# Università della Calabria

**Dipartimento di Matematica e Informatica**


**Corso di Laurea Magistrale in Artificial Intelligence and Computer Science**

Tesi di Laurea

# SmartFit Coach: sviluppo di una piattaforma AI per il coaching personalizzato in ambito nutrizione e fitness


Relatore:

**Prof. Francesco Ricca**

Candidato:

**Cristian Porco**

**Matricola 252049**

Relatori esterni:

**Prof. Carmelo Macrì**



ANNO ACCADEMICO 2024/2025

*"Non è nelle stelle che è conservato il nostro destino, ma in noi stessi:*

***uomini forti, destini forti; uomini deboli, destini deboli.***

*Non c'è altra strada."*

# Sommario

**Sommario** **2**

**1 Introduzione al mercato del fitness digitale** **5**

1.1 Evoluzione della tecnologia nel settore fitness . . . . . 5

1.2 Personalizzazione tramite l'intelligenza artificiale . . . . . 6

1.3 Finalità e obiettivi di SmartFit Coach . . . . . 7

1.4 Considerazioni personali e motivazioni progettuali . . . . . 8

**2 Introduzione alle tecnologie utilizzate** **9**

2.1 Backend: Django e architettura lato server . . . . . 9

2.2 Frontend: SvelteKit, JavaScript e Material Design . . . . . 11

2.3 LangChain: orchestrazione dell'intelligenza artificiale . . . . . 12

2.4 Modelli LLM: opzioni locali e cloud-based . . . . . 12

**3 Approccio metodologico allo sviluppo del progetto** **14**

3.1 Analisi del dominio e raccolta dei requisiti . . . . . 15

3.2 Verifica e validazione con stakeholder . . . . . 16

3.3 Sviluppo della parte statica: backend e frontend . . . . . 17

3.4 Integrazione dell'IA: LangChain e modelli LLM . . . . . 18

**4 Struttura e organizzazione dei dati** **20**

4.1 Gestione dell'utente e profilo personale . . . . . 21

4.1.1 `DetailsAccount`, informazioni biologiche e profilo digitale dell'utente . . . . . 22

4.1.2 `Weight`, tracciamento del peso . . . . . 23

4.1.3 `BodyMeasurement`, tracciamento delle misure corporee 24
4.2 Alimenti e scheda alimentare . . . 25
4.2.1 `FoodItem`, archivio degli alimenti . . . 25
4.2.2 `FoodPlan`, scheda alimentare giornaliera o su intervallo di date . . . 26
4.2.3 `FoodPlanSection`, fasce orarie dei pasti . . . 27
4.2.4 `FoodPlanItem`, assegnazione degli alimenti alle fasce orarie del piano alimentare . . . 28
4.3 Esercizi e schede di allenamento . . . 29
4.3.1 `GymItem`, archivio degli esercizi . . . 30
4.3.2 `GymPlan`, scheda di allenamento settimanale . . . 31
4.3.3 `GymPlanSection`, suddivisione giornaliera della scheda di allenamento . . . 32
4.3.4 `GymPlanItem`, definizione dell'esercizio all'interno della sezione con tecniche d'intensità applicate . . . 33
4.3.5 `GymPlanSetDetail`, dettagli delle serie . . . 35
4.4 Contenuti multimediali . . . 37
4.4.1 `GymMediaUpload`, gestione immagini esercizi . . . 37
4.4.2 Immagini profilo utente: campo `profile_picture` nel modello `DetailsAccount` . . . 38
4.5 Relazioni e coerenza del database . . . 39
4.6 Considerazioni progettuali . . . 40

**5 Interfaccia utente dell'applicazione 42**
5.1 Struttura generale dell'interfaccia . . . 43
5.2 Area account utente . . . 44
5.2.1 Pagina dettagli dell'account . . . 46
5.2.2 Configurazione dell'obiettivo personale . . . 47
5.2.3 Storico del peso . . . 48
5.2.4 Storico delle misure corporee . . . 50
5.3 Gestione delle schede personalizzate . . . 51

5.3.1 Schede alimentari . . . . . 52
5.3.2 Schede di allenamento . . . . . 56

**6 Integrazione dell'IA e prompt engineering 61**

6.1 Classificazione dell'obiettivo dell'utente . . . . . 61
6.2 Generazione di una spiegazione per la classificazione dell'obiettivo 62
6.3 Analisi dell'andamento del peso rispetto all'obiettivo . . . . . 62
6.4 Analisi dell'andamento delle misure corporee rispetto all'obiettivo 63
6.5 Estrazione strutturata dei pasti da una frase in linguaggio naturale 64
6.6 Selezione semantica dell'alimento più pertinente . . . . . 65
6.7 Ottimizzazione nutrizionale di una scheda alimentare . . . . . 66
6.8 Generazione di una scheda alimentare personalizzata . . . . . 68
6.9 Generazione di un alimento con valori nutrizionali realistici . . . 70
6.10 Calcolo personalizzato dei macronutrienti . . . . . 70
6.11 Generazione di alternative bilanciate per un pasto . . . . . 71
6.12 Classificazione del focus muscolare della giornata di allenamento 73
6.13 Generazione di una nota tecnica per la giornata di allenamento . 74
6.14 Generazione di una nota tecnica per la scheda settimanale di allenamento . . . . . 75
6.15 Generazione di una nota sintetica per singolo esercizio . . . . . 76
6.16 Generazione di una scheda di allenamento settimanale . . . . . 77
6.17 Selezione semantica dell'esercizio più pertinente . . . . . 79
6.18 Generazione di un esercizio alternativo per la scheda di allenamento . . . . . 80
6.19 Generazione automatica delle serie di riscaldamento . . . . . 82
6.20 Suggerimento del carico ideale basato su performance recenti . . 83

**7 Considerazioni finali e sviluppi futuri 85**

7.1 Stato attuale dell'applicativo . . . . . 85
7.2 Potenzialità evolutive e nuove piattaforme: Flutter . . . . . 87
7.3 Estensioni funzionali del sistema . . . . . 87
7.4 Sviluppo di un modello proprietario basato su deep learning . . 89

# Capitolo 1

# Introduzione al mercato del fitness digitale

Negli ultimi anni, il settore del **fitness** ha vissuto una profonda trasformazione grazie all'integrazione di **tecnologie digitali avanzate**. L'adozione di **dispositivi indossabili**, **applicazioni mobili** e **piattaforme online** ha rivoluzionato il modo in cui le persone si allenano, monitorano i **progressi** e interagiscono con i **professionisti del settore**. Questa evoluzione ha reso l'esperienza del fitness più **accessibile**, **personalizzata** e **motivante**, rispondendo alle esigenze di una società sempre più **connessa** e attenta al **benessere personale**.

## 1.1 Evoluzione della tecnologia nel settore fitness

La **tecnologia** ha giocato un ruolo fondamentale nel ridefinire il concetto di **allenamento**. Dispositivi come **smartwatch** e **fitness tracker** permettono di monitorare in tempo reale **parametri vitali** quali **frequenza cardiaca**, **calorie bruciate** e **qualità del sonno**. Le **applicazioni** offrono **programmi di allenamento personalizzati**, adattandosi alle **esigenze individuali** e facilitando il **miglioramento** dell'esercizio fisico. Tuttavia, queste applica-

zioni sono spesso gestite da **professionisti umani**, come **personal trainer** o **nutrizionisti**, che si trovano a dover gestire un **notevole carico di lavoro**, spesso costituito da **compiti ripetitivi** come l'aggiornamento dei piani, la verifica dei progressi o l'invio di feedback standardizzati. In questo contesto, l'**intelligenza artificiale** potrebbe intervenire per automatizzare le operazioni più ripetitive, lasciando al professionista umano il compito di occuparsi degli aspetti più **strategici**, **empatici** e **personalizzati**, migliorando così l'efficienza e la qualità del servizio.

## 1.2 Personalizzazione tramite l'intelligenza artificiale

L'**intelligenza artificiale** (IA) ha introdotto un livello di **personalizzazione** senza precedenti nel mondo del **fitness**. Attraverso l'analisi di **dati biometrici** e **comportamentali**, l'IA è in grado di creare **programmi di allenamento su misura**, ottimizzando l'**efficacia** degli esercizi e aumentando la **motivazione** degli utenti. Questa tecnologia consente di **adattare continuamente** i programmi in base ai **progressi** e alle **esigenze specifiche**, migliorando l'"aderenza" al percorso di allenamento e favorendo risultati più **duraturi**. Tuttavia, è fondamentale sottolineare che l'IA, pur essendo uno strumento potente, non è esente da **limiti** e può commettere **errori**, soprattutto in contesti complessi o non standardizzati. Per questo motivo, è necessario un **utilizzo controllato e supervisionato**, dove l'intelligenza artificiale si occupa principalmente dei **compiti ripetitivi**, standardizzati e a **basso rischio**, mentre il **professionista umano** continua a gestire gli aspetti più **delicati**, **decisionali** e **personalizzati**. Questo approccio ibrido permette di coniugare **efficienza automatica** e **giudizio umano**, garantendo maggiore sicurezza e qualità del servizio.

## 1.3 Finalità e obiettivi di SmartFit Coach

*SmartFit Coach* nasce con l'obiettivo di offrire un'esperienza di **fitness altamente personalizzata**, integrando strumenti di **intelligenza artificiale** applicati sia all'**allenamento** che alla **nutrizione**. La piattaforma consente agli utenti di gestire in autonomia il proprio percorso, grazie a **programmi adattivi** basati sull'IA, **monitoraggio continuo dei progressi** e **supporto intelligente** in tempo reale.

Uno degli aspetti distintivi di *SmartFit Coach* è la capacità di **integrare e correlare dinamicamente** diverse fonti di dati personali: **peso**, **misurazioni corporee**, **schede alimentari** e **schede di allenamento** sono centralizzati in un unico sistema, permettendo un'**analisi sinergica** e un adattamento costante alle esigenze dell'utente.

Questa struttura rende possibile un'evoluzione continua del piano nutrizionale e motorio, in funzione dei cambiamenti fisici e degli obiettivi impostati, garantendo così **coerenza** e **ottimizzazione** tra ciò che si desidera raggiungere e ciò che viene effettivamente intrapreso. Anche chi non possiede competenze specifiche in ambito **fitness** o **nutrizionale** può, tramite l'app, gestire in modo **intuitivo e automatizzato** aspetti complessi come il calcolo dei **fabbisogni calorici**, la **distribuzione dei macronutrienti** o la **strutturazione dell'allenamento**.

I **personal trainer** e i **nutrizionisti** intervengono a **supporto** dell'utente, fornendo **validazione**, **consigli personalizzati** o **aggiustamenti mirati** alle proposte generate dall'**intelligenza artificiale**. In questo modello, il professionista non è più il centro operativo, ma un **alleato** che contribuisce ad aumentare la **qualità** del percorso scelto dall'utente, quando necessario.

In questo modo, *SmartFit Coach* si configura come un **ponte intelligente** tra **autonomia dell'utente** e **competenze professionali**, offrendo un'esperienza realmente **personalizzata** e **accessibile**, con il massimo della **flessibilità** e della **efficienza** per tutti gli attori coinvolti.

## 1.4 Considerazioni personali e motivazioni progettuali

La scelta di sviluppare *SmartFit Coach* è stata guidata dalla mia **passione per il fitness** e dalla profonda convinzione che la **tecnologia** possa essere un **alleato prezioso** nel promuovere uno **stile di vita sano**, accessibile a tutti. L'idea è nata dall'osservazione delle **difficoltà comuni** che molte persone incontrano nel mantenere **costanza** e **motivazione** nel proprio percorso di allenamento e benessere. A partire da questa riflessione, e grazie all'integrazione delle mie **competenze nello sviluppo software** e al crescente interesse per l'**intelligenza artificiale**, ho voluto creare uno **strumento concreto** che potesse supportare gli utenti nel raggiungimento dei loro **obiettivi fisici e nutrizionali**, offrendo un'esperienza altamente **personalizzata**, **coinvolgente** e adattiva.

Questo progetto mi coinvolge anche a livello personale: da tempo seguo uno **stile di vita pulito** sia dal punto di vista **nutrizionale** che di **allenamento**, e comprendo perfettamente quanto possa fare la differenza disporre di strumenti che semplifichino la gestione quotidiana della propria routine, senza perdere di vista la qualità. SmartFit Coach nasce quindi anche dalla volontà di **condividere un metodo**, frutto di esperienza diretta e studio, che possa essere d'aiuto sia a chi muove i primi passi in questo mondo sia a chi cerca un supporto più strutturato e intelligente.

In definitiva, questo progetto rappresenta per me l'opportunità di **coniugare le mie competenze professionali** con la volontà di **contribuire attivamente** alla **salute e al benessere delle persone**, offrendo una soluzione moderna, efficiente e soprattutto umana.

# Capitolo 2

# Introduzione alle tecnologie utilizzate

Lo sviluppo di *SmartFit Coach* ha richiesto l'integrazione di diverse tecnologie moderne, ognuna delle quali ha contribuito in modo specifico a rendere la piattaforma solida, reattiva e intelligente. Le soluzioni adottate coprono l'intero stack tecnologico, dal **backend** al **frontend**, fino ai sistemi di **orchestrazione dell'intelligenza artificiale** e ai **modelli linguistici** utilizzati per generare risposte personalizzate e interattive. In questo capitolo vengono descritte le principali tecnologie scelte, suddivise per ambito di applicazione.

## 2.1 Backend: Django e architettura lato server

Per la gestione del lato server è stato utilizzato **Django**, un framework web open-source basato su **Python**, noto per la sua **robustezza** e **versatilità**. Django è stato impiegato per costruire un'**API RESTful** scalabile e sicura, responsabile della **gestione degli utenti**, **autenticazione**, **persistenza dei dati** e comunicazione con i **moduli AI**. Grazie al suo sistema **ORM** (Object-Relational Mapping), Django semplifica l'interazione con il **database** e garantisce una gestione efficiente dei dati strutturati. Inoltre, la sua struttura **modulare** consente una facile integrazione con strumenti avanzati, tra cui

i servizi per l'**inferenziazione AI**, la **gestione dei media** e la **protezione dei dati** sensibili.
Dal punto di vista organizzativo, il progetto Django è strutturato in modo chiaro e funzionale. Il cuore dell'architettura è la directory `data/`, che rappresenta il nucleo logico dell'intera piattaforma.
Tra i file e le directory principali all'interno del progetto, si evidenziano:

- `management/`: contiene **script personalizzati** per la gestione ordinaria dei dati, in particolare funzioni utili per il controllo, l'aggiornamento e la generazione automatica delle **schede alimentari** e delle **schede di allenamento**;
- `utils.py`: gestisce le chiamate verso i **modelli linguistici** (LLM) e la costruzione dei **prompt dinamici**, fungendo da interfaccia tra i dati dell'utente e i sistemi AI;
- `models.py`: definisce tutte le **entità principali** del progetto, come utenti, alimenti, esercizi, sezioni, schede alimentari, schede di allenamento e altri oggetti fondamentali alla logica applicativa;
- `urls.py`: raccoglie e organizza gli **endpoint dell'API**, mappando le varie funzionalità del backend in modo modulare e accessibile.

Accanto a questi componenti, è presente anche la directory `media/`, dedicata alla **gestione dei contenuti multimediali**. Al suo interno si trovano due sottodirectory significative: `gym_media/`, che contiene le **immagini degli esercizi** mostrate nella piattaforma, e `profile_pics/`, in cui vengono salvate le **immagini di profilo** caricate dagli utenti. Questa organizzazione facilita il recupero efficiente dei file e mantiene separati i contenuti visivi dalle logiche applicative.
La struttura adottata favorisce una **manutenibilità elevata**, semplifica lo sviluppo collaborativo e rende la piattaforma facilmente estendibile nel tempo.

## 2.2 Frontend: SvelteKit, JavaScript e Material Design

L'interfaccia utente è stata sviluppata utilizzando **SvelteKit**, un moderno framework **JavaScript** che permette di creare **applicazioni web reattive** e altamente **performanti**. A differenza di altri framework front-end come React o Vue, Svelte non utilizza un **virtual DOM**, ma compila i componenti al momento della **build**, traducendoli in codice JavaScript ottimizzato. Questo approccio riduce significativamente il **carico computazionale** sul browser durante l'esecuzione, migliorando la **velocità**, la **reattività** e l'esperienza generale dell'utente.

Uno degli obiettivi principali nello sviluppo dell'interfaccia è stato quello di offrire un'esperienza d'uso **fluida** e **intuitiva**, soprattutto da **dispositivi mobili**, dove oggi avviene la maggior parte dell'interazione utente. SvelteKit, grazie al suo sistema di **routing semplificato**, al supporto per **rendering lato server** (SSR) e alla gestione nativa di **transizioni animate**, ha permesso di sviluppare una struttura di navigazione leggera ma solida, capace di adattarsi automaticamente a diverse dimensioni di schermo e tipologie di utente.

Dal punto di vista visivo, è stato adottato il **Material Design 3**, lo standard più recente definito da Google per la progettazione di interfacce digitali. Questo sistema di design garantisce uno **stile visivo coerente** in tutta la piattaforma, mantenendo una combinazione efficace tra **modernità**, **accessibilità** e **leggibilità**. È stato posto particolare accento sulla **chiarezza tipografica**, sulla gerarchia visiva dei contenuti e sull'uso di componenti **interattivi** che guidano l'utente nel proprio percorso.

## 2.3 LangChain: orchestrazione dell'intelligenza artificiale

Per l'orchestrazione dei flussi logici basati su **intelligenza artificiale** è stato scelto **LangChain**, una libreria progettata per costruire applicazioni alimentate da modelli linguistici. LangChain consente la gestione di **prompt dinamici**, il controllo dei **dialoghi contestuali** e l'integrazione con modelli LLM sia locali che remoti. Grazie a questa tecnologia, SmartFit Coach è in grado di personalizzare in modo avanzato le risposte dell'agente AI, adattandosi in tempo reale ai dati dell'utente, come le preferenze alimentari, gli obiettivi fisici o lo storico degli allenamenti. LangChain agisce da motore logico tra frontend, backend e modelli linguistici, permettendo uno **scambio fluido** e **intelligente** di informazioni.

## 2.4 Modelli LLM: opzioni locali e cloud-based

L'inferenza dell'intelligenza artificiale è stata implementata attraverso l'uso di **modelli linguistici di grandi dimensioni** (**LLM**), selezionati in base al bilanciamento tra **prestazioni**, **privacy** e **costi operativi**. In una prima fase progettuale si era valutata la possibilità di utilizzare modelli **open-source** in locale, come **LLaMA** o **DeepSeek**, con l'obiettivo di garantire il massimo controllo sui dati e ridurre la dipendenza da fornitori esterni. Tuttavia, l'esecuzione locale di questi modelli richiede **risorse computazionali elevate**, in particolare GPU ad alte prestazioni e grandi quantità di memoria, che difficilmente risultano disponibili su **macchine personali** o ambienti di sviluppo standard. I tempi di risposta ne avrebbero risentito in modo significativo, compromettendo l'esperienza d'uso, soprattutto in fase di test rapido e iterativo.

Per questa ragione, si è optato per soluzioni **cloud-based** come **ChatGPT** di OpenAI, che garantiscono **prestazioni superiori** in termini di comprensione linguistica, fluidità delle risposte e facilità d'integrazione tramite **API**

**standard**. Questa scelta si è rivelata particolarmente **vantaggiosa in fase di sviluppo**, poiché ha permesso di prototipare e validare rapidamente i flussi di interazione, mantenendo nel contempo **costi contenuti** e un elevato grado di affidabilità nelle risposte generate. I modelli sono stati analizzati in base a diversi criteri, tra cui **tempo di risposta**, **carico computazionale**, possibilità di **fine-tuning** e compatibilità con il sistema di orchestrazione fornito da **LangChain**.

Questa flessibilità architetturale consente di operare in contesti diversi, adattandosi alle esigenze di **utenti finali** e **professionisti**.

# Capitolo 3

# Approccio metodologico allo sviluppo del progetto

Lo sviluppo di *SmartFit Coach* è stato guidato da un **approccio metodologico strutturato e iterativo**, articolato in diverse fasi complementari. Il progetto ha seguito un **ciclo di vita evolutivo**, che ha preso avvio da un'attenta **analisi del dominio**, è proseguito con il coinvolgimento diretto degli **stakeholder** e si è concretizzato nella realizzazione tecnica della piattaforma, culminando con l'**integrazione dell'intelligenza artificiale**. In particolare, si è scelto di procedere inizialmente con lo sviluppo della **parte statica** del sistema, ovvero la costruzione dell'architettura backend e dell'interfaccia frontend, in modo da definire una **struttura solida, scalabile e coerente** su cui potesse successivamente innestarsi l'intelligenza artificiale. Solo dopo aver completato le componenti fondamentali della piattaforma, si è passati all'implementazione delle funzionalità intelligenti, per garantire una **separazione chiara dei livelli di responsabilità** e una maggiore **manutenibilità** del progetto nel tempo.

Questo approccio ha permesso di mantenere un **elevato allineamento tra obiettivi funzionali, usabilità e capacità tecnologiche**, riducendo i **rischi progettuali** e massimizzando l'**aderenza alle esigenze reali** del target di riferimento.

## 3.1 Analisi del dominio e raccolta dei requisiti

La fase iniziale del progetto ha previsto un'approfondita **analisi del dominio** fitness e nutrizione, con l'obiettivo di comprendere a fondo le esigenze tipiche degli utenti target: persone interessate a migliorare la propria salute, il proprio stile di vita o la composizione corporea attraverso un percorso **personalizzato**, ma sempre **supervisionato dagli stakeholder** del progetto. Infatti, anche se il sistema è in grado di generare automaticamente contenuti come le **schede alimentari** e **schede di allenamento**, è previsto che queste vengano **controllate e approvate da un personal trainer** prima che siano effettivamente applicate all'utente. Questo passaggio avviene su richiesta dell'utente finale, che può inoltrare la scheda per validazione, garantendo così l'intervento di un esperto in un punto critico del processo.
L'analisi ha incluso lo studio di **applicazioni esistenti**, **forum tematici** e **interviste esplorative con utenti reali**, con l'obiettivo di individuare **pattern di utilizzo ricorrenti** e **problematiche non ancora risolte**, come la difficoltà nel mantenere la costanza negli allenamenti, la confusione nella gestione dei piani alimentari o la mancanza di feedback personalizzati e motivanti.
A partire da questi elementi, sono stati formalizzati i **requisiti funzionali**, ovvero le funzionalità che la piattaforma deve offrire per rispondere in modo diretto ai bisogni degli utenti. Tra questi figurano:

- la **registrazione dell'aderenza** al percorso, con la possibilità di fornire feedback sui progressi;
- il **tracciamento del peso e delle misure corporee** nel tempo, con confronto automatico rispetto agli obiettivi prefissati;
- la **gestione e generazione guidata della scheda alimentare**, costruita in base al fabbisogno calorico, al profilo nutrizionale e alle preferenze alimentari dell'utente;

- la **gestione e generazione guidata della scheda di allenamento**, personalizzata secondo gli obiettivi fisici, il livello di preparazione e la disponibilità settimanale.

Sono stati inoltre identificati i **requisiti non funzionali**, ossia le proprietà qualitative della piattaforma. Questi includono:

- elevate **prestazioni** nell'elaborazione dei dati e nel caricamento delle interfacce;
- un'elevata **usabilità**, con un'interfaccia chiara, reattiva e accessibile anche a utenti meno esperti;
- la **scalabilità**, per supportare l'espansione a un numero crescente di utenti;
- la **sicurezza e protezione dei dati**, soprattutto per le informazioni sensibili come peso, progressi e alimentazione;
- la **compatibilità cross-device**, per un utilizzo fluido su desktop, tablet e smartphone.

Combinando questi aspetti, sono state delineate le **caratteristiche fondamentali della piattaforma**, tra cui l'**integrazione sinergica tra allenamento, nutrizione e intelligenza artificiale** e un'**esperienza utente personalizzata e adattiva**, capace di evolversi nel tempo sulla base del comportamento, dei dati biometrici e delle preferenze individuali.

## 3.2 Verifica e validazione con stakeholder

Una volta definite le funzionalità principali, si è avviata una fase di **verifica e validazione** tramite confronto diretto con **stakeholder esperti**, in particolare **nutrizionisti** e **personal trainer**. Attraverso **interviste strutturate** e **presentazioni delle prime bozze funzionali**, sono stati raccolti **feedback qualitativi** preziosi che hanno permesso di raffinare l'approccio progettuale.

Gli esperti hanno contribuito a validare la **coerenza delle funzionalità** rispetto agli **scenari d'uso quotidiani**, confermando l'utilità dell'intelligenza artificiale per generare risposte contestuali avanzate e l'efficacia dell'interfaccia grafica statica nello snellire alcune operazioni ripetitive, alleggerendo il carico operativo degli specialisti.
Tuttavia, hanno anche evidenziato un aspetto cruciale: il **mondo del fitness è in costante evoluzione**, e proprio per questo motivo gli strumenti tecnologici, per quanto potenti, devono essere utilizzati con **cautela e spirito critico**. L'IA può rappresentare un **valido supporto**, ma non può sostituire il **giudizio professionale umano**, soprattutto in situazioni complesse o non standardizzabili. Di conseguenza, si è rafforzata l'idea di adottare un **approccio ibrido**, in cui l'intelligenza artificiale fornisce un primo livello di supporto automatizzato, mentre le decisioni finali rimangono in capo allo **stakeholder esperto**. Questo passaggio si è rivelato **cruciale per garantire l'aderenza della piattaforma alla realtà operativa** del settore fitness e benessere, mantenendo equilibrio tra **innovazione tecnologica** e **responsabilità professionale**.

## 3.3 Sviluppo della parte statica: backend e frontend

Con una visione chiara e condivisa, è iniziata la fase di **implementazione delle componenti statiche** della piattaforma, ovvero la costruzione dell'infrastruttura software priva, in questa fase, di logiche intelligenti. Il **backend** è stato sviluppato utilizzando **Django**, un framework solido e maturo basato su **Python**, scelto per la sua capacità di offrire strumenti avanzati per la **gestione sicura dei dati**, la **creazione rapida di API RESTful** e la facilità di integrazione con moduli di **intelligenza artificiale**. Durante questa fase, si è lavorato principalmente alla **definizione dei modelli dati**, elemento fondamentale per rappresentare in modo strutturato e coerente le entità chiave della piattaforma, come utenti, peso e misure corporee. Grande attenzione è stata

dedicata alla costruzione di una **struttura modulare** per le **schede alimentari** e **schede di allenamento**, che permettesse un'organizzazione flessibile dei contenuti e una facile estensione futura. Ogni scheda è suddivisa in sezioni logiche (ad esempio, per giorno della settimana o pasto), e ciascuna sezione può contenere elementi dinamici come esercizi o alimenti, con parametri associati (quantità, tecniche d'intensità, valori nutrizionali, ecc.).
Parallelamente, è stato avviato lo sviluppo del **frontend** utilizzando **SvelteKit**, procedendo di pari passo con la realizzazione della logica di **backend**. Questo approccio ha permesso di modellare l'interfaccia grafica in stretta coerenza con l'evoluzione delle funzionalità lato server, favorendo un'integrazione fluida e una maggiore efficienza nello sviluppo complessivo del sistema.
Questa fase ha rappresentato il **fondamento architetturale** su cui si è successivamente innestata l'intelligenza artificiale, garantendo che le **basi tecniche fossero solide, scalabili e ben strutturate**. Aver completato prima lo scheletro statico della piattaforma ha permesso di testare le funzionalità di base in modo indipendente, riducendo la complessità nella successiva integrazione dell'IA e minimizzando potenziali errori legati alla gestione dei dati, alle interfacce o alle logiche di controllo.

## 3.4 Integrazione dell'IA: LangChain e modelli LLM

Nella fase finale, è stato integrato il **supporto intelligente** tramite **modelli linguistici di grandi dimensioni** (LLM), orchestrati con l'ausilio di **LangChain**. La logica ha previsto la costruzione di **prompt dinamici**, generati a partire dai dati dell'utente, per richiedere risposte personalizzate e contestuali da parte del modello. Questi flussi sono stati impiegati, ad esempio, per la **generazione automatica di schede alimentari e di allenamento**, la creazione di **strumenti di supporto personalizzati** a tali piani (come suggerimenti su porzioni, alternative alimentari o varianti di esercizi), e la produzione di **feedback intelligenti sull'andamento del peso e delle misure**

**corporee**, adattando dinamicamente le proposte in base all'evoluzione dell'utente e agli obiettivi prefissati. L'integrazione è avvenuta inizialmente tramite **modelli cloud-based** come ChatGPT, scelti per la rapidità di implementazione e la qualità delle risposte. Questa fase ha segnato la transizione da una piattaforma statica a un sistema realmente **intelligente e adattivo**, capace di apprendere dal contesto e offrire un supporto mirato e personalizzato.

# Capitolo 4

# Struttura e organizzazione dei dati

La progettazione della struttura dati rappresenta una delle fasi più critiche nello sviluppo di una piattaforma orientata alla personalizzazione come *SmartFit Coach*. L'accurata modellazione delle entità principali consente non solo di garantire l'integrità e la coerenza dei dati, ma anche di rendere l'intero sistema flessibile, scalabile e facilmente estendibile nel tempo. In una piattaforma che integra aspetti legati al **benessere fisico**, all'**alimentazione** e alla **programmazione dell'allenamento**, risulta fondamentale tradurre concetti concreti e spesso complessi in strutture dati chiare, modulari e ben separate.
L'approccio adottato per *SmartFit Coach* è stato quello di suddividere logicamente il dominio applicativo in quattro macro-aree fondamentali:

- **gestione dell'utente**: include tutti i dati personali e biometrici, come altezza, peso, misure corporee e obiettivi;
- **nutrizione**: comprende l'archivio alimentare, la costruzione dinamica di schede alimentari e la distribuzione dei pasti nel corso della giornata;
- **allenamento**: riguarda la creazione delle schede settimanali, l'organizzazione degli esercizi giornalieri e il dettaglio delle serie e delle tecniche d'intensità;

- **contenuti multimediali**: gestione delle immagini caricate dagli utenti, sia per quanto riguarda i profili personali che il supporto visivo agli esercizi.

La separazione logica in questi domini ha permesso di mantenere una chiara suddivisione delle responsabilità all'interno del database, facilitando l'evoluzione indipendente di ciascun modulo senza introdurre ridondanze o ambiguità. Ogni modello è stato progettato con l'obiettivo di essere quanto più possibile **autosufficiente**, ma al tempo stesso integrabile con le altre entità tramite **relazioni esplicite**, come chiavi esterne o strutture intermedianti.
Nel corso di questo capitolo, verranno analizzati nel dettaglio i modelli definiti per ciascuna area, evidenziando le scelte architetturali, le relazioni tra entità, le validazioni applicate e le potenzialità di estensione futura.

## 4.1 Gestione dell'utente e profilo personale

In questa sezione vengono analizzati i modelli relativi alla gestione dell'utente e al suo profilo personale all'interno della piattaforma. Poiché *SmartFit Coach* si basa su un approccio fortemente personalizzato, è fondamentale raccogliere e strutturare in modo accurato tutte le **informazioni individuali** che caratterizzano ciascun utente. Questo include sia dati statici — come età, genere, altezza e immagine del profilo — sia dati dinamici, soggetti a variazioni nel tempo, come il peso e le misure corporee.
Oltre alla registrazione di queste informazioni, è previsto anche un sistema per descrivere gli **obiettivi personali** dell'utente, accompagnato da un meccanismo intelligente in grado di inferire e spiegare la categoria di allenamento più adatta. L'obiettivo di questa modellazione è duplice: da un lato garantire una base solida per la personalizzazione dei piani, dall'altro permettere all'intelligenza artificiale di disporre di dati significativi per elaborare suggerimenti contestuali e pertinenti.

I modelli descritti in questa sezione costituiscono il punto di partenza per l'intera esperienza utente e sono strettamente collegati a tutti gli altri moduli della piattaforma, in particolare quelli dedicati alla nutrizione e all'allenamento.

### 4.1.1 `DetailsAccount`, informazioni biologiche e profilo digitale dell'utente

Il modello `DetailsAccount` estende il profilo dell'utente registrato, fornendo una struttura informativa avanzata che combina dati anagrafici, biometrici e obiettivi personali. Questo modello rappresenta un nodo centrale all'interno della piattaforma, poiché fornisce informazioni essenziali per la personalizzazione di schede alimentari e di allenamento. Ogni campo è pensato per raccogliere un aspetto specifico della condizione iniziale e degli obiettivi dell'utente, integrando anche componenti generate automaticamente dalla IA.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna al modello utente, identifica il proprietario del profilo esteso;
- `date_of_birth`: data di nascita dell'utente, utile per calcolare età e valutazioni basate sull'anagrafica;
- `biological_gender`: sesso biologico dell'utente, con scelta obbligata tra `M` (maschio) e `F` (femmina);
- `height_cm`: altezza espressa in centimetri, validata tramite range accettabile (1–300);
- `profile_picture`: immagine del profilo, caricabile dall'utente, con percorso di upload in `profile_pics/` e valore predefinito;
- `goal_description`: breve descrizione dell'obiettivo dell'utente (max 160 caratteri), campo testuale utile per inferenze AI;

- `goal_targets`: risultato dell'inferenza AI, rappresenta la categoria obiettivo scelta tra **fitness**, **bodybuilding**, **powerlifting**, **streetlifting**;
- `goal_targets_explanation`: spiegazione generata automaticamente dal sistema, motiva la scelta dell'obiettivo selezionato.

Nel complesso, `DetailsAccount` fornisce una base coerente e strutturata per costruire l'intero percorso personalizzato dell'utente, sfruttando sia input manuali sia elaborazioni automatiche basate sull'IA.

### 4.1.2 `Weight`, tracciamento del peso

Il modello `Weight` si occupa della registrazione e del monitoraggio del **peso corporeo** dell'utente nel tempo. Trattandosi di un valore soggetto a variazioni frequenti, questo modello consente di costruire una **cronologia ordinata** delle misurazioni, utile per l'analisi dell'andamento fisico e per l'adattamento delle schede alimentari e di allenamento. Ogni rilevazione è legata all'utente che l'ha registrata e associata a una specifica data.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna al modello utente, che identifica l'utente proprietario della rilevazione;
- `date_recorded`: data della misurazione del peso, permette di storicizzare e ordinare i dati nel tempo;
- `weight_value`: valore del peso corporeo espresso in chilogrammi, validato per essere maggiore o uguale a zero.

Questo modello, semplice ma essenziale, consente alla piattaforma di fornire un feedback personalizzato in base alla progressione fisica dell'utente.

### 4.1.3 `BodyMeasurement`, tracciamento delle misure corporee

Il modello `BodyMeasurement` consente la registrazione delle principali **misure antropometriche** dell'utente, espresse in centimetri. Le misurazioni rappresentano un indicatore importante dell'evoluzione fisica e sono utilizzate per valutare l'efficacia delle schede alimentari e di allenamento. L'utente può inserire una o più misure in modo facoltativo, e il sistema è in grado di calcolarne la media attraverso un metodo interno.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna al modello utente, che identifica l'utente proprietario della rilevazione;
- `chest`: circonferenza del torace;
- `bicep`: circonferenza del bicipite;
- `thigh`: circonferenza della coscia;
- `waist`: circonferenza della vita;
- `hips`: circonferenza dei fianchi;
- `abdomen`: circonferenza addominale;
- `calf`: circonferenza del polpaccio;
- `neck`: circonferenza del collo;
- `shoulders`: larghezza delle spalle;
- `date_recorded`: data della misurazione, impostata automaticamente alla data corrente se non specificata.

In aggiunta ai campi, il modello include un metodo `average_measurement()` che calcola la **media aritmetica tra le misure disponibili.** Questo metodo risulta utile per sintetizzare graficamente l'evoluzione corporea dell'uten-

te in una singola metrica e per generare feedback automatici sull'andamento complessivo delle dimensioni corporee.

## 4.2 Alimenti e scheda alimentare

In questa sezione vengono descritti i modelli utilizzati per gestire gli **alimenti**, la composizione delle **schede alimentari** e la loro suddivisione all'interno della giornata in **fasce orarie dedicate ai pasti**. L'organizzazione dei dati relativi alla nutrizione è progettata in modo modulare, così da garantire flessibilità e possibilità di personalizzazione in base alle esigenze individuali dell'utente.
Ogni alimento è rappresentato da un modello che ne descrive i valori nutrizionali per 100 grammi, con informazioni facoltative come il brand e il codice a barre. Le schede alimentari, invece, sono strutturati come **settimane alimentari** definite da una data di inizio e una di fine, all'interno delle quali gli alimenti vengono distribuiti in **sezioni orarie personalizzabili**.
La logica relazionale prevede l'utilizzo di un modello intermedio che consente di associare un alimento a una determinata scheda, in una specifica sezione della giornata, indicando anche la quantità consumata. Questo approccio consente non solo il tracciamento preciso dell'apporto alimentare giornaliero, ma anche l'adattabilità del sistema ai diversi stili di vita e preferenze alimentari.

### 4.2.1 `FoodItem`, archivio degli alimenti

Il modello `FoodItem` rappresenta un **singolo alimento** all'interno della piattaforma, associato a un utente autore. È progettato per contenere tutte le informazioni nutrizionali necessarie a valutare l'impatto di un alimento sulla scheda alimentare complessiva. I valori sono espressi in riferimento a 100 grammi di prodotto, secondo una convenzione standard nel settore alimentare. Il modello include anche attributi opzionali utili per la tracciabilità e la classificazione del prodotto.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna che identifica l'utente autore dell'alimento (utile per alimenti personalizzati);
- `name`: nome dell'alimento, campo obbligatorio e identificativo;
- `barcode`: codice a barre del prodotto, opzionale, utile per l'integrazione con scanner o database esterni;
- `brand`: marca dell'alimento, facoltativa ma utile per specificare il produttore dell'alimento;
- `kcal_per_100g`: valore energetico espresso in chilocalorie per 100 grammi;
- `protein_per_100g`: contenuto proteico per 100 grammi;
- `carbs_per_100g`: quantità di carboidrati totali per 100 grammi;
- `sugars_per_100g`: quantità di zuccheri semplici, facoltativa;
- `fats_per_100g`: quantità di grassi per 100 grammi;
- `saturated_fats_per_100g`: quantità di grassi saturi, campo opzionale;
- `fiber_per_100g`: quantità di fibra alimentare per 100 grammi.

La struttura del modello permette un'analisi precisa della composizione nutrizionale degli alimenti inseriti nelle schede alimentari. La possibilità di includere anche brand e codice a barre ne facilita l'integrazione con sistemi di scansione e catalogazione personalizzata.

### 4.2.2 `FoodPlan`, scheda alimentare giornaliera o su intervallo di date

Il modello `FoodPlan` rappresenta la **scheda alimentare principale** associato a un determinato utente e valido per un intervallo temporale specifico,

che varia da una giornata ad un periodo più lungo. Ogni scheda alimentare raccoglie una serie di alimenti organizzati in sezioni giornaliere tramite un'associazione intermedia e definisce dei **limiti nutrizionali complessivi** per garantire un'alimentazione bilanciata rispetto agli obiettivi personali.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna che collega la scheda alimentare all'utente autore;
- `food_items`: relazione molti-a-molti con il modello `FoodItem`, gestita tramite il modello intermedio `FoodPlanItem`, che consente di associare quantità, sezioni orarie e stato di consumo;
- `start_date`: data di inizio della scheda alimentare;
- `end_date`: data di fine della scheda alimentare;
- `max_kcal`: valore massimo di apporto calorico giornaliero previsto dalla scheda;
- `max_protein`: soglia massima giornaliera per le proteine;
- `max_carbs`: soglia massima giornaliera per i carboidrati;
- `max_fats`: soglia massima giornaliera per i grassi.

Questo modello funge da contenitore strutturato per tutti gli alimenti e le sezioni giornaliere previste nella scheda. I valori limite definiti per i macronutrienti permettono alla piattaforma di calcolare il rispetto della scheda in tempo reale e offrire suggerimenti intelligenti in caso di superamento o squilibrio.

### 4.2.3 `FoodPlanSection`, fasce orarie dei pasti

Il modello `FoodPlanSection` definisce le **fasce orarie** in cui sono distribuiti i pasti all'interno di una scheda alimentare. Ogni sezione può rappresentare una categoria come *colazione*, *spuntino*, *pranzo*, *cena*, ecc., e consente di organizzare gli alimenti in modo strutturato durante l'arco della giornata. La

presenza di questa suddivisione facilita il monitoraggio dei consumi e migliora la leggibilità e la gestione dell'interfaccia lato utente.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna che collega la sezione all'utente autore;
- `name`: nome della fascia oraria personalizzabile, come ad esempio "Colazione", "Spuntino pomeridiano" o "Cena";
- `start_time`: orario indicativo d'inizio della sezione, espresso come intero (es. 8 per le 08:00).

Il modello è progettato per essere riutilizzabile e flessibile, permettendo all'utente di personalizzare l'organizzazione dei propri pasti giornalieri secondo il proprio stile di vita. In combinazione con `FoodPlanItem`, consente una gestione dettagliata degli alimenti all'interno della scheda alimentare.

### 4.2.4 `FoodPlanItem`, assegnazione degli alimenti alle fasce orarie del piano alimentare

Il modello `FoodPlanItem` funge da **entità intermedia che collega un alimento specifico ad una determinata scheda alimentare**, all'interno di una precisa sezione oraria (es. colazione, pranzo). È progettato per rappresentare in maniera dettagliata l'assunzione di un alimento, includendo la **quantità consumata** e lo **stato di completamento** (se il pasto è stato consumato o meno). Questo livello di dattaglio permette un tracciamento preciso dell'aderenza alla scheda e abilita funzionalità come il calcolo dei macronutrienti assunti per fascia oraria.
Il modello è composto dai seguenti attributi:

- `eaten`: campo booleano che indica se l'alimento è stato consumato (`True`) o meno (`False`);
- `food_plan`: chiave esterna che collega l'alimento alla scheda alimentare (`FoodPlan`) di riferimento;

- `food_item`: chiave esterna che identifica l'alimento (`FoodItem`) consumato;
- `food_section`: chiave esterna alla sezione (`FoodPlanSection`) in cui è stato consumato l'alimento;
- `quantity_in_grams`: quantità dell'alimento consumato, espressa in grammi.

Grazie a questa struttura, `FoodPlanItem` permette di tenere traccia in modo dettagliato delle abitudini alimentari quotidiane dell'utente, facilitando la generazione automatica di schede alimentari e l'interazione con moduli intelligenti di suggerimento e ottimizzazione della scheda.

## 4.3 Esercizi e schede di allenamento

In questa sezione vengono descritti i modelli responsabili della **creazione**, **organizzazione** e **gestione delle schede di allenamento**, con particolare attenzione alla struttura settimanale, alla suddivisione per giorno e all'applicazione di **tecniche d'intensità**. La modellazione di questa parte del dominio ha richiesto un approccio flessibile ma ben strutturato, in grado di rappresentare con precisione la varietà e la complessità degli allenamenti moderni, e ha comportato un **notevole investimento di tempo** per garantire la coerenza tra struttura dati, logica applicativa e possibilità di estensione futura.
La logica si articola attorno a una scheda settimanale che copre un intervallo fisso da lunedì a domenica. Ciascuna giornata è rappresentata da una sezione specifica, alla quale vengono assegnati esercizi scelti da un catalogo strutturato e classificato secondo numerosi criteri (difficoltà, attrezzatura, muscoli coinvolti, meccanica del movimento, ecc.).
Ogni esercizio può essere ulteriormente dettagliato in termini di serie, ripetizioni, pause, peso utilizzato e anche tramite l'uso di **tecniche avanzate d'intensità** come rest-pause, drop set, tempo training, isometrie o cluster set.

Il sistema supporta inoltre la definizione di note descrittive e l'ordinamento dei set per facilitare la consultazione e la progressione logica.
Questa struttura dati consente una rappresentazione fedele e altamente personalizzabile del scheda di allenamento, mantenendo al contempo un'elevata coerenza con le esigenze pratiche dei professionisti del settore e dei sistemi intelligenti incaricati della generazione automatica.

### 4.3.1 `GymItem`, archivio degli esercizi

Il modello `GymItem` rappresenta un **singolo esercizio fisico** all'interno del sistema, costituendo l'elemento base per la costruzione delle schede di allenamento. Ogni esercizio è descritto da una serie di attributi che ne definiscono le caratteristiche biomeccaniche, la difficoltà, le attrezzature richieste e i gruppi muscolari coinvolti. L'obiettivo di questo modello è offrire un **catalogo ampio e strutturato** di esercizi da cui l'utente o l'intelligenza artificiale possono attingere per generare schede efficaci e coerenti con il profilo fisico dell'individuo.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna all'utente autore;
- `name`: nome dell'esercizio definito in lingua inglese (es. Panca Piana, Squat, Trazioni);
- `force`: tipo di forza prevalente (*push*, *pull*, *static*), campo opzionale;
- `level`: livello di difficoltà dell'esercizio (*beginner*, *intermediate*, *expert*);
- `mechanic`: meccanica del movimento, ovvero se l'esercizio è *multiarticolare* o di *isolamento*;
- `category`: categoria dell'esercizio (es. cardio, strength, powerlifting, stretching, ecc.);
- `equipment`: attrezzatura richiesta (es. corpo libero, bilanciere, cavi, kettlebell, manubri, ecc.);

- `primary_muscle`: muscolo principale coinvolto nell'esercizio (es. pettorali, dorsali, quadricipiti);
- `secondary_muscles`: lista JSON di muscoli secondari attivati;
- `instructions`: lista JSON contenente istruzioni dettagliate per l'esecuzione corretta dell'esercizio;
- `image_urls`: relazione many-to-many con il modello `GymMediaUpload`, per associare una o più immagini illustrative all'esercizio.

Questo modello consente una classificazione approfondita degli esercizi e può essere utilizzato sia in fase di creazione manuale delle schede, sia in processi automatizzati di generazione intelligente delle schede di allenamento.

### 4.3.2 `GymPlan`, scheda di allenamento settimanale

Il modello `GymPlan` rappresenta la **scheda di allenamento settimanale** assegnata a un utente. Ogni scheda copre esattamente una settimana, da **lunedì a domenica**, e funge da contenitore per le sezioni giornaliere e gli esercizi assegnati. Il vincolo sulla durata di 7 giorni garantisce uniformità nella strutturazione del calendario e coerenza nell'elaborazione dei dati da parte dell'intelligenza artificiale. Il modello è progettato per essere facilmente estendibile con note e metadati.
Il modello è composto dai seguenti attributi:

- `author`: chiave esterna all'utente autore della scheda di allenamento;
- `start_date`: data di inizio della scheda, obbligatoriamente un lunedì (validata nel metodo `clean`);
- `end_date`: data di fine della scheda, obbligatoriamente una domenica (validata nel metodo `clean`);
- `note`: campo testuale libero per aggiungere commenti, indicazioni o annotazioni generali da parte del trainer o dell'utente stesso.

Il metodo `clean`, citato negli attributi `start_date` e `end_date`, integrato nel modello, applica una validazione personalizzata per garantire che la scheda rispetti il formato settimanale fisso. In particolare, verifica che:

- la data di inizio cada di lunedì;
- la data di fine cada di domenica;
- la durata totale sia esattamente di 7 giorni.

Questa struttura garantisce **consistenza e prevedibilità** nella generazione e gestione delle schede di allenamento, semplificando la suddivisione per giorni e l'assegnazione delle attività quotidiane.

### 4.3.3 `GymPlanSection`, suddivisione giornaliera della scheda di allenamento

Il modello `GymPlanSection` rappresenta una **giornata specifica all'interno di una scheda di allenamento settimanale** (`GymPlan`). Ogni sezione è collegata a uno dei sette giorni della settimana e consente di specificare il tipo di allenamento previsto per quella giornata (es. "Push", "Full Body", "Recupero attivo") insieme a eventuali annotazioni. Questa struttura permette una **suddivisione logica e ordinata** delle attività settimanali, agevolando sia la visualizzazione che la generazione automatizzata della scheda.
Il modello è composto dai seguenti attributi:

- `gym_plan`: chiave esterna che collega la sezione alla scheda di allenamento settimanale (`GymPlan`) di riferimento;
- `day`: giorno della settimana, rappresentato tramite una scelta fissa tra i sette giorni (da `"lun"` a `"dom"`);
- `type`: descrizione del tipo di sessione o focus del giorno (es. "Push", "Gambe", "HIIT", "Riposo attivo"), campo libero e opzionale;
- `note`: campo testuale facoltativo per annotazioni aggiuntive sul giorno (es. "Allenamento leggero a causa di affaticamento").

In questo modo, l'allenamento settimanale può essere organizzato in maniera **flessibile e strutturata**, con la possibilità di assegnare a ciascun giorno un focus specifico. La suddivisione in sezioni giornaliere consente inoltre una gestione modulare e rende più semplice l'interazione con le tecniche d'intensità e gli esercizi assegnati per ogni sessione.

### 4.3.4 `GymPlanItem`, definizione dell'esercizio all'interno della sezione con tecniche d'intensità applicate

Il modello `GymPlanItem` rappresenta un **esercizio assegnato a una specifica giornata di allenamento** all'interno di una `GymPlanSection`. Oltre a indicare la sequenza dell'esercizio nella giornata, consente di aggiungere **note personalizzate** e specificare eventuali **tecniche d'intensità**, che ne modificano o aumentano la difficoltà. Il campo dedicato alle tecniche permette di selezionarne anche più di una, offrendo un alto livello di personalizzazione.
Il modello è composto dai seguenti attributi:

- `section`: chiave esterna alla sezione giornaliera (`GymPlanSection`) a cui appartiene l'esercizio;
- `order`: posizione dell'esercizio nel giorno, utile per l'ordinamento;
- `notes`: campo testuale opzionale per aggiungere osservazioni specifiche;
- `intensity_techniques`: campo JSON che memorizza un elenco di tecniche d'intensità applicate.

Le tecniche disponibili, descritte tramite la classe `TechniqueType`, includono le seguenti opzioni:

- **Bilaterale (entrambe gli arti)**: esecuzione dell'esercizio con entrambi gli arti simultaneamente;
- **Unilaterale (singolo arto)**: lavoro concentrato su un solo lato del corpo alla volta;

- **Tempo-Based (durata fissa)**: controllo rigoroso del tempo in ciascuna fase del movimento;
- **Drop Set / Stripping**: riduzione progressiva del carico per proseguire oltre il cedimento;
- **Super Set / Giant Set**: combinazione di due o più esercizi eseguiti senza pause;
- **Serie Forzate**: ripetizioni eseguite oltre il limite con assistenza esterna;
- **Half Reps**: ripetizioni parziali eseguite su metà del range di movimento;
- **Rest-Pause**: serie suddivisa in blocchi separati da pause molto brevi;
- **MyoReps**: serie attivante seguita da micro-serie a breve distanza temporale;
- **Pre-Affaticamento**: attivazione del muscolo target con esercizio isolato prima del multiarticolare;
- **Negativa Forzata**: enfasi sulla fase eccentrica con carico controllato o sovraccaricato;
- **Contrazione di Picco**: pausa nel punto di massima contrazione muscolare;
- **Tempo Training / TUT**: gestione precisa del tempo sotto tensione;
- **Contrazioni Isometriche**: mantenimento statico della posizione per stimolare il muscolo;
- **21 Serie / 7-7-7**: metodo 21: 7 ripetizioni parziali basse, 7 alte e 7 complete;
- **Cluster Set**: serie frammentate in blocchi con pause brevi per migliorare la potenza;

- **Piramidale**: progressione graduale del carico e delle ripetizioni in salita o discesa;
- **Wave Loading**: carico ciclico a onde per stimolare forza e adattamento;
- **Isometric Overload**: contrazione isometrica con carichi elevati o durate estese;
- **Accommodating Resistance**: uso di elastici o catene per modulare la resistenza durante il movimento;
- **Pause Reps**: inserimento volontario di pause durante la fase eccentrica o isometrica;
- **EMOM**: "Every Minute on the Minute": eseguire un esercizio all'inizio di ogni minuto;
- **AMRAP**: "As Many Reps As Possible": eseguire il maggior numero di ripetizioni possibili in un tempo definito;
- **Death Set**: serie estrema eseguita fino all'esaurimento totale, spesso senza schema fisso.

L'integrazione di queste tecniche consente alla piattaforma di gestire sia schede di allenamento tradizionali sia protocolli altamente personalizzati e avanzati, offrendo all'utente finale e al professionista una gamma completa di strumenti per costruire sessioni efficaci e stimolanti.

### 4.3.5 `GymPlanSetDetail`, dettagli delle serie

Il modello `GymPlanSetDetail` rappresenta il **dettaglio di una singola serie all'interno di un esercizio programmato nella giornata di allenamento**. È collegato a un `GymPlanItem` e contiene tutti i parametri fondamentali per definire con precisione l'esecuzione della serie: ripetizioni, carico, riposo, tempo di esecuzione e ordine. Questa struttura consente alla piattaforma di offrire un supporto completo alla pianificazione e al tracciamento degli allenamenti, anche nei casi più complessi.

Il modello è composto dai seguenti attributi:

- `plan_item`: chiave esterna che collega la serie all'esercizio di riferimento (`GymPlanItem`);
- `exercise`: chiave esterna all'esercizio eseguito in quella serie (`GymItem`);
- `order`: ordine della serie all'interno della giornata;
- `set_number`: numero identificativo della serie all'interno dell'esercizio;
- `prescribed_reps_1` / `actual_reps_1`: ripetizioni prescritte ed effettivamente eseguite per l'**arto 1**;
- `prescribed_reps_2` / `actual_reps_2`: ripetizioni prescritte ed effettive per l'**arto 2**, utile per esercizi unilaterali;
- `rir`: "Reps in Reserve", ovvero il numero di ripetizioni che l'atleta ritiene di poter ancora eseguire al termine della serie (indicatore di intensità soggettiva);
- `rest_seconds`: tempo di recupero in secondi tra le serie;
- `weight`: carico utilizzato nella serie, espresso in chilogrammi;
- `tempo_fcr`: tempo sotto tensione suddiviso in tre fasi: eccentrica, pausa, concentrica (es. `"3-1-2"`).

Il metodo `delete()` include una **logica automatica per garantire la pulizia del database**: se, dopo l'eliminazione della serie, il relativo `GymPlanItem` non ha più alcuna serie associata, anche quest'ultimo viene automaticamente eliminato. Questa accortezza mantiene **l'integrità della scheda di allenamento** ed evita la creazione di esercizi vuoti.
Questo modello è essenziale per descrivere con precisione le variabili tecniche dell'allenamento, ed è particolarmente utile per protocolli dettagliati o esercizi che richiedono la distinzione tra i due arti.

## 4.4 Contenuti multimediali

In questa sezione vengono descritti i modelli e i campi relativi alla **gestione dei contenuti multimediali**, fondamentali per migliorare l'esperienza visiva e la fruibilità della piattaforma. I file multimediali, in particolare le **immagini**, svolgono un ruolo chiave sia nel supportare la corretta esecuzione degli esercizi, sia nel rafforzare la personalizzazione attraverso l'inserimento di **immagini di profilo** da parte dell'utente.
Il sistema prevede due componenti principali:

- la gestione delle immagini associate agli esercizi tramite il modello `GymMediaUpload`, che consente il caricamento e la memorizzazione strutturata dei file;
- il campo `profile_picture`, incluso nel modello `DetailsAccount`, che consente agli utenti di caricare un'immagine rappresentativa del proprio profilo personale.

Tutti i contenuti multimediali sono archiviati in una struttura dedicata, organizzata in cartelle come `gym_media/` per gli esercizi e `profile_pics/` per le immagini utente. Questa suddivisione semplifica la gestione, la sicurezza e l'accesso ai file, garantendo coerenza tra struttura dei dati e componenti visivi dell'interfaccia.

### 4.4.1 `GymMediaUpload`, gestione immagini esercizi

Il modello `GymMediaUpload` è responsabile della **gestione dei file multimediali associati agli esercizi**, in particolare immagini illustrative. Questi contenuti visivi aiutano l'utente a comprendere l'esecuzione corretta del movimento e migliorano l'interazione con l'interfaccia. Il modello è progettato per essere semplice e modulare, in modo da poter essere riutilizzato per associare più immagini a uno stesso esercizio o condividere un file tra esercizi diversi.
Il modello è composto dai seguenti attributi:

- `file`: campo per il caricamento del file, memorizzato nella directory `gym_media/`;
- `uploaded_at`: data e ora in cui il file è stato caricato, impostata automaticamente al momento dell'upload.

Ogni file può essere collegato a uno o più esercizi tramite una relazione *molti-a-molti* nel modello `GymItem`. Questa struttura permette di mantenere la flessibilità necessaria per associare diverse viste, angolazioni o versioni di uno stesso esercizio, garantendo una documentazione visiva completa.
Il modello `GymMediaUpload` costituisce una componente essenziale per rendere la piattaforma non solo informativa, ma anche **intuitiva e accessibile**, soprattutto per utenti con poca esperienza pratica nel mondo dell'allenamento.

### 4.4.2 Immagini profilo utente: campo `profile_picture` nel modello `DetailsAccount`

All'interno del modello `DetailsAccount`, il campo `profile_picture` permette all'utente di caricare un'immagine personale che verrà visualizzata come **foto profilo** all'interno della piattaforma. Questa funzionalità, pur semplice, contribuisce a rendere l'esperienza utente più **personale e coinvolgente**, permettendo di associare un volto o un'identità visiva al proprio percorso fitness.
Il modello è composto anche dal seguente attributi:

- `profile_picture`: campo immagine opzionale, memorizzato nella cartella `profile_pics/`, con immagine predefinita in assenza di upload da parte dell'utente.

L'utilizzo di un percorso di upload dedicato consente di organizzare le immagini di profilo separatamente da altri media, migliorando la **manutenibilità del sistema** e semplificando eventuali funzionalità future come la sostituzione o l'eliminazione delle immagini. Inoltre, il supporto per valori nulli o

vuoti garantisce che la funzionalità non sia vincolante, lasciando piena libertà all'utente.

## 4.5 Relazioni e coerenza del database

In questa sezione vengono analizzate le **relazioni tra i modelli** della piattaforma e le scelte effettuate per garantire la **coerenza logica e referenziale** del database. In un sistema articolato come *SmartFit Coach*, dove convivono entità diverse ma fortemente interconnesse (utenti, alimenti, esercizi, schede, file multimediali), è fondamentale progettare con attenzione la struttura relazionale per evitare ridondanze, ambiguità o errori nei dati.
L'implementazione delle relazioni si basa principalmente su:

- **ForeignKey**: per collegare entità in rapporto uno-a-molti (es. un utente può avere più pesate o schede di allenamento);
- **ManyToManyField con modello intermedio**: per rappresentare relazioni complesse e ricche di attributi, come quella tra alimenti e schede alimentari (`FoodPlanItem`);
- **OneToMany impliciti**: attraverso la definizione di modelli figlio, come le sezioni e le serie collegate alle schede di allenamento.

Inoltre, sono stati implementati meccanismi specifici per:

- **ordinare dinamicamente** elementi all'interno di una lista (es. esercizi o serie tramite il campo `order`);
- applicare **vincoli personalizzati** (es. durata della settimana nel modello `GymPlan`);
- gestire in modo sicuro le **eliminazioni in cascata** o condizionali, per mantenere l'integrità tra entità correlate (es. eliminazione automatica di un `GymPlanItem` se tutte le sue serie vengono rimosse);

- supportare **scenari multiutente**, in cui ogni dato è legato all'utente che lo ha generato attraverso il campo `author`.

Questa progettazione relazionale consente alla piattaforma di essere non solo solida dal punto di vista strutturale, ma anche facilmente estendibile in scenari futuri, come l'integrazione di nuovi moduli, il supporto alla condivisione dei dati con altri utenti o l'ottimizzazione personalizzata tramite intelligenza artificiale.

## 4.6 Considerazioni progettuali

La progettazione dell'architettura dati per *SmartFit Coach* è stata guidata da una serie di **principi chiave** volti a garantire chiarezza strutturale, modularità, e soprattutto **scalabilità** nel tempo. Fin dalle prime fasi dello sviluppo, si è scelto di adottare un approccio che favorisse la separazione logica tra i diversi ambiti della piattaforma, definendo per ciascuno un insieme di modelli coerenti e ben delimitati.
Una particolare attenzione è stata rivolta alla:

- **modularità**: ogni gruppo di modelli è autonomo ma integrabile con gli altri, il che consente modifiche o estensioni localizzate senza compromettere la stabilità dell'intero sistema;

- **riutilizzabilità**: molte entità, come `GymItem` o `FoodItem`, possono essere associate a più piani o sezioni, riducendo la duplicazione e facilitando l'organizzazione;

- **manutenibilità**: l'intero schema è stato progettato per essere facilmente leggibile, estendibile e documentabile, qualità essenziali in un progetto in continua evoluzione.

Queste scelte progettuali non solo rispondono alle esigenze attuali della piattaforma, ma pongono anche le basi per sviluppi futuri, come:

- l'introduzione di nuove tipologie di esercizi, alimenti o tecniche d'allenamento;
- l'ampliamento del supporto multimediale (es. video o istruzioni vocali);
- il supporto ad una applicazione mobile.

In sintesi, la modellazione adottata per *SmartFit Coach* si configura come una soluzione **robusta, estensibile e pronta al futuro**, capace di sostenere le funzionalità attuali e l'evoluzione naturale del progetto in direzione di un assistente intelligente per il benessere nutrizionale e fisico.

# Capitolo 5

# Interfaccia utente dell'applicazione

L'interfaccia utente rappresenta il punto di contatto tra l'utente finale e le funzionalità offerte da *SmartFit Coach*. Per questo motivo, la sua progettazione è stata affrontata con grande attenzione, con l'obiettivo di garantire un'esperienza d'uso fluida, intuitiva e accessibile anche a chi non possiede particolari competenze tecnologiche.
Dal punto di vista tecnico, l'interfaccia è stata sviluppata utilizzando il framework **SvelteKit**, che consente la creazione di componenti reattivi, leggeri e altamente performanti. Lo stile visivo si basa sulle linee guida del **Material Design 3**, assicurando coerenza tra le diverse sezioni della piattaforma, ottima leggibilità e una gerarchia visiva ben definita.
L'esperienza utente è stata pensata per adattarsi perfettamente a dispositivi mobili, vista la natura personale e quotidiana con cui la piattaforma viene utilizzata. Ogni componente è progettato per rispondere in tempo reale alle interazioni dell'utente, offrendo un feedback immediato e mantenendo la coerenza visiva tra azione e risultato.
Le funzionalità sono organizzate in macro-sezioni principali:

- la gestione del **profilo utente**, con informazioni personali, obiettivi e immagine di profilo;
- la sezione **analisi dei progressi**, che consente il monitoraggio del peso e delle misure corporee nel tempo;

- la sezione dedicata alla **scheda alimentare**, con navigazione a lista, visualizzazione dei pasti giornalieri, valori nutrizionali e possibilità di inserimento rapido di alimenti;
- la sezione dedicata alla **scheda di allenamento**, con navigazione a lista settimanale, dettagli per ciascun esercizio e inserimento delle serie effettuate;
- infine, le **interazioni con l'IA**, che supportano l'utente nella generazione automatica di feedback e contenuti, tramite pulsanti ben evidenziati.

In questo capitolo verranno analizzati nel dettaglio i principali componenti dell'interfaccia, le logiche UX alla base delle scelte progettuali, e il modo in cui la UI interagisce in tempo reale con il backend e i moduli intelligenti per garantire un'esperienza completa e coerente.

## 5.1 Struttura generale dell'interfaccia

La homepage di *SmartFit Coach* accoglie l'utente con un'interfaccia semplice e pulita, pensata per offrire un **impatto visivo immediato** e guidare l'interazione fin dal primo accesso. La schermata si presenta con una struttura a **layout laterale**, in cui un *barra laterale verticale testuale* a sinistra permette una navigazione rapida verso le sezioni principali pubblicamente accessibili: `Home`, `Chi siamo`, `Contatti`, `Accedi` e `Registrati` (se loggati, quest'ultimi due pulsanti vengono sostituiti dal pulsante `Account`).

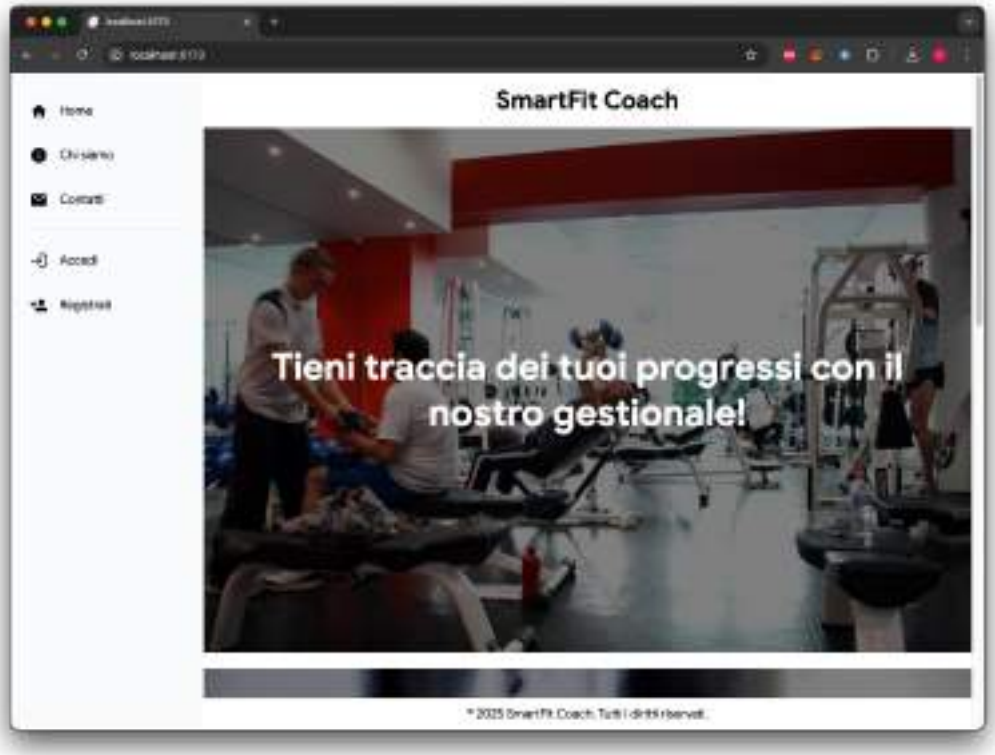


Figura 5.1: Homepage iniziale di SmartFit Coach

Al centro è sovrapposto un **messaggio promozionale chiaro e motivante**. Questo slogan ha lo scopo di comunicare immediatamente la **funzionalità centrale dell'app**: aiutare l'utente a monitorare e gestire in modo efficiente i propri percorsi alimentari e di allenamento.
Questa homepage funge da **landing page informativa e orientativa**, rivolta sia a nuovi visitatori che vogliono saperne di più sul servizio, sia a utenti registrati che desiderano accedere rapidamente alla propria area riservata. L'obiettivo è trasmettere subito un senso di professionalità, ordine e semplicità.

## 5.2 Area account utente

L'area dell'**account utente** rappresenta il cuore pulsante di SmartFit Coach. È da qui che l'utente gestisce in modo centralizzato tutti gli aspetti legati al proprio profilo: **dati personali**, **peso**, **misure corporee**, **schede alimentari** e **schede di allenamento**. Questa area costituisce un punto di accesso

unificato per monitorare l'evoluzione del proprio percorso, aggiornare i dati in tempo reale e interagire con i moduli intelligenti della piattaforma.
L'interfaccia si presenta con una **barra laterale verticale a icone**, sempre visibile, che consente una navigazione rapida e intuitiva tra le varie sezioni del profilo. Il menu è progettato per mantenere la semplicità visiva e l'accessibilità su qualsiasi dispositivo, specialmente mobile.
Ogni icona corrisponde a una specifica area funzionale:

- **Avatar profilo (account)**: accesso alle informazioni anagrafiche e biometriche dell'utente, come nome, data di nascita, altezza, genere, obiettivi e immagine di profilo;
- **Peso**: inserimento e visualizzazione delle precedenti misure del peso;
- **Misure corporee**: inserimento e visualizzazione delle precedenti misure corporee (torace, vita, coscia, spalle, ecc.);
- **Schede alimentari**: sezione dedicata alla consultazione e modifica delle schede alimentari create dall'utente;
- **Schede di allenamento**: sezione dedicata alla consultazione e modifica delle schede di allenamento, con esercizi, tecniche e serie;
- **Logout**: uscita sicura dall'area personale e terminazione della sessione attiva.

L'immagine del profilo è mostrata in alto come avatar circolare, personalizzabile dall'utente. Il design minimale ma funzionale consente di mantenere sempre visibili le voci fondamentali senza compromettere lo spazio centrale dell'interfaccia.
Nel complesso, l'area account utente è progettata per essere il **centro di controllo** dell'esperienza utente, da cui partire per costruire, modificare e monitorare ogni aspetto del proprio percorso di benessere.

### 5.2.1 Pagina dettagli dell'account

All'interno della sezione **account**, l'utente ha la possibilità di visualizzare e aggiornare i propri **dati anagrafici e biometrici** attraverso un'interfaccia intuitiva e ben strutturata.

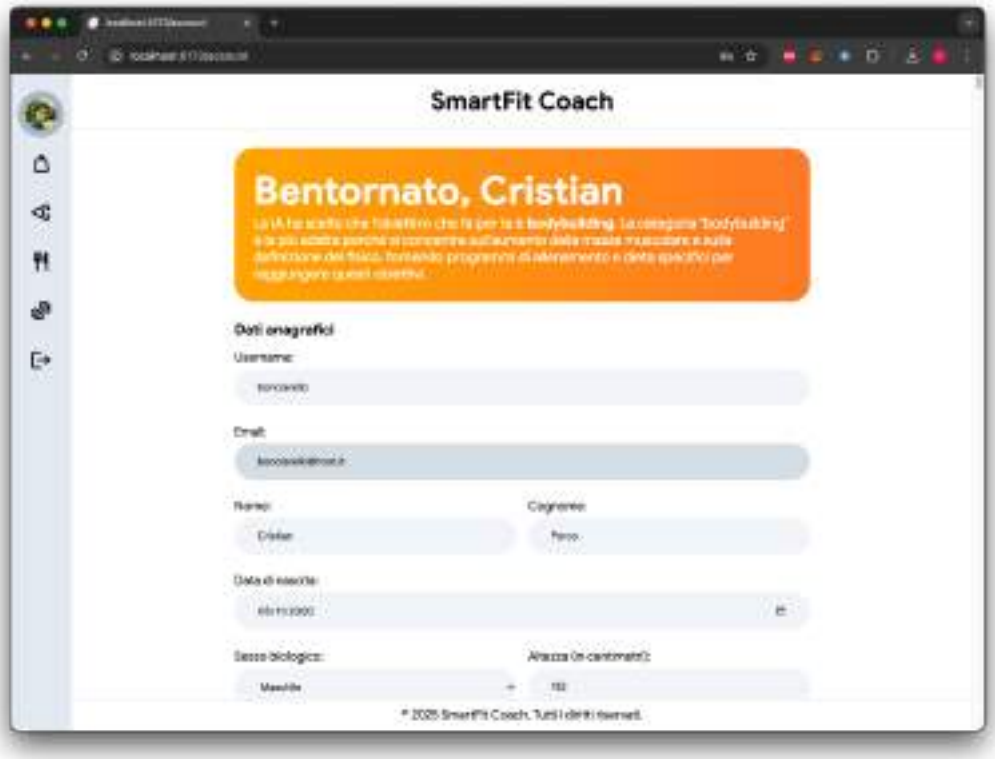


Figura 5.2: Pagina dettagli dell'account sui dati anagrafici e biometrici

La parte superiore della schermata è occupata da un messaggio di benvenuto generato dinamicamente, che integra l'output dell'intelligenza artificiale. Subito sotto si trovano i campi precompilati che l'utente può rivedere e, se necessario, modificare:

- **Username** e **Email**: identificativi dell'account utente, dove l'email è mostrato in sola lettura;
- **Nome** e **Cognome**: dati anagrafici personali modificabili;
- **Data di nascita**: selezionabile tramite un componente calendario;

- **Sesso biologico**: valore selezionabile tra le opzioni disponibili `Maschio` e `Femmina`;
- **Altezza (in centimetri)**: campo numerico validato, corrispondente all'attributo `height_cm`.

Questi dati, gestiti a livello di backend tramite il modello `DetailsAccount`, costituiscono la **base informativa** su cui si costruiscono tutte le funzionalità personalizzate della piattaforma. L'interfaccia è progettata per favorire l'inserimento e l'aggiornamento rapido delle informazioni, garantendo al contempo coerenza e integrità dei dati utente.

### 5.2.2 Configurazione dell'obiettivo personale

Un altro elemento fondamentale nella personalizzazione del profilo è la definizione dell'**obiettivo individuale**, che viene raccolto tramite un campo testuale libero e interpretato successivamente dal sistema di intelligenza artificiale. L'interfaccia offre all'utente un'area dedicata nella quale può rispondere alla domanda: *"Qual è il tuo obiettivo?"*.


Figura 5.3: Pagina dettagli dell'account sull'obiettivo personale.

Il campo sottostante consente di inserire una descrizione breve e spontanea delle proprie intenzioni, ad esempio: *"Il mio obiettivo è aumentare la massa muscolare e scolpire il fisico."*. Questo input viene salvato nel database nel campo:

- `goal_description`: campo testuale (`CharField`) con limite di 160 caratteri, che raccoglie la descrizione libera dell'obiettivo.

Una volta inserito, questo contenuto viene elaborato dall'intelligenza artificiale, che tenta di associare automaticamente l'utente a una delle **categorie di obiettivo predefinite**, corrispondenti al campo:

- `goal_targets`: campo con scelte limitate (`choices`) tra:
  - `Fitness`
  - `Bodybuilding`
  - `Powerlifting`
  - `Streetlifting`

Questo processo consente alla piattaforma di guidare la creazione automatizzata delle schede alimentari e di allenamento in modo più preciso e coerente con le preferenze dell'utente, pur lasciando a quest'ultimo la possibilità di modificare l'input liberamente in qualsiasi momento.

### 5.2.3 Storico del peso

La sezione dedicata allo **storico del peso** consente all'utente di registrare in modo semplice e immediato le proprie pesate, monitorandone l'andamento nel tempo.


Figura 5.4: Pagina dello storico del peso.

L'interfaccia presenta una sezione di inserimento composta da due campi: *"Peso"* (espresso in chilogrammi) e *"Data di registrazione"*, selezionabile tramite calendario. Una volta compilati i dati, è sufficiente cliccare sul pulsante **Aggiungi** per memorizzare la misurazione.

Subito dopo la sezione di inserimento è presente un **grafico interattivo** che visualizza l'evoluzione del peso nel corso dei giorni registrati. Questo strumento fornisce un riscontro visivo immediato sull'andamento delle misurazioni, aiutando l'utente a riconoscere progressi, stalli o oscillazioni significative.

Nella parte inferiore del grafico è inoltre presente un pulsante evidenziato che invita l'utente a **ottenere un'analisi intelligente sui dati**. Tale analisi, gestita da moduli di intelligenza artificiale, può fornire feedback personalizzati sull'andamento del peso e segnalazioni di eventuali anomalie.

Oltre alla registrazione e visualizzazione grafica delle pesate, l'interfaccia offre una sezione dedicata allo **storico completo delle misurazioni**. Questa area consente all'utente di consultare in modo dettagliato tutte le pesate effettuate

nel tempo, fornendo una panoramica cronologica dei progressi raggiunti. I dati sono presentati in una lista ordinata per data, facilitando l'individuazione di pattern e tendenze nel tempo.

### 5.2.4 Storico delle misure corporee

Oltre al monitoraggio del peso, *SmartFit Coach* consente anche la registrazione e il controllo delle **misure corporee**, offrendo uno strumento più completo per valutare i cambiamenti fisici dell'utente nel tempo. Questa funzionalità è particolarmente utile per chi pratica attività fisica con obiettivi di ricomposizione corporea, dove la sola variazione del peso può non essere sufficiente a descrivere i progressi reali.

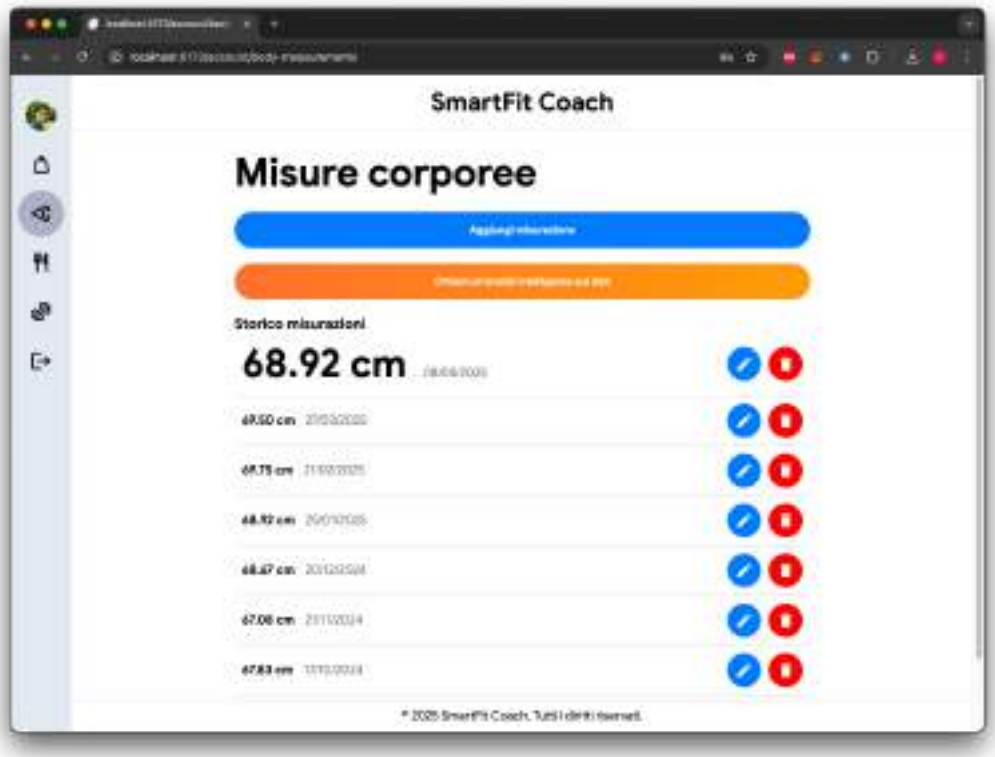


Figura 5.5: Pagina dello storico delle misurazioni corporee.

L'interfaccia della sezione **Misure corporee** consente di:

- aggiungere una nuova misurazione;

- richiedere un'analisi intelligente sull'andamento tramite IA;
- visualizzare lo **storico delle misure** in ordine cronologico, con possibilità di modifica o eliminazione.

Nel contesto dell'interfaccia utente, le misurazioni visualizzate rappresentano una media calcolata automaticamente, utile per fornire un riferimento rapido. La presenza delle icone di *modifica* e di *eliminazione* consente una gestione puntuale dello storico.
L'integrazione con un sistema di **analisi intelligente** permette di valutare le variazioni delle misurazioni nel tempo, evidenziare eventuali miglioramenti o regressi, e ricevere **feedback personalizzati**. Questo strumento sfrutta i dati raccolti per restituire all'utente una **panoramica dinamica e contestualizzata** del proprio percorso fisico. L'analisi non si limita a una semplice lettura numerica delle misure, ma tiene conto anche degli **obiettivi selezionati dall'utente**.

## 5.3 Gestione delle schede personalizzate

*SmartFit Coach* fornisce un'interfaccia dedicata alla **gestione delle schede alimentari e di allenamento**, due elementi fondamentali per la personalizzazione del percorso di benessere dell'utente. Attraverso un'interfaccia semplice e intuitiva, l'utente può visualizzare in tempo reale i contenuti delle schede attive, modificarne i dettagli oppure generarne di nuove con l'aiuto dell'intelligenza artificiale.
Le **schede alimentari** sono strutturate su base giornaliera o intervallo di date e suddivise in **fasce orarie** (es. colazione, pranzo, cena). All'interno di ciascuna fascia è possibile visualizzare gli alimenti assegnati, modificarne le quantità, aggiungerne di nuovi oppure rimuoverli.
Le **schede di allenamento**, invece, seguono solo una logica settimanale e sono composte da sessioni suddivise per giorno della settimana. Ogni giornata contiene uno o più esercizi ordinati, ciascuno con la relativa configurazione di serie, ripetizioni, peso, riposo e tecniche d'intensità.

Entrambe le sezioni sono progettate per offrire all'utente finale il massimo controllo, pur mantenendo semplicità nell'interazione, grazie a un'interfaccia reattiva e coerente tra le due aree. Inoltre, grazie all'integrazione con il sistema di **intelligenza artificiale**, è possibile **aggiungere o modificare automaticamente** gli elementi di entrambe le schede, ricevendo suggerimenti personalizzati in base ai dati inseriti e agli obiettivi dichiarati dall'utente.

### 5.3.1 Schede alimentari

Le **schede alimentari** costituiscono una delle sezioni centrali di *SmartFit Coach*, permettendo agli utenti di pianificare la propria alimentazione in modo preciso, strutturato e adattabile.

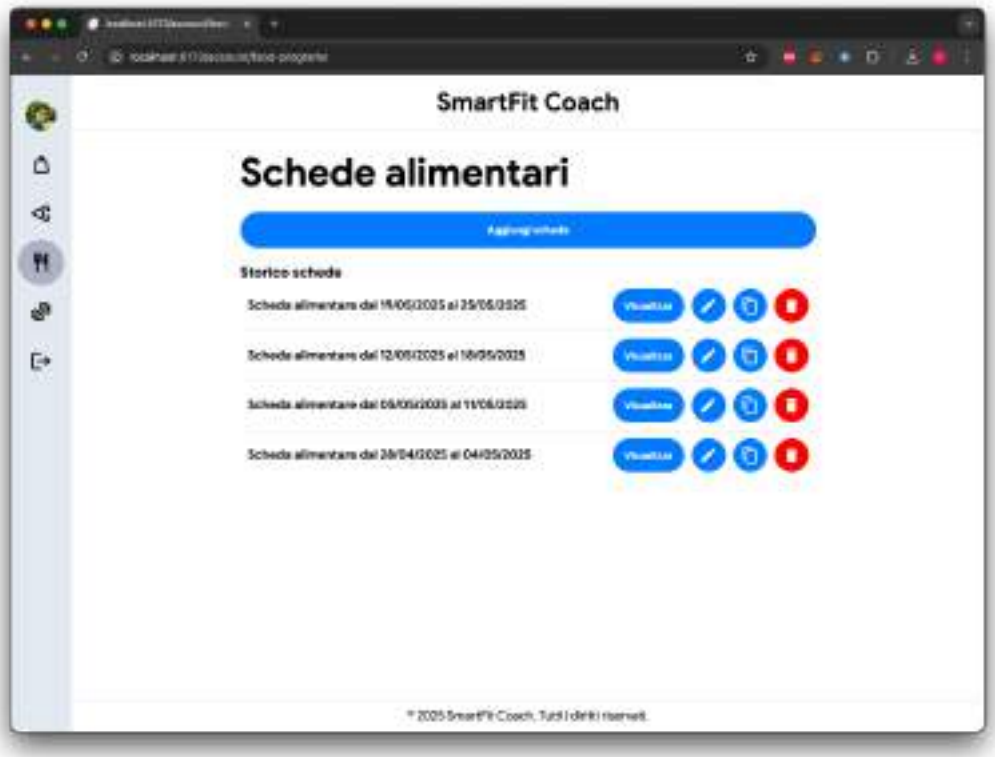


Figura 5.6: Pagina della lista delle schede alimentari dell'utente.

L'interfaccia presenta un elenco delle schede alimentari dall'utente con indicazione dell'intervallo temporale e un set di azioni rapide: *Visualizza*, *Modifica*, *Duplica* e *Elimina*.


Figura 5.7: Pagina della visualizzazione della specifica scheda alimentare dell'utente (parte 1).

Durante la **visualizzazione** di una scheda alimentare, gli alimenti sono organizzati per pasto (es. Colazione, Spuntino, Pranzo, Cena), ciascuno associato ad un orario previsto. Per ogni alimento vengono mostrati nome, quantità in grammi e valori nutrizionali per porzione (calorie, proteine, carboidrati, zuccheri, grassi, fibre).


Figura 5.8: Pagina della visualizzazione della specifica scheda alimentare dell'utente (parte 2).

L'interfaccia evidenzia automaticamente la somma dei nutrienti consumati rispetto ai limiti impostati.

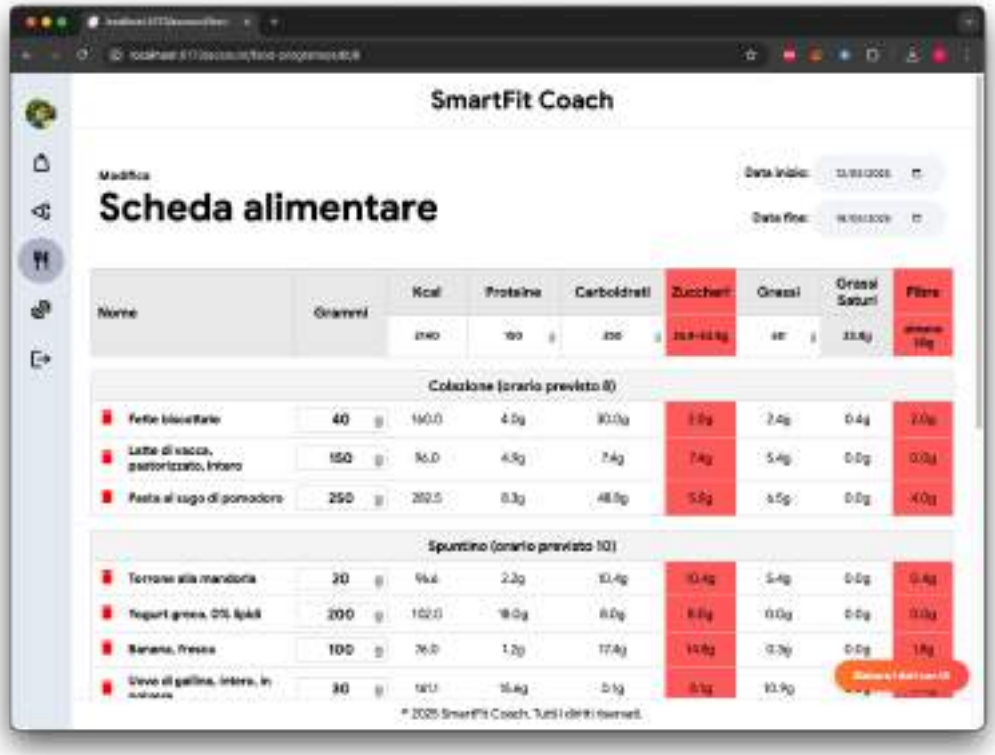


Figura 5.9: Pagina della modifica della specifica scheda alimentare dell'utente.

Durante la **modifica**, l'utente può aggiungere, eliminare o sostituire alimenti per ciascun pasto. I valori nutrizionali vengono ricalcolati in tempo reale. Le colonne che eccedono o non raggiungono i range consigliati vengono evidenziate visivamente con colore rosso, facilitando l'individuazione di squilibri.
Un elemento distintivo della piattaforma è il supporto avanzato offerto dall'intelligenza artificiale, perfettamente integrato all'interno di un pulsante laterale evidenziato e sempre accessibile durante la compilazione o la revisione della scheda alimentare. Questo strumento intelligente offre una serie di funzionalità evolute che semplificano e ottimizzano il processo di pianificazione alimentare. Nello specifico, l'utente ha la possibilità di:

- **inserire alimenti all'interno dei vari pasti semplicemente descrivendo in linguaggio naturale ciò che si è effettivamente mangiato**, ad esempio: "Ho mangiato una porzione abbondante di pasta al sugo

e qualche patata fritta", ottenendo in risposta l'elenco degli alimenti riconosciuti insieme a una stima automatica delle quantità corrispondenti;

- **caricare un'immagine del proprio pasto per avviare un processo automatico di riconoscimento degli alimenti presenti nel piatto**, proponendo una stima accurata degli ingredienti rilevati;
- **ottimizzare le quantità degli alimenti inseriti** in modo da rispettare in maniera precisa i target calorici e i fabbisogni nutrizionali giornalieri (in termini di macronutrienti come carboidrati, proteine e grassi);
- **generare pasti alternativi per una determinata fascia oraria**, in modo da offrire varietà nella dieta mantenendo al contempo il rispetto dei vincoli nutrizionali, delle preferenze alimentari e di eventuali intolleranze;
- **creare in maniera automatica e completa l'intera scheda alimentare** a partire dall'obiettivo scelto dall'utente o dai macronutrienti scelti alla creazione della scheda. Questa funzionalità è visibile e attivabile solamente nel caso in cui la scheda sia completamente vuota, ovvero priva di alimenti o sezioni già compilate, ed è pensata per facilitare l'avvio rapido da parte di nuovi utenti o di chi desidera una proposta offerta dall'intelligenza artificiale personalizzata da cui partire.

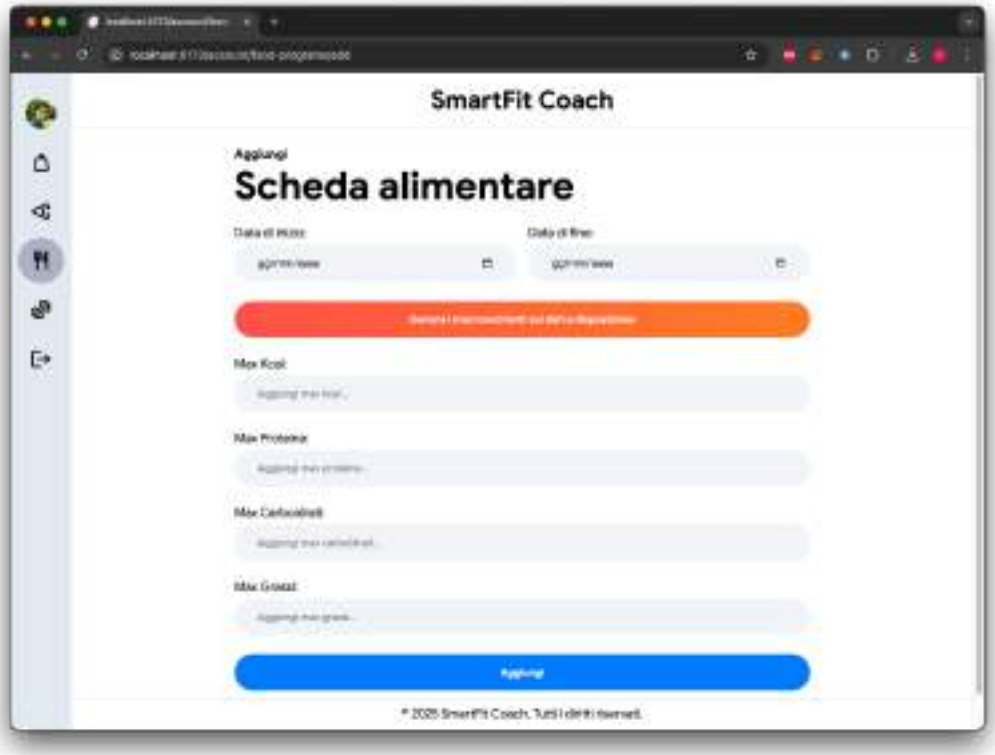


Figura 5.10: Pagina della aggiunta di una nuova scheda alimentare.

Infine, il processo di **aggiunta** di una nuova scheda consente di compilare manualmente i campi relativi all'intervallo di validità e ai limiti nutrizionali oppure di delegarne la generazione automatica all'intelligenza artificiale, che stima i valori consigliati in base all'obiettivo scelto, peso e misure corporee dell'utente.

### 5.3.2 Schede di allenamento

Le **schede di allenamento** costituiscono una delle sezioni fondamentali di *SmartFit Coach*, consentendo agli utenti di pianificare il proprio percorso di allenamento in modo strutturato, settimanale e completamente personalizzabile.

La struttura permette di organizzare gli esercizi per ciascun giorno della settimana, assegnando a ogni esercizio ripetizioni, carico, RIR, tempo di esecuzione, recupero e, ove necessario, tecniche d'intensità. L'utente può costruire

la propria scheda manualmente oppure affidarsi all'intelligenza artificiale per ricevere suggerimenti mirati sulla base degli obiettivi e dello stato fisico inseriti nel profilo.

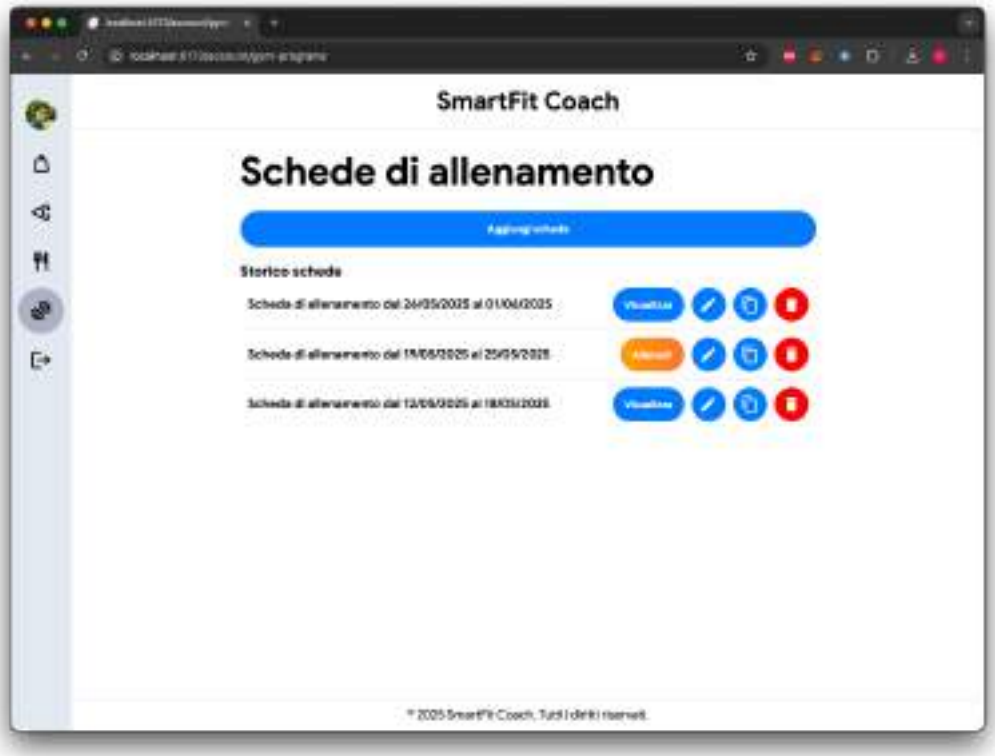


Figura 5.11: Pagina della lista delle schede di allenamento dell'utente.

L'interfaccia presenta un elenco delle schede di allenamento settimanali con indicazione dell'intervallo temporale e un set di azioni rapide: *Visualizza*, *Modifica*, *Duplica* e *Elimina*. Il pulsante *Allenati!*, al posto del pulsante *Visualizza*, evidenzia la scheda della settimana attuale, cioè della settimana in corso.

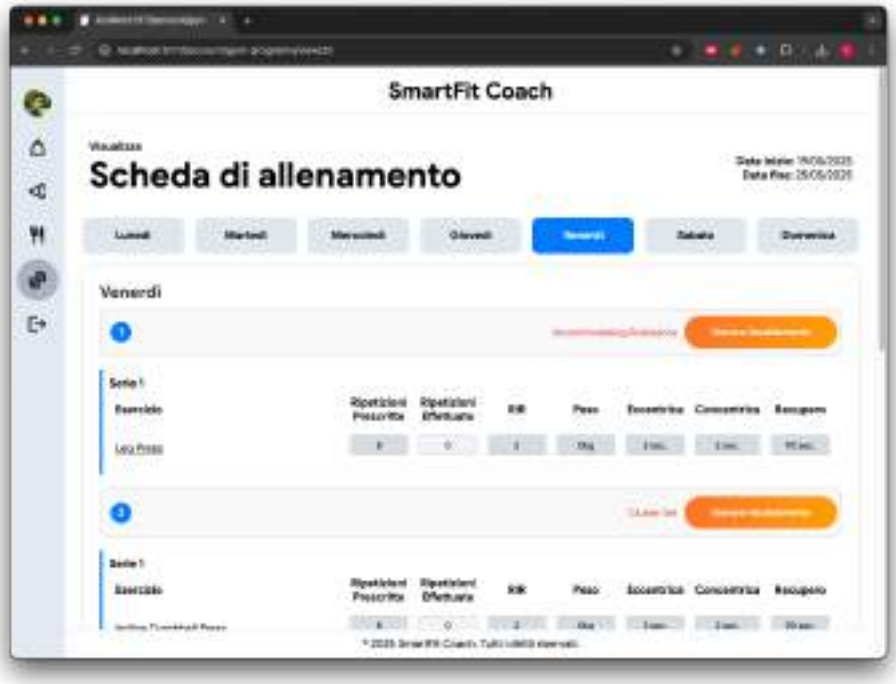


Figura 5.12: Pagina della visualizzazione della scheda di allenamento dell'utente.

Durante la **visualizzazione** di una scheda di allenamento, è possibile consultare il contenuto giorno per giorno, cliccando sulle voci della settimana. Ogni giornata mostra gli esercizi previsti, suddivisi in blocchi numerati, con indicazione di:

- ripetizioni prescritte e effettuate,
- carico utilizzato (in kg),
- valore RIR,
- tempo eccentrico/concentrico/fermo,
- tempo di recupero tra le serie.

Se i campi descritti sopra sono nulli (cioè pari a 0), non verranno visualizzati all'interno dei dettagli della serie.

Seppur una piccola funzionalità, l'intelligenza artificiale può generare uno o più serie di riscaldamento all'esercizio prescelto, per prevenire eventuali infortuni dell'utente.

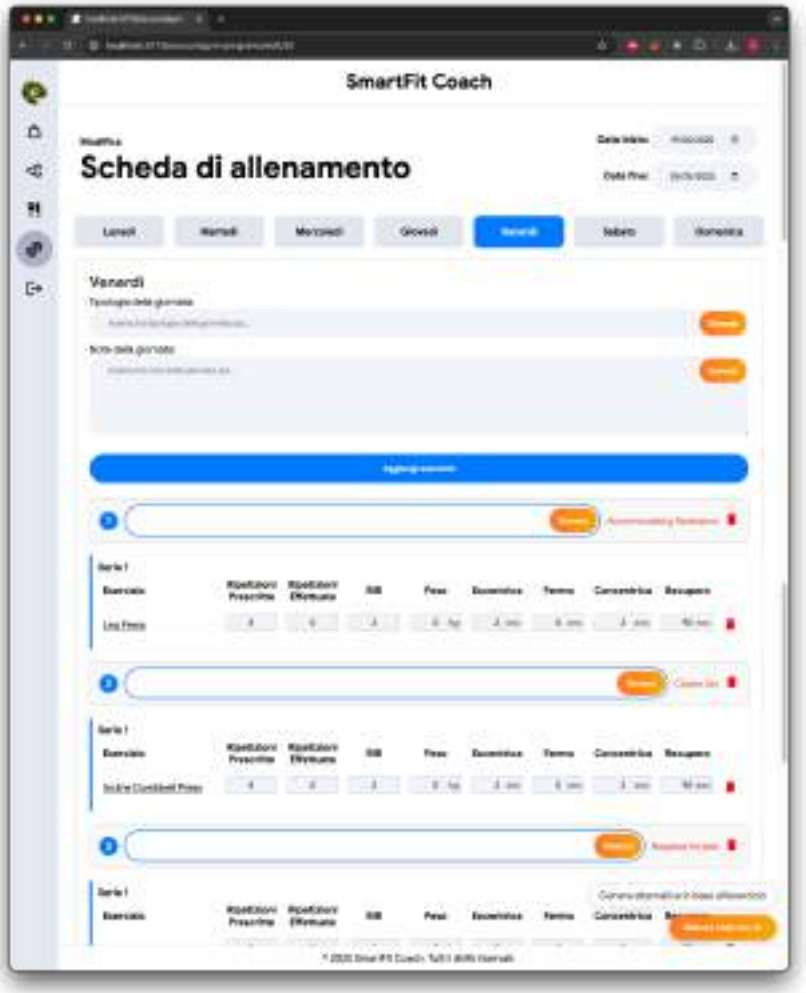


Figura 5.13: Pagina della modifica della scheda di allenamento dell'utente.

Durante la **modifica**, l'utente può inserire o aggiornare informazioni come:

- tipologia e note per ogni giornata, inseribili manualmente o generabili tramite intelligenza artificiale;
- esercizi per giorno, inseribili manualmente o suggeriti tramite intelligenza artificiale;
- configurazione completa di ogni serie tramite input numerici.

Il sistema permette una personalizzazione molto fine della scheda di allenamento, adattandolo a ogni livello di esperienza. Le tecniche d'intensità supportate sono numerose (citate nella sezione 4.3.4) e vengono visualizzate accanto a ciascun esercizio. L'inserimento di note e la generazione assistita rendono il modulo utile sia all'utente autonomo che al personal trainer.
Una funzionalità distintiva è il supporto integrato all'**intelligenza artificiale**, accessibile tramite pulsanti evidenziati "*Genera*" o "*Elabora i dati con IA*". Questo sistema consente di:

- generare automaticamente l'intera scheda settimanale o singole giornate;
- generare una giornata in base al tipo di giornata selezionata;
- generare alternative a esercizi già presenti.

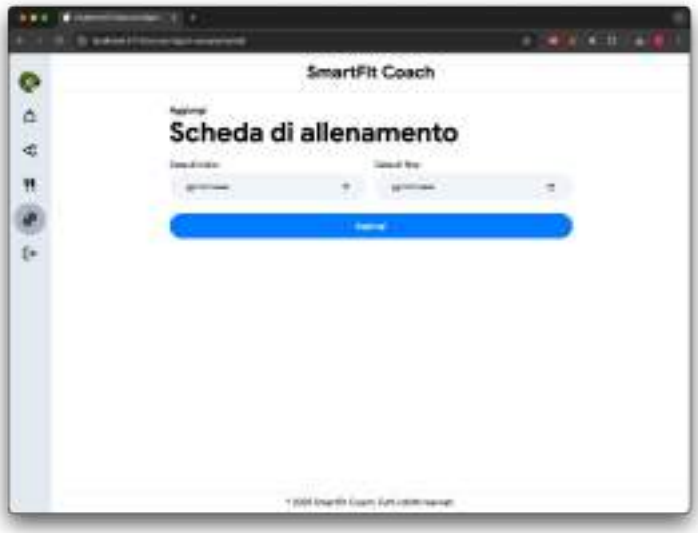


Figura 5.14: Pagina dell'aggiunta della scheda di allenamento dell'utente.

Infine, il processo di **aggiunta** di una nuova scheda consente di compilare manualmente i campi relativi all'intervallo di validità.
Nel complesso, le schede di allenamento offrono un'esperienza avanzata e guidata, con un'interfaccia chiara e il pieno controllo sui parametri. L'integrazione dell'IA permette non solo di risparmiare tempo, ma anche di ottenere proposte coerenti con gli obiettivi personali dell'utente.

# Capitolo 6

# Integrazione dell'IA e prompt engineering

## 6.1 Classificazione dell'obiettivo dell'utente

Il seguente prompt descrive il funzionamento di una catena basata su modello linguistico, impiegata per analizzare automaticamente gli obiettivi di allenamento dell'utente espressi in linguaggio naturale.

```
"Dato il seguente obiettivo descritto da un utente, restituisci
   una sola parola tra:
fitness, bodybuilding, powerlifting, streetlifting.

Descrizione: "{description}""
```

Data una descrizione libera dell'obiettivo dell'utente (es. *"Voglio diventare più forte e migliorare la mia forma fisica"*), il sistema utilizza un modello di intelligenza artificiale per interpretare semanticamente il contenuto testuale e assegnare una sola categoria tra le seguenti: **fitness**, **bodybuilding**, **powerlifting** e **streetlifting**.

L'inferenza non si basa su semplici parole chiave, ma sull'intento complessivo espresso nella frase: questo ci consente di effettuare una classificazione intelligente e flessibile, utile per personalizzare feedback, schede alimentari e di allenamento all'interno di piattaforme fitness.

## 6.2 Generazione di una spiegazione per la classificazione dell'obiettivo

Il seguente prompt descrive una catena che utilizza un modello linguistico per fornire una spiegazione sintetica e comprensibile della classificazione assegnata a un obiettivo di allenamento espresso in linguaggio naturale.

```
"Hai classificato l'obiettivo utente come "{category}".
Obiettivo dell'utente: "{description}"

Spiega brevemente (massimo 300 caratteri) perchè questa categoria
    èla più adatta."
```

A partire da due input:

- la **descrizione** dell'obiettivo fornita dall'utente;
- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);

il modello genera una breve giustificazione (massimo 300 caratteri) che chiarisce perché la categoria scelta rappresenta in modo appropriato l'intento espresso dall'utente.
Questo generazione è utile per visualizzare all'utente il motivo della scelta del precedente prompt e fornisce un riscontro interpretabile e "umano" che può essere utile per aumentare la fiducia dell'utente e facilitare eventuali revisioni da parte di un nutrizionista o personal trainer.

## 6.3 Analisi dell'andamento del peso rispetto all'obiettivo

Il seguente prompt descrive una catena che utilizza un modello linguistico per generare un'analisi sintetica e professionale dell'andamento del peso corporeo dell'utente nel tempo, in relazione al proprio obiettivo di allenamento.

```
"L'utente ha come obiettivo "{goal}".
Ecco i suoi dati di peso nel tempo:
```

```
{weights}

Scrivi un'analisi sintetica (massimo 500 caratteri)
   sull'andamento del peso rispetto all'obiettivo.
Sii chiaro, professionale e conciso."
```

Il sistema riceve in input:

- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);
- una **sequenza di misurazioni del peso** nel tempo (date e valori in chilogrammi).

Sulla base di questi dati, l'intelligenza artificiale valuta se la variazione del peso è coerente con l'obiettivo indicato. Ad esempio, un incremento progressivo è generalmente positivo per il bodybuilding, mentre un calo costante può essere auspicabile in un contesto di fitness.
Il risultato è un breve commento professionale, entro i 500 caratteri, utile per fornire un feedback automatico e comprensibile all'utente e supportare l'utente nel monitoraggio continuo dei progressi, rendendo i dati più leggibili.

## 6.4 Analisi dell'andamento delle misure corporee rispetto all'obiettivo

Il seguente prompt descrive una catena che utilizza un modello linguistico per generare un'analisi sintetica e professionale dell'andamento delle misure corporee corporeo dell'utente nel tempo, in funzione dell'obiettivo dichiarato dall'utente.

```
"L'utente ha come obiettivo: "{goal}".
Questi sono i suoi dati di misurazione corporea nel tempo:

{measurements}

Scrivi un'analisi sintetica (massimo 500 caratteri)
   sull'andamento delle singole aree corporee rispetto
   all'obiettivo.
Sii chiaro, professionale e conciso."
```

Il sistema riceve in input:

- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);
- una **sequenza di misure corporee** nel tempo es. torace, vita, braccia, ecc.), con date associate e valori espressi in centimetri.

Sulla base di queste informazioni, il modello valuta se le variazioni corporee registrate sono coerenti con l'obiettivo, quali aree mostrano miglioramenti, stabilità o regressi ed eventuali progressi significativi o tendenze rilevanti.
Il risultato è un breve commento professionale, entro i 500 caratteri, utile per fornire un feedback automatico e comprensibile all'utente e supportare l'utente nel monitoraggio continuo dei progressi, rendendo i dati più leggibili.

## 6.5 Estrazione strutturata dei pasti da una frase in linguaggio naturale

Il seguente prompt descrive una catena basata su modello linguistico che analizza una descrizione di un pasto in linguaggio naturale, convertendole in una rappresentazione strutturata utile per applicazioni di tracciamento alimentare.

```
"Dalla seguente frase in linguaggio naturale:

"{input_text}"

1. Identifica ogni pasto o alimento menzionato.
2. Per ciascuno, genera un oggetto JSON con i seguenti campi:
   - "meal": nome sintetico del pasto o alimento (es. "Pasta al
     sugo")
   - "keywords": elenco di parole strettamente legate al pasto
     (ingredienti, alimenti, termini specifici).
     Per ogni parola chiave, includi **sia la forma singolare che
     quella plurale**, se rilevante (es. "patata", "patate").
   - "quantity": stima della quantità in grammi, espressa come
     numero intero (es. 150).
     Se nella frase ci sono termini qualitativi (es.
     "abbondante", "qualche", "una porzione"), interpreta e
     traduci in un valore numerico coerente.

Restituisci esclusivamente un array JSON. Nessuna spiegazione o
   testo fuori dal JSON."
```

A partire da un input testuale che descrive cosa è stato mangiato (es. *"Ho mangiato una pasta al sugo con delle patate fritte"*), il sistema genera un array JSON in cui ciascun elemento rappresenta un pasto o alimento identificato. Per ogni voce estratta, vengono forniti:

- **meal**: una descrizione sintetica del pasto (es. `"Pasta al sugo"`);
- **keywords**: un elenco di parole chiave correlate, comprensivo di forma singolare e plurale (es. `["patata", "patate"]`), da utilizzare nel prompt spiegato nella Sezione 6.6;
- **quantity**: una stima numerica (in grammi) della quantità consumata, anche in assenza di riferimenti espliciti, interpretando anche espressioni qualitative (come *"una porzione"* o *"qualche patatina"*) traducendole in quantità approssimative coerenti.

Il risultato è un output JSON puro, senza testo descrittivo aggiuntivo, progettato per essere facilmente integrato nella scheda alimentare di riferimento.

## 6.6 Selezione semantica dell'alimento più pertinente

Il seguente prompt descrive una catena basata su modello linguistico, progettata per selezionare automaticamente l'alimento più coerente da una lista di candidati, a partire dal nome di un pasto fornito in linguaggio naturale.

```
"Tra i seguenti alimenti candidati, scegli quello che corrisponde
    meglio al pasto descritto:

Nome del pasto: "{meal}"

Candidati disponibili:
{candidates}

Restituisci esclusivamente il nome dell'alimento selezionato
    (nessuna spiegazione, solo testo)."
```

Il sistema riceve in input:

- un **nome di un pasto** (es. *"Pasta al sugo"*, *"Salsiccia con spinaci*);

- un **elenco di alimenti candidati**, ciascuno con un nome strutturato (provenienti dal database).

Il modello valuta semanticamente il nome del pasto e confronta ogni candidato per similarità linguistica e coerenza semantica. È in grado di gestire:

- sinonimi e variazioni terminologiche (es. *"riso con pollo"* ↔ *"piatto unico pollo e riso"*);
- nomi incompleti o descrittivi;
- disallineamenti ortografici minori.

Il risultato è il **solo nome dell'alimento** selezionato, senza ulteriori spiegazioni o testo descrittivo. L'associazione può poi essere usata per aggiungere l'alimento alla scheda alimentare.
Se non viene trovata alcuna corrispondenza, il sistema restituisce un risultato nullo, segnalando l'assenza di un match coerente.

## 6.7 Ottimizzazione nutrizionale di una scheda alimentare

Il seguente prompt descrive una catena basata su modello linguistico, progettata per ottimizzare le quantità degli alimenti in una scheda alimentare al fine di rispettare vincoli nutrizionali specificati alla creazione della scheda stessa. L'obiettivo è avvicinarsi il più possibile ai valori massimi consentiti per ogni macronutriente (calorie, proteine, carboidrati, grassi), senza mai superarli, e garantire contemporaneamente il raggiungimento dei valori minimi previsti.

---

```
"L'obiettivo èottimizzare un piano alimentare in termini di
    nutrienti, considerando solo i valori nutrizionali degli
    alimenti, e ignorando il contenuto semantico o la varietà.

Il piano deve rispettare i seguenti vincoli nutrizionali totali:
- Calorie: massimo {max_kcal} kcal
- Proteine: minimo {min_protein} g, massimo {max_protein} g
- Carboidrati: minimo {min_carbs} g, massimo {max_carbs} g
- Grassi: minimo {min_fats} g, massimo {max_fats} g
```

```
L'obiettivo èavvicinarsi **quanto più possibile ai limiti
    massimi**, senza mai superarli, **e garantire il
    raggiungimento dei valori minimi** per ciascun macronutriente.

Ecco l'elenco degli alimenti disponibili nel piano (ogni alimento
    ha quantità iniziale e valori per 100g):

{food_items}

Regole da rispettare:
- Modifica le quantità (in grammi) per ogni alimento.
- Ogni quantità deve essere **compresa tra 10g e 200g**.
- **Non sono ammesse quantità pari a 0g.**
- **Non superare alcun valore massimo.**
- **Raggiungi almeno i valori minimi.**
- Ignora il tipo di alimento, lavora solo sui numeri.

Rispondi solo con un JSON **puro**, senza blocchi di codice,
    backtick o etichette. Il risultato deve essere solo l'array
    JSON, ad esempio:

[
  {{ "id": 1, "adjusted_quantity_in_grams": 100 }},
  {{ "id": 2, "adjusted_quantity_in_grams": 150 }}
]"
```

---

Il sistema riceve in input:

- i **limiti nutrizionali target** della scheda (minimi e massimi per kcal, proteine, carboidrati, grassi);
- un **elenco di alimenti**, ciascuno con:
  - quantità iniziale in grammi;
  - valori nutrizionali per 100g (kcal, proteine, carboidrati, grassi, fibre, zuccheri).

Il risultato è un array JSON che specifica, per ogni alimento, una nuova quantità (compresa tra 10g e 200g) da adottare nel piano. Le regole seguite sono:

- nessuna quantità può essere azzerata;
- non è consentito superare i limiti massimi indicati;
- ogni minimo nutrizionale deve essere raggiunto;

- non viene considerata la varietà o tipologia degli alimenti, ma solo i valori numerici.

Grazie a questo approccio, è possibile ottenere una scheda alimentare matematicamente bilanciata, mantenendo la coerenza con le regole dietetiche impostate.

## 6.8 Generazione di una scheda alimentare personalizzata

Il seguente prompt descrive una catena basata su modello linguistico avanzato, progettata per generare una scheda alimentare completa e personalizzata a partire dai dati recenti dell'utente (obiettivo, peso, misure corporee, ecc.).

```
"L'utente ha come obiettivo nutrizionale: "{goal}".

Ecco i suoi dati recenti:
- Peso nell'ultimo mese:
{weights}

- Misure corporee principali dell'ultimo mese:
{measurements}

Nella precedente scheda alimentare seguiva questi macro:
- Proteine: {prev_protein} g
- Carboidrati: {prev_carbs} g
- Grassi: {prev_fats} g
- Calorie: {prev_kcal} kcal

Genera un piano alimentare giornaliero completo, suddiviso in 4-6
    pasti.

Ogni alimento deve essere un oggetto separato. Es. "Yogurt con
    miele e mandorle" -> 3 oggetti distinti.

Per ogni alimento, restituisci un oggetto JSON con:
- "meal": nome sintetico
- "keywords": parole chiave (singolare/plurale + componenti del
    nome)
- "quantity": quantità stimata in grammi (intero)
- "matched_food_item": oggetto con "name"
- "section": nome della sezione (es. Colazione)
- "section_keywords": parole chiave legate al momento della
    giornata (es. mattina, colazione)

Rispondi solo con un array JSON valido. Nessuna spiegazione,
    nessun blocco ```json."
```

Il sistema utilizza le seguenti informazioni:

- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);
- l'**andamento del peso e delle misure corporee** nell'ultimo mese;
- i **macronutrienti seguiti nella precedente scheda alimentare** (kcal, proteine, carboidrati, grassi).

A partire da questi dati, il modello genera una scheda alimentare giornaliera strutturato in 4/6 pasti. Ogni alimento è descritto da un oggetto JSON contenente:

- **meal**: nome sintetico dell'alimento (es. `"Fesa di tacchino"`);
- **keywords**: parole chiave rilevanti, in forma singolare e plurale (ingredienti, termini descrittivi), da utilizzare nel prompt spiegato nella Sezione 6.6;
- **quantity**: quantità stimata in grammi (intero);
- **matched_food_item**: oggetto contenente il nome del corrispondente alimento di database, risultato del prompt spiegato nella Sezione 6.6;
- **section**: nome del pasto (es. `"Spuntino post workout"`);
- **section_keywords**: parole chiave associate al momento della giornata (es. `"mattina"`, `"post allenamento"`).

L'approccio adottato consente di generare schede alimentari coerenti con l'evoluzione corporea dell'utente e il suo obiettivo dichiarato, senza richiedere input dettagliati da parte del nutrizionista e personal trainer.

## 6.9 Generazione di un alimento con valori nutrizionali realistici

Il seguente prompt descrive una catena basata su modello linguistico che consente di generare automaticamente un alimento nutrizionalmente plausibile, a partire da un semplice nome fornito dal prompt di generazione della scheda alimentare.

```
"Inventa un alimento realistico con il nome "{name}" e genera i
    suoi valori nutrizionali per 100g.

Restituisci solo un JSON con i seguenti campi (nessun testo
    extra):
- "name": nome dell'alimento
- "kcal_per_100g"
- "protein_per_100g"
- "carbs_per_100g"
- "sugars_per_100g"
- "fats_per_100g"
- "saturated_fats_per_100g"
- "fiber_per_100g"

Non includere barcode o brand. Scrivi solo il JSON."
```

Il sistema riceve in input un **nome in linguaggio naturale di un alimento** (es. `"Pane di quinoa integrale"`).
Il risultato è un oggetto JSON strutturato con il nome dell'alimento (`name`) e i valori nutrizionali per 100 grammi ed è pronto per essere salvato nel database alimentare come `FoodItem` ed utilizzato per compilare schede alimentari generati dall'intelligenza artificiale.

## 6.10 Calcolo personalizzato dei macronutrienti

Il seguente prompt descrive una catena che utilizza un modello linguistico per proporre una nuova impostazione giornaliera dei macronutrienti dell'utente, basandosi sul suo obiettivo attuale e sull'evoluzione fisica osservata nel tempo.

```
"L'utente ha il seguente obiettivo: "{goal}".

Andamento del peso nell'ultimo mese:
{weights}

Andamento delle misure corporee nell'ultimo mese:
```

```
{measurements}

Nella precedente scheda alimentare ha seguito questi
    macronutrienti giornalieri:
- Proteine: {prev_protein} g
- Carboidrati: {prev_carbs} g
- Grassi: {prev_fats} g

In base a queste informazioni, proponi una nuova impostazione per
    i macronutrienti giornalieri, coerente con l'obiettivo e
    l'evoluzione fisica.

Restituisci esclusivamente un JSON con i seguenti campi:
- "max_protein": grammi di proteine consigliati (intero)
- "max_carbs": grammi di carboidrati consigliati (intero)
- "max_fats": grammi di grassi consigliati (intero)
- "reason": spiegazione dettagliata **in formato HTML**, che
    illustri il motivo delle modifiche o conferme per ogni
    macronutriente. Usa tag HTML e metti in <b>grassetto</b> le
    parole chiave (es. proteine, aumento, riduzione, ecc.)

Rispondi solo con un oggetto JSON valido. Nessun testo
    introduttivo, nessun blocco ```json."
```

Il sistema riceve in input:

- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);
- l'**andamento del peso e delle misure corporee** nell'ultimo mese;
- i **macronutrienti seguiti nella precedente scheda alimentare** (kcal, proteine, carboidrati, grassi).

A partire da questi dati, il modello fornisce i valori target per proteine, carboidrati e grassi in grammi per la nuova scheda alimentare ed una spiegazione dettagliata, che giustifica le modifiche (o conferme) apportate per ogni macronutriente.

## 6.11 Generazione di alternative bilanciate per un pasto

Il seguente prompt descrive una catena che utilizza un modello linguistico per generare un'alternativa realistica e nutrizionalmente coerente a un pasto

esistente, mantenendo l'equilibrio dei macronutrienti entro una tolleranza del $\pm 10\%$.

```
Ecco la composizione nutrizionale di un pasto della sezione
    "{section_name}".

Macro totali da rispettare (+-10%):
- Proteine: {total_protein} g
- Carboidrati: {total_carbs} g
- Grassi: {total_fats} g

Genera un'alternativa composta da 2-4 alimenti semplici.

Requisiti:
- Ogni alimento èun oggetto JSON con:
  - "meal": nome sintetico (es. "Pane integrale")
  - "quantity": quantità in grammi (es. 60)
  - "section": "{section_name}"
  - "keywords": parole chiave rilevanti per il meal, includendo:
    - le parole componenti il nome (sia singolare che plurale)
    - ingredienti o sinonimi alimentari se rilevanti

Restituisci un singolo array JSON, senza testo extra.

Esempio:
[
  {{ "meal": "Pane integrale", "quantity": 60, "section":
      "Spuntino", "keywords": ["pane", "pani", "integrale"] }},
  {{ "meal": "Tonno al naturale", "quantity": 100, "section":
      "Spuntino", "keywords": ["tonno", "tonni", "naturale"] }}
]
```

Il sistema riceve in input:

- la **sezione del pasto** (es. `Colazione`, `Spuntino`, `Cena`);
- i **valori totali di macronutrienti da rispettare**.

Sulla base di questi dati, il modello genera un'alternativa composta da 2 a 4 alimenti semplici. Ogni alimento è descritto da un oggetto JSON che include:

- **meal**: nome sintetico (es. `"Pane integrale"`);
- **quantity**: quantità suggerita in grammi (es. `60`);
- **section**: nome della sezione a cui appartiene (es. `"Spuntino"`);
- **keywords**: parole chiave legate all'alimento, in modo da essere associato ad un alimento presente all'interno del database.

Il risultato è un array JSON pronto per sostituire un pasto esistente in una scheda alimentare.
Se gli alimenti suggeriti non corrispondono ad alimenti noti, il sistema tenta di generarli dinamicamente in modo realistico, preservando coerenza nutrizionale e compatibilità con i vincoli impostati, come spiegato nella Sezione 6.9.

## 6.12 Classificazione del focus muscolare della giornata di allenamento

Il seguente prompt descrive una catena che utilizza un modello linguistico per classificare automaticamente una giornata di allenamento in base al gruppo muscolare prevalente o al pattern motorio dominante.

```
"Sei un esperto di allenamento in palestra.
Dato l'elenco degli esercizi e dettagli sulle serie, indica quale
    gruppo muscolare viene allenato maggiormente nella giornata.

Restituisci **una sola categoria breve**, come: "Petto",
    "Schiena", "Gambe", "Full Body", "Push", "Pull", "Spalle", ecc.

Dati della giornata:
{section_data}

Risposta (solo una parola o breve frase, senza punteggiatura):"
```

Il sistema riceve in input una descrizione dettagliata della giornata di allenamento, che include:

- elenco degli **esercizi eseguiti**;
- **dettagli per ogni serie** (ripetizioni, carico, tempo sotto tensione, RIR, riposo);
- **tecniche d'intensità** utilizzate.

A partire da queste informazioni, il modello produce una sola etichetta sintetica, scegliendo tra categorie comuni come *Petto*, *Schiena*, *Gambe*, *Spalle*, *Push*, *Pull*, *Full Body*, *Braccia*, ecc.

Il risultato è sempre una singola parola o breve frase, senza punteggiatura, ottimizzata per essere salvata come classificazione della sezione all'interno della scheda di allenamento.
Questa categorizzazione automatica permette di ottenere una visualizzazione rapida e intuitiva della scheda, più precisamente aiuta l'utente finale a distinguere le varie giornate.

## 6.13 Generazione di una nota tecnica per la giornata di allenamento

Il seguente prompt descrive una catena basata su modello linguistico che produce una nota sintetica e tecnica per una giornata di allenamento, spiegando la logica alla base della selezione degli esercizi proposti.

```
"Sei un esperto di allenamento. Ricevi un elenco di esercizi,
   dettagli sulle serie e sulle tecniche usate per una giornata
   di allenamento.

Scrivi una nota sintetica (massimo 300 caratteri) che spieghi
   perché sono stati scelti questi esercizi, includendo logica
   dell'allenamento (es. focus muscolare, intensità, varietà,
   controllo tecnico, recuperi).

Non usare emoji.

Dati della giornata:
{section_data}

Nota:"
```

Il sistema riceve in input un **riepilogo dettagliato della sessione** (esercizi, serie, tecniche d'intensità, ripetizioni, carichi, RIR, tempi di recupero).
Il risultato è una breve nota (massimo 300 caratteri), utile per aumentare la consapevolezza dell'utente e rafforzare l'adesione alla scheda, che riassume:

- il **focus dell'allenamento** (muscolare, tecnico, metabolico, ecc.);
- la **logica della sequenza di esercizi**;
- l'**uso di tecniche particolari** o principi specifici;
- eventuali scelte legate a varietà, stimolo o controllo motorio.

## 6.14 Generazione di una nota tecnica per la scheda settimanale di allenamento

Il seguente prompt descrive una catena basata su modello linguistico progettata per generare una nota descrittiva e professionale di una scheda settimanale di allenamento.

```
"Sei un esperto di programmazione dell'allenamento.

Dato un riepilogo dettagliato dei giorni (tipologia, tecniche
    usate, esercizi), scrivi una **breve descrizione (max 400
    caratteri)** che riassuma:
- da quanti giorni ècomposta la scheda
- che tipologie di allenamento prevede
- su cosa si focalizza (volume, intensità, tecnica, forza, ecc.)
- eventuale varietà o logica strutturata (es. split upper/lower,
    push/pull, full body alternati)
- tecniche speciali usate
- nessuna emoji

Dati della scheda:
{plan_data}

Nota:"
```

Il sistema riceve in input un riepilogo strutturato della scheda, che include per ciascun giorno:

- la **tipologia della giornata** (es. *Push*, *Pull*, *Full Body*, *Gambe*);
- l'elenco degli **esercizi svolti**, con dettagli su ripetizioni, carichi, tempi, tecniche d'intensità.

L'output è una descrizione breve (massimo 400 caratteri) che riassume:

- il **numero di giorni della scheda**;
- le **principali tipologie di allenamento**;
- il **focus predominante** (volume, intensità, tecnica, forza, ecc.);
- eventuali tecniche speciali adottate;
- la **logica strutturale del piano** (es. varietà, progressione, equilibrio dei gruppi muscolari).

La descrizione generata è utile per fornire una panoramica immediata all'utente finale della scheda di allenamento appena viene visualizzata.

## 6.15 Generazione di una nota sintetica per singolo esercizio

Il seguente prompt descrive una catena che utilizza un modello linguistico per generare una breve nota descrittiva relativa a un esercizio specifico all'interno di una scheda di allenamento.

```
"Ricevi i dettagli di un esercizio inserito in una scheda di
   allenamento.

Scrivi una nota sintetica (massimo 8 parole) che descriva
   brevemente il focus dell'esercizio (tecnica, forza, stimolo
   metabolico, controllo, velocità, esplosività, intensità,
   resistenza, volume, ecc.).

Non usare emoji.

Dati esercizio:
{item_data}

Nota:"
```

Il sistema riceve in input i **dettagli per ogni serie** (ripetizioni, carico, tempo sotto tensione, RIR, riposo).
Il risultato è una nota sintetica (massimo 8 parole) che evidenzia il focus allenante principale dell'esercizio, ad esempio:

- stimolo metabolico e alta densità;
- controllo tecnico e precisione del movimento;
- focalizzato sulla forza esplosiva.

La nota consente una comprensione rapida dell'intento programmativo di ciascun esercizio viene mostrata nell'intestazione del singolo esercizio.

## 6.16 Generazione di una scheda di allenamento settimanale

Il seguente prompt descrive una catena che utilizza un modello linguistico per generare automaticamente una scheda di allenamento settimanale, in base ai giorni di allenamento indicati dall'utente e al suo profilo fisico attuale.

```
"Sei un coach esperto. Devi creare una scheda di allenamento
    settimanale per un utente, in base ai giorni in cui si allena.

Per ciascun giorno, genera una lista di esercizi in inglese, con:
- nome esercizio
- ordine
- numero di serie (tra 3 e 6)
- tecnica d'intensità (tra: bilateral, unilateral, tempo-based)
- prescribed_reps_1 (massime ripetizioni per arto sinistro)
- prescribed_reps_2 (massime ripetizioni per arto destro)
- tempo_fcr (formato "eccentrica-pausa-concentrica", es. "3-1-2")
- rir (Reps in reserve, es. 1 o 2)
- weight (carico indicativo in kg, es. 50)
- rest_seconds (tempo di recupero in secondi, es. 90)
- notes (una breve nota in italiano** che spiega il focus
    dell'esercizio, massimo 10 parole, prima lettera maiuscola,
    niente emoji)

Le ripetizioni sono da intendersi per arto.
Se la tecnica è`"tempo-based"`, allora devi generare solo **una
    serie** (`sets: 1`) per quell'esercizio.

Restituisci il risultato in formato JSON valido, strutturato come
    nell'esempio qui sotto.
Non usare blocchi di codice come ```json o simili. Restituisci
    solo il puro JSON.

Esempio di struttura:

{{
  "lun": [
    {{
      "name": "Barbell Squat",
      "order": 1,
      "sets": 4,
      "technique": "tempo-based",
      "prescribed_reps_1": 6,
      "prescribed_reps_2": 8,
      "tempo_fcr": "3-1-1",
      "rir": 1,
      "weight": 60,
      "rest_seconds": 90
    }}
  ]
}}

Giorni selezionati: {days}"
```

Il sistema riceve in input:

- l'**obiettivo di allenamento** dell'utente (es. *fitness*, *bodybuilding*, *powerlifting*, *streetlifting*);
- l'**andamento del peso e delle misure corporee** nell'ultimo mese;
- i **giorni selezionati** dall'utente per l'allenamento.

Il risultato è un array di esercizi, ognuno corredato delle seguenti informazioni:

- **name**: nome dell'esercizio (in inglese);
- **order**: ordine nell'esecuzione del giorno;
- **sets**: numero di serie (da 3 a 6, o 1 se `tempo-based`);
- **technique**: tecnica d'intensità (`bilateral`, `unilateral`, `tempo-based`);
- **prescribed_reps_1**, **prescribed_reps_2**: ripetizioni massime per lato;
- **tempo_fcr**: tempo di esecuzione nel formato "eccentrica-pausa-concentrica" (es. `"3-1-2"`);
- **rir**: *Reps In Reserve*, margine di sforzo (es. 1 o 2);
- **weight**: carico stimato in kg;
- **rest_seconds**: tempo di recupero in secondi;
- **notes**: breve nota in italiano (max 10 parole), che descrive il focus allenante dell'esercizio.

Le caratteristiche principali del prompt di generazione di una scheda di allenamento includono il supporto per esercizi *bilaterali*, *unilaterali* e *tempo-based*, creazione automatica e popolamento della scheda all'interno del database e matching semantico del nome esercizio con quelli già presenti nel database, tramite il prompt spiegato nella Sezione 6.17.

Questo approccio consente di automatizzare completamente la creazione di schede strutturate e coerenti, personalizzare i contenuti in base all'evoluzione fisica dell'utente e garantire un bilanciamento programmato di tecniche, ripetizioni, carichi e recuperi.

## 6.17 Selezione semantica dell'esercizio più pertinente

Il seguente prompt descrive una catena basata su modello linguistico, progettata per selezionare automaticamente l'esercizio più coerente da una lista di candidati, a partire dal nome di un esercizio fornito in linguaggio naturale.

```
"Dato il nome dell'esercizio: "{input_name}"

Trova il nome più simile tra questi presenti nel database (case
    insensitive):

{db_names}

Risposta: (solo uno dei nomi indicati)"
```

Il sistema riceve in input:

- un **nome di esercizio generato** o scritto in modo non standard (es. `"Barbell Bulgarian Split Squat"`);
- la **lista dei nomi di esercizi validi** presenti nel database.

Il risultato è il nome del singolo esercizio dal database che più si avvicina semanticamente e linguisticamente a quello fornito in input, ignorando maiuscole/minuscole.
Per migliorare l'efficienza, viene eseguito un pre-filtraggio degli esercizi del database in base alle parole chiave contenute nell'input. Se non si trovano corrispondenze chiave, viene utilizzato un sottoinsieme di fallback (es. primi 15 esercizi).
Questo processo è fondamentale per evitare duplicazioni semantiche o errori di inserimento e garantire la consistenza tra input utente o AI e il database

strutturato. L'approccio è robusto anche in presenza di sinonimi, formulazioni alternative o variazioni linguistiche minori.

## 6.18 Generazione di un esercizio alternativo per la scheda di allenamento

Il seguente prompt descrive una catena basata su modello linguistico, progettata per generare automaticamente un esercizio alternativo all'interno di una scheda di allenamento settimanale, mantenendo il focus muscolare ma variando lo stimolo (attrezzo, schema motorio o tecnica).

```
"Sei un coach esperto. Ti fornirò un esercizio attuale in una
    scheda di allenamento.

Genera un esercizio alternativo che alleni gli **stessi gruppi
    muscolari**, ma in modo diverso (con attrezzo differente o
    schema diverso).
Lalternativa deve includere:
- name (nome in inglese)
- sets (numero serie)
- prescribed_reps_1 (ripetizioni minime per arto)
- prescribed_reps_2 (ripetizioni massime per arto)
- tempo_fcr (formato "eccentrica-pausa-concentrica")
- rir (Reps in reserve)
- weight (carico in kg)
- rest_seconds (recupero in secondi)
- notes (breve nota in italiano, max 10 parole, prima lettera
    maiuscola)

Se la tecnica dellesercizio attuale è"tempo-based", imposta
    sempre sets: 1.

Non usare blocchi di codice come ```json o simili. Restituisci
    solo il puro JSON.

Esercizio attuale:
- Nome: {current_name}
- Tecnica: {technique}"
```

Il sistema riceve in input:

- il **nome dell'esercizio attuale**;
- la **tecnica d'intensità utilizzata** (es. *tempo-based*, *bilateral*, *unilateral*).

Il risultato è un oggetto JSON contenente tutti i dettagli necessari per sostituire l'esercizio in modo coerente:

- **name**: nome dell'esercizio alternativo (in inglese);
- **sets**: numero di serie (1 se `tempo-based`, altrimenti 3–6);
- **prescribed_reps_1**, **prescribed_reps_2**: ripetizioni minime/massime per arto;
- **tempo_fcr**: tempo d'esecuzione (formato `"eccentrica-pausa-concentrica"`);
- **rir**: *Reps In Reserve*;
- **weight**: carico indicativo in kg;
- **rest_seconds**: recupero tra le serie in secondi;
- **notes**: breve descrizione in italiano (max 10 parole), con la prima lettera maiuscola e senza emoji.

Una volta generato, il nuovo esercizio:

1. viene confrontato semanticamente con l'elenco degli esercizi presenti nel database tramite il prompt spiegato nella Sezione 6.17;
2. viene salvato come sostituzione dell'esercizio originale all'interno del singolo esercizio;
3. viene effettuata la rigenerazione completa dei set associati con i parametri aggiornati.

Questo sistema consente una rapida sostituzione di esercizi mantenendo il target muscolare ed una maggiore varietà nelle schede di allenamento.

## 6.19 Generazione automatica delle serie di riscaldamento

Il seguente prompt descrive una catena basata su modello linguistico progettata per generare serie di riscaldamento specifiche per un esercizio principale all'interno di una scheda di allenamento.

---

```
"Sei un coach esperto. Ti fornirò i dettagli di un esercizio
   principale (con i suoi set) e devi creare una o più serie di
   **riscaldamento** per lo stesso esercizio.

Le serie di riscaldamento devono:
- usare lo stesso esercizio principale
- avere `set_number = 0`
- avere `order` incrementale da 1 in poi
- usare carichi e ripetizioni inferiori rispetto alle serie reali
- essere utili a preparare gradualmente il corpo all'esercizio
   target

Per ogni serie di riscaldamento restituisci:
- order (1, 2, 3...)
- prescribed_reps_1 e 2 (ripetizioni min/max per arto)
- tempo_fcr (formato "3-1-1")
- rir
- weight (in kg, inferiore rispetto a quello reale)
- rest_seconds

Restituisci **esattamente** un array JSON valido, **senza oggetti
   wrapper** e **senza blocchi di codice tipo ```json**.
Esempio corretto:

[
  {{
    "order": 1,
    "prescribed_reps_1": 8,
    "prescribed_reps_2": 10,
    "tempo_fcr": "2-1-2",
    "rir": 4,
    "weight": 20,
    "rest_seconds": 60
  }},
  {{
    "order": 2,
    "prescribed_reps_1": 6,
    "prescribed_reps_2": 8,
    "tempo_fcr": "2-1-2",
    "rir": 3,
    "weight": 30,
    "rest_seconds": 60
  }}
]

Esercizio principale:
- Nome: {exercise_name}
```

```
- Serie target: {series_summary}
- Peso usato: {main_weight} kg"
```

Il sistema riceve in input:

- il **nome dell'esercizio target**;
- il **riepilogo delle serie principali** (ripetizioni, carichi, tempo, ecc.);
- il **carico massimo utilizzato nelle serie principali**.

Il risultato è un array JSON contenente una o più serie di riscaldamento, ciascuna con i seguenti campi:

- **order**: posizione della serie nel riscaldamento (a partire da 1);
- **prescribed_reps_1**, **prescribed_reps_2**: ripetizioni per arto;
- **tempo_fcr**: tempo di esecuzione (`"eccentrica-pausa-concentrica"`);
- **rir**: *Reps In Reserve* (margine di sforzo);
- **weight**: carico inferiore rispetto alle serie target;
- **rest_seconds**: tempo di recupero.

Il risultato è direttamente integrabile nel database. Infatti, ogni serie viene salvata come istanza `GymPlanSetDetail`, viene mantenuto l'ordine di esecuzione (`order`) e le nuove serie non alterano la struttura principale della scheda, ma la completano con un riscaldamento intelligente.

## 6.20 Suggerimento del carico ideale basato su performance recenti

Il seguente prompt descrive una catena che utilizza un modello linguistico per suggerire in modo intelligente un peso ideale da utilizzare per un determinato esercizio, sulla base dei set recentemente eseguiti dall'utente.

```
"Sei un coach esperto. Ti fornirò una serie di set eseguiti da un
    utente per un determinato esercizio.

Set (JSON):
{sets_summary}

In base a questi dati, suggerisci un peso ideale (in kg) da usare
    oggi, coerente con l'andamento.

Restituisci solo il numero, senza testo aggiuntivo, note o
    simboli. Nessuna unità di misura. Nessun blocco di codice."
```

Il sistema riceve in input:

- una **lista di set recenti**, contenente per ciascun set:
  - `reps`: numero di ripetizioni completate;
  - `weight`: carico utilizzato (in kg);
  - `rir`: Reps In Reserve (margine di sforzo);
  - eventuali altri parametri (tempo sotto tensione, ordine, ecc.).

Il risultato è un **peso ideale**, coerente con l'andamento osservato, restituito in forma numerica pura (float), senza testo, simboli o unità di misura.
In caso di errore nel parsing del risultato o risposta non valida, il sistema restituisce `0.0` come fallback sicuro.
Questo strumento è utile per proporre un carico coerente per l'allenamento del giorno e offrire suggerimenti rapidi e contestuali all'interno della piattaforma.

# Capitolo 7

# Considerazioni finali e sviluppi futuri

Questo capitolo si propone di trarre le conclusioni emerse durante lo sviluppo di *SmartFit Coach*, riflettendo in particolare sui risultati ottenuti in termini di funzionalità, usabilità e capacità di personalizzazione. Dopo aver attraversato tutte le fasi del ciclo di vita del progetto — dall'analisi dei requisiti, alla progettazione dell'architettura dati, allo sviluppo dell'interfaccia e all'integrazione dell'intelligenza artificiale — è possibile valutarne l'impatto complessivo, non solo dal punto di vista tecnico, ma anche rispetto all'esperienza utente e agli obiettivi prefissati all'inizio del tirocinio.
Allo stesso tempo, lo spazio per ulteriori miglioramenti rimane ampio: l'architettura modulare e la natura flessibile dell'applicativo offrono numerose possibilità di estensione, sia in termini di funzionalità, che di precisione delle analisi intelligenti. Le considerazioni che seguono intendono quindi fornire uno sguardo critico e propositivo su quanto è stato realizzato e su ciò che potrebbe essere sviluppato in futuro.

## 7.1 Stato attuale dell'applicativo

*SmartFit Coach* è attualmente disponibile in forma di **web app**, accessibile tramite browser sia da dispositivi desktop che mobile. Grazie a un'interfaccia

sviluppata con tecnologie moderne e responsive, l'esperienza d'uso risulta fluida e coerente su diverse risoluzioni e sistemi operativi, rendendo l'applicativo adatto a un utilizzo quotidiano, anche in mobilità.
A livello funzionale, sono state implementate tutte le **funzionalità principali previste nella fase iniziale di progettazione**, come la gestione dei dati biometrici, la creazione delle schede personalizzate e l'integrazione con l'intelligenza artificiale. A queste si sono aggiunte, o sono state confermate come prioritarie, anche **funzionalità espresse direttamente da personal trainer e nutrizionisti**, coinvolti nella fase di raccolta dei feedback. Tra le funzionalità attualmente operative rientrano:

- la gestione del profilo utente e dei dati biometrici (peso, misure corporee);
- la creazione e modifica di schede alimentari personalizzate;
- la creazione e modifica di schede di allenamento personalizzate, con supporto alle tecniche d'intensità;
- il supporto intelligente tramite **intelligenza artificiale**, che consente di:
  - offrire un'**analisi intelligente dei dati raccolti**, come peso e misure corporee, restituendo feedback utili e personalizzati sull'andamento fisico;
  - generare **schede alimentari e di allenamento personalizzate** in base agli obiettivi selezionati e ai dati biometrici inseriti dall'utente;
  - **creare elementi alimentari a partire da input testuali** o tramite il **riconoscimento di immagini** (es. foto di un pasto);
  - proporre **alternative intelligenti a esercizi o alimenti già presenti nella scheda**, mantenendo coerenza con i fabbisogni e le preferenze utente.

## 7.2 Potenzialità evolutive e nuove piattaforme: Flutter

Una delle direzioni di sviluppo più promettenti per il futuro di *SmartFit Coach* riguarda la creazione di un'applicazione nativa multipiattaforma tramite **Flutter**, il framework open-source di Google per lo sviluppo mobile, oppure **React Native**. Attualmente, l'app è accessibile via browser come web app responsive, ma considerando che il mondo dell'informatica è ormai fortemente orientato verso il **mobile-first**, si apre l'opportunità di offrire una **versione nativa per Android e iOS**.
L'adozione di Flutter consentirebbe diversi vantaggi significativi:

- una **migliore integrazione con l'hardware** del dispositivo (sensori, fotocamera, accelerometro, lettura dei dati biometrici, ecc.);
- la possibilità di implementare **notifiche push** per ricordare pasti, allenamenti o check-in dei dati biometrici programmati;
- il supporto a funzionalità **offline**, per consultare le schede e inserire dati anche in assenza di connessione;
- una **user experience ancora più fluida e reattiva**, sfruttando le prestazioni delle app native rispetto al rendering via browser.

## 7.3 Estensioni funzionali del sistema

Oltre al consolidamento delle funzionalità esistenti, *SmartFit Coach* si presta naturalmente all'introduzione di nuove componenti che ne amplierebbero il valore d'uso, sia per l'utente finale che per i professionisti del settore. Le seguenti proposte rappresentano alcune delle direzioni evolutive più rilevanti dal punto di vista funzionale:

- **supporto al coaching a distanza**: l'integrazione di un sistema di **chat interna** consentirebbe a nutrizionisti e personal trainer di offrire

sessioni dirette all'interno della piattaforma, riducendo la necessità di strumenti esterni e migliorando la continuità tra comunicazione, feedback e aggiornamento delle schede;

- **gamification**: l'introduzione di meccaniche ludiche, come badge, livelli di progressione, sfide settimanali o obiettivi a tempo, potrebbe aumentare la motivazione e l'aderenza al percorso, trasformando l'esperienza di benessere in un processo più coinvolgente;

- **community di utenti**: la creazione di uno spazio sociale in cui gli utenti possano condividere risultati, confrontarsi, supportarsi o seguire i progressi di altri, favorirebbe il senso di appartenenza e la motivazione reciproca;

- **diario vocale**: offrire la possibilità di registrare note vocali giornaliere legate ad pasti, allenamenti o stati d'animo rappresenterebbe un modo naturale per favorire la riflessione, migliorare la consapevolezza e raccogliere dati qualitativi utili per l'intelligenza artificiale. Ovviamente, l'utilizzo di queste note vocali possono servire ad integrare voci nelle varie schede tramite l'intelligenza artificiale.

Queste estensioni, tutte perfettamente integrabili all'interno dell'architettura attuale, mirano a trasformare *SmartFit Coach* in una piattaforma completa per il benessere personalizzato, sia fisico che relazionale. Con l'introduzione di tali funzionalità, il prodotto diventerebbe significativamente più **appetibile sia per gli utenti finali**, che troverebbero un'esperienza più coinvolgente e motivante, **sia per i professionisti del settore**, che disporrebbero di strumenti evoluti per l'interazione, la fidelizzazione e il monitoraggio dei propri clienti.

## 7.4 Sviluppo di un modello proprietario basato su deep learning

Un'evoluzione strategica particolarmente interessante per *SmartFit Coach* riguarda la possibilità di sviluppare, in futuro, un **modello di deep learning proprietario** addestrato direttamente sui dati della community di utenti, ovviamente con le dovute autorizzazioni legali da parte dell'utente. Attualmente, l'intelligenza artificiale integrata nella piattaforma si basa su **chiamate esterne alle API di OpenAI**, orchestrate tramite **LangChain**, per l'analisi degli obiettivi, la generazione automatica di schede e la proposta di alternative personalizzate.
Sebbene questa architettura abbia garantito flessibilità e rapidità nello sviluppo, l'adozione di un modello interno offrirebbe diversi vantaggi a lungo termine, tra cui:

- **riduzione dei costi** legati al consumo di API esterne;
- **maggiore controllo sui dati** e sulle logiche decisionali del modello;
- possibilità di un **fine-tuning continuo** basato sul comportamento reale degli utenti;
- **ottimizzazione mirata** per il dominio specifico del fitness e della nutrizione.

L'idea alla base sarebbe quella di raccogliere, in forma anonima e aggregata, i dati generati e confermati dagli utenti (schede alimentari, schede di allenamento, risposte accettate, preferenze, modifiche ricorrenti, andamenti degli obiettivi) per costruire un dataset reale e contestualizzato. A partire da questo dataset, si potrebbe procedere con la progettazione e addestramento di un **modello transformer** o simile, ottimizzato per generare output in linguaggio naturale e strutturato, replicando (e potenzialmente migliorando) le prestazioni dei modelli generalisti oggi utilizzati.

In questa visione, le chiamate a OpenAI verrebbero gradualmente sostituite da inferenze locali o tramite un server dedicato.
L'implementazione di questo modello rappresenterebbe un salto qualitativo importante, trasformando *SmartFit Coach* da piattaforma integrata con servizi AI a **sistema AI nativamente intelligente**, con capacità di apprendimento e adattamento progressivo all'interno della propria base utenti.