



# COMO ENVIAR EMAIL UTILIZANDO PYTHON

Neste ebook você vai aprender a automatizar o envio de emails utilizando Python e o módulo smtplib.

Gere ebooks como este com



Ebookr

em <https://ebookr.ai>

# Crie ebooks profissionais incríveis em minutos com IA



Chega de formatar texto no Google Docs, Word ou ferramentas que só te fazem perder tempo...

E deixe que nossa IA faça o trabalho pesado!



Capas gerados por IA



Infográficos feitos por IA



Edite em Markdown em Tempo Real



Adicione Banners Promocionais

TESTE AGORA



PRIMEIRO CAPÍTULO 100% GRÁTIS

Salve salve Pythonista!

Enviar emails é uma tarefa muito útil em várias aplicações, como envio de notificações, newsletters e mensagens automatizadas.

Felizmente, o Python oferece uma biblioteca chamada `smtplib` que facilita muito o envio de emails por meio do protocolo **SMTP** (*Simple Mail Transfer Protocol*).

Neste artigo, vamos explorar como enviar email no Python.

Vamos nessa!

## Configurando o ambiente

Antes de começar a enviar emails no Python, é necessário configurar o ambiente.

Você precisa ter uma conta de email válida com as informações do seu servidor de SMTP.

Geralmente, essas informações incluem:

- O endereço do servidor SMTP;
- A porta de envio;
- Suas credenciais (usuário e senha).

Por exemplo, para utilizar o GMail do Google para enviar emails, os dados serão:

- Endereço do Servidor SMTP: `smtp.google.com`
- Porta: `587`
- Seu email e senha do Google

# Importando a biblioteca `smtplib`

O primeiro passo é importar a biblioteca `smtplib` para utilizar suas funcionalidades.

Para isso, basta executar o seguinte código:

```
import smtplib
```

Importante ressaltar que a biblioteca `smtplib` faz parte do próprio Python, não necessitando portanto a instalação com o `pip`.

## Estabelecendo uma conexão com o servidor de email

Após importar a biblioteca `smtplib`, é necessário estabelecer uma conexão com o servidor de email.

Para isso, utilize a função `smtplib.SMTP()` e forneça o endereço do servidor SMTP e a porta de envio como argumentos.

Por exemplo:

```
servidor_email = smtplib.SMTP('smtp.gmail.com', 587)
```

No exemplo acima, estamos utilizando o servidor SMTP do Gmail e a porta de envio é a 587.

É importante verificar as informações corretas para o servidor de email que você está utilizando.

# Habilitando a comunicação com o servidor de email

Após estabelecer a conexão com o servidor de email, é necessário habilitar a comunicação usando o método `starttls()`.

Esse método inicia o modo de transporte de camada de segurança (TLS).

Veja o exemplo:

```
servidor_email.starttls()
```

A saída será:

```
(220, b'2.0.0 Ready to start TLS')
```



*Estou construindo o **Ebookr.ai**, uma plataforma onde você cria ebooks profissionais com IA sobre qualquer assunto — do zero ao PDF pronto, com capas e infográficos gerados automaticamente. Dá uma olhada!*

## Crie Ebooks profissionais incríveis em minutos com IA



Chega de formatar texto no Google Docs, Word ou ferramentas que só te fazem perder tempo...

... e deixe que nossa IA faça o trabalho pesado!

**TESTE AGORA! PRIMEIRO CAPÍTULO 100% GRÁTIS** 

 Capas gerados por IA

 Adicione Banners Promocionais

 Edite em Markdown em Tempo Real

 Infográficos feitos por IA

## Autenticando-se no servidor de email

Após habilitar a comunicação, é necessário autenticar-se no servidor de email utilizando suas credenciais de login.

Para isso, utilize o método `login()` fornecendo seu usuário e senha como argumentos.

Por exemplo:

```
servidor_email.login('seu_email@gmail.com', 'sua_senha')
```

# Enviando um email

Agora que você estabeleceu uma conexão com o servidor e autenticou-se com sucesso, é possível enviar um email utilizando o método `sendmail()`.

Esse método recebe três argumentos: o endereço de email remetente, uma lista com os endereços de email destinatários e o conteúdo do email.

Veja um exemplo:

```
remetente = 'seu_email@gmail.com'  
destinatarios = ['destinatario1@gmail.com', 'destinatario2@gmail.com']  
conteudo = 'Olá, este é um email de teste.'  
  
servidor_email.sendmail(remetente, destinatarios, conteudo)
```

No exemplo acima, estamos enviando um email a partir do endereço “`seu_email@gmail.com`” para os destinatários da lista `destinatarios`.

O conteúdo do email é a string “Olá, este é um email de teste.”

## Encerrando a conexão com o servidor de email

Após enviar o email com sucesso, é importante encerrar a conexão com o servidor de email para liberar os recursos.

Utilize o método `quit()` para fazer isso:

```
servidor_email.quit()
```

# Lidando com exceções

Durante o processo de envio de emails, podem ocorrer erros.

Por isso, é importante lidar com exceções para tratar possíveis problemas.

Você pode utilizar um bloco `try-except` para capturar exceções e realizar ações específicas em caso de erro.

Veja um exemplo:

```
try:  
    servidor_email = smtplib.SMTP('smtp.gmail.com', 587)  
    servidor_email.starttls()  
    servidor_email.login('seu_email@gmail.com', 'sua_senha')  
  
    remetente = 'seu_email@gmail.com'  
    destinatarios = ['destinatario1@gmail.com',  
                    'destinatario2@gmail.com']  
    conteudo = 'Olá, este é um email de teste.'  
  
    servidor_email.sendmail(remetente, destinatarios, conteudo)  
except Exception as e:  
    print(f"Erro ao enviar email: {e}")  
finally:  
    servidor_email.quit()
```

No exemplo acima, estamos utilizando um bloco `try-except` para capturar qualquer exceção que ocorrer durante o envio de emails.

Caso ocorra algum erro, o código será executado no bloco `except` e, em seguida, a conexão com o servidor será encerrada.

# Conclusão

Neste artigo, exploramos como enviar emails usando a biblioteca `smtplib` do Python.

Aprendemos a estabelecer uma conexão com o servidor de email, autenticar-se, enviar um email e lidar com exceções.

Agora você tem os conhecimentos necessários para automatizar o envio de emails utilizando o Python.

Experimente essa funcionalidade em suas próximas aplicações e aproveite todos os benefícios que o envio de emails pode trazer.

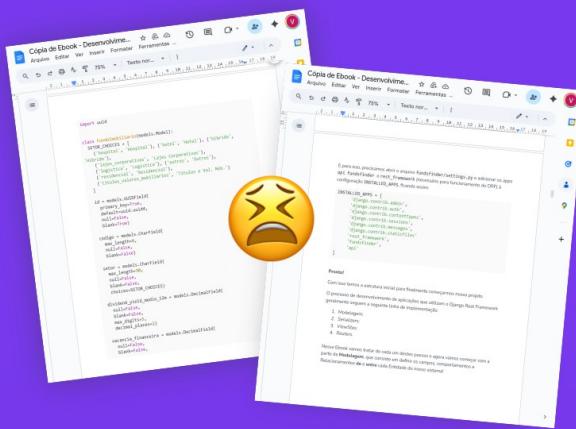
Nos vemos na próxima, Pythonista 

Não se esqueça de conferir!



Ebookr

# Crie Ebooks profissionais em minutos com IA



Chega de formatar código no Google Docs ou Word



**Arquitetura de Software Moderna**

```
import python
import python

class Arquitetura_de_Software_Moderna:
    ...
    def share(self):
        pass
    ...
    return "Arquitetura de Net", "arquitetura"
}

def __init__(self):
    if not self.share():
        raise ValueError("A classe deve implementar o método share()")
    else:
        self._share()

# Exemplo de uso da classe
arquitetura = Arquitetura_de_Software_Moderna()
print(arquitetura.share())

```

**AI-generated system**

A arquitetura com propósito é a mesma software armazena de fusões modernas. Seus sistemas integrados consistem na interface entre os sistemas externos. Chama-se a arquitetura de gerenciamento de sistemas gerados.

```
graph TD
    Motor[Motor gerador] --> Agente[Agente Especializado]
    Agente --> Sistema[Sistema Gerador]
    Sistema --> Orchestrador[Orchestrador]
    Orchestrador --> Motor

```

**Clean layout**

Garantimos que o layout é limpo e organizado para facilitar a leitura e compreensão dos dados.



Capas gerados por IA

Infográficos feitos por IA

Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado

Edite em Markdown em Tempo Real

**TESTE AGORA**

PRIMEIRO CAPÍTULO 100% GRÁTIS