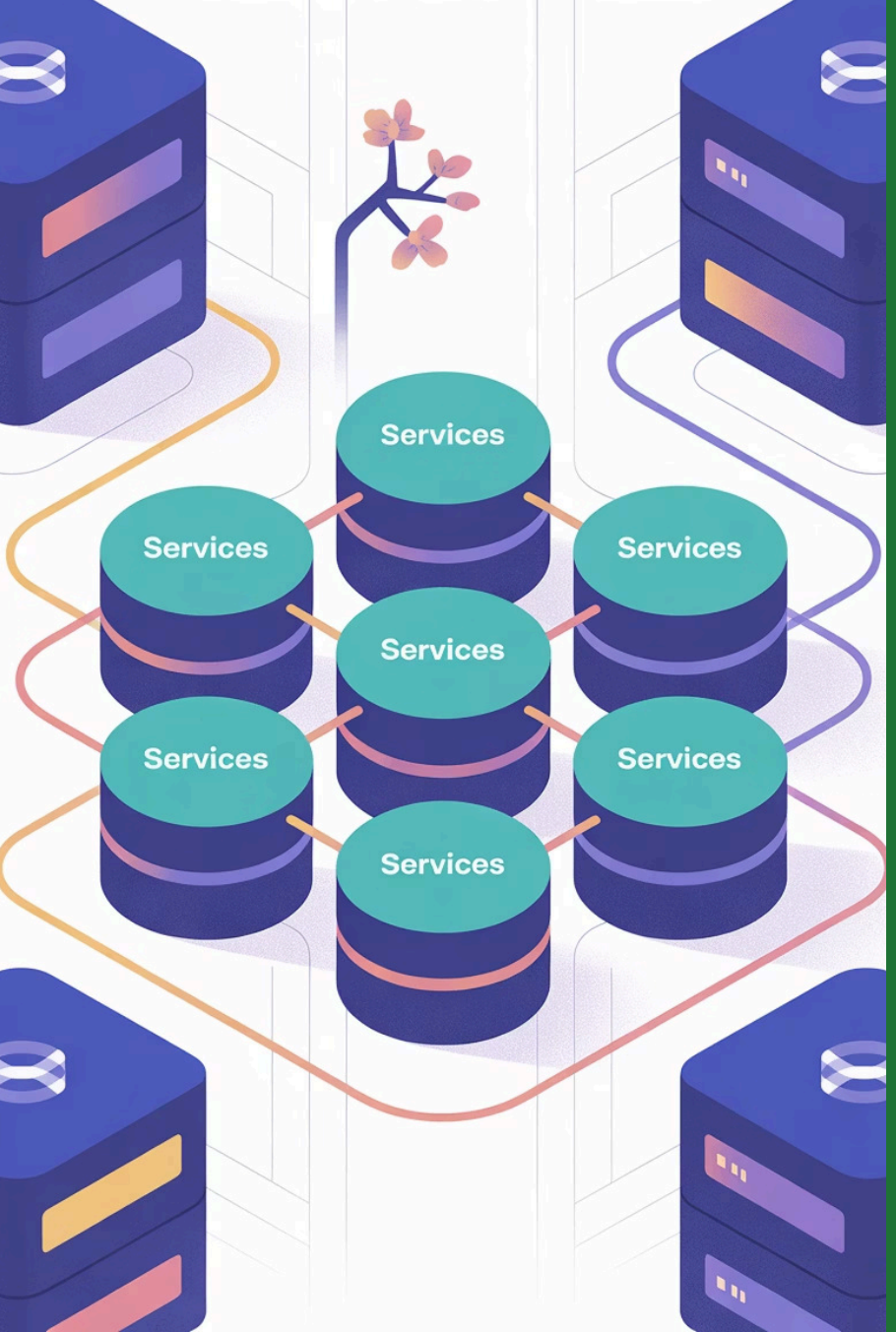
New Relic for Beginners: Monitoring Kubernetes Microservices with Spring Boot & React

Master observability for your cloud-native applications with comprehensive monitoring, tracing, and real-time insights across your entire stack.



Chapter 1

The Challenge of Observability in Modern Microservices

Why Monitoring Kubernetes Microservices is Hard

Dynamic Infrastructure

Containers spin up and down rapidly across nodes, making traditional monitoring approaches inadequate for tracking ephemeral workloads.

Service Complexity

Multiple interconnected services like Spring Boot backends and React frontends create intricate dependency chains that are difficult to trace.

Cascading Failures

Performance issues propagate quickly across services and infrastructure layers, making root cause analysis challenging without proper visibility.

Real-World Impact: When Monitoring Fails

01

Traffic Surge Hits

E-commerce site faces Black Friday peak traffic without adequate monitoring visibility.

02

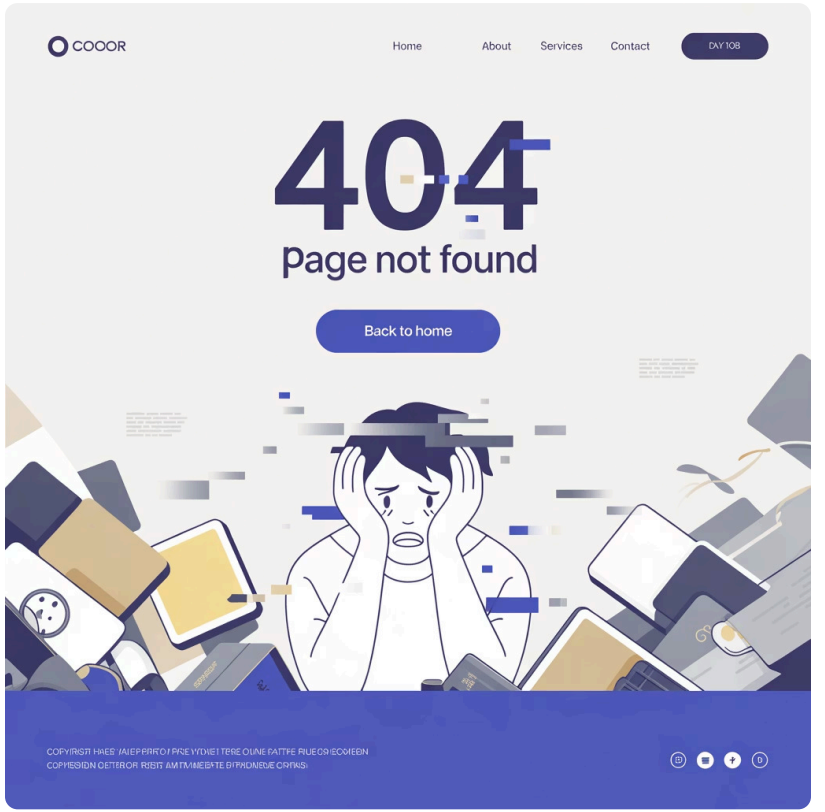
Performance Degrades

Slow responses cascade through microservices as memory and CPU resources become overwhelmed.

03

System Crashes

Complete application failure results in lost revenue, customer frustration, and brand damage.



The cost of poor observability: Early detection and rapid resolution are critical for maintaining system reliability and customer trust.

Chapter 2

Introducing New Relic — Your Observability Ally

What is New Relic?

Unified Platform

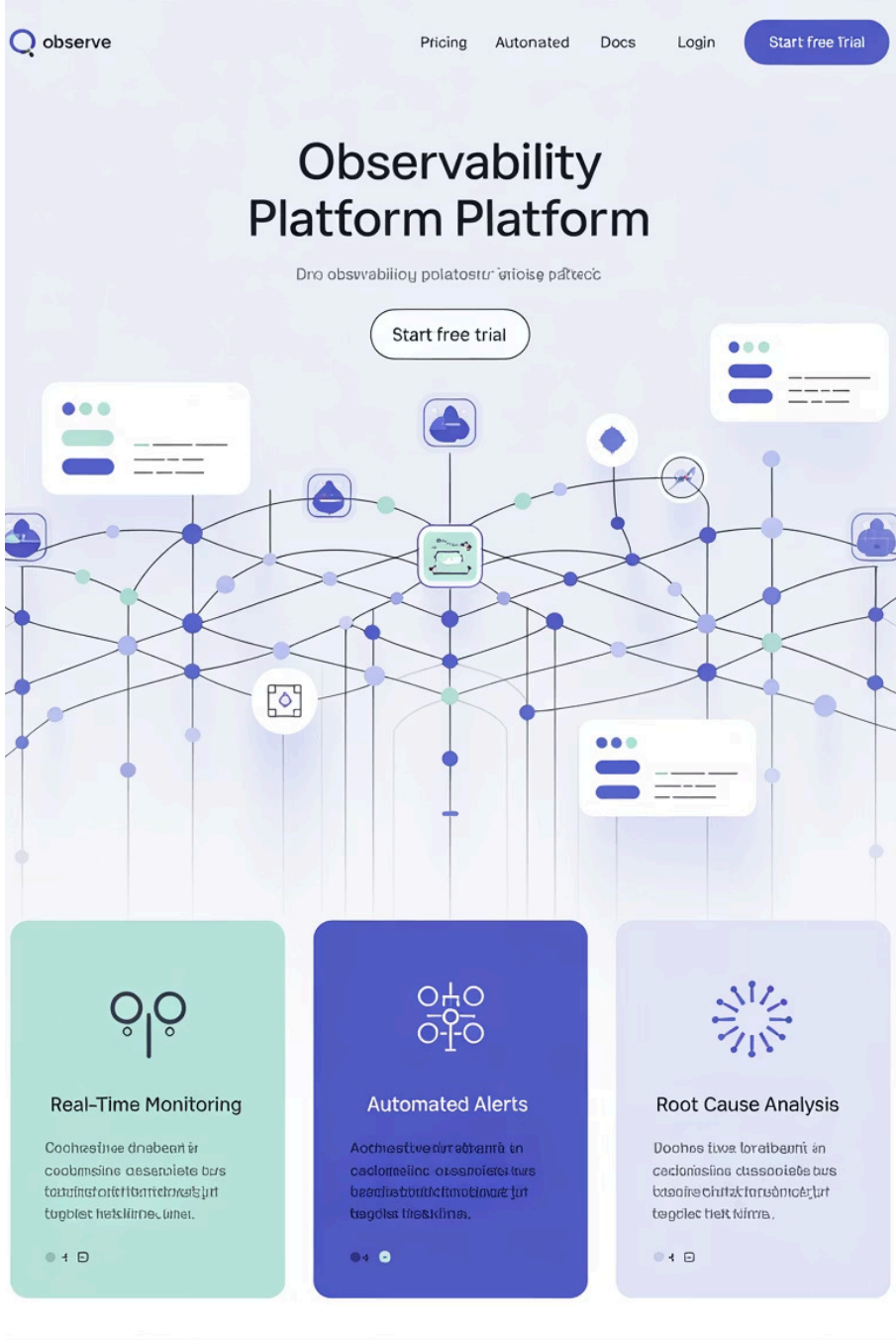
Single pane of glass for monitoring applications, infrastructure, and Kubernetes environments with comprehensive visibility.

Full-Stack Observability

Combines APM, infrastructure monitoring, logs, metrics, and distributed traces in one cohesive dashboard.

AI-Powered Insights

Intelligent alerts and anomaly detection accelerate troubleshooting and reduce mean time to resolution.



New Relic's One-Step Kubernetes Observability

Auto-Instrumentation

Automatically instruments Kubernetes workloads with APM agents, eliminating manual configuration overhead.

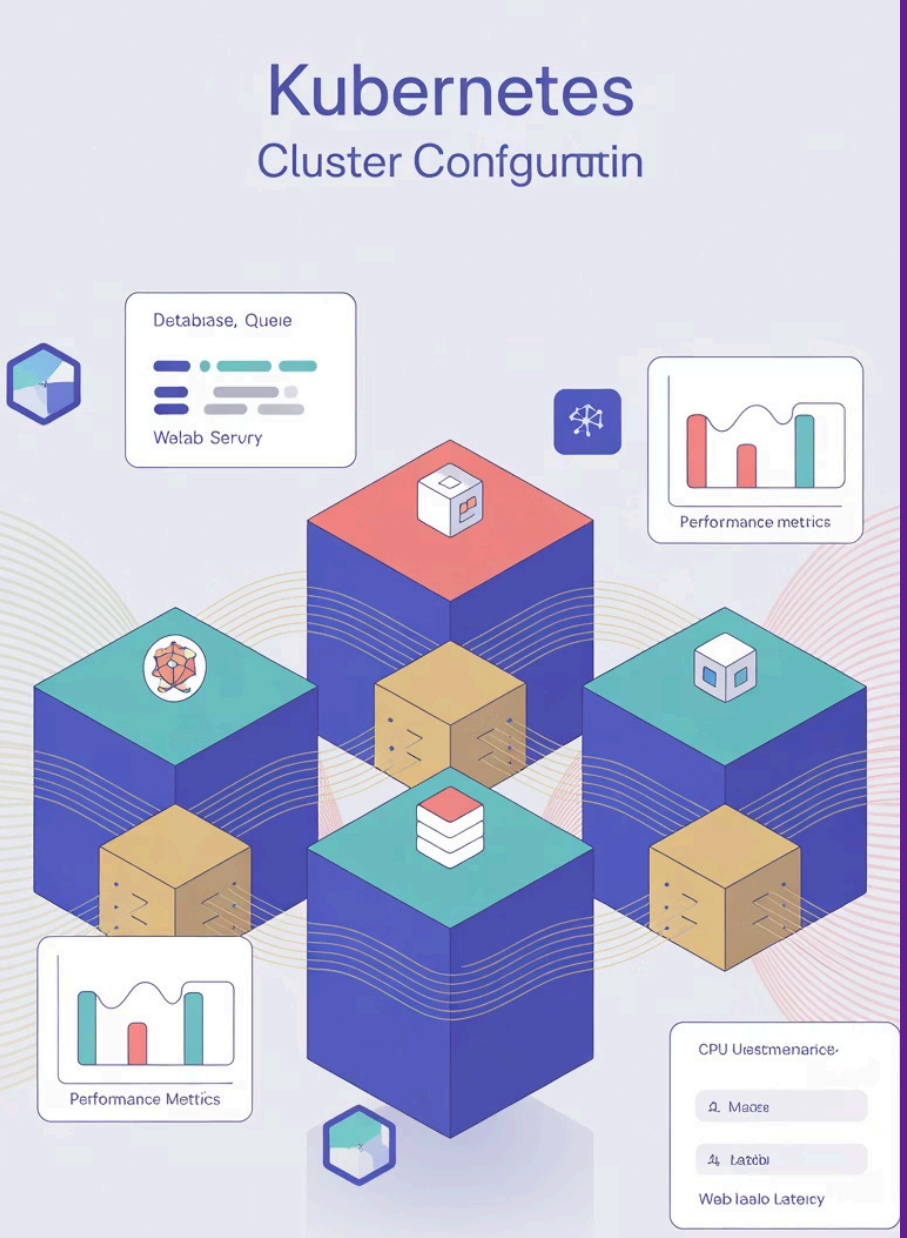
Cluster Health Monitoring

Provides unified visibility into cluster health, pod performance, and application metrics in real-time.

Open Standards Support

Native integration with OpenTelemetry and Prometheus metrics for maximum compatibility and flexibility.

New Relic dashboard providing comprehensive visibility into Kubernetes cluster health, pod performance, and microservices interactions



Chapter 3

Setting Up New Relic for Your Kubernetes Microservices

Step 1: Sign Up & Prepare Your Environment

1

New Relic Account

Create your New Relic account and obtain your license key from the account settings dashboard.

2

Kubernetes Cluster

Ensure your Kubernetes cluster is running (minikube for local development or cloud providers like EKS, GKE, AKS).

3

Management Tools

Install kubectl for Kubernetes management and Helm package manager for streamlined deployments.

Step 2: Install New Relic Kubernetes Integration

1

Deploy Operator

Use New Relic Kubernetes Operator for simplified deployment and automated upgrades across your cluster.

2

Helm Installation

Add New Relic Helm repository and deploy the operator with simple helm commands for streamlined setup.

3

Automated Modules

Operator automatically manages Pixie, Fluentbit, and Prometheus modules for comprehensive observability.

Step 3: Instrument Your Spring Boot Microservices

1

Add Dependency

Include New Relic Java agent dependency in your Spring Boot project using Maven or Gradle build configuration.

2

Configure Agent

Set up newrelic.yml configuration file with your application name and New Relic license key for proper identification.

3

Launch Application

Start your Spring Boot application with -javaagent JVM parameter pointing to the New Relic agent JAR file.

Step 4: Monitor Your React Frontend

Browser Monitoring

Add New Relic Browser monitoring snippet to your React application for frontend performance tracking.

User Experience

Track page load times, user interactions, JavaScript errors, and frontend performance metrics.

End-to-End Visibility

Correlate frontend performance with backend microservices for complete application insights.

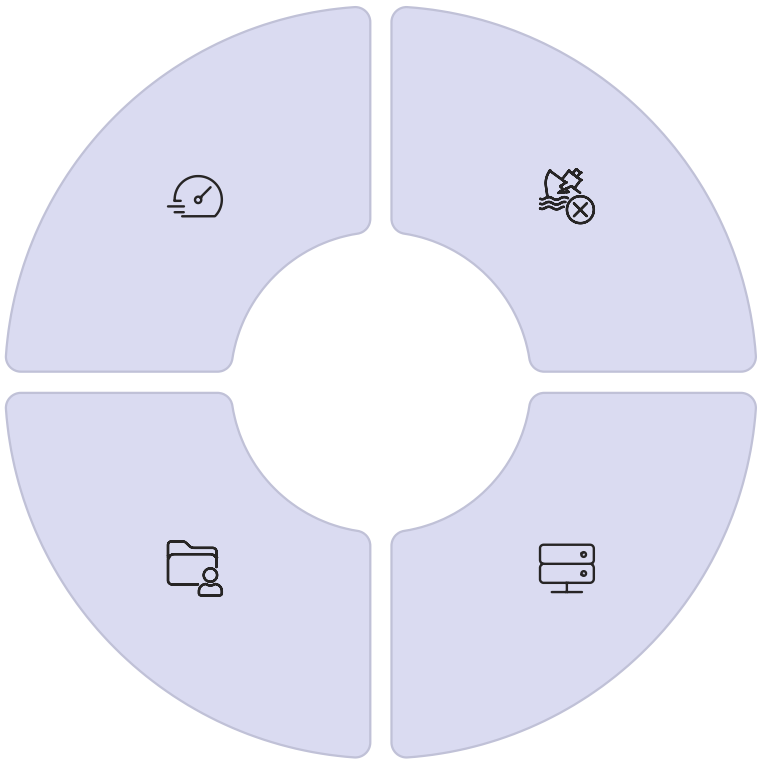
Chapter 4

Understanding Metrics & Traces in New Relic

Key Metrics to Watch in Spring Boot Microservices

Request Performance

Monitor request throughput, response times, and latency patterns across your microservices endpoints.



Error Tracking

Track error rates, exception traces, and failure patterns to identify problematic code paths quickly.

External Dependencies

Analyze database query performance and external service call latency for bottleneck identification.

JVM Health

Monitor JVM metrics including CPU utilization, memory consumption, and garbage collection performance.

Micrometer & New Relic Integration

Automatic Collection

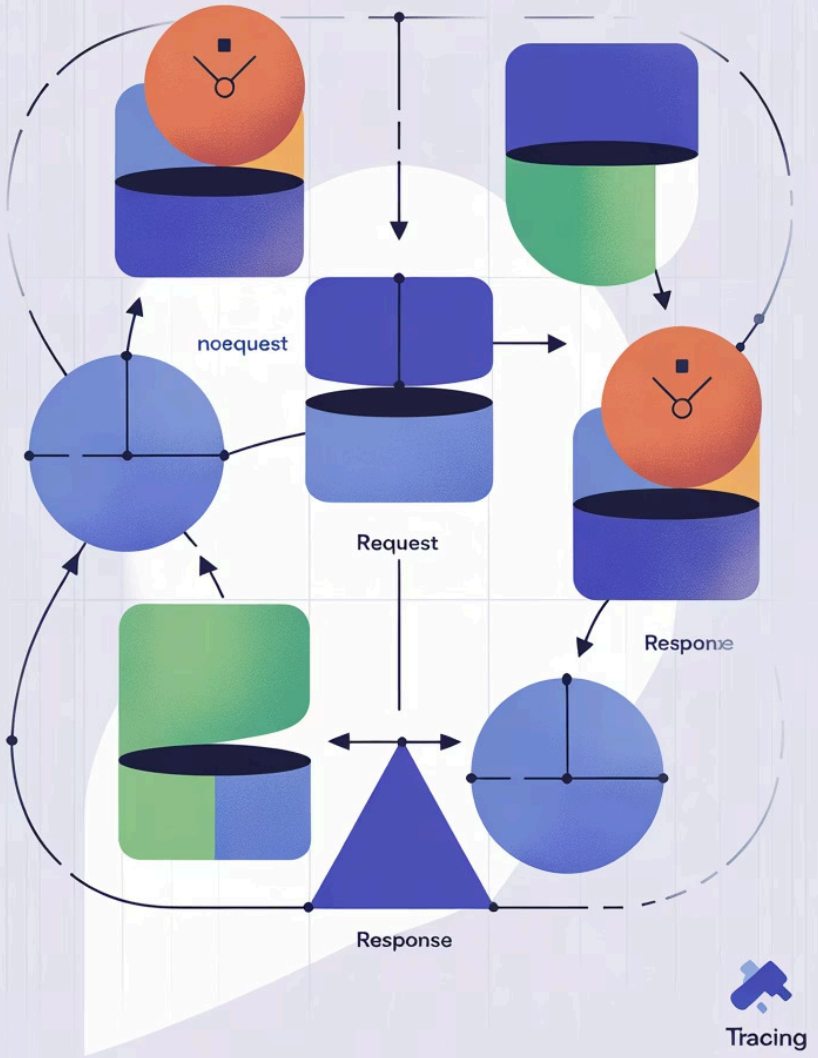
Spring Boot Micrometer automatically collects JVM and application metrics without additional configuration.

Registry Integration

New Relic Micrometer registry seamlessly sends collected metrics to New Relic One platform.

Custom Metrics

Create custom metrics and detailed performance insights tailored to your application's specific needs.



Distributed Tracing Across Kubernetes Microservices

Request Flow Tracking

Trace requests as they flow through Spring Boot services and React frontend for complete visibility.

Bottleneck Identification

Identify performance bottlenecks and error points in complex service interaction chains quickly.

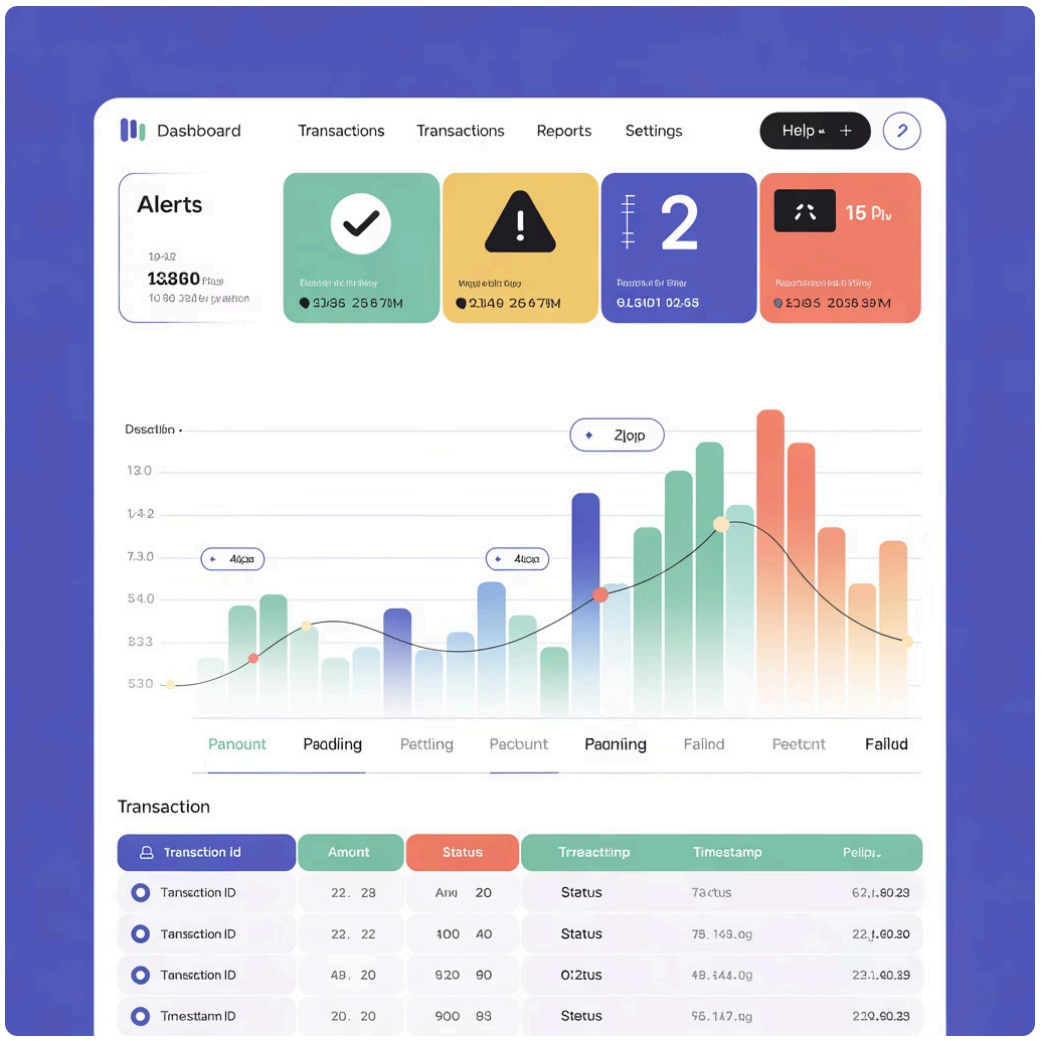
Service Visualization

Visualize service maps and transaction traces in New Relic dashboard for better understanding.

Chapter 5

Real-World Example & Best Practices

Case Study: Fintech Payment Microservice Monitoring



1 Integration Setup

Integrated New Relic Java agent into Spring Boot payment processing microservice for comprehensive monitoring.

2 Performance Tracking

Monitored transaction processing times, database query performance, and external payment gateway latency.

3 Proactive Alerting

Configured intelligent alerts for error rate spikes and slow response times, reducing system downtime by 40%.

40%

Downtime Reduction

Significant improvement in system reliability

2.5s

Average Response Time

Optimized payment processing performance

99.9%

System Uptime

Enhanced availability and customer satisfaction

Conclusion: Start Your New Relic Journey Today

Simplified Observability

New Relic streamlines Kubernetes microservices monitoring for Spring Boot and React applications with minimal setup effort.

Accelerated Troubleshooting

One-step instrumentation and AI-powered insights dramatically reduce time to identify and resolve performance issues.

Reliable Applications

Empower your development and operations teams to deliver high-performance, reliable applications with confidence.

Ready to transform your monitoring strategy? Visit newrelic.com and explore comprehensive documentation for Kubernetes and Java agent integration!