GymGest

Applicazioni e Servizi Web

Ylenia Battistini - 0000936972 ylenia.battistini@studio.unibo.it
Enrico Gnagnarella - 0000926637 enrico.gnagnarella@studio.unibo.it
Matteo Scucchia - 0000926325 matteo.scucchia@studio.unibo.it

09 Settembre 2020

Contents

1 Introduzione

L'idea per questo progetto, nasce dalle nostre esperienze pregresse con il mondo della palestra e dal desiderio di poter disporre di un'applicazione che faciliti l'organizzazione e la gestione degli allenamenti, che fornisca determinate funzionalità, da noi ritenute indispensabili e dalla possibilità di sfruttare le tecnologie viste e le conoscenze impartite durante la frequenza del corso di Applicazione e Servizi Web. Lo scopo del nostro progetto, appunto, è quello di realizzare una applicazione web gestionale per palestre, la quale oltre alla normale gestione di una palestra, introduca concetti come:

- Gestione dei corsi tenuti in palestra.
- Chat private tra utenti e coach.
- Aggiunta di training all'aperto con visualizzazione percorsi attraverso GPS.

2 Requisiti

L'idea è quella di realizzare una applicazione client-server basata su stack MERN. All'interno dell'applicazione prevediamo 3 tipi di utenti:

- **Admin**: gestisce utenti e coach (inserisce coach e cancella coach/utenti) e le informazioni della palestra.
- **Coach**: l'allenatore. Ce ne possono essere più di uno. Ogni coach ha un proprio profilo con una descrizione della sua formazione. Ha un calendario in cui inserisce il proprio orario di lavoro. Quando non è nel suo orario di lavoro è possibile prenotare un personal (allenamento personalizzato). Un coach può visualizzare gli utenti e gestire le schede a loro associate.
- **Utente**: la persona iscritta alla palestra. Può visualizzare i coach e chiedere loro dei personal (lezioni private e personali) e delle schede. Ogni utente visualizza le proprie schede. Ogni utente ha un proprio profilo, che può modificare. Può iscriversi a corsi in diversi giorni della settimana e può tenere traccia dei propri allenamenti outdoor.

3 Funzionalità

Le funzionalità di base previste per il sistema sono:

- Login e registrazione degli utenti (implementando sistema di hashing delle password).
- L'admin deve poter aggiungere/rimuovere coach e rimuovere utenti. Imposta le info generali della palestra.
- Il coach deve poter modificare il proprio profilo. Può aggiungere/modificare l'orario di lavoro. Può aggiungere/modificare le schede degli utenti.
- L'utente deve poter registrarsi e accedere al sistema. Ha un profilo che può modificare. Visualizza le sue schede e i coach. Per ogni coach visualizza il calendario e può prenotare i personal solo in determinati orari (liberi, ma predefiniti). Può iscriversi a corsi e eseguire allenamenti all'aperto, dei quali sono registrati i percorsi e le performance.

Sulla base dell'approccio agile, i requisiti del sistema sono stati modificati in corso d'opera. Effettuando più fasi di analisi e design, in linea con tale metodologia, è stato deciso che si vuole mantenere un storico degli utenti e dei coach, di conseguenza l'amministratore non può più cancellare tali utenti, può solo visualizzarli e inserire nuovi coach. Poiché l'amministratore necessita di maggior controllo sui coach, è lui che ne inserisce e modifica l'orario di lavoro.

4 Design

Il design dell'applicazione è stato svolto seguendo un approccio misto. Abbiamo sicuramente utilizzato un approccio **partecipativo**, infatti tutti i membri del gruppo hanno un trascorso in palestra o attività sportive affini e di conseguenza abbiamo pensato ad una applicazione che potesse soddisfare le esigenze che abbiamo tastato con mano nei nostri anni di allenamenti. Ci siamo quindi fatti designer, sviluppatori e utenti target dell'applicazione stessa. Inoltre si potrebbe dire che abbiamo utilizzato una metodologia riconducibile all'**experience prototyping** in quanto abbiamo fatto uso di mockup per capire se le nostre idee soddisfacessero effettivamente i nostri bisogni. Per quanto riguarda il design dell'interfaccia utente, ci siamo ispirati a WhatsApp, in modo che gli utenti possano associare la loro esperienza con l'applicazione di messaggistica a quella da noi realizzata ed interagire, quindi, facilmente con la nostra UI. Anche per quanto riguarda la chat ci siamo ispirati alla chat di Instagram, una chat semplice ma nota. Dati questi elementi, la User Experience con l'applicazione risulta intuitiva e soddisfacente. E' facile completare qualsivoglia genere di task che è previsto dalle funzionalità, senza incorrere in complicazioni e senza che l'utente si trovi di fronte a comportamenti inaspettati da parte dell'applicazione. La semplicità con cui l'abbiamo progettata fa sì che si adatti al meglio allo schema mentale dell'utente.

4.1 Design partecipativo

Nello specifico del design partecipativo, gli utenti target possono essere descritti come segue.

4.1.1 Matteo

Matteo è un ragazzo che frequenta la palestra da ormai otto anni. Si è appassionato a questo mondo in seconda superiore, quando suo fratello maggiore cominciò ad andare in palestra. È sempre stato interessato all'allenamento con i pesi. Ci tiene molto alla forma fisica e cerca sempre di migliorarsi. Per farlo ha bisogno di un contatto diretto con i suoi coach al fine di chiedere consigli ed allenamenti personalizzati. Gli farebbe comodo poter prenotare attraverso un'app un allenamento singolo con i coach e poter tenere monitorato lo storico delle sue schede e dei suoi allenamenti.

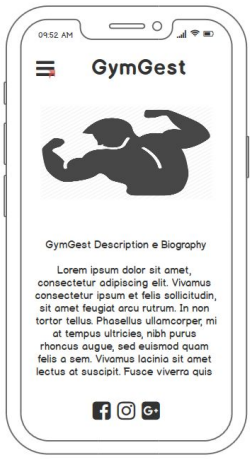
4.1.2 Enrico

Enrico è un ragazzo che pratica equitazione a livello agonistico, nello specifico sta preparando una corsa chiamata Palio del Niballo, molto famosa a Faenza. Per preparasi al meglio ha bisogno di fare preparazione atletica ed è quindi interessato all'allenamento con i pesi e al cardio. Si diletta spesso andando a correre o macinando chilometri in bicicletta nel weekend. Ad Enrico piacerebbe avere un'app al fine di poter registrare i suoi percorsi per poter monitorare lo storico dei suoi allenamenti, oltre ad avere la possibilità di richiedere piani di allenamento personalizzati consultabili facilmente.

4.1.3 Ylenia

Ylenia ha la passione per il beach tennis, ma non è mai stata attirata dall'idea dell'allenamento in palestra, che trova noioso. Quest'anno ha però deciso di iscriversi in palestra assieme alle amiche ed e' interessata ai corsi messi a disposizione. A Ylenia farebbe comodo un'applicazione per poter prenotare delle lezioni relative ai corsi, in diversi giorni della settimana. Inoltre Ylenia si sta appassionando all'allenamento funzionale e per questo avrebbe bisogno di un coach che la segua nel suo percorso, in quanto si sta affacciando per la prima volta a questo mondo.

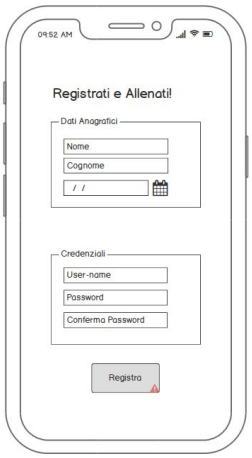
4.2 Experience prototyping e mockup



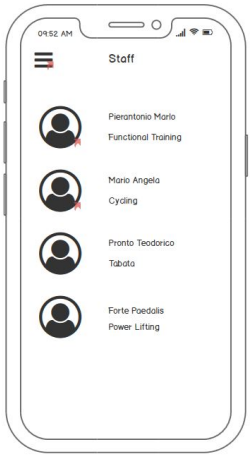
(a) Homepage

(b) Homepage drawer

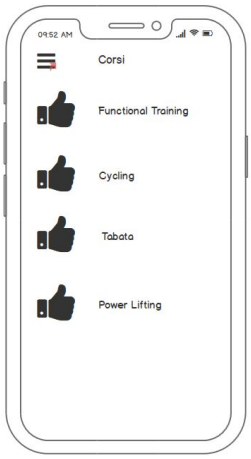
(a) Login



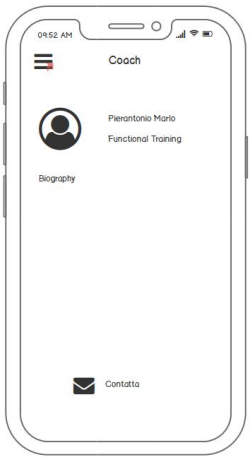
(b) Registrazione



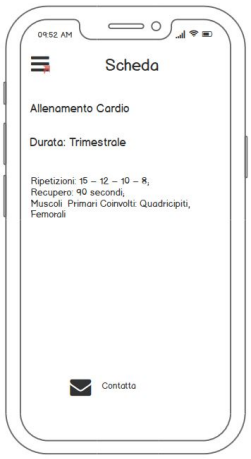
(a) Utenti e coach



(b) Corsi



(a) Profilo e dettaglio



(b) Schede

Nei mock-up sopra mostrati abbiamo cercato di rappresentare gli aspetti più salienti dell'applicazione. Essi sono stati realizzati in fase di design, per poter essere sottoposti a valutazione da parte degli utenti target, secondo l'approccio standard dell'experience prototyping. Di conseguenza alcuni di essi sono cambiati in corso d'opera, mantenendo però una logica coerente con i feedback raccolti.

5 Tecnologie

Le scelta delle tecnologie per la realizzazione di GymGest è ricaduta sullo stack MERN. A seguito dello studio dei possibili stack, la nostra scelta è stata indirizzata verso React, per un semplice interesse personale e la voglia di approfondirlo, non essendoci mai entrati in contatto nonostante le esperienze lavorative fuori dall'università.

5.1 Stack MERN

Lo stack MERN, richiede quindi di garantire la persistenza delle informazioni attraverso MongoDB [5], fa uso di Node.js [2] ed Express [1] per creare facilmente un server Web scritto in Javascript e di utilizzare React [3] come framework Javascript lato client.

5.2 Material UI

Per quanto riguarda l'interfaccia grafica abbiamo poi fatto uso del framework Material UI [4]. Tale scelta è stata presa per due motivi:

1. Material UI mette a disposizione componenti che hanno un'interfaccia e un comportamento che sono facilmente riconoscibili in quanto largamente utilizzati in contesti di uso quotidiano (Google stesso fa uso di Material Design), di conseguenza risulta ottima la Human Computer Interaction, in quanto per l'utente si tratta di un'interfaccia composta da componenti familiari e intuitivi

2. i componenti di Material UI sono estremamente estetici

5.3 Socket.io

Abbiamo utilizzato Socket.io [6] come libreria per la comunicazione bidirezionale ed event-based tra client e server. Essa è stata impiegata nella realizzazione della chat e delle notifiche, per permettere l'aggiornamento in tempo reale delle applicazioni.

6 Codice

Il codice è stato suddiviso in client e server. Il server si occupa di risolvere le richieste del client, inviando i dati recapitati dal database e gestisce socket.io, il client è rappresentato dall'interfaccia utente e dalla logica applicativa.

6.1 Server

Lato server si è seguita la suddivisione vista a lezione di models, controllers e routes. Inoltre è stato implementato un meccanismo di controllo degli accessi con JsonWebToken, che permette di assegnare un token univoco ad ogni utente in fase di login e di conseguenza è possibile controllare il ruolo che l'utente ricopre ad ogni chiamata HTTP. Questo perché, ovviamente, a ruoli diversi corrispondono privilegi di accesso differenti.

6.2 Client

Lato client si è fatto largo uso di un paradigma funzionale per passare callback ai diversi components, questo ci ha permesso di rendere generali alcuni componenti complessi. Il client fa uso di session storage per salvare variabili di sessione. In fase di registrazione il client si occupa di hashare le password mediante algoritmo SHA-512.

7 Test

Nella fase iniziale di analisi e definizione dei requisiti sono stati effettuati test con gli utenti target per verificare se i mock-up e la logica applicativa si adattasse alle esigenze. I feedback rinvenuti sono stati di fondamentale importanza per lo sviluppo vero e proprio dell'applicazione. In seguito alla realizzazione dell'app, questa è stata testata più e più volte con utenti reali, non solo per verificarne il corretto funzionamento, ma anche per analizzarne la Human Computer Interaction, che si è rivelata molto buona. L'applicazione risulta intuitiva nel suo complesso e rispecchia gli obiettivi prefissati.

8 Deployment

Sebbene abbiamo provato a dockerizzare l'applicazione, per via di problemi tecnici non ci siamo riusciti. Abbiamo quindi deciso di inserire all'interno del repository una cartella `collections` che contiene tutte le collections esportate ed un file `mongo.txt` contenente i comandi per importarle facilmente nel database. Ovviamente è necessario cambiare i percorsi delle collections. Per avviare l'applicazione è necessario:

- da una prima shell, avviare il server, posizionandosi sulla cartella `/gymgest/server` e lanciare il comando `node index.js`
- da una seconda shell, posizionarsi in `/gymgest/gymgest` ed eseguire il comando `npm run build` per creare una build ottimizzata dell'applicazione
- usare il package `serve` per lanciarla
- se non si dispone di `serve` è necessario installarlo con `npm install serve -g`
- avviare l'applicazione con `serve -s build` all'interno della cartella `/gymgest/gymgest`

9 Conclusioni

Dal nostro punto di vista, ci riteniamo soddisfati dell'applicazione realizzata, in quanto l'app risulta di facile utilizzo e l'esperienza d'uso risulta molto gradevole. Gymgest, oltretutto, soddisfa tutte le richieste riscontrate da parte degli utenti e definite nella fase iniziale di presentazione del progetto. Questo progetto è stato per noi una sfida, in quanto è stato la nostra prima volta in cui ci siamo approcciati alla tecnologia React. Abbiamo perciò preferito concentrarci sull'utilizzo corretto di questa piattaforma, senza utilizzare nessun pattern organizzativo per la struttura del codice, come Redux. Questa scelta è stata dettata dalla volontà di apprendere React e toccare con mano le problematiche che possono incorrere a non utilizzare nessun pattern, in modo da sviluppare un pensiero critico e capire appieno l'efficacia della modularizzazione del codice in ambito web.

9.1 Sviluppi futuri

Come sviluppi futuri, ci siamo prefissati l'obiettivo di apprendere Redux e riprogettare l'applicazione per adattarla a tale pattern in modo da renderla facilmente mantenibile e migliorabile. Inoltre vorremmo introdurre nuove funzionalità, come la possibilità di scattare foto direttamente dal dispositivo e l'introduzione di bluetooth per coordinarsi con le macchine della palestra.

References

[1] OpenJS Foundation. *Express - Framework per applicazioni web Node.js*. URL: https://expressjs.com/it/.

[2] OpenJS Foundation. *Node.js*. URL: https://nodejs.org/it/.

[3] Facebook Inc. *React - Una libreria JavaScript per creare interfacce utente*. URL: https://it.reactjs.org/.

[4] Material-UI. *Material-UI: a popular React UI framework*. URL: https://material-ui.com/.

[5] Inc MongoDB. *MongoDB - Il database per le applicazioni moderne*. URL: https://www.mongodb.com/it.

[6] Socket.IO. *Socket.IO*. URL: https://socket.io/.