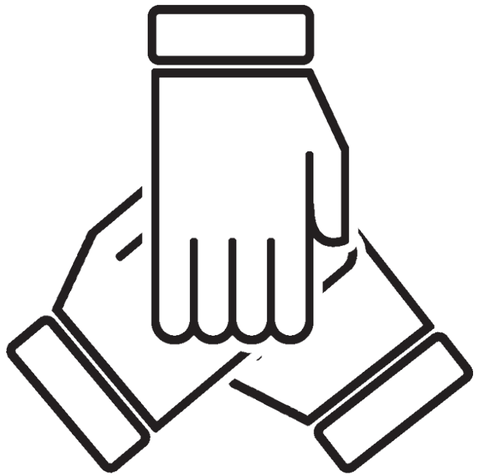
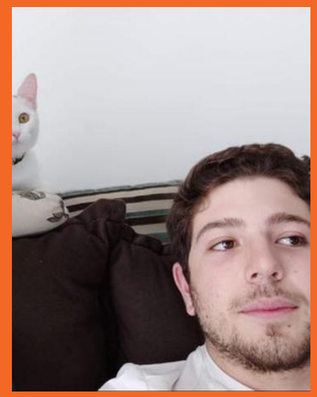
Projeto de pesquisa

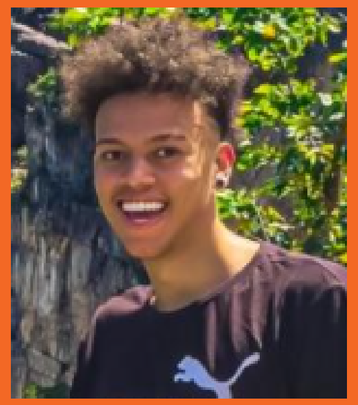
Criando ambientes virtuais de conversação com uso system call select()

Integrantes



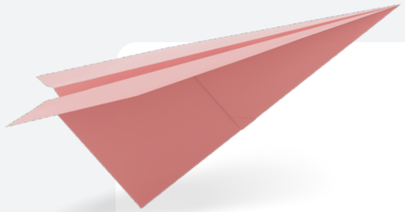


Lucas Lima
18/0105345



Jonathan Jorge
18/0103580

Júlia Farias
18/0103792



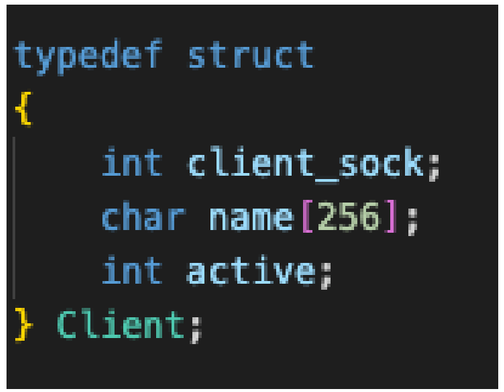
Proposta

Este projeto tem como objetivo, permitir que o aluno compreenda a arquitetura de aplicações de rede (segundo arquitetura TCP/IP) que envolvam envolvam gerência de diálogo. Para isso, devem construir uma aplicação que disponibilize salas de bate-papo virtuais, nas quais os clientes podem ingressar e interagir

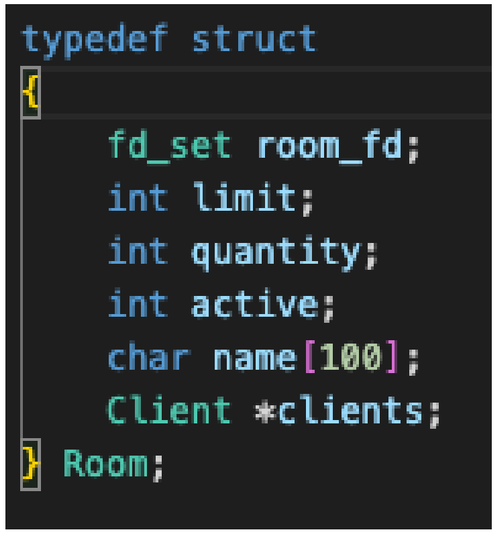
Chamada Select()

A chamada select() monitora a atividade em um conjunto de sockets procurando por sockets prontos para leitura, gravação ou com uma condição de exceção pendente.

Servidor



A figura acima corresponde a struct de clientes, onde encontramos os dados referentes a identificação dele.



A figura acima corresponde a struct de sala.

Servidor

```
// Socket configuration
int sock = socket(AF_INET, SOCK_STREAM, 0);
setsockopt(sock, SOL_SOCKET, SO_REUSEADDR, &yes, sizeof(int));

myaddr.sin_family = AF_INET;
myaddr.sin_addr.s_addr = inet_addr(argv[1]);
myaddr.sin_port = htons(atoi(argv[2]));
memset(&(myaddr.sin_zero), 0, 8);
bind(sock, (struct sockaddr *)&myaddr, sizeof(myaddr));
listen(sock, 10);

// Adding descriptors files
FD_SET(sock, &master);
FD_SET(STDIN, &master);
fdmax = sock;
addrlen = sizeof(remoteaddr);
return sock;
```

Antes da main, primeiro configuramos o servidor e depois inicializamos os métodos de select.

```
int main(int argc, char *argv[])
{

    if (argc < 3)
    {
        printf("Enter the IP and port of the server\n");
        // show format
        printf("With the format xxx.x.x.x xxxx");
        exit(1);
    }
    int sock = initialize_server(argv);

    printf("Server initialized!\n");
    int room;

    for (;;)
    {
        // Inform that the master will receive read descriptors and perform the select operation
        read_fds = master;
        select(fdmax + 1, &read_fds, NULL, NULL, NULL);
        for (int i = 1; i <= fdmax; i++)
        {
            // Test to see if the file descriptor is in the basket
            if (FD_ISSET(i, &read_fds))
```

Já na main faremos o direcionamento dos fluxos de criação de grupose e algumas validações

Cliente

```c
int main(int argc, char *argv[])
{
  if (argc < 3)
  {
    printf("Usage: %s <server IP> <server port>\n", argv[0]);
    exit(1);
  }

  int sock = socket(AF_INET, SOCK_STREAM, 0);
  struct sockaddr_in serverAddr;
  serverAddr.sin_family = AF_INET;
  serverAddr.sin_port = htons(atoi(argv[2]));
  inet_pton(AF_INET, argv[1], &(serverAddr.sin_addr));

  if (connect(sock, (struct sockaddr *)&serverAddr, sizeof(serverAddr)) == -1)
  {
    perror("Failed to connect");
    exit(1);
  }
  fd_set read_fds, master;
  FD_ZERO(&master);
  FD_ZERO(&read_fds);
  FD_SET(STDIN_FILENO, &master);
  FD_SET(sock, &master);
  int fdmax = (STDIN_FILENO > sock) ? STDIN_FILENO : sock;
```

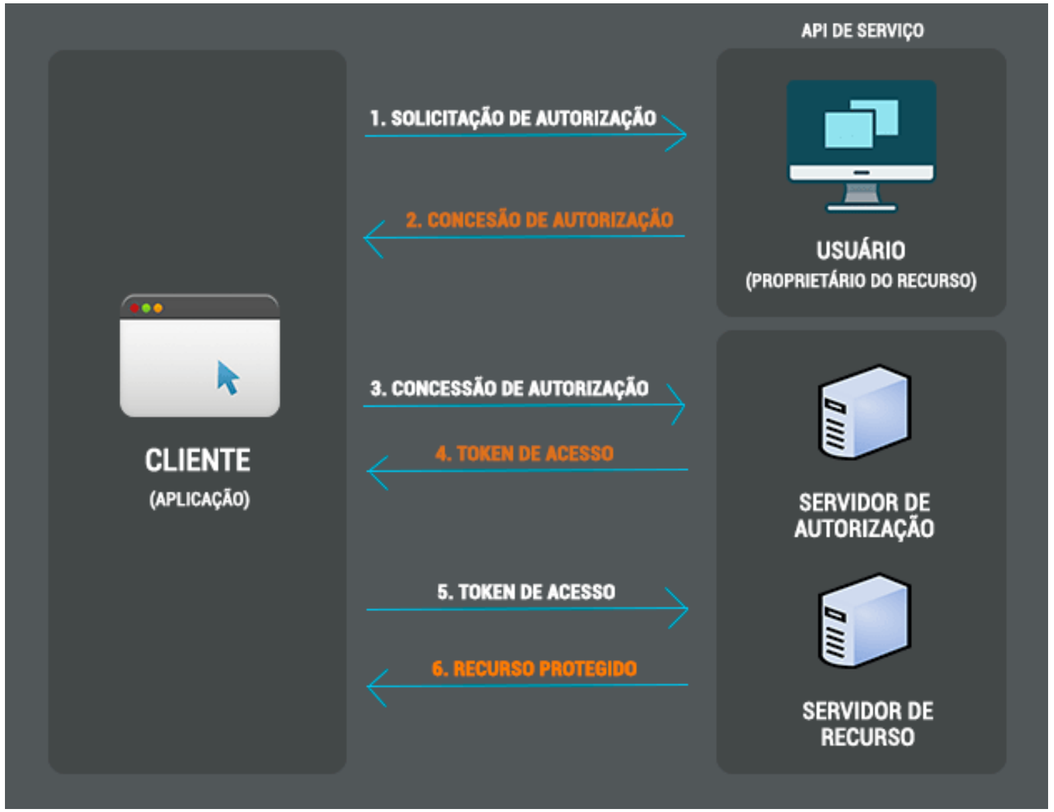
A distribuição da mensagem acontece para todos os descriptors dentro do mesmo cesto ou grupo de select, por isso deve-se saber antes o id do cliente que está transmitindo a mensagem através do socket.

```c
char nome[256];
printf("Enter your name: ");
scanf("%s^\n", nome);
nome[strcspn(nome, "\n")] = '\0';
send(sock, nome, strlen(nome), 0);

int sala_id;
printf("Enter the room ID (-1 to create a new room): ");
scanf("%d", &sala_id);
snprintf(buff, sizeof(buff), "%d", sala_id);
send(sock, buff, strlen(buff), 0);
clear_input_buffer();

if (sala_id == -1)
{

  int limit;
  char room_name[100];
  printf("Enter the room`s name: ");
  scanf("%s^\n", room_name);

  room_name[strcspn(room_name, "\n")] = '\0';
  send(sock, room_name, strlen(room_name), 0);
  clear_input_buffer();

  printf("Enter the room limit: ");
  scanf("%d", &limit);
  clear_input_buffer();

  printf("limit %d\n", limit);
```

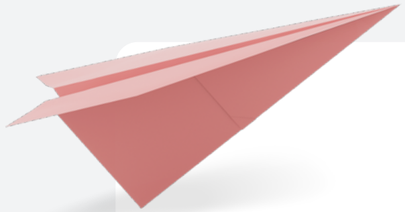
Inserindo as informações do cliente como nome e número da sala conseguimos identificá-lo, depois digitamos “-1” para que o programa peça o limite de pessoas na sala e logo em seguida ela será criada

OAuth2.0

- OAuth2.0 é amplamente utilizado na indústria para permitir que os usuários autentiquem-se em aplicativos por meio de provedores de identidade conhecidos, como Google, Facebook, Twitter, entre outros. Isso elimina a necessidade de os usuários criarem novas credenciais para cada aplicação e simplifica o processo de login.



Fonte: https://www.treinaweb.com.br/blog/o-que-e-oauth-2



Conclusão

O programa final torna possível a comunicação entre os usuários nas salas de bate papo, com suas devidas identificações e possui as funções pedidas no enunciado, como listar participantes, realizar a saída de clientes, assim como o diálogo entre eles.

Obrigado !

30 de Junho de 2023