Coursework 1 Report

Rossa Heraty Quirke
40204617@napier.ac.uk
Edinburgh Napier University - Advanced Web Technologies (SET09103)

Reeling in the Years Website

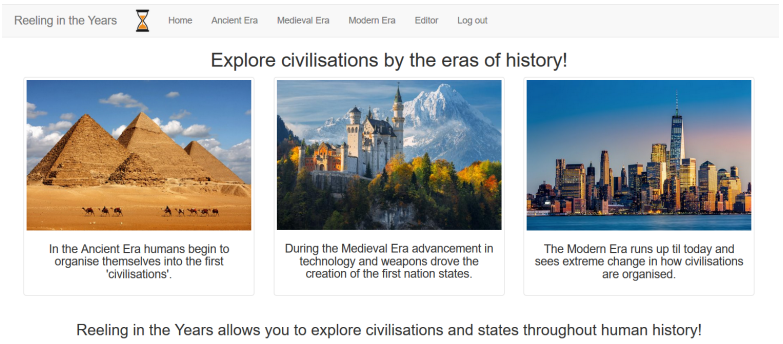


Figure 1: Home page

1 Introduction

'Reeling in the Years' has been built around the concept of exploring a directory of human civilisations/states via the era of history when they existed. After selecting an era and time period, a subset of an era, you can view the directory - displayed as a grid of images and text representing an each individual entry, additionally you can filter by the geographic region, e.g Europe. Entries on civilisations contain metadata such as: religion, government type, year of establishment, time period, region, general information, and much more. The page displaying the entry on the chosen civilisation is also interactive, a user can select either the time period or region field to be brought back to the time period or region that the civilisation belongs to. From the individual civilisation page users can also navigate to indexes of civilisation that share metadata with the civilisation, such as government type and religion - this creates a highly hyperlinked site.

The main page displays clickable images with links to the 3 eras of history: the Ancient Era, the Medieval Era, and the Modern Era. By clicking on any of these the user can then filter and navigate to the entries on civilisations for the chosen era. An era breaks down into a time period and for each time period a page is shown that allows filtering by region allows the user to click on an interactive map of the world to filter and see entries on civilisations just in that region, or click a button to see all civilisations (see A). The index page which includes all entries on civilisations is a grid of images with a carousel at the top of the page, the carousel flips through 3 randomly chosen unique entries from all the entries shown on that page (see Appendix B).

All pages have a navigation bar which can also be used to return home, or to select any of the 3 eras mentioned before. The navigation bar also includes links to the administrator login page, if the user has logged in then the navigation bar will instead present a link to log out. The navigation bar also links to the editor. In the editor a user can go to a page to add a new entry on a civilisation or another page to remove an existing entry, adding an entry also requires an accompanying image to be uploaded. To access the editor page the user must be logged in and the site will redirect the user to the login page if attempts are made to access the editor or its sub-pages to add and remove entries.

2 Design

The site divides entries on civilisations into 2 main levels of abstraction. These are era and time period. I designed it in this way in order to break down a large directory and to provide a novel way for a user to navigate and explore the site. Each level of abstraction has a dynamic web page associated with it, the information and hyperlinks on each of these pages is totally dynamic - I utilised json files to accomplish this. There is a single json file that holds all the information for each of the eras and the time periods that they contain. Then to mirror the levels of abstractions I created a corresponding file structure and folder containing a json, the file structure is $'/era/time-period/civilisations.json'$, for example all the entries for civilisations of the 'Bronze Age' during the 'Ancient Era' are held in the file with the path $'/ancient-era/bronze-age/civilisations.json'$. The application knows which json to read as it takes parses the url as separate variables and uses these variables to construct a path to the desired file, for instance when the Early Middle Ages Civilisation index page with the url: $'/medieval-era/early-middle-ages/'$ is visited the application generates the path: $'medieval-era/early-middle-ages/civilisations.json'$ and so only parses entries from the json that has entries in the Early Middle Ages. This has a number of advantages. It saves the trouble of filtering out large numbers of entries that are not needed, therefore speeding up the site. Another benefit of having this file structure is that when adding a new entry to the json the process is again not reading a large amount of data that is unnecessary. This structure also makes expanding and scaling the site easy, simply creating the folder and editing a json will add a whole new time period or era. In order to parse the jsons I created a sep-

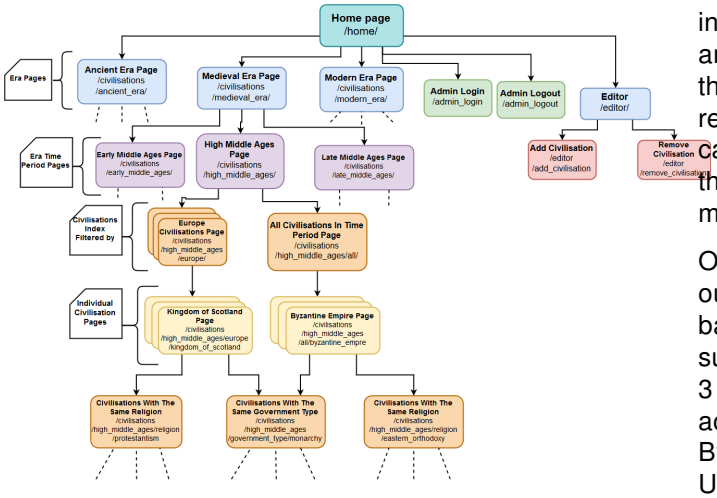


Figure 2: URL Hierarchy

arate class to contain all this logic, in line with the single responsibility principle. This class reads either the json file with all the era information or one of the civilisations jsons and then converts them to objects of the classes Era and Civilisation respectively.

The Era class includes the name of the era, a short bit of information about it, the name of the image associated with it, and a list of time periods. Through this all the links to the eras and time periods are created dynamically. The Civilisation class contains a lot of information mostly used for display on the individual civilisation page but also contains the region field which can be used to filter entries within a era and time period. The Era class covers all the levels of abstraction for navigating the site while the Civilisation class contains all information needed for displaying entries, filtering them, and creating a hyperlinked document. I decided to parse the jsons into objects as objects provide better encapsulation, are easier to manage and validate than dictionaries, and provide more opportunities for expansion. Additionally I find taking an object-oriented approach to this makes the code easier to and more logical in the sense that items in the application space are all represented as objects within the code, this also follows core object-oriented principles such as the single responsibility principle and open/closed principle. There is a separate class that manages reading all the json data and parsing what is read into an instance of the correct class.

The site has an editor suite has two major features, to add a new civilisation and to remove an existing one. A user must first be logged in to access either of these pages, as well as the editor page which links to both the add and remove pages. This was implemented using flask sessions, if the session for allowing editor access has not been set or has the wrong value then the user will be redirected to the login page when trying to access any editor related page. The add and remove pages are presented as forms for the user to fill out(see Appendix C and D). All fields for adding or removing a civilisation are required and the form will not send the post request if they are not filled out. The add form sends a post request with the entered data and writes to the json based on the era and time period in the request. Similarly the remove uses the time period and era to read the correct json file and then removes the chosen entry. To keep low level operations such as reading and writing to json separate from the core application and routing logic I created a manager class to hold these functions, it also contains other functions related to managing the collection of civilisations.

Overall the website has a clear hierarchical structure as outlined in the URL hierarchy diagram, with the navigation bar allowing a user to branch out to all top level pages such as the editor page, login page, home page, and the 3 era pages. Most of the pages within the site can be accessed from each other making it extremely connected. By dividing the data via a file structure that mirrors the URL hierarchy of the site there is a consistent definition of abstraction between the application and code itself. It also keeps in line with the theme of the site - exploring civilisations through history. And through following object-oriented principles the code should be easy to expand upon and maintain.

3 Enhancements

3.1 Edit Civilisations Feature

There are 3 features that I think would improve the site quality. The first is an addition to the existing editor features to edit a selected entry on a civilisation. Ideally the user, once logged in, would see an edit button when viewing an entry and could then click it to make the entry editable. The current work around to achieve this functionality is to remove an entry then add it via the add and remove features, this is less than ideal as the admin may want to simply correct a small mistake or edit a single field.

3.2 Predecessor and Successor Civilisation Navigation

The 2nd enhancement would be an enhancement to the site navigability. This would take the shape of an extension of the civilisation class adding the civilisation that preceded it and the civilisation that succeeded it. For example the Roman Republic would be preceded by the Roman Kingdom and followed by the Roman Empire. This would allow a user to follow a civilisation through its various incarnations throughout history. I feel this would provide a more interactive experience for the user and fits closely with the theme of the site.

3.3 Search Functionality

The final extension I would add is search functionality. A user would be able to select an era and time period and then search by name of civilisation. This would allow a user to quickly find exactly what they are looking for. Though the site is themed around exploration it's possible that after repeat viewing a user could become frustrated having to navigate through at least 2 layers of the site before seeing any entries on civilisations, especially when they are looking for a specificbentry.

4 Critical Evaluation

Overall I feel my website has a good range of functionality and interactive ways for the user to explore its directory of civilisations. The navigation of the site is clear and consistent, using mostly hyperlinked images with titles or captions allowing to quickly reach a set of entries for a time period or region. Each entry in the directory is hyper connected to other entries and a user is able to explore virtually every other entry in a section through clicking links on an entry's individual page. The website is mostly minimalist but provides a large range of good quality images, making good use of static Flask resources. Pages are all dynamic, through use of Jinja2 templates, so there is a low number of actual html pages to maintain and edit. The URL hierarchy is also clearly defined and a user can easily guess what they need to enter to get to a specific page.

5 Personal Evaluation

Reflecting on how I approached this project I am pleased with the concept and theme that I thought up. I think it allowed me a lot of creative options to display entries of my directory of civilisations and to create many interesting ways to navigate the site. I feel content with the overall quality of the website and the code behind it. At the start of the project I could have set out a clearer plan or structure of work, this would have helped me to not get side tracked while working on less important parts of the side - for example I believe I spent more time than necessary in formatting and beautifying pages when I could have been implementing additional functionality. Though considering this has been my first notable website I am, on the whole, pleased with its quality and design.

6 References

https://linuxconfig.org/how-to-parse-data-from-json-into-python

https://docs.python.org/3/tutorial/classes.html

https://realpython.com/primer-on-jinja-templating/#quick-examples

http://flask.pocoo.org/docs/0.12/templating/

https://www.w3schools.com/html/html_images.asp

https://www.w3schools.com/html/html_formatting.asp

(Bootstrap Carousel used in site heavily sourced from this) https://www.w3schools.com/bootstrap/bootstrap_carousel.asp

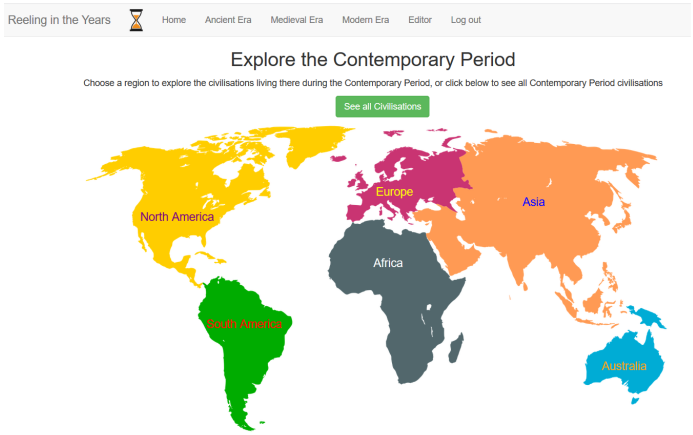
https://bootsnipp.com/snippets/featured/thumbnail-gallery-modal-view-and-key-navigation

https://stackoverflow.com/questions/33241050/trailing-slash-triggers-404-in-flask-path-rule

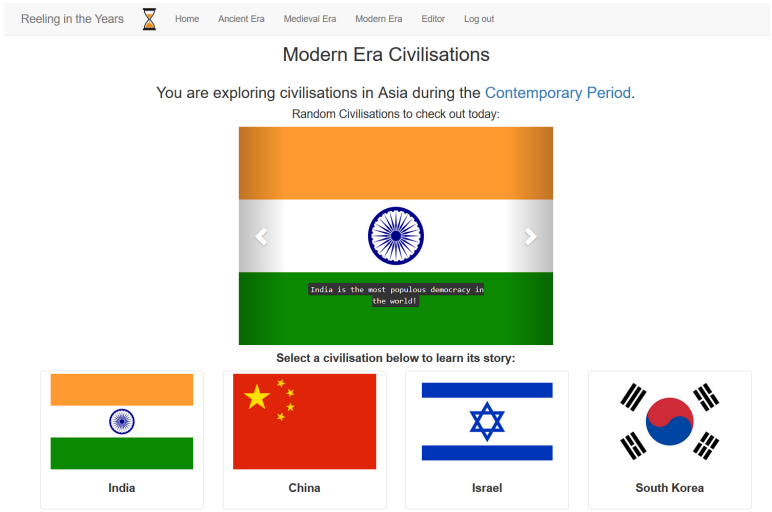
https://stackoverflow.com/questions/17057191/redirect-while-passing-arguments

Appendices

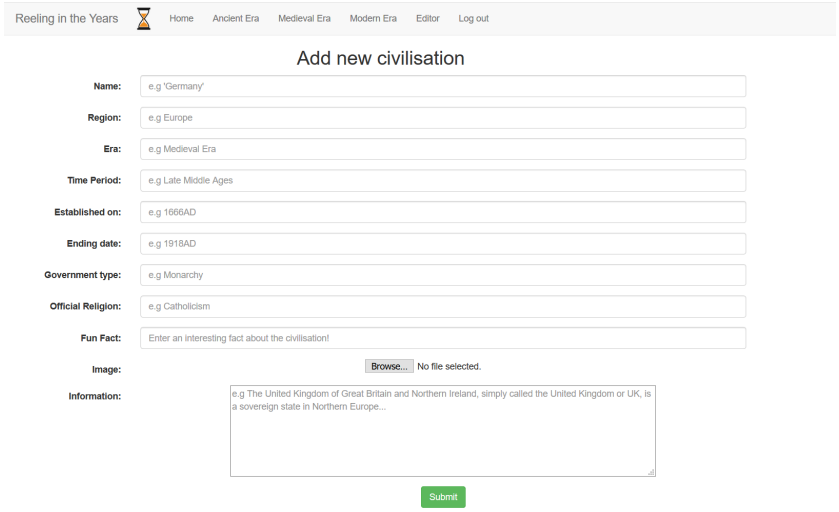
A Appendix A



B Appendix B



C Appendix C



D Appendix D