**Universitatea Alexandru Ioan Cuza**

**Facultatea de informatică**



**Proiect: HealthTrackerNetwork**

Hrițcu Denis Gheorghe

Anul II , Grupa A2

**Cuprins**

1. Introducere..................................pag 3
2. Tehnologii....................................pag 3
3. Structura aplicației......................pag 4
4. Aspecte de implementare...........pag 5
5. Concluzii.....................................pag 8
6. Referințe Bibliografice.................pag 8

**Indroducere:**

**1.) Viziunea generală:** HealthTrackerNetwork este un proiect axat pe simularea unei rețele de dispozitive de monitorizare a sănătații, a căror date sunt trimise unui server central cu scopul de a fi stocate. De aici informațille vor fi analizate prin intermediul unui client expus doar unui personal medical specializat ce are la dispoziție o multidinea de comenzii ce duc căutarea la un nivel avansat.

**2.) Obiective:**

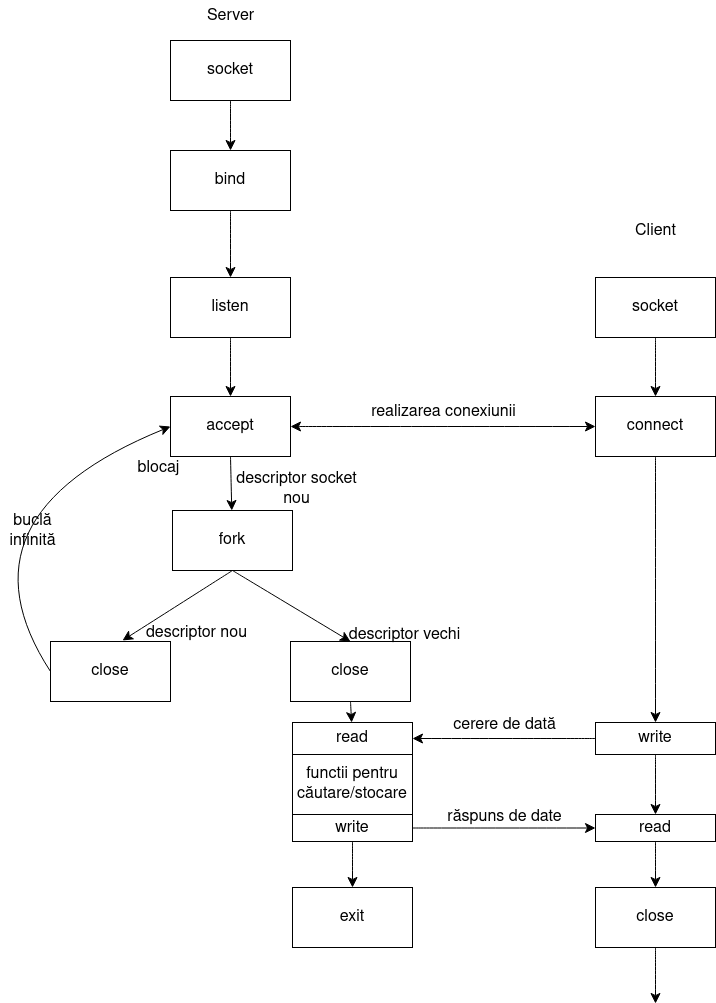
* Crearea unui client adresat pacienților care va genera date precum: nume, vârstă, înălțime, greutate, ritm cardiac, nivel de oxigen din sânge, grupa sanguină, alergii și multe altele.
* Configurarea unui server central, care va primi și va stoca informațiile primite de la clientul destinat pacienților și implementarea unor funcții pentru căutarea avansată, analizarea și alartarea anomaliilor.
* Dezvoltarea unei interfețe ușor de folosit destinată personalului medical printr-un client special.

**Tehnologii utilizate**

Pentru comunicarea între server și client voi folosi un protocal TCP concurent cu fork. Folosesc TCP deorece informațiile sunt primite si trimise integral în ordinare corectă. Utilizarea fork-ului redă aspectul concurent, astfel utilizarea simultană a mai multor cliente este posibilă. Pentru gestionarea datelor voi folosi sqlite, care va permite o stocare ușoară și eficientă a datelor de sănătate dar și a conturilor medicilor.

**Structura aplicației**

Aplicația are la bază o arhitectură de tip client-server în care comunicarea se realizează prin protocolul TCP concurent. Serverul principal se ocupă cu gestionarea conexiunilor concurente, primește, stochează și gestioneaza datele de sănătate utlizând o bază de date SQLite. Două cliente vor comunica cu acest server. Primul destinat pacienților va simula date legate despre sănătate și le va comunica cu serverul, acesta va trimite înapoi acestui client un mesaj de confirmare. Al doilea și cel principal este securizarat print-o username și o parolă astfel încât doar un medic specializat va putea accesa datele și va conține comenzii cu ajutorul va primi de la server informațiile necesare.



**Aspecte de implementare**

**1.)Server**

* Crearea unui socket:

*if ((sd = socket(AF\_INET, SOCK\_STREAM, 0)) == -1) {*

*perror("[server]Eroare la socket().\n");*

*return errno;*

*}*

* Binding:

if (bind(sd, (struct sockaddr \*)&server, sizeof(struct sockaddr)) == -1) {

perror("[server]Eroare la bind().\n");

return errno;

}

* Listen:

if (listen(sd, 1) == -1) {

perror("[server]Eroare la listen().\n");

return errno;

}

* Accept:

client = accept(sd, (struct sockaddr \*)&from, &length);

if (client < 0) {

perror("[server]Eroare la accept().\n");

continue;

}

* Fork+close+read+funcții+write+exit:

int pid;

if ((pid = fork()) == -1) {

close(client);

continue;

} else if (pid > 0) {

close(client);

while (waitpid(-1, NULL, WNOHANG) > 0)

;

continue;

} else if (pid == 0) {

close(sd);

while (1) {

//funcții pentru primire,căutare,trimire,stocare

}

close(client);

exit(EXIT\_SUCCESS);

}

* Crearea table

sqlite3 \*db;

char \*zErrMsg = 0;

int rc;

rc = sqlite3\_open("bazadate.db", &db);

if (rc)

{

fprintf(stderr, "Nu s-a putut deschide/crea baza de date: %s\n", sqlite3\_errmsg(db));

exit(EXIT\_FAILURE);

}

else

{

fprintf(stdout, "Baza de date deschisa/creata cu succes\n");

}

int rows = 0;

sqlite3\_stmt \*stmt;

const char \*rowCountSQL = "SELECT count(\*) FROM sqlite\_master WHERE type='table' AND name='MonitorizareSanatate';";

rc = sqlite3\_prepare\_v2(db, rowCountSQL, -1, &stmt, 0);

if (rc != SQLITE\_OK)

{

fprintf(stderr, "Eroare la pregatirea interogarii: %s\n", sqlite3\_errmsg(db));

sqlite3\_close(db);

exit(EXIT\_FAILURE);

}

if (sqlite3\_step(stmt) == SQLITE\_ROW)

{

rows = sqlite3\_column\_int(stmt, 0);

}

sqlite3\_finalize(stmt);

if (rows == 0)

{

fprintf(stdout, "Tabelul nu există, creând tabelul...\n");

const char \*createTableSQL = "CREATE TABLE IF NOT EXISTS MonitorizareSanatate ("

"Nume TEXT NOT NULL,"

"Prenume TEXT NOT NULL,"

"DataOra TEXT NOT NULL,"

"Varsta INTEGER ,"

"Greutate REAL,"

"Inaltime REAL ,"

"RitmCardiac INTEGER ,"

"SpO2 REAL ,"

"TensiuneSistolica INTEGER ,"

"TensiuneDiastolica INTEGER ,"

"Temperatura REAL ,"

"Glicemie REAL ,"

"ParametriSuplimentari TEXT ,"

"Comentarii TEXT"

");";

rc = sqlite3\_exec(db, createTableSQL, NULL, 0, &zErrMsg);

if (rc != SQLITE\_OK)

{

fprintf(stderr, "Eroare la crearea tabelului: %s\n", zErrMsg);

sqlite3\_free(zErrMsg);

sqlite3\_close(db);

exit(EXIT\_FAILURE);

}

else

{

fprintf(stdout, "Tabel creat cu succes\n");

}

}

else

{

fprintf(stdout, "Tabelul există deja\n");

}

sqlite3\_close(db);

**2.)Client**

* Socket:

if ((sd = socket (AF\_INET, SOCK\_STREAM, 0)) == -1)

{

perror ("Eroare la socket().\n");

return errno;

}

* Connect:

if (connect (sd, (struct sockaddr \*) &server,sizeof (struct sockaddr)) == -1)

{

perror ("[client]Eroare la connect().\n");

return errno;

}

* Read+write+close

while(1)

{

....

if (write (sd,msg,strlen(msg)) <= 0)

{

perror ("[client]Eroare la write() spre server.\n");

return errno;

}

if (read (sd,msg,sizeof(msg)) < 0)

{

perror ("[client]Eroare la read() de la server.\n");

return errno;

}

.....

}

close (sd);

}

**Concluzii:**

Potențialele îmbunătățiri constau în: adăugarea unei interfețe grafice pentru cele două cliente, extinderea funcționalităților de căutare și filtrare, criptarea în transmitarea informațiilor.

**Update:** Implementarea bazelor de date și a interfeței grafice.

**Referințe Bibliografice:**

<https://profs.info.uaic.ro/~computernetworks/>

<https://www.andreis.ro/>

<https://www.tutorialspoint.com/sqlite/sqlite_c_cpp.htm>

https://docs.gtk.org/