POLITECNICO DI TORINO

Dipartimento di Ingegneria Gestionale e della Produzione

Corso di Laurea Triennale in Ingegneria Gestionale
Classe L-8 Ingegneria dell'Informazione

Prova Finale

Assistente alimentare: *tool* per la generazione di una lista della spesa ottimizzata

Relatore:
Professor Giuseppe Bruno Averta

Candidato:
Gabriele Adorni
Matricola: s297466

Marzo 2025

Indice

1 Proposta di progetto

1.1 Titolo della proposta

Assistente alimentare: generatore di lista della spesa

1.2 Descrizione del problema proposto

Il progetto affronta la problematica della creazione di un'applicazione in grado di generare una lista di prodotti alimentari il cui apporto complessivo di nutrienti e calorie soddisfi il fabbisogno giornaliero dell'individuo, tenendo conto anche delle preferenze personali. L'obiettivo dell'applicazione è sollevare l'utente dall'onere di dover ideare quotidianamente una lista della spesa bilanciata: l'*App* provvederà a generarla, valutando lo stato fisico, gli obiettivi nutrizionali e le preferenze alimentari dell'utente.
Per fare ciò l'utente dovrà fornire alcune informazioni biometriche e rispondere a domande relative al proprio stile di vita; da questi dati verranno calcolati i parametri necessari per ottimizzare la lista della spesa, che potrà essere in seguito modificata a condizione che il bilanciamento nutrizionale complessivo venga mantenuto.

1.3 Descrizione della rilevanza gestionale del problema

In un'epoca in cui la correlazione tra nutrizione e salute è ormai consolidata, l'applicazione si propone come uno strumento utile per semplificare la gestione di una corretta alimentazione. Essa funge da assistente alimentare, consentendo all'utente di strutturare la propria lista della spesa in conseguenza della propria dieta, seguendo solide linee guida che rappresentano le fondamenta e i vincoli degli algoritmi impiegati per generare l'*output* finale. Tale approccio non solo mira a garantire un risparmio di tempo nella routine quotidiana, ma contribuisce anche a migliorare lo stile di vita e la salute, nonché a consolidare consapevolezze in campo alimentare.

1.4 Descrizione del *dataset* per la valutazione

La scelta del *dataset* per la valutazione non è stata semplice. Durante la ricerca su siti web che offrono *dataset* open-source (ad esempio, Kaggle) si è riscontrata una maggiore presenza di *dataset* di stampo commerciale, contenenti prodotti alimentari provenienti da supermercati e aziende, corredati da informazioni non strettamente utili all'applicazione (come prezzi di vendita, percentuali di sconto, quantità vendute...). Tra le opzioni orientate all'aspetto nutrizionale, è stato selezionato il seguente (https://www.kaggle.com/datasets/utsavdey1410/food-nutrition-dataset).
Successivamente, è stato necessario effettuare alcune operazioni di *data management* per conformare il *dataset* alle specifiche richieste dall'applicazione. In particolare, le cinque tabelle (FOOD DATA GROUP.csv) contenute nel *dataset* sono state unite, aggiungendo una colonna con una chiave univoca per identificare ciascun prodotto, e rimuovendo alcune colonne ritenute superflue. Inoltre, per aggiungere un ulteriore livello di personalizzazione per l'utente, è stata integrata una tabella aggiuntiva nel database contenente la chiave univoca del prodotto, il relativo nome e numerose colonne con valori booleani che indicano l'aderenza del prodotto a determinati requisiti, sia relativi a preferenze dietetiche (ad es. dieta vegana/non vegana) sia a condizioni mediche (ad es. presenza o assenza di lattosio). Durante le fasi di sviluppo e test dell'applicazione sono state inoltre eliminate alcune classi di prodotti non conformi alle fondamenta etiche del progetto, come quelli provenienti da *fast-food* o relativi ad alcolici.

1.5 Descrizione preliminare degli algoritmi coinvolti

Calcolo del fabbisogno calorico e nutrizionale
L'algoritmo prenderà in *input* i dati personali dell'utente, quali peso, altezza, età, sesso e li-

vello di attività fisica. Sulla base di queste informazioni verrà utilizzata una formula standard, ad esempio l'equazione di Harris-Benedict o quella di Mifflin-St Jeor, per calcolare il TDEE (*Total Daily Energy Expenditure*), ovvero il fabbisogno calorico giornaliero. Successivamente il fabbisogno calorico verrà suddiviso tra i macronutrienti (proteine, carboidrati e grassi) secondo proporzioni personalizzabili o basate su raccomandazioni standard (ad es. 30% proteine, 50% carboidrati, 20% grassi). In corso d'opera è stato aggiunto il calcolo dell'apporto di fibre e di alcuni micronutrienti.

Ottimizzazione lineare
Utilizzando il TDEE e i fabbisogni giornalieri di nutrienti precedentemente calcolati, l'algoritmo genererà una lista della spesa che soddisfi le esigenze nutrizionali dell'utente. Tale processo verrà strutturato come un problema di programmazione lineare, con una funzione obiettivo che minimizza la differenza tra il fabbisogno nutrizionale e il contributo degli alimenti. Durante la fase di sviluppo sono state esplorate diverse soluzioni per la ricerca di combinazioni di prodotti ottimali o subottimali, inclusa una strategia basata su una funzione ricorsiva *greedy*, arricchita da euristiche per semplificare e velocizzare il calcolo. Tuttavia l'approccio basato sulla programmazione lineare si è rivelato il più efficace, consentendo quest'ultimo di ottenere soluzioni subottimali molto vicine all'ottimo in tempi ragionevoli.

1.6 Descrizione preliminare delle funzionalità previste per l'applicazione *software*

L'applicazione sarà progettata con un'interfaccia utente semplice e intuitiva, offrendo le seguenti funzionalità:

1. **Inserimento delle informazioni personali**
   L'utente potrà fornire dettagli quali peso, altezza, sesso, età e livello di attività fisica. Una volta inseriti correttamente, questi dati permetteranno di calcolare il TDEE (fabbisogno calorico giornaliero), che costituirà la base per la strutturazione di un piano alimentare. Per chi desidera una maggiore personalizzazione, sarà possibile aumentare o diminuire il valore calorico giornaliero in incrementi di 100 calorie, così da generare una lista coerente con una dieta iper o ipocalorica.

2. **Preferenze alimentari**
   Prima della generazione della lista, l'utente potrà personalizzare l'*output* attivando degli "switch" per includere o escludere determinate categorie di prodotti nell'ottimizzazione. Tali categorie saranno legate sia a preferenze dietetiche (ad es. dieta vegana) sia a condizioni mediche (ad es. presenza o assenza di lattosio nei prodotti).

3. **Generazione della lista**
   Attraverso un apposito pulsante, l'utente potrà avviare la generazione della lista della spesa, la quale verrà visualizzata a schermo accompagnata da un istogramma che evidenzia le prestazioni dell'ottimizzazione. Per ciascun nutriente verrà mostrata una colonna che rappresenta il *target* e una colonna che indica il totale fornito dagli alimenti, con l'obiettivo di mantenere l'equilibrio.

4. **Ulteriore personalizzazione dell'output**
   Dopo la generazione dell'*output*, l'utente potrà personalizzare ulteriormente la propria esperienza accedendo a una sezione dedicata. In questa sezione sarà possibile effettuare un "refresh" completo dell'*output* oppure sostituire alcuni prodotti indesiderati con altri aventi un profilo nutrizionale simile, in modo da non compromettere il bilanciamento complessivo.

5. **Sezione informativa/carenze nutrizionali**
   L'ultima sezione dello strumento consentirà all'utente di colmare eventuali carenze di alcuni micronutrienti. Selezionando un nutriente (tra quelli non ottimizzati per evitare

un'eccessiva complessità computazionale), l'utente potrà accedere a una descrizione degli effetti di quel nutriente e alle motivazioni per cui è importante assumerlo in una determinata quantità, oltre a ricevere una lista di prodotti che lo contengono, ordinata in ordine decrescente in base alla quantità presente.

2 Descrizione dettagliata del problema affrontato

2.1 Contesto di utilizzo dell'applicazione

Per l'individuo adulto medio, l'attività del "fare la spesa" rappresenta un compito quotidiano imprescindibile, che richiede una pianificazione accurata e l'impiego di tempo ed energie. Spesso, tuttavia, tale impegno non porta a risultati ottimali, a causa di una carenza di competenze specifiche o di altre difficoltà organizzative.

L'applicazione si propone di generare automaticamente una lista della spesa ottimizzata, in grado di assicurare un apporto nutrizionale bilanciato e perfettamente in linea con le esigenze individuali. In particolare, il sistema raccoglie dati biometrici e informazioni sulle preferenze alimentari dell'utente per elaborare un output completamente personalizzato. Se progettato e implementato correttamente, uno strumento di questo tipo potrà apportare notevoli benefici, non solo in termini di comodità, ma soprattutto per il miglioramento della salute di chi lo utilizza.

2.2 Come si produce una dieta bilanciata

La creazione di una dieta equilibrata parte dalla corretta valutazione del fabbisogno energetico individuale, definito attraverso il calcolo del metabolismo basale (BMR) e del fabbisogno calorico totale giornaliero (TDEE). È fondamentale semplificare il concetto di "alimentazione", considerando che ogni alimento è caratterizzato da due aspetti essenziali: un apporto energetico (solitamente espresso in kilocalorie) e un apporto nutrizionale, ossia la composizione in nutrienti quali proteine, carboidrati, grassi e micronutrienti (vitamine e minerali). Questi elementi, analizzati in base alle etichette nutrizionali (tipicamente per 100 g di prodotto), costituiscono la base per determinare la combinazione ottimale di alimenti che soddisfi il fabbisogno individuale.

2.2.1 Composizione degli alimenti

I prodotti alimentari si caratterizzano per la presenza di nutrienti essenziali, suddivisi in due categorie principali: macronutrienti e micronutrienti. I macronutrienti comprendono carboidrati, proteine e grassi, che forniscono l'energia necessaria per il funzionamento quotidiano dell'organismo. I micronutrienti, invece, includono vitamine e minerali, fondamentali per il corretto svolgimento delle funzioni fisiologiche, pur essendo necessari in quantità minori.

Le informazioni nutrizionali riportate sulle etichette degli alimenti, solitamente espresse per 100 g di prodotto, forniscono una panoramica dettagliata del contenuto di ciascun nutriente. Tali tabelle permettono di parametrare in maniera oggettiva il contributo di ogni alimento al fabbisogno giornaliero consigliato.

Il sistema utilizza questi dati nutrizionali per selezionare una combinazione di alimenti che, complessivamente, raggiunga i livelli raccomandati di nutrienti. In altre parole, l'obiettivo è quello di combinare prodotti con differenti profili nutrizionali in modo che il totale assorbito soddisfi sia i requisiti energetici (forniti dai macronutrienti) sia il fabbisogno di micronutrienti indispensabili per la salute. Questo approccio permette di pianificare diete equilibrate, che contribuiscono al benessere generale e alla prevenzione di carenze nutrizionali.

2.2.2 TDEE e BMR

Il metabolismo basale (BMR) rappresenta l'energia minima necessaria per mantenere in funzione le attività vitali a riposo (ad esempio, respirazione, circolazione sanguigna e mantenimento della temperatura corporea). Per calcolare il BMR si utilizzano formule che richiedono input quali peso, altezza, età e sesso. Tra le formule più note troviamo:

- **Formula di Harris-Benedict (versione originale):**
  - Per gli uomini:

$$\text{BMR} = 66 + (13,7 \times \text{peso [kg]}) + (5 \times \text{altezza [cm]}) - (6,8 \times \text{età [anni]})$$

  - Per le donne:

$$\text{BMR} = 655 + (9,6 \times \text{peso [kg]}) + (1,8 \times \text{altezza [cm]}) - (4,7 \times \text{età [anni]})$$

- **Formula di Mifflin-St Jeor (considerata più accurata):**
  - Per gli uomini:

$$\text{BMR} = (10 \times \text{peso [kg]}) + (6,25 \times \text{altezza [cm]}) - (5 \times \text{età [anni]}) + 5$$

  - Per le donne:

$$\text{BMR} = (10 \times \text{peso [kg]}) + (6,25 \times \text{altezza [cm]}) - (5 \times \text{età [anni]}) - 161$$

Il *Total Daily Energy Expenditure* (TDEE) rappresenta il fabbisogno calorico totale giornaliero, includendo sia l'energia necessaria per le funzioni vitali (BMR) sia quella spesa nelle attività quotidiane. Per calcolare il TDEE, il BMR viene moltiplicato per un coefficiente di attività fisica (AF) che varia in base al livello di attività dell'individuo. I valori tipici del coefficiente di attività sono:

- Sedentario: AF ≈ 1.2
- Leggermente attivo: AF ≈ 1.375
- Moderatamente attivo: AF ≈ 1.55
- Molto attivo: AF ≈ 1.725
- Estremamente attivo: AF ≈ 1.9

La formula finale è quindi:

$$\text{TDEE} = \text{BMR} \times \text{AF}$$

Il TDEE non solo rappresenta il quantitativo di energia necessario per mantenere il peso corporeo stabile, ma funge anche da parametro di riferimento per raggiungere specifici obiettivi di variazione del peso. Infatti se l'apporto calorico giornaliero supera il TDEE, l'eccesso energetico viene immagazzinato (generalmente sotto forma di grasso), portando ad un aumento di peso. Al contrario, un apporto inferiore al TDEE genera un deficit calorico che, se prolungato, si traduce in una perdita di peso. Questi calcoli, basati sui dati biometrici e sul livello di attività fisica dell'utente, forniscono il valore energetico totale su cui si fonda la pianificazione nutrizionale.

2.2.3 Calcolo dei fabbisogni nutrizionali per i singoli nutrienti

Per ottenere una dieta bilanciata è essenziale definire con precisione il quantitativo di ciascun nutriente da assumere giornalmente. L'approccio adottato si basa sulla suddivisione del fabbisogno energetico totale (TDEE) e su parametri individuali (peso, sesso, età) per determinare i requisiti di macronutrienti e micronutrienti, seguendo linee guida nutrizionali scientificamente validate.

- **Macronutrienti:**
  - **Grassi:** si prevede che circa il 30% delle calorie totali debba provenire dai grassi. Poiché ogni grammo di grassi fornisce circa 9 kcal, il quantitativo in grammi si ottiene moltiplicando il TDEE per 0.3 e dividendolo per 9.
  - **Grassi Saturi:** viene assegnato il 10% del TDEE a questo sotto-gruppo, anch'esso convertito in grammi dividendo per 9.
  - **Carboidrati:** con il 50% del TDEE destinato ai carboidrati e considerando che ogni grammo fornisce 4 kcal, il fabbisogno in grammi si calcola moltiplicando il TDEE per 0.5 e dividendolo per 4.
  - **Proteine:** il fabbisogno proteico viene determinato in base al peso corporeo, con coefficienti differenziati in base al sesso (0.8 g/kg per le donne e 1.0 g/kg per gli uomini) per riflettere le diverse esigenze metaboliche.
- **Micronutrienti e altri componenti:**
  - **Fibra:** vengono assegnati valori fissi (25 g per le donne e 30 g per gli uomini) che supportano la regolarità intestinale e il benessere generale.
  - **Sodio:** il quantitativo raccomandato varia in funzione dell'età (1500 mg per individui sotto i 50 anni e 1300 mg per quelli oltre), in linea con le raccomandazioni per il controllo della pressione arteriosa.
  - **Vitamina C:** la dose giornaliera è fissata a 75 mg per le donne e 90 mg per gli uomini, in base alle necessità antiossidanti e al supporto del sistema immunitario.
  - **Vitamina D:** viene stabilito un fabbisogno di 15 microgrammi, fondamentale per il metabolismo del calcio e la salute delle ossa.
  - **Calcio:** un apporto standard di 1000 mg al giorno è considerato adeguato per il mantenimento della densità ossea.
  - **Ferro:** le esigenze di ferro sono maggiori nelle donne (18 mg) rispetto agli uomini (8 mg), in considerazione delle perdite ematiche e delle necessità ematologiche.
  - **Potassio:** i valori raccomandati sono 3400 mg per gli uomini e 2600 mg per le donne, essenziali per il corretto funzionamento muscolare e la regolazione della pressione arteriosa.

L'approccio matematico, basato su proporzioni di energia e pesi specifici, permette di tradurre in valori quantitativi raccomandati le linee guida nutrizionali internazionali, garantendo un *output* personalizzato e scientificamente fondato. La suddivisione scelta consente di rispettare sia il fabbisogno energetico (espresso in kcal) sia quello nutrizionale (composto dai singoli nutrienti), offrendo così una base solida per la generazione di una lista della spesa che soddisfi in modo ottimale le esigenze dell'utente.

2.3 Funzionamento pratico dell'assistente alimentare

L'assistente alimentare è strutturato in quattro sezioni operative che guidano l'utente dalla definizione dei propri dati biometrici fino alla personalizzazione finale della lista della spesa, garantendo un processo intuitivo e flessibile.

- **Prima sezione.** L'utente inserisce i propri dati biometrici, fondamentali per il calcolo del TDEE. Una volta validati, il sistema calcola il fabbisogno calorico in kilocalorie. L'utente può poi regolare questo valore, incrementandolo o decrementandolo di 100 kcal per volta, fino a raggiungere un apporto calorico giornaliero coerente con i propri obiettivi. È altresì possibile filtrare gli elementi del *database* per escludere prodotti indesiderati o non conformi alla propria dieta, sia per scelte personali (ad es., dieta vegana) sia per particolari esigenze (ad es. intolleranze al lattosio).

- **Seconda sezione.** In questa fase l'utente avvia la ricerca di una combinazione di alimenti che soddisfi il proprio fabbisogno calorico. Il sistema genera un *output* visualizzato come una lista in cui per ciascun alimento sono indicati il nome e le porzioni suggerite. Se l'*output* non risponde alle aspettative, è possibile effettuare un "refresh" per rigenerare una nuova proposta. Contestualmente viene presentato un grafico (istogramma) che confronta il *target* nutrizionale con il totale dei nutrienti forniti dalla lista, offrendo così una valutazione immediata dell'efficacia dell'ottimizzazione.

- **Terza sezione.** L'utente accede a una finestra contenente l'*output* generato, con la possibilità di personalizzare la lista. In questa fase è possibile sostituire uno o più prodotti, purché il nuovo alimento proposto sia nutrizionalmente simile a quello rimosso, per mantenere l'equilibrio della dieta. Una volta apportate le modifiche desiderate, l'utente le conferma, aggiornando l'*output* sia nella lista che nel grafico, che verranno riproposti nella sezione precedente.

- **Quarta sezione.** Questa parte, concepita principalmente a scopo informativo, fornisce gli strumenti per colmare eventuali carenze nutrizionali già note all'utente. Selezionando un micronutriente dalla lista disponibile, il sistema mostra la lista degli alimenti presenti nel *database* che lo contengono, specificandone anche la quantità, accompagnata da una breve spiegazione della funzione principale del nutriente. Questo consente all'utente di approfondire il proprio profilo nutrizionale e di integrare la dieta in modo mirato.

2.4 Criticità

Tra le principali criticità si evidenzia l'importanza di disporre di un *dataset* di alta qualità. La correttezza e l'accuratezza dei dati nutrizionali e dei parametri biometrici sono fondamentali per garantire un *output* valido e affidabile. Errori o imprecisioni nei dati possono compromettere il calcolo del TDEE e la suddivisione dei fabbisogni nutrizionali, influendo negativamente sull'ottimizzazione finale. Un'altra criticità riguarda la complessità nell'integrare preferenze, restrizioni e variabili individuali in un modello matematico robusto e allo stesso tempo flessibile.

2.5 Potenzialità e rilevanza

Il *tool* presenta numerose potenzialità, sia per l'utente singolo sia per le aziende:

- **Salute e benessere:** offrendo un supporto per la pianificazione di diete equilibrate, lo strumento contribuisce al miglioramento della salute degli utenti e alla prevenzione di carenze nutrizionali.

- **Informazione e cultura alimentare:** la possibilità di visualizzare e comprendere il bilanciamento nutrizionale dei pasti favorisce una maggiore consapevolezza e informazione in tema di alimentazione sana.

- **Innovazione e competitività:** per le piattaforme di spesa online, l'integrazione di un sistema di generazione automatica della lista della spesa può rappresentare un vantaggio competitivo, migliorando l'esperienza utente e differenziando l'offerta nel mercato.

In sintesi, il *tool* intende configurarsi come uno strumento versatile e innovativo, capace di rispondere a esigenze diverse e di avere un impatto positivo sia a livello individuale che aziendale, favorendo una gestione più consapevole e personalizzata dell'alimentazione.

3 Descrizione del *dataset* utilizzato per l'analisi

Il *dataset* rappresenta un elemento fondamentale per il funzionamento dell'applicazione, in quanto fornisce, attraverso *query* mirate, informazioni dettagliate sui prodotti alimentari. Queste informazioni vengono successivamente convertite in oggetti della classe *Food*, costituendo la base per la generazione automatica di una lista della spesa ottimizzata. In questo contesto, il *dataset* non solo costituisce la fonte primaria dei dati nutrizionali, ma svolge anche un ruolo determinante nel garantire la qualità e la coerenza dei risultati ottenuti dall'applicazione.

3.1 Ricerca del *dataset*

Il processo di ricerca del *dataset* ha avuto inizio con l'esplorazione di diverse piattaforme *online*, tra cui Kaggle e altre risorse similari, dove erano disponibili numerosi *dataset* relativi ai prodotti alimentari. Molti di questi presentavano una struttura monolitica, composta da una singola tabella contenente informazioni quali: nome del cibo, peso, calorie, proteine, ecc.. Tuttavia, i requisiti specifici del progetto imponevano la presenza di ulteriori elementi, quali un valore di porzione tipica e categorie esplicative (ad es. etichette per identificare latticini, prodotti vegani, ecc.), oltre ai dettagli sui nutrienti e micronutrienti.

Nessuno dei *dataset* disponibili rispondeva pienamente a tali requisiti; pertanto, è stato scelto un *dataset* che, pur non essendo perfetto, soddisfaceva i requisiti fondamentali. Successivamente è stato sottoposto a processi di *data management* e *data cleaning* per adattarlo alle specifiche esigenze dell'applicazione. Le modifiche apportate saranno illustrate in dettaglio nelle sezioni successive.

3.2 Descrizione del *dataset* scelto

Il *dataset* originario è disponibile al seguente link:

`https://www.kaggle.com/datasets/utsavdey1410/food-nutrition-dataset`

Il materiale, in lingua inglese, comprende cinque file `.csv` contenenti informazioni nutrizionali relative a diversi alimenti.

3.2.1 Modifiche effettuate al *dataset*

I file `.csv` sono stati uniti e rielaborati tramite la libreria *pandas*, con l'obiettivo di:

- Creare un'unica tabella che riunisse i dati provenienti dalle cinque sorgenti.
- Assegnare una chiave primaria (`ID`) a ciascun alimento, utile per l'identificazione univoca all'interno del database.
- Rimuovere le colonne superflue, non rilevanti per l'ottimizzazione o la generazione della lista della spesa.

Dopo questi interventi di *data cleaning* e *data management*, la tabella risultante (denominata, ad esempio, `food_nutrients`) si presenta con i seguenti attributi principali:

- **ID**: identificatore numerico univoco (*primary key*).
- **Food**: nome dell'alimento.

- **Caloric Value**: energia totale fornita dall'alimento, in kcal per 100 g.
- **Fat (in g)**: quantità totale di grassi in grammi per 100 g, comprensiva di sottocategorie.
- **Saturated Fats (in g)**: quantità di grassi saturi per 100 g.
- **Monounsaturated Fats (in g)**: quantità di grassi monoinsaturi per 100 g.
- **Polyunsaturated Fats (in g)**: quantità di grassi polinsaturi per 100 g.
- **Carbohydrates (in g)**: totale dei carboidrati in grammi per 100 g.
- **Sugars (in g)**: zuccheri semplici in grammi per 100 g.
- **Protein (in g)**: contenuto proteico per 100 g.
- **Dietary Fiber (in g)**: contenuto di fibre alimentari per 100 g.
- **Cholesterol (in mg)**: colesterolo per 100 g.
- **Sodium (in mg)**: sodio per 100 g.
- **Water (in g)**: acqua in grammi per 100 g.
- **Vitamin A, B1, B2, . . . , B12, C, D, E, K (in mg)**: vitamine essenziali.
- **Calcium, Iron, Magnesium, . . . , Zinc (in mg)**: minerali essenziali.
- **Nutrition Density**: indice di densità nutrizionale per caloria.

3.2.2 Realizzazione della tabella preferenze

Durante la fase di sviluppo del progetto, è emersa l'esigenza di implementare un sistema di filtraggio dei cibi in base alle preferenze dell'utente (ad es. alimenti vegani, vegetariani, senza lattosio, ecc.). Per questo motivo, è stata creata una tabella `food_preferences`, definita manualmente, con la seguente struttura:

- **ID**: identificatore univoco (*primary key*).
- **Food**: nome dell'alimento.
- **Vegano**: valore 1 se l'alimento non contiene ingredienti di origine animale, 0 altrimenti.
- **Vegetariano**: valore 1 se l'alimento non contiene carne, pesce o pollame (ma può includere latte e uova), 0 altrimenti.
- **Lattosio**: valore 1 se l'alimento contiene derivati del latte, 0 altrimenti.
- **Glutine**: valore 1 se l'alimento contiene cereali contenenti glutine, 0 altrimenti.
- **Arachidi**: valore 1 se l'alimento contiene arachidi, 0 altrimenti.
- **Frutta secca**: valore 1 se l'alimento include noci, mandorle o altri semi oleosi, 0 altrimenti.

Grazie a questa tabella, il sistema può escludere automaticamente dall'ottimizzazione gli alimenti che non soddisfano i requisiti dell'utente.

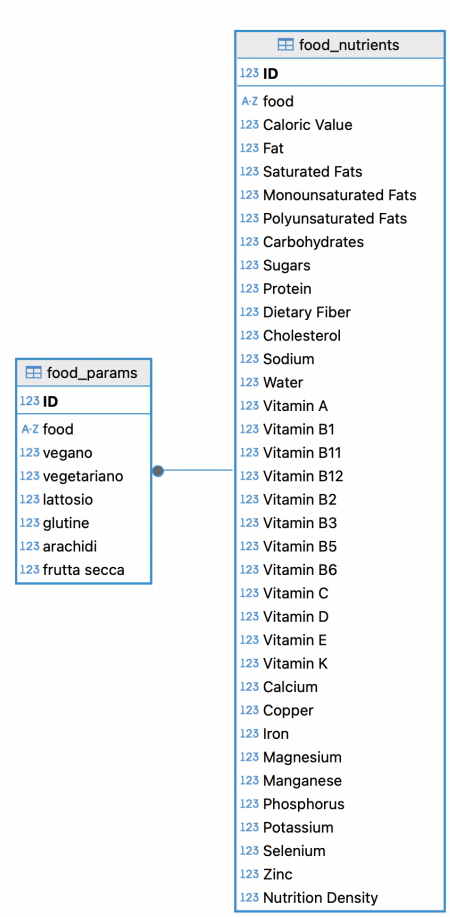

Figura 1: Diagramma ER del *database*

3.3 Criticità

Inizialmente, si pensava che il *dataset* potesse essere facilmente reperito; tuttavia, già in fase di ricerca è emerso che nessuno dei *dataset* disponibili soddisfaceva integralmente tutti i requisiti necessari. In particolare, non esisteva un dataset dotato di informazioni complete, comprensive di dettagli sui nutrienti, valori di porzione tipica e categorizzazioni esplicative (ad es. etichette per prodotti vegani, latticini, ecc.). Di conseguenza è stato necessario selezionare un *dataset* accettabile e adattarlo mediante processi di *data cleaning* e *data management*. Tale esigenza di intervento rappresenta una criticità rilevante: se in futuro si volesse implementare questo *tool* in un contesto commerciale, sarebbe indispensabile sviluppare un *dataset ad hoc*, utilizzando risorse e mezzi significativi per garantire la completezza, l'accuratezza e l'affidabilità dei dati.

4 Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati

4.1 Struttura del Progetto

Il progetto è realizzato in Python e segue l'architettura **MVC (Model-View-Controller)**, che consente di organizzare il codice suddividendo le responsabilità in moduli specifici, facilitando la manutenzione e l'estensione delle funzionalità.

Package Model: Gestione dei Dati e Logica di Business

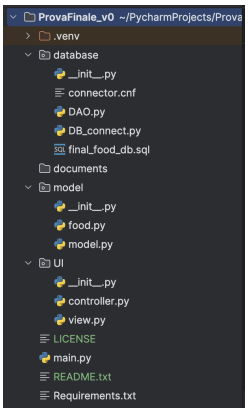

Figura 2: Struttura delle cartelle del progetto

- **model.py:** implementa le funzioni per il calcolo del Basal Metabolic Rate (BMR) mediante la formula di Mifflin-St Jeor e per il calcolo del Total Daily Energy Expenditure (TDEE) in base al livello di attività. Inoltre, fornisce metodi per il recupero e il filtraggio dei dati alimentari e per il calcolo dei fabbisogni nutrizionali, definendo un benchmark personalizzato.
- **food.py:** definisce la classe `Food`, che incapsula le informazioni relative ad un prodotto alimentare (ID, nome, e valori nutrizionali quali calorie, grassi, proteine, carboidrati, fibre, micronutrienti, ecc.), utilizzando il tipo `decimal.Decimal` per garantire la precisione.

Package UI: Interfaccia Utente e Controllo

- **view.py:**

  Il file `view.py` rappresenta la componente di presentazione dell'applicazione ed è implementato estendendo la classe `ft.UserControl` della libreria Flet. La struttura del file è stata progettata in modo modulare per facilitare la manutenzione e l'estensione dell'interfaccia.

  Nel costruttore (`__init__`), vengono inizializzate le variabili d'istanza che rappresentano i vari elementi grafici, quali i campi di input, i pulsanti, i grafici e le sezioni di personalizzazione. Vengono inoltre definiti alcuni parametri della pagina (dimensioni, tema, allineamento, possibilità di ridimensionamento e scroll) per garantire una configurazione ottimale dell'interfaccia.

  La costruzione dell'interfaccia è suddivisa in metodi privati (es. `_build_row_activity()`), ognuno dei quali si occupa di creare una specifica sezione della pagina. Questa organizzazione modulare rende il codice più leggibile e consente di isolare facilmente le modifiche a specifiche aree dell'interfaccia senza impattare sull'intera struttura.

  Infine, il metodo `load_interface()` funge da punto di ingresso per l'assemblaggio complessivo dell'interfaccia, richiamando in sequenza i metodi privati per costruire e aggiungere i componenti alla pagina. L'utilizzo di `Container` con padding, spaziatura e border radius uniformi garantisce un layout coerente e visivamente gradevole. La struttura del file `view.py` è studiata per separare nettamente la definizione degli elementi UI dalla logica di aggiornamento della pagina, assicurando così una facile integrazione con il controller e il modello.

- **controller.py:** implementa la logica di controllo, mediando tra il `Model` e la `View`. Il controller gestisce la validazione degli input, il calcolo del fabbisogno energetico, l'attivazione dell'algoritmo di ottimizzazione e la personalizzazione della lista della spesa.

Database: (file di gestione del *database*)
La cartella `database` contiene:

- `DAO.py`: La classe `DAO` centralizza le operazioni di accesso al database, eseguendo query e trasformando i dati in oggetti `Food`. I metodi principali sono:
  - `getAllFood()` – restituisce l'elenco completo dei prodotti alimentari.
  - `getFoodPers(switches_state)` – estrae i dati filtrati in base alle preferenze dell'utente.
  - `getFoodByNutrient(nutrient)` – restituisce una lista di alimenti ordinati in base a un particolare micronutriente.
- `connector.cnf`: file di configurazione con le credenziali di accesso al database.
- `DB_connect.py`: gestisce la connessione al database tramite un pool di connessioni.
- `DAO.py`: implementa la logica di accesso ai dati.
- `final_food_db.sql`: contiene lo schema e i dati di base.

Il file `main.py` rappresenta il punto di ingresso dell'applicazione, mentre `Requirements.txt` elenca le dipendenze (tra cui **Flet** e **PuLP**). I file `LICENSE` e `README.txt` forniscono le condizioni d'uso e le istruzioni per l'installazione.

4.2 Algoritmi Coinvolti

Il progetto utilizza formule matematiche per il calcolo del TDEE, del BMR e per la partizione dei nutrienti; tuttavia, l'algoritmo centrale è quello di ottimizzazione, implementato tramite un modello di Programmazione Lineare Intera Mista (MILP).

4.2.1 Ottimizzazione

L'approccio adottato consiste nel minimizzare la somma pesata delle deviazioni tra i valori nutrizionali ottenuti dalla combinazione degli alimenti e i target definiti, che si basano sul TDEE e sui dati biometrici dell'utente.

- **Funzione Obiettivo:**
  La funzione obiettivo è definita come:

  $$\min \sum_{n \in N} w_n (d_n^+ + d_n^-)$$

  dove w_n rappresenta il peso assegnato al nutriente n (ad esempio, 4 kcal/g per proteine e carboidrati, 9 kcal/g per i grassi, 0.5 per le fibre), e d_n^+ e d_n^- sono le deviazioni positive e negative dal target.

- **Vincoli:**
  Per ogni nutriente n, il modello impone:

  $$\sum_{i \in I} a_{in} x_i + d_n^- - d_n^+ = T_n$$

  dove a_{in} è il contributo nutrizionale dell'alimento i per il nutriente n, x_i è il numero di porzioni di alimento i e T_n è il target nutrizionale per n. Inoltre, si impone che il numero totale di porzioni rientri in un intervallo prestabilito (ad es., da 5 a 15).

```python
def milp_optimization(self, tdee: float, userdata: dict, preferences: dict):
    foods = self.get_filtered_food(preferences)
    random.shuffle(foods)

    target = self.calculate_nutrients_requirement(tdee, userdata)
    nutrient_keys = ["CaloricValue", "Protein", "Carbohydrates", "Fat", "Fiber"]
    target_floats = {k: float(target[k]) for k in nutrient_keys}

    prob = pulp.LpProblem( name: "GroceryOptimization", pulp.LpMinimize)

    # Definizione variabili decisionali per ogni alimento (numero di porzioni, intero)
    food_vars = {}
    for food in foods:
        food_vars[food.ID] = pulp.LpVariable( name: f"x_{food.ID}", lowBound=0, upBound=5, cat="Integer")

    # Variabili slack per misurare le deviazioni dai target
    d_plus = {}
    d_minus = {}
    for n in nutrient_keys:
        d_plus[n] = pulp.LpVariable( name: f"d_plus_{n}", lowBound=0, cat="Continuous")
        d_minus[n] = pulp.LpVariable( name: f"d_minus_{n}", lowBound=0, cat="Continuous")
    nutrient_weights = {
        "CaloricValue": 1.0,
        "Protein": 4.0,
        "Carbohydrates": 4.0,
        "Fat": 9.0,
        "Fiber": 0.5
    }

    # Funzione obiettivo: minimizzo la somma pesata delle deviazioni
    prob += pulp.lpSum([nutrient_weights[n] * (d_plus[n] + d_minus[n]) for n in nutrient_keys])

    # Vincoli per ciascun nutriente: somma_i (valore_i * x_i) + d_minus - d_plus = target
    for n in nutrient_keys:
        prob += (pulp.lpSum([float(getattr(food, n)) * food_vars[food.ID] for food in foods])
                 + d_minus[n] - d_plus[n] == target_floats[n]), f"constraint_{n}"

    # Vincolo sul numero totale di porzioni (ad es. tra 5 e 15)
    prob += pulp.lpSum([food_vars[food.ID] for food in foods]) >= 5, "min_servings"
    prob += pulp.lpSum([food_vars[food.ID] for food in foods]) <= 15, "max_servings"

    # limit e gapRel per accettare una soluzione sub-ottimale in pochi secondi
    prob.solve(pulp.PULP_CBC_CMD(timeLimit=5, gapRel=0.1, msg=True))
    solution = []
    for food in foods:
        qty = food_vars[food.ID].varValue
        if qty and qty > 0:
            solution.append((food, qty))
    objective_value = pulp.value(prob.objective)
    print(f"Optimal objective (sub-optimal accettata): {objective_value}")
    for n in nutrient_keys:
        print(f"{n}: d_plus = {d_plus[n].varValue}, d_minus = {d_minus[n].varValue}")

    return solution, objective_value
```

Figura 3: funzione milp optimization responsabile dell'ottimizzazione

4.2.2 Spiegazione Passo per Passo della Funzione `milp_optimization`

La funzione `milp_optimization` si articola nei seguenti passaggi:

1. **Acquisizione e Filtraggio dei Dati:**
   Vengono estratti e filtrati i prodotti alimentari dal database in base alle preferenze dell'utente, ottenendo una lista di oggetti `Food` che viene randomizzata per garantire soluzioni variegate.

2. **Calcolo dei Target Nutrizionali:**
   Utilizzando il TDEE e i dati biometrici, il metodo `calculate_nutrients_requirement` definisce il benchmark nutrizionale personalizzato.

3. **Definizione del Problema MILP:**
   Viene creato un oggetto `LpProblem` con l'obiettivo di minimizzare la somma pesata delle deviazioni, dove ciascuna deviazione è moltiplicata per il peso del nutriente corrispondente.

4. **Definizione delle Variabili:**
   Per ogni alimento, viene definita una variabile intera che rappresenta il numero di porzioni, e per ogni nutriente, variabili slack d_n^+ e d_n^- per misurare le discrepanze rispetto al target.

5. **Imposizione dei Vincoli Nutrizionali:**
   Per ogni nutriente, la somma dei contributi degli alimenti selezionati, corretta dalle variabili slack, deve essere uguale al target nutrizionale.

6. **Vincoli sul Totale delle Porzioni:**
   Il modello impone che il numero totale di porzioni sia compreso in un intervallo realistico (ad esempio, da 5 a 15).

7. **Risoluzione del Problema:**
   Il modello viene risolto utilizzando il solver CBC di PuLP, con un limite di tempo e un gap relativo per ottenere una soluzione sub-ottimale in tempi brevi.

8. **Estrazione della Soluzione:**
   I valori delle variabili decisionali vengono raccolti per formare la lista della spesa ottimizzata, che viene restituita insieme al valore dell'obiettivo e alle deviazioni per ciascun nutriente.

Questo approccio consente di individuare in maniera efficiente una combinazione di alimenti che si avvicini al target nutrizionale desiderato, minimizzando le discrepanze e garantendo un output valido per la generazione della lista della spesa.

5 Esempi di Utilizzo e Video Dimostrativo

5.1 Videate in fase di utilizzo

5.1.1 Input utente (Screenshot 1)

L'utente inserisce i propri dati biometrici (peso, altezza, età, genere) e il livello di attività fisica. Queste informazioni saranno utilizzate per calcolare il TDEE.

Figura 4: Schermata di Input Utente: Inserimento dei dati biometrici (peso, altezza, età, genere).

5.1.2 Calcolo TDEE (Screenshot 2)

Una volta inseriti i dati, il sistema mostra il TDEE (fabbisogno calorico giornaliero), che potrà essere incrementato o decrementato di 100 kcal per adattarsi a esigenze specifiche.

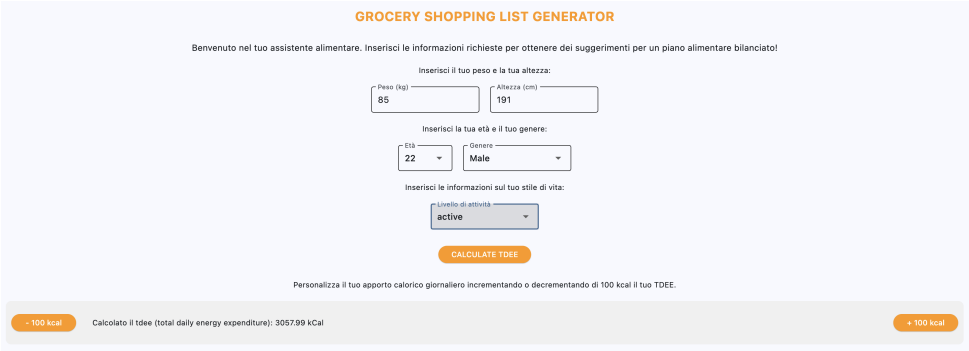

Figura 5: Schermata di Calcolo TDEE: Visualizzazione del fabbisogno calorico giornaliero calcolato.

5.1.3 Filtro alimenti (Screenshot 3)

L'utente può attivare o disattivare vari switch per escludere categorie di cibi in base a preferenze dietetiche, intolleranze o allergie (ad es. lattosio, glutine, frutta secca).

Figura 6: Schermata di Filtraggio Alimenti: Selezione delle preferenze alimentari tramite toggle.

5.1.4 Generazione output (Screenshot 4)

Il sistema produce una lista della spesa ottimizzata, evidenziando per ogni alimento le porzioni suggerite, i nutrienti e l'apporto calorico.

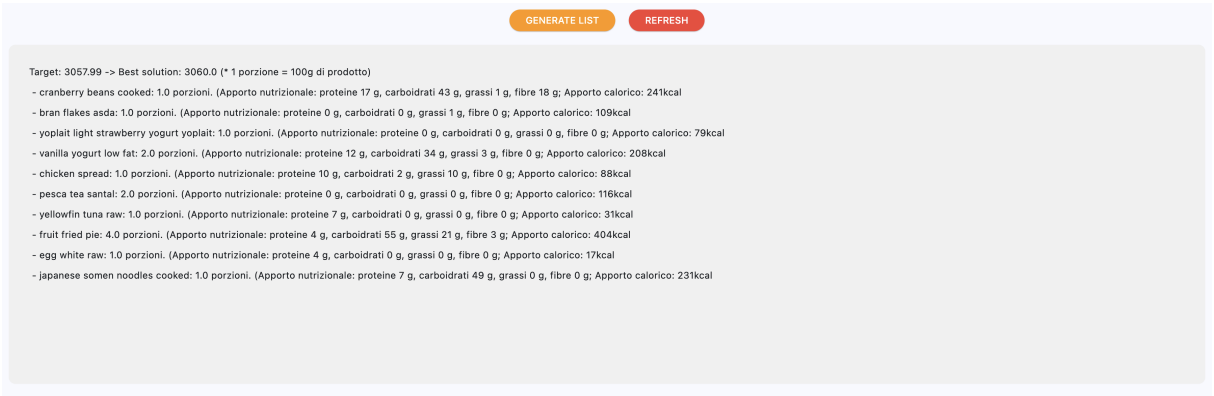

Figura 7: Schermata di Generazione Output: Lista della spesa ottimizzata basata sul TDEE e sui dati inseriti.

5.1.5 Personalizzazione output (Screenshot 5)

L'utente può modificare la lista sostituendo alimenti indesiderati con prodotti nutrizionalmente simili, incrementando o decrementando il numero di porzioni, pur mantenendo l'equilibrio nutrizionale complessivo.

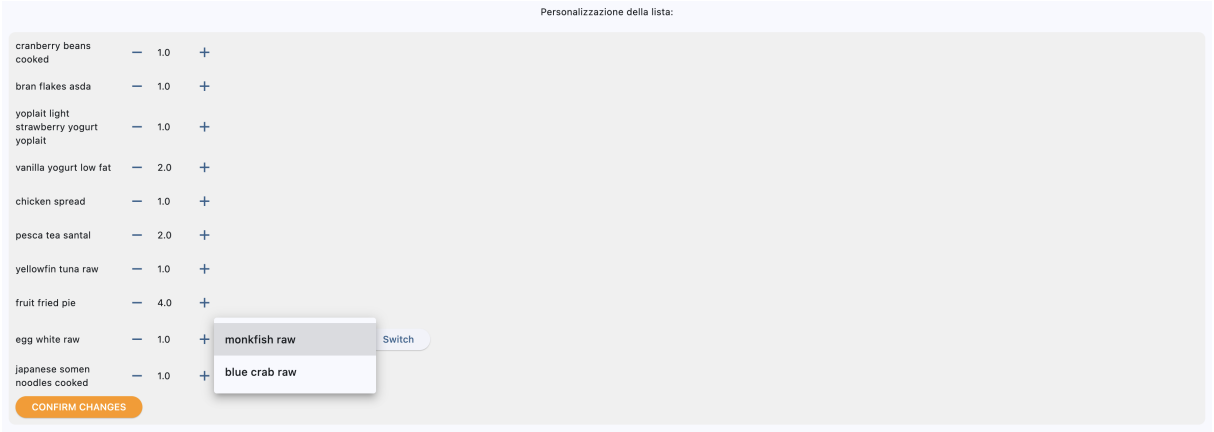

Figura 8: Schermata di Personalizzazione Output: Interfaccia per modificare la lista della spesa e sostituire alimenti.

5.1.6 Sezione micronutrienti (Screenshot 6)

L'utente seleziona un micronutriente dal menu a tendina per visualizzare una breve descrizione informativa e un elenco di alimenti che lo contengono, ordinati in base al quantitativo presente.

Figura 9: Schermata della Sezione Micronutrienti: Visualizzazione dei micronutrienti, relativo testo informativo e listview dei cibi.

5.2 Link al video dimostrativo

Per una dimostrazione completa del funzionamento dell'applicazione, è disponibile il seguente video su YouTube:

`https://youtu.be/zghf-64VrOO`

6 Valutazione dei risultati ottenuti e conclusioni

L'obiettivo principale di questo progetto è quello di semplificare la vita degli utenti automatizzando il processo di organizzazione e bilanciamento della dieta quotidiana. A partire dai dati biometrici e dalle preferenze alimentari, il sistema genera una lista della spesa ottimizzata che soddisfa un *benchmark* nutrizionale personalizzato. Il flusso integrato comprende il calcolo del BMR e del TDEE, il filtraggio dei dati alimentari e l'applicazione di un algoritmo di ottimizzazione basato su un modello MILP, capace di minimizzare le discrepanze tra i valori nutrizionali ottenuti e quelli *target*.

Durante lo sviluppo, sono emerse due criticità principali:

- **Gestione del *dataset*:** la mancanza di un *dataset* completo ha richiesto intensi processi di *data cleaning* e *data management* per adattare i dati alle esigenze del progetto.

- **Complessità computazionale dell'ottimizzazione:** L'algoritmo MILP, sebbene efficace, richiede risorse computazionali significative, soprattutto con l'aumento del numero di variabili.

Il risultato finale dimostra che il sistema è in grado di generare soluzioni che si avvicinano ai *target* nutrizionali in tempi ragionevoli, confermando la validità dell'approccio adottato. Questo strumento offre potenziali applicazioni sia in ambito individuale che commerciale, ad esempio come modulo integrabile in piattaforme di spesa *online* atte a fornire consigli nutrizionali personalizzati.

In conclusione, il progetto ha raggiunto il suo obiettivo di automatizzare la pianificazione alimentare, semplificando un processo spesso trascurato e contribuendo al miglioramento della qualità della vita degli utenti. Le prospettive future includono il perfezionamento del *dataset* e l'ottimizzazione dell'algoritmo per aumentare la scalabilità e la reattività, integrando ulteriori parametri nutrizionali e adattamenti dinamici basati sui *feedback* degli utenti.

7 LICENZA