Backend – Documentação da API

Este backend em Python (FastAPI) expõe uma API para classificar emails como **produtivo** ou **improdutivo** e gerar uma resposta automática sugerida a partir do conteúdo. A ideia é servir de camada de inteligência para o frontend em Next.js, que faz o upload do email (texto ou arquivo) e exibe o resultado.

Visão Geral

- **Linguagem:** Python +
- **Framework web:** FastAPI
- **Funções principais:**
  - Receber o conteúdo do email (texto ou arquivo `.txt`/`.pdf`).
  - Pré-processar o texto com técnicas básicas de NLP.
  - Classificar o email em produtivo ou improdutivo usando um modelo de IA.
  - Gerar uma resposta automática adequada à categoria.
  - Devolver os resultados em JSON para o frontend consumir.

Endpoints

`GET /health`

Endpoint simples de health-check, usado para verificar se a API está de pé.

Resposta – 200

```
{
  "status": "ok"
}
POST /classify-email
Endpoint para classificação via texto direto (JSON). Ideal para quando o
usuário cola o conteúdo do email na interface.

URL

text
/classify-email
Método

text
POST
Content-Type

text
application/json
Corpo da requisição
json
```

```
{
  "text": "Olá, gostaria de saber o status da minha solicitação de abertura de conta."
}
```

text (string, obrigatório): conteúdo do email em texto puro.

Resposta – 200

```
{
  "category": "produtivo",
  "suggested_reply": "Olá! Obrigado pelo contato. Já localizei sua solicitação de abertura de conta e ela está em análise. Assim que houver uma atualização, entraremos em contato por este canal.",
  "confidence": 0.9,
  "explanation": null
}
```

category: "produtivo" ou "improdutivo".

suggested_reply: resposta automática gerada pela IA com base no conteúdo e na categoria.

confidence: valor numérico indicando confiança do modelo (pode ser fixo ou derivado).

explanation: campo opcional para futuras explicações da classificação.

Resposta – 400 (exemplo)

```
{
  "detail": "Texto do e-mail é muito curto ou vazio."
}
```

Resposta – 500 (exemplo)

```
{
  "detail": "Erro ao classificar e-mail ou gerar resposta."
}
```

POST /classify-email-file

Endpoint para classificação via upload de arquivo (.txt ou .pdf). O backend cuida de extrair o texto antes de chamar a IA. [web:40][web:45]

URL

text
/classify-email-file
Método

text
POST
Content-Type

text
multipart/form-data
Campos do formulário

file (obrigatório): arquivo de email nos formatos:

text/plain (.txt)

application/pdf (.pdf)

Exemplo de requisição (cURL):

bash
```
curl -X POST http://localhost:8000/classify-email-file \
  -H "accept: application/json" \
  -H "Content-Type: multipart/form-data" \
  -F "file=@email_exemplo.pdf"
```
Resposta – 200 (exemplo com email improdutivo)
json
```
{
  "category": "improdutivo",
  "suggested_reply": "Olá! Muito obrigado pela mensagem e pelos votos.
Desejamos o mesmo em dobro para você!",
  "confidence": 0.9,
  "explanation": null
}
```
Resposta – 400 (exemplos)
Arquivo com tipo inválido:

json
```
{
  "detail": "Apenas arquivos .txt ou .pdf são suportados."
}
```
Arquivo sem texto relevante:

json
```
{
  "detail": "Arquivo não contém texto suficiente."
}
```
Resposta – 500 (exemplo)
json
```
{
  "detail": "Erro ao processar arquivo ou classificar e-mail."
}
```
Fluxo Interno (alto nível)
A API está organizada em camadas para manter o código mais legível e fácil de manter:

Camada de entrada (FastAPI)

Valida o input.

Chama os serviços de NLP + IA.

Formata a saída no modelo de resposta.

NLP

Normaliza o texto (lowercase, remoção de caracteres estranhos).

Remove stop words em português.

Deixa o texto pronto para o modelo atuar.

IA / Classificação

Usa um modelo de linguagem (ex.: OpenAI / Hugging Face) para:

Classificar o email como produtivo ou improdutivo.

Gerar uma resposta automática, levando em conta a categoria:

produtivo: respostas mais orientadas a ação (status, pedidos de informação, etc.).

improdutivo: respostas mais curtas, educadas, sem assumir ações complexas.

Exemplos de Dados de Teste
Para facilitar o desenvolvimento e os testes, separei alguns exemplos que podem ser usados tanto no backend (via cURL/Postman) quanto na interface.

Email produtivo – Suporte/Status
json
```
{
  "text": "Bom dia, fiz uma solicitação de aumento de limite do meu cartão há 5 dias e ainda não tive retorno. Podem me informar o status desse pedido?"
}
```
Expectativa:

category: "produtivo"

suggested_reply: algo explicando o status, prazo de análise ou próximos passos.

Email produtivo – Dúvida sobre sistema
json
```
{
  "text": "Boa tarde, estou tentando acessar o internet banking e está aparecendo uma mensagem de erro ao fazer login. Podem me orientar como resolver?"
}
```
Expectativa:

category: "produtivo"

suggested_reply: instruções básicas, pedido de mais detalhes ou encaminhamento para suporte.

Email improdutivo – Felicitações
json
```
{
```

```
  "text": "Olá, gostaria apenas de desejar um feliz natal e um ótimo ano novo
para toda a equipe!"
}
Expectativa:

category: "improdutivo"

suggested_reply: mensagem curta de agradecimento e votos em retorno.

Email improdutivo – Agradecimento simples
json
{
  "text": "Muito obrigado pelo atendimento de hoje, foi tudo resolvido!"
}
Expectativa:

category: "improdutivo"

suggested_reply: agradecimento breve, sem criar novas tarefas.

Como o Frontend Deve Usar a API
Para texto colado pelo usuário:

Chamar POST /classify-email com JSON contendo text.

Para upload de arquivo .txt ou .pdf:

Chamar POST /classify-email-file com multipart/form-data e campo file.

O frontend só precisa exibir:

Categoria retornada.

Resposta sugerida.

Opcionalmente, algum indicador de confiança.

text
undefined
```