UNIP
UNIVERSIDADE PAULISTA

5º Semestre Ciência da Computação (CC)

Atividade Prática Supervisionada

Tema: "DESENVOLVIMENTO DE UMA FERRAMENTA PARA COMUNICAÇÃO EM REDE."

Alunos participantes:

Filipi Yukio Iwakami Itoyama, RA – N4453J-1

William Rossi do Carmo Ruiz, RA - N473GF-8

Luiz Gabriel Zeferino Duarte, RA – N454CD-8

Lucas de Oliveira Brandolezi, RA – D9380G-9

Índice

1. Objetivo do Trabalho

Têm-se como objetivo deste trabalho, o desenvolvimento de uma ferramenta para efetuar uma comunicação que funciona em rede. Para isso, utilizando-se de uma linguagem de programação conhecida como “Python”, e alguma de suas bibliotecas famosas para criarmos e mantermos uma conexãos em rede estável, que são o “socket” e o “threading”. Neste trabalho, iremos criar um “chat” em rede, mas que funciona somente em LAN, ou seja, uma rede local.

2. Introdução

Para iniciarmos este trabalho, primeiro precisamos entender que, este chat não vai conseguir iniciar, e muito menos manter uma conexão via “Internet”, mas sim somente via LAN, ou seja, uma rede local. Isso não quer dizer que não podemos nos comunicar usando esse chat com pessoas de locais diferentes, mas para que isso seja possível, ambas partes precisam de “Conectar” em uma rede local, gerada através de programas de terceiros, o mais popular hoje em dia é o Radmin VPN. Desta forma, ambas partes entrando na rede local gerada por esse programa, pode-se manter uma conexão em rede e consequentemente utilizar-se deste chat criado por nós para se comunicarem.

3. Fundamentos da comunicação de dados em rede

Na primeira década dos sistemas computacionais, os computadores eram altamente centralizados, as empresas e universidades possuíam apenas um ou dois computadores e as grandes instituições algumas dezenas. Todos eles eram isolados, não existia nenhuma comunicação entre eles.

Nos anos 70 e 80 começaram a desenvolver as redes de comunicação de dados, que são redes em que a informação circula sob a forma binária. Sendo os computadores máquinas que manipulam informação na forma binária, as redes digitais são adequadas à transferência de informação entre essas máquinas.

O compartilhamento de tais dados se dá por meio de protocolos de rede (conjunto de normas que permitem que qualquer máquina conectada à internet possa se comunicar com outra também já conectada na rede).

- Acesso simultâneo a programas e dados importantes;
- Permitir às pessoas compartilhar dispositivos periféricos;
- Facilitar o processo de realização de cópias de segurança (backup);
- Facilitar as comunicações pessoais com o correio eletrônico ou mensagens instantâneas.

Estes são alguns dos principais benefícios oferecidos pela comunicação em rede.

Dentre os diversos tipos de redes, existe o LAN (Local Area Network - Rede Local de Computadores), no qual são redes usadas em áreas pequenas (tipicamente um edifício ou conjunto de edifícios) e operam a velocidades elevadas (da ordem das dezenas de Mbit/s).

4. Plano de desenvolvimento da aplicação

Para fazermos esse código usamos o Python, já que todos os membros aprenderam essa linguagem no segundo semestre do curso de Ciências da Computação, é uma linguagem que suporta tanto a programação orientada a objetos quanto a programação estruturada, fora que ela é uma linguagem que está em alta, sendo uma das linguagens mais usadas entres os programadores hoje em dia e ele suporta sistemas como Windows, MacOs, Linux, Solaris, Unix e FreeBSD.

Além disso que falamos anteriormente o Python tem uma vasta opção de bibliotecas que nos ajudaram nesse trabalho. Algumas que usamos foram o Socket, que são usados para enviar dados através da rede e o threading, que define a classe Thread, que é um pequeno programa que trabalha como um subsistema, sendo uma forma de um processo se autodividir em duas ou mais tarefas.

Usamos também algumas estruturas simples do Python como o try/except, que serve para o tratamento de exceções, o while, que é uma estrutura de repetição que irá ser executada enquanto a condição for atendida.

Também usamos o comando Def, que serve para definirmos algumas funções e onde terá uma sequencia de comandos e quando precisar, basta chama-la, que ela será executada, basta ter a mesma tabulação.

5. Plano de Desenvolvimento, Projeto e Relatório com as linhas de Código da Aplicação

Este código é dividido em duas partes: o cliente e o servidor. primeiro vamos dar uma olhada no servidor:

Inicialmente definimos uma classe chamada servidor, e definimos como comportamento inicial chamar o método start_server. Esse método era cuidar de toda a conexão que o server precisará fazer com os clientes como, por exemplo, criar um socket, guardar o número da porta inserida pelo usuário, liga-los através da função bind(), definir o servidor como um listener com a função (listen) e criar uma lista e um dicionário para guardar informações que serão geradas com as conexões com o cliente.

```
import socket
import threading

class Server:
    def __init__(self):
        self.start_server()

    def start_server(self):
        self.s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

        host = socket.gethostbyname(socket.gethostname())
        port = int(input('Enter port to run the server on --> '))

        self.clients = []

        self.s.bind((host,port))
        self.s.listen(100)

        print('Running on host: '+str(host))
        print('Running on port: '+str(port))

        self.username_lookup = {}
```

A seguir criamos um while infinito que cuidará de receber e endereçar a nova porta e nome dos clientes(nome de usuários) e guarda-las na lista clientes e username_lookup. Logo na sequência criamos uma thread para toda a função handle_client de forma paralela e saímos do while infinito.

```
while True:
    c, addr = self.s.accept()

    username = c.recv(1024).decode()

    print('New connection. Username: '+str(username))
    self.broadcast('New person joined the room. Username: '+username)

    self.username_lookup[c] = username

    self.clients.append(c)

    threading.Thread(target=self.handle_client,args=(c,addr,)).start()
```

Já na função dandle_client, começamos definindo como parâmetros as variáveis c e addr, que guardam as informações das portas dos clientes e, na sequência, abrimos um while infinito. Dentro desse while começamos tentando receber alguma resposta que venha do cliente, se ela não vir, nos cortamos a conexão com aquele socket e removemos o cliente da nossa lista de clientes. Agora, se o servidor receber uma resposta do cliente ele quebra o while infinito e manda a mensagem para todos os outros clientes conectados exceto o que enviou a mensagem se a mesma foi diferente de um espaço vazio.

```
def handle_client(self,c,addr):
    while True:
        try:
            msg = c.recv(1024)
        except:
            c.shutdown(socket.SHUT_RDWR)
            self.clients.remove(c)

            print(str(self.username_lookup[c])+' left the room.')
            self.broadcast(str(self.username_lookup[c])+' has left the room.')

            break

        if msg.decode() != '':
            print('New message: '+str(msg.decode()))
            for connection in self.clients:
                if connection != c:
                    connection.send(msg)

server = Server()
```

Agora no lado do cliente:

Inicialmente começamos definindo uma classe cliente e definimos na sua inicialização que ela chame a função create_connection(). Logo em seguida criamos a função create_connection que inicialmente criará um socket, e depois entrará em um while infinito. Este while cuidará apenas de receber as informações do ip e porta do servidor, se elas não forem informadas o while fica se repetindo e quando forem o while infinito quebra. Saindo do while guardaremos a informação inserida pelo usuário do seu nome de usuário e logo em seguida criamos e inicializamos duas threads paralelas que cuidarão das funções handle_messages e input_handler, respectivamente.

```
import socket
import threading

class Client:
    def __init__(self):
        self.create_connection()

    def create_connection(self):
        self.s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

        while 1:
            try:
                host = input('Enter host name --> ')
                port = int(input('Enter port --> '))
                self.s.connect((host,port))

                break
            except:
                print("Couldn't connect to server")

        self.username = input('Enter username --> ')
        self.s.send(self.username.encode())

        message_handler = threading.Thread(target=self.handle_messages,args=())
        message_handler.start()

        input_handler = threading.Thread(target=self.input_handler,args=())
        input_handler.start()
```

Na sequência definimos a função handle message que terá um while infinito que vai cuidar de sempre receber e mostrar na tela as mensagens dos outros usuários que serão enviados até ele pelo servidor. e na sequência definimos a outra função input_handler que cuidará de imputar uma mensagem escrita pelo usuário e enviá-la para o servidor com o seu nome na frente.

```
    def handle_messages(self):
        while 1:
            print(self.s.recv(1204).decode())

    def input_handler(self):
        while 1:
            self.s.send((self.username+' - '+input()).encode())

client = Client()
```

6. Bibliografia

http://www.goiania.go.gov.br/sistemas/scmag/dados/refautor/refautor21.pdf

https://www.infopedia.pt/$redes-de-comunicacao-de-dados

https://ead.catolica.edu.br/blog/fundamentos-redes-de-computadores

https://www.weblink.com.br/blog/tecnologia/conheca-os-principais-protocolos-de-internet/

https://computerworld.com.br/carreira/python-10-motivos-para-aprender-a-linguagem-em-2019/

7. Fichas de Atividades Práticas Supervisionadas

UNIP UNIVERSIDADE PAULISTA

FICHA ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME:____ Filipi Yukio Iwakami Itoyama — Turma: CC5P28 — RA: N4453J1

CURSO:Ciência da Computação — CAMPUS: São Jos JK - São José do Rio Preto

CÓDIGO DA ATIVIDADE:_____ SEMESTRE:___ 5º — Ano Grade: 3º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
25/05/2019	Apresentação da disciplina	4	Filipi Yukio Iwakami Itoyama	4	
25/05/2019	Orientação da atividade	4	Filipi Yukio Iwakami Itoyama	4	
25/05/2019	Pesquisa Bibliográfica	10	Filipi Yukio Iwakami Itoyama	10	
25/05/2019	Elaboração de texto	10	Filipi Yukio Iwakami Itoyama	10	
25/05/2019	Correções e orientações	4	Filipi Yukio Iwakami Itoyama	4	
25/05/2019	Montagem do grupo no sistema	2	Filipi Yukio Iwakami Itoyama	2	
25/05/2019	Desenvolvimento do protótipo	10	Filipi Yukio Iwakami Itoyama	10	
25/05/2019	Postagem do trabalho	2	Filipi Yukio Iwakami Itoyama	2	
25/05/2019	Apresentação do trabalho	4	Filipi Yukio Iwakami Itoyama	4	

TOTAL DE HORAS ATRIBUÍDA 50

AVALIAÇÃO:_____

Aprovado ou Reprovado

NOTA:_____

DATA:____/_____/_____

_____

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP UNIVERSIDADE PAULISTA

FICHA ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME:____ Luiz Gabriel Zeferino Duarte — Turma: CC5P28 — RA:N454CD8

CURSO:Ciência da Computação — CAMPUS: São Jos JK - São José do Rio Preto

CÓDIGO DA ATIVIDADE:_____ SEMESTRE:___ 5º — Ano Grade: 3º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
25/05/2019	Apresentação da disciplina	4	Luiz Gabriel Zeferino Duarte	4	
25/05/2019	Orientação da atividade	4	Luiz Gabriel Zeferino Duarte	4	
25/05/2019	Pesquisa Bibliográfica	10	Luiz Gabriel Zeferino Duarte	10	
25/05/2019	Elaboração de texto	10	Luiz Gabriel Zeferino Duarte	10	
25/05/2019	Correções e orientações	4	Luiz Gabriel Zeferino Duarte	4	
25/05/2019	Montagem do grupo no sistema	2	Luiz Gabriel Zeferino Duarte	2	
25/05/2019	Desenvolvimento do protótipo	10	Luiz Gabriel Zeferino Duarte	10	
25/05/2019	Postagem do trabalho	2	Luiz Gabriel Zeferino Duarte	2	
25/05/2019	Apresentação do trabalho	4	Luiz Gabriel Zeferino Duarte	4	

TOTAL DE HORAS ATRIBUÍDA 50

AVALIAÇÃO:_____

Aprovado ou Reprovado

NOTA:_____

DATA:____/_____/_____

_____

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME:____ William Rossi do Carmo Ruiz Turma: CC5P28 RA:n473gf8

CURSO:Ciência da Computação CAMPUS: São Jos JK - São José do Rio Preto

CÓDIGO DA ATIVIDADE:_____ SEMESTRE:____ 5º Ano Grade: 3º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
25/05/2019	Apresentação da disciplina	4	William Rossi do Carmo Ruiz	4	
25/05/2019	Orientação da atividade	4	William Rossi do Carmo Ruiz	4	
25/05/2019	Pesquisa Bibliográfica	10	William Rossi do Carmo Ruiz	10	
25/05/2019	Elaboração de texto	10	William Rossi do Carmo Ruiz	10	
25/05/2019	Correções e orientações	4	William Rossi do Carmo Ruiz	4	
25/05/2019	Montagem do grupo no sistema	2	William Rossi do Carmo Ruiz	2	
25/05/2019	Desenvolvimento do protótipo	10	William Rossi do Carmo Ruiz	10	
25/05/2019	Postagem do trabalho	2	William Rossi do Carmo Ruiz	2	
25/05/2019	Apresentação do trabalho	4	William Rossi do Carmo Ruiz	4	

TOTAL DE HORAS ATRIBUÍDA 50

AVALIAÇÃO:_____

Aprovado ou Reprovado

NOTA:_____

DATA:_____/_____/_____

_____

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO

UNIP
UNIVERSIDADE PAULISTA

FICHA ATIVIDADES PRÁTICAS SUPERVISIONADAS - APS

NOME:____ Lucas de Oliveira Brandolezi Turma: CC5P28 RA:D9380G9

CURSO:Ciência da Computação CAMPUS: São Jos JK - São José do Rio Preto

CÓDIGO DA ATIVIDADE:_____ SEMESTRE:____ 5º Ano Grade: 3º

DATA DA ATIVIDADE	DESCRIÇÃO DA ATIVIDADE	TOTAL DE HORAS	ASSINATURA DO ALUNO	HORAS ATRIBUÍDAS (1)	ASSINATURA DO PROFESSOR
25/05/2019	Apresentação da disciplina	4	Lucas de Oliveira Brandolezi	4	
25/05/2019	Orientação da atividade	4	Lucas de Oliveira Brandolezi	4	
25/05/2019	Pesquisa Bibliográfica	10	Lucas de Oliveira Brandolezi	10	
25/05/2019	Elaboração de texto	10	Lucas de Oliveira Brandolezi	10	
25/05/2019	Correções e orientações	4	Lucas de Oliveira Brandolezi	4	
25/05/2019	Montagem do grupo no sistema	2	Lucas de Oliveira Brandolezi	2	
25/05/2019	Desenvolvimento do protótipo	10	Lucas de Oliveira Brandolezi	10	
25/05/2019	Postagem do trabalho	2	Lucas de Oliveira Brandolezi	2	
25/05/2019	Apresentação do trabalho	4	Lucas de Oliveira Brandolezi	4	

TOTAL DE HORAS ATRIBUÍDA 50

AVALIAÇÃO:_____

Aprovado ou Reprovado

NOTA:_____

DATA:_____/_____/_____

_____

CARIMBO E ASSINATURA DO COORDENADOR DO CURSO