TicketWave

Gruppo "Marcosoft"

Programmazione Dispositivi Mobili
Anno 2023-2024

Partecipanti :
Filippo Nembrini
886135
Francesco Romeo
885880
Gabriele Soranno
885930
Mattia Trombella
879184
Marco Passoni
885873

Prof. di Riferimento :
Daniela Micucci

Indice

1 Introduzione

Benvenuti nell'applicazione di prenotazione eventi TicketWave. Un'innovativa piattaforma progettata per semplificare e arricchire la vostra esperienza nella pianificazione di eventi indimenticabili. Che si tratti di conferenze, concerti, partite o altro, la nostra app offre un modo intuitivo e efficiente per esplorare, prenotare e gestire una vasta gamma di eventi in modo rapido e conveniente. Con funzionalità avanzate e un'interfaccia utente intuitiva, siamo qui per trasformare la vostra esperienza di prenotazione, offrendo soluzioni su misura per soddisfare le vostre esigenze. Scoprite il futuro della prenotazione eventi con la nostra app, rendendo ogni momento un'occasione da celebrare.

Per visualizzare il codice o scaricare l'applicazione è possibile recarssi alla: pagina GitHub del progetto.

2 Sketch Iniziale

Nel nostro sketch iniziale, ci siamo concentrati sulla chiarezza, sull'accessibilità e sull'efficienza. Lavorando in gruppo ci siamo accorti come programmare per rendere migliore l'esperienza dell'utente con l'applicazione, portandoci così a modificare la grafica o rimuovere e implementare funzionalità diverse rispetto all'inizio.

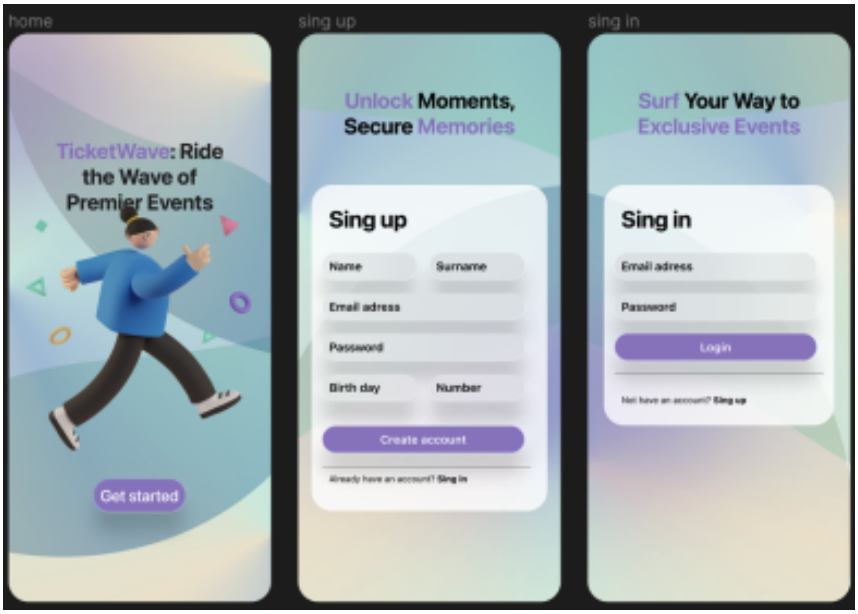

Figura 1: Home Page, Sign Up Page, Log In Page

Rispetto allo sketch iniziale siamo andati a inserire anche la pagina 'Forgot password' per poter permettere all'utente di ricevere una mail per resettare la password.

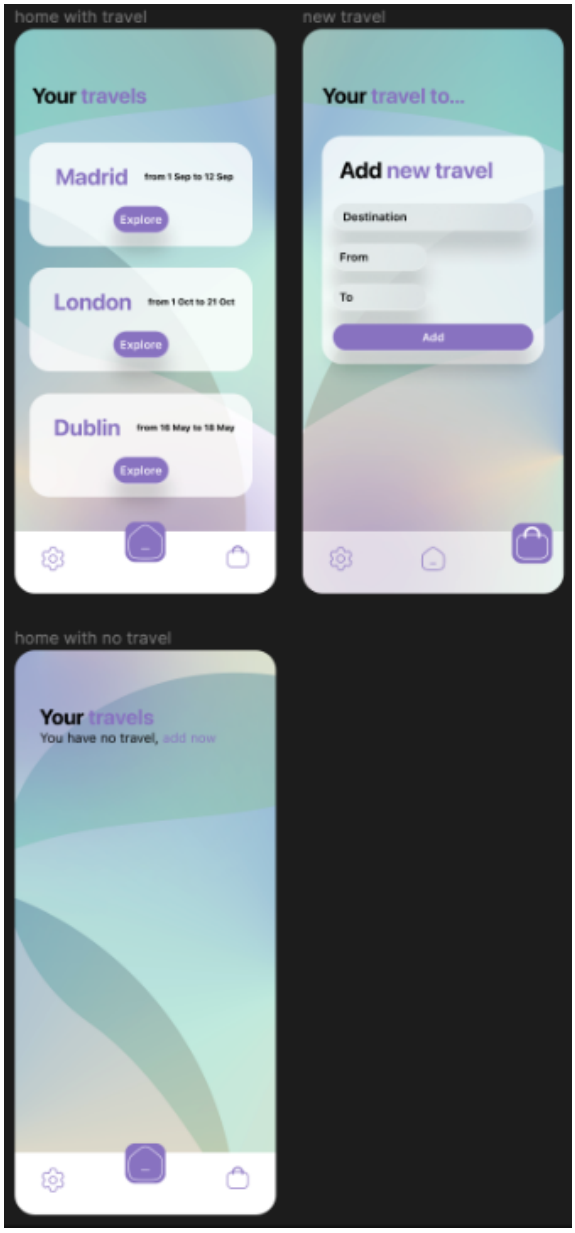

Figura 2: Home with no travel

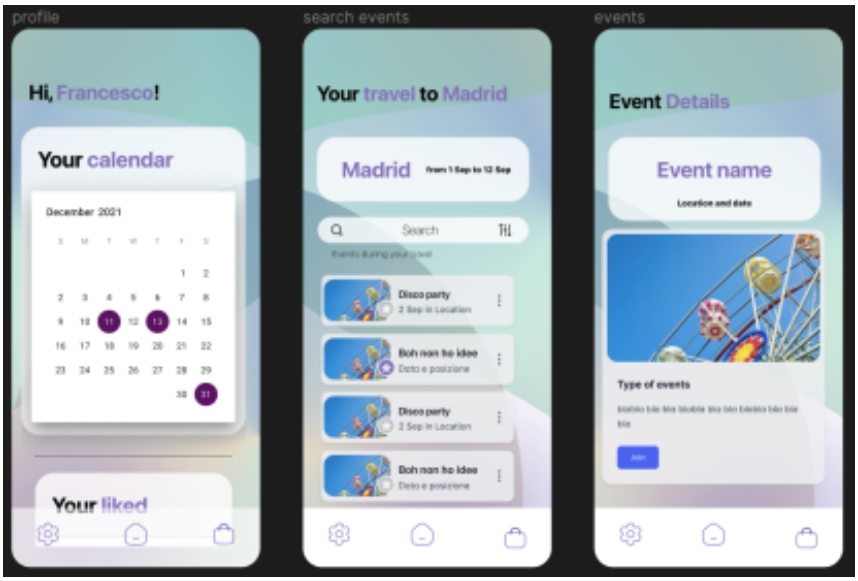

Figura 3: Profile, Search events, Events

3 Gestione del Progetto

I vari programmi usati per la gestione del progetto sono:

3.1 GitHub

Per la gestione delle attività abbiamo utilizzato GitHub Projects, che mette a disposizione una kanban board personalizzabile in cui inserire le attività. Queste attività possono essere convertite in issues in modo da poterle assegnare, mantenere uno stato chiuso/aperto, impostare un'etichetta (per esempio miglioramento, bug o documentazione) e associare una pull request.

3.2 Discord

Con cadenza settimanale il gruppo si riuniva su Discord, un'applicazione che permette di creare server privati con chat testuali e vocali. In queste occasioni ci si aggiornava sul lavoro svolto da ognuno e si decideva quali sarebbero state le attività da svolgere per il prossimo incontro. Con l'avvicinarsi della consegna le riunioni sono dieventate a cadenza giornaliera e durante le stesse si faceva pair programming in modo da procedere più velocemente.

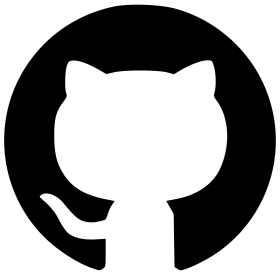

Figura 4: Github

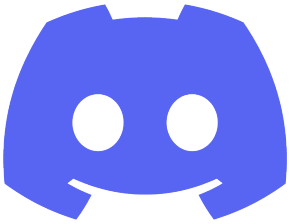

Figura 5: Discord

4 Diagramma dei casi d'uso

Abbiamo deciso di implementare i seguenti casi d'uso:

- Login: questo caso d'uso inizia quando l'utente inserisce le proprie credenziali. Il sistema verifica quindi queste credenziali tramite Firebase. Se le credenziali sono corrette, l'utente viene autenticato.
- Registrazione: se un utente non ha ancora un account, può registrarsi fornendo i dettagli richiesti. Il sistema registra quindi l'utente nel sistema tramite Firebase.
- Logout: se un utente decide di terminare la sessione effettua il logout
- Reset Password: se l'utente per qualsiasi motivo vuole modificare la sua password.
- Ricerca Eventi: una volta che l'utente è autenticato, può cercare eventi selezionando un luogo e un range di date. Il sistema esegue una chiamata API e mostra all'utente gli eventi corrispondenti.
- Visualizza Dettaglio Evento: l'utente visualizza a schermo i dettagli principali dell'evento.
- Aggiungi Evento Ai 'Liked': Se un utente trova un evento di suo interesse, può mettere "like" a quell'evento. Il sistema salva quindi l'evento nel database locale.
- Elimina Evento 'Liked': se un utente perdesse interesse nell'evento può toglierlo dai suoi 'liked'
- Visualizza Evento 'Liked': l'utente ha la possibilità di visualizzare i suoi eventi 'Liked'

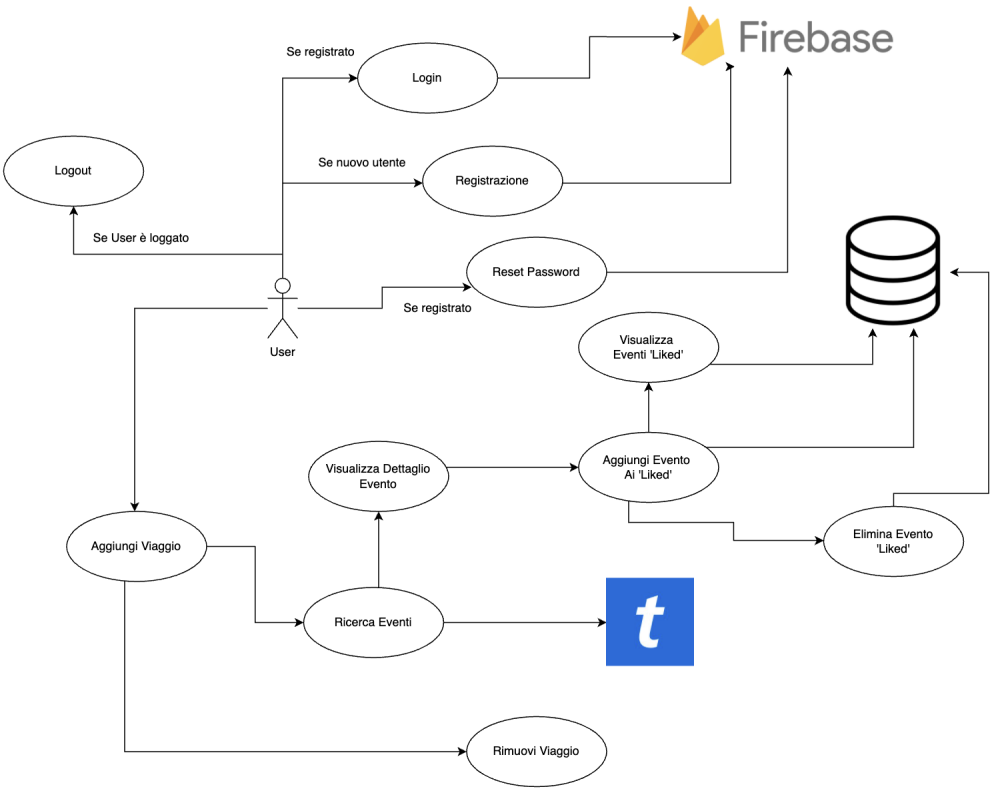

Figura 6: Diagramma dei casi d'uso

5 Architettura

TicketWave è un'applicazione costruita secondo uno standard di modellazione ben preciso: il Model - ViewModel - Repository, che si basa su tre strati principali:

- UI: gestiscono la UI dell'applicazione, popolando le View ed effettuando i vari passaggi tra una schermata e l'altra. L'ottenimento dei dati viene fatto tramite i LiveData contenuti nei ViewModel quando si tratta di dati statici, mentre per i dati che cambiano di volta in volta viene interpellato direttamente il repository.
- ViewModel: è uno strato di collegamento tra la UI e i repository. Il repository si occuperà di scrivere le informazioni sul LiveData contenuto nel View Model e la UI, dopo essere stata notificata della scrittura, leggerà dai LiveData i nuovi dati con cui popolare le View.
- Repository: contiene la logica di business e si interfaccia con API. Si occupano di recuperare i dati dal database e dalle API con cui lavorerà il primo strato.

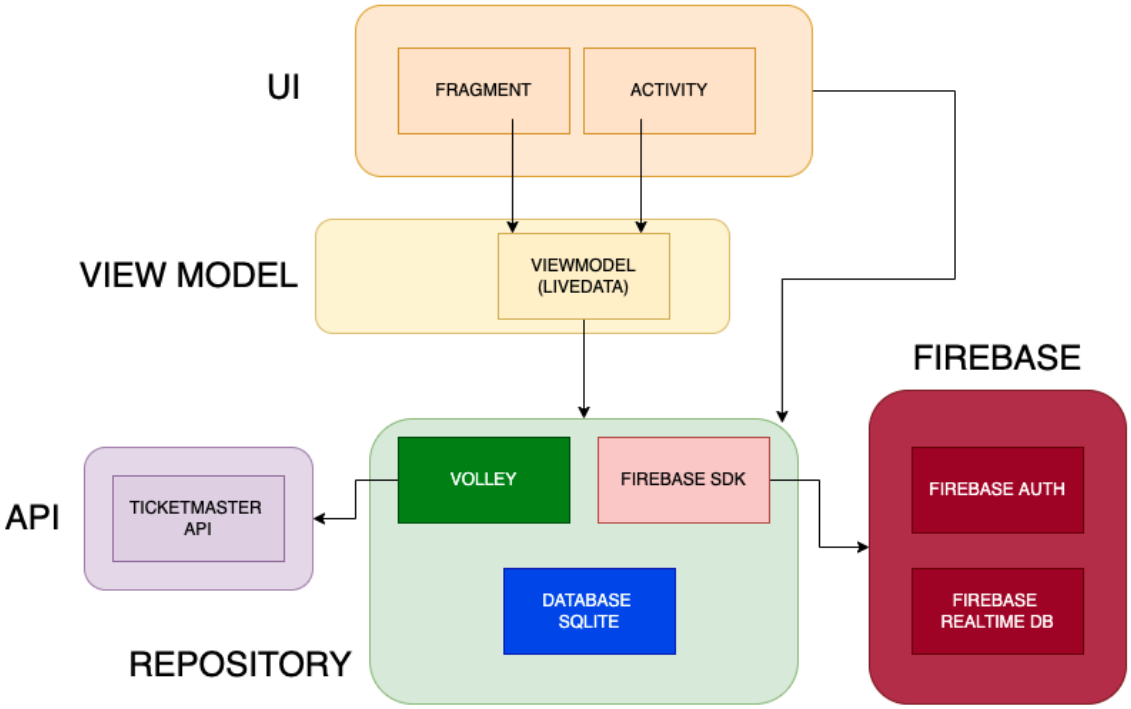

Figura 7: Architettura di TickeTwave

Vengono utilizzati come servizi esterni:

- TicketMaster API: fornisce tutte le informazioni necessarie sugli eventi
- Firebase Auth: utilizzato per l'autenticazione degli utenti
- Firebase RealTime Database: si occupa del salvataggio e la sincronizzazione di informazioni aggiuntive per l'account dell'utente.
- Database SQLite: Il database locale memorizza tutti gli altri dati e viene utilizzato anche per limitare le interazioni con l'API.

6 Funzionalità

Un'applicazione di prenotazione eventi dovrebbe offrire una serie di funzionalità per garantire un'esperienza utente completa e soddisfacente.

6.1 Registrazione

Pagina che permette all'utente di registarsi. L'utente sarà quindi guidato alla registrazione inserendo i propri dati tra i quali: Email, Password, Nome, Cognome, Età.

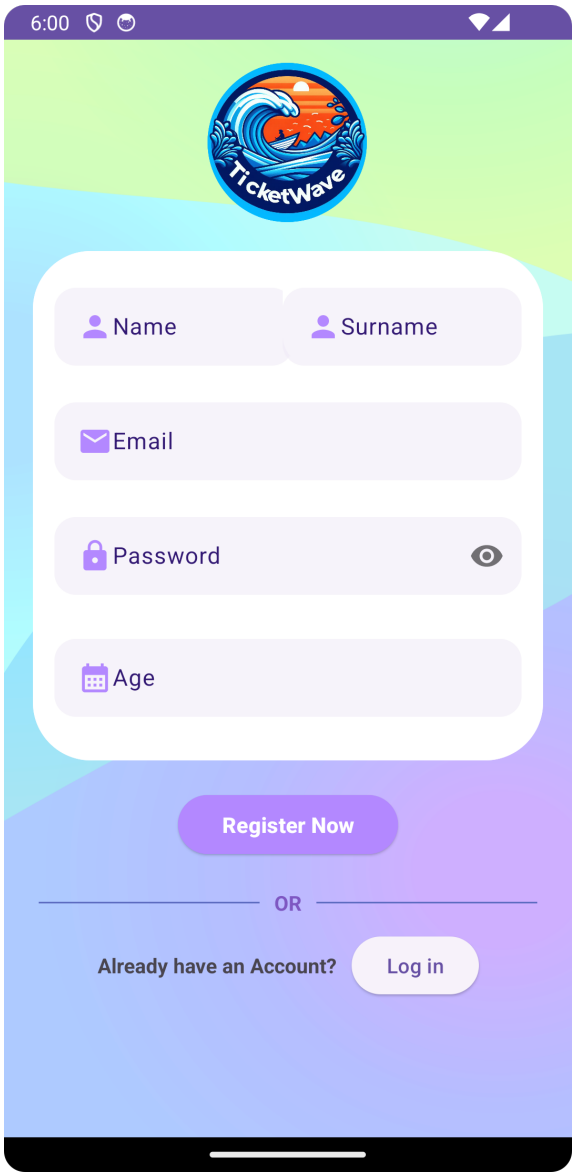

Figura 8: Registration page

6.2 Login

Una volta che l'utente si è registrato può fare l'autenticazione dati, che in caso corretta, lo reindirizza alla home page.

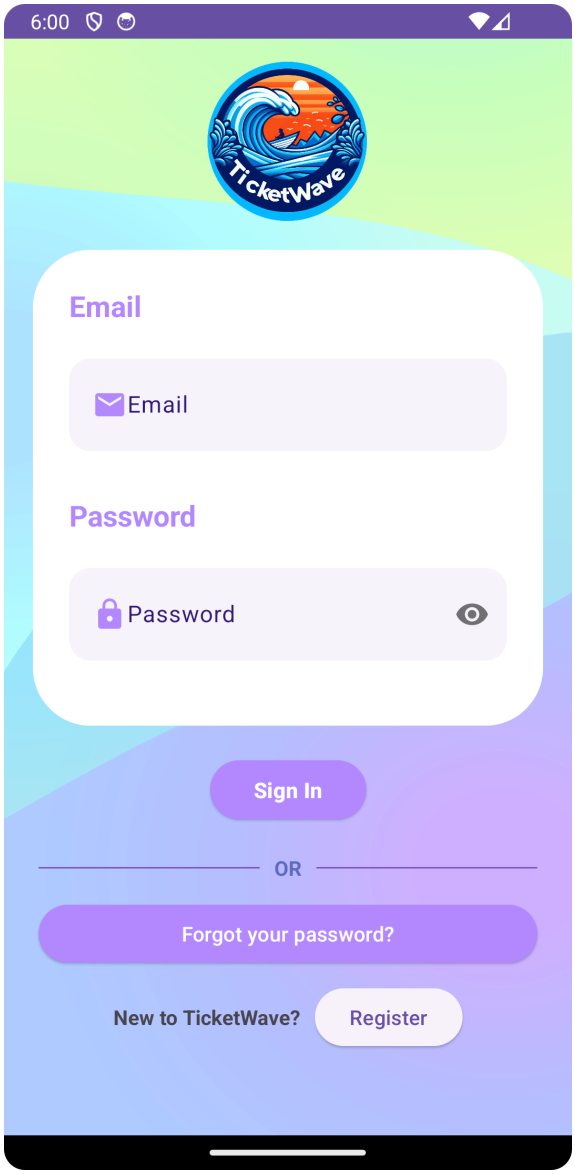

Figura 9: Login Page

6.3 Recupero password

Questa è una funzionalità molto importante per l'utente. Permette ad esso di modificare la propria password inserendo dell'apposito campo la propria mail usata nel momento della registrazione. Una volta chiesto il reset della password, verrà inviata una mail da parte dell'app con un link per il refactor della password.

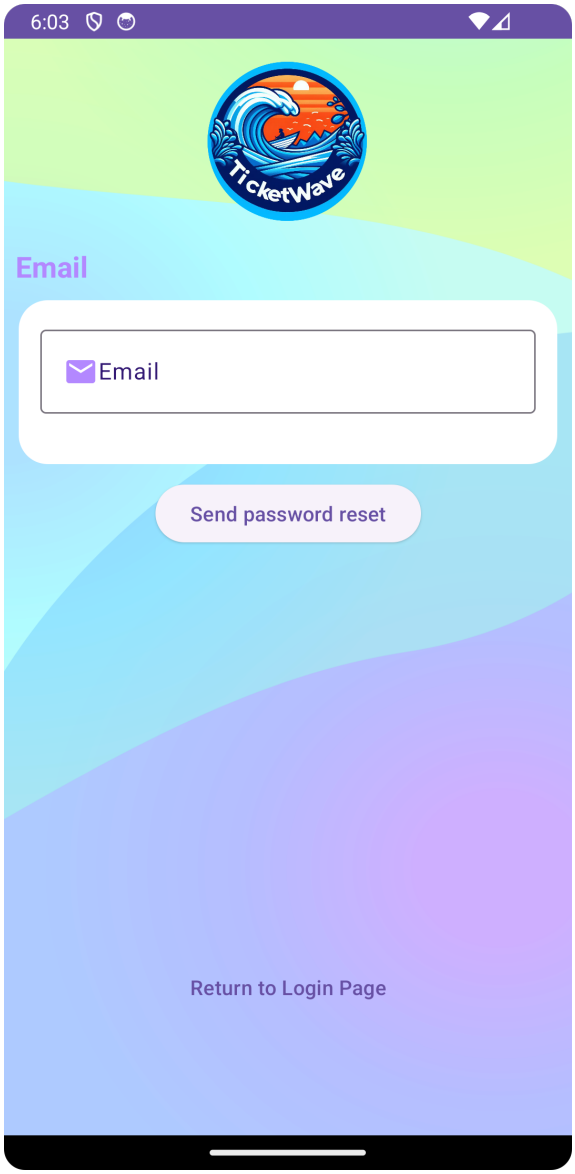

Figura 10: Reset Password

6.4 Riepilogo dati

Implementazione che consente all'utente di rivedere i propri dati e, nel caso, decidere di fare il logout.

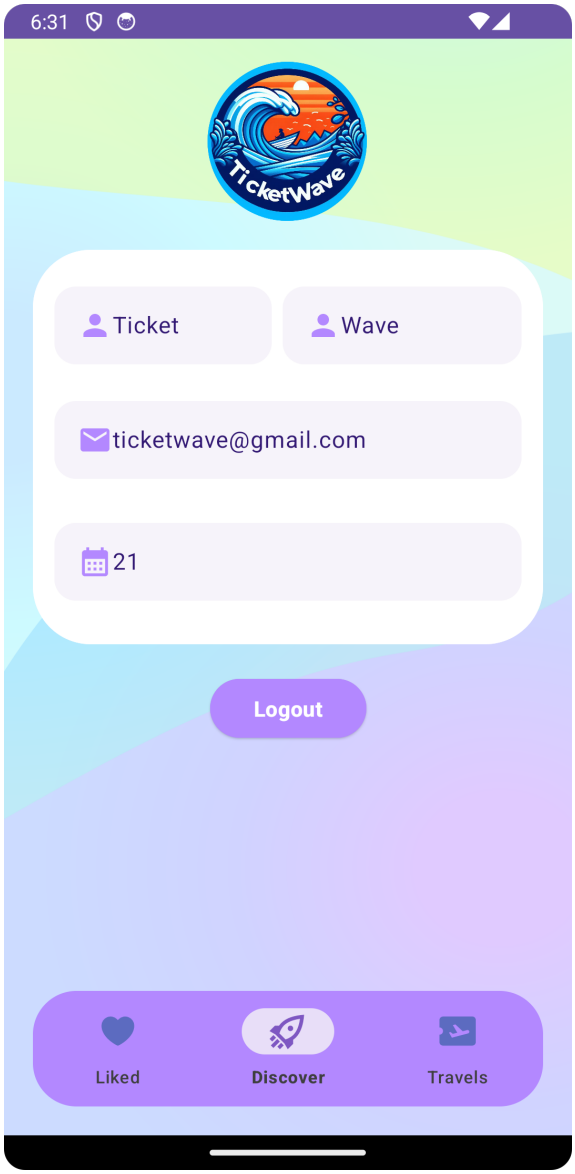

Figura 11: Riepilogo dati

6.5 Aggiungere un Travel

Funzione che permette all'utente di inserire un luogo con un range di date. Verrà aggiunto al database locale.

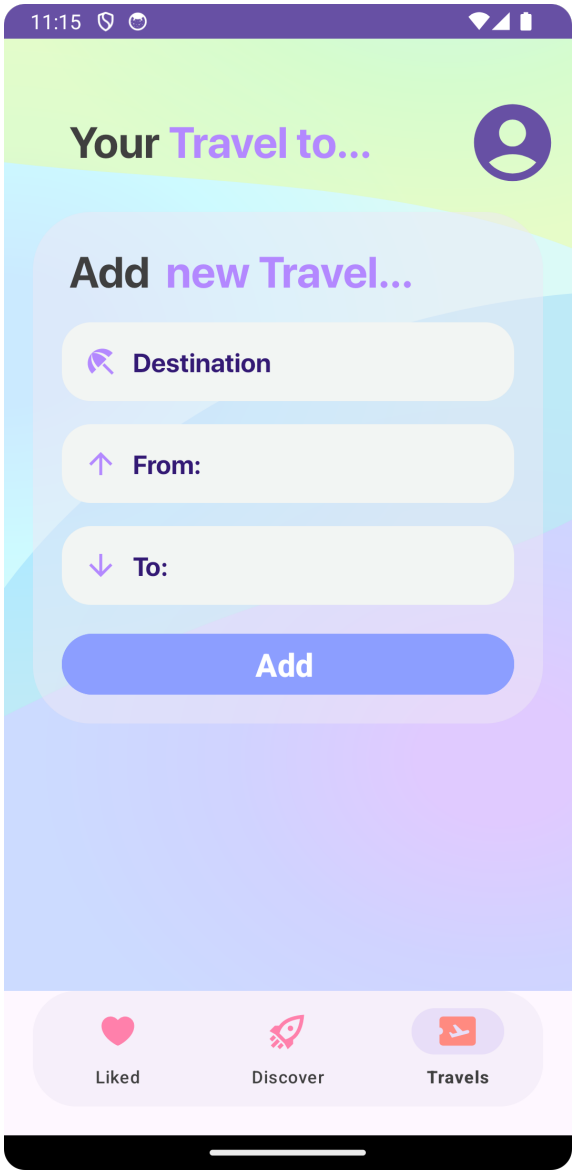

Figura 12: Add New Travel

6.6 Visualizzazione dei travel

Nel fragment l'utente potrà vedere quali viaggi ha intenzione di fare e cliccando su explore d vedere i relativi eventi associati a quel luogo e quella data, cliccando invece su delete di cancellare quel travel.

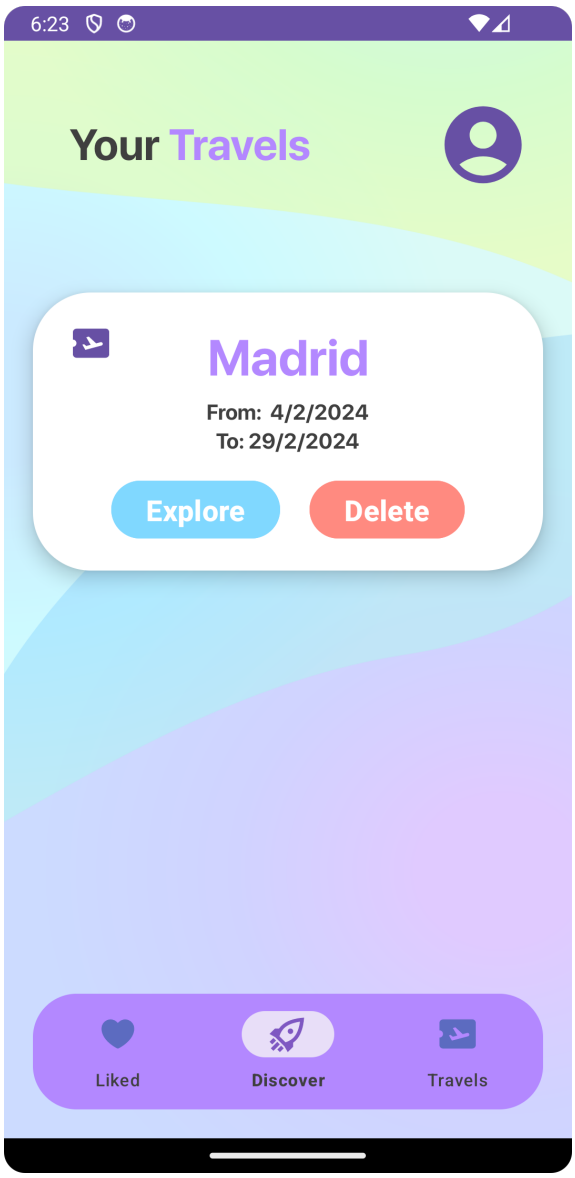

Figura 13: Visualizzazione Travel

6.7 Visualizzazione di eventi

Questa funzionalità che abbiamo introdotto nella nostra applicazione permette all'utente di visualizzare gli eventi di qualsiasi tipo nel luogo in cui l'utente desidera. Viene fatta una chiamata API che ritorna le informazioni più importanti per l'utente.

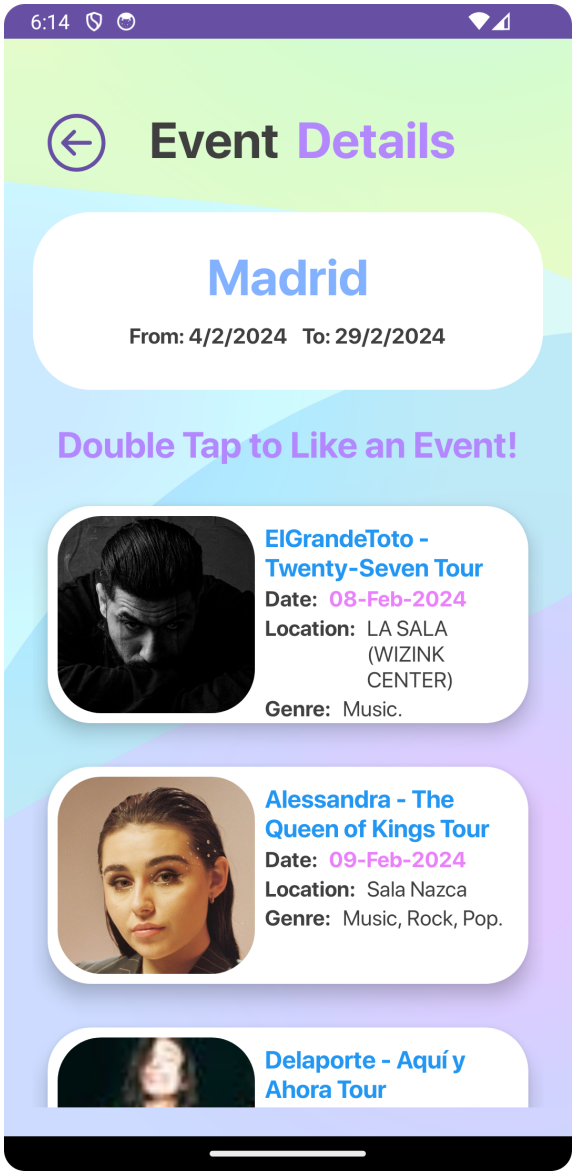

Figura 14: Event Details

6.8 Mettere Like ad eventi e salvarli

Funzionalità che è stata implementata per agevolare l'utente durante l'utilizzo dell'applicazione. Una volta che l'utente ha trovato un evento al quale è particolarmente interessato può decidere di aggiungerlo ai preferiti, ritrovandolo così nella pagina dei liked, nella quale è presente anche un calendario per ricordarsi quando sarà. Nel caso l'utente cambi idea e voglia togliere dall'elenco uno o più eventi, basta che clicchi due volte sopra ad esso e sparirà dalla sua lista!

Figura 15: Liked Page

7 API

Come API abbiamo utilizzato TicketMaster (https://developer.ticketmaster.com/products-and-docs/apis/getting-started/), che permette di ottenere i vari dettagli riguardo agli eventi. (Nome, Data, Descrizione, Luogo).

7.1 Utilizzo API

Nell'app abbiamo integrato l'API in diverse sezioni per fornire un'esperienza utente completa e personalizzata.

Quando l'utente aggiunge una destinazione specifica e definisce un intervallo di date desiderato, cliccando sulla Card relativa al viaggio (Fig. 13), l'app effettua una chiamata all'API utilizzando l'endpoint *"discovery/v2/events"*. Questa chiamata restituisce tutti i dati relativi agli eventi disponibili durante quei giorni nella destinazione scelta. Gli utenti possono così visualizzare e scegliere gli eventi in base alle loro preferenze.

Entrambe queste funzionalità sono state implementate utilizzando principalmente l'endpoint *"discovery/v2/events"*, che fornisce un ampio accesso a tutte le informazioni relative agli eventi.

L'oggetto principale utilizzato per gestire le informazioni sugli eventi è stato un'apposito oggetto "Event", progettato con sottoclassi per rappresentare dettagliatamente tutte le informazioni necessarie. Questo approccio ci ha permesso di garantire una gestione completa ed efficiente degli eventi nell'applicazione.

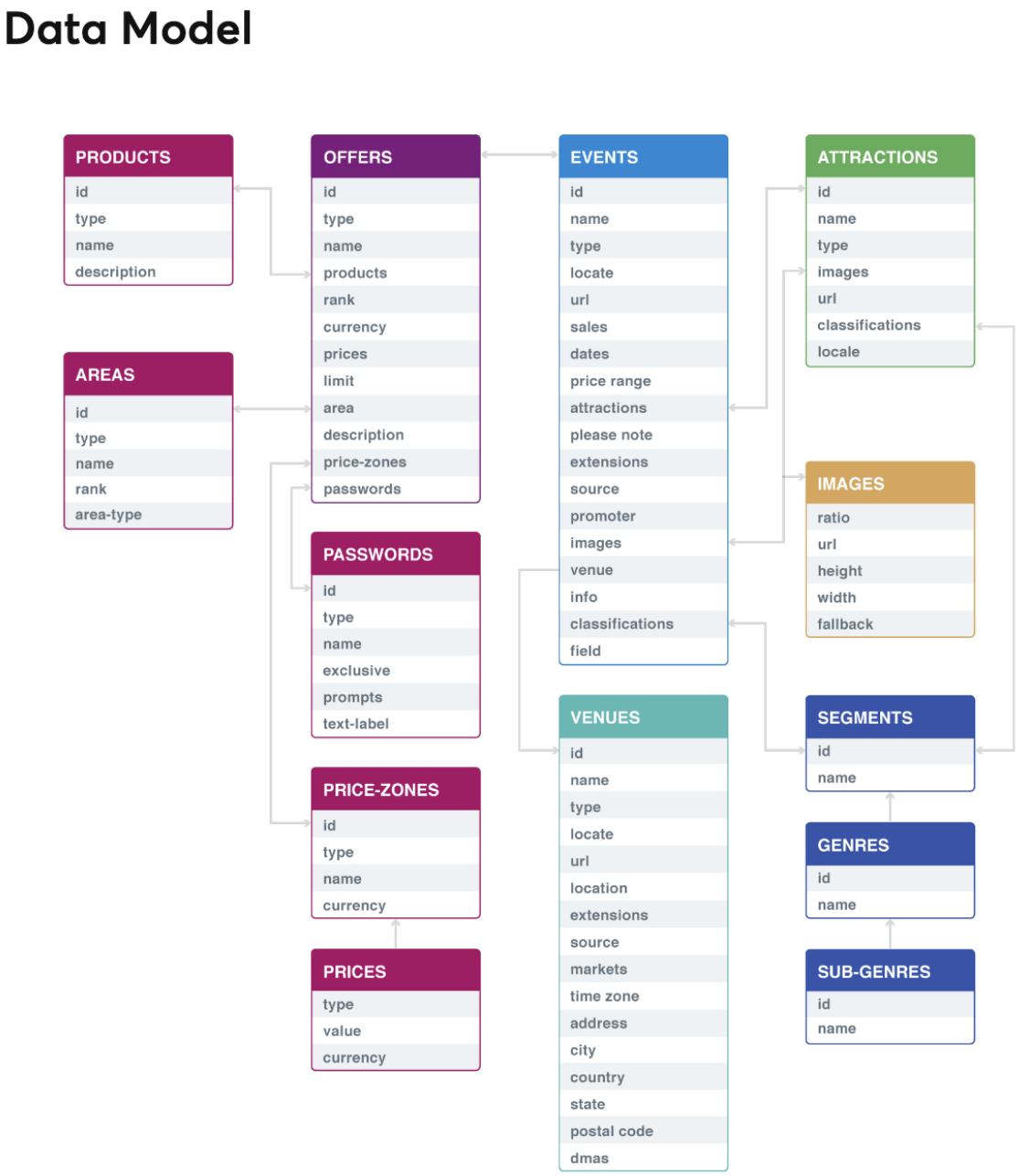

Figura 16: Architettura TicketMasters API

8 Database

L'autenticazione degli utenti e il salvataggio dei loro dati personali e preferenze è gestita da Firebase, utilizzando i servizi di authentication e realtime. Inoltre abbiamo deciso di implementare un database locale.

8.1 Firebase Authentication

Necessario per gestire il login e la registrazione dell'utente. Attualmente il login è disponibile solo tramite Email e Password

8.2 Firebase Realtime Database

Necessario per il salvataggio dei dati personali associati

8.3 Database SQLite

Il database locale memorizza gli eventi preferiti che l'utente ha salvato e le destinazioni per le quali è interessato a cercare eventi.

9 Design

Lo stile di TicketWave è stato pensato per essere gradevole all'utente. La scelta dei colori da parte del team è stata una delle cose più importanti sulle quali ci siamo dedicati. La scelta è ricaduta sui colori pastello, colori gradevoli per l'utente e per migliorarne l'esperienza. Scelta che ha portato notevoli differenze rispetto allo sketch iniziale. I colori dominanti sono i seguenti:

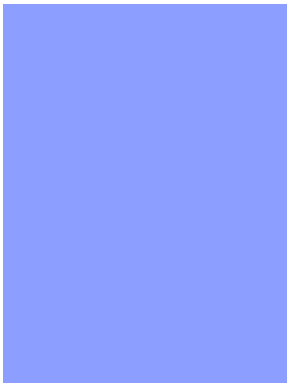

Figura 17: Ciano Scuro: 8C9EFF

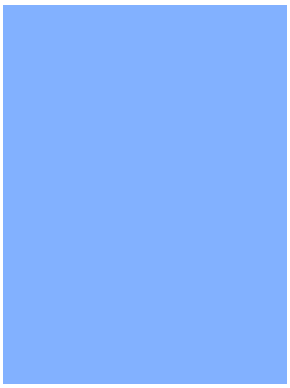

Figura 18: Ciano: 82B1FF

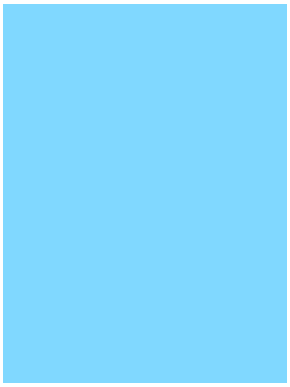

Figura 19: Turchese: 80D8FF

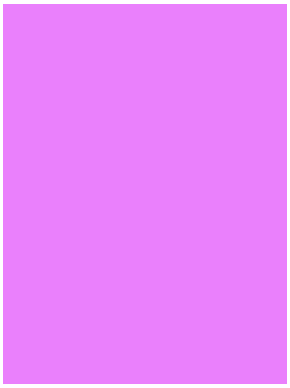

Figura 20: Rosa scuro: EA80FC

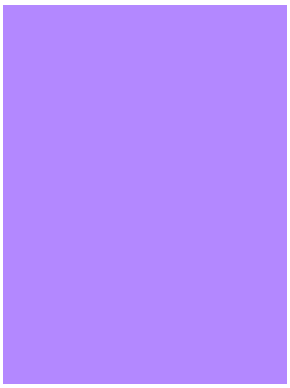

Figura 21: Viola: B388FF

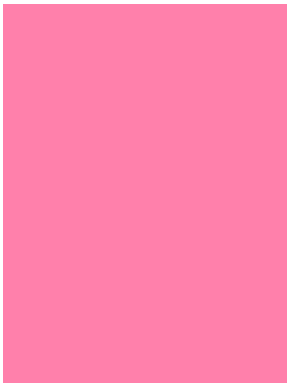

Figura 22: Rosso chiaro: FF80AB

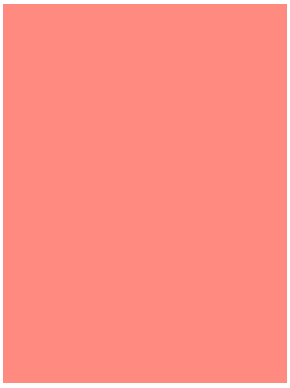

Figura 23: Arancione: FF8A80

10 Considerazioni e Sviluppi Futuri

Ecco alcune considerazione e idee per il futuro:

10.1 Notifiche eventi preferiti

Inviare all'utente delle notifiche quando stanno per verificarsi gli eventi che ha aggiunto ai preferiti.

10.2 Mostrare altre informazioni

Le API che abbiamo scelto mettono a disposizione una quantità incredibile di informazioni, e nella nostra applicazione le abbiamo utilizzate solo in parte. Per il futuro sarebbe pensabile mostrare anche quelle informazioni che, per motivi di tempo e per concentrarci su quelle che secondo noi erano le più importanti per rendere l'applicazione la più completa possibile, abbiamo dovuto tralasciare.

Un esempio potrebbe essere la geolocalizzazione dell'evento, disponendo di Latitudine e Logitudine dello stesso.

10.3 Condivisione su social e chat

Condivisione su social e chat Aggiungere un tasto per condividere un determianto evento al quali si vuole partecipare attraverso i social o le app di messaggistica. Per la condivisione verrebbe utilizzato un URL che, una volta aperto, se l'applicazione è installata sul dispositivo porta nella pagina in cui è presente l'informazione condivisa, altrimenti porta alla pagina dello store in cui è possibile scaricare l'applicazione.

10.4 Pubblicazione sul Play Store

Dopo aver rifinito alcuni dettagli e magari realizzato alcuni degli altri sviluppi futuri, ci piacerebbe rilasciare l'applicazione sul Play Store. Fino a quel momento, sarà possibile installarla tramite APK o compilarla autonomamente a partire dal codice sorgente, disponibile pubblicamente su GitHub.

10.5 Utilizzo di Crashlytics

Crashlytics è uno strumento messo a disposizione da Firebase che serve a raccogliere informazioni sui crash, log di esecuzione ed altre informazioni dai dispositivi degli utenti. In previsione di un eventuale rilascio sul Play Store, questo servizio sarebbe utile per migliorare l'esperienza degli utenti.