Universidade Federal do Ceará

Campus Quixadá
Desenvolvimento de Software para Web

Trabalho 01

Aluno(a): Marianna de Pinho Severo Matrícula: 374856
Professor(a): Júlio Serafim

]

Quixadá, fevereiro de 2019

1) **Com suas palavras, escreva sobre o desenvolvimento histórico da WEB.**

Quando a Internet começou a ser utilizada, a WEB consistia em páginas cheias de textos e tabelas. Esse modelo restringia a utilização dos serviços apenas por pessoas com maior familiaridade com os formatos apresentados e não atraía a atenção de muitos usuários comuns, como acontece atualmente.

Entretanto, em 1993, um novo *browser* chamado Mosaic foi lançado. Ele possuía uma interface muito mais amigável e já apresentava mudanças no conceito de construção de páginas WEB compostas por imagens e texto. Ele foi um grande impulsionador para o desenvolvimento da WEB, atraindo atenções para o grande potencial que ela possui para aplicações em diversas áreas, como comércio, telecomunicações e muitas outras que observamos hoje.

Em 1989, um pesquisador do CERN, chamado Tim Berners-Lee, teve a ideia de criar um sistema computacional que facilitasse a colaboração entre pesquisadores de todo o mundo e, dessa ideia, nasceu o HTML.

Ao longo dos anos, os avanços das tecnologias de *hardware* e telecomunicações forneceram aos desenvolvedores WEB a infraestrutura necessária para modificar e aperfeiçoar a forma como constroem a WEB. Ao mesmo tempo, melhorias das interfaces, dos recursos e dos serviços fornecidos aos usuários pelos softwares WEB fizeram com que, cada vez mais, novos usuários fossem atraídos, impulsionando a melhoria das tecnologias de telecomunicações e *hardware*. Além disso, muitas empresas - como Google, Amazon e Yahoo - surgiram e cresceram, inclusive em momentos de crise econômica.

Embora os recursos e conceitos para o desenvolvimento WEB tenham sofrido várias mudanças ao longo dos anos, eles conservam os princípios básicos estabelecidos no início.

2) **Com suas palavras, explique a arquitetura cliente-servidor.**

A arquitetura cliente-servidor é um modelo de comunicação em redes de computadores em que uma das máquinas (Cliente) envolvidas solicita um serviço a uma outra máquina (Servidor), a qual realiza o serviço solicitado e retorna o resultado. A essas solicitações damos o nome de requisições e esse modo de comunicação é chamado de Requisição-Resposta. É sempre o Cliente quem manda as requisições para o servidor, mas o contrário não acontece. Entretanto, é importante destacar que uma mesma máquina pode ser tanto servidor de uma máquina e cliente de outra, ao mesmo tempo. Os serviços solicitados podem ser de diferentes naturezas, como processamento dos dados enviados, obtenção de informações a partir de uma base de dados, comunicação com outras máquinas, armazenamento de informações, entre outras. Além disso, o Cliente pode fornecer, ou não, uma interface direta de comunicação com o usuário. Um exemplo de sistema que utiliza a arquitetura cliente-servidor é o Youtube.

3) **Com suas palavras, defina os verbos HTTP: GET, PUT, POST, DELETE.**

Os verbos HTTP são utilizados para especificar ações a serem aplicadas em registros localizados em determinadas URLs. Essas ações são sempre acompanhadas de um código de resposta enviado pelo servidor requisitado, indicando o estado da operação solicitada.

O verbo GET é usado para obter um registro ou uma lista deles. Assim, ao ser enviada ao servidor, recebe como resposta os registros solicitados. Na Figura 01, podemos observar um exemplo de utilização desse verbo. Está sendo realizada a requisição do arquivo hello.htm, localizado em www.tutorialspoint.com/hello.htm.

```
GET /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

Figura 01: Exemplo verbo GET
Fonte: https://www.tutorialspoint.com/http/http_requests.htm

O verbo POST, por outro lado, é usado para a criação de um novo registro na URL especificada. O conteúdo desse novo registro é passado no corpo da requisição. Na Figura 02, podemos observar um exemplo de sua utilização.

```
POST /cgi-bin/process.cgi HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Content-Type: application/x-www-form-urlencoded
Content-Length: length
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

licenseID=string&content=string&/paramsXML=string
```

Figura 02: Exemplo verbo POST
Fonte: https://www.tutorialspoint.com/http/http_requests.htm

O verbo PUT é utilizado para atualizar o conteúdo de um registro já existente em determinada URL. Na Figura 03, podemos observar um exemplo de sua utilização.

```
PUT /user/1234567890 HTTP/1.1
Host: http://sookocheff.com

{
        "name": "Kevin Sookocheff",
        "website": "http://kevinsookocheff.com"
}
```

Figura 03: Exemplo verbo PUT

Fonte: https://sookocheff.com/post/api/when-to-use-http-put-and-http-post/

Por último, o verbo DELETE é usado para excluir um determinado registro localizado em uma URL. Na Figura 04, podemos observar um exemplo de seu emprego.

```
DELETE /users/123 HTTP/1.1
Host: www.example.org
If-Match: "q1w2e3r4t5"
```

Figura 04: Exemplo verbo DELETE
Fonte: http://amundsen.com/examples/put-delete-forms/

4) Diferencie Front-End e Back-End.

O Front-End está relacionado à interface com o usuário, ou seja, à composição dos objetos e textos apresentados aos usuários, em seus diversos formatos e organizações. Ele está, portanto, relacionado à parte do software que o usuário pode ver e interagir.

O Back-End, por outro lado, está relacionado à parte do software que "dá vida" ao que é apresentado no Front-End. Portanto, quando o usuário interage com algum objeto apresentado na interface, é o Back-End o responsável por realizar o serviço representado por aquele objeto. Ele é a parte que o usuário não vê, mas que faz o software realizar o que se propõe a fazer.

5) Cite alguns dos principais *frameworks* de Front-End e Back-End.

Os *frameworks* front-end são ferramentas que fornecem uma série de facilidades para essa parte do desenvolvimento, como bibliotecas e pré-processadores. Segundo uma pesquisa realizada pela empresa KeyCDN, utilizando o sistema de estrelas do GitHub para construir um *Rank*, os três *frameworks* front-end mais utilizados no mundo atualmente são o Bootstrap, o Foundation e o Materialize.

O Bootstrap é o *framework* mais utilizado atualmente. Ele é uma ferramenta open-source, que fornece suporte para HTML, CSS e JavaScript. Ele emprega o pré-processador SASS e fornece vários plug-ins construídos em JQuery.

O Foundation é o segundo *framework* mais utilizado. Ele também usa o pré-processador SASS, possui integração com JavaScript, é destaque na construção de códigos concisos e enxutos e permite a construção de interfaces que se adaptam para qualquer tipo de dispositivos.

Em terceiro lugar está o Materialize, que é um *framework* baseado no Material Design para emprego das cores, ícones e formatos. Ele utiliza bastante a JQuery e apresenta um sistema de *grids* semelhante ao do Bootstrap. Além disso, ele possui elementos exclusivos para dispositivos móveis e é considerado mais leve e mais agradável do que muitos outros *frameworks*.

Com respeito aos *frameworks* back-end, três são bastante utilizados: Django, Ruby on Rails e Lavarel.

O Django é um *framework* para desenvolvimento de aplicações WEB em Python. Ele é open-source e se baseia na arquitetura de software MVT. Ele permite a configuração de permissões de usuários nas aplicações desenvolvidas e é bastante empregado em

softwares orientados a dados. O Django possui uma documentação bem escrita e uma ampla comunidade de usuários.

O Ruby on Rails é um *framework* para desenvolvimento na linguagem Ruby e feito para executar em Linux. Ele emprega a arquitetura de software MVC e fornece um ambiente de fácil compilação e testes. Além disso, ele fornece funções relacionadas à segurança, possui uma grande quantidade de plug-ins de terceiros que auxiliam no desenvolvimento e possui uma ampla comunidade de usuários.

Por sua vez, o Lavarel é um *framework open-source* para PHP. Ele também segue o padrão arquitetural MVC e é equipado com eficientes ferramentas de linha de comando e funções de ajuda. Ele possui uma ampla comunidade de usuários e também fornece serviços pagos.