PROGETTAZIONE APP REACT

Dashboard per la creazione di un piano di esercizi fisici settimanale

Consegna:

Si richiede di realizzare una web app adibita alla gestione di pazienti/clienti da parte di personal trainer/medici in ambito fitness e benessere personale. In particolare, si richiede la gestione di piani di allenamento personalizzati, in maniera aggiornata nel tempo, in base ai progressi dei clienti o in base ad esigenze particolari.

L'applicativo prevede due attori principali:

- **Personal Trainer**: figura professionale abilitata alla creazione, visualizzazione e modifica dei piani di allenamento personalizzati per ogni singolo cliente. Ha accesso anche alla gestione dei profili clienti e può comunicare direttamente con tutti i suoi clienti
- **Customer:** utente che può visualizzare il proprio piano di allenamento, monitorare gli obiettivi impostati e comunicare direttamente con il proprio trainer attraverso il sistema di messaggistica

L'obiettivo è creare uno strumento intuitivo, veloce ed efficace per migliorare la comunicazione, la personalizzazione e la gestione quotidiana delle attività di allenamento relativamente ai clienti.

Analisi di requisiti:

Autenticazione e Gestione degli Utenti

- Il sistema deve prevedere una **fase di login/autenticazione** per due tipologie di utenti:
    - **Personal trainer**
    - **Customer**
- Dopo l'autenticazione, deve essere possibile distinguere **il ruolo** dell'utente autenticato a seconda di quelli descritti precedentemente.

Gestione Clienti

- Il personal trainer deve poter:
    - **Creare nuovi profili cliente,** inserendo i dati anagrafici (nome, cognome, età, data di nascita, altezza, peso, livello iniziale, eventuali limitazioni).
    - **Visualizzare l'elenco dei clienti** associati al proprio profilo

    - **Selezionare un cliente** per accedere alla sua scheda di allenamento personalizzata ed effettuare eventuali modifiche
    - **Aggiungere e modificare i piani di allenamento** dei clienti
    - **Aggiungere nuove tipologie di esercizio** o modificare quelle esistenti

Piano di allenamento personalizzato

- Ogni cliente deve avere una **scheda esercizi personalizzata**, con la possibilità di suddivisione dell'allenamento in più sedute
- Ogni esercizio è formato dai seguenti campi: nome dell'esercizio, descrizione, difficoltà e target
- Nel momento della creazione del piano di allenamento, per ogni esercizio, **dovrà essere specificato il numero di serie e di ripetizioni**
- Ogni esercizio e piano di allenamento di ogni persona potrà essere modificato o cancellato in ogni momento **esclusivamente dal personal trainer di riferimento**

Altre funzionalità

- possibilità di creazione e aggiunta da parte dell'utente di *goals*, cioè obiettivi prefissati da raggiungere.
- Sistema di messaggistica **personal trainer - utente**.
- Gestione dei messaggi non letti/letti/inviati.
- Possibilità di recuperare la password in caso di smarrimento

Gestione dati

- Tutti i dati dovranno essere salvati e manipolati tramite l'utilizzo di un database, nello specifico, è richiesto l'utilizzo di **Firebase**.

Project Mockup

Il mockup del progetto, realizzato con lo strumento di design Figma, è stato utilizzato come riferimento visivo e funzionale durante tutte le fasi di sviluppo. Questo prototipo ha permesso di definire con chiarezza l'aspetto grafico dell'interfaccia utente, la disposizione degli elementi e i principali flussi di navigazione all'interno dell'applicazione.

Il mockup ha rappresentato una linea guida fondamentale sia per la fase di progettazione che per l'implementazione del front end, facilitando la coerenza tra ciò che era stato ideato graficamente e il prodotto finale sviluppato.

L'immagine del mockup verrà allegata alla **documentazione del progetto**, per fornire un riferimento concreto e visuale delle scelte progettuali effettuate.

Gestione del database

Struttura del database: NoSQL

Firestore è un database NoSQL, il che significa che i dati non sono organizzati in tabelle come in un database relazionale (SQL). Invece, Firestore utilizza una struttura gerarchica basata su:

- Collezioni: le collezioni sono contenitori di documenti.
- Documenti: i documenti sono unità di dati individuali. Un documento contiene coppie campo-valore, dove:
    - *Campo*: è il nome dell'attributo (es: nome , email , data_di_nascita).
    - *Valore*: è il dato effettivo associato al campo. Il valore può essere di diversi tipi: *stringa, numero, booleano, array, oggetto, data/ora, riferimento ad un altro documento*.

Entità del database:

Users (Collection)
└── UserID (Document)
  ├── Email
  ├── UserType (Customer / Personal Trainer)
  ├── Name
  ├── Surname
  ├── DateOfBirth
  ├── Weight
  ├── Height
  ├── GymLevel (Beginner, Intermediate, Advanged)
  ├── Limitation (Optional)
  ├── **Customers (Collection)** (if UserType == Personal Trainer)
  │ └── CustomerID (Document)
  ├── **Goals (Collection)** (if UserType == Customer)
    └── GoalID (Document)
      ├── Name
      ├── Completed
      ├── TargetValue

Exercises (Collection)

└── ExerciseID (Document)
  ├── Name
  ├── Description
  ├── Difficulty (1-5)
  ├── Target (Chest, Arms, Shoulders, Legs, Abs)

Messages (Collection)

└── MessageID (Document)
  ├── Sender
  ├── Recipient
  ├── Body
  ├── Subject
  ├── Read
  ├── Timestamp

TrainingPlan (Collection)

└── PlanID (Document)
  ├── Day [n]
  │ └── **Exercises (Collection)**
  │ ├── [..Campi base..]
  │ ├── Series
  │ ├── Reps
  .....

Tecnologie utilizzate:

- React (^19.0.0)
- Node (v22.14.0)
- Firebase (Authentication & Cloud Firestore)
- Figma (Mockup)
- Visual Studio Code + GitHub (Developing)
- Librerie esterne tra cui React Router, Tailwind, Material UI e altre ancora

Modifiche e implementazioni future:

Sicuramente, è necessario rivedere la struttura delle entità del database, non tanto nel numero di entità, quanto nella gestione della relazione tra le stesse. Un’altra

implementazione futura sarebbe quella aggiungere dei video esplicativi, gestiti dal personal trainer, in maniera da spiegare agli eventuali clienti l'esecuzione degli esercizi. In questo modo sarebbe possibile seguire e gestire i clienti da remoto in maniera più efficiente e fidelizzata. Oltretutto, sarebbe utile introdurre la condivisione di allegati/file nel sistema di messaggistica, non solo di messaggi, inoltre, anche un sistema di notifica quando una delle due parti invia un qualsiasi messaggio.