wavelabs

ZUUL API GATEWAY

Tharun kumar Bairoju

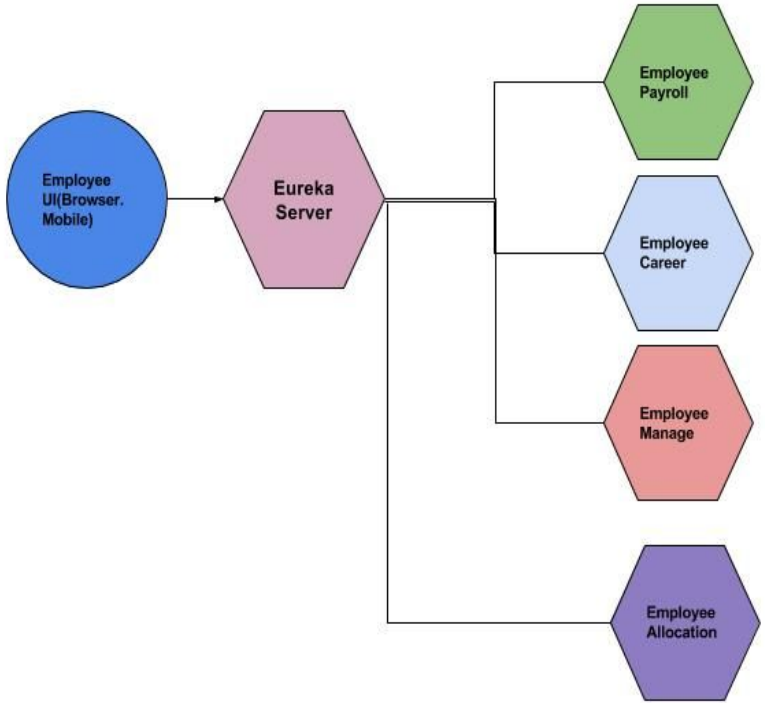
Zuul Existence:

The fact that your services are split into small microservices apps that shouldn’t be visible to users otherwise it may result in substantial development/maintenance efforts. Also there are scenarios when whole ecosystem network traffic may be passing through a single point which could impact the performance of the cluster.

Without Zuul:

-> From a UI developer perspective, to collect information from fifty underlying microservices, it has to call fifty REST APIs, as each microservice exposes a REST API for communication.

-> From backend developer Authentication, security, and monitoring in terms of this design- each microservice team has to develop all these aspects into its own service, so the same code has been replicated over fifty microservices

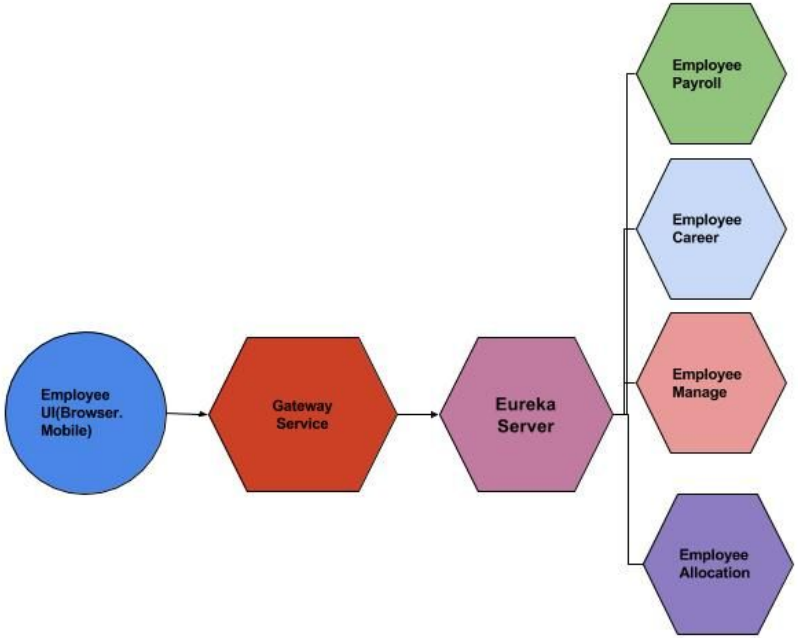


Without Zuul

wavelabs

With Zuul gateway:

Zuul acts as an API gateway or Edge service. It receives all the requests coming from the UI and then delegates the requests to internal microservices. So, we have to create a brand new microservice which is Zuul-enabled, and this service sits on top of all other microservices. It acts as an Edge service or client-facing service. Its Service API should be exposed to the client/UI. The client calls this service as a proxy for an internal microservice, then this service delegates the request to the appropriate service



Zuul Enabled Edge Service

Zuul routes configuration

Open application.properties and add below entries-

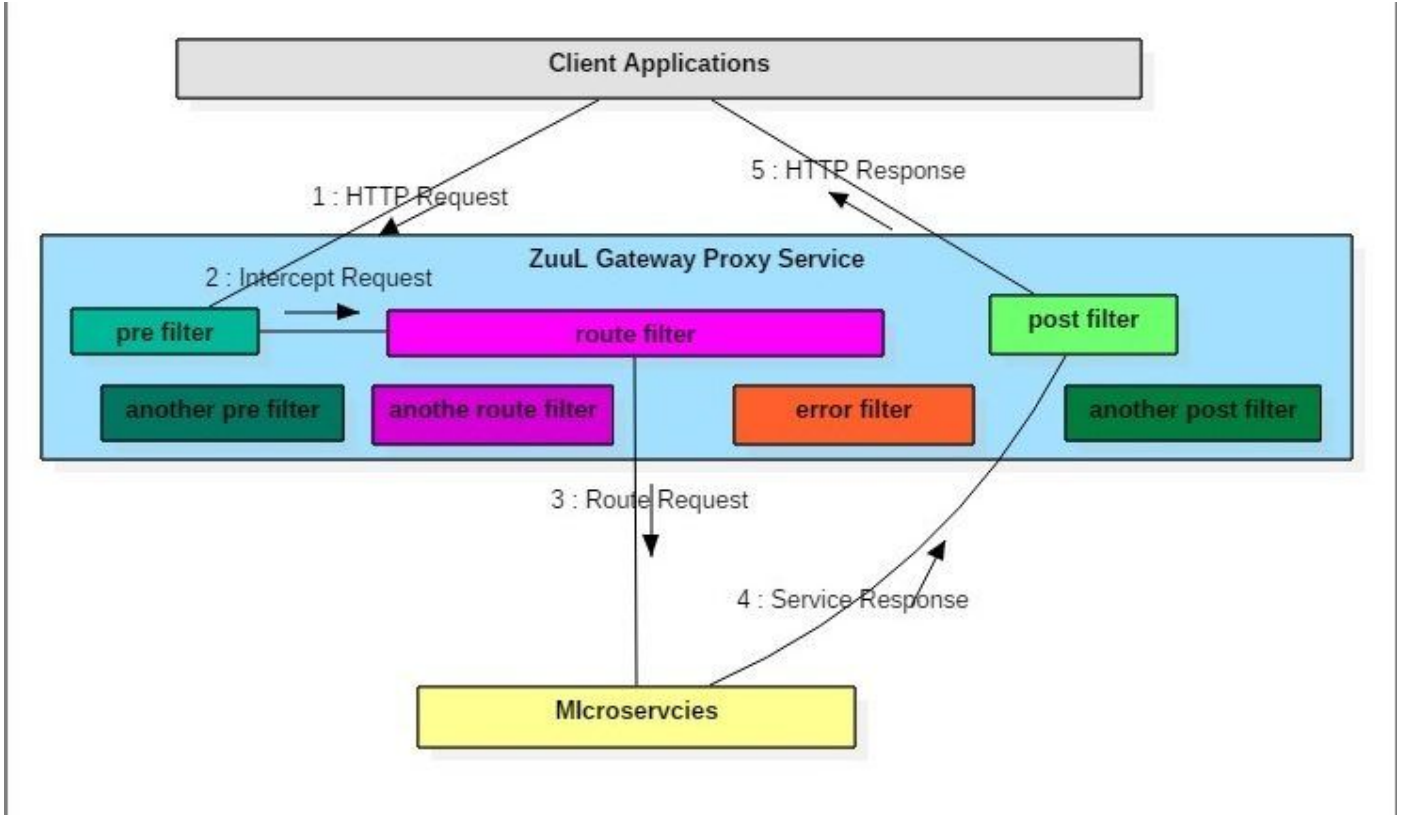
```
@EnableZuulProxy
```

Application.properties:

```
#Zuul routes. Here for /student path, we are routing
to localhost:8090 with extra path after that.

zuul.routes.student.url=http://localhost:8090

#Ribbon is auto integrated with Zuul and for this
exercise we are not using that.
ribbon.eureka.enabled=false

#Will start the gateway server @8080
server.port=8080
```

wavelabs

Zuul gateway processing:



wavelabs

Zuul Components

Zuul has mainly four types of filters that enable us to intercept the traffic in different timeline of the request processing for any particular transaction. We can add any number of filters for a particular url pattern.

- pre filters – are invoked before the request is routed.
- post filters – are invoked after the request has been routed.
- route filters – are used to route the request.
- error filters – are invoked when an error occurs while handling the request.

It’s time for demo

wavelabs

Thank you...

wavelabs