DESENV. WEB EM HTML5, CSS, JAVASCRIPT E PHP – ARA0062

Prof. Kayo Monteiro

Ementa

- Linguagem de marcação de hypertexto - html;
- Linguagem de marcação e estilos - css;
- Linguagem javascript;
- Programação de páginas dinâmicas com ajax & php;
- Criação de um crud;

Objetivos

- Estruturar páginas web, utilizando a linguagem DE MARCAÇÃO DE HYPERTEXTO (**HTML**), para a formação de um arcabouço sobre o qual serão construídas funcionalidades dinâmicas;
- Aplicar características de estilo a páginas WEB, utilizando a linguagem de marcação de estilos (**CSS**), para praticar técnicas de engenharia de software como facilidade de compreensão, reutilização de código, manutenibilidade e interoperabilidade;
- Empregar programabilidade em páginas web, utilizando linguagem **Javascript**, mais usada no mercado, para o desenvolvimento de um sistema web com funcionalidades dinâmicas;
- Empregar programabilidade em páginas web, utilizando linguagem **PHP**, bastante comum em sistemas legado, para o desenvolvimento de um sistema web com funcionalidades dinâmicas;

Avaliação

O processo de avaliação oficial se dá através de **NOTA FINAL ÚNICA**, estabelecida ao fim do semestre. Os procedimentos de avaliação contemplarão competências desenvolvidas durante a disciplina no âmbito presencial.

UNIFAVIP
wyden

Comunicação

Grupo de Whatsapp e/ou email:

UNIFAVIP wyden

Arquitetura Web

Funcionamento da Web

Na Web 1.0 (Web-Panfleto) poucas empresas e anunciantes produziam conteúdo para os usuários.

A Web 2.0 envolve arquiteturas de participação – projeto que encoraja a interação do usuário e as contribuições da comunidade.

Empresas da Web 2.0 dependem quase inteiramente da inteligência coletiva colaborativa.

Funcionamento da Web 1.0

- Web Estática: A Web 1.0 consistia principalmente em páginas estáticas, onde o conteúdo era apresentado de forma fixa e não interativa.
- Buscadores: Os mecanismos de busca, como o Google e o Yahoo, surgiram na Web 1.0, permitindo aos usuários encontrar informações na Internet.
- Conexão discada: A maioria dos usuários acessavam a Web por meio de conexões discadas, o que limitava a velocidade e a experiência online.

Funcionamento da Web

Web 2.0

- **Interação e colaboração:** A Web 2.0 trouxe uma maior interatividade e participação dos usuários, por meio de redes sociais, blogs e wikis.
- **Redes Sociais:** Plataformas como Facebook, Twitter e LinkedIn ganharam popularidade.
- **Conteúdo Gerado pelo Usuário:** Os usuários passaram a criar e compartilhar conteúdo, como vídeos no YouTube e fotos no Instagram.

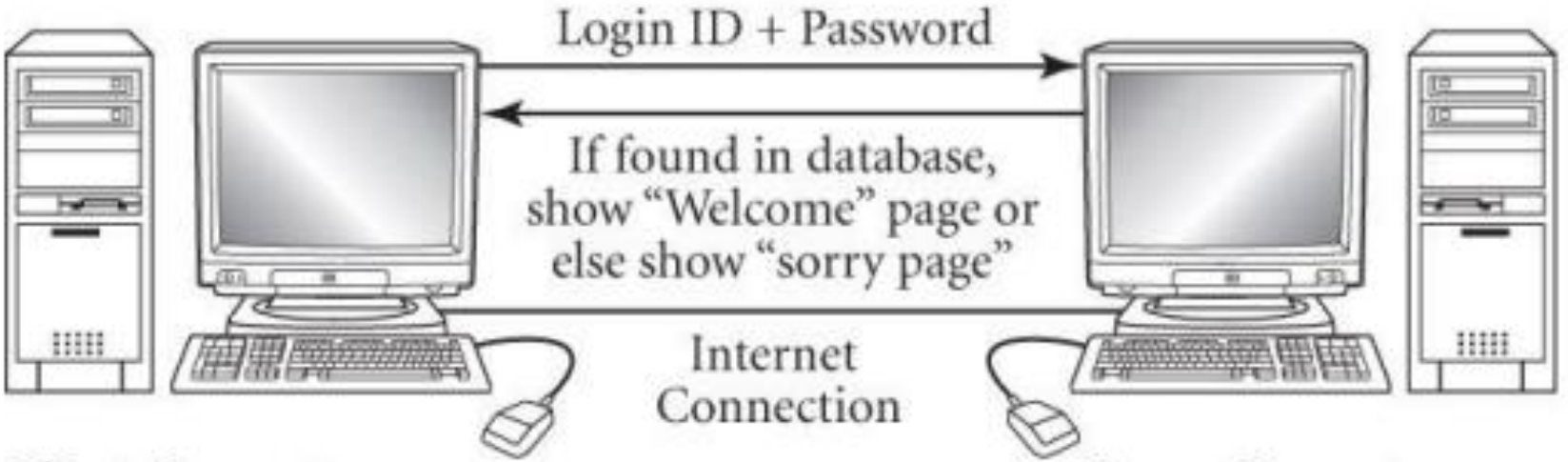
Web 3.0

- **Web Semântica:** A Web 3.0 é marcada pela capacidade de máquinas entenderem o conteúdo, por meio de metadados e informações estruturadas, tornando a busca mais inteligente e eficiente.
- **Internet das Coisas (IoT):** A Web 3.0 inclui a integração de dispositivos inteligentes à Internet.
- **Realidade Virtual e Aumentada**: Tecnologias de RV e RA se tornaram mais comuns na Web 3.0, proporcionando experiências imersivas e interativas aos usuários.

Funcionamento da Web

- **Inteligência Artificial Avançada:** A Web 4.0 pode apresentar uma IA mais sofisticada, com capacidades de aprendizado profundo e tomada de decisões complexas.
- **Integração Mente-Máquina:** Pode haver avanços na interface cérebro-computador, permitindo a comunicação direta entre o cérebro humano e a Internet.
- **Personalização Extrema:** A Web 4.0 poderia oferecer uma experiência altamente personalizada, adaptando-se às necessidades e preferências individuais dos usuários.
- **Internet Quântica:** A computação quântica poderia impulsionar a Web 4.0, resolvendo problemas computacionais complexos e impulsionando a segurança na rede.

Arquitetura Cliente - Servidor



Arquitetura Cliente - Servidor

- A arquitetura cliente-servidor é um modelo amplamente utilizado na construção de projetos web e sistemas distribuídos. Nesse modelo, a comunicação ocorre entre duas entidades principais: o cliente e o servidor. Cada um desempenha um papel específico no processo de troca de informações e execução de tarefas.
- A arquitetura cliente-servidor é um paradigma de comunicação em que um cliente (geralmente um navegador web) solicita recursos ou serviços a um servidor (geralmente um computador remoto). O servidor processa a solicitação e retorna os dados ou resultados necessários para o cliente.

Arquitetura Cliente - Servidor [Componentes]

Cliente: É a interface de usuário que interage com o usuário final. No contexto web, o cliente é geralmente um navegador web (como Chrome, Firefox, Safari, etc.), que exibe páginas HTML, executa scripts e apresenta a interface gráfica para o usuário.

Servidor: É uma máquina remota que hospeda os recursos e serviços. No contexto web, o servidor executa o software de servidor web (como Apache, Nginx, etc.) e é responsável por receber as solicitações dos clientes, processá-las e enviar as respostas adequadas.

Arquitetura Cliente - Servidor [Fluxo]

- O cliente envia uma solicitação para o servidor. Essa solicitação pode ser uma requisição de uma página da web, dados de um formulário, recursos como imagens, ou até mesmo uma ação específica, como submeter um formulário.
- O servidor recebe a solicitação, processa-a e realiza as operações necessárias para atender à solicitação do cliente.
- O servidor, então, envia a resposta de volta ao cliente. Essa resposta pode conter a página HTML solicitada, informações de erro, ou dados para serem exibidos na interface.

Arquitetura Cliente - Servidor [Resumo]

a arquitetura cliente-servidor é uma base fundamental para o desenvolvimento de projetos web, permitindo que os clientes acessem e interajam com recursos e serviços hospedados em servidores remotos. Com esse modelo, é possível criar aplicativos web poderosos e escaláveis que atendam às necessidades dos usuários de forma eficiente e confiável.

Arquitetura Cliente - Servidor [Comunicação]

- A comunicação se dá por requisição e resposta:
  - Cliente e servidor conhecem o protocolo HTTP
  - A troca de informações se dá graças a esse protocolo
- A resposta em geral é em HTML (formato de apresentação), mas a resposta também pode ser uma imagem, .pdf, .php ou o 404 Not Found
- A resposta possui ainda um cabeçalho com informações do protocolo HTTP

Protocolo HTTP

Protocolo HTTP

O Protocolo de Transferência de Hipertexto (HTTP) desempenha um papel fundamental na arquitetura cliente-servidor, especialmente em projetos web. Ele é o protocolo de comunicação mais utilizado na Internet e possibilita a transferência de informações entre o cliente (geralmente um navegador web) e o servidor de forma padronizada. A importância do HTTP na arquitetura cliente-servidor inclui os seguintes aspectos:

Protocolo HTTP

- **Comunicação Cliente-Servidor:** O HTTP define as regras e a estrutura para que o cliente e o servidor possam se comunicar. Quando um cliente faz uma solicitação para acessar uma página web ou recursos específicos, ele utiliza o HTTP para enviar essa requisição ao servidor.

- **Requisição e Resposta:** O HTTP define o formato das mensagens de requisição enviadas pelos clientes e das respostas enviadas pelos servidores. Essa estrutura padronizada permite que os clientes e servidores se entendam mutuamente e troquem informações de maneira confiável.

Protocolo HTTP

- **Métodos de Requisição:** O HTTP fornece diversos métodos de requisição, como GET, POST, PUT, DELETE, entre outros. Cada método tem um propósito específico, permitindo que o cliente solicite diferentes tipos de ações ao servidor, como obter informações, enviar dados ou atualizar recursos.
- **URLs:** O HTTP utiliza URLs para identificar os recursos na Web. As URLs fornecem a localização dos recursos no servidor, permitindo que o cliente especifique quais recursos ele deseja acessar.
- **Stateless:** O HTTP é um protocolo stateless, o que significa que cada requisição é independente e não mantém informações de estado entre as solicitações. Isso simplifica a implementação e escalabilidade dos servidores, tornando o sistema mais robusto e eficiente.

Protocolo HTTP

- **Cookies e Sessões:** Embora o HTTP seja stateless, ele permite o uso de mecanismos adicionais, como cookies e sessões, para manter o estado entre as solicitações do cliente. Esses recursos são essenciais para fornecer uma experiência personalizada ao usuário e permitir a autenticação em sites.
- **Segurança**: O HTTP é a base para a maioria das comunicações seguras da Web, onde o HTTPS (HTTP Seguro) é usado. O HTTPS adiciona uma camada de segurança criptografada à comunicação, protegendo as informações confidenciais dos usuários durante a transmissão.

Protocolo HTTP

Os métodos GET e POST são dois dos principais métodos de requisição definidos pelo Protocolo de Transferência de Hipertexto (HTTP). Eles são usados em aplicações web para enviar informações do cliente (geralmente um navegador) para o servidor ou para solicitar recursos do servidor. Vamos entender como cada um deles funciona:

Protocolo HTTP - GET

O método GET é usado para solicitar recursos do servidor. Quando um cliente faz uma requisição GET, ele está solicitando o conteúdo de uma determinada URL. As solicitações GET são frequentemente usadas para buscar informações e recuperar recursos que não causam alterações no estado do servidor.

Protocolo HTTP – GET

Principais características do método GET:

- **Parâmetros na URL:** Os parâmetros são enviados como parte da URL após o ponto de interrogação (?). Por exemplo: https://www.exemplo.com/pagina?parametro1=valor1&parametro2=valor2
- **Limitação de tamanho:** O tamanho da URL é limitado e, em alguns navegadores e servidores, há um limite para a quantidade de dados que podem ser enviados por meio do método GET.
- **Caching**: As solicitações GET podem ser armazenadas em cache pelo navegador, permitindo que o cliente acesse novamente a mesma URL sem precisar fazer uma nova requisição ao servidor.

Protocolo HTTP - POST

O método POST é usado para enviar dados do cliente para o servidor. Ao contrário do método GET, os dados enviados pelo método POST são incluídos no corpo da requisição, não na URL. O método POST é amplamente utilizado quando os dados enviados podem causar alterações no estado do servidor, como ao enviar formulários, fazer login, ou executar outras ações que requerem envio de dados sensíveis.

Protocolo HTTP - POST

Principais características do método POST:

- **Dados no corpo da requisição**: Os dados são enviados no corpo da requisição HTTP, tornando-os menos visíveis na URL e, potencialmente, mais seguros.
- **Não há limite de tamanho**: Como os dados são enviados no corpo, o método POST não possui a limitação de tamanho da URL, permitindo o envio de grandes quantidades de dados.
- **Não é armazenado em cache**: As solicitações POST não são armazenadas em cache pelo navegador, pois envolvem alterações no servidor e devem ser reenviadas a cada nova requisição.

Protocolo HTTP - GET e POST

Quando usar GET e POST:

- Use o método GET quando você estiver solicitando recursos do servidor, como carregar uma página HTML ou buscar informações.
- Use o método POST quando você estiver enviando dados para o servidor, como enviar formulários, fazer login, ou executar ações que alteram o estado do servidor.

Comunicação

Grupo de Whatsapp e/ou email:

UNIFAVIP wyden