# Project assignment - Fundamentals of Web Design

Marija Stojcheva
201520

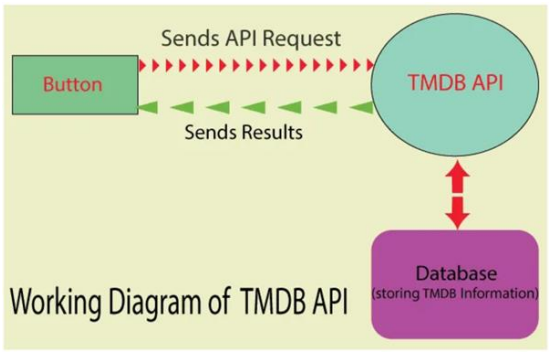
## Designing Responsive Movie Website – Presenting Movie Data with AJAX

Abstract

Nowadays, designing responsive sites with AJAX is widely used. Responsive web design can improve user experience which will translate into a positive perception of the brand and business. On the other hand, AJAX (acronym for Asynchronous JavaScript and XML) is a set of web development techniques using many web technologies on the client-side to create asynchronous web applications. Ajax allows web pages and, by extension, web applications, to change content dynamically without the need to reload the entire page. These two techniques are a perfect combination for designing a movie website, which can be adapted for all devices since it is responsive, and updated asynchronously using AJAX. A well-designed movie website can be extremely helpful, since you can easily find movies that match your interests based on some criteria. On my site users can search the movies/TV shows by title, genre or actors. Title, genres, release date, short plot, actors, user score and teaser about each movie are displayed, as well as short bios for the actors.

## I. Introduction

In my site, I used TheMovieDB API as source for the needed information about the movies. API is the acronym for Application Programming Interface. When an application is used, it connects to the Internet and sends data to a server. The server then retrieves that data, interprets it, performs the necessary actions, and sends it back. The application then interprets that data and presents the needed information in a readable way.

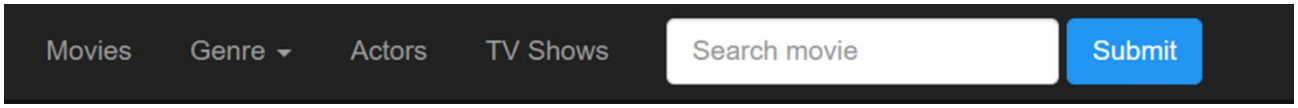



To use TheMovieDB I signed up on TMDB website and got an API key. This API provides details about many movies and actors, free of charge and it is integrated by using jQuery AJAX. jQuery is a fast, small, and feature-rich JavaScript library. It makes HTML document traversal and manipulation, event handling, animation and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. I made my site responsive using Bootstrap libraries.

```
<!DOCTYPE html>
<html lang="en">
<meta charset="UTF-8">
<title>TMDB Application</title>
<meta name="viewport" content="width=device-width,initial-scale=1">
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" />
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<link rel="stylesheet" href="style.css">
<link rel="stylesheet" href="circle.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
<script src="jquery.twbsPagination.min.js"></script>
```

## II. HTML Page Code Description and CSS

The HTML Page contains a menu, input text box and a button. User enters the movie title or actor name in the text box and presses the button. It is made using a <nav> tag as a navbar list. Genre is also an on-click drop down list, followed by an <input type="text" placeholder="Search movie"> and a Submit button.




After that there are empty  tags in which afterwards the information obtained for the movies are shown. There are also close buttons for the  tags in the HTML.
I used CSS to obtain the wanted design of the website.

## III. jQuery AJAX to call TheMovieDB API

In practice, modern implementations commonly utilize JSON instead of XML. The TMDB API provides the result in JSON format as well.
In the HTML page I have the jQuery code given below:

```
$(document).ready(function () {
    $("#submit").click(function (e) {
        var validate = Validate();
        $("#message").html(validate);
        if (validate.length == 0) {
            CallAPI(1);
        }
    });
```

A page can't be manipulated safely until the document is "ready." jQuery detects this state of readiness.
Validate () checks whether the selected form is valid or whether all selected elements are valid.

On the button click the CallAPI() method is called. This method makes the jQuery AJAX call to the TMDB API. It takes the page number as parameter. In this button click event I passed 1 for page number because I wanted the API to return the results of the first page.

```
function CallAPI(page) {
    $.ajax({
        url: "https://api.themoviedb.org/3/search/movie?language=en-US&query=" + $("#searchInput").val() + "&page=" + page + "&include_adult=fal
        data: { "api_key": "65771d19b4a665e86c2dd45d55b5d10e" },
        dataType: "json",
        success: function (result, status, xhr) {
            var resultHtml = $("<div class=\"resultDiv\">");
            for (i = 0; i < result["results"].length; i++) {

                if (!!result["results"][i]["poster_path"]) {
                 var image = "https://image.tmdb.org/t/p/w300/" + result["results"][i]["poster_path"];
                 resultHtml.append("<div class=\"result\" resourceId=\"" + result["results"][i]["id"] + "\">" + "<img src=\"" + image + "\" />
                }


            }

            resultHtml.append("</div>");
            $("#message").html(resultHtml);

            //Paging(result["total_pages"]);
        },
        error: function (xhr, status, error) {
```

## IV. Movie jQuery and Results

In the function CallAPI() for the url I used the following link from The Movie DB API and adapted it accordingly.

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

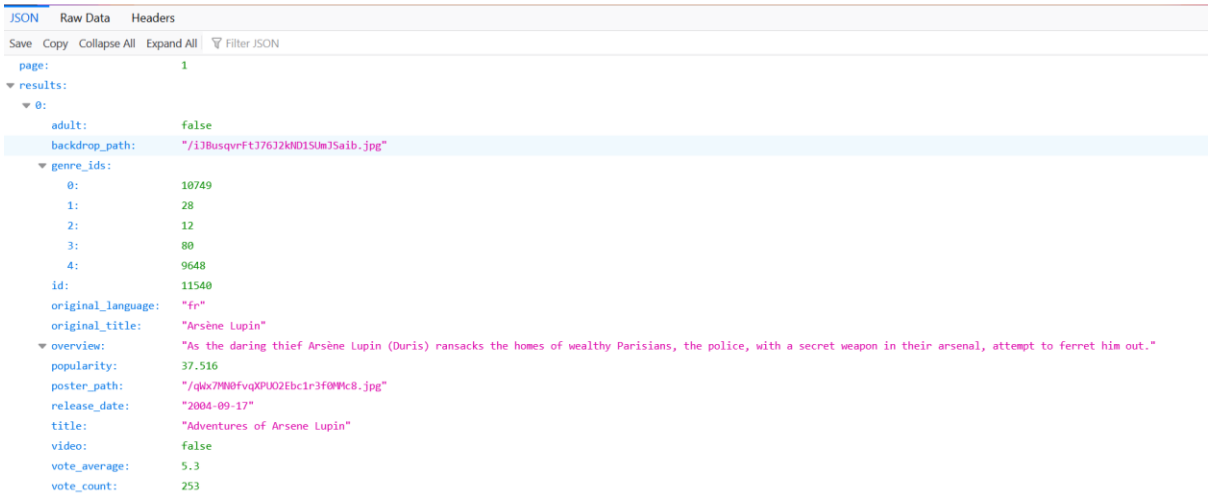
Parameters are found at the very end of the URL or within the path , depending on the implementation. URL Parameters are represented in key/value pairs, beginning with a '?' and separated by an ampersand '&'. In the URL I dynamically inserted the parameters which are different for every single search.
The api_key I got must be included in the data part and datatype should be set as json. Success(result,status,xhr) is a function to be run when the request succeeds. On the jQuery AJAX Success callback function, there is looping through the JSON result and extracting values from it. These values are put in a div structure in which custom attribute called resourceId is created and the id of every result is added to it. Later on these ids are used to make another jQuery AJAX request to fetch the complete information for that id.
So, if the request was successful after entering the movie/Tv show/actor name, with the append() method that inserts specified content at the end of the selected elements, the posters of movies and actors that satisfy the criteria are shown on the screen.

```
resultHtml.append("</div>");
$("#message").html(resultHtml);
```

When the API has started giving back the results, a functionality should be added so that on clicking any movie's picture, the API will give us details about it.These details are shown inside a Bootstrap Modal. For this purpose a Bootstrap Modal is created and another API call is made (this time for getting the details of a movie).

```
$("#message").on("click", ".result", function () {
    var resourceId = $(this).attr("resourceId");
    $.ajax({
        url: "https://api.themoviedb.org/3/movie/" + resourceId + "?language=en-US&append_to_response=videos",
        data: {
            api_key: "65771d19b4a665e86c2dd45d55b5d10e"
        },
        dataType: 'json',
        success: function (result, status, xhr) {
            $("#modalTitleH4").html(result["original_title"]);
```

With the resourceId previously obtained and the api_key, I form the url and search the data that I need in the data base.



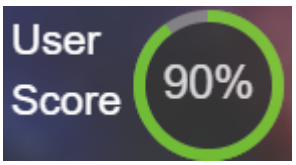

Here are all the information available for a given movie. I used the original title, the poster path, from the release date I took only the year as the first four characters of a string, I listed all the genres the movie belongs to. I was using the ternary operator, because if some of the information wasn't available in the data base, appropriate message to be shown.

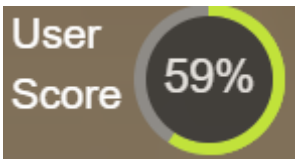
```
var image = result["poster_path"] == null ? "Image/no-image.png" : "https://image.tmdb.org/t/p/w300/" + result["poster_path"];
```

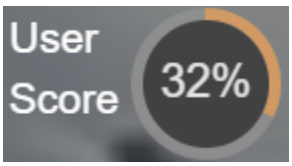
I wanted to show the runtime in format (h) : (min) so I divided the minutes with 60 and rounded the number with the help of the ~~ operator. I got the remaining minutes with the modulo operator.

```
var hours = ~~(runtime / 60);
var minutes = runtime % 60;
```

I paid special attention to the vote average part. In my opinion it is a really nice way to visualize the popularity of the movie and to see the general opinion about it. In the API the vote average was given as a grade (1-10). I decided to multiply it by 10 and show it as a percentage circle. I did this with a help of a .css file I found online. I also decided to add three different colors according to the grading. Green if the film has over 70%, orange if it is less than 50% and some color halfway between them for the remaining percentages.




All previously listed informations are added to the header of the previously mentioned empty  tag. After that the only thing remaining is to write the part where the obtained data is put in body of the  and style it. It consists of an overview and a short trailer for each of the movies. With url from the data base and <iframe>, video from YouTube for each movie appears.

```
var resultHtml = "<div id=\"info\"><div style=\"display:flex\"><div style=\"width:60%\">" + release_date + " " +"&bull;" + " " + ge
if (hours >0) {
    resultHtml += hours + "h" + " " + minutes + "min</div>";
    } else {
    resultHtml += minutes + "min</div>";
    }
resultHtml += "<div style=\"width:40%;float:right\" class=\"clearfix\"><div style=\"float:right\" class=\"c100 p"+ vote_average +" small " + b
resultHtml += "<p><b>Overview</b></p>";
resultHtml += "<p>" + overview + "</p></div>";
var trailer_key = result["videos"]["results"][0]["key"];

resultHtml += "<div class=\"embed-responsive embed-responsive-16by9\"><iframe class=\"embed-responsive-item\" src=\"https://www.youtube.com/e
```

## IV. Actors jQuery and Results

The information about the actors are obtained similarly as for the movies. The following link is used.

https://api.themoviedb.org/3/search/person?api_key=<<api_key>>&language=en-US&page=1&include_adult=false

https://api.themoviedb.org/3/person/{person_id}?api_key=<<api_key>>&language=en-US

From information I get for each of the actors, and I extract the name, the photo and a short biography.

## V. Genre jQuery and Results

In the HTML file for the genre I create the drop down list with all the genres I wanted to include. For the genres I used the following url and adapted it accordingly

https://api.themoviedb.org/3/genre/movie/list?api_key=<<api_key>>&language=en-US


```
$(document).ready(function () {
    $("#action").click(function (e) {

        var validate = Validate(28);
        $("#message").html(validate);
        if (validate.length == 0) {
            CallGenreAPI(1,28);
        }
    });
    $("#adventure").click(function (e) {

        var validate = Validate(12);
        $("#message").html(validate);
        if (validate.length == 0) {
            CallGenreAPI(1,12);
        }
    });
    $("#animation").click(function (e) {

        var validate = Validate(16);
        $("#message").html(validate);
        if (validate.length == 0) {
            CallGenreAPI(1,16);
        }
```

Each genre has its own id, and I send that id to the CallGenreAPI() function and this is the only function that differentiates genres.html from the movies.html file. With this function I list movies from a particular genre with their posters and horizontal scrollbar is available. After a movie is selected, it works in the same principle as previously explained.

```
function CallGenreAPI(page, id) {
        $.ajax({
            url: "http://api.themoviedb.org/3/discover/movie?&with_genres=" +id + "&page=" + page + "&include_adult=false",
            data: { "api_key": "65771d19b4a665e86c2dd45d55b5d10e" },
            dataType: "json",
            success: function (result, status, xhr) {
                var resultHtml = $("<div class=\"resultDiv\">");
                for (i = 0; i < result["results"].length; i++) {

                    if (!!result["results"][i]["poster_path"]) {
                     var image = "https://image.tmdb.org/t/p/w300/" + result["results"][i]["poster_path"];
                     resultHtml.append("<div class=\"result\" resourceId=\"" + result["results"][i]["id"] + "\">" + "<img src=\"" + image + "\"
                    }


                }

                resultHtml.append("</div>");
                $("#message").html(resultHtml);
```

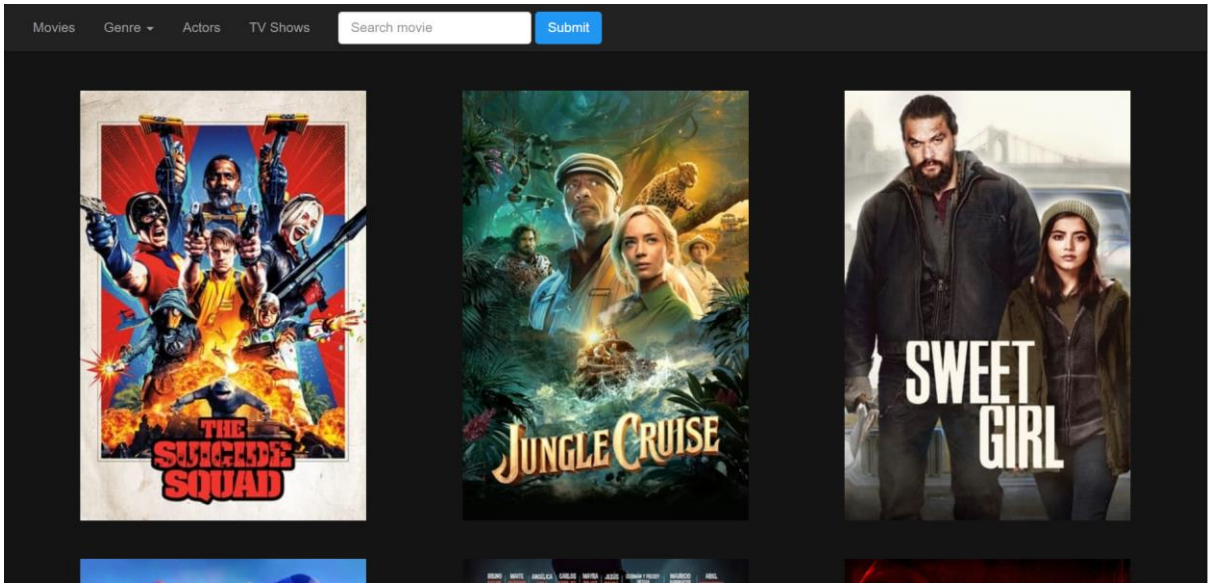
## VI. TV Shows jQuery and Results

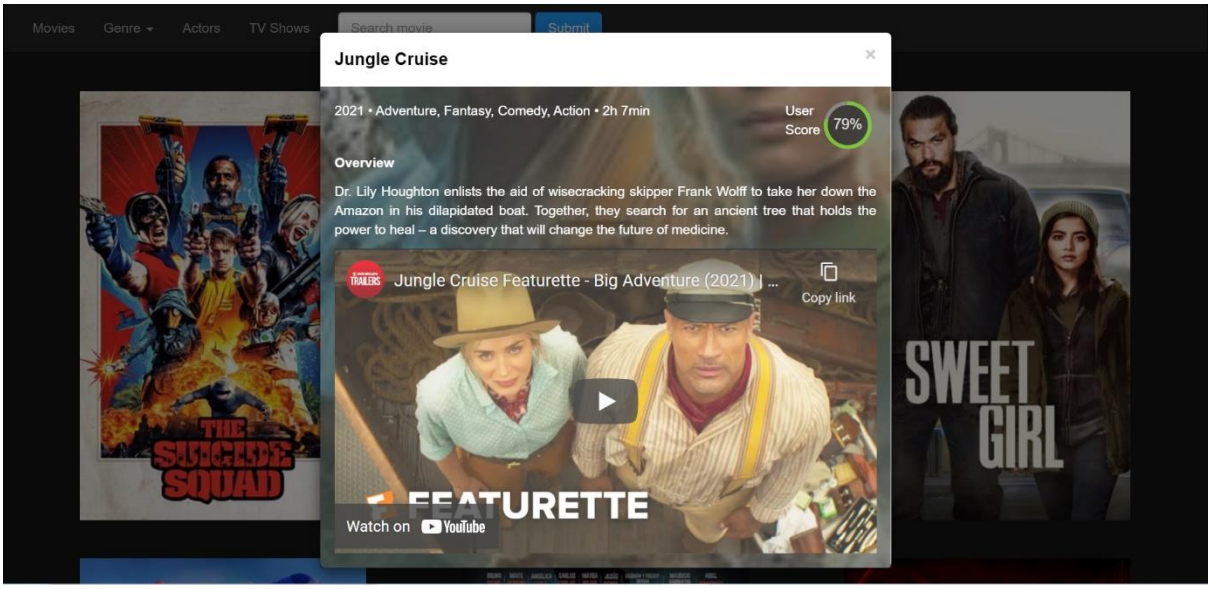
Every movie website needs a TV shows section to be complete. This works on the same principle as the movies part, only the URL part is changed, the var release_date is changed with first_air_date, the var title with name and the runtime is removed since every episode has different duration.

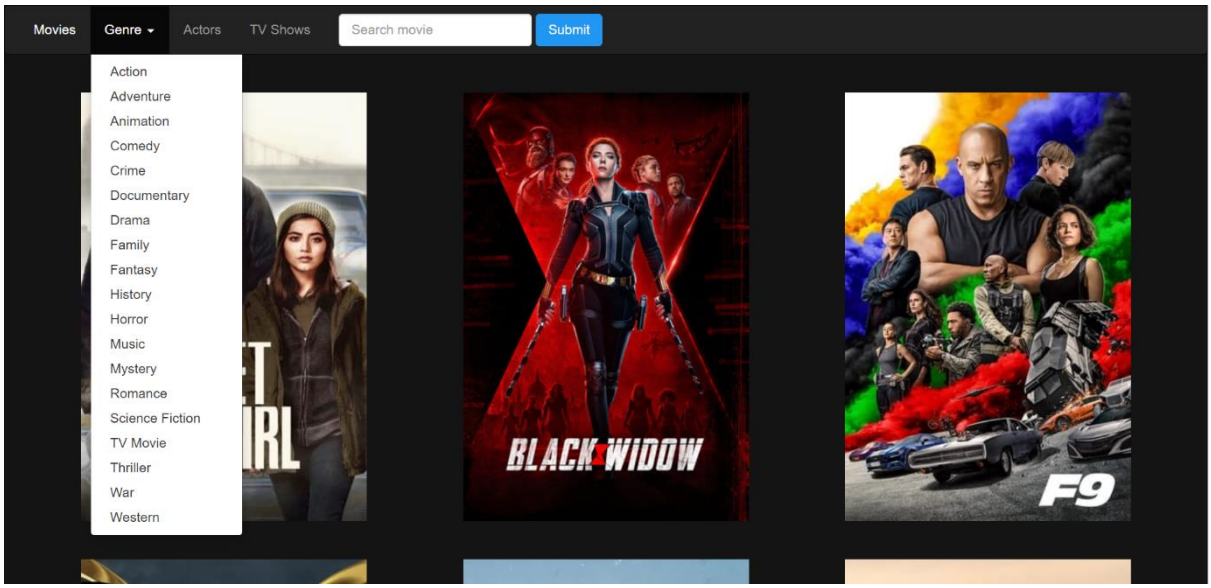
object

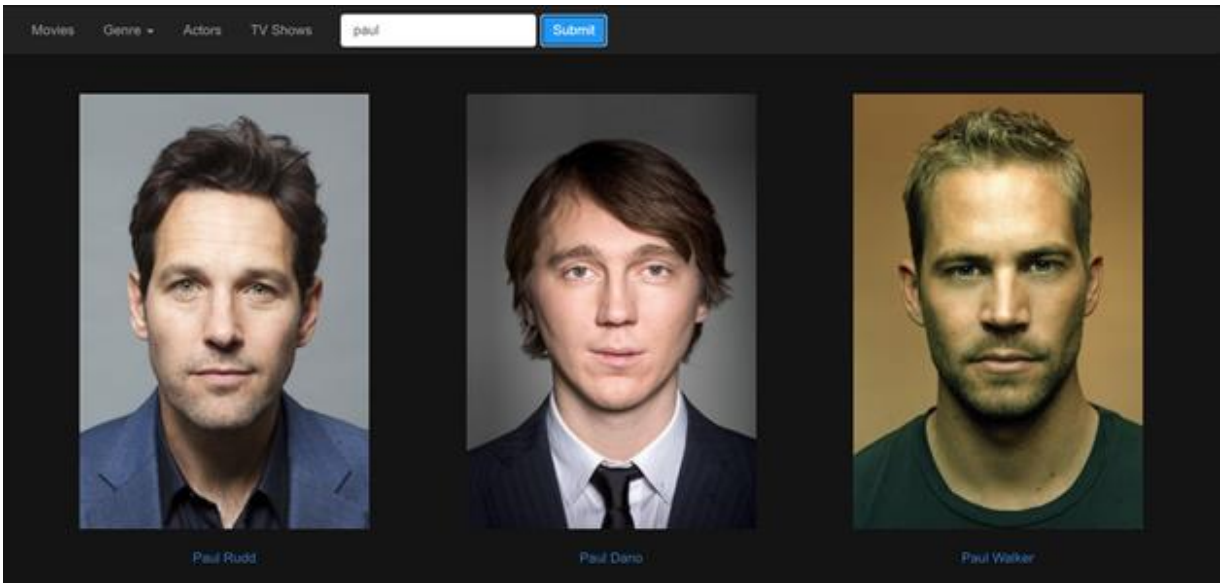
| | | | |
|---|---|---|---|
| page | integer | | optional |
| ▾ results | array[object] | {TV List Result Object} | optional |
| poster_path | string or null | | optional |
| popularity | number | | optional |
| id | integer | | optional |
| backdrop_path | string or null | | optional |
| vote_average | number | | optional |
| overview | string | | optional |
| first_air_date | string | | optional |
| origin_country | array[string] | | optional |
| genre_ids | array[integer] | | optional |
| original_language | string | | optional |
| vote_count | integer | | optional |
| name | string | | optional |
| original_name | string | | optional |
| total_results | integer | | optional |

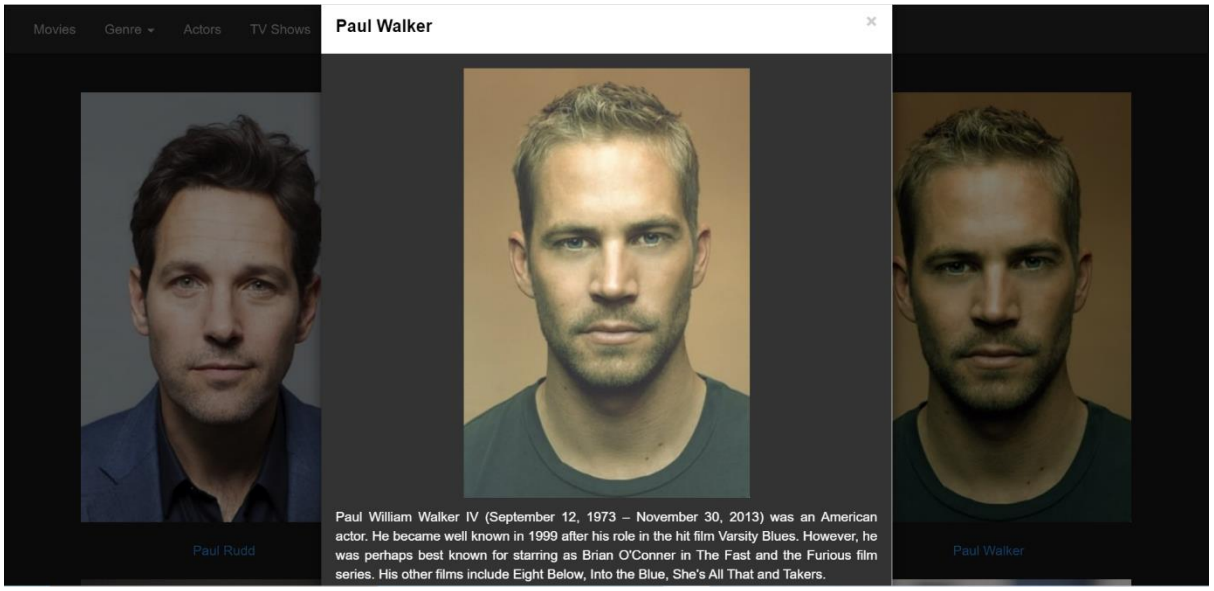
## VII. The Result

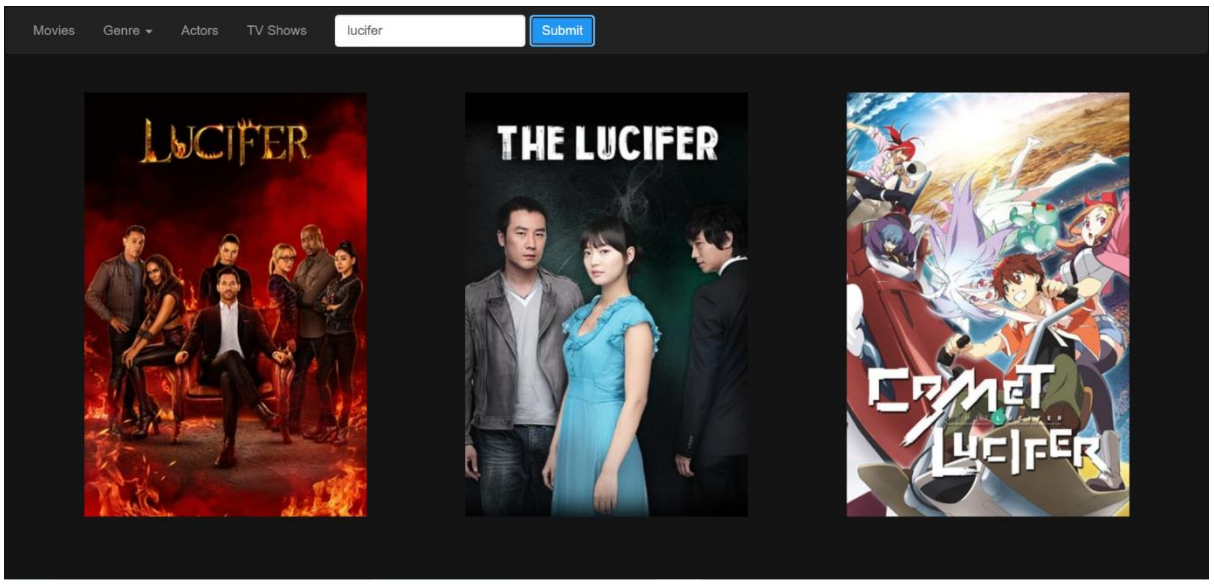


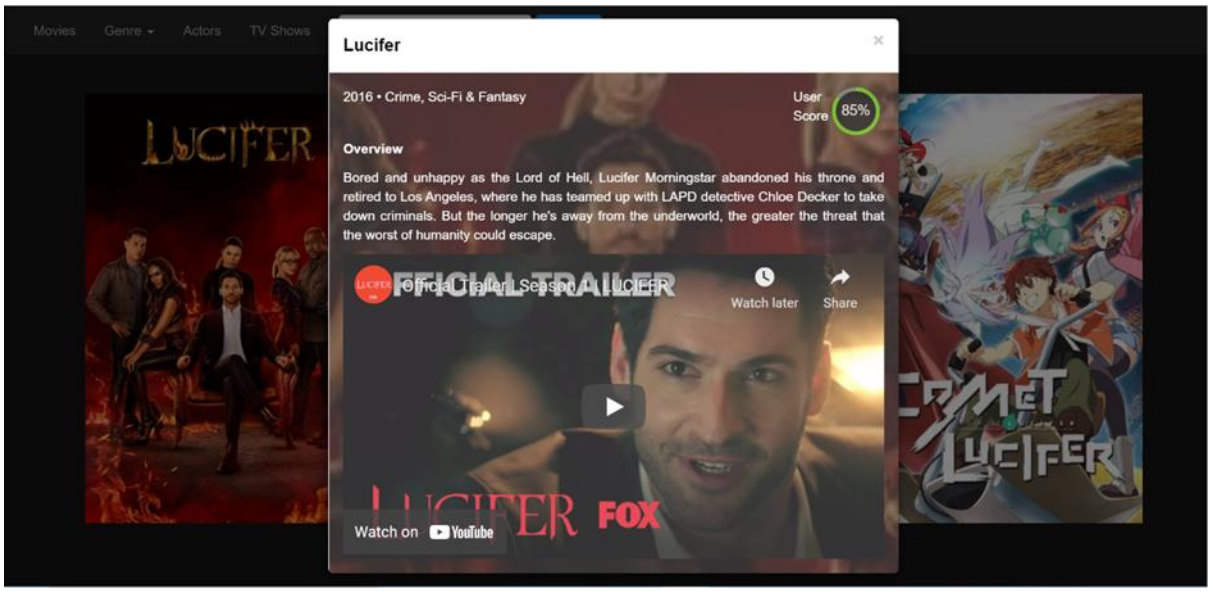

Movies
Genre
Actors
TV Shows
Search movie
Submit
Jungle Cruise
2021 • Adventure, Fantasy, Comedy, Action • 2h 7min
User Score
79%
Overview
Dr. Lily Houghton enlists the aid of wisecracking skipper Frank Wolff to take her down the Amazon in his dilapidated boat. Together, they search for an ancient tree that holds the power to heal – a discovery that will change the future of medicine.
Jungle Cruise Featurette - Big Adventure (2021) | ...
Copy link
FEATURETTE
Watch on YouTube
THE SUICIDE SQUAD
SWEET GIRL

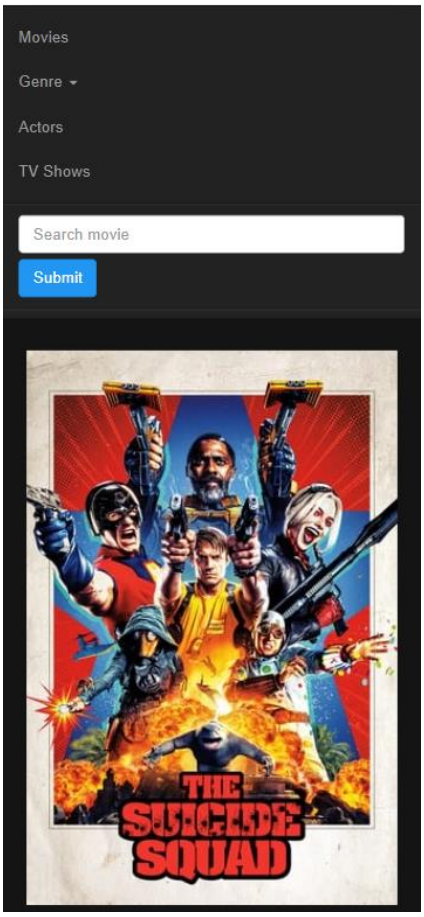

Movies
Genre
Actors
TV Shows
Search movie
Submit
Action
Adventure
Animation
Comedy
Crime
Documentary
Drama
Family
Fantasy
History
Horror
Music
Mystery
Romance
Science Fiction
TV Movie
Thriller
War
Western
BLACK WIDOW
F9

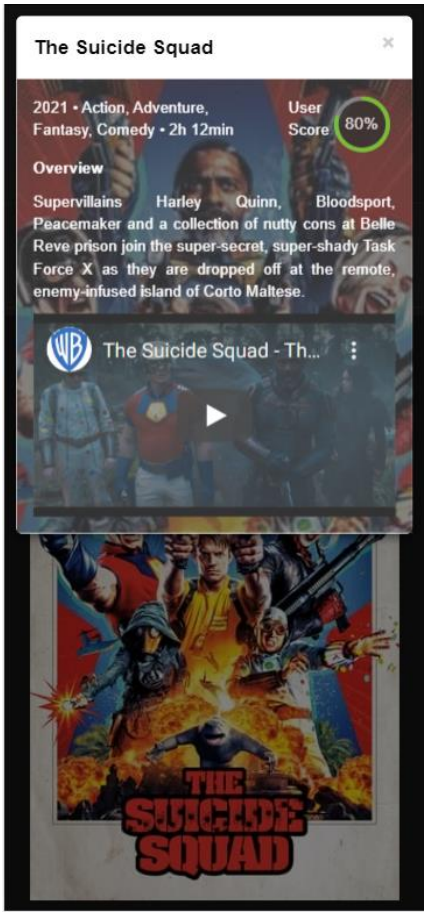

Movies
Genre
Actors
TV Shows
paul
Submit
Paul Rudd
Paul Dano
Paul Walker

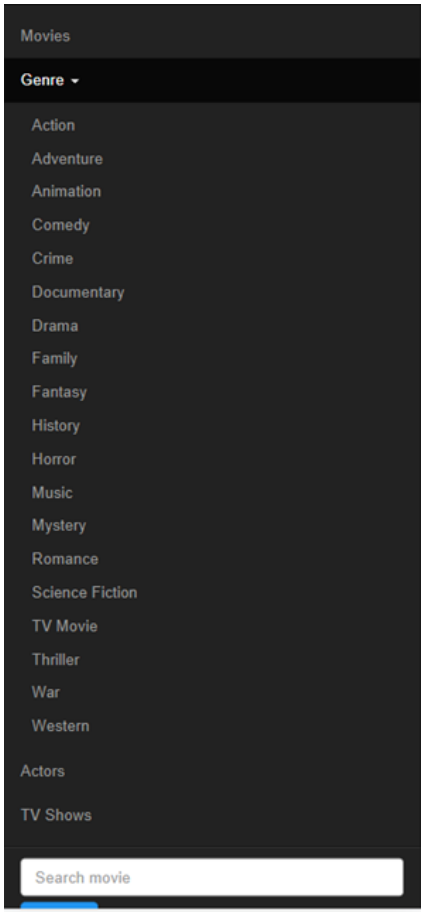

Movies
Genre
Actors
TV Shows
Paul Walker
Paul Rudd
Paul William Walker IV (September 12, 1973 – November 30, 2013) was an American actor. He became well known in 1999 after his role in the hit film Varsity Blues. However, he was perhaps best known for starring as Brian O'Conner in The Fast and the Furious film series. His other films include Eight Below, Into the Blue, She's All That and Takers.
Paul Walker

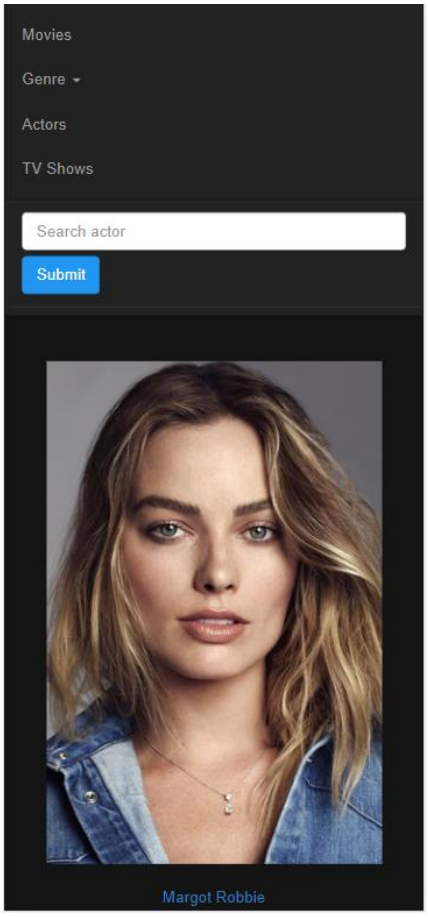

Movies
Genre
Actors
TV Shows
lucifer
Submit
LUCIFER
THE LUCIFER
COMET LUCIFER


Movies
Genre
Actors
TV Shows
Lucifer
2016 • Crime, Sci-Fi & Fantasy
User Score
85%
Overview
Bored and unhappy as the Lord of Hell, Lucifer Morningstar abandoned his throne and retired to Los Angeles, where he has teamed up with LAPD detective Chloe Decker to take down criminals. But the longer he's away from the underworld, the greater the threat that the worst of humanity could escape.
Official Trailer | Season 4 | LUCIFER
Watch later
Share
Watch on YouTube
FOX
LUCIFER
COMET LUCIFER

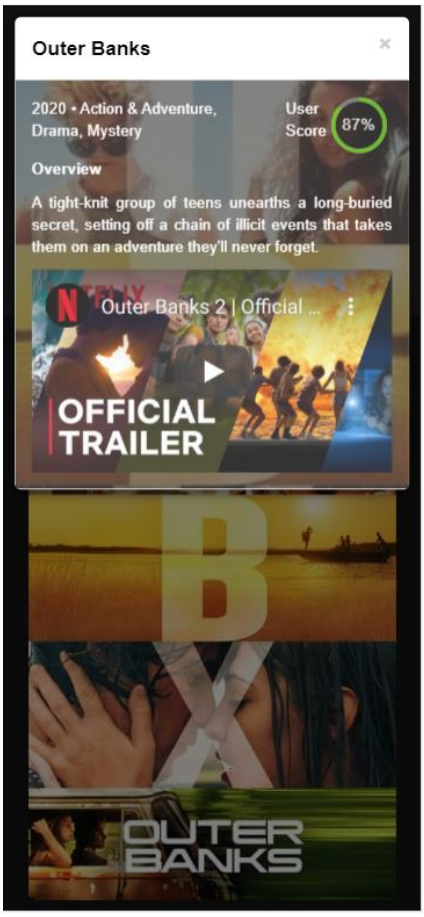

Movies
Genre
Actors
TV Shows
Search movie
Submit
THE SUICIDE SQUAD


The Suicide Squad
2021 • Action, Adventure, Fantasy, Comedy • 2h 12min
User Score 80%
Overview
Supervillains Harley Quinn, Bloodsport, Peacemaker and a collection of nutty cons at Belle Reve prison join the super-secret, super-shady Task Force X as they are dropped off at the remote, enemy-infused island of Corto Maltese.
The Suicide Squad - Th...
THE SUICIDE SQUAD


Movies
Genre
Action
Adventure
Animation
Comedy
Crime
Documentary
Drama
Family
Fantasy
History
Horror
Music
Mystery
Romance
Science Fiction
TV Movie
Thriller
War
Western
Actors
TV Shows
Search movie


Movies
Genre
Actors
TV Shows
Search actor
Submit
Margot Robbie


Margot Robbie
Margot Elise Robbie (born 2 July 1990) is an Australian actress and producer. She has received nominations for two Academy Awards and five BAFTA Awards. In 2017, Time magazine named her one of the 100 most influential people in the world, and in 2019, she was ranked among the world's highest-paid actresses. Robbie studied drama at Somerset College and began her career in Australian independent films in the late 2000s, before working in the soap opera Neighbours (2008–2011). After moving to America, she starred in the ABC drama series Pan Am (2011–2012) and had her breakthrough role in Martin Scorsese's


Outer Banks
2020 • Action & Adventure, Drama, Mystery
User Score 87%
Overview
A tight-knit group of teens unearths a long-buried secret, setting off a chain of illicit events that takes them on an adventure they'll never forget.
Outer Banks 2 | Official ...
OFFICIAL TRAILER
OUTER BANKS

## VIII. References

[1] TMDB [Online]. Available: https://www.themoviedb.org/documentation/api [Accessed: 05- June- 2021].
[2] Wikipedia, ‘Ajax(programming)’ [Online].
Available: https://en.wikipedia.org/wiki/Ajax_(programming) [Accessed: 05- June- 2021].
[3] MuleSoft, ’What is an API?’ [Online].
Available: https://www.mulesoft.com/resources/api/what-is-an-api [Accessed:05- June-2021].
[4]W3 Schools, ‘jQuery’ [Online].
Available: https://www.w3schools.com/jquery/ [Accessed: 07- June- 2021].