MICROSERVICES

Monolithic software

- Difficult to scale
  - Architecture is hard to maintain and evolve
    - Lack of agility
- Long build/test/release cycle
  - New releases take months
    - Lack of innovation
- Operations is a nightmare
  - Long time to add new features
    - Frustrated customers

Microservice: A service-oriented architecture composed of loosely coupled elements that have bounded contexts.

- *Service-oriented architecture*: services communicate each other over the network
- *Loosely coupled elements*: you can update the services independently; updating one service doesn't require changing any other services.
- *Bounded contexts*: Self-contained; you can update the code without knowing anything about the internal of other microservices

Anatomy of a microservice: Data store + Application/logic (code, libraries, etc) + public API

Microservice principles:

1. Microservices only rely on each other's public API
   a. Hide your data
   b. Document your APIs
   c. Define a versioning strategy
2. Use the right tool for the job
   a. Embrace polyglot persistence
   b. Embrace polyglot programming framework
3. Secure your services
   a. Defense-in-depth
   b. Authentication and authorization
4. Be a good citizen within the ecosystem
   a. Have a clear SLAs
   b. Distributed monitoring, logging and tracing
5. More than just technology transformation
   a. Embrace organizational change
   b. Favor small focused dev teams
6. Automate everything
   a. Adopt DevOps

Benefits of microservices

1. Easier to scale each individual micro-service
   a. Easier to maintain and evolve system
      i. Increased agility

2. Rapid build/test/release cycles
    a. New releases take minutes
        i. Faster innovation
3. Clear ownership and accountability
    a. Short time to add new features
        i. Delighted customers