**Understanding of current ecosystem**

The purpose behind the suite of applications is to help lenders to submit/verify the collateral data with government agencies. There are three web applications along with couple of API's which provides required interface to lenders based on type of collateral. While couple of backend services to submit the data to government, to process acknowledgements etc.

**Pain Points**

1. Deployment and configuration management - It is difficult to resolve issues which arises out of misconfiguration and rollouts
2. Duplication of business rules across codebase and services
3. Maintenance of JavaScript code
4. Testability of JavaScript code

**Observations**

1. Different systems interacts over FTP. Though it may work, reliability and extendibility of file based interactions is limited.
2. All the background services are windows based services which are difficult to scale, monitor, and not resilient by nature.
3. XSLT based rule engine may be sufficient to cater the current set of requirements, but may not be maintainable in long term. If the rules does not change often, then it can be better implemented using C#.

**Recommendations**

1. Adopting a Docker based deployment model with automation. This will address configuration issues as well as improve predictability of deployments.
2. Based on nature of business rules they can be implemented with .Net library using patterns like specifications or with rule engines in case business needs a more granular control over them.
3. The three front end applications can be moved under a single umbrella using a single implementation platform like Asp .Net MVC, Angular/React. This will reduce code base, improve modularity and code reuse.
4. Options like Karma or Mocha can be explored for JavaScript unit testing.
5. Moving to a microservice based architecture. The current architecture does provide a good ground to move to microservice based architecture. The services can interact with each other using event based mechanism instead of files or batch processing. This will improve the scalability as well as maintainability of the applications.
6. Microservice based architecture will also enable extending the applications in future and reduce hard coupling of file based interactions.
7. Improve the monitoring and reporting of the applications, processing status etc.