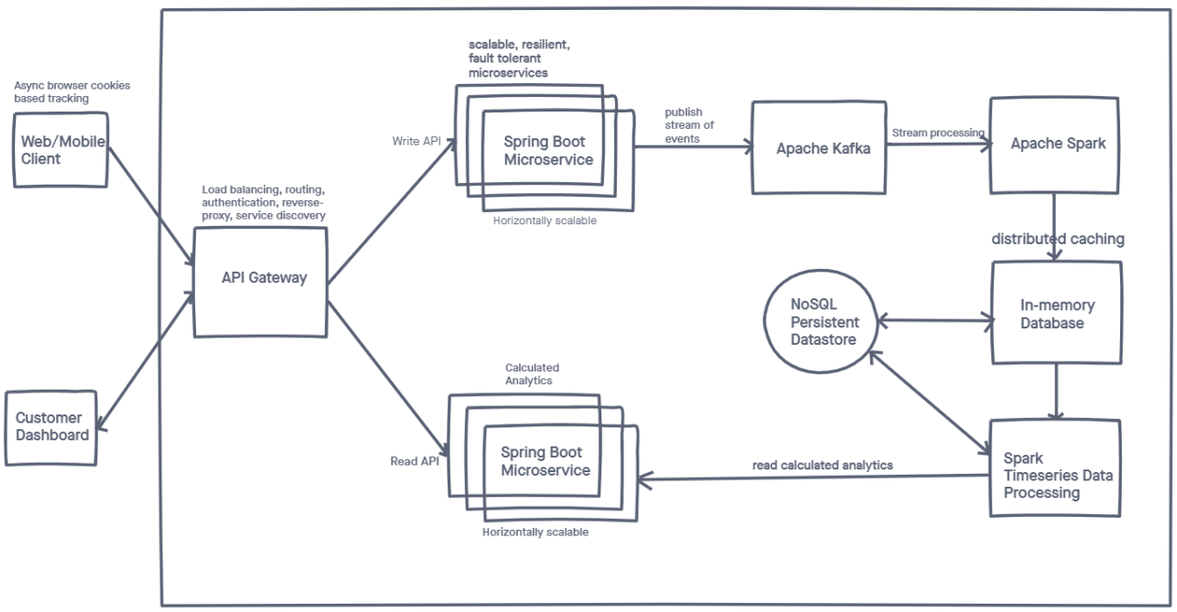
1) Architecture Diagram – Google Analytics Backend



2) System Description

- Google Analytics system tracks user analytics and captures in the system so that customers can view analytics built on top of that. As shown in architecture above, client (mobile, web or tablet) will be sending tracking information to API Gateway. API Gateway do the necessary administration activities like authentication, load balancing, service discovery and routing to correct microservice (i.e., Write Microservice).
- Write Microservice will capture the information and forward that to Apache Kafka. Apache Kafka need to have topics configured already on which this data can be received. Apache Spark will subscribe to these topics and will do the necessary transformation to process the information. This processed and transformed events will be fed into In-memory distributed cache which can be a write-back cache. It will cache the recent published tracking events that can be consumed by Spark timeseries processing engine.
- In-memory distributed cache will write the data to the data store lazily (i.e., upon cache eviction or cache update)
- Spark timeseries processing engine will be producing real-time analytics. Also, it can create historical analytics as well by reading data from NoSQL datastore
- When user requests analytics reports, Read Microservice will read the information produced by Apache Spark data processing component and via Read microservice it will be shown to the user dashboard.

3) Component Description

Web/Mobile Client

- User will be tracked based on browser cookies assigned by async JavaScript script

API Gateway

- API Gateway is a system component that is the entry point in the system.
- It can provide load balancing aspect, which is to route traffic to appropriate handlers based on request
- It can act as a reverse proxy to hide the underlying API interaction
- It takes care of authentication aspect
- It also discovers the service-by-service discovery, service discovery may require additional plugins to be integrated based on the API gateway chosen for implementation
- For e.g., Spring Cloud API Gateway and Apigee are popular API Gateways that can be used

Write API/Spring Boot Microservice

- Write API will record the statistics in the platform
- Spring Boot provides out of the box support for microservices creation real fast
- Microservices are scalable, resilient, fault-tolerant and highly available
- Individual microservices can further be deployed using Docker containers in Docker Swarm or Kubernetes cluster

Apache Kafka

- Apache Kafka is real-time streaming platform that helps to create efficient data pipelines
- Here, we'll be publishing user tracking events to Kafka that it will stream for consumption
- Consumers can subscribe to these topics for processing the events

- Kafka is highly scalable, fault-tolerant and very mature messaging system

Apache Spark

- Spark enables processing of live data streams
- Spark is a perfect choice for the architecture because it provides high performance for both batch and streaming data

Distributed Cache/In memory data store

- To provide high performance for calculating analytics, an in-memory data store is a wise choice
- A write-back cache policy can be adopted to make sure as soon as an event is read, it's kept in cache and then lazily updated in the data-store
- Main advantage of having this in memory data store is to reduce database calls to read the tracking information and also, in case of Spark failover Analytics engine still can be resilient to calculate the analytics
- For e.g., Redis, Apache Ignite, Hazelcast are some of the distributed caches that can be used for this purpose

Persistent Data store

- To store user tracking information a NoSQL database is wise choice as it allows the flexibility to store data in human readable format (i.e., JSON)
- Also, with rapidly changing business requirements, NoSQL databases are capable of adopting the change as they are scalable and flexible
- For e.g., Apache Cassandra, MongoDB are some of the best suited databases for Enterprise scale usage

Spark Timeseries Data Processing

- Calculates and stores the real time analytics for user presentation
- Also, calculate on demand historical analytics from the data store

User Dashboard

- There can be HTML5 based modern analytics dashboard to present the web-analytics to the customers for their website usage or some tools like Tableau which has good market presence in such domain can be used as well