

Popcorn

Project Team: Sytiva Wheeler , Alana Traylor , Sonny Proctor,
Pengxu Zhu

Table of Contents

1. Project Definition
2. Project Requirements
3. Project Specification
4. System – Design Perspective
5. System – Analysis Perspective
6. Project Scrum Report
7. Subsystems
 - a. 7.1. Alana Traylor
 - b. 7.2. Sytiva Wheeler
 - c. 7.3. Sonny
 - d. 7.4. Pengxu Zhu / Aaron
8. Complete System

Project Definition

Why (it is needed)

Due to the increase in streaming platforms due to the pandemic and the increase in screen time due to having to be indoors. The idea of a movie recommendation website that allows users to have access to all movies and the ability to filter through by which streaming service they own so as to make looking for a movie easier and more efficient sprung up.

What (is the goal of the project)

The goal of this project is to have a fully functioning website that recommends the user movies based on their favorite genres, the movies they add to their favorites list and the movies they add to the want to watch list, both places located on the site.

How (how will it be achieved)

This will be achieved by using the IMDB API for a database that contains the movies, the movie posters, and movie descriptions. The Watchmode API which will contain the links to the streaming services attached to the movies and the location of which movie is on which streaming service. Azure will play two roles, one being the host server for the website as well as a database for the user info, preferences, favorites and watchlist information.

Project Requirements

The UI will be easy to navigate with buttons being easily seen and the movies and selections being the more forefront and eye-catching images. While the hope is for the website to run fast and smoothly, the possibility of it running slow is higher due to the numerous API calls from both APIs and the use of Bootstrap 5.

Any desktop or laptop computer capable of using the operating system Microsoft 10 and the web browsers Google Chrome and Microsoft Edge will be suitable for running this web application. The database used is a Microsoft Azure Database. Two API (Watchmode on RapidAPI, IMDB API on imdb-api.com) are used to retrieve necessary information.

All user information will be stored in a database that each member has access to with their microsoft account. The database is secured with a password. Any information the user enters is sent/retrieved from one page to/from another in the web application using POST instead of GET to ensure sensitive information is not visible in any page's URL.

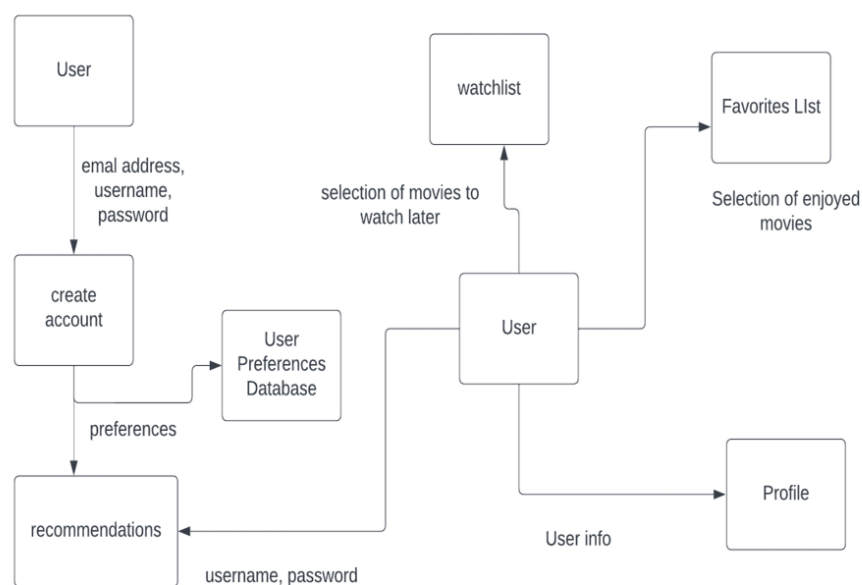
Project Specification

The main focus of this project is to recommend users movies based on the genres they have chosen and being able to filter through by streaming service and have the link to the exact movie on the streaming service. Visual Studio is the development environment used to create this application. The languages used include PHP, HTML, CSS, SQL and Python. The libraries/frameworks used are Bootstrap, Material Design Bootstrap. The platforms available are desktop only and the genre of application is a web application.

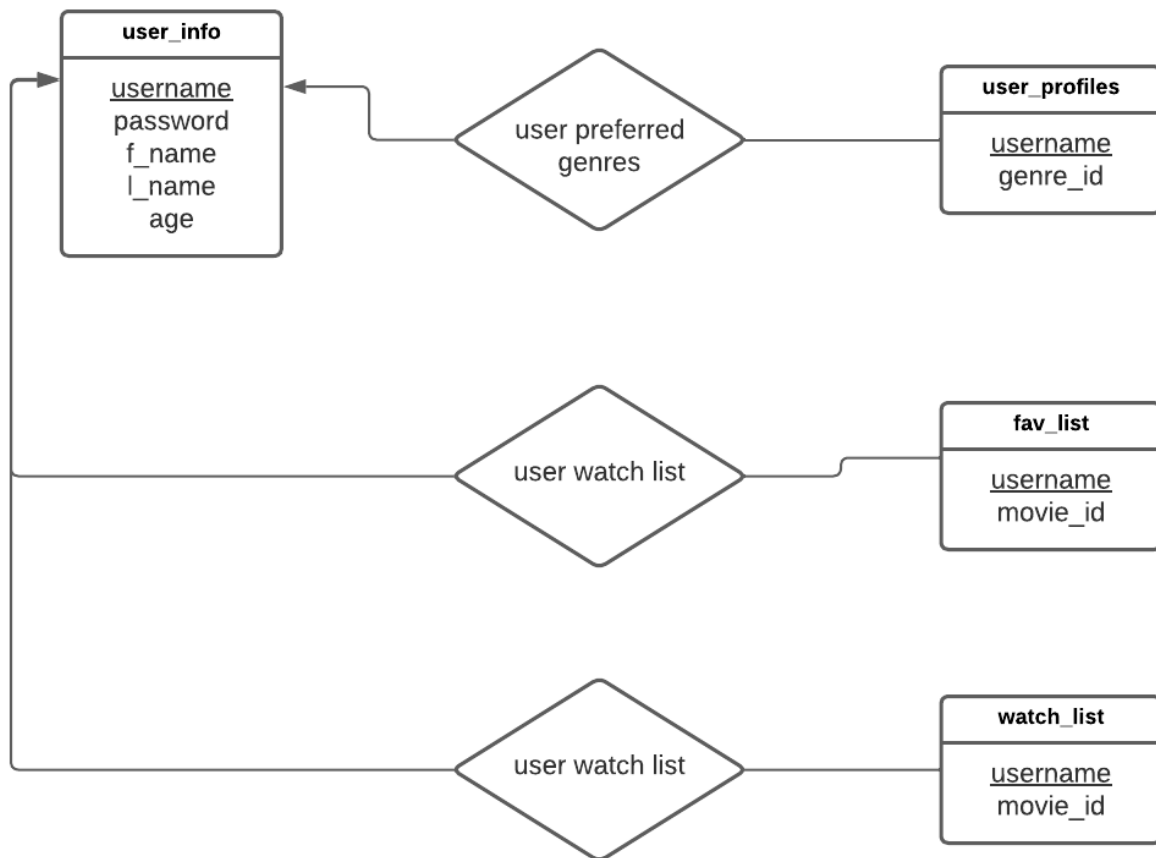
System – Design Perspective

The user will be given 2 main controls on the welcome page, sign in and create an account. Sign in will direct them straight to the homepage and from there they will have controls to filter by genre and streaming service. Also located on the homepage they have a drop down menu that has controls to the user profile, favorites list, watch list, navigation back to the homepage and logout button. The create account control will lead to the page where they put in information needed to create their account and then the page to select at least 3 favorite genres and then directed to the homepage. Input and output will be retrieved and displayed using HTML and PHP.

DataFlow Diagram



Database ER model



Overall operation - System Model

The User Interface, made with css, bootstrap, php, jquery/AJAX and html, will interact with the IMDB API, by retrieving movies the user will prefer. The user profile generated when the user creates their account will be what determines said users preference in movies. The interaction with the api and UI will involve retrieving movies based on that user profile and displaying them to the user after they have signed into the website. The user will be able to go to their own profile page and view their favorites and their watch list in the navigation bar at the top of each page, which has movies they have seen and favorited and movies they would like to see respectively. When the user creates an account, their sign in credentials will be inserted into the database in Microsoft Azure unless it uses a username or email already in use by another user. After which the user is redirected to the user profile creation page where they will choose which genres of movies they prefer to watch, those genres will be inserted into

their profile record in the Microsoft Azure database, then they will be redirected to the homepage, which will populate based on the movie genres they chose.

System – Analysis Perspective

The subsystems are a Microsoft Azure relational database, the website hosted using Microsoft Azure, the Watchmode API and the IMDB API. Below is a table containing the data dictionary.

Data Dictionary

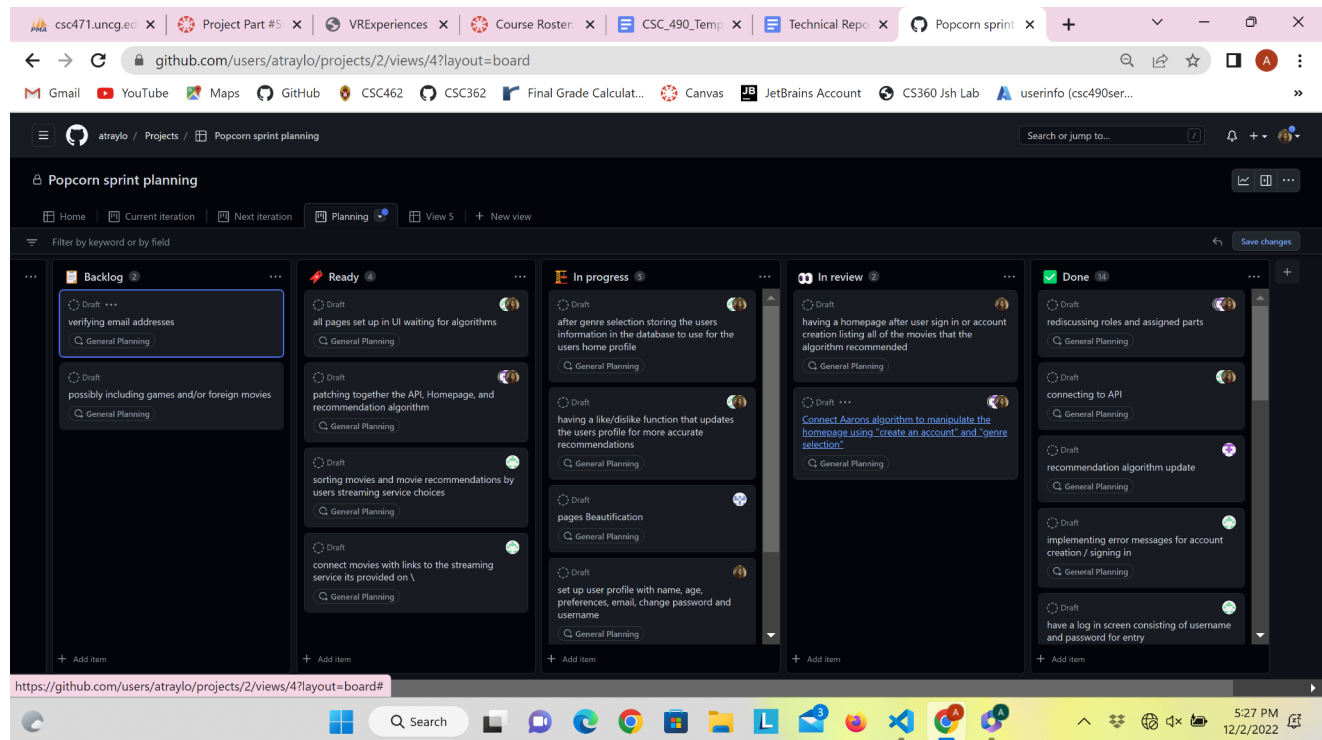
Name	Data Type	Description
imdb id / movie id	String	ID of specific movie item in the Internet Movie Database.
Movie Poster	Image url	Image depicting a specific movie.
Plot details	String	Summary of a movies story
Movie Title	String	Title of each movie
Profile	String	Information about the user (preferred movie genres, name, email address, age).
Favorites List	SQL query result / array object	List of movies the user has liked and wants to save in a list of their favorite movies.
Watch list	SQL query result / array object	List of movies user picks that they would like to watch in the future.
Streaming platforms	Int	Code for one of the various services that stream movies (Netflix, HBO max, Disney+, etc.)

Algorithm Analysis

The overall system for displaying movies for either of the user lists (favorites and watch list) is $O(n)$ as retrieving each movie the user stored will depend on the amount of movies in each list. To display movies to the homepage, both initial population of the movies on login and after sorting by either genre, streaming service or both, will take $O(n)$ as the JSON retrieved from the API will need to be parsed through and all of the movie information displayed in their own bootstrap cards.

Signing in takes $O(n)$ since the table needs to be searched to check if the record with matching password and username exists. Creating an account takes $O(n)$ as it involves checking the already existing records for the username and email used in the account creation.

6. Project Scrum Report

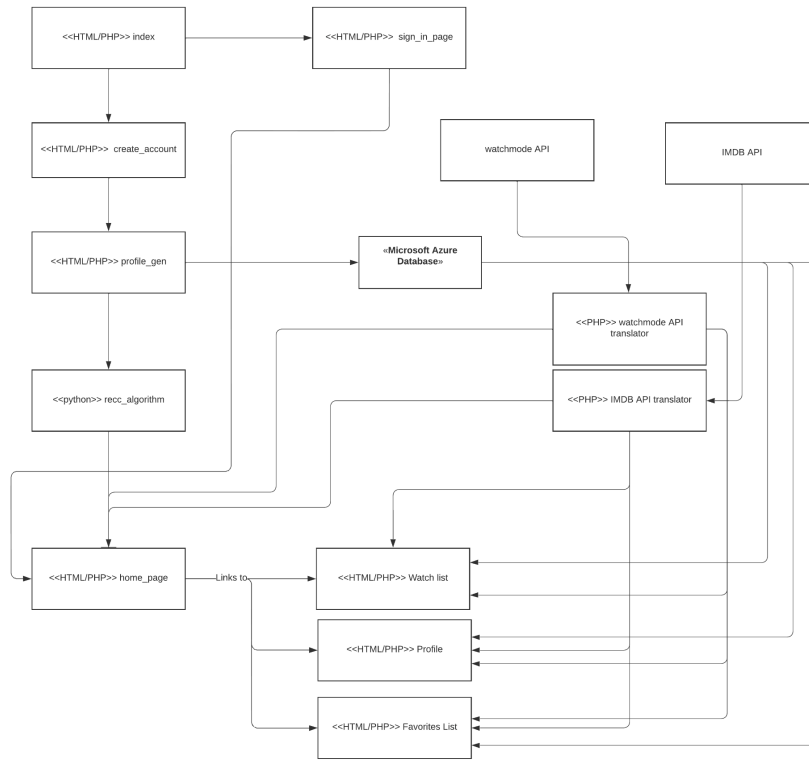


7. Subsystems

7.1 Subsystem 1 – Alana Traylor - *Project management, lead the direction of the design and layout, website building, API connection, assisting with database and API connectivity.*

Initial design and model

UML Diagram
Alana Taylor, Sylvia Winesler | December 2, 2022



Data Dictionary

Name	Data Type	Description
imdb id / movie id	String	ID of specific movie item in the Internet Movie Database.
Movie Poster	Image url	Image depicting a specific movie.
Plot details	String	Summary of a movies story
Movie Title	String	Title of each movie
Profile	String	Information about the user (preferred movie genres, name, email address, age).

Favorites List	SQL query result / array object	List of movies the user has liked and wants to save in a list of their favorite movies.
Watch list	SQL query result / array object	List of movies user picks that they would like to watch in the future.
Streaming platforms	Int	Code for one of the various services that stream movies (Netflix, HBO max, Disney+, etc.)

Reason for refinement:Pros

- JS, HTML and CSS worked better and easier for the connection to Sytiva server setup and the functionality for contribution to additional UI designs such as buttons and locating to and from the page from other pages proved easier.
- New APIs prove to have more useful information along with easier access to them and were able to make more calls as needed without interruption.

Reason for refinement:Cons

- Having to rework the design or functionality from within JS or CSS in order for it to display and implement new functions properly.
- The new APIs offer the information in the form of nested arrays which aren't the biggest problem to get to but it can cause problems if the arrays are not fully expressed.

Changes from initial model

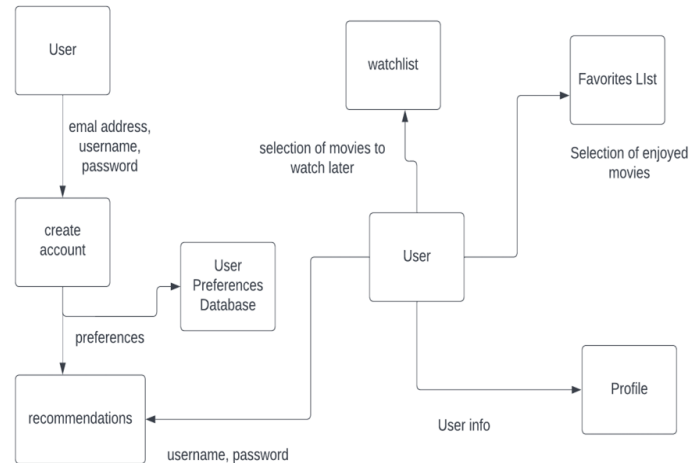
- Using javascript, HTML, and CSS for home page design
- Connecting API database to algorithm instead of preproduced .csv file
- Using imdb-api.com instead of rapid API
- User profile consisting of username, change of password option, possible avatar or picture, age, and preferences

Refined model analysis

Calling into each nested array that's returned from the JSON would be $O(n)$ since you need the title and image for each movie that appears in the JSON. Switching the

algorithm to call from the API instead of a premade CSV holds the same analysis of $O(n)$ since the calls remain the same.

Refined design (Diagram and Description)



Scrum Backlog

Milestone	Title	Assignees	Status	Planning	Labels	Estimate
	rediscovering roles and assigned parts	atraylo, rt22766, S...	Done	General Planning	-	-
	connecting to API	atraylo and SytivaW...	Done	General Planning	-	-
	all pages set up in UI waiting for algorithms	atraylo and SytivaW...	Ready	General Planning	-	-
	patching together the API, Homepage, and recommendation algorithm	atraylo, rt22766, a...	Ready	General Planning	-	-
	having a homepage after user sign in or account creation listing all of the movies that the algorithm	atraylo	In review	General Planning	-	-
	after genre selection storing the users information in the database to use for the users home prof	atraylo and SytivaW...	In progress	General Planning	-	-
	Connect Aarons algorithm to manipulate the homepage using "create an account" and "genre sel	atraylo, rt22766, a...	In review	General Planning	-	-
	having a like/dislike function that updates the users profile for more accurate recommendations	atraylo and SytivaW...	In progress	General Planning	-	-
	having a rough main page	atraylo	Done	General Planning	-	-
	set up user profile with name, age, preferences, email, change password and username	atraylo	In progress	General Planning	-	-
	Collect imdb movie information to a unified http adress	atraylo and SytivaW...	Done	General Planning	-	-
	progress report due 10/25	atraylo, rt22766, S...	Done	General Planning	-	-
	make each movie selection clickable and showing a description, genre, year released and rating	atraylo	In progress	General Planning	-	-

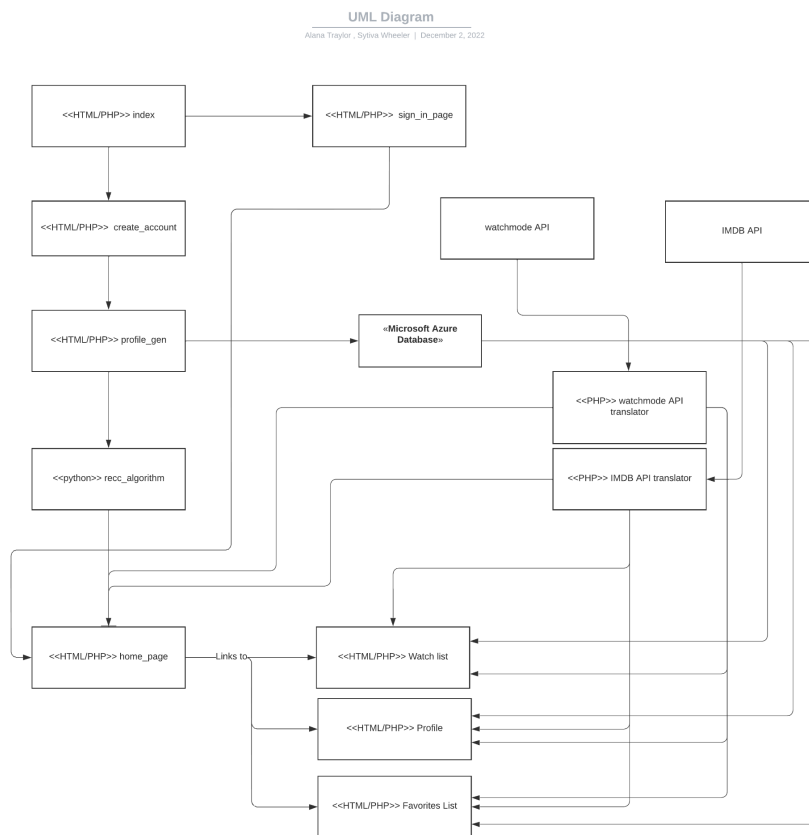
A functional approach with homepage creation, web page layout and design creation was taken. Everything from connecting the API to web page functionality uses Javascript and/ or PHP. All languages used are JavaScript, PHP, HTML, CSS.

User Training

This website includes all movies listed in IMDB from all streaming services located in one place. As a user you can switch between the streaming services in one place, along with different genres. Once you create an account, you are given the opportunity to pick your favorite 3 genres of movies and from there your homepage will be curated with movies recommended in those genres sorted by highest relevance. From the home page you are able to switch between genres and streaming services.

7.2 Subsystem 2 – Sytiva Wheeler - Database creation and management, webpage creation, general debugging / testing, API Management, UI layout and design

Initial design and model



Data dictionary

Refer to Section 7.1. Subsystem 1 – Alana Traylor for data dictionary

Reason for refinement: Pros

- To add more useful documentation.
- Make methods or files for certain blocks of code in order to reduce repeating code.
- More features from new frameworks or libraries can be implemented.

Reason for refinement: Cons

- Other aspects of the project will need to be reworked if the new API or database differs too much.
- Since a new api or database was implemented, we may not know how to remove newer bugs, lengthening the time debugging will take.
- Introduction of some frameworks, like bootstrap, causes the website to load slower.

Changes from initial model

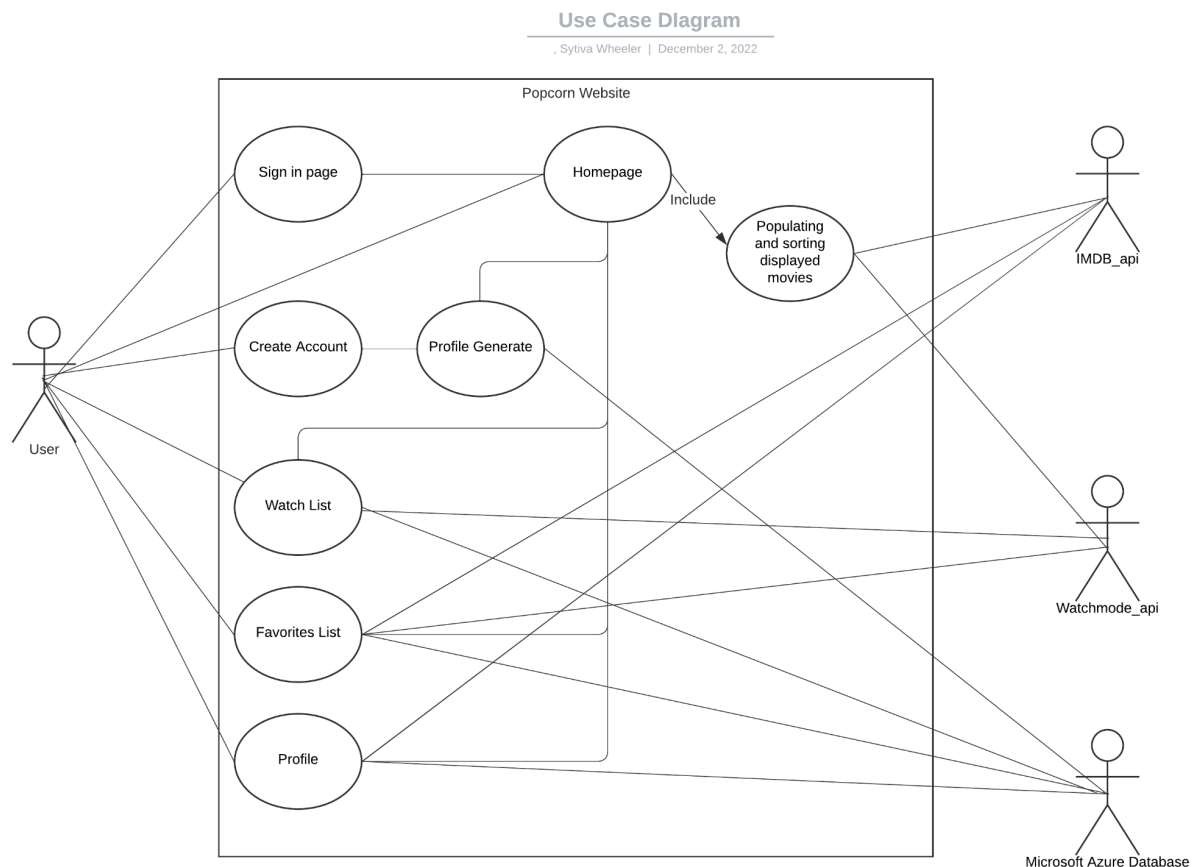
- Watchmode API introduced, AJAX/JQUERY introduced, MDBootstrap introduced, new IMDB API being utilized in place of rapidAPI IMDB api.
- Changed want to watch list to watch list, removed watched list, added favorites list instead.
- Watchmode API used for separating movies by streaming service and retrieving movie posters.
- Tables for watch list, favorites list, and user preferred genres added to database.
- Separate PHP files for code that is used in more than one web page has been created to reduce the amount of repeated code.
- AJAX/Jquery allow for web pages to update information on them without needing to refresh the page, enhancing the user experience.

Refined model analysis

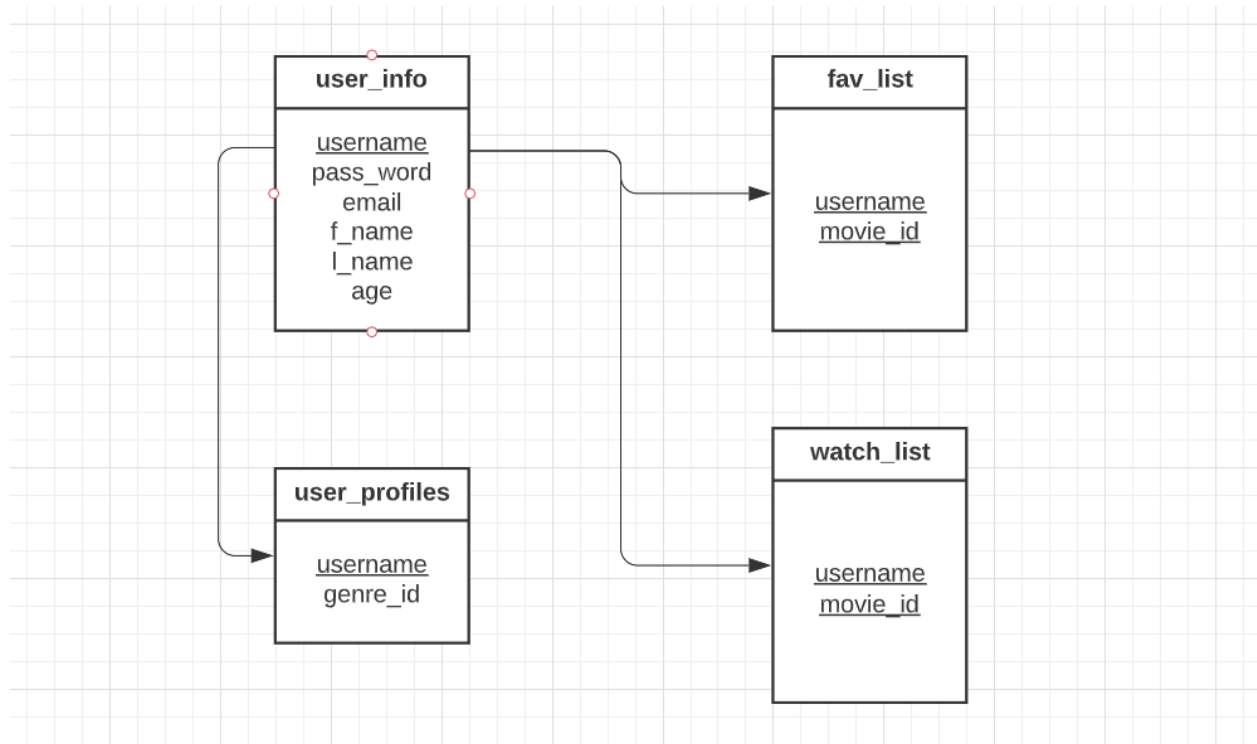
- Calling and displaying movies for the watchlist, favorites list or the homepage will take $O(n)$ since it will depend on the amount of movies in the respective lists or the amount of movies that will display on the homepage. It is this way for both movies populating the homepage on initial login/ account creation and for movies populating the homepage with sorting by either genre or streaming service.
- Placing movies into either favorites or watch list will take $O(1)$ as only one movie is ever inserted into the table at a time. Deleting movies from either list takes $O(n)$ since the record needs to be found in the table, then deleted.
- Creating an account will still take $O(n)$ as it involves checking the already existing records for the username and email used in the account creation.
- Signing in will still take $O(n)$ since the table needs to be searched to check if the record with matching password and username exists.

Refined design (Diagram and Description)

Here are the refined diagrams that represent the web application's functions. A use case diagram served as a better diagram to demonstrate and generalize the functionality of the web application.



An updated schema diagram was made to represent the Microsoft Azure database. After deciding to change the lists the users are able to add movies to, save their genre preferences to a user profile, and allow the input of additional user information in the form of name and age and ability to change already existing information, this is the schema diagram of the new database set up.



Scrum Backlog

For the Scrum Sprint Planning, refer to section 6.

Below is the sprint planning for all parts of the project I was involved in.

🔒 Popcorn sprint planning

📄 Home

📅 Current iteration

📅 Next iteration

📅 Planning

📅 View 5

+ New view










🔍 assignee:SytivaWheeler

15

×

↶

Save changes

Title	...	Assignees	...	Status	...	Planning
1 🔄 rediscovering roles and assigned parts		 atraylo, rt22766, S...	▼	<div>✔ Done</div>	▼	General Planning
2 🔄 connecting to API		 atraylo and SytivaW...	▼	<div>✔ Done</div>	▼	General Planning
5 🔄 implementing error messages for account creation / signing in		 SytivaWheeler	▼	<div>✔ Done</div>	▼	General Planning
6 🔄 have a log in screen consisting of username and password for entry		 SytivaWheeler	▼	<div>✔ Done</div>	▼	General Planning
7 🔄 all pages set up in UI waiting for algorithms		 atraylo and SytivaW...	▼	<div>🔹 Ready</div>	▼	General Planning
8 🔄 patching together the API, Homepage, and recommendation algorithm		 atraylo, rt22766, a...	▼	<div>🔹 Ready</div>	▼	General Planning
9 🔄 sorting movies and movie recommendations by users streaming service choices		 SytivaWheeler	▼	<div>✔ Done</div>	▼	General Planning
12 🔄 connect movies with links to the streaming service its provided on \		 SytivaWheeler	▼	<div>✔ Done</div>	▼	General Planning
14 🔄 after genre selection storing the users information in the database to use for the users home profi		 atraylo and SytivaW...	▼	<div>✔ Done</div>	▼	General Planning

+

You can use Control + Space to add an item

Popcorn sprint planning

Home | Current iteration | Next iteration | Planning | View 5 | + New view

assignee:SytivaWheeler 15 x

Save changes

Title	Assignees	Status	Planning
12 connect movies with links to the streaming service its provided on \	SytivaWheeler	Done	General Planning
14 after genre selection storing the users information in the database to use for the users home profi	atraylo and SytivaW...	Done	General Planning
15 having a create account page consisting of unique username creation and password creation	SytivaWheeler	Done	General Planning
17 Connect Aarons algorithm to manipulate the homepage using "create an account" and "genre sel	atraylo, rt22766, a...	In review	General Planning
19 having a like/dislike function that updates the users profile for more accurate recommendations	atraylo and SytivaW...	Backlog	General Planning
23 during account creation, after unique username and password creation, having the user pick up tc	SytivaWheeler	Done	General Planning
25 Collect imdb movie information to a unified http adress	atraylo and SytivaW...	Done	General Planning
26 progress report due 10/25	atraylo, rt22766, S...	Done	General Planning

+ You can use Control + Space to add an item

Coding Approach

The approach I took to coding my portion of the project was a functional approach. The languages I used were PHP, HTML and CSS, JQuery/AJAX and SQL.

User manual

Refer to Section 7.1. Subsystem 1 – Alana Traylor for user manual and / or user manual file.

Testing / Debugging

Dummy information was inserted into the database to test retrieval and insertion into it. These records were used to test recommendations by genre selection, sorting movies on homepage by genre, displaying and inserting user information on profile page, testing the display of the users favorites/watch lists and to test the streaming links.

7.3 Subsystem 3 – Sonny Proctor - Website user interface design, presentation preparations.

Designed CSS pages for web pages. Also created presentation material or assisted in the creation of presentation material for progress reports and project poster presentation for IAB presentation.

7.4 Subsystem 4 – Pengxu Zhu / Aaron - Project algorithm engineer, designing and implementing movie recommendation algorithms

Initial design and model

The very first models used collaborative filtering, of which there are user-CF and item-CF respectively. item-CF recommendation mechanism is an improved strategy of Amazon on a user-based mechanism, where the number of items is much smaller than the number of users in most of the web, and the number of items and similarity are more stable. The item-based mechanism is also a little better than the user-based mechanism in real time. Regarding user-cf, it is generally used when the number of items is greater than the number of users. The 2 types of collaborative filtering algorithms have obvious disadvantages and advantages for both types of recommendations. The initial plan to choose the model was because I found the dataset because I could use collaborative filtering to calculate the similarity between user and item (Pearson's correlation coefficient or cosine similarity) and the algorithm could recommend similar movies if the user searched for a new movie.

itemCF	UserCF
Suitable for smaller items	Suitable for less users
If the user has a new behavior, it will not cause new changes	it will cause new changes
Difficult to personalize recommendations for new users	easy
If a user is interested in a new movie, it can be immediately recommended to similar users	its cant, it need update data
It's hard to explain why it's recommended	easy

Improved model CNN

By looking at the types of fields in the dataset, we found that some are category fields, the usual processing is to convert these fields into one hot code, but fields like UserID, MovieID become very sparse and the dimensionality of the input expands dramatically, which we don't want to see, after all, my little laptop is not like the big factories that can handle hundreds of millions of dimensions of input at any one time :)So in pre-processing the data these fields were converted to numbers, which we used as an index for the embedding matrix, using the embedding layer in the first layer of the

network, with dimensions $(N, 32)$ and $(N, 16)$. The processing of film types takes one more step, sometimes a film has more than one film type, so that the index from the embedding matrix is an $(n, 32)$ matrix, because there are more than one type well, we have to sum this matrix into a $(1, 32)$ vector. The movie title is a special case, as instead of using a recurrent neural network, a text convolutional network is used, as will be explained below. After indexing the features from the embedding layer, each feature is passed into the fully-connected layer and the output is passed again into the fully-connected layer, resulting in two feature vectors of $(1, 200)$ user features and movie features respectively. Our aim is to train the user features and movie features to be used in the implementation of the recommendation function. Once these two features are obtained, it is possible to choose any way to fit the ratings. I have used two approaches, one is to do a vector multiplication of the two features as drawn above and regress the results against the true ratings, using MSE to optimize the loss. Because essentially this is a regression problem, the other way is to take the two features as input, pass them into the fully connected layer again, output a value and regress the output value on the true score, using MSE optimized loss.

CNN model performance and idea

1. Recommended films in the same genre

2. recommending movies that users like

3. What other movies have people who have seen this movie watched (or liked)

1 Idea is: By calculating the movie similarity, select top_k similar movies, and then add some random movies

2 Idea is Compute ratings for all movies using the user feature vector and movie feature matrix. Take the top_k ones with the highest score, and also add some random movies.

3 Idea is Select top_k people who like this movie to get features, top_k people score all movies, push the movie with the highest score to the user, and add some random movies

8. Complete System

The final product can be found at <https://pop-corn.azurewebsites.net>. The source code and user manual can be found either bundled with this project or on our github page: [atraylo/Popcorn \(github.com\)](https://github.com/atraylo/Popcorn).

Team Member Descriptions

- **Alana Traylor** - Project management, lead the direction of the design and layout. Website building, API connection, assisting with database and API connectivity
- **Sytiva Wheeler** - Project management, lead the direction of the design and layout. Website building, API connection, assisting with database and API connectivity.
- **Sonny Proctor** - Website user interface design, presentation preparations
- **Pengxu Zhu** - Project algorithm engineer, designing and implementing movie recommendation algorithms.