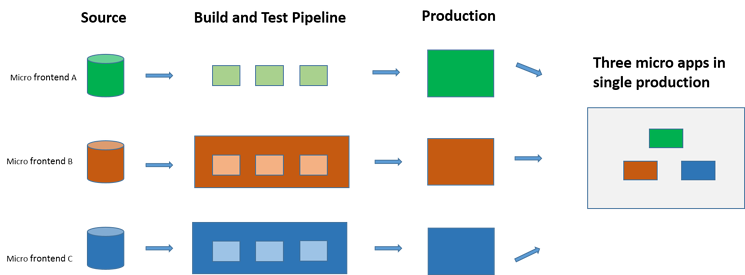
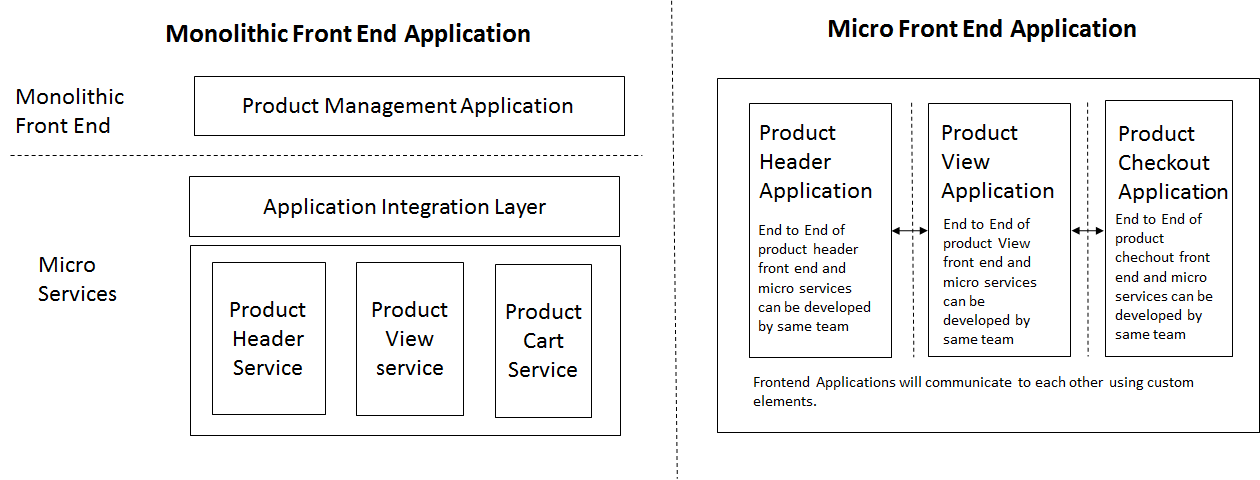
**What is Micro Frontends Architecture?**

Micro frontend is a modular approach to web development wherein an application is broken up by its pages and features, with each feature being owned end-to-end by a single team.

This approach to development is usually confused with web application development using reusable components, but they are different from each other. A reusable, component-based architecture uses multiple pieces of code (components) and then combine them together to create a monolith frontend that sits on the top of backend services. A micro frontend, on the other end, ensures that each feature is developed, deployed, and tested independently from others.

With micro frontends architecture, the codebase is smaller and cohesive and thus is convenient to maintain. Since the features & pages are decoupled and independent teams are working on them, it is comparatively easier (than monoliths) to upgrade, update, and rewrite parts of the application.





**Benefits of Micro Frontend Architecture:**

* **Rapid development:**As different services or features are developed independently and represented as widgets, it cuts down on development time.
* **Speedy deployment:**Since UI and Services can work autonomously, testing (and deployment) of individual UI and Services can be done independently and in parallel, thus, reducing the time required for rollout.
* **Quality assurance:**Parallel testing translates to faster product quality improvements, as individual modules can be released as versions (and improved upon in every next version) without waiting for completion of the entire frontend.
* **Makes regression testing redundant:**As individual components (widgets) are developed separately, they can be tested for dependencies and changes separately. This eliminates the need of resource-intensive regression testing after project-completion. This expedites the frontend’s rollout.
* **Superior maintenance:**The well-defined ownership structure improves collaboration between developers and business teams. This, in turn, speeds up incident response.
* **Business continuity:**In an event of failure in one module, other modules continue to work smoothly—without collateral damage.

**Disadvantages of Micro Frontend:**

* A complex development environment
* Complex Integrations
* Initial down time can increase
* Third party module overlapping / redundancy Composition complexity

**Conclusion:**

We have implemented atomic components in the GB project, these atomic components can be reused in multiple projects having similar UX design. If any other team wants to use same components then they must include that atomic component in their project, which is drawback because same components will be available in multiple project, (e.g. Same atomic components are used in Isseta and GB-CXR project). We can fix this using Micro Frontend.

In Micro Frontend we will create separate repository for Atomic Components, and we can import it in any application as a module using **npm install** and **module name**.