ONE PAGE SUMMARY

IT & Software › Microservices › Saga Patterns

Introduction to Saga Patterns

Mastering Saga Patterns for Reliable Microservices Coordination

DEFINITION

> Saga Patterns are a way to manage distributed transactions in microservices. They ensure data consistency across services without locking resources.

KEY CONCEPTS

Choreography

Services communicate with each other through events, each service knows what to do next.

Orchestration

A central coordinator directs the saga, managing the sequence of service calls.

Compensation

If a step fails, compensation actions are triggered to roll back previous steps.

Timeouts

Each step in a saga can have time limits to prevent long-running transactions.

Eventual Consistency

Data will become consistent over time, rather than immediately, across all services.

EXAMPLES

→ Order processing with payment and shipping services using choreography.

→ Booking a flight with multiple airline services coordinated by an orchestrator.

→ Updating inventory after a failed order with compensation actions.

MEMORY TIPS

★ Think 'Choreography' as a dance where services know their moves.

★ Remember 'Orchestration' as a conductor leading an orchestra.

★ Use 'Compensation' like a refund to fix mistakes in transactions.

COMMON MISTAKES

✗ Confusing choreography with orchestration in service communication.

✗ Neglecting to implement compensation actions for failed steps.

✗ Overlooking timeout settings, leading to stuck transactions.

QUICK RECAP

Saga Patterns help manage distributed transactions in microservices. They use choreography or orchestration to ensure data consistency, with compensation for failures. Remember to consider timeouts and eventual consistency.