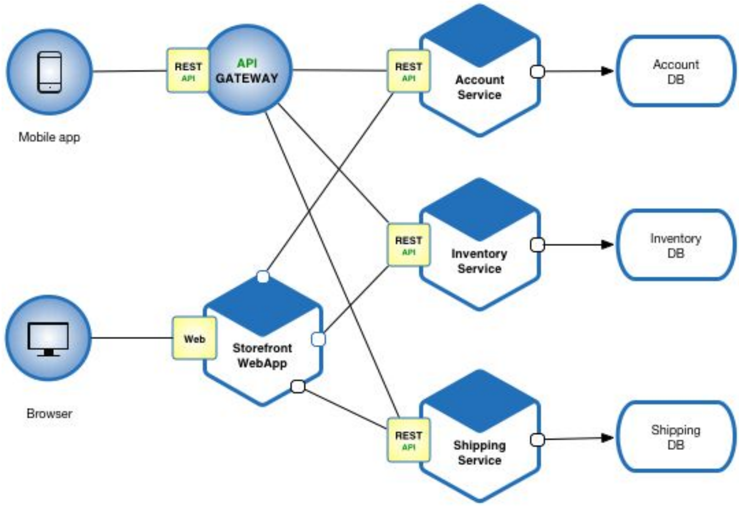
Microservices Architecture

BRAC UNIVERSITY

Inspiring Excellence

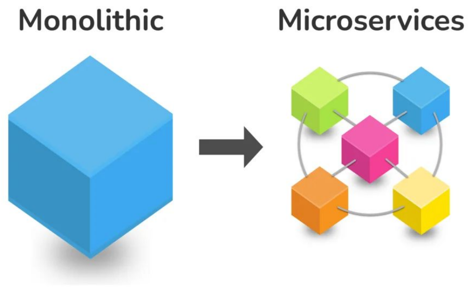
What is Microservice Architecture?

- A design approach where a single application is built as a suite of small, independent services
- Each service run at its own process and communicates via lightweight mechanism (e.g., API)
- Focuses on modularity, scalability and flexibility



Microservices VS Monolithic Architecture

- Monolithic Architecture:
  - Single codebase for entire app.
  - Tightly coupled components.
  - Scaling requires to work on the whole app.
  - Hard to adopt new tech.
- Microservices:
  - Multiple independent services.
  - Loosely coupled, modular design.
  - Scale specific services only.
  - Flexible tech-agnostic (different languages can be used to implement different services)



Key Characteristics of Microservices

- **Independence**: Each service is developed, deployed, and scaled independently
- **Decentralized**: No central control; services use their own databases
- **Single Responsibility**: Each service focuses on a specific business function
- **Interoperability**: Services communicate using standard protocols (HTTP, Message brokers like Apache Kafka)

BRAC UNIVERSITY
Inspiring Excellence

Benefits of Microservices

- **Scalability:** Scale individual services based on demand
- **Flexibility:** Use different technologies for different services
- **Faster Development:** Small teams work on separate services in parallel
- **Resilience:** Failure in one service does not affect the entire system

Challenges of Microservices

- **Complexity:** Managing multiple services increases operational overhead
- **Communication:** Inter-service communication can introduce latency
- **Infrastructure Cost**: Independent CI/CD pipelines for each services increases cost
- **Testing:** End-to-end testing across services is difficult
- **Application:** Only applicable to a large software with multiple large teams with multiple services. Small companies cannot adopt it due to above mentioned challenges.

Real World Examples

- Netflix in **2009** shifted from Monolithic to Microservices architecture.
- The most significant shift involved breaking down customer-facing features like **sign-up, movie selection, and configuration** into separate, independent microservices.
- Netflix completed the core migration to a microservices architecture by **2012**.
- Netflix now utilizes **hundreds** of microservices to power its streaming platform, enabling **high scalability and frequent deployments**.
- Other Examples: **Amazon, Uber, Airbnb** etc.

Conclusion

- Microservices offer scalability and flexibility but come with complexity
- Ideal for large, dynamic systems but require careful design and management
- Essential for modern software engineering in distributed environments

BRAC UNIVERSITY
Inspiring Excellence