Ministry of Education, Culture and Research

Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Mobile Development

Telemedicine

Author:
Vasile Drumea

Supervisor:
Antohi Ion

1 Purposes of the laboratory work

- Develop the Telemedicine mobile app based on the given design and requirements;
- Gain knowledge about the work with drawables, layouts and other resources;
- Implement user authentication;
- Use a storage method for the information used in the application;

2 Laboratory Work Requirements

- **Frontend** Using the provided design implement the UI of all the Views bellow:
  - Splash Screen;
  - Welcome Screen;
  - Login/Signup Screen;
  - Home Screen;
  - Approved Request Screen;
  - Doctor List;
  - Doctor Contacts;

  The UI should be adapted for min 3 screen sizes(mdpi, hdpi, xhdpi). Present in emulator or on a real device.
- **Backend** Implement the functional part of the app according to the API structure from below:
  - Authentication - Login;
  - Registration - Register users;
  - Get Profile - Extract profile information;
  - Get Doctors List - View the list of Doctors;
  - Get Doctor info - View information about a Doctor;
  - User Request - Add an appointment for a user;

3 Laboratory work implementation

3.1 Frontend

To create the Frontend of the application i.e. the Views with the UI and the transitions of the views I've followed the steps below:

- First thing I've opened the provided PSD files in GIMP as layers and select the layers that I need in my application. Next I've exported the layers as PNG files.
- To import them in Android Studio I've used Batch Drawable Importer. It automatically created PNGs for 5 different densities i.e. hdpi, mdpi, xhdpi, xxhdpi and xxxhdpi.
- For some parts I needed to encapsulate more images in one object. For this I've created additional drawable file. For example for confirm button I've created a layer-list in the following way :

```
<layer-list xmlns:android="http://schemas.android.com/apk/res/android"
                        android:opacity="opaque">
        <!-- The background -->
        <item android:drawable="@drawable/btn__2" />

        <!-- The text -->
        <item>
                <bitmap android:src="@drawable/confirm"
                        android:gravity="center" />
        </item>
</layer-list>
```

- Also I've define the appropriate themes for the app in styles.xml. I have 3 themes:
  - TelemedicineTheme - With the background green and with the pattern applied;
  - TelemedicineTheme.Launch - For the Splash Screen;
  - TelemedicineThemeMenu - For the views with white background;
- At last I've created the layout file for each view I needed;
- To make the connection between views and the transitions in each activity class I have the corresponding functions in which I create Intents to start another activity:

```
        Intent intent = new Intent(this, ClassName.class);
        startActivity(intent);
```

- The functions are called with the help of Click Listeners provided for each button;

3.2 Backend

To implement the required functionalities I've used an additional tool i.e. Firebase. Firebase is a web platform which provides developers with tools like Authentication, Database, Hosting etc. to ease the development process for different platforms like Android, IOS, Unity etc.

To bind the application with Firebase services, first we need to create a project in the Firebase console. After creation we can access it. There we have a manager UI from which we can manage the the tools which we are using in the app, e.g. at authentication section we have information about the users, we can manually add them and also the console provides us with nice statistics.

In what follows I'll enumerate the functionalities:

– Authentication/Registration - For Login, Registration and User Storage I've used Authentication tool from Firebase. At the start we also must specify the sign in method. We have different methods e.g. email/password, phone, google etc.

  In my classes I'm using a FirebaseAuth(mAuth) and a FirebaseAuth.AuthStateListener(mAuthListen
  First I'm assigning the listener to the mAuth:

```
mAuth.addAuthStateListener(mAuthListner);
```

  Then I need to initialize the Firebase app and get the instance to it:

```
FirebaseApp.initializeApp(this);

// initialize FirebaseAuth instance
mAuth = FirebaseAuth.getInstance();
```

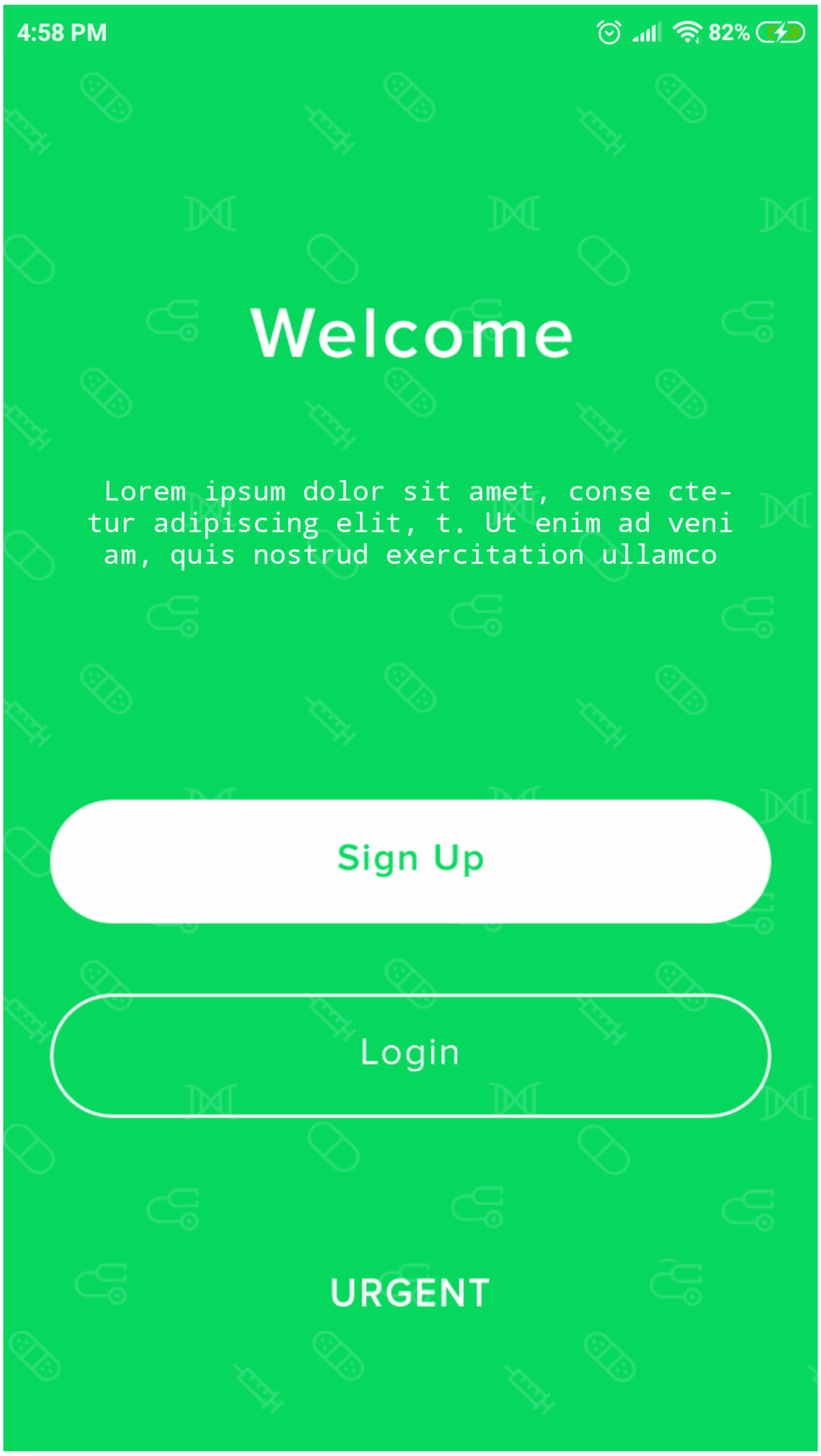
  After this I can use the API provided by FirebaseAuth, for example:

```
// check the current user
if (mAuth.getCurrentUser() != null) {
        startActivity(new Intent(LoginActivity.this,
                                                    HomeAct
        finish();
}
```

  After I have registered some users I can manage them in the Firebase console(See figure 3.5).

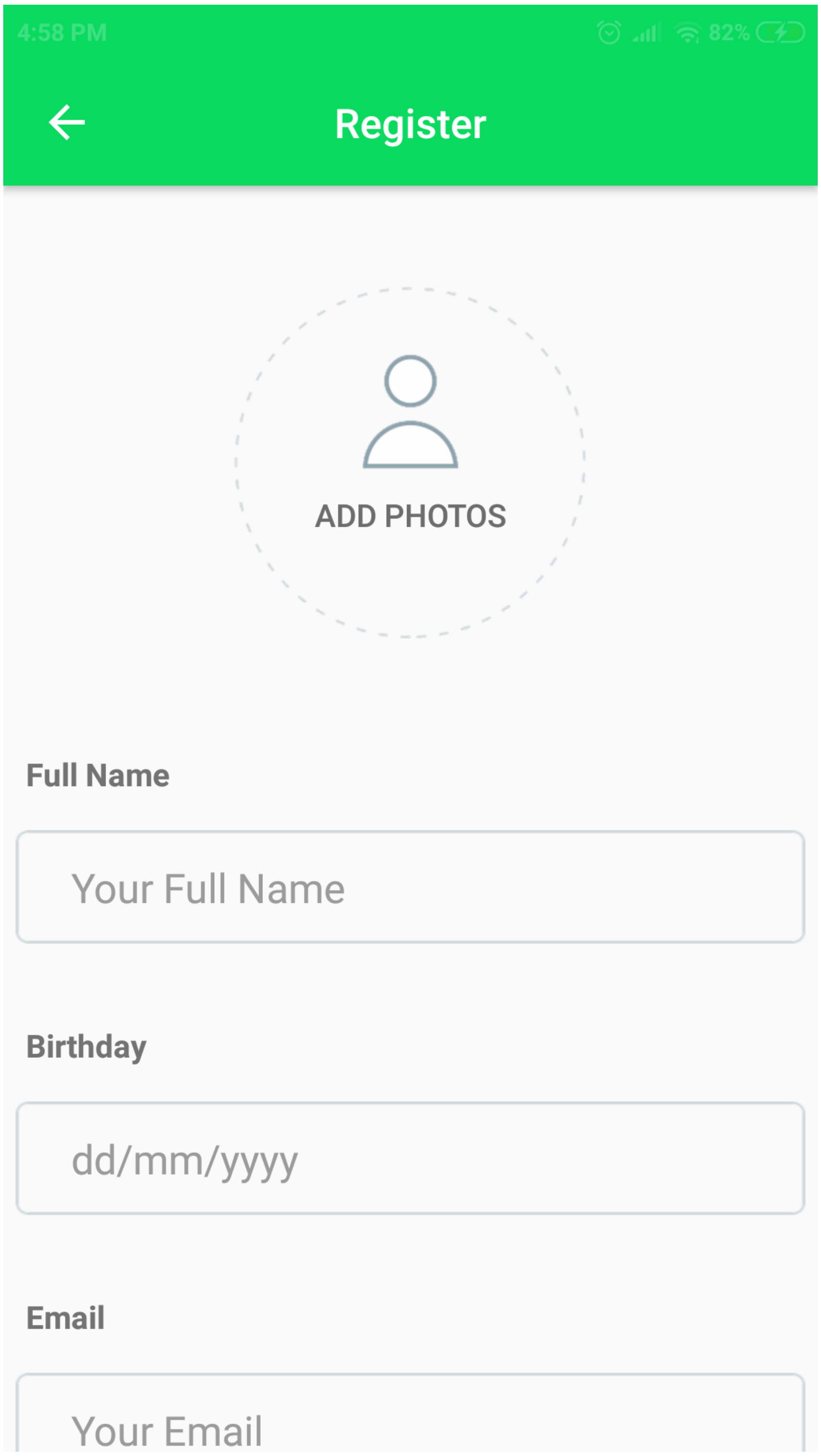
–

3.3 Screenshots



Lorem ipsum dolor sit amet, conse cte-
tur adipiscing elit, t. Ut enim ad veni
am, quis nostrud exercitation ullamco

Figure 3.1 – Main Activity



Full Name

Your Full Name

Birthday

dd/mm/yyyy

Email

Your Email

Figure 3.2 – Sign up Activity

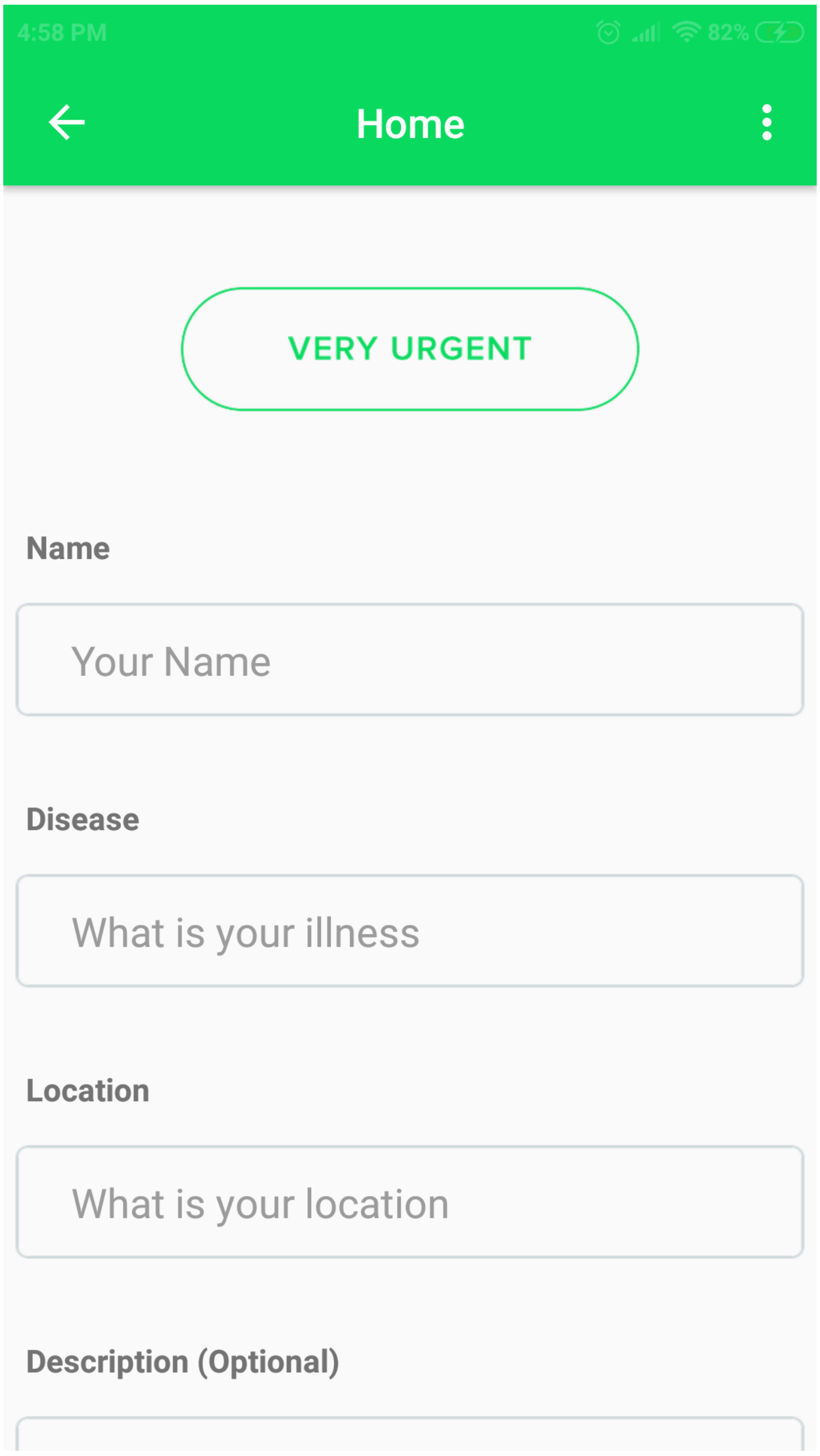
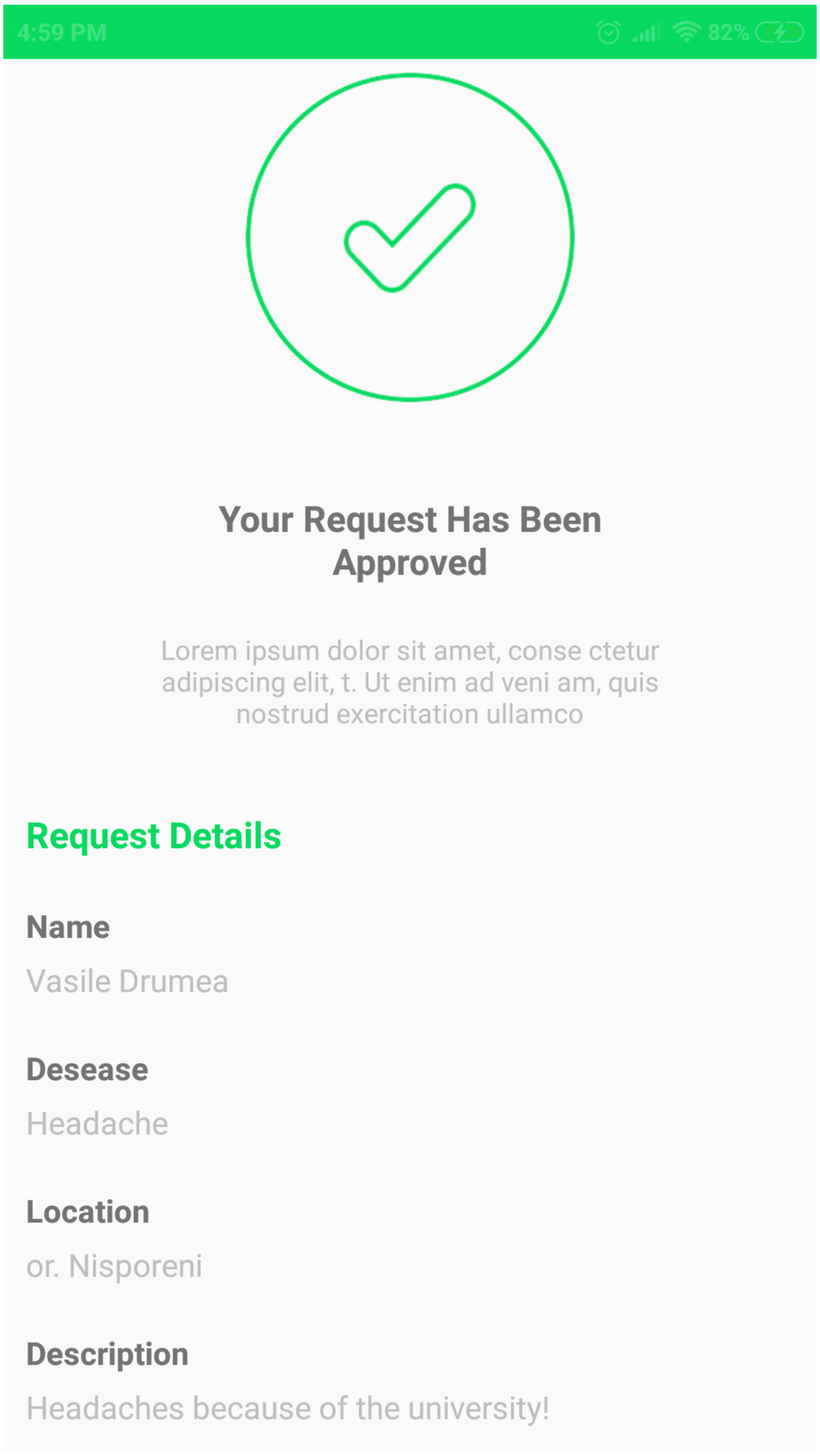


Figure 3.3 – Home Activity



Your Request Has Been Approved

Lorem ipsum dolor sit amet, conse ctetur adipiscing elit, t. Ut enim ad veni am, quis nostrud exercitation ullamco

Request Details

Name

Vasile Drumea

Desease

Headache

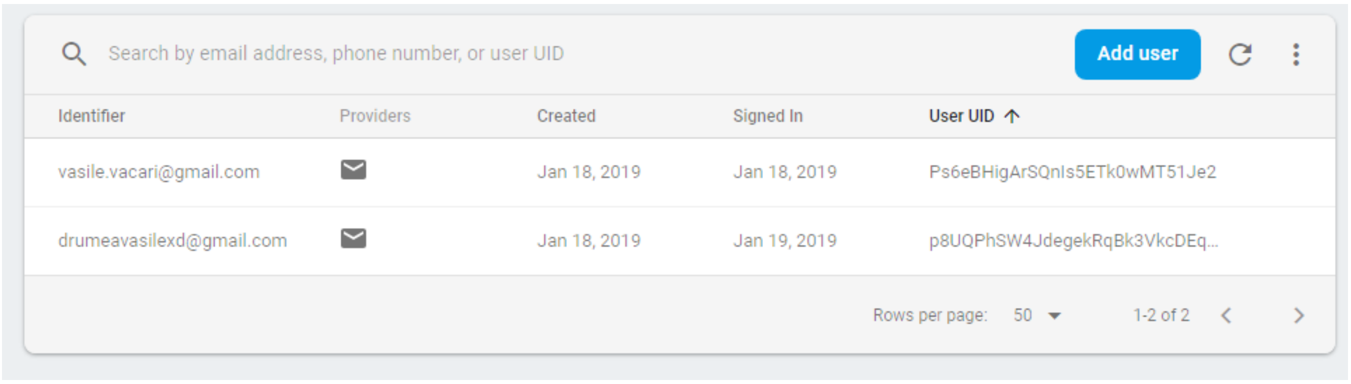
Location

or. Nisporeni

Description

Headaches because of the university!

Figure 3.4 – Request Activity



Identifier	Providers	Created	Signed In	User UID ↑
vasile.vacari@gmail.com	✉	Jan 18, 2019	Jan 18, 2019	Ps6eBHigArSQnIs5ETk0wMT51Je2
drumeavasilexd@gmail.com	✉	Jan 18, 2019	Jan 19, 2019	p8UQPhSW4JdegekRqBk3VkcDEq...

Rows per page: 50 1-2 of 2

Figure 3.5 – Firebase Console Authentication

Conclusion

– In an android app the UI can be made more user-friendly with the help of the design. Besides the options we have in Android Studio or other IDE to style our drawables we can use different software like Photoshop, Illustrator, GIMP etc. to create images, icons and all we need.

– Because of the fact that there are so many different devices that use Android with different resolutions, dimensions were introduced different pixel densities. These are measured either by an abbreviation e.g. tvdpi, mdpi, hdpi etc. or can be a number. Some examples can be seen in Android Studio when we select a device for a preview.

–