Coursera
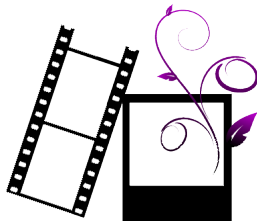## The Hong Kong University of Science and Technology

Full Stack Web Development Specialization

Capstone Project

Author:
Tiago Justino

Supervisor:
Jogesh Muppala

April 29, 2017

1 Introduction

MyMovies is a social network for cinema fans. It keeps track of movies you've watched, want to watch and don't want to watch. By leveraging the IMDB database it allows you to add custom pieces of information, such as, rating and comments. Also, the social network aspect brings you the possibility of discovering new movies you want to watch and new points of view on movies you've watched at the same time that it allows you to share good moments with people you love.

As a bonus, by keeping track of people's movie watching habits MyMovies is able to help the cinema industry on targeting their customer more precisely.

1.1 Expected List of Features

Inside the scope of this project we plan on achieving the following features:

- The user will be able to create an account and login to the system with or without Facebook. The user will also be able to link or unlink their MyMovies account to their Facebook account;

- The user will be able to invite other users to be their friends on My-Movie. The invited user will be able to accept or decline the invite;

- The user will be able to invite their Facebook friends to join MyMovies.

- The user will be able to follow other users. Users added as friends are followed by default. The user will also be able unfollow friend without unfriending them;

- The user will be able to mark a movie as watching, watched, want to watch or don't want to watch;

- For movies previously marked as watched, the user will be able to favorite and to add rating and comments;

- The user will be able to *fan* names, such as actors, actresses, directors and producers;

- Movies marked as watching are automatically marked as watched after the movie duration (known from imdb api);

- When marking a movie as watching or watched the user will be able to mark a friend as "watching with";

- The user will be able to add a date (day, month or year) for movies watched in the past. Date is not mandatory;

- The user will be able to mark a movie as watched more than once;

- The user will be able to access a timeline page where they can see the activities of whom they follow. The timeline will show activities such as watching and watched movies, and comments;

- The user will be able to like or comment on friend's activities,

- The user will be able to mark their comment (or part of it) as spoiler,

- Comments marked as spoiler will appear with a warning message on a user timeline if they haven't watched that movie,

- The user will be able to create and join groups (private and public), such as "Movie Club at Work" or "Stanley Kubrick Fans",

- The user will be able to recommend a movie to a friend.

2 Design and Implementation

2.1 The REST API Specification

Here we describe all the endpoints and the supported operations. Wikipedia[1] was used as reference.

/users

> **POST** Registers a user.
>
> **GET** List registered users.

/users/:id

> **GET** Get user details.
>
> **PUT** Update user details.
>
> **DELETE** Delete a user account.

/invitation

> **POST** When a loggedin user invites another user to be friends.
>
> **GET** Lists friendship invitations.

/invitation/:id

> **DELETE** Cancel a friendship invitation.

/follow

> **POST** When a loggedin user start following another user.
> **GET** Lists people you follow.

/follow/:id

> **DELETE** Stop following someone.

/friendship

> **GET** Lists friends.

/friendship

> **DELETE** Stops being friend to someone.

/friendshiprequest

> **GET** Lists friendship requests.

/friendshiprequest/:id

> **GET** Get friendship request details.
> **POST** Accepts a friendship request.
> **DELETE** Reject a friendship request.

/watching

> **GET** Lists movies currently marked as watching.
> **POST** Marks a movie as watching.

/watching/:id

> **DELETE** Removes a movie from watching.
> **PUT** Change a watching event (add people to watching with).

/watched

> **GET** Lists movies marked as watched.
> **POST** Adds a movie to watched list

/watched/:id

> **GET** Gets details of a watching event.
>
> **PUT** Change a watch event (adds favorite, rating or comments). The user can also change date they watched the movie, times, and with whom.
>
> **DELETE** Remove a movie from the watched list.

/wanttowatch

> **GET** Lists movies marked as want to watch.
>
> **POST** Adds a movie to want to watch list

/wanttowatch/:id

> **DELETE** Remove a movie from the want to watch list.

/dontwanttowatch

> **GET** Lists movies marked as don't want to watch.
>
> **POST** Adds a movie to don't want to watch list

/dontwanttowatch/:id

> **DELETE** Remove a movie from the don't want to watch list.

/group

> **GET** Lists created groups.
>
> **POST** Creates a group.

/group/:id

> **GET** Get a group details.
>
> **PUT** Change group details. Add people to group.
>
> **DELETE** Deletes a group.

/groupjoin

> **POST** Asks to join a group.
>
> **GET** Lists pending requests to join groups.

/group/:id

> **DELETE** Cancels a pending request to join a group.

2.2 Front-end Architecture Design

2.3 Database Schemas, Design and Structure

2.4 Communication

3 Conclusions

References

[1] Wikipedia - rest. `https://en.wikipedia.org/wiki/Representational_state_transfer`. Accessed: 2017-04-29.