

UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
FACOLTÀ DI INFORMATICA



Ingegneria del Software 2022/2023

Ratatouille23

Docenti:

Prof. Sergio Di Martino

Prof. Francesco Cutugno

Prof. Luigi Lucio Libero Starace



Gruppo INGSW2223_N_04

| No. | Nome e Cognome | Matricola |
|-----|-----------------|-----------|
| 1 | Luigi Vessella | N86003354 |
| 2 | Matteo Marino | N86003963 |
| 3 | Biagio Speranza | N86002964 |



Indice

| | | |
|----------|--|----------|
| 1 | Descrizione del progetto | 4 |
| 2 | Documento dei requisiti software | 5 |
| 2.1 | Individuazione target d'utenti | 5 |
| 2.2 | Requisiti Funzionali | 5 |
| 2.2.1 | Admin | 5 |
| 2.2.2 | Cameriere | 8 |
| 2.2.3 | Cucina | 8 |
| 2.2.4 | Supervisore | 9 |
| 2.2.5 | Tutti | 9 |
| 2.3 | Requisiti Non Funzionali | 11 |
| 2.4 | Requisiti di Dominio | 11 |
| 2.4.1 | Tabelle di Cockburn | 12 |
| 2.5 | Valutazione dell'usabilità | 19 |
| 2.6 | Individuazione target d'utenti | 20 |
| 2.7 | Glossario | 21 |
| 2.8 | Architettura del sistema | 22 |
| 2.9 | Introduzione | 22 |
| 2.9.1 | Client | 22 |
| 2.9.2 | Server | 22 |
| 2.10 | Sequence diagram di design | 24 |
| 2.10.1 | 4 funzionalità | 24 |



1 Descrizione del progetto

Ratatouille23 è un sistema finalizzato al supporto alla gestione e all'operatività di attività di ristorazione. Il sistema consiste in un'applicazione performante e affidabile, attraverso cui gli utenti possono fruire delle funzionalità del sistema in modo intuitivo, rapido e piacevole. La nostra visione della richiesta prevede lo sviluppo di un'applicazione mobile su sistema operativo Android che offrirà agli utilizzatori i seguenti servizi:

- *I proprietari (o amministratori) potranno creare account per i dipendenti*
- *I proprietari saranno in grado di gestire uno o più ristoranti*
- *Si avrà la possibilità di visualizzare e/o modificare il menu*
- *I dipendenti (camerieri) saranno in grado di prendere e inoltrare le ordinazioni in cucina*
- *Gli addetti alla cucina potranno avvisare i camerieri nel momento in cui è pronta un'ordinazione*
- *Tutti potranno visionare lo storico delle ordinazioni con i dettagli*

Ovviamente, il tipo di funzionalità messo a disposizione dall'applicativo sarà cambiato dinamicamente a seconda di chi si logga. Gli amministratori e/o supervisor saranno inoltre dotati di tablet con a bordo l'OS di Google Android per una migliore fruizione della loro dashboard.

I dipendenti quali camerieri, operatori di cucina, capisala, saranno dotati di smartphone aziendali sempre con OS Android correttamente configurati per un'ottimale fruizione dell'applicazione.

Tutti i dispositivi dovranno essere in grado di accedere a internet, preferibilmente per tutta la durata del servizio. Il funzionamento del server è invece garantito servizi allo stato dell'arte quali Microsoft Azure.



2 Documento dei requisiti software

2.1 Individuazione target d'utenti

Durante la progettazione di un nuovo software è fondamentale definire qual'è il target di utenti a cui riferirsi. Da una prima analisi dei casi d'uso è possibile individuare per la nostra applicazione quattro diverse utenze:

- Admin (o Amministratore del sistema)
- Supervisore
- Addetto alla cucina
- Addetti alla sala (principalmente camerieri)

2.2 Requisiti Funzionali

Vengono qui presentati i requisiti funzionali dell'applicativo, ossia quei servizi che l'app deve offrire agli utenti:

2.2.1 Admin

| ID | Admin_1 |
|-------------|--|
| Nome | Registrazione account amministratore |
| Descrizione | Il sistema permette ad un amministratore non registrato di registrarsi alla piattaforma utilizzando: <i>nome, cognome, email, codice fiscale, P.IVA e password</i> |

| ID | Admin_2 |
|-------------|--|
| Nome | Modifica account amministratore |
| Descrizione | Il sistema permette ad un amministratore loggato di modificare i campi del proprio account |

| ID | Admin_3 |
|-------------|--|
| Nome | Registrazione dei dipendenti |
| Descrizione | Il sistema permette ad un amministratore di creare utenze per i dipendenti non registrati del ristorante, specificandone <i>nome, cognome, email e ruolo</i> |



| | |
|--------------------|---|
| ID | Admin_4 |
| Nome | Modificare/Eliminare account dipendenti |
| Descrizione | Il sistema permette ad un amministratore loggato di modificare gli account dei propri dipendenti. |

| | |
|--------------------|---|
| ID | Admin_5 |
| Nome | Aggiungere personale della cucina |
| Descrizione | Il sistema permette ad un amministratore loggato di aggiungere al proprio ristorante il personale della cucina. |

| | |
|--------------------|---|
| ID | Admin_6 |
| Nome | Aggiunta dei Ristoranti |
| Descrizione | Il sistema permette ad un amministratore di poter aggiungere le proprie attività di ristorazione (CAMPI DA DEFINIRE). |

| | |
|--------------------|--|
| ID | Admin_7 |
| Nome | Modifica/Eliminazione dei Ristoranti |
| Descrizione | Il sistema permette ad un amministratore di poter modificare ed eliminare le proprie attività di ristorazione del sistema. |

| | |
|--------------------|---|
| ID | Admin_8 |
| Nome | Modifica dati dei Dipendenti |
| Descrizione | Il sistema permette ad un amministratore loggato di poter cambiare i dati personali dei dipendenti (nome, cognome, email, luogo). |



| | |
|--------------------|--|
| ID | Admin_9 |
| Nome | Aggiungere/Modificare elementi nel menù |
| Descrizione | Il sistema permette ad un amministratore o supervisore dell'attività di ristorazione di aggiungere/modificare elementi nel menù dell'attività. Ogni elemento dovrà avere i seguenti campi: Nome, Costo, Descrizione, Elenco di Allergeni, Categoria/e |

| | |
|--------------------|---|
| ID | Admin_10 |
| Nome | Modifica dati personali |
| Descrizione | Il sistema permette ad un amministratore loggato di poter cambiare i propri dati personali. |

| | |
|--------------------|---|
| ID | Admin_12 |
| Nome | Tradurre il menu |
| Descrizione | Il sistema permette ad un Amministratore di poter tradurre gli elementi del proprio menù in un'altra lingua |

| | |
|--------------------|--|
| ID | Admin_13 |
| Nome | Visualizza statistiche personale della cucina |
| Descrizione | Il sistema permette ad un Amministratore di visualizzare, grazie anche all'ausilio di grafici interattivi, informazioni sull'operato degli addetti alla cucina |

| | |
|--------------------|--|
| ID | Admin_14 |
| Nome | Modifica menu |
| Descrizione | Il sistema permette ad un Amministratore di poter aggiornare (aggiungere, modificare ed eliminare elementi) il menu del ristorante |



2.2.2 Cameriere

| | |
|--------------------|---|
| ID | Waiter_1 |
| Nome | Prendere le ordinazioni |
| Descrizione | Il sistema permette ai camerieri di prendere ordinazioni ai tavoli, inoltrandole alla cucina. |

| | |
|--------------------|---|
| ID | Waiter_2 |
| Nome | Gestione delle ordinazioni |
| Descrizione | Il sistema permette ai camerieri di prendere ordinazioni ai tavoli, inoltrandole alla cucina. |

| | |
|--------------------|--|
| ID | Waiter_3 |
| Nome | Evasione degli ordini |
| Descrizione | Il sistema permette a un Cameriere di marcare i singoli elementi di un ordine come conclusi, aggiornando gli addetti in cucina |

| | |
|--------------------|---|
| ID | Waiter_4 |
| Nome | Sollecitare la cucina |
| Descrizione | Il sistema permette a un Cameriere di sollecitare la cucina nel caso in cui un ordine sia da troppo tempo in preparazione |

2.2.3 Cucina

| | |
|--------------------|--|
| ID | Kitchen_1 |
| Nome | Marcare gli ordini pronti |
| Descrizione | Il sistema permette alla cucina di marcare gli ordini pronti alla "consegna ", specificando lo chef che l'ha preparato |

| | |
|--------------------|--|
| ID | Kitchen_2 |
| Nome | Sollecitare i camerieri |
| Descrizione | Il sistema permette alla cucina di notificare i camerieri nel caso in cui un ordine sia pronto alla consegna da troppo tempo |



2.2.4 Supervisore

| | |
|--------------------|---|
| ID | Hypervisor_1 |
| Nome | Avvisare il personale |
| Descrizione | Il sistema permette ad un supervisore ed un amministratore di inviare degli avvisi al personale |

| | |
|--------------------|---|
| ID | Hypervisor_2 |
| Nome | Visualizzare stato ordini per tavolo |
| Descrizione | Il sistema permette ad un Supervisore ed un Cameriere di visualizzare gli stati delle ordinazioni per ogni tavolo |

| | |
|--------------------|--|
| ID | Hypervisor_3 |
| Nome | Visualizzare ordini in arrivo |
| Descrizione | Il sistema permette ad un Supervisore e alla Cucina di visualizzare l'elenco di ordini in arrivo |

| | |
|--------------------|--|
| ID | Hypervisor_4 |
| Nome | Visualizzare ordini in uscita |
| Descrizione | Il sistema permette ad un Supervisore e alla Cucina di visualizzare l'elenco di ordini pronti all'uscita |

| | |
|--------------------|---|
| ID | Hypervisor_5 |
| Nome | Visualizzare storico ordini |
| Descrizione | Il sistema permette ad un Supervisore e alla Cucina di visualizzare lo storico degli ordini |

2.2.5 Tutti

| | |
|--------------------|---|
| ID | All_1 |
| Nome | Recupero/Cambio Password |
| Descrizione | Il sistema permette a tutti gli utenti registrati di poter recuperare la password e di poterla modificare |



| ID | All_2 |
|-------------|--|
| Nome | Login |
| Descrizione | Il sistema permette a tutti gli utenti registrati di poter effettuare il login con Username e Password all'interno della piattaforma |

| ID | All_3 |
|-------------|---|
| Nome | Visualizzare un avviso o sollecitazione |
| Descrizione | Il sistema permette a tutti gli utenti registrati di poter visualizzare gli avvisi e le sollecitazioni ricevuti, marcandoli come visualizzati |



2.3 Requisiti Non Funzionali

Vengono qui elencati i requisiti non funzionali dell'applicativo:

| ID | Unfunctional_1 |
|-------------|---|
| Nome | Policy first password |
| Descrizione | Il sistema richiede al primo accesso di un dipendente il cambio della password provvisoria in una password personale. |

| ID | Unfunctional_2 |
|-------------|--|
| Nome | Verifica esistenza della mail |
| Descrizione | Al fine di evitare registrazioni con e-mail fittizie, il sistema richiede l'autenticazione della mail mediante codice di verifica per poter procedere con il completamento della registrazione |

| ID | Unfunctional_3 |
|-------------|--|
| Nome | Password Strength |
| Descrizione | Al fine di evitare la creazione di password poco sicure, il sistema impone all'utente di utilizzare una password di almeno 8 caratteri che contenga numeri e caratteri speciali. |

| ID | Unfunctional_4 |
|-------------|---|
| Nome | Unique Commercial Code |
| Descrizione | Una P.IVA appartiene ad un solo amministratore. |

2.4 Requisiti di Dominio

| ID | Domain_1 |
|-------------|--|
| Nome | GDPR |
| Descrizione | Il sistema deve essere conforme alla normativa GDPR (Regolamento Generale sulla Protezione dei Dati), per il trattamento dei dati personali e riguardante la privacy dell'utente |



2.4.1 Tabelle di Cockburn

Vengono qui riportate le tabelle di cockburn dei casi d'uso *Aggiungi ristorante*, *Aggiungi piatto*, *Prendi ordinazione*, *Visualizza avvisi*.

Tabella 1:

| Use Case #1 | Aggiunta ristorante | | |
|---|---|---|--|
| Goal in Context | Un admin (proprietario di uno o più ristoranti) vuole aggiungere uno di questi nell'app Ratatuille. | | |
| Precodition | Il proprietario deve essere registrato e loggato nell'app come amministratore | | |
| Success End Condition | Il proprietario aggiunge correttamente il ristorante | | |
| Failed End Condition | Il proprietario non riesce ad aggiungere il ristorante | | |
| Primary Actor | Amministratore | | |
| Trigger | Preme il pulsante <i>Aggiungi</i> | | |
| Description | Step | UserAction | System |
| | 1 | L'amministratore preme sul tasto " <i>AGGIUNGI RISTORANTE</i> " sulla schermata <i>M04</i> | |
| | 2 | | Mostra la schermata <i>M05</i> |
| | 3 | Compila i campi necessari per la registrazione del proprio ristorante e preme sul tasto " <i>SALVA</i> " per aggiungere il Ristorante | |
| | 4 | | Ricarica la schermata <i>M04</i> aggiungendo nella lista dei ristoranti l'ultimo appena inserito |
| Extension # 1 L'amministratore non fa nulla e torna indietro | Step | UserAction | System |
| | 4a | | Torna alla schermata <i>M04</i> |

Continued on next page



Tabella 1: (Continued)

| Use Case #1 | Aggiunta ristorante | | |
|--|---------------------|------------|---|
| Extension #2 L'amministratore ha aggiunto un ristorante già presenta nella propria lista di ristoranti | Step | UserAction | System |
| | 4b | | Mostra uno dei messaggi di errore della schermata M0boh restando allo step 2 dello scenario principale |
| Extension #3 L'amministratore ha lasciato uno o più campi vuoti | Step | UserAction | System |
| | 4c | | Mostra uno o più dei messaggi di errore della schermata M05 restando allo step 2 dello scenario principale |
| Extension #4 L'amministratore ha aggiunto un nome troppo corto | Step | UserAction | System |
| | 4d | | Mostra il messaggio di errore di fianco al campo "Nome" della schermata M05 restando allo step 2 dello scenario principale |
| Extension #5 L'amministratore ha aggiunto un numero di coperti non valido | Step | UserAction | System |
| | 4e | | Mostra il messaggio di errore di fianco al campo "Numero di coperti" della schermata M05 restando allo step 2 dello scenario principale |
| Extension #6 L'amministratore ha aggiunto un indirizzo troppo corto | Step | UserAction | System |
| | 4f | | Mostra il messaggio di errore di fianco al campo "Indirizzo" della schermata M05 restando allo step 2 dello scenario principale |

Continued on next page



Tabella 1: (Continued)

| Use Case #1 | Aggiunta ristorante | | |
|---|---------------------|---------------|--|
| Extension #7 L'amministratore ha aggiunto un numero di telefono non valido | Step | UserAction | System |
| | 4g | | Mostra il messaggio di errore di fianco al campo "Numero di telefono" della schermata M05 restando allo step 2 dello scenario principale |
| Subvariation #1 L'amministratore torna indietro completando solo parzialmente i campi | Step | UserAction | System |
| | 1 | | Mostra la schermata M0boh |
| | 2 | Preme su "SI" | |
| | 3 | | Torna alla schermata M04 |
| Subvariation #2 L'amministratore inizialmente vuole tornare indietro completando solo parzialmente i campi ma cambia idea | Step | UserAction | System |
| | 1 | | Mostra la schermata M0boh |
| | 2 | Preme su "NO" | |
| | 3 | | Torna alla schermata M05 allo step 2 dello scenario principale |



Tabella 2:

| Use Case #2 | Prendere un ordine | | |
|---|---|--|---|
| Goal in Context | Un cameriere vuole prendere l'ordinazione di un tavolo | | |
| Precodition | Il cameriere deve essere registrato e loggato nell'app e il ristorante deve avere un menu con dei piatti al suo interno | | |
| Success End Condition | Il cameriere prendere correttamente un'ordinazione | | |
| Failed End Condition | Il cameriere non prende l'ordinazione | | |
| Primary Actor | Cameriere | | |
| Trigger | Preme il pulsante <i>NUOVO ORDINE</i> | | |
| Description | Step | UserAction | System |
| | 1 | Il cameriere preme sul pulsante " <i>NUOVO ORDINE</i> " sulla schermata <i>M11</i> | |
| | 2 | | Mostra la schermata <i>M19</i> con i vari piatti del menu |
| | 3 | Il cameriere seleziona il numero del tavolo, inserisce le portate all'interno dell'ordine e preme il pulsante " <i>SALVA</i> " | |
| | 4 | | Torna alla schermata <i>M11</i> |
| Extension # 1 Il cameriere non fa nulla e torna indietro | Step | UserAction | System |
| | 4a | | Torna alla schermata <i>M11</i> |
| Subvariation #1 Il cameriere torna indietro dopo aver aggiunto dei piatti all'ordine senza salvare | Step | UserAction | System |
| | 1 | | Mostra la schermata <i>M0boh</i> |
| | 2 | Preme su " <i>SI</i> " | |
| | 3 | | Torna alla schermata <i>M11</i> |

Continued on next page



Tabella 2: (Continued)

| Use Case #2 | Prendere un ordine | | |
|---|--------------------|---------------|----------------------------------|
| Subvariation #2 L'amministratore inizialmente vuole tornare indietro completando solo parzialmente i campi ma cambia idea | Step | UserAction | System |
| | 1 | | Mostra la schermata M0boh |
| | 2 | Preme su "NO" | |
| | 3 | | Torna alla schermata M11 |

Tabella 3:

| Use Case #3 | Aggiungi piatto al menu | | |
|--|---|---|---------------------------------|
| Goal in Context | Un amministratore vuole aggiungere un piatto al proprio menu | | |
| Precodition | Il proprietario deve essere registrato e loggato nell'app come amministratore, deve aver aggiunto almeno un ristorante e deve averne creato un menu | | |
| Success End Condition | L'amministratore aggiunge correttamente un piatto al suo menu | | |
| Failed End Condition | L'amministratore non riesce ad aggiungere un piatto al suo ristorante | | |
| Primary Actor | Amministratore | | |
| Trigger | Preme il pulsante <i>Aggiungi prodotto</i> | | |
| Description | Step | UserAction | System |
| | 1 | L'amministratore preme sul pulsante <i>Aggiungi prodotto</i> sulla schermata M08 | |
| | 2 | | Mostra la schermata M18 |
| | 3 | L'amministratore compila il campi del piatto e preme sul pulsante <i>OK</i> | |
| | 4 | | Torna alla schermata M08 |
| | | | |
| Extension # 1 L'amministratore non fa nulla e torna indietro | Step | UserAction | System |
| | 4a | | Torna alla schermata M08 |

Continued on next page



Tabella 3: (Continued)

| Use Case #3 | Aggiungi piatto al menu | | |
|---|-------------------------|---|---|
| Extension # 2 L'amministratore compila solo parzialmente i campi e preme sul pulsante <i>OK</i> | Step | UserAction | System |
| | 4a | | Mostra l'errore nella schermata <i>M0boh</i> |
| Subvariation #1 L'amministratore vuole aggiungere un prodotto preconfezionato al proprio menu | Step | UserAction | System |
| | 1 | | Mostra la schermata <i>M18</i> |
| | 2 | L'amministratore scrive il nome (completo o parziale) di un prodotto e preme il pulsante <i>Cerca</i> | |
| | 3 | | Mostra i prodotti corrispondenti alla ricerca |
| | 4 | L'amministratore seleziona il prodotto desiderato e preme il pulsante <i>OK</i> | |
| | 5 | | Torna alla schermata <i>M08</i> |



Tabella 4:

| Use Case #4 | Visualizza avvisi | | |
|--|---|---|--|
| Goal in Context | Un dipendente vuole visualizzare un avviso | | |
| Precondition | Il dipendente deve essere registrato e loggato nell'app | | |
| Success End Condition | Il dipendente visualizza gli avvisi | | |
| Failed End Condition | Il dipendente non riesce a visualizzare i propri avvisi | | |
| Primary Actor | Dipendente | | |
| Trigger | Preme il pulsante <i>Avvisi</i> nella propria Dashboard | | |
| Description | Step | UserAction | System |
| | 1 | Il Dipendente preme sul pulsante <i>Avvisi</i> sulla schermata <i>M11</i> o <i>M09</i> o <i>M12</i> | |
| | 2 | | Mostra la schermata <i>M21</i> |
| Extension # 1 Non ci sono avvisi da mostrare | Step | UserAction | System |
| | 4a | | Mostra la schermata <i>M21</i> |
| Subvariation #1 Il Dipendente vuole marcare un avviso come visualizzato | Step | UserAction | System |
| | 1 | Il Dipendente scorre col dito sull'avviso | |
| | 2 | | Aggiorna la schermata <i>M21</i> eliminando l'avviso |



2.5 Valutazione dell'usabilità

Per la valutazione dell'usabilità del nostro applicativo a priori, cioè prima della fase di sviluppo vera e propria, abbiamo deciso di imporci come linee guida le euristiche di Nielsen. Ne sono 10, ma vorremmo richiamare l'attenzione su alcune di esse nello specifico:

- *Visibilità dello stato del sistema.* Il sistema presentava una discreta mancanza di feedback, che prevediamo di colmare con elementi quali Dialog, Toast, AlertDialog e SnackBar.
- *Prevenzione degli errori.* Il sistema reagisce in maniera controllata e predeterminata alle situazioni di errori che gli utenti possono causare. Nulla è lasciato al caso, ed è, nelle build provate dal team, gestita qualsiasi azione eseguibile dagli utenti.
- *Riconoscere piuttosto che ricordare.* La nostra app è dotata di sezioni ben distinte, interfacce dinamiche a seconda del tipo di utente che le usa, e icone e testi esplicativi dell'azione che si va a intraprendere.
- *Guida e documentazione.* E' presente un simpatico topolino (che richiama il logo dell'app) che consiglia tramite vignette e piccoli dialoghi le azioni che si possono intraprendere. Inoltre ci sono piccole note come campi obbligatori ecc.

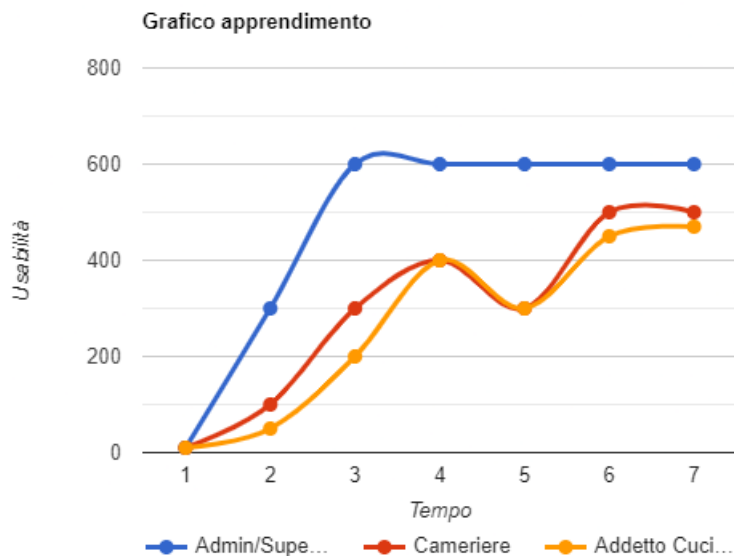


Figura 2.1: **Grafico:** Usabilità



2.6 Individuazione target d'utenti

La conoscenza dell'utente finale è di importanza fondamentale per chi progetti sistemi software di questo tipo. La grande diversità degli esseri umani fa sì che, anche considerando compiti e contesti d'uso simili, un oggetto potrebbe risultare usabile per un certo utente e del tutto inusabile per un altro.

Sicuramente, in base alle richieste del committente abbiamo subito individuato 4 principali categorie di utenti utilizzatori dell' app:

Admin: Amministratore e proprietario del ristorante. Una persona che deve avere tutto sotto controllo, può gestire le sue attività nonché i dipendenti che ne fanno parte, aggiungerne di nuova e talvolta, purtroppo, eliminarli.

Supervisore: Dopo l'amministratore, è nella "gerarchia" da noi definita, la seconda persona con più funzioni disponibili in-app. Anch'esso dispone di una dashboard completa sullo stile dell'admin.

Cameriere/Addetto sala: Senza la figura del cameriere un'attività di ristorazione non va avanti. Sappiamo quant'è importante fornire a questi ultimi un applicativo funzionale, facile da usare e da apprendere: per questo la sua interfaccia è ottimizzata per un palmare o smartphone compatto.

Addetto Cucina: Riceve tutti gli ordini dai camerieri e li inoltra alla cucina. Anche lui dispone di un'interfaccia semplice e dinamica che perfettamente si adatta al suo ruolo nell'attività.

Tutto ciò è reso possibile da uno sviluppo che va incontro alle esigenze dei diversi utenti. La nostra interfaccia riconosce il tipo di utente che logga e cambia in base alle sue esigenze.

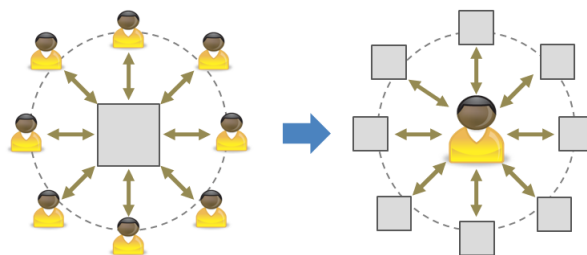


Figura: Da una visione centrata sul sistema a una visione centrata sull'utente



2.7 Glossario

In questa sezione vengono chiariti alcuni termini usati all'interno della documentazione, per rendere la lettura accessibile anche ai non esperti del settore.

- *Dialog*: Tipo di popup che il sistema Android mette a disposizione che mostra una finestra di dialog personalizzabile. Nella nostra app, mostra "Ok" e "Torna indietro".
- *Spring Boot*: Spring è uno strumento che permette a noi programmatori di usare il linguaggio di programmazione ad oggetti java per scrivere ottime app lato server.
- *JPA*: Le Java Persistence API, talvolta riferite come JPA, sono un framework per il linguaggio di programmazione Java che si occupa della gestione della persistenza dei dati di un DBMS relazionale
- *MVC*: Pattern Model-View-Controller, usato da Android Studio e da Spring Boot.



2.8 Architettura del sistema

2.9 Introduzione

Alla luce delle valutazioni effettuate durante l'analisi dei requisiti, abbiamo trovato conveniente strutturare la nostra app seguendo un tipo di architettura client-server. I client, nel nostro caso, sono quasi sempre dei dispositivi portatili dotati di sistema operativo android (con la possibilità volendo di utilizzare emulatori da pc desktop) mentre il server è stato realizzato con il framework di sviluppo Spring Boot.

2.9.1 Client

Come già detto, il client è composto da un'applicazione android distribuita o tramite apk o tramite play store (rispetta tutti i requisiti per essere pubblicata) che mira ad offrire un'interfaccia utente semplice e pulita per l'accesso agli endpoint forniti dal nostro server Spring-Boot. Come ogni app progetto nativo Android, il nostro client adotta un modello di design MVC (Model-View-Controller) e in contemporanea ad un pattern Singleton che tiene traccia dell'utente loggato e si tiene aggiornato quasi in tempo reale con i dati forniti dal server.

2.9.2 Server

Il back-end della nostra applicazione è stato realizzato con la tecnologia offerta da Spring Boot, un potente framework di sviluppo che sfrutta Java. L'architettura che prevede è anch'essa a 3 livelli e prevede i seguenti componenti:

- **Model:** L'insieme di classi che rappresentano le entità della nostra applicazione. Le stesse sono riportate sul client, e, cosa più importante, spring mappa le classi con l'annotation "@entity" 1:1 con le tabelle nel db relazionale.
- **Repository:** Sfruttando il framework JPA (Java Persistence API) riusciamo a gestire persistenza e consistenza dei dati nel db in maniera quasi automatizzata. Spring ci fornisce un gran supporto in questo ambito.
- **Controller:** Qui ci va tutta la logica di controllo del server. Sono queste le classi che espongono gli end-point Rest ai client e, comunicando con le repository, aggiornano i model e le tabelle nel db allo stesso tempo.

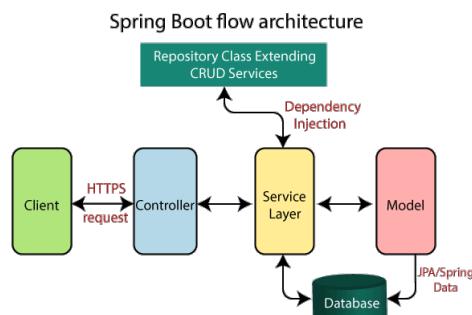


Figura: Spring-Boot



Microsoft Azure

Microsoft Azure è la piattaforma cloud pubblica di Microsoft, che offre servizi di cloud computing. Tra i vari piani che mette a disposizione, abbiamo usufruito di quello "gratuito" che ci permette di eseguire una macchina virtuale linux e un database relazionale, che nel nostro caso è stato PostgreSQL. La configurazione della macchina virtuale non ha richiesto particolari abilità: tramite connessione ssh abbiamo configurato il progetto Spring Boot e avviandolo mettiamo a disposizione gli end-point raggiungibili poi dai client. La potenza computazionale in questa fase di sviluppo e di utilizzo limitato dell'applicativo potrebbe risultare esile, ma il vantaggio di piattaforme quali Azure è possibile in qualsiasi momento aumentare le risorse dedicate a db e/o macchina virtuale, aumentando di fatto la scalabilità della nostra applicazione.



2.10 Sequence diagram di design

2.10.1 4 funzionalità

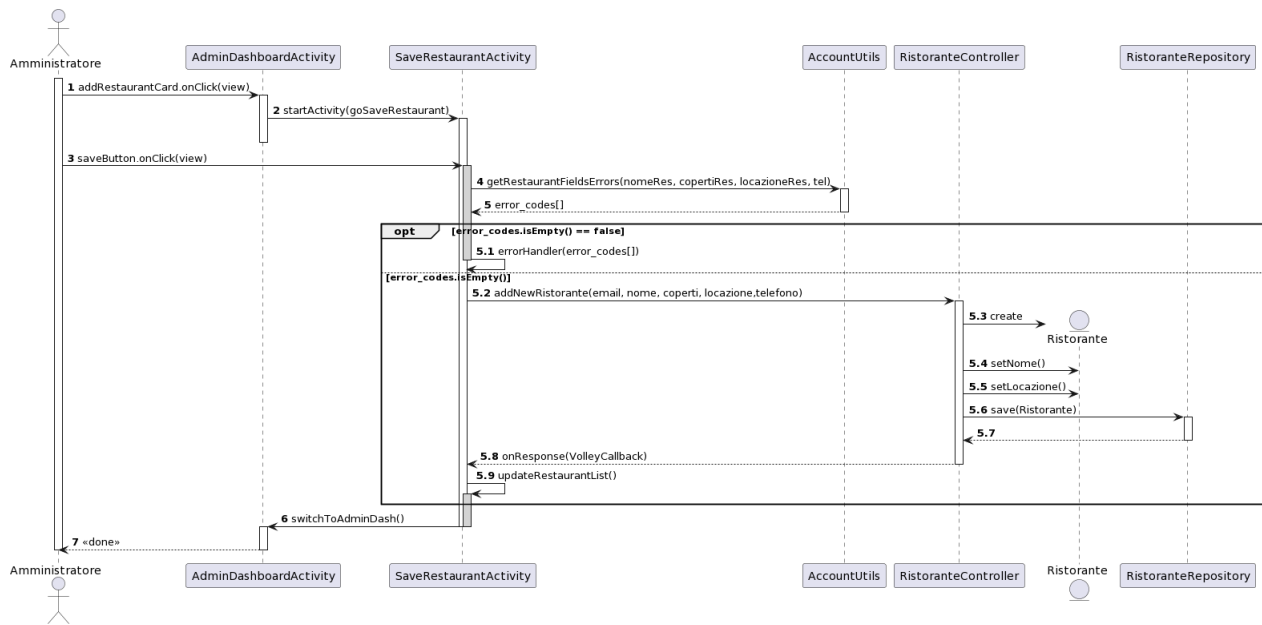


Figura 2.2: Sequence: Salva ristorante