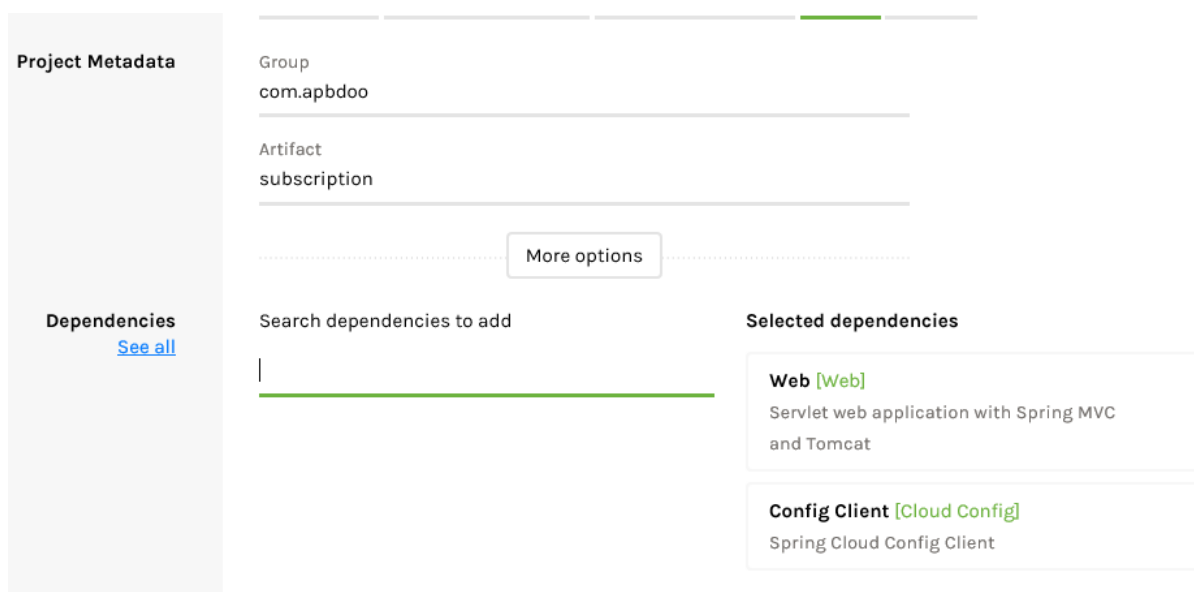


## Laborator 15: RESTful Webservices Ribbon & Eureka

1. Folosind start.spring.io creați un proiect cu Artifact Id: subscription. Adăugați dependențele: Web și ConfigClient



**Project Metadata**

Group  
com.apbdoo

Artifact  
subscription

[More options](#)

**Dependencies**  
[See all](#)

Search dependencies to add

**Selected dependencies**

- Web [Web]**  
Servlet web application with Spring MVC and Tomcat
- Config Client [Cloud Config]**  
Spring Cloud Config Client

2. Adăugați două proprietăți în fișierul application.properties, configurați numele aplicației și portul pe care va rula aceasta:

```
spring.application.name=subscription
server.port=8081
```

3. Creați serviciul pentru requesturi de tipul:

```
http://localhost:8081/subscription/coach/{coach}/sport/{sport}
```

care să întoarcă un obiect de tipul json cu următorul format:

```
{"id":1,"coach":"James","sport":"tennis","price":200}
```

4. Adăugați în fișierul application.properties proprietatea instance.id:

```
instance.id = 1
```

5. Adăugați în clasa Subscription atributul instanceId:

```
@Column  
private Integer instanceId;
```

6. Folosiți un obiect de tipul org.springframework.core.env.Environment pentru a completa instanceId al obiectului returnat de metoda findByCoachAndSport a clasei SubscriptionController.

```
@Autowired  
Environment environment;  
...  
subscription.setInstanceId(Integer.parseInt(environment.getProperty("instance.id")));
```

7. Rulați aplicația pe portul 8082 cu opțiunea instance.id = 2. În run configuration adăugați VM options:

```
-Dinstance.id=2 -Dserver.port=8082
```

8. În proiectul pricecalculator, folosind RestTemplate modificați metoda calculateByCoachAndSport astfel încât să utilizeze serviciul subscription.

```
ResponseEntity<SubscriptionPrice> responseEntity = new  
RestTemplate().getForEntity("http://localhost:8082/subscription/c  
oach/{coach}/sport/{sport}", SubscriptionPrice.class)
```

9. Adăugați în fișierul pom al proiectului pricecalculator dependența pentru clientul feign:

<https://www.baeldung.com/intro-to-feign>

```
<dependency>  
    <groupId>org.springframework.cloud</groupId>  
    <artifactId>spring-cloud-starter-feign</artifactId>  
    <version>1.4.6.RELEASE</version>  
</dependency>
```

**10.** În clasa SubscriptionApplication adăugați adnotarea:

```
@EnableFeignClients("com.apbdoo.subscription")
```

**11.** Creați interfața (proxy pentru comunicarea cu serviciul subscription):

```
@FeignClient(name = "subscription", url = "localhost:8082")
public interface SubscriptionServiceProxy {

}
```

**12.** Adăugați în interfața SubscriptionServiceProxy metoda findByCoachAndSport.

```
@GetMapping("/subscription/coach/{coach}/sport/{sport}")
SubscriptionPrice findByCoachAndSport(
    @PathVariable String coach,
    @PathVariable String sport);
```

**13.** Adăugați un request mapping care să folosească proxy-ul SubscriptionServiceProxy.

```
calculateByCoachAndSportFeign(@PathVariable String coach,
                              @PathVariable String sport,
                              @PathVariable BigDecimal months){

...
subscriptionPrice =
subscriptionServiceProxy.findByCoachAndSport(coach, sport);
...
}
```

**14.** Testați adresa :

```
http://localhost:8091/subscriptionfeign/coach/James/sport/tennis/
months/3
```

**15.** Adăugați dependența pentru Ribbon.

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-ribbon</artifactId>
    <version>1.4.6.RELEASE</version>
</dependency>
```

**16. Adnotați interfața SubscriptionServiceProxy:**

```
@FeignClient(name = "subscription")
@RibbonClient(name = "subscription")
public interface SubscriptionServiceProxy {
```

**17. Adăugați în fișierul application.properties lista serviciilor subscription disponibile:**

```
subscription.ribbon.listOfServers=http://localhost:8081,
http://localhost:8082
```

**18. Adnotați în clasa SubscriptionPrice atributul:**

```
private Integer instanceId;
```

**19. Porniți două cele două servicii Subscription (port 8081 și 8082). Testați:**

```
http://localhost:8091/subscriptionfeign/coach/James/sport/tennis/months/3
```

**20. Creați un naming-server de tip Eureka:**

The screenshot shows the Spring Boot Admin interface. On the left, there's a sidebar with 'Project Metadata' and 'Dependencies' (with a 'See all' link). The main area shows the configuration for a new project:

- Group:** com.apbdoo
- Artifact:** eureka-namingserver
- More options:** A button to expand the configuration.
- Search dependencies to add:** A search bar with the text 'Web, Security, JPA, Actuator, Devtools...'.
- Selected dependencies:**
  - Eureka Server [Cloud Discovery]:** spring-cloud-netflix Eureka Server
  - Config Client [Cloud Config]:** Spring Cloud Config Client

**21. În proiectul eureka-namingserver adnotați clasa EurekaNamingserverApplication:**

```
@EnableEurekaServer
public class EurekaNamingserverApplication {
```

- 22.** În fișierul `application.properties`, configurați numele aplicației și portul pe care va rula aceasta:

```
spring.application.name=eureka-namingserver
server.port = 8761

eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false
```

- 23.** Porniți și testați aplicația:

<http://localhost:8761/>

- 24.** În fișierul `pom` al proiectului `pricecalculator` adăugați dependența:

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-eureka</artifactId>
  <version>1.4.6.RELEASE</version>
</dependency>
```

- 25.** Adăugați clasei `SubscriptionApplication` din proiectul `pricecalculator` adnotarea:

```
@SpringBootApplication
@EnableFeignClients("com.apbdoo.subscription")
@EnableDiscoveryClient
public class SubscriptionApplication {
```

- 26.** Adăugați în fișierul `application.properties` al proiectului `pricecalculator`:

```
eureka.client.service-url.default-zone = http://localhost:8761/eureka
```

- 27.** Înregistrați în același mod în `namingserver` serviciul `subscription`.