UNIVERSITATEA ALEXANDRU IOAN-CUZA
FACULTATEA DE INFORMATICĂ

Raport tehnic - proiect LocalMarketPlacePlatform

Student name: *Ungurean Ana-Maria 3A4*

Course: *Rețele de calculatoare*
Due date: *January 9, 2024*

LocalMarketplacePlatform (B)

Dezvoltați o platformă locală de tip marketplace, care permite utilizatorilor să cumpere și să vândă produse în comunitatea lor, folosind un server central pentru gestionarea tranzacțiilor și a listelor de produse. Se va dezvolta un client care sa poată oferi minim 10 operații (e.g. vizualizarea produselor, crearea de oferte, vizualizarea unui istoric al achizițiilor realizate et.al.)

Sus

1. Introducere

Aplicația LocalMarketplacePlatform constituie o posibiltate de digitalizare a cumpărăturilor din comunitatea locală, oferind utilizatorilor posibilitatea de a cumpăra și vinde produse. Având la bază o arhitectură de comunicare client-server, asigură o gestionare a produselor oferite de vânzătorii locali, memorând totodată tranzacțiile realizate. Astfel, prin accesarea aplicației, utilizatorii au acces facil la cumpărături, platforma oferindule și alte beneficii precum: returnarea unui produs sau vizualizarea celor mai cumpărate produse.

2. Tehnologii utilizate

- Implementarea aplicației a fost realizată utilizând limbajul de programare C.
- Protocolul TCP (Transmission Control Protocol) a fost folosit din punct de vedere a protocolului de comunicare având următoare avantaje:
  - este un protocol de transport orientat către conexiune, oferând calitate maximă în comunicarea între client și server
  - se distinge de protocolul UDP prin capacitatea sa de a preveni pierderea de informații
  - permite transmiterea și recepționarea datelor simultan în ambele direcții
  - se remarcă prin capacitatea sa de a menține secvența corectă a datelor, garantând astfel că pachetele ajung în ordine la destinatar
- Comunicarea între server și client se realizează cu ajutorul primitivelor "read" și "write". Acestea manipulează datele pe canalele de comunicare, cum ar fi socketpair-urile. Funcția "read" este folosită pentru a citi date de pe canal, iar

funcția "write" pentru a transmite date pe canal. Socketpair-urile sunt mecanisme bidirecționale care conectează serverul și clientul, facilitând schimbul de date între cele două părți.

- Din punct de vedere al stocării datelor (numele de utilizatori, tranzacții, produse etc.), am folosit baza de date Sqlite3. Astfel, prin utilizarea interogărilor putem accesa baza de dată și executa comenzile dorite de client.
- Am ales să folosesc thread-uri în loc de fork pentru gestionarea proceselor, deoarece thread-urile partajează eficient resursele și comunică mai simplu, fiind mai puțin consumatoare de timp și memorie.

2. Arhitectura aplicației

Aplicația LocalMarketplacePlatform este structurată în două părți server și client, iar comunicarea dintre aceste două, cum am mai precizat, se va realiza prin mecanisme ce permit comunicarea pentru a se trimite corespunzător fluxul de mesaje și informații. Protocolul aplicației cuprinde următoarele comenzi, care vor fi citite de la tastatură în client :

- **1.Login**
  Utilizatorul se poate autentifica în aplicație, având acces la meniul de bază al aplicație în funcție de rolul acestuia: seller or buyer.
- **2. Register**
  Prin intermediul acestei comenzi, utilizatorul își poate crea un cont nou pe baza unui username unic, o parolă și specificarea statului.
- **3. Exit**
  Utilizatorul poate ieși din aplicație. Utilizatorul poate vedea o listă de produse disponibile, cu detalii precum nume, preț și stoc.
- **4. Add a new product**
  Comanda accesibilă doar pentru seller. Se poate adăugă un produs nou specificându-se numele, categoria, stocul, cantitatea, prețul, dar și unitatea de măsură a produsului.
- **5. Edit a product**
  Comanda accesibilă doar pentru seller. Se poate edita orice caracteristică a produsului doar dacă utilizatorul a introdus acel produs.
- **6. Delete a product**
  Comanda accesibilă doar pentru seller. Se poate șterge orice caracteristică a produsului doar dacă utilizatorul a introdus acel produs.
- **7. See your products**
  Comanda accesibilă doar pentru seller. Utilizatorul poate vedea produsele introduse de acesta.
- **8. See all products**
  Utilizatorul poate vedea toate produsele și informațiile despre acestea.

- **9. See your sales**
  Comanda accesibilă doar pentru seller. Utilizatorul poate vedea vânzările făcute și suma câștigată.
- **10. See the best sellers**
  Utilizatorul poate vedea cei mai buni vânzători din LocalMarket.
- **11. See the most sold products**
  Utilizatorul poate vedea cele mai bine vândute produse din LocalMarket.
- **12. Complete profile**
  Utilizatorul poate să-și updateze profilul prin adăugarea numelui,prenumelui, numărului de telefon și a unei descrii.
- **13. See the profile of a user**
  Utilizatorul poate vedea profilului unui utilizator.
- **14. Logout**
  Are loc delogarea utilizatorului.
- **15. Buy a product**
  Comanda accesibilă doar pentru buyer. Utilizatorul poate cumpăra un produs, specificând id-ul acestuia și cantitatea dorită.
- **16. View a history of purchases made**
  Comanda accesibilă doar pentru buyer. Utilizatorul poate vedea un istoric al cumpărăturilor făcute și suma cheltuită.
- **17. Find a product by category**
  Comanda accesibilă doar pentru buyer. Utilizatorul poate căuta un produs specificând categoria dorită.
- **18. Find a product by price**
  Comanda accesibilă doar pentru buyer. Utilizatorul poate căuta un produs specificând prețul minim și prețul maxim.
- **19. Return a product**
  Comanda accesibilă doar pentru buyer. Utilizatorul poate returna un produs dacă nu au trecut 14 zile de la achiziționarea acestuia.

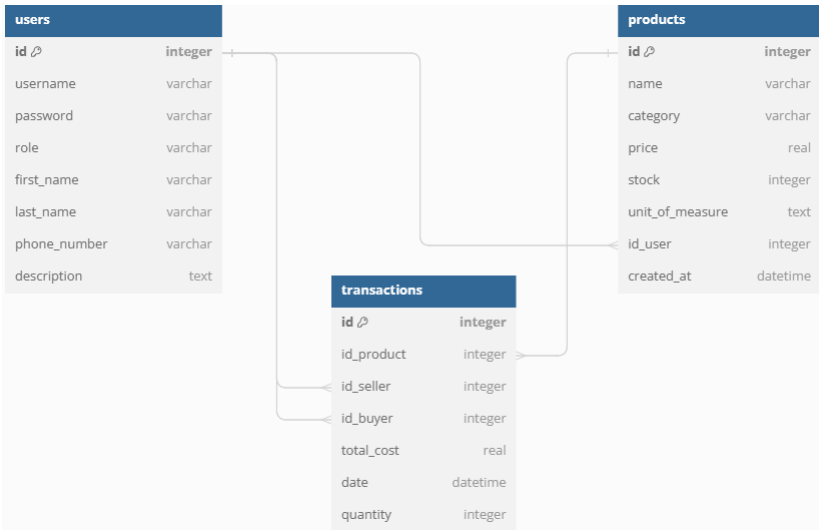

Fig.1 Structura bazei de date

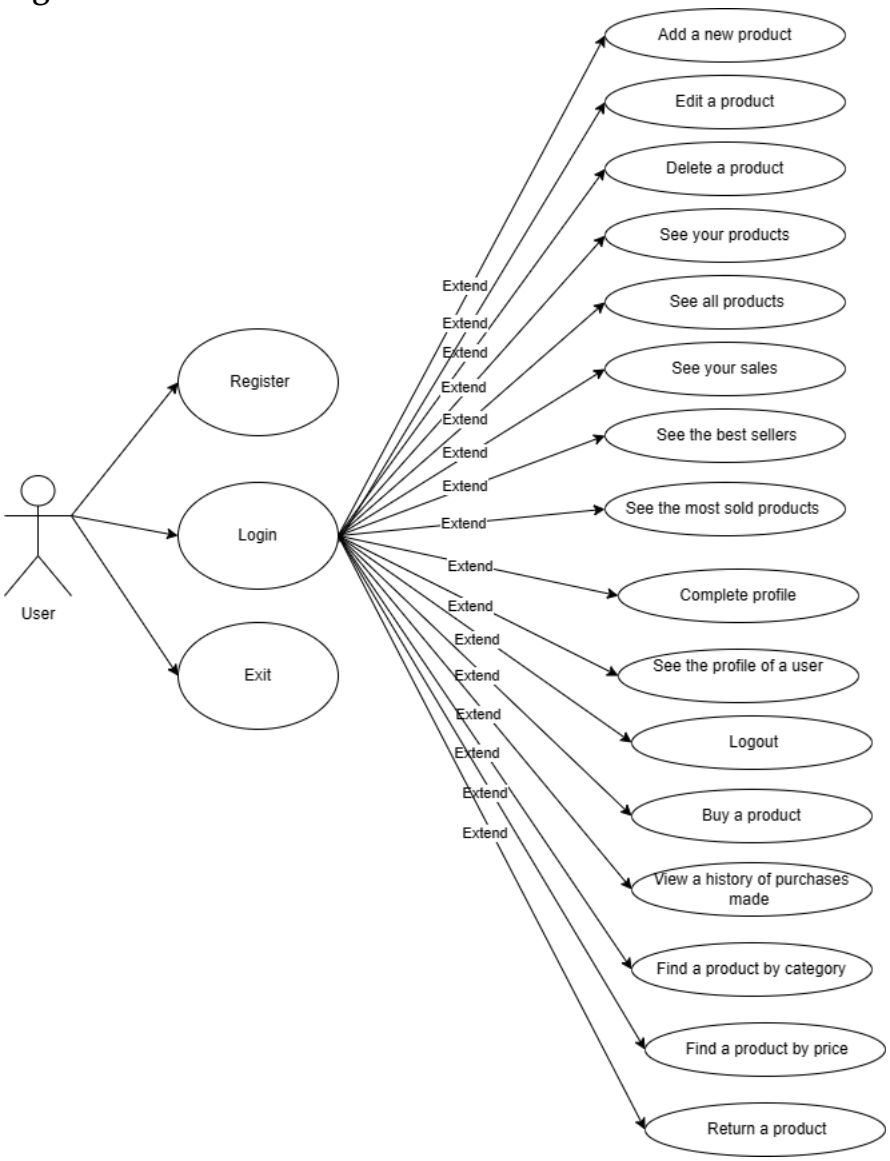

Fig.2 Diagrama Use Case

4. Detalii de implementare

Pentru a facilita comunicarea între client și server, vom utiliza mecanismul de comunicare cunoscut sub denumirea de **socket**. Alegerea acestui mecanism se datorează beneficiilor sale, printre care:

- Permite comunicarea bidirecțională între două procese fără necesitatea implementării unor mecanisme separate de sincronizare.
- Fiecare proces poate să scrie și să citească de la același descriptor, evitând necesitatea creării a două canale de comunicare distincte.
- Nu este necesară închiderea manuală a capetelor canalului de transmisie.

```
if ((sd = socket (AF_INET, SOCK_STREAM, 0)) == -1)
  {
    perror ("[server]Eroare la socket().\n");
    return errno;
  }
```

Fig.3 Crearea unui socket

- **Clientul** se va conecta la Server prin intermediul primitivei connect(), iar apoi va trimite comenzi la server, iar dupa fiecare comandă trimisă se va aștepta răspunsul de la server.

```
if (connect (sd, (struct sockaddr *) &server,sizeof (struct sockaddr)) == -1)
  {
    perror ("[client]Erorr connect().\n");
    return errno;
  }
```

Fig.4 Primitiva connect

- Pentru a organiza datele despre fiecare utilizator, am optat să implementez o bază de date Sqlite3.

```c
void initialize_database(Database *db) {
    db->db = NULL;
}

int open_database(Database *db, const char *db_name) {
    return sqlite3_open(db_name, &db->db);
}

void close_database(Database *db) {
    if (db->db != NULL) {
        sqlite3_close(db->db);
        db->db = NULL;
    }
}
```

Fig.5 Functii pentru conectarea la baza de date și folosirea ei

- **Server-ul** utilizează **thread-uri** pentru a gestiona concurența. La fiecare conexiune de la un utilizator, un nou thread este creat pentru a procesa comenzile respective ale acelui utilizator. Această abordare asigură o gestionare eficientă a cererilor simultane și optimizează performanța serverului.

```c
while (1)
  {
    int client;
    thData * td;
    int length = sizeof (from);

    printf ("[server]Waiting at the port  %d...\n",PORT);
    fflush (stdout);

    if ( (client = accept (sd, (struct sockaddr *) &from, &length)) < 0)
      {
        perror ("[server]Eroare la accept().\n");
        continue;
      }



      td=(struct thData*)malloc(sizeof(struct thData));
      td->idThread=i++;
      td->cl=client;

      pthread_create(&th[i], NULL, &treat_client, td);

      }
```

Fig.6 Servirea concurentă a clienților

5. Concluzii

- Platforma ar putea fi îmbunătățită prin oferirea unei interfețe grafice pentru o experiență mai plăcută pentru utilizatori.

6. Bibliografie

- https://profs.info.uaic.ro/ computernetworks/cursullaboratorul.php
- https://www.andreis.ro/teaching/computer-networks
- https://www.tutorialspoint.com/sqlite/sqliteccpp.htm