MODUL 6
SCALE & LOAD BALANCE

6.1 Topik Pembahasan

1. Pengertian Scaling dan Load Balancing.
2. Jenis-jenis Scaling.
3. Konsep Auto Scaling pada AWS.
4. Konsep Elastic Load Balancer (ELB).
5. Integrasi Auto Scaling dengan Load Balancer.
6. Manfaat penerapan Scaling & Load Balancing pada sistem cloud.

6.2 Tujuan Praktikum

1. Memahami konsep dasar scaling dan load balancing dalam lingkungan cloud.
2. Mengetahui perbedaan antara vertical scaling dan horizontal scaling.
3. Mampu membuat dan mengonfigurasi Auto Scaling Group (ASG) di AWS.
4. Memahami prinsip kerja Elastic Load Balancer (ELB).
5. Mengintegrasikan ELB dengan ASG untuk menjaga kinerja sistem secara otomatis.
6. Memahami dampak scaling dan load balancing terhadap performa dan efisiensi sumber daya cloud.

6.3 Alat dan Bahan

1. Laptop
2. Browser dengan koneksi internet

6.4 Dasar Teori

6.4.1 Pengertian Scaling dan Load Balancing

Dalam arsitektur cloud computing, dua konsep utama yang menjamin fleksibilitas dan kinerja tinggi adalah scaling dan load balancing.

Scaling merupakan proses menyesuaikan kapasitas sumber daya sistem sesuai kebutuhan beban kerja. Dalam konteks cloud, scaling dilakukan secara otomatis dan dinamis tanpa perlu intervensi manual. Ketika permintaan meningkat, sistem menambah sumber daya; ketika menurun, sistem mengurangi sumber daya agar biaya tetap efisien.

Load Balancing, di sisi lain, adalah proses mendistribusikan trafik atau beban kerja ke beberapa server atau instance agar beban tidak menumpuk pada satu titik. Dengan load balancer, sistem menjadi lebih responsif, stabil, dan tahan terhadap kegagalan (fault tolerant). Kedua konsep ini bekerja saling melengkapi: scaling mengatur kapasitas sumber daya, sedangkan load balancing mengatur pembagian beban antar sumber daya tersebut.

6.4.2 Jenis Scaling

Scaling dapat dilakukan dengan dua cara utama, yaitu:

1. Vertical Scaling (Scale Up / Scale Down)

   Vertical scaling dilakukan dengan menambah kapasitas pada satu server, misalnya menambah CPU, RAM, atau storage.

   - Kelebihan: Mudah dilakukan dan tidak memerlukan konfigurasi kompleks.
   - Kekurangan: Memiliki batasan fisik dan dapat menimbulkan downtime saat upgrade.
   - Contoh: Mengubah instance dari t2.micro menjadi t2.large pada AWS EC2.

2. Horizontal Scaling (Scale Out / Scale In)

   Horizontal scaling dilakukan dengan menambah atau mengurangi jumlah server (instance).

   - Kelebihan: Lebih fleksibel, dapat dilakukan tanpa downtime, dan mendukung high availability.
   - Kekurangan: Membutuhkan sistem distribusi beban (load balancer).
   - Contoh: Menambahkan beberapa instance EC2 untuk menangani lonjakan trafik web.

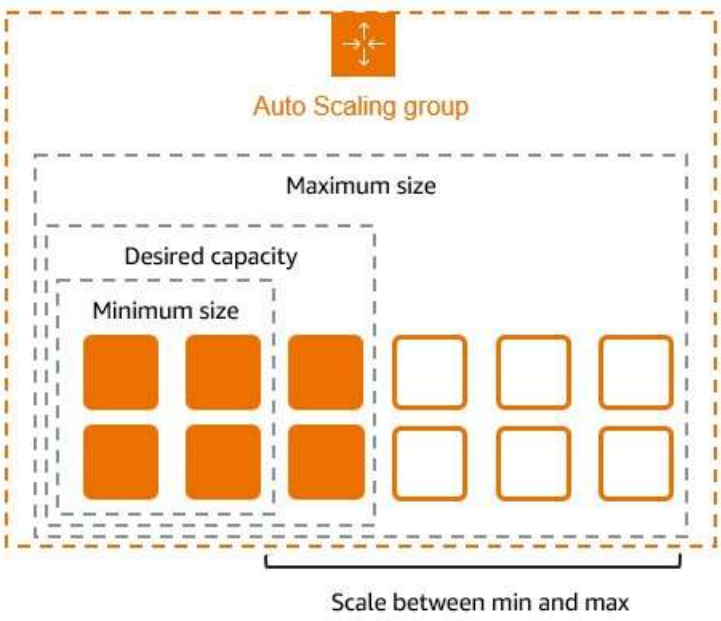
Pada cloud seperti AWS, horizontal scaling lebih umum digunakan karena memungkinkan sistem untuk tumbuh secara elastis sesuai permintaan tanpa batasan fisik.

6.4.3 Auto Scaling pada AWS

AWS Auto Scaling adalah layanan yang secara otomatis menyesuaikan jumlah instance EC2 sesuai beban kerja yang terdeteksi. Auto Scaling bekerja berdasarkan policy atau metrik seperti CPU utilization, memori, atau request count.

Komponen utama Auto Scaling terdiri dari:

1. Launch Template / Launch Configuration: Menentukan spesifikasi instance (jenis, AMI, key pair, security group, dsb) yang akan digunakan oleh Auto Scaling Group.
2. Auto Scaling Group (ASG): Kumpulan instance EC2 yang dikelola oleh Auto Scaling. ASG memastikan jumlah instance selalu berada dalam batas minimal dan maksimal yang ditentukan.



Gambar 1 *Auto Scaling Group*

Contoh Konfigurasi:

- Mininum = 1
- Desired = 2
- Maximum = 5

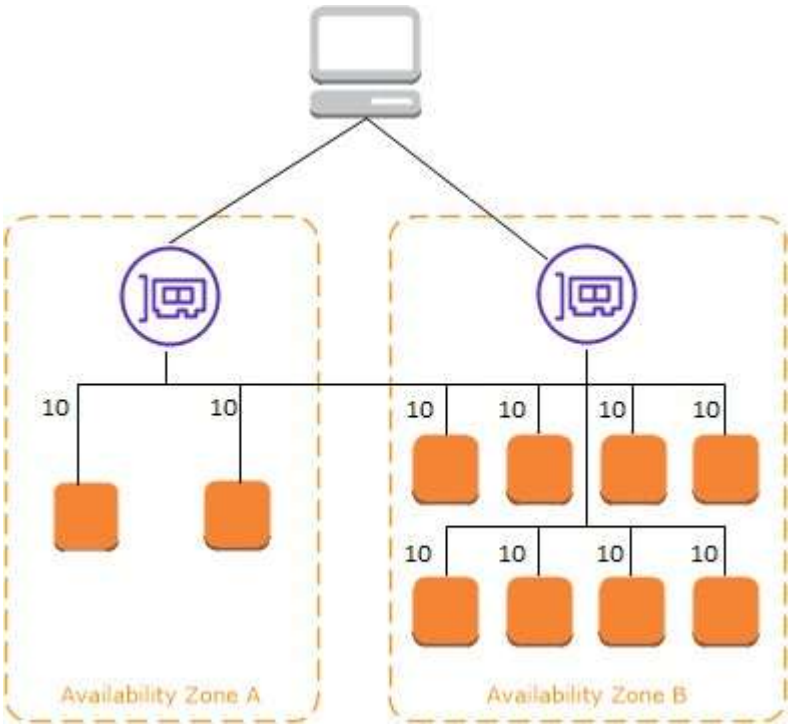
3. Scaling Policy: Menentukan aturan kapan sistem harus menambah atau mengurangi instance.

   Contoh:

   - Scale Out → jika CPU > 70% selama 5 menit.
   - Scale In → jika CPU < 30% selama 10 menit.

4. CloudWatch Alarm: Memantau metrik performa dan memicu scaling berdasarkan kondisi yang telah ditentukan.

6.4.4 Elastic Load Balancer (ELB)



Gambar 2 Arsitektur Kerja Load Balancer

Elastic Load Balancer (ELB) adalah layanan AWS yang secara otomatis mendistribusikan trafik aplikasi ke beberapa instance EC2 di satu atau lebih Availability Zone (AZ).

Fungsi utama ELB:

- Menyebarkan trafik ke beberapa server agar tidak ada yang kelebihan beban.
- Menjamin high availability dan fault tolerance.

- Memantau kesehatan (health check) setiap instance. Jika ada instance gagal, trafik otomatis dialihkan ke instance yang sehat.

AWS menyediakan beberapa jenis Load Balancer:

1. Application Load Balancer (ALB)
    - Untuk trafik berbasis HTTP/HTTPS.
    - Dapat mengarahkan request berdasarkan konten (path-based routing) dan host (host-based routing).
    - Cocok untuk aplikasi web dan API.
2. Network Load Balancer (NLB)
    - Menggunakan layer 4 (TCP/UDP).
    - Dirancang untuk performa tinggi dan latensi rendah.
    - Cocok untuk aplikasi real-time atau gaming.
3. Gateway Load Balancer (GLB)
    - Digunakan untuk trafik jaringan seperti firewall atau inspeksi paket.

6.4.5 Integrasi Auto Scaling dengan Load Balancer

Integrasi **Auto Scaling Group (ASG)** dengan **Elastic Load Balancer (ELB)** memberikan kemampuan penuh dalam otomatisasi skala dan distribusi beban.

Prosesnya sebagai berikut:

1. Pengguna mengakses aplikasi melalui Load Balancer.
2. ELB mendistribusikan trafik ke beberapa instance di ASG.
3. Ketika trafik meningkat, ASG menambah jumlah instance (Scale Out).
4. Ketika trafik menurun, ASG menghapus instance berlebih (Scale In).
5. ELB terus memantau health check agar trafik hanya dialirkan ke instance sehat.

Kombinasi ini menghasilkan sistem yang:

- Elastis: mampu beradaptasi terhadap perubahan beban kerja.
- Toleran terhadap kesalahan: jika satu instance gagal, beban segera dialihkan.

- Efisien: biaya operasional menyesuaikan dengan penggunaan aktual.

6.4.6 Manfaat Scale & Load Balance dalam Cloud

1. *High Availability* (Ketersediaan Tinggi): Aplikasi tetap berjalan walau ada instance yang gagal.
2. *Elasticity* (Elastisitas): Sumber daya bertambah atau berkurang sesuai kebutuhan.
3. Cost Efficiency (Efisiensi Biaya): Bayar hanya untuk sumber daya yang digunakan (pay-as-you-go).
4. *Performance Optimization* (Kinerja Optimal): Trafik didistribusikan merata, menghindari bottleneck.
5. *Scalability* (Dapat Diskalakan): Mendukung pertumbuhan pengguna tanpa harus mendesain ulang sistem.