

ConsoleShopper

Timon Roxana Mihaela

January 2022

1 Introducere

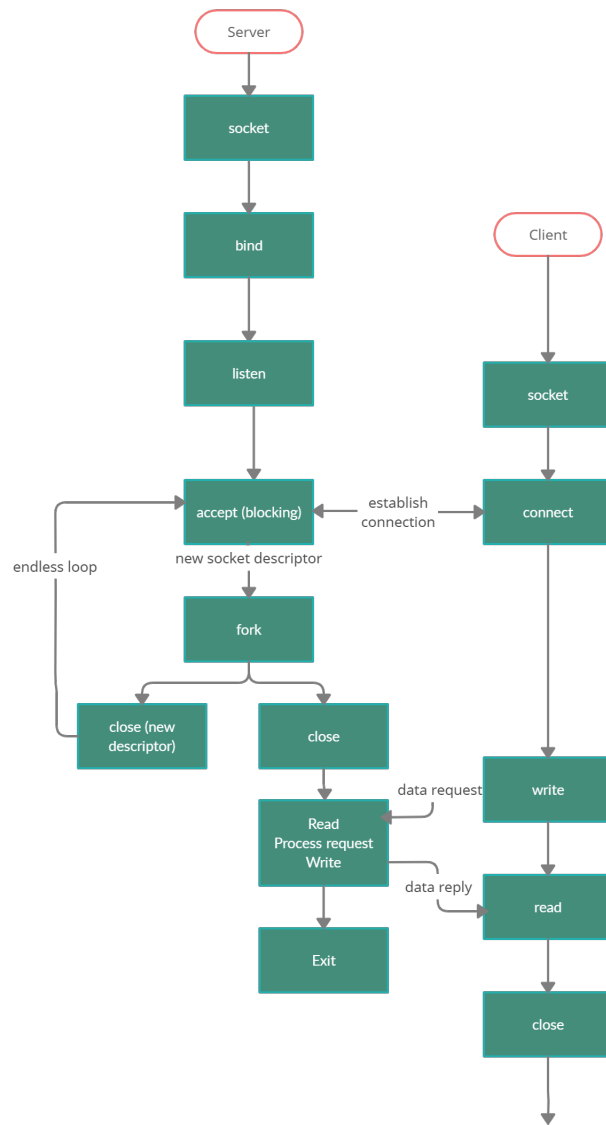
Acest document are rolul de a prezenta documentația corespunzătoare proiectului ConsoleShopper. Proiectul are la bază o aplicație client - server și este realizat folosind limbajul de programare C. Aplicația implementată oferă funcționalitatea unui magazin online de produse. Aceasta suportă autentificarea utilizatorilor pe baza unor conturi definite, organizarea produselor în diverse categorii, afișarea acestora, plasarea de comenzi, salvarea coșului de cumpărături etc.

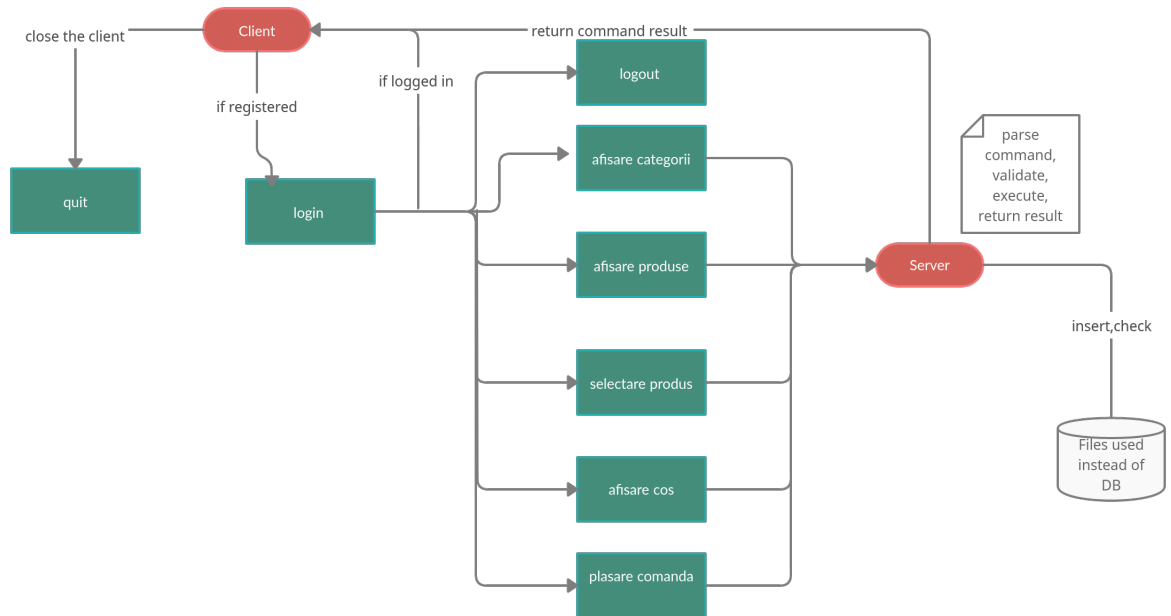
2 Tehnologii utilizate

Pentru realizarea acestui proiect am folosit mecanismul de comunicare TCP (Transmission Control Protocol) concurrent. Acest protocol de comunicare orientat pe conexiunea permite conectarea mai multor clienți la server. TCP este un protocol care face parte împreună cu UDP din nivelul 4 al modelului OSI, mai exact nivelul Transport. O parte din caracteristicile corespunzătoare acestui protocol sunt : se asigură de integritatea datelor transmise, în cazul în care unele pachete cu date se pierd "pe drum" acesta asigură retransmiterea datelor, ordonează pachetele ajunse la destinație, stabilește o conexiune între client și server. Serverul TCP va crea câte un proces copil prin intermediul instrucțiunii `fork()`. Serverul va avea rolul de a înregistra date de la client prin intermediul socket-ului corespunzător, de a le procesa , actualiza, și de a transmite informații cerute înapoi la client. Clientul la rândul său păstrează o conexiune cu serverul, recepționează mesaje de la server, le procesează , preia comenzi de la tastatură. Comunicarea dintre server și client va putea fi observată mai bine prin intermediul digramei de la punctul 3. Pentru stocarea datelor cu privire la utilizatorii aplicației și produse ne vom folosi de fișiere.txt sau de configurare, în care vom memora date despre utilizatori, userii acceptați, respectiv date despre produsele din stoc, grupate pe categorii, cosul de cumpărături pentru fiecare utilizator în parte. În principiu la baza aplicației se folosesc metode de operare (prelucrare) cu fișiere.

3 Arhitectura aplicației

Arhitectura aplicației am realizat-o după modelul TCP concurrent. Concepte implicate: server, client, utilizator, fișiere folosite cu scop de baze de date.





4 Detalii implementare

În ceea ce privește implementarea, la nivelul serverului vom folosi mai multe funcții cu diferite atribuții. În funcția main stabilim toate caracteristicile structurii ce indică la adresa serverului, respectiv a clientului, verificăm argumentele primite la linia de comandă și tratăm erorile, creăm un socket, și îl atașăm la adresa serverului. Totodată realizăm o buclă infinită în care acceptăm pe rând cererile de clienți. La conexiunea cu clientul prin intermediul primitivei accept vom reține informațiile despre client și vom obține file-descriptorul specific socketului cu care vom comunica cu clientul, vom prelua comanda (mesajul) pe care acesta ni-l transmite și o vom prelucra. Fiecare mesaj primit reprezintă o comandă pe care serverul o va parsea, iar în cazul în care este validă, o va executa, va face modificările în rigoare și va trimite clientului un mesaj specific. În cazul în care comanda este gresită, clientul va fi anunțat. Funcția de login are rolul de a loga un utilizator, doar dacă username-ul acestuia se află în fisierul de configurare. Odată logat utilizatorul poate executa următoarele comenzi : afișare categorii, afișare produse, afișare categorii, selectare produs, stergere produs, afișare coș, plasare comandă, logout și quit. Scenarii de utilizare: Nicio comandă nu va putea fi executată dacă utilizatorul nu se loghează. Un utilizator al cărui cont nu aparține fisierului de configurare nu va putea să se logheze în aplicație. Utilizatorul poate să selecteze produse care există, din categorii care există, altfel va primi un mesaj de eroare. Comanda poate fi plasată

doar in cazul în care coșul nu este gol. Afisarea coșului va fi posibila doar daca in acesta exista produse. În cazul in care clientul plasează o comanda, cosul va fi golit. Clientul se poate deloga, doar daca este logat, si poate iesi oricand din aplicație folosind functia quit. Exemple de funcții folosite in implementarea aplicației:

The image displays two screenshots of a Visual Studio Code editor window titled 'servTcpConcc - Proiect - Visual Studio Code'. The editor is showing C++ code for a server application. The top screenshot shows the 'plaseareComanda' function, and the bottom screenshot shows the 'afisareProduse' function.

Top Screenshot: 'plaseareComanda' function

```

285 }
286
287 char *plaseareComanda(char nume[])
288 {
289     char *result;
290     //verificam daca exista cosul de cumparaturi
291     char folder_name[15] = {};
292     char extension[] = ".txt";
293     char cos[] = "cos";
294     long lSize;
295     char *buffer;
296     FILE *fptr1, *fptr2;
297     char ci;
298     char cos_folder_name[20] = {};
299
300     printf("[server]Am intrat in functia de plaseare comanda\n");
301     printf("[server]Numele utilizatorului este:%s\n", nume);
302     strcpy(folder_name, nume);
303     strcat(folder_name, extension);
304     printf("[server]Numele fisierului cautat este: %s\n", folder_name);
305
306     //cream un alt fisier de tipul numecos.txt
307     strcpy(cos_folder_name, nume);
308     strcat(cos_folder_name, cos);
309     strcat(cos_folder_name, extension);
310     printf("[server]Numele noului fisier care se va crea este:%s\n", cos_folder_name);
311
312     if ((fptr1 = fopen(folder_name, "r")) < 0)
313         perror("Eroare la citirea din fisier");
314     if (NULL != fptr1)
315     {
316         fseek(fptr1, 0, SEEK_END);
317         int size = ftell(fptr1);
318         if (0 == size)
319         {
320             printf("file is empty\n");
321             return "Me pare rau, dar cosul dumneavoastra de cumparaturi este gol.";
322         }
323         //...
324     }

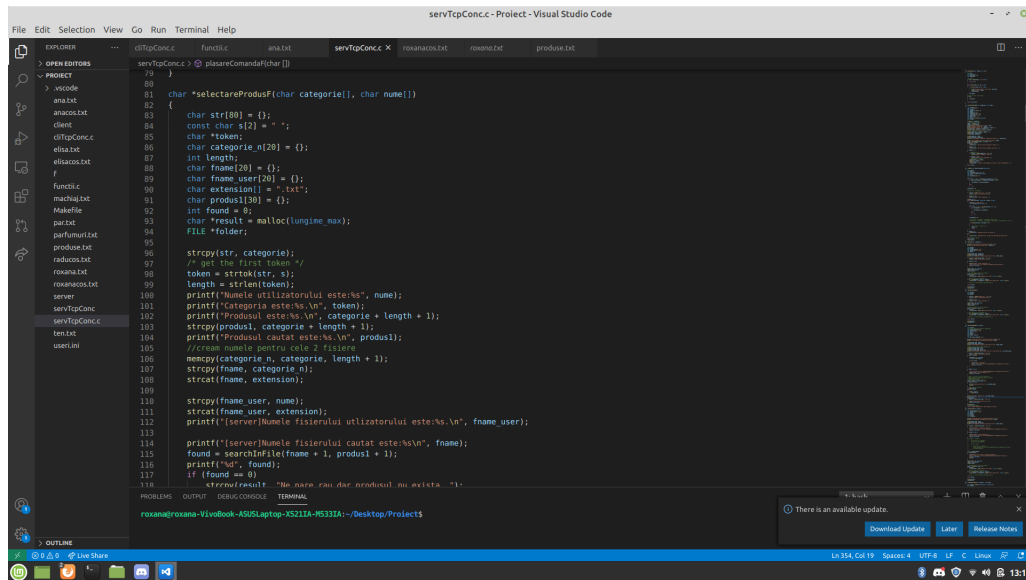
```

Bottom Screenshot: 'afisareProduse' function

```

252 free(buffer);
253 }
254 char *afisareProduse()
255 {
256     char *buffer;
257     FILE *folder;
258     long lSize;
259
260     if ((folder = fopen("produse.txt", "r")) < 0)
261         perror("Eroare la citirea din fisier");
262
263     if (folder == NULL)
264         perror("Eroare, fisier inexistent.");
265
266     fseek(folder, 0L, SEEK_END);
267     lSize = ftell(folder);
268     rewind(folder);
269
270     /* allocate memory for entire content */
271     buffer = calloc(1, lSize + 1);
272     if (!buffer)
273         fclose(folder), fputs("memory alloc fails", stderr), exit(1);
274
275     /* copy the file into the buffer */
276     if (!fread(buffer, lSize, 1, folder))
277         fclose(folder), free(buffer), fputs("entire read fails", stderr), exit(1);
278     /* buffer is a string contains the whole text */
279
280     return buffer;
281
282     fclose(folder);
283     free(buffer);
284 }
285
286 char *plaseareComanda(char nume[])
287 {
288     char *result;
289     //verificam daca exista cosul de cumparaturi

```



5 Concluzii

Aplicatia ConsoleShopper ar putea fi îmbunătățită prin folosirea bazelor de date în schimbul fișierelor, prin adăugare de noi comenzi sau funcționalități precum, anulare comandă, salvarea unui buget, etichetarea produselor cu pret, calcularea pretului comenzii plasate. La nivelul tehnologiei utilizate, consider ca un server UDP ar aduce beneficii în ceea ce privește viteza de transmitere a datelor, însă în acest mod nu s-ar mai păstra integritatea datelor. În ceea ce privește aspectul aplicației, sunt de părere ca o interfață grafică (GUI) ar fi mai atractivă. Totodată, posibilitatea de înregistrare, ar fi un plus pentru utilizatorii dornici să utilizeze produsul nostru software.

6 Bibliografie

Site-uri utilizate pentru realizarea proiectului:

- <https://profs.info.uaic.ro/computernetworks/>
- <https://www.youtube.com/watch?v=cNdlrbZSkyQt=845s>
- <https://ro.wikipedia.org/wiki/TransmissionControlProtocol>
- <https://profs.info.uaic.ro/gcalancea/Laboratorul7.pdf>