

MODUL 8

SERVERLESS

8.1 Topik Pembahasan

1. Pengertian *Serverless*
2. Kelebihan *Serverless*
3. Kekurangan *Serverless*
4. Contoh Pengaplikasian *Serverless*

8.2 Tujuan Praktikum

Setelah menyelesaikan modul ini, Anda diharapkan dapat:

1. Mengetahui dan memahami apa itu *Serverless*.
2. Mengetahui dan memahami kelebihan dan kekurangan *Serverless*.
3. Mengenali kategori *Serverless*.
4. Membangun sebuah *Serverless* menggunakan AWS Educate.

8.3 Alat dan Bahan

1. Laptop
2. Mouse
3. Jaringan Internet
4. Akun AWS Educate

8.4 Dasar Teori

8.4.1 *Serverless*

Serverless adalah sebuah konsep dimana developer tidak perlu lagi memikirkan infrastruktur dan mengelola server sebagai bagian dari aplikasi mereka. Konsep ini memungkinkan developer untuk fokus pada pengembangan kode dan fitur, sementara layanan cloud provider mengelola infrastruktur dan skalabilitas. Kelebihan utama dari arsitektur ini adalah kemampuannya untuk menghilangkan kebutuhan akan pengelolaan infrastruktur secara manual.

Serverless computing adalah model pemrograman di mana berintegrasi dengan cloud computing yang disediakan oleh cloud provider. Dengan menggunakan arsitektur

Serverless, para developer dapat fokus sepenuhnya pada logika bisnis dan fungsionalitas aplikasi, tanpa harus memikirkan tentang skala, pemeliharaan, atau penanganan beban kerja. Salah satu kelebihan yang paling menonjol dari arsitektur *Serverless* adalah skalabilitasnya yang otomatis. Dalam model ini, sistem secara dinamis mengalokasikan sumber daya sesuai dengan permintaan yang masuk. Ini berarti aplikasi dapat dengan mudah menangani lonjakan lalu lintas tanpa mengalami *downtime* atau kinerja yang buruk. Dengan demikian, pengguna dapat menikmati pengalaman yang lebih baik tanpa harus khawatir tentang skalabilitas infrastruktur.

8.4.2 Kelebihan *Serverless*

Ada beberapa manfaat dari *Serverless* yaitu:

1. *Scalling* Otomatis

Scalling otomatis merujuk pada kemampuan infrastruktur jaringan menangani penambahan atau penurunan beban kerja secara otomatis. Jadi, user tidak perlu mengatur server di data center secara manual lagi yang tentu akan berpengaruh pada efisiensi waktu serta performa sistem.

Manfaat ini akan sangat berguna ketika sedang terjadi lonjakan traffic web yang tinggi sehingga diperlukan peningkatan beban untuk mengimbangnya. Tujuannya adalah agar situs tetap berjalan lancar ketika diakses oleh banyak user dalam satu waktu bersamaan.

2. Tarif Server Lebih Terjangkau

Membangun infrastruktur server mandiri tentu akan membutuhkan biaya yang tidak sedikit. Maka dari itu, *serverless computing* jadi salah satu alternatif tepat karena sudah disediakan oleh cloud provider.

Pengguna hanya perlu membayar jasa dan sumber daya yang digunakan untuk mengoperasikan sistem. Selain itu, tarif pembayaran juga dihitung sampai milisecond yang tentu akan menghemat biaya.

3. Ketersediaan Data Dan Sistem Lebih Stabil

Serverless computing terintegrasi dengan cloud computing yang merupakan media penyimpanan data berbasis digital. Dengan integrasi tersebut, data bisa terus diambil kapan pun karena selalu tersedia.

Selain itu, sistem operasional jaringan juga lebih stabil jika dibandingkan dengan infrastruktur jaringan biasa. Dikatakan lebih stabil karena jika terjadi kegagalan pada satu sistem, hal tersebut tidak akan mengganggu operasional jaringan secara keseluruhan. Dengan begitu, Anda tetap bisa menjalankan situs dengan lancar.

8.4.3 Kekurangan *Serverless*

1. Kode Pemrograman Masih Terbatas

Kekurangan utama dari layanan ini adalah belum semua provider mendukung penyediaan bahasa pemrograman terbaru. Pada beberapa provider, mereka hanya menyediakan bahasa, seperti Javascript, Python, dan Java. Namun, tidak menutup kemungkinan bahwa sekarang banyak juga layanan yang sudah didukung oleh bahasa pemrograman terbaru.

2. Pembatasan Sumber Daya

Walaupun hampir seluruh server dikendalikan oleh provider, bukan berarti tidak ada pembatasan dalam penggunaannya. Provider umumnya membatasi kapasitas memori yang dapat dialokasikan untuk setiap sumber daya dalam sistem.

Jika sumber daya tersebut melebihi batasan memori, kegagalan sistem mungkin saja bisa terjadi. Oleh karena itu, Anda perlu memperhatikan dan mengoptimalkan penggunaan memori dalam fungsi layanan ini agar dapat menyesuaikan performa situs.

3. Kesulitan Mengatasi Bug

Dalam mengoperasikan sistem pasti seringkali muncul bug atau permasalahan pada jaringan. Jika menggunakan *serverless computing*, tentu Anda bergantung pada penanganan dari provider yang akan membutuhkan waktu lebih lama.

Jadi, secara tidak langsung, hal ini dapat menyebabkan penurunan pelayanan kepada user atau customer Anda karena mereka tidak bisa mengakses informasi pada situs.

8.4.4 Contoh Pengaplikasian *Serverless*

Layanan **serverless** dapat digunakan dalam berbagai aplikasi bisnis yang mengutamakan efisiensi dan responsivitas tanpa harus mengelola infrastruktur server. Berikut adalah beberapa contoh pengaplikasiannya:

1. **Chatbot Aplikasi**

Membangun aplikasi chatbot yang responsive, aplikasi ini mampu menangani interaksi pengguna secara otomatis, memproses permintaan, dan memberikan respons yang sesuai. Hal ini bermanfaat bagi bisnis karena membantu meningkatkan layanan pelanggan dan mengurangi risiko kesalahan manusia (human error), seperti karyawan yang lupa membalas pesan pelanggan.

2. **REST API**

Untuk membangun API RESTful tanpa memerlukan server. Arsitektur ini memungkinkan API untuk menangani permintaan dari aplikasi pengguna dengan cara yang skalabel dan hemat biaya, mengoptimalkan pengelolaan sumber daya sesuai kebutuhan.

3. **Aplikasi IoT (Internet of Things)**

Untuk memproses data yang dikirim oleh perangkat IoT secara real-time. Lambda dapat memproses data sensor dari perangkat IoT dan mengelola respons atau aksi yang diperlukan berdasarkan kondisi atau peristiwa yang terjadi, mendukung pengelolaan data sensor secara otomatis dan efisien.