🤝 Simplificando o Front-end com BFF

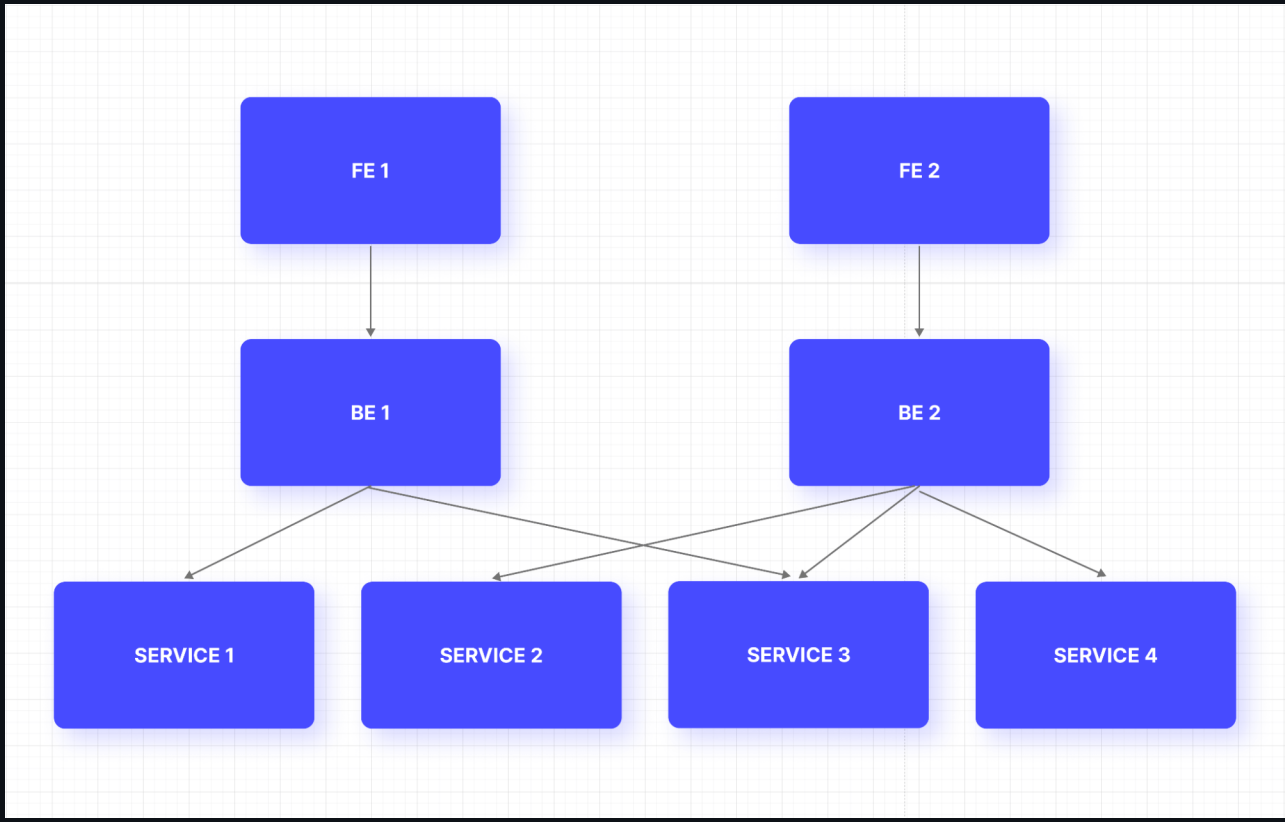
Backend For Frontend

📌 Tópicos

- 🤔 O que é BFF?
- 🚀 Quais problemas resolve?
- 👍 Vantagens
- 👎 Desvantagens
- 🏛️ Quem usa?
- ✨ Conclusão
- 📚 Bibliografia

🤔 O que é BFF?

- Definição básica de BFF e sua função principal.
- Como o BFF atua como uma camada intermediária entre o front-end e o back-end.
- Destaque para a flexibilidade e personalização oferecidas pelo BFF em relação às necessidades do front-end.



🚀 Quais problemas resolve?

1. Over-fetching e under-fetching de dados
2. Dificuldade de adaptação às necessidades do front-end
3. Complexidade e acoplamento entre front-end e back-end
4. Performance e tempo de resposta

👍 Vantagens

1. Agregação de dados
    - Cada página pode receber apenas os dados dos quais precisa.
2. Normalização e adaptação de dados
    - A página recebe os dados traduzidos, ordenados e condensados (podemos combinar informações para gerar um status de uma operação), na estrutura que precisar.
        - Adeus repetir o uso das mesmas funções em diversas páginas para formatar moedas, datas, CNPJ e CPF e etc;

👍 Vantagens

3. Exposição de endpoints customizados e lógica de negócio no BFF
    - Não precisaríamos de vários dos services no front-end;
    - Poderíamos fazer toda a validação dos dados no BFF, retornando a mensagem de erro que o front deve renderizar em cada caso.
4. Imaginem o quão menores os controllers seriam (isso se não matarmos a maioria) e quantos estados seriam extintos.

👎 Desvantagens

1. Complexidade adicional, uma nova camada
    - De fato é um item novo pra se preocupar (só é preciso um novo serviço de início, mas no futuro pode ser necessário quebrar em mais de um).
    - Além das novas rotas dos serviço, seria necessário implementar a rota no client do serviço, e o endpoint no BFF com todas as tratativas que antes eram feitas no front.
2. Aumento da complexidade da infraestrutura (?)
    - Se fossemos usar para validação de dados enviados para o back, precisaríamos hospedar esse serviço no Brasil, para que a latência seja mais baixa.

👎 Desvantagens

3. Potencial SPOF (Single Point Of Failure)
    - Se o BFF enfrentar problemas ou ficar indisponível, todas as solicitações do front-end aos back-ends serão afetadas. Dá-lhe monitoramento para minimizar o impacto de possíveis falhas.
4. Introdução de possíveis gargalos de desempenho
    - Se o BFF não for bem implementado, pode haver a introdução de possíveis gargalos de desempenho (mas aqui seria só uma transferência de um problema que já pode ocorrer no front-end diretamente).

🏛️ Quem usa?

Usado por

- Netflix
- Soundcloud
- Flickr

Recomendado por

- Microsoft
- IBM

✨ Conclusão

- Implementar BFF pode simplificar muito o front, removendo muita lógica do mesmo, e deixando-o focado em uma boa experiência para o usuário.
- Deixa a comunicação do front com o back mais otimizada.
- Desacopla ainda mais o front do back e permite movimentações mais rápidas no front (como trocar um framework).

Mas...

- Introduz um item a mais pra se preocupar;
- Pode ser um SPOF;
- Recapitulação dos principais pontos abordados na apresentação.

Desafios e considerações

- Discussão sobre os possíveis desafios e considerações ao implementar o BFF, como a complexidade da arquitetura.
- Sugestão de boas práticas para superar esses desafios e maximizar os benefícios do BFF.

📚 Bibliografia

- Pattern: Backends For Frontends
- The BFF Pattern (Backend for Frontend): An Introduction
- Why big companies and rapidly growing startups need Back-end for Front-end
- Make microservices more efficient & scalable with Backend For Frontend: Qiwa tech study