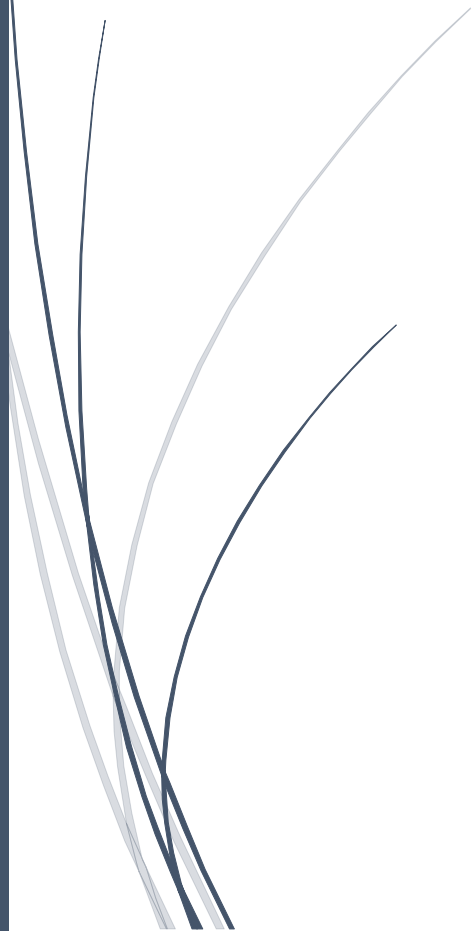
11/16/2020

FINAL REPORT

Team: 5Star

Project: FilmFinder

COMP9900: Computer Science / Information Technology Project



Name: Rohith Korupalli
Student ID: z5231304
Email: z5231304@ad.unsw.edu.au
Role: Scrum Master/ Full Stack Developer

Name: Leela Veera NagendraReddy Padala
Student ID: z5249335
Email: z5249335@ad.unsw.edu.au
Role: Full Stack developer

Name: Shubhankar Mathur
Student ID: z5229053
Email: z5229053@ad.unsw.edu.au
Role: Full Stack Developer

Name: Mustafa Merchant
Student ID: z5238889
Email: z5238889@ad.unsw.edu.au
Role: Full Stack Developer

Name: Karan Inder Singh
Student ID: z5232648
Email: z5232648@ad.unsw.edu.au
Role: Full Stack Developer

Table of Contents

Overview

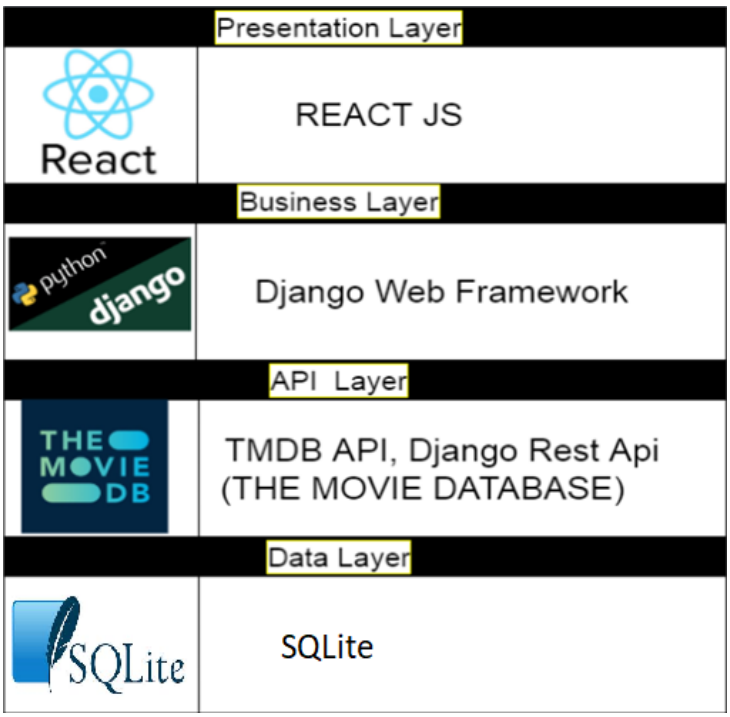
Introduction

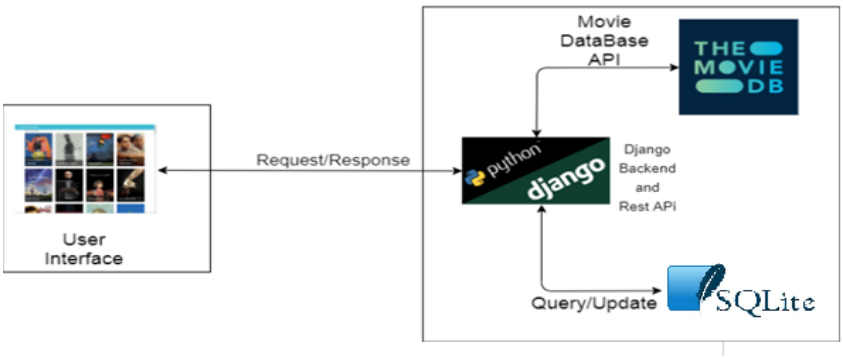
FilmFinder personalizes the user experience by providing movie recommendations based on the movie review history. It also provides the unique ability to the users to add criteria like genre and Wishlist to be added for receiving the movie recommendations. There are many movies in the market, and one of the main pain points for the users is to keep track of the list of movies that they have watched. FilmFinder platform allows users to mark the movies as seen, this will help the users in creating a personal collection and it will also help in providing better recommendations to the users. It is difficult for the users to read all the reviews, so we solve the problem by adding an option to like a review. Often the users abandon the website due to its user interface, FilmFinder is addressing this issue by building a friendly and easy to use user interface.

The following are some of the challenges that the customers face which have been solved by our platform as novelty.

- **Bias created based on Gender** – The existing platforms do not consider the bias that can be created due to the gender. It has been observed that males favor movies which are more male oriented and dislike the counterpart. This bias creates a conflict as the average rating does not consider this factor and the female users are not able to receive relevant review. (Stegner, 2018). The 5Star film finder offers a provision to the users to filter the reviews based on the gender using a dropdown menu.
- **Bias created by fanboys** – It has been often observed that the fanboys are the first ones to watch a movie also, the first ones to provide reviews. They are often inclined to provide a high rating which may not be useful for a general user. Its only after few weeks that unbiased reviews are posted. The existing platforms do not prove the ability to the users to filter out such biased reviews. (Reynolds, 2017). The 5Star film finder offers a provision to the users to filter the reviews based on the calendar dates to overcome this challenge.
- **Ability to like a review** – Currently there is no visibility to the logic behind selecting the most relevant reviews out of the entire lot in the existing platforms. The 5Star film finder offers a provision to the users to trust a review by looking at the number of likes that the review has.
- **Unsatisfactory user Experience** – The existing user interfaces are filled with many features in a small space, most of which are hard to see, and the users have to put a lot of efforts in understanding the information that the website is trying to show. The 5Star film finder addresses this challenge by offering a clean and a light user interface
- **Trusted review Scores** – 5Star film finder has added an additional score for the movie which we get from a trusted source in the form of TMDb website in addition to the user ratings which will be calculated using the user reviews posted in our platform.
- **In-app Notification** - %Start film finder offers higher degree of engagement as users can share movies and follow other users. Their activities can be showcased using the notifications menu. This feature ensures that users don't have to be dependent on other third-party applications to interact with their friends of people the they are interested in following.

System Architecture





Layer Breakdown and Technology Stack

Presentation Layer

In this layer, Browser will display information from Business layer by rendering CSS, React and JavaScript. We will be using react Semantic UI framework to build webpages so that we can reuse CSS stylesheets and leverage Semantic UI themes. When the user interacts with the system through the browser, the presentation layer communicates with the Business layer over the internet to receive results and present to the user.

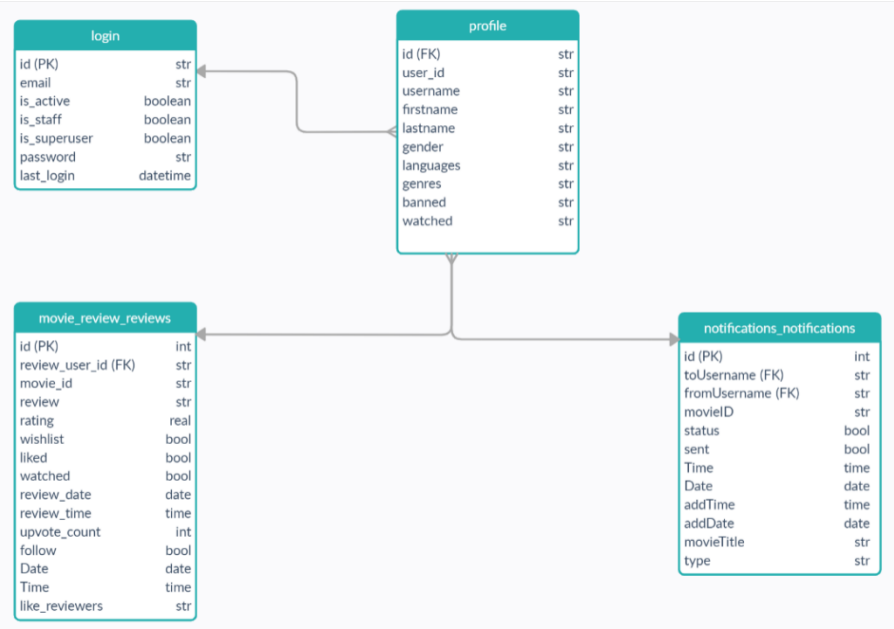
Business Layer

Logic for user requests is implemented in this layer and runs the required functionalities on a server connected to the internet. This includes handling communications between EXTERNAL API, data layer and presentation layer. The server will be running using Django and language used is Python. Some python libraries which we are going to use include requests for Handling HTTP

requests/responses and SciKit-learn for building the recommendation engine. The external API that is used in this project is TMDB. The MovieDB API is used to populate the website with movie details. This API Consists of large data of Movies and used for Movie Recommend function in the Back end. More specifically this API Consists of Movie names, Actor names/images data which is fetched and used for recommendation system.

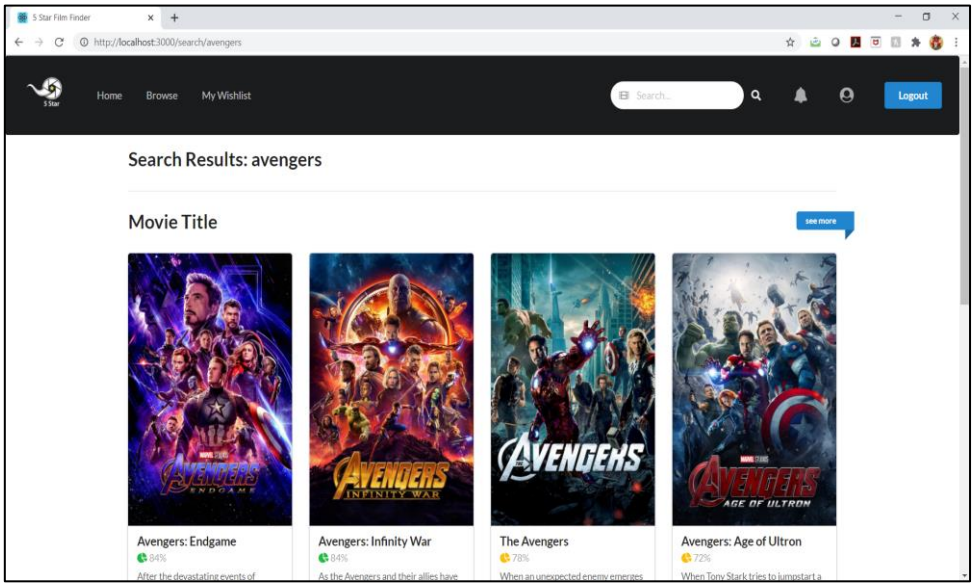
Data Layer

The data layer stores all the user data, reviews and movies and updates. The data layer runs on the commands from business layer for updates to be made or returns results for the queries. In this layer, we used SQLite and the reason for choosing this implementation is support provided for SQL in Django.

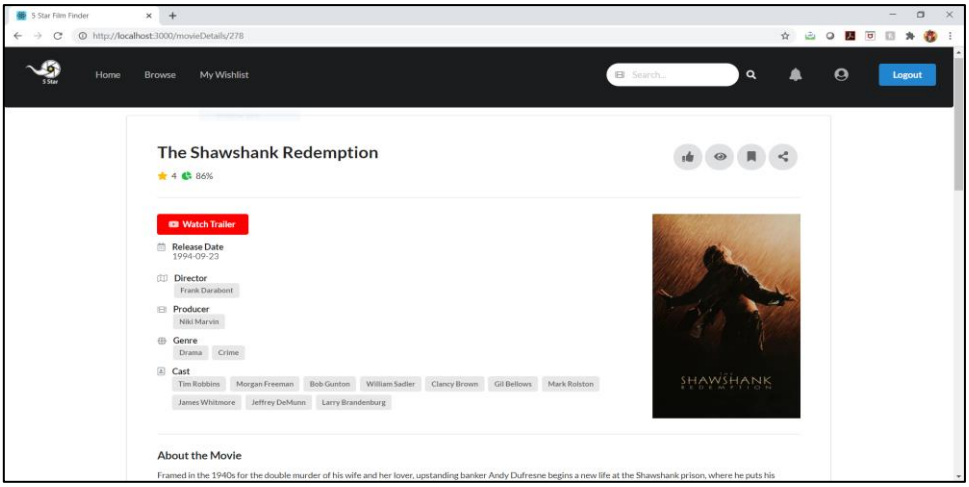


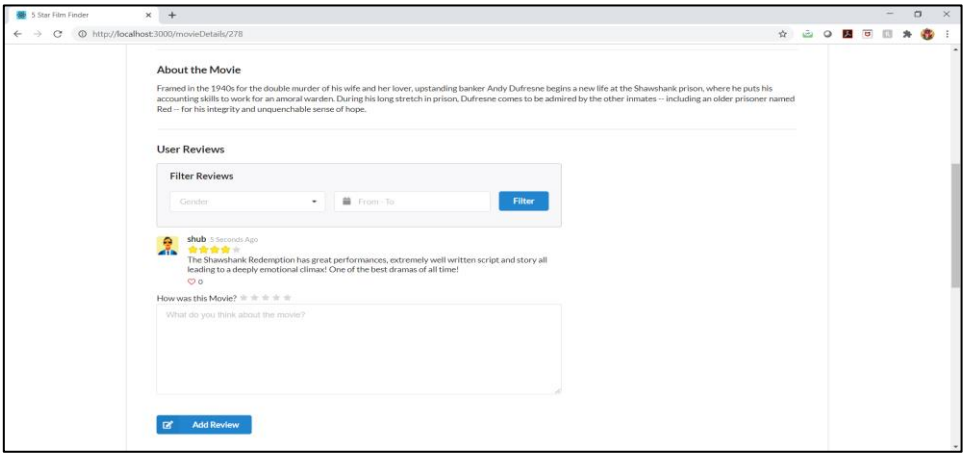
Product Functionality

Project Objective	Film finders must be able to search for movies they are interested by keywords that match to the movie name, description, or genre, with listed results showing matching movie names, and their latest average rating.
Functionality	The user enters the text in the search area located on the Navigation bar. After typing the text, the potential search results will be recommended below the search area to reduce the typing effort. Once the user has selected the movies or finished typing, the results will be generated based on Movie Name, Description or Genre with their latest average rating.

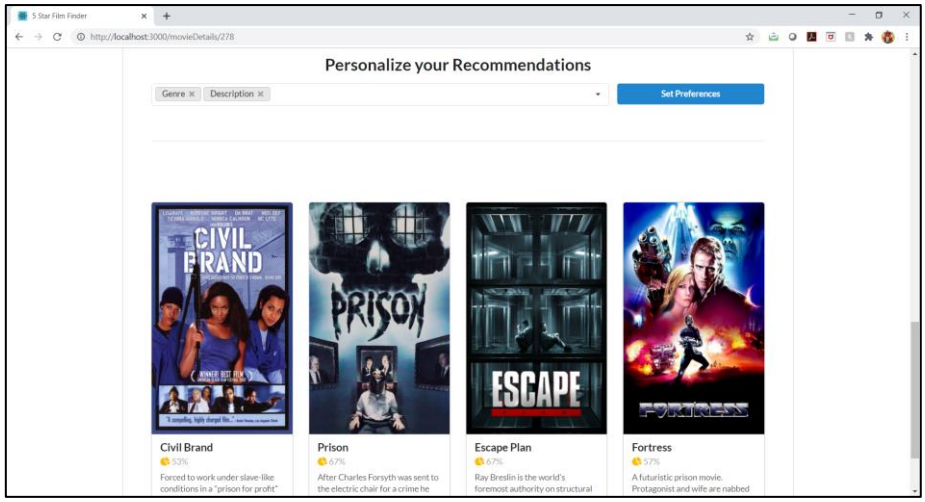


Project Objective	A film finder must be able to view the full details of any movie they come across on the platform, including the movie name, description, genre, cast, director, latest average rating, and all associated reviews.
Functionality	Now it is time for the user to dig a little deeper into our website. The first thing the user sees as soon as they open the website is the homepage. The user will be signed in as guest by default and users can hover over any movie on the homepage and click the view details button to see the full details of a movie. The movie details are perceived differently by a guest and a FilmFinder who has signed up. • Guest User: On the movie details page a guest can view the full details of the movie like movie name, genre, cast, director, average rating, and reviews associated with that movie. • Film Finder: User who has signed up will be able to access everything a guest a user can access. Apart from that, a Film Finder can add their own reviews and upvote other reviews from different users.

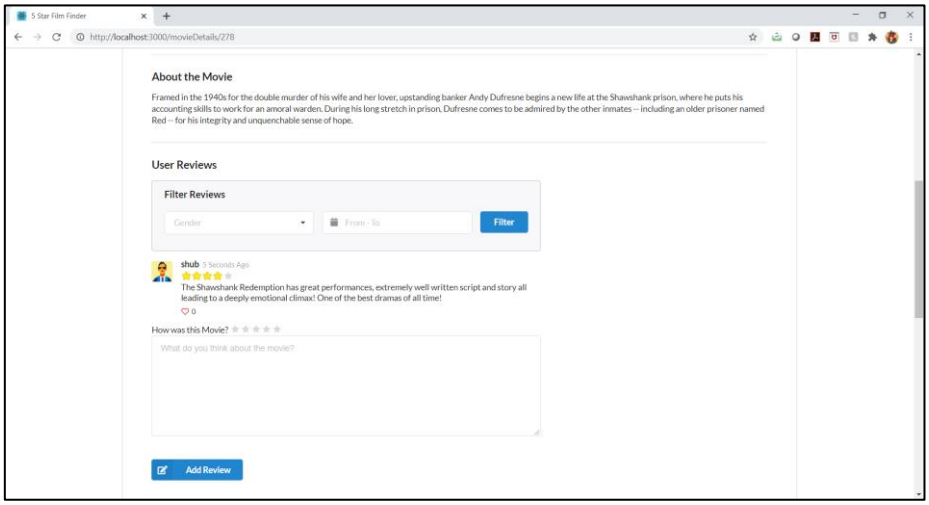




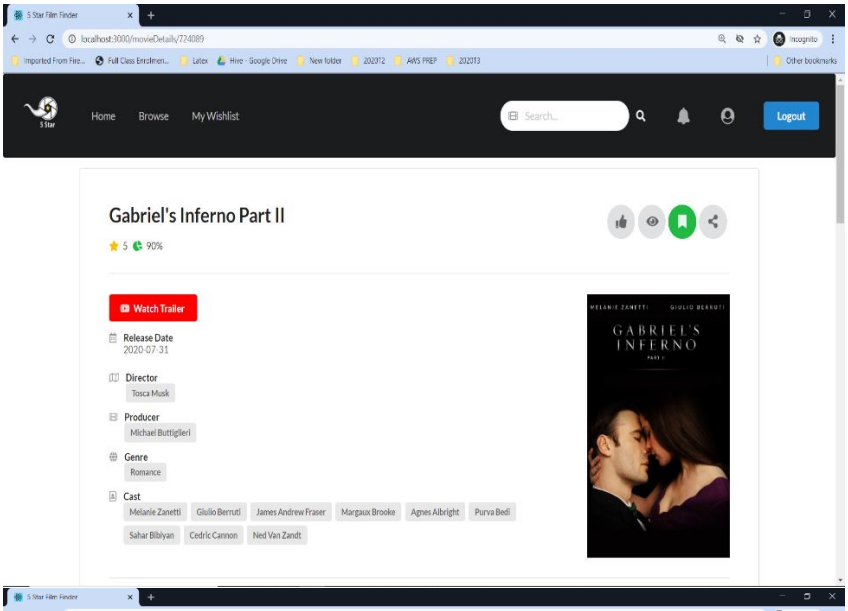
Project Objective	The full details page for a given movie must also show recommendations for other movies that are "similar", where such recommendations must be based on at least review history, and a selection of multiple attributes that a user can select (eg: genre and/or director).
Functionality	In the view details page, a film finder will have the capability to personalize the preferences for the recommendations of the movie they watched. A FilmFinder will be able to select from multiple attributes like genre, director, Wishlist, watch list etc. based on which the recommendations will be displayed.

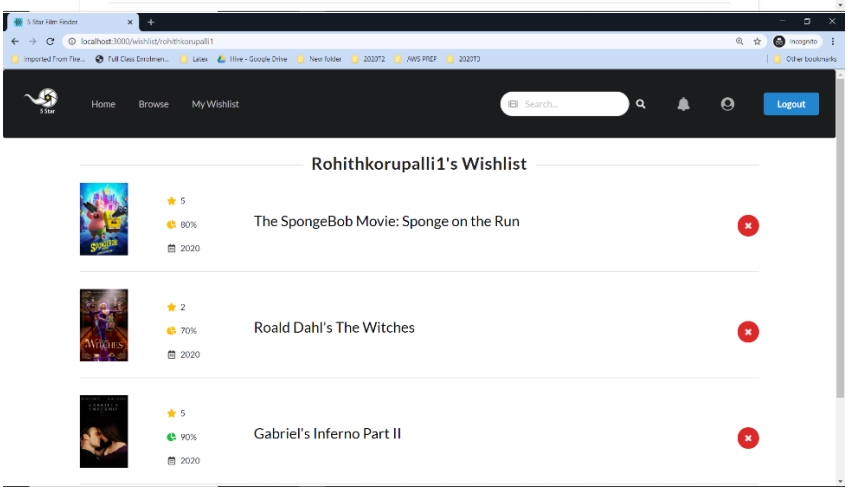


Project Objective	A film finder must also be able to leave a review for any given movie they browse to on the platform, where such a review includes text as well as a rating from 0 to 5.
Functionality	In the view details page, if a film finder has signed up, they will have the capability to leave a review for any movie they select. Users can add a text review in the text box along with the Star Rating “0” being the lowest and “5” being the highest. Users can also give likes to other reviews from different users. A FilmFinder can filter the reviews based on gender and date to get a holistic view of all the reviews for a particular movie.

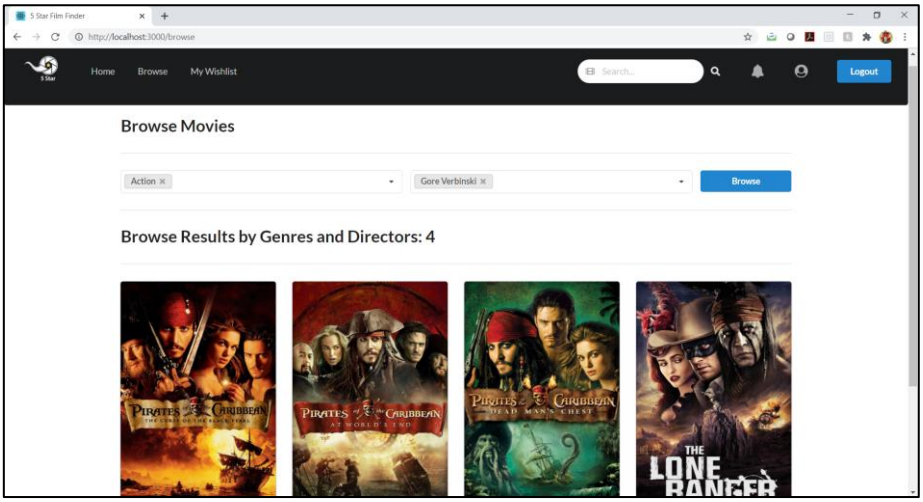


Project Objective	Film finders must be able to add any movie they come across on the platform to a Wishlist of movies they would like to watch and must also be able to remove movies from this Wishlist. Film finders must also be able to view the Wishlist of any other film finder that has left a review for a given movie.
Functionality	In the view details page, users can select the bookmark button on the top right side of the page to add a particular movie to their Wishlist. The users can then view their Wishlist in the My Wishlist tab on the Navigation bar. In the My Wishlist tab the users can also remove the movies they no longer want in the Wishlist.

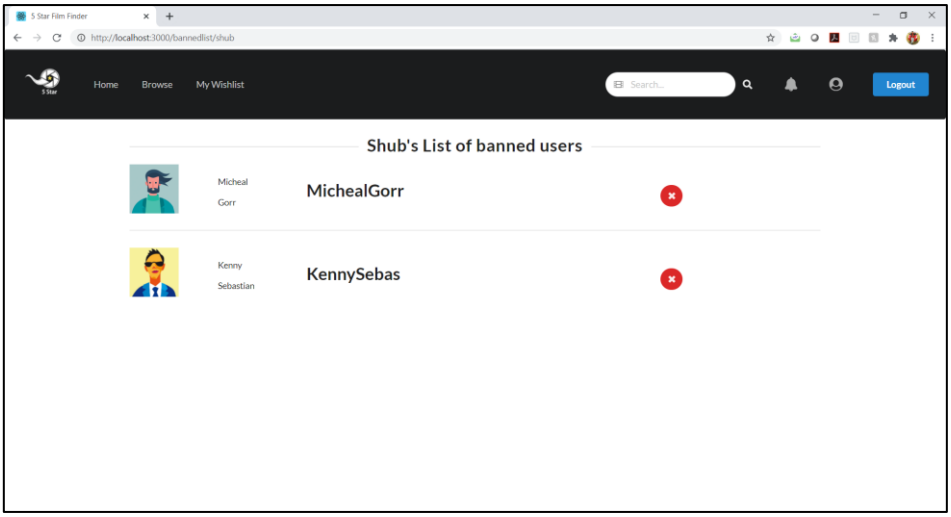




Project Objective	The platform must also allow film finders to browse movies by director or genre, with movies sorted from most popular to least popular based on the movie average rating, and movies with an equal rating being sorted alphabetically (movies with no rating would be treated as having an average rating of 0).
Functionality	The browse tab is located on the Navigation Bar next to the Home tab. In the Browse tab, the users can select multiple genres and directors they wish to find movies associated with. The results will be displayed in alphabetically if they have an equal rating otherwise it will be sorted from least popular to most popular.



Project Objective	The platform must also allow film finders to add any review writer(s) they wish to their "banned" list, meaning that the reviews written by the banned review writer(s) will not be seen or influence the average rating for the FilmFinder who banned the review writer(s).
Functionality	In the Movie Details page, a FilmFinder can hover over a user who has given a review and ban that user. The banned user will no longer be displayed in the feed and the overall ratings will not be affected for that movie. To remove the user from the banned list, a FilmFinder simply needs to go to the profile section on the top right corner of the page and select view Banned List to remove a particular user. In the profile section, FilmFinder can edit their basic information that they used during signing up and view their watchlist and follow list.


5 Star Film Finder
http://localhost:3000/bannedlist/shub
Home
Browse
My Wishlist
Search...
Logout
Shub's List of banned users
Micheal Gorr
MichealGorr
Kenny Sebastian
KennySebas

Third-Party Interface

1. **Django**
   This is a web framework providing REST API. It works at our Backend and useful to read, write to the SQLite Database. This library is offered on the standard open source licence which suits our project needs. Its official documentation can be found at https://docs.djangoproject.com/en/3.1

2. **The Movie Database (TMDb) API**
   The TMDb API is used to receive movie details. This reduced our effort in storing all the movies in the database. API works based on the details sent from Django.
   The following API requests are used in our project:
   - Movie Details: When user clicks on movie to view its details. Movie ID is received at backend and request is formed as below. https://api.themoviedb.org/3/movie/{movie_id}?api_key=<<api_key>>&language=en-US .
   - Discover Movie: To browse movies based on Genre and Directors. Discover API provides option to send Genre ID and Director ID. https://api.themoviedb.org/3/discover/movie?api_key=<<api_key>>&language=en-US&sort_by=popularity.desc&include_adult=false&include_video=false&page=1 .

3. **Local MemCache**
   To reduce the loading speed at the frontend. We are storing the results of Search and Browse in Local Memcache. This is an open-source library and its documentation is available on https://docs.djangoproject.com/en/3.1/topics/cache/ .

4. **Request**
   Requests allows you to send HTTP/1.1 requests extremely easily. There is no need to manually add query strings to your URLs or to form-encode your POST data. It's an open-source library and its documentation can be found at https://requests.readthedocs.io/en/master/ .

5. **Beautiful Soup**
   This library is used to extract text content from HTML pages. BeautifulSoup is a python library easy to install and use. We have used this library to scrape the data from the tbdb.com website for information which was not available through their API. It is an open-source library whose documentation can be found at https://www.crummy.com/software/BeautifulSoup/bs4/doc/ .

6. **secure-smtplib**
   This library is used to send an email notification to a user who is trying to sign-up. Documentation is available on https://docs.python.org/3/library/smtplib.html .

7. **React**
   React facilitates the overall process of writing components and it boosts productivity and facilitates further maintenance. It comes with a helpful developer toolset and it is backed by a strong community. React is released as MIT licensed which is suitable for our project as it does not impose any restrictions that will affect our product's ability to function. Its official documentation can be found at https://reactjs.org/docs/getting-started.html .

8. **Semantic UI React**
   Semantic is a development framework that helps create beautiful, responsive layouts using human-friendly HTML. Semantic comes equipped with an intuitive inheritance system and high-level theming variables that which makes it easy to adopt its elements in our web application. Semantic UI react is released as MIT licensed which is suitable for our project as it does not impose any restrictions that will affect our product's ability to function. Its official documentation can be found at https://react.semantic-ui.com/ .

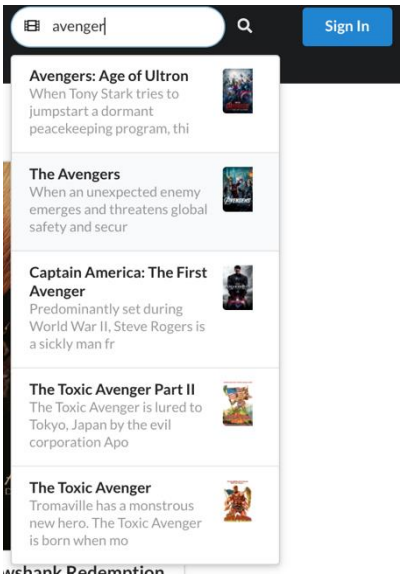
9. **React Router Dom**
   This library guarantees better user experience and higher app performance while navigating to different pages of the application. It is an open-source library which makes it ideal for our project. Its documentation can be found at https://www.npmjs.com/package/react-router-dom?activeTab=versions .

10. **React Select**
    React Select offers a flexible and beautiful Select Input control for ReactJS with multiselect, autocomplete, async and creatable support. This library was widely used in our application for taking inputs where more than one value was admissible. React Select is MIT licensed which is suitable for our project as it does not impose any restrictions that will affect our product's ability to function. Its official documentation can be found at https://github.com/jedwatson/react-select .

11. **Semantic UI Calendar React**
    This is an open-source library which has been created on top of the semantic UI react library. This library offers an input in the calendar format. Since it is built on top of an existing library which we are using, it made the imports lighter as the dependent CSS from the semantic UI react library were already loaded in our application. Its official documentation is available at https://www.npmjs.com/package/semantic-ui-calendar-react .

Implementation Challenges

1. **We faced slow rendering speed in search dropdown as the code had to call API and perform search on every character which was entered**.
   Search results change as user is typing characters in the search bar. Improved the performance of searching movies by movie name. The API is not being invoked on every search call instead it is pulling all the movies information when the backend server has started and storing it at front end *movieData.js*. Improved the visual appearance of the search dropdown by showing movie title, description, and movie poster.



2. **Infinite loop was getting created while changing the state in react hooks.**
   Use State Hooks variables were not unmounting, so we implemented classes and updated states of the variable using thisState().

3. **Display alignment for search results was not perfect due to auto responsive grid**
   We used auto responsive 4 grid architecture for displaying our results of movie tiles. Additional conditions had to be used in case the results were not in multiple of 4.

4. **Movie tiles display alignment had issue as the character lengths are different for each character and hence a restriction on string length was not sufficient for fitting all the text in the same width.**
   Added overflow property in css for movie description in movie tile.

5. **Redirect issue using Doms was not reloading the page.**
   Pages were not getting reloaded if the movie details page was open and we had to view the same page but for a different movie. We used a href redirect function to overcome that challenge as it automatically reloads the page.

6. **As mongoDb was not supported in vLab environment we had to shift from MongoDB to SQLite**
   Initially chose Mongo as our database since it makes backend development easy but later, we found that we cannot install MongoDB on vlab system. So made a shift to SQLite which is a default database.

7. **Connection Between React and Django was getting blocked due to inbuilt security of Django.**
   Need to use Django CORS library for handling requests from browser. We used CORS ALLOW ALL ORIGINS to accept request from Browser. In vLab, Host port number is changing randomly. So, we used CORS library.

8. **Process Data received from API was challenging due to excess of optional elements.**
   API sends response in JSON and we need to process the JSON and send only parameters required to the front end. Few responses have missing keys which are handled in this project.

User Manual

5Star film finder is divided into two main components, the backend that contains the application programming interface in the form of rest services, and the front end which contains the user interface for the users to interact.

Software Requirements

The below mentioned software needs to be installed in the system for the 5Star film finder to function as expected

1. Node.JS
2. Python3
3. SQLite3

Browser Compatibility

The website works well on the latest web browsers like Google Chrome, Mozilla Firefox, etc. However, it works better using latest Google Chrome web browser.

Note: We tested the application in Chromium web browser in vLab where poster image of movies was not displaying on the movie tiles. However, other functionalities work fine on vLab Chromium web browser.

Restrictions on Operating System settings

We make use of smtp server at the time of registration to send our email notifications. We observed that port 25 is blocked by default in many systems which may cause the feature to not work. In vLab the smtp port is blocked and it requires administrative permissions to unblock it.

Step to install Node.JS

A. Installation steps for vLab

Node.js is already installed in vLab. In case its missing please contact vLab administrator for installation.

```
z5232648@vx3:/tmp_amd/cage/export/cage/3/z5232648$ node --version
v10.21.0
z5232648@vx3:/tmp_amd/cage/export/cage/3/z5232648$ npm --version
5.8.0
```

B. Installation steps for Windows OS

1. In a web browser, navigate to https://nodejs.org/en/download/. Click the **Windows Installer** button to download the latest default version. The Node.js installer includes the NPM package manager.
2. Once the installer finishes downloading, launch it. Open the **downloads** link in your browser and click the file. Or browse to the location where you have saved the file and double-click it to launch.
3. The system will ask if you want to run the software – click **Run**.
4. You will be welcomed to the Node.js Setup Wizard – click **Next**.
5. On the next screen, review the license agreement. Click **Next** if you agree to the terms and install the software.

6. The installer will prompt you for the installation location. Leave the default location unless you have a specific need to install it somewhere else – then click **Next**.
7. The wizard will let you select components to include or remove from the installation. Again, unless you have a specific need, accept the defaults by clicking **Next**.
8. Finally, click the **Install** button to run the installer. When it finishes, click **Finish**. Node.js is installed in your system.

C. Installation steps for Mac OS

1. Go to the Node.js Downloads page https://nodejs.org/en/download/.
2. Download Node.js for macOS by clicking the "MacOS Installer" option
3. Run the downloaded Node.js .pkg Installer
4. Run the installer, including accepting the license, selecting the destination, and authenticating for the install.
5. You are finished! To ensure Node.js has been installed, run node -v in your terminal.

Steps to install Python3

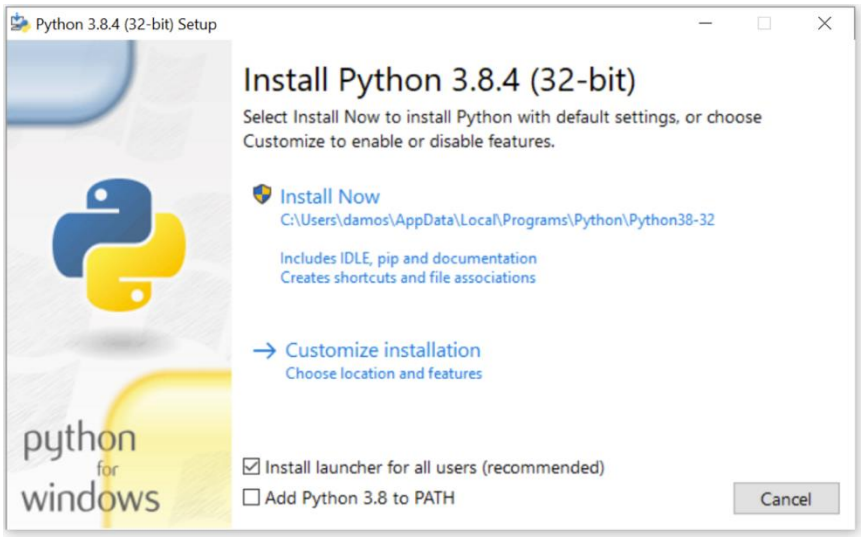
A. Installation steps for vLab

Python3 is already installed in vLab. In case its missing please contact vLab administrator for installation.

```
z5232648@vx3:/tmp_amd/cage/export/cage/3/z5232648$ python3 --version
Python 3.7.3
```

B. Installation steps for Windows OS

1. Open a browser window and navigate to the Python.org Downloads page for Windows.
2. Under the “Python Releases for Windows” heading, click the link for the Latest Python 3 Release - Python 3.x.x.
3. Scroll to the bottom and select either Windows x86-64 executable installer for 64-bit or Windows x86 executable installer for 32-bit.
4. Once you’ve chosen and downloaded an installer, run it by double-clicking on the downloaded file. A dialog box like the one below will appear:



5. The default install path is in the AppData/ directory of the current Windows user.

6. The *Customize installation* button can be used to customize the installation location and which additional features get installed, including pip and IDLE.
7. The *Install launcher for all users (recommended)* checkbox is checked default. This means every user on the machine will have access to the py.exe launcher. You can uncheck this box to restrict Python to the current Windows user.
8. The Add Python 3.x.x to PATH checkbox is unchecked by default.
9. Customize the installation to meet your needs using the options available on the dialog box. Then click Install Now. Python 3.x.x is installed in your system.

C. Installation steps for Mac OS

1. Open a browser window and navigate to the Python.org Downloads page for macOS.
2. Under the "Python Releases for Mac OS X" heading, click the link for the *Latest Python 3 Release - Python 3.x.x*.
3. Scroll to the bottom and click *macOS 64-bit installer* to start the download.
4. When the installer is finished downloading, move on to the next step. Run the installer by double-clicking the downloaded file. You should see the following window:

5. Press Continue a few times until you're asked to agree to the software license agreement. Then click Agree.
6. You'll be shown a window that tells you the install destination and how much space it will take. You most likely don't want to change the default location, so go ahead and click Install to start the installation.
7. When the installer is finished copying files, click Close to close the installer window. Python3 is installed in your system.

Steps to install SQLite 3

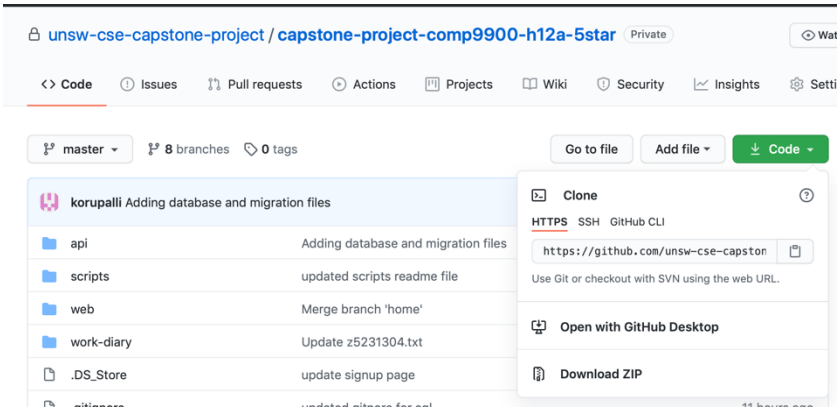
SQLite comes preinstalled with Python3. So, no additional steps are required to install this software.

Building and running 5Star Film Finder

The first step will involve building the application and then start the application. The backend application needs to build and started first followed by the frontend application. Build is a onetime setup which needs to be performed to setup the application.

Download Instructions

1. Download ZIP code from git repository: https://github.com/unsw-cse-capstone-project/capstone-project-comp9900-h12a-5star



2. Save the zip file in your directory folder
3. Go to the directory folder where zip file is saved, unzip/extract the zip file

Build and Run Backend Server

1. Open command prompt in Windows or terminal in Linux/Mac OS.
2. Go to the root directory of the application.
3. Now, cd to the api folder

```
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/api$ pwd
/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/api
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/api$
```

4. We will now upgrade pip and install all the python library dependencies in a python environment by using the below commands. The screenshots are taken from VLAB terminal. Run the below commands in the given order.

   - `python3 -m pip install --upgrade pip`

```
z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$ python3 -m pip install --u
pgrade pip
Defaulting to user installation because normal site-packages is not writeable
Requirement already up-to-date: pip in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (20.2.4)
```

   - `python3 -m pip install --user --upgrade pip`

```
z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$ python3 -m pip install --u
ser --upgrade pip
Requirement already up-to-date: pip in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (20.2.4)
```

   - `python3 -m pip install --user virtualenv`

```
z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$ python3 -m pip install --u
ser virtualenv
Requirement already satisfied: virtualenv in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (20.1.0)
Requirement already satisfied: six<2,>=1.9.0 in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (from virtualenv) (1
.15.0)
Requirement already satisfied: appdirs<2,>=1.4.3 in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (from virtualenv
) (1.4.4)
Requirement already satisfied: filelock<4,>=3.0.0 in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (from virtualen
v) (3.0.12)
Requirement already satisfied: importlib-metadata<3,>=0.12; python_version < "3.8" in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python
3.7/site-packages (from virtualenv) (1.7.0)
Requirement already satisfied: distlib<1,>=0.3.1 in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (from virtualenv
) (0.3.1)
Requirement already satisfied: zipp>=0.5 in /tmp_amd/cage/export/cage/3/z5232648/.local/lib/python3.7/site-packages (from importlib-metadata
<3,>=0.12; python_version < "3.8"->virtualenv) (3.4.0)
z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$
```

- `python3 -m venv env`
- `source env/bin/activate`

```
z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$ source env/bin/activate
(env) z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900/capstone-project-comp9900-h12a-5star-master$
```

- `python3 -m pip install -r requirement.txt`

```
Installing collected packages: soupsieve, beautifulsoup4, six, python-dateutil, bson, asgiref, sqlparse, pytz, Django, django-appconf, zipp,
 importlib-metadata, django-bootstrap4, django-cors-headers, django-multiselectfield, django-select2, djangorestframework, pymongo, djongo,
isort, mccabe, lazy-object-proxy, wrapt, typed-ast, astroid, pylint, pymodm, chardet, idna, urllib3, certifi, requests, tmdbv3api, bs4, secu
re-smtplib, numpy, pandas, joblib, scipy, scikit-learn
  Running setup.py install for bson ... done
  Running setup.py install for djongo ... done
  Running setup.py install for wrapt ... done
  Running setup.py install for bs4 ... done
Successfully installed Django-3.0.5 asgiref-3.2.10 astroid-2.4.2 beautifulsoup4-4.9.3 bs4-0.0.1 bson-0.5.7 certifi-2020.11.8 chardet-3.0.4 d
jango-appconf-1.0.4 django-bootstrap4-2.2.0 django-cors-headers-3.5.0 django-multiselectfield-0.1.12 django-select2-7.4.2 djangorestframewor
k-3.12.1 djongo-1.3.3 idna-2.8 importlib-metadata-1.7.0 isort-4.3.21 joblib-0.17.0 lazy-object-proxy-1.4.3 mccabe-0.6.1 numpy-1.19.4 pandas-
1.1.4 pylint-2.3.1 pymodm-0.4.3 pymongo-3.7.2 python-dateutil-2.8.1 pytz-2020.4 requests-2.22.0 scikit-learn-0.21.2 scipy-1.5.4 secure-smtpl
ib-0.1.1 six-1.15.0 soupsieve-2.0.1 sqlparse-0.2.4 tmdbv3api-1.6.2 typed-ast-1.4.1 urllib3-1.25.11 wrapt-1.12.1 zipp-3.4.0
(env) z5232648@vx4:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/api$
```

5. Now, in the same terminal run this command to run the backend server
   - `python3 manage.py runserver`

```
(env) z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/api$ python3 manage
.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified some issues:

WARNINGS:
movie_review.reviews.review_date: (fields.W161) Fixed default value provided.
        HINT: It seems you set a fixed date / time / datetime value as default for this field. This may not be what you want. If you want to
 have the current date as default, use `django.utils.timezone.now`
movie_review.reviews.review_time: (fields.W161) Fixed default value provided.
        HINT: It seems you set a fixed date / time / datetime value as default for this field. This may not be what you want. If you want to
 have the current date as default, use `django.utils.timezone.now`

System check identified 2 issues (0 silenced).
November 16, 2020 - 02:37:09
Django version 3.0.5, using settings 'Main.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

6. The backend server is up and running. The next step to build and run frontend server while the backend server is still running.
7. Make sure that the backend server is running on port 8000.

Build and Run Frontend Server

1. Open command prompt in Windows or terminal in Linux/Mac OS.
2. Go to the root directory of the application.
3. Now, cd to the web folder

```
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/web$ pwd
/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/web
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/web$
```

4. Now install the Node.js dependencies. Run the below command.

- npm install

```
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_3/capstone-project-comp9900-h12a-5star-master/web$ npm install
npm WARN npm npm does not support Node.js v10.21.0
npm WARN npm You should probably upgrade to a newer version of node as we
npm WARN npm can't make any promises that npm will work with this version.
npm WARN npm Supported releases of Node.js are the latest release of 4, 6, 7, 8, 9.
npm WARN npm You can find the latest version at https://nodejs.org/
[                    ] \ extract:set-value: sill extract set-value@2.0.1 extracted to /tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900
```

5. Run the frontend server.
   - npm start

```
z5232648@vx5:/tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/web$ npm start
npm WARN npm npm does not support Node.js v10.21.0
npm WARN npm You should probably upgrade to a newer version of node as we
npm WARN npm can't make any promises that npm will work with this version.
npm WARN npm Supported releases of Node.js are the latest release of 4, 6, 7, 8, 9.
npm WARN npm You can find the latest version at https://nodejs.org/

> filmfinder@0.1.0 start /tmp_amd/cage/export/cage/3/z5232648/Documents/comp9900_2/capstone-project-comp9900-h12a-5star-master/web
> react-scripts start
```

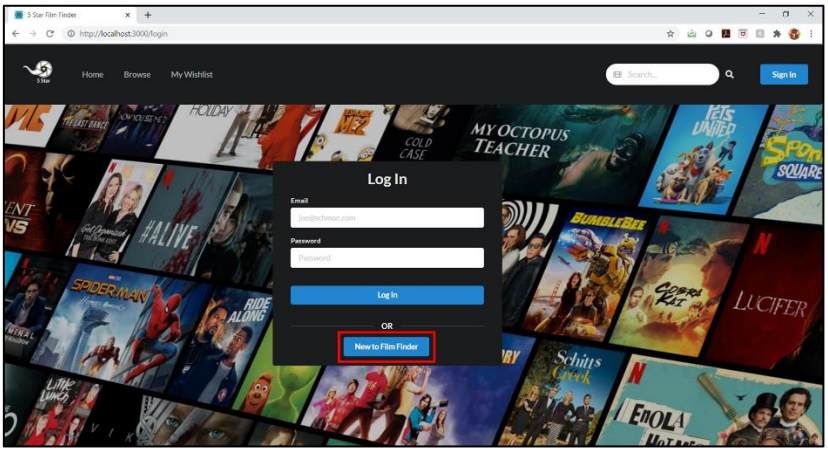
6. Frontend server is up and running.

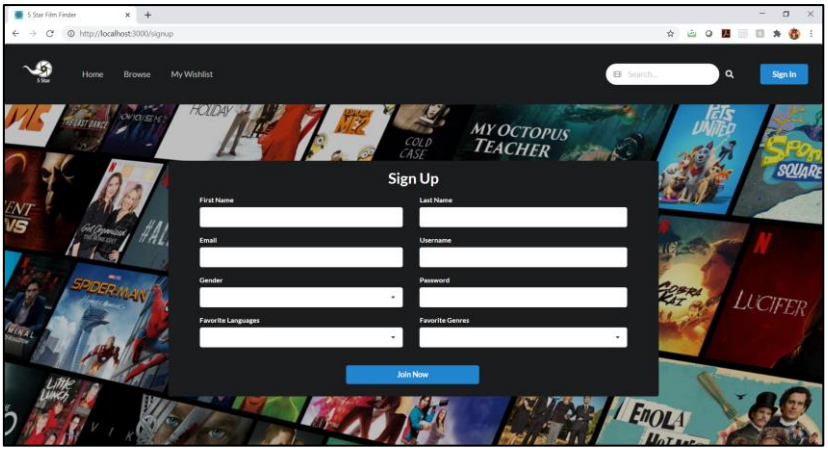
User Guide

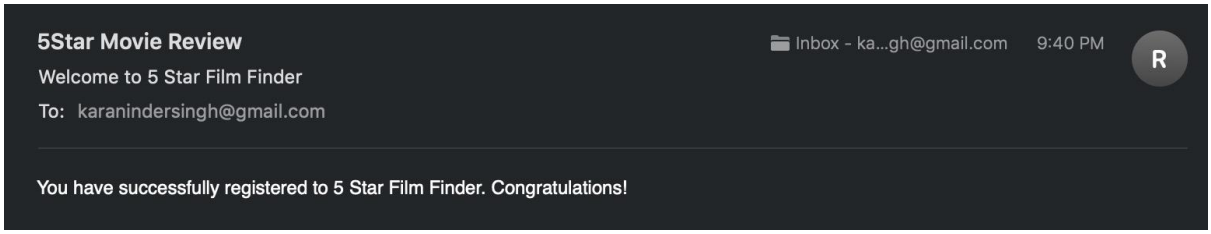
1. Register as a new user

A FilmFinder wishing to sign up on the website can go to the top right signup button to access premium features of the website. If the user is new to the website, they can click the button “New to FilmFinder” to sign up and become a member. The user can sign up with details like Name, Email, Username, Password, Favourite Genres, Favourite Languages etc. A FilmFinder will be identified by their username and email address which will be unique. Once a FilmFinder finishes the signup process, click the “Join Now ” button and an Email will be sent to the registered email ID to notify the user about the creation of the account.

Route: Sign In -> New to film finder



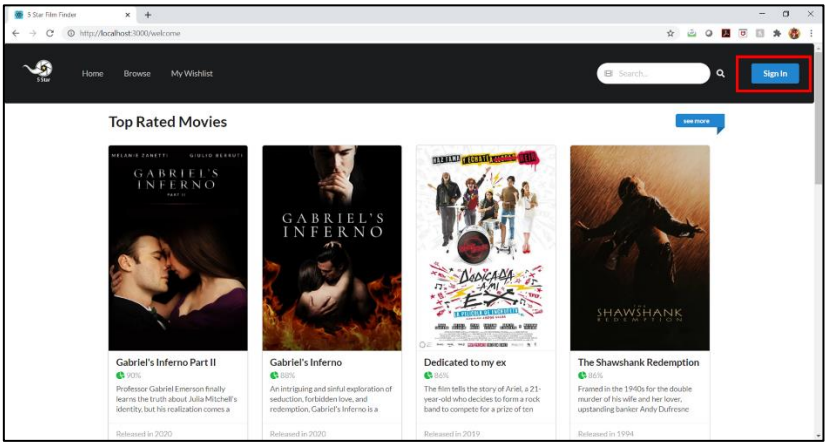


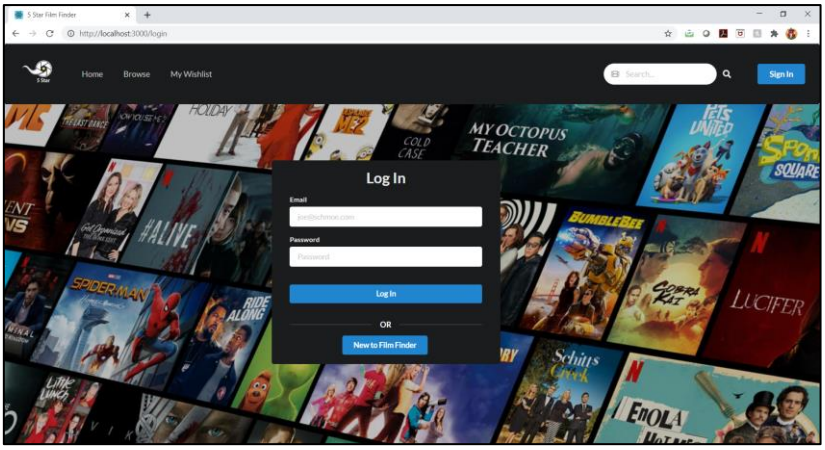


2. Sign In

A FilmFinder is redirected to the Login Page once the signup process is completed. Existing users can just click the top right “Sign In” button. User enters the login details i.e., email and password that they used to register and click “Log In”. After successful Validation, User lands on the Home Page of the Website.

Route: Sign In -> Enter Details -> Log In

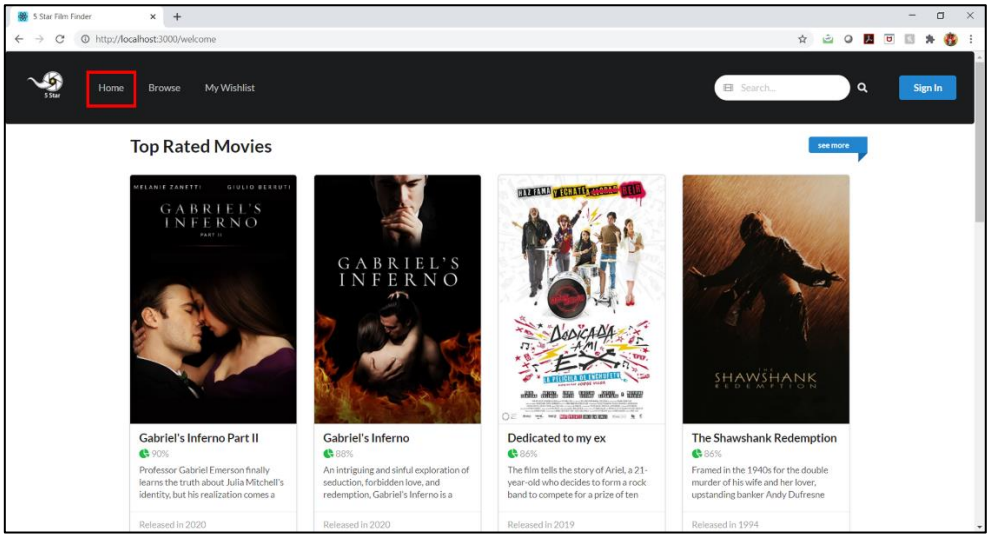


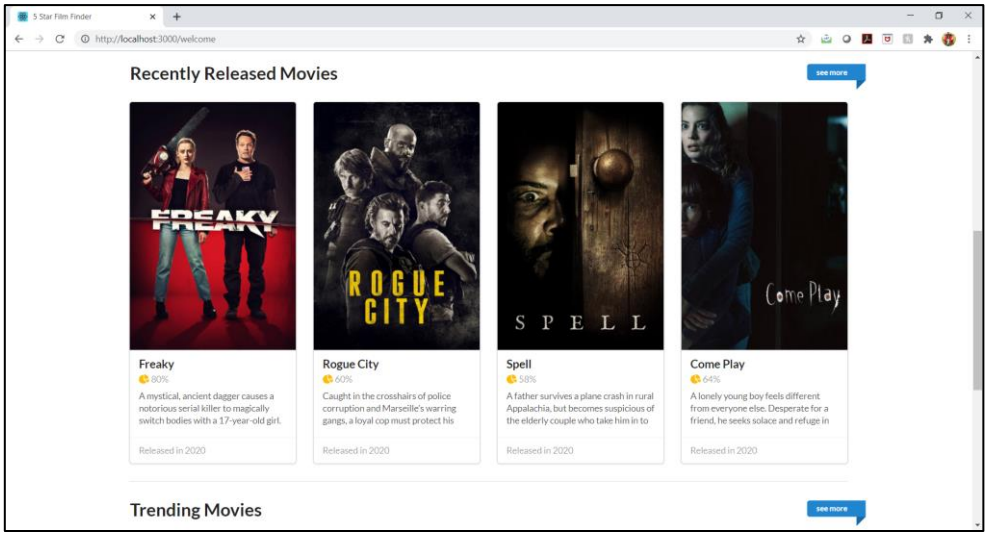


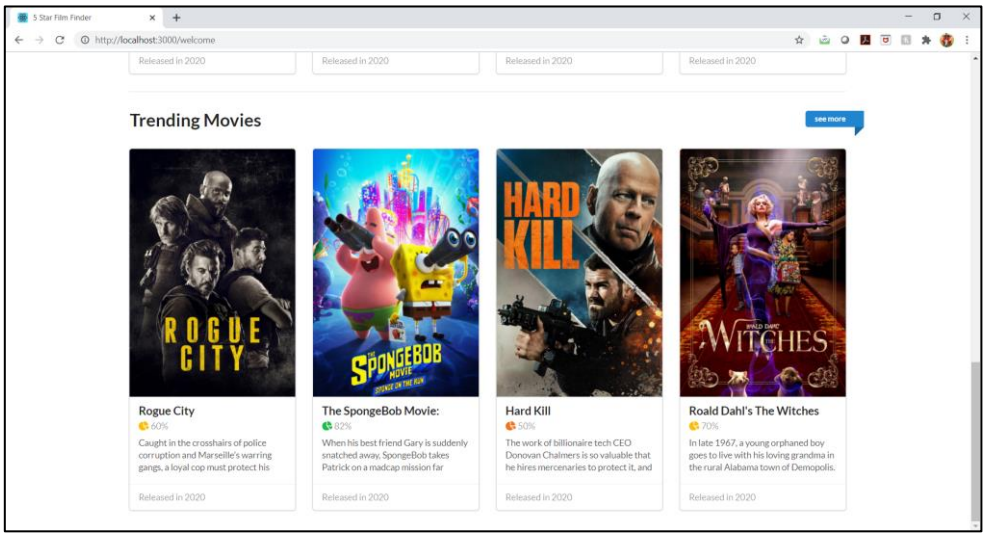
3. View top-rated, recently released, and trending movies

On the Homepage, User can have a gist of logically sorted movie tiles like top rated movies, Trending Movies and Recently added movies. The Movie Tiles for each movie include Movie Name, brief description, Movie Score and release date. Also, users can hop into different functionalities like view their Wishlist, browse movies, profile page etc. These functionalities will be discussed further in detail.

Route: Home



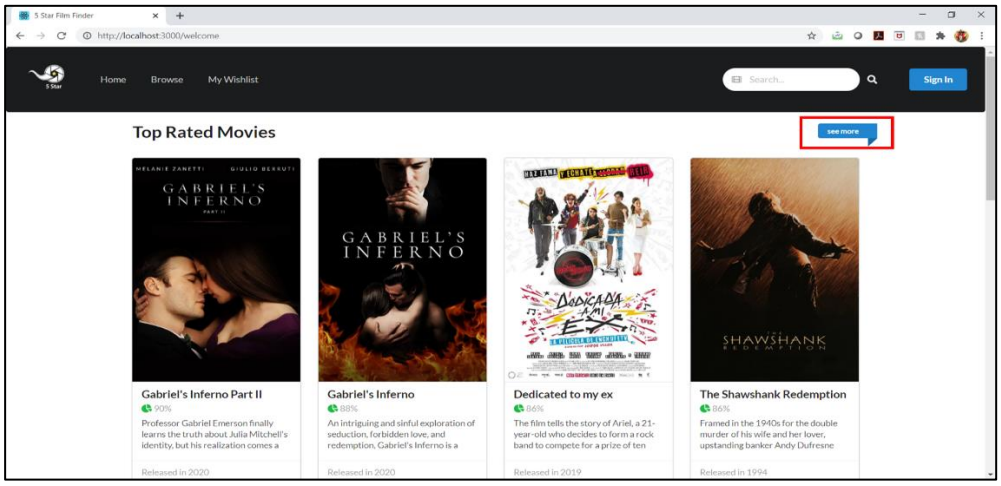


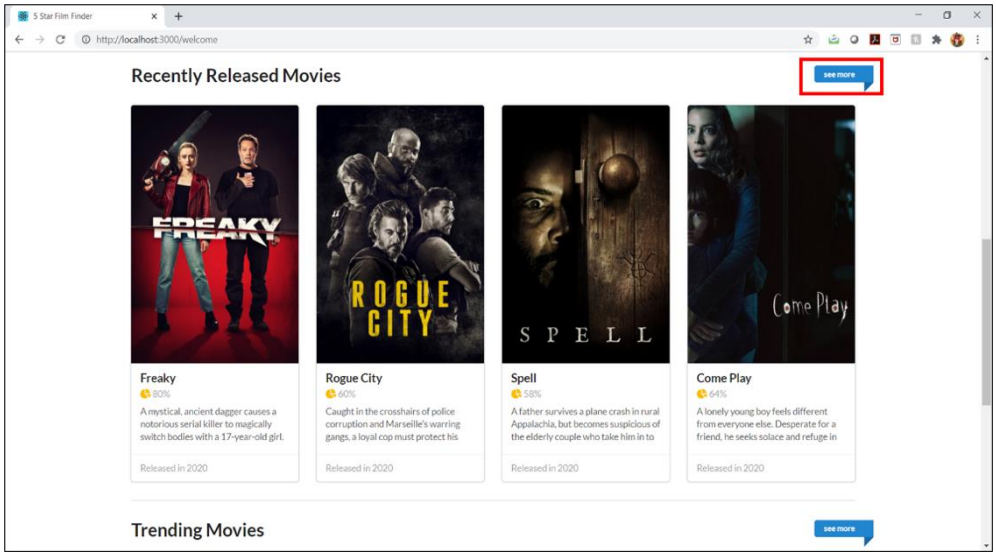


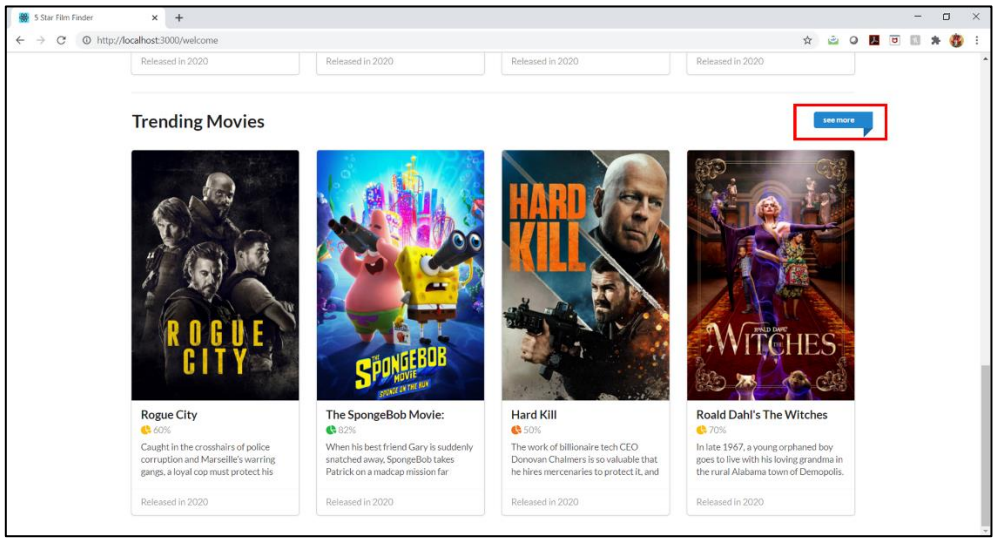
4. View more top-rated, recently released, and trending movies

For each section of homepage, if users wish to view more of those movies, they can click the *see more* button which is located on the top right corner of each section to view the list of all the movies available in that section.

Route: Home -> see more



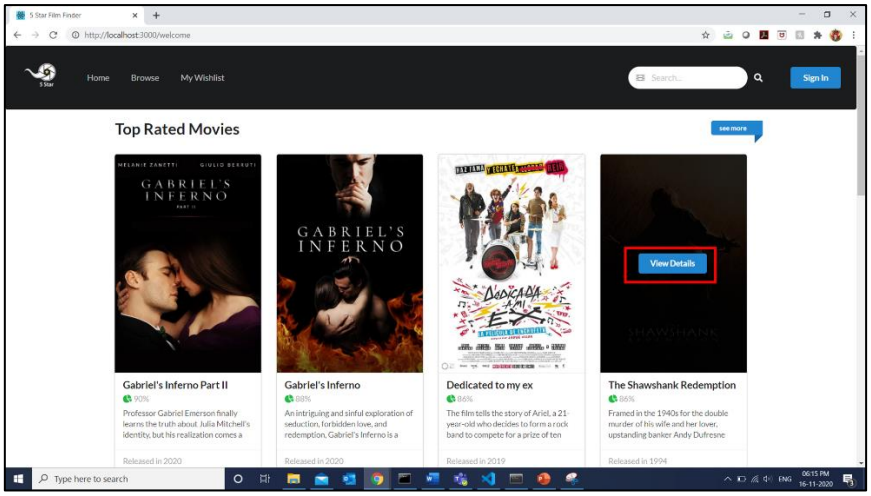


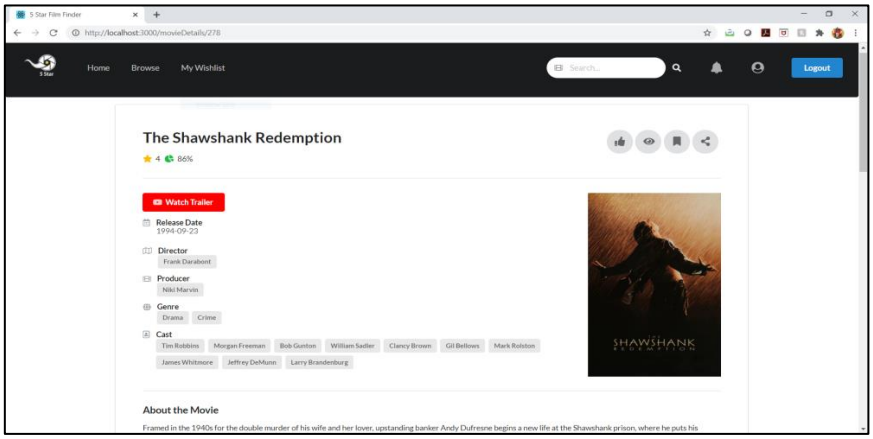


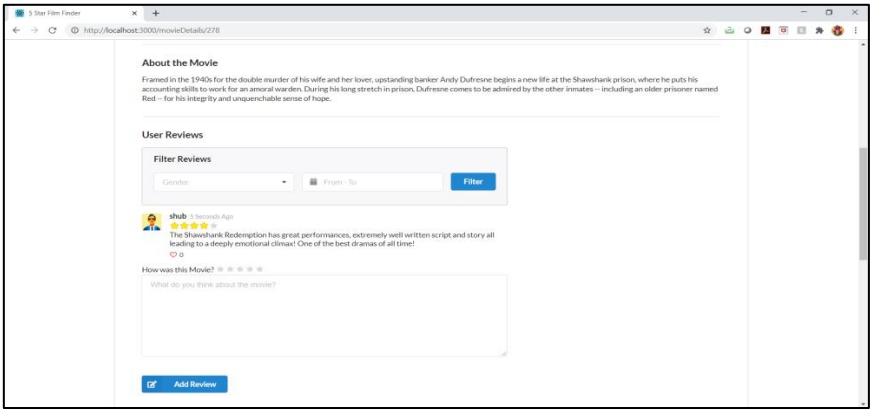
5. View movie details

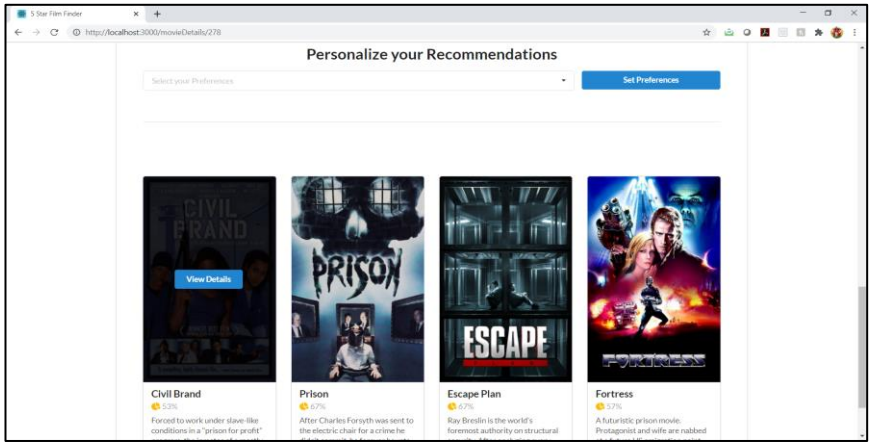
A FilmFinder can hover over any movie tile on the homepage to see the detailed information about that movie by clicking the *view details* button. The user is then redirected to a *movie details* page where a FilmFinder can see the details about the movie like trailer, cast, ratings, reviews, description, genre etc. Users can also add, and view reviews given by other users on the same page (see 14. Manage Reviews: Add, like and rate). Apart from that, a user can also personalize their recommendations (see 16. Manage Recommendations).

Route: Go to a Movie tile -> View Details


Home
Browse
My Wishlist
Sign In
Top Rated Movies
see more
View Details
Gabriel's Inferno Part II
Gabriel's Inferno
Dedicated to my ex
The Shawshank Redemption

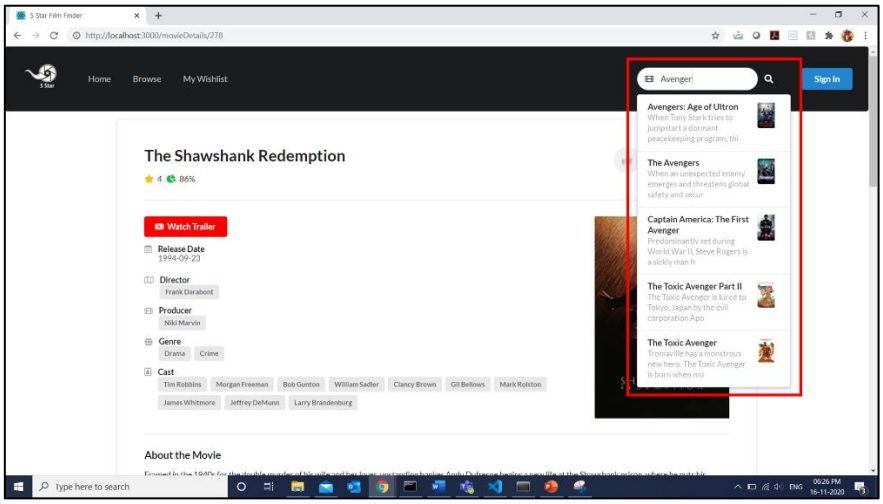

Home
Browse
My Wishlist
Logout
The Shawshank Redemption
Watch Trailer
Release Date
1994-09-23
Director
Frank Darabont
Producer
Niki Marvin
Genre
Drama
Crime
Cast
About the Movie


About the Movie
User Reviews
Filter Reviews
Filter
shub
The Shawshank Redemption has great performances, extremely well written script and story all leading to a deeply emotional climax! One of the best dramas of all time!
How was this Movie?
What do you think about the movie?
Add Review


Personalize your Recommendations
Set Preferences
View Details
Civil Brand
Prison
Escape Plan
Fortress

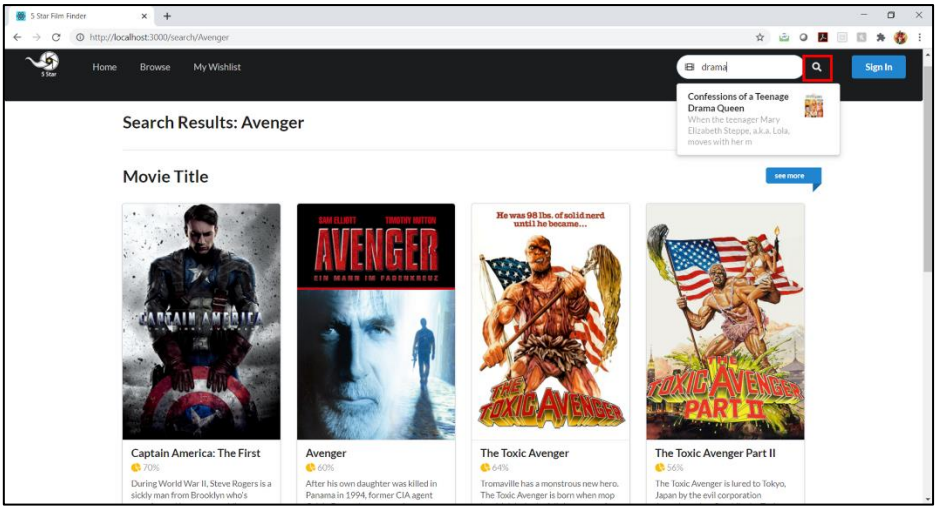
6. Search for a movie

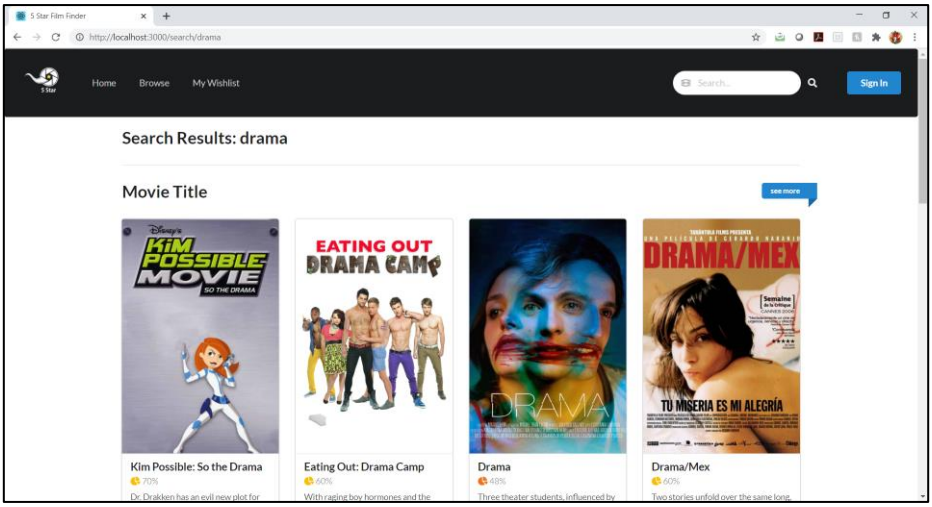
On the Navigation bar, there is a search area where a user can enter a text or the movie they would like to search on our website. The search results will be on the fly and it will keep changing while the user is typing to save extra efforts. After selecting the movie that they find in the recommended movies, User then clicks the search icon next to the search area to see the results. The results page will display all the movies with that text contained in it.

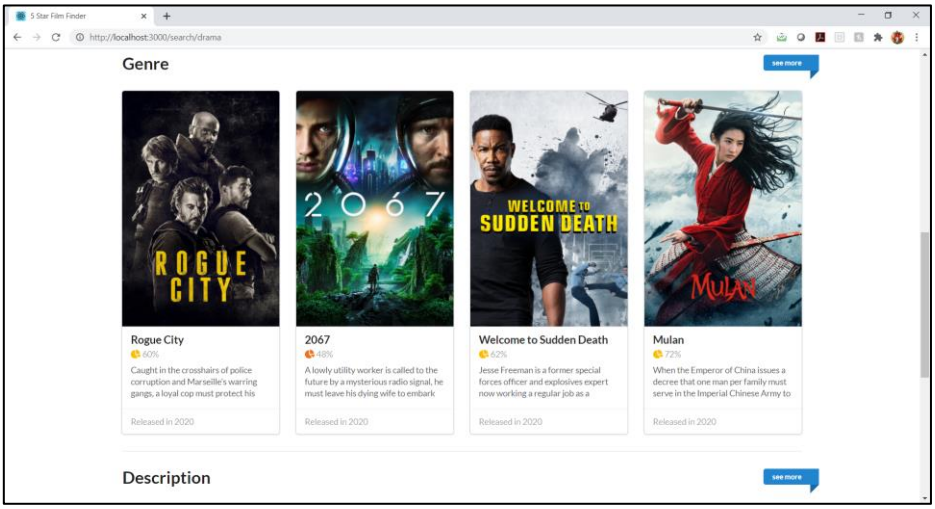


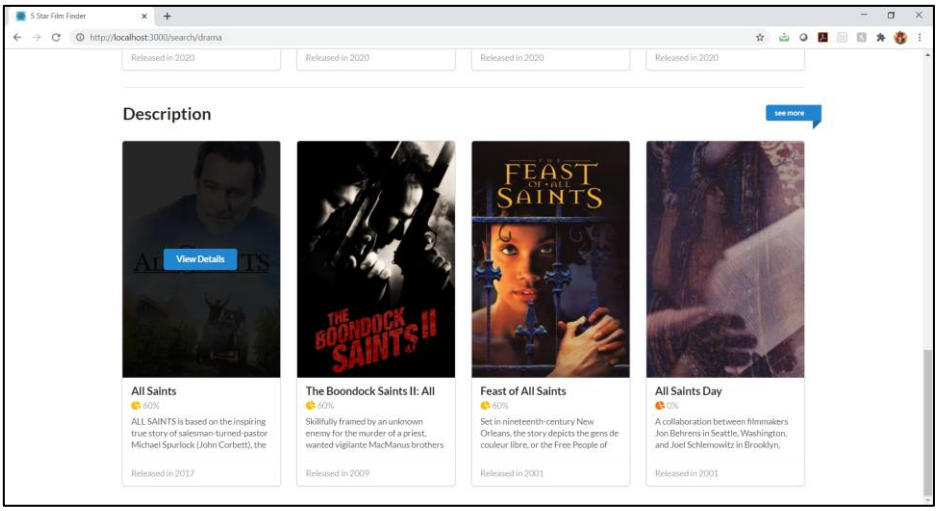
7. Search for movies

After enhancing the capability of the search feature, the FilmFinder can search range of movies based on description and genre of the movie. A FilmFinder types a genre they like, and the search results will generate range of movies based on that genre.

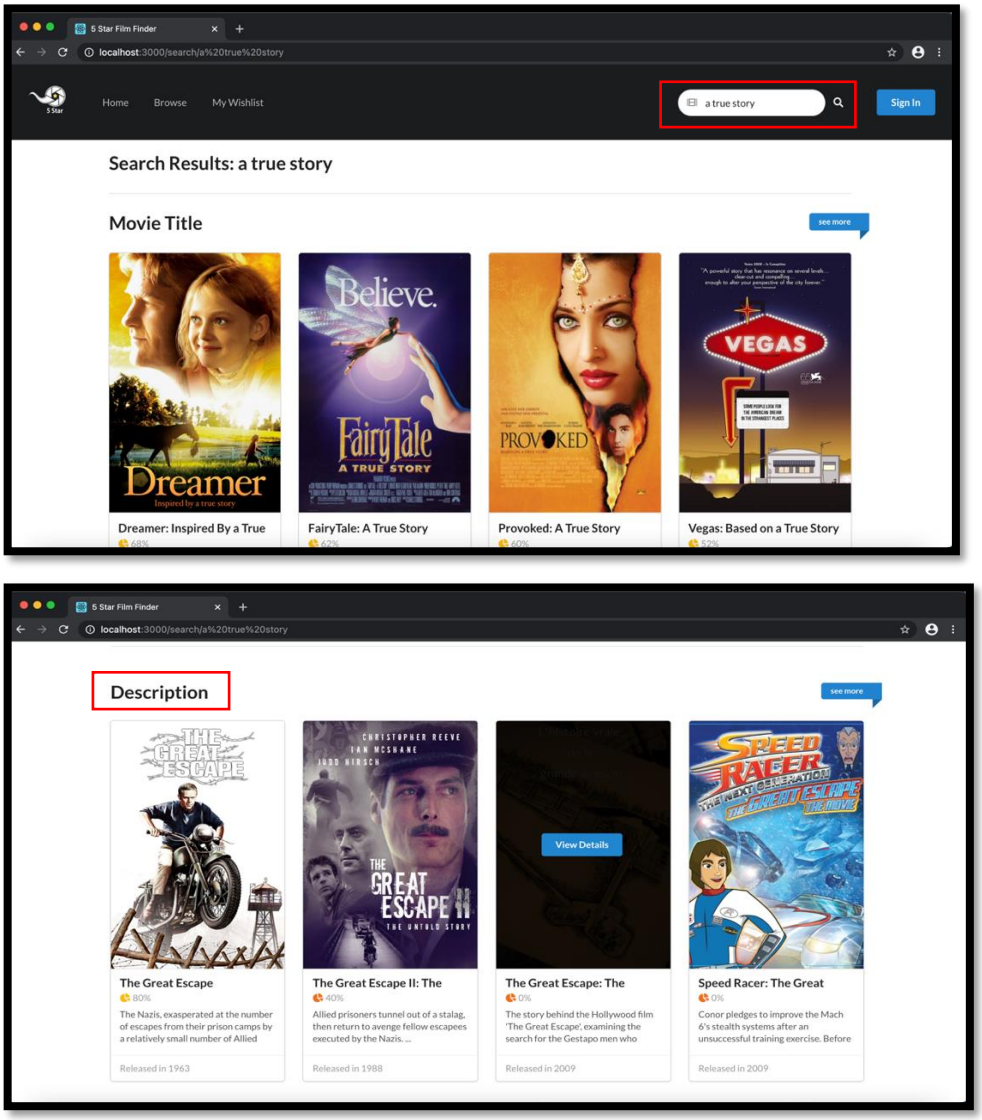








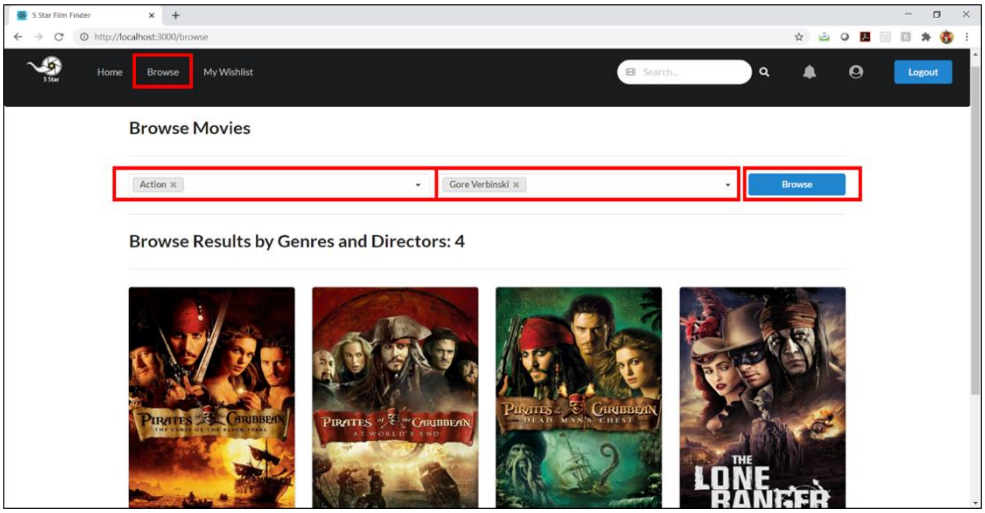
Similarly, user enters some text describing a movie and the results will be generated based on the description.



8. Browse movies

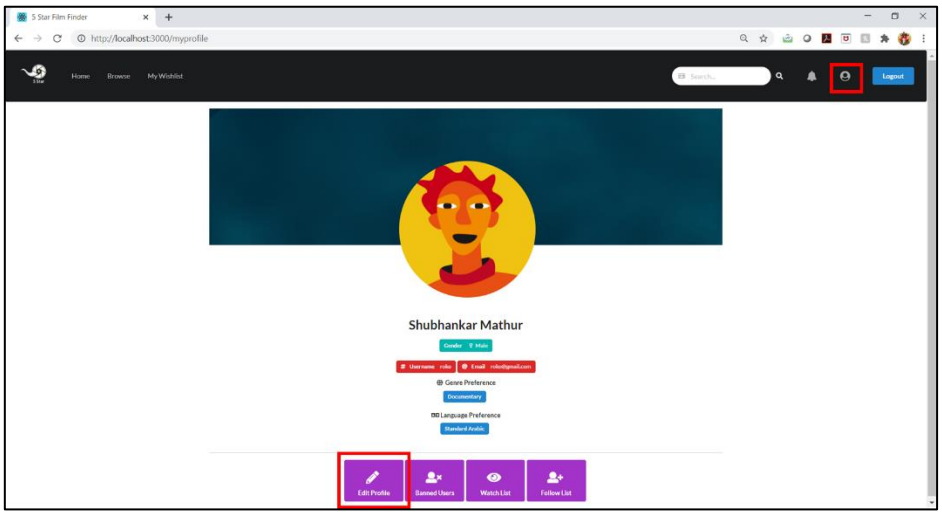
Click the Browse button on the Navigation bar next to the Home button to browse through the list of movies. The browse capability offers the user to search through a combination of different genres and directors. A user can select a “genre” for e.g.: Action or director for e.g. “Gore Verbinski” or both as a browse option and click Browse. This will generate all results based on the user preferences.

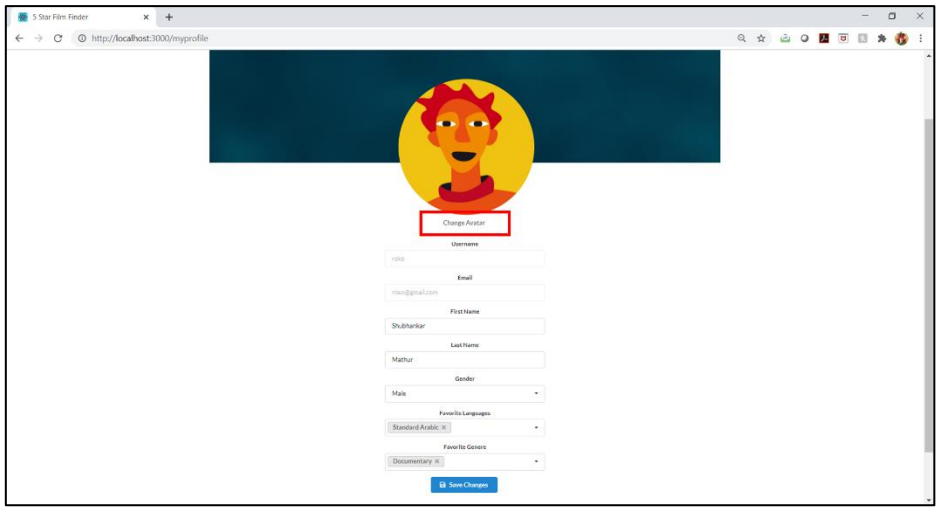
Route : Browse -> Enter preferences -> Browse button

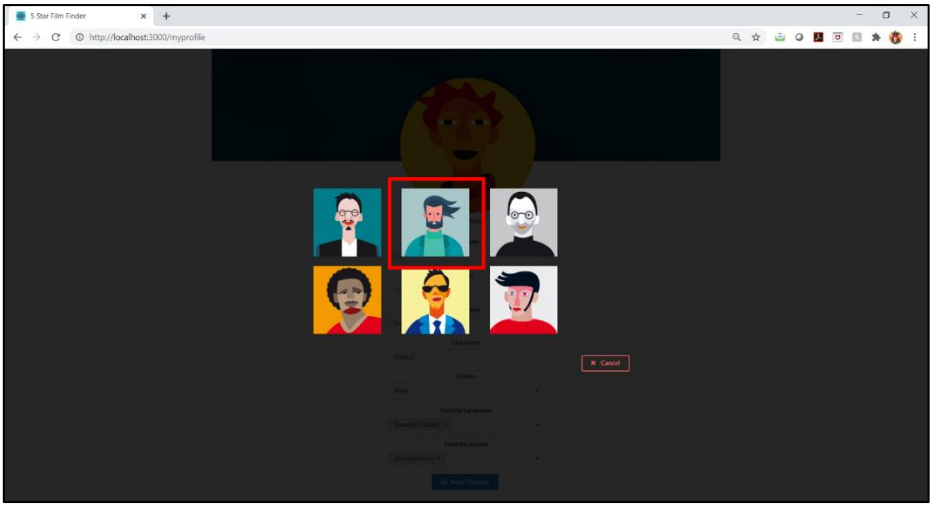


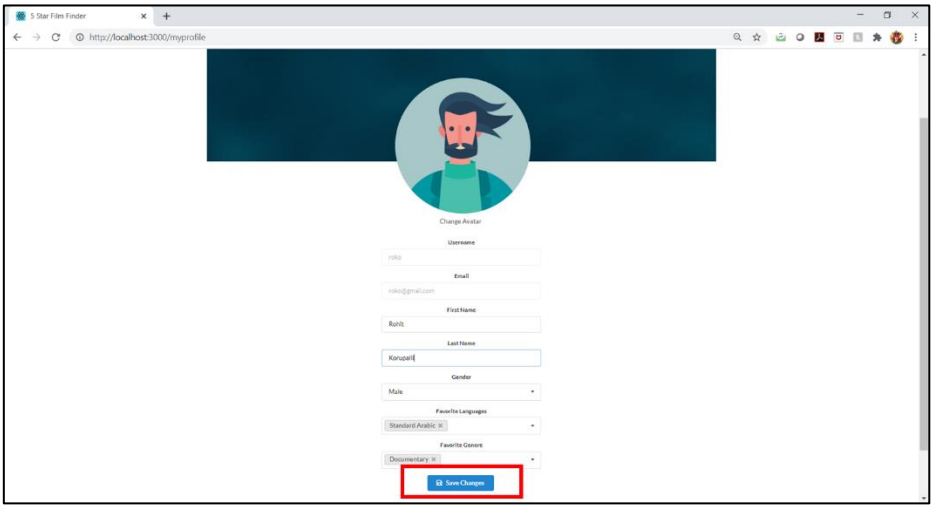
9. Manage my Profile

A FilmFinder who has logged in to the website can view their profile by clicking the profile icon next to the notification icon on the navigation bar. Once the user is redirected to the profile page, they have the ability to manage their profile. A FilmFinder can edit all the information that they used during signing up except email and username. Apart from that, user can Manage their Watchlist, Banned List and Follow List.





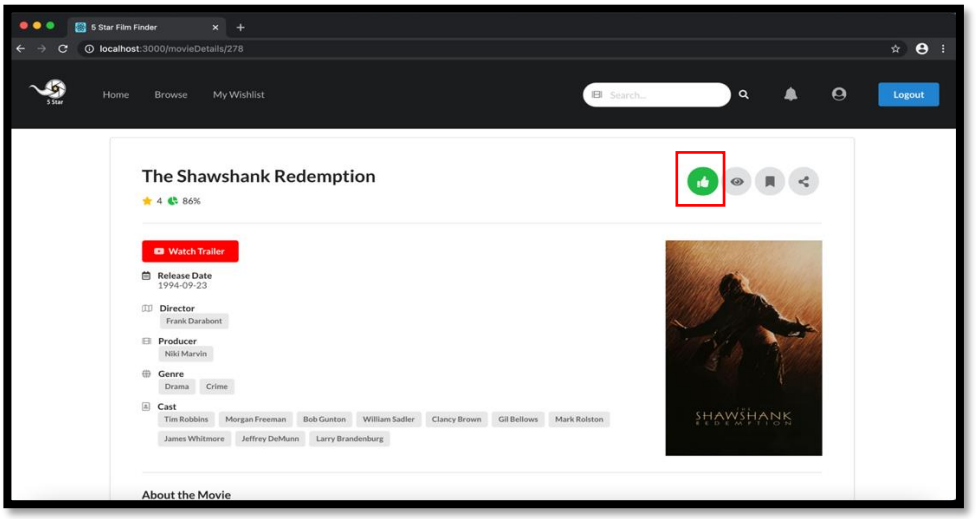




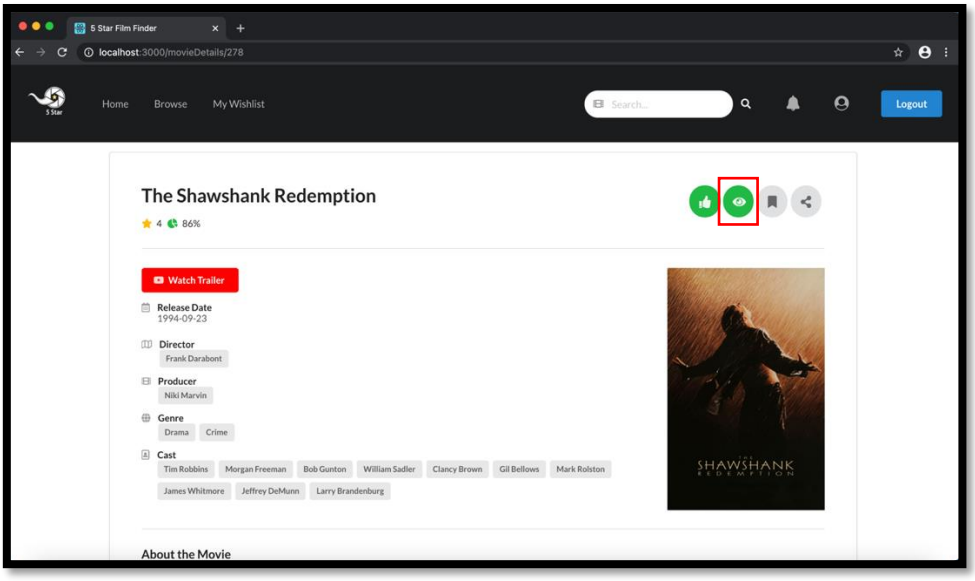
10. Add movies to Wishlist, Watchlist and Share a Movie

If the FilmFinder is logged in, go to the View Details tab by hovering on any movie tile and the user will land on the movie details page. On the top right corner above the movie poster, a user can accomplish various tasks by clicking the different icons.

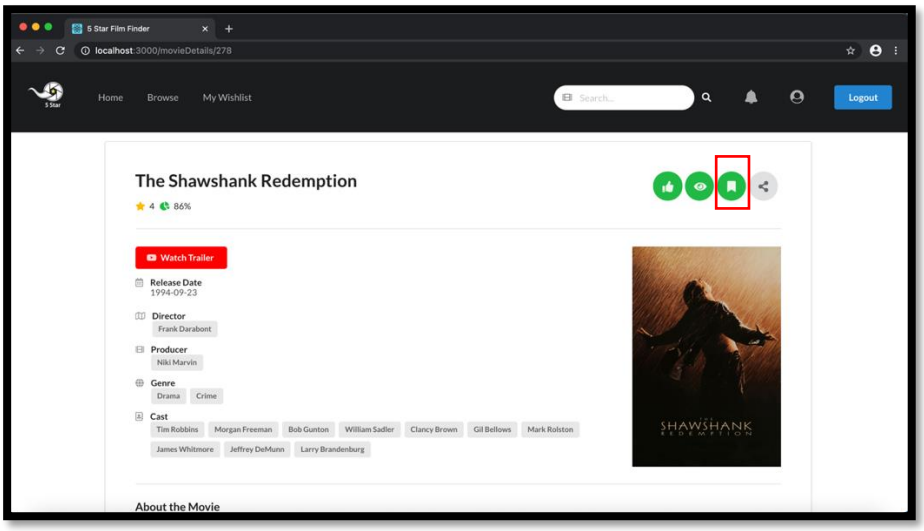
- Like Icon: Indicates if you have liked a movie.



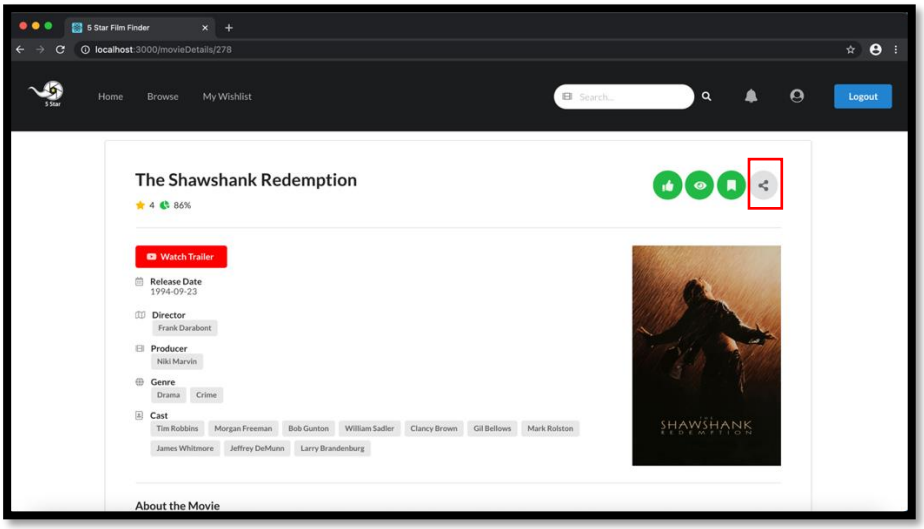
- Watchlist icon: Adds movie to the watchlist

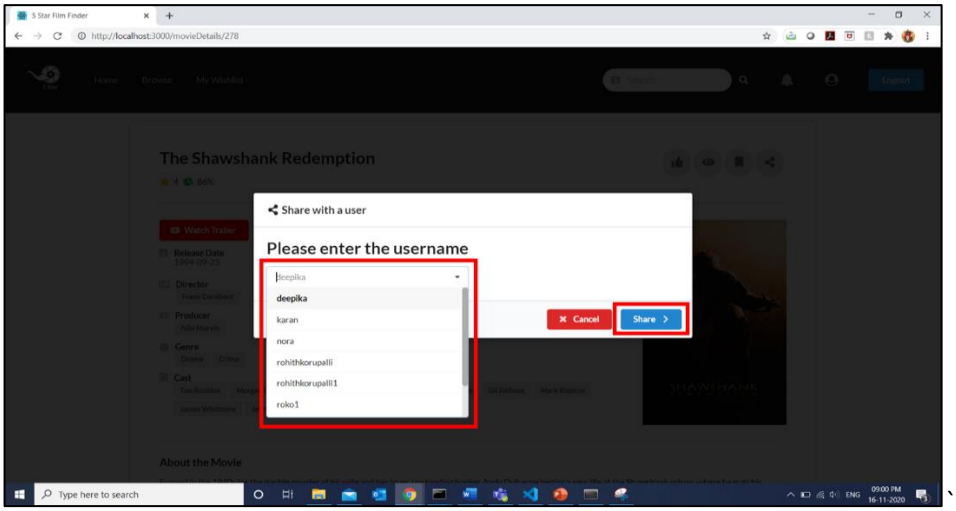


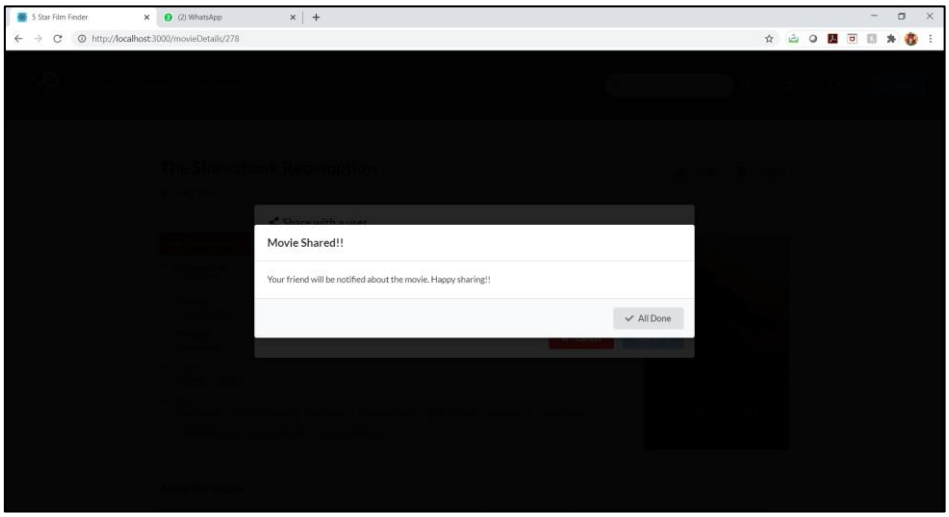
- Bookmark Icon: Adds movie to the Wishlist



- Share Icon: User can share this movie with other users by browsing the username in the pop-up menu

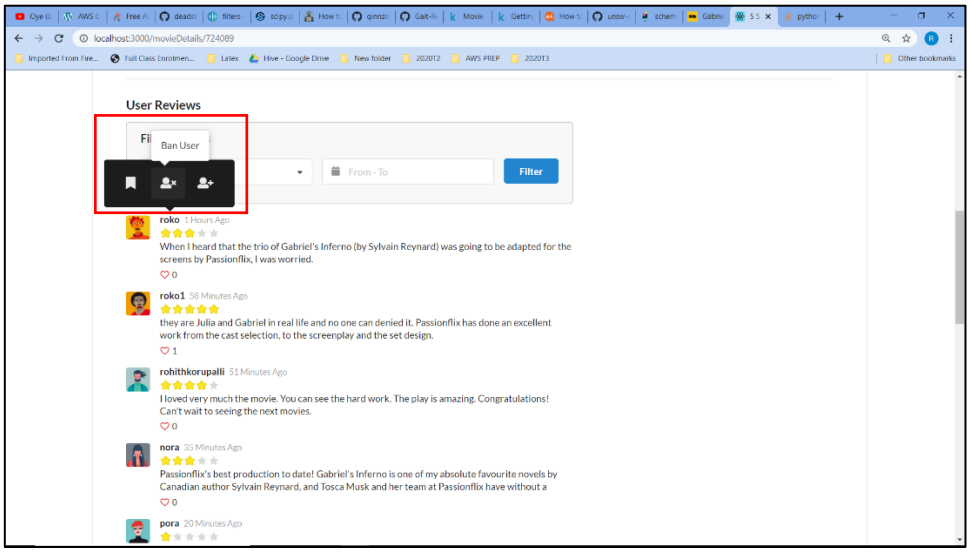

5 Star Film Finder
localhost:3000/movieDetails/278
Home
Browse
My Wishlist
Logout
The Shawshank Redemption
4 86%
Watch Trailer
Release Date
1994-09-23
Director
Frank Darabont
Producer
Niki Marvin
Genre
Drama
Crime
Cast
Tim Robbins
Morgan Freeman
Bob Gunton
William Sadler
Clancy Brown
Gil Bellows
Mark Rolston
James Whitmore
Jeffrey DeMunn
Larry Brandenburg
About the Movie

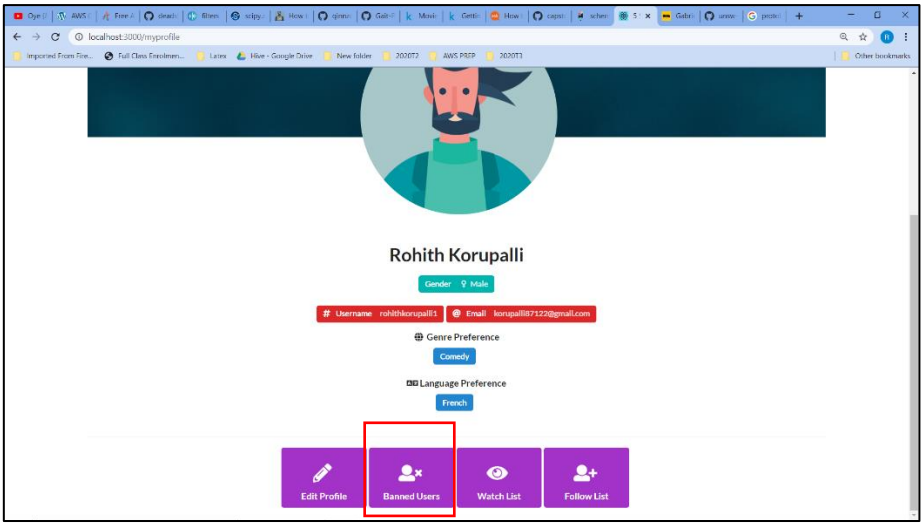

Share with a user
Please enter the username
deepika
deepika
karan
nora
rohithkorupalli
rohithkorupalli1
roko1
Cancel
Share
About the Movie
Type here to search

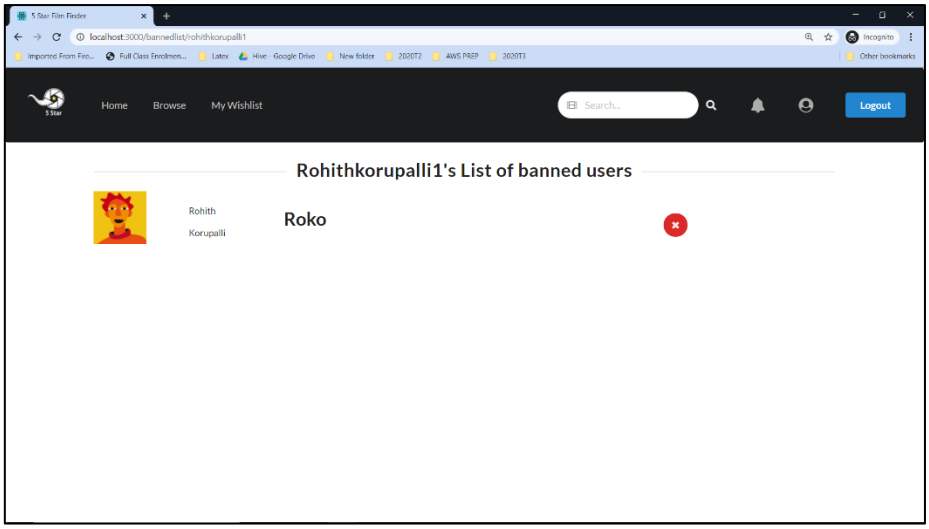

Movie Shared!!
Your friend will be notified about the movie. Happy sharing!!
All Done

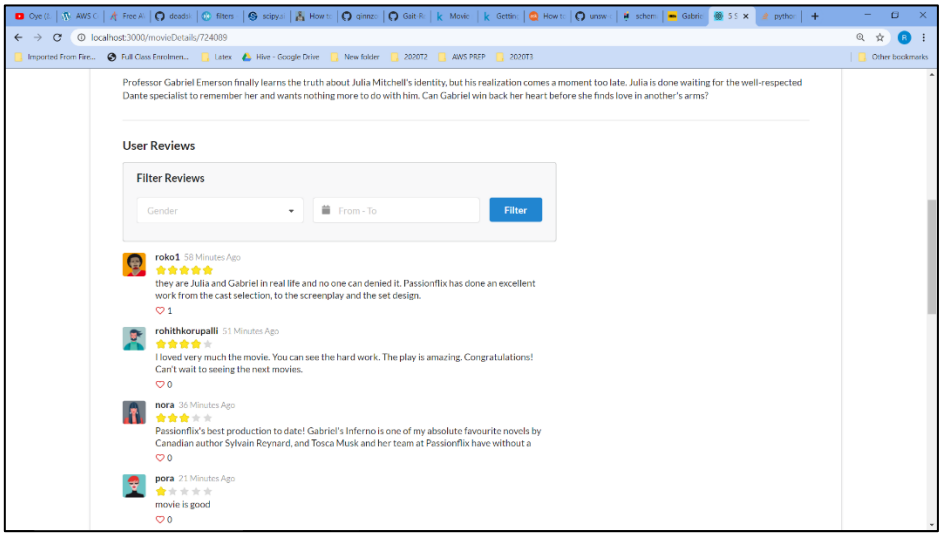
11. Manage ban user list

A FilmFinder can ban a user by hovering over a user's review that they wish to remove from their movie's review list. The user can click the *Ban User* icon in the pop up and the user's review will no longer be shown on the page. This feature will also block the user from appearing in the feeds in future until the user has been removed from the banned list. A FilmFinder can remove a banned user from the banned list by going to the profile page (see 9. Manage my profile).



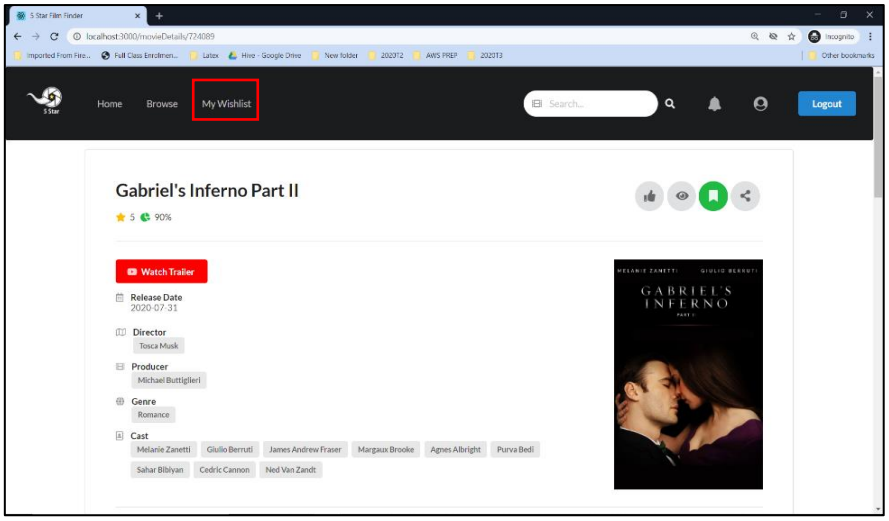


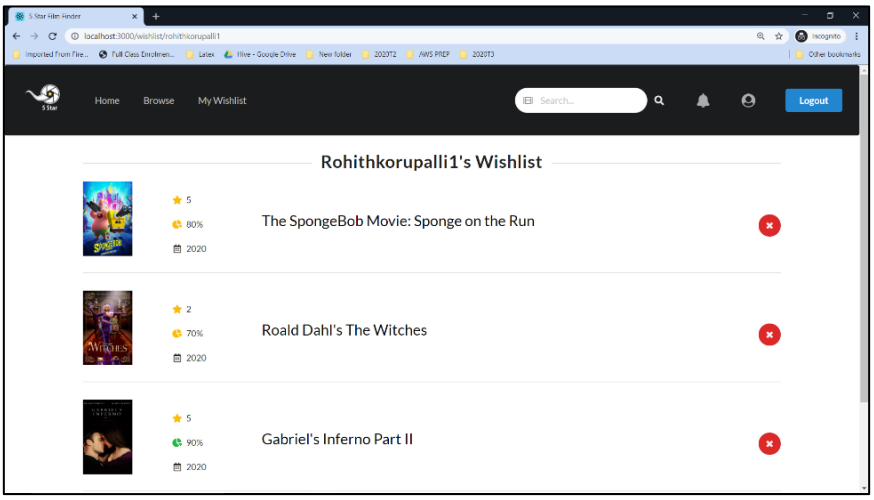




12. Manage my Wishlist

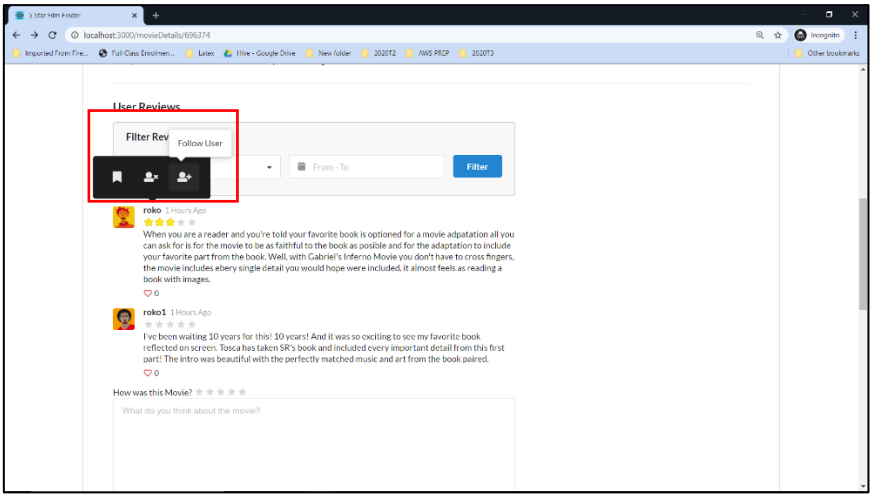
A FilmFinder can view their Wishlist by clicking the My Wishlist tab on the Navigation bar next to the Browse tab. FilmFinders can remove a movie by clicking the red cross button on the right side of the movie.

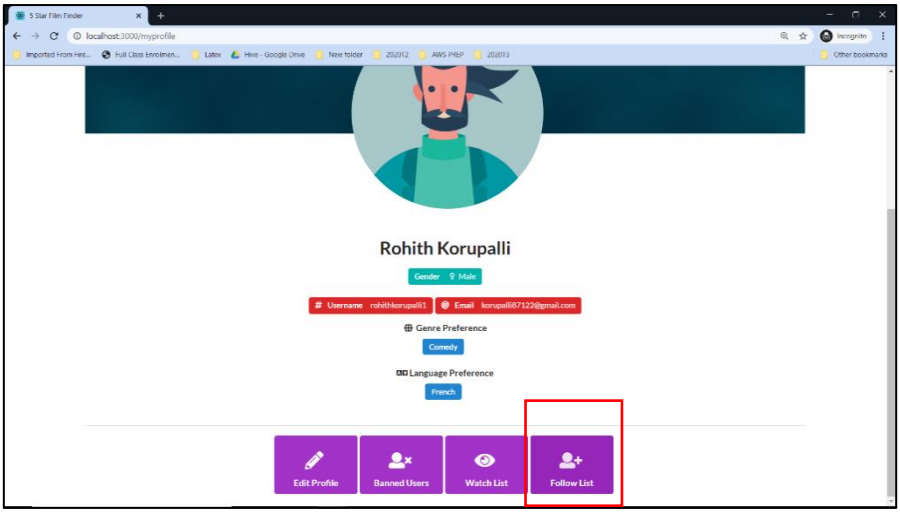


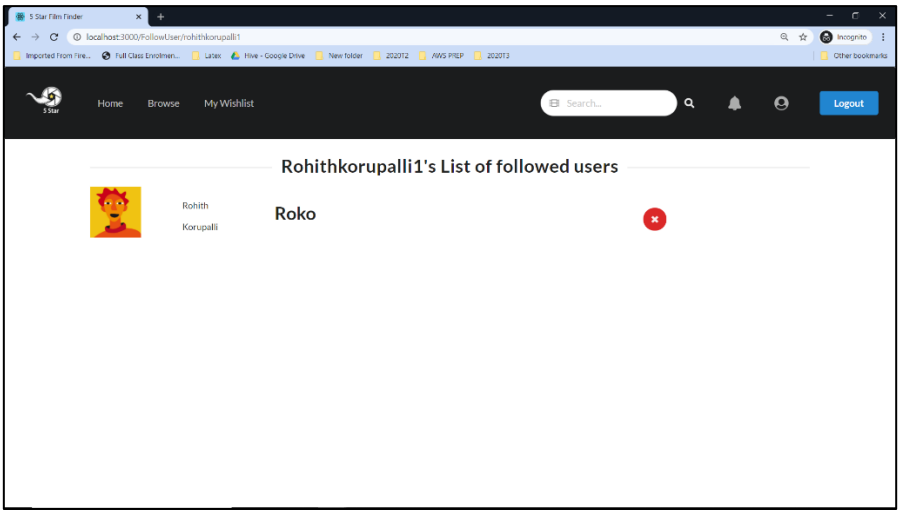


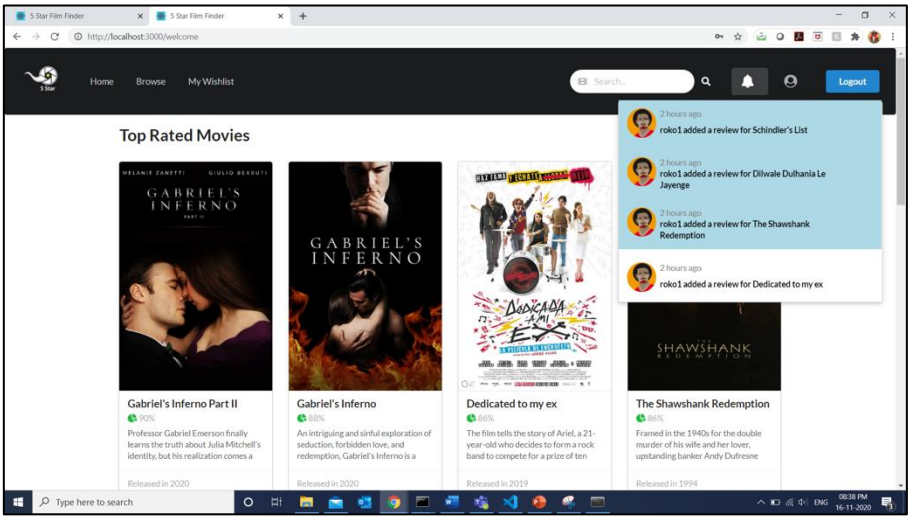
13. Manage followed user list

A FilmFinder can follow a user by hovering over that particular user's review. The user can click the *Follow User* icon in the pop up and that user review will be added to the follow list. A FilmFinder can remove a followed user from the Follow list by going to the profile page (see 9. Manage my profile).



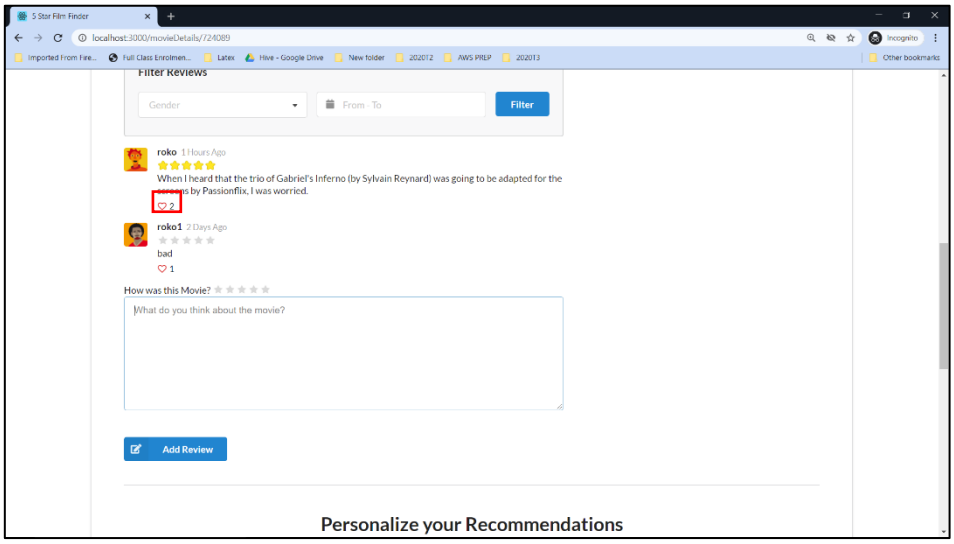

Rohith Korupalli
Gender Male
Username rohithkorupalli1
Email korupalli87122@gmail.com
Genre Preference
Comedy
Language Preference
French
Edit Profile
Banned Users
Watch List
Follow List

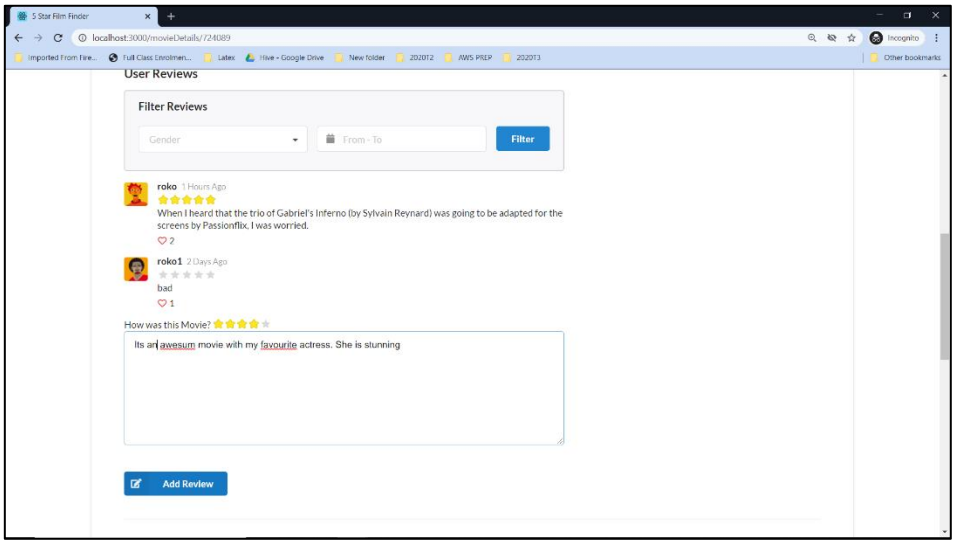

Home
Browse
My Wishlist
Logout
Rohithkorupalli1's List of followed users
Rohith
Korupalli
Roko

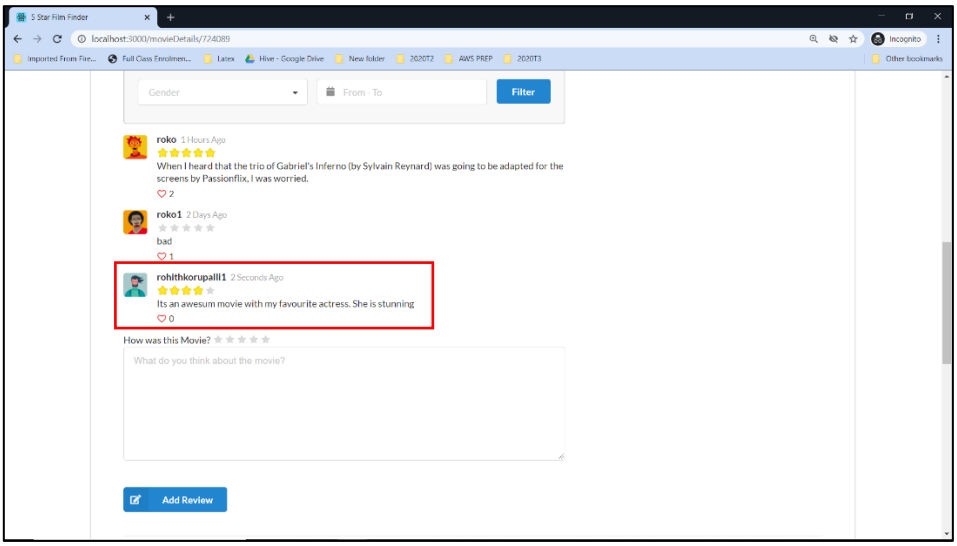

Home
Browse
My Wishlist
Logout
Top Rated Movies
roko1 added a review for Schindler's List
roko1 added a review for Dilwale Dulhania Le Jayenge
roko1 added a review for The Shawshank Redemption
roko1 added a review for Dedicated to my ex
Gabriel's Inferno Part II
Gabriel's Inferno
Dedicated to my ex
The Shawshank Redemption

14. Manage Reviews: Add, like and rate

A FilmFinder can add a text review, like other user's review and rate that particular movie from 0-5 on the movie details page (see 5. View Movie Details).

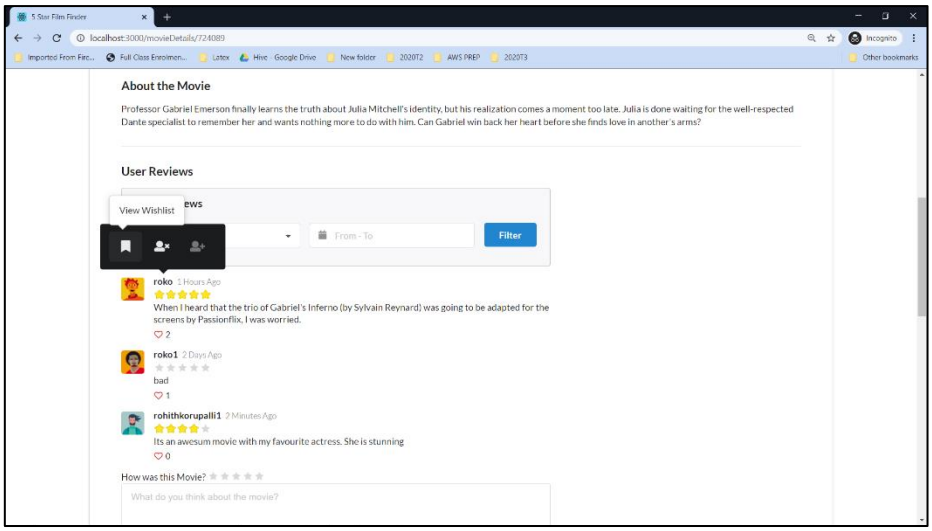


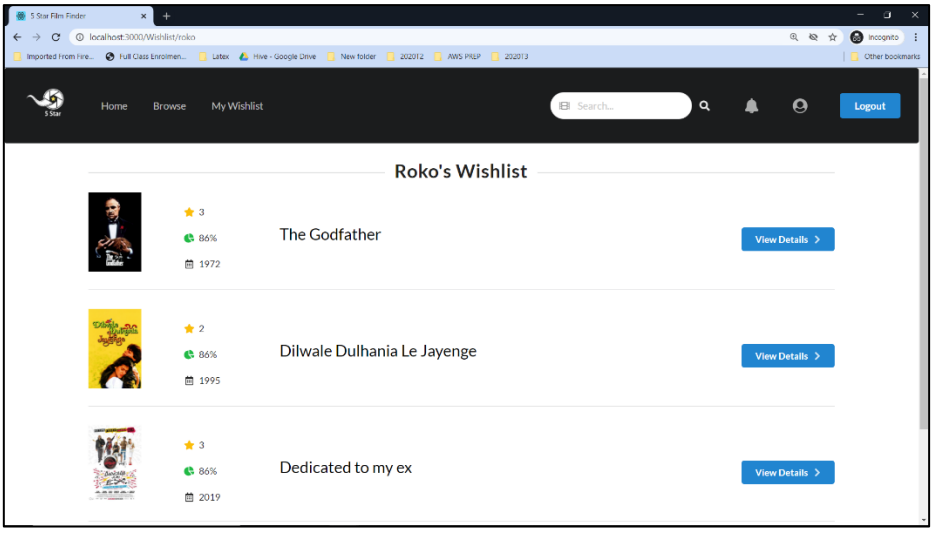




15. View other review writer's Wishlist

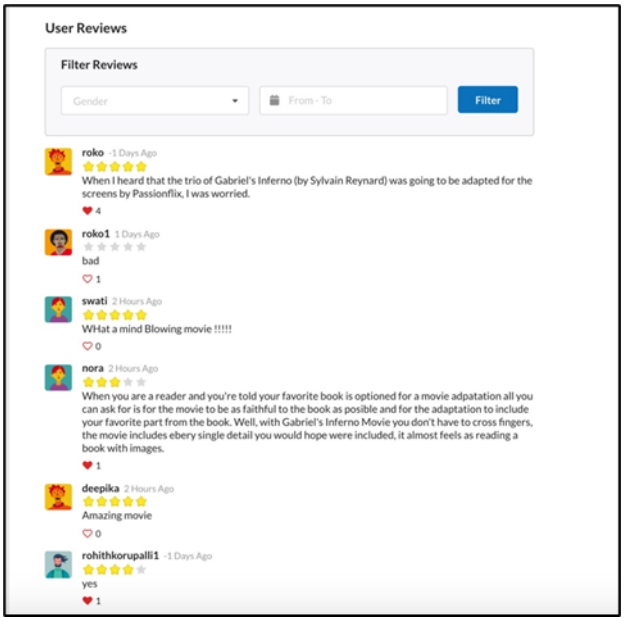
A FilmFinder can hover over a user's review to view that particular user's Wishlist.

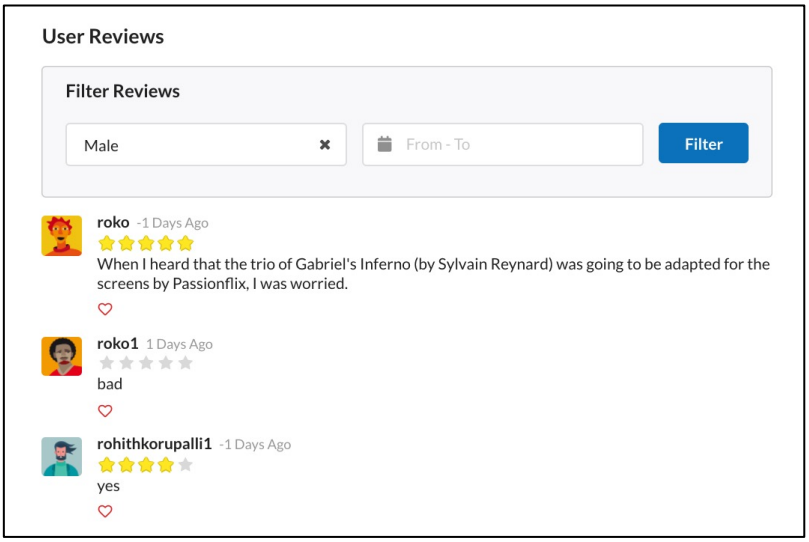


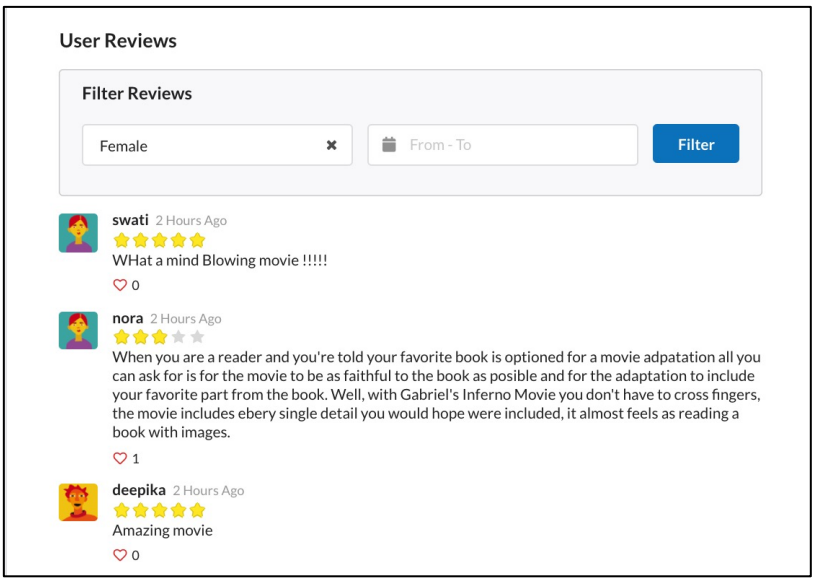


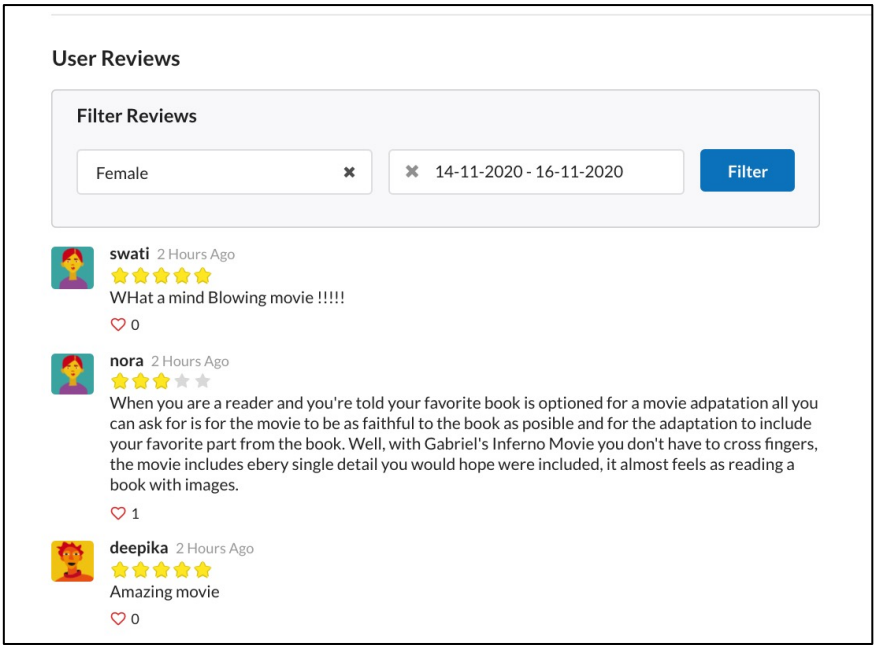
16. Filter Reviews

A FilmFinder has the capability to filter the reviews based on gender and date range. In order to filter the reviews, a user simply has to visit the movie details page by going to view details button on. Movie tile and scroll down to the section to filter reviews. Next, a user selects the preferences from gender and the date range through which the reviews should be filtered. Finally, click the filter button to see the results.


User Reviews
Filter Reviews
Gender
From - To
Filter
roko -1 Days Ago
When I heard that the trio of Gabriel's Inferno (by Sylvain Reynard) was going to be adapted for the screens by Passionflix, I was worried.
4
roko1 1 Days Ago
bad
1
swati 2 Hours Ago
WHat a mind Blowing movie !!!!!
0
nora 2 Hours Ago
When you are a reader and you're told your favorite book is optioned for a movie adpatation all you can ask for is for the movie to be as faithful to the book as posible and for the adaptation to include your favorite part from the book. Well, with Gabriel's Inferno Movie you don't have to cross fingers, the movie includes ebery single detail you would hope were included, it almost feels as reading a book with images.
1
deepika 2 Hours Ago
Amazing movie
0
rohithkorupalli1 -1 Days Ago
yes
1

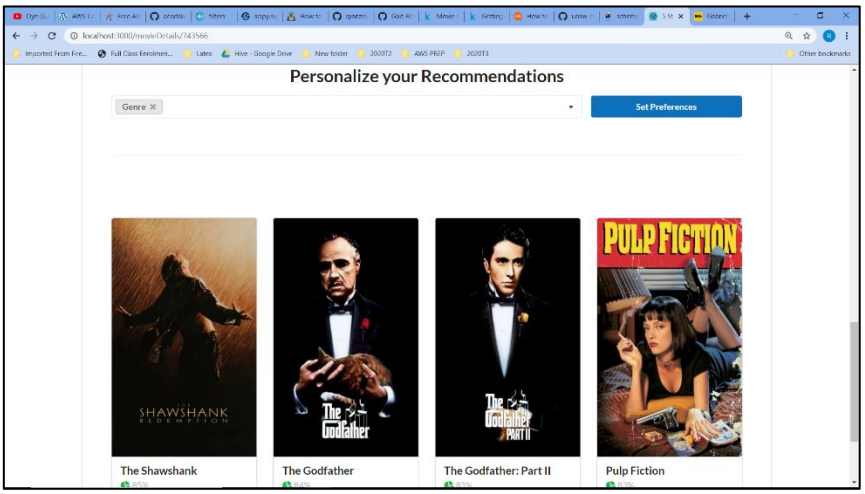

User Reviews
Filter Reviews
Male
From - To
Filter
roko -1 Days Ago
When I heard that the trio of Gabriel's Inferno (by Sylvain Reynard) was going to be adapted for the screens by Passionflix, I was worried.
roko1 1 Days Ago
bad
rohithkorupalli1 -1 Days Ago
yes

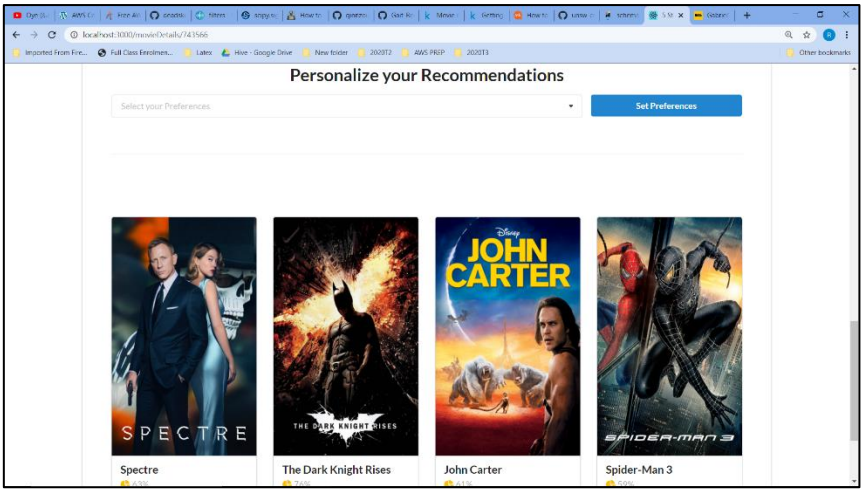

User Reviews
Filter Reviews
Female
From - To
Filter
swati 2 Hours Ago
WHat a mind Blowing movie !!!!!
0
nora 2 Hours Ago
When you are a reader and you're told your favorite book is optioned for a movie adpatation all you can ask for is for the movie to be as faithful to the book as posible and for the adaptation to include your favorite part from the book. Well, with Gabriel's Inferno Movie you don't have to cross fingers, the movie includes ebery single detail you would hope were included, it almost feels as reading a book with images.
1
deepika 2 Hours Ago
Amazing movie
0

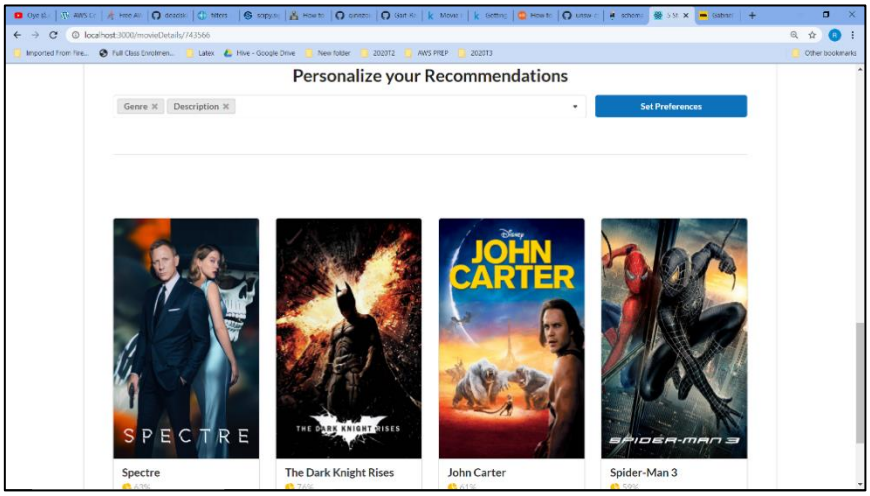


17. Manage recommendations

A FilmFinder can set the preferences for recommendations. These Recommendations will be based on Genre or description. A user has the capability to personalize the recommendations once they scroll down in the Movie Details page. They can select Genre, Director or both to personalize their recommendations.

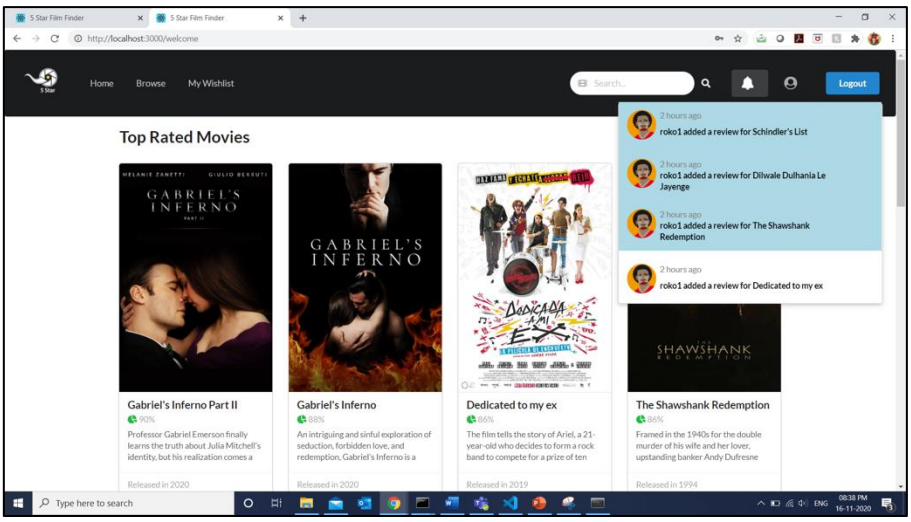






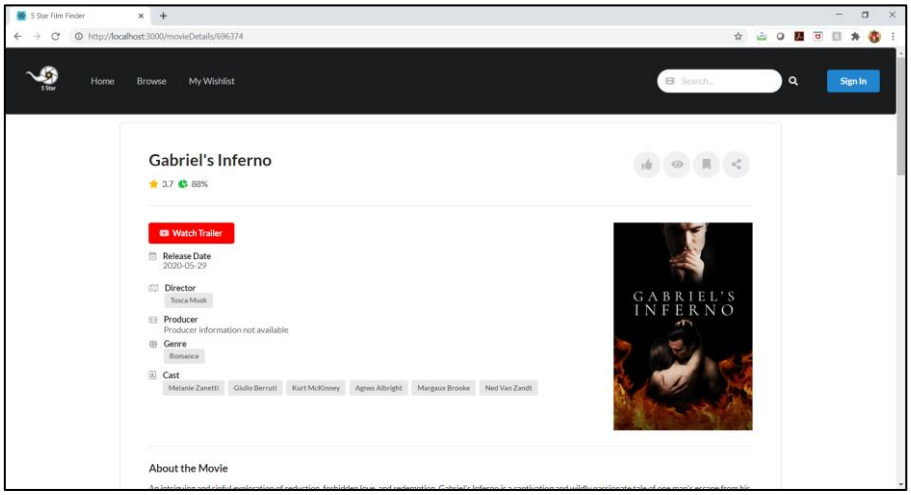
18. View Notifications

A FilmFinder gets notified if the user that they follow (see 13. Manage Followed user list) shared a movie with them or if any user had added any new reviews on the movies. These notifications will be shown in the bell icon next to the search area.



19. Anonymous user access

If a user does not wish to log in, they can still access the website with some restricted functionalities. A guest user can access and view the details of the movie, access the search functionality and browse movies.



Appendix 1 (API'S Implemented)

1) **User Login View:** This API accepts POST requests from the front end. The request body will consist of Email and Password. The user is authenticated with these details. If a user profile exists in our database HTTP 200 Response is sent else error "A User with this email and Password is not found" is sent.

   Request: http://127.0.0.1:8000/api/login

   Request Body: {'email': string, "password": string}

   Response: {"response": {"success": string,"statusCode": int, "message": string","id": string, "email": string,"firstname": string,"lastname": string,"username": string,"profilePic": string}}

2) **Homepage:** This API accepts the GET request from the front end. Users can access this page as a Guest or registered User. This API retrieves data from the External API (TMDB). Data consists of Popular, top-rated, and Currently playing movies list.

   Request: http://127.0.0.1:8000/api/homepage

   Response: {"Popular_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string, "release_date": Date }], "TopRated_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string,"release_date": Date }], "Nowplaying_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string,"release_date": Date }]}

3) **Movie Search:** This API accepts GET requests from the front end. Request Body consists of QUERY (movie name, Genre name, or Description) of a particular movie. This API is designed to return movies based on Genre, Movie Name, and Description. If the query consists of a particular movie name all the movies related to it are displayed in Name results. If the query consists of genre names like Action, Adventure, etc all the Movies related to Genre are displayed. If the query consists of a plot or movie description all the movies with similar descriptions are displayed.

   Request: http://127.0.0.1:8000/api/search/?query= string

   Response: { "Genre_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string, "release_date": Date }],"name_results":[{"id": int , "title": string,"rating": int, "description": string, "poster": string, "release_date": Date }], "description_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string,"release_date": Date }]}

4) **Movie Details:** This API accepts POST requests from the front end. The request body must contain User and Movie ID requested, it also has optional parameters gender sort and Date range sort. API is designed to return details of a particular movie that contains Movie Cast and Crew, Description, Rating, Reviews, etc. Optional parameters in the request body are used to filter the reviews obtained from API. Gender filter has 2 options "Male" or "Female". Date filter needs to provide from a date in "YYYY-MM-DD" format and same with to_date

   Request: http://127.0.0.1:8000/api/moviedetail/

   Mandatory parameters: {'user': string, "id": int}

   Optional Parameters: {"gender_sort: list, from_date:Date, to_date:"Date"}

Response: {"id": int,"tmdb_rating": int,"description": string,"poster":tring,"genres": list,"trailers":list,"cast":list,"director":list,"producer":list,"release_date":Date,"teasers":list,"recommendations":[{"movieID":int,"movieTitle":string,"description":string,"rating":int,"poster": string,"releaseDate":Date}}

5) **Movie Browse:** This API accepts POST requests from the front end. The request body is optional. If the request body is empty Browse page is displayed. If the request body consists only Genre_ID all the movies in that Genre_Id are displayed. If the request body consists only of Director ID, all the movies directed by that director are displayed. If the request body consists of both Genre and Director ID's, all the movies directed by that Director in the genre given are displayed. Results are sorted based on Average rating in Descending order. If both movies have the same rating, then the movie name is sorted based on alphabetical order.

   Request: http://127.0.0.1:8000/api/browse/

   Optional Parameters:{'genre_id': list},{'director_id': list},{'genre_id': list, 'director_id': list}

   Response: {"Genre_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string, "release_date": Date }], "Director_results": [{"id": int , "title": string,"rating": int, "description": string, "poster": string,"release_date": Date }]}

6) **Add Review:** This API accepts POST requests from the front end. Whenever Users want to give review and rating for a particular movie this API is called. This API checks if the user review exists or not. If it exists then the review text, rating, review modified time and date, upvote, and downvote data are updated else new entry is created in Database for that movie by that user.

   Request: http://127.0.0.1:8000/api/addreview/

   Mandatory Parameters: {"movie": string, "user": string, "review": string ," rating": int}

   Response:{ "success": Boolean, "statusCode": int }

7) **Get Review:** This API accepts POST requests from the front end. Whenever Users want to view reviews for a particular movie this API is called. This API is called whenever the user is redirected to the Movie Details page.

   Request: http://127.0.0.1:8000/api/getreview/

   Mandatory Parameters: {"movie": string , "user": string}

   Response::{"success":Boolean, "statusCode": int ,"review":list,"datemodified":Date,"follow":list, "watched":Boolean, "liked":Boolean, "upvote":Boolean}

8) **Add to Wishlist:** This API accepts PUT request from the front end. Whenever a user wants to add a particular movie to his wishlist this API is called. This API requires Movie Id, Username, and Wishlist (True or False). If the request is sent with wishlist=True, then the movie is added to the User's Wishlist database. If a request is sent with wishlist=False, then the movie is deleted from the User's Wishlist database

   Request: http://127.0.0.1:8000/api/addWishlist/

   Mandatory Parameters: {"movie": string, "user": string, "wishlist": string }

   Response:{ "success": Boolean, "statusCode": int ,"message":string}

9) **Get Wishlist:** This API accepts POST requests from the front end. Whenever Users want to view the wishlist of a particular user this API is called. Users can view their wishlist or other users' wishlist.

   Request: http://127.0.0.1:8000/api/viewWishlist/

   Mandatory Parameters: {“movie”: string, “user”: string }

   Response:{ "success": Boolean, "statusCode": int ,”wishlist”:list}

10) **Liked:** This API accepts PUT request from the front end. Whenever Users want to Like/Dislike a particular movie this API is called. This API requires Movie Id, Username, and like the movie (True or False). If the request is sent with likeMovie=True , then the movie is added to the User’s likeMovie database. If a request is sent with likeMovie =False, then the movie is deleted from the User’s likeMovie database

    Request: http://127.0.0.1:8000/api/likeMovie/

    Mandatory Parameters: {“movie”: string, “user”: string, “likeMovie”: string }

    Response:{ "success": Boolean, "statusCode": int ,”message”:string}

11) **Upvote:** This API accepts POST requests from the front end. Whenever a user wants to give Upvote for a particular user this API is called. This API requires Movie Id, Username, and Upvote. All the votes count for that movie is returned.

    Request: http://127.0.0.1:8000/api/upvote/

    Mandatory Parameters: {“movie”: string, “user”: string, “upvote”: string }

    Response: {"success": Boolean, "statusCode": int ,"count": int}

12) **Get Notifications:** There are three scenarios considered for displaying notifications. They are when a user is being followed by other user, when a followed used gives a review and to suggest a movie to other users. This a POST request.

    Request: http://127.0.0.1:8000/api/getNotifications
    Mandatory Params: {“userID” : string }
    Response: {"success": Boolean,"notifications": list,"totNotifications": int,"newNotifications": int}

13) Read Notifications: Keeps track of read notifications. This a POST request.

    Request: http://127.0.0.1:8000/api/NotificationRead
    Mandatory Params: {“userID” : string }
    Response: {"success": Boolean}

14) **Get users:** Gives a list of all the active users. This a POST request.

    Request: http://127.0.0.1:8000/api/users
    Mandatory Params: None
    Response: {"success": Boolean,"users": list}

15) **Banning users:** a user can ban/unban other users using their username This is a PUT request. Doubling this request will unban user.

    Request: http://127.0.0.1:8000/api/banUsername
    Params: {"username": string,"bannedUsername": string }
    Response: {"success": Boolean, "statusCode": int ,"data": list}

16) **Get Banned Users:** A user can view all the users he banned. This is a post request.

    Request: http://127.0.0.1:8000/api/banUsername
    Params: {"username": string }
    Response: {"success": Boolean,"data": list}

17) **Suggest Movie:** A user can suggest a movie to other user. This is a POST request

    Request: http://127.0.0.1:8000/api/suggestMovie
    Params: {"fromUser": string,"toUser": string,"movieId": int}
    Response: {"success": Boolean }

18) **Follow User:** A user can follow other users to get notifications of their activities. This is a PUT request

    Request: http://127.0.0.1:8000/api/followUser
    Params: {"follower": string,"followee": string }
    Response: {"success": Boolean, "statusCode": int ,"message": string}

19) Following Users: Get a list of all the users you are following. This is a post request.

    Request: http://127.0.0.1:8000/api/followUser
    Params: {"userID": string }
    Response: {"success": Boolean, "statusCode": int ,"following": list}

20) **Unfollow User:** A user can unfollow a currently following user. This is a PUT request.

    Request: http://127.0.0.1:8000/api/unfollowUser
    Params: {"folloewer": string,"followee": string }
    Response: {"success": Boolean, "statusCode": int ,"message": string}

References

[1] Munguia, E. (2014, 2 19). prezi. Retrieved from Violence and Media: Are Rating Systems Necessary.

[2] Reynolds, M. (2017, 10 24). You should ignore film ratings on IMDb and Rotten Tomatoes. Retrieved from wired.co.uk: https://www.wired.co.uk/article/which-film-ranking-site-should-i-trust-rotten-tomatoes-imdb-metacritic

[3] Stegner, B. (2018, 2 21). Which Movie Ratings Site Is Best. Retrieved from makeuseof.com: https://www.makeuseof.com/tag/best-movie-ratings-sites/

[4] TMDB (2020) , Movie Detail API, TMDB ORG: https://developers.themoviedb.org/3

[5] Yuefeng Zhang(2015). Machine Learning for Building Recommender System in Python: https://towardsdatascience.com/machine-learning-for-building-recommender-system-in-python-9e4922dd7e97