1. **Decision —**Use Microservice for each domain. Details of each domain can be found in topology diagram

**Status —** Proposed

**Context**— Architecture style is critical for any application. Decision is to go for microservice architecture by considering the easiness in maintenance and possibility of elasticity & auto scaling capabilities. Service mesh, Circuit breaker and service discovery needs to be implemented during implementation phase

**Consequences —**Network calls take much longer time than method calls which will negatively impact the performance. Security check at each endpoint will add additional processing time. Microservice architecture favors decoupling

1. **Decision —**Use Elasticsearch for catalog browsing

**Status —**Proposed

**Context**— Application should expose an API for users to browse the catalog. Currently data resides in an RDBMS and it will impact the performance if each service query from the RDBMS for each search

**Consequences —**Data needs to be replicated in the elastic search cluster. Additional cluster & nodes will add on to the maintenance. Eventual consistency can be achieved by adopting high availability and portion tolerance (CAP theorem) . Elasticsearch does not give good search latency if nodes are deployed in multiple regions as it treats all the nodes as if they are in the same datacenter. It does not consider network latency.

However overall performance will be much better than RDBMS

1. **Decision —**Use centralized service for communication with microservices and route all the requests via the same

**Status —**Proposed

**Context**— All the request from API layer will be served by the appropriate microservice. Either the API layer can directly call the microservice or can route via one service. Since there are multiple services and parallel communication is involved proposal is to have centralized service

**Consequences —**This will bring in more granularity. However additional service will add on to maintenance and communication cost

1. **Decision —**REST endpoint for third-party services

**Status —**Proposed

**Context**— As there are multiple third-party services need to interact with the system there should a means for communication. Proposal is to expose REST endpoint for the same

**Consequences —**Contract & authentication check needs to be handled appropriately while designing the REST endpoint