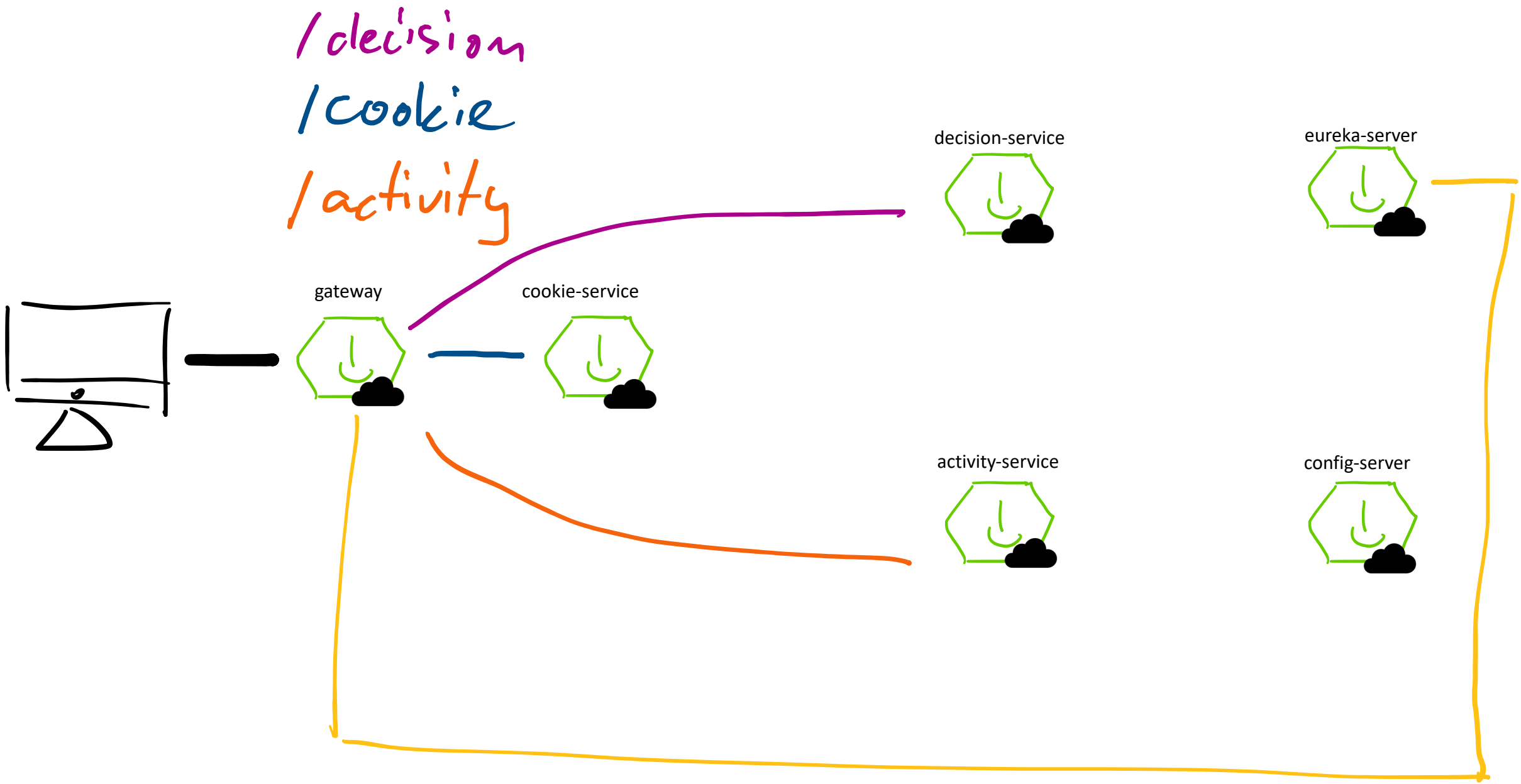
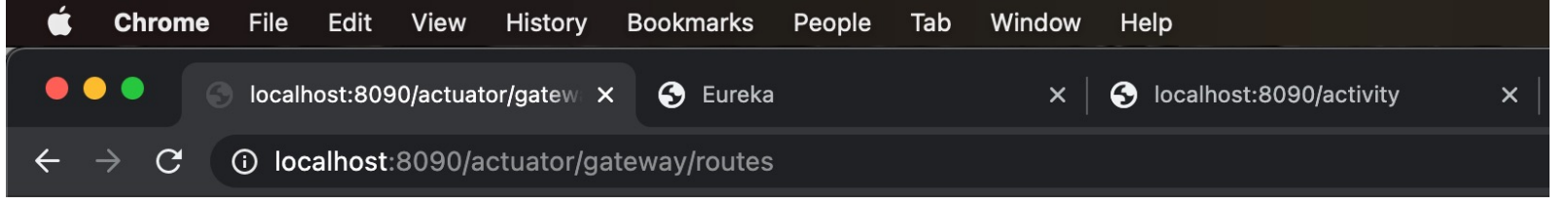
Spring Cloud Gateway

DoD

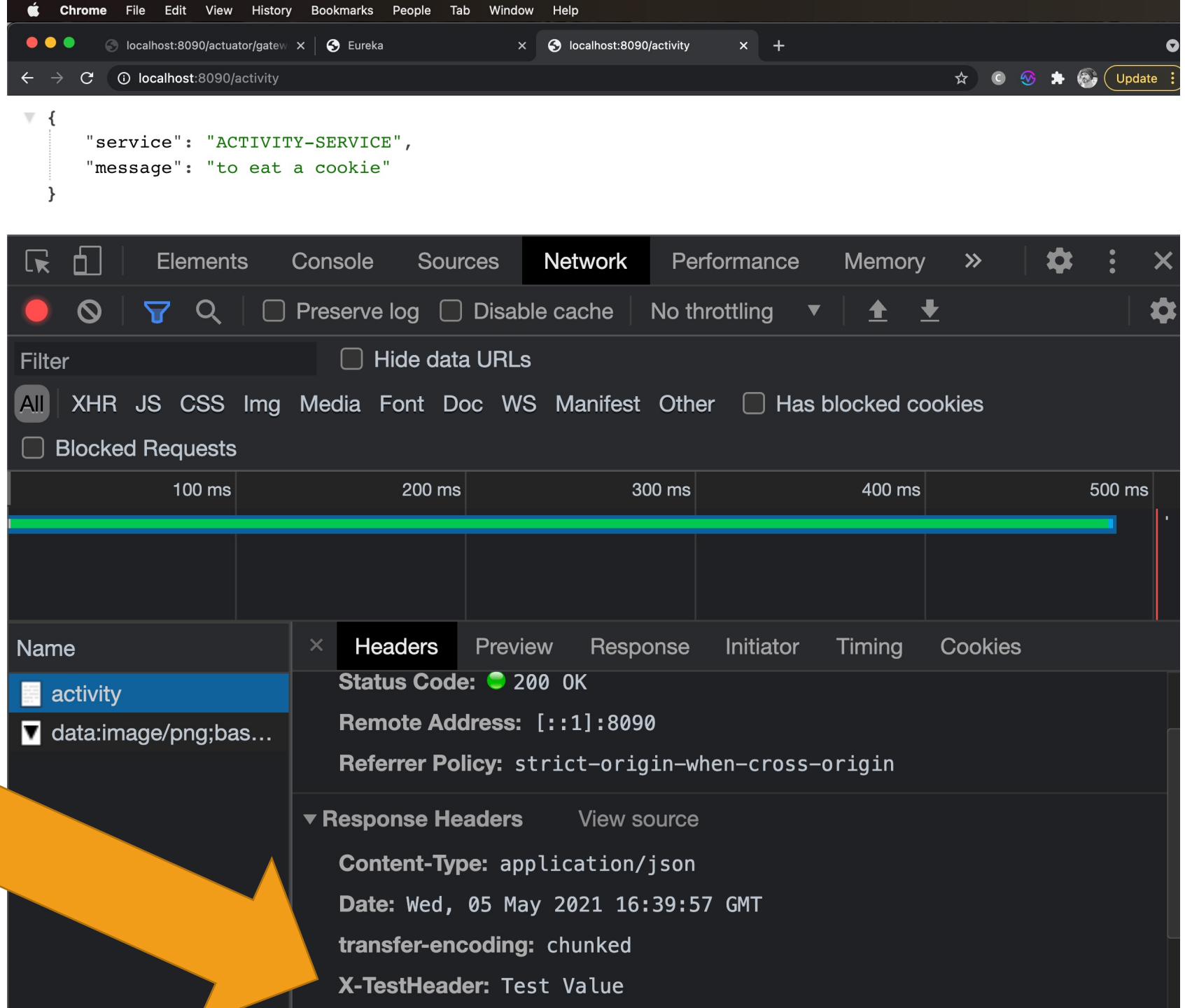

/decision
/cookie
/activity
decision-service
eureka-server
gateway
cookie-service
activity-service
config-server

Instances currently registered with Eureka

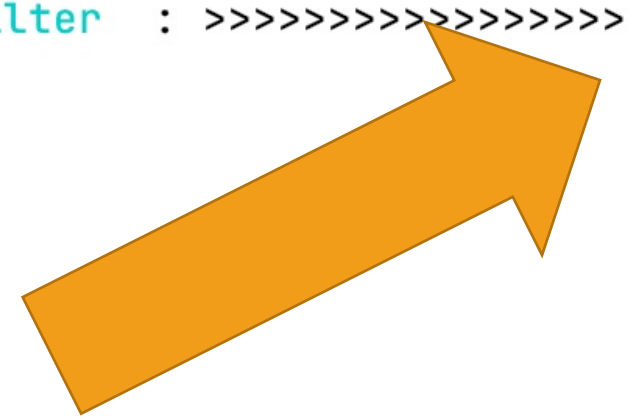
Application	AMIs	Availability Zones	Status
ACTIVITY-SERVICE	n/a (1)	(1)	UP (1) - 192.168.1.107:activity-service:8010
COOKIE-SERVICE	n/a (1)	(1)	UP (1) - 192.168.1.107:cookie-service
DECISION-SERVICE	n/a (1)	(1)	UP (1) - 192.168.1.107:decision-service:8000
GATEWAY	n/a (1)	(1)	UP (1) - 192.168.1.107:gateway:8090




```
[
  {
    "predicate": "Paths: [/fortune], match trailing slash: true",
    "route_id": "cookie_route",
    "filters": [],
    "uri": "lb://COOKIE-SERVICE",
    "order": 0
  },
  {
    "predicate": "Paths: [/activity], match trailing slash: true",
    "route_id": "activity_route",
    "filters": [
      "[[AddResponseHeader X-TestHeader = 'Test Value'], order = 0]"
    ],
    "uri": "http://activity-service:8010",
    "order": 0
  },
  {
    "predicate": "Paths: [/decision], match trailing slash: true",
    "route_id": "decision_route",
    "filters": [
      "[ExampleFilter, order = 0]"
    ],
    "uri": "http://decision-service:8000",
    "order": 0
  }
]
```


Chrome File Edit View History Bookmarks People Tab Window Help
localhost:8090/actuator/gatew
Eureka
localhost:8090/activity
localhost:8090/activity
Update
{
"service": "ACTIVITY-SERVICE",
"message": "to eat a cookie"
}
Elements Console Sources Network Performance Memory
Preserve log Disable cache No throttling
Filter Hide data URLs
All XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies
Blocked Requests
100 ms 200 ms 300 ms 400 ms 500 ms
Name
activity
data:image/png;bas...
Headers Preview Response Initiator Timing Cookies
Status Code: 200 OK
Remote Address: [::1]:8090
Referrer Policy: strict-origin-when-cross-origin
Response Headers View source
Content-Type: application/json
Date: Wed, 05 May 2021 16:39:57 GMT
transfer-encoding: chunked
X-TestHeader: Test Value

```
o.s.c.g.h.RoutePredicateHandlerMapping    : Route matched: decision_route
o.s.c.g.h.RoutePredicateHandlerMapping    : Mapping [Exchange: GET http://localhost:8090/decision] to
o.s.c.g.h.RoutePredicateHandlerMapping    : [bcb18492-1] Mapped to org.springframework.cloud.gateway.
o.s.c.g.handler.FilteringWebHandler       : Sorted gatewayFilterFactories: [[GatewayFilterAdapter{del
c.example.gateway.filters.ExampleFilter   : >>>>>>>>>>>>>>>>>> Custom filter here! <<<<<<<<<<<<<<<<<<
```



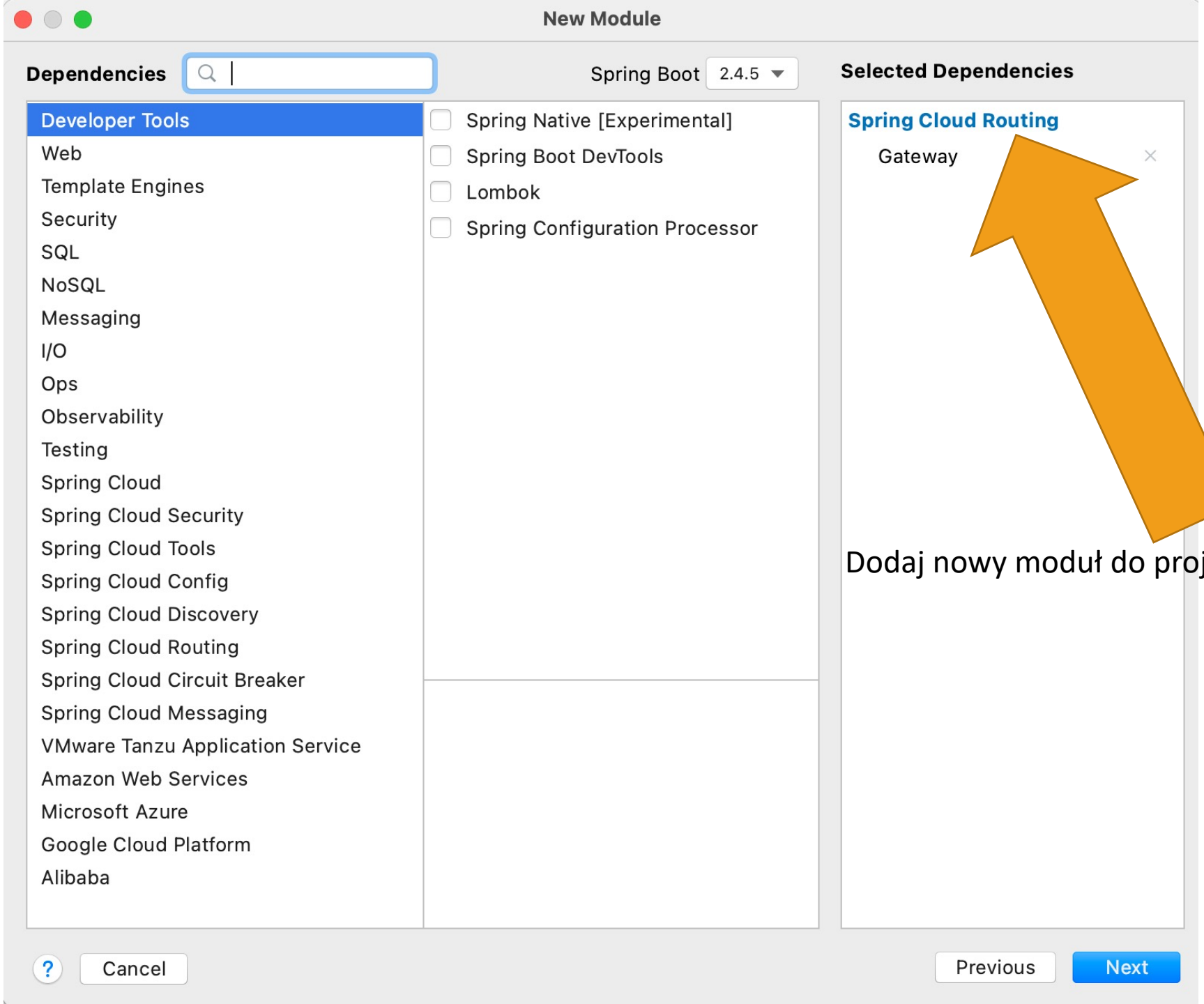
Przepis:

- dodanie serwisów do host
- dodanie modułu Spring Boot do projektu (zależność Spring Cloud Gateway), w nim:
    - dodanie zależności w pom.xml (klient eureki, actuator)
    - konfiguracja w application.yml (klient eureki, actuator)
    - implementacja ExampleFilter
    - konfiguracja przekierowań / filtrów (@Configuration)

Hosts

```
#START  cookie-service
127.0.0.1          eureka-peer1
127.0.0.1          eureka-peer2
127.0.0.1          config-server
127.0.0.1          cookie-service
127.0.0.1          activity-service
127.0.0.1          decision-service
#END fortune-cookie
```

Dodaj moduł do projektu



Dodaj nowy moduł do projektu (Spring Boot)

Dodatkowe zależności

```
<properties>
    <java.version>11</java.version>
    <spring-cloud.version>2020.0.2</spring-cloud.version>
</properties>
<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-gateway</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-dependencies</artifactId>
            <version>${spring-cloud.version}</version>
```

```
    <java.version>11</java.version>
    <spring-cloud.version>2020.0.2</spring-cloud.version>
</properties>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-actuator</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-gateway</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
```

application.yml

application.yml

Side-by-side viewer | Do not ignore | Highlight words

a7ea90f78fab01d1b85aded047bd49d5c60b49cd

```yaml
server:
  port: 8090
eureka:
  client:
    serviceUrl:
# eureka-peer1 needs to be available ( host / port)
      defaultZone: http://eureka-peer1:8761/eureka
spring:
  application:
    name: gateway
# taken from one of the many spring.io examples
logging:
  level:
    org:
      springframework:
        cloud:
          gateway: TRACE
management:
  endpoints:
    web:
      exposure:
        include: '*'
```

Własny Filtr

ExampleFilter.java

Side-by-side viewer | Do not ignore | Highlight words

a7ea90f78fab01d1b85aded047bd49d5c60b49cd

```java
package com.example.gateway.filters;

import org.springframework.cloud.gateway.filter.GatewayFilter;
import org.springframework.cloud.gateway.filter.GatewayFilterChain;
import org.springframework.web.server.ServerWebExchange;
import reactor.core.publisher.Mono;

import java.util.logging.Logger;

public class ExampleFilter implements GatewayFilter{

    Logger LOGGER = Logger.getLogger("com.example.gateway.filters.ExampleFilter");
    @Override
    public Mono<Void> filter(ServerWebExchange exchange, GatewayFilterChain chain) {
        LOGGER.info(">>>>>>>>>>>>>>>>>> Custom global filter here! <<<<<<<<<<<<<<<<<");
        return chain.filter(exchange);
    }

    @Override
    public String toString() {
        return "ExampleFilter";
    }
}
```

Konfiguracja (Java) przekierowań i filtrów

RoutesConfig.java (/Users/uki

Side-by-side viewer | Do not ignore | Highlight words

a7ea90f78fab01d1b85aded047bd49d5c60b49cd

```java
package com.example.gateway;

import com.example.gateway.filters.ExampleFilter;
import org.springframework.cloud.gateway.route.RouteLocator;
import org.springframework.cloud.gateway.route.builder.RouteLocatorBuilder;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;

@Configuration
public class RoutesConfig {

//   Check http://localhost:8090/actuator/gateway/routes
    @Bean
    public RouteLocator customRouteLocator(RouteLocatorBuilder builder) {
        return builder.routes()
/*
                lb://SERVICE-NAME : data taken form Eureka (only for this route):
                spring-cloud-starter-netflix-eureka-client dependency in pom.xml
                eureka.client.serviceUrl.defaultZone in application.yml
*/
                .route("cookie_route", r -> r.path("/fortune")
                        .uri("lb://COOKIE-SERVICE"))
                // an example of a predefined filter
                .route("activity_route", r -> r.path("/activity")
                        .filters(gatewayFilterSpec -> gatewayFilterSpec.addResponseHeader("X-TestHeader","Test Value"))
                        // the host / port as defined in hosts / app's profile
                        .uri("http://activity-service:8010"))
                .route("decision_route", r -> r.path("/decision")
                // an example of a custom filter
                        .filters(gatewayFilterSpec -> gatewayFilterSpec.filter(exampleFilter()))
                        .uri("http://decision-service:8000"))
                .build();
    }

    @Bean
    ExampleFilter exampleFilter() {
        return new ExampleFilter();
    }

}
```