BIBLIOTECA VIRTUALE

FANTOZZI

DANIELE

213819

Il progetto prevede lo sviluppo di una biblioteca virtuale , in cui mettere insieme collezioni, servizi e persone a supporto dell'intero ciclo di vita di creazione, uso, preservazione di dati, informazione e conoscenza. Lo scopo di questo progetto è la digitalizzazione di manoscritti, che costituiscono un patrimonio bibliografico antico per un totale di 60.000 carte (ms. sec. XV-XIX).

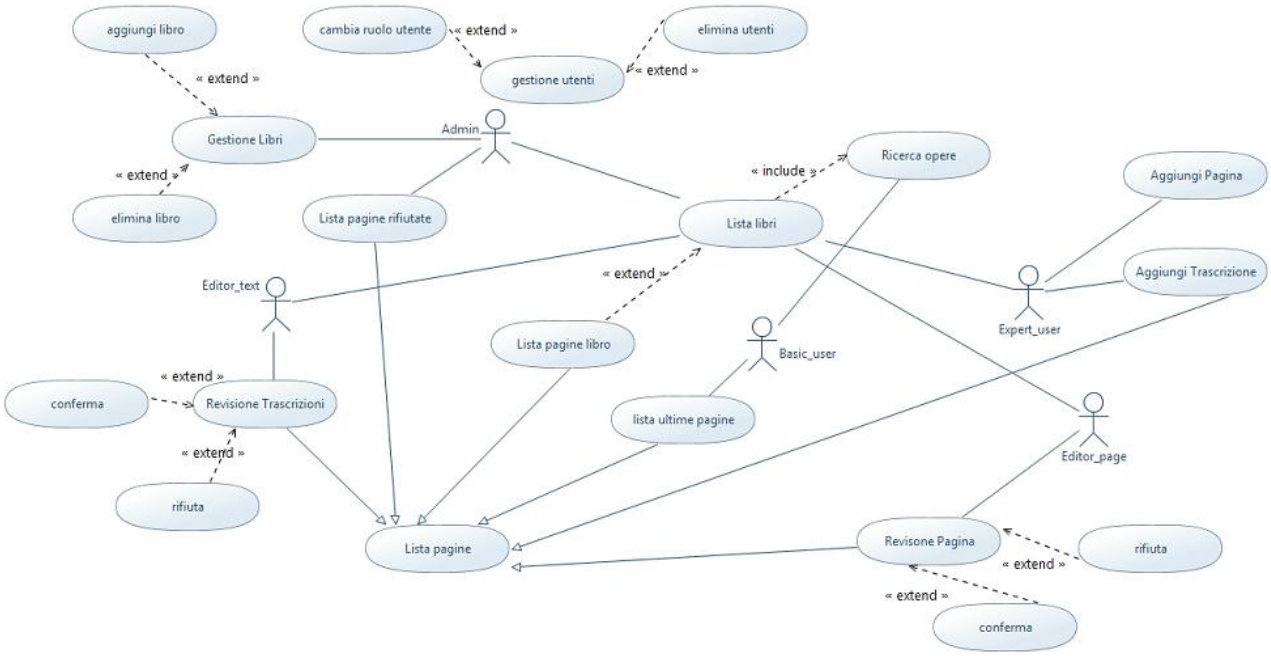
Ogni manoscritto appena digitalizzato passa allo stato “wait” in attesa di essere revisionato da un “revisore di pagine” che controlla che la pagina appartenga al libro selezionato e che la digitalizzazione sia conforme a determinati standard. Se confermato il manoscritto passa allo stato “yes” altrimenti passa allo stato “not”.

Una volta accettata (quindi si trova nello stato “yes”) allora può essere trascritta e la trascrizione passa allo stato ”wait” in attesa di essere revisionata da un “revisore trascrizioni” che verifica se la trascrizione sia conforme a determinati standard. Se confermata la trascrizione passa allo stato “yes” altrimenti passa allo stato “not”.

Le pagine con eventuale trascrizione (entrambe confermate) potranno essere visionate da tutti gli utenti (ad eccezione del basic_user come descritto in seguito) trovando il libro di cui fa parte attraverso la barra di ricerca oppure scorrendo la lista di tutti i libri (solo per gli expert_user); è comunque visualizzabile una pagina confermata ma che non ha una trascrizione nello stato “yes”.

Ogni utente registrato, quindi non basic_user è parte attiva del progetto ed a seconda del gruppo di cui fa parte può contribuire ad espandere il catalogo già presente. Verranno quindi illustrati adesso i vari tipi di utente e le funzioni che possono svolgere:

-USE CASE DIAGRAM



- BASIC_USER: questo utente non è un utente registrato, infatti effettua il login attraverso il pulsante “login come ospite” senza inserire username e password. Questo utente può visualizzare l’elenco delle ultime 5 pagine e trascrizioni pubblicate e può ricercare un libro ma senza visualizzare le pagine che ne fanno parte.

- EXPERT_USER: qualsiasi utente che si vuole registrare verrà registrato come “expert_user”, infatti solo l’amministratore può assegnare un ruolo diverso a tutti gli utenti. Questo utente può:
  -Visualizzare la lista di tutti i libri attraverso il pulsante “LISTA LIBRI”.
  -Aggiungere una nuova pagina (attraverso il pulsante “AGGIUNGI PAGINA”) inserendola in un libro già presente (come già detto la pagina inserita sarà nello stato wait). Ovviamente non può essere aggiunta una pagina con un numero pagina già esistente.
  -Aggiungere una nuova trascrizione (attraverso il pulsante “AGGIUNGI TRASCRIZIONE”) a una pagina nello stato “yes” e che non ha già una trascrizione nello stato “yes”. La trascrizione verrà aggiunta nel formato TEI.

- EDITOR_PAGE: la funzione di questo utente è di revisionare le pagine (premendo il pulsante “LISTA PAGINE”) nello stato “wait” presenti nel database e decidere se accettarle (quindi renderle visualizzabili a tutti gli altri utenti e renderle disponibili per aggiungere una trascrizione) oppure rifiutarle.

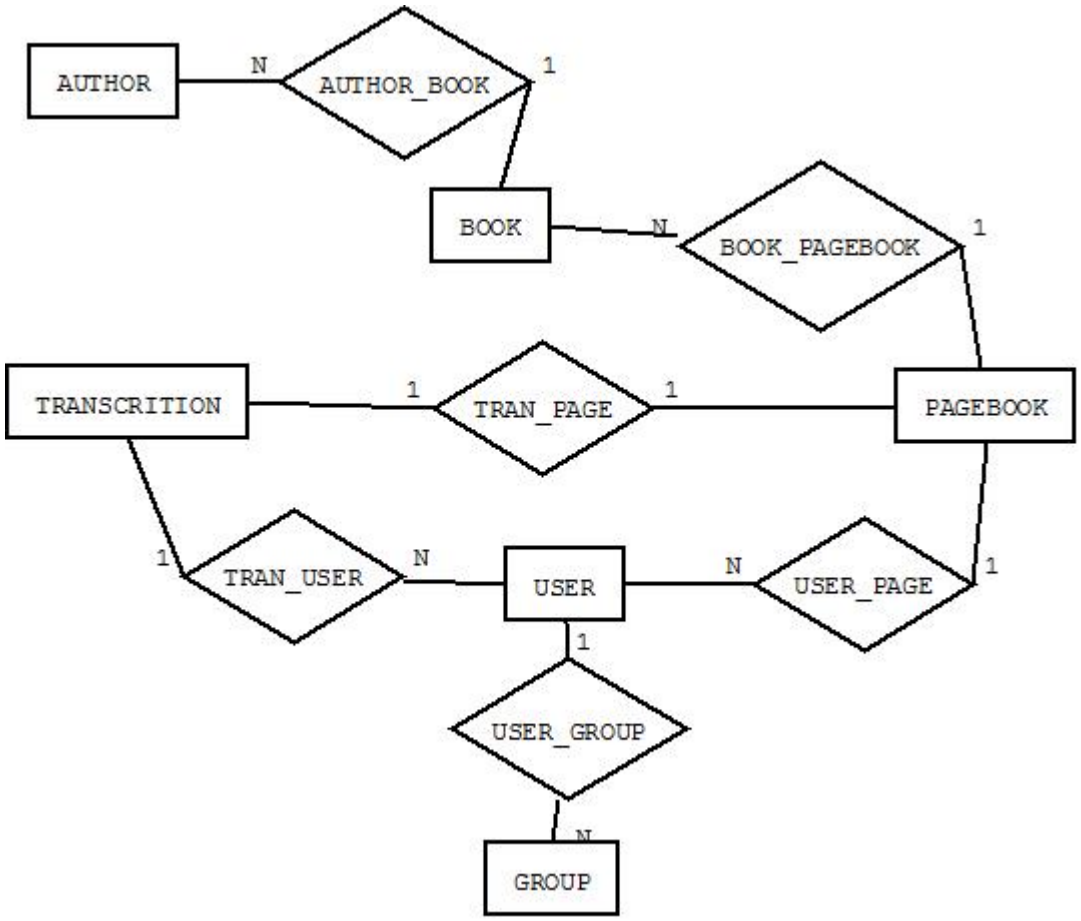
- EDITOR_TEXT: la funzione di questo utente è di revisionare le trascrizioni non ancora confermate, quindi nello stato “wait” (attraverso il pulsante “LISTA TRASCRIZIONI”) e verificare se possono essere accettate e renderle visualizzabili anche da tutti gli altri utenti.

- ADMIN: la funzione di questo utente è quella di gestire e garantire il funzionamento dell’intero sistema.
  Le sue funzioni sono:
  -cambiare il ruolo degli utenti: come già detto prima ogni utente che si registra sarà registrato come expert_user. Sarà quindi compito dell’admin promuovere un utente a un altro ruolo.
  -gestione di libri (premendo il pulsante “LISTA LIBRI”: in questa sezione vengono gestiti i libri presenti nel sistema. Si può quindi decidere se eliminare un libro (comporta l’eliminazione a cascata di tutte le pagine di quel determinato libro), modificare le informazioni del libro selezionato oppure aggiungerne uno nuovo. Da notare che affinché l’aggiunta di un nuovo libro vada a buon fine bisogna specificare l’autore e se non presente nel database si ha quindi la possibilità di aggiungerne uno nuovo.
  -ultime pagine: l’admin è l’unico utente che può visualizzare le ultime ultime pagine non confermate (quindi nello stato “not”) dagli editors. L’admin quindi può avere una visione completa di tutto quello che accade all’interno dell’applicazione.

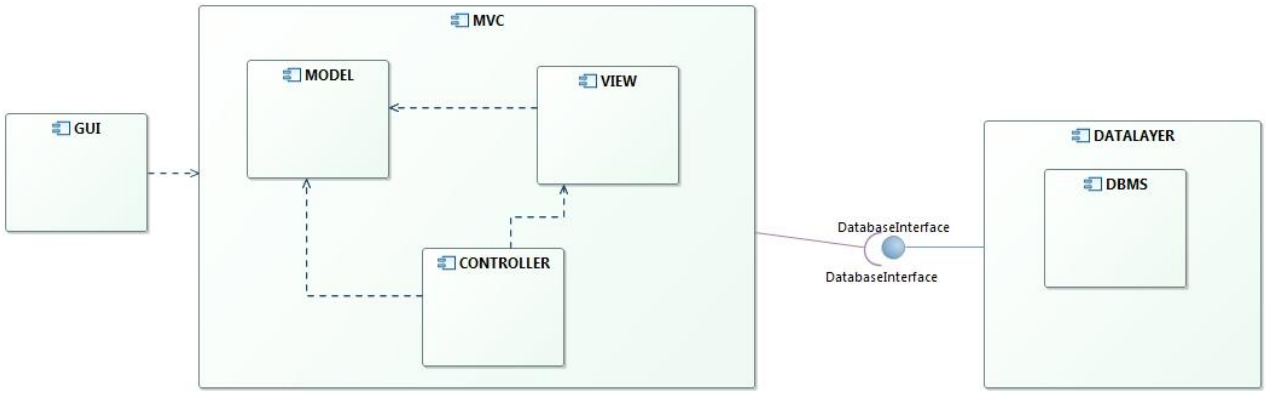
-elimina utenti: l'admin può visualizzare l'elenco completo degli utenti registrati all'applicazione (ad eccetto degli altri admin) e può decidere di eliminarli.

Inoltre tutti gli utenti registrati, quindi non gli utenti che effettuano il login come basic_user, possono visualizzare la lista dei libri dell'autore del libro cercato o facente parte della lista , premendo il pulsante "altri libri ".

DATABASE



ARCHITETTURA SOFTWARE



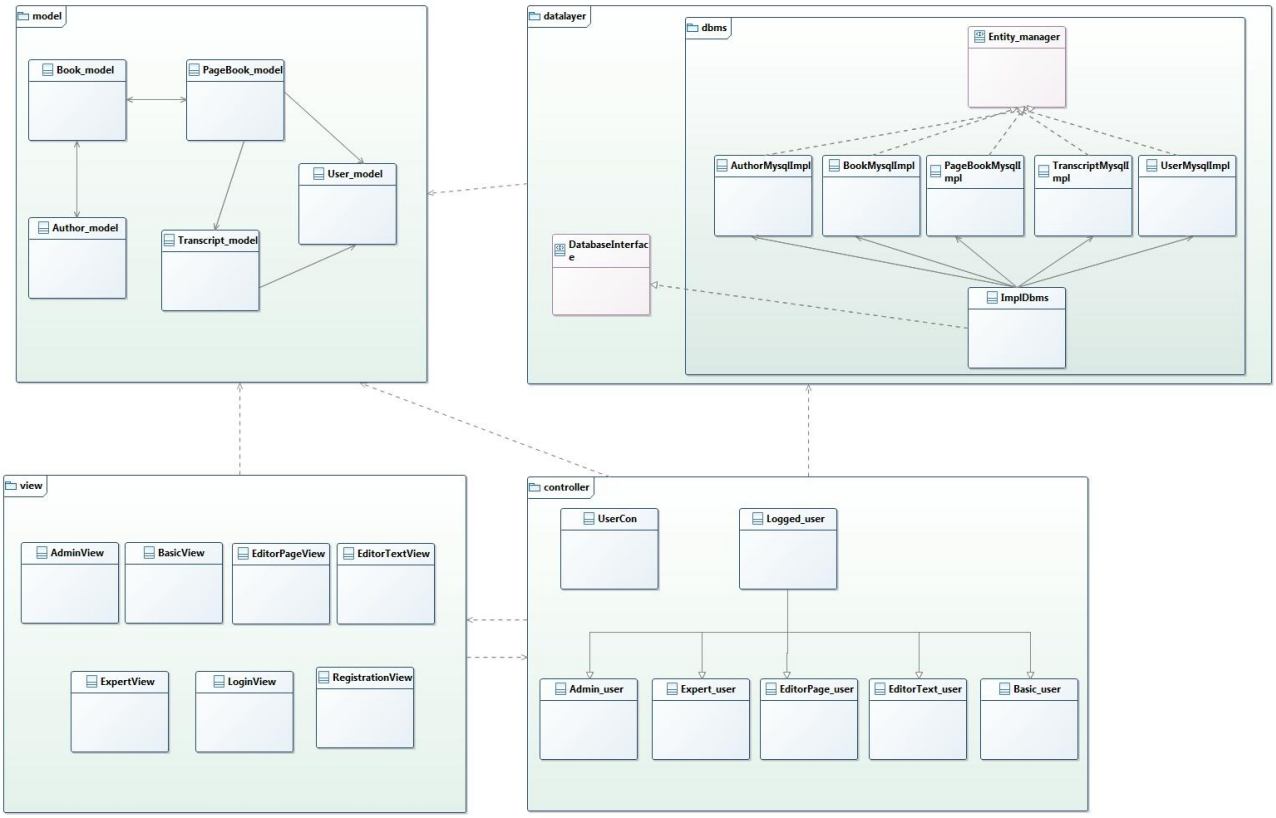
Il sistema è composto da tre componenti separate: GUI, MVC e DATALAYER. La prima coincide con l'interfaccia grafica del sistema e rispecchia tutte le funzionalità di un utente in base ai privilegi del gruppo di cui fa parte.

MVC è la parte di controllo del sistema che risponde alle richieste della GUI richiamando un'altra pagina (che fa parte della GUI) o richiedendo e/o modificando i dati presenti nel database. Per svolgere quest'ultima operazione MVC si interfaccerà con il component DATALAYER che sarà il ponte tra l'applicazione e i dati che saranno presenti in un server esterno.

Essendo l'applicazione stata pensata per interfacciarsi con un database mysql, all'interno del component DATALAYER sarà presente il component DBMS all'interno del quale, attraverso l'implementazione dell'interfaccia "DatabaseInterface" , verranno implementate tutte le query utili all'applicazione per interrogare il database mysql utilizzato. Sarà comunque possibile, in caso si avesse l'intenzione di interfacciarsi con un diverso tipo di database, aggiungere una nuova implementazione dell'interfaccia "DatabaseInterface" senza modificare uno degli altri componenti rispettando i criteri di information hidding che permettono di nascondere tutti i dettagli implementativi che si svolgono all'interno del component.

I dati saranno immagazzinati all’interno di un server esterno accessibili da più utenti contemporaneamente. Il catalogo infatti sarà in continuo aggiornamento, quindi attraverso il server esterno l’utente sarà sempre aggiornato sui manoscritti e trascrizioni presenti e potrà lui stesso aggiungere nuovo materiale che potrà essere subito revisionato dagli editors.

CLASS DIAGRAM



Come già illustrato nel Component Diagram per la parte di controllo si è deciso di utilizzare il pattern architetturale MVC che consente di separare la logica di presentazione dei dati dalla logica di business.

Il sistema quindi è diviso in 5 package che riflette la suddivisione di cui si è già parlato precedentemente:

-gui

-datalayer

-model

-view

-controller

- GUI: in questo package sono presenti tutte le interfacce grafiche utilizzate dagli utenti. Ogni tipo di utente (admin,expert…) ha un pannello principale (homeAdmin,HomeExpert..) dove l'utente può effettuare tutte le operazioni. Esso è composto da una barra del menu con tutte le azioni che può fare l'utente e un pannello centrale che a seconda dell'azione selezionata dall'utente cambia il suo contenuto attraverso il layout "cardlayout" . I diversi contenuti che può assumere il pannello centrale sono implementati nelle diverse classi contenute nel package GUI.
  Nelle classe ListBooks e nella classe ListSearch per esempio attraverso l'overload del metodo "getBooks" per la prima e "getPages" vengono implementati contenuti diversi sia in base all'azione scelta dall'utente sia al tipo di utente . Infatti con il metodo getPages ad eccezione di quando si visualizza la lista delle pagine di un determinato libro si nota la differenza dei contenuti caricati tra un metodo e un altro a seconda della view passata come parametro. Per quanto riguarda la visualizzazione della pagina singola (sia per ricerca, sia per revionarla, sia per aggiungere una trascrizione) è possibile zoomare l'immagine attraverso i due pulsanti "+" e "-" per rendere il lavoro più accurato possibile.

- MODEL: In questo package vengono descritti tutti gli oggetti gestiti all'interno dell'applicazione che riflettono le entità e relazioni presenti all'interno del database. Le classi presenti in questo package vengono utilizzate sia dalle view sia dai controller per creare i contenuti della gui e aggiornare il database

- VIEW: esiste una view per ogni tipo di utente. Ognuna di esse ha il compito di caricare l'interfaccia grafica principale per il tipo di utente a cui si riferisce. Inoltre ha il compito di intercettare tutti gli input dell'utente e in caso di una modifica oppure una richiesta al database di passare la chiamata al controller e ricevuta risposta da quest'ultimo modellarla per essere utilizzata dalla GUI.

- DATALAYER: il datalayer rappresenta la connessione tra database e applicazione locale. In questo package viene definita l'interfaccia "DatabaseInterface" dove sono presenti le dichiarazioni di metodo per la modifica/selezione dei dati gestiti dall'applicazione. È inoltre presente il package "DBMS" dove viene data un'implementazione dell'interfaccia appena descritta definendo le query per il database MySql. Le query verranno poi eseguite in una delle implementazioni dell'interfaccia "Entity Manager" e i risultati delle stesse saranno poi ritornate al controller chiamante .

- CONTROLLER: come per le view esiste un controller per ogni tipo di utente. Ogni controller ha innanzitutto il compito di caricare il pannello del login e registrazione e una volta che un utente ha effettuato il login caricare la view del gruppo dell'utente che caricherà l'interfaccia principale (come descritto prima). Inoltre il controller è l'unico che può modificare o prendere dati dal database (attraverso il datalayer).