2025

PROGETTAZIONE DI APP REACT

RELAZIONE DI PROGETTO

ALESSANDRO DE CARLI – PETROV TONI EMILOV

PROGETTAZIONE DI APP IN REACT

OBIETTIVI

Questa applicazione si pone come obiettivo quello di monitorare il numero di passi giornalieri registrati da un utente e mostrarli sotto forma di grafici per ricavarne dei report giornalieri, settimanali e mensili. Oltre a ciò, è in grado di mostrare all'utente tutti gli esercizi fisici eseguiti ed opportunamente registrati in un altro file caricato automaticamente dal componente predisposto.

LA STRUTTURA

1. **Landing.tsx**: componente principale della nostra applicazione. Mostra un form in cui viene chiesto il numero di passi da impostare come obiettivo e la sorgente dei dati.
2. **Dashboard.tsx**: pagina in cui vengono mostrati i dati giornalieri, settimanali e mensili dei passi registrati.
3. **Exercises.tsx**: pagina in cui è possibile generare automaticamente una scheda di allenamento.
4. **Settings.tsx**: pagina in cui è possibile modificare il numero di passi giornalieri impostati come obiettivo.
5. **History.tsx**: pagina in cui è possibile visualizzare lo storico delle attività registrate (camminate, corse, sessioni di ciclismo).
6. **MonthlyStepsChart.tsx**, **WeeklyStepsChart.tsx**, **StepsChart.tsx**: componenti che mostrano dei grafici.
7. **MonthData.tsx**, **WeekData.tsx**: componenti che mostrano dei box con i dati mensili e settimanali registrati.
8. **ActivityCard.tsx**, **Card.tsx**, **ChooseCard.tsx**, **ExerciseCard.tsx**, **Sidebar.tsx**: componenti secondari utilizzati per migliorare la UI.

LIMITI ATTUALI

Con la versione attuale dell'applicazione, ci si può basare solamente su dati statici caricati dall'utente, sia per quanto riguarda il numero di passo registrati che per i dati relativi alle eventuali attività fisiche. Per far fronte a questo problema e migliorare manutenibilità dei dati, sarebbe opportuno creare un database che viene costantemente aggiornato con nuovi dati.

FUTURI SVILUPPI

In futuro, è possibile integrare l'applicazione con il supporto di database; si può migliorare l'estetica dell'applicazione per renderla più moderna ed appetibile per il mercato. Si potrebbe, inoltre, rendere disponibile l'esportazione in un file PDF la scheda di esercizi generata automaticamente per l'utente in modo che possa usarla per fini personali (attualmente non è possibile salvarla).

È possibile anche aggiungere la funzionalità di generazione di ricette personalizzate secondo preferenze dell'utente per aiutarlo con la forma fisica.

TECNOLOGIE UTILIZZATE

In questo progetto sono state utilizzate diverse librerie / tecnologie:

1. Tailwind CSS: per una stilizzazione più veloce
2. Lucid-react: per l'utilizzo di icone
3. Date-fns: per l'utilizzo di funzioni ottimizzate per le date
4. React con TypeScript: per la struttura dell'interfaccia utente

5. Vite: come bundler e ambiente di sviluppo
6. Papaparse: per eseguire il parsing dei file sorgente
7. Reacharts: per la costruzione di grafici
8. React-datepicker: per l'utilizzo di un data-picker all'interno della dashboard
9. Router-dom: per la navigazione
10. Autoprefixer: per la compatibilità tra browser diversi
11. Eslint con globals: aiuta con errori sintattici nel codice con l'utilizzo di alcune variabili d'ambiente globali
12. Axios: serve per eseguire chiamate API. Nel nostro caso, serve per eseguire una GET per reperire il file di storico delle attività.

CHALLENGE E PROBLEMI INCONTRATI NELLO SVILUPPO

Una prima sfida che abbiamo dovuto affrontare è stata quella di analizzare un file caricato dall'utente. Siamo arrivati alla conclusione tramite ricerche in Internet che esiste una libreria, chiamata Papaparse, che aiuta gli sviluppatori in questo problema, rendendo molto più semplice la scrittura di funzioni apposite.

Un' altra challenge riscontrata è stata quella di reperire automaticamente un file da una determinata cartella del progetto. In questa situazione, abbiamo trovato come soluzione l'utilizzo della libreria denominata 'axios' che permette di eseguire chiamate API (nel nostro caso, abbiamo eseguito una GET) per riuscire e reperire il file necessario.

La sfida più grande, invece, è stata quella di trovare una soluzione per quanto riguarda i dati caricati dall'utente. Infatti, volevamo trovare una soluzione che fosse efficace per fare in modo che tutte le pagine della nostra applicazione potevano accedere ai dati senza doverli passare come parametri ai componenti. Per questo, abbiamo utilizzato il 'localStorage' del browser (altrimenti chiamati cookies).