Object Design Document

GottaBattleEmAll

[1.Introduzione 2](#_Toc56307982)

[1.1. Trade off 2](#_Toc1964345059)

[1.2. Linee Guida 2](#_Toc1864184444)

[1.3.Definizione,Acronimi,abbrevazioni 2](#_Toc268170316)

[1.4 Reference 2](#_Toc2028565037)

[1.5 Convenzioni di tecnologia( nome temporaneo) 2](#_Toc2124801262)

[2. Package 3](#_Toc1692695861)

[2.1 Package gottaBattleEmAll 3](#_Toc1631263723)

[2.1.1 Package config 3](#_Toc1218245035)

[2.1.2 Package controller 3](#_Toc253271512)

[2.1.3 Package entity 3](#_Toc480320579)

[2.1.4 Package repository 3](#_Toc1236876332)

[2.1.5 Package service 3](#_Toc1083654391)

[2.2 Package resources 3](#_Toc1805938631)

[2.2.1 Package fragment 3](#_Toc594643488)

[3.Interfacce delle classi 3](#_Toc776318748)

# 1.Introduzione

## 1.1. Trade off

Performance vs Usabilità

Il sistema è progettato per privilegiare le performance, adottando il rendering lato server tramite pagine Thymeleaf. Questo approccio mira a migliorare tempi di risposta e carico del server, ottimizzando l'esecuzione delle operazioni sul lato server. Tuttavia, è importante bilanciare questa priorità con l'usabilità dell'interfaccia utente.

PAG 378 LIBRO

## 1.2. Linee Guida

Per lo sviluppo saranno utilizzate le seguenti convenzioni:

* Gli oggetti e i metodo saranno nominati a seconda della convenzione CamelCase
* Gli oggetti entita saranno chiamati semplicemente con il nome dell'entita
* Gli oggetti saranno nominati a seconda dell' entita di cui si occupano e del loro ruolo nell' applicazione per esempio TorneoService, UtenteController ecc...
* le pagine html in thydeconposizionemeleaf saranno segnate completamente in minuscolo
* i nomi dei package sono in minuscolo

## 1.3.Definizione,Acronimi,abbrevazioni

Acronimi:

* + RAD: Requirements Analysis Document
  + SDD: System Design Document
  + ODD: Object Design Document

## 1.4 Reference

- R.A.D.

- S.D.D.

## 1.5 Convenzioni di tecnologia( nome temporaneo)

La tecnologia che stiamo utilizzando porta alcune convenzioni dovute alla logica enterprise.

Design Pattern

I Design Pattern che abbiamo utilizzato sono:

Inversion of control

L'inversion of control è un pattern per cui un compon/ente di livello applicativo riceve il controllo da un componente appartenente a una libreria riusabile. Questo lo fa tramite un altro pattern cioè dependency Injection che tramite utilizzo di metadati istruisce i containter.

MVC

MVC è un pattern che divide il codice in tre parti specifiche cioè il Model View e Controller . Il model gestisce i dati, la view si occupa della visualizzazione dei dati e i controller riceve i comandi dell'utente attraverso le view e li attua modificando lo stato del model e delle view

# 2. Package

La decomposizione in package è la seguente:

<eventuale disegno>

il package gottaBattleEmAll , è un package in cui sono contenuti altri package che riguardano le classi Service, Repository, Controller, Configuration(config), Entity e si trova anche un file di startup per lo start del sistema.

il package configuration (per convenzione si chiamerà config) contiene le configurazioni di sistema per operazioni che vengono fatte durante lo startup o la creazione di determinati Bean.

il package controller contiene tutti gli oggetti controller che si occupano delle richieste Http

da parte dell'utente.

il package entity conterrà le entità del sistema con eventuale mapping in sql tramite le annotazioni di Spring.

il package repository contiene tutte le interfacce per la creazione delle repository per le operazioni di CRUD per ogni singola entità.

Il package service contiene l’interfaccia e l’implementazione dei servizi.

il package resources è composto dalle pagine html e dai package dei fragment e dal package static, rappresentano le view del sistema/\*da sistemare\*/

il package fragment contiene le sottopagine html che possibile ripetere su piu pagine il package static contiene le risorse statiche per le pagine html

## 2.1 Package gottaBattleEmAll

### 2.1.1 Package config

### 2.1.2 Package controller

### 2.1.3 Package entity

### 2.1.4 Package repository

### 2.1.5 Package service

## 2.2 Package resources

### 2.2.1 Package fragment

2.2.2 Package static

# 3.Interfacce delle classi

|  |  |
| --- | --- |
| GuestService | Questa classe raggruppa i servizi dedicati all'utente guest |
| Pre-Condizione | Post-Condizione |
| **context**  GuestService::registrazioneGiocatore(Giocatore giocatore)  **pre** : giocatore.username!=null and  UtenteService.findByUsername(giocatore.username,ruolo.GIOCATORE)==null  **pre :** giocatore.nome!=null  **pre:** giocatore.cognome!=null  **pre:** giocatore[.](http://giocatore.email/)email !=null and giocatore.email.re('^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$')  **pre:** giocatore.password!=null and giocatore.passwordmatch==giocatore.password | **post:** UtenteService.findByUsername(@pre.giocatore.username,ruolo.GIOCATORE)!=null  **post:** UtenteService.findByUsername(@pre.organizzatore.username,ruolo.GIOCATORE).stato==Stato.ATTIVO  **post:** result=true |
| **context** GuestService::registrazioneOrganizzatore(Organizzatore organizzatore)  **pre** : organizzatore.username!=null and  UtenteService.findByUsername(organizzatore.username,ruolo.ORGANIZZATORE)==null  **pre :** organizzatore.nome!=null  **pre:** organizzatore.cognome!=null  **pre:** organizzatore.email !=null and organizzatore,email.re('^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$')  **pre:** organizzatore.password!=null and organizzatore.passwordmatch==organizzatore.password | **post:** UtenteService.findByUsername(@pre.organizzatore.username,ruolo.ORGANIZZATORE)!=null  **post:** UtenteService.findByUsername(@pre.organizzatore.username,ruolo.ORGANIZZATORE).stato==Stato.INVERIFICA  //TODO gestore delle richieste  **post:** result=true |

|  |  |
| --- | --- |
| UtenteService | questo servizio fornisce i servizi che accumanano i servizi dei giocatori e organizzatori |
| Pre-Condizione | Post-Condizione |
| **context**  UtenteService::login(Utente utente,Ruolo ruolo)  **pre:** utente.username !=null  **pre:** utente.password!=null  **pre:** utente.ruolo!=null  **pre:** UtenteService.findByUsername(utente.username,ruolo)!=null  **pre:** utente.password ==  UtenteService.findByUsername(utente.username,ruolo).password  **pre:** not UtenteService.findByUsername(utente.username,ruolo).stato!=Stato.Attivo//OPTIMIZE bisognerebbe distinguere i vari casi(ban e in verifica) | **post:** sessions.get(utente.username)!=null (vabbe e presente in session) //FIXME la sintassi non e perfettamente corretta  **post:** result=true (se valgono le pre condizioni)  **post:** result = 'utente bannato' if UtenteService.findByUsername(utente.username,ruolo).stato== 'bannato'  **post:** result = 'utente in verifica' if UtenteService.findByUsername(utente.username,ruolo).stato== 'in verifica'  **post:** result = 'credenziali sbagliate' se non valgono le condizioni sulle variabili  //HACK non so se va bene scritto cosi |
| **context** UtenteService::modificaProfilo(Utente utente , ruolo)  **pre:** utente.username !=null  **pre:** utente.password!=null  **pre:** utente.email !=null and utente.email.re('^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$')  **pre:** utente.ruolo!=null  **pre :** organizzatore.nome!=null  **pre:** organizzatore.cognome!=null  **pre:** UtenteService.findByUsername(utente.username,ruolo)!=null  **pre:** utente.password == utente.repassword | **post:** @pre.utente == UtenteService.findByUsername(utente.username,@pre.ruolo)  **post**: result=true |
| **context** UtenteService::logout(Utente utente)  **pre:** utente.username!=null  **pre:** sessions.get(utente.username)!=null | **post :** sessions= @pre.sessions - sessions.get(utente.username)  //HACK una possibile soluzione |
| **context** UtenteService::findByUsername(String username,Ruolo ruolo)  **pre:** username!=null  **pre:** ruolo!=null | **post:** result= Utente.get(username) //FIXME da sistemare |

|  |  |
| --- | --- |
| TorneoService | questo servizio raggruppa tutti i servizi che riguardano la gestione dei tornei |
| Pre-Condizione | Post-Condizione |
| **context** TorneoService::findByName(String nome)  //NOTE questa funzione deve essere usata internamente e solo per la pagina torneo lato utente , lato organizzatore ci sara un'altra con il controllo dell'organizzatore  **pre:** nome!=null | **post:** result= Torneo.get(nome)//FIXME da sistemare |
| **context** TorneoService::cercareTorneo(String name) | **post:** result= Torneo.select(u | u.substring(0,len(name))==name) |
| **context** Torneo::iscrizioneTorneo(Giocatore giocatore,Torneo torneo)  **pre:** giocatore!=null  **pre:** torneo.stato==AttesaIscrizioni  **pre:** torneo !=null  **pre:** notgiocatore.tornei.include(torneo.nome)  **pre:** torneo.giocatori.len < torneo.capienza | **post:** torneo.giocatori=@pre.torneo.giocatori + giocatore  **post:** giocatore.tornei=@pre.giocatore.tornei + giocatore  **post :** result= true (se le precondizioni vanno bene)  //FIXME scritto poco formale  **post:** result= torneo full con il suo caso  **post:** result = sei stato rifiutato per la precondizione 4 |
| **context** TorneoService::seguireOrganizzatore(Giocatore giocatore, Organizzatore organizzatore)  **pre:** giocatore!=null  **pre:** organizzatore!=null | **post:** giocatore.follow=@pre.giocatore.follow + organizzatore  **post:** result =true |
| **context** TorneoService::getTorneoIscritto(Giocatore giocatore)  **pre:** giocatore!=null | **post :** result=giocatore.getTornei() |
| **context** TorneoService::creaTorneo(Torneo torneo,Organizzatore organizzatore)  **pre:** torneo.nomeTorneo!=null  **pre:** TorneoService.findByName(torneo.username)==null  **pre:** capienza !=null and capienza <=16 and capienza %2==0  **pre:** organizzatore !=null | **post:** TorneoService.findByName(torneo.username)!=null  **post:** TorneoService.findByName(torneo.username).stato==AttesaIscrizioni  **post:** result =true |
| **context** TorneoService::iniziareTorneo(Torneo torneo,Organizzatore organizzatore)  **pre:** torneo!= null  **pre:** torneo.stato==AttesaIscrizioni  **pre:** torneo.organizzatore== organizzatore  **pre:** torneo.giocatori.length== torneo.capienza | **post:** torneo.stato== Incorso  //TODO qua bisognerebbe mettere delle cose per fare creare la partita  **post:** result =true |
| **context** TorneoService::terminareTorneo(Torneo torneo,Organizzatore organizzatore)  **pre:** torneo!= null  **pre:** torneo.stato==InCorso  **pre:** torneo.organizzatore== organizzatore | **post:** torneo.stato== TorneoCompletato  **post:** result =true |
| **context** TorneoService::toglierePartecipanti(Torneo torneo,Giocatore giocatore, Organizzatore organizzatore)  **pre:** torneo.organizzatore== organizzatore  **pre:** torneo!= null  **pre:** torneo.stato==AttesaIscrizioni  **pre:** torneo.giocatori.include(giocatore) | **post:** torneo.giocatori=@pre.torneo.giocatori - giocatore  **post:** result =true |
| **context** TorneoService::visualizzaProfiloUtente(Torneo torneo,Giocatore giocatore, Organizzatore organizzatore)  **pre:** organizzatore !=null and torneo.organizzatore== organizzatore  **pre:** torneo!= null  **pre:** torneo.giocatori.include(giocatore) | **post:** result= UtenteService.findByUsername(giocatore.username,Ruolo.Giocatore) |

|  |  |
| --- | --- |
| PartitaService | Questo servizio riguarda tutti i servizi che servono per gestire la singola partita |
| Pre-Condizione | Post-Condizione |
| **context** PartitaService::aggiungereRisultato(String risultato,Partita partita,Torneo torneo, Organizzatore organizzatore)  **pre:** partita !=null  **pre:** organizzatore !=null and organizzatore= torneo.organizzatore  **pre:** torneo != null  **pre:** torneo.Partita.include(partita) | **post :** partita.winner=risultato |

|  |  |
| --- | --- |
| ModeratoreService | Questo servizio riguarda tutti i servizi che servono per gestire le operazioni del moderatore |
| Pre-Condizione | Post-Condizione |
| **context** ModeratoreService::login(Moderatore moderatore)  **pre:** moderatore.username !=null  **pre:** moderatore.password!=null  **pre:** ModeratoreService.findByEmail(moderatore.username)!=null  **pre:** moderatore.password ==  ModeratoreService.findByUsername(moderatore.username).password | **post:** sessions.get(moderatore.username)!=null (vabbe e presente in session) //FIXME la sintassi non e perfettamente corretta  **post:** result=true |
| **context** ModeratoreService::logout(Moderatore moderatore)  **pre:** moderatore.username!=null  **pre:** sessions.get(moderatore.username)!=null | **post :** sessions= @pre.sessions - sessions.get(moderatore.username)  //HACK una possibile soluzione |
| **context** ModeratoreService::bannare(Moderatore moderatore,Utente utente, Ruolo ruolo)  **pre:** ModeratoreService.findByUsername(moderatore.username)!=null  **pre:** UserService.findByUsername(utente.username,ruolo)!=null | **post:** utente.stato= utente.bannato  **post:** result= true |
| **context** ModeratoreService::sbannare(Moderatore moderatore,Utente utente, Ruolo ruolo)  **pre:** ModeratoreService.findByUsername(moderatore.username)!=null  **pre:** UserService.findByUsername(utente.username,ruolo)!=null | **post:** utente.stato= utente.attivo  **post:** result= true |
| **context context** ModeratoreService::accettare(Moderatore moderatore,Organizzatore organizzatore, Ruolo ruolo)  **pre:** ModeratoreService.findByUsername(moderatore.username)!=null  **pre:** UserService.findByUsername(organizzatore.username,Ruolo.Organizzatore).stato=InVerifica  **pre:** UserService.findByUsername(organizzatore.username,Ruolo.Organizzatore)!=null | **post:** organizzatore.stato= organizzatore.attivo  **post:** result= true |
| **context context** ModeratoreService::rifiutare(Moderatore moderatore,Organizzatore organizzatore, Ruolo ruolo)  **pre:** ModeratoreService.findByUsername(moderatore.username)!=null  **pre:** UserService.findByUsername(organizzatore.username,Ruolo.Organizzatore).stato=InVerifica  **pre:** UserService.findByUsername(organizzatore.username,Ruolo.Organizzatore)!=null | **post:** organizzatore.stato= organizzatore.rifiutato  **post:** result= true |

|  |  |
| --- | --- |
| GiocatoreService | questo servizio fornisce i servizi specifici per i giocatori |
| Pre-Condizione | Post-Condizione |
| **context** GiocatoreService::sostituireMembroTeam(Pokemon pokemon,int posizione, Giocatore giocatore )  **pre:** giocatore!=null and Giocatore.exists(giocatore) | **post :** UtenteService.findByUsername(giocatore.username,Ruolo.giocatore).pokemons[posizione]==pokemon  **post:** result=true |

|  |  |
| --- | --- |
|  |  |
| Pre-Condizione | Post-Condizione |