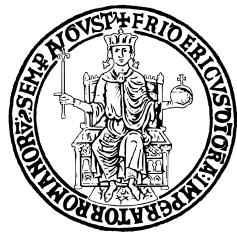


UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE
DELL'INFORMAZIONE



CORSO DI LAUREA IN INFORMATICA

INSEGNAMENTO DI INGEGNERIA DEL SOFTWARE
ANNO ACCADEMICO 2022/2023

Ratatouille23

Autori

Gian Marco ADDATI
N86003795
gi.addati@studenti.unina.it

Simone GIORDANO
N86003660
simone.giordano5@studenti.unina.it

Docenti

Sergio Di MARTINO
Francesco COTUGNO
Luigi Libero Lucio STARACE

26 febbraio 2023

Indice

I Introduzione	6
II Requisiti Software	8
1 Requisiti	8
1.1 Funzionali	8
1.2 Non Funzionali	10
2 Analisi dei Requisiti	11
2.1 Modellazione dei Casi d'Uso	11
2.1.1 Punto 1 - Creazione di Utenze per i Dipendenti	12
2.1.2 Punto 3 - Personalizzazione del Menù	13
2.1.3 Punto 4 - Stampa del QR Code	14
2.1.4 Punto 5 - Specifica degli Elementi in una Seconda Lingua	15
2.1.5 Punto 6 - Registrazione di Ordinazioni	16
2.1.6 Punto 14 - Visualizzazione di Statistiche	17
2.2 Descrizioni Testuali Strutturate	18
2.2.1 Punto 3 - Personalizzazione del Menù	19
2.2.2 Punto 6 - Registrazione di Ordinazioni	20
3 Mockup	21
3.1 Primo Prototipo	21
3.2 Secondo Prototipo (Definitivo)	21
3.3 Schermate	22
3.3.1 Autenticazione	22
3.3.2 Cambio obbligatorio della password al primo accesso	23
3.3.3 Visualizzazione Informazioni e Statistiche del Ristorante	24
3.3.4 Gestione del Menù	25
3.3.5 Gestione Utenze	28
3.3.6 Gestione Tavoli	29
4 Individuazione Target di Utenti	30
4.1 Statistiche	30
4.2 Personas	33
4.2.1 Criteri per l'identificazione delle Personas	33
4.2.2 Personas Identificate	33

5 Usabilità a Priori	36
5.1 Considerazioni sull'Usabilità	36
5.1.1 Apprendibilità	38
5.1.2 Efficienza	39
5.1.3 Memorabilità	40
5.1.4 Errori	41
5.1.5 Soddisfazione	42
5.2 Valutazione Usabilità a Priori	43
5.2.1 Esaminati	44
5.2.2 Task stabilité	46
5.2.3 Sistema di Valutazione	46
5.2.4 Voti Parziali	46
5.2.5 Risultati dei Test	48
5.2.6 Commenti degli Esaminati	48
5.2.7 Conclusioni	49
6 Glossario	50
7 Specifica dei Requisiti	53
7.1 Class Diagram di Analisi	54
7.1.1 Class Diagram delle Entità	54
7.1.2 Punto 1 - Creazione Utenze per Dipendenti	55
7.1.3 Punto 3 - Personalizzazione del Menù	56
7.1.4 Punto 5 - Specifica degli Elementi in una Seconda Lingua	57
7.1.5 Punto 6 - Registrazione di Ordinazioni	58
7.1.6 Punto 4 - Stampa del QR Code	59
7.1.7 Punto 14 - Statistiche	59
7.2 Sequence Diagram di Analisi	60
7.2.1 Punto 3.1 - Personalizzazione del Menù - Categorie	60
7.2.2 Punto 3.2 - Personalizzazione del Menù - Elementi	60
7.2.3 Punto 6 - Registrazione di Ordinazioni	61
7.3 Statecharts Funzionali	62
7.3.1 Punto 3 - Personalizzazione del Menù	62
7.3.2 Punto 6 - Registrazione di Ordinazioni	62
III Design di Sistema	63

8 Analisi di Architettura e Sistemi di Design	63
8.1 Client	64
8.1.1 Tecnologie relative al Client	65
8.2 Server	66
8.2.1 Tecnologie relative al Server	67
9 Diagrammi di Design	69
9.1 Class Diagram di Design	69
9.1.1 Punto 1 - Creazione Utenze per Dipendenti	69
9.1.2 Punto 3 - Personalizzazione del Menù	70
9.1.3 Punto 5 - Specifica degli Elementi in una Seconda Lingua	71
9.1.4 Punto 6 - Registrazione di Ordinazioni	72
9.1.5 Punto 4 - Stampa del QR Code	73
9.1.6 Punto 14 - Statistiche	73
9.2 Sequence Diagram di Design	74
9.2.1 Punto 3.1 - Personalizzazione del Menù - Categorie	74
9.2.2 Punto 3.2 - Personalizzazione del Menù - Elementi	74
9.2.3 Punto 6 - Registrazione di Ordinazioni	75
iv Testing e Valutazione Usabilità	76
10 Testing	76
10.1 Descrizione delle Strategie Adottate per la Progettazione dei Test	76
10.1.1 getByMenuIdAndAliment(Integer id, Aliment_Type aliment)	77
10.1.2 getByEmailAndPassword(String email, String password)	77
10.2 Codice xUnit	78
10.2.1 getByMenuIdAndAliment(Integer id, Aliment_Type aliment)	78
10.2.2 getByEmailAndPassword(String email, String password)	82
11 Valutazione Usabilità sul Campo	84
11.1 Test Sommativi	85
11.1.1 Esaminati	85
11.1.2 Task stabiliti	87
11.1.3 Sistema di Valutazione	87
11.1.4 Voti Parziali	87
11.1.5 Risultati dei Test	89
11.1.6 Commenti degli Esaminati	90
11.1.7 Conclusioni	90
11.2 Analisi	91
11.2.1 Utenti rilevati	91

11.2.2 Schermate Visualizzate	93
11.2.3 Tempo di Utilizzo	94

Parte I

Introduzione

Ratatouille23 è un software sviluppato per la gestione di **Attività di Ristorazione**.

In particolare, l'applicativo, permette un controllo totale dell'esercizio.

Sarà possibile :

- **Autenticazione**

L'utente ha possibilità di effettuare il Login, ed i suoi permessi varieranno a seconda della mansione che deve svolgere all'interno dell'attività.

- **Personalizzazione del Menù**

Admin e Supervisori potranno personalizzare completamente il Menù a loro piacimento, creando/eliminando/ordinando gli elementi del menù in categorie specificandone i dettagli.

- **QR Code**

Il menù potrà essere visualizzato dai clienti tramite un QrCode, che li indirizzerà ad un sito Web.

Questo stesso QrCode potrà essere scaricato in formato PDF dall'applicazione stessa, il che permetterà un'eventuale stampa del codice da poter apporre sui tavoli dell'attività.

- **Statistiche**

Sarà possibile visualizzare statistiche dettagliate sulla frequenza di ordinazione di determinati elementi tramite grafici interattivi.

- **Registrazione di Ordinazioni**

Gli Addetti alla Sala avranno la possibilità di gestire i tavoli dell'attività, potendo controllare il numero di tavoli liberi ed occupati, registrando ordinazioni dei clienti e presentando il conto in fase di liberazione del tavolo.

In questo documento andremo ad analizzare le fasi di Analisi e Specifica dei Requisiti Software, Sviluppo del Design di Sistema e Testing del sistema in questione.

Parte II

Requisiti Software

1 Requisiti

In questa sezione tratteremo e descriveremo nello specifico i **Requisiti Software** ricavati dalla traccia, in particolare da uno studio approfondito dei casi d'uso assegnati, e dal colloquio con i committenti.

Analizzeremo pertanto i requisiti **Funzionali**, **Non Funzionali** e **Di Dominio**.

Dati i casi d'uso a noi assegnati, ci limiteremo, per quanto riguarda i requisiti funzionali, a distinguere tra tre differenti ruoli presenti nell'ambito della gestione di un ristorante : **Admin**, **Supervisore**, **Addetto alla Sala**.

1.1 Funzionali

Sono qui descritti i requisiti funzionali del Software.

- **Admin :**

CREAZIONE DI UTENZE PER I DIPENDENTI.

L'Admin avrà la possibilità di creare Account per i suoi Dipendenti, i quali dovranno, al primo accesso, re-impostare la Password di Sicurezza, assegnata loro in prima istanza una di default dall'Admin.

PERSONALIZZAZIONE DEL MENÙ.

L'Admin potrà personalizzare completamente il Menù del Ristorante, aggiungendo e rimuovendo alimenti, creando categorie in cui organizzare questi ultimi e modificando l'ordine di visualizzazione di tali elementi sul menù.

STAMPA DEL QR CODE.

Sarà possibile per l'Admin stampare un QR Code, che rimanda ad un indirizzo web dal quale si potrà visualizzare il menù, da poter apporre sui tavoli del ristorante.

SPECIFICA DEGLI ELEMENTI IN UNA SECONDA LINGUA.

Al momento dell'inserimento di un alimento nel menù, l'admin, nel caso in cui il ristorante da lui gestito si trovi in una località turistica, avrà la possibilità di specificare il nome e la descrizione di tale alimento in una seconda lingua.

VISUALIZZAZIONE DI STATISTICHE.

L'Admin potrà visualizzare statistiche dettagliate sulla frequenza di ordinazioni degli elementi dal menù, avendo la possibilità di filtrare per elemento e per lasso di tempo.

- **Supervisore :**

PERSONALIZZAZIONE DEL MENÙ.

Permessi uguali a quelli dell'Admin.

- **Addetto alla Sala :**

REGISTRAZIONE DI ORDINAZIONI.

Un Addetto alla Sala avrà come compito principale quello di prendere le ordinazioni ai tavoli e registrarle tramite il Software.

1.2 Non Funzionali

Sono qui descritti i requisiti non funzionali del Software, seguendo le linee guida del modello FURPS.

- **Usability (Usabilità)**
- **Reliability (Attendibilità)**
- **Performance (Prestazioni)**
- **Supportability (Supportabilità)**

2 Analisi dei Requisiti

In questa sezione verranno analizzati i Requisiti Software precedentemente descritti.

Per la realizzazione dei vari Diagrammi di Analisi è stato usato il tool [Visual Paradigm](#).

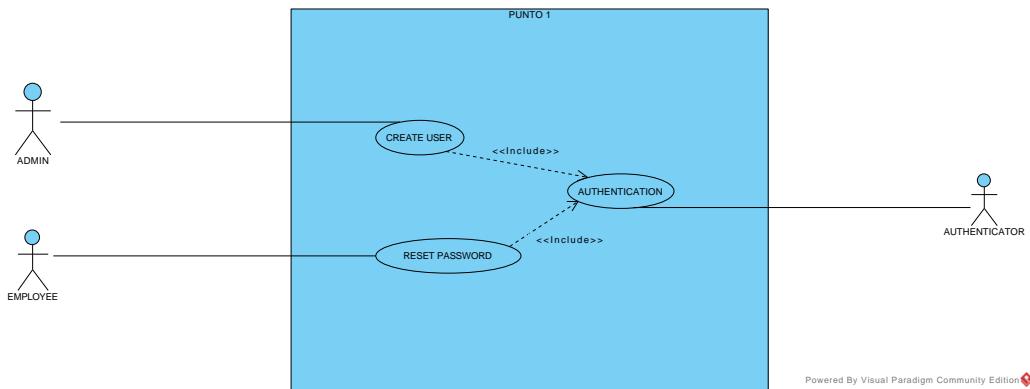
2.1 Modellazione dei Casi d'Uso

Proponiamo qui degli **Use Case Diagram** che descrivono i casi d'uso a noi assegnati.

Per rendere più leggibile il diagramma abbiamo scelto di scorporare lo use case dell'intero sistema in più Use Case Diagram, uno per ogni caso d'uso.

2.1.1 Punto 1 - Creazione di Utenze per i Dipendenti

In questo diagramma rappresentiamo la possibilità da parte dell'Admin di creare utenze per i dipendenti, i quali dovranno cambiare la password di default a loro assegnata quando il loro profilo è stato creato.



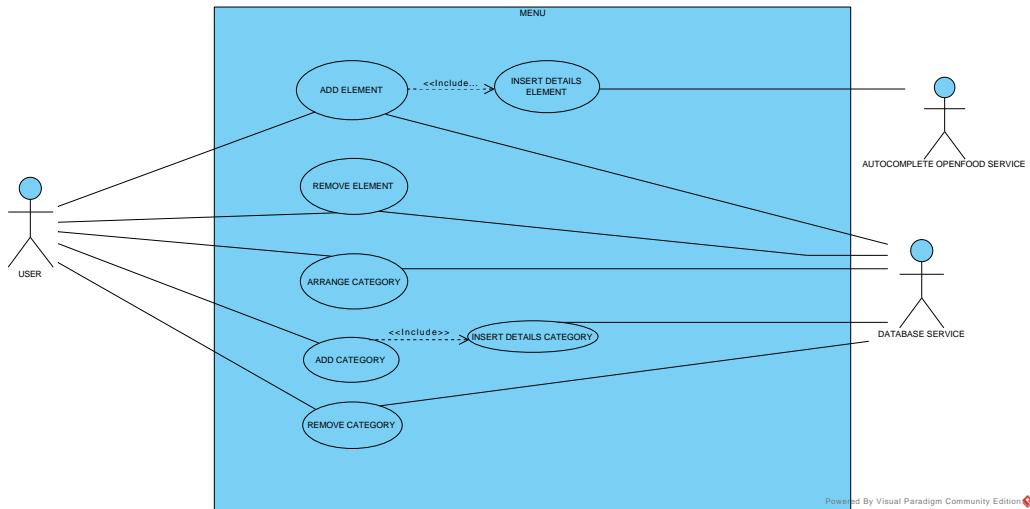
Powered By Visual Paradigm Community Edition

2.1.2 Punto 3 - Personalizzazione del Menù

In questo diagramma rappresentiamo la possibilità, da parte di Admin e Supervisor, di personalizzare il menù a loro piacimento.

In particolare sarà possibile :

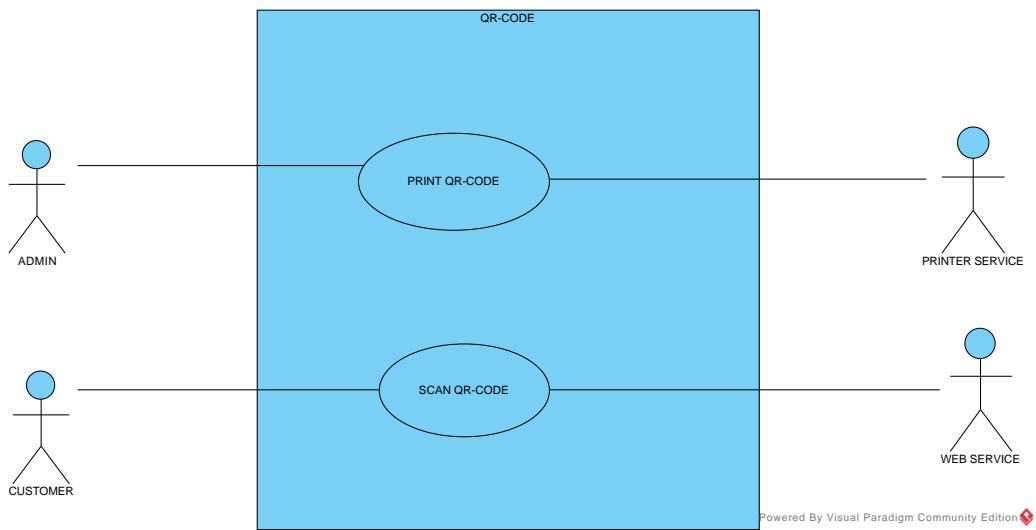
- Creare (specificandone i dettagli), ordinare ed eliminare categorie.
- Creare (specificandone i dettagli) ed eliminare elementi appartenenti ad una categoria.



2.1.3 Punto 4 - Stampa del QR Code

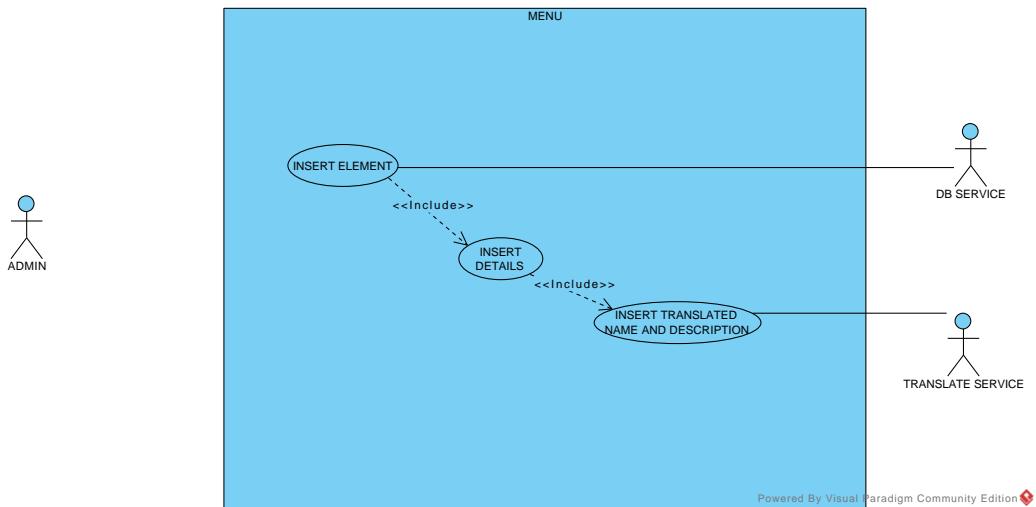
In questo diagramma rappresentiamo la possibilità da parte dell'Admin di scaricare il QrCode da apporre sui tavoli dell'attività in formato PDF.

Tale QrCode potrà essere scannerizzato dai clienti per visualizzare il menù tramite Browser.



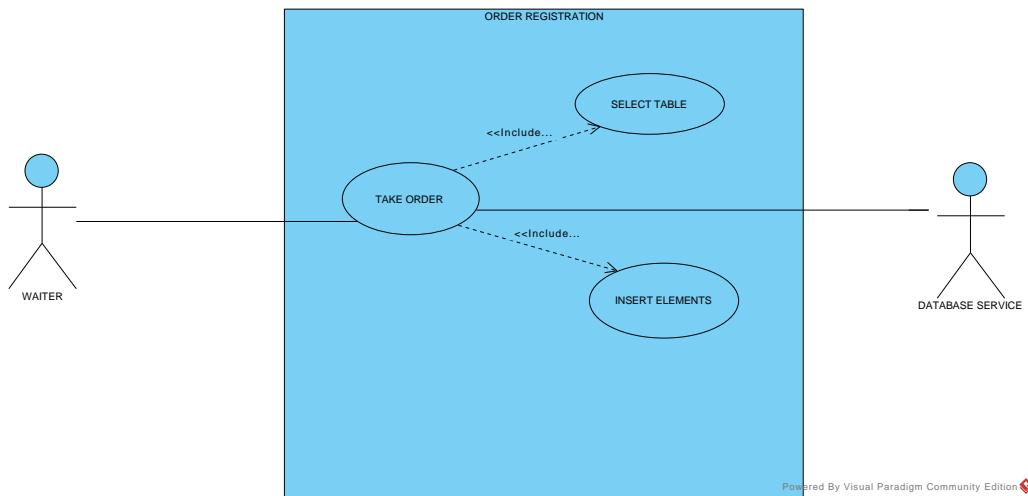
2.1.4 Punto 5 - Specifica degli Elementi in una Seconda Lingua

In fase di inserimento di un elemento all'interno del menù, se l'attività di ristorazione si trova in una località turistica, verranno automaticamente tradotti in lingua inglese i dettagli dell'elemento in questione, che compariranno dunque sul menù.



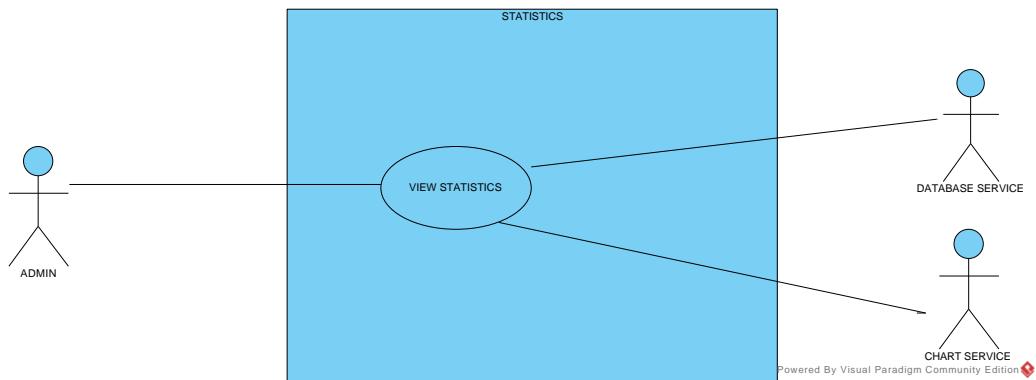
2.1.5 Punto 6 - Registrazione di Ordinazioni

Gli Addetti alla Sala potranno prendere le ordinazioni ai tavoli, prima selezionando il tavolo interessato dall’interfaccia fornita dal sistema, e poi creando un ordine composto dagli elementi del menù desiderati.



2.1.6 Punto 14 - Visualizzazione di Statistiche

Gli Admin avranno la possibilità, tramite grafici interattivi, di monitorare la propria attività da un punto di vista statistico, potendo controllare quante volte, in una finestra di tempo da loro stabilita, un elemento è stato ordinato.



2.2 Descrizioni Testuali Strutturate

Qui di seguito vediamo nello specifico due casi d'uso tra quelli assegnati, tramite una descrizione testuale strutturata, sulla base della **Template di Cockburn**.

Le due funzionalità che abbiamo scelto di analizzare più nello specifico sono :

- PERSONALIZZAZIONE DEL MENÙ
- REGISTRAZIONE DI ORDINAZIONI

2.2.1 Punto 3 - Personalizzazione del Menù

Use Case #01	Menu Management	
Goal in Context	Menu Management by Admin or Supervisor	
Preconditions	A Category must be selected	
Success End Condition	Successful Changes	
Failed End Condition	Unsuccessful Changes	
Primary Actor	Admin or Supervisor	
Trigger	Actor clicks on "Edit Menu Element" bar	
DESCRIPTION	Step n°	Admin or Supervisor
	1	Choose Action
	2	Saves Changes
	3	Return to Step 1 (Main)
EXTENSIONS #1 "Changes Refused"	Step n°	Admin or Supervisor
	3.a	Discard Changes
	4.a	Return to Step 1 (Main)
	Step n°	Admin or Supervisor
SUBVARIATION #1 "Add element"	1	Clicks on "+"
	2	Open "Add Element" Window
	3	Inserts Element Name
	4	Inserts Element Description
	5	Inserts Element Price
	6	Inserts Element Allergens
	7	Checks Prepackaged Box
	8	Clicks on "Confirm"
	9	Verify Element Details
	10	Close "Add Element" Window
	11	Return to Step 2 (Main)
SUBVARIATION #2 "Remove element"	Step n°	Admin or Supervisor
	1	Clicks on "
	2	Changes Elements Interface
	3	Select Elements to Remove
	4	Clicks on "Confirm"
	5	Return to Step 2 (Main)
SUBVARIATION #3 "Arrange elements"	Step n°	Admin or Supervisor
	1	Choose on "Arrange"
	2	Changes Elements Interface
	3	Drags Element to a New Position
	4	Return to Step 2 (Main)

2.2.2 Punto 6 - Registrazione di Ordinazioni

Use Case #02	Order Registration		
Goal in Context	Taking Order		
Preconditions	A Table must be Selected and Occupied		
Success End Condition	Order Sent		
Failed End Condition	Order Unsent		
Primary Actor	Waiter		
Trigger	Actor clicks on "Edit Orders" bar		
DESCRIPTION	Step n°	Actor	System
	1	Clicks on Table	
	2		Loads clicked Table info
	3	Clicks on "+"	
	4		Opens "Add Order" Window
	5		Insert Selected Table ID
	6	Add Elements to Order	
	7	Clicks on "Confirm"	
	8		Closes "Add Order" Window
	9		Save Order
	10		Return to Step 1 (Main)
EXTENSIONS #1 "Table not Selected"	Step n°	Actor	System
	4.a		Opens "Error : Table not selected" Window
	5.a	Clicks on "Ok"	
	6.a		Closes "Error : Table not selected" Window
	7.a		Return to Step 1 (Main)

3 Mockup

Un Mockup è una rappresentazione di un prodotto a scopo puramente illustrativo, utilizzato per mostrare in maniera rapida ed intuitiva l'idea di realizzazione di un progetto.

I Mockup presentati di seguito sono stati realizzati con lo strumento di prototipazione [Figma](#).

3.1 Primo Prototipo

Durante la fase iniziale di progettazione dei Mockup abbiamo ottenuto una prima realizzazione che però non ci soddisfava, sia dal punto di vista meramente estetico sia dal punto di vista funzionale, quindi è stato successivamente scartato.

Per avere una panoramica completa sull'intera evoluzione del progetto ci è sembrato interessante mostrare anche questo primo prototipo.

Il primo prototipo è visualizzabile a questo [link](#), inserendo come password

INGSW2223_N_25.

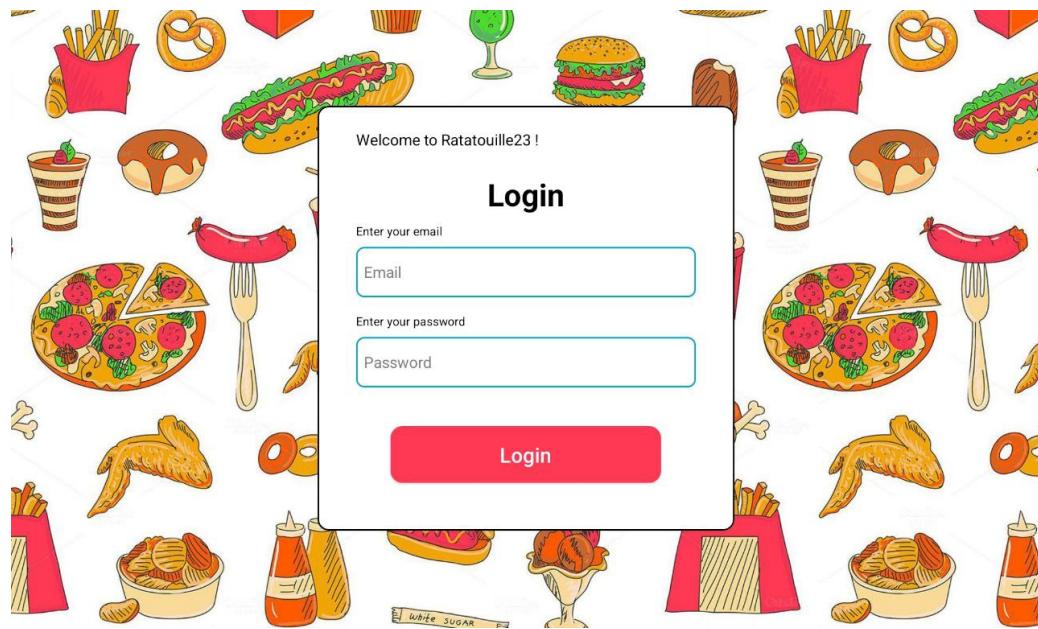
3.2 Secondo Prototipo (Definitivo)

Il secondo prototipo che abbiamo realizzato è invece quello definitivo, ed è visualizzabile a questo [link](#), inserendo come password **INGSW2223_N_25**.

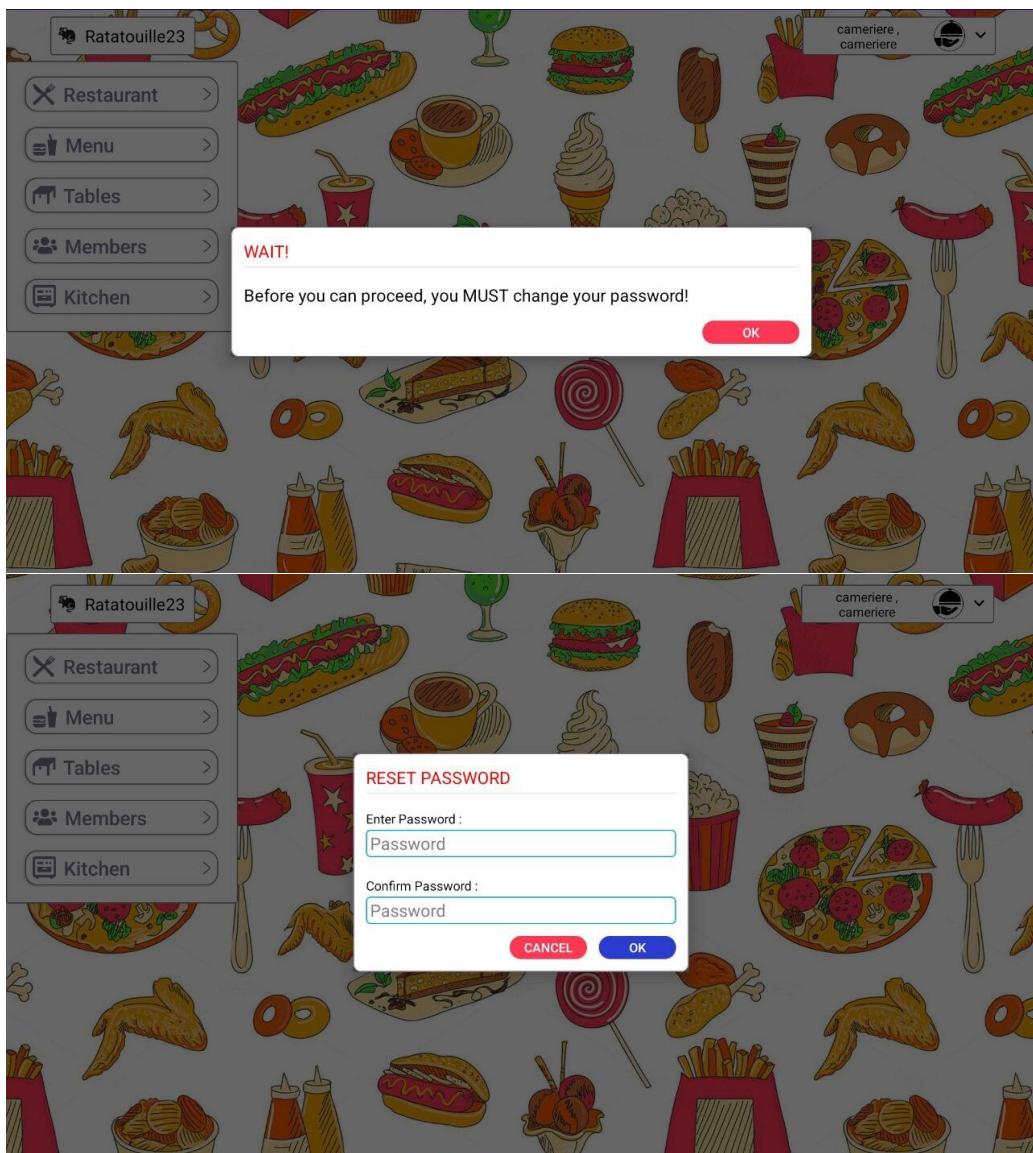
3.3 Schermate

Qui di seguito mostriamo solo alcune delle varie schermate dell'applicativo.

3.3.1 Autenticazione



3.3.2 Cambio obbligatorio della password al primo accesso



3.3.3 Visualizzazione Informazioni e Statistiche del Ristorante

The screenshot displays a user interface for managing a restaurant. At the top, there's a navigation bar with icons for a user profile ('Ratatouille23'), a search bar, and various food-related icons like a pizza, a sandwich, a wine glass, a burger, and a salad. On the right side of the top bar, there's a dropdown menu showing 'admin, admin'.

The main content area has a header 'Restaurant'. On the left, a sidebar lists navigation options: 'Restaurant' (selected), 'Menu', 'Tables', 'Members', and 'Kitchen'. Below the sidebar, there are several food-related illustrations: a pizza, a bowl of soup, a sandwich, a salad, and some fries.

The central part of the screen is divided into two main sections:

- Info:** This section contains details about the restaurant:
 - Name: rest
 - Description: descrizione ristorante rest
 - Locality: italia
 - Touristic: YesA QR code is also displayed here.
- Stats:** This section shows a bar chart comparing two menu items:

Item	Count
POLLO FRITTO	12
RISOTTO ALLO ZAFFERANO	6

3.3.4 Gestione del Menù

The image displays two screenshots of a restaurant management application interface, specifically focusing on menu item selection.

Screenshot 1: The user has selected the "PRIMI" category. The right panel shows three menu items with their descriptions, prices, and allergen information. The first item is "PENNNETTE ALLA NORMA".

Element	Description	Price	Prepackaged	Allergens
PENNNETTE ALLA NORMA	melanzane, pomodoro, ricotta salata	€ 11.99	No	milk, wheat
RISOTTO ALLO ZAFFERANO	zafferano, parmigiano, cipolla	€ 12.99	No	milk
SPAGHETTI AI FRUTTI DI MARE	frutti di mare, prezzemolo, aglio	€ 15.99		

Screenshot 2: The user has deselected the "PRIMI" category, as indicated by the radio button being empty. The right panel now shows an empty list of selected items.

Categories

FOOD	DRINK
ANTIPASTI	
CONTORNI	
SECONDI	
PRIMI	
PIZZE	
DESSERT	

Elements

- +

CREATE CATEGORY

Enter Name:

CANCEL OK

ZAFFERANO

Description: zafferano, pomodoro, ricotta salata

Price: € 12.99

Prepackaged: No

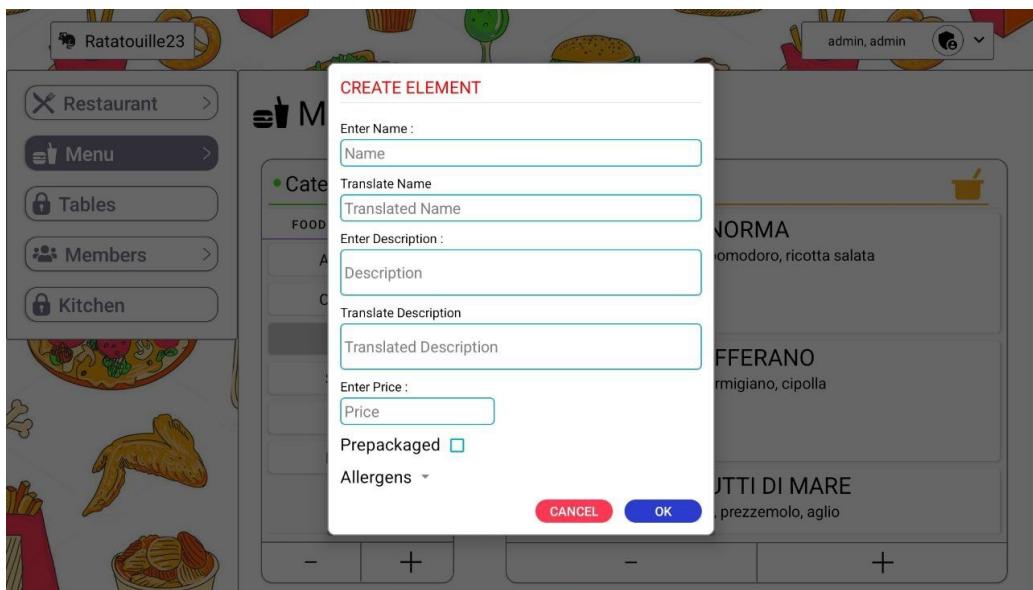
Allergens: milk

SPAGHETTI AI FRUTTI DI MARE

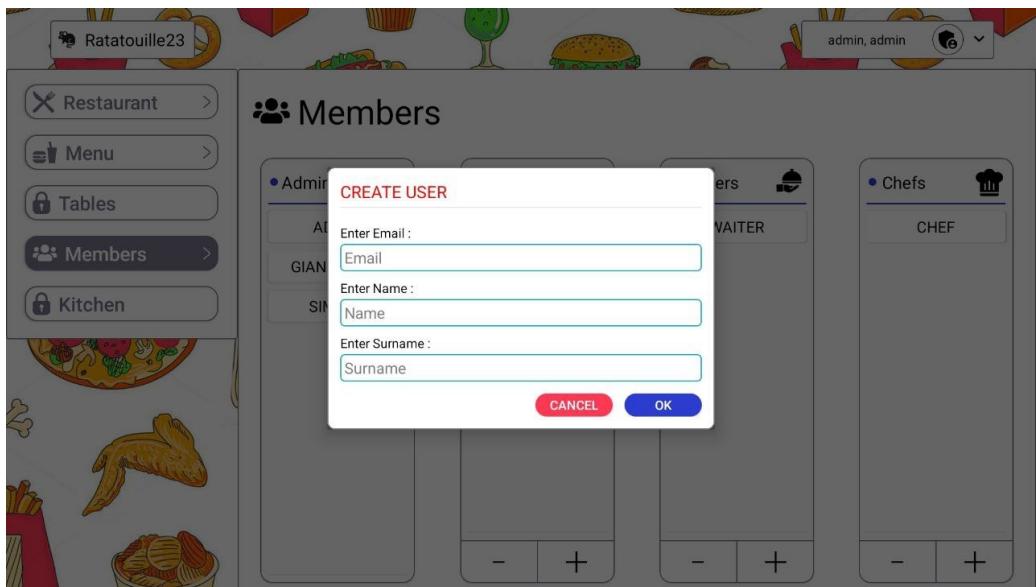
Description: frutti di mare, prezzemolo, aglio

Price: € 15.00

- +



3.3.5 Gestione Utenze



3.3.6 Gestione Tavoli

The screenshot displays two main windows of a restaurant management application.

CREATE ORDER window (Top):

- PIZZA:** MARGHERITA, MARINARA
- DESSERT:** CHEESECAKE, TIRAMISU
- BIRRA:** PERONI, FRANZISKANER
- ANALCOLICI:**

Order summary:

- MARINARA x 2
- PERONI x 2

Buttons: CANCEL, OK

Tables window (Bottom):

- Restaurant**, **Menu**, **Tables** (selected), **Members**, **Kitchen**
- Total:** 8
- Free:** 4
- Occupied:** 4
- Overview:** Table n° 1 Occupied (represented by a red icon)
- Table n° 2:** Free (represented by a green icon)
- Table Selected:**
 - Number of Seats: 2
 - Orders
 - Elements: MARINARA, MARINARA, PERONI, PERONI
 - Total: € 16.0
 - Buttons: -, +
 - LIBERA

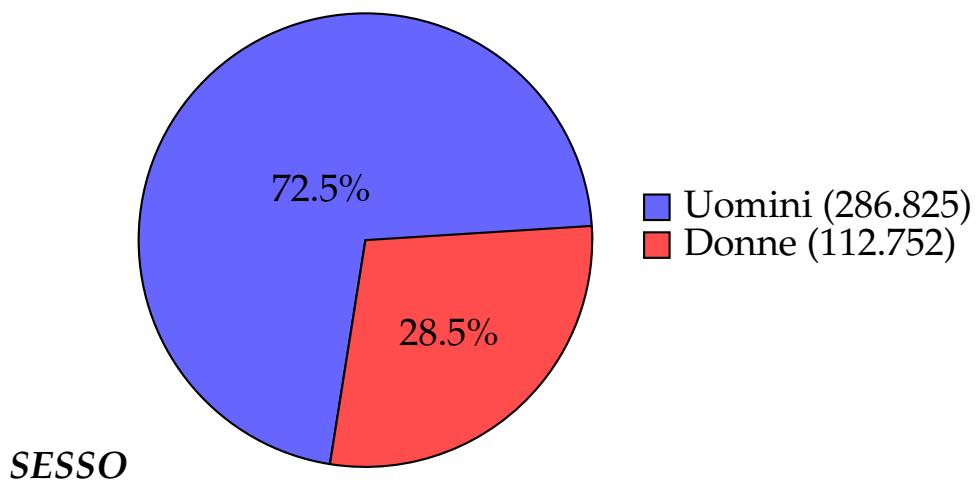
4 Individuazione Target di Utenti

Analizzando i requisiti funzionali dell'applicativo, possiamo identificare un target di utenti, composto principalmente da :

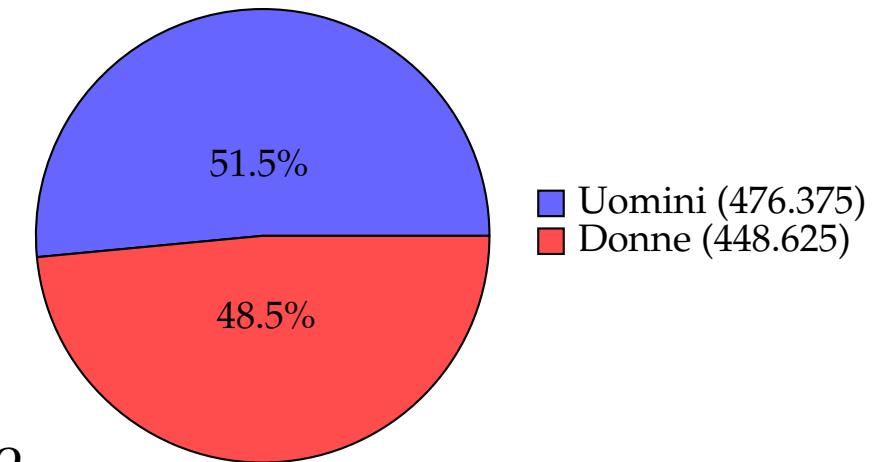
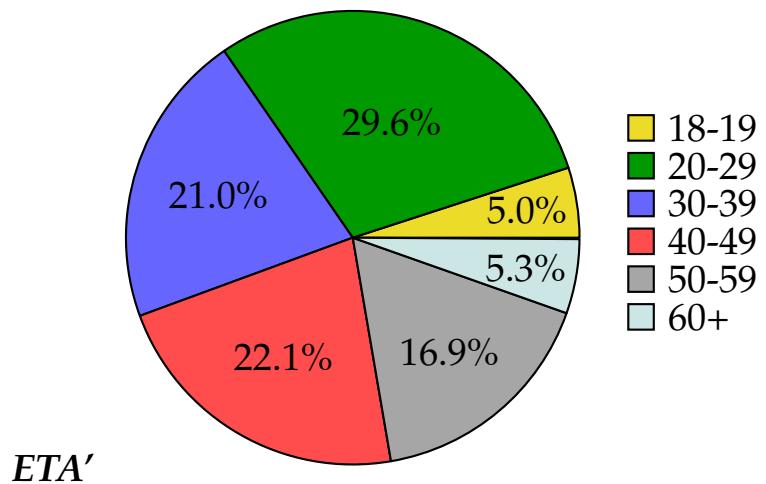
- Proprietari di Attività di Ristorazione
- Dipendenti specializzati in Gestione della Sala

4.1 Statistiche

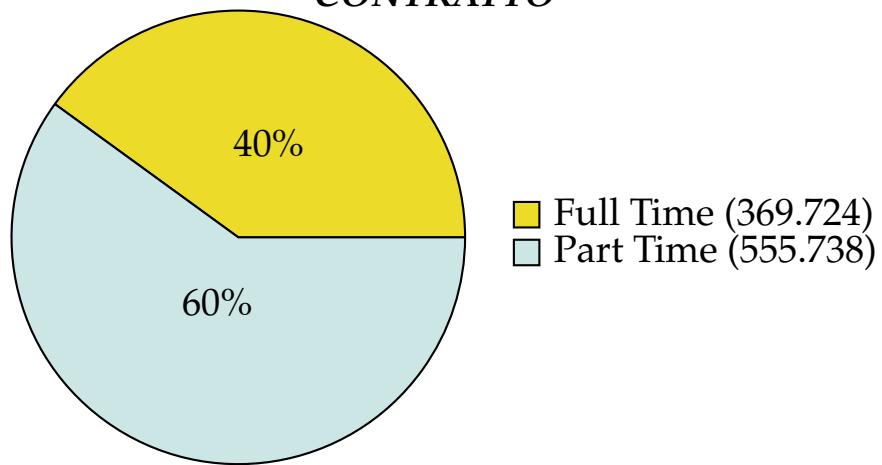
Qui di seguito riportiamo qualche statistica riguardante Proprietari di Attività di Ristorazione.



Qui di seguito riportiamo qualche statistica riguardante Addetti alla Sala.



CONTRATTO



4.2 Personas

Una delle domande principali che un progettista deve porsi nel momento in cui inizia lo sviluppo di un prodotto è : " *A chi è destinato?*".

Comprendere il concetto di **Persona** è fondamentale per una buona riuscita di un prodotto.

Una Persona è un modello, un individuo immaginario che sintetizza ed incarna il consumatore-tipo del prodotto da realizzare.

Identificare le Personas correttamente permette ai progettisti di potersi focalizzare solo sui bisogni di uno o più modelli, senza dover provare a soddisfare i bisogni di un numero indefinito di persone.

4.2.1 Criteri per l'identificazione delle Personas

Analizzando i dati statistici riportati precedentemente, abbiamo deciso di prendere in considerazione, per l'identificazione delle **Personas** :

- Età
- Sesso
- Tipo di Contratto

4.2.2 Personas Identificate

Dati i tre parametri stabiliti in precedenza, abbiamo identificato come Personas quattro individui rappresentativi.

Avremo dunque due proprietari di Attività di Ristorazione e due Addetti alla Sala.

Aurelio De Kvaratskhentiis



Età : 58

Lavoro : Proprietario di
un'attività di
Ristorazione

Lorena Sofia



Età : 72

Lavoro : Proprietaria di
un'attività di
Ristorazione

Luciano Spallusotti



Età : 41

Lavoro : Cameriere

Contratto : Full Time

Caterina Chercofisi



Età : 23

Lavoro : Cameriera

Contratto : Part Time

5 Usabilità a Priori

5.1 Considerazioni sull'Usabilità

La valutazione dell'usabilità a priori del nostro sistema è stato un passo fondamentale per riuscire a comprendere se il lavoro svolto fino al momento della valutazione fosse stato corretto e coerente con i cinque criteri in cui viene scomposta e con cui è misurata l'usabilità secondo Jakob Nielsen.

I criteri in questione sono :

- **APPRENDIBILITÀ'**

Il sistema deve essere di facile utilizzo, dovrebbe essere dunque facile per utenti esperti o novizi capire come raggiungere un obiettivo.

- **EFFICIENZA**

Quantità di risorse spese in relazione all'accuratezza con cui si raggiunge l'obiettivo preposto.

In particolare, un utente esperto deve poter giovare della sua esperienza col prodotto per completare in maniera più rapida un compito.

- **MEMORABILITÀ'**

Il sistema deve essere facile da ricordare, in modo tale da poter agevolare il suo utilizzo anche da parte di utenti "occasionali".

- **ERRORI**

Il sistema deve tentare di impedire all'utente di sbagliare e, qualora non ci riuscisse, deve assicurare di poter risolvere situazioni di errore.

- **SODDISFAZIONE**

Gradimento dell'azione svolta, che si riflette in un'attitudine positiva nei confronti del prodotto.

Andiamo dunque ad analizzare uno ad uno questi criteri focalizzandoci sulle scelte prese a tal proposito per la realizzazione dei prototipi di cui abbiamo discusso in precedenza.

5.1.1 Apprendibilità

Questo fattore ci è parso, considerando i dati statistici a proposito dei tipi di contratto per gli addetti alla sala che abbiamo rilevato, uno dei più importanti.

A nostro parere infatti, conferire un'elevata apprendibilità al nostro sistema avrebbe permesso anche a dipendenti occasionali di poter lavorare al meglio nonostante la scarsa esperienza con l'applicativo. Nostro obiettivo primario è stato quindi ottenere una buona **Affordance**, in modo tale da ridurre l'ampiezza del **Golfo dell'Esecuzione**, e mantenere una **Bassa Soglia di Apprendimento**.

Questo obiettivo è stato raggiunto in vari modi :

- Mantenendo un flusso di operazioni che andasse da sinistra verso destra, sviluppando la nostra applicazione esclusivamente in orizzontale, in quanto è il modo più naturale di "guardare" qualcosa.
- Utilizzando nomi esplicativi per le schermate, i bottoni e le finestre.

5.1.2 Efficienza

Sviluppando l'applicativo su un Tablet, abbiamo ragionato a come poter agevolare gli Addetti alla Sala.

A questo proposito abbiamo optato ad una soluzione che ci permettesse di ridurre la parte di schermo in cui vengono svolte la maggior parte delle operazioni, mantenendo una sezione fissa, sulla sinistra, la cui unica funzione è quella di cambiare la schermata visualizzata nel resto dello schermo.

Questa scelta, a nostro parere, rende possibile l'utilizzo dell'applicativo con una sola mano, in quanto l'altra sarà per lo più occupata a mantenere il dispositivo.

Un'altra scelta presa è stata quella di ridurre al minimo i "tocchi" necessari per svolgere le operazioni.

5.1.3 Memorabilità

Riprendendo quanto descritto nel paragrafo riguardante l'apprendibilità, abbiamo tentato di rendere il più semplice possibile l'utilizzo del sistema.

Ovviamente è opportuno dire che una conoscenza, pur superficiale, del sistema, rende l'esperienza di utilizzo più agevole e rapida.

5.1.4 Errori

Il tema della gestione degli errori è delicato quando si valuta un prodotto.

Riprendendo il concetto di "linearità" che abbiamo conferito al sistema, il nostro obiettivo principale è stato quello di prevenire il più possibile gli errori attraverso varie scelte :

- Gestendo gli spazi per le varie sezioni in maniera efficiente, evitando di avere troppi elementi ravvicinati e riducendo, di conseguenza, tocchi involontari.
- Rendendo estremamente esplicative le sezioni del sistema, evitando dunque errori derivanti da una "non comprensione" di quello che si sta facendo.

Ovviamente siamo sempre stati coscienti del fatto che errori possono sempre capitare durante l'utilizzo di un prodotto, ma la possibilità di eliminare Categorie/Elementi/Ordini inseriti nel DataBase erroneamente conferisce, a nostro parere, una buona ed intuitiva risoluzione degli errori.

Sono stati inoltre inseriti PopUp di errore esplicativi qualora si provasse ad effettuare un'operazione i cui prerequisiti non sono stati soddisfatti.

5.1.5 Soddisfazione

Per garantire un buon livello di soddisfazione all'utente abbiamo cercato di restituire un buon **Feedback**, utile anche per l'apprendibilità, sia, banalmente, tramite i bottoni cliccati, sia aggiornando istantaneamente la visualizzazione in modo tale da far capire all'utente che la modifica effettuata ha avuto effetto.

5.2 Valutazione Usabilità a Priori

I test da noi effettuati sull'Usabilità si sono tenuti subito dopo aver terminato la realizzazione dei Prototipi, concepiti da noi per essere completi dal punto di vista funzionale, in modo tale da ottenere dei risultati validi di conseguenza anche per il prodotto ultimato.

E' importante dire che, essendo test effettuati sulla base di prototipi e non dell'applicativo finito, si chiederà agli esaminati di simulare il flusso di operazioni tramite il Tool di prototipazione , sfruttando la feature di "Transizione" fra schermate offerta da [Figma](#).

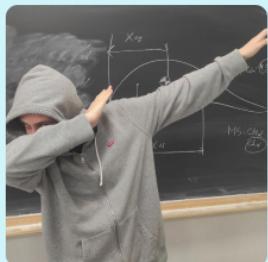
A prodotto ultimato verranno effettuati gli stessi Test con gli stessi Esaminati per valutare l'Usabilità sul Campo dell'applicativo pronto per la distribuzione.

I test da noi effettuati sono così composti :

- 2 Esinatori (gli sviluppatori del sistema).
- 4 Esaminati con competenze diverse (delineate successivamente nella presentazione degli esaminati).
- 4 Task da portare a termine.

5.2.1 Esaminati

Giorgio



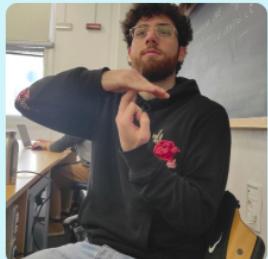
Età : 24

Lavoro : Architetto

Esperienze nel mondo della Ristorazione : Nessuna

Competenze Informatiche : Buone

Renato



Età : 21

Lavoro : Studente di Informatica

*Esperienze nel mondo della Ristorazione : Barista
part-time*

Competenze Informatiche : Ottime

Dario



Età : 22

Lavoro : Studente di Informatica

Esperienze nel mondo della Ristorazione : Nessuna

Competenze Informatiche : Ottime

Marco



Età : 25

Lavoro : Ingegnere Meccanico

Esperienze nel mondo della Ristorazione : Cameriere full-time

Competenze Informatiche : Buone

5.2.2 Task stabilité

Abbiamo identificato 4 compiti da far svolgere alle persone esaminate, cercando di scegliere funzionalità relativamente complesse e complete, in modo tale da poterli porre nel "caso peggiore".

I Task assegnati sono dunque i seguenti :

- Effettuare il login (con credenziali note) e visualizzare statistiche relative ad un lasso di tempo che va dal 01/01/2023 al 31/03/2023.
- Effettuare il login (con credenziali note) e creare una categoria denominata "Pre Dessert".
- Effettuare il login (con credenziali note) e creare un'ordinazione per il Tavolo 3 composta da 2 Margherite e 2 Peroni.
Visualizzare successivamente il totale dell'ordinazione.
- Effettuare il login (con credenziali note) ed eliminare il cameriere chiamato "Kvicha".

5.2.3 Sistema di Valutazione

Abbiamo deciso di far attribuire ad ogni esaminato un voto compreso tra 1 e 10 per ognuno dei criteri fondamentali per la valutazione dell'usabilità precedentemente descritti.

Dopo aver quindi ottenuto un voto per ogni criterio da ogni esaminato, avremo il risultato complessivo del Test effettuando una media aritmetica.

5.2.4 Voti Parziali

Dopo aver spiegato per grandi linee agli Esaminati i cinque criteri con cui abbiamo stabilito di valutare l'Usabilità del prodotto, ab-

biamo chiesto ai candidati di assegnare, per ogni test, un voto ad ognuno di questi parametri, per poi ottenere un voto complessivo per ognuno dei Task assegnati.

DARIO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	8	9	9	10
Efficienza	9	10	8	9
Memorabilità	9	10	8	10
Errori	10	7	9	8
Soddisfazione	8	9	9	9

GIORGIO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	10	9	8	8
Efficienza	7	9	9	8
Memorabilità	10	9	8	8
Errori	9	9	7	9
Soddisfazione	9	8	9	10

MARCO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	9	8	10	8
Efficienza	9	8	7	8
Memorabilità	10	8	7	8
Errori	9	9	10	7
Soddisfazione	8	8	10	9

RENATO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	8	8	9	9
Efficienza	9	8	9	10
Memorabilità	8	8	7	9
Errori	10	9	9	8
Soddisfazione	9	10	8	8

5.2.5 Risultati dei Test

	Test 1	Test 2	Test 3	Test 4
Dario	8,8	9	8,6	9,2
Giorgio	9	8,8	8,2	8,6
Marco	9	8,2	8,8	8,6
Renato	8,8	8,6	8,4	8,8

	Test 1	Test 2	Test 3	Test 4
Apprendibilità	8,75	8,5	9	8,75
Efficienza	9,25	9	8,75	8,75
Memorabilità	9,25	8,75	8,75	9
Errori	9,5	8,5	8,75	8,25
Soddisfazione	8,5	8,75	9	9

5.2.6 Commenti degli Esaminati

Dario	Ho trovato i bottoni dell'applicazione un po' troppo piccoli, il che potrebbe portare a tocchi involontari. Oltre questo però ho trovato il tutto abbastanza intuitivo.
Giorgio	L'ho trovato semplice da utilizzare, anche per chi come me non ha nessuna esperienza in questo ambito.
Marco	Ho gradito molto l'organizzazione delle varie sezioni dell'app. Mi piacciono i colori dell'applicazione.
Renato	Ho apprezzato per il momento i prototipi, aspetto di vedere il sistema completo.

5.2.7 Conclusioni

Abbiamo ottenuto dunque un **Valore di Usabilità** pari a **8,71/10**. Abbiamo considerato tale valore un buon risultato, però abbiamo sfruttato qualche consiglio degli Esaminati per apportare piccole modifiche, perlopiù estetiche, al nostro prototipo e di conseguenza al prodotto finale.

6 Glossario

REQUISITI FUNZIONALI	Descrivono le funzionalità offerte dal sistema all'utente.
REQUISITI NON FUNZIONALI	Descrivono vincoli sui servizi offerti dal sistema.
REQUISITI DI DOMINIO	Descrivono vincoli generali del dominio applicativo.
CASI D'USO	Requisiti funzionali del sistema.
TEMPLATE DI COCKBURN	Descrizione testuale strutturata di uno Use Case Diagram e specifica le interazioni attore/sistema.
MOCKUP	Rappresentazione di un prodotto a scopo puramente illustrativo, utilizzato per mostrare in maniera rapida ed intuitiva l'idea di realizzazione di un progetto.

USE CASE DIAGRAM	Diagramma UML che descrive l'insieme dei requisiti funzionali di un sistema.
CLASS DIAGRAM	Diagramma UML statico che descrive Classi e Relazioni tra classi. Permette di identificare gli elementi di base del Software da sviluppare.
SEQUENCE DIAGRAM	Diagramma UML di interazione che mostra la sequenza di messaggi scambiati tra gli oggetti al fine di completare un'operazione.
STATECHART	Diagramma UML che descrive i possibili stati in cui si può trovare un'entità, e in che modo avvengono le transizioni tra questi stati.

ADMIN	Crea utenze per i dipendenti, gestisce gli elementi del menù, ha accesso alle statistiche che riguardano l'attività di ristorazione e ha la possibilità di stampare il QR Code che rimanda al menù.
SUPERVISORE	Dipendente che si occupa della gestione degli elementi del menù.
ADDETTO ALLA SALA	Dipendente dell'attività che si occupa della gestione dei tavoli della sala, registrando le ordinazioni dei clienti.
MENÙ	L'intero insieme degli elementi ordinabili nell'attività di ristorazione.
ALIMENTI	Insieme di categorie ad alto livello di elementi, costituita da Food e Drink.
CATEGORIA	Insiemi di elementi, utile per visualizzare in maniera più intuitiva il menù.
ELEMENTI	Tutto quello che può essere ordinato dal menù, sono suddivisi in categorie.
TAVOLO	Tavolo di sala, che viene occupato quando i clienti si siedono, e liberato nel momento in cui questi ultimi richiedono il conto.
ORDINAZIONE	Insieme di elementi ordinati dai clienti in un determinato tavolo. Possono essere effettuate più ordinazioni ad uno stesso tavolo.

7 Specifica dei Requisiti

Dopo una prima panoramica generale sulle caratteristiche del Software, scendiamo di un livello, e ci concentriamo sulla specifica dei requisiti, descrivendo questi ultimi sotto forma di:

- **Class Diagram**
- **Sequence Diagram**
- **Statecharts Funzionali**

Per identificare le entità e le relazioni tra di esse abbiamo sfruttato l'**Euristica EBC** :

- **Entity** : Modella le informazioni.
- **Boundary** : Modella le interazioni che si instaurano tra attori e sistema.
- **Control** : Modella la logica incaricata di realizzare le funzionalità.

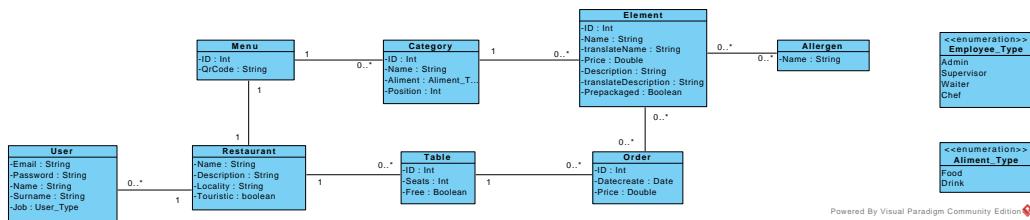
7.1 Class Diagram di Analisi

Presentiamo ora i Class Diagram di Analisi.

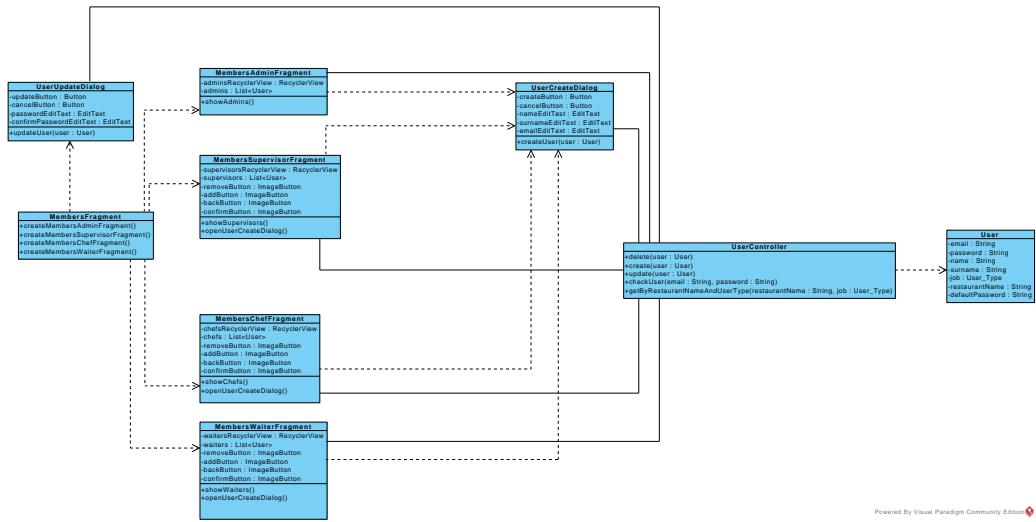
Ci teniamo a sottolineare che metodi Getter&Setter sono stati esclusi dal Diagramma, così come dipendenze ridondanti, per conferire una maggiore leggibilità ai diagrammi.

7.1.1 Class Diagram delle Entità

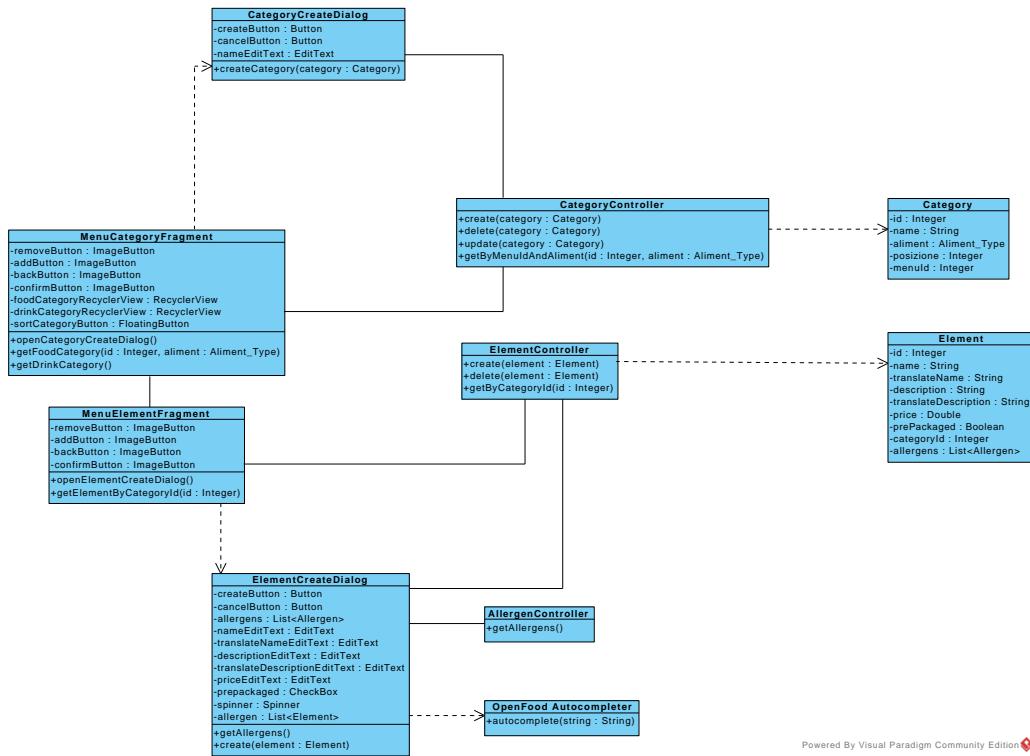
Qui di seguito il Class Diagram rappresentante le entità che abbiamo identificato e le associazioni tra esse.



7.1.2 Punto 1 - Creazione Utenze per Dipendenti

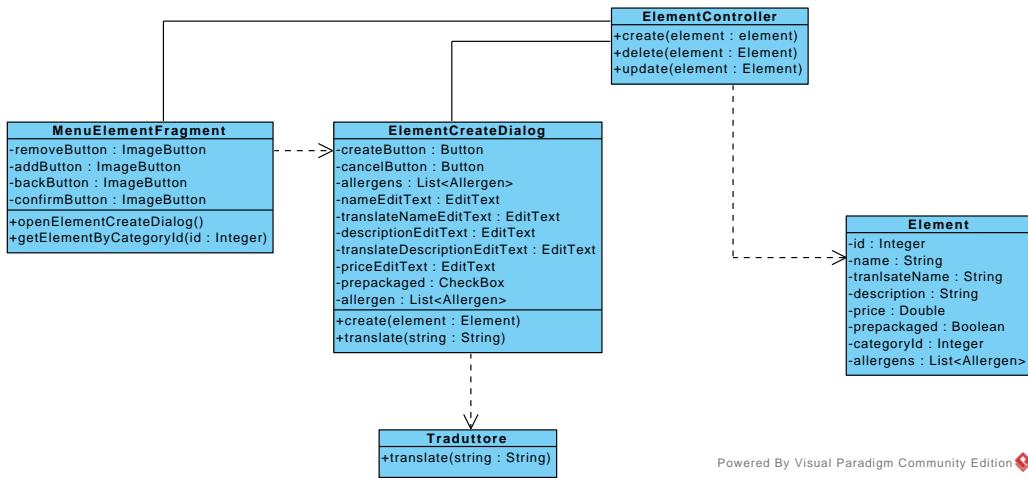


7.1.3 Punto 3 - Personalizzazione del Menù



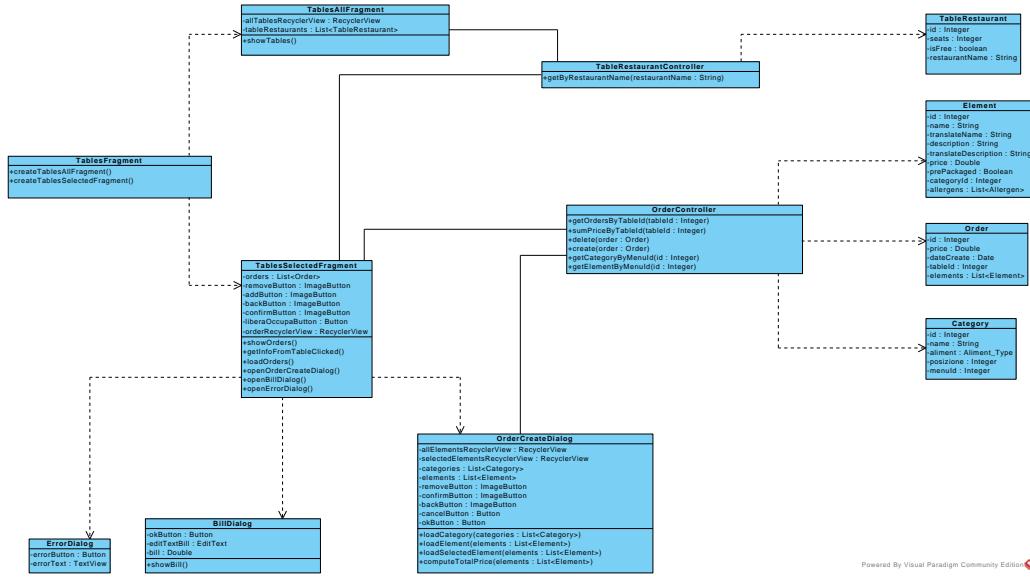
Powered By Visual Paradigm Community Edition

7.1.4 Punto 5 - Specifica degli Elementi in una Seconda Lingua



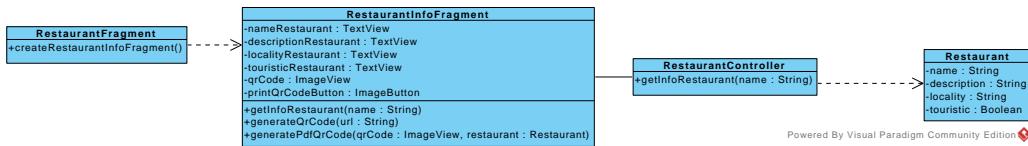
Powered By Visual Paradigm Community Edition

7.1.5 Punto 6 - Registrazione di Ordinazioni

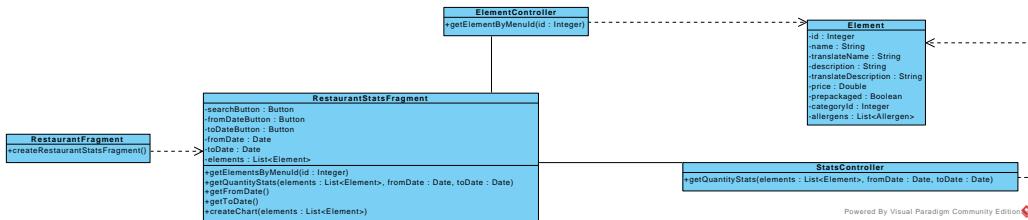


Powered By Visual Paradigm Community Edition

7.1.6 Punto 4 - Stampa del QR Code



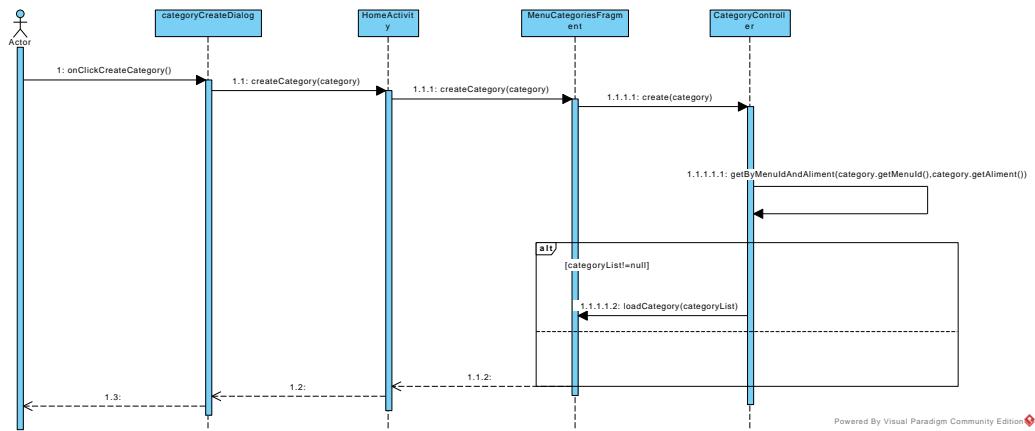
7.1.7 Punto 14 - Statistiche



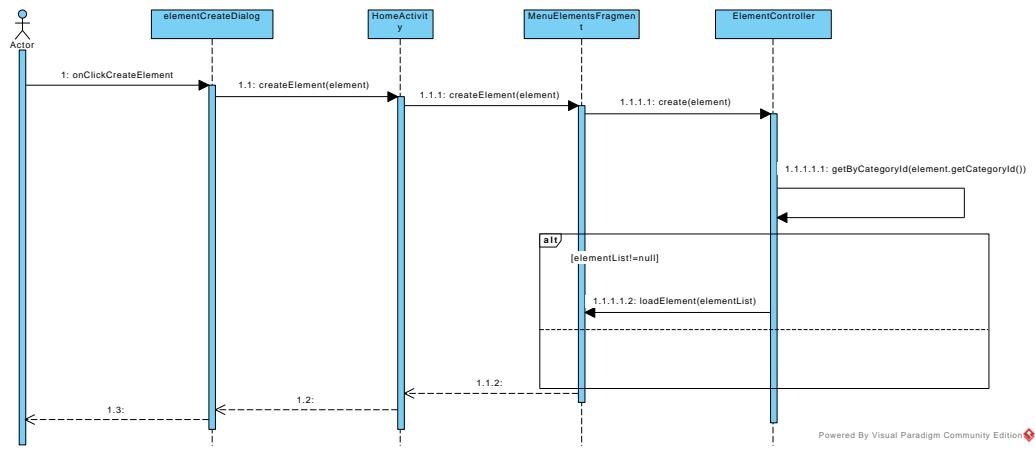
7.2 Sequence Diagram di Analisi

Qui di seguito sono descritte, sotto forma di **Sequence Diagram**, le due funzionalità precedentemente analizzate tramite Template di Cockburn.

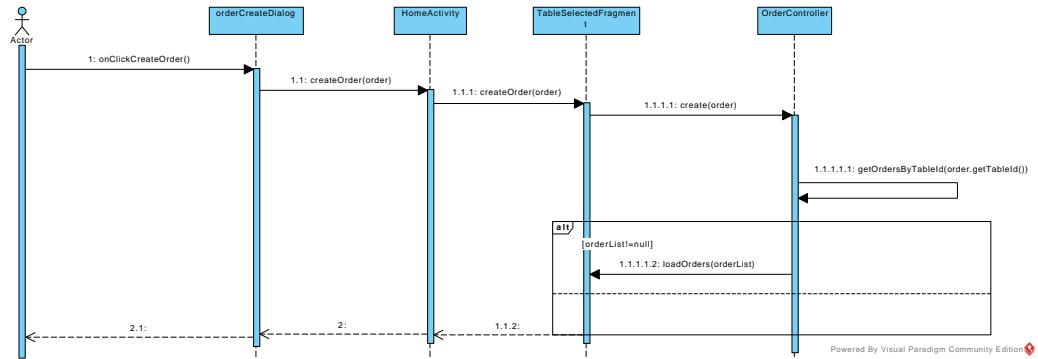
7.2.1 Punto 3.1 - Personalizzazione del Menù - Categorie



7.2.2 Punto 3.2 - Personalizzazione del Menù - Elementi



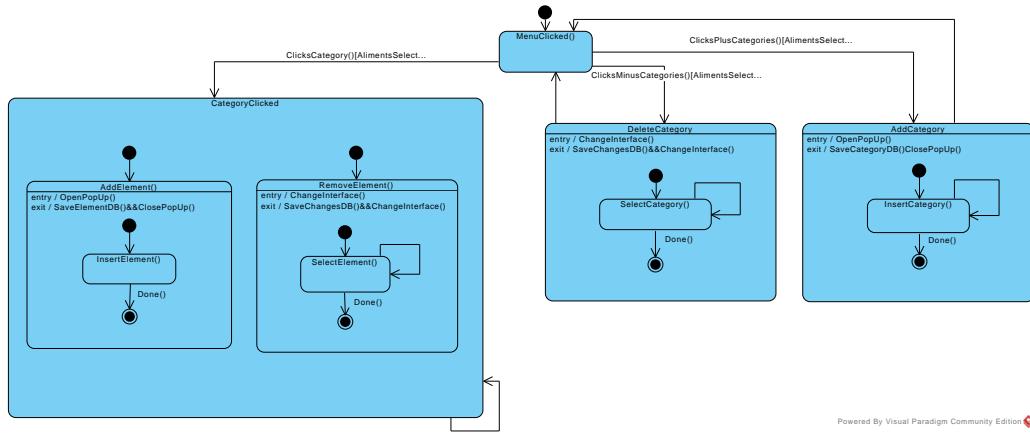
7.2.3 Punto 6 - Registrazione di Ordinazioni



7.3 Statecharts Funzionali

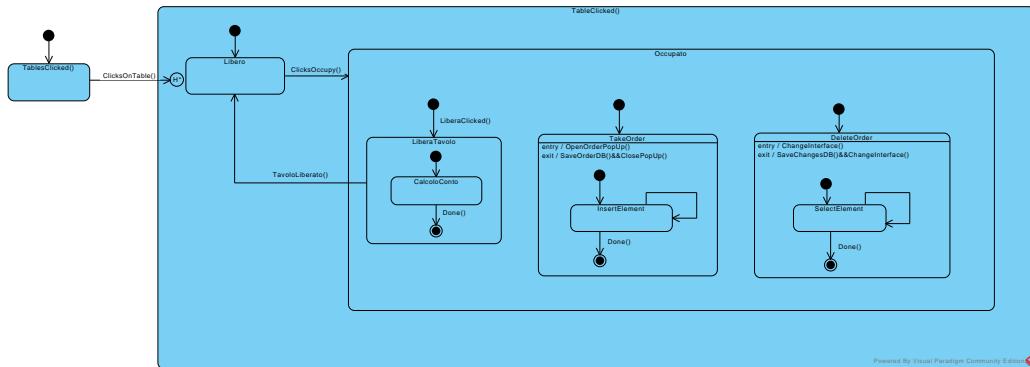
Continuiamo la specifica dei requisiti esprimendo le due funzionalità da noi scelte sotto forma di **Statechart Funzionali**.

7.3.1 Punto 3 - Personalizzazione del Menù



Powered By Visual Paradigm Community Edition

7.3.2 Punto 6 - Registrazione di Ordinazioni

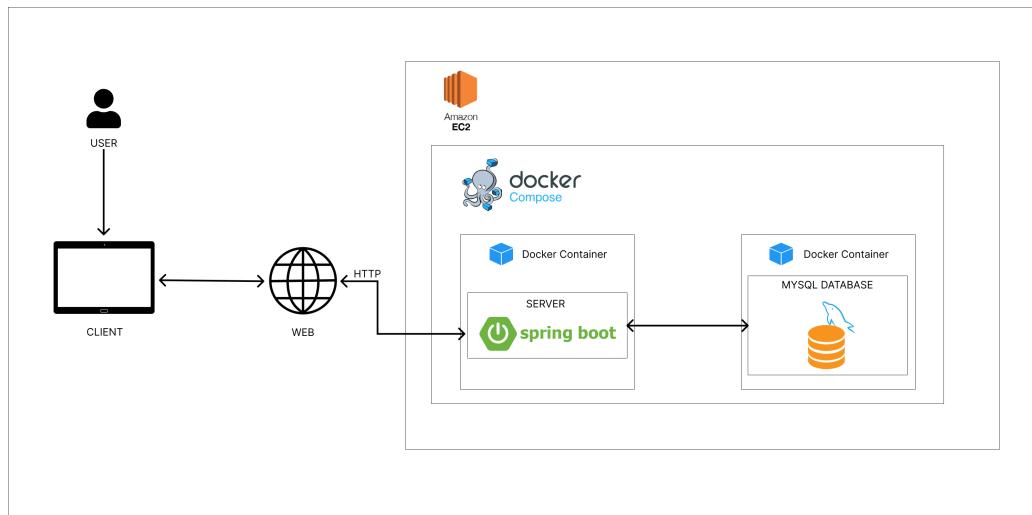


Parte III

Design di Sistema

8 Analisi di Architettura e Sistemi di Design

L'architettura del sistema Ratatouille23 è di tipo Client-Server a **Microservizi**, realizzata con un insieme di tecnologie che descriveremo in questa sezione.



8.1 Client

Il Client del nostro sistema gestionale consiste in un'applicazione Mobile, in particolare per Tablet, eseguibile sul sistema operativo [Android](#).

La struttura del Client si basa su un noto **Design Pattern**, ovvero MVP.

MVP è un Design Pattern che stabilisce relazioni tra tre entità fondamentali :

- **Model**

Definisce i dati da visualizzare e modificare.

- **View**

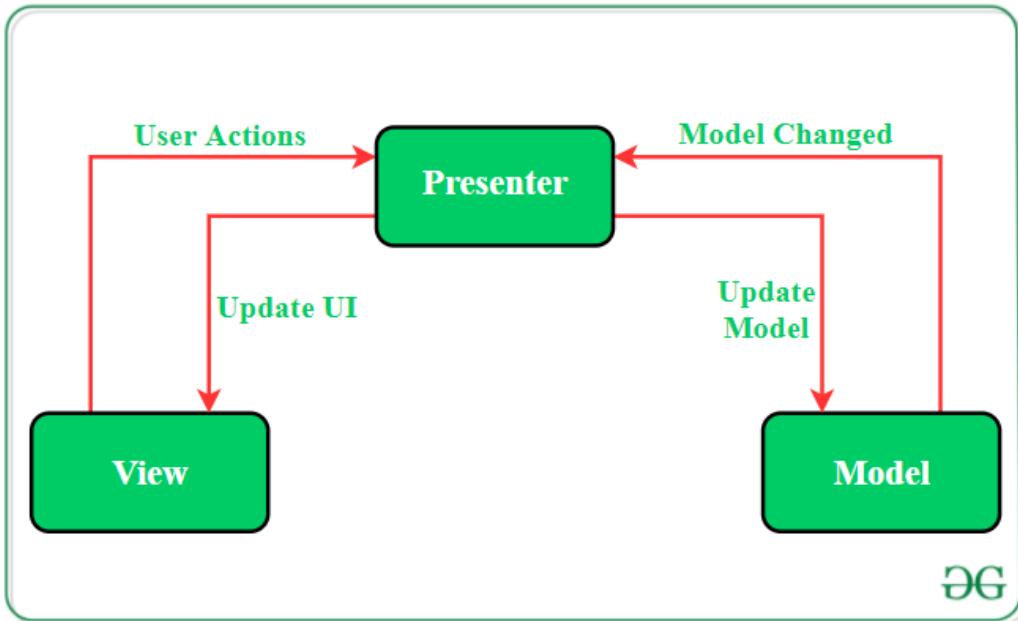
Interfaccia grafica che funge da ponte tra l'utente e l'elaborazione sottostante dei dati.

- **Presenter**

"Man in the Middle", si occupa della gestione dei dati e della modifica dell'Interfaccia Grafica.

Dal Presenter partiranno chiamate a classi ausiliare, generando un flusso che terminerà con una classe "Api", relativa al Model interessato, il cui scopo sarà quello di segnalare, ad un dato URL, una richiesta.

L'immagine che segue schematizza questo pattern ed esplicita in che modo le tre entità in gioco coesistono.



8.1.1 Tecnologie relative al Client

Per quanto riguarda le tecnologie che abbiamo utilizzato per realizzare il Client del nostro sistema, dobbiamo citare :

- **RxJava**

Consiste in una libreria per comporre programmi asincroni e basati su eventi.

Estende il concetto di **Design Pattern Observer**, e supporta sequenze di eventi gestendo in maniera autonoma sincronizzazione, gestione e sicurezza di Thread e Strutture Dati.

- **Retrofit**

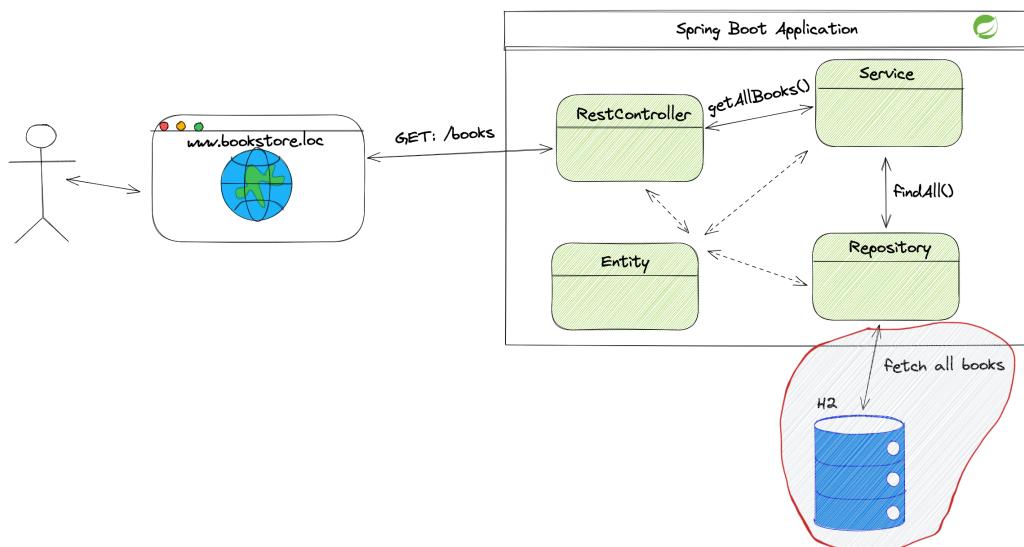
Questo framework supporta l'accesso ai servizi REST, permettendo una comunicazione di tipo HTTP.

8.2 Server

Il Server del nostro sistema sfrutta varie tecnologie che costituiscono la struttura stessa e permettono ad esso di funzionare correttamente.

Per sviluppare il Server ci siamo appoggiati a SpringBoot, un servizio utile alla costruzione di Applicazioni Web e, seguendo la documentazione ufficiale, siamo riusciti ad ottenere una struttura consistente, scalabile ed estremamente esplicativa.

Questa struttura è rappresentata nella figura in basso.



Come possiamo vedere dalla figura, quando viene effettuata una richiesta ad un dato URL, viene costruito un flusso di operazioni che giunge fino al DataBase, permette il recupero dei dati e torna allo stesso URL.

Vediamo nel dettaglio il "flow" del nostro Server :

- **Rest Controller**

Controller incaricati della gestione delle richieste HTTP.

Il compito del Controller è invocare il metodo del Service specifico per la richiesta HTTP rilevata.

Una volta recuperati i dati tramite il Service, convertirà gli oggetti di tipo Model in oggetti di tipo ModelDTO, in modo tale da poter manipolare e filtrare i dati trasmessi.

- **Service**

Classe intermedia, contiene metodi che richiamano metodi delle Repository.

- **Repository**

Classe che si occupa di operazioni CRUD sul DataBase.

Grazie all'utilizzo di JPA offre la possibilità di astrarre completamente dal DataBase tramite metodi composti solo da signature che, automaticamente, inviano la query giusta alla base di dati.

8.2.1 Tecnologie relative al Server

Per quanto riguarda le tecnologie che abbiamo utilizzato per realizzare il Server del nostro sistema, dobbiamo citare :

- **Spring Boot**

Strumento utile alla costruzione di applicazioni Web a microservizi, offrendo una configurazione automatica di base e la possibilità di ottenere un'applicazione autonoma, integrando il server web Tomcat.

- **Spring JPA**

Incluso in Spring Boot, ne abbiamo usufruito in prima istanza per costruire il mapping tra Classi Java e DataBase, e successivamente per effettuare operazioni CRUD sul DataBase.

- **Docker**

Strumento chiave per la containerizzazione del Server e del DataBase, utile per costruire applicazioni indipendenti, modulari e facilmente scalabili.

- **AWS EC2**

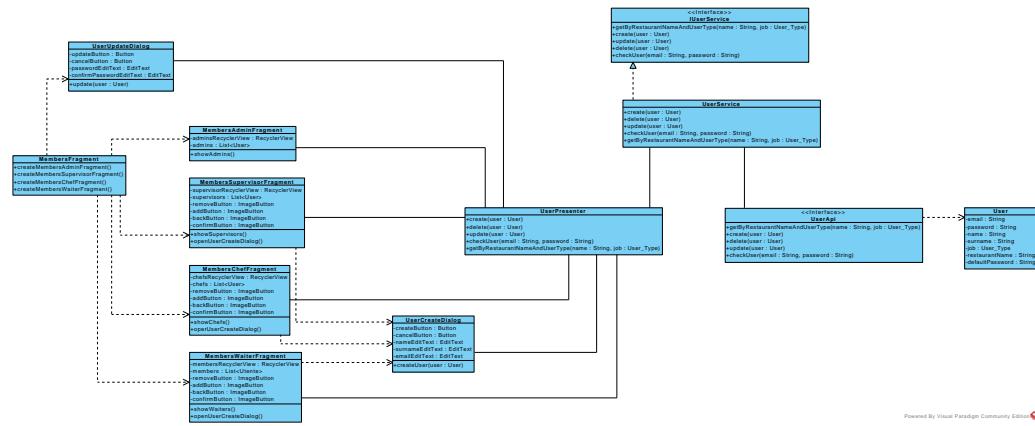
Servizio Amazon che offre servizi di Virtual Machine per le proprie applicazioni web.

9 Diagrammi di Design

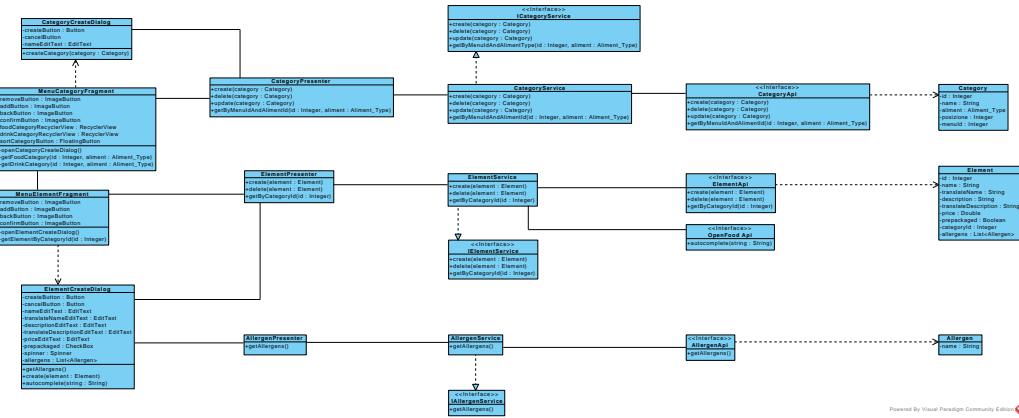
Per la realizzazione dei vari Diagrammi di Analisi è stato usato il tool [Visual Paradigm](#).

9.1 Class Diagram di Design

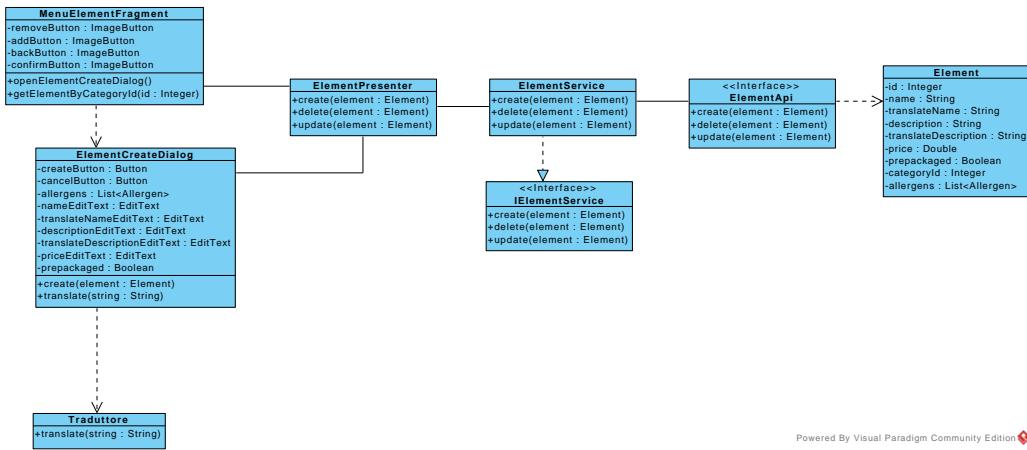
9.1.1 Punto 1 - Creazione Utente per Dipendenti



9.1.2 Punto 3 - Personalizzazione del Menù

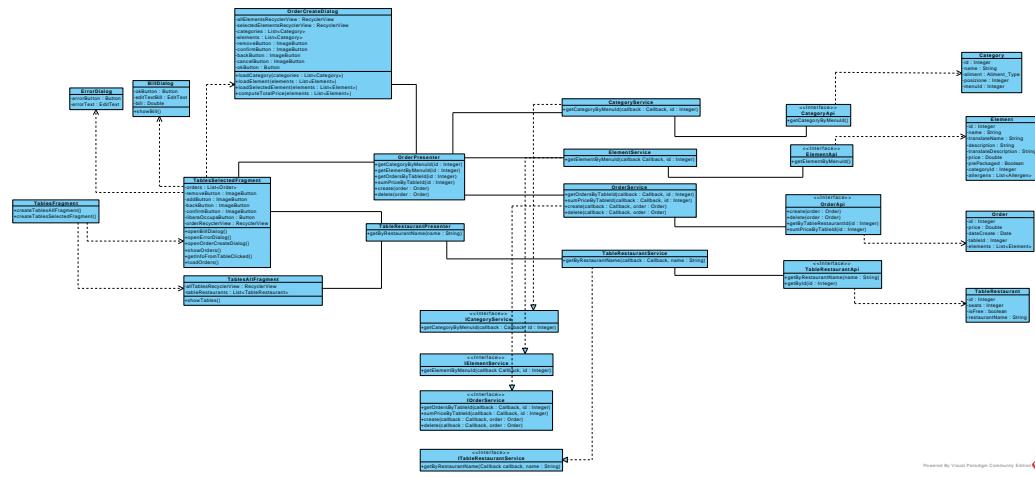


9.1.3 Punto 5 - Specifica degli Elementi in una Seconda Lingua



Powered By Visual Paradigm Community Edition

9.1.4 Punto 6 - Registrazione di Ordinazioni

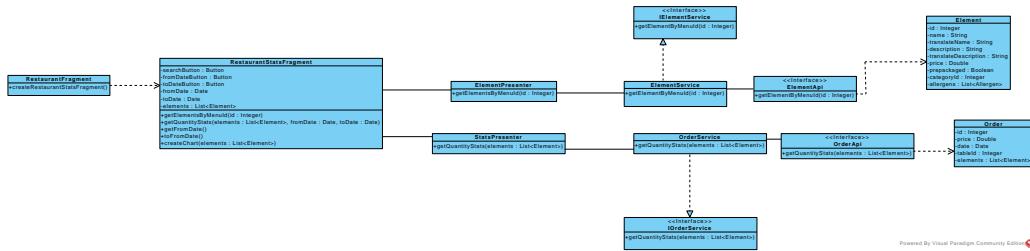


9.1.5 Punto 4 - Stampa del QR Code



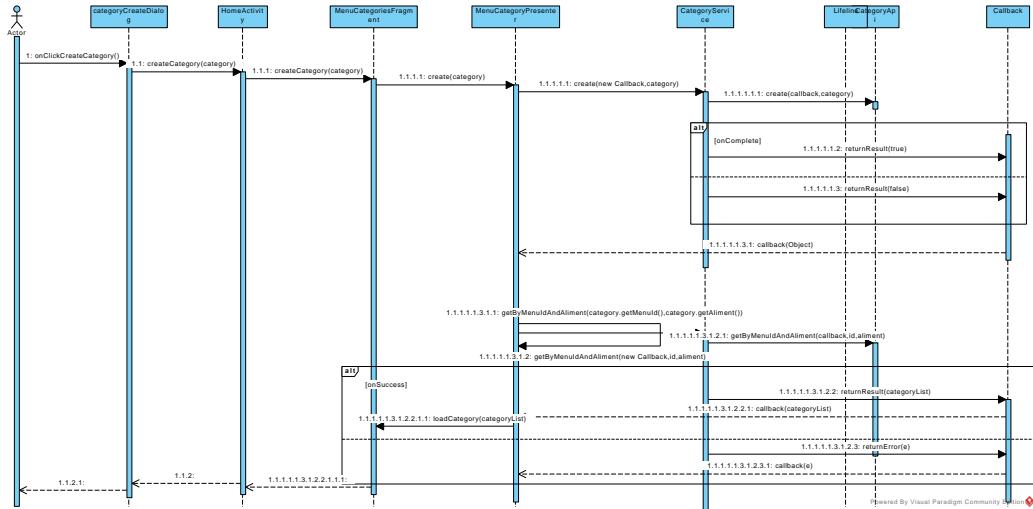
Powered By Visual Paradigm Community Edition

9.1.6 Punto 14 - Statistiche

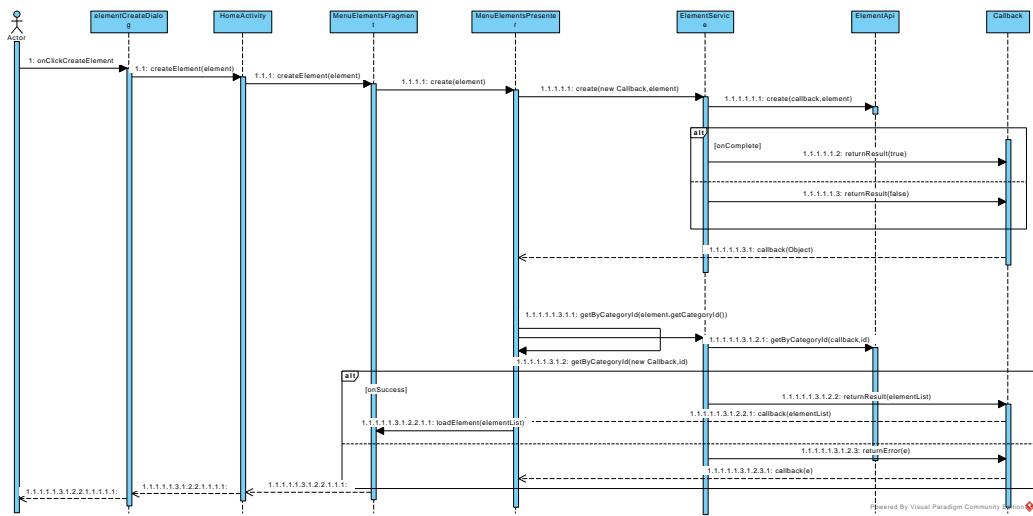


9.2 Sequence Diagram di Design

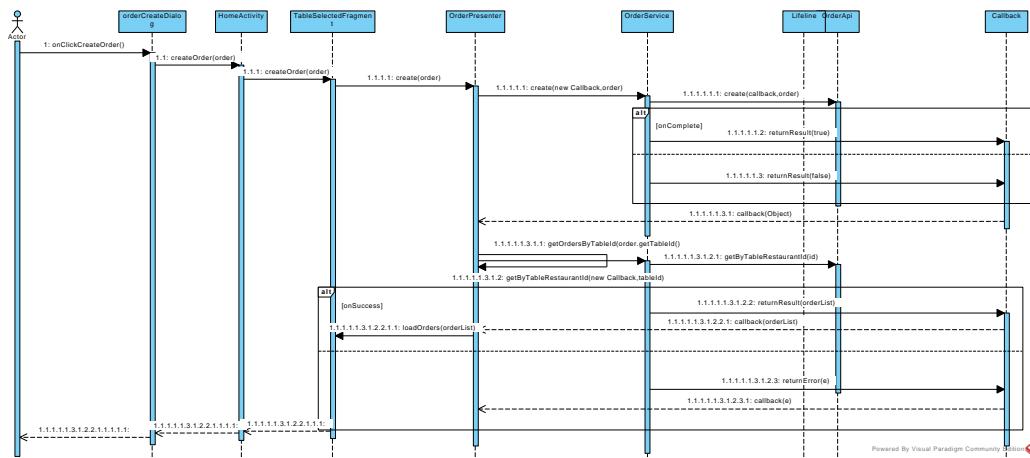
9.2.1 Punto 3.1 - Personalizzazione del Menù - Categorie



9.2.2 Punto 3.2 - Personalizzazione del Menù - Elementi



9.2.3 Punto 6 - Registrazione di Ordinazioni



Parte IV

Testing e Valutazione Usabilità

10 Testing

Per quanto riguarda la fase di Testing del sistema Ratatouille23, abbiamo scelto di testare i seguenti due metodi, con strategia **Black Box** di tipo **WECT** (Weak Equivalence Class Testing).

- **getByIdAndAliment(Integer id, Aliment_Type aliment)**
Metodo della classe CategoryController, recupera dal DB una Collection di Categorie appartenenti ad un Menù ed aventi l'attributo Aliment uguale a quello passato come parametro.
- **getByEmailAndPassword(String email, String password)**
Metodo della classe UserController, recupera dal DB l'utente i cui dati necessari all'autenticazione corrispondono a quelli passati come parametri.

10.1 Descrizione delle Strategie Adottate per la Progettazione dei Test

Dopo aver scelto i metodi da testare, abbiamo identificato le Classi di Equivalenza per ogni parametro di ogni test.

10.1.1 getMenuItemAndAliment(Integer id, Aliment_Type aliment)

Le classi d'equivalenza identificate per questo metodo sono :

- **Integer : id**
 - CE1 : [MinInt,0)
 - CE2 : [0]
 - CE3 : (0,MaxInt]
- **Aliment_Type : aliment**
 - CE4 : "food"
 - CE5 : "drink"
 - CE6 : "non valido"

10.1.2 getByEmailAndPassword(String email, String password)

Le classi d'equivalenza identificate per questo metodo sono :

- **String : email**
 - CE1 : "valida"
 - CE2 : " "
- **String : password**
 - CE3 : "valida"
 - CE4 : " "

10.2 Codice xUnit

Proponiamo dunque il codice con cui abbiamo effettuato la Suite di Testing.

10.2.1 getByMenuIdAndAliment(Integer id, Aliment_Type aliment)

```
1  @Test
2  void testCategory_LegalId_LegalAliment_NotEmptyList () {
3
4      List<CategoryDTO> expected = new ArrayList<>();
5
6      expected.add(
7          new CategoryDTO(7,1,"antipasti",Aliment_Type.valueOf("food"),0));
8
9      expected.add(
10         new CategoryDTO(3,1,"contorni",Aliment_Type.valueOf("food"),1));
11
12     expected.add(
13         new CategoryDTO(1,1,"primi",Aliment_Type.valueOf("food"),2));
14
15     expected.add(
16         new CategoryDTO(2,1,"secondi",Aliment_Type.valueOf("food"),3));
17
18     expected.add(
19         new CategoryDTO(9,1,"pizze",Aliment_Type.valueOf("food"),4));
20
21     List<CategoryDTO> result =
22         categoryController.getByMenuIdAndAliment(1, Aliment_Type.valueOf("food"));
23
24     if(expected.size() != result.size())
25
26         fail();
27
28     else{
29
```

```
30     for(int i = 0; i<expected.size(); i++){  
31  
32         assertEquals(  
33             expected.get(i).getId(), result.get(i).getId());  
34  
35         assertEquals(  
36             expected.get(i).getMenuId(),result.get(i).getMenuId());  
37  
38         assertEquals(  
39             expected.get(i).getName(),result.get(i).getName());  
40  
41         assertEquals(  
42             expected.get(i).getAliment(),result.get(i).getAliment());  
43  
44         assertEquals(  
45             expected.get(i).getPosizione(),result.get(i).getPosizione());  
46  
47     }  
48 }
```

```
1  @Test
2  void testCategory_LegalId_LegalAliment_EmptyList() {
3
4      List<CategoryDTO> expected = new ArrayList<>();
5
6      List<CategoryDTO> result =
7          categoryController.getByMenuIdAndAliment(-1,Aliment_Type.valueOf("food"));
8
9      if(expected.size() != result.size())
10
11          fail();
12
13    else{
14
15        for(int i = 0; i<expected.size(); i++){
16
17            assertEquals(
18                expected.get(i).getId(), result.get(i).getId());
19
20            assertEquals(
21                expected.get(i).getMenuId(),result.get(i).getMenuId());
22
23            assertEquals(
24                expected.get(i).getName(),result.get(i).getName());
25
26            assertEquals(
27                expected.get(i).getAliment(),result.get(i).getAliment());
28
29            assertEquals(
30                expected.get(i).getPosizione(),result.get(i).getPosizione());
31
32        }
33    }
34 }
```

```
1  @Test
2  void testCategory_LegalId_IllegalAliment() {
3
4      assertThrows(IllegalArgumentException.class,
5          () -> categoryController.getMenuItemAndAliment(0, Aliment_Type.valueOf("")));
6
7 }
```

10.2.2 getByEmailAndPassword(String email, String password)

```
1  @Test
2  void testUser_IllegalEmail_LegalPassword() {
3
4      assertThrows(ResponseStatusException.class,
5          () -> userController.getByEmailAndPassword("", "ok"));
6
7 }
```

```
1  @Test
2  void testUser_LegalEmail_IllegalPassword() {
3
4      assertThrows(ResponseStatusException.class,
5          () -> userController.getByEmailAndPassword("admin", ""));
6
7 }
```

```
1  @Test
2  void testUser_LegalEmail_LegalPassword_NotEmptyUser() {
3
4      UserDTO expected =
5          new UserDTO("admin", "rest", "ok", "admin", "admin", User_Type.valueOf("admin"));
6
7      UserDTO result =
8          userController.getByEmailAndPassword("admin", "ok");
9
10     assertEquals(expected.getEmail(), result.getEmail());
11
12     assertEquals(expected.getPwd(), result.getPwd());
13
14     assertEquals(expected.getName(), result.getName());
15
16     assertEquals(expected.getSurname(), result.getSurname());
17
18     assertEquals(expected.getJob(), result.getJob());
19
20     assertEquals(expected.getRestaurantName(), result.getRestaurantName());
21
22 }
```

```
1  @Test
2  void testUser_LegalEmail_LegalPassword_EmptyUser(){
3
4      assertThrows(ResponseStatusException.class,
5          () -> userController.getByEmailAndPassword("notExist", "notExist"));
6
7 }
```

11 Valutazione Usabilità sul Campo

Nella fase finale di questo documento, riportiamo il materiale che riguarda la valutazione dell'Usabilità sul campo.

Riportiamo qui di seguito due valutazioni differenti :

- **Test Sommativi**

Test effettuati col fine di valutare in modo sistematico pregi e difetti del prodotto ultimato.

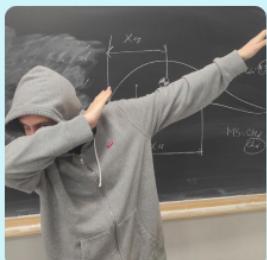
- **Analisi File di Log**

11.1 Test Sommativi

Per questo tipo di Test ci è sembrato utile ed interessante riassegnare lo stesso insieme di Task con cui avevamo valutato l'Usabilità a Priori, agli stessi Esaminati.

11.1.1 Esaminati

Giorgio



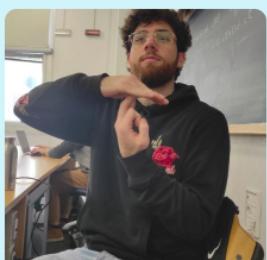
Età : 24

Lavoro : Architetto

Esperienze nel mondo della Ristorazione : Nessuna

Competenze Informatiche : Buone

Renato



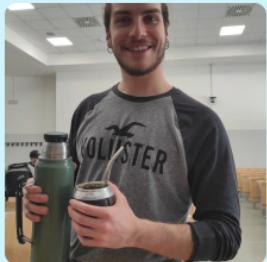
Età : 21

Lavoro : Studente di Informatica

Esperienze nel mondo della Ristorazione : Barista
part-time

Competenze Informatiche : Ottime

Dario



Età : 22

Lavoro : Studente di Informatica

Esperienze nel mondo della Ristorazione : Nessuna

Competenze Informatiche : Ottime

Marco



Età : 25

Lavoro : Ingegnere Meccanico

Esperienze nel mondo della Ristorazione : Cameriere full-time

Competenze Informatiche : Buone

11.1.2 Task stabiliti

I Task assegnati, dunque, sono i seguenti :

- Effettuare il login (con credenziali note) e visualizzare statistiche relative ad un lasso di tempo che va dal 01/01/2023 al 31/03/2023.
- Effettuare il login (con credenziali note) e creare una categoria denominata "Pre Dessert".
- Effettuare il login (con credenziali note) e creare un'ordinazione per il Tavolo 3 composta da 2 Margherite e 2 Peroni.
Visualizzare successivamente il totale dell'ordinazione.
- Effettuare il login (con credenziali note) ed eliminare il cameriere chiamato "Kvicha".

11.1.3 Sistema di Valutazione

Abbiamo deciso di far attribuire ad ogni esaminato un voto compreso tra 1 e 10 per ognuno dei criteri fondamentali per la valutazione dell'usabilità precedente- mente descritti.

Dopo aver quindi ottenuto un voto per ogni criterio da ogni esaminato, avremo il risultato complessivo del Test effettuando una media aritmetica.

11.1.4 Voti Parziali

Dopo aver spiegato per grandi linee agli Esaminati i cinque criteri con cui abbiamo stabilito di valutare l'Usabilità del prodotto, abbiamo chiesto ai candidati di assegnare, per ogni test, un voto ad ognuno di questi parametri, per poi ottenere un voto complessivo per ognuno dei Task assegnati.

DARIO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	9	9	8	10
Efficienza	10	9	9	8
Memorabilità	9	9	8	9
Errori	9	10	9	10
Soddisfazione	8	9	10	10

GIORGIO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	10	10	9	9
Efficienza	9	10	9	8
Memorabilità	9	9	10	9
Errori	9	8	9	9
Soddisfazione	10	10	9	9

MARCO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	10	8	9	10
Efficienza	9	8	10	10
Memorabilità	8	9	9	8
Errori	9	8	9	10
Soddisfazione	10	9	9	10

RENATO	Test 1	Test 2	Test 3	Test 4
Apprendibilità	9	10	9	9
Efficienza	8	9	9	10
Memorabilità	10	9	10	9
Errori	9	8	9	10
Soddisfazione	9	10	8	9

11.1.5 Risultati dei Test

	Test 1	Test 2	Test 3	Test 4
Dario	9	9,2	8,8	9,4
Giorgio	9,4	9,4	9,2	8,8
Marco	9,2	8,4	9,2	9,6
Renato	9	9,2	9	9,4

	Test 1	Test 2	Test 3	Test 4
Apprendibilità	9,5	9,25	8,75	9,5
Efficienza	9	9	9,25	9
Memorabilità	9	9	9,25	8,75
Errori	9	8,5	9	9,75
Soddisfazione	9,25	9,5	9	9,5

11.1.6 Commenti degli Esaminati

Dario	L'applicazione è molto intuitiva, la reputo abbastanza efficiente. Un bug ha sfalsato le interfacce che si scambiano fra loro nel momento in cui va ad eliminare un utente.
Giorgio	Non è stato un problema per me svolgere tutti i compiti assegnati, sono soddisfatto.
Marco	Mi piace molto esteticamente, anche se ho riscontrato un bug che aggiornava le statistiche dopo due click e non uno solo.
Renato	Penso sia un'applicazione buona, utilizzabile sia da utenti esperti sia da novizi.

11.1.7 Conclusioni

Abbiamo ottenuto dunque un **Valore di Usabilità** pari a **9.13/10**.

Abbiamo riscontrato qualche bug grafico grazie alle segnalazioni degli Esaminati, che abbiamo prontamente risolto.

Confrontandoci con coloro a cui avevamo assegnato i Task, abbiamo ricevuto un riscontro positivo circa l'implementazione del sistema, reputata estremamente vicina alla prototipazione proposta per la valutazione dell'Usabilità a Priori.

11.2 Analisi

Per monitorare l'utilizzo del sistema abbiamo utilizzato il Tool **Fire-base Analytics**.

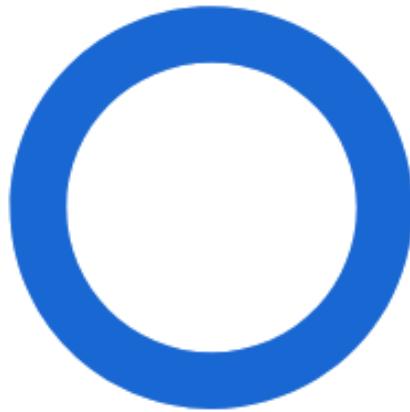
Tramite Analytics abbiamo avuto la possibilità di tenere traccia degli utenti che hanno utilizzato il software, delle schermate visualizzate e del tempo di utilizzo.

Riportiamo di seguito dunque i dati rilevati.

11.2.1 Utenti rilevati



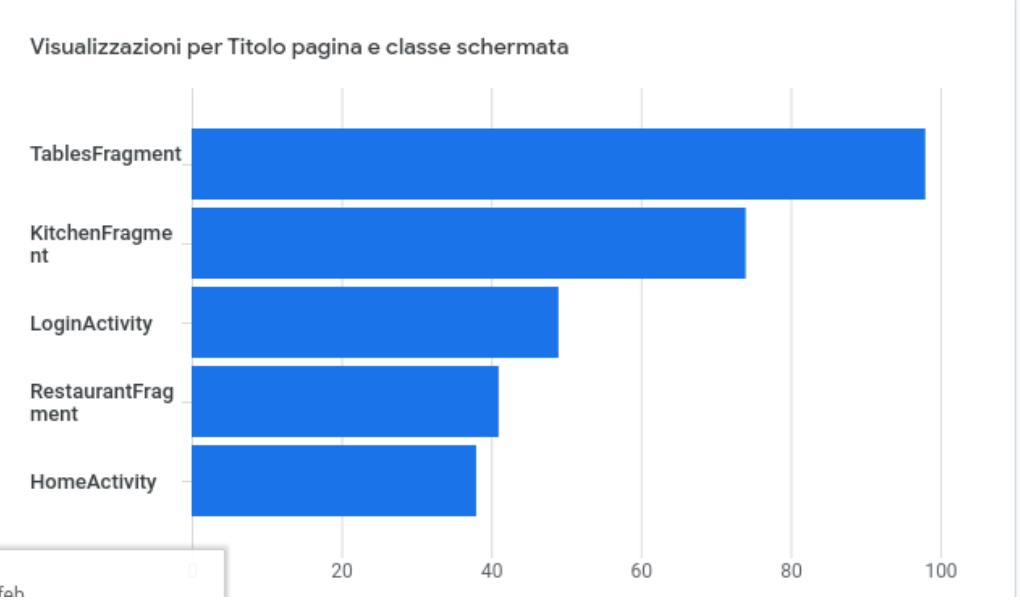
Conversioni ▾ per Piattaforma



● ANDROID
100,0%

[Visualizza i dettagli sulla tecnologia →](#)

11.2.2 Schermate Visualizzate



11.2.3 Tempo di Utilizzo

	Titolo pagina e...asse schermata	+	↓ Visualizzazioni	Utenti	Visualizzazioni per utente	Durata media del coinvolgimento	Conteggio eventi	Conversioni	Entrate totali
			380 100% del totale	2 100% del totale	190,00 Uguale alla media	56 m 16 s Uguale alla media	422 100% del totale	2,00 100% del totale	0,00 \$
1	TablesFragment		98	1	98,00	0 m 44 s	100	0,00	0,00 \$
2	KitchenFragment		74	1	74,00	0 m 12 s	74	0,00	0,00 \$
3	LoginActivity		49	2	24,50	21 m 25 s	57	0,00	0,00 \$
4	RestaurantFragment		41	1	41,00	32 m 48 s	44	0,00	0,00 \$
5	HomeActivity		38	2	19,00	16 m 45 s	54	0,00	0,00 \$
6	MainActivity		32	2	16,00	0 m 00 s	32	0,00	0,00 \$
7	MembersFragment		24	1	24,00	0 m 57 s	24	0,00	0,00 \$
8	MenuFragment		24	1	24,00	1 m 31 s	25	0,00	0,00 \$

