Design E-Commerce application using microservices architecture and single page application

**Problem Statement**

E-commerce provides an easy way to sell products to a large customer base. However, there is a lot of competition among multiple e-commerce sites. When users land on an e-commerce site, they expect to find what they are looking for quickly and easily. Also, users are not sure about the brands or the actual products they want to purchase. They have a very broad idea about what they want to buy. Many customers nowadays search for their products on Google rather than visiting specific e-commerce sites. They believe that Google will take them to the e-commerce sites that have their product.

The purpose of any e-commerce website is to help customers narrow down their broad ideas and enable them to finalize the products they want to purchase. For example, suppose a customer is interested in purchasing a mobile. His or her search for a mobile should list mobile brands, operating systems on mobiles, screen size of mobiles, and all other features as facets. As the customer selects more and more features or options from the facets provided, the search narrows down to a small list of mobiles that suit his or her choice. If the list is small enough and the customer likes one of the mobiles listed, he or she will make the purchase.

The challenge is also that each category will have a different set of facets to be displayed. For example, searching for books should display their format, as in paperpack or hardcover, author name, book series, language, and other facets related to books. These facets are different from other products. Similarly, each category will have different facets and it needs to be designed properly so that customers can narrow down to their preferred products, irrespective of the category they are looking into.

## Benefits of microservices for eCommerce

From the eCommerce perspective, microservices are the ultimate way to prepare eCommerce platforms for traffic peaks, as well as implement and test out the latest trends, like new payment methods, voice assistants or [Progressive Web Apps](https://divante.com/blog/category/progressive-web-apps/), without significant risks.

Most of the newly introduced solutions are built using the headless approach and fit and match perfectly with systems built using microservices architecture.

Microservices can also be a way of setting up complex omnichannel systems. To fulfill your customers’ expectations about omnichannel, we have to integrate each and every piece of information about products, shipments, stocks and orders, and keep it all up to date. And although there are some one-stop solutions, microservices enable companies to use API gateways to integrate POS, ERP or WMS solutions that are best-in-category, and synchronize them with existing processes.

One of the biggest [benefits of microservices](https://divante.com/blog/5-key-advantages-microservices-approach-business/) is that each microservice can be developed, scaled, and deployed separately. In the case of eCommerce, and not only, it means that if your online store needs any changes in payment, then you apply and deploy these changes only on the microservice dedicated to payments and don’t have to re-deploy the other parts of the system. Thanks to this, microservice architecture cuts costs and time-to-market, allows for quicker release cycles, and also encourages innovations.

Single microservices are maintained by independent, specialized teams. That’s why microservice architecture, along with technology, strongly relies on people and processes within an organization.

**Technical Stack**

Identify Provider

* Cognito - Amazon Cognito lets you add user sign-up, sign-in, and access control to your web and mobile apps quickly and easily

Development - UI and Backend Services

* **React/Angular-(UI)**
* **Java 12**
* **Spring Boot** - version 2.1.5 RELEASE
* **Spring Web MVC** - version 5.1.7 RELEASE
* **Spring SESSION** - version 2.1.6 RELEASE
* **Spring Cloud - Open Feign** - version 2.1.1 RELEASE
* **Netflix Zuul** - version 1.3.1
* **Netflix Eureka Client/Server** - version 2.1.1
* **Netflix Ribbon** - version 2.3.0
* **Redis Client : JEDIS** - version 2.9.3
* **Spring Data REDIS** - version 2.1.8 RELEASE
* **Spring Data JPA** - version 2.1.8 RELEASE
* **Hibernate** - version 5.0.4 Finale

Database Development

* **MYSQL Database-MongoDB**
* **NOSQL Database engine** : Redis 3.2.100

Deployment

* Docker-Kubernates
* AWS-EC2
* Jenkins

Testing Strategies

* Junit
* Jest
* Karma
* Selenium

Log monitoring

* Kibana

<https://github.com/venkataravuri/e-commerce-microservices-sample>

https://github.com/idugalic/micro-ecommerce

https://www.skava.com/ecommerce-microservices-for-innovation/

###Functional Microservices

* Search Services
* **Product Catalog Microservice**
* **Cart Microservice**
* Order management service
* Inventory service
* Pricing Service
* Rating Service
* Recommendation Service
* Reporting

###Infrastructure Microservices

* **Global Configuration Microservice**
* **Service Registration & Discovery Miroservice**
* API Gateway