Final Report

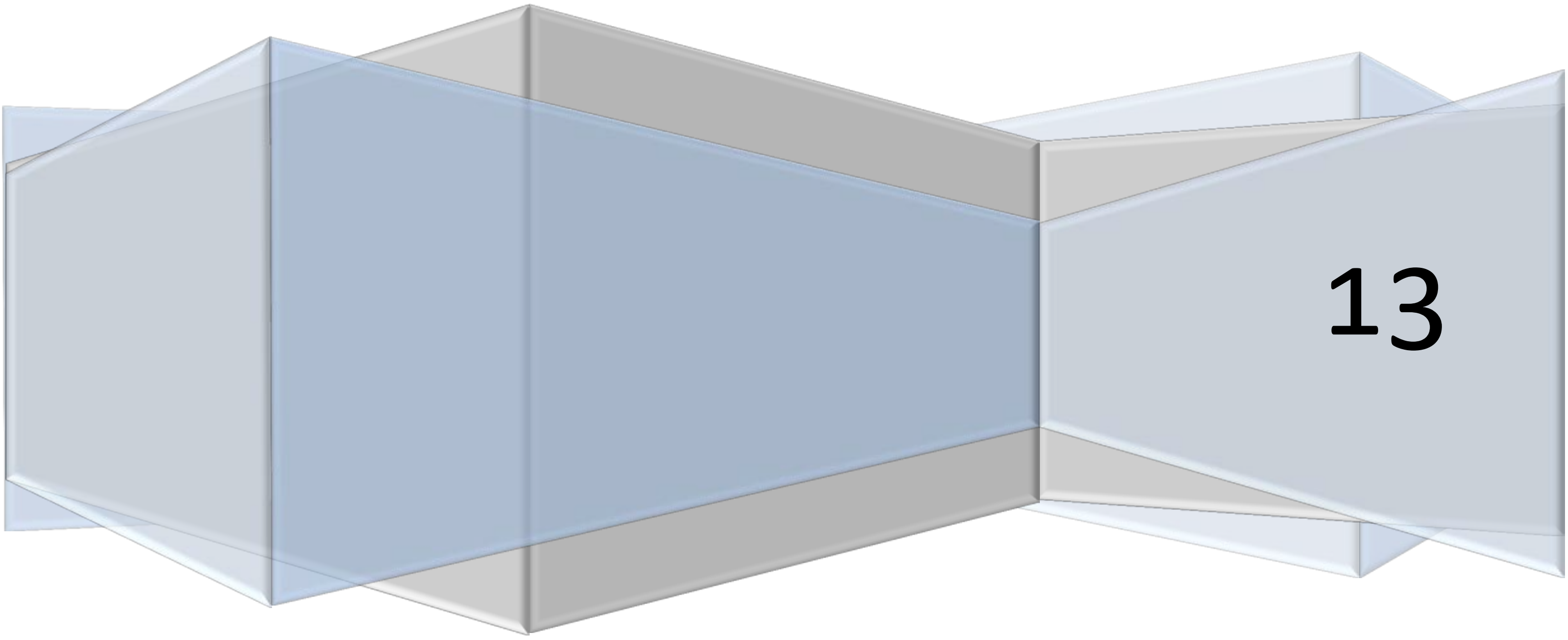
Social Media Visualisation Tools and Techniques

Matthew R Jones

Supervisor: N.J.Avis

Moderator: K.Sidorov

CM0343– 40 Credits



Acknowledgements

I would like to express my very great appreciation to Dr Nick Avis for his valuable and constructive suggestions during the planning and implementation phases of this project. His insights into the project area have been very valuable.

I would also like to provide thanks to the Computer Science department at Cardiff University, for providing the server space I required to work on the implementation of my project.

I would also like to express my thanks to Mr Steffan Walters for taking the time to assist me in evaluating my completed application and providing valuable feedback.

Contents

Introduction

This project is intended to work alongside the research that the Cardiff School of Computer Science and Informatics do with the School of Social Sciences at Cardiff University to create a toolkit of techniques that can be used in the harvesting, analysis and visualisation of social media content. It is intended that the tools that are researched and developed would be useful for researchers in the area of social science and also perhaps a wider scope of audiences such as the emergency service or even private sector companies. I have been looking at content, such as that posted on social media services with an emphasis on the content that is posted on the Twitter. As social media now plays such a large role in the lives of many, it is becoming increasingly more prevalent that the material posted on such services needs to be analysed and considered.

An example of the type of information I have been looking at, is that of tension indicators which may be present within different demographics of the population.

During the interim reporting phase, I commented on my findings of similar services which aimed to make visualisation techniques available to its users from social media data. Following on from this research and also research into techniques of visualisation that are ‘state of the art’, I have then been able to build up a problem solution and provide an implementation of a web application which allows for both the harvesting and visualisation/analysis of social media data, from real time online sources.

Throughout the project I have kept in mind that the intended users of this system would be Social Science researchers who would be able to use these tools for visualising and analysing social media content. These tools could aid them in choosing appropriate techniques and ways to represent data for the task they are carrying out. This being the case, it has been imperative that the toolkit that I have been implementing, is easy to use and understand by those who may not necessarily have a background in technology and may have a limited understanding in how to use visualisation and analysis techniques to suit their needs.

The project has needed different types of analysis and visualisation techniques to be considered and in doing so, I have been determining their suitability’s and capabilities for use within the COSMOS (Cardiff Online Social Media Observatory) initiative and also on a research level, in order to provide a toolkit that is adaptable to the differing needs of the users.

I will also be providing an evaluation of the results of my application development during the course of this report.

The aims which I had set out in the initial project plan that would be completed by the final report are:

- Develop a system to aid Social Science researchers in choosing an appropriate visualisation technique to use for their selected dataset
- Look at how using different API's (provided by Twitter, Google etc.) can benefit the retrieval and basic visualisation of information on the web
- Allow the developed system to be adaptive in learning what visualisation techniques are selected most often by Social Science Researchers and on which types of data
- Test the developed application with Social Science Researchers to gauge feedback into how the system works and if the types of visualisations are appropriate for the datasets they would be using for the research that they are conducting

There have been very few changes in the project from the initial brief I was given, the only slight change has been my interest in the real time collection (harvesting) of data from social media services, and more specifically the tweets that are posted on Twitter. I thought that this collection of data was a key part in starting to build up a toolkit to analyse and visualise this data.

Design

Choice of Application

One of the biggest decisions I faced when thinking about my approach to solving this problem, was whether or not I would develop the application as a desktop piece of software (that is one that would only run on an individual machine) or whether I would make it a web service that would be accessible to a wider range of users. Despite my initial feeling for wanting to create a desktop application, following my research into similar services, which I reviewed as part of my interim report, I then favoured the idea of developing a web application. My decision was then further backed when I took into account the information I had obtained on various APIs and technologies that I could employ whilst developing the application.

Having decided to base my solution to this problem around a web application I was then able to make further decisions as to the different APIs I would use as part of my application. These would include Twitter's API and also the Google Maps API. I have used the Twitter API to aid in the retrieval of tweets from the social media service. I have also used Google's Maps API to help me map my tweets and also as a basis for my heat map overlaying function.

Technologies Used

Tweet Retrieval

In order to retrieve tweets from the twitter social networking service to allow for visualisation and analysis of them, it will be necessary for me to use an API (Application Programming Interface) provided by twitter. The API that I decided on using was their PHP version of the API as this would allow for an easy integration into the application for requests to be sent. The data returned would also be provided in a form that was relatively easy to use in terms of creating the visualisations and also storing it as part of a larger dataset.

Google Maps

Following on from my research into the available technologies that I could utilise to create a mapping service for my tweets, I have decided to use the JavaScript Maps API as provided by Google. It was apparent to me that this API provided a good range of features and would also be helpful in my own personal development with regards to learning the JavaScript language.

The JavaScript API is currently on its third version and is used by a wide range of companies and services including the New York Times. If the API is implemented correctly, then it has the ability to allow the user to move around the map using both the mouse and keyboard as navigational tools. It also allows for zooming into and out of regions of the map so the user can get a better view of certain areas if they wanted or even a broader view over a larger area.

In using this technology, I have also hoped that it would be something that was fairly familiar to anyone who had used Google Maps as a mapping service on the internet as it would have a similar look and feel to that service. By using the Google Maps API it is possible to change the style of the map, e.g. satellite or hybrid. It is also possible to overlay information onto the map using markers, something I would take advantage of within this project.

Data Storage

To be able to store the tweets that are retrieved from twitter I have used an SQL Database. This allows you to easily create data tables to your own specifications. For my implementation I have been able to create one that will store all the main part of the tweets that are retrieved using the Twitter API, this includes:

- Tweet Keyword
- The Tweet Itself
- Geo-location information of the tweet
- Date and Time

The SQL language (Structured Query Language) which is a special purpose programming language used to manage data in database systems, integrates well into web applications as you can use it alongside both PHP and HTML. This has allowed me to easily build in SQL queries to my tweet retrieval method to store new tweets in the database and also in retrieving previous tweets from the database which I have used in many of the other features of the application.

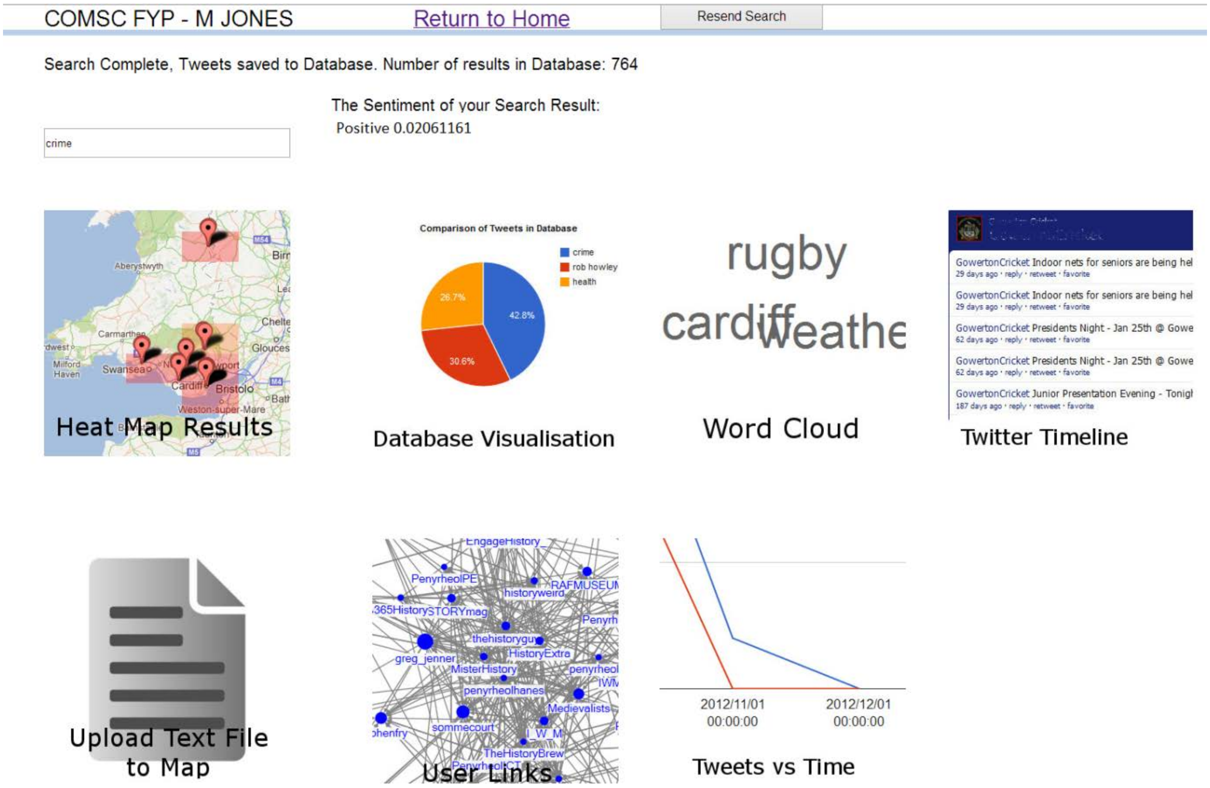
Interfaces

Designing interfaces for a system is always a tricky process as you wish to keep them as plain and simple as possible most of the time, whilst also managing to impart all the information that you need to the intended users. In designing the look of my interface I chose to keep it as clean and free of clutter (anything unnecessary) as possible and in doing this I hoped that it would make the application accessible and easy to use by as many people as possible, especially those who are not technically minded.

With this design strategy in mind, I then created the interfaces for my website so that they keep users well informed as to what is happening within the application through clearly labelled buttons and dialogs. Where instructions have been needed as to how to use certain visualisation services I have made sure that these are provided and that they are easily understandable.

I have tried to keep the navigational elements of the website constant throughout by providing navigational buttons to the applications ‘home’ and ‘dashboard’ in the same places on each web page that I have used. I have made sure that all text that is displayed uses an easy to read, standard font type. Furthermore I have kept the designs of each web page as similar as possible so that the users feel as though they are using a well-integrated, cohesive system.

Whilst thinking about the design of my interfaces I have also taken into account that any errors which could arise within the web application (whilst hopefully few and far between) need to be handled appropriately. I have therefore tried to provide a meaningful way for the system to present any error messages to the user which would allow the user to identify the nature of the problem and hopefully allow them to correct any issues that may have occurred.



Search Complete, Tweets saved to Database. Number of results in Database: 764

The Sentiment of your Search Result:
Positive 0.02061161

GowertonCricket Indoor nets for seniors are being hel
29 days ago · reply · retweet · favorite
GowertonCricket Indoor nets for seniors are being hel
29 days ago · reply · retweet · favorite
GowertonCricket Presidents Night - Jan 25th @ Gowe
62 days ago · reply · retweet · favorite
GowertonCricket Presidents Night - Jan 25th @ Gowe
62 days ago · reply · retweet · favorite
GowertonCricket Junior Presentation Evening - Tonigl
187 days ago · reply · retweet · favorite

Figure 1 - Interface - Dashboard Screen

Overall System Design

Whilst implementing my system I have been able to build up a site map (a diagram) showing the various web pages my project utilises, this has been helpful in my establishing the flow of data through my system. (See Next Page)

The flow of data through my system mainly stems from the retrieval of tweets from Twitter and also the use of previous data contained within the database. Applications such as the:

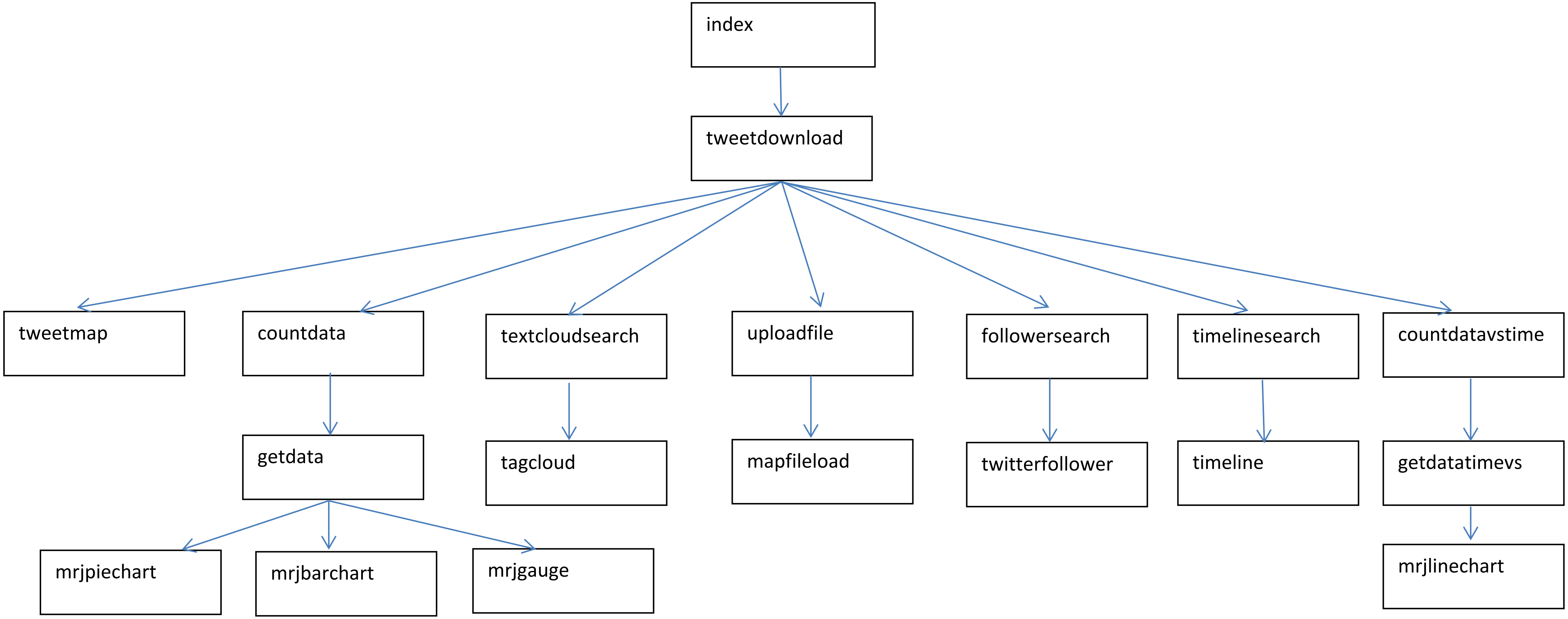
- Heat Map
- Database Visualisation
- Tweet vs. Time
- Sentiment analysis

Then use this data that has been created and stored within the database.

There will only be one feature that makes use of the users own data, which is the 'Upload Text File to Map' functionality, which will allow users to enter their own data in a predefined format.

By maintaing my sitemap from the start of my implementation it has also allowed me to easily keep track of which files I needed to access if any changes need to be made. It also provided me with reference points when trying to code in the transfer of data between the various web pages I was using.

Sitemap of Project 150



This sitemap shows the filenames of the various web pages that I have used during my implementation, it also gives a broad overview as to the data transfer, in terms of the PHP variables that are sent around the application.

Implementation

Approach

As I had decided from my design phase that the application I was going to develop would be a web based one, I then set about finding out how I was going to host the website I created in order for me to be able to develop and test it. I decided upon using the University's project space that is allocated to all students, as this would perform all of the functionality I would require, allowing me to upload my code to the server from my own work station using a SFTP (Secure File Transfer Protocol) file transfer protocol. Once the code had been uploaded to the servers I would then be able to test the functionality of the features I had implemented.

The FTP application that I have used has varied according to which system I have been using, on windows I have made use of the WinSCP software, where as if I have been implementing on a Mackintosh computer I have made use of the Cyberduck piece of software. Both of these as well as other FTP applications that you can use perform the same tasks, where they establish a connection with the server you are using by supplying your login credentials. They allow you to upload the files and for these to be copied across from your workstation to the server securely. In addition to this some of them allow you to perform some editing of the code on the side of the server, however, it is normally easier to edit code in another application and then upload it.

To create and edit my code, I have mainly used text processing applications such as TextPad on a windows machine and the Xcode package on the Mackintosh computers. Applications such as these are useful as once they establish the types of files you are working with, for example HTML and PHP, then they can highlight the syntax of the code and format it neatly so that it is easy to read and edit.

My first port of call would be to design some form of interface which I hoped I would be able to keep the same throughout my project. Once I had created this I would be able to use the code as a template for the various web pages I would be creating, this would be achieved using a combination of HTML and CSS style sheets.

CSS style sheets are a style sheet language that is used to describe the presentation of a document that is written in a mark-up language, of which HTML is a part of. Alongside HTML, CSS style sheets can also be used to format XML documents which can often be useful when dealing with the reply responses from different APIs.

Having developed a suitable interface and a starting point for my home page I could then work on developing the individual visualisation features that my application would use. I decided to start with my heat mapping visualisation of geo-located tweets as this was to be the main feature of the application as stated in my initial report. Once I had created the heat mapping service I would then continue to look into the different types of visualisations I could include within this application, looking at both their usefulness and also prioritising them on their ease of implementation and their vitality to the system. Alongside the background research I had conducted as part of the interim reporting I was then able to make informed decisions as to the visualisations I would include.

My implementation method would be one whereby I would develop the different visualisations one by one, testing each visualisation as I went through development, this would mean that I could easily keep track of my workload, working alongside my time plan, and also prioritise my time accordingly towards different aspects of the system if they needed it. Once I had developed the tweet retrieval functionality, I was then able to use the system with real test data that was downloaded from Twitter, this would mean that I would not have to create my own set of test data. It also means that in testing each of my implemented features I could see exactly how they would interact with real life data.

Features of the Application

The Dashboard

The dashboard of the application is the web page that the users are taken to immediately after they have submitted their original keyword search on the home page. This is where all the visualisations that they can take advantage of are displayed. It is in the implementation of this page that the sentiment analysis is instantiated, as well as the retrieval of new tweets from Twitter (Figure 2).

```
<?php
//Include the keyworddownload script, also get the term entered in the search form
include("keyworddownload.php");
$searchTerm = $_POST['formSearchTerm'];
$twitter = new mrjTwitter();
$twitter->connectAndRetrieveOldTweets($searchTerm);
$results = $twitter->getNewTweets();
$twitter->newSaveTweetsToDataBase();
// Load the AlchemyAPI module code.
include "AlchemyAPI.php";
// Create an AlchemyAPI object.
$alchemyObj = new AlchemyAPI();
// Load the API key from disk.
$alchemyObj->loadAPIKey("api_key.txt");
// Extract sentiment from a web URL.
$result = $alchemyObj->URLGetTextSentiment("https://twitter.com/search?q=$searchTerm&src=typd");
?>
```

Figure 2 - Implementation of the Dashboard

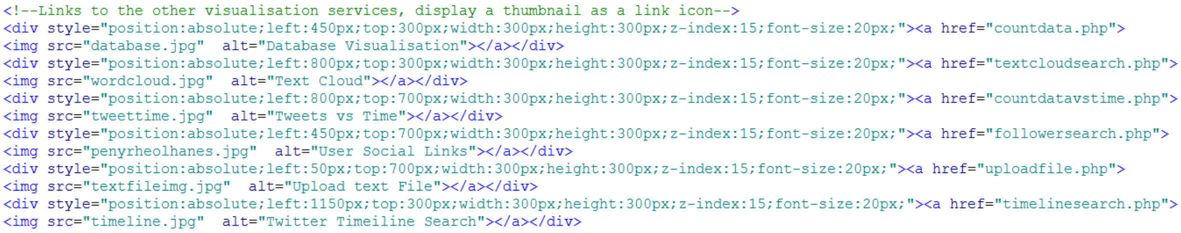
This code shows the inclusion of the file '*keyworddownload.php*' which is where the functions for the Twitter API are stored. It then calls a new version of the code function '*mrjTwitter*'. Also called at this stage are the functions to connect to the database, to retrieve the tweets that are already stored in it relating to the keyword that has been entered on the homepage. The code in this opening statement also makes a call to the code function that connects to the Twitter API so that it can download a set of new tweets from Twitter relating to the keyword searched. Finally these new tweets are then saved to the database so that they too will be included in any future searches for the same keyword that may be made.

The sentiment analysis result of twitter is then called and the result is displayed appropriately on the web page using a HTML 'Div' object (Figure 3)

```
<!-- Divs that contain the Sentiment analysis results -->
<div id="Text1" style="position:absolute;left:400px;top:160px;width:600px;height:32px;z-index:13;">
    <span style="color:#000000;font-family:Arial;font-size:20px;">The Sentiment of your Search Result:</div>
<div id="Text1" style="position:absolute;left:400px;top:180px;width:600px;height:32px;z-index:13;">
    <span style="color:#000000;font-family:Arial;font-size:27px;"></span><?php echo $result; ?></div>
```

Figure 3 - Displaying the Results of Sentiment Analysis

I then implemented the ability to display thumbnails on the dashboard of how the visualisations would look, too allow the users to be able to decide which ones they wished to use. I was originally hoping to achieve this by providing live images, that is creating a view of how they would actually look if they were applied to the dataset being used by the user, however, it soon became clear that for ease of programming and also ease of conveying to the user, that I would display these thumbnails using static images. I created these static images by taking screenshots of how the visualisations looked when they had some test data applied to them. I have then used these thumbnail images as hyperlinks in order to link the user to the relevant web page they need to be directed too (Figure 4). Each image is contained in its own ‘div’ tag so that I could control where it was positioned on screen, there is also some alternative text that can be provided in case the image cannot be loaded for any reason. By including this alternative text then it means the application would then still inform the user of which button they could use for each visualisation. This makes use of HTMLs ‘alt’ feature as part of the ‘a’ tag.



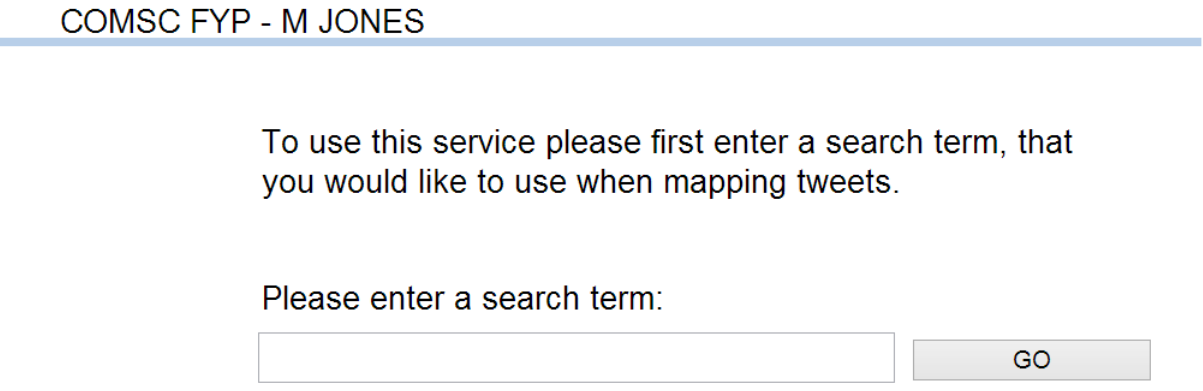
```
<!--Links to the other visualisation services, display a thumbnail as a link icon-->
<div style="position:absolute;left:450px;top:300px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="countdata.php">
<img src="database.jpg"  alt="Database Visualisation"></a></div>
<div style="position:absolute;left:800px;top:300px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="textcloudsearch.php">
<img src="wordcloud.jpg"  alt="Text Cloud"></a></div>
<div style="position:absolute;left:800px;top:700px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="countdatavstime.php">
<img src="tweettime.jpg"  alt="Tweets vs Time"></a></div>
<div style="position:absolute;left:450px;top:700px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="followersearch.php">
<img src="penyrheolhanes.jpg"  alt="User Social Links"></a></div>
<div style="position:absolute;left:50px;top:700px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="uploadfile.php">
<img src="textfileimg.jpg"  alt="Upload text File"></a></div>
<div style="position:absolute;left:1150px;top:300px;width:300px;height:300px;z-index:15;font-size:20px;"><a href="timelinesearch.php">
<img src="timeline.jpg"  alt="Twitter Timeiline Search"></a></div>
```

Figure 4 - Displaying the Thumbnails

The screenshots of the different visualisations hopefully give a clear representation to the user as to how their chosen visualisation could look once it has been applied to their test data.

Implementation of Interface

Implementation of the interface has taken place using a range of different web technology programming languages, the main ones of which have been HTML and PHP. Whilst implementing my interface I tried to keep in mind the design that I would be using for my interface, it was my intention to keep the design of the website as clear and easy to use as possible. I have used a clear white background with clear black text set against it. Hyperlinks are provided in the HTML convention of being blue and underlined, where I have used forms for my data collection then a clear button is provided so the user knows where to click after they have completed the form.



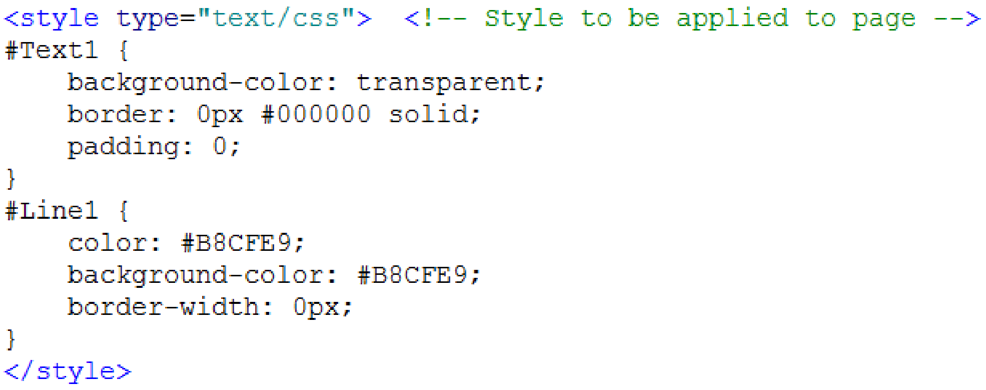
COMSC FYP - M JONES

To use this service please first enter a search term, that you would like to use when mapping tweets.

Please enter a search term:

Figure 5 - An Example of the Interface from the Home Page

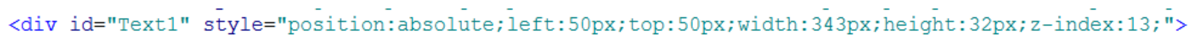
In implementing each of the different web pages that are used to provide the visualisation services, I have decided to keep the same interfaces, this has allowed me to make use of common elements of the CSS style sheet when coding each page (Figure 6). This also allows the feel of the website to flow naturally for the user, which means that the common navigational links are always apparent at the top of the page for the user.



```
<style type="text/css">  <!-- Style to be applied to page -->
#Text1 {
    background-color: transparent;
    border: 0px #000000 solid;
    padding: 0;
}
#Line1 {
    color: #B8CFE9;
    background-color: #B8CFE9;
    border-width: 0px;
}
</style>
```

Figure 6 - An Example of the CSS Code Used

The navigational links that are provided on every webpage are created using the 'href' attribute of HTMLs ‘a’ tags which allow you to specify the target address for the link to be directed to. Positioning of most elements on the webpage including text and images is achieved using HTML ‘div’ tags. This allows for ease of specifying positional information during implementation as you can specify the number of pixels from both the top of the page and the left of the page that the element will be positioned at (Figure 7). It also allows you to specify the width of the textbox.

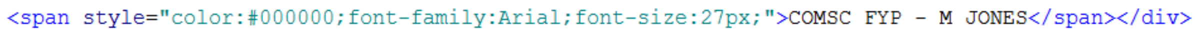


```
<div id="Text1" style="position:absolute;left:50px;top:50px;width:343px;height:32px;z-index:13;">
```

Figure 7 - HTMLs Style within the 'div' Tag

Where text has been used on webpages, I have used HTMLs ‘span’ tags in order to allow me to be able to control properties such as the texts:

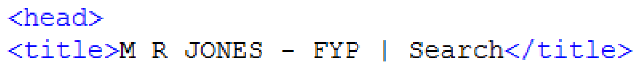
- Colour
- Font Type
- Font Size



```
<span style="color:#000000;font-family:Arial;font-size:27px;">COMSC FYP - M JONES</span></div>
```

Figure 8 - HTMLs Style within 'span' tags

Also within the design I have made sure that the title of the browser page or tab is set so that the user can clearly see which window is for this application if they were to navigate away to another webpage in a different tab. This is achieved using an element that belongs to the HTMLs Head section, the tag that is used is the ‘title’ tag (Figure 9).



```
<head>
<title>M R JONES - FYP | Search</title>
```

Figure 9 - Web Page Title Specified in 'title' Tags

Tweet Retrieval

The retrieval of tweets from twitter was to be the main way of collecting data and inputting it into the system for my application.

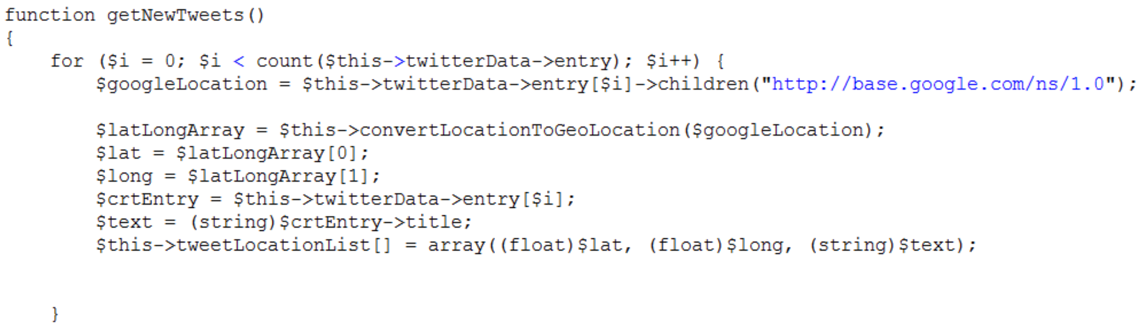
I would be retrieving social media data from twitter by harnessing the Twitter API[1], this would allow me to query twitter for any tweets containing a specific keyword, as input by the user of my system, the keyword is then passed through to the twitter API and as such, Twitter will return a collection of relevant tweets which match the specified query. This makes use of the call ‘GET search/tweets’. As such this is not an exhaustive search of all tweets on twitter and indeed in my implementation of the system, due to the limit on the number of API calls that are able to be made, I have limited this call of the API to return 15 results per query.

Tweets that are returned from the Twitter API are returned in the JSON format and are inputted into my database by using SQL queries, the information that is contained in each response from Twitter is:

- Text contained in the Tweet
- Latitude
- Longitude

JSON is part of the JavaScript Language and is used for data interchange. It is a way of representing simple data structures that are human-readable. Even though it is related to JavaScript it can however be used with many other languages through the use of parsers.

From here, I would also process the keyword that they were using to retrieve tweets and I would be storing this in the database alongside the downloaded twitter content. I would also store the date and time that each request was sent.



```
function getNewTweets()
{
    for ($i = 0; $i < count($this->twitterData->entry); $i++) {
        $googleLocation = $this->twitterData->entry[$i]->children("http://base.google.com/ns/1.0");

        $latLongArray = $this->convertLocationToGeoLocation($googleLocation);
        $lat = $latLongArray[0];
        $long = $latLongArray[1];
        $crtEntry = $this->twitterData->entry[$i];
        $text = (string)$crtEntry->title;
        $this->tweetLocationList[] = array((float)$lat, (float)$long, (string)$text);


    }
```

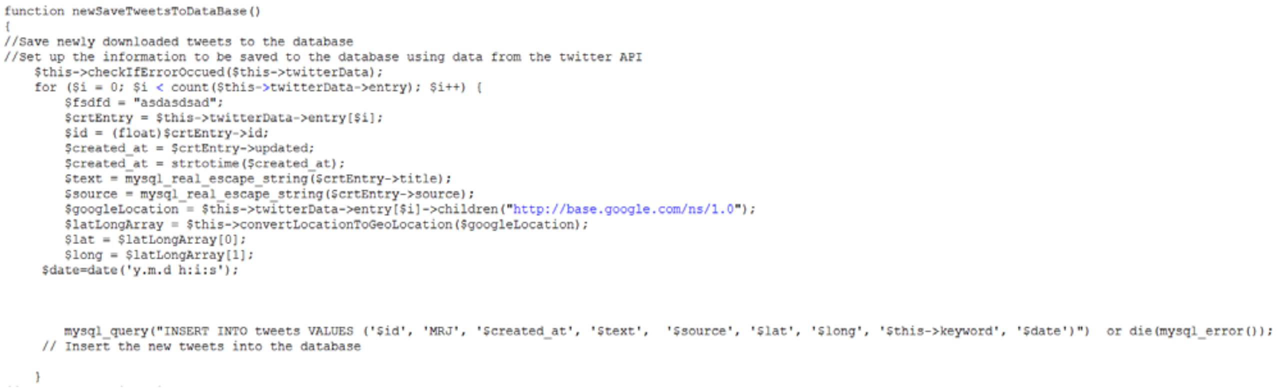
Figure 10 - Retrieving New Tweets

The retrieving of tweets does rely on some other functionality for it to be useful in my application especially seeing as I would need to make use of the locational information provided with the tweets. This meant that if tweets were provided with locational information in a textual context, that is a place name e.g. Cardiff then I would rely on using a

Google API to convert the place name into a geo-location that consisted of Latitude and Longitude coordinates The Google (Figure 1). API that I have employed is the Geocoding API. As with the other APIs I would be using, this too had some usage limitations and my implementation of it would be subject to a limit of 2,500 requests per day. The output of the Geocoding API is returned in JSON format, which returns two root elements;

- Status – this contains metadata on the requests that have been sent and also includes any error codes that may be generated.
- Results – this is where the information that I would need is returned, it is an array of the geocoded address information.

After retrieving the new tweets from Twitter they are then inserted into the database (Figure 11). This is achieved by using the SQL Query 'Insert into *tweets* Values(…..)' where 'tweets' is the name of the database table that I would be using and the values would be those returned from the twitter API call along with the keyword and request time.



```
function newSaveTweetsToDataBase()
{
//Save newly downloaded tweets to the database
//Set up the information to be saved to the database using data from the twitter API
    $this->checkIfErrorOccued($this->twitterData);
    for ($i = 0; $i < count($this->twitterData->entry); $i++) {
        $fsdfd = "asdasdsad";
        $crtEntry = $this->twitterData->entry[$i];
        $id = (float)$crtEntry->id;
        $created_at = $crtEntry->updated;
        $created_at = strtotime($created_at);
        $text = mysql_real_escape_string($crtEntry->title);
        $source = mysql_real_escape_string($crtEntry->source);
        $googleLocation = $this->twitterData->entry[$i]->children("http://base.google.com/ns/1.0");
        $latLongArray = $this->convertLocationToGeoLocation($googleLocation);
        $lat = $latLongArray[0];
        $long = $latLongArray[1];
     $date=date('y.m.d h:i:s');


        mysql_query("INSERT INTO tweets VALUES ('$id', 'MRJ', '$created_at', '$text',  '$source', '$lat', '$long', '$this->keyword', '$date')")  or die(mysql_error());
     // Insert the new tweets into the database

    }
```

Figure 11 - Saving Tweets to the Database

The user of the application does not see any physical evidence of the tweets being downloaded or saved to the database. This process is performed in the background as the transition is made between the home page (search) and the application's dashboard. However, when they are then taken to the dashboard of the application, a message is displayed detailing the number of tweets that are stored within the database for the users chosen keyword. The dashboard is the area of the application (web page) where the user is able to choose the visualisations that they will apply to their data set.

Whilst the Twitter API does most of the legwork for retrieving tweets from Twitter, the challenge was in implementing it into the application so that it could work as a background process and so that the data that was returned could then be processed into a suitable format. A lot of my time here was spend of trying to integrate the results that I was receiving from twitter into the database I had created.

Heat Map Visualisation

The heat map visualisation allows the users of the system to plot the tweets that are downloaded during tweet retrieval and also those that are stored in the database. The plots are made on a map by positioning a marker which is the location of where the tweet was posted (tweeted) from, onto the map at the given location.

Hovering over a marker that has been plotted on the map brings up a window containing the tweet that was sent from that location. The heat map functionality allows users of the system to easily see areas where more tweets containing their chosen keyword are sent from, through a colour overlay being applied to the map. A colour scheme from yellow for a low tweet density through to red for high tweet density show, with ease, different areas of the country for comparisons (Figure 12).

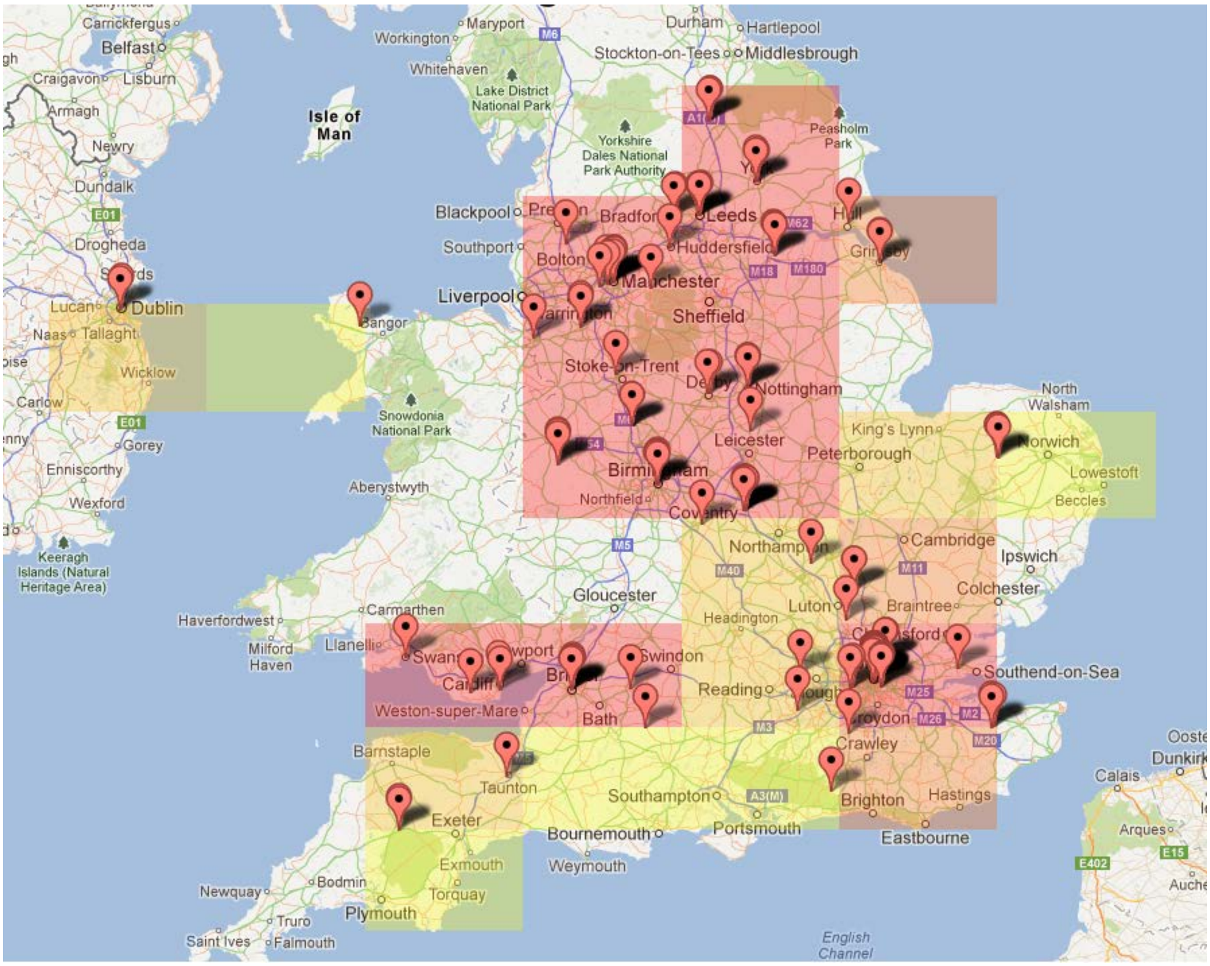


Figure 12 - Heat Map Allows for Easy Visualisation of Tweet Density

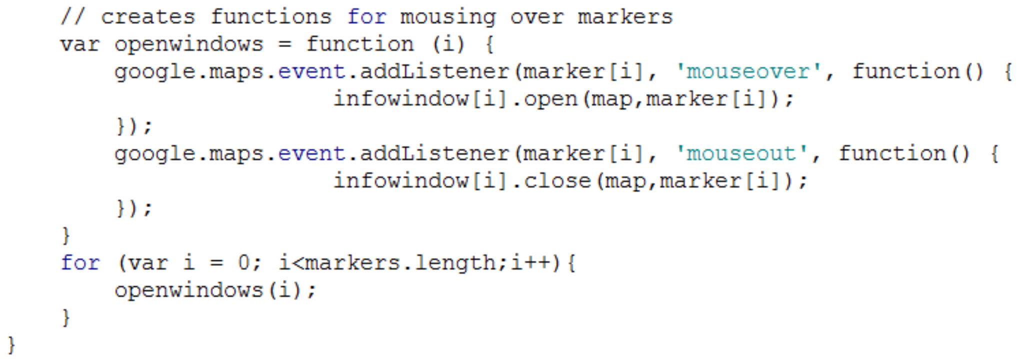
The mapping of tweets primarily makes use of Google's API[2] for maps. I decided upon using the JavaScript version of this API as it would provide the most functionality for users and also meant that I would be able to code in my own functionality for providing the heat map overlay, a feature Google does not provide.

Whilst the basic mapping functionality is provided by the Google API, some difficulties arose in trying to access the correct data from the database. Here trying to provide the Google API with the coordinates of tweet locations was fairly difficult.

The main challenge in developing this feature of the application was providing the heat map functionality, as this is not something that Google Provides, I had to spend a considerable amount of time trying to work out a suitable way of programming the heat map, and then trying to integrate my programming of it, in JavaScript to work alongside the Google Maps API.

The heat map code aims to look at areas of the country where tweets have been posted and determines how many tweets have been plotted there. It then looks to create a scale comparing different areas of the country to the scale that has been created. This then takes into account whether or not the current area it is trying to determine, is in an area that has a high density of tweets or not. The rectangles that are then created to show the heat mapped areas take a specified radius out from the tweet, this is a dynamic radius as it can be altered if the zoom level of the map changes and then the heat map of that area is re-calculated. As such the rectangular areas provided by the heat map are intended to show a broad overview if the map is zoomed out and a more detailed aspect if the map is zoomed in.

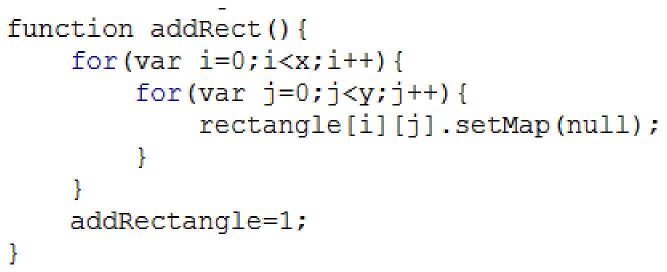
To display the information window that contains the contents of the tweet when you hover over a marker, the Google API makes use of a listener to determine when the user has moved their mouse onscreen to hover over a marker. When the mouse is moved over the marker, an information window is opened and the correct tweet (for that location) is read out of the array that all the tweets on the searched keyword are held in, to be displayed within the marker. Once the mouse is moved away from the marker then the information window is closed. The code below (Figure 13) shows this.



```
        // creates functions for mousing over markers
        var openwindows = function (i) {
            google.maps.event.addListener(marker[i], 'mouseover', function() {
                            infowindow[i].open(map,marker[i]);
            });
            google.maps.event.addListener(marker[i], 'mouseout', function() {
                            infowindow[i].close(map,marker[i]);
            });
        }
        for (var i = 0; i<markers.length;i++){
            openwindows(i);
        }
    }
```

Figure 13 - Info Window Function (Google API)

There are buttons provided onscreen to allow the users to be able to select the areas of the map that they wish to perform the heat map function on. This is implemented by allowing the user to click on two points on the map; the first which is the top left hand corner of the rectangle for the heat map area, the second point would then form the bottom right hand corner of the heat map area rectangle. Within this new user created rectangle, the heat map functionality is then applied, thus allowing the user to choose more specific areas of the country to perform their analysis on. When choosing the areas of the map to perform the heat mapping function, the scale of the map is taken into account by the algorithm and as such adjustments are made to show a broader or more specific overview of different areas.

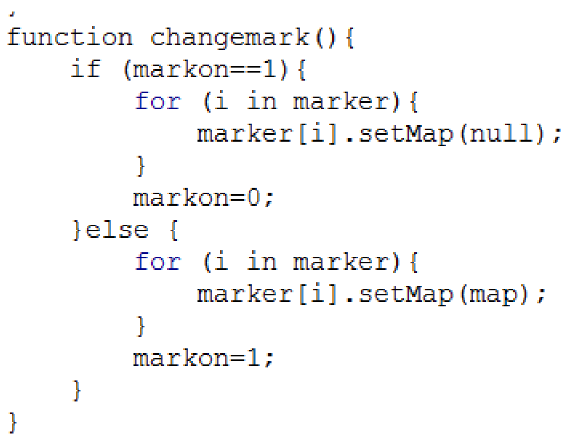


```
function addRect(){
    for(var i=0;i<x;i++){
        for(var j=0;j<y;j++){
            rectangle[i][j].setMap(null);
        }
    }
    addRectangle=1;
}
```

Figure 14 - User's Select Heat Map Area

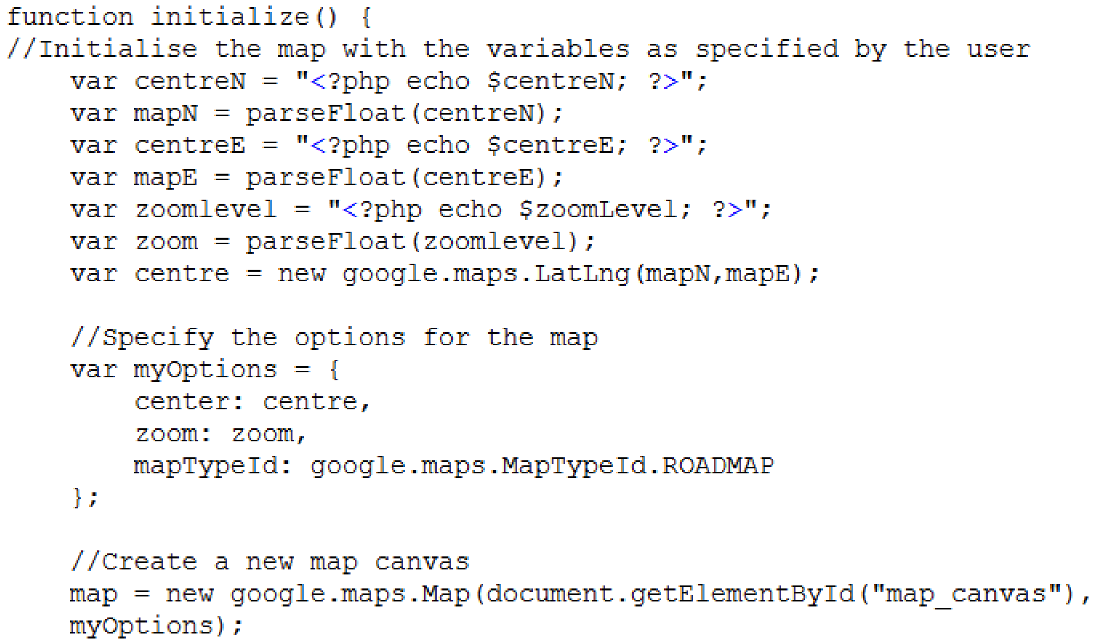
I have also implemented the option, in the form of a button, for the user to select whether or not the markers that denote where a tweet has been posted from are to be shown or hidden on the map. Hiding the markers allows the users to view the map with just the colours that are provided by the heat mapping function. The code used in this implementation (Figure 15) checks to see if the markers are currently being displayed when the button is clicked, if they are currently displayed then the properties of the map are set to display no markers '*marker[i].setMap(null)*'. The heat map overlay is still retained however, as the whole of the map does not have to be refreshed.

Other user driven options include being able to select the zoom level of the map and also the coordinates that the map is centred around, these make use of text boxes that allow the user to enter their desired information and then update the map accordingly. The text boxes and submit button form part of a PHP form for the webpage. The specified options are taken into account in the maps *initialise()* function (Figure 16). This means however that I have had to specify default values for when the map is loaded for the first time, before the user has had chance to enter their own values. These default values are passed by a PHP 'post' method to the *initialise() method* of the tweet map webpage, from the previous webpage.



```
,
function changemark(){
    if (markon==1){
        for (i in marker){
            marker[i].setMap(null);
        }
        markon=0;
    }else {
        for (i in marker){
            marker[i].setMap(map);
        }
        markon=1;
    }
}
```

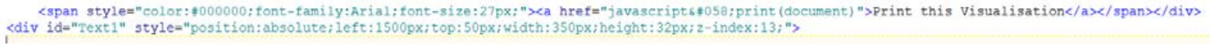
Figure 15 - Setting Markers On/Off



```
function initialize() {
//Initialise the map with the variables as specified by the user
    var centreN = "<?php echo $centreN; ?>";
    var mapN = parseFloat(centreN);
    var centreE = "<?php echo $centreE; ?>";
    var mapE = parseFloat(centreE);
    var zoomlevel = "<?php echo $zoomLevel; ?>";
    var zoom = parseFloat(zoomlevel);
    var centre = new google.maps.LatLng(mapN,mapE);

    //Specify the options for the map
    var myOptions = {
        center: centre,
        zoom: zoom,
        mapTypeId: google.maps.MapTypeId.ROADMAP
    };

    //Create a new map canvas
    map = new google.maps.Map(document.getElementById("map_canvas"),
    myOptions);
```

Figure 16 - Initialize Function (User Option Parameters Set Here)

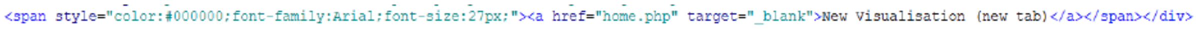
From the mapping visualisation the user is also able to print the map that they have created. This makes use of JavaScript's '*print(document)'* feature, which prints the entire web page as displayed on screen. This option is provided as a hyperlink for the user to click on. The HTML code provides details of where the button should be positioned on the screen. The user would also have the option of using their own web browsers print command to achieve the same effect; however I do feel that by providing a link for the user, it makes it a bit clearer.



```
    <span style="color:#000000;font-family:Arial;font-size:27px;"><a href="javascript&#058;print(document)">Print this Visualisation</a></span></div>
<div id="Text1" style="position:absolute;left:1500px;top:50px;width:350px;height:32px;z-index:13;">
```

Figure 17 - Print Visualisation

There is also a link displayed above that allows the user to easily create a new visualisation in a new browser tab. This could allow the user to create more than one visualisation at time and as such they would be able compare them easily against each other. Again this makes use of the ‘href’ attribute of HTML’s ‘a’ tag code, where you can specify a link to be opened within a new tab.



```
<span style="color:#000000;font-family:Arial;font-size:27px;"><a href="home.php" target="_blank">New Visualisation (new tab)</a></span></div>
```

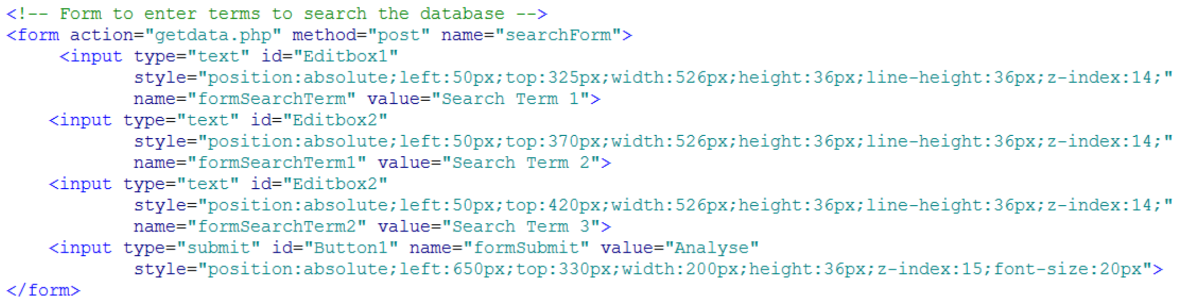
Figure 18 - Visualisation in a New Tab

Database Visualisation

The database visualisation provided as part of this toolkit allows users to query the database to see how many tweets are stored within the database for specific keywords. Currently this feature allows the user to enter three unique keywords. It can then return to the user a graph of the number of tweets that are contained within the database for the chosen keywords. There are three types of graph/charts available which are:

- Pie Chart
- Bar Chart
- Gauge Chart

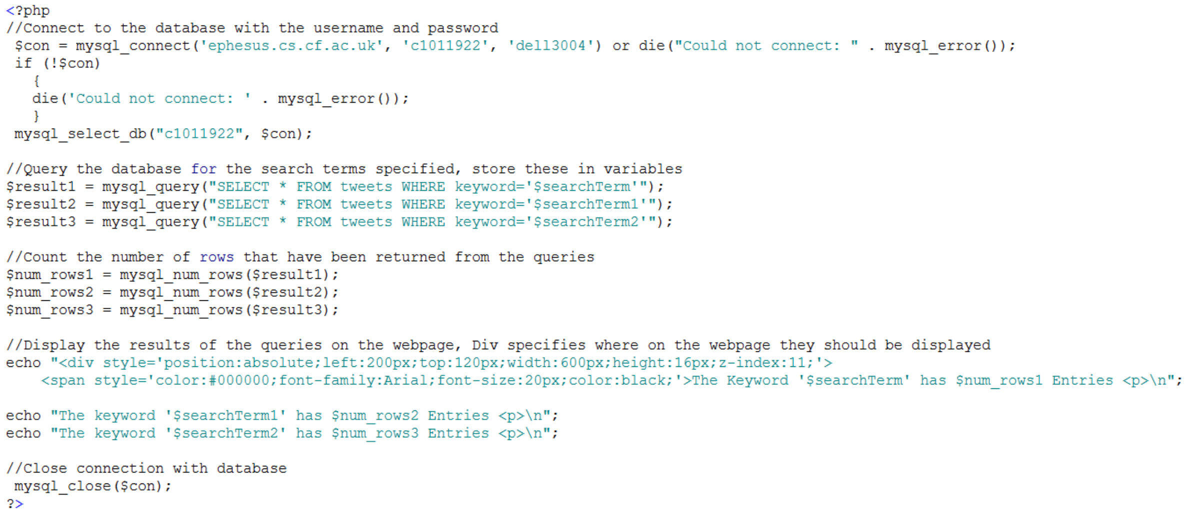
To enter the keywords the user wishes to search, a PHP form is used. This means that the values can be passed to the next webpage using the forms post method (Figure 19). The form's action tag specifies the name of the next file (web page) that is to be loaded, whilst the method specifies, in this instance, that the submitted data is 'posted' to the server. The PHP 'Get' method could also be used here, but, by using the Post method this means that the form data is hidden from the user and does not clutter up the URL in the address bar.



```
<!-- Form to enter terms to search the database -->
<form action="getdata.php" method="post" name="searchForm">
     <input type="text" id="Editbox1"
            style="position:absolute;left:50px;top:325px;width:526px;height:36px;line-height:36px;z-index:14;"
            name="formSearchTerm" value="Search Term 1">
    <input type="text" id="Editbox2"
            style="position:absolute;left:50px;top:370px;width:526px;height:36px;line-height:36px;z-index:14;"
            name="formSearchTerm1" value="Search Term 2">
    <input type="text" id="Editbox2"
            style="position:absolute;left:50px;top:420px;width:526px;height:36px;line-height:36px;z-index:14;"
            name="formSearchTerm2" value="Search Term 3">
    <input type="submit" id="Button1" name="formSubmit" value="Analyse"
            style="position:absolute;left:650px;top:330px;width:200px;height:36px;z-index:15;font-size:20px">
</form>
```

Figure 19 - Form for Entering Keywords

Once the user has entered their keywords and clicked on the analyse button the application then creates a connection to the database using the supplied login credentials. Once a connection is made then SQL queries are employed to search for all tweets *(Select *)* from the database for the inputted keyword, which is retrieved from the previous webpage's form using the post method. Having selected all the tweets for that keyword from the database, then the SQL function for counting the number of rows is then applied *(mysql_num_rows)*. This function counts up the number of rows of data that has been selected from the SQL query and the result of this is displayed as a string on the webpage alongside the relevant keyword term. This process is conducted for the three keywords that have been entered by the user. Once all the processing from the database has been conducted then the connection with the database is closed.



```
<?php
//Connect to the database with the username and password
 $con = mysql_connect('ephesus.cs.cf.ac.uk', 'c1011922', 'dell3004') or die("Could not connect: " . mysql_error());
 if (!$con)
   {
   die('Could not connect: ' . mysql_error());
   }
 mysql_select_db("c1011922", $con);

//Query the database for the search terms specified, store these in variables
$result1 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm'");
$result2 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm1'");
$result3 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm2'");

//Count the number of rows that have been returned from the queries
$num_rows1 = mysql_num_rows($result1);
$num_rows2 = mysql_num_rows($result2);
$num_rows3 = mysql_num_rows($result3);

//Display the results of the queries on the webpage, Div specifies where on the webpage they should be displayed
echo "<div style='position:absolute;left:200px;top:120px;width:600px;height:16px;z-index:11;'>
    <span style='color:#000000;font-family:Arial;font-size:20px;color:black;'>The Keyword '$searchTerm' has $num_rows1 Entries <p>\n";

echo "The keyword '$searchTerm1' has $num_rows2 Entries <p>\n";
echo "The keyword '$searchTerm2' has $num_rows3 Entries <p>\n";

//Close connection with database
 mysql_close($con);
?>
```

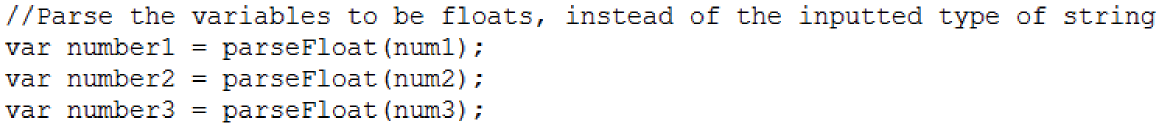
Figure 20 - Querying the Database

Three buttons are then displayed on the webpage giving options for the different types of graph that are available, these are:

- Pie Chart
- Bar Chart
- Gauge

Pie Chart

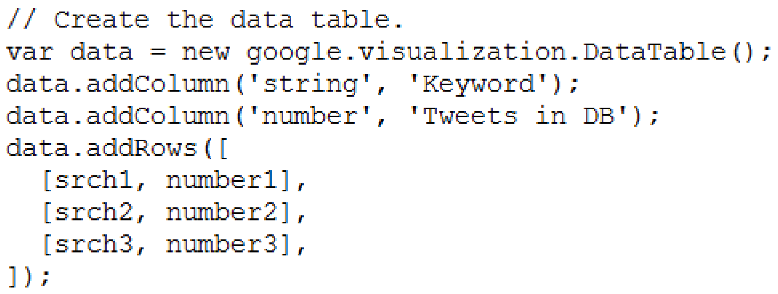
The pie chart uses Google's AJAX API to be able to create its diagram, it takes as its inputs, the number of rows that are in the database for each keyword that were calculated on the previous page. These are parsed from strings to be of type float as used by the graphs code (Figure 21).



```
//Parse the variables to be floats, instead of the inputted type of string
var number1 = parseFloat(num1);
var number2 = parseFloat(num2);
var number3 = parseFloat(num3);
```

Figure 21 - Parse Graph Variables

Once these have been parsed then the chart is set up using the values as their input parameters alongside the keywords that were originally specified by the user (Figure 22). The table is then created and displayed onscreen using a HTML 'Div' object so that the position of it can be controlled.



```
// Create the data table.
var data = new google.visualization.DataTable();
data.addColumn('string', 'Keyword');
data.addColumn('number', 'Tweets in DB');
data.addRows([
  [srch1, number1],
  [srch2, number2],
  [srch3, number3],
]);
```

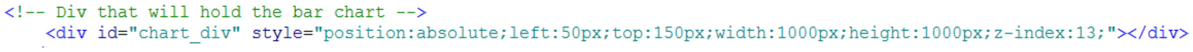
Figure 22 - Creating the Data Table

As with the mapping functionality of the application I also decided to implement the option of allowing the user to print the visualisation, this uses the same technology as mentioned above.

Bar Chart

The bar chart uses Google's AJAX API to be able to create its diagram, it takes as its inputs, the number of rows that are in the database for each keyword that were calculated on the previous page.

The creation of this visualisation works in the same way as the pie chart, once again the chart is displayed onscreen using HTML 'Div' tags.



```
<!-- Div that will hold the bar chart -->
    <div id="chart_div" style="position:absolute;left:50px;top:150px;width:1000px;height:1000px;z-index:13;"></div>
```

Figure 23 - HTML 'Div' Tags

The user is also able to print this visualisation using the same technique as the pie chart above.

Gauge Chart

The gauge chart uses Google's AJAX API to be able to create its diagram, it takes as its inputs, the number of rows that are in the database for each keyword that were calculated on the previous page.

The creation of this visualisation works in the same way as both the pie and bar charts, options are set within the charts AJAX code to specify the colours that are to be used as part of the gauges, a green, yellow and red scheme being used to show an increasing number of tweets (Figure 25).

```
      //Set chart options
      var options = {
        width: 800, height: 300,max:800,
  greenFrom:200, greenTo:700,
        redFrom: 700, redTo: 800,
        yellowFrom:600, yellowTo: 700,
        minorTicks: 5
      };
```

Figure 24 - Chart Options for the Gauge Chart

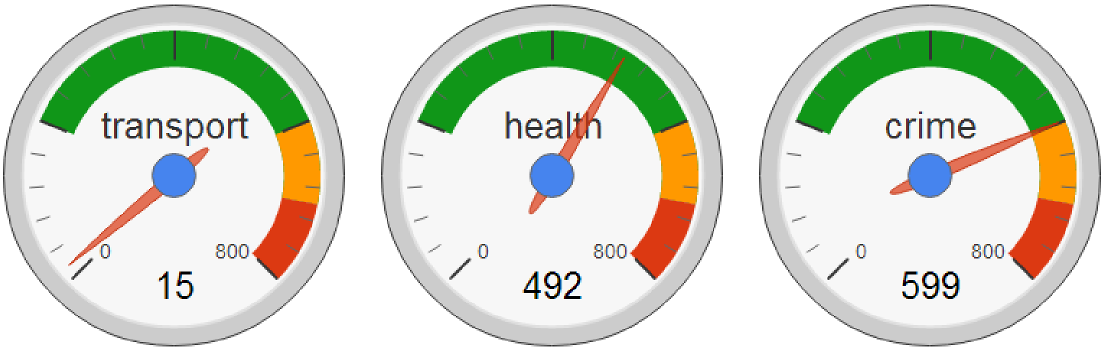


Figure 25 - Gauges Showing Results for Searches on Transport, Health and Crime

As with the mapping functionality of the application I also decided to implement the option of allowing the user to print the visualisation, this uses the same technology as mentioned above.

The three different visualisations are provided as a comprehensive and varied way of displaying data.

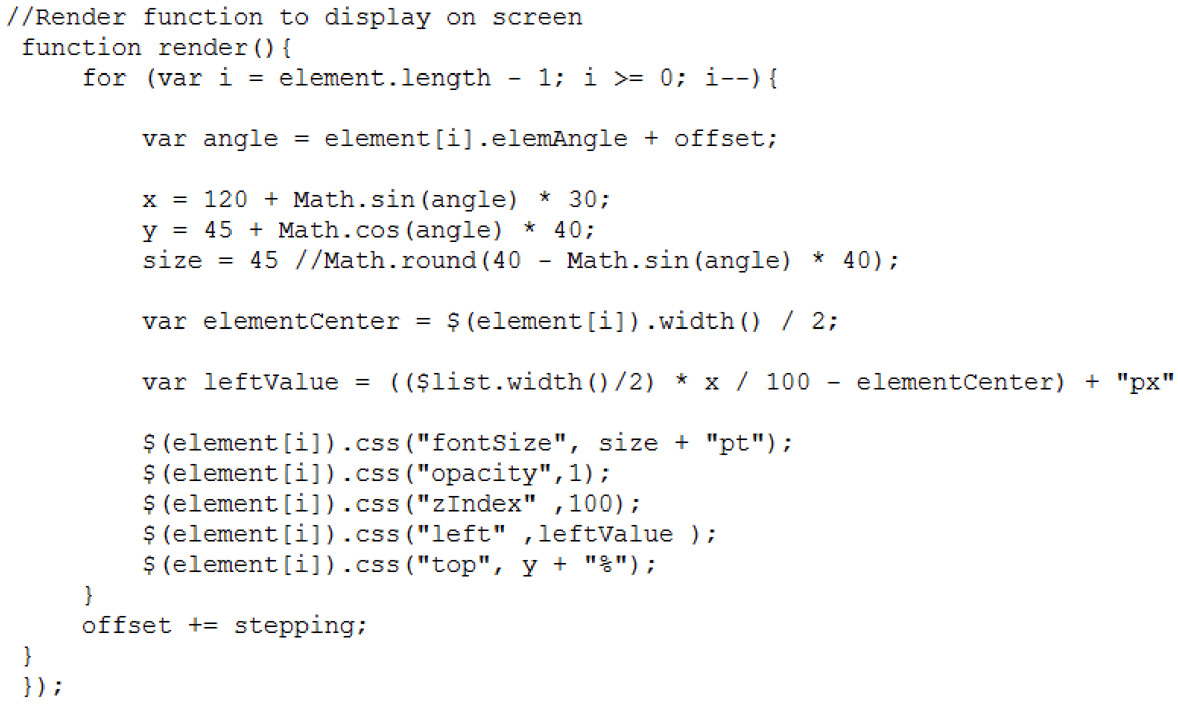
The challenge that was met when creating these visualisations was trying to ensure that all the data that they required were in the correct format for the visualisation APIs to use. This meant some considerable time spent in ensuring the data was always in the correct format.

Word Cloud

The word cloud or tag cloud is a way of visualising keywords in a 3D graphic. Tags are usually single words. In this implementation, the words are put into an array where they are then read out and displayed on screen. The visualisation allows the words to rotate in both directions and at differing speeds depending on how close to or far away from the centre of the visualisation the user's mouse is. All the key words are provided as clickable hyperlinks that if clicked, take the user to the hash tag for that keyword.

To start with, I implemented a way for the users of the service to be able to input their keywords, this was done using a PHP form on the webpage, this takes in all the keywords that are entered by the users and will pass them to the next webpage as variables when the user clicks on the 'Visualise' button.

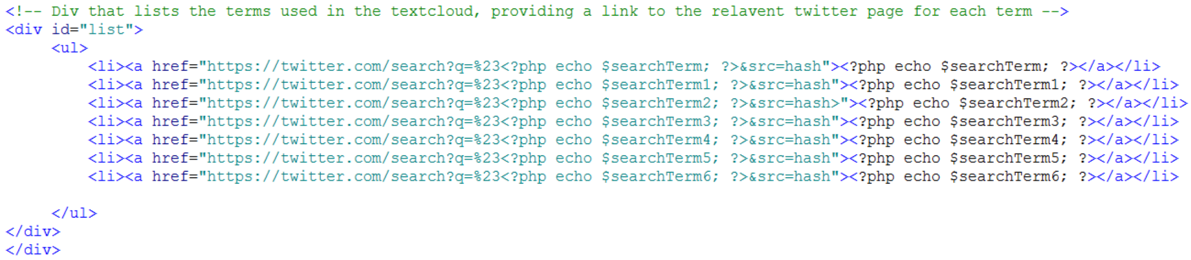
The word cloud makes use of another Google API for visualisation, displaying the words and determining how they are rotated. The render function of the API works out an angle to rotate around and then applies this to the array of elements (in this case the keywords). Other options are also set within this JavaScript code that specifies details such as text size (Figure 26).



```
//Render function to display on screen
 function render(){
     for (var i = element.length - 1; i >= 0; i--){

         var angle = element[i].elemAngle + offset;

         x = 120 + Math.sin(angle) * 30;
         y = 45 + Math.cos(angle) * 40;
         size = 45 //Math.round(40 - Math.sin(angle) * 40);

         var elementCenter = $(element[i]).width() / 2;

         var leftValue = (($list.width()/2) * x / 100 - elementCenter) + "px"

         $(element[i]).css("fontSize", size + "pt");
         $(element[i]).css("opacity",1);
         $(element[i]).css("zIndex" ,100);
         $(element[i]).css("left" ,leftValue );
         $(element[i]).css("top", y + "%");
     }
     offset += stepping;
 }
 });
```

Figure 26 - Render Function for Word Cloud

I have also implemented the word cloud such that it treats each keyword as a hyperlink using the ‘href’ attribute HTMLs ‘a’ tag. This allows you to click on any of the keywords that appear in the word cloud and the user would be taken to the twitter web pages where results for that hash tag would be displayed (Figure 27). The webpage that the user is directed to on Twitter is opened in a new browser tab so that they can easily navigate back to the application.



```
<!-- Div that lists the terms used in the textcloud, providing a link to the relavent twitter page for each term -->
<div id="list">
     <ul>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm; ?>&src=hash"><?php echo $searchTerm; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm1; ?>&src=hash"><?php echo $searchTerm1; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm2; ?>&src=hash>"><?php echo $searchTerm2; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm3; ?>&src=hash"><?php echo $searchTerm3; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm4; ?>&src=hash"><?php echo $searchTerm4; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm5; ?>&src=hash"><?php echo $searchTerm5; ?></a></li>
         <li><a href="https://twitter.com/search?q=%23<?php echo $searchTerm6; ?>&src=hash"><?php echo $searchTerm6; ?></a></li>

     </ul>
</div>
</div>
```

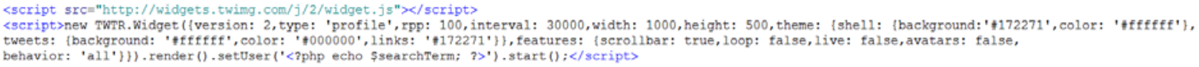
Figure 27 - Keywords as Hyperlinks

One of the main challenges in creating the word cloud was the displaying of it neatly on the interface; this was made difficult due to some of the rotational parameters that are set in the creation of the visualisation. Originally this API was supposed to be used as a standalone visualisation, so trying to integrate it into my website took some time to sort out.

Twitter Timeline

This allows the user of the application to easily visualise another user's twitter timeline, taking in their latest posts. It is a feature that twitter offer however, I think it is a key part of this project as it allows the users of my system to be able to perform the task of viewing a twitter timeline without leaving the application.

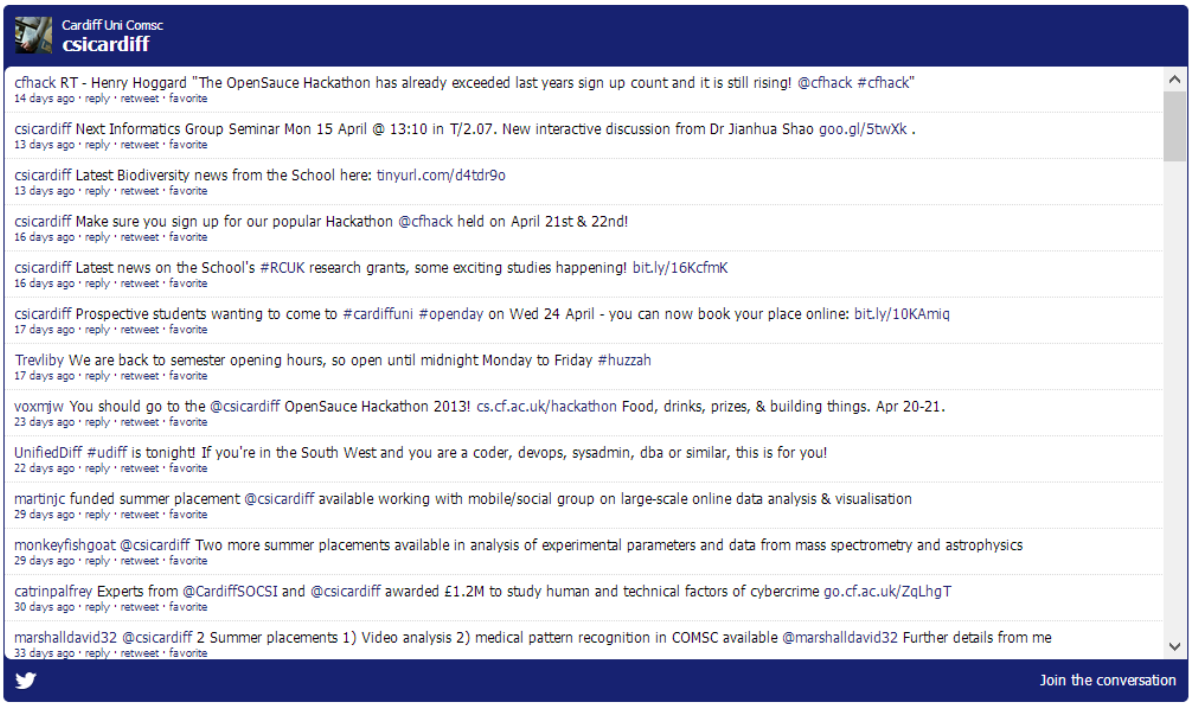
A PHP web form is used to allow the user to be able to input the account name of the user whose timeline they wish to view. The timeline as provided makes use of a twitter widget, which is officially supported. It takes in the variable that contains the account name, as provided by the user and is passed to this webpage by the PHP post method (Figure 28).



```
<script src="http://widgets.twimg.com/j/2/widget.js"></script>
<script>new TWTR.Widget({version: 2,type: 'profile',rpp: 100,interval: 30000,width: 1000,height: 500,theme: {shell: {background:'#172271',color: '#ffffff'},
tweets: {background: '#ffffff',color: '#000000',links: '#172271'}},features: {scrollbar: true,loop: false,live: false,avatars: false,
behavior: 'all'}}).render().setUser('<?php echo $searchTerm; ?>').start();</script>
```

Figure 28 - Twitter Timeline Widget

This script sets the values for the colour of the timeline and also how many tweets are to be displayed from the user. A 'follow on Twitter' button is also provided, but this takes the user to twitters website external of this application where they would have to sign in in order to be able to follow the account.



Cardiff Uni Comsc
csicardiff

cfhack RT - Henry Hoggard "The OpenSauce Hackathon has already exceeded last years sign up count and it is still rising! @cfhack #cfhack"
14 days ago · reply · retweet · favorite

csicardiff Next Informatics Group Seminar Mon 15 April @ 13:10 in T/2.07. New interactive discussion from Dr Jianhua Shao goo.gl/5twXk .
13 days ago · reply · retweet · favorite

csicardiff Latest Biodiversity news from the School here: tinyurl.com/d4tdr9o
13 days ago · reply · retweet · favorite

csicardiff Make sure you sign up for our popular Hackathon @cfhack held on April 21st & 22nd!
16 days ago · reply · retweet · favorite

csicardiff Latest news on the School's #RCUK research grants, some exciting studies happening! bit.ly/16KcfmK
16 days ago · reply · retweet · favorite

csicardiff Prospective students wanting to come to #cardiffuni #openday on Wed 24 April - you can now book your place online: bit.ly/10KAmiq
17 days ago · reply · retweet · favorite

Trevliby We are back to semester opening hours, so open until midnight Monday to Friday #huzzah
17 days ago · reply · retweet · favorite

voxmjw You should go to the @csicardiff OpenSauce Hackathon 2013! cs.cf.ac.uk/hackathon Food, drinks, prizes, & building things. Apr 20-21.
23 days ago · reply · retweet · favorite

UnifiedDiff #udiff is tonight! If you're in the South West and you are a coder, devops, sysadmin, dba or similar, this is for you!
22 days ago · reply · retweet · favorite

martinjc funded summer placement @csicardiff available working with mobile/social group on large-scale online data analysis & visualisation
29 days ago · reply · retweet · favorite

monkeyfishgoat @csicardiff Two more summer placements available in analysis of experimental parameters and data from mass spectrometry and astrophysics
29 days ago · reply · retweet · favorite

catrinpalfrey Experts from @CardiffSOCSI and @csicardiff awarded £1.2M to study human and technical factors of cybercrime go.cf.ac.uk/ZqLhgT
30 days ago · reply · retweet · favorite

marshalldavid32 @csicardiff 2 Summer placements 1) Video analysis 2) medical pattern recognition in COMSC available @marshalldavid32 Further details from me
33 days ago · reply · retweet · favorite

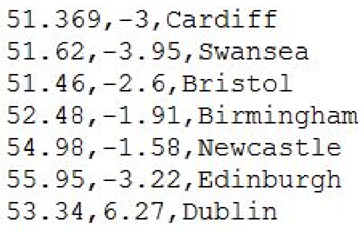
Figure 29 - Twitter Timeline Widget

Twitter are very strict on the guidelines that they release surrounding use of their brand and logos, as such this is why I have made this implementation using official twitter coded applications, this means that all logos and details of how the results of the timeline are displayed conform to Twitter's guidelines.

Upload Text File to Map

Following my implementation of the original map for the purpose of mapping the tweets that are retrieved from twitter and also those stored in the database, and following on from discussions with researchers, I started work on implementing the ability for the user of the system to be able to upload their own data to the map service. This was suggested as a feature that researchers would like as it would also allow them to create similar types of visualisation to the heat map, as those that they would be creating from the data sets retrieved from social media services to data sets that had come from more official sources. In my implementation of this, I was able to manipulate the Google Maps API so that it would take in a '.txt' text file and map the contents. However, the layout of the text file had to be quite specific (Figure 30). The format had to have the latitude first, followed by the longitudinal coordinates of where the marker was to be placed on the map followed then by the information that would be contained within the information window. From the figure below the first example would show the text "Cardiff" being displayed in the information window of a marker that was placed at 51.369^{o}N and -3^{o} West.

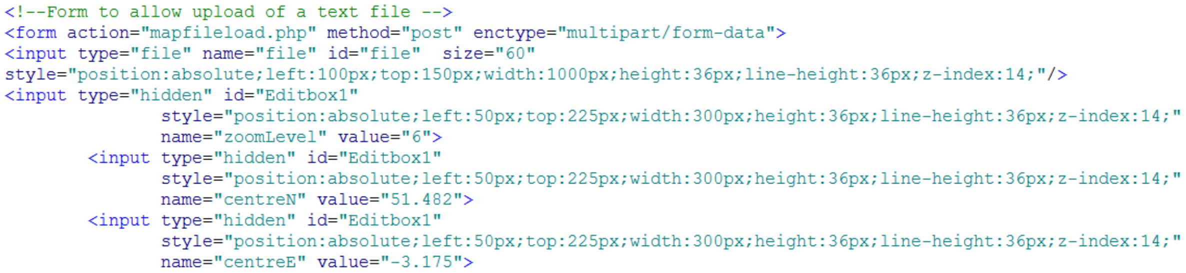
There would be a need for researchers to format their data appropriately but once done they would be able to use the full functionality of the mapping system.



```
51.369,-3,Cardiff
51.62,-3.95,Swansea
51.46,-2.6,Bristol
52.48,-1.91,Birmingham
54.98,-1.58,Newcastle
55.95,-3.22,Edinburgh
53.34,6.27,Dublin
```

Figure 30 - Format of Text Document to be Uploaded

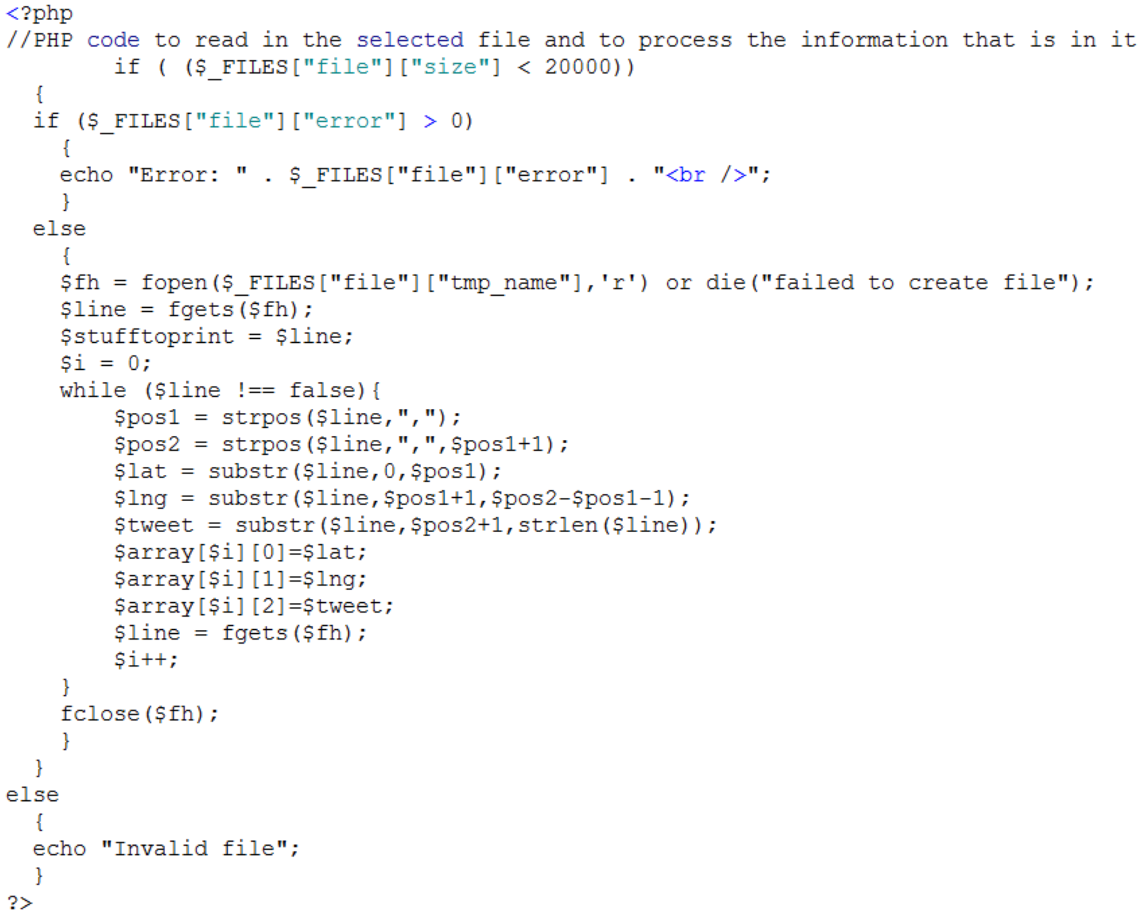
The upload file webpage made use of a PHP form with the type, 'multipart/form-data'. The form also specifies that the input type is supposed to be a file. This creates a button on screen that allows the user to be able select which file they wish to upload. By default the machines file explorer is displayed to aid the user in selecting their file.



```
<!--Form to allow upload of a text file -->
<form action="mapfileload.php" method="post" enctype="multipart/form-data">
<input type="file" name="file" id="file"  size="60"
style="position:absolute;left:100px;top:150px;width:1000px;height:36px;line-height:36px;z-index:14;"/>
<input type="hidden" id="Editbox1"
                     style="position:absolute;left:50px;top:225px;width:300px;height:36px;line-height:36px;z-index:14;"
                     name="zoomLevel" value="6">
           <input type="hidden" id="Editbox1"
                     style="position:absolute;left:50px;top:225px;width:300px;height:36px;line-height:36px;z-index:14;"
                     name="centreN" value="51.482">
           <input type="hidden" id="Editbox1"
                     style="position:absolute;left:50px;top:225px;width:300px;height:36px;line-height:36px;z-index:14;"
                     name="centreE" value="-3.175">
```

Figure 31 - Uploading a File to be Mapped

This uploaded file is then passed by PHP to the Mapping API. On this webpage the code first reads in the file, this once again is achieved with the use of PHP (Figure 32). The ‘fopen’ function is used to open the file and then the contents of the file are printed out line by line. During the print out of each line the contents of the file are passed to the relevant variables that are needed to map the information that has been uploaded, this includes the latitude, longitude and text (tweet). The ‘fclose’ function is finally called when the whole file has been read. If an invalid type of file is uploaded then a message stating this is displayed to the users. PHPs ‘echo’ function allows the message “Invalid File” to be displayed to the user of the application.



```
<?php
//PHP code to read in the selected file and to process the information that is in it
    if ( ($_FILES["file"]["size"] < 20000))
 {
 if ($_FILES["file"]["error"] > 0)
  {
  echo "Error: " . $_FILES["file"]["error"] . "<br />";
  }
 else
  {
  $fh = fopen($_FILES["file"]["tmp_name"],'r') or die("failed to create file");
  $line = fgets($fh);
  $stufftoprint = $line;
  $i = 0;
  while ($line !== false){
    $pos1 = strpos($line,",");
    $pos2 = strpos($line,",",$pos1+1);
    $lat = substr($line,0,$pos1);
    $lng = substr($line,$pos1+1,$pos2-$pos1-1);
    $tweet = substr($line,$pos2+1,strlen($line));
    $array[$i][0]=$lat;
    $array[$i][1]=$lng;
    $array[$i][2]=$tweet;
    $line = fgets($fh);
    $i++;
  }
  fclose($fh);
  }
 }
else
 {
 echo "Invalid file";
 }
?>
```

Figure 32 - Reading in the Contents of the Uploaded File

Once the contents of the file have been read in and the variable values are stored correctly in the array, then the rest of the mapping process is put to work. This is much the same as for the mapping of data retrieved from social media services. I decided to implement this mapping functionality (for the users own data) in the same way so that the user could still make use of the heat map function and also the choice of displaying or not displaying the markers.

I have also made sure that the options for customising the map are available to the user. This means they can create like for like visualisations whether they are using retrieved data from Twitter or even their own. By using the dynamic Google Maps API I have also been able to take full advantage of the features of a Google map that most users would typically expect, these include:

- Zooming in and out of the map using the scroll wheel on a mouse
- Changing the orientation of the map by dragging it
- Full functionality for touchscreen devices
- The ability for the user to use Google's Street View Service if they wish to view images of a specific area

The inclusion of Google's street view could be useful for researchers as it could give greater insight into why an area could be for example more prone to crime than another area.

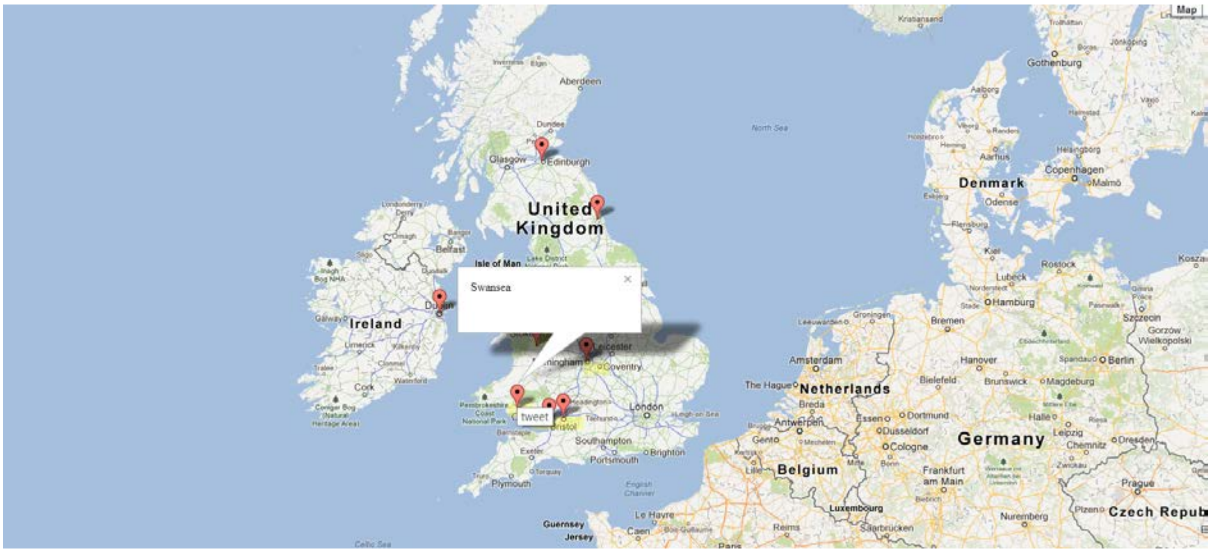


Figure 33 - Map with Uploaded File Information, Visible is an Information Box

One of the main challenges I faced implementing this feature was trying to find a format of text document that could be uploaded so that the data contained within it was useful. This meant that I had to be aware of the way PHP would process the file data, and also what format the Google Maps API would need to be able to display it.

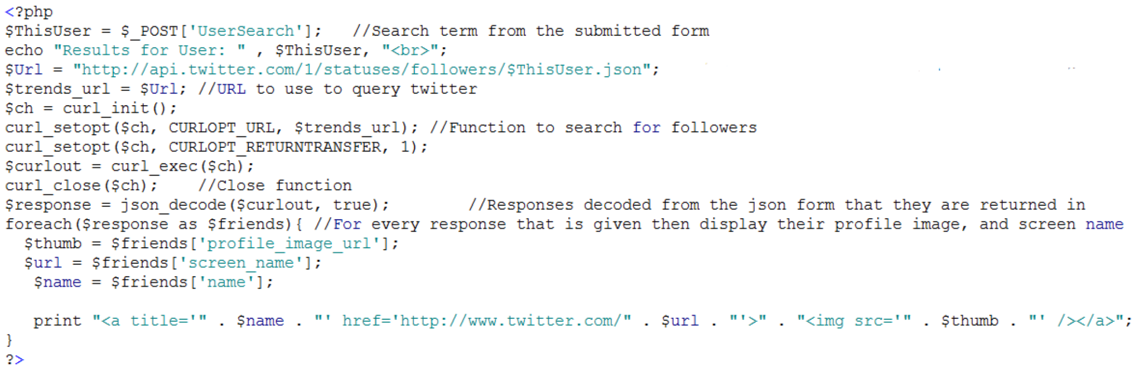
User Links

The user links visualisation of the application is intended to allow the user of this system to easily visualise which other people a specific person on twitter is connected with, either through them following another member of the Twitter community or from them being followed themselves.

I have implemented this once again using a PHP form that allows the user of the application to specify a twitter username for the person they wish to view the links to and from. The text that is entered in this form is then passed to the next webpage.

The PHP code on this webpage then takes the twitter user name that has been entered and queries Twitter on the URL 'http://api.twitter.com/1/statuses/followers/$ThisUser.json' where '$ThisUser' is the PHP variable given. This queries Twitter, looking for the links that user has with other members of the Twitter community. These results are returned in the JSON format. The JSON results are then decoded by PHP. Processing of the JSON data is then carried out for every 'friend' that the user has. This allows the PHP code to display a profile image for that user (as specified on twitter) and also the username of the 'friend'. These are then printed to the webpage in a tabular form and each are created as hyperlinks so that the user of the application can click on any of the profile images shown to them to be taken to the profile page for each of the different users.

Where a 'friend' has not specified a profile image then the default twitter picture of an "egg" is used. The PHP 'print' statement is used to display all the results on screen. The print statement is used here as opposed to the echo statement that I have used mainly throughout the rest of the application as it can be used in expressions such as this code statement as it provides a return value.



```
<?php
$ThisUser = $_POST['UserSearch'];   //Search term from the submitted form
echo "Results for User: " , $ThisUser, "<br>";
$Url = "http://api.twitter.com/1/statuses/followers/$ThisUser.json";
$trends_url = $Url; //URL to use to query twitter
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $trends_url); //Function to search for followers
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
$curlout = curl_exec($ch);
curl_close($ch);    //Close function
$response = json_decode($curlout, true);        //Responses decoded from the json form that they are returned in
foreach($response as $friends){ //For every response that is given then display their profile image, and screen name
  $thumb = $friends['profile_image_url'];
  $url = $friends['screen_name'];
   $name = $friends['name'];

   print "<a title='" . $name . "' href='http://www.twitter.com/" . $url . "'>" . "<img src='" . $thumb . "' /></a>";
}
?>
```

Figure 34 - PHP Code for Retrieving Twitter Followers

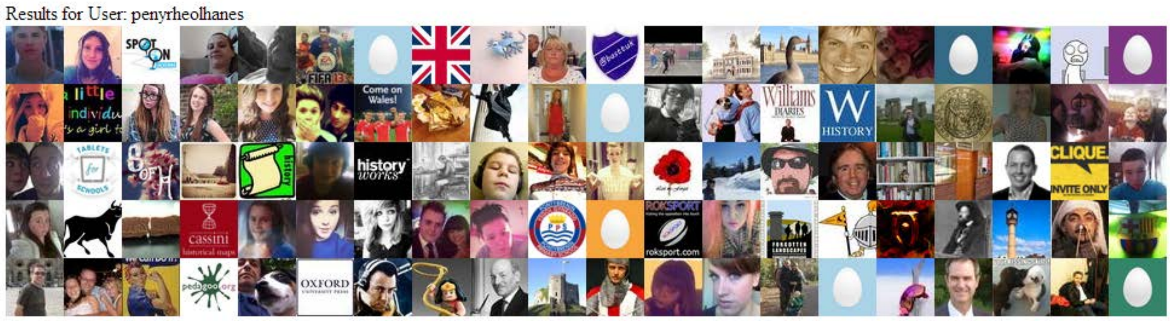

Results for User: penyrheolhanes

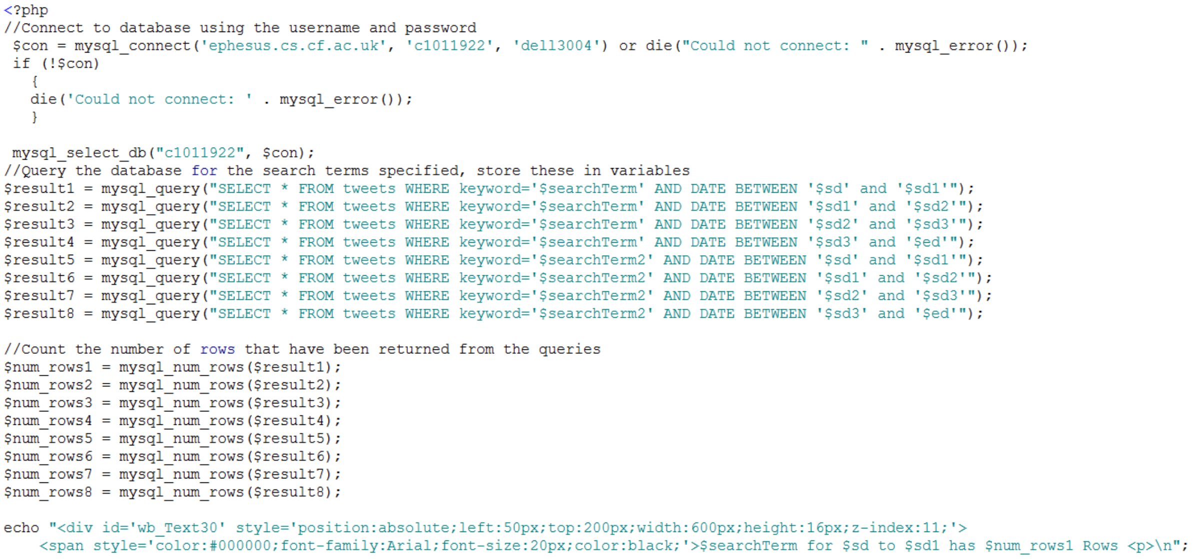
Figure 35 - Displaying the Results

Tweets vs Time

The Tweet vs Time feature that I have implemented is intended to allow the user to search the number of tweets in the database that appear between certain timeframes. These keywords for the tweets can be specified by the user. The timeframes in which the tweets can be searched are also customisable by the user.

I have implemented this feature by providing the user with a form that they can specify the two keyword search terms they wish to view and also the time frames with a five range selection. The form takes these in as variables and will process them onwards to the next webpage when the submit button is pressed.

The date and time however has to be entered in the format ‘yyyy/mm/dd hh:mm:ss’ but this is clearly displayed to the user onscreen so that there would be no confusion.



```
<?php
//Connect to database using the username and password
 $con = mysql_connect('ephesus.cs.cf.ac.uk', 'c1011922', 'dell3004') or die("Could not connect: " . mysql_error());
 if (!$con)
   {
   die('Could not connect: ' . mysql_error());
   }

 mysql_select_db("c1011922", $con);
//Query the database for the search terms specified, store these in variables
$result1 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm' AND DATE BETWEEN '$sd' and '$sd1'");
$result2 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm' AND DATE BETWEEN '$sd1' and '$sd2'");
$result3 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm' AND DATE BETWEEN '$sd2' and '$sd3'");
$result4 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm' AND DATE BETWEEN '$sd3' and '$ed'");
$result5 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm2' AND DATE BETWEEN '$sd' and '$sd1'");
$result6 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm2' AND DATE BETWEEN '$sd1' and '$sd2'");
$result7 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm2' AND DATE BETWEEN '$sd2' and '$sd3'");
$result8 = mysql_query("SELECT * FROM tweets WHERE keyword='$searchTerm2' AND DATE BETWEEN '$sd3' and '$ed'");

//Count the number of rows that have been returned from the queries
$num_rows1 = mysql_num_rows($result1);
$num_rows2 = mysql_num_rows($result2);
$num_rows3 = mysql_num_rows($result3);
$num_rows4 = mysql_num_rows($result4);
$num_rows5 = mysql_num_rows($result5);
$num_rows6 = mysql_num_rows($result6);
$num_rows7 = mysql_num_rows($result7);
$num_rows8 = mysql_num_rows($result8);

echo "<div id='wb_Text30' style='position:absolute;left:50px;top:200px;width:600px;height:16px;z-index:11;'>
    <span style='color:#000000;font-family:Arial;font-size:20px;color:black;'>$searchTerm for $sd to $sd1 has $num_rows1 Rows <p>\n";
```

Figure 36 - Querying the Database

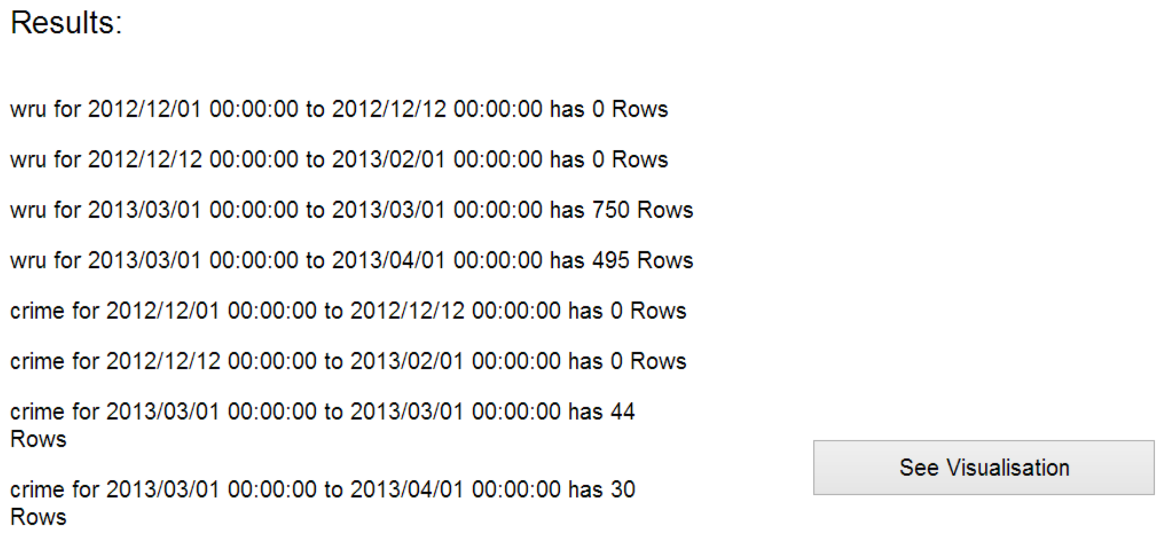
The PHP code that is then used to process this (Figure 36) establishes a connection with the SQL Database by logging in with the required credentials. The database is then queried using the SQL language so that for each keyword the tweets that are between the dates specified are returned.

In the SQL language the ‘*Date Between ** and **’* will search for all tweets where the keyword matches the one entered between the date ranges, for example this could be the time frame of a month. An example of how the commands are processed is, Search Term 1 will be queried between the first date entered and the second date entered for all tweets that contain the search term as their keyword. This process is then repeated for all the date ranges on the first keyword and then for all date ranges on the second keyword.

The PHP code then processes how many rows of data have been returned for the supplied information. This is displayed on the web page in a simple text layout for the user (Figure 37).

I have made use of PHPs echo statements to display this information to the user.

As all the data that is calculated from the above process is held within PHP variables, these can then be passed directly into a hidden form. The hidden form is created by using the forms input type option and setting it to be hidden. This results in a button being displayed on screen that allows the user to click it to be taken to the next web page where the visualisation will be shown.



Results:

wru for 2012/12/01 00:00:00 to 2012/12/12 00:00:00 has 0 Rows

wru for 2012/12/12 00:00:00 to 2013/02/01 00:00:00 has 0 Rows

wru for 2013/03/01 00:00:00 to 2013/03/01 00:00:00 has 750 Rows

wru for 2013/03/01 00:00:00 to 2013/04/01 00:00:00 has 495 Rows

crime for 2012/12/01 00:00:00 to 2012/12/12 00:00:00 has 0 Rows

crime for 2012/12/12 00:00:00 to 2013/02/01 00:00:00 has 0 Rows

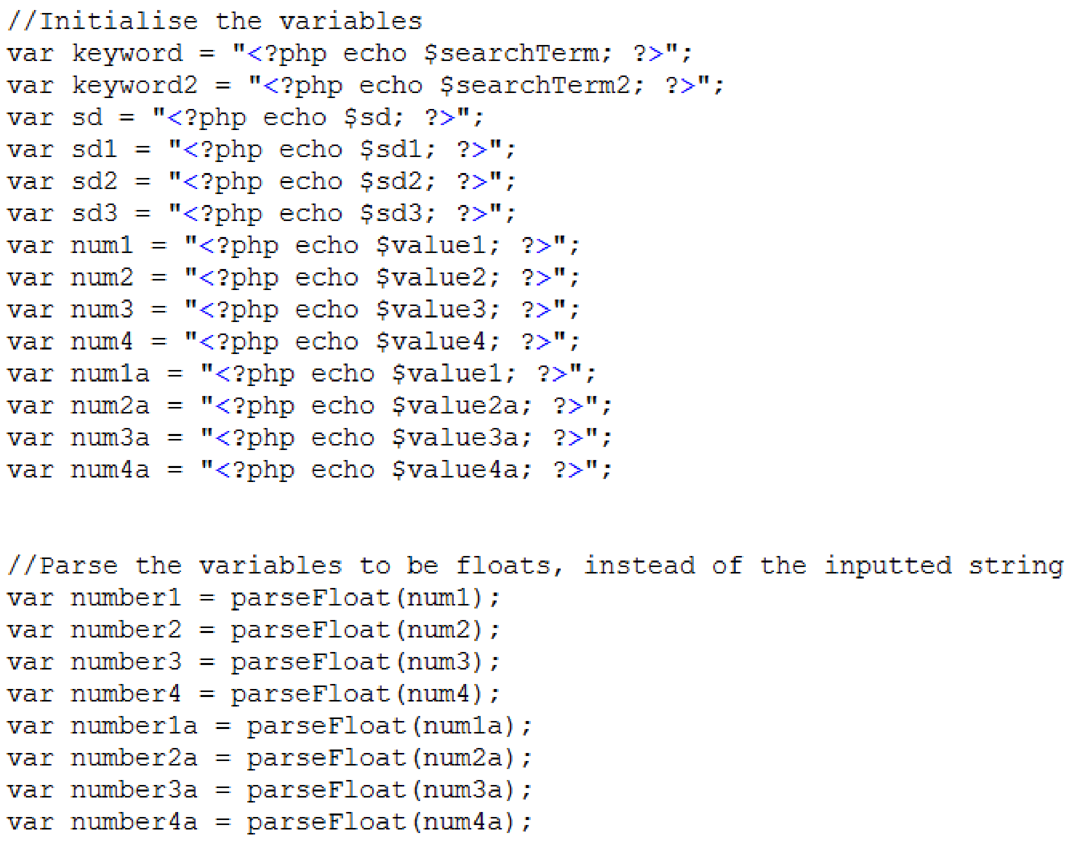
crime for 2013/03/01 00:00:00 to 2013/03/01 00:00:00 has 44 Rows

crime for 2013/03/01 00:00:00 to 2013/04/01 00:00:00 has 30 Rows

Figure 37 - Results Displayed for User

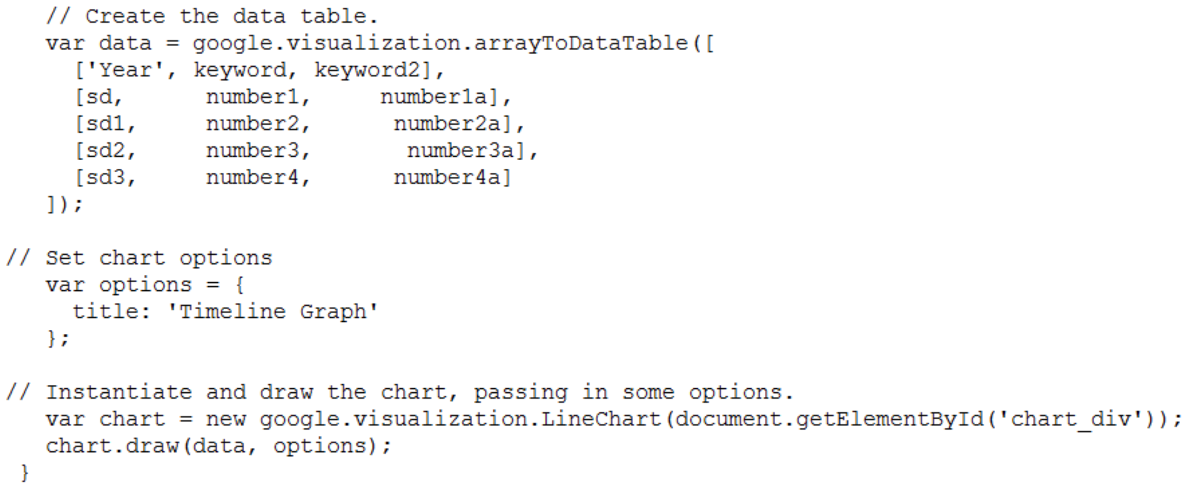
The visualisation that is displayed is a line chart that makes use of Google's AJAX API to be able to create its diagram, it takes as its inputs the number of rows that are in the database for each keyword that were calculated on the previous page, these are passed to it through the use of PHP variables. The variables that are present are in the string format, and as such these need to be parsed from strings to be of type float as used by the graphs data table code (Figure 38).

Once these have been parsed then the chart is set up using the values as their input parameters (Figure 39). The table is then created; it is displayed onscreen using a HTML 'Div' object so that the position of it can be controlled. As with the mapping functionality of the application and also the database visualisations I have also decided to implement the option of allowing the user to print the resulting graph, this uses the same technology as mentioned above.



```
//Initialise the variables
var keyword = "<?php echo $searchTerm; ?>";
var keyword2 = "<?php echo $searchTerm2; ?>";
var sd = "<?php echo $sd; ?>";
var sd1 = "<?php echo $sd1; ?>";
var sd2 = "<?php echo $sd2; ?>";
var sd3 = "<?php echo $sd3; ?>";
var num1 = "<?php echo $value1; ?>";
var num2 = "<?php echo $value2; ?>";
var num3 = "<?php echo $value3; ?>";
var num4 = "<?php echo $value4; ?>";
var num1a = "<?php echo $value1; ?>";
var num2a = "<?php echo $value2a; ?>";
var num3a = "<?php echo $value3a; ?>";
var num4a = "<?php echo $value4a; ?>";


//Parse the variables to be floats, instead of the inputted string
var number1 = parseFloat(num1);
var number2 = parseFloat(num2);
var number3 = parseFloat(num3);
var number4 = parseFloat(num4);
var number1a = parseFloat(num1a);
var number2a = parseFloat(num2a);
var number3a = parseFloat(num3a);
var number4a = parseFloat(num4a);
```

Figure 38 - PHP String Variables Parsed as Floats



```
   // Create the data table.
   var data = google.visualization.arrayToDataTable([
     ['Year', keyword, keyword2],
     [sd,      number1,     number1a],
     [sd1,     number2,      number2a],
     [sd2,     number3,       number3a],
     [sd3,     number4,      number4a]
   ]);

// Set chart options
   var options = {
     title: 'Timeline Graph'
   };

// Instantiate and draw the chart, passing in some options.
   var chart = new google.visualization.LineChart(document.getElementById('chart_div'));
   chart.draw(data, options);
 }
```

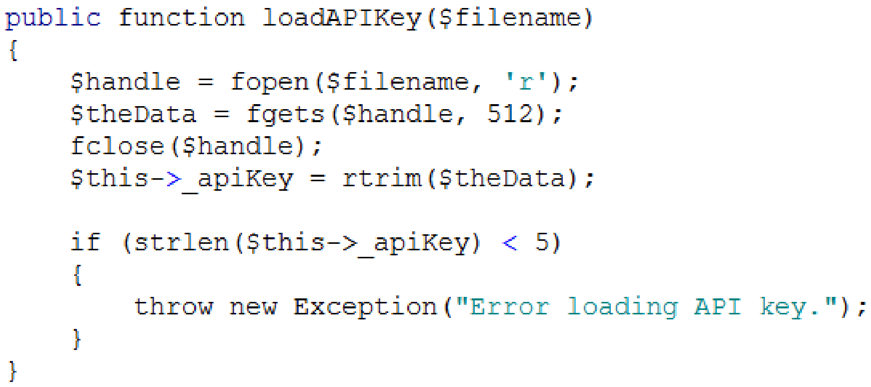
Figure 39 - Creation of Data Table

Sentiment of Tweets

As part of my implementation I decided to include a way of providing the user with some information regarding the sentiment of the tweets currently on Twitter from the keyword that the user has entered.

Sentiment is a term that refers to the application of natural language processing. This is a way of identifying subjective information that is contained within source materials, in this case the tweets that are determined from the keyword search. Sentiment analysis is generally used to determine the attitude of the writer of a document or in the case of this application, the attitudes of the creators of Twitter posts. In my system the sentiment analysis is used to provide an overall view as to whether or not the tweets on a given keyword are of a positive or negative manner.

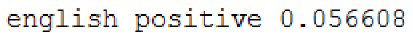
I decided to use the Alchemy API [3] for my sentiment analysis as this provides a cloud-based text analysis infrastructure. Using the Alchemy API requires you to register for one of their API Keys, which I was able to do freely and this is one of the main reasons why I could use them in this implementation of my project. Whilst the API is written in the PHP language, the key that is used has to be read in from a text file (Figure 40).



```
public function loadAPIKey($filename)
{
    $handle = fopen($filename, 'r');
    $theData = fgets($handle, 512);
    fclose($handle);
    $this->_apiKey = rtrim($theData);

    if (strlen($this->_apiKey) < 5)
    {
        throw new Exception("Error loading API key.");
    }
}
```

Figure 40 - Loading in the API Key

The API works by taking the URL Address for Twitter taking in a PHP variable that contains the keyword that the user has searched for. The result of the sentiment analysis is then displayed on screen to the user (on the applications dashboard). It is provided by telling the user whether the sentiment is positive or negative. A number giving the strength of this positivity or negativity is displayed. The format of the outputted result is shown in the figure below. Here we can see that the language that was detected was English, and it was a positive sentiment result.



```
english positive 0.056608
```

Figure 41 - Output of Sentiment Analysis

The sentiment analysis that I have implemented within this web application only takes account of the newly downloaded data from twitter, it will not perform sentiment analysis on the data that is already held within the database. That would need to be implemented with the use of some different APIs and further querying of the database would need to be done, this however in my opinion would not add much value to the system.

This feature of the application was fairly difficult to integrate and support for the API was very limited at the time.

Results and Evaluation

Development of my application has been a success in my opinion. From the description of the project and the aims and objectives I set out in the initial plan, I feel that I have achieved everything that I had stated and that was necessary for the system to provide the functionality needed.

In my evaluation of the results of this web application I will look at a few key areas which are:

- Unique Features to my System
- Evaluating the developed application against the original aims of the project
- Tests of Application Critical Elements
- Evaluating the application in terms of the scenarios I had set out at the interim report
- Evaluation of overall ease of use and functionality of the system

Unique Features

The application that I have developed has a number of unique features which are not present in many of the other similar visualisation services that I had researched and reported on in my interim report. The system provides a wide range of functionality which has the advantage of all being accessible in one place.

Some of the unique features of the system include:

- The ability to upload the users own data which could be official statistics to the mapping service, this means that the users of the system can easily compare the results obtained from the social media data with those that are officially provided. The ability to compare with official data can also help researchers in their justifications and analysis of the information they have retrieved from social media services.
- Heat map overlay which created a visual representation into the density of the tweets that are posted. Most other systems that I viewed only had the ability to plot the locational information of tweets on the map, but did not provide any further analysis as to the amount of tweets or the density of these tweets to their users. The heat map visualises the geographical data attained from the tweets received from twitter and also those that are stored within the database. It provides a colour coded overlay to the map indicating the areas within a country or region with the highest density of tweets relating to the search query undertaken. These high density areas signify the regions in the country which are tweeting about specific topics more than others.

- Displaying the tweet data when you hover over the location of the tweet on the map. Many of the other services I viewed such as the “Twittermap” web service only provided the ability to display the location of the tweets with a marker, but it did not contain the information regarding to the tweets content.
- The displaying of thumbnails of the various visualisations. The system is unique in providing these images, however this is mainly due to the fact that there are numerous visualisation provided in one place, which in itself is a salient feature to other services that I viewed which tended to focus mainly on a single visualisation.

Evaluation against Aims and Objectives

In this evaluation I will look at how closely the application that I have developed is to the aims and objectives of the project that I had set out in the initial project plan. To perform this evaluation I will break down my original aims and objectives and provide appropriate comments to how the results of development match what I had originally set out. (The original aims will be shown in Italics)

Aims and Objectives for the Final Report

- *Develop a system to aid Social Science researchers in choosing an appropriate visualisation technique to use for their selected dataset:*
    - *Provide thumbnail views for the user of what different types of visualisations available could look like for their data set*

My implementation provides the users with thumbnail images which are displayed on the applications dashboard. These give a guide to the users of what the various different types of visualisation the system could provide. However the thumbnails that are shown do not provide images that would match how the visualisations would look for the users own data set. They are instead static images that give a basic overview.

- *Thumbnails will be roughly generated to give an impression of what the visualisation may look like*

Thumbnails were generated by using the application on a set piece of test data, they give a true representation as to how the visualisations would appear however they are not generated at the runtime using the users own data set. I took the decision to implement static images for the thumbnails as in generating the visualisation for every user meant that there would be a significant time delay present in loading the dashboard page of the website.

- *Allow users to select, by clicking on a thumbnail, a visualisation technique which will then be applied fully to their data set for their research purposes*

With the thumbnails that I have implemented, the user is able to click on their chosen one which will then direct them to the next appropriate webpage so that the chosen visualisation can then be applied to their own data set. Allowing the user to select their visualisation technique directly by clicking on one of the images has provided the application with a very simple clear to understand interface that will make it easy for people with a limited background in technology to use.

- *Look at how using different API's (provided by Twitter, Google etc.) can benefit the retrieval and basic visualisation of information on the web*

My implementation to the solution of this project has involved me using APIs from a range of different companies. These have enabled me to achieve the full potential of their applications and services they have offered. In my implementation I have used the Twitter API to successfully query twitter on a given keyword to return a set of tweets that use that keyword.
I have also utilised Google's Map API to successfully implement my plotting of the tweets received onto a map according to the geo-locational data that is supplied with them. This also gave me a good basis to start implementing my heat map from. The APIs provided by Google have also been very useful in creating my graphs and charts that are formed as part of the visualisation packages in provide to the user.

- *Allow the developed system to be adaptive in learning what visualisation techniques are selected most often by Social Science Researchers and on which types of data;*
  - *The system could then provide a list of recommendations based on the most frequent selections and display these to help other researchers who may be using similar types of data sets*

My developed system fulfils this objective in a sense, however, as the research and development phases of the project were undertaken this has altered slightly. All users of the system would now be using similar data sets as all data sets would be produced from the results obtained from twitter. With this in mind, I have then selected appropriate visualisations that the user can create using such a data set. Within the application there is the recommendation of using the upload file feature if the user wishes to upload their own data set. The inclusion of a recommendation system may not provide as useful as first thought as the scope of the system is now to reach a wider audience.

- *Test the developed application with Social Science Researchers to gauge feedback into how the system works and if the types of visualisations are appropriate for the sorts of data and types of research that they are carrying out;*
  - *Look at conducting a study into how they would carry out their research before the tool that I hope to develop would be available, and then compare this to how they select visualisation techniques once this developed software has been made available to them*

Having worked alongside some social science students I have been able to gauge at each point in development of my system how useful they would see the visualisation that I was intending to provide would be to them. Through informal interviews with these students I have gained an insight into some of the tools and techniques they currently use to aid in

their research and also the sorts of visualisations they have required from different sets of data.
This has helped me to make informed decisions as to the types of visualisations I have been able to provide.

Further to the aims and objectives set out in the initial plan, I also set out some main features to focus on in my interim report, these were:

I have decided that I will provide a range of visualisation techniques to the users of the system and that I will be concentrating mainly on:

- *Mapping tweets and providing a form of heat map to show the density of tweets in certain locations*

The application provides the user with an easy means to map the tweets that they download from twitter and also those that are stored within the database. A successfully implemented heat map also means that the user can clearly see the differences in the densities of tweets in certain locations.

- *User links tool that will provide an overview of the links between users depending on who they are following and who follows them, looking at links between people who are tweeting about similar things*

In my implementation I have created a visualisation tool that allows the users to see which other members of the twitter community are following a specified person. This will aid researchers in building up connections between similarly minded groups of people and can provide as a starting point for more in-depth analysis into how the links between specific users of the twitter community interact. The application successfully retrieves the relevant information from twitter and displays it in a clear design on the web page.

- Visualising the trends of tweets over time, looking at the differences in amounts of tweets at given moments

The application allows the users to easily query the database looking at the trends in the numbers of tweets that are stored over time, alongside this the twitter timeline tool also allows a user of my system to easily see the frequency at which specific members of the twitter community are active in posting tweets.

Tests of Application Critical Elements

Evaluation of key methods will allow me to ensure that the main functionality of the system is working. The specifics that I have chosen to test are key parts of the system and to ensure that these are working correctly will mean that the rest of the system is able to function effectively, I have split this into three separate tests.

Test 1 - Google API's

To check the API used as part of the Google Maps service would work alongside the Twitter API that I was using and that they would both be able to function correctly on their own.

Ways to do this:

- A manual check with a set of different locations that are returned from searches to make sure that the locations that are retrieved with the tweets are mapped correctly
- A manual check that the area of the map that is covered by the heat map would be correct and that the scaling distance for the heat map area is altered when the zoom of the map is changed

The use of the mapping function within the application is the key visualisation service that my system will offer and is unique in its way of presenting a heat map. I therefore need to ensure that it is operating correctly to fulfil the specification of the application.

Test 2 - Use of the Database

To check that the database saves and stores the correct information as received from the Twitter retrieval part of the system. I would need to ensure that data is stored in the correct fields so that it is easily available to be queried by the other visualisation techniques I would be using.

Ways to do this:

- Enter validation rules into the setup of the database which would only allow certain types of data to be entered into different fields. It can also specify that every field of a record must contain some form of data
- A manual check to ensure that the data that is being stored is indeed being stored in the correct field, for example, the Longitude that is saved is being saved in the longitude field of the database
- Check that there is enough storage space for the desired amount of data. In the first instance this would only need to be sufficient to be able to carry out testing during development

The data within the database is vital to all parts of the application and therefore maintaining the data and ensuring that it is stored properly is critical.

Test 3 - Interfaces

To ensure that the interface of the application displays well on multiple web platforms and to ensure that any images that are displayed and also input text fields appear in the correct places on the web page.

Ways to do this:

- Check that the dimensions of pictures and also other form fields do not cause too many problems if the web browser window is rescaled by the user.
- Test the application in a variety of different web browsers paying particular attentions to how the interface displays and whether or not it functions properly. Web browsers to be tested include Mozilla Firefox, Google Chrome, Opera as well as Microsoft's Internet Explorer

The main design of the application and the interfaces that are present to the user are critical in terms of the HCI of the developed application. Coupled with this is the fact that the interface should provide the users with a quick and easy way to navigate through the application.

Evaluation of the Results of the application from test case scenarios

In this section of my evaluation I will look at evaluating the results of my development against the scenarios that I have set out in the interim report and that are also detailed within the Introduction of this report.

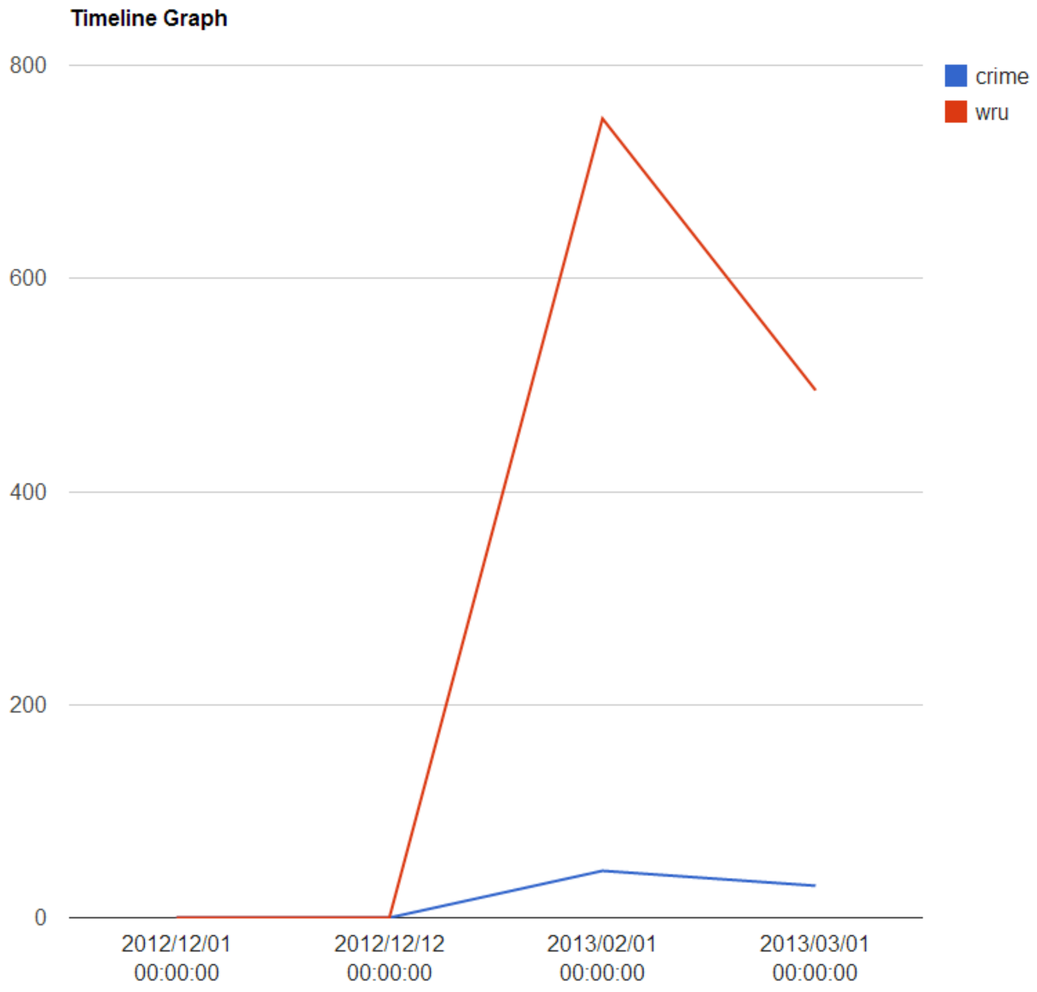
Scenario 1

A group of social science researchers are looking into how many tweets on a certain word the residents of the UK are making in the run up to the elections. They want to be able to easily compare the amount being said about a specific political party on a week by week basis during the months leading up to the election.

This would allow them to analyse the times at which more people were active in tweeting and whether or not this activity showed any correlation with other electioneering events. For example, 'had more people been tweeting around the time of televised debates?' and also were there differences between people tweeting about different political parties at different times.

An intention of the system would be to provide a visualisation, in the form of a graph that would map the volume of tweets that contained a specific keyword against the date and time they were tweeted. These would then be aggregated to form a week by week view of the number of tweets being made.

My application provides an easy to use graphing visualisation that would allow researchers to be able to specify keywords they wished to search and also time frames to search within. The tool would allow them to build up a graph showing the number of tweets for each keyword around different time frames and could help in deriving conclusions such as whether or not more tweets had been made around the time of the televised debates of the general elections.



<u>Scenario 2</u>

Researchers may wish to look at areas of a country to see where the most social media activity about a chosen subject lies.

For example there could be a group of researchers wishing to view which areas of Wales are most active in voicing their thoughts when areas of the country have been flooded. This would allow the researchers to maybe gain an insight into which areas of the country have been affected most by the flooding from the numbers of people that are tweeting from similar locations. It would also allow researchers to perhaps gauge some of the different effects the flooding is having on people in different areas from what they are actually saying in their tweets.

An intention of the system would be to allow the user to map where the tweets are being sent from. By aggregating the number of tweets that are made in a country you could then start to build up a form of 'heat map' as your visualisation to give you a better representation of where tweets are being sent from and also the volumes of tweets at certain places. It would be hoped that the system could also provide the content of the tweet that the poster has made and map these accurately to their locations.

Users are easily able to map tweets on data that has been obtained from the twitter social media service. These tweets are displayed with markers on a Google map overlay where, when hovered over, the contents of the tweets are clearly displayed. The mapping visualisation also provides the functionality for the users to add a heat map overlay to their mapped data. This allows researchers to easily see where the greatest or least numbers of tweets are being posted from. With this tool the researchers would be able to easily see which areas of Wales had been most active in voicing their thoughts at a time of flooding.

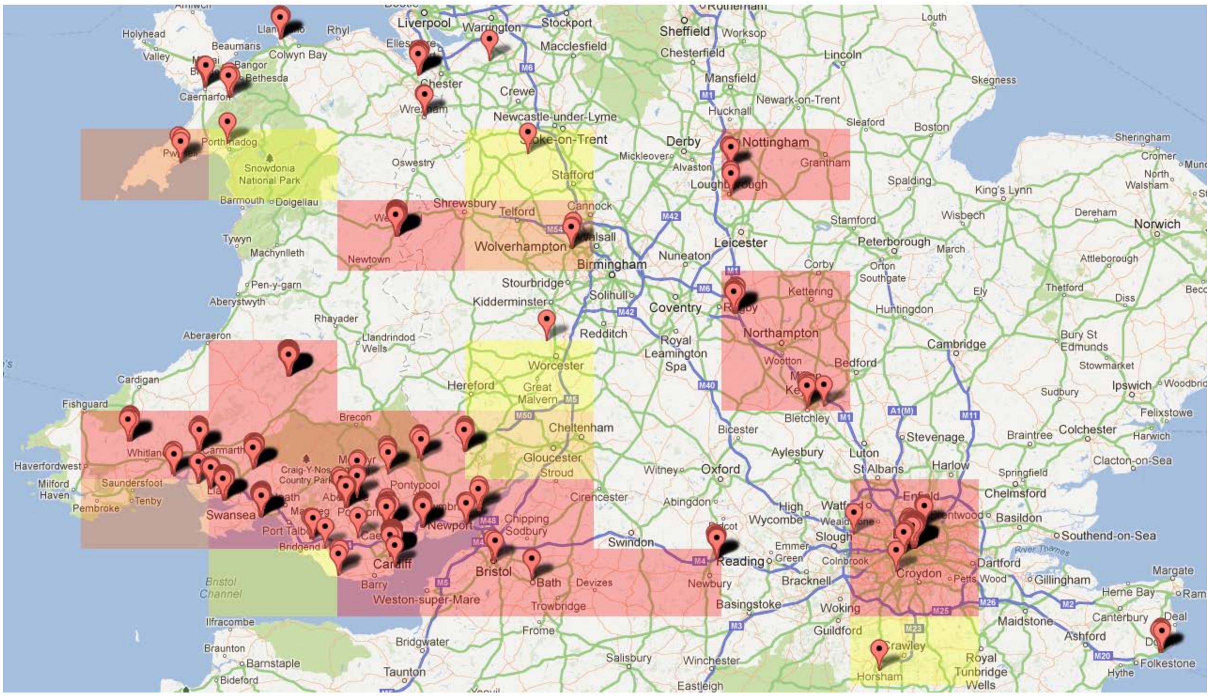


Figure 43 - Heat Map Visualisation

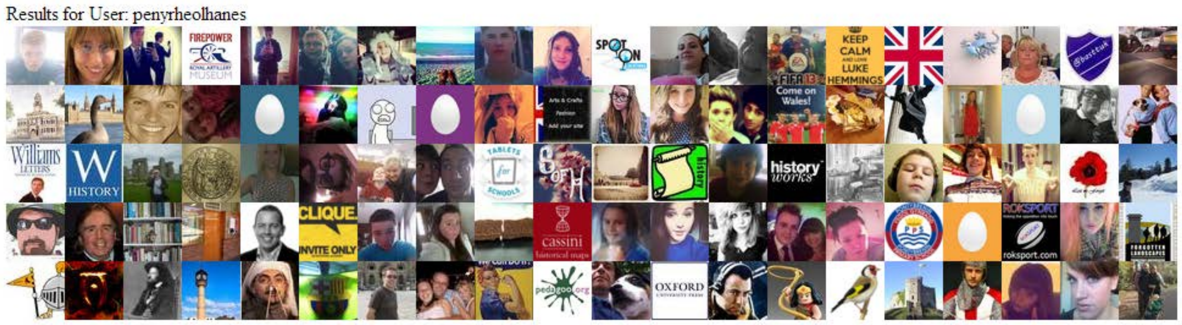
<u>Scenario 3</u>

Researchers may wish to see the connectivity of different group of people and indeed different individuals who may have the same interests or be talking about similar subjects on social networks. This could allow them to analyse interactions between individuals and groups of people on specific topic areas.

For example there could be a group of researchers wishing to view the connections between different individuals on Twitter in order for them to see who is connected in terms of a specific keyword. There would be a need to have a good visualisation that showed these links between different Twitter accounts so that researchers could get an in-depth insight into who is connected to who on different subjects. Analysis could then focus on the trends of people who are talking about different issues and whether or not there were groups of people that were always active for certain fields of interest.

The system would aim to provide a visualisation including the tweeters username with all the relevant links from them displayed in a clear manner that was easy to understand. It would probably be most beneficial to display those users with the most connections at the centre of the visualisation with those less connected on the periphery. Hopefully by displaying the data in this way it would be easier to see where sub groups were forming.

My implementation of the application would allow researchers to be able to build up a list of connections between users of Twitter, this could allow them to build up an idea of twitter users with similar interests. The developed application is somewhat limited in this area however with emphasis needing to be given to this visualisation if future work was to continue. That aside, researchers would be able to easily see user names and profile pictures of the followers and followers of specific twitter members.



Results for User: penyrheolhanes

Figure 44 - User Links Visualisation

Scenario 4

Researchers may wish to compare results obtained from the social media analysis side of this system to those statistics that are provided by official agencies such as the UK National Statistics Office.

This could be useful in justifying the use of social media data for research purposes but would ultimately allow researchers to show if there is any correlation between the results obtained from social media and that that is provided from official sources, by being able to have the same types of visualisations available to be applied to both data sets it would allow for easy comparisons between the obtained results.

The implemented system would aim to provide a means of researchers uploading information that they had received from official sources and be able to visualise them with the use of a heat map in the same way as they would be able to do when harvesting information from social media services. It is my intention that all the options provided as part of the mapping technique in the live visualisation would also be available to be used for visualisation of these pre-determined data sets. By providing a service such as this I would hope that the system could be used as a useful tool to see the correlations between the different data sets.

Implementation of my system has meant that researchers would be able to use their own data sets when using the mapping visualisation of the service. By including this feature it could provide researchers with the ability to create visualisations that all follow a standard format and allow them to easily justify whether or not the results of using the social media data tie in with the data that is provided by official sources. The system allows the user to easily upload a text file with their data (in a specific format) and then apply the mapping visualisation to this, along with the heat map overlay functionality. This is a simple way that allows for the analysis of correlation between the different data sets.

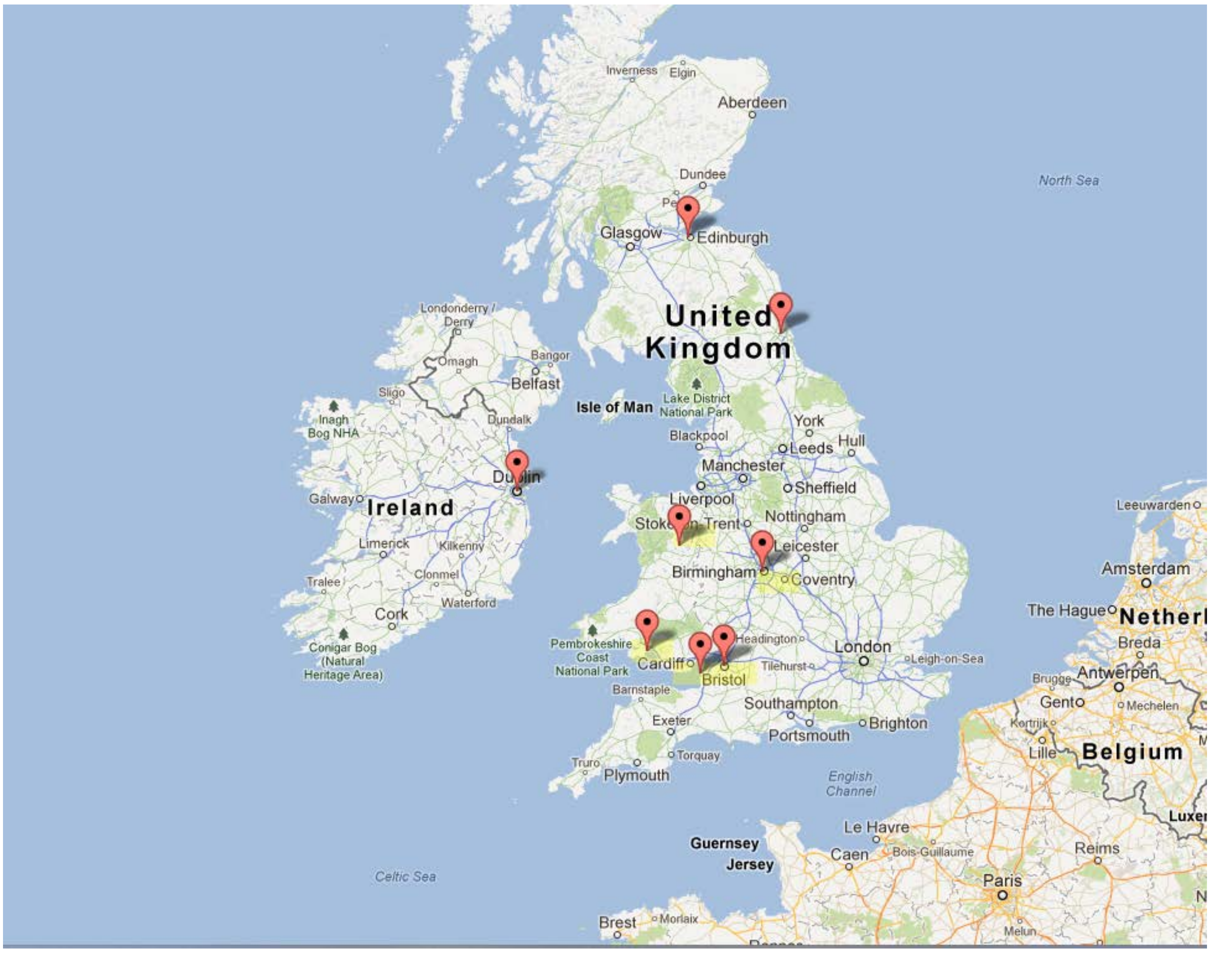


Figure 45 - File Upload Visualisation

Evaluation of Visualisation Features

Here I have tested each feature of the application, I have stated which feature I am testing. The data that has been used, an explanation of the expected results and I have provided screenshots (see Appendix) showing the result of each test.

Test Number	Test Name	Input Data	Expected Results	Results (Figures in Appendix)
1	Retrieve tweets from Twitter	Keyword : Crime	Tweets with the keyword crime would be added to the database and the value of the total number of tweets with that keyword are displayed	Tweets downloaded successfully, number of tweets shown (Figure 46)
2	Heat Map Tweets	Keyword : Crime	Tweets will be plotted on the map, and a heat map overlaid, onto the results	Tweets plotted correctly, heat map successfully applied (Figure 47)
3	Change map properties including heat map area	Previously created map with keyword : crime	Change in zoom level of map, heat map area recalculated	Map zoom successfully changed, heat map function applied (Figure 48)
4	Create database visualisation	Keywords : Crime, Health and Rugby	A choice of charts given to show a visual representation of the number of tweets in the database	Database successfully queried and a choice of charts given. Shown is a pie chart (Figure 49)
5	Create word cloud	Keywords: Crime, Health, Rugby, Wales, Sunny, Transport, Computers	A word cloud created, with the specified keywords. Each word in the word cloud should link to twitter	Word cloud successfully created, words link to twitter when clicked (Figure 50)
6	Create Twitter timeline	Username : csicardiff	The timeline for the specified user should be displayed	Specified user timeline is shown on screen (Figure 51)

7	Upload Text File to map	Text file with data for some UK cities	The cities that have been included in the test data file should be plotted onto the map. The heat map overlay should also be able to be applied	Tweets successfully plotted, heat map overlay can be displayed (Figure 52, Figure 53)
8	Create user links visualisation	Username : bbcnews	The profile pictures for every person the account 'csicardiff' follows should be displayed.	All followers displayed with profile picture, you can click on the picture to be taken to their account. (Figure 54)
9	Tweet Vs. Time visualisation created	Keywords : Crime, WRU Dates : Date ranges from 01/12/12 to 01/04/13 entered	A line chart displayed showing the results of the database query	Chart successfully displayed (Figure 55)

Evaluation of Ease of Use

To evaluate the ease of use of my developed application I made it available for a fellow student to test. In testing it I asked them to report any bugs that they found in the system and also anywhere the system did not perform as they would have expected, I performed the test with the think aloud method as this enabled me to sit in the room with the tester and have a clear understanding of why he was taking certain actions and also where any issues were arising. During the testing phase however I did not answer any questions that the tester asked me as this would have been counterproductive in my evaluation of the systems ease of use.

The results I obtained from using this testing method were very encouraging and my tester was able to complete all tasks effectively. He commented on the simplistic interface that was provided as part of the application and liked the fact that the same interface had been applied to every web page that had been used. The tester also commented on the thumbnails that were used stating that they were clear to use and that they gave a good representation of how the visualisations would look.

There was some confusion with the uploading of users own data for heat mapping as the tester selected the wrong link in the first place, not quite understanding the “Upload Text File to Map” thumbnail that was provided. The tester of the system did also comment on how for the ‘Tweets vs. Time’ visualisation having to enter every date and time was a slightly time consuming exercise and that if you made a single mistake then you would need to start over again.

The Appendix shows the tasks that I had set for the tester including comments that I made whilst observing the evaluating process and also some comments from my tester. Overall I am happy with the feedback that I have received as part of the evaluation process. Most comments and observations were very positive towards the system and the tester did indeed state that if it was released then he would like to be able to use parts of the mapping service for some visualisations he was wishing to create.

Some areas of trouble were pointed out and would need to be addressed further in future work on the system, however they were minor details and did not in any way impede with the overall operation and effectiveness of the application or the services it provided. Taking a few of the points raised, which I have stated above, simple fixes like changing the thumbnail for the file upload could be provided as well as providing a way with the “Tweets vs Time” feature that the data that had been entered was stored if there was an error with input and this being displayed in a clear manner to the user.

Processes such as these are a key part in the evaluation of the results and engaging the feedback of users of the system, they can help shape any ideas for future development and also allow for any bugs or errors to be more easily identified and fixed.

Problems Encountered

The first problem that I encountered when developing my application was due to the fact I had chosen to create a web application. This meant that to test any code that was written it has to be uploaded to a server as PHP is a server side processed language. This created two main problems, the first one was the need for me to be connected to the internet whenever I wanted to work on my application, this often restricted where I could do work on the implementation due to the poor internet connections or even due to a complete lack of internet. The other problem that stemmed from the need to use a server was the due to the FTP client that I was using needing to be authenticated onto the server about every five minutes. This meant that if I was working on code and not uploading it within a period of five minutes then I would need to re-establish the connection with the server again slowing down the development process.

Another problem I encountered from having to use a server was, that by cost implications, I was restricted to using the University's server that they provide to all students, whilst this was to a great extent suitable for my needs, there were often periods of quite considerable downtime on the sever, which then delayed me within the implementation stage. Alongside this I was also subject to using the Universities SQL Database server, which again was also subject to some downtime and scheduled maintenance.

A quite substantial problem that I had encountered after performing my background research was that one of the technologies that I had intended to utilise, NodeXL[4] for graphing links between users on social media services, was not compatible as part of a web application. The NodeXL software package was a standalone piece of software that used Microsoft Office's Excel functionality in order to both import the data required and display the data in a graph form. There was no way that I could implement this into the web application I was developing and also no easy way of providing it for others to use. This ultimately led to me not being able to implement this feature quite as I wished. This is something however that could be looked into further if future development of the application was to go ahead.

Another problem I realised I would encounter during my background research for this application was that many of the APIs that I would be using would indeed provide limits on the number of calls you could make per day, that is to say they limit the number of requests they will accept from a specific user. For example the Google Maps API only allows 2,500 requests to be made to it per day. This was however a problem that I deemed would not affect me too greatly during the implementation and testing phases of the application as I was unlikely to reach the daily limits, and if indeed I did then I could postpone the continued use of the application for another day.

There is a problem that also stems from some people not providing locational data on their tweets, as twitter does not force you to provide locational data. Currently location data is provided either by taking the location that is specified within a user’s profile, which can lead to inaccurate mapping, if for example someone’s profile says they are from Swansea but they are tweeting from their day out in Birmingham. The other way that locational data is collected is much more accurate using the GPS information that is normally supplied if people are using mobile devices. Within the current system this creates a problem whereby tweets that are retrieved with no locational information are placed by default on the map at the equator (coordinates 0^{o}N and 0^{o}W)

An additional minor problem that I did encounter was my unfamiliarity of the JavaScript language, and seeing as the Google Maps API relied heavily on this it was something that I had to try and teach myself quite quickly to be proficient enough in the language to understand how the Map API was functioning and how I could add my own features too it, by also writing them in JavaScript.

As always when developing a website there are discrepancies between how some parts of the website look and act depending on which browser you are using. Too this end, I have tried to ensure that my website will function as intended and display properly on as many of the major web browsers as possible, this includes the latest releases of Opera, Firefox and also Google Chrome.

Over the course of implementation, there have been many other minor problems that I have encountered, although through research from textbooks and online documentation often provided by companies such as Twitter and Google, I have been able to overcome most of these.

Future Work

I was able to achieve all of the objectives that I had set out to implement within my application. All the features that I have included are free from most glitches. However, there would be a fair amount of scope for improvements to be made to my project in the future. I have listed some of the main improvements as I see them below.

Future work on the project could always include the addition of more visualisation techniques and in particular those that are 'state of the art' so that the system could be kept current and allow the researchers who would be using it to provide the most up to date visualisations. This is something that could be looked at by the continuation or research into what visualisations are currently being developed and then implementing these into the system. What I have developed however provides a very good basis to add different visualisation techniques too

Future work into the heat mapping element of the system could provide a way of either not downloading 're-tweets' or even not mapping them, currently the system will map some re-tweets which often means that you can get a few tweets duplicated in similar areas. This could be something that is perhaps provided as a customisable option to the user, there may be some importance in the number of re-tweets that are made to some researchers

The user links visualisation would be a main point for improvement, due to the fact that I could not integrate NodeXL into a web application it meant that I was unable to work as closely with this visualisation feature as I would have liked. At this current stage the foundations are in place for a user name to be entered and twitter to be queried to find out the links that user has with other members of the twitter community, which are those people that they follow and those they are followed by. These users are currently displayed by means of their profile pictures in rows across the screen, which you can click on to be re-directed to their profile. It would be my intention that this would be developed further to show in some form of graph the directed edges between the nodes (people) that you are following or those that follow you. Hopefully this could also be developed to take into account who, in this subset of Twitter users, follows or is followed by others within the subset.

Sometime in the future, development could also be focused on ensuring that the web application that I have developed could be made available to as wide a range of audience as possible by harnessing the increasing use of mobile devices. This could be achieved by simply ensuring that the website displayed in an easy to use and clear navigate style for these devices such as the iPad, other tablets or even mobile phones.

Future work into this project could also include adding support for other social media services. The currently developed application only supports data that is retrieved from Twitter, however there is no reason why the visualisation techniques that are used here could not also be applied to data from other social media platforms such as Facebook and LinkedIn, as all of these have many themes in common, e.g. the posting of updates and also connecting with either friends, in the case of Facebook or contacts in LinkedIn. By adding support for a service such as LinkedIn, which is mainly used by professionals you would be able to start building up a visualisation such as a mapping service that showed the relationships of people who worked in a certain trade and where they lived within a specific country or region.

Work could also be undertaken on the visualisation of the number of tweets that are made over time. Currently this is only taken as a value of the number of tweets that are stored in the database, and not as a whole from twitter. It would be nice if there was a way in which you could see the amounts of tweets made on a topic on a specific day. However from my research this could be a slight problem, as the Twitter API I was using did not return an exhaustive search for all tweets on the keyword submitted for search. With further research into this however, there may be a possibility of combining a few technologies together in order to provide support for something of this manner.

A limitation of the system, as mentioned in my problems that I encountered whilst I was developing the system, is as a result of the limits that are imposed on the number of API calls that can be made per day, whilst these were acceptable during the implementation and testing phase they would cause problems should the application be released onto the world wide web. Future improvements could be made to aid the quality of the service provided by using one of the ‘paid-for’ APIs provided by Twitter. Currently only small samples of Tweets are available to be queried using the free API that this web application takes advantage of. The expansion to use the less restrictive (paid for) APIs would allow much more data to be returned per search and also enable a larger number of users to access the system simultaneously to acquire data from twitter and to perform visualisations. However if we were to look into using the paid for APIs there would be a cost associated with this, meaning that there would be a need to find a way of creating revenue to pay for these. This could either be through advertisements on the website, something I would be keen to avoid as this can detract from design leaving a page feel more cluttered. The other way to raise revenue would be to provide it as a paid-for service which could mean the introduction of monthly charges.

It would be my hope that any work that was conducted in the future would be in fitting with the work that has already taken place in implementing the system, focusing mainly on the ease of use through an intuitive design which is what I have tried to bear in mind throughout development of this project. This was one of the main objectives that I had set out in my initial plan and is key as the system is intended to be used by a wide target audience, some of those who would not necessarily have much knowledge in computing. Attention would also have to be paid to ensuring that functionality would still be acceptable across as many web browsers as possible.

Conclusions

Having completed the implementation and evaluation process of this project, I feel that the application I have developed fulfils all the requirements of the project description. All of the key functionality has been address and the application is in a usable state. Moreover I have also been able to include some features that are above and beyond the scope of what I originally thought would be achievable. Through the phases of my project, I have been able to gain an insight into the types of visualisation tools that are currently available, looking specifically at the functionality they offer and also the different types of data they can be applied to. From here I have then been able to develop my system of a visualisation toolkit to the specifications that I set out in my initial plan.

Overall I am satisfied with the quality of solution I have been able to provide for this project, both in terms of the developed application and also in terms of the research I have carried out whilst conducting various phases of planning and development.

I feel that the application I have created is appropriate for its target audience of researchers in the field of Social Sciences. I believe that my application could help them while they are looking at issues relating to their areas of study, for example looking at tension indicators within specific demographics of the population. As per my original specification, I have been able to provide a software infrastructure that supports the data collection and analysis of information from real time online sources.

From my development of the application one of the aspects that I feel was a key achievement was the implementation of the data retrieval from twitter. This was something that took me a considerable amount of time to implement, especially alongside the storage of the tweets in my database. Alongside this I am quite fond of the simplistic interface of the application that I have been able to use throughout.

I was pleased with my own methods of research and development, I feel that by setting time aside at the start of the project to research the types of visualisations that are currently 'state of the art' I gained a good insight into the types of visualisation and analysis techniques that I would be able to apply to my data sets. Having set aside a fair amount of time for this research I was then able to build up a strong design for my system and all my decisions were well informed at every step.

On the development of the application, I was also very content with how progress was made on the individual parts of the applications infrastructure. My methods of setting time aside within my week and also setting myself achievable deadlines for parts of the system I wished to be completed, meant that I was never at a stage where the workload would be too much for me. I now feel that it was prudent of me at the start of the project and also during my interim report to create a time plan and keep it up to date.

If I was to complete the project again, I would make more detailed notes about the research

I had performed. In a future project I would also set aside more time for evaluation at the various stages of development, for example looking back, I feel that it would have been a good idea to get someone to test and evaluate each visualisation technique as I developed it. It would also have been a good idea to have more regular meetings with researchers from the School of Social Sciences, however, in a university environment it is often difficult to find the time to do this.

Reflection on Learning

Whilst undertaking this project, one of the main skills I feel that I have improved upon is my own programming skills. I have enhanced my knowledge of the programming languages that I already knew and have also added to this range of languages by learning some new programming languages. This project has been a web based application and as such my skills in HTML have been enhanced whilst creating my web pages. I have also gained a further understanding of the PHP language through the continued use my project makes of it as part of some of the APIs I have employed.

With the JavaScript language, this has been a language that I have had to learn for myself throughout the implementation of this project, as it is not a language that I have encountered before within my programming career. I have been able to teach myself through a combination of applying skills of self-learning that I have picked up on during my time at Cardiff University and also by looking through examples and instructions in textbooks as well as completing online tutorials. As a result, I have seen an increase in my confidence while using the JavaScript language. Indeed at the start of the project I was almost trying to avoid the use of it, but now would be more than happy to use it in any other projects that I was involved in.

Another language I have learnt more about is the SQL language and in particular how it can be integrated into web applications, as I had only ever encountered it as a standalone language before, inserting queries into a terminal window. I have now found out how it can be easily integrated with both PHP and JavaScript.

Another major skill that I have had to utilise whilst completing this project is my skills in time management. I have always been able to plan my time effectively however this project has shown me that even if your time is planned, setbacks often dictate changes of plans at short notice. In a project for University, this has been difficult at times with many other coursework's needing completing and work for other modules needing to be done on a week by week basis. The majority of my project time has been time that I have set aside, and by abiding to this time on a weekly basis and setting achievable milestones along the way, I have been able to complete all that I had specified in the initial project plan.

My skills in research have also been enhanced by this project, as when I have come across technologies that I have had little knowledge of, I have had to make the best use of the resources available to me to overcome any problems I had. As a result of this I have become more familiar with finding relevant journals and textbooks on certain areas and also how to use materials such as tutorials that are present on the internet to get the best out of myself.

As with everything that I do, I feel that my skills of organisation have once again played a big part, mainly within the scope of time management but also with things like keeping some form of version control within my project, in case of loss of work or even functionality not working if I have been trying to implement something new into the system. As I have been working across many systems as well as a server this has been quite trying at times but due to a robust process I have created for keeping track of the different versions of the application, I have managed to avoid a few problems which could have turned out to be quite significant at times.

I feel that this module has been a key module as part of the course within Cardiff University, coupled with the experiences of the group project module we undertook last year. I feel that the skill set that I now possess has been enhanced and added to through these two modules mainly, but also alongside some of the other modules I have taken during my time in university. I would now say that I would feel more confident to tackle problems such as a project like this within a working environment, whether this was working as part of a group or indeed by myself. I would hope that the skills I have learnt here would be of value to any future projects or work I undertake.

Glossary

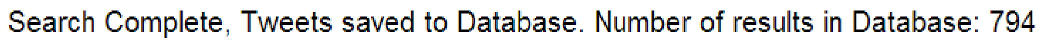
Term	Definition
Client Side	Operations that are performed by the local machine a client is using
Command(s)	A statement in computer programming that details how a task is to be done
Database	An organised collection of data that can be easily processed and queried
Geocoding	The process of finding geographic coordinates from data such as place names or street addresses
Geolocation	The identification of the geographical location of an object
Heat Map	A geographical representation of data where different values are represented by different colours
Hyperlink	A reference to the data that the reader can directly follow, a hyperlink normally points to another document
JavaScript	A scripting programming language that is used in creating dynamic websites
NodeXL	Network analysis and visualisation tool
Protocol	A Set of rules and regulations to determine how data is transferred within computer networks
Sentiment	The natural language processing of pieces of text or documents of text
Server Side	Operations that are performed by the server, for example a web server that runs remotely to the local workstation you are using
Site Map	A list of pages of a website used as a planning tool for web design. Pages are listed in a hierarchical fashion
Social Media	The interaction amongst people in a visual environment, the creation and sharing of information
Street View	A feature of Google's Mapping services which allow the user of the map to zoom in and see imagery of the area
Visualisation	Visual representations of abstract data
Web Application	An application that is accessed by its users over a network such as the internet
Web Hosting	A type of server that allows people to make their websites accessible via the World Wide Web

Table of Abbreviations

Term	Definition
AJAX	Asynchronous JavaScript and XML – a group of web development techniques used to create web applications
API	Application Programming Interface – A protocol that allows different software components to interact with each other
COSMOS	Cardiff Online Social Media Observatory
CSS	Cascading Style Sheets – A style sheet language used for describing the presentation of a document written in a markup language
FTP	File Transfer Protocol – A networking protocol used to transfer files from one host to another over a network such as the internet
HTML	Hyper Text Markup Language – The main programming language that is used to display web pages in a browser
JSON	JavaScript Object Notation – A text based standard for data interchange
PHP	Hypertext Pre-processor – Open Source programming language used to develop dynamic webpages
SFTP	Secure File Transfer Protocol – Network protocol providing file transfer over a reliable data stream
SQL	Structured Query Language – A programming language used for managing data in relational databases
URL	Uniform Resource Locator – Also known as a web address, a string that constitutes a references to a resource
XML	Extensible Mark-up Language – A set of rules for encoding documents in a format that is human and machine readable

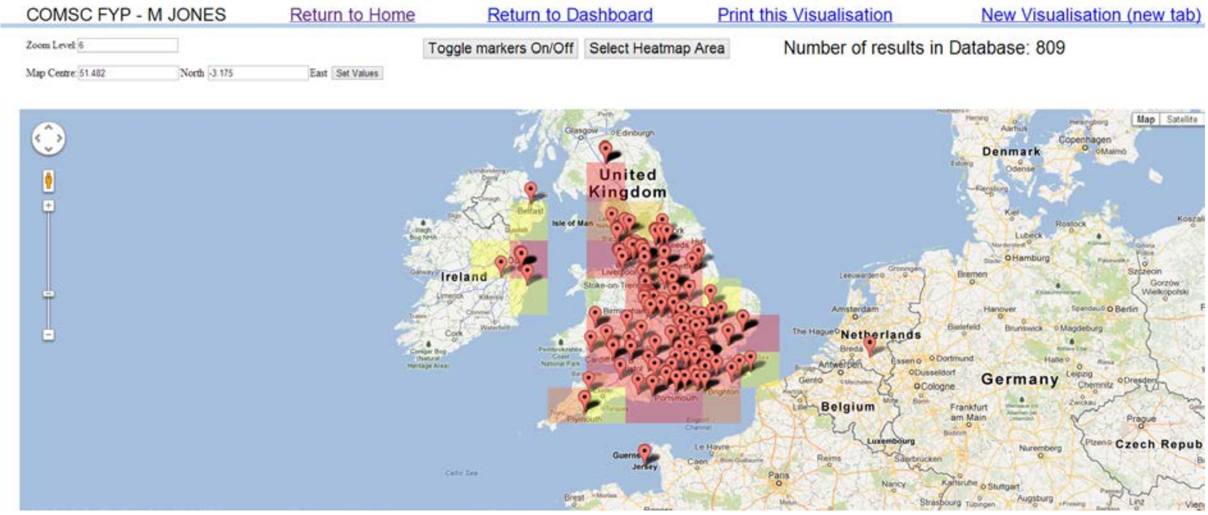
Appendix

Results of Functionality Tests



Search Complete, Tweets saved to Database. Number of results in Database: 794

Figure 46 - Number of Tweets Displayed after Retrieval (Test 1)



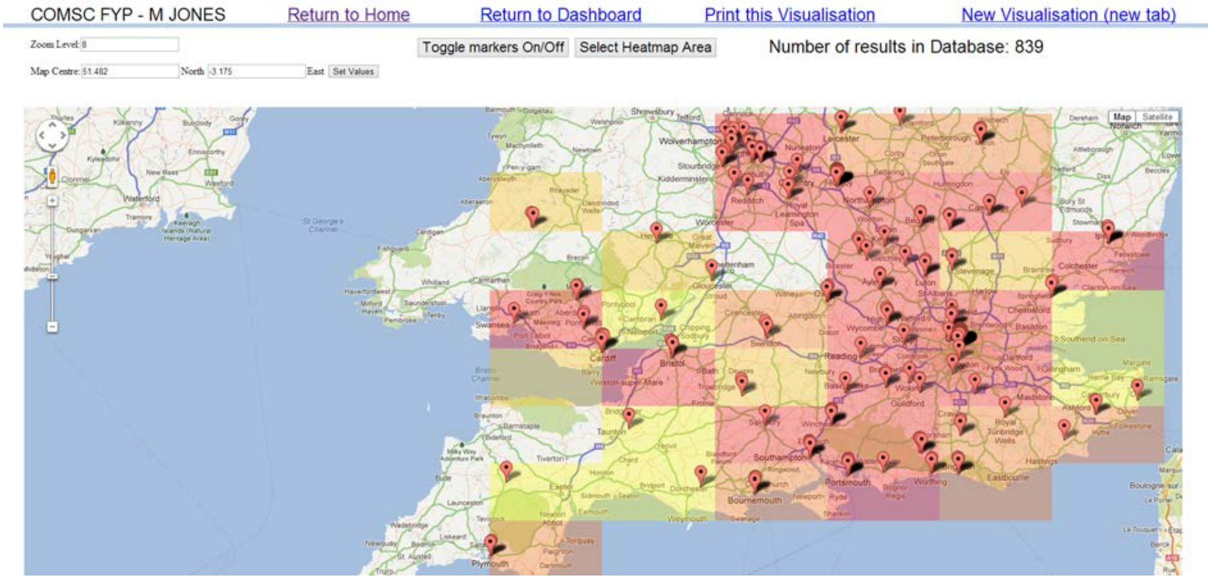
COMSC FYP - M JONES

Zoom Level: 6

Map Centre: 51.482 North -3.175 East

Number of results in Database: 809

Figure 47 - Heat map of Results (Test 2)



COMSC FYP - M JONES

Zoom Level: 8

Map Centre: 51.482 North -3.175 East

Number of results in Database: 839

Figure 48 - Heat map with Changed Parameters (Test 3)

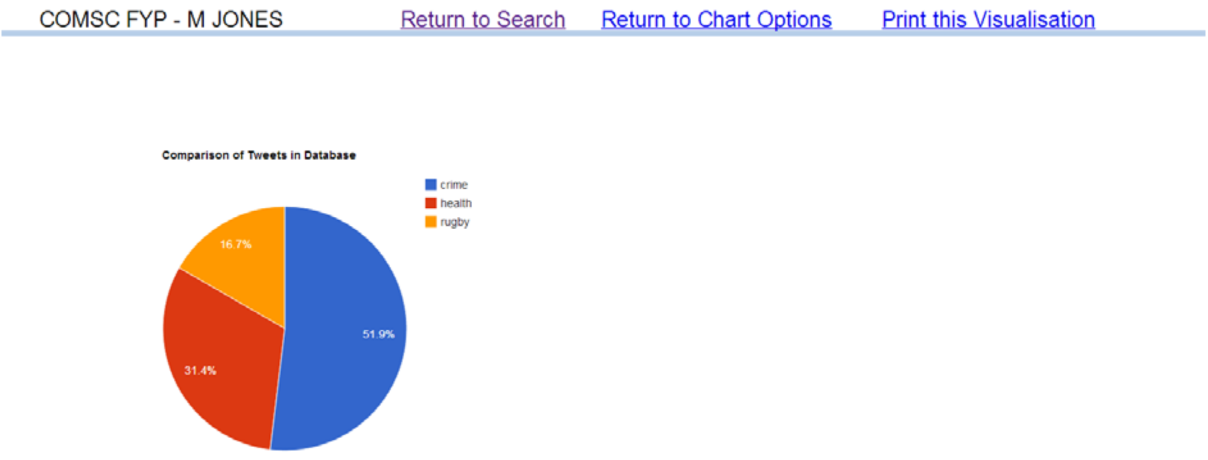

COMSC FYP - M JONES Return to Search Return to Chart Options Print this Visualisation

Figure 49 - Pie Chart Visualisation (Test 4)

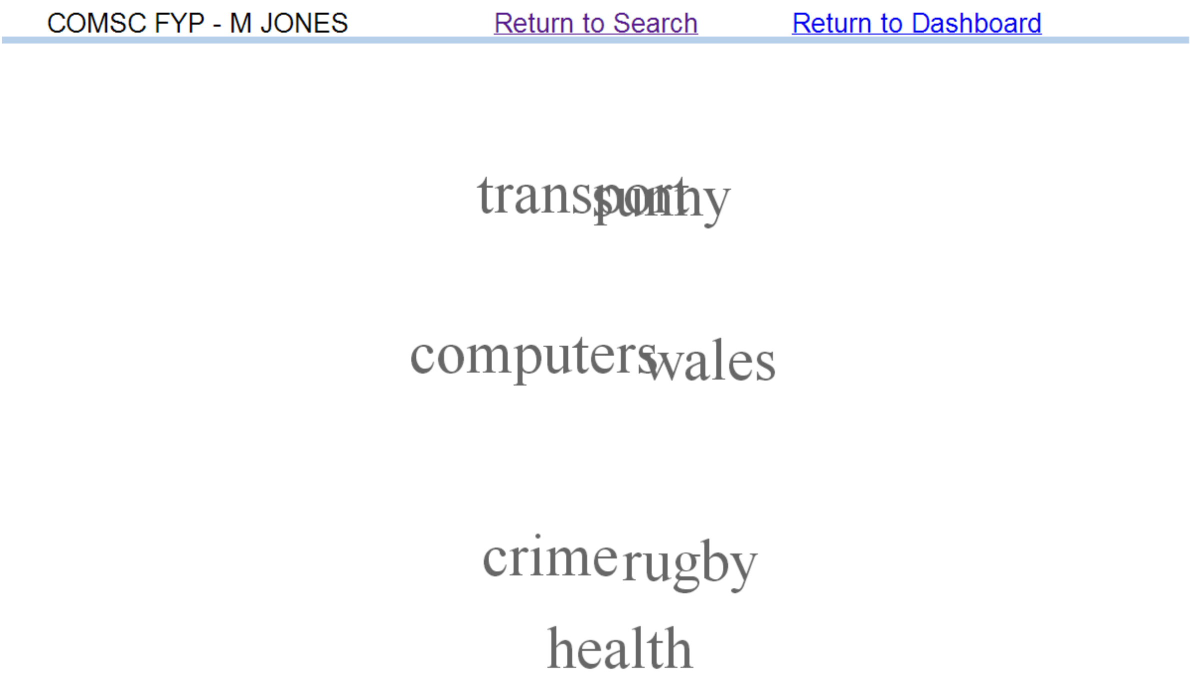
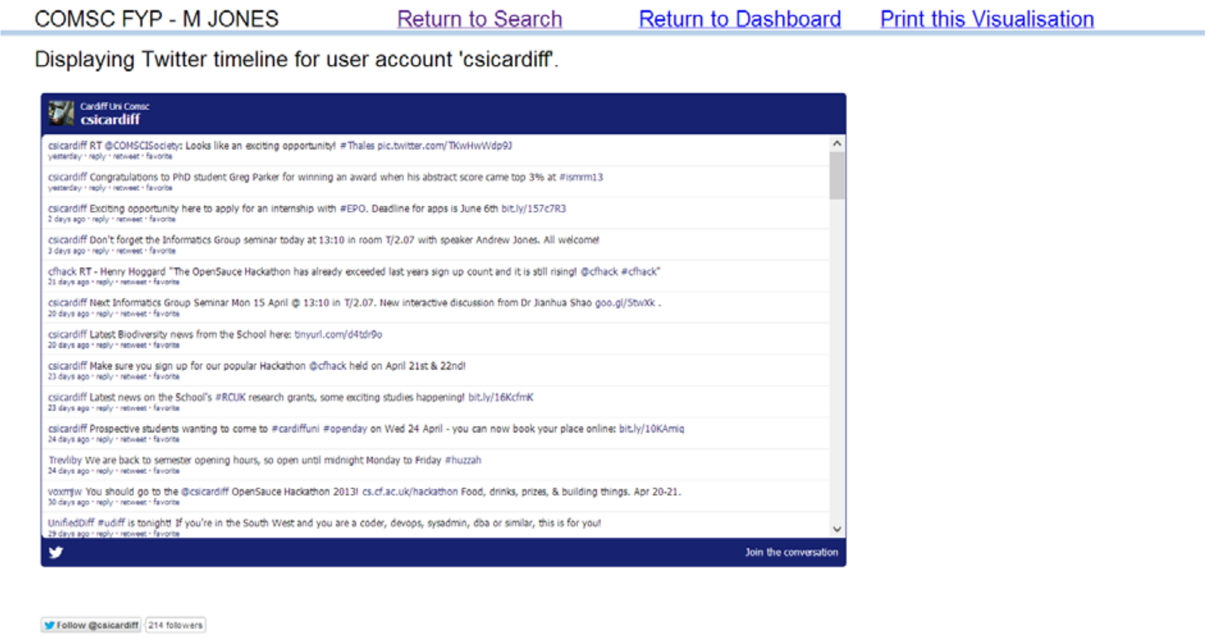

COMSC FYP - M JONES Return to Search Return to Dashboard

Figure 50 - Word Cloud Visualisation (Test 5)



Displaying Twitter timeline for user account 'csicardiff'.

csicardiff RT @COMSCISociety: Looks like an exciting opportunity! #Thales pic.twitter.com/TKwHwWdp9J
yesterday · reply · retweet · favorite

csicardiff Congratulations to PhD student Greg Parker for winning an award when his abstract score came top 3% at #ismrm13
yesterday · reply · retweet · favorite

csicardiff Exciting opportunity here to apply for an internship with #EPO. Deadline for apps is June 6th bit.ly/157c7R3
2 days ago · reply · retweet · favorite

csicardiff Don't forget the Informatics Group seminar today at 13:10 in room T/2.07 with speaker Andrew Jones. All welcome!
3 days ago · reply · retweet · favorite

cfhack RT - Henry Hoggard "The OpenSauce Hackathon has already exceeded last years sign up count and it is still rising! @cfhack #cfhack"
21 days ago · reply · retweet · favorite

csicardiff Next Informatics Group Seminar Mon 15 April @ 13:10 in T/2.07. New interactive discussion from Dr Jianhua Shao goo.gl/StwXk .
20 days ago · reply · retweet · favorite

csicardiff Latest Biodiversity news from the School here: tinyurl.com/d4tdr9o
20 days ago · reply · retweet · favorite

csicardiff Make sure you sign up for our popular Hackathon @cfhack held on April 21st & 22nd!
23 days ago · reply · retweet · favorite

csicardiff Latest news on the School's #RCUK research grants, some exciting studies happening! bit.ly/16KcfmK
23 days ago · reply · retweet · favorite

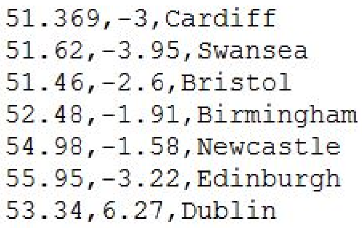
csicardiff Prospective students wanting to come to #cardiffuni #openday on Wed 24 April - you can now book your place online: bit.ly/10KAmiq
24 days ago · reply · retweet · favorite

Trevliby We are back to semester opening hours, so open until midnight Monday to Friday #huzzah
24 days ago · reply · retweet · favorite

voxmjw You should go to the @csicardiff OpenSauce Hackathon 2013! cs.cf.ac.uk/hackathon Food, drinks, prizes, & building things. Apr 20-21.
30 days ago · reply · retweet · favorite

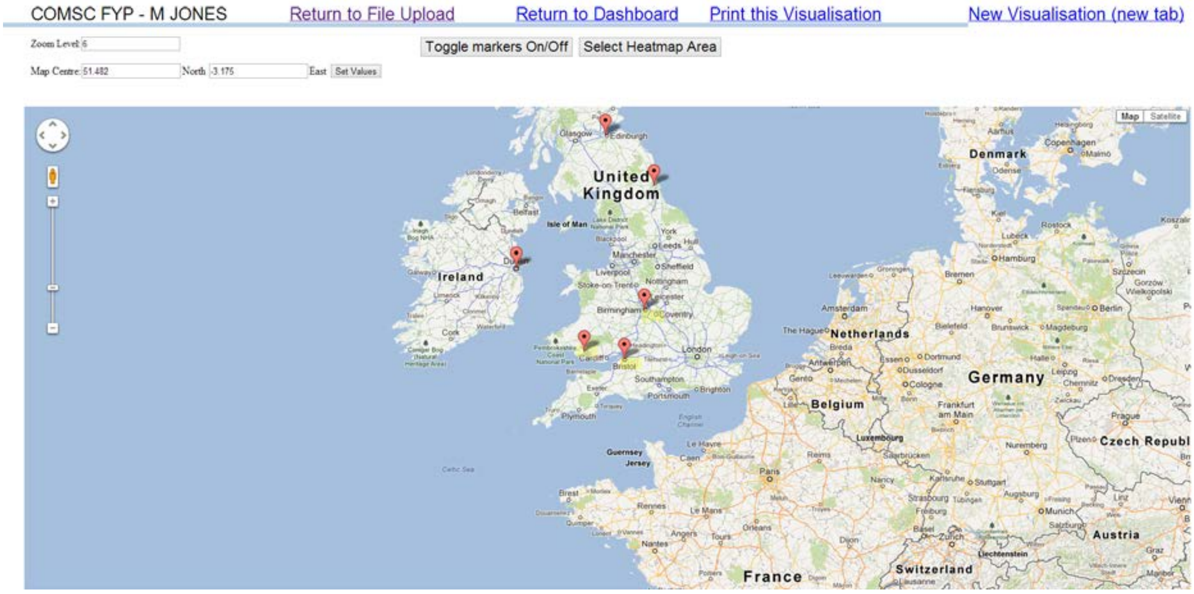
UnifiedDiff #udiff is tonight! If you're in the South West and you are a coder, devops, sysadmin, dba or similar, this is for you!
29 days ago · reply · retweet · favorite

Figure 51 - Timeline for 'csicardiff' (Test 6)



```
51.369,-3,Cardiff
51.62,-3.95,Swansea
51.46,-2.6,Bristol
52.48,-1.91,Birmingham
54.98,-1.58,Newcastle
55.95,-3.22,Edinburgh
53.34,6.27,Dublin
```

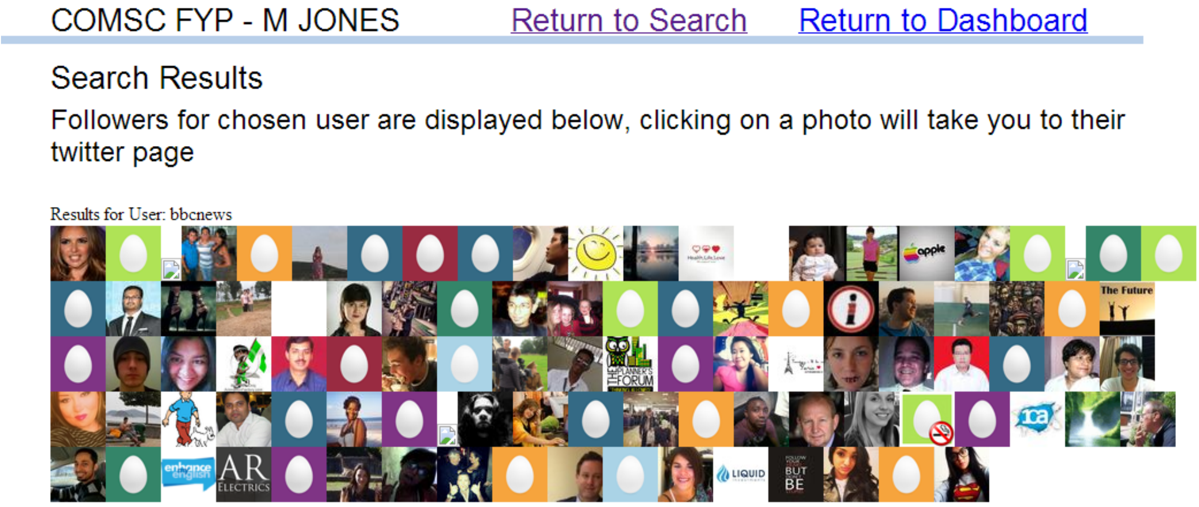
Figure 52 - Test Data Used (Test 7)



Zoom Level: 6

Map Centre: 51.482 North -3.175 East

Figure 53 - Map from Uploaded Data (Test 7)



Search Results

Followers for chosen user are displayed below, clicking on a photo will take you to their twitter page

Results for User: bbcnews

Figure 54 - User Links Visualisation 'bbcnews' (Test 8)

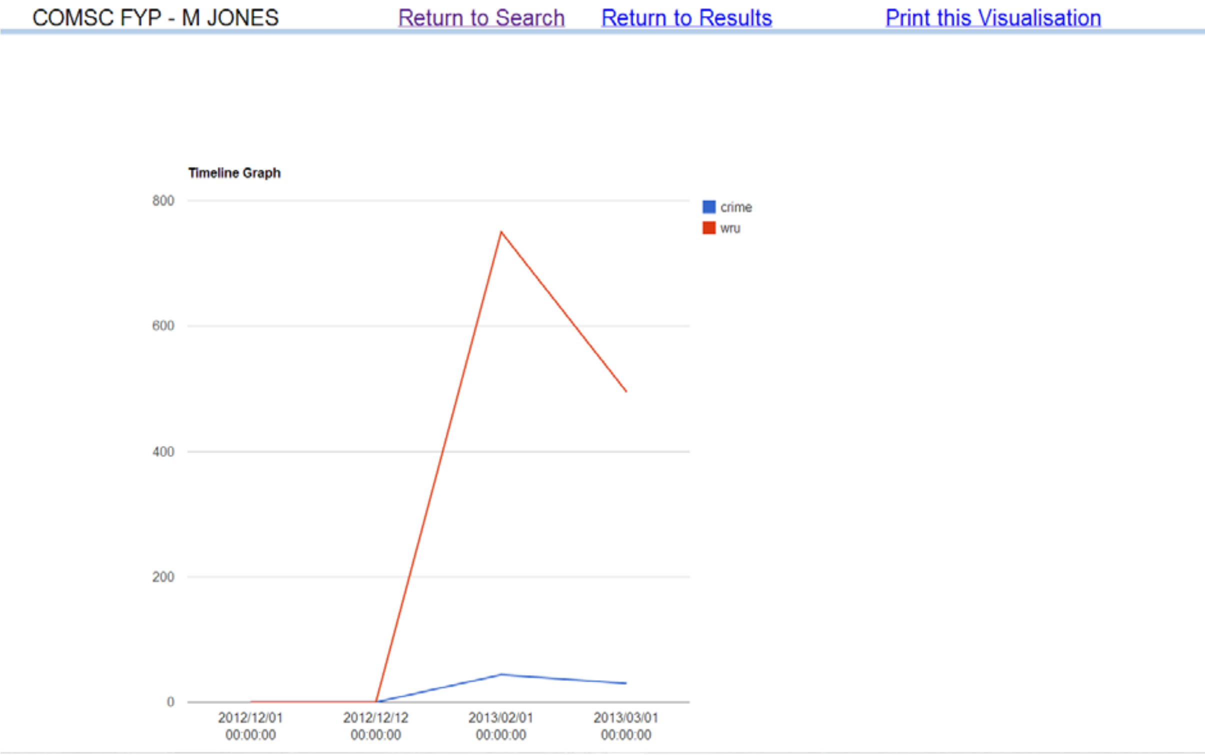
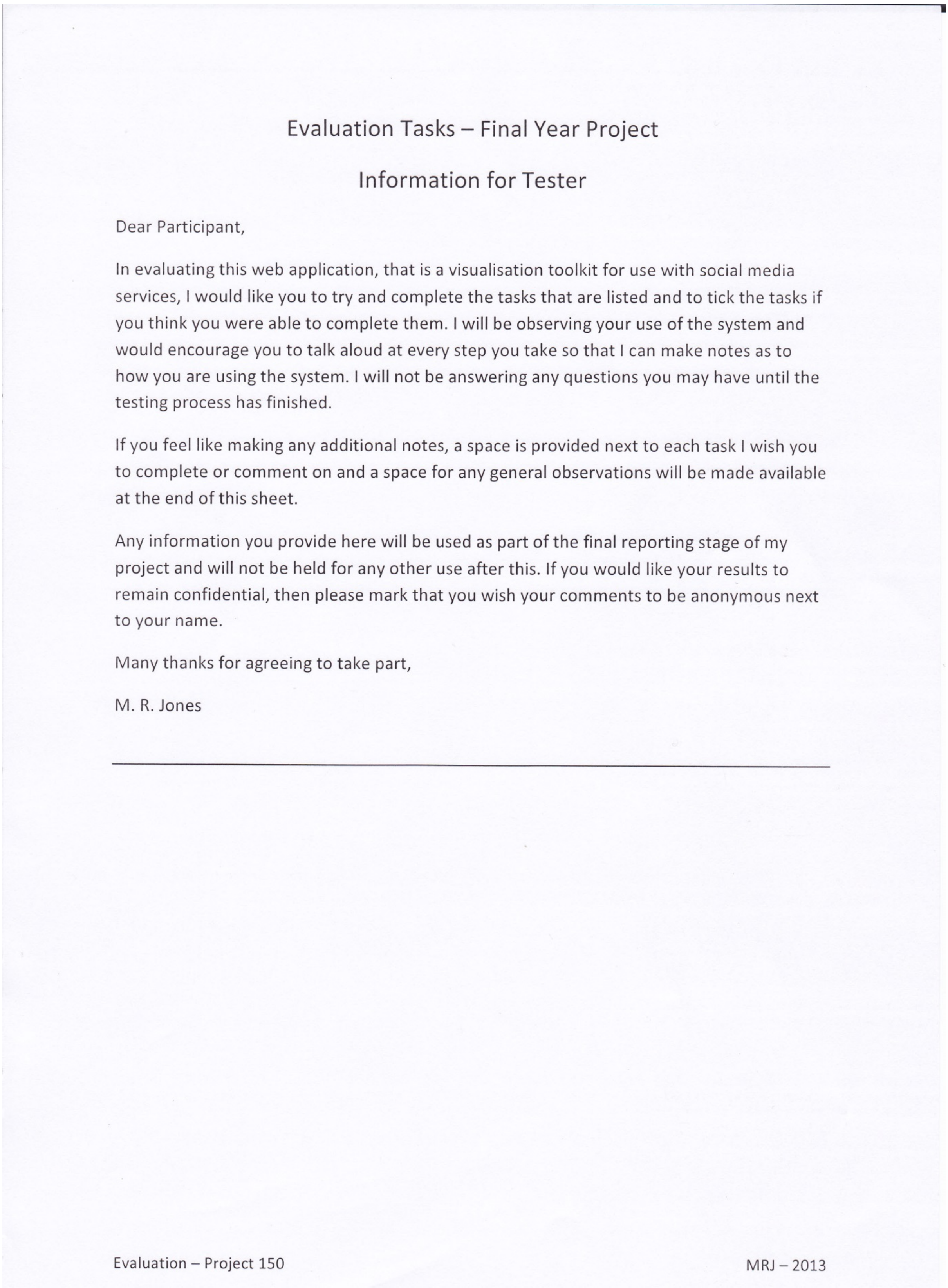


Figure 55 - Tweets Vs. Time Visualisation (Test 9)

Evaluation Test



Evaluation Tasks – Final Year Project

Information for Tester

Dear Participant,

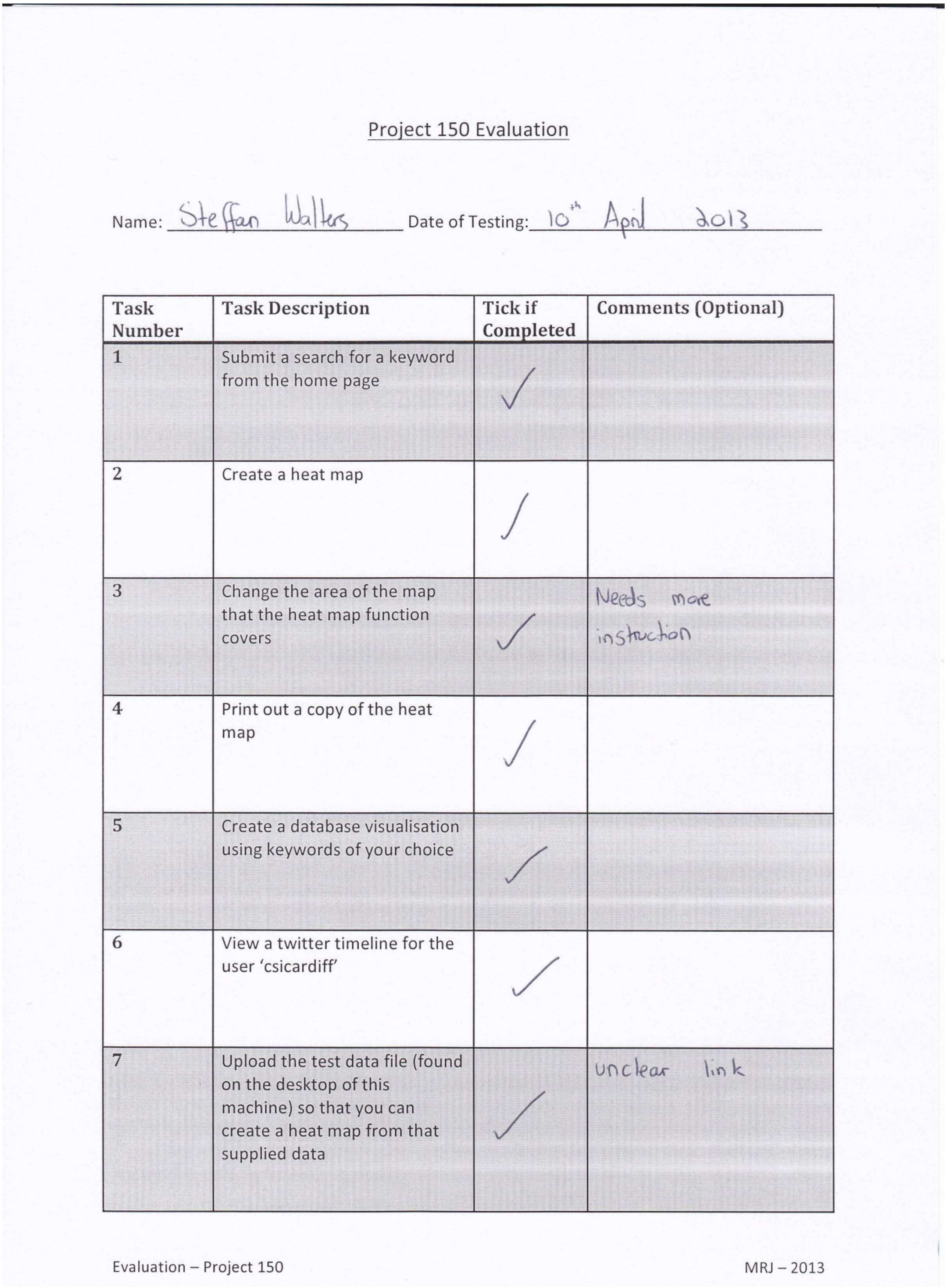
In evaluating this web application, that is a visualisation toolkit for use with social media services, I would like you to try and complete the tasks that are listed and to tick the tasks if you think you were able to complete them. I will be observing your use of the system and would encourage you to talk aloud at every step you take so that I can make notes as to how you are using the system. I will not be answering any questions you may have until the testing process has finished.

If you feel like making any additional notes, a space is provided next to each task I wish you to complete or comment on and a space for any general observations will be made available at the end of this sheet.

Any information you provide here will be used as part of the final reporting stage of my project and will not be held for any other use after this. If you would like your results to remain confidential, then please mark that you wish your comments to be anonymous next to your name.

Many thanks for agreeing to take part,

M. R. Jones

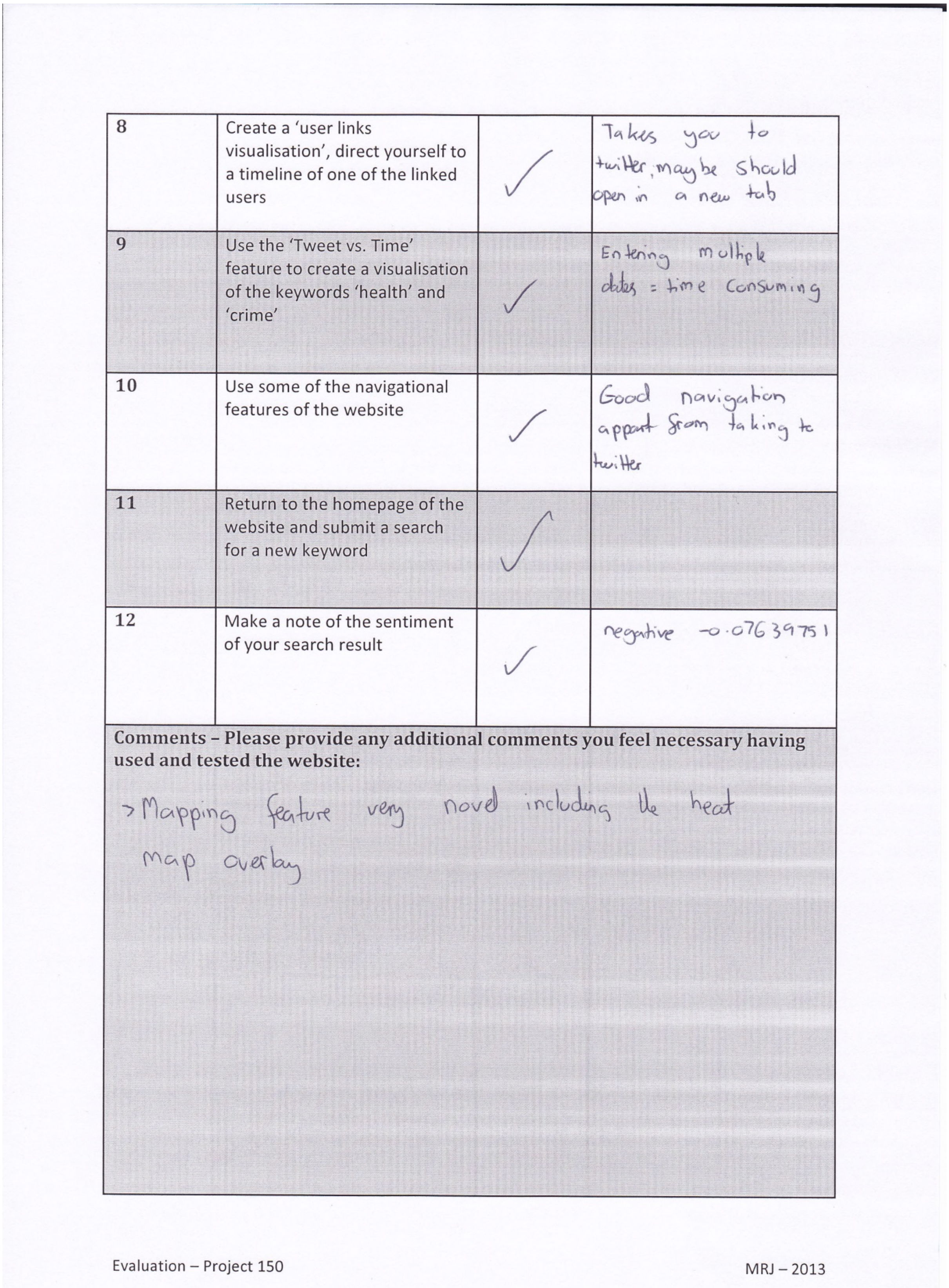


Project 150 Evaluation

Name: Steffan Walters Date of Testing: 10th April 2013

Task Number	Task Description	Tick if Completed	Comments (Optional)
1	Submit a search for a keyword from the home page	✓	
2	Create a heat map	✓	
3	Change the area of the map that the heat map function covers	✓	Needs more instruction
4	Print out a copy of the heat map	✓	
5	Create a database visualisation using keywords of your choice	✓	
6	View a twitter timeline for the user 'csicardiff'	✓	
7	Upload the test data file (found on the desktop of this machine) so that you can create a heat map from that supplied data	✓	unclear link

Evaluation – Project 150 MRJ – 2013

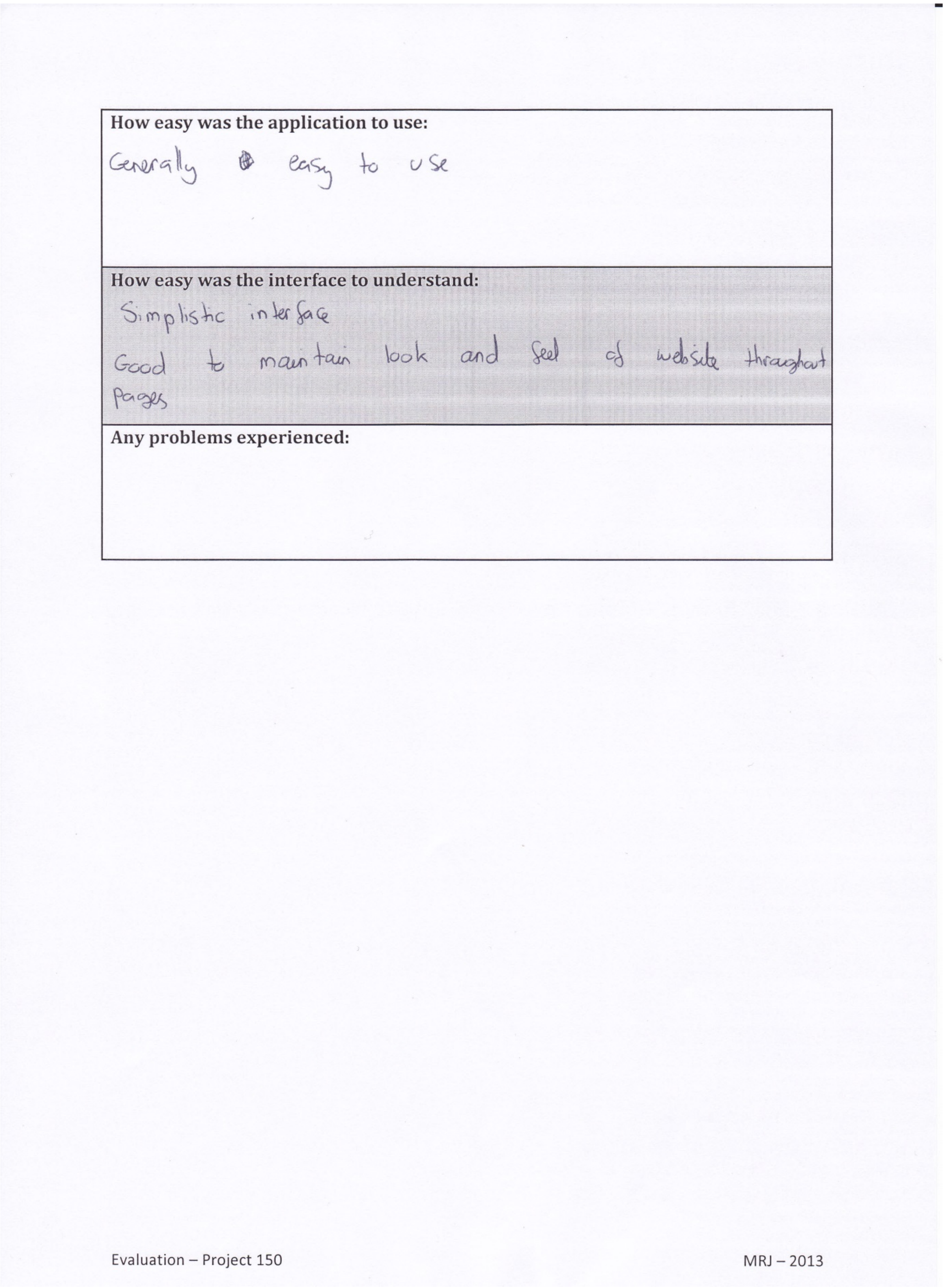


8	Create a 'user links visualisation', direct yourself to a timeline of one of the linked users	✓	Takes you to twitter, maybe should open in a new tab
9	Use the 'Tweet vs. Time' feature to create a visualisation of the keywords 'health' and 'crime'	✓	Entering multiple dates = time consuming
10	Use some of the navigational features of the website	✓	Good navigation apart from taking to twitter
11	Return to the homepage of the website and submit a search for a new keyword	✓	
12	Make a note of the sentiment of your search result	✓	negative -0.07639751

Comments – Please provide any additional comments you feel necessary having used and tested the website:

> Mapping feature very novel including the heat map overlay

Evaluation – Project 150 MRJ – 2013



How easy was the application to use:

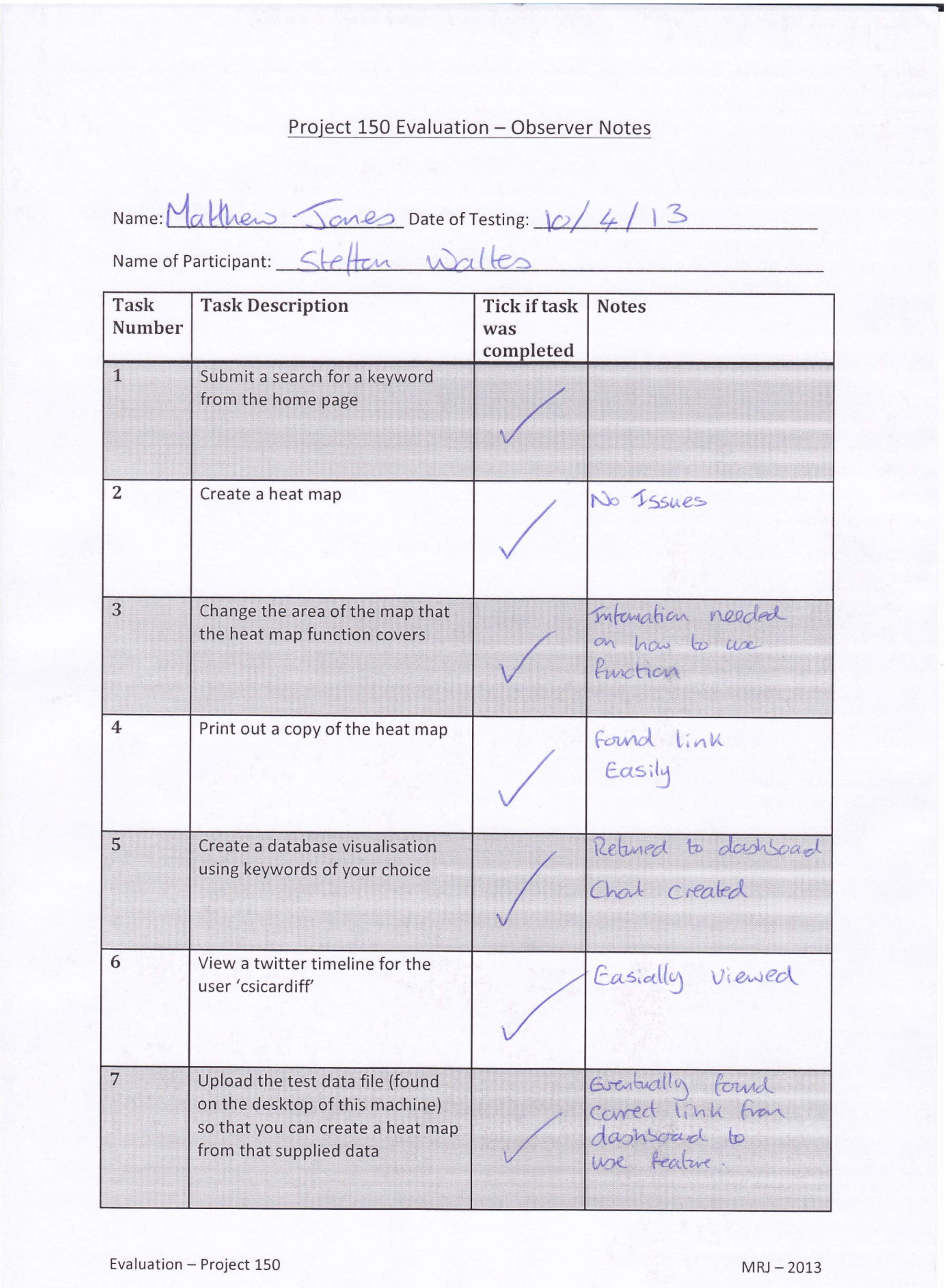
Generally easy to use

How easy was the interface to understand:

Simplistic interface

Good to maintain look and feel of website throughout pages

Any problems experienced:



Project 150 Evaluation – Observer Notes

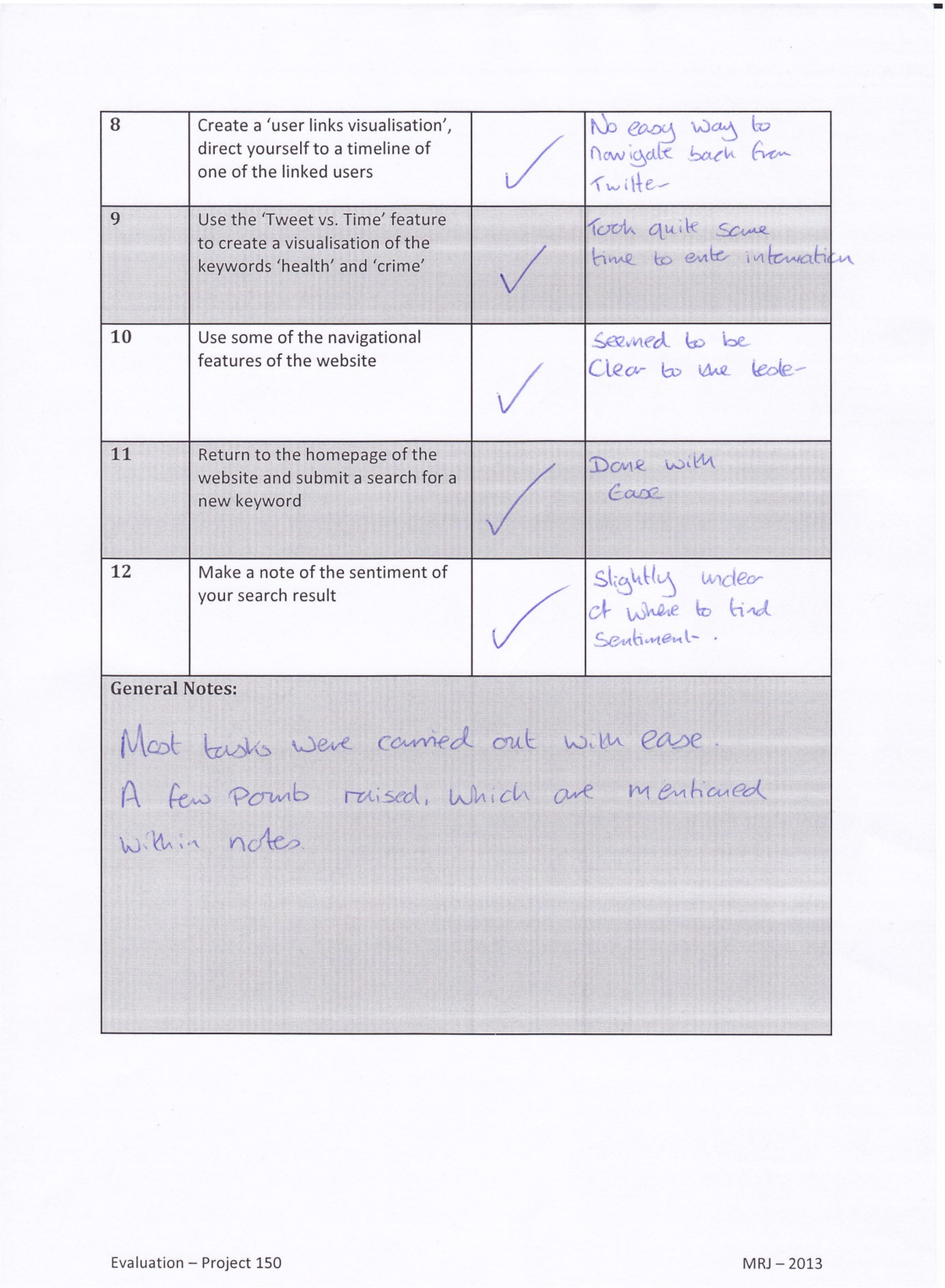
Name: Matthew Jones Date of Testing: 10/4/13

Name of Participant: Steffan Walters

Task Number	Task Description	Tick if task was completed	Notes
1	Submit a search for a keyword from the home page	✓	
2	Create a heat map	✓	No Issues
3	Change the area of the map that the heat map function covers	✓	Information needed on how to use function
4	Print out a copy of the heat map	✓	Found link Easily
5	Create a database visualisation using keywords of your choice	✓	Returned to dashboard Chart created
6	View a twitter timeline for the user 'csicardiff'	✓	Easially Viewed
7	Upload the test data file (found on the desktop of this machine) so that you can create a heat map from that supplied data	✓	Eventually found correct link from dashboard to use feature.

Evaluation – Project 150

MRJ – 2013



8	Create a 'user links visualisation', direct yourself to a timeline of one of the linked users	✓	No easy way to navigate back from Twitter
9	Use the 'Tweet vs. Time' feature to create a visualisation of the keywords 'health' and 'crime'	✓	Took quite some time to enter information
10	Use some of the navigational features of the website	✓	Seemed to be Clear to the reader
11	Return to the homepage of the website and submit a search for a new keyword	✓	Done with Ease
12	Make a note of the sentiment of your search result	✓	Slightly unclear of where to find Sentiment.

General Notes:

Most tasks were carried out with ease. A few points raised, which are mentioned within notes.

References

1. Twitter Developers. Open Source Examples. Available: https://dev.Twitter.com/docs/open-source-examples. Last Accessed 10th Apr 2013

2. Google Developers. Google Maps JavaScript API v3. Available: https://developers.google.com/maps/documentation/javascript/basics. Last accessed 10th Apr 2013.

3. *Alchemy API.* Available: http://www.alchemyapi.com/. Last accessed 10th Apr 2013.

4. Hansen D, Shneiderman B, Smith M.A (2010). *Analyzing Social Media Networks with NodeXL: Insights from a Connected World*. London: Morgan Kaufmann.

- Background Research - Jones MR (2012). Interim Report. Cardiff University

- Aims and Objectives - Jones MR (2012). Initial Plan. Cardiff University