



Fastest Lap

The Coffee Coders

Table of Contents

Introduzione	3
Strumenti di Lavoro	3
1. Funzionalità	4
1.1 Autenticazione.....	4
1.2 Home Fragment	4
1.3 Racing Fragment	4
1.4 Standings Fragment	5
1.5 Driver, Constructor e Track Bio	5
2. Architettura	7
2.1 Struttura del Package.....	8
2.1.1 <i>adapter</i>.....	8
2.1.2 <i>api</i>.....	8
2.1.3 <i>database</i>	8
2.1.4 <i>domain</i>	8
2.1.5 <i>dto</i>.....	9
2.1.6 <i>mapper</i>.....	9
2.1.7 <i>repository</i>.....	9
2.1.8 <i>service</i>	9
2.1.9 <i>source</i>	9
2.1.10 <i>ui</i>.....	10
2.1.11 <i>util</i>	10
3. Design	11
3.1 Layouts.....	11
3.2 Color Palette e Tipografia	15
Conclusioni	16
Punti di Forza.....	16
Ostacoli Incontrati	16
Sviluppi Futuri	17
Risultati delle Sessioni.....	17
Dati Live in Tempo Reale	17

Introduzione

FastestLap è un'applicazione mobile innovativa progettata per trasformare l'esperienza degli appassionati di Formula 1. L'app ridefinisce l'interazione con i contenuti sportivi, creando un ambiente personalizzato dove ogni utente può vivere la propria passione in modo unico e immersivo. Attraverso un sistema di preferenze, permette di selezionare piloti e scuderie preferiti, adattando dinamicamente l'interfaccia e i contenuti per riflettere le preferenze individuali di ciascun appassionato. Fornisce accesso completo al calendario dei Gran Premi, presentando sia gli eventi già conclusi sia quelli futuri con informazioni dettagliate, e offre un'esplorazione approfondita della storia attraverso statistiche inerenti team, piloti e circuiti. FastestLap supporta inoltre le lingue italiano e inglese, per garantire un'esperienza accessibile e su misura per ogni utente.

Strumenti di Lavoro

Per lo sviluppo di FastestLap sono stati utilizzati diversi strumenti, ciascuno con un ruolo specifico all'interno del processo di realizzazione dell'applicazione.

Android Studio è stato l'ambiente di sviluppo scelto per la creazione dell'app, utilizzando il linguaggio Java. Si tratta dell'IDE ufficiale per lo sviluppo di applicazioni Android, che offre strumenti avanzati per il design, il testing e il debug.

GitHub ha rappresentato la piattaforma centrale per la gestione del codice e la collaborazione tra i membri del gruppo. La repository del progetto (<https://github.com/mbroglio/FastestLap>) è stata organizzata in modo strutturato per facilitare il lavoro in team:

- Il branch *master* è utilizzato per le diverse release stabili e definitive dell'applicazione
- Il branch *dev* funge da ambiente di sviluppo condiviso, dove ciascun membro del gruppo può caricare modifiche o correzioni prima che queste siano integrate nella versione finale
- Le fasi intermedie di sviluppo si sono svolte principalmente su branch *ad hoc*, creati specificamente per l'implementazione di una determinata funzionalità oppure il refactoring di codice preesistente

Firebase, la piattaforma cloud offerta da Google, è stata utilizzata per la gestione del database e l'autenticazione. *Realtime Database* ci ha permesso di memorizzare informazioni in merito allo storico di team, piloti e circuiti oltre ai dati degli utenti e le proprie preferenze. Per mezzo di *Authentication* è stato possibile gestire l'autenticazione degli utenti ed infine tramite *Storage* abbiamo avuto una piattaforma per salvare tutti i dati multimediali in modo da facilitare l'aggiornamento di immagini quando necessario.

Infine, è stata integrata l'**API Jolpica-F1** (in precedenza nota come *Ergast*), un servizio gratuito e open source che fornisce dati aggiornati sul mondo della Formula 1. Grazie a questa API è stato possibile ottenere informazioni su piloti, scuderie, classifiche e risultati in tempo reale, rendendo l'app sempre aggiornata e ricca di contenuti.

1. Funzionalità

1.1 Autenticazione

L'accesso a Fastest Lap è semplice e sicuro.

Login e Registrazione: Al primo avvio, l'utente può accedere tramite le proprie credenziali (e-mail e password) se già registrato, oppure creare un nuovo account attraverso un intuitivo processo di registrazione, che richiede l'inserimento di indirizzo e-mail e password. È disponibile anche una funzione per il recupero della password smarrita.

Accesso Automatico: Per una maggiore comodità, se l'utente ha già effettuato l'accesso in precedenza dal medesimo dispositivo, l'autenticazione avverrà automaticamente, indirizzandolo direttamente alla schermata Home.

1.2 Home Fragment

La schermata Home funge da dashboard centrale, fornendo una panoramica immediata delle informazioni più rilevanti:

Ultimo Gran Premio: Nella parte superiore, vengono visualizzati il nome, la data e il layout del circuito dell'ultimo Gran Premio disputato. Accanto al tracciato, sono presentati i risultati principali della gara, nel caso in cui essi non dovessero essere ancora disponibili, l'utente verrà informato con apposito messaggio tramite un layout appositamente creato. Al termine della stagione, questa sezione si aggiornerà mostrando le classifiche finali del campionato mondiale piloti e costruttori.

Prossimo Evento: Nella seconda card, l'utente troverà tutte le informazioni sul prossimo appuntamento in calendario: nazionalità e nome del Gran Premio, accompagnati da un conto alla rovescia dinamico che indica il tempo rimanente alla sessione successiva (prove libere, qualifiche o gara). A stagione conclusa, il countdown lascerà spazio all'indicazione "Season Ended".

Preferiti: La parte inferiore della Home è dedicata alla personalizzazione, essa è composta da due card distinte legate alle preferenze del pilota e del team. L'utente può scegliere di impostare un pilota e una scuderia preferiti. Una volta selezionati, verranno mostrati i loro nomi, nazionalità e le rispettive posizioni attuali nelle classifiche piloti e costruttori. Inizialmente, queste sezioni mostreranno all'utente di indicare la propria preferenza.

1.3 Racing Fragment

La sezione Racing è il cuore pulsante per chi vuole rivivere le emozioni passate o prepararsi per quelle future, offrendo due percorsi distinti:

Gare Disputate ("Past Races"): Selezionando questa opzione, l'utente accede all'archivio storico delle gare. Ogni evento è presentato con data, layout del tracciato e il podio. Un tap sulla card di una specifica gara apre una schermata di dettaglio con il podio, cliccabile al fine di

ottenere la classifica completa del Gran Premio ed informazioni in merito al giro veloce (Fastest Lap, da cui il nome dell'applicazione) oltre al programma dell'intero weekend di gara.

Prossime Gare ("Upcoming Races"): Questa area mostra gli eventi futuri, con indicazioni del tracciato, nome del Gran Premio e data. Un pallino rosso permetterà di identificare facilmente il gran premio in corso. Similmente alle gare passate, selezionando una card si accede al programma dettagliato del weekend e a un countdown specifico per l'inizio della sessione selezionata, un pulsante in fondo alla schermata permetterà di visualizzare le condizioni meteo presenti al momento sul tracciato, così da ottenere un'esperienza ancora più immersiva.

1.4 Standings Fragment

La schermata Standings offre una visione completa delle performance nel campionato, suddivisa per piloti e team:

Classifica Piloti: Selezionando l'apposita opzione, l'utente visualizza la graduatoria completa dei piloti. Ogni pilota è rappresentato da una card contenente: posizione in classifica, foto, logo della scuderia, nome e cognome, e punteggio totale.

Classifica Costruttori: Accedendo a questa sezione, viene mostrata la classifica delle scuderie. Ogni team ha una card che riassume: posizione, denominazione ufficiale, punteggio complessivo e i nomi dei piloti che ne fanno parte.

1.5 Driver, Constructor e Track Bio

FastestLap permette di esplorare in profondità ogni aspetto del motorsport attraverso schermate di dettaglio dedicate:

Track Bio: Cliccando sul layout del tracciato in qualsiasi schermata relativa a un Gran Premio, l'utente accede a una scheda informativa completa del circuito. Questa include:

- Un **layout dettagliato** con numerazione delle curve, indicazione del senso di marcia e localizzazione delle zone DRS.
- **Curiosità e dati tecnici:** prima apparizione nel calendario ufficiale, numero di giri previsti per la gara, lunghezza del singolo giro e lunghezza totale della gara.
- Un **pulsante di geolocalizzazione** che rimanda alla posizione del circuito su Google Maps.
- Una **tabella degli storici** delle edizioni passate su tale tracciato, indicante anno, podio (primi tre classificati) e la possibilità di accedere agli highlights video della gara (se disponibili).

Driver Bio: Dalla classifica piloti, selezionando la card di un singolo pilota, si accede alla sua biografia e al suo storico. La schermata dedicata mostra:

- **Informazioni anagrafiche e sportive:** team di appartenenza attuale, nazionalità, data e luogo di nascita, età, altezza, peso, miglior risultato ottenuto in carriera, numero totale di podi, campionati mondiali vinti e anno della prima apparizione in Formula 1.

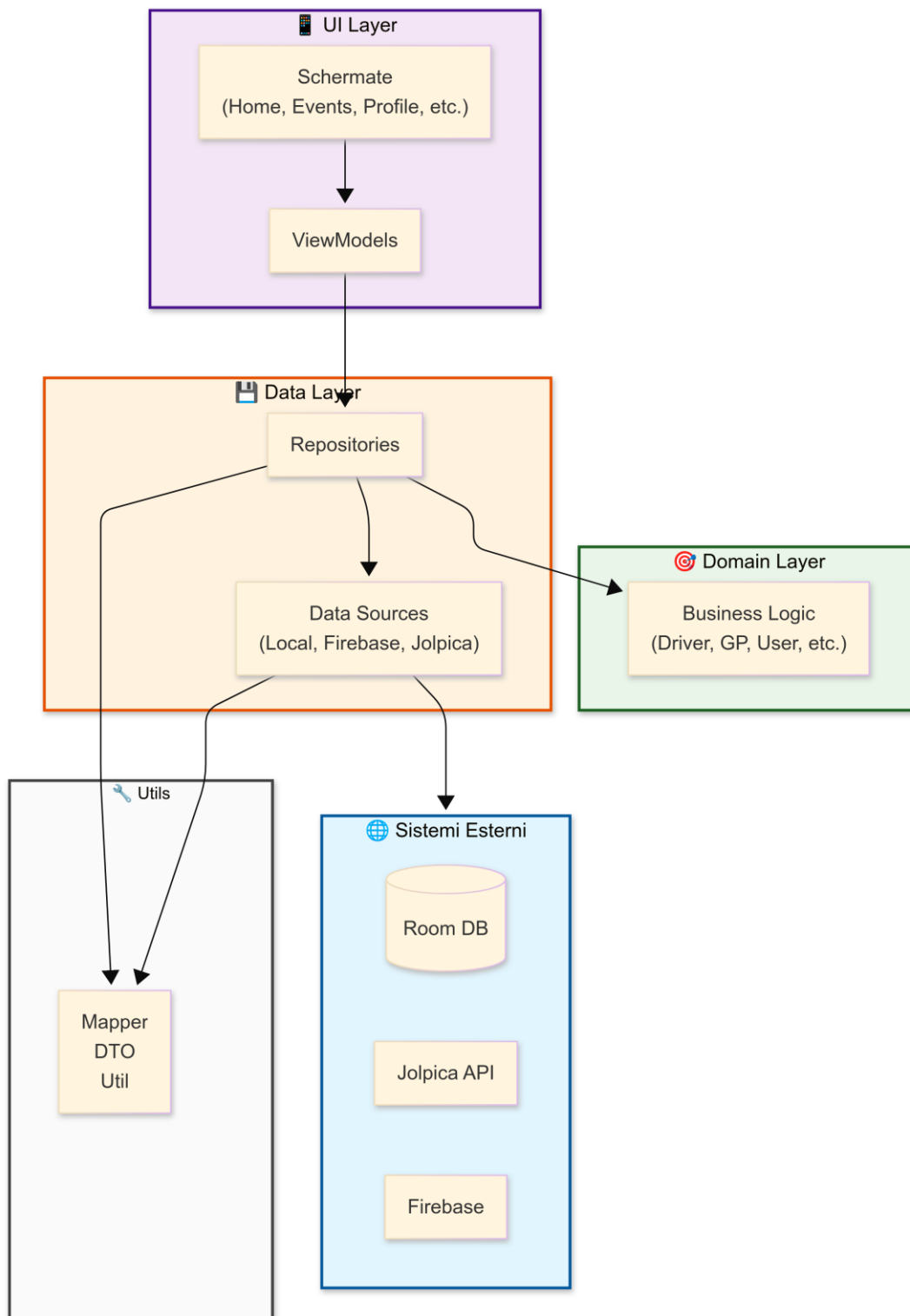
- **Storico delle performance annuali:** una tabella dettagliata che, per ogni anno di partecipazione, riporta la posizione finale in campionato, il team di appartenenza in quella stagione, i punti totalizzati, il numero di vittorie e di podi conseguiti.

Constructor Bio: Dalla classifica costruttori, cliccando sulla card di una scuderia, l'utente visualizza una pagina ricca di informazioni sul team:

- **Dati identificativi e tecnici:** nazionalità, piloti attuali, nome completo ufficiale, sede principale, nome del Team Principal, specifiche del telaio e della power unit utilizzata.
- **Storia:** anno della prima apparizione in Formula 1, numero di campionati mondiali costruttori vinti, numero totale di vittorie e di podi.
- **Storico delle performance annuali del team:** una tabella che, per ogni anno di partecipazione, indica la posizione finale nel campionato costruttori, i punti totalizzati e il numero di vittorie stagionali.

2. Architettura

Fastest Lap segue un'architettura **Clean Architecture** con **Pattern MVVM** (Model-View-ViewModel). L'applicazione gestisce dati relativi a piloti, costruttori, classifiche, gare e risultati attraverso multiple sorgenti dati (API remote, database locale, Firebase) tramite un **Pattern Repository** per ogni dominio.



2.1 Struttura dei Package

2.1.1 adapter

Questo package contiene tutte le classi Adapter utilizzate per popolare le RecyclerView all'interno dell'interfaccia utente. Gli adapter fungono da ponte tra i dati dell'applicazione e le viste che li visualizzano in elenchi.

2.1.2 api

Il package api definisce le classi che modellano le risposte attese dalle API esterne (RESTful). Queste classi sono utilizzate con la libreria di networking Retrofit per deserializzare i dati JSON in oggetti Java.

- *APIResponse*: Classe base generica per le risposte API.
- **APIResponse*: Classi specifiche che mappano le risposte per classifiche piloti, costruttori e risultati di gara.

2.1.3 database

Questo package è responsabile della **persistenza dei dati locali** utilizzando la libreria Room Persistence. Contiene le entità del database, i Data Access Objects (DAO) e la definizione del database stesso.

- *AppRoomDatabase*: Classe principale che definisce il database Room, le sue entità e fornisce l'accesso ai DAO.
- **DAO*: Interfacce DAO che definiscono le operazioni CRUD (Create, Read, Update, Delete) per le rispettive entità (costruttori, piloti, gare, risultati, classifiche).
- *DatabaseConverters*: Contiene i type converter necessari a Room per gestire tipi di dati personalizzati.

2.1.4 domain

Il package *domain* rappresenta il cuore dell'applicazione, contenente la logica di business e le entità del dominio.

- *constructor*:
 - *Constructor*: Rappresenta un team di F1.
- *driver*:
 - *Driver*: Rappresenta un pilota di F1.
- *grand_prix*: Contiene le entità legate agli eventi di gara.
 - *Race*: Rappresenta un evento di gara.
 - *RaceResult*: Modella il risultato di un pilota in una gara.
 - *DriverStandings*, *ConstructorStandings*: Rappresentano le classifiche.
 - *Track*: Descrive un circuito di gara.

- *WeeklyRace*: Rappresenta l'intero fine settimana gi gara
- *user*:
 - *User*: Rappresenta l'utente dell'applicazione.
- *Result*: Classe generica per incapsulare l'esito di un'operazione (successo o errore).

2.1.5 *dto*

Il package *dto* (Data Transfer Objects) contiene oggetti semplici usati per trasferire dati tra i layer dell'applicazione, specialmente tra il layer di rete/database e il layer di dominio, o tra il dominio e l'UI. Questi oggetti mappano direttamente le strutture dati delle API o del database prima di essere convertiti in oggetti di dominio.

2.1.6 *mapper*

Questo package contiene classi Mapper responsabili della conversione tra diversi tipi di oggetti, ad esempio da DTO a oggetti di dominio, o da oggetti di dominio a modelli specifici per l'UI. Aiuta a mantenere i layer disaccoppiati.

2.1.7 *repository*

Il package è organizzato per dominio funzionale, con una coppia di classi principali per ciascuna entità:

- **Repository*: Interfacce o classi concrete che espongono i metodi per l'accesso ai dati.
- **Callback*: Interfacce utilizzate per gestire le risposte asincrone, tipiche dell'accesso a dati remoti (es. API REST, Firebase, ecc.).

Le Callback permettono di gestire in modo efficace l'asincronia tipica delle operazioni I/O, soprattutto in ambienti come Android o app web/mobile, dove le chiamate di rete non possono bloccare il thread principale.

2.1.8 *service*

L'interfaccia *service* è utilizzata per raggiungere i diversi endpoint di Jolpica API

2.1.9 *source*

Il package *source* contiene le implementazioni concrete dei **Data Sources**. Questi sono responsabili dell'effettivo recupero o salvataggio dei dati da una specifica origine (es. Firebase, API remota Jolpica, database Room locale).

La struttura è granulare, con sotto-package per ogni tipo di entità (constructor, driver, ecc.) e implementazioni specifiche per diverse sorgenti.

- *Firebase*DataSource* (es. *FirebaseConstructorDataSource*): Sorgenti dati che interagiscono con Firebase.
- *Jolpica*DataSource* (es. *JolpicaConstructorDataSource*): Sorgenti dati che interagiscono con l'API Jolpica.
- *Local*DataSource* (es. *LocalConstructorDataSource*): Sorgenti dati che interagiscono con il database Room locale.
- *UserAuthenticationFirebaseDataSource*: Gestisce l'autenticazione tramite Firebase.

2.1.10 *ui*

Il package *ui* contiene tutti i componenti relativi all'**interfaccia utente e alla logica di presentazione**. Segue comunemente il pattern MVVM (Model-View-ViewModel).

Ogni package è organizzato per feature o schermate, contiene un package *viewModel* il quale contiene le effettive classi **ViewModel* le quali fungono da interfaccia tra dati e viste. Le classi **ViewModelFactory* sono utilizzate per istanziare i *ViewModel* secondo il **Pattern Factory**.

2.1.11 *util*

Questo package contiene **classi di utilità e helper** generiche, riutilizzabili in diverse parti dell'applicazione.

- *Constants*: Definisce costanti globali per l'applicazione.
- *GlideModule*: Utilizzato al fine di aumentare la Cache di Glide
- *JSONParserUtils*: Utilità per il parsing di JSON.
- *NetworkUtils*: Helper per operazioni di rete (es. controllo connettività).
- *RetryInterceptor*: Implementa la logica di Retry per l'errore 429 ("Too Many Requests")
- *ServiceLocator*: Un semplice meccanismo di dependency injection o service location.
- *SharedPreferencesUtils*: Utilità per la gestione delle *SharedPreferences*.
- *UIUtils*: Helper per operazioni comuni legate all'interfaccia utente.

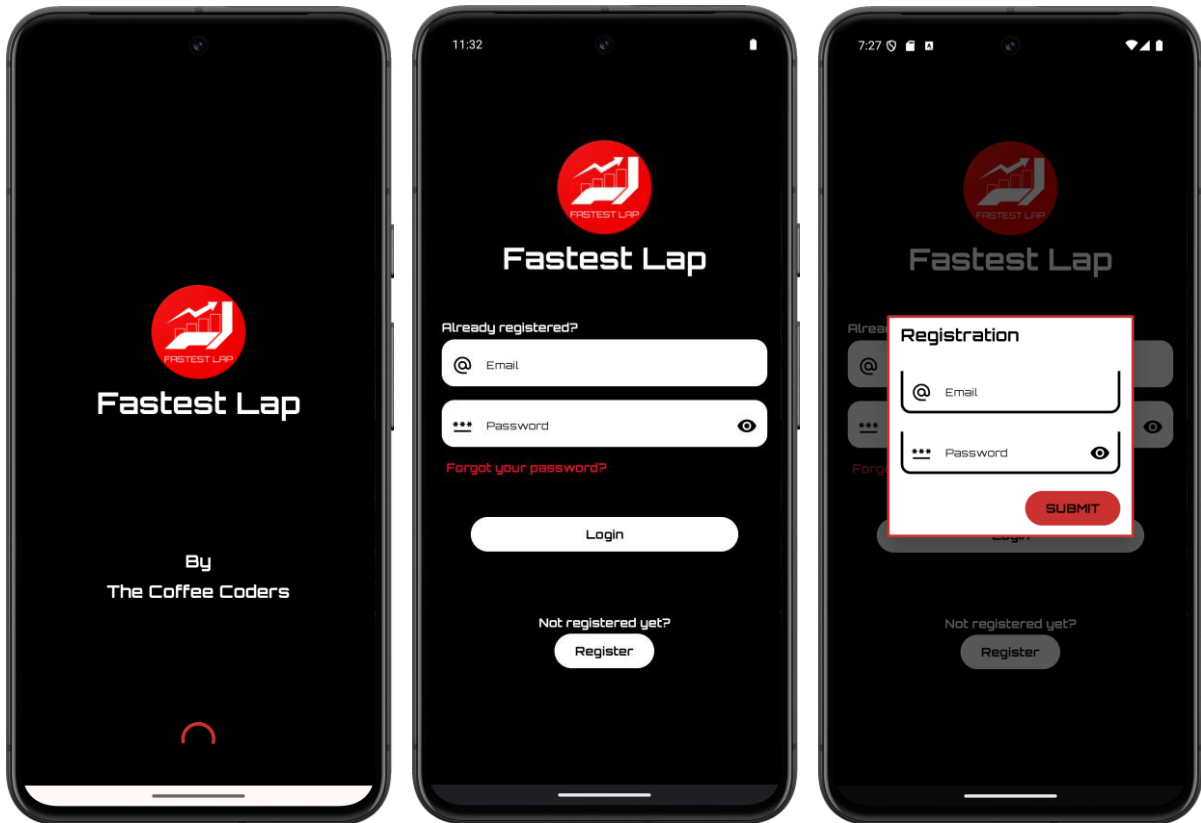
3. Design

Il design di FastestLap è stato concepito per offrire un'esperienza utente moderna, intuitiva e coinvolgente. L'interfaccia grafica è stata progettata seguendo i principi del **Material Design**, garantendo coerenza visiva, facilità di navigazione e accessibilità su diversi dispositivi Android. Ogni elemento visivo è stato studiato in modo tale da attirare il focus dell'utente, con particolare attenzione all'usabilità e alla leggibilità delle informazioni presentate.

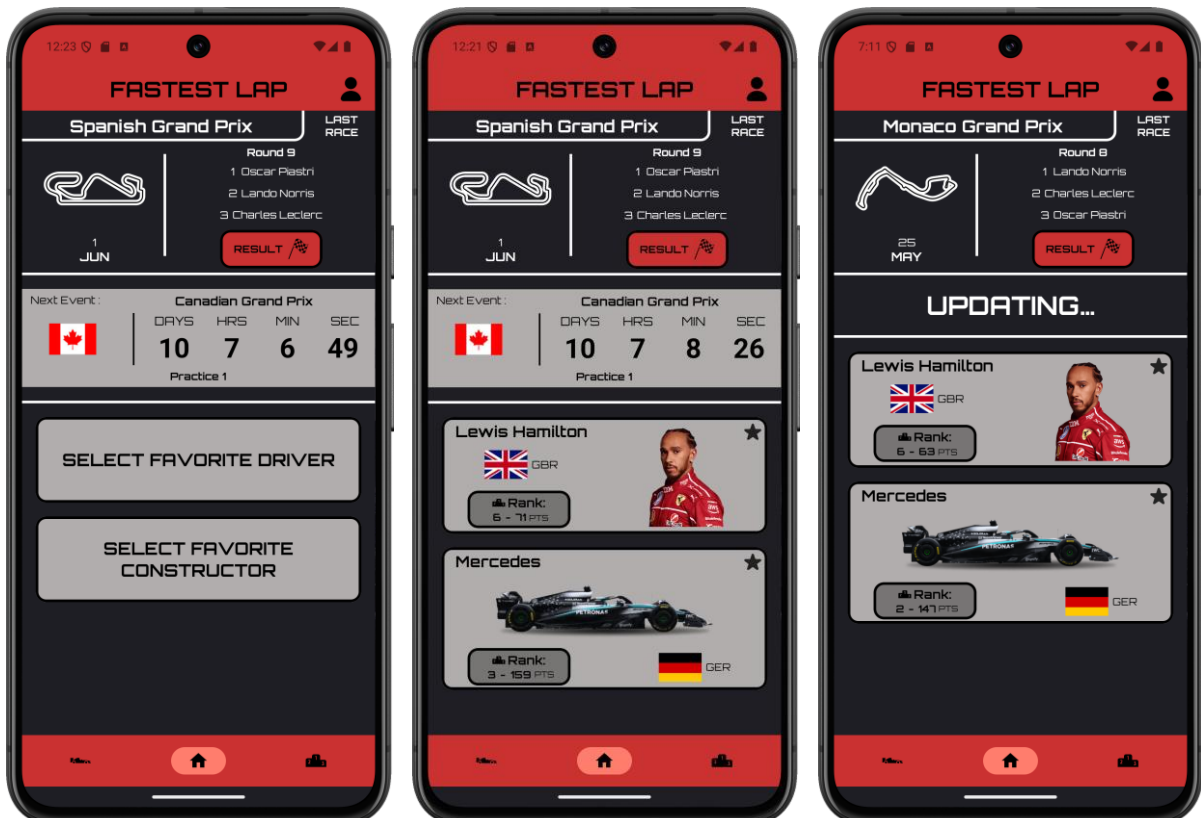
3.1 Layouts

La progettazione dei layout di FastestLap ha seguito un approccio user-centered, privilegiando la chiarezza informativa e la facilità di interazione. Ogni schermata è stata strutturata per massimizzare l'efficienza nella consultazione dei dati, utilizzando card ben organizzate e gerarchie visive chiare che guidano naturalmente l'occhio dell'utente verso le informazioni più rilevanti. I layout sono stati ottimizzati per garantire un'esperienza fluida adattandosi dinamicamente alle diverse dimensioni di schermo per offrire sempre la migliore presentazione possibile dei contenuti.

Login e Registrazione



Home e Varianti



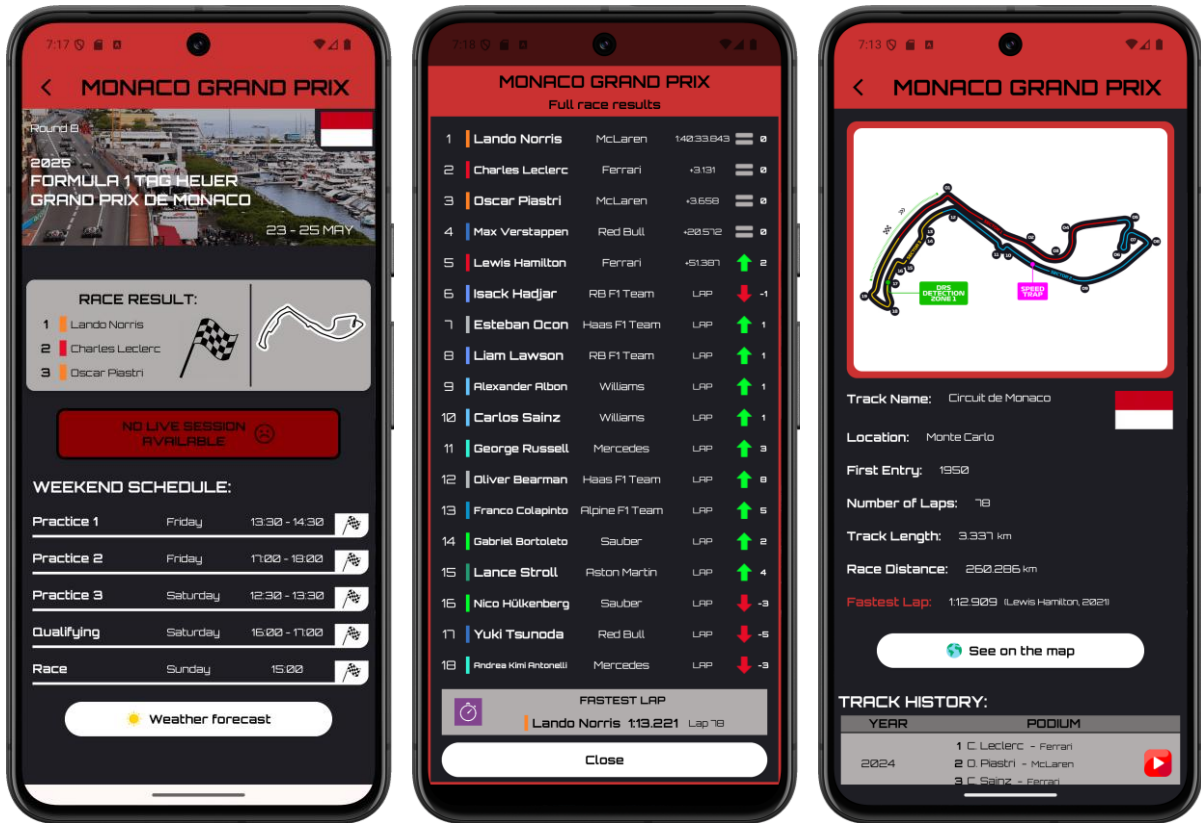
Calendario



Classifiche



Evento



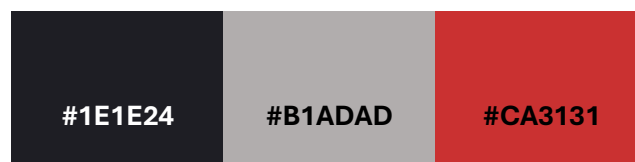
Biografie e Gestione Profilo



3.2 Color Palette e Tipografia

La palette cromatica di FastestLap si basa su tre colori principali che definiscono l'identità visiva dell'applicazione. Il colore primario **#1E1E24** è un grigio scuro che funge da sfondo principale, garantendo una buona leggibilità e riducendo l'affaticamento visivo durante l'utilizzo. Il grigio medio **#B1ADAD** viene utilizzato per le varie card, creando una gerarchia visiva chiara e facilitando la navigazione nell'interfaccia.

Il rosso **#CA3131** rappresenta l'elemento di accento della palette, utilizzato per pulsanti primari, e toolbar. Questa tonalità di rosso è stata scelta per la sua capacità di attirare l'attenzione mantenendo un buon contrasto con i colori di base, richiamando allo stesso tempo il colore Rosso caratteristico di questo motorsport.



Per la tipografia è stato selezionato il font **Orbitron**, un carattere geometrico dalle linee moderne che trasmette dinamismo e velocità. Le sue forme distintive si integrano bene con il tema tecnologico dell'applicazione, mentre la sua leggibilità ottimale su dispositivi mobili garantisce una buona esperienza utente. Il font contribuisce a rafforzare l'identità visiva dell'app, comunicando efficacemente i valori di innovazione e performance associati al mondo della Formula 1.

Fastest Lap

Conclusioni

FastestLap rappresenta una soluzione completa per gli appassionati di Formula 1, offrendo un'esperienza personalizzata che spazia dalla consultazione di dati storici alle informazioni aggiornate sui Gran Premi. L'architettura Clean Architecture adottata, insieme al pattern MVVM, ha garantito un codice manutenibile e scalabile, mentre l'integrazione di multiple sorgenti dati ha permesso di creare un'esperienza fluida e sempre aggiornata.

Punti di Forza

Flessibilità nella Gestione dei Dati: Grazie all'integrazione con Firebase e Firestore, aggiornamenti di piloti, scuderie e immagini vengono gestiti in tempo reale senza necessità di aggiornamenti dell'app, mantenendo l'interfaccia utente sempre aggiornata.

Architettura Modulare: L'adozione del pattern Clean Architecture con MVVM garantisce manutenibilità ottimale e facilita l'aggiunta di nuove funzionalità senza compromettere l'esistente.

Esperienza Personalizzata: Il sistema di preferenze permette una personalizzazione profonda, adattando dinamicamente i contenuti agli interessi specifici di ogni utente.

Resilienza dei Dati: Multiple sorgenti dati (locale, remota, Firebase) garantiscono disponibilità delle informazioni, seppur limitata, anche offline.

Design Moderno: L'interfaccia basata sui principi del Material Design crea un'esperienza visiva coinvolgente che riflette l'energia del mondo della Formula 1.

Ostacoli Incontrati

Gestione dell'API in Continua Evoluzione: L'API Jolpica-F1 ha subito diversi aggiornamenti strutturali durante lo sviluppo, richiedendo continui adattamenti del codice e revisioni dei mapper, con un considerevole overhead di manutenzione.

Rivisitazione dell'API Open-F1: La completa rivisitazione dell'architettura dell'API per i dati live ha comportato un rallentamento significativo del progetto, impedendo l'implementazione dei dati in tempo reale nella versione attuale.

Complessità del Dominio: Modellare correttamente il mondo della Formula 1, con le sue diverse modalità di Gara e statistiche interconnesse, ha rappresentato una sfida significativa in termini di design dell'architettura.

Interfacciamento Firebase - API: L'integrazione tra i dati dell'API Jolpica-F1 e Firebase ha richiesto logiche complesse di sincronizzazione, gestione dei conflitti e ottimizzazione delle performance.

Sviluppi Futuri

Il roadmap di sviluppo di FastestLap prevede l'implementazione di funzionalità avanzate che porteranno l'esperienza utente a un livello ancora più immersivo e professionale.

Risultati delle Sessioni

Integrazione, all'interno della pagina legata all'Evento dei risultati relativi a Qualifiche e Gare Sprint, resi disponibili dall'API Jolpica nell'ultimo mese.

Dati Live in Tempo Reale

L'integrazione di feed live durante i weekend di gara rappresenta la prossima frontiera dell'applicazione. Questa funzionalità includerà:

Intertempi Live: Visualizzazione in tempo reale dei tempi di ogni settore per tutti i piloti durante prove libere, qualifiche e gara, con aggiornamento live.

Team Radio: Accesso alle comunicazioni radio tra piloti e box, offrendo agli utenti la possibilità di ascoltare le strategie, le decisioni tattiche e le emozioni dal vivo direttamente dal cockpit.

Gestione degli Stint: Monitoraggio dettagliato delle strategie di gara, inclusi i cambi gomme, i tipi di mescola utilizzati ed il numero di giri per ogni stint.

Direzione Gara: Integrazione completa delle decisioni della direzione gara, con notifiche istantanee per penalità, investigazioni in corso, bandiere gialle, rosse e di sicurezza, oltre alla gestione della Safety Car e Virtual Safety Car.

L'obiettivo finale è quello di creare l'applicazione di riferimento per tutti gli appassionati di Formula 1, combinando la ricchezza dei dati storici con l'emozione dei dati live, in un'interfaccia sempre più raffinata e coinvolgente.