UNIVERSITÀ DEGLI STUDI DI SALERNO

Ingegneria del Software

ANNO ACCADEMICO 2015/2016

LOGO

OBJECT DESIGN DOCUMENT

Versione 1.0

Top Manager:

Nome
Prof. De Lucia Andrea

Partecipanti:

Partecipante	Matricola
Carfora Andrea	0512102638
Grappone Renato	0512102536

Revision History:

Data	Versione	Descrizione	Autore
23/12/2015	1.0	Prima Stesura	Carfora Andrea Grappone Renato
07/01/16	1.1	Aggiunta di classi all'interno del package Exception	Carfora Andrea Grappone Renato
15/01/16	1.2	Aggiunta classi all'interno del package Gestione.Magazzino	Carfora Andrea Grappone Renato
20/01/16	1.3	Aggiunta classi all'interno del package Servlet.GestioneAccount	Carfora Andrea Grappone Renato

1. Introduzione

1.1. **Compromessi dell'object design**

Nella fase di Object design sorge la necessità di prendere delle decisioni. Alcuni dei trade-off relativi al progetto SI sono:

a) ***Comprensibilità vs Costi***: La comprensibilità del codice è un aspetto molto importante, soprattutto durante la fase di testing. Ogni classe e metodo deve essere facilmente interpretabile anche da chi non ha collaborato al progetto. Una appropriata documentazione del codice sorgente aumentala comprensibilità dello stesso. Ovviamente questa caratteristica comporterà un incremento del tempo necessario per lo sviluppo del progetto.

b) ***Buy vs Build***: Il compromesso adottato è quello espresso dall'affermazione "*If you can get 80% of the functionality you need from a packaged solution, then buy it. Otherwise, build it*". Naturalmente, il sistema SI può essere suddiviso in moduli. Se riusciamo a trovare qualche modulo che include la gran parte delle funzionalità di cui abbiamo bisogno, allora possiamo utilizzarlo, in modo tale da risparmiare tempo e risorse. Quando viene riutilizzato un modulo esistente, quindi, si ha il vantaggio di avere una soluzione pronta per l'uso. Tuttavia, se lo sviluppatore non sa come tale modulo funziona (perché il codice sorgente non è reso disponibile), integrarlo nel sistema potrebbe risultare molto difficile, richiedendo costi maggiori di quelli attesi.

c) ***Funzionalità vs Tempo di consegna***: Le funzionalità costituiscono un aspetto fondamentale del sistema. Tuttavia, la data di consegna è fissata e, pertanto sorge la necessità di trovare un buon bilanciamento. Ciò comporta che prima della consegna, occorre realizzare tutte le funzionalità di base del sistema.

1.2. **Linee guida per la documentazione delle interfacce**Al fine di rendere più uniforme la comunicazione tra i membri del team di sviluppo, e rendere i

programmi più comprensibili, adottiamo specifiche linee guida e convenzioni relative alla documentazione delle interfacce.

Per maggiori dettagli sugli standard di codifica, si rimanda al documento di specifica delle linee guida per il codice Java.

Denominazione:

Le convenzioni sono molto utili nella comprensione del codice sorgente in quanto possono fornire informazioni anche attraverso la semplice funzione di identificatore, ad esempio, se è una costante, un package, o una classe.

Questa sezione fornisce una panoramica sulle convenzione di denominazione. Esse sono descritte singolarmente nella tabella.

Identifier Type	Rules for Naming	Examples
Classi	I nomi delle classi dovrebbero essere sostantivi scritti secondo il sistema Camel. I nomi delle classi che modellano un eccezione devono terminare per " Exception". I nomi delle classi devono essere semplici e descrittivi e per quanto possibile evitare acronimi e abbreviazioni.	Class: AccountManager Exception: EmailException

Metodi	I metodi devono essere verbi, e devono seguire il sistema Camel, in cui la prima lettera di ogni parola è in maiuscolo ad eccezione della prima.	setNomeProdotto();
		
		setDescrizione();

Variabili	I nomi delle variabili devono essere brevi ma significative. Seguono il sistema camel in cui la prima lettera è minuscola e ,in caso di parole composte, la lettera che delinea l'inizio della	codice, nomeProdotto, descrizione

	successiva deve essere maiuscola.	
Costanti	I nomi delle costanti sono maiuscoli con le parole separate da underscore.	ACCOUNT_INSERT
Parametri	I nomi dei parametri devono seguire le convenzioni definite per le variabili	ricercatore;
Package	I nomi dei package seguono il sistema Camel	Exception Entità

1.3. **Definizioni, acronimi e abbreviazioni *Definizioni:***

Trade-off: Situazione in cui si pone una scelta di compromessa da due obiettivi ugualmente desiderabili, ma in conflitto tra di loro.

Package: Insieme di classi java.

JavaDoc: Strumento che estrae dai commenti di un programma una documentazione dettagliata del codice.

Acronimi:

RAD: Requirement Analysis Document.

SDD: System Design Document.

ODD: Object Design Document.

Abbreviazioni:

SI: Sportswear Identification.

1.4. **Referenze**

-RAD; -SDD;

- Linee guida codice Java.

2. PACKAGE

Per il raggruppamento delle classi del nostro progetto si è cercato di mantenere la struttura identificata all'interno del SDD, cioè i tre layer principali, con l'aggiunta di altri. I package sono i seguenti.

PACKAGE
Entità
Exceptions
Managers

Con l'aggiunta dei seguenti per le varie gestioni:

Servlet.GestioneBilancio
Servlet.GestioneCarrello
Servlet.GestioneClienti
Servlet.GestioneFaQ
Servlet.GestioneMagazzino
Servlet.GestioneOrdini
Servlet.GestioneAccount

2.1. **Descrizioni dei package**

2.1.1. SI Principali

Nome package	Descrizione
Entità	Package che contiene le entità chiavi del sistema
Exceptions	Package che contiene le eccezioni che si possono verificare all'interno del sistema
Managers	Package che contiene i gestori delle operazioni che possono essere effettuate su un oggetto contenuto in Entità.

2.1.2. SI Secondari

Nome package	Descrizione
Servlet.GestioneCarrello	Package che contiene la gestione del carrello sia da parte del cliente che del venditore
Servlet.GestioneFaQ	Package che contiene la gestione della visualizzazione delle informazioni dell'account.
Servlet.GestioneBilancio	Package che contiene le classi che implementano le azioni sulla gestione del bilancio dell'admin.
Servlet.GestioneClienti	Package che contiene le classi che implementano le azioni sulla gestione clienti dell'admin

Servlet.GestioneMagazzino	Package che contiene le classi che implementano le azioni sulla gestione del magazzino da parte dell'admin (Visualizza prodotti, categorie, etc)
Servlet.GestioneOrdini	Package che contiene le classi che implementano le azioni sulla gestione ordini sia da client che da admin
Servlet.GestioneAccount	Package che contiene la gestione dell'account (login , creazione,logout,cancellazione)

3. Interfaccia delle classi

3.1 SI Plan

Il sistema SportsWear Reseller non fa uso di interfacce poiché non necessarie.

3.1.1 Package Entità

Class Hierarchy

Account

Carrello

Prodotto

3.1.1.1 Class Account

Constructor Summary
Account(String email, String password, String nome, String cognome, String societa,String indirizzo, String partitaIva, String tipo) Costruttore usato per la istanziazione di un oggetto di tipo account. Oggetto che in seguito servirà per la creazione dell'account all'interno del DB eseguendo la query.

Method Summary	
String	getEmail() Metodo che consente la lettura della mail utente;
String	getPassword() Metodo che consente la lettura della password utente;
void	setPartitaIva() Metodo che consente di impostare un nuovo numero di partita iva;
String	getPartitaIva() Metodo che consente la lettura del numero di partita iva;
String	getTipo() Metodo che consente la lettura del tipo di account utente;
String	getSocietà() Metodo che consente la lettura del nome della società dell'utente registrato;
String	getNome() Metodo che consente la lettura del nome dell'utente;
String	getCognome() Metodo che consente la lettura del cognome dell'utente;
void	setEmail() Metodo che consente di impostare una nuova mail all'utente;
void	setPassword() Metodo che consente di impostare una nuova password all'utente;
void	setTipo() Metodo che consente di settare il tipo di account dell'utente;
void	setSocietà() Metodo che consente di settare il nome della società dell'utente;
void	setNome() Metodo che consente di settare un nuovo nome all'utente;
void	setCognome() Metodo che consente di settare un nuovo cognome all'utente;
Methods inherited from class java.lang.Object	

3.1.1.2 Class Prodotto

Constructor Summary	
Prodotto(int idProdotto, String nome, String descrizione, int numero_pezzi, float prezzo, String categoria)	
Method Summary	
int	getIdProdotto () Metodo che restituisce l'id del prodotto;
String	getNome() Metodo che restituisce il nome del prodotto;
String	getDescrizione() Metodo che restituisce la descrizione del prodotto;
String	getCategoria() Metodo che consente la lettura della categoria del prodotto;
int	getNumeroPezzi() Metodo che consente la lettura del numero pezzi in magazzino di un prodotto;
float	getPrezzo() Metodo che consente la lettura del prezzo di un prodotto;
void	setIdProdotto() Metodo che consente di impostare un nuovo id prodotto;
String	getCognome() Metodo che consente la lettura del cognome dell'utente;
void	setEmail() Metodo che consente di impostare una nuova mail all'utente;
void	setNome() Metodo che consente di impostare un nuovo nome al prodotto;
void	setDescrizione() Metodo che consente di settare una descrizione del prodotto;
void	setCategoria() Metodo che consente di settare la categoria del prodotto;
void	setNumeroPezzi() Metodo che consente di settare il numero pezzi di un prodotto;
void	setPrezzo() Metodo che consente di settare un nuovo prezzo al prodotto;

Methods inherited from class java.lang.Object

3.1.2 Package Exception

Class Hierarchy

EmailException extends java.lang.Exception

PasswordException extends java.lang.Exception

MagazzinoException extends java.lang.Exception

NameException extends java.lang.Exception

NullAccountException extends java.lang.Exception

PivaException extends java.lang.Exception

SocietaException extends java.lang.Exception

3.1.2.1 Class EmailException

Constructor Summary
EmailException(String message); **EmailException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore contenuto nella mail dell'utente;

Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Errore email")
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.2 Class PasswordException

Constructor Summary
PasswordException Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore contenuto nella password dell'utente;
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.3 Class MagazzinoException

Constructor Summary
MagazzinoException(String message); **MagazzinoException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore contenuto nel nome o cognome contenuto in gestione magazzino;

Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Errore nell'inserimento del nome e/o cognome")
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.4 Class NameException

Constructor Summary
NameException(String message); **NameException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore contenuto in un nome al momento dell'inserimento di un prodotto e registrazione di un account ; Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Errore nell'inserimento del nome e/o cognome")
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.5 Class NullAccountException

Constructor Summary
NullAccountException(String message); **NullAccountException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore riscontrato durante la ricerca di un Account all'interno del Database. Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Account non trovato")
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.6 Class pIvaException

Constructor Summary
pIvaException(String message); **pIvaException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore riscontrato all'interno del campo partita iva ; Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Errore Partita Iva")

Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.2.7 Class SocietaException

Constructor Summary
SocietaException(String message); **SocietaException();** Al costruttore viene passato il messaggio di errore opportuno per il tipo di errore contenuto nel nome societa durante la fase di registrazione. Al costruttore non viene passato nulla e il messaggio d'errore generato del sistema è definito in maniera generica ("Errore nome Società")
Method Summary
getMessage() Metodo che consente la lettura del messaggio passato all'eccezione;

3.1.3 Package Managers

Class Hierarchy

AccountManager

ClientiManager

MagazzinoManager

3.1.3.1 Class AccountManager

Constructor Summary	
Method Summary	
Account	logIn() Metodo che stabilisce la connessione al db ed esegue la query con email e password prese dal form HTML per l'esecuzione del logIn; Torna un oggetto di tipo Account;
int	creaAccount() Metodo che prende i parametri dal form HTML , crea una connessione al db e poi crea un nuovo account eseguendo la query; Torna l'identificativo dello stato dell'esecuzione della query per sapere se è andata a buon fine;
void	eliminaAccount() Metodo che consente l'eliminazione di un Account eseguendo la query al db;
Methods inherited from class java.lang.Object	

3.1.3.2 Class ClientiManager

Constructor Summary	
Method Summary	
ArrayList<Account>	listaClienti() Metodo che stabilisce la connessione al db ed esegue la query dall'account venditore per visualizzare la lista dei propri clienti; Torna il risultato in un ArrayList di Account;
ArrayList<Account>	ricercaAccount(String ric) Metodo a cui viene passato un nome e si occupa di eseguire la query per trovare tutti gli account dei clienti con quella parola nel nome all'interno del Db; Torna una lista con almeno un account se la ricerca ha prodotto dei risultati,altrimenti l'arrayList è vuoto;
Methods inherited from class java.lang.Object	

3.1.3.3 Class MagazzinoManager

Constructor Summary	
Method Summary	
int	inserisciProdotto() Metodo che stabilisce la connessione al db ed esegue la query per l'inserimento di un prodotto all'interno del magazzino; torna un intero come risultato che equivale all'esito della query;
ArrayList<Prodotto>	listaProdotti() Metodo che stabilisce una connessione al db ed esegue la query per visualizzare la lista dei prodotti di magazzino; Il risultato viene restituito in un ArrayList di prodotti;
ArrayList<Prodotto>	listaProdottiCategoria(String cat) Metodo a cui viene passata la stringa col nome della categoria di cui visualizzare la lista dei prodotti e che esegue la query tornando il risultato in un ArrayList di prodotti
ArrayList<Prodotto>	ricercaProdotto(String ricercatore) Metodo a cui viene passata una stringa che può essere il numero id del prodotto cercato oppure il nome del prodotto; Torna un ArrayList di prodotti attinenti alla ricerca;
int	verifica() Metodo che viene chiamato da ricercaProdotto e da ricercaProdottoCategoria e si occupa di verificare se chi cerca il prodotto ha digitato un numero o una stringa.
ArrayList<Prodotto>	ricercaProdottoCategoria(String ricercatore, String categoria) Metodo a cui viene passata una stringa che può essere il numero id del prodotto cercato oppure il nome del prodotto e una categoria in cui cercarlo; Torna un ArrayList di prodotti attinenti alla ricerca;

Methods inherited from class java.lang.Object

3.1.4 Package servlet.GestioneBilancio

Class Hierarchy

visualizzaBilancioServlet extends HttpServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account dalla sessione corrente per poi " chiamare " la jsp opportuna ;
Methods inherited from class java.lang.Object	

3.1.5 Package servlet.gestioneClientiServlet

Class Hierarchy

VisualizzaClientiServlet extends HttpServlet

RicercaClientiServlet extends HttpServlet

3.1.5.1 Class VisualizzaMagazzinoServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati del tipo di account dalla sessione in cui l'account è loggato, e in base a questo, chiama il manager che esegue la query per la visualizzazione della lista clienti salvata nel db e chiama la jsp opportuna.
Methods inherited from class java.lang.Object	

3.1.5.2 Class RicercaClientiServlet

Constructor Summary	
Method Summary	
void	doPost() Metodo che prende i dati del tipo di account dalla sessione in cui l'account è loggato, e in base a questo, chiama il manager che esegue la query per la ricerca di un cliente della lista clienti salvata nel db e chiama la jsp opportuna.
Methods inherited from class java.lang.Object	

3.1.6 Package servlet.gestioneMagazzino

Class Hierarchy

VisualizzaMagazzinoServlet extends HttpServlet

VisualizzaInserimento extends HttpServlet

VisualizzaCategoriaServlet extends HttpServlet

VisualizzaCategorieServletStampe extends HttpServlet

VisualizzaCategorieServletScarpette extends HttpServlet

VisualizzaCategoriaServletPalloni extends HttpServlet

VisualizzaCategoriaServletCompletini extends HttpServlet

InsertProdottoServlet extends HttpServlet

3.1.6.1 Class VisualizzaMagazzinoServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account tramite la sessione corrente , chiama il manager che esegue la query per visualizzare la lista dei prodotti e visualizza la jsp opportuna
Methods inherited from class java.lang.Object	

3.1.6.2 Class VisualizzaInserimento

Constructor Summary	
Method Summary	
void	doGet() Prende i dati dalla sessione corrente del tipo di account per poi chiamare la jsp per visualizzare l'inserimento del prodotto.
Methods inherited from class java.lang.Object	

3.1.6.3 Class VisualizzaCategoriaServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che riceve la richiesta http e la manda alla servlet;
Methods inherited from class java.lang.Object	

3.1.6.4 Class VisualizzaCategoriaServletStampe

Constructor Summary	
Method Summary	
void	doPost() Metodo che prende i dati dell'account dalla sessione corrente e tramite l'account chiama il manager che fa la query per visualizzare la lista dei prodotti della categoria .
Methods inherited from class java.lang.Object	

3.1.6.5 Class VisualizzaCategoriaServletScarpette

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account dalla sessione corrente e in base all'account chiama il manager che fa la query per visualizzare la lista dei prodotti della categoria e la jsp opportuna .
Methods inherited from class java.lang.Object	

3.1.6.6 Class VisualizzaCategoriaServletPalloni

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account dalla sessione corrente e tramite l'account chiama il manager che fa la query per visualizzare la lista dei prodotti della categoria palloni , e chiama la jsp opportuna.
Methods inherited from class java.lang.Object	

3.1.6.7 Class VisualizzaCategoriaServletCompletini

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account dalla sessione corrente e tramite l'account chiama il manager che fa la query per visualizzare la lista dei prodotti della categoria completini, e chiama la jsp opportuna.
Methods inherited from class java.lang.Object	

3.1.6.8 Class InsertProdottoServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che prende i dati dell'account dalla sessione , recupera i dati inseriri nella form e li passa al manager opportuno che effettua gli opportuni controlli e se tutto va a buon fine effettua l'inserimento aprendo la jsp opportuna altrimenti lancia un'eccezione
Methods inherited from class java.lang.Object	

3.1.7 Package servlet.visualizzaClientiServlet

Class Hierarchy

visualizzaClientiServlet extends HttpServlet

Constructor Summary	
Method Summary	
void	doGet() Metodo che riceve la richiesta http e la manda alla servlet;
Methods inherited from class java.lang.Object	

3.1.8 Package servlet.GestioneAccount

Class Hierarchy

CreateAccountServlet extends HttpServlet

LoginServlet extends HttpServlet

VisualizzaAccountServlet extends HttpServlet

VisualizzaHomeServlet extends HttpServlet

3.1.8.1 Class CreateAccountServlet

Constructor Summary	
Method Summary	
void	doPost() Questo metodo acquisisce dai campi di registrazione dell'utente i dati ed esegue la query al db per la creazione di un nuovo Account
Methods inherited from class java.lang.Object	

3.1.8.2 Class LoginServlet

Constructor Summary	
Method Summary	
void	doPost() Questo metodo acquisisce dai form "E-mail" e "Password", effettua la verifica e in caso va a buon fine l'utente effettua il login nel sistema. In caso la email o la password non sono corrette viene stampato un allert a video.
Methods inherited from class java.lang.Object	

3.1.8.3 Class VisualizzaAccountServlet

Constructor Summary	
Method Summary	
void	doGet() Prende i dati dalla sessione corrente per farli visualizzare a video dall'account loggato.
Methods inherited from class java.lang.Object	

3.1.8.4 Class VisualizzaHomeServlet

Constructor Summary	
Method Summary	
void	doGet() Tramite la sessione corrente , prende i dati dall'account loggato, ed esegue il controllo sul tipo di account, chiamando poi la home di quell'account.
Methods inherited from class java.lang.Object	

3.1.9 Package servlet.GestioneFaQ

VisualizzaVenditoreServlet extends HttpServlet

3.1.9.1 Class VisualizzaVenditoreServlet

Constructor Summary	
Method Summary	
void	doGet() Tramite la sessione corrente , prende i dati dall'account loggato, e chiama la jsp opportuna per la visualizzazione delle informazioni-Account.
Methods inherited from class java.lang.Object	