**Software Design Specification**

The software system has the following components and technologies:

- API Gateway – Google Cloud API gateway service

- Message Queue for communication - Kafka

- Kubernetes on Cloud – Google Cloud

- Persistence – MongoDB and Cassandra

- Distributed Tracing - Zipkin

- Elastic Search -

- Caching

- Secrete Management – Google Cloud service

- Security – JWT token

Here are the technologies we used to implement the above features (components):

1. We used Google Cloud to deploy our applications because:

- it provides multiple cluster/instance implementation of Kubernetes

- it helps to scale our system up and down easily with a cost effective way.

- provides API Gateway and secrete management services

2 - The API Gateway we used on Google Cloud receives all calls to the system and routes to the Authentication/Authorization service. It makes integration simpler than using other service.

3 - For communication in the system, we used Kafka as a message queue service. No direct service to service communication is allowed in our system.

- Kafka is fast and helps to decouple data streams using different topics

- It supports multiple producer and consumer services – so no blocking in the system

- It provides easy message/queue retention record feature – It will be very important for the system to clear used or expired messages from Kafka – to clean expired events from Kafka control their lifetime on the Kafka.

4 – For persistence we used MongoDB and Cassandra databases

- MongoDB is used in the Notification service since it is flexible, scalable and easy – and we don’t have much to persist in this service.

- Other services (Authentication, Trip Registration and Reservation services), we used Cassandra database.

- It is highly resilient and fault tolerant – very important in high traffic times

- No single point of failure – suitable for the Kafka event based system

- Its rapid writing and fast reading capability will make our system fast and reliable.

- It is highly scalable and can be used as a huge data store with excellent resilient – very suitable for high traffic seasons.

5 – For Tracing – we used Zipkin for its simpler integration with Spring Boot.

6 – Secrete Management

- Google Cloud provides us a good secrete management service, so we used it to store all our secrets there (service names, Kafka topic names, Zookeeper configuration parameters, ports)

7 – Security

- JWT token for every communication in our system. It is secured, reliable and also lightweight.

8 – Caching

9 – Elastic Search

10 – Service Discovery – Google Cloud

11 – Containerization – docker/ docker hub

**Faced Problem**

- Deployment - managing secretes and deployment configurations for some services.

- Communication through Kafka – serializing and deserializing

- Cassandra database/keyspace set up – mapping objects to database entities.

- Elastic search and cashing

**Conclusion**

We managed to decouple our microservices using Kafka, secured with JWT, managed our secretes, service discovery and API Gateway using Google Cloud service. Our microservices are dockerized/containerized and deployed on Google Cloud.