Frontend in Microservice Architecture

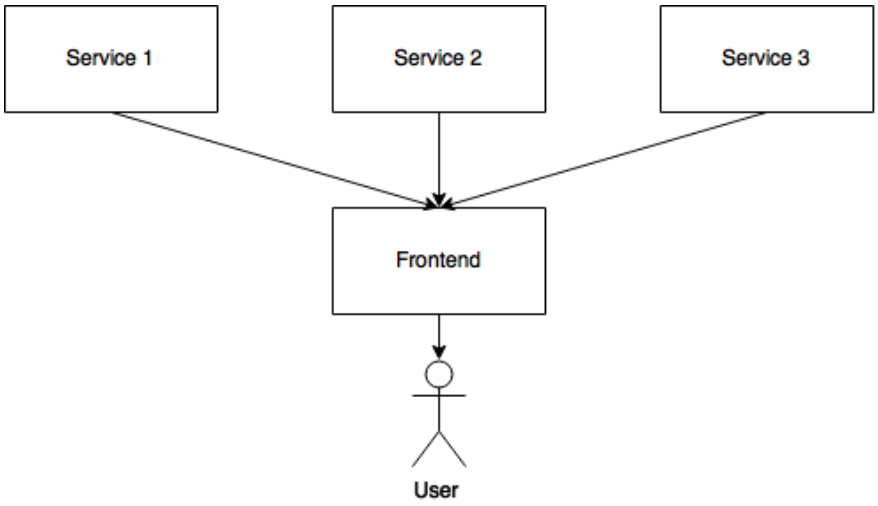
Vivek Madurai Follow
Mar 9, 2018 · 3 min read

Building a Microservice based architecture these days seems to make sense for a lot of big and distributed applications. This style of architecture enables the monolith team to get split to independent team which helps in improving things like scalability, complexity of the code, rolling out to production etc.,

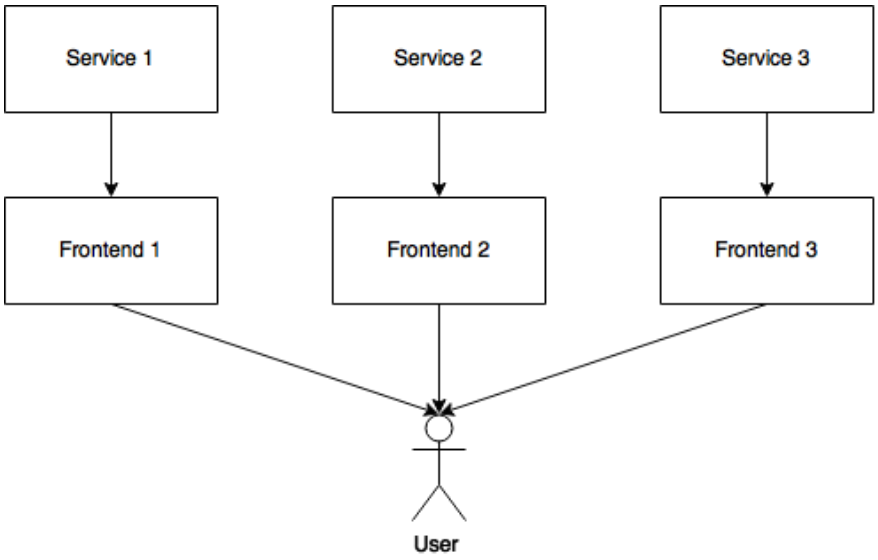
We normally exclude frontend application from these microservice architecture. In this article I will be sharing my thoughts on different frontend approaches in microservice architecture.

- Monolith Frontend
- Modular/Component based Frontend
- Micro Frontend

Monolith Frontend, normally we will be building frontend by a separate team which grows and maintained separately from that of microservice backend which is referred as Frontend Monolith. We used to call this approach as Model View Controller(MVC) architecture.



Micro Frontend, which will be part of the same individual team who builds microservice backend. Each team will own a unique business component and develops its end to end starting from user interface to database layer.



Choosing Micro frontend architecture, there are several benefits in applying micro frontend to our application like,

- Technology agnostic
- Moving integration part of frontend
- Isolated team for development and deployment
- Individual testing and less regression issues.

To know more about building micro frontend apps please read Tom's blog on microservice approach for frontend web development.

Component based Architecture(CBA), this become popular after Facebook released React.js in 2013. In this approach we will break our large user interface into individual self sustaining components. Each components will have their own structure, own method and own API's which intern can be reused inside other components.

We need to be aware of these options before taking a decision on your web application architecture. From the above three micro frontend and component based frontend is the latest and monolith is no more a option.

When to use Micro frontend,

- Migrating legacy app in isolation, while they are still being used. Adding new feature in new framework without disturbing the existing one.

- Helps different frontend frameworks to co-exist(like one in Angular, one in React, Vue, Ember etc.,)

There a some common issues in building micro frontend application like,

- Assets loading(loading libraries), ends up in library conflicts
- Common styling across micro apps
- Upgrading or altering the base framework like (Angular or React)

Component based approach is best suited for application which gets build from the scratch over a single base framework like Angular or React or Vue etc., In this approach we will be building and deploying individual components separately.

Conclusion, Even with clear microservice separation at backend layer its unable to separate our frontend application into multiple entirely separate application. Unlike backend service we cannot build our micro frontend to be completely independent from other micro frontend. We need to maintain a consistent look and feel across all the components and the application should behave as single application.

Although a total split of your application may not be possible, its still possible to have a multiple team working on a separate parts of your application. Instead of splitting your application as pure micro frontend we can still split our application into separate components which can be developed and maintained separately by different teams.