MyBooks
Documentazione

Introduzione 2
Design 3
Progettazione UI 3
Material Design 5
Funzionalità 6
Autenticazione (Welcome/Login/Register) 6
Library 9
Search 11
Detail 13
Profile 15
Architettura e Scelte progettuali 16
MVVM - ViewModel e Fragment 16
Single Activity 17
Firebase e Firestore 17
Room 17
Retrofit 18
Tecnologie 19
Sviluppi futuri 20
Architettura 20
Funzionalità 20

Introduzione

Il Progetto MyBooks consiste nello sviluppo di un'applicazione mobile Android dedicata alla gestione di una libreria personale digitale. L'obiettivo principale dell'app è fornire agli utenti uno strumento semplice e immediato per cercare libri da un catalogo online e salvarli, creando così una raccolta personale facilmente consultabile.

L'applicazione è pensata per lettori abituali che desiderano tenere traccia dei libri di interesse o da leggere. L'interfaccia è suddivisa in tre sezioni principali, una dedicata alla visualizzazione e alla gestione della libreria personale, una dedicata alla ricerca dei libri, e una dedicata alla gestione del profilo.

Design

Progettazione UI

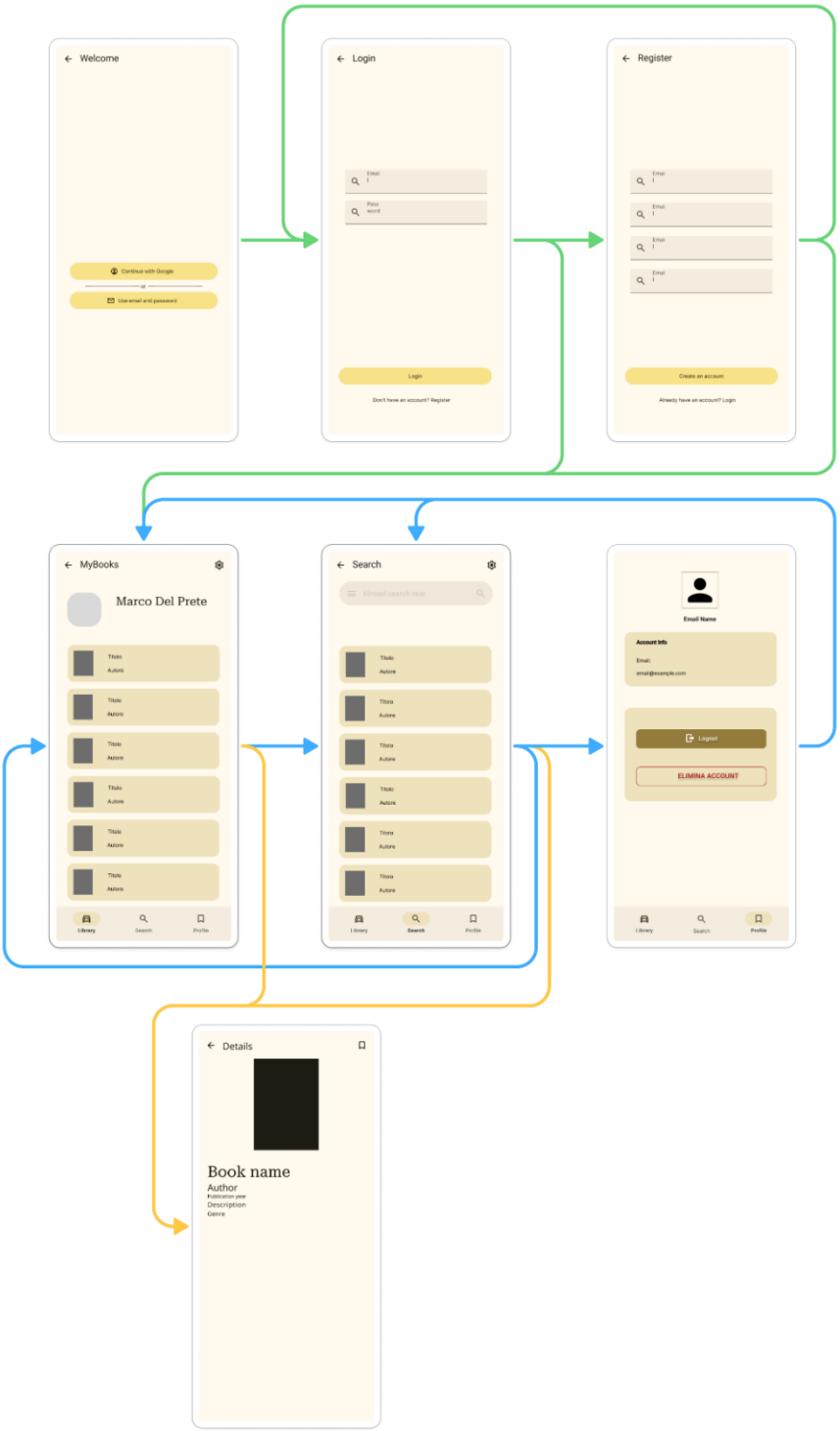
Per la progettazione dell'interfaccia grafica di MyBooks è stato seguito un processo strutturato di prototipazione, con l'obiettivo di garantire coerenza visiva, semplicità d'uso e una navigazione intuitiva. Lo strumento utilizzato per la fase di design è stato Figma.

Abbiamo inizialmente definito in modo chiaro cosa dovesse fare l'applicazione e quali schermate fossero necessarie. L'obiettivo era mantenere l'app semplice, quindi ci siamo focalizzati solo sulle funzionalità essenziali:

- Autenticazione (login e registrazione)
- Libreria personale
- Ricerca libri
- Dettaglio libro
- Profilo

Prima di disegnare le nostre schermate, abbiamo analizzato applicazioni simili con l'obiettivo di prendere spunto da soluzioni già validate dal punto di vista dell'usabilità.

Il lavoro su Figma non è stato approfondito a livello grafico, ma si è trattato di una progettazione mirata a definire il layout degli elementi, in modo da stabilire la gerarchia delle informazioni e verificare la coerenza tra le schermante, simulando il flusso di navigazione.


Welcome
Continue with Google
Use email and password
Login
Login
Don't have an account? Register
Register
Create an account
Already have an account? Login
MyBooks
Marco Del Prete
Search
Email Name
Account Info
Email:
Logout
ELIMINA ACCOUNT
Details
Book name
Author
Publication year
Description
Genre

Material Design

La progettazione dell'interfaccia è basata sui dettami stilistici di Material Design 3, utilizzando la libreria ufficiale (com.google.android.material) e personalizzando gli elementi dove necessario. La scelta è stata coerente con lo sviluppo Android moderno e ha permesso di costruire l'intera UI su componenti standard, limitando l'uso di soluzioni grafiche personalizzate.

La struttura dell'app prevede una *MaterialToolbar* nella parte superiore, un contenitore centrale per la gestione dinamica dei *Fragment* tramite *Navigation Component* e una *BottomNavigationView* per la navigazione tra le sezioni principali. Questo schema garantisce coerenza tra le schermate e una separazione chiara delle funzionalità.

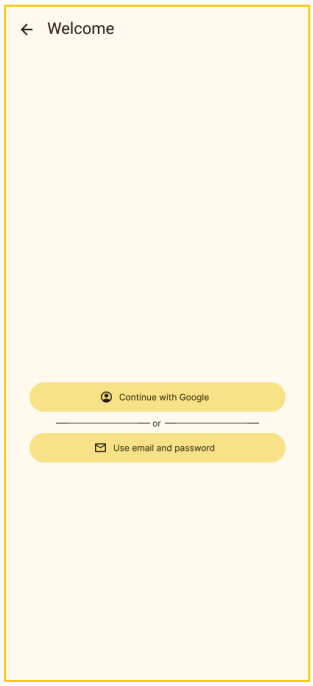
La visualizzazione dei contenuti (come libri e informazioni utente) è basata su *MaterialCardView*, mentre input e form utilizzano *TextInputLayout* e *TextInputEditText*, sfruttando il comportamento standard per gestione di hint, errori e icone. Le azioni principali sono evidenziate tramite pulsanti Material e, nel dettaglio libro, tramite una *ExtendedFloatingActionButton*.

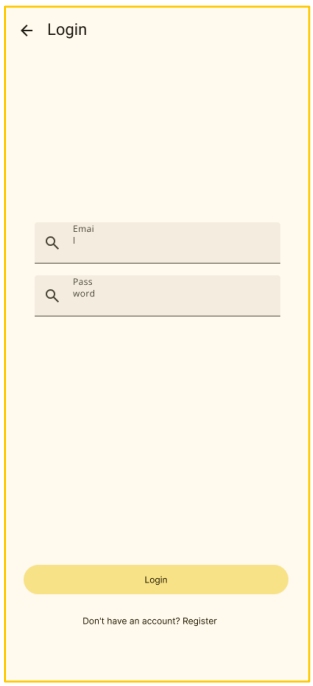
Dal punto di vista grafico, l'interfaccia utilizza esclusivamente attributi del tema (*?attr/colorPrimary*, *colorSurfaceContainer*, *colorOnSurface*, ecc.), evitando colori hardcoded. Questo rende l'app compatibile con il sistema di theming Android, ottenendo coerenza cromatica nei vari elementi dell'interfaccia.

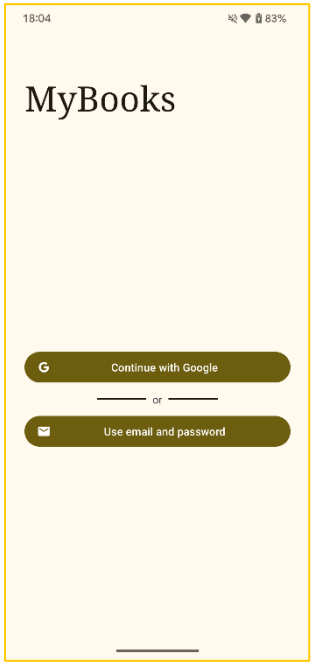
In generale, Material Design è stato utilizzato come base strutturale dell'interfaccia, garantendo uniformità visiva, comportamento prevedibile dei componenti e maggiore semplicità di manutenzione del codice.

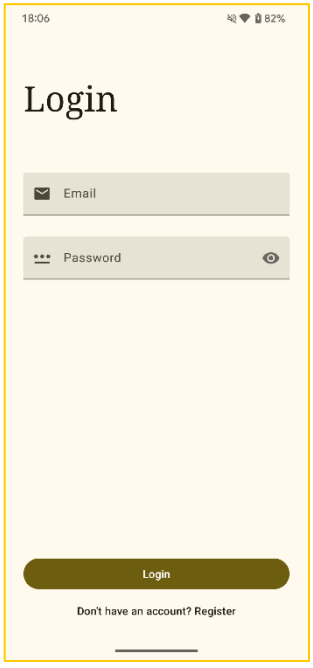
Funzionalità

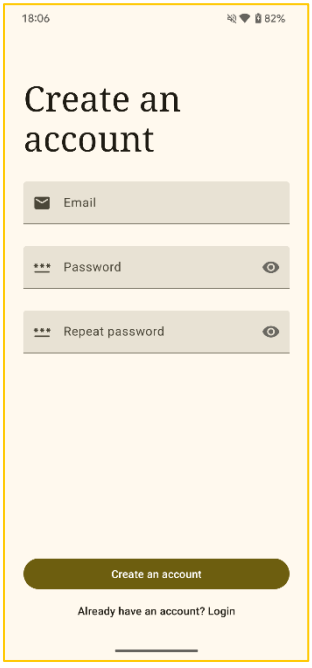
Autenticazione (Welcome/Login/Register)











Al primo avvio dell'applicazione viene mostrata la schermata di Welcome, che rappresenta il punto di ingresso al flusso di autenticazione. L'interfaccia è semplice e immediata e in questa fase l'utente può scegliere la modalità di accesso all'applicazione tramite due pulsanti:

- Continue with Google.
- Use e-mail and password.

Se l'utente seleziona l'accesso con Google, viene avviato il flusso di autenticazione tramite Google Sign-in. Una volta completata correttamente l'autenticazione, l'utente accede direttamente alla schermata principale dell'applicazione.

Se invece viene scelta l'autenticazione con e-mail e password, l'utente viene reindirizzato alla schermata di Login.

La schermata di Login consente all'utente di inserire le proprie credenziali per accedere all'applicazione. L'interfaccia è composta da due campi di input (e-mail e password) e da un pulsante per effettuare l'accesso, oltre a un collegamento alla schermata di registrazione per i nuovi utenti.

Prima di inviare la richiesta di autenticazione, viene effettuata una validazione locale dei dati inseriti:

- L'email deve essere non vuota e conforme al formato standard.
- La password deve essere presente.

Eventuali errori vengono segnalati direttamente nei campi di input, permettendo all'utente di correggere immediatamente le informazioni.

Quando invece le credenziali risultano valide, viene avviata la richiesta di autenticazione. Durante questa fase i pulsanti vengono temporaneamente disabilitati e il testo del bottone viene aggiornato per indicare lo stato di caricamento, evitando invii multipli della stessa richiesta.

Se l'autenticazione va a buon fine, l'utente viene automaticamente reindirizzato alla schermata principale dell'applicazione.

La schermata di registrazione permette la creazione di un nuovo account tramite e-mail e password. L'interfaccia prevede tre campi: e-mail, password e conferma password.

Oltre alla validazione dell'e-mail, la password deve rispettare specifici criteri di sicurezza:

- lunghezza minima di 6 caratteri;
- presenza di almeno una lettera maiuscola;
- presenza di almeno una lettera minuscola;
- presenza di almeno un numero.

È inoltre richiesto che il campo di conferma password coincida con la password inserita. I controlli vengono eseguiti in tempo reale, fornendo all'utente un riscontro immediato in caso di errore.

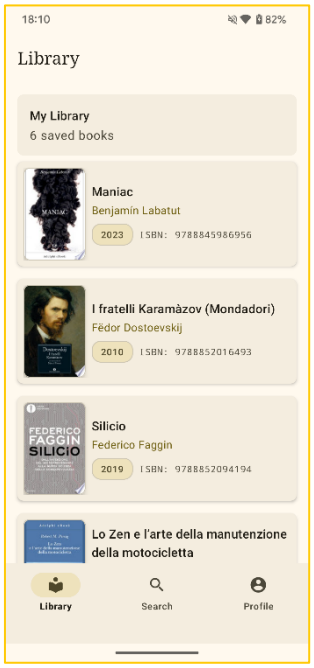
Solo quando tutte le verifiche sono soddisfatte viene avviata la procedura di creazione dell'account. Anche in questo caso, durante l'elaborazione i pulsanti vengono disabilitati e viene indicato lo stato di caricamento.

Se la registrazione ha esito positivo, l'utente accede direttamente all'applicazione. In caso contrario, viene mostrato un messaggio descrittivo dell'errore.

Una volta autenticato, l'utente rimane connesso anche dopo la chiusura dell'applicazione. Alla riapertura, se la sessione è ancora valida, l'accesso avviene automaticamente senza richiedere nuovamente le credenziali.

La navigazione tra schermate di autenticazione e schermata principale è gestita in modo centralizzato, garantendo che, dopo l'accesso, non sia possibile tornare accidentalmente alle schermate di login o registrazione tramite il tasto "indietro".

Library



Dopo l'autenticazione, l'utente accede alla schermata Library, che rappresenta la sezione principale dell'applicazione e raccoglie tutti i libri salvati nel proprio account. La parte superiore della schermata mostra una card riepilogativa con il titolo della sezione e il numero totale di libri presenti nella libreria personale; questo contatore viene aggiornato automaticamente ogni volta che l'elenco cambia, fornendo un riscontro immediato sullo stato della raccolta.

La parte centrale della schermata è occupata da una lista a scorrimento verticale che visualizza i libri salvati, ordinati in base alla data di salvataggio. Ogni elemento è rappresentato da una card che include le informazioni essenziali del libro:

- copertina,
- titolo,
- autore,
- anno di pubblicazione
- ISBN.

Le immagini delle copertine vengono caricate dinamicamente e memorizzate in cache per garantire fluidità nella navigazione e ridurre il consumo di rete.

Nel caso in cui l'utente non abbia ancora salvato alcun libro, la lista viene nascosta e viene mostrato uno stato vuoto con un messaggio informativo e un suggerimento che invita a effettuare una ricerca per aggiungere contenuti alla libreria. Questo evita schermate prive di contesto e orienta chiaramente l'utente verso l'azione successiva.

La libreria si aggiorna automaticamente quando un libro viene aggiunto o rimosso. Le modifiche vengono riflesse immediatamente nell'interfaccia senza necessità di ricaricare manualmente la schermata. All'apertura della sezione, i libri salvati vengono sincronizzati con l'account dell'utente e memorizzati localmente, garantendo tempi di caricamento rapidi e persistenza dei dati anche dopo la chiusura dell'applicazione.

Ogni elemento della lista è selezionabile: toccando una card si accede alla schermata di dettaglio del libro, mantenendo la coerenza del flusso di navigazione. Tornando indietro, l'utente ritrova la libreria nello stesso stato precedente.

Search

La sezione Search consente all'utente di cercare nuovi libri tramite integrazione con Google Books API. L'interfaccia è costruita attorno a una barra di ricerca posizionata nella parte superiore della schermata, accompagnata da un'icona iniziale di ricerca e da un'icona di cancellazione che permette di svuotare rapidamente il campo di testo. Centralmente viene mostrato un messaggio che fornisce indicazioni per effettuare una ricerca. Questo evita schermate prive di contesto e orienta chiaramente l'utente verso l'azione successiva.

La ricerca viene avviata esplicitamente premendo l'azione "Search" sulla tastiera. Questo comportamento evita di inviare un numero elevato di richieste API al servizio. Se la query non è vuota, viene inviata la richiesta remota e la tastiera viene chiusa per lasciare spazio ai risultati. Durante l'esecuzione della chiamata viene mostrato un indicatore di caricamento centrale, che segnala chiaramente che la ricerca è in corso.

Al termine dell'operazione, la lista dei risultati viene aggiornata automaticamente in base ai dati restituiti dalla chiamata API, grazie all'osservazione dei LiveData esposti dal ViewModel.

Se la ricerca produce risultati, questi vengono visualizzati in una lista a scorrimento verticale, riutilizzando lo stesso layout grafico della libreria. Ogni elemento mostra copertina, titolo, autore, anno e ISBN, garantendo coerenza visiva tra le diverse sezioni dell'applicazione. Le immagini vengono caricate dinamicamente e memorizzate in cache per assicurare fluidità nello scorrimento.

Se invece la ricerca non restituisce alcun risultato, viene mostrato un messaggio informativo al centro della schermata. Questo evita liste vuote prive di contesto e comunica chiaramente all'utente che la query non ha prodotto corrispondenze.

Ogni risultato è selezionabile: toccando un libro si accede alla schermata di dettaglio, dove l'utente può visualizzare informazioni più approfondite e decidere eventualmente di salvarlo nella propria libreria.

La barra di ricerca viene inoltre abilitata o disabilitata in base allo stato della connessione di rete, prevenendo tentativi di ricerca quando il dispositivo non è connesso. In questo modo l'esperienza rimane coerente e controllata anche in condizioni di rete non ottimali.

La sezione Search è progettata per offrire un flusso semplice e lineare che prevede una presentazione ordinata dei risultati.

Detail

La schermata di dettaglio rappresenta il punto di approfondimento di un libro selezionato dalla sezione Library o dalla sezione Search. Il libro viene passato tramite Navigation Component e mostrato immediatamente senza ulteriori chiamate remote, garantendo un'apertura rapida e fluida della schermata.

La parte superiore della vista è dedicata alla presentazione visiva del libro: copertina in evidenza, titolo, eventuale sottotitolo e autore. L'impaginazione è centrata e pulita, con particolare attenzione alla gerarchia tipografica per mettere in risalto le informazioni principali. Se disponibili, vengono mostrati anche valutazione media e numero di recensioni; in assenza di questi dati, la sezione viene automaticamente nascosta per evitare spazi vuoti o informazioni parziali.

Sotto la parte introduttiva è presente una sezione "Book details" che raccoglie i metadati strutturati: ISBN, numero di pagine, lingua e categorie. Anche in questo caso ogni blocco viene reso visibile solo se il dato è effettivamente disponibile, mantenendo l'interfaccia compatta e priva di elementi inutili. La descrizione del libro è inserita in una card separata, mostrata esclusivamente quando il contenuto è presente, così da non appesantire la schermata nei casi in cui l'API non fornisca testo descrittivo.

L'intera schermata è contenuta in uno scroll verticale, permettendo di consultare comodamente anche descrizioni lunghe.

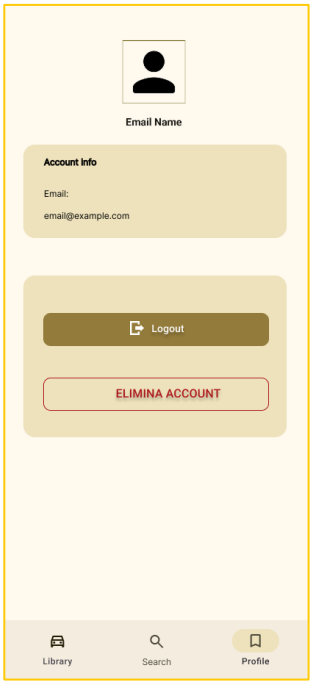
Dal punto di vista funzionale, l'elemento centrale dell'interazione è l'Extended Floating Action Button posizionato in basso a destra. Questo pulsante consente di aggiungere o rimuovere il libro dalla libreria personale. Lo stato del pulsante viene determinato verificando se il libro è già presente nella propria libreria:

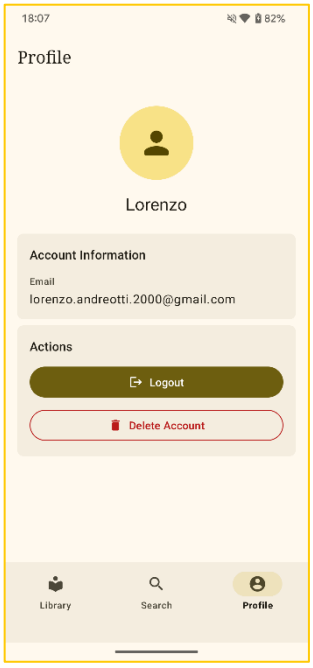
- se il libro non è salvato, il pulsante mostra l'azione "Save" con icona di aggiunta;
- se il libro è già presente, mostra l'azione "Remove" con icona di rimozione.

Il cambio di stato è immediato e accompagnato da un breve messaggio di conferma. L'operazione aggiorna automaticamente la libreria grazie all'osservazione dei dati persistiti, senza necessità di ricaricare manualmente le schermate.

Il pulsante viene inoltre abilitato solo in presenza di connessione, evitando operazioni incoerenti con lo stato dell'applicazione.

Profile





La sezione Profile rappresenta lo spazio personale dell'utente autenticato e consente di visualizzare le informazioni dell'account e gestirne le impostazioni essenziali.

Nella parte superiore è presente un'intestazione con un'icona profilo e il nome dell'utente. Il nome non viene richiesto esplicitamente in fase di registrazione, ma viene derivato automaticamente dall'indirizzo e-mail: se disponibile viene recuperato il nome associato all'account Google, altrimenti viene estratto il nome dell'indirizzo e-mail (prima della "@") rendendo maiuscola la prima lettera. Questo permette di mostrare un identificativo leggibile e coerente anche nel caso di autenticazione tramite e-mail. Sotto il nome viene riportato l'indirizzo e-mail completo associato all'account, come informazione di riferimento.

La seconda sezione contiene le azioni disponibili. L'utente può effettuare il logout oppure eliminare definitivamente il proprio account.

L'eliminazione dell'account è un'operazione irreversibile e viene protetta da una finestra di conferma. Solo dopo l'esplicita accettazione da parte dell'utente viene inviata la richiesta di cancellazione tramite Firebase. In caso di errore viene mostrato un messaggio informativo; in caso di successo, la sessione viene chiusa e l'app ritorna allo stato non autenticato.

La sezione Profile è progettata per essere essenziale e funzionale: mostra in modo chiaro l'identità dell'utente e fornisce strumenti diretti per la gestione dell'account.

Architettura e Scelte progettuali

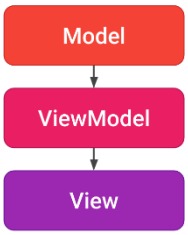
La progettazione di MyBooks è stata impostata fin dalle prime fasi con un'attenzione specifica alla struttura architetturale, con l'obiettivo di garantire separazione delle responsabilità, riduzione dell'accoppiamento tra componenti e facilità di evoluzione nel tempo.

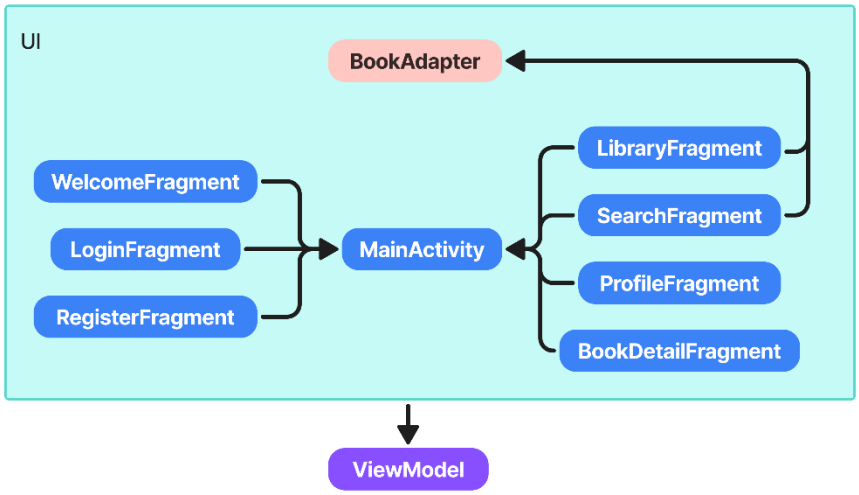
L'applicazione non è stata sviluppata come un insieme di schermate collegate tra loro, ma come un sistema organizzato su livelli distinti, ciascuno con compiti ben definiti.

MVVM- ViewModel e Fragment

L'architettura adottata segue il pattern MVVM (Model-View-ViewModel), supportato dai componenti Jetpack. Questo modello consente di separare la logica di presentazione dalla gestione dei dati, evitando che l'interfaccia utente dipenda direttamente dalle sorgenti dati o dalla logica applicativa.

- View (activity e fragment) si occupa esclusivamente della rappresentazione dai dati e dell'interazione con l'utente.
- Model (repository, datasource, modelli) centralizza l'accesso ai dati, isolando completamente la View dalle implementazioni concrete di rete o persistenza.
- ViewModel (ViewModel) gestiscono lo stato e coordinano le operazioni.





Single Activity

L'applicazione è strutturata secondo il principio della Single Activity con l'obiettivo di centralizzare la gestione del lifecycle e delegare ai Fragment la rappresentazione delle diverse schermate applicative. Questa scelta consente di mantenere un unico punto di controllo per la navigazione e per lo stato globale dell'applicazione.

La navigazione è gestita tramite Navigation Component, che permette di definire il grafo delle destinazioni. In questo modo il flusso tra le schermate di autenticazione e le sezioni interne principali è coerente e controllato centralmente.

Il controllo dello stato di autenticazione è affidato al MainViewModel, il ViewModel principale, che osserva lo stato della sessione utente. MainActivity osserva a sua volta il dato esposto da questo ViewModel e mostra la schermata corretta di conseguenza. Il cambio di schermata al variare dello stato di autenticazione viene gestito con popUpTo dell'intero nav_graph, garantendo la pulizia dello stack e impedendo all'utente di visualizzare le schermate di autenticazione in caso di sessione valida, o viceversa, di mostrare le schermate principali a un utente non autenticato.

Firebase e Firestore

Abbiamo scelto di utilizzare **Firebase Authentication** per la gestione dell'autenticazione e **Cloud Firestore** per la persistenza remota dei libri salvati dall'utente. Questa scelta ci ha permesso di delegare la gestione della sicurezza, delle credenziali e della sessione a un'infrastruttura solida e scalabile, evitando la necessità di implementare un backend proprietario.

L'autenticazione è incapsulata nel livello FirebaseAuthDataSource, che espone lo stato dell'utente tramite LiveData<FirebaseUser>. Questo stato viene propagato da UserRepository fino al MainViewModel, permettendo all'applicazione di reagire automaticamente a login, logout o eliminazione dell'account senza logica di navigazione distribuita nei Fragment.

Per quanto riguarda i dati, Firestore memorizza i libri in una collezione associata all'utente autenticato (users/{uid}/books). L'accesso al database remoto è gestito dal FirestoreDataSource, mentre il BookRepository coordina le operazioni di salvataggio, eliminazione e sincronizzazione, mantenendo separato il livello dati, sia locale, sia remoto, dalla UI.

In questo modo Firebase gestisce l'identità dell'utente, mentre Firestore garantisce la persistenza remota dei dati personali, integrandosi in modo coerente con l'architettura a repository adottata nel progetto.

Room

Abbiamo introdotto **Room** come livello di persistenza locale per garantire reattività dell'interfaccia e supporto offline. I libri salvati dall'utente non vengono letti direttamente da Firestore nella UI, ma passano attraverso il database locale, che rappresenta la **single source of truth** per la sezione Library.

Il BookDao espone query che restituiscono LiveData<List<Book>>, permettendo ai Fragment di osservare automaticamente le variazioni del database senza gestire manualmente aggiornamenti o refresh.

Quando un libro viene salvato o rimosso, il BookRepository svolge il ruolo di coordinatore tra livello remoto e locale. L'operazione viene prima eseguita su Firestore tramite il FirestoreDataSource, utilizzando una callback per intercettarne l'esito. Solo in caso di successo viene avviato l'aggiornamento del database locale Room, così da mantenere coerenza tra stato remoto e stato persistito sul dispositivo. L'accesso a Room avviene su thread dedicato tramite AppExecutors, evitando operazioni di I/O sul main thread e garantendo la reattività dell'interfaccia.

All'avvio del BookViewModel viene inoltre eseguita una sincronizzazione iniziale (refreshSavedBooks) che allinea il contenuto remoto con quello locale.

Room garantisce velocità di accesso, aggiornamento automatico della UI e continuità di utilizzo anche in condizioni di rete instabile.

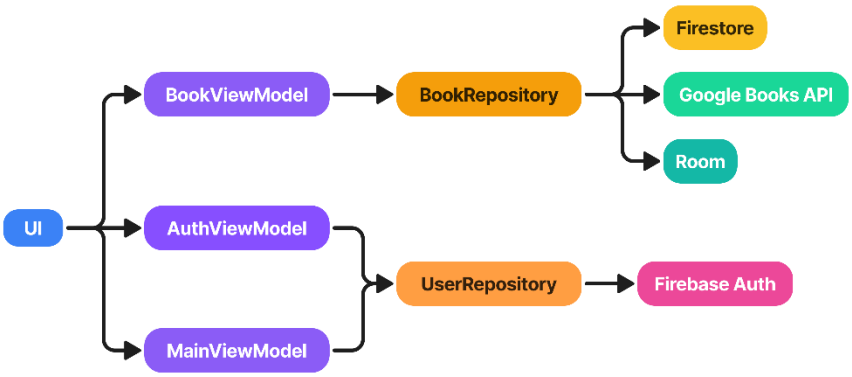
Data la natura effimera dai risultati di ricerca, abbiamo deciso di non memorizzare in modo persistente questi dati.

Retrofit

Per l'integrazione con la **Google Books API** abbiamo adottato **Retrofit**, che consente di definire le chiamate HTTP tramite interfacce tipizzate (BookApiService) mantenendo il codice leggibile e modulare.

Il BookRepository invoca apiService.searchBooks(query) in modo asincrono; durante la richiesta viene notificato lo stato di caricamento, osservato dal SearchFragment, che di conseguenza mostra la ProgressBar. La risposta JSON viene mappata prima in un oggetto compatibile con la struttura della stessa tramite BookApiResponse, e successivamente convertito in un oggetto del dominio tramite BookApiMapper, creando una corrispondenza tra la struttura del dato proveniente da API e quello gestito all'interno dell'app.

Questa scelta permette di isolare completamente il livello di rete dalla UI: il Fragment non conosce Retrofit né la struttura della risposta HTTP, ma osserva semplicemente LiveData<List<Book>> esposto dal BookViewModel. In questo modo la comunicazione con servizi esterni resta confinata nel repository, coerentemente con il pattern MVVM adottato.



Tecnologie







Google Books APIs

Sviluppi futuri

Architettura

Conformità completa a clean architecture, con strato “domain” di disaccoppiamento tra UI e Data. Conseguente stesura di classi di unit test sulla logica di dominio.

Introduzione interfacce DataSource per astrarre e standardizzare i comportamenti delle sorgenti dati utilizzate dal repository.

Introduzione di ServiceLocator per delegare la gestione dei DataSource fuori dal repository.

Migliore applicazione della dependecy injection.

Funzionalità

Supporto a librerie multiple, per catalogare libri in base allo stato di lettura.

Possibilità di aggiungere note e valutazioni personali per ogni libro.

Possibilità di condividere la propria libreria con altri utenti.