

Università degli Studi di Messina



Dipartimento di Scienze matematiche e informatiche,
scienze fisiche e scienze della terra

Corso di Laurea Triennale in INFORMATICA

Progetto di
INGEGNERIA DEL SOFTWARE
PROGRAMMAZIONE II



SOTWARE DEVELOPMENT DOCUMENT

Realizzato da

DE PIETRO BERNARDO
PAOLO MORABITO

Anno Accademico
2019/2020

Summary

1.	Overview.....	6
1.1	Scope.....	6
1.2	References Documents	6
1.3	Project Description.....	6
2.	Overview of Software Development Planning.....	8
2.1	Descriptive Language	8
2.2	Technologies.....	8
2.3	System Cycle Life.....	9
2.4	Team Work.....	10
2.5	Schedules and Resources	10
2.6	Training Requirements.....	10
2.7	Project Directives.....	10
3.	Concept Phase.....	11
3.1	Application Domain	11
3.2	Feasibility Study.....	12
3.3	Risk Analysis.....	15
4.	First Iteration (Prototype 1).....	17
4.1	Requirements Phase.....	17
4.1.1	Functional Requirements.....	21
4.1.2	Non-Functional Requirements.....	21
4.1.3	Matrix of Functional Requirements	22
4.2	Design Phase	25
4.2.1	Software Logical Structure	25
4.2.2	Component Structure	28
4.2.3	System Architecture.....	29
4.2.4	Server.....	30
4.2.5	Communication Protocol.....	31
4.2.6	GUI Design	32
4.2.7	Implementation Guidelines.....	41
4.2.8	Algorithm of Goal's Scheduling.....	41
4.2.9	Activity Diagrams.....	42
4.3	Implementation Phase.....	45
4.3.1	Configuration	42
4.3.2	Database.....	46
4.3.3	Client	46
4.3.4	Security.....	48
4.3.5	Physical Software Structure.....	50
4.3.6	GUI Structure	54

4.3.7	GUI's Screens (UTENTE).....	54
4.3.8	GUI's Screens (DOTTORE).....	57
4.3.9	GUI's Screens (ADMIN).....	58
4.4	Testing Phase.....	60
4.4.1	Validation.....	60
4.4.2	Verification	64
4.5	Customer Review and Feedback	65
5.	Second Iteration (Prototype 2)	66
5.1	Requirements Phase.....	66
5.1.1	New Non-Functional Requirements	66
5.1.2	New Functional Requirements	66
5.1.3	Matrix of Functional Requirements (update)	66
5.2	Design Phase	69
5.2.1	Technical Update	69
5.2.2	Component Structure	69
5.2.3	GUI Design	70
5.3	Implementation Phase.....	71
5.3.1	GUI's Screens.....	72
5.3.2	Software Deployment.....	73
5.4	Testing Phase.....	74
5.4.1	Validation.....	74
5.4.2	Verification.....	75
5.5	Customer Review and Feedback	76
6.	Release Phase	76
6.1	Release 1.0.....	76

LIST OF ACRONYMS AND ABBREVIATIONS

- UML** - Unified Modeling Language
RQ - Requisito Funzionale
RQNF - Requisito Non Funzionale
GUI – Graphical User Interface
HTTP – Hypertext Transfer Protocol

LIST OF TABLES

Matrice dei rischi -----	Pag.15
Matrice dei requisiti funzionali (Prototipo 1) -----	Pag.23
Matrice dei requisiti funzionali (Prototipo 2) -----	Pag.66

LIST OF FIGURES

Figura 1 – Prototipizzazione Evolutiva -----	Pag. 9
Figura 2 – Application Domain (Context Diagram) -----	Pag.11
Figura 3 – Diagramma di GIANTT I -----	Pag.13
Figura 4 – Diagramma: influenza dei costi nel tempo -----	Pag.13
Figura 5 – Use Case diagram (Simplified Version) -----	Pag.17
Figura 6 – Use Case Diagram (Extended version) -----	Pag.18
Figura 7 – State Diagram: Gestione accesso in funzione del utente -----	Pag.25
Figura 8 – State Diagram: Gestione degli obiettivi in funzione dell’utente -----	Pag.26
Figura 9 – State Diagram: Richieste in funzione dell’utente -----	Pag.26
Figura 10 – State Diagram: Gestione Piattaforma -----	Pag.27
Figura 11 – Component Diagram Software-----	Pag.28
Figura 12 – Component Diagram Software(Dataclient)-----	Pag.28
Figura 13 – Architettura Client-Server-----	Pag.30
Figura 14 – Protocollo HTTP (Request/Response)-----	Pag.31
Figura 15 – Interfaccia: Dashboard del software (UTENTE)-----	Pag.32
Figura 16 – Interfaccia: Dashboard del software (DOTTORE)-----	Pag.33
Figura 17 – Interfaccia: Dashboard del software (ADMIN)-----	Pag.34
Figura 18 – Interfaccia: Definizione obiettivo -----	Pag.34
Figura 19 – Interfaccia: Schermata obiettivi -----	Pag.35
Figura 20 – Interfaccia: Schermata assunzione alimento -----	Pag.35
Figura 21 – Interfaccia: Schermata obiettivi dottore-----	Pag.36
Figura 22 – Interfaccia: Schermata richieste dottore(menu utente)-----	Pag.36
Figura 23 – Interfaccia: Schermata invio richieste dottore-----	Pag.37
Figura 24 – Interfaccia: Schermata richieste dottore -----	Pag.37
Figura 25 – Interfaccia: Schermata richieste dottore (dottore)-----	Pag.38
Figura 26 – Interfaccia: Schermata controllo operato-----	Pag.38
Figura 27 – Interfaccia: Schermata gestione piattaforma -----	Pag.39
Figura 28 – Interfaccia: Schermata resoconto -----	Pag.39
Figura 29 – Interfaccia: Schermata gestione accesso -----	Pag.40
Figura 30 – Interfaccia: Schermata gestione utente-----	Pag.40
Figura 31 – Activity Diagram: Pre-Scheduling -----	Pag.42
Figura 32 – Activity Diagram: Calcolo fabbisogno calorico -----	Pag.43
Figura 33 – Activity Diagram: Avvia Scheduling -----	Pag.44

Figura 34 – Tabelle del Database -----	Pag.46
Figura 35 – Class Diagram: Database Server -----	Pag.46
Figura 36 – Class Diagram: DataClient -----	Pag.48
Figura 37 – Sequence Diagram -----	Pag.49
Figura 38 – Package Diagram -----	Pag.50
Figura 39 – Class Diagram: Alimentazione -----	Pag.51
Figura 40 – Class Diagram: Messaggistica -----	Pag.51
Figura 41 – Class Diagram: Autenticazione -----	Pag.52
Figura 42 – Class Diagram: Utente -----	Pag.52
Figura 43 – Class Diagram: Dottore -----	Pag.53
Figura 44 – Class Diagram: Admin -----	Pag.53
Figura 45 – Component Diagram Data-Client (UPDATE)-----	Pag.69
Figura 46 – Interfaccia: Schermata gestione statistiche-----	Pag.70
Figura 47 – Interfaccia: Schermata resoconto (UPDATE)-----	Pag.70
Figura 48 – Package Diagram (UPDATE) -----	Pag.71
Figura 49 – Class Diagram: Statistiche -----	Pag.71
Figura 50 – Deployment Diagram-----	Pag.73

*Sono omessi dalla lista gli screen del Software.

1. Overview

1.1 Scope

L'obiettivo del documento è quello di definire l'approccio, le metodologie, gli strumenti e le procedure per lo sviluppo del software da utilizzare durante l'analisi, la progettazione, lo sviluppo, la verifica, l'integrazione, la distribuzione e la manutenzione del software.

Al termine dello sviluppo il documento può considerarsi come una descrizione dell'intero ciclo di vita del software, dalle fasi iniziali fino al rilascio.

1.2 References Documents

Per ulteriori approfondimenti si consiglia la consultazione dei seguenti documenti:

- **Software Documentation** (documento del prodotto finale).

1.3 Project Description

Il progetto prevede lo sviluppo di un software nell'**ambito health**, capace di gestire il diario alimentare giorno dopo giorno e dunque monitorare l'alimento assunto. L'assioma di base del software è che mangiare alimenti sani è importante, ma lo è anche la consapevolezza della quantità di calorie necessarie ogni giorno. Il problema sorge quando viene incamerata più energia di quella che si consuma o, al contrario, quando si entra in riserva senza rendersene conto. Il software dovrà svolgere diverse funzioni, che contribuiranno ad un monitoraggio continuo:

- 1) *Piano calorico personalizzato per poter perdere peso, mantenere il proprio peso e aumentare il proprio peso*
- 2) *Tabella delle calorie per tutti i pasti con annessi valori nutrizionali in relazione a proteine, carboidrati e grassi*
- 3) *Calcolatore per tenere traccia delle calorie assunte*
- 4) *Tabella analitica con i risultati ottenuti in un dato periodo*

Il funzionamento del software si dovrà basare su delle semplici operazioni che permetteranno di:

- 1) *Impostare gli obiettivi personali per perdere, prendere o mantenere il peso.*
- 2) *Tracciare i risultati degli obiettivi raggiunti*
- 3) *Controllare e analizzare ciò che viene assunto*
- 4) *Tenere traccia dei progressi, negativi o positivi, durante il tempo di utilizzo del software*

Il software, dovrà gestire il diario alimentare in funzione di principi cardine della alimentazione che, implementati nel modo corretto, genereranno un algoritmo la cui esecuzione porterà un risultato univoco e personale. Innanzitutto, verrà calcolato il fabbisogno calorico giornaliero, che dipenderà dal sesso (uomo/donna), dall'età (espressa in anni), dal peso corporeo (espresso in kg), dall'altezza (espressa in cm) e dal tipo di attività svolta, che viene generalmente suddivisa in 2 tipologie:

- 1) *Attività lavorativa: da scrivania, lavoro leggero, lavoro mediamente pesante, lavoro pesante.*
- 2) *Attività fisica: nessuna attività fisica, 2-3 volte a settimana*

Dopo questo calcolo preventivo avremo ottenuto le kcal totali giornaliere che dovranno essere ripartite all'interno dei diversi momenti della giornata, adottando il seguente schema:

- 1) **Colazione**, 15% delle kcal totali
- 2) **Spuntino di metà mattina**, 5% delle kcal totali
- 3) **Pranzo**, 40% delle kcal totali
- 4) **Spuntino di metà pomeriggio**, 5% delle kcal totali
- 5) **Cena**, 35% delle kcal totali

Per rispondere alle richieste dell'utente, bisogna ripartire le calorie giornaliere tra tre tipologie nutrizionali, ovvero proteine, carboidrati e grassi, in funzione degli obiettivi posti al software in precedenza, ovvero:

- 1) **Mantenimento del peso**, 25% proteine, 25% grassi, 50% carboidrati
- 2) **Perdita del peso**, 45% proteine, 25% grassi, 35% carboidrati
- 3) **Aumento del peso(esigenze muscolari)**, 35% proteine, 20% grassi, 45% carboidrati

L'utente finalizzatore, estraneo all'algoritmo, dovrà semplicemente inserire i cibi assunti nelle varie sezioni giornaliere, così potrà valutare se in linea con le specifiche fornite dal software; risulta chiaro che potrà assumere un numero inferiore o maggiore di kcal ma precluderà il non raggiungimento dei propri obiettivi (e dovrà essere notificato l'andamento negativo)

Andando oltre le specifiche "healt" del software, dovranno essere presenti 3 tipologie di utente:

- 1) **Admin**, che potrà aggiungere dottori alla piattaforma e osservare l'operato dei dottori.
- 2) **Dottore**, che potrà essere consultato dagli utenti nel caso in cui un utente non riesce a raggiungere gli obiettivi posti in essere, convalidare l'aggiunta di un alimento che precedentemente è stato proposto dall'utente o altre richieste generiche.
- 3) **Utente**, colui che utilizzerà la piattaforma nella sua totalità e che inoltre, nel caso non trovi un alimento all'interno della piattaforma andrà a inviare una richiesta ad un dottore con il tipo di alimento correlandolo con una descrizione in modo tale da non avere ambiguità (es. patate, ma fritte o bollite?) ma anche come già anticipato, effettuare richieste generiche ai dottori.

Queste tre tipologie avranno un pannello dedicato per il login e nel caso degli utenti sarà presente un pannello dedicato alla registrazione alla piattaforma (i dottori non dovranno registrarsi in quanto l'admin, gestore della piattaforma, avrà il compito di assumere dottori e inserirli).

2. Overview of Software Development Planning

2.1 Descriptive Language

Al fine di utilizzare un linguaggio universale nella comunità della progettazione si sceglie di utilizzare il linguaggio **UML** (*Unified Modeling Language*). Ciò infatti comporta i seguenti vantaggi:

1. Un sistema software grazie al linguaggio UML viene disegnato professionalmente e documentato ancor prima che ne venga scritto il relativo codice, da parte degli sviluppatori. Si sarà così in grado di conoscere in anticipo il risultato finale del progetto su cui si sta lavorando.
2. Poiché, come detto, la fase di disegno del sistema precede la fase di scrittura del codice, ne consegue che la scrittura del codice stessa è resa più agevole ed efficiente oltre al fatto che in tal modo è più facile scrivere del codice riutilizzabile in futuro. I costi di sviluppo, dunque, si abbassano notevolmente con l'utilizzo del linguaggio UML.
3. È più facile prevedere e anticipare eventuali “buchi” nel sistema. Il software che si scrive, si comporterà esattamente come ci si aspetta senza spiacevoli sorprese finali.
4. L'utilizzo dei diagrammi UML permette di avere una chiara idea, a chiunque sia coinvolto nello sviluppo, di tutto l'insieme che costituisce il sistema. In questo modo, si potranno sfruttare al meglio anche le risorse hardware in termini di memoria ed efficienza, senza sprechi inutili o, al contrario, rischi di sottostima dei requisiti di sistema.
5. Grazie alla documentazione del linguaggio UML diviene ancora più facile effettuare eventuali modifiche future al codice. Questo, ancora, a tutto beneficio dei costi di mantenimento del sistema.

2.2 Technologies

Durante il ciclo di vita del software, sono diverse le tecnologie utilizzate per la realizzazione dello stesso.

Innanzitutto il progetto del software si presta molto per una programmazione modulare basata sugli oggetti. Per questo si è scelto di utilizzare il linguaggio **JAVA**, che risulta efficace per creare il prodotto in quanto lo rende indipendente dalla piattaforma, poiché dispone di un “intermediario” che si occupa della traduzione da java a codice macchina, linguaggio che i sistemi informatici devono poter eseguire, e questo strumento prende il nome di *JVM* (Java Virtual Machine).

Essendo completamente orientato agli oggetti, la possibilità di riutilizzare il codice e di gestire facilmente i progetti di vaste dimensioni rappresentano un sicuro vantaggio per gli sviluppatori.

Lo sviluppo del software, a casa della necessità di una struttura **Client-Server**, ha reso necessario affiancare altre tecnologie. In particolare si sceglie di utilizzare un database **MYSQL**, in quanto si sposa perfettamente con la scelta del linguaggio java grazie ai driver **JDBC** (Java Database Connectivity).

2.3 System Cycle Life

Per lo sviluppo del software, viene scelto un approccio **plan-driven** in cui il processo prevede la traduzione delle esigenze iniziali degli utenti in requisiti che devono essere implementati nel proseguo dello sviluppo. All'utilizzo di questo approccio ne consegue la scelta del modello di sviluppo da seguire.

In particolare viene utilizzato un modello incrementale che si basa sullo sviluppo di passaggi ben definiti che porteranno alla creazione di un primo prototipo. La prototipizzazione, con la variante utilizzata della prototipizzazione evolutiva (***evolutionary prototyping – breadboard prototyping***) presenta numerosi vantaggi in quanto il progettista e l'implementatore del software possono ottenere un prezioso

feedback dagli utenti nelle prime fasi del progetto inoltre ogni prototipo che viene realizzato non viene “cestinato”, come accade nel modello **throw-away**, ma è riutilizzato interamente per farlo evolvere a poco a poco nel prodotto finale.

Il problema maggiore di questo modello è il rischio di essere indisciplinati. Per evitare ciò è bene far uso di standard di processo e non perdere i punti migliori del modello a cascata.

Il ciclo di vita del software prevede quindi i seguenti passaggi:

1. **Concept Phase**, nella quale vengono concettualizzate le caratteristiche principali del progetto;
2. **Requirements Phase**, in cui vengono raccolti i requisiti per la realizzazione del prototipo per l'iterazione corrente;
3. **Design Phase**, nella quale si guida la progettazione della fase corrente;
4. **Implementation Phase**, in cui il sistema viene implementato sulla base del progetto specificato e, in seguito, rilasciato al cliente per la revisione;
5. **Testing Phase**, in cui si verifica il funzionamento nel prototipo;
6. **Customer Review and Feedback**, in cui il cliente esamina il prototipo e fornisce feedback sulle modifiche e nuove funzionalità da costruire nella successiva iterazione, ripartendo dal punto 2. Qualora invece il feedback sia positivo e non vengano richieste altre modifiche, si procede con il punto 7;
7. **Release System**, il prototipo viene rilasciato come prodotto finale.

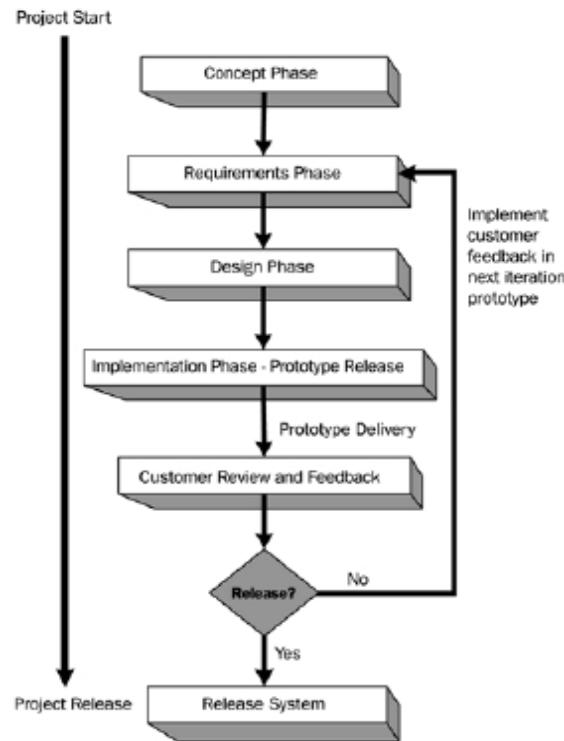


Figura 1 Prototipizzazione Evolutiva

2.4 Team Work

Il team di lavoro si compone di soli due elementi, che ricoprono sia il ruolo di ingegneri del software sia quello di programmatore:

- De Pietro Bernardo
- Morabito Paolo

2.5 Schedules and Resources

Sebbene non sia richiesto un tempo di sviluppo prefissato, si auspica di poter rilasciare il prodotto nell'arco di un mese.

Il piano di sviluppo prevede che entro i primi 20 giorni del mese venga rilasciato il Prototipo del software, in modo tale che il cliente possa valutare il prodotto e ricevere la release finale entro il mese.

Non è possibile prevedere il numero di prototipi necessari, ma per ridurre i tempi può essere utile soffermarsi nella fase di **acquisizione dei requisiti**, in modo che essi siano chiari e dettagliati.

Dal momento che il team di sviluppo risulta essere formato da soli due componenti, non viene effettuata una rigida suddivisione dei compiti e delle risorse. Devono essere gli stessi sviluppatori ad interagire tra loro per riuscire a trovare il giusto equilibrio, man mano che si procede sia con la progettazione che con la programmazione. Proprio per facilitare questo aspetto si è scelta una progettazione modulare.

2.6 Training Requirements

Per poter procedere alla realizzazione del software, il team di lavoro deve avere i seguenti requisiti formativi:

- Conoscenze di **ingegneria del software** (modelli plan-driven);
- Conoscenza della **programmazione ad oggetti**, nel caso specifico del linguaggio **JAVA**;
- Esperienza con uso di **databases**;
- Conoscenza di **reti e protocolli Internet**.

2.7 Project Directives

Prima di procedere con la progettazione vera e propria, vengono stabilite alcune direttive progettuali utili alla stesura della documentazione stessa.

- È necessario essere chiari ed esaurienti per la descrizione sia della progettazione che per l'implementazione del back-end del software;
- Si consiglia di documentare il codice sorgente con commenti approfonditi,
- Per quanto riguarda il front-end (GUI) è possibile trascurarne la descrizione dell'implementazione, ma è bene precisare i dettagli di progettazione.

3. Concept Phase

In questa fase vengono definite le linee guida e le caratteristiche principali del progetto. Sulla base della descrizione delle caratteristiche del progetto (Paragrafo 1.3), si effettua un'analisi del dominio applicativo, uno studio di fattibilità e infine si procede con un'analisi dei rischi.

3.1 Application Domain

Il dominio applicativo del software appartiene al mondo nutrizionale e, nello specifico, si riferisce alle persone e ai loro bisogni alimentari.

Dal momento che il team conosce a pieno questo ambito, non è necessario richiedere l'ausilio di esperti del dominio.

Per descrivere al meglio tale dominio applicativo si fa uso di un Context Diagram. In particolare, si distinguono due macro tipologie di sottodomini: un **machine domain** e un insieme di **problem domain**. Questi ultimi a loro volta vengono suddivisi in **designed domain** (che esprimono la possibilità, in fase di realizzazione della soluzione, di progettarne le caratteristiche) e **given domain** (non è possibile intervenire su nessuna caratteristica di dominio).

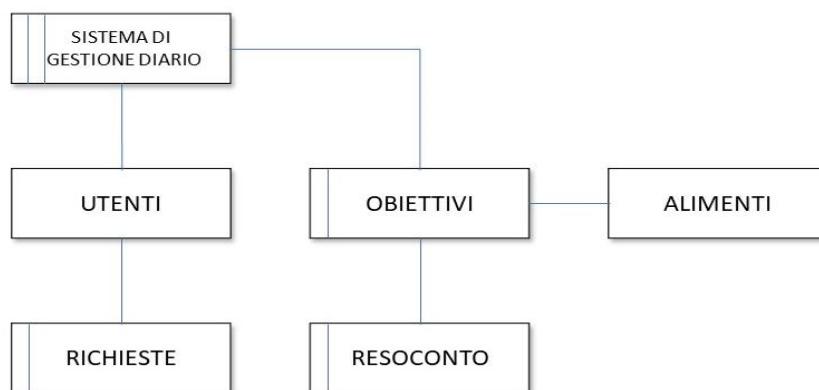


Figura 2 Application Domain (Context Diagram)

1. **Sistema di gestione del diario (machine domain):** è il sistema da sviluppare
2. **Utenti (given domain):** rappresenta gli utenti che useranno il programma.
3. **Richieste (designed domain):** rappresentano le richieste che gli utenti fanno e che i dotti esaminano e che sulla base di esse gli admin valutano l'operato dei dotti.
4. **Obiettivi(designed domain):** sono gli obiettivi inseriti dall'utente, composti dai cibi. Il progettista potrà definire ulteriori caratteristiche utili alla realizzazione del software.
5. **Alimenti (given domain):** sono i cibi inseriti dall'utente per un determinato pasto della giornata.
6. **Resoconto (designed domain):** è il resoconto degli obiettivi che sarà disponibile all'utente

3.2 Feasibility Study

Prima di procedere con lo sviluppo del software, è necessario affrontare uno studio di fattibilità per valutare sia i possibili risultati sia la necessità di rispettare le scadenze previste, oltre che i costi.

Bisogna che il team analizzi delle aree di fattibilità che afferiscono alla fattibilità del progetto. Nello specifico viene analizzata:

- 1) Fattibilità tecnica
- 2) Fattibilità giuridica
- 3) Fattibilità operativa
- 4) Tempo della fattibilità
- 5) Fattibilità economica

3.2.1 Technical Faesability

Il team è in grado di determinare le risorse tecniche e convertire le idee del cliente in sistemi operativi funzionanti, in quanto possiede le nozioni e gli strumenti per operare al meglio.

Le risorse tecniche vengono soddisfatte, dal momento che il team possiede tutti gli strumenti tecnici ed è a conoscenza delle soluzioni algoritmiche e architetturali per la corretta realizzazione del software.

Inoltre, l'elevata conoscenza del dominio applicativo garantisce una posizione favorevole per le scelte progettuali e dunque ogni stakeholder verrà soddisfatto.

3.2.2 Legal Faesability

In nessun modo, gli aspetti del progetto sono in conflitto con requisiti legali né tantomeno con le vigenti normative, che ne regolano inoltre la protezione dei dati.

3.2.3 Operational Faesability

Il team ha già individuato una metodologia operativa, per portare al termine la realizzazione software. Utilizzando uno sviluppo modulare del sistema (modularizzazione) vengono evitati problemi all'interno del team, oltre che favorire l'integrazione dei diversi componenti e la loro eventuale modifica.

3.2.4 Faesability Time

Il team di lavoro, avendo già operato congiuntamente allo sviluppo di software analoghi, possiede la giusta padronanza delle tempistiche e traendo benefici da queste esperienze si ritiene fermamente sicuro che i tempi per la realizzazione (vedi 2.5) sono più che adeguati.

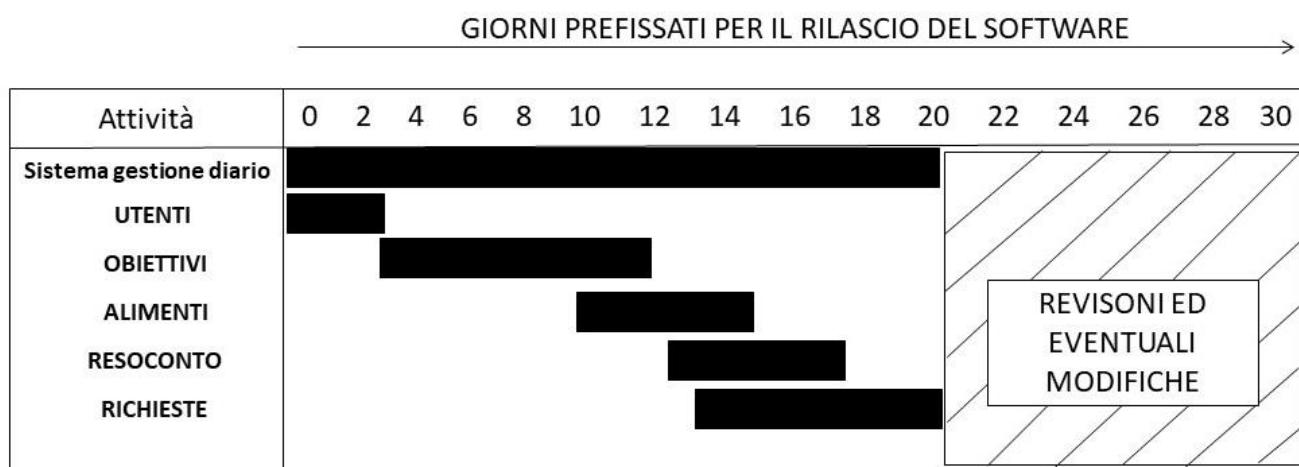


Figura 3 Diagramma di GANTT

Il diagramma di Gantt (fig. 3) rappresenta la Struttura di Ripartizione del Lavoro(WBS) del software ovvero la rappresentazione del progetto suddividendo le attività in livelli e ciò consente un'analisi indispensabile per identificare le attività e i loro tempi che ne pregiudicano lo studio di fattibilità e dunque la corretta riuscita temporale del software. Utilizzando questo grafico, il team ha immediatamente pianificato le tempistiche delle attività da svolgere concedendosi il giusto tempo per eventuali revisioni o modifiche

3.2.5 Economic Faesability

La valutazione economica per la realizzazione del software, mette il team in una posizione vantaggiosa per lo sviluppo, in quanto i costi richiesti sono esigui e dunque il rapporto costi/benefici viene fortemente influenzato dai benefici prodotti dalla realizzazione del software, e per determinare ciò il team ha attentamente valutato l'analisi di una ricerca di mercato.

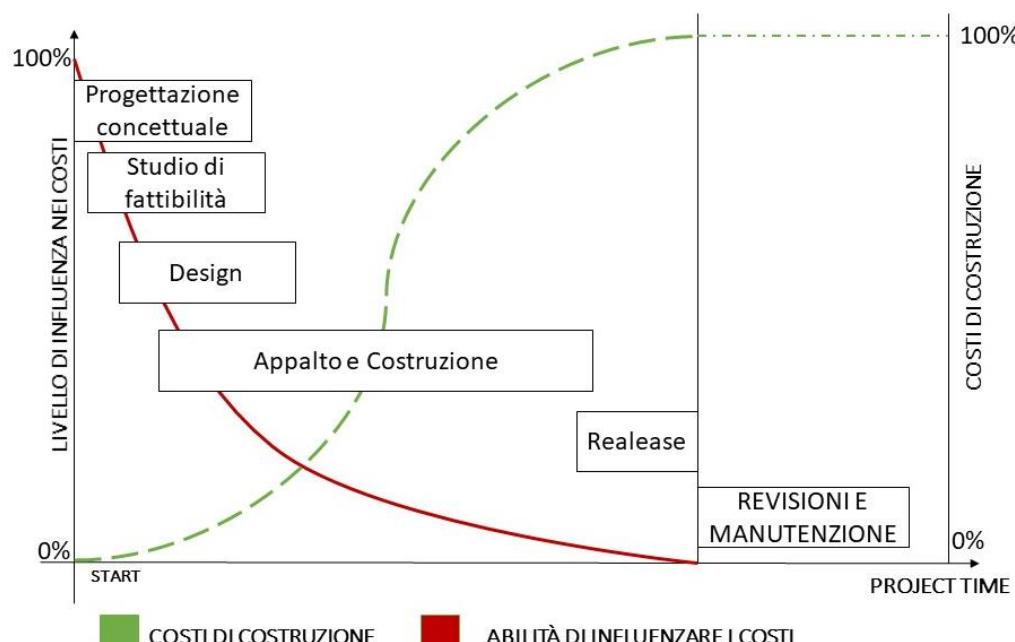


Figura 4 Diagramma: Influenza dei costi nel tempo

Il diagramma (fig. 4) mostra due tracciati evidenti, il primo in **rosso** indica come le fasi del ciclo di vita del software influenzino i costi. Si rende noto, come la pianificazione e lo studio di fattibilità determinino il più dei costi di costruzione poiché sono gli aspetti principali che stanno alla base di una corretta riuscita del software. Il suddetto tracciato risulta inversamente proporzionale all'andamento del tempo poiché le operazioni di cosa saranno marginali e dunque influenzano in modo contenuto i costi. Il tracciato in **verde** indica come nel tempo chiaramente i costi aumentino, ma l'ottimo studio di fattibilità ottenuto ci consente di aver già determinato le tempistiche opportune per lo sviluppo del software e dunque di tenere contenuti i costi senza che ci sia la possibilità di andare oltre alle previsioni del team (**ved. 3.2.4**).

Avendo svolto questa accurata analisi, il progetto risulta ampiamente fattibile e può essere resa al customer una quantificazione diretta del software.

Predisponendo, come limite massimo, **30 giorni lavorativi** per la realizzazione del software si ritiene necessario che vengano impiegate **450 ore-uomo**.

Il grafico precedente ci ha mostrato l'effetto di ogni singola **fase** sui costi in relazione del tempo, e adesso verrà quantificata in modo tale da avere chiara la proposta:

Progettazione concettuale: **90 ore-uomo necessarie.**

Studio di fattibilità: **30 ore-uomo necessarie.**

Design: **80 ore-uomo necessarie.**

Costruzione(implementazione): **100 ore-uomo necessarie.**

Il team dunque ritiene che nelle 300 ore impiegate riuscirà a completare il progetto e si riserva ulteriori 150, eventuali, ore-uomo in modo tale che vengano prese in considerazione possibili revisioni e migliorie in seguito ai feedback del customer.

Come precedentemente descritto il team è composto da due sviluppatori/ingegneri del software di pari competenze e risulta chiaro che verranno suddivise in modo equo le ore complessive, dunque con un totale di 450 ore vengono quantificate 225 ore-uomo ciascuno.

Il team offre un prezzo vantaggioso, proponendosi con un salario fisso di 14,00 €/ora-uomo che concretamente porterà il costo complessivo del software ad una cifra pari a € 6.300,00.

Il team ritiene che avendo predisposto una realizzazione in tempiceleri vengano svolte in modo **straordinario**, rispetto alle solite 5 ore, circa 7.5 ore giornaliere e dunque tutto ciò è stato tenuto in considerazione nella scelta del salario fisso.

$$450\text{ore}/30\text{giorni} = 15\text{ore/giorno} \rightarrow \mathbf{7.5\text{ ore/giorno ciascuno}}$$

$$\mathbf{14\text{ €/ora} \times 450\text{ ore} = 6.300,00\text{ €}}$$

Si vuole precisare, che ci si trova in una posizione fortemente concorrenziale che pone la proposta decisamente appetibile sul piano dello sviluppo ed economico.

3.3 Risk Analysis

All'interno del processo di sviluppo del software ci sono molte aree di rischio ed il team dovrà sviluppare dei modi per assicurare che una possibile occorrenza di un evento indesiderato possa essere individuata, corretta e risolta per evitare conseguenze disastrose.

In relazione al software dobbiamo distinguere due diversi casi possibili di rischi:

- 1) Rischi di progetto
- 2) Rischi di prodotto

3.3.1 Rischi di Progetto

Il rischio principale che è necessario evitare durante la progettazione del software è quello di non riuscire a rendere il prodotto adattabile ad ogni tipologia di utente.

L'alimentazione di una persona è infatti fortemente influenzata da diversi fattori, ovvero, le attività svolte durante la giornata, il proprio stato di salute, la quantità di cibi assunti e dal proprio metabolismo e tutti questi rendono un diario alimentare estremamente personale e univoco. Per evitare di cadere in questo problema, si sceglie di andare a delineare molteplici opzioni per rendere il software universalmente funzionale.

Come la descrizione del problema ci ha mostrato, l'utente in base ad obiettivi, attività e informazioni personali potrà visualizzare gli obiettivi giornalieri da seguire.

Un rischio da non sottovalutare nello sviluppo di un software è quello di non riuscire a consegnare il **prodotto** entro e non oltre le tempistiche pattuite, ma il team tramite le proprie competenze e la scelta di modularizzare lo stesso, potrà aumentare la produttività in quanto lavorando separatamente sui singoli componenti, il consolidamento finale sarà semplice e ben strutturato e nel caso di futuri update del software, le modifiche saranno anch'esse modularizzate (ottimizzandone lo sviluppo). La progettazione utilizzando la metodologia ad incremento permette di avere sin da subito chiare le specifiche del software, dunque non sussiste il rischio che i requisiti funzionali possano evolvere in modo tale da stravolgere un prototipo ed a giovarne è il team di sviluppo.

3.3.2 Rischi di Prodotto

I rischi di prodotto che si possono trovare andando a sviluppare il software sono molteplici, ad esempio l'incapacità di potersi adattare alle piattaforme su cui viene impiantato. Come già ampiamente descritto (**ved. 2.2**), le tecnologie scelte risolvono questo possibile problema in fase di deployment, accertando che non sia un prodotto fortemente agganciato ad una determinata piattaforma.

Il Team, grazie alla propria esperienza consolidata nel tempo, è a conoscenza dei requisiti per rendere un software appetibile all'utente finale, e dunque lo sviluppo delle interfacce e di conseguenza dello sviluppo dei prototipi non è casuale ma certamente mirata in modo tale che non possa esserci il rischio che un errore di "costruzione" delle interfacce possa determinare la non consegna o il ritardo del prodotto finale. La metodologia della prototipizzazione sicuramente avvantaggia il team a rispettare le scadenze e i requisiti funzionali scelti dal cliente.

RISCHI	VALUTAZIONE RISCHIO	VALUTAZIONE PROBABILITÀ CHE ACCADA	POSSIBILI CONSEGUENZE	METODO DI RISOLUZIONE PER AZZERARE IL RISCHIO
Adattabilità software ad ogni utente	ALTO	IMPROBABILE	Utente impossibilitato a creare il proprio diario alimentare	Sul piano funzionale, risulta estremamente improbabile che si verifichi questa situazione poiché le opzioni di base permette di racchiudere tutte le tipologie di utenti possibili.
Basso numero di dottori	ALTO	PROBABILE	Le richieste degli utenti sono più di quanto i dottori possono processarne	Per risolvere questo rischio, sarà compito dell'admin(gestore della piattaforma) ad assumere dottori e inserirli tramite il proprio pannello di inserimento.
Scoraggiamento dell'utente	MEDIO	PROBABILE	L'utente è snervato dalla non presenza dei cibi e dunque non può inserire gli alimenti assunti e abbandona il software.	Per risolvere questo rischio viene a priori inserita una vasta scelta di alimenti con tutte le possibili varianti, ma nonostante ciò viene concessa all'utente la possibilità di suggerire l'inserimento di un alimento al dottore, che valuterà le proprie proprietà nutrizionali e le inserirà
Obiettivi non raggiungibili	ALTO	PROBABILE	Nessun obiettivo raggiunto dopo aver seguito i diktat del software	Il software espone in modo preciso in base alle specifiche dell'utente, le porzioni da assumere. In caso di non raggiungimento, controllare se i valori assunti coincidono con quelli stabiliti altrimenti contattare il dottore
Difficoltà a reperire un dottore	MEDIO	POCO PROBABILE	L'utente in seguito a perplessità o richieste, non riesce a un dottore e rimane in stallo	Viene scelto di risolvere questo rischio, andando ad implementare dei canali di messaggistica tra un utente e più dottori, e il dottore presenterà una "casella" con i vari messaggi degli utenti
Richieste del customer poco chiare	ALTO	PROBABILE	Il team operando senza aver verificato la descrizione del software da parte del customer e porterà a termine un software non congruo	Il Team dovrà obbligatoriamente proporre la software description affinchè il customer non la convaliderà, in modo tale che insieme al team si operi con un obiettivo comune
Prodotto non congruo alla piattaforma	MEDIO	POCO PROBABILE	Il software viene impiantato su un device che non supporta il software	Il team utilizzerà un linguaggio di programmazione a oggetti quale il java che fa della indipendenza di piattaforma la sua forza e dunque il rischio non sussisterà
Schedulazione del fabbisogno calorico che compromette la salute dell'utente	ALTO	PROBABILE	L'utente non avrà un fabbisogno bilanciato e non solo rischia di non raggiungere gli obiettivi ma anche di andare incontro a problemi di salute	Per ovviare a questo rischio il team, inserirà il massimo dei controlli per reperire più informazioni possibili affinchè l'utente presenti il giusto apporto calorico in funzione delle attività svolte e del proprio stato fisico
Tempistiche di consegna non adeguate	BASSO	IMPROBABILE	Il team non ottemperando alle tempistiche concordate avrà ripercussioni economiche da parte del cliente	Lo studio di fattibilità precedentemente discusso pone il team in una situazione di assoluta improbabilità che il software non venga rilasciato nel tempo pattuito, avendo già stabilito a priori le tempistiche
Evoluzione delle richieste del customer	MEDIO	POCO PROBABILE	L'utente trova il software non congruo alle richieste iniziali.	Il team prima di operare effettua una descrizione del progetto e viene sottoposta al customer in modo tale che venga certificata la fondatezza dei requisiti principali. Lavorando con prototipi il team si assicura che le evoluzioni siano di poco conto rispetto alla struttura principale
Conflitti all'interno del team	BASSO	IMPROBABILE	Vedute progettuali diverse si ripercuotono sulla riuscita del software	Per ovviare a questo rischio il team avrà dei compiti ben precisi e non potrà entrare in merito al compito altrui, in quanto la scelta iniziale del team è operare secondo moduli. Ogni modulo concluso verrà accorpato ad altri per la creazione finale del software. Il team congiuntamente valuta la software description studia come fare ma poi opera come già detto secondo moduli.

4. First Iteration (Prototype 1)

Trattandosi del primo prototipo, si parte dapprima da una descrizione ad alto livello delle caratteristiche del software, per poi procedere per raffinamenti successivi.

Occorre innanzitutto effettuare un'accurata analisi dei requisiti, in modo tale da poter avviare lo sviluppo del software verso la giusta direzione.

4.1 Requirements Phase

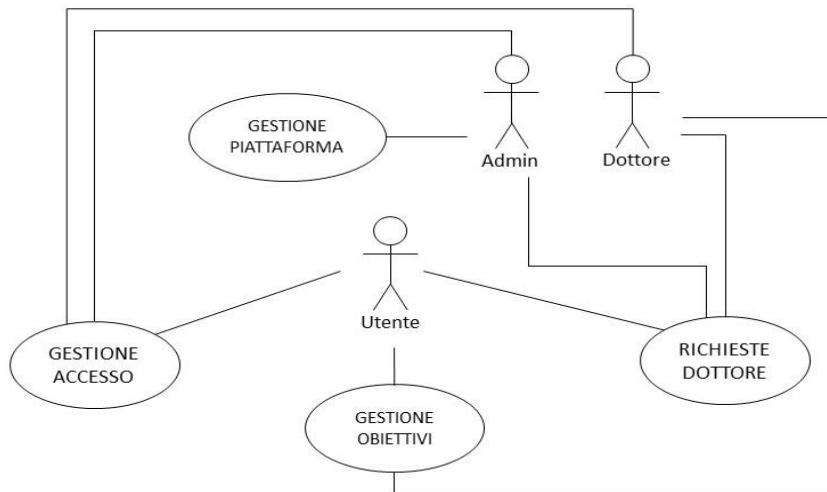


Figura 5 Use case diagram (Simplified Version)

Il software dovrà permettere all'**utente** di eseguire le seguenti macro-funzionalità:

- Gestione Accesso
- Gestione Obiettivi
- Richieste dottore

La GESTIONE ACCESSO è la funzionalità che permette di visualizzare le proprie informazioni personali e soprattutto dovrà occuparsi della creazione e accesso al profilo.

La GESTIONE DEGLI OBIETTIVI deve prevedere che l'utente in base alle proprie attività e caratteristiche, trovi gli obiettivi giornalieri distribuiti nei vari momenti della giornata ma dovrà esserci la possibilità di avere un resoconto, ovvero di tenere traccia gli obiettivi passati, con i relativi alimenti assunti e dunque lo storico dei valori nutrizionali assegnati dal software e quelli realmente assunti. Anche il dottore “entra in gioco” nella gestione degli obiettivi, poiché può inserire o eliminare alimenti presenti nel software in modo tale che l'utente possa avere una vasta scelta di alimenti e soprattutto opportuni.

L'utente dovrà poter relazionarsi con i dottori, che sono elementi di supporto in caso di eventi prettamente nutrizionali mediante le RICHIESTE DOTTORE e tale funzionalità dovrà prevedere che venga implementata una messaggistica per poter avere dei confronti attenti e inoltre nel caso in cui l'utente non trova un alimento all'interno della piattaforma può suggerirlo in modo tale che venga successivamente convalidato e correlato dei propri valori nutrizionali (come precedentemente annunciato nella gestione degli obiettivi).

Il software dovrà permettere all'**admin** la “Gestione della Piattaforma” e il “Controllo operato lista”:

La GESTIONE DELLA PIATTAFORMA dovrà permettere all'admin di poter eliminare utenti e dottori dalla piattaforma e in relazione sempre ai dottori anche l'inserimento. Come già ribadito dovrà essere compito del gestore della piattaforma assumere dottori. Non ha solo questo compito, ma sarà compito sempre di questa tipologia di utente monitorare il lavoro dei dottori, semplicemente andando a visualizzare gli esiti delle richieste (che possono essere alimenti convalidati o meno oppure il giudizio positivo o negativo di un utente in funzione di una determinata richiesta), che permetteranno al gestore di fare valutazioni attente sui dottori della piattaforma. Il CONTROLLO OPERATO LISTE permette all'admin di andare a valutare la laboriosità dei dottori.

Il software dovrà permettere ad un **dottore** di eseguire le seguenti macro-funzionalità:

- Risolvere richieste

Il dottore, tipologia di utente fondamentale, dovrà avere la funzionalità di RISOLVERE LE RICHIESTE degli utenti in modo tale da essere un riferimento per chi è in cerca di un individuo qualificato nell'ambito nutrizionale. Dovrà essere in grado di aggiungere cibi all'interno della piattaforma, anche grazie ai suggerimenti che gli utenti forniranno, ma anche rispondere a domande generiche poste dagli utenti. L'utente bisognoso di informazioni o aiuto dovrà ricevere il massimo supporto da questa tipologia di utente.

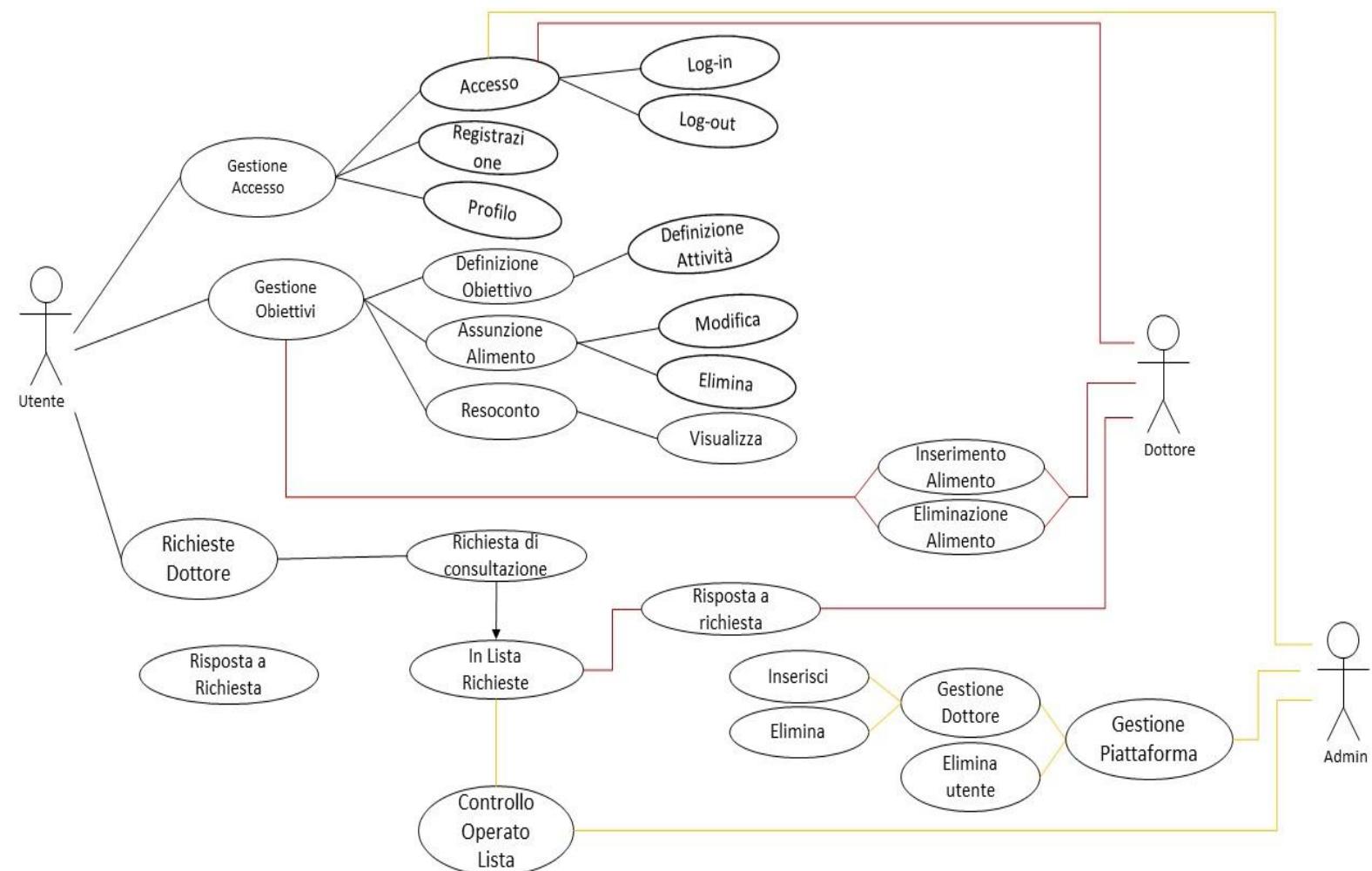


Figura 6 – Use Case Diagram (Extended version)

GESTIONE ACCESSO (tipologia utente:**UTENTE**)

L'utente utilizza questa operazione per registrarsi e accedere alla piattaforma e visualizzare le proprie informazioni

Precondizioni

Nessuna

Descrizione dell'operazione

L'operazione GESTIONE UTENTE è caratterizzata da tre principali sotto-operazioni:

La REGISTRAZIONE permette la creazione di un utente alla piattaforma.

L' ACCESSO permette tramite **log-in** e **log-out** di poter accedere o uscire dalla piattaforma ogni qualvolta che lo si voglia.

L'operazione accessibile dall'utente denominata PROFILO permette di vedere le informazioni personali, sia quelle inserite al momento della registrazione e sia quelle inserite durante l'utilizzo del software

GESTIONE ACCESSO (tipologia utente:**ADMIN**)

L'utente utilizza questa operazione per accedere alla piattaforma

Precondizioni

Inserito in fase di sviluppo dal team

Descrizione dell'operazione

L'operazione GESTIONE UTENTE è caratterizzata da una principali sotto-operazione.

L' ACCESSO permette tramite **log-in** e **log-out** di poter accedere o uscire dalla piattaforma ogni qualvolta che lo si voglia.

GESTIONE ACCESSO (tipologia utente:**DOTTORE**)

L'utente utilizza questa operazione per accedere alla piattaforma e visualizzare le proprie informazioni

Precondizioni

Inserito dall'admin dal pannello di “inserimento dottori”

Descrizione dell'operazione

L'operazione GESTIONE UTENTE è caratterizzata da tre principali sotto-operazioni.

L' ACCESSO permette tramite **log-in** e **log-out** di poter accedere o uscire dalla piattaforma ogni qualvolta che lo si voglia.

L'operazione accessibile dall'utente denominata PROFILO permette di vedere le informazioni personali, sia quelle inserite al momento della registrazione e sia quelle inserite durante l'utilizzo del software

GESTIONE OBIETTIVI (Tipologia utente: **UTENTE**)**Precondizioni**

L'utente è registrato e solo questa tipologia di utente avrà accesso a questo servizio

Descrizione dell'operazione

L'operazione GESTIONE OBIETTIVI è caratterizzata da tre sotto-operazioni:

La DEFINIZIONE OBIETTIVO è il primo passo che un utente compie, potrà scegliere tra mantenimento peso, perdita peso e sviluppo muscolare. Successivamente sarà compito dell'utente andare a **definire l' attività** che comunemente compie, in modo tale da definire gli input per il calcolo del fabbisogno calorico. Dunque alla fine della definizione bisognerà aggiungere gli alimenti che verranno assunti durante la giornata (ASSUNZIONE ALIMENTI) in modo tale da poter fare un calcolo sulle calorie assunte. Un alimento inserito può essere **modificato** nelle quantità o **eliminato** in caso di errore. La terza ed ultima operazione della gestione degli obiettivi è il RESOCONTO che avrà il compito di **visualizzare** lo storico degli obiettivi e se quotidianamente si raggiungono tali obiettivi.

GESTIONE OBIETTIVI (Tipologia utente: **DOTTORE**)

Il dottore utilizza quest'operazione per chiedere supporto agli obiettivi, inserendo ed eliminando alimenti dalla piattaforma

Precondizioni

Il dottore è registrato e solo questa tipologia di utente avrà accesso a questo servizio

Descrizione dell'operazione

L'operazione GESTIONE OBIETTIVI è caratterizzata da due sotto-operazioni:

L'INSERIMENTO ALIMENTI permette di inserire un alimento mancante all'interno della piattaforma. Come visto per l'utente, un alimento va inserito quotidianamente dal momento che viene assunto e se non è presente l'utente può richiedere l'inserimento (lo vedremo in seguito) o comunque un dottore può inserirlo arbitrariamente senza che ci sia necessariamente una richiesta. Inoltre il dottore gestisce anche l'ELIMINAZIONE ALIMENTI che è la funzione al reciproco della precedente e dunque elimina un alimento che, per un qualsiasi motivo, il dottore ritiene inopportuno.

RICHIESTE DOTTORE (Tipologia utente: **UTENTE**)

L'utente utilizza quest'operazione per chiedere supporto ad un dottore

Precondizioni

L'utente è registrato

Descrizione dell'operazione

L'operazione RICHIESTE DOTTORE è caratterizzata da una sotto-operazione:

La RICHIESTA DI CONSULTAZIONE è l'operazione che l'utente utilizzerà nel caso in cui voglia fare delle richieste ad un dottore. Tale operazione avrà due possibili scenari, ovvero generare e dunque inviare una richiesta o leggere un eventuale risposta da parte di un dottore sempre in seguito ad una richiesta. All'invio di una richiesta viene generata una lista di richieste, per il determinato utente che le ha effettuate in base all'"oggetto" scelto, che permetteranno di sapere lo stato e dunque le risposte fornite dal dottore. La richiesta di consultazione avrà dei possibili "oggetto": "non riesco a dimagrire", "consiglio su cosa assumere", "suggerimento alimento mancante".

RICHIESTE DOTTORE (Tipologia utente: DOTTORE)

L'utente utilizza quest'operazione per rispondere alle richieste.

Precondizioni

L'utente è registrato

Descrizione dell'operazione

L'operazione RICHIESTE DOTTORE è caratterizzata da una sotto-operazione:

La RICHIESTA DI CONSULTAZIONE è l'operazione che l'utente utilizzerà per visualizzare le richieste di consultazione in sospeso e rispondere. Per evitare confusione verranno divise in base alla categoria (oggetto) scelta dall'utente al momento dell'invio. Nel caso l'utente proponga il suggerimento di un alimento sarà compito del dottore valutare l'alimento ed eventualmente inserirlo(tramite l'opportuna operazione), in seguito alla decisione di convalida, il dottore risponderà all'utente con la decisione presa per quel suggerimento. Nel caso di richieste generiche, sarà compito del dottore rispondere in relazione al proprio ruolo e alle proprie competenze.

CONTROLLO OPERATO LISTE

L'utente admin utilizza quest'operazione per visionare gli esiti delle richieste poste ai dottori

Precondizioni

Nessuna

Descrizione dell'operazione

L'operazione CONTROLLO OPERATO LISTE avrà come campo di applicazione, la lista delle richieste di consultazioni.

Questa, consentirà all'admin, di visualizzare gli esiti delle richieste dei suggerimenti (convalidato o non convalidato) e di sapere a quante domande i dottori hanno risposto e la valutazione data dall'utente a quella risposta. Questa operazione messa a disposizione all'admin andrà a fornire informazioni utili per sapere se i dottori della piattaforma sono funzionali al loro ruolo e soprattutto efficienti.

GESTIONE PIATTAFORMA

L'utente admin utilizza quest'operazione per gestire gli utenti della piattaforma

Precondizioni

Nessuna

Descrizione dell'operazione

L'operazione GESTIONE PIATTAFORMA è caratterizzata da due sotto-operazioni:

L'operazione ELIMINA UTENTE permetterà ad un admin di poter effettuare un “ban” nei confronti di un utente e quindi cancellarne tutti i dati e di non lasciare traccia della propria presenza all'interno della piattaforma.

L'operazione GESTIONE DOTTORE permetterà all'admin di inserire o eliminare un dottore dalla piattaforma.

4.1.1 Functional Requirements

Riassumendo, è possibile individuare i seguenti requisiti funzionali, ovvero quei requisiti che descrivono cosa il sistema deve fare (le sue funzionalità), in termini delle azioni che il prodotto deve svolgere.

RQ1.0: GESTIONE ACCESSO(UTENTE)

- RQ1.0.1:** Regista utente
- RQ1.0.2:** Login
- RQ1.0.3:** Logout
- RQ1.0.4:** Profilo

RQ1.1: GESTIONE ACCESSO (ADMIN)

- RQ1.1.1:** Login
- RQ1.1.2:** Logout

RQ1.2: GESTIONE ACCESSO (DOTTORE)

- RQ1.2.1:** Login
- RQ1.2.2:** Logout
- RQ1.2.3:** Profilo

RQ2.0: GESTIONE OBIETTIVI(UTENTE)

- RQ2.0.1:** Definizione obiettivo
- RQ2.0.2:** Assunzione alimenti
- RQ2.0.3:** Resoconto

RQ2.1: GESTIONE OBIETTIVI(DOTTORE)

- RQ2.1.1:** Inserimento alimento
- RQ2.1.2:** Eliminazione alimento

RQ3.0: RICHIESTE DOTTORE(UTENTE)

- RQ3.0.1:** Richiesta di consultazione(invio)
- RQ3.0.2:** Presa visione e valutazione della risposta del dottore.

RQ3.1: RICHIESTE DOTTORE(DOTTORE)

- RQ3.1.1:** Richiesta di consultazione(risposta)
- RQ3.1.2:** Inserimento alimenti in seguito a richiesta

RQ4: CONTROLLO OPERATO LISTE

- RQ4.1:** Controllo richieste e feedback

RQ5: GESTIONE PIATTAFORMA

- RQ5.1:** Elimina utente
- RQ5.2:** Inserisci dottore
- RQ5.3:** Elimina dottore

4.1.2 Non-Functional Requirements

I requisiti non funzionali descrivono le qualità del software e i vincoli che esso deve rispettare.

RQ-NF1: Compatibilità con diverse piattaforme

RQ-NF2: Interfaccia GUI user-friendly

RQNF3: Possibilità di accedere al proprio piano alimentare da più dispositivi.

RQNF4: Utilizzo di una database per la memorizzazione dei dati

RQNF5: Basso carico d'elaborazione da parte del dispositivo

4.1.3 Matrix of Functional Requirements

NOME REQUISITO	ID	TIPO	PRIORITÀ	DESCRIZIONE	REQUISITO PADRE	REQUISITO FIGLIO
Gestione accesso	1.0	funzionale	1	Deve consentire la gestione degli account utenti		1.0.1-2-3-4
Registra utente	1.0.1	funzionale	1	Deve consentire all'utente di creare un account	1.0	
Login utente	1.0.2	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.0	
Logout utente	1.0.3	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.0	
Profilo utente	1.0.4	funzionale	1	Deve consentire all'utente di vedere le informazioni personali	1.0	
Gestione accesso	1.1	funzionale	1	Deve consentire la gestione degli account utenti		1.1.1-2
Login admin	1.1.1	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.1	
Logout admin	1.1.2	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.1	
Gestione accesso	1.2	funzionale	1	Deve consentire la gestione degli account utenti		1.2.1-2-3
Login dottore	1.2.1	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.2	
Logout dottore	1.2.2	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.2	
Profilo dottore	1.2.3	funzionale	1	Deve consentire all'utente di vedere le informazioni personali	1.2	
Gestione obiettivi	2.0	funzionale	2	Deve consentire all'utente di pianificare e gestire i propri obiettivi		2.0.1-2-3
Definizione obiettivo	2.0.1	funzionale	2	Deve consentire all'utente di definire l'obiettivo per il quale usa il software	2.0	
Assunzione alimenti	2.0.2	funzionale	2	Deve consentire all'utente di inserire gli alimenti assunti quotidianamente	2.0	
Rresoconto	2.0.3	funzionale	2	Deve consentire all'utente di visualizzare lo storico degli obiettivi passati	2.0	

NOME REQUISITO	ID	TIPO	PRIORITÀ	DESCRIZIONE	REQUISITO PADRE	REQUISITO FIGLIO
Gestione obiettivi	2.1	funzionale	2	Deve consentire al dottore di gestire gli alimenti della piattaforma		2.1.1-2
Inserimento alimenti	2.1.1	funzionale	2	Deve consentire al dottore di inserire alimenti alla piattaforma	2.1	
Eliminazione alimenti	2.1.2	funzionale	2	Deve consentire al dottore di eliminare alimenti dalla piattaforma	2.1	
Richieste dottore	3.0	funzionale	3	Deve consentire all'utente di poter avere un confronto con un dottore		3.0.1-2
Richiesta di consultazione (invio)	3.0.1	funzionale	3	Deve consentire all'utente di poter chiedere ad un dottore, un qualsiasi consulto specialistico	3.0	
Presenza e valutazione risposta del dottore	3.0.2	funzionale	3	Deve consentire all'utente di visualizzare la risposta del dottore e dare un feedback.	3.0	
Richieste dottore	3.1	funzionale	3	Deve consentire al dottore di poter rispondere alle richieste di un utente		3.1.1-2
Richiesta di consultazione (risposta)	3.1.1	funzionale	3	Deve consentire al dottore di rispondere ad una generica richiesta da parte di un utente	3.1	
Controllo operato liste	4	funzionale	4	Deve consentire all'admin di poter controllare gli esiti delle richieste fatte ai dottori		4.1
Controllo richieste e feedback	4.1	funzionale	4	Deve consentire all'admin di visualizzare l'esito di una richiesta. Viene visualizzato il feedback dato dall'utente in modo tale da valutare la laboriosità del dottore	4	
Gestione piattaforma	5	funzionale	5	Deve consentire all'admin di effettuare operazioni sulla piattaforma		5.1-2-3
Elimina utente	5.1	funzionale	5	Deve consentire all'admin di eliminare un utente e i suoi dati	5	
Inserisci dottore	5.2	funzionale	5	Deve consentire all'admin di inserire un dottore alla piattaforma	5	
Elimina dottore	5.3	funzionale	5	Deve consentire all'admin di eliminare un dottore e i suoi dati	5	

4.2 Design Phase

Per la fase di progettazione del prototipo si utilizza una filosofia top-down. Si definiscono quindi dappriama le componenti principali, specificando la loro interazione, andando poi ad approfondirle nel dettaglio in seguito.

4.2.1 Software Logical Structure

Come si può notare già con l'analisi dei requisiti, a livello logico il software può essere suddiviso nelle seguenti parti:

1. GESTIONE ACCESSO
2. GESTIONE OBIETTIVI
3. RICHIESTE DOTTORE
4. GESTIONE PIATTAFORMA

A questo punto può risultare utile descrivere in modo più o meno dettagliato le funzionalità delle diverse parti, in modo da dare una visione completa dell'interazione dei vari elementi al programmatore.

A tal fine si fa uso di alcuni State Diagram, che permettono di dare un'idea chiara del funzionamento logico che si prevede debba avere il programma finale in funzione degli utenti

GESTIONE ACCESSO

Una volta che un utente qualsiasi effettua l'accesso alla piattaforma dovrà identificarsi nella tipologia che lo contraddistingue, per la quale le sue credenziali funzionano. Scelta la tipologia gli verranno presentate alcune opzioni, nel caso dell'**admin** o del **dottore** sarà presente esclusivamente il login, nel caso si identifichi come un **utente** avrà la possibilità di effettuare la registrazione alla piattaforma ma nel caso fosse già registrato usufruirà del login ed accedere. Una volta effettuato l'accesso si potrà sempre effettuare il logout ed uscire dalla piattaforma.

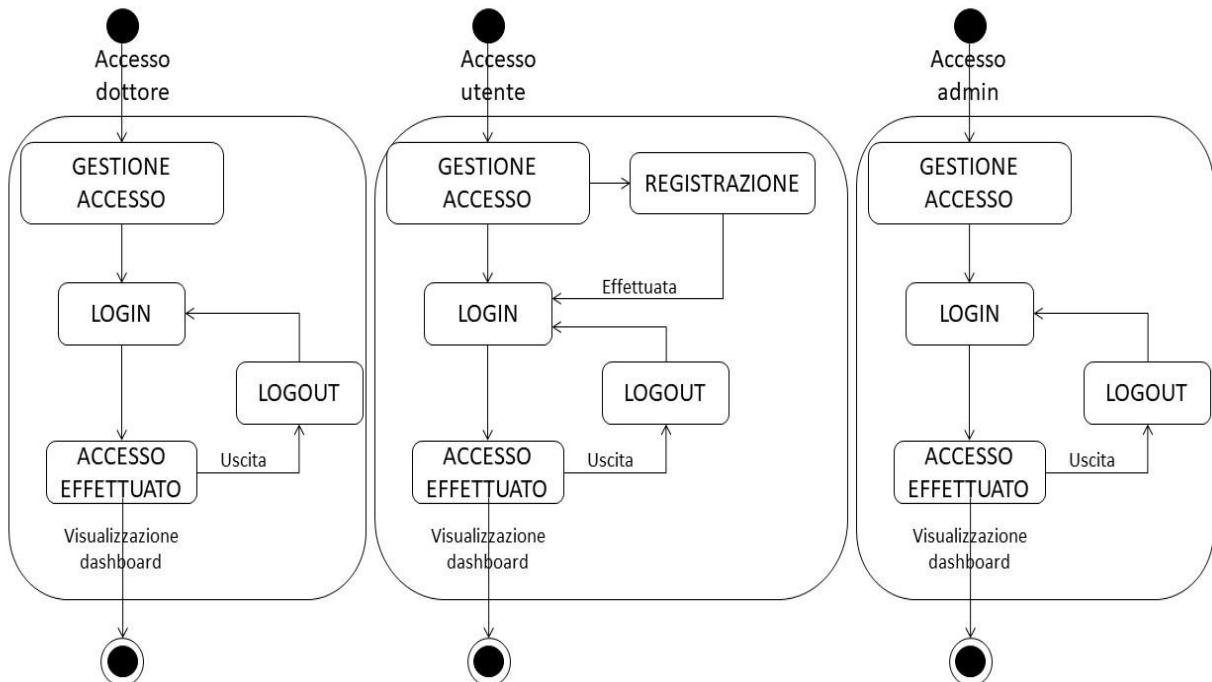
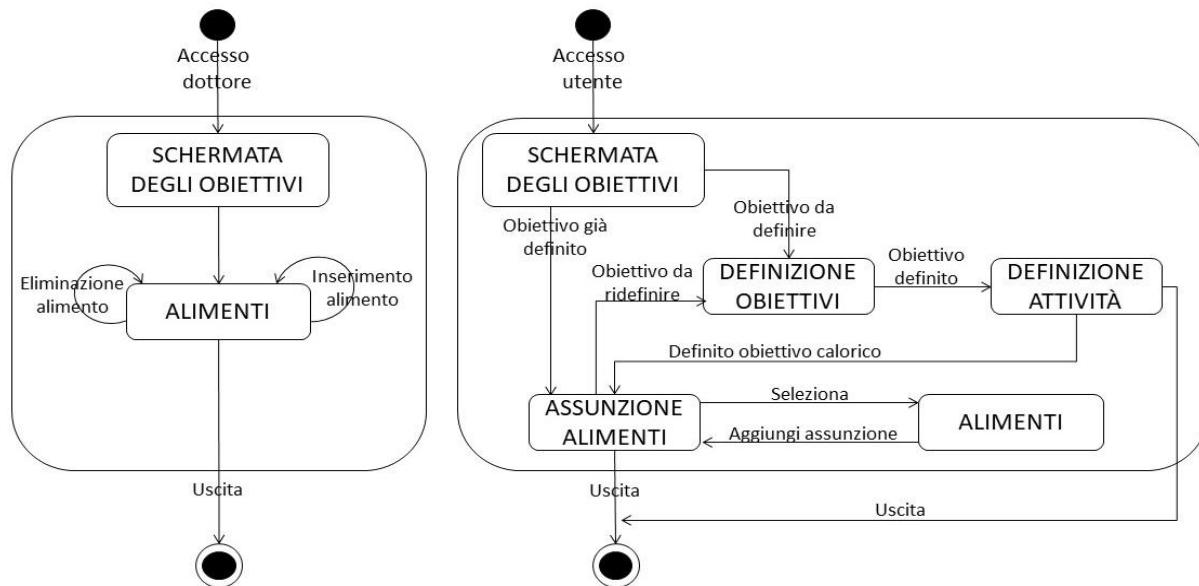


Figura 7 State diagrams: Gestione accesso in funzione del utente

GESTIONE OBIETTIVI

La gestione degli obiettivi passa dalla schermata degli obiettivi. Nel caso in cui non è stato mai definito mai un obiettivo, il software richiederà di definirlo (mantenimento, perdita o aumento muscolare) e successivamente verrà richiesto all'utente, che tipo di attività svolge solitamente. Queste informazioni genereranno l'obiettivo calorico giornaliero da seguire. Se l'obiettivo era stato già impostato o se definito sul momento come descritto pocanzi, sarà il momento di inserire gli alimenti assunti nel “diario”, in modo tale da tener traccia dei cibi se sono conformi agli obiettivi posti dal software. Il dottore aggiornerà la lista



degli alimenti della piattaforma mediante le operazioni di inserimento ed eliminazione.

Figura 8 State diagrams: Gestione degli obiettivi in funzione dell'utente

RICHIESTE DOTTORE

La schermata delle richieste può avere diverse funzionalità in base all'utente che la utilizza, ogni utente genererà uno stato dell'esecuzione del programma differente. L'utente effettuerà le richieste nei confronti del dottore e quest'ultimo prenderà visione di tutte le richieste, inserite nella lista, in modo tale da risolverle. L'admin invece vedrà l'esito della risoluzione delle richieste in modo tale da valutare l'operato dei dottori. L'utente avendo formulato una richiesta può formularne altre ritornando allo stato “formula richiesta” o concludere le operazioni e terminare il suo operato.

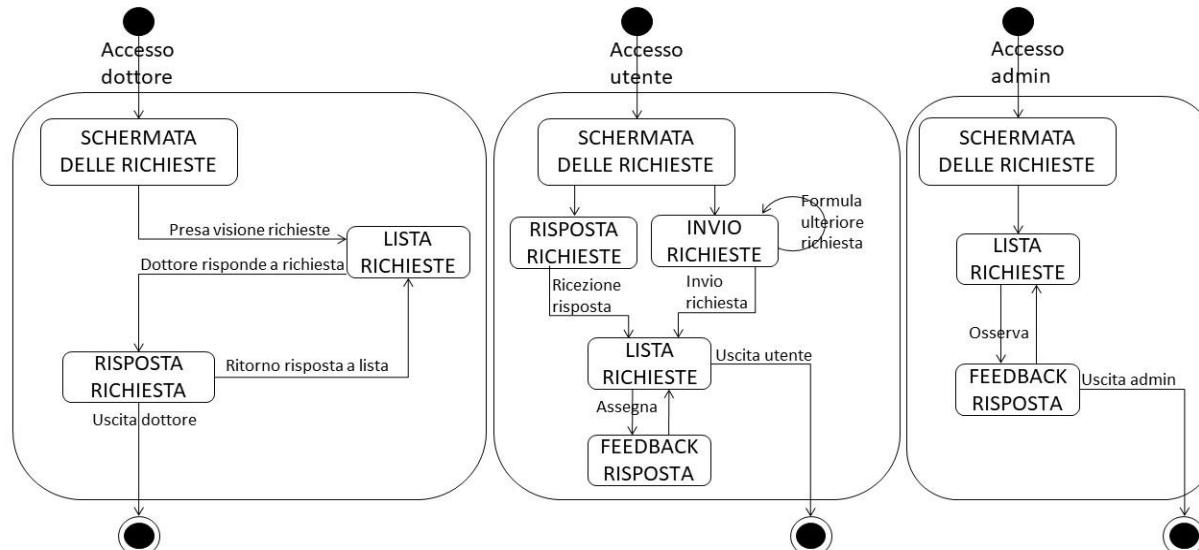


Figura 9 State diagrams : Richieste in funzione dell'utente

FIRST ITERATION (PROTOTYPE 1) – DESIGN PHASE

GESTIONE PIATTAFORMA (ADMIN)

La gestione della piattaforma è una operazione molto importante per l'admin, che avrà il compito di inserire o eliminare dottori dalla piattaforma. Un ulteriore funzionalità fondamentale, è l'eliminazione degli utenti e dunque effettuare un ban persistente di un generico utente.

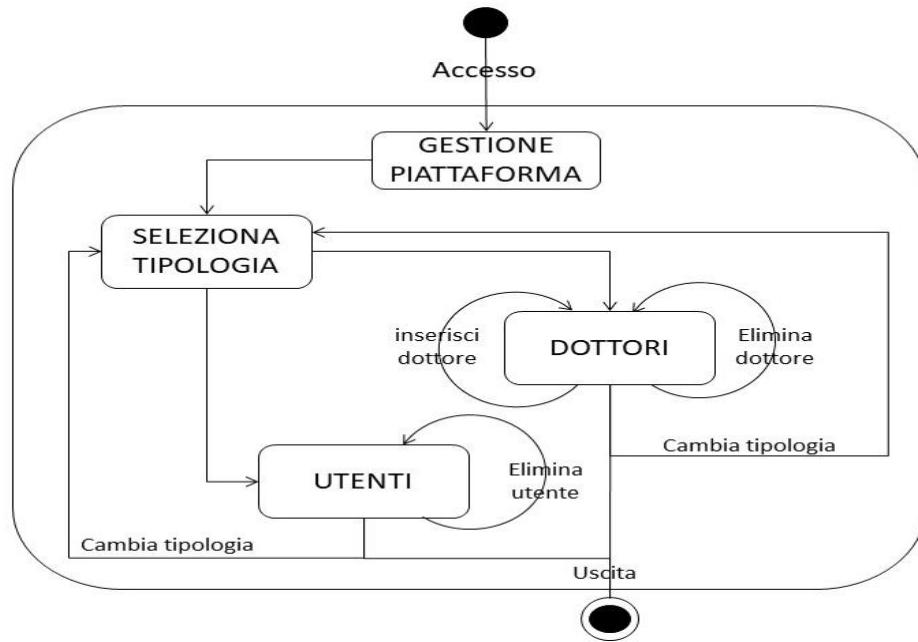


Figura 10 State diagram: Gestione piattaforma

4.2.2 Component Structure

L'architettura del software, in funzione delle specifiche poste a priori per la realizzazione del software, sarà di tipo client-server e risolverà i requisiti non funzionali visti precedentemente. In seguito verranno definite meglio la funzionalità di questa architettura.

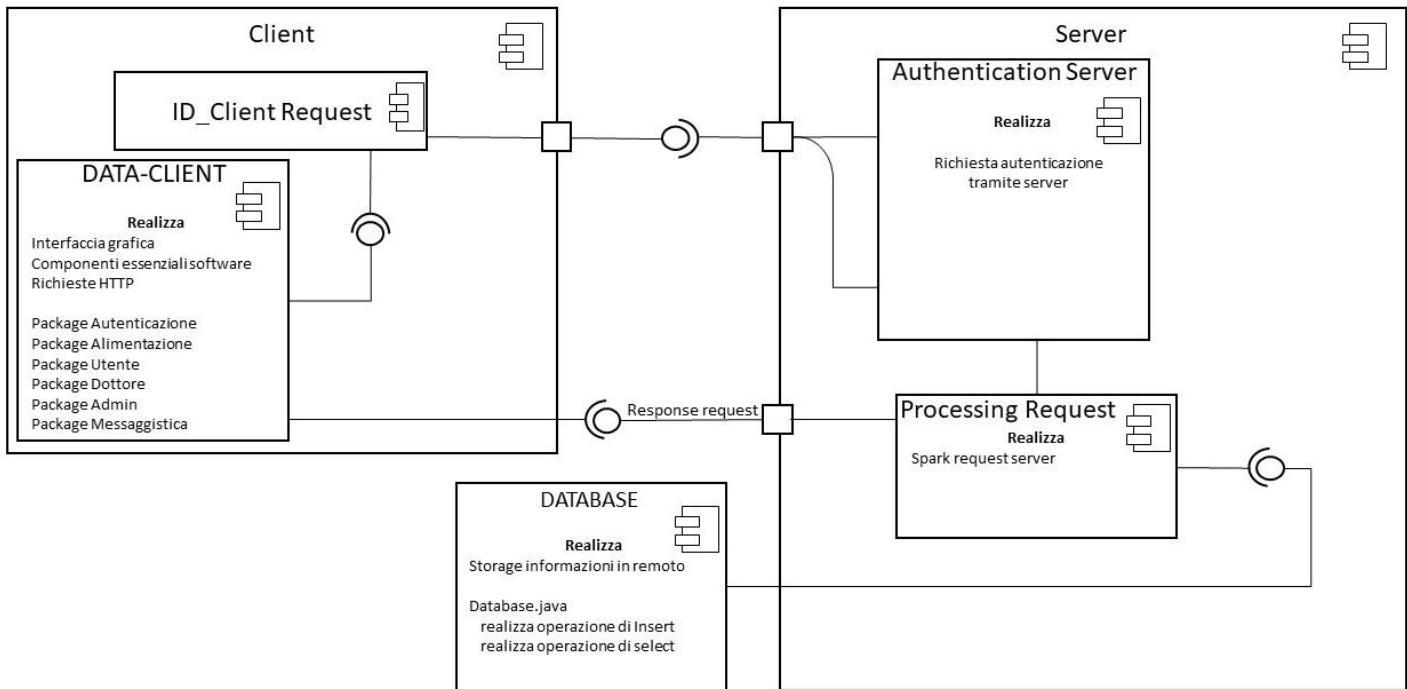


Figura 11 Component Diagram software (CLIENT-SERVER)

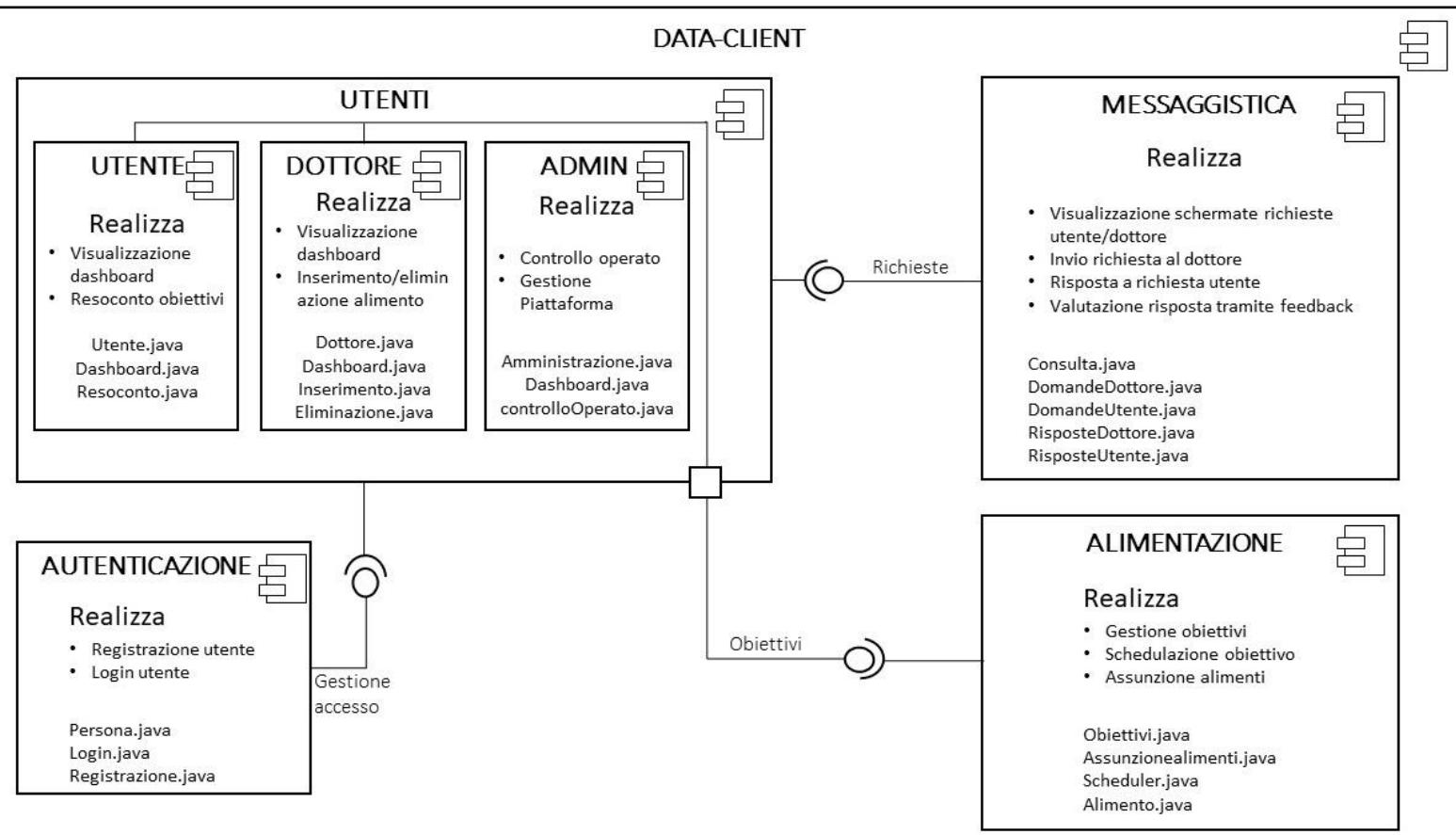


Figura 12 Component Diagram software (DATA-CLIENT)

4.2.3 System Architecture

Per quanto riguarda innanzitutto la definizione dell'architettura, quest'ultima come già anticipato sarà un'architettura di tipo **client-server**, con la possibilità di diverse connessioni client in contemporanea.

L'infrastruttura di rete che si mira ad utilizzare, sarà la rete **Internet**.

Il ruolo centrale del server sarà quello di creare diversi thread per ogni connessione, in modo da poter gestire più richieste in contemporanea, di regolare l'accesso e la registrazione di ogni diverso utente, e di effettuare i calcoli necessari all'elaborazione del tipo di obiettivo in maniera completamente trasparente all'utente, non solo per quanto riguarda il tipo di algoritmo utilizzato, ma anche dal punto di vista dell'efficienza, che verrà così astratta dal tipo di computer utilizzato dal client.

Ciò permetterà di avere diversi vantaggi e qualche svantaggio:

PRO

- Possibilità di modificare, aggiungere o rimuovere differenti algoritmi di scheduling senza dover aggiornare il software lato client.
- Astrazione del tempo di calcolo richiesto rispetto al tipo di dispositivi client, in questo modo anche su dispositivi meno performanti il calcolo verrà effettuato nello stesso quantitativo di tempo, la differenza dipenderà solamente dalla velocità della connessione.
- Possibilità di visualizzare e catalogare eventuali statistiche relative all'utilizzo del software, da poter utilizzare a fini di analisi o commerciali.
- Possibilità per l'utente di accedere al proprio diario alimentare in ogni momento e da diversi dispositivi.
- Possibilità di sviluppare diverse versioni del software funzionanti su altri tipi di dispositivi, quali tablet e smartphone, dovendo agire solo sul lato client.

CONTRO

- Necessità di avere una connessione alla rete per utilizzare il software.
- Possibilità di disservizi nel caso di guasti al server o di congestione nella rete.
- Overhead del tempo richiesto, dovuto alla trasmissione e alla ricezione dei dati.

Dopo aver valutato le caratteristiche positive e negative di tale scelta, si è giunti con il alla conclusione che l'utilizzo di un tipo di architettura distribuita possa fornire un valore aggiunto al software, e si è dunque confermato l'avvio dello sviluppo con questa metodologia. Questa architettura da sviluppare sarà divisa in due parti, la parte del server e la parte del client che dovranno interagire tra di loro come vedremo in seguito.

4.2.4 Server

Il server sarà strutturato come un normalissimo server web: attenderà una richiesta di connessione da parte di un client sulla propria socket adibita al collegamento (che utilizzerà come porta di default la 4240) e, all'arrivo di una nuova richiesta di connessione, creerà una nuova socket di collegamento con il singolo client che verrà utilizzata per tutti i futuri messaggi.

Verrà creato un nuovo thread (**ClientThread**) che si occuperà della gestione di questi ultimi per il singolo client. Il tipo di collegamento sarà persistente, verrà dunque creata una connessione per ogni client e tutti i successivi messaggi verranno inviati su quel collegamento, fino al logout dell'utente.

Il thread di gestione dei messaggi ciclerà fino alla richiesta di chiusura connessione e rimarrà dunque in attesa dell'arrivo di un eventuale messaggio; in base al tipo di quest'ultimo, effettuerà le operazioni necessarie e invierà infine una risposta.

L'attesa del thread client non sarà attiva, essendo le chiamate ai metodi per la lettura di oggetti bloccanti, tuttavia nel caso di un gran numero di utenti collegati in contemporanea potrebbero sorgere problemi relativi alla terminazione dello spazio disponibile per i vari thread.

Nel caso in cui esigenze di affluenza lo richiedano comunque, sarà possibile modificare il tipo di connessione in una connessione non persistente, nel quale venga aperta una connessione per ogni messaggio, che, sebbene aggiunga dei ritardi relativi all'apertura e alla chiusura della connessione per ogni richiesta, permetterà di diminuire la memoria occupata nel server.

Essendo comunque stata preventivamente considerata tale evenienza, nel caso in cui si dovesse insorgere in tale problematica sarà possibile modificare il tipo di connessione semplicemente inviando un messaggio di chiusura connessione alla fine di ogni messaggio, dovendo dunque modificare un numero minimo di righe di codice.

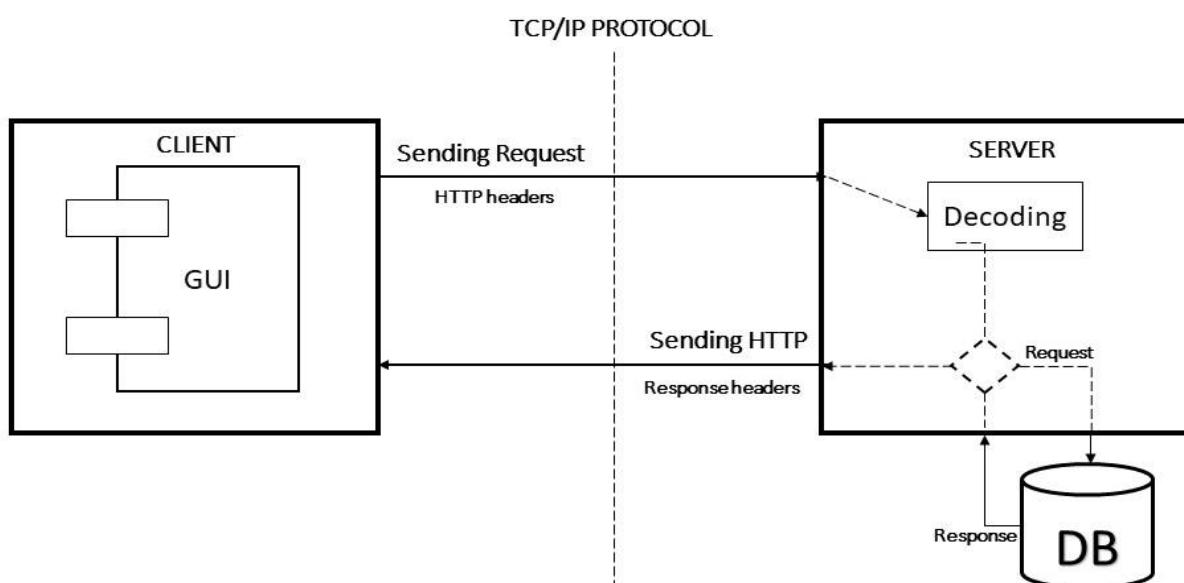


Figura 13 Architettura Client-Server

4.2.5 Communication Protocol http

Per quanto riguarda il protocollo da utilizzare per la comunicazione tra client e server, si è scelto l'impiego del protocollo http, che è il protocollo che sta alla base delle architetture software basate su tale metodologia.

Il protocollo prevede che il Client esegue una richiesta e il server restituisce la risposta, dunque a queste due operazioni susseguono numerosi scambi di messaggi tra i due componenti.

Per via della natura intrinseca del problema, l'unico tipo di protocollo di comunicazione utilizzabile è un protocollo sincrono, nel quale il client attende la risposta del server alla propria richiesta. Per questo motivo ogni messaggio inviato è di tipo bloccante per l'utente, che si ritroverà dunque ad attendere la relativa risposta da parte del server (fatta eccezione per il messaggio di chiusura connessione, il quale verrà inviato subito prima di effettuare il logout, in modo che il Client non rimanga bloccato nel caso di disservizi alla rete).

Per quanto riguarda il protocollo di trasporto, essendo fondamentale l'arrivo a destinazione dei dati così come questi sono stati inviati. Verrà utilizzato il protocollo TCP, il protocollo sicuro di trasporto orientato alla connessione

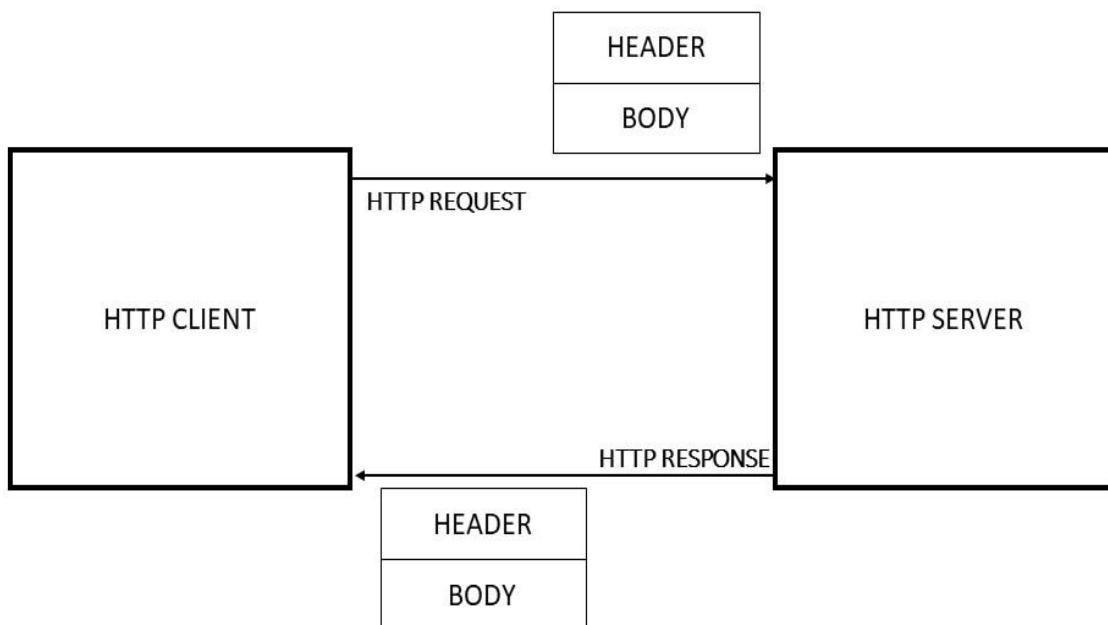


Figura 14 Protocollo HTTP (Request/Response)

4.2.6 GUI Design

In questa fase si attua una progettazione ad alto livello della GUI, ovvero dell'interfaccia grafica fornita all'utente (front-end).

L'obiettivo è quello di dare una descrizione di massima della struttura delle diverse schermate, senza vincoli eccessivi per i programmatore.

DASHBOARD (UTENTE)

La dashboard del software dovrà essere semplice e chiara. L'idea di base è quella di 5 bottoni, disposti verticalmente, attraverso i quali, all'utente, sarà possibile svolgere le operazioni principali del programma.

1. Il bottone “OBIETTIVI” consentirà all'utente di visualizzare la gestione degli obiettivi, quindi definire nuovi obiettivi o nel caso siano stati già definiti andare a inserire gli alimenti assunti
2. Il bottone “RESOCOMTO” sarà utilizzato per accedere alla schermata corrispondente, al cui interno saranno presenti gli obiettivi delle giornate trascorse. Al suo interno potremo vedere l'andamento del nostro diario e valutare se si stanno seguendo gli obiettivi posti dal software.
3. Il bottone “GESTIONE UTENTE” permetterà all'utente di accedere alla rispettiva schermata di gestione utente con le informazioni personali.
4. Il bottone “RICHIESTE DOTTORE” permetterà all'utente di accedere alla rispettiva schermata, in modo tale da poter effettuare richieste al dottore per avere un consulto. In seguito alla risposta di un dottore, l'utente dovrà valutare la risposta dando un feedback.

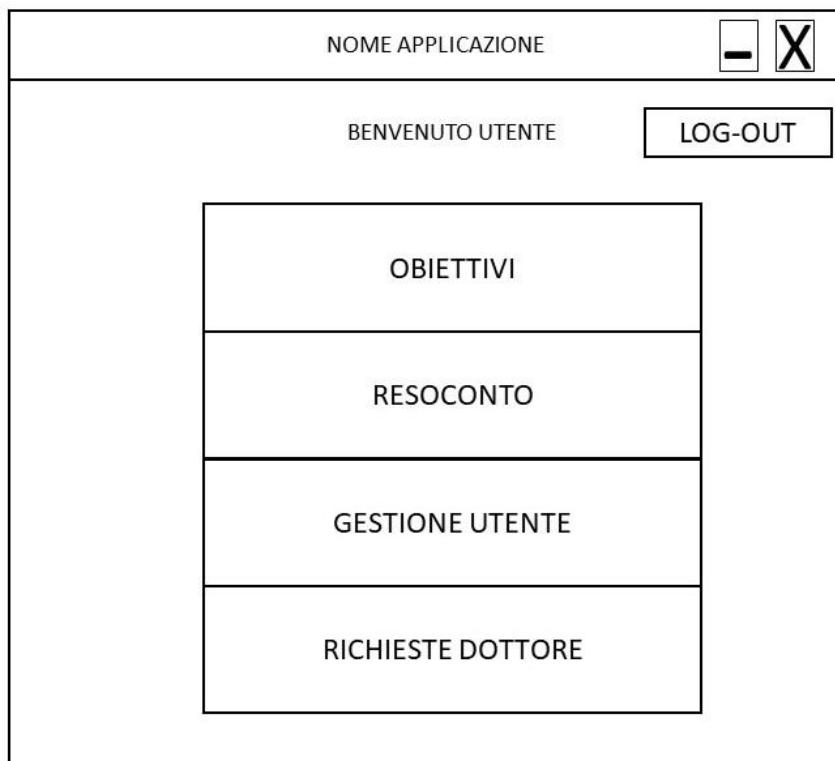


Figura 15 Interfaccia: Dashboard del software(UTENTE)

DASHBOARD (DOTTORE)

La dashboard del software dovrà essere semplice e chiara. L'idea di base è quella di 2 bottoni, disposti verticalmente, attraverso i quali, al dottore, sarà possibile svolgere le operazioni principali del programma.

1. Il bottone “GESTIONE OBIETTIVI” consentirà al dottore di visualizzare la gestione degli obiettivi , e nel caso di questa tipologia di utente sarà permesso intervenire sugli alimenti presenti nella piattaforma e dunque inserimento ed eliminazione di alimenti.
2. Il bottone “RICHIESTE” sarà utilizzato per accedere alla schermata corrispondente, al cui interno saranno presenti tutte le richieste inviate dagli utenti e dunque all'interno troveremo anche il modo di effettuare la risposta.

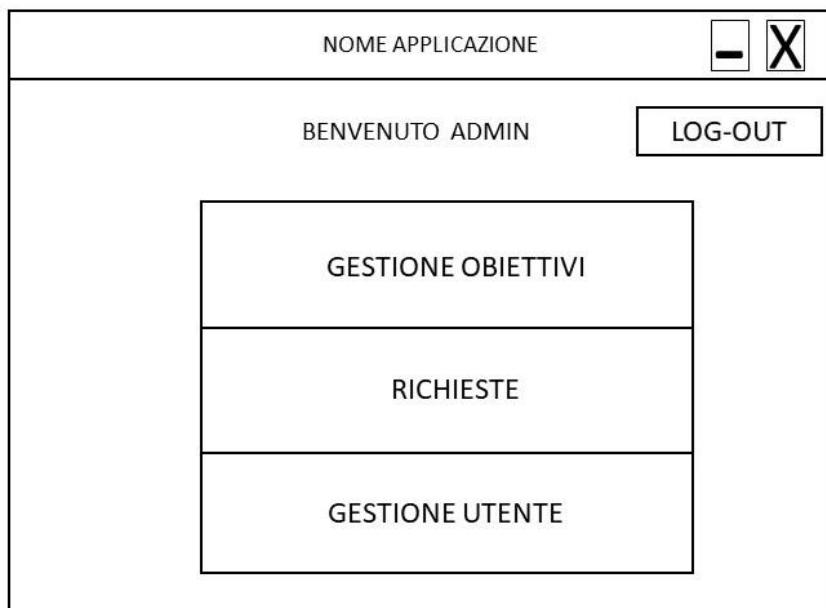
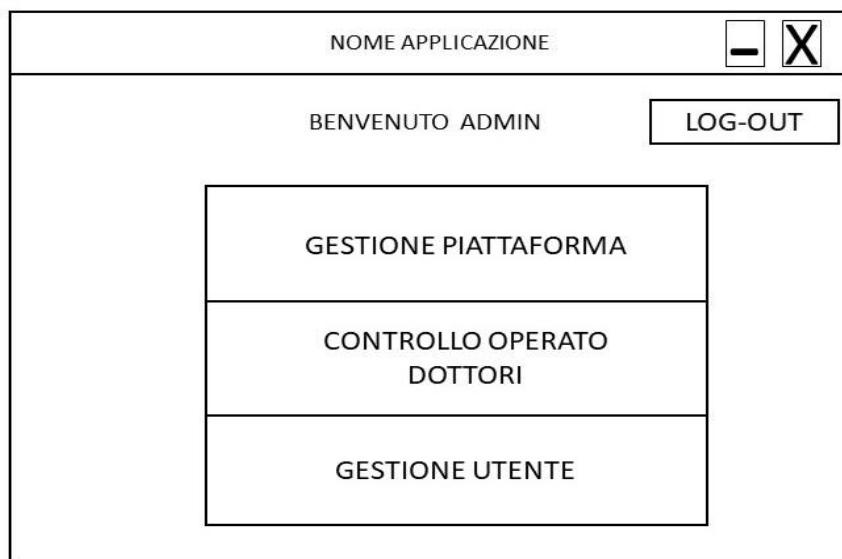


Figura 16 Interfaccia: Dashboard del software(DOTTORE)

DASHBOARD (ADMIN)

La dashboard del software dovrà essere semplice e chiara. L'idea di base è quella di 2 bottoni, disposti verticalmente, attraverso i quali, all'admin, sarà possibile svolgere le operazioni principali del programma.

1. Il bottone “GESTIONE PIATTAFORMA” consentirà all'admin di accedere alla schermata corrispondente, al cui interno saranno presenti le opzioni di inserimento ed eliminazione di un dottore e l'operazione di eliminazione di un utente.
2. Il bottone “CONTROLLO OPERATO DOTTORI” consentirà all'admin di visualizzare i feedback che gli utenti lascieranno ogni qualvolta un dottore risponda ad una richiesta e nel caso in cui la suddetta richiesta, si tratti di un suggerimento di un alimento visualizzerà anche l'alimento convalidato.

**Figura 17** Interfaccia: Dashboard del software(ADMIN)

SCHERMATA OBIETTIVI(UTENTE)

Dalla schermata degli obiettivi l'utente deve poter, innanzitutto, andare a delineare l'obiettivo che vuole raggiungere tramite l'utilizzo del software. Tre input dettati dall'utente, permetteranno una schedulazione più approfondita dell'alimento da assumere:

- Selezionare l'obiettivo da raggiungere
- Selezionare il livello di attività lavorativa
- Selezionare l'attività sportiva settimanale svolta

Figura 18 Interfaccia: Definizione obiettivo

Successivamente, il software avendo, autonomamente, svolto il proprio algoritmo dovrà permettere, tramite l'opportuna schermata di inserire gli alimenti nelle rispettive time-section giornaliere. La figura successiva mostrerà come l'utente visualizzerà questa schermata e le operazioni che può svolgere:

- Cambiare l'obiettivo precedentemente definito, tornando alla figura precedente
- Inserire un alimento in una time-section (colazione nel caso della figura)
- Eliminare un alimento precedentemente inserito.

The screenshot shows a mobile application window. At the top, there's a header bar with 'NOME APPLICAZIONE' and standard window controls (- and X). Below this is a section titled 'OBIETTIVI DI OGGI' (Goals for Today). It displays nutritional goals: Calorie (0/tot kcal), Carboidrati (0/tot g), Proteine (0/tot g), and Grassi (0/tot g). A horizontal bar indicates the current meal period: 'Colazione' (Breakfast) is selected, while 'Spuntino' (Snack), 'Pranzo' (Lunch), and 'Cena' (Dinner) are shown in smaller boxes. Below this, there's a summary of total calories to consume and actual consumed calories. A table lists an item: Mela (Apple) with 100g, 48kcal, 14g carbohydrates, 0g protein, and 0g fat. At the bottom are three buttons: 'CAMBIA OBIETTIVO' (Change Goal), 'INSERISCI ALIMENTO' (Add Food), and 'ELIMINA ALIMENTO' (Delete Food).

Figura 19 Interfaccia: Schermata obiettivi

L'inserimento dell'alimento aprirà un'opportuna schermata al cui interno, come vedremo nella figura successiva, dovremo inserire il nome dell'alimento e ricercarlo in modo tale da trovare tutte le possibili corrispondenze all'interno del software. Nella figura viene posto un semplice esempio, in quanto per un alimento possono esserci diverse possibili varianti e l'utente dovrà selezionare ciò che ritiene corretto (risulta chiaro che se un alimento non è presente va richiesto al dottore, come già discusso nei capitoli precedenti). Dopo aver selezionato l'opportuno alimento va inserito in quale momento della giornata è stato assunto e la quantità espressa in grammi assunta.

The screenshot shows a mobile application window for adding food. At the top, there's a header bar with 'NOME APPLICAZIONE' and standard window controls (- and X). Below this is a section titled 'AGGIUNGI ALIMENTO' (Add Food). It has a search field labeled 'ALIMENTO' containing 'Prosciutto' and a 'Cerca' (Search) button. Below the search field is a dropdown menu showing 'Prosciutto crudo'. There's also a placeholder text 'Ricerca per: «Prosciutto»'. To the right of the search area is a 'PASTO' (Meal) dropdown showing 'Colazione' and a 'GRAMMI' (Grams) input field. At the bottom is a large button labeled 'INSERISCI ALIMENTO' (Add Food).

Figura 20 Interfaccia: Schermata assunzione alimento

SCHERMATA OBIETTIVI(DOTTORE)

Questa schermata deve permettere al dottore di svolgere le operazioni di eliminazione e inserimento di un alimento all'interno della piattaforma. Funzione necessaria affinchè all'aumentare del tempo e dell'utilizzo del software la lista di alimenti venga migliorata e incrementata.

The wireframe shows a user interface for managing food objectives. At the top, there's a header 'NOME APPLICAZIONE' with a close button ('X'). Below it is a section titled 'OBIETTIVI' containing a button 'INSERISCI ALIMENTO'. A row of buttons for inputting nutritional values follows: 'NOME', 'GRUPPO', 'CALORIE', 'CARBOIDRATI', 'PROTEINE', and 'GRASSI'. A large button labeled 'INSERISCI' is centered below these. Another section below is titled 'ELIMINA ALIMENTO' with a button 'ALIMENTO'. To its right is a table with four rows, each consisting of a column 'ALIMENTO' and a column 'VALORI NUTRIENTI'. To the far right of each row is a button labeled 'ELIMINA'. The entire interface is contained within a rectangular frame.

Figura 21 Interfaccia: Schermata obiettivi dottore

RICHIESTE DOTTORE(UTENTE)

Questa schermata deve permettere all'utente di interfacciarsi con un dottore, e quindi inviare le richieste e leggere le risposte.

The wireframe shows a user interface for interacting with a doctor. At the top, there's a header 'NOME APPLICAZIONE' with a close button ('X'). Below it is a section titled 'RICHIESTE DOTTORE'. Two main buttons are present: 'LEGGI RISPOSTE RICEVUTE' (Read Received Answers) and 'INVIA UNA RICHIESTA' (Send a Request). The entire interface is contained within a rectangular frame.

Figura 22 Interfaccia: Schermata richieste dottore (utente)

La figura successiva prende in considerazione la schermata che si attiva avendo premuto il bottone “INVIA UNA RICHIESTA”. L’utente avendo scelto una categoria di domanda (che fungerà da oggetto del messaggio), andrà a descrivere la propria problematica e finalizzerà l’invio mediante l’apposito bottone “INVIO”.

Figura 23 Interfaccia: Schermata invio richiesta dottore

La figura successiva prende in considerazione la schermata che si attiva avendo premuto il bottone “LEGGI RISPOSTE RICEVUTE”. L’utente vedrà tutte le richieste inviate con le relative informazioni : data di invio della richiesta,testo della richiesta inviata al dottore, il dottore che ha risposto alla domanda(nel caso non ci sia ancora la risposta il campo sarà vuoto), la risposta che il dottore ha fornito in seguito alla richiesta(nel caso non ci sia ancora la risposta il campo sarà vuoto) e per ultimo ci sarà il campo del feedback in cui l’utente darà una valutazione da 1 a 5 della risposta fornita.

DATA INVIO	RICHIESTA	DOTTORE	RISPOSTA	FEEDBACK
DATA INVIO	RICHIESTA	DOTTORE	RISPOSTA	FEEDBACK
DATA INVIO	RICHIESTA	DOTTORE	RISPOSTA	FEEDBACK
DATA INVIO	RICHIESTA	DOTTORE	RISPOSTA	FEEDBACK
DATA INVIO	RICHIESTA	DOTTORE	RISPOSTA	FEEDBACK

Figura 24 Interfaccia: Schermata richieste dottore

RICHIESTE DOTTORE(DOTTORE)

La seguente schermata permette al dottore di visualizzare le richieste da parte degli utenti. Le richieste sono suddivise per categorie in modo tale da non generare confusione ed avere le richieste il più ordinate possibile. Ad ogni richiesta il dottore apporrà la risposta nella apposita casella di testo.

LISTA RICHIESTE		
NON DIMAGRIMENTO	AIUTO SUGLI ALIMENTI	SUGGERIMENTO ALIMENTO MANCANTE
UTENTE	TESTO RICHIESTA	RISPOSTA

Figura 25 Interfaccia: Schermata richieste dottore (dottore)

RICHIESTE DOTTORE-CONTROLLO OPERATO(ADMIN)

La seguente schermata permette all'admin di visualizzare le valutazioni, da parte degli utenti, alle risposte dei dottori. Scegliendo un dottore, presente nella piattaforma, l'admin potrà vedere il totale di domande a cui quel determinato dottore ha risposto e inoltre vedrà questo valore distribuito nelle varie valutazioni (da 1 a 5). Inoltre potrà vedere il numero totale di richieste presenti nel software, e quelle a cui è stata data una risposta e quelle che al contrario ancora mancano di una risposta.

CONTROLLO OPERATO											
SCEGLI DOTTORE	INVIO										
TOTALE RISPOSTE	1	2	3	4	5						
TOT	TOT	TOT	TOT	TOT	TOT						
<table border="1"> <tr> <td>TOTALE RICHIESTE</td> <td>RICHIESTE ESPLETATE</td> <td>RICHIESTE DA ESPLETARE</td> </tr> <tr> <td>TOT</td> <td>TOT</td> <td>TOT</td> </tr> </table>						TOTALE RICHIESTE	RICHIESTE ESPLETATE	RICHIESTE DA ESPLETARE	TOT	TOT	TOT
TOTALE RICHIESTE	RICHIESTE ESPLETATE	RICHIESTE DA ESPLETARE									
TOT	TOT	TOT									

Figura 26 Interfaccia: Schermata controllo operato

GESTIONE PIATTAFORMA(ADMIN)

La schermata sottostante, permetterà all'admin di svolgere due sue operazioni fondamentali: l'inserimento di dottori e l'eliminazione di dottori e utenti.

The wireframe shows a window titled "NOME APPLICAZIONE" with a close button. Below it, the title "GESTIONE PIATTAFORMA" is centered. On the left, there are two buttons: "INSERISCI DOTTORE" and "ELIMINA UTENTI/DOTTORI". The "INSERISCI DOTTORE" section contains input fields for "NOME", "COGNOME", "NOME UTENTE", "ETÀ", "GENERE", and "PASSWORD", followed by a "INSERISCI" button. The "ELIMINA UTENTI/DOTTORI" section contains a dropdown menu labeled "SCEGLI UTENTE" and an "ELIMINA" button.

Figura 27 Interfaccia: Schermata gestione piattaforma

RESOCOMTO (UTENTE)

La schermata sottostante, mostrerà all'utente come, in funzione della data, ha affrontato gli obiettivi che il software gli ha posto. Vedrà i cibi e le quantità assunte ma anche i rispettivi valori dei macronutrienti assunti. Il resoconto farà il confronto tra i valori posti dal software come obiettivo e quelli cumulati durante i pasti giornalieri.

The wireframe shows a window titled "NOME APPLICAZIONE" with a close button. Below it, the title "RESOCOMTO" is centered. In the center, there is a large rectangular input field with a double-headed arrow icon at each end, labeled "DATA". Below this, there is a larger green-bordered rectangular area labeled "INFORMAZIONI OBIETTIVI".

Figura 28 Interfaccia: Schermata resoconto

GESTIONE ACCESSO

La schermata sottostante, mostrerà come ogni utente (definendone il tipo) inserirà le proprie informazioni per accedere alla piattaforma. La gestione dell'accesso nel caso esclusivo dell'utente disporrà di un ulteriore schermata: la schermata di registrazione.

The image displays two side-by-side wireframe prototypes for user access management. Both prototypes feature a header bar with 'NOME APPLICAZIONE' and standard window control buttons (minimize, maximize, close).

Left Prototype (User Login):

- NOME UTENTE:** Text input field.
- PASSWORD:** Text input field.
- ACCEDE COME:** A dropdown menu currently set to 'UTENTE'.
- ACCEDEI**: A large rectangular button.
- REGISTRAZIONE UTENTE**: A small rectangular button at the bottom right.

Right Prototype (User Registration):

- NOME UTENTE:** Text input field.
- NOME:** Text input field.
- CONOME:** Text input field.
- GENERE:** A dropdown menu currently set to 'UOMO'.
- ETÀ:** A dropdown menu currently set to '12'.
- ALTEZZA:** A horizontal slider with a circular marker.
- PESO:** A horizontal slider with a circular marker.
- PASSWORD:** Text input field.
- REGISTRATI**: A large rectangular button on the left.
- ACCEDEI**: A large rectangular button on the right.

Figura 29 Interfaccia: Schermate gestione accesso

GESTIONE UTENTE

La schermata sottostante, mostrerà la corrispondente gestione utente, ovvero andare a visualizzare le proprie informazioni personali ma soprattutto modificare la password.

The image shows a wireframe prototype for user profile management. It includes a header bar with 'NOME APPLICAZIONE' and standard window control buttons.

GESTIONE UTENTE: The title of the main section.

INFORMAZIONI UTENTE: A large rectangular area outlined in green, representing the user's personal information.

VECCHIA PASSWORD, **NUOVA PASSWORD**, **RIPETI PASSWORD**, and **CAMBIA**: Buttons located at the bottom of the screen, likely for password management.

Figura 30 Interfaccia: Schermata gestione utente

4.2.7 Implementation Guidelines

OBIETTIVO

Come detto in precedenza, l' **Obiettivo** deve essere giornaliero, e a sua volta viene suddiviso in **timeSection** (COLAZIONE, SPUNTINO MATTUTINO, PRANZO, SPUNTINO POMERIDIANO, CENA).

4.2.8 Algorithm of Goal's Scheduling

PRESUPPOSTI

- Ogni obiettivo giornaliero contiene uno o più alimenti;
- Ogni alimento viene fornito con il numero di calorie proprie;
- Ogni alimento viene fornito con il numero di proteine, carboidrati e grassi;
- Ogni timeSection fornisce un intervallo cronologico in cui è stato assunto l'alimento;
- Ogni timeSection è caratterizzata da un livello di calorie da assumere.

CRITERIO

L'obiettivo dell'algoritmo è quello di creare un diario alimentare. Il criterio utilizzato si basa sull'inserimento di determinate informazioni che serviranno da input per la creazione del fabbisogno calorico giornaliero. Il software analizzerà il peso, l'obiettivo posto in partenza e le attività svolte (fisiche e lavorative) per giungere alla conclusione e dunque l'assegnazione del fabbisogno calorico ma inoltre effettuerà lo "split" nei vari momenti della giornata e anche lo split dei macronutrienti per l'intera giornata.

STRUTTURA BASE

L'algoritmo di schedulazione si suddivide in due parti fondamentali:

- il MANAGER, il quale si occupa sia della fase **PRE-SCHEDULING**;
- lo SCHEDULER, che si occupa dello **SCHEDULING** vero e proprio.

PASSI FONDAMENTALI

MANAGER

- 1) Definizione obiettivo
- 2) Definizione attività
- 3) Ricerca ultimo peso dell'utente

SCHEDULER

- 4) Calcola fabbisogno calorico
- 5) Divisione fabbisogno in macronutrienti
- 6) Split nei Time-section

L'algoritmo sarà meglio chiarito in seguito attraverso l'utilizzo di Activity diagrams.

4.2.9 Activity Diagrams

FASE PRE-SCHEDULING

Questa fase prevede di definire gli obiettivi, le attività sportive e lavorative, che serviranno **come input per la fase successiva e dunque per l'algoritmo**.

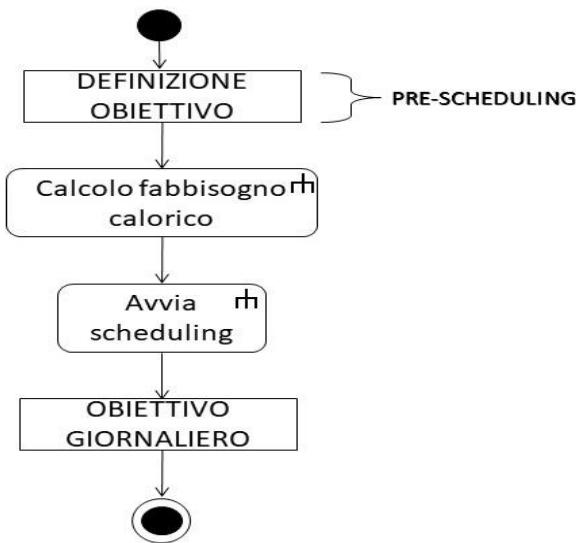


Figura 31 Activity Diagram: Pre-scheduling

Di seguito si definiscono le fasi “*Calcolo fabbisogno calorico*” e “*Avvia Scheduling*”.

CALCOLO FABBISOGNO GIORNALIERO

La fase del calcolo fabbisogno calorico permette di avere come output le calorie totali da assumere giornalmente. Dopo aver scelto il tipo di obiettivo da voler raggiungere e le attività svolte, l'algoritmo preleverà peso, altezza, sesso ed età ed effettuerà il calcolo matematico corretto.

Innanzitutto va calcolato il metabolismo basale:

$$\text{BMR uomo} = [(10 \times \text{PESO}) + (6,25 \times \text{ALTEZZA}) - (5 \times \text{ETÀ}) + 5] \text{ kcal}$$

$$\text{BMR donna} = [(10 \times \text{PESO}) + (6,25 \times \text{ALTEZZA}) - (5 \times \text{ETÀ}) - 161] \text{ kcal}$$

Successivamente, in base al tipo di attività selezionata effettueremo questo ulteriore calcolo:

$$\text{FABBISOGNO CALORICO GIORNALIERO} = \text{BMR} \times \text{coefficiente di attività.}$$

Il coefficiente di attività viene calcolato con il seguente schema in base al tipo di attività lavorativa e attività sportiva selezionata :

attività sedentaria + nessuna attività fisica = **1,2**

attività sedentaria + almeno 2-3 volte = **1,375**

attività leggera + nessuna attività fisica = **1,375**

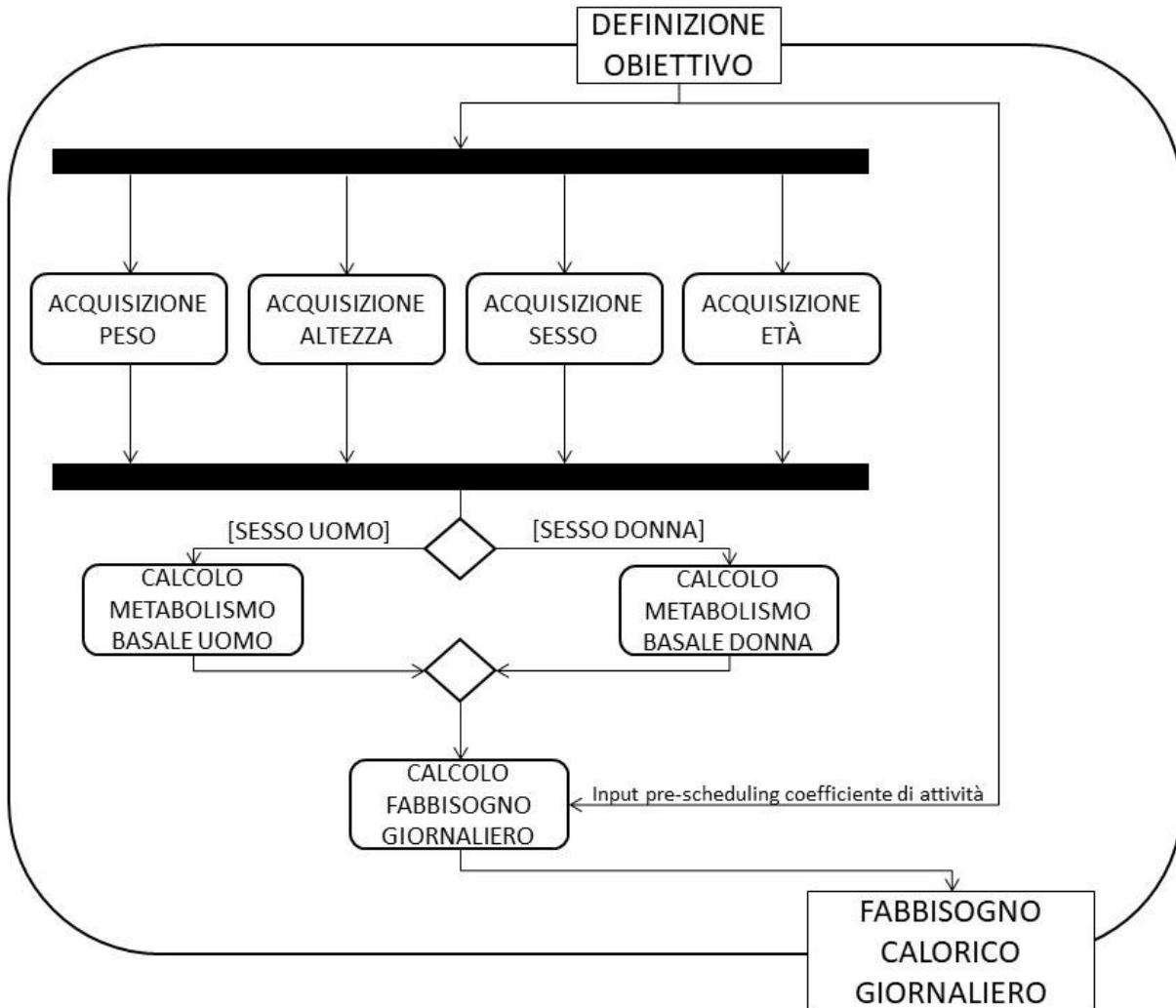
attività leggera + almeno 2-3 volte = **1,55**

attività mediamente pesante + nessuna attività fisica = **1,55**

attività mediamente pesante + almeno 2-3 volte = **1,725**

attività pesante + nessuna attività fisica = **1,725**

attività pesante + almeno 2-3 volte = **1,9**

**Figura 32** Activity Diagram: Calcolo fabbisogno calorico

Succesivamente verrà avviato lo scheduling del fabbisogno calorico giornaliero nei vari momenti della giornata in funzione anche dei macronutrienti.

AVVIA SCHEDULING

In questa fase, inizia lo scheduling vero e proprio del fabbisogno giornaliero. Avendo come input il totale delle calorie da assumere verranno svolte in modo parallelo due attività:

- Lo split nei time-section giornalieri, che consiste nel dividere in modo proporzionato le calorie nei vari pasti della giornata (colazione, spuntini, pranzo e cena)
- Lo split dei nutrienti, che consiste nel dividere in modo proporzionato le calorie secondo i tre macronutrienti fondamentali (proteine, grassi, carboidrati) ma chiaramente in funzione dell'obiettivo scelto a priori.

Dunque, se l'obiettivo posto all'inizio dell'algoritmo è la perdita di peso, il software effettuerà la seguente ripartizione: 45% proteine, 30% grassi, 25% carboidrati. Se l'obiettivo posto è il mantenimento, il software effettuerà la seguente ripartizione: 25% proteine, 25% grassi, 50% carboidrati. Altrimenti se l'obiettivo posto è l'aumento muscolare, il software effettuerà la seguente ripartizione: 35% proteine, 20% grassi, 45% carboidrati.

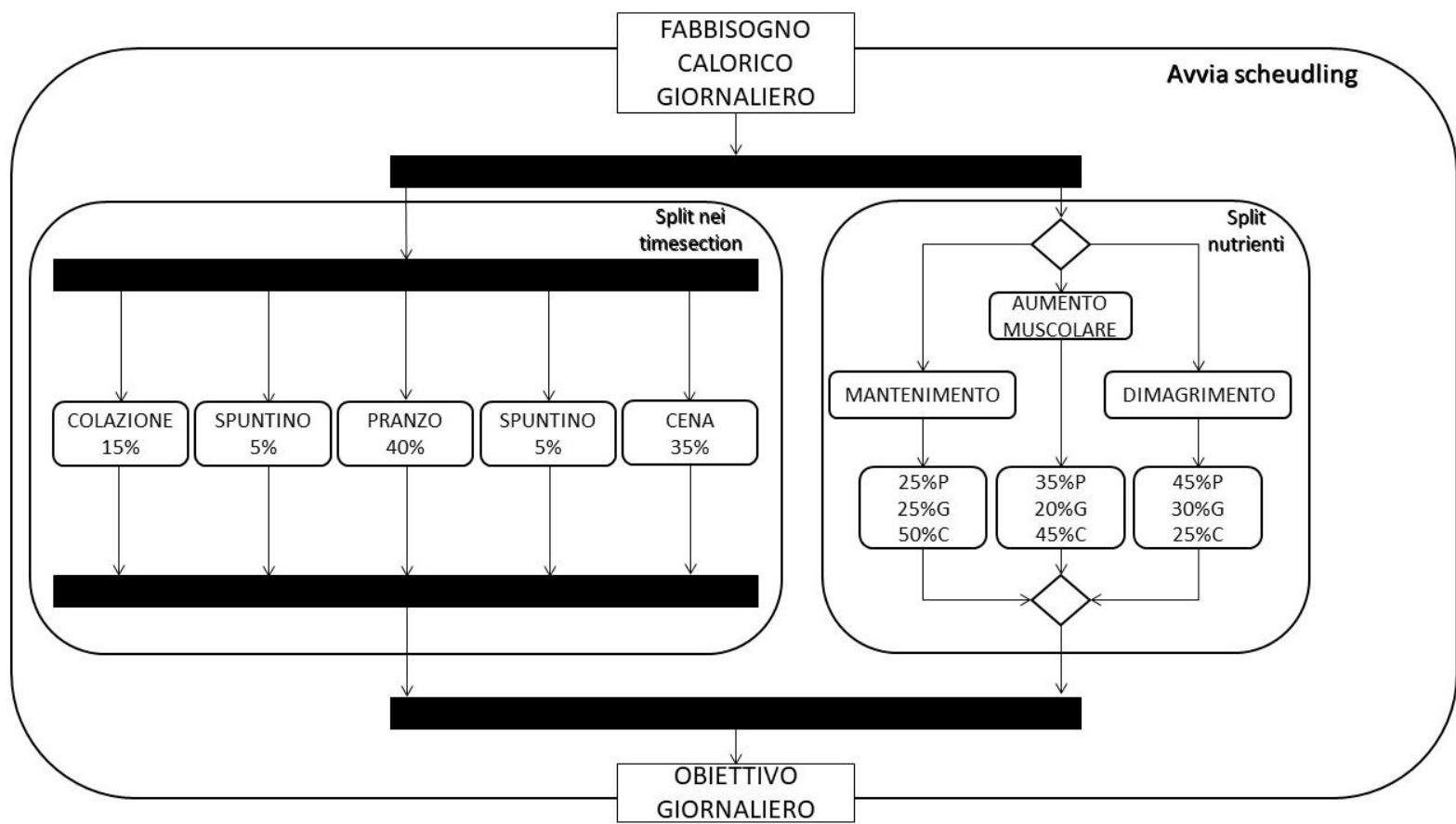


Figura 33 Activity Diagram: Avvia Scheduling

4.3 Implementation Phase

In questa fase, successiva a quella di progettazione, si passa all'implementazione vera e propria.

4.3.1 Configuration

Per quanto riguarda la connessione tra client e server e la connessione tra il server e il database, si è reso necessario inserire diversi parametri di configurazione, quali ad esempio l'**indirizzo ip**, la **porta** del server e del database e il tipo di **dbms** utilizzato. Per questo motivo si è scelto di memorizzare tali dati esternamente al codice sorgente, all'interno di un file **XML**, in modo tale da poterli recuperare e modificare a seconda delle necessità. Saranno dunque richiesti due diversi file XML:

[FILE XML CLIENT]

```
<IMPOSTAZIONI>
  <PROTOCOL>http</PROTOCOL>
  <SERVER_PORTA>8085</SERVER_PORTA>
  <SERVER_ADDRES>localhost</SERVER_ADDRES>
</IMPOSTAZIONI>
```

Dove i parametri avranno il seguente utilizzo:

- 1) ***protocol***: protocollo di connessione
- 2) ***port_server***: porta di connessione del server
- 3) ***server_database***: indirizzo ip del server

Ed uno per il server di questo tipo:

[FILE XML SERVER]

```
<IMPOSTAZIONI>
  <SERVER_DATABASE>localhost</SERVER_DATABASE>
  <PORT_DATABASE>3306</PORT_DATABASE>
  <NAME_DATABASE>diet_db</NAME_DATABASE>
  <PORT_SERVER>8085</PORT_SERVER>
</IMPOSTAZIONI>
```

Con i seguenti parametri:

- 1) ***server_database***: l'indirizzo ip del server che si occupa del database
- 2) ***port_database***: la porta di connessione con il database
- 3) ***name_database***: il nome del database a cui connettersi
- 4) ***port_server***: la porta in cui lasciare in ascolto il server per accettare nuove connessioni dal client

Tali file andranno posizionati nella stessa directory dell'eseguibile, e nel caso in cui non siano presenti, il client utilizzerà i valori di default, mentre il server richiederà tali valori in input all'utente, creando un nuovo file XML.

La scelta di utilizzare un sistema che gestisse il database non è casuale.

Un DBMS (DataBase Management System) permette di accedere in modo semplice e efficiente ad una base di dati mantenendone la consistenza, la privatezza e l'affidabilità. I vantaggi dell'uso di un DBMS sono molteplici:

- 1)** Affidabilità dei dati. Un DBMS offre dei metodi per salvare copie dei dati (backup) e per ripristinare lo stato della base di dati in caso di guasti software e hardware (recovery).
- 2)** Accesso ai dati tramite un linguaggio universale. Ogni DBMS di una certa tipologia mette a disposizione un linguaggio di interrogazione (SQL nel caso relazionale). Tale linguaggio permette la creazione delle strutture che contengono i dati, l'inserimento, la cancellazione, l'aggiornamento dei dati e il recupero delle informazioni dalla base di dati.
- 3)** Accesso concorrente ai dati. Un DBMS permetta a più utenti di accedere contemporaneamente alla base di dati. Più utenti possono accedere nello stesso istante a dati diversi. Inoltre, un DBMS fa in modo che l'accesso concorrente agli stessi dati non generi anomalie, cioè inconsistenza nello stato della base di dati rispetto alla realtà modellata.

Quest'ultimo punto ha spinto l'intero team nella fase dell'implementazione di un'architettura client-server, che dai requisiti del cliente dovesse permettere ad un'utente di poter accedere da più dispositivi, di creare una base di dati modellata ad hoc.

Tale scelta inoltre migliora l'organizzazione dal punto di vista strutturale e ne migliora la velocità e la fluidità gestionale del software. Oltre i tecnicismi, va definito che in particolare l'uso di un database deve essere necessariamente di supporto, in quanto è un software le cui componenti sono fortemente agganciate ad esso. Bisogna tenere traccia dello storico degli obiettivi mediante il resoconto ma soprattutto tenere traccia delle possibili richieste da parte di un utente ad un dottore facendo sì che esse non vengano perse tra un device e l'altro, ma soprattutto l'uso di questo componente permette che ci sia il “non ripudio”, sia da parte dell'utente che da quella del dottore. Senza questo componente, la visualizzazione in remoto da più device non potrebbe essere permessa.

4.3.3 Client

Per quanto riguarda il client, quest'ultimo avrà una struttura semplice, in cui avremo esclusivamente le funzioni legate all'interazione con l'utente. Il client come ribadito più volte tramite le interfacce grafiche, effettueranno richieste al server.

Sarà necessario aggiungere un'interfaccia iniziale per il login/la registrazione dell'utente, in quanto questa architettura viene scelta proprio per permettere l'accesso non solo multiplo tra più utenti, ma anche l'accesso da più dispositivi.

Le classi esportate andranno a formare un package denominato **DietaClient**.

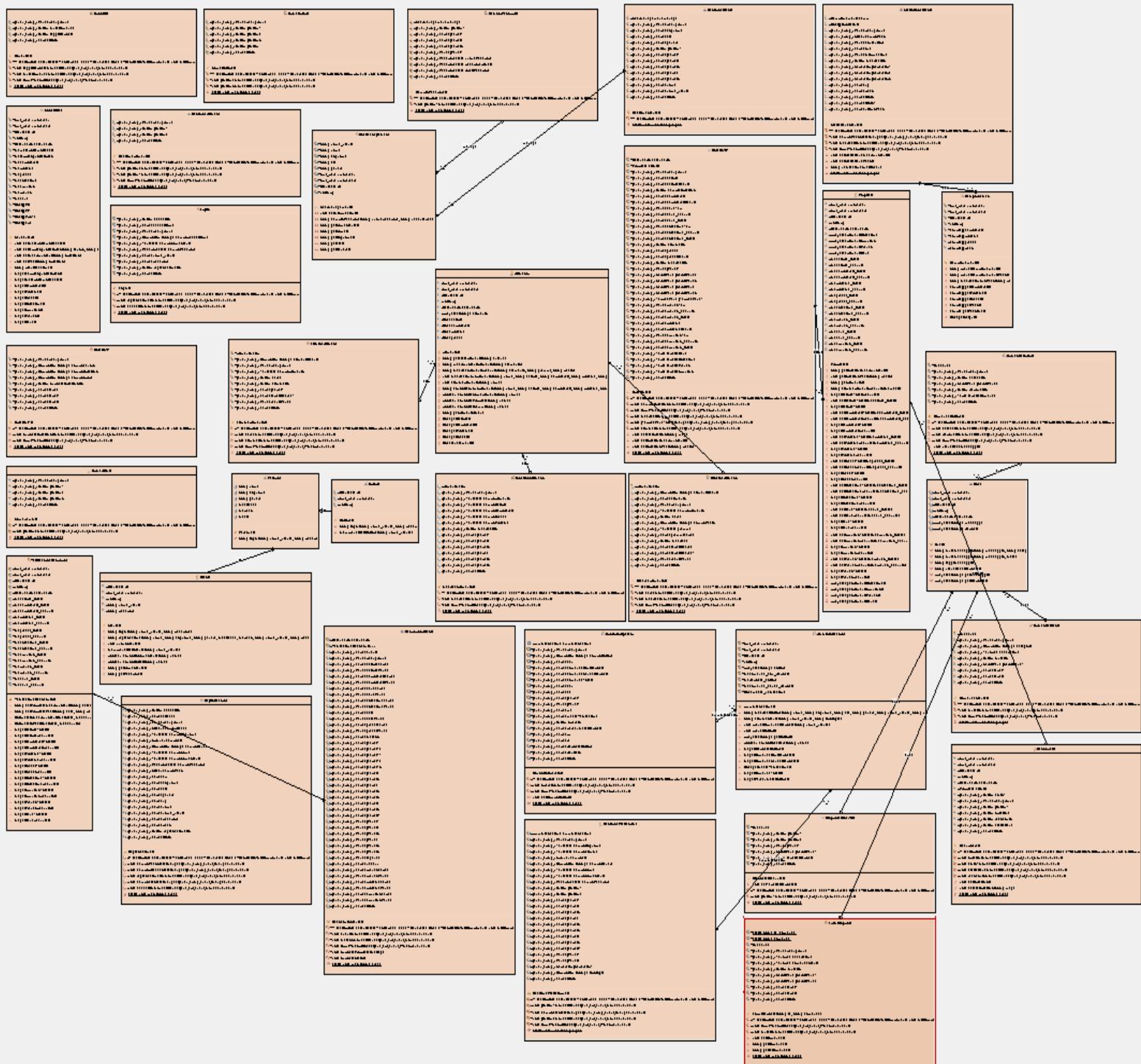


Figura 36 Class Diagram completo del DataClient

Una volta effettuata l'implementazione dell'architettura, il programma dovrà avere il seguente comportamento:

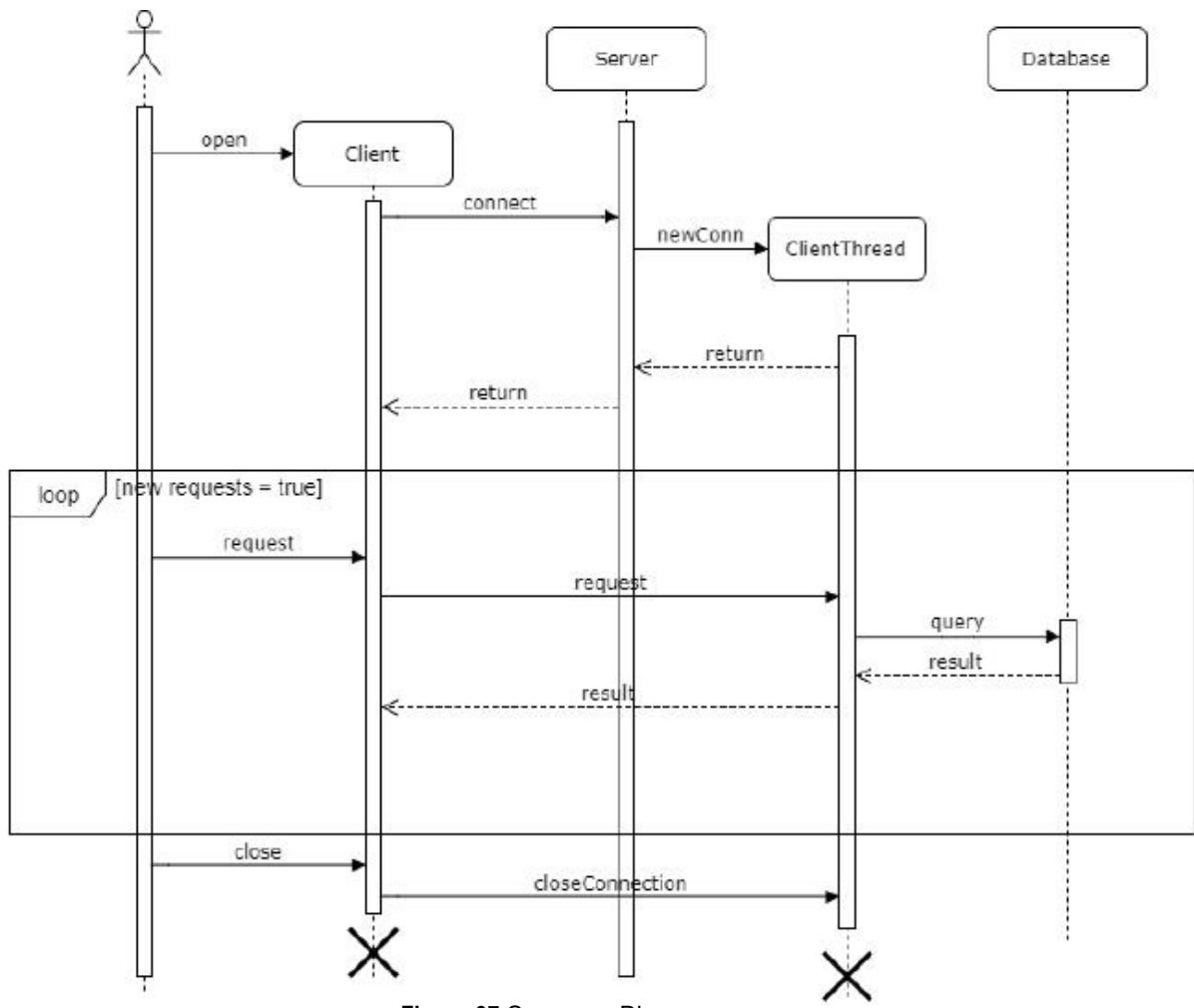


Figura 37 Sequence Diagram

4.3.4 Security

Affinché i dati dell'utente siano al sicuro, viene utilizzato l'algoritmo **MD5** per la codifica e decodifica della password. Per avere una maggiore sicurezza si concatena alla stringa della password un “salt”, in modo da poter contrastare gli attacchi a vocabolario o a forza bruta.

4.3.5 Physical Software Structure

La struttura del software principale è organizzata in sei package, che rispecchiano componenti principali.

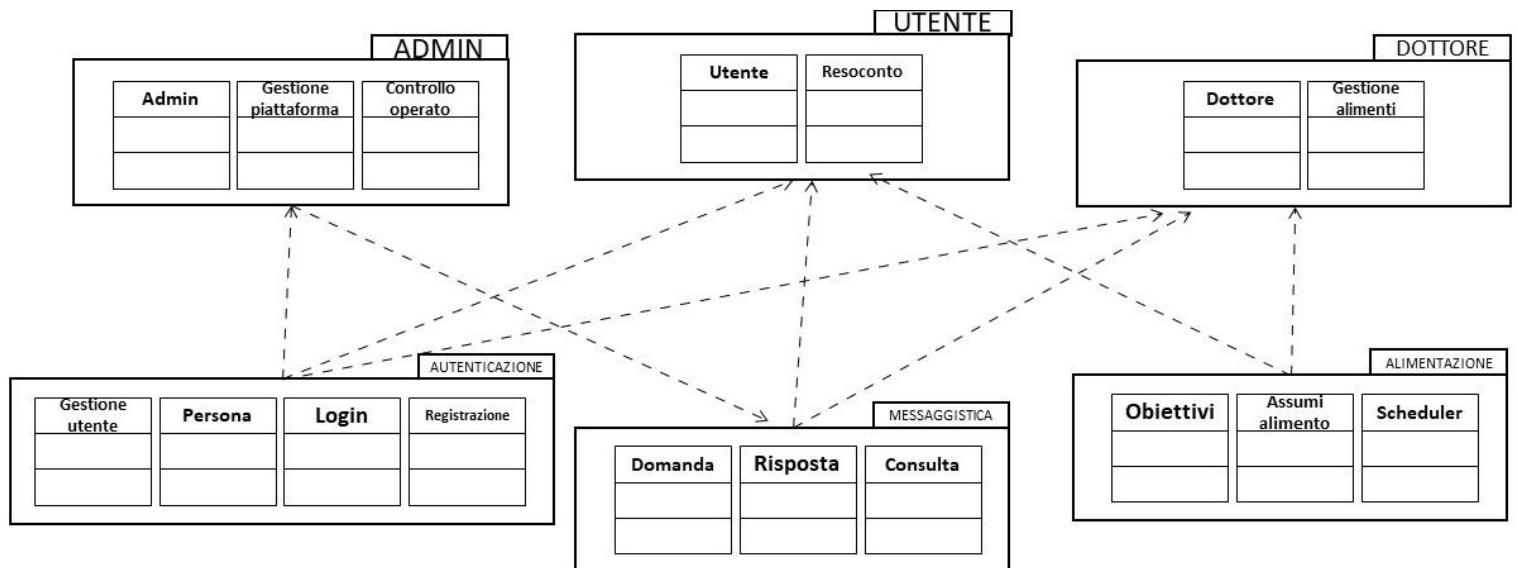


Figura 38 Package diagram

All'interno del precedente diagramma, vengono raffigurate le classi più importanti dei package. Risulta necessario l'utilizzo di diagrammi più efficaci per la descrizione dei package, in maniera dettagliata, in modo da non tralasciare nessuna classe. In seguito verranno riportati i diagrammi delle classi, che consentono di descrivere le entità e le loro caratteristiche ma anche le eventuali relazioni tra loro.

PACKAGE ALIMENTAZIONE

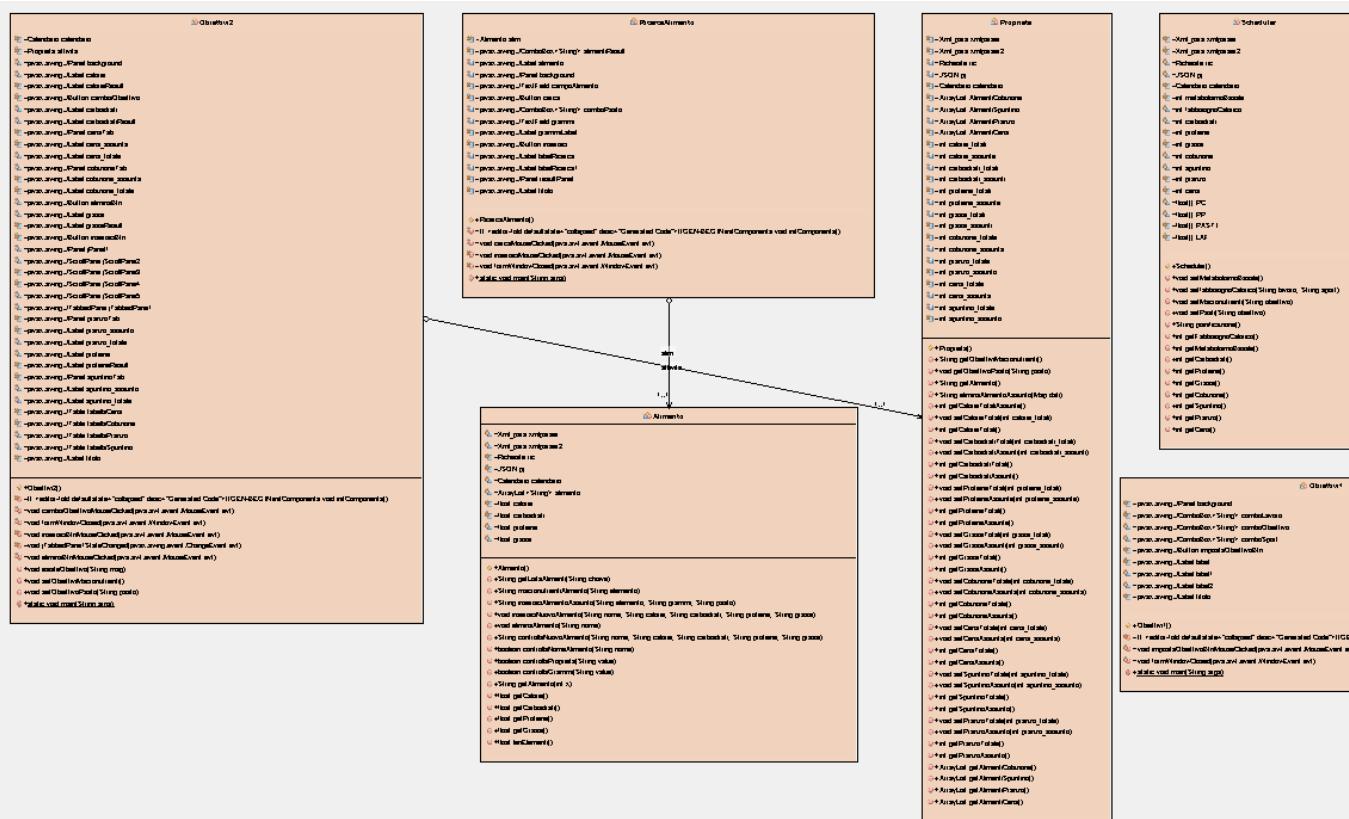


Figura 39 Class diagram: Alimentazione

PACKAGE MESSAGGISTICA

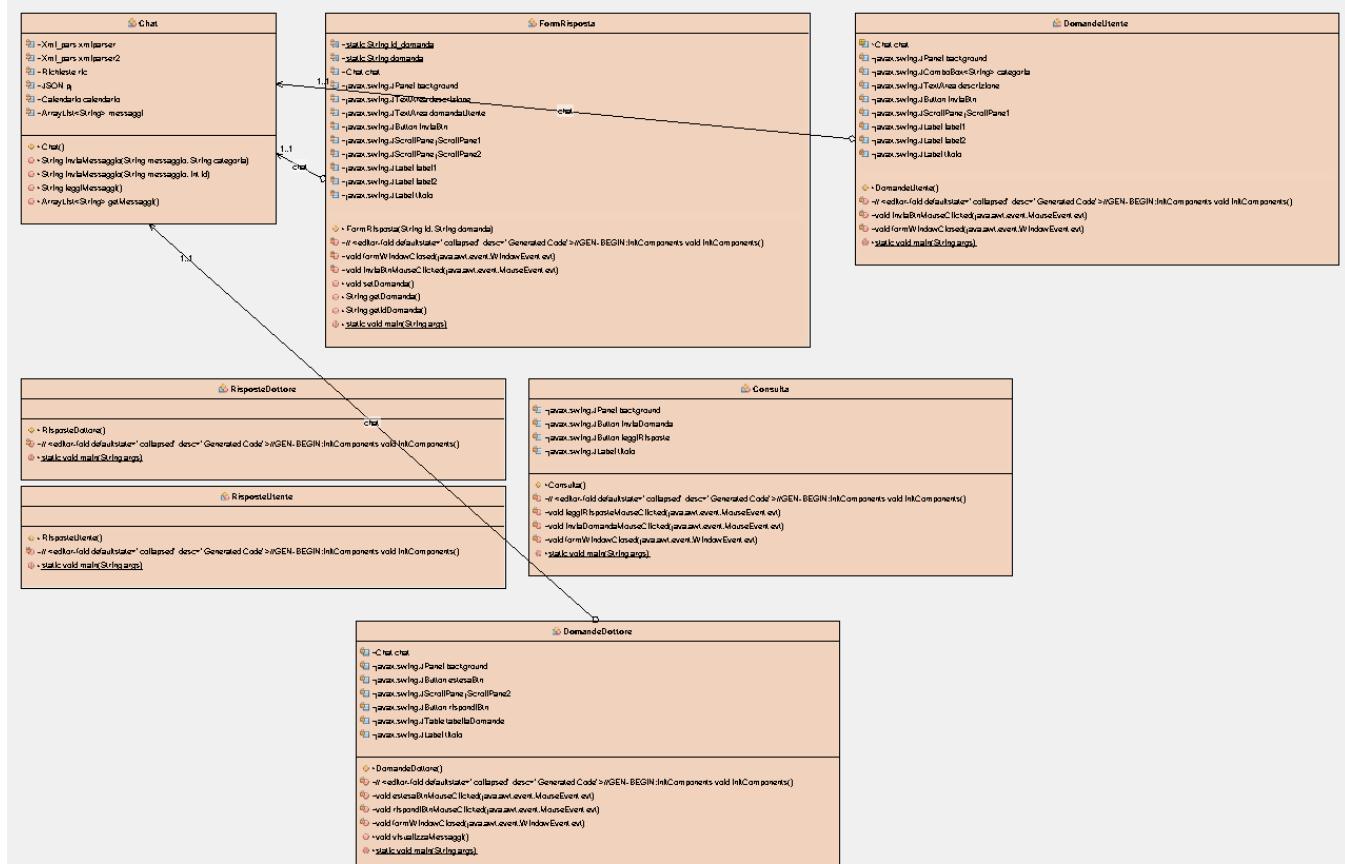


Figura 40 Class diagram: Messaggistica

PACKAGE AUTENTICAZIONE

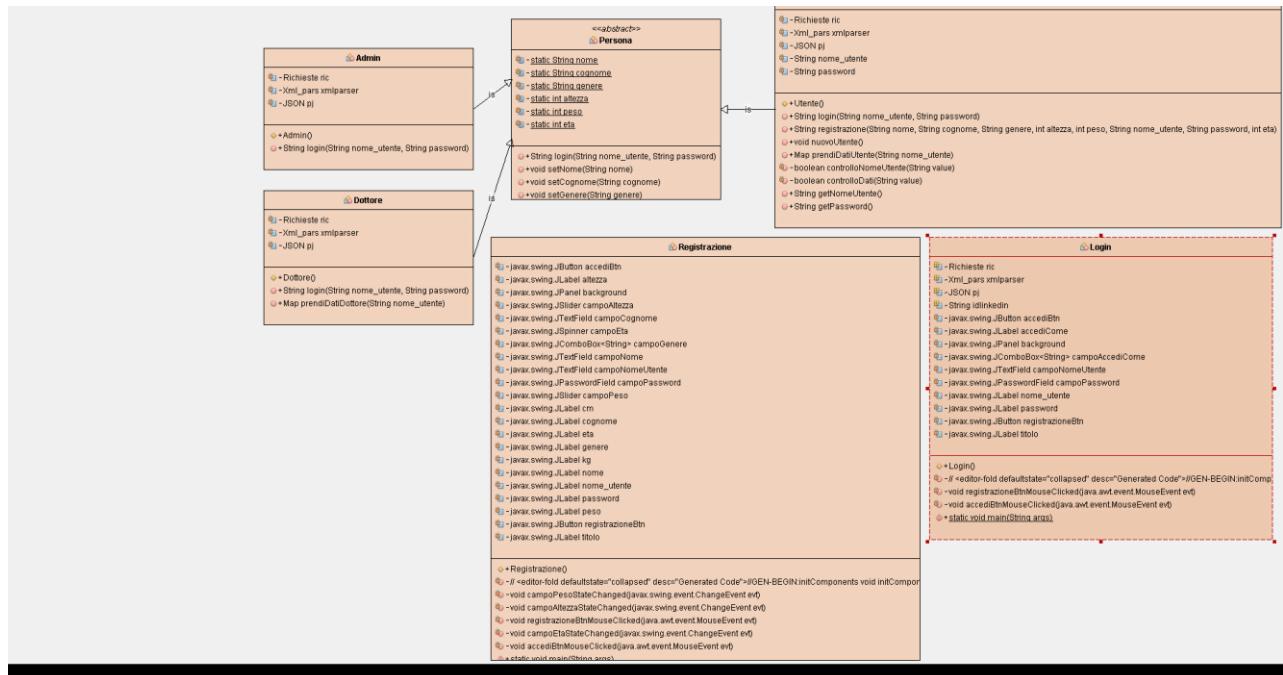


Figura 41 Class diagram: Autenticazione

PACKAGE UTENTE

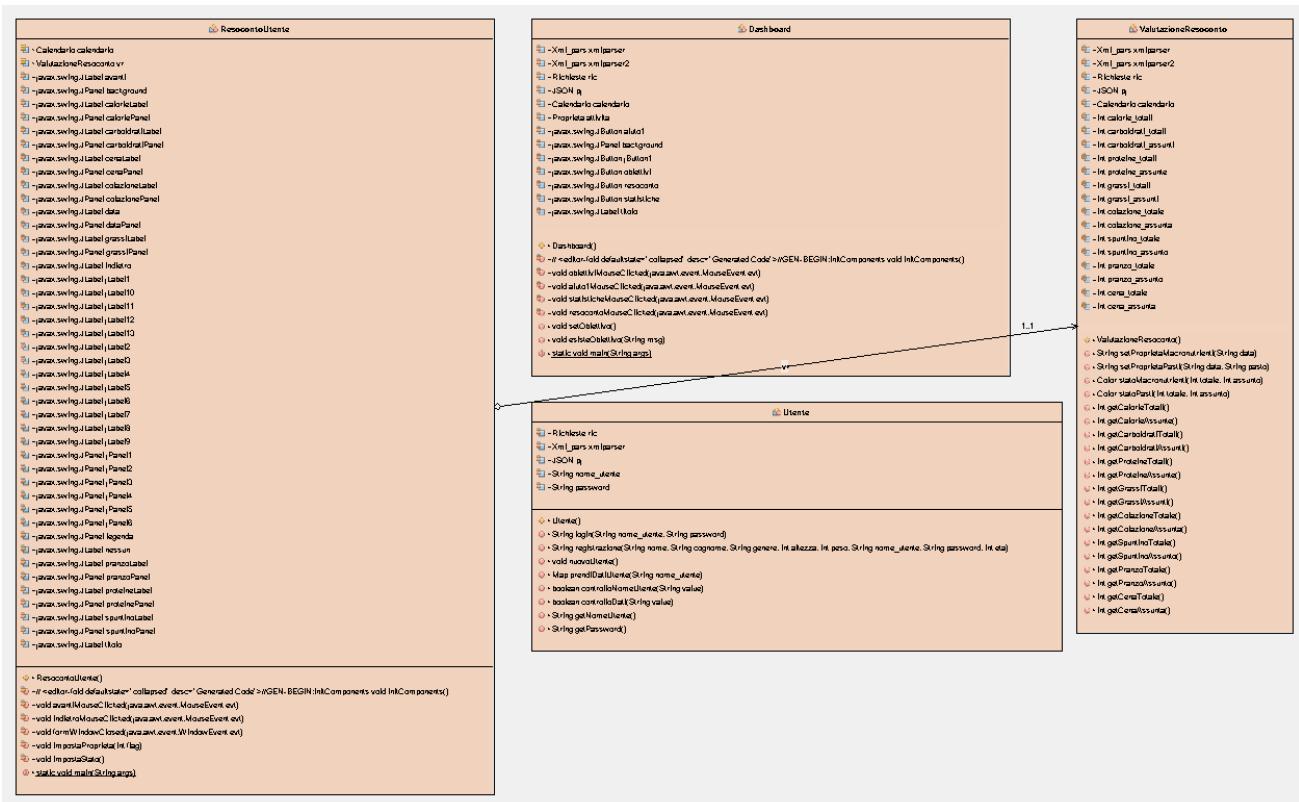


Figura 42 Class diagram: Utente

PACKAGE DOTTORE

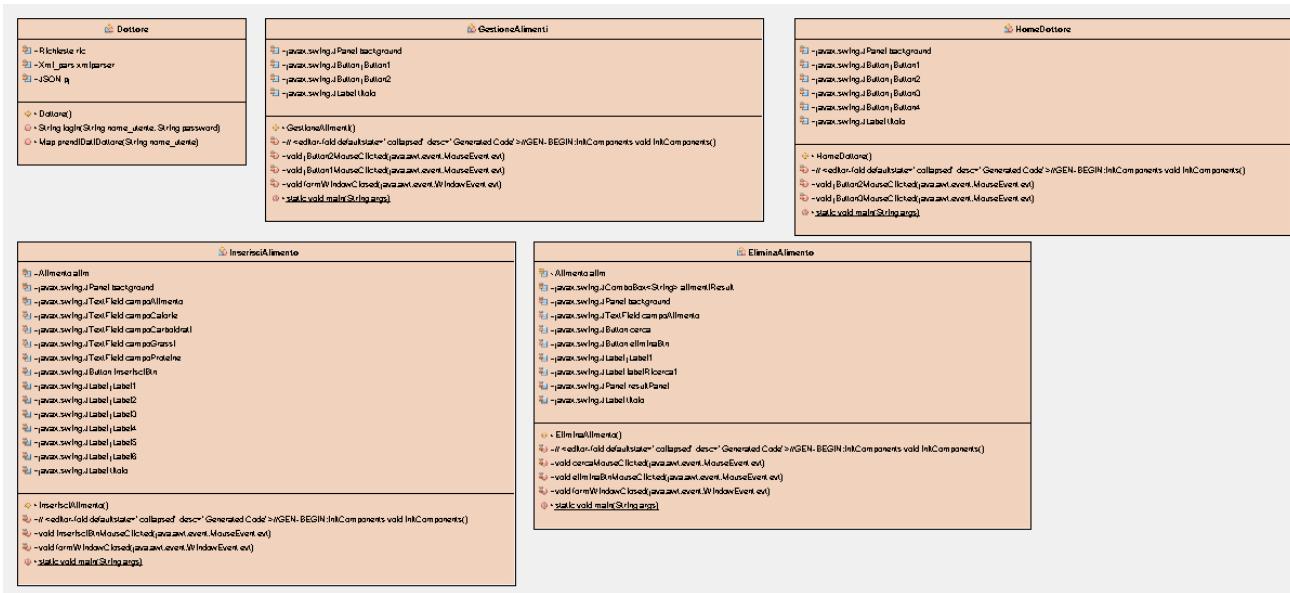


Figura 43 Class diagram: Dottore

PACKAGE ADMIN

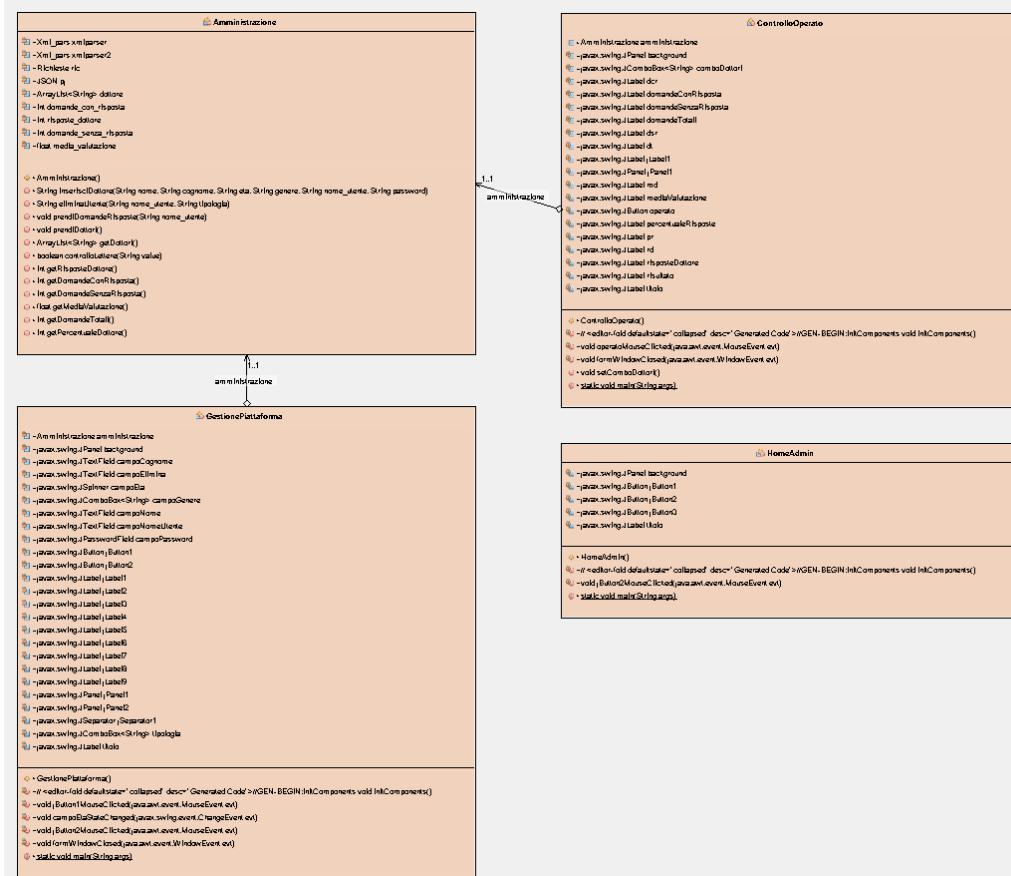


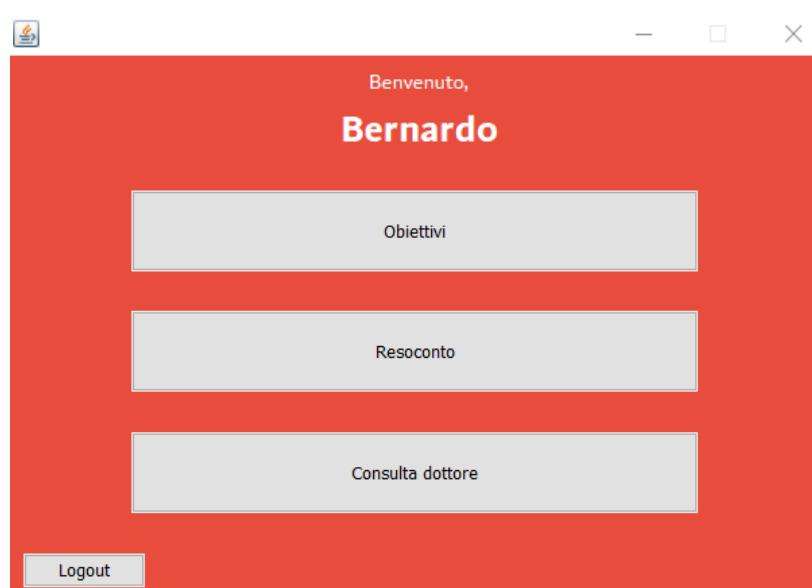
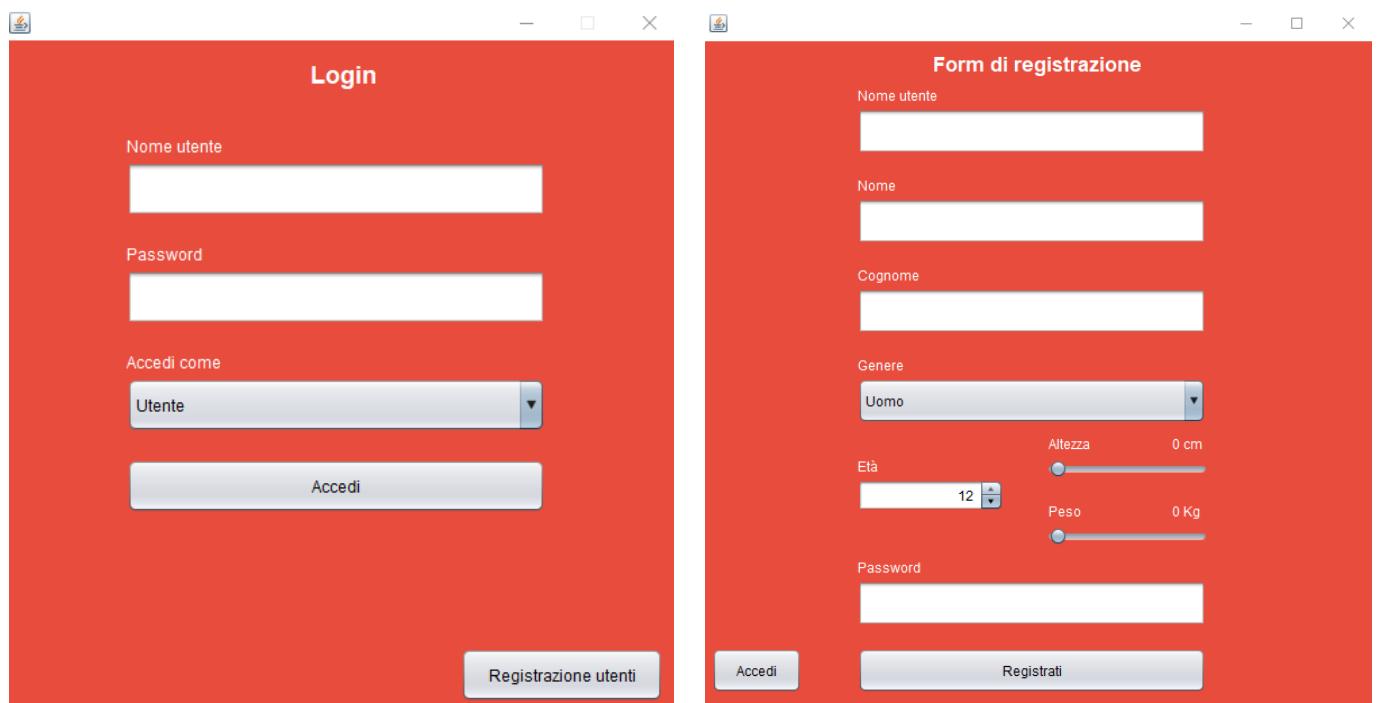
Figura 44 Class diagram: Admin

4.3.6 GUI Structure

La struttura della GUI si basa su un insieme di Frame, al cui interno, tramite dei Button ci sposteremo tra di essi. La struttura è abbastanza semplice in quanto ogni Frame viene richiamato solo quando un evento voluto dall'utente lo farà comparire. Risulta evidente che ogni Frame viene subordinato dalla DashBoard principale in cui l'utente troverà i Button per operare all'interno di tutto il software.

Per cambiare pannello è sufficiente utilizzare la funzione statica *SetVisible*.

4.3.7 GUI's Screens (UTENTE)



Imposta obiettivo

Seleziona l'obiettivo che vuoi raggiungere:

Dimagrimento

Selezione il livello di attività lavorativa:

Lavoro fisicamente pesante ma non eccessivamente

Selezione l'attività sportiva settimanale:

Non pratica sport

Imposta obiettivo

Obiettivi di oggi

Calorie 448/2157kcal	Carboiodrati 52/862g	Proteine 147/1078g	Grassi 32/217g
-------------------------	-------------------------	-----------------------	-------------------

Colazione Spuntino Pranzo Cena

Calorie totali da assumere: 494kcal
Calorie attualmente assunte: 258kcal

Alimento	Grammi	Calorie (kcal)	Carboiodrati (g)	Proteine (g)	Grassi (g)
Mela	100	48	14	0	0
Biscotti frollini	30	180	18	1	6
Latte	50	30	2	1	1

Cambia obiettivo **Inserisci alimento** **Elimina alimento**

Aggiungi alimento

Alimento

Prosciutto

Ricerca per: 'Prosciutto'

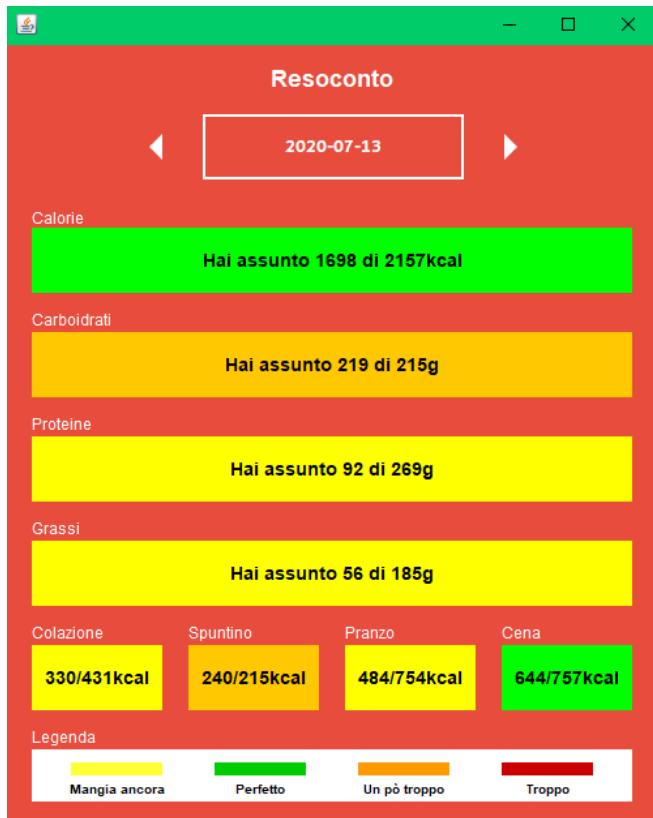
Prosciutto crudo

Pasto

Colazione

Grammi

Inserisci alimento





Consulta dottore

Selezione la categoria della tua domanda

Non riesco a dimagrire

Inserisci la descrizione della tua domanda (max 200 caratteri)

Invia domanda

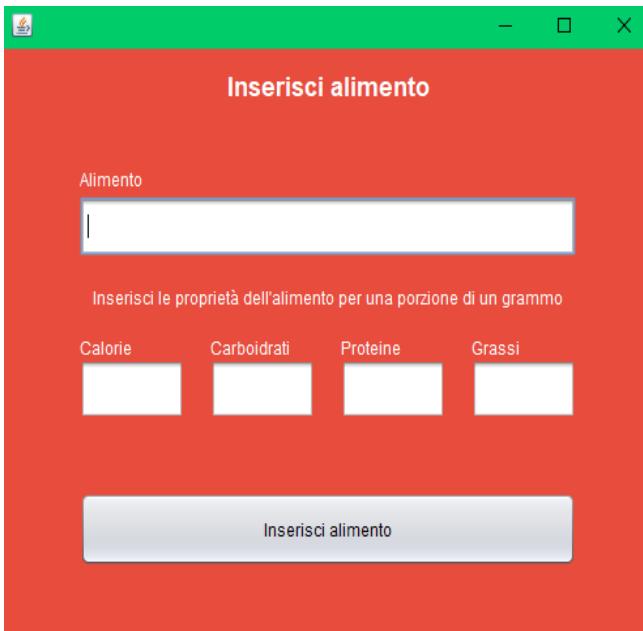
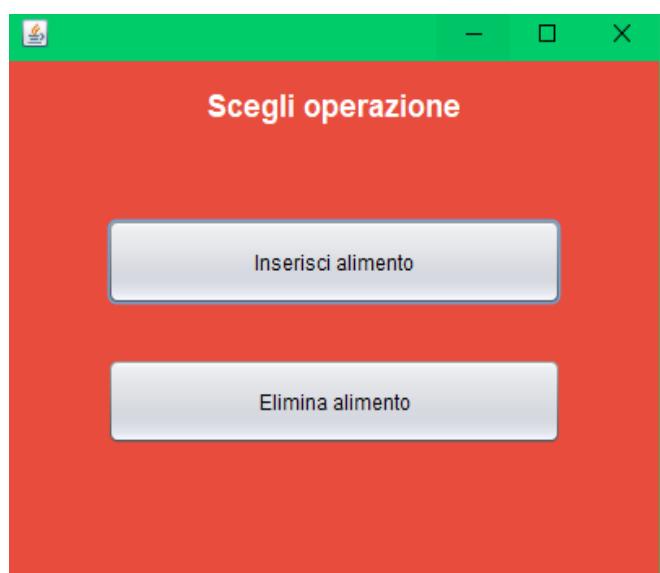
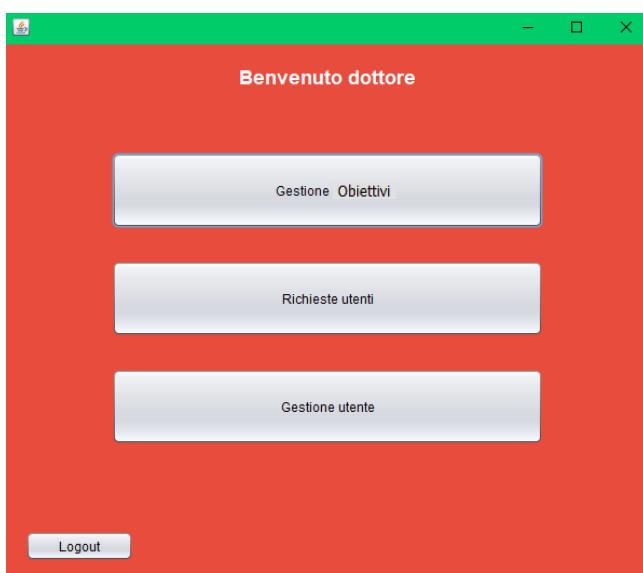
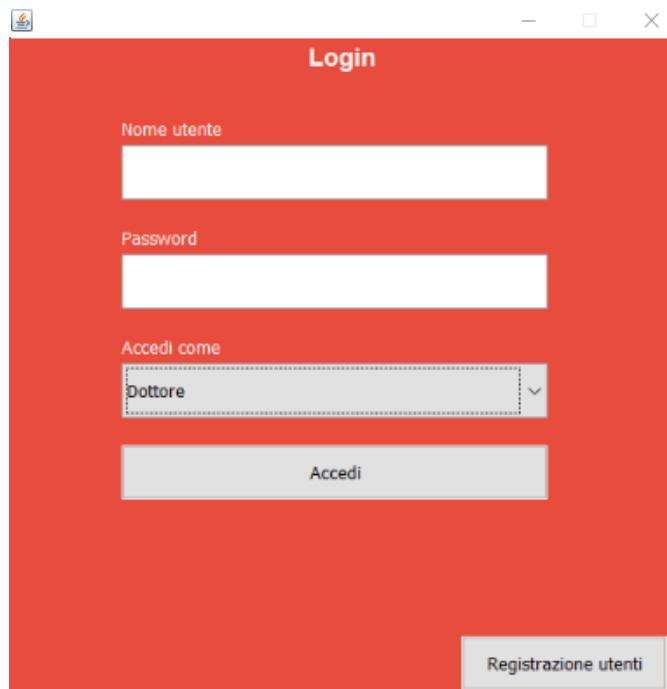
Risposte ricevute

ID	Dottore	Domanda	Risposta
5	DrBianchi	Domanda di prova 1	Va tutto bene
4	Bernardo	Salve dottore, sono intollerant...	Ciao
3	Bernardo	Salve dottore, vorrei sapere s...	Salve utentex, dopo una atten...
2	Bernardo	Salve dottore,yorrei capire me...	Salve utentex, dopo una atten...

Visualizza risposta estesa

Valuta

4.3.8 GUI's Screens (DOTTORE)



Domande utenti

ID	Utente	Categoria	Domanda
5	Bernardo	Non riesco a dimagrire	Domanda di prova 1

Form di risposta

Domanda dell'utente

Salve dottore, sono intollerante al glutine quindi potrebbe consigliarmi alimenti pertinenti al mio stile di vita?

Inserisci la descrizione della tua domanda (max 200 caratteri)

Visualizza domanda estesa

Rispondi

Invia risposta

Gestione utente

Password attuale

Ripeti password

Nuova password

Imposta nuova password

4.3.9 GUI's Screens (ADMIN)

Login

Nome utente

Password

Accedi come

Admin

Accedi

Registrazione utenti

Gestione piattaforma

Inserisci un nuovo dottore

Nome	Età
<input type="text"/>	<input type="text" value="18"/>
Cognome	Genere
<input type="text"/>	<input type="text" value="Uomo"/>
Nome_utente	Password
<input type="text"/>	<input type="text"/>

Elimina dottore/utente

Nome utente	Dottore
<input type="text"/>	<input type="text" value="Dottore"/>

Inserisci

Controllo operato

Dottore

<input type="text" value="DrRossi"/>

Visualizza operato

Risultato dell'operato del dottore "DrRossi"

Domande Totali	Domande con risposta	Domande senza risposta
4	3	1

Risposte dottore Percentuale risposte Media valutazione

0	0	0.0
----------	----------	------------

4.4 Testing Phase

Dopo aver sviluppato il primo prototipo è possibile passare ad una prima fase di testing di sistema, in particolare in questa fase verrà testato il software utilizzando un approccio **black-box**. Si valuta dunque il software “a scatola chiusa”, senza cioè verificare in questa fase la validità del codice (che è comunque stata testata passo passo durante la fase di sviluppo), ma controllando come il programma finale si comporta in determinate condizioni e verificando che rispetti i requisiti funzionali richiesti.

Il software verrà inizialmente testato con un insieme di input “corretti”, per verificare che il risultato soddisfi i requisiti funzionali, infine verranno inseriti degli insiemi di input che tenteranno di verificare il comportamento di quest’ultimo nel caso di input errati.

4.4.1 Validation

In questa fase si verificherà se il software permette di rispondere alle esigenze dell’utente, rispettando i seguenti requisiti funzionali:

RQ1: GESTIONE ACCESSO

RQ : Registra utente

RQ : Login

RQ : Logout

Dopo aver testato l’aggiunta di un nuovo utente, il login e il logout, il tutto risulta perfettamente funzionante ed in linea con i requisiti richiesti. In particolare effettuando l’accesso, si presentano esclusivamente dati ed informazione di quest’ultimo senza che ci siano sovrapposizioni di dati con altri utenti. Inoltre, il menù “Accedi come” contenente la categoria di utente non crea alcun problema, e dunque un utente in alcun modo riesce ad accedere a contenuti altrui. Inoltre è stata testata anche la sicurezza dell’accesso, provando ad accedere con nomi e password errati senza alcuna conseguenza sull’integrità del software.

RQ2.0: GESTIONE OBIETTIVI (UTENTE)

RQ2.0.1: Definizione obiettivo

RQ2.0.2: Assumi alimento

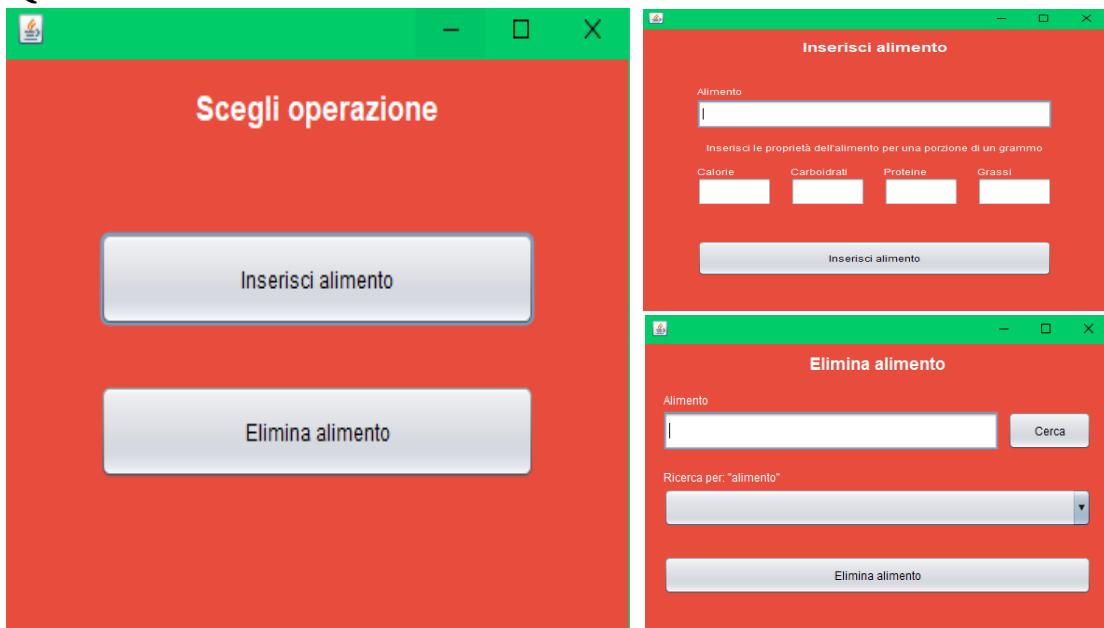
RQ2.0.3: Recosconto

Alimento	Grammi	Calorie (kcal)	Carbo (g)	Proteine (g)	Grassi (g)
Mela	100	48	14	0	0
Biscotti frollini	30	180	18	1	6
Latte	50	30	2	1	1

Anche per quanto riguarda i requisiti relativi al diario, questi risultano rispettati: nessun problema riscontrato nella definizione di un obiettivo, nell'inserimento degli alimenti al diario nelle rispettive time-section. Per la definizione di un nuovo obiettivo, cliccando sul bottone “cambia obiettivo” potremo modificare l’obiettivo iniziale posto dalla schermata “imposta obiettivo”. Inoltre l’assunzione è definita dalla schermata che compare dopo aver effettuato il click sul bottone “inserisci alimento” e con la stessa metodologia andremo ad eliminare un alimento che precedentemente è stato inserito nel diario con il bottone “elimina alimento”.



Per quanto riguarda il **resoconto**, il requisito viene soddisfatto a pieno, andando a delineare giorno per giorno se gli obiettivi sono stato raggiunti con successo o con insuccesso, mediante una grafica decisamente user-friendly.

RQ2.1: GESTIONE OBIETTIVI (DOTTORE)**RQ2.1.1:** Inserisci alimento**RQ2.1.2:** Elimina alimento

Per la gestione degli obiettivi, per quanto concerne l'uso esclusivo del dottore, sono stati testati i due requisiti.

In particolare è stato testato l'inserimento di un alimento alla piattaforma e l'eliminazione dello stesso, senza che siano stati riscontrati errori e dunque le funzionalità rispecchiano i requisiti iniziali.

RQ3.0: RICHIESTE DOTTORE (UTENTE)**RQ3.0.1:** Richiesta di consultazione(invio)**RQ3.0.2:** Presa visone e valutazione della risposta del dottore.

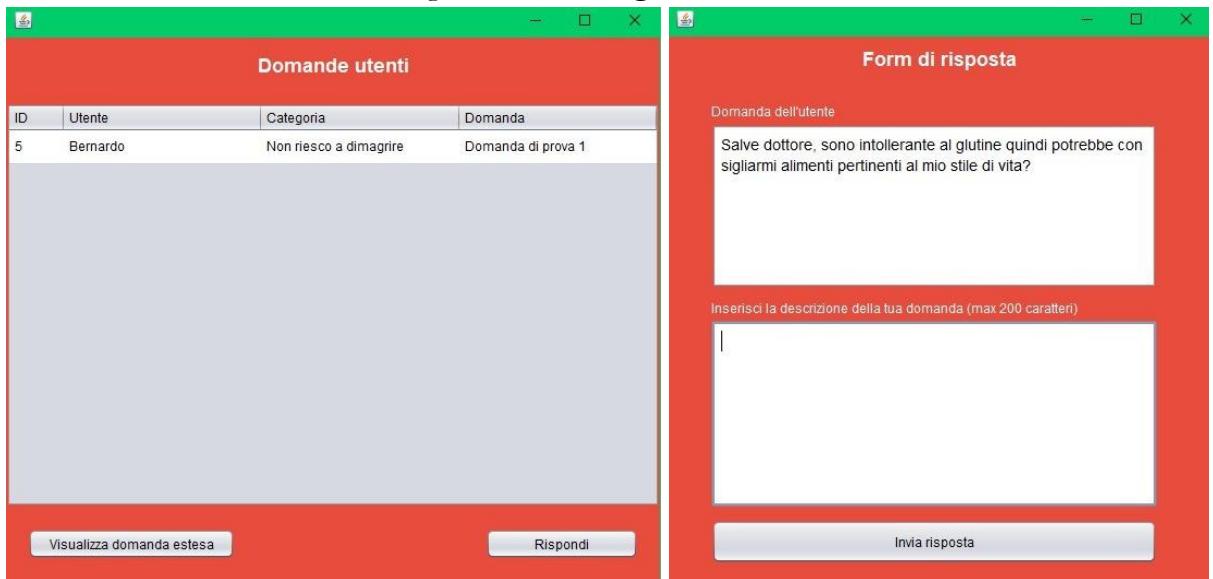
Feedback

Per quanto riguarda le richieste al dottore, da parte dell'utente, è stato testato il corretto invio di una richiesta in funzione della categoria di richiesta selezionata. Nella schermata della risposta da parte del dottore è stata testata la ricezione della risposta e l'invio del feedback. I requisiti anche in questo caso sono conformi alle aspettative iniziali senza che vengano riscontrati errori.

RQ3.1: RICHIESTE DOTTORE (DOTTORE)

RQ3.1.1: Richiesta di consultazione(risposta)

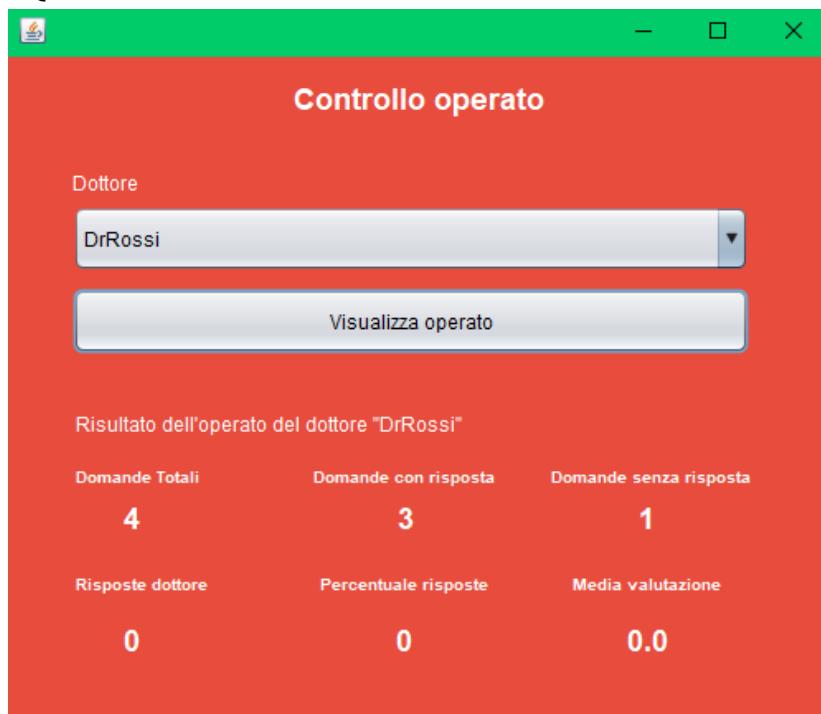
RQ3.1.2: Inserimento alimenti in seguito a richiesta [gestione obiettivi]



Le richieste, come è scontato che sia, devono ritornare all'utente che le ha formulate una risposta. Il requisito posto in esame è proprio questo, ovvero la risposta del dottore ed è stato valutato che lo sviluppo corrisponde a ciò che è stato delineato in fase di progettazione. Qualora un utente suggerisca un alimento mediante la categoria opportuna, sarà compito del dottore spostarsi nella sezione corretta. L'inserimento di un alimento è stato già testato in precedenza.

RQ4: CONTROLLO OPERATO LISTA

RQ4.1: Controllo richieste e feedback



l'admin prenderà sottoesame le valutazioni degli utenti per singolo dottore e valuterà come è stato valutato in seguito alle risposte. Sarà poi l'admin a gestire queste valutazioni opportunamente in funzione del proprio ruolo, in quanto gestore della piattaforma.

RQ5: GESTIONE PIATTAFORMA

RQ5.1: Elimina utente

RQ5.2: Inserisci dottore

RQ5.3: Elimina dottore

The screenshot shows a red-themed user interface for managing a platform. At the top, it says "Gestione piattaforma". Below that, there are two main sections: "Inserisci un nuovo dottore" and "Elimina dottore/utente".

Inserisci un nuovo dottore: This section contains four input fields: "Nome" (with placeholder "I"), "Cognome" (empty), "Nome_utente" (empty), and "Password" (empty). To the right of "Nome" is a dropdown menu set to "18". To the right of "Cognome" is a dropdown menu set to "Uomo". Below these fields is a large grey button labeled "Inserisci".

Elimina dottore/utente: This section contains two input fields: "Nome utente" (empty) and a dropdown menu set to "Dottore". Below these fields is a large grey button labeled "Elimina".

Il testing di questi requisiti, è stato condotto con successo in quanto mediante l'opportuna schermata è stato possibile eliminare un dottore e tutto ciò che all'interno della piattaforma era sotto il proprio nome, dunque tutti i dati che erano presenti nel resoconto. Con il testing degli ultimi due requisiti è stata valutata l'effettiva possibilità di inserire ed eliminare un dottore, e tutto ciò che è stato sviluppato è conforme alle aspettative iniziali.

CONCLUSIONI

E' dunque possibile concludere che il software sia valido, in quanto in grado di rispondere a tutti i requisiti forniti in fase iniziale.

4.4.2 Verification

È tuttavia necessario ora verificare che il software sia non solo funzionale, ma anche privo di errori. Per questo motivo verranno effettuati dei test con lo scopo di cercare di trovare qualche eventuale errore, inserendo degli insiemi di dati in tutti i possibili input del programma:

TEST #1

TEST	VALORE DI INGRESSO	RISULTATO
	Sequenza di caratteri alfabetici	Corretta ricerca
	Sequenza di caratteri speciali	Campo non accettato
	Stringa vuota	Campo non accettato

Questo particolare test è stato ripetuto per tutti i campi di input nei quali era richiesta una stringa ed i risultati sono stati equivalenti, evidenziando dunque come sia necessario fornire un valore in input: nel caso in cui non sia inserito nulla nel campo, la pressione del pulsante di invio non effettuerà alcuna azione, finché l'utente non avrà digitato qualcosa.

TEST #2

TEST	VALORE DI INGRESSO	RISULTATO
	Numero positivo	Corretto inserimento
	Numero negativo	Valore non accettato
	Caratteri alfabetici	Non inseribili

Analogamente ai test precedenti è stato ripetuto per gli altri campi che richiedano numeri come input, ed il risultato è stato il medesimo, in particolare nel caso del peso di ogni singolo utente e dell'alimento. Il peso viene inserito in modo da essere coerente

CONCLUSIONI

I test condotti non hanno dunque rilevato apparenti casi di malfunzionamenti dovuti ad inserimenti errati di dati di input. La natura intrinseca del programma, tuttavia, fa sì che il funzionamento di quest'ultimo sia fortemente dipendente dal trascorrere del tempo. È dunque possibile l'insorgere di qualche bug nel caso di determinate combinazioni di fattori comprendenti la data odierna, motivo per il quale si renderebbe necessaria una fase di **alpha testing** iniziale.

4.5 Customer Review and Feedback

Dopo che il primo prototipo è stato rilasciato, il cliente si dichiara abbastanza soddisfatto del risultato raggiunto mediante il primo prototipo, che si dimostra già potenzialmente pronto per l'utilizzo.

Nonostante la necessità di inserimento di molteplici parametri, il software appare semplice e intuitivo.

Le diverse prove effettuate hanno mostrato un'ottima distribuzione del fabbisogno calorico ed in particolare è stato apprezzato l'alto grado di personalizzazione che viene fornito all'utente. Il punto di forza del software risiede infatti nella possibilità di gestire diversi obiettivi e diverse informazioni iniziali, potendo variare a proprio piacimento la schedulazione degli obiettivi secondo le proprie esigenze e preferenze.

Tuttavia il cliente richiede che vengano inserite ulteriori funzionalità, ovvero delle statistiche personali con annessi grafici che permettano una migliore visualizzazione degli obiettivi e la possibilità che venga stampato il pdf del resoconto in uno specifico giorno.

5. Second Iteration (Prototype 2)

5.1 Requirements Phase

In seguito allo sviluppo del primo prototipo, il confronto con il cliente ha portato alla definizione di ulteriori requisiti:

5.1.2 New Functional Requirements

RQ6: Possibilità di avere una statistica del proprio peso, al variare dei giorni

RQ7: Possibilità di avere una statistica dei macronutrienti assunti, al variare dei giorni

E' stato dunque possibile aggiornare la tabella dei requisiti come segue:

5.1.3 Matrix of Functional Requirements (update)

NOME REQUISITO	ID	TIPO	PRIORITÀ	DESCRIZIONE	REQUISITO PADRE	REQUISITO FIGLIO
Gestione accesso	1.0	funzionale	1	Deve consentire la gestione degli account utenti		1.0.1-2-3-4
Registra utente	1.0.1	funzionale	1	Deve consentire all'utente di creare un account	1.0	
Login utente	1.0.2	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.0	
Logout utente	1.0.3	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.0	
Profilo utente	1.0.4	funzionale	1	Deve consentire all'utente di vedere le informazioni personali	1.0	
Gestione accesso	1.1	funzionale	1	Deve consentire la gestione degli account utenti		1.1.1-2
Login admin	1.1.1	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.1	
Logout admin	1.1.2	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.1	
Gestione accesso	1.2	funzionale	1	Deve consentire la gestione degli account utenti		1.2.1-2-3
Login dottore	1.2.1	funzionale	1	Deve consentire all'utente di accedere al proprio account	1.2	
Logout dottore	1.2.2	funzionale	1	Deve consentire all'utente di disconnettersi dal proprio account	1.2	

Profilo dottore	1.2.3	funzionale	1	Deve consentire all'utente di vedere le informazioni personali	1.2	
Gestione obiettivi	2.0	funzionale	2	Deve consentire all'utente di pianificare e gestire i propri obiettivi		2.0.1-2-3-4
Definizione obiettivo	2.0.1	funzionale	2	Deve consentire all'utente di definire l'obiettivo per il quale usa il software	2.0	
Assunzione alimenti	2.0.2	funzionale	2	Deve consentire all'utente di inserire gli alimenti assunti quotidianamente	2.0	
Resoconto	2.0.3	funzionale	2	Deve consentire all'utente di visualizzare lo storico degli obiettivi passati	2.0	
Gestione obiettivi	2.1	funzionale	2	Deve consentire al dottore gestire gli alimenti della piattaforma		2.1.1-2
Inserimento alimenti	2.1.1	funzionale	2	Deve consentire al dottore inserire alimenti alla piattaforma	2.1	
Eliminazione alimenti	2.1.2	funzionale	2	Deve consentire al dottore eliminare alimenti dalla piattaforma	2.1	
Richieste dottore	3.0	funzionale	3	Deve consentire all'utente di poter avere un confronto con un dottore		3.0.1-2
Richiesta di consultazione (invio)	3.0.1	funzionale	3	Deve consentire all'utente di poter chiedere ad un dottore, un qualsiasi consulto specialistico	3.0	
Presa visione e valutazione risposta del dottore	3.0.2	funzionale	3	Deve consentire all'utente di visualizzare la risposta del dottore e dare un feedback.	3.0	
Richieste dottore	3.1	funzionale	3	Deve consentire al dottore di poter rispondere alle richieste di un utente		3.1.1-2
Richiesta di consultazione (risposta)	3.1.1	funzionale	3	Deve consentire al dottore di rispondere ad una generica richiesta da parte di un utente	3.1	
Controllo operato liste	4	funzionale	4	Deve consentire all'admin di poter controllare gli esiti delle richieste fatte ai dottori		4.1

Controllo richieste e feedback	4.1	funzionale	4	Deve consentire all'admin di visualizzare l'esito di una richiesta. Viene visualizzato il feedback dato dall'utente in modo tale da valutare la laboriosità del dottore.	4	
Gestione piattaforma	5	funzionale	5	Deve consentire all'admin di effettuare operazioni sulla piattaforma		5.1-2-3
Elimina utente	5.1	funzionale	5	Deve consentire all'admin di eliminare un utente e i suoi dati	5	
Inserisci dottore	5.2	funzionale	5	Deve consentire all'admin di inserire un dottore alla piattaforma	5	
Elimina dottore	5.3	funzionale	5	Deve consentire all'admin di eliminare un dottore e i suoi dati	5	
Gestione statistiche	6	funzionale	6	Deve consentire all'utente di valutare il proprio andamento mediante dei grafici		6.1-2-3
Statistiche nutrienti	6.1	funzionale	6	Deve consentire all'utente di prendere visione dei valori assunti, dei macronutrienti, al passare dei giorni mediante un grafico	6	
Statistiche Peso UPDATE	6.2	funzionale	6	Deve consentire all'utente di prendere visione dei pesi che nei vari giorni ha raggiunto mediante un grafico.	6	
Aggiungi peso UPDATE	6.3	funzionale	6	Devo consentire all'utente di poter aggiornare il proprio peso	6	

5.2 Design Phase

5.2.1 Technical Update

I requisiti appena introdotti potranno essere facilmente implementabili con un basso numero di modifiche al codice già esistente, in particolare sarà necessario innanzitutto aggiungere un nuovo pulsante all'interfaccia grafica che stamperà un pdf contenente gli elementi dell'interfaccia "resoconto" e una nuova sezione in cui un'utente visualizzerà le statistiche dei ultimi giorni in merito ai macronutrienti assunti e ai cambiamenti di peso ottenuto.

NEW LIBRARY

Per quanto riguarda la gestione delle statistiche mediante grafici, verrà utilizzata la libreria di terze parti **JFreeChart**. JFreeChart è un framework open source per il linguaggio di programmazione Java, che consente la creazione di un'ampia varietà di grafici sia interattivi che non interattivi e nel caso specifico del software verranno utilizzati grafici XY, ovvero che sfruttando gli assi cartesiani andremo a comporre un tracciato delineato dai valori inseriti.

5.2.2 Component Structure

Le modifiche rispetto al primo prototipo sono minime, in quanto l'aggiunta del componente delle statistiche, grazie alla modularità della piattaforma, viene agganciato perfettamente. La figura successiva infatti mostra come, rispetto al primo prototipo, è stato inserito il modulo delle statistiche.

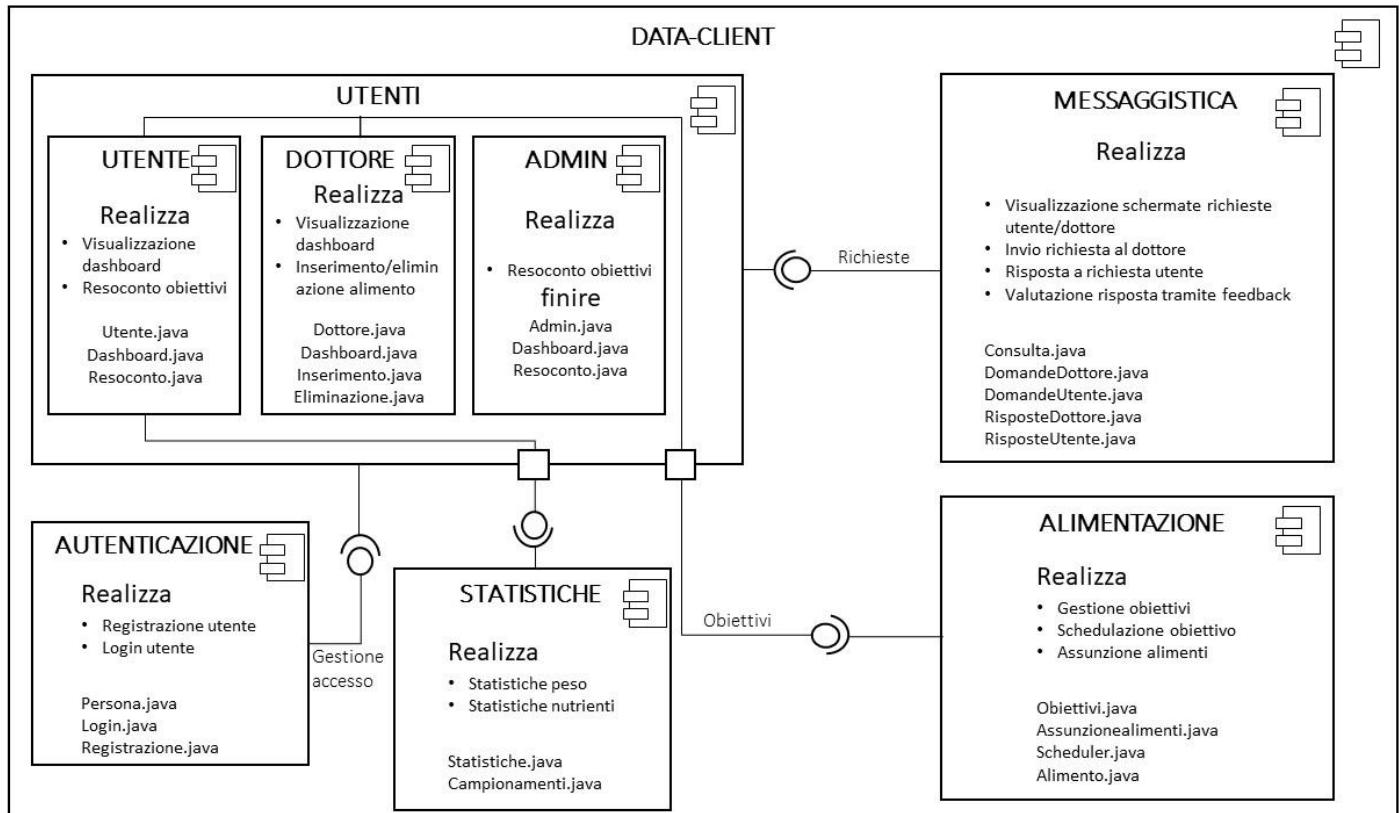


Figura 45 Component Diagram software (DATA-CLIENT UPDATE)

SECOND ITERATION (PROTOTYPE 2) - DESIGN PHASE

5.2.3 GUI Design

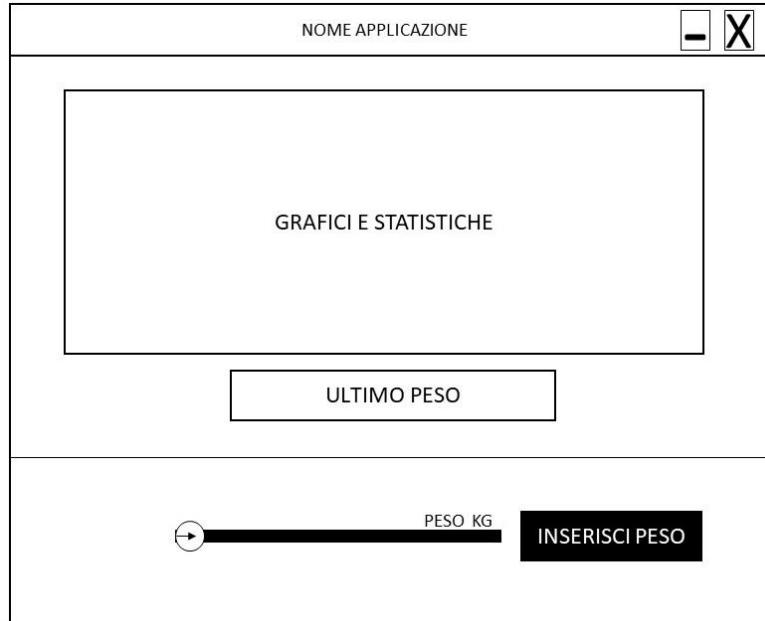


Figura 46 Interfaccia: Schermata gestione statistiche

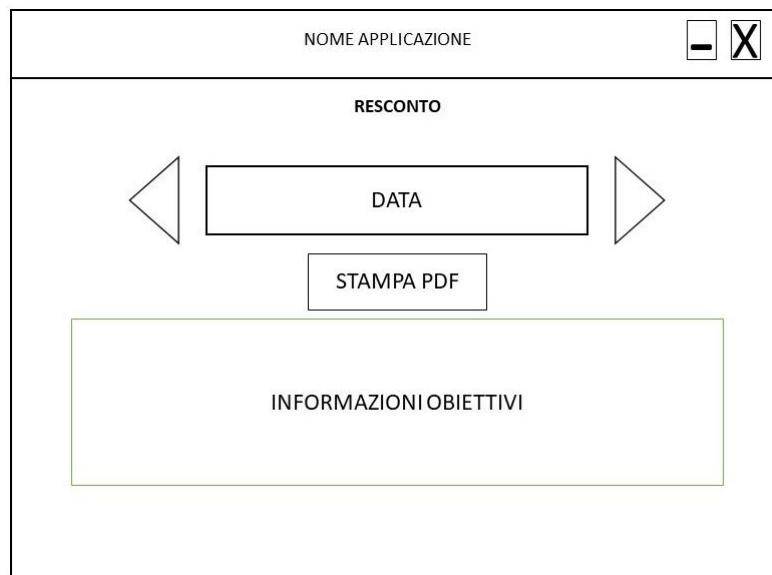


Figura 47 Interfaccia: Schermata resoconto (**UPDATE**)

5.3 Implementation Phase

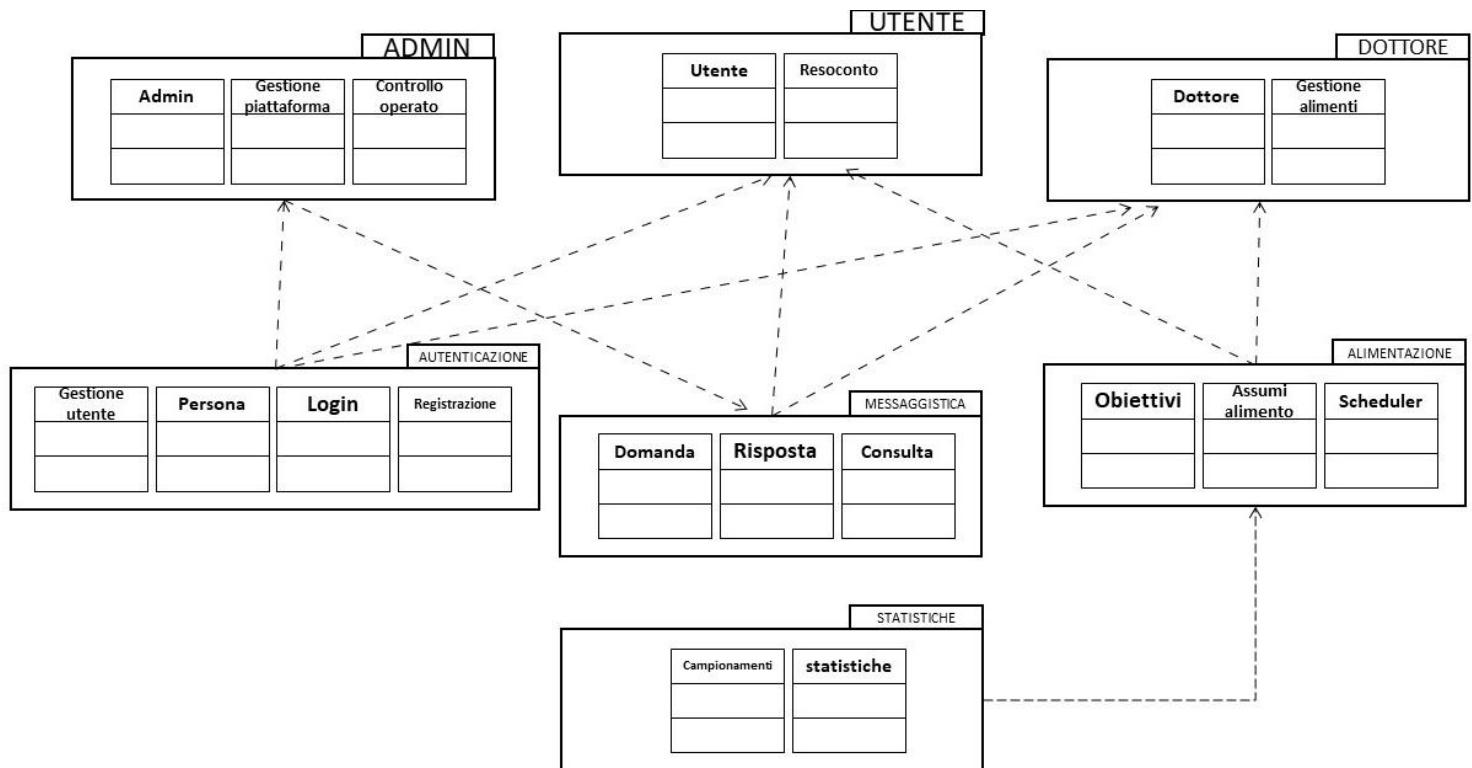


Figura 48 Package diagram (UPDATE)

PACKAGE STATISTICHE

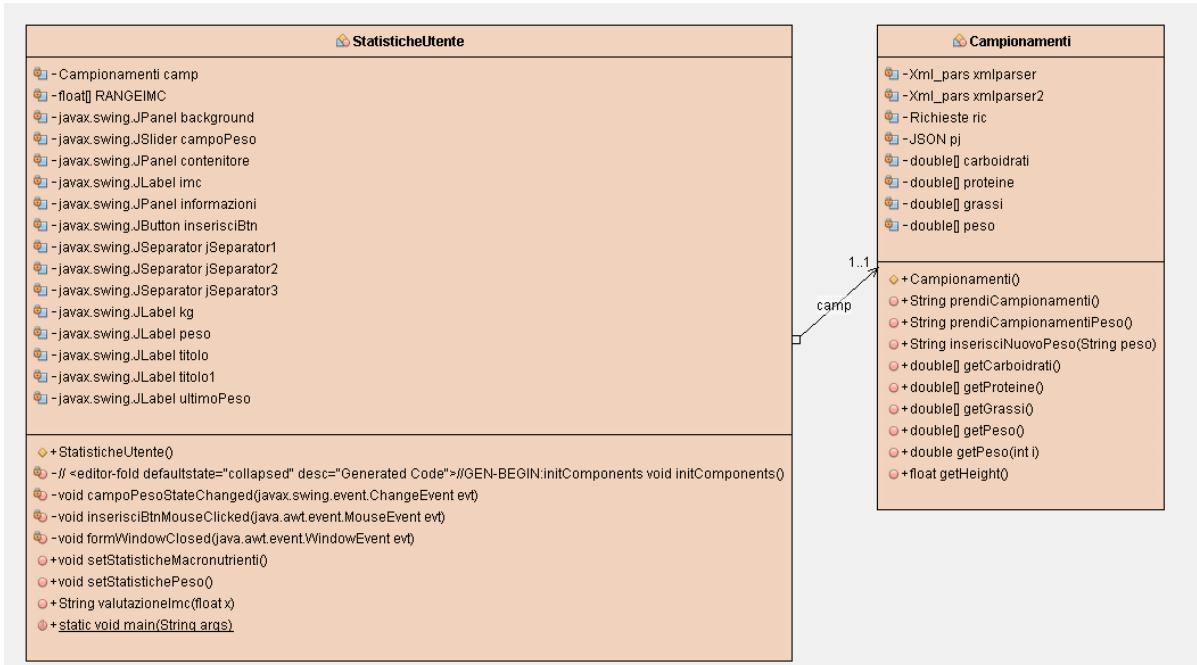
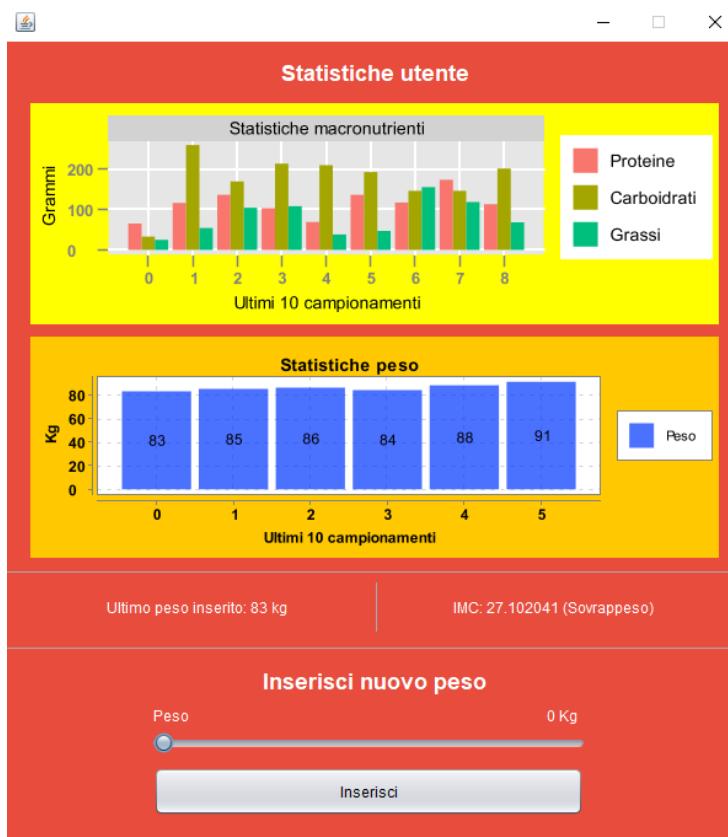
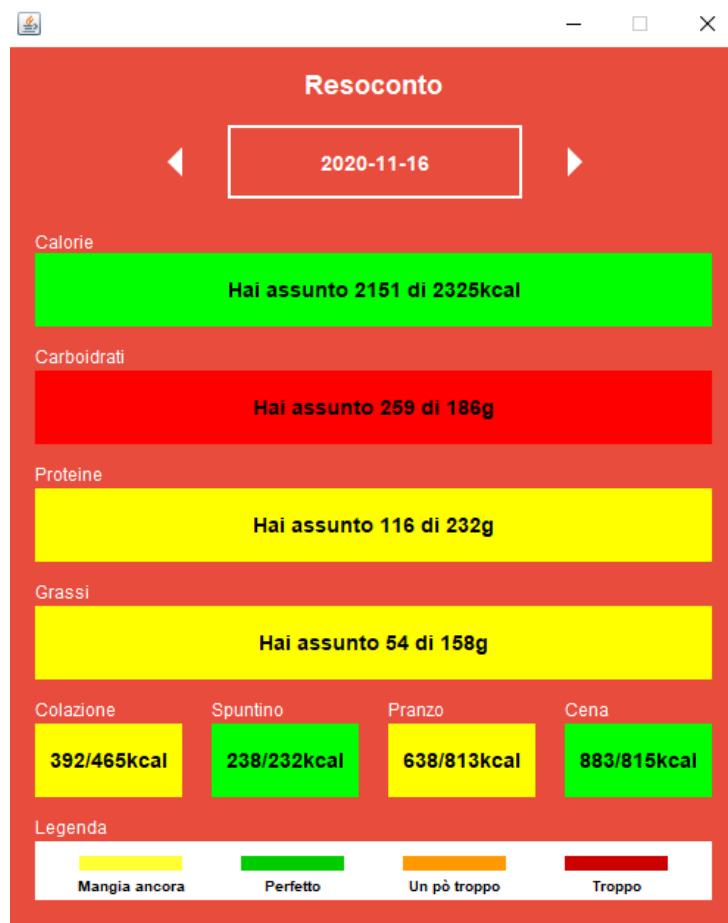


Figura 49 Class diagram: Statistiche

5.3.1 GUI's Screens



5.3.2 Software Deployment

Giunti alla fine dell'implementazione del software, il passo successivo è quello di dover rilasciare lo stesso, e dunque il deployment, in modo tale che vengano eseguiti tutti i test e le verifiche del caso effettuando un **installazione diretta**.

Il software, dopo le modifiche effettuate, avrà la seguente distribuzione:

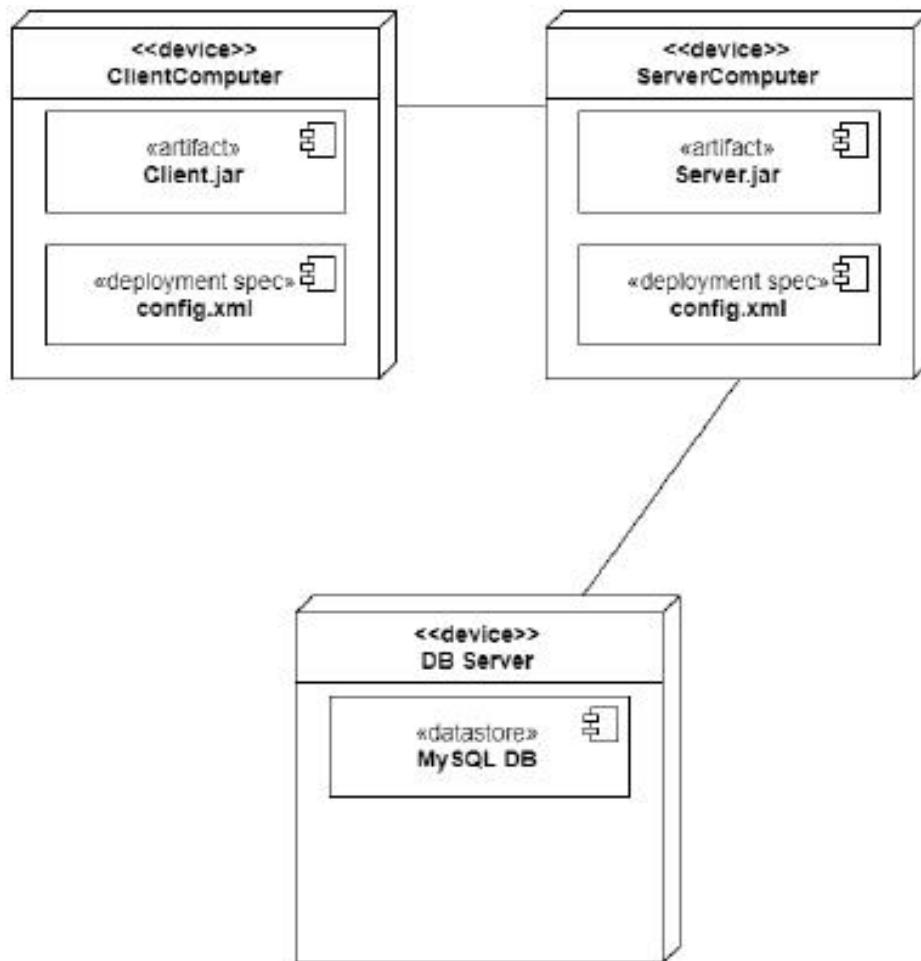


Figura 50 Deployment Diagram

5.4 Testing Phase

Avendo sviluppato un secondo prototipo è ora possibile verificare il corretto funzionamento di quest'ultimo, in modo da poterlo poi riconsegnare al cliente.

In primo luogo sarà necessario controllare che il nuovo prototipo risponda ai nuovi requisiti funzionali.

Successivamente è necessario verificare la correttezza delle parti aggiunte, verificando che il sistema, funzioni effettivamente

Con l'aggiunta del componente della statistica, viene fortemente consigliato di effettuare un **regression test** sulle parti già testate per verificare che l'aggiunta di nuovi elementi non abbia causato instabilità all'interno delle classi già presenti e che non causi comportamenti inattesi. Ci permettiamo tuttavia di omettere tale test dalla documentazione, essendo risultato di fatto una copia del testing della precedente prototipazione.

5.4.1 Validation

Per quanto riguarda la fase iniziale, bisogna innanzitutto considerare i nuovi requisiti funzionali:

RQ6: Possibilità di avere una statistica del proprio peso, al variare dei giorni

RQ7: Possibilità di avere una statistica dei macronutrienti assunti, al variare dei giorni



5.4.2 Verification

Per quanto riguarda il corretto funzionamento del software, è stata testata la molteplice stampa del resoconto giornaliero ed inoltre è stata testata la funzionalità delle statistiche, ovvero all'inserimento di un nuovo peso o di una nuova assunzione è stata verificata la correttezza del grafico.

TEST	VALORE DI INGRESSO	RISULTATO
	Sequenza nel range 0-200 kg	Corretto inserimento
	Sequenza inferiore a 0 kg	Non consentito
	Stringa superiore a 200 kg	Non consentito

5.5 Customer Review and Feedback

Dopo aver preso visione del secondo prototipo, il cliente si dimostra soddisfatto del risultato in funzione delle nuove componenti inserite. Ritenutosi soddisfatto è possibile rilasciare il prodotto.

6. Release Phase

6.1 Release 1.0

Il prototipo 2 è stato giudicato completo in ogni suo aspetto, pertanto può essere rilasciata la prima versione del prodotto finale (Release 1.0).

Eventuali successive release saranno inserite e descritte in questa stessa documentazione.

Il software è accompagnato da una Documentazione finale (**Software Documentation**), in cui è presente una breve descrizione del suo ciclo di vita e il Manuale Utente.