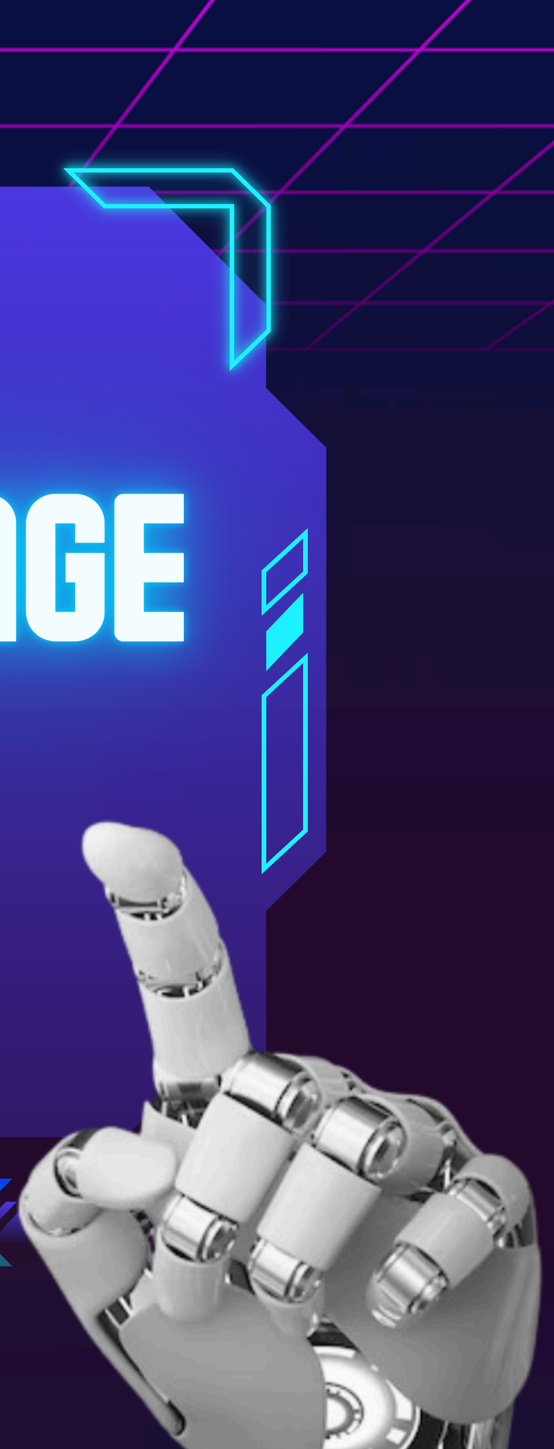
MONITORING CPU USAGE

USING JAVA FRAMEWORK



WHY MONITORY USAGE?

- Monitoring CPU usage is essential to optimize performance, detect bottlenecks, improve resource utilization, ensure application reliability, and enhance user experience. It helps maintain smooth operations and prevents system slowdowns or crashes.

- Identifying and resolving bottlenecks helps optimize performance by addressing resource-heavy processes and eliminating inefficiencies.

- Ensuring system reliability involves monitoring CPU usage to detect issues early, prevent downtime, and maintain consistent performance.

TOOLS MONITORING

SPRING BOOT ACTUATOR PROVIDES BUILT-IN ENDPOINTS TO MONITOR APPLICATION METRICS, HEALTH CHECKS, AND SYSTEM INFORMATION, ENABLING REAL-TIME INSIGHTS AND EFFICIENT MANAGEMENT.

JMX (JAVA MANAGEMENT EXTENSIONS) PROVIDES TOOLS FOR JVM-LEVEL MONITORING, ALLOWING DEVELOPERS TO TRACK METRICS LIKE MEMORY USAGE, THREAD ACTIVITY, AND CPU PERFORMANCE IN REAL TIME.

MICROMETER IS A METRICS COLLECTION LIBRARY THAT INTEGRATES WITH VARIOUS MONITORING SYSTEMS, PROVIDING A UNIFIED API TO TRACK APPLICATION PERFORMANCE AND RESOURCE USAGE.

PROMETHEUS COLLECTS AND STORES APPLICATION METRICS, WHILE GRAFANA VISUALIZES THEM THROUGH CUSTOMIZABLE DASHBOARDS. TOGETHER, THEY ENABLE REAL-TIME MONITORING, ANALYSIS, AND ALERTING FOR PERFORMANCE ISSUES.

WORKFLOW

HOW IT WORKS?

- Integrating Micrometer in your Java app involves adding the Micrometer library to your project, configuring it to collect metrics, and exporting these metrics to a monitoring system like Prometheus.

- Exporting metrics to Prometheus involves exposing your application's metrics through an endpoint and configuring Prometheus to scrape data from it
- Use Grafana for dashboards and alerts

BENEFITS

1. **Real-time monitoring** involves continuously tracking system performance and resource usage (e.g., CPU, memory, network) to detect and address issues immediately, ensuring optimal application performance and uptime.

2. **Proactive** issue detection involves monitoring system metrics continuously to identify potential problems before they impact performance, allowing for timely intervention and preventing downtime or failures.

3. **Scalable** and **flexible** solutions are designed to efficiently manage growing workloads and adapt to changing needs, allowing systems to expand or adjust without compromising performance or reliability.

SUMMARY

- **Java frameworks** simplify monitoring by providing built-in tools and libraries (like Micrometer, Spring Boot Actuator, and JMX) that enable easy integration of real-time performance tracking, system health checks, and metrics collection, making it easier to maintain and optimize applications.
- **Enhancing application** reliability involves using monitoring tools to identify and address potential issues before they affect performance, ensuring consistent uptime, faster issue resolution, and overall stability of the application.
- **Ensuring consistent** performance involves continuously monitoring system resources, identifying bottlenecks, and optimizing application behavior to maintain smooth and reliable operation over time, even under varying loads.

GUEVARRA, 11 ACACIA

THANK YOU