2406 Project RESTful Design

Movies

URL/Pattern: /movies/examplemovieID

HTTP Method: GET

Description:

Retrieves all the public information about the movie and the people that worked on it

Supported Query Parameters: None

Expected Request Body Data:

Supported Response Data Types:

- text/html
- application/json

Expected Response (Success):

- Status Code of 200
- If text/html is requested - An HTML page containing the name of the movie with basic details like runtime, rating, description and a list of similar movies, a list of director(s), writers, and Actors for the movie along with public reviews and an option to add a new review
- If application/json is requested - An object containing the details of the movie including it's writers, directors, actors, reviews and basic details like runtime, rating, and description.

Expected Response (Failure):

- Status Code of 404
- Return an alert stating that the required movie ID is unavailable

---

URL/Pattern: /movies

HTTP Method: GET

Description: Searches the database for movies that match by the provided title, genre, or actor parameter

Supported Query Parameters:

- Title (string) - movie will match if the title value stored in the movie’s object contains the given string value
- Genre (string) - movie will match if the genre value stored in the movie’s object contains the given string value
- Actor (string) - movie will match if an actor’s stored string from a movie object’s array of actors matches the given string value

Expected Request Body Data:

Supported Response Data Types:

- text/html
- application/json

Expected Response (Success):

- Status Code of 200
- text/html - Returns a HTML page displaying a list of movie names upto 20 names that best matches the title, genre, or/and actor parameters in order along with an option to click on that movie and go to it's page. Also an option to navigate to the next page containing upto 20 more names if there are that many matches and so on
- application/json - Returns an object containing a lost of movie names and movie IDs that best matches the title, genre, or/and actor parameters

Expected Response (Failure):

- Status Code of 404
- Returns message that says no matches were found for movies with the submitted parameters

---

URL/Pattern: /movies/addMovie

HTTP Method: POST

Description: Adds a movie to the database when a contributing user chooses to

Supported Query Parameters:

Expected Request Body Data:

- application/json

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200

- Updates the database of movies by adding all the details provided by the user which is requested over in json format from the input text fields
- Displays a message on the contribute page stating that the movie has been added to the database

Expected Response (Failure):

- Status Code of 404

People

URL/Pattern: /people/examplepersonID

HTTP Method: GET

Description:

Retrieves all the public information of the person and their work history in movies

Supported Query Parameters:

Expected Request Body Data: None

Supported Response Data Types:

- text/html
- application/json

Expected Response (Success):

- Status Code of 200
- If text/html is requested - An HTML page containing the name of the person, list of people the person has collaborated with, a list of movies written by the person, a list of movies directed by the person, and a list of movies acted in by the person
- If application/json is requested - An object representing the person with all of their details like collaborations, movies written, movies directed, and movies acted in

Expected Response (Failure):

- Status Code of 404
- Return an alert stating that the required person ID is unavailable

---

URL/Pattern: /people/addPerson

HTTP Method: POST

Description:

Adds a person when a contributing user chooses to

Supported Query Parameters:

Expected Request Body Data:

- application/json

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Updates the database of people by adding all the details provided by the user about the person to be added which is requested over in json format from the input text fields

Expected Response (Failure):

- Status Code of 404
- Returns an alert stating that a person with the same name exists and can't be added again

Users

URL/Pattern: /users/otheruserID

HTTP Method: GET

Description:

Retrieves all the public information regarding movie preferences of the selected user, after sorting if the ID people or user

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html
- application/json

Expected Response (Success):

- Status code of 200
- If text/html - Displays a HTML page containing the User's name, account type, a list of all the reviews the user has made, a list of people the user follows with an option to follow each of those people, and the user's watchlist
- If application/json - Returns an user object containing the User's ID, name, account type, list of people they're following, user's watchlist of movies and user's list of recommended movies

Expected Response (Failure):

- Status code of 404

- Return an alert stating that the required user ID is unavailable

---

URL/Pattern: /user/follow

HTTP Method: POST

Description: Follow another user

Supported Query Parameters:

- ID - will add the queried id to the list of follows in user object, after sorting if the ID is for people or user

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Adds the queried user to the following list of the original user object using the queried ID
- text/html - Changes the follow button to unfollow for the user that was just followed

Expected Response (Failure):

- Status code of 404
- Returns a message that the user ID to be added is invalid

URL/Pattern: /user/unfollow

HTTP Method: PUT

Description: Unfollow an user

Supported Query Parameters:

- ID - will remove the queried id from the list of follows in user object, after sorting if the ID is for people or user

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200

- Removes the queried user to the following list of the original user object using the queried ID
- text/html - Changes the follow unfollow to follow for the user that was just unfollowed

Expected Response (Failure):

- Status code of 404
- Returns a message that the user ID to be removed is invalid

---

URL/Pattern: /user/addMovie

HTTP Method: POST

Description: Add a movie to the user's watch list

Supported Query Parameters:

- ID - The queried ID will be used to add the movie to the watchlist in the user object

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Adds the movie to the user's watchlist
- text/html - Changes the add to watchlist button to remove from watchlist for the movie just added to user's watchlist

Expected Response (Failure):

- Status code of 404

`

URL/Pattern: /user/removeMovie

HTTP Method: PUT

Description: Remove a movie from the user's watch list

Supported Query Parameters:

- ID - The queried ID will be used to remove the movie from the watchlist in the user object

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Removes the movie from the user's watchlist
- text/html - Changes the remove from watchlist button to add to watchlist for the movie just removed from the user's watchlist

Expected Response (Failure):

- Status code of 404

Accounts

URL/Pattern: /profile/my

HTTP Method: GET

Description:

Retrieves all the information of the user who is logged in

Supported Query Parameters:

Expected Request Body Data: None

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- If text/html is requested - An HTML page containing the account type of the profile, a list of people they follow, a list of users they follow, a list of movies they've watched and a list of recommended movies

---

URL/Pattern: /profile/my/typeConfig

HTTP Method: PUT

Description:

Updates the user's account type according to the user's preference

Supported Query Parameters:

Expected Request Body Data: None

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- text/html - The user's profile gets updated and reflects the change in the user object's account type to what the user requested through this route

Expected Response (Failure):

- Status code of 404

URL/Pattern: /profile/my/removeNotif

HTTP Method: PUT

Description:

Updates the user's account and removes the selected notification from the user's list of notifications

Supported Query Parameters:

Expected Request Body Data: None

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- text/html - The user's profile gets updated and reflects the change in the user notifications and no longer shows the removed notification

---

URL/Pattern: /sign/in

HTTP Method: GET

Description: Loads the sign in page

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- text/html - Provides the user a HTML page with text fields to enter their username and password to sign into the website

URL/Pattern: /sign/up

HTTP Method: GET

Description: Loads the sign up page

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- text/html - Provides the user a HTML page with text fields to enter their first name, last name, username, and password to sign up for the website and create their own user

---

URL/Pattern: /sign/up?

HTTP Method: GET

Description: Loads the sign up page

Supported Query Parameters:

- username(string) - Has to be an unique username that is not already registered
- password(string) - The password that the user will use to login to the site
- fName(string) - The first name of the user
- lName(string) - The last name of the user

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Makes a get request to the new user's profile, GET /profile/my

Expected Response (Failure):

- Status code of 404
- Shows an alert stating that the credentials entered may already be in use, in case of a username or that the password entered is inappropriate

URL/Pattern: /sign/in?

HTTP Method: GET

Description: Loads the sign up page

Supported Query Parameters:

- username (string) - A registered username has to be entered in order to sign in
- password (string) - Corresponding password to the username has to be entered to complete sign in

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- Makes a get request to the user's profile, GET /profile/my

Expected Response (Failure):

- Status code of 401
- Shows an alert stating that the user has entered incorrect credentials

URL/Pattern: /contribute

HTTP Method: GET

Description: Loads the contribute page

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- text/html - Provides the user a HTML page to add either a new actor or a new movie if the user has a contributing account type

Search

URL/Pattern: /search

HTTP Method: GET

Description: Loads the static search page

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- text/html - Provides the user a HTML page to search for movies on the database by title, genre and/or actor

Reviews

URL/Pattern: /reviews/examplereviewID

HTTP Method: GET

Description: Displays a complete review including who it was made by, about which movie and further necessary details such as score, brief review or full review text

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html
- application/json

Expected Response (Success):

- Status Code of 200
- If text/html - An entire html page displaying a complete review including which user it was made by, about which movie it was made for and all necessary details like score, brief review or full review text
- If application/JSON - returns a review object containing all the details including what movie the review is for and by which user

Expected Response (Failure):

- Status code of 404
- Return an alert stating that the required review ID is unavailable

URL/Pattern: /reviews/addReview

HTTP Method: POST

Description: Adding a review made to a movie by a user

Supported Query Parameters:

Expected Request Body Data:

Supported Response Data Types:

- text/html

Expected Response (Success):

- Status Code of 200
- A Review object is created and the unique id from the review object is added as a string to the movie object's array of reviews
- text/html - The movie page is updated and displays the new review added

Expected Response (Failure):

- Status code of 404