Mikroservisi sa Spring Cloud Netflix

ANTONIO KATUŠIĆ
INOVATIVNI TRENDOVI D.O.O

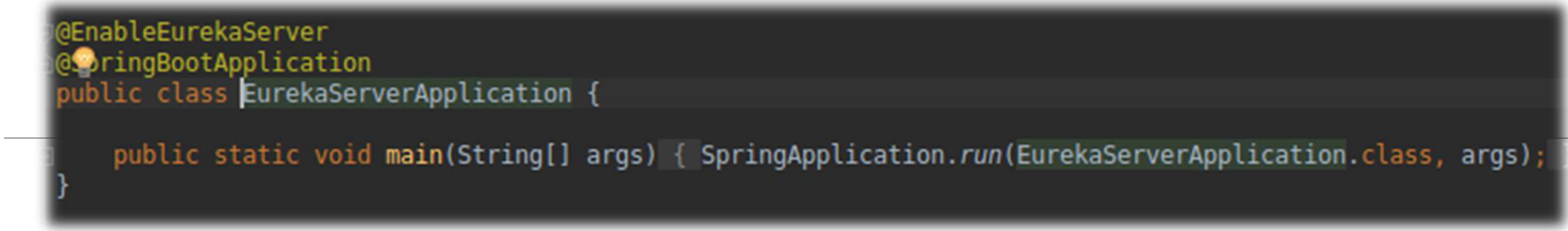
Sadržaj

Mikroservisi na Spring Boot 2.0
Otkrivanje servisa (Service Discovery) – Eureka
Uravnotežavanje opterećenja (Load Balancer) – Ribbon
Zaštita (Circuit Breaker) – Hystrix
Nadgledavanje (Monitoring) – Hystrix Dashboard i Turbine
Preusmjeravanje i filtriranje (Routing and filtering) - Zuul

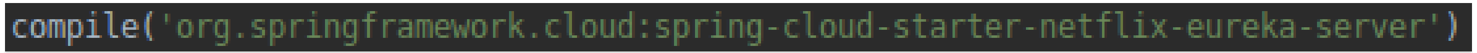
NETFLIX
OSS

Eureka

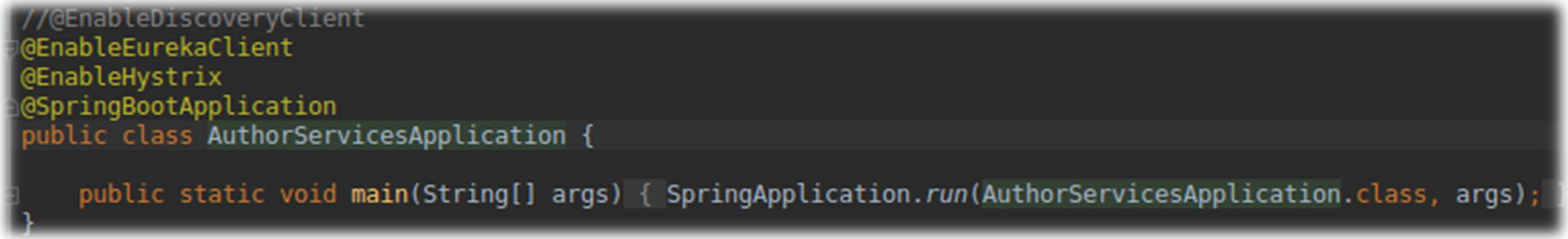
- Omogućuje otkrivanje servisa
- Podijeljeno na Eureka Server i Eureka Klijente – abstrakcija:
  1. Eureka Server – registar ostalih mikroservisa
  2. Eureka Client – mikroservisi
- Konfiguracija:
  1. application.yml/.properties-a ili
  2. Spring cloud config server
  3. Oboje – koristiti application.yml kao fallback
- Heartbeat-ovi i preuzimanje informacija
- Sigurnost preko url-a:
  - ◦ http://user:password@localhost:8761/eureka
  - ◦ HTTPS
- Eureka dashboard – za monitoring stanja eureka klijenata
- Više aktivnih instanci istog mikroservisa
- Statusi: UP, STARTING, DOWN i UNKNOWN
- Koristi Ribbon za uravnoteženje otperećenja – opcionalno



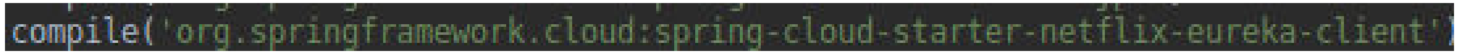
```
@EnableEurekaServer
@SpringBootApplication
public class EurekaServerApplication {

    public static void main(String[] args) { SpringApplication.run(EurekaServerApplication.class, args); }
}
```



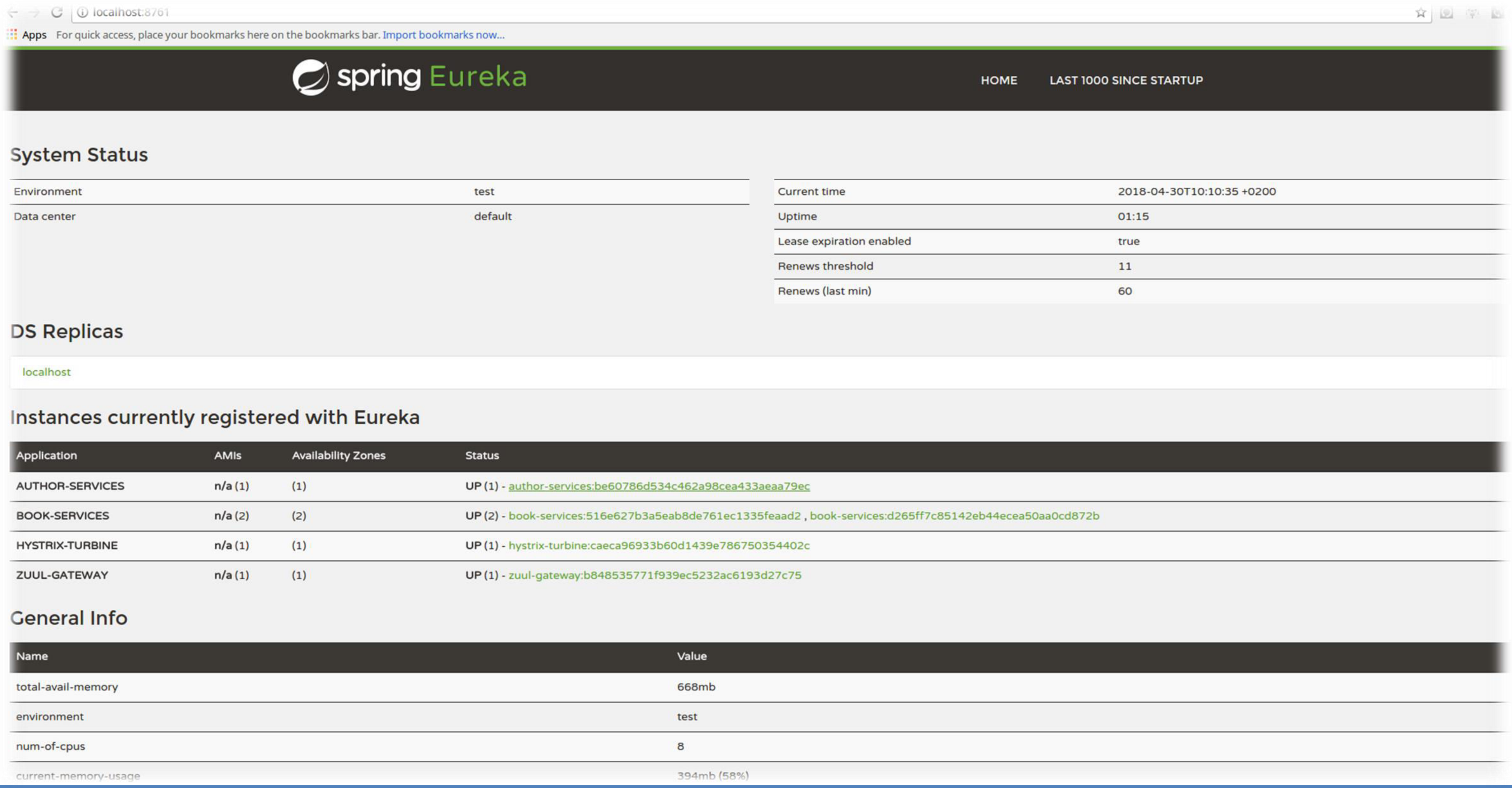
```
compile('org.springframework.cloud:spring-cloud-starter-netflix-eureka-server')
```



```
//@EnableDiscoveryClient
@EnableEurekaClient
@EnableHystrix
@SpringBootApplication
public class AuthorServicesApplication {

    public static void main(String[] args) { SpringApplication.run(AuthorServicesApplication.class, args); }
}
```



```
compile('org.springframework.cloud:spring-cloud-starter-netflix-eureka-client')
```


localhost:8761
Apps
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
spring Eureka
HOME
LAST 1000 SINCE STARTUP

System Status

Environment	test
Data center	default

Current time	2018-04-30T10:10:35 +0200
Uptime	01:15
Lease expiration enabled	true
Renews threshold	11
Renews (last min)	60

DS Replicas

localhost

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
AUTHOR-SERVICES	n/a (1)	(1)	UP (1) - author-services:be60786d534c462a98cea433aeaa79ec
BOOK-SERVICES	n/a (2)	(2)	UP (2) - book-services:516e627b3a5eab8de761ec1335feaad2 , book-services:d265ff7c85142eb44ecea50aa0cd872b
HYSTRIX-TURBINE	n/a (1)	(1)	UP (1) - hystrix-turbine:caeca96933b60d1439e786750354402c
ZUUL-GATEWAY	n/a (1)	(1)	UP (1) - zuul-gateway:b848535771f939ec5232ac6193d27c75

General Info

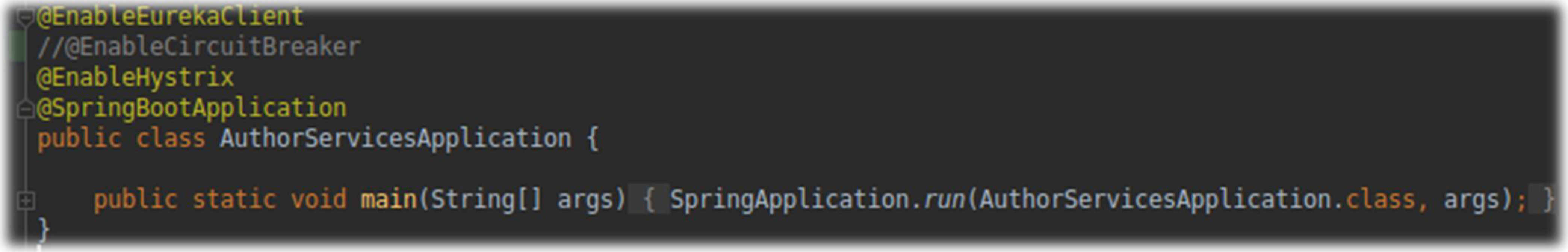
Name	Value
total-avail-memory	668mb
environment	test
num-of-cpus	8
current-memory-usage	394mb (58%)

Ribbon

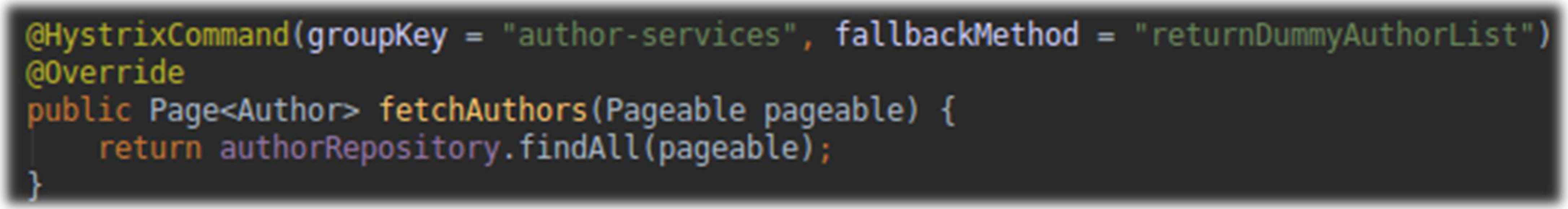
- Uravnoteženje opterećenja na klijent strani „client-side load balancer”
- Riješava problema koji su prisutni kod server-side load balancer-a
- Konfigurabilnost
  - konfiguracija ribbon-a po servisu
  - mogućnost korištenja Ribbon API-a direktno
- Predefinirani algoritmi za uravnoteženje upita (Round Robin po default-u)
- U Spring Cloud Netflix Eureka i Ribbon se koriste zajedno
  - Moguće je koristiti i Ribbon bez konfiguracije
  - U tom slučaju mora se u konfiguraciji dodati lista server-a
- Može služiti također za otkrivanje servisa umjesto Eureke
  - U tom slučaju morate definirati sami listu server-a

Hystrix

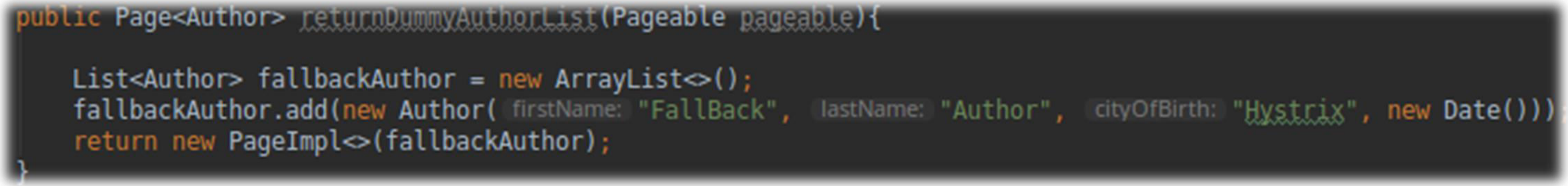
- Implementacija „Circuit Breaker Pattern-a" (Analogija strujnog kruga)
- Pruža visoku otpornost mikroservisa na kaskadna rušenja
- Pruža mogućnost definiranja „fallback-ova" u slučaju „otvaranja strujnog kruga"
- Omogućuje vrijeme oporavka mikroservisu
- Upiti su obavijeni u HystrixCommand objekt (Hystrix – Javanica)
- Pruža veliku fleksibilnost za konfiguraciju ponašanja pojedine metode
- HystrixCommand moguće staviti za sinkrone, asinkrone i reaktivne (observables) pozive
- Na /hystrix endpoint-u dostupne informacije o metodama sa HystrixCommand anotacijom



```java
@EnableEurekaClient
//@EnableCircuitBreaker
@EnableHystrix
@SpringBootApplication
public class AuthorServicesApplication {

    public static void main(String[] args) { SpringApplication.run(AuthorServicesApplication.class, args); }
}
```



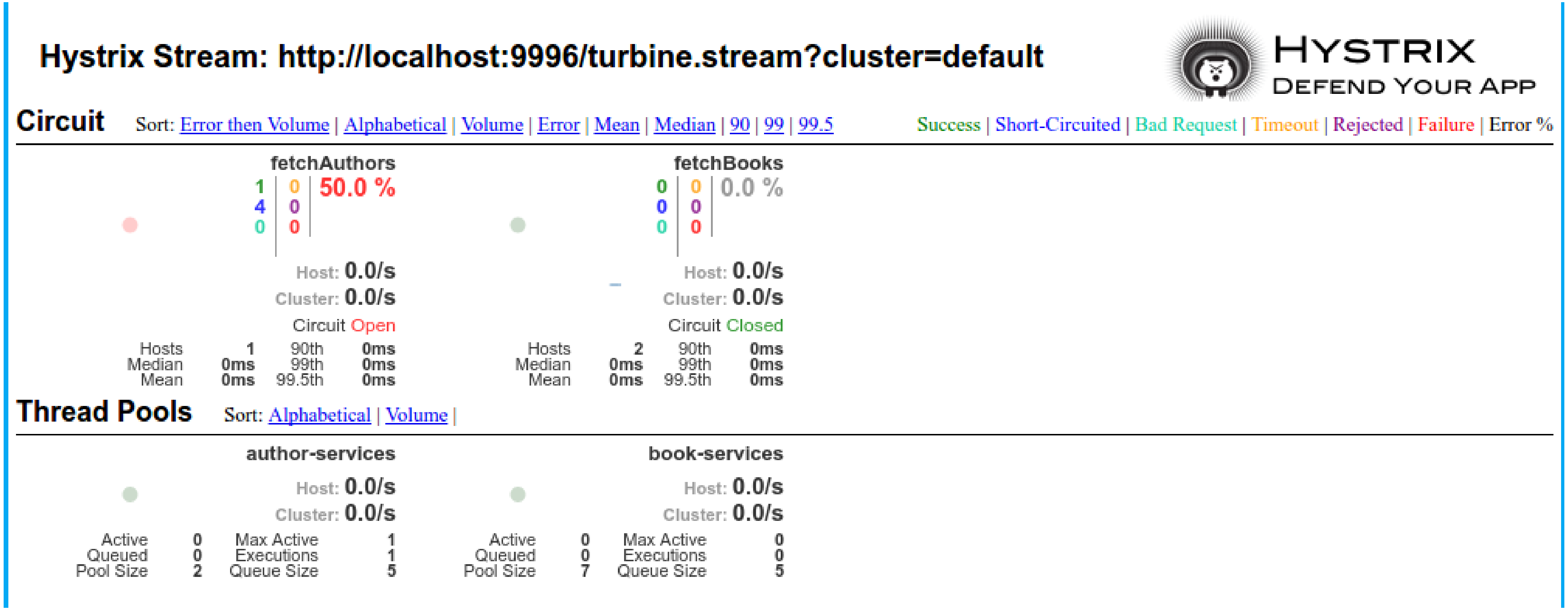
```
@HystrixCommand(groupKey = "author-services", fallbackMethod = "returnDummyAuthorList")
@Override
public Page<Author> fetchAuthors(Pageable pageable) {
    return authorRepository.findAll(pageable);
}
```



```
public Page<Author> returnDummyAuthorList(Pageable pageable){

    List<Author> fallbackAuthor = new ArrayList<>();
    fallbackAuthor.add(new Author( firstName: "FallBack", lastName: "Author", cityOfBirth: "Hystrix", new Date()));
    return new PageImpl<>(fallbackAuthor);
}
```

Hystrix dashboard i turbine

- Hystrix dashboard pruža grafički prikaz HystrixCommand statistike i stanja
- U svakom HystrixCommand-u se može definirati metadata za dashboard
- Za nadgledanje jednog mikroservisa dovoljan je samo dashboard
- Za nadgledanje većeg broja mikroservisa potrebno je izložiti njihove podatke
- Turbine služi za agregaciju hystrix podataka i real time statistike



Hystrix Stream: http://localhost:9996/turbine.stream?cluster=default

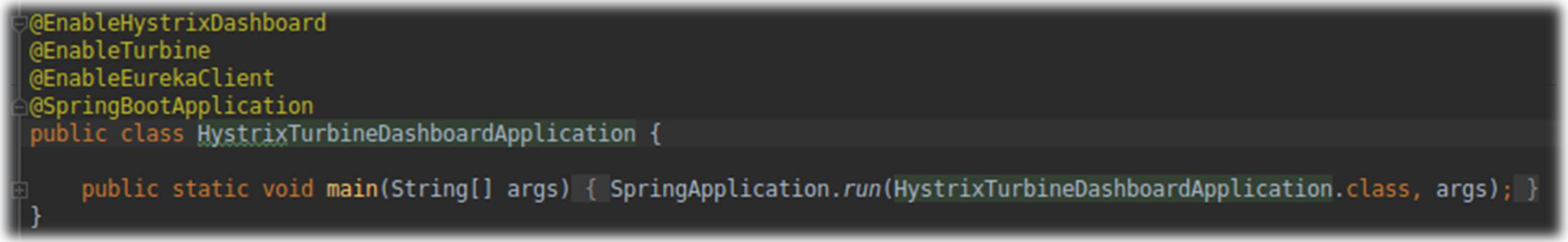
Circuit Sort: Error then Volume | Alphabetical | Volume | Error | Mean | Median | 90 | 99 | 99.5

Success | Short-Circuited | Bad Request | Timeout | Rejected | Failure | Error %

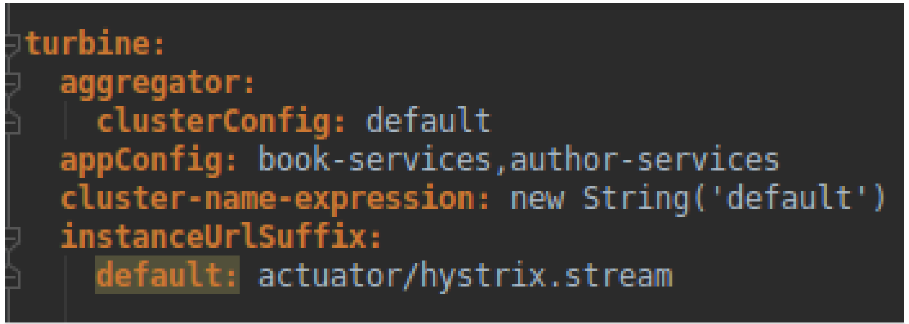
```
                 fetchAuthors                         fetchBooks
              1 | 0 | 50.0 %                       0 | 0 | 0.0 %
              4 | 0 |                              0 | 0 |
              0 | 0 |                              0 | 0 |
               Host: 0.0/s                          Host: 0.0/s
            Cluster: 0.0/s                       Cluster: 0.0/s
               Circuit Open                         Circuit Closed
Hosts      1   90th    0ms           Hosts      2   90th    0ms
Median   0ms   99th    0ms           Median   0ms   99th    0ms
Mean     0ms   99.5th  0ms           Mean     0ms   99.5th  0ms
```

Thread Pools Sort: Alphabetical | Volume |

```
              author-services                      book-services
               Host: 0.0/s                          Host: 0.0/s
            Cluster: 0.0/s                       Cluster: 0.0/s
Active     0  Max Active   1         Active     0  Max Active   0
Queued     0  Executions   1         Queued     0  Executions   0
Pool Size  2  Queue Size   5         Pool Size  7  Queue Size   5
```



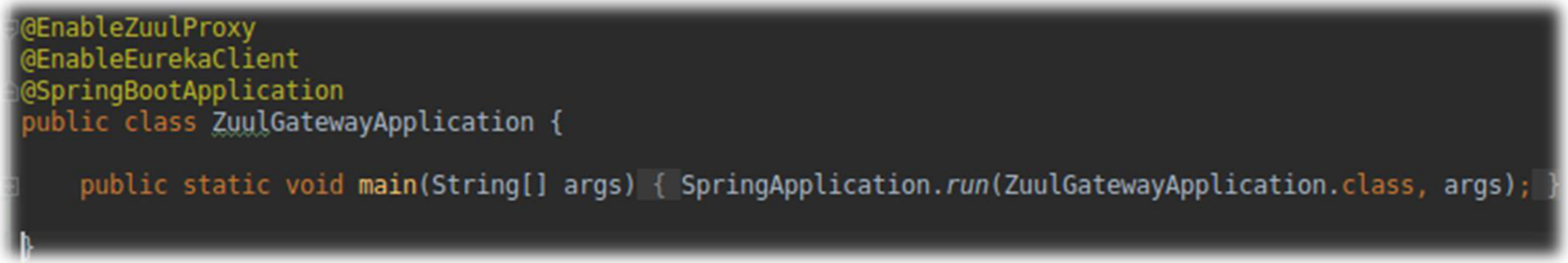
```java
@EnableHystrixDashboard
@EnableTurbine
@EnableEurekaClient
@SpringBootApplication
public class HystrixTurbineDashboardApplication {

    public static void main(String[] args) { SpringApplication.run(HystrixTurbineDashboardApplication.class, args); }
}
```



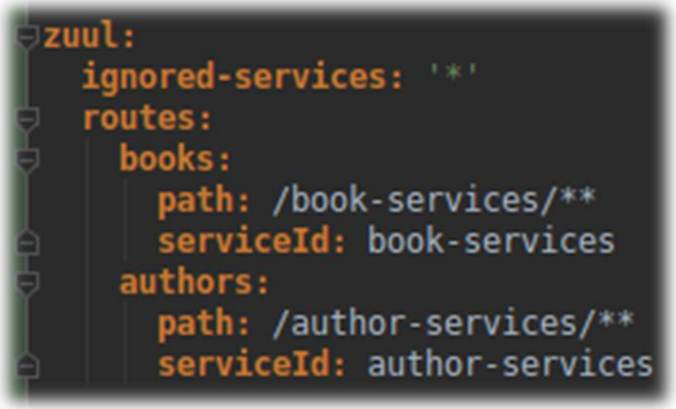
```yaml
turbine:
  aggregator:
    clusterConfig: default
  appConfig: book-services,author-services
  cluster-name-expression: new String('default')
  instanceUrlSuffix:
    default: actuator/hystrix.stream
```

ZUUL

- Gateway za preusmjerevanje i filtriranje upita na mikroservise
- Koristi ostale netflix tehnologije Eureka, Ribbon i Hystrix
  - Korištenje ovih tehnologija u zuulu je po „default-u" uključeno, po želji moguće isključiti
- Konfiguracija putanji **ne** mora biti ovisna o ip-u mikroservisa
- U konfiguraciji se mogu dodati putanje za ostale mikroservise na dva načina:
  1. Definirati URL (requestovi na ovakav način neće koristiti ribbon i hystrix)
  2. Koristiti serviceId <- preferiran način
- Pruža mogućnost definiranja vlastitih filtera
- Tipovi filtara za upite: PRE, ROUTING i POST
- @EnableZuulProxy vs @EnableZuulServer



```java
@EnableZuulProxy
@EnableEurekaClient
@SpringBootApplication
public class ZuulGatewayApplication {

    public static void main(String[] args) { SpringApplication.run(ZuulGatewayApplication.class, args); }

}
```



```yaml
zuul:
  ignored-services: '*'
  routes:
    books:
      path: /book-services/**
      serviceId: book-services
    authors:
      path: /author-services/**
      serviceId: author-services
```

HVALA NA PAŽNJI