Laurea Triennale in informatica-Università di Salerno
Corso di *Ingegneria del Software*-Prof.ssa F. Ferrucci e Prof. F. Palomba

System Design Document

Riferimento	SDD
Versione	2.0.0
Data	11/12/2024
Destinatario	Prof.ssa Filomena Ferrucci, Prof.re Fabio Palomba
Presentato da	Team Members
Approvato da	Raffaella Spagnuolo, Alessia Ture

Revision History

Data	Versione	Descrizione	Autori
12/11/2024	0.1.0	Design Goals	Tutti i TMs
12/11/2024	0.2.0	Trade offs	Tutti i TMs
13/11/2024	0.3.0	Stesura Introduzione	FR, MR, ATg, ATr
13/11/2024	0.4.0	Stesura Architettura Software del Sistema Corrente	FR, MR, ATg, ATr
14/11/2024	0.5.0	Stesura Architettura Software del Sistema Proposto	FR, MR, ATg, ATr
15/11/2024	0.6.0	Servizi e sottosistemi	FR, MR, ATg, ATr
16/11/2024	0.7.0	Condizioni Limite	FR, MR, ATg, ATr
16/11/2024	0.8.0	Stesura Glossario	FR, MR, ATg, ATr
16/11/2024	0.9.0	Stesura Acronimi	FR, MR, ATg, ATr
17/11/2024	0.10.0	Aggiunta Diagrammi	FR, MR, ATg, ATr
18/11/2024	1.0.0	Prima Revisione	GB, BF, MM, GR
11/12/2024	2.0.0	Revisione PMs	RS

Project Managers

Nome	Acronimo	Contatto
Raffaella Spagnuolo	RS	r.spagnuolo6@studenti.unisa.it
Alessia Ture	ATu	a.ture@studenti.unisa.it

Team Members

Nome	Acronimo	Contatto
Giovanni Balzano	GB	g.balzano10@studenti.unisa.it
Benito Farina	BF	b.farina5@studenti.unisa.it
Marco Meglio	MM	m.meglio4@studenti.unisa.it
Ferdinando Ranieri	FR	f.ranieri12@studenti.unisa.it
Marco Renella	MR	m.renella1@studenti.unisa.it
Giuseppe Russo	GR	g.russo248@studenti.unisa.it
Anna Tagliamonte	ATg	a.tagliamonte9@studenti.unisa.it
Alessandra Trotta	ATr	a.trotta56@studenti.unisa.it

Sommario

1 Introduzione

1.1 Scopo del sistema

Il sistema *ZeroWaste Home* ha lo scopo di offrire una soluzione innovativa per la gestione efficiente delle risorse alimentari domestiche, rispondendo al crescente bisogno di ridurre gli sprechi alimentari. Il sistema permette agli utenti di monitorare in tempo reale il contenuto del frigorifero e della dispensa, gestire le scadenze degli alimenti e ricevere suggerimenti su come utilizzare gli ingredienti prossimi alla scadenza. Questo approccio contribuisce a ottimizzare le scorte domestiche e a migliorare la pianificazione dei consumi alimentari, con una significativa riduzione dello spreco di cibo.

I principali utenti del sistema includono:

- **Famiglie e individui** che desiderano ridurre lo spreco alimentare e risparmiare tempo e denaro,
- **Consumatori attenti alla sostenibilità**, che sono sensibili agli impatti ambientali degli sprechi e interessati a ridurre la propria impronta ecologica.

Il valore principale che *ZeroWaste Home* porta agli utenti risiede nella capacità di trasformare le abitudini alimentari verso uno stile di vita più sostenibile, semplificando la gestione delle scadenze e offrendo incentivi tangibili per ridurre lo spreco. Inoltre, attraverso una community digitale, il sistema incoraggia la condivisione di ricette e suggerimenti per l'utilizzo di ingredienti in scadenza, creando un'esperienza collaborativa e promuovendo una cultura di consumo più consapevole.

1.2 Design Goals

Gli obiettivi principali di design di *ZeroWaste Home* sono orientati a garantire usabilità, affidabilità, performance, e manutenibilità del sistema. Ogni obiettivo è stato identificato per rispondere a requisiti non funzionali (RNF) e migliorare l'esperienza utente, la gestione dei dati e la scalabilità. Di seguito sono descritti gli obiettivi principali:

Rank	ID Design Goal	Descrizione	Categoria	RNF di origine
1	**DG_10 Facile manutenzione**	Il sistema deve facilitare la manutenzione tramite architettura modulare con commenti e documentazione correlata.	Maintenance	RNF_SU_01, RNF_SU_02
2	**DG_01 Navigazione e Usabilità dell'Interfaccia**	Il sistema deve offrire un'interfaccia intuitiva, con navigazione semplice (max 4 clic), FAQ, e una chiara guida per il processo di pagamento e gestione errori.	End User Criteria	RNF_US_01, RNF_US_02, RNF_US_03
3	**DG_02 Feedback Visivo e Informativo**	Il sistema deve fornire feedback visivo durante l'interazione (moduli e pagamenti) e offrire informazioni chiare sulle operazioni effettuate.	End User Criteria	RNF_US_04, RNF_US_05, RNF_US_08, RNF_US_09
4	**DG_03 Sicurezza dei Dati e Crittografia**	Il sistema deve implementare un algoritmo per la gestione dei dati che garantisca la sicurezza e la riservatezza delle informazioni degli utenti. Tutti i dati sensibili, incluse le credenziali di accesso, devono essere protetti, in modo da prevenire accessi non autorizzati e proteggere la privacy degli utenti.	Dependability	RNF_RE_10

5	**DG_04 Gestione Affidabile dei Dati**	Il sistema deve garantire la convalida degli input e l'aggiornamento dei dati validati gestendo correttamente le eccezioni entro 1 secondo.	Dependability	RNF_RE_01, RNF_RE_04, RNF_RE_05, RNF_RE_06
6	**DG_05 Capacità e prestazioni del sistema**	Il sistema deve supportare almeno 500 utenti simultanei, rispondendo entro 2 secondi per operazioni generali, e completare i pagamenti in meno di 10 secondi.	Performance	RNF_PR_01, RNF_PR_02, RNF_PR_03
7	**DG_06 Scalabilità e ottimizzazione del carico**	Il sistema deve scalare facilmente, mantenendo prestazioni ottimali con tempi di caricamento sotto i 3 secondi e risposte inferiori a 1 secondo per modifiche alla lista della spesa, anche sotto carico elevato.	Performance	RNF_PR_04, RNF_PR_05, RNF_PR_06
8	**DG_07 Gestione dati utenti**	Il sistema deve garantire backup orari dei dati, il salvataggio della lista della spesa e la memorizzazione delle preferenze alimentari dell'utente, assicurando integrità, persistenza e ripristino in caso di guasto.	Dependability	RNF_RE_02, RNF_RE_07, RNF_RE_09
9	**DG_08 Interfaccia Responsive**	Il sistema deve avere un'interfaccia che si adatti ai vari dispositivi con un layout responsive e tecnologie moderne.	End User Criteria	RNF_US_10

10	**DG_09 Supporto Utente per la Lista e le Statistiche**	Il sistema deve include la funzione di ricerca per la lista della spesa e una visualizzazione chiara delle statistiche ambientali.	End User Criteria	RNF_US_06, RNF_US_07
11	**DG_11 Facilmente aggiornabile**	Il sistema deve permettere l'aggiornamento del database senza impatti sulle altre funzionalità.	Maintenance	RNF_SU_03, RNF_SU_04
12	**DG_12 Efficienza del calcolo dell'impatto**	Il sistema deve effettuare il calcolo dell'impatto ambientale in meno di 2 secondi per articoli scartati.	Performance	RNF_PR_07
13	**DG_13 Gestione errori**	Il sistema deve includere un sistema di monitoraggio delle performance e degli errori per garantire una pronta risposta a eventuali problemi, se si verifica un errore di pagamento il sistema offre soluzioni alternative.	Dependability	RNF_RE_03, RNF_RE_08

1.2.1 Trade-Offs

Nel progetto *ZeroWaste Home*, i **trade-offs** rappresentano le scelte tra diverse esigenze del sistema, dove migliorare un aspetto può influire negativamente su un altro. Questi compromessi sono inevitabili nello sviluppo, e richiedono una valutazione attenta per bilanciare requisiti come affidabilità, prestazioni, usabilità e tempi di rilascio. Gestire questi compromessi significa assicurare che il sistema soddisfi le necessità principali degli utenti.

Trade-Offs	Descrizione
Affidabilità vs Prestazioni	Per garantire l'affidabilità (backup dei dati, aggiornamenti precisi), il sistema potrebbe richiedere risorse aggiuntive, influendo sulle prestazioni. Funzionalità di affidabilità potrebbero rallentare i tempi di risposta, specialmente sotto carico. Tuttavia, **la priorità viene data all'affidabilità**, poiché è fondamentale assicurare l'integrità e la sicurezza dei dati anche a costo di tempi di risposta leggermente più lunghi.
Usabilità vs Performance	Al fine di migliorare l'usabilità, il sistema fornirà un'interfaccia utente intuitiva, con funzionalità come feedback visivi, guide interattive e una navigazione chiara. Tuttavia, l'implementazione di queste funzionalità potrebbe rallentare i processi, come il caricamento delle pagine o il tempo di risposta. In questo caso, **la priorità viene data all'usabilità**, poiché un'esperienza utente soddisfacente è cruciale per l'adozione e il successo del sistema, anche se ciò comporta un sacrificio delle performance.
Manutenibilità vs Tempo di rilascio	Per garantire una manutenzione facile e rapida nel tempo, il sistema deve essere sviluppato con una struttura modulare, codice ben documentato e un'architettura facilmente estensibile. Questo richiede più tempo di sviluppo e risorse, ritardando il rilascio. **La priorità viene data alla manutenibilità**, poiché un sistema che può essere facilmente aggiornato e mantenuto è più sostenibile nel lungo periodo, anche a scapito di ritardi iniziali nel rilascio.
Dependability vs Tempo di rilascio	Per aumentare la dependability del sistema, implementiamo caratteristiche come back-up continui, gestione degli errori in tempo reale e tolleranza ai guasti. Queste misure richiedono tempo e risorse, ritardando il rilascio del progetto. **La priorità viene data alla dependability**, poiché un sistema affidabile e sicuro fin dall'inizio è essenziale per evitare problemi critici a lungo termine.

1.3 Definizioni, acronimi e abbreviazioni

Definizioni

Questo Glossario fornisce una raccolta di termini tecnici e concetti chiave utilizzati nel progetto.

Acronimi

- **DG**: Design Goals
- **SDD**: System Design Document
- **GUI**: Graphical User Interface
- **DAO**: Data Access Object
- **UCB**: Use Condition Boundary
- **DBMS MySQL:** DataBase Management System My Structured Query Language
- **RNF**: Requisiti Non Funzionali
- **FAQ**: Frequently Asked Questions
- **API**: Application Programming Interface – Interfaccia per l'integrazione tra componenti software.

Abbreviazioni

- **TTAL:** Three-Tier Architecture Layer
- **RNF_US**: Requisiti Non Funzionali di Usabilità
- **RNF_RE**: Requisiti Non Funzionali di Affidabilità
- **RNF_PR**: Requisiti Non Funzionali di Performance
- **RNF_SU**: Requisiti Non Funzionali di Manutenzione
- **PT**: Presentation Tier- Interfaccia utente o GUI (Grapichal User Interface)
- **LT**: Logic Tier- Livello logico o intermedio, che gerstisce la logica applicativa
- **BE**: Backend Tier- Livello di gestione dei dati, solitamente un database o un sistema di archiviazione
- **UI**: User Experience / User Interface – Esperienza utente e interfaccia grafica

1.4 Riferimenti

- Requirement Analysis Document, versione 2.0.0
- Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition - Bernd Bruegge & Allen H. Dutoit

1.5 Panoramica del documento

Questo documento di progettazione del sistema descrive l'architettura e le principali caratteristiche del progetto *ZeroWaste Home*, un'applicazione per la gestione sostenibile delle risorse alimentari domestiche.

Il documento è strutturato come segue:

- **Introduzione:** Questa sezione fornisce una panoramica del sistema *ZeroWaste Home,* descrivendone lo scopo principale e i benefici attesi per gli utenti. Viene spiegato come il sistema favorisca una gestione sostenibile delle scorte alimentari e riduca gli sprechi, permettendo agli utenti di monitorare in tempo reale il contenuto del frigorifero e della dispensa. In questa sezione vengono elencati gli obiettivi chiave del design, tra cui usabilità, affidabilità e scalabilità, con una valutazione dei compromessi necessari per ottimizzare le prestazioni. Sono inoltre incluse definizioni, acronimi e abbreviazioni per uniformare la terminologia, seguiti dai riferimenti a fonti e documenti rilevanti per il progetto. Un breve riassunto dei contenuti delle varie sezioni orienta infine il lettore nella comprensione del documento;

- **Architettura Software del Sistema Corrente:** Questa sezione analizza i sistemi simili esistenti, per mostrare come *ZeroWaste Home* si differenzi per caratteristiche e funzionalità innovative. Vengono esaminati i limiti e le potenzialità dei sistemi attuali, evidenziando in che modo *ZeroWaste Home* combina e amplia funzionalità non ancora presenti in un'unica applicazione, come la gestione in tempo reale delle scorte, il monitoraggio della data di scadenza e la fornitura di suggerimenti per un uso efficiente degli alimenti;

- **Architettura del Sistema Proposto:** In questa sezione si approfondisce l'architettura del sistema *ZeroWaste Home*, descrivendone lo stile (es., architettura a tre livelli o client-server) e le scelte tecnologiche effettuate. Sono presentati i principali sottosistemi, come la gestione utente, l'interfaccia, la gestione dei dati, e viene illustrato un diagramma che mostra le relazioni tra essi. Viene descritto in dettaglio come i componenti hardware e software sono mappati tra loro, supportato da un diagramma di deployment che visualizza la distribuzione fisica dei componenti software. Infine, la sezione copre i meccanismi di autenticazione e sicurezza del sistema, descrive il controllo del flusso e il coordinamento tra componenti e sottosistemi, e specifica le condizioni limite per valutare il sistema in situazioni di carico elevato o utilizzo intenso;

- **Servizi dei Sottosistemi**: Questa sezione dettaglia i servizi forniti da ciascun sottosistema per soddisfare i requisiti funzionali e non funzionali del sistema. Viene spiegato come ogni sottosistema contribuisce alla realizzazione degli obiettivi di *ZeroWaste Home*, per garantire una gestione efficiente e affidabile delle scorte alimentari e del monitoraggio dell'impatto ambientale;

- **Glossario**: Il glossario contiene un elenco di termini tecnici e relative definizioni, per facilitare la comprensione dei contenuti del documento.

2 Architettura software del sistema corrente

Attualmente, esistono sistemi che offrono alcune funzionalità simili a quelle proposte da *ZeroWaste Home*, come applicazioni per la gestione delle scadenze dei prodotti alimentari o per il monitoraggio della dispensa. Tuttavia, nessun sistema attuale integra tutte le funzionalità previste in una singola applicazione, come il monitoraggio in tempo reale delle scorte, la gestione delle scadenze, la condivisione di ricette basate sugli ingredienti disponibili, e un sistema di incentivi per comportamenti sostenibili.

ZeroWaste Home rappresenta dunque una soluzione unica e innovativa che combina:

- **Gestione delle Scorte e delle Scadenze**: Monitoraggio automatizzato di frigorifero e dispensa con notifiche per i prodotti in scadenza,
- **Suggerimenti Basati sulle Scadenze**: Raccomandazioni per l'uso di ingredienti prossimi alla scadenza, evitando così lo spreco,
- **Community di Condivisione**: Spazio dedicato per la condivisione di ricette e consigli tra utenti,
- **Sistema di Incentivi per la Sostenibilità**: Calcolo dell'impatto ambientale degli sprechi e premi per gli utenti che riducono i rifiuti.

3 Architettura software del sistema proposto

3.1 Panoramica

Il sistema sarà sviluppato seguendo lo stile architetturale **Three-Tier**, che consente una chiara separazione tra i vari livelli per migliorare l'organizzazione e la scalabilità del sistema. In particolare, le componenti saranno suddivise come segue:

- **Livello delle Interfacce (Presentation Layer)**: Questa è la parte che interagisce direttamente con l'utente e ne gestisce la comunicazione con il sistema. Il front-end sarà sviluppato utilizzando **Angular**, un framework versatile che consente di creare un'interfaccia utente semplice, reattiva e responsiva, ottimizzata per un'esperienza d'uso intuitiva. La scelta di Angular garantisce un rendering efficiente delle viste, soprattutto in contesti dinamici, e una struttura modulare che facilita l'espansione futura;
- **Livello di Logica Applicativa (Business Logic Layer)**: In questo livello risiederanno i componenti di controllo e di logica, ovvero le parti del sistema responsabili dell'elaborazione dei dati e dell'applicazione delle regole di business. Il back-end di questa componente sarà sviluppato con il framework **Spring**, che offre un'infrastruttura solida e scalabile per la gestione dei servizi e delle operazioni aziendali. Spring permette di implementare facilmente principi di sicurezza, transazioni e gestione delle sessioni, elementi fondamentali per un sistema come *ZeroWaste Home*. Utilizzando Spring Boot, sarà possibile avviare rapidamente servizi RESTful che comunicano con il front-end Angular, assicurando flessibilità e modularità. L'uso di Spring garantisce anche un'alta affidabilità, supportando la gestione di carichi di lavoro simultanei e facilitando l'integrazione con il livello di persistenza dati;
- **Livello dei Dati (Data Layer)**: Questo livello si occupa della gestione dei dati persistenti, ovvero delle informazioni che vengono salvate e recuperate in modo sicuro e duraturo. Per questo, verrà utilizzato il **DBMS MySQL**, che offre affidabilità, flessibilità e velocità nelle operazioni su database. MySQL garantisce anche una scalabilità adeguata per supportare un aumento del volume dei dati e delle operazioni, essenziale per un sistema come *ZeroWaste Home*.

Le tecnologie selezionate per questo progetto rispondono in modo efficace ai requisiti del sistema:

- **Efficienza e Scalabilità**: La combinazione di Angular per il front-end e Spring per il back-end consente di mantenere elevate prestazioni e di scalare facilmente il sistema. Angular permette di aggiornare solo le porzioni dell'interfaccia necessarie, mentre Spring, insieme al server applicativo, gestisce le richieste con un'alta efficienza;
- **Interfaccia Utente Responsive**: **Angular** permette di progettare un'interfaccia semplice, reattiva e intuitiva per l'utente, favorendo un'interazione fluida e piacevole;
- **Affidabilità nella Gestione dei Dati**: **MySQL** è ideale per garantire la persistenza, l'integrità e l'affidabilità delle informazioni archiviate, assicurando che i dati degli utenti siano sicuri e accessibili in modo rapido e scalabile.

Questa combinazione tecnologica di **Angular, Spring, e MySQL** è stata scelta per soddisfare le esigenze di scalabilità, affidabilità e sicurezza necessarie per il progetto *ZeroWaste Home*, offrendo allo stesso tempo un'architettura modulare che facilita la manutenzione e l'evoluzione del sistema nel tempo. In aggiunta, per la gestione dei codici a barre, il sistema integrerà il servizio **Open Food Facts**, che permetterà di accedere a un database pubblico per ottenere informazioni sugli alimenti tramite il codice a barre, semplificando la gestione e catalogazione dei prodotti alimentari nel sistema *ZeroWaste Home*.

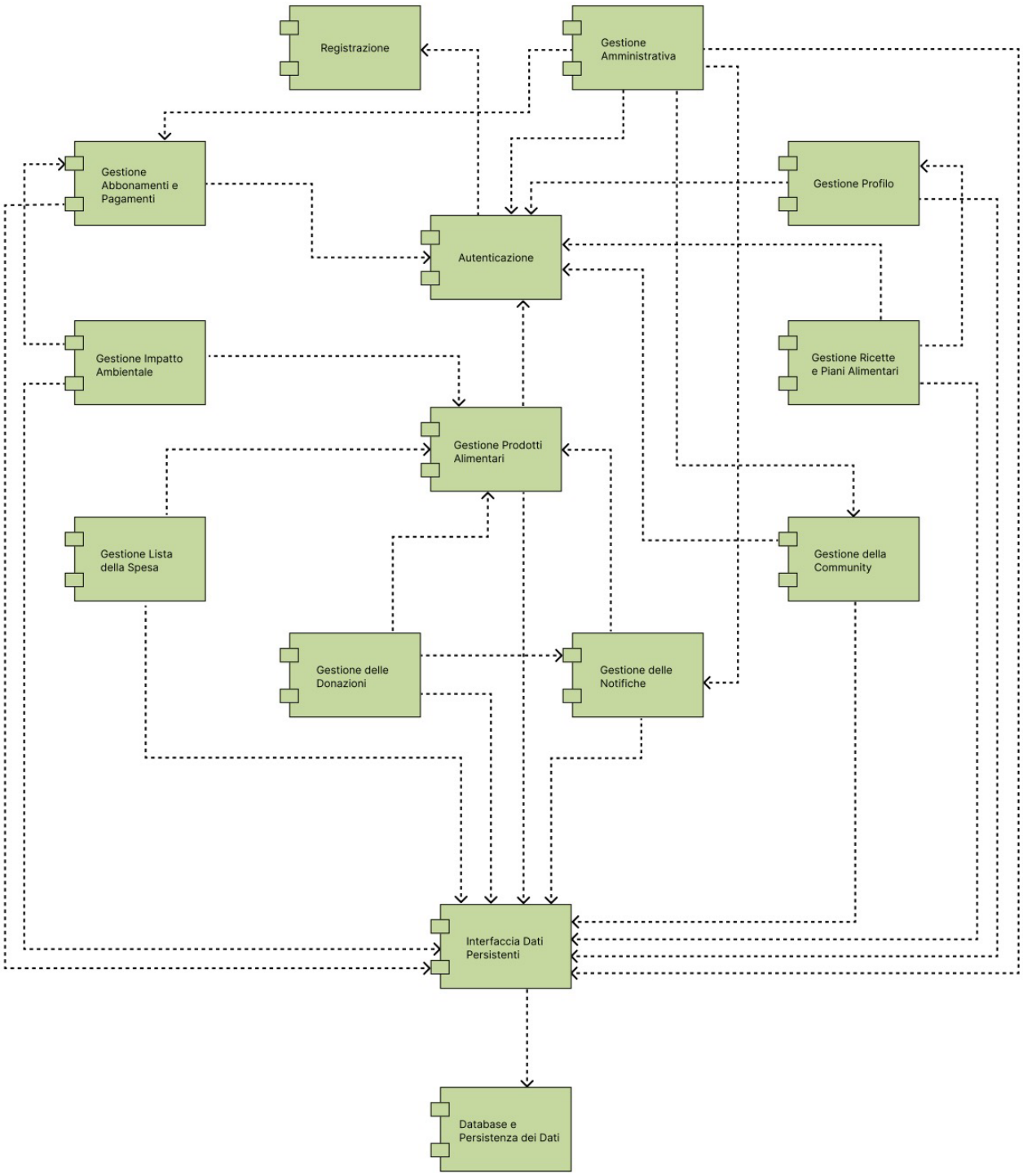
3.2 Decomposizione in sottosistemi

Basandoci sui requisiti e sugli scenari d'uso precedentemente individuati, il sistema *ZeroWaste Home* è stato suddiviso nei seguenti sottosistemi principali:

- **Registrazione:** Si occupa di gestire la registrazione degli utenti, consentendo loro di creare un account per accedere ai servizi dell'applicazione. Il sistema supporta la registrazione di due tipi di utenti:
  - Utente Standard: utente che utilizza il sistema per gestire i propri prodotti alimentari e ridurre gli sprechi,
  - Ente di Beneficenza: utente che accetta donazioni di prodotti alimentari;
- **Autenticazione:** Si occupa di gestire l'accesso al sistema da parte di ogni tipo di utente:
  - Utente Standard,
  - Ente di Beneficenza,
  - Gestore Utente,
  - Gestore Community;
- **Gestione Profilo:** Si occupa della modifica e visualizzazione delle informazioni del profilo degli utenti;
- **Gestione Prodotti**: Si occupa di inserimento, aggiornamento e visualizzazione delle informazioni sui prodotti presenti nel frigorifero e nella dispensa. Questo sottosistema è alla base della gestione delle scorte alimentari e della riduzione degli sprechi;
- **Gestione delle Notifiche**: Si occupa della gestione delle notifiche, monitora le date di scadenza dei prodotti presenti in frigo e dispensa, notificando l'utente per evitare sprechi alimentari.
  Si occupa inoltre della visualizzazione delle notifiche delle donazioni per enti di beneficenza;
- **Gestione Impatto Ambientale**: Si occupa della visualizzazione di una panoramica sull'andamento del consumo alimentare; calcola e visualizza l'impatto ambientale generato dagli sprechi alimentari dell'utente. In base agli alimenti scartati, assegna un malus e aggiorna la sezione dei progressi, incentivando comportamenti sostenibili;
- **Gestione della Community**: Si occupa della visualizzazione e pubblicazione di ricette sulla community; consente agli utenti di condividere, commentare o visualizzare ricette sulla community. Si occupa inoltre della segnalazione di ricette con contenuto inadeguato;
- **Gestione Ricette e Piano Alimentare**: Si occupa della creazione di piani alimentari settimanali personalizzati. In questo modo, gli utenti possono pianificare i pasti e ridurre gli sprechi alimentari. Include la creazione e il suggerimento di ricette sulla base dei prodotti disponibili;

- **Gestione Lista della Spesa**: Si occupa della generazione, aggiornamento e condivisione della lista della spesa. Genera automaticamente una lista della spesa basata sulle scorte attuali e sugli articoli mancanti per realizzare le ricette selezionate. Facilita la gestione degli acquisti e ottimizza il rifornimento degli alimenti;
- **Gestione delle Donazioni**: Si occupa della donazione di alimenti in eccedenza a enti di beneficenza locali, offrendo una mappa interattiva e permettendo la selezione di enti specifici che accettano cibo. Questo contribuisce ulteriormente alla riduzione degli sprechi;
- **Gestione Abbonamenti e Pagamenti**: Si occupa della gestione degli abbonamenti al servizio, includendo la selezione dei piani, inserimento e modifica di metodi di pagamento. Si occupa inoltre della segnalazione di eventuali problemi legati ai pagamenti;
- **Gestione Amministrativa:** Si occupa della gestione del sistema, includendo la visualizzazione, sospensione ed eliminazione degli utenti. Si occupa inoltre della visualizzazione e gestione delle segnalazioni e della visualizzazione delle statistiche sull'utilizzo del sistema da parte degli utenti;
- **Interfaccia dati persistenti**: Si occupa di fare intermediario tra i vari sottosistemi e la gestione dei dati persistenti;
- **Database e Persistenza dei Dati**: Si occupa della memorizzazione e l'accesso a tutti i dati persistenti del sistema, come i dati utente, le scorte alimentari, le ricette, le liste della spesa e le donazioni. Questo sottosistema garantisce la disponibilità dei dati necessari per tutte le funzionalità, assicurandone consistenza e integrità.

3.2.1 Diagramma dei sottosistemi

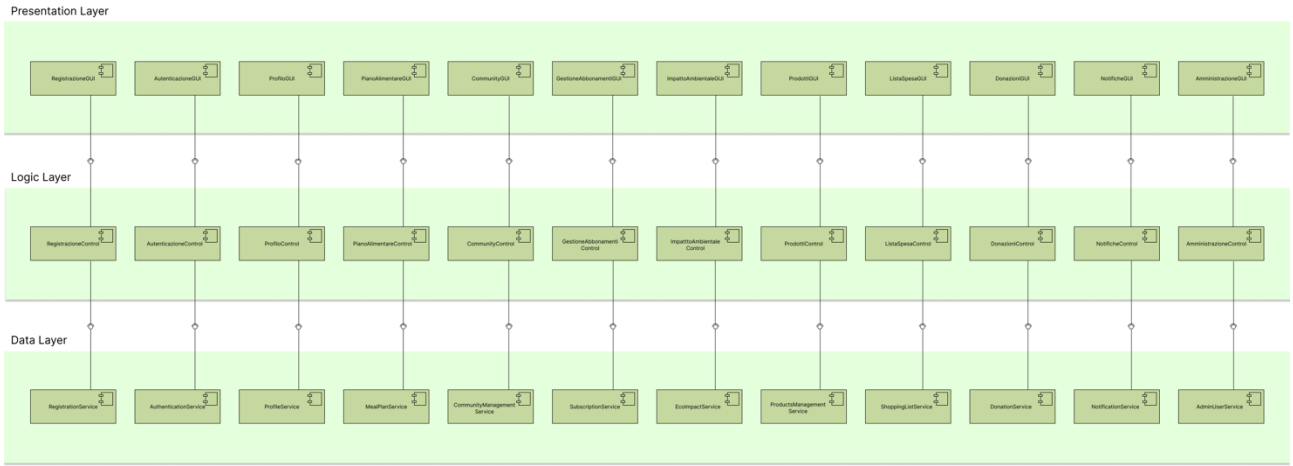


3.2.2 Diagramma Architetturale

Come descritto in precedenza, il seguente Diagramma Architetturale suddivide il sistema in tre livelli principali:

- **Presentation Layer (GUI):** che gestisce l'interazione con l'utente per ciascun sottosistema, come ad esempio Registrazione, Autenticazione, Profilo, Community, Prodotti, Lista della Spesa, Donazioni, Notifiche, Amministrazione, Gestione Abbonamenti e Impatto Ambientale;
- **Logic Layer (Control):** che implementa la logica di business per ciascun modulo. Ogni sottosistema ha un proprio componente di controllo, come RegistrazioneControl, ProfiloControl, CommunityControl, ProdottiControl, ecc., per elaborare i dati e coordinare le operazioni richieste dal Presentation Layer;
- **Data Layer (Service):** che fornisce l'accesso ai dati persistenti. I servizi come RegistrationService, AuthenticationService, ProfileService, e altri, mettono a disposizione i dati per la logica di business;

Questa architettura garantisce una struttura modulare e scalabile, in cui ogni livello è responsabile di specifiche funzionalità, migliorando la manutenibilità e l'efficienza del sistema.



3.3 Mapping Hardware/Software

Il sistema *ZeroWaste Home* è progettato come un'applicazione web accessibile online, che consente agli utenti di interagire con le funzionalità del sistema da qualsiasi dispositivo dotato di browser. Gli utenti possono così gestire le scorte alimentari, monitorare le scadenze e accedere ad altre funzionalità direttamente tramite un'interfaccia web.

La **gestione dei dati persistenti** sarà supportata da un server centrale connesso a un database. Il database utilizzato sarà MySQL, scelto per la sua affidabilità e velocità nell'accesso ai dati, garantendo integrità e coerenza anche in situazioni di carico elevato. Il server applicativo si occuperà della gestione della logica di business, mentre i dispositivi client gestiranno l'interfaccia utente.

L'applicazione sarà composta da un frontend sviluppato in **Angular** e un **backend** basato su **Spring Boot**. L'applicazione Angular sarà ospitata su un **Web Server Nginx**, che avrà il compito di distribuire e servire i file statici al Web Browser. Nginx fungerà anche da **reverse proxy**, inoltrando le richieste per le **API REST** al server **Spring Boot**. La comunicazione tra il frontend e Nginx avverrà tramite il protocollo **HTTPS**, garantendo che tutti i dati scambiati siano protetti e crittografati.

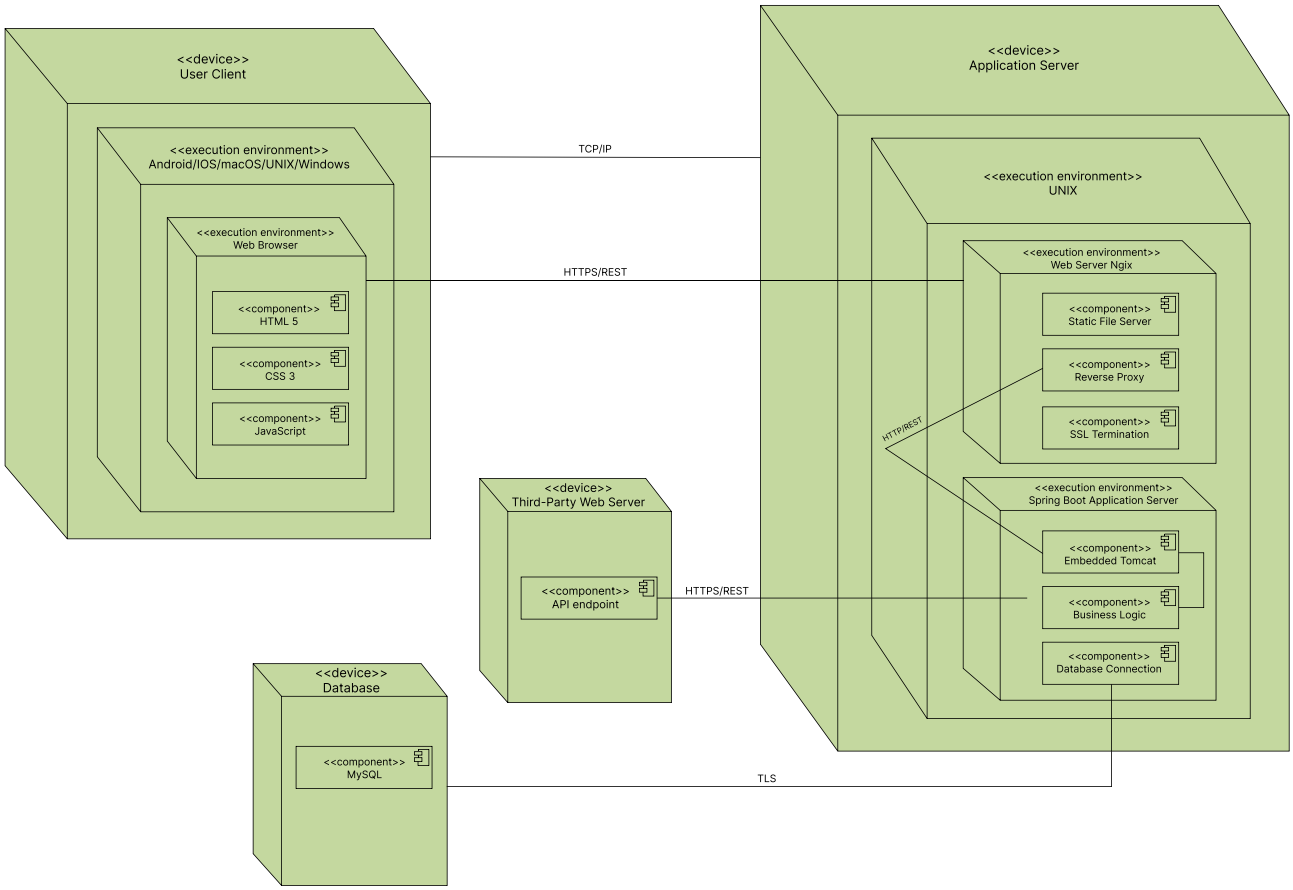
Il backend, basato su **Spring Boot**, gestirà le **API REST** e si collegherà al **DBMS MySQL** tramite una connessione **sicura e ottimizzata**. Inoltre, per ottenere dati sui prodotti tramite codice a barre, il sistema si integrerà con un server di terza parte che espone endpoint API per l'accesso ai dati sui prodotti. La comunicazione con questo server avverrà anch'essa tramite HTTPS, assicurando la protezione dei dati scambiati.

Questa architettura consente una **chiara separazione tra frontend e backend**, migliorando le performance e facilitando la manutenzione, oltre a rendere l'applicazione facilmente scalabile. Il sistema risiederà su un singolo nodo, semplificando l'implementazione e riducendo i tempi di risposta.

Il **razionale** alla base di questa architettura si fonda su principi di efficienza e sicurezza. Nginx gestisce le richieste esterne in modo sicuro, fungendo da primo livello di protezione per il sistema. Successivamente, inoltra le richieste al backend Spring Boot a livello locale, eliminando la necessità di una connessione HTTPS interna e riducendo così l'overhead associato alla crittografia. Inoltre, Nginx è significativamente più efficiente e scalabile rispetto a Tomcat nella distribuzione dei file statici del frontend, migliorando le performance complessive del sistema. La collocazione di Nginx e Spring Boot sullo stesso nodo ottimizza la comunicazione interna, rendendola più rapida e diretta, oltre a semplificare la manutenzione, creando un'architettura compatta e facilmente gestibile.

3.3.1 Deployment Diagram

Di seguito è presentato un **UML Deployment Diagram** che illustra il mapping tra hardware e software del sistema *ZeroWaste Home*.



3.4 Controllo degli accessi e sicurezza

Questa sezione presenta la matrice degli accessi, che indica i permessi concessi a ciascun attore del sistema per accedere ai diversi servizi offerti da *ZeroWaste Home*. La tabella utilizza una legenda specifica per distinguere gli accessi autorizzati da quelli non autorizzati, permettendo di visualizzare in modo chiaro chi può compiere determinate azioni e chi no. Gli attori principali includono l'Utente non Registrato, l'Utente Registrato, il Gestore Community, il Gestore Utenti e l'Ente Benefico. Questa configurazione di accesso garantisce che ciascun attore disponga solo dei permessi necessari, preservando la sicurezza e l'integrità del sistema.

Legenda

✗ = indica che l'attore non ha il permesso per l'azione.

✓ = indica che l'attore ha il permesso per l'azione.

Attori / Oggetti	Utente non Registrato	Utente Registrato	Gestore Community	Gestore Utenti	Ente Benefico
Registrazione					
Registrazione	✓	✗	✗	✗	✗
Autenticazione					
Autenticazione	✗	✓	✓	✓	✓
Gestione Profilo					
Visualizza Dati Personali	✗	✓	✓	✓	✓
Modifica Dati Personali	✗	✓	✓	✓	✓
Elimina Account	✗	✓	✗	✗	✓
Logout	✗	✓	✓	✓	✓
Recupero Password	✗	✓	✗	✗	✓
Gestione Prodotti					
Aggiungi Prodotto Frigo	✗	✓	✗	✗	✗
Modifica Prodotto Frigo	✗	✓	✗	✗	✗
Elimina Prodotto Frigo	✗	✓	✗	✗	✗

Visualizza Prodotti Frigo	✗	✓	✗	✗	✗
Aggiungi Prodotto Dispensa	✗	✓	✗	✗	✗
Modifica Prodotto Dispensa	✗	✓	✗	✗	✗
Elimina Prodotto Dispensa	✗	✓	✗	✗	✗
Visualizza Prodotti Dispensa	✗	✓	✗	✗	✗
Ricerca Prodotti Per Nome	✗	✓	✗	✗	✗
Gestione delle Notifiche					
Notifiche Scadenze	✗	✓	✗	✗	✗
Notifiche Donazioni	✗	✗	✗	✗	✓
Visualizzazione Notifiche Ricevute	✗	✓	✗	✗	✓
Gestione della Community					
Condivisione Ricette	✗	✓	✗	✗	✗
Modifica Ricette	✗	✓	✗	✗	✗
Segnalazione Contenuti	✗	✓	✗	✗	✗
Visualizzazione Ricette	✗	✓	✓	✗	✗
Visualizzazione Segnalazioni Effettuate	✗	✓	✗	✗	✗
Gestione Ricette e Piano Alimentare					
Creazione Piano Alimentare	✗	✓	✗	✗	✗

Modifica Piano Alimentare	✗	✓	✗	✗	✗
Visualizzazione Piano Alimentare	✗	✓	✗	✗	✗
Creazione Ricetta	✗	✓	✗	✗	✗
Visualizzazione Suggerimenti Ricette	✗	✓	✗	✗	✗
Gestione Lista della Spesa					
Generazione Lista della Spesa	✗	✓	✗	✗	✗
Modifica Lista della Spesa	✗	✓	✗	✗	✗
Visualizzazione Lista della Spesa	✗	✓	✗	✗	✗
Gestione Donazioni					
Visualizzazione Enti Disponibili	✗	✓	✗	✓	✗
Donazione Prodotti	✗	✓	✗	✗	✗
Tracciamento Donazioni Effettuate	✗	✓	✗	✗	✗
Tracciamento Donazione Ricevute	✗	✗	✗	✗	✓
Gestione Impatto Ambientale					
Calcolo Impatto Ambientale	✗	✓	✗	✗	✗
Visualizzazione Impatto Ambientale	✗	✓	✗	✗	✗
Suggerimenti per Ridurre Sprechi	✗	✓	✗	✗	✗
Gestione Abbonamenti e Pagamenti					
Gestione Piano Abbonamenti	✗	✓	✗	✗	✗

Inserimento Metodo di Pagamento	✗	✓	✗	✗	✗
Modifica Metodo di Pagamento	✗	✓	✗	✗	✗
Visualizzazione Pagamenti Effettuati	✗	✓	✗	✗	✗
Richiesta Assistenza	✗	✓	✗	✗	✗
Gestione Amministrativa					
Gestione Utenti	✗	✗	✗	✓	✗
Visualizzazione Segnalazioni Community	✗	✗	✓	✗	✗
Risoluzione Segnalazione Community	✗	✗	✓	✗	✗
Visualizzazione Segnalazioni Pagamenti	✗	✗	✗	✓	✗
Risoluzione Segnalazione Pagamento	✗	✗	✗	✓	✗
Monitoraggio Attività	✗	✗	✗	✓	✗

3.5 Controllo del flusso globale del sistema

Il sistema *ZeroWaste Home* adotta un controllo del flusso **Event-driven** (guidato dagli eventi), in cui i sottosistemi forniscono gestori specifici per ogni evento. Quando un utente interagisce con l'interfaccia, un evento viene generato e catturato da un handler all'interno del sottosistema responsabile. Ogni sottosistema è quindi dotato del proprio gestore degli eventi, il quale indirizza il flusso di controllo alla logica applicativa necessaria per processare la richiesta.

L'integrazione con l'API **RxJS** di Angular supporta la gestione concorrente degli eventi in modo reattivo, garantendo un trattamento sicuro anche nelle situazioni di concorrenza. Qualora si presentino sezioni critiche all'interno di un sottosistema, vengono implementati i meccanismi necessari per garantire la mutua esclusione, evitando effetti collaterali dovuti all'accesso simultaneo alle risorse condivise. In questo modo, il sistema mantiene la coerenza e l'integrità dei dati, rispondendo in maniera sicura ed efficace agli input multipli degli utenti.

3.6 Condizioni Limite

Il sistema *ZeroWaste Home* è progettato per mantenere la continuità operativa anche in situazioni limite, come interruzioni impreviste o condizioni di sovraccarico. In caso di carichi elevati o errori critici, il sistema attiva misure di gestione delle eccezioni per preservare l'integrità dei dati e ridurre al minimo i disagi per l'utente. Queste misure includono il bilanciamento del carico, la ridondanza dei dati e meccanismi di ripristino automatico. In futuro, **l'amministratore si avvarrà di due gestori**: il **gestore della community** e **quello dell'utente**, per migliorare la gestione operativa e la risposta agli eventi critici.

3.6.1 Avvio del sistema

UCB_1			*Avvio del sistema*	**Data**	*16/11/2024*
				Versione	*1.0*
				Autore	*FR, MR, ATg, ATr*
Descrizione			*Lo UC consente l'avvio del sistema ZeroWaste Home, da parte dell'amministratore.*		
Attore Principale			**Amministratore.**		
Motivazione Attore			Un membro del team, responsabile delle funzioni amministrative, è incaricato di avviare il server del sistema *ZeroWaste Home*.		
Attori Secondari			**N/A.**		
Motivazione Attori Secondari			N/A.		
Entry Condition			L'Amministratore è connesso al server.		
Exit condition On success			Il sistema viene avviato con successo.		
Exit condition On failure			Il sistema non viene avviato.		
Flusso di Eventi Principale/Main Scenario					
1		**Attore:**	Esegue il comando per avviare il server.		
2		**Sistema:**	Verifica l'integrità dei dati persistenti.		
3		**Sistema:**	Avvia il server.		
Eccezioni/Flussi alternativi					
I Scenario/Flusso di eventi Alternativo *UCB_1.1*: Il server non era stato arrestato correttamente.					
Al passo 2:	**2.1**	**Sistema:**	Notifica all'Amministratore del precedente errore nell'arresto del server.		
	2.2	**Sistema:**	Avvia il server.		
II Scenario/Flusso di eventi Alternativo *UCB_1.2*: I dati persistenti sono danneggiati.					
Al passo 3:	**3.1**	**Sistema:**	Notifica all'Amministratore che sono presenti problemi relativi ai dati persistenti e non avvia il server.		
	3.2	**Attore:**	Corregge i dati persistenti e riesegue il comando di avvio server.		
	3.3	**Sistema:**	Ripete il punto 3.		

3.6.2 Spegnimento del sistema

UCB_2	*Spegnimento del sistema*	**Data**	*16/11/2024*
		Versione	*1.0*
		Autore	*FR, MR, ATg, ATr*
Descrizione	*Lo UC consente l'arresto controllato del sistema, rilasciando risorse e notificando eventuali sottosistemi.*		
Attore Principale	**Amministratore.**		
Motivazione Attore	Un membro del team, responsabile delle funzioni amministrative, è incaricato dell'arresto del server del sistema *ZeroWaste Home*.		
Attori Secondari	**N/A.**		
Motivazione Attori Secondari	N/A.		
Entry Condition	Il sistema è stato precedentemente avviato e l'Amministratore è connesso al server.		
Exit condition On success	Il sistema si arresta correttamente, salvando i dati e rilasciando risorse.		
Exit condition On failure	Il sistema notifica l'amministratore dei problemi riscontrati e non viene spento.		
Flusso di Eventi Principale/Main Scenario			
1	**Attore:**	Esegue il comando di spegnimento.	
2	**Sistema:**	Verifica che tutte le connessioni attive siano terminate.	
3	**Sistema:**	Rilascia le risorse e si arresta.	
Eccezioni/Flussi alternativi			
I Scenario/Flusso di eventi Alternativo *UCB_2.1*: Connessioni attive non chiuse.			
Al passo 2:	**2.1**	**Sistema:**	Notifica all'Amministratore e sospende l'arresto.
	2.2	**Attore:**	Chiude manualmente le connessioni e ripete il comando di spegnimento.

3.6.3 Fallimento del sistema

UCB_3	*Fallimento del sistema*	**Data**	*16/11/2024*
		Versione	*1.0*
		Autore	*FR, MR, ATg, ATr*
Descrizione	*Lo UC gestisce il comportamento del sistema in caso di errore fatale o di terminazione imprevista.*		
Attore Principale	**Amministratore**		
Motivazione Attore	Un membro del team, responsabile delle funzioni amministrative, è incaricato del comportamento del server del sistema *ZeroWaste Home, in caso di errore*.		
Attori Secondari	**N/A.**		
Motivazione Attori Secondari	N/A.		
Entry Condition	Il sistema subisce un'interruzione improvvisa.		
Exit condition On success	Il sistema si riavvia correttamente, recuperando i dati.		
Exit condition On failure	Il sistema non riesce a riprendersi e rimane spento.		
Flusso di Eventi Principale/Main Scenario			
1	**Attore:**	Riceve una notifica dell'errore del Sistema*.	
2	**Sistema:**	Include (*UCB_1: Avvio del Sistema*) – UC che si occupa della gestione dell'avvio del Sistema.	

* I casi di errore sono documentati nel RAD nei vari casi d'uso. Il sistema gestisce gli errori utente mostrando messaggi chiari che lo assistono nella risoluzione. In caso di interruzioni della rete, salva automaticamente lo stato corrente, consentendo di riprendere le operazioni senza perdita di dati al ritorno della connessione.

4 Servizi dei sottosistemi

La seguente sezione descrive i principali servizi forniti dai vari sottosistemi del sistema. Ogni sottosistema è stato progettato per soddisfare specifiche esigenze funzionali e di gestione, consentendo agli utenti finali e agli amministratori di interagire con le funzionalità offerte in modo semplice e intuitivo.

Sottosistema: **Registrazione**

Servizio	Descrizione	Interfaccia
Registrazione	Consente agli utenti non registrati di creare un nuovo account.	RegistrazioneService

Sottosistema: **Autenticazione**

Servizio	Descrizione	Interfaccia
Autenticazione	Consente agli utenti registrati di accedere al sistema tramite autenticazione.	AutenticazioneService

Sottosistema: **Gestione Profilo**

Servizio	Descrizione	Interfaccia
Visualizza Dati Personali	Consente agli utenti registrati di visualizzare i dati personali.	ProfiloService
Modifica Dati Personali	Consente agli utenti registrati di aggiornare le proprie informazioni personali.	ProfiloService
Elimina Account	Permette agli utenti registrati di eliminare il proprio account e i dati associati.	ProfiloService
Logout	Consente agli utenti autenticati di uscire dal sistema in modo sicuro.	ProfiloService
Recupero Password	Permette agli utenti di recuperare la password tramite e-mail.	ProfiloService

Sottosistema: **Gestione Prodotti**

Servizio	Descrizione	Interfaccia
Aggiungi Prodotto Frigo	Consente agli utenti di aggiungere un prodotto al frigo.	ProdottoService
Modifica Prodotto Frigo	Permette agli utenti di aggiornare le informazioni di un prodotto già registrato.	ProdottoService
Elimina Prodotto Frigo	Permette di rimuovere un prodotto dal frigo.	ProdottoService
Visualizza Prodotti Frigo	Mostra la lista di prodotti nel frigo dell'utente.	ProdottoService
Aggiungi Prodotto Dispensa	Consente agli utenti di aggiungere un prodotto alla dispensa.	ProdottoService
Modifica Prodotto Dispensa	Permette agli utenti di aggiornare le informazioni di un prodotto già registrato.	ProdottoService
Elimina Prodotto Dispensa	Permette di rimuovere un prodotto dalla dispensa.	ProdottoService
Visualizza Prodotti Dispensa	Mostra la lista di prodotti nella dispensa dell'utente.	ProdottoService
Ricerca Prodotti Per Nome	Mostra la lista di prodotti disponibili in base al nome.	ProdottoService

Sottosistema: **Gestione delle Notifiche**

Servizio	Descrizione	Interfaccia
Notifiche Scadenze	Invia notifiche agli utenti per i prodotti prossimi alla scadenza.	NotificheService
Notifiche Donazioni	Invia notifiche agli enti di beneficenza per le donazioni ricevute.	NotificheService
Visualizza Notifiche Ricevute	Visualizza le notifiche ricevute.	NotificheService

Sottosistema: **Gestione della Community**

Servizio	Descrizione	Interfaccia
Condivisione Ricette	Permette agli utenti di condividere ricette nella community.	RicettaService
Modifica Ricette	Permette agli utenti di modificare ricette condivise sulla community.	RicettaService
Segnalazione Contenuti	Consente di segnalare contenuti inappropriati o non conformi alle linee guida della community.	RicettaService
Visualizzazione Segnalazioni Effettuate	Consente agli utenti di visualizzare lo stato delle proprie segnalazioni.	RicettaService
Visualizzazione Ricette	Consente agli utenti di accedere e visualizzare ricette condivise dalla community.	RicettaService

Sottosistema: **Gestione Ricette e Piano Alimentare**

Servizio	Descrizione	Interfaccia
Creazione Ricetta	Consente all'utente di creare una nuova ricetta.	RicettaService
Creazione Piano Alimentare	Consente di creare un piano alimentare settimanale personalizzato.	PianoAlimentareService
Modifica Piano Alimentare	Consente di modificare il proprio piano alimentare.	PianoAlimentareService
Visualizzazione Piano Alimentare	Permette agli utenti di accedere al proprio piano alimentare settimanale.	PianoAlimentareService
Visualizzazione Suggerimenti Ricette	Visualizza suggerimenti di ricette da aggiungere al piano alimentare.	PianoAlimentareService

Sottosistema: **Gestione Lista della Spesa**

Servizio	Descrizione	Interfaccia
Generazione Lista della Spesa	Genera una lista della spesa basata sulle ricette e sul piano alimentare selezionato.	ListaSpesaService
Modifica Lista della Spesa	Permette agli utenti di modificare la lista della spesa.	ListaSpesaService
Visualizzazione Lista Spesa	Mostra la lista della spesa generata per il piano alimentare selezionato.	ListaSpesaService

Sottosistema: **Gestione Donazioni**

Servizio	Descrizione	Interfaccia
Visualizzazione Enti Disponibili	Mostra una lista di enti beneficiari registrati per le donazioni.	DonazioneService
Donazione Prodotti	Permette agli utenti di donare prodotti in scadenza agli enti benefici.	DonazioneService
Tracciamento Donazioni Effettuate	Consente agli utenti di visualizzare le donazioni effettuate.	DonazioneService
Tracciamento Donazioni Ricevute	Consente agli enti benefici di visualizzare i cibi donati disponibili.	DonazioneService

Sottosistema: **Gestione Impatto Ambientale**

Servizio	Descrizione	Interfaccia
Calcolo dell'Impatto Ambientale	Calcola l'impatto ambientale degli sprechi alimentari e delle donazioni effettuate.	ImpattoAmbientaleService
Visualizzazione Impatto Ambientale	Permette agli utenti di consultare le statistiche sul proprio impatto ambientale.	ImpattoAmbientaleService
Suggerimenti per Ridurre gli Sprechi	Suggerisce pratiche sostenibili basate sulle abitudini di consumo dell'utente.	ImpattoAmbientaleService

Sottosistema: **Gestione Abbonamenti e Pagamenti**

Servizio	Descrizione	Interfaccia
Gestione Piano Abbonamento	Permette agli utenti di selezionare ed abbonarsi ad un piano a pagamento.	PagamentoService
Inserimento Metodo di Pagamento	Consente agli utenti di inserire metodi di pagamento.	PagamentoService
Modifica Metodo di Pagamento	Consente agli utenti di modificare i propri metodi di pagamento.	PagamentoService
Visualizzazione Pagamenti Effettuati	Consente agli utenti di visualizzare i pagamenti effettuati.	PagamentoService
Richiesta Assistenza	Consente agli utenti di richiedere assistenza.	PagamentoService

Sottosistema: **Gestione Amministrativa**

Servizio	Descrizione	Interfaccia
Gestione Utenti	Consente ai gestori degli utenti di visualizzare, sospendere o eliminare utenti registrati.	AmministrazioneService
Visualizzazione Segnalazioni Community	Consente ai gestori della community di visualizzare le segnalazioni dei contenuti sulla community.	AmministrazioneService
Risoluzione Segnalazione Community	Consente ai gestori della community la risoluzione di una segnalazione relativa a un contenuto sulla community.	AmministrazioneService
Visualizzazione Segnalazioni Pagamenti	Consente ai gestori di visualizzare le segnalazioni relative a problemi di pagamento.	AmministrazioneService
Risoluzione Segnalazione Pagamento	Consente ai gestori degli utenti la risoluzione e risposta ad una segnalazione relativa ai pagamenti.	AmministrazioneService
Monitoraggio Attività	Fornisce strumenti per monitorare l'attività complessiva del sistema.	AmministrazioneService

5 Glossario

A.

Angular: Un framework open source basato su TypeScript, sviluppato per creare applicazioni web dinamiche e responsive.

B.

Back-End: La parte del sistema responsabile della logica applicativa e della gestione delle operazioni sul server. Per il progetto, il back-end è sviluppato utilizzando Spring.

Business Logic Layer: La parte di un sistema informatico che implementa le regole di business e gestisce l'elaborazione dei dati.

D.

Data Layer: Il livello di un sistema responsabile della gestione dei dati persistenti, come il salvataggio, la modifica e il recupero delle informazioni.

DBMS (Database Management System): Un sistema software progettato per creare, gestire e interrogare database. Nel contesto del progetto, si utilizza MySQL, un DBMS relazionale open source**.**

Design Goal: Obiettivi principali che guidano la progettazione di un sistema o di un software. Rappresentano i criteri fondamentali che il sistema deve soddisfare, come affidabilità, scalabilità, usabilità e manutenibilità, al fine di garantire che risponda efficacemente ai requisiti e alle aspettative degli utenti.

E.

Event-Driven: Un modello di programmazione in cui il flusso delle operazioni è determinato da eventi, come azioni dell'utente (clic, input), messaggi ricevuti o cambiamenti di stato.

F.

FAQ (Frequently Asked Questions): Sezione dedicata alle domande più comuni poste dagli utenti riguardo a un sistema, un servizio o un progetto. Le FAQ forniscono risposte rapide e chiare per risolvere dubbi frequenti, migliorando l'accessibilità e l'efficienza nella comunicazione tra sviluppatori e utenti.

Front-End: La parte del sistema che interagisce direttamente con l'utente finale. Nel progetto, il front-end è sviluppato con Angular, un framework per creare interfacce utente dinamiche e reattive.

H.

Handler: Un componente software progettato per gestire eventi specifici all'interno di un sistema **event-driven**. Gli **handler** processano l'evento, eseguendo azioni appropriate in risposta a esso, come notificare l'utente, aggiornare i dati o attivare altre funzionalità del sistema.

M.

MySQL: Un sistema di gestione di database relazionale (DBMS) noto per affidabilità, scalabilità e flessibilità, utilizzato per memorizzare e gestire i dati persistenti.

N.

Node.js: Un ambiente di runtime JavaScript che consente di eseguire codice lato server per creare applicazioni scalabili e ad alte prestazioni.

P.

Presentation Layer: Il livello di un sistema che gestisce la presentazione delle informazioni agli utenti finali e l'interazione con essi.

R.

Rank: Indicatore di priorità utilizzato per classificare gli obiettivi o le funzionalità di un sistema. Nel contesto del progetto, il **rank** determina l'ordine di importanza o di implementazione degli **Design Goals**, per assicurare che le caratteristiche più critiche siano affrontate per prime.

S.

Scadenza: Data in cui l'alimento non è più commestibile, si differenzia in "Da consumare entro" e "Da consumare preferibilmente entro".

Spring Framework: Una piattaforma open source per lo sviluppo di applicazioni Java che semplifica la creazione di sistemi robusti e scalabili, specialmente nel contesto delle applicazioni back-end.

T.

Three-Tier Architecture: Uno stile architetturale che separa un sistema in tre livelli distinti:

- Presentation Layer (Livello delle Interfacce): Gestisce l'interazione con l'utente;
- Business Logic Layer (Livello di Logica Applicativa): Contiene la logica del sistema e le regole di business;
- Data Layer (Livello dei Dati): Si occupa della gestione e dell'archiviazione dei dati.

Trade-Offs: Compromessi necessari tra due o più caratteristiche o requisiti di un sistema, spesso in conflitto tra loro.

TypeScript: Un linguaggio di programmazione basato su JavaScript che aggiunge tipizzazione statica opzionale per migliorare la manutenzione e l'affidabilità del codice.