Università degli Studi di Salerno

Corso di Ingegneria del Software

La Cantinella del Bonsignore
Object Design Document
Versione 1.2

LOGO PROGETTO

Data: 18/12/2017

Partecipanti

Nome	Matricola
Graziuso Catello	0512103726
Fornaro Daniele	0512103864
Ciano Francesco	0512103726

Indice

1. Introduzione

1.1 Object Design Trade-offs

In seguito alla stesura del RAD e SSD, in cui si è descritto in linea di massima quello che sarà il sistema e i nostri obiettivi, si procede con la stesura di un nuovo modello capace di integrare in modo coerente e preciso tutte le funzionalità individuate.

Si definiranno le interfacce delle classi, le operazioni, i tipi, gli argomenti e la signature dei sottosistemi definiti nel System Design.

Comprensibilità vs Tempo:

Nonostante la comprensibilità del codice faciliti la fase di testing e di future modifiche, lo sviluppo del sistema dovrà rispettare delle tempistiche prestabilite.
Per tale motivo, si cercherà di essere più chiari possibile nel codice, aggiungendo commenti che ne semplificano la comprensione.
Si cercherà quindi di produrre un software di alta qualità nei tempi prestabiliti.

Prestazioni vs Costi:

Essendo il progetto sprovvisto di budget, al fine di mantenere prestazioni elevate, per alcune funzionalità verranno utilizzati dei template open source esterni in particolare Bootstrap.

Interfaccia vs Usabilità:

Grazie all'ausilio di pattern pre-esistenti, l'interfaccia grafica è stata realizzata in modo da essere user friendly.
Pulsanti e form sono disposti in maniera da rendere semplice l'utilizzo.

Sicurezza vs Efficienza:

La sicurezza rappresenta uno dei requisiti che intendiamo garantire.
Nonostante ciò, come su scritto, rispettare la deadline del progetto è fondamentale, perciò ci limiteremo ad implementare sistemi di sicurezza basati su username e password degli utenti e filtri per evitare di accedere a sezioni non permesse.

1.2 Linee Guida per la Documentazione delle Interfacce

Gli sviluppatori dovranno seguire alcune linee guida per la scrittura del codice:

Naming Convention

- E' buona norma utilizzare nomi:
    1. Descrittivi
    2. Pronunciabili
    3. Di uso comune
    4. Lunghezza medio-corta
    5. Utilizzando solo caratteri consentiti (a-z, A-Z, 0-9)

Variabili:

- I nomi delle variabili devono cominciare con una lettera minuscola, e le parole seguenti con la lettera maiuscola. Quest'ultime devono essere dichiarate ad inizio blocco, solamente una per riga e devono essere tutte allineate per facilitare la leggibilità.
  E' inoltre possibile, in alcuni casi, utilizzare il carattere underscore "_".

Esempio: nomeAzienda oppure nome_azienda

Metodi:

● I nomi dei metodi devono cominciare con una lettera minuscola, e le parole seguenti con la lettera maiuscola. Il nome del metodo tipicamente consiste di un verbo che identifica una azione, seguito dal nome di un oggetto.
Esempio: getNomeAzienda(), setNomeAzienda

● I commenti dei metodi devono essere raggruppati in base alla loro funzionalità, la descrizione dei metodi deve apparire prima di ogni dichiarazione di metodo, e deve descriverne lo scopo. Deve includere anche informazioni sugli argomenti, sul valore di ritorno, e se applicabile, sulle eccezioni.

Classi e pagine :

● I nomi delle classi e delle pagine devono cominciare con una lettera maiuscola, e anche le parole seguenti all'interno del nome devono cominciare con una lettera maiuscola. I nomi di quest'ultime devono fornire informazioni sul loro scopo.
Esempio: RispostaMultipla.java

Ogni file sorgente java contiene una singola classe e dev'essere strutturato in un determinato modo:

● L'istruzione include che permette di importare all'interno della classe gli altri oggetti che la classe utilizza.

● La dichiarazione di classe caratterizzata da:

1. Dichiarazione della classe pubblica
2. Dichiarazioni di costanti
3. Dichiarazioni di variabili di classe
4. Dichiarazioni di variabili d'istanza
5. Costruttore
6. Commento e dichiarazione metodi.

ESEMPIO:

```
#include:XXXXX
        public class Vino
        {
                private int idVino;       //identificativo univoco
                private String Nome;   //nome del vino

                /*    costruttore
                    @Param idVino identificativo univoco
                    @Param cfUser nome del vino
                */
                public Vino(int idVino, String cfUser )
                {
                        this.idVino=idVino;
                        this.cfUser = cfUser;
                }
                /*    costruttore
```

```
        restituisce l'id del prodotto
        @return Int id del Prodotto
    */
    //restituisce l'id del prodotto
    //@return Int id del Prodotto
    public int getIdVino()
    {
        return idVino;
    }

    //Setta il nome del Vino
    //@param string nome del Vino
    public void setNome(String nome)
    {
        this.nome = nome;
    }
}
```

1.3 Definizioni, acronimi e abbreviazioni

Acronimi:

- RAD: Requirements Analysis Document
- SDD: System Design Document
- ODD: Object Design Document

Abbreviazioni:

- DB: DataBase

1.4 Riferimenti

- C. Ghezzi M. Jazayeri D. Mandrioli, Software engineering, Foundations and principles 2nd edition, 2004
- Documento SDD del progetto LaCantinella
- Documento RAD del progetto LaCantinella

2. Packages

La gestione del nostro sistema è suddivisa in tre livelli (three-tier):

- Interface layer
- Application Logic layer
- Storage layer

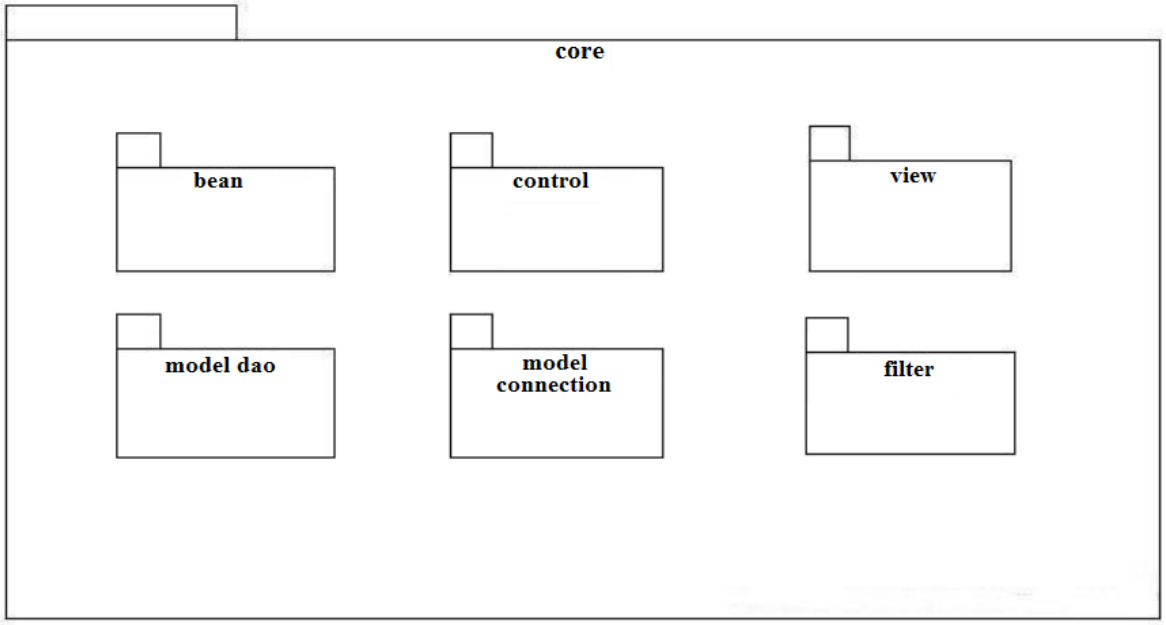
Il package "CantinellaaBonsignore" contiene sottopackage che a loro volta inglobano classi atte alla gestione delle richieste utente. Le classi contenute nel package svolgono il ruolo di gestore logico del sistema.

Interface layer	Rappresenta l'interfaccia del sistema, ed offre la possibilità all'utente di interagire con quest'ultimo, offrendo sia la possibilità di inviare, in input, che di visualizzare, in output, dati.

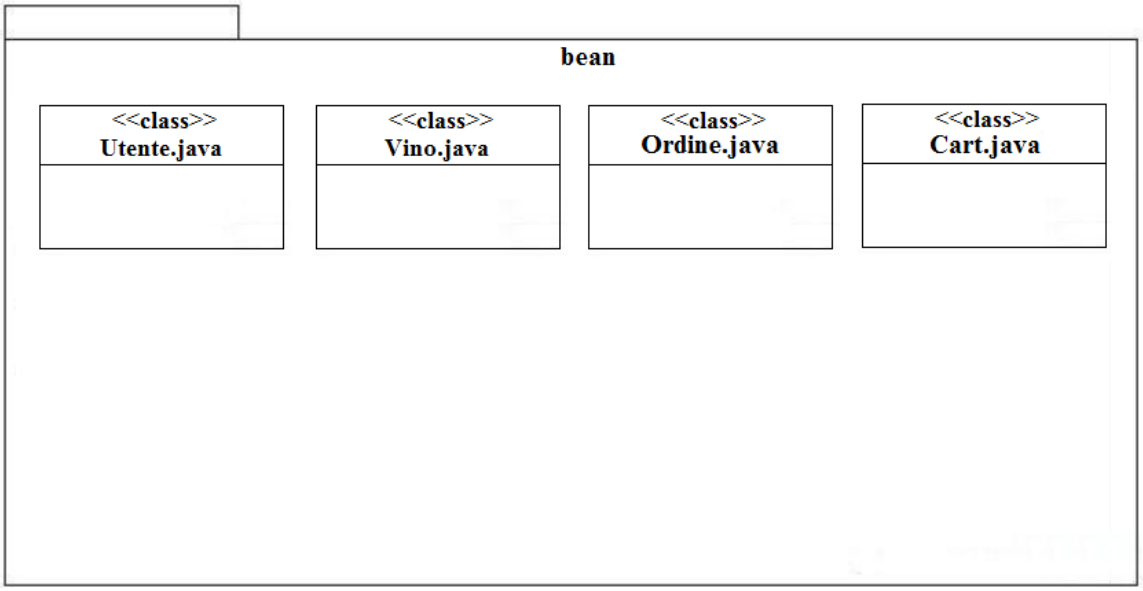
Application Logic layer	Ha il compito di elaborare i dati da inviare al client, e spesso grazie a delle richieste fatte al database, tramite lo Storage Layer, accede ai dati persistenti. Si occupa di varie gestioni quali: 1. Gestione Utenti 2. Gestione Like 3. Gestione Ordini 4. Gestione Carrello 5. Gestione Articolo

Storage layer	Ha il compito di memorizzare i dati sensibili del sistema, utilizzando un DBMS, inoltre riceve le varie richieste dall' Application Logic layer inoltrandole al DBMS e restituendo i dati richiesti.

2.1 Package core

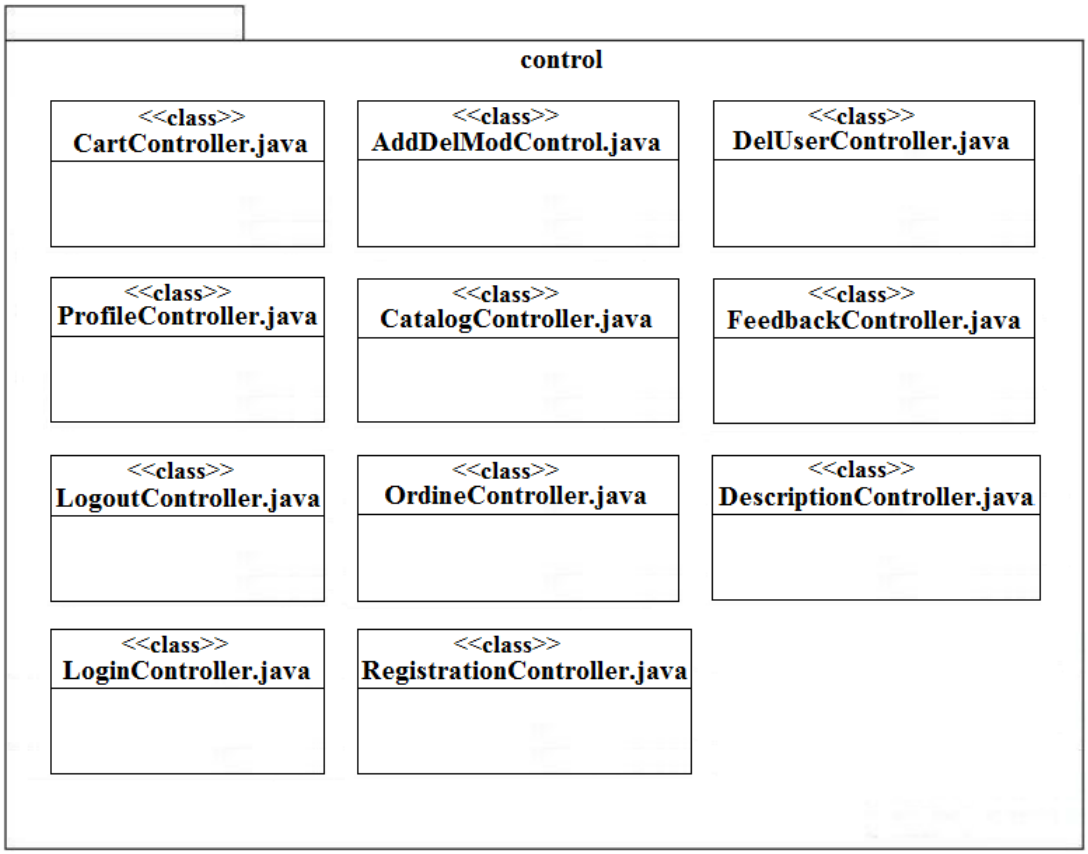


2.1.1 Package bean



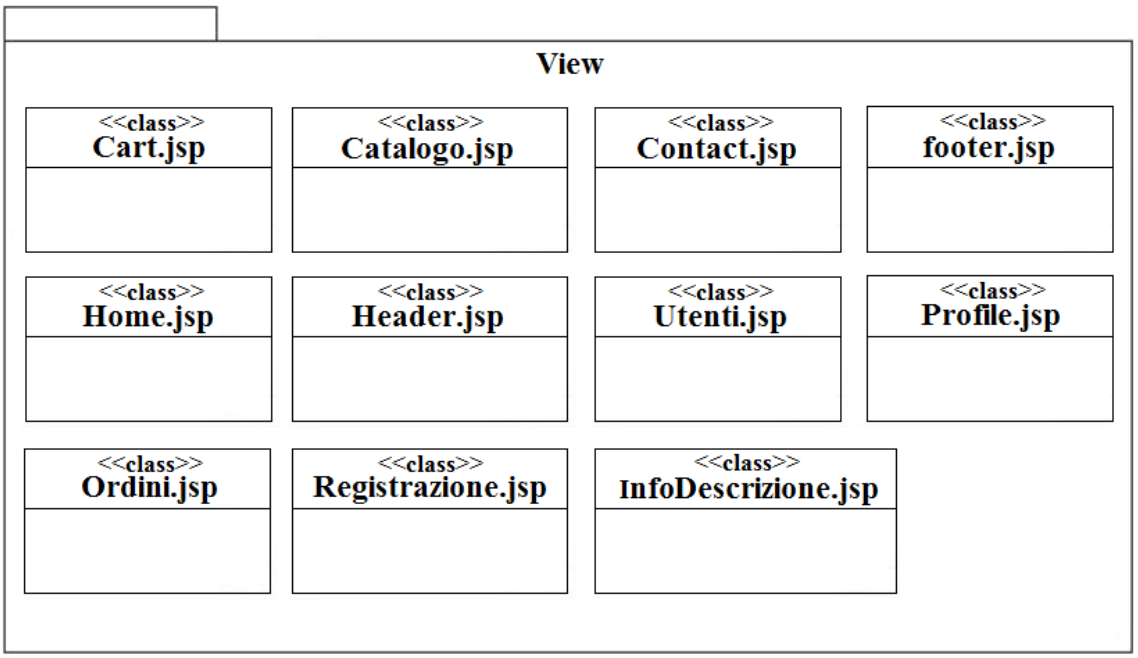
Classe	Descrizione
Utente.java	Descrive l'utente registrato
Vino.java	Descrive il vino venduto
Ordine.java	Descrive l'ordine di vini effettuati da utenti
Cart.java	Descrive il Carrello dell'utente in sessione

2.1.2 Package control



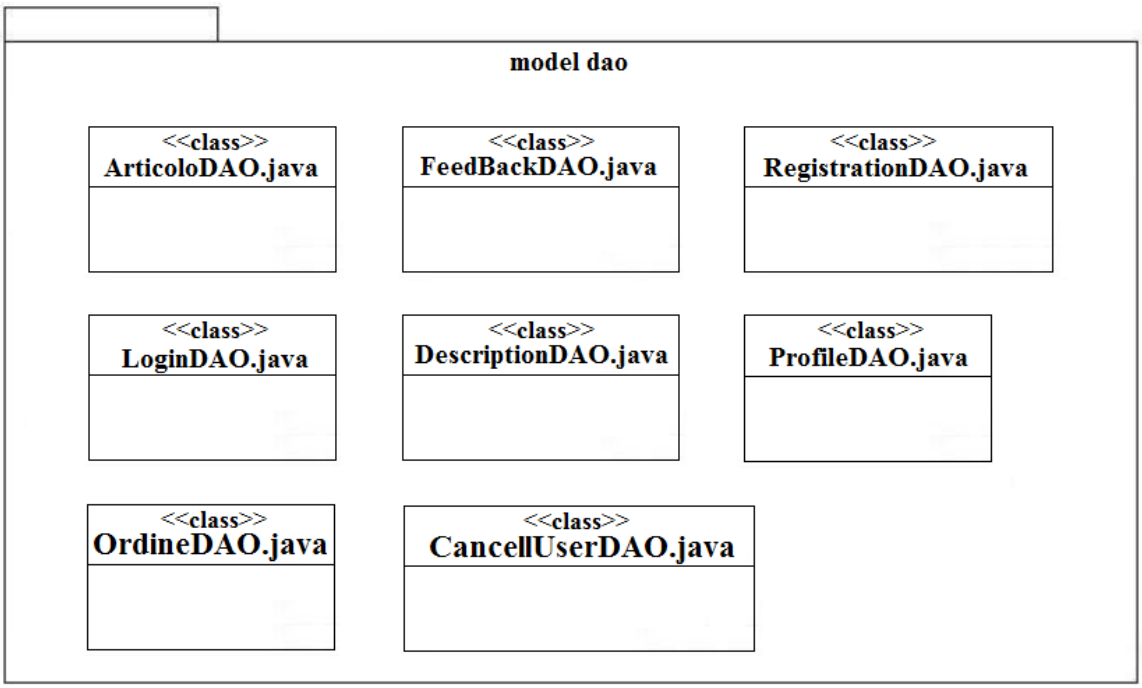
Classe	Descrizione
CartController.java	Controller che permette la creazione un carrello
AddModControl.java	Controller che permette l'inserimento del vino in un carrello
LoginController.java	Controller che permette la gestione del login
LogoutController.java	Controller che permette la gestione del logout
ProfileController.java	Controller per la gestione dei dati del profilo dell'utente
RegitrationControlle.java	Controller che permette la gestione della registrazione
feedbackController.java	Controller che permette la gestione dei feedback dei vini
CatalogController.java	Controller per la visualizzazione del catalogo dei vini
DescriptionControlle.java	Controller per la visualizzazione della descrizione del vino
DelUserControlle.java	Controller che permette la gestione della cancellazione degli user
OrdineController.java	Controller che permette la gestione degli ordini effettuati dagli user

2.1.3 Package View



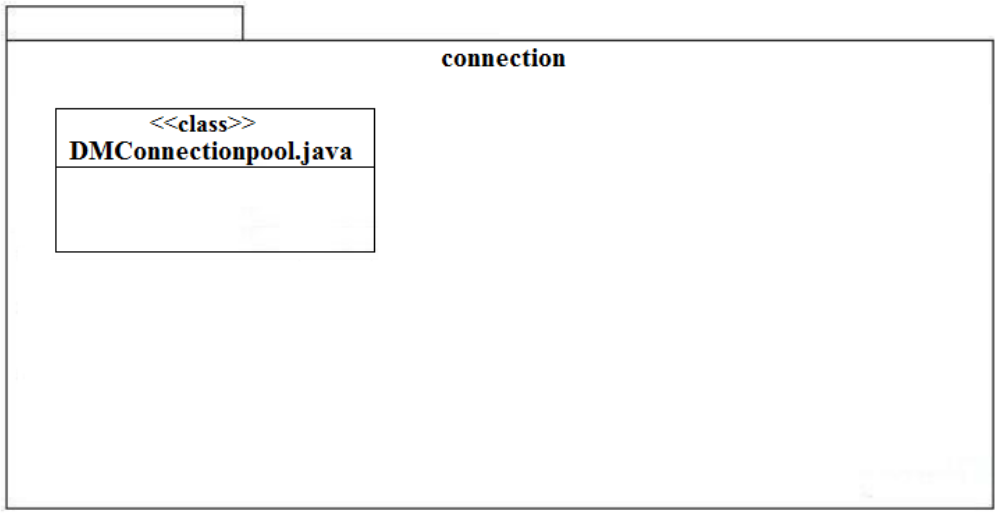
Classe	Descrizione
Cart.jsp	La view che consente all'user di visualizzare il carrello
Catalog.jsp	La view che consente all'user di visualizzare il catalogo
Contact.jsp	La view che consente all'user di visualizzare una form per l'invio di una e-mail
Footer.jsp	La view che consente all'user di visualizzare contatti utili del sito
Home.jsp	La view che consente all'user di visualizzare la pagina di benvenuto
Header.jsp	La view che consente all'user di visualizzare menu di navigazione
InfoDescrizione.jsp	La view che consente all'user di visualizzare la descrizione di un vino
Profile.jsp	La view che consente all'user di visualizzare il riepilogo dei propri dati
Utenti.jsp	La view che consente all'Admin di gestire gli utenti registrati al sito
Registrazione.jsp	La view che consente all'user di visualizzare la form per effettuare la registrazione al sito
Ordini.jsp	La view che consente all'admin di visualizzare la lista di ordini effettuati dagli utenti registrati

2.1.4 Package model dao



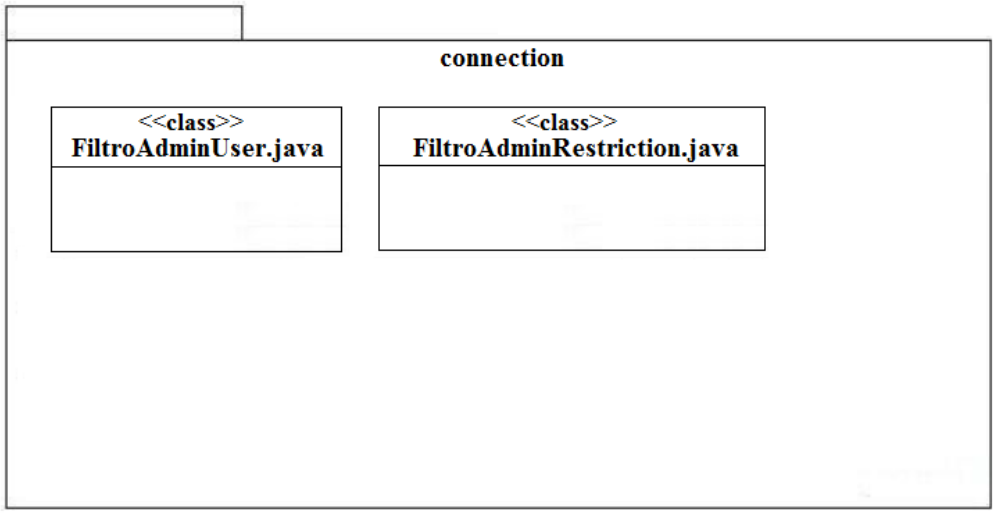
Classe	Descrizione
ArticoloDAO.java	Il model che effettua tutte le query riguardanti le funzionalità degli articoli in vendita interfacciandosi al db al quale e connesso
FeedbackDAO.java	Il model che effettua tutte le query riguardanti le funzionalità dei feedback interfacciandosi al db al quale e connesso
LoginDAO.java	Il model che effettua tutte le query riguardanti le funzionalità di login interfacciandosi al db al quale e connesso
ProfileDAO.java	Il model che effettua tutte le query riguardanti le funzionalità degli account, interfacciandosi al db al quale e connesso
RegistrationDAO.java	Il model che effettua tutte le query riguardanti le funzionalità della registrazione interfacciandosi al db al quale e connesso
DescriptionDAO.java	Il model che effettua le query riguardanti le informazioni legate alla descrizione del vino interfacciandosi al db al quale e connesso
OrdineDAO.java	Il model che effettua tutte le query riguardanti le funzionalità della registrazione interfacciandosi al db al quale e connesso
CancellUserDAO.java	Il model che effettua le query riguardanti le informazioni legate alla descrizione del vino interfacciandosi al db al quale e connesso

2.1.5 Package connection



Classe	Descrizione
DMConnectionpool.java	Il model che gestisce una pool di connessioni interfacciandosi al db al quale e connesso

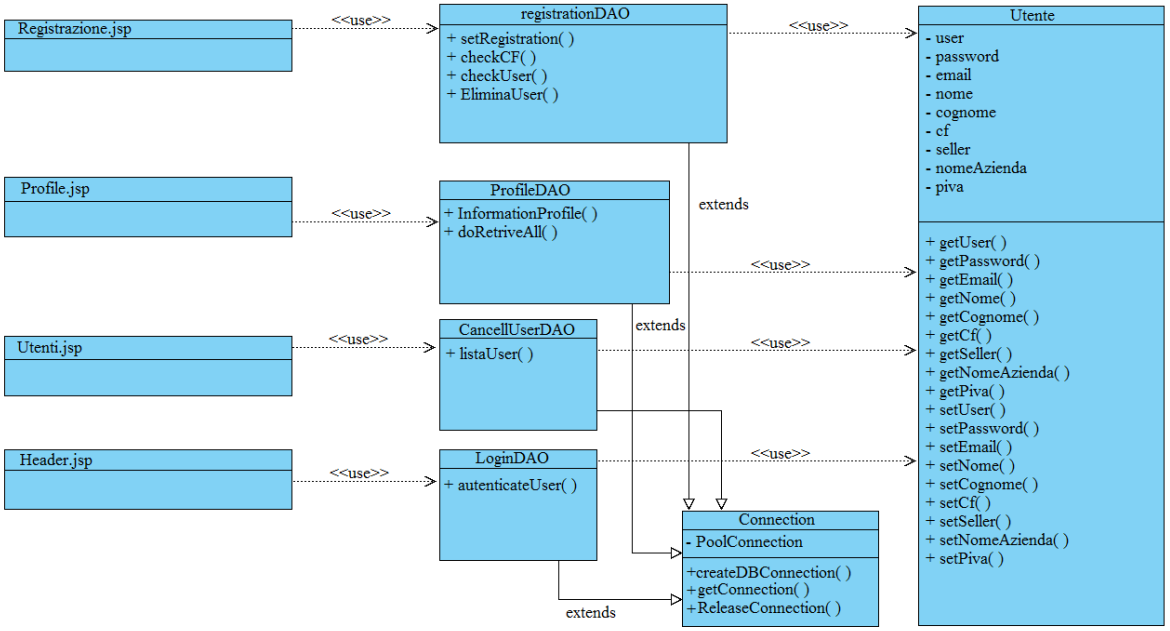
2.1.6 Package Filter



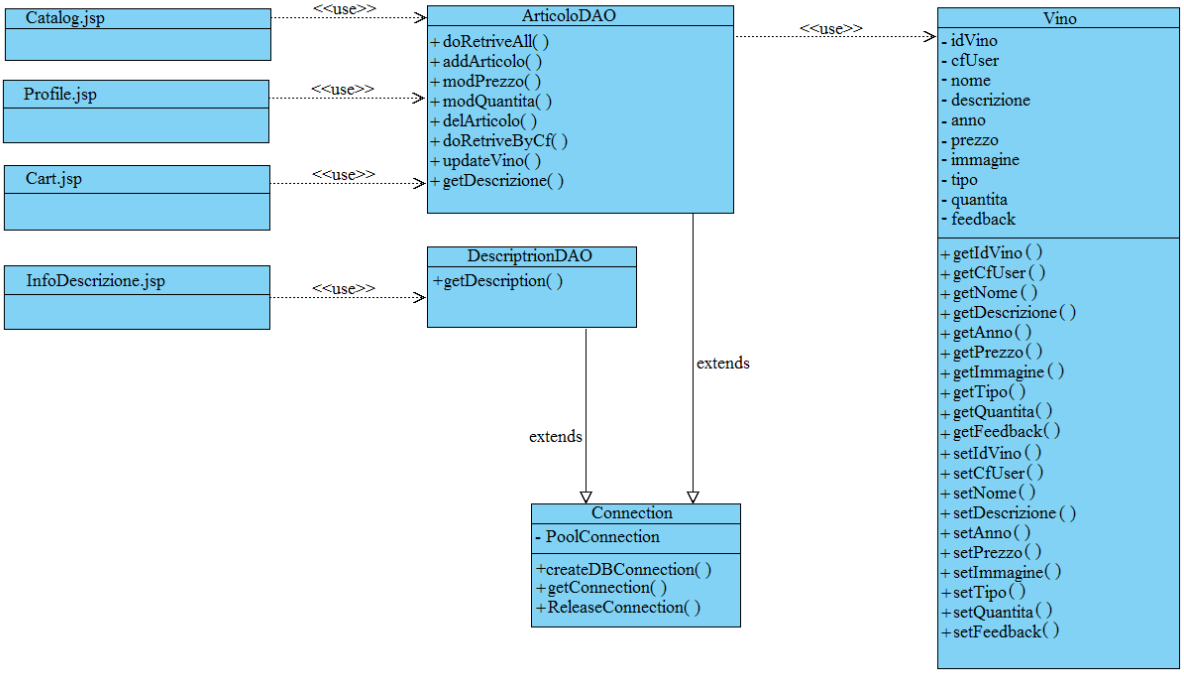
Classe	Descrizione
FiltroAdminUser.java	Filtro java che preclude l'accesso a tutti ad eccezione dell'Amministratore
FiltroAdminRestriction.java	Filtro java preclude l'accesso all'Amministratore del sito

3. Class Interfaces

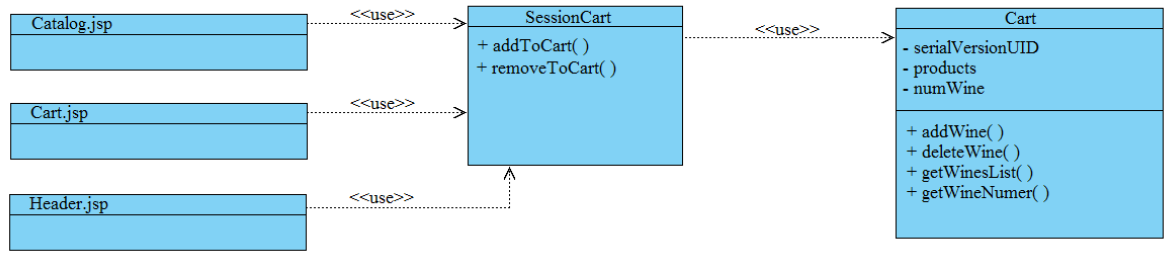
3.1 Sottosistema Gestione utenti



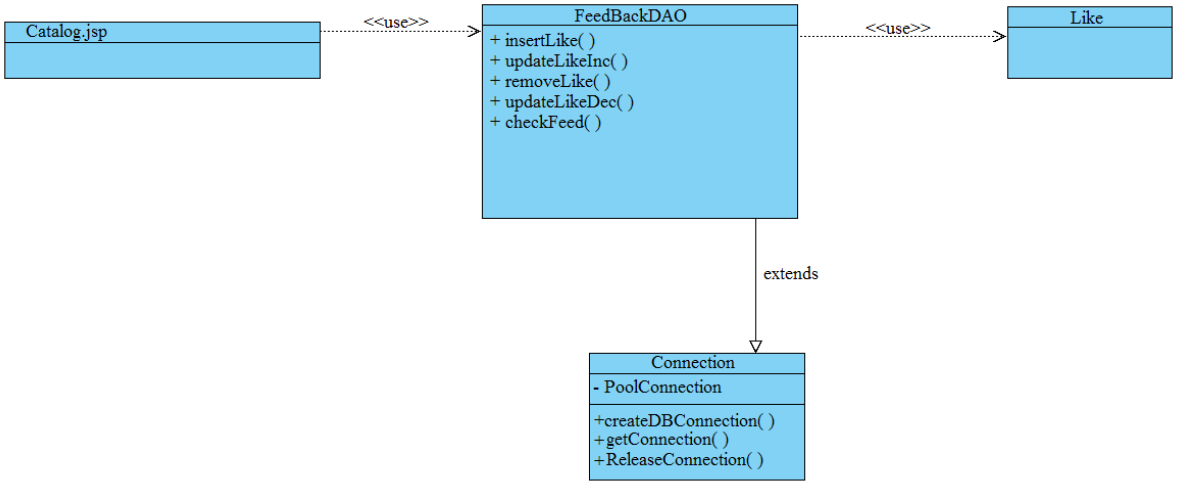
3.2 Sottosistema Gestione vini



3.3 Sottosistema Gestione Carrello



3.4 Sottosistema Gestione like



3.5 Sottosistema Gestione ordini