Coursework 1 - Report

Ivan ROGER
40285021@napier.ac.uk
Edinburgh Napier University - Advanced Web Technologies (SET09103)

Abstract

This is the my report for the first coursework of the Advanced Web Technologies module (code SET09103) at Edinburgh Napier University.
This coursework consists in the creation of a basic web application to serve a catalogue of entries. This application should be developed using Python Flask.

Keywords – Napier, University, Python, Flask, Web app, Report, MetaChar

1 Introduction

The theme of this coursework being free I had to select mine. Here are my choices.

1.1 Subject and Name

As a choice for my subject I chose to create a catalogue of fictional characters that can appear in movies, comics, tv series and other medias. As a consequence the app is named **MetaChar**, meaning that it is a tool to get some informations and meta-data on characters.

This tools also provides informations on the universe of each character. The main two aspects of this application are the characters and the universes they are from.

1.2 Goals

When I thought of this application I had multiple ideas of what I wish this tools would be able to do.

First of all I wanted to give access to the list of universes present in the database. From this list you should be able to get some details about the universe. And to see all the character that are related to it. Similarly for the characters you can get details and see other characters related to it.

On an other hand i wanted also to be able to search through the entries using a search system. It's implementation is detailed bellow.

At last I tried to give the application a specific graphic style that unified through all the pages.

1.3 Universes

A universe is the data structure that represents the englobing container for a world and a collection of characters.

1.4 Characters

2 Design

This section will focus on detailing the way this application is designed and how the features were implemented.

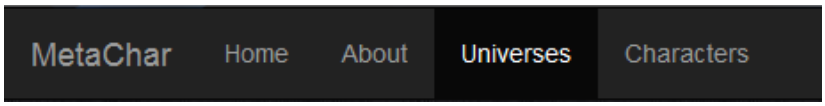

Figure 1: **Navigation** - The navigation links

This figures represents the main pages available.

The first one corresponds to the root of the application. On this page you get general informations on the content of the application. Like the number of entries or other news. (See proposals for enhancements [cf. section 3]).

The second page consists on a page where you can find all the content about the application. For instance information on the creator and links to the documentation and the source code.

The main routes for this application are as following :

/**universes** This route shows the list of universes. (cf. section 2.4)

/**universes**/**<univID>** This URL gives access to a single universe details (cf. section 2.5)

/**universes**/**<univID>**/**characters** Lists all characters in a specific universe.

/**universes**/**<univID>**/**characters**/**<charID>** Similar to the second route this URL gives access to a single character details.

/**characters** Similar to the third route, this one shows the list of all characters regardless of their universe.

The following subsections will explain more deeply how each aspect of the application has been developped.

2.1 Static serving

In order to provide images and other dependancies it has been mandatory to create a specific way to serve files that do not need to be processed on the server.

To do so I used the native static file system provided by *Flask*. Everything to be served staticaly is under a '*static*' directory. Files are ordered by type. Style sheets are under the '*css*' folder, '*img*' contains pictures, '*js*' groups the Javascript needed for Bootstrap as well as '*fonts*'.

2.2 Templates

Each page that can be seen uses a *Jinja* template.

There is a base template that sets up the navigation at the top and the footer of the page.
Each page extends this base template and has a specific content.

Some elements are used on multiple pages. To avoid code duplication I used the macro system that *Jinja* provides. It allows to create a sort of function that returns a specific html content depending on the arguments provided.
Moreover for the character list, two URL use the same template.

Listing 1: The 'data' object

```
1  data = {'config': ..., 'nav': ..., 'active': ..., 'list': ..., 'pages': ..., 'links': ..., 'search': ...}
```

All the data that is needed by the page is passed through a single dictionary called '*data*'.
The '*data*' object has information on the content of the page, the navigation links and the application configuration.

2.3 Storage

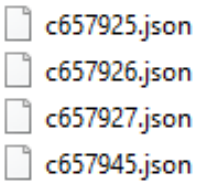

Figure 2: **File storage** - Some files used to store the items

Each item is stored in a plain-text JSON encoded file. All this files are in the '*data*' folder and in a sub-folder depending on whether it is a '*character*' or an '*universe*'.

Folders are automatically scanned at start-up and the minimal content is cached in variables.
The basic data of each item is used the create lists and links to avoid long loading times. When display an entry details we need the full object data so we load the file corresponding to the selected item.

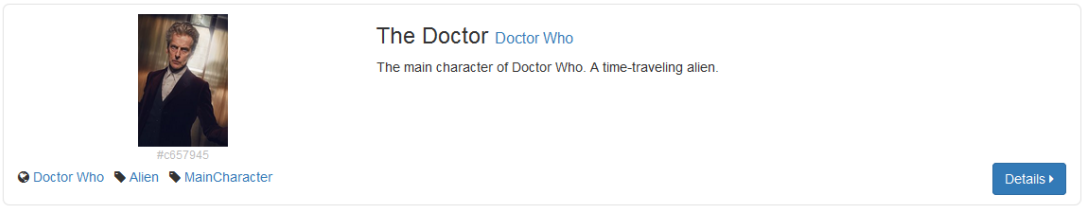

Figure 3: **List entry** - An example of a character entry

2.4 Entry list

The 'Universes' and 'Characters' pages display a list of the entries that are registered in the application.

Each entry is exposed in a card of its minimal information (that are stored in the cache) via a Jinja macro. These informations consists of the picture of the item and it's name, the associated ID and the short description of the entry.

Some tags are associated to each item and can be used to search through the catalogue (cf. section 2.6).
If the entry is a character we can also see the related universe.

2.5 Item details

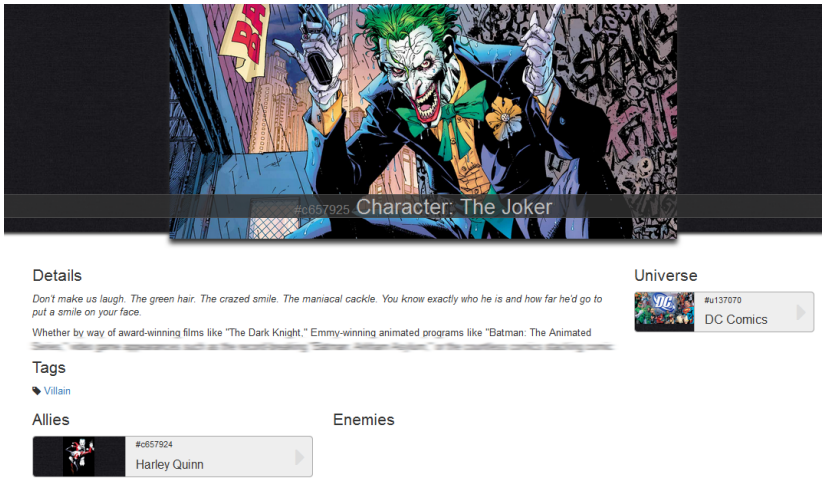

Figure 4: **Entry details** - An example of a character details

Each universe and character can be seen in details

2.6 Search system

2.7 Pagination

2.8 Graphic style

3 Enhancements

Creation and Edition One of the first things that should be implemented if this project was to be really used would be to provide the ability to create new entries in the catalogue and the possibility to edit them afterwards.

Featured items on the home page The home page is currently quite empty but could show a list of the last items that were added, the one that are the most viewed etc ...

More user-friendly search system It is currently not as easy to use the search system that I wanted it to be. For example we could provide an auto-complete search for the tags (in Javascript).

Graphic style re-use In the universes or characters list we should re-use the graphic style of the links in the details page.

4 Evaluation

4.1 Advantages

4.2 Disadvantages

5 Development process

5.1 Difficulties

5.2 Things I learned

6 Ressources

7 Eratum

As an International student I had to pay a specific attention to the redaction of this report. Could you please give me some feedback on the way it is redacted and the weaknesses of my written expression.