**Workshop: Integração do ChatGPT com Python e Consulta Api Via Cep.**

**1. Introdução ao Workshop**

* **Objetivo: Aprender a integrar o ChatGPT com Python usando a API da OpenAI.**
* **Público-Alvo: Desenvolvedores e entusiastas com conhecimentos em Python.**

**2. Fundamentos do Python**

O que iremos precisar entender para fazer esse implementação??

So sair criando código sem entender o contexto???

Etapas:

Conhecimento de Funções em Python.

Funções / Dicionarios/Criação Ambiente Virtual/Importação de Biblioteca/

**2.1 Funções em Python**

* Definição: Blocos de código reutilizáveis que executam uma tarefa específica.
* Exemplo:

def saudacao(nome):

return f"Olá, {nome}!"

print(saudacao("Maria"))

**O que e parâmetro?**

Em Python, um parâmetro é uma variável listada como parte da definição de uma função. Os parâmetros atuam como espaços reservados para valores que serão fornecidos à função quando ela for chamada. Eles permitem que funções sejam mais flexíveis e reutilizáveis, já que você pode passar diferentes valores (argumentos) para a função em diferentes chamadas.

**Parâmetros em Funções**

**Definindo Funções com Parâmetros**

Quando você define uma função, você pode especificar parâmetros dentro dos parênteses da função. Estes parâmetros são usados como variáveis locais dentro da função.

Obs: Def (Palavra Reservada)

**Exemplos de Palavras-Chave em Python**

Além de def, aqui estão algumas outras palavras-chave em Python:

* **if**: Usada para criar instruções condicionais.
* **else**: Usada em conjunto com if para definir um bloco de código a ser executado se a condição do if não for satisfeita.
* **for**: Usada para criar loops que iteram sobre uma sequência (como uma lista ou um intervalo).
* **while**: Usada para criar loops que executam um bloco de código enquanto uma condição for verdadeira.
* **class**: Usada para definir uma classe.
* **return**: Usada para devolver um valor de uma função.
* **import**: Usada para importar módulos e bibliotecas.
* **try** e **except**: Usadas para tratar exceções (erros) no código.

**2.2 Importações**

* Definição: Permite usar bibliotecas e módulos externos.
* Exemplo:

import math

print(math.sqrt(16)) # Usa a função sqrt do módulo math

Alguem sabe o que sqrt faz????

Estudar sempre as bibliotecas da linguagem.....

----------------------------------------------------------------------------

Entendendo Biblioteca.....

 # meubiblioteca.py

def saudacao(nome):

return f"Olá, {nome}!"

def soma(a, b):

return a + b

-----------------------------------------------------------------------------

 **Importe o Módulo**: Em outro arquivo, você pode importar e usar o módulo que você criou.

# usar\_biblioteca.py

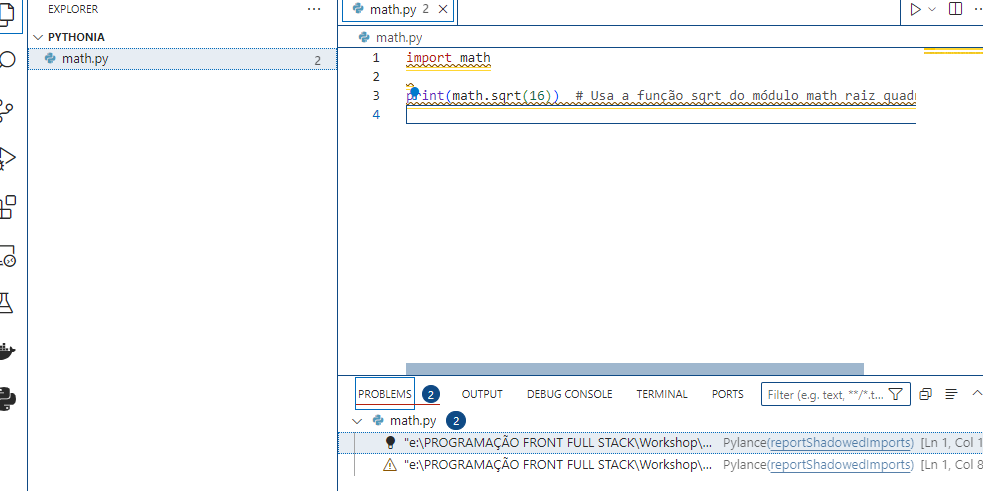
import meubiblioteca

print(meubiblioteca.saudacao("Ana")) # Saída: Olá, Ana!

print(meubiblioteca.soma(5, 7)) # Saída: 12

ERRO.....

 O módulo math é uma biblioteca padrão do Python que fornece funções matemáticas, como sqrt, sin, cos, etc. O arquivo que você criou e está usando é chamado math.py, que tem o mesmo nome do módulo padrão math.



Uso do dicionário em Python....

contatos = {

"João": {

"telefone": "1234-5678",

"email": "joao@example.com"

},

"Maria": {

"telefone": "9876-5432",

"email": "maria@example.com"

}

}

contatos["Ana"] = {

"telefone": "5555-5555",

"email": "ana@example.com"

}

contatos["João"]["telefone"] = "1111-1111"

nome\_contato = "Maria"

if nome\_contato in contatos:

print(f"Detalhes do contato {nome\_contato}:")

print(f"Telefone: {contatos[nome\_contato]['telefone']}")

print(f"Email: {contatos[nome\_contato]['email']}")

else:

print(f"Contato {nome\_contato} não encontrado.")

print("\nTodos os contatos:")

for nome, info in contatos.items():

print(f"\nNome: {nome}")

print(f"Telefone: {info['telefone']}")

print(f"Email: {info['email']}")

**Obs: O Que São F-Strings?**

F-strings são uma forma concisa e eficiente de formatar strings em Python. Elas permitem embutir expressões dentro de strings de forma direta e legível.

Como Funcionam?

* Sintaxe: Uma f-string é uma string precedida pela letra f ou F, seguida por aspas duplas (") ou simples ('). Dentro da string, você pode incluir expressões Python entre chaves {}.
* Substituição de Variáveis: Qualquer expressão dentro das chaves {} é avaliada e o resultado é inserido na string resultante.

**2.3 Bibliotecas Comuns para APIs**

* requests: Para fazer chamadas HTTP.
* json: Para manipular dados JSON.
* Exemplo de Importação:

import requests

import json

**3. Ambiente Virtual**

**3.1 O que é um Ambiente Virtual?**

* Definição: Espaço isolado para instalar pacotes Python, evitando conflitos entre projetos.

**3.2 Criar e Ativar o Ambiente Virtual:**

* **Criar:**

**python -m venv myenv**

* **Ativar:**
  + Windows (Prompt de Comando):

**myenv\Scripts\activate**

* + **Windows (PowerShell):**

.\myenv\Scripts\Activate.ps1

* + **macOS/Linux:**

source myenv/bin/activate

**3.3 Instalar Pacotes Necessários:**

**pip install requests openai**

**4. Conceito de API (Application Programming Interface)**

**4.1 O que é uma API?**

* Definição: Conjunto de regras e protocolos para interação entre diferentes sistemas de software.

**Principais Conceitos do HTTP**

HTTP, que significa Hypertext Transfer Protocol, é o protocolo fundamental usado para a troca de informações na web. Ele define as regras e convenções para a comunicação entre um cliente (como um navegador da web) e um servidor web.

1. Protocolo:
   * O que é: Um protocolo é um conjunto de regras que define como dados devem ser transmitidos e recebidos em uma rede.
   * No Caso do HTTP: Especifica como as mensagens são formatadas e transmitidas, e como o servidor e o cliente devem responder a várias solicitações.
2. Requests e Responses:
   * Request (Solicitação): Quando um cliente (por exemplo, um navegador) quer obter uma página da web ou enviar dados a um servidor, ele faz uma solicitação HTTP. A solicitação inclui um método, um URL, e pode incluir cabeçalhos e um corpo de mensagem.
   * Response (Resposta): O servidor processa a solicitação e responde com um código de status, cabeçalhos e, em muitos casos, um corpo de mensagem com os dados solicitados.

4.2 Métodos HTTP Comuns:

* GET: Recupera dados do servidor.
* POST: Envia dados para o servidor para criar algo novo.
* PUT: Atualiza dados existentes no servidor.
* DELETE: Remove dados do servidor.

4.3 Exemplos:

* GET:

python

Copy code

response = requests.get("https://api.exemplo.com/usuarios")

* POST:

python

Copy code

response = requests.post("https://api.exemplo.com/usuarios", json={"nome": "João"})

* PUT:

python

Copy code

response = requests.put("https://api.exemplo.com/usuarios/1", json={"nome": "João Silva"})

* DELETE:

python

Copy code

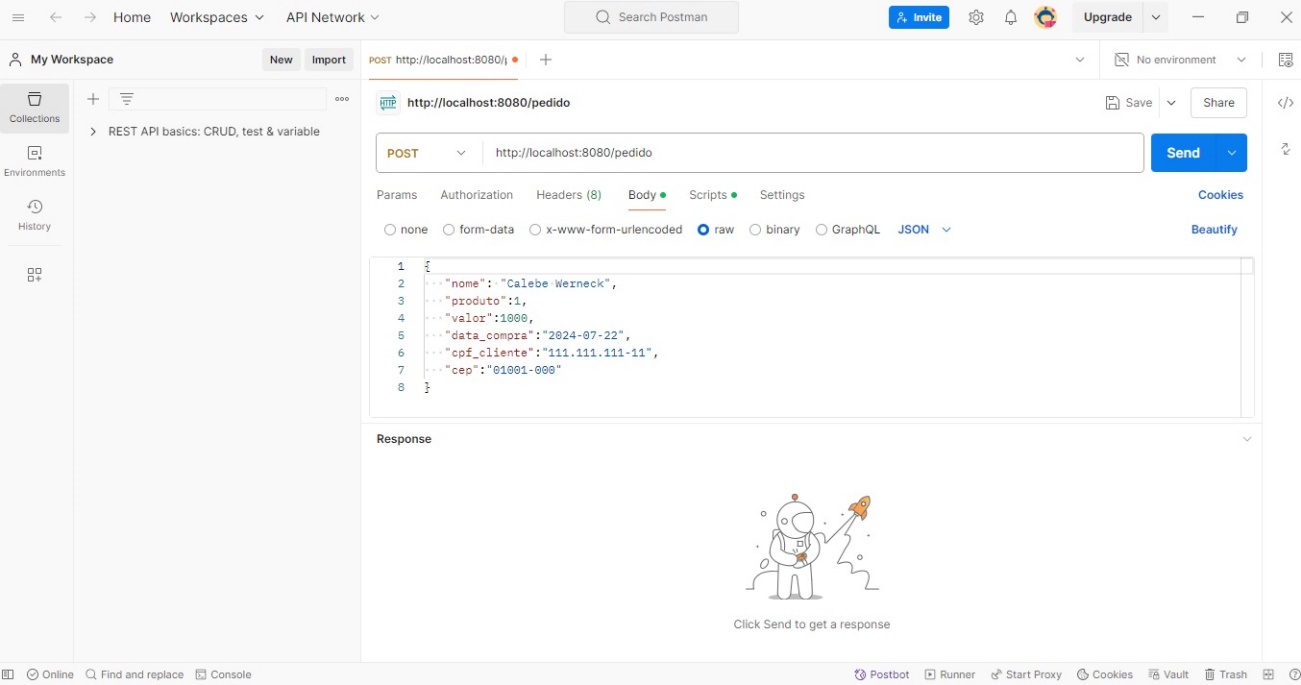
response = requests.delete("https://api.exemplo.com/usuarios/1")

uso do postman para testar os endpoinst ou Swagger (<https://cwc3d.net/imc/swagger-ui.html>)

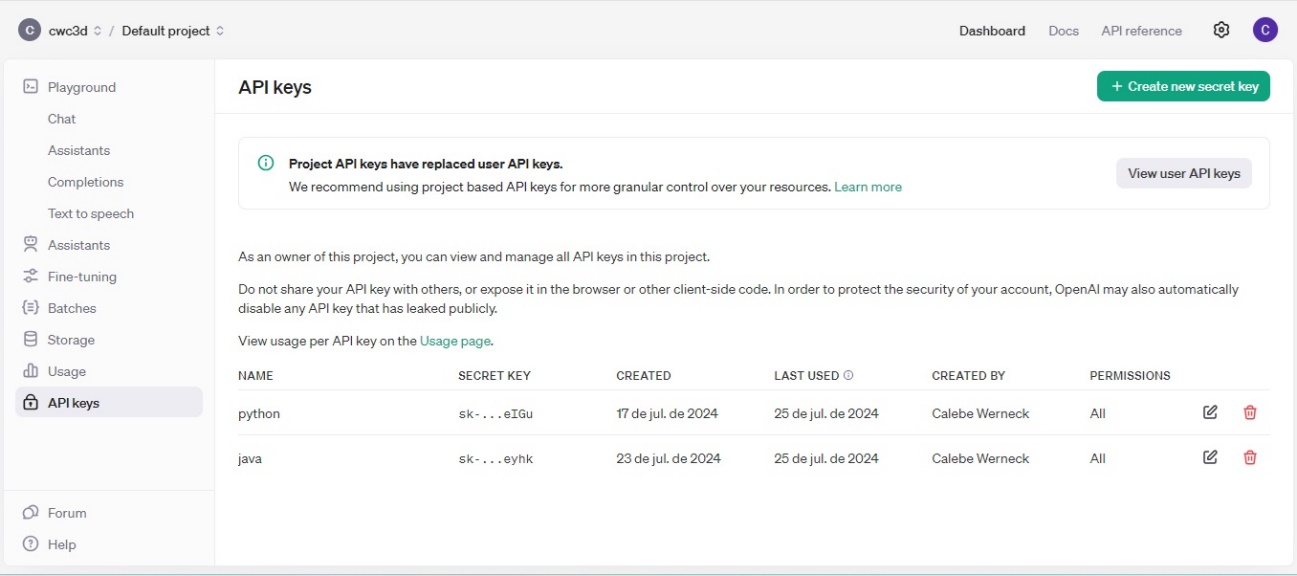
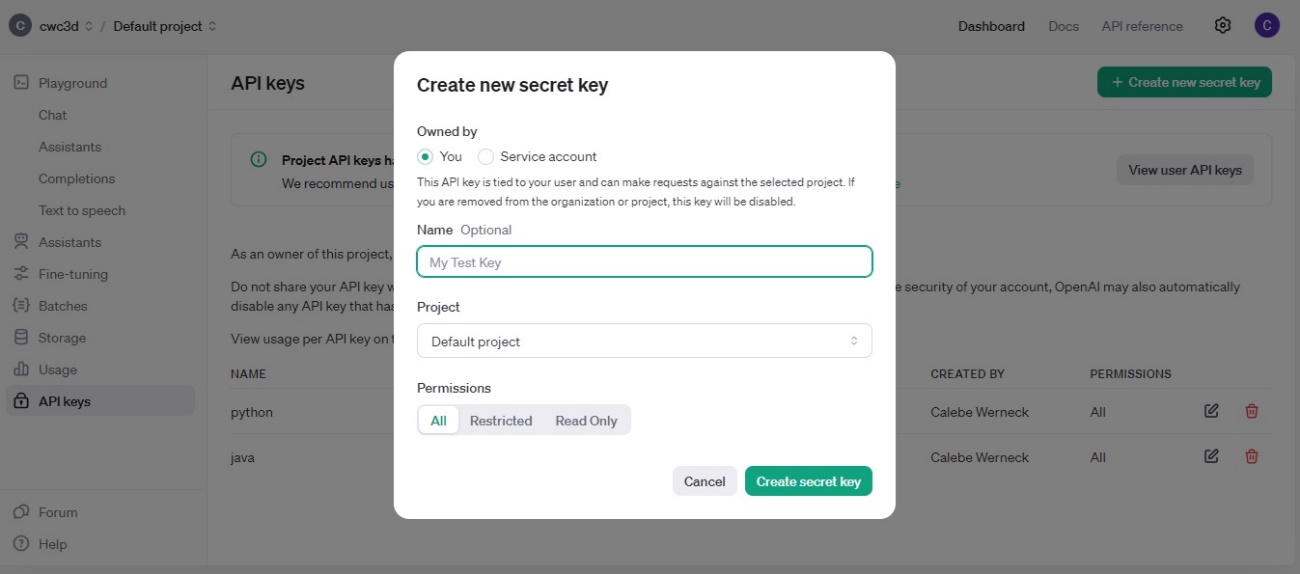
SWAGGER



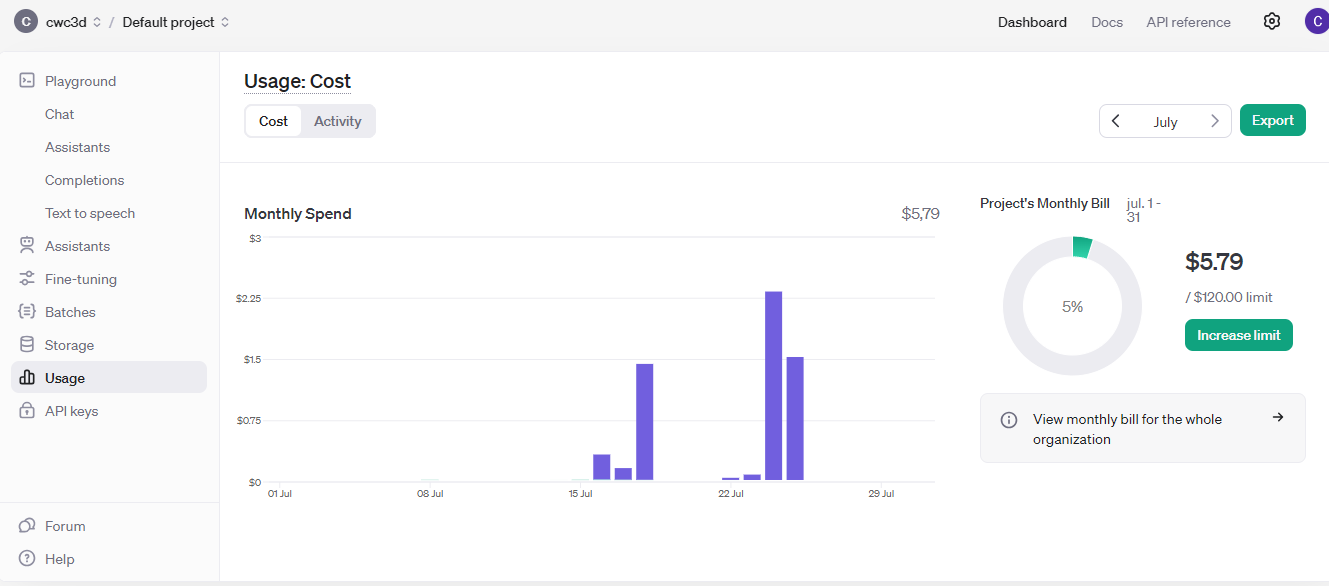
POSTMAN



**5.2 Configuração da API Key da OpenAI**

* **Criar Conta:**
  + Vá para https://openai.com/ e crie uma conta.
* **Obter Chave da API:**
* https://platform.openai.com/api-keys
  + **Acesse a seção de API Keys e copie sua chave de API.**
  + ****
  + ****

**5.3 Desenvolvimento do Código Python**

* **O que são Tokens?**
  + Unidades de texto para processamento. Aproximadamente 1 token é igual a 4 caracteres ou 0,75 palavras em inglês.
  + Ver Custo.... O que já foi utilizado...
  + https://platform.openai.com/usage
  + 

<https://platform.openai.com/docs/api-reference/introduction>

Documentação principal com Exemplos de implementação Python e demais Linguagens...

**Exemplos:**

** Create Speech (POST https://api.openai.com/v1/audio/speech)**

* O que faz: Converte texto em áudio.
* Descrição: Este endpoint gera um arquivo de áudio a partir de um texto fornecido. Você envia o texto para a API, e a resposta é um arquivo de áudio que lê o texto em voz alta.

** Create Transcription (POST https://api.openai.com/v1/audio/transcriptions)**

* O que faz: Transcreve áudio para texto.
* Descrição: Este endpoint converte um arquivo de áudio em texto. Você envia um arquivo de áudio para a API, e a resposta é o texto transcrito do áudio.

** Create Translation (POST https://api.openai.com/v1/audio/translations)**

* O que faz: Traduz áudio para o inglês.Descrição: Este endpoint traduz o áudio enviado para o inglês. Você envia um arquivo de áudio em um idioma específico, e a resposta é o texto traduzido para o inglês.

**Estrutura Codigo Python + Openai**

**pip install openai==0.28**

**pip install python-dotenv**

**import openai**

**from dotenv import load\_dotenv**

**import os**

**# Carregar variáveis de ambiente do arquivo .env**

**load\_dotenv()**

**# Obter a chave API do ambiente**

**openai.api\_key = os.getenv("OPENAI\_API\_KEY")**

**def chat\_with\_gpt(prompt):**

**try:**

**response = openai.ChatCompletion.create(**

**model="gpt-3.5-turbo",**

**messages=[**

**{"role": "user", "content": prompt}**

**]**

**)**

**return response.choices[0].message['content']**

**except Exception as e:**

**return f"Error: {e}"**

**def main():**

**while True:**

**prompt = input("Usuario: ")**

**if prompt.lower() in ["exit", "quit"]:**

**print("Exiting chat...")**

**break**

**response = chat\_with\_gpt(prompt)**

**print(f"GPT-3.5: {response}")**

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

**main()**

**criar arquivo .env e salvar a chave.**

**OPENAI\_API\_KEY=”suachave api”**

**VIA CEP**

O ViaCEP é um serviço gratuito e público de consulta de endereços brasileiro que fornece informações sobre a localização de endereços a partir de um Código de Endereçamento Postal (CEP). A API do ViaCEP é amplamente utilizada em aplicações para obter dados sobre endereços de maneira rápida e confiável.

Principais Características do ViaCEP

1. **Consulta de CEP:**
   * Permite a consulta de informações sobre um endereço a partir de um CEP.
   * Retorna detalhes como logradouro, bairro, cidade, estado, e complemento.
2. **Formato de Resposta:**
   * A resposta da API é geralmente no formato JSON, que inclui informações detalhadas sobre o endereço consultado.
   * Exemplo de resposta JSON:

**{**

**"cep": "01001-000",**

**"logradouro": "Praça da Sé",**

**"bairro": "Sé",**

**"localidade": "São Paulo",**

**"uf": "SP",**

**"unidade": "",**

**"ibge": "3550308",**

**"gia": "1004"**

**}**

1. **Endpoints da API:**
   * O endpoint básico para consultar um CEP é https://viacep.com.br/ws/{cep}/json/, onde {cep} deve ser substituído pelo CEP que você deseja consultar.
   * **Exemplos de uso:**
     + **Consulta direta:** https://viacep.com.br/ws/01001000/json/
     + **Consulta com um CEP inválido ou inexistente retorna um** erro: https://viacep.com.br/ws/00000000/json/
2. **Funcionalidade Adicional:**
   * A API também pode ser usada para consultar endereços em formato XML, embora o formato JSON seja o mais comum.

**Como Funciona**

Quando você faz uma requisição para o ViaCEP com um CEP específico, o serviço consulta sua base de dados e retorna as informações correspondentes ao endereço associado a esse CEP. Isso é útil para:

* Validação de Endereço: Verificar se um CEP é válido e obter informações de endereço.
* Preenchimento Automático: Auxiliar no preenchimento automático de formulários com base no CEP inserido pelo usuário.
* Correção de Dados: Corrigir endereços com base em dados precisos fornecidos pelo ViaCEP.

**import requests**

**def consulta\_cep(cep):**

**# Verifica se o CEP tem 8 dígitos**

**if len(cep) != 8:**

**print("CEP inválido. O CEP deve ter exatamente 8 dígitos numéricos.")**

**return**

**# Faz a requisição à API ViaCEP**

**url = f"https://viacep.com.br/ws/{cep}/json/"**

**response = requests.get(url)**

**# Verifica o status da resposta**

**if response.status\_code == 200:**

**data = response.json()**

**# Verifica se o CEP retornou dados válidos**

**if "erro" in data:**

**print("CEP não encontrado.")**

**else:**

**print(f"CEP: {data.get('cep')}")**

**print(f"Logradouro: {data.get('logradouro')}")**

**print(f"Bairro: {data.get('bairro')}")**

**print(f"Cidade: {data.get('localidade')}")**

**print(f"Estado: {data.get('uf')}")**

**print(f"Complemento: {data.get('complemento', 'Não informado')}")**

**else:**

**print("Erro ao acessar a API. Verifique sua conexão com a Internet.")**

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

**# Solicita o CEP ao usuário**

**cep\_usuario = input("Digite o CEP (somente números e 8 dígitos): ")**

**consulta\_cep(cep\_usuario)**