Desenvolvimento WEB

-

Full Stack Completo: Java + React

Prof. Roberto Alencar

Introdução a Sistemas Web

Introdução a Sistemas Web

- Os sistemas Web surgiram em 1990 quando Tim Berners Lee fez a publicação do primeiro servidor Web, primeiro navegador e de um conjunto mínimo de páginas com hipertextos.

- Desde então, a World Wide Web (WWW) se popularizou e está presente na vida diária de bilhões de pessoas do mundo.

- Em agosto de 2022, números publicados pelo portal NetCraft* indicavam a existência de mais 12 milhões de servidores Web servindo mais de 1 bilhão de páginas Web.

- Na área de TI, por consequência, os sistemas Web se tornaram o padrão dominante para o desenvolvimento de aplicações corporativas.

* https://news.netcraft.com

Introdução a Sistemas Web

- Quando estudamos o desenvolvimento de um sistema web, a primeira coisa que pensamos é qual tecnologia irei utilizar.

- Entretanto, independente da tecnologia que seja utilizada (Python, JavaScript, PHP, Java ou outra), o esquema de um sistema Web é similar.

- A isso chamamos de modelo de referência.

Introdução a Sistemas Web

- A figura ao lado apresenta um modelo de referência para sistemas de informação Web.

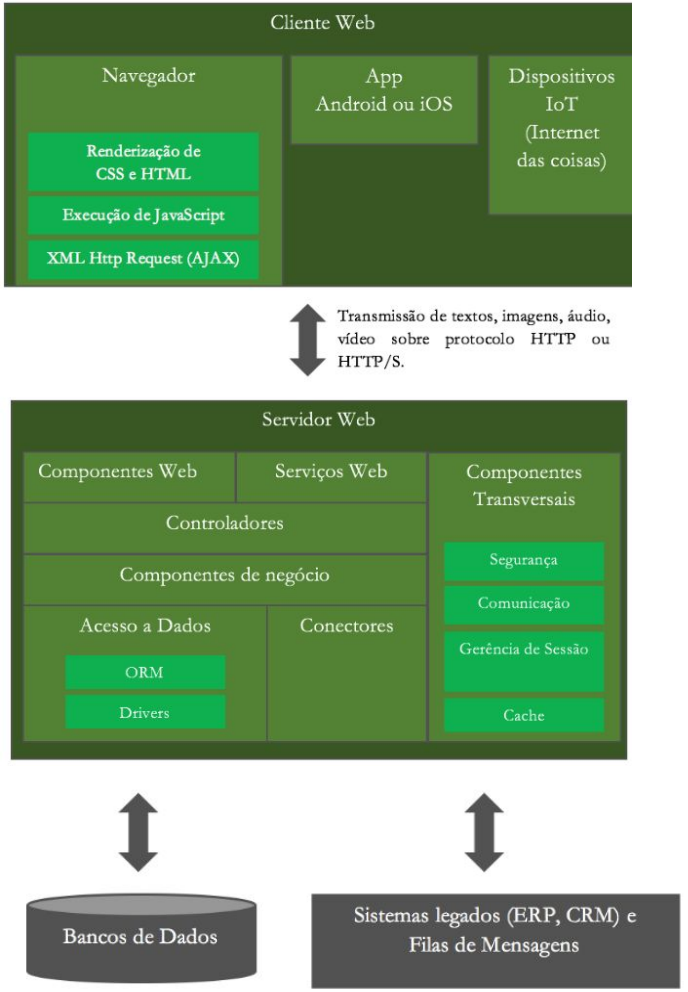


Figura 1: Arquitetura de referência de um sistema Web

O Protocolo HTTP

- O *Hypertext Transfer Protocol* (HTTP) é um protocolo de camada de aplicação para transmissão de documentos hipermídia.
- Foi desenvolvido para comunicação entre navegadores web e servidores web, porém pode ser utilizado para outros propósitos também.
- Segue um modelo cliente-servidor clássico, onde um cliente abre uma conexão, executa uma requisição e espera até receber uma resposta.

O Protocolo HTTP

- O protocolo HTTP é também um protocolo que não guarda estado (*stateless*), isso significa que o servidor não mantém nenhum dado entre duas requisições.

O Protocolo HTTP

- O protocolo HTTP possui diversos métodos, sendo o **GET** o método mais comum nas requisições feitas ao servidor.

- O método GET solicita a representação de um recurso específico. Requisições utilizando o método GET devem retornar apenas dados.

- Outros métodos HTTP comuns:
    - **POST**: Usado para enviar dados para processamento no servidor. É o método mais usado para submeter dados de formulários HTML e pode ser usado para submeter textos e anexos binários diversos;
    - **PUT**: Edita as informações de um determinado recurso Web;
    - **DELETE**: Remove um determinado recurso Web.

A Nomeação de Recursos na Web - URIs

- Um identificador uniforme de recursos, ou **URI**, é usado para identificar objetos e sítios Web.

- Uma URI pode ser um localizador URL, usado para designar um endereço de um sítio Web como por exemplo http://www.google.com.

- Uma URI também pode ser uma URN, usada para designar um objeto dentro de um sítio Web, como por exemplo /imagens/logo.gif.

- Requisições HTTP no cliente usam o esquema de URIs para indicar como acessar recursos no servidor.

Conceitos de Sistemas WEB

- Escolher o **estilo arquitetural** é a primeira e grande decisão realizada pelo time em um projeto.

- Um estilo arquitetural representa uma abordagem de desenho, implementação e distribuição de uma aplicação Web.

- Dentro de arquiteturas Web, existem diversos estilos arquiteturais. Cada um desses estilos é mais apropriado a um certo contexto e um arquiteto deve conseguir enumerá-los, compará-los e recomendá-los conforme o problema em sua mão.

- Não existem estilos melhores ou piores em termos absolutos. **O contexto é fundamental em arquitetura.**

Conceitos de Sistemas WEB

Estilos Arquiteturais Utilizados na Web 1.0

- Este é um estilo legado, que foi dominante na primeira década Web - 1990 a 2000. Ele se caracteriza por formulários simples, com baixa interatividade, uso limitado de JavaScript e componentes visuais simples.

- Ele é ainda usado em aplicações cadastrais de Intranet ou por alguns geradores de código Web criados no começo do século.

- Algumas das tecnologias de mercado que são utilizadas neste estilo incluem:
  - Microsoft ASP
  - Microsoft ASP.NET WebForms
  - Java EE (JSP e Servlets)
  - PHP
  - Ruby
  - Python

Conceitos de Sistemas WEB

Estilos Arquiteturais Utilizados na Web 2.0

- A partir de 2005, o AJAX se popularizou e a quantidade de código de JavaScript começou a aumentar dentro das aplicações Web. Isso trouxe um movimento de enriquecimento de componentes Web e produção de páginas mais ricas e interativas.

- Este é o estilo mais encontrado na produção de portais corporativos e aplicações de e-commerce por toda a Internet. Algumas das tecnologias que são utilizadas neste estilo incluem:
    - Microsoft ASP.NET MVC
    - Java EE JSF (PrimeFaces, RichFaces)
    - PHP (Laravel, Cake)
    - Ruby on Rails
    - Python (Django)

Conceitos de Sistemas WEB

Estilos Arquiteturais Utilizados na Web 2.0

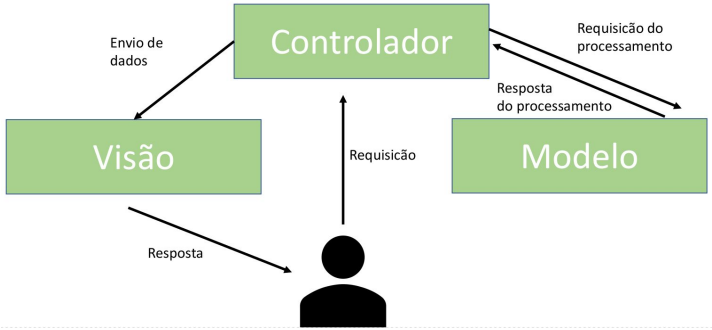
- Essas tecnologias são usadas com bibliotecas JavaScript para prover componentes Web mais ricos e interativos. Um exemplo desse tipo de biblioteca é o jQuery.

- Em termos de organização de camadas, observamos o uso do padrão arquitetural **MVC** com padrão dominante nesta abordagem de desenvolvimento.

Estilo Arquitetural - MVC

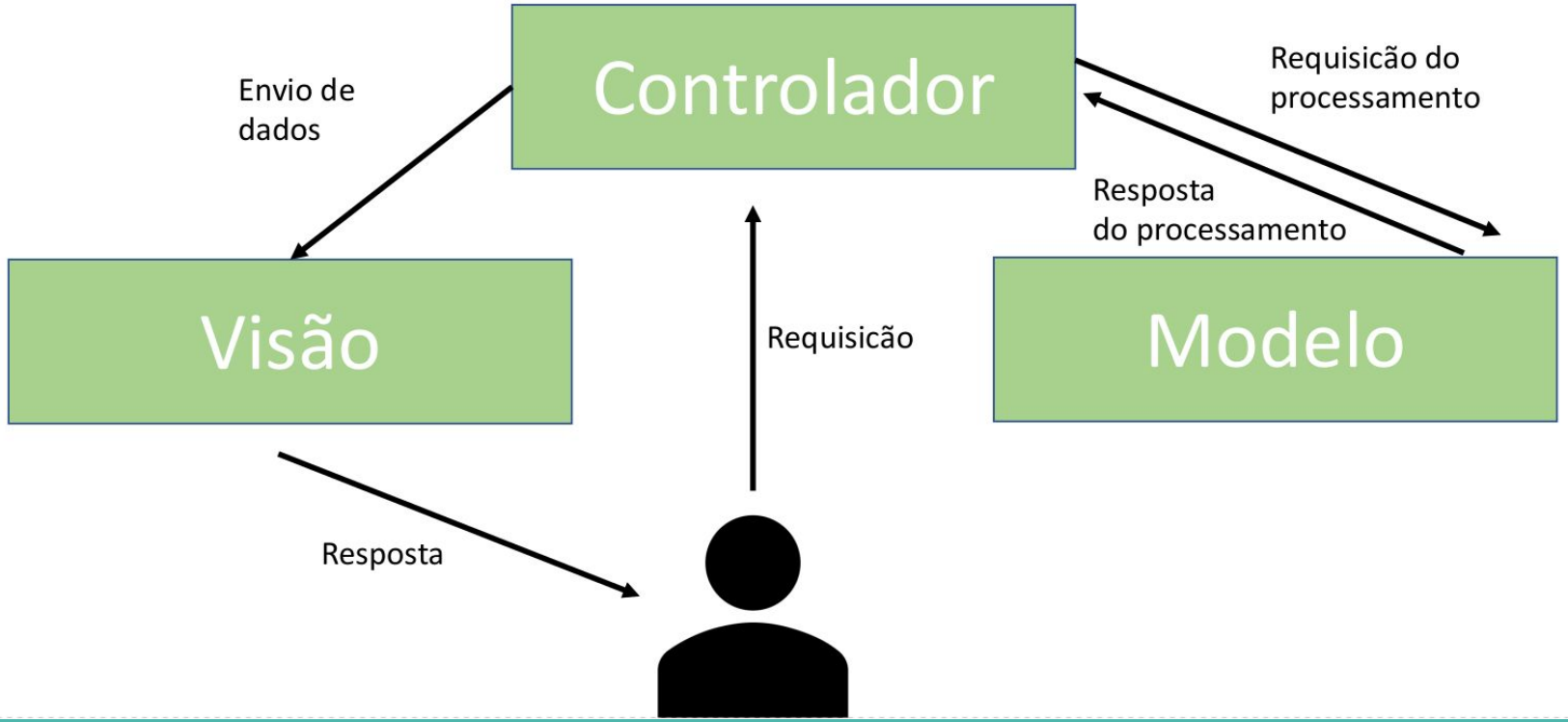
MVC – Model View Controller

- O padrão que provavelmente foi o mais utilizado no desenvolvimento web na última década foi o **Model View Controller (MVC).** O MVC é um estilo utilizado para organizar as camadas lógicas de uma arquitetura de software.



Estilo Arquitetural - MVC

MVC – Model View Controller



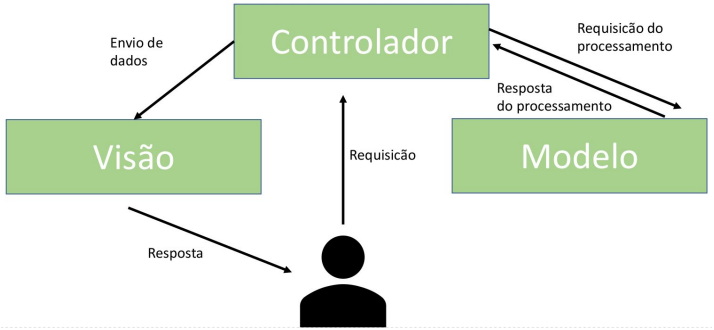
Estilo Arquitetural - MVC

MVC – Model View Controller

- **Modelo** - O componente modelo armazena dados e lógica relacionada.

- Por exemplo, uma requisição ao Modelo possibilita você recuperar as informações do cliente do banco de dados. O Controlador que fez a requisição ao Modelo, pode manipular os dados recebidos e os enviar de volta para o Modelo ou usá-los para renderizar os mesmos enviando estes para a visão.

- **M** representa o modelo de que contém a informação da e os dados.



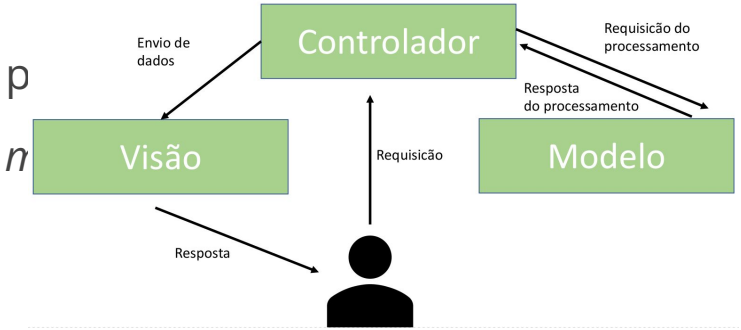
Estilo Arquitetural - MVC

MVC – Model View Controller

- **Visão** - A Visão é a parte do aplicativo que representa a apresentação dos dados ao usuário.

- As visualizações são montadas a partir dos dados coletados do Modelo. Uma visualização solicita (através do *Controller*) que o Modelo dê informações para que ela renderize a saída para o usuário.

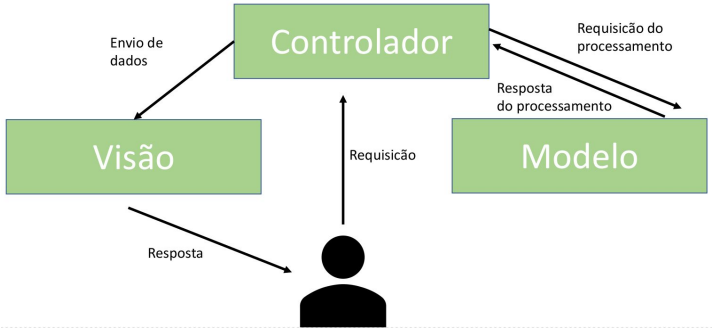
- A *view* é responsável p
será apresentada,
informações do
o usuário.



Estilo Arquitetural - MVC

MVC – Model View Controller

- **Controlador** - É a camada de controle, responsável por ligar o Modelo a Visão, fazendo com que as requisições/dados dos Modelos possam ser repassados para as Visões e vice-versa.



Estilo Arquitetural - MVC

Ordem de chamadas no MVC

1.
   - Usuário invoca algum controlador
   - Controlador valida requisição e seleciona o modelo a ser chamada
2.
   - Modelo executa regras de negócio, prepara e retorna dados para controlador
3.
   - Controlar entrega dados para Visão através de uma projeção do modelo
   - Visão renderiza dados para usuários finais

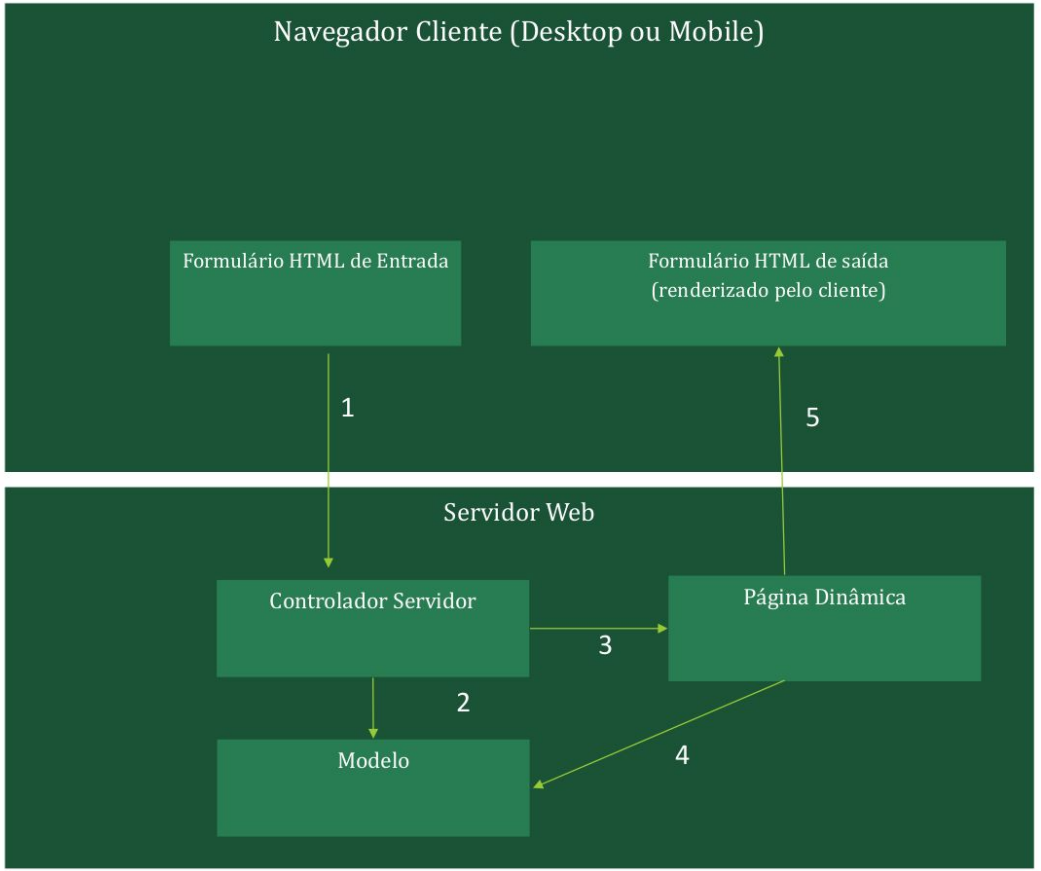
Estilo Arquitetural - MVC

Ordem de chamadas no MVC

V

C

M



SPA (*Single Page Application*)

- A partir de 2010 os Web Designers começaram a trazer ainda mais inovação e complexidade nas páginas Web.

- Essa riqueza trouxe alguns desafios na programação Web cliente em termos de desempenho das suas aplicações. Aspectos que impactam o desempenho em páginas com muita interação incluem:
  - Tempo de carga de uma árvore DOM para renderizar uma página HTML;
  - Grande número de interações entre a camada de visão e a camada de controle, onde cada interação gera uma ou mais requisições HTTP.

SPA (*Single Page Application*)

- Com isso, alguns projetistas Web propuseram algumas modificações no desenho dessas aplicações, que incluem:
    - Movimentar a execução do controlador do servidor para o cliente, reduzindo a quantidade de requisições HTTP entre o cliente e o servidor Web. Essa movimentação é batizada no dialeto como MVVM (*Model View ViewModel*) ou MVP (*Model View Presenter*).

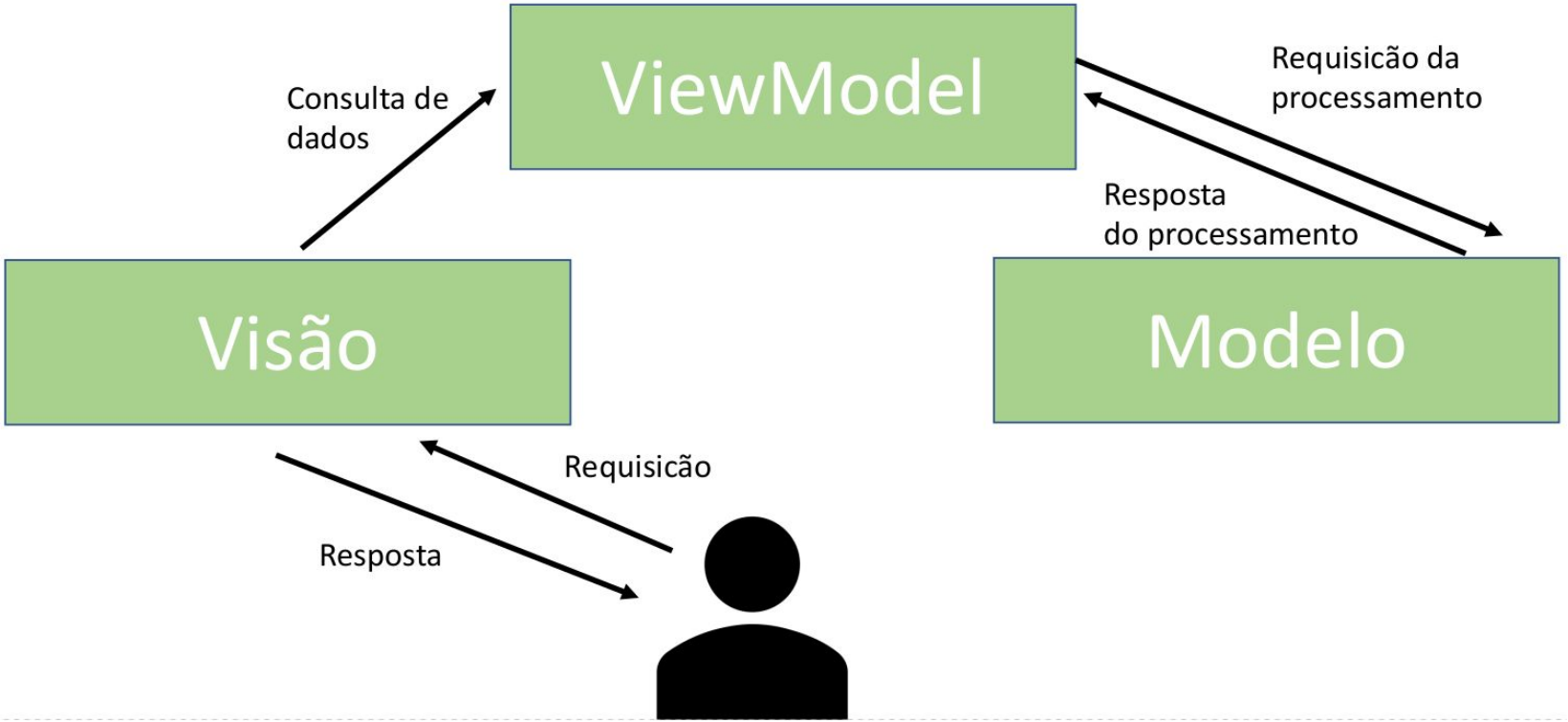
Estilo Arquitetural - MVVM

MVVM – Model View ViewModel

- **MVVM**. É uma evolução do padrão MVC. É utilizado para separar camadas com melhorias de desempenho em aplicações Web e móveis.
- **M** representa o modelo de domínio da aplicação, que contém a informação da lógica de negócio e os dados.
- **V** representa a Visão, que é a parte do aplicativo que representa a apresentação dos dados ao usuário.
- **VM** representa o elemento de mediação e de cache das projeções de dados retornadas pelo modelo.

Estilo Arquitetural - MVVM

MVVM – Model View ViewModel



Estilo Arquitetural - MVVM

Ordem de chamadas no MVVM

1
- O Usuário invoca uma View
- A View aciona o processamento para alguma ViewModel

2
- A ViewModel retorna os dados presentes internamente ou solicita algum processamento de regras e retorno de novos dados para o Model
- O Model faz qualquer tipo de processamento necessário.

3
- A ViewModel disponibiliza os dados para a View que os exibe para o seu usuário.

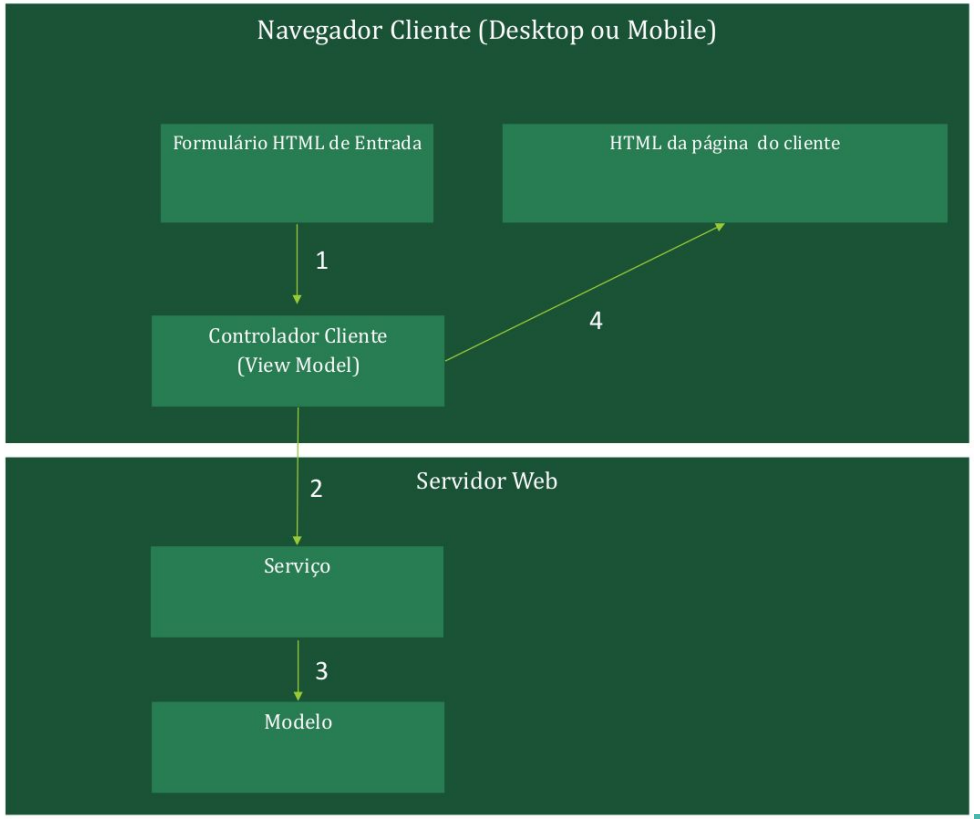
Estilo Arquitetural - MVVM

Ordem de chamadas no MVVM

V

VM

M

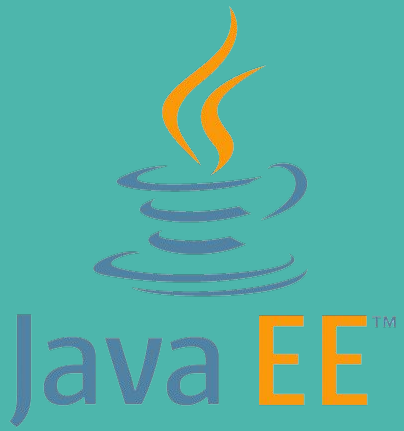


SPA (*Single Page Application*)

- De maneira geral, em uma aplicação **SPA**, o carregamento dos recursos (como CSS, JavaScript e HTML das páginas) ocorre apenas uma única vez: na primeira vez em que o usuário acessa a aplicação. Nesse primeiro acesso, todo o conteúdo HTML, CSS e JavaScript já é transferido para o cliente. A partir deste momento, quando o usuário transitar pelas páginas da aplicação, não será necessário mais fazer requisições para o servidor para a carga dessas novas páginas: o conteúdo relacionado a elas já foi baixado no primeiro acesso.

- Algumas tecnologias que facilitam a implementação deste modelo incluem:
  - Angular JS (Google)
  - React JS (Facebook)
  - Vue JS

Desenvolvimento WEB com Java EE

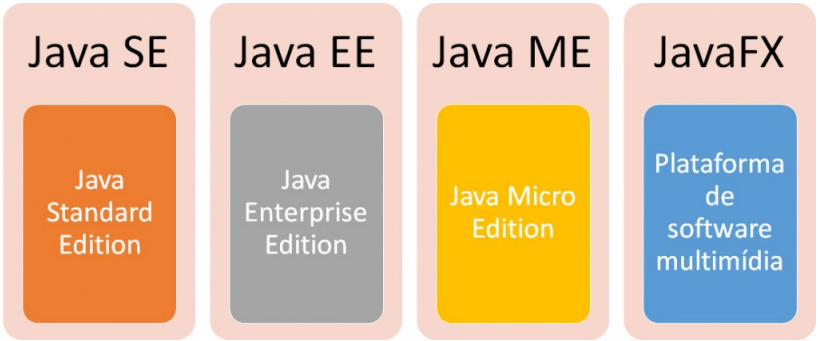


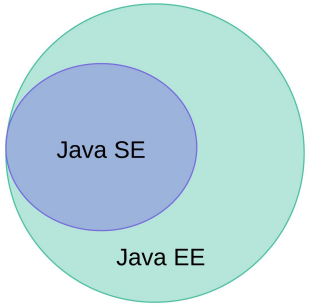
Plataforma Java EE

JEE

- O Java EE (*Java Enterprise Edition*) é uma pacote de tecnologias coordenadas cuja proposta é reduzir o custo e a complexidade do desenvolvimento, implantação e gerenciamento de **aplicações corporativas complexas**.

- É construído **sobre a plataforma Java SE** e oferece um conjunto de APIs para o desenvolvimento e execução de aplicativos portáteis, robustos, escaláveis e seguros no **lado do servidor**.

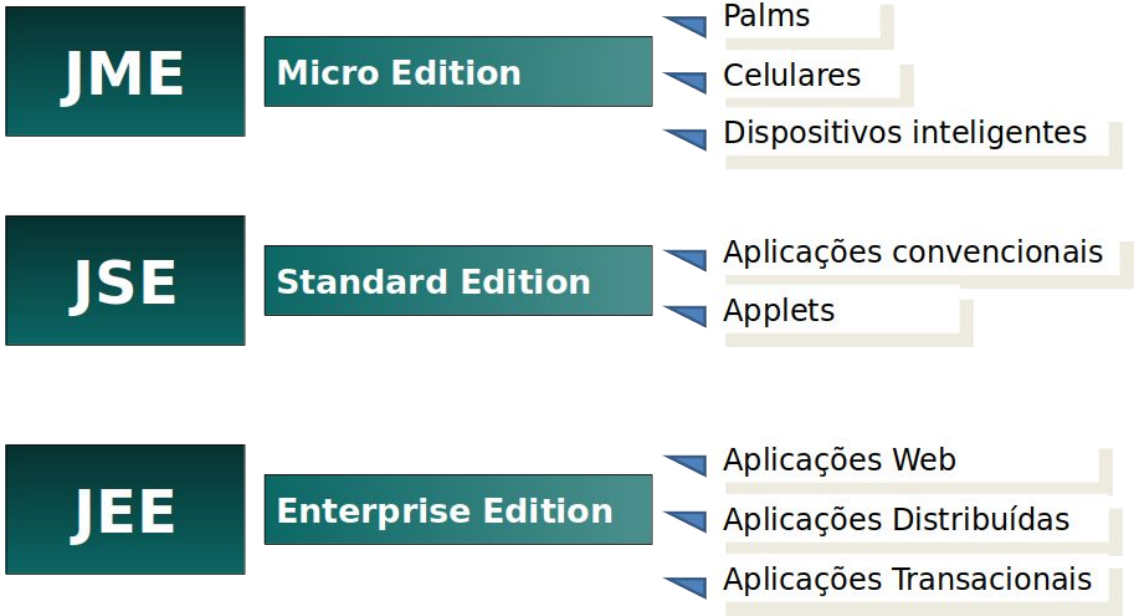




Plataforma Java EE

JEE

- Java EE estende a Java Platform, Standard Edition (Java SE), fornecendo uma API para mapeamento objeto-relacional, arquiteturas multi-camada e distribuídas, web services, entre outras.



Plataforma Java EE

JEE - Principais APIs

- **JDBC** (Java Database Connectivity): utilizado no acesso a bancos de dados;
- **JTA** (Java Transaction API): é uma API que padroniza o tratamento de transações dentro de uma aplicação Java.
- **EJB** (Enterprise Java Beans): utilizados no desenvolvimento de componentes de software. Eles permitem que o programador se concentre nas necessidades do negócio do cliente, enquanto questões de infra-estrutura, segurança, disponibilidade e escalabilidade são responsabilidade do servidor de aplicações.
- **JCA** (Java Connector Architecture): API que padroniza a ligação a aplicações legadas.
- **JPA** (Java Persistence API): API que padroniza o acesso a banco de dados através de mapeamento Objeto/Relacional.
- **JMS** (Java Message Service): API para middleware orientado a mensagens.

Plataforma Java EE

JEE - Principais APIs

- **Servlet:** Classes Java que agem entre a requisição HTTP e o servidor web. Cada Servlet é um objeto Java responsável por receber as requisições do navegador e entregar uma página dinamicamente gerada (HTML).
- **JSP** (JavaServer Pages): Tecnologia para embutir código Java dentro de páginas HTML. Similar a PHP, ASP.
- **Taglibs**: Biblioteca de tags com funções específicas JSTL (JavaServer Pages Standard Tag Library): Taglib padrão da Sun para utilizar for, if-else, manipulação de XML, internacionalização, etc.
- **JSF** (JavaServer Faces): Especificação Java para interface web baseada em componentes. Bibliotecas mais usadas (Primefaces, Richfaces)

Plataforma Java EE

JSF (*Java Server Faces*)

- O JSF é um framework de componentes Web servidor. Em termos práticos, tem por objetivo facilitar a criação de componentes Web ricos, encapsulando eventos e código JavaScript. O JSF é construído sobre a tecnologia de páginas dinâmicas chamada de JSP. Algumas bibliotecas de componentes JSF incluem:
    - PrimeFaces
    - JBOSS RichFaces
    - Apache MyFaces

- Uma página JSF é montada como um arquivo .XHTML que é uma linguagem de marcação de visão chamada de *Facelets*. Os componentes declarados em uma página JSF tem um suporte rico de eventos, que são ligados a classes Java chamada de beans gerenciados (*Managed Beans*). Esses beans são classes Java comuns e permitem com isso fazer o tratamento dos eventos Web em linguagem Java.

Plataforma Java EE

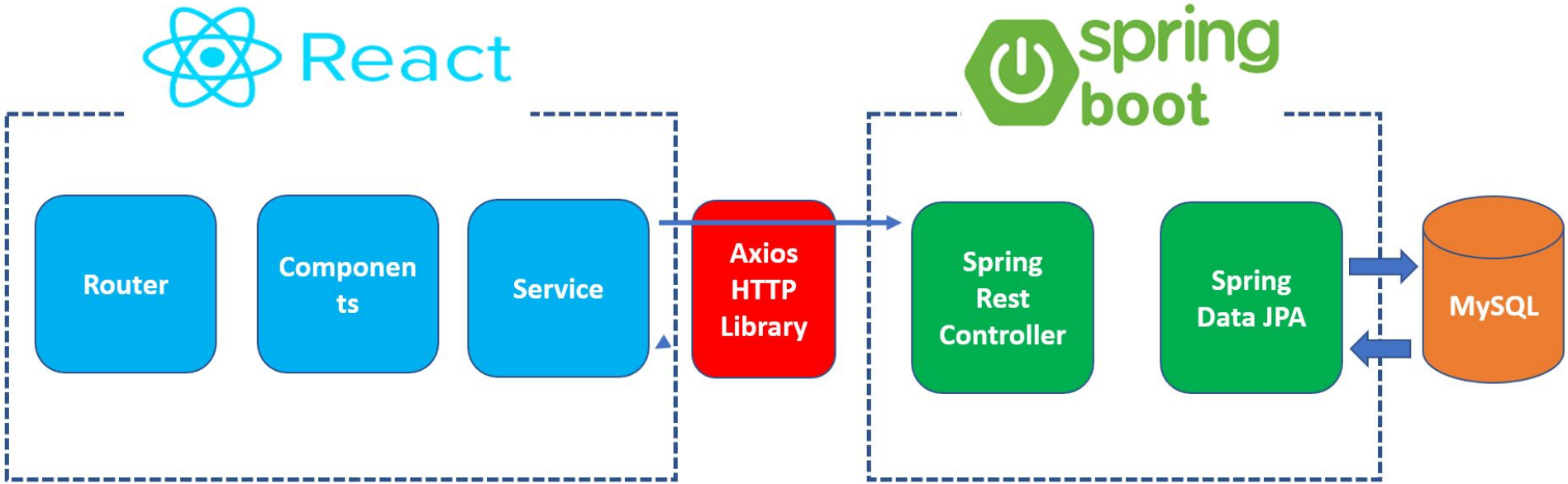
JSF (*Java Server Faces*)

- Na última década, o JSF caiu em desuso no mercado devido às dificuldades com escalabilidade, uso demasiado de recursos de servidor, e outros problemas.

Plataforma Java EE

O que iremos usar neste curso !!!

ReactJS + Spring Boot CRUD Full Stack



Plataforma Java EE

Linguagem de Programação ≠ Frameworks

Plataforma Java EE

Framework

- Em desenvolvimento de software, o framework é aquilo que está na base de um sistema, funcionando como um suporte. Ele permite compartilhar um conjunto de códigos entre aplicações e oferece algum tipo de funcionalidade. Assim, essa ferramenta torna o processo de codificação mais rápido.

- Um Framework é uma espécie de *template* que conta com diversas funcionalidades que podem ser utilizadas pelo desenvolvedor em seus projetos. Ele conta com ferramentas, sistemas, componentes e guias que agilizam o processo de criação de soluções, sendo, portanto, um artifício essencial na vida de um programador.

- É possível aplicar diferentes frameworks para o desenvolvimento de um mesmo produto.

Plataforma Java EE

Exemplos de Frameworks (mundo web)

	JavaScript	PHP	Java	Python	Ruby
Front-end	- Angular - Vue **- React** - Ember				
Back-end	- Express - Node - Adonis	- Laravel - CodeIgniter - Symfony - CakePHP	**- Spring** - JHipster - Grails - JSF - Play - Struts	- Django	- Rails

Plataforma Java EE

Spring Boot

- O Spring Boot é uma ferramenta Open Source que nasceu a partir do Spring framework e veio para facilitar as configurações iniciais de um projeto. O Spring Boot fez decolar a plataforma Spring, pois ele tem tudo o que existe no outro, de forma embutida. Ele já traz um servidor embarcado e todas as configurações iniciais prontas.

spring BOOT

Vantagens de usar Spring Boot

- **Redução drástica no tempo de desenvolvimento**
  - O Spring Boot reduz consideravelmente o tempo de criação e configuração inicial do projeto.

- **Servidor embutido**
  - O Spring Boot já nos fornece um servidor embarcado para subir a nossa aplicação, esse servidor geralmente é o Tomcat. Ele fica responsável por definir como as solicitações de servidor vão ser tratadas, além de servir como um ambiente de servidor HTTP.
  - Além disso, o Spring Boot também permite que seja feita a troca de servidor. Se você quiser usar o Jetty, por exemplo, basta trocar as dependências.

- **Pom.xml organizado**
  - O Spring Boot permite que o nosso arquivo de configuração de dependências (pom.xml) fique bem organizado, pois ao colocar uma dependência no seu projeto, o próprio Spring Boot irá adicionar as sub dependências necessárias automaticamente para você.

Spring Boot vs Spring MVC

- O **Spring MVC** (Model, View, Controller) é uma estrutura montada para tratar requisições HTTP, ou seja, ele “escuta” as requisições feitas na aplicação web, depois ele envia essas requisições para os destinos corretos e só então ele retorna uma resposta.

- Já o Spring Boot, como vimos até aqui, é um framework que facilita as configurações iniciais de um projeto, mas por si só, o **Spring Boot não faz com que o nosso projeto seja web**.

- Portanto, o **Spring Boot e o Spring MVC são frameworks diferentes**. Se você tivesse que criar um projeto web utilizando o Spring Framework, você precisaria configurar o Spring MVC para que o seu projeto se tornasse web. Já com o Spring Boot, você não precisa realizar a configuração do Spring MVC, basta adicionar o módulo Web que você teria as configurações iniciais do Spring MVC.

O que aprendemos nesta aula

- Breve histórico do surgimento dos sistemas web
- Arquitetura de referência para um sistema e informação web
- Conceitos de sistemas web: HTTP e URI
- Estilos Arquiteturais
  - MVC
  - SPA / MVVM
- A plataforma Java EE
- O que é um Framework
- O que é o Spring Boot

Dúvidas

Obrigado !