

System Design Document



## Partecipanti

|  |  |
| --- | --- |
| Partecipante | Matricola |
| Salvatore Fasano | 0512105068 |
| Mariantonia Candela | 0512105374 |

# Sommario

# 

# 1. Introduzione

## 1.1 Obiettivi del sistema

Sempre più atleti, amanti del body building, ricercano personal trainer con cui iniziare un percorso di miglioramento del proprio fisico con scopo amatoriale o di partecipare a delle gare. Tuttavia c’è una limitazione notevole: potrebbero non essere presenti pt in un determinato paese o semplicemente si preferisce essere seguiti da un pt che si trova distante. L’obiettivo di urCoach è questo: permettere ad atleti di entrare in contatto con diversi pt sparsi sul territorio e poter selezionare quello che più fa al proprio caso (perchè specializzato in una determinata area, es. dimagrimento). urCoach inoltre fa una selezione in ingresso dei pt, in modo da permettere l’accesso alla piattaforma solo a coloro che sono davvero preparati valutando il loro curriculum. Ogni pt potrà, una volta accettato dai recruiter, pubblicare i propri servizi sulla piattaforma. Gli atleti potranno invece registrarsi sulla piattaforma, ricercare vari pacchetti ed acquistare i pacchetti messi in vendita.

## 1.2 Design goals

### 1.2.1 Criteri di performance

**Throughput**

* Il sistema deve essere in grado di supportare fino a 100000 visite mensili

**Tempi di risposta**

* Il sistema dovrà eseguire le query all’interno del database in un tempo inferiore a 1 secondo, assicurando così una risposta rapida ad una richiesta di accesso o di ricerca all’interno della piattaforma

### 1.2.2 Criteri di affidabilità

**Robustezza**

* Eventuali errore di input da parte dell’utente devono essere notificati con opportune segnalazione

**Sicurezza**

* Utilizzo di certificato SSL per offrire una connessione su protocollo HTTPS, offrendo quindi una comunicazione su un canale sicuro
* La piattaforma, mediante l’utilizzo di servizi come Stripe/Paypal garantisce pagamenti sicuri

**Disponibilità**

* Uptime garantito 99.9%

### 1.2.3 Criteri di costo

* Bisognerà cercare di abbattere i costi nelle fasi di sviluppo, manutenzione ed amministrazione per poter allocare maggiori risorse al deployment

### 1.2.4 Criteri di manutenzione

**Portabilità**

* Browser Compatibility: l’interfaccia deve rimanere consistente anche in browser diversi, in modo da garantire la stessa esperienza nel caso in cui si cambi browser
* Responsive Design: la piattaforma deve essere utilizzabile su dispositivi di grandezze diverse

### 1.2.4 Criteri per l’utente finale

**Usabilità**

* Ripetizione degli UI Pattern: l’utente dopo pochi utilizzi della piattaforma riconosce gli elementi dell’interfaccia ed il loro compito, inoltre la ripetizione di tali pattern gli permette di comprendere velocemente l’utilizzo di nuove funzionalità
* Call-to-Action chiare: ogni button presente nell’interfaccia contiene un testo che indica chiaramente l’azione che svolgerà e non può essere frainteso
* Contrasto: la piattaforma segue le linee guida nell’utilizzo dei colori per fornire un contrasto testo/background ottimale in modo da garantire la leggibilità del testo

## 1.3 Definizioni, acronimi ed abbreviazioni

* PT = Personal Trainer
* Piattaforma = Applicazione web
* CRUD = Create, Read, Update, Delete

## 1.4 Riferimenti

## 1.5 Panoramica

Nel documento verranno affrontati l’analisi delle architetture di sistemi simili, la decomposizione in sottosistemi del sistema proposto con la definizione della strategia di deploy e le condizioni limite. Verranno quindi definiti i servizi esposti da ciascun sottosistema.

# 2. Architettura di sistemi simili

Non ci sono piattaforme simili ad urCoach. Attualmente pt che offrono online coaching utilizzano i social network per farsi conoscere e stabilire relazioni con gli atleti, nel caso in cui l’atleta sia interessato ad intraprendere un percorso con un pt dovrà contattarlo tramite email e così avanzerà il rapporto tra i due. In alcuni casi vengono anche utilizzati sistemi creati per freelancer come Fiverr. Fiverr è probabilmente costruito con un’architettura MVC, la tecnologia di backend usata è Ruby mentre per il frontend utilizza un framework Javascript (Ract). Non si possono avere informazioni sulla gestione dei dati persistenti perchè hanno deciso di dividere il backend dal frontend: su un server risiede il backend con un probabile DBMS, il backend realizza delle REST API ed il frontend, posizionato su un server differente, sfrutta tali API per poter accedere ai dati con lo scopo di mostrarli, crearne di nuovi, aggiornarli o eliminarli (CRUD operation).

# 

# 3. Architettura del sistema proposto

## 3.1 Panoramica

urCoach è un’applicazione distribuita che fornisce un’interfaccia web. Essa, inoltre, ricorre all’utilizzo di un database relazionale per il salvataggio dei dati persistenti.

## 3.2 Decomposizione in sottosistemi

Si è deciso di basare urCoach su un’architettura MVC organizzata in modo da minimizzare l’accoppiamento e favorire un’alta coesione, vediamo il sistema suddiviso su tre livelli:

* Il livello di presentazione che contiene solo il Frontend, ovvero l’interfaccia che gli utenti utilizzeranno per interagire col sistema
* Il livello di Business Logic, ovvero tutta la logica applicativa, troviamo infatti la gestione di ordini, servizi, recruiting e degli utenti
* Il livello Persistenza col sottosistema Data Access che ha l’accesso al database per prelevare o modificare dati.

Realizziamo quindi la View al livello di Presentazione, il Control è localizzato totalmente al livello di Business Logic mentre il Model è diviso tra il livello di Business Logic e Persistenza: nel primo troviamo le definizioni di tutte le entità mentre il secondo contiene i metodi di accesso al database per poter prelevare tali entità o eventualmente modificarle.

## 3.3 Hardware/software mapping

urCoach consiste di un’applicazione distribuita installabile su un qualsiasi server in grado di eseguire Java e MySQL. L’architettura scelta prevede l’interazione della piattaforma con un database: date le risorse si ritiene opportuno installare sia il web server che il database in un unico nodo. Il sistema sarà accessibile tramite comuni browser web installati sui dispositivi a disposizione degli attori.

## 

## 3.4 Gestione dei dati persistenti

Per la gestione dei dati persistenti si è deciso di utilizzare MySQL, questo per due motivi principali:

* I dati sono ben strutturati e per questo si prestano particolarmente ad essere memorizzati in questo DBMS
* Per abbattere ulteriormente i costi, strutturare dei file avrebbe richiesto maggior lavoro e quindi più costi; inoltre MySQL è gratuito e di conseguenza comporta un costo in meno

Infine ovviamente MySQL ci permette di trattare i dati ed effettuare operazioni con estrema semplicità e di aggiungere un ulteriore livello di sicurezza per l’accesso ai dati.

## 

## 3.5 Controllo degli accessi e sicurezza

Il controllo degli accessi è garantito tramite l’utilizzo di username e password per gli utenti del sistema che hanno possibilità di creare o modificare gli oggetti che modellano entità di dominio, così da prevenire accessi non autorizzati ad informazioni sensibili. Sottolineiamo che il sistema non fornirà un metodo di recupero della password.. Si ricorrerà all’utilizzo della sessione del server per tenere traccia dell’utente loggato. Per questioni di efficienza, la sessione sarà attiva per soli 30 minuti dopo l’ultima interazione dell’utente col sistema. Il salvataggio delle password nel database sarà cifrato con [CIFRATURA]. Le operazioni che gli utenti dell’applicazione web possono effettuare sugli oggetti sono riportate nella tabella che segue:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Utente** | **Recruiting** | **Servizi** | **Ordini** |
| |  | | --- | | **Personal Trainer registrato** | | · Login  · Visualizzare il proprio profilo  · Modificare i propri dati personali  · Eliminare il proprio account e i relativi servizi dalla piattaforma |  | · Aggiungere un nuovo servizio alla piattaforma  · Modificare un proprio servizio  · Eliminare un proprio servizio  · Visualizzare tutti i servizi inseriti |  |
| **Personal Trainer**  **non registrato** | · Registrazione |  |  |  |
| **Atleta registrato** | · Login  · Visualizzare le sue informazioni personali  · Modificare i propri dati personali  · Eliminare il suo account dalla piattaforma  · Visualizzare lo storico dei suoi ordini |  | · Visualizzare il catalogo  · Effettuare ricerche all’interno del catalogo | · Aggiungere prodotti al proprio carrello  · Visualizzare il carrello  · Eliminare i servizi presenti nel carrello  · Acquistare i prodotti indicati nel carrello (checkout) |
| **Atleta**  **non registrato** | · Registrazione |  | · Visualizzare il catalogo  · Effettuare ricerche all’interno del catalogo | · Aggiungere prodotti al proprio carrello  · Visualizzare il carrello  · Eliminare i servizi presenti nel carrello |
| **Recruiter** | · Login | · Accettare o rifiutare l’iscrizione di un personal trainer alla piattaforma |  |  |
| **Gestore degli ordini** | · Login |  | · Visualizzare lo storico degli ordini globale |  |

## 

## 3.6 Controllo flusso globale del sistema

Il sistema adotta un controllo del flusso globale di tipo thread-driven, questo perché il web container (Tomcat) web permette l’interazione concorrente tra la WebApp e più client tramite l’intercettazione di eventi generati proprio da questi ultimi.

Questa soluzione permette al sistema di poter rispondere a più utenti contemporaneamente ma richiede che gli accessi in scrittura ai dati persistenti avvengano sequenzialmente, gestendo opportunamente le sezioni critiche. In generale, ogni richiesta da parte di un utente verrà eseguita in un thread dedicato.

## 3.7 Boundary condition

### 3.7.1 Startup sistema

|  |  |
| --- | --- |
| Nome Caso d’uso | Startup Sistema |
| Attori Partecipanti | Amministratore |
| Sequenza degli eventi | |  |  | | --- | --- | | **Amministratore** | **urCoach** | | 1. L’amministratore scrive in console “./startup.sh” |  | |  | 1. Il sistema inizia l’esecuzione creando manager ed inizializzandoli. | |
| Pre-condizioni | L’amministratore è connesso tramite una shell al server che ospita il web server |
| Post-condizioni | Il sito è raggiungibile da qualsiasi web browser connesso alla rete |

## 

### 3.7.2 Shutdown Sistema

|  |  |
| --- | --- |
| Nome Caso d’uso | Shutdown Sistema |
| Attori Partecipanti | Amministratore |
| Sequenza degli eventi | |  |  | | --- | --- | | **Amministratore** | **urCoach** | | 1. L’amministratore scrive in console “/shutdown.sh” |  | |  | 1. Il sistema esegue lo script di shutdown | |
| Pre-condizioni | L’amministratore è connesso tramite una shell al server che ospita il web server.  Il web server è stato precedentemente avviato. |
| Post-condizioni | Il sito non è più raggiungibile. |

### 

# 4. Subsystem services

|  |  |
| --- | --- |
| **Sottosistema** | **Descrizione** |
| **Gestione Ordini** | Permette di visualizzare il carrello, aggiungere e rimuovere un servizio da quest’ultimo. Consente di inserire dati di pagamento e ad un Atleta registrato di effettuare il checkout. |

|  |  |
| --- | --- |
| **Servizio** | **Descrizione** |
| **visualizzaCarrello** | Consente di visualizzare i servizi aggiunti nel carrello e il prezzo totale |
| **aggiungiProdottoAlCarrello** | Consente di aggiungere un servizio al carrello |
| **rimuoviProdottodAlCarrello** | Consente di rimuovere un servizio dal carrello |
| **inserisciDatiPagamento** | Permette di inserire i dati della carta di credito, necessari per effettuare il checkout |
| **acquistaServiziNelCarrello** | Permette ad un Atleta registrato di acquistare i servizi inseriti nel carrello. Dopo tale operazione il carrello verrà svuotato |
| **visualizzaStoricoOrdiniGlobale** | Permette al Gestore degli ordini di visualizzare tutti gli ordini effettuati dagli Atleti registrati |
| **visualizzaStoricoOrdini** | Permette all’Atleta di visualizzare tutti gli ordini effettuati |

|  |  |
| --- | --- |
| **Sottosistema** | **Descrizione** |
| **Gestione Servizi** | Permette l’aggiunta,la modifica e la rimozione di un servizio da parte di un Personal Trainer. Consente inoltre la visualizzazione e la ricerca all’interno di tutto il catalogo da parte di ogni Utente |

|  |  |
| --- | --- |
| **Servizio** | **Descrizione** |
| **inserimentoServizio** | Permette ad un Personal Trainer di inserire un nuovo servizio al catalogo |
| **rimuoviServizio** | Permette ad un Personal Trainer di rimuovere un servizio dal catalogo |
| **modificaServizio** | Permette ad un Personal Trainer di modificare le informazioni relative ad un servizio precedentemente inserito |
| **visualizzaCatalogo** | Permette di visualizzare i servizi presenti nel catalogo |
| **ricercaServizi** | Consente la ricerca di un servizio nel catalogo tramite una keyword |
| **filtraServiziPerCategoria** | Consente la ricerca di un servizio nel catalogo di una specifica categoria |
| **filtraServiziPerPrezzo** | Consente la ricerca di un servizio nel catalogo di uno specifico prezzo |

|  |  |
| --- | --- |
| **Sottosistema** | **Descrizione** |
| **Gestione Utente** | Consente ad un Utente non registrato di effettuare la registrazione e a un Utente registrato di effettuare il login. Permette a quest’ultimo di accedere alla proprio pagina utente per modificare i dati personali inseriti all’atto di registrazione ed inoltre è possibile eliminare definitivamente il proprio account |

|  |  |
| --- | --- |
| **Servizio** | **Descrizione** |
| **login** | Consente ad un Utente registrato di accedere al sistema tramite una coppia di username e password |
| **logout** | Consente ad un Utente loggato di rimuovere l’accesso al sistema |
| **registrazione** | Permette ad un Utente non registrato di creare una coppia di username e password per poter accedere al sistema |
| **accessoPaginaUtente** | Consente ad un Utente registrato di visualizzare le informazioni relative al proprio account |
| **modificaDatiPersonali** | Consente ad un Utente registrato di modificare le informazioni relative al proprio account |
| **rimuoviUtenteRegistrato** | Consente ad un Utente registrato di cancellare il proprio account dal sistema |

|  |  |
| --- | --- |
| **Sottosistema** | **Descrizione** |
| **Gestione Recruiting** | Consente di accettare o rifiutare un Personal Trainer che ha richiesto l’iscrizione alla piattaforma |

|  |  |
| --- | --- |
| **Servizio** | **Descrizione** |
| **rifiutoPersonalTrainer** | Rifiuta la richiesta del Personal Trainer di iscriversi alla piattaforma |
| **confermaPersonalTrainer** | Conferma la richiesta del Personal Trainer di iscriversi alla piattaforma |