**Documentație pentru**

**ToDoApp**

**Introducere**

ToDoApp este o aplicație de gestionare a sarcinilor dezvoltată în Kotlin pentru platforma Android. Aplicația permite utilizatorilor să creeze, să vizualizeze și să gestioneze sarcini zilnice. Utilizează Firebase pentru autentificare și stocare a datelor, oferind o experiență de utilizare intuitivă și eficientă. Datorită implementării SharedPreferences, aplicația păstrează utilizatorii autentificați chiar și după închiderea acesteia. Interfața este prietenoasă și adaptată pentru diferite dimensiuni de ecran, iar utilizarea componentelor moderne precum RecyclerView și Navigation Component asigură o navigare fluidă și o gestionare eficientă a datelor.

A black background with a black square

Description automatically generated with medium confidence**1. Acces la GIT**

Proiectul este disponibil pe GIT și poate fi accesat prin următorul link: <https://github.com/Vlazzzz/ToDoApp-ANDROID>

**2. Activity, Fragment & Navigation Component**

Aplicația ToDo utilizează componentele Android pentru a gestiona tranzacțiile între activități și fragmente.

* **Activitate Principală (MainActivity)**: Aceasta este activitatea principală care conține un NavHostFragment. Acest fragment special acționează ca un container pentru fragmentele din aplicație și gestionează navigarea între ele utilizând NavController.
* **Navigație între Fragmente**: Navigația este configurată folosind componenta de navigație din Android Jetpack. Aceasta permite definirea unui grafic de navigație (nav\_graph.xml) în care sunt specificate toate fragmentele și tranzacțiile între ele.
* **Fragmente**:
  + **SignInFragment:** Acesta este fragmentul în care utilizatorii se autentifică. Conține un formular pentru email și parolă și un buton pentru navigare către SignUpFragment sau pentru a se autentifica.
  + **SignUpFragment**: Aici utilizatorii se pot înregistra. Formularul conține câmpuri pentru email, parolă și confirmarea parolei.
  + **HomeFragment**: Fragmentul principal al aplicației unde sunt afișate sarcinile utilizatorului. De aici, utilizatorii pot adăuga, edita sau șterge sarcini.
  + **SplashFragment**: Primul contact al utilizatorului cu aplicația. Acesta apare doar în momentul deschiderii aplicației.
* **NavController**: Acesta este utilizat pentru a naviga între fragmente. Fiecare fragment utilizează NavController pentru a declanșa navigația, asigurându-se că stiva de back este gestionată corect.

Screens screenshot of a sign up form

Description automatically generated

* 1. **NavGraph**

**3. Autentificare (Login + Register)**

Autentificarea utilizatorilor este realizată folosind Firebase Authentication.

* **Înregistrare (SignUpFragment)**: Utilizatorii își introduc email-ul și parola pentru a crea un nou cont. Datele sunt validate înainte de a fi trimise la Firebase. Dacă înregistrarea are succes, utilizatorul este redirecționat către HomeFragment.
* **Autentificare (SignInFragment)**: Utilizatorii introduc email-ul și parola pentru a se autentifica. Dacă autentificarea reușește, utilizatorul este redirecționat către HomeFragment.
* **Persistența Autentificării**: Firebase gestionează automat persistența autentificării, ceea ce înseamnă că utilizatorii rămân autentificați chiar și după ce închid și redeschid aplicația.

A screenshot of a login form

Description automatically generatedA screenshot of a black screen

Description automatically generated

**3.1 SignIn Fragment 3.2 SignUp Fragment**

**4. Stocarea datelor în baza de date și afișarea acestora în RecyclerView**

Aplicația utilizează Firebase Realtime Database pentru a stoca și prelua sarcinile utilizatorilor.

* **Firebase Realtime Database**: Fiecare utilizator are propriul său nod în baza de date unde sunt stocate sarcinile sale. Structura bazei de date este organizată astfel încât fiecare sarcină să aibă un ID unic și un text descriptiv.
* **RecyclerView și Adapter Personalizat**: Sarcinile sunt afișate într-un RecyclerView folosind un adapter personalizat (ToDoAdapter). Acest adapter este responsabil pentru crearea și legarea elementelor de vizualizare pentru fiecare sarcină din listă.
* **Evenimente de Actualizare**: Aplicația utilizează ValueEventListener pentru a asculta modificările în baza de date Firebase și pentru a actualiza RecyclerView în consecință.

A screenshot of a computer

Description automatically generatedA screenshot of a phone

Description automatically generated **4.1 Interfață Firebase**

A white rectangular box with black text

Description automatically generated

**4.2 New task popup 4.3 Home Fragment with recyclerView**

**5. Date stocate în SharedPreferences**

SharedPreferences este utilizat pentru a salva informațiile de autentificare ale utilizatorului (email și parolă).

* **Stocare**: La autentificare sau înregistrare, email-ul și parola sunt salvate în SharedPreferences. Acest lucru permite aplicatiei să pre-completeze câmpurile de autentificare la deschiderea aplicației.
* **Retragere**: La deschiderea aplicației, dacă există date salvate în SharedPreferences, acestea sunt utilizate pentru a autentifica automat utilizatorul sau pentru a pre-completa formularul de autentificare.

A screenshot of a login form

Description automatically generated

**5.1 Păstrarea informațiilor de autentificare de la o sesiune la alta**

**6. Cereri HTTP**

Aplicația utilizează Firebase pentru autentificare și stocare a datelor, care implică cereri HTTP implicite.

* **Autentificare Firebase**: Cererile HTTP sunt utilizate pentru autentificarea utilizatorilor prin intermediul Firebase Authentication.
* **Firebase Realtime Database**: Cererile HTTP sunt utilizate pentru citirea și scrierea datelor în Firebase Realtime Database.

**7. Crearea shape drawables**

Aplicația include mai multe elemente grafice personalizate pentru a îmbunătăți interfața utilizatorului.

* **Shape Drawable**: Un fișier XML care definește un element grafic personalizat, cum ar fi un triunghi transparent.
* **Drawable Selector**: Un selector care definește diferite stări pentru un element grafic, cum ar fi starea apăsată sau inactivă a unui buton.
* **Color Selector**: Un fișier XML care definește diferite culori pentru un element grafic în funcție de starea sa.
* **Gradient**: Un fișier XML care definește un gradient de culori, utilizat pentru fundaluri sau alte elemente grafice.

**8. Aplicație user-friendly**

Aplicația a fost proiectată pentru a fi intuitivă și ușor de utilizat.

* **Design Responsiv**: Elementele UI sunt aranjate astfel încât să nu se suprapună și să fie accesibile pe diferite dimensiuni de ecran.
* **Interfață Intuitivă**: Butoanele și câmpurile de introducere a datelor sunt etichetate clar și poziționate logic pentru a facilita utilizarea aplicației.

În fragmentele de SignIn si SignUp există un singur buton de validare a datelor introduse în casetele de text, în timp ce în Home fragment există doar două butoane, unul de logout și unul de adăugare de noi task-uri.

**Concluzie**

Aplicația ToDo oferă o gestionare eficientă a sarcinilor utilizând Firebase pentru autentificare și stocare a datelor, SharedPreferences pentru păstrarea informațiilor de autentificare și un design prietenos pentru utilizator. Este un exemplu complet de integrare a diferitelor componente Android pentru a crea o aplicație funcțională și estetică.