|  |
| --- |
| **Università degli Studi di Salerno Corso di Ingegneria del Software** |

**Buy To Eat  
Documento Object Design  
Versione 1.0**

****

Data: 19/12/2017

Indice

1. Introduzione ………………………………………………………………………..3

* 1. Object Design Trade-offs.………………………………………………..3
  2. Linee Guida per la documentazione delle interfacce..…………………...3

2. Definizione dei Packages…………………………………………………………...3

3. Interfaccia delle classi……………………………………………………………...7

1. **Introduzione**
   1. **Object Design Trade-offs**

In quest’ultima fase, cioè quella di Object-Design, l’obiettivo è cercare di ottimizzare il modo di implementazione del modello di analisi (SDD e RAD), minimizzando i tempi di esecuzione, i tempi di risposta, la memoria ed i costi, rispettando però le decisioni prese nelle fasi precedenti.

Ogni funzionalità è stata associata ad una classe Java, implementata tramite SDK, in modo da rendere il codice più leggibile e più flessibile, rendendo semplice modificare sezioni di codice in futuro.

Durante l’Object-Design, il compito principale è quello di specificare le interfacce in modo tale da non creare difficoltà d’ integrazione tra le implementazioni dei vari programmatori. A tal proposito sono stati definiti gli attributi, le signature, la visibilità, i contratti di ogni classe e metodo.

L’interfaccia pubblica ed i sottosistemi applicano il principio “Need to know” , ovvero l’informazione è resa possibile solo attraverso ben definiti canali di accesso.

* 1. **Linee Guida per la documentazione delle interfacce**

Le variabili sono scritte in minuscolo.

I metodi sono scritti partendo con il primo carattere in minuscolo ma qualora siano composti da più parole, il primo carattere costituente il nome di tali parole, dalla seconda in poi, è scritto in maiuscolo (doGet, doPost)

Gli argomenti dei metodi hanno nomi chiari, che specificano bene il valore da passare.

1. **Definizione dei Packages**

I package del sistema sono:

* **Log**: contiene la classe che effettua il login sia dell’utente che dell’azienda.

La classe contenuta è:

* Login.java.
* **Registrazione**: contiene le classi che effettuano la registrazione sia per l’utente, sia per l’azienda.

Le classi contenute sono:

* regut.java;
* regaz.java.

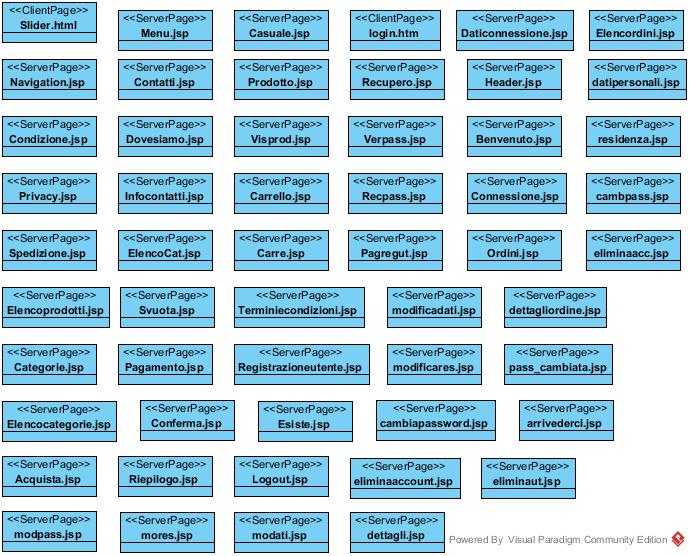
Oltre ai due package il sistema è costituito dalle seguenti servlet:

* Acquista.jsp
* AggiungiProdotto.jsp
* aggprod.jsp
* aggprodotto.jsp
* arrivederci.jsp
* azienda.jsp
* benvenuto.jsp
* cambiapassword.jsp
* Cambiapasswordaz.jsp
* cambpass.jsp
* cambpassaz.jsp
* Carre.jsp
* Carrello.jsp
* Casuale.jsp
* Categorie.jsp
* Condizione.jsp
* Conferma.jsp
* Connessione.jsp
* Contatti.jsp
* DatiConnessione.jsp
* datipersonali.jsp
* dettagli.jsp
* dettagliordine.jsp
* Dovesiamo.jsp
* ElencoCat.jsp
* Elencocategorie.jsp
* Elencoprodotti.jsp
* Elencordini.jsp
* elimaccaz.jsp
* eliminacc.jsp
* eliminaaccount.jsp
* Eliminaaccountaz.jsp
* eliminaaz.jsp
* EliminaProdotto.jsp
* eliminaut.jsp
* elimprodotto.jsp
* elprod.jsp
* Esiste.jsp
* Header.jsp
* Index.jsp
* Infocontatti.jsp
* login.html
* Logout.jsp
* Menu.jsp
* modati.jsp
* modificadati.jsp
* ModificaEff.jsp
* ModificaProdotto.jsp
* modificares.jsp
* modpass.jsp
* modpassaz.jsp
* modprod.jsp
* modProdotto.jsp
* mores.jsp
* mprod.jsp
* Navigation.jsp
* ordini.jsp
* Pagamento.jsp
* Pagregaz.jsp
* Pagregut.jsp
* pass\_cambiata.jsp
* Privacy.jsp
* Prodotto.jsp
* Recpass.jsp
* Recupero.jsp
* Registrazione.jsp
* Registrazioneazienda.jsp
* Registrazioneutente.jsp
* residenza.jsp
* Riepilogo.jsp
* Slider.html
* Spedizione.jsp
* Svuota.jsp
* Terminiecondizioni.jsp
* Verpass.jsp
* Visprod.jsp

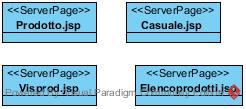
**Azienda**



**Utente**



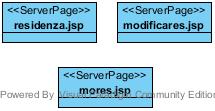
**Prodotto**

****

**Categoria**

****

**Dati Spedizione**

****

1. **Interfaccia delle classi**

In questa sezione vengono descritte le classi e le loro interfacce pubbliche, mettendo in rilievo le dipendenze con le altre classi e package, gli attributi, operazioni e casi eccezionali.

Nella specifica delle interfacce, verranno quindi specificate le attività, identificati gli attributi e le operazioni mancanti non descritte nei documenti precedenti.

Inoltre verranno specificate le segnature, la visibilità di ogni operazione, le precondizioni, post-condizioni e le invarianti.

**package Log:**

**Login.java**

**Descrizione Principale della Classe:**

Questa classe si occupa di effettuare il login alla piattaforma sia per l’utente che per l’azienda.

**Metodi della Classe:**

**public void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException,IOException**

Questo metodo permette di effettuare il login dell’utente o dell’azienda. Per farlo esegue in ordine questi passi e prende in input questi valori:

* String url: Stringa di connessione al database.
* Connection connect,connect1,connect2: prepara la connessione al database.
* String query,query1,query2: conterrà la query da sottoporre al database.
* PreparedStatement stmt,stmt1,stmt2: conterrà la query prima di sottoporla.
* String email e password: conterranno i valori passati dal form html al server.
* String emaild e passd, codf: conterranno i valori che riceveranno dal database.
* String nomeaz,idaz: conterranno i dati dell’azienda.
* Int flag: sarà usato per capire se il login è andato a buon fine per l’utente, per l’azienda o per nessuno dei due.

Precondizione: Bisogna controllare che la login e la password non corrispondano ad un utente già loggato.

**package registrazione:**

**regaz.java**

**Descrizione Principale della Classe:**

Questa classe si occupa di effettuare la registrazione dell’azienda alla piattaforma.

**Metodi della Classe:**

**public void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException,IOException**

Questo metodo permette di effettuare il login dell’utente o dell’azienda. Per farlo esegue in ordine questi passi e prende in input questi valori:

* String url: Stringa di connessione al database.
* Connection connect,connect1: prepara la connessione al database.
* String query,query1: conterrà la query da sottoporre al database.
* PreparedStatement stmt,stmt1: conterrà la query prima di sottoporla.
* String email, password, nomeazienda, stato, comune, provincia, indirizzo, cdf, piva, cap: conterranno i valori passati dal form html al server.
* String vere,veriva,vericod: conterranno i valori che riceveranno dal database.

Precondizione: Bisogna controllare che l’azienda non sia già registrata.

**regaz.java**

**Descrizione Principale della Classe:**

Questa classe si occupa di effettuare la registrazione dell’utente alla piattaforma.

**Metodi della Classe:**

**public void doPost(HttpServletRequest req, HttpServletResponse res) throws ServletException,IOException**

Questo metodo permette di effettuare il login dell’utente o dell’azienda. Per farlo esegue in ordine questi passi e prende in input questi valori:

* String url: Stringa di connessione al database.
* Connection connect,connect1: prepara la connessione al database.
* String query,query1: conterrà la query da sottoporre al database.
* PreparedStatement stmt,stmt1: conterrà la query prima di sottoporla.
* String email, password, cognome, nome, stato, indirizzo, comune, cdf, datanascita, provincia,cap: conterranno i valori passati dal form html al server.
* String vere,vericdf: conterranno i valori che riceveranno dal database.

Precondizione: Bisogna controllare che l’utente non sia già registrato.