Future State Architecture

NGC Architecture

Future state Next Gen Commerce architecture is designed to build commerce services for commerce & transactional needs of current & future channels . Architecture is designed to provide omni-channel domain services for shopping cart, promo & checkout . Current Identified channels to be onboarded on this architecture are -

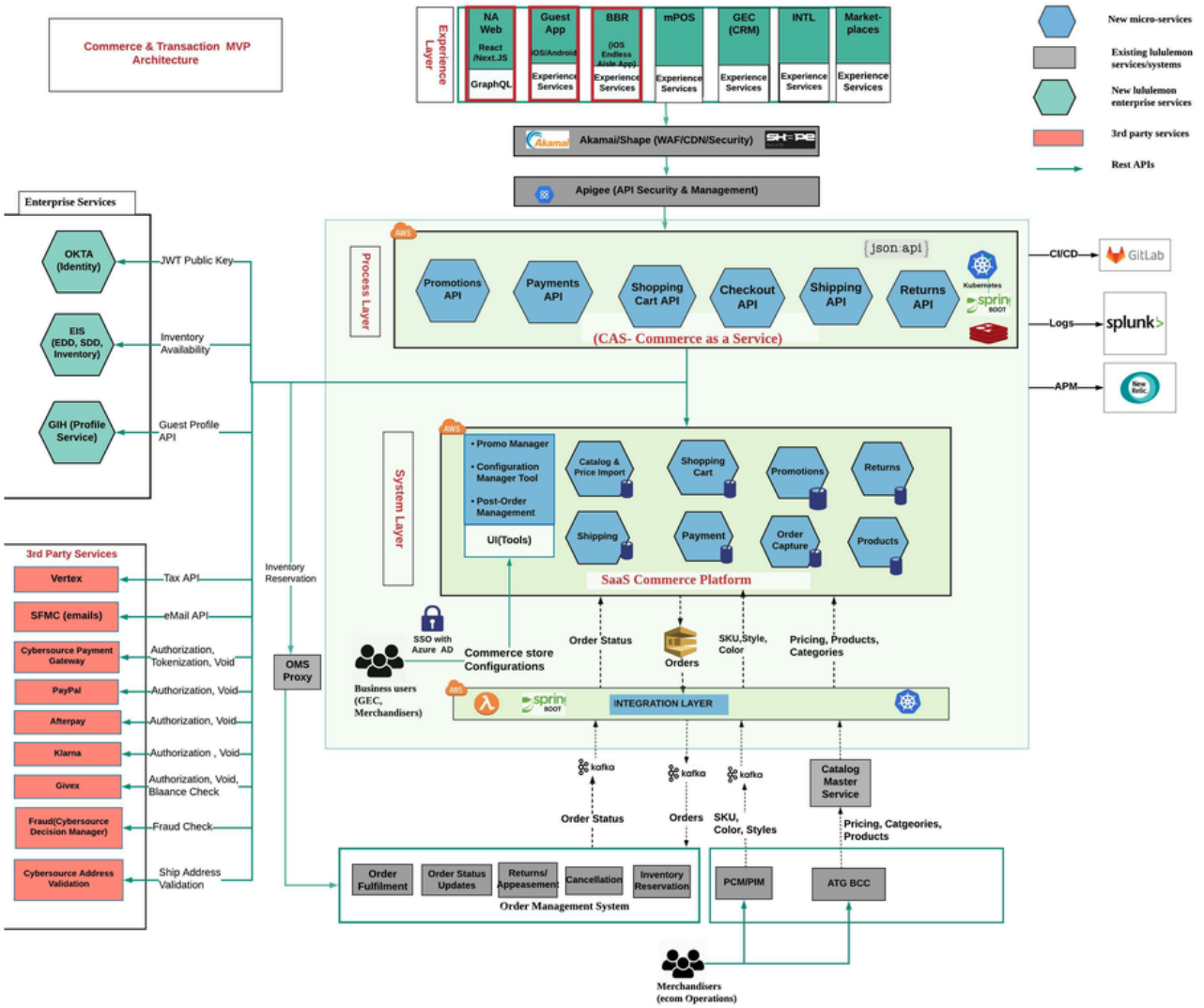
- NA Web
- NA Guest App
- NA BBR
- NA GEC
- NA mPOS

Design Principles

Below design principles are applied in building Next Gen Commerce platform . These principles have been chosen to provide scale needed for a modern commerce platform .

- **MicroServices** - Single pieces of functionality that can be individually designed/deployed, leading to faster development and release cycles and quicker updates. Provides flexibility to easily replace smaller pieces in future.
- **API First** - Enabling the flow of data between microservices and acting as the ‘glue’ to link it all together
- **Cloud Native** **-** For on demand access and unrivaled scalability. Making it easy to expand your reach across regions and channels and provide a better customer experience**.**
- **Headless** **-** The ability to deliver amazing front-end experiences without getting distracted by the back-end architectures. Freeing you up to respond rapidly to changing market conditions.
- **Stateless** **-** Enabling scalability by design and provide consistent guest experience across all channels.

MVP Architecture



Main Architecture Components

Process Layer (CaaS- Commerce as a service)

Process layer is abstraction layer for underlying commerce services. This layer will enable building aggregation of different responses from services like Inventory(EIS), Omni (EDD,SDD) , Profile (GIH), Shopping Cart (Commerce Platform) , legacy platform (ATG) outside core commerce platform . This layer will enable future evolution of commerce platform as we bring more new core commerce services or replace existing ones. All channels will interact with this layer for all commerce functions. This is very essential to hide internal complexities of legacy platform like ATG during migration process . This design will not increase any latency in API performance as some layer has to do a aggregation of all APIs . Instead of hooking Commerce platform with enterprise APIs, this layer will provide more cleaner design and flexibility to replace underlying microservices in future.

System Layer (Core Commerce Platform)

This will be a cloud native multi-tenant SaaS platform which will provide set of microservices for shopping cart, promotions, checkout & order capture. Each microservice will be as per true microservice design & will have individual database providing complete isolation. This platform will provide base commerce APIs , business tooling for promotion & orders , API provisioning & access roles, store specific

configurations , API & Database extensions patterns , integration hooks (lambda, SQS, Pub/Sub, Kafka etc.). Platform will be auto-scalable , available in multi-regions . There will be no platform version upgrades , APIs will be backword compatible .

Integration Layer

Integration layer will be connecting commerce platform with 3rd part services like Vertex, Payments, email etc. Integration layer will also enable data transfer between commerce platform & enterprise platform like OMS, PCM, RMS , BI , external services etc . Integration layer will be a set of microservice which will glue commerce platform with 3rd party services /enterprise systems . This layer will work with best of breed enterprise Integration platforms like Apigee, Kafka, Vault etc.

Engineering Framework