# Cloud Computing

# Learning Outcomes

- Understanding fundamentals of Cloud Computing and underlying architecture
- Implement resilience and high performance Cloud Applications
- Learn underlying theorem, algorithms and patterns in building Cloud Applications
- Design and implement a simple Cloud Applications.

# Assessment

- Continuous Assessments (50%)
  - Take home assignment
  - MCQ
  - 1 Project
- End Semester Examination (50%)
- Complete all components of continuous assessment

# Recommended Readings


- Erl, T., Puttini, R. and Mahmood, Z., 2013. Cloud computing: concepts, technology & architecture. Pearson Education.
- Mather, T., 2009. Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance.
- Kavis, M., 2014. Architecting the cloud: design decisions for cloud computing service models (SaaS, PaaS, and IaaS). John Wiley & Sons, Inc., Hoboken, New Jersey.
- Richardson, C., 2018. Microservices patterns: with examples in Java. Simon and Schuster.

# Introduction

Cloud Computing

# Learning Outcome

- Understand what cloud computing is
- Learn different computing models
- Advantages and challenges of cloud computing
- Definition of Cloud Computing

# Cloud

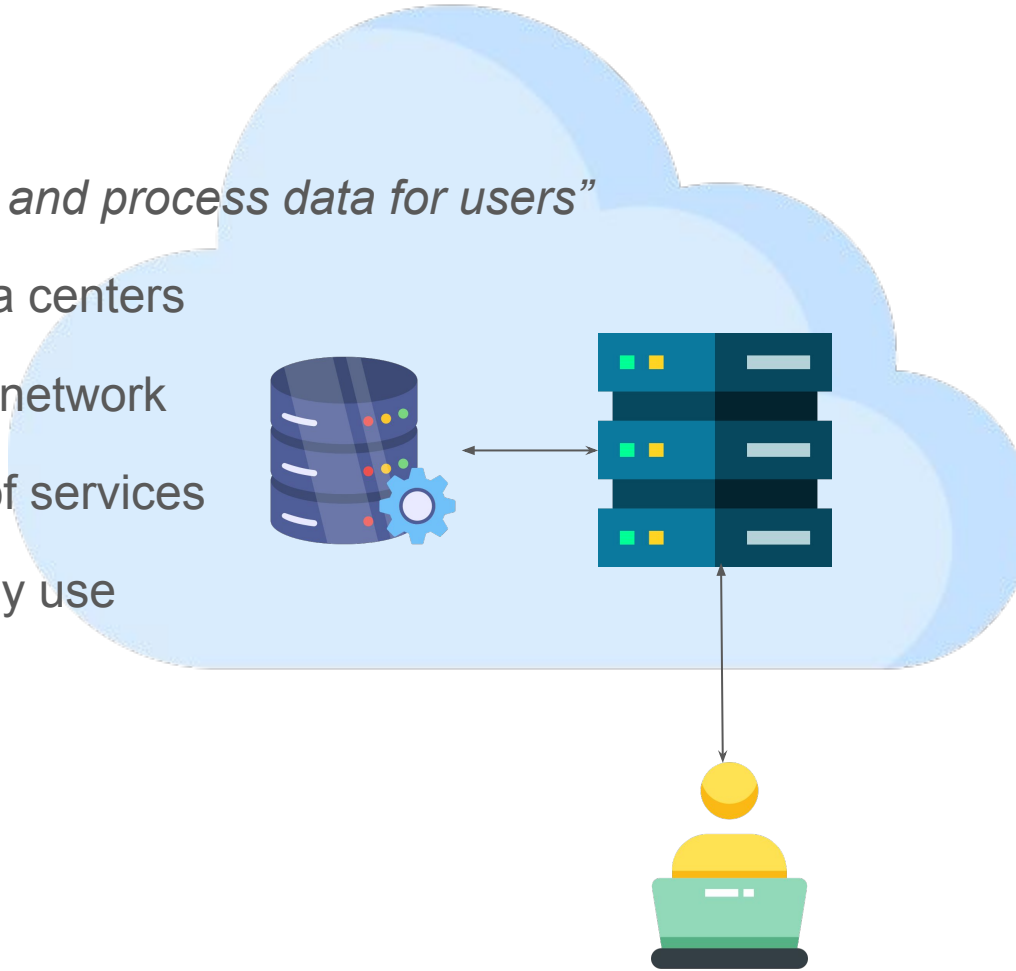
*“A network of remote servers that store and process data for users”*

**Storage**: Data stored on servers in data centers

**Access**: User access their data over a network

**Service**: Cloud providers offer variety of services

**Pricing**: User pay for the resources they use



# Computing Models

- Personal Computing
- Mobile computing
- Utility Computing
- Distributed Computing
- Parallel Computing

# Personal Computing

- Hardware customization
- Local maintenance
- Low utilization
- Low accessibility
- High Capex, low Opex

Capex: **Ca**pital **Ex**penditures

Opex: **Op**erational **Ex**penditures

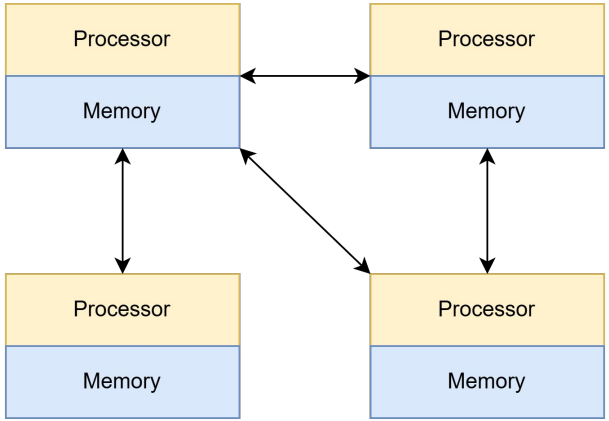
# Mobile Computing

- Since 1990s
- Computing at anytime - anywhere
- Affordable
- Connected through wifi or cellular network
- Low power consumption processing
- Limited power and computation capacity

# Utility Computing

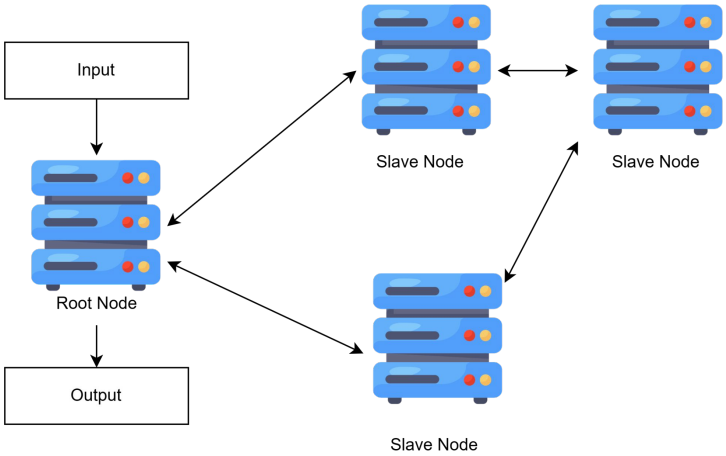
- Begin from 2005-2006
- Infrastructure provided to users on demand
- Resources are scalable and flexible based on demand
- Pricing based on resource usage
- No upfront cost
- No need to maintain hardware

# Distributed Computing




- Multiple computers connected over a network
- Each node has private memory
- *Cluster Computing*: Tightly coupled computers, single system image
- *Grid Computing*: Loosely coupled computers, can be owned by multiple organizations

# Cluster Computing




- Tightly coupled computers that work together as a single system.
- Identical hardware and software
- Each node has private memory but is connected through a fast local network.
- Communication between nodes is faster and more reliable.
- Used for high-performance computing tasks such as scientific simulations and data analysis.

# Cluster Computing

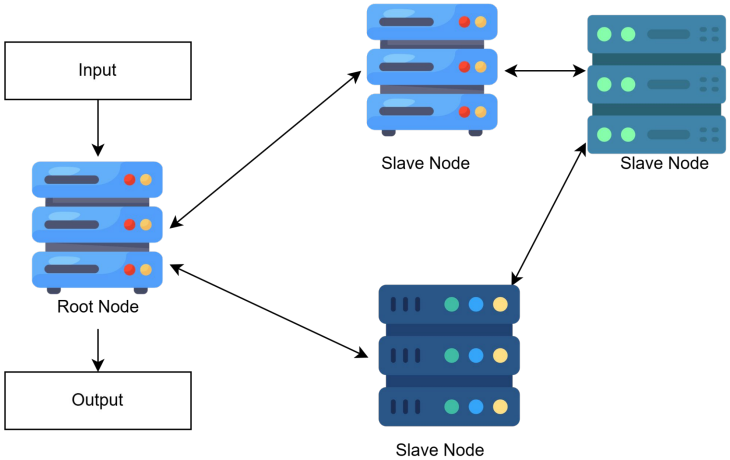
**Pros**

- Easy to manage due to homogeneity
- Scalable

**Cons**

- Higher cost due to homogeneity
- Limited by space

# Grid Computing




- Loosely coupled computers
- Can be owned by multiple organizations
- Each node has private memory
- Connected through a network
- Used for large scale tasks

# Grid Computing

**Pros**

- High resource utilization
- Scalability
- Flexibility as heterogeneous
- Availability

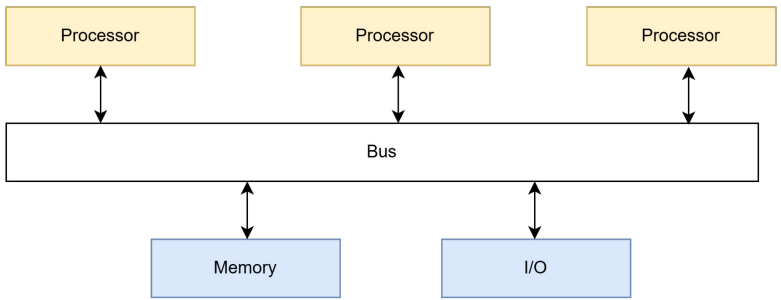
**Cons**

- Complexity due to heterogeneous
- Performance impact on network latency

# Grid vs Cluster Computing

| **Cluster Computing** | **Grid Computing** |
| --- | --- |
| Homogeneous | Heterogeneous |
| Dedicated resources | Unused resource |
| Nodes located closer | Nodes can be locate far |
| Network with high speed LAN | Network with low speed WAN |

# Parallel Computing




- Utilize multiple nodes to concurrently compute smaller problems.
- Combines many processors to work on a single task by dividing the problem into smaller problems
- Smaller problems calculated concurrently on multiple nodes
- Uses shared memory
- Ideal for tasks that can be broken down into smaller, independent parts

# Types of Parallelism

- Bit-level parallelism
  - Ex 32-bit to 64-bit instructions set
- Instruction-level parallelism
  - Multiple instructions in multiple cores
- Superword level parallelism
  - Apply same instruction over array data
- Task parallelism
  - Set of instructions distributed over cores

# Amdahl’s Law

- Illustrates the potential speedup of a process
- Dependent on parallelizable and non-parallelizable components
- Parallelizable portion sped up due to multiple processors
- Non-parallelizable portion remains constant, limiting overall speedup
- Highlights the importance of identifying and optimizing bottlenecks

# Amdahl's Law

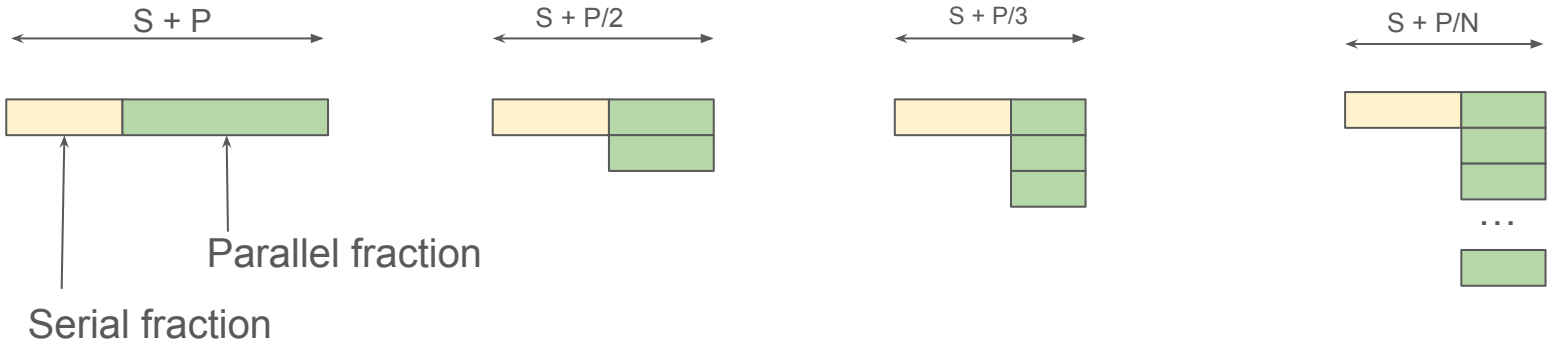
$$Speedup = \frac{Old\ Execution\ Time}{New\ Execution\ Time}$$

$$Overall\ Speedup = \frac{1}{1 - P + \frac{P}{N}}$$

P is the proportion of the system that can be improved

N is the number of processors in the system

# Amdahl's Law



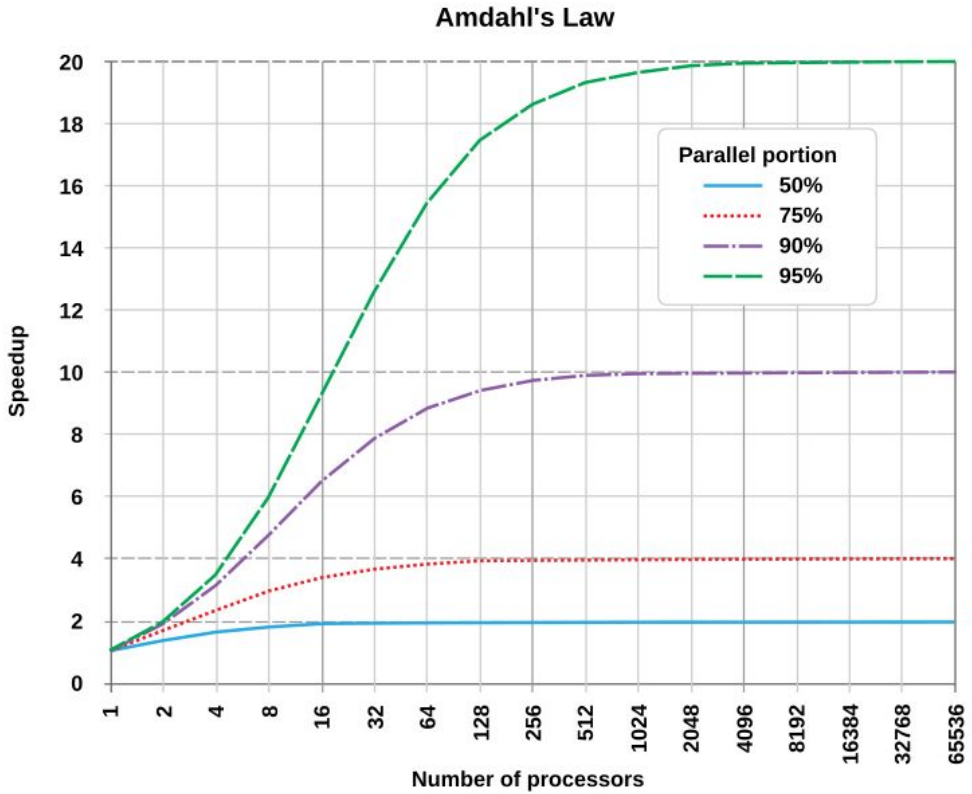

$$Improved\ Time = S + \frac{P}{N}$$

$$Improved\ Time = 1 - P + \frac{P}{N}$$

$$Overall\ Speedup = \frac{1}{1 - P + \frac{P}{N}}$$

S - Serial fraction
P - Parallel fraction
N - Number of cores

# Amdahl’s Law


Amdahl's Law
Speedup
Number of processors
Parallel portion
50%
75%
90%
95%
0
2
4
6
8
10
12
14
16
18
20
1
2
4
8
16
32
64
128
256
512
1024
2048
4096
8192
16384
32768
65536

# Cloud Computing Definition

Cloud computing is the **on-demand access** of computing resources—physical servers or virtual servers, data storage, networking capabilities, application development tools, software, AI-powered analytic tools and more—**over the internet** with **pay-per-use pricing**.

- IBM

cloud computing is the **delivery of computing services**—including servers, storage, databases, networking, software, analytics, and intelligence—**over the internet** (“the cloud”) to offer faster innovation, flexible resources, and economies of scale. You typically **pay only for cloud services you use**, helping you lower your operating costs, run your infrastructure more efficiently, and scale as your business needs change.

- Azure

Cloud computing is the **on-demand delivery** of IT resources **over the Internet** with **pay-as-you-go** pricing. Instead of buying, owning, and maintaining physical data centers and servers, you can access technology services, such as computing power, storage, and databases, on an as-needed basis from a cloud provider like Amazon Web Services (AWS).

- AWS

# NIST definition for Cloud Computing

Cloud computing is a model for enabling **ubiquitous, convenient, on-demand network access** to a **shared pool** of **configurable computing resources** (e.g., networks, servers, storage, applications, and services) that can be **rapidly provisioned and released** with minimal management effort or service provider interaction. This cloud model promotes **availability** and is composed of **five essential characteristics**, **three service models**, and **four deployment models**.

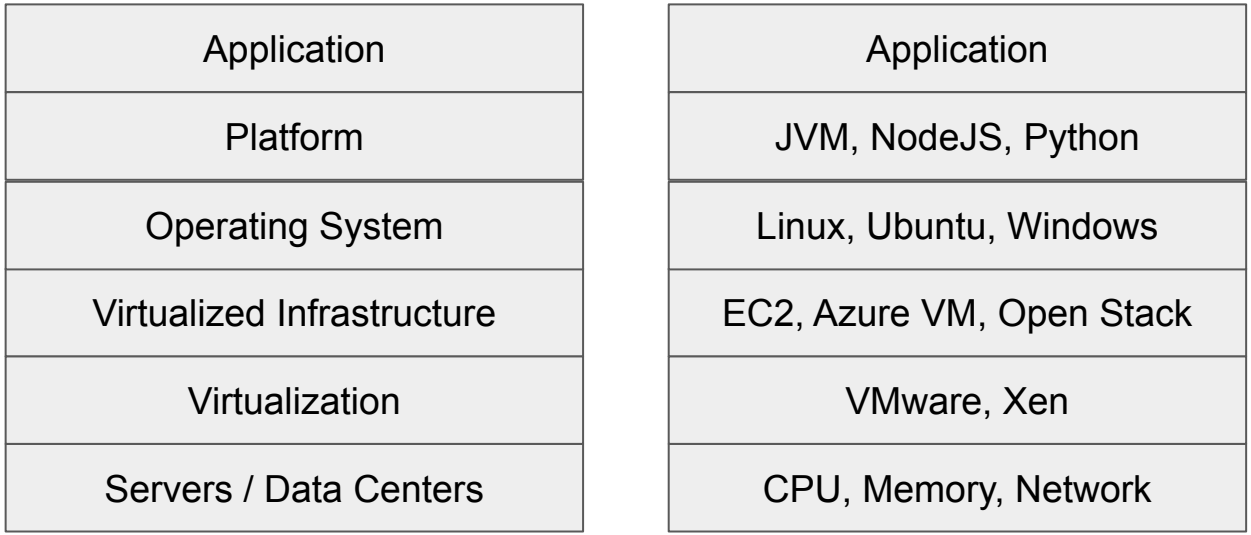
# NIST Definition’s Essential Characteristics

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

# Cloud deployment models

- **Public Cloud** - Cloud available for general public. The cloud owned by the cloud service provider (Ex AWS, GCP, Azure)
- **Private Cloud** - Not for general public. The cloud owned by a cloud service provider.
- **Hybrid Cloud** - Composition of using both public and private cloud.

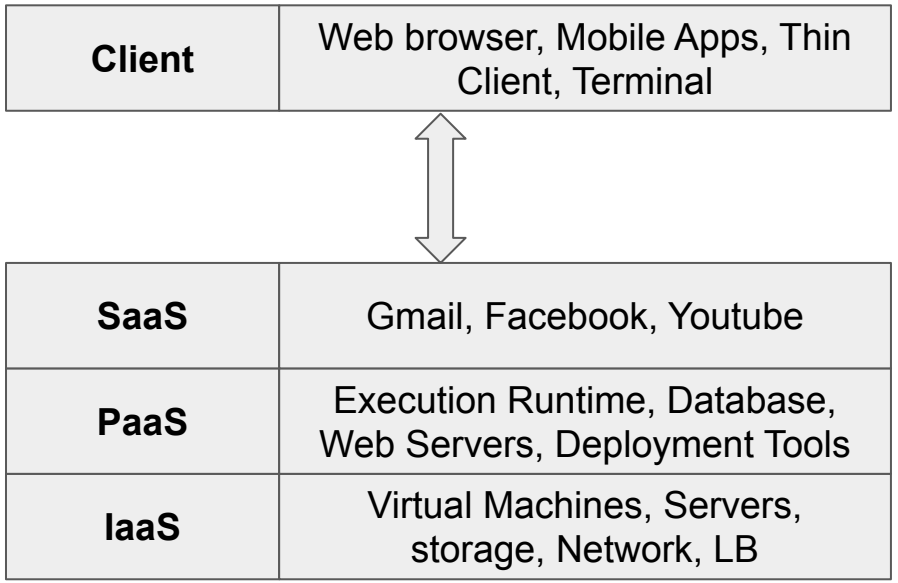
# Layers of Cloud Computing



# Cloud Computing Service Models

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

- Database as a Service (DBaaS)
- Container as a Service (CaaS)
- X as a Service (XaaS)



# Software as a Service (SaaS)

- Applications are accessible from various client devices through a thin client interface, such as a web browser.
- Users do not manage or control the underlying cloud infrastructure.
- Cloud provider manages the infrastructure, including network, servers, storage, and databases.
- Cloud provider handles software maintenance, including updates and patches.
- Ex: GMail, Facebook

# Platform as a Service (PaaS)

- Software Developers layer
- User has control on the business logic of the application
- No control over the underlying infrastructure
- Ex: Google App Engine, Heroku

# Infrastructure as a Service (IaaS)

- Gives user control over VM, CPU, Storage, Memory, OS, and Network
- Users can run any application they want on the rented servers
- Cloud provider maintains the hardware infrastructure
- User maintains the virtualized infrastructure
- Examples: Amazon EC2, Microsoft Azure, Google Compute Engine

# Cloud Computing Advantages

- Less initial investment cost
- Pay as you go / Cost efficiency
- Global accessibility
- Disaster Recovery
- Scalability and flexibility

# Challenges of Cloud Computing

- Data confidentiality and security
  - Storing data securely (Encryption)
  - Authentication and authorization
  - Audits
  - Intrusion detection and prevention
  - Physical security
- Vendor locking
- Service availability
- Datacenter resource management

# Challenges of Cloud Computing

- De-perimeterization
  - When cloud grows who own which is problem.
  - Erosion of traditional network boundaries
- Key Considerations
  - Data protection across distributed resources
  - Identity and access management
  - Network security in cloud environments
- Best practices
  - Zero-trust security model (Ex Whatsapp)
  - Multi-layer security approach
  - Continuous monitoring and assessment

# Cloud Computing Emerging Trends

- Serverless computing
  - No need to manage underlying infrastructure
  - Stateless
  - Ex: AWS Lambda, Azure Functions
- Edge computing
  - Processing data closer to the source
- Cloud Native
  - Cloud application runs on microservices
  - Ex: Kubernetes, Docker Swarm
- Quantum computing on Cloud
  - Providing quantum computing over cloud
  - Ex: Amazon Bracket

# Deployment Architecture

Multi Tier architecture

- Single tier
- Two tier
- Three tier

Service architecture

- Service Oriented Architecture
- Microservices Architecture

Serverless architecture

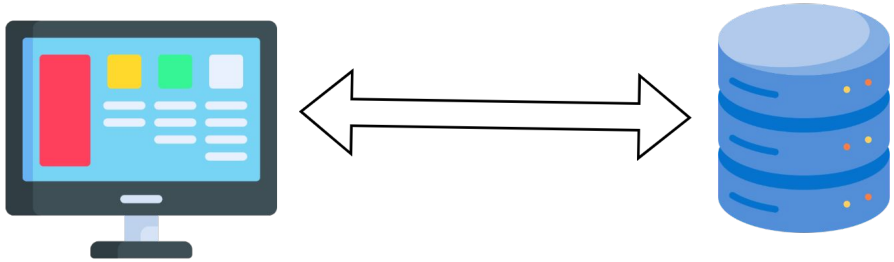
# One-Tier Architecture

- User interfaces, business logics, databases reside in the same machine
- Simplest application architecture
- Data accessible for on the same computer

Ex: MS Word, Photoshop

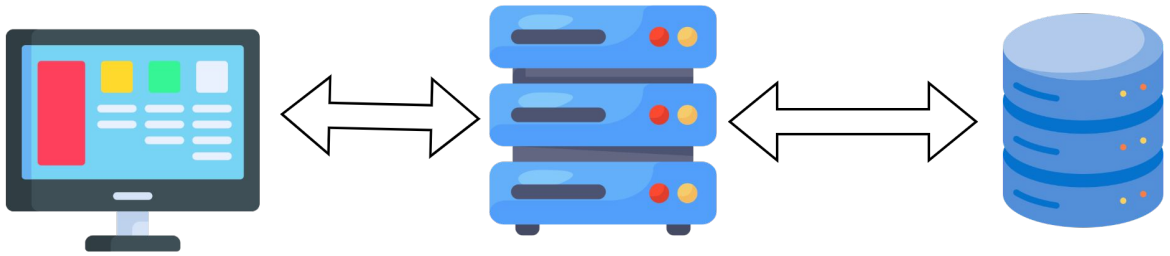
# Two-Tier Architecture

- Separated client and server
- User interface to access data but data stored in a remote server
- Backend and frontend communicate over a network
- Business logics on the client side



# Three-Tier Architecture

- Three layers in the system
  - Presentation layer(Web, mobile, desktop)
  - Application layer
  - Data access layer(SQL, NoSQL)
- Each layer connected with an API
- Easier update and modification
- Reusable code
- Scalable architecture



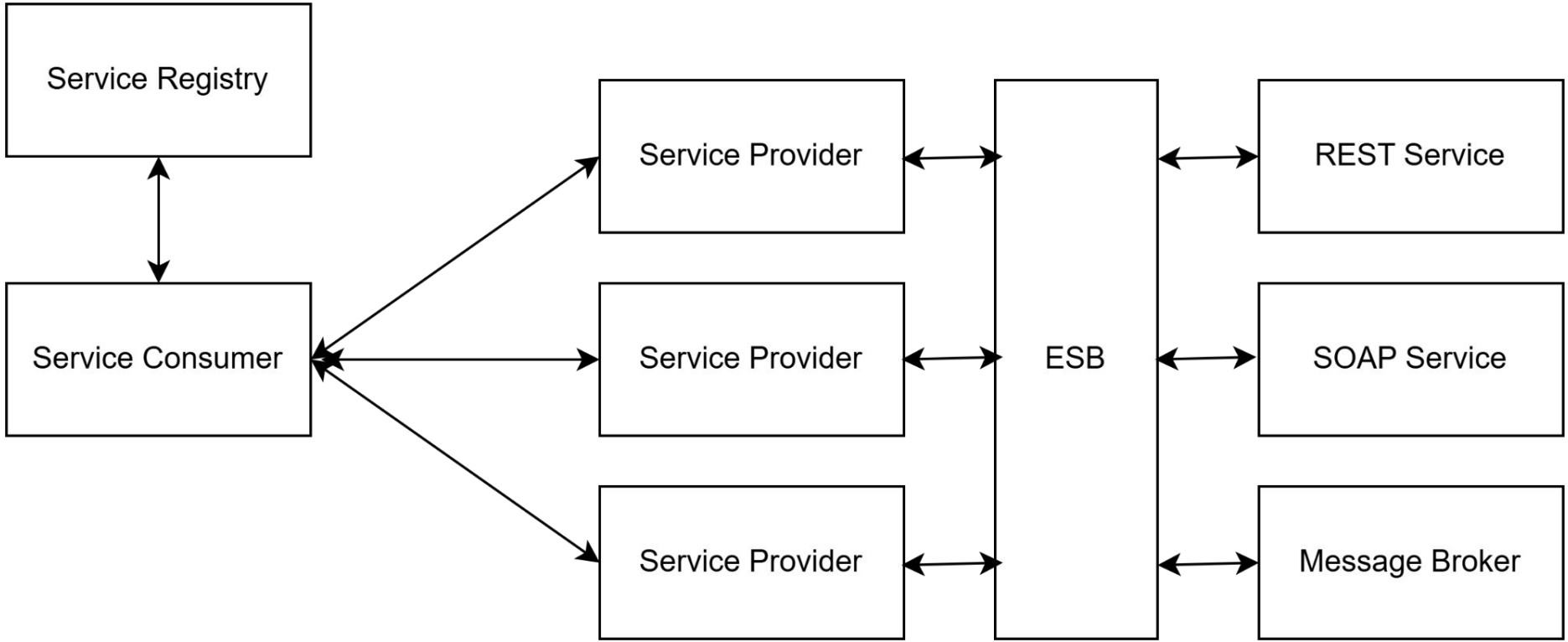
# Service Oriented Architecture (SOA)

- Services with well defined scope
- Service abstraction / Hidden inner logic
- Standard service contract
- Service discovery
- Loosely coupled services
- Stateless
- Independent of underlying technology

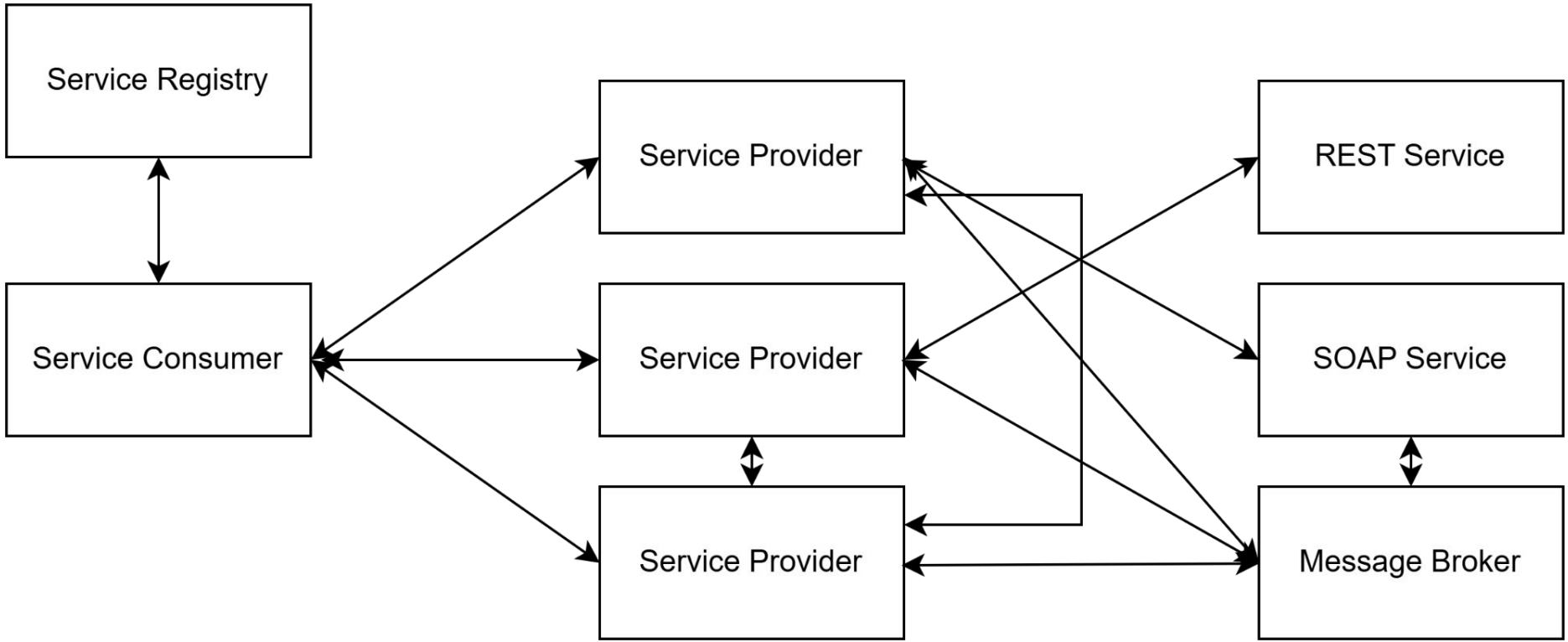
# Service Oriented Architecture Terms

- Service contract written in **Web Services Description Language** (WSDL)
- Service communication through **Simple Object Access Protocol** (SOAP)
- Service discovery with **Universal Description, Discovery and Integration** (UDDI) registry
- Message and request routing with Enterprise Service Bus (ESB)

# Service Oriented Architecture



# Service Oriented Architecture Without ESB



# Service Oriented Architecture

Pros

- Promote Reusability by services
- Facilitate Integration
- Support Scalability
- Technology Independence

Cons

- Complexity
- Performance Overhead
- Infrastructure cost

# Microservices Architecture

Micro - manageable by a single development team (5 - 6 members)

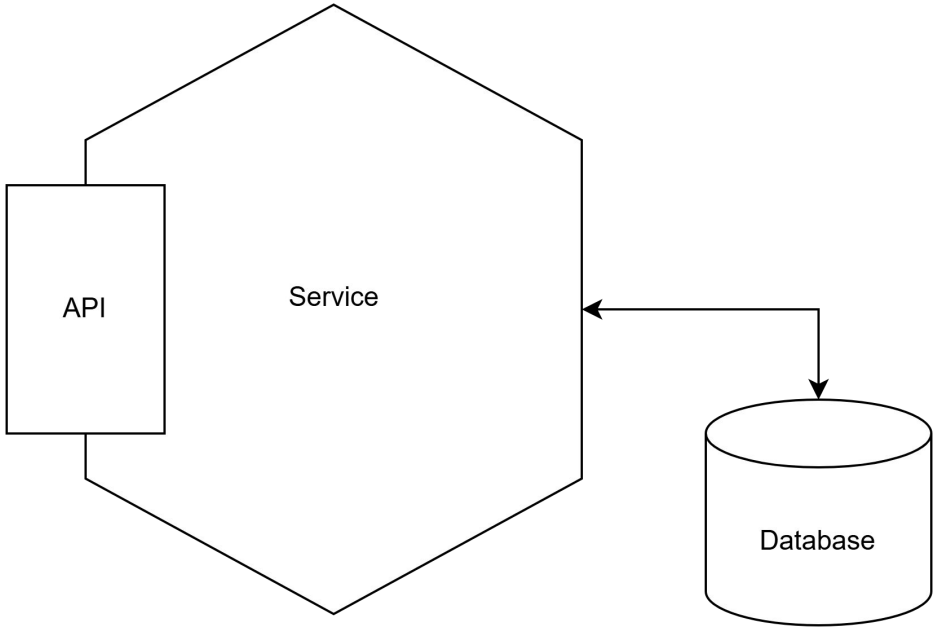
Handle only one entity (Ex Customer, Order)

Database per service design pattern

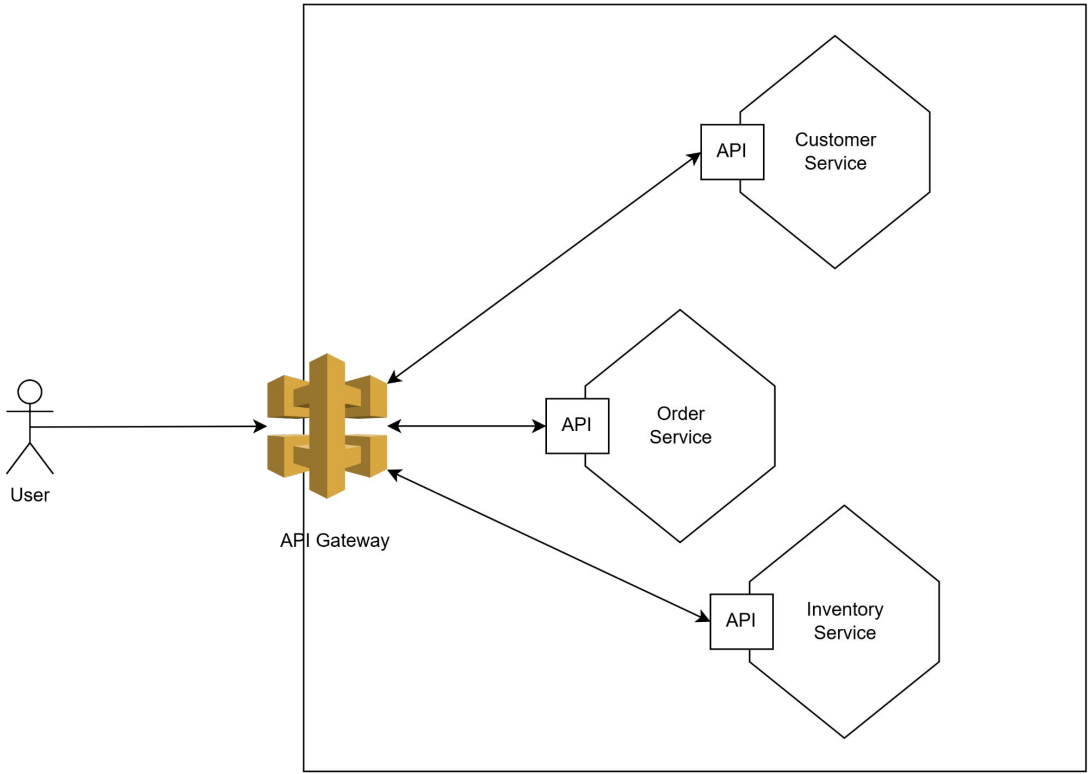
Components and connectors

- Components: The program
- Connectors: Communication protocols (Ex: REST, JSON)

# Microservices Architecture



# Microservices Architecture



# Microservices Architecture

Pros

- Scalability
- Technological flexibility
- Faster development

Cons

- Complexity
- Resource overhead
- Network latency
- Testing complexity
- Hard debug

# SOA vs MSA

| **SOA** | **MSA** |
| --- | --- |
| Shared data storage | Independent storage |
| Various different protocol | Mainly the HTTP/REST |
| SQL | NoSQL |
| ESB for communication | Simple messaging system |

# Thank You