

Lebanese American University

Department of Computer Science and Mathematics



CSC599: Capstone Project

Instructor: Dr. Nadine Abbas

Track My Health: Telemedicine Capstone Project



Rami Chehab

201804820

Alaa Halwani

201804370

Submitted on December 20, 2021

Table of Contents

I.	ABSTRACT	3
II.	INTRODUCTION	4
a.	Overview	4
b.	Literature Review	5
III.	Project Requirements and Functions	6
a.	User Requirements	6
b.	Application Functions	7
c.	Chatbot Functions.....	7
IV.	Project Design.....	8
a.	Overview Flowchart.....	8
b.	Detailed Flowchart	8
c.	Chatbot Flowchart	9
d.	Login Page	11
e.	Sign Up Page	11
f.	Dashboard.....	12
g.	Patient Records	13
h.	Schedule Appointment.....	14
i.	Lab Test Results	16
j.	Renew Prescription	17
k.	Admin Page – Dashboard.....	17
l.	Admin Page – Add Medication	18
m.	Admin Page – Add Patient records	19
n.	Chatbot.....	19
V.	Integration and Testing	20
VI.	Problems Faced.....	21
VII.	Improvement for the Future	22
VIII.	Conclusion.....	23
IX.	Additional Resources	24
X.	References.....	24

I. ABSTRACT

In a world that is heading toward virtualizing almost everything in order to ensure more practical and accessible solutions to people around the globe, Artificial Intelligence and Machine Learning are two major buzzwords nowadays, and combined with the healthcare industry, marvelous creations can be obtained. The following paper presents a virtual hospital which allows patients to sign up for an account, schedule and book appointments or consultations and receive medical advice regarding their symptoms' severity through the integrated chatbot. The latter uses a machine learning model that predicts the patient's suggested measures to be taken and whether what they're experiencing requires a trip to the ER. This virtual hospital deems itself valuable for it saves time and effort for both doctors and patients, increases access to specialists around the globe and provides patients with the best healthcare services within the comfort of their own homes. To the best of our knowledge, Lebanon lacks such a platform that integrates all the previously mentioned features which makes it more essential to develop, especially in regard to Lebanon's deteriorating economy, fuel crisis and the COVID-19 pandemic.

II. INTRODUCTION

a. Overview

With the advancement of telecommunications and technology, telemedicine has started gaining popularity especially within the past few years. Telemedicine is the practice of providing the necessary health care for patients remotely. The latter is vital these days for it is far more accessible, practical and convenient.

For regular check-ins, telemedicine facilitates communication between patients and their health professionals without having to go to the hospital [\[1\]](#). When it comes to post-surgical patients, telemedicine also facilitates the post-surgical monitoring for patients by their healthcare providers, where doctors can check-in with patients regarding their recovery process and patients can contact their doctors at any time in case any pain or discomfort is experienced. For patients who live in rural areas, telemedicine can provide a great solution which doesn't entail patients having to travel long distances solely for a doctor's visit [\[1\]](#). A telemedicine app can, therefore, increase accessibility for patients no matter their location. The app is also far more practical than live doctor visits for it decreases absenteeism – having to stay away from work or school for long periods of time. Patients who work or study full-time might have it hard to give up several hours, if not their whole day, for a doctor's visit that can be done remotely. This reduces time and cost for patients while still providing them with their needed service [\[1\]](#). In some cases, research has shown that telemedicine proved to be more effective than live visits. For example, psoriasis patients – a skin condition where the latter turns scaly and itchy – received much better care remotely where they were able to have a live video call with their health professional and show them their skin through the camera [\[1\]](#). Telemedicine also helps reduce ER hospital readmissions, saving the spots for patients who are in actual need of urgent care and attention [\[1\]](#).

For all previously mentioned reasons, telemedicine proves to be an important feature to be integrated in our health system here in Lebanon. Due to the COVID pandemic, the deteriorating economy and the spike of the dollar rate, medical services have witnessed a great peak in prices, thus disabling most people with average income to give up seeking medical care due to the lack of finances. Furthermore, with the fuel crisis making transportation extremely expensive and difficult, telemedicine could be the solution we need to ensure that most, if not all, our patients get the health care they need and deserve.

b. Literature Review

When it comes to the literature behind a telemedicine app, we are certainly not the first to attempt creating one. However, based on a list of features that we set to compare the existing apps on the market, we found ours to be integrating the most features in the MENA region, or at least, in Lebanon. MyAUBHealth app [\[2\]](#) is the first that comes to mind for a telemedicine app in Lebanon. The app was created by AUBMC to help patients view their medical health records, schedule appointments, renew their prescriptions and view their lab test results. Another app in the field is My MediCal app [\[3\]](#) which is based in California and created California Medical Assistance program that was mainly designed for low-income families and individuals. The app allows the patients to access their personal health records, schedule appointments, renew their prescriptions, view their lab test results and receive medication reminders. One more telemedicine app is MTBC PHR [\[4\]](#) which allows patients to access their personal health records, book appointments and consult doctors, pay through the app and view their lab report results. Another similar telemedicine app is MedFusion Plus [\[5\]](#), originated in North Carolina, which allows patients to view their personal health records, book appointments, view lab test results and receive medication reminders. One more fairly new telemedicine app that originated in the MENA region is MyMediClinic (UAE) [\[6\]](#). The latter connects seven hospitals and twenty outpatient clinics all around the UAE and allows patients to book in-person or virtual appointments, choose their health care professionals as well as schedule appointments for their family members.

Although there exist telemedicine apps, none of the mentioned ones contain any AI or ML features. Our proposed application allows a patient to view their personal health records and medical history, book and schedule appointments with health care professionals, renew their medical prescriptions, view lab test results, receive medication reminders and benefit from communicating with an AI powered chatbot integrated in the app that can help the user navigate around the app, hand off the conversation to their assigned health care professional if needed as well as provide patients with medical advice in case they were experiencing mild symptoms or urge them to go to the hospital in case they were experiencing serious ones.

Below is the table checklist that was used to compare pre-existing telemedicine apps with our own for a simplified overview.

APP FEATURES CHECKLIST

APP NAME	PATIENT RECORDS	APPOINTMENT SYSTEM	PAYMENT GATEWAY	AI FEATURES	ML-BASED DIAGNOSIS	PRESCRIPTION	LAB TEST RESULTS	MEDICATION REMINDERS	DOCTOR CONSULTATIONS
MYAUBHEALTH	X	X				X	X		X
MYMEDICLINIC	X	X							X
HEALTH HUB	X	X					X		
MTBC PHR	X	X	X				X		X
MY MEDICAL	X	X				X	X	X	X
MEDFUSION PLUS	X	X					X	X	X
OUR APP	X	X		X	X	X	X	X	X

III. Project Requirements and Functions

Given the fact we are implementing a mobile application for our capstone project, it was key to set some requirements before starting the implementation. The requirements for our project can be divided into three parts – the user requirements, the application functions and the chatbot functions.

a. User Requirements

The user requirements are as follows:

- The user should be able to register for an account.
- The user should be able to log in with their registered unique credentials.
- The user should have access to the dashboard page, which contains all the activities the user can perform in the application.
- The user should have a READ ONLY access to his medical records.
- The user should be able to schedule new appointments and view scheduled ones.
- The user should be able to access his lab test results.
- The user should be able to request a renewal for their medical prescription, provided they have one.
- The user should be able to use the chatbot if they are logged in.
- The user should be able to get their symptoms checked or speak to a live agent from the health organization through the chatbot.

b. Application Functions

The application functions are as follows:

- **Login:** This function is responsible for logging the user in by checking whether the user is registered and has used the correct credentials (medical ID and password).
- **Register:** This function is responsible for allowing new users to register for an account in the application by asking the user to input their name, email and password. If the fields are valid, the user is redirected to a page informing him of his medical ID so that he is able to log in.
- **Personal Records:** This function allows the user to view their personal health records which a doctor – admin – fills in. The user only has READ rights and cannot edit this information.
- **Schedule Appointment:** This function allows the user to schedule an appointment by selecting a date and time as well as the appointment type from the dropdown menu.
- **View Upcoming Appointments:** This function is responsible for displaying the appointments scheduled by the user (if any)
- **View Lab Test results:** This function allows the user to download a document in which are his lab test results on a given date.
- **Renew Prescription:** This function allows the user to request a refill for their prescription (if any) by redirecting them to the email app with an auto filled email that they should send.

The application also has extra functions that are only accessible to the admin.

- **Add Medication:** This function allows the doctor to add a medication for a given patient.
- **Add Patient Records:** This function allows the doctor to add and edit medical records for a given patient.

c. Chatbot Functions

The integrated chatbot mainly has two functions.

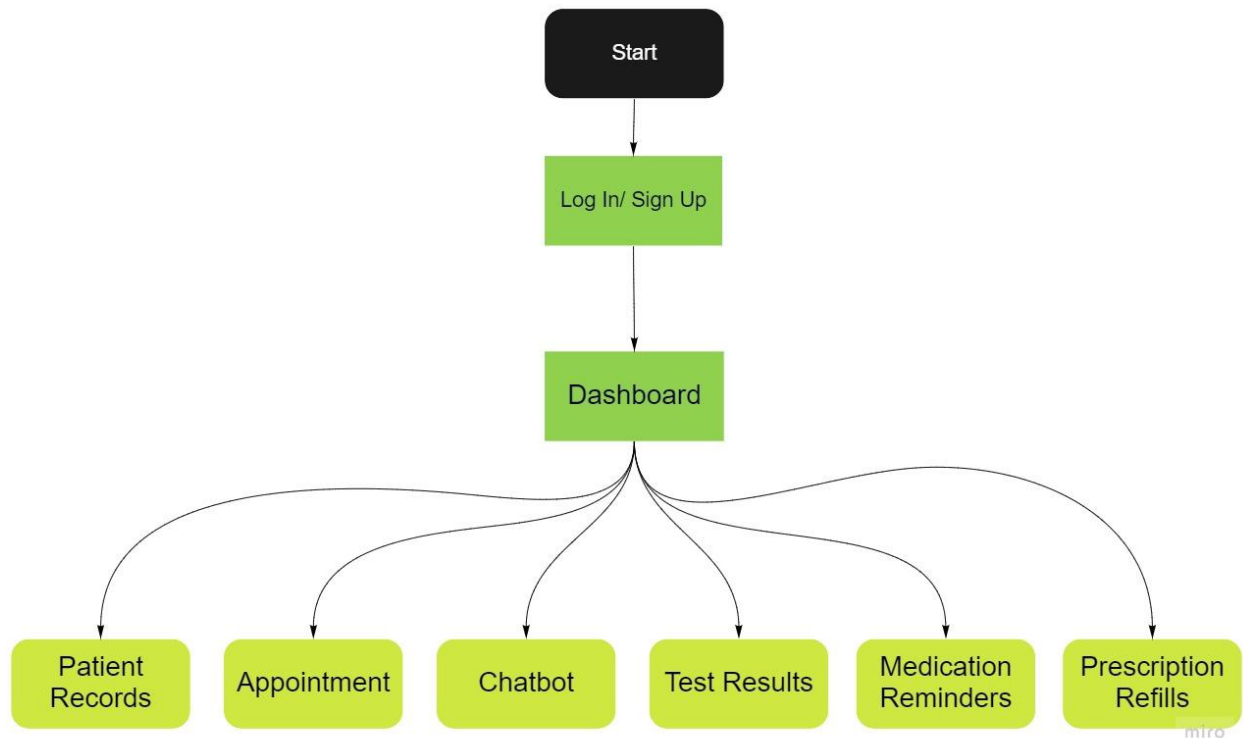
- **Talk to live agent:** This function allows the user to speak to an agent from their medical organization to answer any inquiries or concerns.
- **Check symptoms:** This functions guides the user through a series of questions regarding their symptoms and predicts whether this case is extremely severe – asks the user to go to

the ER immediately, somewhat severe – asks the user to contact the medical organization and schedule an appointment within a couple of hours, slightly severe – asks the user to contact the medical organization and schedule an appointment within the next day, and not severe – asks the user to maintain home care procedures and notify the medical organization in case of any of their symptoms become worse.

IV. Project Design

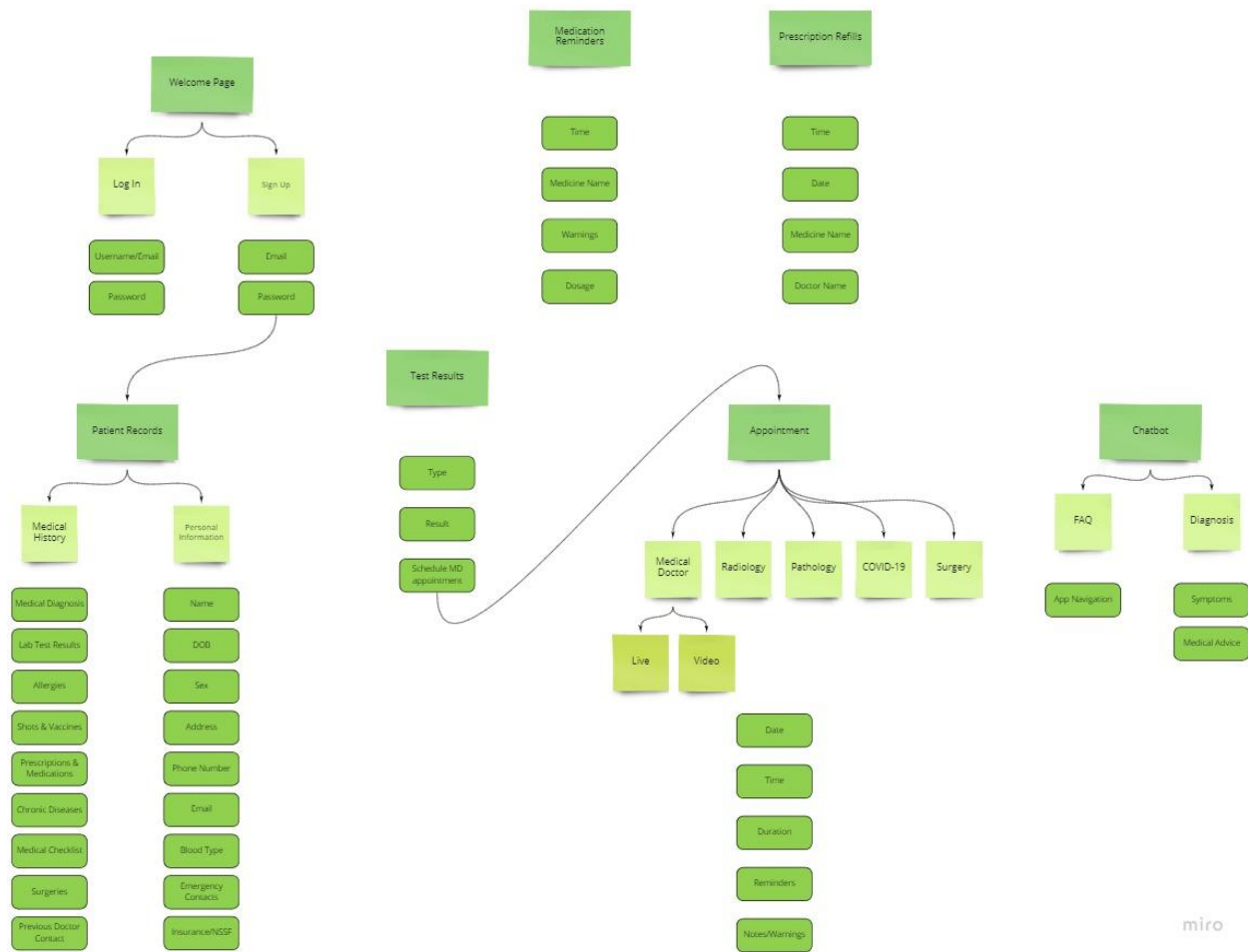
a. Overview Flowchart

Before we started implementing, it was important that we map out the flow of our application. [\[view pdf\]](#)



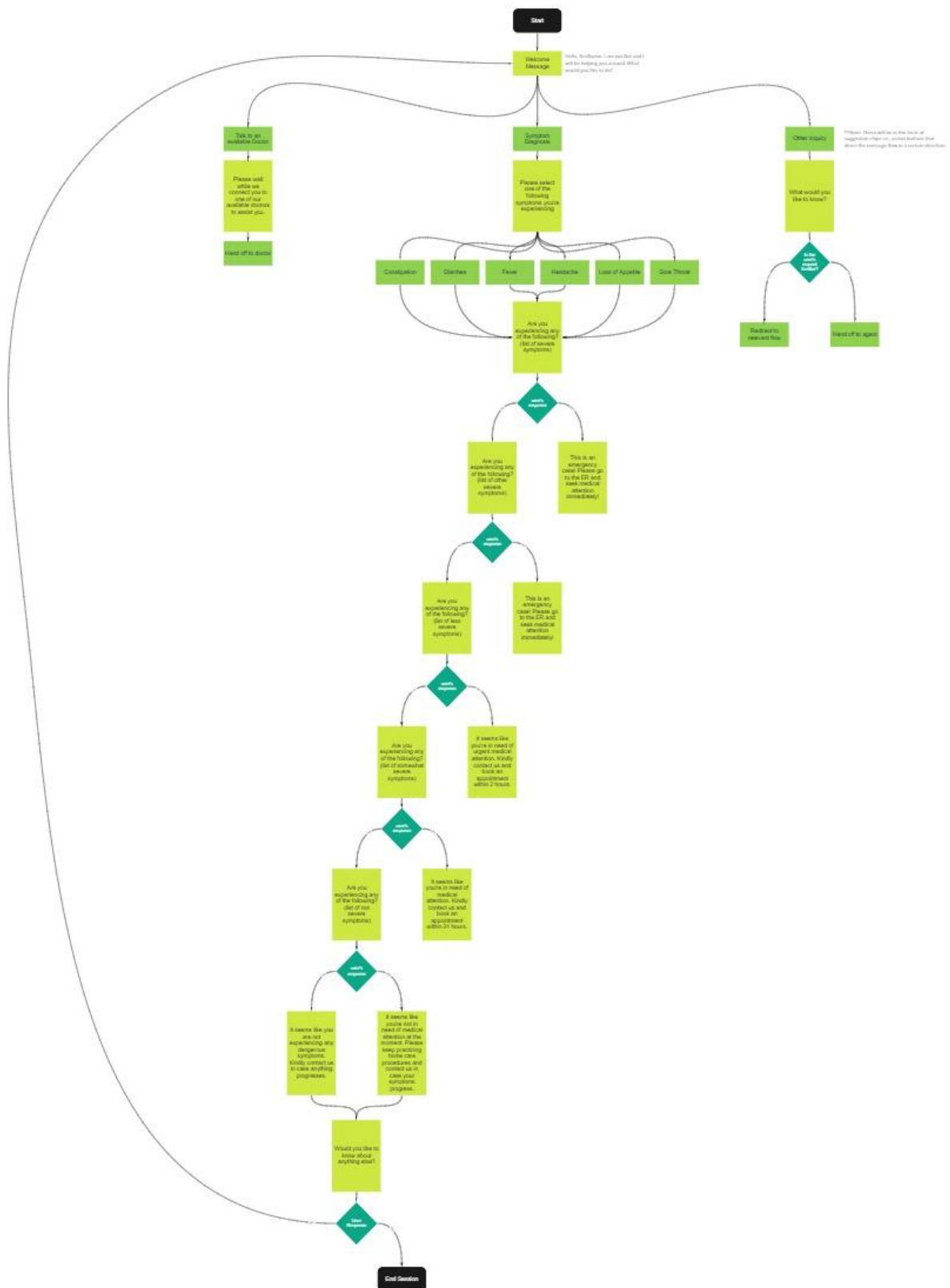
b. Detailed Flowchart

After we came up with our main functionalities, we delved into the details of each to create a more elaborate and detailed flowchart. [\[view pdf\]](#)



c. Chatbot Flowchart

After we mapped out the flowcharts for our app, it was important to map out the way the chatbot will behave. [\[view pdf\]](#)



d. Login Page

The code checks whether the entered credentials match the ones in the database. If they do, the user is logged in, otherwise an error message is shown.



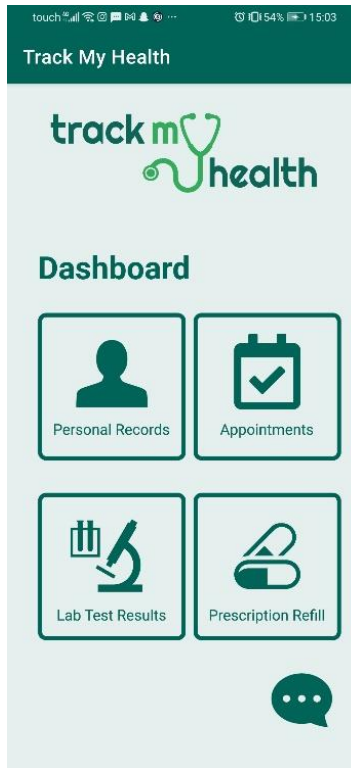
The screenshot shows a mobile application interface for 'Track My Health'. At the top, there is a dark green header with the text 'Track My Health'. Below the header, the app's logo 'track my health' is displayed, with 'track' in dark green, 'my' in a lighter green, and 'health' in dark green, with a stylized heart and pulse line. Below the logo, there are two input fields: the first is labeled 'Med. ID' and the second is labeled 'password'. Both fields have horizontal lines indicating where to enter text. At the bottom of the form, there are two dark green buttons: the top one is labeled 'LOG IN' and the bottom one is labeled 'SIGN UP'.

e. Sign Up Page

The sign-up page is responsible for creating a new user and adding them into the database. Once the user is created, a medical ID is instantiated for them, and they are redirected to a page informing them of their medical ID in order to log in. They can then log into the app successfully.

f. Dashboard

The dashboard allows the user to navigate around all the functions they can use in the app: viewing their personal records, scheduling appointments, viewing their scheduled appointments, viewing their lab test results, renewing and viewing their prescriptions and communicating with the integrated chatbot to check their symptoms severity or talk to a live agent from their medical organization.



g. Patient Records

The patient records allow the user to view their medical records that the doctor created for them.



h. Schedule Appointment

For scheduling the appointment, the user input is saved into an object of type Appointment and then using the firebase database reference the Appointment is saved under the patient's medical ID under an auto generated ID.

```
ArrayAdapter<String> adapterType = new ArrayAdapter<>( context: this,
    android.R.layout.simple_spinner_item, arraySpinnerType);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinnerType.setAdapter(adapterType);

calendarView.setMinDate(c.getTimeInMillis());
calendarView.setOnDateChangeListener(new CalendarView.OnDateChangeListener() {
    @Override
    public void onSelectedDayChange(@NonNull CalendarView view, int year, int month, int dayOfMonth) {
        date=dayOfMonth+"/"+month+"/"+year;
        Log.i( tag: "date",date);
    }
});

confirm.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v){

        Appointment app1=new Appointment(date,spinnerTime.getSelectedItem().toString(),
            spinner.getSelectedItem().toString(),spinnerType.getSelectedItem().toString());

        myRef.child(medID).push().setValue(app1);

        startActivity(intent);

        Toast.makeText(getApplicationContext(), text: "Appointment booked!", Toast.LENGTH_LONG).show();
    }
});
```

[medical-assistant-e661a-default-rtdb](#) > [appointments](#)

appointments

0

-MrWa_EuQyuxm2yGMVVi

```
appType: "Radiology Appointmen"
date: "30/11/2021"
time: "14:00"
type: "Online"
```

Using the firebase database reference, all the appointments stored under the same child with is the section for the exact patient that is logged into the app are fetched and saved in array lists. Then they are arranged in a list view where each value is fetched from each array list according to the position in the list view where the patient can view them.

```
appType=new ArrayList<>();
appType2=new ArrayList<>();
appDate=new ArrayList<>();
appTime=new ArrayList<>();
id=new ArrayList<>();

DatabaseReference myRef2 = database.getReference( path: "appointments").child(medID);

myRef2.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot snapshot) {
        for(DataSnapshot medicine:snapshot.getChildren() ){...}
    }

    @Override
    public void onCancelled(@NonNull DatabaseError error) {

    }
});

adapter=new MyListAdapter( context: this, appType, appDate,appTime,appType2);
list=(ListView)findViewById(R.id.list);
list.setAdapter(adapter);
```

```
this.context=context;
this.appType=appType;
this.appDate=appDate;
this.appTime=appTime;
this.appType2=appType2;
}

public View getView(int position, View view, ViewGroup parent) {
    LayoutInflater inflater=context.getLayoutInflater();
    View rowView=inflater.inflate(R.layout.activity_mylist, root: null, attachToRoot: true);

    TextView titleText = (TextView) rowView.findViewById(R.id.title);
    TextView dateText = (TextView) rowView.findViewById(R.id.date);
    TextView timeText = (TextView) rowView.findViewById(R.id.time);
    TextView typeText = (TextView) rowView.findViewById(R.id.type);

    titleText.setText(appType.get(position));
    dateText.setText(appDate.get(position));
    timeText.setText(appTime.get(position));
    typeText.setText(appType2.get(position));

    return rowView;
```

touch 54% 15:12

Track My Health

December 2021

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Time

8:00

Appointment Type

Cardiology Appointment

CONFIRM **VIEW APPOINTMENTS**

i. Lab Test Results

The lab test results are arranged by date in a list view. Upon choosing one, the patient can view the results for that particular lab.

touch 54% 15:14

Track My Health

Lab 11/2021

Lab 3/2021

Lab 6/2019

touch 54% 15:14

Track My Health

GNU Solidario Hospital
Avenida del Norte 22483
Las Palmas de Gran Canaria
Spain

LABORATORY REPORT

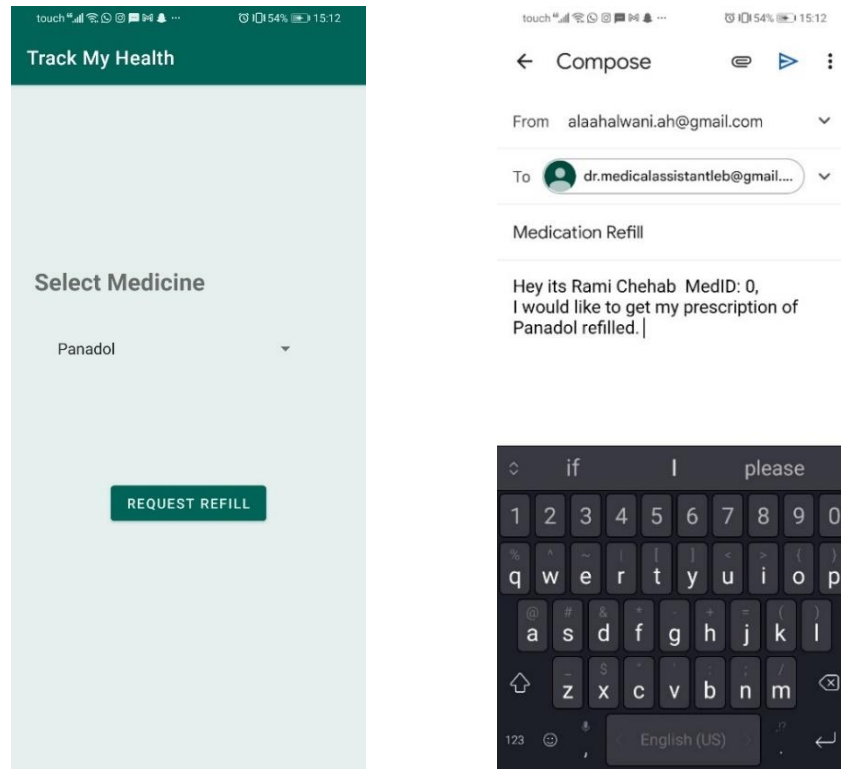
Name: **Ana Betz** Patient ID: **PM201**
Date: **2011-08-25 09:32** Age: **25y 10m 26d** Sex: **Female**
Doctor: **Carmen Cordara** Test id: **B105AAT6**

Test Name	Result	Normal Range	Units
Hemoglobin	12	11.0 - 16.0	g/dL
HCT	33	35.5-50.0	%
MCV	83	82-95	fL
MCH	29	27-31	pg
MCHC	33	32.0-36.0	g/dL
RDW CV	12	11.5-14.5	%
RDW SD	44	35-56	fL
WBC	6.7	4.5-11	10 ⁹ /3uL
NEUT%	60	40-70	%
LYMP%	30	20-45	%
MON%	8	2-10	%
EOS%	2	1-6	%
BA%#	0	0-2	%
LYMP	2	1.5-4.0	10 ⁹ /3uL
GRA#	4.7	2.0-7.5	10 ⁹ /3uL
PLT	256	150-450	10 ⁹ /3uL
ESR	2	up to 15	mm/hr

Digitally signed by
Dr. Carmen Cordara
GNU Public Key : E44311F4
Test id : B105AAT6

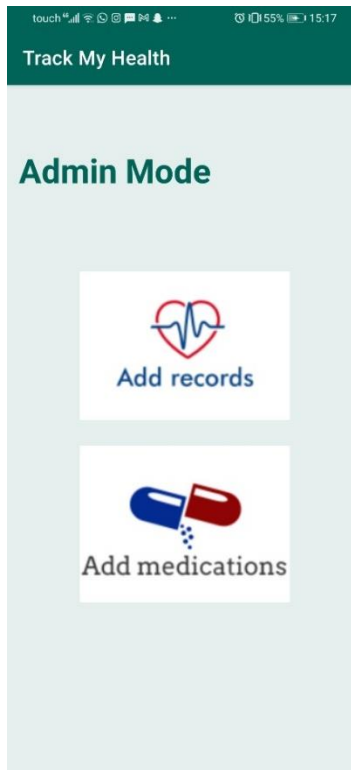
j. Renew Prescription

This function allows the user to request a refill for his prescription (if any). The app will redirect the patient to sending a prefilled email containing their medical ID, name and the prescription medication name.



k. Admin Page – Dashboard

This page is to be used by the Doctor in order to add medication for a patient or add their records.



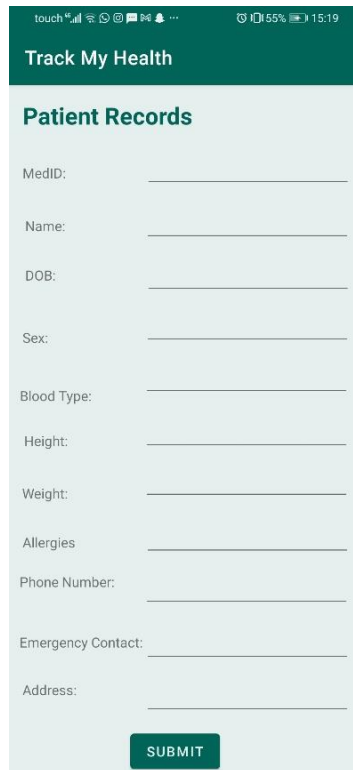
l. Admin Page – Add Medication

This page allows the doctor to add a medication for a patient using their medical ID.

A screenshot of the "Track My Health" app showing the "Add Medication" form. The form has two input fields: "Patient Med. ID" and "Medicine Name:". Below the second field is a green "SUBMIT" button. The status bar at the top shows "touch" with signal icons, 54% battery, and the time 15:16.

m. Admin Page – Add Patient records

This page allows the doctor to add medical records for a patient using their medical ID.

A screenshot of a mobile application interface titled "Track My Health". Below the title is a section labeled "Patient Records". The form contains several input fields: "MedID:", "Name:", "DOB:", "Sex:", "Blood Type:", "Height:", "Weight:", "Allergies", "Phone Number:", "Emergency Contact:", and "Address:". Each field has a corresponding text input line. At the bottom of the form is a green button labeled "SUBMIT". The top of the screen shows a status bar with various icons and the time "15:19".

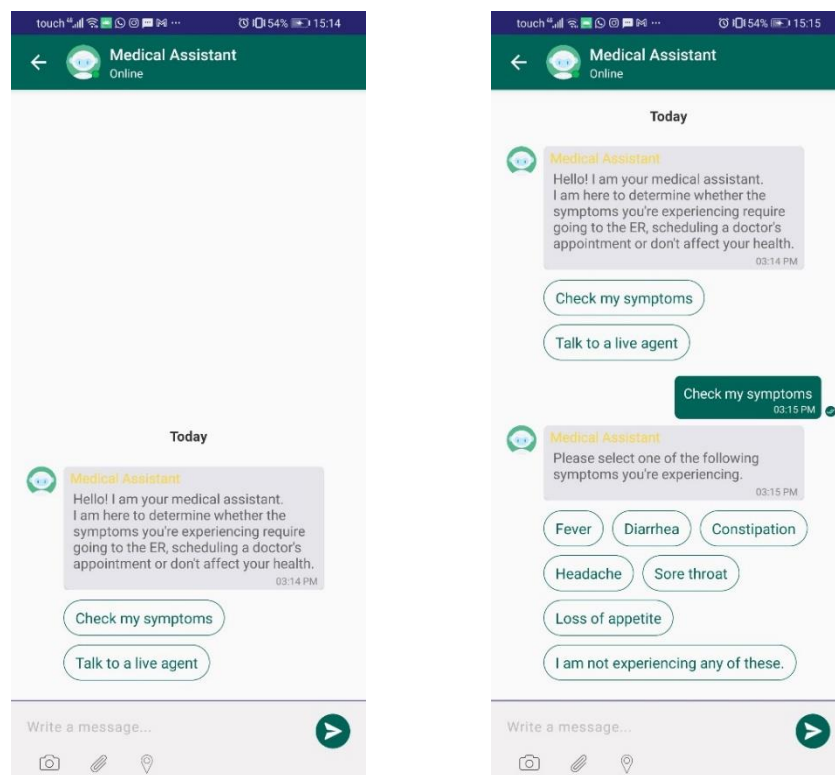
n. Chatbot

The chatbot was created on a platform called Kommunicate [\[7\]](#) which has several functionalities and allows for bot integrations. The platform allows importing bots that were built on several other platforms as well as its own bot-building platform – Kompose. The way this particular chatbot is built is a decision tree. First, the chatbot asks the user if they would like to speak to a live agent or check their symptoms. If the user picks the latter, the bot asks the user to select a symptom they're experiencing from a list as shown in the image. After the user selects one, the bot proceeds to display another set of symptoms and ask the user if they're experiencing any of those. If the user picks one, then this is a severe case, and the user is advised to immediately go to the hospital and seek medical attention. If the user selects none, the bot proceeds to ask the user about a different set of symptoms and so on and so forth until one of the following decisions are reached:

- This sounds like an emergency case, please head straight to the ER and seek medical attention immediately!

- This sounds like you need some urgent medical attention, kindly contact us and book an appointment within the next 2 hours or seek medical attention within the next few hours.
- It sounds like you could use some medical attention, kindly contact us and book an appointment within the next 24 hours.
- It seems like you're not in need of a doctor at the moment. Kindly resume your home care procedures and contact us in case any of the symptoms you're experiencing progresses.

All the symptoms that were used in the bot were supplied from medical research [\[8\]\[9\]\[10\]\[11\]\[12\]\[13\]\[14\]\[15\]\[16\]\[17\]](#) as well as some help from our relatives who are doctors.



V. Integration and Testing

The integration of Firebase [\[18\]](#) in Android Studio is seamless since both are developed by google. Android Studio already integrates the Firebase libraries and can be easily setup and connected. The app is connected via a configuration file that is setup once the developer imports the SDK files when selecting Firebase as a tool to be used in the project. With the help of a few YouTube tutorials [\[19\]\[20\]\[21\]\[22\]\[23\]\[24\]\[25\]](#), the integration went smoothly.

After building the chatbot on Kommunicate and the mobile application on Android Studio, it was time to integrate the two together to obtain our final version of the app. The steps of the integration were present in the Kommunicate documentation [\[26\]](#), and they were clear and straightforward. The first step was to install the Kommunicate dependency in Android Studio. The version of the package provided in the documentation was '2.1.8'. However, after facing some issues and doing some research, we found out the correct version to use was '2.1.5'. Next, it was time to authenticate the bot [\[27\]](#) using its unique ID that we got from the Kommunicate Dashboard. Then we set up the conversation as mentioned in the documentation [\[28\]](#). After following all the corresponding steps, our app was finally ready and fully functional.

To test our app, we tried all our application's functionalities and, thankfully, they were all successful.

VI. Problems Faced

In this project, we, unfortunately faced several issues. When it comes to the application itself, we first started using Android Studio for frontend and Firebase for the backend. A week or two in, we decided to switch to Ionic due to the flexibility and better UI features. However, when trying to integrate ionic with firebase we faced several issues in regard to authentication and form validation techniques. Since the deadline was close, we made the decision to switch back to Android Studio about a week prior to the submission deadline. Luckily, we were able to implement all the functionalities we initially planned to do.

During the implementation on Android Studio, we faced several issues as well. To start, the UI wasn't easy to implement due to the fact that the platform doesn't support intricate designs and beautiful UIs. Also, the functionalities in Android also took an extremely long time to implement due to the platform's complexity.

As for the implementation, it was tough to integrate and connect the backend to the frontend which is why we used Firebase, a NoSQL database service. Firebase was not an easier route, but a more convenient one. It still has drawbacks which is the lack of proper querying which helps in the process of fetching, editing, and deleting data.

Given that fact that we were taking the Mobile Development and Web development courses alongside the Capstone course, we were still new to the idea of full stack development. It was

tricky at first, but with the help of our instructors, several tutorials and documentation we were able to figure out backend logic and implementation of front end with backend.

Regarding the chatbot, a lot of issues were faced as well. The initial plan for the chatbot was to help the user with scheduling and viewing appointments, requesting prescription refills, viewing lab test results, checking the symptoms and speaking to a live agent. All the functionalities with the exception of the last two did not end up working due to their need of webhooks. Webhooks are custom API calls that link different platforms or applications together. However, due to the lack of resources about using webhooks in the chatbot's platform – Kommunicate – and firebase, it became impossible to implement eventually. We therefore decided to remove the functionalities than can be done by the user through the app and keep the symptom checker and the hand off to a live agent to add to the app's functionalities and reduce redundancies.

VII. Improvement for the Future

Regarding future improvements, the webhook that we weren't able to implement can be a great way to take the app to the next level, helping the user navigate through the app even more, and facilitating any activity they have to perform in case they weren't coming from a solid tech background.

When it comes to the app itself, several modifications can also be made such as having three different kinds of users: the main admin (Hospital IT) responsible for creating doctors, patients and have full access to everything within the app, the doctor who has assigned patients and is able to add patient medical records, their medication and view his upcoming scheduled appointments and the patient who can only view his records, schedule and view appointments, view his lab results, request prescription refills and use the chatbot.

Another improvement we can do is better structure the appointment system, where it shows the user the available time slots only rather than have them pick a time, as well as add an option to cancel the appointments at any given time.

Furthermore, checking for user input errors such as entering a non-existing email can be implemented. For a more secure app, a confirmation email to activate a new account can be sent and implemented.

VIII. Conclusion

To sum up, we were able to create a telemedicine application [\[view app\]](#) that integrates an AI component – the chatbot. The app helps doctors from medical organizations such as hospitals or clinics as well as their patients keep track of their health and their personal health records. This application also facilitates many hardships patients may suffer such as dangers on their health due to the COVID-19 pandemic, the economic crisis Lebanon is facing and the inflation in transportation fees. We believe this app still has so much growth potential and would be of great help if implemented in Lebanon.

IX. Additional Resources

- a. [Overview Flowchart PDF](#)
- b. [Detailed Flowchart PDF](#)
- c. [Chatbot Flowchart PDF](#)
- d. [Track My Health Application](#)
- e. [Poster](#)

X. References

- [1] [10 Reasons Telemedicine is Gaining Popularity in 2021 - Health SAF](#)
- [2] [MyAUBHealth - Login Page \(aubmc.org.lb\)](#)
- [3] [My Medical - Apps on Google Play](#)
- [4] [MTBC PHR on the App Store \(apple.com\)](#)
- [5] [Medfusion Plus mobile health record app: Your health data, at your fingers.](#)
- [6] [MyMediclinic 24x7](#)
- [7] [Customer Support Automation Software | Support Bots | Kommunicate](#)
- [8] [Fever - Symptoms and causes - Mayo Clinic](#)
- [9] [Diarrhea - Symptoms and causes - Mayo Clinic](#)
- [10] [Diarrhea When to see a doctor - Mayo Clinic](#)
- [11] [Strep throat - Symptoms and causes - Mayo Clinic](#)
- [12] [Sore throat in adults - Mayo Clinic](#)
- [13] [Sore throat - Symptoms and causes - Mayo Clinic](#)
- [14] [Headache When to see a doctor - Mayo Clinic](#)
- [15] [Constipation in adults - Mayo Clinic](#)
- [16] [Constipation - Symptoms and causes - Mayo Clinic](#)
- [17] [Loss of Appetite: Symptoms, Signs, Causes & Treatment \(medicinenet.com\)](#)

- [18] [Firebase console \(google.com\)](#)
- [19] <https://www.youtube.com/watch?v=hUViewFochtk&t=313s>
- [20] <https://www.youtube.com/watch?v=WeoryL3XyA4&t=461s>
- [21] <https://www.youtube.com/watch?v=9JdbgoYgCyA&t=506s>
- [22] <https://www.youtube.com/watch?v=AyGK4O9m2hA>
- [23] <https://www.youtube.com/watch?v=2duc77R4Hqw&t=8s>
- [24] <https://www.youtube.com/watch?v=kGkd1hrbnWY&t=499s>
- [25] <https://www.youtube.com/watch?v=mpd0Al01jjY&t=75s>
- [26] [Installation · Developer Docs | Kommunicate](#)
- [27] [Authentication · Developer Docs | Kommunicate](#)
- [28] [Conversation · Developer Docs | Kommunicate](#)