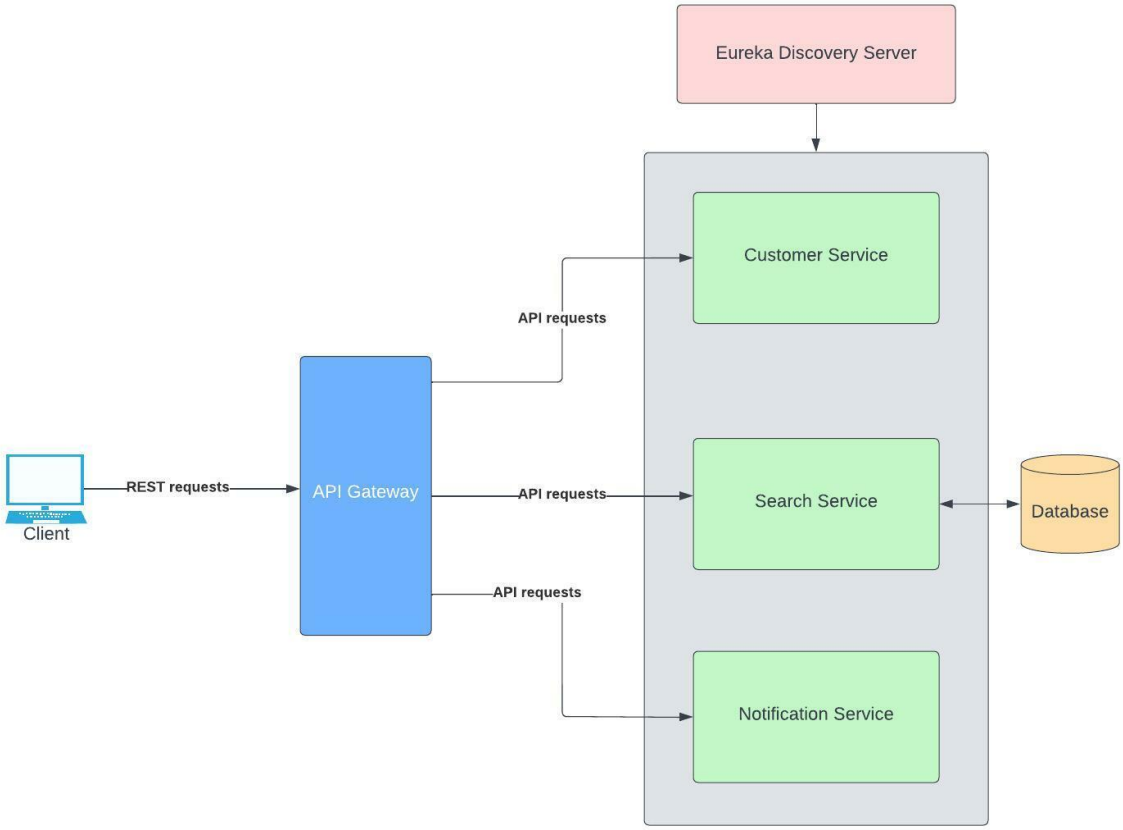
Nagarro NAGP Microservices



- Services

  - Customer Service : -

    Customer service will provide the details of the customer like customer id, name, location, mobile number and type of service customer wants from the service provider.

  - Search Service : -

    Search service will get the details of the customer through REST API and search for the service providers according to the customer's location and type of service customer wants from the service providers from the database. (Here database is the array list of the service providers which is coded in the program and database connectivity is not used).

Search service will return the list of all service providers and their details which are in the customer's location.

- Notification Service : -

  Notification service will send the notification to all the service providers which are given by search service so that the service providers can accept or deny the request of the customer. Then the notification service will return detailed list of those service providers who have accepted the request. (In this service a method is written in which a random integer is taken. If its value is 1 then we assume that the service provider has accepted the request and if the value is 0 then we assume that the service provider has denied the request. We are not sending the actual notification).

  Notification service will also send the customer details along with the notification to the service providers.

- API gateway : -

  Spring cloud gateway is used for all the incoming traffic and provides a single endpoint or URL for the client apps and then internally maps the requests to a group of internal microservices (In this, port 8000 is used for the requests).

- Eureka discovery server : -

  Eureka discovery server will hold the information about all service applications. Every microservice will register into the eureka server and eureka server knows all the client applications running on each port and IP address.

➢ Interservice Communication

WebClient is used for synchronous communication. In contrast to its precursor soon-to-be-deprecated RestTemplate, it offers modern non-blocking and reactive API. Under the hood, it's based on HttpClient introduced in Java 11. WebClient is an interface representing the main entry point for performing web requests. To work properly with the client, we need to know how to:

- create an instance
- make a request
- handle the response

```
@Bean
@LoadBalanced
public WebClient.Builder getWebClientBuilder() {
    return WebClient.builder();
}
```

```
@Autowired
private WebClient.Builder webClientBuilder;
```

```
Customer customer = webClientBuilder.build()
        .get()
        .uri("http://customer-service/customer/"+customerId)
        .retrieve()
        .bodyToMono(Customer.class)
        .block();
```

- Github link :- https://github.com/ParasGautam26/Microservices-NAGP
- DockerHub link :- https://hub.docker.com/r/parasgautam26/nagp_microservice/tags