**Tugas Microservice**

Nama : Dimas Erlangga

NIM : 2B

Prodi : TKOM

Matkul : Microservice Teori

**Untuk hari ini silahkan di kerjakan tugas berikut, hasilnya upload ke GIT**

1. **Jelaskan Tentang Server side Discovery**

**Server-Side Discovery** adalah pola dalam service discovery di mana proses menemukan layanan dilakukan di sisi server, bukan oleh klien. Dalam pola ini, klien tidak langsung berkomunikasi dengan layanan tujuan. Sebagai gantinya, klien mengirimkan permintaan ke komponen perantara seperti **Load Balancer**, **Service Router**, atau **API Gateway**.

**Cara Kerja:**

1. **Klien Mengirim Permintaan ke Endpoint Tetap:**

* Klien hanya perlu mengetahui endpoint tetap, misalnya URL atau alamat IP dari load balancer atau gateway.

1. **Service Registry:**

* Komponen server-side discovery (seperti load balancer) menggunakan informasi dari **Service Registry** untuk mengetahui lokasi (alamat IP dan port) dari layanan tujuan.

1. **Routing Permintaan:**

* Load balancer meneruskan permintaan ke salah satu instans layanan berdasarkan mekanisme seperti round-robin, weighted balancing, atau aturan tertentu.

1. **Pengembalian Respons:**

* Setelah layanan memproses permintaan, respons dikirim kembali ke klien melalui load balancer.

**Contoh Implementasi:**

* **NGINX** dengan Consul atau Eureka.
* **Envoy Proxy** sebagai service router.

1. **Jelaskan Tentang Cleint side Discovery**

**Client-Side Discovery** adalah pola di mana klien secara langsung bertanggung jawab untuk menemukan lokasi layanan yang ingin diakses. Klien mengambil informasi tentang layanan dari **Service Registry** dan berkomunikasi langsung dengan instans layanan tanpa melalui perantara seperti load balancer.

**Cara Kerja:**

1. **Service Registry:**

* Klien terhubung langsung ke Service Registry untuk mengambil daftar lokasi layanan aktif.

1. **Load Balancing di Klien:**

* Klien memiliki logika untuk memilih salah satu instans layanan berdasarkan algoritma seperti round-robin atau random.

1. **Komunikasi Langsung:**

* Klien mengirim permintaan langsung ke layanan tujuan yang dipilih.

**Contoh Implementasi:**

* **Netflix Ribbon** untuk load balancing dalam aplikasi berbasis Spring Cloud.
* Klien HTTP yang terintegrasi dengan Consul.

1. **Jelaskan kekurangan dan kelebihan dari no 1 dan 2 diatas**
2. **Server-Side Discovery**

**Kelebihan:**

* **Sederhana bagi klien:** Klien tidak perlu memahami logika service discovery atau load balancing.
* **Fleksibel:** Load balancer dapat diatur untuk menerapkan aturan routing kompleks.
* **Terpusat:** Semua logika routing dan load balancing dikendalikan dari satu tempat, sehingga mudah dikelola.

**Kekurangan:**

* **Titik kegagalan tunggal:** Jika load balancer atau gateway gagal, seluruh sistem terganggu (kecuali ada mekanisme redundansi).
* **Overhead tambahan:** Menambahkan latensi karena setiap permintaan harus melalui perantara.
* **Kompleksitas server:** Server memerlukan konfigurasi dan sumber daya tambahan.

1. **Client-Side Discovery**

**Kelebihan:**

* **Efisien:** Permintaan langsung dikirim ke layanan tujuan tanpa melalui perantara, mengurangi latensi.
* **Tidak ada titik kegagalan tunggal:** Klien memiliki kontrol penuh terhadap proses discovery.

**Kekurangan:**

* **Kompleksitas di klien:** Klien harus mengimplementasikan logika untuk mengambil data dari service registry dan melakukan load balancing.
* **Pembaharuan:** Jika terjadi perubahan pada layanan, semua klien harus diperbarui untuk mendukung logika baru.
* **Overhead pada klien:** Menambah beban pemrosesan pada klien.

1. **Jelaskan tentang Service Registry**

**Service Registry** adalah komponen utama dalam arsitektur service discovery. Registry ini berfungsi sebagai direktori yang menyimpan daftar layanan yang tersedia beserta detail lokasinya (seperti alamat IP dan port). Klien atau server-side discovery menggunakan informasi ini untuk menemukan layanan.

**Fungsi Utama:**

* **Pendaftaran Layanan (Service Registration):** Layanan secara aktif mendaftarkan dirinya ke registry saat layanan mulai berjalan. Informasi yang didaftarkan meliputi:
  + Nama layanan.
  + Alamat IP dan port.
  + Metadata tambahan (seperti zona atau versi).
* **Penghapusan Layanan (Service Deregistration):** Ketika layanan mati atau tidak aktif, registry menghapus layanan tersebut, baik melalui pemberitahuan langsung dari layanan atau secara otomatis dengan mendeteksi tidak adanya heartbeat.
* **Querying (Pencarian Layanan):** Komponen seperti klien atau load balancer melakukan query ke registry untuk mendapatkan daftar layanan aktif.

**Karakteristik:**

* **Konsistensi:** Informasi dalam service registry harus selalu diperbarui untuk mencerminkan keadaan layanan saat ini.
* **Replikasi:** Data pada service registry sering direplikasi ke beberapa node untuk memastikan ketersediaan tinggi.

**Contoh Implementasi:**

* **Consul:** Penyedia registry dengan fitur tambahan seperti konfigurasi kunci/nilai dan deteksi kesehatan.
* **Eureka:** Bagian dari ekosistem Spring Cloud.
* **Zookeeper:** Umumnya digunakan di lingkungan Hadoop atau Kafka.

**Keuntungan Service Registry:**

* Mempermudah koordinasi dan pengelolaan layanan dalam sistem mikroservices.
* Mendukung skalabilitas dan keandalan sistem.

**Kekurangan Service Registry:**

* Membutuhkan infrastruktur tambahan untuk menjalankan registry.
* Jika registry gagal atau tidak konsisten, sistem dapat terganggu.