BOOK RECOMMENDER

Manuale tecnico

Università degli Studi dell'Insubria – Laurea Triennale in Informatica

Progetto Laboratorio A: Book Recommender

Sviluppato da: Giulia Kalemi, m.756143, Chiara Leone m.759095

Introduzione

BookRecommender è un progetto sviluppato nell'ambito del progetto di Laboratorio A per il corso di informatica dell'Università degli Studi dell'Insubria.

Il progetto è sviluppato in Java21, usa un interfaccia grafica realizzata con OpenJFx 22.02 ed è stato sviluppato sul sistema operativo Windows 11

Strumenti e ibrerie esterne utilizzate

- **OpenJFx 22.02**
  La libreria OpenJFx 22.02 contiene tutti gli elementi per lo sviluppo dell'interfaccia grafica

- **Maven**

Maven è utilizzato nel progetto per gestire le dipendenze, semplificare il processo di build e distribuire il progetto

Struttura dell'Applicazione

L'applicazione è basata su un'architettura modulare, dove ogni classe ha una responsabilità unica.

Le core classes:

- **`Ricerca`**: Si occupa principalmente di gestire le operazioni di ricerca relative ai libri nel progetto. La sua funzione principale è quella di permettere la ricerca di libri e di effettuare altre operazioni legate alla gestione dei dati, come il controllo della disponibilità di un titolo nel sistema.

- **`Valutazione`**: Essa consente agli utenti di inserire una valutazione per un libro, specificando vari criteri, come stile, contenuto, gradevolezza, originalità, ed edizione.

- **`Librerie`**: Permette la gestione dei libri, come la selezione del libro da valutare, la registrazione e la gestione dei dati relativi ai libri.

- **`Visualizza`**: Responsabile della visualizzazione dei dettagli di un libro, delle valutazioni degli utenti e dei suggerimenti per altri libri.

Successivamente, ci sono le classi per la gestione dell'interfaccia grafica:

Classi Controller

MainController

HomeController

Home2Controller

TrovatoController

LibroController

LoginController

RegController

ARController

LibController

ValutaController

SuggController

NuovaLibController

NuovaLib2Controller

Nel dettaglio verranno presentate le classi core con tutte le scelte algoritmiche e le strutture dati utilizzate e i pattern file CSV.

Classi Core

1.BookRecommender

2.MainStart

3.Ricerca

- La classe **`Ricerca`** è incaricata di effettuare ricerche su un file CSV contenente informazioni sui libri.
- Le ricerche possono essere fatte per titolo, autore o combinazione di autore e anno.
- Ogni metodo di ricerca restituisce una lista di titoli di libri che soddisfano i criteri di ricerca.

Strutture Dati

- **`List<String> result = new ArrayList<>();`**

Per memorizzare i risultati della ricerca, ossia i titoli dei libri che soddisfano i criteri di ricerca.

- **`String[] tipo = riga.split(",");`**

Per suddividere ogni riga del file CSV in campi separati da virgole.

Dettagli dei metodi

- **`List<String> cercaTitolo(String ricerca)`**

Descrizione: Questo metodo effettua una ricerca dei libri in base al titolo. Accetta come input una stringa e restituisce una lista di titoli di libri che contengono la stringa di ricerca.

⇨**Effettua una ricerca lineare su un file CSV leggendo ogni riga e verificando se il titolo contiene la stringa di ricerca.**

- **`List<String> cercaAutore(String ricerca)`**

Descrizione; Questo metodo effettua una ricerca dei libri in base all'autore. Accetta come input una stringa e restituisce una lista di titoli di libri che contengono il nome dell'autore.

- **`List<String> cercaAutoreAnno(String ricerca, String year)`**

Descrizione:Questo metodo effettua una ricerca dei libri in base sia all'autore che all'anno di pubblicazione.
-Restituisce una lista di titoli che corrispondono ai criteri specificati.

Pattern file CSV
I dati sono contenuti all'interno di un file CSV denominato **`Libri.dati.csv`**
Il file è strutturato con cinque campi separati da virgola:

1. Titolo: Il titolo del libro.
2. Autore: Il nome dell'autore.
3. Anno: L'anno di pubblicazione.

HomeController
Le tre funzioni della classe **`Ricerca`** vengono richiamate all'interno del metodo

`void cercaLibro (ActionEvent event) throws IOException`

contenuto nella classe controller **`HomeController`** (per la schermata HomePage.fxml). Tramite uno switch viene selezionato il metodo da utilizzare.

4.RegController

- La classe **`RegController`**, nonostante faccia faccia parte delle classi controller, si occupa di gestire la registrazione e l'accesso all'area riservata degli utenti (e per questo viene inserita anche tra le classi core).

La classe include due metodi principali:

1. **`registrazione():`** registra i dati di un utente in un file CSV.
2. **`apriAreaRiservata():`** Apre la schermata dell'area riservata per l'utente registrato.

Dettagli dei metodi

- `void` **`registrazione(ActionEvent event)`**

Descrizione:
-Raccoglie i dati dell'utente tramite input da GUI:

- Nome, cognome, codice fiscale, email, UserID e password.

-Combina questi dati in una stringa formattata separata da virgole.
-Scrive questa stringa in un file CSV denominato **`UtentiRegistrati.dati.csv`**, aggiungendola in coda al file.

- **`login(String userid, String password)`**

Descrizione:
-Legge il file CSV riga per riga.
-Divide ciascuna riga nei suoi campi.
-Confronta i campi userid e password con i dati forniti

Pattern file CSV

I dati sono contenuti all'interno di un file CSV denominato **`UtentiRegistrati.dati.CSV`**
Il file è strutturato con cinque campi separati da virgola:

1. Nome e Cognome (Stringa)
2. Codice Fiscale (Stringa)
3. Email (Stringa)
4. UserID (Stringa)
5. Password (Stringa)

5.Visualizza

- La classe **`Visualizza`** permette di visualizzare informazioni dettagliate su un libro specifico, incluse le sue valutazioni e i suggerimenti correlati, leggendo i dati da file CSV.

Strutture dati

- **`int[] val = new int[7]:`**

Utilizzato per memorizzare i punteggi medi per diversi criteri di valutazione e il numero totale di recensioni.

- **`List<String> sugg = new ArrayList<>();`**

Utilizzato per memorizzare i suggerimenti correlati a un libro specifico.

Dettagli dei metodi

1.int[] recapVal(String titolo)

Descrizione: Calcola la media delle valutazioni per ciascun criterio e il numero totale di recensioni per il libro specificato.

2.List<String> recapSugg(String titolo)

Descrizione:Restituisce una lista di suggerimenti correlati al libro specificato e il numero totale di utenti che hanno suggerito il libro.

3.String[] infoLibro(String titolo)

Descrizione:Visualizza le informazioni principali del libro, incluse le valutazioni e i suggerimenti.

4.List<String> note (String titolo)

Descrizione: Restituisce una lista di note fornite dagli utenti per un libro specificato.

Pattern file CSV

In questa classe utilizziamo due file CSV che non abbiamo ancora visto:

- **`ValutazioniLibri.dati.csv`** che ha 7 campi separati da virgola e contiene le valutazioni dei libri

1. Titolo
2. Stile
3. Contenuto
4. Gradevolezza
5. Originalità
6. Edizione
7. MediaTotale

- **`ConsigliLibri.dati.csv`**: che ha 2 campi separati da virgola e contiene i suggerimenti dei libri.

1.Titolo
2.Suggerimenti

LibroController

Le quattro funzioni della classe **`Visualizza`** vengono richiamate all'interno del metodo

`void visualizzaLibro (String libro)`

contenuto nella classe controller **`LibroController`** (per la schermata Libro.fxml)

6.Librerie

- La classe **`Librerie`** rappresenta una gestione delle librerie personali degli utenti in un sistema. Consente agli utenti autenticati di creare librerie, visualizzarle e selezionare libri al loro interno.
- La classe eredita da **`RegUtente`**, che gestisce l'autenticazione degli utenti.

Strutture dati

- **`List<String> nomeLib = new ArrayList<>();`**
- **`List<String> listaLib = new ArrayList<>();`**
  Utilizzata per memorizzare le librerie personali di un utente e i libri contenuti nelle librerie.

Dettagli dei metodi

`1.boolean registraLibreria(String id, String nomeLib, String tit)`

Descrizione:Consente all'utente di creare una nuova libreria e di aggiungere libri.

`2.List<String> visLib (String userid, String nomeLib)`

Descrizione:Permette agli utenti di visualizzare i libri contenuti nelle loro librerie.

Pattern file CSV

- La classe memorizza i dati delle librerie nel file **`Librerie.dati.csv`** e segue il seguente pattern:

1.UserID
2.NomeLibreria
3.Libri

7.Valutazione

- La classe **`Valutazione`** gestisce la valutazione dei libri, consentendo agli utenti di assegnare voti a diverse categorie, aggiungere note, e fornire suggerimenti per altri libri.
- La classe estende **`Librerie`**, ereditandone i metodi e le funzionalità per la gestione delle librerie.

Strutture dati

- **`public int[] val;`**
- **`this.val = new int[6];`**

Memorizza le valutazioni assegnate dall'utente per ciascuna categoria.

Dettagli dei metodi

`1. void inserisciValutazioneLibro (String id, String titolo, int [] val, String note)`

Descrizione:Consente agli utenti di valutare un libro in diverse categorie e aggiungere una nota.

`2. boolean inserisciSuggerimentoLibri (String id, String titolo, String sugg)`

Descrizione:Permette agli utenti di aggiungere suggerimenti per un libro già presente nella propria libreria.

Pattern file CSV

La classe utilizza due file CSV:

- **`ValutazioniLibri.dati.csv:`** contiene le valutazioni dei libri e segue il seguente pattern:

1. Titolo: Titolo del libro.
2. Valutazione1, Valutazione2, ...: Voti per ciascuna categoria.
3. VotoFinale: Media dei voti.
4. Nota: Nota aggiunta dall'utente.

- **`ConsigliLibri.dati.csv:`**contiene suggerimenti associati a un libro e segue il seguente pattern:

- UserID: Identificativo univoco dell'utente.
- Libro: Titolo del libro.
- Suggerimenti