## 13. What is the need of Zuul gateway in micro services?

The volume and diversity of Netflix API traffic sometimes results in production issues arising quickly and without warning. Netflix need a system that allows us to rapidly change behavior in order to react to these situations. So they implemented zuul gateway to incorporate these requirements.  
Zuul uses a range of different types of filters that enables us to quickly and nimbly apply functionality to our edge service. These filters help us perform the following functions:

* **Authentication and Security** – identifying authentication requirements for each resource and rejecting requests that do not satisfy them.
* **Insights and Monitoring** – tracking meaningful data and statistics at the edge in order to give us an accurate view of production.
* **Dynamic Routing** – dynamically routing requests to different backend clusters as needed.
* **Stress Testing** – gradually increasing the traffic to a cluster in order to gauge performance.
* **Load Shedding** – allocating capacity for each type of request and dropping requests that go over the limit.
* **Static Response handling** – building some responses directly at the edge instead of forwarding them to an internal cluster
* **Multiregion Resiliency** – routing requests across AWS regions in order to diversify our ELB usage and move our edge closer to our members

**What is Zuul?**

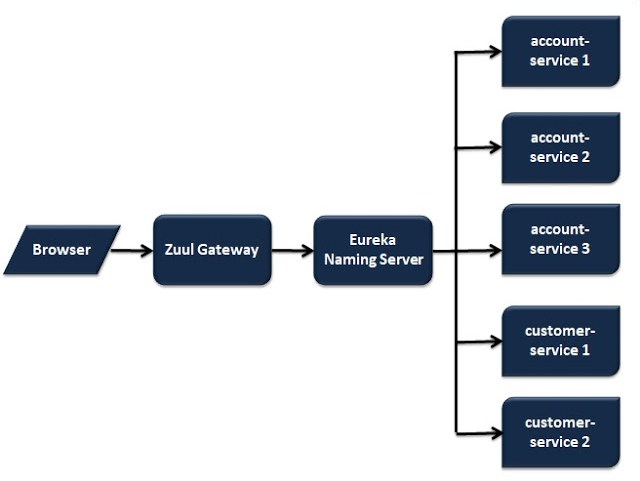
Zuul is the front door for all requests from devices and web sites to the backend of the Netflix streaming application. As an edge service application, Zuul is built to enable dynamic routing, monitoring, resiliency and security. It also has the ability to route requests to multiple Amazon Auto Scaling Groups as appropriate.

**What is the need of Zuul Gateway?**

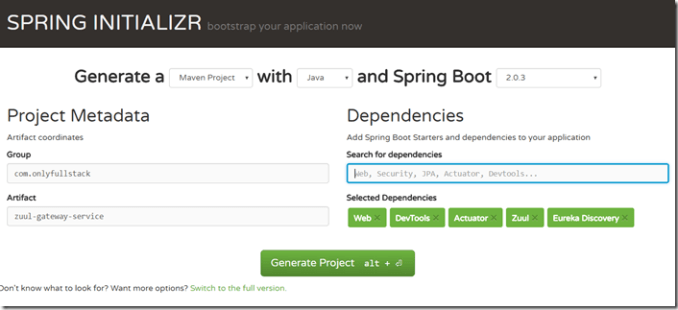
The volume and diversity of Netflix API traffic sometimes results in production issues arising quickly and without warning. Netflix need a system that allows us to rapidly change behavior in order to react to these situations. So they implemented zuul gateway to incorporate these requirements.  
Zuul uses a range of different types of filters that enables us to quickly and nimbly apply functionality to our edge service. These filters help us perform the following functions:

* **Authentication and Security** – identifying authentication requirements for each resource and rejecting requests that do not satisfy them.
* **Insights and Monitoring** – tracking meaningful data and statistics at the edge in order to give us an accurate view of production.
* **Dynamic Routing** – dynamically routing requests to different backend clusters as needed.
* **Stress Testing** – gradually increasing the traffic to a cluster in order to gauge performance.
* **Load Shedding** – allocating capacity for each type of request and dropping requests that go over the limit.
* **Static Response handling** – building some responses directly at the edge instead of forwarding them to an internal cluster
* **Multiregion Resiliency** – routing requests across AWS regions in order to diversify our ELB usage and move our edge closer to our members

Lets see how our components will connect with each other after adding the Zuul gateway.

**[](https://i2.wp.com/4.bp.blogspot.com/-740ArDDsfiY/W7BXPB5mGII/AAAAAAAAAH8/Wg8BwafJankCSwQdir1XQ35ybifHGOuhACLcBGAs/s1600/ZuulGateway.jpg?ssl=1)**

Lets implement the Zuul Gateway in our application.

1. Create new project names as zuul-gateway-service  
**[](https://lh3.googleusercontent.com/-LHvK42BCeLc/W6-H_qYIN0I/AAAAAAAAAGc/Kqk_jqmTW-s3ThdLQRX3Only3wwWpWakQCHMYCw/s1600-h/Part%2B51%2B-%2BCapture%255B3%255D)**

2. Update the file and add below annotations **@EnableZuulProxy** &**@EnableDiscoveryClient**

**package** *com.onlyfullstack.zuulgatewayservice*;

**import** *org.springframework.boot.SpringApplication*;

**import** *org.springframework.boot.autoconfigure.SpringBootApplication*;

**import** *org.springframework.cloud.client.discovery.EnableDiscoveryClient*;

**import** *org.springframework.cloud.netflix.zuul.EnableZuulProxy*;

@EnableZuulProxy

@EnableDiscoveryClient

@SpringBootApplication

**public** **class** ZuulGatewayServiceApplication {

**public** **static** **void** main(**String**[] args) {

SpringApplication.run(ZuulGatewayServiceApplication.class, args);

}

}

3. Update application.properties file as below

server.port=8765

spring.application.name=zuul-gateway-service

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

logging.level.root= DEBUG

logging.pattern.console= %d{yyyy-MMM-dd HH:mm:ss.SSS} %-5level [%thread] %logger{15} - %msg%n

4. Create a zuul filter which will get all the requests before going to the actual service

**package** *com.onlyfullstack.zuulgatewayservice.component*;

**import** *org.slf4j.Logger*;

**import** *org.slf4j.LoggerFactory*;

**import** *org.springframework.stereotype.Component*;

**import** *com.netflix.zuul.ZuulFilter*;

**import** *com.netflix.zuul.exception.ZuulException*;

@Component

**public** **class** ZuulLoggingFilter **extends** ZuulFilter {

Logger logger = LoggerFactory.getLogger(ZuulLoggingFilter.class);

@Override

**public** Object run() **throws** ZuulException {

logger.info("\*\*\*\*\*\*\*\*\*\*\*\*\*##############\*\*\*\*\*\*\*\* printing logs : ");

**return** **null**;

}

@Override

**public** **boolean** shouldFilter() {

**return** **true**;

}

@Override

**public** **int** filterOrder() {

**return** 1;

}

@Override

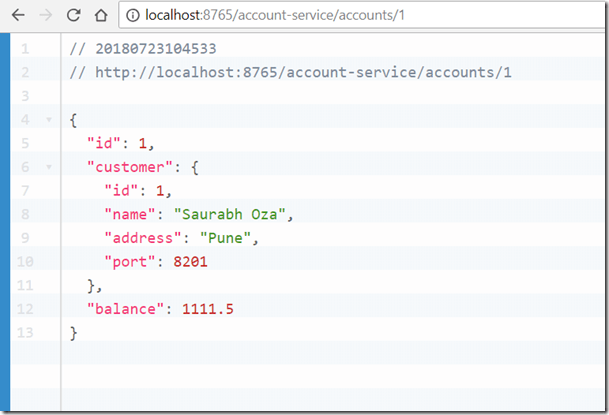
**public** **String** filterType() {

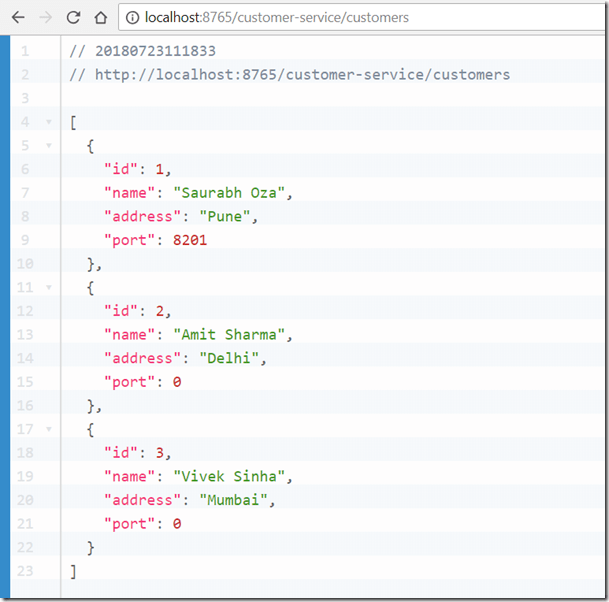
**return** "pre"; // 3 types are available - pre - to execute before processing the request, post- after processing the req and error

}

}

5. To call any service through zuul gateway, we need to form the url as below  
[**http://localhost:8765/{application-name}/{uri}**](http://localhost:8765/%7Bapplication-name%7D/%7Buri%7D)  
so the urls for our account-service and customer-service will be  
[**http://localhost:8765/account-service/accounts/1**](http://localhost:8765/account-service/accounts/1)  
[**http://localhost:8765/customer-service/customers**](http://localhost:8765/customer-service/customers)

6. If we hit the urls we will get the output as below  
**[](https://lh3.googleusercontent.com/-l8QHBwgYu94/W6-IC4z9ltI/AAAAAAAAAGk/PCjw7s8WPBg4HqJ0kIw0b-0mR5wOx1IHQCHMYCw/s1600-h/Part%2B52%2B-%2BCapture%255B5%255D)**

**[](https://lh3.googleusercontent.com/-7QV5RC0jrx8/W6-IFYZ4u9I/AAAAAAAAAGs/KoLsUGI07Rod7dCckCsx9FbKE5gVbqAvwCHMYCw/s1600-h/Part%2B53%2B-%2BCapture%255B12%255D)**