Università degli Studi di Salerno

Corso di Ingegneria del Software

GAP
Object Design Document

Data: 12/12/2021

Partecipanti:

Nome	Matricola
Giammarino Emanuele	0512108088
Adinolfi Giacinto	0512107764

Indice

1. Introduzione

L'Object Design Document (O.D.D.) ha come obiettivo quello di fornire una baseline per l'implementazione progettuale. Questo documento descrive i servizi forniti da ogni sottosistema, in termini di operazioni, i tipi, gli argomenti e le loro signatures. Inoltre, sono specificati i trade-off e le linee guida.

1.1 Object Design trade-offs

Quando si definiscono degli obiettivi, spesso solo un piccolo sottoinsieme di essi può essere tenuto in considerazione. Ad esempio, non è realistico sviluppare software che sia simultaneamente sicuro e costi poco. Per questo motivo, si è scelto di dare maggiore importanza ai seguenti punti.

Interfaccia vs Usabilità.

L'interfaccia è stata realizzata per risultare intuitiva e utilizzabile da un'ampia fascia di utenti, utilizzando descrizioni, immagini e pulsanti quanto più chiari e visibili possibile, sacrificando in parte l'estetica.

Comprensibilità vs Tempo

La stesura del codice risulterà essere quanto più dettagliata possibile a favore di una rapida comprensione per eventuali modifiche ed espansioni future. Così facendo, però, si incrementano leggermente i tempi di sviluppo, ma il lavoro finale avrà una qualità maggiore.

Response Time vs Hardware

Dal momento che il sistema dovrà garantire dei tempi di risposta rapidi nonostante un carico di utenti elevato, l'hardware del sistema dovrà essere adeguato e comprendere una fascia di componenti opportuni a tale scopo.

Sicurezza vs Efficienza

Dati i tempi di sviluppo contenuti, ci limiteremo ad implementare sistemi di sicurezza basati su username e password degli utenti e, in generale, di garantire un controllo sugli accessi basato su ruoli (attori) ben definiti.

Prestazioni vs Costi

Essendo il progetto esente da budget, il suo sviluppo dovrà essere basato su componenti open source e fare affidamento sulle prestazioni da esse offerti.

1.2. Interface documentation guidelines

Naming conventions

• È buona norma utilizzare nomi descrittivi, di uso comune di lunghezza medio-corta.

Variabili

• I nomi delle variabili devono cominciare con una lettera minuscola, se il nome della variabile è costituito da più parole, solo l'iniziale delle altre parole sarà maiuscola.

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 un'azione, seguito dal nome di un oggetto;

• I nomi dei metodi per l'accesso e la modifica delle variabili dovranno essere del tipo getNomeVariabile() e setNomeVariabile().

Classi

• I nomi delle classi devono essere singolari e iniziare con una lettera maiuscola, le parole contenute al suo interno devono cominciare con lettera maiuscola. Il nome deve fornire informazioni utili relative al loro scopo.

Packages

• I nomi dei packages devono iniziare con lettera minuscola e le parole successive con lettera maiuscola. I nomi, inoltre, saranno descrittivi delle funzionalità che comprenderanno le classi al suo interno. JavaServer Pages

• I nomi delle pagine .jsp devono essere in minuscolo e descrittivi dell'interfaccia che offrono all'utente.

1.3. Definitions, acronyms, and abbreviations

APACHE	Server Web gratuito sviluppato su piattaforma UNIX. Viene utilizzato, di solito, su macchine Unix o Linux (ma ne esiste anche una versione Windows)
BASI DATI	Genericamente è un insieme di dati, scritti in una forma specifica, che può essere elaborata direttamente da opportuni programmi
CLIENT	Software utilizzato dall'utente in grado di accedere, tramite rete, a un servizio offerto da un programma che gira su un altro computer (Server)
DBMS	Data Base Management System
GUI	Graphic User Interface
HTML	HyperText Markup Language
JDBC	Java DataBase Connectivity
JVM	Java Virtual Machine
PDF	Portable Document Format
URL	Uniform Resource Locator

1.4 Riferimenti

• Requirements Analysis Document (R.A.D.)

• System Design Document (S.D.D.)

2. Packages

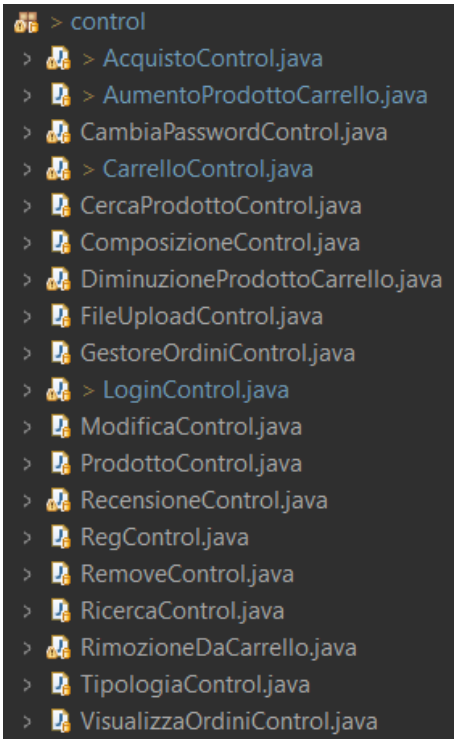
L'implementazione del back-end è situata (è organizzata) seguenti nei pacchetti:

- control
- model
- bean
- utility

L'implementazione del front-end è composta dai vari file .jsp e .html situati nella cartella WebContent, organizzati in questo modo:

- WebContent/admin
- WebContent/common
- WebContent

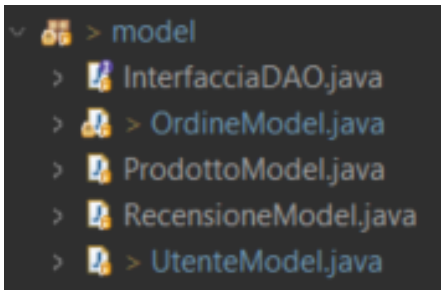
2.1 Package control



Control	
Classe	**Descrizione**
AcquistoControl	Servlet che si occupa di effettuare il salvataggio di un ordine.
AumentoProdottoCarrello	Servlet che si occupa dell'aumento della quantità di un prodotto all'interno del carrello.
DiminuzioneProdottoCarrello	Servlet che si occupa del decremento della quantità di un prodotto all'intero del carrello.
CarrelloControl	Servlet che si occupa dell'inserimento di un prodotto all'interno del carrello.
CercaProdottoControl	Servlet che si occupa della ricerca di un prodotto all'interno del database.
ComposizioneControl	Servlet che recupera i prodotti di un ordine.
CambiaPasswordControl	Servlet che si occupa di effettuare il cambio password ad un account.
FileUploadControl	Servlet predisposta al caricamento di una immagine, per l'inserimento di un prodotto nel database.
GestoreOrdiniControl	Servlet che si occupa del caricamento della pagina e dei dati nella pagina del gestore del catalogo
LoginControl	Servlet che si occupa del Login di un utente.

ModificaControl	Servlet che si occupa della modifica di un prodotto.
ProdottoControl	Servlet che si occupa del caricamento del catalogo.
RecensioneControl	Servlet che si occupa dell'inserimento di una recensione ad un prodotto.
RegControl	Servlet che si occupa della registrazione di un utente.
RemoveControl	Servlet che si occupa della rimozione di un prodotto.
RicercaControl	Servlet che si occupa della ricerca di un prodotto.
RimozioneDaCarrello	Servlet che si occupa della rimozione di un prodotto dal carrello.
TipologiaControl	Servlet per la restituzione di prodotti di una determinata tipologia
VisualizzaOrdiniControl	Servlet per la visualizzazione degli ordini effettuati

2.2 Package model

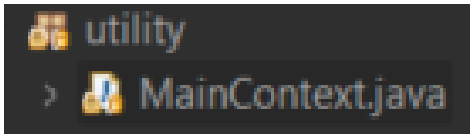


Model	
Classe	**Descrizione**
InterfacciaDAO	Interfaccia che implementa i metodi delle transazioni
OrdineModel	Comprende le funzionalità per le gestione degli ordini
ProdottoModel	Comprende le funzionalità per le gestione del prodotto
RecensioneModel	Comprende le funzionalità per le gestione delle recensioni utente
UtenteModel	Comprende le funzionalità per le gestione degli utenti

2.3 Package bean

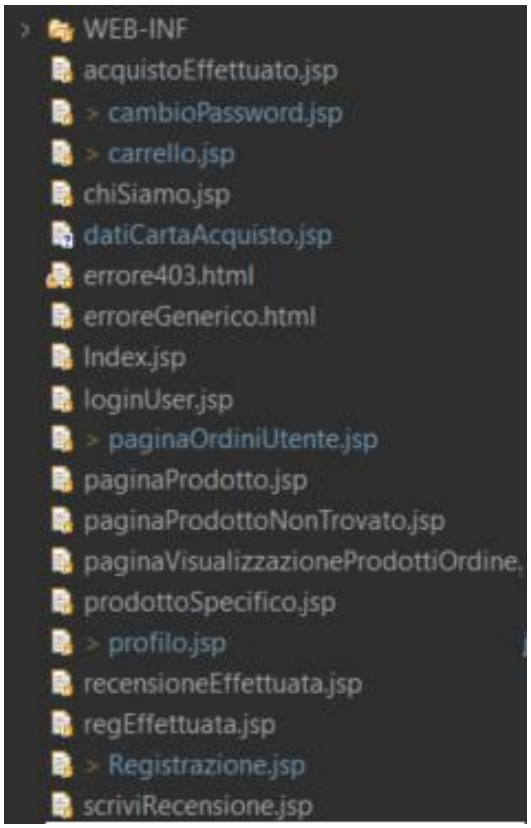
Bean	
Classe	**Descrizione**
CarrelloBean	Le istanze di questa classe rappresentano il carrello dell'utente che si è loggato al sito e lo sta visitando.
ComposizioneBean	Le istanze di questa classe rappresentano il singolo prodotto e materiale che fa parte di un ordine.
CompostoBean	Le istanze di questa classe rappresentano i materiali a cui un prodotto è associato.
MaterialeBean	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto materiale.
OrdineBean	Le istanze di questa classe rappresentano gli ordini effettuati dall'utente
ProdottoBean	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto prodotto.
RecensioneBean	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto recensione.
UtenteBean	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto utente.

2.4 Package utility



Utility	
Classe	**Descrizione**
MainContext	È una classe che viene lanciato appena viene aperta l'applicazione. Ha il compito di accedere al contesto della servlet, creare un DataSource, creare una connessione al database e rendere disponibile a tutte le servlet il DataSource.

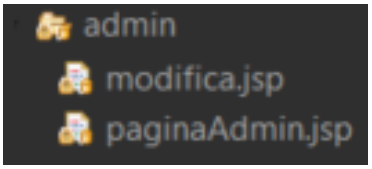
2.5 View WebContent



View	
Classe	**Descrizione**
acquistoEffettuato.jsp	View che mostra all'utente una pagina che indica il completamento dell'ordine.
cambioPassword.jsp	View che mostra all'utente un form per l'inserimento di una nuova password.
carrello.jsp	View che mostra all'utente il carrello.
chiSiamo.jsp	View che mostra all'utente la pagina di informazione del sito web.
datiCartaAcquisto.jsp	View che mostra all'utente un form per l'inserimento dei dati della carta di pagamento.
errore403.html	View che mostra all'utente, in caso di errore, una pagina di errore per l'errore 403.
erroreGenerico.jsp	View che mostra all'utente in caso di errore che non sia 403 una pagina di errore generico.
index.jsp	View che mostra all'utente il catalogo del sito web.
loginUser.jsp	View che mostra all'utente un form per l'inserimento dei dati di accesso.
paginaOrdiniUtente.jsp	View che mostra all'utente una pagina contenente gli ordini che ha effettuato.
paginaProdotto.jsp	View che mostra all'utente la pagina con i dettagli di un prodotto.
paginaProdottoNonTrovato.jsp	View che mostra all'utente una pagina che indica che il prodotto ricercato non è stato trovato.
paginaVisualizzazionePrdottoOrdine.jsp	View che mostra all'utente una pagina con i prodotti acquistati in un determinato ordine
prodottoSpecifico.jsp	View che mostra all'utente una pagina con i prodotti di una determinata tipologia.
profilo.jsp	View che mostra all'utente la pagina del suo profilo.
recensioneEffettuata.jsp	View che mostra all'utente il corretto inserimento della recensione.

regEffettuata.jsp	View che mostra all'utente che la registrazione effettuata è andata a buon fine.
registrazione.jsp	View che mostra all'utente un form di inserimento dei dati per la registrazione al sito.
scriviRecensione.jsp	View che mostra all'utente una pagina dove poter inserire la recensione di un prodotto.

2.6 View (WebContent/admin)



View	
Classe	**Descrizione**
modifica.jsp	View che mostra al gestore del catalogo i form per l'inserimento, cancellazione e modifica del prodotto.
paginaAdmin.jsp	View che mostra al gestore degli ordini tutti gli ordini effettuati sul sito web.

2.7 View (WebContent/common)

View	
Classe	**Descrizione**
Futher.jsp	View che viene importata in tutte le pagine .jsp e ne rappresenta il footer.
Header.jsp	View che viene importata in tutte le pagine .jsp e ne rappresenta il l'header.

3.Class Interface

3.1 Control

Nome Classe	AcquistoControl
Descrizione	Servlet che si occupa di effettuare il salvataggio di un ordine.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	AumentoProdottoCarrello
Descrizione	Servlet che si occupa dell'aumento della quantità di un prodotto all'interno del carrello.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	CarrelloControl
Descrizione	Servlet che si occupa dell'inserimento di un prodotto all'interno del carrello.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	CercaProdottoControl
Descrizione	Servlet che si occupa della ricerca di un prodotto all'interno del database.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	DiminuzioneProdottoCarrello
Descrizione	Servlet che si occupa del decremento della quantità di un prodotto all'intero del carrello.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	GestoreOrdiniCrontrol
Descrizione	Servlet che reindirizza il gestore degli ordini all'area a lui dedicata.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	ProdottoControl
Descrizione	Servlet che si occupa del caricamento del catalogo.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	RicercaControl
Descrizione	Servlet che si occupa della ricerca di un prodotto.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	RemoveControl
Descrizione	Servlet che si occupa della rimozione di un prodotto.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	ModificaControl
Descrizione	Servlet che si occupa della modifica di un prodotto.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	RecensioneControl
Descrizione	Servlet che si occupa dell'inserimento di una recensione ad un prodotto.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	fileUploadControl
Descrizione	Servlet predisposta al caricamento di una immagine, per l'inserimento di un prodotto nel database.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	ComposizioneControl
Descrizione	Servlet che recupera i prodotti di un ordine.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	RimozioneDaCarrello
Descrizione	Servlet che si occupa della rimozione di un prodotto dal carrello.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	LoginControl
Descrizione	Servlet che si occupa del Login di un utente.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	CambiaPasswordControl
Descrizione	Servlet che si occupa di effettuare il cambio password ad un account.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	RegControl
Descrizione	Servlet che si occupa della registrazione di un utente.
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	TipologiaControl
Descrizione	Servlet per la restituzione di prodotti di una determinata tipologia
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

Nome Classe	VisualizzaOrdiniControl
Descrizione	Servlet per la visualizzazione degli ordini effettuati
Metodi	+ doGet (HttpServlet request, HttpServletResponse response) + doPost (HttpServlet request, HttpServletResponse response)
Pre-condizione	
Post-condizione	

3.2 Bean

Nome Classe	ComposizioneBean
Descrizione	Le istanze di questa classe rappresentano il singolo prodotto e materiale che fa parte di un ordine.
Metodi	+ ComposizioneBean (): void + getIdMateriale (): int + setIdMateriale (int: idmateriale): void + getNumeroOrdine (): double + setNumeroOrdine (double: numeroOrdine): void + getCodiceProdotto (): int + setCodiceProdotto (int; codiceProdotto): void + getQuantità (): int + setQuantità (int: quantità): void
Pre-condizioni	ComposizioneBean (): pre: getIdMateriale () pre: int pre: composizioneBean!= null
Post-condizioni	

Nome Classe	MaterialeBean
Descrizione	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto materiale.
Metodi	+ MaterialeBean() + getId () : Int + setId (int : idmateriale) : void + getTipologiaMateriale () : String + setTipologiaMateriale (String : tipologiaMateriale) : void + getColore () : String + setColore () : void
Pre-condizioni	
Post-condizioni	

Nome Classe	CarrelloBean
Descrizione	Le istanze di questa classe rappresentano il carrello dell'utente che si è loggato al sito e lo sta visitando.
Metodi	+ CarrelloBean () + getPrezzoTotale () : float + setPrezzoTotale (float : prezzo) + setPrezzoTotaleRimozione (float : prezzo) + getProdotti () : ArrayList<ProdottoBean> + getMateriali () : ArrayList<MaterialeBean> + set Prodotti (ArrayList<ProdottoBean> : prodotti) + setMateriali (ArrayList<MaterialiBean> : materiali) + getQuantità () : int + addProdotto(ProdottoBean : prodotto, MaterialeBean : materiale)
Pre-condizioni	
Post-condizioni	

Nome Classe	CompostoBean
Descrizione	Le istanze di questa classe rappresentano i materiali a cui un prodotto è associato.
Metodi	+ CompostoBean () + getIdMateriale (): int + setIdMateriale (int: idMateriale): void + getCodiceProdotto (): int + setCodiceProdotto (int codiceProdotto): void
Pre-condizioni	
Post-condizioni	

Nome Classe	OrdineBean
Descrizione	Le istanze di questa classe rappresentano gli ordini effettuati dall'utente.
Metodi	+OrdineBean (), + getNumeroOrdine () : double + setNumeroOrdine (double : numeroOrdine) : void + getEmail (String : email) : String + setEmail (String : email) : void + getPrezzoTotale () : float + setPrezzoTotale (float : prezzoTotale) : void + getNumeroProdotti () : int + setNumeroProdotti (int : numeroProdotti) : void + getDataOrdine () : LocalDate + setDataOrdine (LocalDate : localDate) : void + getNumeroCarta () : String + setNumeroCarta (String : numeroCarta) : void + getMeseScadenzaCarta () : String + setMeseScadenzaCarta (String : meseScadenza) : void + getAnnoScadenzaCarta () : String + setAnnoScadenzaCarta (String : annoScadenza) : void + getCvvCarta () : String + setCvvCarta (String : cvv) : void
Pre-condizioni	
Post-condizioni	

Nome Classe	ProdottoBean
Descrizione	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto prodotto.
Metodi	+ProdottoBean (), + getCodice () : int + setCodice (int : codice) : void + getNome () : String + setNome (String : nome) : void + getAltezza () : int + setAltezza (int : altezza) : void + getProfondità () : int + setProfondità (int : profondità) : void + getLarghezza () : int + setLarghezza (int : larghezza) : void + getTipologia () : String + setTipologia (String : tipologia) : void + getQuantità () : int + setQuantità (int : quantità) : void + setPrezzo (float : prezzo) : void + getSconto () : int + setSconto (int : sconto) : void
Pre-condizioni	
Post-condizioni	

Nome Classe	RecensioneBean
Descrizione	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto recensione.
Metodi	+ RecensioneBean (), + getTesto () : String + setTesto (String : testo) : void + getData () : String + setData (String : data) : void + getCodice () : int + setCodice (int : codice) : void + getEmail () : String + setEmail (String : email) :_ void
Pre-condizioni	
Post-condizioni	

Nome Classe	UtenteBean
Descrizione	Le istanze di questa classe rappresentano le informazioni e metodi applicabili sull'oggetto utente.
Metodi	+ UtenteBean () + getCf () : String + setCf (String : cf) : void + getNome () : String + setNome (String : nome) : void + getCognome () : String + setCognome (String : cognome) : void + getEmail () : String + setEmail (String : email) : void + getPassword () : String + setPassword (String : password) : void + getIndirizzo () : String + setIndirizzo (String : indirizzo) : void + setIndirizzo (String : indirizzo_fatturazione) : void + getTelefono () : String + setTelefono (String : telefono) : void + getRuolo () : String + setRuolo (String : ruolo) : void
Pre-condizioni	
Post-condizioni	

3.3 Model

Nome Classe	**OrdineModel**
Descrizione	Comprende le funzionalità per le gestione degli ordini
Metodi	+ OrdineModel (DataSource : ds) + composizioneOrdine (string : numOrdine) : Collection<ComposizioneBean> + doRetriveAllPerUtente (UtenteBean : utente) : Collection<OrdineBean> + doRetriveAll (String : ordine) : Collection<OrdineBean> + doSave (OrdineBean : ordine) : void + doSaveComposizione (ComposizioneBean : item) : void
Pre-condizioni	-composizioneOrdine (string : numOrdine) → numOrdine != null. -doRetriveAllPerUtente (UtenteBean : utente) → utente != null. -doRetriveAll (String : ordine) → ordine != null. -doSave (OrdineBean : ordine) →ordine != null. -doSaveComposizione (ComposizioneBean : item) → item != null.
Post-condizioni	-composizioneOrdine (string : numOrdine) → Collection<ComposizioneBean> != null. -doRetriveAllPerUtente (UtenteBean : utente) → Collection<OrdineBean> != null. -doRetriveAll (String : ordine) → Collection<OrdineBean> != null.

Nome Classe	**ProdottoModel**
Descrizione	Comprende le funzionalità per le gestione del prodotto
Metodi	+ ProdottoModel (DataSource : ds) + doRetriveByCodice (int : codice) : ProdottoBean + doRetriveByKey (String : nomeProdotto) : ProdottoBean + doRetriveAll () : Collection<ProdottoBean> + doSave (ProdottoBean : prodotto) : void + doUpdate Quantita (ProdottoBean : prodotto) : void + doUpdateCatalogo (ProdottoBean : prodotto) : void + restituisciQuantita (ProdottoBean : prodotto) : int + doDelete (ProdottoBean : prodotto) : void + doSaveComposizioneProdotto (CompostoBean : item) : void + doRetriveAllMateriale (String : nomeProdotto) : Collection<MaterialeBean> + doRetriveByKeyMateriale (String : id) : MaterialeBean

	+ doRetriveAllTipologia (String : tipologia) : Collection<ProdottoBean>
Pre-condizioni	- doRetriveByCodice (int : codice) →codice != null - doRetriveByKey (String : nomeProdotto) → numeProdotto != null - doSave (ProdottoBean : prodotto) → prodotto != null - doUpdate Quantita (ProdottoBean : prodotto) → prodotto != null - doUpdateCatalogo (ProdottoBean : prodotto) → prodotto != null - restituisciQuantita (ProdottoBean : prodotto) → prodotto != null - doDelete (ProdottoBean : prodotto) → prodotto != null - doSaveComposizioneProdotto (CompostoBean : item) → item != null - doRetriveAllMateriale (String : nomeProdotto) → nomeProdotto != null - doRetriveByKeyMateriale (String : id) → id != null - doRetriveAllTipologia (String : tipologia) →tipologia != null
Post-condizioni	- doRetriveByCodice (int : codice) → ProdottoBean != null - doRetriveByKey (String : nomeProdotto) → ProdottoBean != null - doRetriveAll ()→ Collection<ProdottoBean> != null - restituisciQuantita (ProdottoBean : prodotto) → quantita != null - doRetriveAllMateriale (String : nomeProdotto) → Collection<MaterialeBean> != null - doRetriveByKeyMateriale (String : id) → MaterialeBean != null - doRetriveAllTipologia (String : tipologia) → Collection<ProdottoBean> != null

Nome Classe	**RecensioneModel**
Descrizione	Comprende le funzionalità per le gestione delle recensioni utente
Metodi	+ RecensioneModel (DataSource : ds) + doRetrieveByKey (String : code) : RecensioneBean + doRetrieveAll (String : code) : Collection<RecensioneBean>

	+ doSave (RecensioneBean : recensione) : void
Pre-condizioni	-doRetrieveByKey (String : code) → code =! null - doRetrieveAll (String : code) → code =! Null - doSave (RecensioneBean : recensione) → recensione != null
Post-condizioni	-doRetrieveByKey (String : code) → RecensioneBean != null - doRetrieveAll (String : code) → Collection<RecensioneBean> != null

Nome Classe	**UtenteModel**
Descrizione	Comprende le funzionalità per le gestione degli utenti
Metodi	+ UtenteModel (DataSource : ds) + doUpdate (UtenteBean: utente) : void + doRetriveByKey (String : codice) : UtenteBean + cercaUtente (String : email, String : password) : UtenteBean + cercaSimili (UtenteBean : utente) : boolean + doSave (UtenteBean : utente) : void
Pre-condizioni	-doUpdate(UtenteBean : utente) → UtenteBean != null e password != null. -doRetriveByKey (String : codice) → codice != null. -cercaUtente (String : email, String : password) → email != null e password != null. -cercaSimili (UtenteBean : utente) → utente != null. -doSave (UtenteBean : utente) → utente != null.
Post-condizioni	-doRetriveByKey (String : codice) → utente != null. -cercaUtente (String : email, String : password) →utente != null -cercaSimili (UtenteBean : utente) → boolean : true