**Warrior’s Adventure – Object Design**

**1. Introduzione**

**1.1 Object Design trade-off:**

* **[Prestazioni vs Costi]:** Visto le scarse risorse finanziarie, le prestazioni punteranno ad offrire un esperienza di gioco accettabile, accettando qualche compromesso lato automatismi aggiornamenti senza riavviare la pagine di gioco.
* **[Interfaccia vs Easy-use]:** l’interfaccia verrà progettata in modo da risultare il più semplice e leggibile possibile, i contrasti saranno elevati in modo da semplificarne la lettura e le pagine saranno in numero ridotto in modo da non creare un senso di disorientamento nel giocatore. Per quanto riguarda la responsività dell’interfaccia si è scelto di adattarla unicamente a sistemi desktop per non andare contro ai requisiti precedentemente espressi.
* **[Interfaccia vs Tempo di risposta]:** L’interfaccia manterrà tempi di risposta più bassi possibili, in modo da permettere ai giocatori un esperienza d’uso ottimale compatibilmente alle risorse hardware della macchina che fungerà da server.
* **[Costi vs Mantenimento]:** Grazie all’utilizzo di Java EE il progetto sarà diviso in una componente client e una componente server questo permetterà un ottima scalabilità permettendo al progetto di girare su un'unica macchina server. Per quanto riguarda modifiche e aggiornamenti al codice, insieme al documento verrà rilasciato anche un java doc.

**1.2 Linee guida per l’interfaccia:**

Il Sistema è multi-utente (possono accedervi sia utenti comuni che amministratori, moderatori, gestori di gilda o gestori di torneo), per tale motivo il sistema nasconderà o mostrare funzionalità in base all’utente attualmente connesso.

**1.3 Componenti off-the-shell**

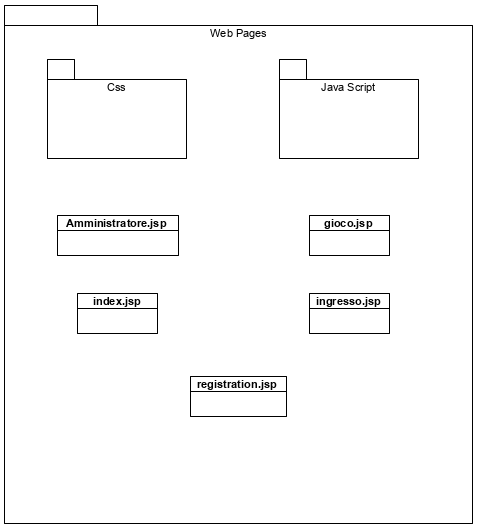
Componenti che andremmo a utilizzare sono:

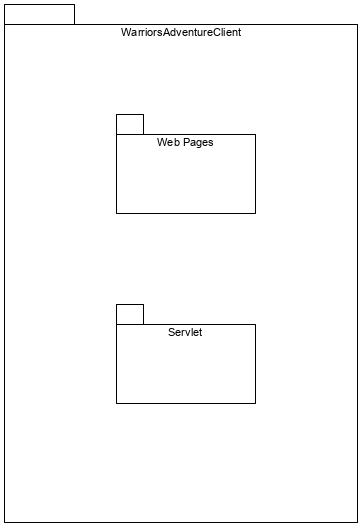
* Libreria open-source GSON distribuita da Google per linguaggio Java.
* Libreria jQuery per linguaggio Java Script.
* Libreria Open- Source AJAX per garantire aggiornamenti asincroni tra client e server.

**1.4 Definizioni, acronimi e abbreviazioni**

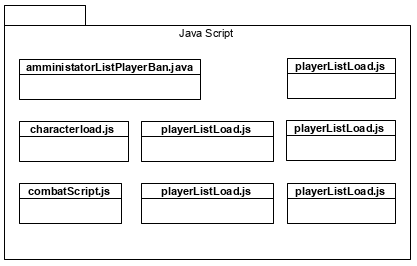
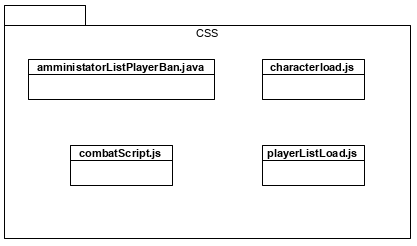
* DB = Database;
* DERBY = Apache Derby;
* RAD = Requirement Analysis Document;
* SDD = System Design Document;
* ODD = Object Design Document;
* PvP = Player vs Player;
* PvE = Player vs Environment;
* Player/s = Giocatore/I;
* JDBC = Java Data Base Connectivity;
* HTML = Linguaggio di Mark-Up per pagine web;
* CSS = Cascate Style Sheets;
* Amministratore: utente che si occupa di bannare e rimuovere e promuovere i giocatori all’interno del sito;

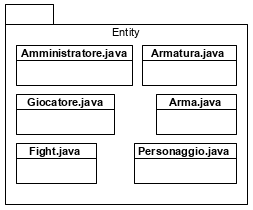
**2. Package**

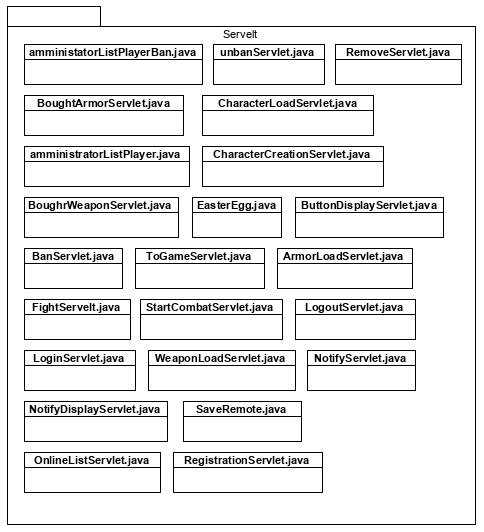
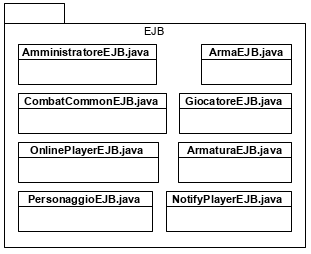


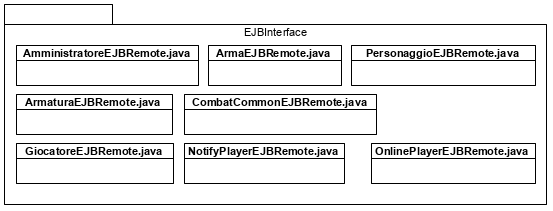
**2.1 Package Generale:**

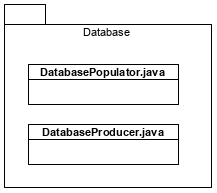
**2.2** **Package CSS e Java Script:**



**2.3 Package Servlet: 2.5 Package EJB:**

****

**2.5 Package Entity, Database e EJBInterface:**

****

**3. Interfaccia delle classi**

**3.3 Servlet:**

Le servlet implementano tutte il medesimo metodo con la seguente firma:

***protected void processRequest(HttpServletRequest request, HttpServletResponse response);***

In modo da garantire, il passaggio dei dati in modo sicuro e non visibile dalla barra degli indirizzi.

|  |  |
| --- | --- |
| **Nome Servlet:** | **AmministratorListPlayerServlet** |
| **Descrizione:** | Restituisce in formato JSON la lista dei giocatori iscritti al sito in modo che possa essere visualizzata all’interno dell’interfaccia dell’amministratore. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **AmministratorListPlayerServletBan** |
| **Descrizione:** | Restituisce in formato JSON la lista dei giocatori bannati dall’amministratore in modo che possa essere visualizzata all’interno dell’interfaccia dell’amministratore. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **BanServlet** |
| **Descrizione:** | Preleva dalla form contente il bottone di ban, il nome del giocatore del giocatore che l’amministratore desidera bannare e lo banna. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **BoughtArmorServelt** |
| **Descrizione:** | Preleva dalla form contente il bottone di acquisto, l’ID dell’item che il giocatore desidera acquistare e lo acquista. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **BoughtWeaponServelt** |
| **Descrizione:** | Preleva dalla form contente il bottone di acquisto, l’ID dell’item che il giocatore desidera acquistare e lo acquista. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **ButtonDisplayServlet** |
| **Descrizione:** | Attiva l’interfaccia di combattimento ad entrambi gli sfidanti. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **CharacterCreationServlet** |
| **Descrizione:** | Preleva dalla form contenente il nome e la classe del personaggio che il giocatore desidera creare e lo crea. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **CharachterLoadServlet** |
| **Descrizione:** | Restituisce in formato JSON la lista di tutti i personaggi associati a un giocatore in modo che possano essere visualizzati da quest’ultimo all’interno della pagina di ingresso. |

|  |  |
| --- | --- |
| **Nome Servlet:** | **FightServlet** |
| **Descrizione:** | Gestisce il combattimento tra due personaggi, gestendo il tipo di bottone che viene cliccato e inviando sotto forma JSON la stringa risultante dall’interazione per essere visualizzata. |