INTRODUÇÃO AO PROTOCOLO HTTP

Ultimamente diversas das aplicações migraram da modalidade Desktop e vieram a ser Web, o que isso nos diz? Nos diz que nos tempos atuais a internet se tornou o método padrão de comunicação, e essa popularização fez com que as aplicações migrassem para WebSites, Aplicações Mobile e outros. Para melhor entendimento do funcionamento dessas aplicações devemos conhecer o principal protocolo que rege a comunicação na Wide Word Web (WWW) .

HYPERTEXT TRANSFER PROTOCOL (HTTP)

O protocolo HTTP é um conjunto de regras que permite a transferência de mensagens entre os diversos sistemas na internet, hoje está na versão 2.0 mas a maioria da internet é criada na versão 1.1. No HTTP é utilizado a arquitetura Cliente-Servidor, o que significa isso? O Cliente pode ser um navegador ou linha de comando, este realiza um pedido de processamento de informações (**REQUEST)** e o servidor responde enviando os dados processados, chamamos isso de **RESPONSE**. Em nossas **request's** passamos um cabeçalho neste conta um métodos ou verbo, um corpo com as informaçõs que queremos processar e outros campos. Já nas **responses** é informado no cabeçalho um status code, um corpo contendo as respostas já processadas. Durante as **requests e responses** manipulamos informações que são descritas como recursos (**resource)**. Um recurso é tudo aquilo que pode ser representado e gerenciado pela WWW, seja imagem, vídeo, arquivos de texto ou binários.

Sabe-se que o protocolo HTTP é utilizado para transferir informações, mas como temos acesso a estas? Dentro de um servidor definimos um endereço que é composto por um método, domínio, caminho e um identificador de um recurso. Exemplo: **"**http://127.0.0.1/api/pessoas**",** como identificar cada um dos componentes de um endereço?

- método: http.
- dominio: 127.0.0.1
- caminho: /api.
- identificador de um recurso: /pessoas.

O domínio é composto por quatro campos numéricos, isso dificulta muito o entendimento e aprendizado não ? Para isso existe o Sistema de Nomeação de Domínios ou Domain Name System (DNS), que tem como responsabilidade tornar os domínios humanamente lembraveis.

Outro ponto importante é quando se trata de manipulação de informações sensíveis elas sempre são transportadas por toda a rede em texto, sem nenhuma confidencialidade ou segurança? Não, para isso é utilizado os protocolos SSL/TSL Transport Layer Security, através de um sistema de certificados digitais que criam sockets seguros, onde as informações são transportadas criptografadas. Estas informações podem ser descriptografadas no meio da rota de transporte? Não existem dois métodos de criptografia os de chave simétrica e assimétrica, qual a diferença ? A chave simétrica é exatamente igual uma fica no cliente e outra no servidor com acesso a esta chave facilmente pode-se ter acesso às informações criptografadas. O método de chave assimétrica é o contrário

tem-se uma chave no cliente e outra diferente no servidor desta forma a informação é transportada de maneira mais segura. Mas qual método é utilizado no HTTPS? ambos, na primeira requisição é criada uma chave simétrica que é utilizada para as próximas requisições.

O servidor quando responde as requisições deve informar um status code coerente com o estado da requisição solicitada pelo Cliente, existem diversos casos e para melhor responder os códigos são categorizados da seguinte forma:

1. 1XX - Informativos: servem para informar.
2. 2XX - Sucesso: servem para informar que as requisições tiverem sucesso.
3. 3XX - Redirecionamento: Servem para redirecionar o cliente para algum novo endereço, pode ser utilizado em diversos casos.
4. 4XX - Error no Cliente: Servem para informar que a requisição do cliente contém algum erro.
5. 5XX - Error no Servidor: Servem para informar que a requisição obteve um erro de processamento no servidor.

cabe ao servidor informar o código mais assertivo a situação, de forma a melhorar a experiência do cliente.

COMO UTILIZAR O HTTP NAS APLICAÇÕES WEB?

Como dito anteriormente sabemos que no HTTP temos um cliente e um servidor, para melhor entendimento vamos imaginar o site da magazine luiza quando utilizamos o navegador ele será o cliente, que fará as requisições para o servidor através do endereço. Para melhor compreensão vamos imaginar que queremos cadastrar um novo produto no site e depois visualizar a lista de produtos disponíveis, quando formos montar nossa **request** teremos que escolher o método ou verbo HTTP, para isso devemos conhecer quais estão disponíveis. Dentro do HTTP os métodos mais utilizados são os que permitem executar a criação, listagem (leitura), atualização e remoção de um recurso, ou seja, que permitam executar uma CRUD. Estes métodos são:

- POST: É utilizado para criar recursos em bases de dados, quando a requisição que utiliza este método é executada com sucesso é devolvido o status code **201, Created**.
- GET: É utilizado para listar, ler, obter e consultar recursos, quando uma requisição que utiliza este método é executada com sucesso é devolvido o status code **200, Ok**.
- PUT: É utilizado para alterar, atualizar as informações de um recurso, quando uma requisição que utiliza este método é executada com sucesso é devolvido o status code **200, Ok**.
- DELETE: É utilizado para deletar um recurso da base de dados, quando uma requisição que utiliza este método é executada com sucesso é devolvido o status code **204, No content**.

Voltando ao caso das operações de cadastro de produtos, deve-se escolher o método **POST,** pois é através dele que iremos criar este recurso em nossa base de dados. Ao definir o método surge a dúvida como enviaremos as informações? No método **POST** as informações são trafegadas no corpo da requisição HTTP. Já no caso da listagem iremos optar pelo método **GET**, a diferença é que a passagem de valores é feita via URL, para informar o início da passagem de atributos deve-se utilizar o operador “?” o nome do

atributo em seguida adicionar o operador “=” e o valor que iremos utilizar como parâmetro. Veja o exemplo: “**www.magazineluiza.com.br/produtos?title=fone-de-ouvidos-xiomi**”.

HTTP 1.1 VS HTTP 2.0

As principais diferenças entre as versões do protocolo estão na eficiência e segurança, enquanto a versão 1.1 temos que implementar compressão e criptografia, a versão 2.0 já contém esses recursos nativos, ou seja, o desenvolvedor do servidor não deve se preocupar em como utilizá-lo. Uma das vantagens da versão 2.0 é o recurso de SERVER PUSH, que ao ler uma requisição deste documento ele é capaz de processar suas dependências e fazer empurrá-las sem a necessidade de requests do cliente. Observer a tabela abaixo listando as diferenças entre as duas versões:

Recurso	HTTP 1.1	HTTP 2.0
CRIPTOGRAFIA EM CABEÇALHOS	X	V
GZIP NA RESPONSE	X	V
BINÁRIO + HPACK	X	V
TSL DEFAULT	X	V
SERVER PUSH	X	V
CABEÇALHO STATEFULL	X	V