**Credit Card Application system:**

**Applcation Environment :** Spring Boot, Eureka Server, Rest API, Spring Security, Fault Toleance (Hystrix), Feign Client

**Micro Service Names:** customer-service, credit-card-service.

**Customer Service:**

* Configured in port : **8011**
* Application Name : **customer-service**
* Configured as **Discovery Client** and **c**onfigured to connect to eureka-server in port **8761**
* Enabled **FeignClients** so that Feign clients can be used in the application
* Configured Circuit Breaker **Hystrix** (Fault Tolerance) with fallback method
* Configured **Spring Security** with Inmemory Authentication with one user (User name: **user**, password: **password**, Role: **USER**) uses Basic Authentication
* Configure **Ribbon** Client Side Load Balancer
* Configured Global Error Controller
* Configured Global Exception Handler
* Configured **Feign Client** for Credit Card Service as proxy interface.
* Uses auth Token (**auth\_token**) to transfer from Customer Service to Credit Card Service while sending a call to Credit Card Service
* Implemented Logging

**Credit Card Service:**

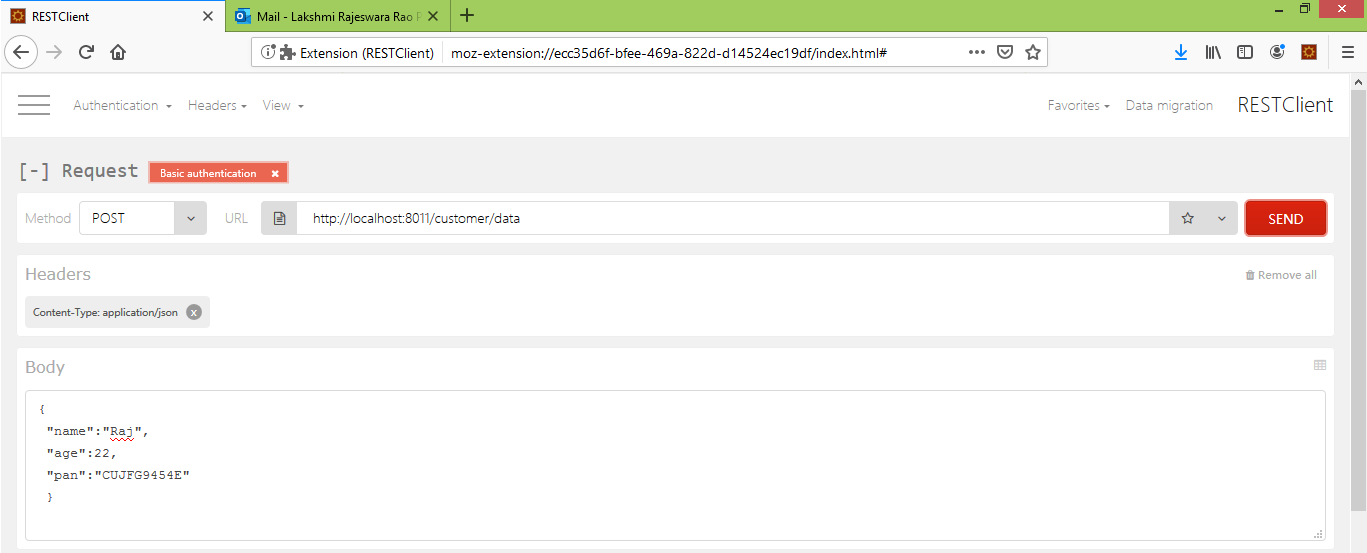
* Configured in port : **8021** (Also configured duplicate instance with port **8022** while testing the application)
* Application Name : **customer-service**
* Configured as **Discovery Client** and **c**onfigured to connect to eureka-server in port **8761**
* Uses auth Token (**auth\_token**) to authenticate the call from from Customer Service and not directly
* Configured Global Error Controller
* Configured Global Exception Handler
* Hard coded Credit Card data in Repository (Used a Credit Card Repository Class)
* Implemented Logging

**Eureka Service:**

* Configured in port : **8761**
* Application Name: **eureka-server**

**Flow:**

Using Postman, we will make a call with **Cusmer** **Object (name, age, pan)** to Customer Service(Micro Service-1) wih uri **http://localhost:8011/customer/data** by providing basic authentication details like username, password and selecting **post** method and by selecting **content type** as **application/json**.



{

"name":"Raj",

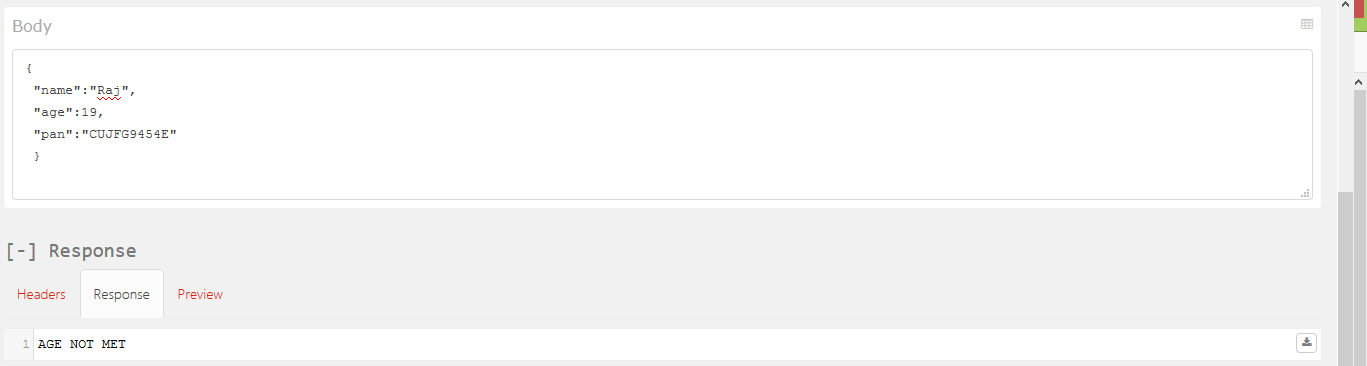
"age":25,

"pan":"CUJFG9454E

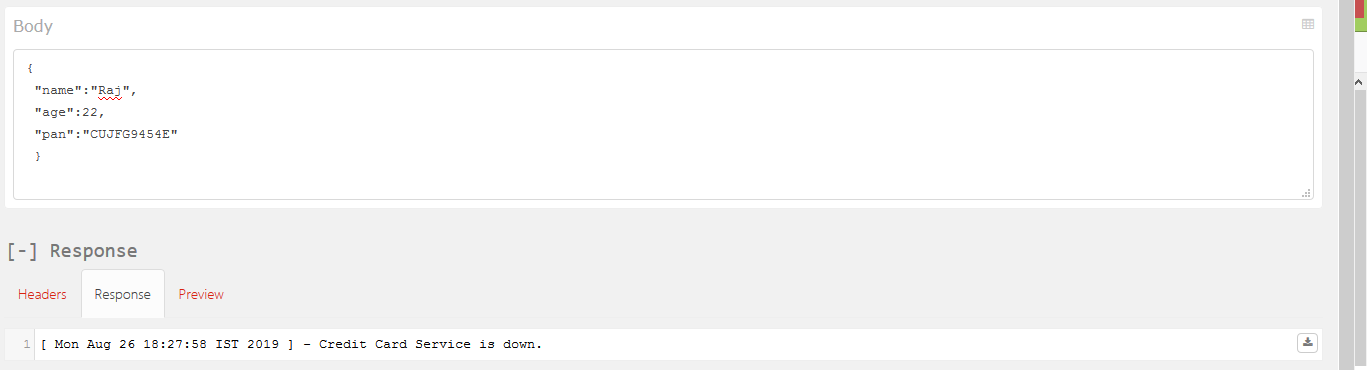
}

Sample Json:

Case1:) In Customer Service, it will check for the age criteria [20 > age < 60]. If the customer age is not met with the criteria, this microservice will reject it.



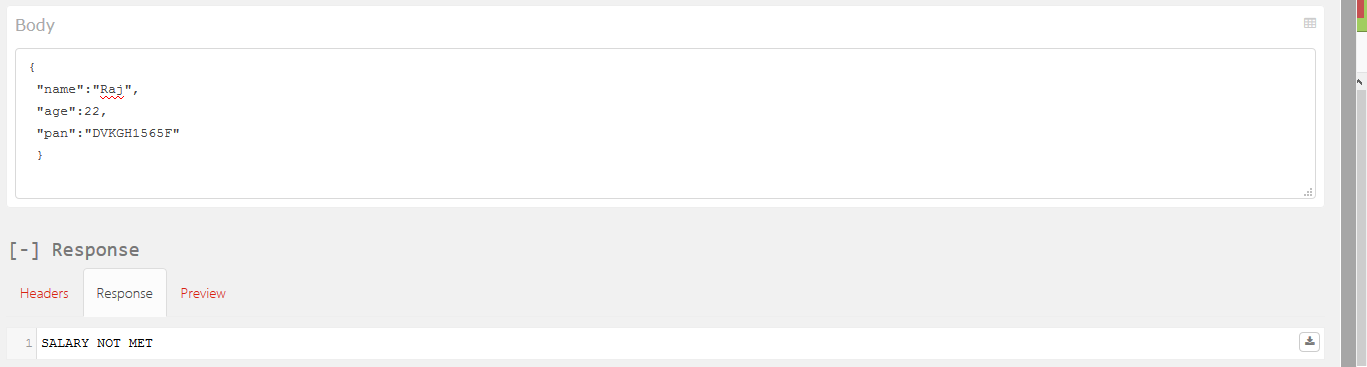
Case2:) In Customer Service, If the customer age is met with the criteria, this microservice will call Credit Card Microservice with **pan** and **customer salary** by calling its Fiegn Client Proxy. While calling, it will add the auth\_token in the **Header**. If the Credit Card Service is down, It will call the Fallback method and sends the response as follows



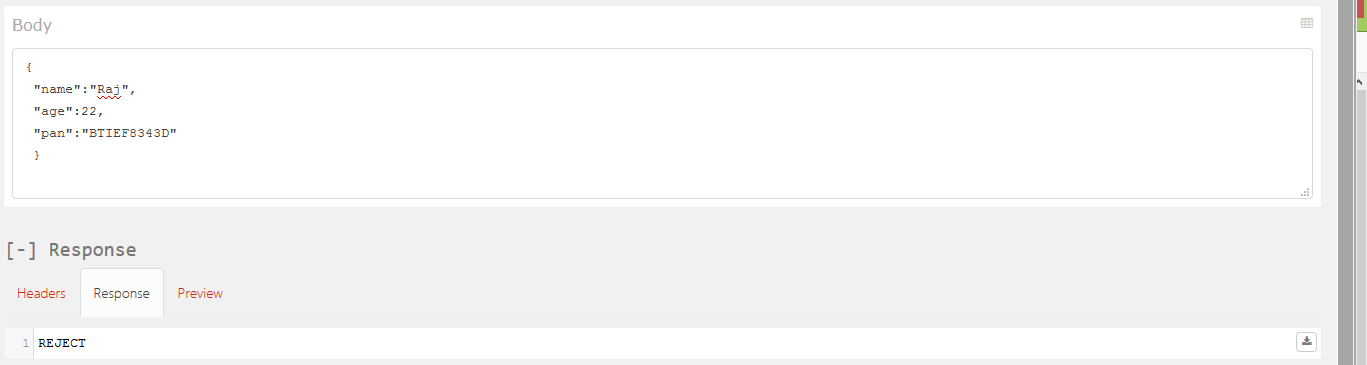
Case3:) While calling Credit Card Microservice with **pan** and **customer salary** by calling its Fiegn Client Proxy, if the Credit Card Service is Up and running, it will add the auth\_token in the **Header**. It will match the token sent in the header and with credit card service auth\_token.

Assuming PAN No. is available with Credit Card Service, and while sending the PAN No. and salary, It will check for the Minimum Salary Condition, If not met it will **REJECT**. it will check whether the PAN No. is existed in the Repository or not. If not availabe, then service returns as **SELECT**. If it is available and loans are not available then it will send **SELECT**. If Pan is available and loans are present then it eill check if EMI is < 10 % of his salary. If met then **SELECT** else **REJECT.**

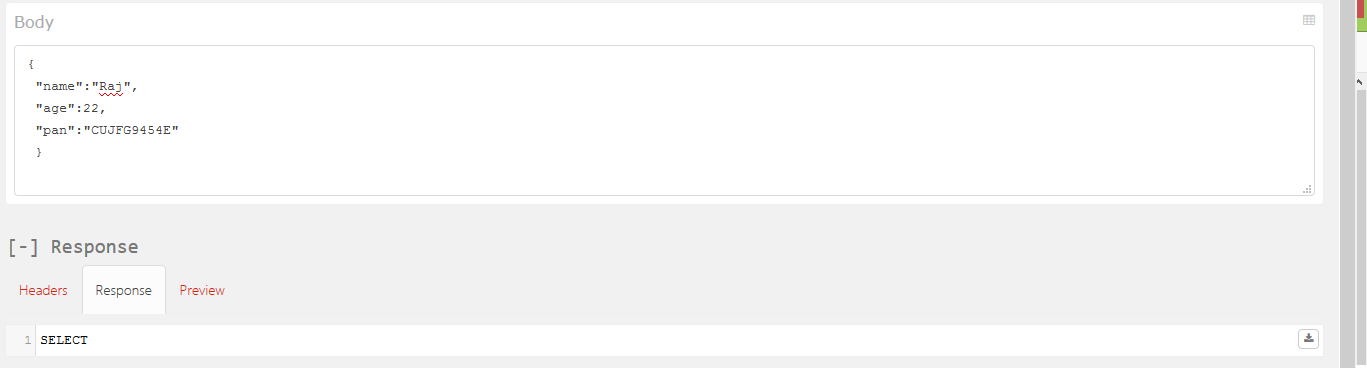
*Case4:)* ***If Salary Not met (< 50000)***



*Case5:)* ***If Salary Condition not met (10% of Salary)***



*Case6:)* ***If Salary Condition met (10% of Salary)***



Sample Data:

PAN: "ASHDE7232C", Loans : false, EMI: 0.0,

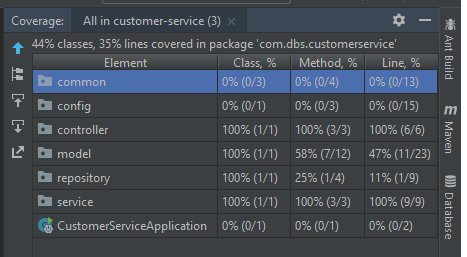
PAN: "BTIEF8343D", Loans : true, EMI: 12500,

PAN: "CUJFG9454E", Loans : true, EMI: 6900,

PAN: "DVKGH1565F", Loans : false, EMI: 0.0,

PAN: "EWLHI2676G", Loans : true, EMI: 9900

Customer Service Code Coverage



Credit Card Service Code Coverage

