Università degli Studi di Milano Bicocca
Scuola di Scienze
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di laurea in Informatica

Il ruolo dei LLM nello sviluppo di applicazioni mobile

Relatore: Micucci Daniela
Co-relatore: Rossi Maria Teresa

Relazione della prova finale di:
Simone Santin
Matricola 886116

Anno Accademico 2023-2024

A mia madre per avermi sostenuto in questo percorso e in tutti questi anni.

Ai miei nonni per essere stati sempre una fonte di ispirazione nella vita.

"All we have to decide is what to do with the time that is given us."

J.R.R Tolkien

ABSTRACT

Negli ultimi anni, i LLM (Large Language Models) hanno conosciuto un'ampia diffusione in svariati contesti. Sono utilizzati in chatbot e assistenti virtuali, traduttori automatici e per il supporto nello sviluppo software. L'obiettivo di questa tesi è quindi esplorare l'efficacia di ChatGPT nell'assistere un progettista nello sviluppo di un'applicazione mobile utilizzando Flutter.

Il progetto di tesi ha riguardato lo sviluppo di un'applicazione chiamata EcoSwap, un'applicazione in tema ambientale basata sullo scambio e noleggio di oggetti, utilizzando il supporto di ChatGPT durante le diverse fasi del ciclo di vita del software. L'obiettivo principale è stato valutare come ChatGPT possa assistere nella definizione dei requisiti, implementazione e testing finale, oltre a supportare l'apprendimento di Flutter e ulteriori tecnologie.

L'elaborato inizia con un'analisi dettagliata delle diverse fasi del progetto, inclusi i requisiti individuati e implementati, il processo di sviluppo e le tecnologie utilizzate, e le modalità di testing dell'applicazione. Successivamente, negli ultimi capitoli, viene condotta un'analisi approfondita del ruolo di ChatGPT in queste fasi, valutando la sua efficacia complessiva.

Indice

Introduzione

Negli ultimi anni, i modelli di linguaggio di grandi dimensioni (LLM) come ChatGPT [1] hanno rivoluzionato vari settori, offrendo soluzioni innovative e supporto in molteplici ambiti. Questi modelli, basati sull'intelligenza artificiale, sono in grado di comprendere e generare testo umano in maniera naturale, aprendo nuove possibilità per lo sviluppo software e il miglioramento dell'efficienza lavorativa.

Nel contesto dello sviluppo di applicazioni mobili, l'adozione di LLM può rappresentare un significativo vantaggio, facilitando diverse fasi del processo di sviluppo, dalla raccolta dei requisiti alla scrittura del codice, fino alla fase di testing.

Questo lavoro si propone quindi di investigare quanto effettivamente ChatGPT possa essere efficace e di supporto nel processo di sviluppo di un'applicazione mobile in un campo prescelto. In particolare, abbiamo sviluppato un'applicazione nel settore ambientale e documentato l'utilizzo di ChatGPT durante l'intero ciclo di sviluppo. L'obiettivo è valutare quindi l'utilità di ChatGPT nelle seguenti fasi:

1. **Analisi dei Requisiti**: Identificazione e definizione delle funzionalità richieste dall'applicazione.
2. **Implementazione**: Supporto nella progettazione e sviluppo dell'applicazione.
3. **Progetto di Testing**: Assistenza nella creazione e nell'esecuzione dei test per garantire la qualità del software.

L'applicazione EcoSwap, da noi sviluppata, mira a migliorare l'impatto ambientale degli utenti attraverso un sistema di scambio e noleggio di oggetti. Offrendo un'alternativa ecologica alle tradizionali app di acquisto e vendita, EcoSwap riduce così l'uso di plastica e carta di imballaggi, promuovendo uno stile di vita più sostenibile e meno consumista portato più allo scambio che all'acquisto.

. Le principali funzionalità previste sono:

- **Scambio di oggetti**: Gli utenti possono mostrare gli oggetti che desiderano scambiare sul proprio profilo, facilitando così il riutilizzo di beni tra membri della community.

- **Noleggio di oggetti**: È possibile noleggiare qualsiasi tipo di oggetto per un periodo determinato, effettuando il pagamento attraverso una piattaforma integrata.
- **Messaggistica**: Un sistema di chat integrata permette agli utenti di comunicare per definire i dettagli dello scambio o del noleggio.

Inoltre, abbiamo deciso di testare quanto ChatGPT possa facilitare lo sviluppo di un'applicazione anche per sviluppatori non familiari con un determinato linguaggio, scegliendo di utilizzare Flutter, un linguaggio di programmazione ad oggetti basato su Dart, che non avevamo mai utilizzato prima. In questo modo, sarà possibile determinare quanto ChatGPT possa agevolare l'apprendimento di un nuovo linguaggio di programmazione durante lo sviluppo.

Per questo lavoro abbiamo utilizzato GPT-3.5, senza ricorrere a versioni migliorate a pagamento. Ogni domanda posta a GPT-3.5 è stata salvata in un diario, annotando: la fase del ciclo di vita del progetto a cui apparteneva, il numero di rielaborazioni necessarie e l'utilità dell'output generato.

Nel complesso, possiamo dire che GPT-3.5 ci ha permesso di imparare facilmente un nuovo linguaggio di programmazione e di sviluppare un'applicazione perfettamente funzionante. Tuttavia, abbiamo riscontrato alcuni problemi durante lo sviluppo e nella generazione di risposte corrette. In questo elaborato esamineremo quindi più a fondo come abbiamo utilizzato GPT-3.5 e quali output sono stati forniti.

Capitolo 1

Analisi dei requisiti

1.1 Obiettivo

L’analisi dei requisiti è l’attività preliminare all’implementazione di un sistema software. L’obiettivo di questa analisi è definire in modo chiaro e dettagliato le funzionalità necessarie che un'applicazione deve offrire, in modo da soddisfare le aspettative degli stakeholder. [2]

Questo processo è fondamentale per garantire che l’applicazione soddisfi le esigenze degli utenti e raggiunga gli obiettivi prefissati. Inoltre, aiuta gli sviluppatori fornendo una guida durante tutto il processo di sviluppo.

Nell’uso comune, i requisiti vengono categorizzati come funzionali (comportamentali) e non funzionali (tutto il resto). I requisiti funzionali descrivono “cosa deve fare il sistema” mentre quelli non funzionali definiscono “come deve funzionare”. [3]

Solitamente, i requisiti funzionali hanno una priorità maggiore poiché delineano gli aspetti principali del software.

Nella nostra applicazione, abbiamo scelto di identificare esclusivamente i requisiti funzionali, poiché il nostro obiettivo principale non è la commercializzazione, ma piuttosto la conduzione di una ricerca, abbiamo quindi preferito concentrarci sull'analisi di come il sistema deve essere strutturato. In questo capitolo, verranno quindi trattati i requisiti della nostra applicazione e come sono stati identificati.

1.2 Identificazione dei requisiti

Per identificare i requisiti, ci siamo serviti sin dall'inizio di ChatGPT, utilizzandolo per ottenere una serie di funzionalità, le quali poi siamo andati a delineare e sistemare in base agli obiettivi della nostra applicazione in campo ambientale. In particolare, l’applicazione doveva rispettare delle linee guida fondamentali per essere considerata utile per l’ambiente.

Pertanto, avrebbe dovuto principalmente:

- **Permettere agli utenti di pubblicare annunci**: Gli utenti devono poter pubblicare annunci per scambiare o noleggiare oggetti.
- **Attivare noleggi o scambi**: Gli utenti devono poter avviare e gestire il processo di noleggio o scambio degli oggetti.
- **Ricercare gli annunci**: Deve essere possibile cercare e visualizzare gli annunci sfruttando la propria posizione in modo da vedere gli annunci vicini.
- **Implementare un sistema di comunicazione**: È essenziale integrare un sistema di messaggistica per facilitare la comunicazione tra gli utenti, rendendo più semplici e trasparenti gli scambi e i noleggi.

In questo modo è stato così possibile definire delle linee guida per quanto riguarda i punti principali del nostro progetto. Nel prossimo paragrafo andremo a definire più a fondo i requisiti funzionali implementati nell'applicazione.

1.3 Requisiti funzionali

A seguire i principali requisiti identificati per l'applicazione EcoSwap:

- **RF01 – Registrazione:**
  - **Priorità:** Alta
  - **Descrizione:**
    - Il sistema deve permettere agli utenti di registrarsi e creare un account tramite e-mail e password, inserendo anche nome, cognome, luogo e data di nascita;
    - L'utente può effettuare la registrazione anche tramite account Google.
- **RF02 – Autenticazione degli utenti:**
  - **Priorità:** Alta
  - **Descrizione:**
    - L'autenticazione degli utenti deve essere gestita tramite e-mail e password.
    - Una volta autenticato, l'utente deve essere in grado di accedere all'applicazione automaticamente, senza rieseguire il login.
    - L'utente può effettuare anche l'autenticazione tramite account Google.

- **RF03 – Gestione dati:**
  - **Priorità:** Media
  - **Descrizione:**
    - Il sistema deve permettere all'utente di effettuare il recupero della password tramite l'utilizzo della mail usata per registrarsi.
    - Il sistema deve permettere all'utente di modificare i propri dati: password, nome, cognome, data di nascita, luogo e foto profilo.
- **RF04 – Pubblicazione di annunci:**
  - **Priorità:** Alta
  - **Descrizione:**
    - Il sistema deve permettere agli utenti di pubblicare annunci con foto e descrizione (luogo di scambio, descrizione oggetto, titolo) degli oggetti che desiderano scambiare.
    - Il sistema deve permettere agli utenti di pubblicare annunci con foto e descrizione (luogo di noleggio, descrizione dell'oggetto, titolo, tempo massimo di noleggio, costo giornaliero, numero di oggetti disponibili) degli oggetti che desiderano mettere a noleggio.
- **RF05 – Funzionalità di ricerca:**
  - **Priorità:** Media
  - **Descrizione:**
    - Il sistema deve fornire una funzione di ricerca avanzata per aiutare gli utenti a trovare annunci specifici tramite parole chiave e fornire un'interfaccia per visualizzarli, mostrandoli in ordine di distanza.
- **RF06 – Gestione degli annunci preferiti:**
  - **Priorità:** Bassa
  - **Descrizione:**
    - Il sistema deve permettere all'utente di aggiungere o rimuovere gli annunci dalla lista dei preferiti, fornendo un'interfaccia che consente di visualizzare la lista, indicando se si tratta di noleggio o di scambio.
- **RF07 – Noleggio di oggetti:**
  - **Priorità:** Alta
  - **Descrizione:**

        - Il sistema deve permettere agli utenti di selezionare la durata del noleggio, il numero di oggetti da noleggiare, ed effettuare il pagamento tramite la piattaforma integrata.
- **RF08 – Scambio di oggetti:**
    - **Priorità:** Alta
    - **Descrizione:**
        - Il sistema deve permettere agli utenti che intendono effettuare uno scambio di contattare l'autore dell'annuncio per definire i dettagli dello scambio.
- **RF09 – Comunicazione tra utenti:**
    - **Priorità:** Media
    - **Descrizione:**
        - Il sistema mette a disposizione una live chat integrata che consente agli utenti di comunicare direttamente per organizzare i dettagli dello scambio o del noleggio, come luogo, data e orario.
- **RF10 – Feed utente:**
    - **Priorità:** Alta
    - **Descrizione:**
        - Il sistema deve permettere all'utente di visualizzare gli annunci sulla home page, basandosi sulla sua posizione geografica e mostrandoli in ordine di distanza.
- **RF11 – Gestione degli annunci:**
    - **Priorità:** Media
    - **Descrizione:**
        - Gli utenti devono poter di gestire i loro annunci pubblicati, inclusa la rimozione e operazioni di restituzione o reclamo di un oggetto in noleggio.
- **RF12 – Cronologia degli annunci:**
    - **Priorità:** Bassa
    - **Descrizione:**
        - La piattaforma deve fornire strumenti per mostrare gli annunci, inclusi la visualizzazione dello storico dei noleggi, dei noleggi in corso (sia in

vendita che in acquisto), dei noleggi pubblicati, degli scambi pubblicati, e delle comunicazioni passate.

- **RF13 – Recensioni e valutazioni:**
- **Priorità:** Bassa
- **Descrizione:**
  - Il sistema deve fornire un'interfaccia di feedback e valutazioni per gli utenti, in modo che sia possibile valutare la reputazione degli altri membri della comunità, creando così un ambiente di scambio affidabile e trasparente.

I requisiti sono stati elencati in base alla loro priorità per guidare lo sviluppo dell'applicazione. Nel prossimo capitolo tratteremo l'implementazione, la quale procederà in conformità con questa scala di priorità, garantendo che le funzionalità essenziali siano sviluppate per prime.

Capitolo 2

Descrizione dell'implementazione

2.1 Panoramica dell'implementazione

In questo capitolo tratteremo come è stata svolta l'implementazione dell'applicativo, seguendo principalmente i seguenti obiettivi delineati dai requisiti:

- **Pubblicazione e gestione degli annunci**: Permettere agli utenti di pubblicare annunci sia per noleggi che per scambi, e gestirli efficacemente sia dalla parte del compratore che del venditore.
- **Miglioramento dell'esperienza utente**: Mostrare gli annunci in ordine di distanza nella homepage per facilitare la ricerca degli oggetti desiderati.
- **Servizio di messaggistica**: Garantire un sistema di messaggistica integrato per permettere agli utenti di comunicare e coordinare facilmente scambi e noleggi.
- **Gestione del profilo utente**: consentire agli utenti di visualizzare e modificare le proprie informazioni personali.

Il primo passo è stato identificare le tecnologie necessarie per realizzare l'applicazione. Questo ha comportato la scelta delle API e delle librerie più adatte per le funzionalità richieste, la selezione di servizi cloud affidabili per la gestione dei dati, e la configurazione di un database per memorizzare le informazioni sugli utenti.

Successivamente, è stata definita l'architettura del sistema, che includeva la struttura dell'applicazione e il flusso dei dati tra le varie componenti in modo da garantire scalabilità e facilità di manutenzione.

Con le tecnologie e l'architettura definite, si è proceduto con lo sviluppo vero e proprio dell'applicazione. Il processo di sviluppo è stato guidato dai requisiti prioritari, assicurando che le funzionalità essenziali fossero implementate per prime.

Nei prossimi paragrafi, esploreremo nel dettaglio come sono stati eseguiti tutti questi passi, fornendo una visione approfondita del processo di implementazione e delle scelte tecniche effettuate.

2.2 Tecnologie utilizzate

Durante lo sviluppo sono state sfruttate utilizzate diverse tecnologie, in questa sezione verranno descritte le principali tecnologie, come API (Application Programming Interface) [4], framework o librerie utilizzate per sviluppare l'applicazione EcoSwap.

2.2.1 Flutter

Flutter [5]è stato scelto come framework di sviluppo principale per la sua capacità di creare applicazioni cross-platform con una singola base di codice. Flutter utilizza il linguaggio di programmazione **Dart** [6] e offre una vasta gamma di widget per la creazione di interfacce utente.

La scelta di Flutter è stata motivata dalla sua efficienza nello sviluppo e dalla facilità con cui consente di implementare funzionalità complesse.

2.2.2 Firebase

Firebase è una piattaforma per lo sviluppo di applicazioni mobile e web, è stato integrato nell'applicazione per fornire una serie di servizi backend, essenziali per il funzionamento dell'app, come il salvataggio degli annunci e delle informazioni degli utenti in un cloud database.

Le principali componenti di Firebase utilizzate in questo progetto sono:

- **Firebase Authentication**: Per la gestione dell'autenticazione degli utenti tramite e-mail, password e autenticazione Google. [7]
- **Firebase Realtime Database**: Per l'archiviazione e la sincronizzazione dei dati in tempo reale, permettendo agli utenti di vedere aggiornamenti immediati sugli annunci. [8]
- **Firebase Storage**: Per l'archiviazione sicura di immagini e altri file caricati dagli utenti. [9]

2.2.3 Stripe

Stripe Checkout è un'API di Stripe scelta per la gestione dei pagamenti all'interno dell'applicazione. Stripe offre una soluzione sicura e affidabile per elaborare pagamenti online, fornendo un'interfaccia di pagamento predefinita. Questa interfaccia garantisce che le

transazioni degli utenti, effettuate con carta di credito o altre modalità di pagamento, siano gestite in modo sicuro e conforme agli standard internazionali. [10]

2.2.4 Geolocator e Geocoding

Per le funzionalità basate sulla posizione geografica, sono state utilizzate le librerie **Geolocator** e **Geocoding**:

- **Geolocator**: Questa libreria è utilizzata per ottenere la posizione geografica attuale dell'utente, fondamentale per mostrare gli annunci vicini.
- **Geocoding**: Utilizzata per convertire indirizzi in coordinate geografiche (e viceversa), migliorando la precisione e l'usabilità delle ricerche geografiche, fondamentale anch'essa nel calcolo della distanza tra un utente e un annuncio.

2.2.5 Other libraries

Oltre alle tecnologie principali sopra menzionate, l'applicazione utilizza diverse altre librerie per migliorare l'esperienza utente e facilitare lo sviluppo:

- **shared_preferences**: Per memorizzare dati semplici in modo persistente, utilizzate per salvare localmente le informazioni dell'utente essenziali per riuscire ad effettuare un login automatico.
- **sqflite**: Per il salvataggio dei dati in locale, utilizzato per memorizzare le informazioni dell'utente e altri dati, permettendo così l'accesso ai dati anche offline.
- **image_picker**: Per consentire agli utenti di selezionare e caricare immagini dai loro dispositivi, utilizzato specialmente per caricare le foto degli annunci o aggiornare l'immagine profilo.
- **flutter_local_notifications**: Questa libreria consente di programmare e gestire notifiche che possono avvisare gli utenti di eventi importanti.
- **provider**: Per la gestione dello stato dell'applicazione in modo reattivo e scalabile.

2.3 Architettura del Sistema

2.3.1 Architettura generale

Per lo sviluppo dell'applicazione, avevamo bisogno di un pattern architetturale che ci permettesse di separare in modo pulito la logica di business dall'interfaccia utente. Abbiamo quindi deciso di utilizzare un'architettura Model-View-ViewModel (MVVM), la quale risulta

un'ottima architettura soprattutto da un punto di vista di manutenzione per diversi motivi [11]:

- **Separazione delle preoccupazioni**: MVVM ci permette di separare chiaramente la logica di business (gestita dai ViewModel) dalla logica di presentazione (gestita dalle View), facilitando la gestione e la manutenzione del codice.
- **Facilità di test**: Grazie a MVVM, possiamo testare la logica di business senza coinvolgere la UI, migliorando la qualità del codice attraverso test unitari.
- **Riutilizzo del codice**: La separazione delle logiche consente una maggiore riusabilità dei componenti del software, come molteplici ViewModel.

2.3.2 Componenti dell'architettura

View

La View è responsabile della presentazione dell'interfaccia utente e dell'interazione con esso. In EcoSwap, le View sono implementate utilizzando i widget di Flutter [12]. Questi widget osservano i dati forniti dai ViewModel e aggiornano automaticamente l'interfaccia utente in base alle modifiche.

Le principali responsabilità delle View nell'applicazione includono:

- **Visualizzazione degli annunci**: Mostrare gli annunci di scambio e noleggio agli utenti, organizzati in base alla distanza geografica.
- **Interazione con l'utente**: Gestire l'input dell'utente, come la pubblicazione di nuovi annunci e l'invio di messaggi.
- **Aggiornamento dinamico**: Aggiornare l'interfaccia in risposta ai cambiamenti nei dati del ViewModel

ViewModel

Il ViewModel agisce come intermediario tra la View e il repository (parte del Model), gestendo così la logica di presentazione e fornendo i dati necessari alla View. Questa separazione consente una maggiore modularità facilitando soprattutto il testing unitario della logica di business in quanto non convolge la UI.

In EcoSwap, i ViewModel sono due:

- **AdViewModel:** Gestisce tutto ciò che riguarda gli annunci, come caricamenti, eliminazioni e avvio dei noleggi, passando i dati dal repository alla View e viceversa.
- **UserViewModel:** Gestisce tutto ciò che riguarda gli utenti, come login, registrazione e aggiornamento della pagina profilo, anch'esso passando i dati dal repository alla View e avviando chiamate dalla View al repository.

Repository

Il Repository funge da intermediario tra i ViewModel e i DataSource. In EcoSwap, il Repository si occupa di gestire e indirizzare le chiamate tra i molteplici DataSource, decidendo, ad esempio, se salvare un'informazione localmente o in un database in cloud. Analogamente ai ViewModel, anche i repository sono due:

- **AdRepository:** Gestisce le chiamate tra i diversi DataSource dedicati agli annunci, selezionando il DataSource appropriato in base al tipo di annuncio.
- **UserRepository:** Gestisce le chiamate relative agli utenti, indirizzandole tra database locale o remoto.

Data Source

Il Data Source è la componente che interagisce direttamente con le sorgenti dati esterne, come Firebase e database locali[1]. Il Data Source si occupa di effettuare operazioni CRUD [13], gestendo quindi la creazione, lettura, aggiornamento e cancellazione dei dati da database locali o remoti.

In EcoSwap vi sono molteplici Data Source, uno per ogni modello rappresentato nell'applicazione. I principali sono:

- **UserDataSource:** Gestisce tutte le operazioni relative agli utenti, come la registrazione, il login o la modifica delle informazioni dell'utente, salvando tali informazioni in locale o in remoto, a seconda del contesto.
- **RentalDataSource e RentalLocalDataSource:** Gestiscono le operazioni relative ai noleggi, come la creazione, l'eliminazione o l'avvio di un noleggio.

[1] Il salvataggio in database locali può risultare utile in caso di mancanza di connessione, permettendo comunque un utilizzo limitato dell'applicazione, nel nostro caso, è possibile solamente visualizzare le informazioni personali.

- **ExchangeDataSource e ExchangeLocalDataSource:** Gestiscono le operazioni relative agli scambi, come la creazione e l'eliminazione.
- **ChatDataSource:** Gestisce le operazioni relative alle chat, come la creazione stream per l'inserimento di nuove chat o messaggi nel database.

2.3.3 Interazioni nell'architettura

Il flusso dei dati nell'architettura MVVM di EcoSwap segue questo schema:

1. **Utente interagisce con la View**: L'utente compie un'azione, come pubblicare un annuncio o avviare un noleggio.
2. **View comunica con il ViewModel**: La View invia l'azione al ViewModel appropriato.
3. **ViewModel interagisce con il Repository**: Il ViewModel richiama le operazioni necessarie nel Repository.
4. **Repository accede ai Data Source**: Il Repository effettua le operazioni sui dati richiamando i DataSource necessari.
5. **DataSource aggiorna i dati**: Il DataSource effettua le operazioni di aggiornamento dei dati.
6. **Aggiornamento dei dati**: Una volta che i dati sono aggiornati, la View viene notificata dei cambiamenti grazie al meccanismo di Future di Flutter[2], aggiornando di conseguenza l'interfaccia utente.

Le interazioni tra le componenti dell'architettura sono illustrate nel diagramma seguente:

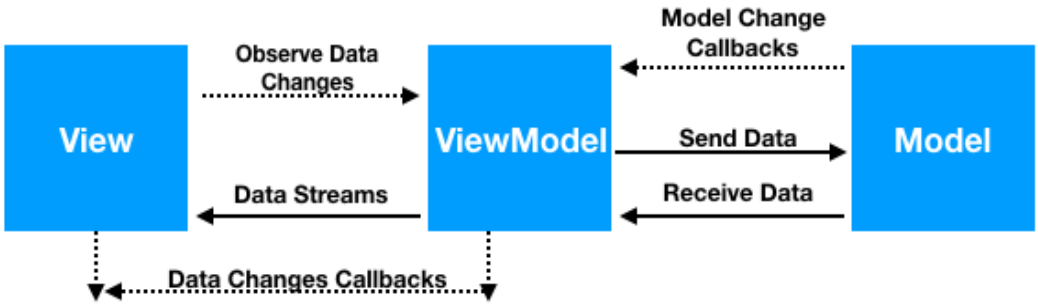

Figura 2.1: Schema rappresentate il funzionamento del Pattern MVVM

[2] In Flutter, un Future è un oggetto che rappresenta un'operazione asincrona. Quando l'operazione è completa, il Future notifica i suoi ascoltatori, permettendo la gestione reattiva degli aggiornamenti dello stato.

2.4 Implementazione delle pagine principali

EcoSwap è composta da cinque pagine principali, ognuna accessibile tramite la barra di navigazione inferiore (bottom bar). Ogni pagina svolge un ruolo fondamentale nell'implementazione di diverse funzionalità chiave, garantendo una user experience fluida e intuitiva:

2.4.1 HomePage

La Home Page è la schermata principale dell'applicazione, dove gli utenti possono visualizzare gli annunci disponibili per scambi e noleggi. Gli annunci sono ordinati per distanza dalla posizione dell'utente per facilitare la ricerca. La HomePage è composta da due sezioni, una dedicata ai noleggi e l'altra agli scambi, accessibili tramite un pulsante in cima alla pagina. Per implementare la visualizzazione degli annunci è stata utilizzata una *GridView* che, insieme a un meccanismo di caricamento progressivo, permette di visualizzare continuamente nuovi annunci in base alla distanza dalla posizione attuale dell'utente.

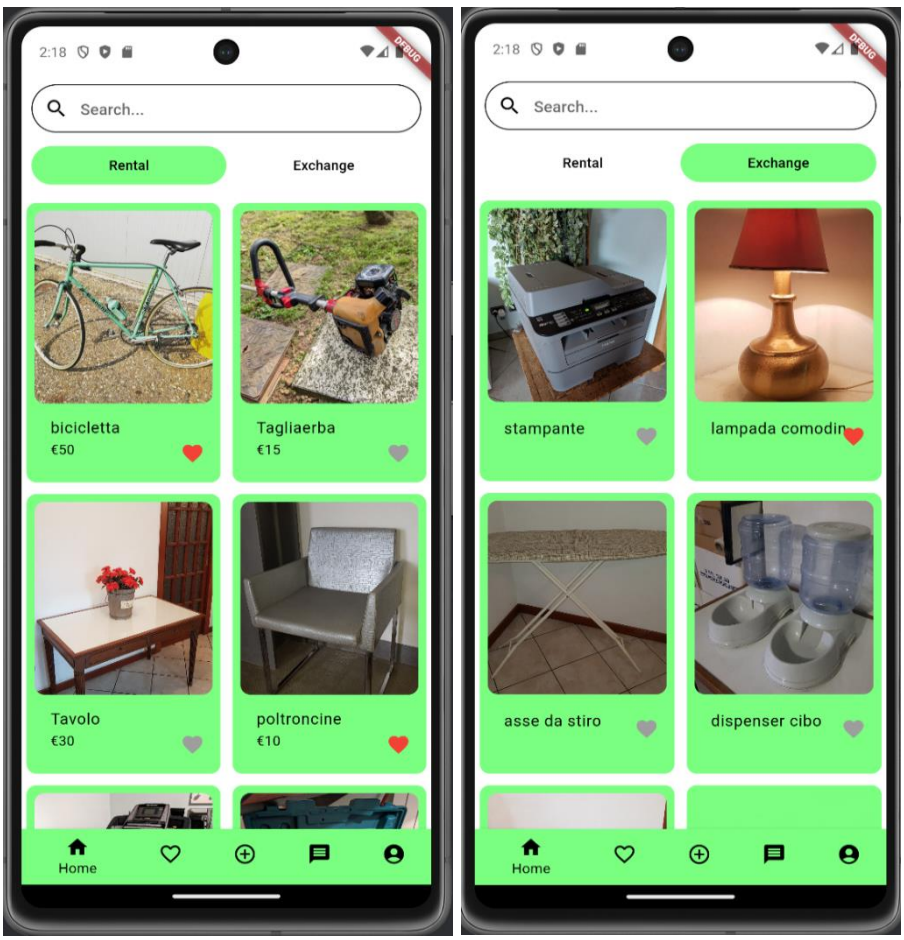

Figura 2.2: Immagine rappresentate la HomePage di EcoSwap

In cima alla pagina è presente una barra di ricerca che consente di cercare annunci specifici. Dalla Home Page, gli utenti possono anche salvare gli annunci nei preferiti o cliccare su di essi per visualizzare i dettagli. Da qui, è poi possibile decidere se avviare il noleggio, lo scambio o aprire una chat con il venditore.

2.4.2 Pagina dei preferiti

Attraverso la pagina dei preferiti è possibile visualizzare gli annunci salvati grazie a una *ListView*, organizzati in modo simile alla Home Page con una divisione tra noleggi e scambi per garantire maggiore chiarezza. Cliccando su uno degli annunci, si può visitare la pagina specifica dell'annuncio, proprio come avviene nella Home Page.

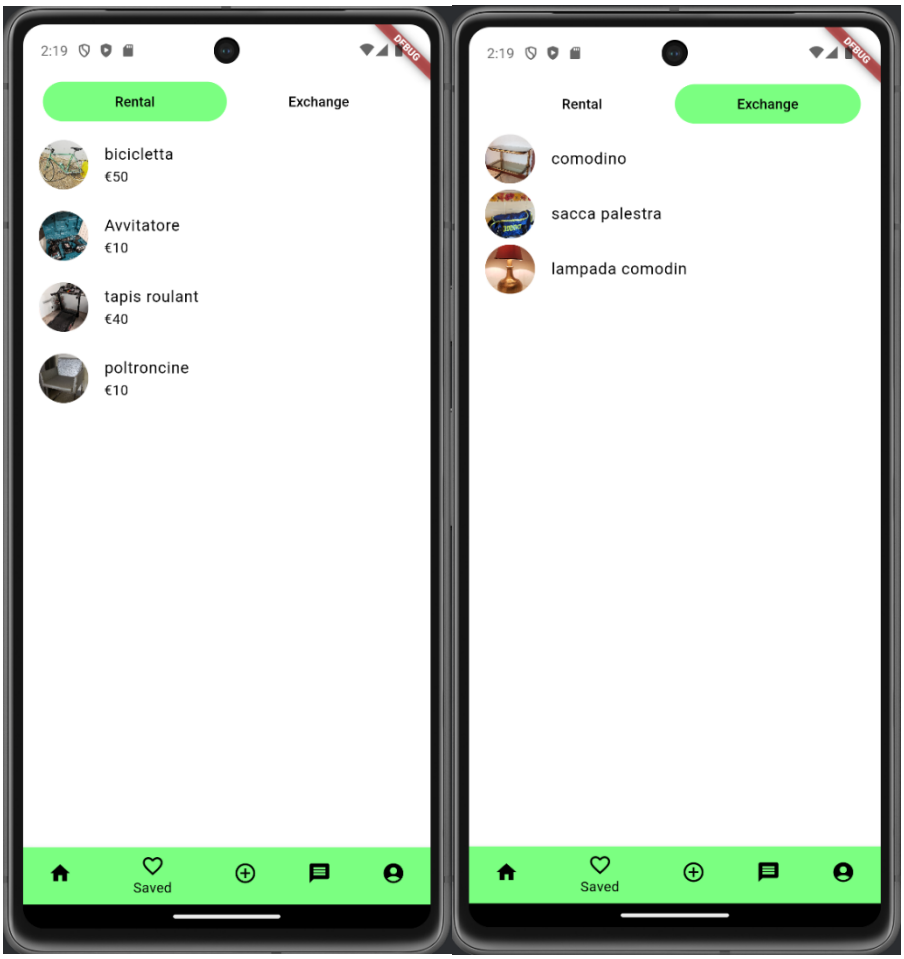

Figura 2.3: Immagine rappresentate la Pagina dei preferiti di EcoSwap

2.4.3 Pagina di caricamento

Questa pagina implementa l'obiettivo principale dell'applicazione, ossia il caricamento degli annunci. La struttura della pagina è divisa in modo da distinguere tra noleggi e scambi; infatti, in base al tipo di annuncio che si vuole caricare, la pagina presenta campi differenti. I campi

comuni a entrambi i tipi di annuncio sono il titolo, la descrizione e la posizione geografica, che può essere indicata sulla mappa. Per facilitare l'inserimento della posizione, è stata integrata una mappa interattiva che consente agli utenti di selezionare facilmente il luogo desiderato. Oltre a questi campi comuni, la pagina offre sezioni specifiche per raccogliere informazioni dettagliate pertinenti a noleggi e scambi, assicurando che tutti i dettagli necessari siano forniti in modo chiaro e organizzato.

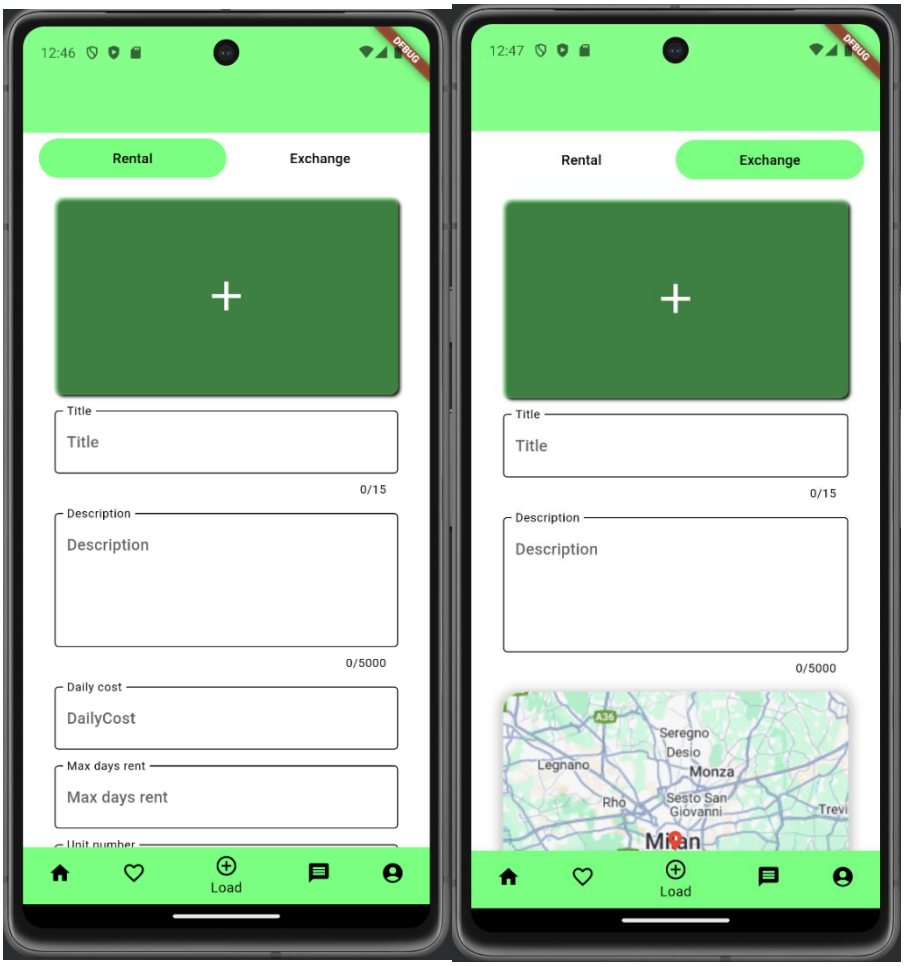

Figura 2.4: Immagine rappresentate la Pagina di caricamento di EcoSwap

Per implementare questa pagina, ci siamo serviti della libreria *ImagePicker* per permettere all'utente di caricare immagini dal proprio telefono direttamente sull'applicazione premendo il tasto (+). Dopo aver caricato l'immagine, essa verrà mostrata immediatamente. Per creare il meccanismo della mappa, abbiamo sviluppato un nuovo widget chiamato *MapWidget*. Utilizzando il widget *FlutterMap* e la posizione attuale dell'utente ottenuta grazie alle librerie di geolocalizzazione, la mappa viene inquadrata su un'area attorno ad esso, permettendogli di selezionare un qualsiasi punto per segnare il luogo dello scambio/noleggio.

2.4.4 Pagina delle chat

In questa pagina è possibile visualizzare tutti i messaggi scambiati con altri utenti. Le chat sono ordinate temporalmente e sono basate su un annuncio specifico. È quindi possibile avere più chat con un solo utente, ma solo per annunci differenti. Aprendo la chat, è possibile controllare a quale annuncio si sta facendo riferimento. Inoltre, è stato implementato un meccanismo per mostrare se l'ultimo messaggio è stato letto oppure no, grazie a un campo booleano aggiunto ai messaggi.

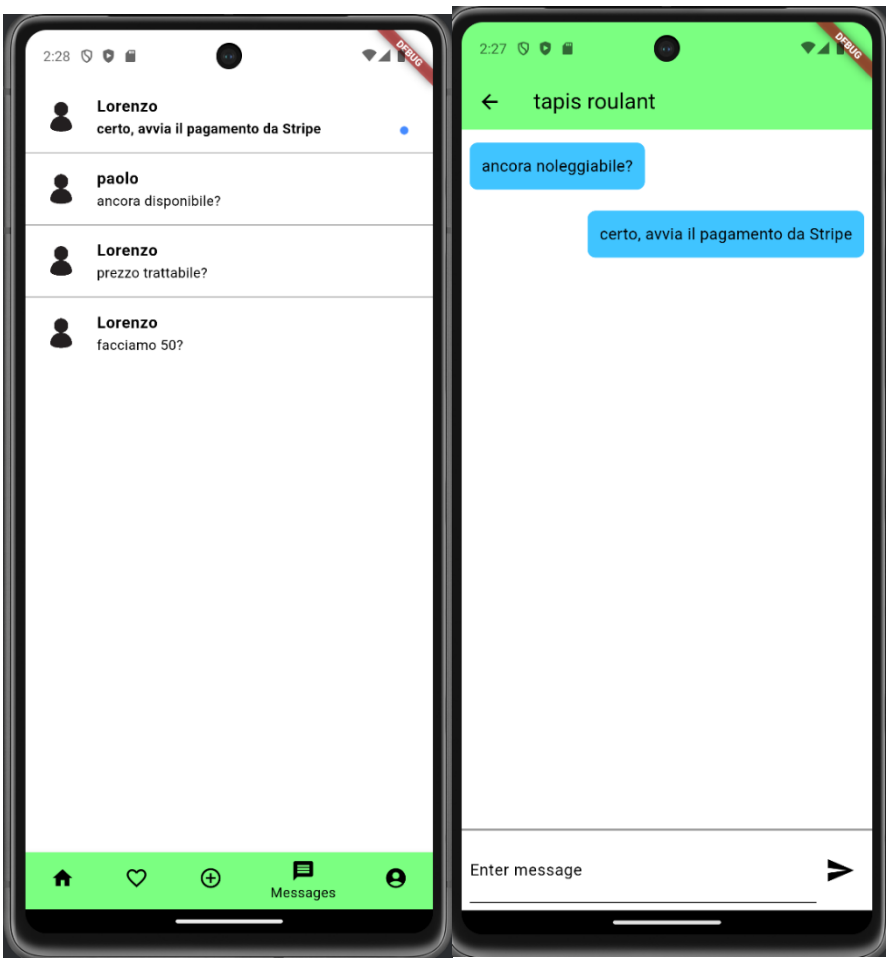

Figura 2.5: Immagine rappresentate la Pagina delle chat di EcoSwap

Per avere una chat con messaggi in tempo reale, abbiamo utilizzato uno stream di *DatabaseEvent* connesso a Firebase, che consente di visualizzare un nuovo messaggio ogni volta che esso viene salvato su Firebase.

2.4.5 Pagina del profilo

Nella pagina del profilo, è possibile visualizzare le informazioni dell'utente loggato, come le recensioni premendo l'apposito bottone o le varie liste degli annunci, quali i noleggi attivi

venduti, i noleggi attivi acquistati, i noleggi pubblicati e gli scambi pubblicati. È inoltre possibile accedere ai precedenti ordini effettuati, modificare le informazioni del proprio profilo, oppure effettuare il logout premendo i tre puntini nella parte superiore dello schermo.

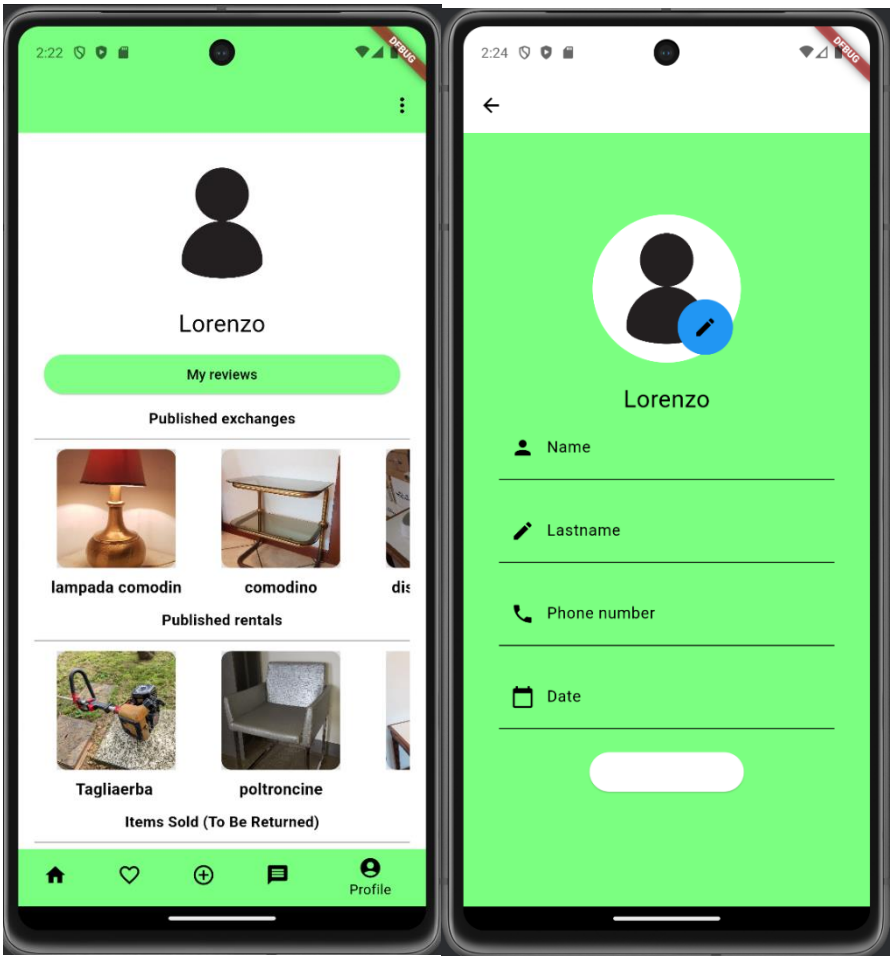

Figura 2.6: Immagine rappresentate la Pagina del profilo di EcoSwap

Inoltre, è possibile cliccare su un annuncio nelle varie liste per effettuare operazioni specifiche. Ad esempio, è possibile avviare il processo di reclamo o restituzione per un noleggio, gestire le interazioni con gli altri utenti o eliminare un annuncio dalla lista degli annunci pubblicati.

2.4.6 Pagina di Login e Registrazione

Altre pagine molto importanti sono quelle di login e registrazione dato che attraverso queste pagine è possibile registrarsi o accedere all'app. Grazie all'utilizzo della libreria *SharedPreferences*, il login diventerà automatico fino a che non si rieffettuerà un logout attraverso la pagina del profilo. Vi è anche la possibilità di accedere tramite un account Google, rendendo il processo di login più rapido e semplice. Inoltre, nel caso si sia dimenticata la password, è possibile recuperarla facilmente inserendo il proprio indirizzo e-mail. Una volta

inserito l'indirizzo e-mail, verrà inviata una mail con le istruzioni per reimpostare la password, garantendo così un processo di recupero intuitivo per l'utente.

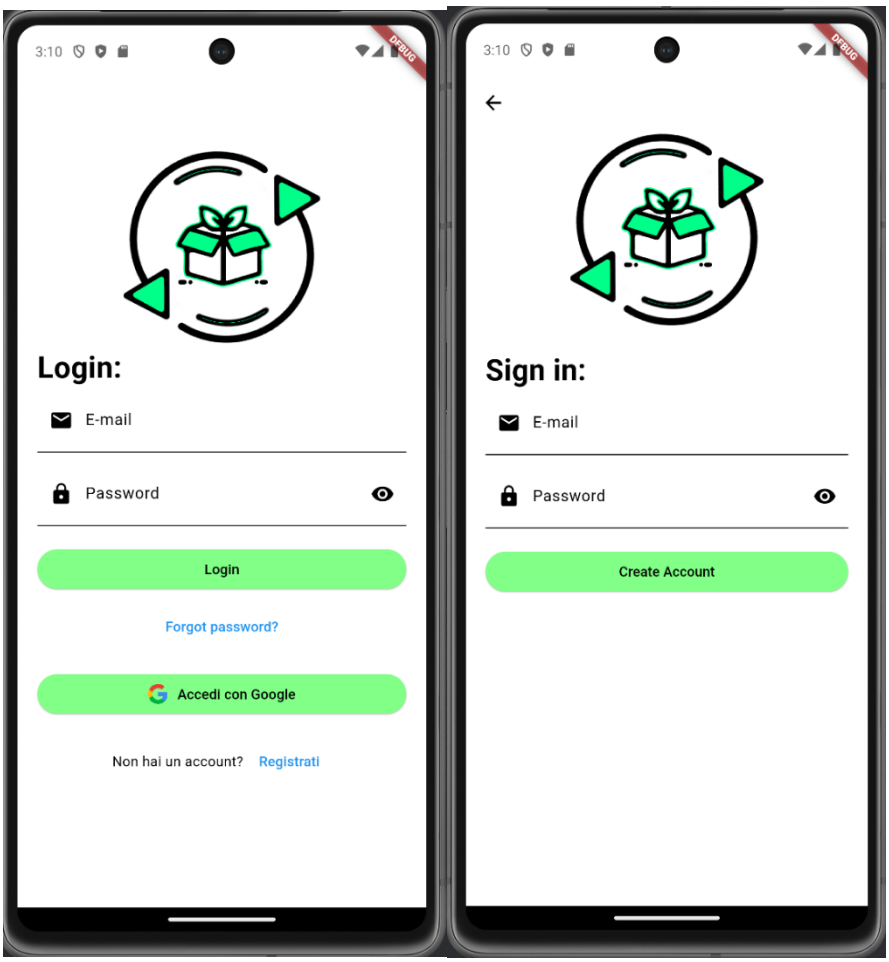

Figura 2.6: Immagine rappresentate la Pagina di Login di EcoSwap

2.5 Conclusioni

Nello sviluppo di EcoSwap, il nostro obiettivo era offrire un'alternativa sostenibile alle tradizionali piattaforme di consumo. Le funzionalità principali dell'applicazione, come la gestione degli annunci, le chat in tempo reale e il profilo utente, sono state implementate con successo per promuovere scambi eco-sostenibili. Tuttavia, riteniamo che ci siano opportunità per migliorare ulteriormente l'esperienza degli utenti. Un'interfaccia con un design più pulito e un sistema di notifiche più avanzato, che avvisi gli utenti quando un noleggio sta per scadere o altre informazioni rilevanti, potrebbe essere considerato per migliorare l'usabilità e l'esperienza degli utenti.

Capitolo 3

Testing

3.1 Panoramica ai test

In questo capitolo verrà trattato l'approccio che è stato utilizzato durante la fase di testing dell'applicazione, le difficoltà incontrate durante questa fase e i diversi test effettuati. La fase di testing rappresenta un passaggio cruciale nel ciclo di sviluppo del software, essenziale per garantire che l'applicazione sia stabile, funzionale e risponda ai requisiti degli utenti.

Gli obiettivi principali del testing sono [14]:

- **Verificare la Correttezza Funzionale**: Assicurare che ogni funzione e componente dell'applicazione esegua correttamente il proprio compito, identificando e correggendo eventuali errori logici o di implementazione.
- **Garantire la Stabilità**: Testare l'applicazione in vari scenari e condizioni per assicurare che sia stabile ed evitare crash inaspettati.
- **Migliorare le Prestazioni**: Identificare e risolvere i problemi di performance per garantire un'esperienza utente fluida, anche sotto carico.
- **Assicurare l'Usabilità**: Raccogliere feedback dagli utenti per migliorare l'interfaccia e l'esperienza utente complessiva.
- **Valutare l'Interoperabilità**: Verificare che l'applicazione funzioni correttamente su diverse piattaforme e dispositivi.

3.2 Strategia di testing

La strategia di testing ha compreso tre diverse tipologie di test, ognuna mirata a verificare specifici aspetti e funzionalità dell'applicazione [15]:

Test di unità:

- **Descrizione:** I test di unità sono stati utilizzati per verificare la correttezza delle singole unità di codice, come funzioni e metodi isolati.
- **Obiettivo:** Garantire che ogni singola parte del codice funzioni correttamente e produca i risultati attesi senza errori.

Test di widget:

- **Descrizione:** I test sui widget sono stati utilizzati per verificare il comportamento dei componenti dell'interfaccia utente (UI) in isolamento.
- **Obiettivo:** Assicurare che ogni componente UI risponda correttamente agli input e che la sua interfaccia utente sia costruita correttamente secondo le specifiche.

Test di integrazione:

- **Descrizione:** I test di integrazione sono stati utilizzati per verificare che diversi moduli e componenti dell'applicazione funzionino correttamente insieme.
- **Obiettivo:** Assicurare che l'interazione tra i vari moduli e componenti sia conforme alle specifiche e che l'applicazione nel suo complesso funzioni come previsto quando i vari componenti sono combinati.

3.3 Strumenti e Framework Utilizzati

Nella realizzazione dei test sono stati utilizzati principalmente i seguenti strumenti e framework:

- **Flutter Test Framework:** Framework di testing fornito da Flutter per eseguire test su widget e integrazione.
- **Mockito:** Libreria per creare mock[3] e stub[4] necessari per i test, facilitando la simulazione di comportamenti e risposte.
- **Firebase Auth Mocks:** Libreria specifica per creare mock di Firebase, i quali permettono di simulare le funzionalità di autenticazione, permettendo di testare la gestione degli utenti senza connessione a un effettivo database.

3.4 Descrizione dei Test Effettuati

In questa sezione verranno descritti i test effettuati per garantire la correttezza e la funzionalità dell'applicazione. I test sono stati suddivisi in due categorie principali: test di registrazione e test di login.

[3] Oggetto simulato che replica il comportamento di un oggetto reale in modo controllato.
[4] Tipo di oggetto simulato utilizzato nei test unitari e di integrazione per sostituire componenti o dipendenze reali di un sistema con versioni semplificate che forniscono risposte predefinite.

- **Test di Registrazione:**
  - **Registrazione con Successo**
    - **Descrizione:** Questo test verifica che un utente possa registrarsi correttamente utilizzando un'e-mail e una password valide.
    - **Scenario:** Viene fornita un'e-mail valida (test@example.com) e una password valida (password123).
    - **Risultato atteso:** Il metodo *createUserWithEmailAndPassword* restituisce un oggetto *UserCredential*, e la funzione *registration* restituisce il messaggio "Success".
  - **Password Debole:**
    - **Descrizione:** Questo test verifica che il sistema gestisca correttamente il caso in cui la password fornita sia troppo debole.
    - **Scenario:** Viene fornita un'e-mail valida (test@example.com) e una password debole (123).
    - **Risultato atteso:** Il metodo *createUserWithEmailAndPassword* lancia un'eccezione *FirebaseAuthException* con associato il codice *weak-password*, e la funzione *registration* restituisce il messaggio "The password provided is too weak."
  - **E-mail già in Uso:**
    - **Descrizione:** Questo test verifica che il sistema gestisca correttamente il caso in cui l'e-mail fornita sia già associata ad un account esistente.
    - **Scenario:** Viene fornita un'e-mail già in uso (test@example.com) e una password valida (password123).
    - **Risultato atteso:** Il metodo *createUserWithEmailAndPassword* lancia un'eccezione *FirebaseAuthException* con associato il codice *email-already-in-use*, e la funzione *registration* restituisce il messaggio "The account already exists for that email."
  - **Errore Generico:**
    - **Descrizione:** Questo test verifica che il sistema gestisca correttamente eventuali errori generici che possono verificarsi durante la registrazione.
    - **Scenario:** Viene fornita un'email valida (test@example.com) e una password valida (password123).

        - **Risultato atteso:** Il metodo *createUserWithEmailAndPassword* lancia un'eccezione *FirebaseAuthException* con un codice e un messaggio generico associato, e la funzione *registration* restituisce il messaggio corrispondente.
    - **Errore Non di FirebaseAuth:**
        - **Descrizione:** Questo test verifica che il sistema gestisca correttamente errori che non sono specifici di *FirebaseAuthException*.
        - **Scenario:** Viene fornita un'e-mail valida (test@example.com) e una password valida (password123).
        - **Risultato atteso:** Il metodo *createUserWithEmailAndPassword* lancia una generica eccezione *Exception*, e la funzione *registration* restituisce il messaggio corrispondente.
- **Test di login:**
    - **Login con Credenziali Valide:**
        - **Descrizione:** Questo test verifica che un utente possa effettuare il login correttamente utilizzando un'e-mail e una password valide.
        - **Scenario:** Viene fornita un'e-mail valida (1@2.com) e una password valida (123456).
        - **Risultato Atteso:** Il metodo *signInWithEmailAndPassword* restituisce un oggetto *UserCredential*, e la funzione login restituisce il messaggio "Success".
    - **Login con Credenziali Errate:**
        - **Descrizione:** Questo test verifica che il sistema gestisca correttamente il caso in cui le credenziali fornite siano errate.
        - **Scenario:** Viene fornita un'e-mail non valida (invalid@example.com) e una password non valida (invalidpassword).
        - **Risultato Atteso:** Il metodo *signInWithEmailAndPassword* lancia un'eccezione *FirebaseAuthException* con associato il codice *wrong-password*, e la funzione login restituisce il messaggio "Wrong password provided for that user."

Questi test sono stati effettuati per assicurare che le funzionalità critiche di registrazione e login dell'applicazione funzionino correttamente, gestendo sia i casi di successo sia i possibili errori che possono verificarsi.

3.5 Sfide e Soluzioni

Durante il processo di testing, abbiamo incontrato diverse sfide, tra cui la mancanza di una documentazione completa per l'integrazione con Firebase Database. Questa carenza ha reso difficile eseguire test di integrazione accurati per alcune funzionalità chiave dell'applicazione. Inoltre, nonostante l'utilizzo di modelli di linguaggio per ottenere assistenza, le informazioni fornite non sono state sufficienti per risolvere i problemi riscontrati.
Queste problematiche saranno esplorate nei capitoli successivi, dove verrà analizzato il valore aggiunto derivante dall'utilizzo di un LLM come ChatGPT nello sviluppo del progetto. Nonostante l'incapacità di ChatGPT di eseguire test funzionanti per la nostra applicazione, questa informazione è comunque significativa, dato che il nostro obiettivo principale è comprendere l'utilità di un LLM nel contesto dello sviluppo di un'app mobile.

Capitolo 4

Utilizzo di ChatGPT

4.1 Introduzione

Questo capitolo esplora come ChatGPT è stato utilizzato nelle diverse fasi dello sviluppo dell'applicazione EcoSwap, evidenziando il suo ruolo e la sua efficacia nel supportare lo sviluppo. L'obiettivo è valutare quindi l'impatto di ChatGPT durante l'analisi dei requisiti, l'implementazione delle funzionalità e il testing del software. L'analisi si concentrerà su come ChatGPT ha contribuito a migliorare la comprensione dei requisiti, a risolvere problemi di programmazione come la correzione di errori, e a verificare la correttezza delle funzionalità implementate. Attraverso esempi pratici e valutazioni critiche, verranno esaminati i vantaggi tangibili e i limiti riscontrati nell'uso di questo strumento. La panoramica fornita metterà in luce le potenzialità di ChatGPT nel migliorare il processo di sviluppo di applicazioni mobili. Alla fine del capitolo, una sezione conclusiva presenterà delle statistiche che riassumono per ogni fase dello sviluppo quanto ChatGPT è stato utilizzato e quanto è stato utile.

4.2 Analisi dei requisiti

4.2.1 Ruolo di ChatGPT nell'Analisi dei requisiti

Durante la fase di analisi dei requisiti, ChatGPT ha svolto un ruolo fondamentale come strumento di supporto, contribuendo in maniera significativa all'identificazione dei requisiti necessari per la nostra applicazione. L'uso del modello di linguaggio ha permesso di individuare prontamente i requisiti essenziali, evidenziando tempestivamente eventuali problematiche o soluzioni relative all'applicazione. Questo ha consentito di ridurre notevolmente i tempi necessari per delineare i punti fondamentali di questa fase del progetto, ottimizzando il processo e migliorando l'efficienza complessiva dell'analisi.

Durante la fase di analisi dei requisiti per l'applicazione EcoSwap, sono emerse diverse sfide significative che hanno richiesto un'attenzione particolare per essere risolte efficacemente. L'integrazione di ChatGPT si è rivelata cruciale per superare questi ostacoli, migliorando l'efficienza e la precisione del processo di definizione dei requisiti. Di seguito, una descrizione delle principali sfide affrontate e del contributo di ChatGPT nella loro risoluzione.

Identificazione dei Requisiti Fondamentali

Sfida: La determinazione dei requisiti fondamentali dell'applicazione EcoSwap, che includevano la registrazione degli utenti, la pubblicazione di annunci, la funzionalità di ricerca avanzata, il noleggio di oggetti, la comunicazione tra utenti, la verifica dell'identità, la gestione degli annunci e delle transazioni, il riscontro e le valutazioni.

Contributo di ChatGPT: ChatGPT ha facilitato l'identificazione dei requisiti fondamentali attraverso un'analisi dettagliata e strutturata delle esigenze dell'applicazione. Grazie alla sua capacità di elaborare informazioni in modo coerente e organizzato, ha permesso di delineare con chiarezza ogni aspetto necessario per il funzionamento ottimale dell'app, diminuendo il carico di lavoro effettivo di questa fase di progetto che si è limitato ad aggiustare eventuali requisiti a seconda delle nostre esigenze.

Evidenziazione delle Problematiche Potenziali

Sfida: Riconoscere e affrontare tempestivamente le potenziali problematiche che potrebbero sorgere durante lo sviluppo dell'applicazione, come la sicurezza delle transazioni, la protezione dei dati personali e le modalità di noleggio e scambio.

Contributo di ChatGPT: Il modello di linguaggio ha messo in luce sin da subito possibili problematiche e ha suggerito soluzioni efficaci per affrontarle. Questo ha permesso di prevenire potenziali bug e ottimizzare le prestazioni fin dalle prime fasi dello sviluppo.

Riduzione dei Tempi di Definizione dei Requisiti

Sfida: Ridurre i tempi necessari per definire i requisiti fondamentali dell'applicazione, mantenendo al contempo un alto livello di dettaglio e precisione.

Contributo di ChatGPT: Grazie alla sua capacità di elaborare rapidamente grandi quantità di informazioni, ChatGPT ha consentito di abbreviare significativamente il tempo necessario per la definizione dei requisiti. Inoltre, ha permesso di definire i requisiti in dettaglio, spiegandone la struttura e fornendo una chiara definizione di ciascun elemento.

4.2.2 Vantaggi e limiti

L'impiego di ChatGPT durante la fase di analisi dei requisiti per l'applicazione EcoSwap ha portato numerosi benefici che hanno notevolmente migliorato l'efficacia e l'efficienza del processo. Uno dei principali vantaggi è stato l'incremento dell'efficienza e della rapidità. ChatGPT ha accelerato significativamente il processo di identificazione e definizione dei requisiti, riducendo i tempi tradizionalmente necessari per questa fase critica a una semplice revisione e eventuale correzione di quanto generato dal LLM.

Inoltre, ChatGPT ha contribuito a una maggiore precisione e completezza nella definizione dei requisiti. Il modello ha fornito analisi dettagliate e strutturate, aiutando a evitare omissioni e ambiguità. Questa chiarezza ha ridotto il rischio di fraintendimenti e problemi successivi nelle fasi di sviluppo e implementazione. La capacità di ChatGPT di evidenziare tempestivamente potenziali problematiche e suggerire soluzioni efficaci ha ulteriormente migliorato il processo.

Nonostante i numerosi vantaggi, l'utilizzo di ChatGPT ha evidenziato anche alcuni limiti e aree di miglioramento. Uno dei principali limiti è la comprensione contestuale limitata. Sebbene ChatGPT sia in grado di fornire risposte dettagliate e pertinenti, la sua capacità di comprendere appieno il contesto specifico del progetto e le sfumature delle esigenze degli utenti è inferiore rispetto a quella umana. Questo può portare a risposte che, pur essendo tecnicamente corrette, potrebbero non rispecchiare completamente le esigenze specifiche del progetto.

Un altro limite riguarda la dipendenza dalla qualità degli input forniti. La precisione e l'utilità delle risposte di ChatGPT dipendono fortemente dalla qualità e dalla completezza delle informazioni iniziali inserite. Input incompleti o poco chiari possono compromettere la qualità delle risposte. Pertanto, è essenziale fornire informazioni strutturate e dettagliate per ottenere il massimo beneficio dall'uso di ChatGPT.

La capacità di innovazione di ChatGPT rappresenta un ulteriore limite. Il modello tende a basarsi su informazioni preesistenti e potrebbe non sempre proporre soluzioni innovative o creative, limitandosi a suggerimenti basati su dati esistenti. Questo può restringere la gamma di soluzioni proposte, riducendo il potenziale per l'innovazione.

Infine, nonostante le capacità avanzate di ChatGPT, è necessaria una supervisione umana per verificare la correttezza e la rilevanza delle risposte fornite.

4.3 Implementazione

4.3.1 Ruolo di ChatGPT nell’Implementazione

Durante la fase di implementazione delle funzionalità dell'applicazione, ChatGPT ha svolto un ruolo significativo come strumento di supporto per lo sviluppo. In particolare, ChatGPT è stato utilizzato per implementare specifiche funzionalità e ottimizzare il codice esistente. Piu specificatamente, ChatGPT è stato utilizzato per i seguenti aiuti:

- **Consigli su sintassi flutter**

  Come già detto, abbiamo voluto testare anche quanto un LLM può aiutare nell'apprendimento di nuovi linguaggi. Uno dei primi utilizzi di ChatGPT è stato per porre domande su come scrivere del codice in Flutter. Ad esempio, abbiamo chiesto come funziona la creazione di pagine in Flutter, la creazione di elementi, come scrivere uno specifico algoritmo in Flutter. Inoltre, conoscendo meglio il linguaggio Java [16] per lo sviluppo Android, abbiamo chiesto quali sono i corrispondenti di Flutter di alcuni elementi di Android, come i bottoni o una bottom/top bar, e come implementare alcune meccaniche presenti in Android [17], come i ViewModel o il meccanismo di Observer, in Flutter.

- **Suggerimenti per le funzionalità**

  Un altro ambito in cui ChatGPT dimostra particolare efficacia è nel suggerimento di funzionalità per l'applicazione, fornendo numerosi spunti creativi per la creazione di pagine. Nel nostro progetto, l'abbiamo impiegato principalmente per lo sviluppo delle pagine principali, consultandolo su come integrare al meglio un requisito specifico in una pagina, o su come implementare determinate funzionalità e quali approcci adottare per ottenere i risultati desiderati. In particolare, abbiamo chiesto consigli su quali altre funzionalità potesse avere una pagina, come ad esempio l'aggiunta di un meccanismo per segnalare se l'ultimo messaggio in una chat è stato letto o ancora no.

- **Implementazione delle funzionalità**

  ChatGPT ha fornito un grande aiuto nell'implementazione delle funzionalità, assistendo nella scrittura del codice per particolari requisiti complessi da implementare. Ad esempio, ha supportato l'integrazione dell'ImagePicker per la selezione delle immagini, la realizzazione di un meccanismo di caricamento progressivo degli elementi nella homepage, e l'implementazione di un sistema di ricerca degli item in base al loro nome, permettendo di cercare una stringa presente nel nome o nella descrizione. Inoltre, è stato molto utile nel calcolo della distanza tra utenti e oggetti. In quest'ultimo caso, dopo la richiesta, ChatGPT ha generato in linguaggio Dart la funzione di Haversine per calcolare la distanza tra due punti dati la loro latitudine e longitudine su un asse terrestre [18]:

```
double _calculateDistance(
    double latUser, double longUser, double latRent, double longRent) {
  const int earthRadiusKm = 6371; // Raggio medio della Terra in chilometri
  double lat1Rad = radians(latUser);
  double lat2Rad = radians(latRent);
  double lon1Rad = radians(longUser);
  double lon2Rad = radians(longRent);
  double deltaLat = lat2Rad - lat1Rad;
  double deltaLon = lon2Rad - lon1Rad;

  double a = sin(deltaLat / 2) * sin(deltaLat / 2) +
      cos(lat1Rad) * cos(lat2Rad) * sin(deltaLon / 2) * sin(deltaLon / 2);
  double c = 2 * atan2(sqrt(a), sqrt(1 - a));

  return earthRadiusKm * c;
}

double radians(double degrees) {
  return degrees * pi / 180;
}
```

Figura 4.1: Codice rappresentante la formula di Haversine

- **Design di pagine**

  ChatGPT è stato poi utilizzato per velocizzare il design delle pagine, permettendo tramite una semplice descrizione di costruire lo scheletro di una pagina, inserendo ogni widget necessario. Tuttavia, nonostante l'aiuto nell'inserimento degli elementi, che consente di risparmiare molto tempo, GPT e gli LLM mancano di sensibilità nel design effettivo. Infatti, dopo aver costruito la pagina con il codice generato, anche se non vi sono errori di formattazione o di elementi sovrapposti, c'è una totale mancanza di gusto e di design, generando una pagina che, dal punto di vista della UI e UX, risulta esteticamente poco piacevole. Questo comporta, nonostante l'aiuto nel trovare e inserire i widget necessari, una certa dose di lavoro per riuscire a dare una estetica piacevole alla pagina. Tuttavia, l'aiuto fornito nella costruzione della pagina contribuisce a risparmiare molto tempo, specialmente se non si conoscono bene i widget del linguaggio Flutter, rendendolo quindi molto utile per questo tipo di compiti, nonostante il punto di vista creativo.

- **Generazione di codice**

  Un'altra funzione molto utilizzata di ChatGPT, simile alla precedente nel design delle pagine, è la generazione di codice data una domanda specifica. Per risparmiare tempo, abbiamo spesso sfruttato ChatGPT per generare codice da copiare, nonostante questa sia una pessima pratica. Tuttavia, è stato fatto solo per quei codici che erano stati scritti molteplici volte e che, quindi, risultavano tediosi e ripetitivi. Per trasformare questa pratica da cattiva a ottima, è essenziale comprendere a fondo il codice prima di copiarlo, in modo da evitare errori e conoscere ciò che si sta scrivendo. Detto questo, posso affermare che la generazione di codice da parte di ChatGPT, come ad esempio cicli per il riempimento di liste o creazioni di *GridView* o *ListView* in Flutter, aiuta a risparmiare molto tempo, essendo questi codici scritti un gran numero di volte.

- **Ottimizzazione di codice**

  Un utilizzo in cui ChatGPT eccelle particolarmente è l'ottimizzazione del codice. Grazie a questa capacità, è possibile fornire a ChatGPT un codice che verrà ottimizzato rimuovendo bad practice o errori di sintassi, consentendo così di risparmiare tempo in futuro nella correzione di errori. Di seguito, viene riportato un esempio di una funzione ottimizzata da ChatGPT, il cui compito è semplicemente caricare un oggetto di noleggio in Firebase Database e la sua immagine su Firebase Storage.

  Da come si può notare dalla funzione prima e dopo l'elaborazione, ChatGPT ha contribuito significativamente all'ottimizzazione del codice. Ha rimosso il codice ridondante, spostandolo in altre funzioni e ha aggiunto un miglior controllo sugli errori, rendendo il codice più pulito ed efficiente:

```
@override
Future<String?> loadRental(Rental rental) async {
  try {
    final String databasePath = 'rentals';
    String imageUrl;
    await databaseReference
      .child(databasePath)8
      .child(rental.idToken)
      .set(rental.toMap());
    imageUrl = await uploadImage(rental.imagePath);
    await databaseReference
      .child(databasePath)
      .child(rental.idToken)
      .set(imageUrl);
    return 'Success';
  } catch (error) {
    return error.toString();
  }
}
```

Figura 4.2: Funzione di caricamento del noleggio prima dell'ottimizzazione

```
@override
Future<String?> loadRental(Rental rental) async {
  try {
    final String databasePath = 'rentals';
    String imageUrl = await uploadImage(rental.imagePath);
    rental.imageUrl = imageUrl; // Aggiungi l'URL dell'immagine all'oggetto
    Rental await databaseReference
        .child(databasePath)
        .child(rental.idToken)
        .set(rental.toMap());
    return 'Success';
  } catch (error) {
    print('Errore durante il caricamento del rental: $error');
    return 'Errore durante il caricamento del rental: $error';
  }
 }
```

Figura 4.3: Funzione di caricamento del noleggio dopo l'ottimizzazione

- **Correzione di errori**

Infine, un utilizzo in cui ChatGPT è stato impiegato in maniera estensiva è la correzione di errori. Fornendo il codice e la stampa dell'errore in runtime, ChatGPT offre una lista di possibili cause per il malfunzionamento del codice. In particolare, avendo accesso al

codice, ChatGPT correggeva i punti che avrebbero potuto generare quegli errori. Tuttavia, quando non si tratta di errori di sintassi o di errori semplici da risolvere, affidarsi a ChatGPT in maniera estensiva potrebbe risultare controproducente. Dopo diverse correzioni sullo stesso codice, nel caso non riesca a risolvere l'errore, ChatGPT potrebbe iniziare a mostrare allucinazioni[5], indicando errori che non esistono e modificando il codice in modo errato. Questo rende il codice sempre più problematico e lontano dalla soluzione desiderata. Nonostante ciò, quando ci si trova di fronte a problemi difficili da risolvere, utilizzare ChatGPT come supporto piuttosto che come unica risorsa può decisamente aiutare a identificare il problema, facilitando così la risoluzione di problemi ostici.

4.3.2 Vantaggi e limiti

L'utilizzo di ChatGPT nello sviluppo dell'applicazione ha mostrato diversi vantaggi significativi. Uno dei principali benefici è stata la capacità di ottimizzare il codice esistente, rendendolo più efficiente e meno soggetto a errori grazie alla rimozione di pratiche di programmazione non ottimali e all'introduzione di migliori controlli sugli errori. Questo ha comportato un risparmio di tempo e un miglioramento della qualità del codice. ChatGPT si è rivelato utile anche nella generazione di codice ripetitivo e nella creazione di strutture base delle pagine, permettendo una maggior concentrazione sulle funzionalità specifiche e meno sulle operazioni di base. Inoltre, la capacità di ChatGPT di fornire suggerimenti di design e funzionalità stimolando la creatività, ha contribuito a sviluppare un'interfaccia utente più completa e funzionale. Tuttavia, l'uso di ChatGPT ha anche evidenziato alcuni limiti. La generazione di codice, seppur utile, può diventare una cattiva pratica se non viene accompagnata da una piena comprensione del codice prodotto, portando a possibili problemi di manutenzione futura. Inoltre, la correzione degli errori, se affidata completamente a ChatGPT, può diventare problematica. In casi complessi, il modello può produrre allucinazioni, complicando ulteriormente la risoluzione dei problemi. Questo indica che, sebbene ChatGPT possa essere un prezioso strumento di supporto, non può sostituire completamente l'intervento umano, soprattutto nelle fasi critiche di debug e ottimizzazione del codice.

[5] "Allucinazione" negli LLM si riferisce alla produzione di informazioni errate o fuorvianti, dove il modello genera risposte che sembrano plausibili ma non sono basate su dati reali o verificati.

4.4 Testing

4.4.1 Ruolo di ChatGPT nel Testing

Durante la fase di testing, ChatGPT è stato utilizzato per creare test e risolvere errori relativi ai test falliti. In particolare, data la scarsa documentazione disponibile per il testing di Flutter e Firebase, ChatGPT è stato impiegato in maniera estensiva per documentarsi su come scrivere i casi di test e superare le difficoltà incontrate, fornendo nel dettaglio le librerie e i framework necessari per scrivere i test.

ChatGPT ci ha aiutato innanzitutto da un punto di vista organizzativo, supportandoci nella definizione di una struttura per i test e nella creazione di una serie di test generali per le funzionalità implementate. È stato particolarmente utile mantenere tutte le conversazioni il più possibile in una singola chat, permettendo a ChatGPT di acquisire una conoscenza di base della nostra applicazione. Questo approccio ci ha facilitato nella definizione di una serie completa di test per tutte le funzionalità implementate, garantendo una copertura più ampia e dettagliata.

In particolare, ChatGPT ci ha permesso di identificare e scrivere i molteplici casi di fallimento di un test, consentendoci di individuare tutte le varie combinazioni di test per un dato requisito, come ad esempio il login o la registrazione. Abbiamo potuto creare casi di test per ogni combinazione differente di errore, assicurando così una verifica approfondita e completa delle funzionalità critiche dell'applicazione.

Inoltre, ChatGPT ci ha permesso di risolvere diversi errori causati durante il testing, come i problemi di collegamento a Firebase e al suo mock, dovuti soprattutto alla mancanza di documentazione a riguardo. Ha sostituito efficacemente questa documentazione guidandoci passo dopo passo nel testing integrato con Firebase. Ad esempio, ChatGPT ci ha aiutato a capire come effettuare il setup dei mock di Firebase Storage e Firebase Database. Inoltre, ci ha guidato nel comprendere che avremmo dovuto sovrascrivere i metodi della classe FirebaseAuth per far funzionare correttamente i test:

```
@override
Future<UserCredential> signInWithEmailAndPassword({
  required String? email,
  required String? password,
}) =>
    super.noSuchMethod(
        Invocation.method(#signInWithEmailAndPassword, [email, password]),
        returnValue: Future.value(MockUserCredential()));

@override
Future<UserCredential> createUserWithEmailAndPassword({
  required String? email,
  required String? password,
}) =>
    super.noSuchMethod(
        Invocation.method(#createUserWithEmailAndPassword, [email, password]),
        returnValue: Future.value(MockUserCredential()));
```

Figura 4.4: Override delle funzioni di Firebase come consigliato da ChatGPT

4.4.2 Vantaggi e limiti

L'utilizzo di ChatGPT durante il testing ci ha permesso di documentarci e approfondire significativamente la nostra conoscenza sull'argomento, aiutandoci a comprendere il modo migliore per gestire la fase di testing nel ciclo di sviluppo di un'applicazione. Grazie a ChatGPT, abbiamo acquisito informazioni preziose su quali strumenti utilizzare per scrivere i test in Flutter e su come strutturare efficacemente i nostri test.

Tuttavia, come già accennato, la documentazione disponibile sul testing in Flutter è carente, e questa mancanza si riflette anche nelle risposte fornite da ChatGPT in molti casi. Nella nostra applicazione, siamo riusciti a testare solo le funzioni di registrazione e login, che hanno funzionato correttamente solo dopo numerosi tentativi e correzioni con l'aiuto di ChatGPT. Questo processo ha richiesto molto tempo e sforzi, ma alla fine siamo riusciti ad ottenere risultati positivi.

Per altri tipi di test, come il testing sul caricamento dei noleggi, l'uso di ChatGPT si è rivelato fallimentare a causa di continui errori nel tentativo di risolvere un problema nel collegamento del mock di Firebase Database[6] con Firebase. Abbiamo riscontrato difficoltà significative nel

[6] Nei test di Login e Registrazione, Firebase Database non è utilizzato essendo che viene solamente gestita l'autenticazione, viene utilizzato solamente Firebase Authentication e Firebase Storage, per questo motivo infatti questo problema non si presenta nei primi due casi di test.

far funzionare questi test, nonostante i suggerimenti e le correzioni forniti da ChatGPT. Questo è un chiaro esempio di come un affidamento continuo su ChatGPT per la risoluzione degli errori possa portare solo ad ulteriori problemi. Infatti, in questo caso, i tentativi di correzione hanno spesso introdotto nuovi errori, complicando ulteriormente la situazione.
La mancanza di documentazione ha reso questo problema particolarmente difficile da risolvere per noi, che, essendo principianti con Flutter, stavamo anche cercando di imparare e padroneggiare questo strumento durante il progetto, non permettendoci quindi di risolvere tali problemi.

4.5 Conclusione e statistiche

Durante il corso del nostro progetto, abbiamo esplorato e sfruttato ChatGPT come una risorsa chiave per supportare le diverse fasi di sviluppo dell'applicazione mobile. Lo abbiamo utilizzato in maniera estensiva, talvolta anche oltre il necessario, per apprendere appieno le potenzialità di questo strumento. Da questa esperienza, abbiamo tratto delle statistiche e valutazioni sull'utilità complessiva di ChatGPT, ottenendo preziose informazioni sul suo impatto nel processo di sviluppo.

4.5.1 Sintesi dell’utilizzo di ChatGPT

ChatGPT si è dimostrato estremamente utile nella fase di analisi dei requisiti, offrendo suggerimenti pertinenti che ci hanno aiutato a definire in modo chiaro e dettagliato le funzionalità necessarie per l'applicazione. Ha giocato un ruolo significativo anche durante l'implementazione delle funzionalità, fornendo soluzioni pratiche per ottimizzare il codice e migliorare le prestazioni dell'applicazione. Ad esempio, ci ha assistito nella generazione di codice ripetitivo, come la scrittura di widget, e nel design delle pagine, contribuendo a definire lo scheletro delle pagine sia da un punto di vista implementativo che creativo.

Tuttavia, è importante notare che ChatGPT ha mostrato alcune limitazioni, soprattutto in compiti avanzati come il testing e la correzione di errori complessi. Sebbene abbia aiutato nella strutturazione di casi di test di base e nella comprensione di come organizzare i test necessari, ha avuto difficoltà nel gestire scenari più intricati specifici dell'applicazione. Ad esempio, nella risoluzione di errori durante il testing, ChatGPT non è stato sempre in grado di fornire soluzioni efficaci, specialmente in assenza di documentazione specifica o per casi particolarmente complessi.

In conclusione, l'utilizzo di ChatGPT è stato prezioso nel supportare varie fasi di sviluppo del progetto, fornendo suggerimenti creativi e pratici. Nonostante le limitazioni riscontrate, il contributo di ChatGPT nel definire requisiti e ottimizzare l'implementazione è stato significativo, dimostrandosi un'utile risorsa per migliorare l'efficienza e la qualità del nostro lavoro di sviluppo.

4.5.2 Statistiche sull'utilizzo di ChatGPT

Durante il periodo di sviluppo, ChatGPT è stato utilizzato principalmente per la parte di implementazione, essendo la fase più lunga e complessa. Successivamente, è stato impiegato per l'analisi dei requisiti, e infine, per il testing, la quale è stata la fase in cui è stato utilizzato di meno. Tuttavia, considerando la durata del ciclo di vita del progetto e l'uso complessivo di ChatGPT, si può affermare che la fase di analisi dei requisiti è stata quella in cui ChatGPT ha avuto un impatto maggiore. Senza la necessità di scrivere codice o correggere errori, ChatGPT ci ha permesso di definire prima uno schema generale dei requisiti e poi di dettagliare ciascun requisito, aiutandoci anche a comprendere come strutturare l'architettura dell'applicazione.
Se dovessimo valutare l'utilizzo di ChatGPT da un punto di vista statistico, si potrebbe dire che è stato utilizzato circa l'80% del tempo durante la fase di analisi dei requisiti, il 50% durante la fase di implementazione, e il 30% durante la fase di testing.

4.5.3 Valutazione Globale dell'Utilità

ChatGPT si è rivelato uno strumento prezioso durante tutto il ciclo di sviluppo della nostra applicazione mobile, offrendo supporto in vari ambiti e dimostrando una certa versatilità. In fase di analisi dei requisiti, ChatGPT ha fornito un aiuto significativo, facilitando la comprensione e la definizione dettagliata delle funzionalità necessarie. Ha offerto suggerimenti creativi che hanno arricchito il nostro progetto, permettendoci di esplorare soluzioni che non avremmo considerato inizialmente.
Durante l'implementazione, ChatGPT ha continuato a essere un valido aiuto. Ha contribuito a ottimizzare il codice e a risolvere problemi complessi, aumentando l'efficienza e riducendo il tempo necessario per completare le varie attività. L'assistenza nella generazione di codice ripetitivo, la costruzione di strutture di base per le pagine e il design ha dimostrato quanto possa essere utile per accelerare il processo di sviluppo, soprattutto per sviluppatori meno esperti o in contesti dove la documentazione è scarsa.

Nel testing, ChatGPT ha mostrato alcune limitazioni. Mentre è stato efficace nel suggerire approcci generali e nel definire i casi di test di base, ha faticato a gestire situazioni più complesse e specifiche della nostra applicazione.

In sintesi, ChatGPT ha dimostrato di essere uno strumento versatile e potente, capace di migliorare la produttività e la creatività in molte fasi del ciclo di sviluppo. Tuttavia, il suo utilizzo ottimale richiede un equilibrio, sfruttandone i punti di forza senza dipendere eccessivamente da esso per compiti che richiedono un giudizio umano avanzato o una comprensione profonda delle specifiche tecnologie utilizzate.

Capitolo 5

Conclusioni

5.1 Sommario del progetto

Il progetto ha riguardato lo sviluppo di un'applicazione mobile utilizzando Flutter e Firebase, con il supporto di ChatGPT per le diverse fasi del ciclo di vita del software. L'obiettivo principale era esplorare l'efficacia di ChatGPT nell'assistere lo sviluppo, dalla definizione dei requisiti fino al testing finale, valutandone anche le sue capacità nell'aiutarci ad apprendere un nuovo strumento di sviluppo come Flutter.

5.2 Lezioni apprese

Durante il progetto, abbiamo imparato molte lezioni preziose. Una delle principali è l'importanza di avere una chiara comprensione dei requisiti prima di iniziare l'implementazione. ChatGPT ci ha aiutato a strutturare questi requisiti in modo dettagliato. Inoltre, abbiamo appreso che, mentre ChatGPT è un ottimo strumento per risolvere problemi e generare codice, non può sostituire però la necessità di una comprensione profonda delle tecnologie e degli strumenti utilizzati, necessitando soprattutto una costante supervisione.

5.3 Miglioramento delle competenze tecniche

L'utilizzo di ChatGPT ha notevolmente migliorato le nostre competenze tecniche. Abbiamo acquisito una migliore comprensione di Flutter e Firebase, grazie ai suggerimenti e alle soluzioni proposte da ChatGPT. Inoltre, il processo di revisione e implementazione delle soluzioni suggerite ci ha permesso di consolidare la nostra conoscenza pratica delle best practice di sviluppo e delle tecniche di ottimizzazione del codice.

5.4 Gestione delle problematiche

Durante lo sviluppo, abbiamo incontrato diverse problematiche, soprattutto nella fase di testing. ChatGPT ci ha aiutato a risolvere molte di queste, fornendo suggerimenti su come strutturare i test e risolvere errori comuni. Tuttavia, ci siamo anche resi conto dei limiti di ChatGPT, specialmente quando si trattava di problemi specifici e complessi legati alla nostra applicazione. Questo ci ha insegnato l'importanza di bilanciare l'utilizzo di strumenti di supporto con l'expertise umana, senza fare completo affidamento a ChatGPT.

5.5 Efficienza e produttività

L'integrazione di ChatGPT nel nostro flusso di lavoro ha aumentato significativamente la nostra efficienza e produttività. La possibilità di generare rapidamente codice e ottenere risposte a domande tecniche ha ridotto i tempi di sviluppo. Tuttavia, abbiamo riconosciuto la necessità di rivedere e comprendere a fondo le soluzioni proposte per evitare errori e garantire la qualità del codice. In diverse occasioni, è stato necessario tornare al punto di partenza dei codici scritti da ChatGPT, poiché questi generavano più errori di quelli che avrebbero dovuto risolvere.

5.6 Valutazione dell'utilità di ChatGPT

In generale, ChatGPT si è dimostrato uno strumento molto utile, specialmente nelle fasi di analisi dei requisiti e implementazione. Ha facilitato la comprensione dei problemi e la generazione di soluzioni, migliorando la nostra produttività. Tuttavia, durante l'implementazione, ha mostrato alcuni limiti nel design delle pagine: pur aiutando a risparmiare tempo implementando lo scheletro delle pagine, il risultato finale spesso mancava di estetica e di un design efficace. Inoltre, la sua utilità è stata limitata nella fase di testing, dove problemi specifici richiedevano una comprensione più approfondita e un approccio più mirato. Questo ha evidenziato che ChatGPT è uno strumento di supporto potente, ma non può sostituire completamente la necessità di competenze tecniche e di una revisione umana accurata.
Infine, un elemento in cui ChatGPT eccelle è la creatività, offrendo consigli innovativi per i requisiti o per il design delle pagine. Questo supporto creativo è stato particolarmente utile nei momenti di blocco, fornendo spunti interessanti e soluzioni alternative che hanno arricchito il nostro processo di sviluppo.

5.7 Conclusioni finali

In conclusione, l'integrazione di ChatGPT nel nostro progetto ha fornito numerosi vantaggi, migliorando la nostra efficienza e le nostre competenze tecniche. Abbiamo imparato a sfruttare al meglio le sue capacità, riconoscendo al contempo i suoi limiti. Questo progetto ha dimostrato che, con un uso equilibrato e consapevole, ChatGPT può essere un alleato prezioso nello sviluppo di applicazioni complesse. Esso supporta i team nello svolgimento di compiti ripetitivi e nella risoluzione di problemi, richiedendo comunque sempre l'intervento e il giudizio umano per le decisioni critiche e le situazioni più complesse. ChatGPT si è inoltre rivelato particolarmente utile dal punto di vista creativo, offrendo spunti e soluzioni innovative.

Bibliografia

[1] ChatGPT. *GPT tools.* Url: *https://openai.com/chatgpt/*

[2] Wikipedia. *Analisi dei requisiti.* Url: *https://it.wikipedia.org/wiki/Analisi_dei_requisiti*

[3] Larman, C. (2015). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development.* Pearson. (p. 110).

[4] API. *Cosa è un'API.* Url: *https://aws.amazon.com/it/what-is/api/*

[5] Flutter. *Flutter development documentation.* Url: *https://flutter.dev/development*

[6] Dart. *Dart development documentation.* Url: *https://dart.dev/guides*

[7] Google Firebase Authentication. *Google Firebase Authentication documentation.* Url: *https://firebase.google.com/docs/auth?hl=it*

[8] Google Firebase Realtime Database. *Google Firebase Realtime Database documentation.* URL: *https://firebase.google.com/docs/database?hl=it*

[9] Google Firebase Storage. *Google Firebase Storage documentation.* Url: *https://firebase.google.com/docs/storage?hl=it*

[10] Stripe. *Online Payment Processing for Internet Businesses.* URL: *https://docs.stripe.com/*

[11] MVVM. *Model-view-viewmodelgn Pattern.* Url: *https://it.wikipedia.org/wiki/Model-view-viewmodel*

[12] Flutter widgets. *Flutter widgets catalog.* Url: *https://docs.flutter.dev/ui/widgets*

[13] CRUD. *Cosa sono le operazioni CRUD.* Url: *https://it.wikipedia.org/wiki/CRUD*

[14] Testing. *Cos'è il test del software.* Url: *https://www.ibm.com/it-it/topics/software-testing*

[15] Flutter testing. *Flutter testing documentation.* Url: *https://docs.flutter.dev/testing/overview*

[16] Java. *Java (Programming language).* Url: *https://it.wikipedia.org/wiki/Java_(linguaggio_di_programmazione)*

[17] Android Studio. *Developer Android Studio.* Url: *https://developer.android.com/studio?hl=it*

[18] Formula di Haversine. *Haversine formula.* Url: *https://en.wikipedia.org/wiki/Haversine_formula*