Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

*Факультет Программной инженерии и компьютерной техники*

*Дисциплина «Сервис-ориентированная архитектура»*

**Лабораторная работа №3**

Группа: P3417

Выполнил: Ореховский А.

Проверил: Усков И. В.

Санкт-Петербург,

2021г

# Задание

По очевидным причинам вариант на данное задание я выбрал сам. Тем не менее, я выбрал максимально разнообразный вариант, в котором мне пришлось выполнить все базовые действия, которые, как мне кажется, ожидались по выполнению. Номер варианта: 1706.15.1

Переработать веб-сервисы из лабораторной работы #2 таким образом, чтобы они реализовывали основные концепции микросервисной архитектуры. Для этого внести в оба сервиса -- "вызываемый" (из лабораторной работы #1) и "вызывающий" (добавленный в лабораторной работе #2) перечисленные ниже изменения.

Изменения в "вызываемом" сервисе:

* Переработать сервис, обеспечив возможность его развёртывания на платформе Spring Cloud.
* Использовать все основные компоненты архитектуры Spring Cloud: Config Service, Load Balancer, Service Discovery.
* В качестве Service Discovery использовать Eureka.
* В качестве балансировщика нагрузки использовать Ribbon.
* Все запросы к сервисам должны прокироваться через API Gateway. В качестве API Gateway использовать Zuul Proxy.

Изменения в "вызывающем" сервисе:

* Разделить приложение на два модуля -- веб-приложение с веб-сервисом и EJB-jar с бизнес-компонентами.
* Переместить всю логику из класса сервиса в Stateless EJB. В классе сервиса оставить только обращение к методам бизнес-интерфейса. EJB-компонент должен быть доступен удалённо (иметь Remote-интерфейс).
* Сформировать на уровне сервера приложений пул компонентов EJB настраиваемой мощности, динамически расширяемый при увеличении нагрузки.
* Настроить второй экземпляр сервера приложений на другом порту, "поднять" на нём вторую копию веб-сервиса и пула EJB.
* Настроить балансировку нагрузки на оба запущенных узла через Haproxy.

Оба веб-сервиса и клиентское приложение должны сохранить полную совместимость с API, реализованными в рамках предыдущих лабораторных работ.

# Листинг кода

Код доступен по ссылке: <https://github.com/sempaw/ifmo.soa.lab3>

# Доступ к приложению

Для корректной работы приложения необходимо пробросить порты:

> ssh -L 7143:localhost:7143 -L 6781:localhost:6781 s225123@helios.cs.ifmo.ru -p 2222

Клиент первого сервиса доступен по ссылке: <https://se.ifmo.ru/~s225123/service1/>

Клиент второго сервиса доступен по ссылке: <https://se.ifmo.ru/~s225123/service2/>

# Итоги

По итогам данной лабораторной работы я преобразовал сервисы в микро-сервисном стиле.