  - CHALLENGE 2023

DESENVOLVIMENTO DE INTERFACE EM PYTHON PARA USO DOS FUNCIONÁRIOS

Ser acessível é abrir portas para um mundo onde todos têm igualdade de oportunidades, independentemente das suas diferenças.

Equipe e Responsabilidade:

Cauã Ferrigoli Loureiro – RM 553093

Claudio Silva Bispo – RM 553472

Sara Ingrid da Silva Pereira – RM 554021

Responsabilidade: Claudio vai criar a interface em Python e Cauã e Sara validar o conteúdo escrito como parte do mapeamento do processo.

Sumário

[Contexto 3](#_Toc148371654)

[Desafio do projeto 3](#_Toc148371655)

[Objetivo do grupo 3](#_Toc148371656)

[Resultados esperados pela matéria 4](#_Toc148371657)

[Análise e diagnóstico 4](#_Toc148371658)

[Script em Python: 7](#_Toc148371659)

[Tela de teste 12](#_Toc148371660)

# Contexto

As turmas de agosto do curso de Análise e Desenvolvimento de Sistemas têm o desafio de criar um portal para a Salesforce, focando em acessibilidade e atração de clientes.

O objetivo é tornar o site mais intuitivo, educativo e alinhado com os serviços da empresa.

# Desafio do projeto

Desenvolver um site que fosse verdadeiramente inclusivo e que consiga atrair o máximo de pessoas possíveis, criando leads para novos negócios. Tínhamos que pensar em cada detalhe, desde o esquema de cores até a funcionalidade de leitura por voz, para que ninguém ficasse de fora e que atenda a todos os requisitos e necessidades.

Tornar um site mais inclusivo para pessoas com ou sem deficiência, com funcionalidades educativa e vinculada aos serviços da empresa.

# Objetivo do grupo

Estamos em processo de desenvolvimento de uma interface que permitirá aos usuários (funcionários) acessar e consultar os bancos de dados sem a necessidade de conhecimento em programação ou criação de consultas. Esta interface abrangerá informações essenciais para o funcionamento do CHATBOT, como descrições de produtos e dados previamente disponíveis em mais de 1700 perguntas frequentes. Além disso, possibilitará a consulta de LEADS para novas oportunidades de negócio, provenientes de logins feitos por visitantes, clientes e preenchimento de formulários.

Para atingir esse objetivo, estamos integrando as linguagens Python e Java para criar uma interface que conectará o front-end e back-end, permitindo consultas diretas aos bancos de dados e conexões através de APIs. Com isso, buscamos modernizar o chatbot e aprimorar a dinâmica e performance do site, tornando a experiência do usuário mais ágil.

Uma das funcionalidades que planejamos oferecer é a possibilidade de indicar ou solicitar modificações no conteúdo HTML. Essas modificações visam destacar informações relevantes para os clientes, como a divulgação de produtos específicos e suas características, na página de listagem de produtos. Estamos empenhados em aprimorar a interação e a experiência do usuário no nosso site, tornando-o mais intuitivo e informativo.

# Resultados esperados pela matéria

O pensamento computacional se baseia em quatro pilares fundamentais para a solução de problemas. Para esta entrega, o grupo deve considerar o desafio proposto e desenvolver uma solução que abranja integralmente os quatro pilares do pensamento computacional.

Entregas:

• Decomposição (20 pontos): Como vocês quebraram o problema em partes menores e mais gerenciáveis? •

• Reconhecimento de Padrão (20 pontos): Como os padrões foram identificados para ajudar na solução? •

• Abstração (20 pontos): Quais detalhes foram considerados essenciais e quais foram abstraídos? •

• Algoritmo (20 pontos): Como vocês organizaram as etapas e processos para encontrar a solução?

Implementação (20 pontos):

Desenvolva um Script Python contendo um `menu` que apresente as funcionalidades mais relevantes para o projeto. Para cada funcionalidade destacada, é necessário implementar, no mínimo, as regras de negócio planejadas para essa funcionalidade. Após executar uma funcionalidade, o programa deve fornecer um resumo da operação realizada e oferecer opções ao usuário para realizar uma nova operação ou encerrar o programa. Entrega: Produza um vídeo com duração máxima de 3 minutos. Neste vídeo, todos os integrantes do grupo devem participar para explicar como os quatro pilares do pensamento computacional foram aplicados no contexto do desafio proposto e demonstrem o código em execução

# Análise e diagnóstico

Decomposição:

Identificamos cada elemento como página inicial, produtos, serviços, suporte, e área de contato. Foi quebrado o problema em partes como por exemplo:

1. O que será necessário manter na página principal? e o que vamos retirar.

2. Como será feito o novo menu? e o que vamos fazer com as informações que acreditamos que estão poluindo o site. Neste momento não vamos descartar as informações, vamos incluir as sessões necessárias em algumas páginas e pegar essas informações que não são necessárias ter no site diretamente e inserir essas informações no chatbot e aproveitar para ensinar o CHATBOT a tirar dúvidas dos clientes em suas pesquisas.

Obs.: Hoje o objetivo do CHATBOT é somente coletar informações do usuário, depois propor um formulário para que ele faça o preenchimento e depois direciona ele para o atendimento humano, o que muitas vezes demorou ou fez a página cair/erro.

Foi considerado os aspectos de acessibilidade de cada componente como: modelo de visualização em tela clara ou escura, aplicação de um modelo que leia o texto completo da página para o cliente ou ele ao clicar na opção específica, cada vez que encostar em uma letra ou frase, será lido para ele. Melhorar aspectos das fontes, deixando algumas maiores, outras com fontes visíveis, mudar o tom da cor para que foque onde precise, mas que não dificulte a leitura de qualquer usuário. Navegação mais intuitiva, onde as imagens terão pequenas legendas para mostrar do que se trata, ao encostar nela, será mostrado o texto do significado para que todos consigam ler e entender. Facilidade de procurar alguma informação com o uso do CHATBOT, seja para procurar por produtos ou saber mais informações sobre alguma característica. Também mostrar um link para direcioná-lo para uma aba correta, sem precisar ficar procurando ou clicando. Por último iremos aplicar a API do Chat-GPT para ampliar o modelo de aprendizado e de passar melhores informações para o usuário, podendo assim criar um lead para um novo negócio.

Reconhecimento de Padrão:

Identificamos padrões de design e usabilidade de cada elemento mencionado.

Reconhecer padrões de acessibilidade, como a utilização correta de etiquetas em formulários, preenchimento do formulário, foco de teclado e como pode obter as informações uma vez que não pode digitar, semântica HTML apropriada, textos que estão fora do padrão, tamanho, cor ou que não simplifique ou ajude na leitura. Ajustar o botão de pesquisa para que o usuário localize algo através de texto ou por voz, também poder realizar pesquisas no chatbot. São alguns dos padrões identificados e com oportunidade de melhorias.

Abstração:

Focar em elementos essenciais para a experiência do usuário, garantindo acessibilidade em diferentes dispositivos, diferentes tipos de pesquisas de informações no site, seja por escrita, fala ou vídeos com legendas (até mesmo gifs).

Algoritmo:

Organizamos as etapas de desenvolvimento de forma lógica, como:

**Primeiro:** ouvimos qual era a solicitação/problema;

**Segundo:** fizemos uma varredura em todo o site para aprender sobre a empresa e os produtos.

**Terceiro** fizemos todo o processo de navegação como se fossemos o cliente para encontrar determinadas informações e aprender sobre elas para aplicar em "nossa empresa".

**Quarto:** foram feitos testes de usabilidade e acessibilidade e de acordo com isso, definimos que criaríamos uma interface única usando a linguagem Python, onde o funcionário da Sales Force vai conseguir por exemplo consultar os bancos de dados de logins feitos por visitantes, clientes e dos preenchimentos de formulários. Também vai conseguir consultar o conteúdo dos produtos que teremos no site e suas características, podendo indicar (abrir chamado) para que informações sejam alteradas de acordo com as oportunidades. Isso vai gerar os Leads para novos negócios, sem a necessidade de outras áreas entregarem a informação ou ter somente um canal de contato que hoje são os formulários.

**Quinto:** estudamos as tecnologias para saber quais aplicar conforme o problema raiz;

**Sexto:** separamos em partes como iriamos construir as soluções e quem irá fazer cada etapa, respeitando o que foi solicitado.

Plano de Ação:

Implementação de script em Python com objetivo de criar uma interface onde contenha um menu e sua funcionalidades serão relevantes para que cada usuário consiga verificar o conteúdo que contém no site e em determinadas páginas poder indicar uma mudança no conteúdo, deixando-o de forma dinâmica a fim de entender qual é o melhor produto naquele determinado período e assim chamar atenção do cliente. Também consultar os bancos de dados disponíveis para localizar possíveis leads e saber o comportamento do usuário/visitante. Não podendo esquecer das conexões via API para deixar o chatbot mais inteligente e com respostas melhoradas a cada pesquisa.

Com esse menu será possível consultar informações no banco de dados de login de visitantes e saber por onde ele passou e quanto tempo ficou analisando uma informação.

No chatbot teremos um banco de dados com informações sobre produtos, serviços e contatos. O usuário vai poder avaliar se as informações estão de acordo com cada momento, se estão simples, e poder comunicar uma alteração ou incluir o que for necessário. Vamos aproveitar para integrar o chat via API no gpt usando a linguagem para deixar o chat mais smart e com conteúdo mais atualizados, de acordo com o segmento da nossa empresa.

Conforme a navegação do usuário, vamos inserir um modelo de login simples com dados de redes sociais como (Linkedin, Gmail, Github, Outlook, Apple, entre outros) para coletar os dados de contato e inserir essas informações no banco de dados para consulta de Leads e o usuário vai poder consultar em tempo real, sem a necessidade de uma área ou pessoa ter que entregar essa informação para ele. O Objetivo é evitar o preenchimento de formulários e coletar mais rápido os dados.

Objetivo principal é concentrar os dados em um único lugar, tornando dinâmico, simples e rápido para tomar qualquer decisão.

# Script em Python:

#-----------( Baixar bibliotecas )----------#

import json

import os

print(os.getcwd())

#-----------( Função para chama o json menu )----------#

# Função para carregar os dados do arquivo JSON - Lá precisa conter a nossa base de busca, onde vou adicionar, excluir, alterar dados

def carregar\_dados():

    with open('./menu\_data.json', 'r') as arquivo:

        dados = json.load(arquivo)

        return dados

#-----------(Função para salvar dados )----------#

# Função para salvar os dados de volta no arquivo JSON

def salvar\_dados(*dados*):

    with open('./menu\_data.json', 'w') as arquivo:

        json.dump(*dados*, arquivo, *indent*=4)

#-----------( Função para voltar ou encerrar o programa )----------#

def opcao():

    print("Para voltar ao menu anterior, digite 0 ou 1 para encerrar o programa")

    opcao = input("Escolha uma opção: ")

    opcao = int(opcao)

    if opcao == 0:

        print("Você definiu a opção de voltar ao menu principal")

    elif opcao == 1:

        print("Encerrando o progama...")

        exit()

    else:

        pass

#-----------( Função para Exibir o menu )----------#

# Função para exibir os menus

def exibir\_menus():

    dados = carregar\_dados()

    menus = dados.get('menus', [])

    print("### As opções disponiveis hoje! ###")

    for menu in menus:

        print(f"ID: {menu['id']} | Nome: {menu['nome']} | Link: {menu['link']}")

    opcao()

#-----------( Função para realizar a consulta no banco, através de conexão ao banco - Estamos imaginando usar o SQL Server por exemplo )----------#

def conectar\_sql():

    server = ''

    database = ''

    user = ''

    password = ''

    tentativas = 3

    for tentativa in range(1, tentativas + 1):

        try:

            conexao = pymssql.connect(*server*=server, *database*=database, *user*=user, *password*=password)

            print(f"Conexão bem-sucedida na tentativa {tentativa}")

            return conexao

        except Exception as e:

            print(f"Tentativa {tentativa} falhou. Erro: {e}")

            if tentativa == tentativas:

                raise

def gerar\_query():

    # Query - Aqui pode modificar se precisar

    query = """ Inserir a consulta aqui quando criarmos o banco """

    # Fazer a conexão e olhar os dados

    conexao = conectar\_sql()

    df = pd.read\_sql\_query(query, conexao)

    conexao.close()

    return df

#-----------( Função para consultar o banco de dados - Log de visitantes )----------#

# Função para exibir os menus

def consultar\_banco\_visitantes():

    print("### Acesso ao banco de dados dos visitantes ###")

    # query para consulta do banco de dados. Aqui será onde a consulta vai ocorrer sem a necessidade de montar outras querys

    #df = gerar\_query()

    #print(df)

    opcao()

#-----------( Função para consultar o banco de dados de logs de clientes)----------#

# Função para exibir os menus

def consultar\_banco\_clientes():

    print("### Acesso ao banco de dados - Logs de clientes ###")

    # query para consulta do banco de dados. Aqui será onde a consulta vai ocorrer sem a necessidade de montar outras querys

    #df = gerar\_query()

    #print(df)

    opcao()

#-----------( Função para consultar o banco de dados - Formulários preenchidos )----------#

# Função para exibir os menus

def consultar\_banco\_formularios():

    print("### Acesso ao banco de dados - Formulários preenchidos ###")

    # query para consulta do banco de dados. Aqui será onde a consulta vai ocorrer sem a necessidade de montar outras querys

    #df = gerar\_query()

    #print(df)

    opcao()

#-----------( Função para consultar o banco de dados )----------#

# Função para exibir os menus

def consultar\_leads():

    print("### Acesso a consulta de Leads ###")

    # query para consulta do banco de dados. Aqui será onde a consulta vai ocorrer sem a necessidade de montar outras querys

    #df = gerar\_query()

    #print(df)

    opcao()

#-----------( Função para adicionar o menu )----------#

# Função para adicionar um novo menu

def adicionar\_conteudo():

    nome = input("Digite o nome do novo menu: ")

    link = input("Digite o link do novo menu: ")

    dados = carregar\_dados()

    menus = dados.get('menus', [])

    novo\_menu = {'id': len(menus) + 1, 'nome': nome, 'link': link}

    menus.append(novo\_menu)

    dados['menus'] = menus

    salvar\_dados(dados)

    print(f"Menu '{nome}' adicionado com sucesso!")

    opcao()

#-----------( Função para remover algum menu )----------#

# Função para remover um menu

def remover\_conteudo():

    id\_menu = int(input("Digite o ID do menu que deseja remover: "))

    dados = carregar\_dados()

    menus = dados.get('menus', [])

    for menu in menus:

        if menu['id'] == id\_menu:

            menus.remove(menu)

            dados['menus'] = menus

            salvar\_dados(dados)

            print(f"Menu ID {id\_menu} removido com sucesso!")

            return

    print(f"Nenhum menu encontrado com ID {id\_menu}.")

    opcao()

#-----------( Função para fechar o programa )----------#

# Função para remover um menu

def sair\_programa():

    print(f"Programa encerrado.")

    exit()

#-----------( Função para o menu de opções )----------#

# Menu de opções

def menu\_principal():

    while True:

        print("\n### Menu de Administração para funcionários ###")

        print("1. Exibir Menus do site")

        print("2. Adicionar conteúdo")

        print("3. Remover conteúdo")

        print("4. Consultar Banco de dados logs - visitantes")

        print("5. Consultar Banco de dados logs - clientes")

        print("6. Consultar Banco de dados formulário")

        print("7. Consultar Leads")

        print("8. Sair")

        opcao = input("Escolha uma opção: ")

        if opcao == "1":

            exibir\_menus()

        elif opcao == "2":

            adicionar\_conteudo()

        elif opcao == "3":

            remover\_conteudo()

        elif opcao == "4":

            consultar\_banco\_visitantes()

        elif opcao == "5":

            consultar\_banco\_clientes()

        elif opcao == "6":

            consultar\_banco\_formularios()

        elif opcao == "7":

            consultar\_leads()

        elif opcao == "8":

            sair\_programa()

            break

        else:

            print("Opção inválida. Tente novamente.")

#-----------( Função para chamar as demais funções )----------#

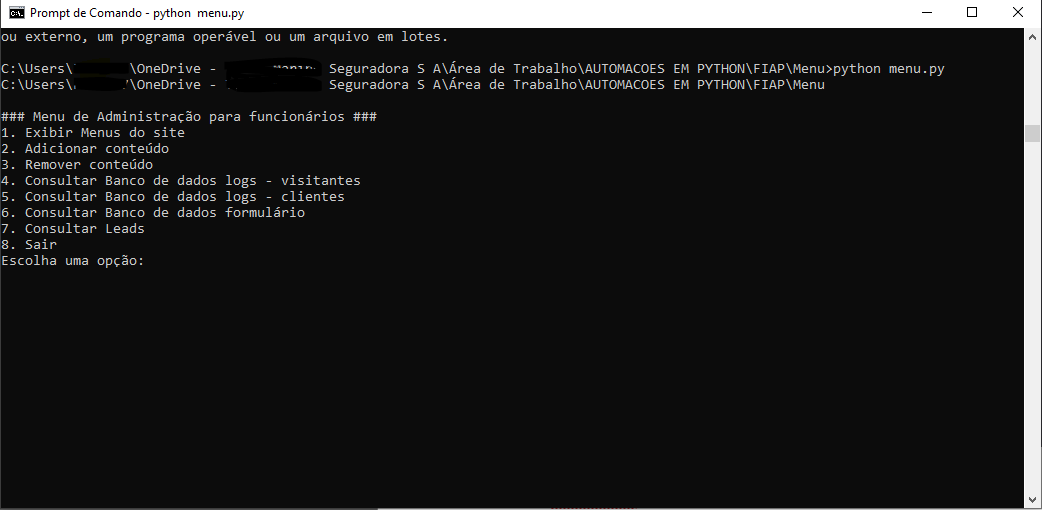
# Iniciar o menu principal - Aqui vou rodar todos os processos

if \_\_name\_\_ == "\_\_main\_\_":

    menu\_principal()

# Tela de teste

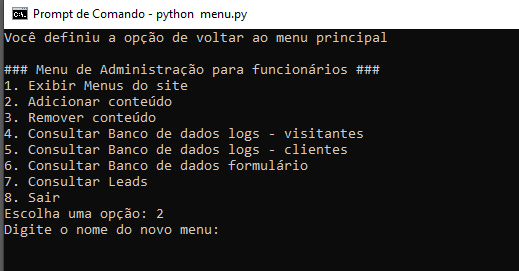
1. Exibir menu inicial

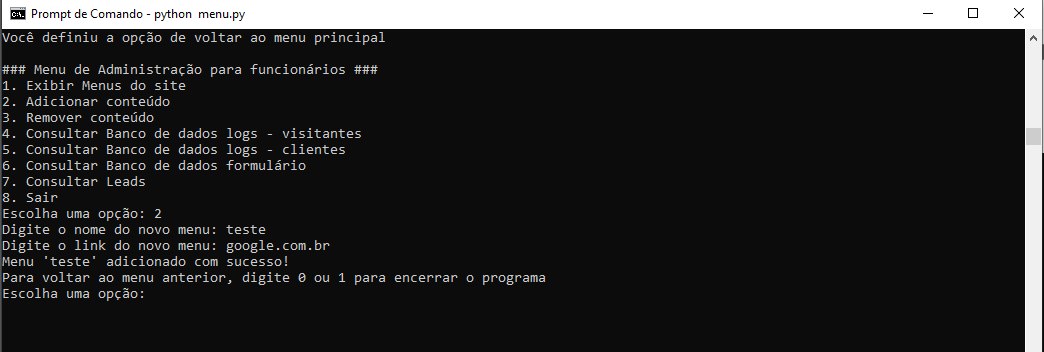


1. Ao clicar no menu inicial, ele vai mostrar as opções que temos disponíveis no site hoje e também duas opções ao usuário, onde uma será para voltar a tela anterior e outra para encerrar a aplicação.

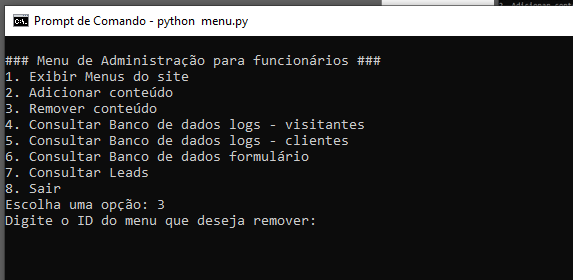


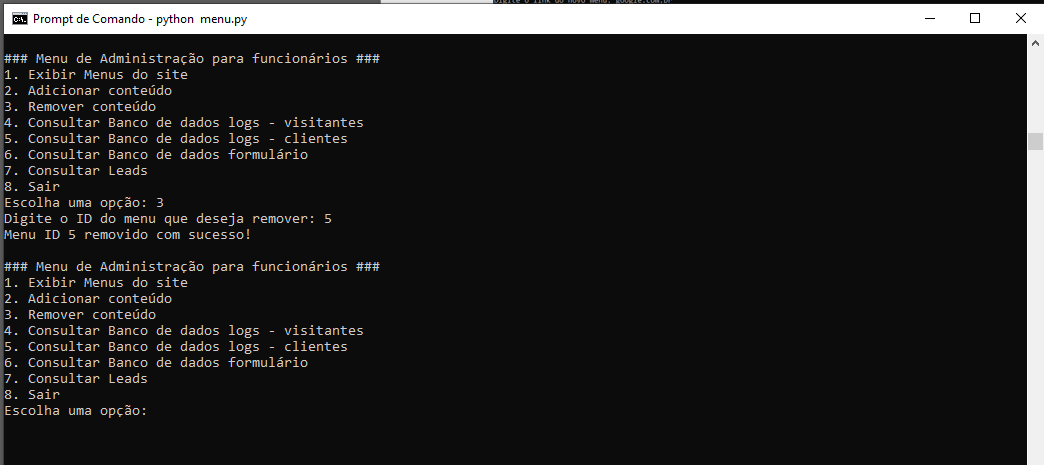
1. Na opção 2, ele será capaz de informar uma opção do menu que quer incluir/adicionar e após isso, voltar para o menu principal ou fechar a aplicação.



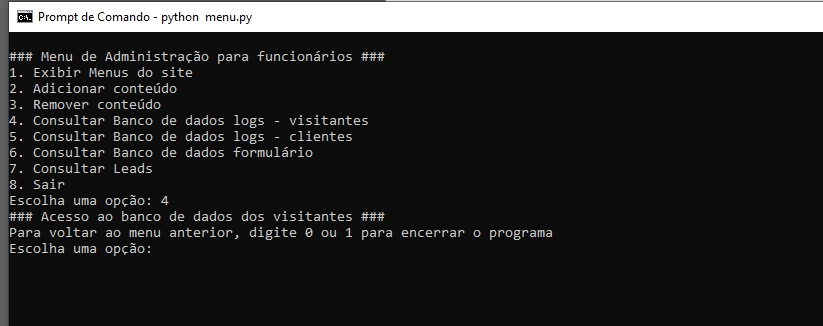


1. Na opção 3, ele será capaz de indicar uma exclusão, assim como as opções de voltar e encerrar a aplicação.





1. Opção 4,5 e 6 será para realizar consulta ao bando de dados de logins feitos pelos visitantes, clientes e preenchimentos de formulários, onde teremos a data, hora, e-mail, nome e o que ele pesquisou. Assim gerando um possível lead para novos negócios. Inserimos a opção de voltar e encerrar a aplicação.



1. A opção 7 é uma possível integração ao banco de dados da empresa, onde contenham informações de clientes para contatar, de acordo com o cadastro do chat, dos consultores do atendimento, até mesmo alguma base de dados existentes que eles tenham criado.
2. Última opção é a 8 que é para encerrar a aplicação.

