PROJECT DOCUMENTATION

MediCare

STUDENȚI

Ivan Bogdan

Hrihoreanu Radu

Project presentation

The MediCare application is an application in the medical field created to facilitate long and repetitive actions such as waiting in line to make an appointment, presenting to the office to find out information such as the schedule and name of the nurse you need to contact in order to an end to be seen by a doctor.

This information can be easily computerized, in some places it is, in others it is not, the main idea being that the information is spread around and you do not know where to look. In the current context of the pandemic, large groups of people gathering due to the disorganization of these medical institutions increase the exponential risk of disease. A good organization of doctor's appointments, a detailed situation of the hospital's medical resources can greatly reduce the risk of infection and provide safety and confidence to people. To this end, MediCare was born.

MediCare aims to solve all the problems mentioned above, having as main goal the good organization of doctor's appointments but also their management. Thus, no need for physical contact before the actual consultation MediCare reduces the chances of infection exponentially. The application allows both patients and doctors to determine in the most efficient way the next meeting, or to inform the opposite party of an unforeseen change that leads to the cancellation of the appointment.

At the end of an appointment, if necessary, the doctor can prescribe a prescription to the patient, very easily, using the application to send him the necessary medicines and in what quantity they must be bought.

In addition to the fact that it offers the possibility of online appointments, in this application you can also manage the stock of medicines and essential materials in the hospital with the possibility to replenish the stock without the need for a physical inventory made by medical staff.

Technologies used

As previously mentioned, MediCare is a mobile application, currently only available for users of phones with Android operating system.

To develop this application we used the Android Studio development environment based on the IntelliJ IDEA software from JetBrains.

As programming languages ​​we used Java, one of the most well-known and powerful languages ​​on the market, but also the Kotlin language. The Kotlin language first appeared in 2011, developed by JetBrains and was made to fully interpolate with the Java language.

Why did we choose to use the Kotlin language as well? Because, thanks to the syntax, you write less code than in other programming environments, the syntax is concise, more precisely designed to take advantage of the powerful capabilities of the new compilers. One of the most important reasons why we chose to use Kotlin is error reduction. The compiler is able to identify potential errors due to the null safety build-in system.

For data handling, but also for login and register functionality we chose to work with Firebase because it is an easy to use tool and is integrated into the IDE used in the development of the application. For authentication we used the Authentication product (FirebaseAuth) to be able to manage users in a simple and secure way. As an authentication method, I chose the email-password option.

For data distribution and storage we used the Cloud Storage product (FirebaseFirestore). Cloud Storage is built so that we can quickly and easily store content in a database such as photos or videos. The Firebase SDK for cloud storage integrates very well with Firebase Authhentication, providing simple and intuitive access control.

In short, we chose Firebase because it helps us develop high quality applications with a large number of users.

**Backend**

Firebase authentication provides backend services, easy-to-use SDKs and ready-made UI libraries to authenticate users in the application. Accept authentication using passwords, phone numbers but also through other popular identity providers, such as Google, Facebook and Twitter and many more.

The backend part of our application is represented by Firebase. First we have the authentication part (FirebaseAuth) where we opted for the login option using email and password. Thanks to Firebase we didn't have to worry about building a backend to support the user authentication process.

How does it work? To log in to a user, you must first receive your user credentials. These credentials can be the user's email address and password. Then pass these credentials to the Firebase authentication SDK. Firebase Services will verify those credentials and return a response to the customer.

As a storage service we have Cloud Storage from Firebase, a powerful, simple and cost-effective object storage service. We use it to store users, as well as their appointments but also details about available drugs.

How does it work? Cloud Storage stores files in a Google Cloud Storage container, making them accessible through both Firebase and Google Cloud. This allows you the flexibility to upload and download files from mobile clients via Firebase SDKs and do server-side processing, such as image filtering or video transcoding using the Google Cloud platform.

Firebase SDKs for Cloud Storage integrate seamlessly with Firebase authentication to identify users and provide a declarative security language that allows us to set access controls on individual files or groups of files so we can make files as public or private we want.

**Arhitecture**

The architecture of our project is structured as follows:

• 5 activities

LoginActivity - is the activity with which the project is launched (if you are not already logged in). In it we take the data from the email and password fields and with FirebaseAuth we try to authenticate. If the authentication was not successful then we notify the user of the cause of the error, otherwise we retain the current user connected to the application in MainActivity after which we open it.

MainActivity - contains all the buttons that define the functionalities of the application. Also here we set the welcome message and depending on the role of the user we show or not certain options. Also in this activity we have the method that deals with logout.

RegisterActivity - in this activity the user creates his account. There are several fields that need to be filled in, some with validation, others not. If the registration succeeds then the user is redirected to the login page to be able to log in, if not the user is informed of the reason behind the failure of the registration.

AppointmentActivity - offers the ability to add, view and cancel appointments. This activity loads snippets dynamically, default loads the menu snippet, then adds the manage appoinment or add appointment snippet to the backstack.

PrescriptionActivity - this activity deals with the management of recipes and depending on the choice of the user dynamically loads fragments to serve this purpose.

• 7 fragments

AppointmentFragment - it contains the buttons corresponding to the actions that the user can perform on the programs

AddAppointmentFragment - it contains the view elements needed to create an appointment (doctor, date, time)

DatePickerFragmet - uses us to choose the date of an appointment

ManageAppointentFragment - CRUD operations by appointment

PrescriptionFragment - it contains the buttons corresponding to the actions that the user can perform on recipes

AddPrescriptionFragment - it contains the view elements needed to create a recipe

PrescriptionAppointentFragment - CRUD operations per appointment

Actions and fragments communicate through static fields in MainActivity, and can be modified from anywhere in the application.

• 9 models

We created a generic model (AbstractUser) that contains the common data (name, email, password, role) between the 5 types of users (nurse, patient, doctor, admin, hospital manager). In addition, we also have models for Medicine, Programming and Prescription.

Besides these 9 models, we also have 3 lists for the application roles, the programming status and the type of doctor.

**Usage**

The application has 5 types of users, each with different rights. The roles are: Patient, Doctor, Nurse, Hospital Manager and Admin.

1. Patient

An account with the role of patient is created every time someone registers in our application. As a patient you can set appointments but also to modify or delete them in case of an unpleasant incident. You can also see the prescriptions given by the doctor.

2. Doctor

A doctor can write a patient's prescription after the consultation. He prescribes what medications to take and what time of day.

3. Support

The nurse is responsible for the hospital's medication needs.

4. Hospital Manager

The hospital manager has access to all users of the application. He can also create accounts as a doctor or nurse, but that's all.

5. Admin

The admin can manage all users of the application. Only he can create a user who has a different role of patient. He can add, delete or modify any account of the application.

Due to the number of quite diverse roles, this application can be used to manage hospital appointments and medications, as well as prescribing.

**Conclusions**

In conclusion, MediCare is a medical application, centralizing and managing various information in the medical field. With its help we can reduce the inter-human interaction which is also recommended in this period of crisis.

With a rich architecture, it has multiple functionalities mentioned in the previous steps.

Being a mobile application is available to the general public for a better organization of time in terms of our health and those around us.

It is a modern application, with a simplistic design, which will help the good handling of this application, offering a pleasant experience to the users.

The application has a large number of roles which offers a wide range of features allowing both hospital staff and patients to feel more comfortable when it comes to the medical field, not to perceive it as something to be feared.

We hope that our application will be useful in these times of crisis and will fluidize the crowded corridors of hospitals.