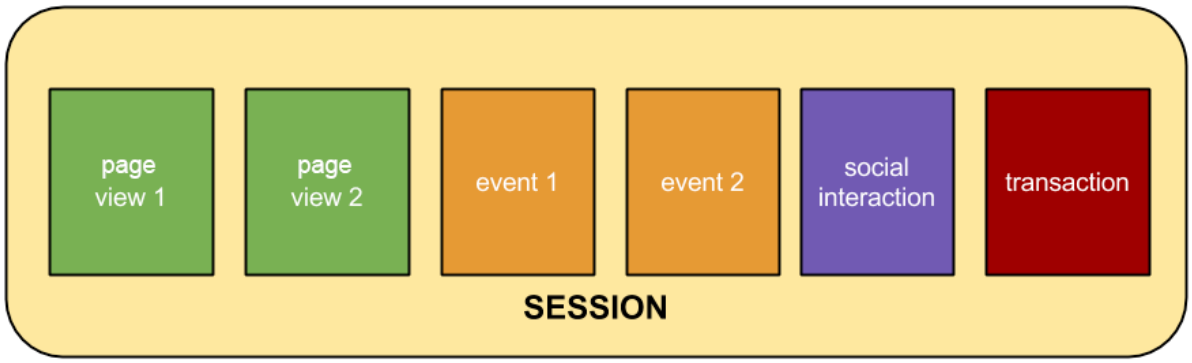
NGC Stateless Architecture & Sessions Management

Problem Statement

As NGC is providing completely stateless services for shopping bag & checkout services , there is a question among several teams how session management would work in future state when different front ends like web and app will integration NGC.

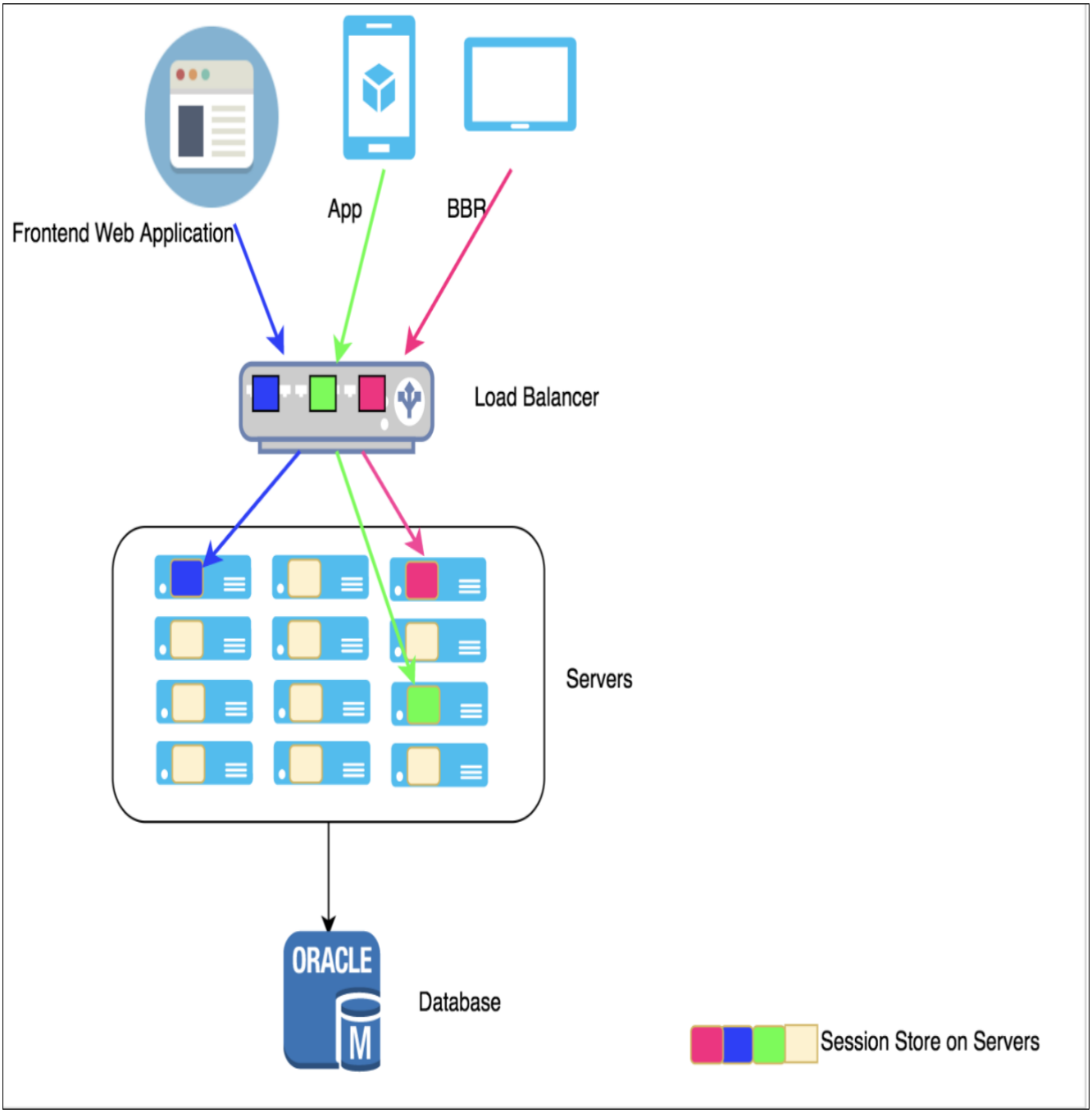
What is a Session ?

A session is a group of user interactions with website that take place within a given time frame. For example a single session can contain multiple page views, events, social interactions, and eCommerce transactions.



In traditional monolithic eCommerce , a session is a server side tech component which binds users interactions across different requests through a unique identifier and keep user state updated throughout the interactions.

e.g. ATG Commerce starts a unique session (**JSessionID**) on first interaction and manage guest state throughout all the interactions within that session.



Different Types of Sessions in lululemon eCommerce

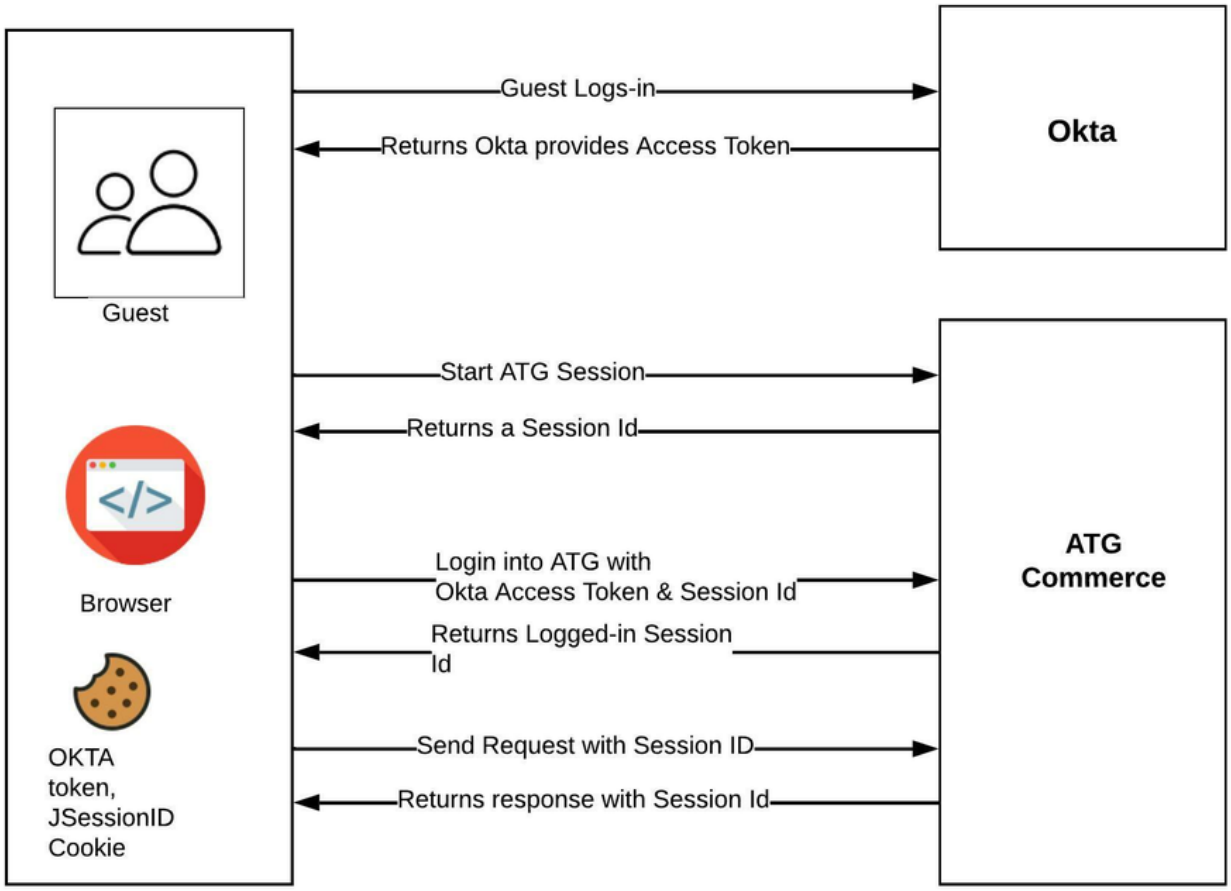
Adobe Analytics Session (ECID)

This Session ID is generated by Adobe Analytics to track guest across different requests . This is stored in **browser cookie** and used to enable personalization across the website. This session ID has a longer expiration time and is not used in transactional activities.

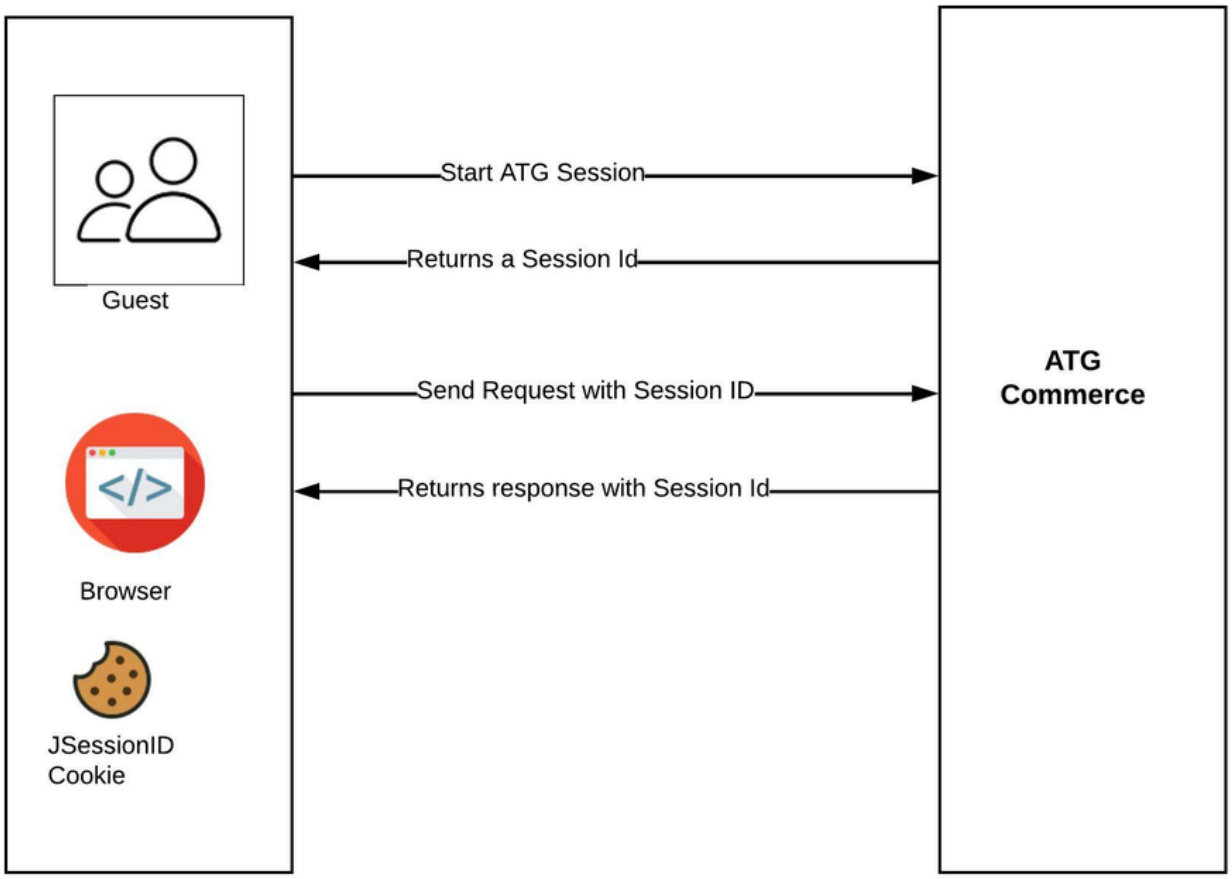
ATG Session ID (JSessionID)

This Session ID is generated by ATG to track guest activities across different requests . This SessionID is stored in **browser cookie** & must for consistent guest experience which are powered by ATG APIs. This is required to be present in all requests sent to ATG for any transactional or non-transactional APIs powered by ATG.

ATG Session has very short duration and it expires after 15 minutes (based on last guest interaction).



Logged-in guest flow with Okta token & ATG sessions



Anonymous guest flow with ATG sessions

What is Statelessness ?

This has been described in details on below confluence page in lululemon context:

Stateless Commerce services Roadmap

Service statelessness is an architecture principle whereby a service that provides a function is decoupled from the state required to perform that function.

This is done by having the state data passed to the service as part of the function request or externalized the state to 3rd party storage.

NGC Architecture Reference

MVP Architecture - NGC

NGC Services Architecture

NGC API Reference

http://ngc-swagger-api-spec.s3-website-us-west-2.amazonaws.com/?urls.primaryName=Carts - Internal

Key NGC services principles on statelessness

Token Based AuthN & AuthZ

- NGC services relies on Token based **authentication** & **authorization** for each API request. NGC does not have any concept of a session .
- There is no state remembered by NGC between two requests.
- Tokens are only way to pass claims between different micro-services eg. **NGC** , **GIH**, **OKTA** , **EDD**, **Membership** , **Wishlist** etc . Tokens are the binding mechanism among different micro-services in organization which is a proven way for resilient & scalable micro-service architecture.

OAuth Tokens

- NGC services use OKTA access token for **logged-in** guests & Apigee access Token for **non logged-in** guests. These are OAuth tokens and provide fine grained security for each API.

Ephemeral Servers

- There is no state stored on any computation server. Each server is ephemeral and stateless. This allows any request to be served from any computation instance .
- Same architecture leads towards **ephemeral environments** in NGC.

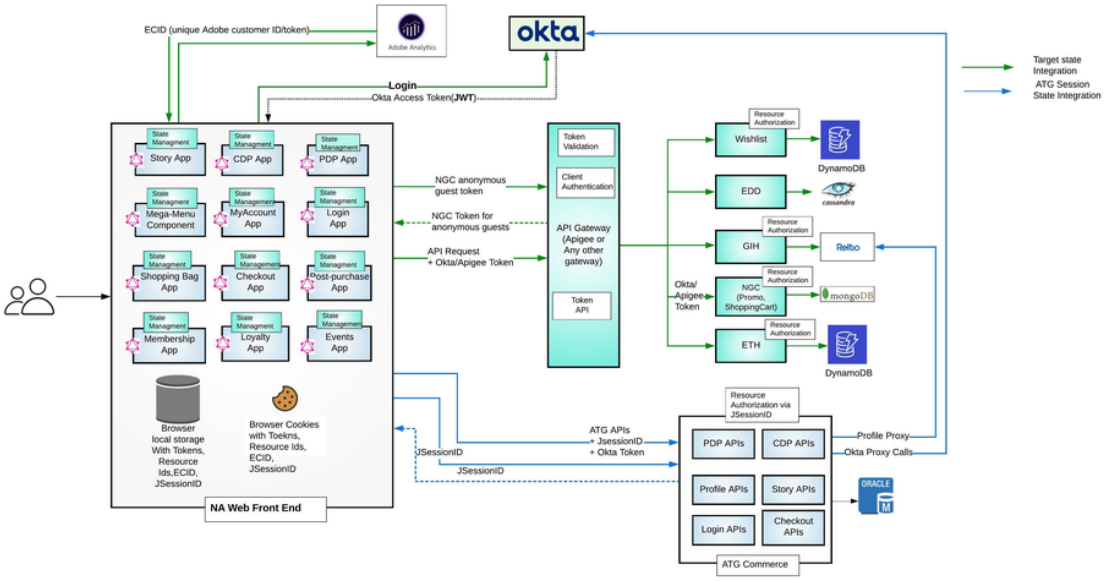
Externalized State

- Any storage is externalized to MongoDB and platform uses CQRS pattern to separate out write & reads . This allows APIs to be extremely performant (~100ms response time) even in stateless architecture.

Rest Standards

- Rest mandates Services to be stateless by design.
- APIs follows REST standards which completely states about APIs to be stateless.
- APIs follows Idempotency as per REST standards .

Proposed NGC Token based Integration with Web



NGC-Web Integration

Where State would be managed in future shopping experience

Stateless architecture leans heavily on consumers of APIs like Web , App to maintain state between requests. NGC architecture on same line will rely on web and app layers to manage state between requests , orchestrate among different services based on state managed locally to them.

Example:

Cart operations - Restore and Merge

Above example shows how Web experience team will have to manage state between guest transitions which was handled by implicitly by ATG in current eCommerce.

To achieve scalability, resiliency & stability in modern micro-service architecture , experience layers will have to identify patterns and practices to manage state between different apps .

Will ATG Session still be there after NGC ?

ATG session will not exist for Shopping Bag & Checkout apps. But Any other app in eco-system which interact with ATG APIs would have to rely on ATG Session management for APIs other than checkout.

Discoveries NGC teams is doing with web and app teams on State Management

- Guest App & BBR team is comfortable managing state at their end as that has been the principle of mobile app engineering practices from beginning.
- Mobile app teams are doing state management to an extent even in current state and less affected by ATG Session Management overall.
- Web checkout teams are aligned with managing state as per new NGC APIs . Some discovery work happening and under active discovery as of now is -
  - How different apps like PDP , Checkout, Post-purchase components will share state among them ?
  - How new sequence design looks from GraphQL/Experience services in new world when every operation is explicit & every NGC request does a single operation.

- How shopping cart will be accessed on different components like header, PDP buy , shopping bag etc without having a global cookie managed by NGC.