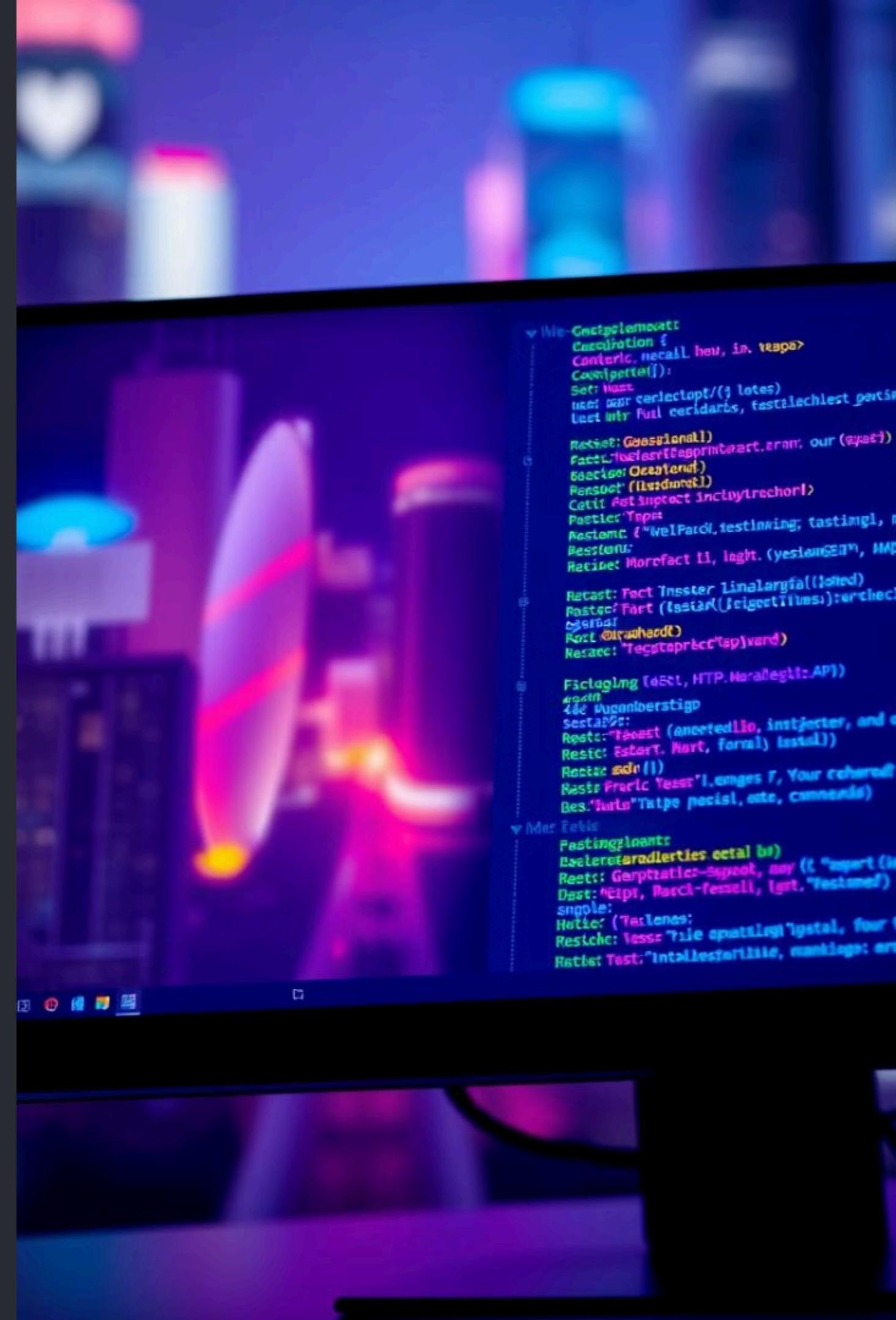
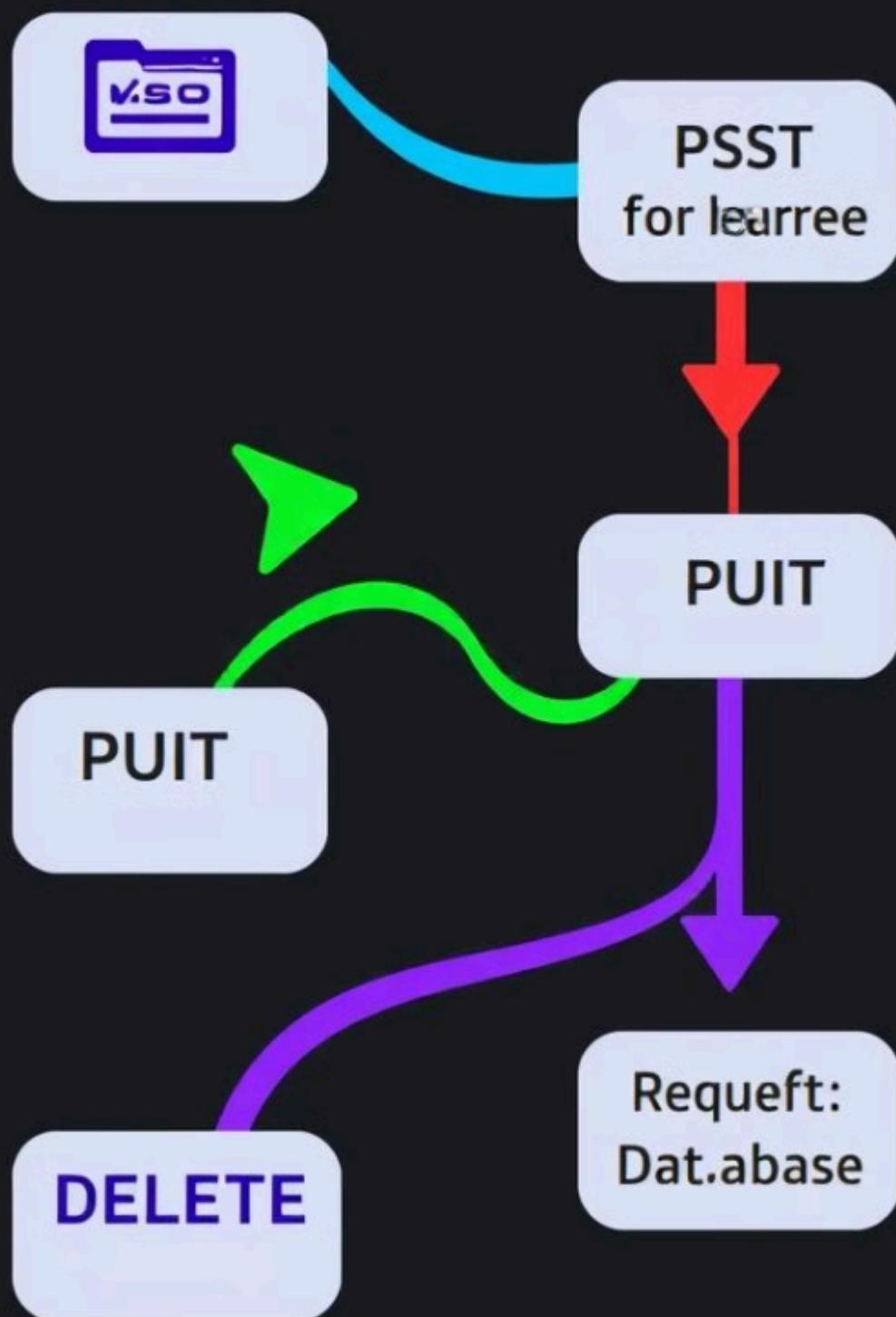


# Introdução ao Pydantic no FastAPI

Bem-vindos! Nesta apresentação, vamos explorar o poder do Pydantic para desenvolver APIs robustas com o FastAPI. Mergulharemos na criação de schemas, validação de dados, tratamento de erros e a integração com métodos HTTP.



# Métodos de Requisição HTTP (GET, POST, PUT, DELETE)



- 1 **GET**  
Obtém informações de um recurso.
- 2 **POST**  
Cria um novo recurso.
- 3 **PUT**  
Atualiza um recurso existente.
- 4 **DELETE**  
Deleta um recurso.

# O que é Pydantic?

## Validação de Dados

Pydantic é uma biblioteca Python para validação e serialização de dados. Ele define schemas que especificam o tipo de dados, formato e restrições, garantindo a integridade dos dados.

## Tipo Seguro

Pydantic fornece tipos de dados robustos e seguros, como str, int, float, list, dict, etc., que garantem que os dados atendam aos requisitos definidos.

## Documentação Automática

Pydantic gera automaticamente documentação para seus schemas, tornando o desenvolvimento de APIs mais rápido e fácil.

## Erros Descritivos

Em caso de erros de validação, Pydantic fornece mensagens de erro detalhadas, tornando a depuração mais eficiente.

# Benefícios de usar Pydantic no FastAPI

## Validação de Entrada

Pydantic garante que os dados de entrada das solicitações HTTP estejam corretos e no formato esperado.

## Tratamento de Erros

Pydantic facilita o tratamento de erros de validação, fornecendo mensagens de erro claras e concisas.

## Documentação Automática

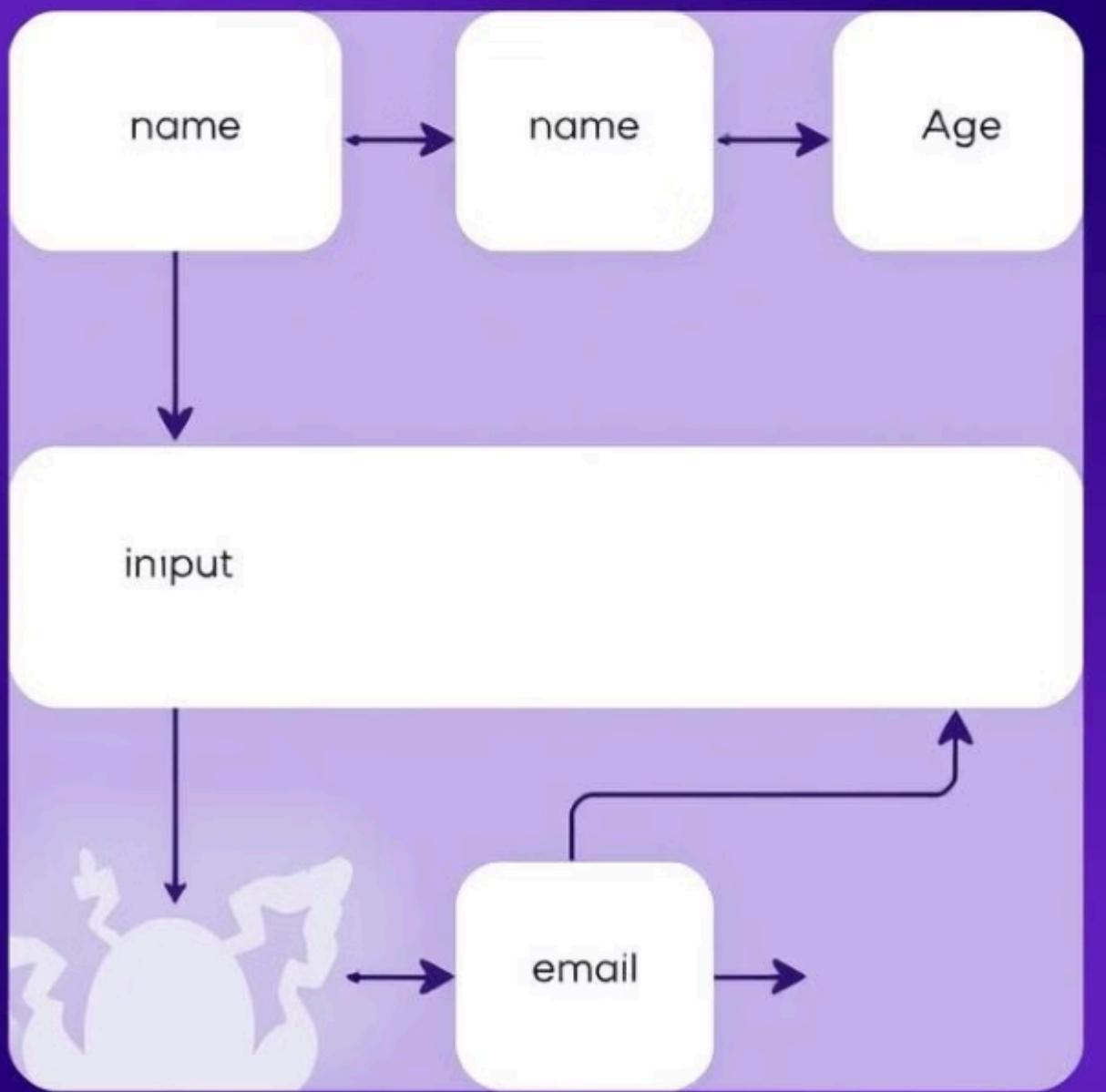
Ele gera documentação detalhada para suas APIs, incluindo informações sobre schemas de dados e validações.

## Integração Fluída

Pydantic integra-se perfeitamente com o FastAPI, proporcionando um desenvolvimento rápido e eficiente de APIs.



Youy durry the trigh identin gecans and beloment fnadse besyone. prect cller mace farge encht you tiper schapes.



# Criação de Schemas Pydantic

## Exemplo de Schema

### Definindo o Schema

Criamos schemas Pydantic usando classes Python. Cada atributo representa um campo de dados, com o tipo de dado especificado.

```
from pydantic import  
BaseModel  
  
class User(BaseModel):  
    id: int  
    name: str  
    signup_ts: datetime =  
        None  
    friends: List[int] =  
        []
```

### Atributos Adicionais

Podemos adicionar atributos como `required=True`, `default=None` e `alias='nome'` para personalizar o schema.

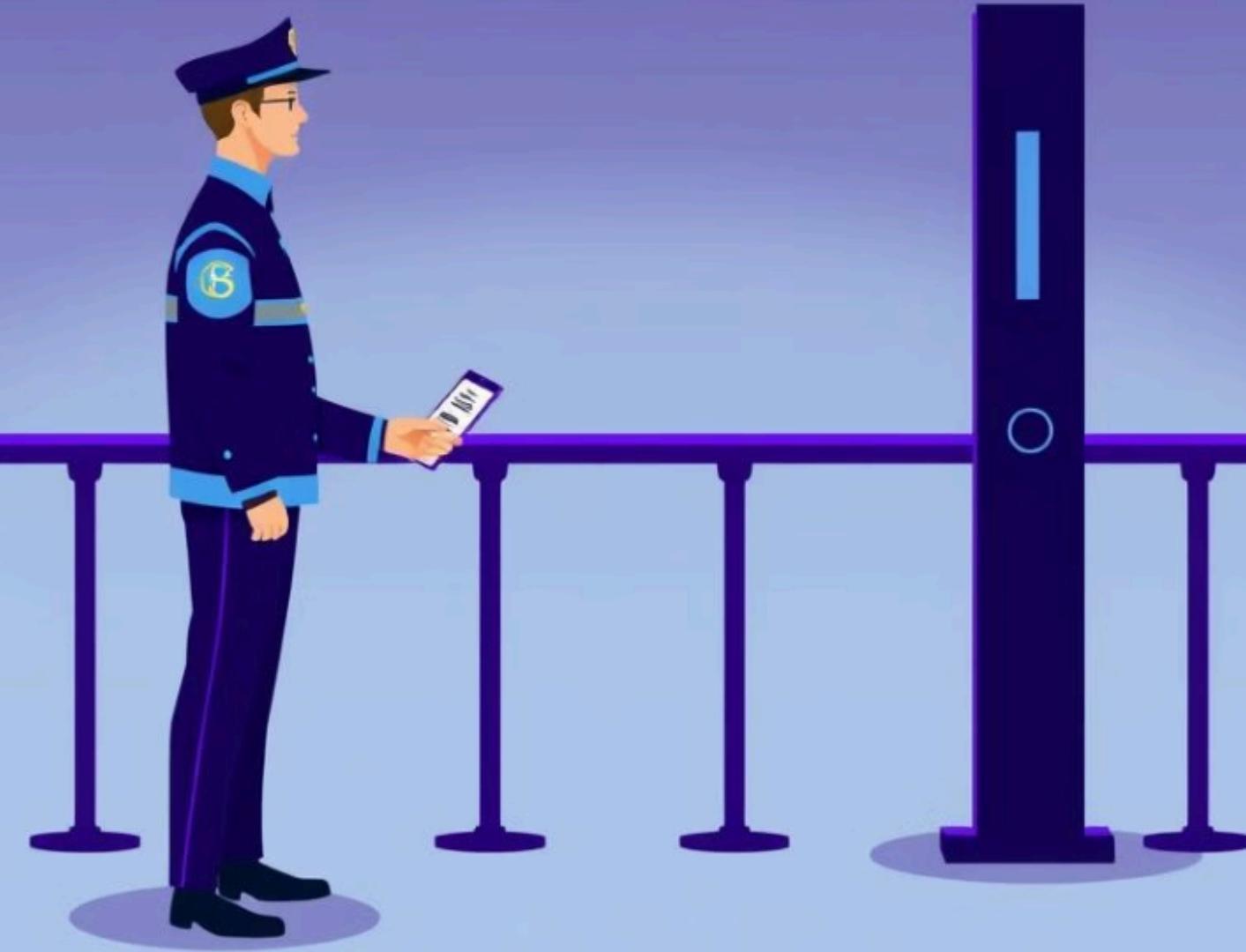
# Validações Customizadas com Pydantic Fields

Campo	Descrição
<code>min_length</code>	Comprimento mínimo da string.
<code>max_length</code>	Comprimento máximo da string.
<code>regex</code>	Padrão regex que a string deve seguir.
<code>ge</code>	Valor maior ou igual.
<code>le</code>	Valor menor ou igual.
<code>gt</code>	Valor maior que.
<code>lt</code>	Valor menor que.
<code>multiple_of</code>	Número deve ser múltiplo de.

# Custom validations

## custom vallidations

- indarararies toatid to ta lhe trecors  
restitution, she .atting...



# Validações Customizadas com Pydantic



## Validação de Dados

Pydantic permite criar validações personalizadas usando funções Python.



## Exemplo de Validação

```
from pydantic import validator

class User(BaseModel):
    email: str
```



## Tratamento de Erros

As validações personalizadas podem lançar exceções Pydantic para tratamento de erros centralizado.

# Utilizando Schemas Pydantic em Rotas HTTP

1

## Definição de Rotas

Utilizamos schemas Pydantic como tipos de dados para parâmetros de rota, corpo de requisição e resposta.

2

## Exemplo de Rota

```
from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()

class User(BaseModel):
    id: int
    name: str

@app.post('/users')
async def create_user(user: User):
    return {'message': f'Usuário {user.name} criado com sucesso!'}
```

3

## Validação Automática

O FastAPI utiliza Pydantic para validar os dados automaticamente, garantindo que os dados sejam consistentes.

# Respostas HTTP com Schemas Pydantic

## Estrutura de Dados

Schemas Pydantic também podem ser utilizados para definir a estrutura da resposta HTTP.

```
from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()

class User(BaseModel):
    id: int
    name: str

@app.get('/users/{user_id}')
def get_user(user_id: int):
    user = User(id=user_id,
name='John Doe')
    return user
```

# Atividade 1

## Requisitos

Objetivo: criar um sistema para gerenciamento de tarefas contendo todas as operações do CRUD;

Validação com Pydantic:

id: Inteiro positivo (gerado automaticamente).

title: String com no mínimo 5 e no máximo 100 caracteres.

description: String opcional com no máximo 300 caracteres.

status: Enum que pode ser "pending", "in\_progress" ou "completed".

priority: Inteiro entre 1 e 5.

# Atividade 2

## Descrição da Atividade

Objetivo: fornecer backend para aplicação de sistema climático.

### 1. Adicionar dados climáticos de uma cidade:

- Receber informações como nome da cidade, temperatura, umidade e data da coleta.
- Validar que a temperatura e umidade estejam dentro de limites realistas.
- Verificar se o nome da cidade já existe na base simulada.

### 2. Listar todos os registros climáticos:

- Retornar todos os dados climáticos armazenados.

### 3. Obter o relatório do dia mais quente:

- Identificar e retornar a cidade com a maior temperatura registrada.

### 4. Calcular a média de temperaturas:

- Retornar a média de temperaturas de todas as cidades registradas.

# Tratamento de Erros com Pydantic

1

## Validação de Entrada

Pydantic lança exceções Pydantic em caso de falha na validação de dados de entrada.

---

2

## Tratamento de Erros

O FastAPI intercepta essas exceções e gera respostas HTTP com informações de erro.

---

3

## Mensagens de Erro

Pydantic fornece mensagens de erro detalhadas, auxiliando na depuração.

# Conclusão e Próximos Passos

1

## Conclusão

O Pydantic é uma ferramenta poderosa para construir APIs robustas e seguras com o FastAPI.

2

## Próximos Passos

Explore a documentação do Pydantic para descobrir mais funcionalidades avançadas.

3

## Pratique

Crie suas próprias APIs usando Pydantic e FastAPI para consolidar o aprendizado.

