UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE CASTANHAL
FACULDADE DE COMPUTAÇÃO
SISTEMAS DE INFORMAÇÃO

CRIAÇÃO DE UM QUIZ EM SISTEMAS DISTRIBUÍDO COM PYTHON

George Lucas Assad[1]
Lucas dos Santos Soares[2]

INTRODUÇÃO

Os sistemas distribuídos são amplamente utilizados em vários campos, como computação em nuvem, Internet das Coisas (IoT), jogos online e redes sociais. Isso permite que diferentes partes do sistema sejam executadas em redes, hardwares, computadores com sistemas operacionais e até mesmo usando linguagens de programação totalmente diferentes uns dos outros.

Os *Quizzes* ainda são amplamente utilizados em vários campos, como educação, entretenimento e marketing. Estes questionários são uma maneira divertida de testar seu conhecimento sobre um tópico específico e podem ser usados para fins educacionais, como por exemplo, para avaliar o conhecimento de alunos.

Este artigo tem como objetivo o desenvolvimento de um questionário em sistemas distribuídos usando a linguagem de programação Python. O tema foi escolhido por ser uma área em constante crescimento e por apresentar desafios interessantes para os desenvolvedores. Python é uma linguagem de alto nível amplamente utilizada para desenvolvimento de aplicações distribuídas devido a sua facilidade de uso e flexibilidade. Ao decorrer do artigo, discutiremos os conceitos

[1] Discente de Sistemas de Informação da UFPA Castanhal
[2] Discente de Sistemas de Informação da UFPA Castanhal

básicos de sistemas distribuídos e Python, bem como as etapas envolvidas no desenvolvimento de um quiz em sistemas distribuídos com Python.

Palavras-chave: Quiz. Sistemas distribuídos. Python.

REFERÊNCIAL TEÓRICO

QUIZ

Quiz é um jogo com várias perguntas, podendo ser jogado individualmente ou em grupo com o objetivo de acertar o máximo de respostas. É uma forma popular de avaliar o conhecimento, seja em um ambiente educacional ou em uma plataforma de entretenimento. Os questionários podem ser projetados para serem executados em várias plataformas, incluindo aplicativos móveis, sites e até mesmo ambientes de linha de comando.

SISTEMAS DISTRIBUÍDOS

Um sistema distribuído é definido como um conjunto de componentes de hardware ou software em computadores conectados em rede que se comunicam e coordenam suas ações através do envio de mensagens entre si. (COULOURIS, G. et al, 2001). Tais sistemas oferecem benefícios como desempenho aprimorado, escalabilidade e confiabilidade. Esses sistemas podem ser usados em uma variedade de aplicações, como rede de computadores, computação em nuvem, processamento de big data e em jogos multiplayer online.

PYTHON

Josué Labaki (2003, p. 05) define que “Python é uma linguagem de programação de altíssimo nível (VHLL – Very High Level Language), criada pelo holandês Guido Van Rossum sob o ideal de ‘Programação de Computadores para todos’ “. Python é uma linguagem fácil de aprender e poderosa. Ela apresenta estruturas de dados eficientes de alto nível e uma abordagem simples, mas eficaz, para programação orientada a objetos. A sintaxe simples e a digitação dinâmica do Python, combinadas com sua natureza interpretada, o tornam uma linguagem ideal para o desenvolvimento rápido de programas e aplicativos em muitos domínios e na maioria das plataformas.

BIBLIOTECAS PYTHON

Bibliotecas em Python são conjuntos de módulos e funções úteis que reduzem o uso de código no programa (M., 2022). Esses módulos são bem documentados e oferecem uma maneira eficiente de realizar tarefas comuns sem a necessidade de escrever código do zero. Desenvolvedores profissionais aproveitam esses módulos para economizar tempo e aumentar a eficiência no processo de programação.

Há uma ampla variedade de bibliotecas Python disponíveis para atender a diversas necessidades. Algumas das bibliotecas mais populares incluem NumPy e pandas para processamento de dados, scikit-learn e TensorFlow para aprendizado de máquina, Matplotlib e Seaborn para visualização de dados, e Django e Flask para desenvolvimento web, entre muitas outras. Para o desenvolvimento do Quiz em Sistemas Distribuído será utilizado as bibliotecas *socket*, *pickle* e *tkinter.*

SOCKET

A biblioteca socket em Python fornece acesso à interface de soquetes BSD (Berkeley Socket Distribution), que é uma maneira de se comunicar com outros programas usando o protocolo TCP/IP (“HOWTO sobre a Programação de Soquetes — documentação Python 3.8.16”, [s.d.]). Essa biblioteca permite a criação de conexões de rede, que envia e recebe dados através dessas conexões.

PICKLE

A biblioteca pickle em Python implementa protocolos binários para serializar e deserializar uma estrutura de objeto Python. “Pickling” é o processo pelo qual uma hierarquia de objetos Python é convertida em um fluxo de bytes, e “unpickling” é a operação inversa, na qual um fluxo de bytes (de um arquivo binário ou objeto semelhante a bytes) é convertido de volta em uma hierarquia de objetos (“pickle — Python object serialization”, [s.d.]). Isso permite o salvamento e carregamento de objetos complexos em Python.

TKINTER

A biblioteca Tkinter é a interface padrão do Python para o kit de ferramentas GUI Tcl/Tk. Ela permite que você crie interfaces gráficas de usuário (GUIs) para seus programas em Python (“tkinter — Interface Python para Tcl/Tk”, [s.d.]). Com Tkinter, é possivel criar janelas, diálogos, botões, menus e outros elementos de interface mais comuns.

DESENVOLVIMENTO

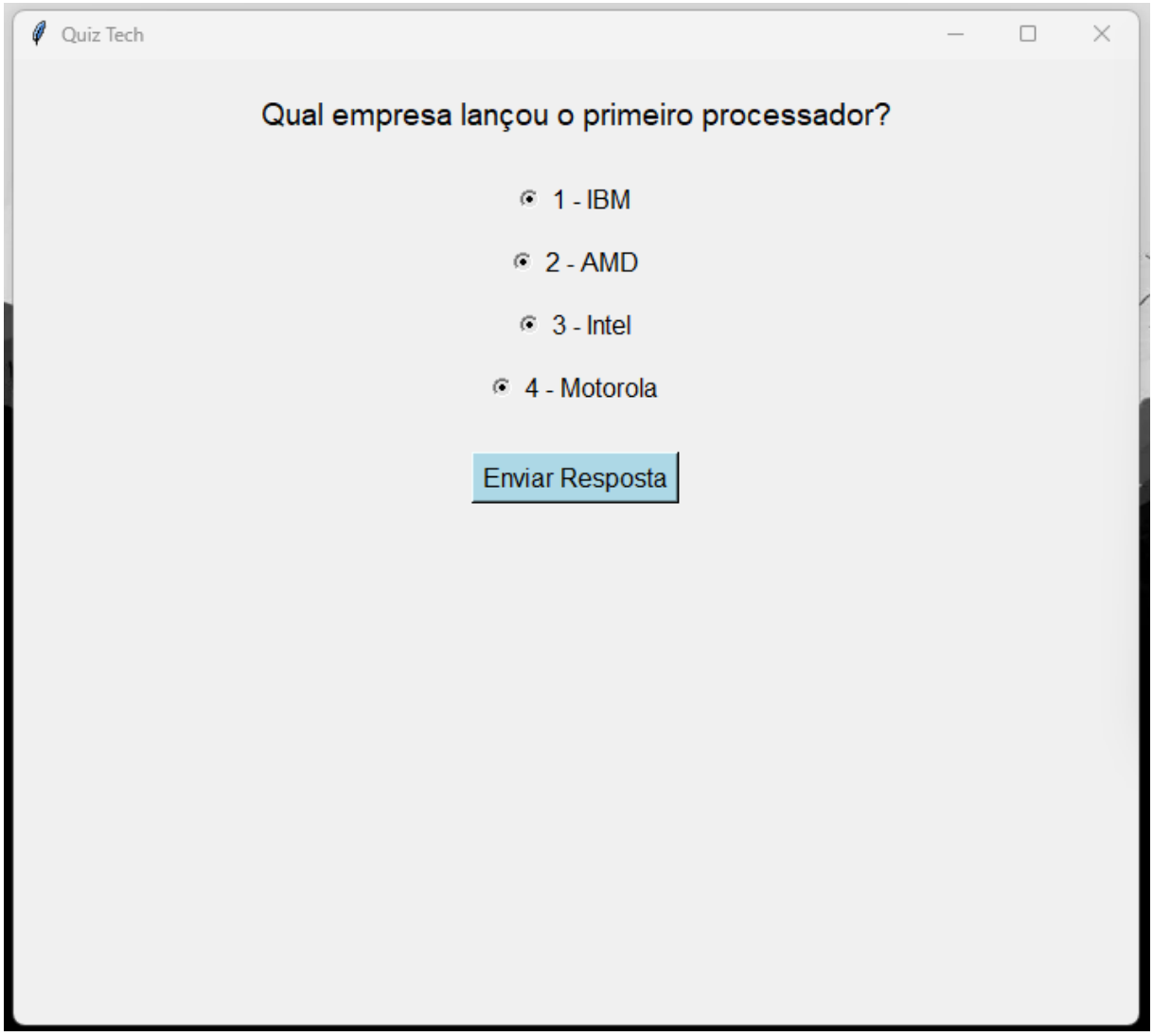
METODOLOGIA

Para o desenvolvimento do quiz, foi utilizado o Python 3. Foram criados dois arquivos principais, o *server.py* e o *client.py.* Usando a biblioteca socket para fazer a conexão entre o servidor e o cliente. A biblioteca pickle foi utilizada para transformar as tuplas de perguntas em bytes e serem transportadas e a biblioteca tkinter foi usada para a criação da interface no lado do cliente.

INTERFACE GRÁFICA

Utilizando o *Tkinter* foi criado uma interface exibindo 4 perguntas e um botão para respondê-las, mostrado na **Imagem 01** abaixo.

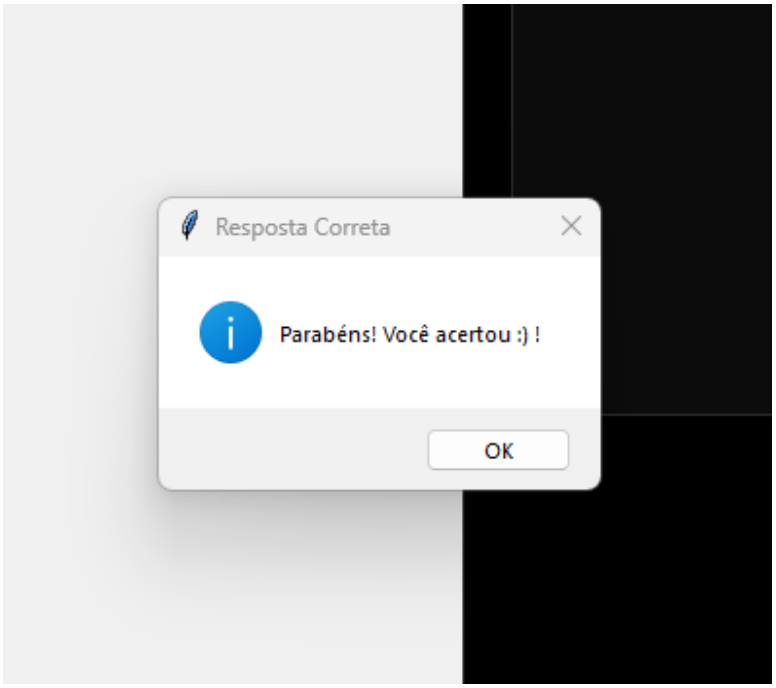
Imagem 01 – Interface do Cliente



FONTE: Autores

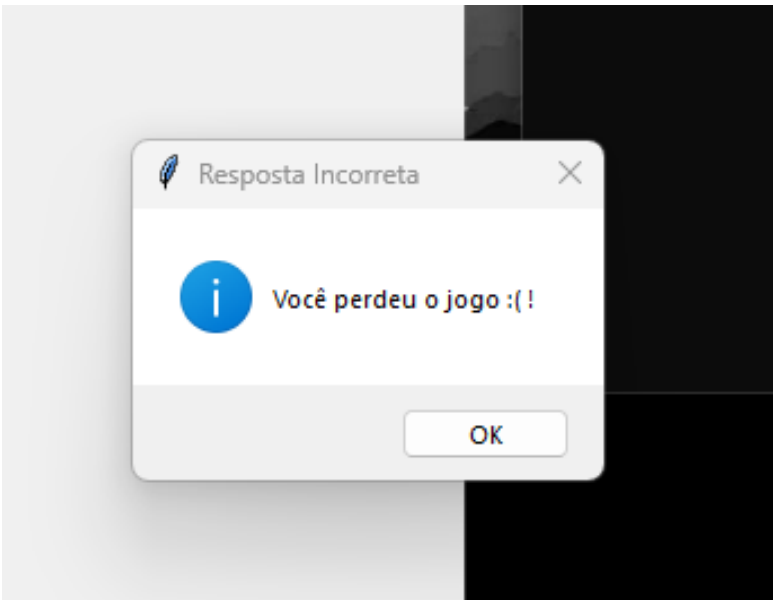
O jogo continua até que todas as perguntas tenham sido respondidas ou até que o jogador erre uma resposta. Se o jogador acertar todas as perguntas, uma mensagem de parabéns é exibida, **Imagem 02**. Caso contrário, uma mensagem de fim de jogo é exibida, **Imagem 03**.

Imagem 02 – Resposta Correta



FONTE: Autores

Imagem 03 – Resposta Errada



FONTE: Autores

DESENVOLVIMENTO DO QUIZ

O código consiste em um sistema de quiz em rede, composto por dois arquivos: server.py e client.py. O sistema de quiz utiliza o protocolo TCP/IP para comunicação entre o servidor e o cliente.

SERVIDOR

No código do servidor é definido duas funções principais, ***handle_client()*** e ***start_server().***

Na função *start_server()*, o servidor aguarda uma conexão de cliente em um determinado endereço IP e porta, mostrado na **Imagem 04**, foi criado as variáveis HOST e PORT, respectivamente com os valores, '127.0.0.1' e 1234.

Imagem 04 – start_server()

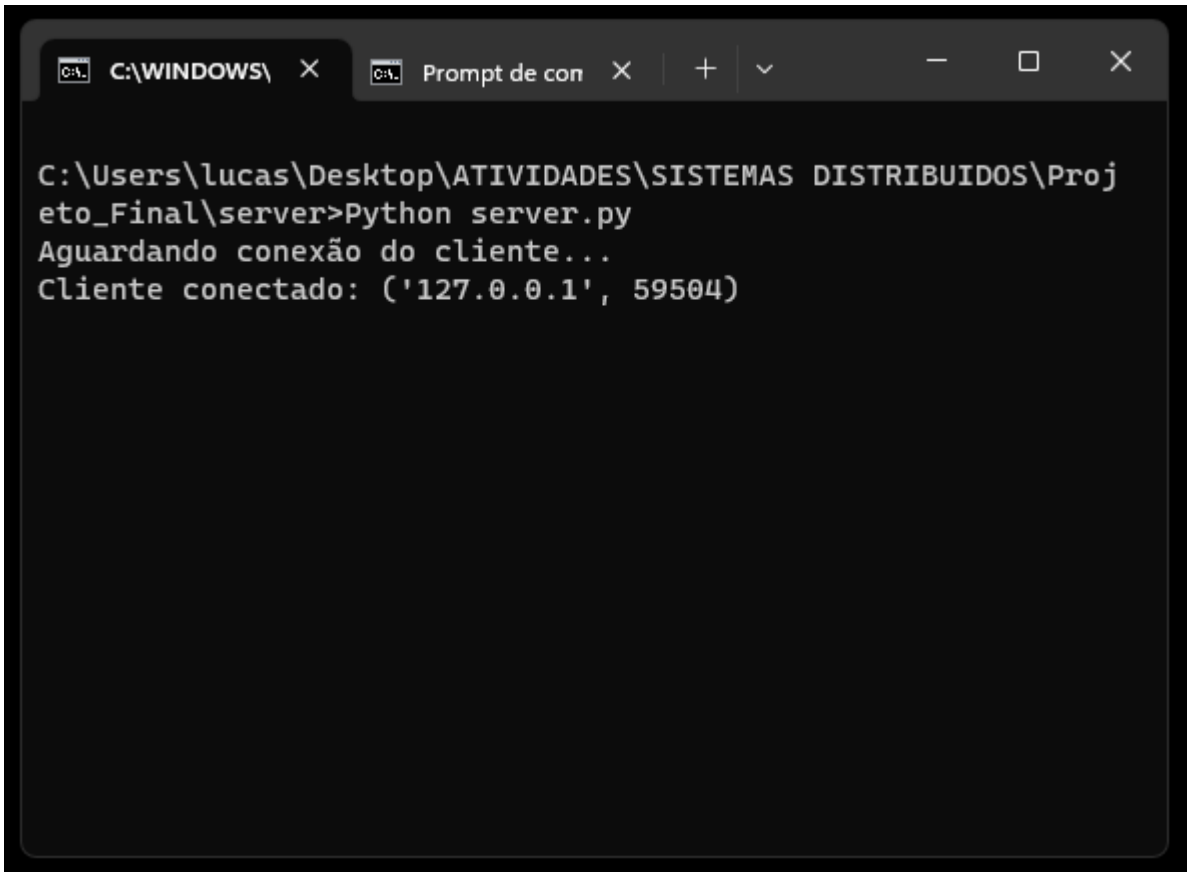
FONTE: Autores

Na linha 2, a variável *server_socket* é instanciada com a criação do servidor na configurações IPV6 e o protocolo TCP/IP, respectivamente *socket.AF_NET* e *socket.SOCK_STREAM.* A função *server_socket.**bind((HOST, PORT))*** define o endereço e porta de conexão do servidor, já definido anteriormente e o

*server_socket.**listen()*** aguarda a conexão com um cliente. Na linha 9 mostra é onde a conexão com o cliente é aceita.

Assim que um cliente se conecta, o servidor estabelece uma conexão e inicia o jogo. A **Imagem 05** mostra a conexão bem-sucedida do cliente no servidor.

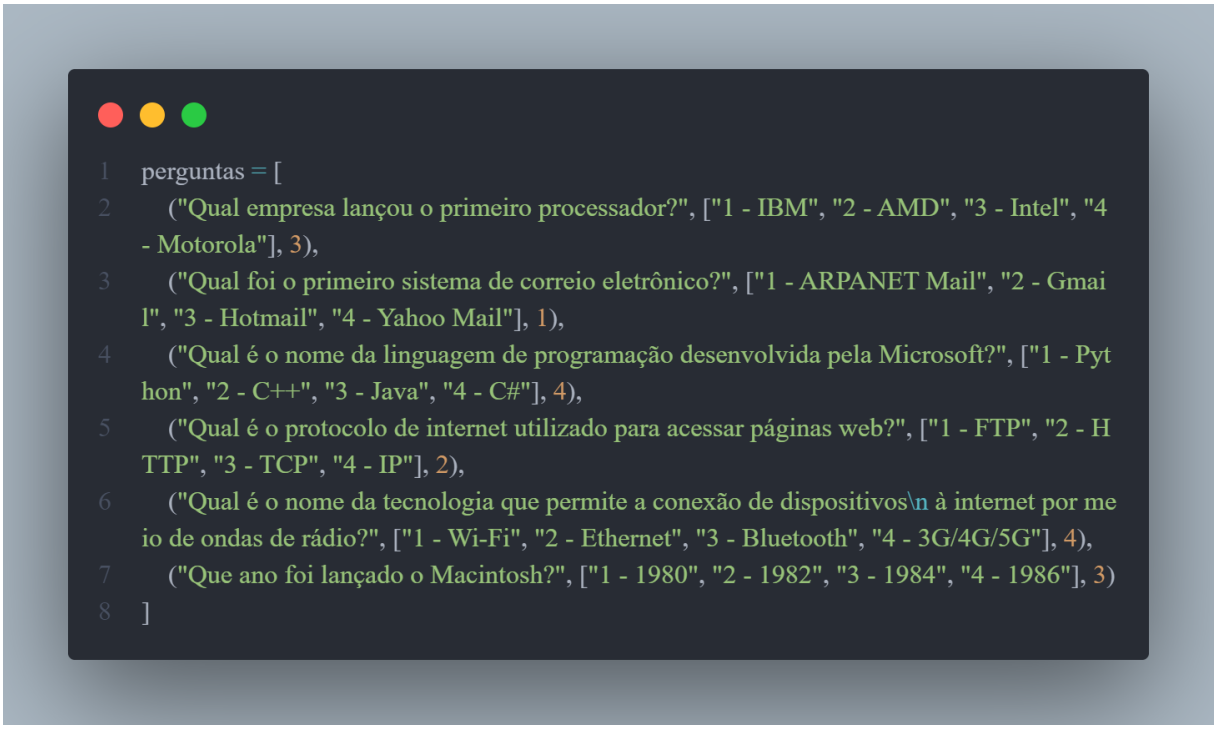
Imagem 05 – Cliente Conectado



FONTE: Autores

Já na função *handle_client()* o servidor possui uma lista de perguntas, opções de resposta e respostas corretas. Cada pergunta é representada como uma tupla contendo a pergunta em si, uma lista de opções e o índice da resposta correta dentro da lista de opções. A seguir na **Imagem 06.**

Imagem 06 – Perguntas e Resposta



```
perguntas = [
    ("Qual empresa lançou o primeiro processador?", ["1 - IBM", "2 - AMD", "3 - Intel", "4 - Motorola"], 3),
    ("Qual foi o primeiro sistema de correio eletrônico?", ["1 - ARPANET Mail", "2 - Gmail", "3 - Hotmail", "4 - Yahoo Mail"], 1),
    ("Qual é o nome da linguagem de programação desenvolvida pela Microsoft?", ["1 - Python", "2 - C++", "3 - Java", "4 - C#"], 4),
    ("Qual é o protocolo de internet utilizado para acessar páginas web?", ["1 - FTP", "2 - HTTP", "3 - TCP", "4 - IP"], 2),
    ("Qual é o nome da tecnologia que permite a conexão de dispositivos\n à internet por meio de ondas de rádio?", ["1 - Wi-Fi", "2 - Ethernet", "3 - Bluetooth", "4 - 3G/4G/5G"], 4),
    ("Que ano foi lançado o Macintosh?", ["1 - 1980", "2 - 1982", "3 - 1984", "4 - 1986"], 3)
]
```

FONTE: Autores

As perguntas são iteradas usando um loop for. Durante cada iteração, o servidor envia a pergunta, as opções e um sinalizador para indicar se é o fim do jogo para o cliente. O servidor usa o módulo *pickle* para serializar os dados e enviá-los como bytes pelo socket. Abaixo em **Imagem 07.**

Imagem 07 – Envio de perguntas

```
1  acertos = 0
2
3  for pergunta, opcoes, resposta_correta in perguntas:
4      client_socket.send(pickle.dumps((pergunta, opcoes, False)))
5
6      resposta_cliente = int(client_socket.recv(1024).decode())
7
8      if resposta_cliente == resposta_correta:
9          acertos += 1
10         client_socket.send('acertou'.encode())
11     else:
12         client_socket.send('errou'.encode())
13         break
14
15 if acertos == 3:
16     client_socket.send(pickle.dumps(("Você acertou todas as perguntas!", [], True)))
17 else:
18     client_socket.send(pickle.dumps(("Fim de Jogo", [], True)))
19
20 client_socket.close()
```

FONTE: Autores

Por fim na linha 20 mostrado na **Imagem 07,** o cliente é fechado assim que o quiz é finalizado.

CLIENTE

O cliente, por sua vez, inicia uma conexão com o servidor. Ele utiliza a biblioteca *Tkinter* para criar uma interface gráfica simples com uma janela, uma pergunta, opções de resposta e um botão para enviar a resposta, mostrado na **Imagem 08.**

Imagem 08 – Conectando e criando uma interface

```
1  HOST = '127.0.0.1'
2  PORT = 1234
3
4  client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5  client_socket.connect((HOST, PORT))
6
7  root = tk.Tk()
8  root.title("Quiz Tech")
9  root.geometry("700x600")
10
11 pergunta_label = tk.Label(root, text="", font=("Arial", 14))
12 pergunta_label.pack(pady=20)
```

FONTE: Autores

Nas linhas 1 a 5, da **Imagem 08** é a conexão do cliente ao servidor e das linhas 7 a 12 a criação da interface com o *Tkinter.*

Quando o cliente envia uma resposta, através da função *enviar_resposta()* visto na **Imagem 09.** A resposta é convertida em string e enviada para o servidor através do socket. Em seguida, o cliente recebe o resultado do servidor, que pode ser "acertou" ou "errou". Com base nesse resultado, uma caixa de diálogo é exibida informando se a resposta estava correta ou incorreta.

A pergunta e as opções de resposta são atualizadas usando a função atualizar_pergunta() mostrado na **Imagem 10.** Se a resposta estiver correta, a função é chamada novamente para atualizar a próxima pergunta. Se a resposta estiver incorreta, o cliente encerra a conexão com o servidor e fecha a janela.

Imagem 09 – Função enviar_resposta()

```
def enviar_resposta():
    resposta = resposta_var.get()
    client_socket.send(str(resposta).encode())

    resultado = client_socket.recv(1024).decode()

    if resultado == 'acertou':
        messagebox.showinfo("Resposta Correta", "Parabéns! Você acertou :) !")
    elif resultado == 'errou':
        messagebox.showinfo("Resposta Incorreta", "Você perdeu o jogo :( !")
        client_socket.close()
        root.destroy()
        return

    atualizar_pergunta()
```

FONTE: Autores

Imagem 10 – Função atualizar_pergunta()

```
def atualizar_pergunta():
    pergunta, opcoes, fim_de_jogo = pickle.loads(client_socket.recv(1024))

    if fim_de_jogo:
        messagebox.showinfo("Parabéns!", "Você venceu o jogo :) !")
        client_socket.close()
        root.destroy()
        return

    pergunta_label.config(text=pergunta)
    for i, opcao in enumerate(opcoes):
        opcoes_radiobuttons[i].config(text=opcao)

```

FONTE: Autores

O jogo continua até que todas as perguntas tenham sido respondidas ou até que o jogador erre uma resposta. Se o jogador acertar todas as perguntas, uma mensagem de parabéns é exibida. Caso contrário, uma mensagem de fim de jogo é exibida.

UTILIZANDO O PROGRAMA

Para utilizar o sistema, é precisa executar o arquivo server.py em um terminal ou ambiente Python. Em seguida, execute o arquivo client.py em outro terminal ou ambiente Python. A interface gráfica será exibida e será possível interagir respondendo às perguntas do quiz.

Os arquivos estão disponíveis em um repositório público no GitHub, no endereço: https://github.com/DEV-LUK4Z/projeto_de_sistemas_distribuidos

CONCLUSÃO

O artigo apresentou uma metodologia para o desenvolvimento de uma implementação básica de um quiz em sistemas distribuído, demonstrando como é possível criar uma aplicação cliente-servidor para esse propósito.

A metodologia utilizada pode ser aplicada a outros projetos semelhantes, o que amplia ainda mais sua relevância, sendo possível personalizar as perguntas, as opções de resposta e o número necessário de acertos para vencer o jogo de acordo com as necessidades do utilizador.

REFERÊNCIAS

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. **Sistemas distribuidos.** Addison Wesley, 2001.

HOWTO sobre a Programação de Soquetes — documentação Python 3.8.16. Disponível em: <https://docs.python.org/pt-br/3.8/howto/sockets.html>. Acesso em: 10 jun. 2023.

LABAKI, Josué; WOISKI, E. R. **Introdução a python–Módulo A.** Grupo Python, UNESP-Ilha Solteira, 2003.

M., L. **Bibliotecas Python: Qual é a melhor para cada situação?** Disponível em: <https://br.bitdegree.org/tutoriais/bibliotecas-python/>. Acesso em: 10 jun. 2023.

O tutorial de Python. Disponível em: <https://docs.python.org/pt-br/3/tutorial/index.html>. Acesso em: 10 jun. 2023

pickle — Python object serialization. Disponível em: <https://docs.python.org/3/library/pickle.html>. Acesso em: 10 jun. 2023.

Quiz - **Conceito, Definição e O que é Quiz.** Disponível em: <https://www.meusdicionarios.com.br/quiz/>. Acesso em: 10 jun. 2023.

tkinter — Interface Python para Tcl/Tk. Disponível em: <https://docs.python.org/pt-br/3/library/tkinter.html>. Acesso em: 10 jun. 2023.