# Monolithic & Microservices

## Monolithic Architecture

The entire application is built as **one single unit**:

- UI
- Business logic
- Database access
  All packaged and deployed together.

**Pros**

- Simple to build and deploy
- Easy debugging
- Easy to test
- Simple deployment
- No network latency between modules
- Lower infrastructure cost

**Cons**

- Tight coupling makes changes risky (one bug can crash everything)
- Technology lock-in (stuck with one tech stack)
- Hard to scale selectively (must scale whole app even if only one feature needs it)
- Longer deployment times as the app grows
- Difficult for large teams (everyone working in same codebase causes conflicts)

## Microservices Architecture

Application is split into **small, independent services**, each responsible for a single business capability.

Each service:

- Has its own codebase
- Often its own database
- Communicates via HTTP / gRPC / message queues / REST apis

**Pros**

- Loose coupling
- Independent scaling (scale only what is required)
- Independent deployment
- Better fault isolation
- Team autonomy
- Supports polyglot tech stacks

**Cons**

- Complex architecture
- Complicated testing and debugging
- Higher infrastructure & ops cost
- Requires strong DevOps and monitoring
- Distributed system problem
  - Network failures
  - Latency
  - Data consistency

## Interview Tip

If asked *"Which one would you choose?"*, say:

"It depends on scale, team size, and business requirements."

| Aspect | Monolithic | Microservices |
|---|---|---|
| Codebase | Single | Multiple |
| Deployment | One unit | Independent |
| Scaling | Whole app | Per service |
| Coupling | Tight | Loose |
| Fault isolation | Poor | Strong |
| Complexity | Low | High |
| DevOps need | Minimal | Heavy |
| Best for | Small apps | Large systems |

**When to Use Each**

**Choose Monolithic when:**

- Building a new product or MVP (get to market faster)
- Small team (under 10 developers)
- Simple, well-defined domain
- Limited traffic/scaling needs initially
- Want to minimize operational overhead

**Choose Microservices when:**

- Large, complex application with distinct business domains
- Large development team
- Different parts need independent scaling
- Want to use different technologies for different problems
- Need high availability and fault tolerance
- Have mature DevOps capabilities