ONE PAGE SUMMARY

IT & Software › Software Architecture › Scalable Architecture

Introduction to Scalable Architecture

Mastering Scalable Architecture for Future-Ready Applications

DEFINITION

> Scalable architecture is a design approach that allows software systems to handle increased loads efficiently. It ensures that applications can grow and adapt without performance loss.

KEY CONCEPTS

Horizontal Scaling

Adding more machines or instances to handle increased load, distributing traffic evenly.

Vertical Scaling

Increasing the resources (CPU, RAM) of existing machines to improve performance.

Load Balancing

Distributing incoming network traffic across multiple servers to ensure no single server is overwhelmed.

Microservices

Breaking applications into smaller, independent services that can be developed and scaled individually.

Caching

Storing frequently accessed data in memory to reduce load times and database queries.

EXAMPLES

→ A web application that uses multiple servers to manage user requests.

→ An e-commerce site that scales up during holiday sales.

→ A video streaming service that adjusts quality based on user bandwidth.

MEMORY TIPS

★ Think of scaling like adding more lanes to a highway to handle more cars.

★ Remember 'microservices' as tiny services working together like a team.

★ Use 'load balancing' as a metaphor for a waiter distributing food evenly among tables.

COMMON MISTAKES

✗ Confusing horizontal and vertical scaling.

✗ Neglecting to implement load balancing in high-traffic scenarios.

✗ Overlooking the importance of caching for performance.

QUICK RECAP

Scalable architecture enables applications to grow without sacrificing performance. Key strategies include horizontal and vertical scaling, load balancing, microservices, and caching. Understanding these concepts is crucial for designing robust systems.