



UNIVAQ

— ★ —
BIBLIOTECA DIGITALE

DOCUMENTAZIONE

Object Oriented Programming

Progetto biblioteca digitale

Marco D'Ettorre 222737

Francesco Proietti 238047

BIBLIOTECA DIGITALE.....	4
1 Requisiti	5
1.1 Requisiti funzionali.....	5
1.2 Requisiti non funzionali.....	8
2 System design	9
2.1 Component diagram	9
2.2 Sequence diagram.....	10
2.2.1 <i>Sequence trascrizione</i>	10
2.2.2 <i>Sequence verifica trascrizione</i>	12
2.3 Deploymentdiagram.....	13
2.4 Schema entità relazioni	14
2.4.1 <i>Entità</i>	14
2.4.2 <i>Relazioni</i>	14
2.4.3 <i>Diagramma ER</i>	15
2.5 Design decisions.....	16
2.6 Problemi noti.....	18
2.7 Design patterns.....	18
2.7.1 <i>Model-View-Controller</i>	19
2.7.2 <i>Data-Access-Object</i>	20
3 Software design	21
3.1 Class diagram.....	21
3.1.1 <i>Package Controller</i>	21
3.1.2 <i>Package Framework</i>	22
3.1.2.1 Sottopackage result.....	23
3.1.2.2 Sottopackage util	23
3.1.2.3 Sottopackage data.....	23
3.1.3 <i>Package data</i>	24
3.2 Class diagram.....	26
3.3 Problemi noti.....	27
4 Screenshots	27
4.1 Homepage	27
4.2 Registrazione.....	28
4.3 Accesso acquisitore.....	29
4.4 Accesso revisore acquisizione.....	29

4.5	Trascrizione	31
4.6	Revisione trascrizione.....	32
4.7	Accesso admin	33
4.8	Accesso da dispositivo mobile.....	34

BIBLIOTECA DIGITALE

Il progetto si propone di realizzare una biblioteca digitale di testi e studi che contribuiscono alla formazione della cultura all'interno dell'Università degli Studi dell'Aquila.

Il progetto è presente al seguente URL:

<https://github.com/marcodettorre/Biblioteca-Digitale>

ed è strutturato come segue:

- *src*: contiene il codice sorgente del progetto
- *doc*: contiene i documenti di design del progetto
- *javadoc*: contiene il codice javadoc del progetto

1 REQUISITI

1.1 REQUISITI FUNZIONALI

CODICE	TIPO	DESCRIZIONE
RF1.	RUOLI	I ruoli identificati all'interno dell'applicazione sono: revisore acquisizione, revisore trascrizione, acquisitore e trascrittore agli utenti registrati.
RF2.	ADMIN	L'amministratore può attribuire i vari ruoli agli utenti registrati.
RF3.	ADMIN	L'amministratore può rimuovere gli utenti.
RF4.	ADMIN	L'amministratore può inserire una nuova opera nel sistema.
RF5.	UTENTE	Un utente non loggato al sistema viene identificato come utente base.
RF6.	UTENTE	L'utente base può visualizzare solo l'elenco delle opere con i loro attributi.
RF7.	UTENTE	L'utente base può registrarsi al sistema tramite una form indicando: nome, cognome, username, email e password, diventando utente avanzato.

CODICE	TIPO	DESCRIZIONE
RF8.	UTENTE	L'utente avanzato può consultare completamente le opere pubblicate con le immagini, seguite eventualmente, dalla trascrizione correlata.
RF9.		
RF10.		
RF11.	UTENTE	Gli utenti possono cercare le opere in base ai seguenti criteri: titolo, autore, editore, anno di pubblicazione e lingua, da ricercare in AND.
RF12.	ACQUISIZIONE	L'acquisitore inizia la procedura di acquisizione di una nuova opera, facendo l'upload di un'immagine scannerizzata relativa ad una sua pagina e il cui processo di acquisizione non è stato iniziato da un altro.
RF13.	ACQUISIZIONE	L'immagine scannerizzata deve essere validata successivamente, da un revisore di acquisizione che verifica la correttezza dell'acquisizione.
RF14.	ACQUISIZIONE	L'immagine che passa il controllo del revisore diventerà immagine validata.
RF15.	ACQUISIZIONE	Le opere che hanno tutte le immagini validate risultano "acquisite".

CODICE	TIPO	DESCRIZIONE
RF16.	ACQUISIZIONE	L'opera che non ha tutte le immagini validate è un'opera in "revisione acquisizione".
RF17.	ACQUISIZIONE	La pubblicazione delle opere acquisite è riservata all'utente admin.
RF18.	ACQUISIZIONE	Il "revisore acquisizione" avrà a disposizione un catalogo di opere che sono in "revisione acquisizione".
RF19.	TRASCRIZIONE	Il trascrittore può trascrivere solo pagine di opere già acquisite e il cui processo di trascrizione non è stato iniziato da un altro.
RF20.	TRASCRIZIONE	Ad ogni pagina corrisponde la sua trascrizione.
RF21.	TRASCRIZIONE	La trascrizione deve essere validata da un revisore di trascrizione.
RF22.	TRASCRIZIONE	Un'opera che ha tutte le pagine trascritte e validate viene considerata "Trascritta".
RF23.	TRASCRIZIONE	L'opera acquisita che non ha tutte le trascrizioni validate, risulta in "revisione trascrizione".
RF24.	TRASCRIZIONE	La pubblicazione delle opere trascritte è riservata all'utente admin.

CODICE	TIPO	DESCRIZIONE
RF25.	TRASCRIZIONE	Il “revisore trascrizioni” avrà accesso ad un catalogo di opere in revisione trascrizione.
RF26.	REVISORI	Il “revisore acquisizione” e il “revisore trascrizione”, possono convalidare o rigettare l’immagine o la trascrizione che stanno revisionando.
RF27.	OPERE	Ogni opera è caratterizzata da: titolo, autore, editore, e metadati.
RF28.	OPERE	I metadati consistono di: num. pagine, lingua, data di pubblicazione e descrizione.

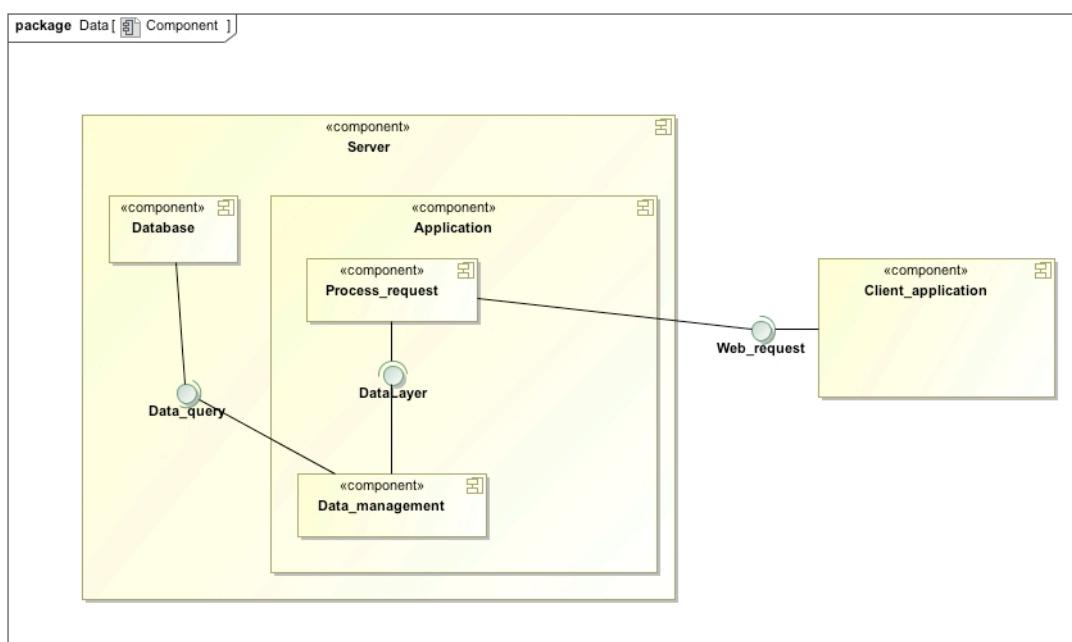
1.2 REQUISITI NON FUNZIONALI

CODICE	DESCRIZIONE
RNF1	La trascrizione deve essere in formato TEI
RNF2	Il trascrittore e il revisore trascrizione conoscono lo standard TEI
RNF3	il progetto deve essere scritto utilizzando il linguaggio Java
RNF4	Il sistema deve garantire 50 connessioni al database in contemporanea

CODICE	DESCRIZIONE
RNF5	L'interfaccia del client deve essere responsive e quindi fruibile da device mobili e computer desktop

2 SYSTEM DESIGN

2.1 COMPONENT DIAGRAM



L'architettura è stata decomposta in 4 componenti principali che sono:

Database: unità che gestisce la memorizzazione dei dati.

Data_management: componente dell'applicazione che interagisce con il database per la gestione dei dati.

Process_request: componente dell'applicazione per la gestione delle richieste web.

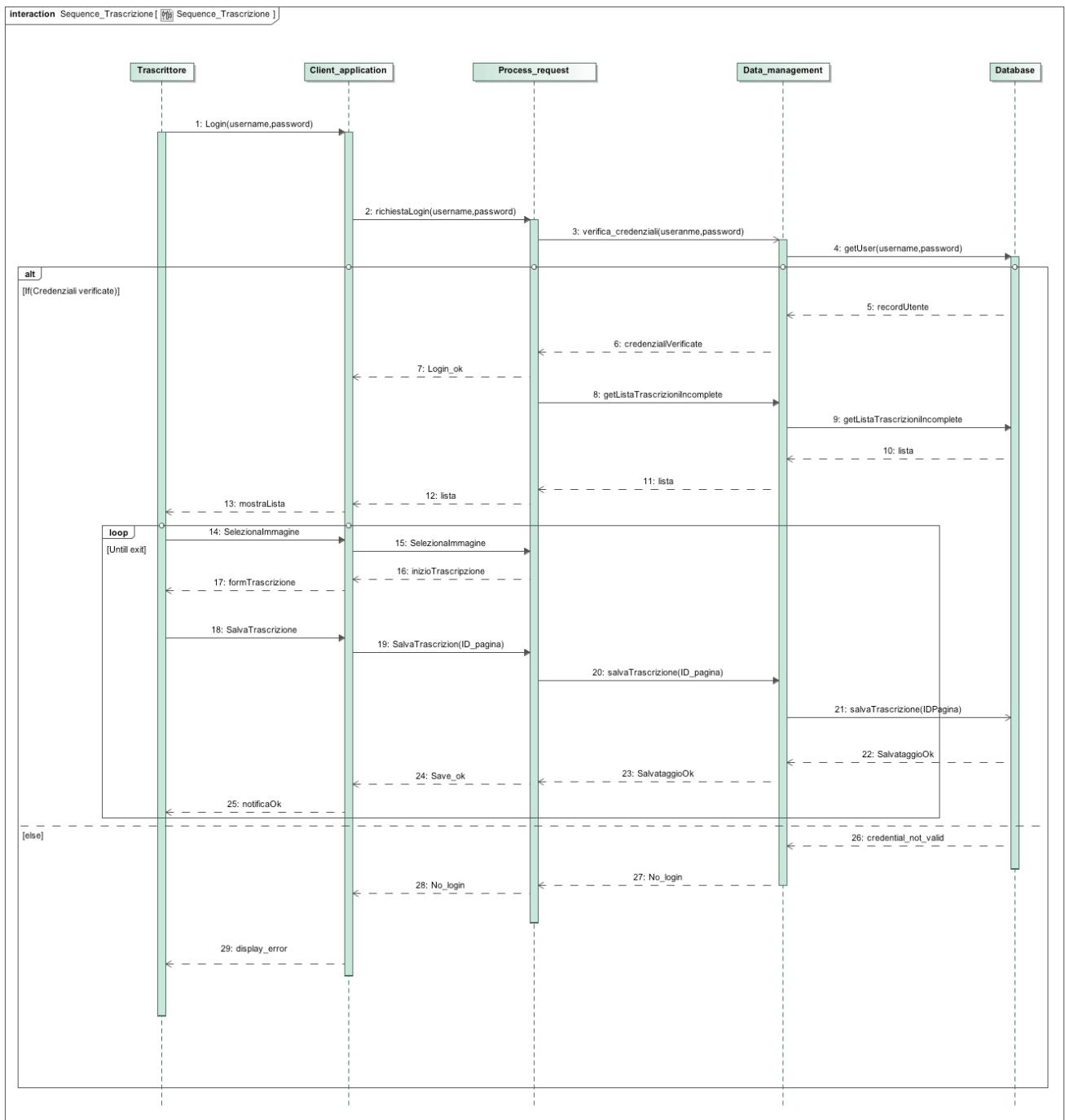
Client_application: web browser per l'interazione con l'applicazione.

2.2 SEQUENCE DIAGRAM

2.2.1 Sequence trascrizione

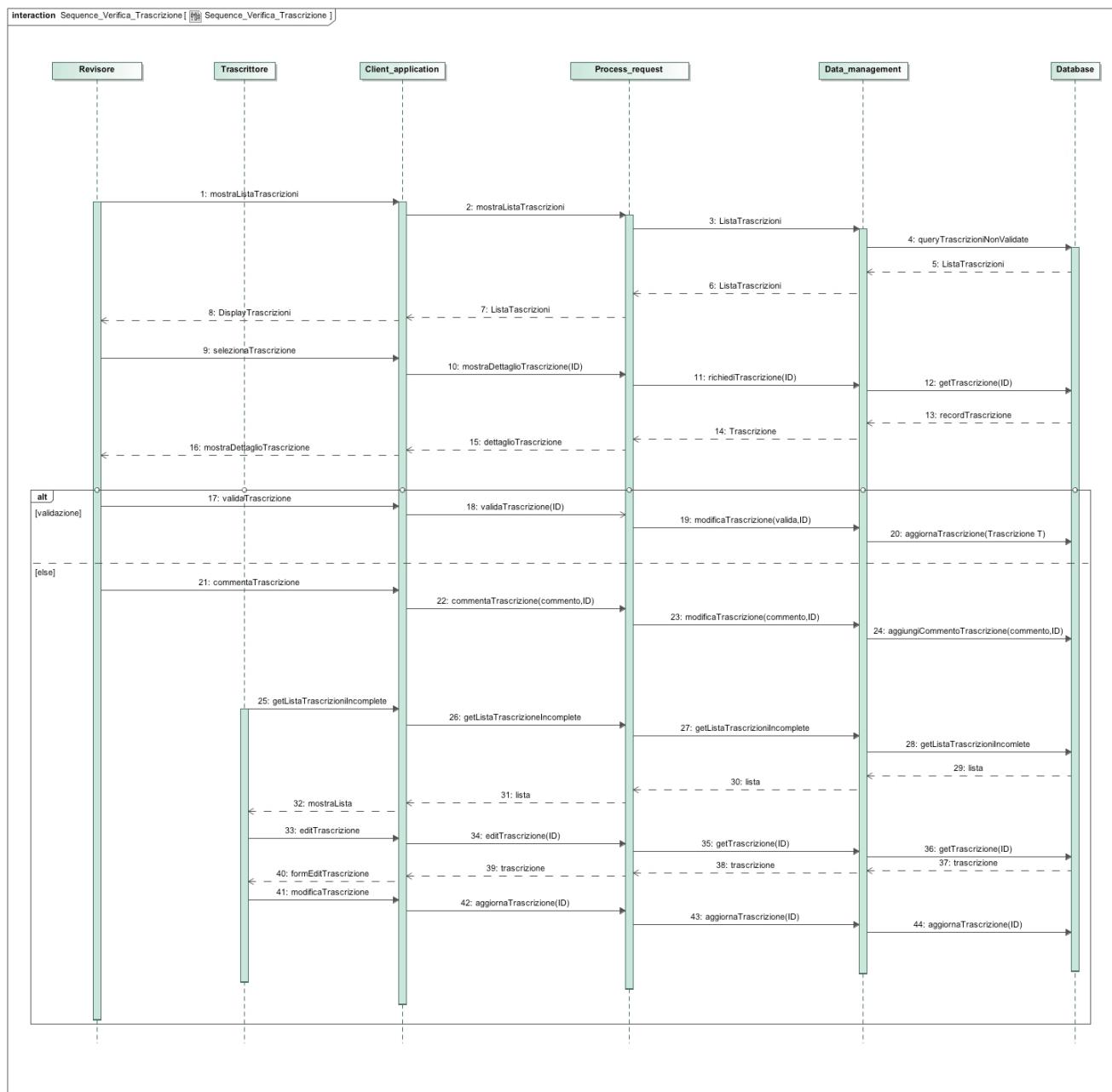
Nel seguente schema viene esplicitato il processo con cui un utente “trascrittore” si autentica nel sistema ed effettua una trascrizione.

Il sistema, una volta verificate le credenziali di accesso, visualizza direttamente all’utente la pagina contenente le opere che hanno immagini senza una trascrizione. L’utente, scegliendo un’opera, potrà scegliere l’immagine da trascrivere dall’elenco delle pagine.

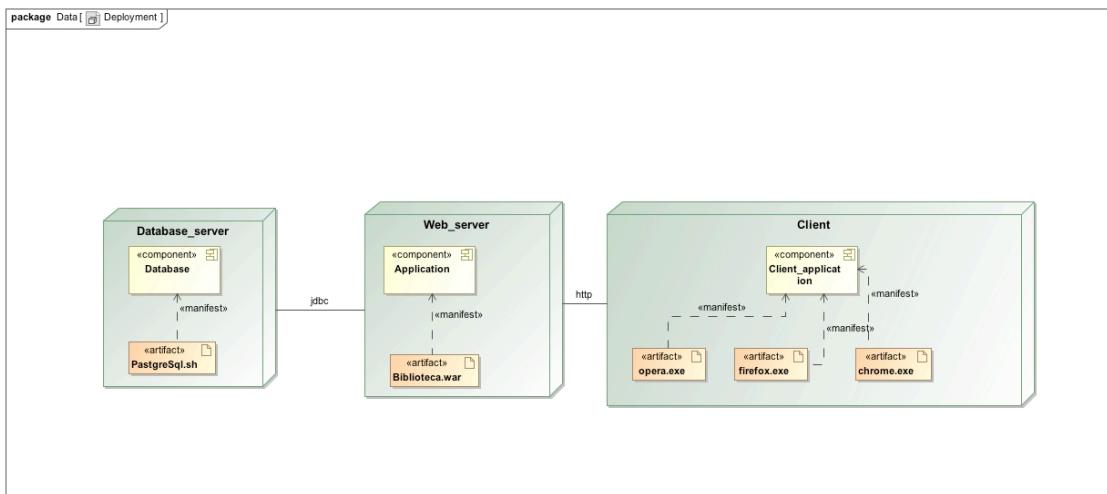


2.2.2 Sequence verifica trascrizione

Nel seguente diagramma viene descritto il processo con cui una trascrizione viene validata da un “revisore trascrizione”.



2.3 DEPLOYMENTDIAGRAM



Le componenti logiche possono essere distribuite in 3 dispositivi diversi:

- **Database_server**: sul quale è installato il DBMS PostgreSQL che implementa la componente Database. Si è scelto di mettere questa componente separata dal server, pensando di poterla utilizzare in maniera distribuita in modo da non sovraccaricare il workload dei vari componenti, garantendo così un servizio più fruibile.

In un lavoro futuro è possibile anche utilizzare un database di tipo cloud.

- **Web_server**: la componente centrale sulla quale è deployata la parte logica dell'applicazione. Fornisce accesso al database e interfaccia per le richieste del client.
- **Client**: workstation o device mobile da cui poter accedere al servizio tramite browser che realizzano la clientapplication.

2.4 SCHEMA ENTITÀ RELAZIONI

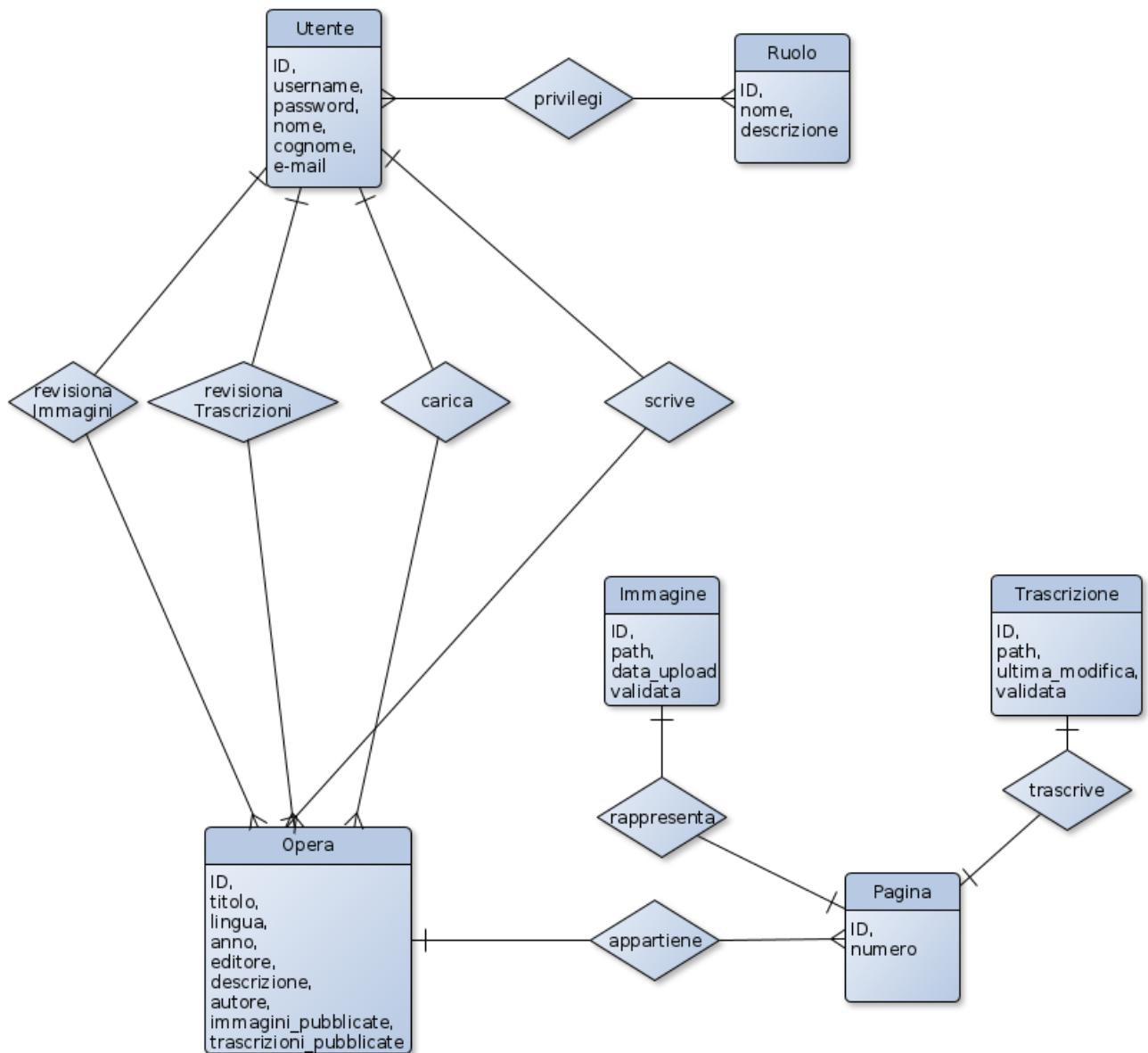
2.4.1 Entità

ENTITÀ				
NOME	DESCRIZIONE	CAMPPI	CHIAVI ESTERNE	
Utente	Informazioni su utenti registrati al sistema (comprende amministratori, revisori, trascrittori ed acquisitori)	ID nome cognome username password e-mail	NULL	
Ruolo	Contiene i ruoli definiti dalla logica di business del sistema per definire l'accesso Alle varie parti dell'applicazione da parte di un utente registrato	ID nome descrizione	NULL	
Immagine	L'entità contiene i metadati delle immagini caricate nel sistema	ID path data_upload validata	NULL	
Trascrizione	L'entità contiene le informazioni sulle trascrizioni di una pagina	ID path ultima_modifica validata	NULL	
Pagina	identifica le pagine di cui le opere sono costituite	ID numero	opera → Opera(ID)	
Opera	indica l'oggetto principale della biblioteca e ne colleziona tutti gli attributi	ID titolo lingua autore anno editore descrizione pubblicata numero_pagine	Acquisitore → Utente(id) Trascrittore → Utente(id)	

2.4.2 Relazioni

RELAZIONI				
NOME	DESCRIZIONE	CAMPPI	ENTITÀ 1	ENTITÀ 2
Privilegi	attribuisce classi di accesso all'utente	NULL	Utente <i>Molti</i>	Ruolo <i>Molti</i>
Appartiene	relazione di appartenenza di una pagina ad un'opera	NULL	Pagina <i>Uno</i>	Opera <i>Molti</i>
Revisionsimmagini	revisione di validazione delle acquisizioni di un'opera	NULL	Utente <i>Uno</i>	Opera <i>Molti</i>
Carica	caricamento di un'immagine da parte di un'acquisitore	NULL	Utente <i>Uno</i>	Opera <i>Molti</i>
RevisioneTrascrizioni	revisione di validazione delle trascrizioni di un'opera	NULL	Utente <i>Uno</i>	Opera <i>Molti</i>
Scrive	scrittura delle trascrizioni di un'opera da parte di un Trascrittore	NULL	Utente <i>Uno</i>	Opera <i>Molti</i>
Rappresenta	relazione di appartenenza di una immagine ad una pagina	NULL	Immagine <i>Uno</i>	Pagina <i>Uno</i>
Trascrive	relazione di appartenenza di una trascrizione ad una pagina	NULL	Trascrizione <i>Uno</i>	Pagina <i>Uno</i>

2.4.3 Diagramma ER



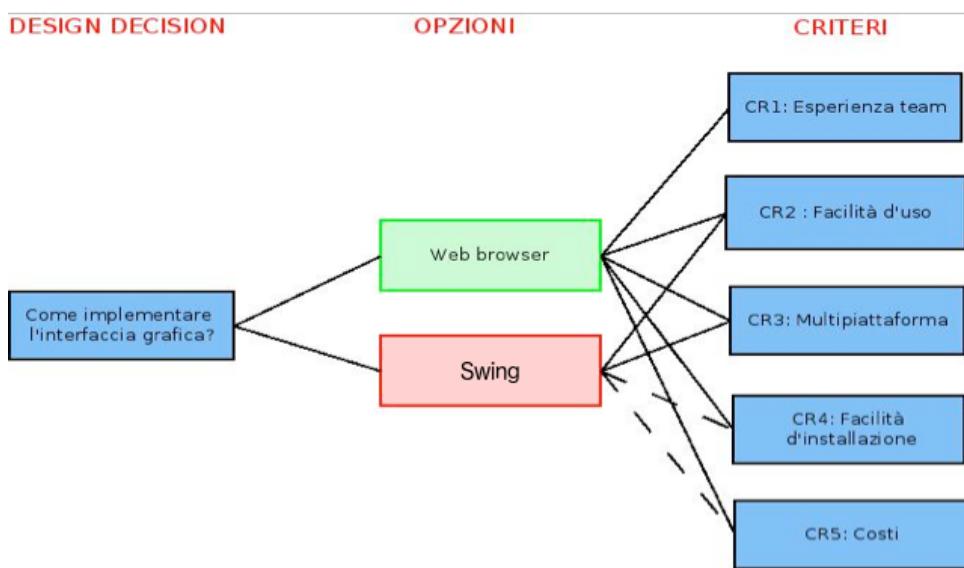
2.5 DESIGN DECISIONS

Nell'ambito del progetto, tre sono state le scelte di design principali sulle quali il team si è soffermato: implementazione dell'interfaccia grafica, template engine ed editor TEI. Di seguito, una descrizione schematica del processo di decisione. Negli schemi, il box a sinistra rappresenta la domanda di design, i box centrali rappresentano le opzioni individuate ed i box a destra rappresentano i criteri di valutazione adottati. Con il colore verde viene evidenziato il box dell'opzione scelta.

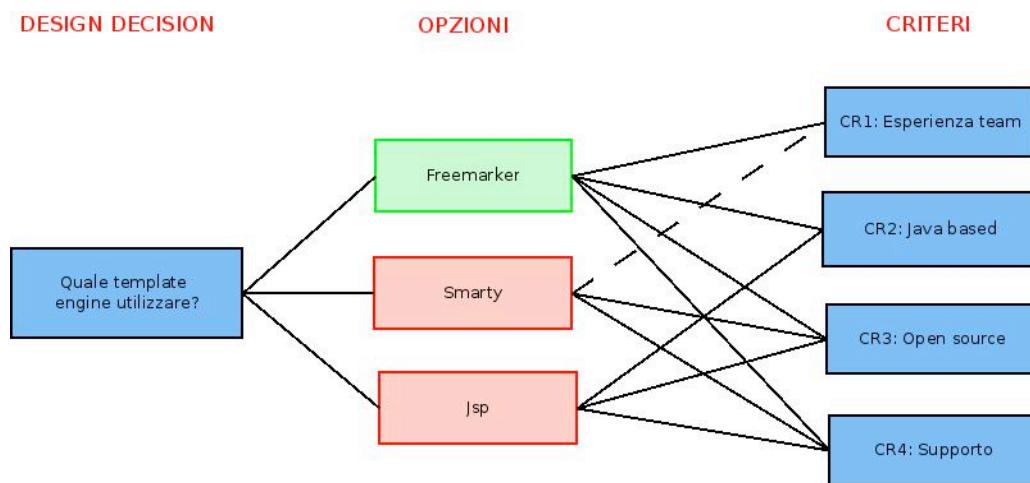
Inoltre, le frecce indicano come il criterio venga soddisfatto dall'opzione da cui la freccia parte: freccia piena -> soddisfatto pienamente, freccia tratteggiata -> soddisfatto parzialmente, nessuna freccia -> non soddisfatto

1. Per quanto riguarda la scelta su "Come implementare l'interfaccia grafica", è stato considerato il fatto che in un'applicazione web based non è necessario installare alcun software sul client, differentemente dall'interfaccia swing, rendendo l'utilizzo del sistema immediato a tutti.

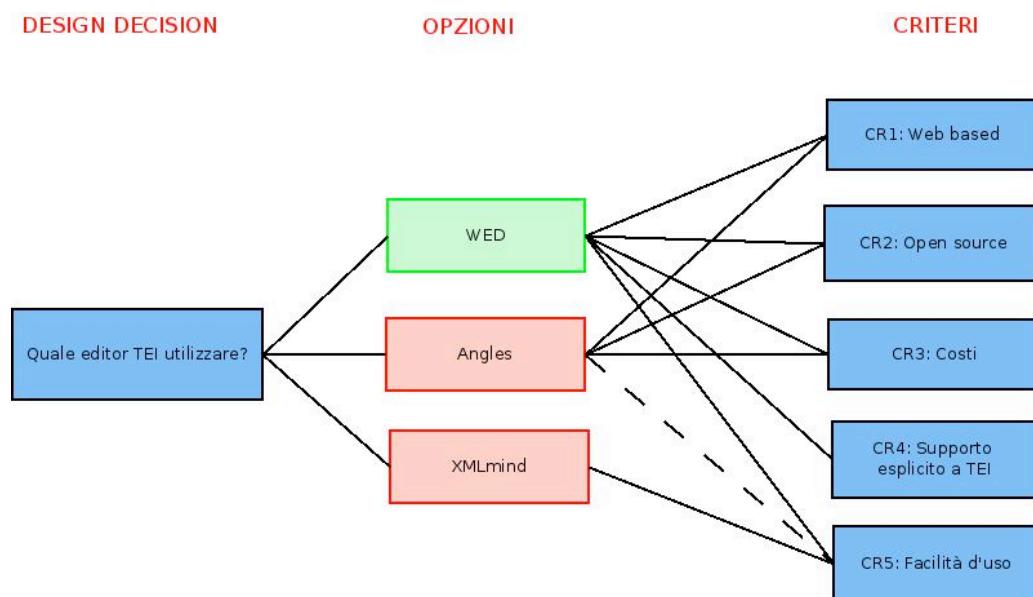
Inoltre, un'applicazione web rientra fra l'expertise del team, è multiplattaforma e ci permette di applicare pattern ben noti nelle applicazioni web.



2. Fra le varie opzioni per quanto riguarda “Quale templateengine utilizzare?”, la scelta ricadeva in particolare tra Freemarker e JSP. Il criterio che ha determinato la scelta difreemarker è il fatto che questo tool risulta già familiare al team di sviluppo.



3. Per quanto riguarda la scelta su “quale editor TEI utilizzare”, si è preso in considerazione il fatto che possa essere facilmente integrabile all’interno del browser (importando una libreria javascript) e la completezza del supporto dell’editor rispetto allo standard TEI.



2.6 PROBLEMI NOTI

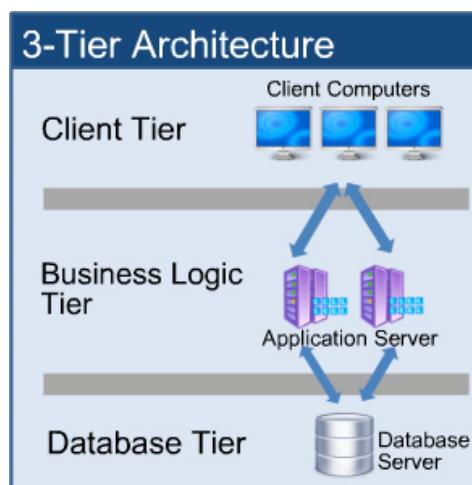
Nella fase implementativa si sono trovate grosse difficoltà nell'integrare gli editor TEI considerati nelle decisioni.

Il team ha dunque valutato un quarto editor (TEA), contattando personalmente anche gli sviluppatori per riuscire nell'intento di integrarlo nell'applicazione. Tuttavia, il progetto TEA è ancora in una fase di sviluppo e non ancora pronto per l'utilizzo all'interno di un'applicazione con TEI.

Si è quindi scelto di implementare un parser TEI in grado di riconoscere i tag dello standard e un'area di testo per l'immissione del contenuto.

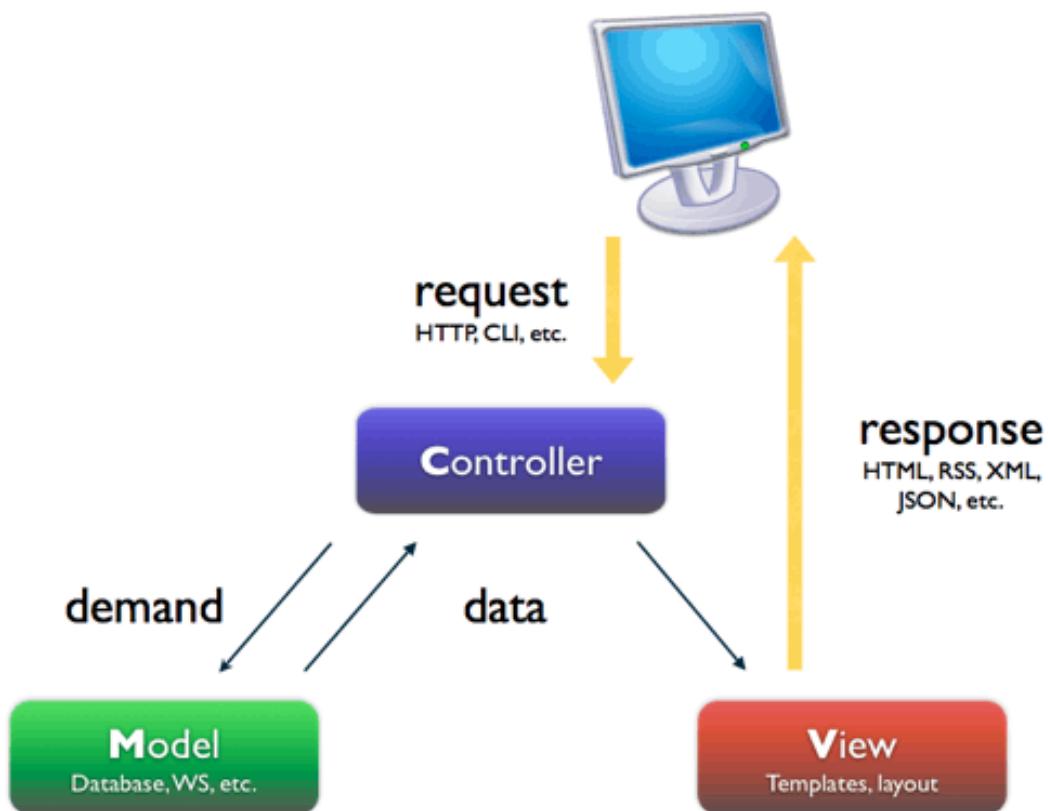
2.7 DESIGN PATTERNS

L'architettura adottata è un'architettura 3-tier classica con il livello di presentazione, di logica applicativa e di accesso ai dati. Per realizzarla sono stati adottati i design pattern MVC e DAO.



2.7.1 Model-View-Controller

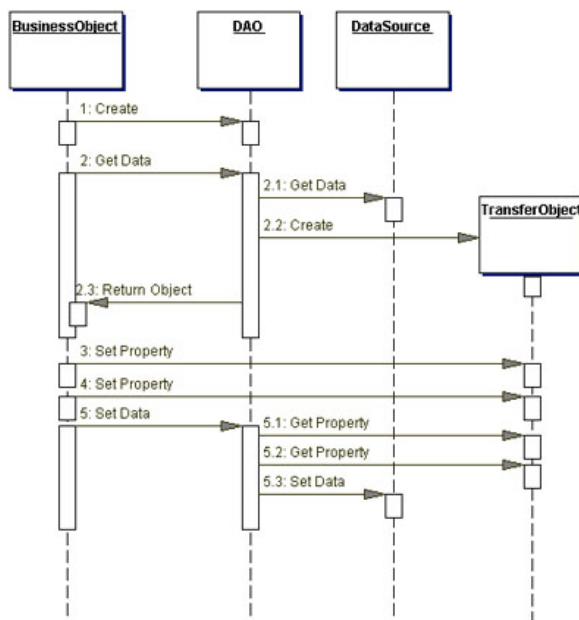
Il pattern MVC ci permette di organizzare in modo snello ed adattabile il livello di presentazione andando a definire una parte MODEL che si occupa solo della gestione dei dati del dominio, una parte VIEW che viene implementata dalle pagine web e definisce diverse rappresentazioni di output di informazioni ed infine una parte CONTROLLER che accetta input e li converte in comandi per il MODEL e/o per la VISTA.



2.7.2 Data-Access-Object

Per la gestione della persistenza dei dati si è invece adottato il pattern Data-Access-Object che fa parte dei Core J2EE Patterns. Questo pattern fornisce operazioni sui dati senza esporre dettagli della base di dati mappando chiamate dell'applicazione al livello di persistenza. Questo ci permette di realizzare l'information hiding dei dettagli della memorizzazione dei dati. Infatti, con l'introduzione dell'interfaccia DAO, i cambiamenti alla logica di business potranno fare comunque affidamento sulla stessa interfaccia DAO, mentre cambiamenti alla logica di persistenza non coinvolgeranno i client dell'interfaccia se questa rimarrà correttamente implementata.

Nello schema seguente una sintesi del processo di accesso ai dati tramite DAO.



Il BusinessObject rappresenta il client che richiede accesso al database per memorizzare e ottenere dati. Il DataAccessObject astrae l'implementazione dell'accesso ai dati per garantire accesso trasparente dal BusinessObject. DataSource rappresenta il database o altra forma di sorgente dati (e.g. file system, XML repository). Il TransferObject rappresenta un oggetto usato come contenitore di dati. Esso viene usato dal DataAccessObject per ritornare dati al client o per aggiornare dati nel DataSource.

3 SOFTWARE DESIGN

3.1 CLASS DIAGRAM

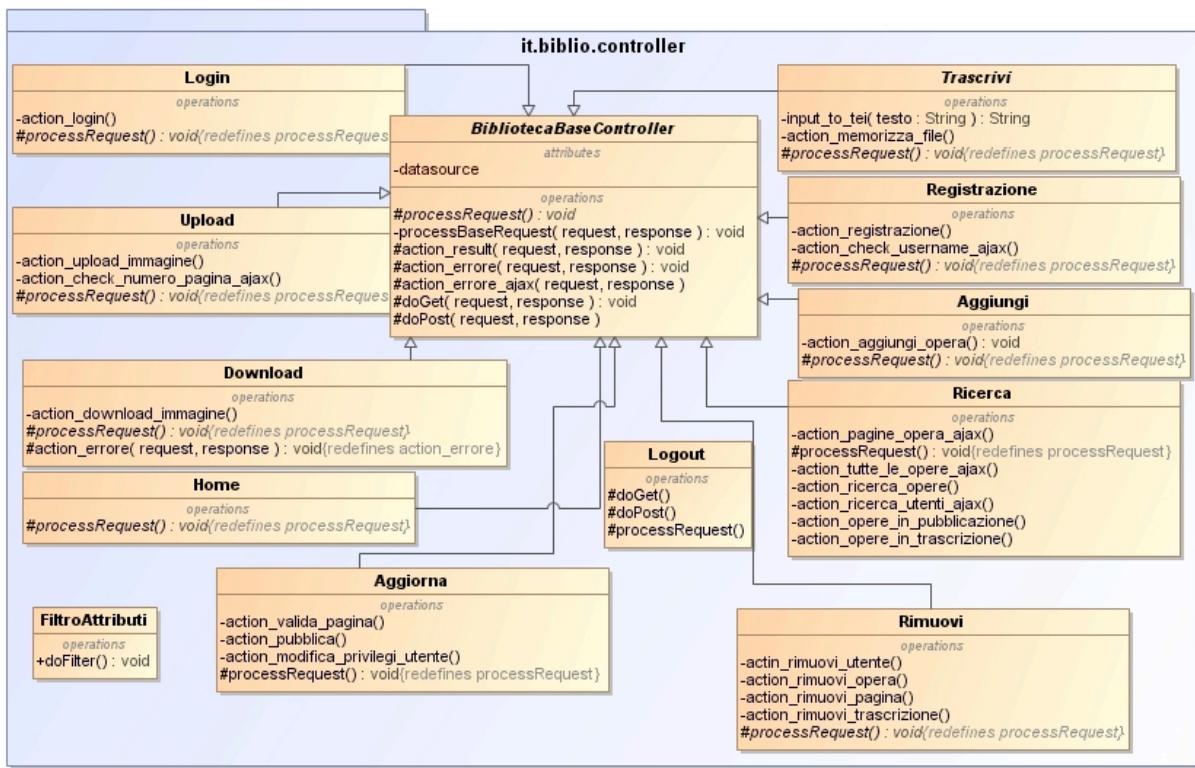
3.1.1 Package Controller

Questo package rappresenta la componente Controller del pattern MVC. Esso contiene infatti tutte le Servlet, ovvero il codice che può essere direttamente chiamato dal browser, che eseguono le funzionalità di business dell'applicazione.

La classe principale è la classe astratta **BibliotecaBaseController**, che viene estesa da tutte le servlet che abbiano la necessità di accedere alla base di dati. Infatti, tale classe oltre ad estendere la classe HttpServlet del core J2EE ed implementarne alcuni metodi ereditati fondamentali (doGet, doPost), possiede la reference alla risorsa di tipo database definita nel deployment descriptor dell'applicazione ed effettua l'inizializzazione della connessione al database rendendola disponibile come attributo della HttpServletRequest alle classi che la estendono. Essa definisce inoltre un metodo standard per la renderizzazione di una pagina di errore ed anche un metodo per la generazione automatica della pagina di risposta. Il metodo *processRequest* è astratto e viene ridefinito dalle classi figlie come punto di accesso per iniziare la computazione specifica.

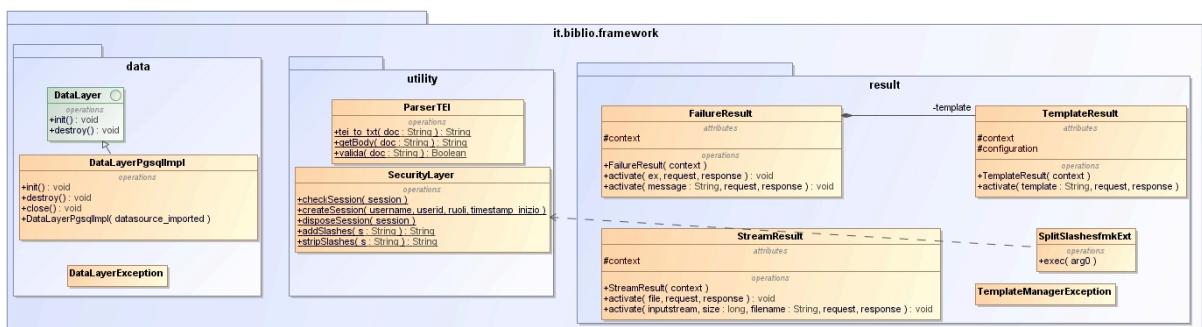
La classe **Logout** è l'unica a non estendere BibliotecaBaseController in quanto altro non fa che invalidare la sessione. La classe astratta si serve delle classi del package framework.result per la generazione delle pagine dinamiche e di conseguenza anche le classi figlie. Infine, è stata definita anche una classe filtro (**FiltroAttributi**) in ingresso che analizza tutte le richieste ed inizializza alcune variabili usate dalle servlet.

Le Servlet hanno un metodo privato per ogni azione che sono chiamate a fare dal client. La logica di accesso a tali operazioni viene implementata al loro interno.



3.1.2 Package Framework

Il package rappresenta la base dello sviluppo di un'applicazione WEB che utilizza il pattern MVC. Infatti, si potranno usare le classi del package result per la generazione della grafica (VIEW), le classi del package data come base per gli oggetti DAO (MODEL) e le classi del package util per operazioni comuni a tutta l'applicazione.



3.1.2.1 Sottopackage result

Tale package dispone di tutte le classi che si interfacciano direttamente con la libreria FreeMarker per la gestione dei template. Esse mettono a disposizione del controller, dunque, funzioni per la generazione delle pagine dinamiche.

La classe **TemplateResult** fornisce funzioni per la generazione delle pagine HTML.

La classe **FailureResult** fornisce la funzione per la generazione della pagine d'errore dinamica.

La classe **StreamResult** fornisce funzioni per l'invio di file e flussi di byte in generale.

La classe **SplitSlashesFmkExt** estende le funzionalità del template manager FreeMarker per poter richiamare la funzione di logica di rappresentazione nella quale vengono tolti i caratteri di slash dalle stringhe direttamente nel codice HTML del template.

Infine, è stata definita l'eccezione **TemplateManagerException** generata ogniqualvolta si riscontra un errore nella logica del template.

3.1.2.2 Sottopackage util

Tale package dispone delle classi di utilità che vengono utilizzate trasversalmente da tutta l'applicazione. Le seguenti sono state individuate come classi di utilità:

SecurityLayer: classe che fornisce funzioni per la sicurezza dell'applicazione come la gestione delle sessioni e prevenzione da SQL Injection.

PaserTEI: classe usata per analizzare testi TEI.

3.1.2.3 Sottopackage data

Tale package fornisce delle classi basilare da cui poter creare classi di tipo risorsa da usare per la gestione delle connessioni con il database.

L'interfaccia **DataLayer** definisce le operazioni base da effettuare per un oggetto di tipo AutoCloseable (come una risorsa).

La classe **DataLayerPgsqllmpl** realizza tale interfaccia.

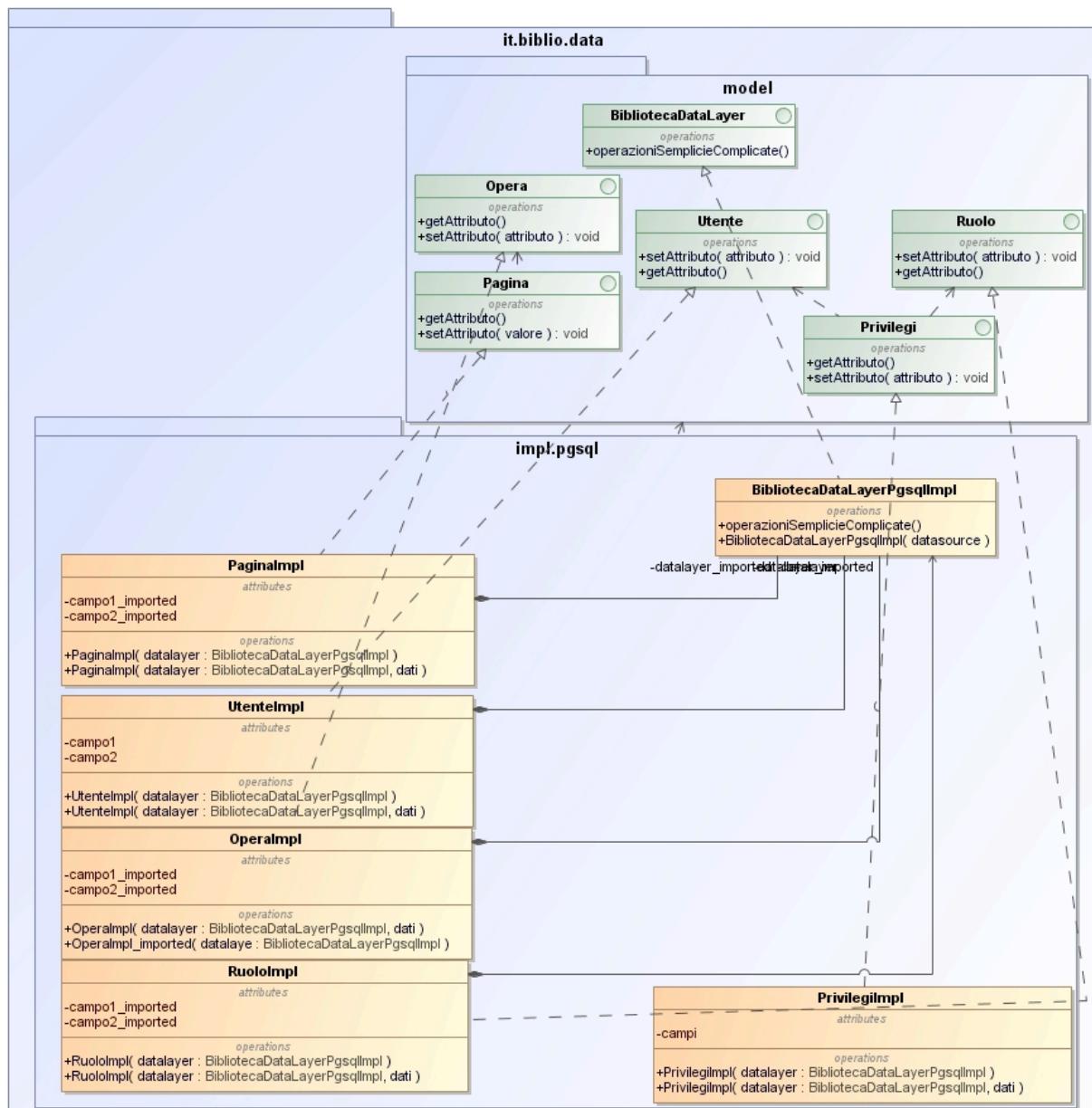
È stata inoltre definita l'eccezione DataLayerException da sollevare ogni volta che ci saranno errori nella gestione del livello dei dati.

3.1.3 Package data

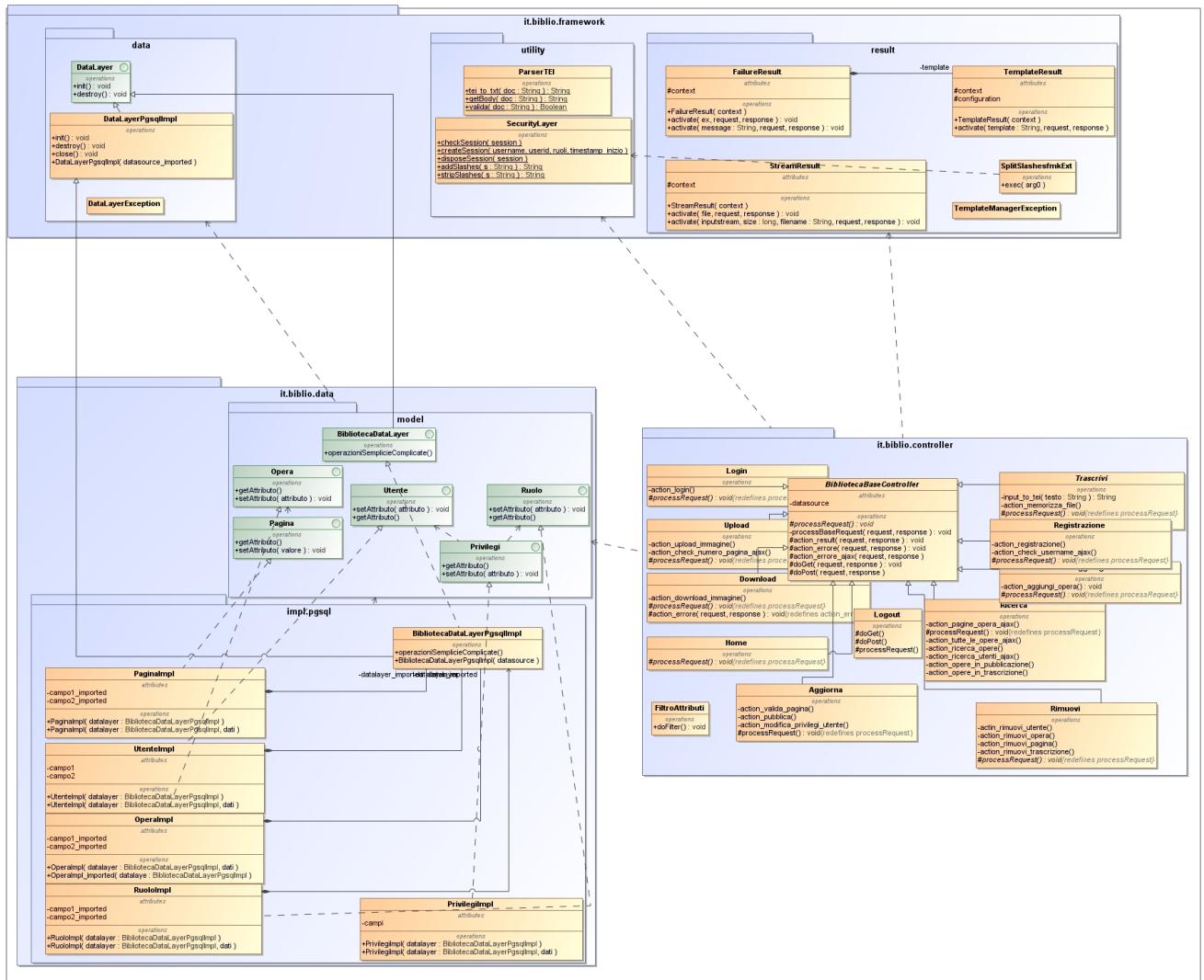
Tale package contiene tutte le interfacce e le classi che realizzano lo strato di persistenza dell'applicazione.

Le interfacce vengono posizionate nel sottopackage model mentre le realizzazioni sono in implpgsql. Infatti, le interfacce fungono da contratto verso i client (Controller) mentre l'implementazione può essere diversa per ogni tipo di database utilizzato. Infatti se volessimo un giorno utilizzare Mysql, basterà creare un sottopackage chiamato impl.mysql ed implementare le interfacce di model in maniera dedicata a quel DB senza dover cambiare le classi dell'applicazione che utilizzano il modello dei dati (o quantomeno cambiare solo l'invocazione del costruttore della classe che implementa BibliotecaDataLayer che nel nostro caso è una sola riga di codice posizionata di proposito in it.biblio.controller.BibliotecaBaseController).

Ad ogni interfaccia corrisponde un entità di business alla quale a sua volta corrisponde una tabella del database relazionale. I metodi get e set si riferiscono agli attributi delle Entità specificate nel diagramma Entità/Relazioni.



3.2 CLASS DIAGRAM



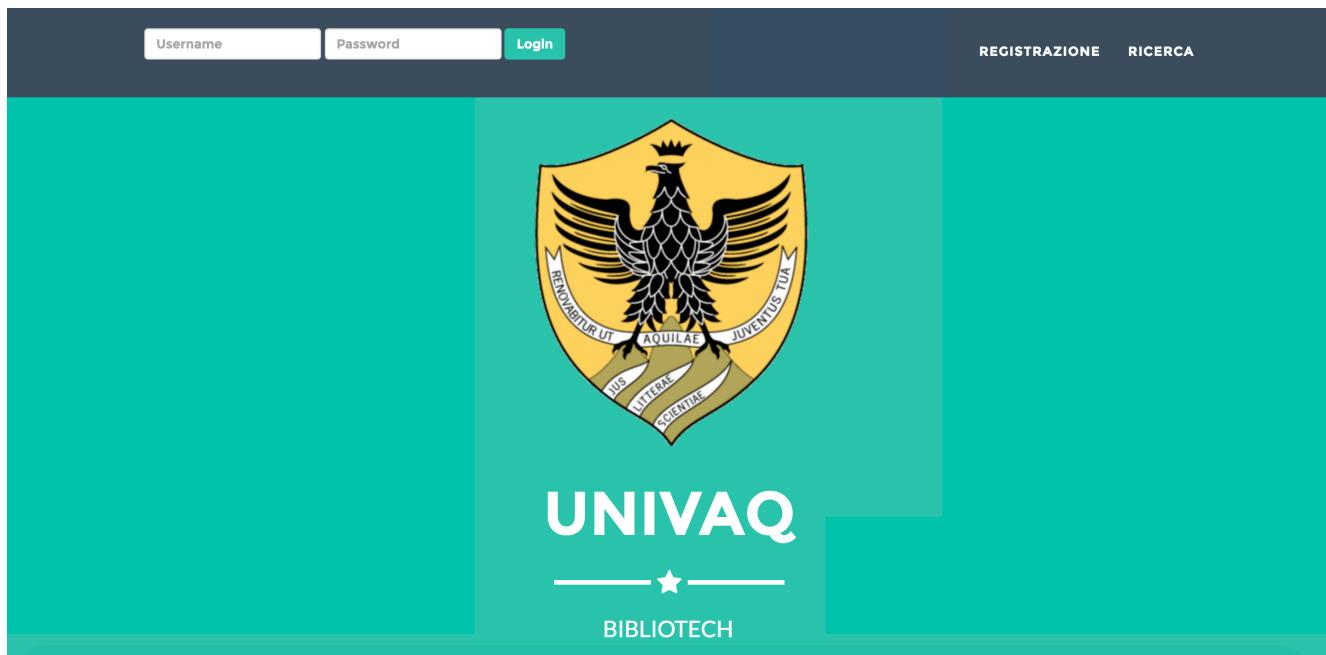
Da questo diagramma si evince che le relazioni dei vari package rispettano il pattern MVC.
Il controller dipende solo dalle interfacce presenti in model per la gestione dei dati e non dipende dall'implementazione.
Le classi all'interno del package controller sfruttano inoltre quelle di `framework.result` e `framework.util`.
Il package `it.biblio.data.model` fa uso del `framework.data` per costruire il datalayer.

3.3 PROBLEMI NOTI

Poiché trattasi di oggetti POJO che mappano direttamente le entità del database e poiché si è limitati da una licenza demo di MagicDraw (il tool utilizzato per la modellazione) non sono stati esplicitati i campi e le operazioni get e set. Lo stesso vale per la classe BibliotecaDataLayerPgsqlImpl. Per questo può essere usata la documentazione generata da Javadoc nell'apposita cartella.

4 SCREENSHOTS

4.1 HOMEPAGE



Nella schermata homepage si trova la form per il login e scorrendo in basso o cliccando i link nella navbar, è possibile vedere la form per la registrazione e per la ricerca.

4.2 REGISTRAZIONE

REGISTRATI



Nome

Marco

Cognome

D'Ettorre

Email

acquisitore@email.

- Indirizzo email non valido

Username

acquisitore



Password

Invia

Nella form registrazione vengono inseriti i campi relativi alle informazioni sull'utente con controlli sulla correttezza del campo email e controllo sullo username già presente.

Una volta registrati, viene aperta la pagina relativa all'utente base, dove è possibile cercare un'opera e sfogliarla completamente.

4.3 ACCESSO ACQUISITORE

CIAO ACQUISITORE Logout CARICA IMMAGINE RICERCA

CARICA UN'IMMAGINE

Seleziona l'opera da acquisire:

Il Piccolo Principe - Tascabili Bompiani

Numero di pagina inserita:

1

Scegli file il-piccolo-principe.jpg

Carica

Il caricamento di un'immagine può essere effettuato solo da un acquirente che tramite una select seleziona l'opera di cui vuole fare l'upload. Viene effettuato un controllo sul numero della pagina: se è già stata inserita non sarà possibile caricare l'immagine per quella pagina.

4.4 ACCESSO REVISORE ACQUISIZIONE

CIAO REVISOREACQUISIZIONE Logout DA CONVALIDARE RICERCA

OPERE DA CONVALIDARE

1 2

#	ID opera	Titolo opera	Descrizione opera	Numero pagine
1	53	Il Piccolo Principe	Tutti i grandi sono stati bambini una volta	124
2	52	Segreti e misteri	Misteri antichi	3
3	6	Sogno di un attimo di primavera	Poema tragico, tragicissimo	550

Seleziona un'opera dalla lista di quelle con acquisizione da convalidare



CONVALIDA

PAGINA N.

1



Previous

Next



"H�ngrebbie mettiti uno sopra l'altro."

Ma nonno ragionava.

"Primo di crescere, i baobab sono piccoli."

"Estarà? Ma perché vedi che le tue pycore mangiano i piccoli baobab?"

"No", e ovvia", mi rispose come se la risposta fosse evidente. E mi sfiorai con tutto me stesso di

venera e capo del problema.

Infatti, sul pianeta del piccolo principe, come su tutti gli altri pianeti, ci sono delle erbe buone e

delle cattive. Le erbe buone sono dei piatti sani di gheie, ma le cattive sono di maledizione. Ma le cattive sono insatiate. Dicono maledizioni orribili. Infine non hanno che un solo segnale di

riconoscimento. Allora si strappano e fanno spuntare bellissimi e letti baobab nascosti verso il

sole. Se sono insaziati di riconoscere e di morire, si può lasciar crescere come vogliono.

Ma se si tratta di una pianta cattiva, lo si deve strappare subito, appena la si riconosce.

Sal prima del piccolo principe c'erano dei simboli veri e seri di baobab. Avevano

infestato il suolo. Il baobab è una pianta di cui non ci si riuscì più a sbucare, se si arrivò

troppo tardi. Occupa tutto il pianeta. E se il pianeta è troppo piccolo e i baobab troppo numerosi,

lo fanno esplosione.

"E' una questione di disciplina", mi disse poi il piccolo principe. "Ogni mattina, dopo che ci si è lavati, bisogna pulire per bene tutti il pianeta. Bisogna colpire salvo i baobab appena li si

distingue dai rossi, e a cui semiglano molto quando sono piccoli. È un lavoro noioso,

ma è facile."

Un giorno mi consigliò di fare un bel disegno per spiegare meglio questa idea

al pubblico dei suoi paesi.

"È una storia di viaggio", mi disse. "Questo consiglio gli potrà

tenere alto. A volte rimandare il proprio lavoro non comporta

conseguenze, ma se si tratta dei baobab è sempre una catastrofe. Sono state una volta su un pianeta abitato da un

pugno. Non aveva baobab a tre piccoli affusti..."

Convalida

Rimuovi

Il revisore acquisizione può sfogliare le varie pagine inserite e convalidare o non convalidare l'immagine. Se viene convalidata l'immagine non può essere più invalidata.

4.5 TRASCRIZIONE

CIAO TRASCRITTORE [Logout](#)

SCRIVI LA PAGINA

TRASCRIVI LA PAGINA N.
1

[Previous](#) [Next](#)



Bisognerebbe metterli uno sopra l'altro.
Ma osservi saggiamente:
Prima di crescere i baobab sono piccoli.

< /p >

Invia

Il trascrittore una volta selezionata l'opera da trascrivere, potrà scorrere tra le pagine e riempire il campo testo con il codice TEI relativo solo al body. Una volta terminato il processo per ogni pagina, farà click sul bottone invia che salverà la trascrizione nel sistema.

4.6 REVISIONE TRASCRIZIONE



SCRIVI LA PAGINA

TRASCRIVI LA PAGINA N.

1

Previous

Next



```
<?xml version="1.0" encoding="UTF-8"?><?xml-model
href="http://www.tei-
c.org/release/xml/tei/custom/schema/relaxng/tei_lite.rng"
type="application/xml"
schematypens="http://relaxng.org/ns/structure/1.0"?><?
xml-model href="http://www.tei-
c.org/release/xml/tei/custom/schema/relaxng/tei_lite.rng"
type="application/xml"schematypens="http://purl.oclc.org/
dsdl/schematron"?><TEI xmlns="http://www.tei-
c.org/ns/1.0"><teiHeader><fileDesc><titleStmt>
<title>1</title></titleStmt><publicationStmt>
<p>Bontempi</p></publicationStmt><sourceDesc>
<p>libro per bambini </p></sourceDesc></fileDesc>
</teiHeader><text><body><p>Bisognerebbe metterli uno
sopra l'altro.
```

Convalida

Rimuovi

Il revisore trascrizione, una volta selezionata l'opera da validare, potrà scorrere le pagine e convalidare o rigettare la pagina già trascritta.

4.7 ACCESSO ADMIN

CIAO FRANCISKITTU Logout OPERE OPERE DA PUBBLICARE UTENTI INSERISCI OPERA

OPERE IN PUBBLICAZIONE

ACQUISIZIONE



1

#	ID	Titolo opera	Editore	Rimuovi	Pubblica
1	53	Il Piccolo Principe	Tascabili Bompiani	<button>Rimuovi</button>	<button>Pubblica</button>

l'amministratore può rimuovere o pubblicare le opere in stato di “acquisite” o “trascritte” (come nello screenshot sopra), oppure può gestire gli utenti (screenshot sotto), inserire nuove opere e visualizzarne i dettagli.

GESTISCI RUOLO DI MARCO D'ETTORRE



ID ruolo	Nome ruolo	Descrizione ruolo	Applica ruolo
1	admin	gestore generale del sistema	<button>Applica</button>
2	acquisitore	acquisizione-digitalizzazione delle immagini	<button>Applica</button>
3	trascrittore	trascrizione del testo di una pagina	<button>Applica</button>
4	revisore acquisizioni	revisione e verifica correttezza acquisizione	<button>Applica</button>
5	revisore trascrizioni	revisione e validazione della trascrizione delle pagine	<button>Applica</button>

4.8 ACCESSO DA DISPOSITIVO MOBILE

The screenshots illustrate a mobile application's user interface. The left screen shows a form for uploading an image, with a title 'CARICA UN'IMMAGINE' and a dropdown menu showing 'Sogno di un attimo di primavera - Treves'. The right screen shows a page titled 'SCRIVI LA PAGINA' with a section 'TRASCRIVI LA PAGINA N. 1'. It includes a text area with a story about baobabs, a 'Testo' button, and a code snippet below it.

Screenshot 1: CARICA UN'IMMAGINE

Selezione l'opera da acquisire:

Sogno di un attimo di primavera - Treves

Numero di pagina inserita

Scegli file Nessun file selezionato

Carica

Screenshot 2: SCRIVI LA PAGINA

TRASCRIVI LA PAGINA N. 1

Bisognerebbe metterli uno sopra l'altro.
Ma osservò saggiamente:
"Prima di crescere, i baobab sono piccoli."
"Ecco! Ma perché vedi che le tue pecore mangiano i piccoli baobab?"
"Be', è ovvio", mi rispose come se la risposta fosse evidente. E mi sforzai con tutto me stesso di venire a capo del problema.
Infatti, sul pianeta del piccolo principe, come su tutti gli altri pianeti, ci sono delle erbe buone e delle erbe cattive. Di conseguenza ci sono dei buoni semi di erbe buone e dei cattivi semi di erbe cattive. Ma i semi sono invisibili. Dormono nascosti sotterranei finché non viene loro voglia di rivesgliersi. Allora si stendono e fanno sputare bellissimi e innocui ramoscelli verso il sole. Se sono ramoscelli di ravanello o di rosolio, li si può lasciar crescere come vogliono. Ma se si tratta di una pianta cattiva, la si deve strappare subito, appena la si riconosce. Sul pianeta del piccolo principe c'erano dei terribili semi: erano semi di baobab. Avevano infestato il suolo. Il baobab è una pianta di cui non ci si riesce più a sbarrare, se si arriva troppo tardi. Occupa tutto il pianeta. E se il pianeta è troppo piccolo e i baobab troppo numerosi, la flora esplode.

Testo

<p>Bisognerebbe metterli uno sopra l'altro.
Ma osservò saggiamente:
Prima di crescere i baobab sono piccoli.
</p>

Tutte le funzionalità del sito sono accessibili anche da dispositivo mobile.