



# UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

## *Ratatouille23*

Cognome	Nome	Matricola	Indirizzo e-mail istituzionale
Cesarano	Agostino	N86003587	<a href="mailto:ag.cesarano@studenti.unina.it">ag.cesarano@studenti.unina.it</a>
Fortino	Alessandro	N86003884	<a href="mailto:ale.fortino@studenti.unina.it">ale.fortino@studenti.unina.it</a>

# Indice

Introduzione .....	5
Analisi dei requisiti .....	5
Modellazione dei casi d'uso .....	8
1# Login System Use Case .....	9
2# Order System Use Case .....	10
3# Admin e Supervisore System Use Case .....	11
Descrizione testuale dei casi d'uso .....	11
Mock-up dell'interfaccia .....	22
Valutazione dell'usabilità.....	33
<b>Tecniche adoperate</b> .....	34
<b>Tabelle di valutazione</b> .....	43
Tecnica di valutazione.....	46
Risultati dei test .....	46

Conclusioni.....	48
<b>Glossario</b> .....	49
Specifica dei requisiti.....	52
Classi, oggetti e relazioni di analisi. ....	52
Diagrammi di sequenza di analisi per due casi d'uso .....	53
Prototipazione funzionale via statechart dell'interfaccia grafica	55
Design del sistema .....	57
Front-end .....	57
Back-end.....	58
Database.....	59
Come funziona?.....	59
Motivazioni delle scelte.....	61
Diagramma delle classi di design.....	64
Classi parser o serializer (Modelli).....	64

Classi viste.....	67
Diagrammi di sequenza di design per due casi d'uso .....	68
Sequence “Aggiungi elemento al menu”.....	68
Sequence “Aggiungi dipendente” .....	69
Testing e valutazione sul campo dell'usabilità .....	70
Unit Testing .....	70
Valutazione dell'usabilità sul campo.....	79
<i>Metodo euristico</i> .....	88

# Introduzione

La gestione dei ristoranti richiede un'organizzazione accurata e un'efficienza nello svolgimento delle operazioni quotidiane, per garantire un'esperienza ottimale ai clienti.

Ratatouille è un'applicazione sviluppata per semplificare il lavoro dei camerieri e degli amministratori, consentendo loro di inviare ordini alla cucina, gestire i dipendenti e il menu del ristorante. Grazie all'integrazione con un generatore di QR code, Ratatouille consente inoltre di visualizzare il menu in diverse lingue tramite un sito web. Il supervisore ha accesso solo alla gestione del menu. In questa relazione tecnica verranno analizzate le *tecnologie utilizzate per lo sviluppo dell'applicazione, l'architettura del sistema* e le *principali funzionalità offerte*, nonché le *metodologie utilizzate per testare l'applicazione e garantire la sua sicurezza*.

## Analisi dei requisiti

Ratatouille23 è un sistema finalizzato alla gestione e all'operatività di attività di ristorazione. L'applicazione offre numerose funzionalità per semplificare la gestione del ristorante e migliorare l'esperienza

dei clienti. Di seguito analizziamo in dettaglio i requisiti principali dell'app:

**Gestione utenti:** uno dei requisiti principali dell'app è la possibilità per l'amministratore di creare utenze per i propri dipendenti, specificando il loro ruolo all'interno del ristorante (*sala, cucina, supervisori*). Questo consente di limitare l'accesso a determinate funzionalità in base al ruolo dell'utente. Inoltre, ogni utente deve reimpostare la password al primo accesso per motivi di sicurezza.

**Personalizzazione menù:** l'app consente all'amministratore o ai supervisori di personalizzare il menù del ristorante. In particolare, è *possibile creare e/o eliminare elementi dal menu*, organizzarli in categorie personalizzabili e definire l'ordine con cui gli elementi compaiono nel menù. Inoltre, è richiesto l'auto completamento di alcuni prodotti utilizzando open data come quelli disponibili in <https://it.openfoodfacts.org/data>. *Questo consente di creare un menù coerente con il tema del ristorante e di tenere traccia degli ingredienti utilizzati.*

**QR Code personalizzato:** un altro requisito dell'app è la possibilità per l'amministratore di stampare un *QR Code personalizzato* che rimanda ad un indirizzo web dove i clienti possono visualizzare il menu. Questo consente di rendere l'esperienza dei clienti più interattiva e di semplificare la scelta del menù.

**Gestione multilingue:** un requisito aggiuntivo per i ristoranti che operano in località turistiche è la possibilità di stampare un *QR Code del menu in una seconda lingua*. Questo consente di migliorare l'esperienza dei clienti stranieri e di rendere il ristorante più accogliente.

**Statistiche sulla produttività del personale:** un altro requisito dell'app è la possibilità per l'amministratore di visualizzare statistiche dettagliate sulla produttività del personale addetto alla sala. In particolare, è possibile visualizzare quanti ordini ciascun addetto alla sala ha registrato e il valore cumulativo di quegli ordini in un lasso di tempo personalizzabile. *Questo consente*

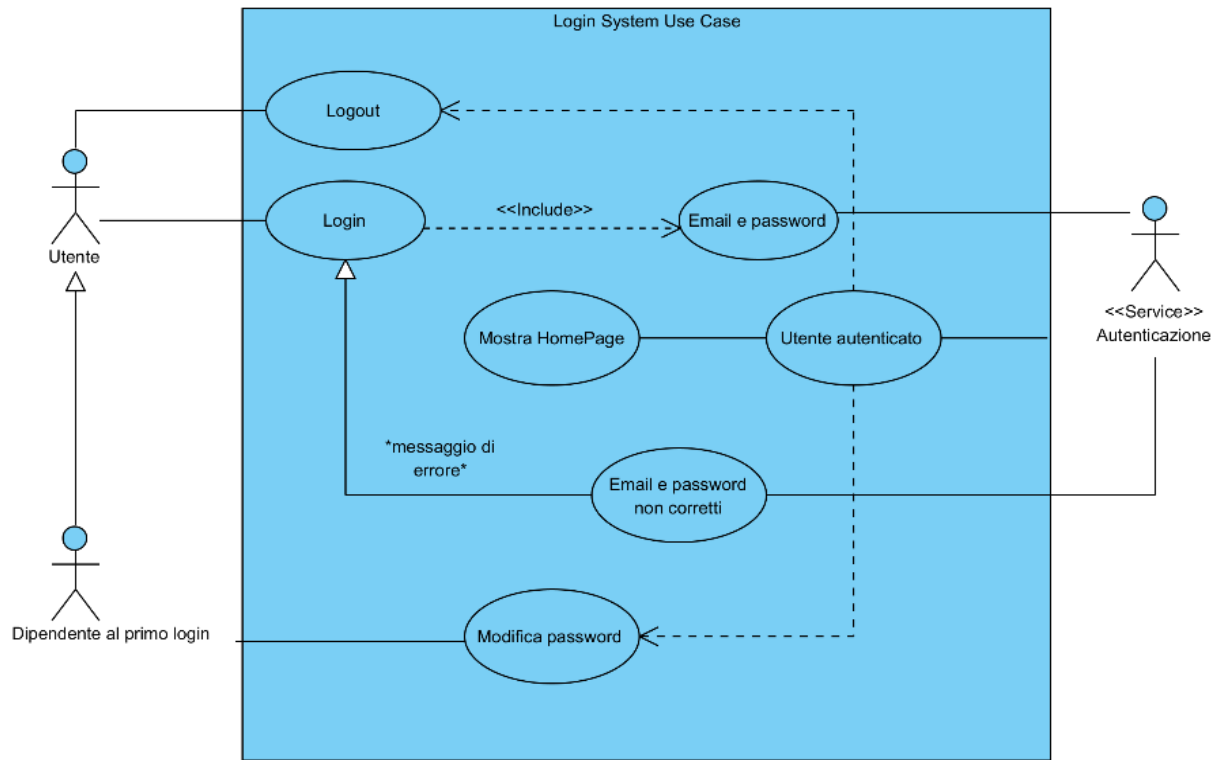
*all'amministratore di monitorare l'efficienza del personale e di identificare eventuali problemi.*

## **Modellazione dei casi d'uso**

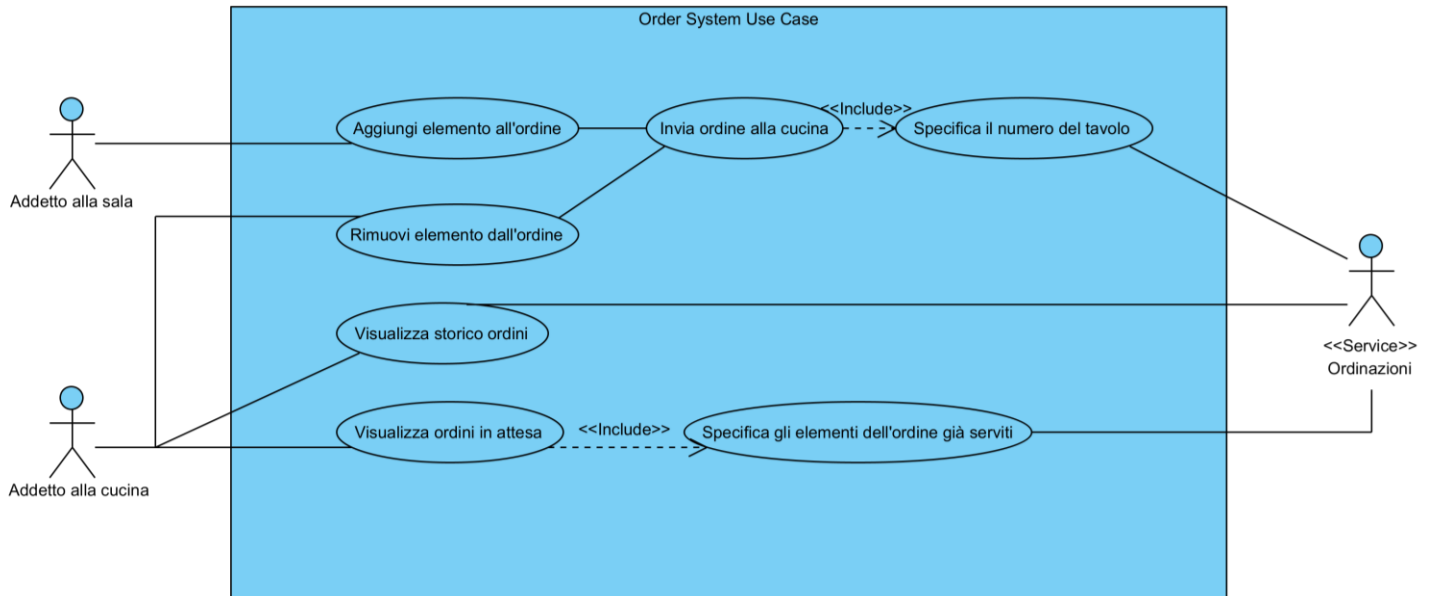
L'analisi dei casi d'uso tramite *use case* rappresenta un metodo di modellazione utile per comprendere le funzionalità offerte dall'applicazione Ratatouille e le relazioni tra gli utenti del sistema. Verranno analizzati i principali casi d'uso che interessano i diversi tipi di utenti (cameriere, amministratore, supervisore) e le loro interazioni con l'applicazione. L'obiettivo è quello di fornire una panoramica completa delle funzionalità dell'applicazione e delle interazioni tra gli utenti e il sistema. Saranno dunque descritti i vari casi d'uso, evidenziando requisiti funzionali e non funzionali, attori, precondizioni e post condizioni, con l'obiettivo di fornire una guida esaustiva per lo sviluppo e l'utilizzo del sistema.



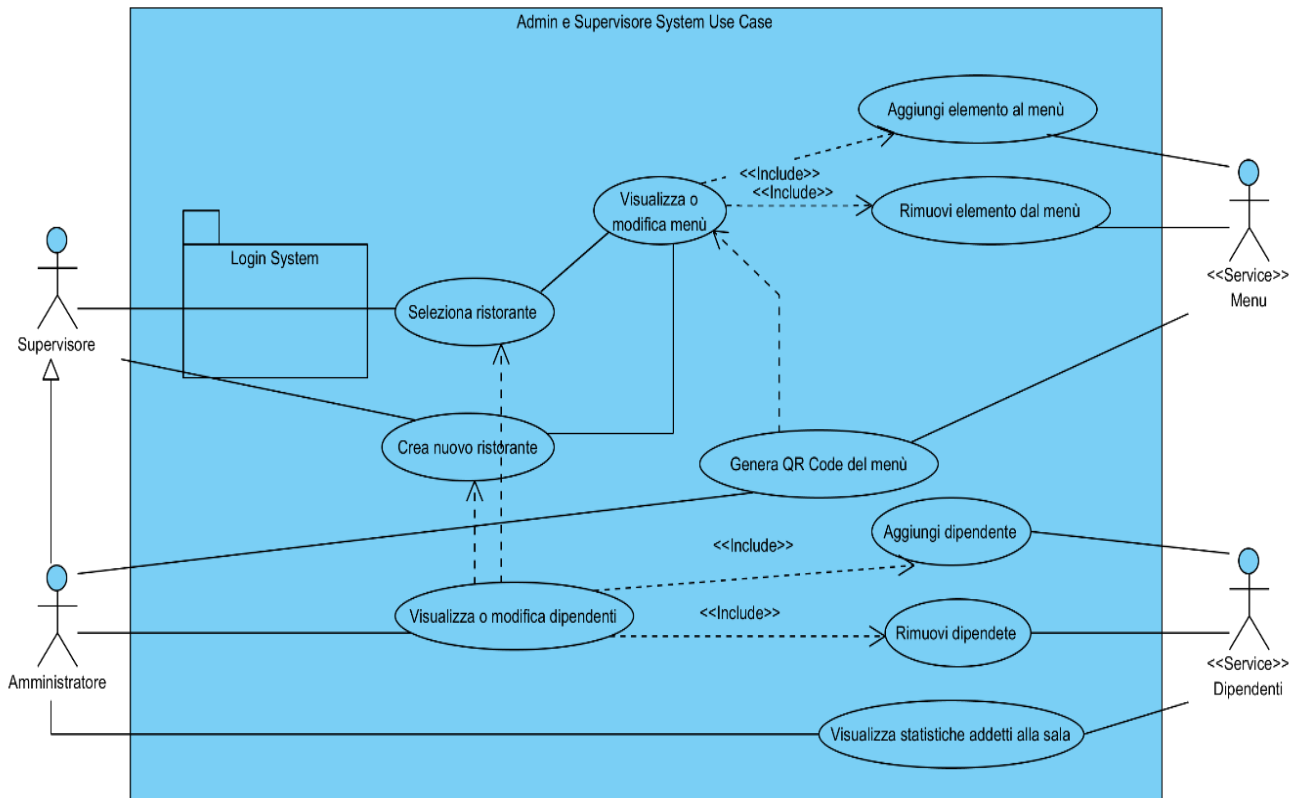
## 1# Login System Use Case



## 2# Order System Use Case



### 3# Admin e Supervisore System Use Case



### Descrizione testuale dei casi d'uso

Le tabelle di Cockburn sono utilizzate per descrivere i requisiti funzionali di un sistema software, in particolare per descrivere le interazioni tra gli utenti e il sistema. Grazie alla loro struttura tabellare, queste tabelle permettono di rappresentare in modo chiaro e strutturato le interazioni tra gli attori e il sistema.

## USE CASE “AGGIUNGI ELEMENTO AL MENU”

<b>USE CASE #2</b>	Aggiungi elemento al menù		
<b>Goal in Context</b>	L’amministratore o il supervisore vuole aggiungere un elemento al menù.		
<b>Preconditions</b>	Il supervisore o l’amministratore deve aver effettuato il login in precedenza.		
<b>Success End Condition</b>	Il supervisore o l’amministratore aggiunge con successo l’elemento al menù.		
<b>Failed End Condition</b>	Il supervisore o l’amministratore ha un problema mentre cerca di aggiungere l’elemento al menù		
<b>Primary Actor</b>	Supervisore e amministratore		
<b>Trigger</b>	Il supervisore o l’amministratore clicca il bottone “+” nel frame dedicato alla gestione del menù		
<b>Description</b>	<b>Step n°</b>	<b>Primary Actor</b>	<b>Sistema</b>

	1	Seleziona la “Categoria” dell’elemento che si vuole aggiungere facendo click su di essa.	
	2		Mostra il frame dedicato all’aggiunta di un nuovo elemento alla categoria del menù
	3	Compila il form per l’aggiunta dell’elemento.	
	3.1	Inserisci un	

		“Nome”	
	3.2	Inserisci un “Costo”	
	3.3	Inserisci uno o più “Allergeni”	
	3.4	Inserisci una “Descrizione”	
	4	Clicca il bottone “Salva”	
	5		Invia i dati al server.
	6		Mostra il frame che indica l’avvenuto salvataggio.

	7		Reindirizza il supervisore o l'amministratore al frame di partenza,
<b>Extension #1</b>  Il sistema non riesce a connettersi al server.	6.a		Mostra il frame che indica l'errore riscontrato.
	7.a		Reindirizza il supervisore o l'amministratore al frame di partenza,
<b>Extension #2</b>  Esiste un elemento con lo stesso nome del nuovo elemento.	6.b		Mostra un popup che indica l'errore riscontrato

<b>Subvariato #1</b>  La categoria non è presente in quelle già aggiunte.	1	Clicca il bottone “+” per aggiungere una nuova categoria	
	2		Mostra il popup con il form per aggiungere la categoria.
	3	Compila il form per l’aggiunta della categoria.	
	3.1	Inserisci il “Nome” della nuova categoria	



	3.2	Clicca il botton “Salva”	
	4		Invia i dati al server.
	5		Mostra un popup che indica l’avvenuto salvataggio.
	6		Chiudi il popup con il form per l’aggiunta della categoria.
<b>Extension #3</b> Esiste una categoria con lo stesso come della nuova categoria	5.a		Mostra un popup che indica l’errore riscontrato.

## USE CASE “AGGIUNGI DIPENDENTE”

<b>USE CASE #2</b>	Aggiungi elemento al menù		
<b>USE CASE #1</b>	L'amministratore aggiunge un dipendente		
<b>Goal in Context</b>	L'amministratore vuole aggiungere un dipendente		
<b>Preconditions</b>	L'amministratore deve aver effettuato il login		
<b>Success End Condition</b>	L'amministratore riesce ad aggiungere un dipendente		
<b>Failed End Condition</b>	L'amministratore non riesce ad aggiungere un dipendente		
<b>Primary Actor</b>	Amministratore		
<b>Trigger</b>	L'amministratore si trova nella pagina di gestione del personale e clicca sul bottone “+” per aggiungere un dipendente		
<b>Description</b>	<b>Step</b>	<b>Primary Actor</b>	<b>Sistema</b>

	<b>n°</b>		
	1	<p>Compila tutti i campi che servono per la registrazione di un nuovo dipendente (nome, cognome, e-mail, password temporanea, ruolo dipendente)</p>	
	2	<p>Preme il pulsante “Salva” in fondo alla schermata</p>	

	3		Mostra la pagina di gestione del personale con il nuovo dipendente aggiunto
<b>Extension #1</b>  (L'amministratore non ha inserito alcun dipendente)	3a		Mostra la pagina di gestione del personale ma non viene mostrata alcuna modifica a quella iniziale
<b>Extension #2</b>  (L'amministratore non ha inserito i	2a	L'amministratore non ha compilato tutti i campi	

dati  correttamente)	3a		Mostra un  messaggio di  errore  all'amministratore  fino a quando non  compila tutti i  campi per la  registrazione  correttamente
<b>Extension #3</b>  (L'amministratore  ha inserito un'e-  mail già  registrata)	3a		Mostra  all'amministratore  un messaggio di  errore avvisandolo  che l'e-mail  inserita è già  registrata e  invitandolo a  inserire un'altra e-

			mail.
<b>Extension #4</b>  (Il sistema non riesce a connettersi al server)	3a		Il sistema mostra all'amministratore un messaggio di errore avvisandolo che non riesce a connettersi al server e invitandolo a riprovare più tardi

### Mock-up dell'interfaccia

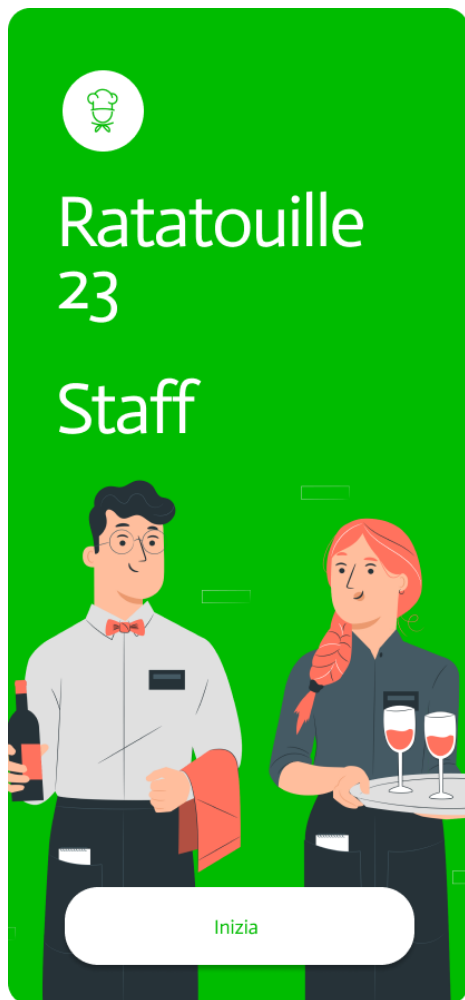
Dopo un'attenta valutazione dei casi d'uso richiesti, si è optato per la scelta di avere due applicazioni indipendenti:

**Ratatouille Staff** che permette ad Addetti alla Sala e Addetti alla Cucina di gestire gli ordini e ideata per *piattaforme mobile*.

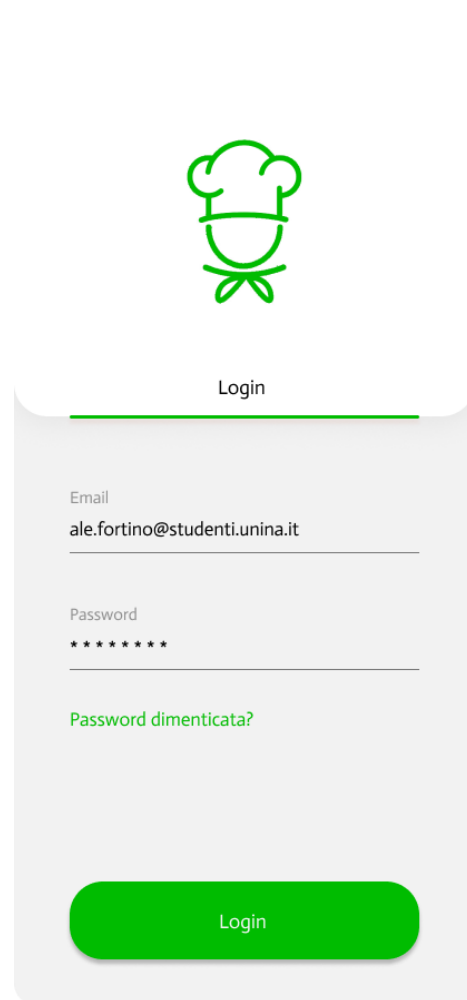
**Ratatouille Admin** che permette ad Amministratori di gestire il menu (in questo caso anche Supervisor), gestire utenti, e controllare le statistiche di produttività dei propri camerieri.

## MOCKUP RATATOUILLE STAFF

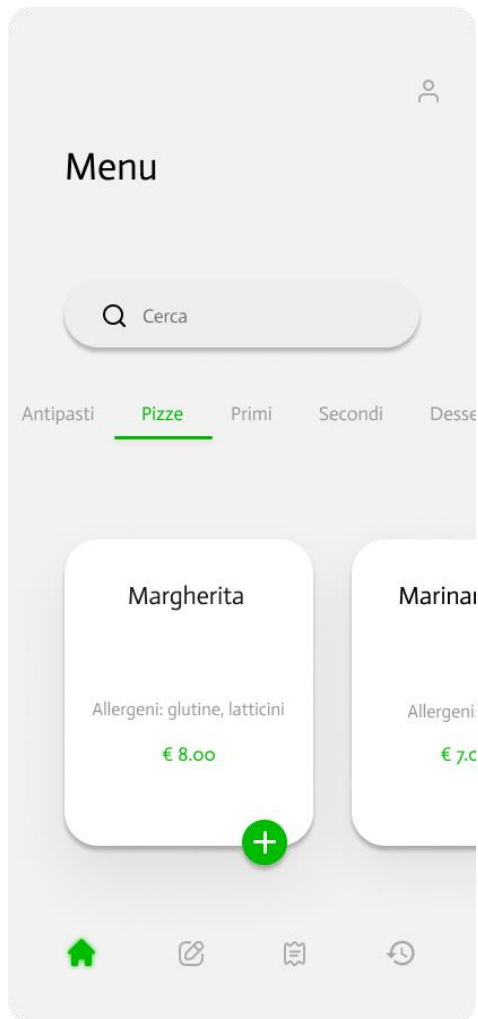
### Pagina di benvenuto



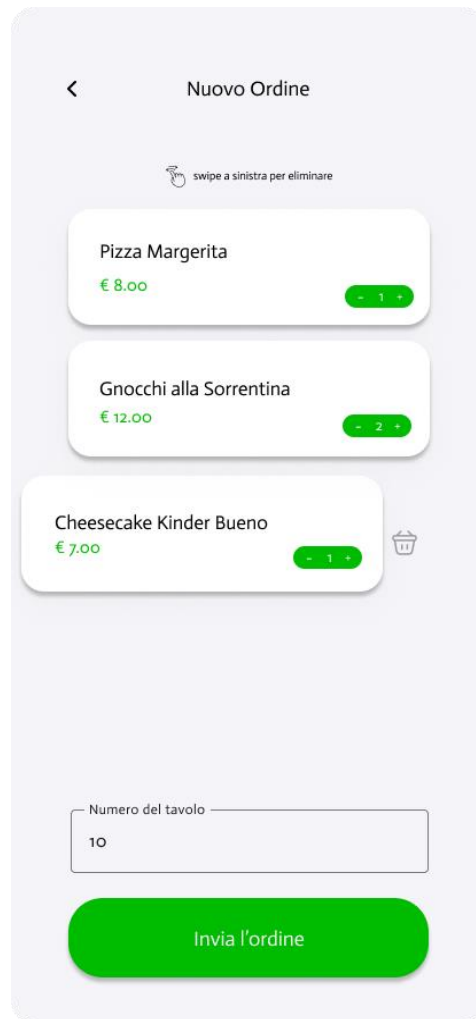
### Login Page



## Home Page

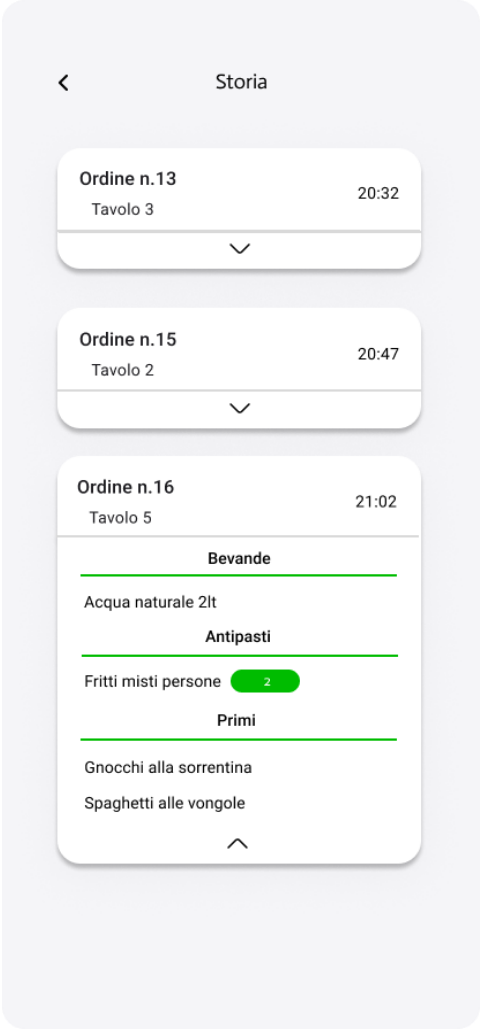


## Pagina Nuovo Ordine

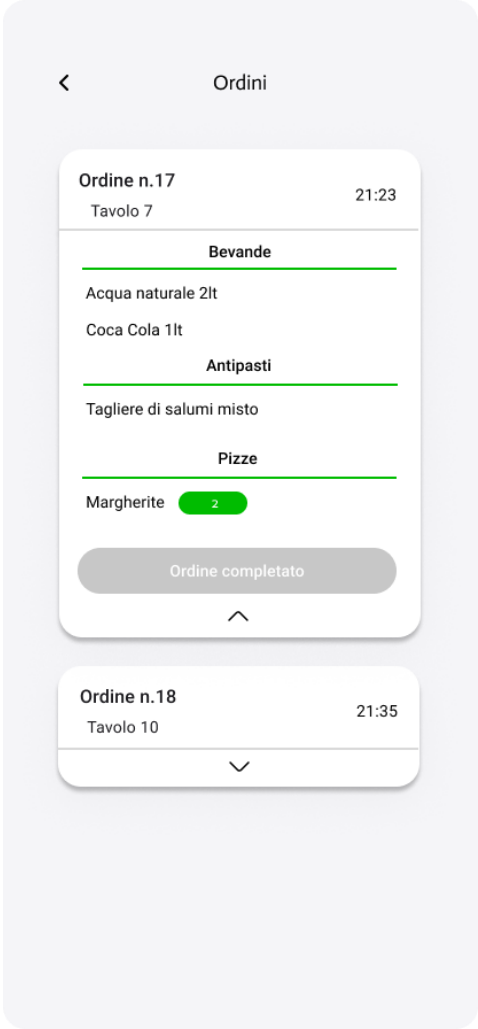





Pagina Storico degli ordini



Pagina Ordini in attesa



## Pagina Profilo utente

 Il mio profilo

Informazioni

**Alessandro Fortino**  
ale.fortino@studenti.unina.it  
Il Castello delle Cerimonie  
[Addetto alla sala](#)

[Modifica password!](#)

Sign-out

## Pagina modifica password

Modifica password

Vecchia password

\*\*\*\*\*

Nuova password

\*\*\*\*\*

Conferma la nuova password

\*\*\*\*\*

Salva

Per godere dei Mockup nel dettaglio e visualizzare la progettazione prototipale dell'interfaccia grafica di seguito sarà allegato il link per i progetti Figma:

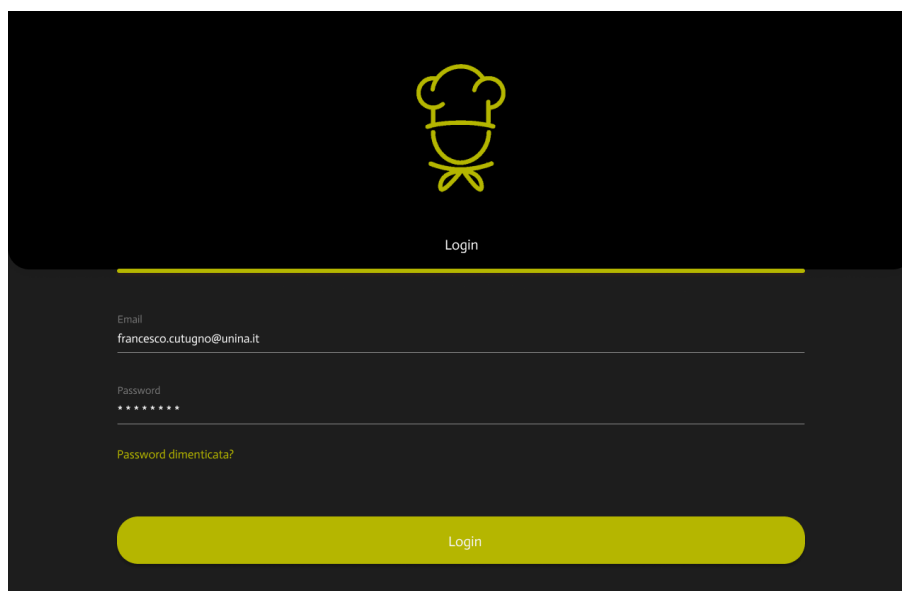
<https://www.figma.com/file/103S60gLcxa9X3drbrB8VA/Ratatuille-Staff?t=AxTQx6KDI0gQau2f-6>

## MOCKUP RATATOUILLE ADMIN

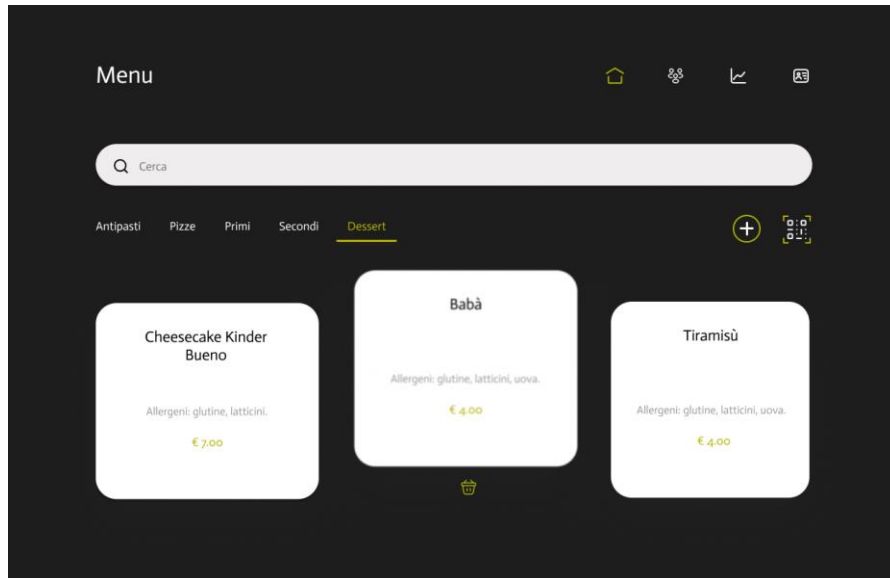
### Pagina di benvenuto



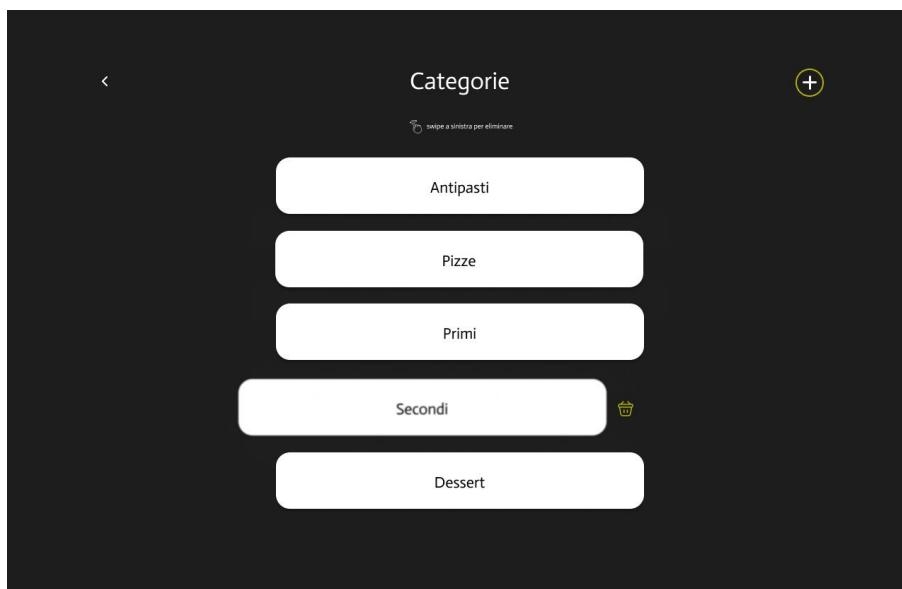
### Login Page



## Home Page Menu



## Pagina Categorie



# Pagina Aggiungi pietanza

<

Aggiungi Primi

Nome

Costo

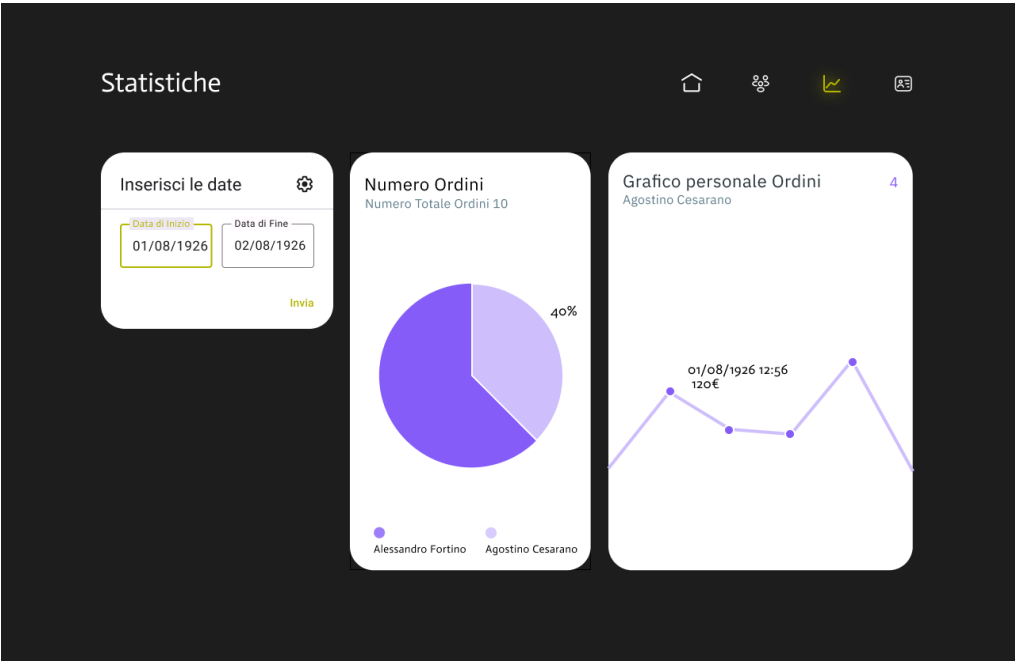
Allergeni

Nessun allergene X

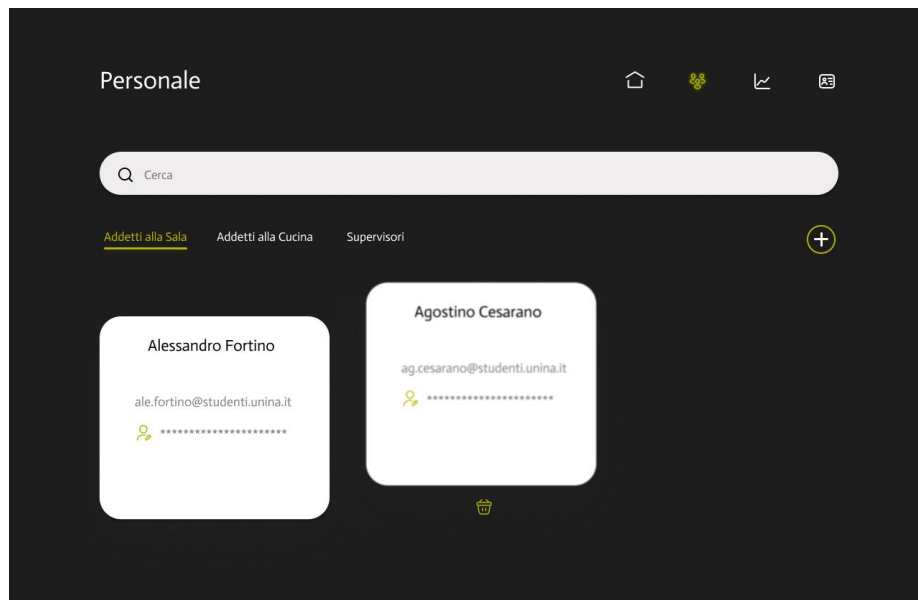
Descrizione

Salva

# Pagina Statistiche



## Menu personale

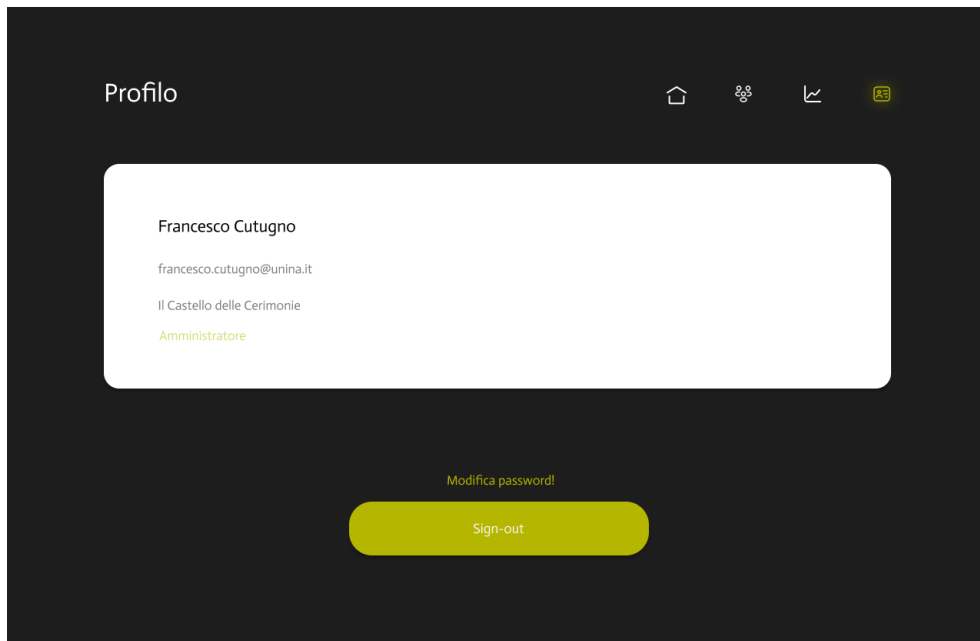


## Pagina aggiungi personale

The screenshot shows a mobile application interface titled 'Aggiungi Personale'. It features a back arrow on the top left. The form contains the following fields and elements:

- Nome e Cognome**: A text input field.
- Email**: A text input field.
- Password temporanea**: A text input field.
- Ruolo**: A dropdown menu with three options: 'Addetto alla Sala' (highlighted in green), 'Addetto alla Cucina', and 'Supervisore'.
- Salva**: A green button at the bottom.

## Pagina Profilo amministratore



Per godere dei Mockup nel dettaglio e visualizzare la progettazione prototipale dell'interfaccia grafica di seguito sarà allegato il link per i progetti Figma:

<https://www.figma.com/file/xYrKVI7LxoeaoAiYvByNZ3/Ratatouille-Admin?t=AxTQx6KDI0gQau2f-6>



## **Valutazione dell'usabilità**

Gli obbiettivi principali dei test di usabilità per Ratatouille sono:

*Valutare la facilità di utilizzo dell'app da parte dei dipendenti, in particolare l'interfaccia grafica e la navigazione tra le funzionalità.*

*Verificare la capacità dell'app di supportare le attività quotidiane dei dipendenti del ristorante, come l'inserimento degli ordini, la gestione del menu e la visualizzazione delle statistiche di produttività.*

*Esaminare l'accessibilità dell'app per le persone con daltonismo o altre disabilità visive, per assicurarsi che l'interfaccia grafica sia user friendly per tutti gli utenti.*

*Raccogliere feedback dagli utenti su eventuali problemi o difficoltà riscontrati durante l'utilizzo dell'app, al fine di migliorarne la usabilità e l'esperienza d'uso complessiva.*

In particolare, per quanto riguarda l'aspetto di interfaccia grafica user friendly per le persone con daltonismo, l'obiettivo sarebbe quello di verificare che il design dell'app sia conforme agli standard

di accessibilità e permetta una chiara distinzione tra i colori utilizzati. Si chiede ai partecipanti del test di usabilità di esaminare la app e fornire un riscontro sulla leggibilità dei contenuti e la facilità di distinguere i diversi elementi dell'interfaccia grafica. In base a queste osservazioni, si potrebbero apportare eventuali modifiche per migliorare la leggibilità e l'accessibilità dell'app per le persone con daltonismo.

## ***Tecniche adoperate***

### *Creazione di mockup dinamici ed interattivi*

Per quanto riguarda la valutazione dell'usabilità a priori sono state adoperate varie tecniche e strumenti che avevamo a disposizione. In particolare, per ottenere una valutazione dell'usabilità migliore abbiamo creato dei prototipi Figma ad alta fedeltà sull'aspetto e sulle funzionalità su quella che sarà la nostra app finale così da avere una valutazione a priori quanto più vicina a quella a posteriori in beta testing.

Inoltre, i mockup dinamici e interattivi che offre Figma sono una forma avanzata di prototipazione che consente di simulare

l'esperienza utente in modo molto realistico, consentendo di testare e verificare l'usabilità dell'interfaccia e l'esperienza dell'utente.

Per creare un'interfaccia più completa possibile abbiamo seguito le regole di **Shneiderman** che sono un insieme di otto regole fondamentali per la progettazione di interfacce utente efficaci. Abbiamo utilizzato queste regole durante la progettazione dell'interfaccia utente dell'app Ratatouille per migliorarne la usabilità e l'esperienza dell'utente.

Ad esempio, abbiamo adottato la regola della *visibilità del sistema*, assicurandoci che tutte le funzionalità dell'app fossero facilmente accessibili all'utente e che i comandi necessari fossero chiaramente indicati. Inoltre, abbiamo adottato la regola della coerenza e dello standard, garantendo che l'interfaccia utente dell'app fosse uniforme e rispettasse gli standard di design comuni.

Abbiamo anche adottato la regola della flessibilità e dell'efficienza dell'utente, consentendo all'utente di personalizzare l'esperienza dell'app in base alle proprie preferenze e fornendo funzionalità che aiutano a velocizzare l'utilizzo dell'app.

Infine, abbiamo adottato la regola della prevenzione degli errori, implementando funzionalità che aiutano a prevenire errori dell'utente, come la conferma prima di eliminare dati o l'indicazione degli errori di input.

L'utilizzo di queste regole ci ha permesso di creare un'interfaccia utente più intuitiva e facile da usare per gli utenti dell'app Ratatouille.

Dopo la realizzazione dei prototipi, abbiamo deciso di effettuare un test di usabilità. La pianificazione del test di usabilità inizia con una attenta scelta dei valutatori che devono utilizzare i prototipi e poi valutarli.

### *Personas*

Nella progettazione abbiamo identificato diverse personas, ovvero personaggi immaginari che rappresentano i possibili utenti della mia applicazione Ratatouille. Ciascuna persona è stata creata in base a diverse caratteristiche come età, professione, livello di istruzione, abilità tecnologiche, in modo da poter testare l'usabilità dell'applicazione in modo completo e rappresentativo. Questa analisi

delle personas è fondamentale per comprendere le esigenze e le aspettative degli utenti e per valutare l'efficacia dell'interfaccia grafica e delle funzionalità dell'applicazione.

## Carmen Manna

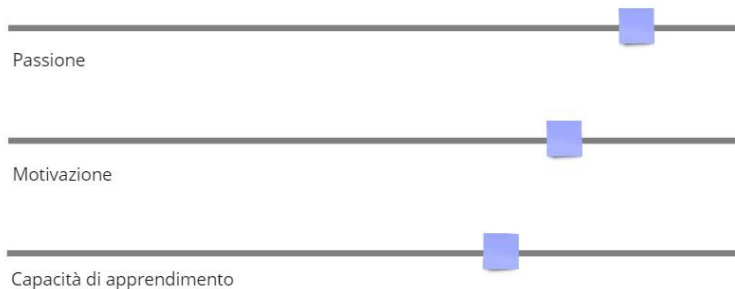
Titolare del ristorante la Carmensita.



Sono la titolare del rinomato ristorante La Carmensita, situato nel cuore della città. Nata e cresciuta in una famiglia di cuochi ho da sempre coltivato la passione per la cucina e il cibo di qualità per questo motivo Nel 2010 ho deciso di aprire il mio ristorante.

<b>Età</b>	52
<b>Lavoro</b>	Imprenditrice
<b>Titolo di studi</b>	Diploma istituto alberghiero

### Fattori chiave



### Obbiettivi

*Cosa stai davvero cercando di fare?*

- Il nostro obiettivo principale è offrire ai nostri clienti un'esperienza culinaria indimenticabile, basata su ingredienti freschi e di alta qualità, cucinati con passione e creatività.
- Allo stesso tempo, il servizio per i nostri clienti è altrettanto importante. Vogliamo che ogni ospite si senta accolto e a proprio agio.

### Insoddisfazioni

*Cosa ti turba di più?*

- Nonostante il mio grande amore per la cucina e la mia passione per la gestione del mio ristorante, ci sono alcune cose che mi turbano e mi preoccupano. Una di queste è la gestione del personale.
- Ho una buona conoscenza delle tecnologie ma ho difficoltà a leggere il testo su schermi di piccole dimensioni.

# Aldo Mancini

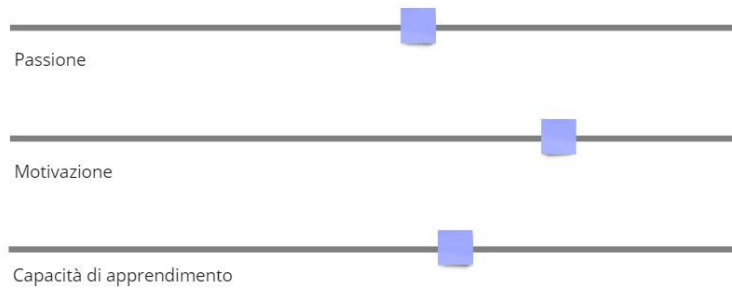
Cameriere del ristorante la Carmensita.



Sono Aldo Mancini, ho scoperto subito che mi piaceva molto il contatto con la gente e l'atmosfera di un ristorante. Dopo la laurea, ho deciso di dedicarmi completamente alla ristorazione.

<b>Età</b>	38
<b>Lavoro</b>	Dipendente
<b>Titolo di studi</b>	Laurea in sociologia

## Fattori chiave



## Obbiettivi

*Cosa stai davvero cercando di fare?*

- Sono convinto che la collaborazione tra camerieri e chef possa portare molti vantaggi e migliorare l'esperienza culinaria dei nostri ospiti, e mi impegnerò per promuovere e incentivare questo tipo di cooperazione.

## Insoddisfazioni

*Cosa ti turba di più?*

- Mi pesa quando il mio lavoro non viene apprezzato. Spesso ci sono situazioni in cui mi sforzo al massimo per garantire un servizio eccellente ai clienti, ma non vengo ringraziato o apprezzato per il mio lavoro.
- Non sono molto bravo con la tecnologia.

# Raimondo Ricci

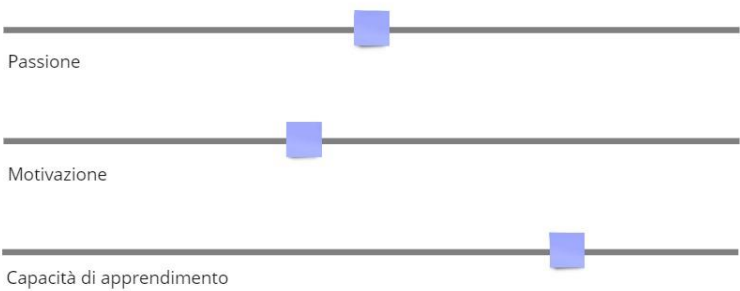
Cameriere del ristorante la Carmensita.



Sono Raimondo Ricci, cameriere del ristorante La carmensita, sono molto apprezzato dai clienti del ristorante per la mia cortesia, la mia attenzione ai dettagli e la mia capacità di creare un'atmosfera accogliente e rilassante.

Età	27
Lavoro	Dipendente
Titolo di studi	Diploma di Ragioneria

## Fattori chiave



## Obbiettivi

Cosa stai davvero cercando di fare?

- Desidero continuare a migliorare le mie competenze professionali e a crescere come cameriere. Ci sono sempre nuove cose da imparare in questo campo, e voglio essere sempre aggiornato sulle ultime tendenze e tecniche di servizio.
- Migliorare le mie conoscenze tecnologiche in futuro,

## Insoddisfazioni

Cosa ti turba di più?

- Mi rendo conto che non tutti i miei colleghi hanno la stessa passione e dedizione per questo lavoro come me. Ci sono alcuni camerieri che sembrano fare il minimo indispensabile, e questo può influire sulla qualità del servizio e sull'immagine del ristorante.

# Libero Udinesi

Chef del ristorante la Carmensita.



Sono Libero Udinesi, fin da bambino ho trascorso molte ore in cucina con mia madre e mia nonna, imparando i segreti delle loro ricette e apprezzando il piacere di condividere un pasto con le persone che ami.

<b>Età</b>	35
------------	----

<b>Lavoro</b>	Chef
---------------	------

<b>Titolo di studi</b>	Istituto alberghiero
------------------------	----------------------

## Fattori chiave

Passione

Motivazione

Capacità di apprendimento

## Obbiettivi

*Cosa stai davvero cercando di fare?*

- Il mio obiettivo principale è quello di migliorare la collaborazione con i camerieri, con cui ho instaurato un'ottima relazione di lavoro. Lavoriamo insieme per offrire un servizio impeccabile ai nostri clienti, coordinandoci costantemente per assicurare che ogni piatto sia servito al momento giusto e nella giusta quantità.

## Insoddisfazioni

*Cosa ti turba di più?*

- Ci sono situazioni in cui c'è molta confusione con gli ordini che arrivano in cucina e soprattutto purtroppo capita spesso di perdere alcuni ordini, questo rallenta molto il mio lavoro in cucina.



# Elonora Bianchi

Supervisore del ristorante la Carmensita.



Sono Eleanora Bianchi dopo la laurea in scienze gastronomiche, ho avuto la fortuna di lavorare in alcuni dei migliori ristoranti della città, dove ho acquisito una grande esperienza e competenza nella gestione del personale e nell'organizzazione del lavoro.

**Età** 31

**Lavoro** Dipendente

**Titolo di studi** Istituto Laurea in scienze gastronomiche

## Fattori chiave

Passione

Motivazione

Capacità di apprendimento

## Obbiettivi

*Cosa stai davvero cercando di fare?*

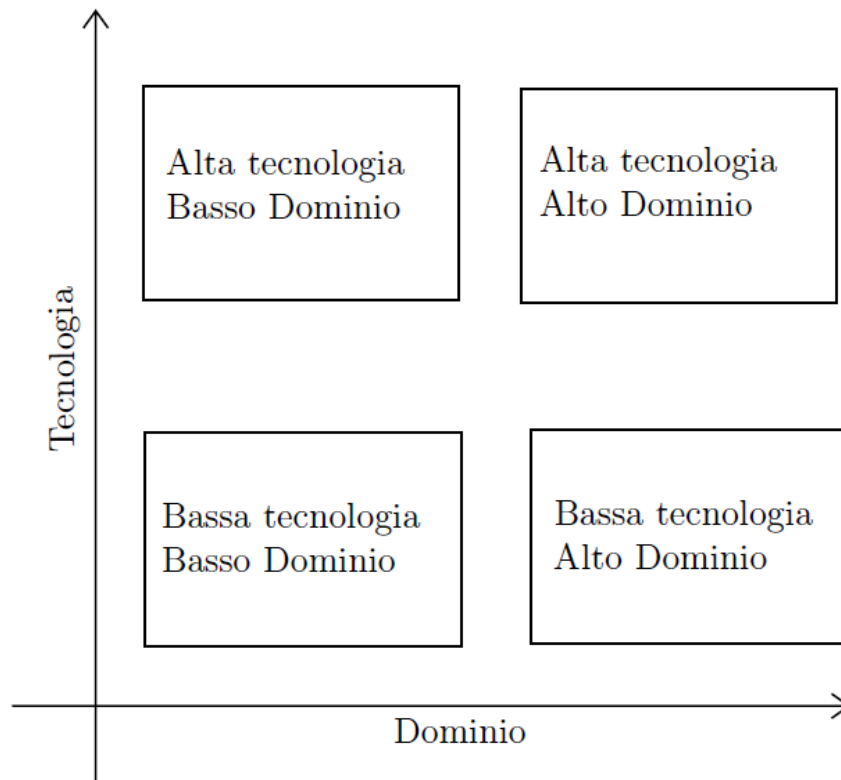
- Il mio obiettivo per il futuro è continuare a migliorare le mie capacità di leadership e di gestione del personale, con l'obiettivo di portare il Ristorante La Carmensita al top della ristorazione udinese e garantire un'esperienza indimenticabile ai nostri ospiti

## Insoddisfazioni

*Cosa ti turba di più?*

- Anche se metto tutto me stessa per rendere a meglio nel mio lavoro, ho alcune difficoltà con i turisti, il grande afflusso crea molti disagi dovuti alla lingua

Individuiamo di solito quattro tipologie di utenti:



Abbiamo scelto cinque persone, in quanto sappiamo (come ci ricorda la regola di Nielsen) che copriamo oltre l'85% di problemi di usabilità:

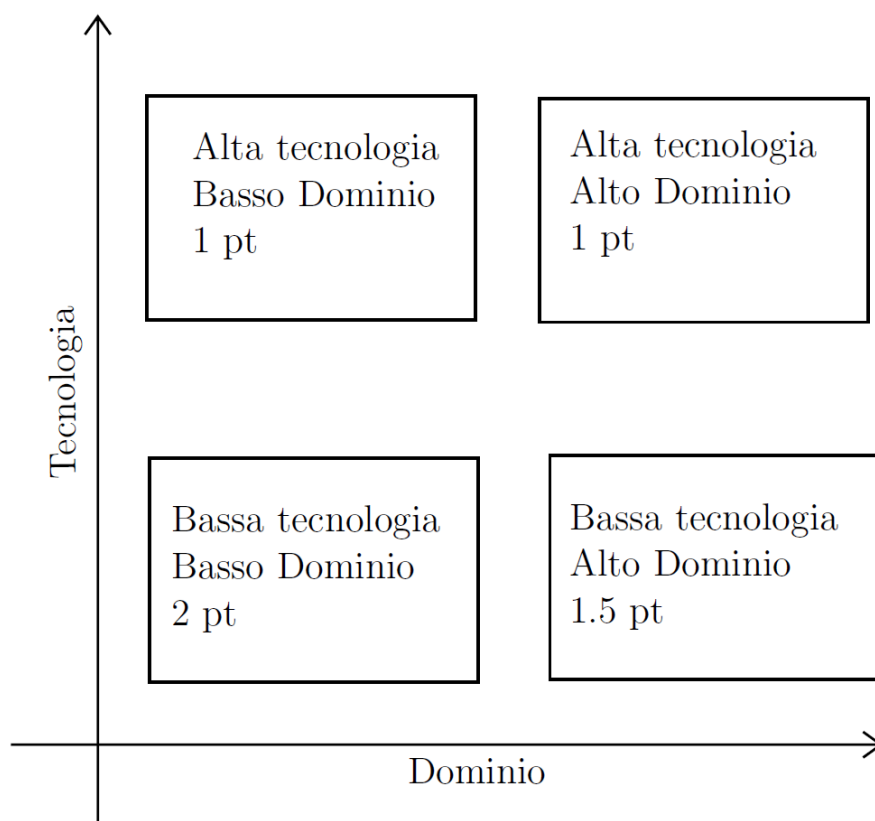
- **(1) Valutatore** con bassa conoscenza tecnologica e bassa conoscenza del dominio (Aldo Mancini)
- **(2) Valutatori** con bassa conoscenza tecnologica e alta conoscenza del dominio (Libero Udinesi, Eleonora Bianchi).

- **(1) Valutatore** con alta conoscenza della tecnologia e bassa conoscenza del dominio (Raimondo Ricci)
- **(1) Valutatore** con alta conoscenza della tecnologia e alta conoscenza del dominio (Carmen Manna)

### ***Tabelle di valutazione***

Le tabelle di valutazione sono strumenti utilizzati per valutare le prestazioni degli utenti in diversi contesti, per una valutazione accurata, ad ogni tipo di utente viene assegnato un punteggio.

I punteggi sono assegnati in base al livello di esperienza e competenza degli utenti nell'utilizzo del web e delle tecnologie informatiche in generale, un utente meno esperto che riesce a compiere il compito avrà più peso di un utente già esperto.



Inoltre, assegniamo un punteggio per il tempo di esecuzione di ogni compito

<b>0-2 Minuti</b>	<b>2-3 minuti</b>	<b>Più di tre minuti</b>
2pt	1pt	0.5pt

## *Definizione dei compiti*

I compiti andranno distribuiti tra le due applicazioni, ogni compito ha un suo peso specifico.

### RATATOUILLE STAFF

**Compito 1:** Invia ordine alla cucina;

**Compito2:** Visualizza storico ordine;

**Compito3:** Visualizza ordini in attesa.

Compito1	Compito2	Compito3
1.5pt	1pt	1pt

### RATATOUILLE ADMIN

**Compito 1:** Inserisci pietanza al menu;

**Compito2:** Elimina pietanza dal menu;

**Compito3:** Riordina ordine categorie;

**Compito4:** Elimina categoria.

<b>Compito1</b>	<b>Compito2</b>	<b>Compito3</b>	<b>Compito4</b>
1.5pt	1.5pt	2pt	2pt

### *Tecnica di valutazione*

La tecnica di valutazione usata è quella del *Mago di Oz*.

La Tecnica del Mago di Oz prevede l'utilizzo di attori o membri del gruppo di sviluppo per simulare l'interazione dell'utente con l'interfaccia utente.

Questa tecnica consente di testare l'interfaccia utente in un ambiente controllato ed è possibile individuare eventuali problemi o difficoltà dell'utente nell'utilizzo dell'interfaccia e correggerli prima di procedere con lo sviluppo completo dell'interfaccia.

### *Risultati dei test*

**P:** successo parziale (0.5).

**S:** successo (1).

**F:** fallimento (0).

	<b>Compito1</b>	<b>Compito2</b>	<b>Compito3</b>
<b>Aldo Mancini</b>	S (3m)	S (1m)	S (1m)
<b>Raimondo Ricci</b>	S (2m)	S (1m)	S (1m)
<b>Libero Udinesi</b>	P (4m)	S (1m)	S (1m)

	<b>Compito1</b>	<b>Compito2</b>	<b>Compito3</b>	<b>Compito4</b>
<b>Eleonora Bianchi</b>	S (2m)	P (3m)	P (3m)	S (1m)
<b>Carmen Manna</b>	P (4m)	S (1m)	S (4m)	F

## *Conclusioni*

### *Ratatouille Staff*

Ricordano la somma dei pesi, calcolando la somma ottenuta si ha che sul massimo punteggio possibile cioè 32 abbiamo totalizzato 28, *pari al 92% del successo.*

Il risultato ottenuto ci ha soddisfatto, non abbiamo ricevuto proposte in particolare l'app ha soddisfatto le aspettative.

### *Ratatouille Admin*

Ricordando la somma dei pesi, calcolando la somma ottenuta si ha che sul massimo punteggio possibile cioè 26 abbiamo totalizzato 19, *pari all'80% del successo.*

Il risultato ottenuto è certamente da migliorare, abbiamo anche ricevuto proposte per migliorare l'ultimo compito dove abbiamo avuto i risultati più deludenti.

Cioè di utilizzare icone che indicano che l'elemento è trascinabile, poiché non è molto chiaro nell'immediato e di ingrandire la



dimensione dei font utilizzati per migliorare quella che è la lettura delle informazioni.

## ***Glossario***

**Organizzazione:** strutturazione e pianificazione di attività o processi al fine di raggiungere determinati obiettivi.

**Efficienza:** capacità di eseguire le attività richieste nel modo più veloce ed economico possibile, senza sprechi.

**Flessibilità:** la capacità di adattarsi facilmente ai cambiamenti o alle nuove situazioni, di essere modificato o adeguato senza grandi difficoltà. Nel contesto dello sviluppo software, la flessibilità può essere intesa come la capacità di un'applicazione o di un sistema di adattarsi alle esigenze degli utenti o alle variazioni delle condizioni di utilizzo, senza compromettere la qualità o la funzionalità del software.

**QR code:** codice a barre bidimensionale che può essere scansionato tramite la fotocamera di uno smartphone o di un tablet, consentendo di accedere a informazioni o siti web.

**Open data:** dati resi disponibili al pubblico e condivisibili liberamente da parte di organizzazioni o istituzioni.

**Use case:** rappresentazione di un'interazione tra un attore e un sistema che descrive le azioni che l'attore compie e le risposte del sistema.

**Mock-up:** rappresentazione grafica di un'interfaccia utente che mostra l'aspetto e il layout di un'applicazione o di un sito web. Pre-release: fase del ciclo di vita del software in cui il prodotto è in fase di sviluppo e test, ma non è ancora pronto per essere rilasciato ufficialmente.

**Bug:** problema o errore che causa il malfunzionamento dell'applicazione o del software.

**SDK:** acronimo di Software Development Kit, un insieme di strumenti e risorse utilizzati dagli sviluppatori per creare applicazioni o software.

**Dashboard:** una pagina web o un'applicazione che fornisce una panoramica dei dati e delle statistiche rilevanti.

**Crash:** termine che indica il blocco o l'interruzione improvvisa del funzionamento di un'applicazione o di un software.

**Traccia degli stack (stack trace):** una lista di chiamate di funzione che indica la sequenza di eventi che hanno portato al crash dell'applicazione o del software.

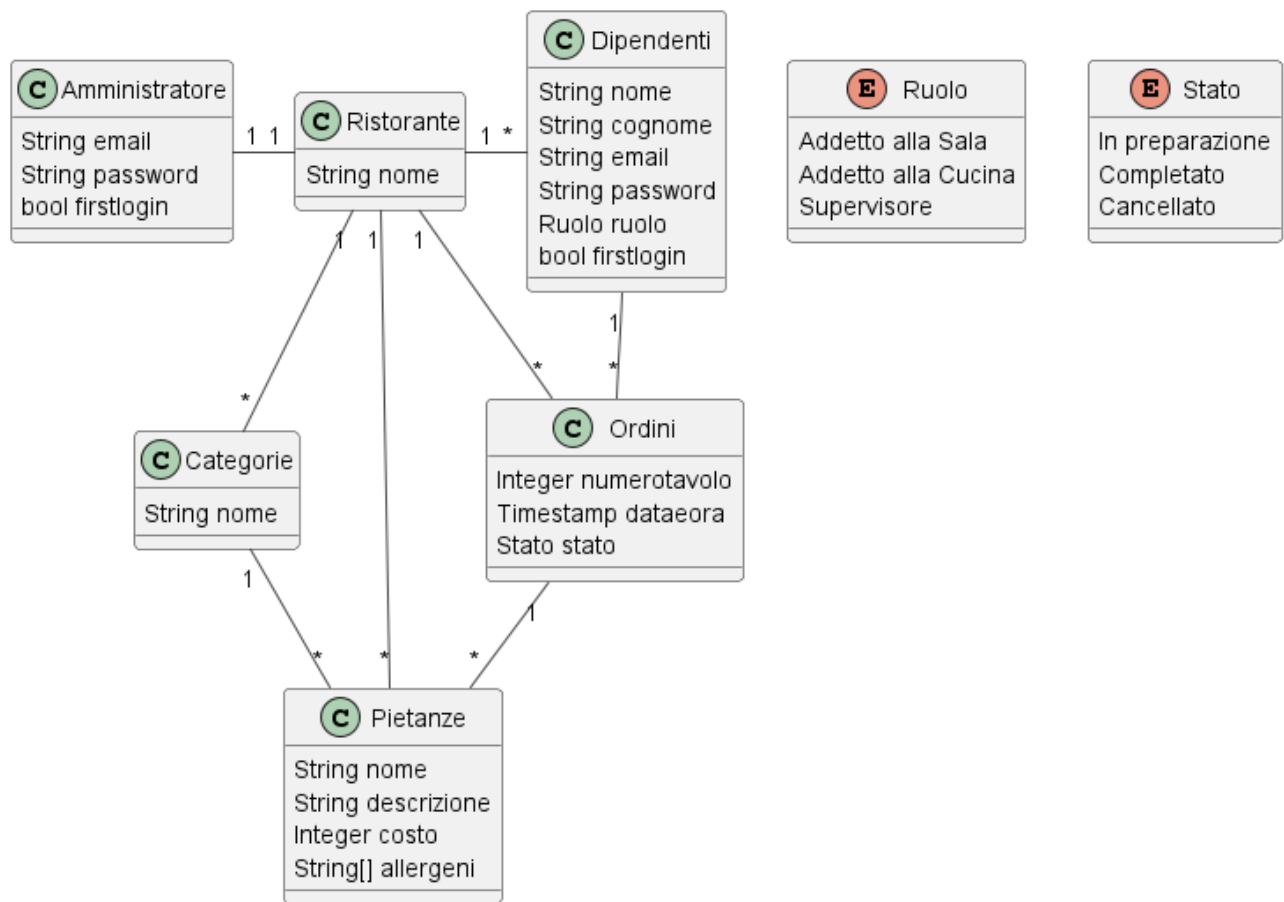
**Report:** un documento che contiene dati e analisi su un determinato argomento o processo.

**Tasso di crash:** il numero di volte in cui l'applicazione o il software ha smesso di funzionare correttamente rispetto al numero totale di volte in cui è stato utilizzato.

**Esperienza degli utenti:** la percezione e l'esperienza che gli utenti hanno durante l'utilizzo di un'applicazione o di un software.

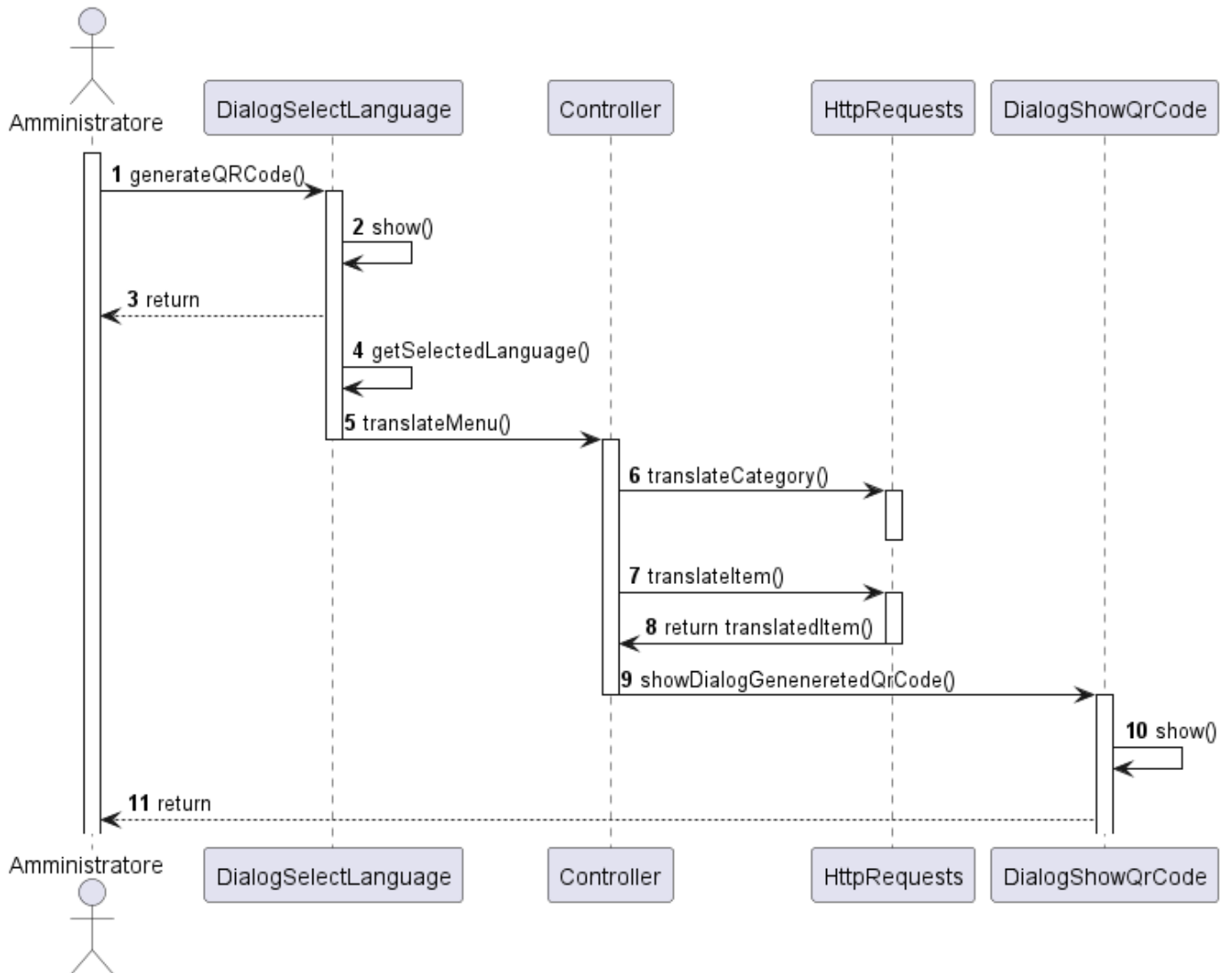
# Specifica dei requisiti

*Classi, oggetti e relazioni di analisi.*

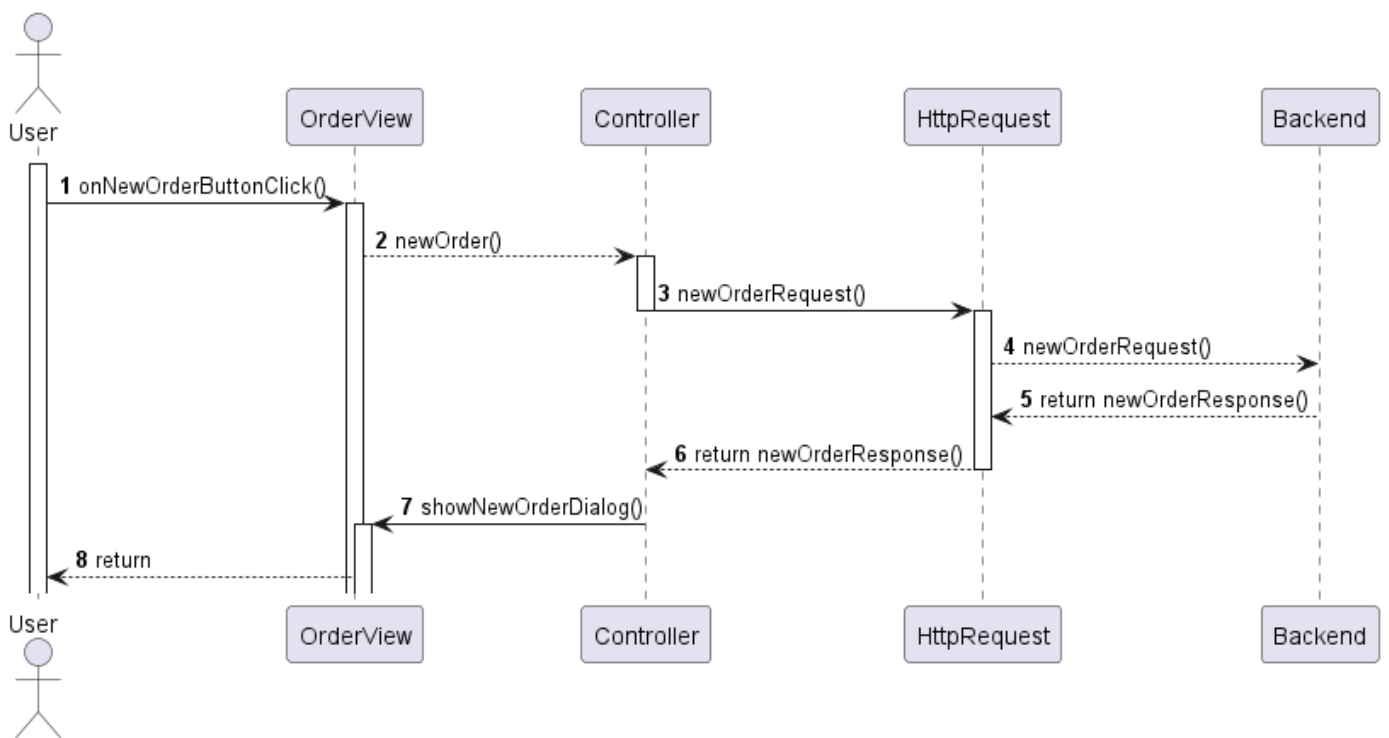


## Diagrammi di sequenza di analisi per due casi d'uso

### Generate QR Code

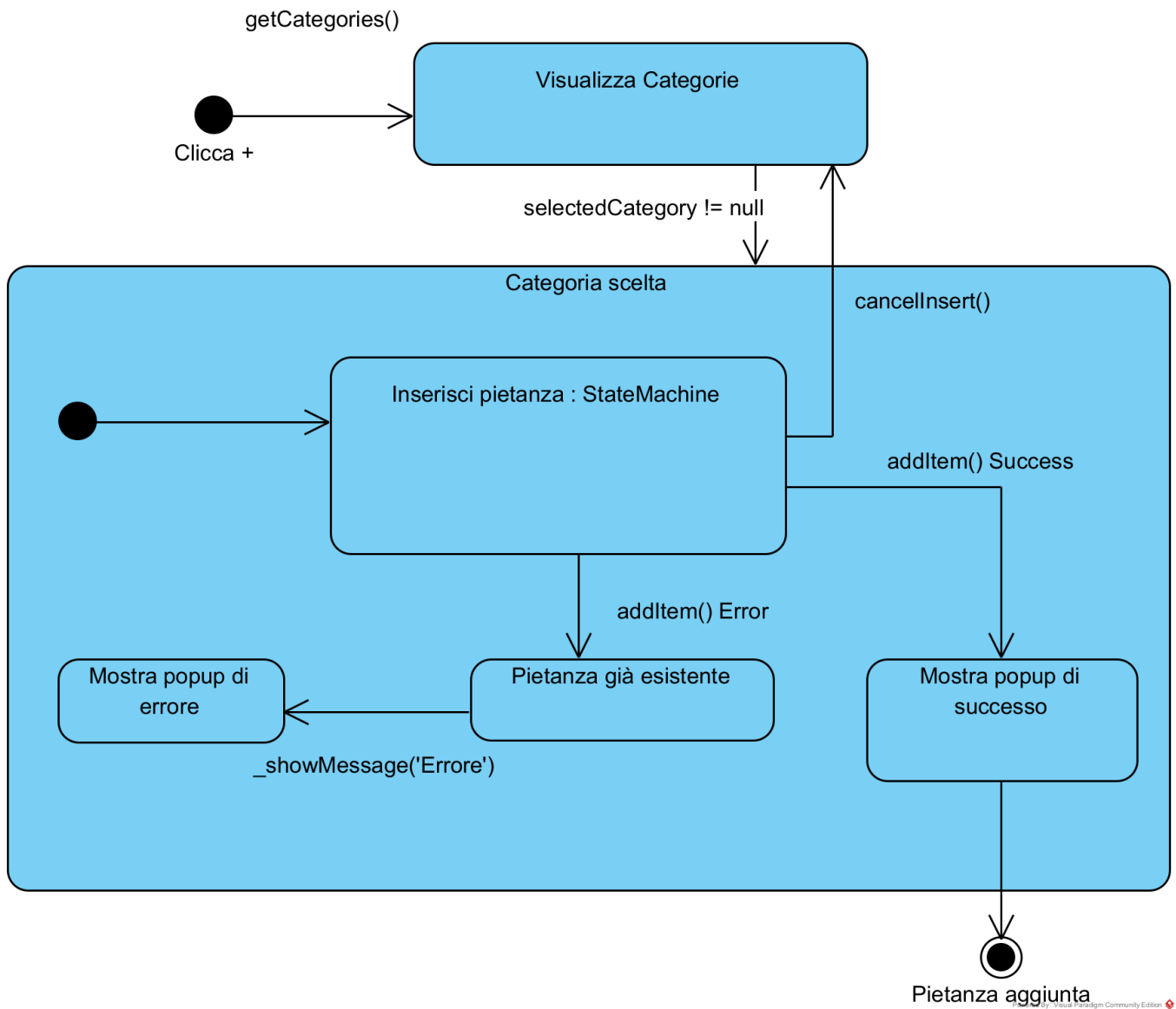


### Insert New Order

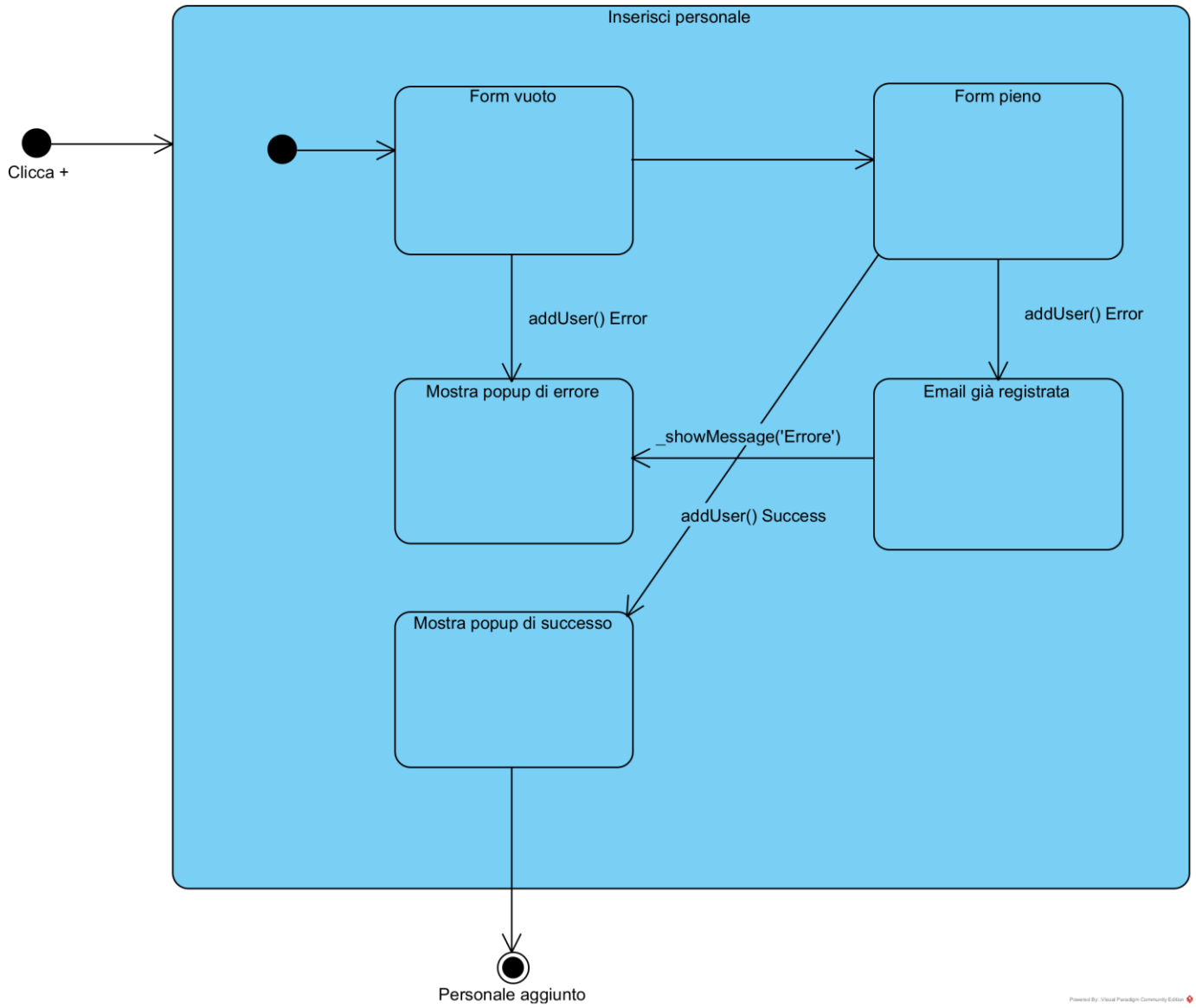


## Prototipazione funzionale via statechart dell'interfaccia grafica

### AGGIUNGI ELEMENTO AL MENU



## AGGIUNGI PERSONALE





## **Design del sistema**

L'architettura a tre livelli che descriverò utilizza Flutter come front-end, Node.js ed Express come back-end e MongoDB come database. Qui di seguito fornirò una descrizione dettagliata di ciascun livello:

### *Front-end*

Il front-end dell'architettura utilizza Flutter come framework di sviluppo per creare applicazioni mobili. Flutter è un framework open source sviluppato da Google che utilizza il linguaggio di programmazione Dart per creare app native su piattaforme multiple come iOS, Android e web. Flutter fornisce un'ampia gamma di widget, librerie e strumenti che semplificano la creazione di interfacce utente interattive e reattive. Il front-end Flutter comunica con il back-end attraverso API RESTful.



### *Back-end*

Il back-end dell'architettura utilizza Node.js e Express come framework di sviluppo per creare l'applicazione server. Node.js è un ambiente di runtime JavaScript basato su Chrome V8 che consente di eseguire codice JavaScript lato server. Express è un framework di sviluppo web per Node.js che semplifica la creazione di applicazioni web. Express fornisce un'ampia gamma di funzionalità per la creazione di server web, come la gestione delle richieste HTTP, il routing, la gestione dei cookie e delle sessioni e la connessione a database come MongoDB.

## *Database*

Il database utilizzato in questa architettura è MongoDB, un database NoSQL orientato ai documenti. MongoDB memorizza i dati in documenti JSON che possono essere facilmente scalati e distribuiti su più server. MongoDB supporta le operazioni di lettura e scrittura su dati strutturati e non strutturati, supporta l'indicizzazione dei dati per una ricerca veloce e offre funzionalità di aggregazione avanzate per la manipolazione dei dati.



### **Come funziona?**

Nella nostra architettura a tre livelli, il front-end comunica con il back-end attraverso API RESTful. Quando un utente interagisce con l'applicazione Flutter, il front-end invia una richiesta HTTP al back-

end attraverso un'API RESTful. Il back-end elabora la richiesta e invia la risposta al front-end sotto forma di dati JSON. Quando il front-end riceve i dati JSON, li elabora e li presenta all'utente finale.

Il back-end utilizza Node.js e Express per gestire le richieste HTTP dal front-end. Quando una richiesta HTTP arriva al back-end, Express esegue il routing della richiesta verso il corretto controller che si occupa dell'elaborazione della richiesta. Il controller interagisce con MongoDB per recuperare o aggiornare i dati richiesti e invia la risposta sotto forma di dati JSON al front-end.

Infine, MongoDB viene utilizzato per memorizzare i dati dell'applicazione. MongoDB fornisce un'interfaccia intuitiva e flessibile per l'interazione con i dati. Il back-end comunica con MongoDB attraverso un driver MongoDB specifico per Node.js **Moongose**, che semplifica l'accesso ai dati e offre funzionalità avanzate per la manipolazione dei dati.

In sintesi, l'architettura a tre livelli descritta utilizza Flutter come front-end, Node.js e Express come back-end e MongoDB come database.

## **Motivazioni delle scelte**

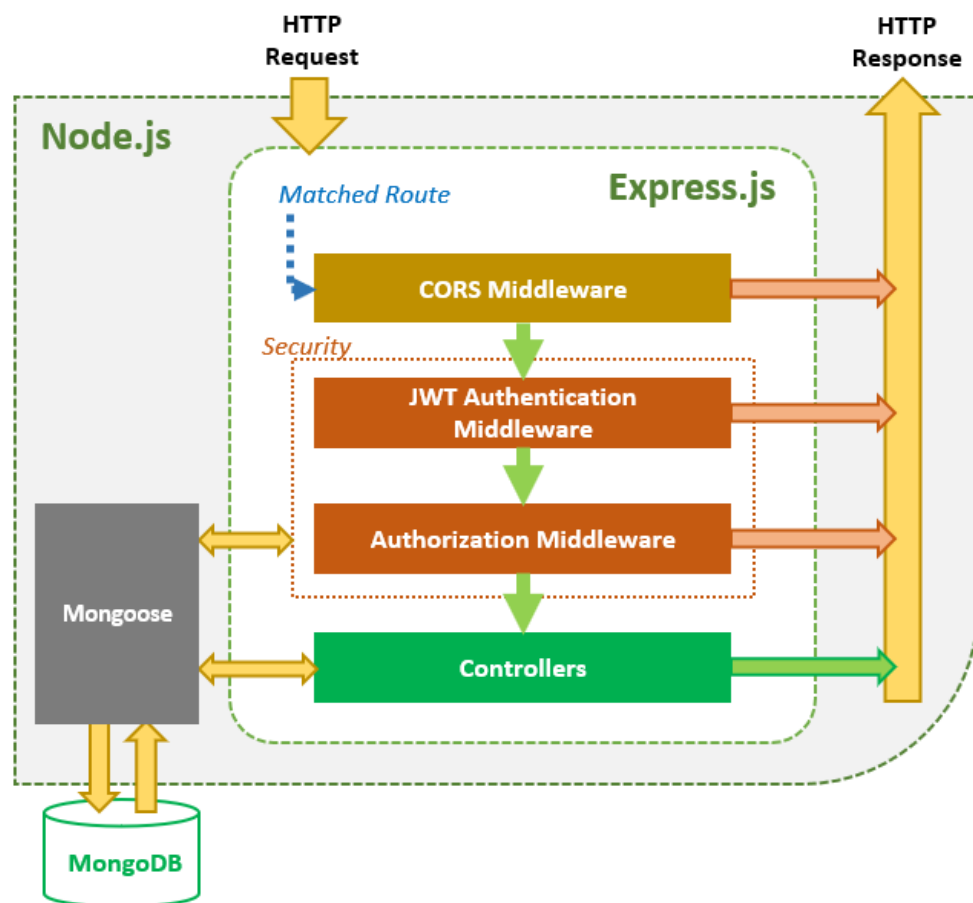
Le nostre esigenze erano molto chiare, volevamo creare due applicazioni distinte.

Ratatouille Staff dedicata ai dipendenti del ristorante, accessibile tramite dispositivi mobile e Ratatouille Admin dedicata ad Amministratori e Supervisor e accessibili tramite dispositivi desktop.

Valutando le varie possibilità abbiamo deciso di usare *Flutter*, un framework molto recente e soprattutto multiplatforma, che permette il riciclo di molto codice tra i progetti delle due piattaforme e soprattutto offre un grande supporto con **Firestore Analytics**, che ci sarà molto utile per la valutazione dell'usabilità sul campo.

Flutter ha un grande supporto con tutti i prodotti Firebase, essendo entrambi di casa **Google**, ma per fini didattici abbiamo deciso di non usare sistemi come **Firestore Authentication** che si hanno una

grande affidabilità ma ci semplificavano molto la vita per ciò che riguarda la sicurezza, infatti è un aspetto che abbiamo studiato con molto interesse ed infine abbiamo implementato utilizzando il classico metodo dei **Token JWT**, quindi la nostra Rest api, gestisce l'autenticazione e soprattutto gestisce la sicurezza tramite un sistema **Bearer Token**.



Attraverso un sistema **Middleware** abbiamo dato la possibilità agli utenti delle nostre app di mantenere la sessione corrente, con l'utilizzo un sistema di **Shared Preferences** nella front-end che salva il Token e al prossimo accesso verifica la sua validità.

Tutto questo è retto da Node.js, Express e Mongoose che offrono una soluzione back-end altamente personalizzabile e altamente performante.

Node.js infatti è noto per la sua velocità e scalabilità, Express fornisce un'infrastruttura leggera e veloce per la gestione delle richieste HTTP, consentendo di creare rapidamente applicazioni web efficienti e Mongoose fornisce inoltre un'API semplice e intuitiva per la gestione dei dati in MongoDB.

## **Diagramma delle classi di design**

### *Classi parser o serializer (Modelli)*

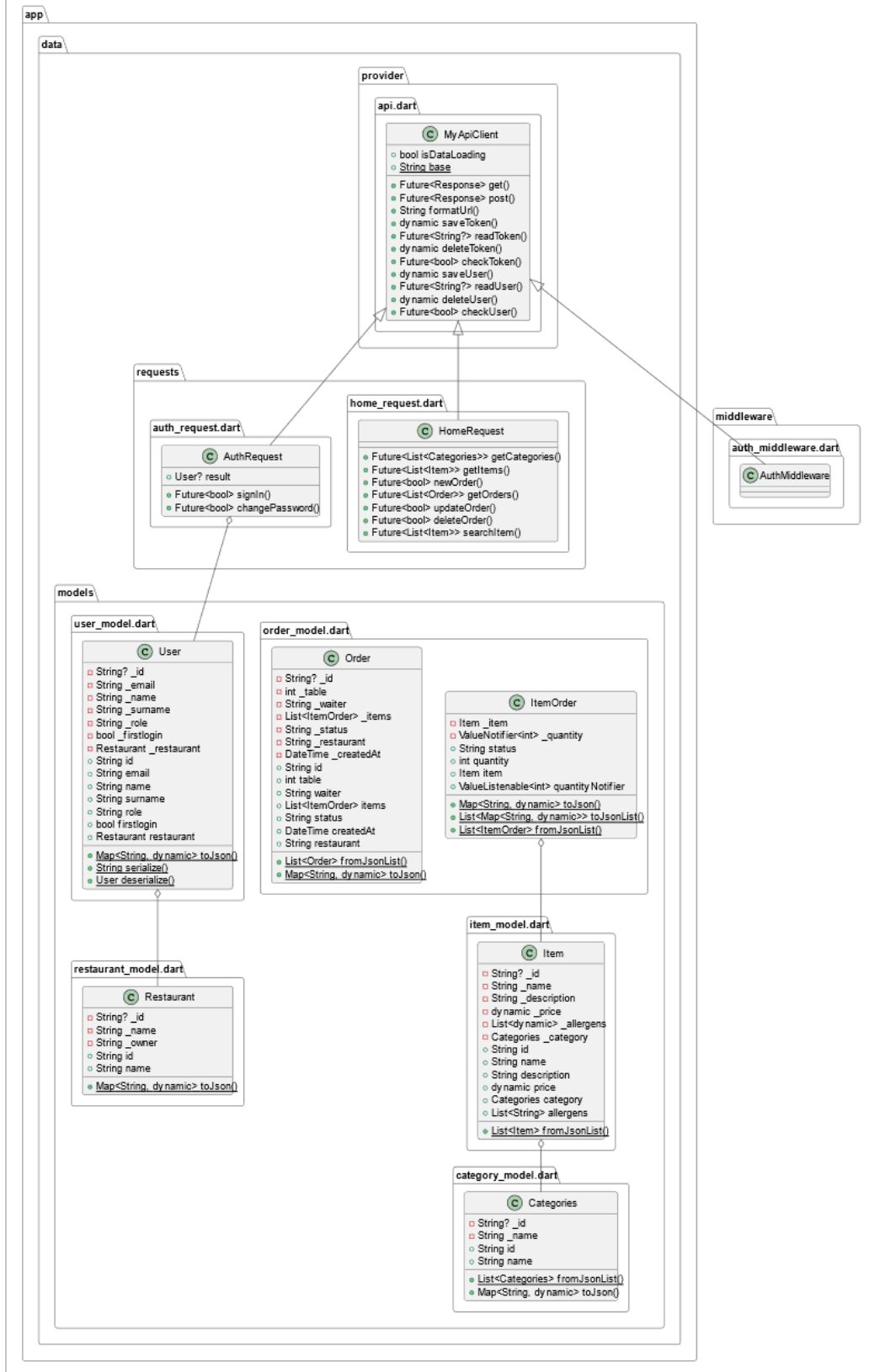
La serializzazione è il processo di conversione di un oggetto in un formato dati che può essere trasmesso o salvato, mentre la deserializzazione è il processo inverso, ovvero la conversione di un formato dati in un oggetto nativo dell'applicazione.

La maggior parte dei linguaggi di programmazione supporta la serializzazione e la deserializzazione di oggetti, nel nostro caso i parser JSON convertono i dati JSON in un formato utilizzabile da un linguaggio di programmazione specifico, mentre i serializer JSON convertono i dati dal formato utilizzato dal linguaggio di programmazione in JSON.

Sono classi molto importanti poiché ci permettono di gestire i dati che riceviamo dalla Restapi ed inserirli in Oggetti per poi essere utilizzati nel nostro Front-end.

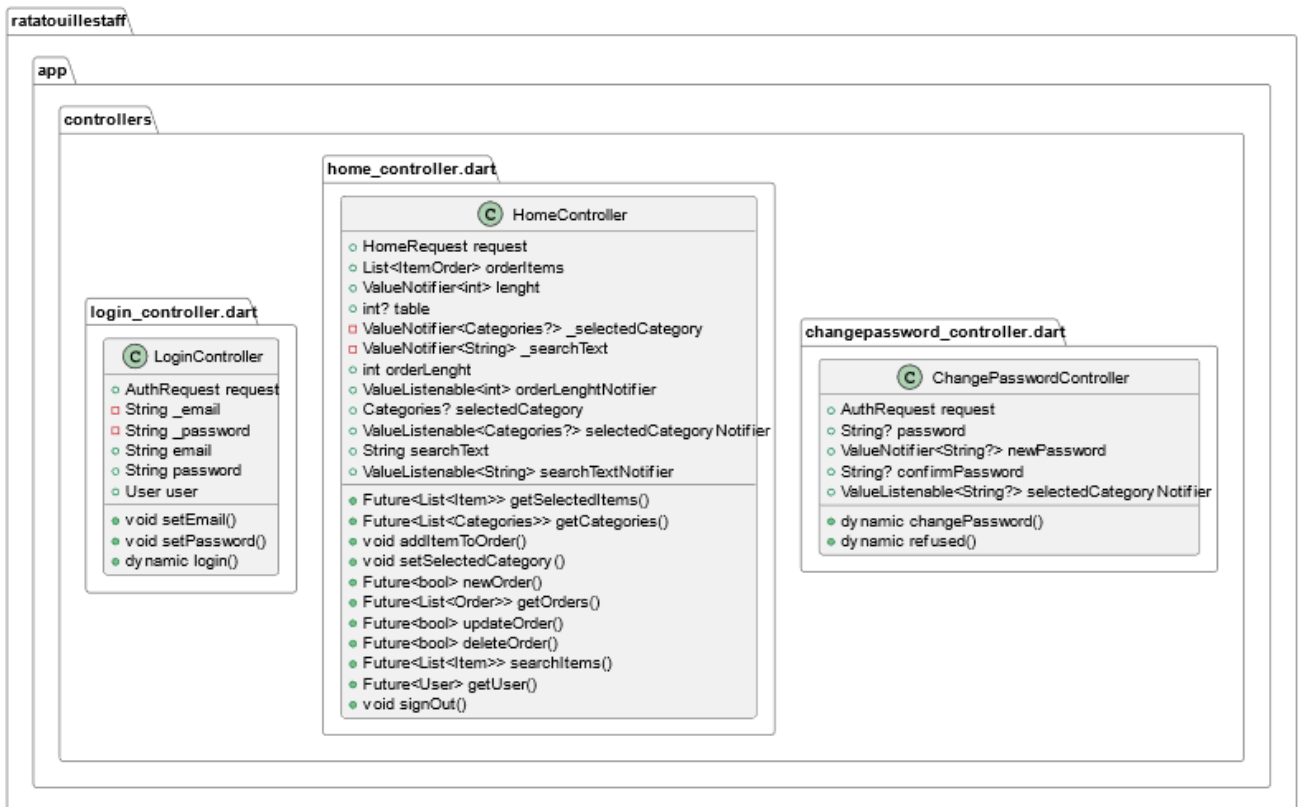


ratatouillestaff



## Classi controller

Il compito principale di una classe controller per front-end è quello di gestire la logica dell'interfaccia utente e la logica di interazione con l'utente. Ciò significa che il controller riceve gli eventi dell'utente (come i clic del mouse o le pressioni dei tasti) e modifica la UI di conseguenza.



## *Classi viste*

Le classi per le viste (o "view classes" in inglese) sono utilizzate per rappresentare l'interfaccia utente (UI) in un'applicazione web. In un'architettura **MVC (Model-View-Controller)**, la classe per la vista è responsabile di visualizzare i dati provenienti dal modello e di gestire gli eventi dell'utente.

La classe per la vista definisce come i dati devono essere presentati all'utente e come l'utente può interagire con l'interfaccia utente.

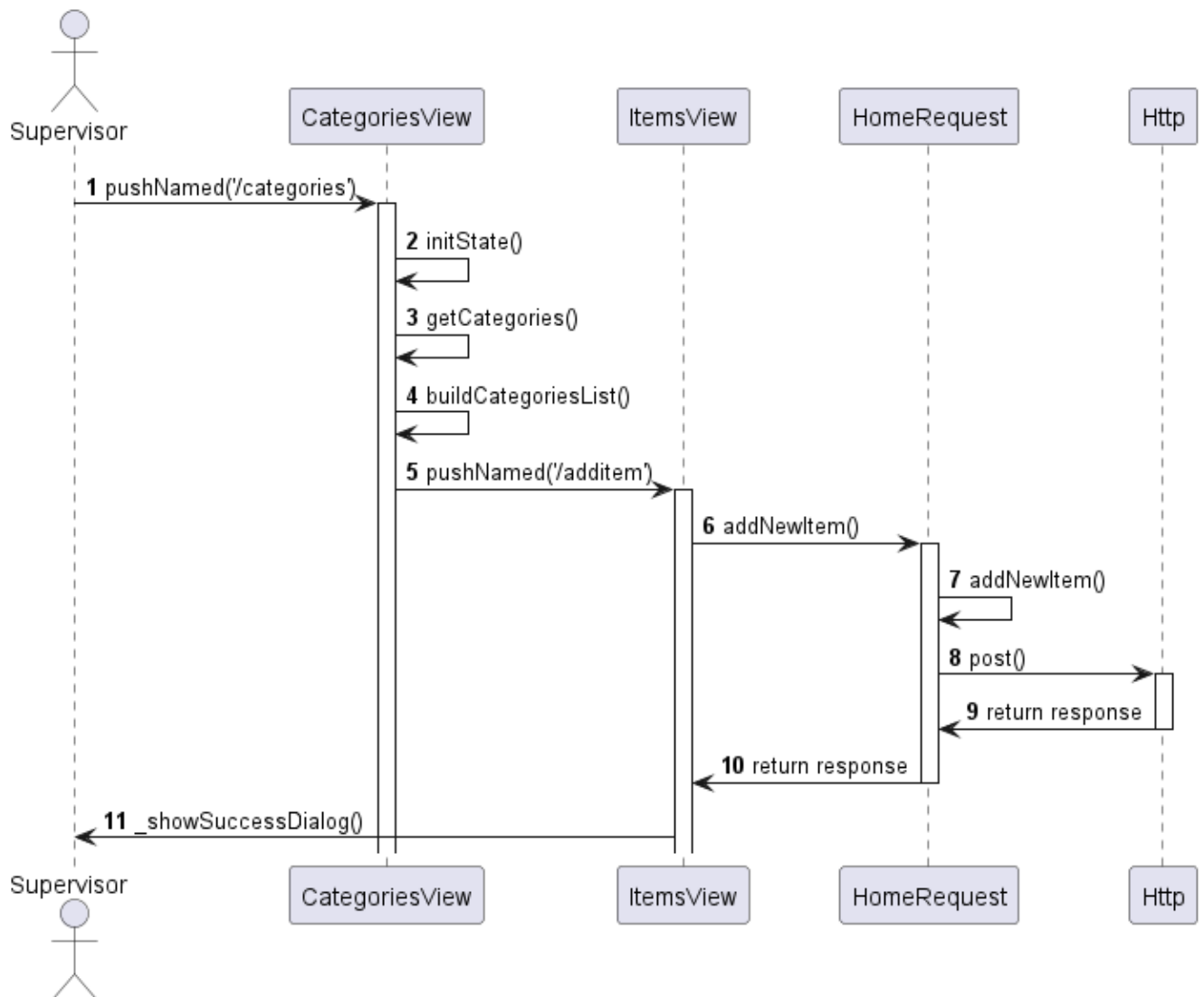
Le classi per le viste sono strettamente legate alle *classi controller e ai modelli*. Quando l'utente interagisce con l'interfaccia utente, il controller elabora l'evento e comunica con il modello per ottenere i dati necessari. Successivamente, il controller utilizza la classe per la vista per visualizzare i dati ottenuti dal modello.

Per godere delle Classi di design nel dettaglio di seguito sarà allegato il link:

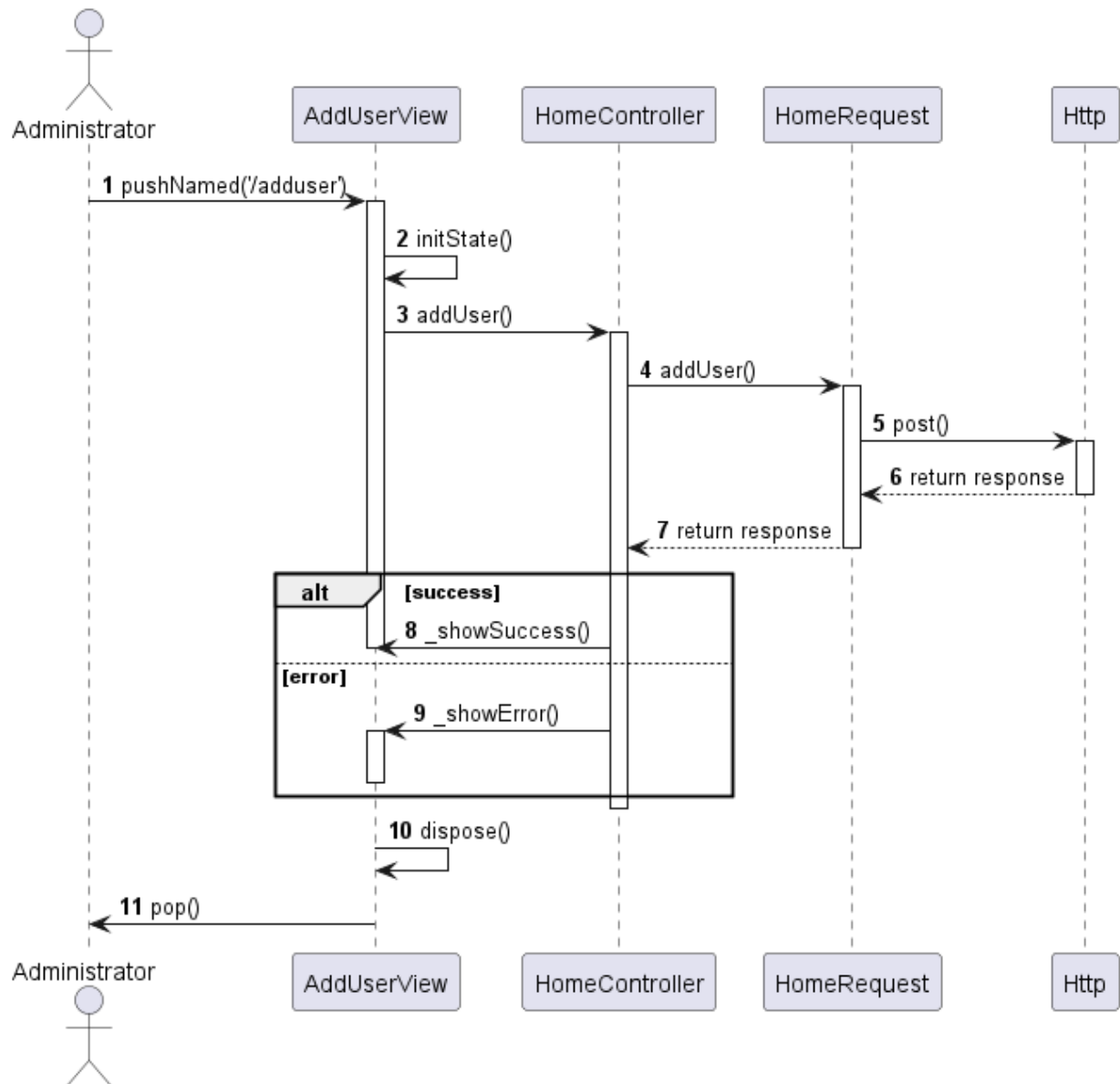
[https://drive.google.com/file/d/1k41l1W3EDDiNi2dv-dd7EOPYys-Xyfa\\_/view?usp=sharing](https://drive.google.com/file/d/1k41l1W3EDDiNi2dv-dd7EOPYys-Xyfa_/view?usp=sharing)

## Diagrammi di sequenza di design per due casi d'uso

### *Sequence “Aggiungi elemento al menu”*



### Sequence “Aggiungi dipendente”



# Testing e valutazione sul campo dell'usabilità

## *Unit Testing*

### LOGIN BLACKBOX SECT

Il test del login è uno dei test più comuni nel testing del software, poiché l'accesso ad un'applicazione web o mobile solitamente richiede la registrazione e l'autenticazione dell'utente.

Per testare il login come un SECT black box, si seguono i seguenti passaggi:

Identificare i requisiti funzionali e non funzionali dell'applicazione rispetto alla funzione di login, come ad esempio le credenziali richieste, il formato di inserimento delle credenziali, le politiche di sicurezza per le password, ecc.

Sviluppare una serie di test case per il login, basati sui requisiti identificati. Ogni test copre uno scenario unico, ad esempio:

Testare la validità dell'username e della password

Testare la gestione degli errori quando si inseriscono  
credenziali errate

Con il testing SECT black box, non si ha accesso al codice sorgente, quindi i test case sono sviluppati sulla base dei requisiti e delle specifiche dell'applicazione, non c'è bisogno di conoscere l'architettura interna dell'applicazione o la logica del codice per testare la funzione di login.

```
19 void wrongEmail() {
20     test('test request signin with wrong email', () async {
21         final result = await http.post(Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
22             headers: <String, String>{
23                 'Content-Type': 'application/json; charset=UTF-8',
24             },
25             body: jsonEncode(<String, String>{
26                 'email': 'wrongemail',
27                 'password': 'Temporanea123@'
28             }));
29         expect(result.statusCode, 400);
30     });
31 }
32
33 void wrongPassword() {
34     test('test request signin with wrong password', () async {
35         final result = await http.post(Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
36             headers: <String, String>{
37                 'Content-Type': 'application/json; charset=UTF-8',
38             },
39             body: jsonEncode(<String, String>{
40                 'email': 'carmen.manna@gmail.com',
41                 'password': 'wrongpassword'
42             }));
43         expect(result.statusCode, 401);
44     });
45 }
46
47 void correctLogin() {
48     test('test request signin with correct credentials', () async {
49         final result = await http.post(Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
50             headers: <String, String>{
51                 'Content-Type': 'application/json; charset=UTF-8',
52             },
53             body: jsonEncode(<String, String>{
54                 'email': 'carmen.manna@gmail.com',
55                 'password': 'Temporanea123@'
56             }));
57         expect(result.statusCode, 200);
58     });
59 }
```



```
61 void nullEmail() {
62     test('test request signin with null email', () async {
63         final result = await http.post(
64             Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
65             headers: <String, String>{
66                 'Content-Type': 'application/json; charset=UTF-8',
67             },
68             body: jsonEncode(<String, String>{
69                 'email': '',
70                 'password': 'Temporanea123@'
71             }));
72         expect(result.statusCode, 400);
73     });
74 }
75
76 void nullPassword() {
77     test('test request signin with null password', () async {
78         final result = await http.post(
79             Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
80             headers: <String, String>{
81                 'Content-Type': 'application/json; charset=UTF-8',
82             },
83             body: jsonEncode(<String, String>{
84                 'email': 'carmen.manna@gmail.com',
85                 'password': ''
86             }));
87         expect(result.statusCode, 400);
88     });
89 }
90
91 void nullEmailAndPassword() {
92     test('test request signin with null email and password', () async {
93         final result = await http.post(
94             Uri.parse('https://13.74.188.142:5000/api/v1/admin/signin'),
95             headers: <String, String>{
96                 'Content-Type': 'application/json; charset=UTF-8',
97             },
98             body: jsonEncode(<String, String>{
99                 'email': '',
100                 'password': ''
101             }));
102         expect(result.statusCode, 400);
103     });
104 }
```

*Le classi di equivalenza per il test di login che coinvolgono e-mail e password:*

**Email e password corretti:** in questo caso, l'utente inserisce una email e una password valide che corrispondono a quelle memorizzate nel database dell'applicazione. L'accesso dovrebbe essere consentito.

**Email corretta e password errata:** in questo caso, l'utente inserisce una email valida, ma una password non valida che non corrisponde a quella memorizzata nel database dell'applicazione. L'accesso dovrebbe essere negato.

**Email non valida:** in questo caso, l'utente inserisce una email che non è nel formato corretto o che non esiste nel database dell'applicazione. L'accesso dovrebbe essere negato.

**Password non valida:** in questo caso, l'utente inserisce una password che non soddisfa i requisiti di sicurezza dell'applicazione.

Ad esempio, la password potrebbe essere troppo corta, non contenere caratteri speciali, o essere stata utilizzata in passato.

L'accesso dovrebbe essere negato.

**Email e password vuote:** in questo caso, l'utente non inserisce alcuna email o password. L'accesso dovrebbe essere negato.

**Email o password vuota:** in questo caso, l'utente inserisce solo uno dei due campi, lasciando l'altro vuoto. L'accesso dovrebbe essere negato.

## SIGNUP WHITEBOX WECT

Il testing del codice White Box si concentra sulla valutazione interna del codice; quindi, è possibile analizzare il flusso di esecuzione e la logica del programma. Ciò significa che, durante il testing White Box, i tester conoscono l'implementazione del codice e utilizzano questa conoscenza per creare test specifici per raggiungere la massima copertura possibile delle diverse funzionalità dell'applicazione.

In particolare, il testing White Box della funzione di sign-up si concentra sul controllo del flusso del programma, sulla valutazione delle variabili e sulle decisioni prese all'interno della funzione di sign-up. Ciò può essere fatto utilizzando diverse tecniche di testing

come la code coverage analysis, la path coverage analysis, la branch coverage analysis e la statement coverage analysis.

Inoltre, il testing WECT (White-box Equivalence Class Testing) viene utilizzato per testare gli input della funzione di sign-up. In questo caso, si dividono gli insiemi degli input in classi di equivalenza e vengono testati solo alcuni membri di ogni classe. Questo approccio aiuta a ridurre il numero di test necessari per coprire tutti i casi possibili di input, ma garantisce comunque una copertura sufficiente delle possibili combinazioni di input.

In sintesi, il testing White Box e WECT per la funzione di sign-up si concentra sull'analisi della logica del codice e sulla valutazione degli input della funzione. Ciò aiuta a garantire che la funzione di sign-up funzioni correttamente in ogni possibile scenario, riducendo al minimo il rischio di bug e malfunzionamenti nell'applicazione.

```
24 void validSignUp1() {
25     test('Test user sign up with role waiter', () async {
26         var result = await request.userSignUp('test', 'test', 'test@test.it', 'test', 'waiter');
27         expect(result, true);
28     });
29 }
30
31 void validSignUp2(){
32     test('Test user sign up with role kitchen', () async {
33         var result = await request.userSignUp('test', 'test', 'test@test.it', 'test', 'kitchen');
34         expect(result, true);
35     });
36 }
37
38 void validSignUp3(){
39     test('Test user sign up with role waiter', () async {
40         var result = await request.userSignUp('test', 'test', 'test@test.it', 'test', 'supervisor');
41         expect(result, true);
42     });
43 }
44
45 void notValidEmail() {
46     test('Test user sign up', () async {
47         var result = await request.userSignUp('test', 'test', 'test', 'test', 'waiter');
48         expect(result, false);
49     });
50 }
51
52 void notValidRole() {
53     test('Test user sign up', () async {
54         var result = await request.userSignUp('test', 'test', 'test@test.it', 'test', 'test');
55         expect(result, false);
56     });
57 }
```

*Le classi di equivalenza per il test del metodo userSignUp*

*Email e password valide, nome e cognome validi, ruolo waiter. La registrazione dovrebbe essere completata con successo.*

*Email e password valide, nome e cognome validi, ruolo kitchen. La registrazione dovrebbe essere completata con successo.*

*Email e password valide, nome e cognome validi, ruolo supervisor. La registrazione dovrebbe essere completata con successo.*

*Email non valida, password valida, nome e cognome validi, qualsiasi ruolo. La registrazione dovrebbe fallire.*

*Email valida, password non valida, nome e cognome validi, qualsiasi ruolo. La registrazione dovrebbe fallire.*

## *Valutazione dell'usabilità sul campo*

Per la valutazione dell'usabilità sul campo abbiamo usato strumenti forniti da Firebase per il monitoraggio dell'applicazione post-rilascio.

Firebase Analytics e Crashlytics sono due strumenti importanti per la valutazione dell'usabilità di un'applicazione mobile.

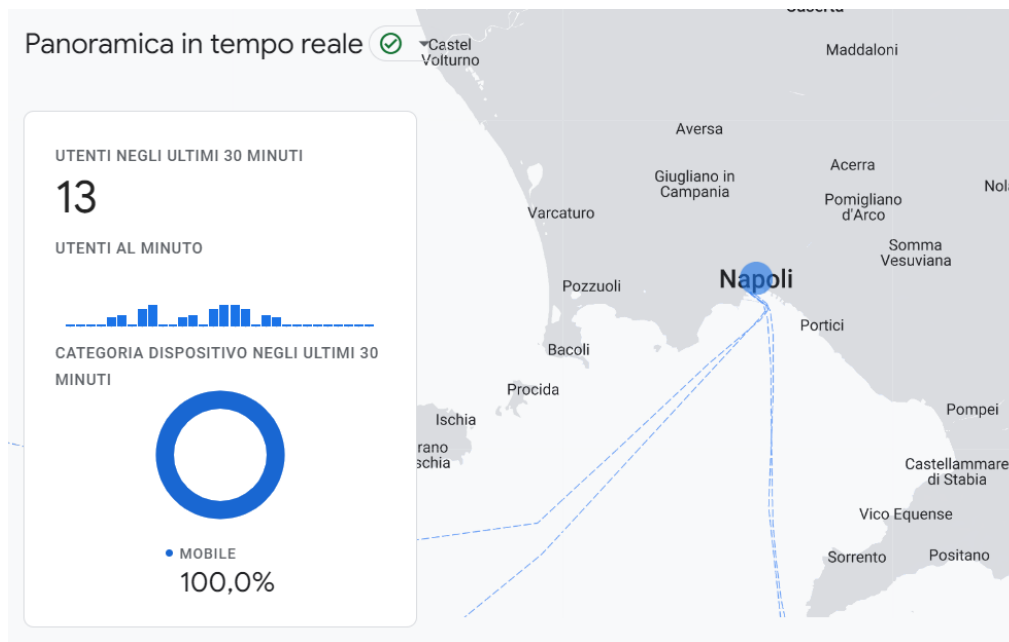
Firebase Analytics è uno strumento di analisi che consente agli sviluppatori di monitorare l'utilizzo dell'applicazione mobile. In particolare, consente di raccogliere dati sulle attività degli utenti, come le azioni compiute all'interno dell'applicazione, le fonti di traffico, le conversioni e le retention rate. Firebase Analytics fornisce inoltre una serie di report predefiniti e personalizzabili, che consentono agli sviluppatori di ottenere una visione d'insieme dell'utilizzo dell'applicazione.

Crashlytics, invece, è uno strumento di monitoraggio degli errori che consente agli sviluppatori di identificare e risolvere i problemi dell'applicazione. In particolare, Crashlytics raccoglie informazioni sulle cause dei crash dell'applicazione, come i messaggi di errore, le

tracce di stack e le informazioni sul dispositivo. Crashlytics fornisce inoltre report dettagliati sui crash, che consentono agli sviluppatori di identificare i problemi più comuni e di risolverli rapidamente.

## FIREBASE ANALYTICS

Firebase Analytics offre diversi tipi di grafici per aiutare gli sviluppatori a comprendere le attività degli utenti nelle loro app.

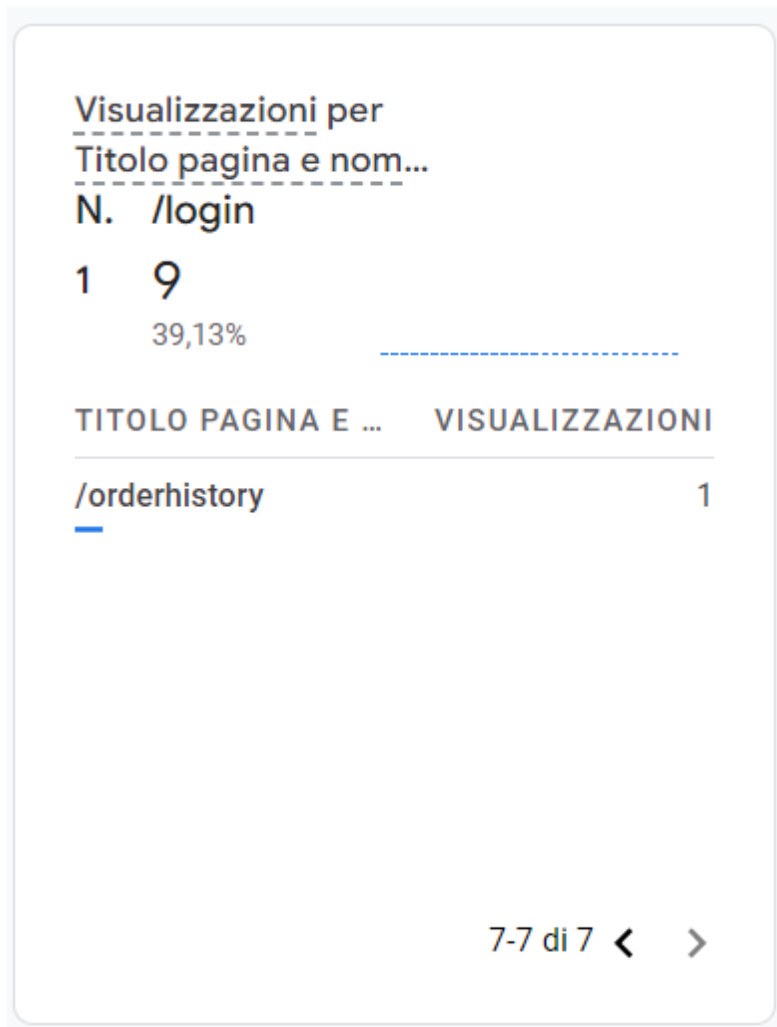


**Grafico delle sessioni:** questo grafico mostra il numero di sessioni totali dell'applicazione nel periodo di tempo selezionato. Una sessione viene definita come un periodo di attività continuativa dell'utente nell'applicazione.



**Grafico degli utenti attivi:** questo grafico mostra il numero di utenti unici che hanno interagito con l'applicazione durante il periodo di tempo selezionato.





**Grafico delle schermate:** questo grafico mostra il numero di visualizzazioni di schermate dell'applicazione durante il periodo di tempo selezionato. Può aiutare gli sviluppatori a identificare le schermate più popolari e quelle meno utilizzate.

### Conteggio eventi per Nome evento

N. screen\_view

1 18

22,22%

NOME EVENTO	CONTEGGIO EVENTI
<a href="#">is_already_login</a>	3
<a href="#">user_engagement</a>	3
<a href="#">check_first_login</a>	2
<a href="#">is_valid_token</a>	2
<a href="#">new_order</a>	2
<a href="#">login</a>	1

7-12 di 13 < >

### Conteggio eventi per Nome evento

N. screen\_view

1 18

22,22%

NOME EVENTO	CONTEGGIO EVENTI
<a href="#">screen_view</a>	18
<a href="#">get_categories</a>	14
<a href="#">add_item_to_order</a>	13
<a href="#">get_selected_items</a>	11
<a href="#">get_orders</a>	7
<a href="#">update_order</a>	4

1-6 di 13 < >

**Grafico degli eventi:** questo grafico mostra il numero di eventi specifici che si sono verificati nell'applicazione durante il periodo di tempo selezionato. Gli eventi possono essere qualsiasi azione dell'utente che l'utente effettua nell'applicazione,

Da questi grafici abbiamo evidenziato, un afflusso di Utenti nella città di Napoli e un grande uso delle funzionalità di “aggiungi elemento all’ordine”.

## FIREBASE CRASHLYTICS

Firebase Crashlytics rileva automaticamente quando l'applicazione si blocca o crasha e raccoglie informazioni sulle circostanze del crash.

### *Versione Alpha pre-release*

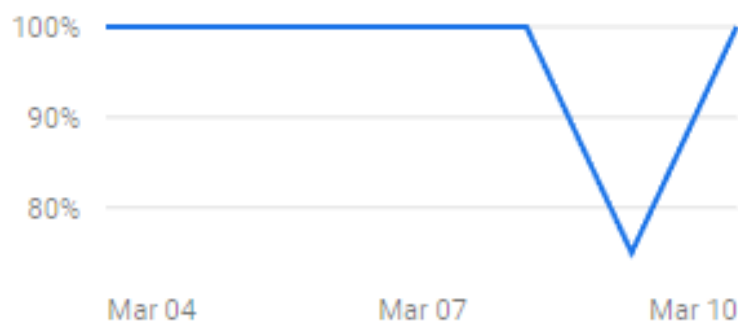
La prima fase di analisi è stata quella, pre-rilascio infatti noi sviluppatori abbiamo testato attentamente l'app e abbiamo riscontrato un solo caso di crash che ci ha portato a risolvere alcuni errori.

Crashlytics ci ha riportato lo stack trace dell'errore che abbiamo prontamente risolto e dopo un'altra fase di testing abbiamo appurato la stabilità dell'applicazione.

## Statistiche senza arresti anomali

Utenti senza arresti anomali ⓘ

91,67%



Fatal Exception: java.lang.RuntimeException

Test Crash

com.example.natour21.View.SplashScreen.lambda\$onCreate\$0 (SplashScreen.java:46)

com.example.natour21.View.SplashScreen\$\$ExternalSyntheticLambda0.run

android.os.Handler.handleCallback (Handler.java:938)

android.os.Handler.dispatchMessage (Handler.java:99)

android.os.Looper.loop (Looper.java:246)

android.app.ActivityThread.main (ActivityThread.java:8653)

java.lang.reflect.Method.invoke (Method.java)

com.android.internal.os.RuntimeInit\$MethodAndArgsCaller.run (RuntimeInit.java:602)

com.android.internal.os.ZygoteInit.main (ZygoteInit.java:1130)

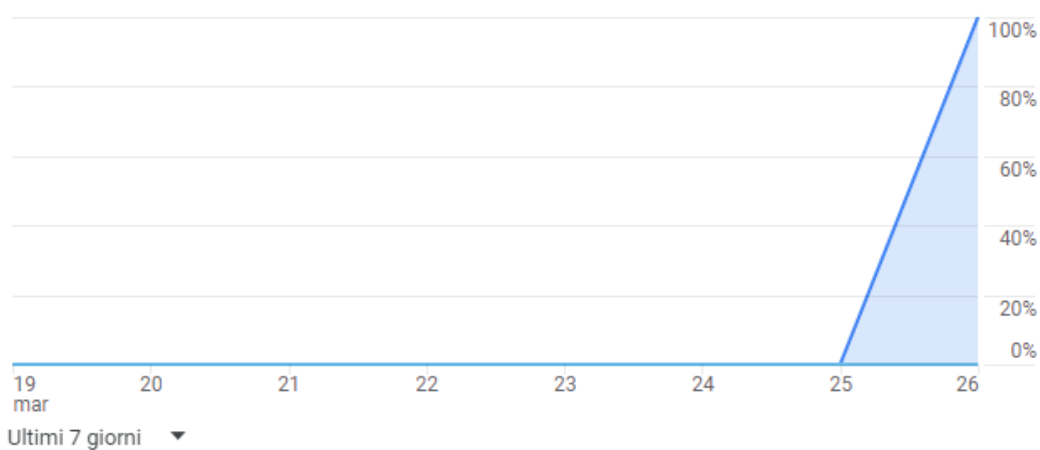
## Prima versione e fase post-relase

Ultimo aggiornamento dello stato: 02:04:02 GMT+1

### Versione 1.0.0: **Versione che funziona correttamente**

100% di utenti attivi e 100% di utenti che non hanno sperimentato arresti anomali

Utilizzo di più versioni dell'app ⓘ



Ultimi 7 giorni ▼

1.0.0 ⓘ

Stato: **Versione che funziona correttamente**

Data stimata per il lancio: 26 mar 2023 (oggi)



% di utenti attivi  
100%

Coinvolgimento per sessione  
5 m 02 s

[Visualizza i dettagli per la release](#) →

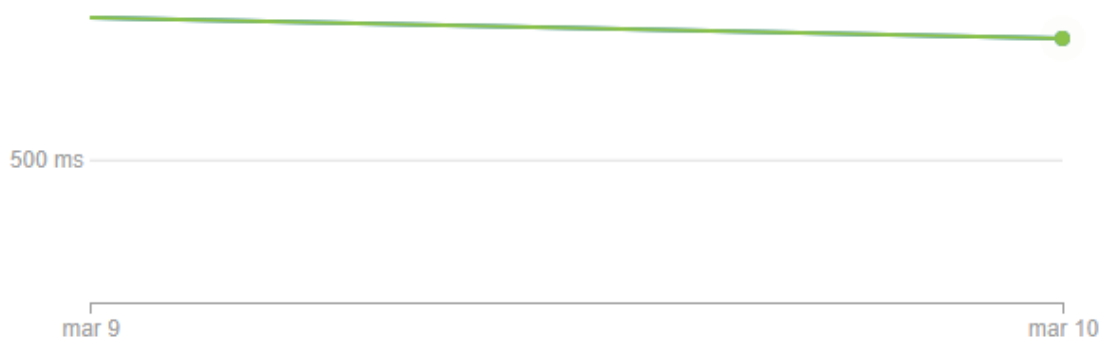
Utenti che non hanno sperimentato arresti anomali	Arresti anomali ogni 10.000 sessioni
100%	0

[Visualizza i dettagli degli arresti anomali](#) →

Crashlytics continuerà a raccogliere le informazioni durante la fase post-release, gli sviluppatori possono utilizzare Crashlytics per monitorare il tasso di crash dell'applicazione e identificare eventuali problemi di stabilità che si verificano tra gli utenti. Inoltre, Crashlytics fornisce anche report sulle prestazioni dell'applicazione, come il tempo di risposta dell'interfaccia utente e il tempo di caricamento delle schermate, che possono essere utili per migliorare l'esperienza degli utenti.

Tempo di avvio dell'app

**Tempo di avvio dell'app (924 ms) è 7% più veloce rispetto a 1 giorno prima**



Queste informazioni possono essere utilizzate per identificare eventuali problemi di stabilità dell'applicazione e pianificare gli interventi correttivi necessari.

### *Metodo euristico*

Per ciò che riguarda la valutazione euristica, abbiamo individuato 3 esperti a cui sottoporre sia qualche domanda, con particolare attenzione alle 8 regole d'oro di **Shneiderman**, così da trovare dei difetti.

Le domande sottoposte ai valutatori sono le seguenti:

Quali sono i punti di forza e le debolezze dell'interfaccia utente dell'app?

Secondo la vostra esperienza, gli utenti possono facilmente capire come usare l'app? Ci sono degli ostacoli o delle difficoltà che dovremmo risolvere?



L'app rispetta le norme e le convenzioni del settore? Ci sono delle buone pratiche che potremmo implementare per migliorare l'esperienza utente?

Quali sono i problemi più critici riscontrati durante l'analisi euristica? Quali problemi dovrebbero essere affrontati prioritariamente?

Quali sono i suggerimenti specifici che potrebbero essere utili per migliorare l'usabilità dell'app?

Ci sono eventuali problemi di accessibilità o di usabilità per utenti con disabilità che dovrebbero essere affrontati?

Le risposte dei tre esperti sono state molto chiare:

L'interfaccia utente dell'app è molto pulita e ben organizzata.

Il design minimalista è uno dei suoi punti di forza. Tuttavia, abbiamo notato che la navigazione tra le pagine non è sempre intuitiva per gli utenti."

In generale, l'app è abbastanza facile da usare. Tuttavia, gli utenti potrebbero avere difficoltà a trovare alcune funzionalità avanzate. Abbiamo suggerito di aggiungere icone o descrizioni più chiare per migliorare l'usabilità."

L'app rispetta molte delle convenzioni del settore. Tuttavia, abbiamo suggerito di utilizzare icone più comuni e di adottare la terminologia standard per alcune funzionalità per rendere l'app ancora più intuitiva per gli utenti."

Il principale problema riscontrato durante l'analisi euristica è stato la mancanza di feedback visivi per alcune azioni eseguite dagli utenti. Questo ha portato a una certa confusione e frustrazione. Dovrebbe essere una delle priorità per la correzione."

Abbiamo suggerito di aggiungere alcune animazioni per rendere l'esperienza utente più coinvolgente."

"Non abbiamo riscontrato problemi di accessibilità o usabilità per utenti con disabilità."

Nel complesso, la valutazione euristica ha permesso di individuare alcune aree di miglioramento per l'interfaccia utente dell'app, che potrebbero contribuire a garantire un'esperienza utente più soddisfacente e coinvolgente.