04

O que é uma API

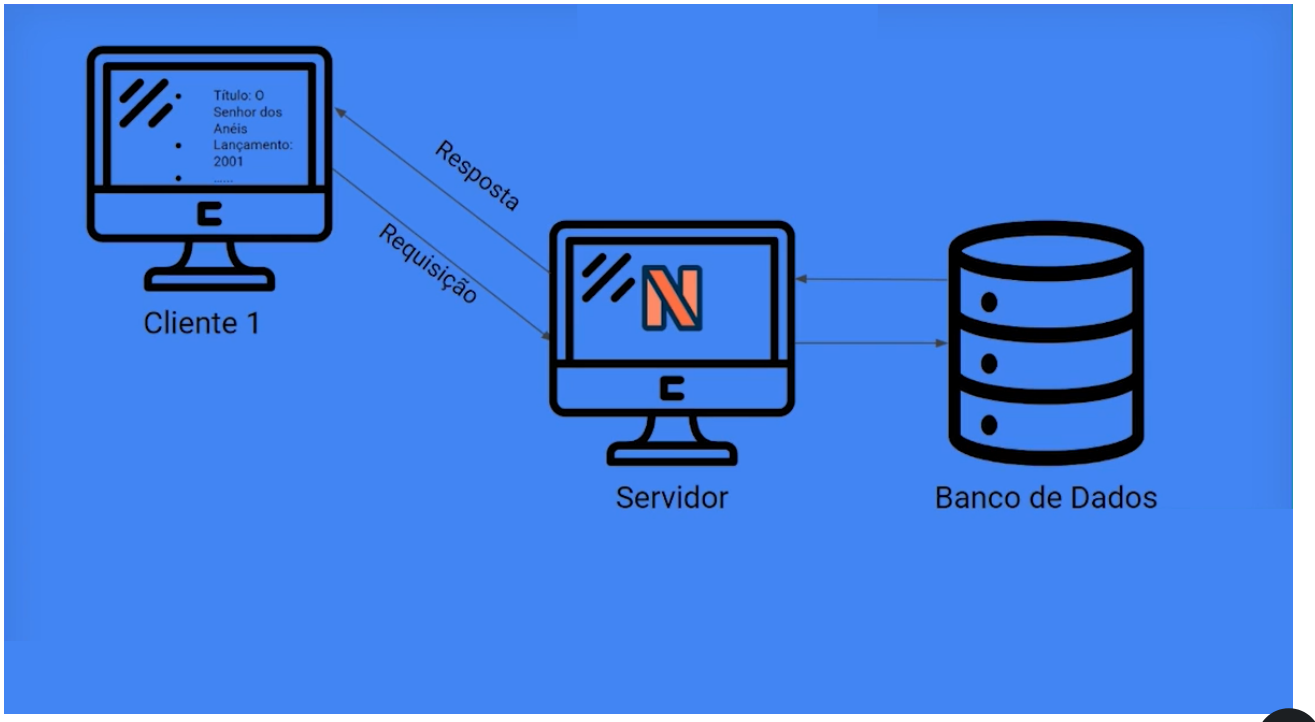
Transcrição

Antes de começar a desenvolver nosso projeto, vamos entender **o que são APIs e para que servem**.

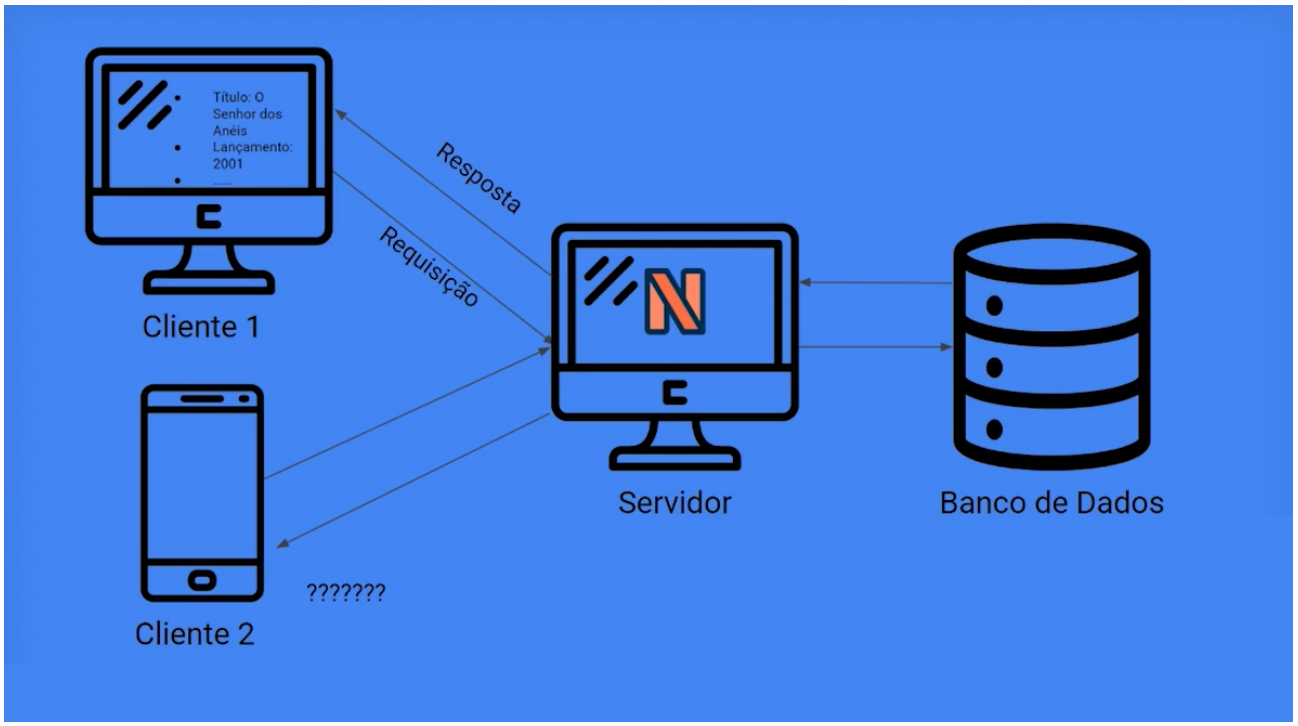
Primeiramente, "API" é a sigla para o termo "Application Programming Interface" — em português, **interface de programação de aplicações**. Mas o que isso significa?

Para entender o conceito de API, pensaremos num exemplo relacionado a filmes, já que nosso projeto terá esse tema. Vamos imaginar que o cliente 1 deseja conferir informações sobre o filme "O senhor dos anéis" pelo computador. Por exemplo, o título, o ano de lançamento, o nome do diretor, o gênero e o tempo de duração.

A máquina do cliente fará uma **requisição** para o servidor. O servidor, por sua vez, consultará os dados cadastrados no banco de dados e os enviará para o servidor, que mandará a **resposta** de volta para o cliente.



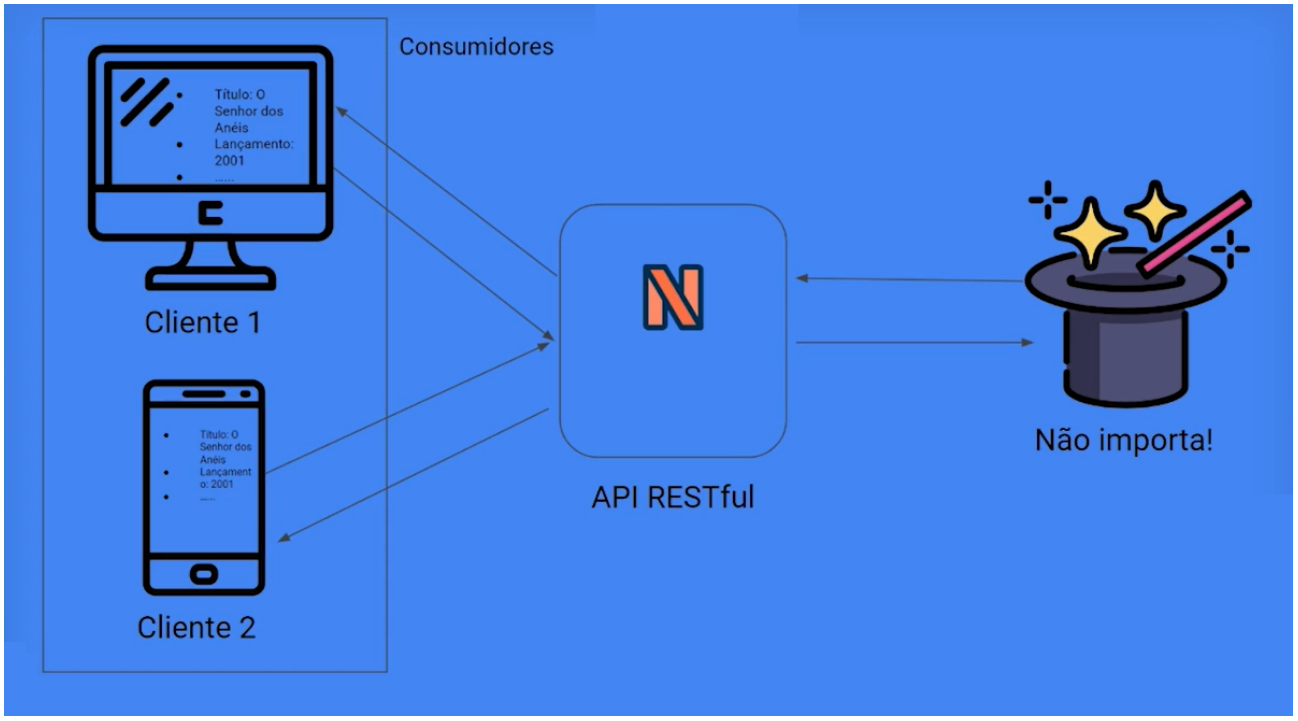
O cliente 2 está utilizando um aplicativo no smartphone ou o próprio navegador do celular e deseja obter as mesmas informações para outros fins. Será que a resposta enviada ao cliente 1 pode ser o mesmo tipo de resposta dada ao cliente 2?



Cada cliente pode estar esperando dados em determinado escopo. O servidor precisa sempre estar atento a que tipo de resposta deve ser enviado. Como podemos solucionar essa questão?

Em vez de um servidor, os clientes podem enviar requisições para uma API. Ela será responsável por resolver esse pedido e dar uma resposta que os clientes consigam utilizar os dados designados.

Mas como a API consulta os dados e os devolve para o usuário? No final das contas, isso não importa! O processo é análogo ao conceito de orientação a objetos: basta que sigamos as regras impostas pela interface para receber a resposta padronizada, que pode ser consumida pelos nossos clientes.



Em outras palavras, podemos imaginar a API como uma "casca" que contém um conjunto de regras. Cada cliente que quiser consumir da API, conseguirá interagir com esse sistema, desde que siga essas regras.

Quando implementamos determinado conjunto de regras arquiteturais, um dos mais difundidos é o **REST — Representational State Tranfer**. Seguindo esse padrão, sempre saberemos o que vamos receber e o que precisamos enviar nas duas pontas, de modo que a comunicação fica padronizada.

Ao segue o padrão REST, uma API é chamada de API RESTful. Ou seja, REST é o nome do conceito arquitetural e RESTful é quem implementa esse conceito.

Recapitulando: os clientes (consumidores) fazem requisições para a API RESTful e não importa como funciona a lógica por trás da API não importa. Pode ser um banco de dados (relacional ou não) ou estar em memória. Desde que os consumidores sigam as regras impostas pela API, eles receberam os dados.

Assim, a API visa disponibilizar informações para outras aplicações, seja para operações de escrita, leitura, atualização ou remoção.

Para consumir seus recursos, é necessário seguir as regras estabelecidas (que entenderemos ao longo do curso). Como as APIs abstraem detalhes de implementação, não precisamos compreender como o *back-end* está sendo implementado.

Além disso, APIs controlam o que pode ou de ve ser acessado. Se criamos um *endpoint* para devolver determinada informação para o usuário, conseguimos ter um ótimo controle do que será exposto ou não a quem consome a API.