Documentazione progetto Basi di Dati 2: WorldBookShelf

# Fase preliminare - dataset

Nella fase preliminare vengono utilizzati diversi script python per estrarre i dati rilevanti da vari dataset, ognuno con colonne diverse, e mapparli alle variabili della classe Book (file book.py). Questo in modo da creare un unico formato di dati che rappresenta i libri, indipendentemente dalle differenze tra i dataset originali.

Dataset utilizzati:

* <https://www.kaggle.com/datasets/bahramjannesarr/goodreads-book-datasets-10m>

Vari dataset contenenti informazioni su rating libri. Questi dataset sono stati utilizzati come base di partenza a cui sono state aggiunte altre informazioni prese dai dataset successivi.

* <https://www.kaggle.com/datasets/dk123891/books-dataset-goodreadsmay-2024>
* <https://www.kaggle.com/datasets/sahilkirpekar/goodreads10k-dataset-cleaned>
* <https://github.com/zygmuntz/goodbooks-10k>
* <https://www.kaggle.com/datasets/ishikajohari/best-books-10k-multi-genre-data>
* <https://www.kaggle.com/datasets/thedevastator/comprehensive-overview-of-52478-goodreads-best-b>

I vari script utilizzati in sequenza sono:

1. **save\_books.py**

In questo file attraverso la funzione *get\_books\_from\_csv(…)*, a cui vengono passati come parametri il percorso e i nomi delle colonne del dataset di nostro interesse, vengono letti i valori e costruito un dizionario di libri che ha come chiave l’id del libro e come valore l’oggetto Book costruito con i valori estratti dal dataset.

Viene costruito un dizionario di libri per ogni dataset utilizzato, e successivamente questi vengono serializzati in modo da avere direttamente i dati estratti per gli script successivi e non dover rileggere e rielaborare i dataset più volte.

1. **books\_to\_load.py**

In questo file al dizionario di libri relativo ai dataset scaricati dal primo link, vengono aggiunte informazioni come il genere, la descrizione e l’url della cover del libro, etc.. prese dai dizionari relativi agli altri dataset.

Da questo dizionario vengono filtrati in un altro dizionario “*books\_to\_load*” solo i libri che hanno i generi, scartando quelli che non li hanno.

Successivamente di quest’ultimi vengono eliminati anche i libri che non hanno descrizione e che non hanno editore (publisher).

Successivamente, essendo che le informazioni sulla lingua vengono prese da più dataset, con la conseguenza che ogni dataset utilizza un modo diverso per rappresentare la lingua, ad esempio un dataset utilizza “en”, un altro “English” etc.. , si è usata la funzione *normalize\_languages(…)* per uniformare i nomi delle lingue, in modo che siano rappresentate in modo coerente.

A questo punto, il dizionario di libri risultante viene serializzato.

1. **export\_books\_db,py**

In questo file viene deserializzato il dizionario serializzato nello script precedente e per ogni libro in esso presente, i generi che erano rappresentati da un’unica stringa vengono trasformati in un array di stringhe, e il numero di stelle viene convertito nell’intero corrispondente. Dopo viene caricato sul database attraverso una richiesta HTTP POST al server locale all’endpoint specificato:

*response = requests.post('http://localhost:4000/api/book/addBook', json=new\_document)*

# WebApplication

1. **backend**

è stato utilizzato **Node.js** con **Express** e **MongoDB** come database. In particolare, è stato utilizzato **Mongoose**, una libreria Node.js per l’interazione con MongoDB.

I vari modelli (o schema?) utilizzati sono:

* + ***Book*** che rappresenta le informazioni di un libro
  + ***User*** che rappresenta le informazioni di un utente
  + ***Review*** che rappresenta la recensione lasciata da un utente ad un libro
  + ***Shelf*** che rappresenta l’insieme di libri salvati da un utente

Operazioni:

* + Registrazione al sito
  + Login al sito
  + Modifica di e-mail e password
  + Aggiunta di un libro alla propria libreria
  + Rimozione di un libro dalla propria libreria
  + Aggiunta di una valutazione ad un libro
  + Rimozione di una valutazione ad un libro
  + Ricerca libro attraverso titolo [possibilità di ordinare i libri per titolo, autore, numero di pagine,anno pubblicazione]
  + Aggiunta libri al database
  + Lista dei 5 libri con valutazione media più alta
  + Lista libri presenti nelle librerie dell’utente
  + Lista dei 10 libri con valutazione media più alta che abbiano come genere, i generi facenti parte della libreria dell’utente [se l’utente ha almeno un libro nella libreria]

1. **frontend**

è stato utilizzato **React.js** e altre librerie come **Bootstrap**, **MUI** e **FontAwesome** per realizzare la parte grafica della WebApplication. Le richieste dirette al backend vengono effettuate utilizzando **axios**.

Contenuto pagine:

* Nella *pagina di login* un utente può loggarsi alla piattaforma
* Nella *pagina di registrazione* un utente può registrarsi alla piattaforma
* Nella *home page di un utente non loggato* vengono visualizzati i 5 libri che hanno la valutazione media più alta.
* Nella *home page di un utente loggato* vengono visualizzati i 10 libri consigliati
* Nella *pagina Account* vengono visualizzate le informazioni dell’utente come: username, nome, cognome. Vengono poi visualizzate le due librerie dell’utente: “want to read” e “read”. È possibile, inoltre, in questa pagina e-mail e password dell’utente.
* Nella *pagina relativa ad un libro* vengono visualizzate le informazioni del libro, come: titolo, autore, immagine di copertina, descrizione, generi, anno pubblicazione, publisher, lingua, numero di pagine, la media delle valutazioni, numero di stelle. Un utente in questa pagina può lasciare una valutazione al libro e/o salvarlo in una delle sue due librerie.
* Nella *pagina di ricerca* di un libro vengono visualizzati i libri