***REST API***

**. O que é REST API:** É uma aplicação que permite a integração entre diferentes sistemas padronizando o fluxo de requisições, a forma de envio e recebimento de informações e o protocolo de codificação das mensagens.

O REST é um conjunto de convenções arquiteturais que utiliza algumas camadas do HTTP para envio e recebimento de informações. Logo uma REST API é uma API que permite a integração entre sistemas utilizando as convenções arquiteturais do REST.

Exemplos que usam: Youtube, Instagram, AnyAPI (é uma coleção de APIs)

**. O que é RESTful:** É um sistema que segue todas as boas práticas do estilo de arquitetura REST. Isso envolve seguir padrões para definição de recursos, status de resposta, cache, codificação dos dados, forma de envio e de recebimento, entre outras coisas.

Algumas APIs são construídas seguindo padrões arquiteturais antigos como o RPC, aplicando alguns padrões REST. Normalmente são aplicações confusas, difíceis de dar manutenção, evoluir e principalmente integrar. O RESTful traz um foco no consumidor, que no caso, é o desenvolvedor.

**. Endpoint:** Os principais dados necessários para um sistema se comunicar com uma API são:

-Protocolo: HTTP ou HTTPS

-Host: api.sistemanoticias.com.br (url)

-Path: /categorias/1/noticias

O Endpoint é a junção de todas essas informações, ou seja, ele é o endereço específico de determinado recurso na API. Cada recurso tem o seu endpoint.

**. Estrutura de endpoint:** protocolo://domínio/nome-da-api/versão/recurso

Exemplo: https://api.sistemanoticias.com.br/news-api/v1/categorias/1/noticias/12

**. Verbos REST:**

- GET: Envio de parâmetros via URL, recebimento de informações no corpo da resposta

- POST: Envio de parâmetros via corpo, recebimento de informações no corpo da resposta

- PUT: Envio de parâmetros via URL e via corpo, recebimento de informações no corpo da resposta

- DELETE: Envio de parâmetros via URL, recebimento de informações no corpo da resposta

- PATCH: Envio de parâmetros via URL e corpo, recebimento de informações no corpo da resposta

- OPTIONS: Envio de parâmetros via URL, recebimento de informações no cabeçalho da resposta

- HEAD: Envio de parâmetros via URL, recebimento de informações no cabeçalho da resposta

**. Utilidades dos verbos no RESTful:**

- GET: Método de consulta de registos, utilizado para buscar informações

- POST: Método para criação de registro, utilizado para enviar informações

- PUT: Método para atualização completa de registro, utilizado para enviar informações

- DELETE: Método para remoção de registro, utilizado para enviar informações

- PATCH: Método para atualização parcial de registro, utilizado para enviar informações

- OPTIONS: Método para verificar quais métodos são permitidos, utilizado para buscar informações

- HEAD: Método para consultar apenas o cabeçalho dos registros, utilizado para buscar informações

**. Versionamento:** Ao invés de “quebrar o cliente” alterando a estrutura de serviços que já estão sendo utilizados em ambiente de produção, é criada uma nova versão da API com as novas alterações e sugerido aos clientes que façam a atualização de seu sistema. Até que todos os clientes tenham atualizado para a nova versão, a versão anterior é mantida em operação.

**. Status code:** Padrão de 3 dígitos que indica o resultado da tentativa de tratar a requisição. São divididos em 4 classes, identificadas pelo primeiro dígito (1xx Informational) (2xx Successful) (3xx Redirection) (4xx Client Error) (5xx Server Error). Clients não precisam entender todos os códigos, mas sim as classes e alguns mais importantes

**. Documentação de apoio:** restfulapi.net; swagger.io/resources/articles/best-practices-in-api-design/; developer.mozilla.org/en-US/docs/Web/HTTP

**. DX: Develop Experience** é uma nova perspectiva de mercado que foca na experiência dos desenvolvedores enquanto estão desenvolvendo um projeto. É equivalente ao User Experience (UX) para o usuário final. Isso significa se preocupar em fornecer informações suficientes e precisas para os desenvolvedores, além de tomar ações que facilitem a jornada do desenvolvedor desde o início até o fim de um projeto.

**. Documentação de API:** É uma definição do que é e de como se comportam os seus recursos. Nela estão contidas informações como protocolos, endpoints, parâmetros, formatos de dados, entre outras coisas.

Sua principal utilidade é servir como referência para os desenvolvedores que irão implementar integrações com a API e fornecer a eles todas as informações necessárias para esse trabalho. Muitas vezes é gerada antes da implementação da API de fato e serve com “contrato” entre o backend e o frontend para saberem o que esperar um do outro.

**. Principais informações de um documento API:** Protocolo (HTTP;HTTPS), Host, Recursos, Métodos, Formato dos parâmetros, Formato dos retornos, Status code, Autenticação

**. Swagger:** Trata-se de uma aplicação open source que auxilia desenvolvedores nos processos de definir, criar, documentar e consumir APIs REST.

Swagger é uma linguagem de descrição de interface para descrever APIs RESTful expressas usando JSON. O Swagger é usado junto com um conjunto de ferramentas de software de código aberto para projetar, construir, documentar e usar serviços da Web RESTful. Swagger inclui documentação automatizada, geração de código e geração de casos de teste.

**. Documentação de apoio:** swagger.io/docs