Team Name: armRESTs
Roster: Robin H. (PM), Aleksandra K., Johnny W.
Project Title: Groovy Movie

Proposal Summary:

This team wishes to create a website that provides the user with information regarding movies. Users would be able to see trending movies, movies sorted under certain categories, search for movies through keywords or their mood at the time, add movies to their personal lists such as favorites or to-watch, and rate movies themselves.

Roles:

Robin H: As project manager, Robin maintains the devlog, makes sure the team works well together, maintains the design documentation, stays on top of design changes, and handles minor coding issues.

Aleksandra K: Will write API card for Fandango and MovieDB. Will work on the searching component, generating movie pages (and all of the associated features outlines below),

Johnny W: Will write API card for IPStack and ipify. Will work on front-end development, templates, current user location feature

Timeline:

11/20 Begin work on design doc

11/21 Build basic flask app and and set up devlog

11/22- 11/25 Flesh out basic pages based on sitemap, establish database, complete
    login/logout/create account functionality.

11/26 Setup homepage with currently trending movies (begin basic API work)

11/27 Write API cards for Fandango API and Movie DB and use them to implement
    basic search functionality

11/28 Continue work on Search (ie. Mood, Feeling Lucky) and configure templates for
    movies.

11/29 Add commenting, rating, and other user interfacing to movie pages.

11/30 Write API cards for IPStack and ipify APIs and then begin work on nearby movies
    currently streaming in local theaters

12/1-12/3 Finish work on displaying movies based on current location/ Tune search if
    necessary

12/4 Build User Profile page

12/5 Add similar movies features on movie pages/ recommendations based on user
    ratings

12/6 Add documentation to code and robustify

12/7- 12/9 Debug (find niche situations where there may be bugs) and spice up front end

12/10 Make sure all documentation up to date after weekend tweaking

12/11 Release and Demo!

API Links:

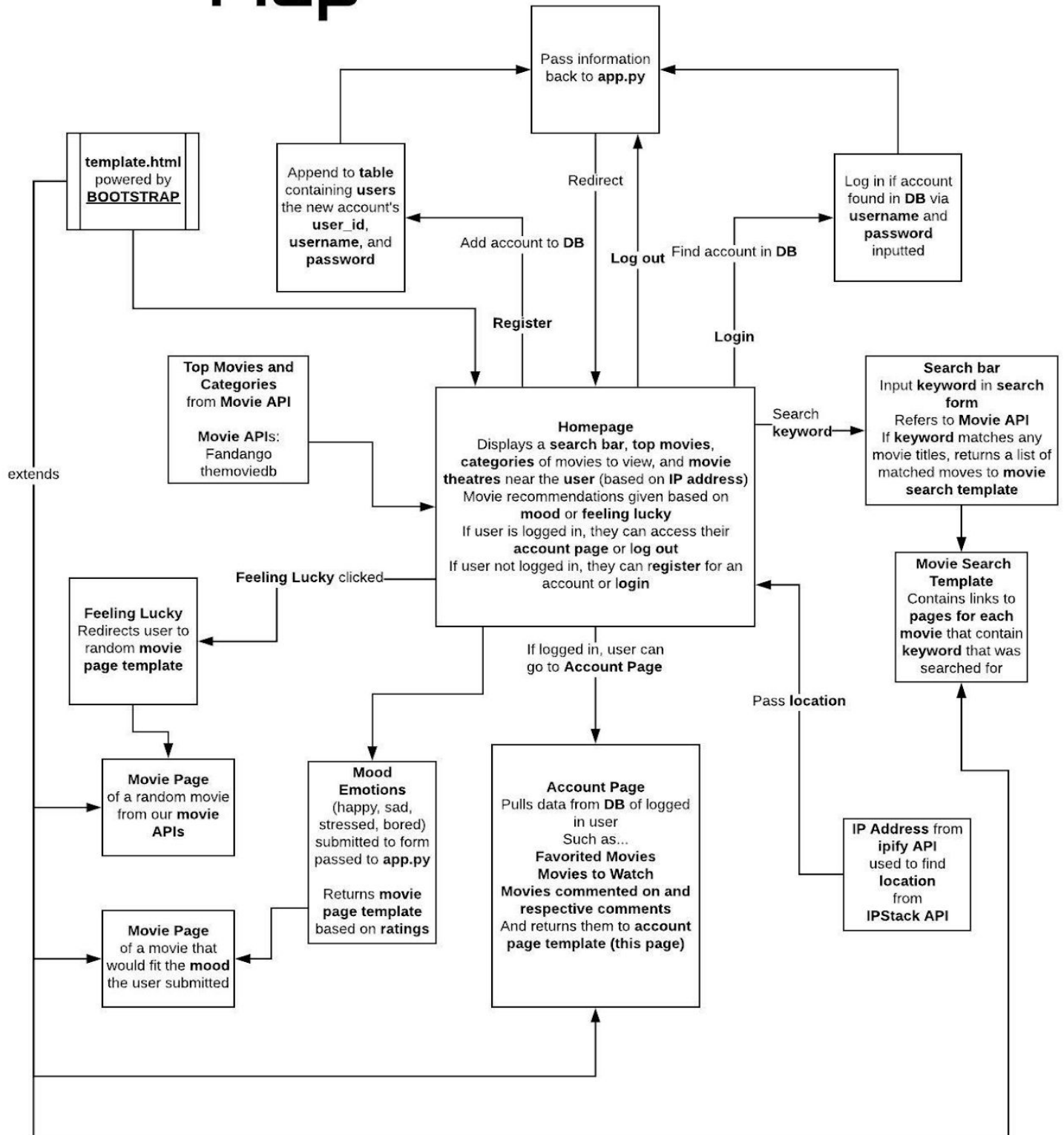We will be using several API's to build this site.

1) Fandango API (https://developer.fandango.com/fandango)
   - Available information includes: theaters, movies, showtimes, and editorial content.
   - User can identify theaters based on location and easily narrows results to just theaters playing a particular movie. Also has available amenities.
   - We will also be getting showtimes for necessary theaters from this API as well as movie ratings, runtime, genre, synopsis, and cast.

2) Movie DB (https://www.themoviedb.org/documentation/api)
   - We will be using this API to retrieve top rated movies, now playing movies, and those that are currently popular.
   - Used to cross-reference/ add to the information pulled from Fandango API.

3) MovieQuotes API (https://market.mashape.com/juanroldan1989/moviequotes**)**
   - Used to retrieve movie quotes for home page.
   - A fun feature for the user

4) ipstack API (https://ipstack.com/)
   - Used to locate and identify website visitors by IP address
   - Necessary for Currently Playing section of the site since we display movies in the user's locality

5) ipify API (https://www.ipify.org/)
   - Pulls your current IP address
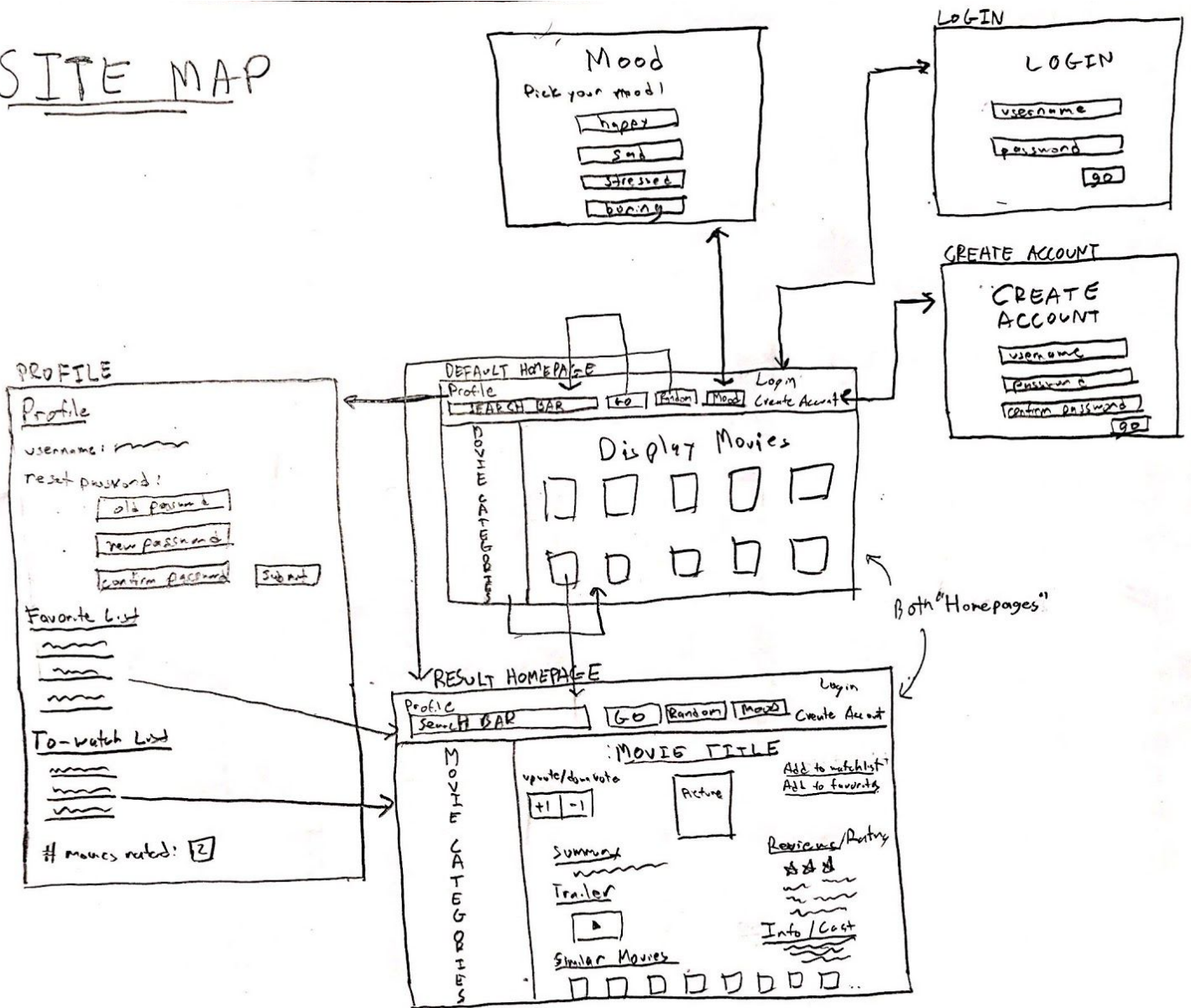   - Necessary for ipstack API since we need to pull the user's IP

Component Map

Team Name: armRESTs
Roster: Robin H. (PM), Aleksandra K., Johnny W.
Project Title: Groovy Movie

Pass information back to **app.py**

template.html powered by **BOOTSTRAP**

Append to **table** containing **users** the new account's **user_id**, **username**, and **password**

Log in if account found in **DB** via **username** and **password** inputted

Redirect

Add account to **DB**

Log out

Find account in **DB**

Register

Login

Top Movies and Categories from **Movie API**

Movie APIs: Fandango themoviedb

Homepage
Displays a **search bar**, **top movies**, **categories** of movies to view, and **movie theatres** near the **user** (based on **IP address**)
Movie recommendations given based on **mood** or **feeling lucky**
If user is logged in, they can access their **account page** or **log out**
If user not logged in, they can r**egister** for an account or l**ogin**

Search keyword

Search bar
Input **keyword** in **search form**
Refers to **Movie API**
If **keyword** matches any movie titles, returns a list of matched moves to **movie search template**

extends

Feeling Lucky clicked

Feeling Lucky
Redirects user to random **movie page template**

Movie Search Template
Contains links to **pages for each movie** that contain **keyword** that was searched for

If logged in, user can go to **Account Page**

Pass **location**

Movie Page
of a random movie from our **movie APIs**

Mood Emotions
(happy, sad, stressed, bored) submitted to form passed to **app.py**

Returns **movie page template** based on **ratings**

Account Page
Pulls data from **DB** of logged in user
Such as...
Favorited Movies
Movies to Watch
Movies commented on and respective comments
And returns them to **account page template (this page)**

IP Address from **ipify API** used to find **location** from **IPStack API**

Movie Page
of a movie that would fit the **mood** the user submitted

SITE MAP

Mood
Pick your mood!
- happy
- sad
- stressed
- boring

LOGIN
LOGIN
- username
- password
- go

CREATE ACCOUNT
CREATE ACCOUNT
- username
- password
- confirm password
- go

DEFAULT HOMEPAGE
Profile | SEARCH BAR | Go | Random | Mood | Login Create Account
MOVIE CATEGORIES
Display Movies

PROFILE
Profile
username:
reset password!
- old password
- new password
- confirm password | Submit
Favorite List
To-Watch List
movies rated: 2

RESULT HOMEPAGE
Profile | Search BAR | Go | Random | Mood | Login Create Account
MOVIE CATEGORIES
MOVIE TITLE
upvote/downvote
+1 | -1
Picture
Add to watchlist
Add to favorites
Summary
Trailer
Reviews/Rating
Info/Cast
Similar Movies

Both "Homepages"

Note: The user can return to the home page from any page on our website.

Home:

a) If the user is not logged in, they can browse movies based on categories, use the search function (outlined later), login, and create account.

b) Else, if the user is logged in, they can do all of the things outlined above with the addition of checking user-specific data, rating movies, leaving comments on movie entries, and adding to a To-Watch cart. They can also have the option to log out.

c) The search bar is positioned on the upper left of the home page. The user can type in a movie name to search directly. They can also select the "Mood" option which caters to the user's indicated mood (which is a separate input). The last option is an "I'm Feeling Lucky" button that leads to a random chosen movie

d) In general, the home page displays currently trending movies in the user's area. There is also a sidebar full of categories the user can browse through.

Login:
The Login page check's the user's input credentials against the information stored in the database. Successful login leads back to the home page.

Create Account:
The Create Account page allows the user to create a new account. User input is referenced against information stored in db to make sure that the username does not already exist.

Mood:
The "Mood" option below the search bar leads to a separate page that allows the user to select from four moods: happy, sad, bored, and stressed.

Search Results:
A successful search results in a page of recommended movies to watch. If the user searches for a specific movie, or a section of the name, movies with parts of the name show up. If the user searches by mood, corresponding movies show up (eg. top ten comedy for a "Sad" query). The last option, "I'm Feeling Lucky", just returns some random currently trending movies.

Movie:
Clicking on individual movies within Search Results directs the user to a page about that movie. Displayed on each movie's page is the title, a trailer, the director, some actors, a short summary, and average ratings. All of this is retrieved from the Fandango API and Movie DB. Apart from displaying information about the movie, each movie page allows logged in users to rate movies, leave comments, add the movie to a "To Watch" list, and mark the movie as a favorite.
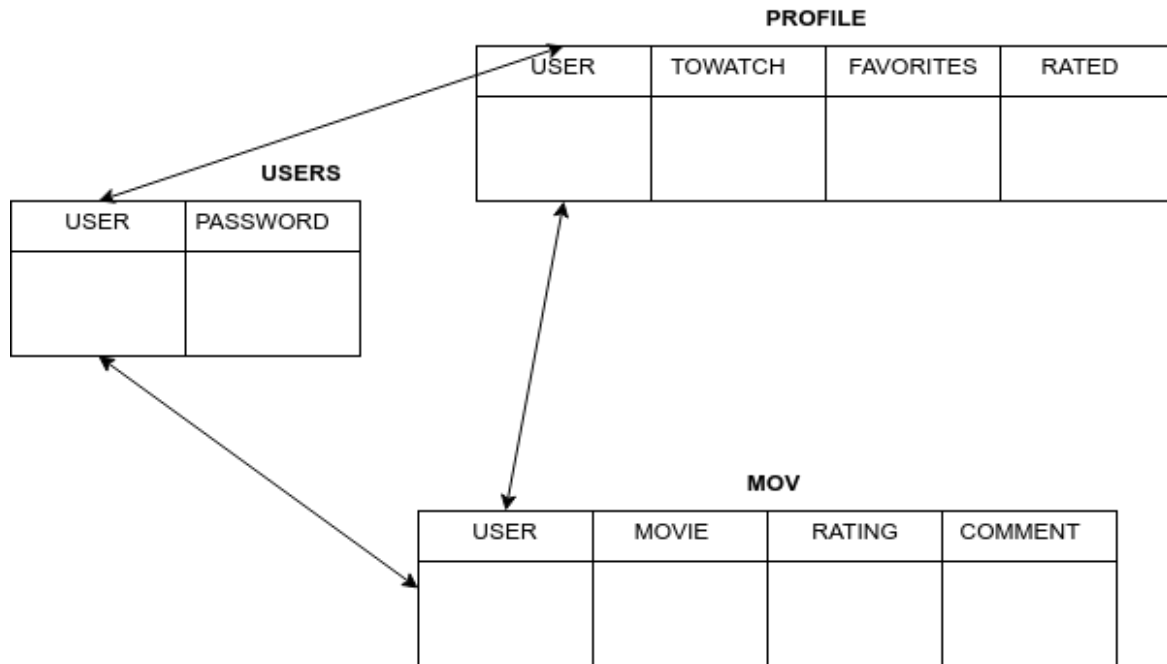
User Profile:
A logged in user can reference their profile from the home page. The User Profile page displays the user's "To Watch" list, the number of movies they have rated, and favorite movies. Users can also reset password from this location.

Currently Running:
Another page accessible from the home page. Using the user's location we show movies running in theaters in their area and their showtimes.

Database Schema:

PROFILE

USER	TOWATCH	FAVORITES	RATED

USERS

USER	PASSWORD

MOV

USER	MOVIE	RATING	COMMENT

USERS:

This table is used to store usernames and hashed passwords upon the creation of an account.

MOV:

This table is updated every time the user leaves a new rating or comment on a movie. Used for a more interactive user experience and to store user's decisions between sessions.

PROFILE:

Used to store lists of favorite movies, movies to watch later, and the number of movies rated so far. Information then displayed on User Profile page and is used to craft the kind of movies shown to the user.

Note: In all cases, movies are identified by their titles and are stored as titles in the appropriate tables.