Fresh Tomatoes: All You Search for in Movies

By: Team Fresh Tomatoes
Yvonne Chan, Kevin Zhang, Jonathan Buhler, Nala Sharadjaya

I. Directory Structure

```
~/Github/FreshTomatoes/
  -  app.py
  -  /utils/
       -  authen.py
       -  self.py
       -  search.py
       -  movie.py
       -  interactAPI.py
  -  /templates/
       -  login.html
       -  register.html
       -  home.html
       -  results.html
       -  savedMovies.html
       -  movie.html
       -  account.html
       -  editAccount.html
  -  /static/
       -  /bootstrap/
       -  jquery.min.js
       -  style.css
  -  /data/
       -  database.db
```

II. File Functionality

app.py

Main file that controls routing. Calls the functions of other files.

authen.py

Contains functions for login and register. It will access *database.db* for authentication information and check if information matches. In addition, it will check username for redundancy. Also edits *database.db*.

self.py

Access *database.db* for the account table and list table. The Account table will be used in conjunction with *account.html* and *editProfile.html*, while the list table will be used in conjunction with *myMovies.html*. This file will also edit the dataBase to commit any of the user's changes.

search.py

This file processes the user's search query. This file will work in conjunction with *interactAPI.py* in order to get the information it needs, namely the movie titles that match and their poster images.

movie.py

This file is responsible for building the *movie.html* page. This will work in conjunction with *interactAPI.py* and *search.py* in order to access the requested information.

interactAPI.py

This file is responsible for getting access to the APIs when given a query, and return all relevant information in the form as a dictionary. This file will not do anything else, only for access to the necessary APIs.

login.html

Web page that contains form for logging in and registering. It will contain two fields, one for username and one for password. Two buttons will also be included, one for register and one for submit.

register.html

Web page that looks a lot like *login.html* just with a third field for confirming the password.

home.html

Essentially the homepage. Looks a lot like the Google homepage, with a search bar in the middle and a navigation bar along the top. Also includes a random "I'm feeling bored" button, which will redirect the user to a random movie.

results.html

Returns the results of the search, particularly useful for multiple editions of the same movie. This page will give a list of movies, organized alphabetically by title. A brief description is included as well, but for more information, you have to click on the movie link itself.

savedMovies.html

List of saved movies that the user has recorded. This looks very similar to *results.html*. Each movie has a separate link to its own html page.

movie.html

Displays the information regarding a specific movie, such as release date, ratings, links to reviews, summaries, etc. Includes any data that we could get using our APIs, although if the information is missing, then we will fill out whatever we can. If the user has mistyped, then it will simply display a 'movie not found' message.

account.html

Allows the user to see their account information. This will generally be filled by some personal and some movie-related information (such as favorite genres). The user is able to edit any of this information at any time. In addition, the user is capable of modifying their security details.

editAccount.html

Html page for editing the account information.

database.db

Database file. Includes username and salted passwords. Also includes users and their saved movies.

III. Database Schema

Users

TEXT user	TEXT pass
Stores usernames	Stores salted passwords

Account

TEXT user	TEXT genre	TEXT fav
Stores usernames	Stores user's favorite genre shown	Stores user's favorite movie

	on profile	

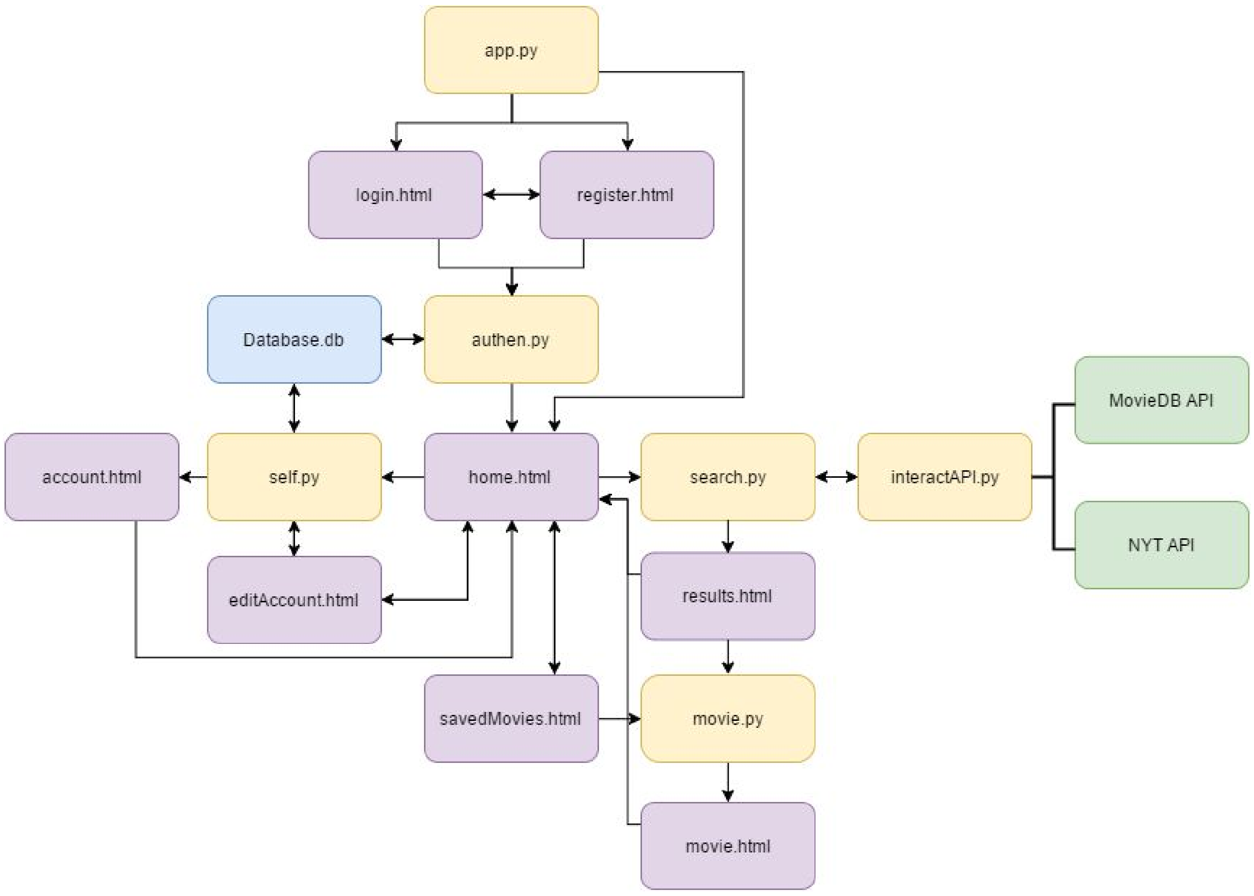
List

TEXT user	TEXT movie
Stores usernames	Movies on user’s list

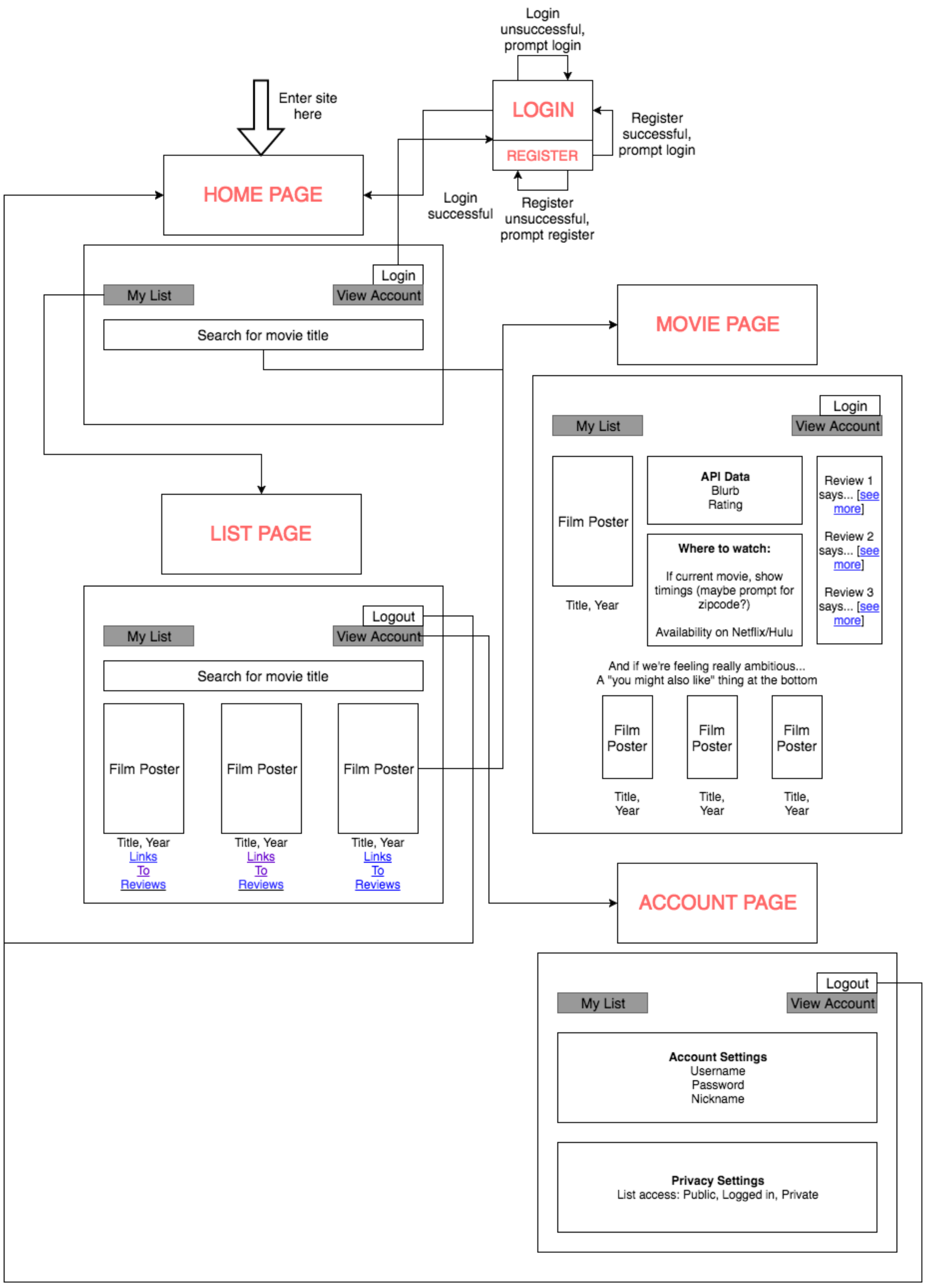
IV. APIs

A. New York Times Movie Review (FOR RATING): Link
B. Moviedb (FOR SEARCHING + RANDOM MOVIE): Link

V. Flow Chart



VI. Site Map



VII. Task Delegation

Yvonne - Project Manager

- Will debug and test everything
- Ensures everything works and is running smoothly
- Comments on other people's code to provide suggestions/ideas

Kevin - Front End

- Writes all HTML templates
- Help connect templates to app.py and backend files

Jonathan - Back End (+ database)

- Write authen.py to handle login/register
- Handle all writing to databases in dataAccess.py and other files

Nala - Back End

- Write interactAPI.py to handle retrieving information
- Access APIs and providing all information from them

VIII. TimeLine

[Will update devlog everyday, subject to change depending on problems we encounter]

12/5 - design document done, create files and get started

12/6 - get login/register/logout working and having working templates (no bootstrap/css)

12/7 - get search working initially

12/8 - put together movie pages with ratings, summaries, etc

12/9 - allow users to save movies and edit their profile

12/12 - make site look pretty with bootstrap/css, add "I'm feeling bored" button for random movie, ensure everything's working