

Trabalho Final: Arquiteturas de Frontend e Backend

Disciplina: Engenharia de Software

Professor: César Olavo

Tema: Comparação entre MVC, MVP e MVVM no Frontend, integrados a Backends REST e Reativos

Modalidade: Equipe de até 4 membros

Entrega: Código + evidências + relatório curto

Objetivo do Trabalho

Este trabalho tem como objetivo **comparar arquiteturas de frontend** (MVC, MVP e MVVM) e **estilos arquiteturais de backend** (REST tradicional e arquitetura reativa/event-driven), analisando:

- separação de responsabilidades
- impacto da arquitetura do backend sobre o frontend
- esforço de implementação
- clareza do fluxo de dados
- facilidade de manutenção

O foco **não é estética nem complexidade funcional**, mas **arquitetura**.

Escopo Funcional (obrigatório)

Implemente um **ToDo App simples**, contendo no mínimo:

- Listagem de tarefas
- Criação de tarefa
- Exclusão de tarefa

Não é necessário autenticação, paginação ou UI sofisticada!

Parte 1 — Frontend (obrigatório)

O **mesmo ToDo App** deve ser implementado **três vezes**, em **React**, usando:

1. **MVC**
2. **MVP**
3. **MVVM**

Cada implementação deve:

- Ter estrutura de pastas separando responsabilidades
- Conter código funcional (executável)
- Consumir o backend definido na Parte 2
- Obs1: Um componente React já mistura View + lógica, então MVC em React parece um tanto **artificial**. No entanto, isso é bom **pedagogicamente**, pois evidencia os problemas.
- Obs2: **Não** usar Redux, MobX, Zustand, RxJS, etc. O React com seus hooks já está perfeitamente preparado para as implementações necessárias. A arquitetura deve aparecer na organização do código, não na **ferramenta**.

Ferramentas sugeridas (frontend)

É fortemente aconselhado o uso de **ambientes de execução imediata**, como:

- **CodeSandbox**
- **StackBlitz**
- **React local com Vite**

Tais plataformas oferecem setup mínimo e quase nenhum ruído de infraestrutura, permitindo ao aluno montar um ambiente de desenvolvimento React rapidamente e, assim, focar no código.

Parte 2 — Backend (obrigatório)

Utilize **UMA única plataforma de backend**, escolhendo **apenas uma** das opções abaixo:

- **Supabase**
- **Render.com**
- **Railway** (ótimo para protótipos)

Nesta plataforma, o sistema deve oferecer **dois modos de acesso aos dados**:

Backend A — REST tradicional (pull)

- Frontend realiza chamadas HTTP (GET, POST, DELETE)
- Atualizações ocorrem apenas após requisição explícita

Backend B — Reativo / Event-driven (push)

- Frontend recebe atualizações automaticamente
- Exemplos:
 - WebSockets
 - Server-Sent Events (SSE)

Comportamento obrigatório a demonstrar

Quando **duas instâncias do frontend** estiverem abertas ao mesmo tempo:

- No **REST**:
→ a atualização só aparece após nova requisição manual
- No **Reativo**:
→ a atualização aparece automaticamente, sem ação do usuário

Evidências obrigatórias:

Todos os itens abaixo são **obrigatórios**:

1. Prints de tela

- Projeto aberto na IDE / sandbox
- Estrutura de pastas visível
- Código de cada arquitetura (MVC, MVP, MVVM)

2. Vídeo curto (40–60 segundos)

O vídeo deve mostrar:

- duas instâncias da aplicação abertas
- inclusão/exclusão de tarefa em uma instância
- comportamento observado:
 - via REST
 - via backend reativo

3. Relatório técnico curto (máx. 2 páginas)

Responder **com base no código implementado**:

- Onde ficou a maior parte da lógica em cada arquitetura?
- Qual arquitetura foi mais simples de integrar com o backend reativo?
- O que mudou no frontend ao trocar REST por reativo?
- Qual arquitetura você escolheria para um sistema maior? Por quê?

Critérios de Avaliação

Critério	Peso
Implementação correta das 3 arquiteturas	40%
Integração REST + Reativa funcionando	25%
Evidências (prints + vídeo)	20%
Análise crítica no relatório	15%

Observações Importantes

- Códigos genéricos, sem relação com as evidências apresentadas, **não serão aceitos**.

- Respostas teóricas desacopladas da implementação **serão desconsideradas**.
 - O uso de IA é permitido como **ferramenta de apoio**, mas a implementação e análise devem refletir **experiência prática real**.
-