****

**Object Design Document**

# **Partecipanti del progetto**

| **Nome** | **Matricola** | **E-mail** |
| --- | --- | --- |
| **Ermanno Allocco** | **0512109570** | [**e.allocco1@studenti.unisa.it**](mailto:e.allocco1@studenti.unisa.it) |
| **Melania Leonelli** | **0512110395** | [**m.leonelli3@studenti.unisa.it**](mailto:m.leonelli3@studenti.unisa.it) |
| **Salvatore Scafa** | **0512113101** | [**s.scafa1@studenti.unisa.it**](mailto:s.scafa1@studenti.unisa.it) |

# **1. Introduzione**

Dopo la realizzazione del Requirement Analysis Document e System Design Document, abbiamo descritto in linea di massima quello che sarà il nostro sistema e, quindi, i nostri obiettivi, tralasciando gli aspetti implementativi.  
Il presente documento ha lo scopo di produrre un modello capace di integrare in modo coerente e preciso tutte le funzionalità individuate nelle fasi precedenti.  
In particolare, definisce le interfacce delle classi, le operazioni, i tipi, gli argomenti e le signatures dei sottosistemi definiti nel System Design Document. Inoltre, sono specificati i trade-offs e le linee guida.

## **1.1 Object Design Trade-Off**

Durante la fase dell’Object Design, sono stati attentamente individuati ed analizzati i seguenti trade-off:

* **SICUREZZA vs COSTI SVILUPPO**  
  La sicurezza è uno degli aspetti chiave nello sviluppo della piattaforma, la protezione dei dati sensibili degli utenti al sito è una priorità assoluta. Tuttavia i costi per un sistema altamente sicuro sono troppo elevati,quindi, per limitare i costi senza intaccare la sicurezza ci limiteremo allo stretto necessario, implementando sistemi di sicurezza basati sulla crittografia dei dati sensibili.
* **DISPONIBILITA’ vs FAULT TOLLERANCE**  
  La disponibilità del sistema all’utente (anche in caso di errori di una funzionalità) è un trade-off molto importante e necessario.  
  A volte la Fault Tollerance non può essere evitata, al fine di salvaguardare un continuo funzionamento e una continua disponibilità del sistema
* **DESIGN vs USABILITA’**  
  L’interfaccia grafica è stata realizzata in modo da essere molto semplice, chiara e comprensibile. Fa uso di form e pulsanti disposti in maniera da rendere semplice l’utilizzo del sistema a quanti più utenti possibili.
* **RESPONSE TIME vs HARDWARE**  
  Il sistema garantisce una certa reattività alle richieste, e quindi è in grado di poter comunque offrire più servizi in contemporanea a più utenti.  
  Ovviamente questa caratteristica è limitata dalla potenza dell’Hardware del sistema.
* **MEMORIA vs ESTENSIBILITA’**  
  Il sistema deve permettere l’estensibilità a discapito della memoria utilizzata, in modo tale da permettere al cliente di richiedere agli sviluppatori nuove funzionalità.

## **1.2 Linee guida per la documentazione delle interfacce**

Gli sviluppatori dovranno seguire le seguenti convenzioni per la struttura del codice:

### **Naming convention**

È buona norma utilizzare nomenclature che rispettino le seguenti caratteristiche:

-descrittivi

-di uso comune

-lunghezza media

### **Variabili**

I nomi delle variabili (eccetto quelle del database) sono scritti in “lowerCamelCase”.

### **Classi e pagine**

I nomi delle classi e delle pagine devono iniziare con una lettera maiuscola. Il nome devono essere inerenti all’azione che svolgeranno.

Ogni file sorgente contiene una singola classe e deve essere strutturato in un determinato modo:

-l’istruzione package che permette di inserire la classe in un determinato package

-l’istruzione import che importa le librerie necessarie alla classe

-il codice del file sorgente

### **Metodi**

I nomi dei metodi devono cominciare con una lettera minuscola e seguono anch’essi la struttura "lowerCamelCase".

Il nome del metodo deve identificare l’azione che segue.

### **Pagine HTML**

Le pagine HTML, sia in forma dinamica che statica, devono essere conformi allo standard HTML.

### **Fogli di stile CSS**

I fogli di stile CSS devono seguire la seguente convenzione:

-tutti gli stili non in-line devono essere collocati in fogli di stile separati

## **1.3 Definizioni, acronimi e abbreviazioni**

### **Acronimi:**

-**RAD**: Requirement Analysis Document

-**SDD**: System Design Document

-**ODD**: Object Design Document

### **Definizioni:**

-**upperCamelCase**: è una tecnica di Naming delle variabili, consiste nell’iniziare la parola con una lettera maiuscola e le rimanenti lettere minuscole

-**lowerCamelCase**: è la pratica di scrivere parole composte o frasi unendo tutte le parole tra di loro, consiste nello scrivere più parole insieme delimitando la fine e l’inizio di una nuova parola con una lettera minuscola

-**HTML**: linguaggio di programmazione utilizzato per lo sviluppo di pagine Web

-**CSS**: acronimo di Cascading StyleSheets è un linguaggio usato per definire la formattazione delle pagine Web

-**JavaScript**: JavaScript è un linguaggio di scripting orientato agli oggetti e agli eventi, comunemente utilizzato nella programmazione Web lato client per la creazione di effetti dinamici interattivi

-**MVC**: acronimo di Model-View-Controller, è un pattern architetturale molto diffuso nello sviluppo di sistemi software

## **1.4 Riferimenti**

I Riferimenti alle Documentazioni Inerenti allo Sviluppo della Piattaforma sono elencati di seguito:

* RequirementAnalysisDocument.docx
* System Design Document.docx
* ProblemStatement.docx

# 

# **2.0 Packages**

L’implementazione del back-end si declina nei seguenti pacchetti:

* Model
* Controller

Per l’implementazione del front-end abbiamo i seguenti pacchetti:

* webapp
  + adminpage
  + userpage

**2.1 Back-end Packages**

| **Controller** | |
| --- | --- |
| **Classe** | **Descrizione** |
| *AddPreferitiServelet* | Servlet che gestisce l’aggiunta di un prodotto nei preferiti |
| *AddToCartServlet* | Servlet che gestisce l’aggiunta di un prodotto nel carrello |
| *AddToDBServlet* | Servlet che gestisce l’aggiunta di un prodotto nel database |
| *AllOrdersServlet* | Servlet che restituisce tutti gli ordini effettuati da un utente |
| *CartServlet* | Servlet che restituisce il carrello di un utente |
| *CatalogoServlet* | Servlet che gestisce il catalogo |
| *DetailsServlet* | Servlet che gestisce i dettagli di un acquisto |
| *DriverManagerConnectionPool* | Classe che permette l'interazione con il database |
| *EditProductServlet* | Servlet che gestisce la modifica di un prodotto |
| *ExecuteOrderServlet* | Servlet che gestisce l’esecuzione di un ordine |
| *MenuDGServlet* | Servlet che gestisce il menu del giorno |
| *OrderUtenteServlet* | Servlet che gestisce gli ordini di un utente |
| *PreferitiPageServlet* | Servlet che gestisce la pagina dei preferiti di un utente |
| *ProductPageServlet* | Servlet che gestisce la pagina dei prodotti |
| *ProductUpdateServlet* | Servlet che gestisce l’aggiornamento di un prodotto |
| *ProfilePageServlet* | Servlet che gestisce la pagina profilo di un utente |
| *RegisterServlet* | Servlet che gestisce la registrazione di un utente |
| *RemoveFromCartServlet* | Servlet che gestisce la rimozione di un prodotto dal carrello |
| *RemoveFromDBServlet* | Servlet che gestisce la rimozione di un prodotto dal database |
| *SearchAjaxServlet* | Servlet che gestisce la ricerca di un prodotto dal catalogo |
| *SearchbyCategoryServlet* | Servlet che gestisce la ricerca per categoria di un prodotto |
| *SearchbyPreferitiServlet* | Servlet che gestisce la ricerca di un prodotto tra i preferiti |
| *SvuotaCestinoServlet* | Servlet che gestisce la rimozione dei prodotti dal carrello |
| *UpdateBalanceServlet* | Servlet che gestisce l’aggiornamento del saldo di un utente |

| **Model** | |
| --- | --- |
| **Classe** | **Descrizione** |
| *Acquisto* | Modello che delinea le caratteristiche di un acquisto |
| *Carrello* | Modello che delinea le caratteristiche di un carrello |
| *Contiene* | Modello che delinea le caratteristiche dei prodotti che aggiungi al carrello |
| *Ordine* | Modello che delinea le caratteristiche di un ordine |
| *Preferisce* | Modello che delinea le caratteristiche delle preferenze |
| *Prodotto* | Modello che delinea le caratteristiche di un prodotto |
| *Tessera* | Modello che delinea le caratteristiche di una tessera |
| *Utente* | Modello che delinea le caratteristiche di un utente |

**2.2 Front-end Packages**

| **webapp** | |
| --- | --- |
| **JSP** | **Descrizione** |
| *Index* | View della landing page di LunchTime |

| **webapp/adminpage** | |
| --- | --- |
| **JSP** | **Descrizione** |
| *addProduct* | View dell'aggiunta di un prodotto nel catalogo |
| *addToDBResult* | View dell’aggiunta di un prodotto del database |
| *deleteProduct* | View per la rimozione di un prodotto dal catalogo |
| *editProducts* | View per la scelta di un prodotto da modificare |
| *headerAdminPage* | View delle operazioni possibili dell’admin |
| *removeFromDBResult* | View della rimozione di un prodotto dal database |
| *editPage* | View per modifica di un prodotto |
| *headerAdmin* | View della header dell’admin |
| *headerEditPage* | View della Header di modifica del prodotto |

| **webapp/userpage** | |
| --- | --- |
| **JSP** | **Descrizione** |
| *AddPreferitiResult* | View per l’aggiunta di un prodotto nei preferiti |
| *aggiornato* | View di un prodotto aggiornato |
| *allOrders* | View di tutti gli ordini effettuati |
| *BalanceResult* | View del saldo |
| *Carrello* | View del carrello |
| *Catalogo* | View del catalogo dei prodotti |
| *chiSiamo* | View delle informazioni riguardanti i membri di LunchTime |
| *fattura* | View della fattura di un ordine |
| *headerGuest* | View della Header dell’ospite |
| *headerLogin* | View della header per il Login |
| *headerRegEff* | View della header per la registrazione effettuata |
| *headerRegister* | View della header per la registrazione |
| *headerUtente* | View della header per la dell’utente |
| *login* | View del login |
| *loginResult* | View del risultato del login |
| *MenuDG* | View del menu del giorno |
| *metodiPagamento* | View dei metodi di pagamento |
| *ordineeffettuato* | View della conferma dell’ordine effettuata |
| *productPage* | View del prodotto |
| *ProfilePage* | View del profilo utente |
| *register* | View della registrazione |
| *registerResult* | View del risultato della registrazione |
| *result* | View del risultato della ricerca di un prodotto |
| *resultsPreferiti* | View dei prodotti preferiti di un utente |
| *RicaricaSaldo* | View per la ricarica del saldo |
| *saldoinsufficiente* | View del saldo insufficiente |
| *searchbar* | View della barra di ricerca |

# **3.0 Packages**

**3.1 Controller**

| **Nome Classe** | ***AddPreferitiServelet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’aggiunta di un prodotto nei preferiti |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * checkPreferito * addPreferito |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***AddToCartServelet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’aggiunta di un prodotto nel carrello |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getProduct * addToCart |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***AddToDBServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’aggiunta di un prodotto nel database |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * addProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***AllOrdersServlet*** |
| --- | --- |
| **Descrizione** | Servlet che restituisce tutti gli ordini effettuati da un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getAllOrders |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***CartServlet*** |
| --- | --- |
| **Descrizione** | Servlet che restituisce il carrello di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getCart |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***CatalogoServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce il catalogo |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getAllProducts |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***DetailsServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce i dettagli di un acquisto |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getAcquisti |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***DriverManagerConnectionPool*** |
| --- | --- |
| **Descrizione** | Classe che permette l'interazione con il database |
| **Signature dei metodi** | * createDBConnection() : Connection * getConnection() : Connection * releaseConnection(Collection) : void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***EditProductServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la modifica di un prodotto |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***ExecuteOrderServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’esecuzione di un ordine |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getCart * getTotal * getLatestOrder * verificaSaldo * effettuaPagamento * getProduct * getPrezzo * addAcquisto * emptyBasket * getOrderByCode |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***MenuDGServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce il menu del giorno |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getMenuDG |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***OrderUtenteServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce gli ordini di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getOrdersUtente |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***PreferitiPageServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la pagina dei preferiti di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***ProductPageServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la pagina dei prodotti |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***ProductUpdateServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’aggiornamento di un prodotto |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * updateProduct * getProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***ProfilePageServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la pagina profilo di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getUserDatabyEmail * getTesseraData |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***RegisterServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la registrazione di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * register * addTessera |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***RemoveFromCartServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la rimozione di un prodotto dal carrello |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getCart * removeFromCart |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***RemoveFromDBServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la rimozione di un prodotto dal database |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * deleteProduct |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***SearchAjaxServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la ricerca di un prodotto dal catalogo |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * doRetrieveByName |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***SearchbyCategoryServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la ricerca per categoria di un prodotto |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * doRetrieveByCategoria |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***SearchbyPreferitiServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la ricerca di un prodotto tra i preferiti |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * getPreferiti |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***SvuotaCestinoServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce la rimozione dei prodotti dal carrello |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * cleanCart |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***UpdateBalanceServlet*** |
| --- | --- |
| **Descrizione** | Servlet che gestisce l’aggiornamento del saldo di un utente |
| **Signature dei metodi** | # doGet(HttpServletRequest request, HttpServletResponse response) : void  action:   * ricaricaTessera |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

**3.2 Model**

| **Nome Classe** | ***Acquisto*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di un acquisto |
| **Signature dei metodi** | * Acquisto(int, String, float) * getCodordine(): int * setCodordine(int): void * getCodprodotto(): String * setCodprodotto(String): void * getCosto(): float * setCosto(float): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Carrello*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di un carrello |
| **Signature dei metodi** | * Carrello(String) * getSEmailutente(): String * setEmailutente(int): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Contiene*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche dei prodotti che aggiungi al carrello |
| **Signature dei metodi** | * getIdprodotto(): String * setIdprodotto(String): void * getEmailutente(): String * setEmailutente(String): void * getQuantita(): int * setQuantita(int): void * getimportoTotale(): double * setimportoTotale(double): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Ordine*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di un ordine |
| **Signature dei metodi** | * getCodordine(): int * setCodordine(int): void * getData(): String * setData(String): void * getImportototale(): double * setImportototale(double): void * getEmailutente(): String * setEmailutente(String): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Preferisce*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche delle preferenze |
| **Signature dei metodi** | * getIdprodotto(): String * setIdprodotto(String): void * getEmailutente(): String * setEmailutente(String): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Prodotto*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di un prodotto |
| **Signature dei metodi** | * getIdprodotto(): String * setIdprodotto(String): void * getNome(): String * setNome(String): void * getCategoria(): String * setCategoria(String): void * getPrezzo(): float * setPrezzo(float): void * getImmagine(): String * setImmagine(String): void * getDescrizione(): String * setDescrizione(String): void * getDisponibile(): int * setDisponibile(): int |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Tessera*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di una tessera |
| **Signature dei metodi** | * Tessera(): * getCodicetessera(): int * setCodicetessera(int): void * getCategoria(): int * setCategoria(int): void * getSaldo(): int * setSaldo(int): void * getEmailutente(): String * setEmailutente(String): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |

| **Nome Classe** | ***Utente*** |
| --- | --- |
| **Descrizione** | Modello che delinea le caratteristiche di un utente |
| **Signature dei metodi** | * Utente(String, String): * getEmailutente(): String * setEmailutente(int): void * getPassword(): String * setPassword(String): void * getNomeutente(): String * setNomeutente(String): void * getRuolo(): int * setRuolo(int): void |
| **Pre-condizioni** |  |
| **Post-condizioni** |  |
| **Invariante** |  |