**API Gateway :**

In microservices architectures, there would be hundreds of microservices and these microservices have a lot of common features. Like Authentication, authorization, and logging.

Where do you implement all these common features?

The solution is to go for an API gateway.

**In the older versions of Spring Cloud, the popular API gateway to use was Zool.**

**Since Zool is no longer supported by Netflix,**

Spring Cloud has moved on and now the recommended option as an API gateway

is Spring Cloud Gateway.

1. Add Depedency(Reactive api gateway)

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-gateway</artifactId>  
</dependency>

1. Add Dependency (This is required to register the gateway with eureka)

<dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  
</dependency>

1. Set application.properties

**spring.cloud.gateway.discovery.locator.enabled=true**  
(When enabled, the gateway will use the configured service registry (e.g., Eureka) to discover services and dynamically route requests to them based on their registered service IDs)

(we would want the API gateway to talk to Eureka with that name, find the server location**)**

AND  
**spring.cloud.gateway.discovery.locator.lower-case-service-id=true**(When set to true, service IDs are converted to lowercase before being used in route definitions. This can be useful for case-insensitive matching)

Now with the help of new API gateway URL we can access the microservice and also we can perform common feature before envoking

[localhost:8765/car-service/car\_service/ford](http://localhost:8765/car-service/car_service/ford)

1. car-service is in lower case because of above properties else by default it is in uppercase

ie. [localhost:8765/CAR-SERVICE/car\_service/ford](http://localhost:8765/car-service/car_service/ford)

1. car-service this is the name of microservice service with eureka and same can be found in eurake dashboard.

**To create custom router**

In this step we are redirect from gateway using naming server

*@Bean*

*public RouteLocator customRoute(RouteLocatorBuilder builder){*

*Function<PredicateSpec, Buildable<Route>> routeFunction =  
 p -> p.path("/dealer/\*\*")  
 .uri("lb://dealer-service");*

*return builder.routes()  
 .route(routeFunction)  
 .build();*

*}*

So here path() will be endpoint url so with eureka url if have to provide url of the dealer application

And uri is im getting naming server to find dealer-service naming server mention inside .uri() and try to find the path mention

