ONE PAGE SUMMARY

IT & Software › Full Stack Development › Performance Tuning

Introduction to Performance Tuning

Boost Your Application's Speed with Performance Tuning Techniques

DEFINITION

> Performance tuning involves optimizing software applications to run faster and more efficiently. It focuses on improving speed, responsiveness, and resource usage.

KEY CONCEPTS

Profiling

Profiling helps identify bottlenecks in code by measuring execution time and resource usage.

Caching

Caching stores frequently accessed data in memory to reduce load times and improve performance.

Database Optimization

Optimizing database queries and structures can significantly enhance application speed and efficiency.

Load Balancing

Distributing workloads across multiple servers ensures no single server is overwhelmed, improving overall performance.

Code Refactoring

Refactoring involves restructuring existing code to improve readability and performance without changing its functionality.

EXAMPLES

→ Implementing a caching layer for frequently accessed API responses.

→ Using a load balancer to distribute traffic across multiple web servers.

→ Refactoring a slow database query to use indexes for faster retrieval.

MEMORY TIPS

★ Remember 'P-C-D-L-C' for Profiling, Caching, Database optimization, Load balancing, Code refactoring.

★ Think of 'Cache' as a 'Quick Access' for faster retrieval.

★ Use the acronym 'BOLT' for Bottlenecks, Optimization, Load, Tuning.

COMMON MISTAKES

✗ Neglecting to profile before making changes.

✗ Overusing caching, leading to stale data.

✗ Ignoring database indexing which can slow down queries.

QUICK RECAP

Performance tuning is essential for optimizing application speed and resource usage. Key techniques include profiling, caching, and database optimization. Avoid common mistakes like neglecting profiling and overusing caching.