Discworld Books App

Christine Griffiths

40402000@live.anpier.ac.uk

Edinburgh Napier university – Advanced Web Technology (SET09103)

Introduction

The Discworld books web-app is an application which allows users to find information about Terry Pratchett's Discworld books, according to their categories, in an intuitive and logical way. The application features a user-centred organisation of the data relating to the series of books; allowing the user to browse the collection of books and its categories in several different ways and without the requirement of instruction in how to use the application. Dynamic content is offered to the user through HTML templates which display the requested content, taken from a static (and easily modifiable) json file. Due to the integration of the Bootstrap framework the application is fully responsive, this combined with a correct HTML mark-up means the application is also cross-device and cross-platform compatible.

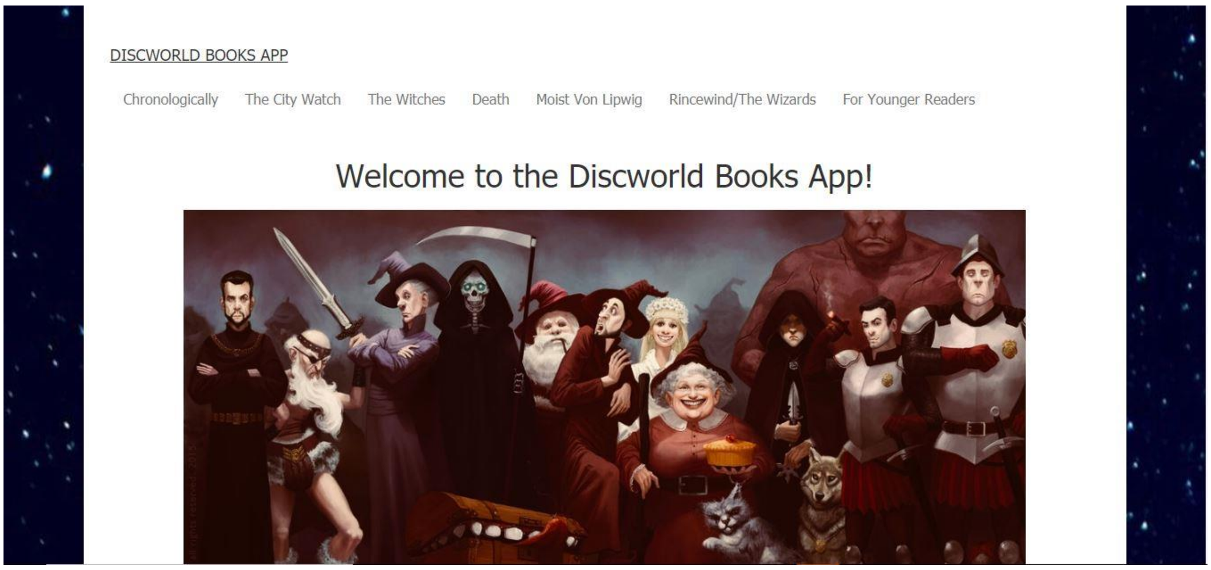
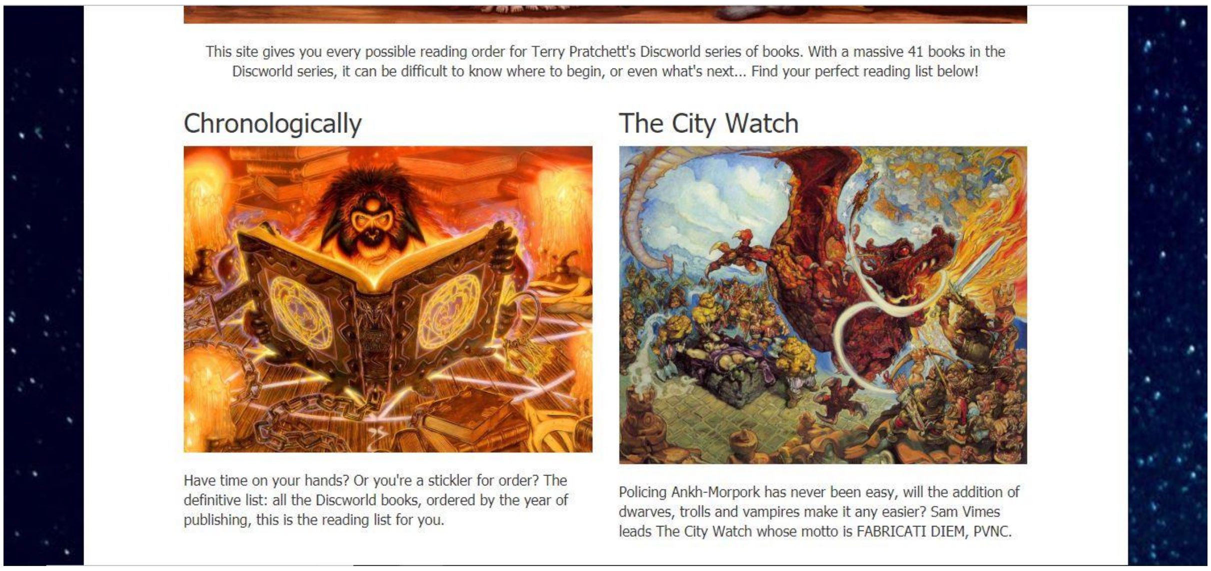


Figure 1 Index page - above fold



This site gives you every possible reading order for Terry Pratchett's Discworld series of books. With a massive 41 books in the Discworld series, it can be difficult to know where to begin, or even what's next... Find your perfect reading list below!

Chronologically

Have time on your hands? Or you're a stickler for order? The definitive list: all the Discworld books, ordered by the year of publishing, this is the reading list for you.

The City Watch

Policing Ankh-Morpork has never been easy, will the addition of dwarves, trolls and vampires make it any easier? Sam Vimes leads The City Watch whose motto is FABRICATI DIEM, PVNC.

Figure 2 Index page - below fold

Design

The design of the web-app's structure was influenced by two factors; the user's needs when using the application and the information to be displayed.

Identifying the user group for this web-app was difficult, the fantasy books appeal to an extremely wide-raging audience and, apart from those written for younger readers, have no specific target audience. As Terry Pratchett himself said: "Fantasy is uni-age. You can start it in the creche, and it follows you to death." (Moss, 2013) Given the conclusion that there was no specific audience for the books, and therefor no specific user group to be identified, but that this series of books could potentially appeal to people of all ages and with hugely varying levels of computing skills, the website was structured simply, information displayed logically and clear and consistent layouts used throughout.

The simple layout of the web-app's pages reflected the need to provide an application suitable for a wide-ranging group of users, elements are grouped into rows and columns using Bootstrap's grid system whilst white space and use of the page hierarchy separates and emphasises different elements. Each page features the same font, use of language, colour scheme and illustrations from the illustrator of the Discworld book series (Kidby, n.d.), together with a similar layout. This consistency follows one of the 10 Usability Heuristics for User Interface Design: Consistency and Standards, "Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions." (Nielsen, 1994) Again, following Neilsen's Usability heuristics (Nielsen, 1994) users are provided with error messages and given an 'emergency exit' with the implementation of error handlers which offer a quick 'back' link to return to the index of the web-app.

When accessing the web-app it was naturally assumed that users wished to gain more information about the Discworld series of books. The structure of the web-app allows users to find the information they are looking for quickly and efficiently, through the clear navigation available or using the correct URL path to access the page they wish to visit. "Well-designed URLs make it easier for users to intuitively find resources" (Smyth, 2018) URL paths were named logically and simply: each of the book categories has a corresponding path as does each individual book title.

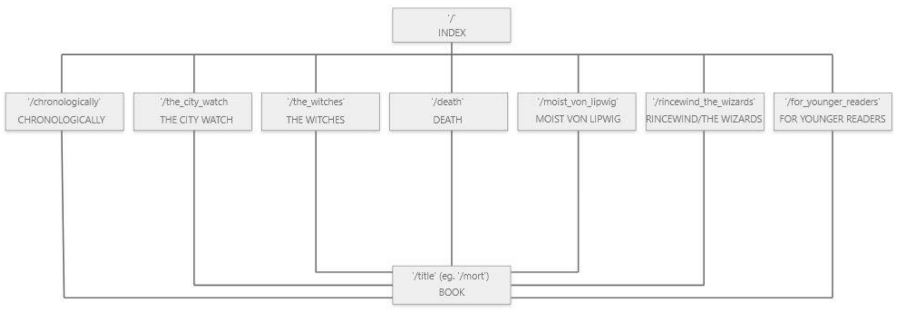


Figure 3 Navigation Map

Structuring the information in a way that would be easy for the user to access was achieved by defining the categories the Discworld books fell into and providing the user the lists of books pertaining to each category. The lists of books are links to each individual book's URL path, which matches the book's title. When on the individual book page of the app the user is presented with several options; they can return to the parent category, use the ever-present navigation bar to navigate to a different category or navigate to a different boook or category page using the URL path.

The information for the categories, lists of books and individual books is generated from a json file, this was used to create dynamic content without using a database and to prevent the need to create around fifty individual HTML pages (one for each URL path). Using a json file gives the project scope for expansion or alteration in the future, as the file can easily be updated or added to.

Enhancements

Ideally the web-app would benefit from the number of HTML templates being reduced, consequently reducing the likelihood of errors and improving consistency, with the implementation of Jinja2's functions to set the active navigation item and blocks to reuse code (Jinja2, 2008) the number of templates could be reduced from ten to just five.

An enhancement to the web-app would be the addition of an option for users to create their own reading lists or categories. User generated content has been shown to increase user engagement and benefit the organisation providing the platform. "Through user engagement, organizations obtain information on user interests that can be leveraged for a variety of reasons (e.g., market research, advertising revenue, new products and/or services, etc.). While for users, an environment is created where users can create personalized experiences that are both meaningful and fulfilling to the user." (Di Gangi, 2009) This feature would most likely be database driven for both a login system and saving the lists/categories to.

Critical Evaluation

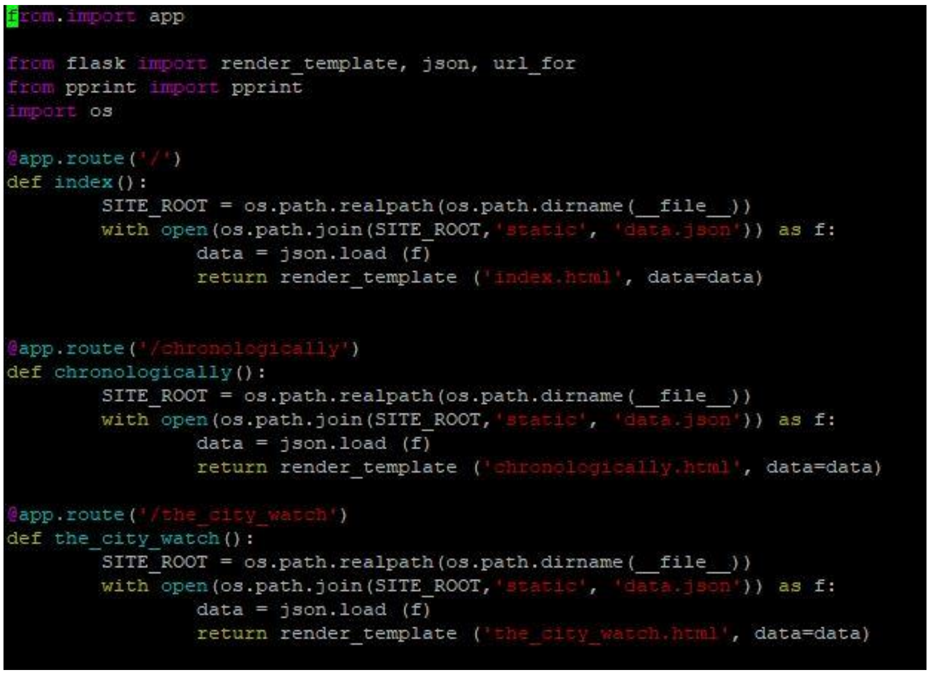
One of the features that works well in the web-app is the dynamically generated content for the individual books page, the same template is used and the content changes dependent on the URL path, which is the book's title.

The URL paths are well designed; using the books' category or title for the URL path means the user can find the information for categories and individual books quickly, easily and intuitively.

The access to the json file and its data could be better organised, each route contains the function for loading the json file and its data, in retrospect it would have been far simpler and better practice to make this a global function as it would reduce the resources being called for each page, consequently reducing the likelihood of errors and simplifying the code.

The error handlers and corresponding error HTML templates were a last-minute edition and lack the depth of content in the other HTML templates. Unit tests were not properly conducted, having written one and finding it very time consuming, this was left to the last minute and never completed as time ran out. This lends a negative and unprofessional impression to the application.

Upon reflection, the categorisation of the collection could be considered a little simple and the addition of the year of publishing and IBSN number to the information about each book provided to the user could be very beneficial.



```
from.import app

from flask import render_template, json, url_for
from pprint import pprint
import os

@app.route('/')
def index():
        SITE_ROOT = os.path.realpath(os.path.dirname(__file__))
        with open(os.path.join(SITE_ROOT,'static', 'data.json')) as f:
                data = json.load (f)
                return render_template ('index.html', data=data)


@app.route('/chronologically')
def chronologically():
        SITE_ROOT = os.path.realpath(os.path.dirname(__file__))
        with open(os.path.join(SITE_ROOT,'static', 'data.json')) as f:
                data = json.load (f)
                return render_template ('chronologically.html', data=data)

@app.route('/the_city_watch')
def the_city_watch():
        SITE_ROOT = os.path.realpath(os.path.dirname(__file__))
        with open(os.path.join(SITE_ROOT,'static', 'data.json')) as f:
                data = json.load (f)
                return render_template ('the_city_watch.html', data=data)
```

Figure 4 Loading json for each route

A better implementation of Jinja2's features would have meant that only one 'category' template was necessary instead of seven. This is not a critical feature but would create a more RESTful API structure, managing the resources of the application in a more efficient way.

The addition of the flask extension 'clacks overhead' which adds the header "X-Clacks-Overhead: GNU Terry Pratchett" (Mayor, 2016) would have been poignant, but seemed inappropriate to a University project, therefore "X-Clacks Overhead" resides in the head of the HTML templates as meta data.

Personal Evaluation

Many new skills were developed throughout the project, most importantly, a better understanding of RESTful APIs. Some basic skills in using the Python language and Flask framework were developed. The management of resources in an API, with the use of Json and the Jinja2 templating system, was another new skill and one that will likely be developed further in future projects. Creating and using a GitHub repository is a very useful new skill and will undoubtedly be of great use in future employment, particularly in collaborative projects. Use of the command shell and VIM as a text editor was a difficult learning process, non-graphical interfaces take some getting used to, but some skills were developed here too.

Time management was a challenge in the completion of this project. The environment for building the web-app (Linux ubuntu) was almost entirely new, as were the tools used (Python, Flask and Json) and therefore a lot of time was consumed in becoming familiar with this new environment, through following tutorials such as the course workbook and online (Kennedy, 2016) (Smyth, 2018). Conversely, a sound understanding of HTML, CSS, the Bootstrap framework, JavaScript and naming conventions for URL paths was incredibly useful and allowed for some of that time to be recuperated. Working to a deadline meant that extra hours of learning had to be put in to meet that deadline, often working late into the night to complete the project, perhaps not the most productive time to work, but with no other option available, this is how the work had to be completed.

Overall the web-app meets the coursework criteria of creating an online directory and a URL hierarchy which makes finding and retrieving information simple and is appropriate to the project. The web-app uses the Python Flask microframework effectively to deliver the appropriate content to the user. The project was successful, but there is certainly scope for improvement.

References

Di Gangi, P. a. (2009). The co-creation of value: Exploring user engagement in user-generated content websites. *Proceedings of JAIS theory development workshop. sprouts: working papers on information systems.* , Vol. 9. No. 50.

Discworld Emporium. (2018). *Books*. Retrieved from Discworld Emporium: https://www.discworldemporium.com/12-books

F., X. F. (2017, 12 21). *How to design a RESTful API architecture from a human-language spec*. Retrieved from O'Reilly Learning: https://www.oreilly.com/learning/how-to-design-a-restful-api-architecture-from-a-human-language-spec

Jinja2. (2008). *Jinja2 Documentation (2.10)*. Retrieved from Jinja 2 Documentation: http://jinja.pocoo.org/docs/2.10/

Kennedy, P. (2016, 6 18). *Creating a Simple Flaks Web Application*. Retrieved from Patrick's Software blog: https://www.patricksoftwareblog.com/tag/tutorial/

Kidby, P. (n.d.). *© Images and Illustrations*. Retrieved from Paul Kidby: https://www.paulkidby.com/

Mayor, W. (2016). *William Mayor/flask-clacks*. Retrieved from Github: https://github.com/WilliamMayor/flask-clacks

Moss, S. (2013, 4 22). *The Guardian - Terry Pratchett, Raising Steam*. Retrieved from www.thegaurdian.com: https://www.theguardian.com/books/2013/apr/22/terry-pratchett-raising-steam

Mozilla. (2018, 1 26). *Python*. Retrieved from MND Web Docs: https://developer.mozilla.org/en-US/docs/Learn/Drafts/Python

Nielsen, J. (1994). Enhancing the Explanitory Power of Usability Heuristics. *CHI '94 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, (pp. 152-158). Boston, Massachusetts, USA.

Oleska, K. (2015). *© Index page main illustration*. Retrieved from Deviant Art: https://www.deviantart.com/katea/art/Terry-Pratchett-book-spine-529272616

Smyth, P. (2018, 4 2). *Creating Web APIs with Python and Flask*. Retrieved from The Programming Historian: https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask

Table of Figures