The Project 2 Web programming report

Jirayut Saengsiwarit Sec 3 5988200

Namchew Sanguandikul Sec 3 5988204

Vasu Senawongse Sec 3 5988211

Supakorn Sillapathadapong Sec 3 5988214

**1. The web accessibility**

For the web accessibility, we have created a website that can use a responsive function by using bootstrap grid tools. Then it can use easily on the full screen (windows mode) like figure 1 or small screen such as smartphone like figure 2.

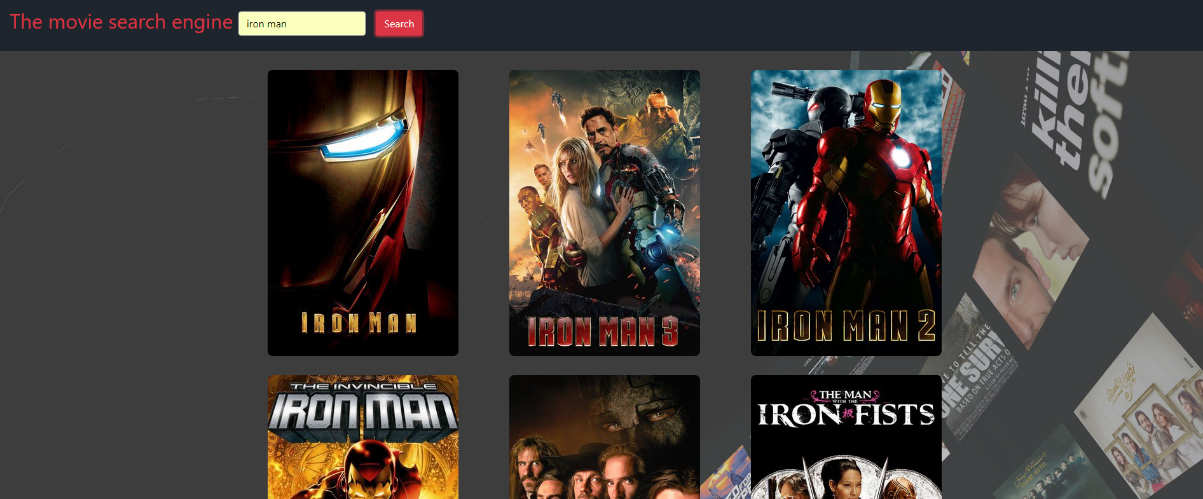


Figure 1: Full screen mode



Figure 2: small screen mode

**2. How is the webpage friendly?**

To make the webpage friendly for user, we use a big to poster to represent each movie. The User will have more convenience due to they can click to a huge size poster to movie information instead of clicking word.

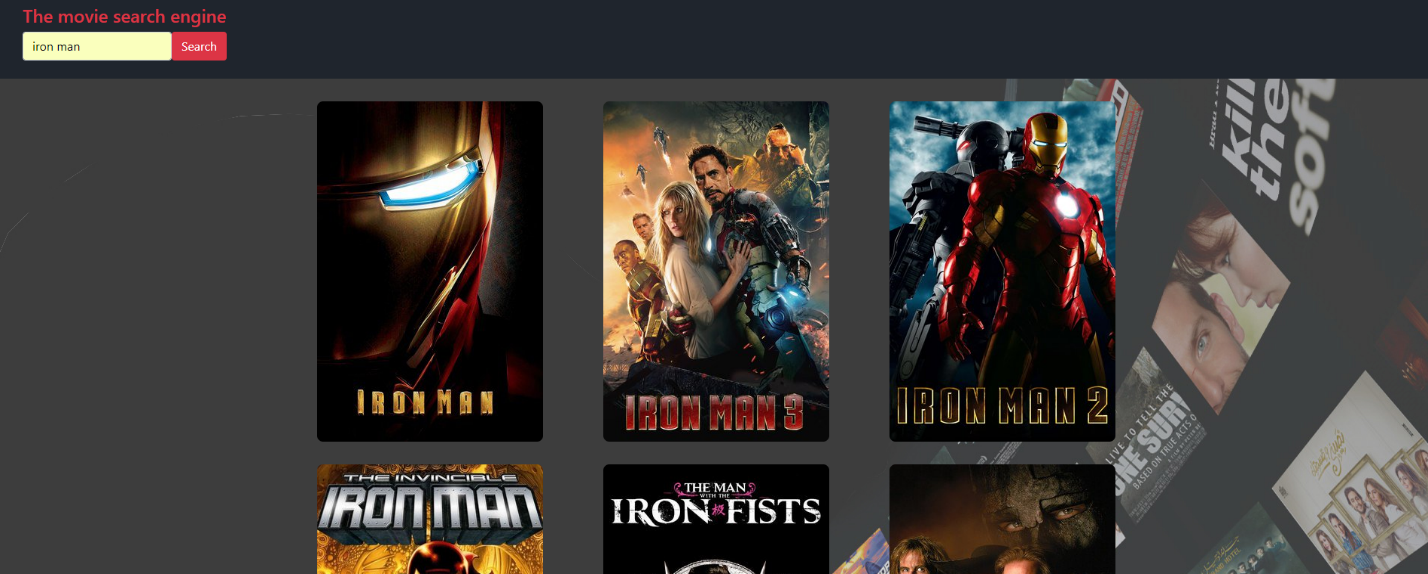


Figure 3

**3. How can you interact with all web services?**

**Search movie information API**

For the searching movie data, we use a themoviedb API that provide by the themoviedb

website. We interact with API by using jquery.get to send the query data and the API key

in the form of url to the web service then receive the data back

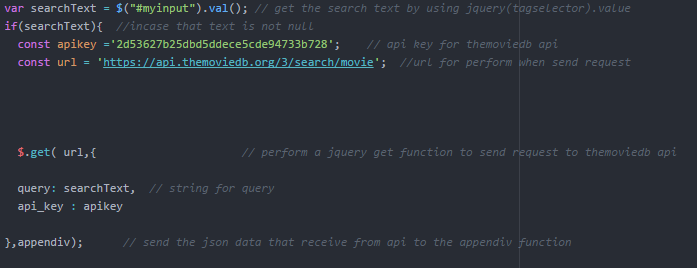


Figure 4

**YouTube API for searching movie trailers**

For the YouTube API, we use using jquery.get to send the url query and API key to the web service and receive the data back in the case we also use a sync: false due to we need to complete receive all data before perform next code to avoid null data.

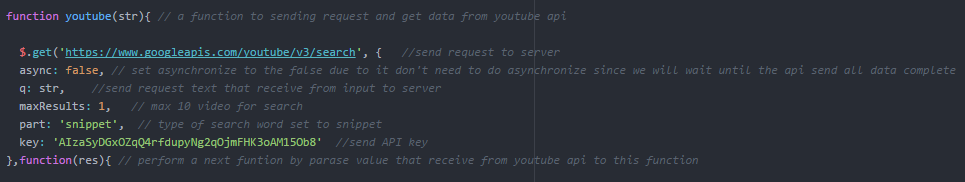


Figure 5

**Twitter API for searching comment**

For the twitter API, we use a twitteroauth library from the Twitter developer website to send a key and token to the web service and receive back data by using get method like figure 6,7 and 8.



Figure 6: create an object TwitterOAuth= from class file in library and use the method get to send the query

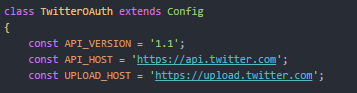


Figure 7: constant variable the keep API url

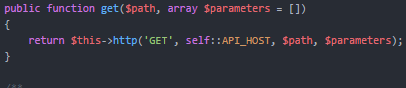


Figure 8: get method in the TwitterOAuth object

**Spotify API for searching movie soundtracks**

For Spotify API, we use the PHP library that downloaded from developer.spotify.com and include it in our file. As Spotify API required client ID, client secret ID, redirectURL and Access Token so we create function doIt to get an authorization before retrieving the token and set it via the setAccessToken function defined in library function. The search function is defined in SpotifyWebAPI.php which is a library function file.



Figure 9: Include 4 library files inside

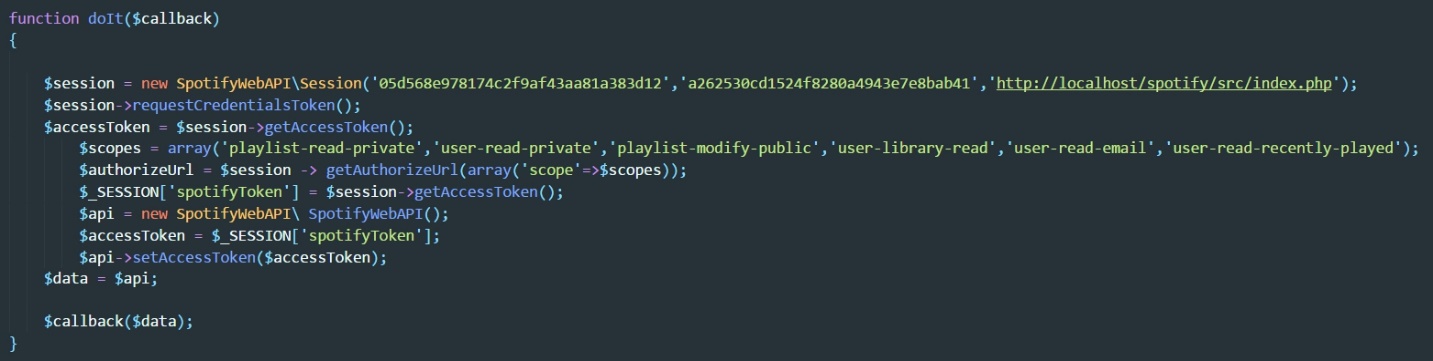


Figure 9: doIt function code

While the keyword is being sent by AJAX from the mainpage, mycallback function is created for retrieving the keyword for searching using GET method. Search function is used to create an object and send it to the server before sending the result back and demonstrate the result on the screen by using an iframe.



Figure 10: search function inside library (SpotifyWebAPI.php)



Figure 11: myCallback function – retrieving keyword, calling search function and demonstrating the result

**4. The architecture of system.**

YouTube API

.getjson

.getjson

index.HTML

JavaScript

JQuery

The moviedb API

Use get method perform ‘GET’

load

Create object, pass value to the method

twitter.php

echo content

Twitter web service

TwitterOAuth library

TwitterOAuth subject

Ajax

Create object, pass value to the method

Use get method perform ‘GET’

load

Spotify.php

mycss.css

echo content

External css

Spotify web API

Spotify web service

Spotify web API object

**5. The structure of your code (e.g. how does each code interact the others?)**

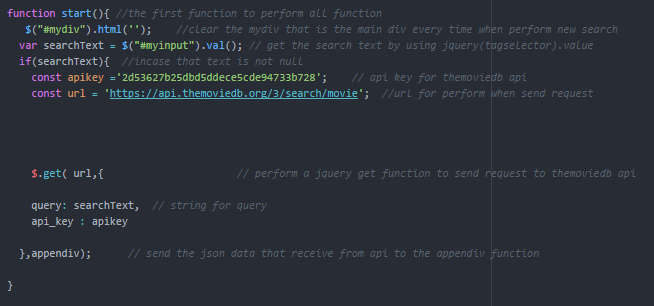
First when the user type a search string and click search button to summit start() function will perform and get the data from the themoviedb API and sent to the appendiv() function like figure 12

Figure 12: start() function

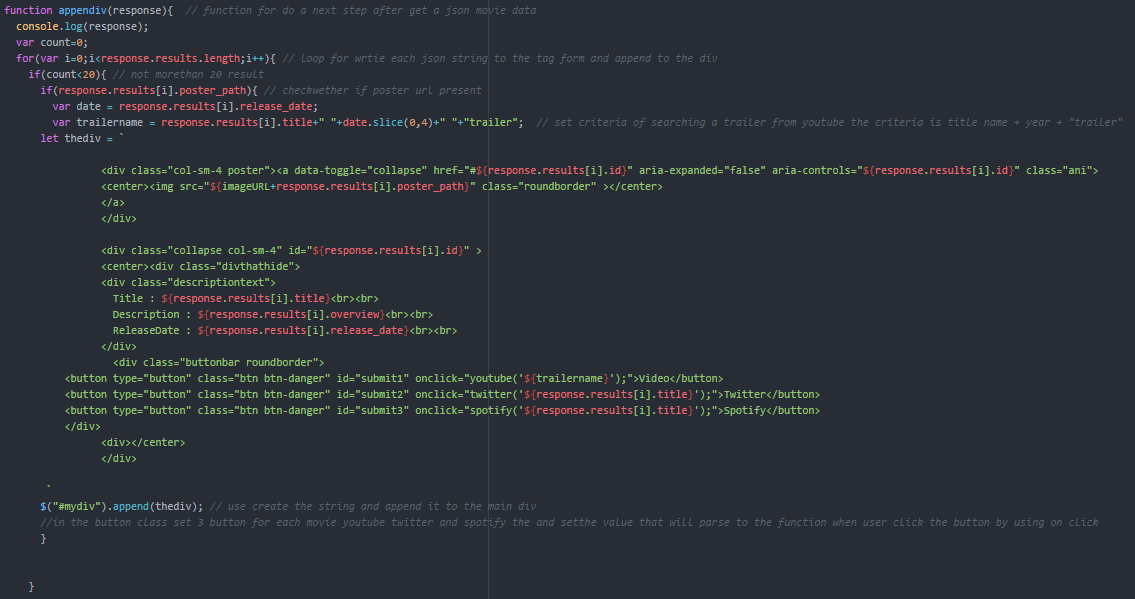
The appenddiv() function is use to looping each movie to create a div and append to div in the main page each div will have a 1 hidden div for show descriptions of movie it will show when click <a> tag in the poster image. Moreover, inside the hidden has a tree button to perform a youtube() function twitter() function and spotify() function that we already set the value in the onclick(). When button clicked it will pass these value to the function. 

Figure 13: appendiv() method

The function youtube() will perform will click the button. It will receive the trailer name string that already set in the function before and search a video for that movie title after that it will append the iframe movie tag to the hidden div() that already set and show the modal popup of video trailer like figure 14.

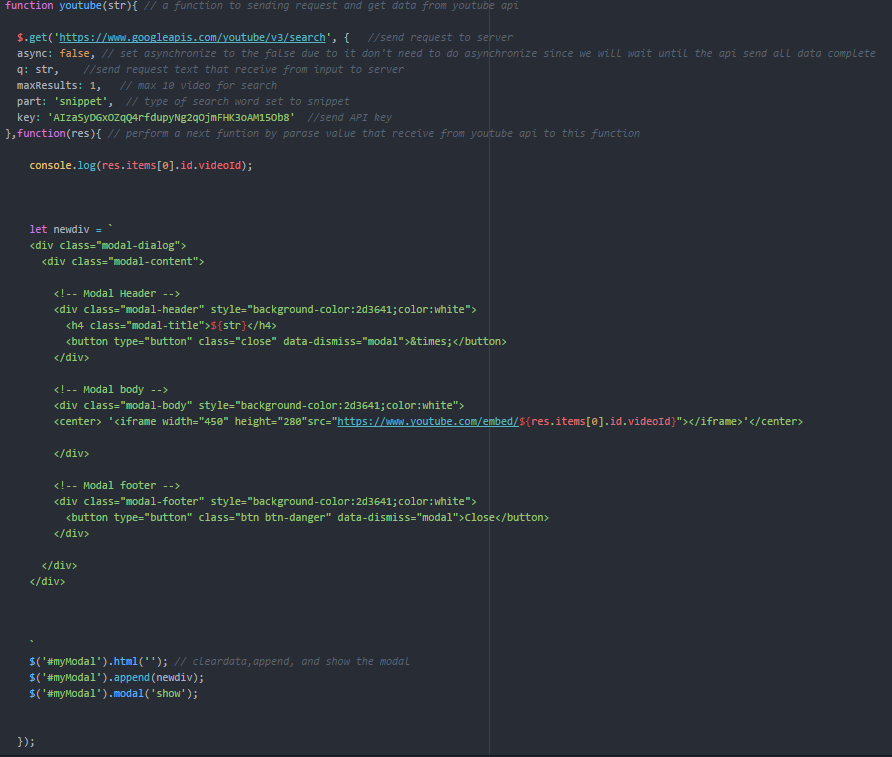
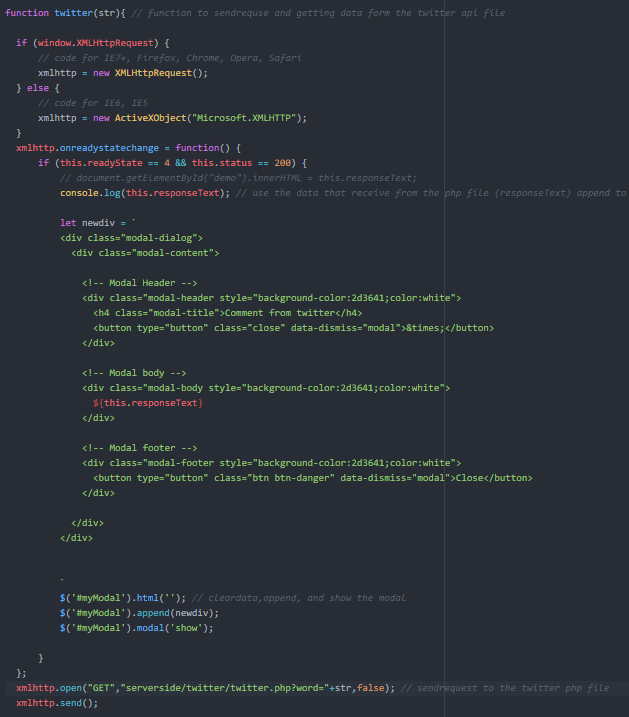


Figure 14: youtube() function

The function twitter() will perform will click the button. It will receive the title string and searching for twitter comment after that it will perform by sending and receive the data with twitter.php trough ajax and append it to the hidden div() that already set and show the modal of popup comment like figure 15 and 16.

Figure 15: twitter() function

Get the word form twitter() then set the key and token. Use the key and token to create the object of TwitterOAuth, then use get method from TwitterOAuth to identify accounts and send query to web service and receive value in statuses variable.



Figure 16: twitter.php

The function spotify() will perform will click the button. It will receive the title string and searching for a movie soundtrack after that it will perform by sending and receive the data with spotify.php trough the jquery .ajax and append it to the hidden div() that already set and show the modal of Spotify iframe like figure 17 and 18

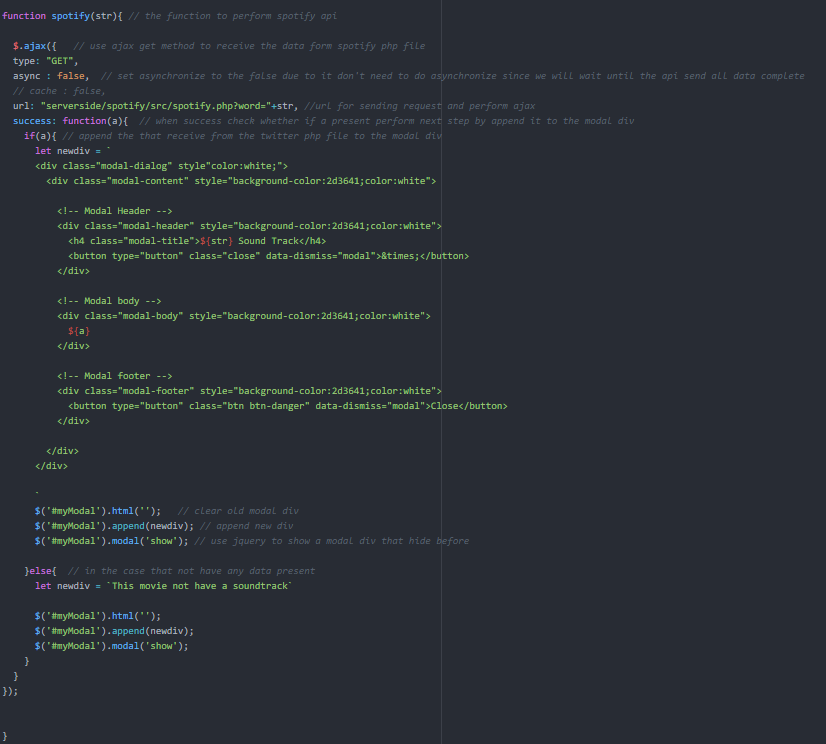


Figure 17: spotify() function



Figure 18: spotify.php send and receive the data back

**Extra  
How our extra work?**

First string lower, remove hashtag symbol (remove #) and explode the tweet. Then make a list of positive and negative words, and declare a variable “score” for count the point of comment.

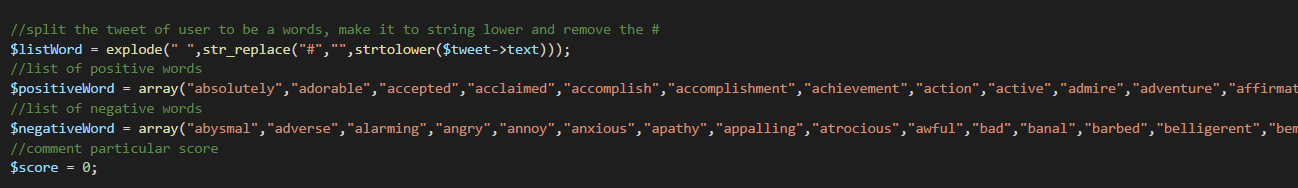


Figure 19: twitter.php declare variable

Next loop for count score of each comment. This loop will check form every single word, if positive word the score will increase but if negative word score will decrease.

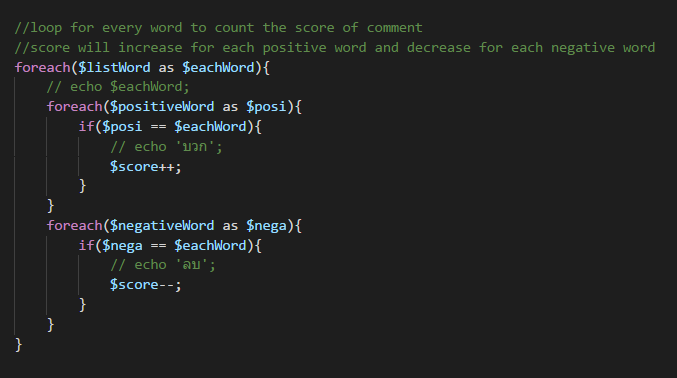


Figure 20: twitter.php count score from each word of tweet

Finally, checking the score if it equal to 0 the comment is Neutral, if more than 0 the comment is Positive, if less than 0 the comment is Negative.

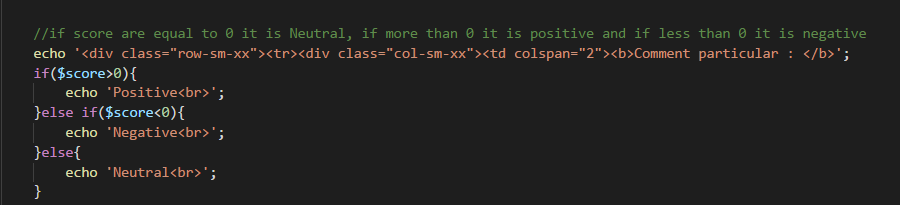


Figure 20: twitter.php checking score for result