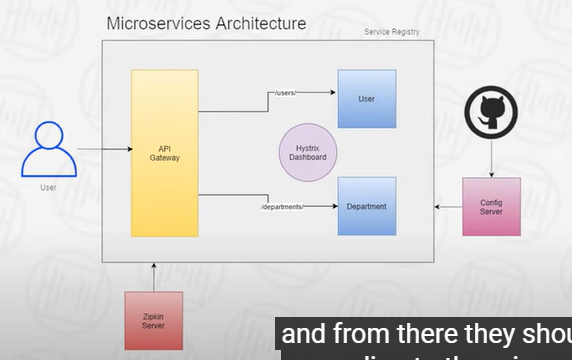
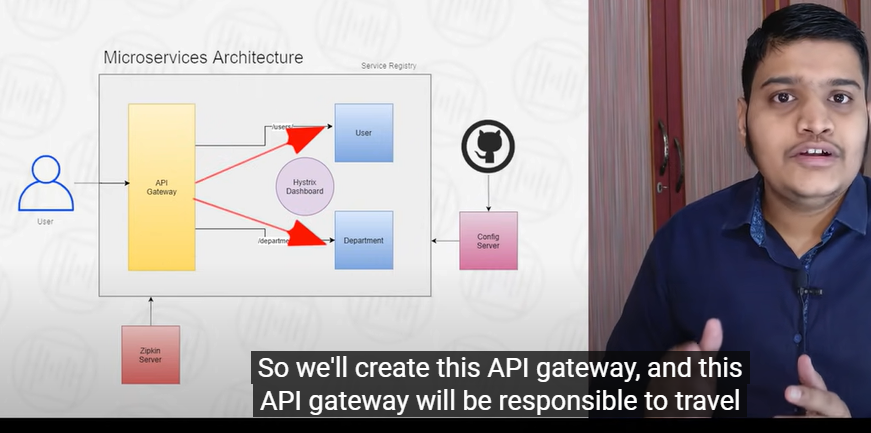
**MICROSERVICES**

****

They should be going from 1 Api gateway and from there they should be travel to the micro services URL.

This Api gatway will be responsible to travel.



Agar koe microservices fail hojayegi tw hum 1 faul back method banayengay jo dosri services ko batayegi k yai microservices fail hai.

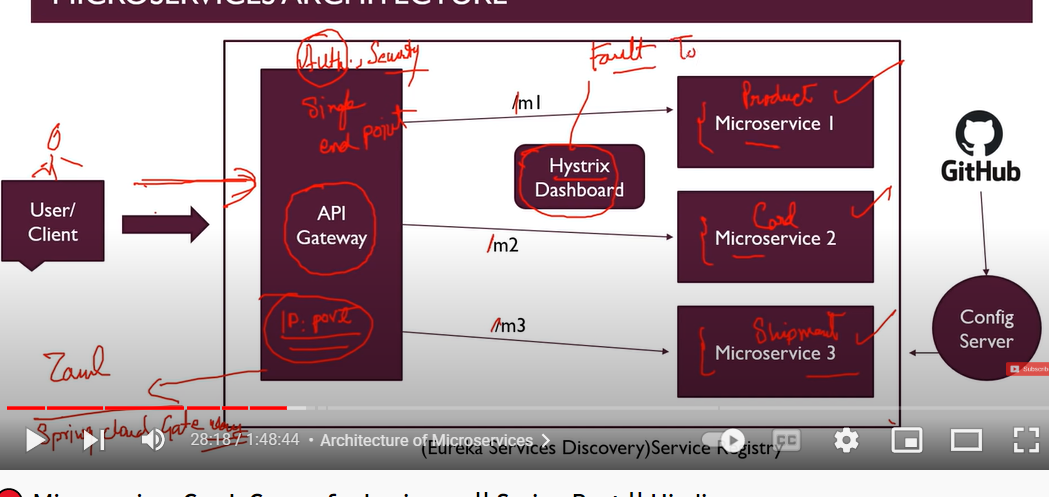
Now once we implement this History circuit braker, we should also be able to visualize how much my endpoints are failing and how much getting succesfull and also which of microservices is not working. So this we will implement the historic dashboard. With the help of this we will notify which of our service is not working.

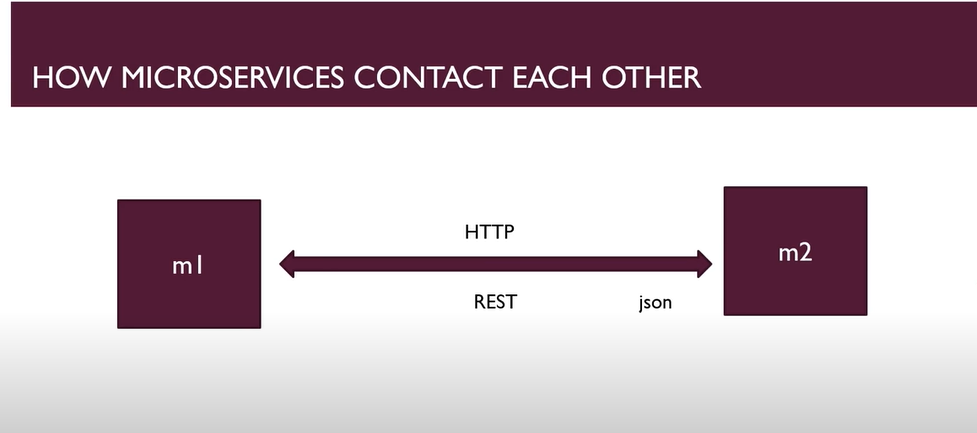
Now we will implement a config server

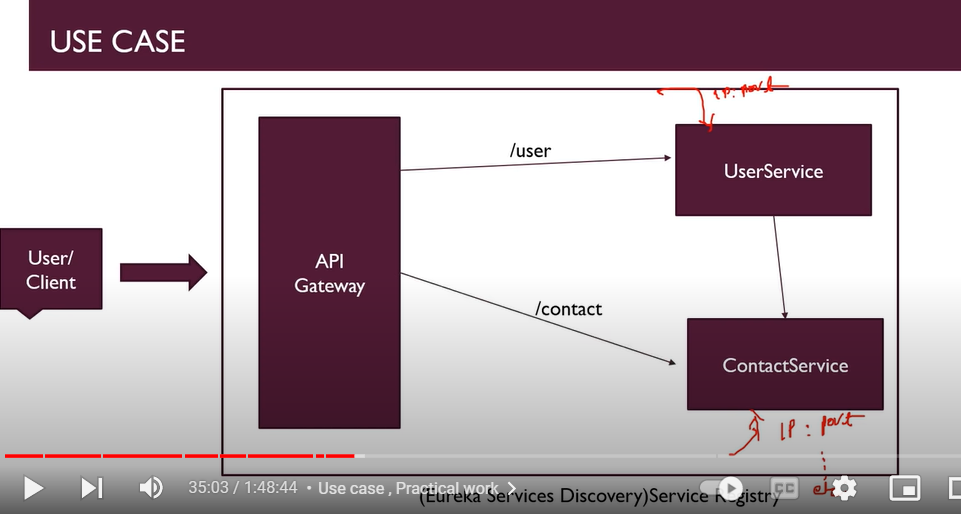
Configuration saari git par save kar k config server ko implement karengay phr wo har services ko configuration degi saari. So just in one place it will be available for all the microservices.

After that we will add the distributed login so if there are multiple microservices, we should be able to identify which microservices is failing and where our request is traversing.So for that we will use zepkin/sluth libraries. We will create a zipkin server and then we will implement the zipkin client and sleuth libraries in all microservices. So from there we will able to implement a distributed login.

Now this sluth provide multiple options over there we will be able to identify which of the services been called. What is the trace id. What is the span id the service will contain from which service this log has been traced. TraceId will be a unique id along the entire request. So if you are requesting to get a user details. So our call is like to get a user details alongside to get a department detail from department service. So that request will contain two service ideas like user service, Department service for both of them trace Id would be unique that is a single one and both of them there will be different span Id

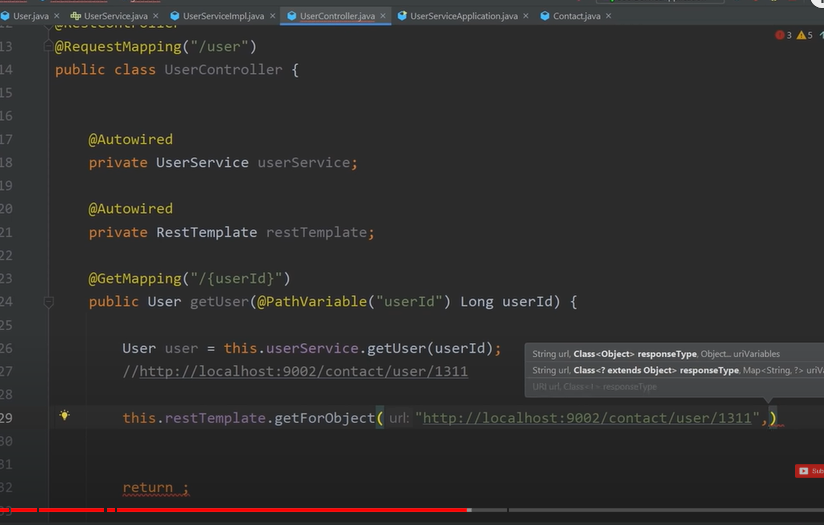






We can use RestTemplate to communicate two different services.

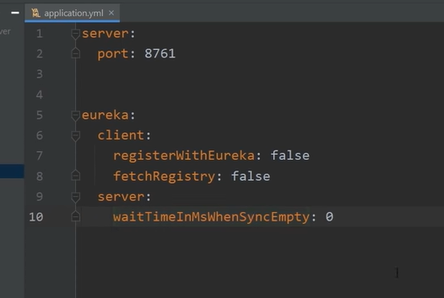
1 bean banayengay RestTemplate ka phr usko kahi bhi autowired kardengay.



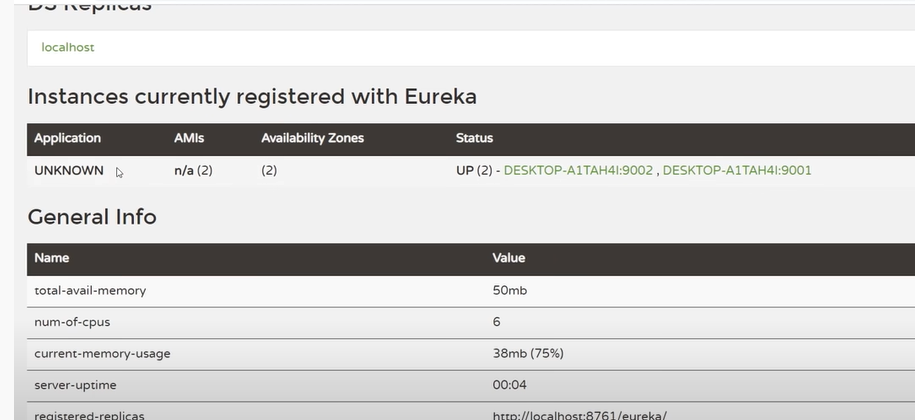
New project banayengay eureka server k liye q k wo khud 1 microservices hai.

Phr main class mai @EnableEurekaServer kardengay

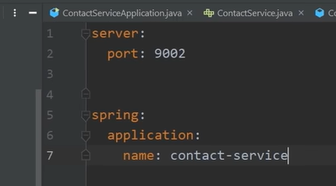
Khud sai na register hona yai batana parega

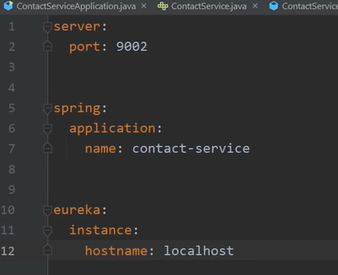


Ab humay dosri services ko register karwanay k liye unhay client banana paray ga.



Services ko name dainay k liye simple hum application properties mai jakar day dengay





Now we will create a project for API GATEWAY because it is also a microservice.

Phr main class mai @EurekaClient lagadengay

