Northeastern University
CS 4500 – Software Development

Rotten Tomatillos Project Description

Travis Bergmann
Xuanyu Li
Ranran He
Zewei Chen
Qiye Chen

1/25/2018

Overview of the Problem

The clients of our project are regular internet users, especially people who are interested in movies. We know there are many competitors in this market already, but both our client and many target customers have dissatisfactions with these existing services. None of the competitors seem to provide the complete social movie experience. Movie databases, movie recommendation services, and social networks where friends recommend movies to each other are all separate. The goal of our team is to build a social media platform which offers an end to end experience to our customers. We use third party API to retrieve movie information which users can then read from our website's user friendly interface. Our platform will recommend movies to users based off their interests and the interests of their friends. It will offer links to iTunes, Amazon Prime, or movie ticket systems which provides a more convenient way to help our customers purchase and watch movies directly. More importantly, it is a social media platform, movie lovers can find and connect with each other from here. Users are able to express their opinions about movies easily, which will affect what movies they are recommended in the future. Users can write reviews, rate movies, and even create a collection of their favorite movies they can then share with friends. We use this data to study user behavior and to help us to improve our machine learning algorithm which analyzes user preferences as well as the preferences of friends to intelligently find movies the user is likely to be interested in. The system gets smarter and provides better movie recommendations the more the user interacts with it. Lastly, the client is external to us, but we believe that by providing a more complete movie service and better user experience, we are going to build them a superior movie website.

Background

There is no legacy project for this project to extend or replace. Instead, this is an entirely new project inspired by existing systems such as IMDb, Netflix, and Facebook. It will have a movie database similar to IMDb where users can view movie information and write reviews. It will be able to suggest relevant movies to users like on Netflix. Finally it will have a social network friend system similar to Facebook. However, it will be the first system to integrate all of these features into a single website.

At first, the Rotten Tomatillos team will operate and maintain the system, but ideally by FY2019 the property will be acquired by a larger company which will then be responsible for operating the site.

Scope

Based on the basic ideas of this project and currently available resources, we will be building several things. The work will be divided into two major parts: the client side and the server side. By analysing system-facing and user-facing use cases, as seen in the use case file, we generate important things that the server side must support and also a bunch of things that the client side must offer. After satisfying those must-have features, we may want to include some nice-to-have features to make the project better, if time and resources permit. The below describes what we will be including in this project.

The server side will include the following (the order does not imply any priorities):

1. Database table(s) that store(s) registered user profiles, movie playlists, records of movie ratings and comments, friend lists, and so on. These are needed for basically all use cases. Since AWS offers both relational and non-relational DBs and we haven't had possible designs of database table schemas ready, we will come back to pick a DB at a later time.
2. A user data management component. This component will be responsible for managing user data and enabling other components access user data. This will be highly affecting how database table schemas, as described in section 1, will be designed. All data that is related to a particular user, such as his or her movie lists, friend lists, should go through this component.
3. A movie data management component. This component will be responsible for utilizing third party APIs to query any movie related information, managing all ratings and comments made by users, caching frequently searched movies, and returning any movie search results to the users. This will also be highly affecting how database table schemas, as described in section 1, will be designed. We assume that third party APIs are available.
4. Implementations of the machine learning algorithms. These algorithms will be used to generate movie recommendations for each user, based on a bunch of factors such as his or her preferences, movies rated, and movies touched by friends. These algorithms will be provided externally.
5. Storage that mainly stores log files. The logs will include the behaviors that users have performed from time to time. The system may utilize the logs to debug any

issues or improve any efficiencies. The movie recommendation algorithms may also use the logged data to improve the recommending process. AWS has storage services available.

The client side will include the following (the order does not imply any priorities):

1. A movie search functionality. This will be basically querying the server side and fetching any search results. If any result is found, users can view its information, such as descriptions, categorizations, and any ratings and comments against the movie made by users. In addition, it will show available resources for the movie, such as links to major video streaming websites, ticket information by theatres.
2. A user interaction functionality. Since this is not only a movie search engine but also a social website, this functionality will be offering users to take a look at what movies other users are liked, make connections with each other, and share movie info with friends. Users may feel like using an alternative Facebook/Twitter.
3. A user account management platform. Most of the features, as promised in the use cases, must be having users signed in their own accounts to proceed. Thus, this platform will allow users to register new accounts or log in to their accounts, and view or edit their account information, including their basic profiles, movie search history, movie ratings and comments, friend lists, and more.

The following are out-of-scope but nice-to-have features that we are considering:

1. Functionality that allows users use their Facebook/Amazon/Netflix accounts to sign in to the system. This can facilitate users to share their favorite movies with their friends on those existing platforms. The difficulty of this feature is it requires those platforms to have related APIs available as well as the user management component be able to handle accounts from different sources.
2. Expansion of the system to include entertainment types beyond movies. Music, games, and any others are also equally important for people to enjoy their lives. However, this requires the movie management component be able to query new types of data from new sources and support new features that are not for movies, which needs a lot of time, may be as many as for the movie part, to implement.
3. Feature that allows users to buy movie tickets directly on the platform, instead of just having links to websites where tickets can be bought. This will definitely bring a lot of convenience for users who are heavily interested in seeing certain movies at nearby theatres. However, this requires the theatres not only have online platforms but also have some APIs available; in addition, this feature requires payment integration, which needs access to payment platforms such as PayPal, Visa, Master, and so on, and also the payment process must be secured. This will consume a lot of non-familiar efforts to make the feature just work.

4. VIP accounts. We haven’t thought of what kind of features can be privileges that only certain users may have, unless we have made a gougeres business model.