INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ

IFCE CAMPUS FORTALEZA

ENGENHARIA DA COMPUTAÇÃO

MISAEL FERNANDES COSTA

IZADORA DE OLIVEIRA ALBUQUERQUE MONTENEGRO

MARCELO ANTÔNIO DANTAS FILHO

DAVI CARVALHO BARBOSA DE MENEZES

RELATÓRIO: Arquiteturas de Frontend e Backend

CÉSAR OLAVO

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

FORTALEZA

2026

1. INTRODUÇÃO

Este relatório analisa a implementação de um *ToDo App* em React sob as arquiteturas MVC, MVP e MVVM. O objetivo é avaliar a organização da lógica e a facilidade de manutenção ao utilizar dois modelos de backend: REST (*pull*) e Reativo via WebSockets (*push*).

2. DESENVOLVIMENTO

2.1 Onde ficou a maior parte da lógica em cada arquitetura?

A distribuição da lógica variou conforme as premissas de cada padrão, utilizando o padrão Repository para isolar o acesso a dados e evitar a sobrecarga das camadas principais:

- **MVC (Model–View–Controller): A lógica concentrou-se no *Controller*. No React, essa divisão mostrou-se artificial, gerando alto acoplamento, já que os componentes da biblioteca naturalmente integram lógica e renderização.**
- **MVP (Model–View–Presenter): O *Presenter* centralizou a lógica de apresentação. Para evitar o acúmulo de funções de dados, utilizou-se um Repository, mantendo o *Presenter* focado apenas em mediar a View passiva.**
- **MVVM (Model–View–ViewModel): A lógica residiu no *ViewModel*, que se integrou perfeitamente aos *hooks* (useState, useEffect). O uso do Repository aqui foi crucial para abstrair a complexidade das chamadas de API, mantendo o *ViewModel* limpo e escalável.**

2.2 Qual arquitetura foi mais simples de integrar com o backend reativo?

Abaixo, a comparação da integração reativa considerando o suporte da camada de repositório:

Arquitetura	Facilidade	Observação (Uso de Repository)
MVVM	Alta	O ViewModel e o Repository gerenciaram os eventos de forma fluida e reativa.
MVP	Média	O Repository ajudou a isolar a lógica, mas o Presenter exigiu mais código para sincronia.

MVC	**Baixa**	**Mesmo com Repository, o Controller acumulou muitas responsabilidades de fluxo.**

2.3 O que mudou no frontend ao trocar REST por reativo?

A transição alterou o comportamento operacional: no modelo REST (Pull), o frontend solicitava dados via requisições explícitas; no modelo Reativo (Push), o backend notifica os clientes via WebSockets. O frontend passou de um estado de "pedido" para um estado de "escuta", permitindo que o estado local reflita mudanças em tempo real sem intervenção manual do usuário.

3. ANÁLISE CRÍTICA E RECOMENDAÇÃO

Para sistemas de grande porte, a arquitetura MVVM com Repository é a escolha ideal. O Repository isola a lógica de dados, enquanto o ViewModel gerencia o estado da interface. Essa combinação reduz o acoplamento, facilita testes unitários e alinha-se perfeitamente ao funcionamento nativo do React, garantindo que o sistema cresça de forma sustentável.

4. CONSIDERAÇÕES FINAIS

A prática demonstrou que o padrão MVVM, apoiado por uma camada de dados (Repository), potencializa a robustez do React. A integração com backends reativos reforça a necessidade de arquiteturas que tratem o estado de forma centralizada, sendo esta a abordagem mais eficiente para aplicações modernas e distribuídas.