Progetto OOSD 2017/2018

Teammates:

Lorenzo De Filippis: 230011;
Davide Fontana: 249194;
Alessandro Di Matteo: 249192;

REQUISITI

1) Registrazione/login da parte di ogni tipo di utente: il sistema deve garantire la registrazione ad ogni utente, controllando l'unicità degli account; deve anche garantire agli utenti registrati di poter effettuare il login sulla piattaforma.

2) Consultazione e download opere: per determinati tipi di utenti il sistema deve garantire la consultazione ed eventualmente un download della stessa in caso l'utente abbia i privilegi richiesti.

3) Caricamento opere: possibilità per certi utenti di caricare determinate opere all'interno della piattaforma.

4)Richiesta da parte dei Viewer di essere promossi a Transcriber.

1. Documento dei requisiti:

1.1 VIEWER:

-Sezione consultazione opere digitali e ricerca: ogni utente di tipo viewer ha la possibilità di consultare e cercare le opere digitali.
-Sezione modulo di richiesta download: un utente può richiedere l'abilitazione al download.
-Modulo richiesta promozione a transcriber: un utente di tipo viewer può richiedere la promozione a transcriber.
-Sezione anagrafica: permette agli utenti di visualizzare i propri dati personali.

1.2 UPLOADER:

- Sezione digitalizzazione di opera: permette ad un utente di tipo uploader di effettuare l'upload sulla piattaforma di determinate opere o parti di essa.
- Sezione anagrafica: permette all'utente di visualizzare i propri dati personali.

1.3 TRANSCRIBER:

- Sezione textedit digitalizzazione opere: ogni utente di tipo transcriber ha la facoltà di trascrivere in lingua corrente determinate parti di opere.

1.4 MANAGER:

- Sezione assegnamento opera ad uno o più transcriber: permette ad un utente di tipo manager l'assegnamento di un'opera da trascrivere ad uno o più transcriber.
- Sezione valutazione, correzione e validazione: ogni utente di tipo manager deve correggere, valutare ed eventualmente validare determinate trascrizioni.
- Sezione gestione livello dei transcriber: permette ad un utente di tipo manager di gestire il livello di privilegi degli utenti transcriber.

1.5 ADMINISTRATOR:

- Sezione eliminazione utente: permette di eliminare utenti dal sistema

- Sezione promozione utenti: permette di promuovere un utente con un nuovo ruolo
- Sezione declassamento utenti: permette di declassare un utente
- Sezione promozione a Downloader: permette di promuovere un utente a Downloader

1.2 Requisiti non funzionali:

Usability: il sistema deve garantire affidabilità e facilità di uso per l'utente;

Reliability: il sistema dovrà garantire all'utente le funzioni a disposizione senza errori;

Security: protezione dei dati e dei libri digitalizzati.

Attori:

Viewer:

Tale parte del sistema consente la consultazione delle opere digitali a utenti registrati. Consente la ricerca nel catalogo per metadati (descritti in seguito) oppure all'interno del testo della trascrizione. Le opere possono essere suddivisi in categorie. Appena si sceglie un'opera, verrà visualizzata con una schermata che avrà sulla destra il testo della trascrizione (se disponibile) e sulla sinistra l'immagine della pagina dell'opera che si sta visualizzando, sarà possibile sfogliare le pagine tramite un paginatore. Alcuni utenti con particolari privilegi possono effettuare il download dell'opera. L'utente può fare richiesta tramite un modulo per essere collaboratore del sistema (trascrittore). Gli utenti possono accedere al loro profilo personale dove saranno visualizzati i dati inseriti nella registrazione, tra cui: titolo di studio, professione, indirizzo, email, etc.

Uploader

Ogni opera è formata da più immagini (scansioni), ognuna delle quali rappresenta una pagina del manoscritto. Per ogni opera vengono caricati dei metadati (titolo, anno, ...). Al fine di rendere agevole il caricamento delle immagini e il successivo controllo, per ogni opera si vuole fornire la visualizzare sia tutte le pagine in miniatura e di una pagina per volta da scorrere con un paginatore. La digitalizzazione viene controllata da supervisori all'acquisizione per assicurarne la correttezza (ad esempio, in accordo con standard richiesti) e la qualità.

Transcriber

Ogni opera acquisita deve essere trasformata in un testo digitale, ciò avviene attraverso operazioni di trascrizioni in formato TEI (Text Encoding Initiative). Le trascrizioni sono digitate manualmente attraverso un text editor TEI integrato. Quindi ogni immagine (pagina) avrà il corrispondente testo associato. Più trascrittori possono lavorare sulla stessa pagina, è necessario

sincronizzare le modifiche. Le trascrizioni sono oggetto di revisione da parte di revisori alle trascrizioni.

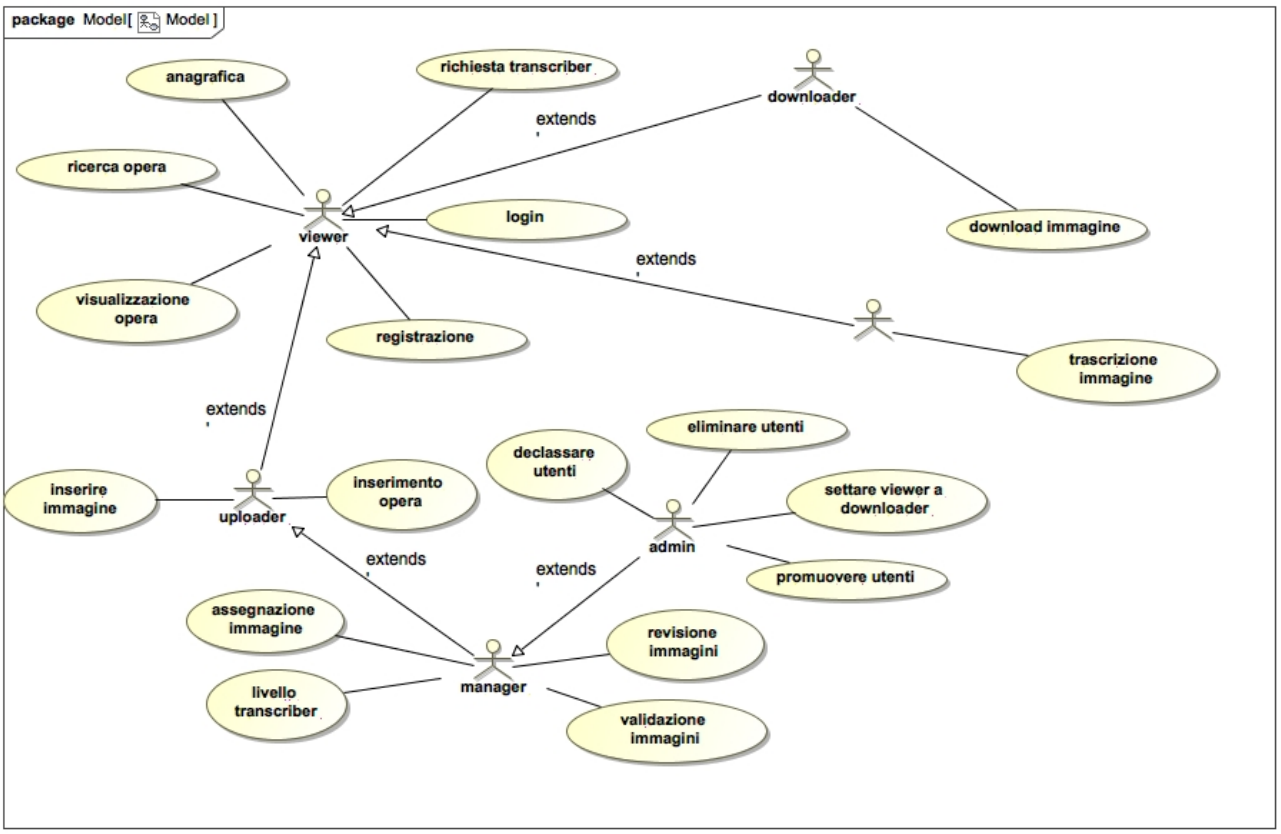
Manager

Questo sottosistema gestisce le assegnazioni, ovvero consente di assegnare parte di un'opera (1 o più immagini) a 1 o più trascrittori. Inoltre, consente di revisionare le trascrizioni concluse, di effettuare correzioni e validazione. E' possibile anche riassegnare delle pagine ai trascrittori. Consente la pubblicazione delle trascrizioni e delle opere (solo immagini). Gestisce i livelli dei trascrittori (ogni trascrittore ha un livello 1-5 in base alla sua esperienza). Consente la supervisione dell'acquisizione immagini.

Administrator

Questo sistema gestisce il ruolo degli utenti e possiede accesso a tutte le aree del sistema. Può declassare o promuovere un utente ad un nuovo ruolo e può dare il permesso agli utenti di scaricare le opere.

1.2) Use Case:



Scenari:

- Autenticazione: ogni user(Viewer, Transcriber, Uploader, Manager e Administrator) dovrà all'inizio registrarsi all'interno della piattaforma mediante email e password. Questi dati permetto di accedere alla piattaforma una volta avvenuta con successo la registrazione.

- Sezione consultazione opere digitali: l'utente viewer deve avere il modo di ricercare un libro in base al titolo o l'autore, dopodiché, se trovato deve avere il modo di consultarlo e se possibile scaricarlo.

- Sezione textedit: l'utente Transcriber ha il compito di digitalizzare un'opera, perciò deve poter avere un'ambiente di lavoro per poter trascrivere l'opera e per modificare le trascrizioni in corso.

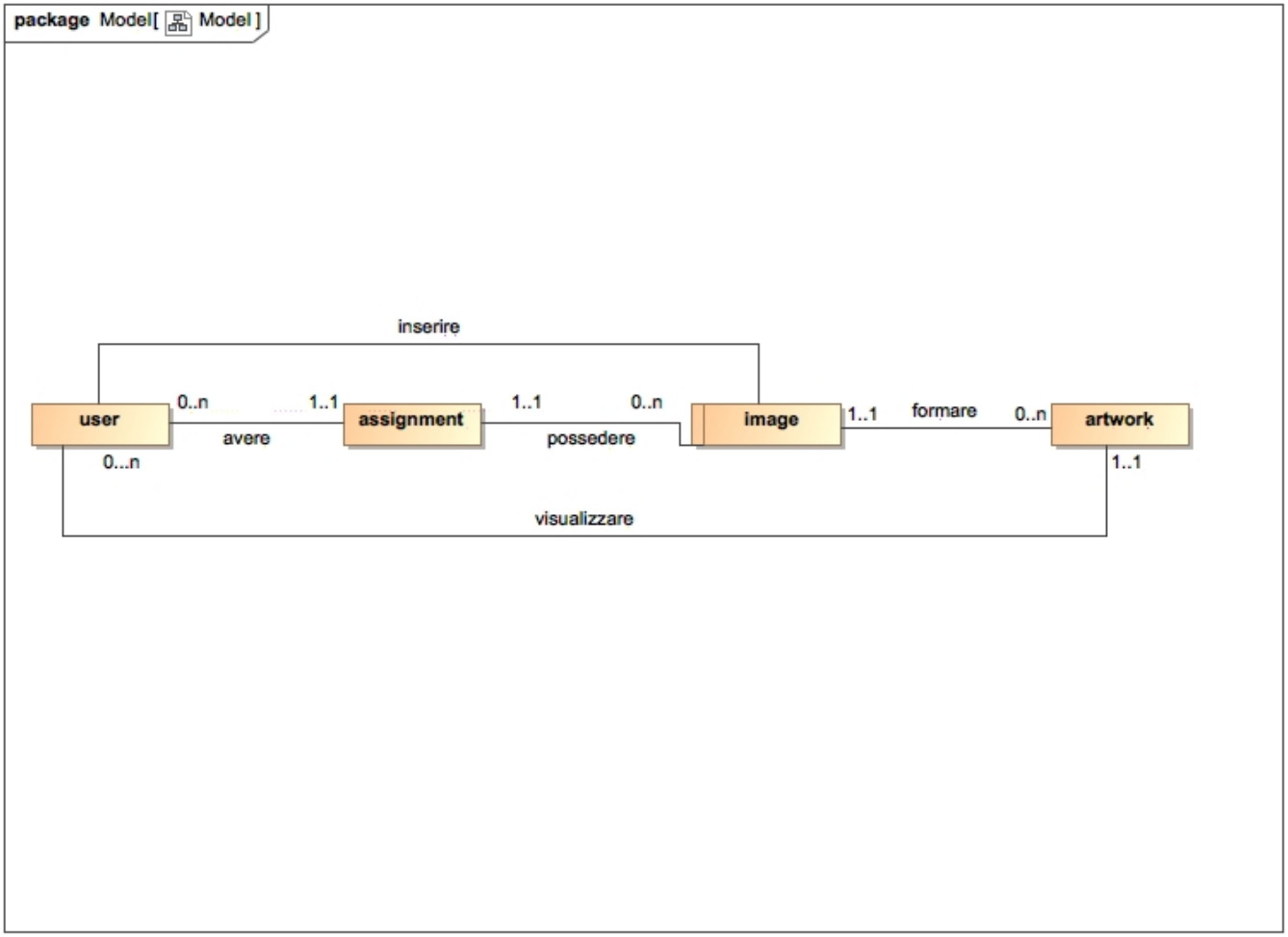
- Sezione revisione e controllo: l'utente Manager deve poter gestire le assegnazioni delle opere ai Transcriber, in più deve poter revisionare, correggere e pubblicare le trascrizioni

- Sezione upload opere: l'utente Uploader ha il compito di scannerizzare le pagine di ogni opera fornendo una visione miniaturizzata e una "pagina per pagina" tramite un paginatore.

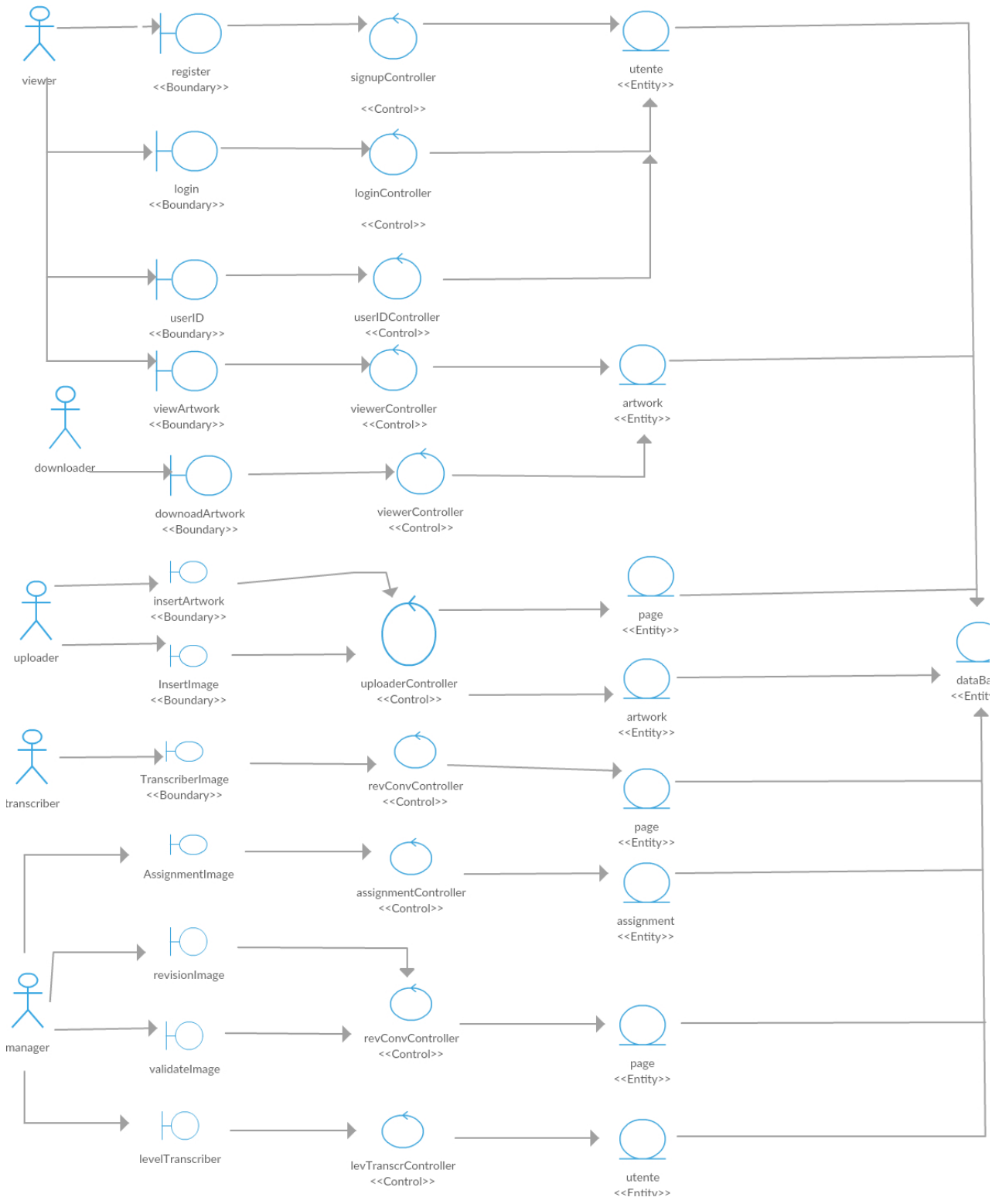
- Sezione gestione di tutto il sistema: l'utente Administrator ha il compito di gestire in back-end tutto il sistema, ovvero anagrafica utenti, gestione delle opere e manutenzione del sistema.

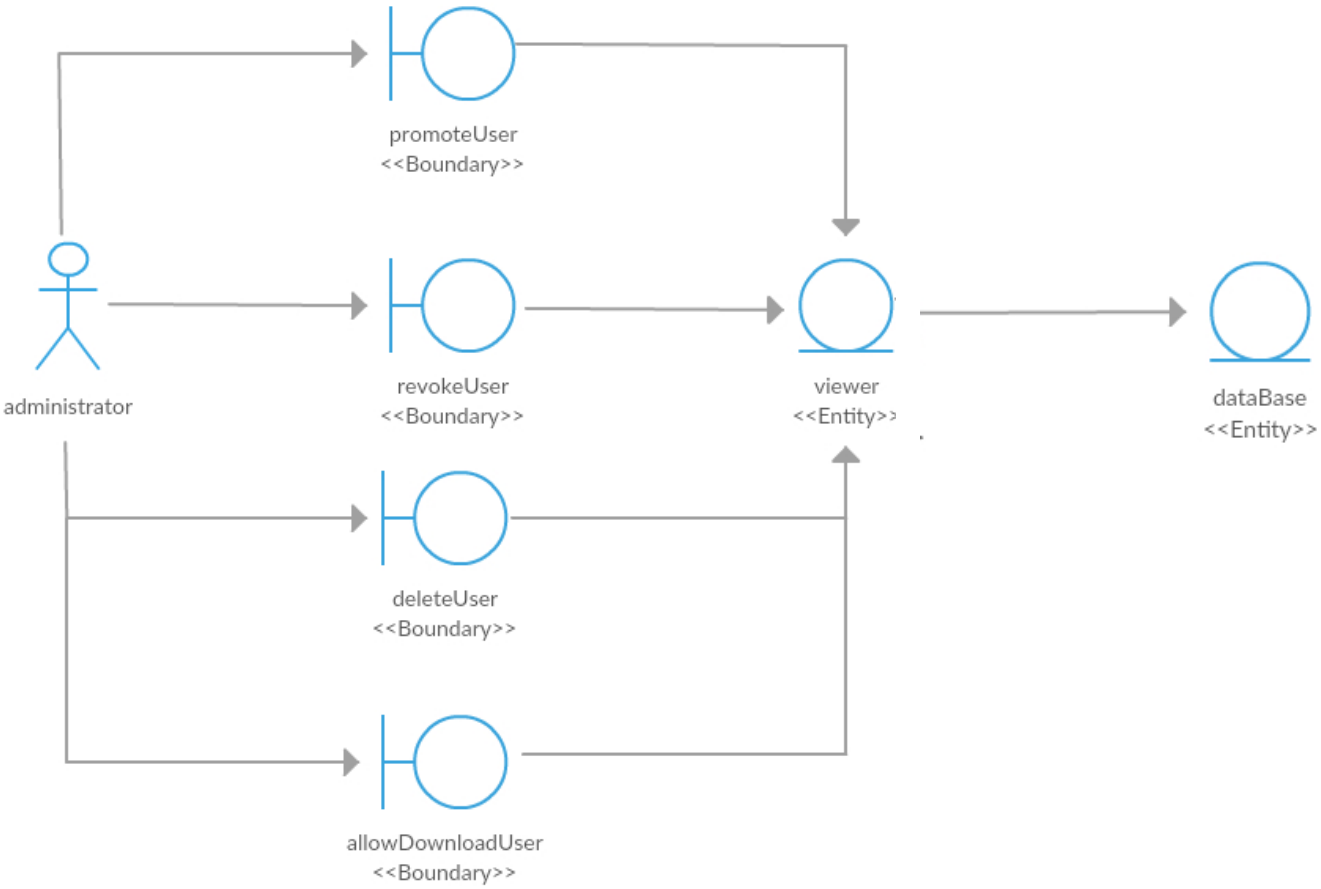
- Sezione promozione da Viewer a Transcriber: i Viewer hanno la possibilità di fare domanda per diventare Transcriber.

1.3) Modello class domain:

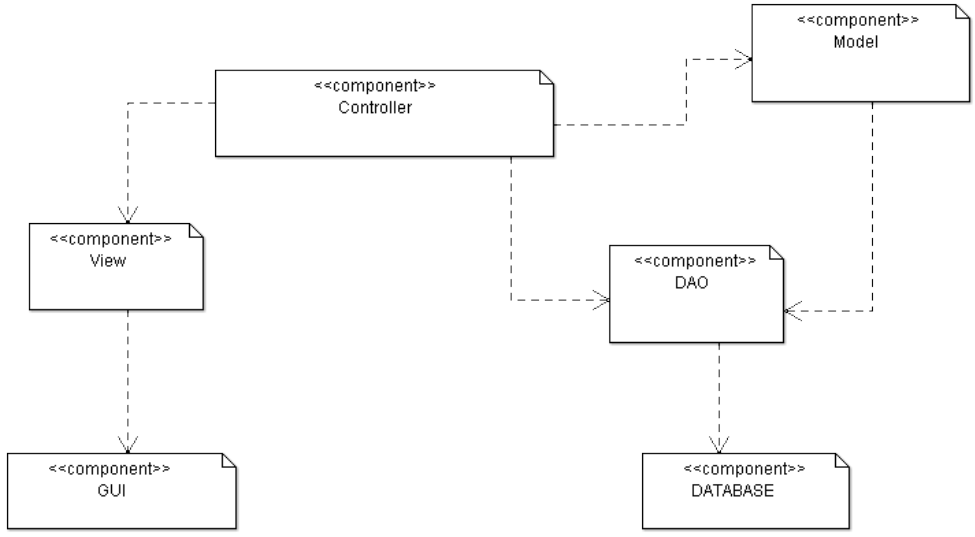


1.3) Modello class domain:





2.1) Modello dell'architettura del sistema



Il modello dell'architettura che abbiamo deciso di utilizzare è di tipo MVC (Model-View_controller).

Questo particolare tipo di pattern architetturale è basato sulla separazione dei componenti software che finiscono per esplicarsi come da nome in:

-Model: è la parte che si occupa di mettere a disposizione metodi per accedere ai dati utili dell'applicazione.

-View: è la parte che si occupa dell'interazione diretta con gli utenti finali (composta principalmente da viste) dell'applicazione permettendo l'interazione con il software. Si collega al model dato che permette di visualizzare i dati contenuti in quest'ultimo.

-Controller: è la parte del design che riceve i comandi impartiti dall'utente finale dell'applicazione portandoli a compimento. Si collega con la view infatti è tramite la quest'ultima che il controller ricevi in input i vari segnali impartiti dagli utenti finali.

Il motivo principale della nostra scelta è stato influenzato principalmente dalla riusabilità del codice che porta un notevole vantaggio, infatti ogni singola parte del modello incapsula la sua parte di informazioni rendendo tutto riutilizzabile separatamente.
L'altro motivo è la leggibilità del codice, infatti dividendo in maniera adeguata i vari package è molto più semplice il lavoro in team.

2.2) Descrizione dell’architettura

Il sistema come già detto innumerevoli volte è suddiviso in tre parti separate realizzando come da progetto il pattern architetturale di tipo MVC (Model-View-Controller).

In dettaglio abbiamo preso le seguenti decisioni:

-VIEW: la parte della view è stata suddivisa in front-view dove sono situate tutte le pagine con estensione FXML (le viste che presenta il sistema) tutti creati con il tool SceneBuilder. L'altra parte della view è chiamata front-view-controller dove sono stati inseriti tutti i file relativi ai controller delle pagine sopracitate con estensione FMXL.
Questo permette una notevole facilità nella comprensione del codice e nei collegamenti tra file XML e i vari controller che le gestiscono.

-MODEL: il package MODEL è stato suddiviso in due sotto-package: model-dao e model-vo.
Il model-dao contiene tutte le classi relative alla gestione delle operazioni da effettuare sul data-base che si trova dietro la nostra applicazione. Utilizzando questo pattern architetturale andiamo a garantire un maggiore livello di astrazione ed una più facile manutenibilità del nostro sistema.

L'altra parte del package model è il model-vo all'interno del quale troviamo troviamo tutte le classi relative agli oggetti principali che gestiremo all'interno della nostra piattaforma quali: utenti, opere, immagini.
Tramite il VO (value object) si vanno ad incapsulare i dati per ottenere una migliore gestione dei dati.

-CONTROLLER: il package CONTROLLER contiene tutte le classi che gestiscono le chiamate a funzioni relative ai nostri oggetti principali quali: image, user, artwork e assignment.
All'interno dei vari controller troviamo i principali metodi per la gestione degli oggetti sopracitati.
Il controller è dunque collegato al DAO (data access object) che abbiamo scelto di utilizzare per garantire una rigida separazione tra le componenti dell'applicazione.

2.3) Descrizione delle scelte adottate

I design pattern che abbiamo deciso di utilizzare sono i seguenti:

-SINGLETON: è un design pattern di tipo creazionale che ha lo scopo di garantire che di una determinata classe venga creata una e una sola istanza, e di fornire un punto di accesso globale a tale istanza.
Abbiamo scelto di utilizzare il pattern SINGLETON in quanto per gli oggetti di tipo user è importante averne instanziato uno solo oggetto per volta.

-ABSTRACT FACTORY: è un design pattern di tipo creazionale utilizzato fondamentalmente per garantire che un sistema sia indipendente dall'implementazione degli oggetti concreti.
Questa scelta è stata sfruttata principalmente all'interno delle classi del DAO.

-DATA ACCESS OBJECT: L’uso del pattern Data Access Object (DAO) permette di astrarre e incapsulare tutti gli accessi al data source. L’oggetto DAO gestisce la connessione con il data source per estrarre e/o immagazzinare i dati. Gli oggetti DAO implementano i meccanismi di accesso richiesti per lavorare con il data source. Il DAO nasconde del tutto ai suoi client i dettagli implementativi di accesso al data source. Poiché l’interfaccia esposta dal DAO non cambia quando il sottostante data source cambia implementazione, questo pattern permette al Data Access Object di adattarsi a differenti schemi implementativi senza che questo abbia alcun effetto sui client o sui componenti del business layer. In definitiva il DAO agisce da adapter tra i componenti del business tier e il data source.
Il vantaggio relativo all'uso del DAO è dunque il mantenimento di una rigida separazione tra le componenti dell'applicazione.

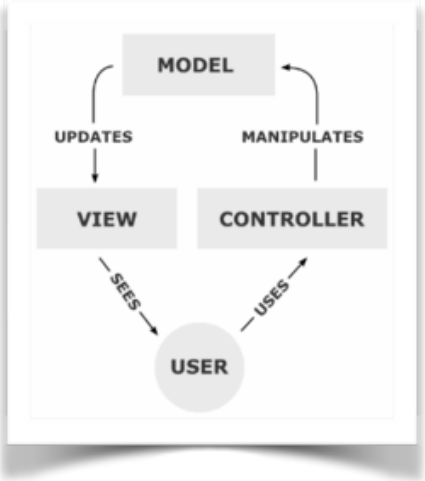
MVC

Il pattern è basato sulla separazione dei compiti fra i componenti software che interpretano tre ruoli principali:

- I model fornisce i metodi per accedere ai dati utili all'applicazione;
- I view visualizza i dati contenuti nel model e si occupa dell'interazione con utenti e agenti;

- l controller riceve i comandi dell'utente (in genere attraverso il view) e li attua modificando lo stato degli altri due componenti.
- Questo schema, fra l'altro, implica anche la tradizionale separazione fra la logica applicativa (in questo contesto spesso chiamata "logica di business"), a carico del controller e del model, e l'interfaccia utente a carico del view.[3]

I dettagli delle interazioni fra questi tre oggetti software dipendono molto dalle tecnologie usate (linguaggio di programmazione, eventuali librerie, middleware e via dicendo) e dal tipo di applicazione (per esempio se si tratta di un'applicazione web, o di un'applicazione desktop). Quasi sempre la relazione fra view e model è descrivibile anche come istanza del pattern Observer. A volte, quando è necessario cambiare il comportamento standard dell'applicazione a seconda delle circostanze, il controllerimplementa anche il pattern Strategy.



3.1) UML class diagrams rappresentanti l'object design con classi, interfacce e membri