

Report Progetto

Corso: 66860 - LABORATORIO DI APPLICAZIONI MOBILI



LIVECHAT

MANUEL ARTO 974775

manuel.arto@studio.unibo.it

NAPOLI ANDREA 988997

andrea.napoli4@studio.unibo.it

July 9, 2023

Contents

1	Introduzione	2
1.1	Funzionalità	3
2	Panoramica dell'App	4
2.1	Login	5
2.2	Home	6
2.3	Chat	8
2.4	Map	10
2.5	Friends	11
2.6	Profile	13
3	Scelte Progettuali	16
3.1	Flutter	16
3.1.1	Persistent Data - Isar	16
3.1.2	State Management - Provider	17
3.2	Storage immagini profilo	18
3.3	Divisione e struttura del codice	19
3.4	Dati real-time	20
3.4.1	Scambio messaggi	20
3.4.2	Posizione	21
3.4.3	Contapassi	22

1 Introduzione

Livechat è un'applicazione mobile di messaggistica istantanea e funzionalità social avanzate come la condivisione della posizione e una classifica basata sul numero di passi effettuati giornalmente.

L'obiettivo principale di Livechat è creare un ambiente interattivo e coinvolgente, in cui gli utenti possono comunicare tra loro, organizzare le loro conversazioni tramite etichette e condividere file, immagini e messaggi audio.

Livechat permette la creazione di un profilo utente che avviene tramite autenticazione al primo avvio dell'app. Successivamente l'utente può inviare richieste di amicizia, visualizzare la posizione dei propri amici e monitorare i loro passi.

Uno dei più importanti aspetti dell'applicativo riguarda la reattività istantanea dei dati personali e dei propri amici. Grazie ai sensori del dispositivo e all'utilizzo del protocollo websocket è stato possibile realizzare un ambiente totalmente dinamico che reagisce a qualunque input generato dai propri amici come: invio di messaggi, cambio di posizione e incremento dei passi.

L'applicazione è stata sviluppata utilizzando Flutter, un framework che ci ha permesso di realizzare un'applicazione nativa per Android e iOS, che dopo una fase di login/registrazione, permette agli utenti di accedere ai vari servizi offerti.

Ecco il link da cui scaricare l'APK per la versione Android: [link](#)

1.1 Funzionalità

L'applicazione fornisce le seguenti funzionalità principali:

- **Autenticazione e gestione del profilo utente**

Gli utenti possono creare un account inserendo le informazioni richieste e accedere successivamente utilizzando le credenziali registrate.

Dopo l'autenticazione viene registrato un token JWT all'interno dell'applicazione, che verrà poi utilizzato per autenticare le successive richieste effettuate al server.

Una volta autenticati è possibile visualizzare i propri amici o cercarne di nuovi, visualizzando i contatti iscritti all'app o utilizzando la barra di ricerca inserendo il nome utente.

- **Scambio di messaggi tramite conversazioni private**

È possibile comunicare con i propri amici tramite chat private in tempo reale.

Gli utenti possono inviare e ricevere testo, file, immagini (tramite utilizzo della fotocamera) e messaggi audio (tramite utilizzo del microfono).

Inoltre è stato implementato un sistema di notifiche per ogni messaggio in arrivo.

- **Condivisione della posizione**

È possibile condividere la propria posizione e visualizzare quella di ogni amico su una mappa interattiva in tempo reale. Nel caso un utente sia offline, verrà visualizzata l'ultima posizione registrata.

- **Condivisione dei passi effettuati**

L'applicazione registra il numero di passi effettuati dagli utenti utilizzando il sensore contapassi del dispositivo. Gli utenti possono visualizzare il numero di passi effettuati durante la giornata e durante la settimana.

Inoltre, è possibile visualizzare una classifica giornaliera o settimanale basata sul numero totale di passi effettuati tra gli amici dell'utente.

2 Panoramica dell'App

Al primo accesso all'app, verrà richiesto all'utente di autenticarsi per accedere alle funzionalità. Una volta effettuato l'accesso, si aprirà la schermata principale, nota come *Home*. Questa schermata è il punto di partenza dell'applicazione e fornisce un'interfaccia intuitiva per navigare tra le diverse sezioni disponibili.

L'app è suddivisa in 5 sezioni principali: Home, Chat, Map, Friends e Profile.

Queste sezioni consentono all'utente di accedere a diverse funzionalità dell'app e di interagire con altri utenti.

Per spostarsi tra le sezioni, l'utente può utilizzare la bottom bar posizionata nella parte inferiore dello schermo. La bottom bar fornisce icone rappresentative di ciascuna sezione e permette di passare facilmente da una all'altra.

In alternativa, l'utente può anche navigare tra le sezioni utilizzando le card presenti all'interno della Home. Queste card forniscono un'anteprima delle funzionalità presenti nelle altre sezioni e consentono di accedervi direttamente tramite un semplice tocco.

Abbiamo inoltre introdotto una funzionalità che consente agli utenti di personalizzare il tema dell'applicazione in base alle loro preferenze, oppure passare facilmente dalla modalità chiaro (light mode) o alla modalità scuro (dark mode). Di seguito sono presentate alcune immagini di esempio:

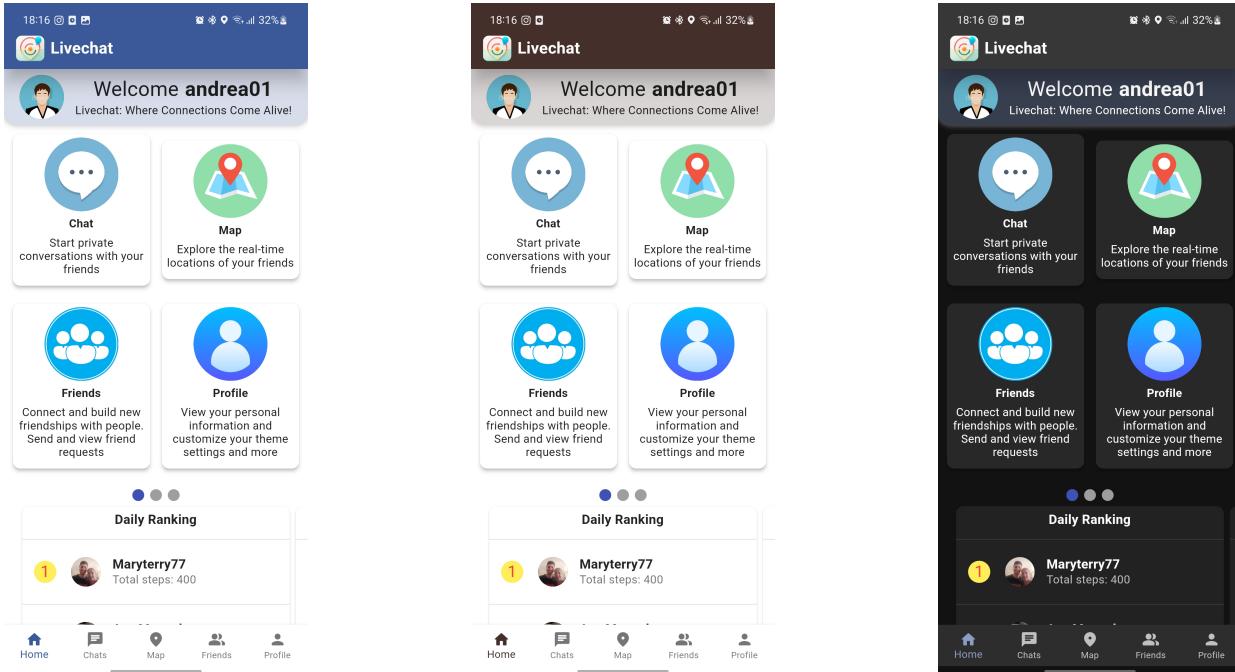


Figure 1: Personalizzazione del tema

Procediamo con un'analisi dettagliata delle sezioni principali di Livechat.

2.1 Login

Questa sezione offre agli utenti la possibilità di autenticarsi tramite login o creazione di un nuovo account. Abbiamo introdotto dei controlli per garantire la correttezza dei dati inseriti nel form:

- L'username deve avere una lunghezza compresa tra 6 e 30 caratteri
- L'email deve essere inserita nel formato corretto (mario@test.com)
- la password deve contenere almeno 8 caratteri alfanumerici
- È obbligatorio fornire un'immagine. Nel caso in cui l'utente non desideri aggiungere un'immagine personale, il sistema selezionerà automaticamente un'immagine predefinita in base al sesso selezionato.

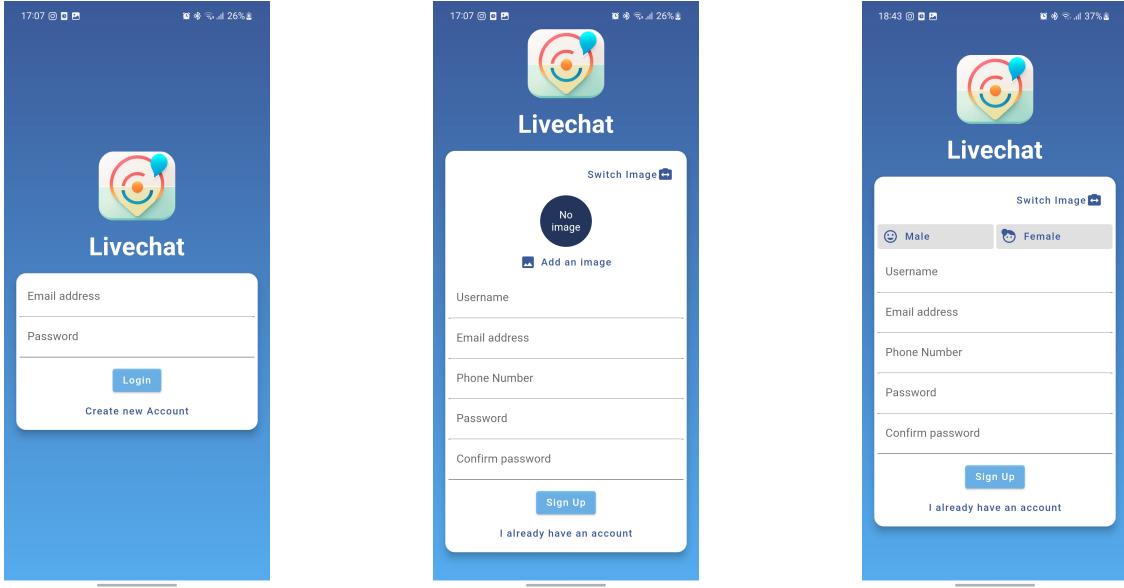


Figure 2: Login/Registrazione

2.2 Home

La sezione Home rappresenta la pagina principale dell'applicazione, offrendo un'ampia panoramica dei dati e fornendo un punto di partenza per la navigazione tra le diverse sezioni disponibili. Questa schermata è concepita per offrire agli utenti un quadro completo e intuitivo delle informazioni essenziali, consentendo loro di avere un'idea chiara dei dati e delle funzionalità a disposizione.

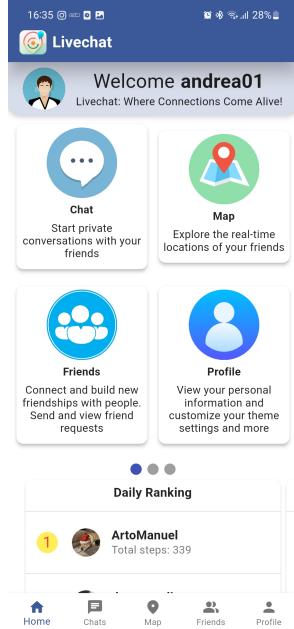


Figure 3: Home screen

Inoltre abbiamo utilizzato un meccanismo di scorrimento laterale per visualizzare le varie anteprime contenenti tutte le informazioni del nostro profilo. Sono 3 le preview principali:

- **Classifica giornaliera**

Permette di visualizzare la propria posizione in classifica insieme a quella dei due amici con più passi. Inoltre, all'interno è presente un collegamento che permette

di visualizzare la classifica totale giornaliera e settimanale.

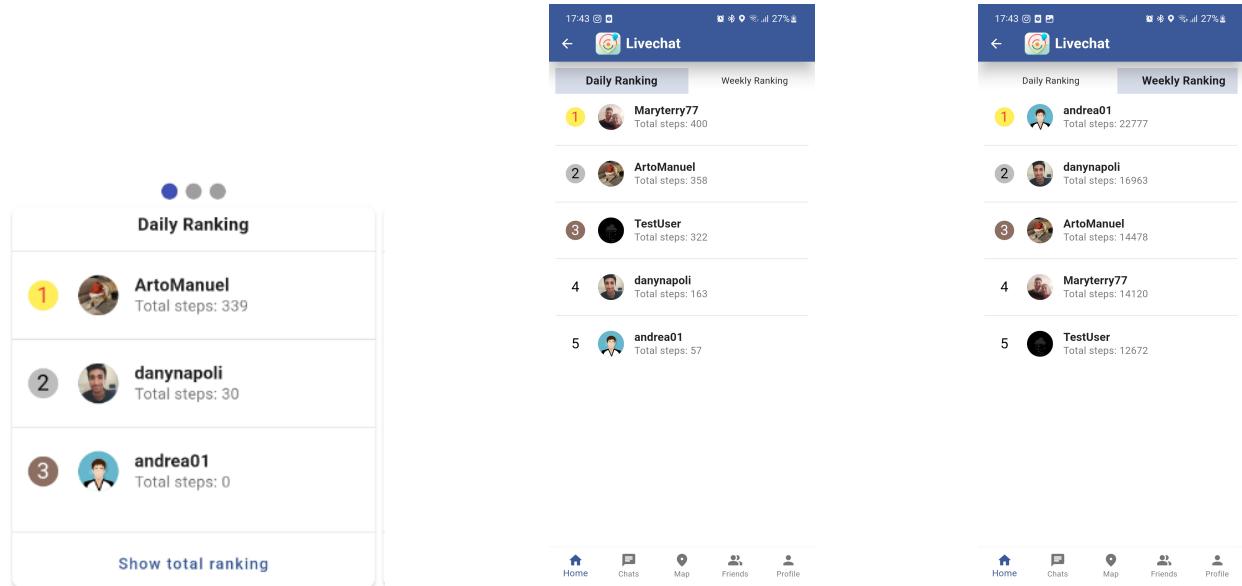


Figure 4: Anteprima e classifica totale

• Passi giornalieri

Attraverso un grafico circolare è possibile visualizzare il rapporto tra il numero di passi effettuati e l'obiettivo che l'utente si è prefissato (personalizzabile nella sezione profilo).

Inoltre è presente un collegamento che permette di visualizzare un grafico a barre contenente il numero di passi effettuati nella settimana passata.

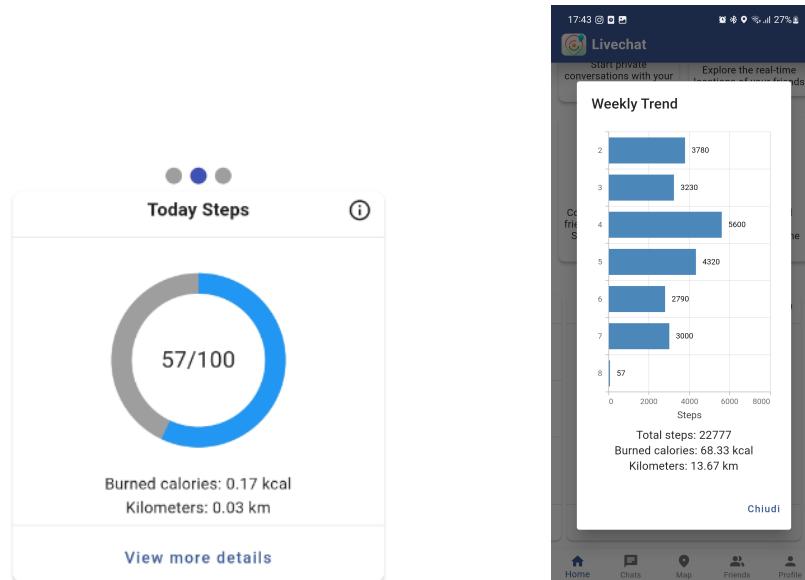


Figure 5: Passi giornalieri e grafico settimanale

- **Anteprima delle chat**

Attraverso questa card è possibile visualizzare un'anteprima delle chat, contenente le 3 conversazioni più recenti. In fondo alla card è presente un collegamento per poter navigare all'interno della pagina Chat.

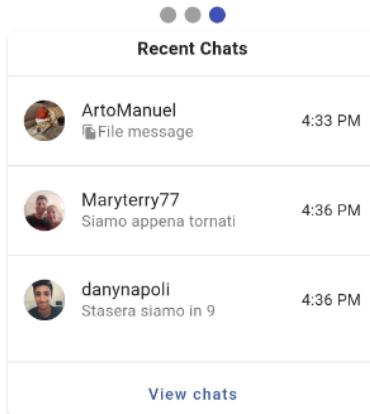


Figure 6: Anteprima delle chat

2.3 Chat

La chat rappresenta uno dei punti forti della nostra applicazione, offrendo agli utenti un modo semplice ed efficace per comunicare in tempo reale. La sua interfaccia intuitiva e user-friendly permette agli utenti di inviare e ricevere messaggi testuali, immagini, file e audio, rendendo la comunicazione ancora più ricca e coinvolgente.

Nella parte superiore dell'interfaccia è possibile notare una lista degli utenti attualmente connessi per capire chi è disponibile per una chat instantanea.

Per creare una nuova sezione, basta premere sul bottone apposito posizionato sulla destra dell'interfaccia della chat. Una volta cliccato, si aprirà un dialog che ti permetterà di assegnare un nome alla nuova sezione. Potrai scegliere un nome significativo o utilizzare una denominazione che rifletta il contenuto delle chat che desideri inserire in quella sezione specifica.

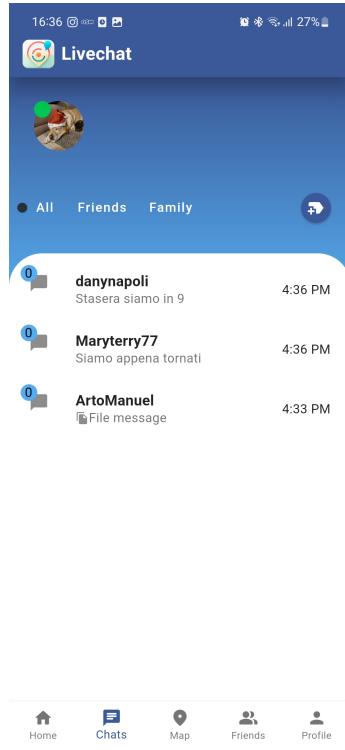


Figure 7: Pagina principale della chat

Per eliminare una sezione, basta tenere premuto a lungo su di essa. Apparirà un’alert per chiedere conferma dell’eliminazione, consentendoti di rimuoverla dalla lista delle sezioni disponibili. La sezione predefinita *All* contiene tutte le chat e di conseguenza non può essere eliminata.

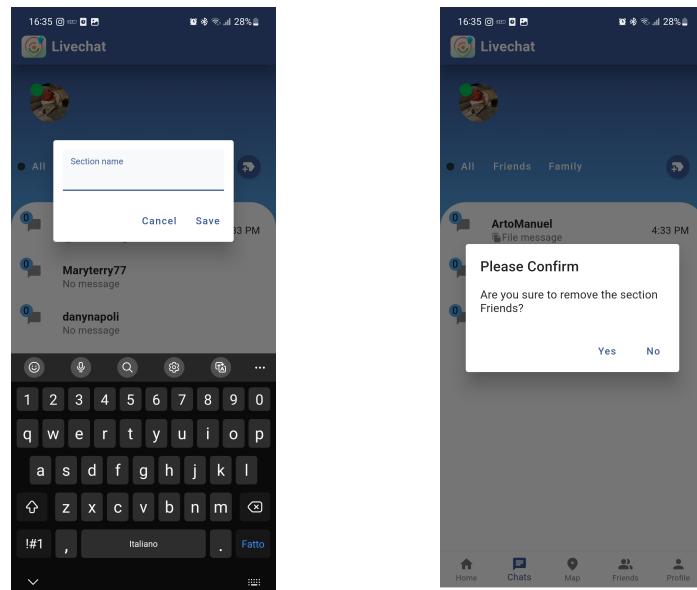


Figure 8: Creazione ed eliminazione di una sezione

All'interno di ciascuna chat, è presente una barra che permette di digitare e inviare messaggi di testo in modo rapido e semplice. A sinistra della barra è presente il bottone che permette di mandare immagini scattate al momento o prelevate dalla galleria e il bottone per inoltrare file.

Infine sulla destra si può trovare il pulsante che consente di registrare messaggi vocali.

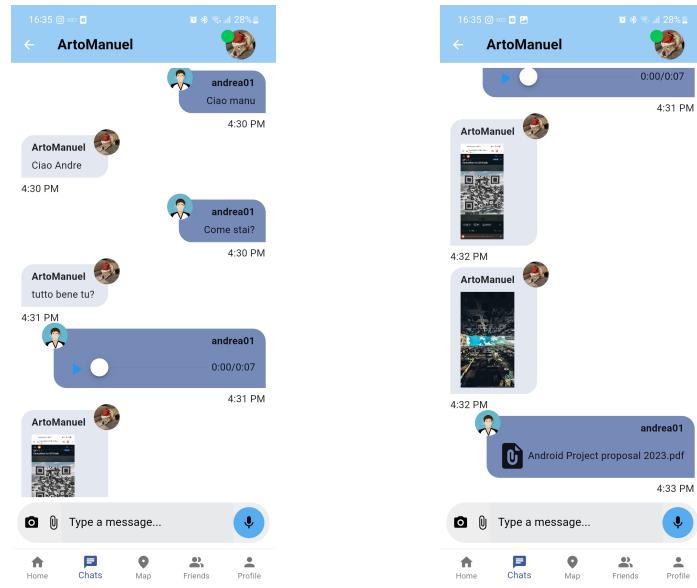


Figure 9: Esempio di una conversazione

2.4 Map

Un'altra funzionalità implementata è la mappa, che consente di visualizzare in tempo reale la posizione dei propri amici. Ogni amico è rappresentato da un marker e premendo su di esso è possibile visualizzare il numero di passi effettuati. In basso a destra è presente un pulsante che permette di riposizionare automaticamente la mappa sulle coordinate del proprio utente.

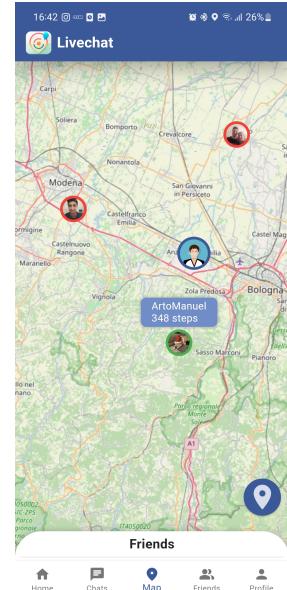


Figure 10: Map screen

Scorrendo verso l’alto si aprirà un comodo menu a comparsa che elenca tutti gli amici presenti. Premendo su un singolo amico, la mappa si sposterà automaticamente sulla posizione corrispondente dell’amico selezionato. Questa funzionalità semplifica la navigazione tra i propri amici e permette di concentrarsi su una specifica posizione della mappa.

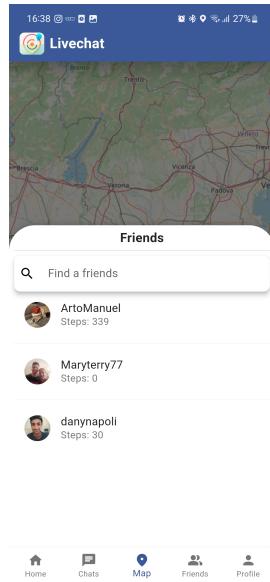


Figure 11: Menu mappa amici

2.5 Friends

La sezione relativa agli amici rappresenta un elemento fondamentale all’interno della nostra applicazione, poiché permette agli utenti di connettersi con nuove persone e avviare conversazioni o gareggiare nelle classifiche.

La pagina degli amici è strutturata in modo intuitivo e suddivisa principalmente in tre sezioni principali:

- **Friends:** viene visualizzata la lista dei propri amici;
- **Suggested:** è suddivisa in due parti, la prima mostra i contatti iscritti alla piattaforma e la seconda presenta invece suggerimenti di utenti presenti sull’applicazione, che potrebbero essere potenziali nuovi amici;
- **Requests:** comprende le richieste di amicizia ricevute in attesa di essere accettate e le richieste che l’utente ha inviato;

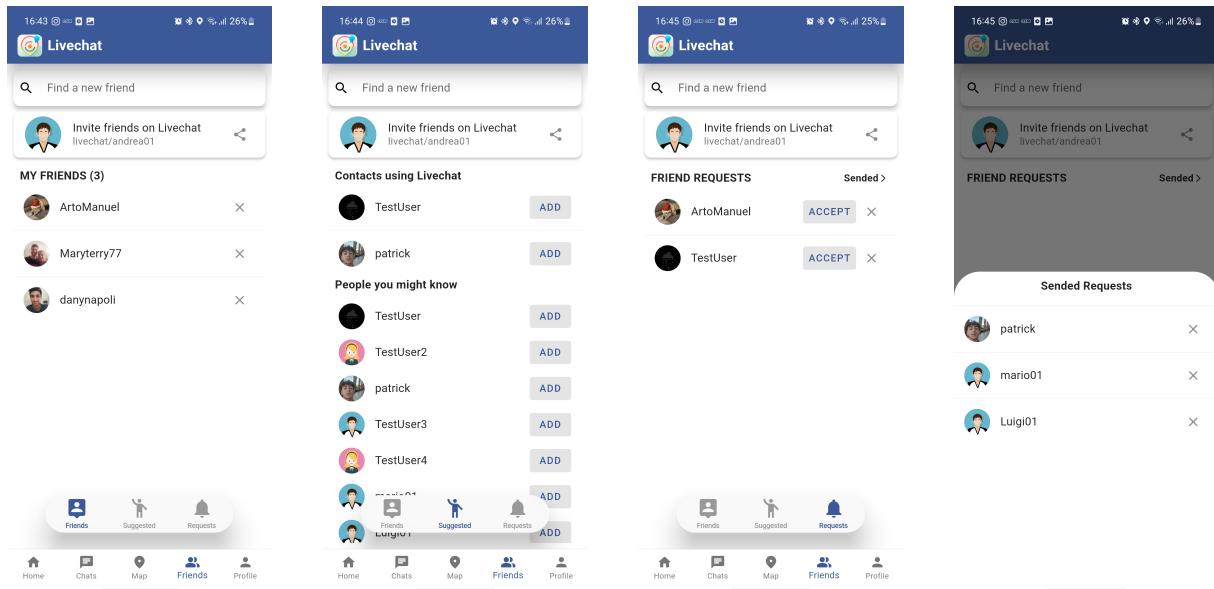


Figure 12: Friends, Suggested, Requests e Sended

Gli utenti hanno diverse possibilità per trovare nuove amicizie: oltre alla section *suggested*, si possono cercare nuovi utenti attraverso la barra di ricerca posta in alto. È inoltre possibile condividere un link d’invito per il download dell’applicazione, direttamente dalla stessa pagina.

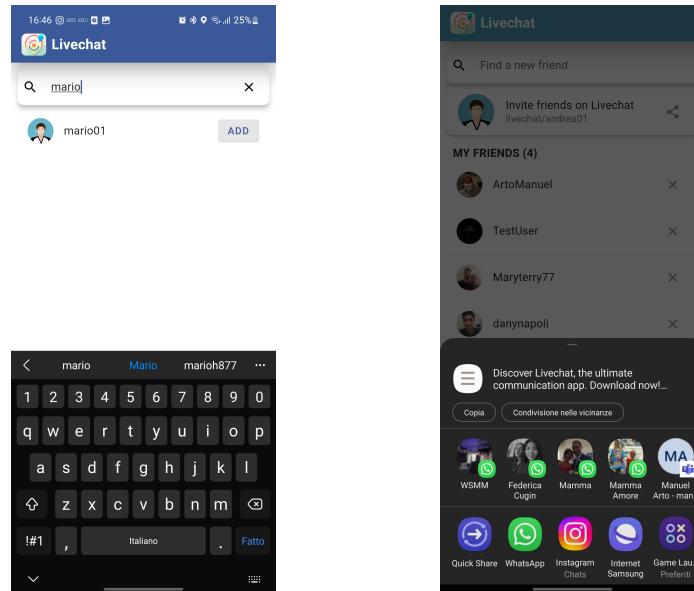


Figure 13: Ricerca e condivisione

2.6 Profile

La sezione *Profile* permette di gestire le impostazioni dell'applicazione e di visualizzare i dati con cui ci siamo registrati.

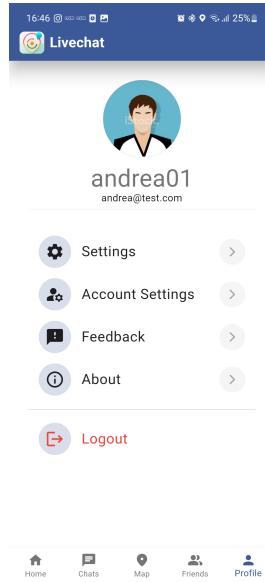


Figure 14: Pagina profilo

Andiamo ad analizzare le varie componenti nello specifico:

- **Settings**

Sono presenti le impostazioni dell'app. È possibile abilitare o disabilitare la dark mode, modificare l'obiettivo giornaliero dei passi e personalizzare il tema.

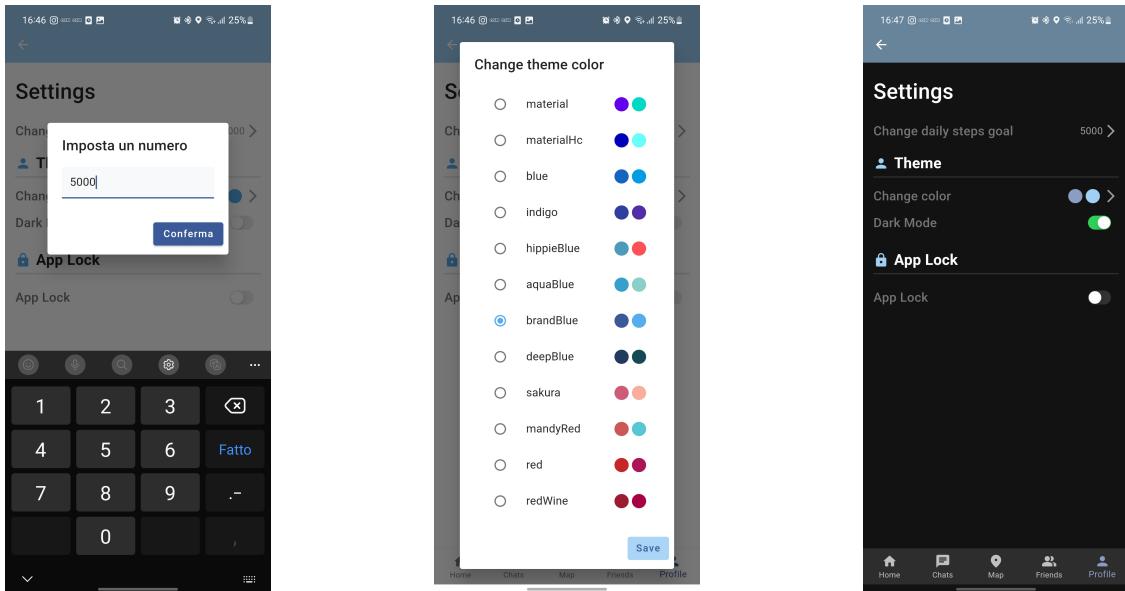


Figure 15: Cambio obiettivo, tema e dark mode

- **Account Settings**

È possibile visualizzare i dati utilizzati durante la fase di registrazione: username, email, numero di telefono e foto profilo;

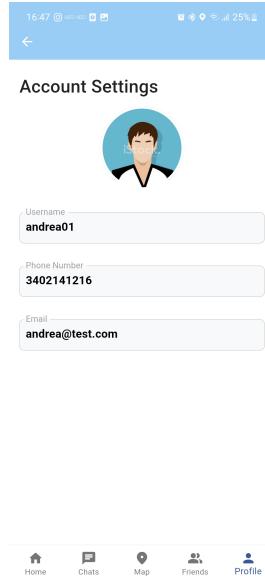


Figure 16: Impostazioni Utente

- **Feedback**

Permette agli utenti di lasciare una recensione sull'utilizzo dell'applicazione insieme a un voto;

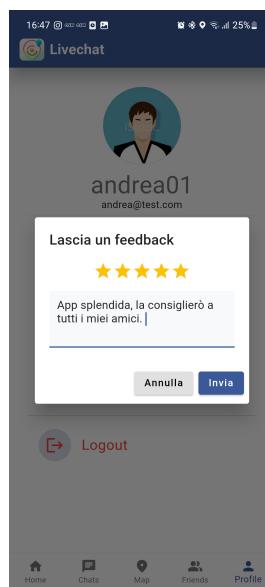


Figure 17: Lasciare un feedback

- **About**

Dialog contenente informazioni generali sull'applicazione, funzionalità principali, il numero della versione, gli autori e la data di rilascio;



Figure 18: About dialog

3 Scelte Progettuali

3.1 Flutter

Per lo sviluppo dell'applicazione è stato utilizzato il framework Flutter, un framework open-source sviluppato da Google per la creazione di applicazioni cross-platform. Utilizza un linguaggio di programmazione moderno (*Dart*) che semplifica lo sviluppo dell'applicazione e permette, partendo da uno stesso codebase, di generare applicazioni native per diverse piattaforme.

La scelta di utilizzare Flutter è stata motivata dall'alta qualità degli strumenti e utility che mette a disposizione e dalle ottime performance che riesce ad ottenere.

A differenza dei suoi competitor Flutter utilizza un motore di rendering personalizzato per controllare ogni pixel dello schermo sul quale *disegna* gli elementi a runtime. Grazie a questo approccio, Flutter permette di ottenere alte prestazioni mantenendo un framerate costante di 60 fps e implementa inoltre altre funzionalità tra cui caching e modalità di rebuild dinamiche.

3.1.1 Persistent Data - Isar

È stato deciso di implementare un sistema di salvataggio dati sul dispositivo che permetta all'utente di rimanere loggato, visualizzare le chat passate e molto altro.

Per la gestione dei dati persistenti è stato utilizzato il package *Isar*, un database NoSQL open-source che offre prestazioni elevate e una facile integrazione con il framework. Permette infatti di aggiungere una qualunque classe alle insieme di *collections* gestite da Isar, fornendo poi una serie di api per l'ottenimento o inserimento dei dati.

La scelta di utilizzare Isar come database è stata motivata dalla sua semplicità d'uso ma soprattutto dalle ottime performance che riesce ad ottenere in confronto ai suoi competitor.

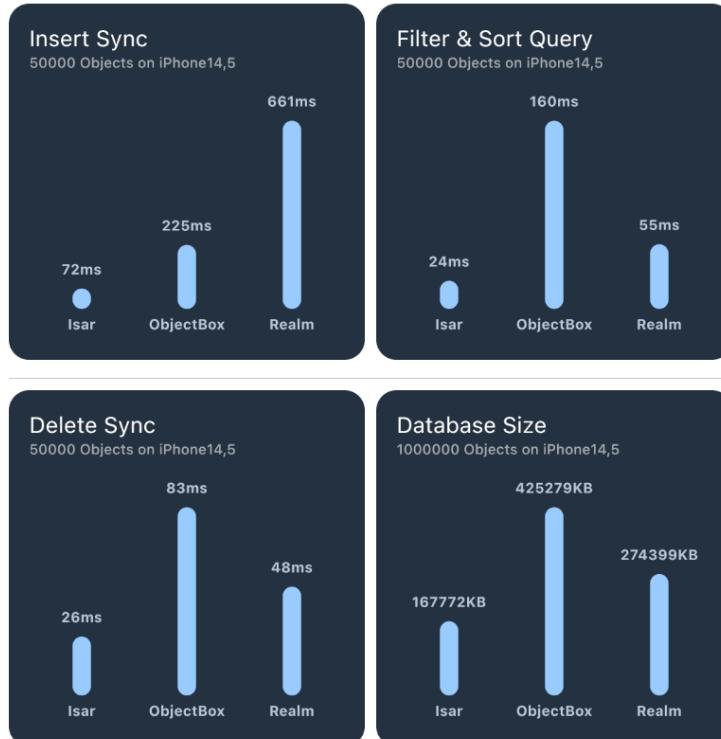


Figure 19: Isar benchmark

Di seguito un elenco delle classi relative ai dati memorizzati sul dispositivo:

- **AuthUser:** gestisce le informazioni dell’utente loggato e il token JWT utilizzato durante le chiamate al server. Nota più importante riguarda l’id dell’ utente, utilizzato dalle classi successive per permettere una gestione multi-user dell’app e caricare solo i dati dell’utente attualmente loggato;
- **Chat:** gestisce le informazioni relative alle chat dell’utente, come la lista di sezioni a cui fa parte e la lista di messaggi in ordine cronologico;
- **Section:** gestisce le section create dall’utente, oltre a quella di default *All*;
- **Steps:** gestisce le informazioni relative ai passi effettuati durante la giornata, si resetta al primo avvio giornaliero dell’app;
- **Settings:** gestisce le informazioni relative al tema dell’app e all’abilitazione della dark mode;

3.1.2 State Management - Provider

Per gestire lo stato dell’applicazione e la condivisione dei dati tra i vari componenti, è stato utilizzato il package *Provider*, un package di state management molto popolare

in Flutter che offre un modo semplice e scalabile per gestire e condividere lo stato tra i widget.

La scelta di utilizzare Provider come soluzione è stata guidata dalla sua facilità d'uso e dalla sua flessibilità. Ha permesso di mantenere un'architettura pulita e separare la logica dell'applicazione dalla parte di UI.

Sono stati utilizzati provider per la fruizione da tutta l'app dei seguenti dati: utente loggato, chat caricate, amici e richieste di amicizia, passi effettuati, posizione dell'utente e impostazioni personalizzabili dell'app.

3.2 Storage immagini profilo

Per la gestione delle immagini dei profili utente è stato utilizzato il servizio cloud Firebase Storage, che offre uno spazio di archiviazione affidabile e scalabile per l'archiviazione delle immagini.

La scelta di utilizzare Firebase Storage è stata motivata dalla sua facilità d'uso e dall'integrazione diretta con l'applicazione backend python realizzata. Ha permesso di archiviare e recuperare facilmente le immagini profilo degli utenti.

3.3 Divisione e struttura del codice

Durante lo sviluppo dell'applicazione, è stata adottata una struttura ben definita per garantire un codice pulito e facilmente estendibile. Sono stati utilizzati design pattern come il pattern MVC (Model-View-Controller) e il pattern Provider per separare la logica di business dalla parte di UI e migliorare la modularità dell'applicazione.

- **models:** sono presenti le classi relative ai dati utilizzati nell'applicazione;
- **providers:** sono presenti i vari providers, che forniscono un punto di accesso e manipolazione dei dati. Sono responsabili della quasi totale logica di business dell'applicativo;
- **screens:** sono presenti le schermate principali dell'applicazione. Ogni schermata rappresenta una parte dell'interfaccia utente e contiene la logica di visualizzazione dei dati e di gestione delle interazioni dell'utente;
- **widgets:** questa directory contiene una serie di widget utilizzati da più parti nell'applicazione come la bottom bar o la barra di ricerca;
- **services:** i services forniscono funzionalità specifiche come la gestione dei file, le richieste al backend, l'accesso al database Isar e i servizi di notifica. Questi servizi consentono di separare la logica di basso livello dal resto dell'applicazione;

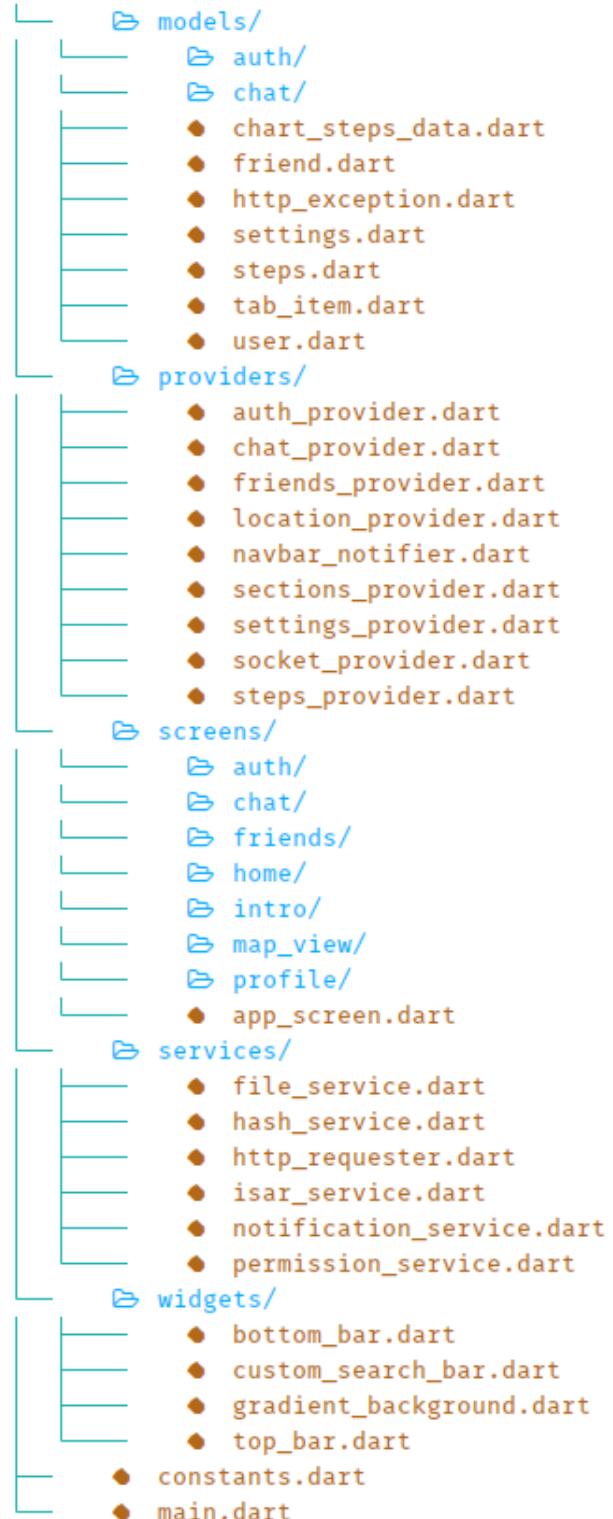


Figure 20: Project structure

3.4 Dati real-time

Come accennato durante l'introduzione, una delle caratteristiche più esaltanti di Livechat è la capacità di fornire una reattività istantanea dei dati personali e dei propri amici.

Tale funzionalità è stata raggiunta scegliendo di utilizzare due approcci:

- **Listener locale:**

Grazie all'utilizzo di alcuni package per interfacciarsi ai sensori del dispositivo, è stato possibile implementare un listener che ad ogni cambiamento triggerà una chiamata api al server per aggiornare i dati;

- **Listener amici:**

Grazie all'utilizzo del protocollo websocket lato server e client, siamo riusciti a realizzare una serie di funzioni che, ad ogni input da parte dei propri amici, permettono di aggiornare i dati dell'applicativo e mostrare tale aggiornamento in tempo reale;

3.4.1 Scambio messaggi

La sezione Chat è stata implementata fornendo all'utente la possibilità di inviare contenuti testuali, immagini, file e audio ai propri amici attraverso conversazioni private.

Tutto ciò è stato realizzato grazie all'instaurazione di una comunicazione websocket client-server che permette a ogni utente di rimanere in ascolto per nuovi messaggi in arrivo.

La parte più complessa da progettare è stata quella dell'invio degli audio/immagini/file. Non essendo contenuti testuali è stato scelto di leggere il bytestream del file e successivamente inviarlo codificato in base64.

Viceversa, in ricezione di un messaggio, viene verificato il tipo di quest'ultimo e convertito nel formato corretto.

```

void sendMessage(dynamic raw, String type, String receiver, {String? filename}) {
    debugPrint("Sending $type to $receiver");
    String message = type == "text"
        ? raw
        : type == "file"
            ? base64Encode((raw as PlatformFile).bytes!) // file
            : base64Encode((raw as File).readAsBytesSync()); // images

    final data = {
        "sender": authUser.username,
        "message": message,
        "receiver": receiver,
        "type": type,
        if (filename != null) "filename": filename
    };
    _socketIO?.emit("send_message", json.encode(data));

    _storeNewMessage(data);
}

void _storeNewMessage(jsonData) async {
    Content content = jsonData["type"] == "text"
        ? TextContent(content: jsonData["message"])
        : await FileService.saveAndCreateFileMessage(jsonData);

    chatProvider.addMessage(
        content,
        jsonData["sender"],
        authUser.username == jsonData["receiver"]
            ? jsonData["sender"]
            : jsonData["receiver"],
    );
}

```

Figure 21: Send message

3.4.2 Posizione

Per la condivisione della posizione in tempo reale, è stato utilizzato il package *geolocator*, che fornisce una facile integrazione con i servizi di posizione del dispositivo e permette di ottenere le coordinate geografiche accurate in tempo reale.

Utilizzando geolocator, l'applicazione è in grado di aggiornare costantemente la posizione dell'utente e effettuare di conseguenza una chiamata al server. Per una questione di overload delle chiamate effettuate è stato scelto di implementare una distanza minima di 100 metri prima di comunicare l'aggiornamento della posizione ai propri amici.

```

void _startListener() {
    _positionListener = Geolocator.getPositionStream(locationSettings: locationSettings)
        .listen((Position position) {
            _position = position;
            debugPrint("Update Location ${position.latitude} ${position.longitude}");

            _updateLocation();
            notifyListeners();
        },
    );
    _positionListener?.onError((_) => _errorCallBack());
}

void _updateLocation() {
    HttpRequester.post(
        {},
        URL_UPDATE_LOCATION.format(_position.latitude, _position.longitude),
        token: _authUser?.token,
    );
}

```

Figure 22: Location listener

3.4.3 Contapassi

Per il monitoraggio dei passi effettuati dagli utenti, è stato utilizzato il package *pedometer*, che offre un’interfaccia semplice per accedere ai dati del contapassi del dispositivo.

Utilizzando il sensore contapassi l’applicazione è in grado di monitorare il numero di passi effettuati dall’utente durante la giornata e la settimana, consentendogli di tenere traccia del proprio livello di attività fisica e confrontarlo con quello dei suoi amici.

Durante l’aggiornamento dei passi se, rispetto all’ultimo aggiornamento, questo avviene durante la stessa giornata i passi vengono incrementati altrimenti il counter si resetta ripartendo da 0.

```
void onStepCount(StepCount event) {
    debugPrint('$_event');
    _steps ?.updateSteps(
        event,
        _steps ?.timeStamp != null
            ? isSameDayAccess(event.timeStamp, _steps!.timeStamp!)
            : false,
    );
    _updateSteps();
    notifyListeners();
    IsarService.instance.insertOrUpdate<Steps>(_steps!);
}

void _updateSteps() {
    HttpRequester.post(
        {},
        URL_UPDATE_STEPS.format(steps),
        token: _authUser ?.token,
    );
}
```

Figure 23: Steps listener