Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica

Sviluppo di un'app mobile per la gestione dei pasti aziendali con controllo automatico delle presenze

Tesi di laurea

Relatore

Prof. Ombretta Gaggi

Laureando

Erica Cavaliere - 2013450

Anno Accademico 2022-2023

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

1.1 L'azienda

RiskApp S.r.l. (Figura 1.1) è un'azienda con sede a Conselve (PD) che si occupa di sviluppo software per il mondo assicurativo.
È stata fondata nel 2016 e il suo *core business* è lo sviluppo e il mantenimento dell'omonima applicazione, che viene costantemente aggiornata ed estesa per garantire un prodotto che possa rispondere ad ogni esigenza.
Il principale punto di forza di questa piattaforma è quello di stimare le possibili perdite economiche di un'impresa attraverso un algoritmo proprietario che, anche attraverso l'uso dell'intelligenza artificiale, valuta il rischio raccogliendo e combinando una moltitudine di dati da diverse fonti.
Il personale aziendale lavora costantemente per migliorare i propri servizi, ragionando sui possibili problemi che l'utente e l'aziende possono andare incontro, fanno riunioni e call per capire come migliorare e ampliare la piattaforma, tutto svolto in un clima di calma e rispetto tra colleghi.

Figura 1.1: Logo dell'azienda RiskApp

1.2 L'idea

Per poter gestire le spese per i pasti, che preparano in azienda, è stato scelto di sviluppare un'app mobile che permetta di monitorare i versamenti degli utenti, scegliere il piatto del giorno da un menu condiviso e monitorare la *cassa comune*[g].
Deve essere gestita l'autenticazione di ogni utente, dividendo tra utente semplice e utente amministratore e permettere il controllo delle presenze in azienda durante i pranzi.
Ogni utente potrà aggiungere un piatto nel menu, proporre il pasto del giorno, monito-

rare la sua *quota stornata*[g]e la cassa comune, indicare le spese effettuate e modificare i dati personali.
L'amministratore potrà anche gestire le presenze e le spese effettuate dagli stagisti.
L'applicazione dovrà essere sviluppata con *Flutter*[g], *Dart*[g]e *Firebase*[g].

1.3 Organizzazione del testo

Il secondo capitolo descrive in che modo è stato creato il prodotto desiderato, quale metodo di sviluppo è stato utilizzato e quali sono le tecnologie adottate per lavorare al progetto.

Il terzo capitolo approfondisce i requisiti con una analisi dettagliata di cosa è stato richiesto.

Il quarto capitolo approfondisce la progettazione, i *design pattern* utilizzati e la struttura del codice.

Nel quinto capitolo vengono riportate le valutazioni e le conclusioni personali del prodotto.

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

- gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;
- per la prima occorrenza dei termini riportati nel glossario viene utilizzata la seguente nomenclatura: *parola*[g];
- i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

Capitolo 2

Processi e metodologie

In questo capitolo viene spiegato il Material Design che sta alla base della progettazione dell'app, viene poi riportato il metodo di lavoro utilizzato e infine le tecnologie adottate per lo sviluppo del progetto.

2.1 Material Design

Alla base dell'applicazione, è stato scelto di seguire il Material Design (Figura 2.1) sviluppato da Google, che si concentra su un maggiore uso di *layout* basati su una griglia, animazioni, transizioni ed effetti di profondità come l'illuminazione e le ombre. Si tratta di una serie di regole ideate per consentire una buona *User Experience (UX)*[g]e definire una *User Interface (UI)*[g]per l'utente da implementare in ambiente Web, Android e in *Flutter*.
Viene annunciato per la prima volta da Google il 25 giugno del 2014 durante il Google I/O, una conferenza organizzata annualmente da Google a Mountain View, in California.

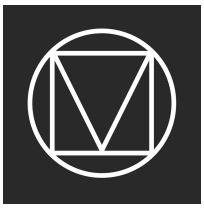

Figura 2.1: Logo del Material Design di Google

Venne rinnovato nel 2018 con il Material Design 2, anche chiamato Google Material Theme, introducendo un maggiore utilizzo di angoli arrotondati, spazi bianchi e icone colorate, infine viene rinnovato nel 2021 con il Material Design 3, oppure Material You, introducendo l'uso di tasti più grandi e maggiore uso delle animazioni.
Oggi viene ancora utilizzato il Material Design 3 ed è stato seguito per lo sviluppo dell'app dei pranzi.

Per consentire l'uso dei propri prodotti software a più utenti possibili, il Material

Design segue le regole del *Web Content Accessibility Guidelines (WCAG)*[g], mettendo alla base di ogni progetto l'accessibilità, creando così dei prodotti inclusivi, cioè usabili da tutti i tipi di utenti, anche con disabilità, consentendo a ciascuno un'esperienza fluida e semplice da usare.

I *layout* devono essere studiati in modo da guidare l'utente nella navigazione della pagina e devono essere dinamici, in modo che le pagine si adattino ad ogni tipo di schermo.
Vengono indicate delle regole precise su come devono essere impostate le *componenti*[g], come devono essere raggruppate, lo spazio che deve esserci e tanti altri piccoli ma importanti dettagli che lo sviluppatore deve considerare per permettere all'utente di orientarsi su qualsiasi dispositivo.
Anche *Flutter* offre una guida sulle *componenti* che mette a disposizione per lo sviluppatore e che sono state ideate per rispettare le regole di Material Design appena descritte.

2.2 Metodo di lavoro

Durante lo stage, RiskApp contava circa dieci dipendenti e ognuno era incaricato di sviluppare e mantenere una parte della loro piattaforma, confrontandosi tra loro ogni giorno per capire come continuare a lavorare.
Il loro metodo di lavoro si avvicina a un metodo Agile, più precisamente ad uno SCRUM, utilizzato anche per lo sviluppo del progetto di stage.

Il Manifesto per lo sviluppo Agile (*Manifesto Agile*. URL: https://agilemanifesto.org/iso/it/manifesto.html) è composto da dodici principi fondamentali che descrivono il modo in cui deve lavorare il team, permettendo possibili cambiamenti in corso d'opera e mettendo al primo posto il cliente, rilasciando varie versioni del prodotto funzionante dopo brevi periodi e privilegiando le comunicazioni faccia a faccia.
Lo SCRUM è un *framework* di gestione dei progetti Agile che mira a cinque valori fondamentali: impegno, focus, apertura, rispetto e coraggio.
Questo *framework* ha acquisito negli ultimi anni una straordinaria popolarità nel mondo dell'informatica grazie ai vantaggi offerti, come maggiore collaborazione con l'utente finale, il suo contributo al miglioramento continuo e la superiore gestione dei rischi.
L'idea di fondo consiste nel suddividere i periodi di lavoro in *sprint* di durata fissata, caratterizzati da un insieme di obiettivi da realizzare (*sprint backlog*).

Per lo sviluppo del progetto di stage, ogni giorno veniva riportato quanto era stato fatto e veniva mostrato il funzionamento, raccogliendo possibili idee per migliorare o modificare l'app.
Se in corso d'opera venivano incontrate eventuali problematiche sullo sviluppo, si ragionava su come affrontare o modificare il prodotto per risolvere questi problemi, permettendo così di soddisfare ogni esigenza degli utenti finali, in questo caso per soddisfare le esigenze dei dipendenti dell'azienda.

2.3 Tecnologie

2.3.1 Flutter

Flutter (Figura 2.2) è un progetto open-source di Google il cui vantaggio principale è la generazione di applicazioni multipiattaforma a partire da un unico codice sorgente. Permette quindi allo sviluppatore di concentrarsi sul prodotto da realizzare senza dover preferire un sistema operativo mobile ad un altro.
Per questo motivo è stato scelto di utilizzare Flutter come *framework* principale, dato che il prodotto finale deve funzionare sia per dispositivi Android sia per dispositivi iOS.

Figura 2.2: Logo di Flutter

2.3.2 Dart

Il linguaggio sul quale si basa Flutter è Dart (Figura 2.3), nato con l'intento di sostituire JavaScript come protagonista delle applicazioni web.
Tra i suoi pregi si elencano il compilatore JIT, migliore gestione della sicurezza, la velocità e la maggiore scalabilità.
Il paradigma principale è l'orientamento agli oggetti, una sua particolarità è data dalla sua attenzione alla *null safety*, per la quale nessun valore può essere nullo a meno che questa possibilità non sia esplicitamente dichiarata.

Figura 2.3: Logo di Dart

2.3.3 Firebase

Firebase (Figura 2.3) è una piattaforma *open-source* per la creazione di applicazioni per dispositivi mobili e web sviluppata da Google.
Firebase sfrutta l'infrastruttura di Google e il suo cloud per fornire una suite di strumenti per scrivere, analizzare e mantenere applicazioni *cross-platform*.
Infatti offre funzionalità come analisi, database (usando strutture noSQL), messaggistica e segnalazione di arresti anomali per la gestione di applicazioni web, iOS e Android.
Per lo sviluppo dell'app sono stati utilizzati:

- Firebase Autentication, per permettere la registrazione e l'autenticazione di un utente tramite mail e password;

- Cloud Firestore, per la gestione del database.

Figura 2.4: Logo di Firebase

2.3.4 Figma

Figma (Figura 2.5) è un software per la progettazione di User Interface(UI).
Permette infatti di realizzare prototipi delle interfacce, detti anche *mockup*, che permettono di illustrare il risultato finale che si desidera ottenere.
Questo strumento è stato utilizzato per mostrare e concordare l'interfaccia dell'app al tutor aziendale, prima della fase di codifica.

Figura 2.5: Logo di Figma

2.3.5 Android Studio

Android Studio (Figura 2.6) è un *Integrated Development Environment (IDE)*[g]adibito per la creazione di applicazioni Android e mette a disposizione dei simulatori virtuali di uno o più cellulari con il sistema operativo di Google.
Il progetto è stato sviluppato interamente con l'uso di questo *IDE* ed è stato utilizzato il simulatore virtuale di Google Pixel 7 con sistema operativo Android 13 per testare la *build*[g]dell'app.

Figura 2.6: Logo di Android Studio

2.3.6 Xcode

Xcode (Figura 2.7) è un *IDE* completamente sviluppato e mantenuto da Apple, contenente una suite di strumenti utili allo sviluppo di software per i sistemi macOS, iOS, iPadOS, watchOS e tvOS.
Per poter testare la *build* del progetto, è stato utilizzato il simulatore virtuale di iPhone 15 con sistema operativo iOS 17, messo a disposizione da questo software.

Figura 2.7: Logo di Xcode

2.3.7 GitHub

GitHub (Figura 2.8) è una piattaforma di *hosting* per per ospitare codice all'interno di repository basato sul software Git.
Fornisce agli sviluppatori strumenti per migliorare e mantenere il codice come:

- *features* utilizzabili da linea di comando;
- gestione delle *pull request* e *code review*;
- strumenti per l'*issue tracking*.

La codebase della piattaforma RiskApp è suddivisa in varie repository su GitHub.
Per questo progetto, l'azienda ha riservato una repository apposita per permettermi di lavorare in autonomia al codice.

Figura 2.8: Logo di GitHub

2.3.8 Slack

Slack (Figura 2.9) è un applicazione multipiattaforma per la messaggistica istantanea tra membri di un gruppo di lavoro.
Una delle funzioni di Slack è la possibilità di organizzare la comunicazione del team attraverso canali specifici, canali che possono essere accessibili a tutto il team o solo ad alcuni membri.
È possibile inoltre comunicare con il team anche attraverso chat individuali private o chat con due o più membri.
Questo software è stato utilizzato per comunicare con il tutor aziendale da remoto e per condividere materiale.

Figura 2.9: Logo di Slack

Capitolo 3

Analisi dei requisiti

3.1 Casi d'uso

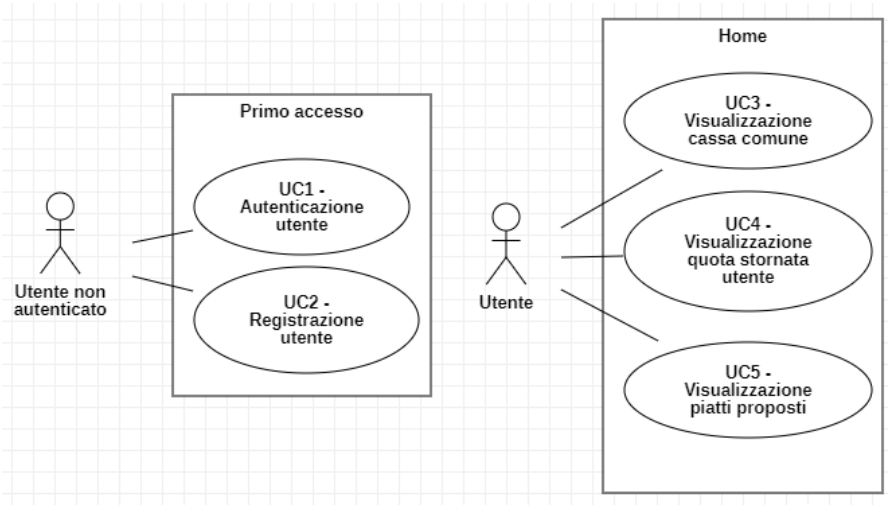

Figura 3.1: Use Case - Primo accesso e Home

UC1: Autenticazione utente

Attori Principali: Utente non autenticato.

Precondizioni: L'utente entra nell'app senza essersi autenticato.

Descrizione: L'utente inserisce la propria mail e la propria password per effettuare l'accesso.

Postcondizioni: Si visualizza la schermata Home dell'app.

UC2: Registrazione utente

Attori Principali: Utente non autenticato.

Precondizioni: L'utente entra nell'applicazione senza essersi mai registrato.

Descrizione: L'utente inserisce i propri dati per registrare i propri dati nel database e effettuare l'accesso.

Postcondizioni: L'utente visualizzerà la schermata Home dell'app.

UC3: Visualizzazione *cassa comune*

Attori Principali: Utente.

Precondizioni: L'utente non è ancora entrato nell'app o non ha ancora effettuato un accesso.

Descrizione: Viene effettuato l'accesso per entrare nell'app e visualizzare la *cassa comune*.

Postcondizioni: Si visualizza la *cassa comune*.

UC4: Visualizzazione *quota stornata* utente

Attori Principali: Utente.

Precondizioni: L'utente non è ancora entrato nell'app o non ha ancora effettuato un accesso.

Descrizione: Viene effettuato l'accesso per entrare nell'app e visualizzare la *quota stornata* dell'utente.

Postcondizioni: Si visualizza la *quota stornata* dell'utente.

UC5: Visualizzazione piatti proposti

Attori Principali: Utente.

Precondizioni: L'utente non è ancora entrato nell'app o non ha ancora effettuato un accesso.

Descrizione: Viene effettuato l'accesso per entrare nell'app e visualizzare i piatti proposti del giorno.

Postcondizioni: Si visualizza la lista dei piatti proposti oggi.

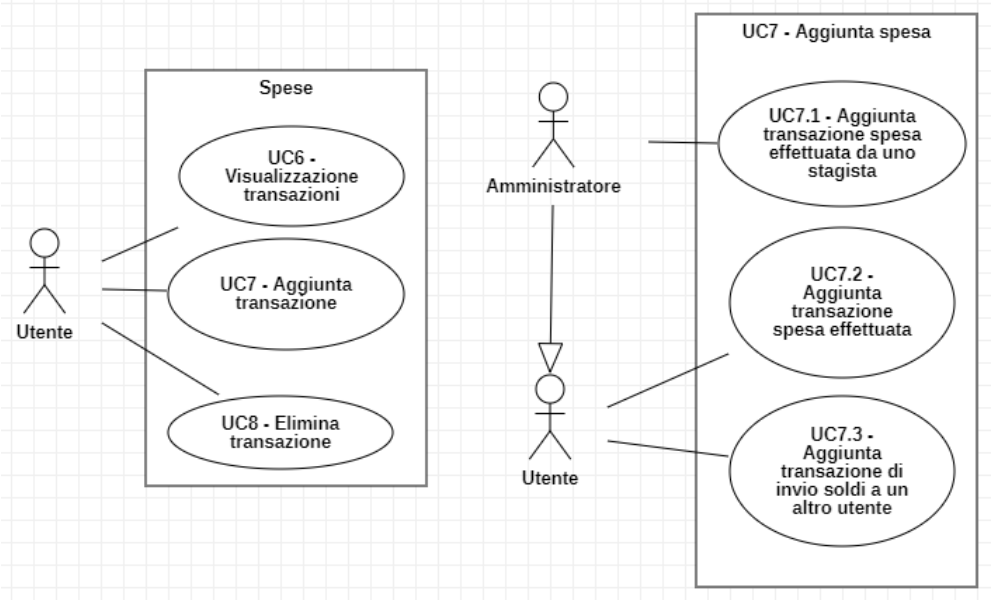

Figura 3.2: Use Case - Spese e UC7

UC6: Visualizzazione transazioni

Attori Principali: Utente.

Precondizioni: L'utente non è ancora entrato nell'app o non ha ancora effettuato un accesso.

Descrizione: Viene effettuato l'accesso per entrare nell'app e visualizzare la lista delle transazioni.

Postcondizioni: Si visualizza la lista delle transazioni.

UC7: Aggiunta transazione

Attori Principali: Utente.

Precondizioni: La transazione non è presente nel database e non è visibile nella lista delle transazioni.

Descrizione: L'utente inserisce i dati della transazione interessata e la salva nel database.

Postcondizioni: La transazione è presente nel database e visibile nella lista delle transazioni.

UC7.1: Aggiunta transazione spesa effettuata da uno stagista

Attori Principali: Amministratore.

Precondizioni: La spesa effettuata da un stagista non è presente nel database e non è visibile nella lista delle transazioni.

Descrizione: L'amministratore inserisce i dati della spesa effettuata da un stagista e

la salva nel database.

Postcondizioni: La spesa effettuata da un stagista è presente nel database e visibile nella lista delle transazioni.

UC7.2: Aggiunta transazione spesa effettuata

Attori Principali: Utente.

Precondizioni: La spesa effettuata dall'utente non è presente nel database e non è visibile nella lista delle transazioni.

Descrizione: L'utente inserisce i dati della spesa effettuata e la salva nel database.

Postcondizioni: La spesa dell'utente è presente nel database e visibile nella lista delle transazioni.

UC7.3: Aggiunta transazione di invio soldi a un altro utente

Attori Principali: Utente.

Precondizioni: L'invio dei soldi ad un altro utente non è presente nel database e non è visibile nella lista delle transazioni.

Descrizione: L'utente inserisce i dati dell'invio dei soldi e lo salva nel database.

Postcondizioni: L'invio dei soldi ad un altro utente è presente nel database e visibile nella lista delle transazioni.

UC8: Elimina transazione

Attori Principali: Utente.

Precondizioni: La transazione è presente nel database e visibile nella lista delle transazioni.

Descrizione: L'utente elimina la transazione interessata dall'app e viene eliminata dal database.

Postcondizioni: La transazione non è presente nel database e non è visibile nella lista delle transazioni.

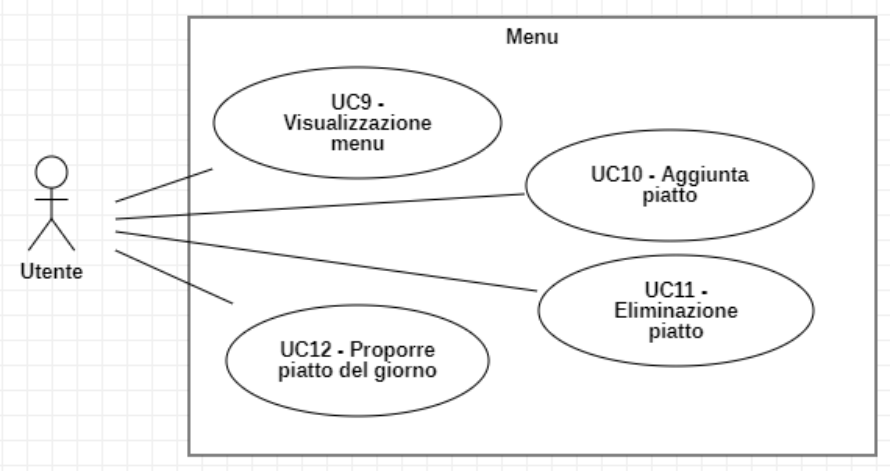

Figura 3.3: Use Case - Menu

UC6: Visualizzazione menu

Attori Principali: Utente.

Precondizioni: L'utente non è ancora entrato nell'app o non ha ancora effettuato un accesso.

Descrizione: Viene effettuato l'accesso per entrare nell'app e visualizzare il menu.

Postcondizioni: Si visualizza la lista dei piatti presenti nel menu.

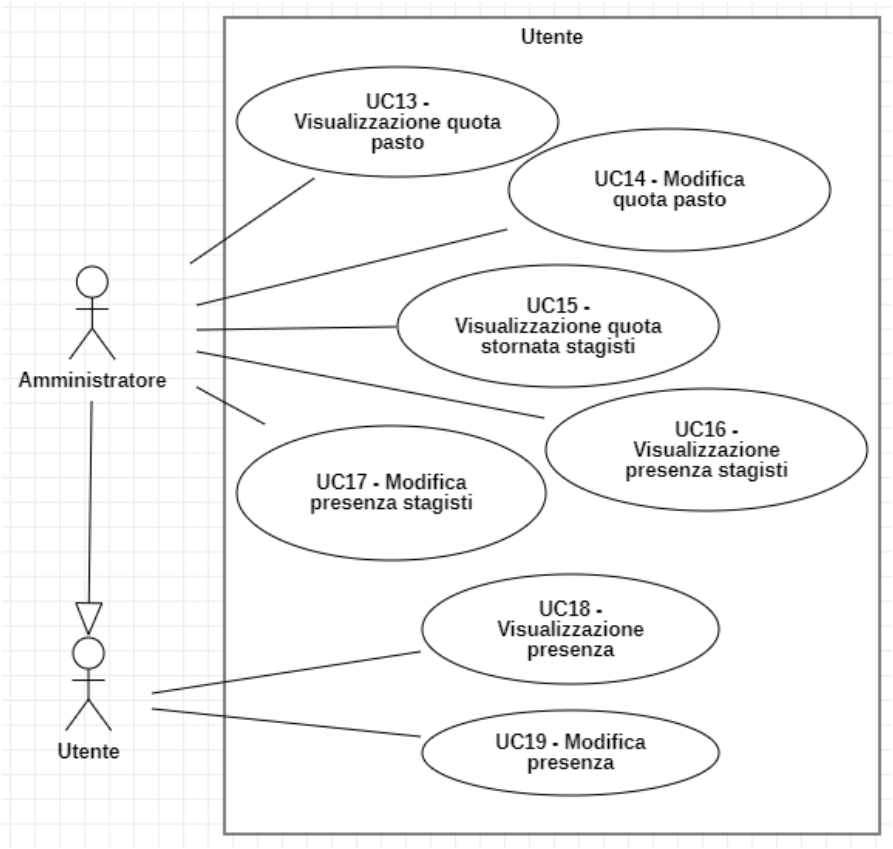

Figura 3.4: Use Case - Utente

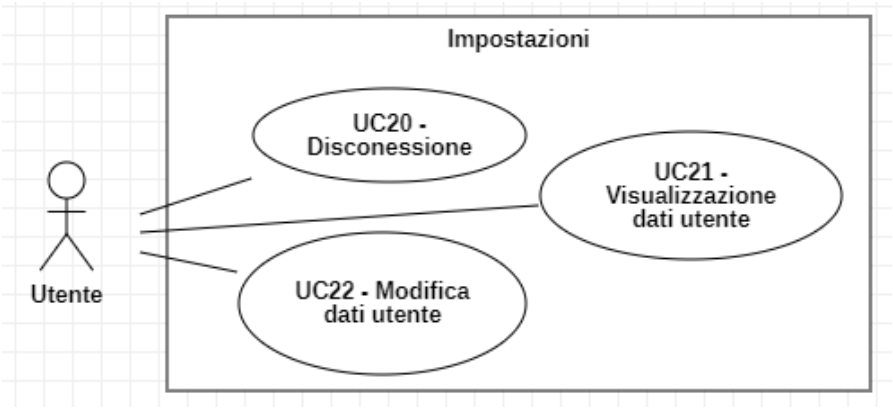

Figura 3.5: Use Case - Impostazioni

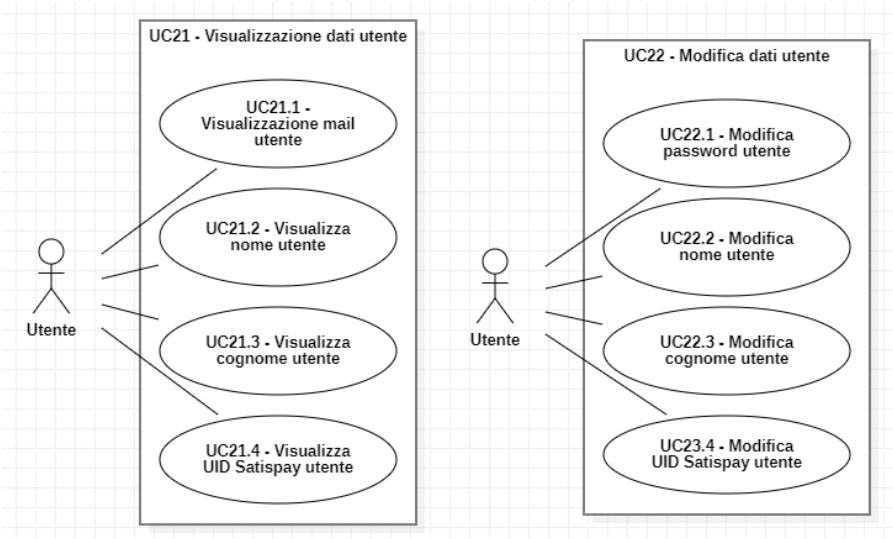

Figura 3.6: Use Case - UC21 e UC22

3.2 Tracciamento dei requisiti

Capitolo 4

Progettazione e codifica

4.1 Progettazione

4.2 Design Pattern utilizzati

4.3 Codifica

Capitolo 5

Conclusioni

Acronimi e abbreviazioni

IDE Integrated Development Environment. 6, 20, 21

UI User Interface. 3, 21

UML Unified Modeling Language. 21

UX User Experience. 3, 22

WCAG Web Content Accessibility Guidelines. 4, 22

Glossario

Build indica la trasformazione del codice in un prodotto software eseguibile. 6, 7

Cassa Comune viene utilizzato questo termine per indicare i fondi dati dagli operatori aziendali per coprire i pasti. 1, 10

Componenti sono un insieme di *widget* e di elementi che insieme costituiscono un prodotto software. 4

Dart linguaggio di programmazione *open-source* sviluppato da Google. È il linguaggio principale utilizzato per scrivere applicazioni con *Flutter*. Dart è noto per la sua velocità ed efficienza nella creazione di applicazioni mobili e web. Risulta inoltre staticamente tipizzato, cioè consente una dichiarazione esplicita dei tipi delle variabili e garantisce maggiore robustezza in programmazione. 2, 21

Firebase piattaforma di sviluppo di app mobile di Google che offre una serie di servizi tra cui *database* in tempo reale, autenticazione utente, *hosting* di applicazioni e molto altro. È ampiamente utilizzato per la costruzione di app mobile e web in modo rapido e scalabile, grazie alle funzionalità *cloud*, di notifica e di monitoraggio in *real time*. 2

Flutter *framework open-source* di Google per lo sviluppo di applicazioni mobile, desktop e webapp utilizzando il linguaggio *Dart*. È basato su *widget* personalizzabili, puntando su un rapido sviluppo, eccellenti performance, una comunità attiva e supporto per molte piattaforme. 2–4, 21

IDE è un ambiente di sviluppo integrato che supporta i programmatori nello sviluppo e nel *debug* del codice. 6, 7

Quota Stornata indica i soldi che il singolo utente deve dare o ricevere dagli altri utenti per i pasti effettuati e le spese sostenute. 2, 10

UI indica l'interfaccia grafica che viene utilizzata per le comunicazioni tra uomo e macchina. 20

UML in ingegneria del software *UML, Unified Modeling Language* (ing. linguaggio di modellazione unificato) è un linguaggio di modellazione e specifica basato sul paradigma object-oriented. L'*UML* svolge un'importantissima funzione di "lingua franca" nella comunità della progettazione e programmazione a oggetti. Gran parte della letteratura di settore usa tale linguaggio per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile a un vasto pubblico. 20

UX indica l'insieme di sensazioni e ricordi che una persona prova quando si rapporta con un prodotto, cioè tutti gli aspetti che condizionano il prodotto per consentire all'utente di utilizzarlo e capirlo con facilità. 20

WCAG si tratta di una serie di linee guida per l'accessibilità, fornisce una serie di criteri tecnici per rendere siti web, applicazioni e altri contenuti facilmente utilizzabili da tutti i tipi di utente. 20

Bibliografia

Riferimenti bibliografici

Ken Schwaber, Jeff Sutherland. *La Guida Scrum - La Guida Definitiva a Scrum: Le Regole del Gioco*. Novembre 2020.

Siti web consultati

Cloud Firestore. URL: https://firebase.flutter.dev/docs/firestore/usage/.

Figma Learn. URL: https://help.figma.com/hc/en-us.

Figma Tutorial. URL: https://help.figma.com/hc/en-us/sections/4405269443991-Figma-for-Beginners-tutorial-4-parts-.

Firebase Autentication. URL: https://firebase.flutter.dev/docs/auth/usage/.

Flutter Documentation. URL: https://docs.flutter.dev/.

Flutter Material. URL: https://docs.flutter.dev/ui/widgets/material.

FlutterFire. URL: https://firebase.flutter.dev/docs/overview/.

Manifesto Agile. URL: https://agilemanifesto.org/iso/it/manifesto.html (cit. a p. 4).

Material Design. URL: https://m3.material.io/.

WAI Standards Guidelines. URL: https://www.w3.org/WAI/standards-guidelines/wcag/.