Note to the Marker

Please consider the following candidate's written work in respect of specific learning difficulties

AB20712
(Candidate Number)

Status of Application:
The Personalised Examination Provisions Committee (PEPC) has approved the use of a coursework cover note in respect of specific learning difficulties

For The Marking of Work for Language Modules
● In the case of language modules, please consider carefully what specifically is being assessed. If it is purely the accuracy of the use of the language (e.g. the spelling, grammar and punctuation) that is being marked, then exceptions should NOT be made for specific learning difficulties. However, if the paper is being assessed either fully or partly on other criteria, then please follow the guidance described below as appropriate.
Guidelines for Markers Assessing Coursework of Students with Specific Learning Difficulties
● Although students with specific learning difficulties should be marked with regards to the elements of content in their work, the logical argument of an essay or report may not be constructed in a very sequential manner. In addition, grammatical and sentence structure errors can be missed by the student.
● Students with specific learning difficulties commonly make errors in the form of the omission of small function words, the addition or repetition of such words, the transposition of words and the substitution of other function words (the /a / an / for / from). Such errors should be disregarded.
● With the exception of essential technical vocabulary, poor spelling, grammar and punctuation should be disregarded.
● The use of spell- and grammar-checkers has a limited use for students with specific learning difficulties. Word substitution, phonetic equivalent and American spelling errors can occur.
● A summary of approaches for markers is given below (from "Guidelines for Examiners" document available from the Examinations and Awards Office)

○ Read passage quickly for content ○ Include positive/constructive comments ○ Use clear English ○ Use non-red coloured pens for comments ○ In correcting English, explain what is wrong and give examples	
Academic year:	**2020/1**
Alteration or misuse of this document in any way will result in referral to the Disciplinary Committee	

KING'S College LONDON

Candidate number: AB20712

Module code: 7AAVDM14

Album Covers

Index

Introduction

The website created[1] for this project aims at cataloging the covers of a large number of music albums by creating a user navigation based on the selection of colors. In fact, each album has been accompanied by a variable number of metadata that have been extracted by an algorithm that in turn has scanned a variable number of colors within the cover of the various albums. In this way the user, thanks to the dedicated navigation interface, is able to select macro-categories of color in order to display the corresponding albums.
The color is not the only way to access a certain element, there are also a textual search and a graph that allows you to navigate through the various genres belonging to the music albums highlighting how they are related to each other.
My role within the group (consisting of 2 people) was to deal mostly with the technical choices related to the structure and internal functioning of the website while the role of my colleague was directed more towards the artistic component of the project even if it is appropriate to clarify that these two tasks are reversed several times during the realization.

Realization

Input data

The album data was taken from the API[2] of the famous Swedish music service Spotify[3]. These contain information related to genre, artist, year and numerous other information inherent to many music albums. Using a color classification algorithm[4], a series of macro categories based on colors were identified for each album. These were then assigned to each of them with the result that each album had a metadata indicating the presence of one or more macro colors[5] that are part of its cover. This dataset was then

---

[1] Github repository*, final-assessment-group-3*, accessed December 24, 2020, https://github.com/kclddh-7aavdm14-2020/final-assessment-group-3

[2] Spotify API documentation, *Web API Libraries*, accessed December 24, 2020, https://developer.spotify.com/documentation/web-api/libraries/

[3] Spotify official website, *Spotify*, accessed December 24, 2020, https://www.spotify.com/

[4] Classification algorithm, *analysis_music* ,accessed December 27, 2020, https://github.com/Sblbl/analysis_music/blob/main/assign_colour_labels.ipynb

[5] Netlify project page, *Album Covers*, accessed December 24, 2020, https://infallible-thompson-c65378.netlify.app/labels.html

transformed, thanks to a script purposely realized by my colleague, into a series of files in "md"[6] format, one for each album, in order to integrate them into the website.

Structure of the Website

The website created consists of four main pages that will be described here in the following sections.

Album Covers

This page, which is also the home page of the site, offers the user a three-dimensional navigation experience where it is possible to select the cube of the color corresponding to the colors of the album cover you want to visualize. This page has been realized thanks to the Threejs library[7], a library written in JavaScript which allows the developer to manage three-dimensional polygons in a simple and fast way.

Albums

This page shows, in sections of ten results at a time, previews of the pages dedicated to the individual albums. These have been generated from markdown files and displayed using the AutoPages library[8].

Genres

Realized in part in a similar way to the page just described, using the AutoPages library, this page allows you to access the sections dedicated to individual albums using the genres of the same as a navigation medium. Thanks to the JavaScript library D3.js[9] it is possible to select the genre we are interested in and to see how it is related to the others because the links of the genres that concur in the same album are highlighted.

Labels

In this page it is possible to search for albums by choosing in a discreet way the colors of interest through checkboxes. This part of the site will be described in detail in the next section where all the related design choices and problems encountered during the implementation will be outlined.

---

[6] Markdown format, *Basic Syntax*, accessed December 24, 2020, https://www.markdownguide.org/basic-syntax/

[7] threejs Javascript library, *threejs.org*, accessed December 24, 2020, https://threejs.org/

[8] Jekyll Autopages plugin, *Jekyll::Paginate V2::AutoPages*, accessed December 24, 2020, https://github.com/sverrirs/jekyll-paginate-v2/blob/master/README-AUTOPAGES.md

[9] d3js Javascript library, *Data-Driven Documents*, accessed December 24, 2020, https://d3js.org/

Technical choices and issues

The "Labels" page

Among the numerous problems encountered during the realization of the site, the one described in this section was certainly one of the most challenging to solve and required the constant collaboration of both candidates. The "Labels" page, even if apparently equal to the others, has a very different internal functioning. It uses JavaScript and dynamically generates the results starting from the input of the user who can select the checkboxes corresponding to the colors with which the albums have been encoded within the site. We chose to use JavaScript instead of Liquid because Liquid does not support two different settings as results per page in the "pagination" field expressed in the "config.yml" file. In fact, in order to better manage the results per page in the sections of the site "Genres" and "Albums" the limit has been set to 10 and it would not have been possible to perform this kind of search using the single labels. The only possibility would have been to generate in advance all the pages of the search results corresponding to every possible choice of colors by the user, but this would have invalidated the structure of the site as there would have been too high a dimension of pages.

The solution found was to add a dataset in "JSON" format containing the same information that generated the files in markdown format so that thanks to JavaScript we could access that file and generate results dynamically for each search by the user.

Character encoding

The website was created from two computers, one running Windows and the other running macOS. This, especially in the early stages, has generated problems inherent in the compatibility of the character encoding used. In fact, these two operating systems have different interpretations, rules and tolerances with respect to how the characters are managed within the files. For example, Windows, unlike macOS, does not tolerate the character ":" as a file name.
The most relevant problem was the one encountered with the Ruby programming language that during the initial phases of the project had generated this error that prevented from being able to start Jekyll:

```
"C:/Ruby26-x64/lib/ruby/gems/2.6.0/gems/liquid-
4.0.3/lib/liquid/block_body.rb:97:in `join': incompatible character
encodings: Windows-1252 and UTF-8 (Encoding::CompatibilityError)"
```

After several attempts, the problem was solved by forcing the character mapping of Ruby's output to UTF-8 within the system files that govern the operation of the programming language in object.

Conclusion

The website created does not take inspiration from any other website previously created. We tried to make a virtual museum that would offer the most original user experience possible by focusing on navigability and interaction instead of the actual content. It turns out to be very different from the sites consulted during the course expeditions for this exact reason. The various modalities of research inside the site have been realized starting from the potentialities offered by the various technologies of data visualization like D3.js and Threejs.
The theme chosen obviously reflects what is the passion of both candidates that is to listen to music albums and this was a way for us to pay tribute to the visual component of the latter.

Bibliography

Github repository, *final-assessment-group-3*, accessed December 24, 2020,
https://github.com/kclddh-7aavdm14-2020/final-assessment-group-3

Spotify API documentation, *Web API Libraries*, accessed December 24, 2020,
https://developer.spotify.com/documentation/web-api/libraries/

Spotify official website, *Spotify*, accessed December 24, 2020,
https://www.spotify.com/

Classification algorithm, *analysis_music*, accessed December 27, 2020,
https://github.com/Sblbl/analysis_music/blob/main/assign_colour_labels.ipynb

Netlify project page, *Album Covers*, accessed December 24, 2020,
https://infallible-thompson-c65378.netlify.app/labels.html

Netlify project page, *Album Covers*, accessed December 24, 2020,
https://infallible-thompson-c65378.netlify.app/labels.html

Markdown format, *Basic Syntax*, accessed December 24, 2020,
https://www.markdownguide.org/basic-syntax/

threejs Javascript library, *threejs.org*, accessed December 24, 2020,
https://threejs.org/

Jekyll Autopages plugin, *Jekyll::Paginate V2::AutoPages*, accessed December 24, 2020,
https://github.com/sverrirs/jekyll-paginate-v2/blob/master/README-AUTOPAGES.md

d3js Javascript library, *Data-Driven Documents*, accessed December 24, 2020,
https://d3js.org/