

# [IF977] Engenharia de Software

---

Prof. Vinicius Cardoso Garcia  
[vcg@cin.ufpe.br](mailto:vcg@cin.ufpe.br) :: [@vinicius3w](https://twitter.com/vinicius3w) :: [assertlab.com](http://assertlab.com)



## Licença do material

---

Este Trabalho foi licenciado com uma Licença

**Creative Commons - Atribuição-NãoComercial-  
Compartilhualgual 3.0 Não Adaptada.**

Mais informações visite

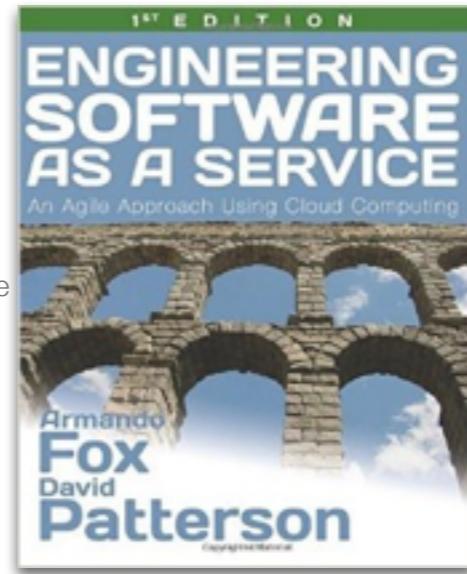
<http://creativecommons.org/licenses/by-nc-sa/3.0/deed.pt>



## Referências

---

- A biblioteca do Desenvolvedor de Software dos dias de hoje
  - <http://bit.ly/TDOA5L>
- SWEBOK
  - Guide to the Software Engineering Body of Knowledge (SWEBOK): <http://www.computer.org/web/swebok>
- Engineering Software as a Service: An Agile Approach Using Cloud Computing (Beta Edition)
  - <http://www.saasbook.info/>



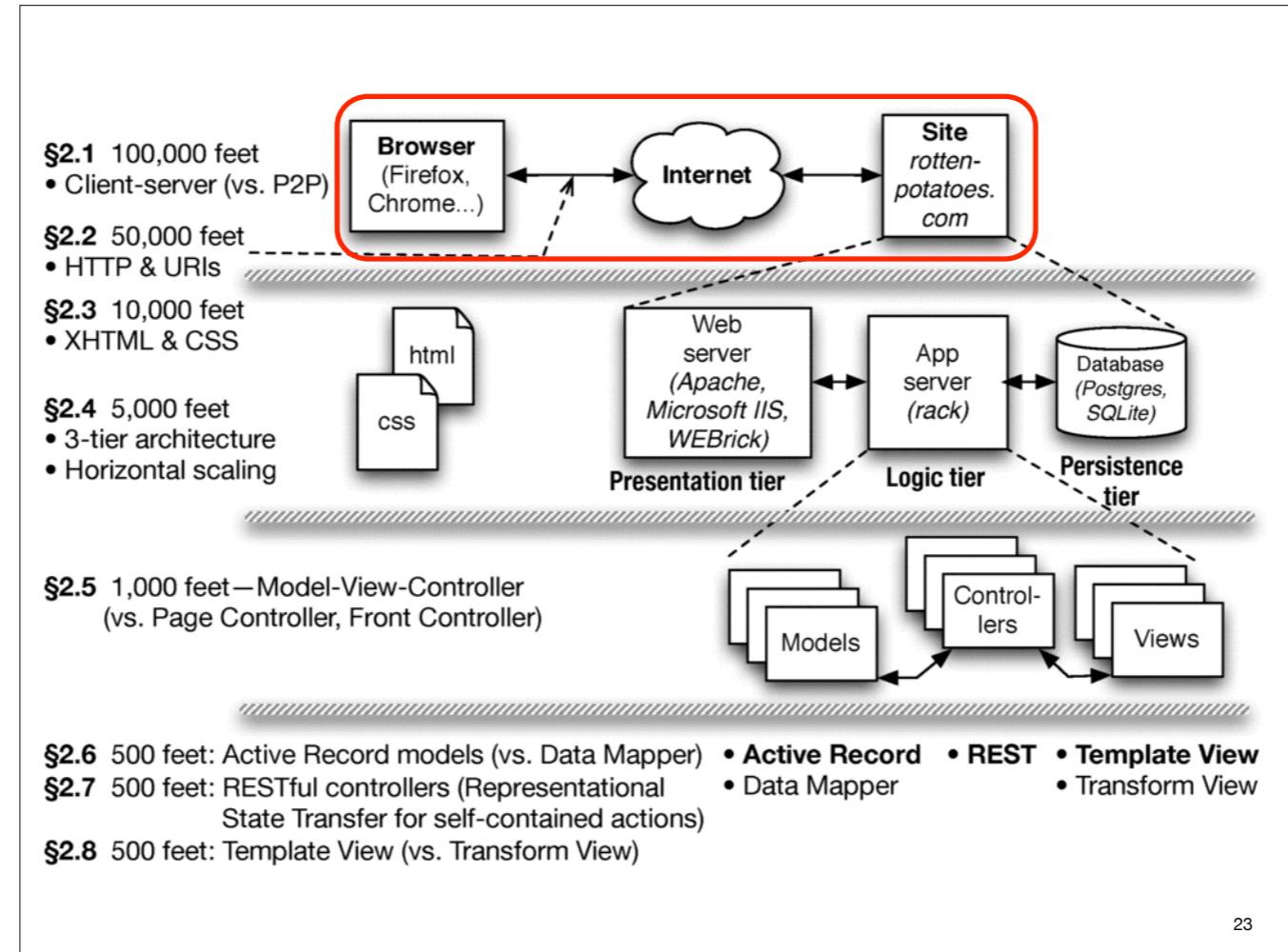
# Outline

---

- (ESaaS 8.1) Intro to RSpec & Unit Tests
- (ESaaS 8.2) FIRST, TDD, and Getting Started With RSpec
- (ESaaS 2.1-2.2) The Web as a Client-Server System; TCP/IP intro
- (ESaaS 2.3) HTML + CSS
- (ESaaS 2.4) 3-tier shared-nothing architecture & scaling

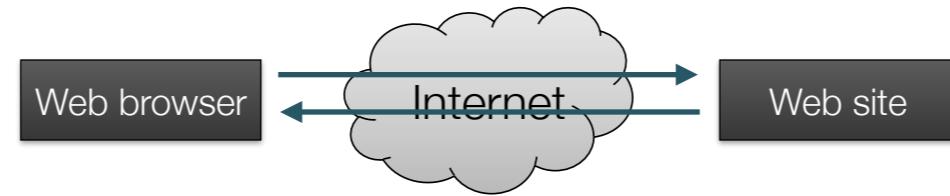


## A Web como um sistema cliente-servidor; Introdução ao TCP/IP

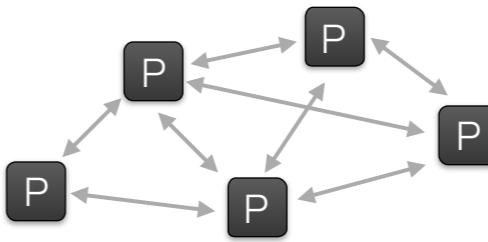
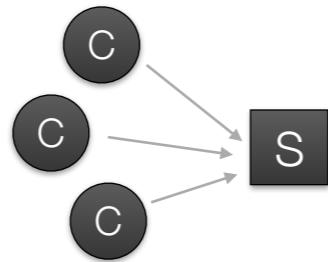


## Web a 100.000 metros

- A Web é uma arquitetura cliente/servidor
- É fundamentalmente uma estrutura orientada a requisição e resposta



## Cliente-Servidor vs. Peer-to-Peer

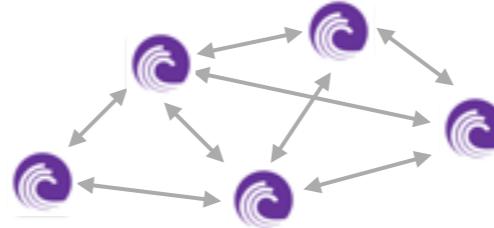


- Cliente e Servidor são especializados em suas tarefas
  - Cliente: fazer perguntas em nome do usuário
  - Servidor: esperar e responder às perguntas, serve a muitos clientes
- Cliente/Servidor é um padrão arquitetural!
  - Padrões de projeto capturam soluções estruturais comuns para problemas recorrentes!

## Cliente-Servidor vs. Peer-to-Peer



S



- Cliente e Servidor são especializados em suas tarefas
  - Cliente: fazer perguntas em nome do usuário
  - Servidor: esperar e responder às perguntas, serve a muitos clientes
- Cliente/Servidor é um padrão arquitetural!
  - Padrões de projeto capturam soluções estruturais comuns para problemas recorrentes!

# Protocolos TCP/IP

- Um endereço IP (Internet Protocol) identifica uma interface física de rede com quatro octetos, ex. **128.32.244.172**
  - O endereço **127.0.0.1** é “este computador”, chamado de **localhost**, se não estiver conectado na internet!
- TCP/IP (Transmission Control Protocol/Internet Protocol)
  - IP: sem garantias, serviço de melhor esforço que oferece pacotes a partir de um endereço IP para outro
  - TCP: torna o IP confiável através da detecção de pacotes “perdidos”, dados que chegam fora de ordem, erros de transmissão, redes lentas, etc, e respondem de forma apropriada
  - Portas TCP permitem múltiplas apps no mesmo computador
- Vint Cerf & Bob Kahn: 2004 Turing Award for Internet architecture & protocols, incl. TCP/IP

GET /bears/



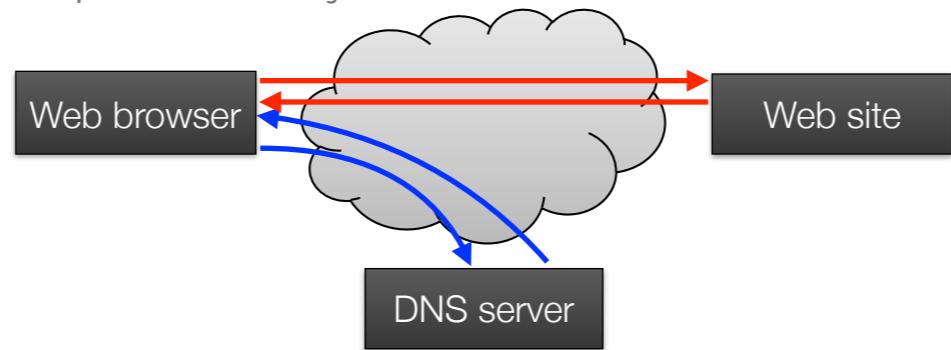
HTTP/0.9 200 OK



Berkeley Unix was first open-source reference implementation of TCP/IP in an open-source OS!

## Web a 100.000 metros

- A Web é uma arquitetura cliente/servidor
- É fundamentalmente uma estrutura orientada a requisição e resposta
- Domain Name System (DNS) é outro serviço que mapeia nomes para endereços IP



# Hypertext Transfer Protocol

- Protocolo requisição/resposta baseado em ASCII para transmissão de informação na Web
- Requisição HTTP inclui
  - Método de requisição (GET, POST, etc)
  - Uniform Resource Identifier (URI)
  - Versão do protocolo HTTP entendido pelo cliente
  - Headers – informações extra sobre a requisição
- Resposta HTTP inclui
  - Protocol version & Status Code =>
  - Cabeçalho da resposta
  - Corpo da resposta

## HTTP status codes:

**2xx — *all is well***  
**3xx — *resource moved***  
**4xx — *access problem***  
**5xx — *server error***

29

DEMO: nc -l 8000, then GET /la/di/da

DEMO: telnet www.cs.berkeley.edu 80

GET /bears/ HTTP/1.0

## Cookies

---

- Observação: HTTP é **stateless**
- Problemas da Web 1.0: como guiar o usuário “através” de um fluxo de páginas?
  - Usar endereço IP para identificar retorno de usuários?
    - Computadores públicos, usuário compartilham IP
    - Embarcar identificação de usuários nas URI de busca?
      - Quebra o uso de cache
  - Rapidamente substituído por cookies
    - Rails gerencia as violações evidentes dos cookies para você

30

Demo: see cookie arriving from Google.com  
curl -D - http://www.google.com -o /dev/null

## Pergunta

---

\_\_\_\_\_ pode criar e modificar os cookies; \_\_\_\_\_ é responsável por incluir o cookie correto com cada requisição.

- A. Browser; App SaaS
- B. App SaaS; browser
- C. Requisição HTTP; browser
- D. App SaaS; resposta HTTP

## Pergunta

---

\_\_\_\_\_ pode criar e modificar os cookies; \_\_\_\_\_ é responsável por incluir o cookie correto com cada requisição.

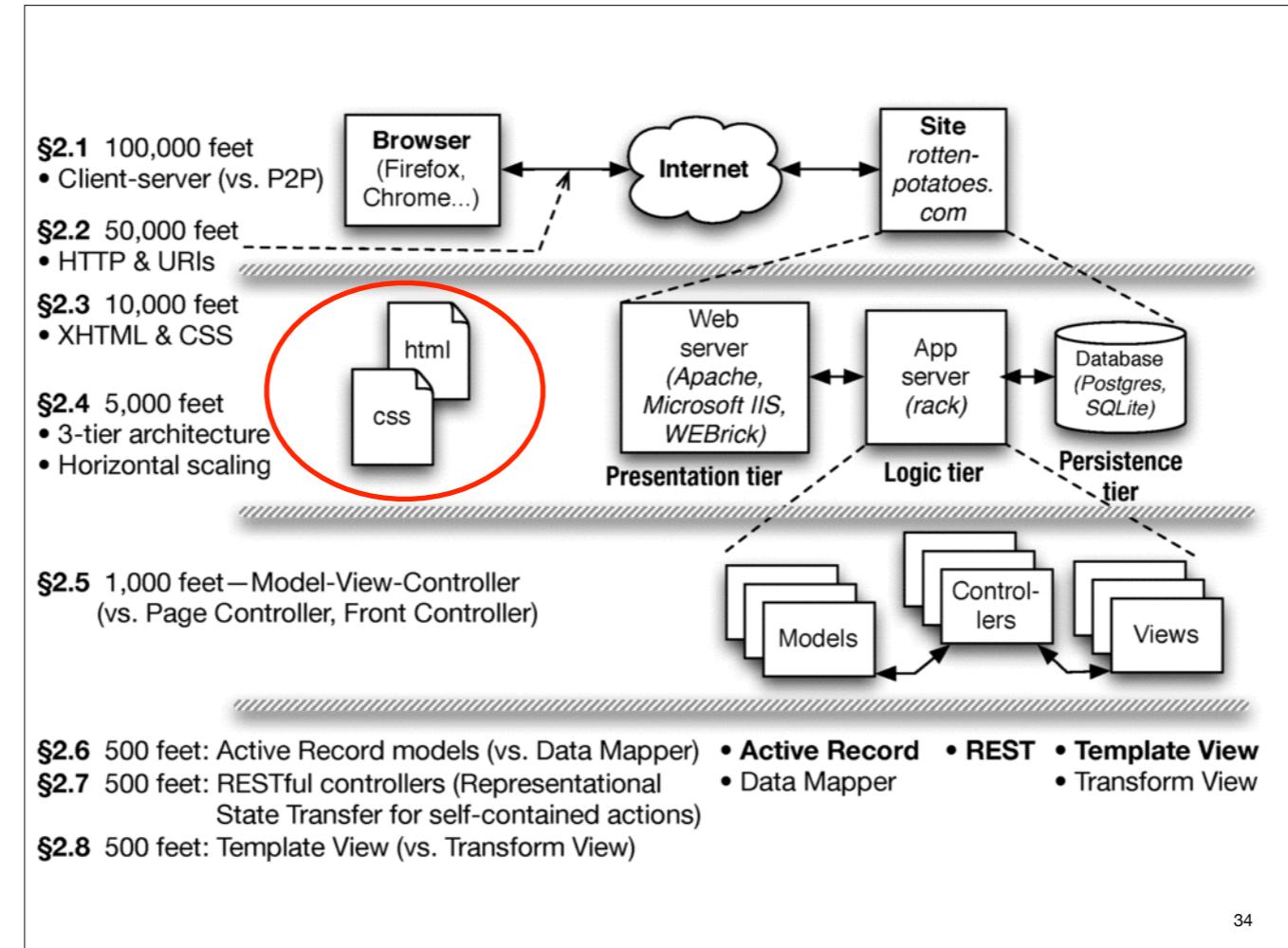
- A. Browser; App SaaS
-  B. App SaaS; browser
- C. Requisição HTTP; browser
- D. App SaaS; resposta HTTP

# HTML & CSS



```
</form>
top:0px;
border
<div>
position:absolute;
color:#999;
margin:0px; </h2> </html>
</span>
text-align:center; <h2>
<td>
<!DOCTYPE html>
font-style
<head>
<form></div> margin:4px; <ul>
<html> </body><tr><img>
<div><span> <h1><body>
<ul>
```

HTML & CSS



## **Introduction**

This article is a review of the book *Dietary Preferences of Penguins*, by Alice Jones and Bill Smith. Jones and Smith's controversial work makes three hard-to-swallow claims about penguins:

First, that penguins actually prefer tropical foods such as bananas and pineapple to their traditional diet of fish

Second, that tropical foods give penguins an odor that makes them unattractive to their traditional predators

```
<h1>Introduction</h1>
<p>
  This article is a review of the book
  <i>Dietary Preferences of Penguins</i>,
  by Alice Jones and Bill Smith. Jones and Smith's
  controversial work makes three hard-to-swallow claims
  about penguins:
</p>
<ul>
  <li>
    First, that penguins actually prefer tropical foods
    such as bananas and pineapple to their traditional diet
    of fish
  </li>
  <li>
    Second, that tropical foods give penguins an odor that
    makes them unattractive to their traditional predators
  </li>
</ul>
...

```

## Introduction

This article is a review of the book *Dietary Preferences of Penguins*, by Alice Jones and Bill Smith. Jones and Smith's controversial work makes two hard-to-swallow claims about penguins:

- First, that penguins actually prefer tropical foods such as bananas and pineapple to their traditional diet of fish
  - Second, that tropical foods give penguins an odor that makes them unattractive to their traditional predators
- ...

```
<h1>Introduction</h1>
<p>
  This article is a review of the book
  <i>Dietary Preferences of Penguins</i>,
  by Alice Jones and Bill Smith. Jones
  and Smith's controversial work makes
  three hard-to-swallow claims about
  penguins:
  <ul>
    <li>
      First, ...
    </li>
  </ul>
```

# Hypertext Markup Language

---

- Documento = hierarquia de **elementos**
  - inline (headings, tables, lists, paragraphs)
  - embedded (images, JavaScript)
  - forms – permitem a submissão de entradas simples pelo usuário (text, radio/check buttons, dropdown menus...)
- Elementos delimitados por `<tag>....</tag>`
  - Alguns têm **conteúdo**: `<p>Hello world</p>`
  - Alguns têm **atributos**: ``
  - Os atributos `id` e `class` são úteis para **estilo**

## Cascading Style Sheets (CSS)

---

- `<link rel="stylesheet" href="http://..."/>` (no elemento `<head>`): qual stylesheet(s) vai com esta página HTML
- Atributos HTML `id` & `class` são muito importantes no CSS
  - `id` deve ser único na página
  - Uma mesma classe pode ser anexada a muitos elementos

```
<div id="right" class="content">
  <p>
    I'm Vinicius. I teach IF977 and do
    research in the ASSERT Lab and INES Lab.
  </p>
</div>
```

39

Open Web Developer toolbar

Show two audience1st.com pages side by side

Show "Info > display elemet information"

# Seletores CSS identificam elementos específicos para estilo

---

```
<div class="pageFrame" id="pageHead">
  <h1>
    Welcome,
    <span id="custName">Vinicius</span>
  </h1>
  
</div>
```

- tag name: `h1`
- class name: `.pageFrame`
- element ID: `#pageHead`
- tag name & class: `div.pageFrame`
- tag name & id: `img#welcome` (**normalmente redundante**)
- descendant relationship: `div .custName`
- Atributos herdam padrões do browser a menos que sejam substituídos
- Objetivo: **Marcações HTML não contém informações visuais de estilo**

## Pergunta

---

Qual seletor CSS vai selecionar apenas a palavra “bar” para estilo

```
<p class="myClass">foo,  
<span class="myClass">bar<span></p>
```

- A. span.myClass
- B. p .myClass
- C. .myClass span
- D. Todas as anteriores

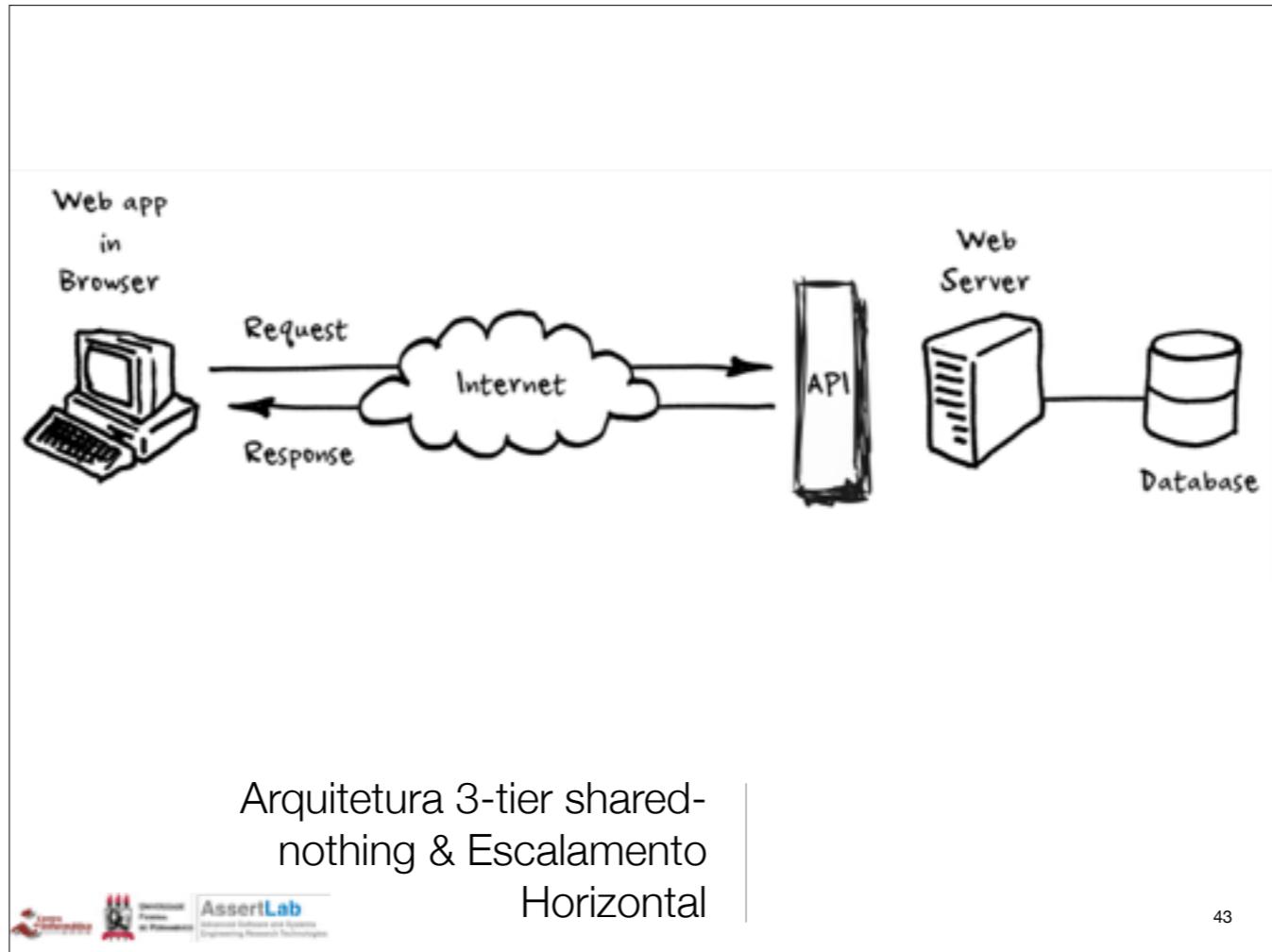
## Pergunta

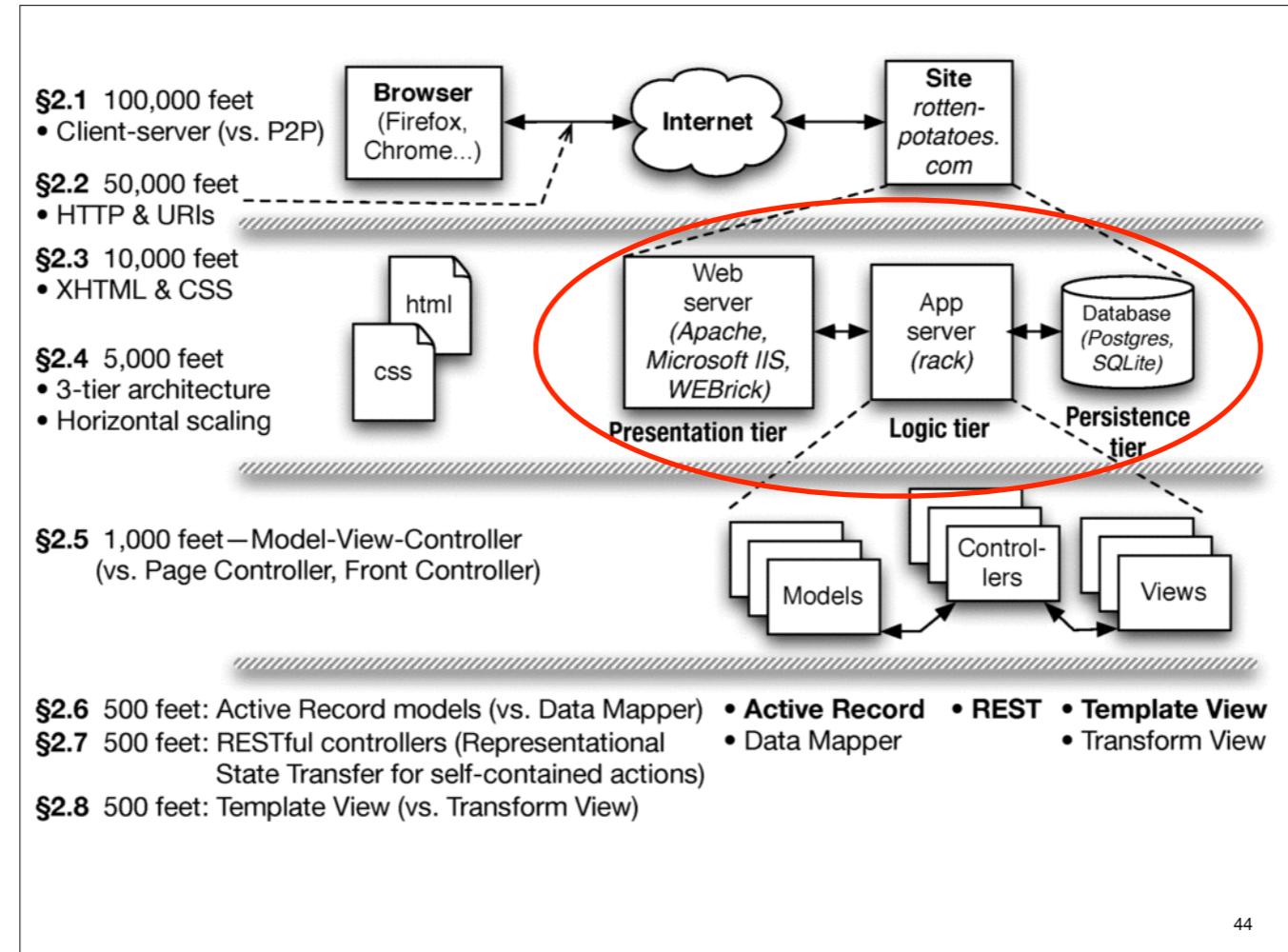
---

Qual seletor CSS vai selecionar apenas a palavra “bar” para estilo

```
<p class="myClass">foo,  
<span class="myClass">bar<span></p>
```

- A. span.myClass
- B. p .myClass
- C. .myClass span
-  D. Todas as anteriores





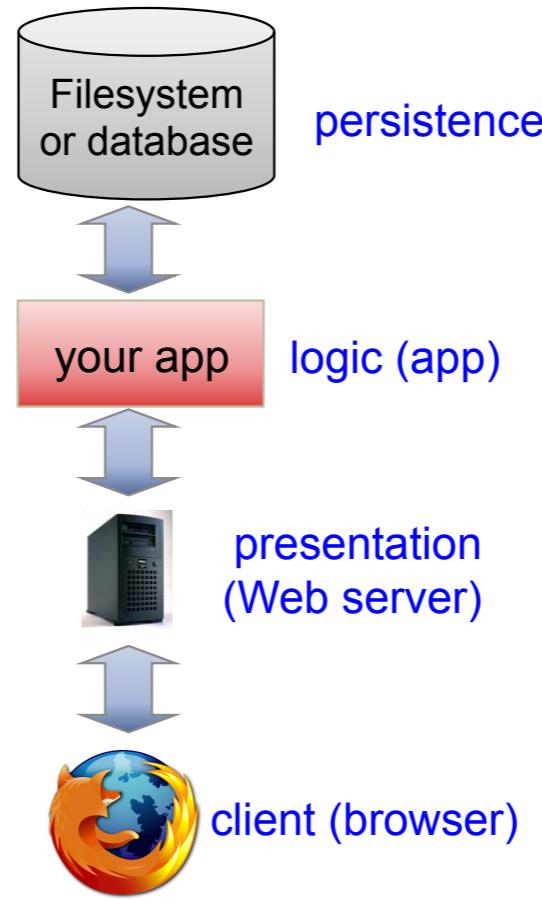
## Geração de conteúdo dinâmico

---

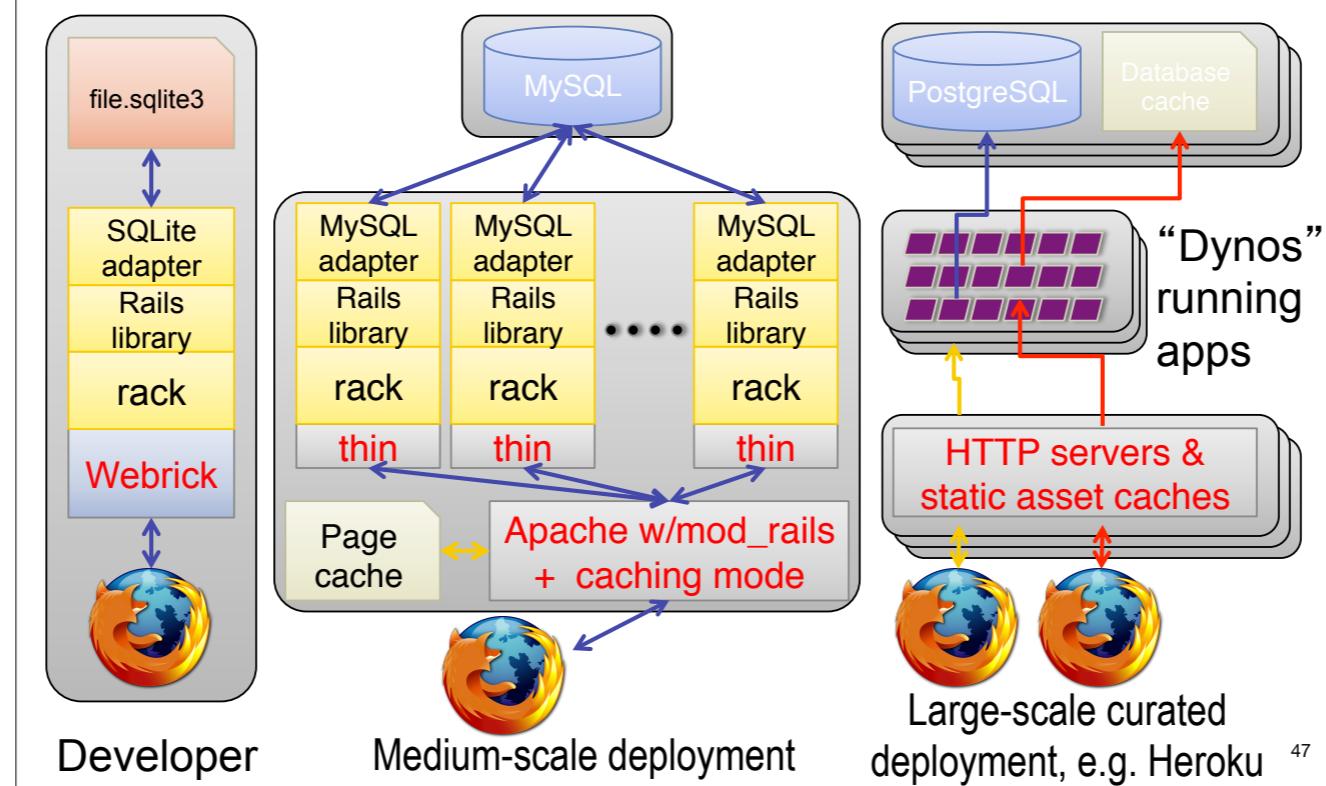
- Antigamente a maioria das páginas da web eram (coleções de) arquivos antigos simples
- Mas os sites mais interessantes da Web1.0/e-commerce, na verdade, executavam um programa para gerar a "página"
- Originalmente: templates com "trechos" de código incorporados
- Eventualmente, o código tornou-se "rabo que abanou o cachorro" e mudou-se para fora do servidor Web

## Sites que são programas (SaaS)

- Como você:
  - “mapeia” a URI para o programa e função correta?
  - Passa argumentos?
  - Evoca programas em um servidor?
  - Trata armazenamento persistente?
  - Trata cookies?
  - Trata erros?
  - Empacota saídas de volta para o usuário?
  - Frameworks dão suporte a estas tarefas comuns



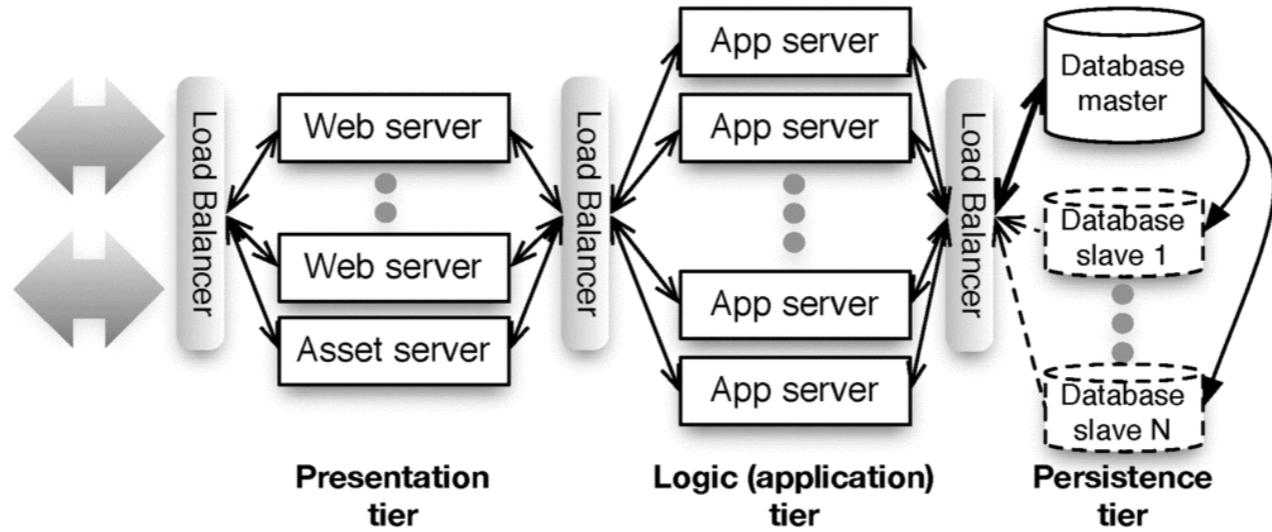
## Ambiente de Desenvolvimento vs. Implantação em Média-Escala



Why small machine for DB?

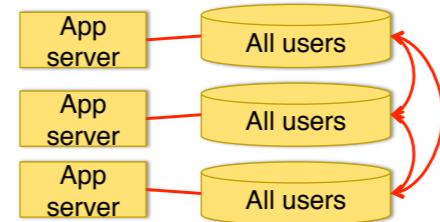
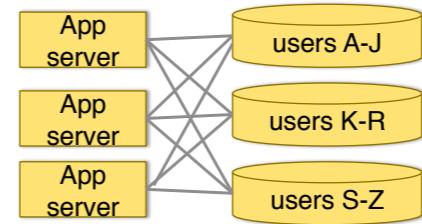
How many processes & how many machines in each example?

## “Shared-nothing”



## Sharding vs. Replication

- Particionamento dos dados através dos “shards”
  - + Escala
  - Ruim quando operações acessam >1 tabelas
  - Ex: user profile
- Replicação em todos os lugares
  - + Consultas multi-table rápidas
  - Difícil de escalar: escritas devem ser propagadas (inconsistência temporária)
  - Ex.: Posts no mural facebook /"likes"



49

MESSAGE: can't magically scale persistence tier. Subject of active R&D, including NoSQL and other stuff we will learn about in 2nd half of class.

## Sumário: Web 1.0 SaaS

---

- Browser requisita recursos (URI) usando HTTP
  - HTTP é um protocolo simples de requisição-resposta que depende do TCP/IP
  - Em SaaS, a maioria das URIs disparam eventos de uma aplicação
- HTML é utilizado para codificar conteúdo, CSS para estilizar o visual
- Cookies permitem ao servidor rastrear os clientes
  - O browser automaticamente envia o cookie para o servidor em cada requisição
  - O servidor pode atualizar o cookie em cada resposta
- Frameworks abstraem tudo isso de maneira conveniente para os desenvolvedores
- ...e ajuda a mapear SaaS para uma arquitetura 3-camadas, shared-nothing

## Pergunta

---

Qual a alternativa correta que representa: (a) camada de apresentação, (b) camada de lógica de negócio, (c) camada de persistência

- A. (a) Apache web server (b) Rack+Rails  
(c) Banco de Dados Relacional
- B. (a) Firefox (b) Apache web server  
(c) PostgreSQL
- C. (a) Microsoft Internet Information Server  
(b) Rack+Rails (c) Apache web server
- D. (a) Firefox (b) Microsoft Internet Information Server (c) MySQL

## Pergunta

---

Qual a alternativa correta que representa: (a) camada de apresentação, (b) camada de lógica de negócio, (c) camada de persistência

- A. (a) Apache web server (b) Rack+Rails  
(c) Banco de Dados Relacional
- B. (a) Firefox (b) Apache web server  
(c) PostgreSQL
- C. (a) Microsoft Internet Information Server  
(b) Rack+Rails (c) Apache web server
- D. (a) Firefox (b) Microsoft Internet Information Server (c) MySQL

