## Visão geral

* Você digita uma pergunta na página de Chat.
* O site envia essa pergunta para um endereço interno: /api/gemini.
* O servidor do Next.js recebe essa requisição, chama a API do Google (Gemini) com a sua chave secreta e devolve a resposta para o navegador.
* A página de Chat mostra a resposta da IA na tela.

Importante: a chave da API fica no servidor (segura), não no navegador.

## Fluxo da requisição (passo a passo)

* Usuário escreve e clica em “Enviar”.
* O navegador faz um POST para `/api/gemini` com um JSON, por exemplo:

{ "message": "Explique fotossíntese", "history": [] } { "message": "Explique fotossíntese", "history": [] }

* No servidor, a rota `/api/gemini` lê esse JSON e chama a API do Google:

https://generativelanguage.googleapis.com/.../models/gemini-2.0-flash:generateContent?key=SUA\_CHAVE https://generativelanguage.googleapis.com/.../models/gemini-2.0-flash:generateContent?key=SUA\_CHAVE

* O Google Gemini devolve um JSON com o texto gerado.
* O servidor repassa esse texto para o navegador.
* A tela atualiza mostrando a resposta no chat.

## Onde isso acontece no projeto

* Rota interna do site (server): app/api/gemini/route.js
  + Recebe o POST do navegador
  + Lê a variável de ambiente GEMINI\_API\_KEY
  + Chama a API do Google
  + Retorna só o texto para a página
* Página do chat (client): app/chat/page.jsx
  + Quando você envia uma mensagem, usa fetch("/api/gemini", { method: "POST", body: ... })
  + Recebe a resposta e mostra na lista de mensagens
* Variáveis de ambiente: .env.local
  + Exemplo:

```

GEMINI\_API\_KEY=coloque\_sua\_chave\_aqui

```

* + Nunca enviar para o repositório público

## O que são “routes” no Next.js?

* “Routes” são as rotas/endpoints do seu site.
* No App Router (pasta app/), cada pasta normalmente vira uma rota.
  + app/page.jsx → rota / (página inicial)
  + app/projeto/page.jsx → rota /projeto
  + app/sobre/page.jsx → rota /sobre
* Para APIs internas, usamos app/api/.../route.js.
  + app/api/gemini/route.js → rota /api/gemini
  + Essas rotas rodam no servidor (não aparecem para o usuário) e podem usar segredos.

Por que usar uma rota interna? Para esconder a chave da API. O navegador fala com o seu servidor, e só o servidor fala com o Google.

## O que é JSX (bem simples)

* JSX é a “linguagem” usada dentro do React para montar a tela.
* Parece HTML, mas é JavaScript escrevendo HTML.
* Exemplo básico:

function Titulo() { return <h1>Olá, mundo!</h1>; } function Titulo() { return <h1>Olá, mundo!</h1>; }

* Você pode usar chaves { } para colocar valores dinâmicos:

const nome = "João"; return <p>Bem-vindo, {nome}!</p>; const nome = "João"; return <p>Bem-vindo, {nome}!</p>;

* No Next.js (App Router), cada arquivo page.jsx retorna um JSX que é a interface daquela página.

## Dúvidas comuns

* Minha chave aparece no navegador?
  + Não, ela é usada somente no servidor, nas rotas /api/....
* Posso mudar o modelo da IA?
  + Sim, na URL da API no arquivo app/api/gemini/route.js (modelo gemini-2.0-flash).
* Posso reaproveitar esse padrão para outras APIs?
  + Sim. Crie outra rota em app/api/minhaapi/route.js e chame o serviço externo lá dentro.

## Resumo

* A página de chat envia pergunta → /api/gemini (servidor).
* O servidor chama o Google Gemini com sua chave.
* A resposta volta e aparece na tela.
* Rotas de página (page.jsx) montam as telas com JSX.
* Rotas de API (route.js) protegem segredos e conversam com serviços externos.