II.3510 – Mobile Application Development for Android

ISEP/Android Group 6 - News App

Student ID	Student Name
61571	WANG Ruiqi
61548	XU Yuan

Institut supérieur d'électronique de Paris

IEMDP

January 2022

TABLE OF CONTENTS

(The url to our git repository:https://github.com/wrq77/Android-App_NewsApp)

- Project features

Our project, News app, allows users to browse through news headlines for different categories such as Business, Technology, Entertainment, and so on. Users can choose the category they are interested in, and bookmark their favorite news.

The basic functions are as follows:

1. Splash Screen

The splash screen corresponds to the introduction page of our application, which can be seen when the application is opened, where you can find information such as our logo and the publishers' names. It will jump to the login interface after 5 seconds or users can also click the "SKIP" button to skip this page.

2. Registration

In the registration interface, users will be asked to enter or select the following information (all items are required):

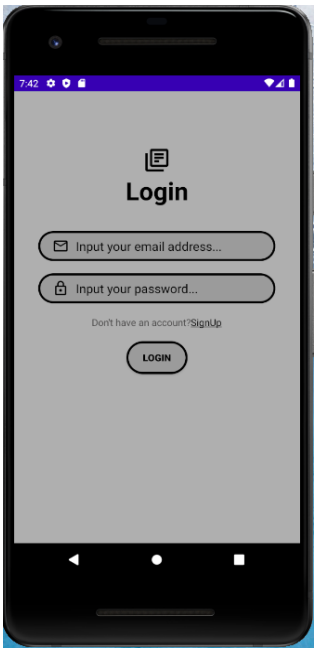
- E-mail: Enter the user's email address.
- Username: Enter the user's name.
- Password: The password must be more than 6 characters.
- Confirm Password: As a security check, users will be asked to retype their password.The password must be the same before and after.

- Sign up: Press the “SIGN UP” button, submit the data and jump to our main dashboard interface.
- If the user already has an account, he/she can click “Already have an account?” below the “SIGN UP” button to jump to the login.

3. Login

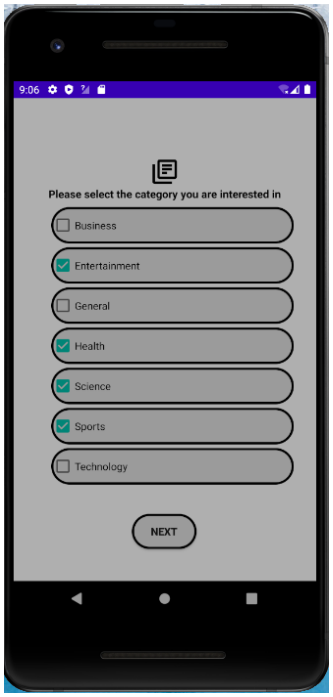
In the login interface, users will be asked to enter or select the Email and Password information:

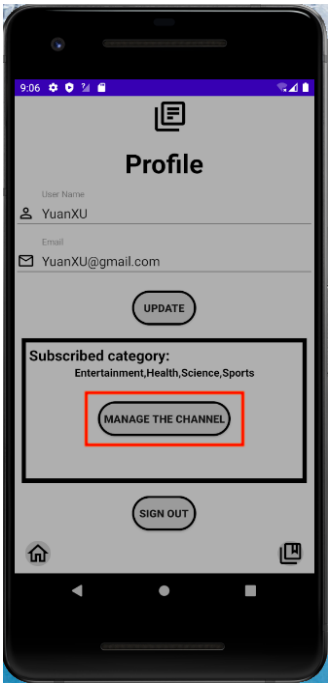
- Login: press the “LOGIN” button, submit the data and jump to our main dashboard interface.
- If the user hasn’t got an account, he/she can click “Sign up” above the “LOGIN” button to jump to the registration.



4. Select/Manage the Category

The category selection page lets users click the box left to the news categories based on their interest, including Business, Entertainment, General, Health, Science, Sports, Technology. There are two ways to reach the category selection page: second step of registration; click the “MANAGE THE CHANNEL” button on the profile page to reselect.

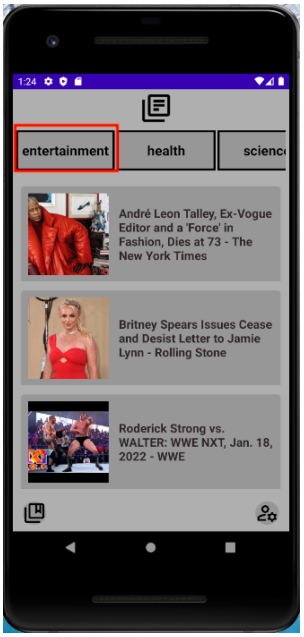


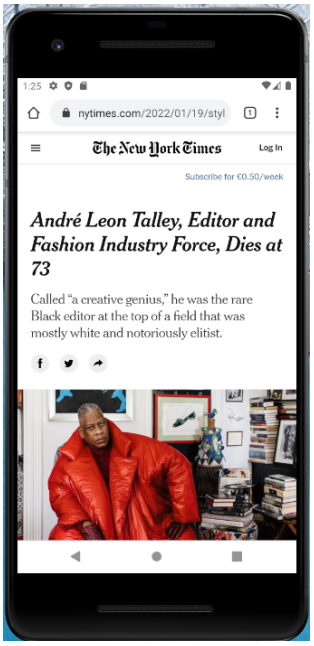


5. Show News Detail

On the main dashboard page, there is a row of category buttons (that users already chose) at the top. We use the card to display news, and click on the card will jump to the news details page.

- Category button: Users can click on the corresponding category button to view the news of that category.
- News card: Displays news pictures, news title. After clicking the news card, jump to the news details page.
- Read More: Click the “READ MORE ABOUT THIS NEWS…” button will redirect to the browser to get the news source to know more details about the news.

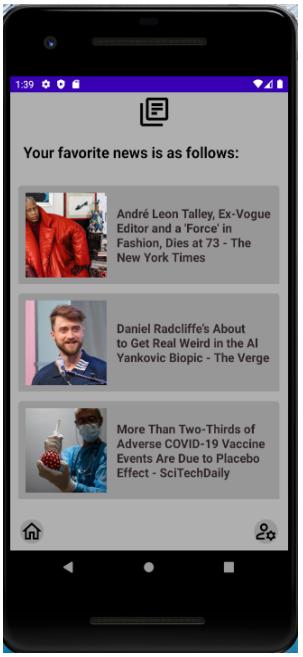




6. Collect the News/unfavorite the News

Users can favorite news, unfavorite news, and view a list of all favorite news.

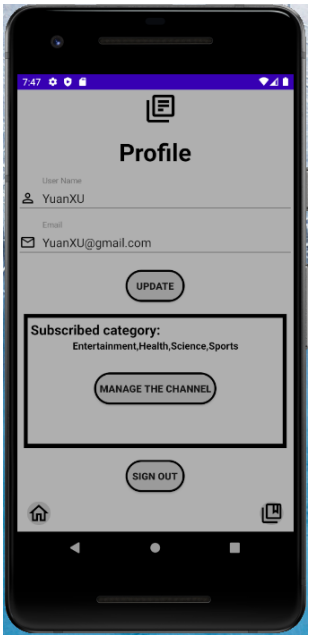
- Heart icon: Users can click this icon to bookmark the news, and the icon will turn black after the bookmark; Users can click the icon again to cancel the favorite news, and the icon will turn to the original state.
- Collection icon: Users can click this icon to view the list of their favorite news.



7. Profile Update

The profile page displays the user's personal information, news categories information. And the user can change his/her username and email.

- Update: fill in the Username/Email the user wants to modify and click the “UPDATE” button to update the user's information.
- Manage the Channel: press the “MANAGE THE CHANNEL” button will redirect to the category selection page to manage their interested channels.
- Sign out: users can click “SIGN OUT” to log out his/her account and redirect to the login interface

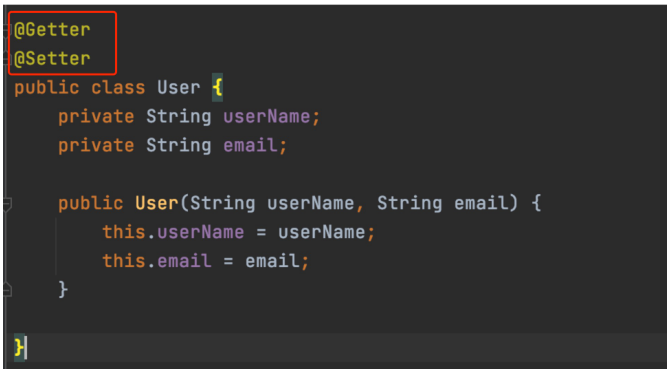


- Used technology

The technology we used is as follows:

1. Lombok

We applied Lombok to add the annotation(@Getter @Setter) on the class to automatically generate getters and setters for the objects in the Model to make the code more concise.



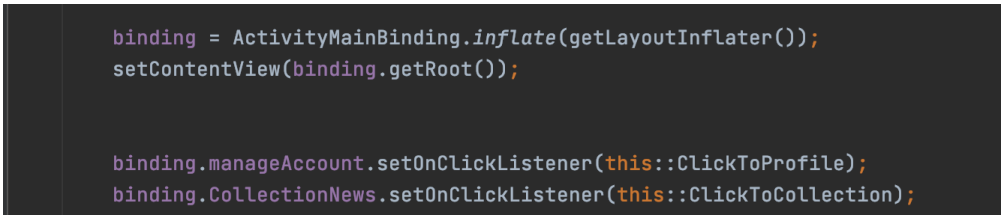
```
@Getter
@Setter
public class User {
    private String userName;
    private String email;

    public User(String userName, String email) {
        this.userName = userName;
        this.email = email;
    }

}
```

2. View binding

We chose View binding to interact with views. It can reduce repetitive code like findViewById() and guarantee the Null safety and Type safety .

After View binding is used, for instance, the activity_main.xml file, an ActivityMainBinding.java file will be automatically generated, then we can directly use it in the Activity and call the getRoot() method to provide a direct reference for the root view of the corresponding layout file.

```
private ActivityMainBinding binding;
```



```
binding = ActivityMainBinding.inflate(getLayoutInflater());
setContentView(binding.getRoot());


binding.manageAccount.setOnClickListener(this::ClickToProfile);
binding.CollectionNews.setOnClickListener(this::ClickToCollection);
```

3. Recyclerview

We used Recyclerview to show the news categories and the list of news.It can display large amounts of data efficiently.

Recyclerview contains the ViewGroup of the view corresponding to the data. Each individual element in the list is defined by a ViewHolder object, and RecyclerView can request these views and bind the views to its data by calling methods in the Adapter

4. Handler, Thread

For the Welcome Activity, the splash screen, we used the handler mechanism and opened a thread to realize the page jump after displaying the welcome page for 5s. We called the Thread.sleep() method to set the 5 seconds.

```
private void run() {
    try {
        Thread.sleep(SPLASH_TIME);
        message.what = 1;
        handler.sendMessage(message);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
```

A Handler allows us to send and process Message objects associated with a thread's MessageQueue to control whether to wait 5 seconds or skip this page directly. In the second case, we called the interrupt() method in Thread to stop it.

```
handler = new Handler(Looper.getMainLooper()) {
    @Override
    public void handleMessage(Message message) {
        super.handleMessage(message);
        if (message.what == 1) {
            Intent intent = new Intent(getApplicationContext(), LoginActivity.class);
            startActivity(intent);
            finish();
        } else if (message.what == 0) {
            thread.interrupt();
        }
    }
};
```

5. Picasso

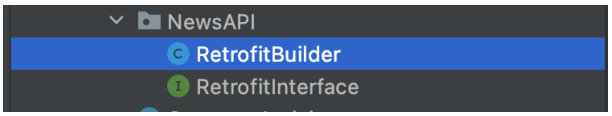
For image loading in our application, we use Picasso.

Picasso can handle ImageView recycling and download cancellation in the adapter; perform complex image transformations with minimal memory; and automatic memory and disk caching.

```
Picasso.get().load(CurrentArticle.getUrlToImage()).into(holder.Image);
```

6. Retrofit

We used Retrofit to interact with APIs and send network requests. Retrofit can turn the HTTP API into a Java interface.




```
public interface RetrofitInterface {
    @GET("top-headlines?country=us")
    Call<News> getNewsByCategory(@Query("category") String category, @Query("apiKey") String apiKey);
}
```

And in the RetrofitBuilder class generates an implementation of the GitHubService interface, and we used Gson for its deserialization.

```
public class RetrofitBuilder {
    private static Retrofit retrofit;
    public static Retrofit getRetrofitInstance(){
        if(retrofit == null){
            retrofit = new Retrofit.Builder()
                    .baseUrl("https://newsapi.org/v2/")
                    .addConverterFactory(GsonConverterFactory.create())
                    .build();
        }
        return retrofit;
    }
}
```

And we also used Call from the created RetrofitInterface to make an asynchronous HTTP request to the remote web server

```
RetrofitInterface retrofitInterface = RetrofitBuilder.getRetrofitInstance().create(RetrofitInterface.class);
Call<News> call = retrofitInterface.getNewsByCategory(CategoryName,apiKey);

//asynchronous
call.enqueue(new Callback<News>() {
    @Override
    public void onResponse(Call<News> call, Response<News> response) {
        News news = response.body();
        ArrayList<Newsdetail> newsdetails = news.getArticles();
```

(For your information, this is our API url: https://newsapi.org/ , and our API Key is 2a75f3dbcae446c4868c3e50e889dab7)

7. Firebase: Authentication, Cloud FireStore

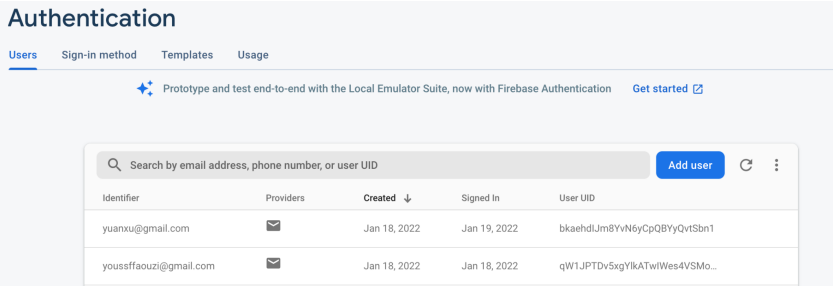
With Firebase Authentication, we can let our users authenticate with Firebase using their email addresses and passwords. For creating a new account, we passed the new user's email address and password to the createUserWithEmailAndPassword method after completing any new account verification steps our app requires.

```
mAuth.createUserWithEmailAndPassword(email,password)
        .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
```

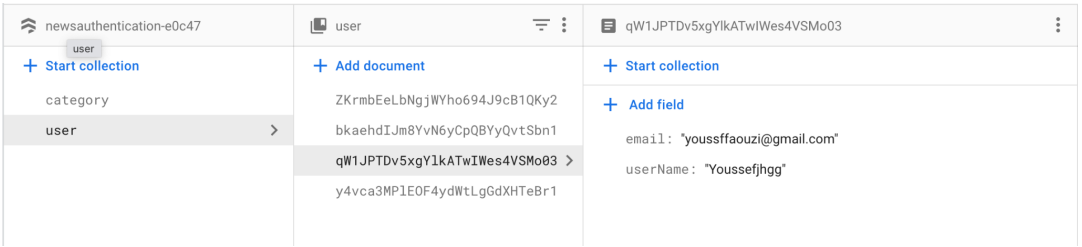
When a user signs in to our app, we pass that user's email address and password to the signInWithEmailAndPassword method.

```
mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
```

After a user logs in for the first time, a new user account is created and associated with the credentials the user used to log in, this new account is stored in our Firebase project.



We applied Cloud FireStore, a NoSQL cloud database to store and synchronize data. It can store our data in documents and organize them into collections. Documents can also contain subcollections.



8. ConnectivityManager

We use getActiveNetwork() to check whether a network interface is available, if it is not available, there will be a message showing up in the dashboard.

```
public boolean isOnline() {
    ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    Network networkInfo = connMgr.getActiveNetwork();
    return (networkInfo != null);
}
```

- Conclusion

Through this practical project, we have integrated the programming knowledge for android development we learned this semester, from UI design to backend implementation. We practiced the operation for the Firebase database. Also, we experienced a collaborative programming environment, which would be helpful to our future career.

The News APP currently has all the basic functions, as we proposed at the beginning of the project. Of course, there are many areas that can be improved on ourapp, such as better UI design, letting users to search related news, adding more channels for the news, implementing user behavior monitoring based on user favorites, optimizing network access and usage and so on.