**RESTFUL DESIGN: Joshua Delfin (101159384) and Joshua Hills ()**

**URL/Pattern:** /

**HTTP Method:** GET

**Description:**

* Opens main page display (index)

**Supported Query Parameters:**

* movie (object) - a movie object/link leading to the specific movie page (/movie)
* person (object) - a person object/link leading to the specific person’s page (/person)
* otheruser (object) - an otheruser object/link leading to the specific user’s page (/otheruser)

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code of 200.
* If text/html is requested: and HTML(PUG) page containing the most popular movie, most popular person, and most popular user is displayed and they link to their own specific pages (/movie, /person, /otheruser)
* If application/json is requested: The objects requested for the page (movie, person, and otheruser).

**URL/Pattern:** /profile

**HTTP Method:** GET

**Description:**

* Opens the user’s profile page displaying profile information
  + Notifications
  + Contributor
  + Watchlist
  + Followed People
  + Followed Users

**Supported Query Parameters:**

* myProfile - json object that uses the me.json file containing all the user information
  + myProfile.name - username
  + myProfile.notifications
  + myProfile.contributor
  + myProfile.watchlist - list containing the user’s watchlist
  + myProfile.followedPeople - list containing the people the user follows
  + myProfile.followedUsers - list containing the people the user follows

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code of 200.
* If text/html is requested: An HTML (PUG) page including the myProfile parameters (name, notifications, contributor, watchlist, followedPeople, followedUsers)
* If application/json is requested: a list of each of the required myProfile parameters

**URL/Pattern:** /otheruser/{otheruserName}

**HTTP Method:** GET

**Description:**

* Opens a different user’s profile page displaying their profile information

**URL Parameters:**

* otherusersName - the unique name of the user  (if it exists)

**Supported Query Parameters:**

* userProfile - json object that uses the otheruser.json file containing all the otheruser information
  + userProfile.name - the other user’s name
  + userProfile.watchlist - the other user’s watchlists
  + userProfile.followedPeople - people the other user is following
  + userProfile.followedUsers - other users the current other user is following

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code of 200.
* If text/html is requestedL An HTML (PUG) page including the other user’s required information (name, watchlist, followedPeople, followedUsers)
* If application/json is requested: object representation of the current otheruser’s information.

**URL/Pattern:** /movie/{movieTitle}

**HTTP Method:** GET

**Description:**

* Opens a movie page with information on that movie (name, actors, directors, reviews etc.)

**URL Parameters:**

* movieTitle - the unique name of the movie (if it exists)

**Supported Query Parameters:**

* myMovie - the json object that uses the movie-data json files containing all the movie data.
  + myMovie.Title - title of the specific movie
  + myMovie.Director - list of the movie directors
  + myMovie.Actors - list of actors in the movie
  + myMovie.Writer - list of the movie writers
  + myMovie.review - list of movie reviews
  + myMovie.plot - description of the movie plot

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code 200.
* If text/html is requested: An HTML (PUG) page including the the movie’s required information (title, director, actors, writer, review, plot) as well as a “add review” prompt.
* If application/json is requested: Object representation of the movie information

**URL/Pattern:** /review

**HTTP Method:** GET

**Description:**

* Opens the review page of a movie and displays that movie’s review

**Supported Query Parameters:**

* myProfile - a json object that uses the me.json file containing the user’s information
* myReview - an object that contains the myProfile.watchlist[“movie”].review[“int”] aka  the specific review for that movie within the watchlist

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code 200.
* If text/html is requested: An HTML (PUG) page including the review
* If application/json is requested: Object representation of the current review/rating

**URL/Pattern:** /person/{personName}

**HTTP Method:** GET

**Description:**

* Opens the person page, displaying the person’s information

**URL Parameters:**

* personName - the unique name of the person (if they exist)

**Supported Query Parameters:**

* myProfile - json object that uses the me.json file to get the user information
* myPerson - an object that contains the myProfile.followedPeople[id] aka the specific person within the followed list.
  + myPerson.topColabs - the person’s top collaborators.
  + myPerson.moviesDirected - a list of movies that the person has directed.
  + myPerson.moviesActed - a list of movies that the person has acted in.
  + myPerson.moviesWritten - a list of movies that the person has written.

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code 200.
* If text/html is requested: An HTML (PUG) page including the the person’s information ( name, topColabs, moviesDirected, moviesWritten.
* If application/json is requested: Object representation of the current person’s information.

**URL/Pattern:** /signin

**HTTP Method:** GET

**Description:**

* Opens the signin/up page displaying the login prompt

**Supported Query Parameters:**

None

**Expected Request Body Data:**

* Username
* Password

**Supported Response Data Type:**

* text/html

**Expected Response (Success):**

* Status code 200
* When text/html is requested: A signin/signup prompt to get the username and password of the user.

**URL/Pattern:** /search

**HTTP Method:** GET

**Description:**

* Opens the search page displaying the search prompt

**Supported Query Parameters:**

None

**Expected Request Body Data:**

* Movie Title
* Movie Genre
* Actor Name

**Supported Response Data Type:**

* text/html

**Expected Response (Success):**

* Status code 200
* When text/html is requested: A search prompt to get movie title, movie genre, actor name.

**URL/Pattern:** /contribute

**HTTP Method:** GET

**Description:**

* Opens the contribution page displaying the “add actor” and “add movie” prompts

**Supported Query Parameters:**

None

**Expected Request Body Data:**

* TO ADD AN ACTOR
  + Actor Name
* TO ADD A MOVIE
  + Movie title
  + Movie runtime
  + Movie release year
  + Movie writers
  + Movie Directors
  + Movie Actors

**Supported Response Data Type:**

* text/html

**Expected Response (Success):**

* Status code 200
* When text/html is requested: two prompts to either add an actor or add a movie

**URL/Pattern:** /searchresults

**HTTP Method:** GET

**Description:**

* Opens the searchresults page displaying the results from the user’s search from the search page

**Supported Query Parameters:**

* Movie

**Expected Request Body Data:**

None

**Supported Response Data Type:**

* text/html
* application/json

**Expected Response (Success):**

* Status code 200
* If text/html is requested: An HTML (PUG) page that display the search results
* If application/json is requested: Object representation of movie list in search results

**URL/Pattern:** /signin/{username}

**HTTP Method:** POST

**Description:**

* Creates or logs into a user account

**URL Parameters:**

* username - a unique valid username, if creating check if username is available, if login, check if username exists and password is valid.

**Expected Request Body Data:**

* username - user’s username
* password - user’s password
* Create account or login (choice)

**Supported Response Data Type:**

* application/json

**Expected Response (Success):**

* Status code 200
* JSON string representing the user password and username (new profile)

**URL/Pattern:** /contribute{actorName}

**HTTP Method:** POST

**Description:**

* Creates a new actor

**URL Parameters:**

* actorName - the unique actor name

**Expected Request Body Data:**

* actorName - requests the user for the actor name to create a new actor

**Supported Response Data Type:**

* application/json

**Expected Response (Success):**

* Status code 200
* JSON string representing the the new actor

**URL/Pattern:** /contribute{movieName}

**HTTP Method:** POST

**Description:**

* Creates a new actor

**URL Parameters:**

* movieName - the unique movie name

**Expected Request Body Data:**

* movieName - requests the user for the movie name to create a new movie
* movieRuntime - requests the new movie’s runtime
* movieRelease - requests the new movie’s release year
* movieWriter - requests the new movie’s writers
* movieDirector - requests the new movie’s directors
* movieActors - requests the new movie’s actors

**Supported Response Data Type:**

* application/json

**Expected Response (Success):**

* Status code 200
* JSON string representing the the new movie and its information