Corso di Programmazione Mobile

**Corsi per Tutti**

Progetto didattico per il Corso di Programmazione Mobile dell’anno accademico 2020/21

linea orizzontale

# Introduzione

A partire dalle specifiche fornite dal docente del corso, relativamente a un App per fruire di **corsi online**, abbiamo sviluppato due applicativi basati su una connessione real-time a Firebase per gestire l’autenticazione di un utente e la fruizione di corsi.

Una prima implementazione completa di tutte le caratteristiche richieste e con funzionalità aggiuntive è stata sviluppata su ambiente android, utilizzando Kotlin; in questa versione è stata resa possibile la fruizione dei corsi da parte di un utente registrato e la possibilità di interazione tramite iscrizione, inserimento nella wishlist, possibilità di recensire il corso, accessibilità a un forum…

Nella seconda implementazione, l’applicativo è stato sviluppato su un ambiente cross-platform, utilizzando Flutter. In questa versione alcune funzionalità sono state ridotte, quali la possibilità di interagire nel forum con altri utenti.

L’architettura generale di entrambe le applicazioni è la seguente:

* Una prima pagina di Login/Registrazione visualizzata alla prima accensione dell’app; una volta che l’utente si registra con i propri dati, l’app si ricorda del Login effettuato.
* Una volta effettuato l’accesso, viene mostrata all’utente l’interfaccia principale dell’applicazione, in cui vengono mostrati i vari corsi, raggruppati in base a dei criteri di popolarità, suggeriti e aggiunti di recente. Nella versione Android è stato inoltre aggiunto un player per riprendere l’ultima lezione interrotta.
* Varie sezioni sono state aggiunte, fra cui il catalogo in cui i corsi vengono suddivisi per categoria e vi è la possibilità di filtrare per ricerca, una pagina con i corsi a cui è stata sottoscritta un iscrizione, e una wishlist.
* Un Area Utente è stata aggiunta per dare la possibilità all’utente di gestire i propri dati, fra cui la possibilità di specificare categorie preferite, e gestire impostazioni relative al tema dell’app
* L’apertura di un corso, permette all’utente di aprire una pagina contenente le informazioni del corso. Tramite un apposito pulsante di iscrizione, l’utente può abilitare contenuti nascosti, tra cui dispense, lezioni, e un forum ( Forum: solo nella versione Android).

# Firebase - Autenticazione e Real-Time Database

Per quanto concerne la memorizzazione di dati, la fruizione di contenuti dinamici e le funzionalità di autenticazione è stato utilizzato il framework Firebase, fornito da Google.

Firebase mette a disposizione diversi servizi tra cui autenticazione, cloud-storage, database, etc. Per i nostri scopi ci siamo limitati ad implementare le funzionalità di autenticazione e un database no-sql.

I servizi di Autenticazione permettono la registrazione di utenti in un apposito database, in cui, userId, email e password vengono memorizzate e la possibilità di login agli utenti registrati. Per una migliore qualità dei contenuti fruibili dagli utenti abbiamo integrato delle tabelle custom collegate ad una database esterno, il real-time database di Firebase. Quindi, oltre a memorizzare dati strettamente legati all’autenticazione dell’utente, abbiamo creato delle tabelle Utente del real-time database legate a quelle di autenticazione dallo UserId, in cui abbiamo memorizzato dati personali, iscrizioni ai corsi, wishlist, ultime lezioni e categorie preferite. Segue un esempio delle tabelle di utenti registrati. 



Abbiamo “mappato” l’utente nel real-time database usando lo userId come identificativo e vari attributi per memorizzare nome e cognome, categorie preferite, iscrizioni ai corsi, wishlist, ultime lezioni visualizzate…

La struttura generale del real-time database è stata già brevemente illustrata per l’Utente, possiamo dire che esso fornisce una rappresentazione di tipo NoSQL e permette un’interazione con i dati in tempo reale; per tale motivo abbiamo deciso di utilizzarlo per la memorizzazione dei corsi, i quali sono costituiti da contenuti statici, quali le informazioni relative il corso, e dinamici, ad esempio il forum e le recensioni.

Per il singolo corso è stata pensata la seguente struttura:



dove molti attributi sono statici e modificabili solo all’interno del database stesso, mentre altri come il forum e le recensioni sono dinamici, modificabili dall’utente. Le recensioni sono infatti legate al singolo utente tramite la chiave relativa allo userId dell’utente stesso; il forum è invece parte integrante dell’applicazione Android, in cui ogni utente può porre una o più domande, a cui è possibile fornire una o più risposte.

L’interazione con Firebase è resa possibile in entrambe le piattaforme tramite dei pacchetti forniti online, e in particolare per stabilire una connessione ai due servizi è necessario: istanziare una *FirebaseAuth* per l’autenticazione, e una *Reference* al real-time database per poi “ascoltare” gli eventi, i cambiamenti nel db, e leggere un *DataSnapshot* poi iterato per restituire i dati nel formato desiderato.

In entrambe le applicazioni sono stati costruiti dei modelli basati sulla base del singolo Corso e dell’Utente per effettuare un mapping delle tabelle dal database a oggetti del relativo ambiente di sviluppo.

# Utenti - Casi d’uso

# La piattaforma realizzata per la fruizione di corsi è stata resa disponibile solo a utenti registrati su Firebase Authentication, infatti sono state impostate nel real-time database, regole di lettura/scrittura accessibili solo all’utente loggato.

Quindi un utente che accede per la prima volta all'applicazione, o che semplicemente non ha effettuato la procedura di Login, dovrà in fase di apertura dell’app Registrarsi, con Nome, Cognome, Email e password, o effettuare il Login se già registrato.

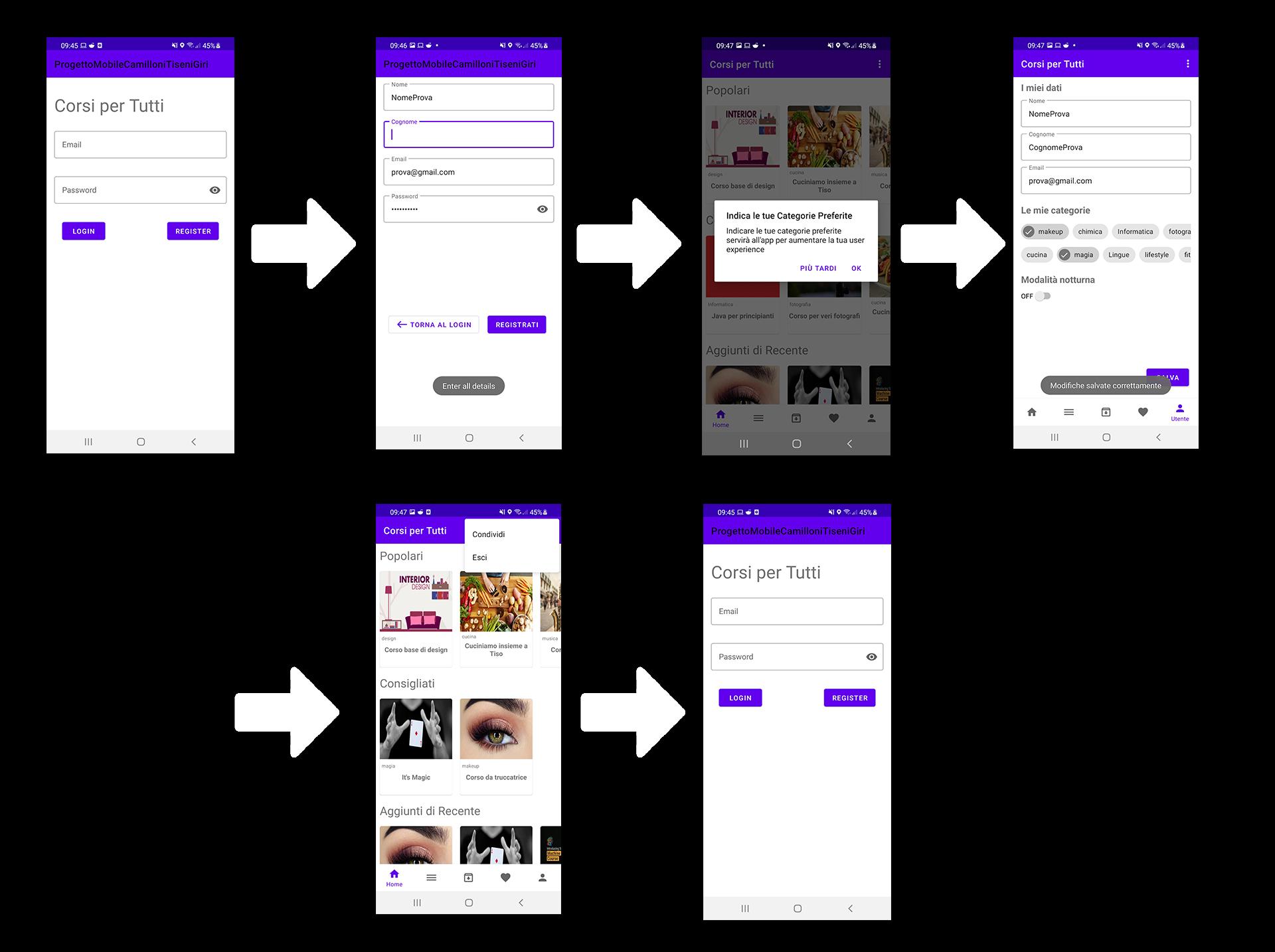
Un utente registrato potrà quindi accedere alle funzionalità principali dell’applicativo, quali la navigazione fra i corsi, visualizzazione della pagina relativa al corso e possibilità di modificare informazioni personali nella propria area utente.

Inoltre un utente registrato potrà poi accedere tramite iscrizione a un corso a svariati contenuti, tra cui una serie di video lezioni, dispense online, un forum dedicato in cui potrà interagire con altri partecipanti del corso, ed infine l’iscrizione rende possibile all’utente di recensire il corso. Esso può inoltre, indipendentemente dall’iscrizione inserire un corso nella propria wishlist.

Oltre ad accedere a una lista completa di corsi, l’utente specificando le proprie categorie preferite potrà accedere a tutti corsi consigliati in base alle proprie preferenze; altre liste visualizzabili sono la propria wishlist, i corsi a cui è iscritto e i corsi con le recensioni migliori.

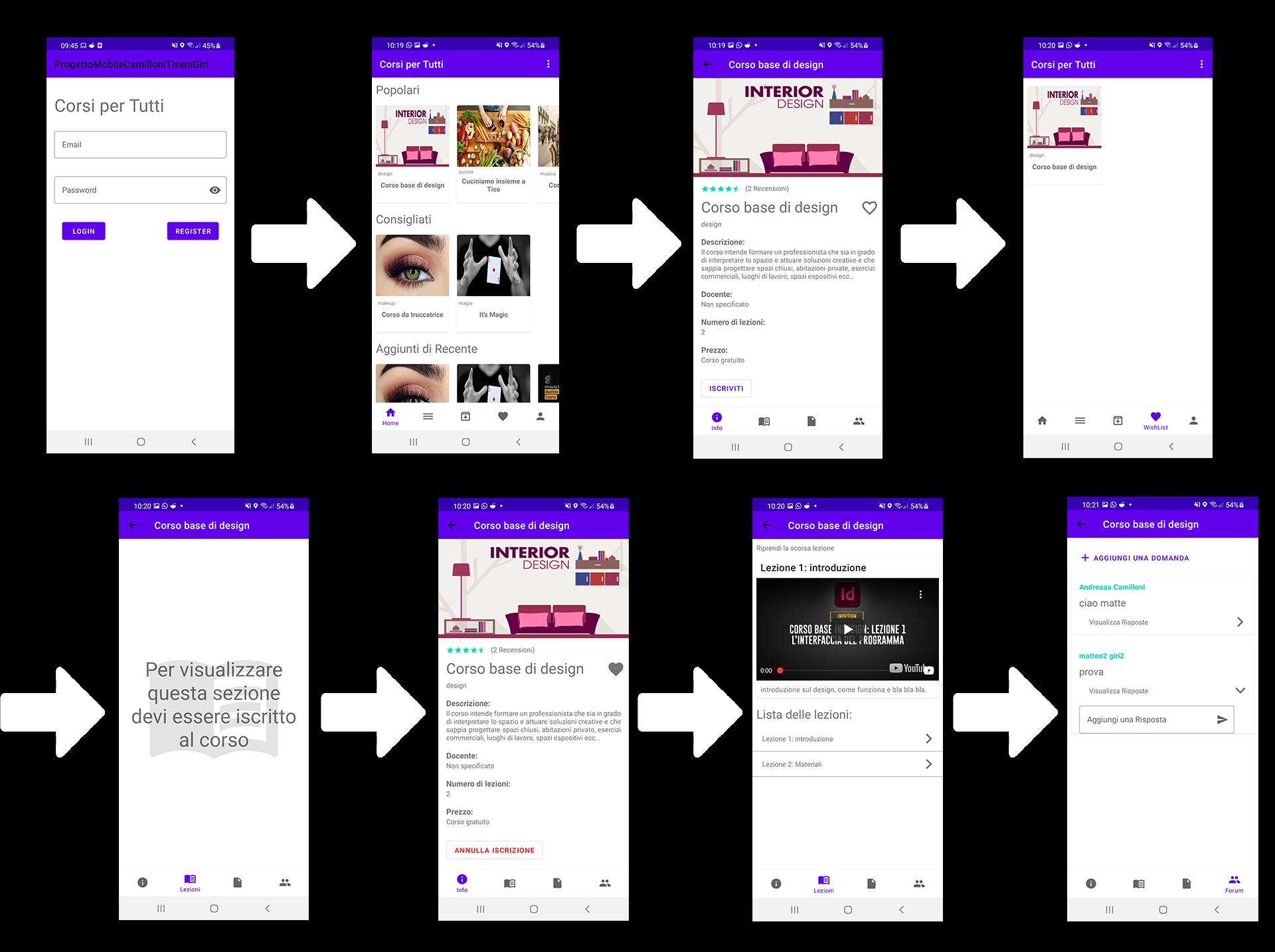
Verranno di seguito mostrati alcuni casi d’uso possibili, effettuati nell’applicazione android, ma che sono simili nell’applicazione flutter, tranne per alcune funzionalità ridotte.

Un primo esempio grafico di caso d’uso è mostrato nella figura sotto:

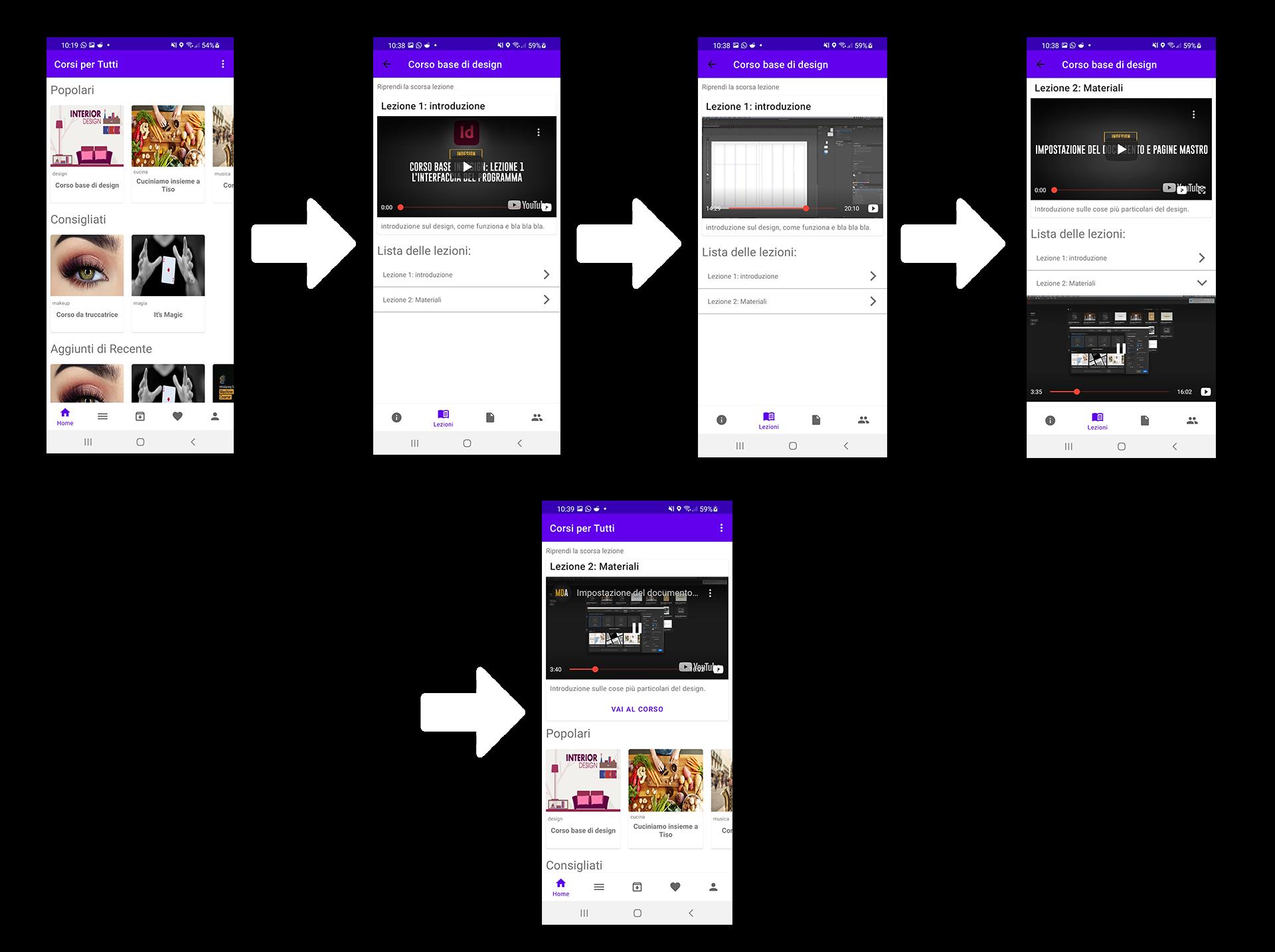


In questo esempio un utente che apre per la prima volta l’applicazione si ritrova nella pagina di login. Da quella passa alla pagina di registrazione tramite il bottone “registrati”. Registrandosi (per la registrazione e login sono stati inseriti vari controlli sulla correttezza dei dati), l’utente viene mandato alla homepage nella quale sarà presente un popup che chiede all’utente di inserire alcune categorie preferite, in modo tale da potergli mostrare nella homepage i corsi consigliati, ovvero alcuni corsi delle categorie che ha scelto. In questo popup ci sono due scelte: premendo “più tardi” il popup scomparirà e l’utente potrà visualizzare la homepage, nella quale i consigliati saranno semplicemente alcuni corsi presi dal database; premendo “Ok” l’utente verrà mandato all’area utente dove potrà scegliere le categorie spuntandole all’interno di una chipgroup e premendo il pulsante “salva”. Tornando alla homepage si vede infatti che nei corsi consigliati ci sono quelli delle categorie scelte. Premendo sui tre puntini in alto a destra l’utente può infine eseguire il logout, dopo il quale viene mandato nuovamente alla pagina di login.

Un secondo caso d’uso è mostrato nella figura seguente:

In questo esempio un utente già registrato effettua il login e viene mandato alla homepage. Avendo l’utente già delle categorie preferite, il messaggio di popup non viene visualizzato. Dalla homepage l’utente clicca su un corso, e viene rediretto alla pagina del corso nella quale può visualizzarne le informazioni. In questa pagina l’utente può iscriversi al corso o inserirlo nella wishlist. Nell’esempio l’utente aggiunge il corso alla wishlist premendo il cuoricino in alto a destra, e, tornando indietro e andando alla sezione “wishlist” può quindi vedere il nuovo corso aggiunto. Tornando al corso, l’utente non essendo iscritto non riesce a visualizzare le sezioni delle lezioni, dispense e forum. Iscrivendosi tramite il pulsante “iscriviti” in basso può quindi accedere alle varie sezioni. Nell’esempio vediamo l’utente accedere alla pagina delle lezioni e poi a quella del forum, nella quale può fare domande o rispondere a domande di altri utenti.

Un terzo ed ultimo caso d’uso è mostrato nella figura seguente:

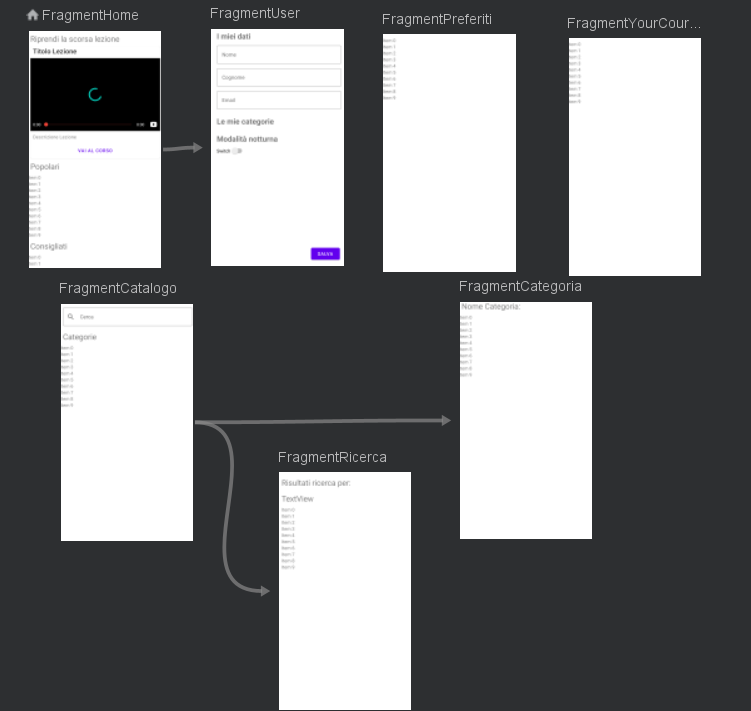


In questo esempio l’utente è già loggato ma non ha mai visto una lezione di nessun corso. Se l’utente non ha mai visto lezioni, l’homepage si mostra come in figura, mentre se un utente ha iniziato a vedere lezioni di qualche corso, comparirà una card nella quale continuare l’ultima lezione. In questo esempio l’utente apre un corso e va nella sezione delle lezioni. Se un utente non ha ancora visto nessuna lezione di uno specifico corso (come in questo caso), in alto come “riprendi ultima lezione” verrà mostrato il player della prima lezione del corso. L’utente potrà utilizzare questo player o quello all’interno del “cassetto della lezione” per vedere la lezione. Il player salva lo stato di visualizzazione, quindi se l’utente stoppa il video e ritorna successivamente in quel corso, potrà continuare dallo stesso minuto in cui è stata interrotta la lezione precedentemente. Nell’esempio l’utente sceglie poi di guardare la seconda lezione del corso. Questo comporta un aggiornamento della parte alta relativa all’ultima lezione vista, che mostrerà da ora la seconda lezione. Nell’esempio l’utente torna infine alla homepage, e come si vede dall’immagine, questa volta viene visualizzata una card in alto nella quale riprendere l’ultima lezione dell’ultimo corso visto, con un pulsante per andare direttamente al corso.

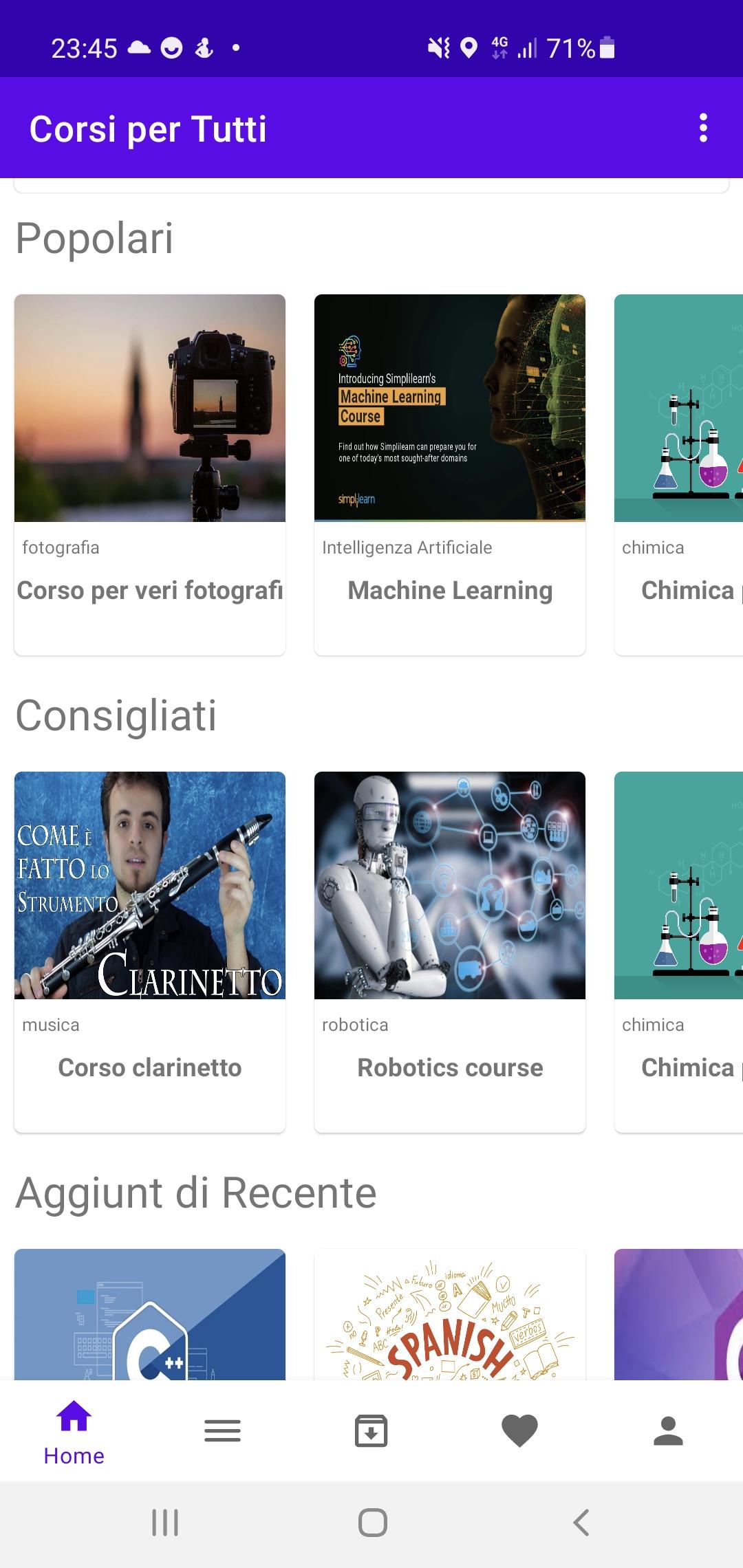
## Android

La versione Android è stata sviluppata partendo dalla realizzazione dell’interfaccia grafica, articolata su 4 principali activity, relative a Login, Registrazione, MainActivity e CorsoActivity e diversi fragment fra cui navigare.

La navigazione all’interno dell’interfaccia è stata realizzata attraverso l’utilizzo di diversi fragment, articolati in 2 diversi livelli. Un primo livello relativo alla visualizzazione dei vari corsi disponibili, legato alla MainActivity, e strutturato in diversi Fragment per la visualizzazione dei contenuti e un Fragment relativo l’area utente; segue la struttura di questo livello.



Nei Fragment relativi la visualizzazione delle liste di corsi sono state utilizzate delle RecyclerView orizzontali, le quali racchiudono degli adapter relativi ad ogni corso, visualizzati in Card. Segue un esempio delle 3 RecyclerView utilizzate nel FragmentHome per visualizzare corsi popolari, consigliati e aggiunti di recente.



Il secondo livello di navigazione è relativo al singolo corso e comprende il FragmentInfoCorso che racchiude tutte le principali informazioni del corso, altri Fragment il cui contenuto è fruibile dall’utente solo se iscritto al corso.



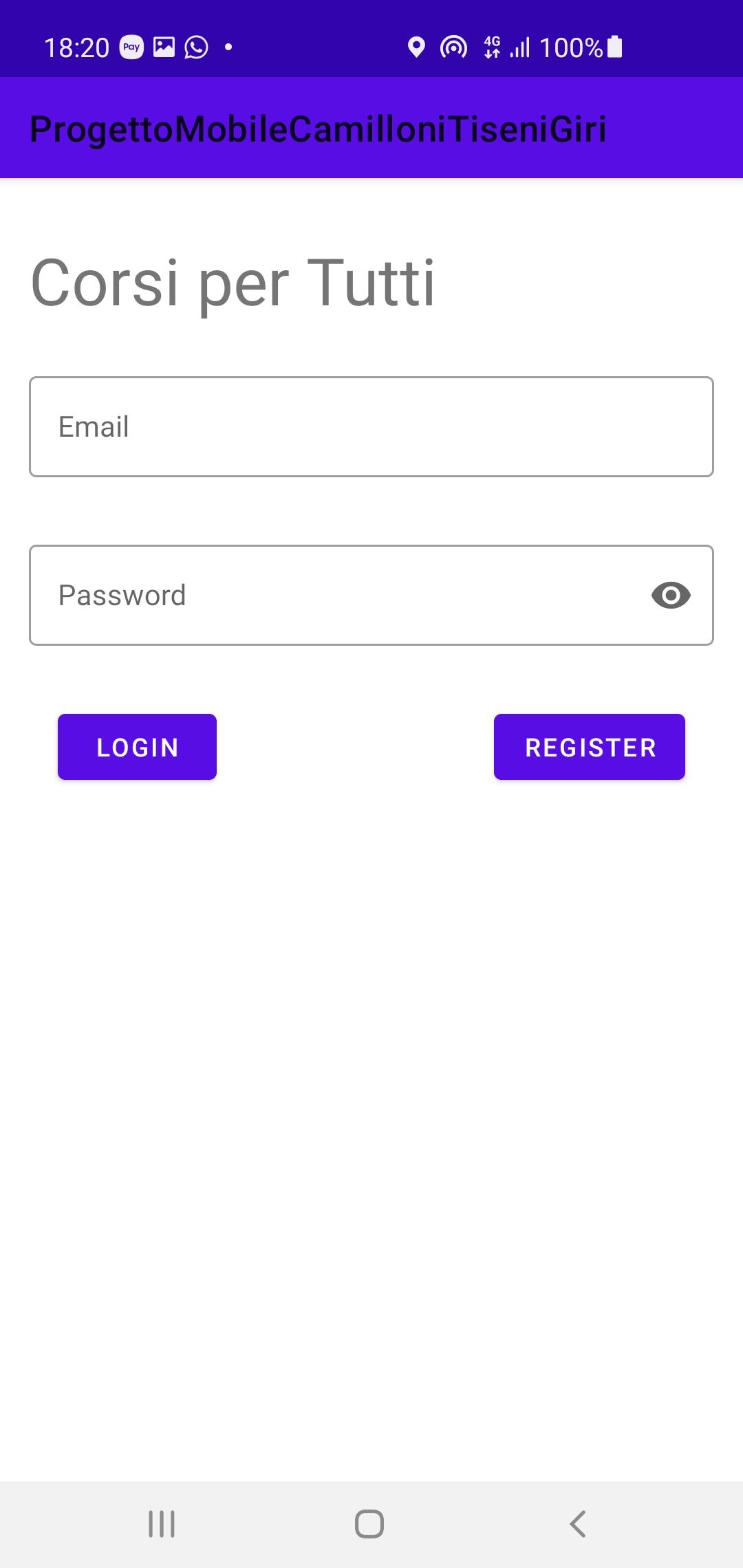
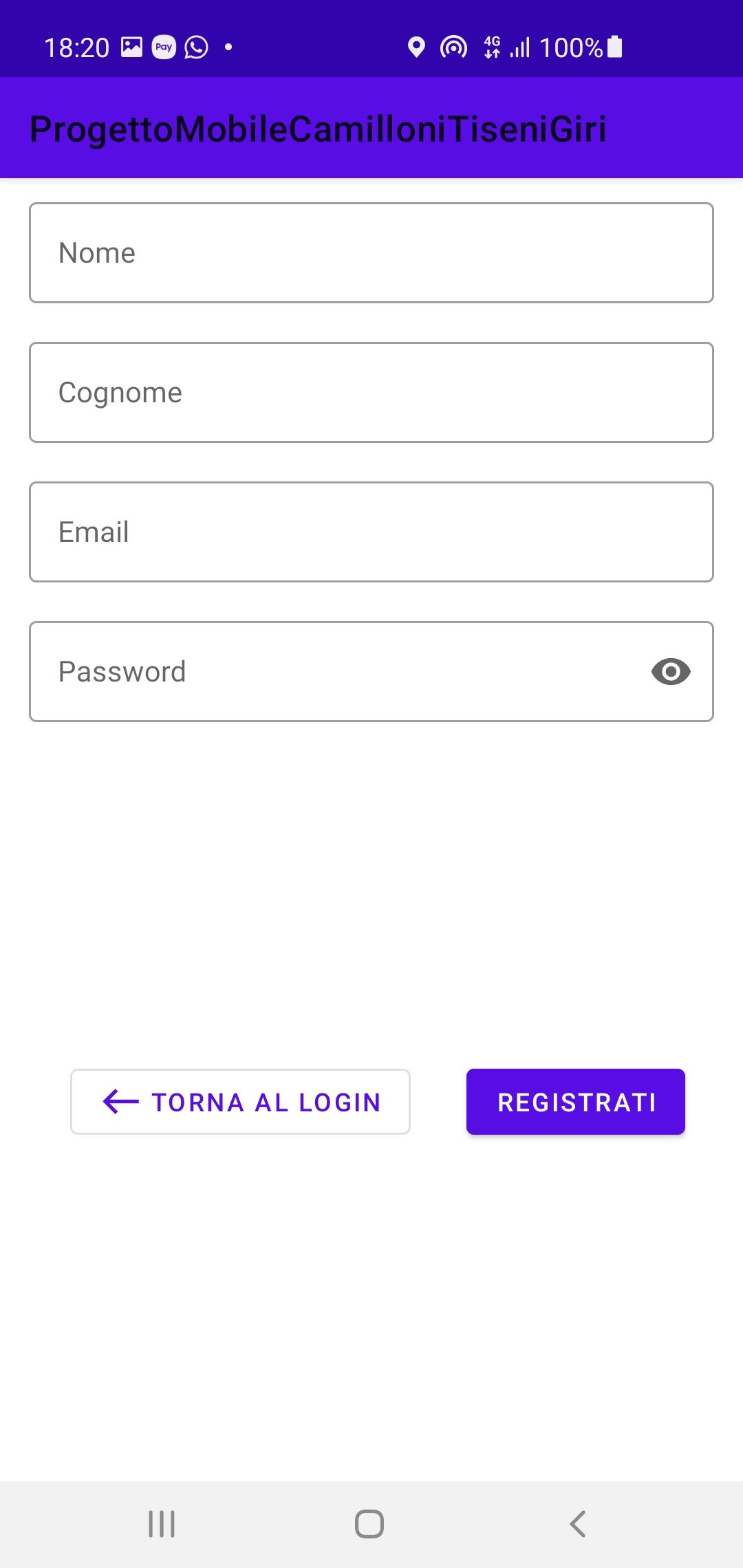
All’interno dei Fragment relativi i contenuti del corso, quali le lezioni, le dispense e il forum, sono state implementate anche qui delle RecyclerView.

Una particolare aspetto del FragmentLezioni è il player di YouTube, il quale non è stato possibile implementare con le API di YouTube per alcuni problemi di compatibilità, ed è quindi stata adottata una libreria resa pubblica da un ingegnere di Google, Pierfrancesco Soffritti, disponibile al seguente [link](https://github.com/PierfrancescoSoffritti/android-youtube-player). Infatti questa libreria ci permette di inserire un player di youtube direttamente in un fragment e non per forza in un activity, cosa che invece non era possibile utilizzando la webView e le normali API di youtube. Questa soluzione ci ha permesso, per prima cosa di riprodurre video YouTube passando semplicemente l’identificativo del video, e secondo, di utilizzare un particolare Tracker per memorizzare il momento in cui la lezione è stata interrotta. Memorizzando l’ultima lezione interrotta siamo riusciti a salvare lo stato di questa lezione per poi poterla riprodurre anche dalla schermata principale in qualsiasi momento, oppure anche dal fragment stesso delle lezioni.



Altro punto di forza per la sezione del corso è stato il Forum, reso possibile dal Real-Time database di Firebase, il quale ha permesso un interazione tra utenti in tempo reale.

Di seguito sono riportate le due activity in cui vengono gestite Login e Registrazione.



## 

## 

## 

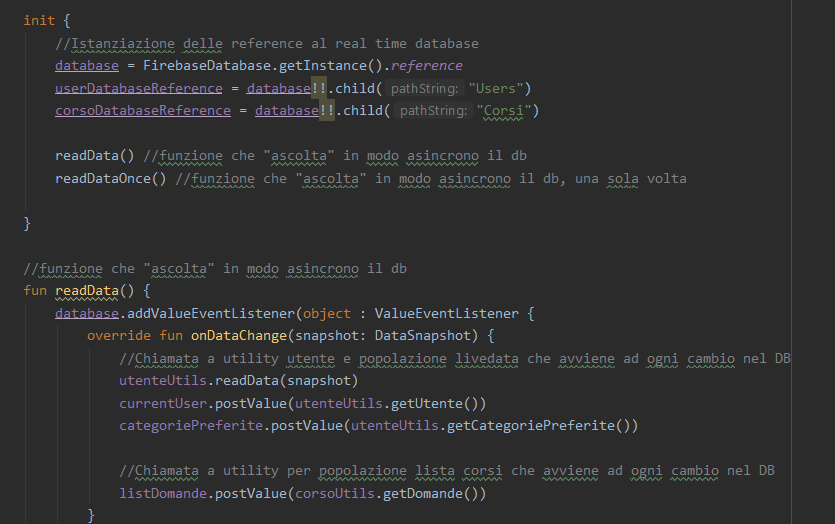
## FirebaseConnection

L’interazione con Firebase avviene interamente all’interno di un viewModel in cui vengono eseguite le seguenti operazioni:

* Istanziazione delle reference al Database Firebase e istanziazione dell’utente corrente, cioè che ha effettuato l’accesso
* Chiamata a 2 funzioni readData() e readDataOnce(), le quali eseguono delle funzionalità di “ascolto” del real-time database e restituiscono un DataSnapshot
* All’interno delle due funzioni sono state utilizzate due utils relative a Corsi e Utente, per la “decodifica” del DataSnapshot restituito
* Letto lo snapshot, sono poi stati popolati dei MutableLiveData utilizzati poi all’interno dell’applicazione per leggere le liste dei corsi e l’utente loggato
* Inoltre sono state definite all’interno di questo viewModels alcune funzionalità per scrivere/aggiornare direttamente alcuni parametri nel database

Le due utils relative al Corso e all’Utente sono state utilizzate solo come appoggio per la lettura del DataSnapshot e la restituzione di alcuni dati.

Segue un esempio dell’utilizzo del metodo addValueEventListener() utilizzato per leggere i dati di un percorso e ascoltare le modifiche sul riferimento al database instanziato.



All’interno del ValueEventListener abbiamo letto e ascoltato tramite il metodo onDataChange() il quale permette di leggere un’istantanea statica dei contenuti nel relativo percorso. Questo metodo, passando al callback dell’evento uno snapshot contenente i dati ha permesso la loro lettura, che è stata effettuate nei relativi metodi readData implementati all’interno delle utils del Corso e dello User.

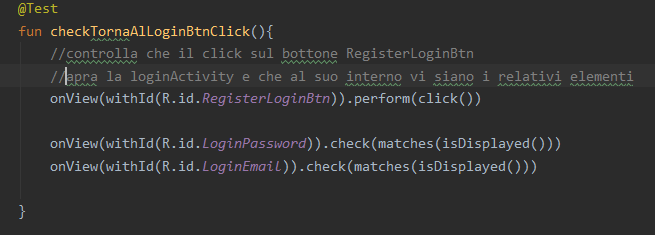
### Test

Sono state testate diverse funzionalità dell’app, ma a causa dell’utilizzo di un viewModel abbiamo avuto problemi nelle fasi di test della FirebaseConnection.

Per quanto riguarda i test dell’UI abbiamo effettuato test sulle activity di Login e di Registrazione, quali:

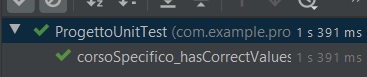
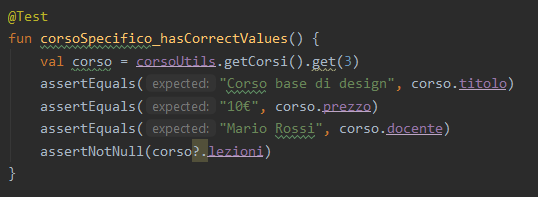
* Test di scrittura in EditText e verifica della correttezza del testo
* Verifiche che le caselle di Login siano vuote al lancio dell’activity
* Verifica di corretto funzionamento dei bottoni, in particolare se il click dei bottoni lanciasse nuove activity

Di seguito viene mostrato un test effettuato nella RegisterActivity per verificare che il click sul bottone di “Torna al Login” lanciasse effettivamente l’activity di Login, ricercando in nuovi elementi mostrati nella nuova activity lanciata.

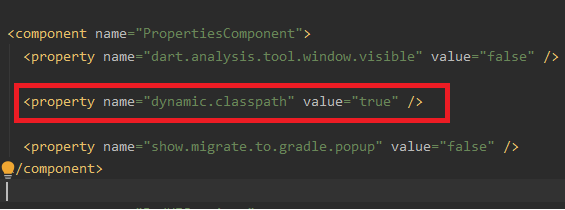


Mentre per quanto concerne lo Unit Testing, sono stati effettuati dei test sulle funzionalità della CorsoUtils e della FirebaseConnection, verificando che ad esempio una funzione restituisca il valore corretto, oppure che il Corso ritornato contenga i valori corretti..

In figura viene mostrato un esempio di Unit Test relativo al controllo del contenuto del corso preso da una lista, e il relativo risultato.



Per far fronte a un problema sorto al momento del lancio degli Unit test, abbiamo dovuto aggiungere la seguente proprietà all’interno del file workspace.xml nella cartella .idea; il messaggio di errore mostrato, era il seguente: *Command line is too long. shorten command line for...*



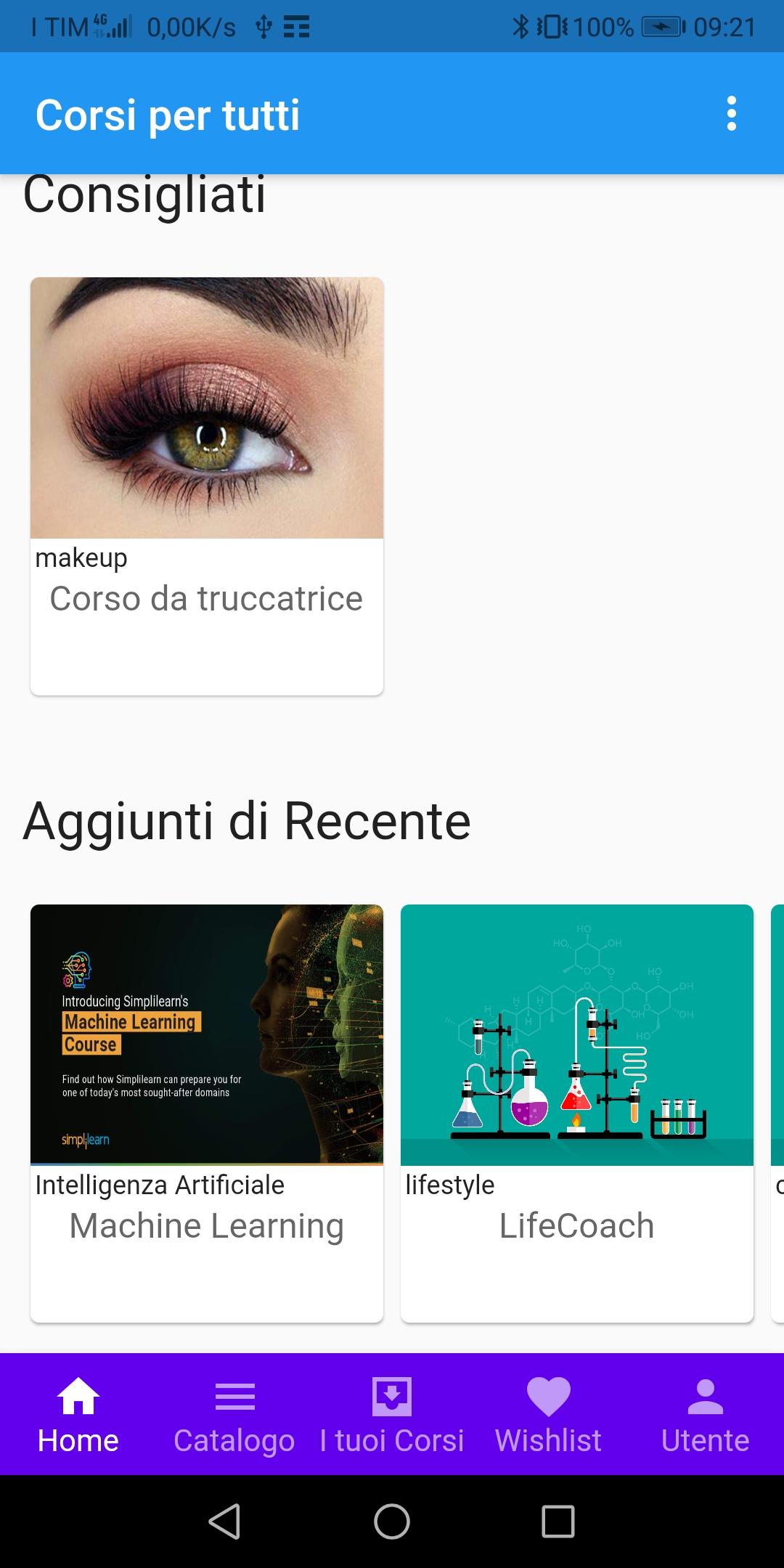
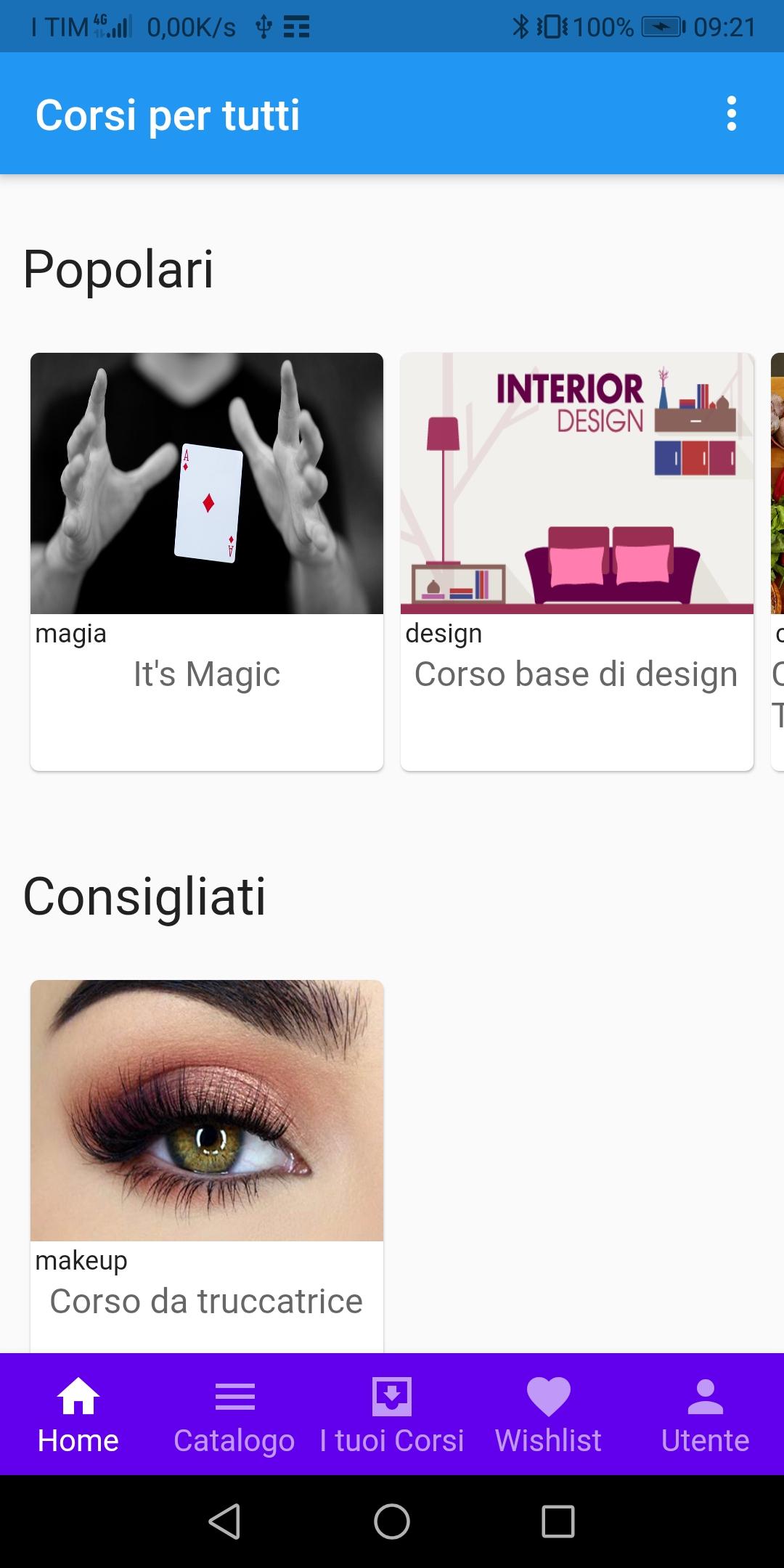
### Flutter

Nell’applicazione Flutter abbiamo riprodotto tutte le principali funzionalità introdotte in Android. Abbiamo mantenuto le funzionalità di Autenticazione dell’utente tramite Firebase, e la possibilità di visualizzare contenuti dei corsi, quali video, informazioni e dispense.

Abbiamo mantenuto la pagina principale con le liste di corsi popolari, consigliati e aggiunti di recente, mentre non abbiamo implementato la possibilità da parte dell’utente di riprendere l’ultima lezione.

Sono stati mantenuti inoltre pagina del catalogo e relativa funzionalità di ricerca tramite input o filtro per categoria, la pagina dei corsi a cui è stata sottoscritta una partecipazione dell’utente, la wishlist e l’area utente in cui vi è la possibilità di modificare i propri dati.

Nella pagina del corso abbiamo mantenuto la visualizzazione di informazioni di carattere generale, la possibilità dell’utente di iscriversi e di inserire il corso alla wishlist, la fruizione di video lezioni e di documenti. In questo caso abbiamo dato la possibilità a qualsiasi utente di visualizzare i contenuti del corso seppur non iscritto a esso.



Per quanto riguarda l’architettura dell’app, per implementare il concetto di navigazione mediante bottom navigation bar si è pensato di creare un widget contenitore con stato che contiene la bottom navigation bar, e la app bar dell’app come parti fisse. Dopodiché sfruttando il concetto di stato in base al pulsante che viene premuto sulla bottom navigation bar nel body di questo contenitore viene disegnato uno dei cinque widget che sono accessibili dalla bottom navigation bar:

* Widget Home
* Widget Catalogo
* Widget per visualizzare i propri corsi
* Widget per visualizzare la wishlist
* Widget per visualizzare l’area utente

Dopodichè da Home, Catalogo, I tuoi Corsi, e wishlist, al click su una carta associata a un corso si può poi navigare verso un’ altra pagina relativa alle info del corso.

Da catalogo cliccando su una delle action chip visualizzate in alto si può accedere a un widget che mostrava solo i corsi appartenenti alla categoria indicata nella chip. Infine mediante il submit della barra di ricerca viene visualizzato un nuovo widget che contiene i corsi contenenti nel titolo, o nella categoria o nella descrizione la query immessa. Oltre a questo livello di navigazione principale c’è un altro sotto livello di navigazione che è quello riferito alla bottom navigation bar che viene mostrata dopo aver cliccato sulla carta del corso. Infatti a seguito del click sulla card del corso ci si ritrova nella pagina di info del corso dove però si può notare un cambio di UI relativamente alla bottom navigation bar. Anche qui infatti quando si clicca sulla carta del corso il contenitore principale viene sostituito da un contenitore secondario che è associato a una bottom navigation bar diversa con tre possibili opzioni di navigazione:

* Info Corso
* Lezioni
* Dispense

La app bar rimane sostanzialmente uguale a quella precedente solo che in alto a sinistra si aggiunge un’icona che permette di tornare indietro al widget in cui precedentemente era stata cliccata la card del corso.

Le principali funzionalità di connessione a Firebase sono state implementate con lo stesso principio utilizzato nella versione Android, utilizzando dei pacchetti resi disponibili dal Framework.

Ciò che concerne l’interazione con il real-time database è stato elaborato all'interno della classe database\_utils, nella quale viene istanziato il riferimento al database e tramite l’iterazione con lo snapshot restituito vengono popolati i contenuti, cioè i corsi, e l’utente loggato; tramite opportuni metodi sono stati realizzate funzionalità per aggiornare dati nel database. L’interazione con il real time database però, è possibile solo in android in quanto nessuno dei componenti del gruppo aveva un PC macOS. Infatti non potendo utilizzare xCode non abbiamo potuto implementare su iOS la connessione con firebase.

L’autenticazione è stata gestita nella classe authentication\_service, la quale istanzia sempre il riferimento al database per collegarsi alla tabella utente, e istanzia inoltre la FirebaseAuth per fruire delle funzionalità di signIn e signUp.

Altra funzionalità principale dell’app in flutter è la visualizzazione dei video di youtube per la fruizione delle lezioni in streaming. In flutter la visualizzazione dei video di youtube è stata notevolmente semplificata dall ‘utilizzo della classe YoutubePlayerIFrame che ci ha permesso facilmente di riprodurre i video di youtube semplicemente passando ad essa l’ID del video di youtube. Inoltre tale classe ci ha permesso di personalizzare il player usato per la riproduzione dei video (ad esempio il pulsante per lo schermo intero è nascosto).



**Bugs e segnalazioni**

In base ai test effettuati sull’app non sono stati riscontrati gravi bugs o problemi che potrebbero ostacolare la normale esperienza dell’utente. Ci sembra comunque giusto segnalare alcuni aspetti che possono sembrare anormali:

* Nell’app android, nel forum di un corso, dopo aver scritto una risposta ad una domanda, il contenitore della domanda viene chiuso automaticamente. Questo accade perché la recycler view percepisce il cambiamento e si aggiorna automaticamente, tornando allo stato di visualizzazione iniziale. Il forum continua comunque a funzionare correttamente, infatti riaprendo la stessa domanda si può vedere la risposta appena inviata.
* Sia nell’app android che nell’app flutter, se si torna indietro (tramite pulsanti di indietro del telefono o dell’app) su alcune schermate, eventuali aggiornamenti non vengono visualizzati subito, ma bisogna ricaricare la sezione o andare in un’altra e tornarci. Un esempio su android è quando dalla homepage si clicca su un corso e si visualizza una lezione. Tornando indietro alla homepage la card dell’ultima lezione non viene subito aggiornata con quella appena vista, ma bisogna uscire dalla homepage e rientrare, o cliccare sulla sua icona nella bottom navigation bar.
* Soltanto alcune volte, se si fa partire l’applicazione, nella homepage il video dell’ultima lezione non riesce a caricarsi, e bisogna aggiornare la sezione o andare su un’altra e rientrarci per far caricare correttamente la lezione. Questo potrebbe essere dovuto ad alcune incompatibilità tra firebase e la libreria di Pierfrancesco Soffritti per lo YoutubePlayer, il quale per qualche motivo non riesce a caricare il video se magari il suo layout viene caricato prima che l’id del video gli venga iniettato(?).

Autori del progetto:

* Camilloni Andrea, 1090325
* Tiseni Lorenzo, 1087649
* Giri Matteo, 1090263

Link ai repository di github:

* Android: <https://github.com/S1087649/ProgettoMobileCamilloniTiseniGiri>
* Flutter: <https://github.com/AndreaCamilloni/progetto_camilloni_tiseni_giri>