

```
n says hello and asks for m
world!')
s your name?') # ask for th
t()
good to meet you, ' + myNam
length of your name is:')
ame))
s your age?') # ask for the
()
ll be ' + str(int(myAge) +
```

FastAPI

Bem-vindos à nossa jornada de exploração do FastAPI! Descubra um framework Python moderno e poderoso para a construção de APIs eficientes e escaláveis.

```
if "fotovirs" <? if ($_COOKIE['lang'] =='eng'){
    echo "Photo gallery";}
elseif ($_COOKIE['lang'] =='rus') {
    echo "Фотогалерея";
}
else {
    echo "Foto galerija";
}
</h3>-->
<?if($_GET[type]==1||!$_GET[type])echo"current"?
if="foto-galerija.php?type=1#text_margin">


<div id="left_ico"> </div>
<p <?if($_COOKIE['lang'] =='rus')echo "style"?
E['lang'] =='eng'){
    'Wood-frame houses';
}
if($_COOKIE['lang'] =='rus'){
    'Деревянные каркасные дома';
}
    'Koka karkasa mājas';


```

Configuração e instalação do FastAPI

- 1 Instalação do FastAPI**
Utilize o comando pip para instalar o FastAPI e suas dependências.
- 2 Criar um projeto FastAPI**
Crie um arquivo Python (ex: main.py) e importe a biblioteca FastAPI.
- 3 Configurar o servidor Uvicorn**
Use o servidor Uvicorn para executar o aplicativo FastAPI e tornar sua API acessível.

Criação de Rotas GET: Exibir dados

Definir uma rota GET

Utilize a função `@app.get("/")` para criar uma rota GET que responde à raiz do aplicativo.

Crie um endpoint para exibir dados com a função ``@app.get("/")``.

Criar uma função de resposta

Defina uma função que retorna dados no formato JSON, contendo as informações desejadas.

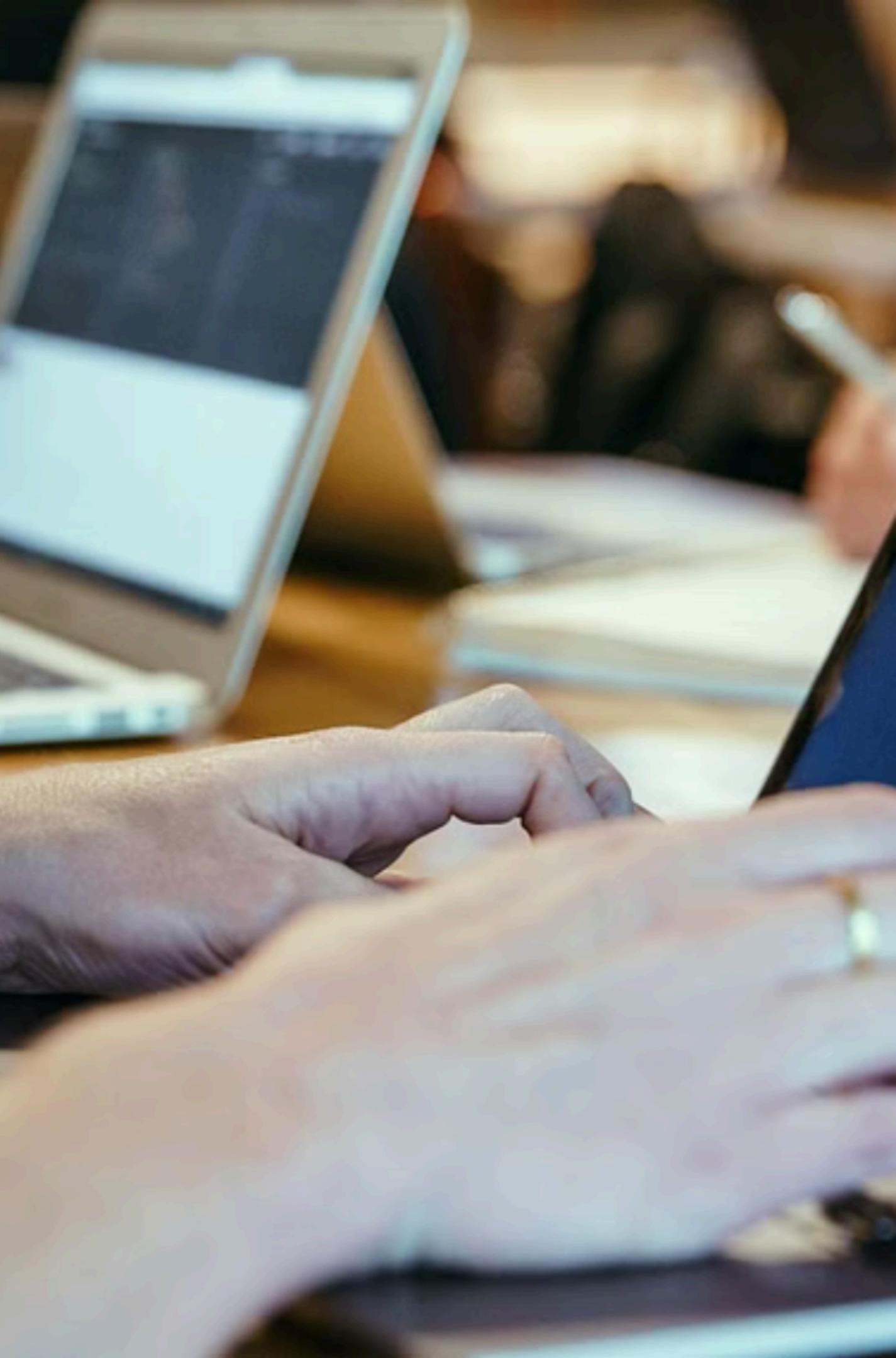
Crie um endpoint que retorna um JSON com as informações necessárias.

Demonstração de código: Rota GET

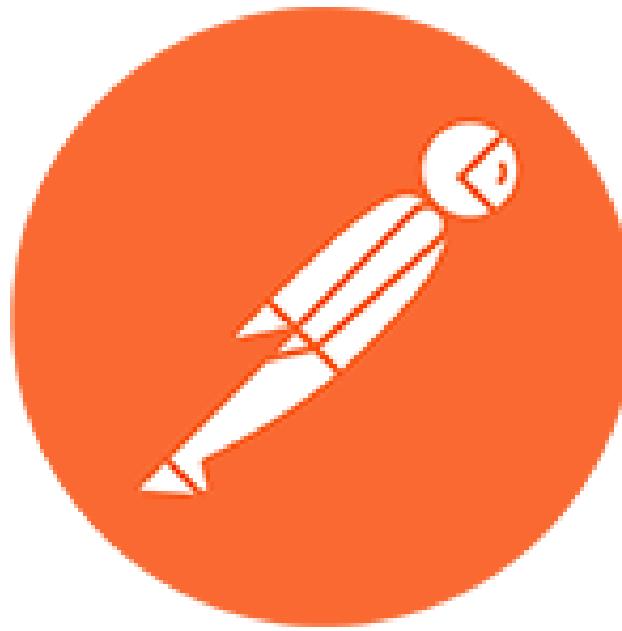
```
from fastapi import FastAPI

app = FastAPI()

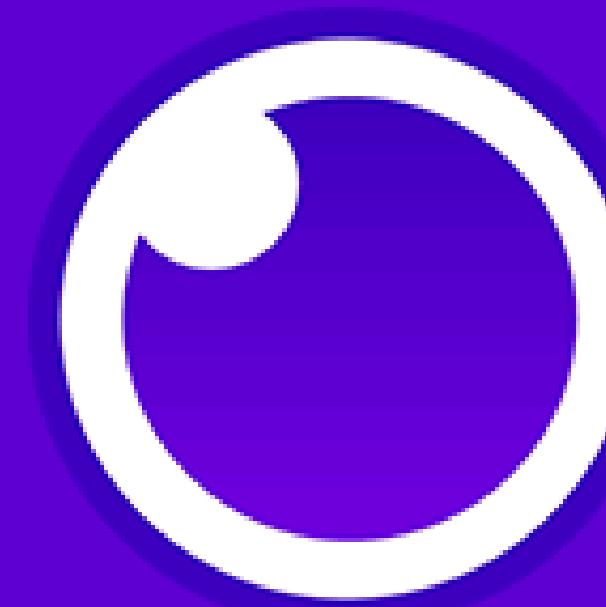
@app.get("/")
async def root():
    return {"message": "Bem-vindo ao FastAPI!"}
```



Ferramentas de Teste



POSTMAN



INSOMNIA

Criação de Rotas POST: Criar dados

1 Criar uma rota POST

Defina a função `@app.post("/")` para lidar com solicitações POST, que criam novos dados.

2 Criar um esquema de dados Pydantic

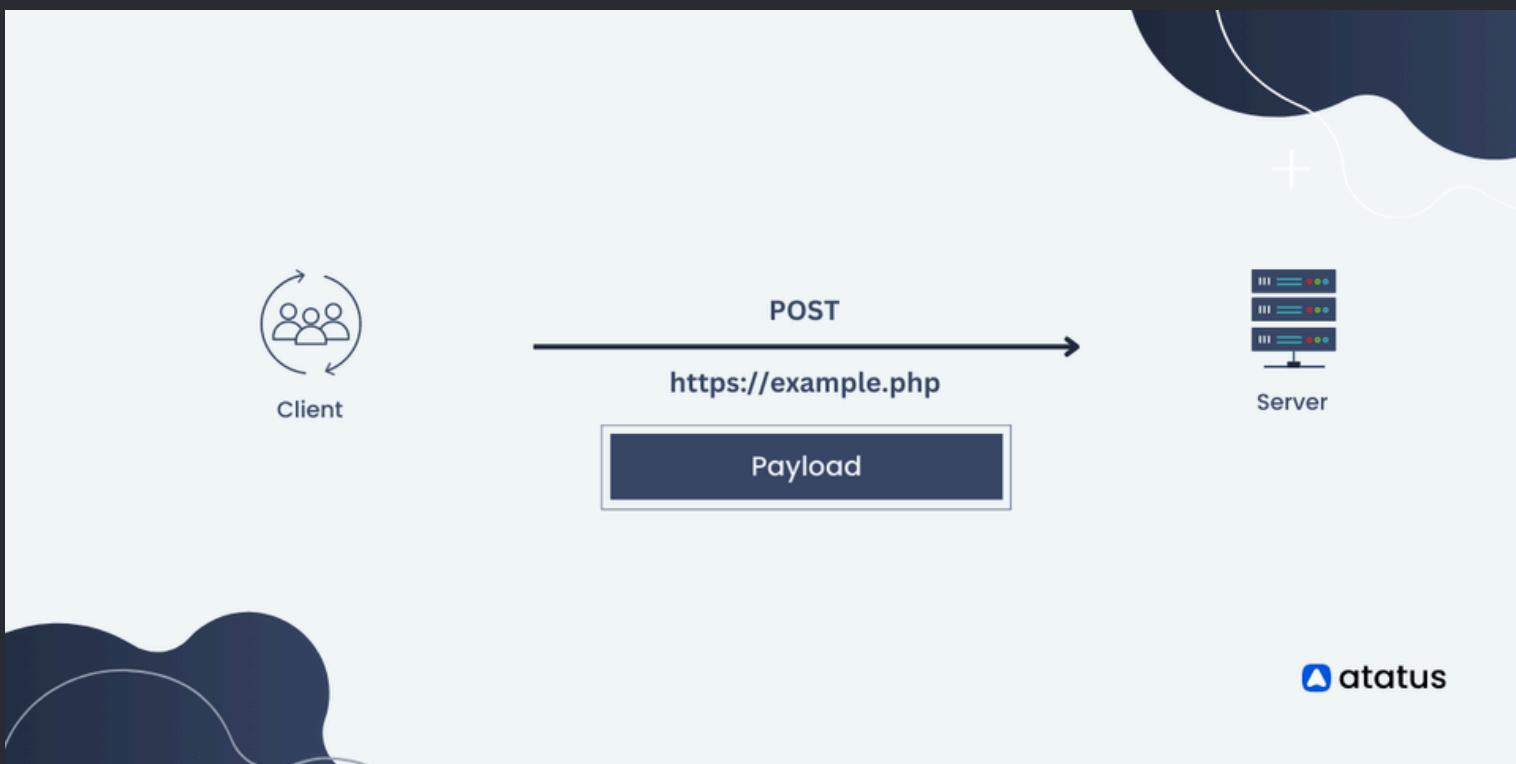
Utilize Pydantic para definir o formato dos dados que serão recebidos pela rota POST.

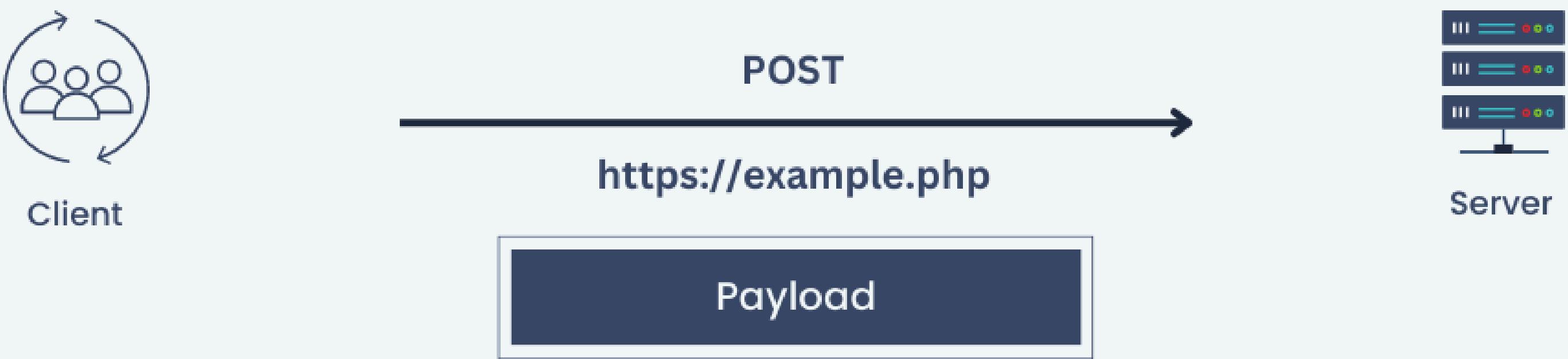
3 Processar os dados

Receba os dados da solicitação POST, processe-os e armazene as informações.

4 Retornar resposta

Retorne uma resposta, como uma mensagem de sucesso ou um código HTTP indicando o resultado.





Demonstração de código: Rota POST

```
from fastapi import FastAPI, Body  
from pydantic import BaseModel  
  
app = FastAPI()  
  
  
@app.post("/")  
def create_item(item: Item = Body(embed=True)):  
    return {"message": f"Item {item.name} criado com sucesso!"}
```

Atividade 1

Objetivo: Ensinar os alunos a criar um endpoint para trabalhar com comunicação de dados;

Descrição:

- Implemente um endpoint /produtos que retorne uma lista de produtos fictícios no formato JSON.
- Implemente um endpoint /criar-produto que permita enviar um produto (com id, nome, preço, quantidade) e retorne uma mensagem confirmando a criação



Criação de Rotas PUT: Atualizar dados

Rota PUT

Atualiza um recurso existente com novos dados.

Identificação do Recurso

Utilize um parâmetro na URL (ex: /items/{item_id}) para identificar o recurso a ser atualizado.

Dados de Atualização

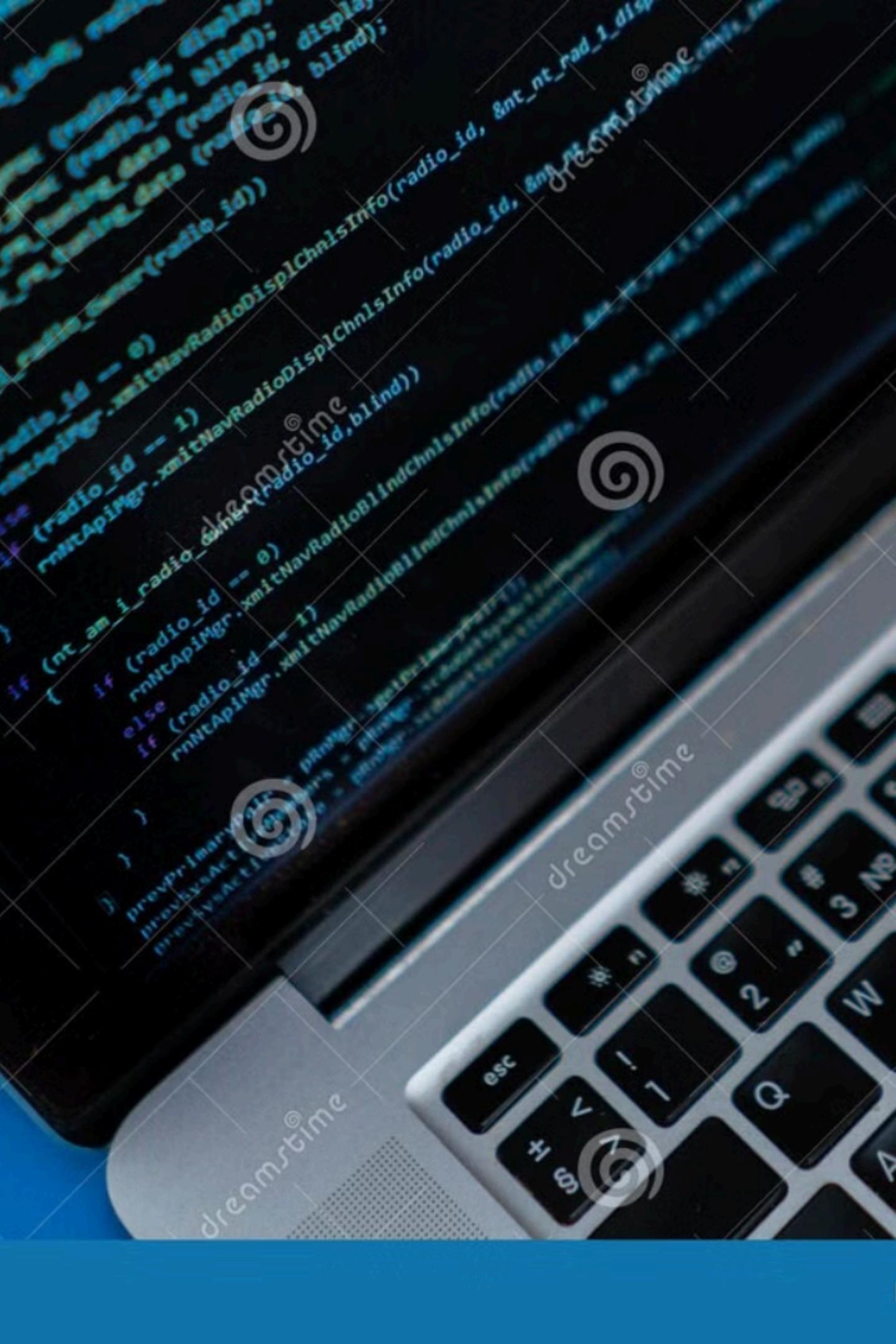
Receba novos dados através do corpo da requisição PUT, utilizando Pydantic para validar o formato.

Atualizar o Recurso

Efetue a atualização do recurso com os novos dados recebidos.

Resposta

Retorne uma resposta indicando o sucesso da operação, como uma mensagem de confirmação ou um código HTTP de sucesso.



Demonstração de código: Rota PUT

```
from fastapi import FastAPI, Body
from pydantic import BaseModel

app = FastAPI()

@app.put("/items/{item_id}")
def update_item(item_id: int, item: Item =
Body(embed=True)):
    return {"message": f"Item {item_id} atualizado com
sucesso!"}
```

Criação de Rotas DELETE: Excluir dados

Criar uma rota DELETE

- 1 Use a função `@app.delete("/")` para criar uma rota DELETE, responsável por remover dados.

Identificar o recurso

- 2 Utilize um parâmetro na URL (ex: `/items/{item_id}`) para indicar o item a ser excluído.

Excluir o recurso

- 3 Efetue a exclusão do item com base na identificação fornecida.

Retornar resposta

- 4 Retorne uma mensagem de sucesso, indicando que o item foi excluído com sucesso.



Demonstração de código: Rota DELETE

```
from fastapi import FastAPI

app = FastAPI()

@app.delete("/items/{item_id}")
def delete_item(item_id: int):
    return {"message": f"Item {item_id} excluído com sucesso!"}
```

Atividade 2

Objetivo: Ensinar os alunos a criar um endpoint para trabalhar com comunicação de dados;

Descrição:

- Implemente um endpoint /update-product que retorne uma lista de produtos fictícios no formato JSON.
- Implemente um endpoint /delete-product deletar um produto através do id.

Erros e Exceptions

O FastAPI fornece uma maneira robusta e flexível de lidar com erros e exceções. Quando algo dá errado em um endpoint, você pode retornar respostas adequadas usando o `status_code` apropriado e mensagens de erro claras.

- 404 Not Found: Quando um recurso não é encontrado.
- 422 Unprocessable Entity: Quando a entrada de dados não é válida (por exemplo, violação de validação de tipos de dados).

Erros e Exceptions

```
from fastapi import HTTPException

@app.get("/items/{item_id}")
def read_item(item_id: int):
    if item_id not in items:
        raise HTTPException(status_code=404, detail="Item not found")
    return {"item": items[item_id]}
```

Atividade 3

Criar um Usuário (Create) POST

Endpoint: /users

Listar Todos os Usuários (Read All) GET

Endpoint: /users

Obter um Usuário por ID (Read by ID) GET

Endpoint: /users/{user_id}

Atualizar um Usuário (Update) PUT

Endpoint: /users/{user_id}

Deletar um Usuário (Delete) DELETE

Endpoint: /users/{user_id}

Atividade 4

Criar um Item (Create) POST

Endpoint: /items

Listar Todos os Itens (Read All) GET

Endpoint: /items

Obter um Item por ID (Read by ID) GET

Endpoint: /items/{item_id}

Atualizar um Item (Update) PUT

Endpoint: /items/{item_id}

Deletar um Item (Delete) DELETE

Método: DELETE

Endpoint: /items/{item_id}