**NEED OF CUSTOM ROUTING : how to build custom routes ?**

Solution : creating a configuration file

@Configuration

public class ApiGatewayConfiguration {

@Bean

public RouteLocator gatewayRouter(RouteLocatorBuilder builder){

return builder.routes().build();

}

This is when you don’t want any customisation of urls and routes

}

**To customise further will be the changes**

*public* RouteLocator gatewayRouter(RouteLocatorBuilder builder){  
 Function<PredicateSpec, Buildable<Route>> routeFunction =  
 p->p.path("/get")  
 .uri("http://httpbin.org:80");  
 *return* builder.routes()  
 .route(routeFunction).build();  
}

**below code shows how to add headers and requestparameters with custom url for authentication purpose:**

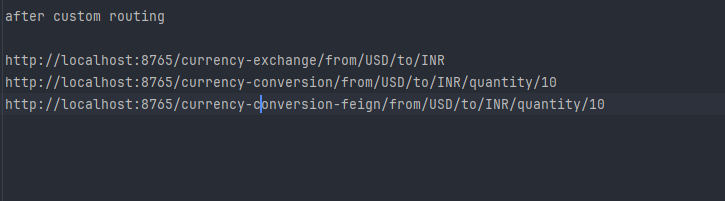
*}  
 @Bean  
 public* RouteLocator gatewayRouter(RouteLocatorBuilder builder){  
 Function<PredicateSpec, Buildable<Route>> routeFunction =  
 p->p.path("/get")  
 .filters(f->f.addRequestHeader("MyHeader","MyUrI")  
 .addRequestParameter("Param","Myvalue"))  
 .uri("http://httpbin.org:80");  
 *return* builder.routes()  
 .route(routeFunction).build();  
 }  
  
}

**After making lambda function inline it looks like this**

*@Bean  
public* RouteLocator gatewayRouter(RouteLocatorBuilder builder){  
 *return* builder.routes()  
 .route(p->p.path("/get")  
 .filters(f->f.addRequestHeader("MyHeader","MyUrI")  
 .addRequestParameter("Param","Myvalue"))  
 .uri("http://httpbin.org:80")).build();  
}

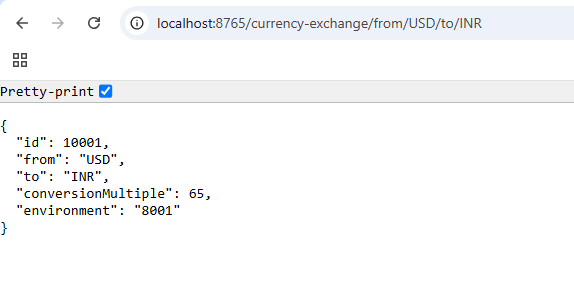
**Adding custom routings of all 2 apis :**

*@Bean  
public* RouteLocator gatewayRouter(RouteLocatorBuilder builder){  
 *return* builder.routes()  
 .route(p->p.path("/get")  
 .filters(f->f.addRequestHeader("MyHeader","MyUrI")  
 .addRequestParameter("Param","Myvalue"))  
 .uri("http://httpbin.org:80"))  
 .route(p->p.path("/currency-exchange/\*\*")  
 .uri("lb://currency-exchange-service"))  
 .route(p->p.path("/currency-conversion/\*\*")  
 .uri("lb://currency-conversion-service"))  
 .route(p->p.path("/currency-conversion-feign/\*\*")  
 .uri("lb://currency-conversion-service"))  
 .build();  
}

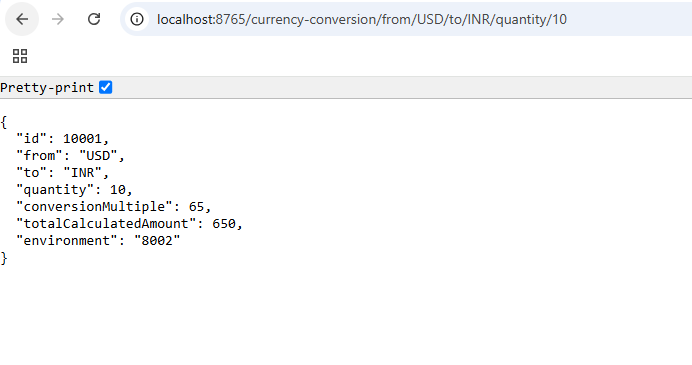


**Output**

**Exchange service**



**Conversion Service – by Resttemplate exchange service call**



**Conversion Service – by feignclient exchange service call to implement load balancing**

