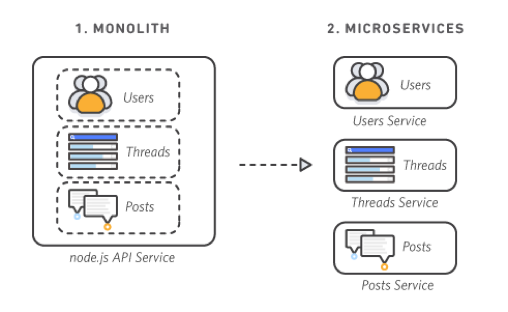
**Microservices** are an **architectural and organizational approach to software development** where **software is composed of small independent services** that **communicate over well-defined lightweight APIs**. These services are owned by small, self-contained teams.

**Microservices architectures make applications easier to scale and faster to develop**, enabling innovation and accelerating time-to-market for new features.



**Characteristics of Microservices**

* **Autonomous**
  + **Each component service in a microservices architecture** can be **developed, deployed, operated, and scaled without affecting the functioning of other services**. Services do not need to share any of their code or implementation with other services. Any **communication between individual components happens via well-defined APIs**.
* **Specialized**
  + **Each service is designed for a set of capabilities and focuses on solving a specific problem**. If developers contribute more code to service over time and the service becomes complex, it can be broken into smaller services.

**Benefits of Microservices**

* **Agility**
  + Microservices **foster an organization of small, independent teams that take ownership of their services**. Teams act within a small and well-understood context and are **empowered to work more independently and more quickly**. This **shortens development cycle times** and **benefits significantly from the aggregate throughput** of the organization.
* **Flexible Scaling**
  + Microservices **allow each service to be independently scaled to meet the demand** for the application feature it supports. This enables teams to **right-size infrastructure needs**, **accurately measure the cost of a feature**, and maintain availability if a service experiences a spike in demand.
* **Easy Deployment**
  + Microservices **enable continuous integration and continuous delivery**, making it **easy to try out new ideas and to roll back if something doesn’t work**. The low cost of failure enables experimentation, makes it easier to update code and accelerates time-to-market for new features.
* **Technological Freedom**
  + Microservices architectures **don’t follow a “one size fits all” approach**. Teams have the **freedom to choose the best tool** to solve their specific problems. As a consequence, teams building microservices can choose the best tool for each job.
* **Reusable Code**
  + Dividing software into **small, well-defined modules** **enables teams to use functions for multiple purposes**. A service written for a certain function can be **used as a building block for another feature**. This allows an application to bootstrap off itself, as developers can create new capabilities without writing code from scratch.
* **Resilience**
  + **Service independence increases an application’s resistance to failure**. In a monolithic architecture, if a single component fails, it can cause the entire application to fail. With microservices, **applications handle total service failure by degrading functionality** and not crashing the entire application.