# CMT112

# Web Application Development


**Student Number**: 1766249

**Date Set:** 6th November 2017

**E-Submission Date:** 9:30 am 8th January 2018

# Contents

# Introduction

For the Web Application module (CMT112) final assignment we were asked to build a front-end website using HTML, CSS and JavaScript language that allows users to interact with and view data from a number of online APIs.

In addition, the website should use more than one API as a data source. It should also be designed to be easy to use and navigate for the user.

There were no restrictions on what APIs we should use for our website. Therefore, I have created a website that present information from the OpenDota and PandaScore APIs to the user, as well as allowing the user to provide input to the website to change the data displayed by using the YouTube API.

# Rationale for API Choice and Design

Based on the list of APIs that we were given and from the free APIs I have searched online I have found UFC Data, Riot Games, OpenDota and PandaScore APIs the most exciting to work with to create front-end website for this assignment.

First, I started working with UFC Data API, but unfortunately, I was getting a CORS error which I believe was based on the fact that their API was only based on iPhone and Android devices. Moreover, for the Riot Games API I needed to get authorization by emailing them personally, in order for them to give me access to a particular set of their data (for example their league rankings).

Therefore, I ended up with the OPenDota and PandaScore APIs which both had a big amount of data to explore and get. Another reason of why I chose those two APIs was that I was a little familiar with some of their data terminologies I had played the game in the past.

In addition, I added YouTube Data API as I thought it would fit well with the other gaming APIs. For example, the user could be interested to see one of the pro league matches available at the match history table in the Esport tab

and then go straight to the YouTube search bar at the side navigation menu of the page and search for that specific game to watch.

Furthermore, the inspiration for my website's design was based on the official Dota2 website (http://www.dota2.com). Although the main idea for my design was to present the game and its competitiveness as much as possible, thus I have added the main game trailer in the home page (index.html).

## Website Functionality

The website consists of six pages (html files) and are all based on the same theme. First, I designed my side navigation bar to overlay (Figure 1.0) when the user clicks onto the three bars (lines Figure 1.1) and close by clicking onto the cross icon. The side navigation bar I believe is a nice way to "store" the social media links of Dota2 and have a page for the assignments references.

***Figure 1.1***

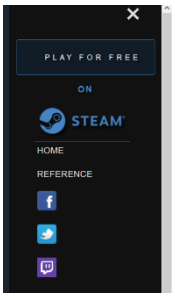



***Figure 1.0***

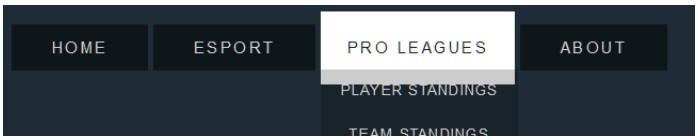
To create an overlay side bar, I just created two functions in my JavaScript files that changes the width of the specified element id (Figure 1.2) and calling the functions through the html files.

```
/* Set the width of the side navigation to 250px */
function openNav() {
    document.getElementById("mySidenav").style.width = "250px";
}

/* Set the width of the side navigation to 0 */
function closeNav() {
    document.getElementById("mySidenav").style.width = "0";
}
```

***Figure 1.2***

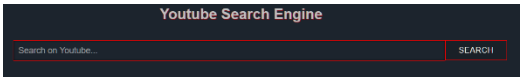
The next step was to create a way for the user to navigate through the website. A simple normal navigation bar with a dropdown menu was created for the website. I took the liberty of creating a more delicate design by adapting some features from Karim Balaa (http://codepen.io/karimbalaa/pen/WboBBY) as I have referenced in the code files as well.




The reason was because the specific design had some code lines I was not aware of, (Figure 1.4) where it made the background color and padding when hovering on top of each tab to fade and appear in a smoother motion.


***Figure 1.4***

Moreover, the YouTube search function was created by using some jQuery as well by following some of the YouTube API Data guide (https://developers.google.com/youtube/v3/docs/search/list). But just creating a search button, implementing and GET requests from the YouTube API was not enough to create a search engine.




In addition, I have the FancyBox tool that offers a nice and elegant way to add zooming functionality as well as present the data from the YouTube API not only as pictures and strings but in a video form. FancyBox jQuery library creates a lightbox when clicking on the YouTube output data that allows the user to see the YouTube video (Figure 1.5).

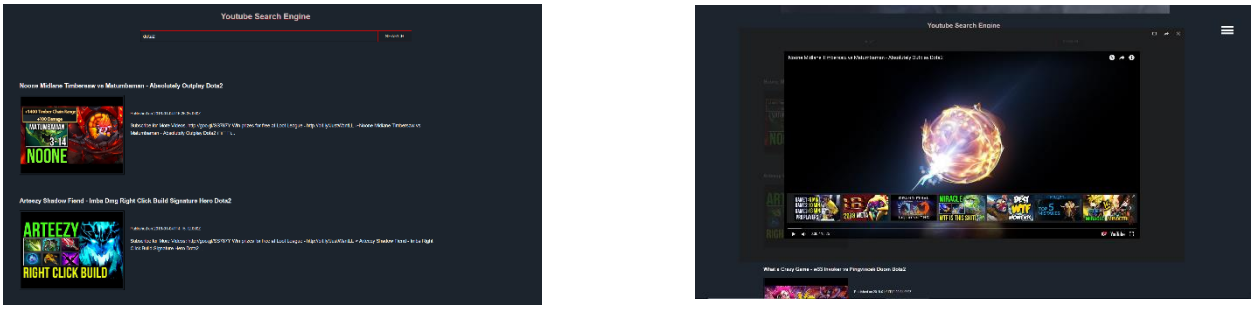



***Figure 1.5***

For the home page, I have also used another YouTube's functionality, Iframe API. The IFrame player API lets you embed a YouTube video player on the website and control it using JavaScript jQuery to create the video window that is loaded as soon as the home page (index.html file) run (Figure 1.6).

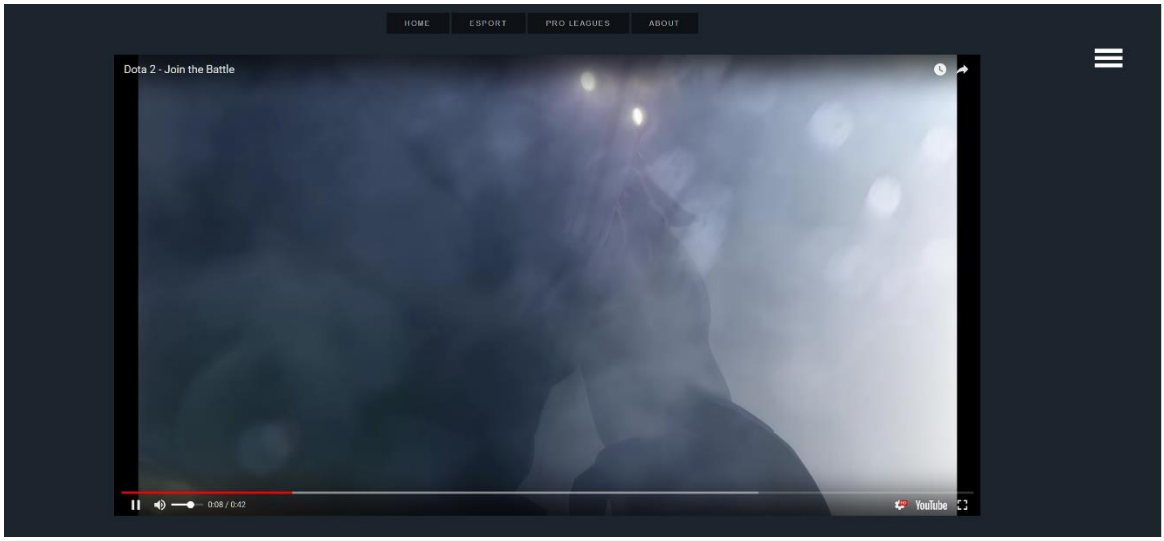



***Figure 1.6***

Furthermore, onto the Esport tab of the website I designed two sortable tables that get data from OpenDota API and PandaScore API that present the last 20 pro league matches that took place and their final scores as well as another table showing series, their tournaments history, and the pro teams that attended them. This was done purely in JavaScript by an XMLH get request (Figure 2.0 & 2.1). Similarly done, were the Player and Team Standings in the pro leagues tab (DotaTeam.html &DotaPlayer.html).


***Figure 2.0***

```
var tournaments_url = "https://api.pandascore.co/tournaments";
param = {
  token: API,
  // // tournament_id: "644"
  videogame_id: "4",
  // filter:"The-International-2016",
  // //serie_id: "1324"
  // sort: "name",
  "search[name]": "Playoffs"
}

var assembleTournaments = function(parameters){
  var query_strings = [];
  for(var key in parameters) {
    if(parameters.hasOwnProperty(key)){
      var param_string = encodeURIComponent(key) + "=" + encodeURIComponent(parameters[key]);
      query_strings.push(param_string);
    }
  }
  return query_strings.join("&");
}

var tourni_url = tournaments_url + "?" + assembleTournaments(param);
console.log(tourni_url);

var request = new XMLHttpRequest();
request.open("GET" , tourni_url);
request.addEventListener('load' , function(){
  var tourni = JSON.parse(this.response)
  console.log(tourni);
  for(var i = 0; i < 8;i++){
    var output = getTourniTable(tourni);

    document.getElementsByClassName('theTournamentbody')[0].innerHTML += output;
    z++;
  }

});
request.send();
```

***Figure 2.1***

For the latter pages, I have also created another YouTube search button inside the side navigation bar for the users to search for videos onto the same page without having to go back to the home page.

In addition, the "about" page is describing some of the game's main features and has a basic guide for the user to understand the way Dota2 is played. The last function that I have added to the website was the "spinner". By creating two functions in the JavaScript files (Figure 2.2) and the appropriate design, whenever a data table is loading inside the pages a loading icon pops until the tables are fully loaded (Figure 2.3).

```
////////////////Loading icon functions////////////////////////
var timeOut;
function stopLoader() {
    timeOut = setTimeout(showLoader, 1000); //amount of time the icon is shown
}

function showLoader() {
  document.getElementById("loader").style.display = "none";
  document.getElementById("table_box").style.display = "block";
}
```

***Figure 2.2***

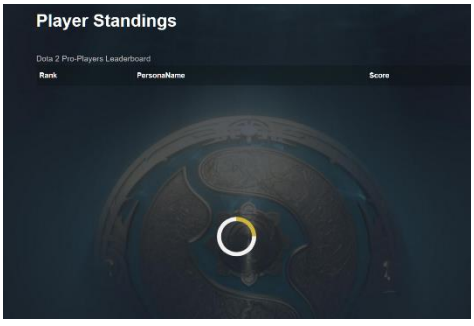



***Figure 2.3***

# Critical View and Learning Outcomes

Based on the assignment we were given I believe I have included all the features we were asked to create front-end website that allows the user to view and interact with data from multiple APIs.

I have also added some extra features that give a nice aesthetic to the website. For example, the lightbox that appears when use clicks onto a YouTube video, that was created by using a jQuery library from the fancyBox tool, give a more elegant look than a plane player on the page. Although jQuery is getting outdated, I had to use it as it was the only way possible to my knowledge to "translate" the YouTube Data API from a string to a video format (see figure 3.0).

On the other hand, I believe I could have done a better job on the design part as it is not as unique as I wanted it to be. In addition, I would also like to have included more information on the Esport part of the website as well as a live match table using the live pandaScore API.

Throught this coursework, I have learned a lot as it was my first time working with APIs, and I believe I have implemented correctly what I have learn. The websites' HTML pages have the correct structure and a variaty of different elements are used. In the "screen shots validation" section of this report you can see that I have validated all the html and css files which I believe this shows the correct use of the elements used.

In the end, I believe my website has a consistent and well-ordered code, and a visually appealing design. It was a challenging coursework, and it took a lot of time for me to get the hang of APIs and how to retrieve GET requests, manipulate them and display their data to the website. Then again, I enjoyed this coursework a lot and I would want to work on projects like this in the future.

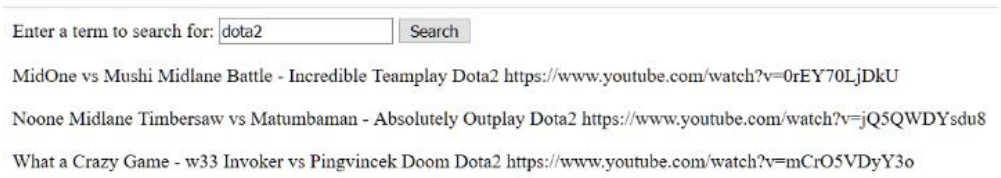
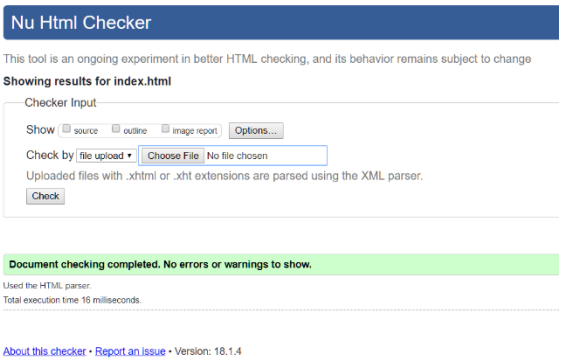


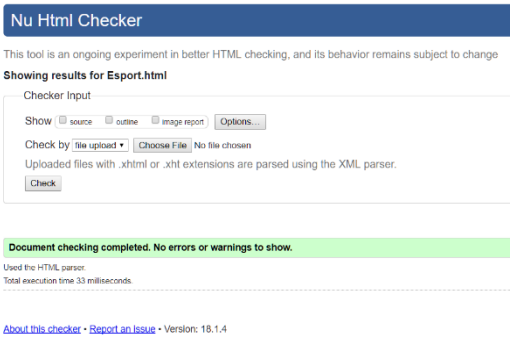

***Figure 3.0***

# Screen shots Validation

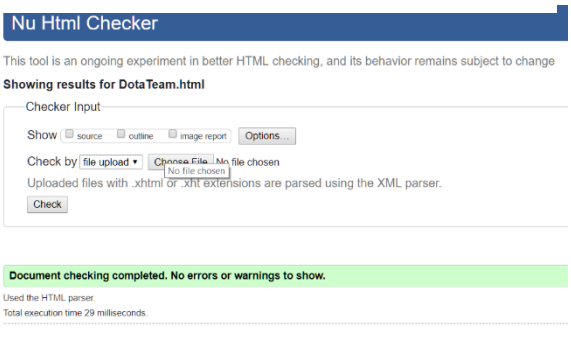
## HTML



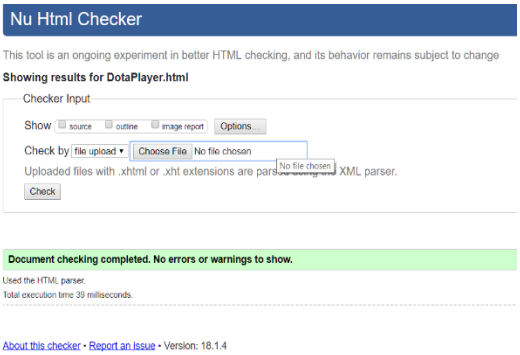

***Index.html***



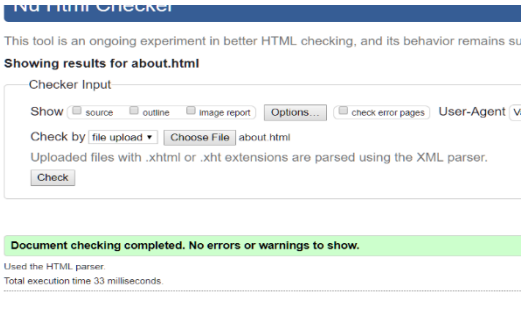

***Esport.html***



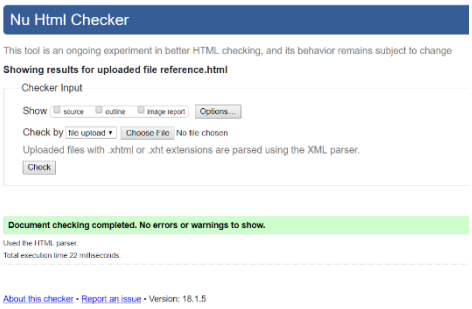

***DotaTeam.html***




***DotaPlayer.html***



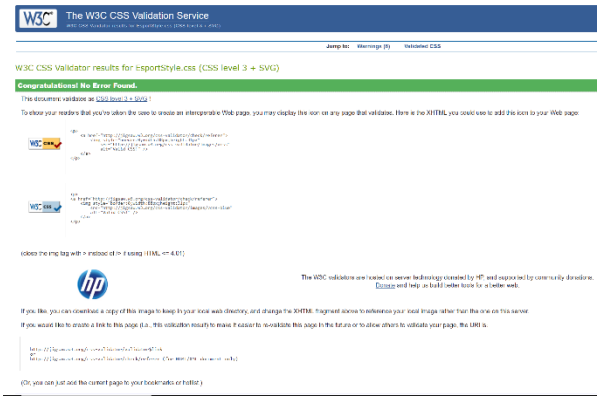

***About.html***




***Reference.html***

# CSS

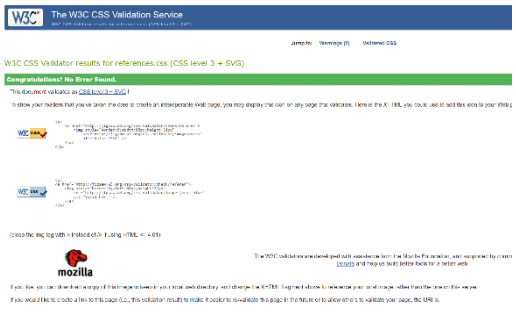

***Style.css***




***EstportStyle.css***


***SotaStyle.css***


***aboutStyle.css***




***References.css***

# Reference List


1. YouTube Search Engine Paging Buttons Lightbox by Merab Gvantselatze, Apr 1,2016, https://www.youtube.com/watch?v=fljo4EPjUDk
2. Copyright © 2008 - 2018 fancyApps. All Rights Reserved, http://fancyapps.com/fancybox/3/docs/#iframe
3. Add youtube functionality by google developers, last updated November 16, 2017, https://developers.google.com/youtube/v3/docs/search/list
4. Google hosted libraries, last updated December 18, 2017, https://developers.google.com/speed/libraries/
5. Valve Corporation, Dota 2, 2017, http://www.dota2.com/play/
6. Open Dota API, Powered by ReDoc , https://docs.opendota.com/
7. Design Inspirations, private Cloud by Stackscale, https://www.awwwards.com/websites/
8. Argn0, 23rd December 2017, OpenDota API parameter commands, https://github.com/odota/web/blob/master/src/lang/en-US.json
9. Tyleruebele, Sort-Table, April 30, 2014, Released under MIT license, https://github.com/tyleruebele/sort-table
10. Loading icon (Loader CSS) by W3Schools, 1999-2018,https://www.w3schools.com/howto/howto_css_loader.asp