

# Aplicação de Integração de APIs com Flask

Este exemplo de aplicação demonstra como integrar a API do GitHub usando Python e Flask. A aplicação permite que o usuário insira um nome de usuário do GitHub e veja a lista de repositórios públicos desse usuário.

## 1. INSTALAR **python**

Baixar do site oficial [aqui](#)

## 2. Adicionar **VARIÁVEIS**

Adicionar nas variáveis de ambiente

```
C:\Users\Marcio Fernando Maia\AppData\Local\Programs\Python\Python312\Scripts\
```

```
C:\Users\Marcio Fernando Maia\AppData\Local\Programs\Python\Python312\
```

```
C:\Users\Marcio Fernando Maia\AppData\Local\Programs\Python\Launcher\
```

Ou simplesmente remita executar Python e pip a partir de qualquer diretório no prompt de comando.

```
set PATH=%PATH%;C:\Python39\Scripts;C:\Python39
```

## 3. Instalar **DEPENDÊNCIAS** (Depende para aqual uso.)

```
pip install Flask requests
```

## 4. **ATUALIZAR** o python

```
python -m pip install --upgrade pip
```

## 5. **VERIFICAR** Instalação do python

```
python --version
```

## 6. Ambiente **VIRTUAL** no python

Veja o arquivo [ambientes\\_virtuais\\_python.html](#)

# MÃO NA MASSA

# Estrutura do Projeto

```
github_integration/  
├── app.py  
├── templates/  
│   └── index.html  
└── requirements.txt
```

## Código HTML para index.html

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
  <meta charset="UTF-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1.0">  
  <title>GitHub Repositories</title>  
<style>  
  body {  
    font-family: Arial, sans-serif;  
    margin: 0;  
    padding: 20px;  
    background-color: #f4f4f4;  
  }  
  h1 {  
    color: #333;  
  }  
  form {  
    margin-bottom: 20px;  
  }  
  label {  
    display: block;  
    margin-bottom: 10px;  
    font-weight: bold;  
  }  
  input[type="text"] {  
    padding: 10px;  
    width: 300px;  
    margin-bottom: 10px;  
    border: 1px solid #ddd;  
    border-radius: 4px;  
  }  
  button {  
    padding: 10px 15px;  
    background-color: #28a745;  
    color: white;  
    border: none;  
    border-radius: 4px;  
    cursor: pointer;  
  }  
  button:hover {  
    background-color: #218838;  
  }  
  p {  
    color: red;  
  }  
  ul {  
    list-style-type: none;  
    padding: 0;  
  }  
  li {  
    background-color: #fff;  
    margin-bottom: 10px;  
    padding: 10px;  
    border-radius: 4px;
```

```

        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    }
    li a {
        text-decoration: none;
        color: #007bff;
    }
    li a:hover {
        text-decoration: underline;
    }
</style>
</head>
<body>
    <h1>Buscar Repositórios do GitHub</h1>
    <form method="post">
        <label for="username">Nome de Usuário:</label>
        <input type="text" id="username" name="username" required>
        <button type="submit">Buscar</button>
    </form>
    {% if error %}
        <p>{{ error }}</p>
    {% endif %}
    {% if repos %}
        <h2>Repositórios de {{ repos[0].owner.login }}</h2>
        <ul>
            {% for repo in repos %}
                <li><a href="{{ repo.html_url }}" target="_blank">{{ repo.name }}</li>
            {% endfor %}
        </ul>
    {% endif %}
</body>
</html>

```

## Código Python para app.py

```

from flask import Flask, render_template, request
import requests

app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])
def index():
    repos = []
    error = None
    if request.method == "POST":
        username = request.form.get("username")
        url = f"https://api.github.com/users/{username}/repos"
        response = requests.get(url)

        if response.status_code == 200:
            repos = response.json()
        else:
            error = f"Não foi possível encontrar repositórios para o usuário {username}"

    return render_template("index.html", repos=repos, error=error)

if __name__ == "__main__":
    app.run(debug=True)

```

## Arquivo requirements.txt

```

Flask
requests

```

# Como Executar a Aplicação

Siga os passos abaixo para executar a aplicação:

1. Crie a estrutura de arquivos conforme mostrado acima.
2. Adicione os códigos fornecidos nos arquivos correspondentes.
3. Execute a aplicação com o comando:

```
python app.py
```

4. Acesse `http://127.0.0.1:5000/` no seu navegador para ver a aplicação em funcionamento.

Esse exemplo cria uma interface simples para buscar repositórios de um usuário no GitHub e exibir os resultados na página.

Todos os direitos reservados - 2024 - Márcio Fernando Maia