OS MINI PROJECT

BHAVIL SHARMA
IMT2021041

GITHUB LINK:-https://github.com/Bhavil-13/OS_Ecommerce_Project

This is a server-client based C project on an E-Commerce Store made using some concepts taught to us in Operating System Course. The communication between the user and admin is done using socket programming. File locking is also used.

RUNNING THE CODE:

Step-1: Run these commands in a terminal:

```
$ gcc -o server connect_socket.h connect_socket.c locks.h locks.c product.h product.c
server_admin.h server_admin.c server_user.h server_user.c server.h server.c

$ ./server
```

This will start the server.

Step-2: Now, in a different terminal, run these commands:

```
$ gcc -o client client_admin.h client_admin.c client_user.h client_user.c client.c
connect_socket.h connect_socket.c login.h menues.h menues.c product.h product.c
read_write_shm.h read_write_shm.c sending_requests.h shm.h shm.c locks.h locks.c

$ ./client
```

This will start the client side application and connect it to the server

Step-3: Now, navigate through the program.

USE OF DIFFERENT FILES:

- Client_user.c and client_admin are subsets of client.c.
- Client.c has the main for the frontend of the project, which shows the menu with the help of menue.c file.
- The connect_socket.c is used for making the socket connections. It has functions that make the connections on the server side and on the client side.
- The product.c has the product and cart structure. It also has function that help in taking input of the product, and showing a product(codes for both server and client side).
- The sending_requests.c, read_write_shm.c and shm.c were not used, they were originally intended for uses that I never implemented. The read_write_shm.c and shm.c are there to help with reading and writing to a shared memory.

- The locks.c has all the different types of locks used in the server side.
- The server_user.c and server_admin.c are subsets of server.c
- The server.c is the backend of this project.
- All the text files are there for storing data.

OS CONCEPTS USED IN THE PROJECT:

- **Socket Programming :-**
    1. Server side:- socket(), bind(), listen(), accept()
    2. Client side:- socket(), connect()
    3. Read and Write are also a part of it. They are blocking system calls.

- **File Locking :-**
    1. Read only lock
    2. Write only lock
    3. It was done using the flock structure.
- **Shared Memory :-**
    1. Writing and Reading to shared memory:- ftok(), shmget(), shmat(), shmdt()
    2. Destroying a memory block:- shmctl()
- **File Handling :-**
    1. Open, read and write
    2. Using various permissions, like read and write only, etc.

CLIENT SIDE:

- Client has 2 files, one for user and one for admin.
- The user file has the following functions:
    1. get_cart(int user_ID, int sock_fd)
    2. add_item_to_cart(int user_ID, int sock_fd, struct product prod)
    3. update_cart(int sock_fd, int user_ID)
    4. pay_for_cart(int sock_fd, int user_ID)
    5. generate_reciept(struct cart user_cart)
    6. register_customer(int sock_fd)
- The admin file has the following functions:
    1. add_product(int sock_fd)
    2. delete_product(int sock_fd, int P_ID)
    3. update_price(int sock_fd)
    4. update_quantiy(int sock_fd)
- These functions send various requests to the server side using write() and the server gets things done.

SERVER SIDE:

- Server also has 2 files, one for the user and one for admin.
- The user file has the following functions:
    1. post_cart(int sock_fd, int cart_fd, int user_fd)
    2. add_product_to_cart(int sock_fd, int cart_fd, int record_fd, int user_fd)
    3. edit_cart(int sock_fd, int cart_fd, int record_fd, int user_fd)
    4. post_pe(int sock_fd, int cart_fd, int record_fd, int user_fd)
    5. add_user(int sock_fd, int cart_fd, int user_fd)
- The admin file has the following functions:
    1. add_products(int sock_fd, int admin_fd, int records_fd)
    2. delete_products(int sock_fd, int admin_fd, int records_fd, int P_ID)
    3. generate_receipt(int admin_fd, int records_fd)
    4. update_product_quantity(int sock_fd, int admin_fd, int records_fd)
    5. update_product_cost(int sock_fd, int admin_fd, int records_fd)
- These functions first get requests from the client and then they do their work and send back a response.
- This is like a middle-layer between the client side and the data-base(.txt files). The server side functions deal with the database and fetches the data from it and sends relevant data to the client.