

# ARQUITETURA CLIENT-SIDE

---

André Luiz Mendes Pereira e Júlio Henrique Araújo Pereira Machado

“

Nenhum software é **100% seguro.**

”

*Daniel Callegari*

# Conheça o livro da disciplina

## CONHEÇA SEUS PROFESSORES 3

*Conheça os professores da disciplina.*

## EMENTA DA DISCIPLINA 4

*Veja a descrição da ementa da disciplina.*

## BIBLIOGRAFIA DA DISCIPLINA 5

*Veja as referências principais de leitura da disciplina.*

## O QUE COMPÕE O MAPA DA AULA? 6

*Confira como funciona o mapa da aula.*

## MAPA DA AULA 7

*Links de artigos científicos, informativos e vídeos sugeridos.*

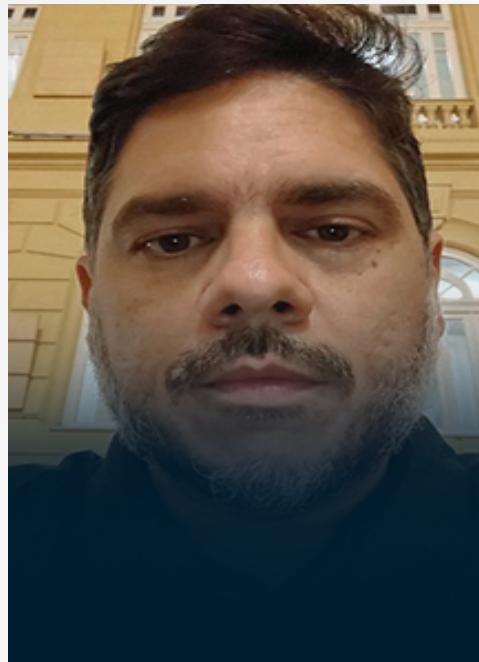
## RESUMO DA DISCIPLINA 33

*Relembre os principais conceitos da disciplina.*

## AVALIAÇÃO 34

*Veja as informações sobre o teste da disciplina.*

# Conheça seus professores



## ANDRÉ LUIZ PEREIRA

Professor Convidado

Fundador da Evolve, uma plataforma digital com propósito de mudar a clusterização do varejo, Andre lidera equipes de desenvolvimento ágeis multidisciplinares em multiplataformas, buscando a essência e resultados focados no business agility. Formado com honras ao mérito em Ciência da Computação, possui pós-graduação em Melhoria no Processos de Software, Arquitetura em Sistemas Distribuído e com MBA em Gestão Empresarial pela FGV. Agrega conhecimento nas diversas verticais na construção de softwares, dentre elas: arquitetura de soluções e de software, Devops e Cloud, contribuindo para que as empresas de TI encontrem as melhores soluções tecnológicas.

## JÚLIO HENRIQUE MACHADO

Professor PUCRS

Possui graduação em Ciência da Computação pela Universidade Federal do Rio Grande do Sul (1997) e mestrado em Computação pela Universidade Federal do Rio Grande do Sul (2000). Atualmente é professor assistente na Escola Politécnica da Pontifícia Universidade Católica do Rio Grande do Sul. Tem experiência na área de Ciência da Computação, com ênfase em Linguagem Formais, Teoria da Computação e Linguagens de Programação, atuando principalmente nos seguintes temas: teoria dos autômatos, modelos de hipertexto, teoria das categorias, cursos hipermídia, programação de sistemas para Web, frameworks multiplataforma para dispositivos móveis. Atua como revisor técnico da Sagah.



# Ementa da Disciplina

Estudo de Arquitetura cliente-servidor para aplicações web SPAs (Single Page Applications). Estudo sobre frameworks cliente-side: React, Next.js, Redux, React Router, React Hook Form, Jest, Styled Components.

# Bibliografia da Disciplina

As publicações destacadas têm acesso gratuito.

## Bibliografia básica

ROLDÁN, C. S. React 17 Design Patterns and Best Practices. 3 ed. Packt, 2021.

SILVA, Maurício Samy. React Aprenda Praticando: Desenvolva Aplicações web Reais com uso da Biblioteca React e de Seus Módulos Auxiliares. Novatec, 2021.

PRESSMAN, Roger S; MAXIM, Bruce R. Engenharia de software: uma abordagem profissional. 9a. Ed. Porto Alegre: AMGH, 2021.

## Bibliografia complementar

[WHATWG. HTML living standard. WHATWG Web Hypertext Application Technology Working Group, 2021.](#)

ROZENTALS, Nathan. Mastering TypeScript. 4 ed. Packt, 2021.

GRONER, Loiane. Learning JavaScript Data Structures and Algorithms. Third Edition. Birmingham: Packt, 2018.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. JavaScript Descomplicado: programação para web, iot e dispositivos móveis. São Paulo: Érica, 2020.

GALLOTI, Giocondo M. Antonio. Arquitetura de Software. São Paulo: Pearson do Brasil, 2016.

# O que compõe o Mapa da Aula?

## MAPA DA AULA

São os capítulos da aula, demarcam momentos importantes da disciplina, servindo como o norte para o seu aprendizado.



## EXERCÍCIOS DE FIXAÇÃO

Questões objetivas que buscam reforçar pontos centrais da disciplina, aproximando você do conteúdo de forma prática e exercitando a reflexão sobre os temas discutidos. Na versão online, você pode clicar nas alternativas.



## PALAVRAS-CHAVE

Conceituação de termos técnicos, expressões, siglas e palavras específicas do campo da disciplina citados durante a videoaula.



## VÍDEOS

Assista novamente aos conteúdos expostos pelos professores em vídeo. Aqui você também poderá encontrar vídeos mencionados em sala de aula.



## PERSONALIDADES

Apresentação de figuras públicas e profissionais de referência mencionados pelo(a) professor(a).



## LEITURAS INDICADAS

A jornada de aprendizagem não termina ao fim de uma disciplina. Ela segue até onde a sua curiosidade alcança. Aqui você encontra uma lista de indicações de leitura. São artigos e livros sobre temas abordados em aula.



## FUNDAMENTOS

Conteúdos essenciais sem os quais você pode ter dificuldade em compreender a matéria. Especialmente importante para alunos de outras áreas, ou que precisam relembrar assuntos e conceitos. Se você estiver por dentro dos conceitos básicos dessa disciplina, pode tranquilamente pular os fundamentos.

## CURIOSIDADES

Fatos e informações que dizem respeito a conteúdos da disciplina.

## DESTAQUES

Frases dos professores que resumem sua visão sobre um assunto ou situação.

## ENTRETENIMENTO

Inserções de conteúdos para tornar a sua experiência mais agradável e significar o conhecimento da aula.

## CASE

Neste item, você relembra o case analisado em aula pelo professor.

## MOMENTO DINÂMICA

Aqui você encontra a descrição detalhada da dinâmica realizada pelo professor.

# Mapa da Aula

Os tempos marcam os principais momentos das videoaulas.

## AULA 1 • PARTE 1

### FUNDAMENTO

#### Business agility

Capacidade de uma organização se adaptar rapidamente às mudanças do mercado, mantendo-se ágil e eficiente. Ela envolve a capacidade de tomar decisões estratégicas com rapidez, responder às necessidades dos clientes e aproveitar oportunidades emergentes.

### PALAVRAS-CHAVE

**Gartner.com:** Site oficial da empresa Gartner, Inc., uma das principais empresas de consultoria e pesquisa em tecnologia da informação do mundo.

### EXERCÍCIO DE FIXAÇÃO

Qual é o objetivo do sistema distribuído?



00:00



00:56

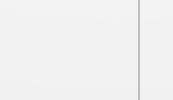
#### Introdução



03:25



A Arquitetura Client-Side é composta por três grandes partes: a **camada front-end**, que engloba tecnologias como HTML e CSS; a **camada back-end**, que consiste em serviços e servidores; e a **camada intermediária**, que faz a ponte entre as duas anteriores. Essas camadas se interligam e proporcionam convergências que impulsionam a arquitetura moderna, como mobilidade, internet das coisas, redes sociais, arquitetura de infraestrutura em nuvem e arquitetura de análise. A terceira plataforma, resultando dessa convergência, lida com bilhões de usuários, centenas de milhares de aplicativos e serviços em nuvem, promovendo a transparência e a distribuição dos sistemas. A arquitetura de software, dentro desse contexto, é uma disciplina que busca suportar decisões técnicas para a construção harmoniosa de software, considerando o ambiente, contexto e modelo de negócio.



## Padrões e elementos

Os principais **padrões arquiteturais** são o de **três camadas**, que envolve a comunicação entre camadas; a **arquitetura limpa** (onion architecture), que estabelece uma estrutura em camadas seguindo um roteiro de comunicação; e a **arquitetura hexagonal**, que utiliza adaptadores de entrada e saída para interagir com o mundo externo. Além disso, é importante considerar o contexto, tempo e cultura da empresa na escolha da arquitetura a ser implementada. Os elementos da arquitetura client-side incluem dispositivos diversos, plataformas (como browsers e sistemas operacionais), mensageria (como chatbots) e design system, que auxilia na criação de produtos consistentes e com identidade visual.

**“** Os padrões arquiteturais vêm para auxiliar a implantar o estilo que a gente decidir utilizar. **”**

## EXERCÍCIO DE FIXAÇÃO

Quando falamos de micro serviços, estamos nos referindo a:

## PALAVRA-CHAVE

**NFC:** É denominada a comunicação por campo de proximidade para troca de informações entre dispositivos.

## PALAVRA-CHAVE

**Design System:** É um conjunto de diretrizes e padrões para unificar a aparência e a experiência de uma marca em suas interfaces digitais.

## PALAVRA-CHAVE

**Vicent Cerf:** Cientista da computação e uma das figuras mais influentes na história da internet, é um dos criadores do protocolo TCP/IP.

## Fundamentos web

São alguns elementos essenciais na arquitetura client-side, que contribuem para uma comunicação eficiente entre os clientes e servidores:

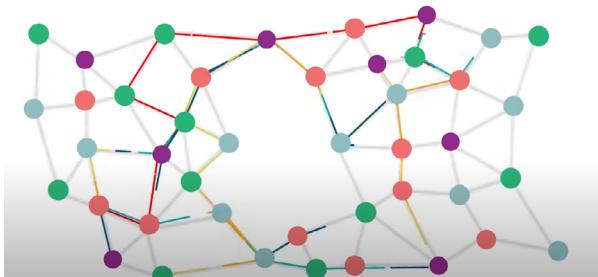
Os conceitos de URL, URI e URN, e o uso de CDN (rede de distribuição de conteúdo) podem ser utilizados para melhorar a entrega de conteúdo estático e dinâmico;

Os protocolos, como o HTTP são fundamentais para suportar as requisições e transferências de objetos na web.

## VÍDEO

### What is the Internet?

A DISTRIBUTED PACKET-SWITCHED NETWORK



[Clique aqui para assistir.](#)



21:46



26:00

### PALAVRA-CHAVE

**URI:** Ou Identificador Uniforme de Recursos, é utilizado para identificar recursos na web por meio de um padrão único.



30:00

“ O CDN é peça fundamental para Arquitetura Client-side. ”



33:55

### PALAVRAS-CHAVE

**W3C:** World Wide Web Consortium é a organização que desenvolve os padrões para a web.



34:58

### Arquitetura Client-Side

É importante considerar o contexto, o sistema e os componentes ao projetar uma aplicação. Elementos como CSS, HTML e JavaScript, desempenham papéis essenciais na construção de interfaces web interativas. Além disso, aspectos como meta tags, influenciam a forma como o conteúdo é apresentado em mecanismos de busca e em compartilhamentos em redes sociais. Esses elementos são fundamentais para a arquitetura front-end e para a experiência do usuário.



39:43

### PALAVRAS-CHAVE

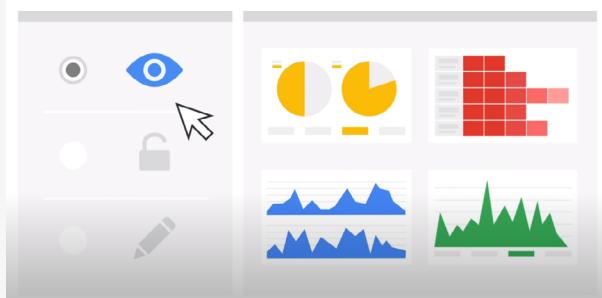
**Ryan Dahl:** Renomado programador e engenheiro de software, conhecido por ser o criador do Node.js, uma plataforma de desenvolvimento em tempo de execução baseada em JavaScript.

## AULA 1 • PARTE 2

“ SEO não é uma tecnologia, é um guia de orientação de como você deve construir seu site para ser visto nos sistemas de busca. ”

### VÍDEO

#### Explicando SEO | Pesquisa para iniciantes, Ep. 8



[Clique aqui para assistir.](#)

### Serviços web

O contexto histórico em que as empresas tradicionais surgiram, criou uma busca por agilizar suas operações financeiras e integrar sistemas legados. Nesse contexto, surge o serviço web, um recurso de software disponibilizado pela web, que retorna um determinado conteúdo ao ser consumido.

O conceito de SOA (Arquitetura Orientada a Serviços) foi uma evolução para facilitar a integração entre diferentes aplicações, por meio de protocolos como HTTP e SOAP. No entanto, o SOA também apresentou desafios, como a complexidade dos arquivos WSDL, indisponibilidade e problemas de gerenciamento. Com o tempo, houve uma transformação para arquiteturas mais granulares, como os micro serviços e arquiteturas orientadas a eventos.



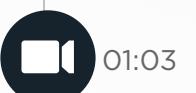
00:30



04:12

### Arquitetura Web

André Luiz Pereira aborda os diferentes aspectos relacionados à arquitetura web, como SEO (Search Engine Optimization) e boas práticas de CSS. Ele destaca a importância de otimizar um site para ser encontrado nos mecanismos de busca, seguindo um conjunto de diretrizes e critérios. Traz aspectos como a organização e nomeação adequada de arquivos e classes CSS, evitando vazamento de estilos e mantendo a coerência entre os componentes. Também cita os conceitos DOM (Document Object Model) e o uso de pré-processadores CSS. Esses elementos são fundamentais para o desenvolvimento de aplicações web eficientes e com boa usabilidade.



01:03



12:36

### PALAVRA-CHAVE

**React:** Desenvolvido pelo Facebook e lançado em 2013, o React é uma biblioteca JavaScript de código aberto para construção de interfaces de usuário (UI).



13:18



18:00

### PALAVRA-CHAVE

**SOAP:** É um protocolo de comunicação baseado em XML, utilizado para a troca de mensagens estruturadas entre ambientes distribuídos.

## API's

O web service é uma categoria específica de API, que é utilizada para a interação entre sistemas complexos e legados. O protocolo SOAP é frequentemente utilizado nesse contexto. Já as APIs são interfaces de programação de aplicativos e o uso de boas práticas de design, como o REST, é fundamental para sua construção. É importante fazer o uso adequado de práticas como o uso de métodos HTTP, padronização, versionamento, paginação e códigos de retorno.



23:36

## EXERCÍCIO DE FIXAÇÃO

O que é REST?



**“** Se preocupem com o response code. **”**



31:01

## PALAVRAS-CHAVE

**CloudFront:** É um serviço de distribuição de conteúdo (CDN) da Amazon Web Services (AWS) que acelera a entrega de conteúdo web e oferece baixa latência aos usuários finais.



36:03

## Contexto e complexidade I

André Luiz Pereira apresenta um estudo de caso sobre a jornada de um usuário, que precisa realizar algumas ações em um aplicativo móvel. Traz questões relacionadas à experiência do usuário, ciclo de vida de aplicativos e autenticação. Ele destaca a importância de estabelecer fronteiras na arquitetura, utilizar serviços específicos para determinadas funcionalidades, como emissão de nota fiscal e o uso de mensageria e chatbots para interação com o usuário. Também menciona o uso de CDNs, bancos de dados e outras tecnologias como parte da arquitetura.

**“** Vocês não podem aceitar que tudo seja feito pela arquitetura client-side. **”**



39:55

## PALAVRAS-CHAVE

**BFF:** Backend for Frontend é um padrão de arquitetura que consiste em ter diferentes APIs do backend para cada frontend específico, garantindo a independência e otimização de cada interface.

**“**Definir o contexto e os tipos de abordagem arquiteturais levam à boa construção dos componentes. **”**

## PALAVRAS-CHAVE

**XSS:** Cross-Site Scripting é uma vulnerabilidade de segurança em aplicações web que permite a injeção de código malicioso nos navegadores dos usuários.

42:10



## PALAVRA-CHAVE

45:27



**Watson IBM:** É uma plataforma de inteligência artificial e computação cognitiva desenvolvida pela IBM, que oferece serviços de análise de dados, reconhecimento de linguagem natural, processamento de linguagem, entre outros.

46:57



## Contexto e complexidade II

47:00



As abordagens arquiteturais de renderização devem ser pensadas, tanto do lado do cliente, quanto do lado do servidor. No caso da renderização do lado do cliente, o cliente solicita a primeira página, que é entregue pelo CDN, e o JavaScript é executado para interações posteriores. Já na renderização do lado do servidor, toda a página é renderizada no servidor e entregue ao cliente pronta para visualização. O client-side render permite maior interatividade e atualizações rápidas, enquanto o server-side render facilita a indexação por mecanismos de busca.

52:12



## AULA 1 • PARTE 3



## PALAVRAS-CHAVE

**Manifesto Web:** O Manifesto do Aplicativo Web fornece informações sobre uma aplicação (como nome, autor, ícone e descrição) em um arquivo de texto. O propósito do manifesto é instalar aplicações web na tela inicial de um aparelho, propiciando aos usuários um acesso mais rápido e uma experiência enriquecida.



20:27



32:46

## Frameworks: parte I

É importante escolher a abordagem de desenvolvimento adequada para o produto. Existem diversas técnicas de renderização no frontend, como o micro frontend, que permitem estruturar o produto de forma híbrida.

Os frameworks, como React e Vue.js, são conjuntos de ferramentas e bibliotecas que facilitam a criação de interfaces de usuário interativas e responsivas. Eles agilizam o desenvolvimento e permitem que os desenvolvedores se concentrem mais na lógica do que nas questões técnicas. No entanto, também podem apresentar desvantagens, como restrições e sobrecarga de aprendizado. A escolha do framework adequado depende do contexto e dos requisitos do projeto.

## PALAVRAS-CHAVE

**Stack Overflow:** É um site de perguntas e respostas voltado para programação e desenvolvimento de software.



35:44

## AULA 1 • PARTE 4



00:00



06:50

## PALAVRA-CHAVE

O professor apresenta uma análise dos principais frameworks de desenvolvimento, como Angular, React e Vue. O Angular nasceu em 2009, baseado em JavaScript, e em 2016 foi lançada a versão 2, com Typescript. O React, criado pelo Facebook em 2013, trouxe a ideia de Virtual DOM para melhorar a performance. Já o Vue.js, lançado em 2014, é um framework progressivo, flexível e reativo, com a opção de utilizar Typescript.

Cada framework possui características específicas e a escolha deve considerar o contexto e os requisitos do projeto. A flexibilidade pode ser positiva, mas também requer cuidado para evitar a adoção excessiva de bibliotecas sem necessidade.

**Evan You:** Desenvolvedor de software e engenheiro de front-end, criador do Vue.js, um framework JavaScript para construção de interfaces de usuário.

**“** Não se apeguem a tecnologia, o conhecimento que você tem num framework é facilmente aplicado em outro. **”**

### Frameworks multiplataformas

Os frameworks de desenvolvimento multiplataforma surgiram com o aumento do uso de smartphones em 2008. Eles permitem compartilhar códigos-base entre diferentes plataformas, como Android, iOS e web, trazendo eficiência e reutilização de código.

Entre os principais frameworks, destacam-se o React Native e o Flutter, que possibilitam criar aplicações nativas de alto desempenho com código único, embora o Flutter seja considerado mais eficiente por ter uma renderização baseada em Dart e um menor uso de bridge para recursos nativos.

### PALAVRAS-CHAVE

**Arquitetura Bridge:** É um padrão de projeto que separa a interface do usuário das classes de implementação, permitindo que elas variem independentemente.



08:25

### Frameworks: parte III

O framework Next.js é uma extensão do React, desenvolvido para otimizar a arquitetura de sites e aplicativos. Ele oferece recursos de renderização otimizados com estratégias de pré-renderização e hidratação, além de possibilitar o uso de renderização do lado do cliente e do servidor. O framework é flexível e permite a combinação de desempenho aprimorado com rica interatividade, facilitando o desenvolvimento de aplicações web. É importante dominar os conceitos e entender o contexto de negócio para escolher o framework mais adequado.



11:47



17:10



### EXERCÍCIO DE FIXAÇÃO

Qual é o melhor benefício dos frameworks multiplataformas?



23:22-



28:03

### Flutter

O Flutter usa a linguagem de programação Dart e possui uma vasta biblioteca de pacotes pré-fabricados para acelerar o desenvolvimento e uma arquitetura MVP. Além disso, em comparação com o React Native, o Flutter é mais fluido na abstração de recursos nativos e oferece suporte a várias plataformas, como Android, iOS, web e desktop, através de um único código.



**Arquitetura MVP:** Model-View-Presenter é um padrão de arquitetura de software que separa a lógica de negócios (Model), a interface do usuário (View) e a apresentação (Presenter).

## AULA 2 • PARTE 1

### Serviços

Os micro serviços são uma abordagem que busca maior granularidade e produtividade, permitindo lidar com serviços próprios e de terceiros de forma mais eficiente. No entanto, é importante ressaltar que o uso de micro serviços pode não ser a melhor opção para todas as empresas, especialmente aquelas com sistemas legados complexos. Em alguns casos, optar por modelos mais monolíticos, como o uso de SPAs ou aplicações web mais integradas, pode ser mais adequado, especialmente para startups que buscam validar hipóteses e ampliar seus negócios de forma mais ágil.



01:24

05:10



*“Você tem que pensar numa forma arquitetural que o serviços mudam, que o escopo muda, que as plataformas mudem.”*

12:47



### Equipes multifuncionais

Os times possuem uma grande complexidade nas empresas, sendo necessário adotar estratégias para lidar com as dependências e conflitos entre eles. A divisão dos times em especialistas, como desenvolvedores, testadores e analistas, pode gerar problemas de sincronização e dependências excessivas. Para solucionar essas questões, é possível utilizar equipes multifuncionais baseadas em técnicas como o Design Orientado a Domínio (DDD) para alinhar a linguagem e o modelo de negócio da empresa. Dessa forma, a organização pode buscar maior agilidade e resiliência, bem como explorar abordagens de micro serviços e micro frontends.

### PALAVRAS-CHAVE



16:32

**Cross-functional teams:** São equipes multidisciplinares compostas por membros com diferentes habilidades e especialidades, trabalhando juntos para alcançar um objetivo comum.

**“** Um dos princípios básicos da arquitetura é a modelagem do negócio. **”**



19:38

## Fronteiras e DDD

Antes de dividir as equipes e implementar a estratégia de monolito modular, é importante definir fronteiras e subdomínios dentro de uma empresa. O uso do Design Orientado a Domínio (DDD) é sugerido para mapear os subdomínios e facilitar a divisão dos monolitos em micro serviços estratégicos. A abordagem de monolito modular pode ajudar na escalabilidade, no foco na entrega de valor e na resolução de gargalos, permitindo uma migração gradual para micro serviços. A estratégia é especialmente relevante para empresas que buscam oferecer novos serviços e atender diferentes públicos além dos usuários consumidores.

20:29



23:25



## FUNDAMENTO

### Domain driven design

O DDD é uma abordagem de design de software que se concentra na modelagem do domínio de negócios e na colaboração entre especialistas do domínio e desenvolvedores. Visa criar um modelo de software alinhado com a linguagem e as regras do negócio, melhorando a comunicação e a compreensão das complexidades do domínio.

24:39



## PALAVRA-CHAVE

**“** DevOps fazem parte da arquitetura moderna. **”**



31:34

## Atores estratégicos

É importante incluir atores estratégicos, como o gateway e o BFF (Backend For Frontend) para garantir o funcionamento do modelo de micro serviços ou monolitos modulares. O gateway atua como uma camada de API, gerenciando o acesso e a autorização entre os serviços, enquanto o BFF é uma evolução do modelo do gateway, descentralizando a comunicação e permitindo retornar informações de acordo com o contexto de uso, melhorando a performance. Além disso, o uso de GraphQL é uma solução sofisticada para atender diferentes dispositivos e interfaces de forma eficiente, permitindo que os clientes solicitem apenas os campos necessários.

35:03



38:28



## PALAVRA-CHAVE

**Strangler Fig:** É um padrão de migração de sistemas que consiste em substituir partes de um sistema legado gradualmente até que ele seja completamente substituído.

## EXERCÍCIO DE FIXAÇÃO



Qual é a principal diferença entre Gateway e BFF?

## AULA 2 • PARTE 2

### GraphQL

A abordagem de arquitetura de microserviços e monolitos modulares é essencial para desenvolvedores front-end que dependem de serviços e consomem APIs. A inclusão do gateway e do BFF é importante para gerenciar o acesso e autorização entre os serviços de forma descentralizada. As vantagens incluem heterogeneidade de tecnologia, equipes independentes e escalabilidade, mas é importante considerar o contexto da organização ao optar por essa abordagem. A evolução da arquitetura Cloud-Native é um desafio, mas esses conceitos são fundamentais para desenvolver uma estrutura robusta e diferenciada no mercado.



00:00

05:06



### PALAVRA-CHAVE

**Blue-Green e Canary:** São estratégias de implantação de software que permitem a liberação controlada de novas versões em produção, minimizando riscos e impactos.

05:07



### PALAVRA-CHAVE

**Feature Tags:** São tags utilizadas no código para habilitar ou desabilitar determinadas funcionalidades de forma controlada.

**“**A arquitetura Client-Side vem ganhando complexidade, exigindo o maior dessa estrutura.**”**

09:35

11:03



### Micro frontends

Os micro frontends são uma evolução estratégica e necessária para empresas que buscam escalar seus produtos e atender às demandas de clientes mais exigentes. A abordagem de micro frontends divide as interfaces em pedaços menores e mais gerenciáveis, permitindo que equipes independentes trabalhem em subdomínios específicos. Ao aplicar essa estratégia, as equipes podem sincronizar micro arquiteturas, criar aplicações com visões comerciais mais estratégicas e possibilitar maior agilidade no desenvolvimento. Além disso, a adoção de padrões como o Domain-Driven Design (DDD) pode ser benéfica para a organização e o alinhamento entre os times.

**“**Pense no micro serviço como uma evolução natural.**”**

12:17



13:10



## PALAVRAS-CHAVE

**White Label:** É um produto ou serviço desenvolvido por uma empresa, mas comercializado e personalizado por outras marcas como se fossem suas.

23:55



**“**Nós temos a obrigação ou o privilégio de investir um tempo suficiente para olhar para o time.**”**

24:38



### Estratégias para micro frontends

São duas as **estratégias para a organização de micro frontends**: a abordagem horizontal, dividindo as telas em fragmentos, e a abordagem vertical, em que cada equipe é responsável por um subdomínio de negócio, desenvolvendo páginas SPA com páginas únicas. A **estratégia vertical**, defendida por Luca Mezzalira, é considerada pertinente e aplicável em contextos de times grandes e sistemas complexos. Já a adoção do Domain-Driven Design (DDD) é recomendada para alinhar as micro arquiteturas e criar sinergia com os microservices. A **abordagem vertical** permite maior visão comercial e competitividade saudável dentro da empresa, otimizando o tempo de entrega e facilitando a escalabilidade do negócio. No entanto, existem desafios relacionados à estrutura front-end, estado da aplicação e comunicação entre os micro frontends.

24:43



## PALAVRA-CHAVE

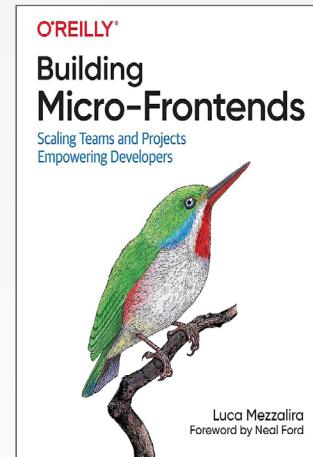
**Luca Mezzalira:** É um arquiteto de software, evangelista técnico e palestrante conhecido por seu trabalho em micro-frontends.

25:11



## LEITURA INDICADA

Livro: Building micro-frontends



### PALAVRAS-CHAVE

**Business Agility:** É a capacidade de uma organização se adaptar rapidamente e de forma flexível às mudanças do mercado e às demandas dos clientes.

31:55



### PALAVRAS-CHAVE

**Teste A/B:** Técnica de teste que compara duas versões diferentes de uma página ou recurso para determinar qual delas tem melhor desempenho ou aceitação pelos usuários.

34:59



### PALAVRAS-CHAVE

**Stakeholders:** São indivíduos ou grupos que têm interesse ou influência em um projeto ou decisão, incluindo clientes, acionistas, colaboradores e outros envolvidos.

35:30



### PALAVRAS-CHAVE

**App Shell:** É uma estrutura de organização do aplicativo web, usualmente uma SPA de nível mais alto, que organiza o acesso a um API Gateway ou BFFs, abstraindo o conhecimento do usuário da aplicação, encapsulando o acesso a microfrontends.

38:52



### Estratégia de combinação

A combinação de microservices e micro frontends é uma estratégia coerente para a organização. Sugere-se a divisão gradual de monolitos modulares no backend e criação de microservices, bem como a criação de micro frontends baseados em páginas únicas de SPAs. Essa abordagem permite maior agilidade, escalabilidade e eficiência das equipes, proporcionando uma evolução progressiva do sistema. O uso de Web Components e um BFF (Back-end for Front-end) é recomendado para sincronizar os micro frontends e garantir a renderização adequada das páginas de forma sob demanda, melhorando a performance e a experiência do usuário.

41:43



43:32

**Exemplo prático**

O professor apresenta um exemplo prático de aplicação de micro frontends utilizando os conceitos de abordagem horizontal e vertical. O cenário consiste em um container de aplicação (app shell) que recebe a demanda do navegador. Dentro desse container, são utilizados diversos micro frontends para renderizar diferentes funcionalidades. O uso do conceito de SPA e roteamento é utilizado para redirecionar e renderizar os componentes necessários de acordo com as ações do usuário no navegador.

É importante dar atenção e ser resiliente na arquitetura, tanto no back-end quanto no front-end, devido ao desenvolvimento web moderno e complexo. A decomposição de monolitos sem front-end e a adoção de micro frontends são um caminho sem volta para melhorar a renderização e estrutura do projeto.

**AULA 2 • PARTE 3**

02:26

**Requisitos arquiteturais**

Os requisitos arquiteturais na definição da estrutura do projeto são importantes tanto no back-end quanto no front-end. É necessário uma separação clara de responsabilidades e como isso podemos evitar problemas de código confuso, dificuldade de manutenção e escalabilidade.

A arquitetura limpa é uma abordagem para dividir o projeto em camadas independentes e coerentes, utilizando o princípio SOLID como apoio para implementar essas camadas. A iniciativa do “Clean Frontend” visa desacoplar e abstrair as camadas de dependência para permitir maior flexibilidade e adaptabilidade tecnológica.

**PALAVRAS-CHAVE**

**Clean Architecture:** Robert C. Martin desenvolve uma arquitetura de software que busca separar as regras de negócio da implementação técnica, facilitando a manutenção e evolução do sistema.

07:20



09:54



## PALAVRA-CHAVE

**Clean Dart:** É uma arquitetura de código aberto para projetos em Dart, inspirada nos princípios da Clean Architecture.

### Componentização

A componentização no desenvolvimento busca uma distribuição sustentável, o reuso de código e a manutenibilidade. Para isso, é importante utilizar bibliotecas como CSS Modules e Styled Components para encapsular estilos, evitando duplicação de código e aumentando a produtividade. Os componentes web são uma solução para criar widgets independentes que podem ser compartilhados com micro frontends. Já o gerenciamento de estado utilizando bibliotecas como Redux e Flux mantém a integridade das telas e facilita o desenvolvimento.

18:07



29:24



### Performance

André Luiz Pereira aborda a importância da performance no desenvolvimento de aplicações, alertando sobre os riscos de negligenciar a otimização e os possíveis problemas de carregamento e experiência do usuário. Sugere a realização de testes de desempenho constantes e a aplicação de técnicas de otimização, como a compressão de recursos e o uso de CDNs. E destaca a relevância de ferramentas como o Lighthouse para analisar a performance e oferece dicas para melhorar a eficiência, como reduzir o tamanho das imagens e eliminar código não utilizado.

## EXERCÍCIO DE FIXAÇÃO

Qual é a principal característica do TTFB?

31:37



## PALAVRA-CHAVE

**HTTPArchive:** É um projeto que registra e mede o desempenho de sites e páginas da web, fornecendo dados e análises sobre o estado da web.

## Acessibilidade



43:26

A acessibilidade é muito importante para tornar sites e aplicativos acessíveis para pessoas com deficiência e evitar exclusão social. Recomenda-se adotar práticas e guias de acessibilidade, como o Lighthouse, para garantir que a interface seja sensível às necessidades dos usuários. Além disso, há a responsabilidade em tornar os aplicativos responsivos para diferentes dispositivos, utilizando quebras de página e a mídia query para adaptar o layout. Sugere-se começar com funcionalidades específicas ou micro frontends para reduzir o esforço e testar a responsividade ao longo do desenvolvimento.

## AULA 2 • PARTE 4

### Tendências

Há duas tendências na área de frontend: o uso de ferramentas low code ou no code para acelerar o desenvolvimento de aplicativos com menos código; e a utilização de inteligência artificial, como chatbots, para melhorar a experiência do usuário. É importante escolher ferramentas com cautela e considerar a manutenção a longo prazo.

O professor André Luiz Pereira menciona algumas ferramentas como o Twilio, para implementar chatbots e trabalhar com workflows.

00:16



### PALAVRA-CHAVE

00:33



00:56



### VÍDEO

#### Watch the overview

Acesse clicando [aqui](#) ao vídeo demonstrativo do OutSystems.

## EXERCÍCIO DE FIXAÇÃO

A arquitetura do chatbot é orientada a:



08:56



### Take Blip e Chatbots

A plataforma de inteligência conversacional Take Blip trabalha com chatbots e inteligência artificial para criar fluxos de conversação e interação com usuários em diferentes canais, como WhatsApp, Telegram e Instagram. Os chatbots são uma oportunidade para atingir diferentes públicos e melhorar a experiência do usuário, além de outras tecnologias como uso de inteligência artificial, como reconhecimento facial e sugestão de produtos com base em características físicas.

### LLMs

Modelos de linguagem, como o Chat GPT, CoPilot e Code Whisper, são algumas das tendências em aprendizado de máquina (LLMs). Esses modelos utilizam algoritmos de aprendizado para processar e entender a linguagem humana, sendo treinados com grande quantidade de dados na web e compostos por múltiplas camadas de redes neurais. O CoPilot e o Code Whisper são assistentes de desenvolvimento que auxiliam na criação de código, fornecendo sugestões, criando funções, testes e validações do que pode revolucionar o processo de desenvolvimento de software.



19:46

35:34



### Ética e tendências

A cada dia surgem novas ferramentas e plataformas, como o CoPilot e Code Whisper, que utilizam IA para auxiliar no desenvolvimento de código. No entanto, é necessário ter cautela quanto ao uso de códigos proprietários e estar atento aos desafios e questões éticas envolvidas no uso da IA. A cultura empresarial resiliente, foco no cliente e o contínuo aprendizado são fundamentais para enfrentar as transformações tecnológicas e alcançar o sucesso na área de desenvolvimento de software.

“ Softwares mudam e isso significa que precisamos construir uma cultura resiliente. ”



38:48

40:13



### PALAVRA-CHAVE

41:51



“ Precisamos das pessoas engajadas, inseri-las, discutir, envolvê-las. Sem isso, você não tem resultado. ”

**MVA:** Minimum Value Architecture é uma abordagem de desenvolvimento que prioriza a entrega rápida e incremental de valor ao cliente, focando no desenvolvimento das funcionalidades essenciais primeiro.

## AULA 3 • PARTE 1

### Node.js

Antes de iniciar um projeto, é necessário a configuração de um ambiente local de desenvolvimento e, para isso, podemos usar dependências como o Node.js, para a construção e a implantação de sites. O uso do React como uma biblioteca do lado do cliente auxilia na criação de componentes visuais e é um framework necessário para o desenvolvimento de aplicações web. Sugere-se o uso de plataformas online gratuitas para simplificar a configuração do ambiente local.

O objetivo é explorar conceitos essenciais do React e a documentação oficial é uma fonte importante de aprendizado.



01:23

12:08



### Toolchains

Julio Henrique Machado apresenta a configuração de um ambiente de desenvolvimento utilizando o React e duas ferramentas, Create React App e Vite. Ambas possuem estruturas de projeto tradicionais com arquivos de configuração e dependências. A partir da configuração, é possível criar componentes visuais do React, e os arquivos são escritos em JSX, que combina HTML com JavaScript. Ele destaca a importância do processo de compilação e empacotamento (build) para otimizar e preparar a solução para distribuição. As ferramentas Vite e Create React App são apresentadas como opções para criar projetos React de forma rápida e interativa.



15:19

24:41



### Componentes visuais: parte I

O professor apresenta uma discussão sobre o desenvolvimento de componentes visuais com React e menciona ferramentas online, como CodeSandbox, StackBlitz e Replit, que permitem criar e testar projetos React na nuvem.



33:04

A importância de componentização e o uso de componentes funcionais são destacados. A renderização dos componentes é feita pelo React, que gerencia o DOM de forma eficiente. A abordagem enfatiza a organização dos componentes e a utilização de padrões arquiteturais na construção de soluções multiplataforma.



35:24

### PALAVRAS-CHAVE

**Bootstrap:** É um framework front-end de código aberto desenvolvido pelo Twitter, que oferece um conjunto de ferramentas e estilos para agilizar o desenvolvimento web responsivo.

**“** A primeira coisa que é o nosso foco, é o desenvolvimento dos componentes visuais. **”**

**“** A ideia de conseguir enxergar os componentes e como eles se organizam é essencial para quem trabalha com react. **”**

## EXERCÍCIO DE FIXAÇÃO

Qual é a abordagem mais moderna para o desenvolvimento de componentes no React?

35:43



## FUNDAMENTO

### Princípio da responsabilidade única

Princípio de design de software que afirma que uma classe ou um módulo deve ter apenas uma única responsabilidade. Isso promove a coesão e a manutenção do código, tornando-o mais fácil de entender, modificar e estender. Nele, cada componente deve se concentrar em uma única tarefa ou objetivo.

“ Um componente do react funcional só pode retornar elementos que tem uma única raiz. ”

39:20



### Componentes visuais: parte II

O desenvolvimento de componentes visuais com React e o uso do JSX são usados para combinar aspectos visuais com código JavaScript. O professor Julio Henrique Machado apresenta um exemplo de um componente funcional simples que renderiza “Alô mundo” na página. O uso de prompts para receber dados externos é abordado, assim como a importância de manter a responsabilidade única nos componentes. E traz a importância do pré-processamento correto dos estilos CSS, destacando a simplicidade e rapidez do desenvolvimento de soluções SPA com React.

“ Solução SPA é perfeitamente válida para validar um negócio, além de ser rápida de ser produzida. ”

42:09



50:29



## AULA 3 • PARTE 2

00:00



### Componentes visuais: parte III

O professor apresenta a criação de componentes com React, utilizando o JSX para combinar aspectos visuais com código JavaScript. Traz um exemplo de componente funcional simples que renderiza “Alô mundo” na página.

08:36



**“**A decisão é do designer de quais são as estruturas do HTML e os estilos. O programador coloca em código. **”**

09:10



### Componentes visuais: parte IV

O JSX pode ser utilizado em conjunto com o JavaScript para desenvolver componentes no React.

Julio Henrique Machado apresenta exemplos de renderização condicional e a utilização de prompts para passar dados para os componentes. Além disso, mostra como lidar com eventos e a importância de nomear as funções adequadamente. Ele destaca a separação de responsabilidades entre designers e programadores na criação dos componentes.

14:04



**“**Jamais invente de gerar um valor de ID programaticamente. **”**

18:17



## PALAVRAS-CHAVE

**Bubbling:** É um conceito em eventos de JavaScript, onde um evento ocorrido em um elemento se propaga para os elementos pai na hierarquia do DOM.

21:50



### Uso de eventos

O professor apresenta exemplos de uso de eventos em componentes do React, mostrando como os eventos são tratados e propagados na árvore do DOM. Ele destaca a utilização de React Hooks, que são funções especiais utilizadas para controlar o ciclo de vida dos componentes e gerenciar dados e eventos. Existem diversos tipos de React Hooks, cada um com um propósito específico, portanto, é importante revisar os conceitos básicos de componentes e props antes de utilizá-los.

28:48

**useState**

O professor apresenta a criação de um componente React “Contador” utilizando React Hooks useState para gerenciar o estado interno do componente. O estado “Contagem” é mantido e atualizado através da função “setContagem”, que pode ser utilizada para alterar o valor do estado. Um botão é utilizado para incrementar a contagem e exibir o valor atualizado em um elemento “span”.

31:45



*“As funções que gerenciam o estado devem ser puras, entra um estado e eu devolvo um novo estado, eu nunca altero o estado.”*

**EXERCÍCIO DE FIXAÇÃO**

O que é recebido ao utilizar o useState no React?

41:18

**useReducer**

O React Hook useReducer é utilizado para gerenciar estados complexos que requerem atualizações condicionais. A função useReducer permite utilizar uma função redutora que recebe o estado atual e uma ação para calcular o novo estado. Essa abordagem é especialmente útil para lidar com várias ações possíveis sobre um estado, evitando atualizações diretas. O exemplo apresentado traz um contador que pode ser incrementado ou decrementado por botões distintos, onde cada botão dispara a ação apropriada. O useReducer separa a lógica de gerenciamento de estado do componente, permitindo uma arquitetura mais clara e escalável.

43:16



*“Usualmente o estado e a ação são objetos complexos ou tão complexos quanto eu desejo.”*

## AULA 3 • PARTE 3

### UseEffect

O React Hook useEffect deve ser usado com cuidado, pois permite a execução de efeitos colaterais após cada renderização do componente. É indicado para casos em que a lógica de renderização não pode ter efeitos colaterais diretamente. O exemplo apresentado mostra o uso do useEffect para atualizar um relógio na tela, utilizando um temporizador externo para disparar a atualização. A função retornada pelo useEffect pode ser utilizada para limpar o efeito colateral, como interromper o temporizador. O useEffect pode executar apenas uma vez ou ser condicionado a dependências específicas para ser disparado novamente. Sua utilização avançada pode envolver diversos tipos de processamento.



00:00

00:30



**Dentro do react a lógica de renderização dos nossos componentes não pode possuir efeitos colaterais.**

### EXERCÍCIO DE FIXAÇÃO

O que o useEffect observa para determinar quando deve executar no React?

### Biblioteca React

Na biblioteca React podemos criar Hooks personalizados para gerenciar estados externos. O professor Julio Henrique Machado traz um exemplo de um Hook customizado chamado useOnlineStatus, que observa as mudanças de status da rede do navegador e notifica os componentes que o utilizam sobre o estado online/offline. O componente StatusBar é utilizado para renderizar uma mensagem na página com base no estado obtido do Hook personalizado. Ele menciona a importância de gerenciar estados complexos e persistentes utilizando APIs externas para armazenamento, como cookies ou estado do servidor. Além disso, enfatiza a importância da reutilização de componentes e a separação de responsabilidades para criar uma aplicação mais modular e escalável.



14:32

20:50



### PALAVRA-CHAVE

**Cookies:** São pequenos arquivos armazenados no navegador dos usuários para rastrear informações e melhorar a experiência do usuário em sites da web.

## SPA e React

Ao desenvolver uma Single Page Application com o React podem surgir alguns desafios, devido às múltiplas bibliotecas e dependências para gerenciar o estado global da aplicação, autenticação e autorização. O Recoil é, uma biblioteca que facilita o gerenciamento de estado global na aplicação, proporcionando hooks e funções úteis para trabalhar com o estado. É importante ter uma arquitetura bem definida, como a arquitetura hexagonal, para separar as responsabilidades de diferentes camadas da aplicação.

21:36



## Formulários

O professor discute o gerenciamento de estado em formulários usando React e apresenta dois métodos: formulários não controlados, onde o estado é mantido pelo próprio HTML do navegador e acessado por referência usando o Hook useRef; e formulários controlados, onde o estado é gerenciado pelo componente do formulário usando o Hook useState.

O professor destaca a importância do uso de componentes ricos e bibliotecas para manipulação de formulários para facilitar a validação, aplicação de máscaras e simplificar a lógica de manipulação de dados.

## AULA 3 • PARTE 4

00:00



## Frameworks

A estruturação de uma Single Page Application complexa apenas com React não é viável, sendo necessário o uso de bibliotecas adicionais como Redux ou Recoil para o gerenciamento de estados, além de outros elementos para separar as responsabilidades dos componentes visuais. Isso leva à necessidade de frameworks como Next.js, que oferecem uma estrutura pronta para desenvolver aplicações com React, fornecendo roteamento, gerenciamento de estado, otimizações de conteúdo, autenticação, autorização, entre outras funcionalidades. O Next.js é um framework opinado que permite desenvolver aplicações de forma mais eficiente, além de oferecer suporte para diferentes tipos de processamento de CSS e integração com serviços de nuvem, como o Vercel.

**“** O framework fornece as ferramentas. Use, integre-as com seus componentes React e tudo vai funcionar. **”**

## PALAVRAS-CHAVE

**Tailwindcss:** Framework CSS altamente configurável e utilitário que permite criar estilos personalizados de maneira rápida e eficiente, usando classes pré-definidas. É conhecido por sua abordagem “utility-first”, onde você combina classes para estilizar os elementos HTML.

## PALAVRAS-CHAVE

**Metadados:** Informações descritivas que fornecem detalhes sobre um recurso, como um documento, imagem ou página da web. Essas informações incluem dados que são usados para indexação e organização de conteúdo por máquinas de busca e sistemas de gerenciamento de informações.

## LEITURAS INDICADAS

### How to style a react app

<https://www.freecodecamp.org/news/style-react-apps-with-css/>

10:25



### Aprender e entender

Julio Henrique Machado aborda o uso do Next.js, um framework para desenvolver aplicações com React, oferecendo recursos como roteamento, gerenciamento de estado, otimizações de conteúdo e integração com serviços de nuvem. O Next.js utiliza componentes funcionais e possui um índice CSS próprio com diversas classes pré-definidas para estilização. Além disso, é possível explorar exemplos fornecidos pelo próprio Next.js para aprender e entender melhor sua estrutura e funcionalidades.

17:45



19:04



### Roteamento

O professor explora o roteamento no Next.js, destacando o uso do app Router como uma opção moderna que suporta o desenvolvimento hierárquico das páginas. O framework utiliza arquivos no sistema de arquivos para criar rotas e permite o uso de layouts compartilhados e templates para diferentes segmentos de rota. O Next.js também oferece componentes prontos para uso, como layouts e erros, facilitando o desenvolvimento. O processo de roteamento é produtivo e eficiente, proporcionando uma estrutura organizada para a aplicação.

28:43



O uso do app halter para roteamento hierárquico é destacado, permitindo a criação de layouts e templates compartilhados. O componente Link é utilizado para navegação entre as páginas no lado do cliente, enquanto o uso do servidor de renderização torna o processo eficiente e permite o uso de componentes de cliente. O Next.js traz uma praticidade para o gerenciamento de rotas e a integração de funcionalidades como o uso de layouts e a navegação entre páginas.

43:15



Os componentes podem ser renderizados tanto no lado do servidor quanto no lado do cliente. O Next.js é uma solução produtiva para gerenciar as rotas e integrar diferentes estilos de CSS.

48:23



### Exemplo prático

O professor Julio Henrique Machado apresenta um exemplo prático de uma aplicação com arquitetura client-side utilizando componentes React e o gerenciamento de rotas do Next.js. Destaca a facilidade de integração entre essas tecnologias e a utilização de templates pré-prontos fornecidos pelo Next.js. O exemplo demonstra o uso de roteamento dinâmico e a obtenção de dados por meio de requisições API. O professor conclui enfatizando a importância de compreender os conceitos para navegar entre diferentes frameworks de forma mais eficiente.

# Resumo da disciplina

Veja, nesta página, um resumo dos principais conceitos vistos ao longo da disciplina.



# Avaliação

Veja as instruções para realizar a avaliação da disciplina.

Já está disponível o teste online da disciplina. O prazo para realização é de **dois meses a partir da data de lançamento das aulas**.

Lembre-se que cada disciplina possui uma avaliação online.  
A nota mínima para aprovação é 6.

Fique tranquilo! Caso você perca o prazo do teste online, ficará aberto o teste de recuperação, que pode ser realizado até o final do seu curso. A única diferença é que a nota máxima atribuída na recuperação é 8.

