TWORZENIE MIKROSERWISÓW Z WYKORZYSTANIEM SPRING CLOUD I DOCKER

TWORZENIE MIKROSERWISÓW Z WYKORZYSTANIEM SPRING CLOUD I DOCKER

STARTER

DOŚWIADCZENIE

CZEMU JESTEM NA SZKOLENIU?

ULUBIONE ZAJĘCIE PO PRACY :)

DO PRZERWY OBIADOWEJ:

WERYFIKACJA DANYCH NA LIŚCIE

ZAZNACZENIE OBECNOŚCI

Dbamy o Ciało

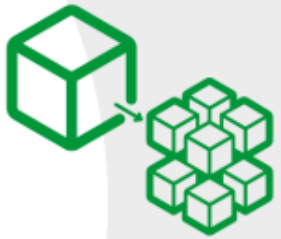
Pijemy dużo Płynów [bezalkoholowych :)]

Minimum 1 przerwa na godzinę

punktualność


SPRING BOOT
PLAN SZKOLENIA
SPRING CLOUD
docker

Dlaczego Spring Boot?



Wstrzykiwanie, konfiguracja, profile etc.

{RESTful API}

Korzystanie z oficjalnych 'starterów'

HIBERNATE

EUREKA

CONFIG SERVER

RIBBON

FEIGN

HYSTRIX

CLOUD BUS

API GATEWAY

PODSTAWY DOCKER'A

DOCKERFILE

DOCKER-COMPOSE.YML


Wprowadzenie
Podstawy
Spring.io


WPROWADZENIE
?
IoC
DI


?
CZYM JEST?
DLACZEGO?

BARDZO ŁATWY START ZE SPRINGIEM

KONWENCJA PONAD KONFIGURACJĄ

PRZEJRZYSTE 'WEJŚCIE' DO APLIKACJI

STARTOWE POM'Y

UPROSZCZENIE KONFIGURACJI APLIKACJI

(NIE)LUBIANY XML

MIKROSERWISY

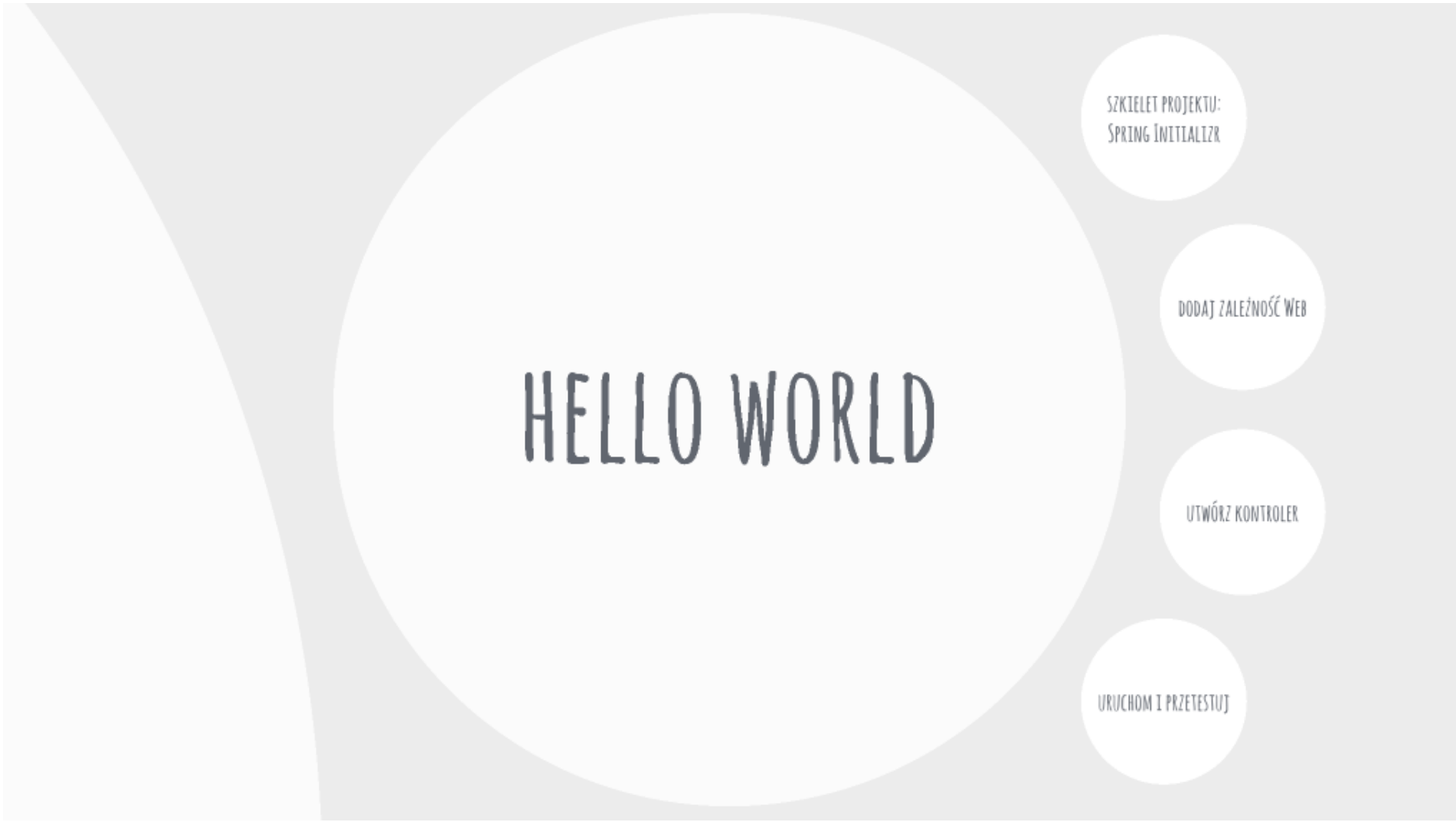
INTEGRACJA Z BIBLIOTEKAMI (NP. NETFLIX)

ZAPOTRZEBOWANIE NA DEWELOPERÓW ;)

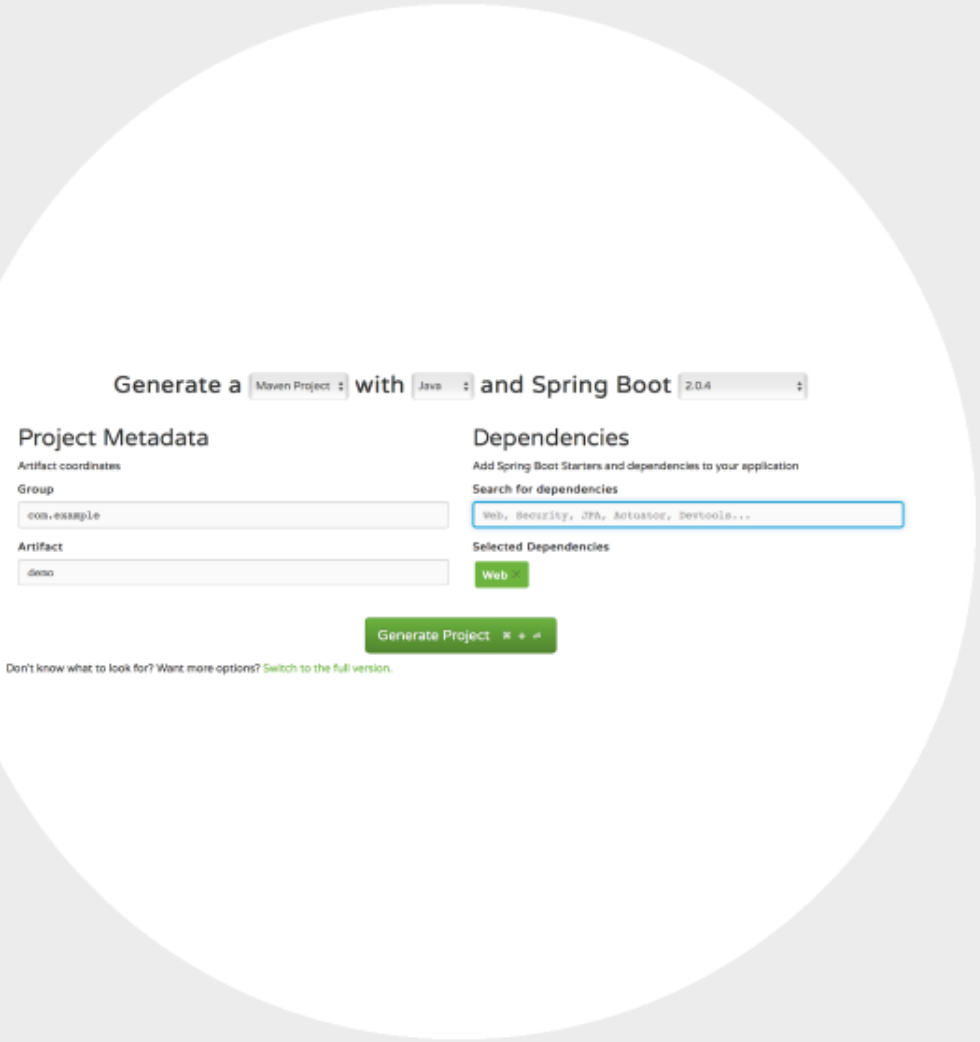
ĆWICZENIA

HELLO WORLD

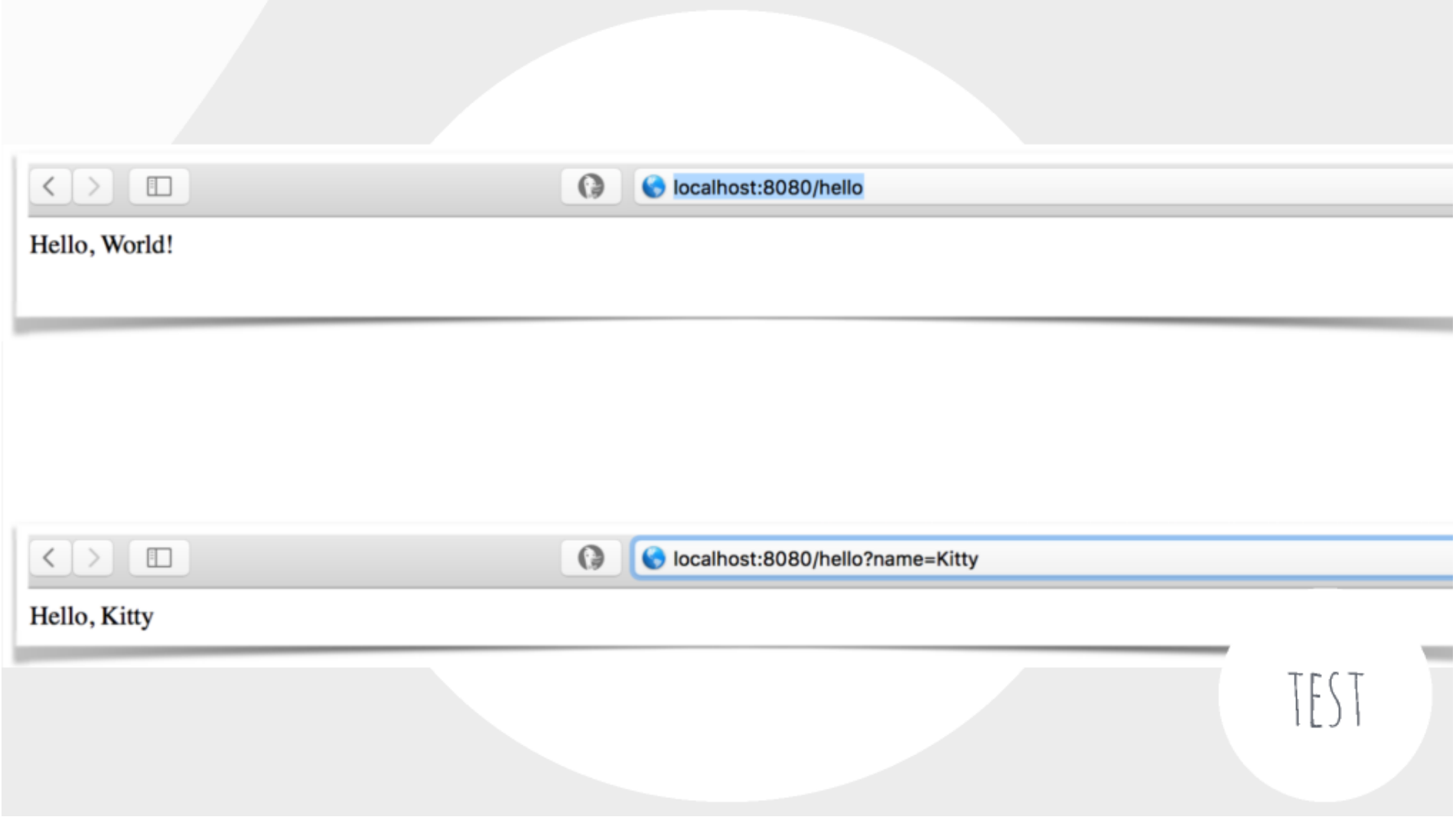
CALCULATOR


HELLO WORLD
SZKIELET PROJEKTU:
SPRING INITIALIZR
DODAJ ZALEŻNOŚĆ WEB
UTWÓRZ KONTROLER
URUCHOM I PRZETESTUJ

WEB LUB KLIENT W IDEI


Generate a Maven Project with Java and Spring Boot 2.0.4
Project Metadata
Artifact coordinates
Group
com.example
Artifact
demo
Dependencies
Add Spring Boot Starters and dependencies to your application
Search for dependencies
Web, Security, JPA, Actuator, Devtools...
Selected Dependencies
Web
Generate Project
Don't know what to look for? Want more options? Switch to the full version.

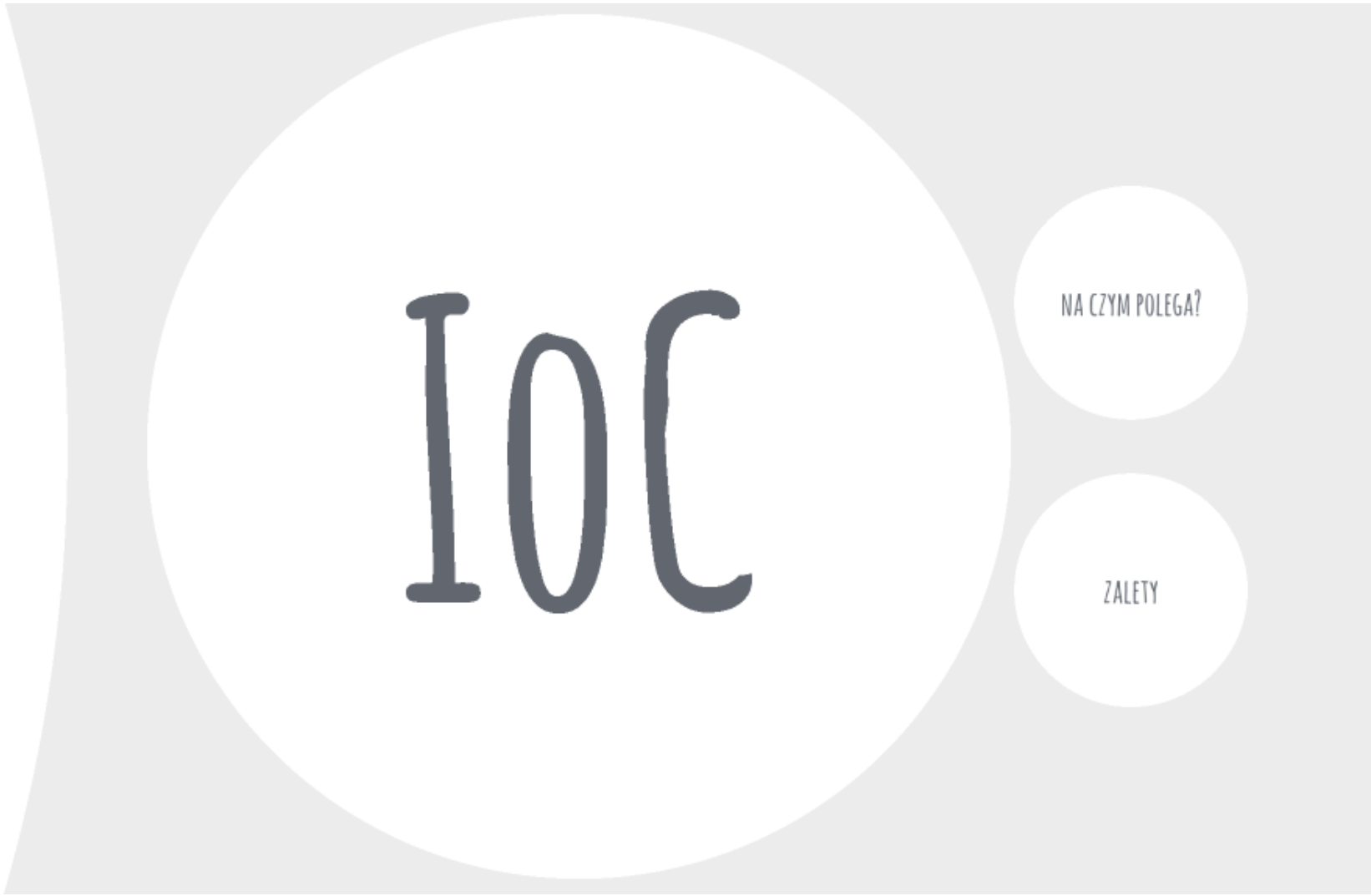
```java
@RestController
public class HelloController {

    @GetMapping("/hello")
    public String greetName(@RequestParam(value="name", required=false) String name) {
        String greeting = "Hello, ";
        String defaultName = "World!";
        return name != null ? greeting + name : greeting + defaultName;
    }
}
```


localhost:8080/hello
Hello, World!
localhost:8080/hello?name=Kitty
Hello, Kitty
TEST

```java
@RunWith(SpringRunner.class)
@WebMvcTest
public class HelloworldApplicationTests {


    @Autowired
    private MockMvc mockMvc;

    @Test
    public void shouldReturnDefaultMessage() throws Exception {
        this.mockMvc.perform(get( urlTemplate: "/hello"))
                .andExpect(status().isOk())
                .andExpect(content().string(containsString( substring: "Hello, World!")));
    }

    @Test
    public void shouldReturnHelloWithNamePassedAsParam() throws Exception {
        this.mockMvc.perform(get( urlTemplate: "/hello?name=Kitty"))
                .andExpect(status().isOk())
                .andExpect(content().string(containsString( substring: "Hello, Kitty")));
    }

}
```

PROJEKT STARTOWY :
_2_CALCULATOR_START

ROZWIĄZANIE :
_2_CALCULATOR_FINAL


IoC
NA CZYM POLEGA?
ZALETY

KONTROLA NAD OBIEKTEM LUB CZĘŚCIĄ PROGRAMU ZOSTAJE PRZEKAZANA DO KONTENERA

IoC MOŻNA ZAIMPLEMENTOWAĆ NA RÓŻNE SPOSOBY JEDNYM Z NICH JEST WZORZEC DI

LEPSZA SEGMENTACJA PROGRAMU

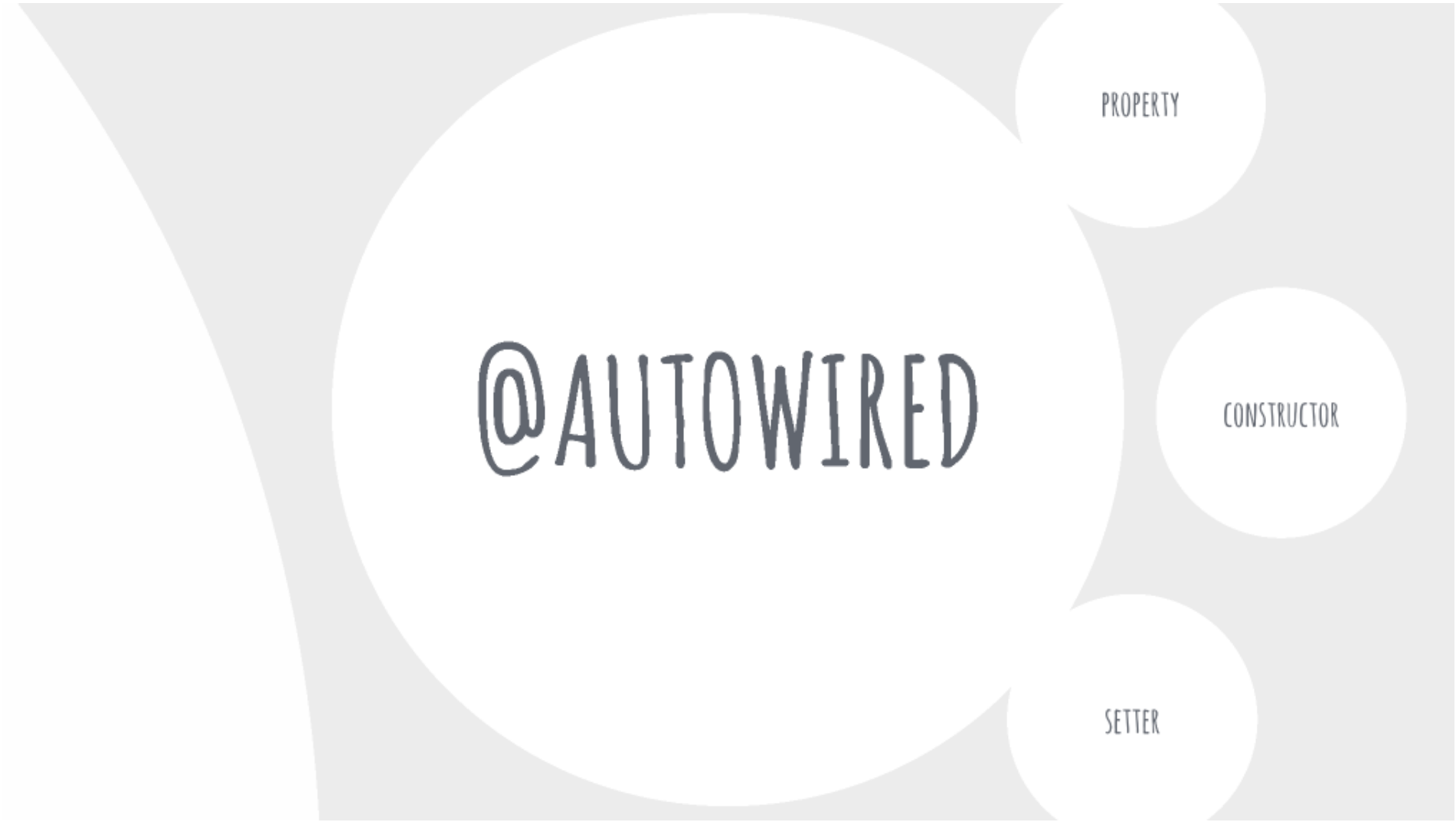
UŁATWIENIE ZMIANY IMPLEMENTACJI

UŁATWIENIE TESTOWANIA

"Dependency Injection" is a 25-dollar term for a 5-cent concept. (...) Dependency injection means giving an object its instance variables. Really. That's it.

- James Shore

WSTRZYKIWANIE W SPRINGU


@AUTOWIRED
PROPERTY
CONSTRUCTOR
SETTER

```java
@RestController
public class GreetController {

    @Autowired
    private GreetService greetService;



    // Użycie
}
```

Spring >= 4.3 && 1 konstruktor = ~~@Autowired~~

```java
@RestController
public class GreetController {

    private GreetService greetService;

    @Autowired
    public GreetController(GreetService greetService) {
        this.greetService = greetService;
    }

    // Użycie
}
```

```java
@RestController
public class GreetController {


    private GreetService greetService;

    @Autowired
    public void setGreetService(GreetService greetService) {
        this.greetService = greetService;
    }

    @GetMapping("/greet")
    public String greet() { return greetService.getMsg(); }
}
```

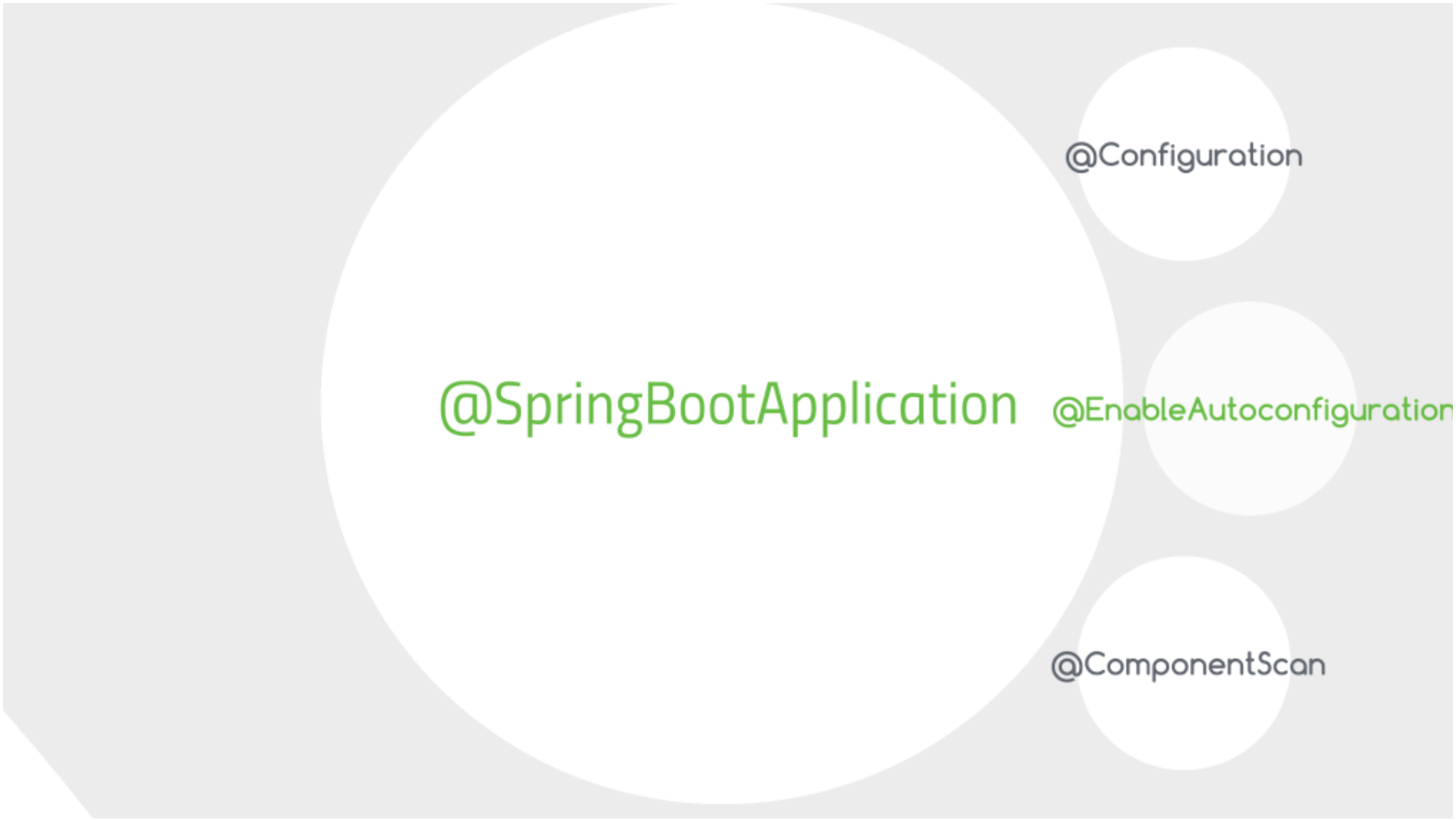
PROJEKT STARTOWY :

_3_CONSTRUCTOR-SETTER-FIELD-START

ROZWIĄZANIE :

_3_CONSTRUCTOR-SETTER-FIELD-FINAL


PODSTAWY
@SpringBootApplication
BEANS
&
COLABORATORS
@Primary
&
@Qualifier
KONFIGURACJA
SCOPE


@Configuration
@SpringBootApplication
@EnableAutoconfiguration
@ComponentScan

ŹRÓDŁO DEFINICJI ZIAREN DLA KONTEKSTU APLIKACJI

KONWENCJA PONAD KONFIGURACJĄ

SPRINGBOOT AUTOMATYCZNIE ZAŁADUJE NIEZBĘDNE ZIARNA

W ZALEŻNOŚCI OD TEGO CO ZNALAZŁ NA CLASSPATH

MOŻNA 'CUSTOMIZOWAĆ' TO ZACHOWANIE

PRZESKANUJ OD BIEŻĄCEGO PAKIETU W DÓŁ

obiekty zarządzane przez Spring, tworzące aplikację & wstrzykiwane do obiektu zależności


@Primary
&
@Qualifier
@Primary
@Qualifier

```java
@Service
@Primary
public class GreetServiceFirstImplementation implements GreetService {
```

```java
@Autowired
private GreetService greetService;
```

Projekt startowy :

_4_using-primary-start

Rozwiązanie :

_4_using-primary-final

```java
@Service("second")
public class GreetServiceSecondImplementation implements GreetService
```

```java
@Autowired
@Qualifier("second")
private GreetService greetService;
```

Projekt startowy :

_5_using-qualifier-start

Rozwiązanie :

_5_using-qualifier-final


KONFIGURACJA
15+
PROPS
VS
YML
@ConfigurationProperties
@Profile
JAVA
VS
XML

https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-external-config.html

```
pageController.msg=Hello from properties
pageController.foo=foo
pageController.bar=bar

pageController:
  msg: Hello from YAML
  foo: foo
  bar: bar
```

PROJEKT STARTOWY :

_6_PROP-AND-YML-START

ROZWIĄZANIE :

_6_PROP-AND-YML-FINAL

@ConfigurationProperties

DLACZEGO

UŻYCIE

mapowanie props'ów do POJO

używamy, gdy wiele wartości

@Valid'acja

wsparcie IDE

możliwe problemy z YAML

@ConfigurationProperties

dodanie zależności (IDEA podpowiada)

setter'y dla ustawianych pól

PROJEKT STARTOWY :

_7_CONFIGURATION-PROPERTIES-START

ROZWIĄZANIE :

_7_CONFIGURATION-PROPERTIES-FINAL

@Profile

DLACZEGO?

UŻYCIE

KONTROLA, CZY DANE ZIARNO MA BYĆ ŁADOWANE DO KONTEKSTU APLIKACJI

NP. LOKALNA BAZA VS PRODUKCYJNA

@Profile("development")

spring.profiles.active=development

-Dspring.profiles.active=development

IDE

PROJEKT STARTOWY :
_8_PROFILES-START

ROZWIĄZANIE :
_8_PROFILES-FINAL

Spring pozwala na różne sposoby konfiguracji

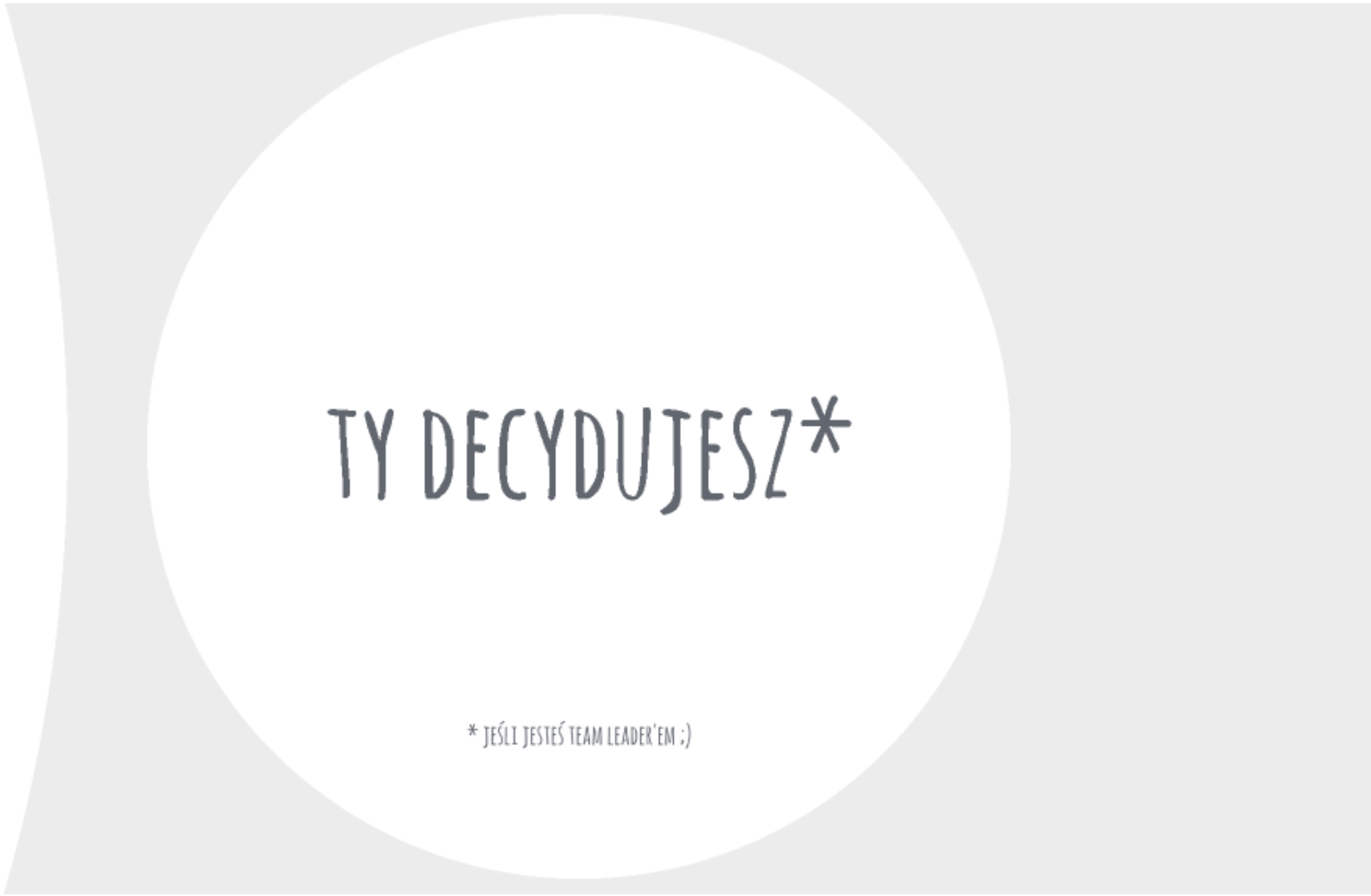
Java Configuration
xml
@dnotacje
Groovy Configuration

Java
vs
xml

co lepsze?

```java
@Configuration
@ComponentScan(basePackages = {"pl.altkom.configuration.controller"})
@PropertySource("classpath:service.properties")
public class JavaConfig {

    @Value("${service.first.implementation.msg}")
    public String firstImplMsg;

    @Bean
    @Profile("java")
    public GreetService firstImpl() { return new GreetServiceFirstImplementation(firstImplMsg); }
}
```

```xml
<beans profile="xml">
    <bean id="secondImpl" class="pl.altkom.di.service.GreetServiceSecondImplementation">
        <constructor-arg value="${service.second.implementation.msg}"/>
    </bean>
    <context:component-scan base-package="pl.altkom.di.controller"/>
    <context:property-placeholder location="classpath:service.properties" />
</beans>
```


TY DECYDUJESZ*
* JEŚLI JESTEŚ TEAM LEADER'EM ;)

PROJEKT STARTOWY :
_9_CONFIGURATION-START

ROZWIĄZANIE :
_9_CONFIGURATION-FINAL

@Scope("<wybrany scope>")

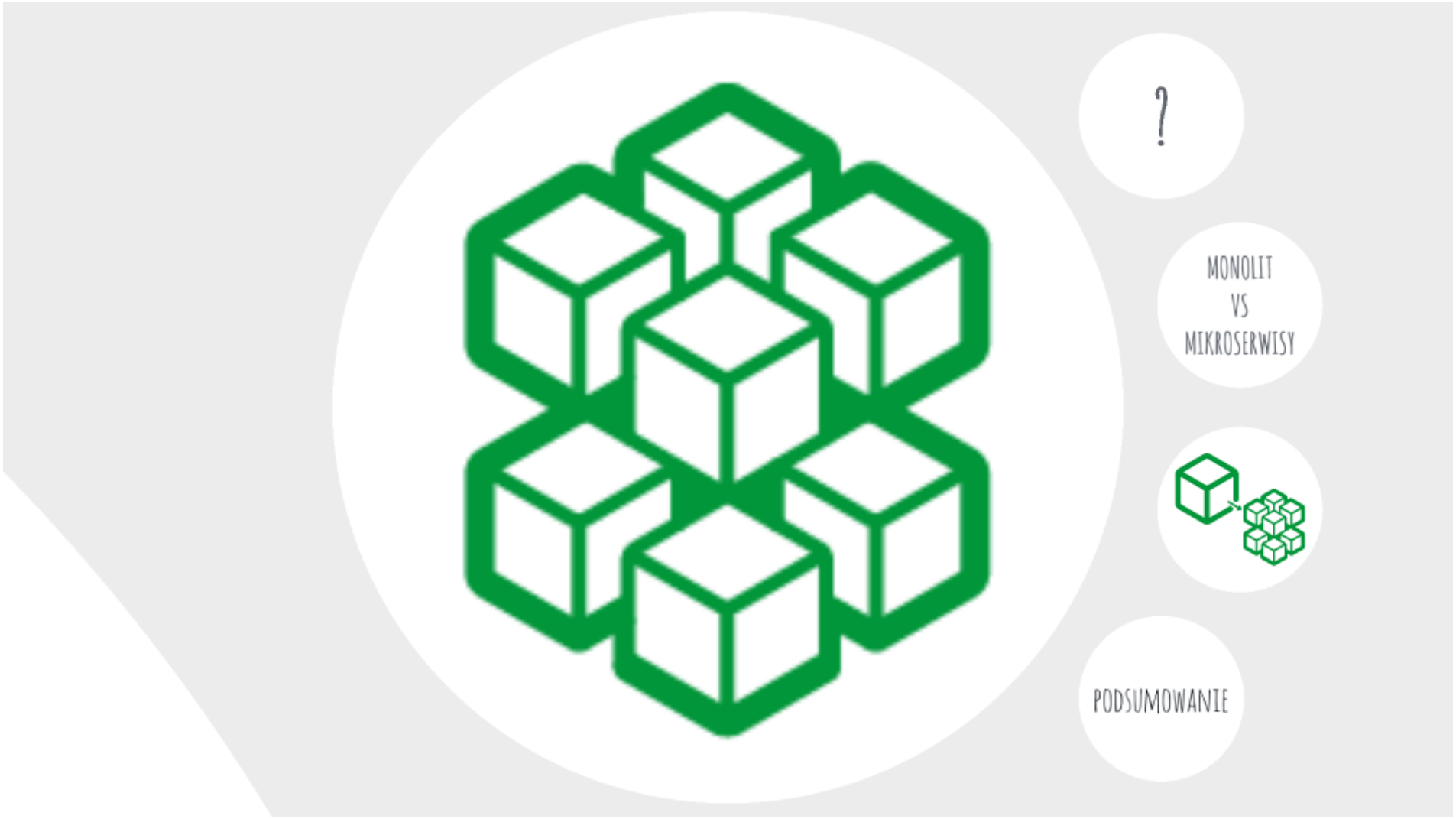
SINGLETON (default)
PROTOTYPE
REQUEST
SESSION
GLOBAL SESSION

PROJEKT STARTOWY :
_10-SCOPE-START

ROZWIĄZANIE :
_10-SCOPE-FINAL

https://spring.io/guides/gs/accessing-data-rest/


WPROWADZENIE
FORTUNE
CLOUD
BUS
API
GATEWAY


?
MONOLIT
VS
MIKROSERWISY
PODSUMOWANIE

STYL ARCHITEKTONICZNY

ADRIAN COCKROFT (NETFLIX)

ALTERNATYWA DLA MONOLITYCZNYCH ROZWIĄZAŃ

MARTIN FOWLER

DEKOMPOZYCJA SYSTEMU

NA ZESTAW MAŁYCH SERWISÓW,

FUNKCJONUJĄCYCH W NIEZALEŻNYCH PROCES/

I KOMUNIKUJĄCYCH SIĘ ZE SOBĄ

PRAKTYCZNA DEFINICJA

Fine-grained SOA

SOA
vs
Mikro


Mikroserwisy to nie SOA!
SOA
MIKRO

INTEGRACJA SYSTEMÓW

'INTELIGENTNA' TECHNOLOGIA INTEGRACJI

'GŁUPIE' SERWISY

DEKOMPOZYCJA APLIKACJI

'GŁUPIA' TECHNOLOGIA INTEGRACJI

'INTELIGENTNE' SERWISY

ANALOGIA

```
PS AUX | GREP SOMETHING | AWK '{PRINT $2}
```

Developing a single application
as a suite of small services,
each running in its own process and
communicating with lightweight mechanisms,
often an HTTP resource API

- Zestaw małych serwisów składający się na całość
  - zamiast pojedyńczej, monolitycznej aplikacji
- ...każdy z nich w osobnym procesie
- ...komunikujących się ze sobą
  - HTTP/REST, messaging etc.
- ...osobno pisane i utrzymywane
- ...enkapsulują biznesowe funkcjonalności
  - zamiast konstruktów języka (klasy, pakiety)
- ...niezależnie wymieniane / upgrade (może być trudno)

CHARAKTERYSTYKA

ZALETY

WYZWANIA


CHARAKTERYSTYKA
ZALETY
WYZWANIA

APLIKACJA ŁATWA DO OGÓLNEGO ZROZUMIENIA

TRUDNO SIĘ W NIĄ WGRYŹĆ

POJEDYŃCZY, WYKONYWALNY PLIK

MODUŁOWOŚĆ APLIKACJI OPARTA O JEZYK, W KTÓRYM JEST NAPISANA (PAKIETY, KLASY, METODY, FRAMEWORKI ETC.)

DO PEWNEGO MOMENTU (ROZMIARU), ŁATWE:

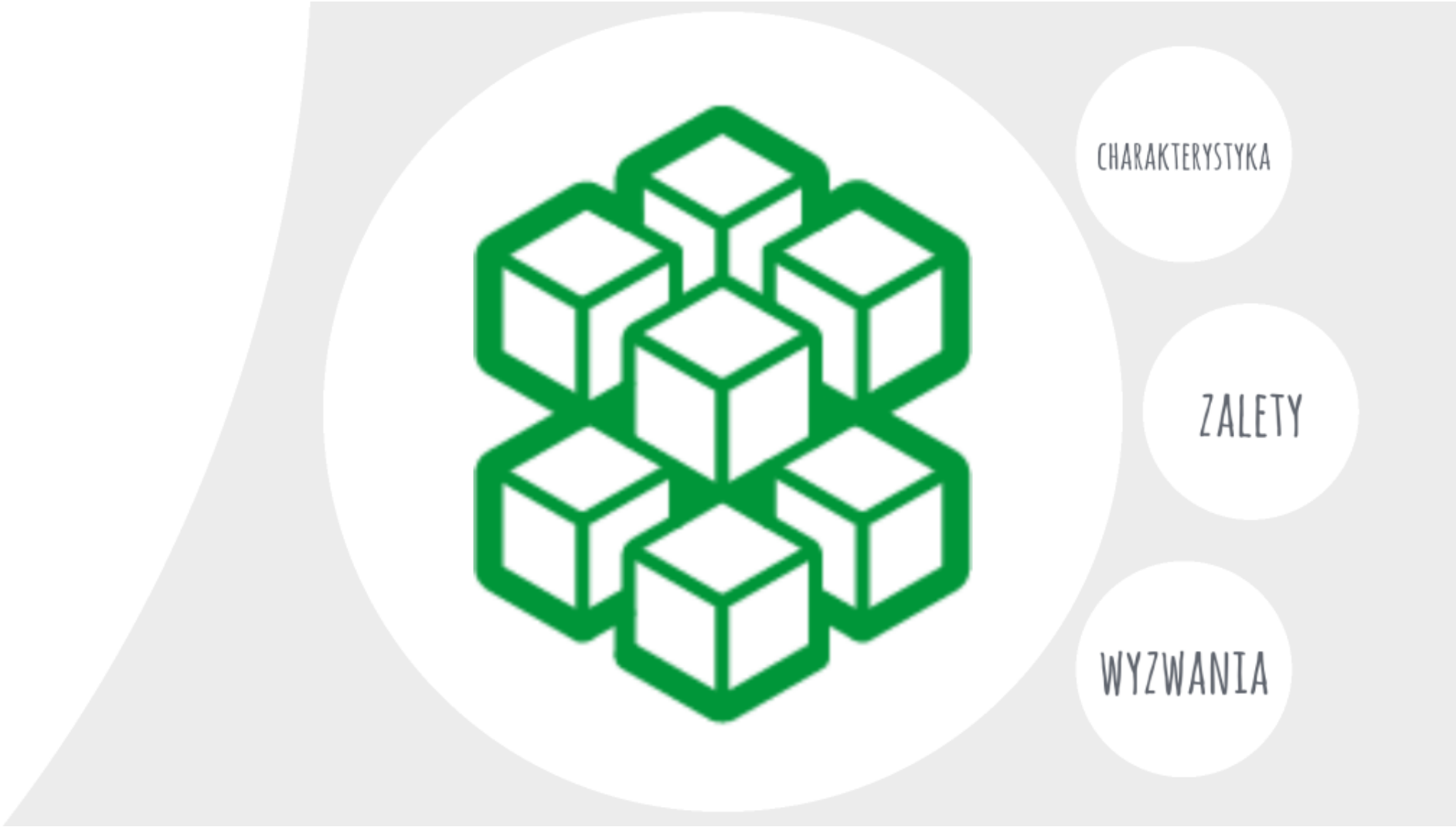
TESTY

DEPLOY

ZARZĄDZANIE (ZMIANAMI)

SKALOWANIE (LB I DODATKOWE INSTANCJE)

- PRZYWIĄZANIE DO FRAMEWORK'U / JĘZYKA
  - TRUDNO EKSPERYMENTOWAĆ Z NOWYMI TECHNOLOGIAMI
  - TRUDNO DOBRAĆ NAJLEPSZĄ TECHNOLOGIĘ DLA ZADANIA
- 1 DEV: TRUDNOŚCI ZE ZROZUMIENIEM CODEBASE DUŻEJ APLIKACJI
  - AMAZON: 2 PIZZA RULE
- DEPLOYMENT POJEDYŃCZEJ APLIKCJI
  - ZMIANY SĄ 'ZAKŁADNIKAMI' OPÓŹNIEŃ /WYKRYTYCH PROBLEMÓW W INNYCH ZMIANACH


CHARAKTERYSTYKA
ZALETY
WYZWANIA

MAŁE, NIEZALEŻNIE DEPLOY'OWANE APLIKACJE WYMUSZAJĄ PRZEJRZYSTĄ ARCHITEKTURĘ INTERFEJSÓW

ZAKRES ZMIAN DOTYCZY JEDNEGO SERWISU

POLYGLOT PERSISTENCE

DECENTRALIZACJA ZARZĄDZANIA

LOW COUPLING

RÓŻNE ROZWIĄZANIA BAZODANOWE DLA RÓŻNYCH SERWISÓW

RDBMS NIE ZAWSZE JEST NAJLEPSZE
(KONTROWERSYJNE DLA NIEKTÓRYCH - BRAK TRANSAKCJI,
MODELU WSPÓLNEGO DLA CAŁEJ APLIKACJI)

NAJLEPIEJ DOBRANEGE NARZĘDZIA
DLA KONKRETNEGO ZADANIA

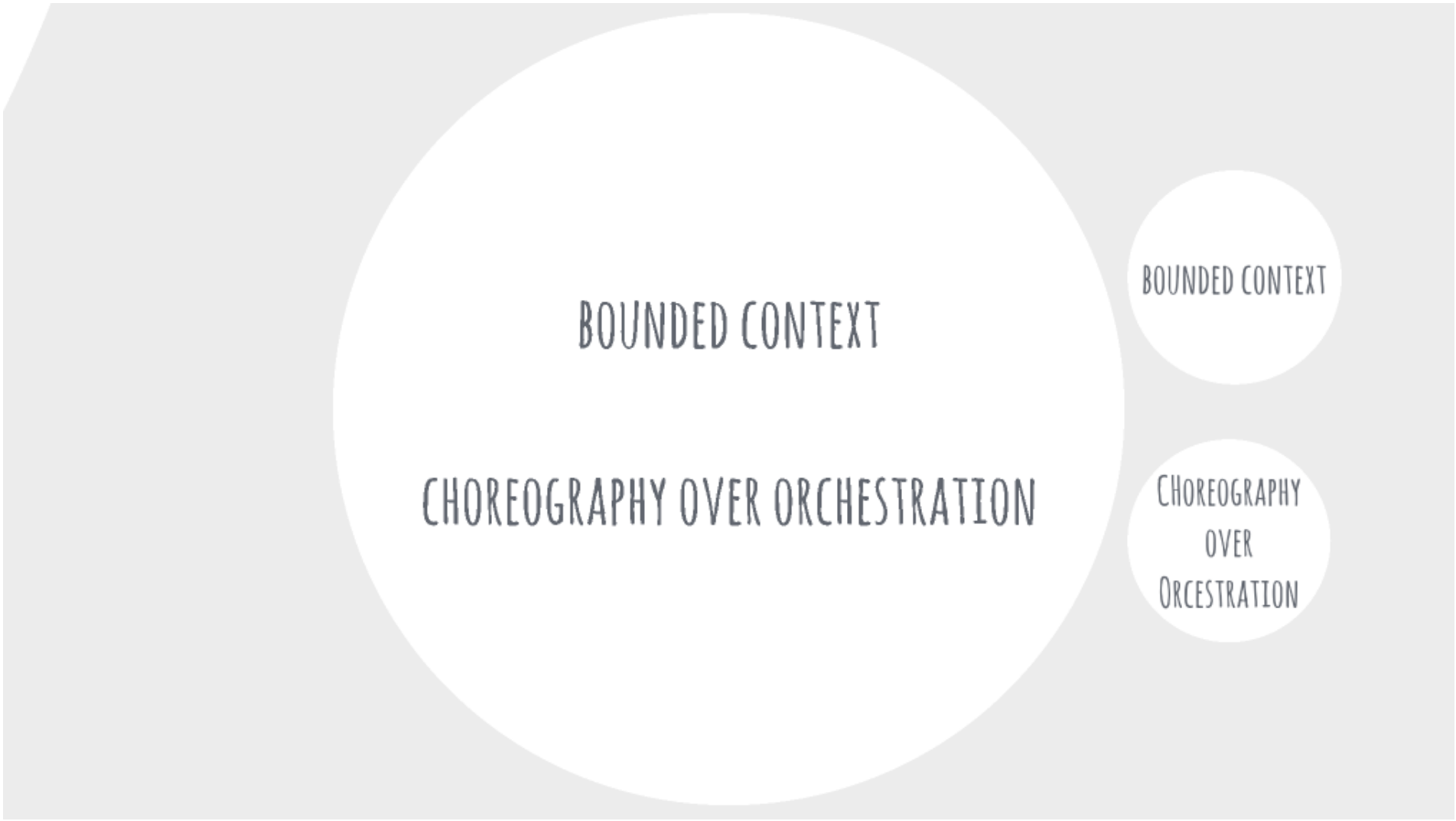
TOLERANT READERS

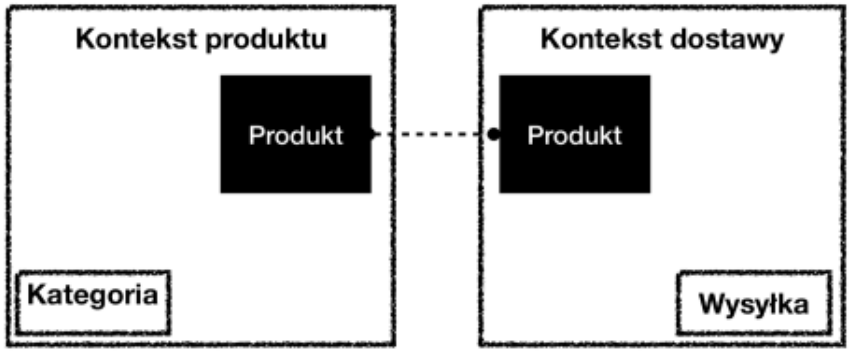
SERWISY ROZWIJANE W SWOIM TEMPIE,
W ZALEŻNOŚCI OD POTRZEB

KONTRAKTY
CONSUMER-DRIVEN

Be conservative in what you do,
be liberal in what you accept from others.
-- Jon Postel

SERWISY EWOLUUJĄ,
NADAL WYPEŁNIAJĄC
OCZEKIWANIA KLIENTÓW


BOUNDED CONTEXT
CHOREOGRAPHY OVER ORCHESTRATION
BOUNDED CONTEXT
CHOREOGRAPHY OVER ORCESTRATION



Bounded Context = high cohesion


Event
Stream

NISKI PRÓG WEJŚCIA DLA NOWYCH OSÓB

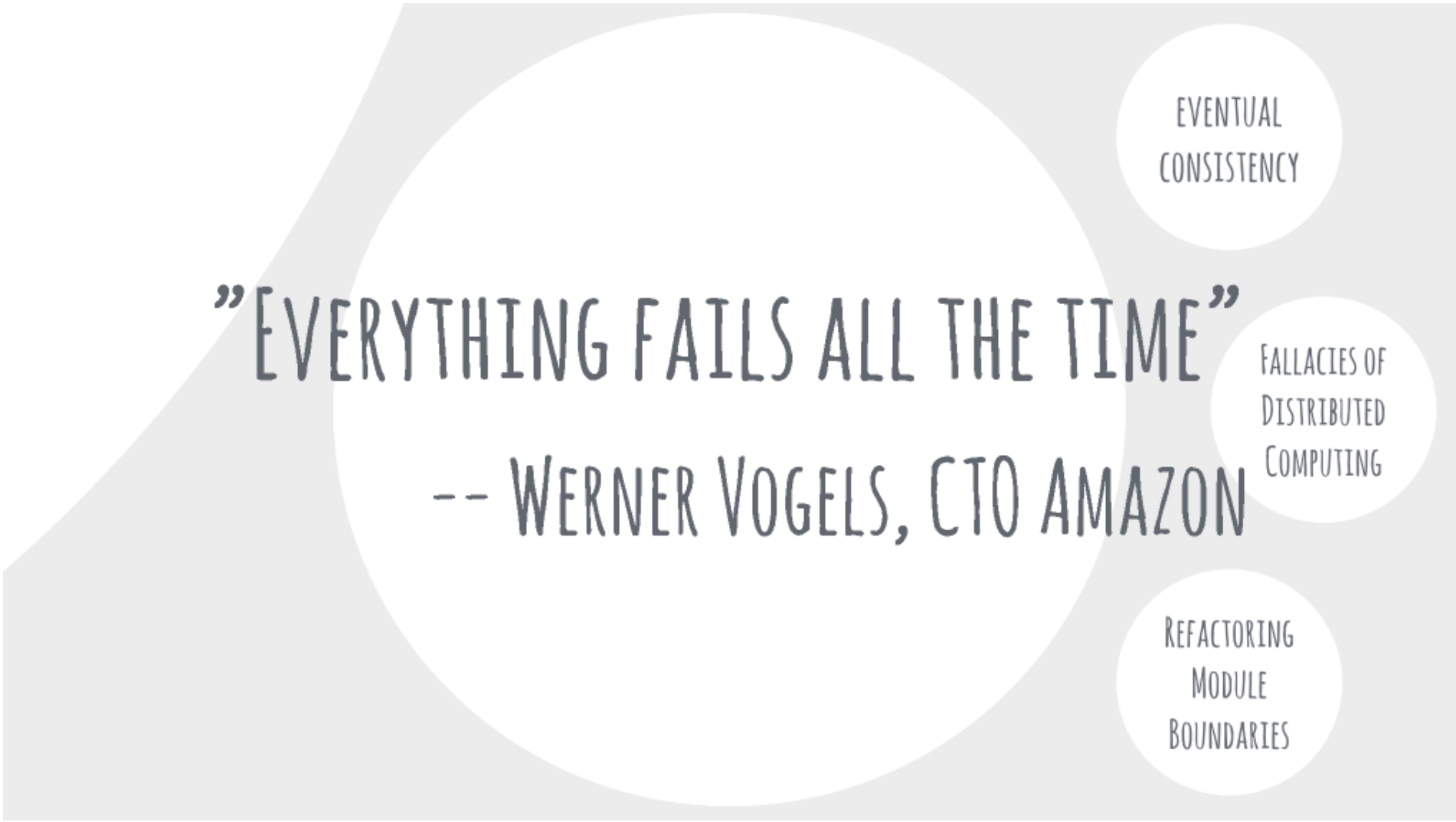
POJEDYNCZY SERWIS, BARDZO ŁATWE:
TESTOWANIE, DEPLOY, WERSJONOWANIE, SKALOWANIE, ZARZĄDZANIE

BRAK PRZYWIĄZANIA DO JĘZYKA / FRAMEWORKU

DYNAMICZNE SKALOWANIE

ŁATWA ADAPTACJA NOWYCH TECHNOLOGII

CYKL PRODUKCJI NIEZALEŻNY (TEORETYCZNIE) OD INNYCH KOPONENTÓW


EVENTUAL CONSISTENCY
"EVERYTHING FAILS ALL THE TIME"
-- WERNER VOGELS, CTO AMAZON
FALLACIES OF DISTRIBUTED COMPUTING
REFACTORING MODULE BOUNDARIES


ACID

MOŻNA POLEGAĆ NA SIECI

LATENCY = 0

PRZEPUSTOWOŚĆ JEST NIESKOŃCZONA

SIEĆ JEST BEZPIECZNA

TOPOLOGIA SIECI NIE ZMIENIA SIĘ

JEST JEDEN ADMINISTRATOR

PRZESYŁ DANYCH NIC NIE KOSZTUJE

SIEĆ JEST HOMOGENICZNA

Bardzo łatwy refaktoring w obrębie serwisu, może jednak sprawić wiele kłopotów, gdy wykracza poza niego

Dekompozycja monolitu

Jak bardzo mikro ?

przykładowe wzorce

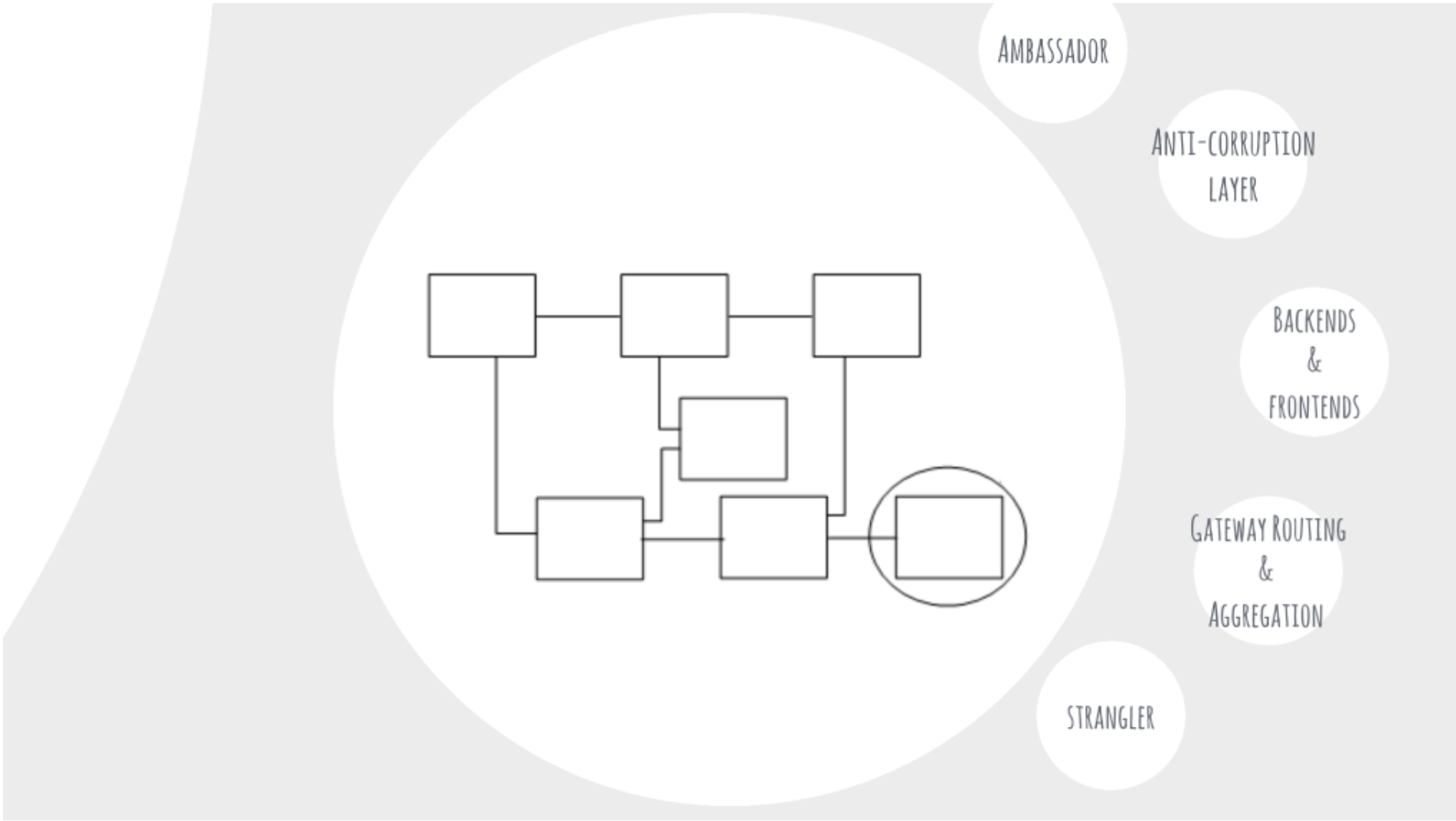
Rozmiar nie ma znaczenia...

jeden dev (ka) / 2 Pizza Rule

dokumentacja do przeczytania i zrozumienia

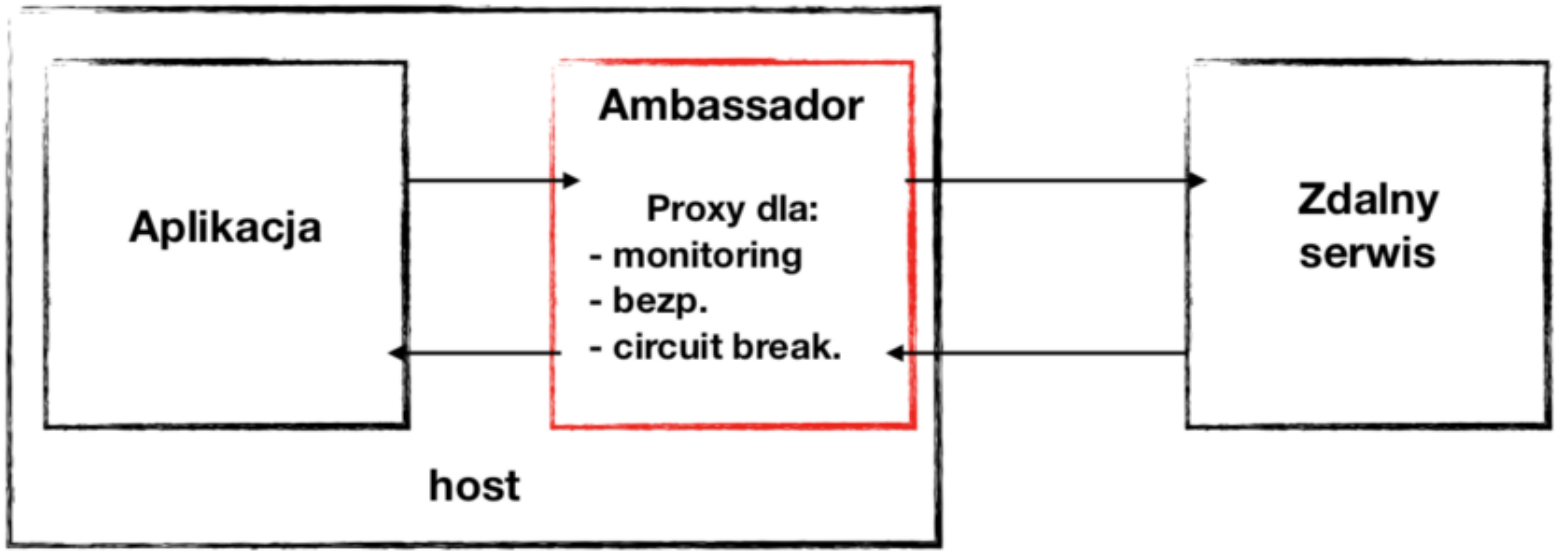
dziesiątki smaczków, nie setki

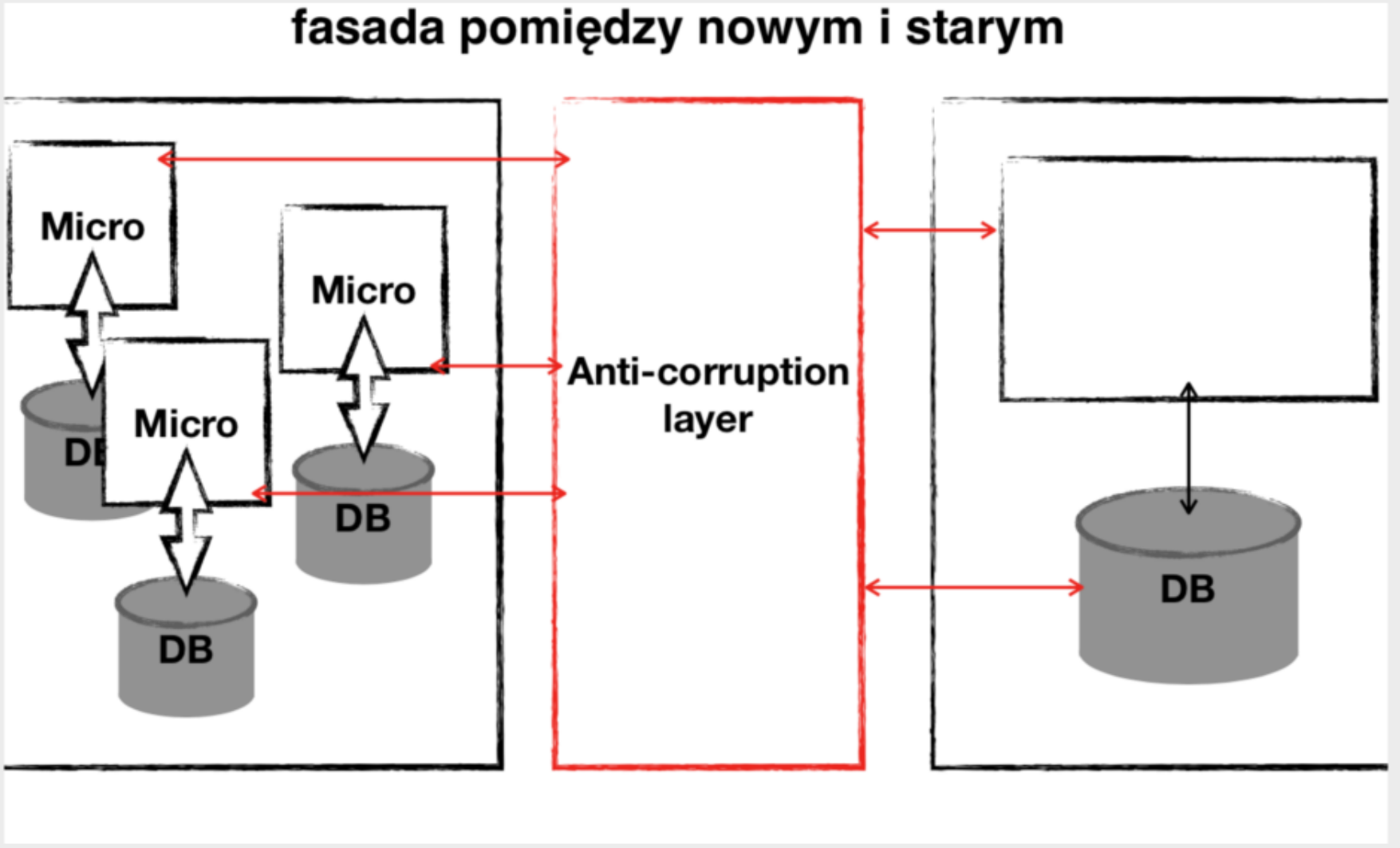
przewidywalny = łatwy do eksperymentowania z nim

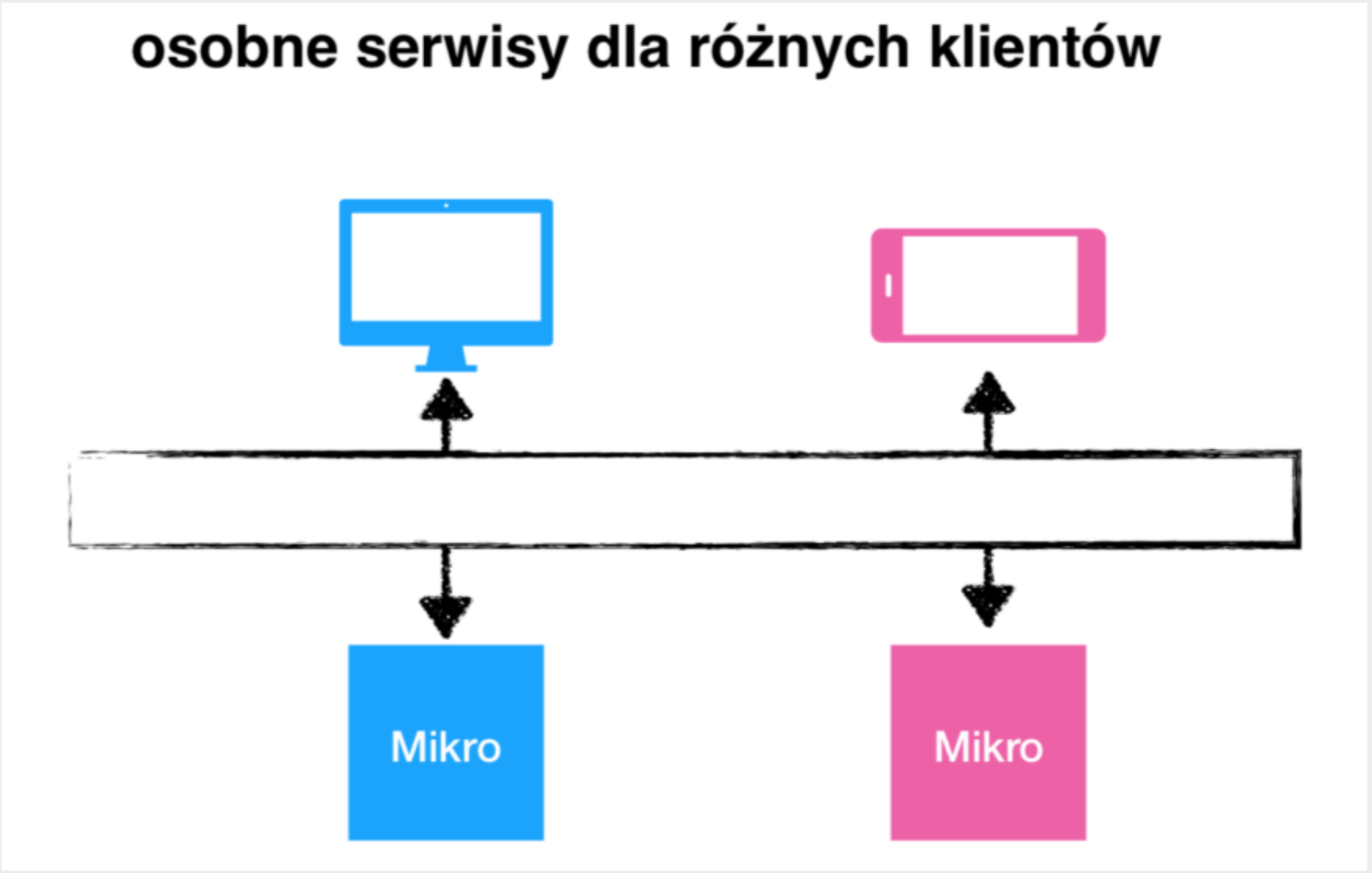

Ambassador
Anti-corruption layer
Backends & frontends
Gateway Routing & Aggregation
strangler

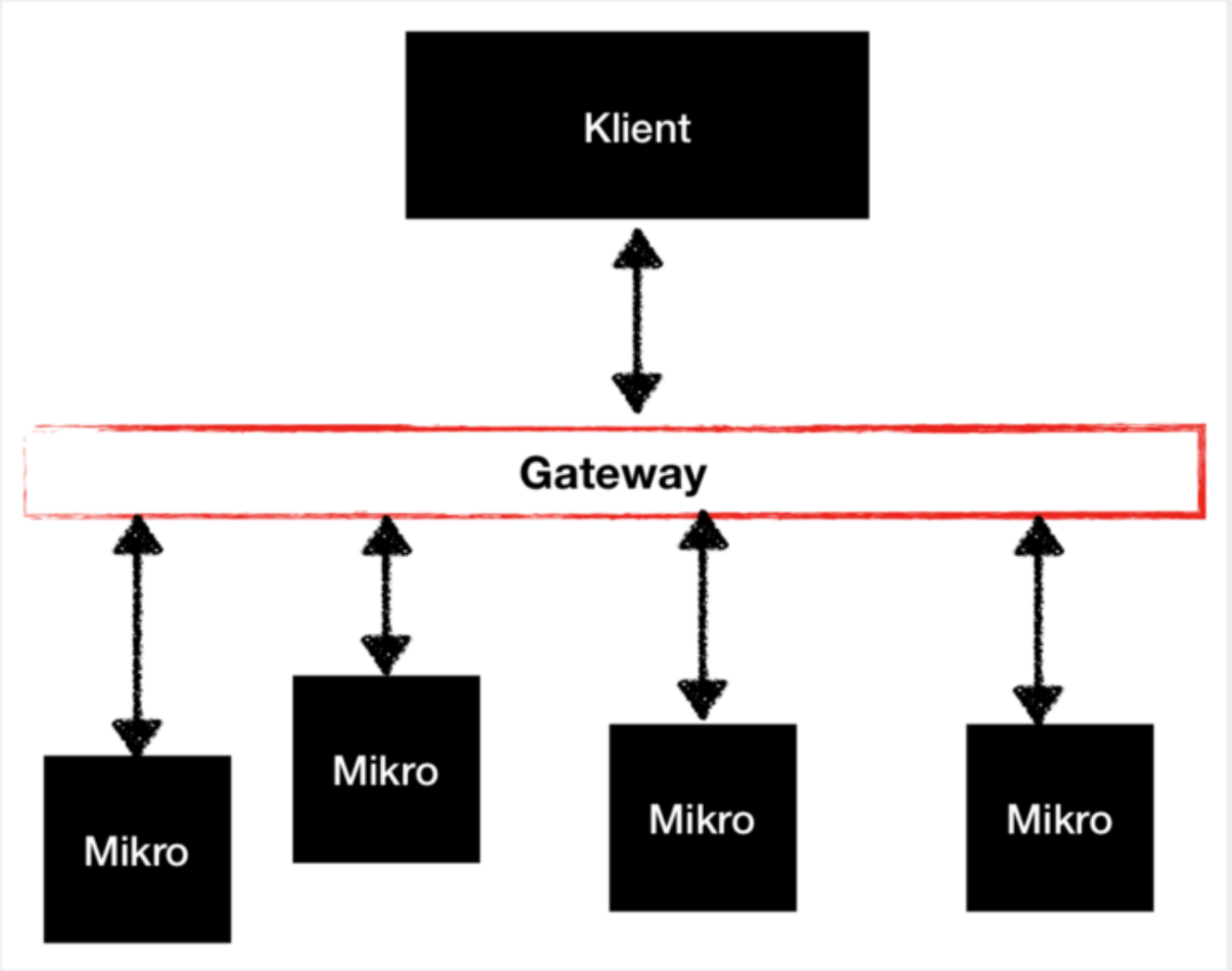
Ambassador

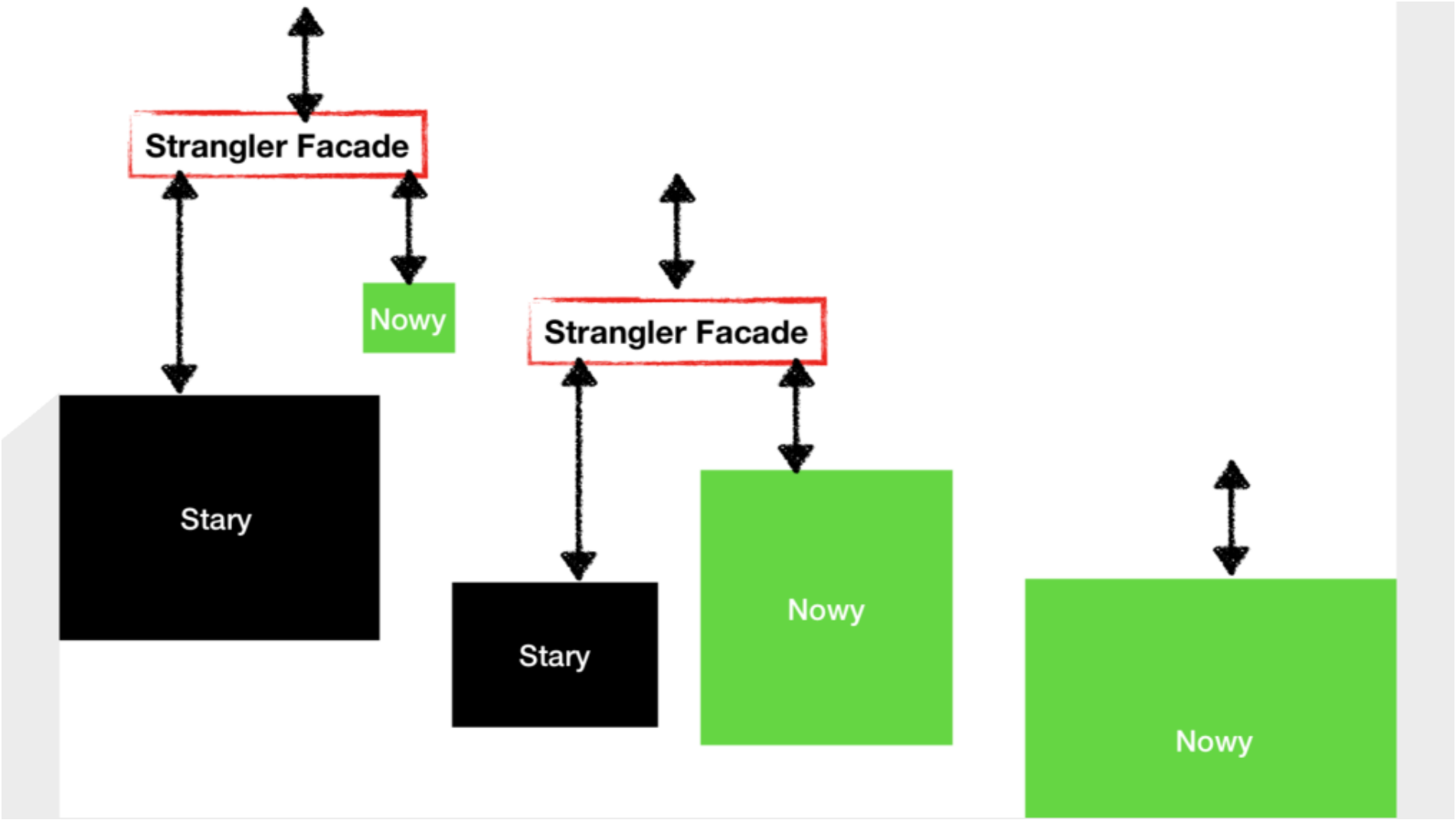
odciążenie klienta




fasada pomiędzy nowym i starym
Micro
Micro
Micro
DB
DB
DB
Anti-corruption
layer
DB


osobne serwisy dla różnych klientów
Mikro
Mikro


Klient
Gateway
Mikro
Mikro
Mikro
Mikro

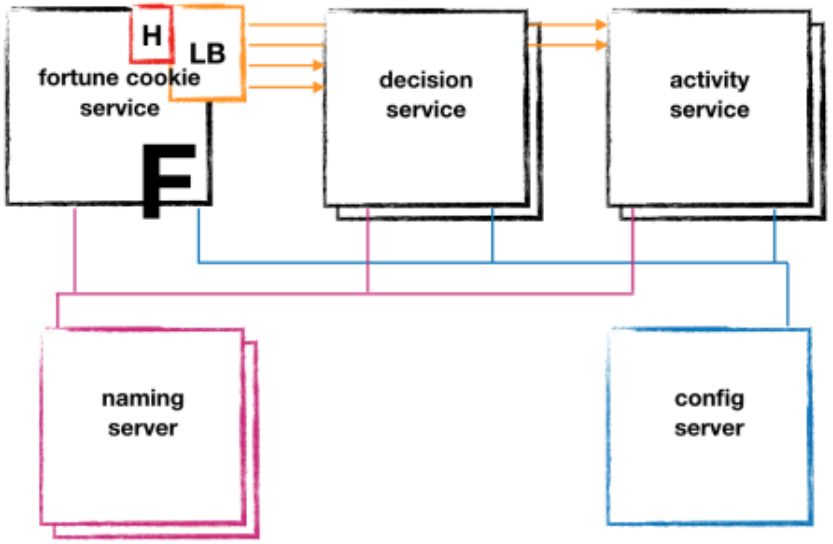

Strangler Facade
Nowy
Stary
Strangler Facade
Stary
Nowy
Nowy

Mikroserwisy to jeszcze jedno architektoniczne podejście do tworzenia systemów

dekompozycja monolitu na niezależnie istniejące, komunikujące się ze sobą procesy

Mikroserwisy tak jak i rozwiązania monolityczne, mają swoje zalety i wady

mem frontend backend :)



EUREKA

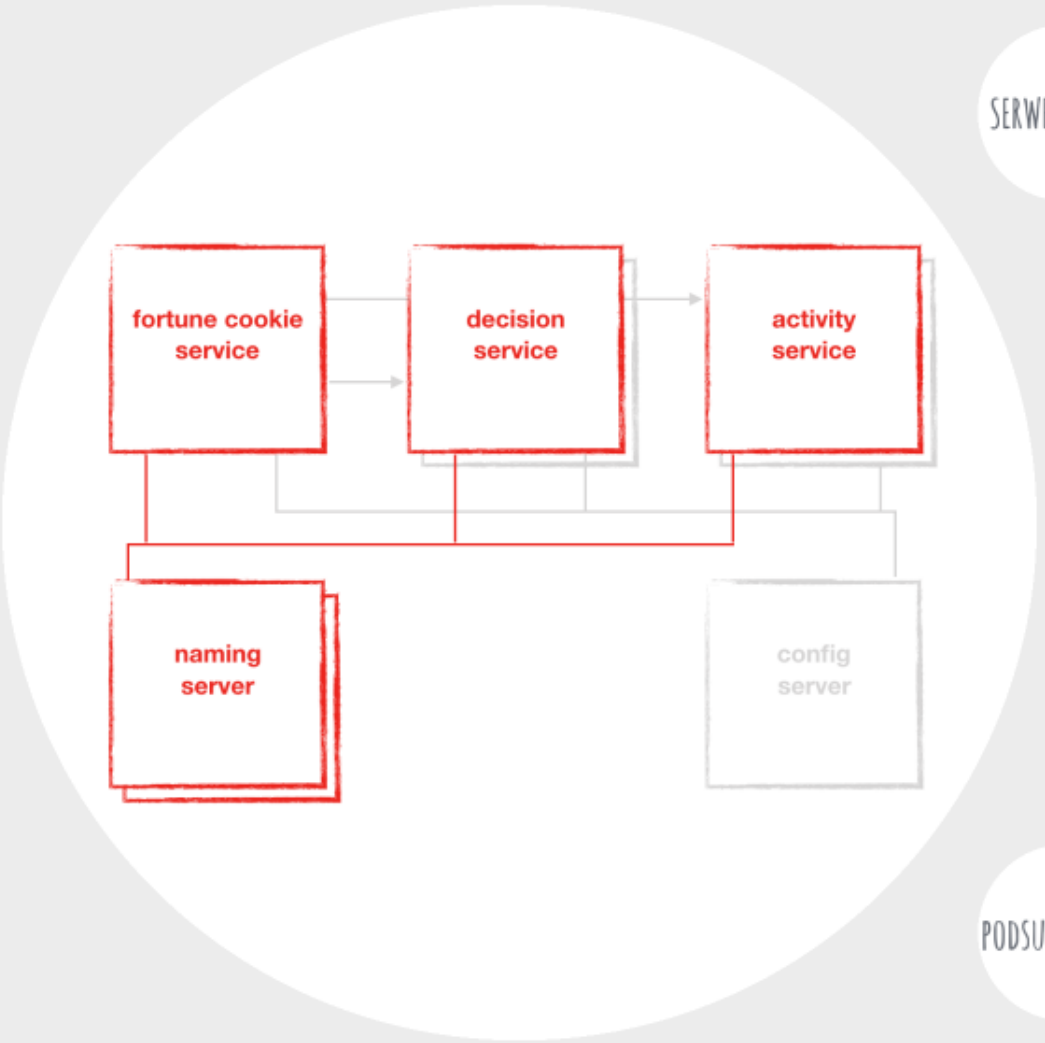
CONFIG SERVER

RIBBON

FEIGN

HYSTRIX

OAUTH2



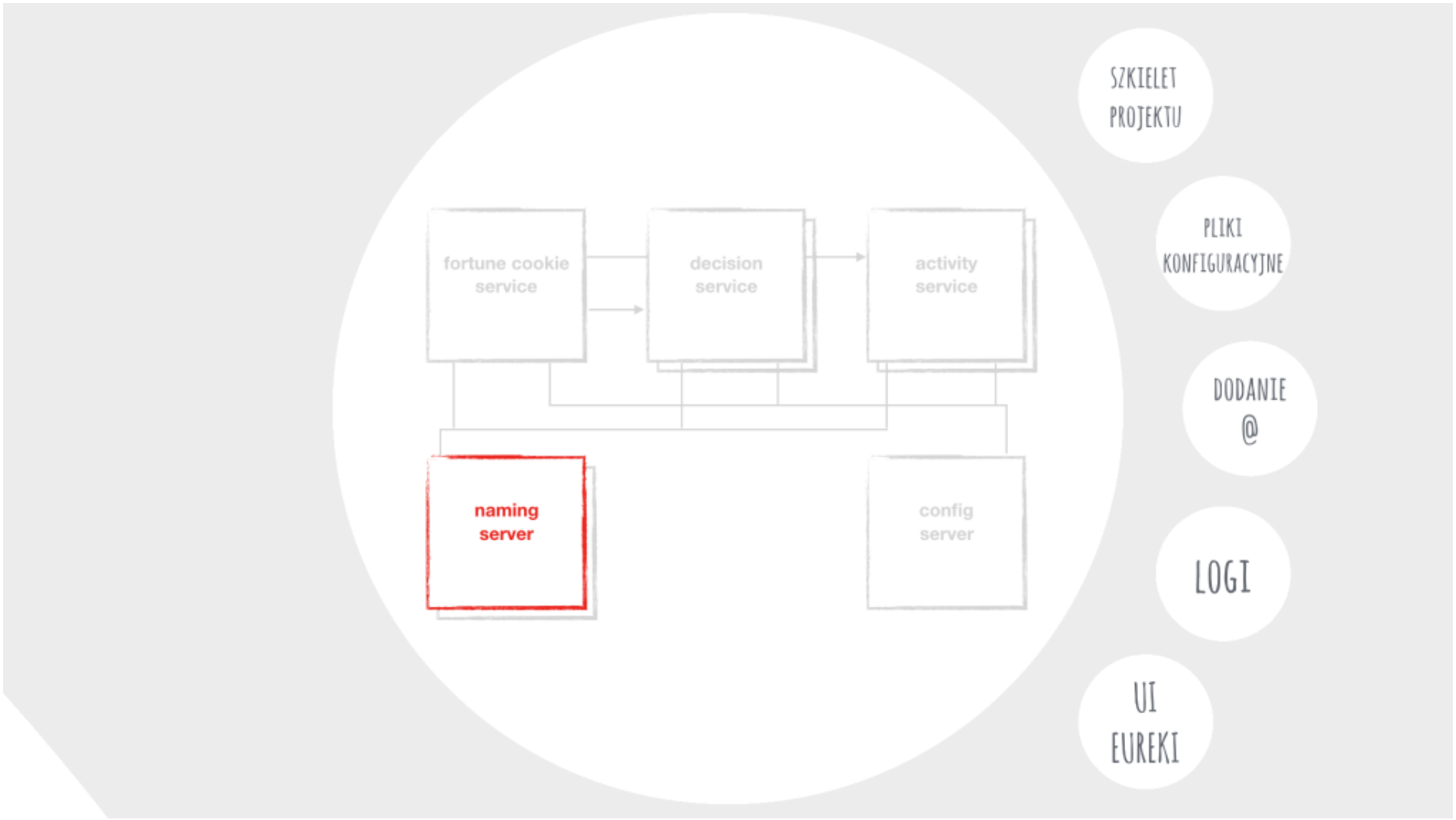
SERWER EUREKA

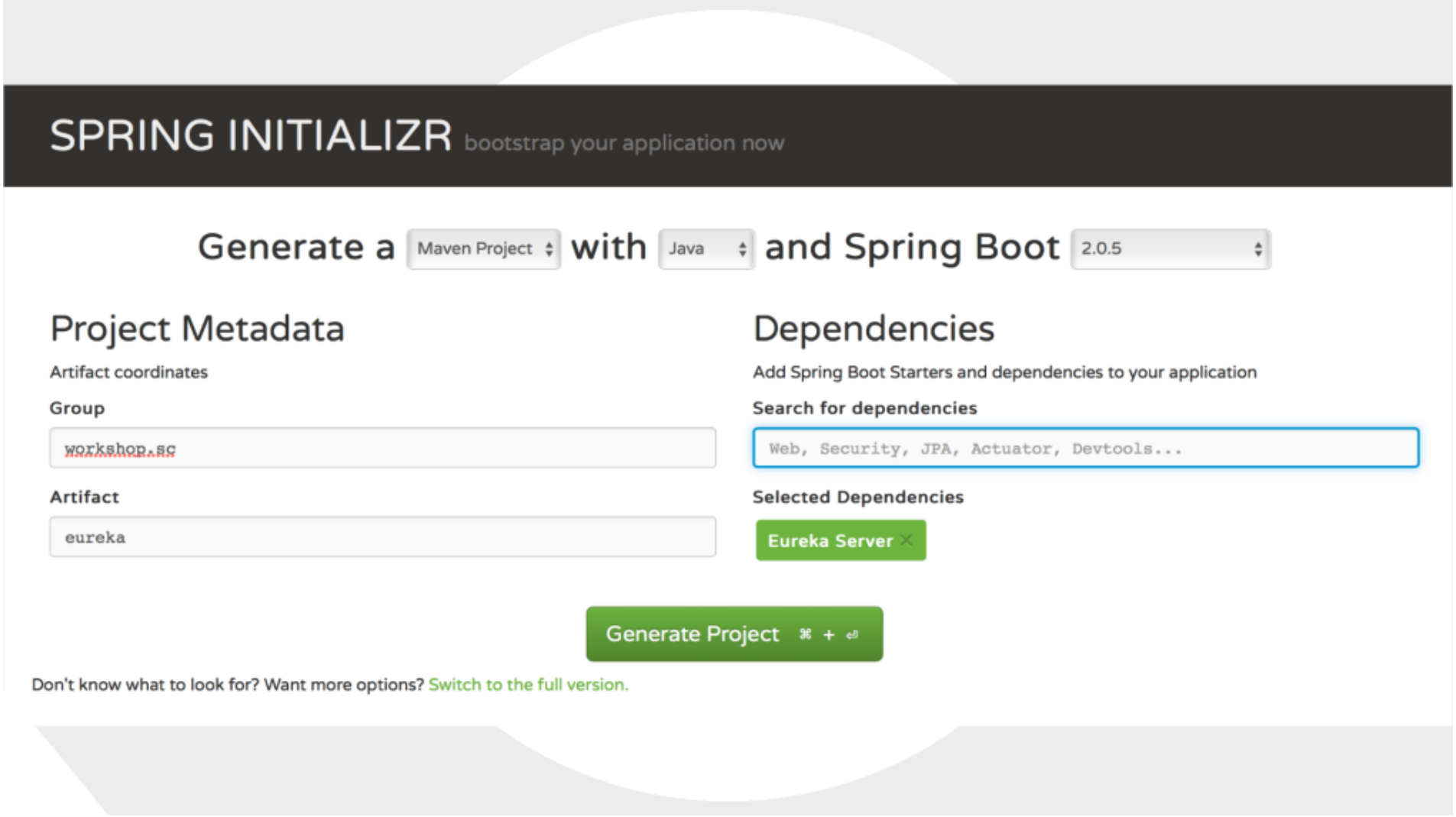
SERWISY

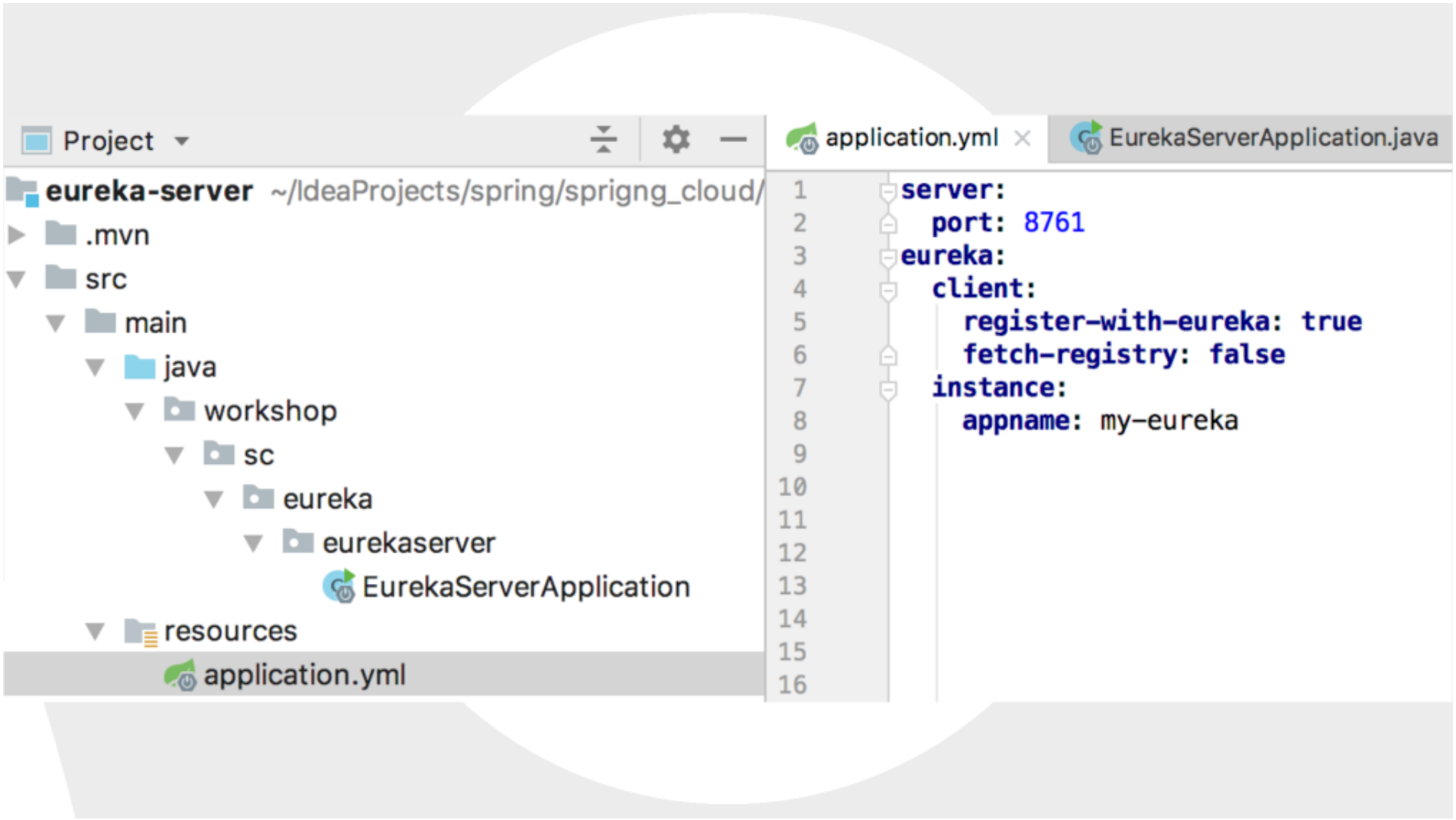
REFAKTOR FCS

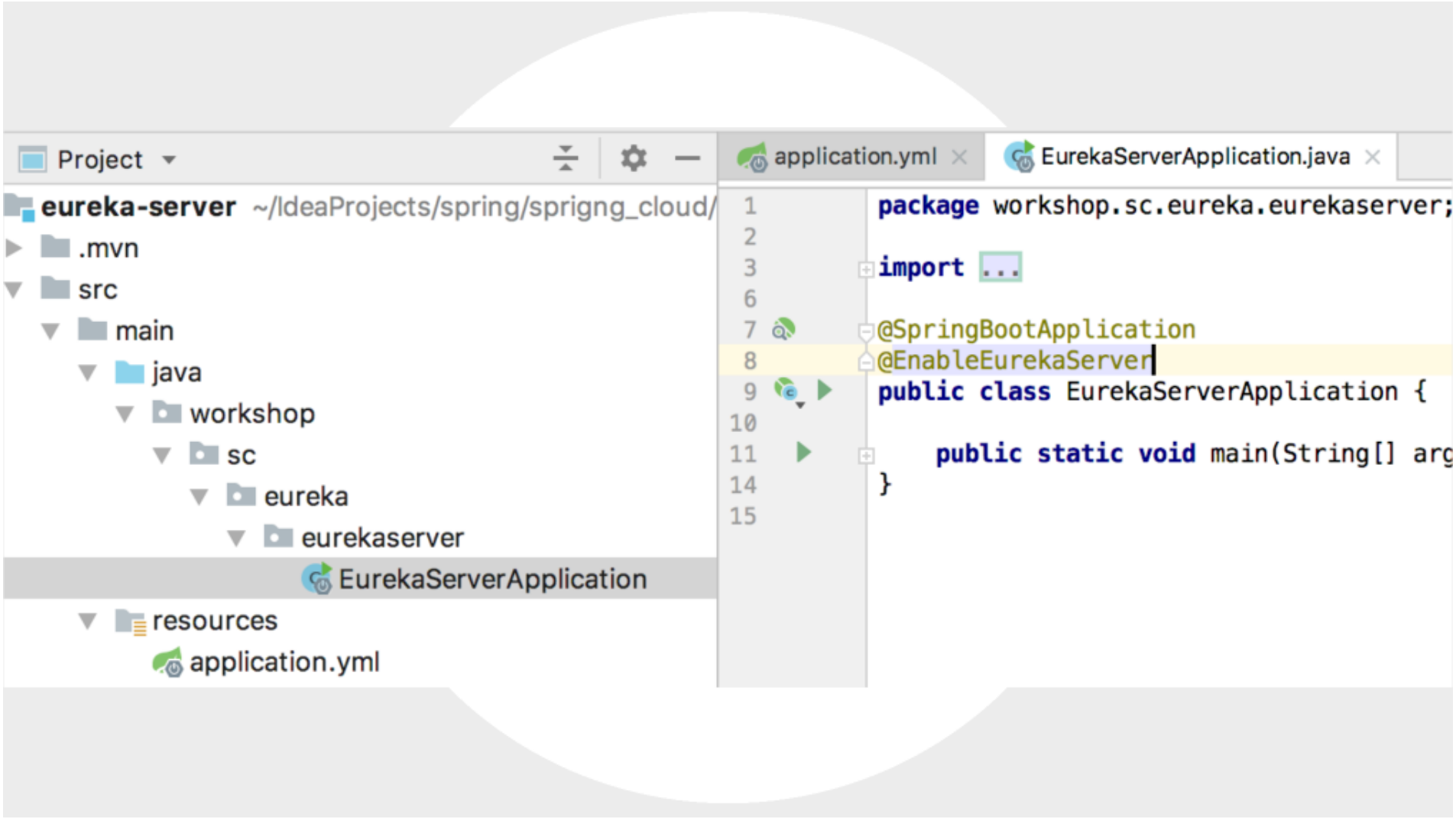
REFAKTOR EURKEKA

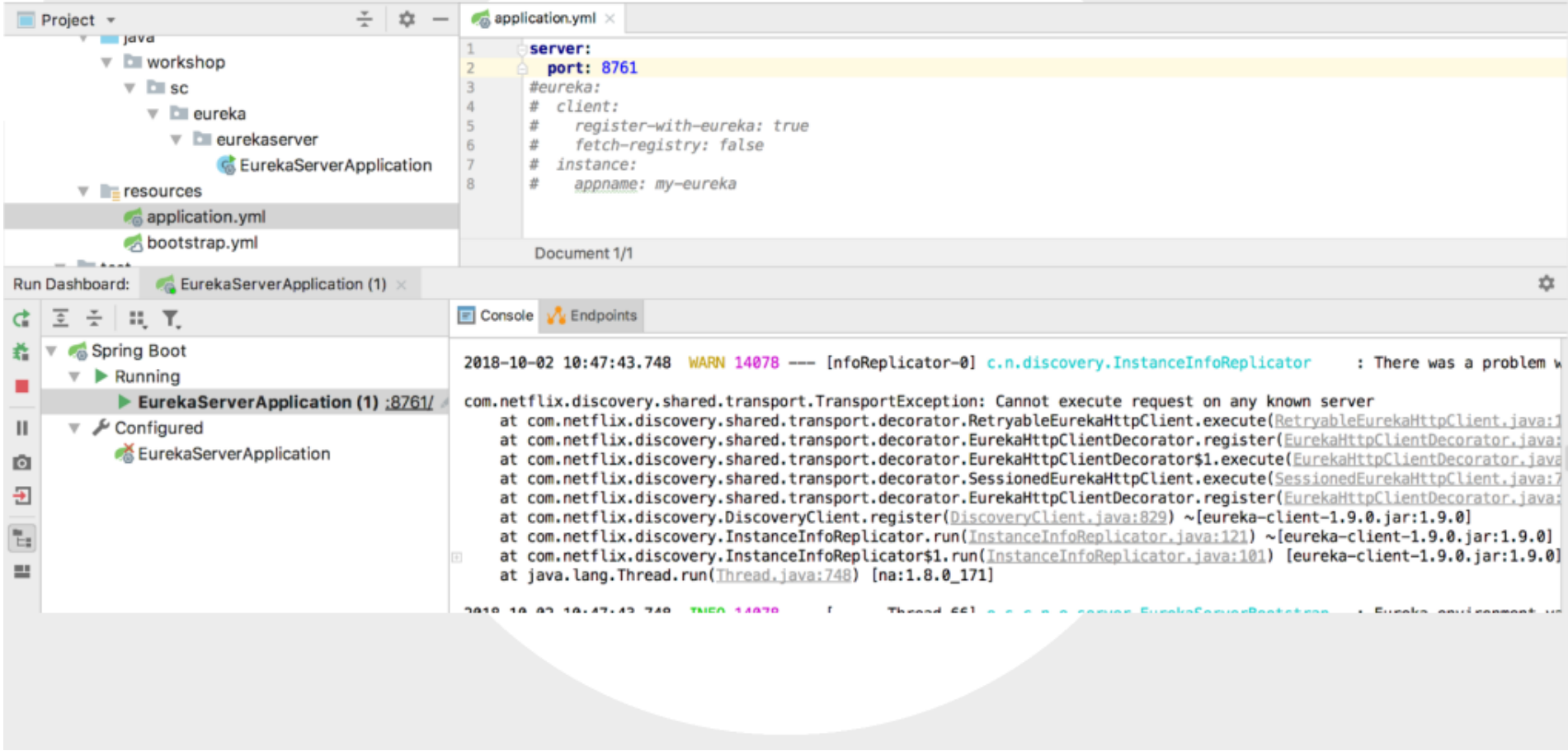
PODSUMOWANIE

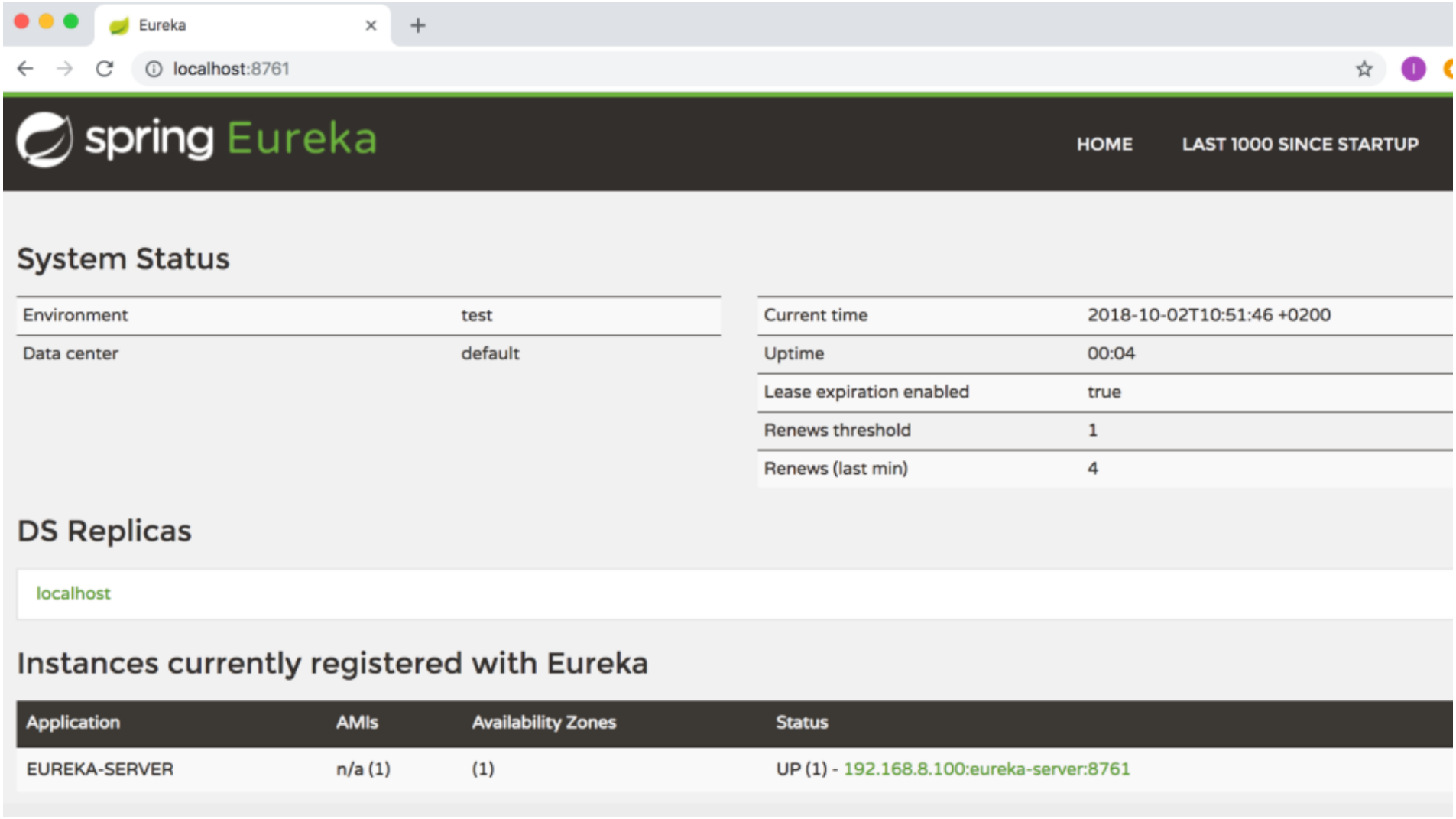

fortune cookie service
decision service
activity service
naming server
config server
SZKIELET PROJEKTU
PLIKI KONFIGURACYJNE
DODANIE @
LOGI
UI EUREKI

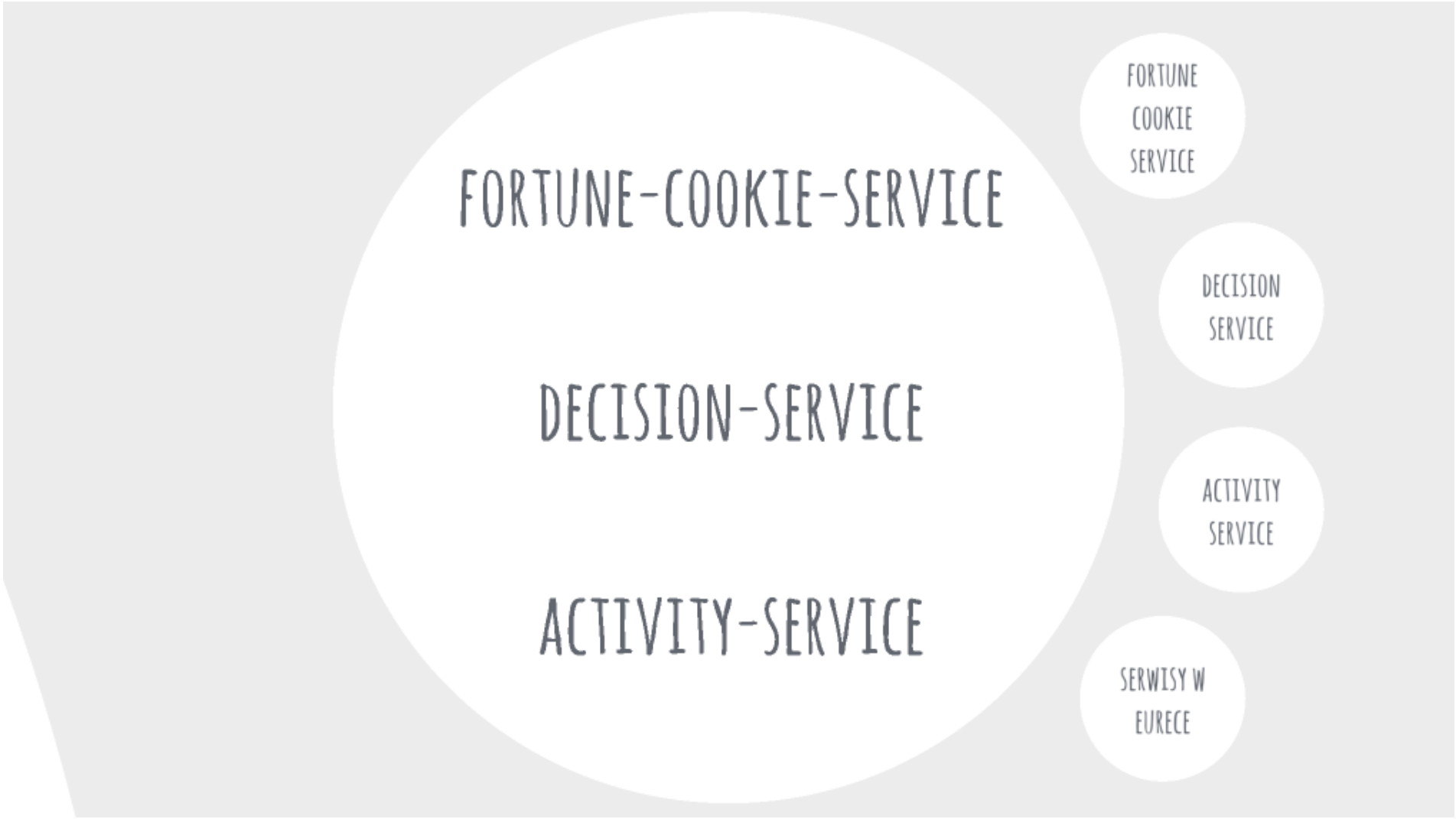

SPRING INITIALIZR bootstrap your application now
Generate a Maven Project with Java and Spring Boot 2.0.5
Project Metadata
Artifact coordinates
Group
workshop.sc
Artifact
eureka
Dependencies
Add Spring Boot Starters and dependencies to your application
Search for dependencies
Web, Security, JPA, Actuator, Devtools...
Selected Dependencies
Eureka Server
Generate Project ⌘ + ⏎
Don't know what to look for? Want more options? Switch to the full version.


Project
eureka-server ~/IdeaProjects/spring/sprigng_cloud/
.mvn
src
main
java
workshop
sc
eureka
eurekaserver
EurekaServerApplication
resources
application.yml
application.yml
EurekaServerApplication.java
server:
port: 8761
eureka:
client:
register-with-eureka: true
fetch-registry: false
instance:
appname: my-eureka

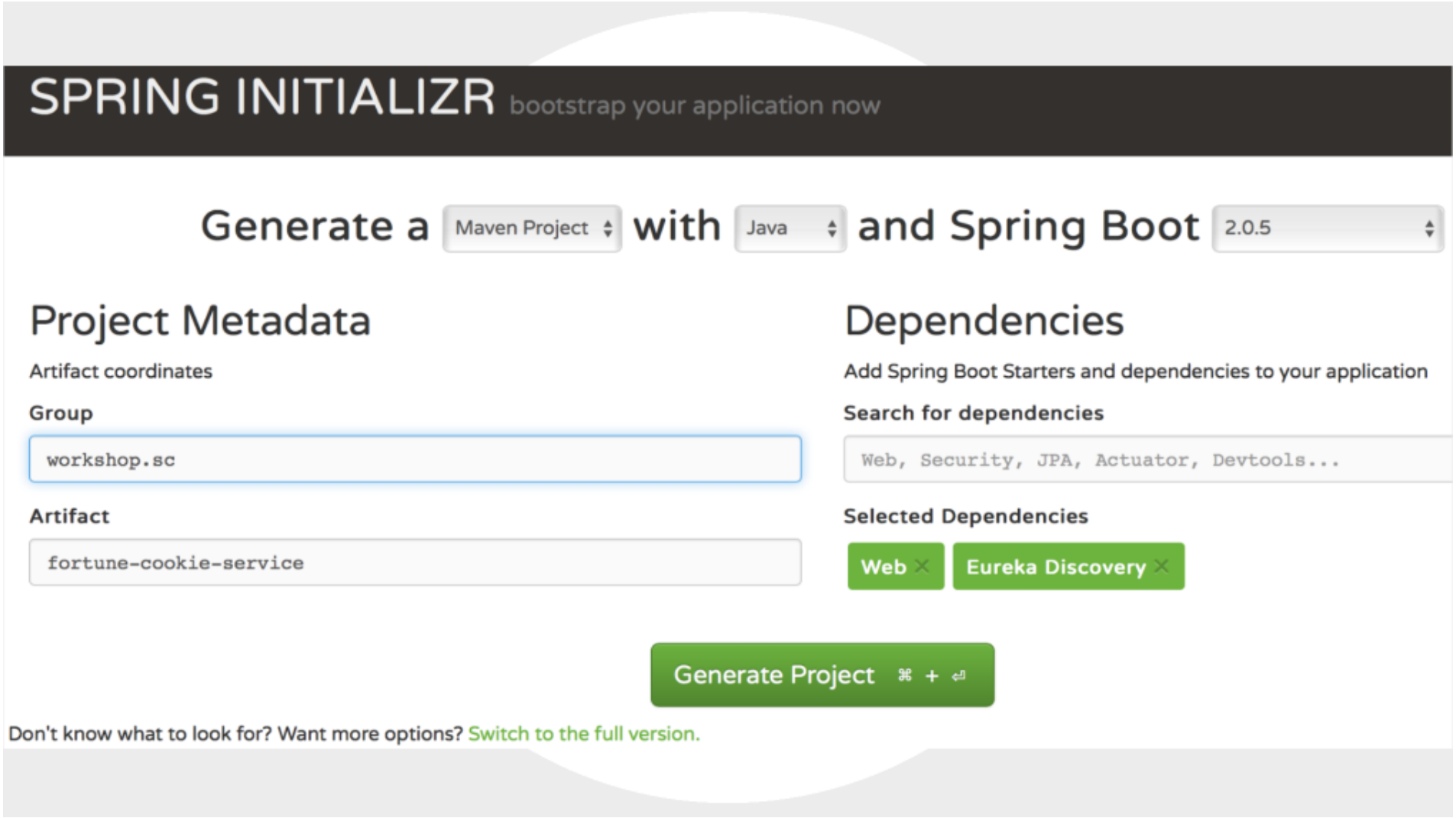

Project
eureka-server ~/IdeaProjects/spring/sprigng_cloud/
.mvn
src
main
java
workshop
sc
eureka
eurekaserver
EurekaServerApplication
resources
application.yml
application.yml
EurekaServerApplication.java
1 package workshop.sc.eureka.eurekaserver;
2
3 import ...
6
7 @SpringBootApplication
8 @EnableEurekaServer
9 public class EurekaServerApplication {
10
11 public static void main(String[] arg
14 }
15

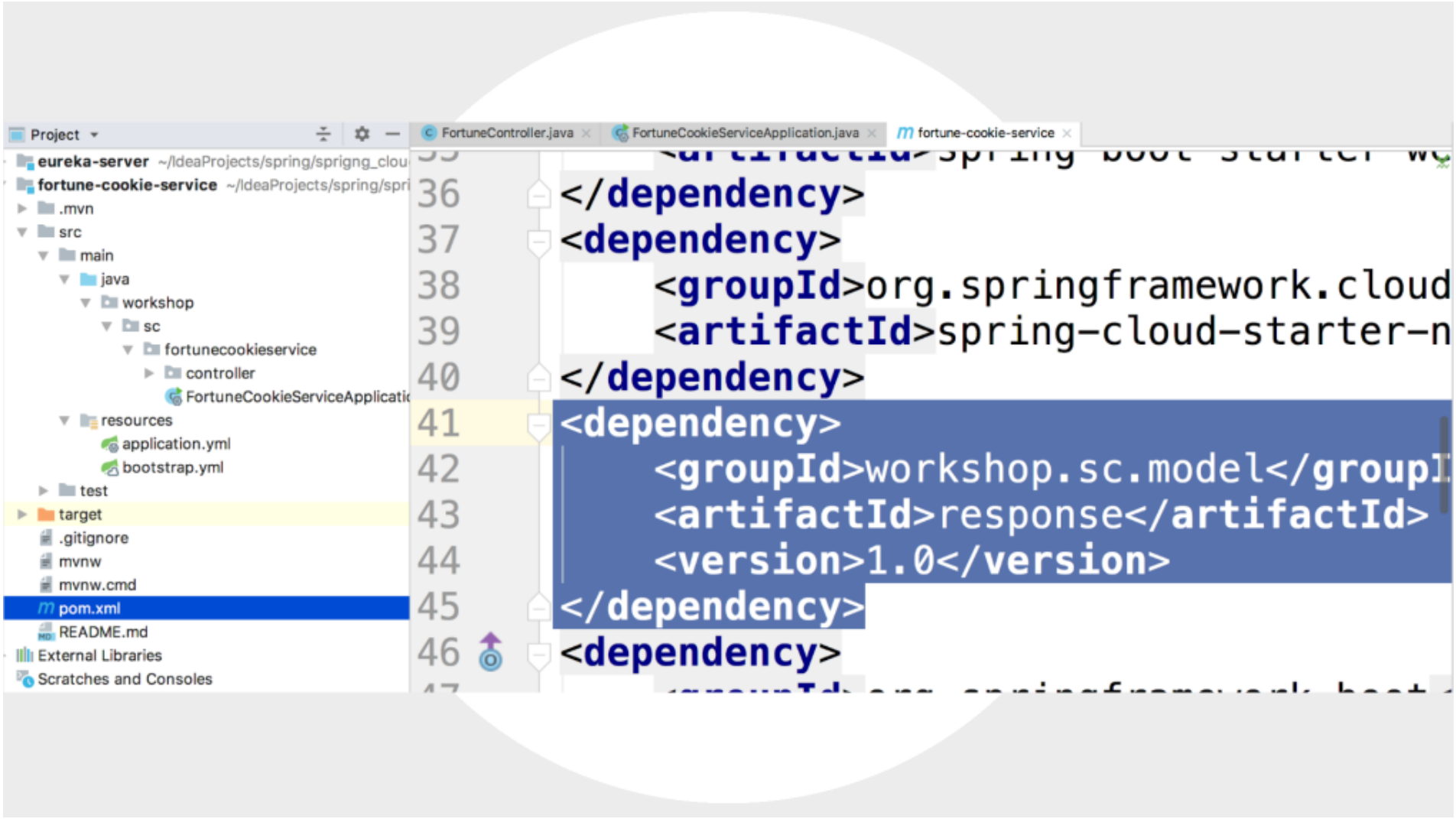

Project
workshop
sc
eureka
eurekaserver
EurekaServerApplication
resources
application.yml
bootstrap.yml
application.yml
server:
port: 8761
#eureka:
client:
register-with-eureka: true
fetch-registry: false
instance:
appname: my-eureka
Document 1/1
Run Dashboard:
EurekaServerApplication (1)
Console
Endpoints
Spring Boot
Running
EurekaServerApplication (1) :8761/
Configured
EurekaServerApplication
2018-10-02 10:47:43.748 WARN 14078 --- [nfoReplicator-0] c.n.discovery.InstanceInfoReplicator : There was a problem w
com.netflix.discovery.shared.transport.TransportException: Cannot execute request on any known server
at com.netflix.discovery.shared.transport.decorator.RetryableEurekaHttpClient.execute(RetryableEurekaHttpClient.java:1
at com.netflix.discovery.shared.transport.decorator.EurekaHttpClientDecorator.register(EurekaHttpClientDecorator.java:
at com.netflix.discovery.shared.transport.decorator.EurekaHttpClientDecorator$1.execute(EurekaHttpClientDecorator.java
at com.netflix.discovery.shared.transport.decorator.SessionedEurekaHttpClient.execute(SessionedEurekaHttpClient.java:7
at com.netflix.discovery.shared.transport.decorator.EurekaHttpClientDecorator.register(EurekaHttpClientDecorator.java:
at com.netflix.discovery.DiscoveryClient.register(DiscoveryClient.java:829) ~[eureka-client-1.9.0.jar:1.9.0]
at com.netflix.discovery.InstanceInfoReplicator.run(InstanceInfoReplicator.java:121) ~[eureka-client-1.9.0.jar:1.9.0]
at com.netflix.discovery.InstanceInfoReplicator$1.run(InstanceInfoReplicator.java:101) [eureka-client-1.9.0.jar:1.9.0]
at java.lang.Thread.run(Thread.java:748) [na:1.8.0_171]

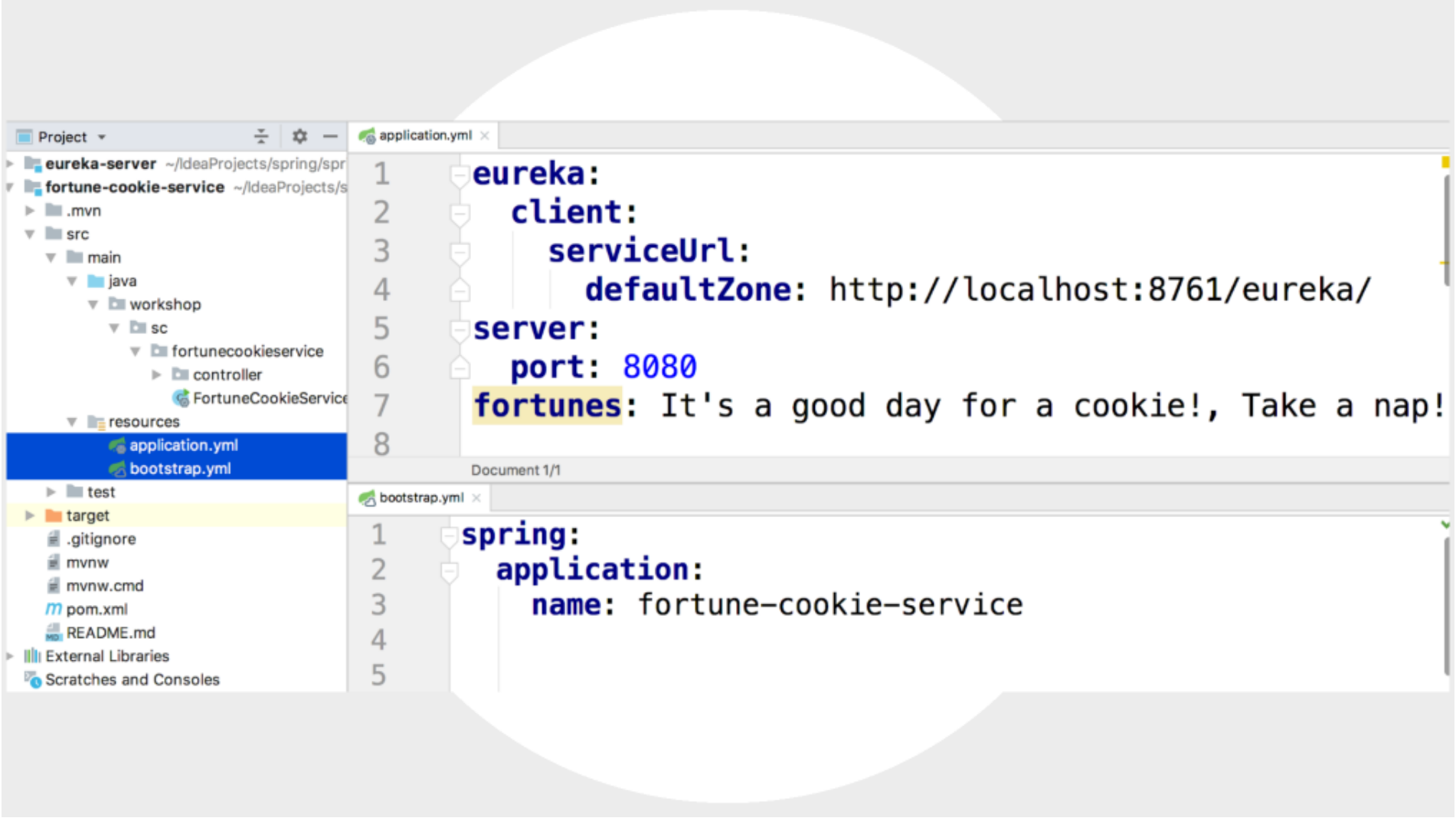

Eureka
localhost:8761
spring Eureka
HOME
LAST 1000 SINCE STARTUP
System Status
Environment
test
Data center
default
Current time
2018-10-02T10:51:46 +0200
Uptime
00:04
Lease expiration enabled
true
Renews threshold
1
Renews (last min)
4
DS Replicas
localhost
Instances currently registered with Eureka
Application
AMIs
Availability Zones
Status
EUREKA-SERVER
n/a (1)
(1)
UP (1) - 192.168.8.100:eureka-server:8761

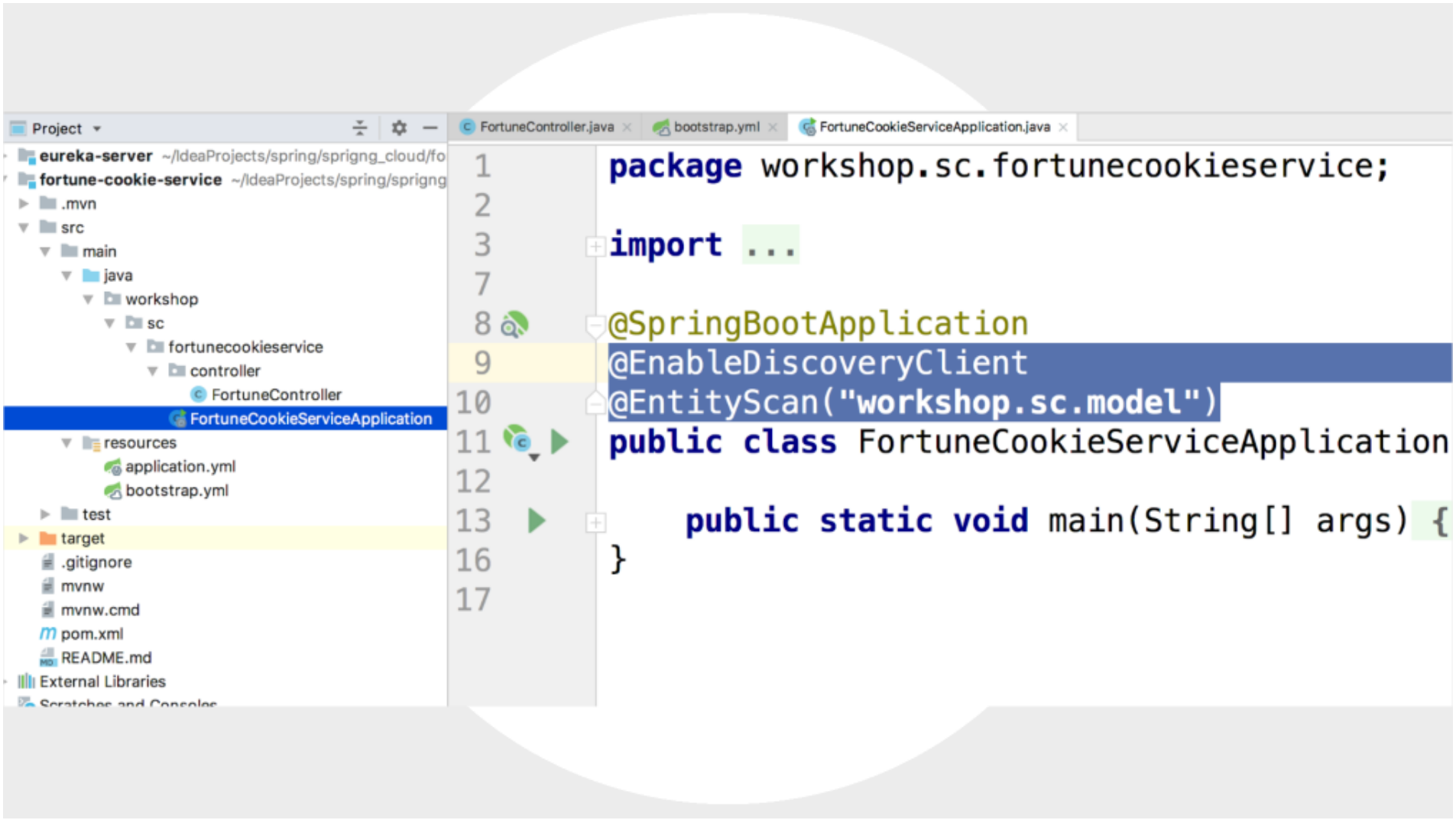

FORTUNE-COOKIE-SERVICE
DECISION-SERVICE
ACTIVITY-SERVICE
FORTUNE COOKIE SERVICE
DECISION SERVICE
ACTIVITY SERVICE
SERWISY W EURECE

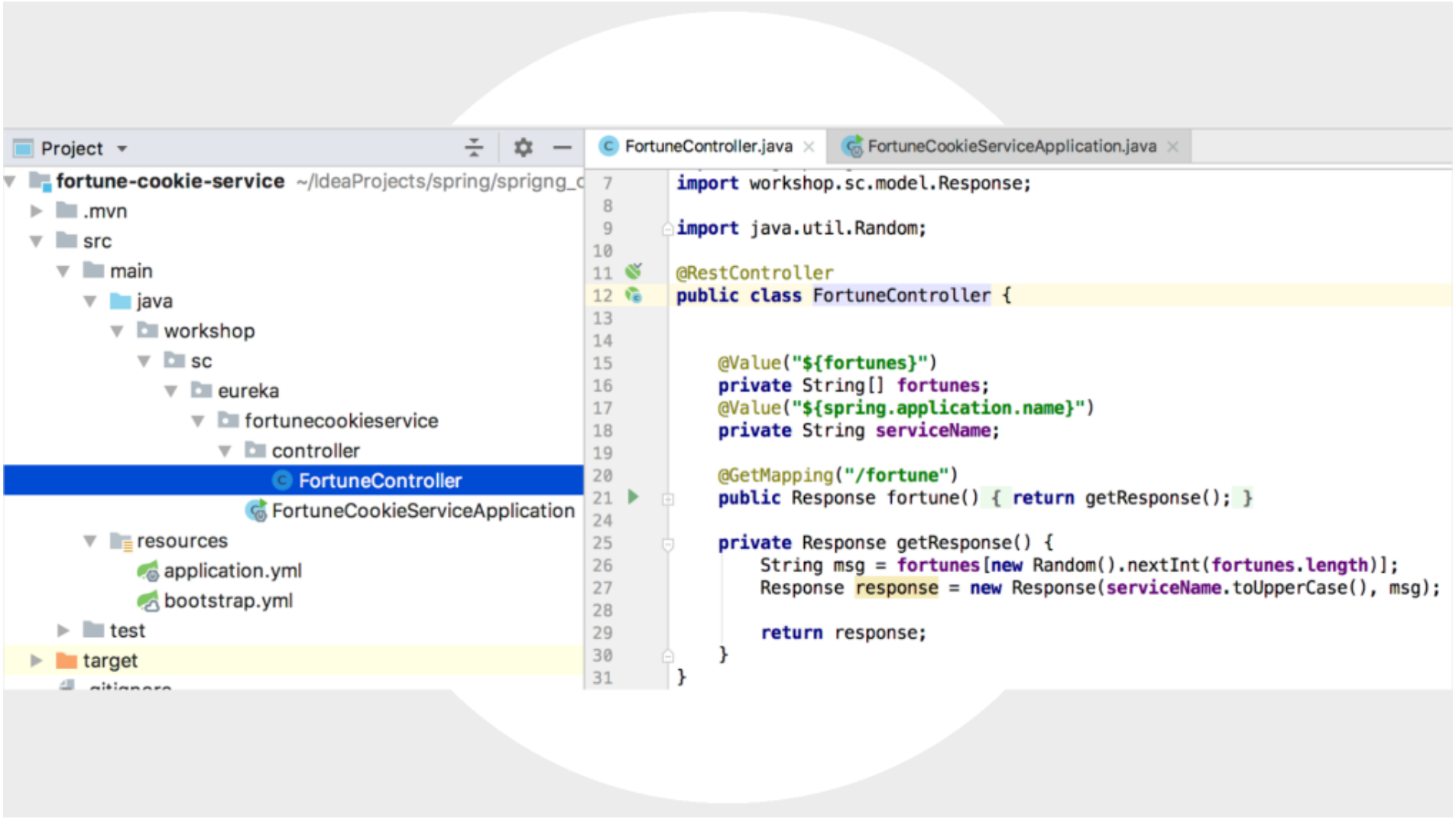

fortune cookie service
decision service
activity service
naming server
config server
SZKIELET PROJEKTU
MAVEN
PLIKI KONFIGURACYJNE
@
KONTROLER
TEST

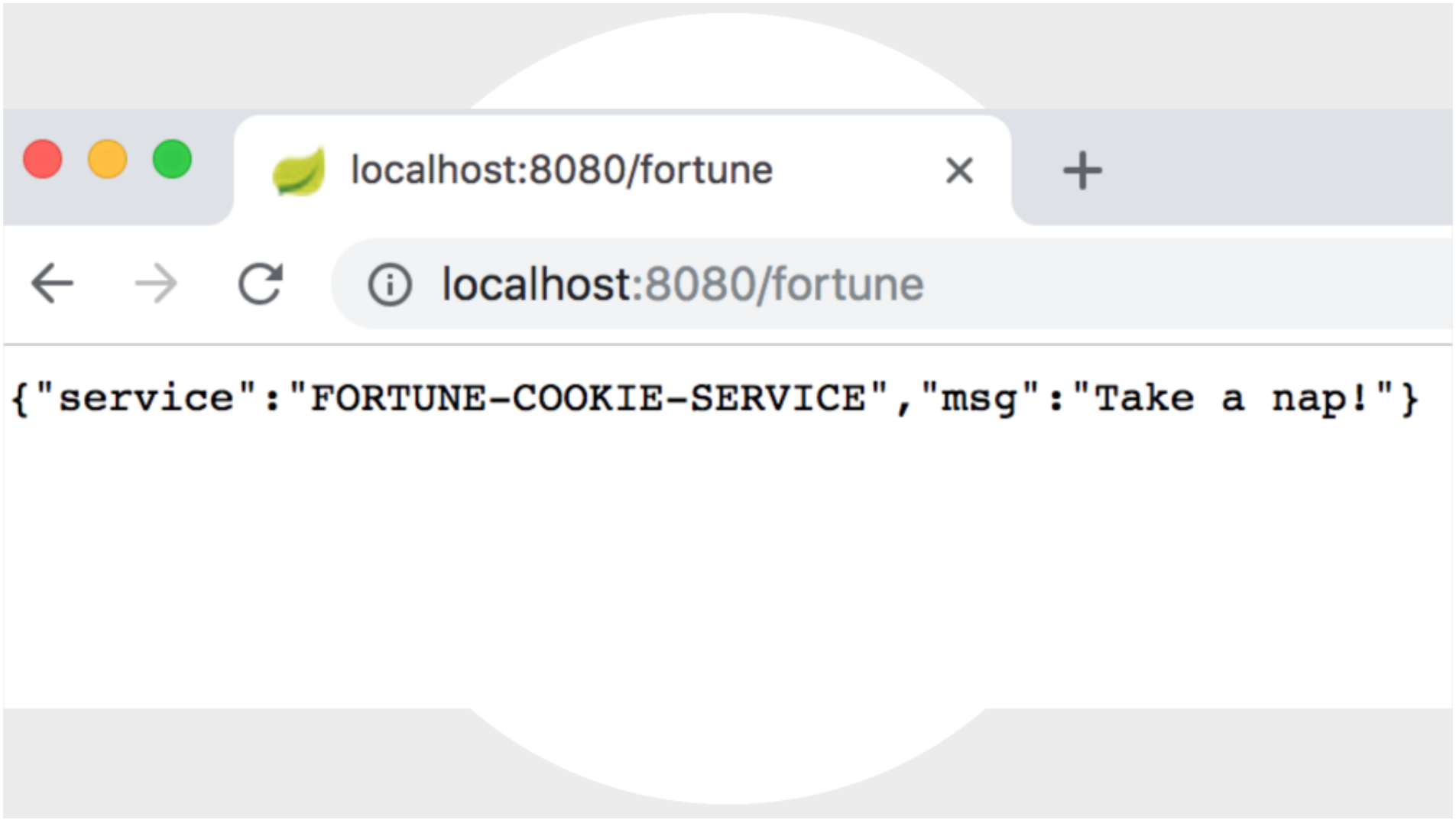

SPRING INITIALIZR bootstrap your application now
Generate a Maven Project with Java and Spring Boot 2.0.5
Project Metadata
Artifact coordinates
Group
workshop.sc
Artifact
fortune-cookie-service
Dependencies
Add Spring Boot Starters and dependencies to your application
Search for dependencies
Web, Security, JPA, Actuator, Devtools...
Selected Dependencies
Web
Eureka Discovery
Generate Project ⌘ + ⏎
Don't know what to look for? Want more options? Switch to the full version.

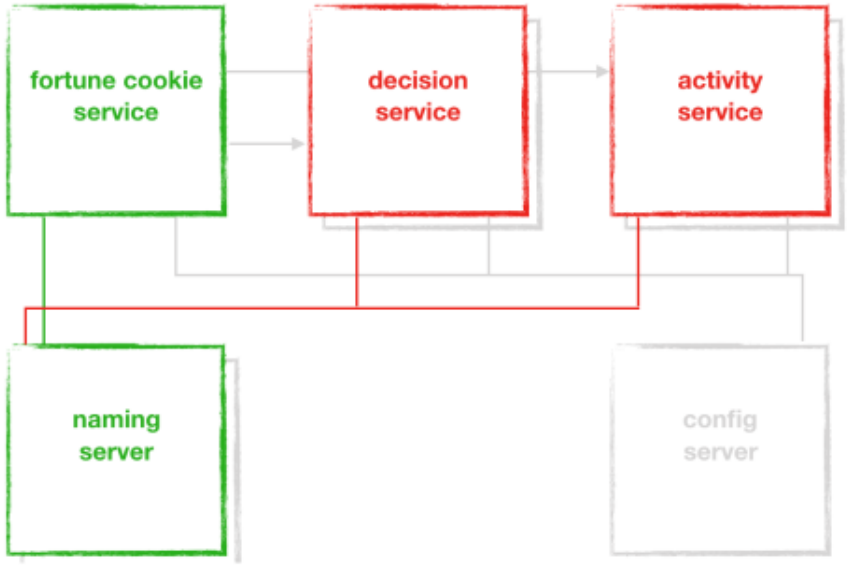

Project
eureka-server ~/IdeaProjects/spring/sprigng_clou
fortune-cookie-service ~/IdeaProjects/spring/spri
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneCookieServiceApplicatio
resources
application.yml
bootstrap.yml
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
External Libraries
Scratches and Consoles
FortuneController.java
FortuneCookieServiceApplication.java
fortune-cookie-service
36 </dependency>
37 <dependency>
38 <groupId>org.springframework.cloud
39 <artifactId>spring-cloud-starter-n
40 </dependency>
41 <dependency>
42 <groupId>workshop.sc.model</groupI
43 <artifactId>response</artifactId>
44 <version>1.0</version>
45 </dependency>
46 <dependency>


Project
eureka-server ~/IdeaProjects/spring/spr
fortune-cookie-service ~/IdeaProjects/s
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneCookieServic
resources
application.yml
bootstrap.yml
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
External Libraries
Scratches and Consoles
application.yml
eureka:
client:
serviceUrl:
defaultZone: http://localhost:8761/eureka/
server:
port: 8080
fortunes: It's a good day for a cookie!, Take a nap!
Document 1/1
bootstrap.yml
spring:
application:
name: fortune-cookie-service


Project
eureka-server ~/IdeaProjects/spring/sprigng_cloud/fo
fortune-cookie-service ~/IdeaProjects/spring/sprigng
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
FortuneCookieServiceApplication
resources
application.yml
bootstrap.yml
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
External Libraries
FortuneController.java
bootstrap.yml
FortuneCookieServiceApplication.java
1 package workshop.sc.fortunecookieservice;
2
3 import ...
7
8 @SpringBootApplication
9 @EnableDiscoveryClient
10 @EntityScan("workshop.sc.model")
11 public class FortuneCookieServiceApplication
12
13 public static void main(String[] args) {
16 }
17


Project
fortune-cookie-service ~/IdeaProjects/spring/sprigng_c
.mvn
src
main
java
workshop
sc
eureka
fortunecookieservice
controller
FortuneController
FortuneCookieServiceApplication
resources
application.yml
bootstrap.yml
test
target
FortuneController.java
FortuneCookieServiceApplication.java
7 import workshop.sc.model.Response;
8
9 import java.util.Random;
10
11 @RestController
12 public class FortuneController {
13
14
15 @Value("${fortunes}")
16 private String[] fortunes;
17 @Value("${spring.application.name}")
18 private String serviceName;
19
20 @GetMapping("/fortune")
21 public Response fortune() { return getResponse(); }
24
25 private Response getResponse() {
26 String msg = fortunes[new Random().nextInt(fortunes.length)];
27 Response response = new Response(serviceName.toUpperCase(), msg);
28
29 return response;
30 }
31 }


localhost:8080/fortune
localhost:8080/fortune
{"service":"FORTUNE-COOKIE-SERVICE","msg":"Take a nap!"}



IMPLEMENTACJA

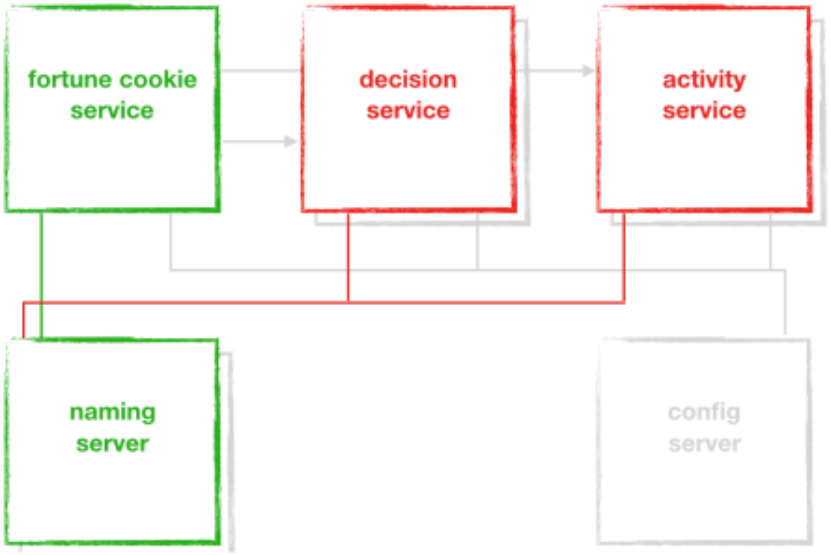
TEST

Analogicznie do FCS

http://localhost/8000/decision

It's a good idea, It's a bad day etc.


localhost:8000/decision
localhost:8000/decision
{"service":"DECISION-SERVICE","msg":"It's a good day"}



IMPLEMENTACJA

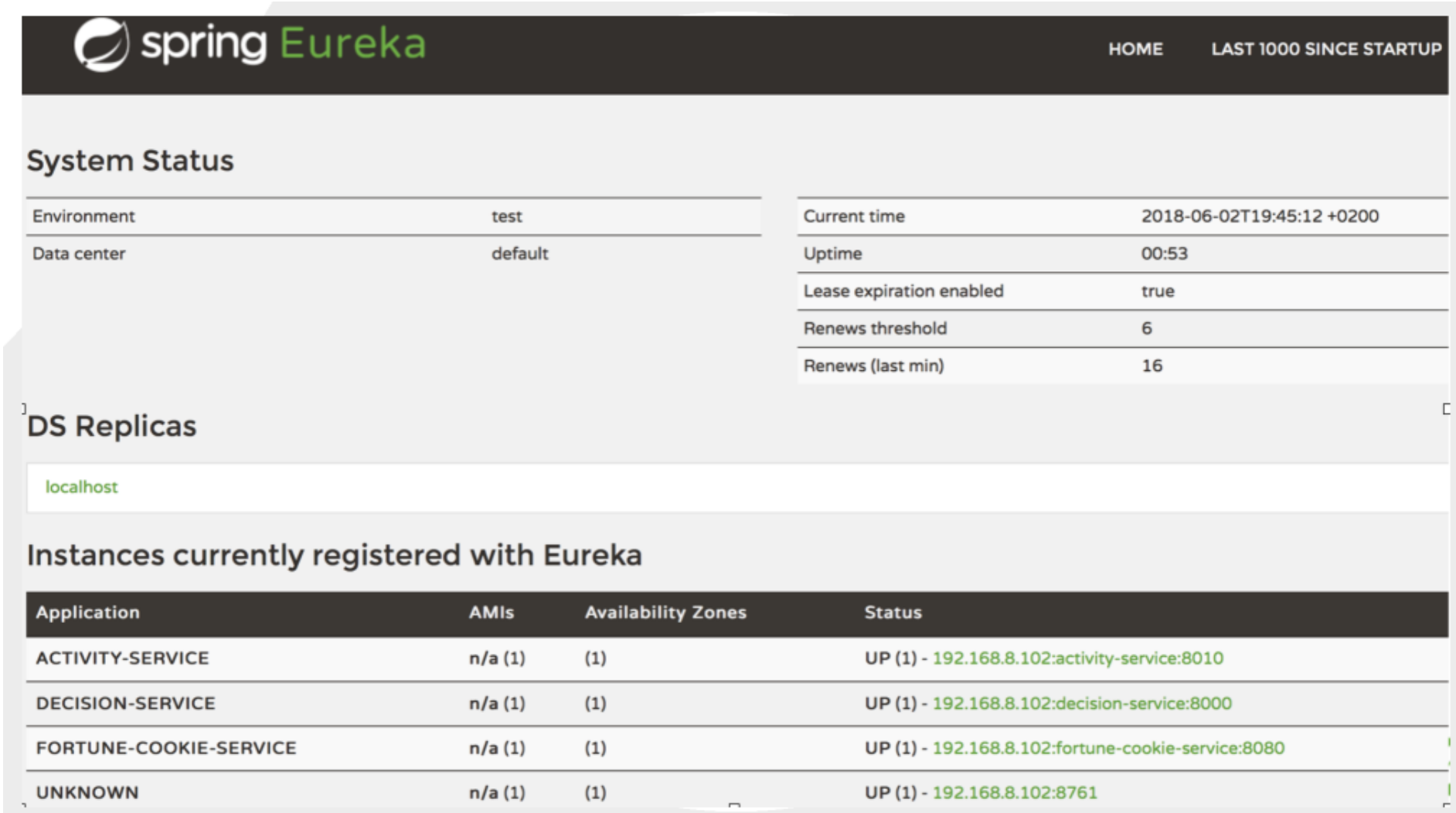
TEST

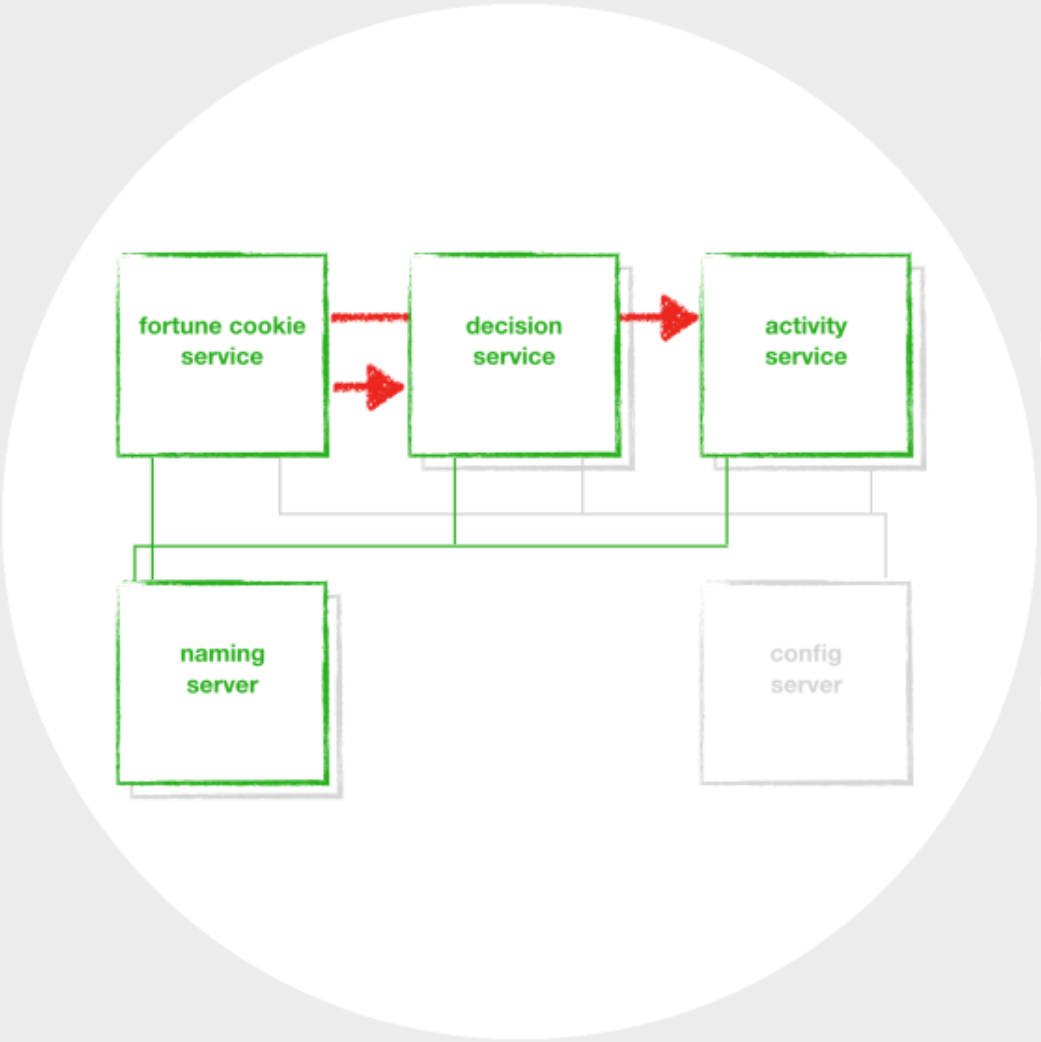
ANALOGICZNIE DO FCS

HTTP://LOCALHOST/8010/ACTIVITY

TO RUN A MARATHON, FOR COOKIES ETC.


localhost:8010/activity
localhost:8010/activity
{"service":"ACTIVITY-SERVICE","msg":"take a nap"}

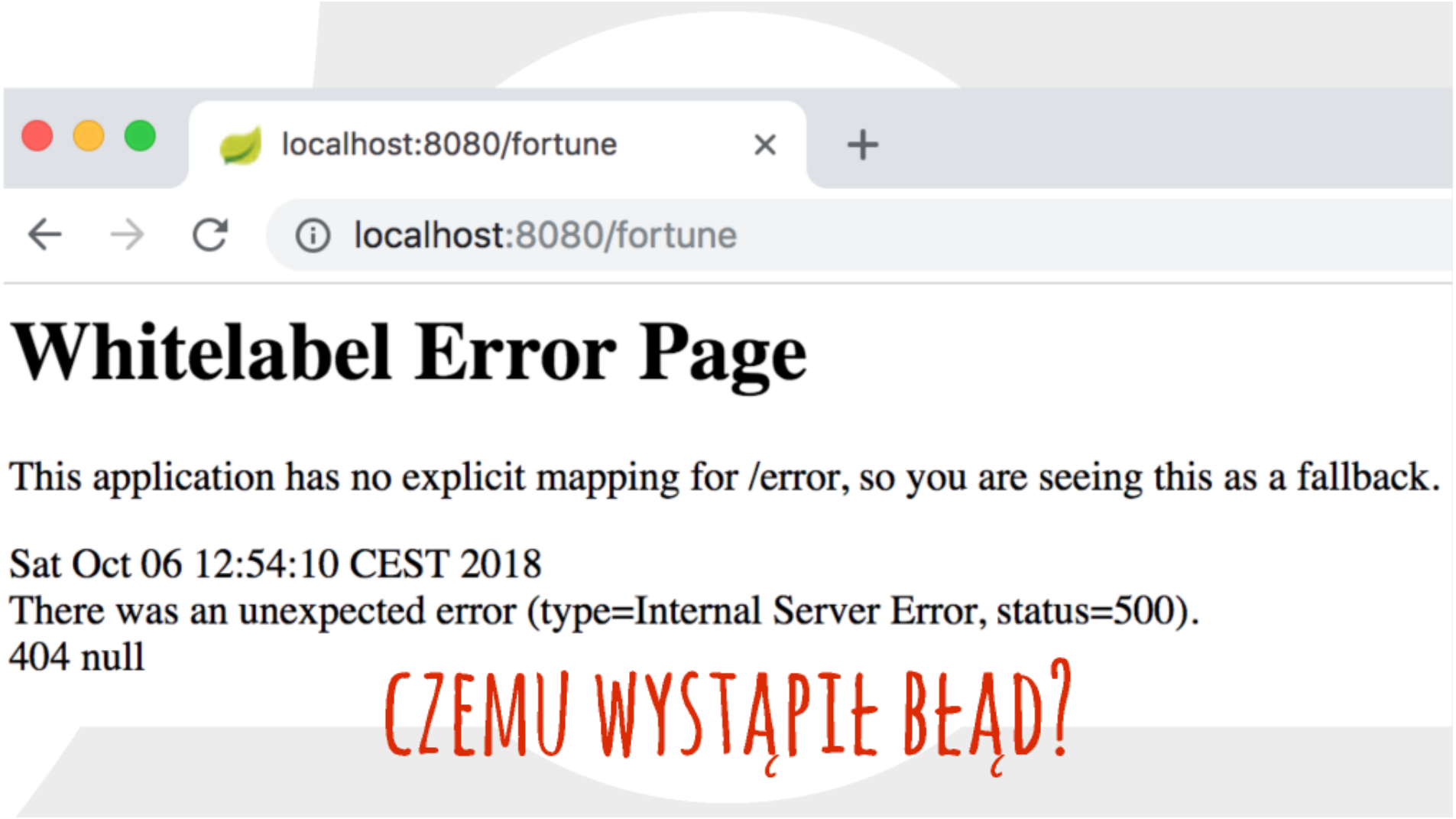

spring Eureka
HOME
LAST 1000 SINCE STARTUP
System Status
Environment
test
Data center
default
Current time
2018-06-02T19:45:12 +0200
Uptime
00:53
Lease expiration enabled
true
Renews threshold
6
Renews (last min)
16
DS Replicas
localhost
Instances currently registered with Eureka
Application
AMIs
Availability Zones
Status
ACTIVITY-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:activity-service:8010
DECISION-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:decision-service:8000
FORTUNE-COOKIE-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:fortune-cookie-service:8080
UNKNOWN
n/a (1)
(1)
UP (1) - 192.168.8.102:8761

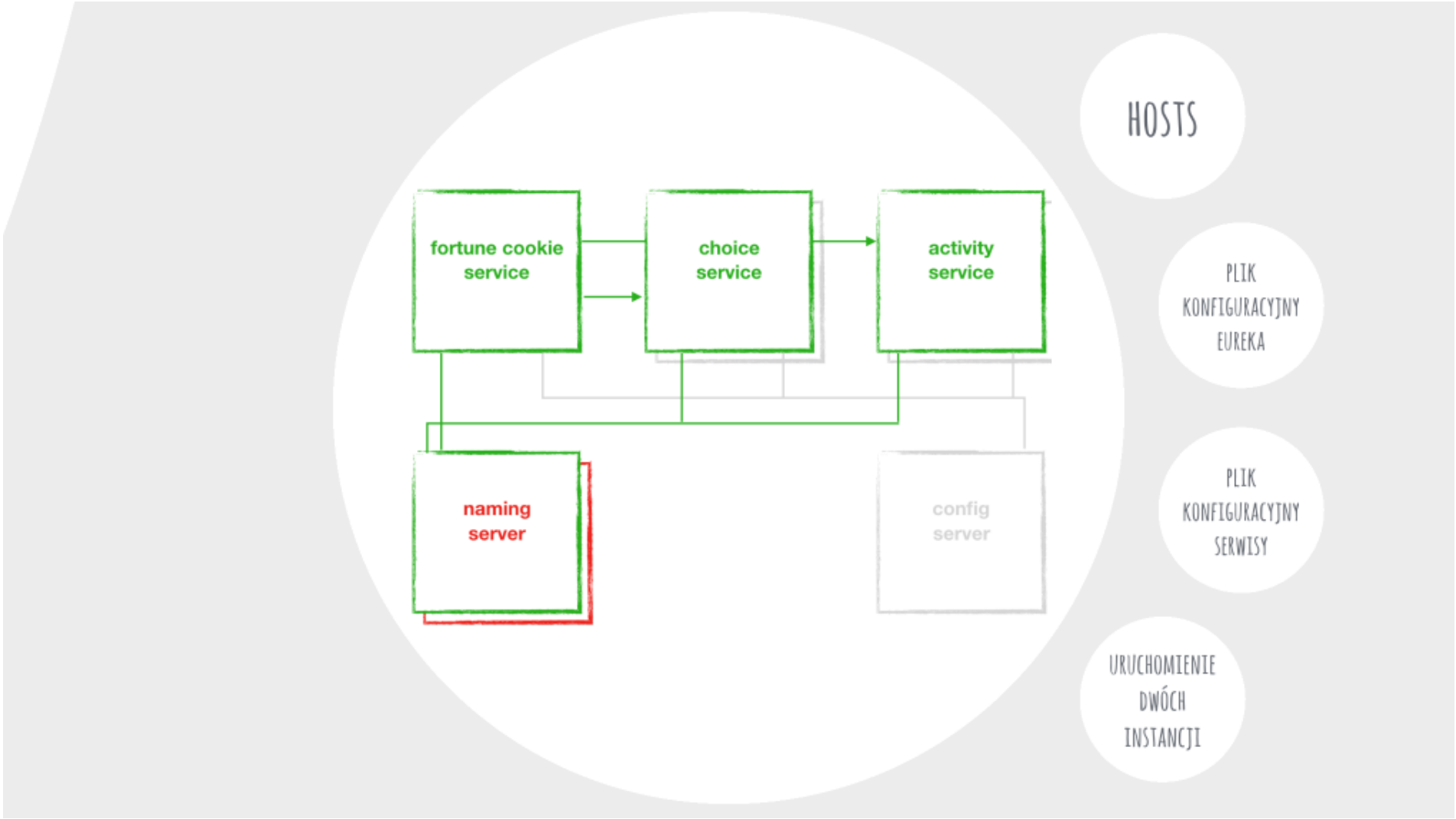


KONTROLER

TEST

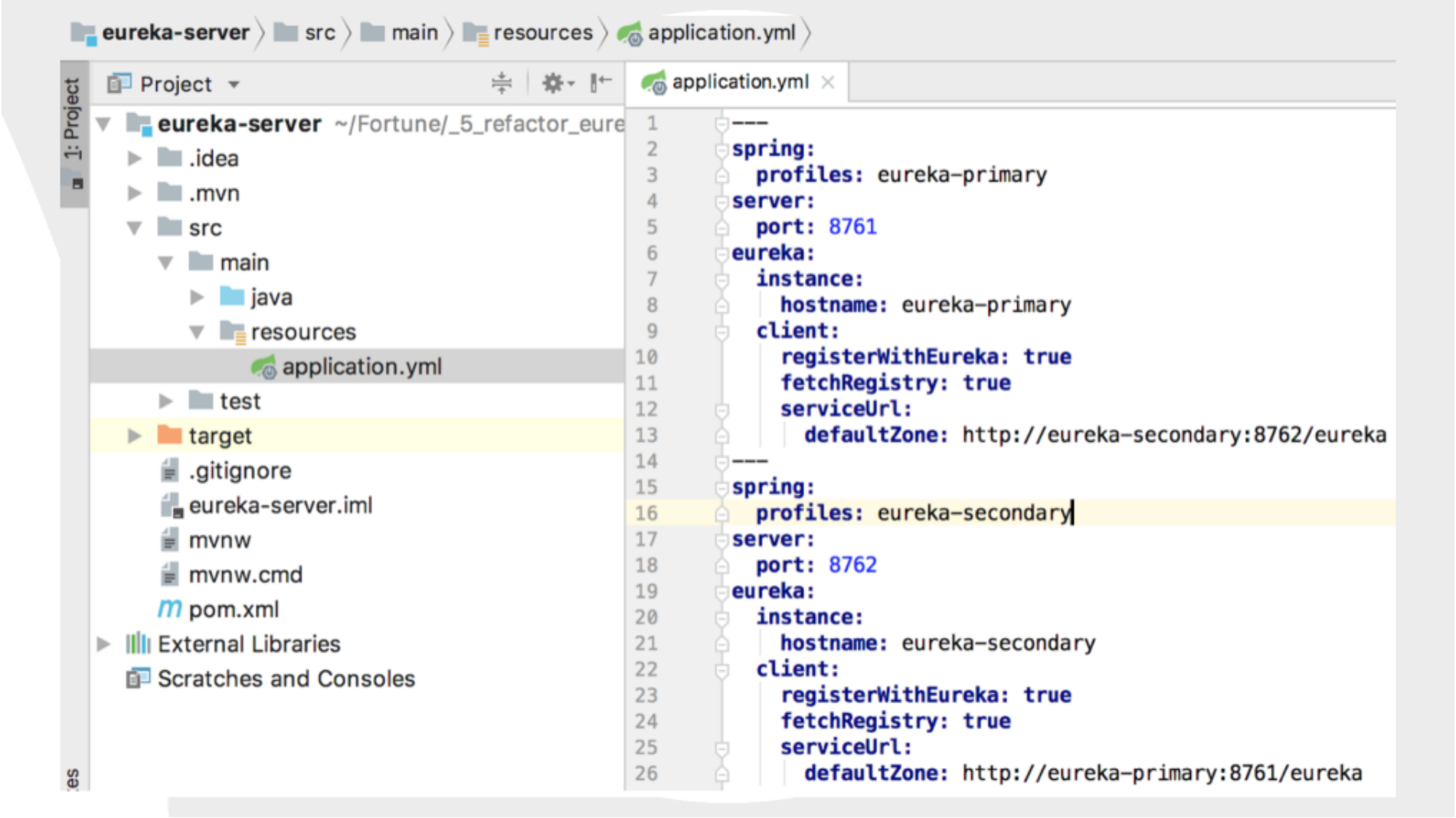
```java
@RestController
public class FortuneController {

    @Autowired
    private org.springframework.cloud.client.discovery.DiscoveryClient discoveryClient;

    @Value("${fortunes}")
    private String[] fortunes;
    @Value("${spring.application.name}")
    private String serviceName;

    @GetMapping("/fortune")
    public Response fortune() { return getResponse(); }

    private Response getResponse() {

        return getFortune();
    }

    private Response getFortune() {
        Response decision = getDataFromService("DECISION-SERVICE");
        Response activity = getDataFromService("ACTIVITY-SERVICE");
        return new Response(serviceName.toUpperCase(), msg: decision.getMsg() + " " + activity.getMsg());
    }

    private Response getDataFromService(String service) {
        List<ServiceInstance> list = discoveryClient.getInstances(service);
        if (list != null && list.size() > 0) {
            URI uri = list.get(0).getUri();
            if (uri != null) {
                return new RestTemplate().getForObject(uri, Response.class);
            }
        }
        return null;
    }
}
```


localhost:8080/fortune
localhost:8080/fortune
Whitelabel Error Page
This application has no explicit mapping for /error, so you are seeing this as a fallback.
Sat Oct 06 12:54:10 CEST 2018
There was an unexpected error (type=Internal Server Error, status=500).
404 null
CZEMU WYSTĄPIŁ BŁĄD?


fortune cookie service
choice service
activity service
naming server
config server
HOSTS
PLIK KONFIGURACYJNY EUREKA
PLIK KONFIGURACYJNY SERWISY
URUCHOMIENIE DWÓCH INSTANCJI

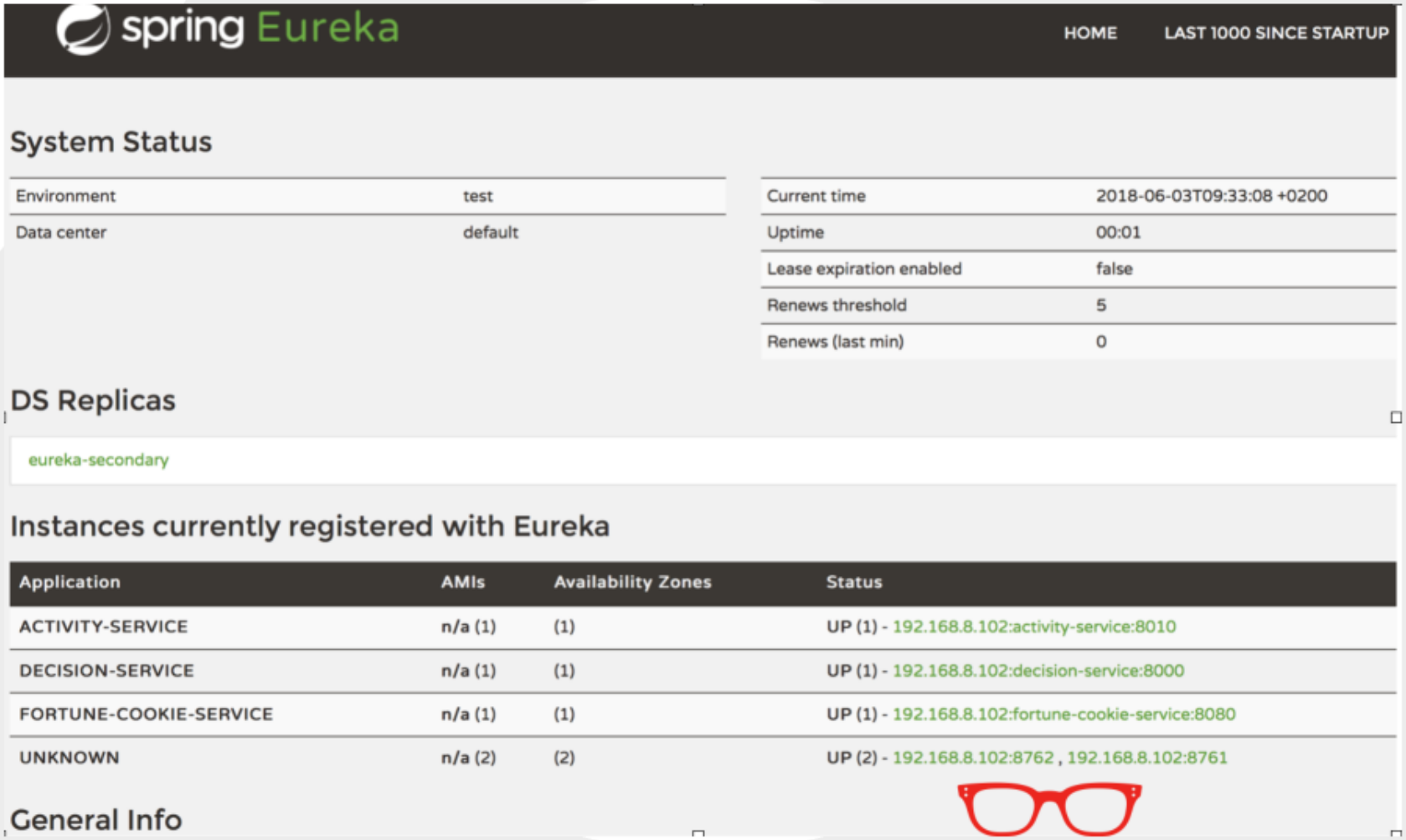
c:\WINDOWS\system32\drivers\etc\hosts

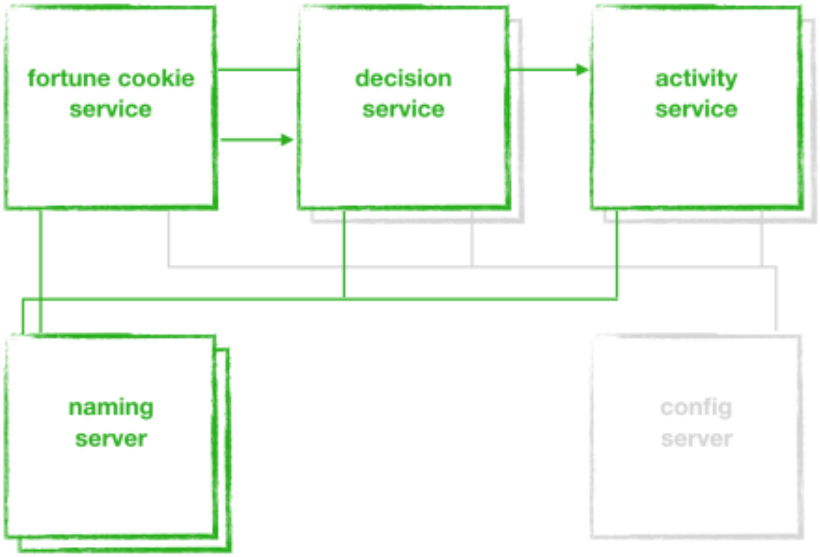
```
# START Eureka
127.0.0.1    eureka-primary
127.0.0.1    eureka-secondary
# END Eureka
```

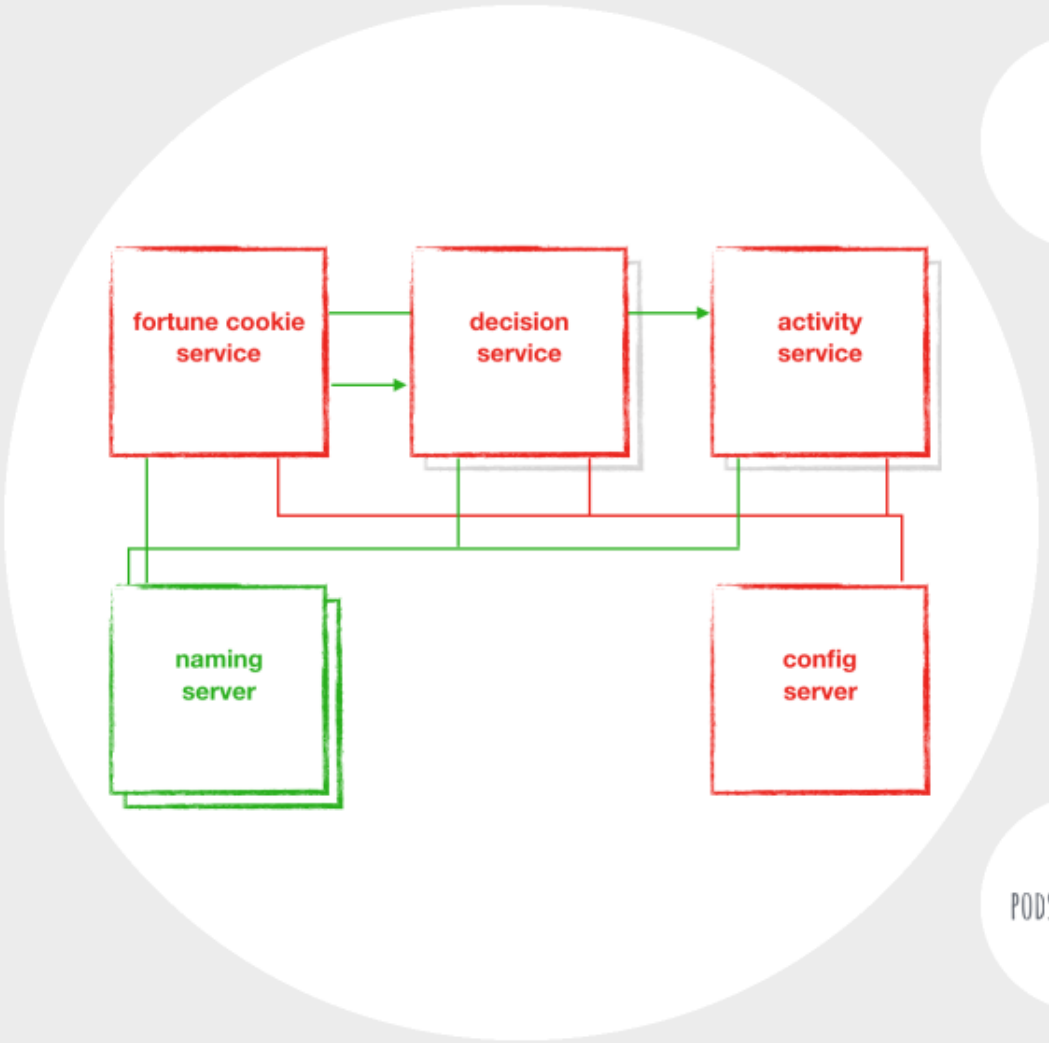

eureka-server 〉 src 〉 main 〉 resources 〉 application.yml 〉
1: Project
Project
eureka-server ~/Fortune/_5_refactor_eure
.idea
.mvn
src
main
java
resources
application.yml
test
target
.gitignore
eureka-server.iml
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
application.yml
---
spring:
profiles: eureka-primary
server:
port: 8761
eureka:
instance:
hostname: eureka-primary
client:
registerWithEureka: true
fetchRegistry: true
serviceUrl:
defaultZone: http://eureka-secondary:8762/eureka
---
spring:
profiles: eureka-secondary
server:
port: 8762
eureka:
instance:
hostname: eureka-secondary
client:
registerWithEureka: true
fetchRegistry: true
serviceUrl:
defaultZone: http://eureka-primary:8761/eureka

YML: FORTUNE, DECISION, ACTIVITY

```yaml
eureka:
  client:
    serviceUrl:
      defaultZone: http://eureka-primary:8761/eureka,http://eureka-secondary:8762/eureka
```


spring Eureka
HOME
LAST 1000 SINCE STARTUP
System Status
Environment
test
Data center
default
Current time
2018-06-03T09:33:08 +0200
Uptime
00:01
Lease expiration enabled
false
Renews threshold
5
Renews (last min)
0
DS Replicas
eureka-secondary
Instances currently registered with Eureka
Application
AMIs
Availability Zones
Status
ACTIVITY-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:activity-service:8010
DECISION-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:decision-service:8000
FORTUNE-COOKIE-SERVICE
n/a (1)
(1)
UP (1) - 192.168.8.102:fortune-cookie-service:8080
UNKNOWN
n/a (2)
(2)
UP (2) - 192.168.8.102:8762 , 192.168.8.102:8761
General Info


fortune cookie service
decision service
activity service
naming server
config server



GIT

CONFIG SERVER

PODŁĄCZENIE SERWISÓW

PODSUMOWANIE

git init

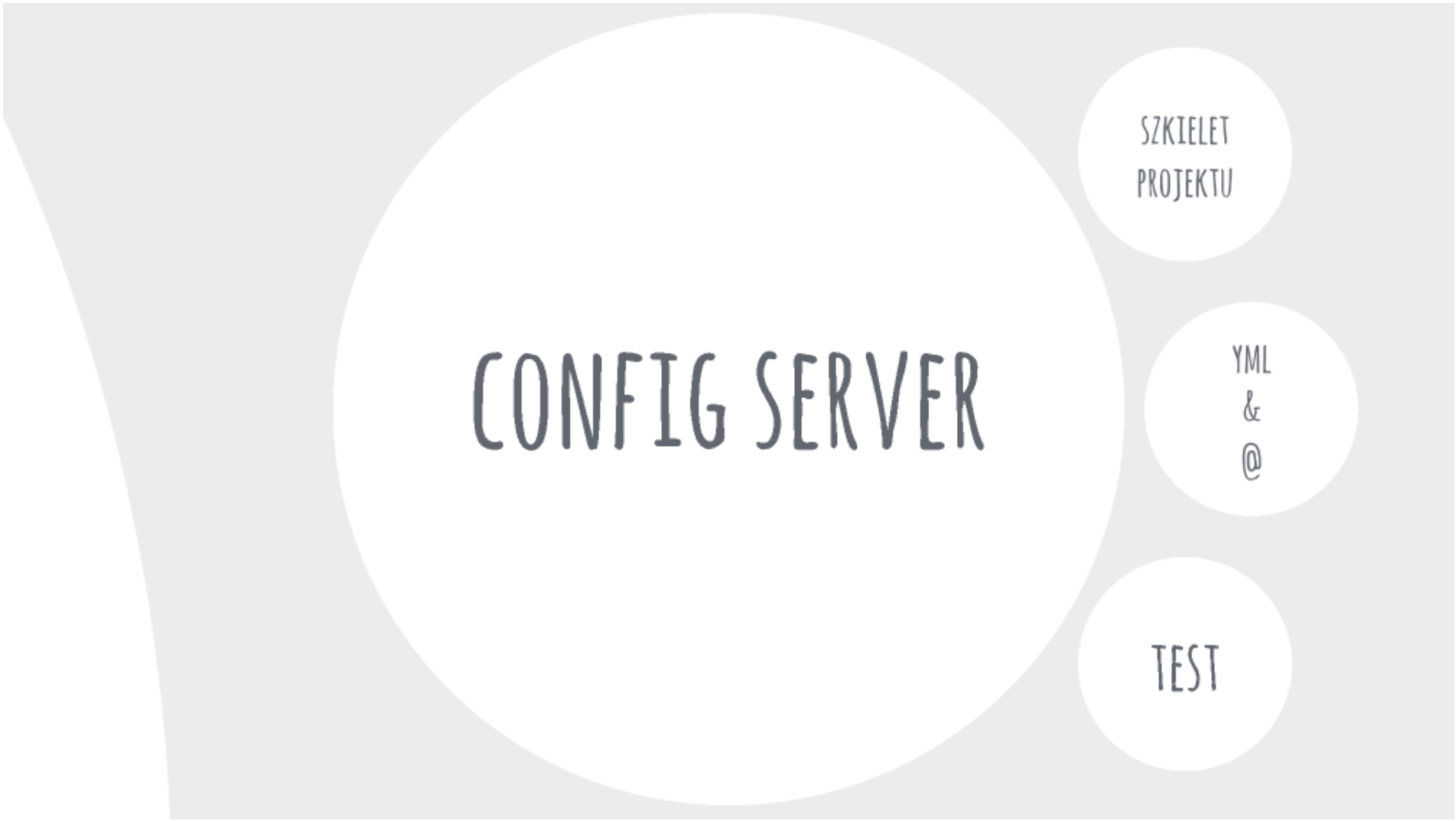
vi activity-service.yml

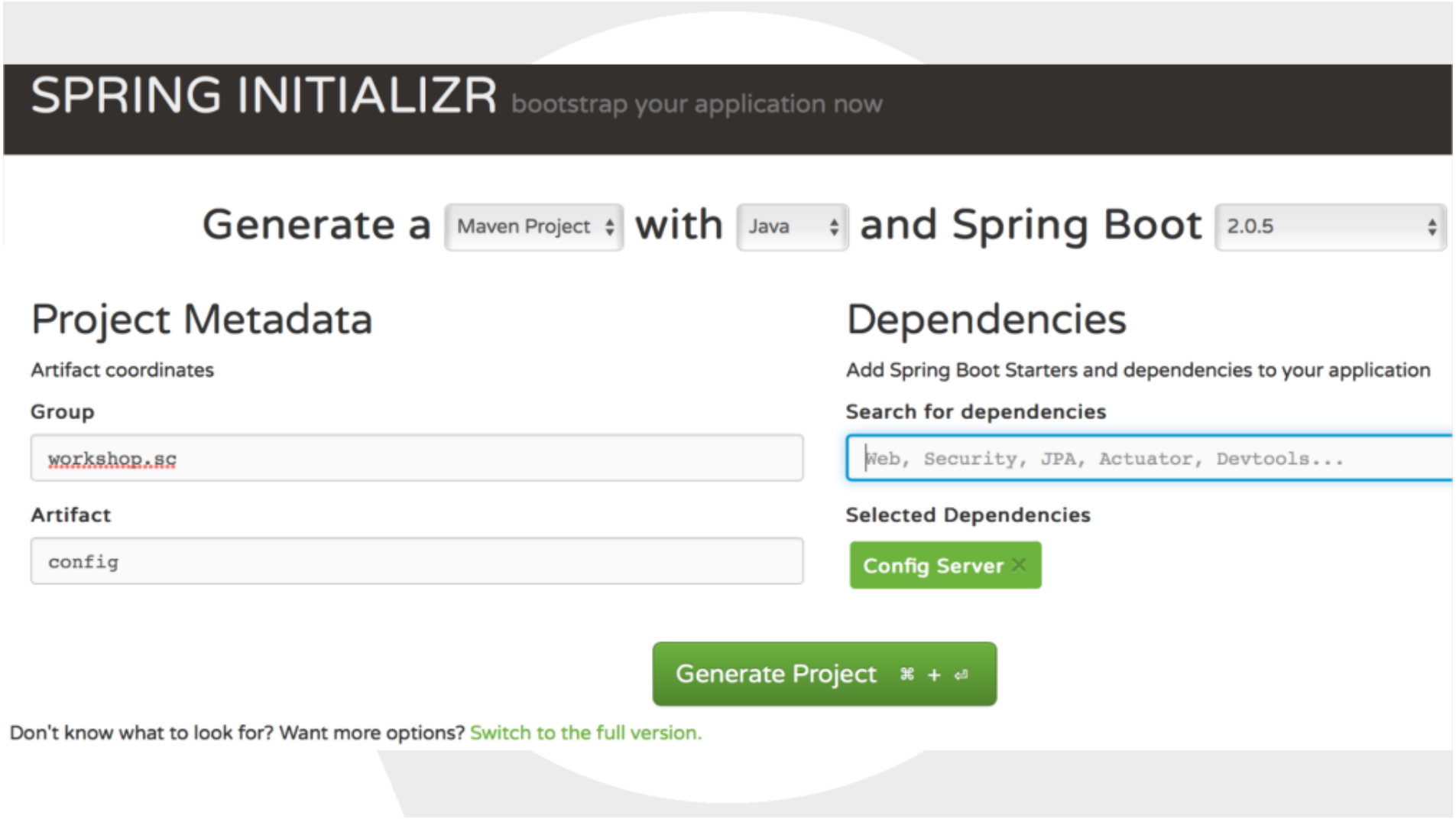
git add -A && git commit -m 'as'

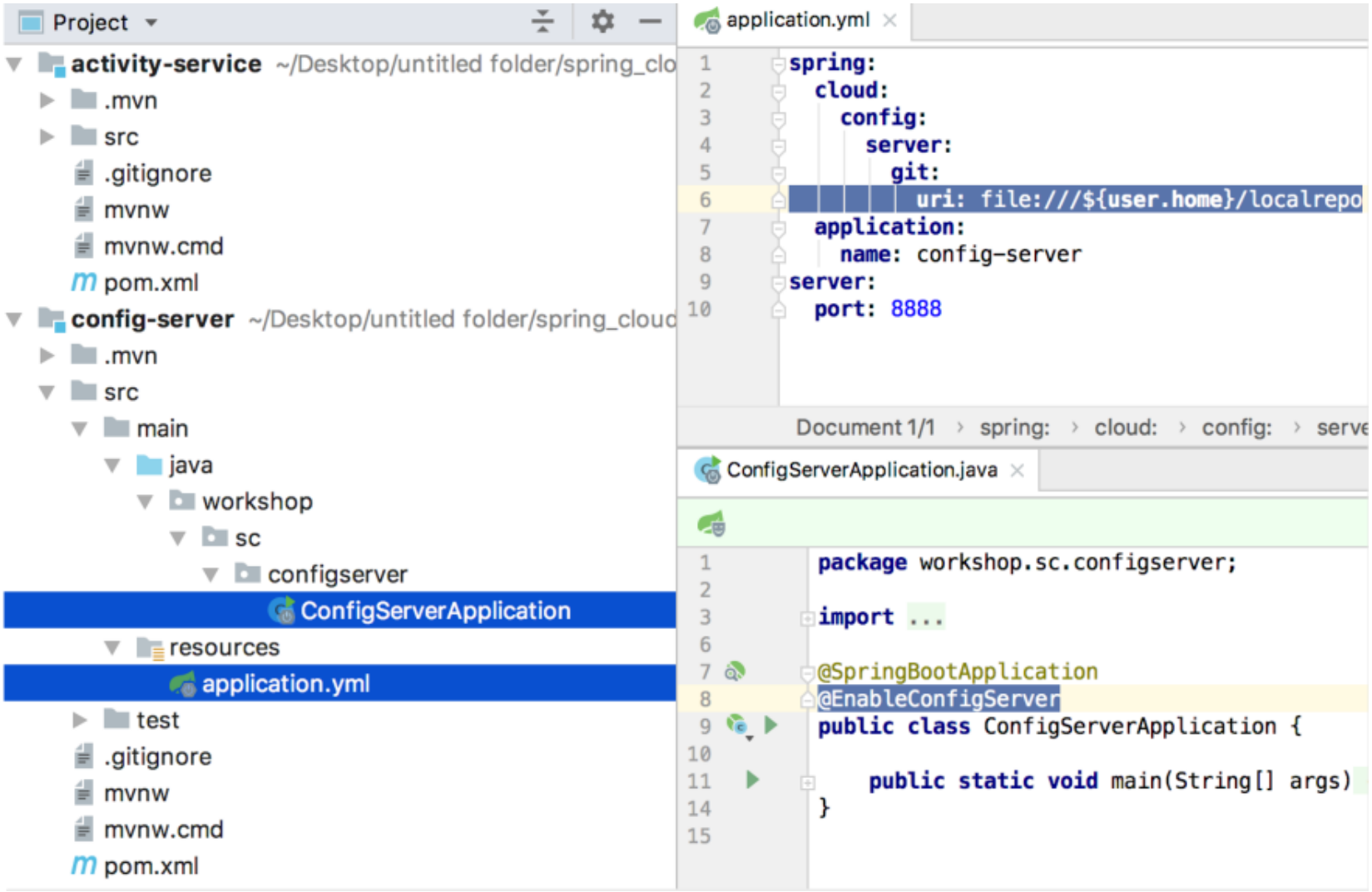
przy commicie ustawienie fake danych

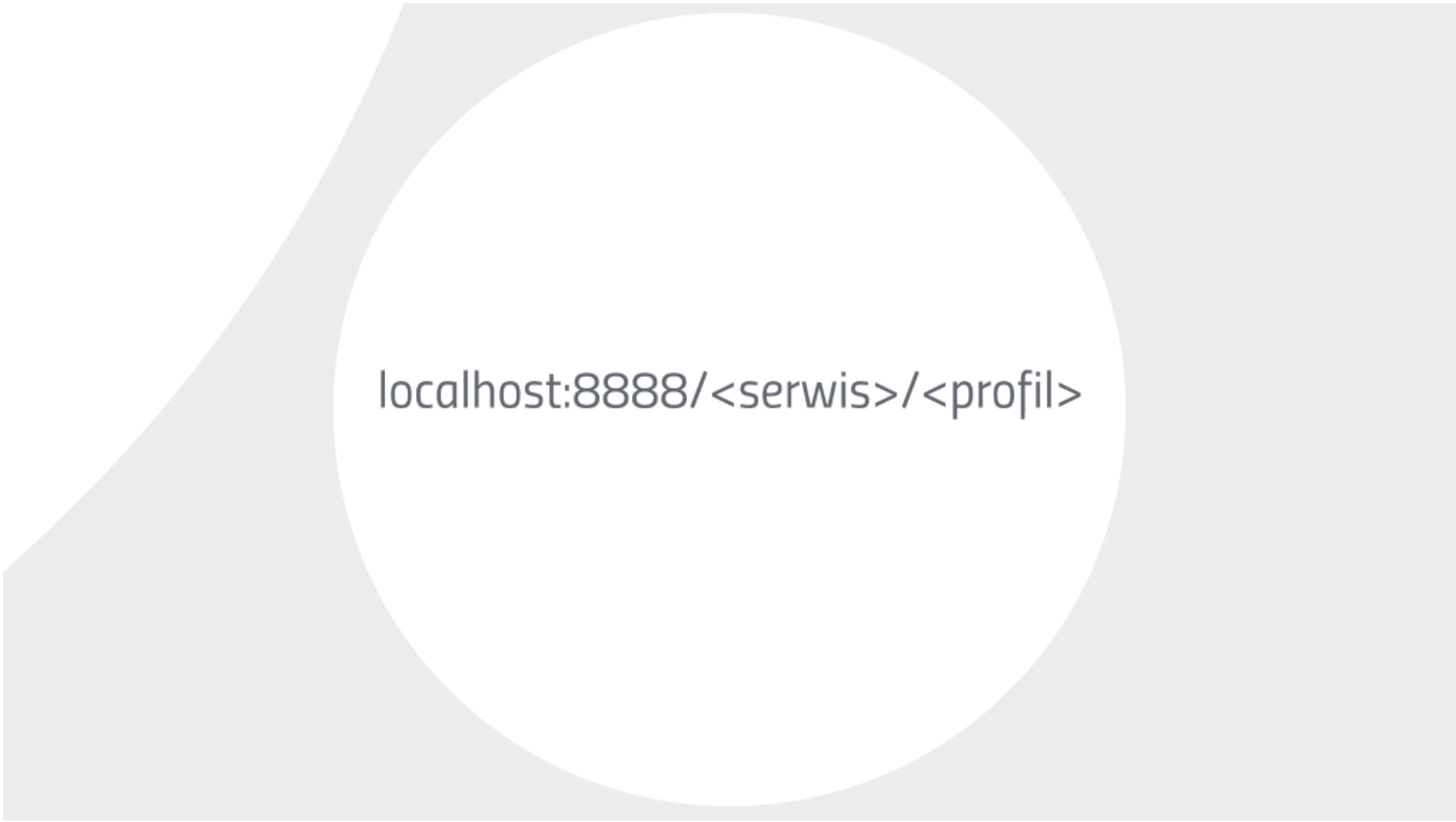
activity-service.yml

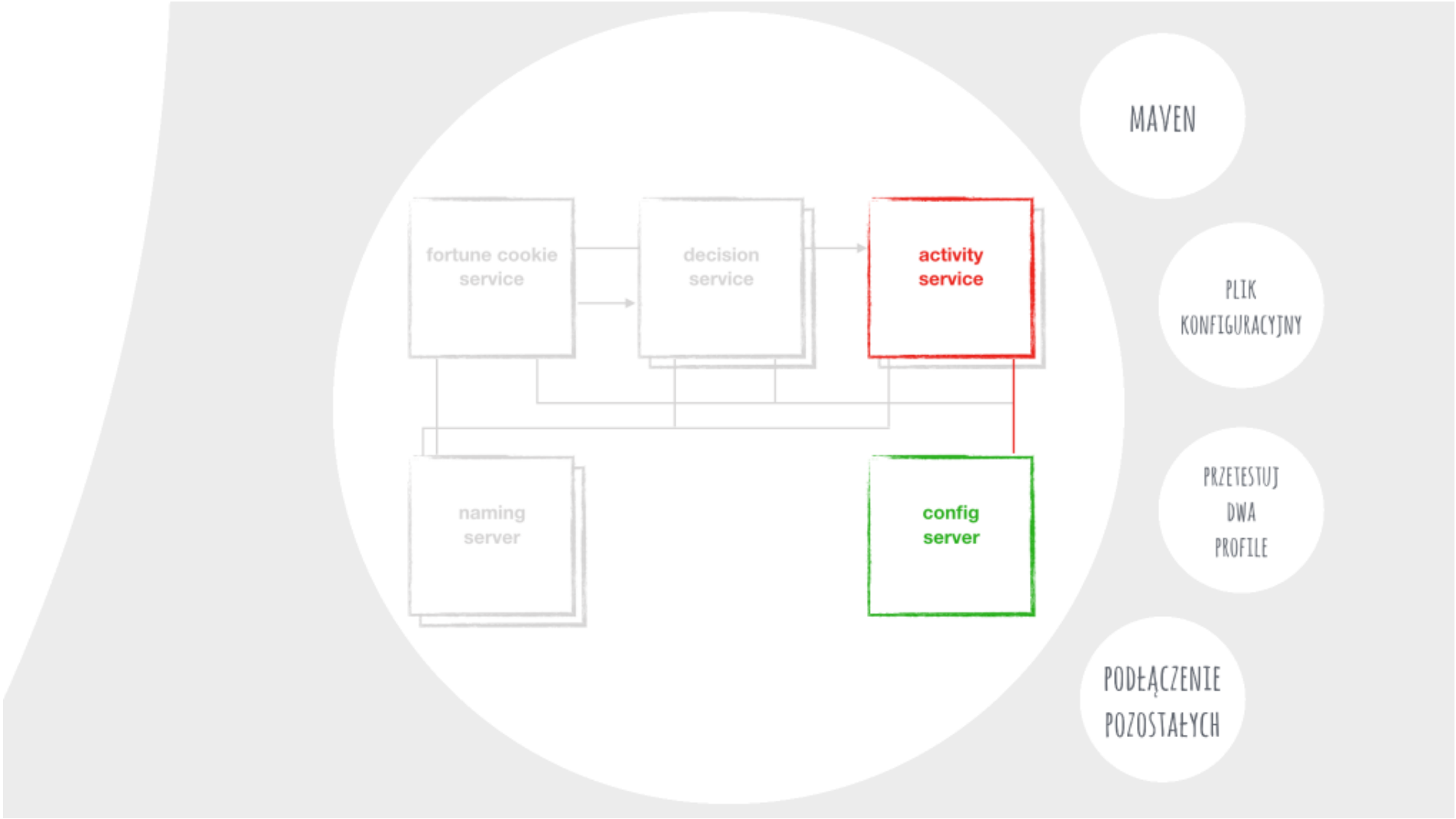
```
eureka:
  client:
    serviceUrl:
      defaultZone: http://eureka-primary:8761/eureka,http://eureka-secondary:8762/eureka
---
spring:
  profiles: lazy
activities: to grab a bear,to eat a cookie,to take a nap
---
spring:
  profiles: crazy
activities: to run a marathon, to go hiking
```


CONFIG SERVER
SZKIELET PROJEKTU
YML & @
TEST

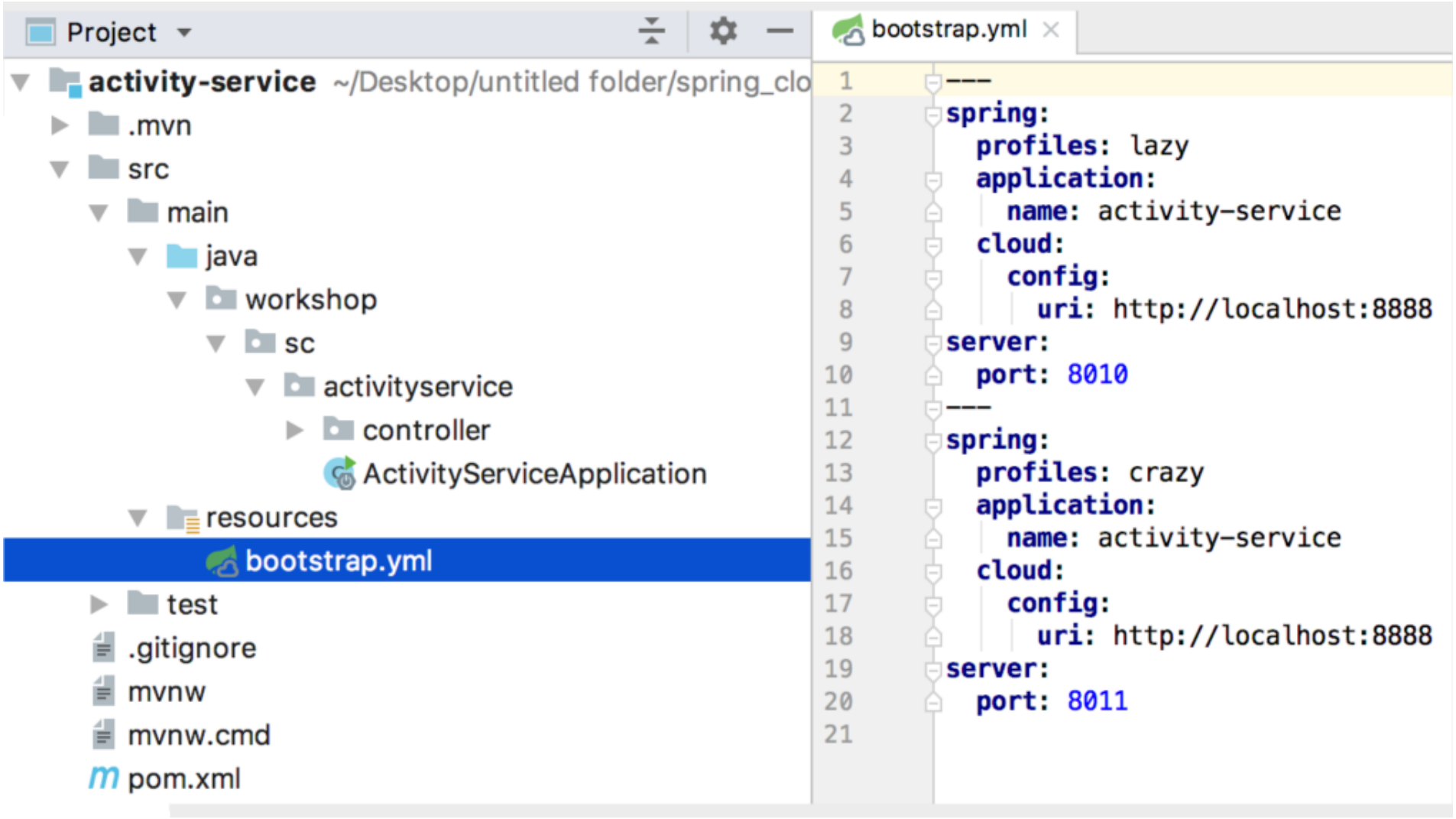

SPRING INITIALIZR bootstrap your application now
Generate a Maven Project with Java and Spring Boot 2.0.5
Project Metadata
Artifact coordinates
Group
workshop.sc
Artifact
config
Dependencies
Add Spring Boot Starters and dependencies to your application
Search for dependencies
Web, Security, JPA, Actuator, Devtools...
Selected Dependencies
Config Server
Generate Project ⌘ + ⏎
Don't know what to look for? Want more options? Switch to the full version.


Project
activity-service ~/Desktop/untitled folder/spring_clo
.mvn
src
.gitignore
mvnw
mvnw.cmd
pom.xml
config-server ~/Desktop/untitled folder/spring_cloud
.mvn
src
main
java
workshop
sc
configserver
ConfigServerApplication
resources
application.yml
test
.gitignore
mvnw
mvnw.cmd
pom.xml
application.yml
spring:
cloud:
config:
server:
git:
uri: file:///${user.home}/localrepo
application:
name: config-server
server:
port: 8888
Document 1/1 › spring: › cloud: › config: › serve
ConfigServerApplication.java
package workshop.sc.configserver;
import ...
@SpringBootApplication
@EnableConfigServer
public class ConfigServerApplication {
public static void main(String[] args)
}


localhost:8888/<serwis>/<profil>


fortune cookie service
decision service
activity service
naming server
config server
MAVEN
PLIK KONFIGURACYJNY
PRZETESTUJ DWA PROFILE
PODŁĄCZENIE POZOSTAŁYCH

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
```

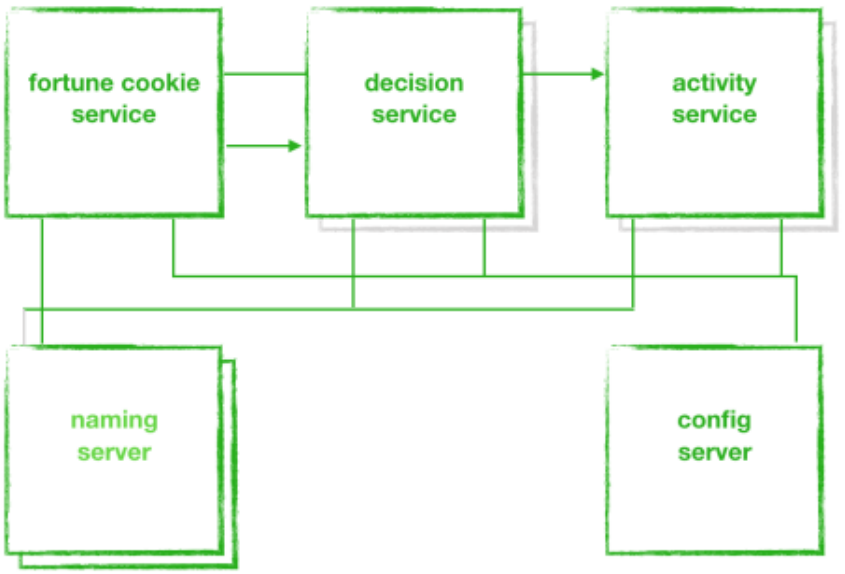

Project
activity-service ~/Desktop/untitled folder/spring_clo
.mvn
src
main
java
workshop
sc
activityservice
controller
ActivityServiceApplication
resources
bootstrap.yml
test
.gitignore
mvnw
mvnw.cmd
pom.xml
bootstrap.yml
1 ---
2 spring:
3 profiles: lazy
4 application:
5 name: activity-service
6 cloud:
7 config:
8 uri: http://localhost:8888
9 server:
10 port: 8010
11 ---
12 spring:
13 profiles: crazy
14 application:
15 name: activity-service
16 cloud:
17 config:
18 uri: http://localhost:8888
19 server:
20 port: 8011
21

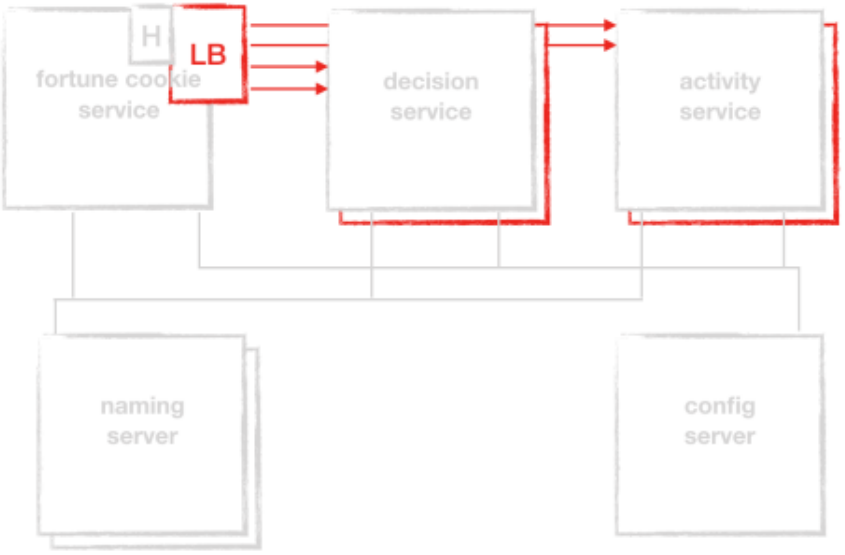

http://localhost:8010
http://localhost:8011

analogicznie jak activity-service

decision-service (profil polite, rough)

fortune-cookie-service (bez profilu)

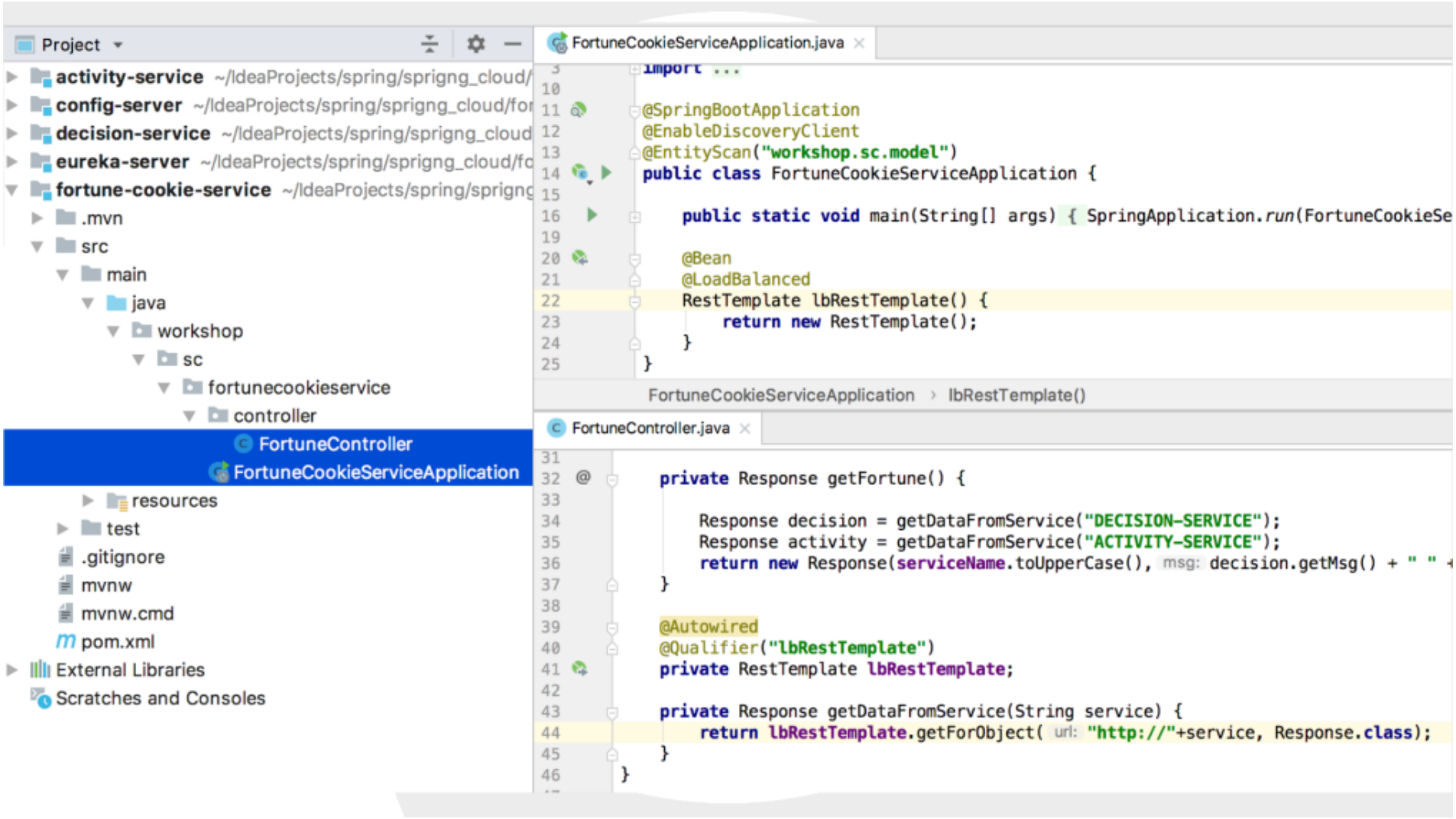

fortune cookie service
decision service
activity service
naming server
config server



REFAKTOR FCS

TEST

PODSUMOWANIE


Project
activity-service ~/IdeaProjects/spring/sprigng_cloud/
config-server ~/IdeaProjects/spring/sprigng_cloud/fo
decision-service ~/IdeaProjects/spring/sprigng_cloud
eureka-server ~/IdeaProjects/spring/sprigng_cloud/fo
fortune-cookie-service ~/IdeaProjects/spring/sprigng
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
FortuneCookieServiceApplication
resources
test
.gitignore
mvnw
mvnw.cmd
pom.xml
External Libraries
Scratches and Consoles
FortuneCookieServiceApplication.java
import ...
@SpringBootApplication
@EnableDiscoveryClient
@EntityScan("workshop.sc.model")
public class FortuneCookieServiceApplication {
public static void main(String[] args) { SpringApplication.run(FortuneCookieSe
@Bean
@LoadBalanced
RestTemplate lbRestTemplate() {
return new RestTemplate();
}
}
FortuneCookieServiceApplication › lbRestTemplate()
FortuneController.java
private Response getFortune() {
Response decision = getDataFromService("DECISION-SERVICE");
Response activity = getDataFromService("ACTIVITY-SERVICE");
return new Response(serviceName.toUpperCase(), msg: decision.getMsg() + " "
}
@Autowired
@Qualifier("lbRestTemplate")
private RestTemplate lbRestTemplate;
private Response getDataFromService(String service) {
return lbRestTemplate.getForObject(url: "http://"+service, Response.class);
}
}

URUCHOMIENIE PODWÓJNYCH INSTANCJI

DECISION-SERVICE

PROFILE POLITE I ROUGH

ACTIVITY-SERVICE

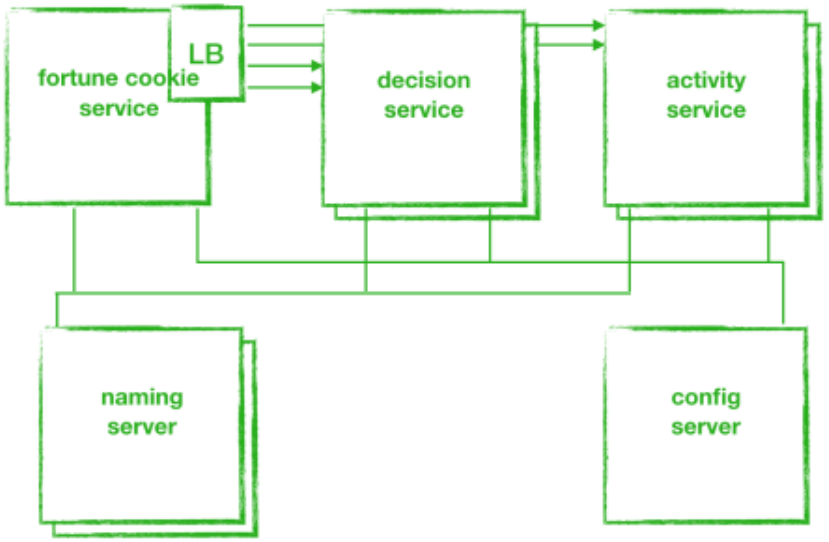
PROFILE LAZY I CRAZY

W LOGACH FORTUNE-COOKIE-SERVICE

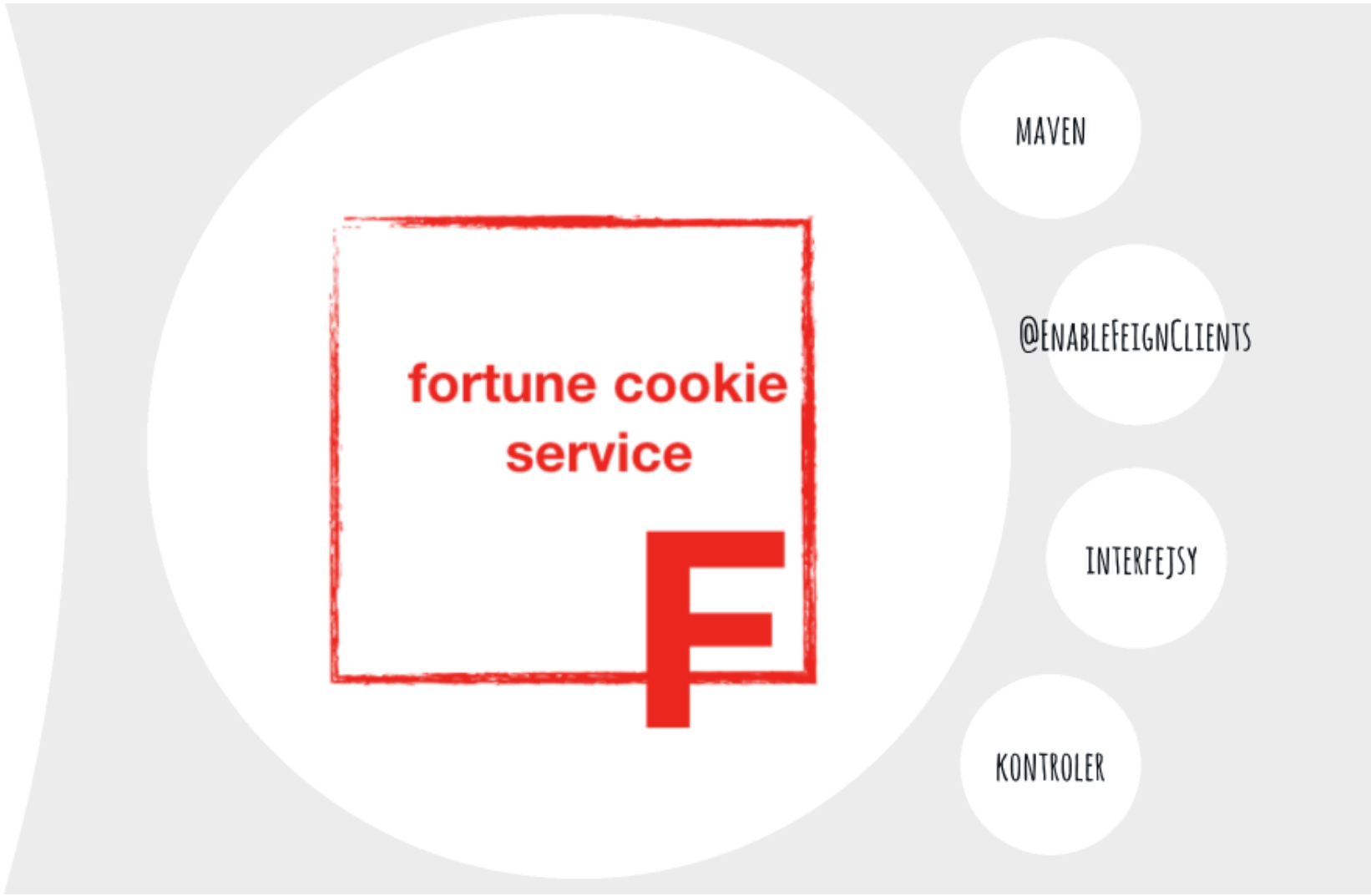
POWINNO BYĆ WIDAĆ INSTANCJE W CURRENT LIST OF SERVERS

REQUESTY DO FORTUNE-COOKIE-SERVICE

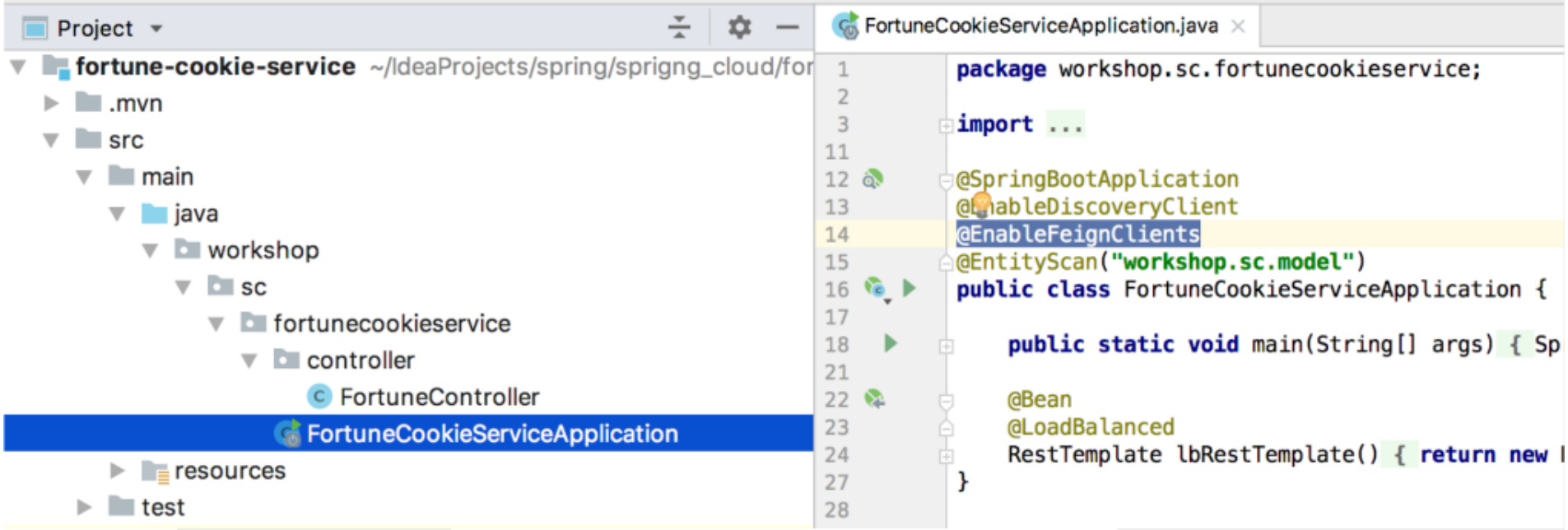
POWINNY ZWRACAĆ MIX 4 INSTANCJI (W CONFIGU RÓŻNE DANE DLA KAŻDEJ)

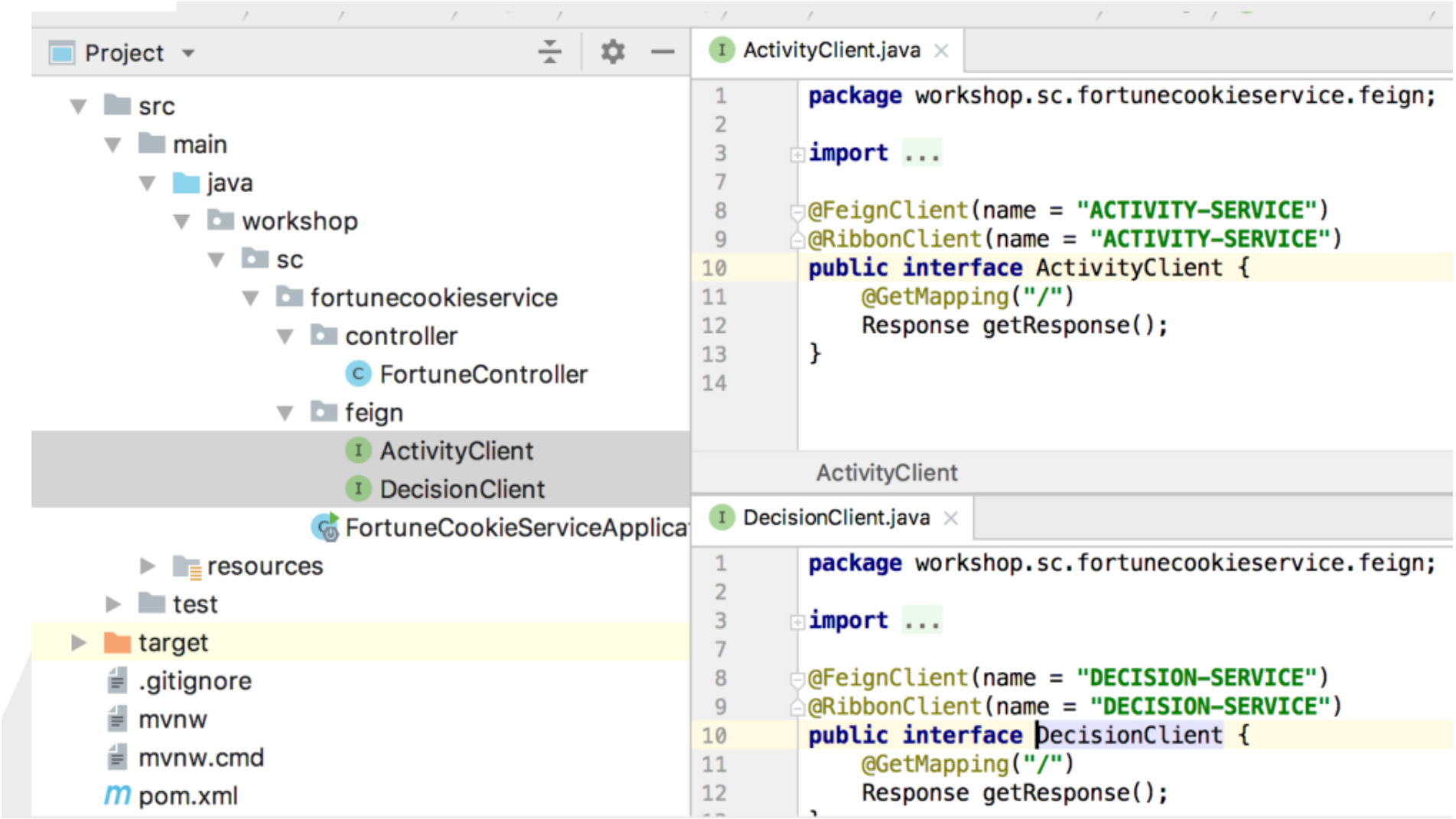

fortune cookie service
LB
decision service
activity service
naming server
config server

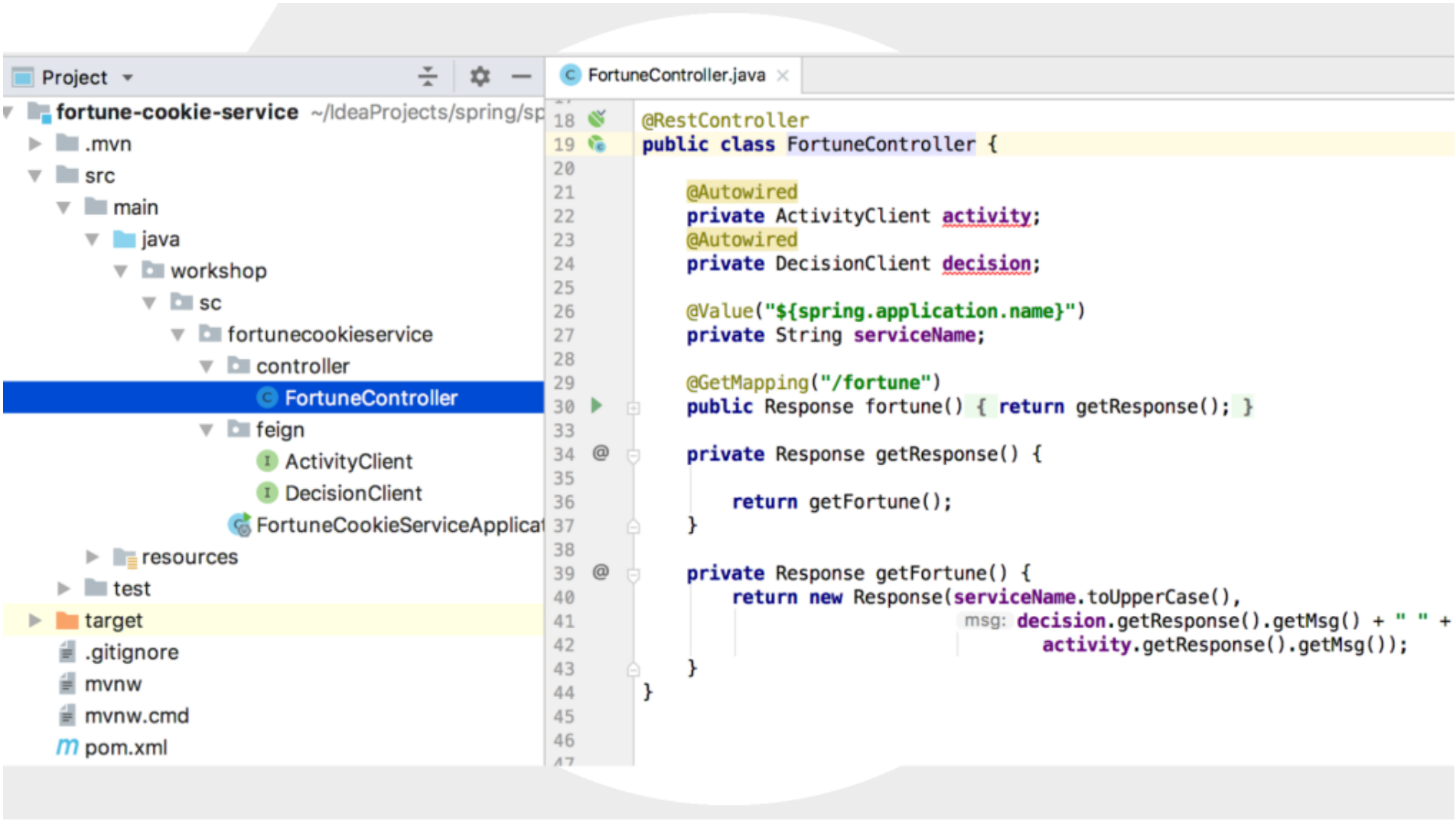

fortune cookie service
F
decision service
activity service
naming server
config server
REFAKTOR
FCS

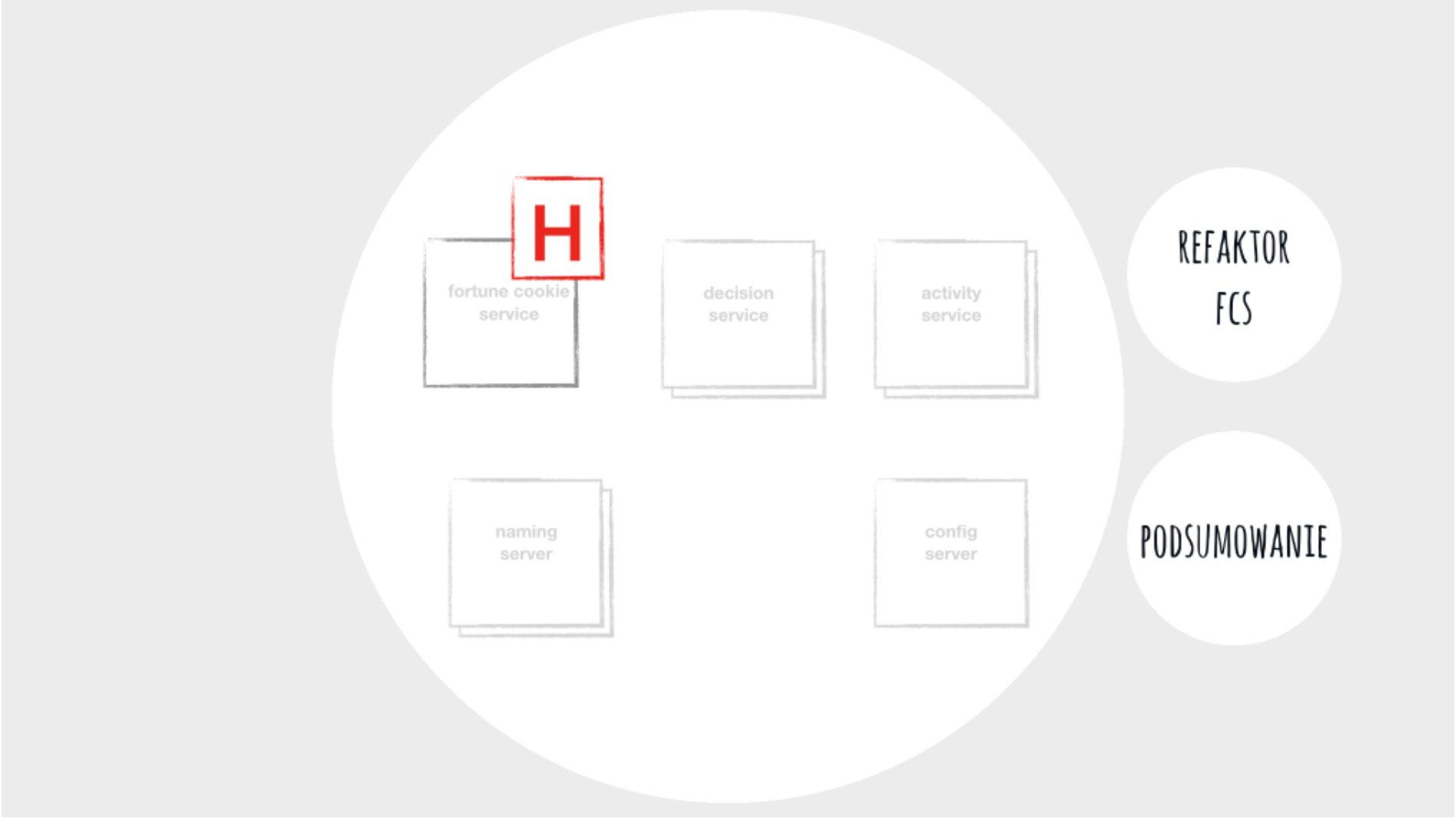

fortune cookie service
MAVEN
@EnableFeignClients
INTERFEJSY
KONTROLER

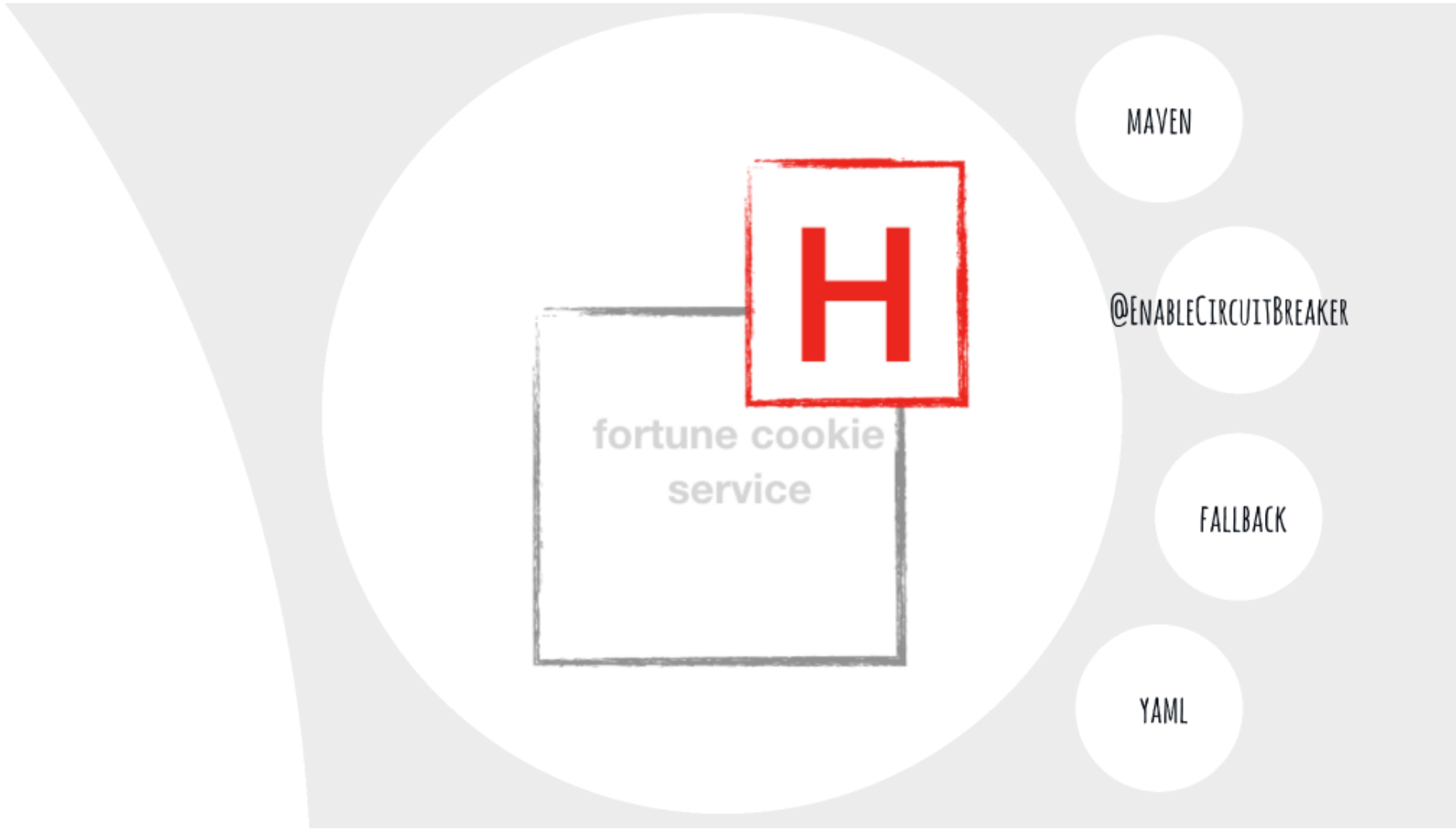
```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-openfeign</artifactId>
</dependency>
```


Project
fortune-cookie-service ~/IdeaProjects/spring/sprigng_cloud/for
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
FortuneCookieServiceApplication
resources
test
FortuneCookieServiceApplication.java
1 package workshop.sc.fortunecookieservice;
2
3 import ...
11
12 @SpringBootApplication
13 @EnableDiscoveryClient
14 @EnableFeignClients
15 @EntityScan("workshop.sc.model")
16 public class FortuneCookieServiceApplication {
17
18 public static void main(String[] args) { Sp
21
22 @Bean
23 @LoadBalanced
24 RestTemplate lbRestTemplate() { return new
27 }
28

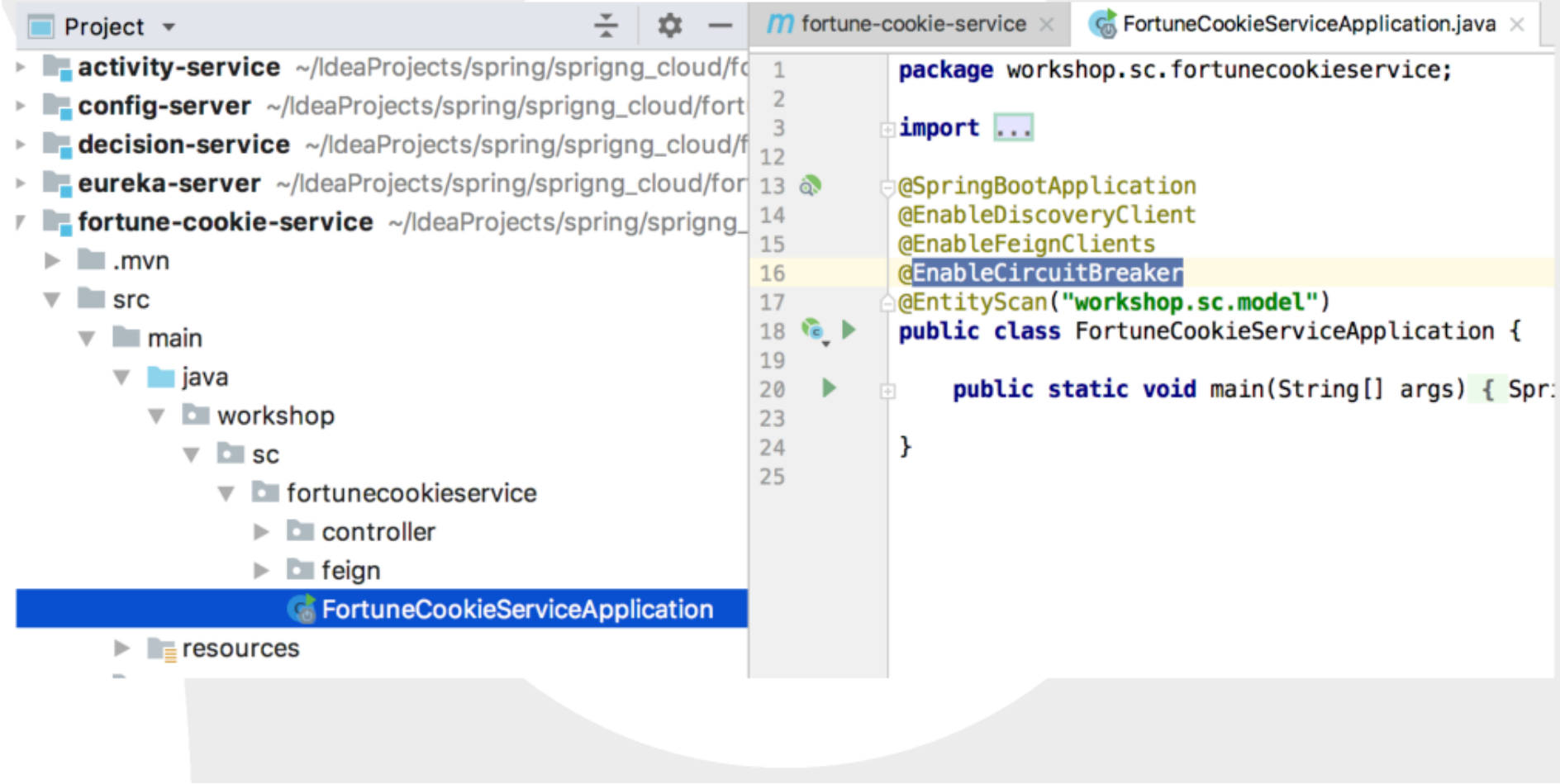

Project
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
feign
ActivityClient
DecisionClient
FortuneCookieServiceApplica
resources
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
ActivityClient.java
1 package workshop.sc.fortunecookieservice.feign;
2
3 import ...
7
8 @FeignClient(name = "ACTIVITY-SERVICE")
9 @RibbonClient(name = "ACTIVITY-SERVICE")
10 public interface ActivityClient {
11 @GetMapping("/")
12 Response getResponse();
13 }
14
ActivityClient
DecisionClient.java
1 package workshop.sc.fortunecookieservice.feign;
2
3 import ...
7
8 @FeignClient(name = "DECISION-SERVICE")
9 @RibbonClient(name = "DECISION-SERVICE")
10 public interface DecisionClient {
11 @GetMapping("/")
12 Response getResponse();

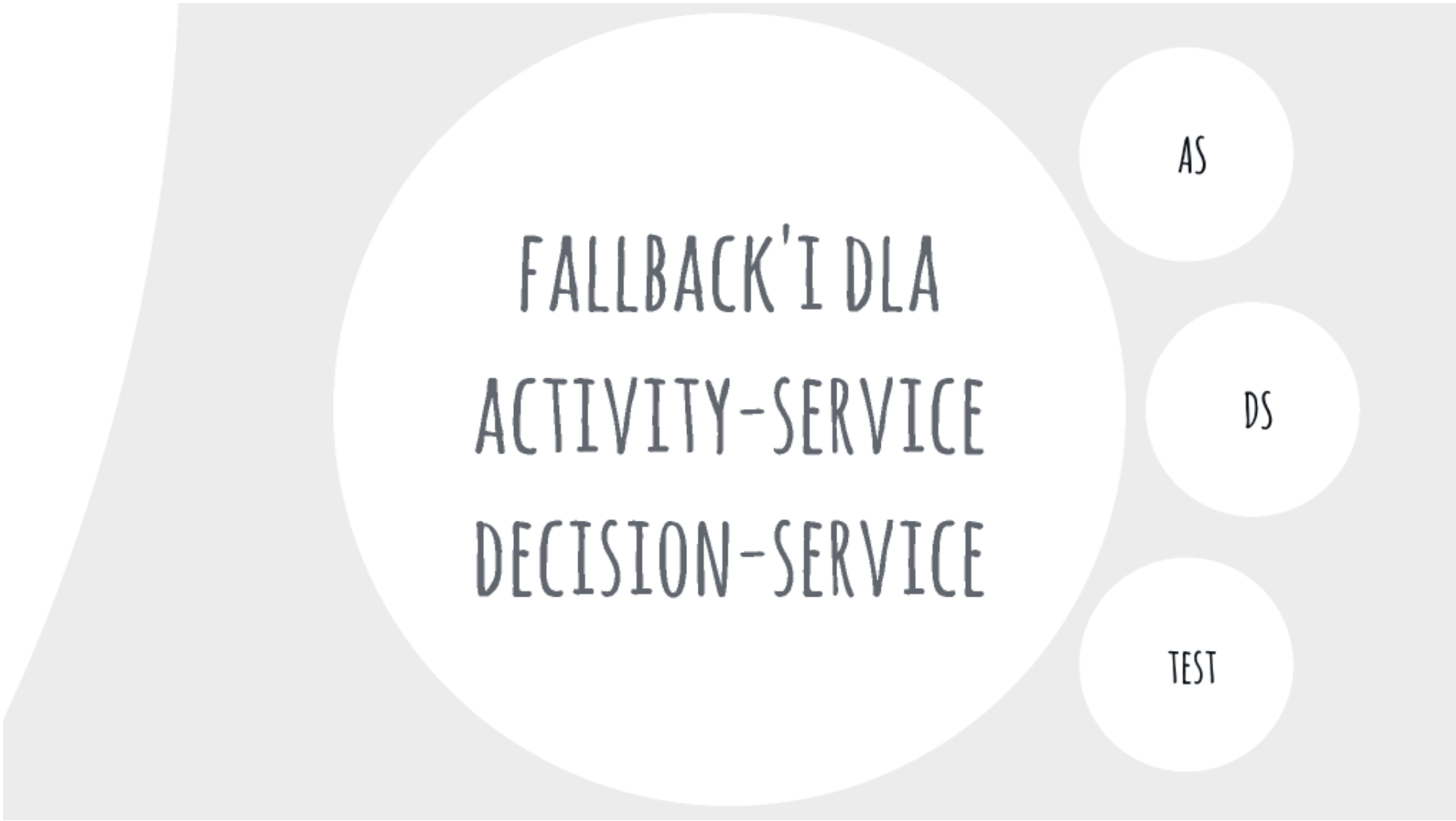

Project
fortune-cookie-service ~/IdeaProjects/spring/sp
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
feign
ActivityClient
DecisionClient
FortuneCookieServiceApplicat
resources
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
FortuneController.java
@RestController
public class FortuneController {
@Autowired
private ActivityClient activity;
@Autowired
private DecisionClient decision;
@Value("${spring.application.name}")
private String serviceName;
@GetMapping("/fortune")
public Response fortune() { return getResponse(); }
private Response getResponse() {
return getFortune();
}
private Response getFortune() {
return new Response(serviceName.toUpperCase(),
msg: decision.getResponse().getMsg() + " " +
activity.getResponse().getMsg());
}
}

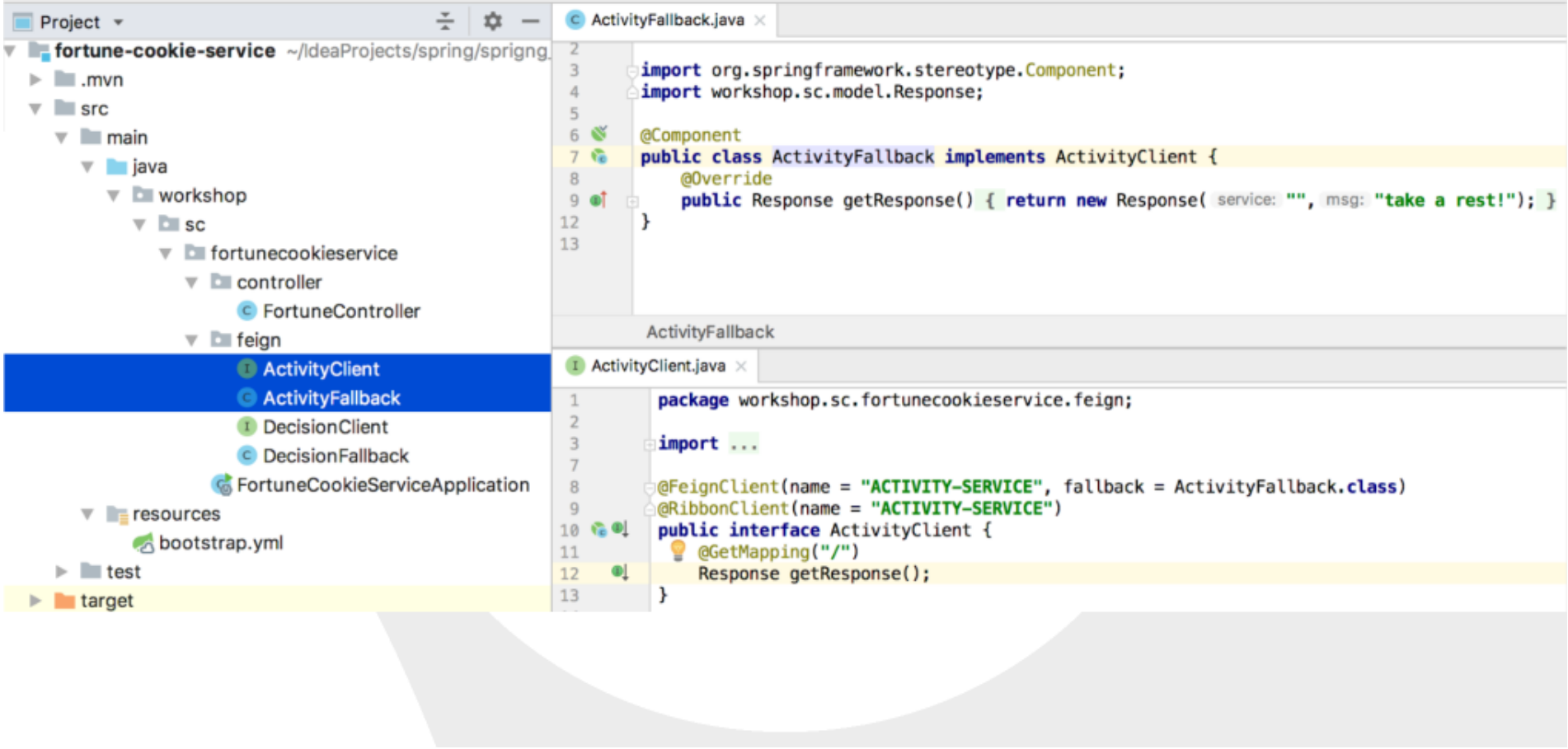

H
fortune cookie service
decision service
activity service
naming server
config server
REFAKTOR FCS
PODSUMOWANIE


MAVEN
@EnableCircuitBreaker
FALLBACK
YAML
fortune cookie
service

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-hystrix</artifactId>
    <version>1.4.4.RELEASE</version>
</dependency>
```


Project
activity-service ~/IdeaProjects/spring/sprignng_cloud/fc
config-server ~/IdeaProjects/spring/sprignng_cloud/fort
decision-service ~/IdeaProjects/spring/sprignng_cloud/f
eureka-server ~/IdeaProjects/spring/sprignng_cloud/for
fortune-cookie-service ~/IdeaProjects/spring/sprignng_
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
feign
FortuneCookieServiceApplication
resources
fortune-cookie-service
FortuneCookieServiceApplication.java
1 package workshop.sc.fortunecookieservice;
2
3 import ...
12
13 @SpringBootApplication
14 @EnableDiscoveryClient
15 @EnableFeignClients
16 @EnableCircuitBreaker
17 @EntityScan("workshop.sc.model")
18 public class FortuneCookieServiceApplication {
19
20 public static void main(String[] args) { Spr
23
24 }
25

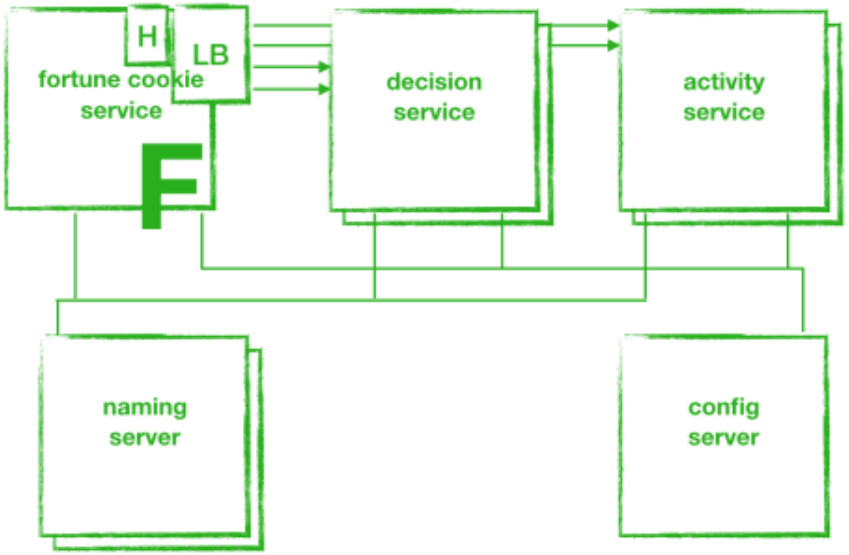

FALLBACK'I DLA
ACTIVITY-SERVICE
DECISION-SERVICE
AS
DS
TEST

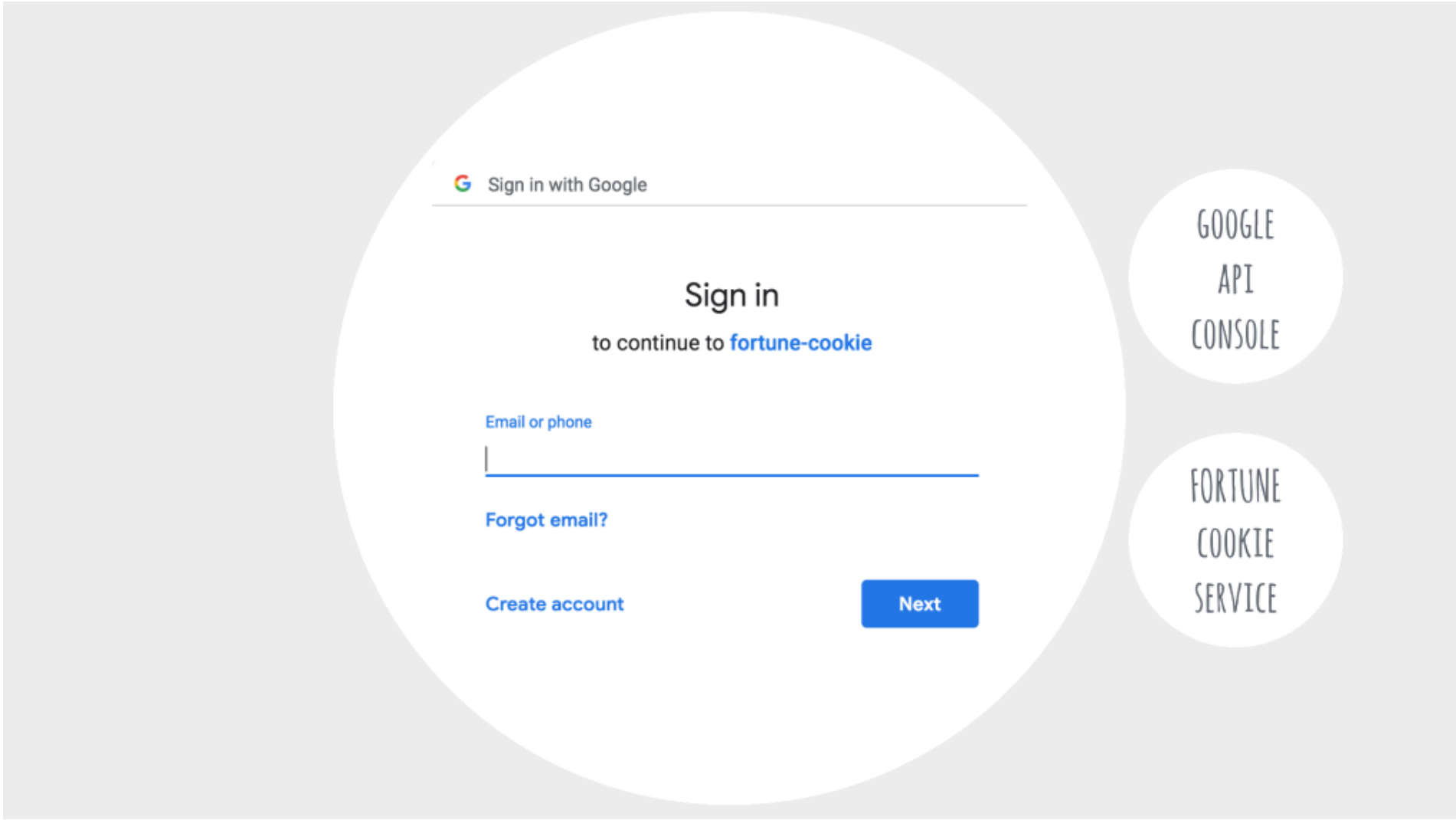

Project
fortune-cookie-service ~/IdeaProjects/spring/sprignq
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
FortuneController
feign
ActivityClient
ActivityFallback
DecisionClient
DecisionFallback
FortuneCookieServiceApplication
resources
bootstrap.yml
test
target
ActivityFallback.java
import org.springframework.stereotype.Component;
import workshop.sc.model.Response;
@Component
public class ActivityFallback implements ActivityClient {
@Override
public Response getResponse() { return new Response(service: "", msg: "take a rest!"); }
}
ActivityFallback
ActivityClient.java
package workshop.sc.fortunecookieservice.feign;
import ...
@FeignClient(name = "ACTIVITY-SERVICE", fallback = ActivityFallback.class)
@RibbonClient(name = "ACTIVITY-SERVICE")
public interface ActivityClient {
@GetMapping("/")
Response getResponse();
}

ANALOGICZNIE DO ACTIVITY-SERVICE

PRZETESTUJ FCS PO WYŁACZENIU AS / DS

```
feign:
    hystrix:
        enabled: true
```


H
LB
fortune cookie service
F
decision service
activity service
naming server
config server


Sign in with Google
Sign in
to continue to fortune-cookie
Email or phone
Forgot email?
Create account
Next
GOOGLE API CONSOLE
FORTUNE COOKIE SERVICE

https://console.developers.google.com/

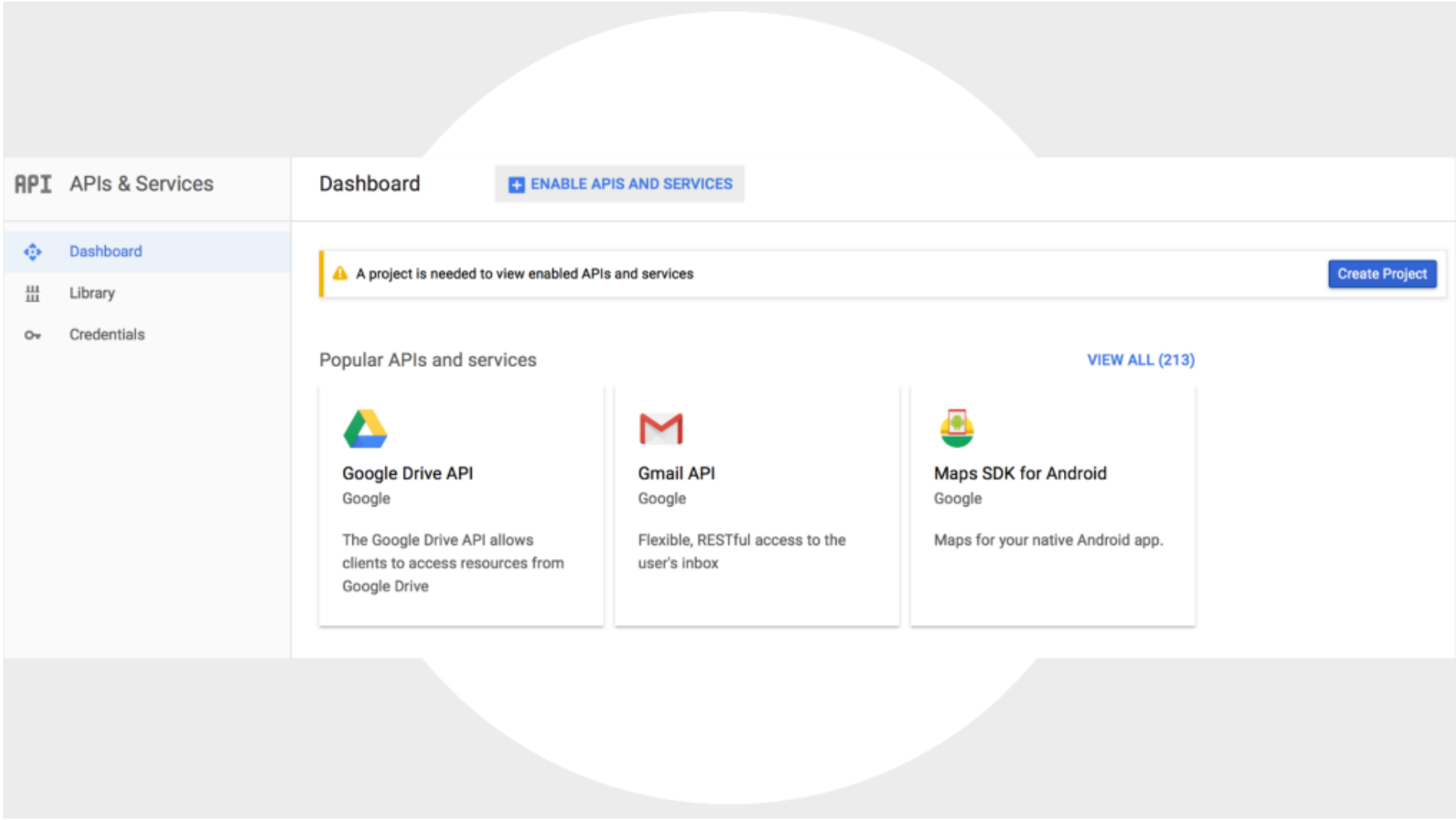
I, II, III UTWORZENIE PROJEKTU

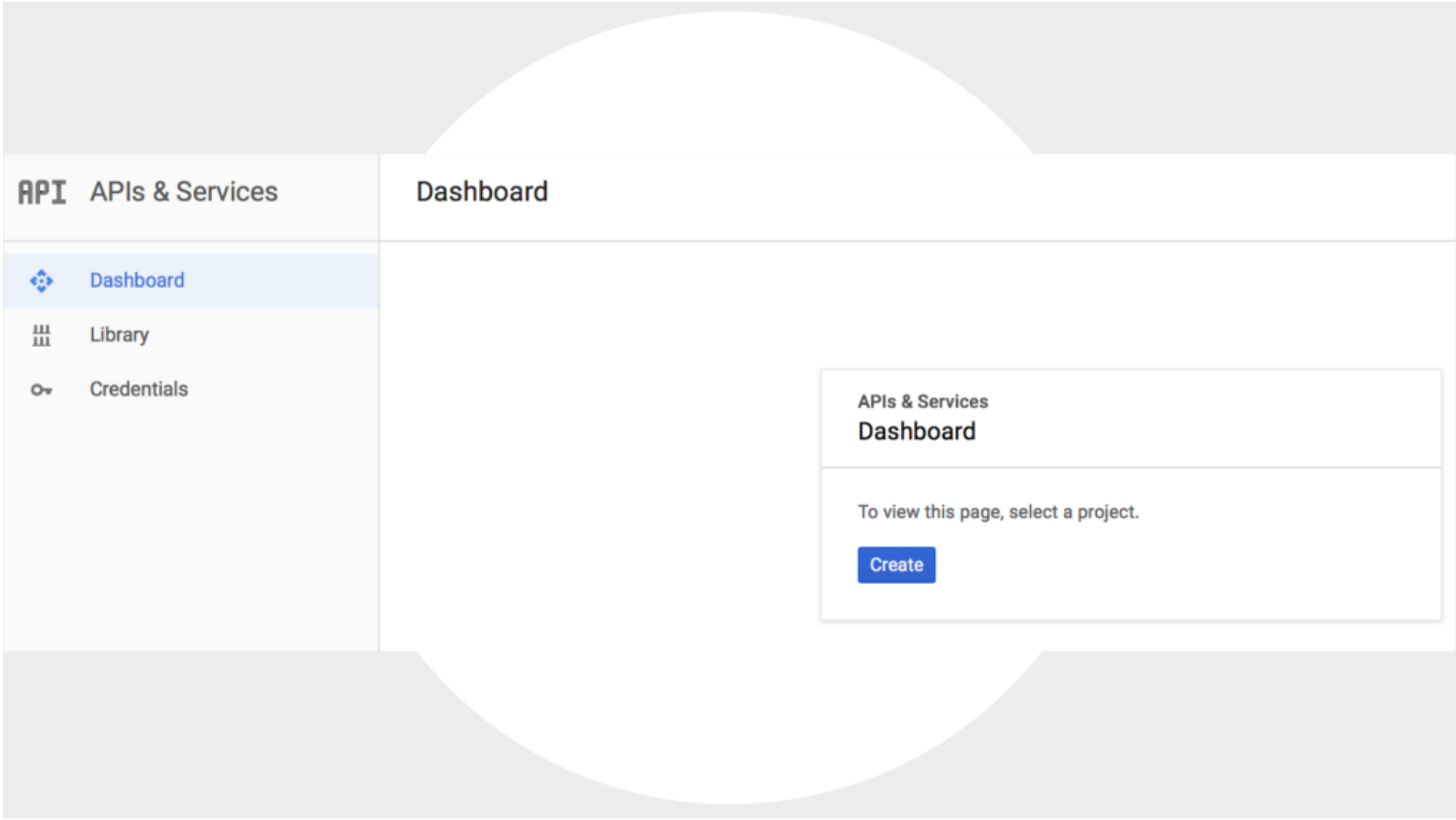
IV-VII USTAWIENIA CREDENTIALS

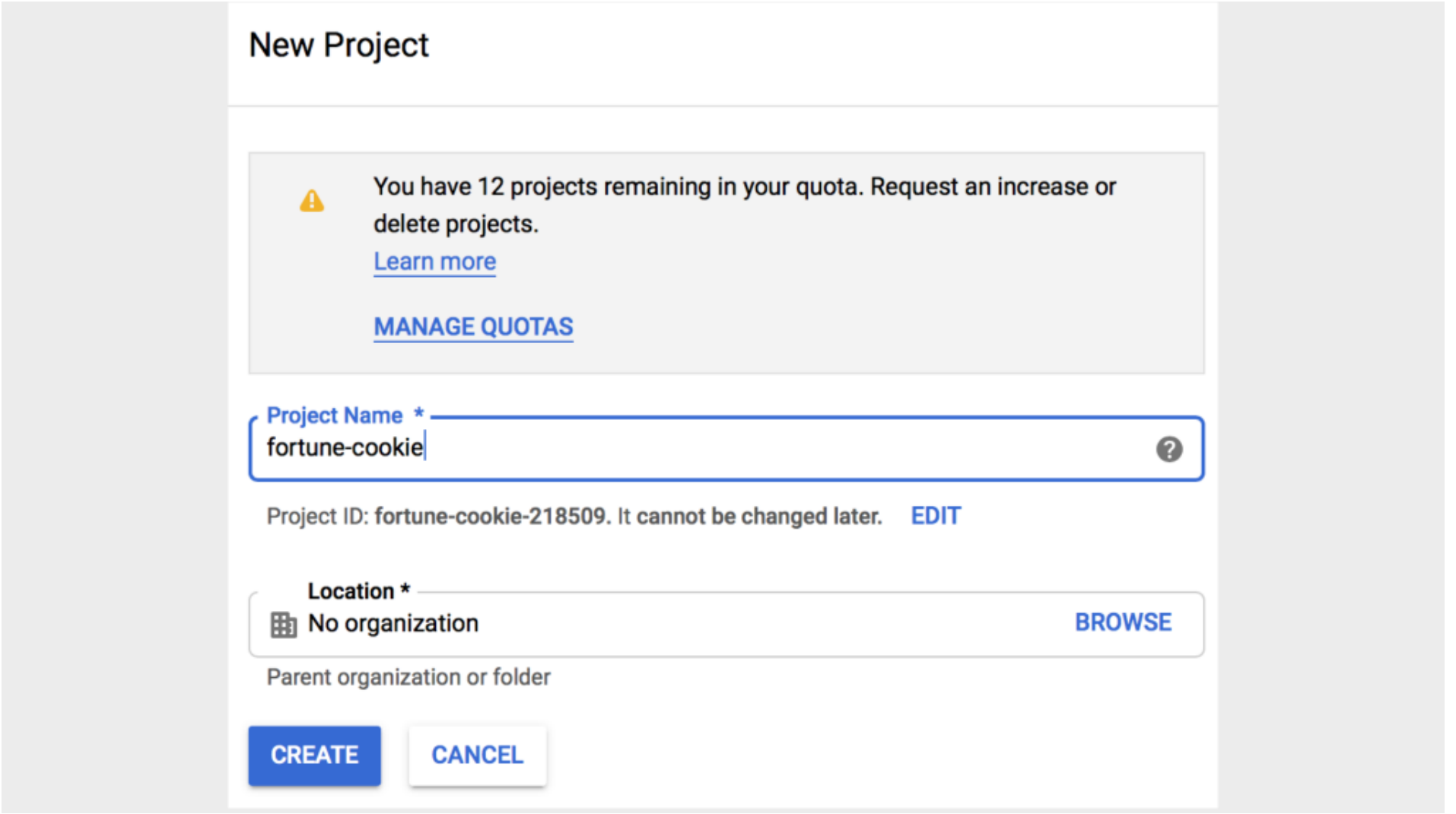
VIII TYP APLIKACJI I URL'E

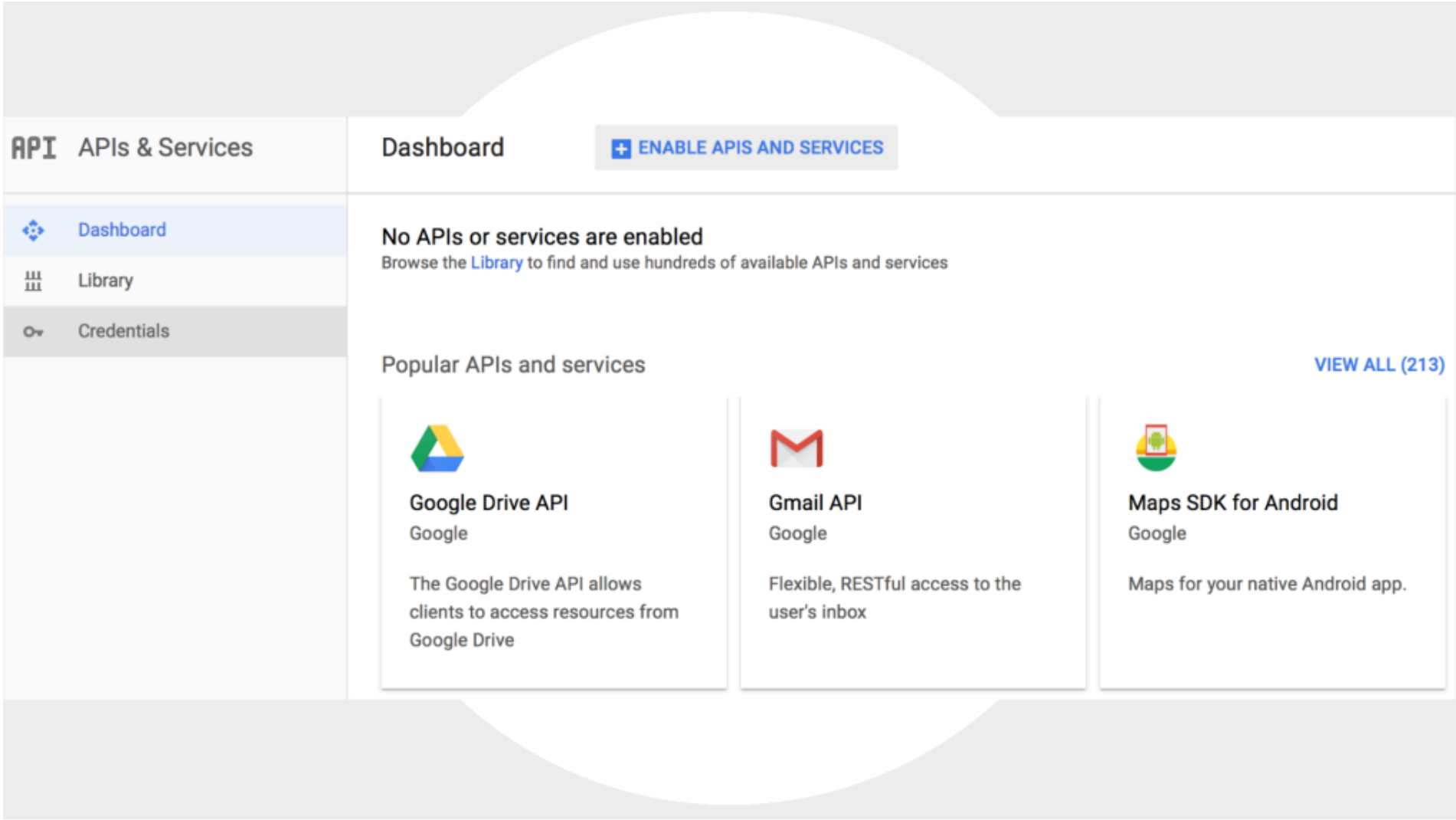
IX CLIENT ID / SECRET

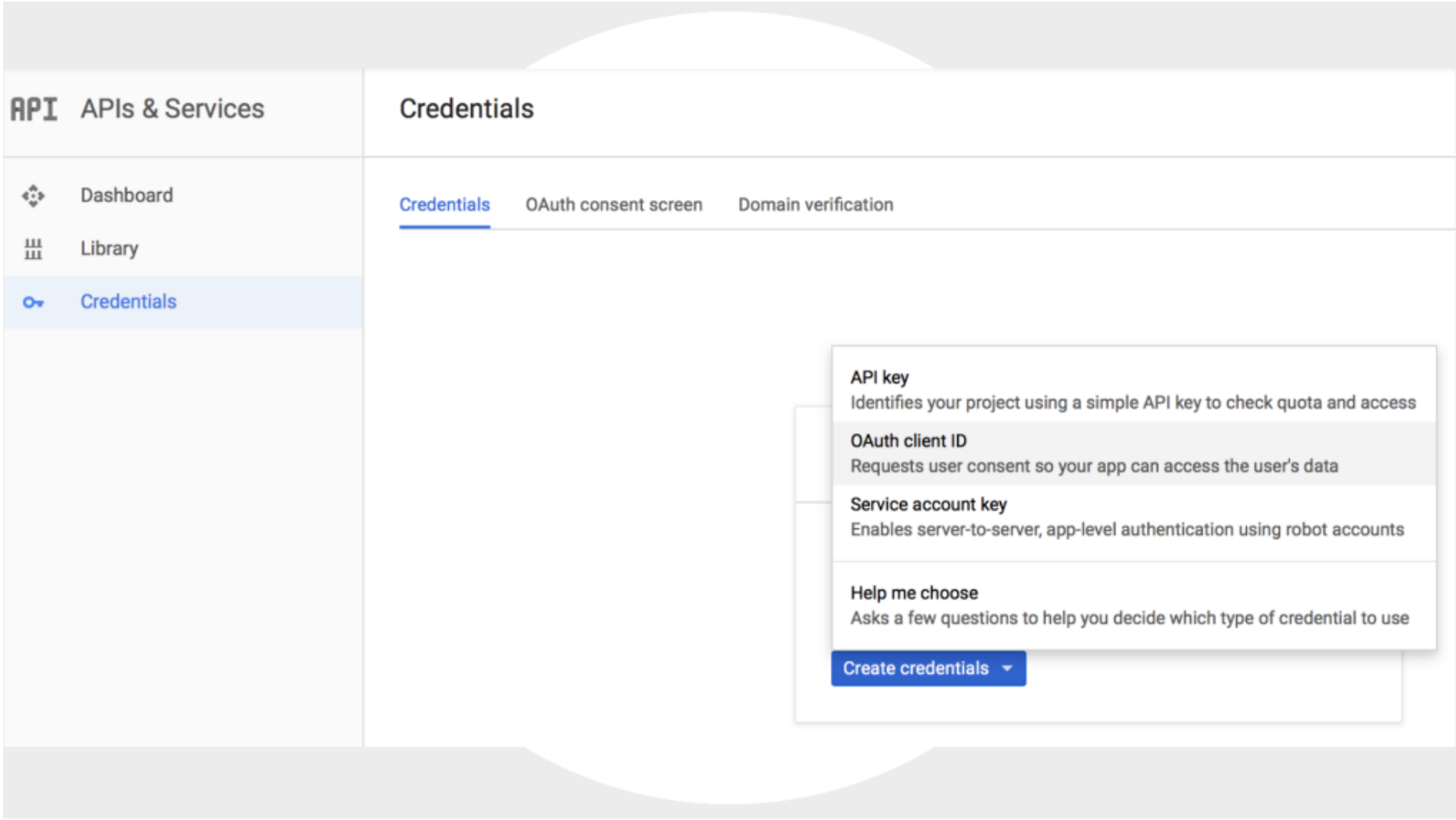
I
II
III
IV
V
VI
VII
VIII
IX

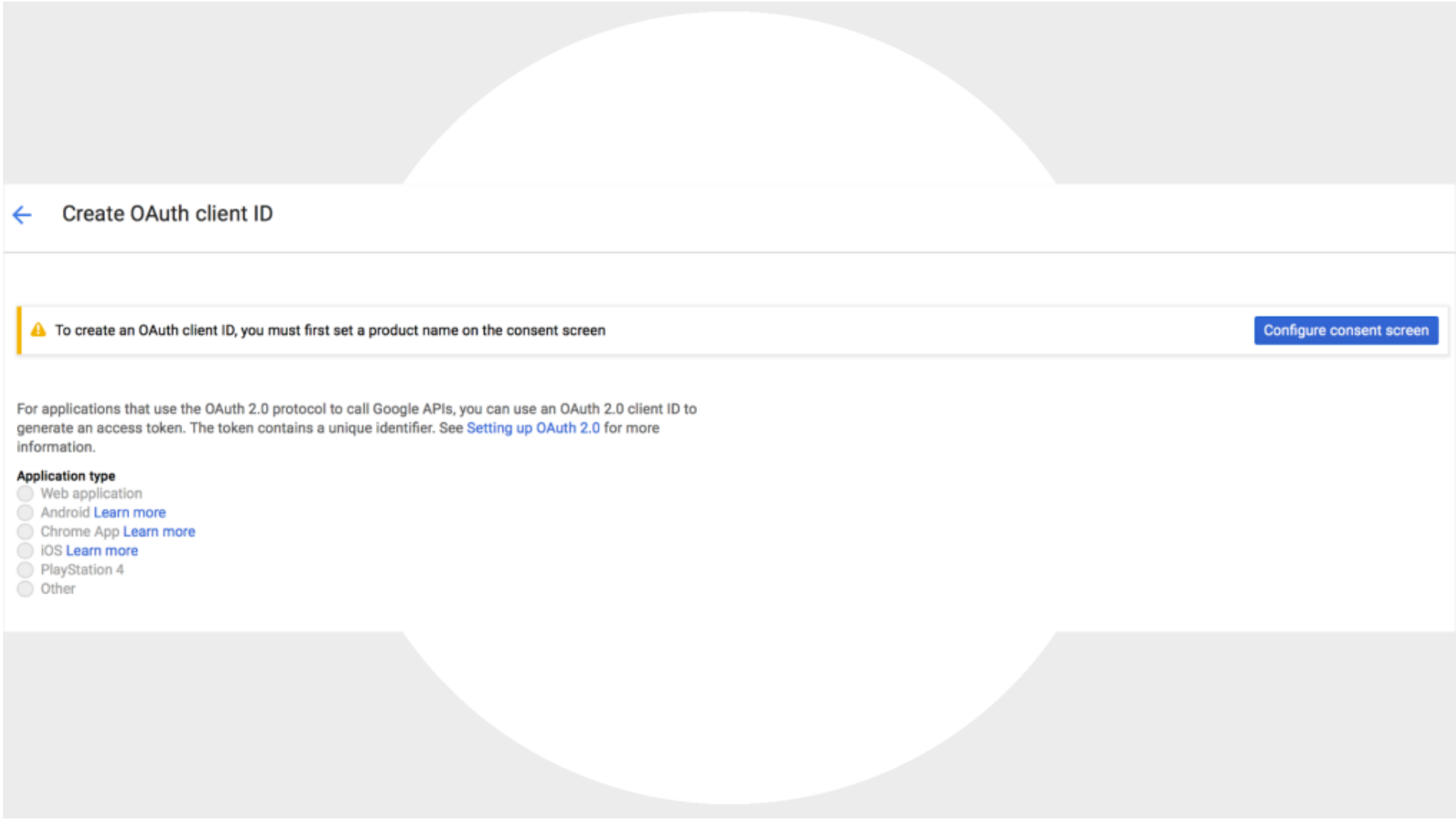

API APIs & Services
Dashboard
ENABLE APIS AND SERVICES
Dashboard
Library
Credentials
A project is needed to view enabled APIs and services
Create Project
Popular APIs and services
VIEW ALL (213)
Google Drive API
Google
The Google Drive API allows clients to access resources from Google Drive
Gmail API
Google
Flexible, RESTful access to the user's inbox
Maps SDK for Android
Google
Maps for your native Android app.

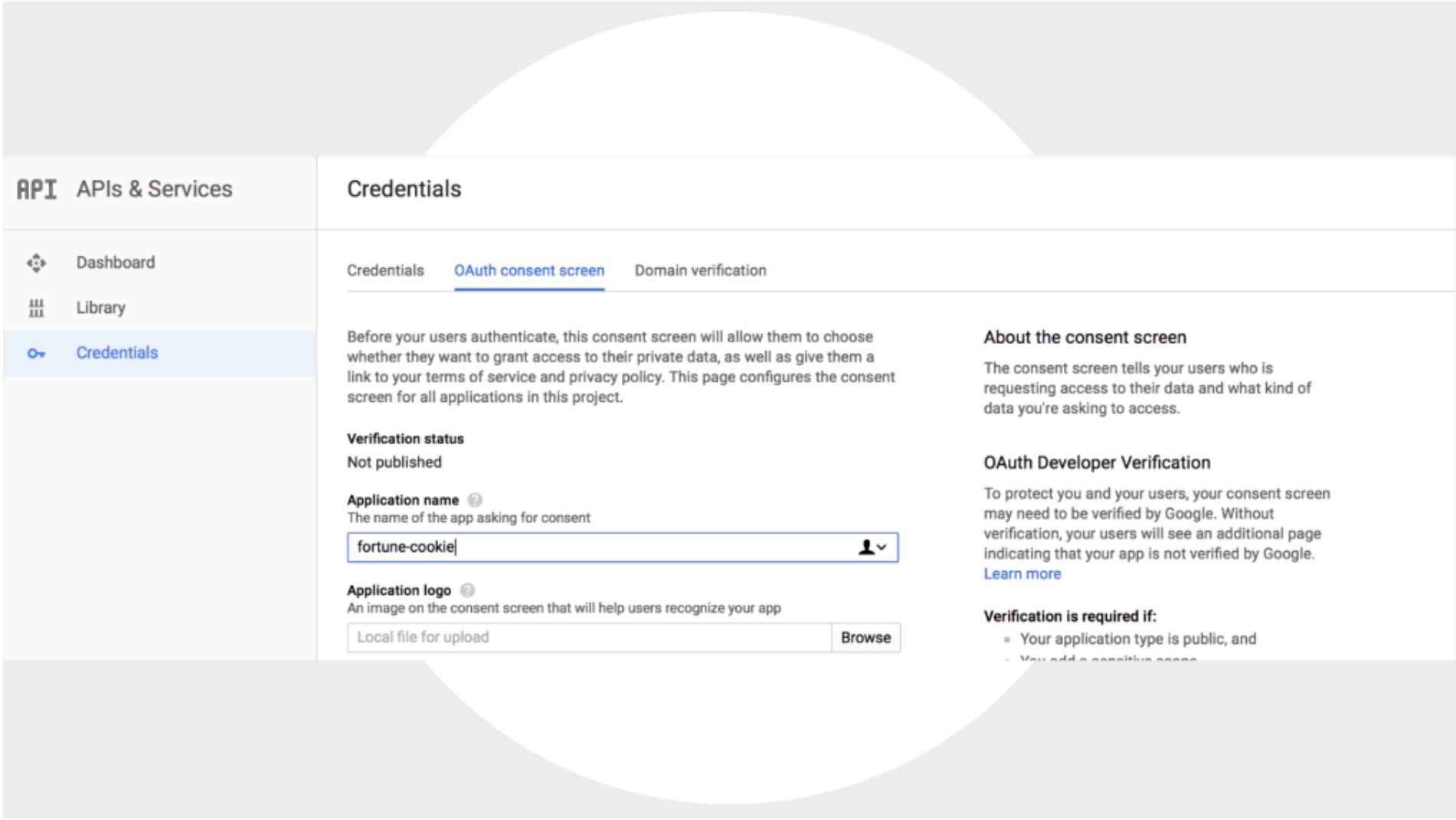

APIs & Services
Dashboard
Dashboard
Library
Credentials
APIs & Services
Dashboard
To view this page, select a project.
Create

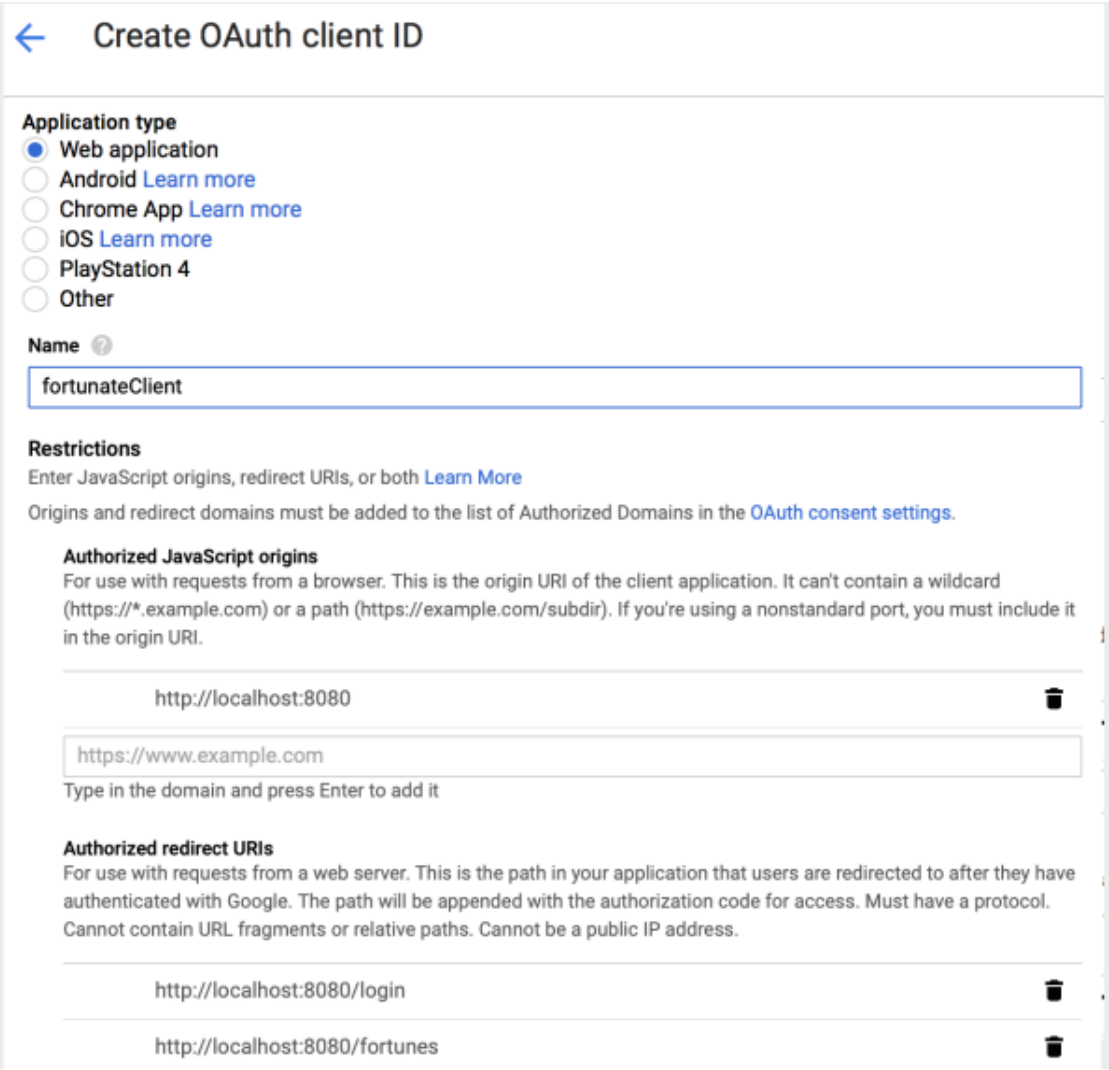

New Project
You have 12 projects remaining in your quota. Request an increase or delete projects.
Learn more
MANAGE QUOTAS
Project Name *
fortune-cookie
Project ID: fortune-cookie-218509. It cannot be changed later.
EDIT
Location *
No organization
BROWSE
Parent organization or folder
CREATE
CANCEL

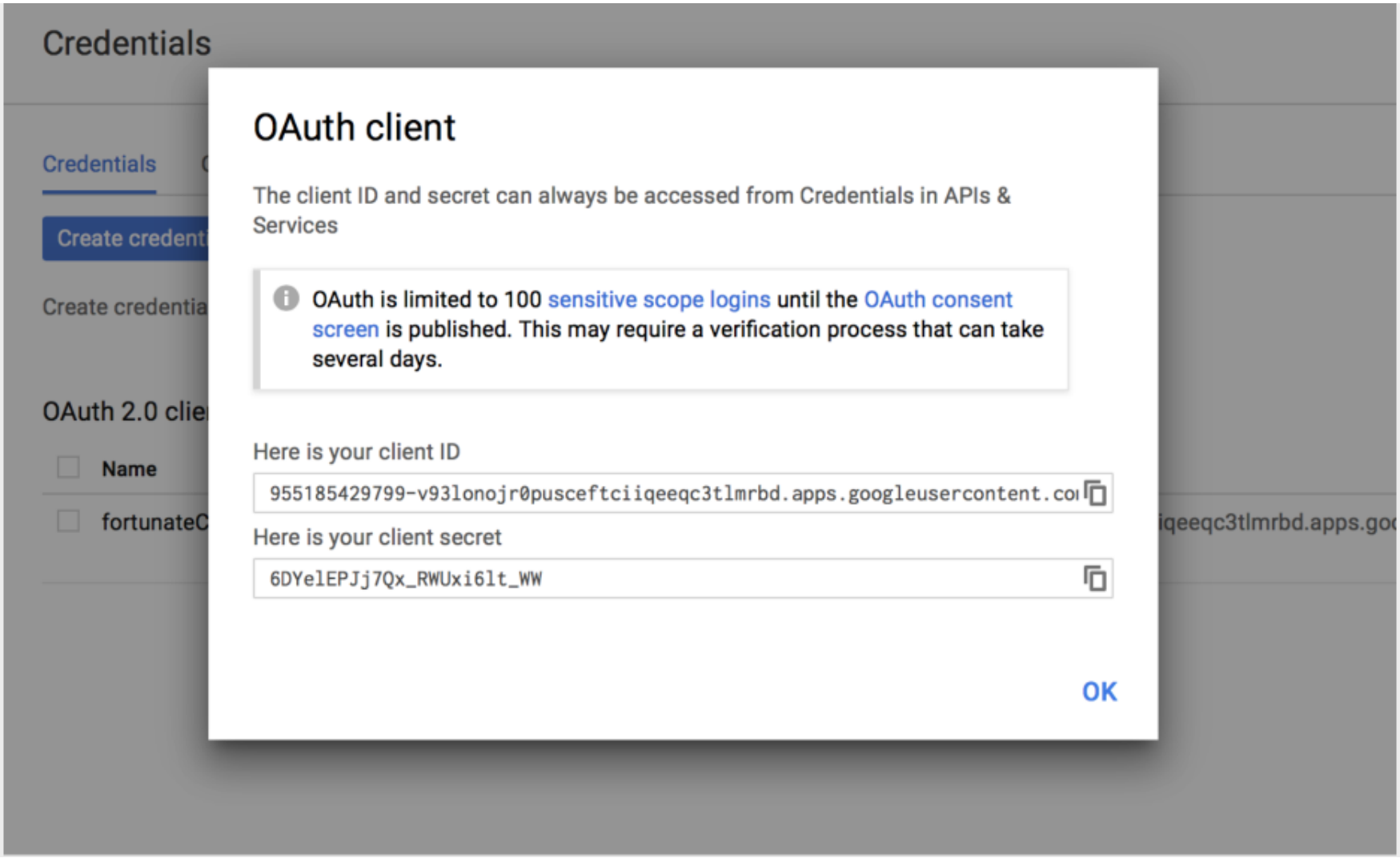

API APIs & Services
Dashboard
ENABLE APIS AND SERVICES
Dashboard
Library
Credentials
No APIs or services are enabled
Browse the Library to find and use hundreds of available APIs and services
Popular APIs and services
VIEW ALL (213)
Google Drive API
Google
The Google Drive API allows clients to access resources from Google Drive
Gmail API
Google
Flexible, RESTful access to the user's inbox
Maps SDK for Android
Google
Maps for your native Android app.


API APIs & Services
Credentials
Dashboard
Library
Credentials
Credentials
OAuth consent screen
Domain verification
API key
Identifies your project using a simple API key to check quota and access
OAuth client ID
Requests user consent so your app can access the user's data
Service account key
Enables server-to-server, app-level authentication using robot accounts
Help me choose
Asks a few questions to help you decide which type of credential to use
Create credentials


Create OAuth client ID
To create an OAuth client ID, you must first set a product name on the consent screen
Configure consent screen
For applications that use the OAuth 2.0 protocol to call Google APIs, you can use an OAuth 2.0 client ID to generate an access token. The token contains a unique identifier. See Setting up OAuth 2.0 for more information.
Application type
Web application
Android Learn more
Chrome App Learn more
iOS Learn more
PlayStation 4
Other


API APIs & Services
Dashboard
Library
Credentials
Credentials
Credentials
OAuth consent screen
Domain verification
Before your users authenticate, this consent screen will allow them to choose whether they want to grant access to their private data, as well as give them a link to your terms of service and privacy policy. This page configures the consent screen for all applications in this project.
Verification status
Not published
Application name
The name of the app asking for consent
fortune-cookie
Application logo
An image on the consent screen that will help users recognize your app
Local file for upload
Browse
About the consent screen
The consent screen tells your users who is requesting access to their data and what kind of data you're asking to access.
OAuth Developer Verification
To protect you and your users, your consent screen may need to be verified by Google. Without verification, your users will see an additional page indicating that your app is not verified by Google.
Learn more
Verification is required if:
Your application type is public, and


Create OAuth client ID
Application type
Web application
Android Learn more
Chrome App Learn more
iOS Learn more
PlayStation 4
Other
Name
fortunateClient
Restrictions
Enter JavaScript origins, redirect URIs, or both Learn More
Origins and redirect domains must be added to the list of Authorized Domains in the OAuth consent settings.
Authorized JavaScript origins
For use with requests from a browser. This is the origin URI of the client application. It can't contain a wildcard (https://*.example.com) or a path (https://example.com/subdir). If you're using a nonstandard port, you must include it in the origin URI.
http://localhost:8080
https://www.example.com
Type in the domain and press Enter to add it
Authorized redirect URIs
For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorization code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.
http://localhost:8080/login
http://localhost:8080/fortunes


Credentials
Credentials
Create credenti
Create credentia
OAuth 2.0 clie
Name
fortunateC
OAuth client
The client ID and secret can always be accessed from Credentials in APIs & Services
OAuth is limited to 100 sensitive scope logins until the OAuth consent screen is published. This may require a verification process that can take several days.
Here is your client ID
955185429799-v93lonojr0pusceftciiqeeqc3tlmrbd.apps.googleusercontent.co
Here is your client secret
6DYelEPJj7Qx_RWUxi6lt_WW
OK
iqeeqc3tlmrbd.apps.goo

I MAVEN

II APPLICATION.YML

III AKTYWACJA SSO - @

IV RESTART I TEST

I

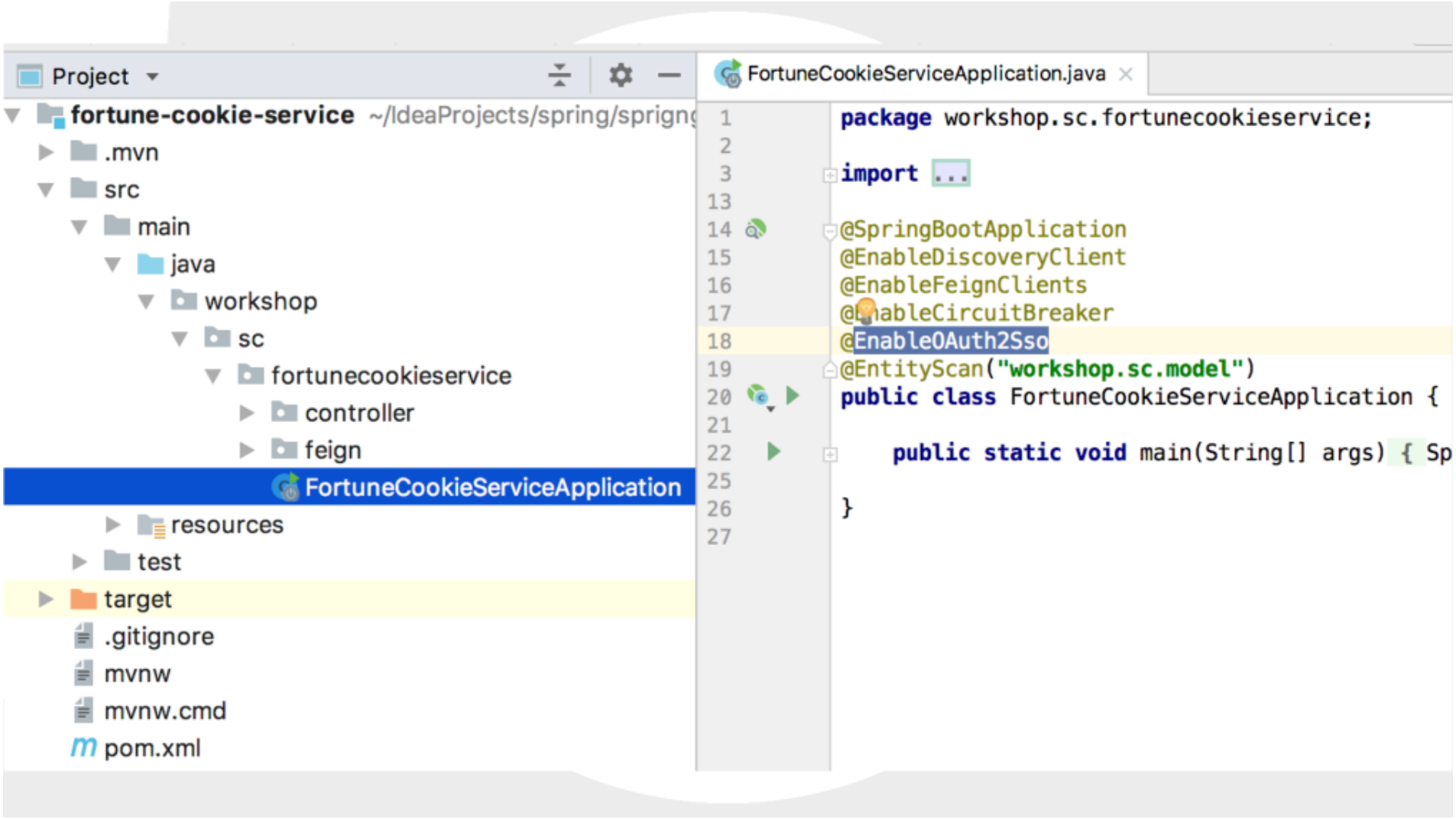
II

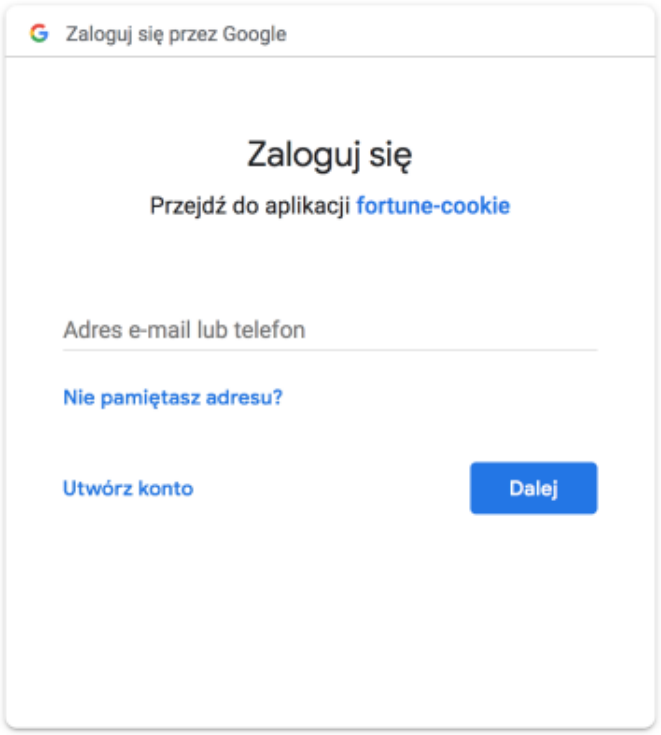
III

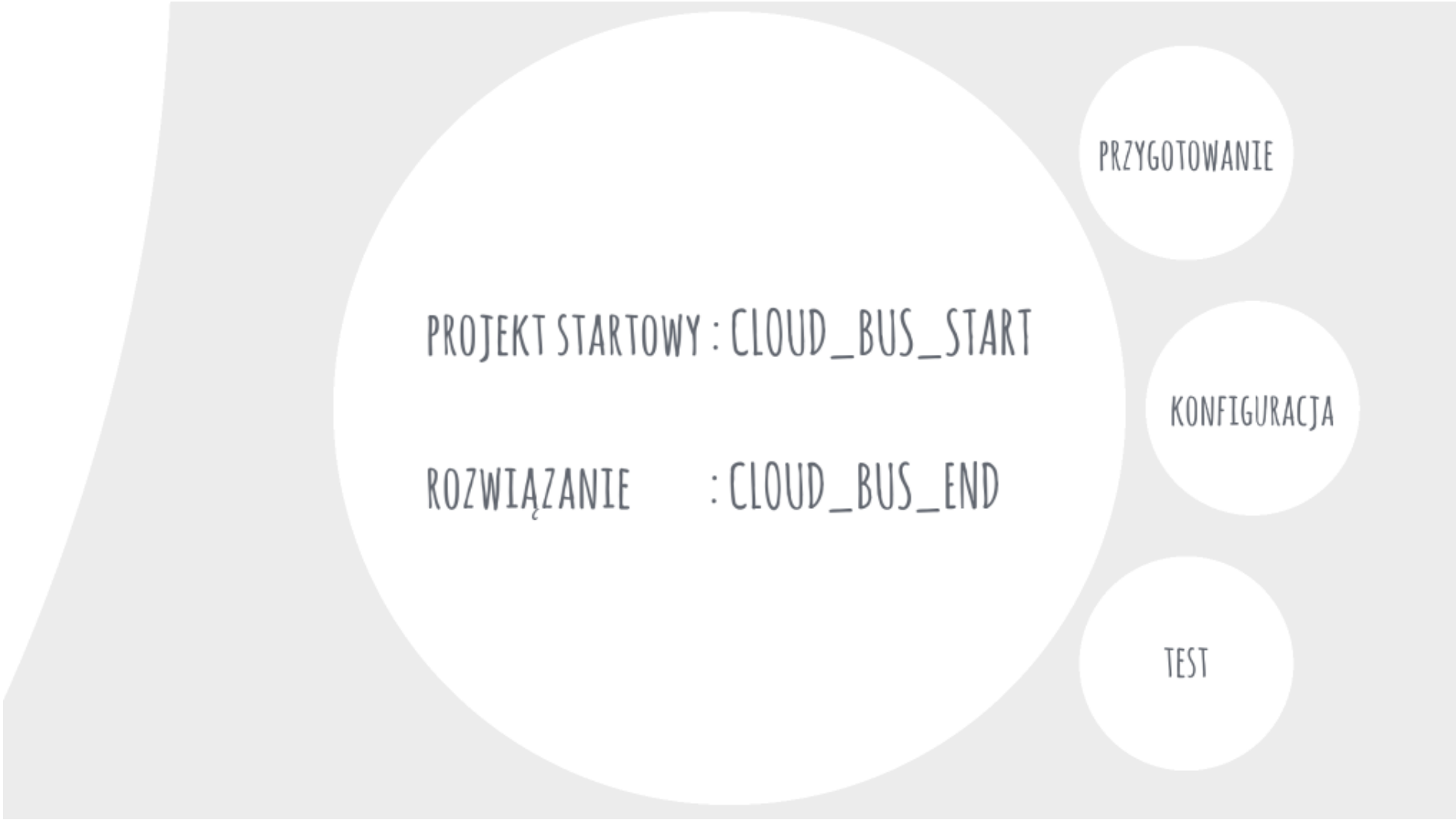
IV

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-security</artifactId>
  <version>2.0.0.RELEASE</version>
</dependency>
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-starter-oauth2</artifactId>
  <version>2.0.0.RELEASE</version>
</dependency>
```

```
security:
  oauth2:
    client:
      clientId: <wygenerowane-id>
      clientSecret: <wygenerowany-secret>
      accessTokenUri: https://www.googleapis.com/oauth2/v3/token
      userAuthorizationUri: https://accounts.google.com/o/oauth2/auth
      clientAuthenticationScheme: form
      scope: profile
    resource:
      userInfoUri: https://www.googleapis.com/userinfo/v2/me
      preferTokenInfo: false
```


Project
fortune-cookie-service ~/IdeaProjects/spring/sprign
.mvn
src
main
java
workshop
sc
fortunecookieservice
controller
feign
FortuneCookieServiceApplication
resources
test
target
.gitignore
mvnw
mvnw.cmd
pom.xml
FortuneCookieServiceApplication.java
1 package workshop.sc.fortunecookieservice;
2
3 import ...
13
14 @SpringBootApplication
15 @EnableDiscoveryClient
16 @EnableFeignClients
17 @EnableCircuitBreaker
18 @EnableOAuth2Sso
19 @EntityScan("workshop.sc.model")
20 public class FortuneCookieServiceApplication {
21
22 public static void main(String[] args) { Sp
25
26 }
27

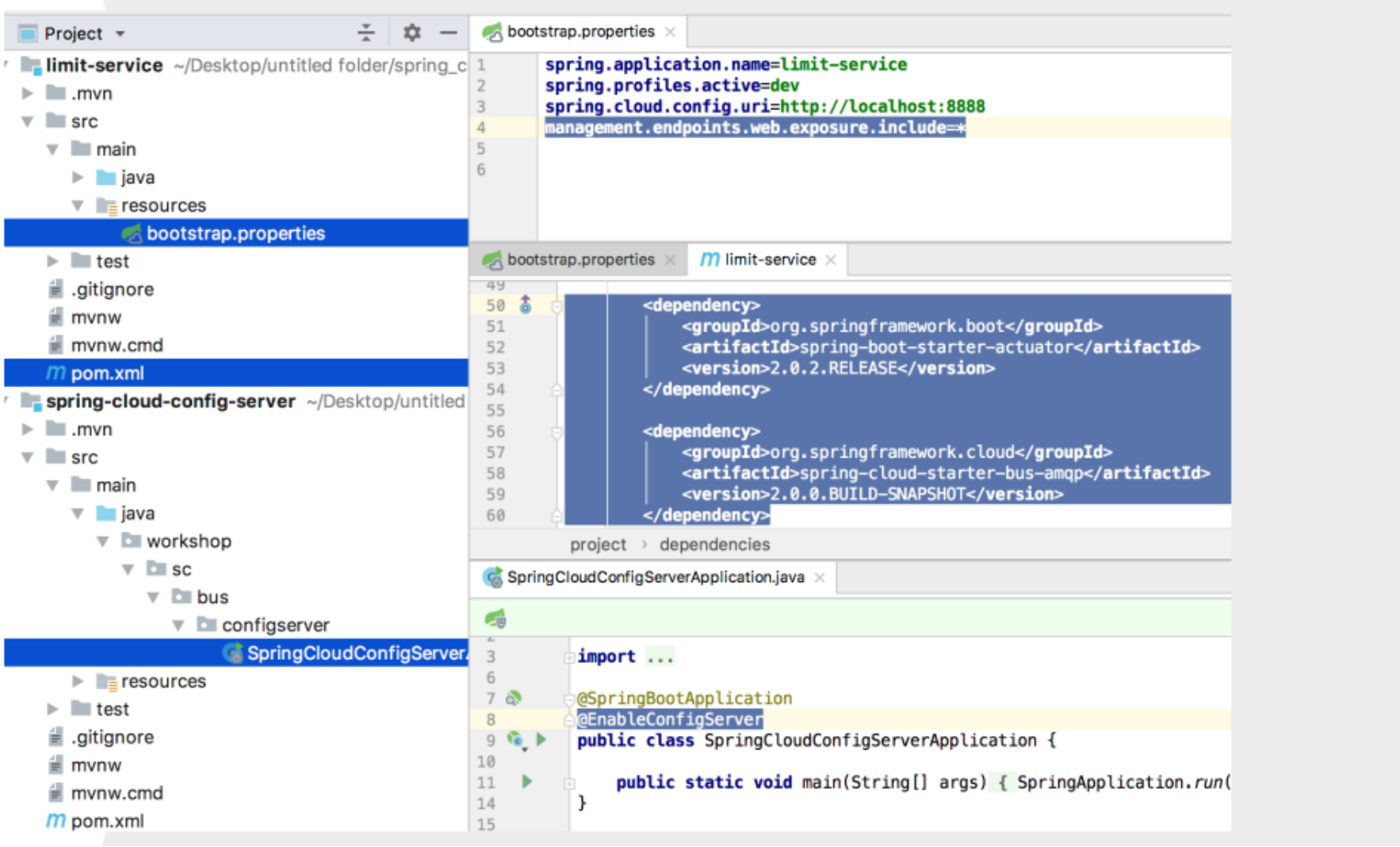

Zaloguj się przez Google
Zaloguj się
Przejdź do aplikacji fortune-cookie
Adres e-mail lub telefon
Nie pamiętasz adresu?
Utwórz konto
Dalej


PROJEKT STARTOWY : CLOUD_BUS_START
ROZWIĄZANIE : CLOUD_BUS_END
PRZYGOTOWANIE
KONFIGURACJA
TEST

ZAINSTALUJ RABBITMQ
HTTPS://WWW.RABBITMQ.COM/

HTTPS://WWW.JAVAINUSE.COM/MISC/RABBITMQ-HELLO-WORLD

ZAPOZNAJ SIĘ Z PROJEKTEM
ROZRYSUJ NA KARTCE ANALOGICZNIE JAK FORTUNE
MOŻNA DOPISAC PORTY / URL ETC.


Project
limit-service ~/Desktop/untitled folder/spring_c
.mvn
src
main
java
resources
bootstrap.properties
test
.gitignore
mvnw
mvnw.cmd
pom.xml
spring-cloud-config-server ~/Desktop/untitled
.mvn
src
main
java
workshop
sc
bus
configserver
SpringCloudConfigServer
resources
test
.gitignore
mvnw
mvnw.cmd
pom.xml
bootstrap.properties
1 spring.application.name=limit-service
2 spring.profiles.active=dev
3 spring.cloud.config.uri=http://localhost:8888
4 management.endpoints.web.exposure.include=*
5
6
bootstrap.properties
limit-service
50 <dependency>
51 <groupId>org.springframework.boot</groupId>
52 <artifactId>spring-boot-starter-actuator</artifactId>
53 <version>2.0.2.RELEASE</version>
54 </dependency>
55
56 <dependency>
57 <groupId>org.springframework.cloud</groupId>
58 <artifactId>spring-cloud-starter-bus-amqp</artifactId>
59 <version>2.0.0.BUILD-SNAPSHOT</version>
60 </dependency>
project › dependencies
SpringCloudConfigServerApplication.java
3 import ...
6
7 @SpringBootApplication
8 @EnableConfigServer
9 public class SpringCloudConfigServerApplication {
10
11 public static void main(String[] args) { SpringApplication.run(
14 }
15


TEST
URUCHOMIENIE
TEST
W
PRZEGLĄDARCE

SPRING-CLOUD-CONFIG-SERVER

LIMIT-SERVICE

- MINIMUM 2 INSTANCJE
- 1 PROFIL
- RÓŻNE PORTY

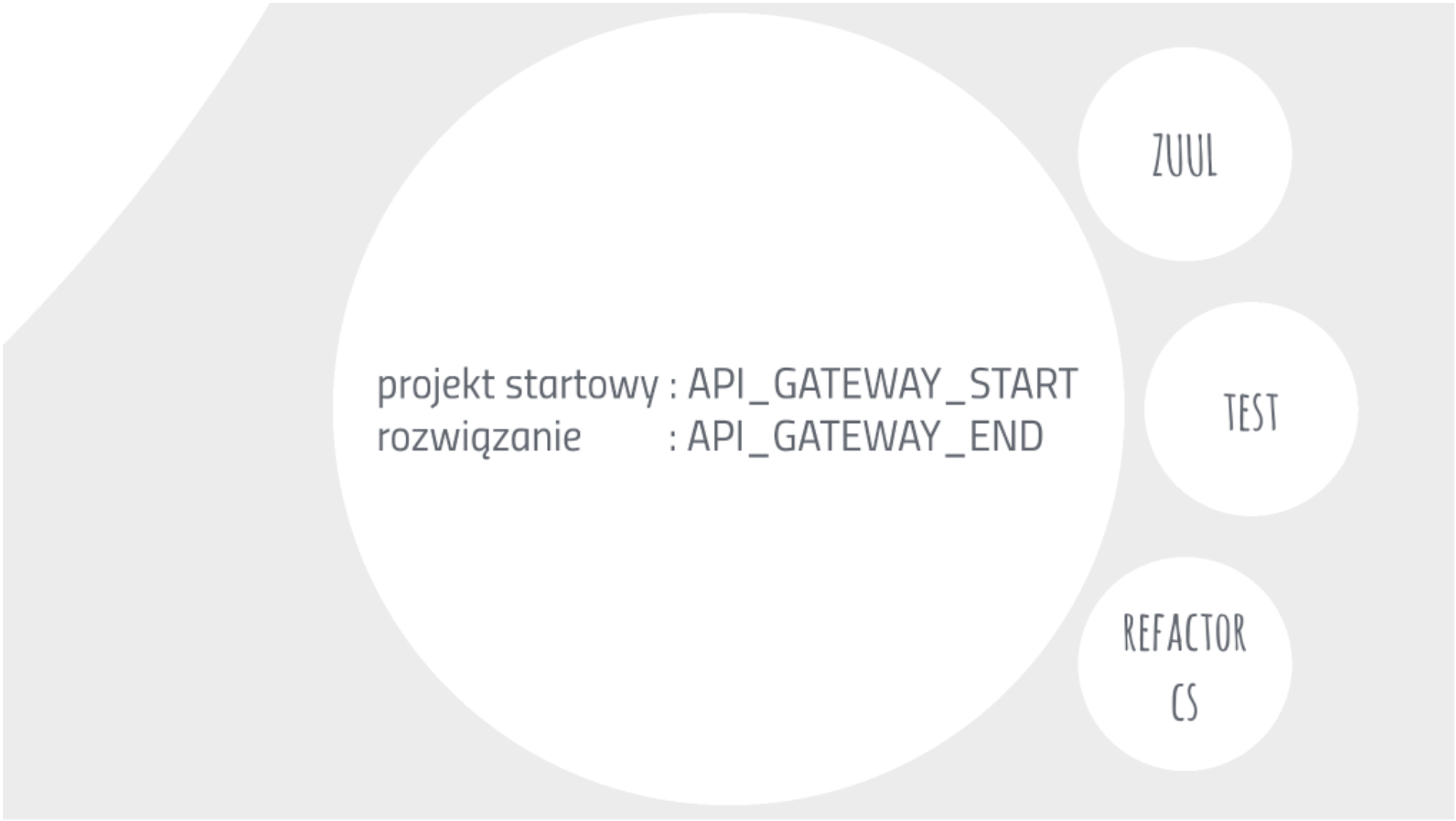
http://localhost:8080[8081]/limits

zmień plik konfiguracyjny dla profilu

zrestartuj serwer konfiguracyjny

odśwież obie instancje jednym komunikatem
    wybierz dowolną instancję (tylko jedna!)
    localhost:[port-instancji]/actuator/bus-refresh

sprawdź czy zaszła zmiana na obu instancjach


projekt startowy : API_GATEWAY_START
rozwiązanie : API_GATEWAY_END
ZUUL
TEST
REFACTOR
CS

nowy projekt - server zuul
spring-cloud-starter-netflix-eureka-client
spring-cloud-starter-netflix-zuul

OPIS

@

.PROPERTIES

ZUULFILTER

NOWY PROJEKT - SERVER ZUUL

- SPRING-CLOUD-STARTER-NETFLIX-EUREKA-CLIENT
- SPRING-CLOUD-STARTER-NETFLIX-ZUUL

W KLASIE GŁÓWNEJ

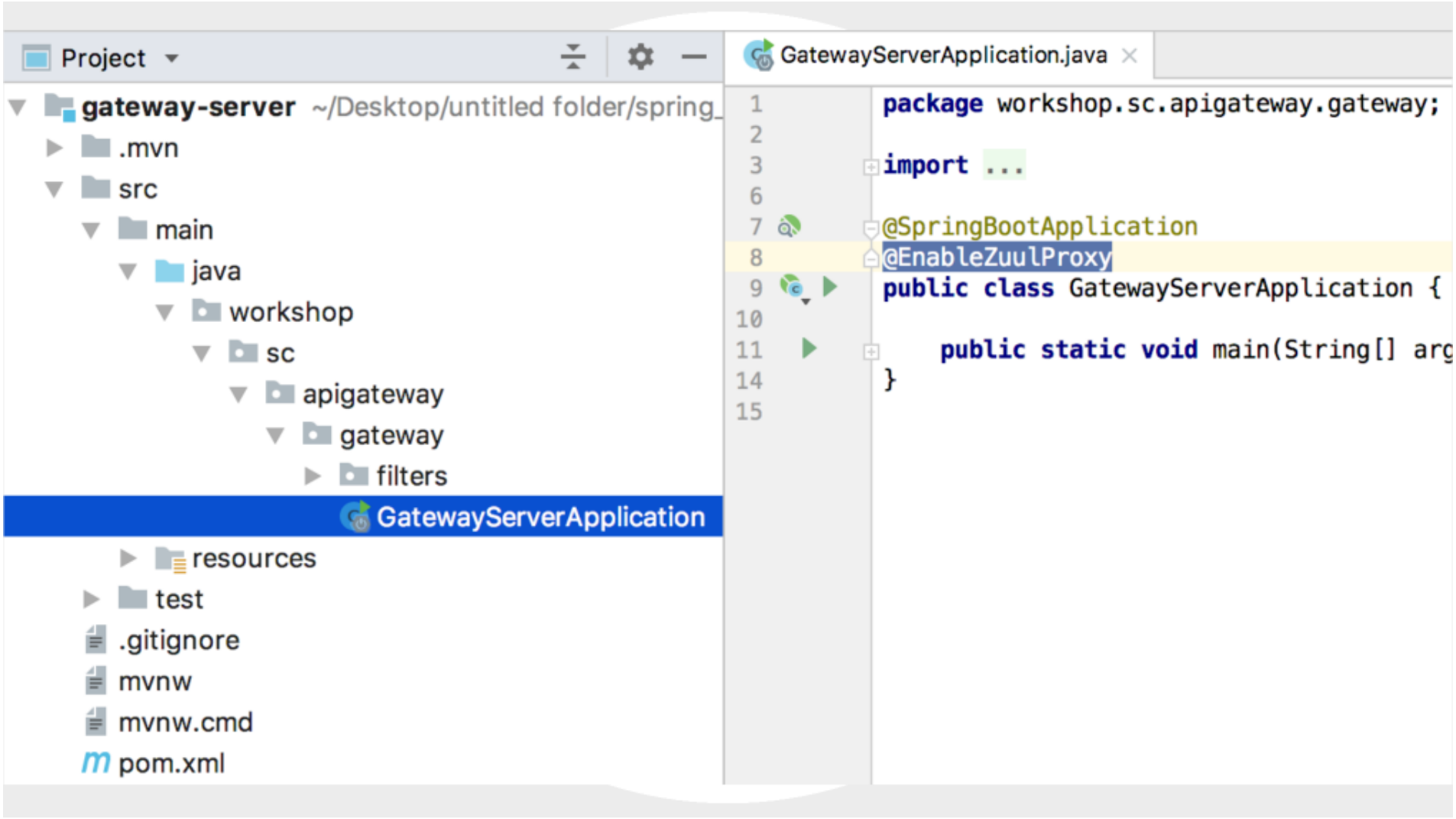
- @EnableZuulProxy

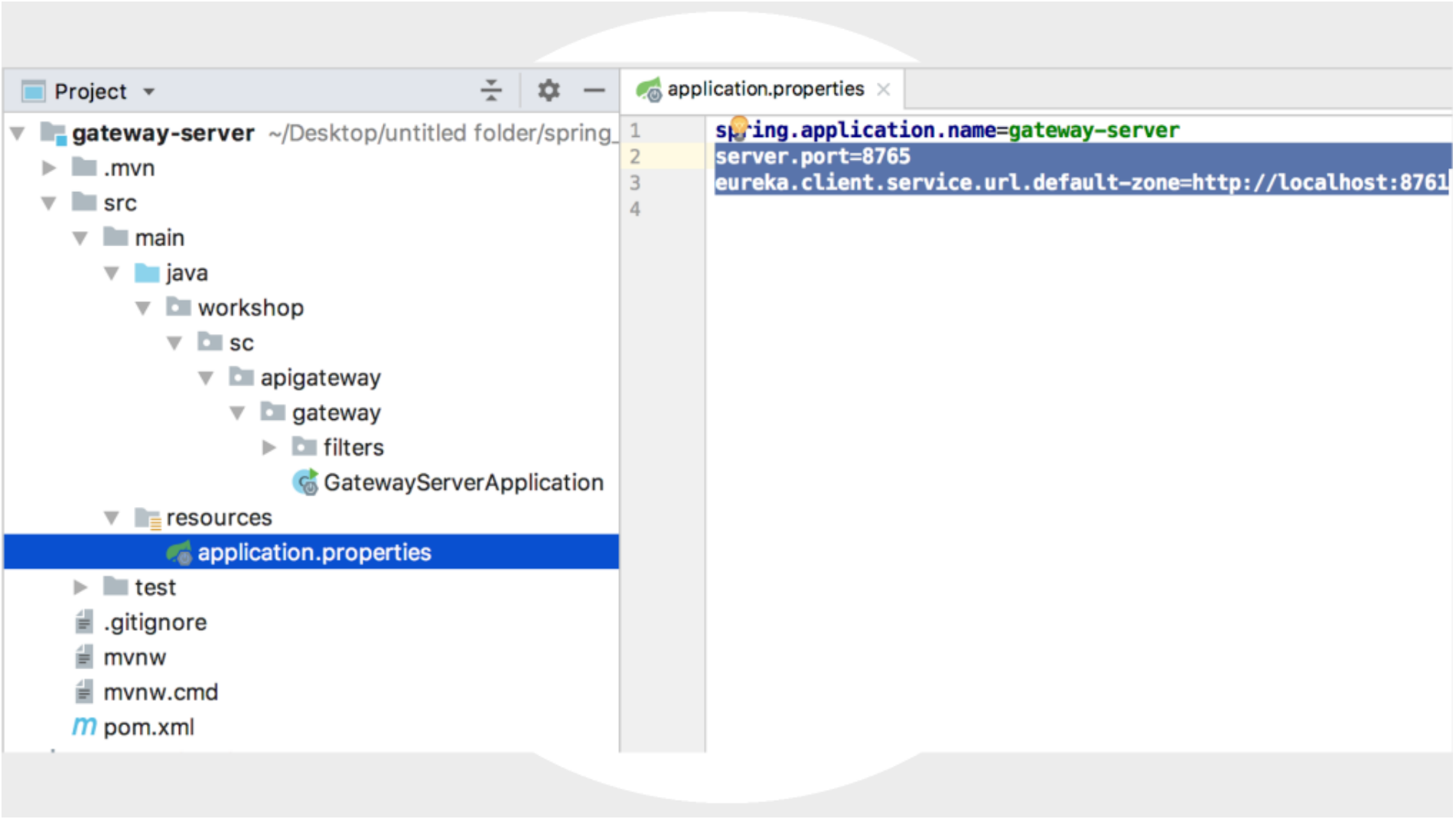
APPLICATION.PROPERTIES

- EUREKA.CLIENT.SERVICE.URL.DEFAULT-ZONE=HTTP://LOCALHOST:8761
- SERVER.PORT=8765

@Component ROZSZERZAJĄCY ZuulFilter

- W METODZIE RUN ZAIMPLEMENTUJ LOGOWANIE URI REQUESTU
- SHOULDFILTER ZWRACA TRUE
- FILTERTYPE ZWRACA "PRE"


Project
gateway-server ~/Desktop/untitled folder/spring_
.mvn
src
main
java
workshop
sc
apigateway
gateway
filters
GatewayServerApplication
resources
test
.gitignore
mvnw
mvnw.cmd
pom.xml
GatewayServerApplication.java
1 package workshop.sc.apigateway.gateway;
2
3 import ...
6
7 @SpringBootApplication
8 @EnableZuulProxy
9 public class GatewayServerApplication {
10
11 public static void main(String[] arg
14 }
15


Project
gateway-server ~/Desktop/untitled folder/spring_
.mvn
src
main
java
workshop
sc
apigateway
gateway
filters
GatewayServerApplication
resources
application.properties
test
.gitignore
mvnw
mvnw.cmd
pom.xml
application.properties
1 spring.application.name=gateway-server
2 server.port=8765
3 eureka.client.service.url.default-zone=http://localhost:8761
4

```java
@Component
public class LoggingFilter extends ZuulFilter {
    Logger log = Logger.getLogger(LoggingFilter.class.getName());

    @Override
    public String filterType() { return "pre"; // "post", "error" }

    @Override
    public int filterOrder() { return 1; }

    @Override
    public boolean shouldFilter() { return true; }

    @Override
    public Object run() throws ZuulException {
        HttpServletRequest request = RequestContext.getCurrentContext().getRequest();
        log.info(String.format("Zuul request log, URI: %s", request.getRequestURI()));
        return null;
    }
}
```

uruchamiamy

naming server, exchange i conversion

request przez gateway - ‘maska’

http://localhost:8765/<app-name>/<url>

przykład

http://localhost:8000/currency-exchange/from/EUR/to/PLN

http://localhost:8765/exchange-service/currency-exchange/from/EUR/to/PLN

```java
//@FeignClient(name="exchange-service")
@FeignClient(name = "gateway-server")
@RibbonClient(name="exchange-service")
public interface CurrencyExchangeServiceFeign {

//    @GetMapping("/currency-exchange/from/{from}/to/{to}")
    @GetMapping("/exchange-service/currency-exchange/from/{from}/to/{to}")
    CurrencyConversionBean exchangeRate(@PathVariable("from") String from,
                                        @PathVariable("to") String to);
}
```


docker
Wprowadzenie
Kontener
Obraz
Sieć
Dockerfile
Docker-compose


WPROWADZENIE
DLACZEGO
EDYCJE
KONTENER
VS
VM
KONTENER
VS
OBRAZ

UŁATWIA / PRZYSPIESZA CYKL WYTWARZANIA

MOŻLIWOŚĆ SZYBKIEGO URUCHAMIANIA PROGRAMU W RÓŻNYCH ŚRODOWISKACH

UŁATWIA DEVELOP. / TEST / DEPLOY W ROZPROSZONYCH APP.

W SKRÓCIE: ZWIĘKSZA WYDAJNOŚĆ

Linux vs Mac/Win vs chmura(AWS/Azure/Google)

CE (darmowa) vs EE (płatna, support, extras)

Edge(beta, miesiąc) vs Stable(kwartał)

KONTENERY TO NIE MINI-VM

TO PROCESY

O OGRANICZONYM DOSTĘPIE DO ZASOBÓW (PLIKI, SIEĆ, URZĄDZENIA, INNE PROCESY)

OBRAZ TO APLIKACJA, KTÓRĄ CHCEMY URUCHOMIĆ

KONTENER TO URUCHOMIONY PROCES - INSTANCJA TEGO OBRAZU

MOŻESZ MIEĆ WIELE KONTENERÓW URUCHOMIONYCH Z TEGO SAMEGO OBRAZU

GŁÓWNE REPO OBRAZÓW DLA DOCKER'A: DOCKER HUB

INSTALACJA, SPRAWDZENIE WERSJI DOCKER'A

UTWORZENIE KONTENERA Z OBRAZU

POZNANIE PODSTAWOWYCH KOMENDY DOCKER'A:
(RUN, START, STOP, PS, LOGS, RM)

RUN

RUN
VS
START

-P 8888:80

#

```
docker container run -p 80:80 nginx
```

CO SIĘ WYDARZYŁO?

Docker:

wyszukał obraz w lokalnym buforze

nie znalazł, załadował z Docker Hub najnowszy

utworzył nowy kontener na podstawie obrazu

przypisał wirtualne IP w prywatnej, wewnętrznej sieci

otworzył port 80 na hoście i przekierował na 80 kontenera

uruchomił kontener korzystając z CMD w Dockerfile obrazu

RUN TWORZY NOWY KONTENER

START URUCHAMIA ISTNIEJĄCY

STOP ZATRZYMUJE

JEŚLI UŻYWAMY ID KONTENERA Z KOMENDĄ (RUN, START ETC.)
ZAZWYCZAJ WYSTARCZĄ PIERWSZE 3 ZNAKI

1 KONTENER -> 1 PORT NA HOŚCIE

-P 8888:80

DOCKER CONTAINER TOP - PROCESY W KONTENERZE

NA WIN / MAC ZACHOWUJE SIĘ INACZEJ NIŻ NA LINUX

DOCKER CONTAINER INSPECT - SZCZEGÓŁY KONFIGURACJI

DOCKER CONTAINER STATS - CPU, MEM ETC.

I ZAPRZYJAŹNIENIE SIĘ Z DOKUMENTACJĄ

II UTWORZENIE 3 KONTENERÓW

III TEST I USUNIĘCIE KONTENERÓW

I

II

III

ROZWIĄZANIE

ZAPRZYJAŹNIJ SIĘ Z --HELP I DOCS.DOCKER.COM :)

wyświetl listę kontenerów (ps / ls [-a])

do wystartowania 3 kontenery : nginx, httpd, mysql

każdy z nich w tle (-d)

każdy z nich z nadpisaną nazwą (--name)

porty dla serwerów: nginx 80:80, httpd 8080:80 ,mysql 3306:3306

dla mysql'a przekaż parametr:
MYSQL_RANDOM_ROOT_PASSWORD=yes (--env)

WYŚWIETL LISTĘ KONTENERÓW

SPRAWDŹ W LOGACH MYSQL GENERATED ROOT PASSWORD

USUŃ UTWORZONE KONTENERY

WYŚWIETL LISTĘ KONTENERÓW

I utworzenie 3 kontenerów

II logi w MySQL

III usunięcie kontenerów

I

II

III

```
docker container ls (-a)

docker container run -d -p 80:80 --name the_nginx nginx

docker container run -d -p 8080:80 \
--name the_httpd httpd

docker container run --name the_mysql -d -p 3306:3306
--env MYSQL_RANDOM_ROOT_PASSWORD=yes mysql

docker container ls (-a)
```

```
docker container logs ID_KONTENERA 2>/dev/null \
  grep "GENERATED ROOT PASSWORD"
```

lub docker logs ID_KONTENERA i wyszukać w wyświetlonym logu

```
docker stop the_nginx the_httpd the_mysql
# jeśli bez stop, przy rm flaga -f

docker rm the_nginx the_httpd the_mysql

# poniżej niebezpieczna alternatywa - wszystkie!
docker rm -f $(docker ps -aq)

docker container ls (-a)
```

```
docker container run -it ubuntu bash

docker container exec -it <id> bash

docker container attach <id>

# exit vs Ctrl + PQ
```

2 kontenery z dystrybucjami linux: ubuntu i centos

instalacja curl w kontenerze i sprawdzenie jego wersji

ubuntu: apt-get update && apt-get install curl

centos: yum update curl

ROZWIĄZANIE

przy uruchomieniu użyj opcji --rm (auto-sprzątanie)

```
docker container run -it --rm ubuntu

#apt-get update && apt-get install curl
#curl —version
#exit

docker container run -it --rm centos
#yum update curl
#curl —version
#exit
```

docker image<polecenie> ?

ls

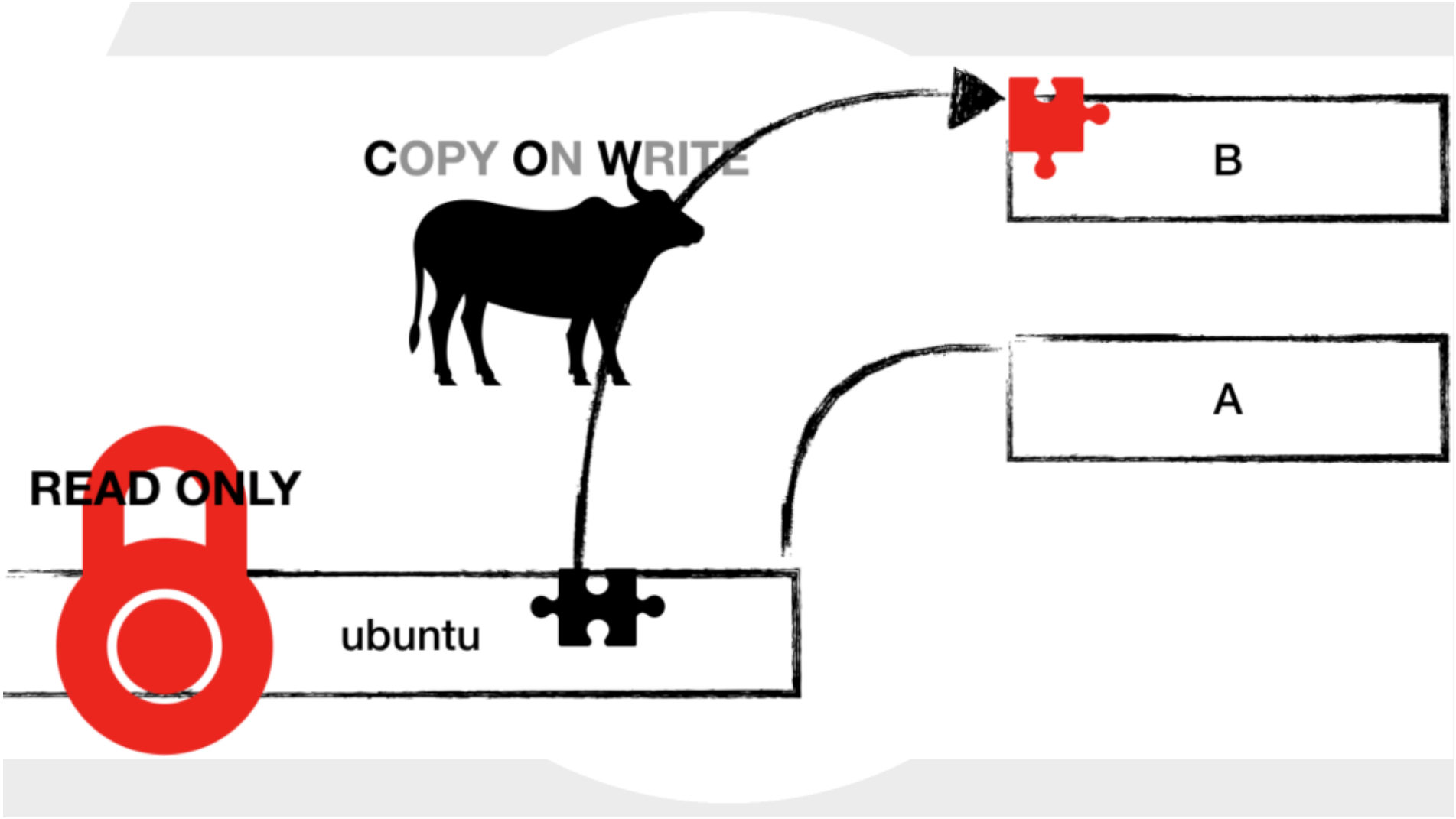
history <image>

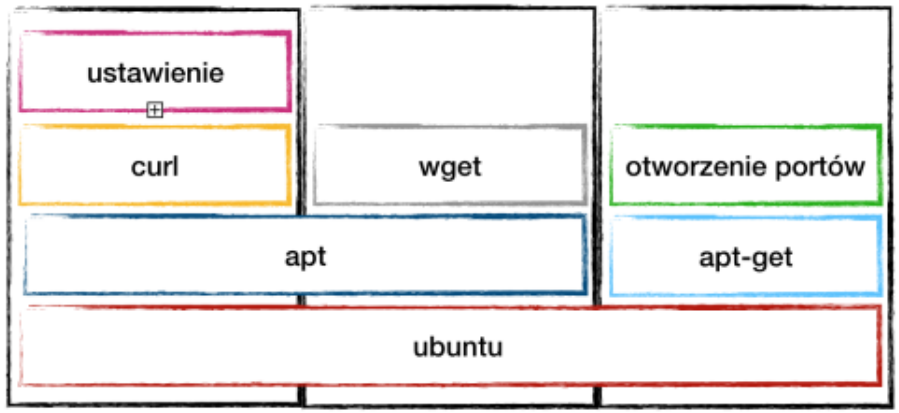
inspect <obraz>

build [-f custom-name]

KANAPKA

KROWA


COPY ON WRITE
B
A
READ ONLY
ubuntu


ustawienie
curl
wget
otworzenie portów
apt
apt-get
ubuntu

OBRAZ TO
'BINARKI' APLIKACJI I POTRZEBNE ZALEŻNOŚCI
ORAZ METADANE JAK JE RUCHAMIAĆ

OBRAZ NIE JEST KOMPLETNYM OS'EM !

docker network <polecenie>

ls

inspect --format '{{.NetworkSettings}}'

create --driver

connect

disconnect

OPIS

DNS

DIAGRAM

KAŻDY KONTENER PODŁĄCZANY DO WIRTUALNEJ SIECI

W RAMACH PRYWATNEJ SIECI K. MOGĄ KOMUNIKOWAĆ SIĘ BEZPOŚREDNIO (BEZ WYSTAWIANIA PORTÓW)

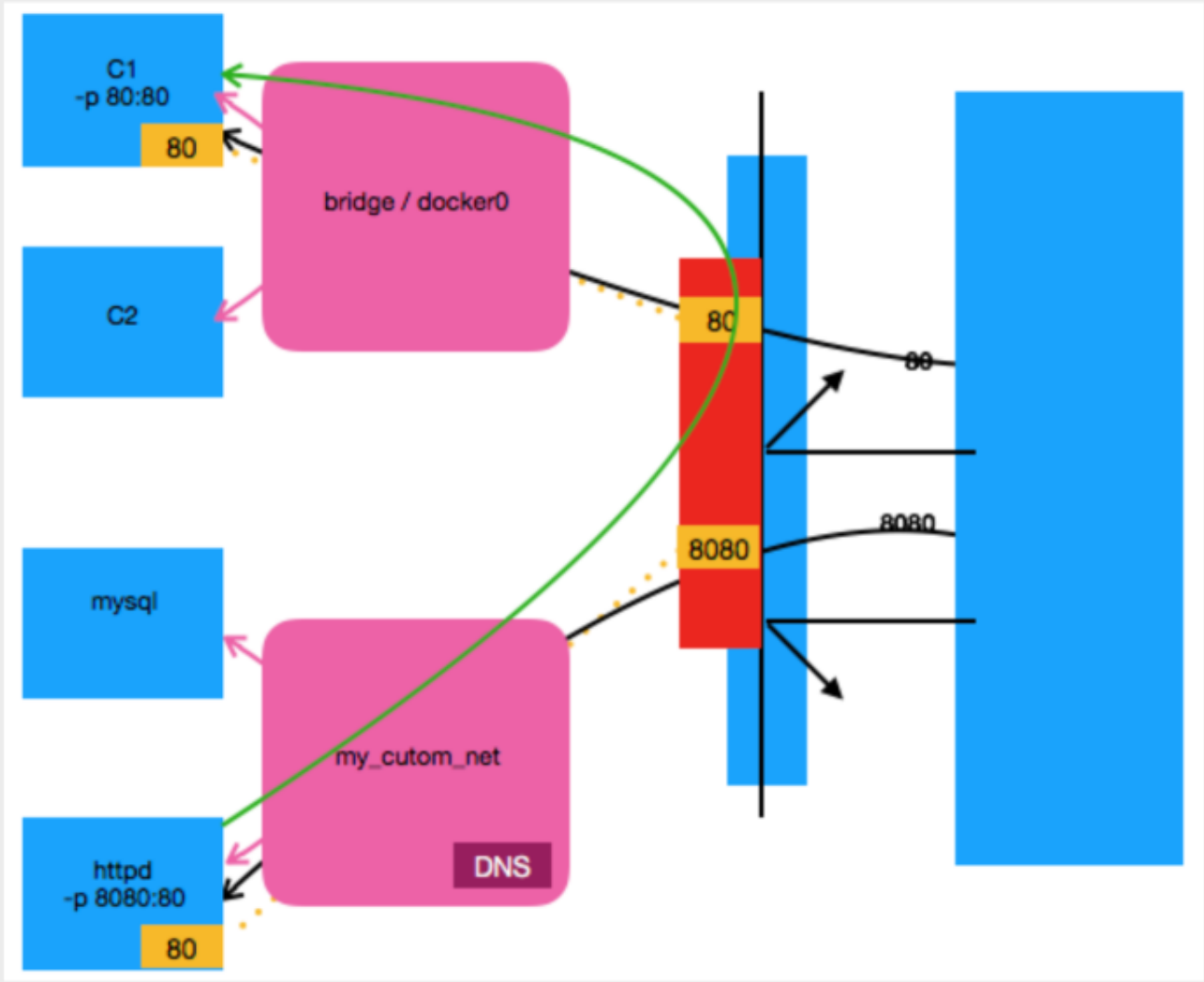
MOŻNA ZMIENIĆ ZACHOWANIE ZE STEROWNIKA BRIDGE (DOMYŚLNY) NA
HOST, OVERLAY, MACVLAN, NONE, LUB 3RD-PARTY PLUGIN

DOCKER MA WBUDOWANY SERVER DNS

UTWORZONA, WŁASNA SIEĆ: AUTOMATIC DNS RESOLUTION, DLA KONTENERÓW W TEJ SIECI

W DOMYŚLNEJ SIECI (BRIDGE) NIE MA DNS'A (MOŻNA UŻYĆ --LINK, ALE LEPSZA WŁASNA SIEĆ)

DOMYŚLNA NAZWA - NAZWA KONTENERA (MOŻNA UŻYĆ ALIAS)


C1
-p 80:80
80
C2
bridge / docker0
mysql
my_cutom_net
DNS
httpd
-p 8080:80
80
80
8080
80
8080

I UTWÓRZ WŁASNĄ SIEĆ

II WYSTARTUJ 2 KONTENERY:
NGINX:ALPINE (DETACHED)

III PRZETESTUJ:
(UŻYJ EXEC) PING Z JEDNEGO K. DO DRUGIEGO,
UŻYWAJĄC DOMYŚLNEGO DNS'A

I

II

III

```
docker network create my_network
```

```
docker container run --name alpine1 -d --network=my_network nginx:alpine

docker container run --name alpine2 -d --network=my_network nginx:alpine
```

```
docker network inspect my_network

docker container exec -it alpine1 ping alpine2
```

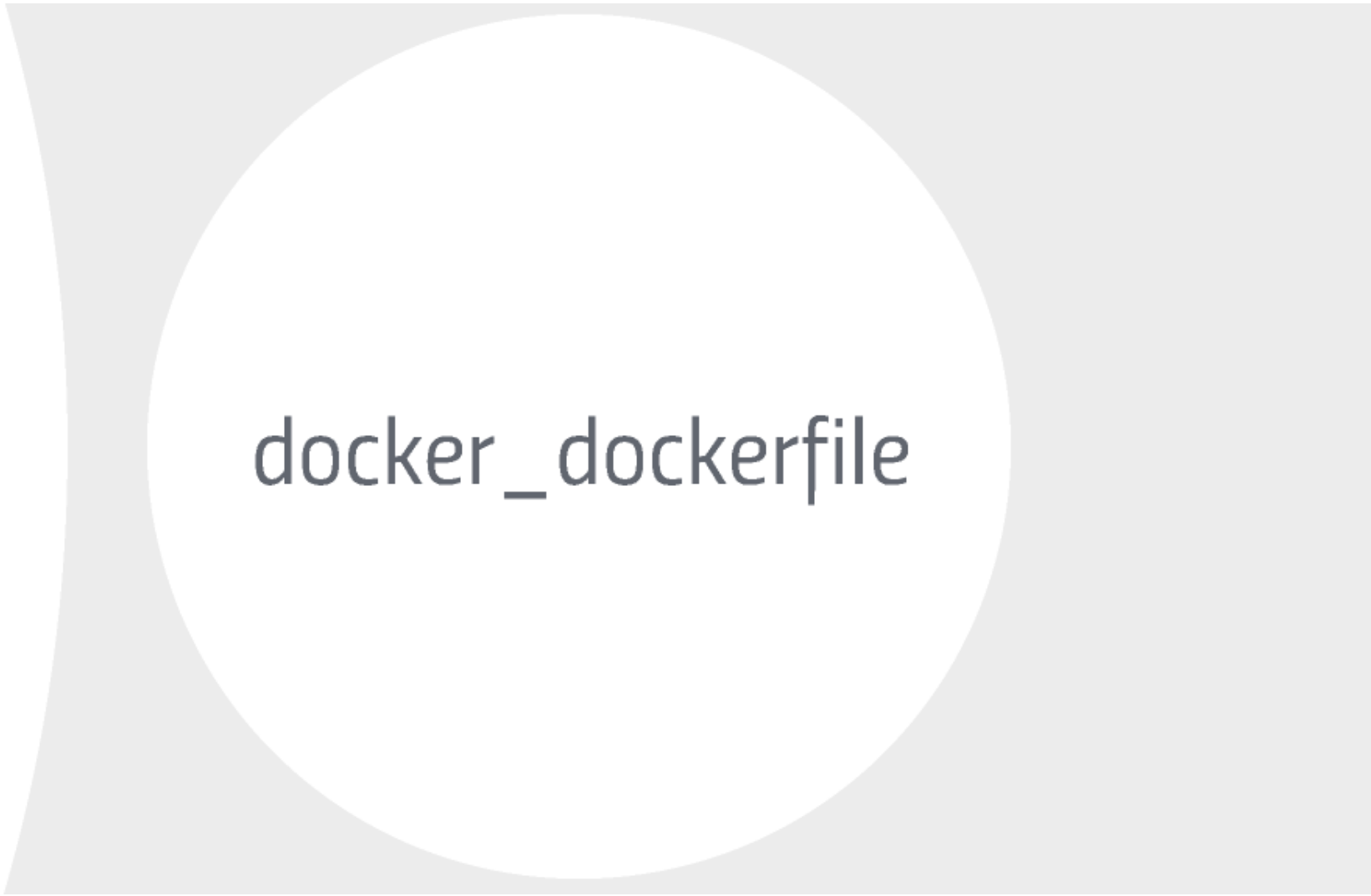
FROM - ZAZWYCZAJ Z PARENT IMAGE, MOŻNA OD 0 (SCRATCH)

ENV - ZMIENNE SYSTEMOWA

RUN - KOMENDY URUCHAMIANE PODCZAS BUDOWANIA KONTENERA

EXPOSE - WYSTAWIENIE PORTÓW
FLAGA -P PODCZAS DOCKER RUN LUB
PORTS W DOCKER-COMPOSE

CMD - ZDEFINIOWANIE, CO BĘDZIE ODPALONE PRZY URUCHOMIENIU KONTENERA


docker_dockerfile

KONFIGURACJA ZALEŻNOŚCI POMIĘDZY KONTENERAMI

ZAPISANIE KONFIGURACJI W JEDNYM PLIKU

'ONE-LINER' DLA POSTAWIENIA ROZPROSZONEGO ŚROD.

?

NA PRODUKCJI - DOCKER SWARM, KUBERNETES ETC.

SWARM VS KUBERNETES

HTTPS://PLATFORM9.COM/BLOG/KUBERNETES-DOCKER-SWARM-COMPARED/

PLIK KONFIGURACYJNY DOCKER-COMPOSE.YML

docker-compose.yml

OPCJE DLA KONTENERÓW, SIECI, ITD.

CLI DOCKER-COMPOSE

docker-compose CLI

NARZĘDZIE DLA AUTOMATYZACJI, UŻYWA PLIKU CONFIG.

format YAML, pierwsza linia to wersja pliku

1, 2, 2.1, 3, 3.1

wyższa wersja daje więcej opcji

domyślna v1 - rekomendowana: >= v2

lokalnie z docker-compose(test / dev)

używany z docker produkcyjnie

od v1.13, ze Swarm

docker-compose.yml domyślna

własna: docker-compose -f

DO LOKALNEJ PRACY

```
DOCKER-COMPOSE UP     # START
DOCKER-COMPOSE DOWN # CLEAN UP
```

ZAPOZNANIE NOWEJ OSOBY Z PROJEKTEM:

```
GIT CLONE GITHUB.COM/REPO/PROJEKTU
DOCKER-COMPOSE UP
```

docker_compose

docker_compose_multi

TWORZENIE MIKROSERWISÓW Z WYKORZYSTANIEM SPRING CLOUD I DOCKER