

Projet : Inteco

Programmation mobile

Mohamad Satea Almallouhi - Tony Nguyen
M1 Génie Logiciel
Faculté des Sciences
Université de Montpellier.

12 mai 2024



Résumé

Durant cet exercice, nous avons vu l'utilisation des Fragments, ModelAndView et MutableLiveData.

Table des matières

| | |
|--|----------|
| Introduction | 2 |
| 1 Backend avec Firebase | 3 |
| 1.1 Sign up - Job Seeker | 3 |
| 1.2 Login | 4 |
| 1.3 Récupération d'un document | 4 |
| 2 Barre de navigation | 5 |
| 3 Conclusion | 5 |

Introduction

Dans le cadre de l'Unité d'Enseignement Programmation Mobile, nous allons vous parler de notre avancé par rapport à l'application d'intérim.

Les fonctionnalités implémentées sont la connection et l'inscription ainsi que la récupération d'un Curriculum Vitae (CV).

Démonstration vidéo

En ligne sur Youtube, à l'adresse URL <https://youtu.be/r30-L7iI1IE> une démonstration vidéo de notre travail.

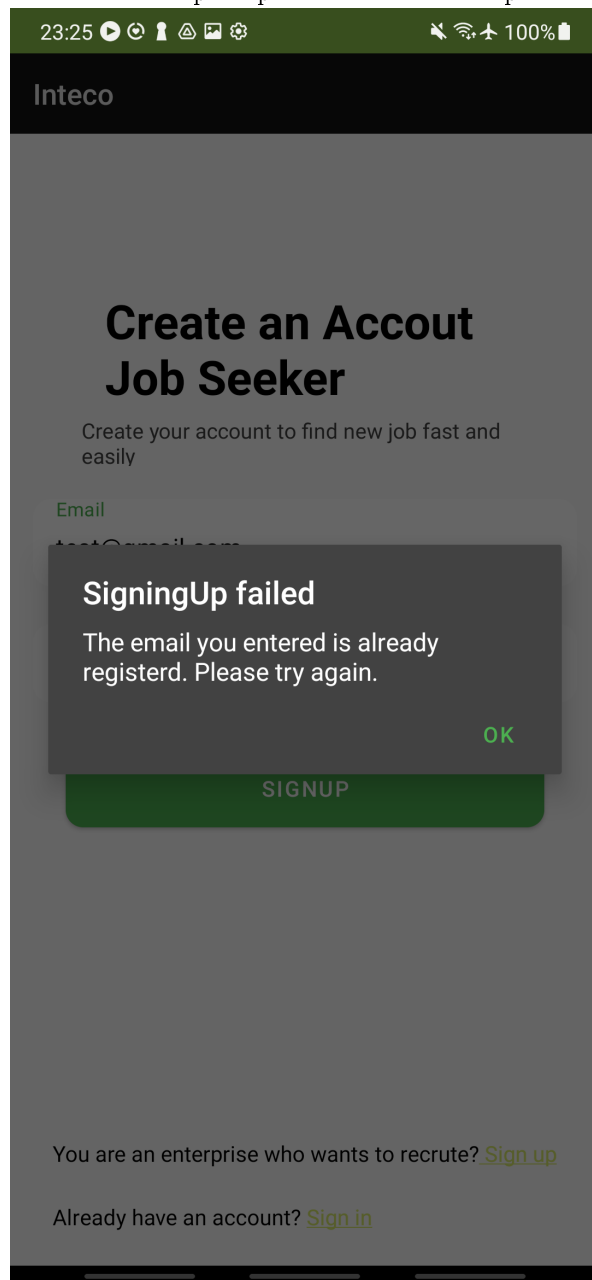
En ligne sur github, notre dépôt à l'adresse suivante : <https://github.com/tony-nguyen1/Inteco/>

1 Backend avec Firebase

1.1 Sign up - Job Seeker

Afin de créer un compte, vous devez entrer vos informations petit à petit sur 3 pages successives.

La 1er page demande une adresse mail et un mot de passe. Nous considérons les adresses mail comme unique. C'est pourquoi si l'adresse est déjà dans la base de données, l'utilisateur ne peut poursuivre son inscription.



```
private boolean verifyEmailExists(String email, MutableLiveData<Boolean> response){
    //TODO complete ( true if it exists and false if not)
    Log.d(TAG, msg: "verifyEmailExists: "+email);
    //response.setValue(Boolean.FALSE); //Intililize with a value

    // read
    Task<QuerySnapshot> q = db.collection( collectionPath: "users" ) CollectionReference
    .get() Task<QuerySnapshot>
    Tony Nguyen
    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
        Tony Nguyen
        @Override
        public void onComplete(@NonNull Task<QuerySnapshot> task) {
            if (task.isSuccessful()) {
                boolean foundEmail = false;
                for (QueryDocumentSnapshot document : task.getResult()) {
                    //Log.d(TAG, document.getId() + " => " + document.getData());
                    String s = document.get( field: "email", String.class);
                    //Log.d(TAG, "onComplete: "+s);
                    //Log.d(TAG, "onComplete: "+email);

                    assert s != null;
                    if (email.equals(s)){
                        response.postValue(Boolean.TRUE); // post a value
                        foundEmail = true;
                    }
                }
                //Log.d(TAG, "onComplete: foundEmail="+foundEmail);
                if (!foundEmail) {
                    //Log.d(TAG, "onComplete: inside if"+foundEmail);
                    response.postValue(Boolean.FALSE);
                } else {
                    Log.w(TAG, msg: "Error getting documents.", task.getException());
                }
            }
        }
    });
}
```

À la dernière page, en appuyant sur le bouton "SIGN UP", le compte est créé. On crée un nouveau utilisateur avec `createUserWithEmailAndPassword()`, une fonction qui communique avec le service Authentification de Firebase. Puis, une fois fait, on enregistre les informations de l'intérimaire dans la BD Firestore.

```
FirebaseFirestore db = FirebaseFirestore.getInstance();

// Create a new user with a first and last name
Map<String, Object> user = new HashMap<>();
user.put("email", email);
user.put("password", password);
user.put("firstname", firstName);
user.put("lastname", lastName);
user.put("birthday", birthday);
user.put("nationality", nationality);
user.put("phoneNumber", phoneNumber);
user.put("city", city);
user.put("address", address);
user.put("sex", sex);

db.collection( collectionPath: "users" ) CollectionReference
.add(user) Task<DocumentReference>
Tony Nguyen
.addOnSuccessListener(new OnSuccessListener<DocumentReference>() { Tony Nguyen
    ...
})
.addOnFailureListener(new OnFailureListener() { Tony Nguyen
    ...
});

mAuth.createUserWithEmailAndPassword(email, password)
Tony Nguyen
.addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
    Tony Nguyen
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        if (task.isSuccessful()) {
            // Sign in success, update UI with the signed-in user's information
            Log.d(TAG, msg: "createUserWithEmail:success");
            FirebaseUser user = mAuth.getCurrentUser();
            updateUser(user);
            Log.d(TAG, msg: "onComplete: "+user.toString());
        } else {
            // If sign in fails, display a message to the user.
            Log.w(TAG, msg: "createUserWithEmail:failure", task.getException());
            Toast.makeText( context: SignUpSeeker3.this, text: "Authentication failed.",
                Toast.LENGTH_SHORT).show();
            updateUser(null);
        }
    }
});
```

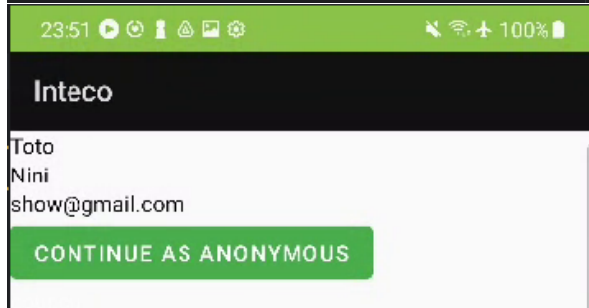
1.2 Login

Pour réaliser la connexion, nous avons fait un simple appel au service d'Authentification fourni par Google Firebase.

```
 mAuth.signInWithEmailAndPassword(email, password)
    & tony-nguyen1
    .addOnCompleteListener( activity, this, new OnCompleteListener<AuthResult>() {
        & tony-nguyen1
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                // Sign in success, update UI with the signed-in user's information
                Log.d(TAG, msg: "signInWithEmail:success");
                FirebaseUser user = mAuth.getCurrentUser();
                updateUser(user);
                responseUser.postValue(user);
            } else {
                // If sign in fails, display a message to the user.
                Log.w(TAG, msg: "signInWithEmail:failure", task.getException());
                Toast.makeText(context, LoginFirstActivity.this, text: "Authentication failed.",
                    Toast.LENGTH_SHORT).show();
                updateUser(null);
                response.postValue(null);
            }
        }
    });
```

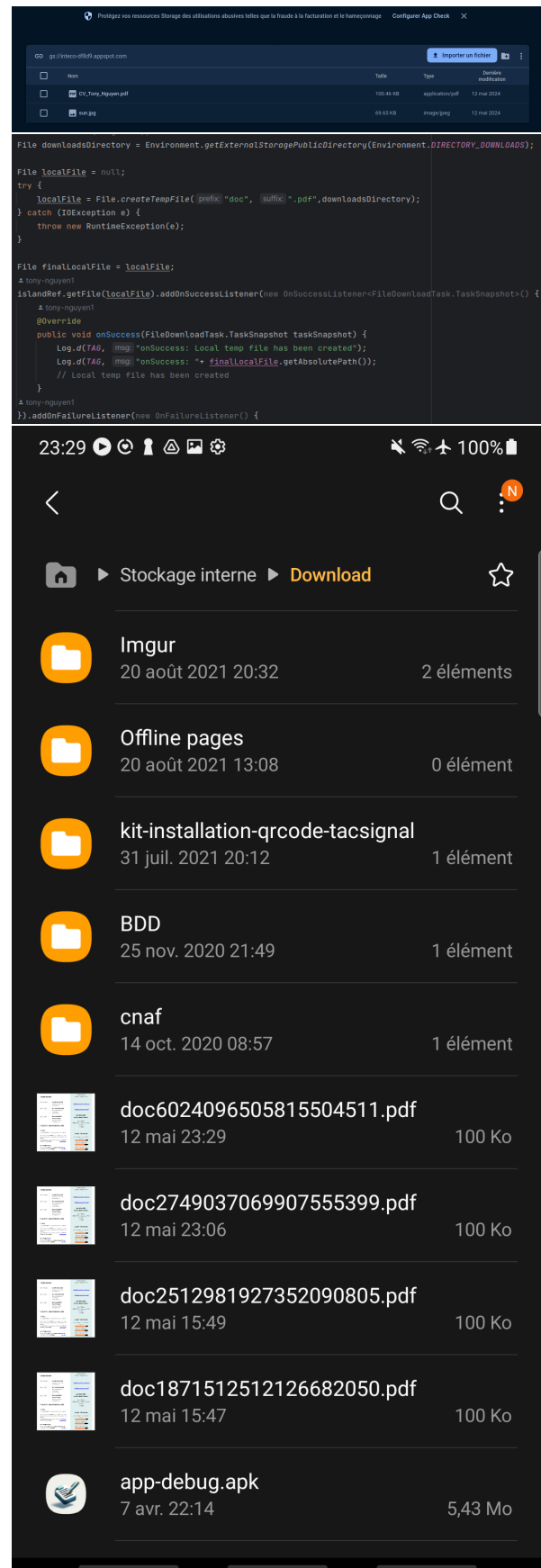
Ensuite, nous faisons un appel à la base de données Firestore pour récupérer les informations de ce compte.

```
 responseUser.observe( owner: LoginFirstActivity.this, new Observer<FirebaseUser>() {
    & tony-nguyen1
    @Override
    public void onChanged(FirebaseUser firebaseUser) {
        db.collection( collectionPath: "users", CollectionReference
        .get() Task<QuerySnapshot>
        & tony-nguyen1
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            & tony-nguyen1
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    boolean found = false;
                    QueryDocumentSnapshot infoUser = null;
                    for (QueryDocumentSnapshot document : task.getResult()) {
                        String s = document.get( field: "email", String.class);
                        if (firebaseUser.getEmail().equals(s)) {
                            found = true;
                            infoUser = document;
                        }
                    }
                    Log.d(TAG, document.getId() + " => " + document.getData());
                    responseInfo.postValue(infoUser.getData());
                } else {
                    Log.w(TAG, msg: "Error getting documents.", task.getException());
                }
            }
        });
    }
});
```



1.3 Récupération d'un document

En ce qui concerne la récupération de document comme les CV et lettre de motivation, nous utilisons Firebase Storage. Il nous suffit de faire un appel à la fonction getFile() pour télécharger notre document.

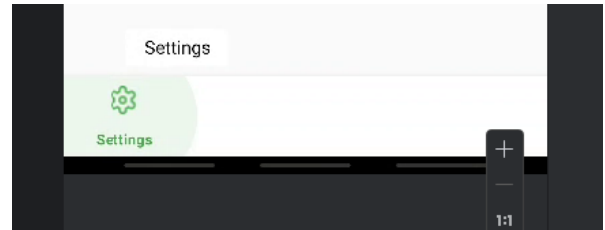


2 Barre de navigation

Pour réaliser, la barre de navigation, nous avons utilisé des balises déjà existantes.

```
<com.google.android.material.bottomnavigation.BottomNavigationView
    android:background="#FFFFFF"
    android:id="@+id/bottom_navigation"
    android:layout_width="match_parent"
    android:layout_height="180px"
    android:layout_alignParentBottom="true"
    app:menu="@menu/bottom_nav_menu_seeker" />
```

```
1 <!-- res/menu/bottom_nav_menu.xml -->
2 <menu xmlns:android="http://schemas.android.com/apk/res/android">
3
4     <item
5         android:id="@+id/nav_settings"
6         android:icon="@drawable/icon_setting"
7         android:title="Settings" />
8
9     <item
10        android:id="@+id/nav_home"
11        android:icon="@drawable/icon_home"
12        android:title="Home" />
13
14     <item
15        android:id="@+id/nav_notifications"
16        android:icon="@drawable/icon_bell"
17        android:title="Notifications" />
18
19     <item
20        android:id="@+id/nav_saved"
21        android:icon="@drawable/icon_save"
22        android:title="Saved" />
23
24 </menu>
```



3 Conclusion