Aplicație pentru ajutor medical virtual

MedBuddy

Candidat: Cotulbea Adrian-Ionuț
Coordonator științific principal: Prof. dr. habil. Ing. Marius Marcu
Coordonator științific secundar: ing. Claudiu Groza

Motivația

- Neglijarea sănătății datorită lipsei de timp
- Contextul pandemic recent
- Îmbunătățirea comunicării dintre medic și pacient

Ideea

- Aplicație mobilă
- Experiențe adecvate rolurilor
- Istoric medical digital

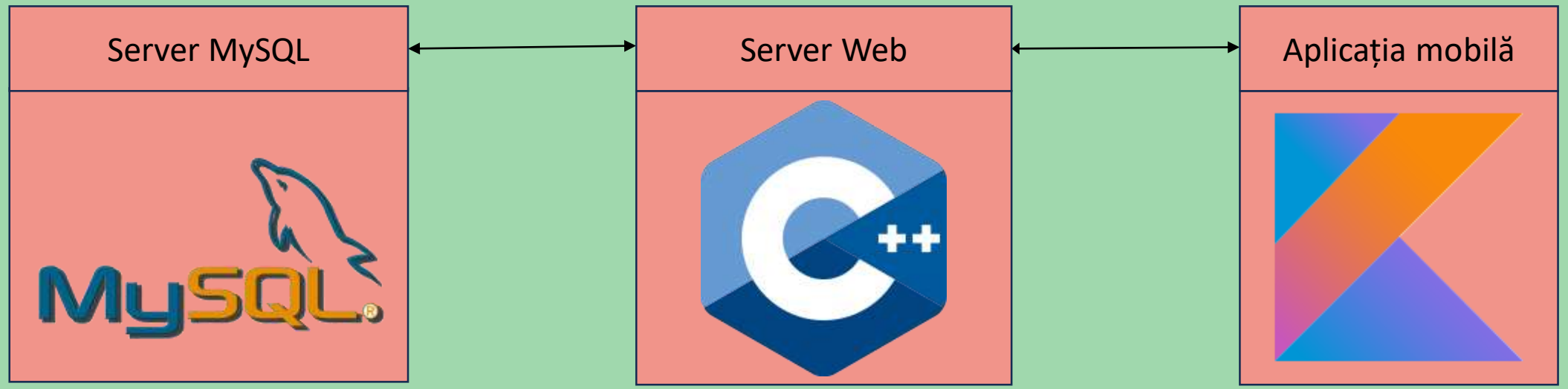
Beneficii

- Obținerea mai rapidă a unui diagnostic și a unui tratament corespunzător
- Interacțiunea dintre medic și pacient
- Gestionarea digitală de programări la consult

Aplicații similare

- Practo
- Doctor On Demand
- Talkspace

Arhitectura sistemului



Server Web

Funcționalitatea serverului:

- Citirea cererii pe socket
- Identificarea acesteia
- Extragerea eventualilor parametrii
- Interogarea bazei de date corespunzător
- Trimiterea răspunsului către client

Imaginea personalizată de docker pentru serverul web:

```
FROM gcc:latest

COPY ../server.cpp /app/

WORKDIR /app

RUN apt-get update && apt-get install -y libasio-dev libmysqlcppconn-dev

RUN g++ -o server server.cpp -lmysqlcppconn

EXPOSE 8080

CMD ["./server"]
```

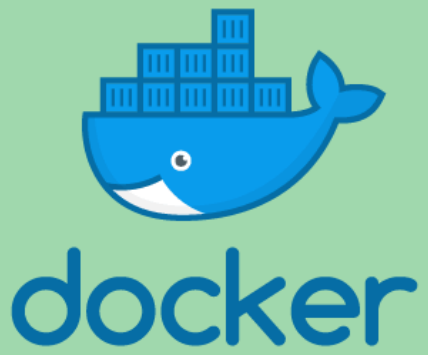
Biblioteci folosite:

- MySQL Connector/C++
- Asio

Docker-compose

Compus din 2 servicii:

- medbuddy-db, fiind o imagine de MySQL
- medbuddy-sv, fiind imaginea personalizată a serverului C++ construită anterior



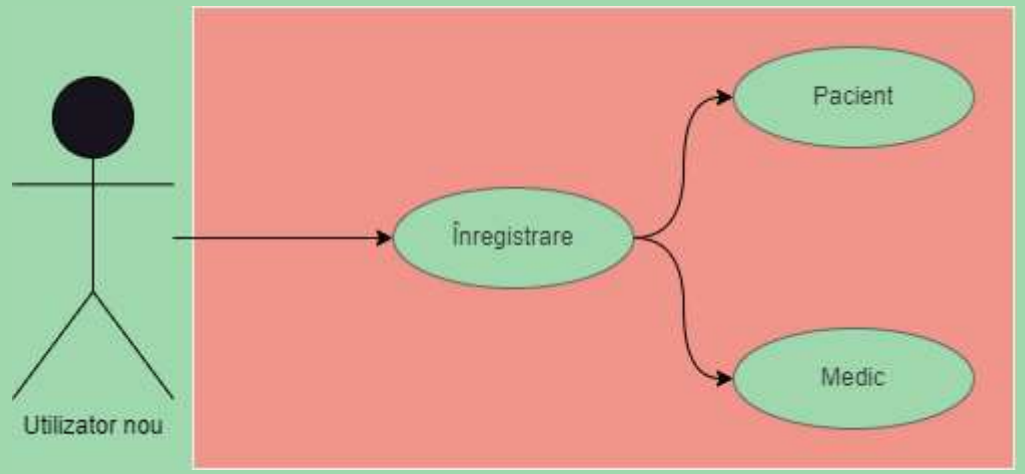

```yaml
services:
  medbuddy-sv:
    build:
      context: .
      dockerfile: Dockerfile

############################
    depends_on:
      - medbuddy-db
############################
  medbuddy-db:
    image: mysql:latest
```

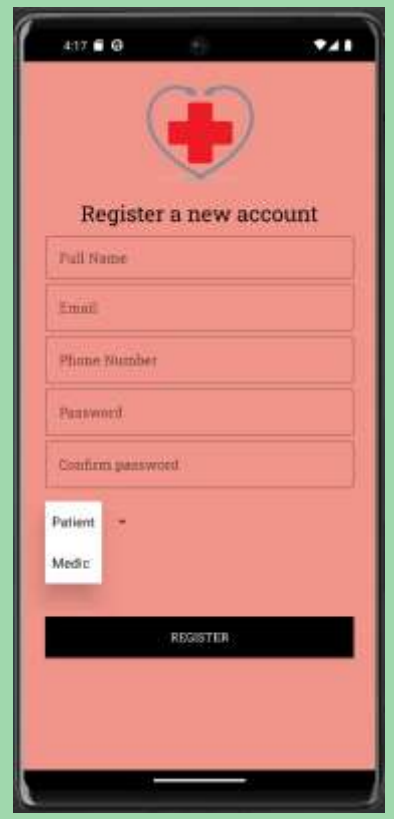
Server MySQL

Baza de date medbuddy conține următoarele tabele:

- users
- userDetails
- doctorSpecialty
- medical_records
- appointment
- messages

Aplicația MedBuddy





Manipularea datelor

Perspectiva pacientului

Apelarea API-ului construit specific pentru această aplicație, folosind biblioteca Retrofit:

```
sendButton.setOnClickListener {
    val message = messageBox.text.toString()
    val aux2Call = apiService.sendMessage(interactionID, doctorID, patientID, message)
    aux2Call.enqueue(object : Callback<String> {
```

```
@FormUrlEncoded
@POST("/getMessages")
fun getMessages(
    @Field("roomID") roomID: String?
): Call<String>

@FormUrlEncoded
@POST("/sendMessage")
fun sendMessage(
    @Field("roomID") roomID: String?,
    @Field("senderID") senderID: String?,
    @Field("receiverID") receiverID: String?,
    @Field("message") message: String
): Call<String>
```

Perspectiva medicului

Toate operațiile executate în aplicație se reflectă în baza de date SQL prin apelurile API.

Experiența pacientului

Meniul principal

Actualizarea profilului

Crearea unei cereri

Experiența pacientului

Meniul principal

Actualizarea profilului

Răspunsul unei cereri

Concluzii

- Aplicația Medbuddy reușește să gestioneze cererile efectuate de pacienți.
- Medicii și pacienții dispun de istoricul digital.
- Aceștia pot comunica printr-o fereastră de dialog în cadrul unui tratament virtual.

Direcții de viitor

Aplicația mobilă

- Adăugarea funcționalităților de notificări și recenzii
- Îmbunătățirea aspectului

Server

- Adaptare la rularea pe mai multe fire de execuție
- Trecere de la HTTP la HTTPS
- Criptarea datelor pentru o securitate sporită

Mulțumesc!