Universidade Cruzeiro do Sul

UNIVERSIDADE CRUZEIRO DO SUL
TECNICAS DE PROGRAMAÇÃO
PROF. MSC ENG VINICIUS HELTAI

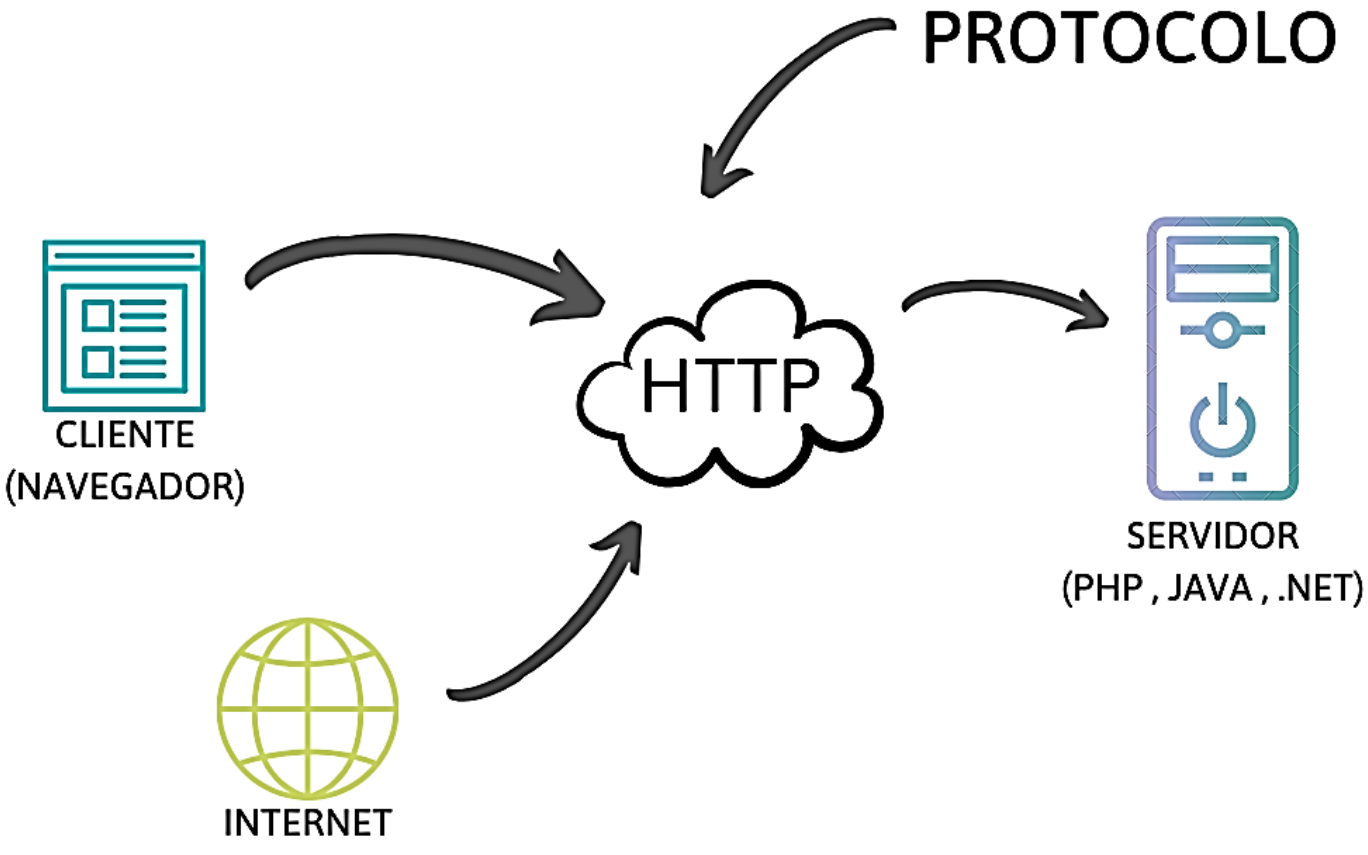
SERVIÇOS WEB
FUNDAMENTOS E PROTOCOLOS

FUNDAMENTOS WEB

PROTOCOLO – HTTP:

- HTTP é um protocolo de comunicação (conjunto de regras). Esta definida na documentação: https://datatracker.ietf.org/doc/html/rfc2616
- Entender o protocolo HTTP facilita para o desenvolvimento de ferramentas como PHP, Java, iOS, Dot.Net, Android, etc.
- HTTP tem sua principal utilização em navegadores de internet. Isso se deve pelo fato de ser um protocolo na pilha TCP/IP (comunicações Web).
- O HTTP é muito utilizado em arquitetura chamada cliente-servidor:

FUNDAMENTOS WEB



CRIAÇÃO E EVOLUÇÃO HISTORICA DO HTTP:

- O protocolo HTTP (Hypertext Transfer Protocol) foi criado em 1989 por Tim Berners-Lee, um cientista da computação britânico que trabalhava no CERN (Organização Europeia para a Pesquisa Nuclear) na Suíça.

- Berners-Lee estava desenvolvendo uma forma de compartilhar informações entre seus colegas de trabalho e achou que seria útil ter um sistema que permitisse o acesso a documentos em hipertexto (textos com links para outros textos).

- O HTTP foi originalmente projetado para ser usado em conjunto com o HTML (Hypertext Markup Language), uma linguagem de marcação que permite a criação de documentos hipertexto para a web.

- O HTTP é um protocolo cliente-servidor, ou seja, ele permite que um navegador (o cliente) solicite informações de um servidor web e receba uma resposta.

- A primeira versão do HTTP foi lançada em 1991 e era bastante simples, consistindo apenas de um conjunto de comandos que permitiam solicitar documentos e receber respostas do servidor.

FUNDAMENTOS WEB

- O HTTP já passou por diversas atualizações e revisões para acomodar novas tecnologias, como a criptografia SSL/TLS, a compressão de dados e a transferência de arquivos em vários formatos.

- Atualmente, o HTTP é amplamente utilizado como o protocolo padrão para a comunicação na web e é suportado por praticamente todos os navegadores e servidores web.

- A versão mais recente do protocolo é o HTTP/3, que foi aprovado em 2020 e tem como objetivo melhorar o desempenho e a segurança da web.

- Atualmente, não há um protocolo único que possa substituir completamente o HTTP (Hypertext Transfer Protocol) na web. No entanto, existem várias tecnologias e protocolos que podem ser utilizados em conjunto com o HTTP ou que oferecem funcionalidades específicas para determinados tipos de aplicações na web.

FUNDAMENTOS WEB

- Protocolos com funcionalidades especificas (similares ao HTTP):

  - **HTTPS (HTTP Secure):** uma versão do HTTP que utiliza criptografia SSL/TLS para proteger as informações transmitidas entre o cliente e o servidor. O HTTPS é uma evolução natural do HTTP e oferece maior segurança e privacidade na web.

  - **SPDY:** um protocolo desenvolvido pelo Google que tem como objetivo melhorar a velocidade de carregamento de páginas web, reduzindo o tempo de espera entre as solicitações do cliente e as respostas do servidor.

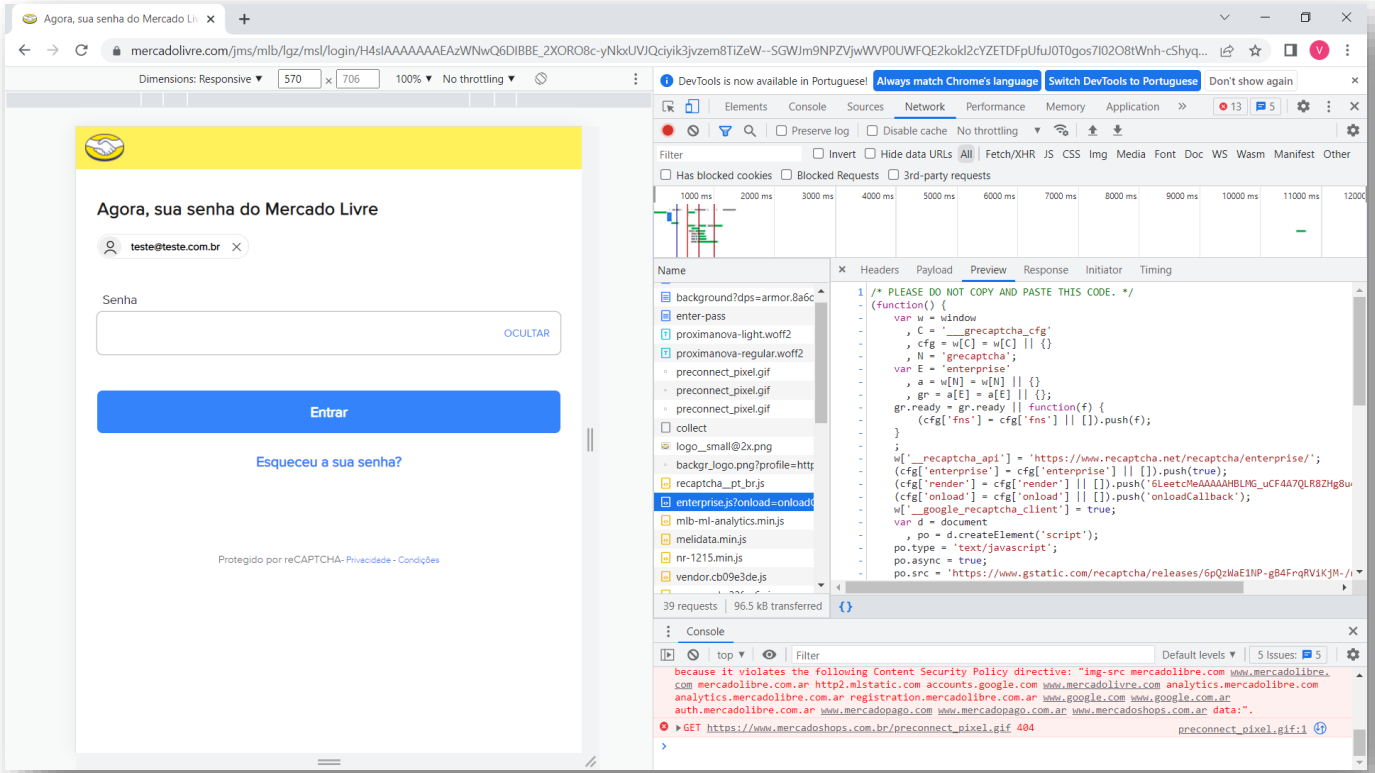
  - **HTTP/2:** uma versão mais recente do HTTP que introduziu novas funcionalidades para melhorar o desempenho da web, como a compressão de cabeçalhos e a multiplexação de conexões.

  - **QUIC:** um protocolo de transporte desenvolvido pelo Google que visa reduzir a latência e melhorar a segurança da web, combinando elementos do TCP e do UDP.

  - **WebSockets:** um protocolo que permite que uma conexão bidirecional entre o cliente e o servidor seja estabelecida, permitindo a transmissão de dados em tempo real.

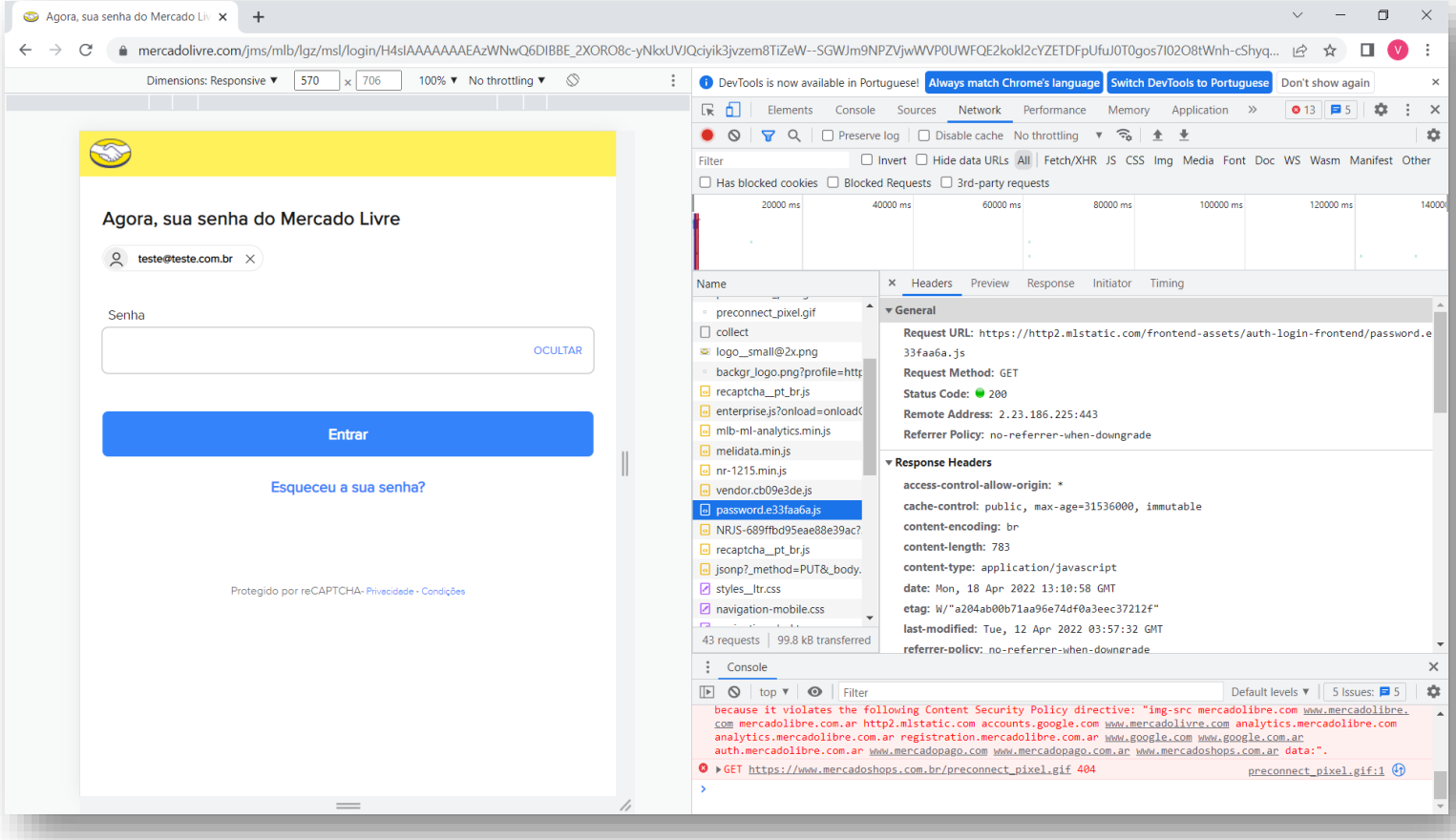
VERSÃO SEGURA DO HTTP:

- Abrindo o navegador Chromer e entrando em um site aleatório, observe pelo modo desenvolvedor na aba Network que a comunicação de rede acompanha alguns itens de comunicação (pacotes e arquivos):

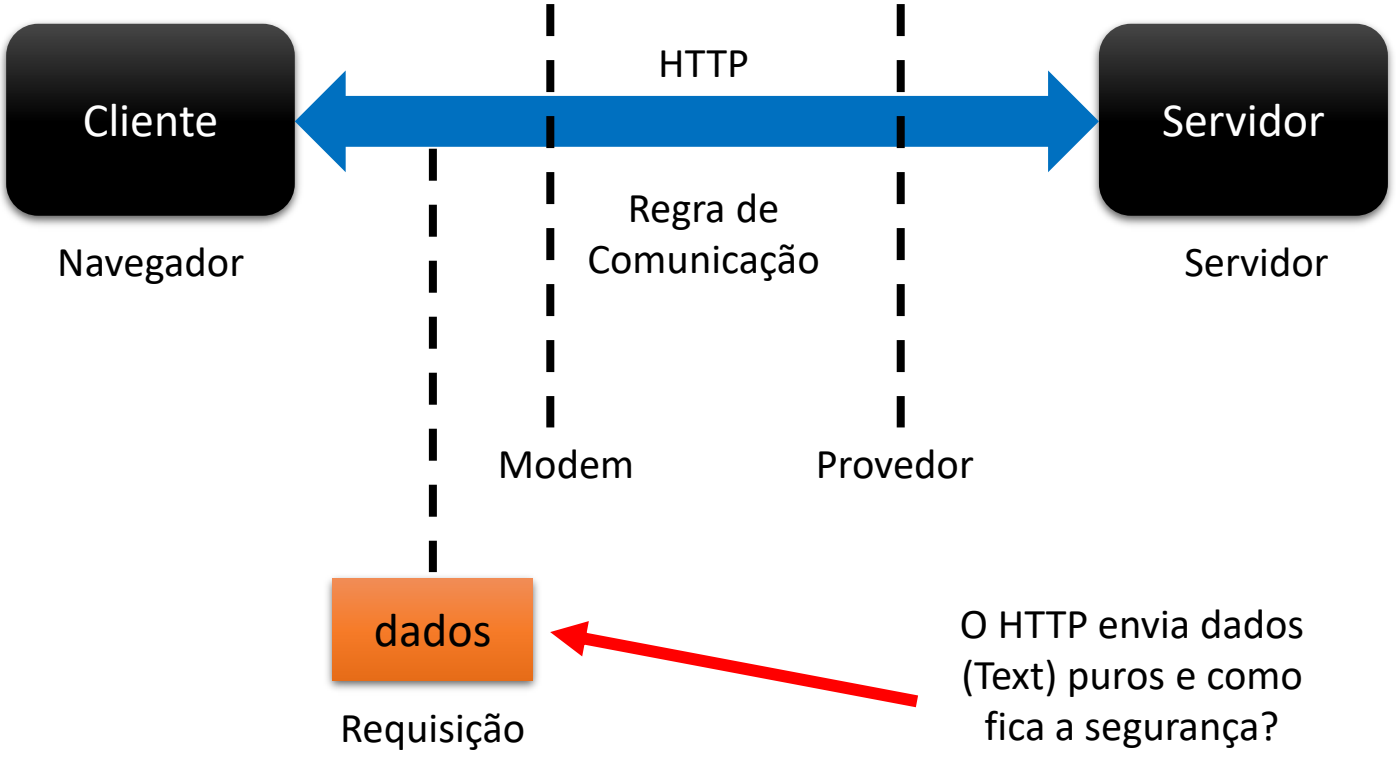


FUNDAMENTOS WEB

- Essa comunicação ocorre com endereços de URLs, métodos, endereços, etc.

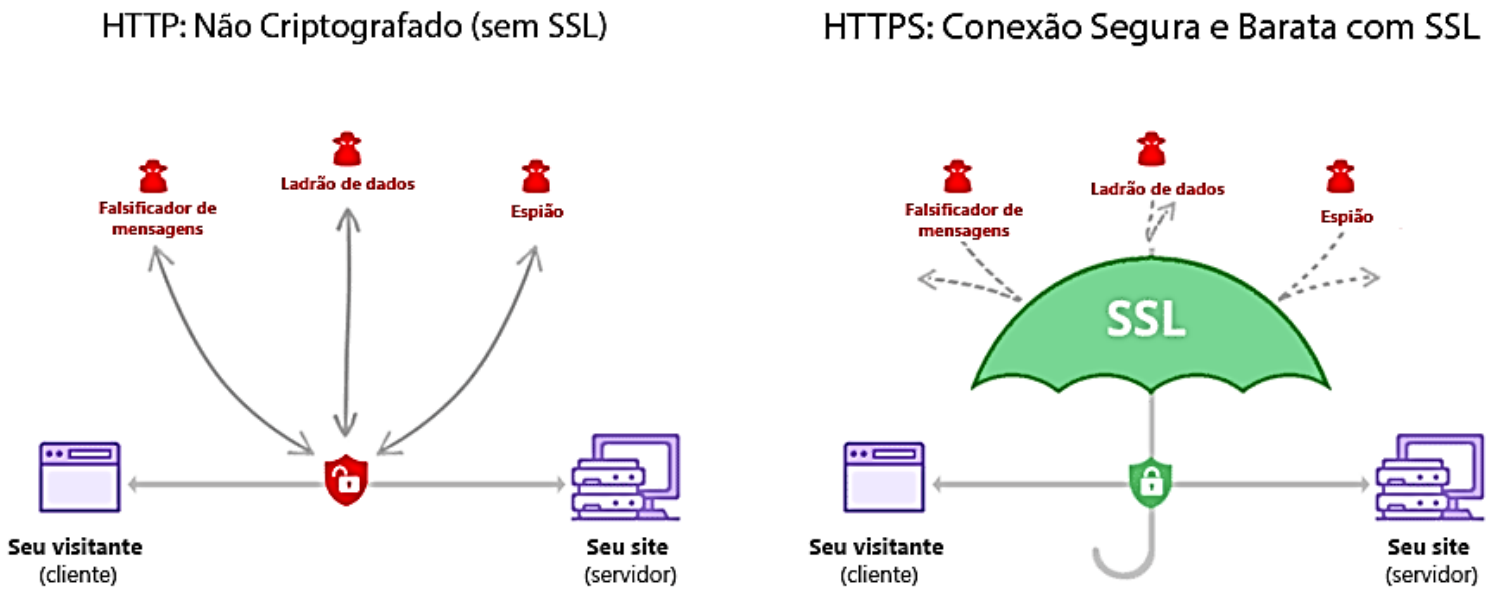


- Isso demonstra que quando ocorre a comunicação do cliente com o servidor, ocorre o envio de vários arquivos, pacotes que passam por outros equipamentos como roteadores, modem, provedores, etc.



PROTOCOLO – HTTPS:

- Para isso existe o HTTPS (HTTP + SSL/TLS):
- *HyperText Transfer Protocol + Secure Sockets Layer / Transport Layer Security*



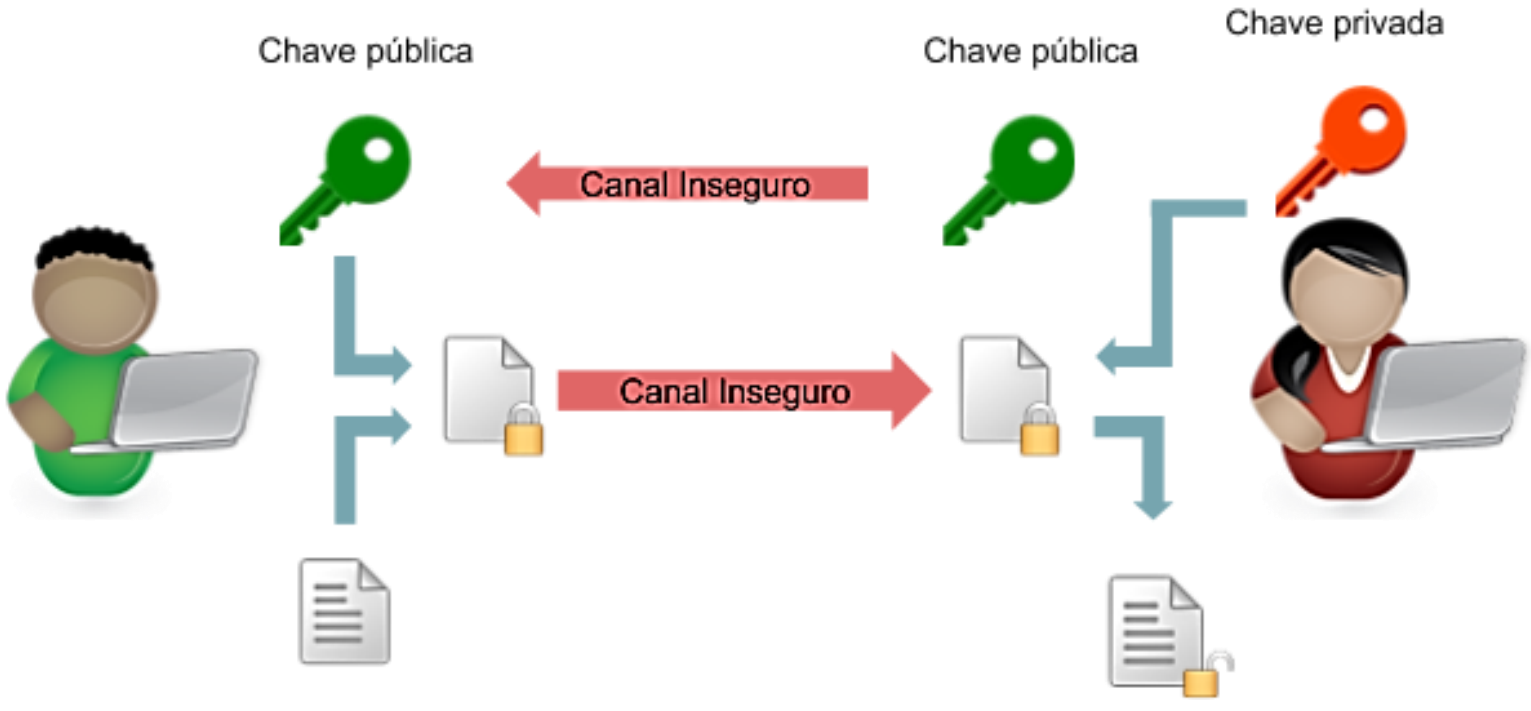
CERTIFICADO DIGITAL:

- Para que a conexão seja segura (HTTPS), exige a necessidade de uma identificação no qual chamamos de **CERTIFICADO DIGITAL**.

- O Ceritificado Digital, apresenta uma “chave” que vai criptografar os dados que serão enviados. Chamamos essa chave de **CHAVE PUBLICA**.

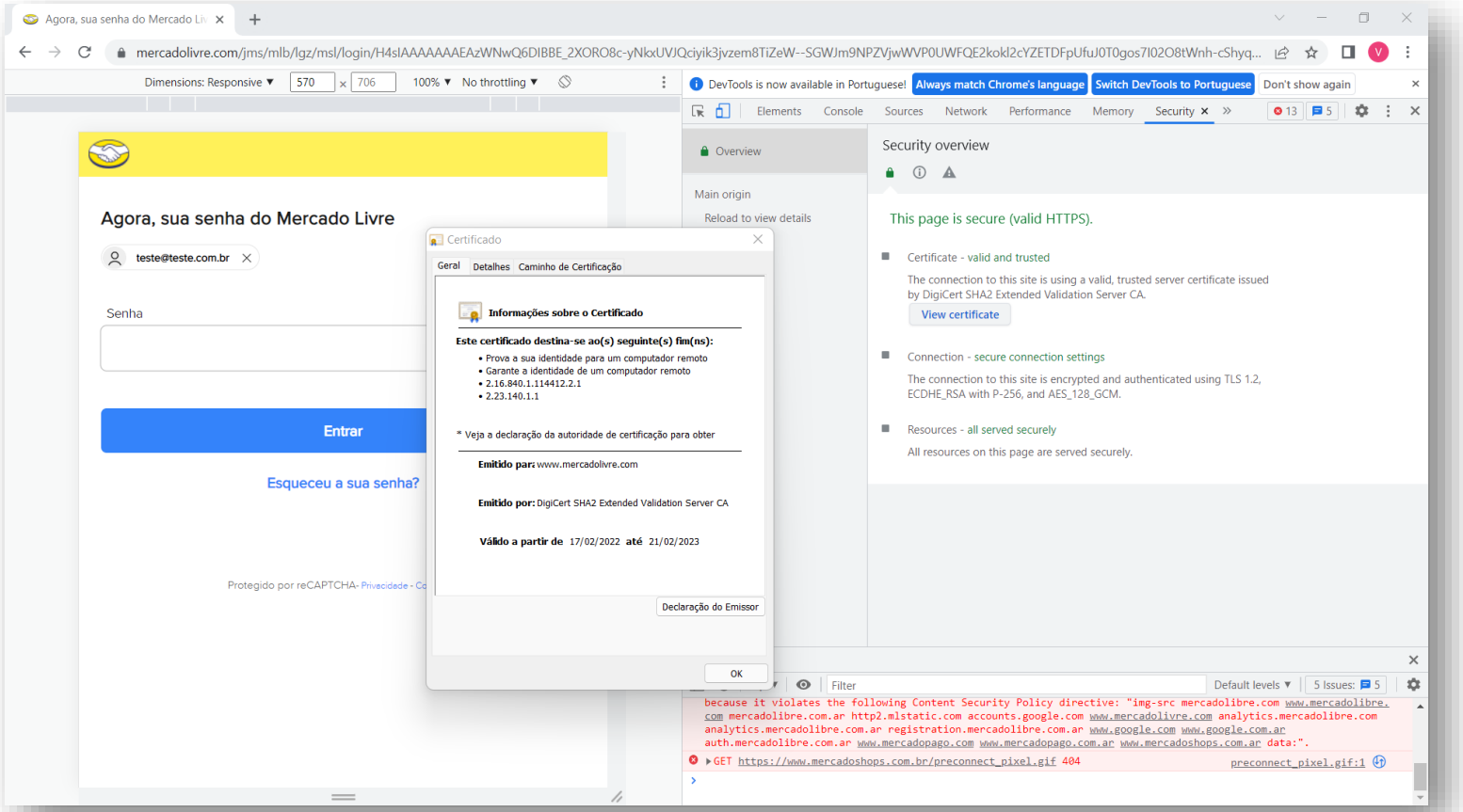
- No outro lado (servidor), haverá uma chave que apenas o desenvolvedor (proprietário) vai conhecer no qual assina a certificação no qual chamamos de **CHAVE PRIVADA (CHAVE SECRETA)**

- Isso permite uma comunicação entre CLIENTE-SERVIDOR de forma segura sem “invasão” de dados:



FUNDAMENTOS WEB

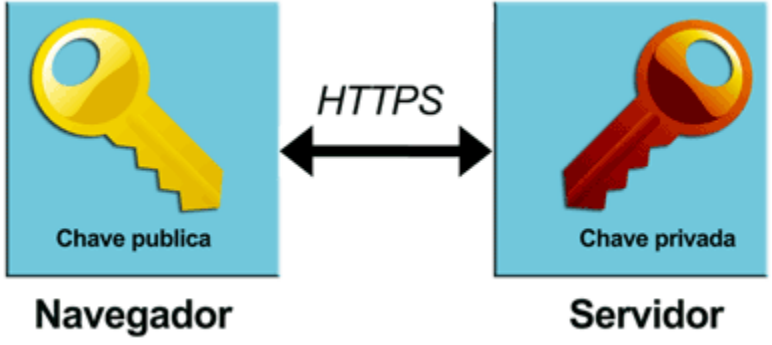
- Retornando ao nosso exemplo www.mercadolivre.com.br observe que existe uma aba security que mostra esse certificado de comunicação:



- As chaves estão ligadas matematicamente, o que foi cifrado pela chave pública só pode ser decifrado pela chave privada.

- Isso garante que os dados cifrados pelo navegador (chave pública) só podem ser lidos pelo servidor (chave privada).

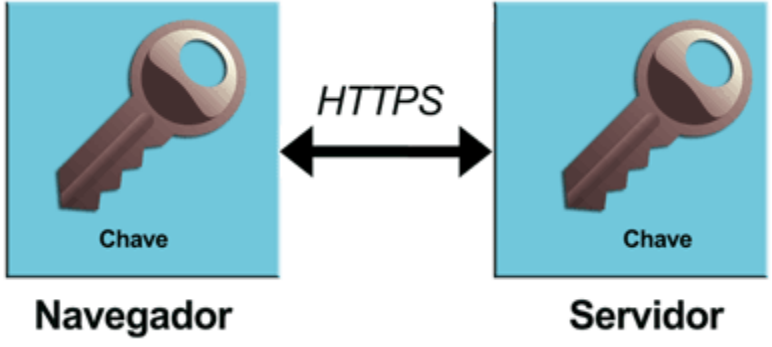
- Como temos duas chaves diferentes envolvidas, esse método de criptografia é chamado de **criptografia assimétrica**. No entanto, a criptografia assimétrica tem um problema, ela é lenta.



- Por outro lado, temos a **criptografia simétrica**, que usa a mesma chave para cifrar e decifrar os dados, como na vida real, onde usamos a mesma chave para abrir e fechar a porta.

- A criptografia simétrica é muito mais rápida, mas infelizmente não tão segura.

- Como existe apenas uma chave, ela ficará espalhada pelos clientes (navegadores) e qualquer um, que tem a posse dessa chave, pode decifrar a comunicação.



- o HTTPS usa ambos os métodos de criptografia, assimétrica e simétrica.
- Uma chave só para ele e o servidor com o qual está se comunicando naquele momento!
- Essa chave exclusiva (e simétrica) é então enviada para o servidor utilizando a criptografia assimétrica (chave privada e pública) e então é utilizada para o restante da comunicação.
- O HTTPS começa com criptografia assimétrica para depois mudar para criptografia simétrica.
- Essa chave simétrica será gerada no início da comunicação e será reaproveitada nas requisições seguintes. Bem-vindo ao mundo fantástico do HTTPS

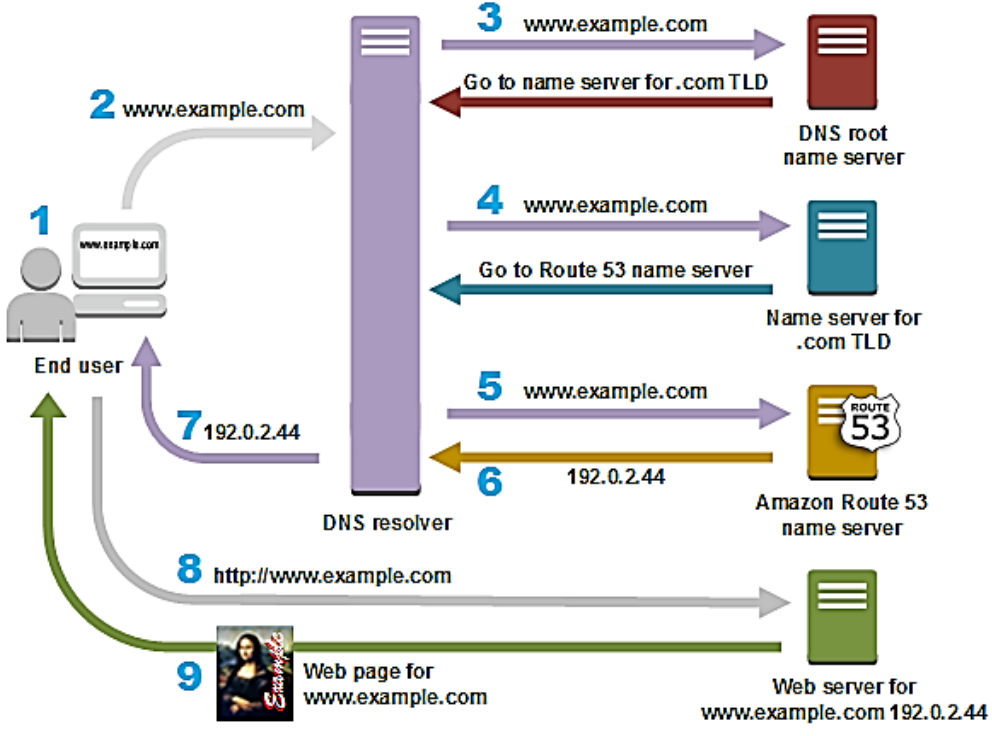
ENDEREÇOS NO HTTP:

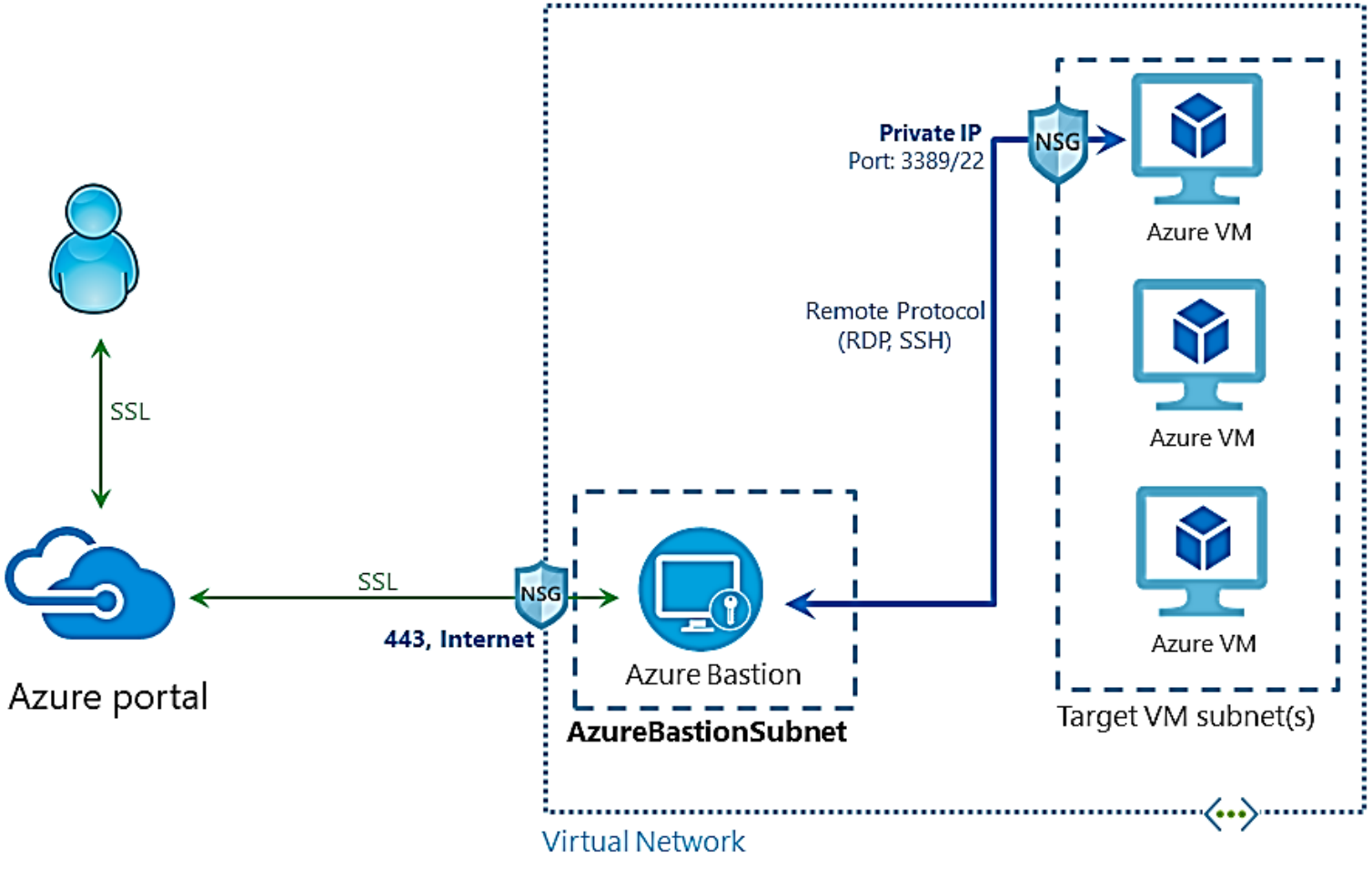
- Na internet é acessado alguns endereços que representa um endereço IP no qual é resolvido por DNS.
- Em sites de internet o endereço é apresentado em domínios (textuais), porem em servidores o mais comum é por endereços IPs

FUNDAMENTOS WEB

- Esse mapeamento e resolução de endereço www.exemplo.com para um IP é realizado pelo DNS:

- DNS (Domain Name System ou servidor de domínios) realiza a tradução do nome de um domínio para o endereço de IP. Existem vários servidores DNS no mundo e é fundamental para a nossa web o funcionamento deles.




Private IP
Port: 3389/22
NSG
Azure VM
Azure VM
Azure VM
Remote Protocol
(RDP, SSH)
SSL
SSL
NSG
443, Internet
Azure Bastion
AzureBastionSubnet
Azure portal
Target VM subnet(s)
Virtual Network

SERVIDOR WEB – WEB SERVER

- Um web server, em português servidor web, é um computador ou programa que é responsável por fornecer conteúdo e recursos da web para os usuários que acessam a internet.

- Imagine que você queira acessar um site, como o Google, por exemplo. O seu navegador (como o Chrome, Firefox, Safari) envia uma solicitação para o servidor web que hospeda o site do Google. O servidor web processa essa solicitação e envia de volta ao seu navegador o conteúdo da página que você solicitou, como imagens, vídeos e textos.

- O servidor web também pode executar outras funções, como processamento de formulários de contato, verificação de senha de usuários e armazenamento de dados em bancos de dados.

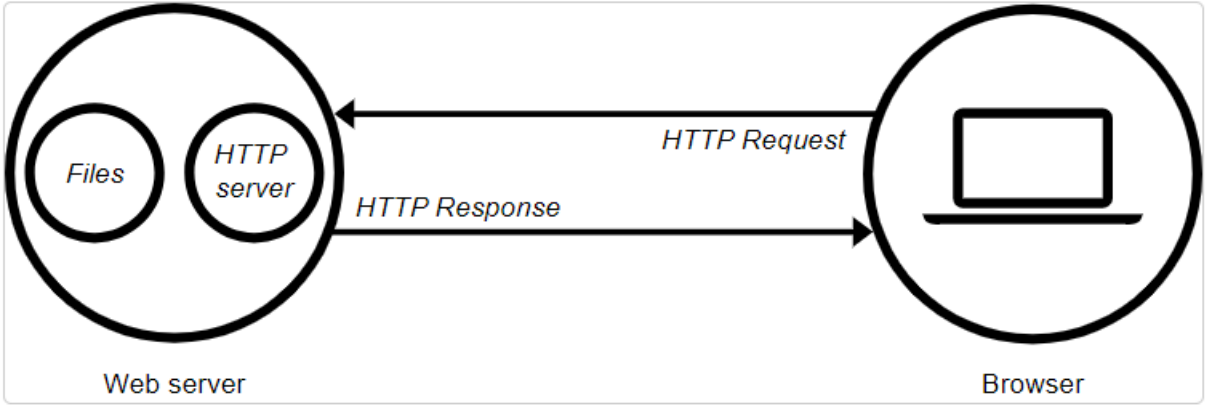
- Portanto, podemos pensar em um servidor web como um "mordomo da web" que cuida de tudo para que você possa acessar e interagir com os sites na internet.

FUNDAMENTOS WEB

- Quando falamos de um Web Service, sempre usamos o protocolo da web, ou seja o HTTP. Um Web Service disponibiliza uma funcionalidade na web, através do protocolo HTTP.

- O importante é que sempre usamos o protocolo HTTP.

- A grande diferença de um Web Service é que os dados não vem no formato HTML, e sim em algum formato independente da visualização, como XML ou JSON.

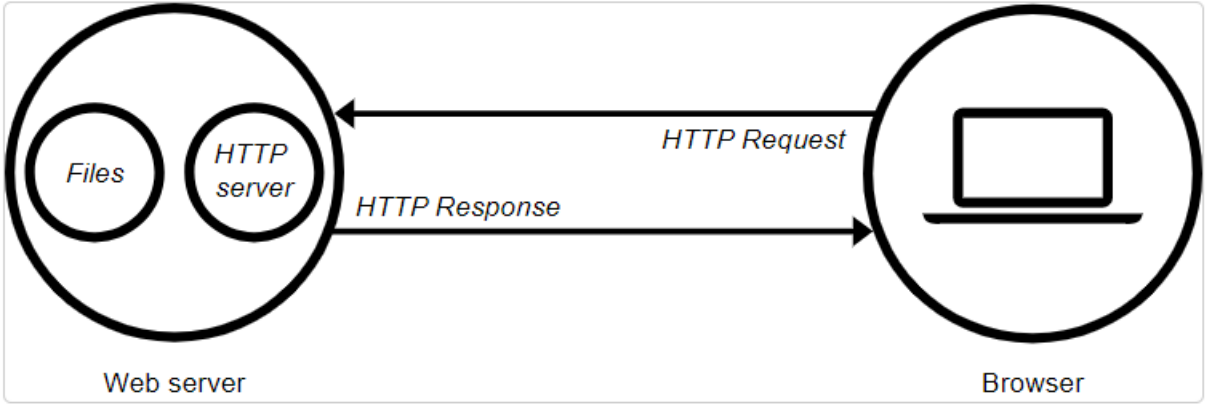


RESQUEST:

- Uma Request é um pedido que um cliente faz ao servidor, contendo informações sobre o que o cliente precisa.

- Exemplo: Caso o cliente quiser cadastrar um novo produto, ele deve fornecer os dados necessários para o cadastro, incluindo aqueles inseridos em um formulário.

- Sempre que o usuário interage com uma aplicação web, como ao mudar de página ou pressionar enter na barra de endereço, uma nova request é feita, independentemente da ação realizada, seja para exibir uma página, cadastrar, atualizar ou excluir um recurso.

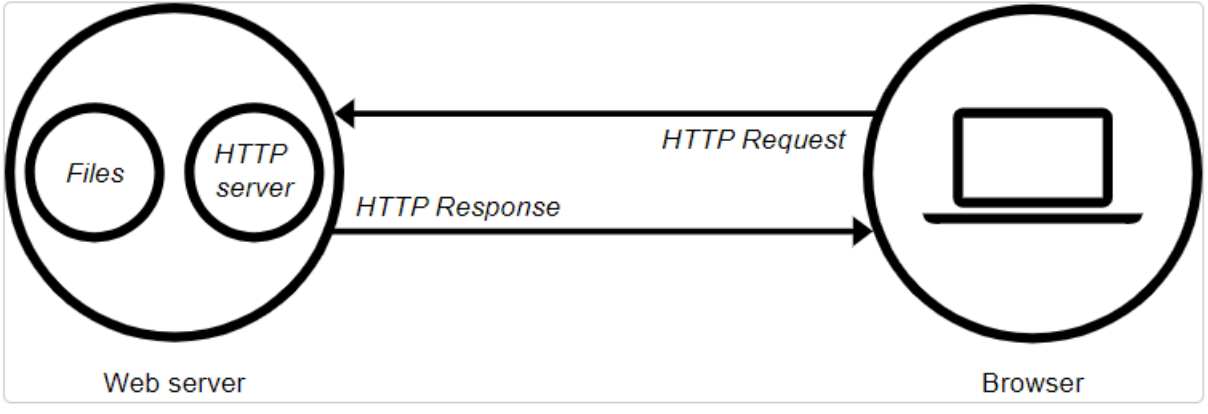


RESPONSE:

- Quando o cliente faz uma Request (pedido) para o servidor web, ele espera receber uma resposta de volta, chamada Response (resposta).

- A resposta pode conter os dados que o cliente solicitou, ou uma mensagem de erro caso algo tenha dado errado.

- Em resumo, a Response é a resposta que o servidor envia de volta para o cliente após receber a Request.



CATEGORIA DO HTTP:

- Os códigos HTTP (ou HTTPS) tem três dígitos, sendo que o primeiro dígito do código significa a classificação dentro das cinco categorias.

- **1XX:** Informativo – a solicitação foi aceita ou o processo continua em andamento;
- **2XX:** Confirmação – a ação foi concluída ou entendida;
- **3XX:** Redirecionamento – indica que algo mais precisa ser feito ou precisou ser feito para completar a solicitação;
- **4XX:** Erro do cliente- indica que a solicitação não pode ser concluída ou contém a sintaxe incorreta;
- **5XX:** Erro no servidor – o servidor falhou ao concluir a solicitação.

- Traduzidos para o português estes termos significam **código de status e frase razão**, que são os elementos que compõe o HTTP.

- Status-code são os três dígitos que indicam qual o erro para o servidor e navegador enquanto a frase razão é uma curta descrição do que este erro significa para melhor compreensão dos usuários.

- Estes códigos e razões estão descritos na tabela abaixo para o seu conhecimento.

- Leia mais detalhes em: https://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html#sec6.1.1

- Site para teste de status de código HTTP: https://savanttools.com/test-http-status-codes

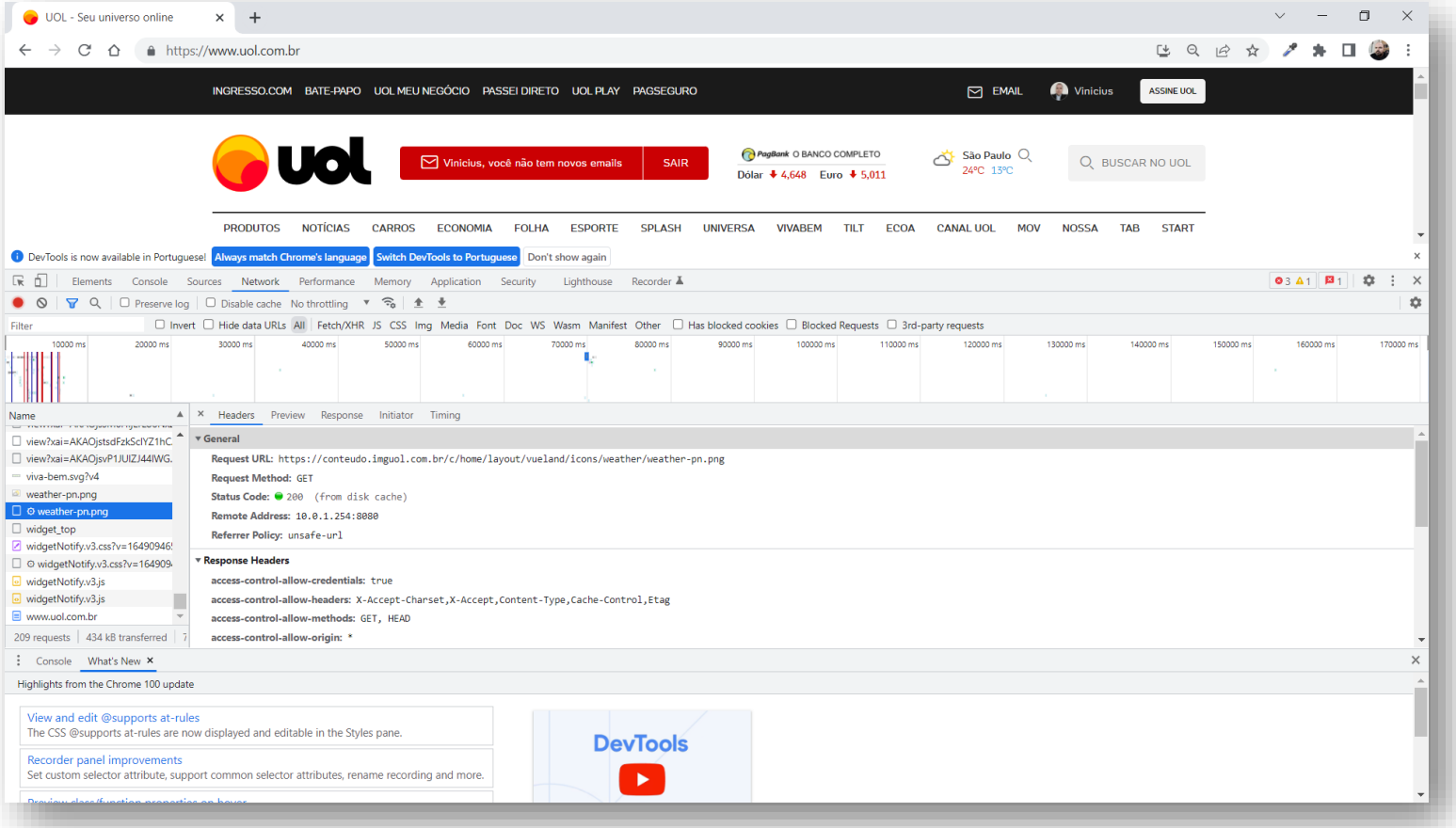
FUNDAMENTOS WEB

Código do Status HTTP (Status-code)	Significado do código HTTP (Reason-Phrase)	Significado do código HTTP
100	Continue	Continuar
101	Switching Protocols	Mudando Protocolos
102	Processing	Processando
200	Ok	Ok
201	Created	Criado
202	Accepted	Aceito
203	Non-Authoritative Information	Não autorizado
204	No Content	Nenhum Conteúdo
205	Reset Content	Resetar Conteúdo
206	Partial Content	Conteúdo Parcial
300	Multiple Choices	Múltipla Escolha
301	Moved Permanently	Movido Permanentemente
302	Found	Encontrado
303	See Other	Veja outro
304	Not Modified	Não modificado
305	Use Proxy	Use Proxy
306	Proxy Switch	Proxy Trocado
400	Bad Request	Solicitação Inválida
401	Unauthorized	Não autorizado
402	Payment Required	Pagamento necessário
403	Forbidden	Proibido
404	Not Found	Não encontrado
405	Method Not Allowed	Método não permitido
406	Not Acceptable	Não aceito
407	Proxy Authentication Required	Autenticação de Proxy Necessária
408	Request Time-out	Tempo de solicitação esgotado
409	Conflict	Conflito
410	Gone	Perdido
411	Length Required	Duração necessária
412	Precondition Failed	Falha de pré-condição
413	Request Entity Too Large	Solicitação da entidade muito extensa
414	Request-URL Too Large	Solicitação de URL muito Longa
415	Unsupported Media Type	Tipo de mídia não suportado
416	Request Range Not Satisfiable	Solicitação de faixa não satisfatória
417	Expectation Failed	Falha na expectativa

500	Internal Server Error	Erro do Servidor Interno
501	Not Implemented	Não implementado
502	Bad Gateway	Porta de entrada ruim
503	Service Unavailable	Serviço Indisponível
504	Gateway Time-out	Tempo limite da Porta de Entrada
505	HTTP Version Not Supported	Versão HTTP não suportada

FUNDAMENTOS WEB

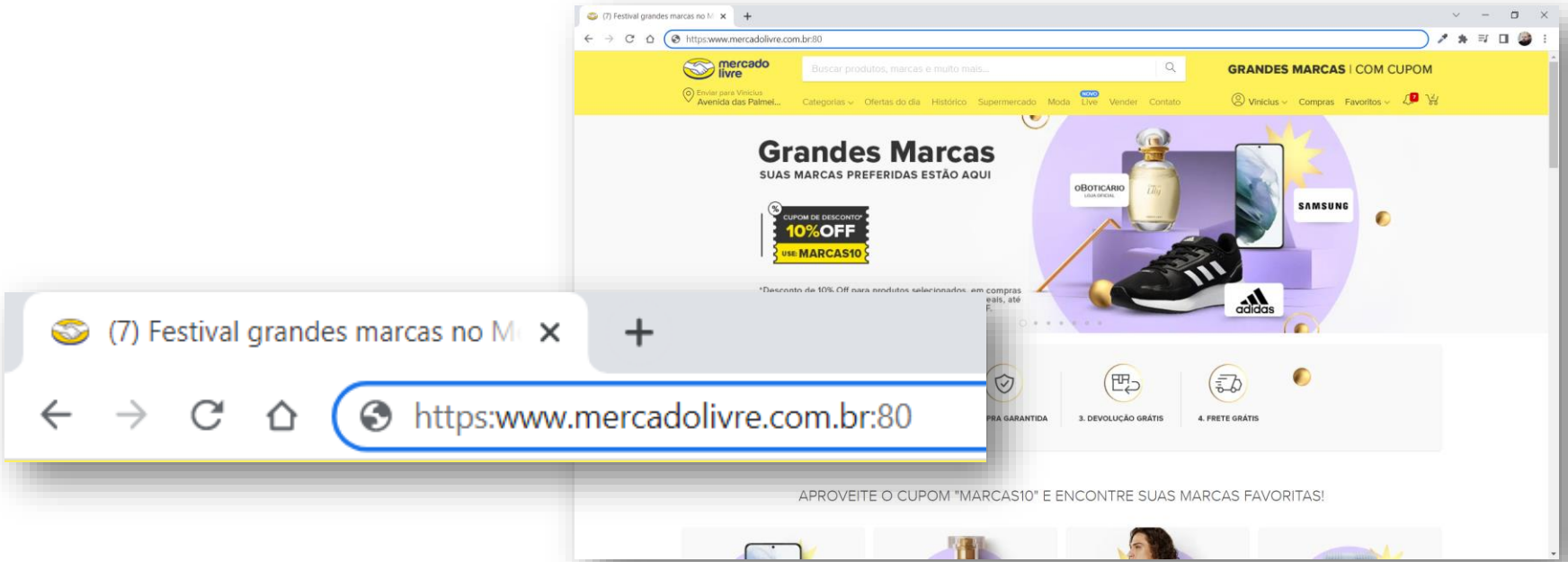
DEPURANDO A REQUISIÇÃO:



PORTAS

- Portas funciona como “apartamentos”. Dentro de um condomínio o acesso a determinado apartamento é realizado pelo numero do apartamento. Em um servidor, tem diversas portas que representam serviços, comunicações de acessos;

- O HTTP usa a **porta 80**. no HTTPS usa a **porta 443** (tente alterar a porta de comunicação):



FUNDAMENTOS WEB

- As portas são utilizadas no protocolo web para permitir que diferentes serviços sejam executados em um único servidor compartilhando o mesmo endereço IP, e para garantir que as solicitações HTTP sejam enviadas ao serviço correto no servidor.

- Quando um cliente faz uma solicitação HTTP a um servidor, ele especifica a porta que deve ser usada para essa conexão.

- Por padrão, o HTTP utiliza a porta 80 para solicitações não criptografadas (HTTP) e a porta 443 para solicitações criptografadas (HTTPS). No entanto, é possível configurar o servidor para utilizar outras portas para serviços específicos, como o SMTP (Simple Mail Transfer Protocol) para e-mails.

- Por exemplo, se um usuário digita "http://www.exemplo.com" em seu navegador, a solicitação é enviada ao servidor na porta 80.

- Caso a solicitação fosse "https://www.exemplo.com", a solicitação seria enviada na porta 443.

- No entanto, se o servidor estiver configurado para utilizar uma porta diferente, o cliente deve especificar a porta correta na solicitação, como "http://www.exemplo.com:8080".

FUNDAMENTOS WEB

- Principais portas e finalidades:
  - **Porta 80:** Acesso a páginas web através do protocolo **HTTP** (Hypertext Transfer Protocol).
  - **Porta 443:** Acesso a páginas web através do protocolo **HTTPS** (Hypertext Transfer Protocol Secure), que é utilizado para garantir a segurança na transmissão de informações.
  - **Porta 21:** Conexões **FTP** (File Transfer Protocol), utilizadas para transferência de arquivos.
  - **Porta 22:** Conexões **SSH** (Secure Shell), utilizadas para acesso remoto seguro a servidores.
  - **Porta 25:** Conexões **SMTP** (Simple Mail Transfer Protocol), utilizadas para envio de e-mails.
  - **Porta 110:** Conexões **POP3** (Post Office Protocol version 3), utilizadas para recebimento de e-mails.
  - **Porta 143:** Conexões **IMAP** (Internet Message Access Protocol), utilizadas para acesso a e-mails em servidores.
  - **Porta 53:** Conexões **DNS** (Domain Name System), utilizadas para resolver nomes de domínios em endereços IP.

- Para saber qual porta está sendo utilizada em um computador com Windows através do Prompt de Comando, você pode utilizar o comando "**netstat**".

- O comando "**netstat**" permite visualizar as conexões de rede ativas no computador, incluindo as portas que estão sendo usadas.

```
C:\>netstat

Conexões ativas

  Proto  Endereço local          Endereço externo       Estado
  TCP    10.67.226.109:49429     52.226.139.180:https   ESTABLISHED
  TCP    10.67.226.109:55753     82.202.184.205:https   ESTABLISHED
  TCP    10.67.226.109:55765     218:https              ESTABLISHED
  TCP    10.67.226.109:55768     47:https               ESTABLISHED
  TCP    10.67.226.109:55871     52.113.207.4:https     ESTABLISHED
  TCP    10.67.226.109:55881     47:https               ESTABLISHED
  TCP    10.67.226.109:56078     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56086     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56087     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56098     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56102     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56109     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56112     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56168     52.97.4.162:https      ESTABLISHED
  TCP    10.67.226.109:56191     52.226.139.121:https   ESTABLISHED
  TCP    10.67.226.109:56219     52.226.139.121:https   ESTABLISHED
  TCP    10.67.226.109:56694     47:https               ESTABLISHED
  TCP    10.67.226.109:56823     25:https               TIME_WAIT
  TCP    10.67.226.109:56894     ce-in-f188:5228        ESTABLISHED
  TCP    10.67.226.109:57003     52.97.12.66:https      ESTABLISHED
  TCP    10.67.226.109:57549     104.18.2.161:https     ESTABLISHED
  TCP    10.67.226.109:58254     82.202.184.202:https   ESTABLISHED
  TCP    10.67.226.109:58288     52.113.207.4:https     ESTABLISHED
  TCP    10.67.226.109:58537     52.97.34.98:https      ESTABLISHED
```

FUNDAMENTOS WEB

- Digite o comando "**netstat -ano**" para exibir uma lista de todas as conexões de rede ativas no computador, juntamente com as portas associadas a elas.

```
C:\>netstat -ano

Conexões ativas

  Proto  Endereço local          Endereço externo       Estado           PID
  TCP    0.0.0.0:135             0.0.0.0:0              LISTENING        1340
  TCP    0.0.0.0:445             0.0.0.0:0              LISTENING        4
  TCP    0.0.0.0:3306            0.0.0.0:0              LISTENING        5712
  TCP    0.0.0.0:5040            0.0.0.0:0              LISTENING        10964
  TCP    0.0.0.0:5357            0.0.0.0:0              LISTENING        4
  TCP    0.0.0.0:33060           0.0.0.0:0              LISTENING        5712
  TCP    0.0.0.0:49664           0.0.0.0:0              LISTENING        1068
  TCP    0.0.0.0:49665           0.0.0.0:0              LISTENING        888
  TCP    0.0.0.0:49666           0.0.0.0:0              LISTENING        3148
  TCP    0.0.0.0:49667           0.0.0.0:0              LISTENING        3440
  TCP    0.0.0.0:49668           0.0.0.0:0              LISTENING        4288
  TCP    0.0.0.0:49683           0.0.0.0:0              LISTENING        1044
  TCP    0.0.0.0:55759           0.0.0.0:0              LISTENING        9452
  TCP    0.0.0.0:57621           0.0.0.0:0              LISTENING        9452
  TCP    10.67.226.109:139       0.0.0.0:0              LISTENING        4
  TCP    10.67.226.109:49429     52.226.139.180:443     ESTABLISHED      5540
  TCP    10.67.226.109:55753     82.202.184.205:443     ESTABLISHED      4672
  TCP    10.67.226.109:55765     34.158.253.218:443     ESTABLISHED      9452
  TCP    10.67.226.109:55768     35.186.224.47:443      ESTABLISHED      9452
  TCP    10.67.226.109:55871     52.113.207.4:443       ESTABLISHED      19324
  TCP    10.67.226.109:55881     35.186.224.47:443      ESTABLISHED      6096
  TCP    10.67.226.109:56078     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56086     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56087     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56098     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56102     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56109     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56112     52.97.4.162:443        ESTABLISHED      20284
  TCP    10.67.226.109:56168     52.97.4.162:443        ESTABLISHED      20284
```

```
TCP    127.0.0.1:57854        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:57855        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:57974        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:57982        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:57996        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:57997        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:58051        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:58052        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:58252        127.0.0.1:58253        ESTABLISHED     9304
TCP    127.0.0.1:58253        127.0.0.1:58252        ESTABLISHED     9304
TCP    127.0.0.1:58378        127.0.0.1:62522        ESTABLISHED     15504
TCP    127.0.0.1:58451        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:58452        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:58653        127.0.0.1:49678        ESTABLISHED     26004
TCP    127.0.0.1:62522        0.0.0.0:0              LISTENING       12656
TCP    127.0.0.1:62522        127.0.0.1:58378        ESTABLISHED     12656
TCP    192.168.56.1:139       0.0.0.0:0              LISTENING       4
TCP    [::]:135               [::]:0                 LISTENING       1340
TCP    [::]:445               [::]:0                 LISTENING       4
TCP    [::]:3306              [::]:0                 LISTENING       5712
TCP    [::]:5357              [::]:0                 LISTENING       4
TCP    [::]:33060             [::]:0                 LISTENING       5712
TCP    [::]:49664             [::]:0                 LISTENING       1068
TCP    [::]:49665             [::]:0                 LISTENING       888
TCP    [::]:49666             [::]:0                 LISTENING       3148
TCP    [::]:49667             [::]:0                 LISTENING       3440
TCP    [::]:49668             [::]:0                 LISTENING       4288
TCP    [::]:49683             [::]:0                 LISTENING       1044
TCP    [::1]:42050            [::]:0                 LISTENING       24600
TCP    [::1]:49669            [::]:0                 LISTENING       5796
```

PORTQRY - PORTQRY COMMAND LINE PORT SCANNER VERSION 2.0

- Portqry é um utilitário de linha de comando da Microsoft que é usado para verificar o status das portas em um computador.

- O Portqry permite verificar se as portas de um determinado host estão abertas ou fechadas, além de exibir informações sobre os serviços que estão escutando nessas portas.

- O utilitário Portqry suporta vários protocolos de rede, incluindo TCP, UDP, ICMP e SNMP. Ele pode ser executado a partir do Prompt de Comando do Windows ou de um arquivo de script.

- Necessita instalar, podendo ser adquirido pelo site da Microsoft (sem custo):

https://www.microsoft.com/en-us/download/details.aspx?id=17148

- Para verificar quais portas estão livres com o Portqry, você pode usar o seguinte comando:

portqry -n <nome_do_host> -e <número_da_porta>

- Necessário estar dentro do diretório (após instalação):

```
C:\PortQryV2>portqry -n localhost -e 8080

Querying target system called:

 localhost

Attempting to resolve name to IP address...


Name resolved to 127.0.0.1

querying...

TCP port 8080 (unknown service): NOT LISTENING

C:\PortQryV2>
```

PARÂMETROS DE REQUISIÇÃO E WEB SERVER (VERBOS)

- Os métodos GET e POST são usados em requisições HTTP, que são formas de comunicar com um servidor.

- O método GET é usado quando queremos obter informações do servidor, enquanto o método POST é usado quando queremos enviar dados para o servidor para serem processados, como em um formulário. Esses são os métodos mais usados em aplicações web.

- Quando digita-se um endereço no navegador, é realizada uma requisição GET, e quando preenchemos um formulário e clicamos em enviar, é usado o método POST.

- Existem outros métodos HTTP além desses dois.

  - **GET** – é usado quando o cliente deseja obter recursos do servidor

  - **POST** – é usado quando o cliente deseja enviar dados para processamento ao servidor, como os dados de um formulário, por exemplo.

COOKING:

- São pequenos arquivos de texto que são armazenados no computador.
- São usados pelos sites para lembrar de informações sobre suas preferências, como suas escolhas de idioma ou suas credenciais de login.
- Os cookies são criados pelo site que foi visitado e são armazenados no computador por um período de tempo específico, dependendo das configurações do site.
- São usados para fornecer uma experiência de navegação personalizada, para fins de análise de dados e para fins de publicidade.
- Alguns cookies são essenciais para o funcionamento adequado de um site, como cookies de sessão que permitem que fazer login ou carrinho de compras que mantém os itens adicionados durante uma sessão de compras.
- Outros cookies são usados para rastrear o comportamento de navegação na internet e exibir anúncios relevantes com base em seus interesses.

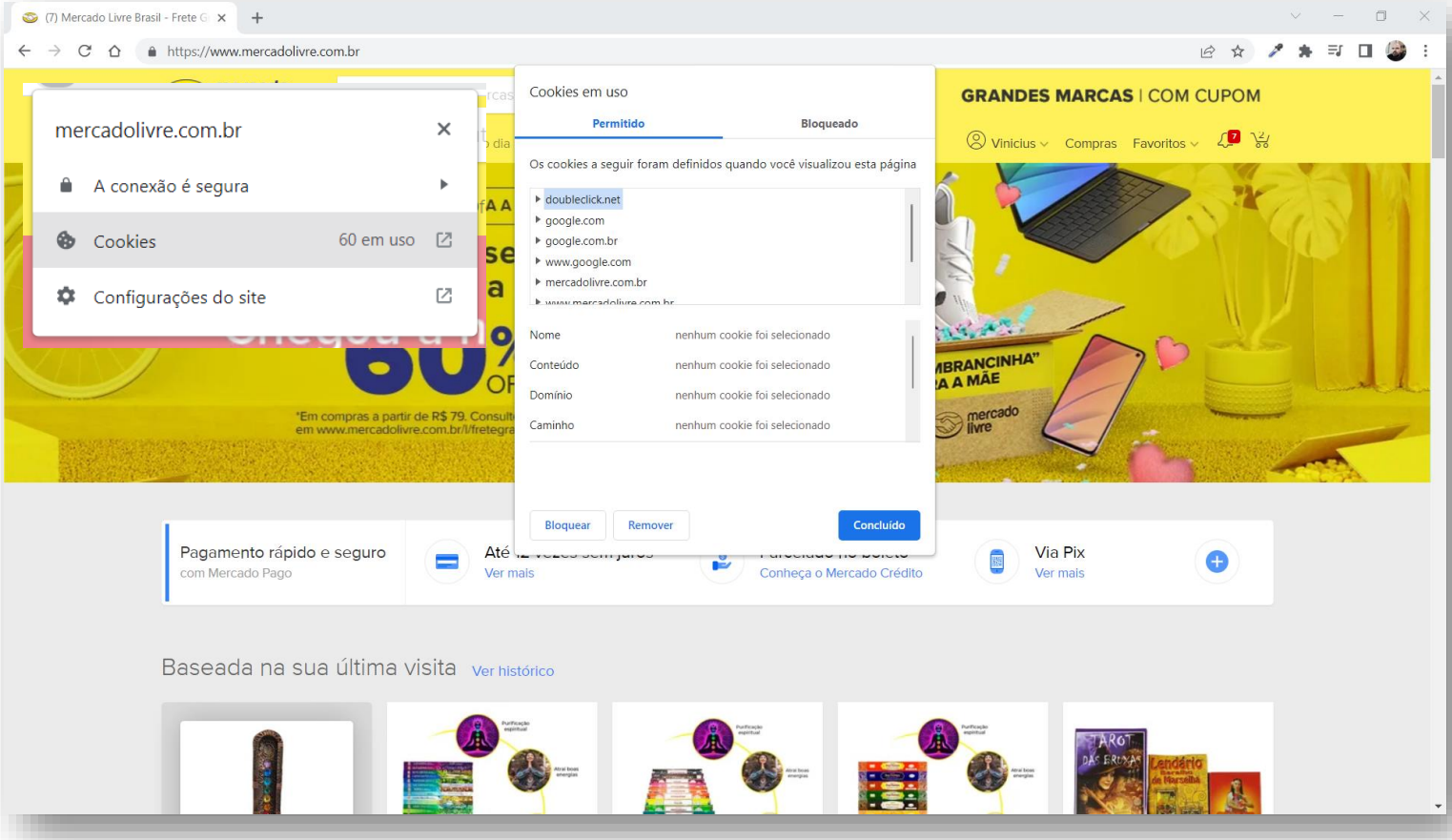
- Um cookie é um pequeno arquivo de texto, normalmente criado pela aplicação web, para guardar algumas informações sobre usuário no navegador.

- Quais são essas informações depende um pouco da aplicação. Pode ser que fique gravado alguma preferência do usuário. Ou algumas informações sobre as compras na loja virtual ou, como vimos no vídeo, a identificação do usuário. Isso depende da utilidade para a aplicação web.

- Um cookie pode ser manipulado e até apagado pelo navegador e, quando for salvo no navegador, fica associado com um domínio.

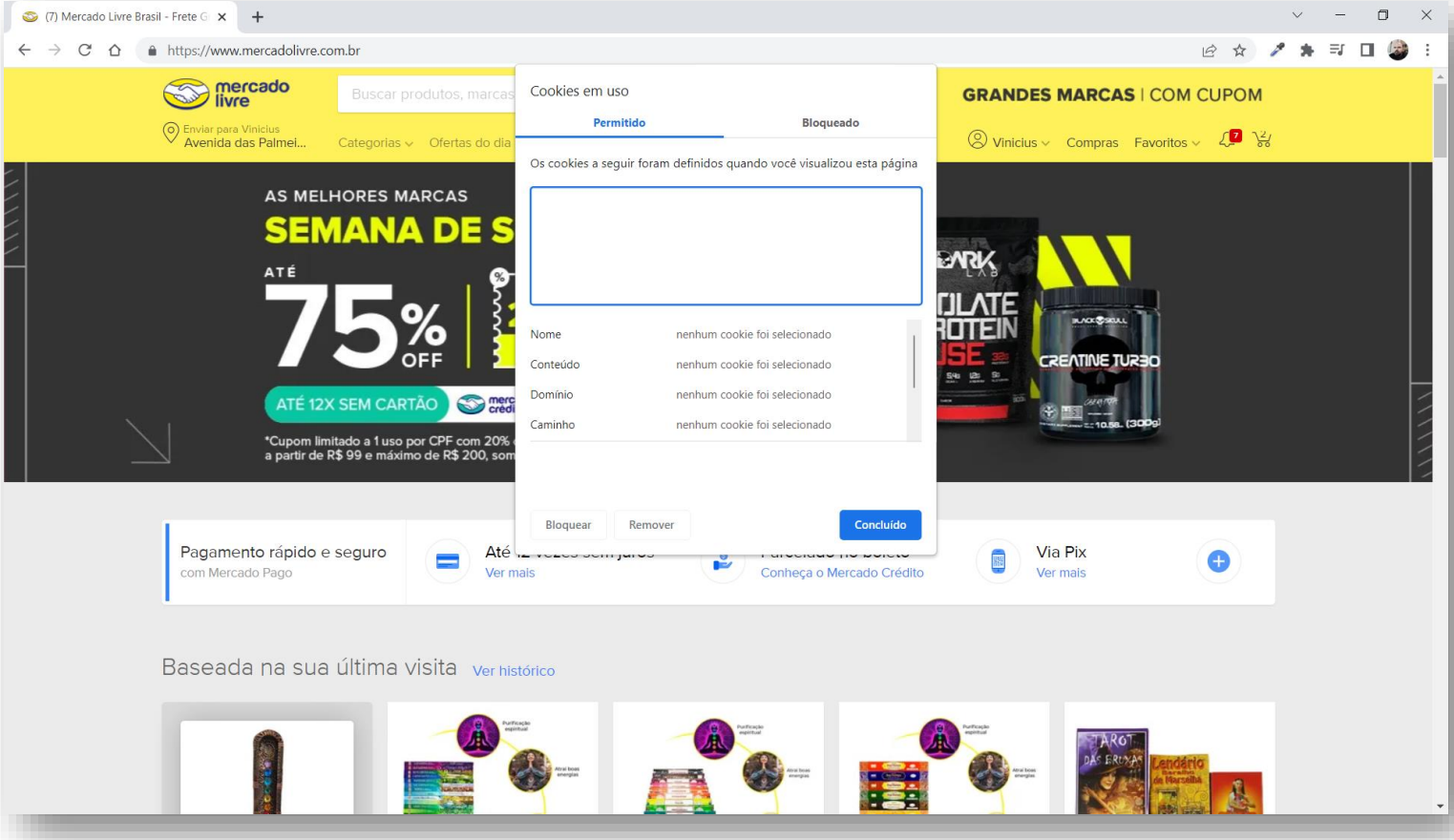
FUNDAMENTOS WEB

- Abrindo o navegador no site www.mercadolivre.com.br, observe que temos alguns cookies em uso:

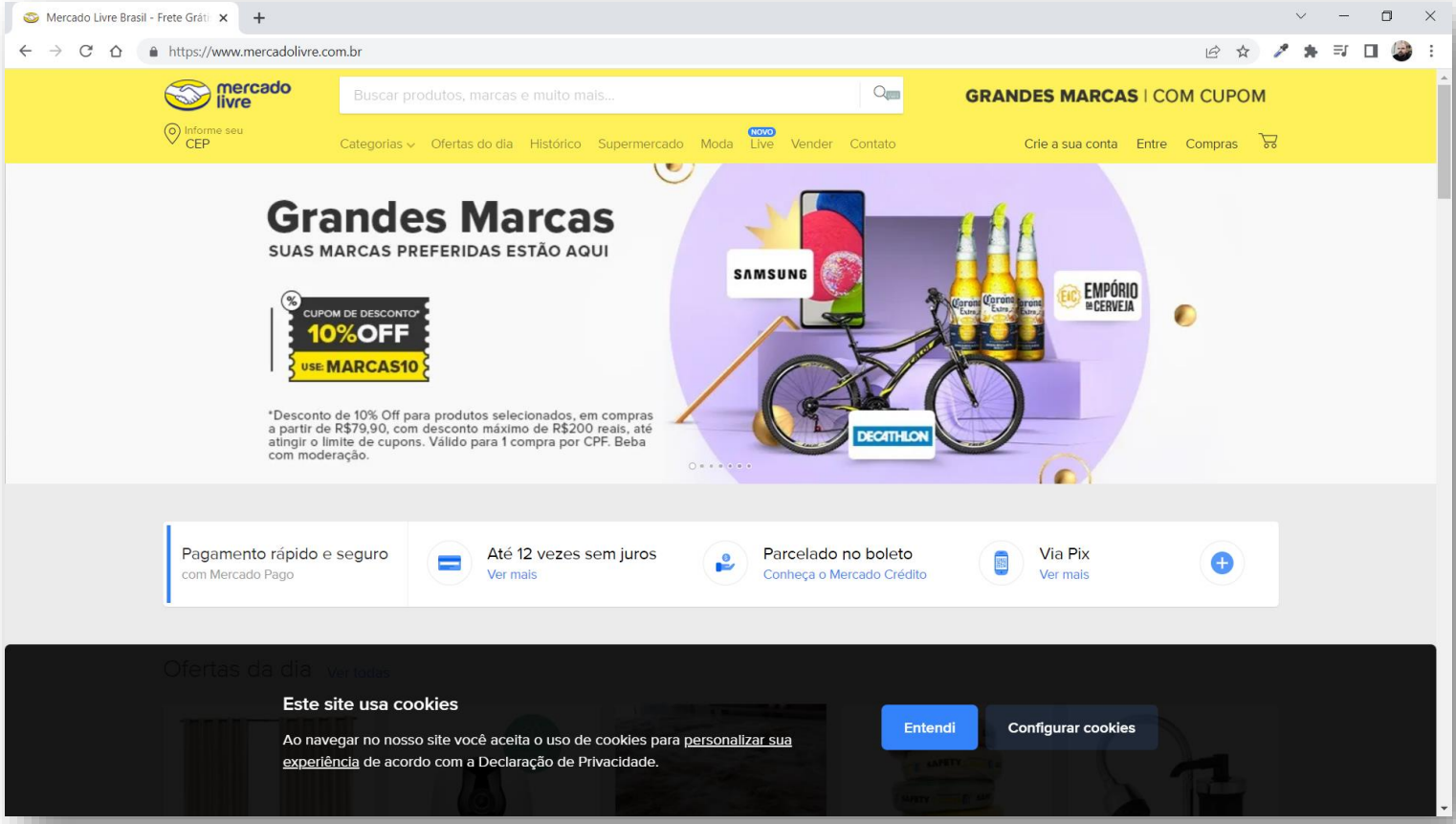


FUNDAMENTOS WEB

- Apagando os cookies

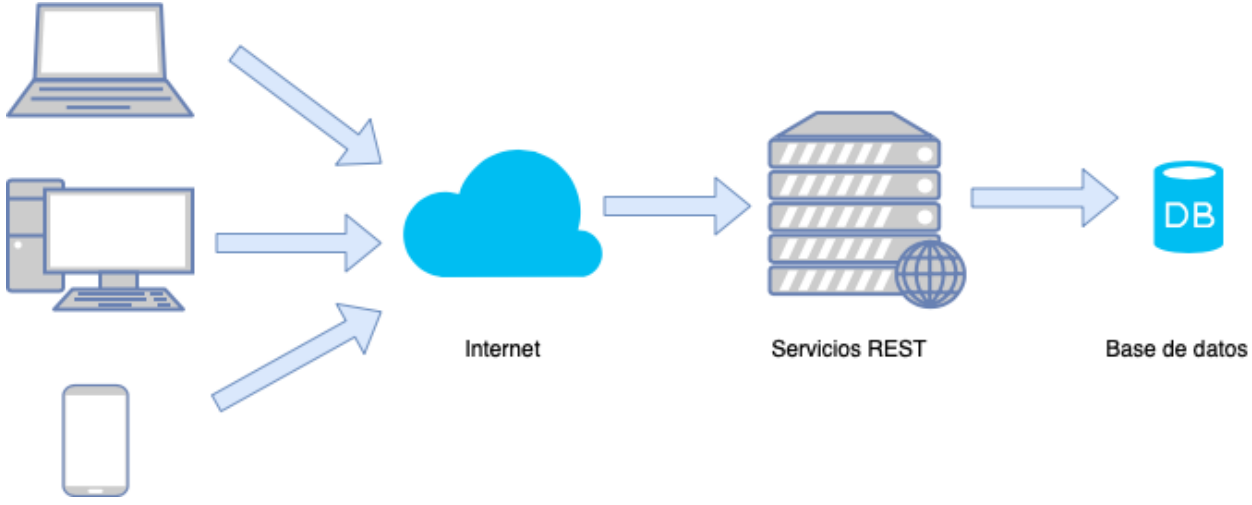


- Após apagando os cookies



SERVIÇOS REST

- REpresentational State Transfer – REST é um modelo utilizado para projetar arquiteturas de software distribuído, que se baseia na comunicação via rede.
- É um conjunto de princípios descrito por Roy Fielding, um dos criadores do protocolo HTTP.
- O modelo REST é amplamente utilizado na evolução da arquitetura do protocolo HTTP e também é utilizado para a implementação de Web Services.
- É importante aplicar corretamente os princípios do REST para se beneficiar com a arquitetura e padrões da Web.



FUNDAMENTOS WEB

- O modelo REST funciona através de uma arquitetura cliente-servidor, em que o cliente envia requisições para o servidor e o servidor retorna as respostas correspondentes.
- As requisições e respostas são trocadas através de uma interface uniforme, que segue os princípios do REST.
- A interface é composta por quatro elementos principais: identificação dos recursos, manipulação dos recursos através de representações, mensagens autodescritivas e hipermídia como mecanismo de estado da aplicação.
- O REST utiliza métodos HTTP, como GET, POST, PUT e DELETE, para manipular os recursos identificados por URLs.
- As representações dos recursos são trocadas em formato de dados, como JSON ou XML.
- A utilização desses elementos permite que as aplicações sejam escaláveis, flexíveis e facilmente integradas, além de serem aderentes aos padrões da Web.

EXEMPLO DE APLICAÇÃO REST:

- Uma aplicação que permite que os usuários visualizem, criem, atualizem ou excluam informações sobre produtos em um catálogo online.
- O servidor armazena as informações dos produtos e expõe esses recursos através de URLs, como "/produtos".
- Um cliente pode enviar uma requisição GET para a URL "/produtos" para obter uma lista de todos os produtos disponíveis. Ele também pode enviar uma requisição POST para a mesma URL para criar um novo produto no catálogo.
- Caso o cliente quiser atualizar ou excluir um produto existente, ele pode enviar uma requisição PUT ou DELETE para a URL correspondente a esse produto.
- As respostas do servidor podem ser em formato JSON ou XML, que representam as informações dos produtos.
- O uso dos métodos HTTP, das URLs e das representações dos recursos permite que a aplicação siga os princípios do modelo REST e possa ser facilmente integrada com outras aplicações que utilizam a mesma arquitetura.

DIFERENÇAS ENTRE OS TERMOS:

- **REST** é um modelo arquitetural que define os princípios e restrições para projetar sistemas distribuídos baseados em comunicação via rede.
  - É um modelo abstrato que define os princípios e restrições que devem ser seguidos para se projetar um sistema distribuído baseado em comunicação via rede.

- **REST API** (Application Programming Interface) é uma implementação específica do modelo REST, que utiliza as suas características e restrições para expor uma interface de programação para acessar recursos de um sistema.
  - É uma aplicação concreta que implementa esses princípios e restrições para expor recursos de um sistema de forma padronizada, permitindo que outros sistemas possam interagir com esses recursos de maneira simples e eficiente.

- **Conclusão**: REST API é uma implementação concreta dos princípios do modelo REST, que é utilizado para expor uma interface de programação para acesso aos recursos de um sistema.

Universidade Cruzeiro do Sul

CREDITOS DO MATERIAL:

Elaborado por:
Prof. Vinicius Heltai

Colaboração de Conteúdo:
Sem colaborador

Ultima atualização: 2023/1

PROF. VINICIUS HELTAI

vhpacheco@cruzeirodosul.edu.br

https://www.facebook.com/vheltai

https://www.instagram.com/vheltai