**Introdução a Programação Web**

**1.Palavras-Chave**

1. **Internet**
2. **Intranet**
3. **Websites**
4. **Web design**
5. **Programação web**
6. **Banco de dados**
7. **Engenharia de servidores**

**2.Funcionamento de um Website**

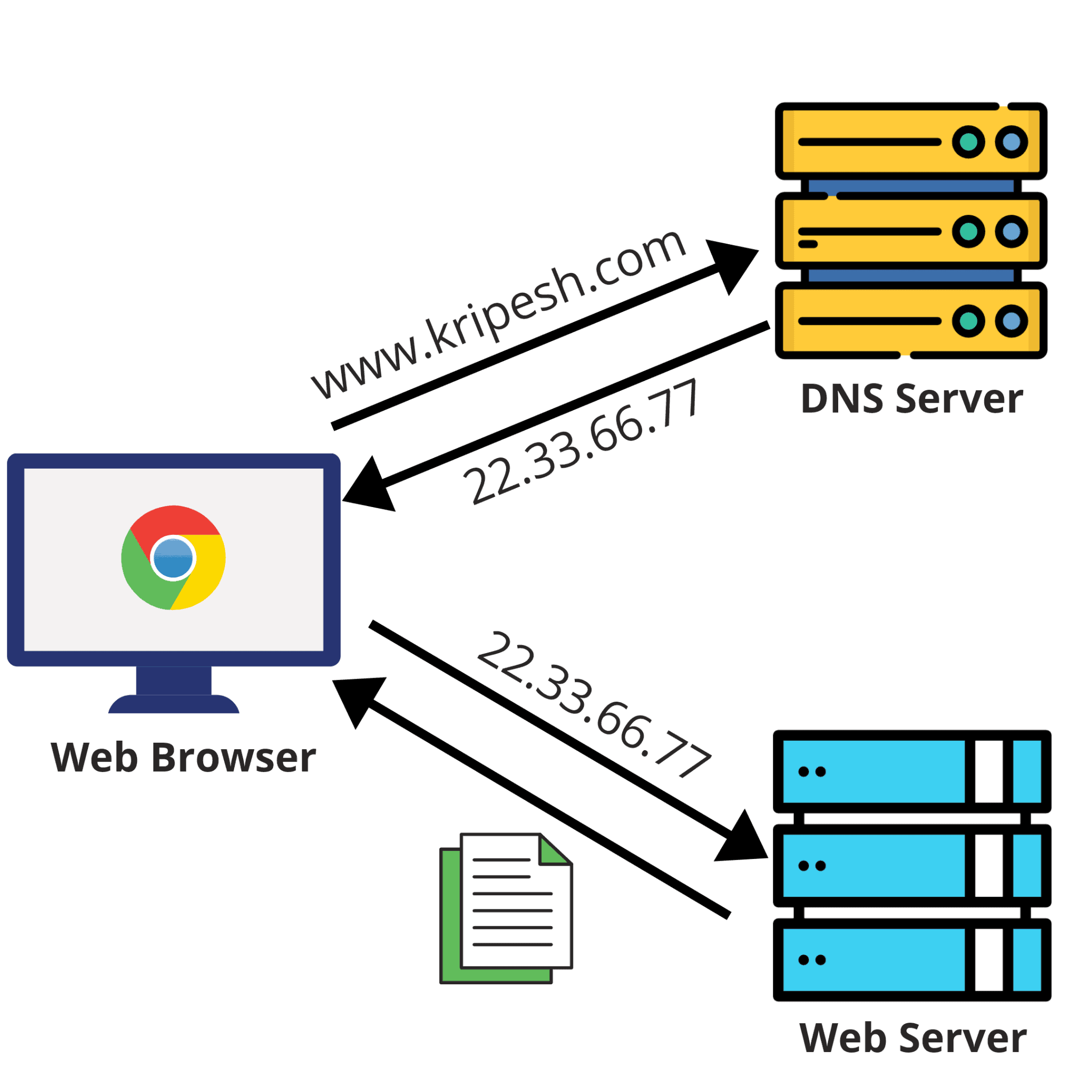
**1.Solicitação do usuário:** Tudo começa com o usuário preenchendo um **URL** (**significa localizador uniforme de recursos**e é um endereço da Web que aponta para um site, página ou documento) no navegador ou clicando em um link.

**2.Resolução de DNS:** O URL é traduzido em um endereço **IP** (**significa** protocolo de internet que define as regras de como as informações serão transmitidas entre dispositivos conectados em rede) através de um sistema chamado **DNS** (Domain Name System).

**3.Conexão com servidor:** O navegador utiliza o endereço IP para estabelecer conexão com um **servidor** (**computador ou sistema de dispositivos conectados em rede que recebe, processa e responde a solicitação de outros computadores denominados “clientes”, fornecendo serviços e recursos para os usuários da internet**) que hospeda o site.

**4.Resposta do servidor:** O servidor processa a solicitação **HTTP** (significa Hypertext Transfer Protocol é um protocolo de comunicação utilizado para a transferência de informações na **World Wide Web** e em outros sistemas de rede, sendo a base para que o cliente e um servidor web troquem informações) e envia de volta os arquivos (geralmente em HTML, CSS e JavaScript).

**5.Renderização do navegador:** O navegador interpreta estes arquivos e exibe o site ao usuário.



Demais tecnologias envolvidas:

**SSL/TLS (segurança):** são **protocolos de segurança** aplicados em sites para garantir que a navegação dos usuários esteja protegida contra vazamento de dados e ataques de hackers.

**API(s) (interatividade):** Interface de Programação de Aplicações

**Banco de Dados (armazenamento)**

**3.Tecnologias front-end e back-end**

**3.1. Front-end**

Desenvolvimento web que trata da interface do usuário.

Tecnologias principais:

HTML (Hypertext Markup Language): estrutura o conteúdo da web.

CSS (Cascading Style Sheets): estiliza e apresenta o conteúdo HTML.

JavaScript: torna as páginas web interativas e dinâmicas.

**3.2. Back-end**

Inclui servidor, aplicação e banco de dados. O usuário não vê e nem tem acesso. É responsável por gerenciar e processar os dados, garantindo que tudo no front-end ocorra corretamente.

Tecnologias principais:

Linguagens de programação: Python, Ruby, PHP, Java, JavaScript, etc.

Banco de dados: PostgreSQL, MySQL, MongoDB, Oracle, etc.

Frameworks: Django (Python), Express (JavaScript), Spring Boot (Java)

**3.3. Full Stack**

Trabalha tanto o front-end quanto o back-end.

**3.API(s) e Conceitos Fundamentais**

Funciona como um intermediário permitindo que requisições sejam realizadas e respondidas entre diferentes sistemas de softwares.

Exemplo de API (API de transação de comércio da Vindi): <https://vindi.github.io./api-docs/dist/#/>

**4.Tipos de API(s)**

**1.RESTful:** API(s) que seguem os princípios do REST (Representational State Tranfer). São baseadas em padrões HTTP e utilizas para interações web.

Características:

* Uso dos métodos HTTP: GET, POST, PUT, DELETE para operações CRUD;
* Curva de menor aprendizado;
* Fácil de entender e implementar.

Preferido pela simplicidade.

**2.Soap (Simple Object Acess Protocol):** é um protocolo que define um padrão de troca de mensagens baseadas em XML.

Exemplo: <https://www.w3schools.com/xml/xml_soap.asp>

Características:

* Baseado em XML para troca de informações;
* Independente de linguagem e plataforma de transporte;
* Suporte para operações complexas e segurança avançada.

Ideal quando o foco está na segurança e as operações são complexas.

**3.API GraphQL:** uma linguagem de consulta para a sua API e um servidor capaz de executar estas consultas, retornando apenas os dados especificados.

Exemplo: <https://studio.apollographql.com/public/SpaceX-pxxbxen/variant/current/explorer>

Características:

* Permite que os clientes especifiquem exatamente quais dados querem;
* Eficiente na redução de solicitações e no tamanho dos dados transferidos;
* Flexível e fortemente tipada, facilitando a evolução das API(s).

Usado para dados dinâmicos e personalizados.

**5.Verbos de HTTP**

**GET: leitura;**

**PUT/PATCH: atualização**

**POST: criação**

**DELETE: exclusão**