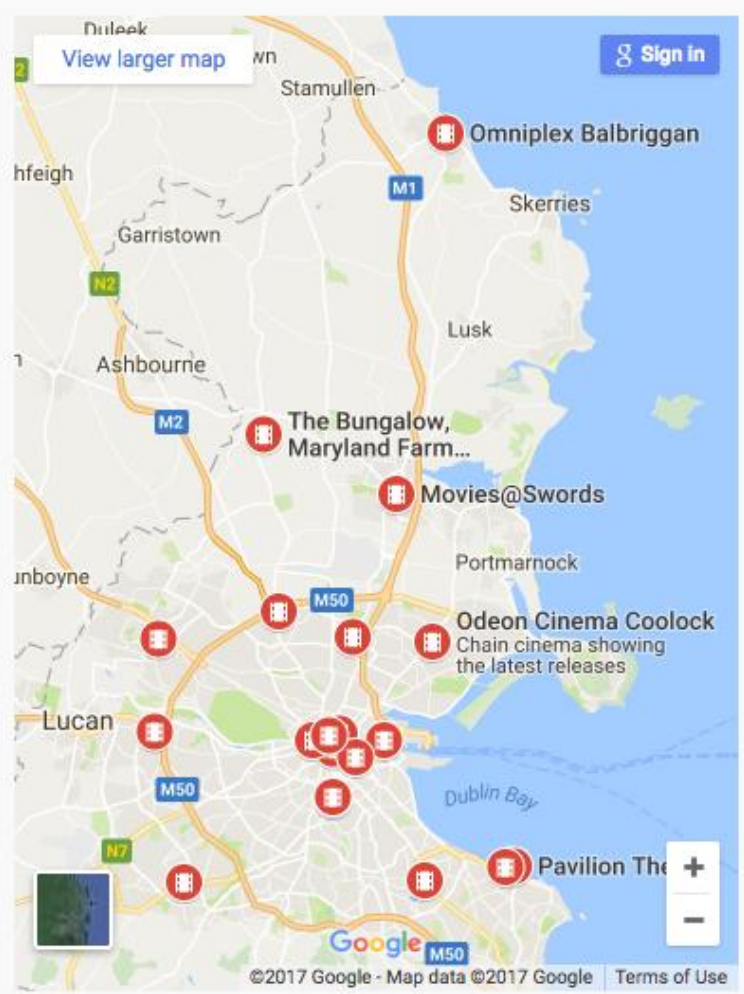
Comments:



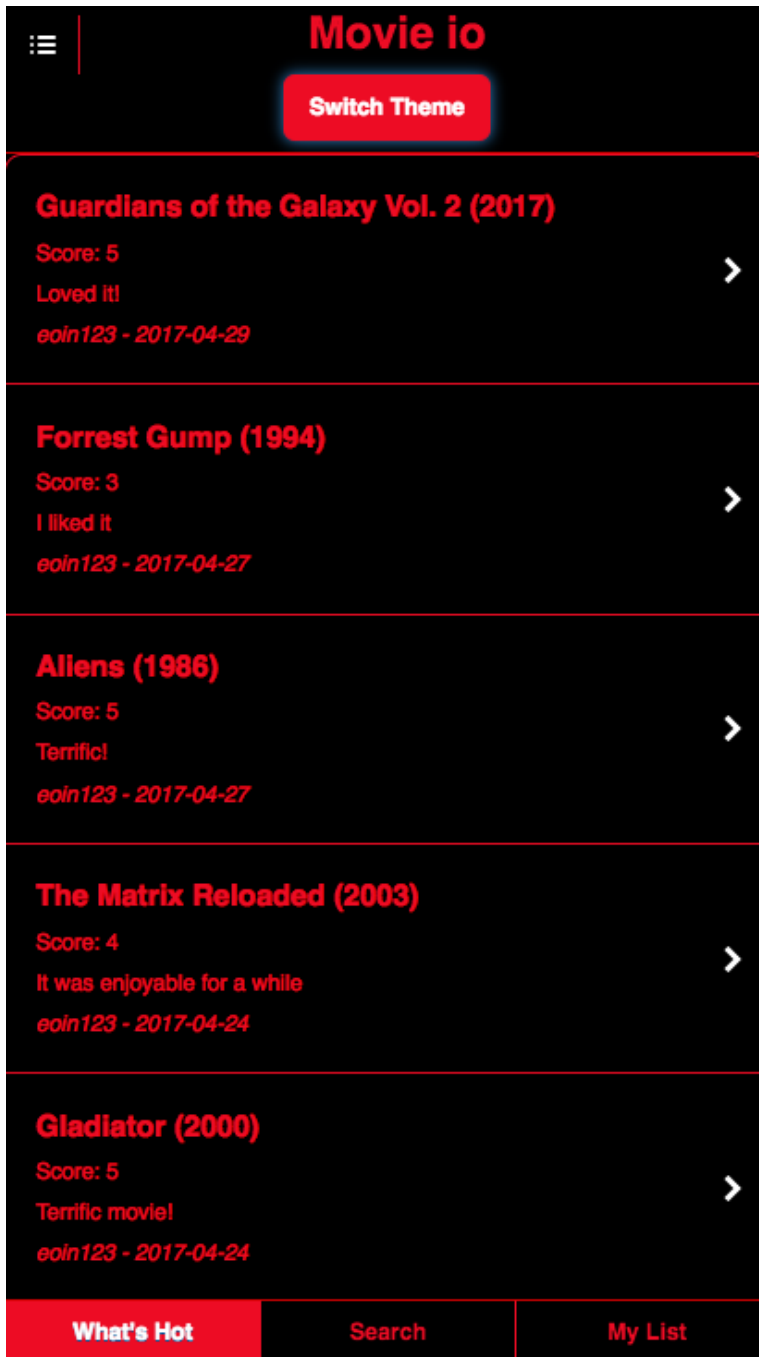
What's Hot

Search

My List



Dark theme example GUI:



3.3 Testing

Testing for the project consisted of a mix between unit testing and system testing. Unit testing was conducted during development and included using a dedicated javascript testing suite called QUnit. I also extensively used the console by outputting data via `console.log()` and by using alerts to ensure I was receiving the correct data back from the server. I will discuss a number of examples here.

One area I was keen to ensure correct data flow between functions was when using the Haversine formula for the 'What's Hot' page. It was absolutely imperative that the latitude and longitude values were the correct format and the function could read these and output the correct great circle distance between two points.

To this end I wanted to conduct unit testing to ensure everything worked as it should. I took two GPS location co-ordinates ('53.45114686611449', '-6.233047340469966') and ('53.451090', '-6.233027') and went to the website where I referenced the Haversine formula (<http://www.movable-type.co.uk/scripts/latlong.html>) which has a great circle distance calculator available. The distance between these two points was measured here at 0.006465 km so I knew this was the figure I had to match.

To do this I created a dedicated test bench using QUnit. QUnit is a free javascript unit testing framework. I created Qunit-test-home.html into which I included the QUnit javascript and css files and also the Haversine functions.

```

<script>
var userLoc = new LatLon('53.45114686611449', '-6.233047340469966');
var reviewLoc = new LatLon(Number('53.451090'), Number('-6.233027'));

function LatLon(lat, lon) {
// allow instantiation without 'new'
if (!(this instanceof LatLon)) return new LatLon(lat, lon);
this.lat = Number(lat);
this.lon = Number(lon);
}

//calculate radians
function toRad(x) {
return x * Math.PI / 180;
}

LatLon.prototype.distanceTo = function(point, radius) {
if (!(point instanceof LatLon)) throw new TypeError('point is not LatLon object');
radius = (radius === undefined) ? 6371e3 : Number(radius);
console.log("Point: " + point + " radius: " + radius);

var R = radius;
var v1 = toRad(this.lat), a1 = toRad(this.lon);
var v2 = toRad(point.lat), a2 = toRad(point.lon);
var v_total = v2 - v1;
var a_total = a2 - a1;

var a = Math.sin(v_total/2) * Math.sin(v_total/2)
+ Math.cos(v1) * Math.cos(v2)
* Math.sin(a_total/2) * Math.sin(a_total/2);
var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
var d = R * c;

return d;
};

QUnit.test('gps test', function(assert) {
assert.ok(userLoc.distanceTo(reviewLoc), "distance calc returns something");
assert.equal( (userLoc.distanceTo(reviewLoc) /1000).toFixed(6), 0.006465, 'great circle distance between these two points matches!');
});
</script>
</head>
<body>
<h1 id="qunit-header">QUnit Test Suite</h1>
<h2 id="qunit-banner"></h2>
<div id="qunit-testrunner-toolbar"></div>
<h2 id="qunit-userAgent"></h2>
<ol id="qunit-tests"></ol>

```

For the test I included two methods – `assert ok()` and `assert equal()`. `Ok()` is a Boolean check and it returns true if the distance function returns a value (so I know it is working) and `equal()` returns true where a comparison between the output of the Haversine function equals the known distance I got from the website.

Opening the Qunit-test-home.html page runs the tests and the output is below.

```
QUnit Test Suite
 Hide passed tests  Check for Globals  No try-catch Filter:  Go
QUnit 2.3.2; Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/603.1.30 (KHTML, like Gecko) Version/10.1 Safari/603.1.30
1 tests completed in 2 milliseconds, with 1 failed, 0 skipped, and 0 todo.
1 assertions of 2 passed, 1 failed.
1. gps test (1, 1, 2) ✖ Rerun
  1. distance calc returns something
  2. great circle distance between these two points matches!
  Expected: 0.006465
  Result: 0.006465081702847382
  Diff: +8.170284738172329e-8
  Source: runTest@https://code.jquery.com/qunit/qunit-2.3.2.js:1442:34
Source: global code@file:///Applications/MAMP/htdocs/Movie-IO/Qunit-Test-Home.html:52:15
```

Initially the test failed but as you can see this was only due to a rounding issue. When I forced the output of the function to 6 digits to match the output from the website I received the below.

```
QUnit Test Suite
 Hide passed tests  Check for Globals  No try-catch Filter:  Go
QUnit 2.3.2; Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_6) AppleWebKit/603.1.30 (KHTML, like Gecko) Version/10.1 Safari/603.1.30
1 tests completed in 3 milliseconds, with 0 failed, 0 skipped, and 0 todo.
2 assertions of 2 passed, 0 failed.
1. gps test (2) ✔ Rerun
  1. distance calc returns something
  2. great circle distance between these two points matches!
  Expected: 0.006465
  Result: 0.006465081702847382
  Diff: +8.170284738172329e-8
  Source: runTest@https://code.jquery.com/qunit/qunit-2.3.2.js:1442:34
Source: global code@file:///Applications/MAMP/htdocs/Movie-IO/Qunit-Test-Home.html:52:15
```

Both tests passed, so I knew I could include the Haversine function in the main javascript file.

I had some difficulty getting the QUnit framework to work with ajax requests, so for functions using ajax I tested using console.log() and alerts. For functions like loadGeoData(), as they returned JSON Objects I used console.log() to output the values of the responses from the server/PHP files, in order to check they were returning the correct entries; for example:

```

▼ [Object, Object, Object, Object, Object, Object] ⓘ
  ▼ 0: Object
    date: "2017-04-24"
    lat: "53.451127"
    lng: "-6.233035"
    movie_id: "39"
    movie_title: "The Matrix Reloaded (2003)"
    review: "It was enjoyable for a while"
    score: "4"
    username: "eoin123"
    ► __proto__: Object
  ▼ 1: Object
    date: "2017-04-24"
    lat: "53.451123"
    lng: "-6.233073"
    movie_id: "36"
    movie_title: "Gladiator (2000)"
    review: "Terrific movie!"
    score: "5"
    username: "eoin123"
    ► __proto__: Object
  ► 2: Object

```

This is the return of two of the objects from `loadGeoData()`. As I knew both were within 100km of my current location, I knew the data was correct and the format was what I needed. I use this to populate the list items on the 'What's Hot' page.

Another function I tested was the function which loads at `$(document).ready()` (on load) of the `public-list-page.php`. This took an id parameter from the URL posted to Facebook, so I needed to ensure it returned the correct data for that id. During testing my user id was 30 so from a query on the database, I knew that I had reviewed 8 movies and this was what I was expecting on the page:

movie_title	release_date	review	score	id
Logan (2017)	1 March 2017 (UK)	Loved it!	5	34
Gladiator (2000)	12 May 2000 (Ireland)	Terrific movie!	5	36
The Matrix (1999)	11 June 1999 (Ireland)	Best Sci-Fi I've ever seen	5	38
Hannibal (2001)	16 February 2001 (UK)	Great	4	35
The Matrix Reloaded (2003)	21 May 2003 (Ireland)	It was enjoyable for a while	4	39
Logan (2017)	1 March 2017 (UK)	Meh, was ok!	3	34
Hannibal (2001)	16 February 2001 (UK)	Meh!!	1	35
It (2017)	8 September 2017 (USA)	Awful	1	37

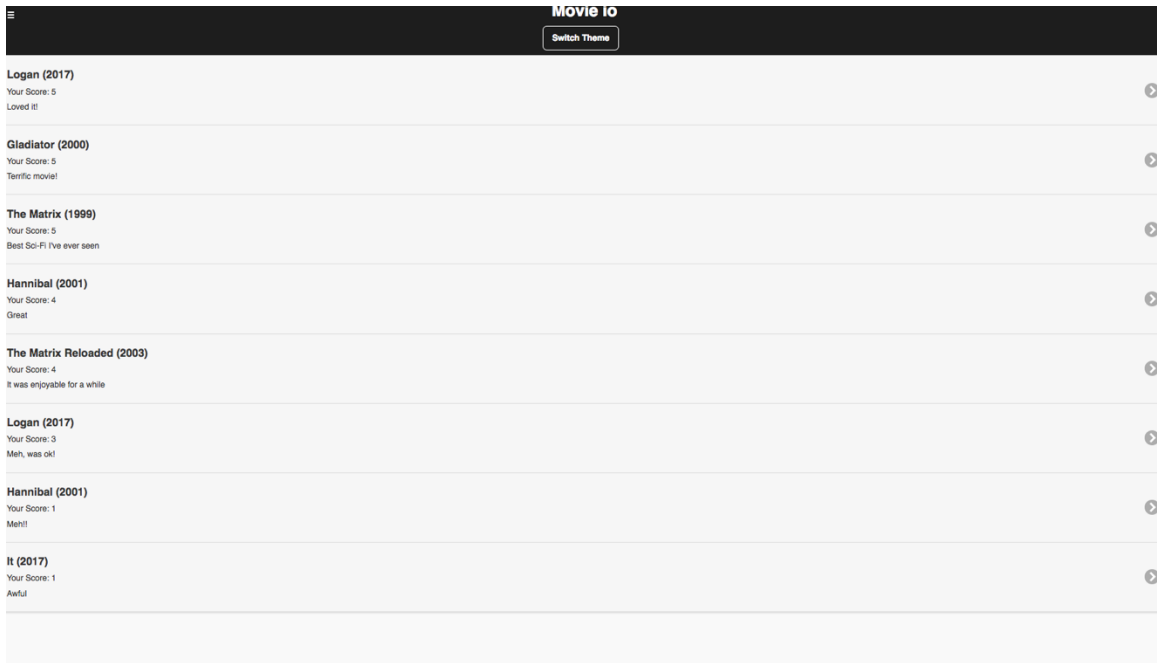
I purposefully set the id parameter for the function to 30 and reloaded the page.

```
$(document).ready(function(){
  $.ajax({
    type: "POST",
    url: "load_myList.php",
    data: "id="+30,
    success: function(result){
      result = $.parseJSON(result);
      $.each(result, function (key, value) {
        //$('#listviewpage').append(list);
        $('#listviewpage').append("<li onClick='loadMovieDataFromList(\x22 + value.id + "\x22)'><a href='https://movie-io.byethost7.com/index.php'>"
          + "<h2>" + value.movie_title + "</h2>"
          + "<p> Your Score: " + value.score + "</p>" + "<p>" + value.review + "</p>" + "</a></li>").listview('refresh');
      });
    },
    error: function(xhr, status, error){
      //$('#message')[0].value = "Ajax error!" + result;
      alert("User review fetch error " + xhr.responseText);
    }
  });
});
```

It populated the 8 movies – but they were not in the right order. I needed to include an ORDER BY clause to the SQL query to sort that. I added this to the query in load_mylist.php:

```
//query
$query = "
SELECT a.movie_title, a.release_date, a.review, a.score, b.id
FROM movie_reviews a join movies b on (a.movie_title = b.title)
WHERE user_id = :id
ORDER BY a.score desc
";
```

This fixed the problem:



Continuous testing also took place of the PHP code. I added the below lines to some of the PHP files in order to output messages to the PHP_error log file.

```
//debug
ini_set('display_errors', 1);
error_reporting(E_ALL);
ini_set('error_log', 'path_to_log_file');
ini_set('log_errors_max_len', 0);
ini_set('log_errors', true);
```

I also experimented with using the below with some PHP files (load_movies_from_list.php for example) but found I could also just use the console.log() in the javascript to achieve the same output.

```
//debug
function debug_to_console( $data ) {
    $output = $data;
    if ( is_array( $output ) )
        $output = implode( ',', $output);

    echo "<script>console.log( 'Debug Objects: " . $output . " ');</script>";
};
```

The PHP error log was invaluable. When testing review_movie.php (which saves users reviews), I was dumbfounded as to why I was getting an error when running the function. Everything seemed fine to my eye in the code. When I output errors to the log, I was instantly able to see the issue:

```
movie-io/scraperv2.php on line 28
[28-Mar-2017 21:52:16 Europe/Berlin] PHP Parse error: syntax error, unexpected '=>' (T_DOUBLE_ARROW),
expecting ')' in /Applications/MAMP/htdocs/movie-io/review_movie.php on line 21
```

It was an error in my syntax – in this case I had added the latitude and longitude to the review query parameters but had omitted a comma between the lines:

```
// parameters
$query_params = array(
    ':user_id' => $_POST['user_id'],
    ':score' => $_POST['score'],
    ':review' => $_POST['review'],
    ':release_date' => $_POST['release_date'],
    ':movie_title' => $_POST['title'],
    ':movie_id' => $_POST['movie_id'],
    ':lat' => filter_var($_POST['lat'], FILTER_VALIDATE_FLOAT),
    ':lng' => filter_var($_POST['lng'], FILTER_VALIDATE_FLOAT)
);
```

Once I added it, the function worked perfectly. I continued this process for other PHP files where I was having an issue, and I was consistently using alerts and console.log() in the javascript functions to output the responses from PHP and manually checking against the correct data.

System testing consisted of testing the entire application to ensure it was meeting its requirements. System testing was conducted at the end of April 2017 and each of the requirements were individually tested and the results are below:

Requirement	Test	Test Result
User Registration	Create new account	Passed
User Login	Login with new account	Passed
Movie Search	Search for movie 'Gladiator'	Passed
Rate Movie	Review and rate 'Gladiator'	Passed
Share Movie Lists	Share My List via Facebook, click link ensure correct data	Passed
Review Users	Review and rate test user 'eoin123'	Passed
View Cinemas	View Cinemas page, ensure local cinemas listed	Passed

Additional features were also tested including theme switcher, profile and the 'What's Hot' page location specific reviews. All passed.

Non-functional requirements were also tested:

Requirement	Test Result
Performance/Response time < 2 seconds	Passed
Security (user cannot access site if not logged in)	Passed. Online hosted version of app also uses HTTPS.
Reliability (no maintenance required)	Passed

Usability (no manual or training required by users)	Passed
---	--------

At the time of writing I am satisfied that system testing has passed and all the requirements set out have been met, along with some additional functionality added.

3.4 Customer testing

Customer, or Usability testing was conducted using the same participants that I used to help gather the user requirements for the app. I felt they were best placed to test the app as they had an understanding of the project and had shaped much of it through the requirements elicitation process.

I focused on three areas for this testing phase:

1. Effectiveness
2. Efficiency
3. Satisfaction

For effectiveness, I conducted a Think Aloud test. This is where a participant is asked to interact with a system by performing certain tasks, all the while vocalising their thoughts and opinions.

As I have five participants, I asked them all to register with the site, login, search for a movie, review that movie and then share it on Facebook. I asked them to verbalise their thoughts while doing so and recorded both their thoughts in writing, and their behaviour in a table below. To measure efficiency, I timed each participant.

A1: Beke Hollenbach: "Ok I can see the login page and a register link. I clicked on register, put my details in and have now been taken back to login. I am now able to login. The site looks good, very clean. Ok I see a search button on the bottom so I've clicked on that and now I can enter the movie. Ok let's see, how about 'Arrival' which I saw recently. When I press enter it takes a second or so for the movie to show up, but its there now. When I click on that I'm taken to the page for

the movie – it seems to have all the information on it. I like it. It's laid out well and I understand what to do. I added a review and I can see it on the bottom there. When I click on My List I see the review is there and I can share on Facebook. It asks me to input a captcha code before posting to facebook. Ok done. I like it, very easy to use.”

Observations:

Task	Start Time (sec)	End Time (sec)	Expected Behaviour	Actual Behaviour	Notes
Register	0	0:10	Click register, add details	Same	
Login	0:10	0:30	Login with same details	Same	
Search Movie	0:30	0:40	Click Search, enter movie, name, click on movie name	User hesitated to look around site first, before clicking Search	
Review Movie	0:40	1.20	Click on a star rating, click on review textarea, review movie and submit	Same	

Share on Facebook	1.20	1.35	Click on Facebook share, enter Facebook details and submit	Same	
-------------------	------	------	--	------	--

A2: Sean Sutton: “After I register I can login fine. The design of the site looks good. What’s Hot looks interesting. I like the theme switcher. I can search for movies fine – for this test I’ve searched for ‘Aliens’ which comes up after a second. When I click on it it brings me to the Aliens page. I can click on a start out of 5 to rate it and say what I think. Done – and I can see the review below. Nice touch. To share it, I take it I go to My List. Yep, there’s the facebook icon. Sharing is fine, I can’t see any issues.”

Observations:

Task	Start Time (sec)	End Time (sec)	Expected Behaviour	Actual Behaviour	Notes
Register	0	0:05	Click register, add details	Same	
Login	0:05	0:08	Login with same details	Same	
Search Movie	0:08	0:22	Click Search, enter movie, name,	User also stopped to get familiar with site look before	

			click on movie name	clicking on search	
Review Movie	0:22	0:32	Click on a star rating, click on review textarea, review movie and submit	Same	
Share on Facebook	0:32	0:45	Click on Facebook share, enter Facebook details and submit	User stopped to check if there was a facebook icon on review page	Note facebook icon placement (on right page?)

A3: Tim O'Leary: "Login and registration are fine. I suppose the app title should be on the page header for the registration? Other than that looks good. I can see the movie list for What's Hot and a search page. My List I guess is my own list. Searching is fine, it's quick enough. I searched for 'Gladiator' there and it came up. Clicking on that then seems to bring me to the movie page – nice and quick. I'll enter a review here. Done. Alright, I saw the Facebook icon on the other page so I'll try that. Asks me to enter a captcha code but apart from that it's looking good. It would be nice to have the movie images on the list items themselves."

Observations:

Task	Start Time (sec)	End Time (sec)	Expected Behaviour	Actual Behaviour	Notes
Register	0	0:06	Click register, add details	Same	Note – no title on page!
Login	0:06	0:10	Login with same details	Same	
Search Movie	0:10	0:27	Click Search, enter movie, name, click on movie name	Again user looked around first	
Review Movie	0:27	0:38	Click on a star rating, click on review textarea, review movie and submit	Same	
Share on Facebook	0:38	0:50	Click on Facebook share, enter Facebook details and submit	Same	

A4: David Sullivan: “Ok I can register easy enough and login was ok. Site looks minimalist and it seems snappy. Search is easy to find, and searching for ‘Forrest Gump’ gets a result within a couple of seconds. The layout of the movie page is simple but effective. I see the adding a review adds it to the bottom of the page aswell. If I click on My List I can see the facebook share link. Ok this works fine, pity about the captcha but not a big deal.”

Observations:

Task	Start Time (sec)	End Time (sec)	Expected Behaviour	Actual Behaviour	Notes
Register	0	0:07	Click register, add details	Same	Note – no title on page!
Login	0:07	0:10	Login with same details	Same	
Search Movie	0:10	0:15	Click Search, enter movie, name, click on movie name	Same	
Review Movie	0:15	0:30	Click on a star rating, click on review textarea,	User clicked What’s Hot link again and was	User error

			review movie and submit	taken away from page, had to search again	
Share on Facebook	0:30	0:42	Click on Facebook share, enter Facebook details and submit	Same	

A5: John O'Shea: "It seems intuitive, I can register and login. I like the layout, not too much not too little. Search is intuitive. I searched for 'The Dark Knight' and it found it in less than 2 seconds. The movie page looks good. I like the fact it adds the review to the page, and my picture is on it. The My List page lists the review I see and the Facebook icon is there. I'm able to share it ok and the link takes me to the list again. Nice."

Observations:

Task	Start Time (sec)	End Time (sec)	Expected Behaviour	Actual Behaviour	Notes
Register	0	0:05	Click register, add details	Same	Note – no title on page!
Login	0:05	0:09	Login with same details	Same	

Search Movie	0:09	0:15	Click Search, enter movie, name, click on movie name	Same	
Review Movie	0:15	0:30	Click on a star rating, click on review textarea, review movie and submit	Same	
Share on Facebook	0:30	0:42	Click on Facebook share, enter Facebook details and submit	Clicked 'What's Hot' page first, then My List	

Overall the think aloud test seemed to go well. It captured the fact that the site seems intuitive for users to know where to go and what to do. It helped me to identify that there was a title missing on the registration page (now corrected) and that the time to complete the steps was broadly in line with what I had tested myself. As a baseline I found that 40-45 seconds was about right to complete all the steps in a leisurely fashion. User comments like the movie images being available in the list items will also be taken into account for future development.

To gather data on user satisfaction, I asked participants to fill out a System Usability Scale survey and calculated the scores for each. Instructions given were “Based on your experience *today*, check the box that reflects your immediate response to each statement. Don’t think too long about each statement. Make sure you respond to every statement. If you don’t know how to respond, simply check box ‘3’.” (Software Usability Scale (Positive), 2017)

	<i>Beke Hollenbach</i>	Strongly disagree					Strongly agree
		1	2	3	4	5	
1	I think that I would like to use the website frequently.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
2	I found this website unnecessarily complex	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3	I thought the website was easy to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
4	I think that I would need assistance to be able to use this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5	I found the various functions in the website were well integrated.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
6	I thought there was too much inconsistency in this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
7	I would imagine that most people would learn to use the website very quickly.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
8	I found the website very cumbersome/awkward to use.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
9	I felt very confident using the website.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
10	I needed to learn a lot of things before I could get going with this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

To score a System Usability Scale:

- “For each of the odd numbered questions, subtract 1 from the score.
- For each of the even numbered questions, subtract their value from 5.

- Take these new values which you have found, and add up the total score. Then multiply this by 2.5.” (Usabilitygeek.com, 2015)

Beke Score: **97.5**

		Strongly disagree 1	2	3	4	Strongly agree 5
	Sean Sutton					
1	I think that I would like to use the website frequently.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	I found this website unnecessarily complex	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	I thought the website was easy to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	I think that I would need assistance to be able to use this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	I found the various functions in the website were well integrated.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	I thought there was too much inconsistency in this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	I would imagine that most people would learn to use the website very quickly.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
8	I found the website very cumbersome/awkward to use.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	I felt very confident using the website.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
10	I needed to learn a lot of things before I could get going with this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Sean Score: **95**

		Strongly disagree 1	2	3	4	Strongly agree 5
	Tim O’Leary					
1	I think that I would like to use the website frequently.	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	I found this website unnecessarily complex	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	I thought the website was easy to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

4	I think that I would need assistance to be able to use this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	I found the various functions in the website were well integrated.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	I thought there was too much inconsistency in this website.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	I would imagine that most people would learn to use the website very quickly.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
8	I found the website very cumbersome/awkward to use.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	I felt very confident using the website.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
10	I needed to learn a lot of things before I could get going with this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Tim Score: **90**

		Strongly disagree				Strongly agree
	David Sullivan	1	2	3	4	5
1	I think that I would like to use the website frequently.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	I found this website unnecessarily complex	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	I thought the website was easy to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	I think that I would need assistance to be able to use this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	I found the various functions in the website were well integrated.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
6	I thought there was too much inconsistency in this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	I would imagine that most people would learn to use the website very quickly.	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	I found the website very cumbersome/awkward to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	I felt very confident using the website.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
10	I needed to learn a lot of things before I could get going with this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

David Score: **87.5**

	John O'Shea	Strongly disagree 1	2	3	4	Strongly agree 5
1	I think that I would like to use the website frequently.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
2	I found this website unnecessarily complex	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	I thought the website was easy to use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4	I think that I would need assistance to be able to use this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	I found the various functions in the website were well integrated.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
6	I thought there was too much inconsistency in this website.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	I would imagine that most people would learn to use the website very quickly.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
8	I found the website very cumbersome/awkward to use.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	I felt very confident using the website.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
10	I needed to learn a lot of things before I could get going with this website.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

John Score: **87.5**

Overall I feel the customer testing went well. It helped me identify how users interacted with the app and that it is not unintuitive or difficult to get tasks done. It also provided insight into what future features users would like to see. Usability scores also fall into the higher category which tells me that users are happy with the layout and workings of the app.

3.5 Heuristic Evaluation

Heuristic evaluation looks at evaluating a website to look for violations of common usability principles (for example, established guidelines). For the purposes of this

report I have selected the following heuristics that most apply to the project and summarised the thoughts of the testers.

Visibility of System Status

Where am I and where can I go next?

The consensus here was that it was generally easy to navigate the site. The leftmost menu drawer is always available and the bottom links are always visible on each page making it easy to navigate. Both links and page content give users a clear indication of where they are.

Match between the System and the Real World

The language and naming used throughout, at the user level, is very casual and uses understandable everyday language. Words are designed to be clear, concise and without ambiguity. Testers expressed no issues understanding entity names on the site.

User Control and Freedom

As per the above, a user always has access to links to each main area of the site. They are free to move around from any page. The freedom is also there to switch CSS themes between light and dark tones. No issues were brought up by testers here.

Aesthetic and Minimalist Design

The testers enjoyed the design and layout of the site. It is designed to be centred around the information and to display only information that is relevant. A few of the testers made positive comments on the site layout. No issues reported.

4 Conclusions

This project was a pleasure and a challenge to implement. Challenging as the javascript files contain nearly 700 lines of code combined and the server side PHP files have circa. 1200 lines of code in total. It was complex at times ensuring the correct information was being sent to and retrieved from the database, but also rewarding when everything worked correctly. I think the layout of the site works well for both desktop and mobile use and user feedback from testing indicates that users are happy with the finished product.

Overall I think this was a very interesting and rewarding project. It presented challenges, opportunities for further learning and allowed me to showcase a full mobile ready application using modern technologies. It has allowed me to put into practice elements of programming, testing, requirements elicitation and others I have learned over the course of my degree; elements that I can bring to the workplace to improve my skills and, ultimately, my employability in the industry.

5 References

Software Usability Scale (Positive). (2017). 1st ed. [ebook] p.1. Available at: <http://uspto.github.io/designpatterns/docs/guides/downloads/PostTestSurvey-SUS.docx> [Accessed 28 Apr. 2017].

Usabilitygeek.com. (2015). *How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website*. [online] Available at: <http://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/> [Accessed 28 Apr. 2017].

Code References:

Secure PHP Login (2012). Available at: <http://forums.devshed.com/php-faqs-stickies-167/program-basic-secure-login-system-using-php-mysql-891201.html> [Accessed 01 Jan. 2017].

connection.php – changed DB connection details to reflect my DB credentials

```
* @reference http://forums.devshed.com/php-faqs-stickies-167/program-basic-secure-login-system-using-php-mysql-891201.html *
*/
//variables
$username = "root";
$password = "root";
$host = "localhost";
$dbname = "Movieio";

try{
    $db = new PDO("mysql:host={$host}; port=8889; dbname={$dbname};charset=UTF8", $username, $password);
} catch (PDOException $ex) {
    die("Failed to connect to the database: " . $ex->getMessage());
}

//throw exceptions for errors
$db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

//return db rows using associative array
$db->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);
//initialise session
session_start();
?>
```

register.php – each query changed to reflect local DB instance

```
// Check if name already taken
$query = "
    SELECT
        1
    FROM login
    WHERE
        username = :username
";
```

login.php – Session variables added to capture user data, query changed and json_encode array added with data I needed to return.

```

if($login_ok)
{

    // This stores the user's data into the session at the index 'user'.
    // but first remove sensitive data from it
    unset($row['salt']);
    unset($row['password']);
    $_SESSION['user'] = $row;
    $_SESSION['username'] = $row['username'];
    $_SESSION['email'] = $row['email'];
    $_SESSION['about_me'] = $row['about_me'];
    $_SESSION['location'] = $row['location'];
    $_SESSION['avatar'] = $row['avatar'];

    echo json_encode(array(
        "username" => $row['username'],
        "email" => $row['email'],
        "about_me" => $row['about_me'],
        "location" => $row['location'],
        "avatar" => str_replace(' ', '%20', $row['avatar']),
        "id" => $row['id']
    ));

}
else
{

    $result = "Login Failed: Invalid Details";
    echo json_encode($result);

}
}

```

- Haversine great circle distance calculator (2017). Available at:
<http://www.movable-type.co.uk/scripts/latlong.html> [Accessed 27 Mar. 2017].
- Used to calculate great circle distance between user and reviews submitted in geolocation.js


```

//haversine function to return great circle distance between two points
LatLon.prototype.distanceTo = function(point, radius) {
  if (!(point instanceof LatLon)) throw new TypeError('point is not LatLon object');
  radius = (radius === undefined) ? 6371e3 : Number(radius);

  var R = radius;
  var v1 = toRad(this.lat), a1 = toRad(this.lon);
  var v2 = toRad(point.lat), a2 = toRad(point.lon);
  var v_total = v2 - v1;
  var a_total = a2 - a1;

  var a = Math.sin(v_total/2) * Math.sin(v_total/2)
    + Math.cos(v1) * Math.cos(v2)
    * Math.sin(a_total/2) * Math.sin(a_total/2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c;

  return d;
};

function LatLon(lat, lon) {
  // allow instantiation without 'new'
  if (!(this instanceof LatLon)) return new LatLon(lat, lon);
  this.lat = Number(lat);
  this.lon = Number(lon);
}

//calculate radians
function toRad(x) {
  return x * Math.PI / 180;
}

```

Libraries imported:

Raty star rating (2016). Available at: <https://github.com/wbotelhos/raty/> [Accessed 02 Feb. 2017].

- Used for Star rating on movie page (index.php).

Simple HTML DOM Parser (2017). Available at: <http://simplehtmldom.sourceforge.net/> [Accessed 25 Feb. 2017].

- Used to allow me to parse HTML content in scraperv2.php.

6 Appendix A

Interview questions & responses.

Q1: How frequently do you watch movies or TV shows per month?

A1: Beke Hollenbach: More than 5 a month, more movies than TV but generally more than 5.

A2: Sean Sutton: About 2 or 3 a month in terms of TV shows, and about 1 a month or so for movies.

A3: Tim O'Leary: Not much at the moment, I would say maybe 1 a month at this stage.

A4: David Sullivan: I watch roughly 4 to 5 movies and TV shows each month, on average.

A5: John O'Shea: Maybe 4 times a month at most.

Q2: How do you currently keep track of the movies you watch and what is most frustrating about it?

A1: Beke Hollenbach: I keep a list written in a notepad on my phone, and add to it for each movie I see. It's frustrating because it's quite basic.

A2: Sean Sutton: I do the same as Beke, but I forget sometimes to write them down in the phone. It's not very intuitive.

A3: Tim O'Leary: I mostly just try and remember the movies but it would be nice to have an easy way to keep track of them.

A4: David Sullivan: I use Flixster to keep track, it's fairly good but I would like the ability to share what I've watched with the lads.

A5: John O'Shea: Like the other lads I mostly keep notes in my phone, its tough to remember what you watched and I think an app to do that and share your content would be great.

Q3: What features in an app to rate/review movies would you most like to see?

A1: Beke Hollenbach: A nice look and feel and the ability to share your favourite movie lists with friends would be great.

A2: Sean Sutton: I'm with Beke. If it's easy to use and it had say links to your nearest cinema that would be awesome.

A3: Tim O'Leary: I think I would like the geo-location functionality aswell, and once the app is easy to use, that would be important to me.

A4: David Sullivan: Ease of use and the ability to send my lists to friends would be good yeah.

A5: John O'Shea: Overall it has to be snappy and the above features would be good, especially the geo-location function.

I asked the group to define ease of use. The general consensus I got back was that it had to be responsive, modern-looking and the user shouldn't have to wait too long (1-2 seconds at most) to complete a task in the app.

6.1 Project Proposal

Project Proposal

Movie Rating App

Eoin Sutton, x13116053, x13116053@student.ncirl.ie

BSc (Hons) in Computing

Software Development

13/10/2016

6.1.1 Objectives

The objectives of this project are to create a mobile responsive website/app to gather information about movies and tv shows from the web and enable users to rate and review them. It will track trends and ratings from users over time and also have geo-location functionality to show the nearest cinemas to the user and show times with links to buy tickets.

Users will have the ability to create an account and save the movies they have reviewed/rated and also share their movie lists with friends. It is designed to be a one stop shop for people to see reviews, recommendations and ratings for movies, where they can also find their nearest cinema and film times to book tickets to those films.

In order to share your list of favourite movies or the movies you've rated and would recommend, you will need to register on the site and this will allow you to save the lists to a database and share them with friends.

You will also be able to rate or review the reviewers – this will help users see who are regular and popular reviewers.

6.1.2 Background

Digital media like movies and TV Shows have only grown in popularity in the last few years. Services like Netflix have made it incredibly easy to watch content whenever and wherever you are. For fans of movies, wouldn't it be great to have an application that will enable you to rate and review the content you've watched, share the lists of your favourite films with friends, gather information about those films and read the reviews of trusted reviewers who you too can rate?

This project was part of the list of approved projects in NCI. I chose this project because, as a fan of movies in particular I found that although there are similar apps out there, none of them combine all of the functionality I would like. Some apps and website such as IMDB (The Internet Movies Database) only have complete functionality for the US and a few other select countries. With other services like Flixster, I cannot share the lists of movies I've rated with friends.

I've often found that between friends it's difficult to remember what films we've watched and rated and what we would recommend to each other. Often we resorted to writing down the names of movies in lists on our phones in plain text.

Ideally I intend to create this app, through a mobile responsive website, that will collate much of this functionality for movie lovers like myself.

6.1.3 Technical Approach

To create this application, I intend to utilise HTML5, Javascript and related libraries like jQuery, CSS, PHP and MySQL for the database backend.

I have conducted some research on current apps and services that exist. The main one available is IMDB. They have an app for iOS and Android and a mobile website. Some other players are Flixster and TodoMovies. All have similar features but with different approaches and technologies.

For this app, I researched some Javascript frameworks that would make the creation of a mobile responsive site more straightforward. jQuery and jQuery mobile are set libraries that can help me in this regard, as well as looking quite modern and user friendly.

For the geo location functionality I will be using the Google Maps Javascript API. I have some familiarity of this from a previous project and it's fairly flexible and has good functionality. For the database end, I will be using PHP and MySQL. CSS will be used for styling.

In terms of getting the film information online, there are a few APIs available. IMDB has one but it seems quite restrictive. There is also the OMDb or Open Movie Database which has freely available movie information which I hope to use in the project.

I will begin requirements capture shortly and this will include a full requirements document with all requirements fleshed out. At this early stage it's hard to pin down the specific technologies and libraries I will be using and I expect for some to change as I move forward with the project.

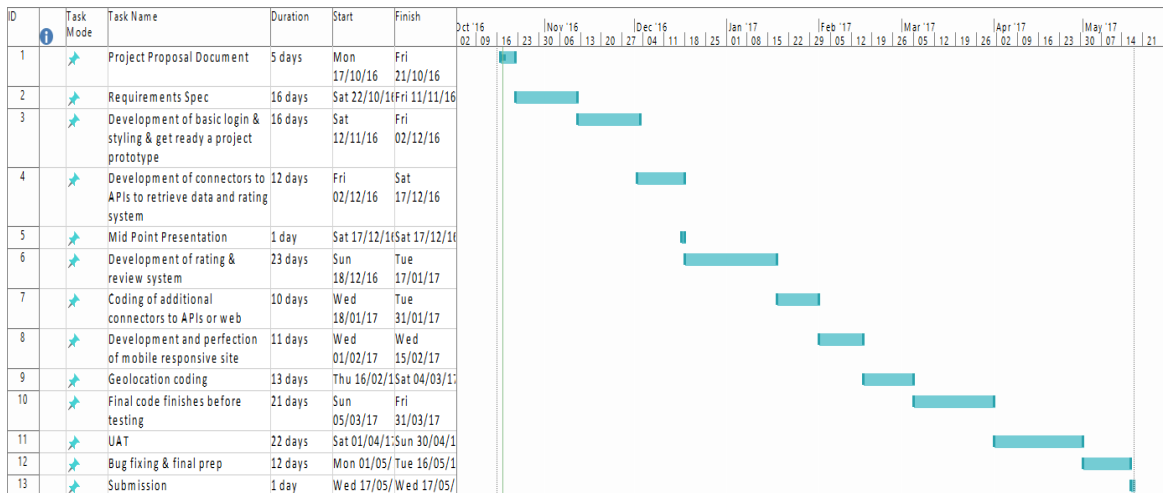
Implementation as far as development goes will be gradual, in a couple of weeks I hope to have started development of a basic website with basic functionality such as account registration, login, and a prototype of the styling. By Christmas I intend to have successfully connected to at least one API online and to display information relating to films the user searches, and an ability to rate/review. After Christmas I intend to focus more on APIs or scraping information from online myself (if needed),

the rating system for both movies and the reviewers and on geolocation functionality.

6.1.4 Project Plan

Below is a Gantt chart showing the project plan from October 2016 to May 2017. I have split my project plan into several manageable steps. Beginning with this project proposal I will then complete the requirements spec and subsequently begin development of the application. Initially I hope to get together a basic mobile website and login and a connector to at least one web API to gather movie/tv show data. Once complete I plan to get a basic rating system working for the gathered data, perhaps only saving the ratings to a database and relying on a net connection to gather the movie info (not sure yet). That is what I hope to show at the midpoint presentation.

Over Christmas and into the new year then I hope to further develop the rating system to enable ratings of reviewers themselves, and also further developing the API connectors or scraping from the web via my own code if necessary. February will be spent refining the mobile version of the website, enabling some stats showing trends of users favourite movies as well as the coding of the geolocation functionality with a map interface if I can. The remaining few weeks in March will be for refinement of the code and tidying up bits and pieces to be ready for user testing. UAT will begin in April and I have allowed most of the month for this as I plan to get my friends to test and some of this time will be used for bug fixing. Final bug fixing and prep is at the beginning of May with the app to be completed before the submission date of 17th May.



6.1.5 Technical Details

As mentioned at this stage of the project I believe it will be made up of HTML, CSS and Javascript at the front end, and PHP and MySQL at the backend. jQuery mobile will likely feature as a library. The app will be mobile responsive.

For the movie data I will be connecting to publicly available APIs such as the Open Movie Database. Where necessary I will scrape data from other movie related websites although I am not sure at this point what technology I will use to do that. Good responsiveness and user friendliness will be crucial though the priority is getting the movie info from the web and having a working rating system. If I can, I will package the website into specific Android or iOS native apps via a packager such as Cordova (PhoneGap) for release to the respective app stores.

6.1.6 Evaluation

This app will be unit tested continuously throughout development, particularly by using the console.log() function with javascript to output various elements to the screen. Overall user testing will be completed by end users who are likely to want to use the app themselves i.e. my friends. I intend to ask them to register with the application, login, enter movie details of the movies they are watching or have

watched and rate them, add reviews, review each other, and test the geo location functionality. I hope to get them to use the app over a period of a few weeks to get a good picture of their usage habits and ensure most of the functionality is working. Ultimately I think this will be an interesting project to work on. Perhaps ambitious for me but I plan to do my best to get all functionality in and showcase my skills and hopefully learn something along the way.

Signature of student and date

6.2 Monthly Journals

Student name: Eoin Sutton

Programme: BSc in Computing

Month: **September**

My Achievements

As this is my first month I don't have much in terms of achievements. The scope of the project is now only kicking in and it is a lot to digest. I had the best intentions of starting the project over the summer but life got in the way!

The biggest challenge is to come up with an idea. One that is both doable in the time I have and enough to impress the examiners. I had ideas of maybe a java app or website but I am not as comfortable in those areas as I would like to be. I work as a technical analyst and developer in the area of Oracle databases, using PL/SQL and Oracle Apex, which is a development framework. It might be a good idea to stick with that area to leverage my skills, I just need to come up with an idea.

I work in an airport so maybe something that could be useful to passengers, or somewhere with a large group of employees. I used to work administering absence so maybe something around absence management or a version of. It's really difficult to come up with an idea – some more specific guidelines or topics from the college would have been a big help.

Or maybe something passengers in the airport could use, like a website/app that allows restaurant owners to post and manage their menus and special offers in one place.

Or I might leverage the knowledge I am getting from this semesters Multimedia App Development module and create an android app on one of the above.

Another idea is maybe creating a CPU/GPU benchmark tool for computers which runs several intensive algorithms and creates a score based on how quickly the task was performed. This score is then uploaded to a website to share and compare with others who use the tool.

Student name: Eoin Sutton

Programme: BSc in Computing

Month: **October**

My Achievements

This month I decided on a project to do. I had a look through the approved project list and chose project 60 – an app that will scrape movie/tv show data from the web and create a rating/review system for them, and would include geolocation functionality for listing nearest cinemas and film times.

I have an avid interest in films so this should be an interesting project. I spoke to the lecturer, Eugene, who's idea it was and he was happy for me to go ahead. Gave me some tips and pointers aswell, saying it would be good to have a review/rating system for the reviewers themselves, so people could review other reviewers. I thought this was a great idea and decided to include that in my project.

Other than that there hasn't been much else in terms of the project. I am starting to feel the pressure now. I don't even know if I have all the skills or knowhow to complete this project but I will try my best. The lecturer mentioned previous students had created their own APIs to "scrape" info from the web – to be honest this is the first I've heard of this scraping – it sounds complicated! I haven't a clue where to begin with that. I do have some experience connecting to free movie based APIs online like OMDb – I would hope this is enough. It would be interesting to learn this nonetheless. If I have time.

Student name: Eoin Sutton

Programme: BSc in Computing

Month: **November**

My Achievements

This month I met with my project supervisor, Eugene. He had a number of interesting suggestions around the movie app. He suggested adding the ability to "review the reviewer" which I thought was quite interesting. I also completed the project requirements document this month. This was a huge task and consisted of creating diagrams and use cases for the main functionality of the app. It involved me interviewing my friends to get some user requirements and creating mockups of what I think the app should look like.

It was a good exercise though as it really gave me a good idea of what the app should look like and how I should bring it together. It was tough but I got there in the end with a 30 page document and I got good feedback from Eugene on a subsequent meeting. He also suggested perhaps sharing the movie lists via facebook/twitter which was interesting but I'm not sure if feasible for myself – time will tell!

Despite the massive workload I managed to complete a draft of how the site will look. It's in HTML/CSS and jquery mobile. No functionality just yet but it's a general outline of the look and flow of the app. Next is to try get some functionality before

the mid point presentation – very difficult though as time is at a premium given I have 4 other projects to hand up that week also!!

Student name: Eoin Sutton

Programme (e.g., BSc in Computing): BSc in Computing

Month: **December**

My Achievements

This month I began proper work on the app prototype. My main concern was getting a decent look and feel to the site (mobile especially) and so I began by creating the site with jQuery mobile. I went with a darker theme using black overall for background and red and white content. I thought it looked quite original. I created the skeleton of the site and its pages including the search page and the movie page which shows movie details. I did some work with javascript to connect the site to OMDb (the free open movie database) so when a person searches for a movie title it will return all the details of that movie including synopsis, cast, release date and poster. I also implemented a way to rate and review the movie.

I also met with my mentor again this month who suggested some additional features such as implementing some kind of geo-fencing so that the main page will show popular movies based on the ratings people in your vicinity have given them (i.e. it might be within 50km). I think this is a good idea and certainly a challenge to implement.

It was a tough month with all of my projects due for the other modules so it was quite stressful. I was surprised I was able to get the prototype working to the level I did (even though there is still a fair bit to go yet). I had the mid-point presentation this month also which I felt went well although the look of the site was not as well received as I thought. I will have to re-think its design and also get to work on the desktop version of the site. Overall though a tough but productive month.

Student name: Eoin Sutton

Programme (e.g., BSc in Computing): BSc in Computing

Month: **January**

My Achievements

This month wasn't the most productive for my project as I had a number of other things to do. After New Year I took a good break from study and project work which I felt was necessary, or I would have burned out! I also had three exams to study for which were quite tough. I got through the exams ok and went on a new year getaway to Rome for a weekend with my girlfriend.

It was only when I got back, and started Semester 2 that I really got thinking about the project. I still haven't decided what look and feel I will bring to it from the dark theme it has now, but I researched some colour pallet tools online which give suggestions for complimentary colours. In terms of work done I managed to get the login working via php and associated database set up with the correct table etc. Registration and login now working, which I'm happy with.

Student name: Eoin Sutton

Programme (e.g., BSc in Computing): BSc in Computing

Month: **February**

My Achievements

This month I had more time to concentrate on the main project, even though the work from other modules is building up. I fixed some login issues, decided to change the html pages to php and use that along with jQuery mobile which I feel works much better. I initially developed the web scraper API in python & BeautifulSoup but found it difficult to link javascript and php to the python script. It proved somewhat impossible, so I decided to re-write the entire API in PHP. So now PHP goes and fetches movie data from IMDB. It's a little slower than I'd like but not sure there is much I can do about that at present.

I also met my supervisor Eugene again to discuss the project and its progress. I'm a little bit behind where I would like to be but still making good progress. Other projects and assignments are starting to build up so putting some pressure on me. I feel the college are putting way too much of a workload on us 4th years at the moment. 4th year so far (including semester 1) has been incredible tough and stressful. The workload is immense. Just have to get through it as best I can. I'm not going to over stress myself!

Student name: Eoin Sutton

Programme (e.g., BSc in Computing): BSc in Computing

Month: **March**

My Achievements

This month I really worked on my main project and got a whole lot done. In fact, at the time of writing (early April) I would say I am 95% done. The web scraper is complete in PHP. The movie reviews functionality is done and so is the functionality to view and review users of the site. On save of a review, I capture a users GPS location – this is used when the Box Office/What's Hot page is loaded. On load of that, I loop through all reviews over 3 stars and check whether their location is within 100km of the users current position. I then display those movies – so only what is popular near you is shown.

I also began saving movie data to the database to speed up retrieval – so on click of a movie now, on say the My List page, if it exists in the database we'll fetch it from there. I've also implemented a facebook share link on the my lists page, which shares a page of the app with your reviewed movies that does not require logon. It's not working in my localhost at the moment so I will need to test that again. I've also uploaded everything to a free web hosting service, though that is still a work in progress.

Overall very happy with the progress. I will continue some small bug fixing, and maybe try to get a lighter theme created with CSS. And start on the technical report again so I can finish it. Another daunting task! But the end is in sight!

6.3 Supervisor Meetings

Meeting 1

Date	15/10/2016
Time	1pm
Duration of Meeting	15 mins
Current Challenges Discuss movie app idea	
Goals of Meeting (Student to fill out) Discuss movie app idea – get permission and ideas	
Goals/Actions for next Meeting (Student and Lecturer to fill out) Get started on the project	

Meeting 2

Date	19/11/2016
Time	1pm
Duration of Meeting	30 mins
Current Challenges	

Discuss requirements document and ideas for app
Goals of Meeting (Student to fill out) Get go ahead on requirements via requirements document
Goals/Actions for next Meeting (Student and Lecturer to fill out) Potentially add 'Review the reviewer' functionality to the app, and also try to get sharing enabled via Facebook.

Meeting 3

Date	13/12/2016
Time	7pm
Duration of Meeting	30 mins
Current Challenges Discuss current state of project.	
Goals of Meeting (Student to fill out) Get feedback on the current layout.	
Goals/Actions for next Meeting (Student and Lecturer to fill out) Look at implementing some kind of geo fencing ability to show movies popular around the user.	

Meeting 4

Date	23/02/2017
Time	6pm
Duration of Meeting	30 mins
Current Challenges Implementing the main functionality	
Goals of Meeting (Student to fill out) Discuss current state of the project, get feedback	
Goals/Actions for next Meeting (Student and Lecturer to fill out) Try get a lighter theme running as the current one is a bit dark. Good feedback otherwise.	

Meeting 5

Date	30/03/2017
Time	6pm
Duration of Meeting	30 mins
Current Challenges Tidying up the code/getting lighter theme going – bug fix and finish	
Goals of Meeting (Student to fill out) Get feedback, showcase current functionality to supervisor	
Goals/Actions for next Meeting (Student and Lecturer to fill out) Most of the functionality is done – feedback was good when showcasing the location specific movie lists, reviewing functionality and profile. Lighter theme to be worked on for final presentation.	

6.4 Links

Online version of the app is hosted at:

<https://movie-io.byethost7.com/index.php>

GitHub Link:

<https://github.com/e-sutton/Movie-io>