Homework 6

CSCI 571 Spring 2021

Server-side Scripting with Flask, JSON, and TMDB API

1. Objectives

- Get experience with the **Python** programming language and the **Flask** framework.
- Get experience creating web pages using **HTML, CSS, JavaScript, DOM, JSON** and the **XMLHttpRequest** object.
- Get experience with making third-party API calls (The Movie Database API) from your web application.
- Getting hands-on experience in deploying a Flask web application on **Azure**.

1.1 Cloud Exercise

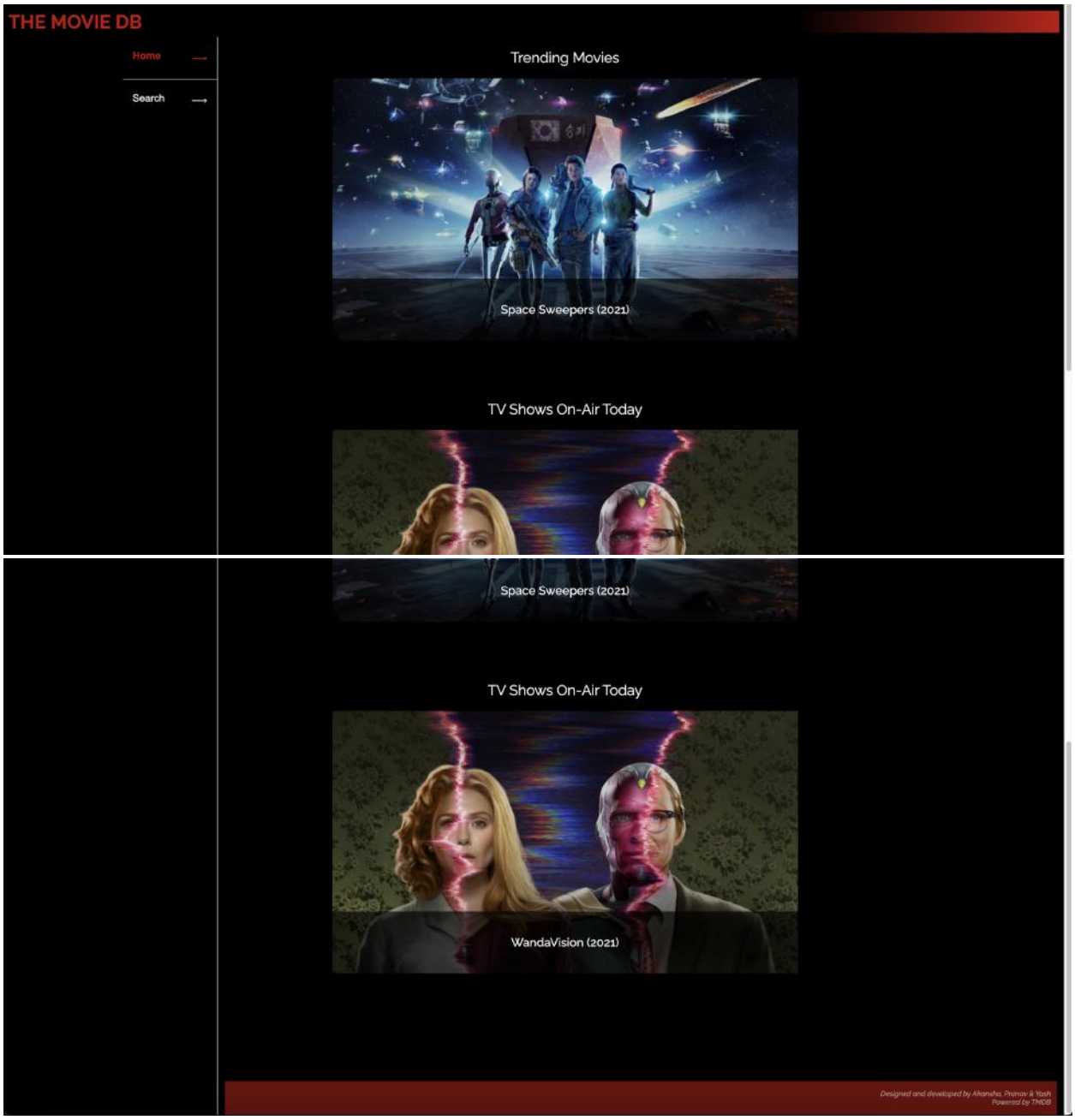
- The backend of this homework must be implemented on the cloud on GCP, AWS, or Azure using Python.
- See homework 5 for installation of either one of these platforms. You only have to select one platform to implement your backend.
- You must refer to the grading guidelines, the video, the specs, and Piazza. Styling will be graded and the point's breakup is mentioned in the grading guidelines.
- The link for the video is - https://youtu.be/o9G33ck4mQE

2. Description

In this exercise, you are asked to create a web page that allows you to search for information regarding movies and TV shows using the TMDB API (This is probably one of the most exciting themes to be chosen for CSCI 571!), and the results will be displayed on cards below the search query. Upon clicking a button in the card, a modal will pop up and display more information about that selected movie/TV show.

2.1 Home Tab

On entering the home page of your web application, the following is displayed:

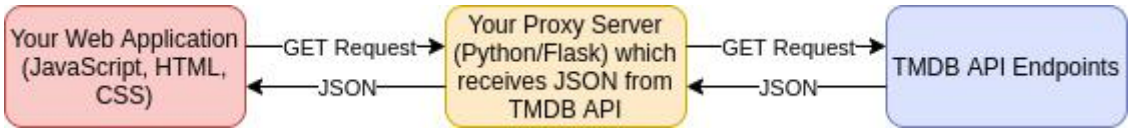


On the left, two tabs are used to toggle between the Home tab and the Search tab. On the home page, there are two sections: the Trending Movies slideshow and the TV Airing Today slideshow. The image sizes are w780. Repeating the slideshow after it has been played once is optional.

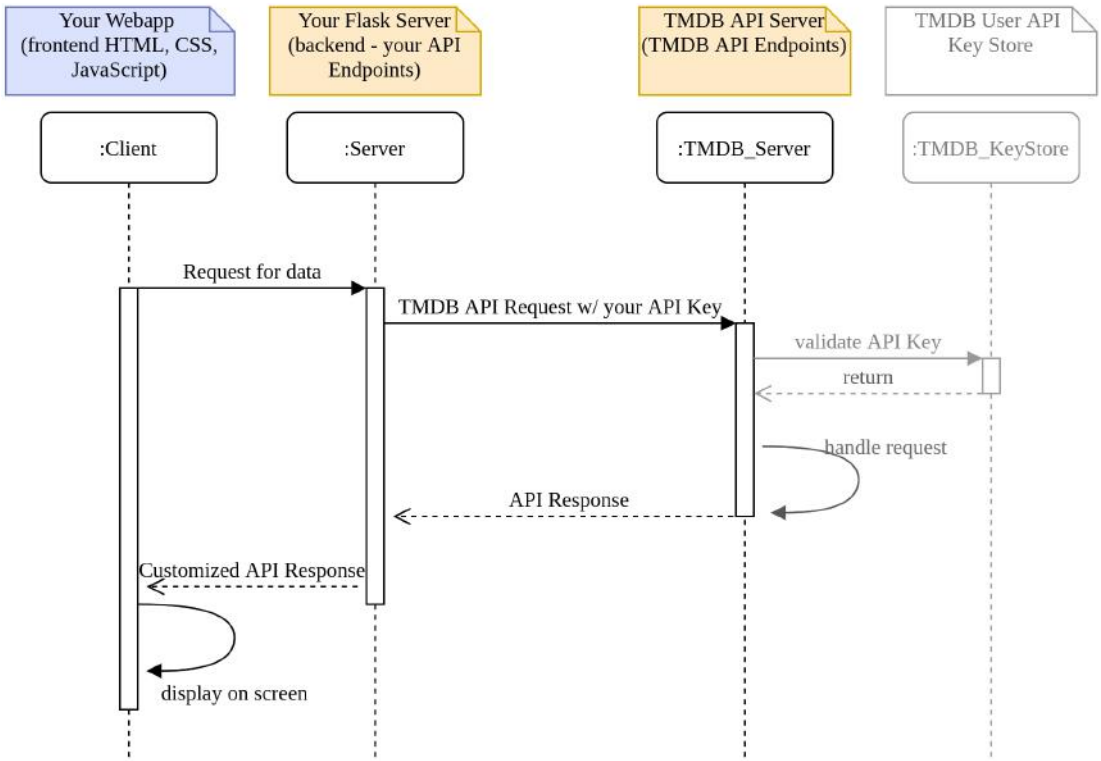
If we are in the home section, the text “Home” and the arrow next to it will have its text color red (shade of red shown in the image). If we are in the Search section, then the text “Search” and the arrow next to it will be red. When the webpage first loads, Home should be in red.

2.1.1 TMDB Trending Endpoint

This endpoint will be used to get information about trending movies. To receive data, you must use **GET** to send a request to your web server (do not use POST, as you would be unable to provide a sample link to your cloud services). A Python script running a Flask server (your proxy server) will receive the request. A proxy server is a server application that acts as an intermediary for requests from clients seeking resources from servers that provide those resources. A proxy server thus functions on behalf of the client when requesting service, potentially masking the true origin of the request to the resource server. Proxy servers are also used to modify the results received from the actual server and change them before sending them back to the web application.



The sequence diagram below shows a sample API request to a third-party API service such as TMDB. The client (the JavaScript in our webpage) makes a **GET** request to our Flask server written in Python. Once the Flask server receives the request, it constructs and makes another **GET** request to the TMDB API Server. The TMDB API server validates the request by verifying our API key against its key store (a table of registered API keys) and sends back a valid response if the key is an authorized key. This response is received by our Python server in the form of a **JSON formatted object**. When we receive this response, we manipulate the JSON object according to our needs and finally send to the client the modified JSON object.



You need to use the Flask Python framework to make an API call to the TMDB Trending Endpoint. **Using XMLHttpRequest or any other JavaScript calls for anything other than calling your own "cloud" backend will lead to a 4-point penalty.** Do not call the TMDB API directly from JavaScript.

Define routing endpoints and make your TMDB API calls from the Python backend. The recommended tutorial for Flask and more importantly, routing, can be found at the following link: **Welcome to Flask — Flask Documentation (1.1.x).** MDN has an excellent introductory article on server-side web frameworks which can be found **here**.

API Sample:
https://api.themoviedb.org/3/trending/**{media_type}**/**{time_window}**?***api_key*=<<*api_key*>>**

This API endpoint has three parameters that we have to supply to construct the request URL:

1. ***media_type***- this is the type of media for which we want trending items- use ***movie***.
2. ***time_window***- this is the time window within which we want trending items- use ***week.***
3. ***api_key*** - this is the API KEY that you create as described in Section 3.

API Example:
https://api.themoviedb.org/3/trending/movie/week?api_key=97588ddc4a26e3091152aa0c9a40de22

The response received by this Python request is a **JSON-formatted object**. The figure below shows a sample response received from the request. You need to **parse this JSON object and extract some fields as required**.


```
{
   page: 1,
  - results: [
     - {
          original_language: "ko",
          original_title: "승리호",
          poster_path: "/bmemsraCG1kIthY74NjDnnLRT2Q.jpg",
          video: false,
          vote_average: 6.9,
          overview: "In the year 2092, space is full of dangerous floating garbage like discarded satellites and deserted spaceships. The crew of a space
          junk collector ship called The Victory discovers a humanoid robot that's known to be a weapon of mass destruction. They get involved in a risky
          business deal and travel through space looking for garbage they can make money off of while also competing with rival junk collectors.",
          release_date: "2021-02-05",
          vote_count: 77,
          title: "Space Sweepers",
          adult: false,
          backdrop_path: "/drulhSX7P5TQlEMQZ3JoXKSDEfz.jpg",
          id: 581389,
        - genre_ids: [
              18,
              14,
              878
          ],
          popularity: 145.638,
          media_type: "movie"
      },
     - {
          adult: false,
          backdrop_path: "/srYya1ZlI97Au4jUYAktDe3avyA.jpg",
        - genre_ids: [
              14,
              28,
              12
          ],
          vote_count: 3420,
          original_language: "en",
```


You will need to collect **only 5** trending results, and for each result, extract only:

1. ***title*** - the title of the movie.

2. ***backdrop_path*** - the path for the image of the movie.
3. ***release_date*** - the date the movie was released.

After you have extracted these fields for 5 results, send them back to your web application to display them on the screen.

2.1.2 TMDB TV Airing Today Endpoint

Similar to the TMDB Trending Endpoint, this endpoint will be used to get information about TV shows airing today.

API Sample:
https://api.themoviedb.org/3/tv/airing_today?**api_key=<<*api_key*>>**

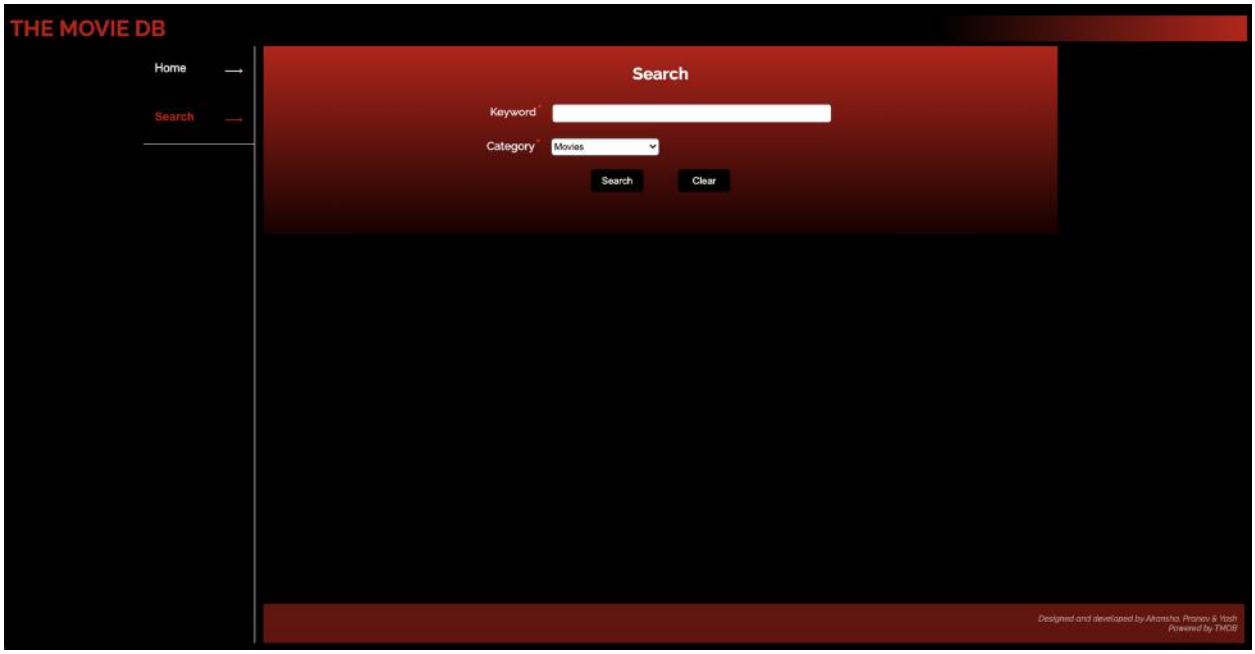
API Example:
https://api.themoviedb.org/3/tv/airing_today?api_key=97588ddc4a26e3091152aa0c9a40de22

You will need to collect **only 5** results, and for each result, extract only:

1. ***name*** - the name of the show.
2. ***backdrop_path*** - the path for the image of the show.
3. ***first_air_date*** - the date the show was first aired.

2.2 Search Tab

On clicking the search tab, you are presented with the following screen.



There are two input fields:

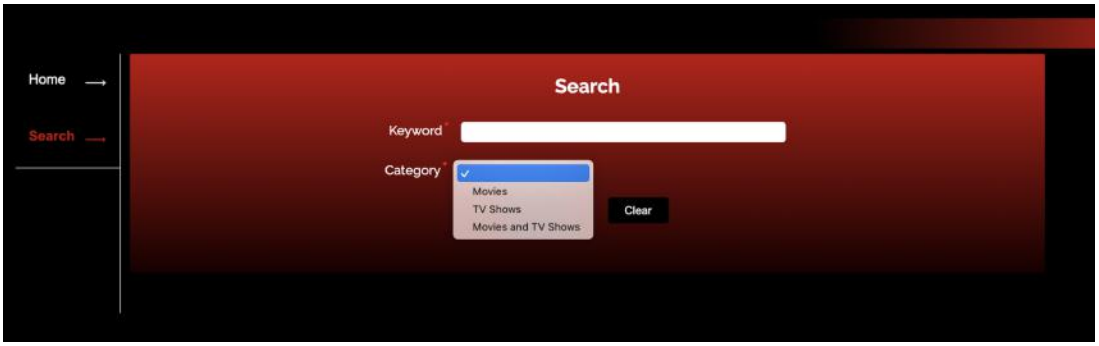
1. Search query
2. Search Category (select one of *Movies, TV Shows,* or *Movies and TV Shows*)

There are two buttons:

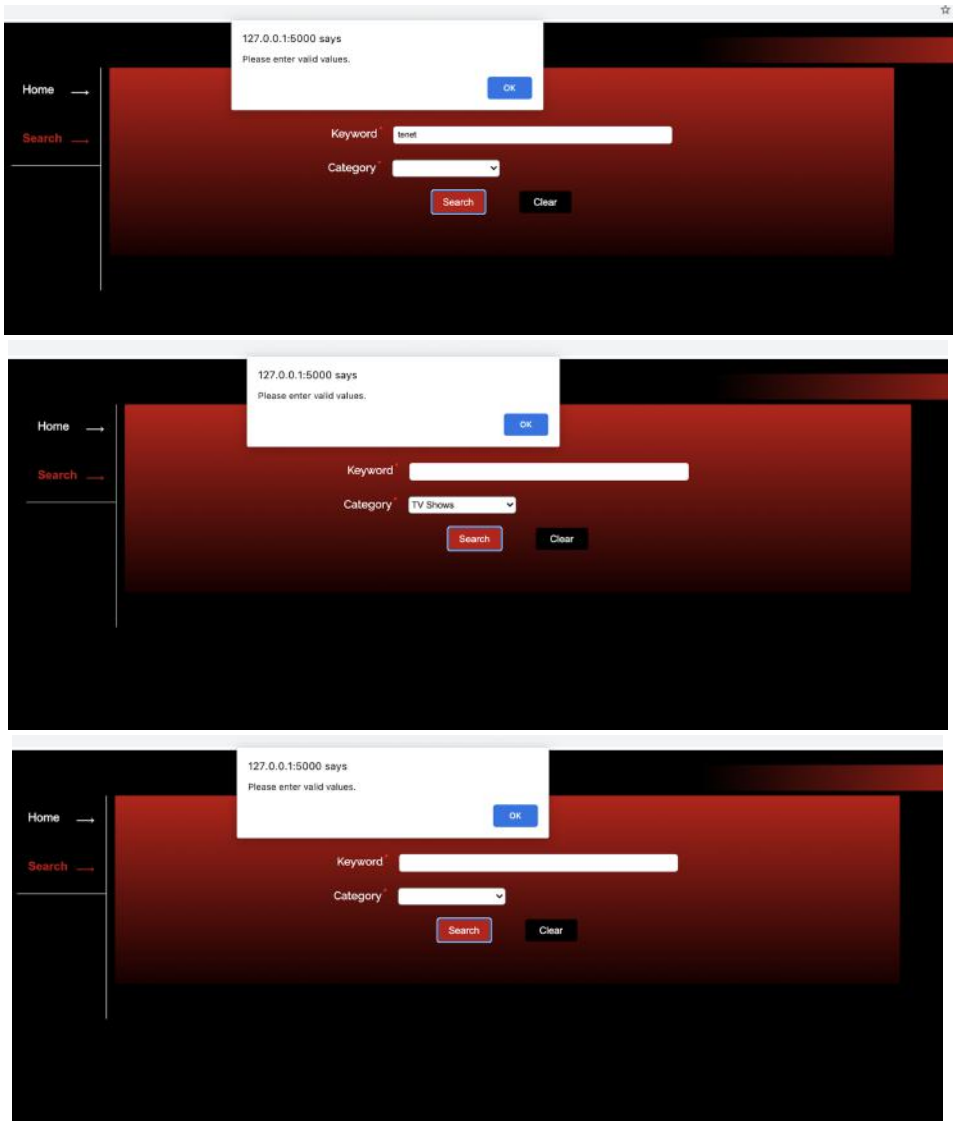
1. Search Button
2. Clear Button

2.2.1 Search Button

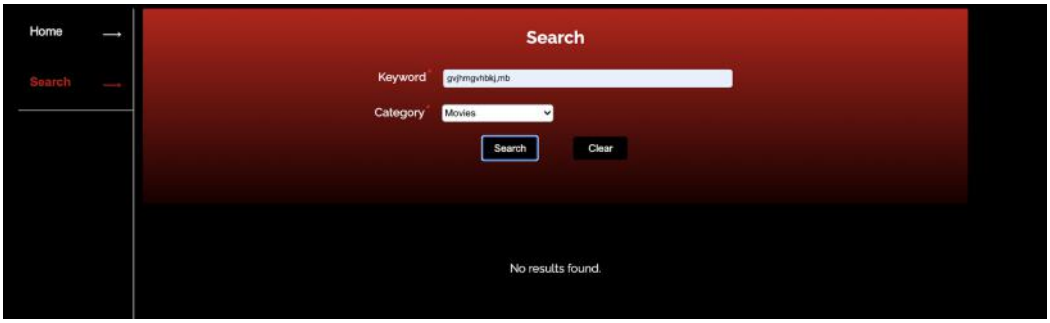
On clicking the search button after entering a valid search query and selecting one of the three options in the search category, several cards are displayed below that match the search query's results.



If the user clicks on the Search button without providing a value in the field, or without selecting a category, an alert should alert "Please fill out this field" (examples shown below).

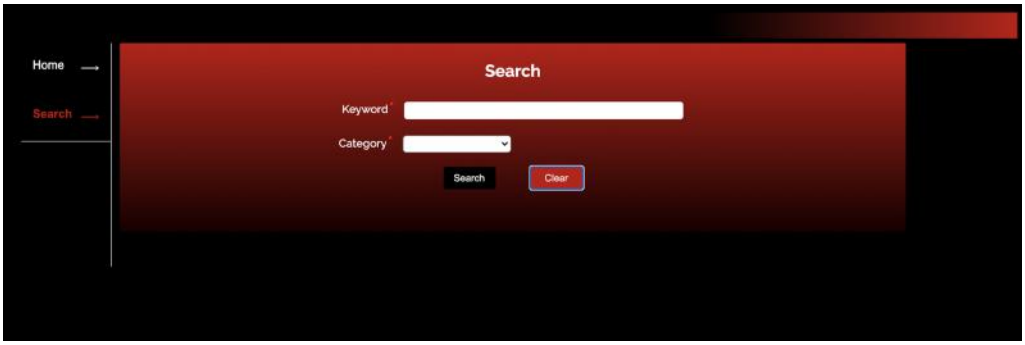


If the user clicks on search with an invalid or garbled search query, then a message should be displayed saying “No results found.” as shown below:



2.2.2 Clear button

This button must clear the result area (below the search area) and the search field without reloading the page.



The Search and Clear buttons should have their background color as black and text color as white. When we hover over either of these buttons, the background should change to a share of red (as shown in the image).

2.3 Displaying Search Results

In this section, we outline how to use the form data to construct the calls to the RESTful web services of the TMDB API and display the results on the web page. To get this information, we use the TMDB Search Endpoints. We use the data returned from these endpoints to populate the movie cards with information.

2.3.1 Search Movie Endpoint to search only for movies:

From this endpoint, you will get a lot of information about the movies that relate to your search query.

API Sample:
https://api.themoviedb.org/3/search/movie?**api_key**=**<<*api_key*>>**&**query**=**<<*search_query*>>**&language=*en-US*&page=*1*&include_adult=*false*

The process to create your API key is explained in Section 3. When constructing the python request, you will need to provide two values in the URL:

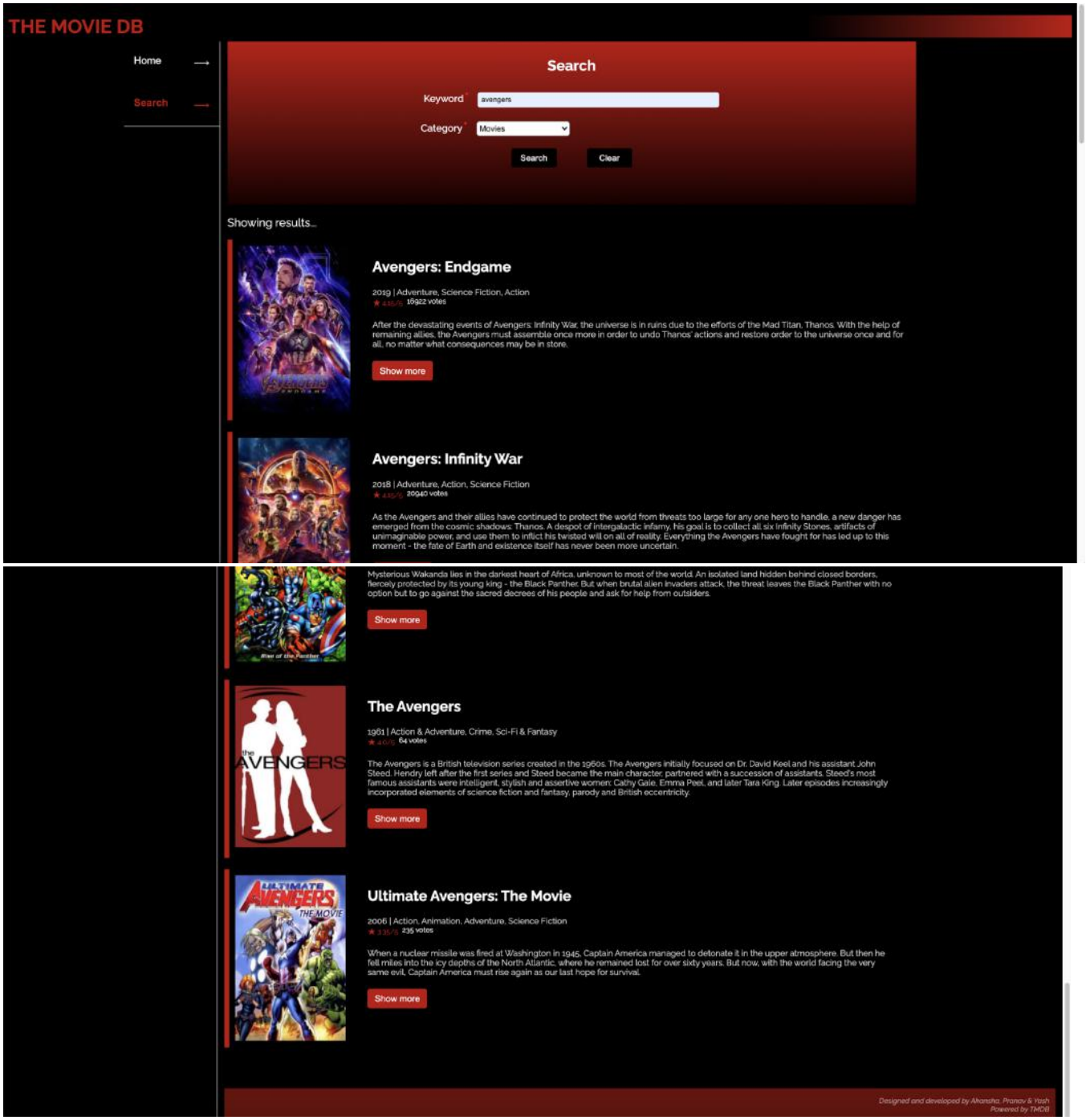
1. The first value, the ***search_query***, is the text entered for the search query, with spaces replaced with '**%20**'.
2. The second value, ***api_key*** is the API KEY that you create as described in Section 3.

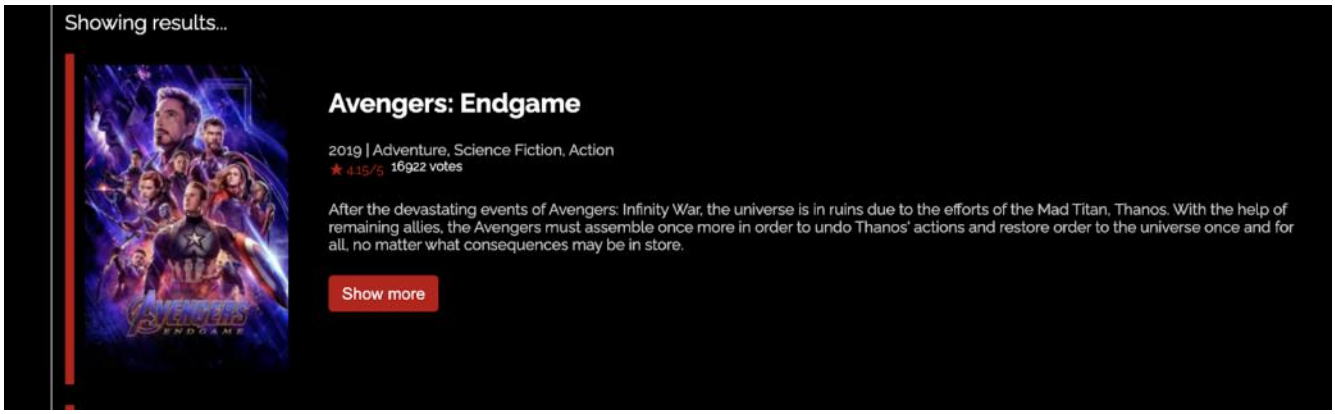
API Example:
https://api.themoviedb.org/3/search/movie?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US&query=avengers&page=1&include_adult=false

For each movie, you will only need these fields:

1. ***id*** - the ID of the movie

2. ***title*** - the title of the movie
3. ***overview*** - the synopsis of the movie.
4. ***poster_path*** - path for the image of the poster.
5. ***release_date*** - the release date of the movie.
6. ***vote_average*** - the average of all the ratings for the movie.
7. ***vote_count*** - count of the number of ratings received for the movie.
8. ***genre_ids*** - genres of the movie.





2.3.2 Search TV Endpoint to search only for TV shows:

From this endpoint, you will get a lot of information about the shows that relate to your search query.

API Sample:
https://api.themoviedb.org/3/search/tv?**api_key**=<<*api_key*>>&language=en-US&page=1&**query**=<<*search_query*>>&include_adult=false

API Example:
https://api.themoviedb.org/3/search/tv?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US&page=1&query=game%20of&include_adult=false

The JSON response received from the endpoint for the query “game of” is:


```
{
   page: 1,
  - results: [
     - {
          backdrop_path: "/suopoADq0k8YZr4dQXcU6pToj6s.jpg",
          first_air_date: "2011-04-17",
        - genre_ids: [
              10765,
              18,
              10759,
              9648,
              10768
          ],
          id: 1399,
          name: "Game of Thrones",
        - origin_country: [
              "US"
          ],
          original_language: "en",
          original_name: "Game of Thrones",
          overview: "Seven noble families fight for control of the mythical land of Westeros. Friction between the houses leads to full-scale war. All while a very ancient evil awakens in the farthest north. Amidst the war, a neglected military order of misfits, the Night's Watch, is all that stands between the realms of men and icy horrors beyond.",
          popularity: 403.21,
          poster_path: "/u3bZgnGQ9T01sWNhyveQz0wH0Hl.jpg",
          vote_average: 8.4,
          vote_count: 12801
      },
     - {
          backdrop_path: "/35szhk4sNfjOuomIcr6KOHzUR0x.jpg",
          first_air_date: "2016-04-12",
        - genre_ids: [
              18
          ],
          id: 64482,
          name: "Game of Silence",
        - origin_country: [
```

For each TV show, you will only need these fields:

1. ***id*** - the ID of the TV Show
2. ***name*** - the name of the TV Show
3. ***overview*** - the synopsis of the TV Show.
4. ***poster_path*** - path for the image of the poster.
5. ***first_air_date*** - the release date of the TV Show.
6. ***vote_average*** - the average of all the ratings for the TV Show.
7. ***vote_count*** - count of the number of ratings received for the TV Show.
8. ***genre_ids*** - genres of the TV Show

THE MOVIE DB

Home →

Search →

Search

Keyword* game of

Category* TV Shows

Search Clear

Showing results...

Game of Thrones

2011 | Sci-Fi & Fantasy, Drama, Action & Adventure, Mystery, War & Politics
★ 4.2/5 12845 votes

Seven noble families fight for control of the mythical land of Westeros. Friction between the houses leads to full-scale war. All while a very ancient evil awakens in the farthest north. Amidst the war, a neglected military order of misfits, the Night's Watch, is all that stands between the realms of men and icy horrors beyond.

Show more

Game of Silence

2016 | Drama
★ 2.5/5 30 votes

A successful Atlanta attorney's long-lost childhood friends unexpectedly reappear after 25 years. When a dark secret they thought they'd buried resurfaces, the brotherhood bands together to right the wrongs of their shared past - a journey that will push the limits of their loyalty and quench their thirst for revenge.

Show more

Game of Stones follows a team of gemologist around the world and gives you a behind the scenes look at where the gems and the jewlery that you wear comes from, how its made, hows it's bought, and the lengths these guys go through to get their hands on the finest gems on the planet.

Show more

A Game of Murder

1966 |
★ 0.0/5 0 votes

The once famous athlete Bob Kerry dies mysteriously on a golf course. His son Jack, a detective, suspects murder and decides to investigate the case. He gets caught up in a web of intrigue set in the Soho red-light district in London, with its pimps and prostitutes.

Show more

NFL Game of the Week

2015 |
★ 1.6/5 2 votes

The NFL Films Game of the Week, formerly known as the NFL Game of the Week, is a program that airs on NFL Network, the official television channel of the National Football League. On this show, NFL Films compresses one or two NFL games from the previous week into a one-hour program.

Show more

Designed and developed by Akansha, Pranav & Yash
Powered by TMDB

Showing results...

Game of Thrones

2011 | Sci-Fi & Fantasy, Drama, Action & Adventure, Mystery, War & Politics
★ 4.2/5 12845 votes

Seven noble families fight for control of the mythical land of Westeros. Friction between the houses leads to full-scale war. All while a very ancient evil awakens in the farthest north. Amidst the war, a neglected military order of misfits, the Night's Watch, is all that stands between the realms of men and icy horrors beyond.

Show more

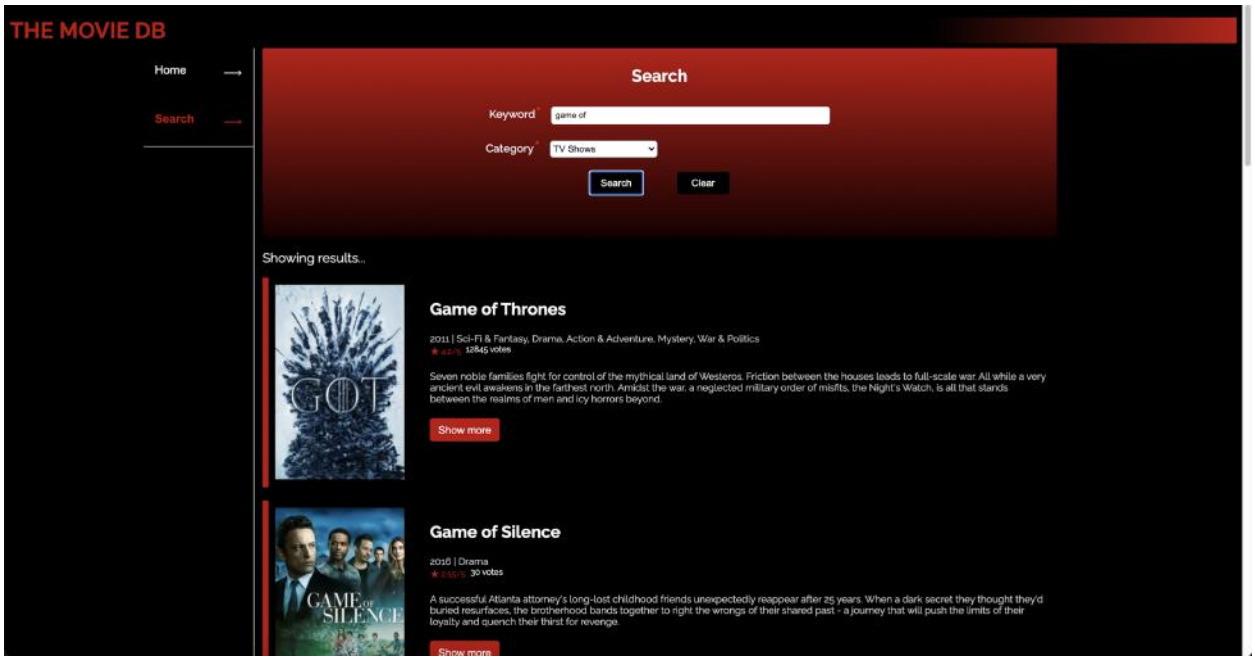
2.3.3 Multi-Search Endpoint to search for both Movies and TV Shows:

From this endpoint, you will get a lot of information about the shows, movies, and people that relate to your search query. You will have to filter out the people from the list in your Python Backend before sending the JSON back to your web application.

For each object that is returned, find out if it is a movie, show, or a person through the ***media_type*** property returned for each object. If it is a movie or a tv show, collect it, otherwise, ignore it.

API Sample:
https://api.themoviedb.org/3/search/multi?**api_key**=**<<*api_key*>>**&language=en-US&**query**=**<<*search_query*>>**&page=1&include_adult=false

API Example:
https://api.themoviedb.org/3/search/multi?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US&query=game%20of&page=1&include_adult=false



For a movie, collect the same fields as shown in 2.3.1. For a TV Show, collect the same fields as shown in 2.3.2.

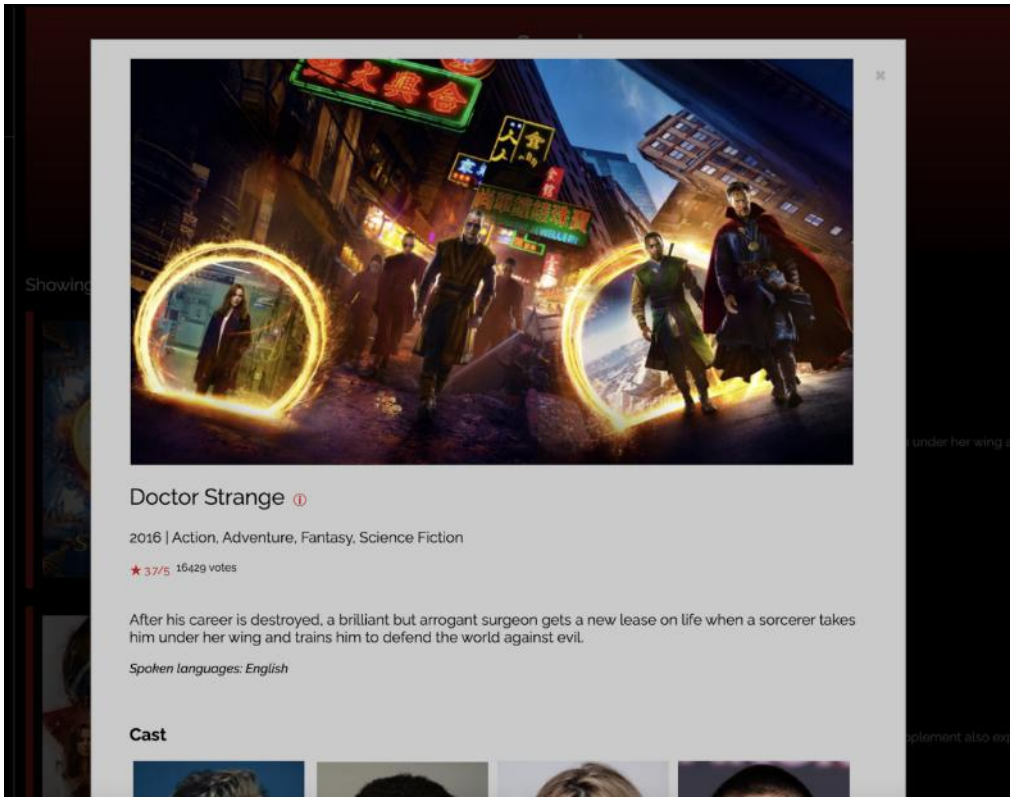
2.4 Displaying Show More Popup

On clicking **show more** in each search listing, a popup opens up and more details about the movie or TV show is displayed to the user. We will need the following API endpoints to get more information about a movie/tv show. On clicking the “x” button at the top-right of the popup, the popup will close.

2.4.1 Get Movie Details Endpoint

When constructing the python request, you will need to provide two values in the URL:

1. The first value, the ***movie_id***, is the ***id*** of the movie received through the Search Endpoint.
2. The second value, ***api_key*** is the API KEY that you create as described in Section 3.



API Sample: https://api.themoviedb.org/3/movie/**{movie_id}**?**api_key=<<*api_key*>>**&language=en-US

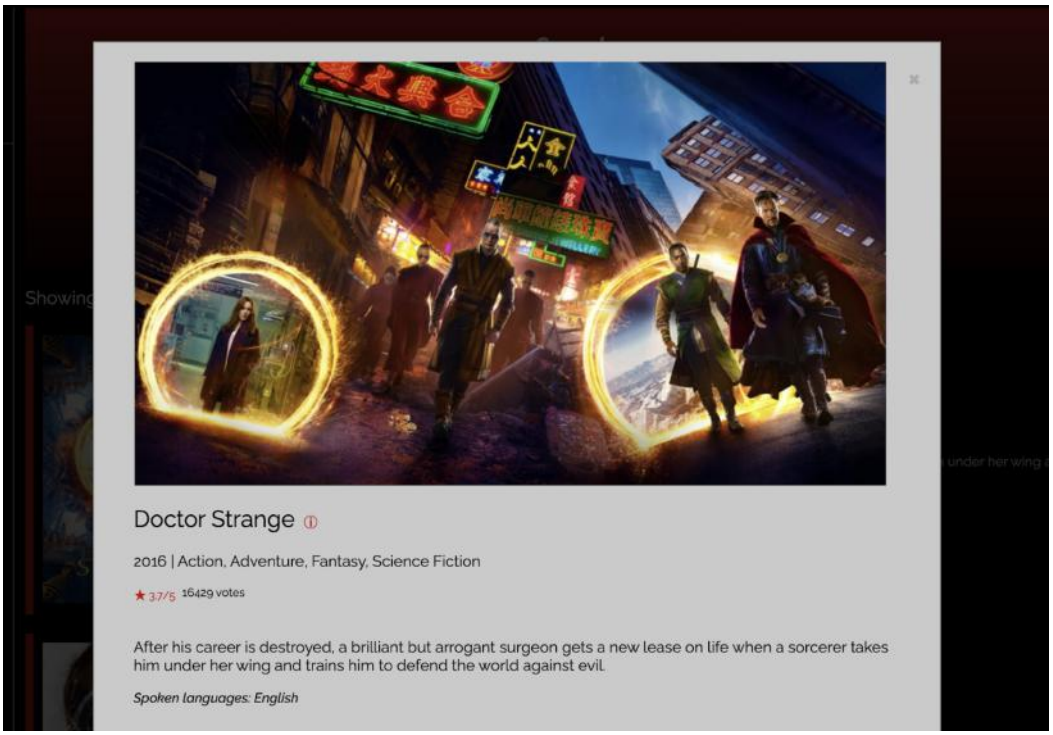
API Example:
https://api.themoviedb.org/3/movie/284052?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US

You will only need these fields:

1. ***id*** - the id of the movie.
2. ***title*** - the title of the movie.
3. ***runtime*** - The runtime of the movie.
4. ***release_date*** - the date the movie was released.
5. ***spoken_languages*** - different audio languages the movie is available in.
6. ***vote_average*** - the average of ratings given by reviewers.
7. ***vote_count*** - total number of reviews received by the movie.
8. ***poster_path*** - path for the image of the poster.
9. ***backdrop_path*** - path for the larger backdrop image.

10. ***genres*** - genres of the movie.

There needs to be an Information button as indicated next to the movie's title. On clicking this button, the TMDB website of this movie will open up. The format of this link is: https://www.themoviedb.org/movie/284052 where 284052 is the id of the movie. Here, the image size being used is w780.



2.4.2 Get Movie Credits Endpoint

API Sample:
https://api.themoviedb.org/3/movie/**{movie_id}**/credits?**api_key=<<*api_key*>>**&language=en-US

API Example:
https://api.themoviedb.org/3/movie/284052/credits?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US

You will only need details of **at most 8** actors. For each actor, only pick these details:

1. ***name*** - Name of the actor
2. ***profile_path*** - Path for the image of the actor
3. ***character*** - The character played by the actor.

For each actor, you display their profile picture, the actual name followed by their character name. All the names need to be in one line. So if any name goes beyond one line, then it should end in an ellipse (...). You can see this for Stephen Strange/Doctor Strange.

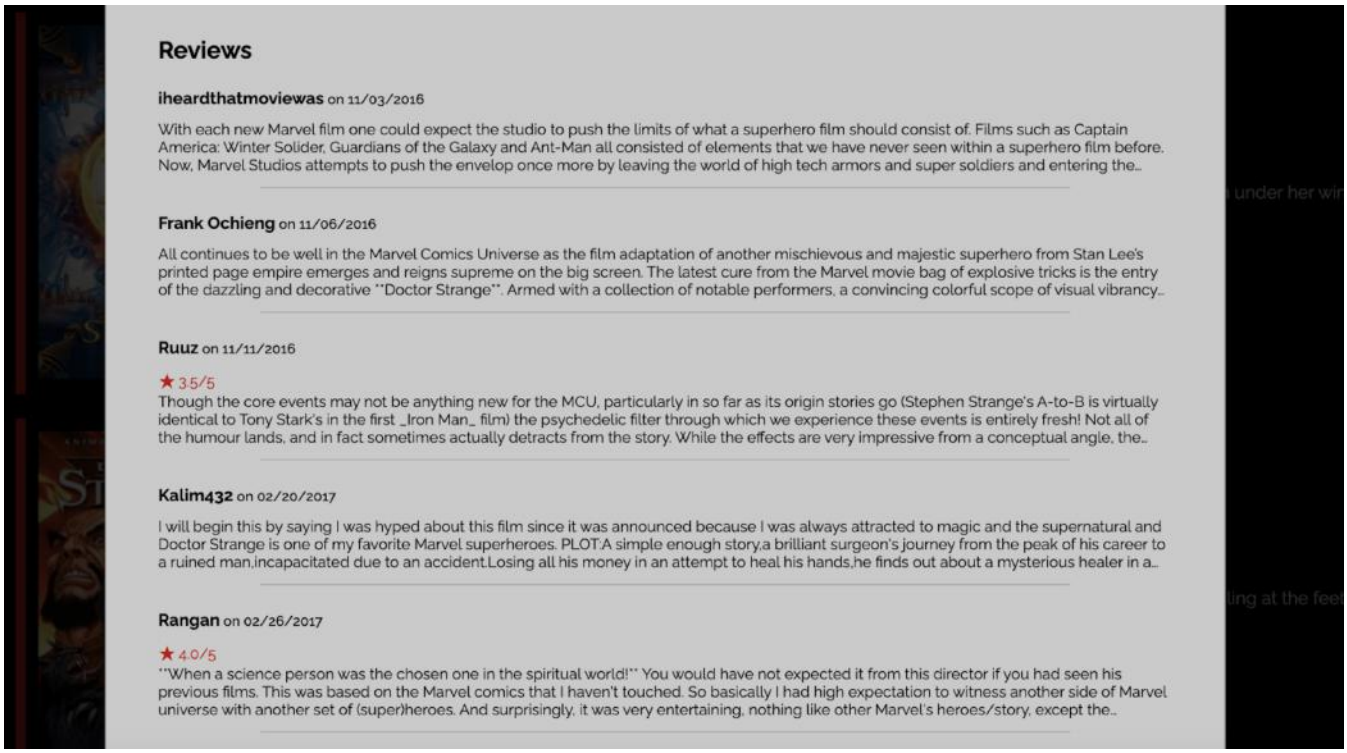
2.4.3 Get Movie Reviews Endpoint

API Sample:
https://api.themoviedb.org/3/movie/**{movie_id}**/reviews?**api_key=<<*api_key*>>**&language=en-US&page=1

API Example:
https://api.themoviedb.org/3/movie/284052/reviews?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US&page=1

You will need details of **at most 5 reviews** for each movie. For each review, only pick these details:

1. ***username*** - The username of the reviewer, inside the author_details object.
2. ***content*** - The content of the review
3. ***rating*** - The rating given by the reviewer for the movie, inside the author details object.
4. ***created_at*** - The date the review was created.



The content of the reviews should be at most 3 lines. So if any content goes beyond 3 lines, then it should end in an ellipse (...).
If a review doesn't have the rating value (is null), then don't display the rating for that review, just the comment if any.

2.4.4 Get TV Show Details Endpoint

API Sample: https://api.themoviedb.org/3/tv/**{tv_show_id}**?**api_key=<<*api_key*>>**&language=en-US

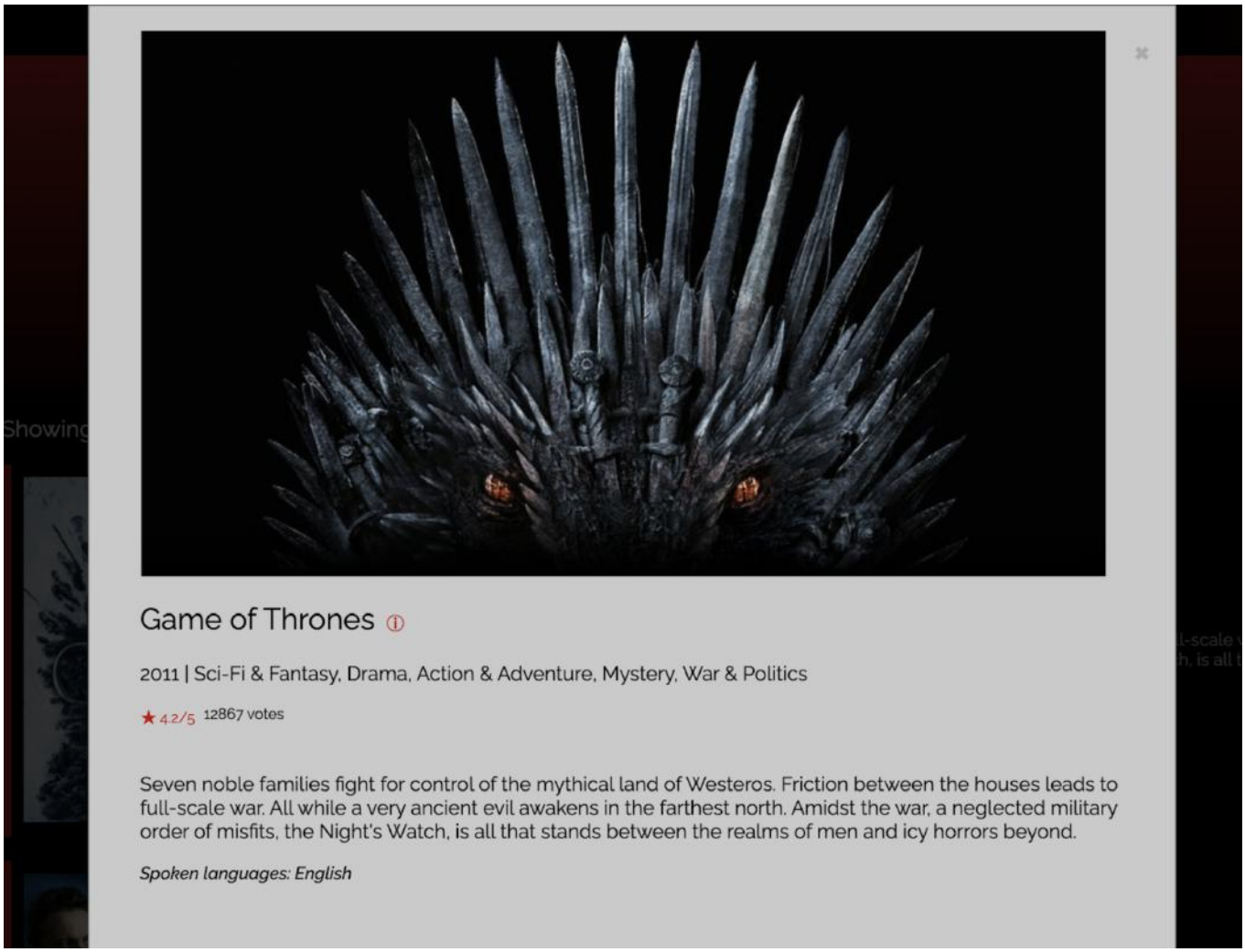
API Example:
https://api.themoviedb.org/3/tv/1399?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US

We will need only these fields:

1. ***backdrop_path*** - path for the larger backdrop image.
2. ***episode_run_time*** - duration of each episode.
3. ***first_air_date*** - first air date of the show
4. ***genres*** - genres of the tv show.
5. ***id*** - id of the tv show.
6. ***name*** - the name of the tv show.
7. ***number_of_seasons*** - number of seasons the tv show was aired for.
8. ***overview*** - the synopsis of the tv show.
9. ***poster_path*** - path for the poster image of the tv show.

10. ***spoken languages*** - different audio languages the show is available in.
11. ***vote_average*** - the average of all ratings given by reviewers
12. ***vote_count*** - total number of reviews received by the tv show.

There needs to be an Information button as indicated next to the movie's title. On clicking this button, the TMDB website of this TV show will open up. The format of this link is: https://www.themoviedb.org/tv/1399 where 1399 is the id of the tv show.



2.4.5 Get TV Show Credits Endpoint

API Sample:
https://api.themoviedb.org/3/tv/**{tv_show_id}**/credits?**api_key=<<*api_key*>>**&language=en-US

API Example:
https://api.themoviedb.org/3/tv/1399/credits?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US

You will only need details of **at most 8** actors. For each actor, only pick these details:

4. ***name*** - Name of the actor
5. ***profile_path*** - Path for the image of the actor
6. ***character*** - The character played by the actor.

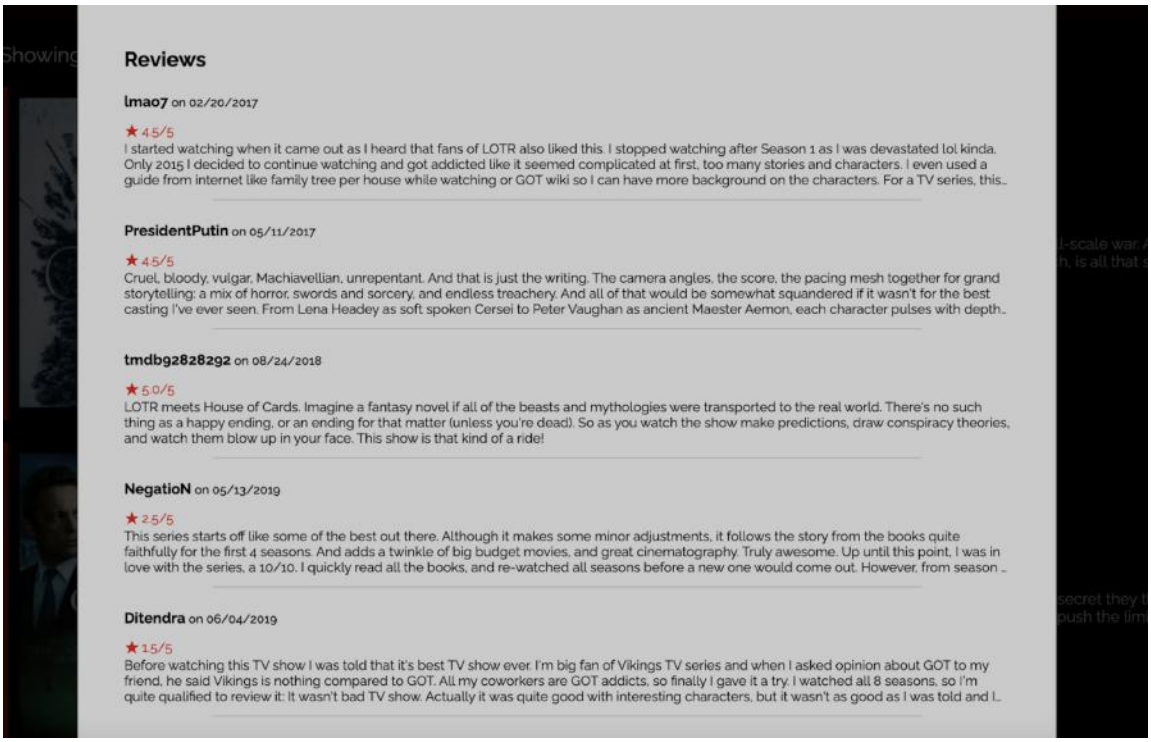
2.4.6 Get TV Show Reviews Endpoint

API Sample:
https://api.themoviedb.org/3/tv/**{tv_show_id}**/reviews?**api_key**=**<<*api_key*>>**&language=en-US&page=1

API Endpoint:
https://api.themoviedb.org/3/tv/1399/reviews?api_key=97588ddc4a26e3091152aa0c9a40de22&language=en-US&page=1

You will need details of **at most 5 reviews** for each show. For each review, only pick these details:

5. ***username*** - The username of the reviewer, inside the author_details object.
6. ***content*** - The content of the review
7. ***rating*** - The rating given by the reviewer for the show, inside the ***author_details*** object.
8. ***created_at*** - The date the review was created.



2.5 Extracting Genres from Genre IDs

2.5.1 TMDB Movie Genres Endpoint

This endpoint gives you a list of all genre IDs and their names, for movies. To convert all Genre IDs to text, you can store the results received from this endpoint and parse each genre ID with its string counterpart.

2.5.2 TMDB TV Genres Endpoint

This endpoint gives you a list of all genre IDs and their names, for TV Shows. To convert all Genre IDs to text, you can store the results received from this endpoint and parse each genre ID with its string counterpart.

2.6 Ratings for Movies and TV Shows

The ratings for movies and TV Shows received from the TMDB API are rated **out of 10**. While displaying it on your web application, scale it down to **5**. For example, if a movie or TV Show is rated as **8.5/10**, you would display **4.25/5**.

3. Other Information

3.1 Creating an API key for TMDB Requests

To create an API key for use in this and the next two assignments, create an account on TMDB and follow the instructions given in this link.

3.2 Placeholder Images

1. If the poster_path field of any object is null, use this image instead.
2. If the backdrop_path field of any object is null, use the image uploaded along with the homework description instead.
3. If the profile_path field of any object for an actor is null, use the image uploaded along with the homework description instead.

3.3 Getting Images from Paths, Image Sizes, and Symbols (Icons)

To get the whole links of images from the paths given, follow the instructions given in this link.

1. For poster_path, use width **w185**.
2. For backdrop_path, use width **w780**.
3. For profile_path, use width **w185**.
4. The **star** and **information** symbols can be created using Unicode in HTML.

3.4 Deploy Python file to the cloud (Azure)

You should use the domain name of the Azure service you created in HW #5 to make the request. For example, if your Azure server domain is called **example.azurewebsites.net**, the following links will be generated:

- Azure - http://example.azurewebsites.net/index.html

The example subdomain in the above URLs will be replaced by your choice of a subdomain from the cloud service. You may also use a different page than index.html.

3.5. Files to Submit

On your course homework page,

1. You should update the Homework 6 link to refer to your new **initial web search page** for this exercise (for example, index.html). Your files must be hosted on the Azure cloud service. Graders will verify that this link is indeed pointing to one of the cloud services.
2. You must also add another link to any **one of your Python Backend endpoints** below the above link.

3. You must submit a **zip file of your code** on DEN.

3.6 Important

- All discussions and explanations in Piazza related to this homework are part of the homework description and grading guidelines. So please review all Piazza threads, before finishing the assignment. If there is a conflict between Piazza and this description and/or the grading guidelines, **Piazza** always rules.
- You **should not** use **JQuery** or **Bootstrap** for Homework 6.
- You **should not** call any of the APIs directly from JavaScript, bypassing the Python proxy. Implementing any one of them in JavaScript instead of Python will result in a 4-point penalty.

All the best! - Prof. Saty, Pranav, Akansha, and Yash.