Raport Tehnic

Caplea Veronica

1. Introducere
   1. Viziune Generala

Proiectul Offline Messenger are ca scop crearea unei aplicatii de tip client-server care sa sustina trimiterea de mesaje intre utilizatori, serverul fiind unul concurent. Un utilizator care nu are cont va avea optiunea de a-si crea unul si de a putea schimba apoi mesaje cu alti clienti. Numele pe care il va alege va fi verificat, deoarece fiecare client retinut in baza de date o sa aiba un nume de utilizator unic. De asemenea, asocierea nume-parola si alegerea celui de al doilea participant la conversatie vor fi verificate pentru a putea ajunge la conectarea catre server si interschimbarea de mesaje. Aplicatia va oferi posibilitatea de a trimite mesaje si utilizatorilor ce nu sunt activi. Toate mesajele interschimbate intre utilizatori, precum si numele de utilizator si parola vor fi retinute intr o baza de date. Aceasta baza de date va facilita afisarea istoricului de mesaje prin comanda “istoric”, implementarea functiei reply prin comanda : “reply - mesaj\_id | raspuns” (cu ajutorul variabilei mesaj\_id mesajul asupra caruia se aplica reply este identificat), verificari rapide asupra conectarii la un cont sau a adaugarii unuia. Utilizatorul va primi mesajele necitite (mesaje primite atunci cand acesta nu a fost activ), atunci cand se va conecta. Utilizatorul pentru a se deconecta in timp ce vorbeste cu un client va folosi comanda “exit1”.

* 1. Obiectivele Proiectului

Primul obiectiv al proiectului este de implementa o aplicatie client-server ce faciliteaza trimiterea de mesaje in timp real intre clienti. De asemenea, este dorita gasirea unei solutii pentru a transmite si mesajele primite de un utilizator atunci cand utilizatorul curent nu este activ in aplicatie. Implementarea functiei reply reprezinta o alta componenta importanta a programului.

Un alt obiectiv al proiectului este implementarea unui sistem de autentificare. Conectarea la fiecare cont necesita asocierea numelui utilizatorului cu o parola. Aceasta parola este stocata in baza de date a programului. Asocierea va fi verificata, in caz de este incorecta clientului nu ii va fi permisa conectarea la server.

1. Tehnologii Aplicate

### TCP (Transmission Control Protocol). Prin folosirea acestui protocol este asigurata o comunicare sigura intre client si server. Datele ce sunt transmise corect si complet.

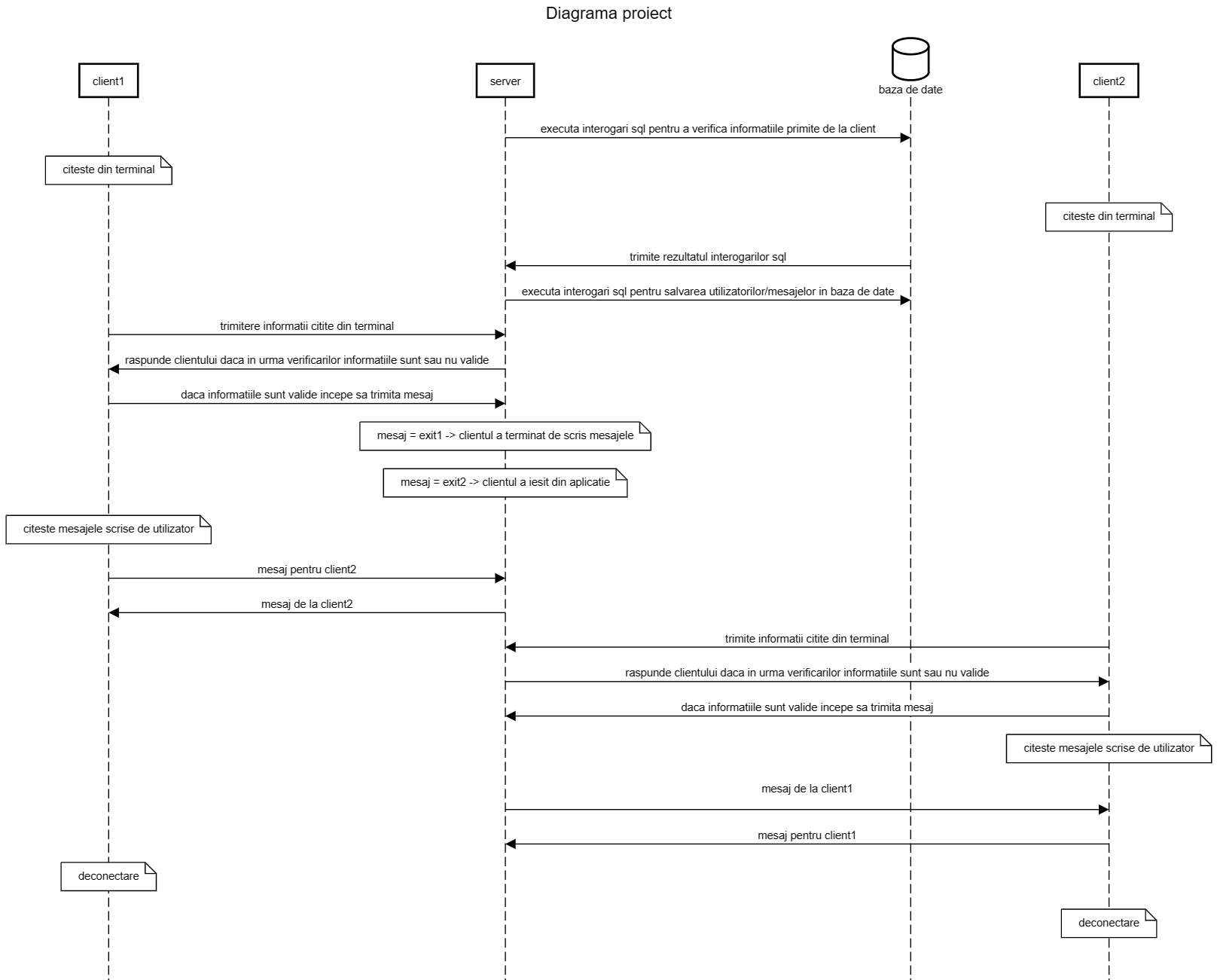
### Socket TCP. Este o componenta folosita pentru comunicarea dintre server si client.

### SQLite. Faciliteaza stocarea informatiilor necesare aplicatiei si ofera o accesibilitate rapida asupra lor.

### Thread. Ajuta la implementarea unui server concurent.

1. Structura Aplicatiei

Proiectul va avea un client, un server si o baza de date. Clientul citeste de la tastatura informatiile date de catre utilizator (siruri de caractere). Aceste informatii sunt mai departe trimise serverului care le verifica si care trimite apoi raspunsul clientului. In functie de rezultatul verificarilor clientul si serverul parcurg bucatile de cod specifice. De exemplu, daca un utilizator nu are cont, dar vrea sa si creeze, va introduce informatiile necesare pe care serverul le va verifica. Daca acestea sunt corecte utilizatorul este inregistrat in baza de date, altfel va fi respins. Odata ce toate verificarile sunt executate (inclusiv numele utilizatorului cu care actualul client doreste sa vorbeasca), clientul poate incepe conversatia.



**Fig. 1.** Diagrama aplicatiei Offline Messsenger.

1. Aspecte de Implementare

Un scenariu real de utilizare : utilizatorul doreste sa se conecteze; acesta are cont deja facut; introduce numele si parola, acestea sunt verificate si I se permite intrarea in aplicatie; utilizatorul introduce numele celui de al doilea participant; acest nume introdus nu se afla in baza de date; utilizatorul este avertizat de acest lucra , iar clientul este deconectat de la server.

Mai jos sunt prezentate 2 verificari facute de server asupra informatiilor primite de la client :

//verificare asociere nume - parola

sqlite3\_stmt \*stmt;

char comanda\_verificare\_user[256];

snprintf(comanda\_verificare\_user, sizeof(comanda\_verificare\_user), "SELECT \* FROM UTILIZATORI WHERE NUME = '%s' AND PAROLA = '%s'", nume, parola);

if(sqlite3\_prepare\_v2(baza\_de\_date, comanda\_verificare\_user, -1, &stmt, 0) != SQLITE\_OK)

{

printf("Eroare la verificare utilizator : %s!\n", sqlite3\_errmsg(baza\_de\_date));

sqlite3\_free(eroare);

sqlite3\_close(baza\_de\_date);

return 4;

}

int exista = 0;

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

{

exista = 1;

}

//verific daca numele celui de al doilea participant (utilizatorul cu care clientul deja //conectat doreste sa vorbeasca) exista in baza de date

int exista = 0;

sqlite3\_stmt \*stmt;

char comanda\_verificare\_nume[256];

snprintf( comanda\_verificare\_nume, sizeof(comanda\_verificare\_nume), "SELECT \* FROM UTILIZATORI WHERE NUME = '%s'", al\_doilea\_participant);

if(sqlite3\_prepare\_v2(baza\_de\_date, comanda\_verificare\_nume, -1, &stmt, 0) != SQLITE\_OK)

{

printf("Eroare la verificare utilizator : %s!\n", sqlite3\_errmsg(baza\_de\_date));

sqlite3\_free(eroare);

sqlite3\_close(baza\_de\_date);

return NULL;

}

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

{

exista = 1;

}

else

{

exista = 0;

}

1. Referinte Bibliografice
2. <https://sequencediagram.org>
3. <https://www.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines>
4. https://profs.info.uaic.ro/~computernetworks/