Text

Description automatically generated

**CS 590-Software Architecture**

**MIU University**

Logo

Description automatically generated

**This project Handed to:**

Prof. Anthony Sander

**Date Handed:**

22nd October 2021

**Group Members Name:**

Ghidiom Nugusse, #612729

Samuael Melake, #612798

Yafiet Weldegarit, #612758

Meron Tedros, #612760

Final Project Documentation

System Design:

Diagram

Description automatically generated

Business Logic:

A Client (without registering to the system) can see all the products and add them to shopping cart. When the client wants to place an order, he/she needs to register with the system. In addition to that the customer can update the shopping cart, but if he/she want to add extra products there will be a communication with the product via the Feign client. However, there are some sub processes the application needs to execute before confirming the final shopping cart requested by the client. For instance, the application checks if there is enough quantity of products requested by the client in the repository. Hence, if the is not enough number of products available, it will notify him/her that there is not enough supply of products available within our service and recommend on making acceptable orders. Furthermore, when a client places an order, the customer id is attached with the order and the system directly notifies the client through email address.

Meeting function and NFR:

**Customer Service**

|  |  |  |
| --- | --- | --- |
|  | Method | End Point |
| 1. Customer-POST-Add-New | POST | 35.201.79.214/customer/add |
| 1. Customer-UPDATE | PUT | 35.201.79.214/customer/update |
| 1. Customer-GET-By-Id | GET | 35.201.79.214/customer/get/{id} |

**Product Service**

|  |  |  |
| --- | --- | --- |
|  | Method | End Point |
| 1. Product-POST-Add-New | POST | http://35.201.79.214/product/add |
| 1. Product-GET-By-Id | GET | 35.201.79.214/product/get/{id} |
| 1. Product-UPDATE | POST | 35.201.79.214/product/update |
| 1. Product-GET-Quantity-By-Id | GET | 35.201.79.214/product/getQuantity/{id} |

**Shopping Cart Service**

|  |  |  |
| --- | --- | --- |
|  | Method | End Point |
| 1. ShoppingCart-POST-Creat Cart | POST | localhost:8085/customer/add |
| 1. ShoppingCart-POST-Add -Product-to-Cart | POST | 35.201.79.214/shoppingCart/addProduct/{CartId} |
| 1. ShoppingCart-POST-Change Product Quantity | POST | 35.201.79.214/shoppingCart/changeQuantity/{cartId}  /{prodId}/{quantity} |
| 1. ShoppingCart-POST-Delete Product | POST | 35.201.79.214/shoppingCart/removeProduct/{cartId} |
| 1. ShoppingCart-POST-CheckOut | POST | 35.201.79.214/shoppingCart/checkOut/  {cartId}/{Quantiry} |
| 1. ShoppingCart-GET-CheckOut-Cart | GET | 35.201.79.214/shoppingCart/get |

**Order Service**

|  |  |  |
| --- | --- | --- |
|  | Method | End Point |
| 1. ShoppingCart-GET-CheckOut-Cart | GET | 35.201.79.214/shoppingCart/get |
| 1. Order-getAll-Orders | GET | 35.201.79.214/order/getorder |
| 1. Order-getOrder-By-ID | GET | 35.201.79.214/order/getorder/{Id} |

What we will do different:

1. Redis: for session, shopping-cart
2. Elastic Search: for searching product using any related data of a product
3. Security: between microservices and JWT for client side
4. Add shipping, Product-Review and payment microservices
5. Add CI-CD: Jenkins

Bonus:

* We implemented Server mesh (Isto).
* We implemented Zookeeper

Technology List:

Those are the list of technology that we have used to develop the system.

* Spring boot frameworks
* Kafka
* Docker
* Kubernetes
  + Ingress
  + Config Map
  + Secret management
  + Services, Pod, Node, and cluster
* GCP platform
* Mongo database
* Feign Client

Source Code Link: