TV List-ting

Ross Langan
40276526@live.napier.ac.uk
Edinburgh Napier University - Advanced Web Technology (SET09103)

Abstract

This report covers the design, implementation and evaluation of my web app, titled TV List-ting. Providing way to discover new and known programmes, this app is a collection of television programmes presented in a suitable format and contains information on several aspects of each show.

Keywords – Web, Technology, Website, Web-app, Flask, Python, Bootstrap

1 Introduction

TV List-ting is an informative web app designed to provide users with useful information on a range of popular TV shows. These include programmes such as Friends and Breaking Bad. Users can navigate easily through the web-app to discover different programmes, genres and ratings and find information on each show.

The homepage utilises a toolbar at the top of the page to link to each of three category pages. In the center of the screen are directions for finding the programmes. A click on any of the categories in the toolbar will take the user to the appropriate page.

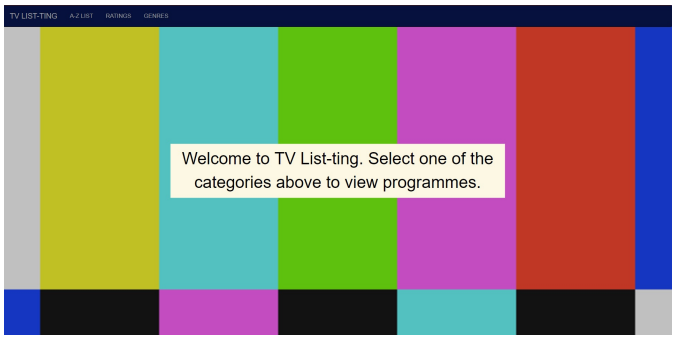

Figure 1: **Homepage for TV List-ting** - Simple, containing instructions to select a category

The programmes are sorted in three different ways, by A-Z, Rating and Genre. Each of these category pages contains a list of titles and programme names as shown in figure 2. The category pages link to all ten programme pages and use a simple url relevant to the page.

Each programme has its own page containing the full title of the show, details of what channel it is aired on and when, the logo of the show and the IMDb rating. As well as this there is an external link to the IMDb page and a short description of the show (Figure 2). This page also contains the nav-bar at the top of the page, allowing the user to reach any part of the web-app in a single click.

Also included in the web-app is the ability to view the programme's logo in a separate page. This page can be opened by clicking the 'View Image' button at the bottom of each programme page. A back button can be used to return to this page after viewing the image.

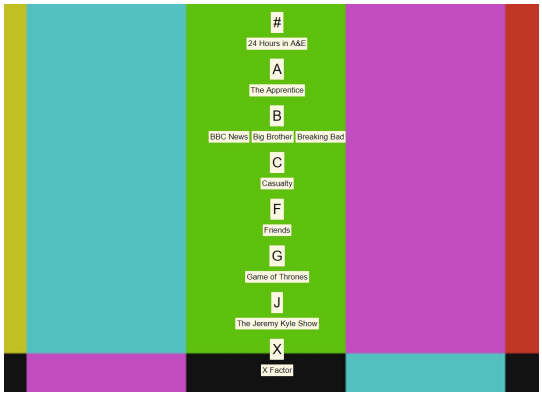

Figure 2: **A-Z List Page** - Showing each programme ordered alphabetically. Each programme is a linking to that page.

I chose the topic of a TV-guide style web-app because I watch all my TV shows on streaming websites such as BBC IPlayer[1] and 4 On-Demand[2]. I use these websites regularly to find programme listings so that I can watch live TV online and stay up-to-date with my favourite shows. I also discover new and interesting shows by browsing these sites and reading relevant reviews on IMDb[3].

2 Design

2.1 Page Structure

The web-app I have designed uses a hierarchical structure based around a single homepage and several categories branching from this. Each category branches further, containing links to all ten programmes, which in turn link to their relevant image page. Each page regardless of the category or programme links back to the homepage and all 3 category pages for simple and fast navigation around the app.

In my research I found that hierarchical structures are well suited to website organisation, and would definitely be useful for my web-app. This creates a natural flow to the app allowing the user to reach a programme or image from the homepage with minimal amount of clicks and page loads. Furthermore, the user does not have to navigate through

the homepage or other pages each time to reach the categories.

To help implement this structure effectively, I created several templates, beginning with the base.html template, for each page to inherit parts from.

The base.html template includes features used in the home and category pages. These are the bootstrap toolbar containing the home page link and three categories, and the background. In the body tags I use {% block content %} and {% end block %} to contain the page content for each page.

To add content to each of the category pages, I use a further template which inherits from the base template. This ensures consistency regarding the nav-bar and background for each of the pages. It is in each of these templates I add the content between the start and end blocks as seen above. I include titles, show names and hyperlinks to fill each of these pages with useful content.

Following any of the programme links will take the user to the relevant programme page. These use the template page-Base.html, which does not inherit from base template. The reason for this is because I chose to use different sized margins at the top and side of the page compared to the home and category page, as well as a black nav-bar to go with any of the multiple background colours. The content is added to the page by taking in matched string in my python file and displaying it. At the bottom of each page is a button, which when pressed, will load a new page containing the programme's image and another button, this time returning the user to the previous page.

Just like the programme page, the image page also uses a separate template. This is because this page only contains a 500x500 image and a button. Both of these pages can be seen below.

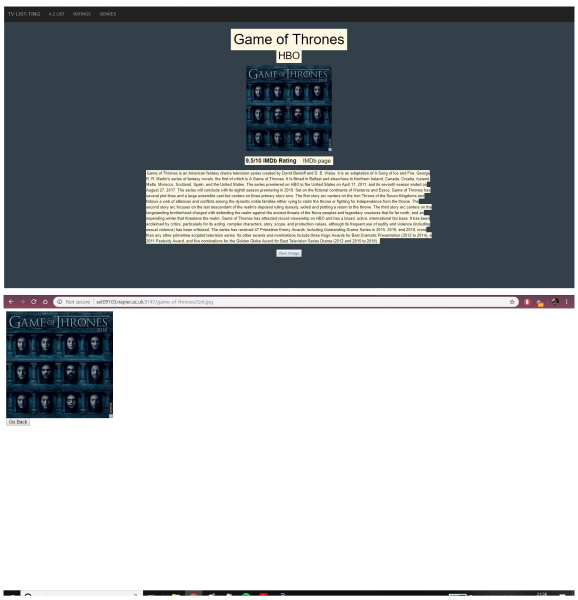

Figure 3: **Programme page (top) and image page (bottom)**

2.2 URL Structure

The url structure of TV List-ting matches closely to the page hierarchy- every new layer adds a new part of the url. The homepage of the web-app can be found by navigating to 'set09103.napier.ac.uk:9147/home'. I chose to include /home as it is a common and logical way to find a homepage. The topic I chose and the decision not to include sub-menus for inside the categories (separate page for each rating, letter or genre) means that all ten programmes are in all three categories. I chose to use three separate URL's for each programme, one for each of the categories followed programme name. Each image can also be reached in this way- with one of the three categories, followed by the programme name, followed by the image name. The navigation map for my URL hierarchy can be seen below in Figure 3.

Figure 4: **Navigation map for URL hierarchy**- A higher quality image is available in the appendices section

I chose to use a url structure that is both simple and consistent. Each time a user navigates to a page on a lower level (an expansion of the previous page) another section of the url will be added. By using key words including the category names and programme names in each part of the url, I ensure that the user is able to understand what is on each page simply by reading it. The user can navigate directly to each page by replacing the words in brackets below.

/home
/categories/(category_name)
/categories/(category_name)/(programme_name)
/categories/(category_name)/(programme_name)/
(image_name).jpg

2.3 Visual Design

The first visual element I used is the bootstrap toolbar. I chose not to center any of the text buttons in case I chose to have a search bar later in development. I chose to use a dark blue for the nav-bar because it would suit most colour schemes, and does not restrict me to a light coloured background (A light toolbar and very dark background may look strange).

The background initially was an image of the all too familiar black and white TV static, however due to difficulty making the text clear to read, I decided to use the TV test screen instead. By centering text on each contents page, I was able to use the vertical lines of the background as an aid for displaying the text clearly in a list. For each of the programme pages I used a hex colour picker to choose a background colour similar to the main colour in the programme image. As expected, this makes the background of the X-Factor and BBC page red. The background of each image page is white.

In order to make the text very clear and readable, I experimented with highlighting the text using mark tag, filling whole sections of the page with light backgrounds, modifying

the background to have lighter areas for text and using different coloured text. I settled with using highlighted text as the formatting on different browsers and window sizes made it difficult to use modified backgrounds and the highlighted text suited the style of my web-app. I used this style of highlighted text in my whole webpage, using only different sizes of text. I decided not to change the font in any parts of the app, as the default font suited the needs of my project.

Figure 5: **Highlighted text**- achieved using mark tags in html. CSS used to set text size.

3 Enhancements

Although my web-app performs the basic function of informing people about programmes and where to watch them, perhaps the most obvious addition to the app would be adding more television shows. With only ten shows currently displayed, many users may find there is not enough content to be useful for more than a few minutes.

A significant improvement I would make to my web-app which could help interest users for longer is adding a more interactive homepage. I would have liked to display an interactive slide-show of images of each tv programme with the user able to navigate back and forward through the range of images. Furthermore I think that a short body of text detailing the purpose of the website, or making the purpose obvious to the user would also have been beneficial to the design. A more advanced alternative to this would have been an interactive TV guide in the form of a table for the most popular channels, as seen on websites such as radiotimes.com [4] and tv-guide.co.uk [5].

A common way of discovering programmes on websites such as the above is the inclusion of a search function so that users can search for all their favourite programmes. I chose not to include a search bar at quite a late stage of the development on the basis that the amount of programmes included is not enough to warrant the use of a search bar. If my website was to include more programmes, a search bar would be an essential feature, greatly improving the user experience and allowing the user to filter the content that they would like.

To improve each of the category pages, a sensible approach would have been to implement a client-side sorting method as seen on amazon and many other websites, where the user can select the sorting method in a drop-down menu and a set of data is sorted client-side. The purpose of this would be to reduce the amount of separate web-pages that need to be loaded, ultimately reducing stress on the server in the case of a website with heavy traffic.

To improve the aesthetics of the web-app, I would make it a priority to choose a strong colour scheme and have a solid idea of the formatting of the web-app and each page very early on in the development. To improve the home screen and category pages, a less bold and striking background is required in order to avoid taking the attention from the real content(the text). The text would not have to be highlighted with an appropriate background, increasing readability and the overall look of each page. A combination of a dark background, such as a grey or navy blue, and a lighter background such as light blue or white would allow the text to remain a solid black and stand out regardless of page position.

Further improving the aesthetics, each of the programme pages would be improved by following the same colour scheme as the rest of the site. Although currently each page has a similar colour to the image shown [6], it does not show consistency between the pages, and can look somewhat random.

Figure 6: **Two of the different coloured page backgrounds**

Finally, a small change that could have made a big difference is using the IMDb rating plugin to show the rating in a recognisable and effective way. I found this plugin at the very end of development, and did not have time to implement it. An example of some of the designs I could have used can be seen below.

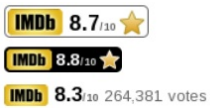

Figure 7: **IMDb plugins**

4 Critical Evaluation

The home page of my web-app remained incomplete for the majority of my project because it was the page I thought would be the most simple and require the least amount of work. I think that the bold colours of the background make the page striking and exciting, as well as being potentially nostalgic and relevant to the topic of the website, however the page may confuse users and make it more difficult to read. This page requires the most improvement to fill the empty spaces and make it more attractive to the user. As the first page the majority of people will see, I do not think this page is entirely effective at grabbing the user's attention or conveying what the web-app is about.

The nav-bar present on many pages is extremely good at helping users navigate between the pages in my web-app. This feature is at the core of the website and I feel I included this in a prime page position. The buttons are simple, to the point, and contain relevant titles- the name of the app and the name of the categories. I am happy with my performance on this part of this element.

The category pages are very simple, however the background of these pages make them surprisingly complicated for the eye. These pages could have been designed with a more subtle background and without the highlighting of text. The links all work and take the user to exactly where they expect, meaning they work well. Similarly, the IMDb link on each of the programme pages also work well. The hyperlink text is 'IMDb Link', making it clear and meaning users are exactly sure of where clicking the link will take them. The only improvement to this feature is opening the IMDb page in a new tab/window, rather than reloading the same page which the user might still be reading.

The informative text I have used is from Wikipedia [6] [7] [8] [9] [10] [11] [12] [13] [14] [15]. The text is obviously informative and relevant to each programme, as it is quoted from a trusted source. The inclusion of well written, informative text is a must-have on a web-app of this nature, and I think that the size of each block of text, text position and content all works well for each page.

The ability to view each image in a separate page is also a feature which works well, however I think this feature would work better if the image page was loaded on the click of the image, rather than the button at the bottom of the page. The button navigation is useful, but looks very poor, and with a little bit more time, could have been refined into a much better feature of the web-app.

Another thing I think was not executed entirely well is the displaying of the channel and show time. This is displayed as plain text below the title and above the picture of each programme. It only details when the current series is on, and only the main channel it is broadcast on. This feature is one of the main parts of my webpage, and it does not stand out to the user as one. In the figure below, you can see my webpage, and what would have been a good example of this feature. The table below is an interactive guide from radiotimes, and is a staple feature of their website.

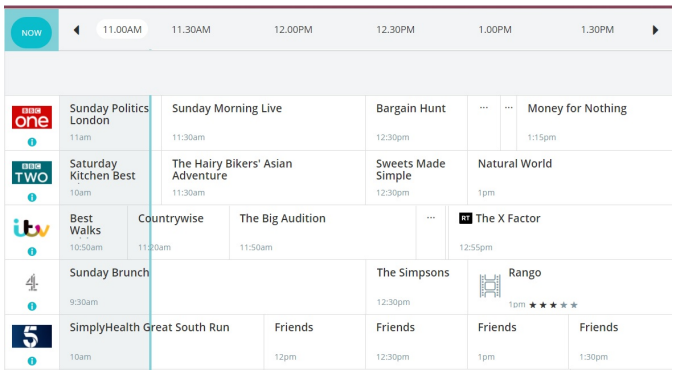

Figure 8: **Radio Times Website**- uses an interactive live tv guide to provide showing times for 70+ channels.

5 Personal Evaluation

Working on TV List-ting was rewarding being able to see the web-app coming together bit by bit very quickly. At the beginning of the project, I spent a considerable amount of time trying to decide an appropriate topic that would allow me to organise the data by several different parameters. This may have lead to a slightly delayed start on the planning of the hierarchy and page contents, leaving less time for development. I spent several days deciding on the programmes url hierarchy and the navigation between each of the pages. This planning proved helpful when inputting data repetitively to the ten programme pages, as well as when correctly mapping the urls to each page.

During the planning stage, I did not consider or test any colour-schemes, page layouts or desired features. Throughout the process and particularly after, I realised that I would have saved a lot of time picking these properties and sticking with them for the whole project. This includes and appropriate background and font-size. Changing these properties multiple times on several occasions required time that could have been used adding more content to my web-app.

Using vim at times proved to be repetitive and infuriating, however I also enjoyed the welcome change from basic text editors I have previously used. I struggled many times to find working shortcuts to paste text from the external clipboard. I also could not find a way to copy text internally for copying html tags and larger repeating parts of code. After searching extensively, I discovered two commands on stackoverflow- one for activating the mouse within vim (set mouse=a), and the other for pasting plain text from the clipboard (Shift-Insert). This allowed me to copy text with the mouse and paste into vim. IF I known these at the start, I could have saved a lot of time during the development. Vim also gets confused when using symbols such as quote marks and apostrophes, as these are used to surround areas of text and variables. Using these in a paragraph already surrounded by the same symbol acts as the end of the boundary, so in order to use the symbol within the variable, using a back slash will stop the symbol from doing this.

Previously I have had no experience with python, and very little experience with HTML and CSS. Not only did I find each of these languages easy to pick up very quickly, but I enjoyed using them to create a personalised and detailed app. Python is good at linking the pages together, adding some of the main content and handling errors and redirects. HTML and CSS were very enjoyable to use because every tiny change can make a huge change on the webpage, so precision and consistency seemed essential.

A problem I ran into regarding html and css is that I did not use a style sheet for the css. At the beginning of my project I was having huge problems applying css to certain aspects of my website such as text and headers. Finding solutions online was easy enough, however many of them included solutions which plugged the bits of css straight into the html, rather than using a style sheet and avoiding huge amounts of repetition. I got into a habit of this and had included a considerable amount of css in html tags before I realised that time and effort could be saved using a style sheet. Researching this further or doing further reading in

the workbook could have avoided this mishap and saved a lot of time.

Not working fast enough was a common problem throughout the project. I started the planning and development with more than enough time to produce an exceptional web-app, however I spent far too much time changing backgrounds, worrying about text placement and margins. I would have been able to add much more content and add more features to TV List-ting if I had the right priorities. In this sense I am disappointed with the final product, because I know it could have been more.

A smaller problem I faced was issues with making my web-app live, as I discovered another person was using my port for some of the time. This meant that when their web-app was live, the port was being used and I could not run my own. I faced challenges with pushing my project to git, as I recieved the error 'Updates were rejected because the remote contains work that you do not have'. This was due to inexperience, and I found the solution as simple as pulling the project, before pushing it once more successfully. I dealt with this problem quickly, and I am starting to understand the basics of using git.

If faced with the same task again, I would research the use of JavaScript and JSON to gain an idea of how I can implement these in my web-app.

Overall I feel my performance has been good, and I have enjoyed most parts of planning and making TV List-ting. I enjoyed making the url structure and having such freedom in content and design however I spent too long on the design and look of my web-app, and not long enough on the content and features.

References

[1] "https://www.bbc.co.uk/iplayer,"

[2] "https://www.channel4.com/on-demand,"

[3] "https://www.imdb.com/,"

[4] "https://www.radiotimes.com/tv/tv-listings/,"

[5] "https://www.tvguide.co.uk/,"

[6] "https://en.wikipedia.org/wiki/big_brother_(uk_tv_series),"

[7] "https://en.wikipedia.org/wiki/the_jeremy_kyle_show,"

[8] "https://en.wikipedia.org/wiki/the_x_factor_(uk_tv_series),"

[9] "https://en.wikipedia.org/wiki/the_apprentice_(uk_tv_series),"

[10] "https://en.wikipedia.org/wiki/casualty_(tv_series),"

[11] "https://en.wikipedia.org/wiki/casualty_(tv_series),"

[12] "https://en.wikipedia.org/wiki/24_hours_in_a%26e,"

[13] "https://en.wikipedia.org/wiki/friends,"

[14] "https://en.wikipedia.org/wiki/game_of_thrones,"

[15] "https://en.wikipedia.org/wiki/breaking_bad,"

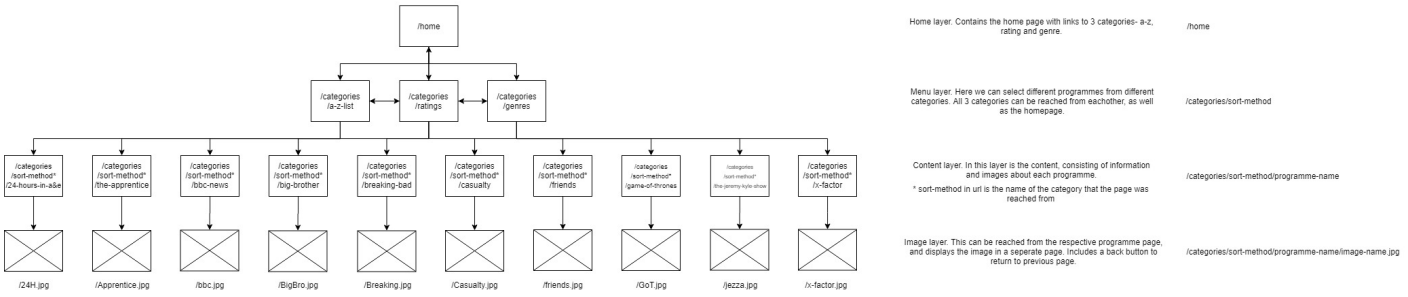

Figure 9: **URL hierarchy**