

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CORNÉLIO PROCÓPIO
ENGENHARIA DE COMPUTAÇÃO

TERCEIRO RELATÓRIO (FINAL)
MONITORAMENTO DE QUALIDADE DE INTERNET

EC36G – Oficina de Integração

Prof. Dr. Fabricio Martins Lopes

Alunos:

Eduardo Durante Godinho

Matheus Bueno Faria

CORNÉLIO PROCÓPIO

JUNHO/2022

Resumo

O objetivo desse relatório é apresentar um software, o qual foi desenvolvido com o intuito de levantar um monitoramento acerca de uma rede de internet específica. Sua proposta é de coletar dados através de vários testes realizados, e através destes realizar uma rigorosa análise, com a finalidade de informar ao usuário se sua qualidade está boa ou não. Serão explicadas de forma detalhada as funções utilizadas, além indicar várias motivações para a criação do programa, como por exemplo, o uso da internet na pandemia e os requisitos da ANATEL.

Palavras-Chave: internet, testar velocidade, velocímetro.

SUMÁRIO

1. INTRODUÇÃO.....	7
2. OBJETIVOS	9
2.1- Objetivos Gerais.....	9
2.2- Objetivos Específicos	9
3. JUSTIFICATIVA.....	11
3.1- O Uso da internet na pandemia.....	11
3.2- Principais operadoras de internet.....	12
3.3- Requisitos da ANATEL.....	13
4. METODOLOGIA (OU MATERIAIS E MÉTODOS)	15
4.1- Tecnologias usadas	15
4.2- Bibliotecas utilizadas.....	15
4.3- Detalhes do funcionamento inicial.....	16
4.4- Função Jitter	18
4.5- Interface Gráfica.....	19
5. FUNCIONALIDADES E RESULTADOS.....	22
5.1- Resumo das funcionalidades	22
5.2- Funcionalidades Principais.....	22
5.2.1- Realizar um teste.....	22
5.2.2- Velocidade Média	24
5.2.3- Requisitos da ANATEL	25
5.2.4- Análise do ping e jitter.....	28
5.2.5- Resultados.....	29
5.3- Funcionalidades extras	31
5.3.1- Criar tabela	31
5.3.2- Obter IP	33
5.3.3- Monitorar rede	34
5.3.4- Comparar períodos.....	36
6. CRONOGRAMA	38
7. CONCLUSÃO	40
8. REFERÊNCIAS BIBLIOGRÁFICAS.....	41

LISTA DE FIGURAS

Figura 1: Teste sendo realizado no Speedtest.....	7
Figura 2: Tendência do consumo brasileiro em alguns dispositivos eletrônicos.	11
Figura 3: Participação do mercado das operadoras de fibra óptica.	12
Figura 4: Evolução da qualidade da Banda Larga Fixa no Brasil	13
Figura 5: Instalação da biblioteca MySQL.connector	15
Figura 6: Menu de interação inicial do programa	17
Figura 7: Configuração da função conecta_MySQL()	17
Figura 8: Parte inicial do código main	18
Figura 9: Configuração da função Get_Dados_MySQL()	18
Figura 10: Exemplo da função de testar o Jitter	19
Figura 11: Menu final do programa	20
Figura 12: Detalhes da funcionalidade "Requisitos da ANATEL"	20
Figura 13: Exemplo da inicialização de uma janela e configuração dos botões.....	21
Figura 14: Configuração da função Efetuar_Teste()	23
Figura 15: Configuração da função teste_internet()	23
Figura 16: Configuração da função Verificar_Velocidade_Media()	25
Figura 17: Janela referente aos resultados da velocidade média	25
Figura 18: Configuração da função Entrada_Dados_Anatel()	26
Figura 19: Entrada de dados para as velocidades contratadas.....	26
Figura 20: Configuração da função VerificarCondicaoAnatel().....	27
Figura 21: Resultado do requisito de download	27
Figura 22: Resultado do requisito de upload.....	27
Figura 23: Configuração da função Verificar_Conexao_Jogos()	28
Figura 24: Resultado da análise do PING	28
Figura 25: Resultado da análise do JITTER.....	29
Figura 26: Configuração da função Entrada_Dados_Tabela()	29
Figura 27: Entrada de dados para o usuário	29
Figura 28: Configuração da função Mostrar_Tabela()	30
Figura 29: Resultados dos testes da rede móvel 4G da TIM	31
Figura 30: Configuração da função Entrada_Dados_Cria_Tabela().....	32
Figura 31: Configuração da função Criar_Tabela()	32
Figura 32: Comando SQL da função Criar_Tabela().....	33

Figura 33: Configuração da função Verificar_IP()	33
Figura 34: Dados referentes ao IP externo e interno.....	34
Figura 35: Resultados pelo editor MySQL Workbench.....	35
Figura 36: Resultados dos testes no programa.....	35
Figura 37:Configuração da função Melhores_Piores_Horarios().....	36
Figura 38: Resultado dos melhores e piores períodos	37

Lista de Siglas

EAQ – Entidade Aferidora da Qualidade de Banda Larga

ANATEL – Agência Nacional de Telecomunicações

Nº - Número

G1 – Portal de Notícias da Globo

MySQL – My Widenius (filha de Michael Widenus) Structured Query Language

IP – Internet Protocol

API – Application Programming Interface

IPv4 – Internet Protocol version 4

IPv6 – Internet Protocol version 6

TIM - Telecom Italia Mobile

MB – Megabyte

Mbps – Megabits per second / Megabits por segundo

CAT5e – Category 5 enhanced

AVG – Average

1. INTRODUÇÃO

Existem vários tipos de velocímetros para teste de internet, dentre os quais muitos em formato web, como por exemplo, o EAQ. O site, regulado pela ANATEL, permite que o usuário faça testes simultâneos, dos quais são medidas as velocidades de download e upload, além da latência, do jitter e da perda de pacote. Nele são mostrados os resultados, como também a região do servidor e o local onde o teste foi realizado (BRASIL BANDA LARGA, 2017).

Há também o velocímetro denominado Speedtest, famigerado medidor da empresa Ookla e o mais popular do mundo. No site é possível escolher entre diversos servidores espalhados pelo mundo. No teste realizado, são medidas as velocidades de download e upload, assim como a latência (ping). No caso de o usuário possuir uma conta no site, os resultados são armazenados em uma tabela de histórico com os resultados (SPEEDTEST, 2022)

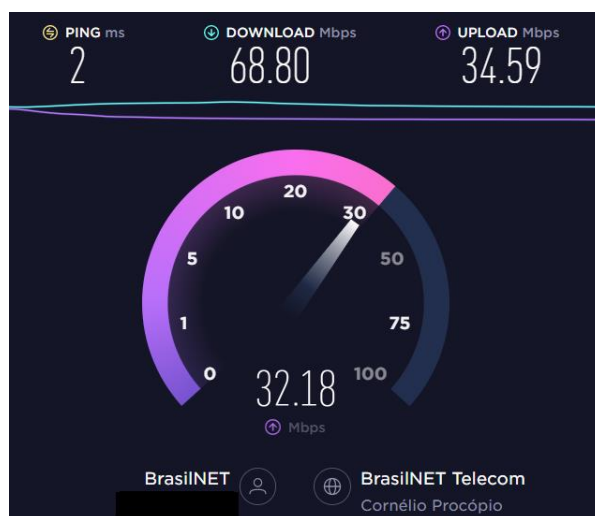


Figura 1: Teste sendo realizado no Speedtest

FONTE: Speedtest.net, 2022

A empresa Ookla, fundada em 2006 em Seattle, vem trabalhando no desenvolvimento de seu site, com a finalidade de sempre aprimorar seu medidor de testes. Desde seu lançamento, mais de 40 bilhões de testes já foram realizados. Dado o número de testes já feitos, o número de usuários desse site pode ser extremamente alto, apesar de não ser comparável a sites famosos, como o Google ou Youtube. A desenvolvedora utiliza os dados de testes armazenados para a

formação de estatísticas de velocidade para cada país, incluindo velocidades de banda da internet fixa e a de redes móveis (SPEEDTEST, 2022).

Portanto, será desenvolvido um aplicativo o qual será responsável pelo monitoramento de determinada rede de comunicação específica. Muitas residenciais e ambientes corporativos enfrentam problemas de estabilidade de conexão, principalmente lugares em que possuem servidores dedicados. O nosso aplicativo vem com o intuito de analisar os dados de uma rede e conseguir utilizar os dados a favor do usuário.

2. OBJETIVOS

2.1- OBJETIVOS GERAIS

O aplicativo vem com a proposta de saber se a conexão está estável em determinados horários do dia. Além disso, ele coletará dados da conexão em um determinado horário do dia, tais como a latência (ping), jitter, perda de pacote, velocidades médias de download e upload, além do pico momentâneo de ambas. Ele coletará dados de várias redes de conexão as quais estão utilizando o software, e com uma considerável quantidade de dados, poderá ser feita uma tabela, realizando desta forma uma análise rigorosa acerca do plano contratado.

2.2- OBJETIVOS ESPECÍFICOS

O programa terá várias funcionalidades, tais como o teste de velocidade, a verificação de velocidade média do histórico de testes, análise da velocidade média para verificar se as velocidades atingidas satisfazem os requisitos mínimos exigidos pela ANATEL, verificar se a latência (ping) e o jitter são recomendados para jogos e transmissões ao vivo, criação de uma nova tabela no banco de dados, opção para mostrar uma tabela desejada do banco de dados, entre outras funções (OFICINA DA NET, 2021).

Além disso, como se trata de um software de monitoramento, ele poderá ficar na maior parte do tempo em segundo plano, realizando os testes e armazenando os dados obtidos no banco de dados, como será o caso da funcionalidade de monitoramento da rede.

Por fim, também temos como objetivo de criar algumas funcionalidades extras, como a obtenção dos IP externo e interno, além de uma para a comparação entre os melhores e piores testes de download e upload, para que assim o usuário esteja ciente de qual horário seria mais satisfatório utilizar um tipo de transferência de dados como, por exemplo, verificar qual seria o melhor horário da velocidade de

download, com o intuito de o usuário poder baixar filmes e séries sem a necessidade de ter que esperar longos períodos para baixar os dados do servidor.

3. JUSTIFICATIVA

3.1- O USO DA INTERNET NA PANDEMIA

Com a chegada da pandemia do Coronavírus, a internet se tornou um recurso essencial para qualquer indivíduo ter em casa. No entanto, a qualidade da internet, principalmente no Brasil, nem sempre foi boa. E acrescentando ao fato de que muitas famílias passaram a conviver de modo muito mais freqüente em suas residências, é crucial para toda residência ter um serviço que pelo menos tenha uma estabilidade razoável.

Muitos são os serviços utilizados pelas famílias tradicionais, tais como aula online, consumo de filmes e séries em streaming, livestream, pedidos de delivery e entre outros. A maioria desses serviços requer uma banda considerável, dependendo da resolução optada pelo usuário (G1 GLOBO, 2019).

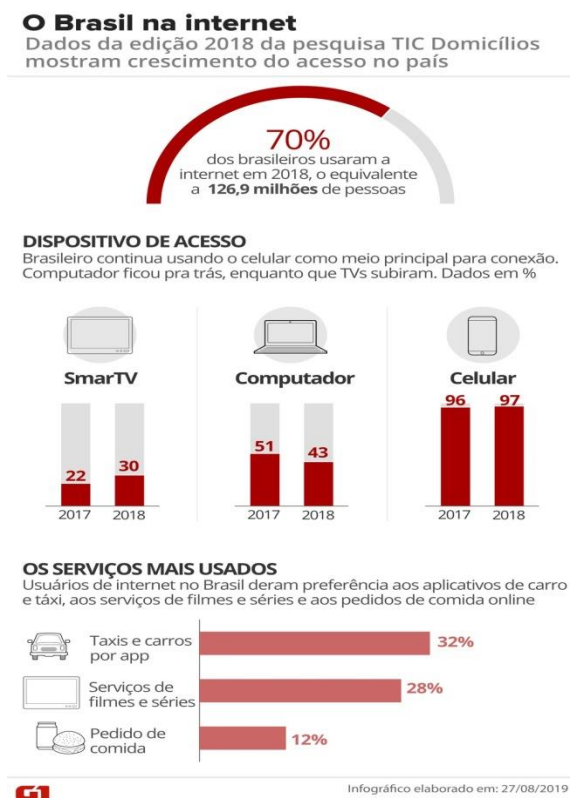


Figura 2: Tendência do consumo brasileiro em alguns dispositivos eletrônicos.

FONTE: G1 – Notícias sobre a economia, 2019

3.2- PRINCIPAIS OPERADORAS DE INTERNET

As principais operadoras de internet banda larga utilizando a tecnologia de fibra óptica do Brasil, Algar Fibras, Claro net virtua, Oi Fibras, TIM Live e Vivo Fibras, possuem cobertura pela maior parte do país. Entretanto, nos últimos anos os pequenos provedores de internet que alugam links dedicados das principais fornecedoras de links do Brasil, como por exemplo, que estão se destacando no mercado atualmente, com mais de 56% dos planos contratados do Brasil, seguido da Vivo Fibras com 25.87% e Oi Fibras com 9.92% (MEIRALSP, 2020)

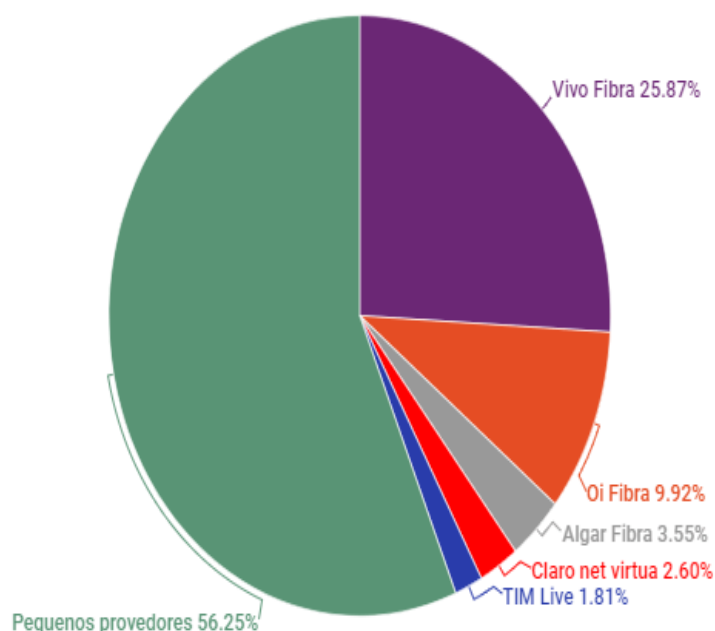


Figura 3: Participação do mercado das operadoras de fibra óptica.

FONTE: MeiralSP – Comparativo entre provedoras, 2020

Levando em consideração os fatos apresentados, se tornou mais do que necessário saber se o plano de internet contratado está condizente com a realidade do usuário. Com um medidor de velocidade que faça os testes periodicamente ao longo do dia, será possível perceber se a conexão está conforme o contratado.

Isso certamente irá melhorar a qualidade da internet em geral, pois com o software, testes serão realizados periodicamente, e o software ainda poderá

comparará redes de internet locais. A maior motivação para criação desse software foi por conta de muitos medidores de velocidades disponíveis na internet não usarem os dados de uma forma que seja benéfica para o usuário, além do teste tradicional apenas mostrar a qualidade da internet no momento realizado.

3.3- REQUISITOS DA ANATEL

Segundo a ANATEL, no Brasil as operadoras de internet banda larga não são obrigadas a oferecer 100% da velocidade contratada. Entretanto, a ANATEL exige que as provedoras precisem fornecer ao menos 80% da taxa de velocidade média de download e upload mensal, e 40% da taxa de transmissão instantânea de download e upload (OFICINA DA NET, 2021).

Estas regras foram definidas com o intuito de manter ao menos um nível mínimo de satisfação pelos usuários, pois há muitos anos atrás as operadoras precisavam entregar apenas 10%, até que de forma gradual esta taxa foi aumentando, até chegar na taxa atual de 80% da velocidade contratada. (ANATEL, 2020)

Entretanto, mesmo diante dessas melhorias, muitos consumidores ainda acabam sendo prejudicados, pois eles relatam que na prática acabam recebendo menos do que foi estipulado dentro das normas exigidas pela reguladora nacional.



Figura 4: Evolução da qualidade da Banda Larga Fixa no Brasil

FONTE: ANATEL, 2020

De acordo com os gráficos e os dados da ANATEL mostrados na imagem anterior, 74.8% dos provedores de internet fixa no Brasil obtiveram êxito em cumprir a meta estipulada pela Resolução nº 575, de 28 de outubro de 2011, a qual aborda a exigência mínima de 80% da transmissão média contratada pelo cliente no plano ofertado (ANATEL, 2011).

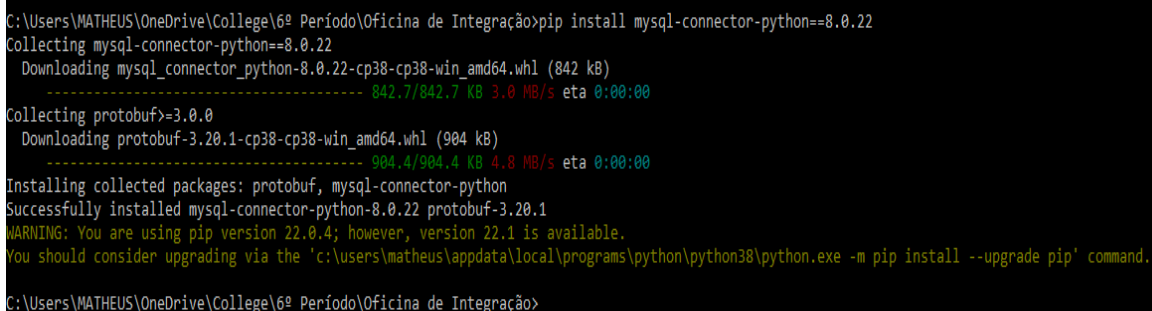
4. METODOLOGIA (OU MATERIAIS E MÉTODOS)

4.1- TECNOLOGIAS USADAS

Através da coleta de dados feitos e do monitoramento das redes de internet, irá ser montado tabelas com estatísticas das respectivas redes de internet, gerando desta forma um banco de dados por meio do MySQL sobre a base de informações coletadas, utilizando o WampServer como um simulador de host para o. Essas tabelas irão servir de auxílio e referência ao usuário para saber se sua conexão em tempo real está de acordo com o plano contratado feito com provedora de internet. (MySQL Workbench, WampServer)

Utilizamos como editor de texto o software Sublime Text, pois ele nós dá uma melhor visualização acerca da indentação código, objetos, funções e entre outros, além de possuir uma interface bastante elegante. (Sublime Text)

O software executável de medidor de velocidade será programado em Python, com a exportação de várias bibliotecas necessárias para seu funcionamento através do comando “pip install (nome da biblioteca)”.



```
C:\Users\MATHEUS\OneDrive\College\6º Período\Oficina de Integração>pip install mysql-connector-python==8.0.22
Collecting mysql-connector-python==8.0.22
  Downloading mysql_connector_python-8.0.22-cp38-cp38-win_amd64.whl (842 kB)
    ----- 842.7/842.7 KB 3.0 MB/s eta 0:00:00
Collecting protobuf==3.20.1
  Downloading protobuf-3.20.1-cp38-cp38-win_amd64.whl (904 kB)
    ----- 904.4/904.4 KB 4.8 MB/s eta 0:00:00
Installing collected packages: protobuf, mysql-connector-python
Successfully installed mysql-connector-python-8.0.22 protobuf-3.20.1
WARNING: You are using pip version 22.0.4; however, version 22.1 is available.
You should consider upgrading via the 'c:\users\matheus\appdata\local\programs\python\python38\python.exe -m pip install --upgrade pip' command.
C:\Users\MATHEUS\OneDrive\College\6º Período\Oficina de Integração>
```

Figura 5: Instalação da biblioteca MySQL.connector

FONTE: Tela do terminal para instalação das bibliotecas, 2022

4.2- BIBLIOTECAS UTILIZADAS

A primeira biblioteca utilizada no código é denominada como speedtest, que se caracteriza pela importação das funcionalidades do site speedtest.net, tais como a realização de testes de download, upload e do ping. A

segunda biblioteca chamada de datetime, a qual será encarregada de receber os dados referentes aos testes realizados, tais como a data e o horário do teste realizado. (The Python Standard Library)

Da biblioteca time, foi importada apenas a função sleep, para que haja uma pausa no programa executável, para uma melhor visualização do usuário. O MySQL.connector foi colocado com a finalidade de estabelecer uma conexão entre o programa executável e o banco de dados, tendo como duas importações específicas o errorcode e o connection, os quais serão utilizados para verificar se os dados do banco fornecidos estão corretos e para estabelecer uma conexão entre o programa e o banco de dados, respectivamente. (The Python Standard Library)

A biblioteca os irá ser utilizada para exportar o comando CLS, que será responsável pela limpeza de informações na tela do usuário. E por fim, a importação da biblioteca socket será essencial para obter os dados do IP interno da máquina, bem como a do get, que será responsável por receber o IP externo da rede de internet. (The Python Standard Library)

Há também a biblioteca Tkinter, a qual será usada para a construção de uma interface gráfica para o programa. Por se tratar de uma interface gráfica simples, sua estrutura estética não é de alto nível. No entanto, como no nosso projeto o foco principal é a análise de dados, optamos por colocá-la.

4.3- DETALHES DO FUNCIONAMENTO INICIAL

O programa é inicializado com uma breve pausa de 1 segundo, e em seguida, é feita uma tentativa de conexão com o servidor do banco de dados, através da função conecta_MySQL(), que retornará o valor 1 caso obtenha sucesso ou 0 no caso de falha (figura 6). Se for sucedida a conexão, o programa é iniciado e é aberta a interface gráfica (figura 11). Caso alguma falha aconteça, é mostrada uma mensagem de erro ao usuário.


```
Inicializando o programa...
Conexão com o Banco de Dados feita com sucesso!
Bem-vindo ao Monitoramento de Qualidade de Internet. Selecione uma opção:

1 - Efetuar um teste de velocidade
2 - Verificar a velocidade média do histórico de testes
3 - Verificar se a velocidade média está condizente com os requisitos da ANATEL
4 - Verificar se a latência (ping) e o Jitter são ideais para uma conexão estável
5 - Mostrar os resultados de uma tabela desejada do banco de dados
6 - Criar uma nova tabela de resultados no banco de dados
7 - Verificar IP interno e externo da rede conectada
8 - Sair do Programa

Digite uma funcionalidade:
```

Figura 6: Menu de interação inicial do programa

FONTE: Tela piloto do Monitoramento de Qualidade de Internet, 2022

O comando do objeto `db_connection` é essencial para o funcionamento de todo o programa, pois será a partir dele que será estabelecida uma conexão com o servidor do banco de dados, que servirá tanto para enviar informações como também receber (figura 7).

```
def conecta_MySQL():
    try:
        db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');
        #aqui é realizado a tentativa de conectar ao host local do computador
        print("Conexão com o Banco de Dados feita com sucesso!");
        return 1;
    except mysql.connector.Error as error:
        if error.errno == errorcode.ER_BAD_DB_ERROR:
            print("O banco de dados não existe!"); #caso isso aconteça, é porque o banco de dados não existe
        else:
            print("O servidor do banco de dados está indisponível no momento!"); #o servidor está offline;
            return 0;
    else:
        db_connection.close();
```

Figura 7: Configuração da função `conecta_MySQL()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Logo após que a conexão é estabelecida, são obtidos a média da velocidade de download, velocidade de upload, ping e jitter através da função `Get_Dados_MySQL()` (figura 9). Isto é feito para que haja valores de referência para usar a maior parte das funcionalidades do programa, pois estes valores serão enviados para grande parte das funções, como será visto no 5º tópico do estudo de forma detalhada.

```

quantidade_testes = 1 #utilizado nas funções de teste
intervalo_minutos = 1 #utilizado nas funções de teste
velocidade_download = 0
velocidade_download = float(velocidade_download);
velocidade_upload = 0;
velocidade_upload = float(velocidade_upload);
ping = 0;
jitter = 0;

print('Iniciando o programa...\n');
sleep(1);

while(1):

    servidorEstado = 0;

    servidor = conecta_MySQL(); #a variável irá receber um valor para indicar se a conexão foi estabelecida ou não

    if(servidor == 1):
        velocidade_download, velocidade_upload, ping, jitter = Get_Dados_MySQL();

```

Figura 8: Parte inicial do código main

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Por convenção, em toda função em que a conexão com o servidor MySQL Workbench for estabelecida, no final sempre haverá um comando denominado `db_connection.close()` o qual será responsável por desconectar com o servidor de forma temporária, para que desta forma não ocorra um processamento ou tráfego de dados desnecessários. Um exemplo de sua utilização está função `Get_Dados_MySQL()`, situado na figura a seguir:

```

def Get_Dados_MySQL():
    db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');
    cursor = db_connection.cursor();
    sql = ("SELECT AVG(Download), AVG(Upload), AVG(Ping), AVG(Jitter) FROM Resultados");
    cursor.execute(sql);
    for(Download, Upload, Ping, Jitter) in cursor:
        velocidade_download = float(Download);
        velocidade_upload = float(Upload);
        ping = float(Ping);
        jitter = float(Jitter);

    cursor.close();
    db_connection.close();
    return velocidade_download, velocidade_upload, ping, jitter;

```

Figura 9: Configuração da função `Get_Dados_MySQL()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

4.4- FUNÇÃO JITTER

Um exemplo das várias funções criadas no programa é a de calcular o valor do Jitter. Como o Speedtest não possui a opção automática de calcular o jitter, criamos manualmente uma função para desempenhar esta funcionalidade. Na 4ª

linha dentro da condição “else”, temos a utilização da função abs. Ela foi usada com o intuito de não gerar um possível erro de soma de um ou mais números negativos, pois se a variação entre dois pings for negativa, a soma para a variável jitter_final ficará incorreta.

Essa função é de suma importância para o projeto, pois ela será usada toda vez em que um teste for inicializado, seja pelo teste normal ou pelo monitoramento de rede.

```
def testa_jitter():
    test = speedtest.Speedtest();
    ping_result = [0,0,0,0,0]; #é criado um vetor para armazenar os valores do ping
    jitter_result = [0,0,0,0]; #é criado um vetor para armazenar as variações entre (n - 1)... pings para o cálculo do jitter
    for c in range(0, 5):
        if(c == 0):
            best = test.get_best_server(); #é pego o melhor servidor baseado no quesito distância x ping
            print(f"Ping {c + 1} : {test.results.ping}");
            ping_result[c] = test.results.ping; #o valor do ping é adicionado ao vetor
        else:
            best = test.get_best_server();
            print(f"Ping {c + 1} : {test.results.ping}");
            ping_result[c] = test.results.ping;
            jitter_result[c - 1] = abs(ping_result[c] - ping_result[c - 1]); #o valor pego entre a subtração da variação
            #entre o ping (n - 1) é armazenado no vetor jitter_result, para que no final o resultado seja dividido por 4,
            #que foi o valor optado para trabalhar neste programa

    jitter_final = (sum(jitter_result))/4; #a função sum soma todos os valores obtidos das variações obtidas das latências
    return jitter_final;
```

Figura 10: Exemplo da função de testar o Jitter

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

4.5- INTERFACE GRÁFICA

Para uma melhor visualização para o usuário, foi criada uma interface gráfica para o programa. Sua configuração é feita a partir da biblioteca Tkinter, a qual já foi mencionada anteriormente na seção de bibliotecas utilizadas.

Inicialmente foram criados sete botões, os quais são referentes as funcionalidades já implementadas no programa inicial, o qual funcionava apenas pelo prompt de comando. Todas as funcionalidades oriundas do programa piloto foram transferidas com sucesso para o modelo com interface gráfica. Posteriormente foram acrescentadas funcionalidades adicionais, o que resultou num total de nove botões.

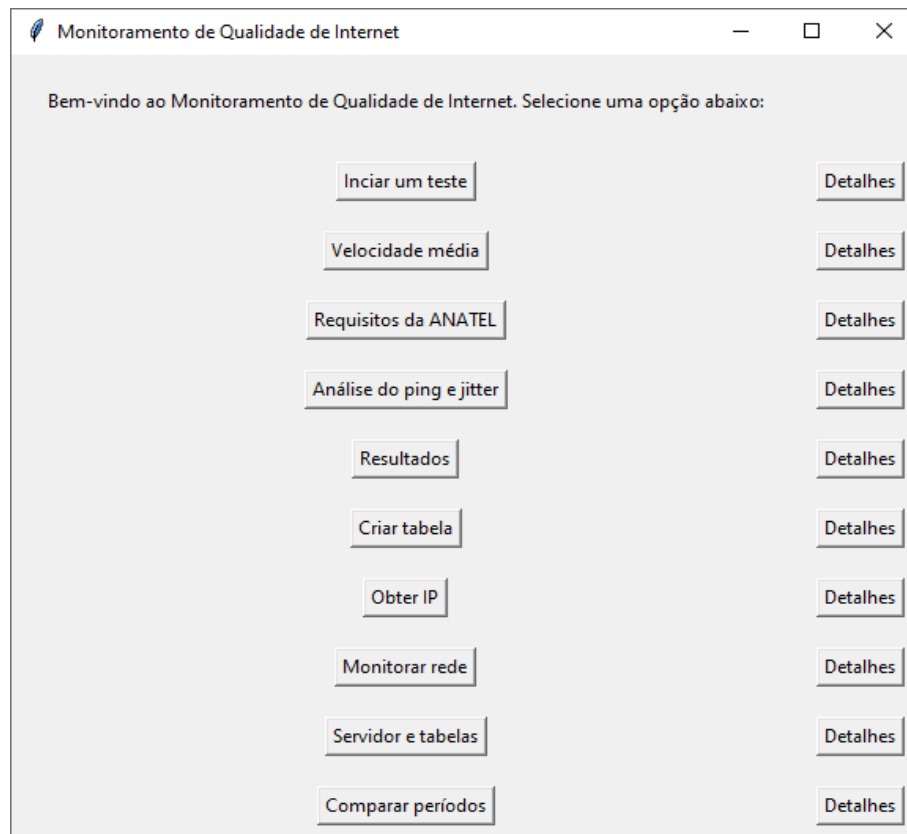


Figura 11: Menu final do programa

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

Foram criados botões denominados “Detalhes” com o intuito mostrar de forma clara de que forma cada botão irá funcionar, além de deixar bem resumido o nome das funcionalidades, para que desta forma o usuário obtenha uma melhor visualização acerca da interface gráfica. Exemplo dos detalhes da funcionalidade “Requisitos da ANATEL”:

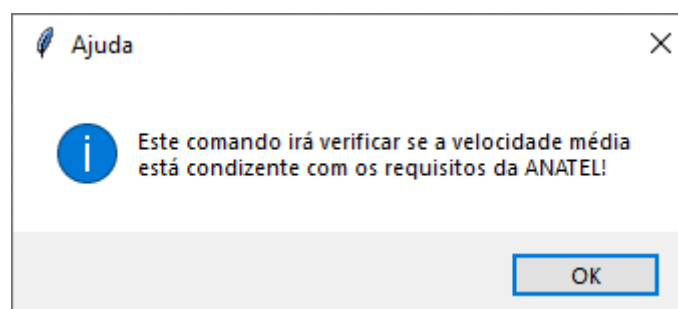


Figura 12: Detalhes da funcionalidade "Requisitos da ANATEL"

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

```
janela = Tk();
janela.title("Monitoramento de Qualidade de Internet");

texto_MenuPrincipal = Label(janela, text="Bem-vindo ao Monitoramento de Qualidade de Internet. Selecione uma opção abaixo:");
texto_MenuPrincipal.grid(column = 0, row = 0, padx = 20, pady = 20);

botao_1 = Button(janela, text = "Inciar um teste", command = Efetuar_Teste);
botao_1.grid(column = 0, row = 1, padx = 10, pady = 10);
```

Figura 13: Exemplo da inicialização de uma janela e configuração dos botões

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

5. FUNCIONALIDADES E RESULTADOS

5.1- RESUMO DAS FUNCIONALIDADES

Várias são as funcionalidades implementadas no programa. Dentre elas, as principais funcionalidades inicialmente planejadas estão inclusas a função para medir o teste de velocidade, a conectividade necessária a ser realizada para conectar o software executável com o banco de dados MySQL, verificação dos requisitos da ANATEL e por fim a obtenção da média de velocidade de uma respectiva tabela.

Além das funcionalidades principais, também acrescentamos algumas adicionais, tais como: a criação de novas tabelas, obtenção dos endereços de IP externo e interno do usuário, monitoramento da rede de internet, função para mostrar o nome do servidor do banco de dados e suas respectivas tabelas padrões e por fim a comparação dos melhores e piores momentos da internet.

5.2- FUNCIONALIDADES PRINCIPAIS

5.2.1- Realizar um teste

O teste de velocidade é a primeira funcionalidade do programa, tendo em vista que ela envolve três funções do programas, denominadas como Efetuar_Teste(), teste_internet() e ImprimeResultados(). Primeiramente é chamada a função Efetuar_Teste(), a qual será encarregada de estabelecer uma conexão com o banco de dados e, logo em seguida, é inicializado o teste através da chamada da função teste_internet().

```
def Efetuar_Teste():
    messagebox.showinfo(title = "AVISO", message = "O teste será inicializado após clicar\nem ok ou a aba ser fechada!");
    for q in range(quantidade_testes):
        data_atual, hora_atual, velocidade_download, velocidade_upload, ping, jitter = teste_internet();
        db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');
        cursor = db_connection.cursor();
        sql = "INSERT INTO Resultados (Data, Horário, Download, Upload, Ping, Jitter) VALUES (%s, %s, %s, %s, %s, %s)";
        values = (data_atual, hora_atual, velocidade_download, velocidade_upload, ping, jitter);
        cursor.execute(sql, values);
        current_date = date.today();
        formatted_date = current_date.strftime('%d/%m/%Y');

        ImprimeResultados(q, data_atual, hora_atual, velocidade_download, velocidade_upload, ping, jitter);

        db_connection.commit();
        cursor.close();
        db_connection.close();

    os.system("cls");
```

Figura 14: Configuração da função Efetuar_Teste()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

A função teste_internet() retorna os valores do dia, hora, velocidade de download, velocidade de upload, ping e jitter do teste feito, tendo em vista que ela foi projetada a partir das funções da biblioteca speedtest, tal como as funções get_servers, get_best_server, download, upload e o ping. A função jitter não é disponibilizada pela biblioteca. Portanto, tivemos que projetar de forma manual uma função para calcular o jitter. O teste é realizado através do código da figura 15, o qual está comentado de forma auto-explicativa, retornando no final as informações referentes ao teste realizado.

```
def teste_internet():
    # Instanciando a função de test do Speedtest
    teste = speedtest.Speedtest();
    print("\nCarregando a lista de servidores...");
    teste.get_servers(); #a função get_servers serve para obter as informações referentes ao servidores disponíveis
    print("\nEscolhendo o melhor servidor...");
    melhor_servidor = teste.get_best_server();
    #o objeto melhor_servidor recebe o melhor servidor escolhido baseado na medida ping x distância
    print(f"\nMelhor servidor: {melhor_servidor['host']} localizado em {melhor_servidor['country']}\n");

    # Testando velocidades
    print('\nRealizando o teste de download...');
    velocidade_download = round(teste.download(threads=None)*(10**-6)) #armazena os dados do teste de download nesta variável
    # é dividido por 10^6 por conta do valor não ser retornado na medida de Mb/s (megabits por segundo)
    print('\nRealizando o teste de upload...');
    velocidade_upload = round(teste.upload(threads=None)*(10**-6)); #armazena os dados do teste de upload nesta variável

    print('\nTestando a latência (ping)...');
    ping_result = round(teste.results.ping); #A função round serve para arredondar as casas decimais
    #para que dessa forma o número final esteja com casas decimais satisfatórias.
    print('\nCalculando o Jitter...');
    jitter_result = round(testa_jitter());

    # Capturando data e hora do teste através das funções da biblioteca datetime.
    data_atual = datetime.now().strftime('%d/%m/%Y');
    hora_atual = datetime.now().strftime('%H:%M');

    return data_atual, hora_atual, velocidade_download, velocidade_upload, ping_result, jitter_result;
```

Figura 15: Configuração da função teste_internet()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Depois de retornado as informações referentes ao teste, elas são salvas no banco de dados através do objeto declarado como sql, o qual executa uma função da linguagem do MySQL denominada “INSERT INTO” a qual é responsável por armazenar esses valores (figura 14). Logo em seguida é chamada a função `Imprime_Resultados()`, nos quais os resultados do teste são mostrados ao usuário, tanto pelo terminal como também em uma nova janela que será aberta pelo programa. Depois de mostrado, os dados são finalmente armazenados pela função `commit`, e a conexão com o banco de dados é encerrada temporariamente.

Na função principal, que é realizar um teste de velocidade, é carregada uma lista de servidores e escolhido o melhor dentre eles. Após isso, é mostrado na tela do usuário o nome do servidor escolhido e sua devida localização geográfica. Logo em seguida os testes de download e upload se inicializam, assim como o teste do ping e do jitter.

Os dados obtidos dos testes, acrescentado as informações do horário e data, são transferidos para um banco de dados MySQL, sendo utilizado como intermédio o software WampServer para que esta conexão seja efetuada com sucesso. Os dados são armazenados em tabelas, em que possuem informações como o número do teste, a data e o horário em que o teste foi feito, os dados das velocidades de Download e Upload medidas, assim como os valores da latência e do jitter. Por fim, no terminal do executável é mostrado o resultado dos testes realizado.

5.2.2- Velocidade Média

A funcionalidade velocidade média obtém a média de todos os testes realizados. Nela se utiliza o comando “SELECT AVG” do MySQL para obter a média dos valores das velocidades de download e upload, como também do ping e do jitter, como se pode ver na figura 16.


```
def Verificar_Velocidade_Media():
    db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');
    cursor = db_connection.cursor();
    sql = ("SELECT AVG(Download), AVG(Upload), AVG(Ping), AVG(Jitter) FROM Resultados");
    cursor.execute(sql);

    janela = Tk();
    janela.geometry("300x80");
    janela.title("Velocidade Média");

    for(Download, Upload, Ping, Jitter) in cursor:
        print('\n\n'); #quebra de linha
        texto = f'''Velocidade de Download: {Download:.02f} Mbps
Velocidade de Upload: {Upload:.02f} Mbps
Ping: {Ping:.02f} ms
Jitter: {Jitter:.02f} ms'''
        texto_print = Label(janela, text = texto);
        #texto_print.grid(column = 0, row = 2, padx = 10, pady = 10);
        texto_print.pack();

        print(f'Média de Download: {Download:.02f} Mbps');
        print(f'Média de Upload: {Upload:.02f} Mbps');
        print(f'Média da Latência (Ping): {Ping:.02f} ms');
        print(f'Média do Jitter: {Jitter:.02f} ms');
        print('\n');

    cursor.close();
    db_connection.close();

    input('Pressione qualquer tecla para continuar');
    os.system("cls");
```

Figura 16: Configuração da função Verificar_Velocidade_Media()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Após obter as médias, elas são mostradas ao usuário por meio de uma nova janela, como pode ser vista na figura 17:

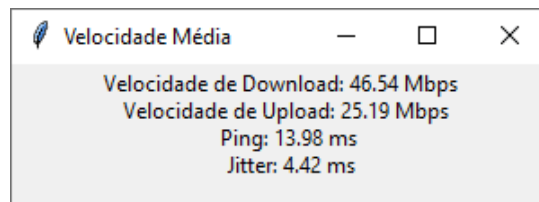


Figura 17: Janela referente aos resultados da velocidade média

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.2.3- Requisitos da ANATEL

De modo a colocar as exigências da ANATEL em prática, criamos uma funcionalidade especificamente para isso. Como é preciso uma entrada de dados do usuário, foram utilizadas duas funções para a tarefa, denominadas como Entrada_Dados_Anatel() e VerificarCondicaoAnatel().

```
def Entrada_Dados_Anatel():
    janela2 = Tk();
    janela2.geometry("300x160");
    janela2.title("Entrada de Dados");

    textExample = Label(janela2, text = "Velocidade de Download contratada:");
    textExample.pack();

    textExample2 = Text(janela2, height = 2, padx = 3, pady = 3);
    textExample2.pack();

    textExample3 = Label(janela2, text = "Velocidade de Upload contratada:");
    textExample3.pack();

    textExample4 = Text(janela2, height = 2, padx = 3, pady = 3);
    textExample4.pack();

    btnRead = Button(janela2, height = 1, width = 10, padx = 5, pady = 5, text = "Verificar",
        command = lambda: VerificarCondicaoAnatel(textExample2.get("1.0","end"), textExample4.get("1.0","end")));
    btnRead.pack();
```

Figura 18: Configuração da função `Entrada_Dados_Anatel()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Na função `Entrada_Dados_Anatel()` é aberto uma nova janela para que o usuário digite os dados referentes as velocidades de download e upload contratadas.

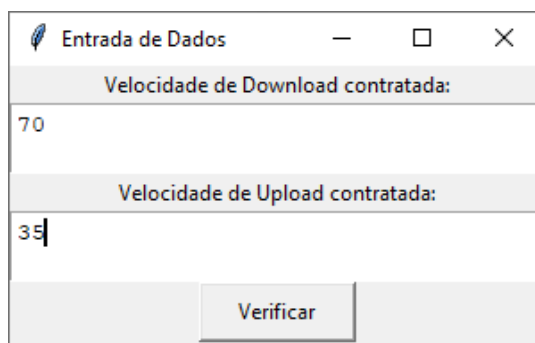


Figura 19: Entrada de dados para as velocidades contratadas

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

Ao clicar no botão “Verificar”, a função `VerificarCondicaoAnatel` é chamada, a qual recebe os dois valores digitados pelo usuário (figura 20). No entanto, antes é analisado se os dados obtidos são realmente números válidos. Logo, é utilizado um tratamento de exceção, tentando forçar os valores obtidos a se tornarem do tipo `float`. Caso isto não seja possível, é mostrada uma notificação de erro ao usuário.

```
def VerificarCondicaoAnatel(Vel_Down_Contratada, Vel_Up_Contratada):
    try:
        Vel_Down_Contratada = abs(float(Vel_Down_Contratada));
        Vel_Up_Contratada = abs(float(Vel_Up_Contratada));
    except Exception as erro:
        messagebox.showinfo(title = "Erro", message = "Falha ao obter os números. Por favor, tente novamente!");
        print(f"Falha ao obter números. O erro foi {erro.__class__}!");
        print(f"Por favor, tente novamente em alguns momentos...");
        return 0;

    if(velocidade_download >= Vel_Down_Contratada * 0.8):
        messagebox.showinfo(title = "Requisito de Download", message =
            "A velocidade de download atingiu pelo menos no mínimo dos 80% da velocidade contratada. Logo, este requisito da ANATEL foi cumprido!");
    else:
        messagebox.showinfo(title = "Requisito de Download", message =
            "A velocidade de download não atingiu pelo menos no mínimo dos 80% da velocidade contratada. Portanto, este requisito da ANATEL não foi cumprido!");

    if(velocidade_upload >= Vel_Up_Contratada * 0.8):
        messagebox.showinfo(title = "Requisito de Upload", message =
            "A velocidade de upload atingiu pelo menos no mínimo dos 80% da velocidade contratada. Logo, este requisito da ANATEL foi cumprido!");
    else:
        messagebox.showinfo(title = "Requisito de Upload", message =
            "A velocidade de upload não atingiu pelo menos no mínimo dos 80% da velocidade contratada. Portanto, este requisito da ANATEL não foi cumprido!");
```

Figura 20: Configuração da função VerificarCondicaoAnatel()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Se os valores forem válidos, é analisado se as velocidades atendem o requisito de 80% da velocidade mínima contratada. Os valores de entrada foram comparados com os resultados da velocidade média (figura 17). Com isso obtivemos os seguintes resultados, os quais não foram satisfatórios:

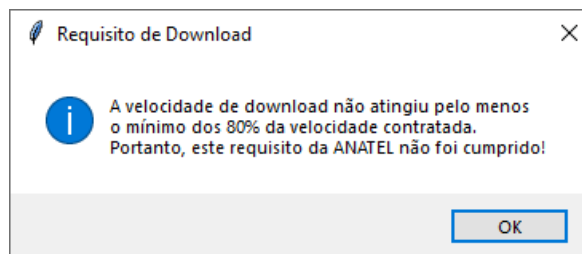


Figura 21: Resultado do requisito de download

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

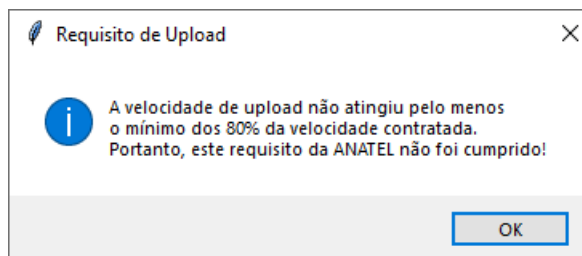


Figura 22: Resultado do requisito de upload

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.2.4- Análise do ping e jitter

Para uma análise da estabilidade da conexão, decidimos criar esta função que, apesar de seu público direcionado ser os gamers, também pode ser utilizada por usuários comuns que fazem utilização de recursos que precisam de uma conexão momentânea estável, como transmissão de reuniões via Google Meet ou serviços de streaming ao vivo.

A função utilizada é a `Verificar_Conexao_Jogos()`, a qual recebe dois argumentos como parâmetros denominados “ping” e “jitter”. Sua estrutura é bem simples e de fácil compreensão. Os valores de base utilizados para a definição se os valores são bons ou ruins são baseados em nossa experiência de jogos online, aliado aos resultados obtidos dos mesmos nos testes realizados.

```
def Verificar_Conexao_Jogos(ping, jitter):  
    if(0 < ping < 60):  
        messagebox.showinfo(title = "Análise do PING", message =  
            "Esta internet possui uma boa latência para jogar em servidores próximos!");  
    elif (60 <= ping < 100):  
        messagebox.showinfo(title = "Análise do PING", message =  
            "Esta internet possui uma latência razoável para jogar em servidores próximos!\nPode apenas apresentar um leve delay!");  
    else:  
        messagebox.showinfo(title = "Análise do PING", message =  
            "Esta internet possui uma alta latência para se conectar em servidores próximos!\nPortanto, pode apresentar um delay considerável!");  
  
    if(0 < jitter < 20):  
        messagebox.showinfo(title = "Análise do JITTER", message =  
            "Esta internet possui uma um jitter ideal para jogar em servidores próximos!\n");  
    elif(50 <= jitter):  
        messagebox.showinfo(title = "Análise do JITTER", message =  
            "Esta internet possui um jitter razoável para jogar em servidores próximos\nPode apenas apresentar algumas leves travadas!");  
    else:  
        messagebox.showinfo(title = "Análise do JITTER", message =  
            "Esta internet pode apresentar muitas inconsistências na conexão, pois seu jitter não é o ideal\nTravadas podem ser bem frequentes!!");
```

Figura 23: Configuração da função `Verificar_Conexao_Jogos()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Os resultados obtidos foram satisfatórios, como se pode perceber nas seguintes figuras:

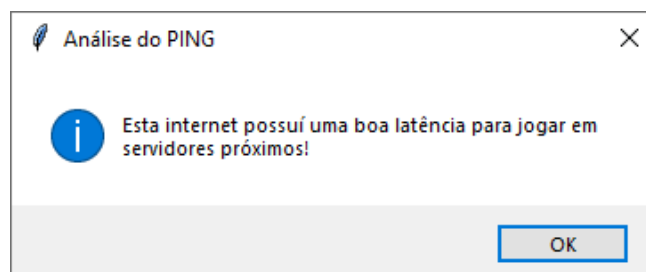


Figura 24: Resultado da análise do PING

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

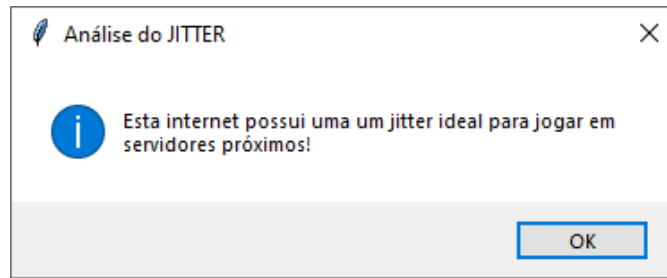


Figura 25: Resultado da análise do JITTER

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.2.5- Resultados

Esta funcionalidade foi criada especificamente para os usuários que usam o monitoramento de rede pelo prompt de comando, onde é possível criar uma tabela específica para fazer vários testes em diferentes horários e armazená-los em uma nova tabela.

```
def Entrada_Dados_Tabela():
    janela = Tk();
    janela.geometry("300x160");
    janela.title("Entrada de Dados");
    texto_print = Label(janela, text = "Digite o nome da tabela: ");
    texto_print.pack();

    textoEntrada = Text(janela, height= 1, width = 20, padx = 3, pady = 3);
    textoEntrada.pack();

    btnRead = Button(janela, height=1, width=5, padx = 5, pady = 5, text="Verificar", command =
    Lambda: Mostrar_Tabela(textoEntrada.get("1.0","end")));
    btnRead.pack()
```

Figura 26: Configuração da função Entrada_Dados_Tabela()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

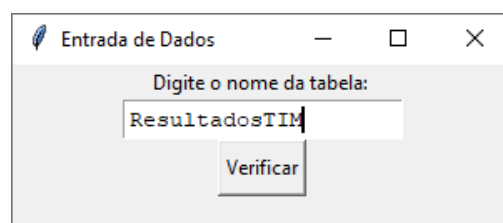


Figura 27: Entrada de dados para o usuário

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

São utilizadas duas funções, nomeadas como `Entrada_Dados_Tabela()` e `Mostrar_Tabela()`, na qual esta ultima recebe como parâmetro a variável `nome`, obtido pela caixa de entrada da janela aberta para a entrada de dados, como se pode ver na figura 28.

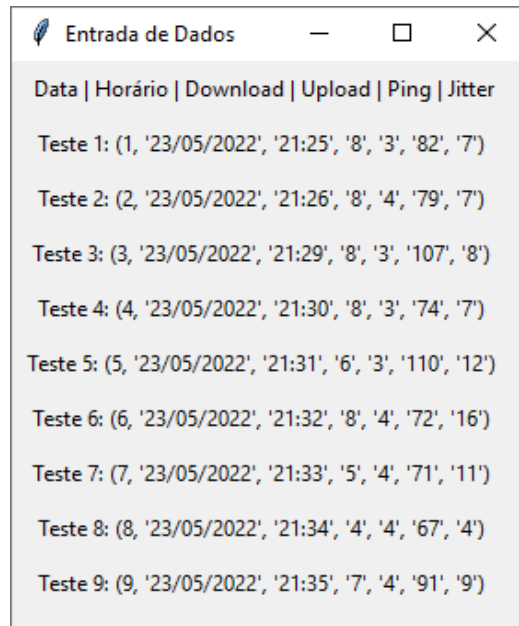
```
def Mostrar_Tabela(nome):  
  
    nome = str(nome);  
    db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');  
    cursor = db_connection.cursor();  
    sql = (f"SELECT * FROM resultado_testes.{nome};");  
  
    try:  
        cursor.execute(sql);  
    except Exception as erro:  
        messagebox.showinfo(title = "Erro", message =  
            "A tabela digitada não existe. Por favor, digite uma tabela\nválida na próxima tentativa!")  
        print(f"A tabela digitada não existe. Por favor, digite uma tabela válida na próxima tentativa!");  
        return 0;  
  
    janela = Tk();  
    janela.geometry("500x320");  
    janela.title("Entrada de Dados");  
  
    cont_aux = 1;  
  
    texto_descricao = Label(janela, text = "Data | Horário | Download | Upload | Ping | Jitter");  
    texto_descricao.grid(column = 0, row = 0, padx = 5, pady = 5)  
  
    for (Linha) in cursor:  
        texto_print = Label(janela, text = f"Teste {cont_aux}: {Linha} ");  
        texto_print.grid(column = 0, row = (1 + cont_aux), padx = 5, pady = 5);  
        print(f'\nTeste {cont_aux} : {Linha}');  
        cont_aux += 1;  
  
    cursor.close();  
    db_connection.close();
```

Figura 28: Configuração da função `Mostrar_Tabela()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Na função `Mostrar_Tabela()`, antes de tudo é verificado se a tabela realmente existe no banco de dados. Portanto, fizemos um tratamento de exceção que força executar o comando “`SELECT * FROM resultados_testes.{nome};`” para tentar mostrar a tabela com o nome digitado pelo usuário mesmo que ela não existe. Caso ela realmente não exista, é mostrada uma notificação de aviso. Se ela existir, uma nova janela é aberta mostrando todos os testes armazenados na respectiva tabela.

Para este estudo, utilizamos a tabela “ResultadosTIM”, criada para testar a rede 4G da internet móvel da empresa TIM. Foi possível realizar nove testes antes de atingir o limite de tráfego de 100 MB estabelecido pela empresa. Os resultados foram razoavelmente satisfatórios, apenas o ping que não obteve bons resultados.



	Data	Horário	Download	Upload	Ping	Jitter
Teste 1:	(1,	'23/05/2022'	'21:25'	'8', '3'	'82', '7')	
Teste 2:	(2,	'23/05/2022'	'21:26'	'8', '4'	'79', '7')	
Teste 3:	(3,	'23/05/2022'	'21:29'	'8', '3'	'107', '8')	
Teste 4:	(4,	'23/05/2022'	'21:30'	'8', '3'	'74', '7')	
Teste 5:	(5,	'23/05/2022'	'21:31'	'6', '3'	'110', '12')	
Teste 6:	(6,	'23/05/2022'	'21:32'	'8', '4'	'72', '16')	
Teste 7:	(7,	'23/05/2022'	'21:33'	'5', '4'	'71', '11')	
Teste 8:	(8,	'23/05/2022'	'21:34'	'4', '4'	'67', '4')	
Teste 9:	(9,	'23/05/2022'	'21:35'	'7', '4'	'91', '9')	

Figura 29: Resultados dos testes da rede móvel 4G da TIM

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.3- FUNCIONALIDADES EXTRAS

5.3.1- Criar tabela

Esta opção foi criada com a finalidade que o usuário possa salvar seus dados em uma nova tabela no banco de dados, pois caso ele tenha interesse de salvar testes de uma data ou horário específicos, ele terá essa possibilidade. Entretanto, devido a algumas inconsistências na transferência de dados da linguagem Python para o MySQL utilizando a interface gráfica, infelizmente essas novas tabelas poderão apenas serem utilizadas na versão do prompt de comando, por meio da função “Monitorar rede”.

```
def Entrada_Dados_Cria_Tabela():
    janela = Tk();
    janela.geometry("220x105");
    janela.title("");

    texto_print = Label(janela, text = "Digite o nome da tabela a ser criada: ");
    texto_print.grid(column = 0, row = 0, padx = 10, pady = 3);

    textoEntrada = Text(janela, height= 1, width = 20, padx = 3, pady = 3);
    textoEntrada.grid(column = 0, row = 1);

    btnRead = Button(janela, height = 1, width = 8, padx = 5, pady = 5, text="Criar tabela",
                     command = lambda: Criar_Tabela(textoEntrada.get("1.0","end")));
    btnRead.grid(column = 0, row = 2);
```

Figura 30: Configuração da função Entrada_Dados_Cria_Tabela()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Foram utilizadas duas funções, denominadas como Entrada_Dados_Cria_Tabela() e Criar_Tabela(), que podem ser encontradas suas estruturas nas figuras 30 e 31, respectivamente, na qual a segunda receberá como parâmetro o nome da tabela digitada pelo usuário.

```
def Criar_Tabela(nome):
    db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');
    cursor = db_connection.cursor();
    sql = (f"CREATE TABLE `{nome}`( `Teste` SERIAL PRIMARY KEY,`Data` VARCHAR(20) NOT NULL ,`Horário` VARCHAR(30) NOT NULL,`D`

    try:
        cursor.execute(sql);
    except Exception as erro:
        messagebox.showinfo(title = "Erro", message =
            "A tabela digitada já existe. Por favor, digite um nome\ndiferente na próxima tentativa!")
        print(nome);
        print(f"Falha ao obter números. O erro foi {erro.__class__}!");
        print(f"Por favor, tente novamente em alguns momentos...");
        return 0;

    cursor.close();
    db_connection.close();

    messagebox.showinfo(title = "Mensagem de notificação", message = "Tabela criada com sucesso!");

    input('Pressione qualquer tecla para continuar');
    os.system("cls");
```

Figura 31: Configuração da função Criar_Tabela()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

Na função Criar_Tabela(), é verificado se o nome não é equivalente a nenhuma outra tabela do banco de dados. Caso seja igual, é mostrada uma mensagem de notificação ao usuário de que a tabela já existe. Caso ao contrário, a tabela é criada e armazenada na base de dados.


```

• CREATE TABLE `{nome}` (
  `Teste` SERIAL PRIMARY KEY,
  `Data` VARCHAR(20) NOT NULL ,
  `Horário` VARCHAR(30) NOT NULL,
  `Download` VARCHAR(20) NOT NULL,
  `Upload` VARCHAR(20) NOT NULL,
  `Ping` VARCHAR(20) NOT NULL,
  `Jitter` VARCHAR(20) NOT NULL
);

```

Figura 32: Comando SQL da função Criar_Tabela()

FONTE: Código da figura 22 inserido no MySQL Workbench, 2022

5.3.2- Obter IP

Em numa ocasião em que o usuário tenha interesse em saber seus endereços de IP interno e externo, criamos uma função simples que é capaz de realizar isto (figura 33).

```

def Verificar_IP():
    janela = Tk();
    janela.geometry("150x75");
    janela.title("IP Externo e Interno");

    IP_Externo = get('https://api.ipify.org').text;
    texto_print = Label(janela, text = "IP Externo: " + IP_Externo);
    texto_print.grid(column = 0, row = 0, padx = 5, pady = 5);

    texto_print2 = Label(janela, text = "IP Interno: " + f"{socket.gethostbyname(socket.gethostname())}");
    texto_print2.grid(column = 0, row = 2, padx = 5, pady = 5);

```

Figura 33: Configuração da função Verificar_IP()

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

A função utiliza o site ipify.org, o qual consegue obter o endereço de IPv4 externo do usuário. O site também fornece o endereço de IPv6. No entanto, como nem todos os provedores do Brasil têm suporte para esse tipo de IP, optamos por não colocarmos no programa (ipify API, 2019)

Para a obtenção do IP interno é utilizado a função `gethostbyname(socket.gethostname())`, derivada da biblioteca `socket`. Logo após

conseguir os dados, é aberta uma nova janela ao usuário indicando os dados referentes ao IP. Por questões de segurança, decidimos censurar o IP externo na figura 34.

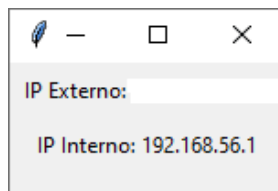


Figura 34: Dados referentes ao IP externo e interno

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.3.3- Monitorar rede

Uma funcionalidade que surgiu literalmente a partir do nome do projeto. Ela consiste na realização de vários testes em um intervalo de minutos especificado pelo usuário. O tempo pode levar alguns segundos a mais do que o desejado pelo usuário, pois não leva em consideração o tempo gasto quando o teste está em andamento, e ainda levando em consideração que a duração de cada teste pode variar.

São usadas três funções, nomeadas como `Entrada_Dados_Monitoramento()`, `Monitorar_SegundoPlano()` e `Efetuar_Teste_Monitoramento()`. Como as duas primeiras seguem o mesmo padrão como foi visto em exemplos anteriores de abertura de uma nova janela com entradas de dados e tratamento de exceção, respectivamente, optamos por não pôr no relatório.

A função `Efetuar_Teste_Monitoramento()` é uma replica da função inicial `Efetuar_Teste()`, com a diferença que no final existe a condição para aplicar o o intervalo inserido pelo usuário.

Pela versão do prompt de comando é possível criar uma nova tabela para inserir as informações de uma respectiva série de testes. Também é possível fazer pelo modelo da interface gráfica. No entanto, os dados serão salvos numa tabela padrão estabelecida pelo programa.


Foi feito uma série de 10 testes, realizados num intervalo de 5 minutos entre cada um, utilizando uma rede de internet de 200 Mbps de download e 100 Mbps de upload, via tecnologia fibra óptica e utilizando um cabo de rede CAT5e

conectado direto ao roteador local. Os resultados obtidos foram bastante satisfatórios, a velocidade de download, ping e o jitter obtiveram excelentes resultados. Apenas a velocidade de upload poderia estar melhor:

	Teste	Data	Horário	Download	Upload	Ping	Jitter
	1	29/06/2022	09:19	205	19	10	1
	2	29/06/2022	09:25	203	100	10	1
	3	29/06/2022	09:30	205	68	9	2
	4	29/06/2022	09:36	204	108	9	2
	5	29/06/2022	09:41	210	114	8	1
	6	29/06/2022	09:47	223	69	10	2
	7	29/06/2022	09:52	223	55	9	2
	8	29/06/2022	09:58	211	30	8	1
	9	29/06/2022	10:03	198	21	9	0
	10	29/06/2022	10:08	223	79	9	1

Figura 35: Resultados pelo editor MySQL Workbench

FONTE: Tabela dos resultados dos testes no MySQL Workbench, 2022



Resultados dos testes

—





Data | Horário | Download | Upload | Ping | Jitter

Teste 1: (1, '29/06/2022', '09:19', '205', '19', '10', '1')

Teste 2: (2, '29/06/2022', '09:25', '203', '100', '10', '1')

Teste 3: (3, '29/06/2022', '09:30', '205', '68', '9', '2')

Teste 4: (4, '29/06/2022', '09:36', '204', '108', '9', '2')

Teste 5: (5, '29/06/2022', '09:41', '210', '114', '8', '1')

Teste 6: (6, '29/06/2022', '09:47', '223', '69', '10', '2')

Teste 7: (7, '29/06/2022', '09:52', '223', '55', '9', '2')

Teste 8: (8, '29/06/2022', '09:58', '211', '30', '8', '1')

Teste 9: (9, '29/06/2022', '10:03', '198', '21', '9', '0')

Teste 10: (10, '29/06/2022', '10:08', '223', '79', '9', '1')

Figura 36: Resultados dos testes no programa

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

5.3.4- Comparar períodos

Esta ideia foi sugerida pelo professor Fabricio Martins Lopes, a qual tratava sobre notificar o usuário os melhores e piores horários para a utilização da internet. Então foi decidido que mostraríamos os momentos mais convenientes para utilizar o download e upload e os momentos menos inconvenientes para o seu uso.

Para isso, foi criada uma função nomeada como `Melhores_Piores_Horarios()`, tendo uma estrutura de abertura de janelas bem semelhantes a funcionalidade de “Mostrar Tabela”. Portanto, destacamos apenas a parte diferente da função na figura abaixo:

```
def Melhores_Piores_Horarios():
    db_connection = mysql.connector.connect(host = '127.0.0.1', user = 'root', password = '', database = 'Resultado_testes');

    cursor = db_connection.cursor();
    cursor2 = db_connection.cursor();
    cursor3 = db_connection.cursor();
    cursor4 = db_connection.cursor();

    sql1 = ("SELECT Data, Horário, Download, Upload from resultados order by Download DESC LIMIT 3");
    sql2 = ("SELECT Data, Horário, Download, Upload from resultados order by Upload DESC LIMIT 3");
    sql3 = ("SELECT Data, Horário, Download, Upload from resultados order by Download ASC LIMIT 3");
    sql4 = ("SELECT Data, Horário, Download, Upload from resultados order by Upload ASC LIMIT 3");
```

Figura 37:Configuração da função `Melhores_Piores_Horarios()`

FONTE: Código do Monitoramento de Qualidade de Internet, 2022

O diferencial nesta função é a utilização dos comandos “DESC” e “ASC”, que pegam os maiores valores de forma decrescente e os menores valores de forma crescente, respectivamente. Ou seja, na função o comando “DESC” servirá para obter os maiores valores de download e upload, enquanto que o “ASC” obterá os menos valores de download e upload, mostrando desta forma os melhores e piores horários das duas velocidades (figura 36).

Melhores testes de download:				Piores testes de download:			
Data	Horário	Download	Upload	Data	Horário	Download	Upload
1º	('21/05/2022', '13:31'	'71'	'16')	1º	('22/05/2022', '08:57'	'45'	'31')
2º	('21/05/2022', '14:24'	'71'	'33')	2º	('22/05/2022', '09:00'	'49'	'35')
3º	('21/05/2022', '13:35'	'70'	'34')	3º	('22/05/2022', '08:49'	'50'	'35')
Melhores testes de Upload:				Piores testes de upload:			
Data	Horário	Download	Upload	Data	Horário	Download	Upload
1º	('22/05/2022', '08:44'	'56'	'36')	1º	('21/05/2022', '13:31'	'71'	'16')
2º	('21/05/2022', '13:58'	'65.0'	'36')	2º	('22/05/2022', '08:48'	'62'	'16')
3º	('21/05/2022', '13:44'	'52.0'	'35')	3º	('22/05/2022', '08:50'	'53'	'19')

Figura 38: Resultado dos melhores e piores períodos

FONTE: Interface do Monitoramento de Qualidade de Internet, 2022

6. CRONOGRAMA

Logo abaixo, encontra-se o cronograma:

Períodos	1ª	2ª	3ª	4ª	5ª	6ª	7ª	8ª	9ª	10ª	11ª
Início	11/03	18/03	25/03	01/04	11/04	22/04	29/04	13/05	27/05	10/06	20/06
Fim	18/03	25/03	01/04	11/04	22/04	29/04	13/05	27/05	10/06	20/06	01/07
Definição Grupo e Projeto											
Análise de testes para escolher as tecnologias mais adequadas ao programa											
Levantamento das funcionalidades											
Estabelecimento de conexão entre o executável e o banco de dados											
Desenvolvimento das funções											
Relatório Parcial											
Desenvolvimento da interface gráfica e outras funcionalidades											
Testes Finais											
Relatório Final											
Revisão e Entrega											

Nós utilizamos este cronograma como referência para realizar as atividades. Entretanto, nem sempre foi possível fazer todas as tarefas da forma com que se foi planejado. Logo, houve de fato algumas pequenas alterações ao longo do projeto.

O membro Matheus Bueno Faria ficou responsável por definir o projeto do grupo, bem como criar os objetivos do programa, além de criar a maior parte das funções do programa, dentre as quais se destacam: realizar teste, monitoramento de rede, conexão ao servidor do banco de dados e análise geral das redes testadas. Também ficou responsável por gerenciar o relatório.

O membro Eduardo Durante Godinho ficou responsável por organizar o cronograma e suas respectivas tarefas. Além disso, por realizar reajustes no código quando necessário, geralmente por pequenos erros ortográficos em caixas de mensagens, além de erros de indentação ou sintaxe do código.

As tarefas foram divididas de acordo com o que foi dito anteriormente e com base no cronograma. Contudo os membros do trabalho, Matheus Bueno Faria e Eduardo Durante Godinho, trabalharam de forma conjunta na maior parte do desenvolvimento do programa, de forma remota, por meio de um aplicativo de

comunicação por meio de microfone e compartilhamento de telas (Google Meet, 2022). Sendo assim, na maior parte do tempo, um obteve o auxílio do outro.

7. CONCLUSÃO

Os objetivos gerais e específicos em sua maior parte foram alcançados. Foi possível criar um programa o qual realiza vários testes de velocidade, bem como conseguir armazenar as informações obtidas em um servidor de banco de dados. Além disso, conseguimos utilizar esses dados para conseguir analisar de forma criteriosa as redes testadas.

Para esta solução, nossas maiores dificuldades foram as seguintes: criar uma função para conseguir manipular de forma conveniente os dados entre a linguagem Python e o banco de dados MySQL, manipular os tipos de dados (String e float) entre essas linguagens de modo que não ocorresse erros no momento em que as funções fizessem a análise dos dados.

Foi possível implementar uma interface gráfica que, apesar de ter uma estrutura simples, seu design ficou razoavelmente bom para o programa. Os tratamentos de exceção foram incluídos com sucesso, o que reduziu significativamente os erros do programa por entrada de dados incorreta feita pelo usuário.

Alguns dos objetivos que o grupo possuía em mente, mas que não foi possível desenvolver para o programa por conta de vários contratempos seria a utilização de APIs para tentar gerar gráficos com a finalidade de realizar um levantamento acerca dos resultados dos testes obtidos. Além disso, não foi possível incrementar a criação de uma nova tabela para a funcionalidade de monitoramento da internet no modelo da interface gráfica.

Por fim, o estudo do projeto permitiu-nos entender a manipular dados entre uma linguagem e outra, além de compreender mais a respeito sobre o trabalho em equipe, de que forma organizar os objetivos e dividir as tarefas para cada membro, que é um fator imprescindível no mercado de trabalho.

8. REFERÊNCIAS BIBLIOGRÁFICAS

BRASIL BANDA LARGA; “**Meça a qualidade de sua conexão**”. Disponível em:
<<https://www.brasilbandalarga.com.br/bbl/>>

WEBOOST; “**Top 9 Internet Speed Test Apps**”. Disponível em:
<<https://www.weboost.com/blog/top-8-internet-speed-test-apps>>

SPEEDTEST; “**Global Index, About**”. Disponível em: <<https://www.speedtest.net/>,
<https://www.speedtest.net/global-index/about>,
<https://www.speedtest.net/global-index>>

SANTISTA; “**Most visited websites worldwide**”. Disponível em
<<https://www.statista.com/statistics/1201880/most-visited-websites-worldwide>>

SELECTRA; “**Internet lenta e sem conexão na quarentena, o que fazer?**”.
Disponível em <<https://selectra.net.br/anatel/noticias/consumidor/como-manter-boa-qualidade-internet-quarentena>>

G1 GLOBO; “**Uso da internet no Brasil cresce e 70% da população está conecta**”. Disponível em
<<https://g1.globo.com/economia/tecnologia/noticia/2019/08/28/uso-da-internet-no-brasil-cresce-e-70percent-da-populacao-esta-conectada.ghtml>>

MEIRA ISP; “**Comparativo: o preço da fibra óptica das operadoras brasileiras**”.
Disponível em
<<https://meiraisp.com.br/comparativo-o-preco-da-fibra-optica-das-operadoras-brasileiras#:~:text=Algar%20Fibra%2C%20Claro%20net%20virtua,os%20grandes%20players%20n%C3%A3o%20chegam.>>

ELETRONET; “**Soluções Eletronet**”. Disponível em

<https://www.eletronet.com/?utm_source=GoogleAds&utm_medium=SEARCH&utm_campaign=2022-01_Institucional_NACIONAL&utm_term=eletronet&gclid=CjwKCAjwjtOTBhAvEiwASG4bCGTdaanM86i2TOSVHaYU6NIIsW6n2MOywgukKgXpVal_cyaFKfWJ6xoC0CMQAvD_BwE>

PINGMAN; “**What is jitter**”. Disponível em

<[https://www.pingman.com/kb/article/what-is-jitter-57.html#:~:text=To%20measure%20Jitter%2C%20we%20take,them%2C%20divide%20by%205\).](https://www.pingman.com/kb/article/what-is-jitter-57.html#:~:text=To%20measure%20Jitter%2C%20we%20take,them%2C%20divide%20by%205).>)>

OFICINA DA NET; “**Por que as operadoras não precisam entregar 100% da conexão contratada?**”. Disponível em

<<https://www.oficinadanet.com.br/post/15990-por-que-no-brasil-as-operadoras-nao-precisam-entregar-100-da-conexao-contratada>>

MINHA CONEXÃO; “**O que é jitter e como ele influencia sua conexão?**”.

Disponível em

<<https://www.minhaconexao.com.br/blog/jitter/>>

MELHOR ESCOLHA; “**Velocidade mínima da internet: ela é direito do consumidor**”. Disponível em

<<https://melhorescolha.com/blog/velocidade-minima-da-internet-v2/>>

ALLEN B. DOWNEY; “**Pense em Python: pense como um cientista da computação**”.

Disponível

em

<https://www.google.com.br/books/edition/Pense_em_Python/qN6RDwAAQBAJ?hl=pt-BR&gbpv=1&dq=inauthor:%22Allen+B.+Downey%22&printsec=frontcover>

DATE, C. J. **Introdução a sistemas de bancos de dados**. Rio de Janeiro, RJ: Elsevier, 2004. 865 p. ISBN 8535212736.

PYTHON; **“The Python Standard Library”**. Disponível em <<https://docs.python.org/3/library/>>

IPIFY API; **“A Simple Public IP Address API”**. Disponível em <<https://ipify.org>>

SUBLIME TEXT; **“Text Editing, Done Right”**. Disponível em <<https://www.sublimetext.com/>>

WAMP SERVER; **“WAMP SERVER, a Windows web development environment”**. Disponível em <<https://www.wampserver.com/en/>>

GOOGLE MEET; **“Videochamadas premium. Agora gratuitas para todos”**. Disponível em <<https://meet.google.com>>