

Teste da Skallar Digital para Desenvolvedor Sênior

Uester Jacinto Lacerda da Silva

Documentação Swagger

<http://127.0.0.1:5000/apidocs/>

default

POST /send-email Envia um e-mail para o destinatário especificado. post_send_email

Essa rota envia um e-mail utilizando as informações fornecidas no corpo da requisição. É necessário fornecer o endereço de e-mail do destinatário, o assunto e o corpo do e-mail.

Parameters

Try it out

Name	Description
body <small>* required</small> object (body)	<div>Example Value Model</div> <pre>{ "body": "Conteúdo do e-mail", "subject": "Assunto do E-mail", "to": "destinatario@example.com"} </pre> <div>Parameter content type application/json</div>

Responses

Response content type application/json

Code	Description
200	<div>E-mail enviado com sucesso!</div> <div>Example Value </div> <pre>{ "message": "E-mail enviado com sucesso!"}</pre>
400	<div>Erro ao enviar o e-mail.</div> <div>Example Value </div> <pre>{ "message": "Erro ao enviar o e-mail."}</pre>

Powered by [Flasgger 0.9.7.1](#)

Teste da Skallar Digital para Desenvolvedor Sênior

Uester Jacinto Lacerda da Silva

Executando a Aplicação

Clique duas vezes no arquivo **run.bat** para executar a aplicação ou abra o cmd do Windows e proceda da seguinte forma:

1) Entrar na pasta email_service

cd "C:\Teste Skallar\email_service"

2) Ativar o ambiente virtual

venv\Scripts\activate

3) Instalar as dependências

pip install -r requirements.txt

4) Executar a aplicação

python app.py

```
Microsoft Windows [versão 10.0.22621.4169]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\ueste>cd "C:\Teste Skallar\email_service"

C:\Teste Skallar\email_service>venv\Scripts\activate

(venv) C:\Teste Skallar\email_service>pip install -r requirements.txt
Requirement already satisfied: Flask in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from -r requirements.txt (line 1)) (3.0.3)
Requirement already satisfied: Flask-RESTful in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from -r requirements.txt (line 2)) (0.3.10)
Requirement already satisfied: pytest in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from -r requirements.txt (line 3)) (8.3.3)
Requirement already satisfied: python-dotenv in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from -r requirements.txt (line 4)) (1.0.1)
Requirement already satisfied: flasker in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from -r requirements.txt (line 5)) (0.9.7.1)
Requirement already satisfied: Werkzeug>=3.0.0 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask->-r requirements.txt (line 1)) (3.0.4)
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask->-r requirements.txt (line 1)) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask->-r requirements.txt (line 1)) (2.2.0)
Requirement already satisfied: click>=8.1.3 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask->-r requirements.txt (line 1)) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask->-r requirements.txt (line 1)) (1.8.2)
Requirement already satisfied: aniso8601>=8.0.2 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask-RESTful->-r requirements.txt (line 2)) (9.0.1)
Requirement already satisfied: six>=1.3.0 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask-RESTful->-r requirements.txt (line 2)) (1.16.0)
Requirement already satisfied: pytz in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Flask-RESTful->-r requirements.txt (line 2)) (2024.2)
Requirement already satisfied: packaging in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from pytest->-r requirements.txt (line 3)) (24.2)
Requirement already satisfied: pluggy>=2.1 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from pytest->-r requirements.txt (line 3)) (1.5.0)
Requirement already satisfied: colorama in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from pytest->-r requirements.txt (line 3)) (0.4.6)
Requirement already satisfied: PyYAML>=3.0 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from flasker->-r requirements.txt (line 5)) (6.0.2)
Requirement already satisfied: jsonschema>=3.0.1 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from flasker->-r requirements.txt (line 5)) (4.23.0)
Requirement already satisfied: mistune in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from flasker->-r requirements.txt (line 5)) (3.0.2)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from Jinja2>=3.1.2->Flask->-r requirements.txt (line 1)) (2.0.0)
Requirement already satisfied: attrs>=22.2.0 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from jsonschema>=3.0.1->flasker->-r requirements.txt (line 5)) (24.2.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from jsonschema>=3.0.1->flasker->-r requirements.txt (line 5)) (2024.10.1)
Requirement already satisfied: referencing>=0.28.4 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from jsonschema>=3.0.1->flasker->-r requirements.txt (line 5)) (0.35.1)
Requirement already satisfied: rpds-py>=0.7.1 in c:\users\ueste\onedrive\área de trabalho\trabalho\teste skallar\email_service\venv\lib\site-packages (from jsonschema>=3.0.1->flasker->-r requirements.txt (line 5)) (0.20.0)

(venv) C:\Teste Skallar\email_service>python app.py
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5080
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 556-192-838
```

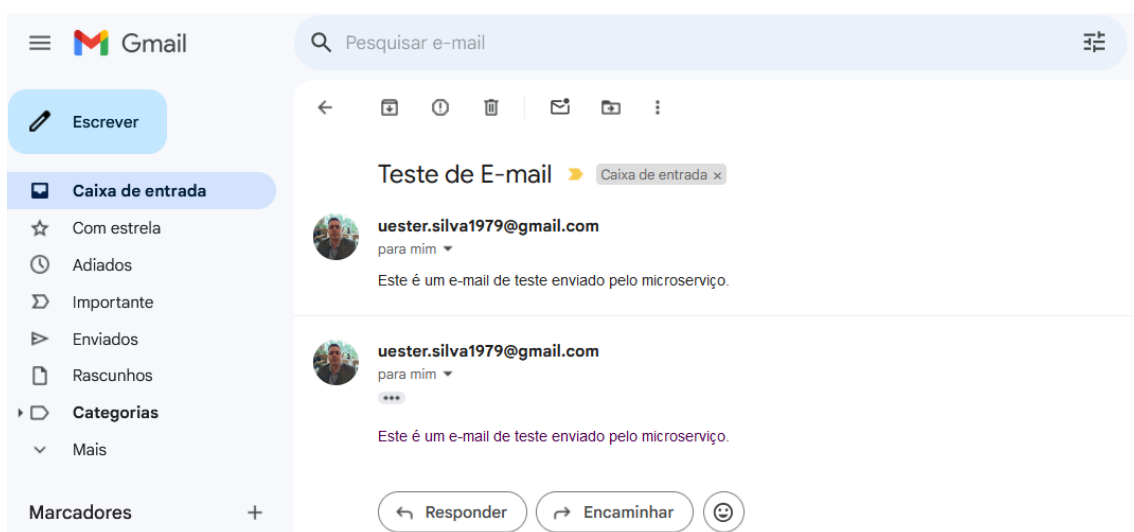
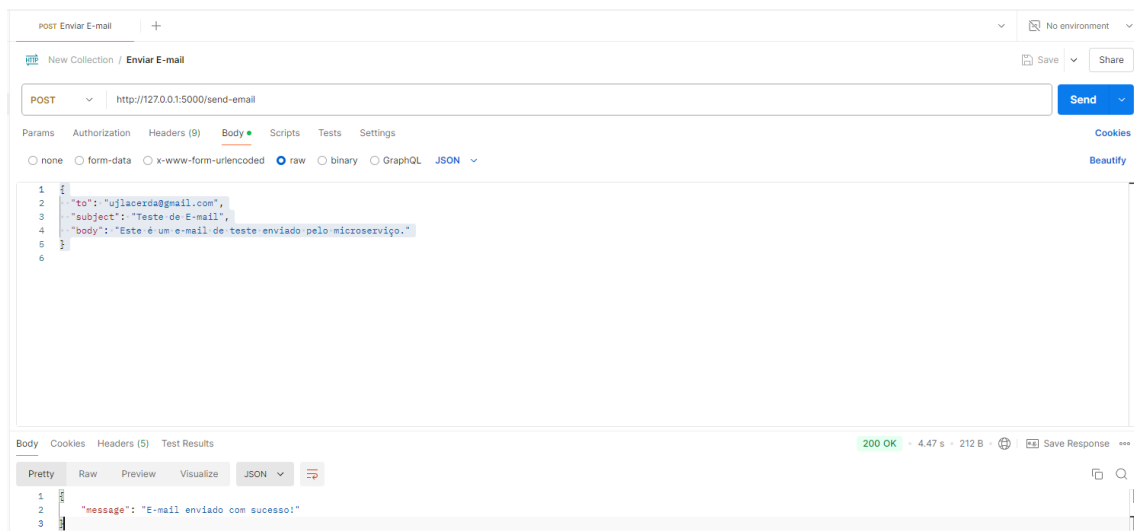
Teste da Skallar Digital para Desenvolvedor Sênior

Uester Jacinto Lacerda da Silva

Postman

<http://127.0.0.1:5000/send-email>

```
{
  "to": "ujlacerda@gmail.com",
  "subject": "Teste de E-mail",
  "body": "Este é um e-mail de teste enviado pelo microserviço."
}
```



Estrutura de Pastas e Arquivos

```
email_service/
├── app.py           # Arquivo principal que inicializa a aplicação Flask
├── .env             # Arquivo para variáveis de ambiente
├── requirements.txt # Lista de dependências
├── controllers/
│   └── email_controller.py # Controlador de e-mail
├── routes/
│   └── email_routes.py    # Definição das rotas da API
├── services/
│   └── email_service.py   # Serviço que realiza o envio de e-mails
├── docs/                 # Diretório para armazenar os arquivos de documentação
├── email_send.yml        # Arquivo de documentação para a rota /send-email
├── tests/
│   └── test_email.py     # Testes unitários para verificar a funcionalidade
└── venv/                 # Ambiente virtual (gerado pelo Python)
```

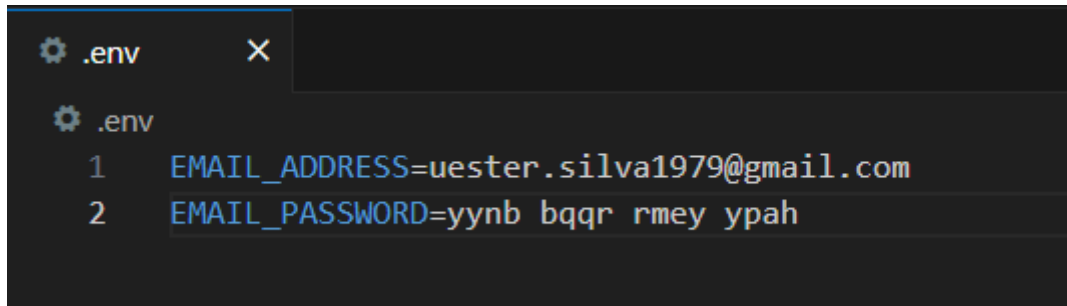
Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

app.py

```
app.py 1 X
app.py > ...
1  from flask import Flask
2  from flasgger import Swagger
3  from routes.email_routes import email_blueprint
4
5  # Inicializa a aplicação Flask
6  app = Flask(__name__)
7
8  # Inicializa o Swagger e define algumas informações básicas
9  swagger = Swagger(app, template={
10      "swagger": "2.0",
11      "info": {
12          "title": "Microserviço de Envio de E-mails",
13          "description": "Documentação para a API de envio de e-mails utilizando Flask",
14          "version": "1.0.0"
15      },
16      "basePath": "/",
17      "schemes": [
18          "http",
19          "https"
20      ]
21  })
22
23  # Registra o Blueprint da rota de e-mail
24  app.register_blueprint(email_blueprint)
25
26  # Ponto de entrada da aplicação
27  if __name__ == "__main__":
28      app.run(debug=True)
```

Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

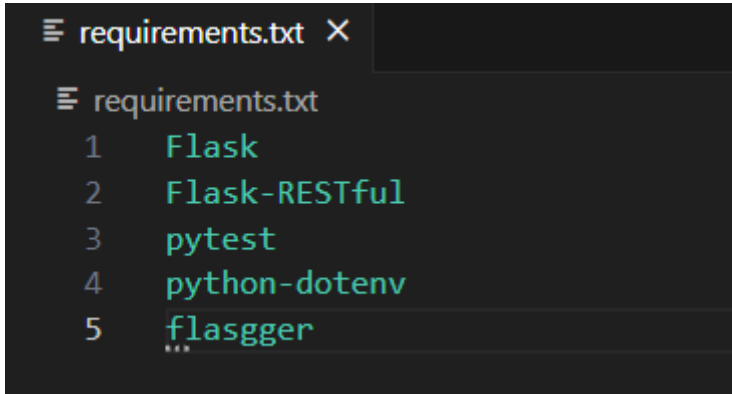
.env

A screenshot of a code editor with a dark theme. The editor has a tab labeled '.env' with a gear icon and a close button. The file content is as follows:

```
1 EMAIL_ADDRESS=uester.silva1979@gmail.com
2 EMAIL_PASSWORD=yynb bqqr rmey ypah
```

* O e-mail foi criado exclusivamente para testar a aplicação. Foi gerada uma senha de aplicativo específica para esse propósito, que não corresponde à senha real do e-mail.

requirements.txt

A screenshot of a code editor window. The title bar at the top shows a hamburger menu icon, the filename 'requirements.txt', and a close button 'X'. The editor area has a dark background with light green text. It shows a list of five dependencies, each preceded by a line number from 1 to 5. The dependencies are 'Flask', 'Flask-RESTful', 'pytest', 'python-dotenv', and 'flasgger'. The text '...' is visible at the end of the fifth line.

```
1  Flask
2  Flask-RESTful
3  pytest
4  python-dotenv
5  flasgger
...
```

controllers/email_controller.py

```
email_controller.py X
controllers > email_controller.py > ...
1  from services.email_service import send_email_service
2
3  def send_email(to, subject, body):
4      try:
5          # Chama o serviço de envio de e-mail
6          send_email_service(to, subject, body)
7          return "E-mail enviado com sucesso!"
8      except Exception as e:
9          return f"Erro ao enviar o e-mail: {str(e)}"
10
```


Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

routes/email_routes.py

```
email_routes.py x
routes > email_routes.py > send
1  from flask import Blueprint, request, jsonify
2  from flasgger import swag_from
3  from controllers.email_controller import send_email
4
5  # Define o Blueprint para as rotas de e-mail
6  email_blueprint = Blueprint('email_blueprint', __name__)
7
8  # Rota para envio de e-mails
9  @email_blueprint.route('/send-email', methods=['POST'])
10 @swag_from('../docs/email_send.yml') # Carrega a documentação a partir do arquivo YAML
11 def send():
12     data = request.get_json()
13     to = data.get("to")
14     subject = data.get("subject")
15     body = data.get("body")
16
17     # Chama o controlador para enviar o e-mail
18     result = send_email(to, subject, body)
19
20     return jsonify({"message": result})
```

Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

services/email_service.py

```
email_service.py X
services > email_service.py > send_email_service
1  import smtplib
2  from email.mime.text import MIMEText
3  from email.mime.multipart import MIMEMultipart
4  from dotenv import load_dotenv
5  import os
6
7  # Carrega as variáveis de ambiente do arquivo .env
8  load_dotenv()
9
10 # Variáveis de configuração de e-mail
11 EMAIL_ADDRESS = os.getenv("EMAIL_ADDRESS")
12 EMAIL_PASSWORD = os.getenv("EMAIL_PASSWORD")
13
14 def send_email_service(to, subject, body):
15     """
16     Função para enviar um e-mail utilizando o protocolo SMTP.
17     Parâmetros:
18     - to (str): Endereço de e-mail do destinatário.
19     - subject (str): Assunto do e-mail.
20     - body (str): Corpo do e-mail.
21     """
22     # Cria a mensagem de e-mail
23     msg = MIMEMultipart()
24     msg['From'] = EMAIL_ADDRESS
25     msg['To'] = to
26     msg['Subject'] = subject
27
28     # Anexa o corpo do e-mail na mensagem
29     msg.attach(MIMEText(body, 'plain'))
30
31     # Conecta ao servidor SMTP e envia o e-mail
32     try:
33         with smtplib.SMTP('smtp.gmail.com', 587) as server:
34             server.starttls() # Inicializa o TLS para segurança
35             server.login(EMAIL_ADDRESS, EMAIL_PASSWORD) # Faz login no servidor
36             server.sendmail(EMAIL_ADDRESS, to, msg.as_string()) # Envia o e-mail
37             print(f"E-mail enviado com sucesso para {to}")
38     except Exception as e:
39         print(f"Erro ao enviar o e-mail: {e}")
```

Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

docs/email_send.yml

```
! email_send.yml X
docs > ! email_send.yml
1  summary: "Envia um e-mail para o destinatário especificado."
2  description: |
3      Essa rota envia um e-mail utilizando as informações fornecidas no corpo da requisição.
4      É necessário fornecer o endereço de e-mail do destinatário, o assunto e o corpo do e-mail.
5  parameters:
6      - name: "body"
7        in: "body"
8        required: True
9        schema:
10         type: "object"
11         properties:
12             to:
13                 type: "string"
14                 example: "destinatario@example.com"
15                 description: "O endereço de e-mail do destinatário."
16             subject:
17                 type: "string"
18                 example: "Assunto do E-mail"
19                 description: "O assunto do e-mail a ser enviado."
20             body:
21                 type: "string"
22                 example: "Conteúdo do e-mail"
23                 description: "O corpo do e-mail a ser enviado."
24  responses:
25      200:
26          description: "E-mail enviado com sucesso!"
27          examples:
28              application/json:
29                  message: "E-mail enviado com sucesso!"
30      400:
31          description: "Erro ao enviar o e-mail."
32          examples:
33              application/json:
34                  message: "Erro ao enviar o e-mail."
```

Teste da Skallar Digital para Desenvolvedor Sênior
Uester Jacinto Lacerda da Silva

tests/test_email.py

```
test_email.py X
tests > test_email.py > test_send_email
1  import pytest
2  from app import app
3
4  @pytest.fixture
5  def client():
6      with app.test_client() as client:
7          yield client
8
9  def test_send_email(client):
10     response = client.post('/send-email', json={
11         "to": "ujlacerda@gmail.com",
12         "subject": "Teste de E-mail",
13         "body": "Este é um e-mail de teste."
14     })
15     assert response.status_code == 200
16     assert response.get_json()["message"] == "E-mail enviado com sucesso!"
```