O que é API REST?

O acrônimo API é a abreviação de *Application Programming Interface,* que significa "Interface de Programação de Aplicações". Representa um conjunto de rotinas e padrões estabelecidos e documentados para que uma determinada aplicação de software tenha autorização para utilizar as funcionalidades oferecidas por essa aplicação, sem precisar conhecer as anuências dessa implementação. Isso garante segurança de código e, principalmente, das regras de negócio do software e a interoperabilidade entre aplicações, em que essa comunicação ocorra utilizando as requisições HTTP responsáveis pelas operações de manipulação de dados.

API REST é uma abstração de arquitetura de software que fornece dados em um formato padronizado para modelos de requisições HTTP.

A arquitetura REST possui um conjunto de regras e princípios que devem ser seguidos:

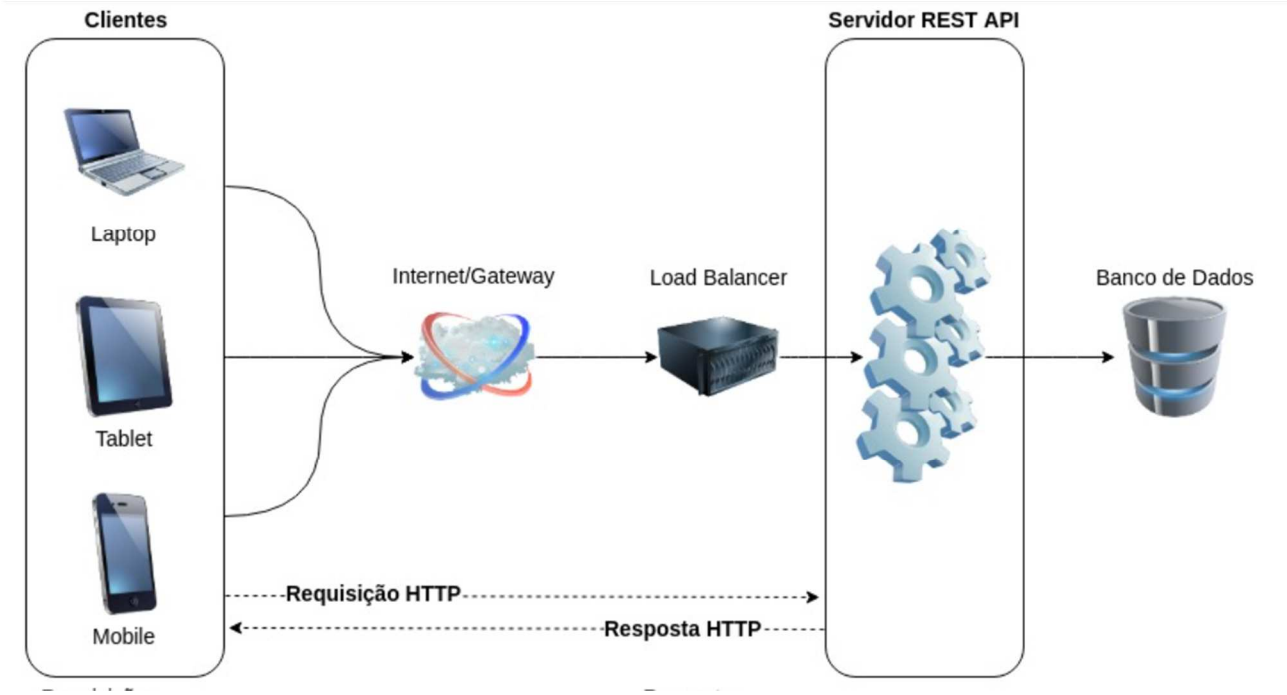
Cliente-Servidor: Trata a respeito da separação de responsabilidades, ou seja, separar as preocupações de interface do usuário (*User Interface*) do banco de dados, abstraindo a dependência entre os lados clientes/servidor e permitindo a evolução desses componentes sem impacto e quebra de contrato.

Interface Uniforme: É a interoperabilidade entre os componentes cliente e servidor. Como o cliente e servidor compartilham da mesma interface, é necessário estabelecer um contrato para a comunicação entre essas partes. Para isso, há quatro princípios a serem seguidos:
1) Identificação dos recursos (por exemplo,*Swagger*);
2) Representação dos recursos;
3) Mensagens auto-descritivas;
4) Componente HATEOAS.

Stateless: Cada requisição acionada entre a comunicação cliente-servidor deve possuir toda a informação necessária e compreensível para realizar a origem da requisição, não sendo de responsabilidade do servidor armazenar qualquer tipo de contexto. Isso pode gerar alto tráfego de dados e impacto na performance da aplicação, porém pode-se utilizar recursos de cache nesses casos.

Cache: É utilizado para melhorar a performance de comunicação entre aplicações, otimizando o tempo de resposta na comunicação entre cliente-servidor. É de responsabilidade do servidor controlar as diretivas de cache através do cabeçalho HTTP (*HTTP Header*).

Camadas: A separação de responsabilidades é importante nesse modelo de arquitetura. Os princípios e as boas práticas na arquitetura e design de um projeto, sugerem a construção de camadas independentes e auto gerenciadas, em que cada camada não pode conhecer as demais camadas. Caso ocorra mudanças em uma delas, as demais não serão impactadas. Nesse modelo, o cliente não deve conectar-se diretamente ao servidor da aplicação, porém uma camada de balanceamento de carga deverá ser acionada para essa responsabilidade.



É utilizada para estruturar qualquer modelo de aplicações web para os dias atuais, onde temos um alto volume de trocas de dados sendo processados de forma assíncrona entre diversas aplicações. Conheça algumas delas:

- Redes sociais
- Sites de E-Commerce
- Internet das coisas
- Inteligência Artificial
- Aplicações executadas pelo próprio navegador (Web App)
- Aplicações Mobile
- Serviços de troca de mensagens (WhatsApp, Slack)
- Repositório de dados (Google Drive, GitHub, Azure Blob Storage)

Quais os tipos de API REST?

Utilizar API REST significa utilizar uma API para o consumo de dados em aplicações back-end, de modo que essa comunicação seja estabelecida utilizando padrões definidos no estilo arquitetural REST. Os três tipos de APIs são:

Privadas ou Locais

Modelo restritivo, utilizado entre aplicações internas de uma empresa, ou seja, local.

Geralmente são aplicações que disponibilizam acessos a dados mais críticos e sigilosos da empresa, em que o acesso a aplicações externas é bloqueado utilizando Proxy e sistema de autenticação de alto nível.

Parceiras ou baseadas em programa

Modelo restritivo, utilizado entre parceiros de negócios ou para a integração entre diferentes softwares de uma mesma empresa ou entre terceiros.

Nesse modelo, a autenticação de acesso aos dados é conhecida apenas entre as partes estabelecidas.

Públicas ou baseadas em web

Modelo que pode ser acessado praticamente por qualquer aplicação de software, mediante um pedido de requisição do tipo *HTTP GET*.

Uma das suas principais utilidades é na realização de testes de acesso ou validação de uma aplicação durante o seu desenvolvimento.

Conheça alguns modelos públicos de API REST:

- Nasa
- Marvel
- Football
- Clima Tempo
- Youtube
- MovieDB

Quais as vantagens e desvantagens de usar API REST?

API REST apresenta vantagens competitivas, porém também há desvantagens que precisam ser consideradas. São elas:

Vantagens

· Separação entre cliente-servidor

· Praticidade no acesso aos contratos da aplicação

· Confiabilidade e segurança na aplicação

· Escalabilidade para aplicações, principalmente nos modelos de microsserviços

· Multiplataforma, considerando que os dados podem retornar nos padrões XML e JSON

Desvantagens

· Conhecimento dos padrões de projeto para o modelo REST API

· Implementação “Lo-rest”, ou seja, desenvolver aplicações que disponibilizam apenas dois endpoints: HTTP GET e HTTP POST

· Trabalham de forma assíncrona, aumentando a sobrecarga das solicitações, podendo comprometer o desempenho da comunicação entre as aplicações

Quais as diferenças entre API REST e API RESTFUL?

API REST e API RESTFUL possuem objetivos distintos, mas complementares. A diferença encontra-se somente no cumprimento das exigências da arquitetura de software.

O estilo de arquitetura de uma API REST deve possuir alguns princípios:

· Não possui criptografia

· Não possui sessão

· Utiliza apenas o protocolo HTTP

· Os recursos devem ser acessados somente utilizando o protocolo de internet URI (Uniform Resource Identifier)

· O retorno dos dados devem ser nas formas mais conhecidas, exemplos: JSON e XML

API RESTFUL é o cumprimento de todos os princípios citados anteriormente, ou seja, um serviço baseado em REST é chamado de serviços RESTFUL, com o objetivo de oferecer uma excelente interatividade e rapidez na comunicação entre os serviços.