Università degli Studi di Napoli Parthenope

Corso: Programmazione 3

GYMtool

February 28, 2021

Autore	*Matricola*
Cozzolino Raoul	0124001690
Solmonte Sergio	0124001785

Indice

1 Descrizione del progetto

Si vuole realizzare un'applicazione web che consente la creazione di programmi di allenamento in modo semplice ed intuitivo agli utenti registrati alla piattaforma prevedendo la possibilità di consultare le informazioni riguardo ai vari esercizi e di utilizzare diversi tool quali:

- Calcolo del BMI
- Calcolo del BF
- Tabelle nutrizionali
- Calcolo kcal

Il nome scelto per l'applicazione è GYMtool.

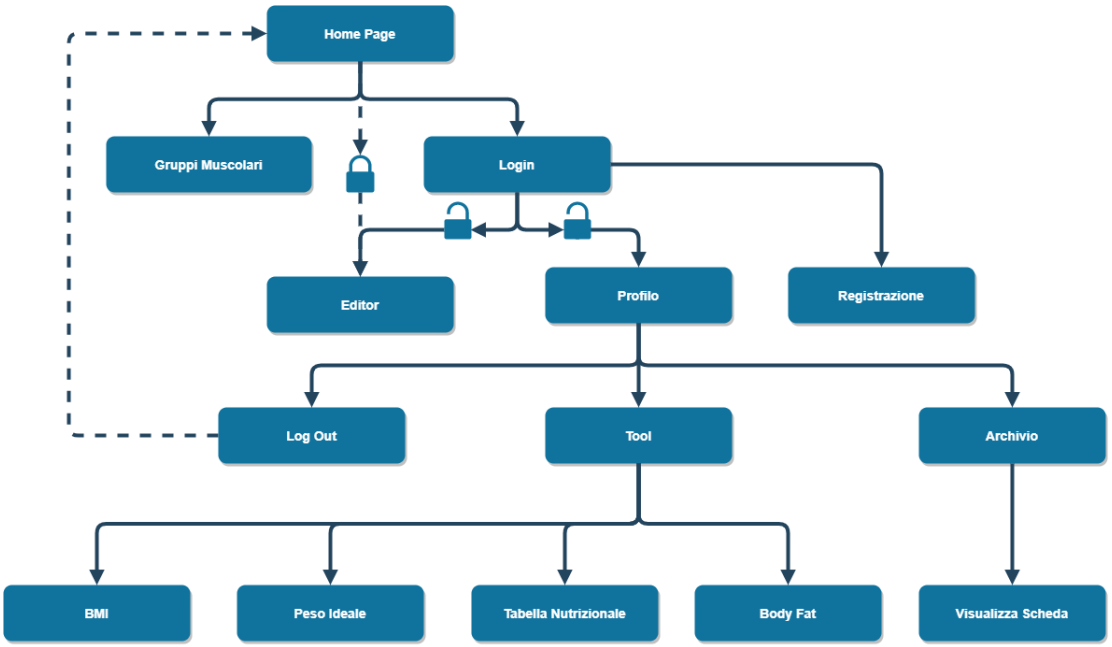
2 Scelte progettuali

L'applicazione è stata realizzata su base **Java EE 8** utilizzando le Servlet ovvero oggetti che operano su un web server (o application server) permettendo la creazione di applicazioni web. E' stato scelto come application server **Glassfish** in quanto offre pieno supporto alle specifiche **Java EE 8**.
Inoltre si è scelto l'utilizzo della pagine **JSP** ovvero una tecnologia che consente lo sviluppo della logica di presentazione fornendo contenuti dimanici in formato HTML grazie all'utilizzo di tag speciali inclusi nell documento html.
La libreria di tag utilizzata è **JSTL** che consente la manipolazione e la visualizzazione dei dati dell'applicazione all'interno delle pagine JSP. E' stato utilizzato **EL** (Expression Language) per semplificare l'accesso alle informazioni dell'applicazione da parte delle pagine JSP. Per quanto riguarda la persistenza delle informazioni è stato scelto come Database Server **MariaDB Server**, quindi un database relazionale MySQL.
Inoltre l'applicazione è stata realizzata rispettando i principi **SOLID**.
Ai fini della progettazione sono stati utilizzati i seguenti **Design Pattern**:

- **Mediator**
- **Factory**
- **Strategy**
- **Singleton**
- **DAO**

Le motivazioni riguardo l'utilizzo di ciascun pattern sono riportate nel relativo capitolo.

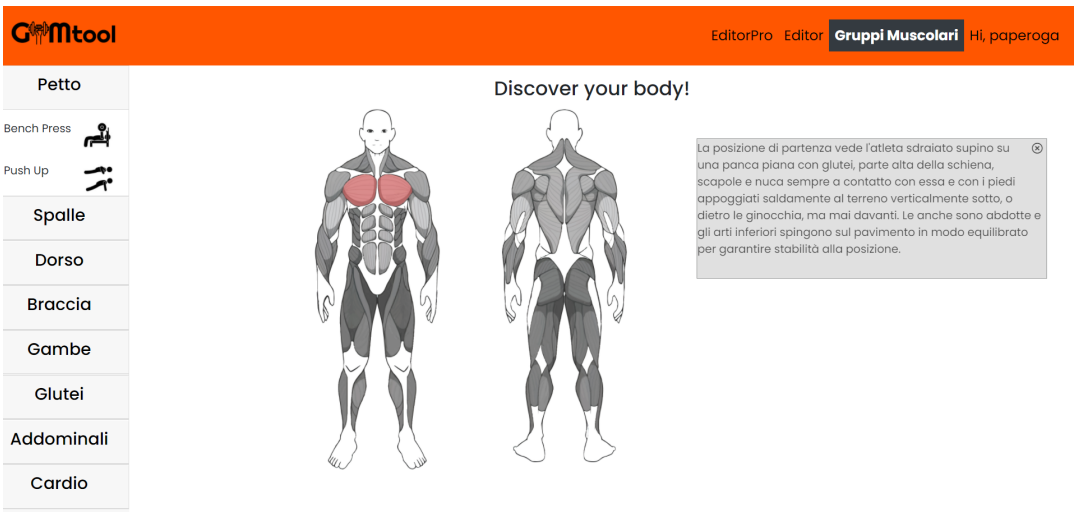
3 Schema delle operazioni



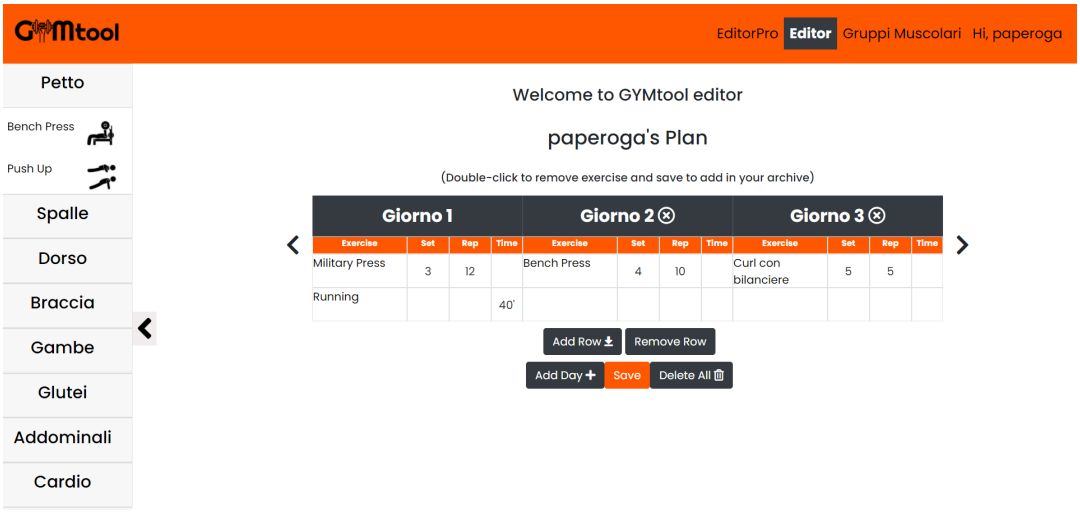
La pagina principale è l'Homepage dalla quale, come mostrato nella figura precedente, sono accessibili tutte le funzioni che saranno implementate dalla web app. In particolare la pagina **Gruppi Muscolari** riporta l'elenco di tutti gli esercizi organizzati per gruppi muscolari e con le relative descrizioni. Autenticandosi nella pagina di Login è possibile accedere a funzioni quali:

- **Editor** dei programmi di allenamento
- **Profilo** personale dell'utente
- **Archivio** dei programmi di allenamento
- **Tool** quali BMI, BF, Peso ideale, Tabelle nutrizionali

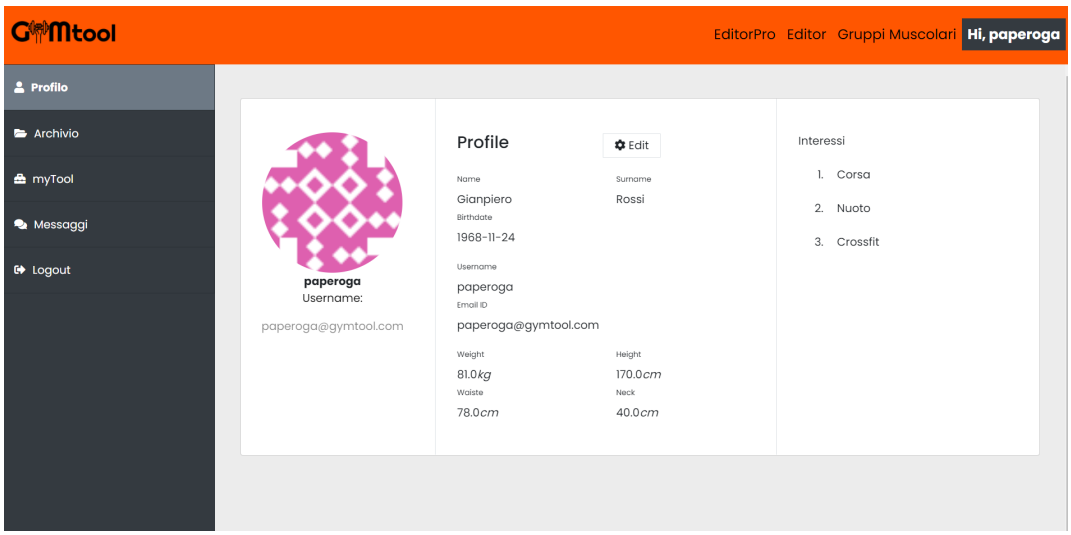
Come è possibile notare la selezione del gruppo musolare e dell'esercizio è contestualizzata dall'anatomia al centro dello schermo che evidenzierà il gruppo muscolare selezionato riportando le informazioni principali riguardo l'esecuzione del suddetto esercizio.



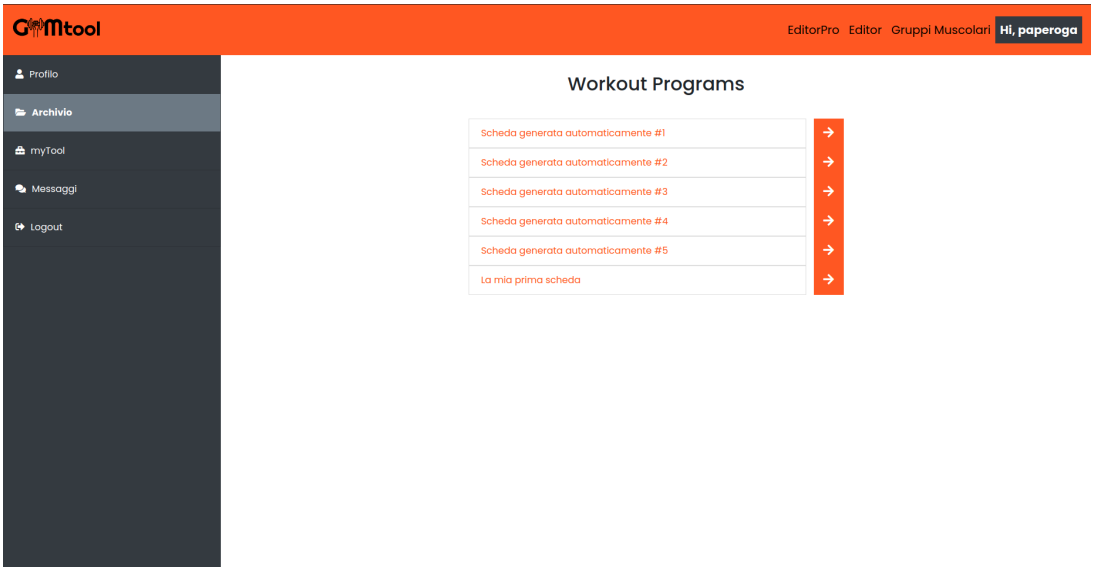
L'editor consente la creazione dei programmi attraverso un'interfaccia intuitiva che grazie ad un semplice *Drag and drop* consente di aggiungere gli esercizi alla tabella che rappresenta il programma. Il numero di giorni e di esercizi è facilmente rimodulabile grazie ai relativi bottoni. La navigazione nei vari giorni può avvenire con le frecce posti ai lati della tabella. Una volta ultimata la realizzazione della scheda è possibile effettuarne il salvataggio cliccando sul tasto **Save**.



Nel profilo personale è possibile consultare e modificare le proprie informazioni che comprendono inoltre le dimensioni fisiche. Queste dimensioni fisiche saranno poi utilizzate dai vari tool implementati grazie ad API.



Nell'archivio si trovano tutti i programmi di allenamento salvati dall'utente che è possibile consultare con un semplice click.
I tool sono realizzati grazie ad API che a partire dalle informazioni anagrafiche e fisiche dell'utente consentono di calcolare i valori indicati in precedenza.

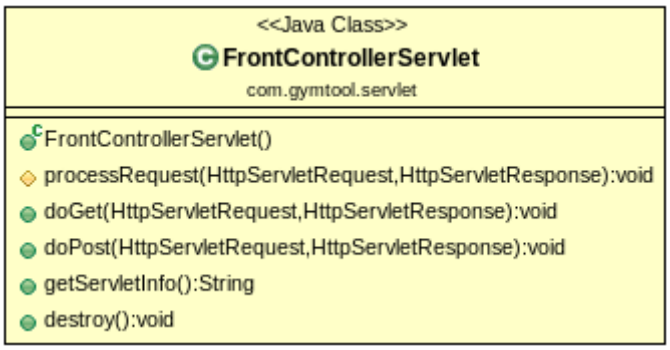


4 Design Pattern

Per i fini accademici a cui è rivolto questo progetto, sono stati utilizzati Design Pattern descritti dalla **Gang of Four** fatta eccezione per il DAO (Data Access Object), non presente, ma utilizzato per semplificare l'iterazione tra il Database e la logica di business dell'applicazione grazie alle funzioni **CRUD** (Create, Retrive, Update, Delete).
L'utilizzo dei Design Pattern permette di avere delle linee guida per l'organizzazione del software dato che rappresentano delle soluzioni generali e riutilizzabili a problemi che si verificano frequentemente.
Un corretto utilizzo dei suddetti rende il software più snello e comprensibile, favorendone il riuso e la sua manutenzione. Di seguito è riportata un'analisi specifica per ogni pattern.

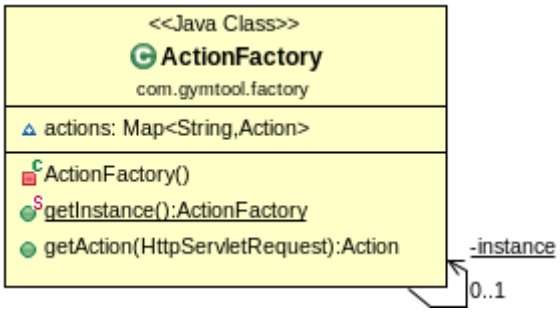
4.1 Front Controller

Si tratta di un pattern comportamentale basato su oggetti e viene utilizzato per permettere lo scambio di messaggi tra diversi attori tramite un intermediario. In questo modo gli attori sono collegati indirettamente tramite un intermediario. Disaccoppiare gli attori consente di gestire meglio una serie di problematiche come: centralizzazione delle connessioni, modifiche più rapide, semplificazione dei protocolli.
Viene utilizzato per fornire un meccanismo di gestione delle richieste centralizzato, in modo tale che tutte le richieste vengono gestite da una singola entità. Il front controller viene utilizzato per funzioni essenziali come l'autorizzazione, l'autenticazione, il log-in oppure per passare le richieste alla pagina corretta.
Il front controller è quindi il responsabile di tutte le richieste del sito. Nelle applicazioni web è molto utile perchè consente agli sviluppatori di riutilizzare il codice senza doverlo aggiungere altre volte.
Il Front Controller rappresenta un caso particolare del pattern Mediator.



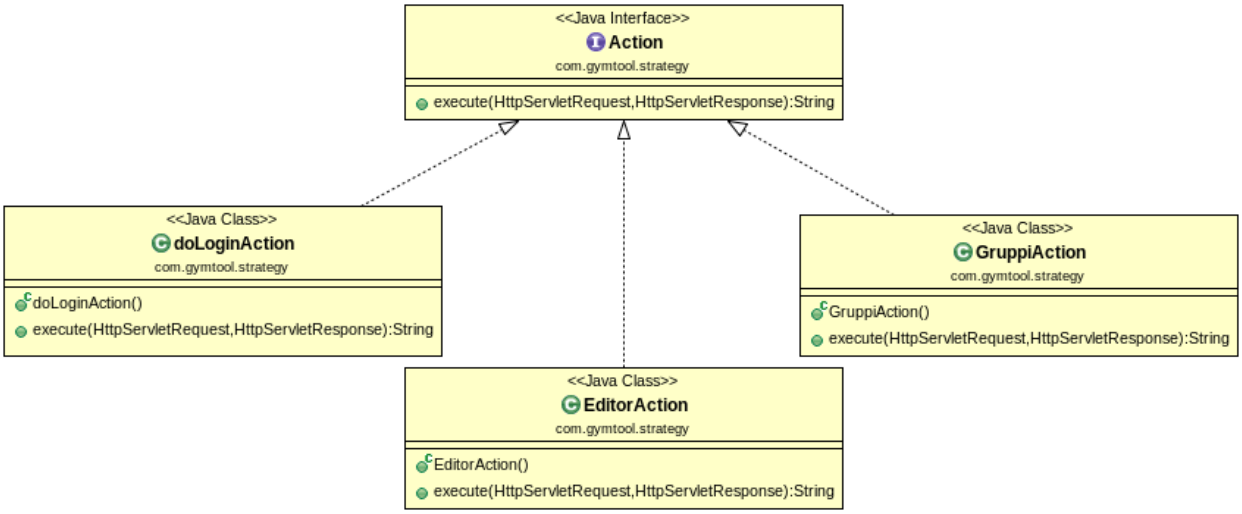
4.2 Factory

Il Factory pattern è uno dei più usati design pattern in Java. Questo tipo di design pattern fa parte dei pattern creationali ed è rappresenta uno dei modi migliori per creare un oggetto. Nel Factory viene creato un oggetto senza esporlo la logica della creazione al client facendo riferimento al nuovo oggetto creato usando un'interfaccia comune. Lascia il compito alle sottoclassi di decidere quale classe inizializzare e quindi quale oggetto instanziare. Nel progetto il pattern factory viene utilizzato alla classe Action Factory. Ad ogni richiesta ricevuta dal front controller, richiama l'azione giusta e la restituisce.



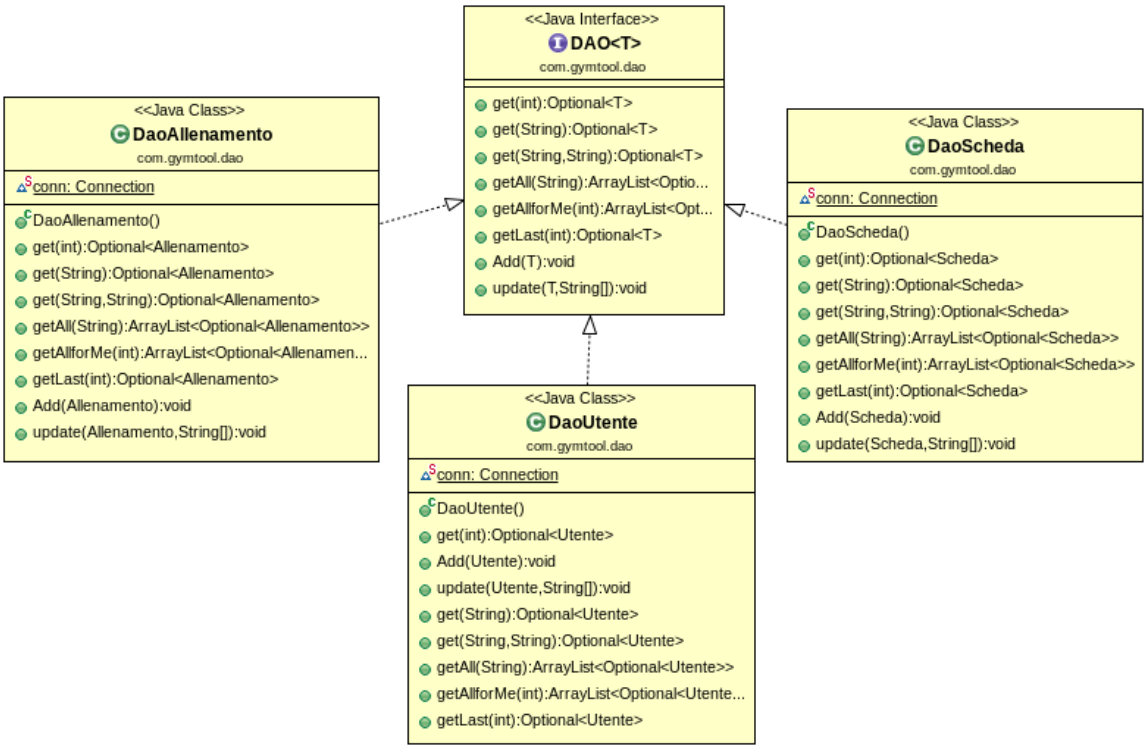
4.3 Strategy

Si tratta di un pattern comportamentale basato su oggetti e viene utilizzato per definire una famiglia di algoritmi, incapsularli e renderli intercambiabili. Il client definisce l'algoritmo da utilizzare, incapsulandolo in un contesto, il quale verrà utilizzato nella fase di elaborazione. Il contesto detiene i puntamenti alle informazioni necessarie al fine della elaborazione, cioè dati e funzione. Gli algoritmi definiscono il modo in cui vengono elaborate le informazioni. Possiamo creare delle classi di algoritmi che implementano in modo diverso uno stesso algoritmo oppure possiamo creare delle nuove classi di algoritmi.
Nel progetto l'algoritmo che viene eseguito è quello delle implementazioni dell'interfaccia *Action*.
Tramite l'input ricevuto dal front controller, l'Action Factory richiama l'implementazione di action giusta.



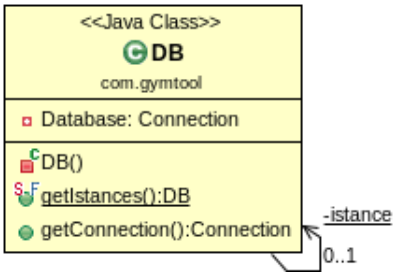
4.4 DAO

Il pattern DAO è usato per separare la logica di business dalla logica di acceso ai dati. Infatti, i componenti della logica di business non dovrebbero mai accedere direttamente al database: questo comporterebbe scarsa manutenibilità. Solo gli oggetti previsti dal pattern Dao possono accedervi. Inoltre, se dovessimo modificare il tipo di memoria persistente utilizzata non sarà necessario stravolgere il codice della nostra applicazione, ma basterà modificare i DAO utilizzati. Chiunque voglia poter accedere al DataBase dovrà usare la relativa interfaccia Dao. Se dovessimo cambiare persistenza, sarà suffficiente creare una nuova implementazione delle interfacce Dao, senza stravolgere tutto.
Nel nostro progetto, il DAO e le sue implementazioni sono presenti nell'omonimo package.



4.5 Singleton

Il singleton è un pattern che garantisce una sola istanza di una determinata classe. Per esempio quando si desidera avere solo un Window Manager oppure una sola Coda di Stampa oppure un unico accesso al database si può usare il pattern singleton. Nel nostro caso il singleton è stato implementato con il metodo **Pattern Singleton Lazy**, cioè sono stati dichiarati una variabile statica dello stesso tipo della classe, il costruttore privato (quindi nessuno lo può chiamare, e di conseguenza nessuno può istanziare la classe, al di fuori di essa) e un metodo statico che restituisce l'unica istanza esistente della classe e, nel nostro caso, un'unica connessione con il database.



5 Miglioramenti e future implementazioni

Come verrà naturale chiedersi, questo non è il metodo migliore per la realizzazione di una web application. L'utilizzo esclusivo delle pagine jsp per la visualizzazione può essere sostituito da un front-end in html e da un back-end formato da API.
Il front controller realizzato tramite pattern mediator può essere sostituito dal RequestMapping del framework Spring MVC in modo da rispettare la struttura Model-View-Controller, mentre nel nostro caso la parte del view viene accorpata nel controller che gestisce le action per la visualizzaizone.
Questa struttura è stata realizzata con l'intento di rispettare i requisti accademici necessari imposti per una web application e questa è da noi reputata la migliore opzione.
Nelle future implementazioni verrano aggiunti lo scambio dei messaggi tra utenti e ulteriori tool che rendono la web app unica nel suo genere.

6 Editor Pro

La pagina editor Pro permette ad un utente registrato di ottenere una scheda di allenamento in maniera automatica. Dopo aver completato un breve questionario sulle sue volontà e sulla sua disponibilità , verrà calcolato il valore massimo che potrà raggiungere la scheda in base alla complessità degli esercizi e (in future implementazioni) dalla quantità di serie e ripetizioni.
L'algoritmo sostanzialmente segue 4 passi:

1. Inizializzazione dei parametri;
2. Generazione casuale di una scheda prototipo con relativo calcolo del valore;
3. Aggiornamento degli esercizi nella scheda prototipo
4. Assegnazione della scheda all'utente.

Il passo fondamentale è l'aggiornamento dei valori all'interno della scheda generata, poichè l'algoritmo segue l'idea di migliorare il suo valore ad ogni iterazione.
La scheda sarà pronta quando il valore raggiunto dalla scheda – il valore richiesto dall'utente sarà inferiore ad un epsilon.
Questo algoritmo è ancora agli albori dato che non è il perno centrale della nostra web app, però potrà avere una sua importanza per le persone sprovviste di preparatore atletico o utenti alle prime armi, ovviamente dopo i giusti miglioramenti e sviluppi.

7 Ringraziamenti

Per la realizzazione e lo sviluppo di tale progetto, si ringrazia:

- Il profressore **Angelo Ciaramella** per chiarimenti e consigli sull'utilizzo dei design Pattern;
- Il professore **Raffaele Montella** per chiarimenti e consigli sulla struttura della web app e sulla tipologia di database utilizzato.
- Teoria e applicazioni dei design Pattern
- Manuale di Java 9, Hoepli, C. De Sio Cesari