MICROSERVICES
&
CLOUD COMPUTING

Hansini Gunasena
BSc. In Computer Science

Senior Software Engineer – Emerging Tech
Virtusa pvt Ltd

What Are Software Architectural Styles?

- A **software architectural style** (or **architecture pattern**) is a **standard way of organizing and structuring software systems**.
- It defines **how components of a system interact**, **how data flows**, and **how the system is deployed and maintained**.

- **Purpose of Architectural Styles**

They help you:

- Decide **how to organize** your code and components
- Plan **communication** between modules
- Manage **scalability, reliability, and performance**
- Choose **tools, databases, and technologies**

Common Software Architectural Styles

Style	Description	Example Use Case
Layered (N-Tier)	Application divided into layers (e.g., presentation, business, data).	Web apps like online banking systems.
Client-Server	Client requests services; server provides them.	Websites, email systems.
Monolithic	Entire app built as one single unit.	Simple applications or startups.
Service-Oriented Architecture (SOA)	System made of reusable services connected via a message bus.	Enterprise systems.
Microservices	Application divided into small, independent services.	Cloud-native apps, e-commerce.
Event-Driven Architecture	Components react to and communicate via events.	Real-time apps like stock trading or IoT systems.
Pipe-and-Filter	Data flows through a sequence of processing steps (filters).	Data processing or compiler design.

What are Microservices?

- Microservices is a ***software architectural style.***
- application is built as a **collection of small, independent services** that communicate with each other — usually over APIs.
- Each service:
  - Focuses on a **single business function** (e.g., user login, payment, inventory).
  - Can be **developed, deployed, and scaled** independently.
  - Often has its **own database** and runs in its **own process**.

Examples

What Is Cloud Computing?

- **Cloud computing** is the **delivery of computing services** — such as **servers, storage, databases, networking, software, and analytics — over the internet ("the cloud")**, instead of running them on your own computer or local servers.

In simple terms:

- Instead of buying and managing your own hardware or software, you **rent computing power or storage from a cloud provider** (like AWS, Azure, or Google Cloud) and pay only for what you use.

- **Key Idea**

You don't have to:

- Build or maintain physical servers
- Install software locally
- Worry about scaling or security
- The cloud provider manages all that — you just focus on your application.

References

1. Title : Spring Microservices in Action
Author : John Carnell, Kalpit Patel
Publisher : Manning Publications

2. Title : The Cloud Computing Book: The Future of Computing Explained
Author : Douglas Comer
Publisher : Chapman and Hall/CRC

Exercise :

We are going to design a library application using microservices principles. The goal is to break down the system into small, independent services that can work together.

Keep it simple: borrowing books, returning books, user management, catalog search, and notifications.