BIUD Talks

Arquitetura de software

Luk - TechLead

Cronograma do Programa

22/04	**Wanderson**	Inteligência Artificial para Dev
29/04	**Praxedes**	Testes Unitários
06/05	**Luk**	Arquitetura de Software
13/05	**Regis**	Integração de Sistemas
20/05	**Rogério**	Root Cause Analysis (RCA)
⚡ 27/05	**Gustavo**	Metodologia Ágil
⚡ 03/06	**Gabriel Pereira**	Banco de Dados

Estrutura de cada Sessão

Apresentação

20-30 minutos

Conteúdo e expertise do palestrante

Debate & Discussão

30 minutos

Dúvidas, insights e networking

? Moderador: Rogério

Responsável por mediar o debate e garantir engajamento de todos os participantes

POR QUE PLANEJAR IMPORTA?

A arquitetura define a fundação, as instalações e a viabilidade de crescimento do sistema.

decisões tomadas no início do desenvolvimento de software são difíceis e caras de mudar posteriormente.

A arquitetura não serve para o código que você escreve hoje, mas para garantir que o código que você escreverá amanhã ainda tenha um lugar sólido para ficar.

Uma boa arquitetura

- SUSTENTAÇÃO E SEGURANÇA
- FACILIDADE DE MANUTENÇÃO
- PREVISIBILIDADE E CUSTO
- COMUNICAÇÃO
- EXTENSIBILIDADE

A cidade do Codigo

Estilo arquitetônico

- Arquitetura Monolítica
- Microsserviços
- Arquitetura em Camadas
- Model-View-Controller
- SOA (Orientada a Serviços)
- Arquitetura Cliente-Servidor

O Grande Armazém

Arquitetura Monolítica

Imagine um shopping que é um único e enorme galpão. **Tudo está sob o mesmo teto.**

- Simples de construir no início
- Se a luz apaga, o armazém inteiro para.
- Difícil de reorganizar internamente.

A Feira Livre

Microsserviços

Bancas independentes que formam um mercado. **Serviços pequenos e autônomos**.

- Se uma banca fecha, a feira continua.
- Fácil de adicionar novas bancas (escalar).
- Logística complexa entre as bancas.

O Shopping Vertical

Arquitetura em Camadas

Organização por níveis hierárquicos, como os andares de um shopping mall.

- Térreo: Apresentação (Lojas)
- 1° Andar: Negócio (Administração)
- Subsolo: Persistência (Estoque)

A Operação de Cinema

Model-View-Controller

Separação total de papéis para uma experiência fluida.

- View (A Tela): O que o público vê.
- Controller (O Staff): Gerencia pedidos e entradas.
- Model (A Projeção): Onde reside o filme/dados.

O Condomínio de Serviços

SOA (Orientada a Serviços)

Casas independentes que compartilham infraestrutura centralizada (Portaria, Clube, Manutenção).

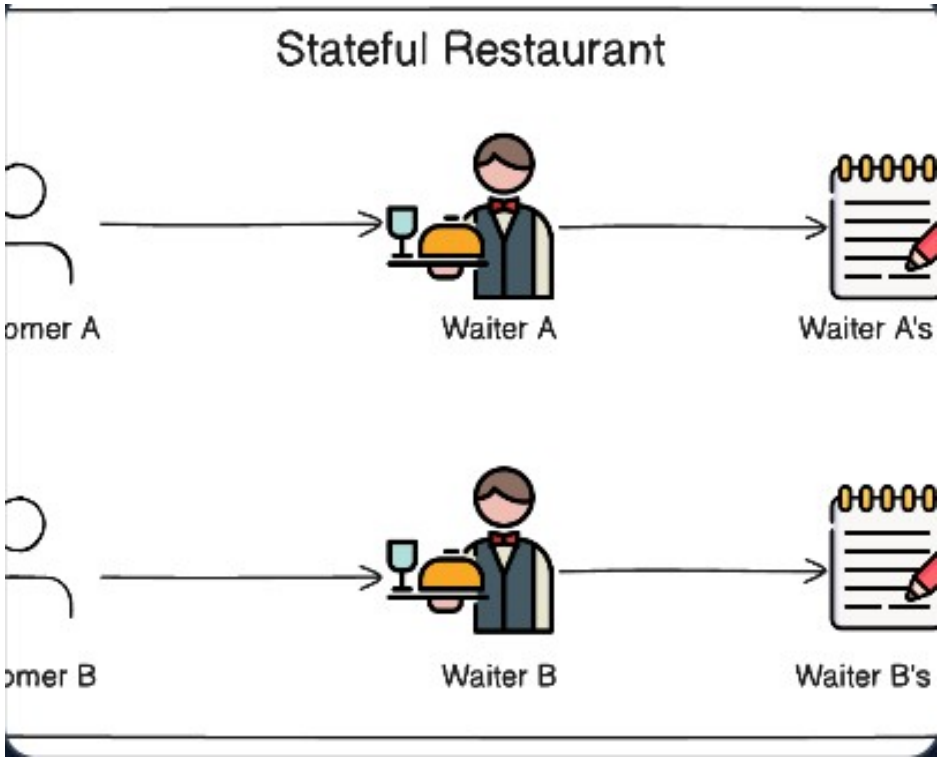
- Focada na reutilização de componentes comuns através de protocolos de comunicação definidos.

O Condomínio de Serviços

Arquitetura Cliente-Servidor

A divisão clara entre quem pede e quem provê.

- Cliente (Mesa): Solicita o prato (recurso).
- Servidor (Cozinha): Processa e entrega o prato.



TRÊS NÍVEIS DE QUALIDADE

Para construir software de alta qualidade, precisamos olhar para o projeto em **três escalas diferentes**.
Cada nível possui suas ferramentas e princípios que garantem a integridade da obra final.

1. SOLID: O MICRO

2. DESIGN PATTERNS: O MÉDIO

3. CLEAN ARCHITECTURE: O MACRO

Construir software sem esses guias é como erguer um prédio na areia: ele pode subir rápido, mas nunca terá futuro.

Area Interna (DESIGN PATTERNS)

1. PADRÕES CRIACIONAIS

Fabricação de Materiais

Evitam a fabricação manual de cada peça no canteiro de obras.

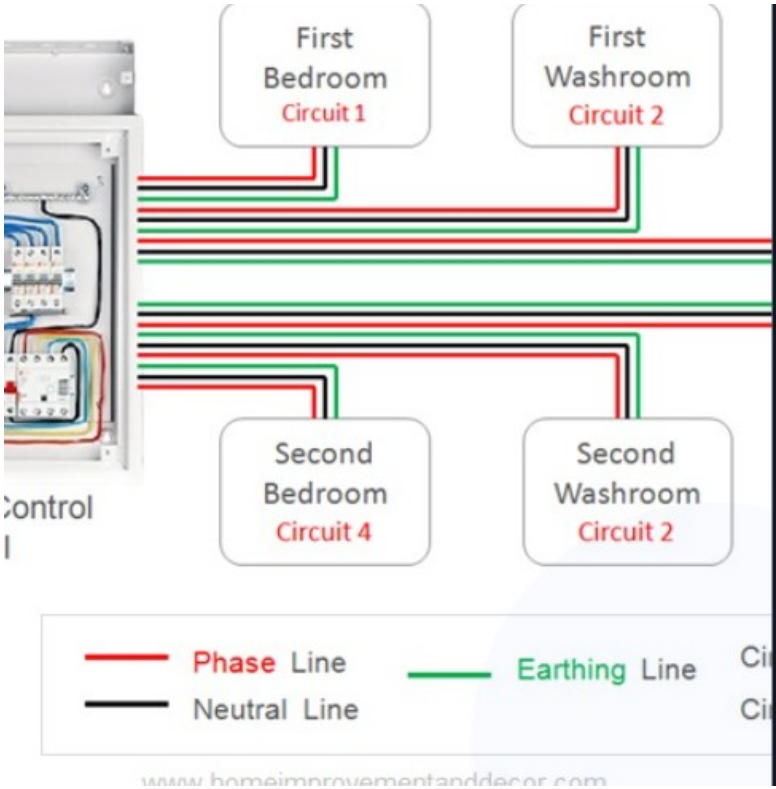
Factory: Encomendar janelas sob medida de uma fábrica. (Imagine que seu sistema precisa enviar notificações por Email ou SMS. Sem uma factory, você teria if/else espalhados pelo código. Com a factory, você separa essa lógica.)
Singleton: O quadro de energia único da casa. (serve para garantir que uma classe tenha **apenas uma única instância** em toda a execução do programa e fornecer um ponto de acesso global para esse objeto)

Area Interna (DESIGN PATTERNS)

2. PADRÕES ESTRUTURAIS

Encanamento e Acabamento

Definem como as partes da casa se conectam para funcionar juntas:



Adapter: O adaptador de tomada. (Interface, Handles)
Facade: O painel de automação central.(fornecer uma **interface simplificada** para um conjunto de interfaces complexas em um subsistema)
Decorator: O papel de parede estético. (Adicionar recursos a objetos em tempo de execução sem afetar outros objetos da mesma classe)

Area Interna (DESIGN PATTERNS)

3. PADRÕES COMPORTAMENTAIS

Automação e Rotina

Tratam da interação, inteligência e o fluxo de comunicação entre os moradores ou módulos da casa.

Observer: O sensor de movimento que avisa o alarme e as luzes. (Subject/Observable, Consumer)
Strategy: O aquecimento que alterna entre gás ou solar conforme o sol. (evitando o uso excessivo de condicionais (if-else ou switch-case))
State: O termostato que reage se a temperatura está alta ou baixa. (Imagine o status de um pedido: Novo, Pago, Enviado, Entregue.)

VLW

Planeje hoje a sua cidade de amanhã

Abrindo o Debate

❓ Questões para reflexão:

1. Qual foi o principal aprendizado para você?
2. Como você pode aplicar isso em seu dia a dia?
3. Quais foram as dúvidas geradas?
4. Que conexões você vê com outros tópicos?

Obrigado!

Até o próximo episódio da BIUD Talks

 Inovando juntos | Crescendo juntos