Laurea Triennale in informatica-Università di Salerno
Corso di *Ingegneria del Software*-Prof. C. Gravino

System Design Document

SushiStar

Riferimento	SDD
Versione	2.0
Data	17/11/2024
Destinatario	Prof. C. Gravino
Presentato da	Antonio Avino, Antonio Bisogno, Matteo Coraggio, Paolo Balestriere

Revision History

Data	Versione	Descrizione	Autori
26/11/2024	0.1	Stesura scopo del sistema	MC
28/11/2024	0.2	Stesura Design Goal	AB
28/11/2024	0.2.1	Stesura Trade Off	AB
30/11/2024	0.3	Stesura architettura sistema corrente	MC
01/12/2024	0.3.1	Stesura decomposizione in sottosistemi	PB
01/12/2024	0.3.2	Stesura diagramma architetturale	PB
01/12/2024	0.4	Stesura Mapping Hardware Software	AA
02/12/2024	0.5	Stesura Deployment Diagram	AA
02/12/2024	0.6	Gestione dati persistenti	AA,AB,MC,PB
02/12/2024	1.0	Prima revisione del documento	AA,AB,MC,PB
14/12/2024	1.1	Stesura Dizionario Dei Dati	AA
14/12/2024	1.2	Stesura servizi dei sottosistemi	PB
14/12/2024	1.3	Stesura controllo degli accessi	AA
15/12/2024	1.4.1	Stesura Boundary Use Case 1	AA
15/12/2024	1.4.2	Stesura Boundary Use Case 2	AB
15/12/2024	1.4.3	Stesura Boundary Use Case 3	PB
15/12/2024	1.4.4	Stesura Boundary Use Case 4	MC
21/12/2024	1.5.1	Stesura Design Pattern Proxy	AA
21/12/2024	1.5.2	Stesura Design Pattern Façade	AA
29/12/2024	2.0	Seconda revisione del documento	AA,AB,MC,PB

Team Members

Nome	Acronimo	Contatto
Antonio Avino	AA	a.avino29@studenti.unisa.it
Antonio Bisogno	AB	a.bisogno53@studenti.unisa.it
Matteo Coraggio	MC	m.coraggio8@studenti.unisa.it
Paolo Balestriere	PB	p.balestriere@studenti.unisa.it

Sommario

Revision History ***2***

Team Members ***3***

1 Introduzione ***5***

1.1 Scopo del Sistema **5**

1.2 Obiettivi di Design (Design Goal) **5**

1.3 Definizioni, acronimi e abbreviazioni **8**

Definizioni **8**

Acronimi **8**

Abbreviazioni ..... Errore. Il segnalibro non è definito.

1.4 Riferimenti **8**

1.5 Organizzazione del documento **9**

2 Architettura del sistema corrente ***10***

2.1 Panoramica **10**

3 Architettura del sistema proposto ***11***

3.1 Panoramica della sezione **11**

3.2 Decomposizione in sottositemi **12**

3.3 Mapping hardware/software **18**

3.4 Gestione Dati Persistenti **19**

Introduzione ..... **Errore. Il segnalibro non è definito.**

CD_SDD Entity Class Diagram Ristrutturato ..... **Errore. Il segnalibro non è definito.**

Dizionario dei dati ..... **Errore. Il segnalibro non è definito.**

3.5 Controllo degli accessi e sicurezza **21**

3.6 Controllo globale del software **22**

3.7 Condizioni limite **22**

Avvio del sistema 23

3.7.1 Spegnimento del sistema 23

Fallimento del sistema 24

Fallimento nell'accesso ai dati persistenti 25

4 Servizi dei sottosistemi ..... *Errore. Il segnalibro non è definito.*

5 Glossario ***34***

1 Introduzione

1.1 Scopo del Sistema

Il sistema SushiStar intende principalmente facilitare la gestione degli ordini del ristorante fisico offrendo agli utenti un'interfaccia per poter acquistare i piatti desiderati con velocità e in tutta semplicità.
Il sistema è gestito da tre tipi di amministratori a cui è affidato un compito ben preciso:

- l'admin degli ordini si occupa dell'interazione utente-ordine preoccupandosi che tutti gli ordini effettuati vadano a buon fine;
- l'admin degli utenti si occupa della moderazione degli utenti assicurandosi che ogni utente rispetti le normative del sito;
- l'admin dei prodotti che si occupa del catalogo.

Il sistema, in base alle valutazioni degli utenti, stila una lista dei piatti preferiti e dei più acquistati in modo che i nuovi utenti siano in grado di comprendere quali sono le migliori prelibatezze del ristorante.
Agli utenti più fedeli sono riservate delle promozioni e degli sconti.

1.2 Obiettivi di Design (Design Goal)

Di seguito sono presentati i Design Goals che identificano le qualità su cui si focalizza il sistema permettendo, tramite la loro formalizzazione, di prendere decisioni di design con gli stessi criteri che qui vengono divisi in 4 gruppi:

1. **Performance Criteria:** requisiti imposti sul sistema in termini di spazio e velocità;
2. **Dependability Criteria:** sforzo necessario per minimizzare i crash di sistema in termini di robustezza, affidabilità, disponibilità e sicurezza;
3. **Maintenance Criteria:** sforzo necessario per modificare il sistema in termini di estensibilità, modificabilità, adattabilità, portabilità e leggibilità;
4. **End User Criteria:** criteri che un utente potrebbe desiderare in termini di utilità e usabilità.

Ciascun Design Goal è descritto da:

1. **ID Design Goal:** identificatore univoco accompagnato da un nome esplicativo;
2. **Categoria:** a quale criterio appartiene;
3. **Descrizione:** poche parole per far comprendere il Design Goal;
4. **Rank:** classificazione da 1 a 15 (1 massima, 15 minima);
5. **RNF di riferimento:** il requisito non funzionale da cui prende origine.

Design Goal

ID Design Goal	Categoria	Descrizione	Rank	RNF di Origine
DG_1 Tempi di risposta	Performance Criteria	Il sistema deve garantire un tempo di risposta non superiore a 3 secondi.	5	RNF_PE_1
DG_2 Navigazione concorrente	Performance Criteria	Il sistema deve sopportare il carico di 2000 utenti connessi contemporaneamente senza deterioramento di prestazioni.	6	RNF_PE_3
DG_3 Velocità di caricamento	Performance Criteria	Il sistema deve essere in grado di caricare le immagini in massimo 2 secondi.	11	RNF_PE_4
DG_4 Ordini concorrenti	Performance Criteria	Il sistema deve poter gestire fino a 2000 ordini effettuati contemporaneamente	7	RNF_PE_5
DG_5 Uptime	Dependability Criteria	Il sistema deve essere accessibile e garantire tutte le sue funzionalità a partire dalle 4:00 AM alle 3:00 AM.	3	RNF_PE_2
DG_6 Fallimento del sistema	Dependability Criteria	Il sistema deve notificare l'utente dell'impossibilità di completare un'operazione a causa di un errore o un fallimento del sistema.	10	RNF_RE_1
DG_7 Divisione dei ruoli utente	Dependability Criteria	Il sistema deve avere una precisa separazione delle operazioni che può compiere ogni utente in base al ruolo per mantenere l'integrità del sito.	4	RNF_RE_2
DG_8 Sicurezza passowrd	Dependability Criteria	Il sistema deve garantire la sicurezza delle password tramite crittografia SHA-512.	2	RNF_RE_3 RNF_RE_4

DG_9 Privacy	Dependability Criteria	Il sistema deve garantire le norme sulla privacy.	1	RNF_LE_1
DG_10 Normativa fiscale	Dependability Criteria	Il sistema deve mantenere in archivio i dati finanziari e contabili per un minimo di 7 anni.	8	RNF_LE_2
DG_11 Modificabilità	Maintenance Criteria	Il sistema deve offrire un'interfaccia per modificare il catalogo senza dover chiedere supporto tecnico.	14	RNF_SU_2
DG_12 Piattaforma web	Maintenance Criteria	Il sistema deve essere sviluppato per piattaforme web-based secondo gli standard di un modello architetturale adatto.	13	RNF_IM_1
DG_13 Colorblind	End-User Criteria	Il sistema deve essere accessibile anche a persone con daltonismo.	15	RNF_US_1
DG_14 Tempo di apprendimento	End-User Criteria	Il sistema deve offrire un'interfaccia che sia semplice e apprendibile in meno di 5 minuti.	9	RNF_US_2
DG_15 Applicazione web	End-User Criteria	Il sistema deve garantire l'accesso tramite pc o dispositivi mobili come pagina web.	12	RNF_PA_1

Trade-off

Trade-off	Descrizione
Sicurezza vs Tempi di risposta	Al fine di aumentare la sicurezza, si implementano funzioni per la protezione dei dati che potrebbero aumentare i tempi di risposta del sistema SushiStar.
Performance vs Usabilità	Per garantire tempi di caricamento e risposta rapidi e un'esperienza ottimale in caso di carico elevato, si evita l'uso di guide interattive e funzionalità con feedback visivi eccessivamente onerosi in termini di performance.

1.3 Definizioni, acronimi e abbreviazioni

Definizioni

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

Acronimi e abbreviazioni

- **CD**: Class Diagram
- **RF**: Requisiti Funzionali
- **DG**: Design goal
- **RNF**: Requisito non funzionale
- **PE**: Performance
- **RE**: Reliability
- **LE**: Legal
- **PA**: Packaging
- **IM**: Implementation
- **SU**: Supportability
- **GUI**: Grafic User Interface
- **DAO**: Data Access Object
- **UCBC**: Use Case Boundary Condition

1.4 Riferimenti

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

1.5 Organizzazione del documento

Questo documento è organizzato in modo da fornire una chiara panoramica del sistema SushiStar e delle sue caratteristiche principali, supportando il lettore nella comprensione dei dettagli tecnici e progettuali. Di seguito, una breve descrizione delle sezioni che lo compongono:

- **Introduzione:** Include lo scopo del sistema, gli obiettivi di design e i riferimenti utilizzati durante la stesura del documento.
- Architettura del Sistema Corrente: Descrive lo stato attuale delle soluzioni disponibili e i limiti che il sistema proposto intende superare.
- **Architettura del Sistema Proposto:** Presenta una panoramica dell'architettura scelta, con dettagli sulla decomposizione in sottosistemi, il mapping hardware/software e la gestione dei dati persistenti.
- **Servizi dei Sottosistemi:** Elenca e descrive le funzionalità fornite da ciascun sottosistema, evidenziando le interazioni tra di essi.
- **Cenni di ODD:** Fornisce una panoramica dei pattern di design adottati nel sistema per migliorarne modularità, scalabilità e manutenibilità.
- Glossario: Contiene le definizioni dei termini tecnici utilizzati, acronimi e abbreviazioni per facilitarne la comprensione.

2 Architettura del sistema corrente

2.1 Panoramica

Attualmente, non esiste alcun software che riunisca tutte le funzionalità necessarie per il ristorante SushiStar in un'unica piattaforma integrata. Sebbene esistano soluzioni software che gestiscono singoli aspetti come le ordinazioni online, le prenotazioni, il menu digitale o il pagamento online, nessuna di queste piattaforme combina tutte le funzionalità richieste dal ristorante, come la gestione in tempo reale delle scorte, la personalizzazione del menu e l'integrazione automatica delle ordinazioni con il sistema di gestione interno.

Alcune soluzioni simili si concentrano sulla digitalizzazione delle ordinazioni e sull'automazione della gestione del ristorante, ma nessuna di esse fornisce un sistema completamente integrato che includa anche la gestione delle scorte, l'aggiornamento dinamico del menu, il monitoraggio in tempo reale dello stato degli ordini, e la possibilità per il cliente di personalizzare i propri ordini in modo dettagliato.

SushiStar, quindi, ha bisogno di una piattaforma che possa gestire l'intero flusso di lavoro del ristorante, dalla ricezione degli ordini alla gestione delle risorse, senza affidarsi a più software separati o intermediari.

3 Architettura del sistema proposto

3.1 Panoramica della sezione

Il sistema progettato per **SushiStar** utilizza un'architettura **Three Tier**, che suddivide il sistema in tre livelli principali: presentazione, logica applicativa e gestione dei dati. Questa architettura è ideale per applicazioni web come SushiStar, poiché consente di organizzare il sistema in modo modulare e strutturato, garantendo flessibilità e facilità di manutenzione.

Per il livello di presentazione, verranno utilizzati **Angular HTML5** e **CSS3**.
Angular è un framework front-end basato su JavaScript che permette di sviluppare interfacce web dinamiche e interattive, offrendo un'esperienza utente fluida e moderna.
Grazie a queste tecnologie, gli utenti potranno navigare nel menù, ordinare prodotti e gestire le proprie informazioni in maniera intuitiva e reattiva.

Il livello di logica applicativa sarà sviluppato con il framework **Spring**, che offre una struttura solida e ben definita per gestire tutte le funzionalità principali del sistema, come la gestione degli ordini, la consultazione del menù, e l'elaborazione delle informazioni degli utenti.

Infine, il livello di gestione dei dati integrerà un **database** per memorizzare in modo sicuro e affidabile tutte le informazioni relative ai prodotti, agli utenti e agli ordini.

Grazie a questa separazione, il sistema sarà facilmente scalabile e aggiornabile nel tempo, assicurando un'implementazione ordinata e sostenibile.
Questa scelta architetturale permette a **SushiStar** di offrire un sistema performante, leggibile e pronto a soddisfare le esigenze di un'applicazione web moderna.

3.2 Decomposizione in sottosistemi

I sottosistemi individuati sono:

- **Registrazione:** si occupa di gestire la registrazione dei vari tipi di utente: cliente, gestore menù, gestore ordini e gestore utenti;
- **Autenticazione:** si occupa di gestire le funzionalità di Login, Logout, visualizzazione area utente e la modifica dei dati degli account;
- **Acquisto Prodotti:** responsabile delle operazioni riguardanti l'aggiunta dei prodotti al carrello e l'acquisto dei prodotti;
- **Menu:** responsabile delle operazioni di visualizzazione del menu e delle informazioni dei prodotti;
- **Recensioni Prodotti:** si occupa della gestione delle recensioni inserite dagli utenti;
- **Gestione Prodotti:** ha come scopo quello di organizzare, aggiornare e rendere disponibili le informazioni sui prodotti memorizzati nel sistema;
- **Gestione Info Utenti:** ha come scopo quello di organizzare, aggiornare e rendere disponibili le informazioni degli utenti iscritti alla piattaforma;
- **Gestione Ordini:** si occupa della visualizzazione e dell'aggiornamento degli ordini della piattaforma;
- **Accesso Dati:** responsabile del collegamento tra gli altri sottosistemi e il sottosistema che gestisce i dati persistenti dell'applicazione;
- **Persistent Data:** sottosistema contenente il software DBMS che gestisce la base di dati in cui sono memorizzati i dati persistenti dell'applicazione.

L'immagine sottostante descrive le dipendenze tra i vari sottosistemi descritte in uno schema UML:

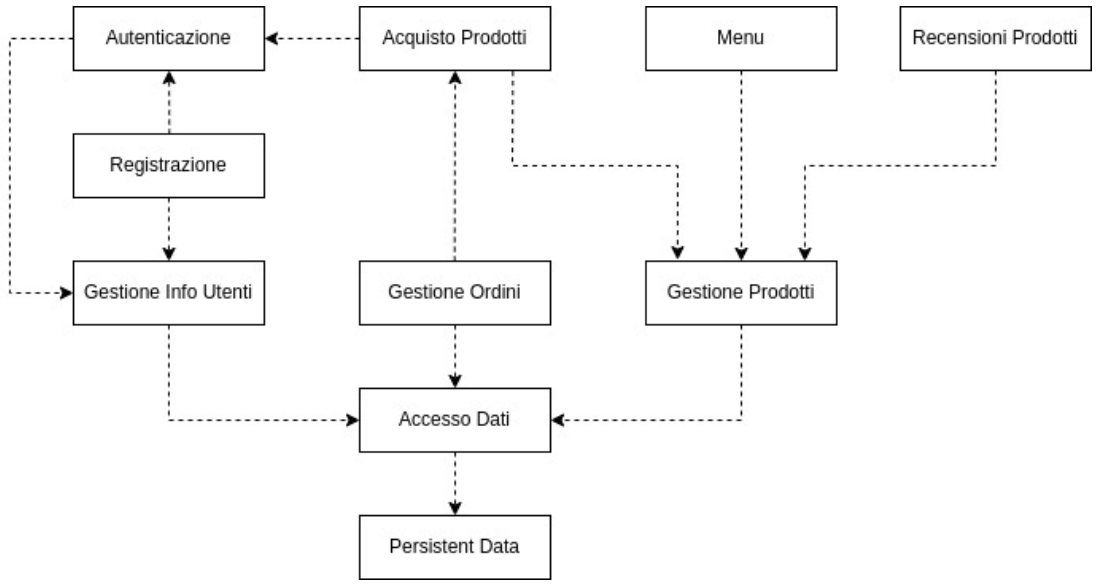
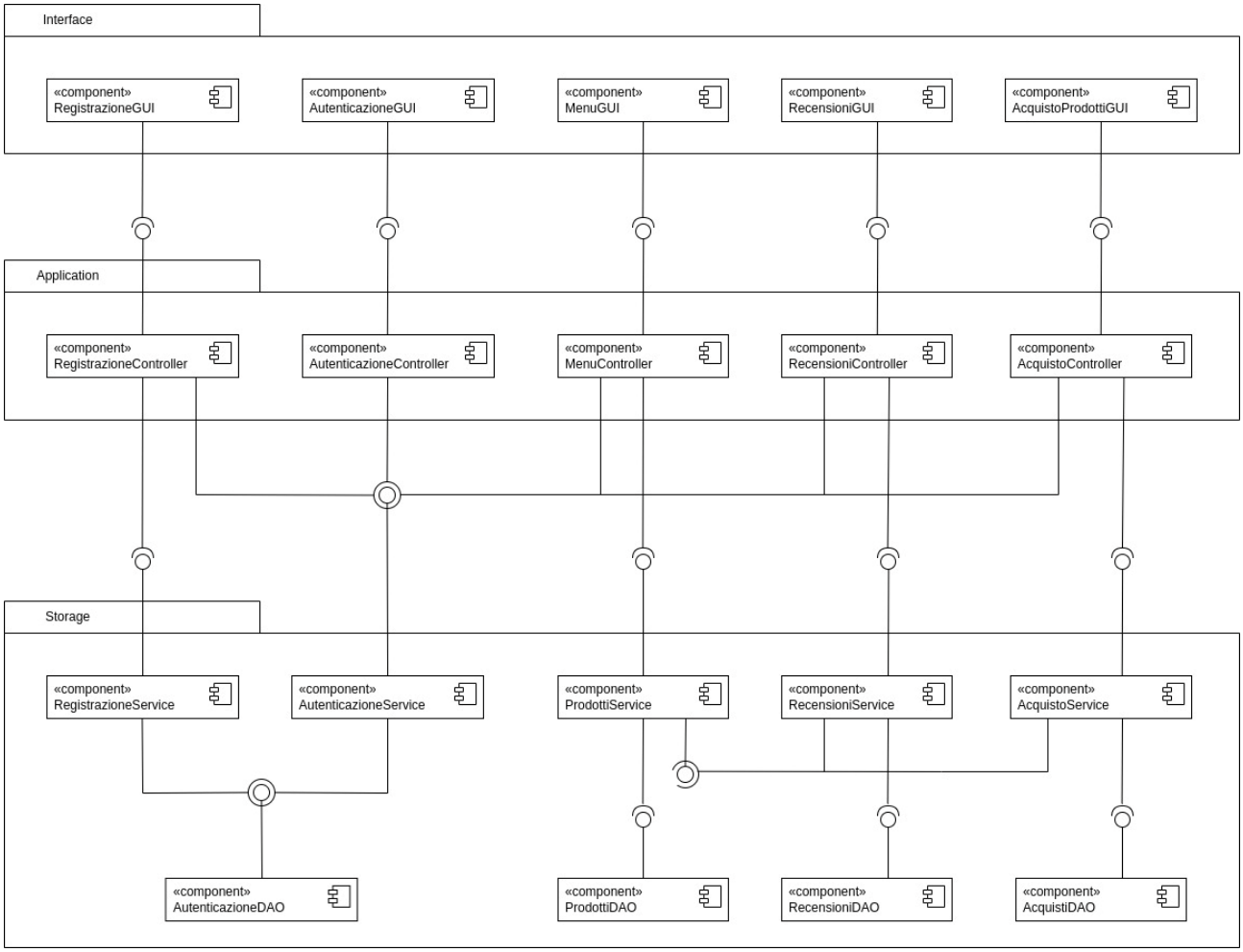


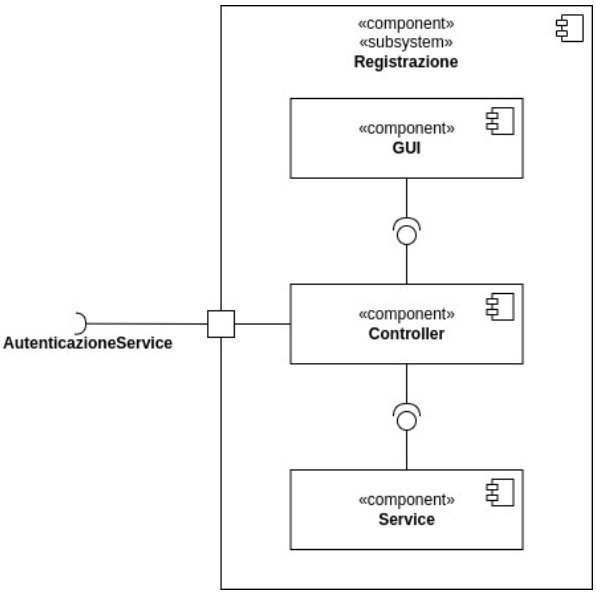
Diagramma Architetturale

In questo schema vengono rappresentati tutti i sottosistemi in modo dettagliato, evidenziando per ognuno le seguenti componenti:

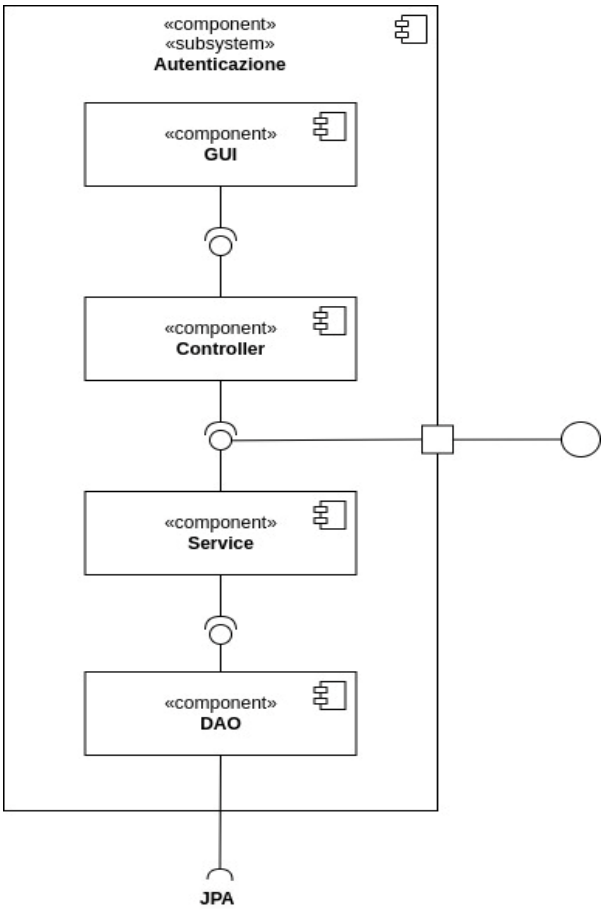
- **GUI:** Graphic User Interface, ovvero le varie View che saranno mostrate all'utente;
- **Controller:** logica di controllo per il sistema, intermediario tra le View e i Service;
- **Service:** sezione che si occupa di eseguire la logica di business;
- **DAO:** Data Access Object, responsabile dell'accesso ai dati persistenti.



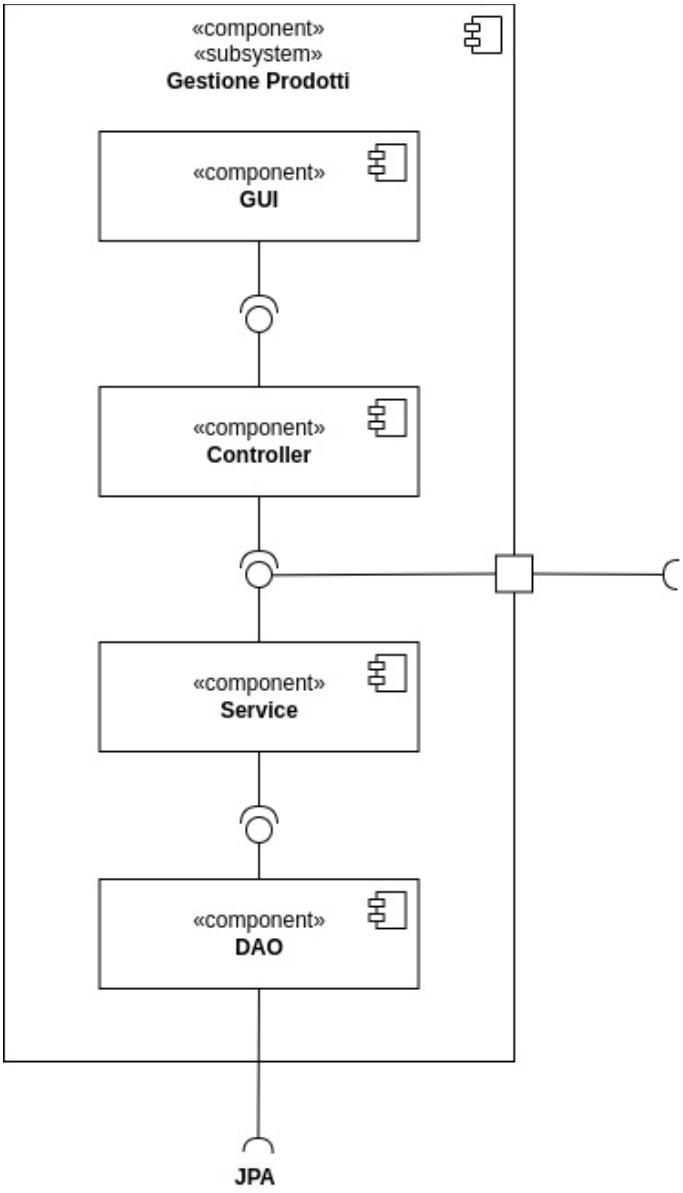
Sottosistema Registrazione



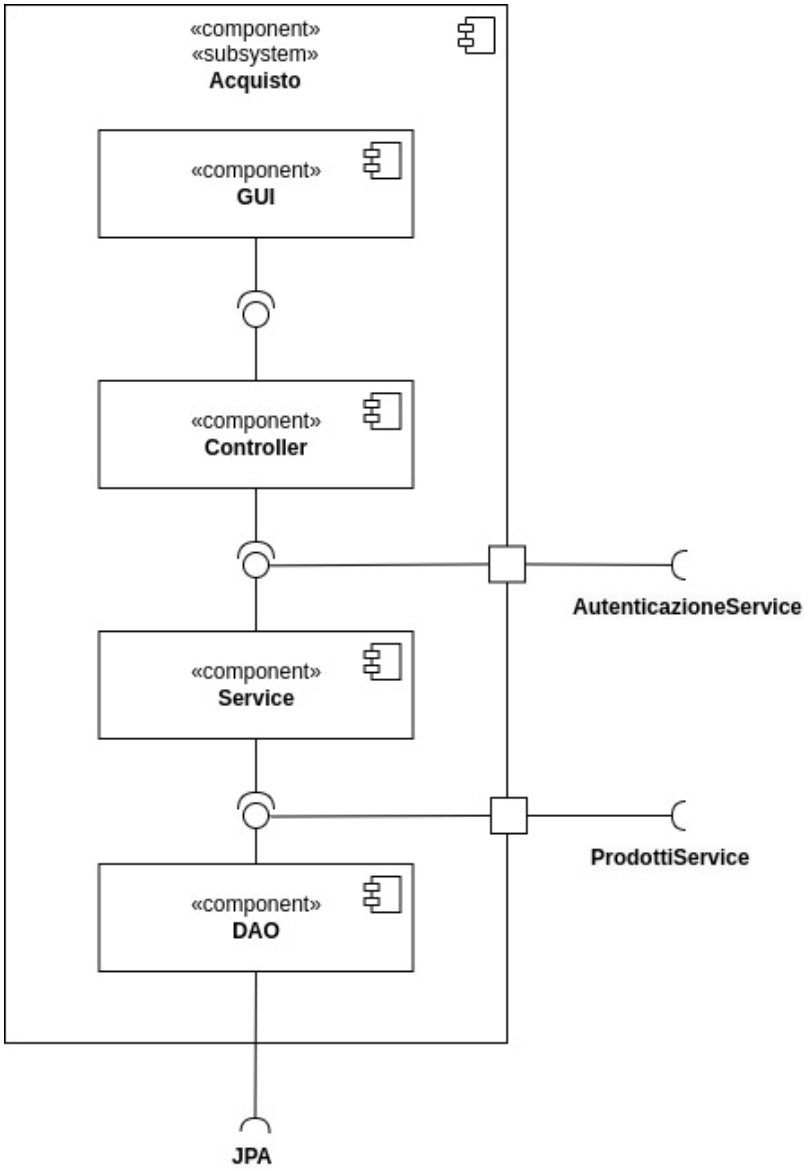
Sottosistema Autenticazione



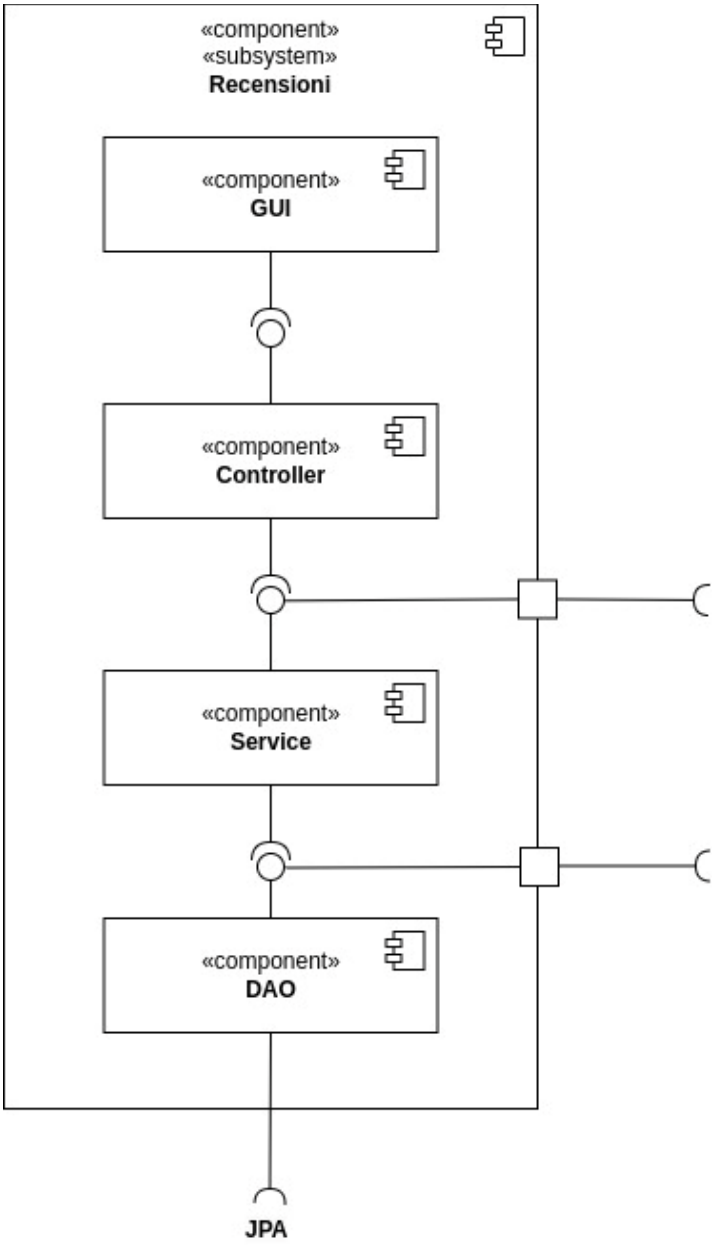
Sottosistema Gestione Prodotti



Sottosistema Acquisto Prodotti



Sottosistema Recensioni



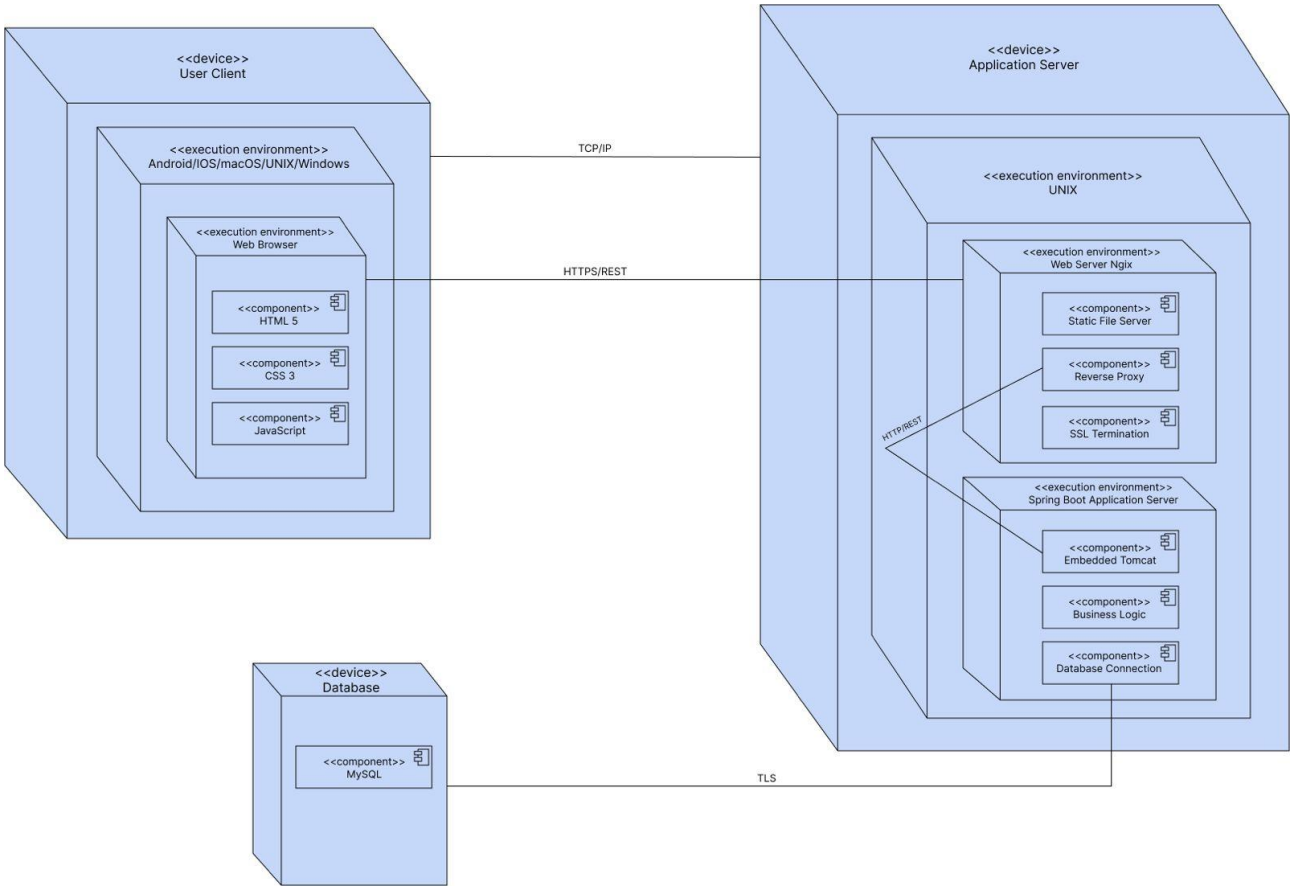
3.3 Mapping hardware/software

Il sistema **SushiStar** è progettato come un'applicazione web accessibile online, consentendo agli utenti di navigare nel menù, effettuare ordini per il ritiro o la consegna e accedere ad altre funzionalità da qualsiasi dispositivo dotato di browser.
Gli utenti interagiranno con il sistema tramite un'interfaccia web intuitiva e reattiva, progettata per offrire una navigazione fluida.

Il Front-End dell'applicazione è realizzato in **Angular**, un framework che consente di creare interfacce utente moderne e dinamiche.
I file statici dell'interfaccia vengono distribuiti tramite un web server **Nginx**, scelto per la sua affidabilità e velocità nella gestione delle richieste. La comunicazione avviene tramite protocollo **HTTPS** garantendo che i dati degli utenti siano sempre protetti.

Il Back-End, costruito utilizzando Spring Boot, gestisce la logica applicativa e fornisce le **API REST** necessarie per il dialogo tra il Front-End e il sistema centrale. Questo approccio modulare consente una separazione netta tra la presentazione e la logica di business, rendendo l'architettura scalabile e facile da aggiornare. Il Back-End comunica direttamente con un database **MySQL**, che gestisce i dati persistenti come informazioni sugli utenti, sugli ordini e sul menù. MySQL è stato scelto per la sua capacità di mantenere integrità e coerenza anche in presenza di carichi elevati, garantendo al contempo velocità ed efficienza.

3.4 Deployment Diagram



3.5 Gestione Dati Persistenti

La gestione dei dati persistenti del sistema Sushistar viene operata con una base di dati relazionale gestita dal software di gestione MySQL. Questo software è stato scelto per la sua affidilità e per il supporto offerto per l'integrazione con il linguaggio Java e il software JPA (Java Persistence API).

STRUTTURA DEL DATABASE

Le seguenti tabelle rappresentano gli oggetti più importanti per l'applicativo:

- Utente: contiene i dati relativi agli utenti registrati nel sistema

- Prodotto: rappresenta gli articoli messi in vendita dal ristorante e ne memorizza nome, prezzo, descrizione e quantità
- Carrello: registra informazioni sui carrelli degli utenti
- Preferiti: associa ad ogni utente i suoi prodotti preferiti

CONNESSIONE AL DATABASE

L'applicazione è connessa alla base di dati e compie operazioni sui dati conservati in essa utilizzando il modulo JPA (Java Persistence API) e Spring Data JPA per semplificare le operazioni CRUD grazie all'organizzazione in repository.

3.6 Controllo degli accessi e sicurezza

Attori / Oggetti	Utente	Gestore Menù	Gestore Utenti	Gestore Ordini
Autenticazione	Login Logout	Login Logout	Login Logout	Login Logout
Registrazione	Registrazione			
Gestione info utenti	VisualizzaDatiPersonali	VisualizzaDatiUtenti CancellaUtenti		
Acquisto prodotti	AggiuntaProdottoCarrello CompletaPagamento			
Gestione ordini	VisualizzaStatoOrdine AnnullaOrdine VisualizzaStoricoOrdini			CambiaStatoOrdine VisualizzaStoricoOrdiniUtenti
Menù	VisualizzaMenù			
Gestione prodotti		AggiuntaProdotto ModificaProdotto RimuoviProdotto		

3.7 Controllo globale del software

Il sistema SushiStar utilizza un approccio basato sugli eventi (Event-driven), in cui i sottosistemi dispongono di gestori specifici per ogni evento. Quando un utente interagisce con l'interfaccia, viene generato un evento che viene catturato da un handler nel sottosistema pertinente. Ogni sottosistema ha il proprio gestore degli eventi, che indirizza il flusso di controllo verso la logica applicativa necessaria per elaborare la richiesta. In questo modo, il sistema è in grado di rispondere in modo efficace e ordinato agli input degli utenti, mantenendo una gestione fluida e precisa del flusso di lavoro.

3.8 Condizioni limite (Boundary Conditions)

3.8.1 Avvio del sistema

Identificativo UCBC_1	Avvio del sistema	Data	24/11/2024
		Vers.	1.0
		Autore	Avino Antonio
Descrizione	Lo UC fornisce una descrizione dell' avvio del sistema SushiStar da parte dell' admin		
Attore Principale	Admin		
Attori secondari	NA		
Entry Condition	L' admin è connesso al server		
Exit condition On success	Il sistema si avvia correttamente		
Exit condition On failure	Il sistema non viene avviato		
Flusso di Eventi Principale/Main Scenario			
1	Admin:	Esegue il comando per avviare il server	
2	Sistema:	Verifica l' integrità dei dati persistenti	
3	Sistema:	Avvia il server	
Scenario/Flusso di eventi Alternativo: Il server non era stato arrestato correttamente			
3.a1	Sistema:	Notifica all' admin del precedente errore nell' arresto del server	
3.a2	Sistema:	Avvia il server	
Scenario/Flusso di eventi Alternativo: I dati persistenti sono danneggiati			
3.b1	Sistema:	Notifica all' admin che sono presenti problemi relativi ai dati persistenti e non avvia il server	
3.b2	Admin:	Corregge i dati persistenti e riesegue il comando di avvio server	

3.8.2 Spegnimento del sistema

Identificativo UCBC_2	Spegnimento del sistema	Data	27/12/2024
		Vers.	1.0

		Autore	Bisogno Antonio
Descrizione	Lo UC fornisce una descrizione dello spegnimento del sistema SushiStar da parte dell'Admin		
Attore Principale	Admin		
Attori secondari	NA		
Entry Condition	Il sistema è attivo.		
Exit condition On success	Il sistema si spegne correttamente.		
Exit condition On failure	Il sistema non si spegne.		
Flusso di Eventi Principale/Main Scenario			
1 Admin:	Esegue il comando per spegnere il sistema.		
2 Sistema:	Controlla che non ci siano connessioni attive.		
3 Sistema:	Se non ci sono connessioni attive, procede allo spegnimento.		
Scenario/Flusso di eventi Alternativo: Ci sono ancora connessioni attive			
3.a1 Sistema:	Notifica che ci sono ancora connessioni attive.		
3.a2 Sistema	Attende che si chiudano tutte le connessioni già attive impedendo la creazione di nuove connessioni.		
3.a3 Sistema	Controlla che non ci siano altre connessioni attive.		
3.a4 Sistema	Se non ci sono altre connessioni attive, procede allo spegnimento.		

3.8.3 Fallimento del sistema

Identificativo UCBC_3	Fallimento del sistema	Data	30/11/2024
		Vers.	1.0
		Autore	Coraggio Matteo

Descrizione		Questo UC definisce il comportamento del sistema in caso di fallimento.
Attore Principale		Admin
Attori secondari		NA
Entry Condition		Il sistema presenta un fallimento inaspettato.
Exit condition On success		Il sistema si riavvia correttamente.
Exit condition On failure		Il sistema non si riavvia.
Flusso di Eventi Principale/Main Scenario		
1	Sistema:	Si arresta a causa di un imprevisto.
2	Admin:	Risolve la causa del fallimento
3	Admin	*riferimento UCBC_1*

3.8.4 Fallimento nell'accesso ai dati persistenti

Identificativo UCBC_4	Errore di accesso ai dati persistenti	Data	16/11/2024
		Vers.	1.0
		Autore	Paolo Balestriere

Descrizione		Lo UC definisce il comportamento del sistema in caso di fallimento nell'accesso ai dati persistenti del sistema
Attore Principale		Admin
Attori secondari		NA
Entry Condition		Il sistema non può accedere ai dati persistenti
Exit condition On success		Il sistema riprende il normale funzionamento
Exit condition On failure		Il sistema non riprende il normale funzionamento
Flusso di Eventi Principale/Main Scenario		
1	Sistema:	Notifica l'amministratore dell'impossibilità di accedere ai dati persistenti in seguito ad una richiesta da parte degli utenti
2	Sistema:	Mostra agli utenti un messaggio di errore quando questi eseguono operazioni che richiedono l'accesso ai dati persistenti
3	Admin:	*riferimento UCBC_2*
4	Admin:	Ripristina l'accessibilità o la sanità dei dati persistenti
5	Admin:	*riferimento UCBC_1*

4 Servizi dei sottosistemi

4.1 Sottosistema Autenticazione

Servizio	Descrizione	Interfaccia
Login	Permette agli utenti di effettuare l'accesso alla piattaforma SushiStar	AutenticazioneService
Logout	Permette ad un utente autenticato di disconnettersi dalla piattaforma	AutenticazioneService

4.2 Sottosistema Registrazione

Servizio	Descrizione	Interfaccia
Sign In	Permette agli utenti di creare un account e registrarsi alla piattaforma SushiStar	RegistrazioneService
Creazione Utente Impiegato	Permette agli utenti Amministratore di creare account per i dipendenti del ristorante SushiStar	RegistrazioneService

4.3 Sottosistema Gestione prodotti

Servizio	Descrizione	Interfaccia
Menu Prodotti	Fornisce la lista completa dei prodotti registrati nel sistema	MenuService
Dettaglio Prodotto	Fornisce le informazioni dettagliate di un singolo prodotto presente nel sistema	MenuService

Aggiungi Prodotto	Permette a determinati utenti di aggiungere prodotti al menu	MenuService
Rimuovi Prodotto	Permette a determinati utenti di rimuovere prodotti dal menu	MenuService
Modifica Prodotto	Permette a determinati utenti di modificare prodotti presenti nel menu	MenuService

4.4 Sottosistema Recensioni

Servizio	Descrizione	Interfaccia
Aggiungi Recensione	Permette agli utenti di inserire una recensione per ogni prodotto presente nel sistema	RecensioniService
Visualizza Recensione	Permette agli utenti di visualizzare le recensioni presenti nel sistema	RecensioniService

4.5 Sottosistema Acquisto Prodotti

Servizio	Descrizione	Interfaccia
Aggiungi Prodotto al Carrello	Permette agli utenti di aggiungere un prodotto al carrello	AcquistoProdottiService
Rimuovi Prodotto dal Carrello	Permette agli utenti di rimuovere un prodotto dal carrello	AcquistoProdottiService
Effettua Acquisto	Permette agli utenti di creare un ordine con i prodotti presenti nel carrello	AcquistoProdottiService

4.6 Sottosistema Gestione Ordini

Servizio	Descrizione	Interfaccia
Crea Nuovo Ordine	Crea un nuovo ordine con le informazioni di consegna dell'utente e i prodotti che questi intende acquistare	GestioneOrdiniService
Aggiornamento Stato Ordine	Permette agli utenti dipendenti di SushiStar di aggiornare lo stato di avanzamento di un ordine	GestioneOrdiniService

Info Ordine	Fornisce le informazioni di un ordine registrato nel sistema	GestioneOrdiniService

5 Cenni di ODD

5.1 Design Patterns

5.1.1 Proxy

Il Proxy è un design pattern strutturale che fornisce un surrogato o un rappresentante per un altro oggetto, controllandone l'accesso o aggiungendo funzionalità. Viene utilizzato principalmente per ottimizzare le prestazioni o aggiungere un livello di controllo nell'interazione con oggetti costosi o complessi.

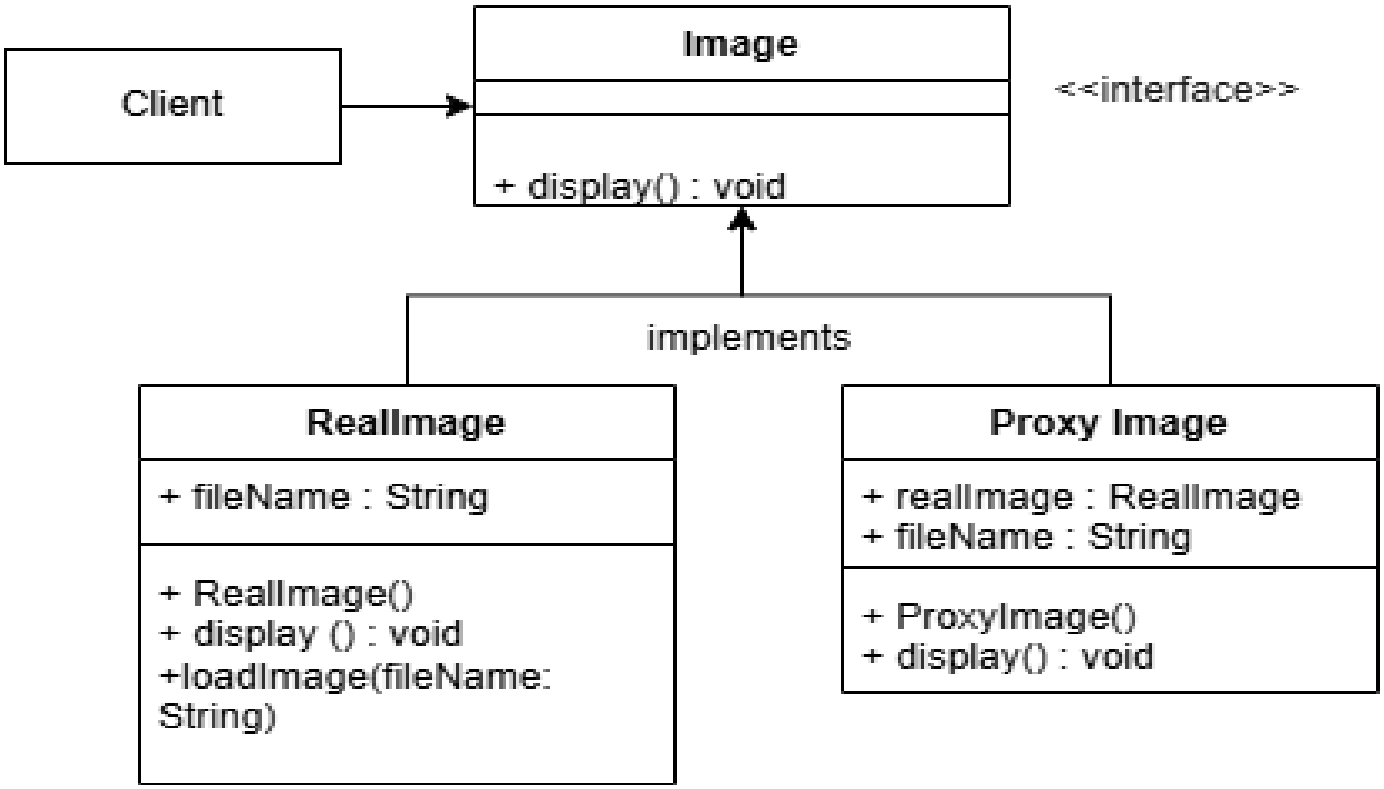
Obiettivi:

- **Ottimizzare le prestazioni:** Gestisce il caricamento di risorse pesanti, come le immagini, caricandole solo quando necessario (lazy loading).

- **Migliorare l'esperienza utente:** Fornisce un placeholder o un contenuto alternativo mentre le risorse principali vengono caricate in background.
- **Ridurre i tempi di attesa:** Specialmente utile in condizioni di connessione instabile, mostrando prima i contenuti principali (ad esempio, testo) e caricando le immagini successivamente.

Implementazione nel progetto:

- **Gestione del caricamento delle immagini:** Il Proxy agirà come sostituto per le immagini dei prodotti, visualizzando un placeholder iniziale (es. un'icona generica o un'animazione di caricamento) e caricando le immagini effettive solo quando necessario, ad esempio, quando la connessione è sufficiente o l'elemento è visibile nella viewport.
- **Migliorare l'esperienza utente:** Durante il caricamento delle immagini, il testo (nome, descrizione e prezzo del prodotto) verrà visualizzato immediatamente, evitando ritardi e rendendo l'interfaccia più fluida anche in caso di connessione instabile.
- **Ottimizzazione delle risorse:** Le immagini verranno caricate in modo progressivo (lazy loading), risparmiando larghezza di banda e riducendo il tempo di caricamento iniziale della pagina.
- **Sicurezza e controllo:** Il Proxy potrà intercettare richieste per immagini non disponibili o fallite e gestire eventuali errori, mostrando un'immagine alternativa o un messaggio di avviso.


Client
Image
<<interface>>
+ display() : void
implements
RealImage
+ fileName : String
+ RealImage()
+ display () : void
+loadImage(fileName: String)
Proxy Image
+ realImage : RealImage
+ fileName : String
+ ProxyImage()
+ display() : void

5.1.2 Façade

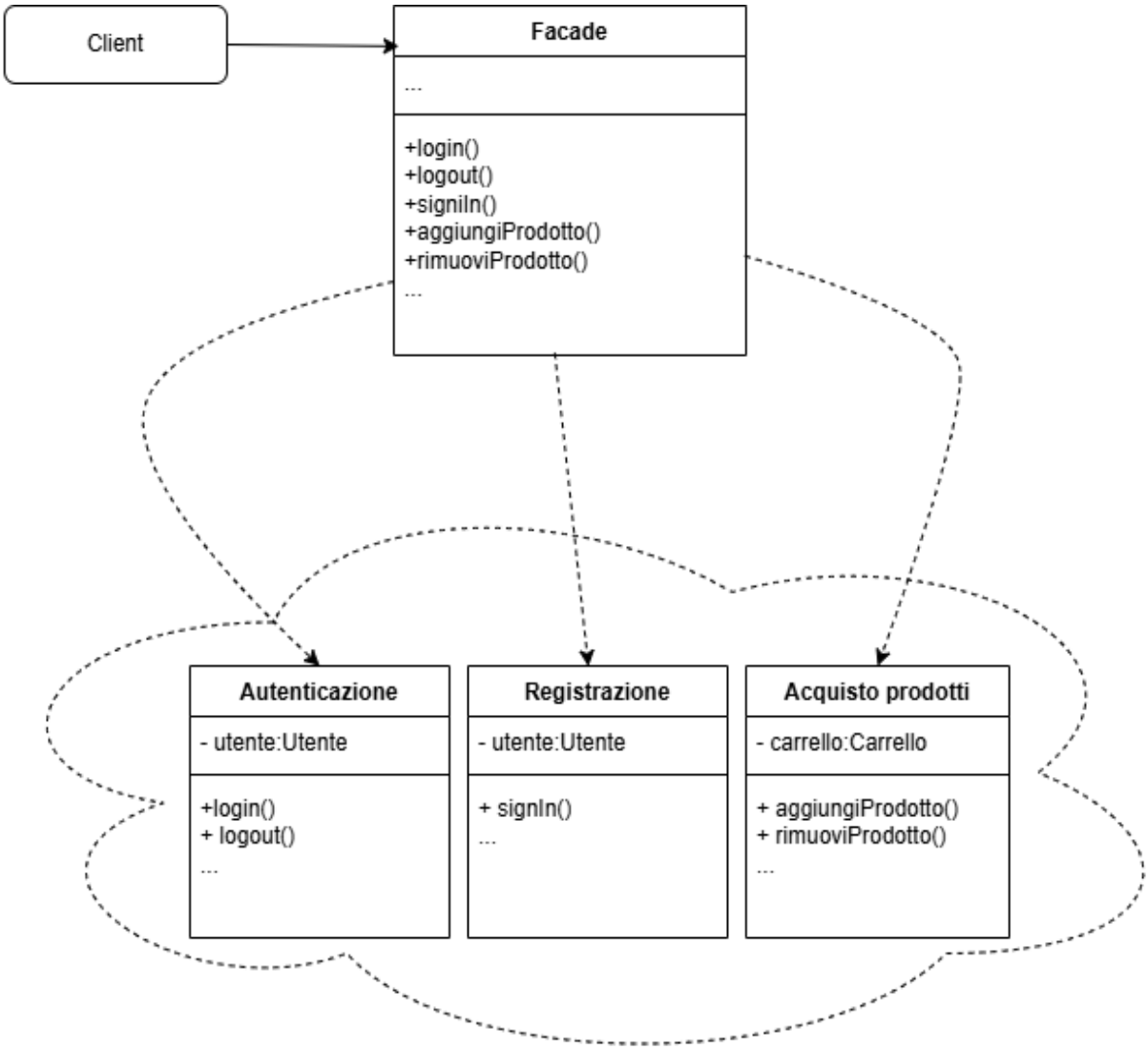
Il Façade è un design pattern strutturale che fornisce un interfaccia semplificata a un sistema complesso o a un insieme di sottosistemi. Il suo scopo principale è ridurre la dipendenza e la complessità tra il client (chi utilizza il sistema) e le componenti interne del sistema stesso.

Obiettivi:

- **Riduzione della complessità del sistema:** Nasconde ai client (es. GUI) le interazioni dettagliate con i sottosistemi, presentando un'interfaccia unica e intuitiva. Questo semplifica la comunicazione e consente di gestire le operazioni attraverso un singolo punto di accesso;
- **Disaccoppiamento tra i livelli:** I client non hanno bisogno di conoscere la struttura interna del sistema né di interagire direttamente con i controller, i servizi o i DAO. Ciò riduce l'accoppiamento e favorisce la modularità, rendendo il sistema più robusto ai cambiamenti;
- **Miglioramento della manutenibilità e della scalabilità:** Isolando i client dai dettagli implementativi dei sottosistemi, il Facade consente di apportare modifiche interne al sistema senza impattare le interfacce esterne. Questo semplifica la manutenzione e agevola l'introduzione di nuove funzionalità;
- **Orchestrazione delle operazioni complesse:** Il Facade può combinare e coordinare chiamate multiple ai sottosistemi per eseguire operazioni più articolate in modo trasparente per il client. Questo elimina la necessità per i client di comprendere o orchestrare manualmente le dipendenze tra i componenti.

Implementazione nel progetto:

- **Semplificare la comunicazione tra livelli:** I componenti GUI non interagiranno direttamente con i controller o i servizi, ma utilizzeranno il Facade per accedere alle funzionalità del sistema.
- **Nascondere la complessità del backend**: Fornirà un'interfaccia uniforme per le operazioni di registrazione, autenticazione, gestione del menù, recensioni e acquisto prodotti.
- **Migliorare la flessibilità:** Se un controller, un servizio o un DAO cambia, l'interfaccia del Facade rimane stabile, proteggendo i client da modifiche interne.


Client
Facade
...
+login()
+logout()
+signiIn()
+aggiungiProdotto()
+rimuoviProdotto()
...
Autenticazione
- utente:Utente
+login()
+ logout()
...
Registrazione
- utente:Utente
+ signIn()
...
Acquisto prodotti
- carrello:Carrello
+ aggiungiProdotto()
+ rimuoviProdotto()
...

6 Glossario

A.

Angular: Framework front-end basato su JavaScript utilizzato per sviluppare interfacce web e interattive.

APIREST: Interfaccia di programmazione che consente la comunicazione tra sistemi software attraverso operazioni HTTP (GET, POST, PUT, DELETE).

B.

Back-End: Parte del sistema responsabile della logica applicativa, elaborazione dei dati e interazione con il database.

C.

Carrello: Spazio virtuale in cui gli utenti raccolgono i prodotti selezionati prima di procedere con l'ordine.

D.

Database: Sistema di gestione per memorizzare e organizzare dati in modo strutturato, garantendo persistenza e integrità. Nel contesto di SushiStar, il database utilizzato è MySQL.

Design Patterns: Una soluzione riutilizzabile e collaudata per risolvere problemi ricorrenti di progettazione software in un contesto specifico.

E.

F.

Front-End: Parte di un sistema software che interagisce direttamente con l'utente finale, generalmente attraverso un browser web.

G.

Gestore menù: Attore responsabile dell'aggiornamento dei prodotti nel menù, inclusa l'aggiunta, modifica o rimozione di piatti.

Gestore ordini: Attore che si occupa di monitorare e aggiornare lo stato degli ordini, gestendo eventuali richieste di modifica o annullamento.

Gestore utenti: Attore che gestisce le informazioni sugli utenti, come registrazione, accessi, e risoluzione di eventuali problematiche relative agli account.

H.

HTTPS: Protocollo per la trasmissione sicura dei dati sul web, utilizzando crittografia per proteggere le informazioni scambiate tra client e server.

I.

L.

M.

MySQL: Sistema di gestione di database relazionale scelto per memorizzare e gestire i dati nel sistema SushiStar.

N.

Nginx: Web server che funge anche da reverse proxy e load balancer, utilizzato per distribuire i file statici del Front-End e inoltrare le richieste al Back-End.

O.

Ordine: Rappresenta l'insieme dei prodotti acquistati dall'utente, con dettagli come quantità, metodo di pagamento e stato dell'ordine (es. in preparazione, consegnato).

P.

Prodotto: Elemento del menù che gli utenti possono selezionare e ordinare. Ogni prodotto ha attributi come nome, descrizione, prezzo e disponibilità.

Q.

S.

Scalabilità: Capacità del sistema di gestire un aumento del carico di lavoro mantenendo le prestazioni.

T.

Three Tier: Stile architetturale che suddivide un sistema software in tre livelli distinti: livello di presentazione (interfaccia utente), livello di logica applicativa (gestione delle funzionalità) e livello di gestione dei dati (persistenza e accesso ai dati).

U.

Utente: Attore che interagisce con il sistema per visualizzare il menù, effettuare ordini e accedere ad altre funzionalità.

V.

X.

Y.

Z.