

Python para Producción



Observabilidad en Aplicaciones Web Críticas  

Pythonistas GDL - Agosto 2025

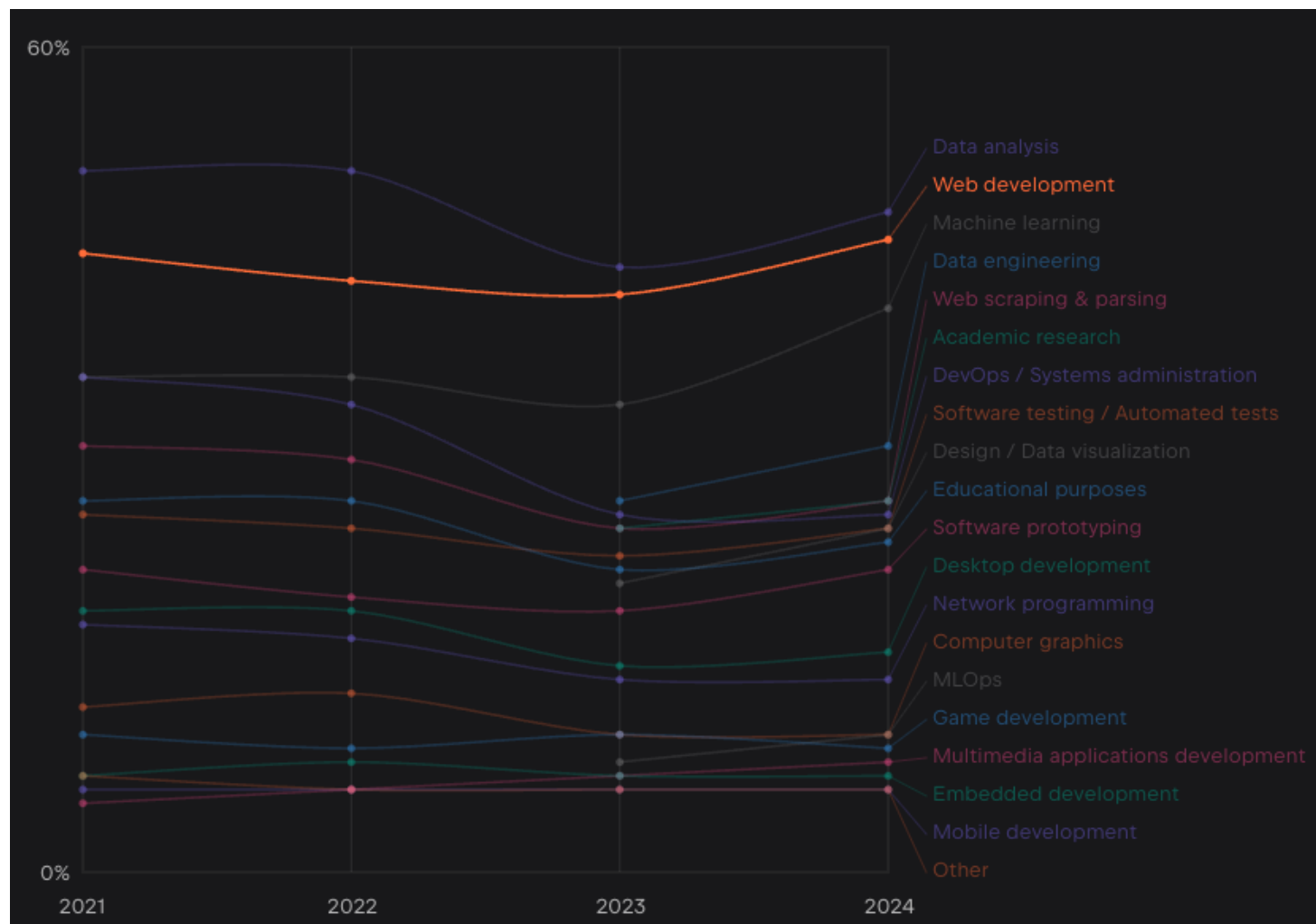


Acerca de mí



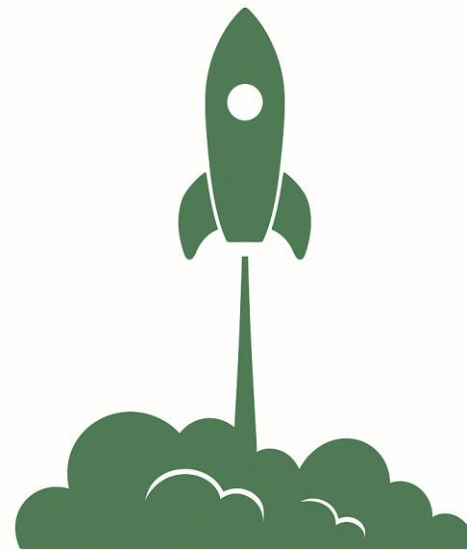
- ▶ Juan Carlos Pérez
- ▶ Principal Software Engineer @ LexisNexis Risk Solutions
- ▶ ESCOM - IPN, 2014

Python y el desarrollo web



*46% - Python Developers Survey 2024 Results by JetBrains

Producción es
solo el inicio
del viaje





No existen
los sistemas
perfectos




Las pruebas
nunca serán
suficientes








¿Qué es una aplicación web crítica?

Sistema cuya falla impacta continuidad del negocio, ingresos, seguridad de personas o cumplimiento regulatorio.

► Cómo identificarla:

- Alto costo por minuto de caída 
- Manejo de datos sensibles/PII/PHI 
- Soporta SLAs con clientes 

► Requisitos típicos:

- Alta disponibilidad y tolerancia a fallos 
- Consistencia/atomicidad de datos, backups y DRP 
- Seguridad, escalabilidad y observabilidad end-to-end   

► Ejemplos:

- Pasarelas de pago  , Banca en línea  , SaaS B2B  , Historia clínica electrónica  



¿Qué es observabilidad? 👁👁

- ▶ Origen: teoría del control (Kalman, 1960) 📐
 - ▶ Propiedad matemática: si, a partir de las salidas, puedo reconstruir el estado interno en tiempo finito ⏱
- ▶ Capacidad de entender qué pasa y por qué en un sistema usando señales externas **sin cambiar código** 🚀
- ▶ Se logra recolectando, correlacionando y analizando datos acerca de la ejecución de código, del estado interno de los componentes del sistema y de la comunicación entre entidades

Monitoreo vs Observabilidad



Beneficios en aplicaciones web críticas

- ▶ Prevención de incidentes
- ▶ Respuesta rápida
 - ▶ Menor tiempo de detección (MTTD) y de recuperación (MTTR)
- ▶ Ayuda en la toma de decisiones
- ▶ Ayuda a la operación del sistema
- ▶ Mejora la experiencia de los usuarios -> \$\$\$\$
- ▶ Cumplimiento normativo
 - ▶ Permite auditar accesos y transacciones sistemas regulados
 - ▶ Ejemplos: NOM-024 en México /HIPAA  

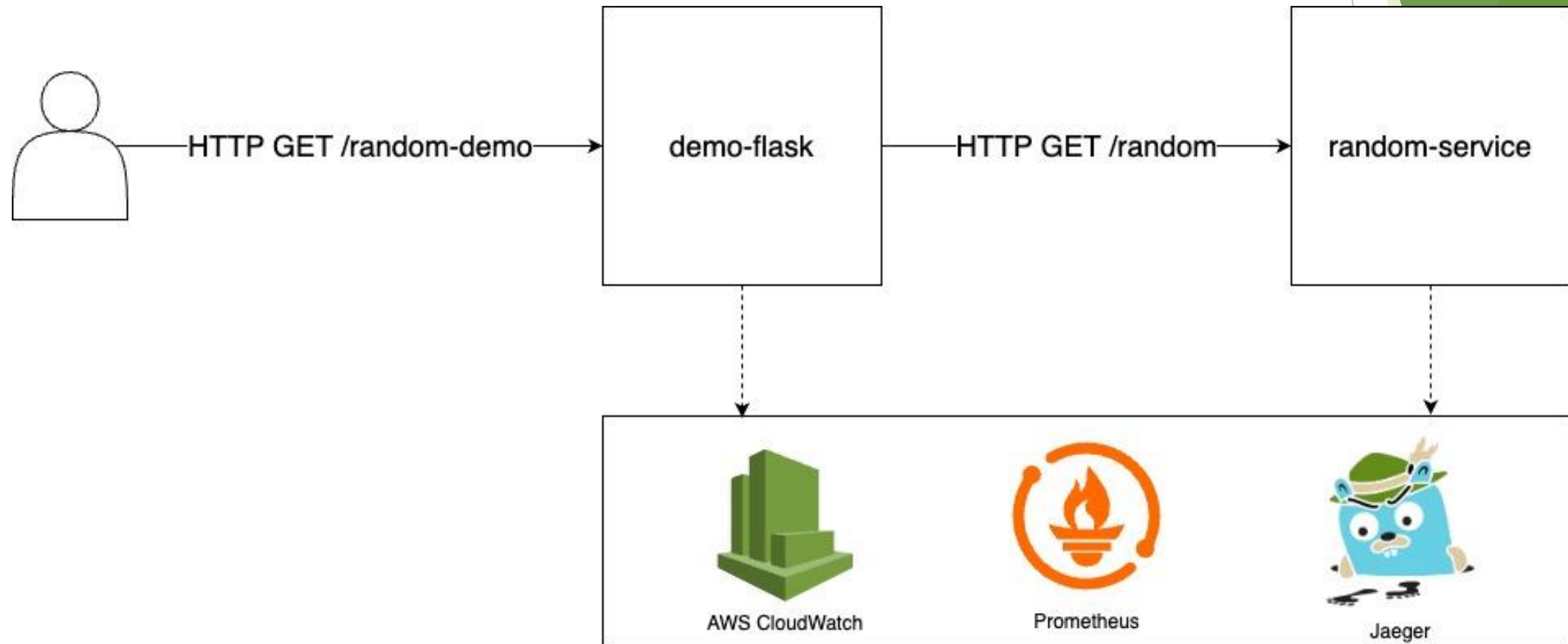
Pilares de la Observabilidad

	Logs	Métricas	Trazas
Propósito	Registro de eventos	Medir tendencias y rendimiento	Flujo de eventos
Retención	Medio	Largo	Corto a medio
Uso	Debugging, auditoria	Alertas	Diagnóstico de rendimiento
Datos	Texto libre o estructurado	Tipos: <ul style="list-style-type: none">• Contadores (counter)• Medidor (gauge)• Histograma (histogram)• Resumen (summary)	Conjunto de spans
Herramientas	AWS CloudWatch, Loki, Datadog Logs	Prometheus, Datadog, AWS CloudWatch Metrics	Jaeger, Zipkin, Datadog APM

Demo :D



Demo - Arquitectura



Logs en CloudWatch

Selection criteria

Select up to 50 log groups

demo-flask X

random-service X

Clear all

Show fewer chosen log groups

```
1 fields @timestamp, message, otelServiceName, otelTraceID, otelSpanID, httpbin_status, @logStream, @log
2 | filter otelTraceID = '20fa6c9d7ae896345cfd15711c15562'
3 | sort @timestamp desc
4 | limit 10000
```

Run query

Cancel

Save

Actions ▼

History

Logs Insights QL query can run for maximum of 60 minutes.

Complete.

Logs (6)

Visualization

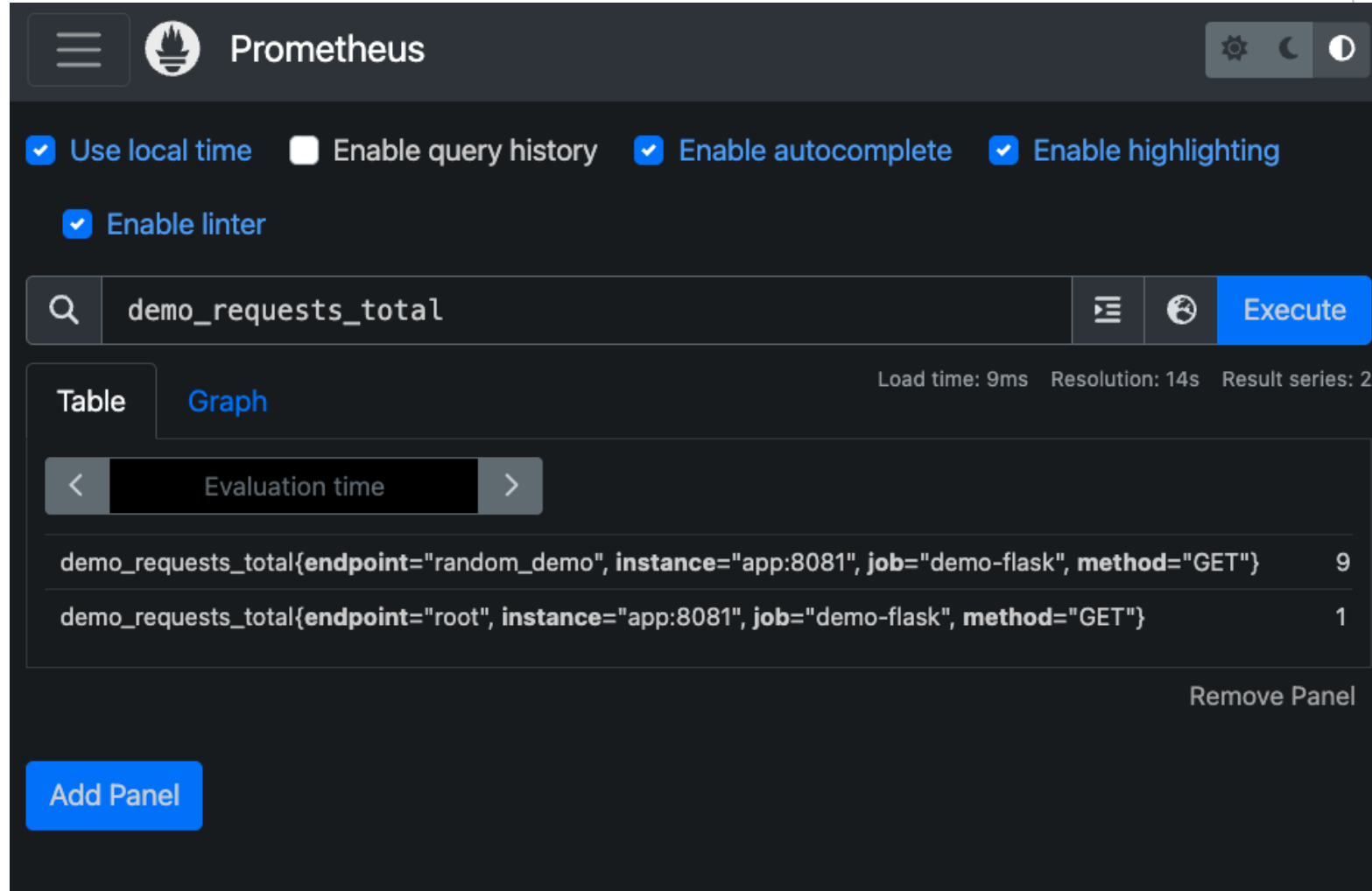
Logs (6)

Export

Showing 6 of 6 records matched ⓘ
16 records (5.3 kB) scanned in 1.2s @ 12 records/s (4.2 kB/s)

#	@timestamp	message	otelServiceName	otelTraceID	otelSpanID
▶ 1	2025-08-28T20:06:45.541-06:00	completed request	demo-flask	20fa6c9d7ae896345cfd15711c15562	cc0ec03f2e66ec...
▶ 2	2025-08-28T20:06:45.538-06:00	successfully called random service	demo-flask	20fa6c9d7ae896345cfd15711c15562	3e77c28498ec31...
▶ 3	2025-08-28T20:06:45.369-06:00	completed request	random-service	20fa6c9d7ae896345cfd15711c15562	00985982ae8ae5...
▶ 4	2025-08-28T20:06:45.368-06:00	generated random number	random-service	20fa6c9d7ae896345cfd15711c15562	d674a904f22eda...
▶ 5	2025-08-28T20:06:45.277-06:00	starting request	random-service	20fa6c9d7ae896345cfd15711c15562	00985982ae8ae5...
▶ 6	2025-08-28T20:06:45.263-06:00	starting request	demo-flask	20fa6c9d7ae896345cfd15711c15562	cc0ec03f2e66ec...

Métricas en Prometheus



The screenshot displays the Prometheus web interface. At the top, there's a header with the Prometheus logo and name, and settings icons. Below the header, there are several toggle switches for configuration: 'Use local time' (checked), 'Enable query history' (unchecked), 'Enable autocomplete' (checked), 'Enable highlighting' (checked), and 'Enable linter' (checked). The main query area contains the text 'demo_requests_total' and an 'Execute' button. Below the query area, there are tabs for 'Table' and 'Graph', with 'Table' currently selected. To the right of the tabs, it shows 'Load time: 9ms', 'Resolution: 14s', and 'Result series: 2'. Below the tabs, there's a table with two columns: the metric name and its value. The table contains two rows of data. At the bottom left, there's an 'Add Panel' button, and at the bottom right, there's a 'Remove Panel' link.

Prometheus

☒ Use local time ☐ Enable query history ☒ Enable autocomplete ☒ Enable highlighting

☒ Enable linter

demo_requests_total

Execute

Table Graph

Load time: 9ms Resolution: 14s Result series: 2

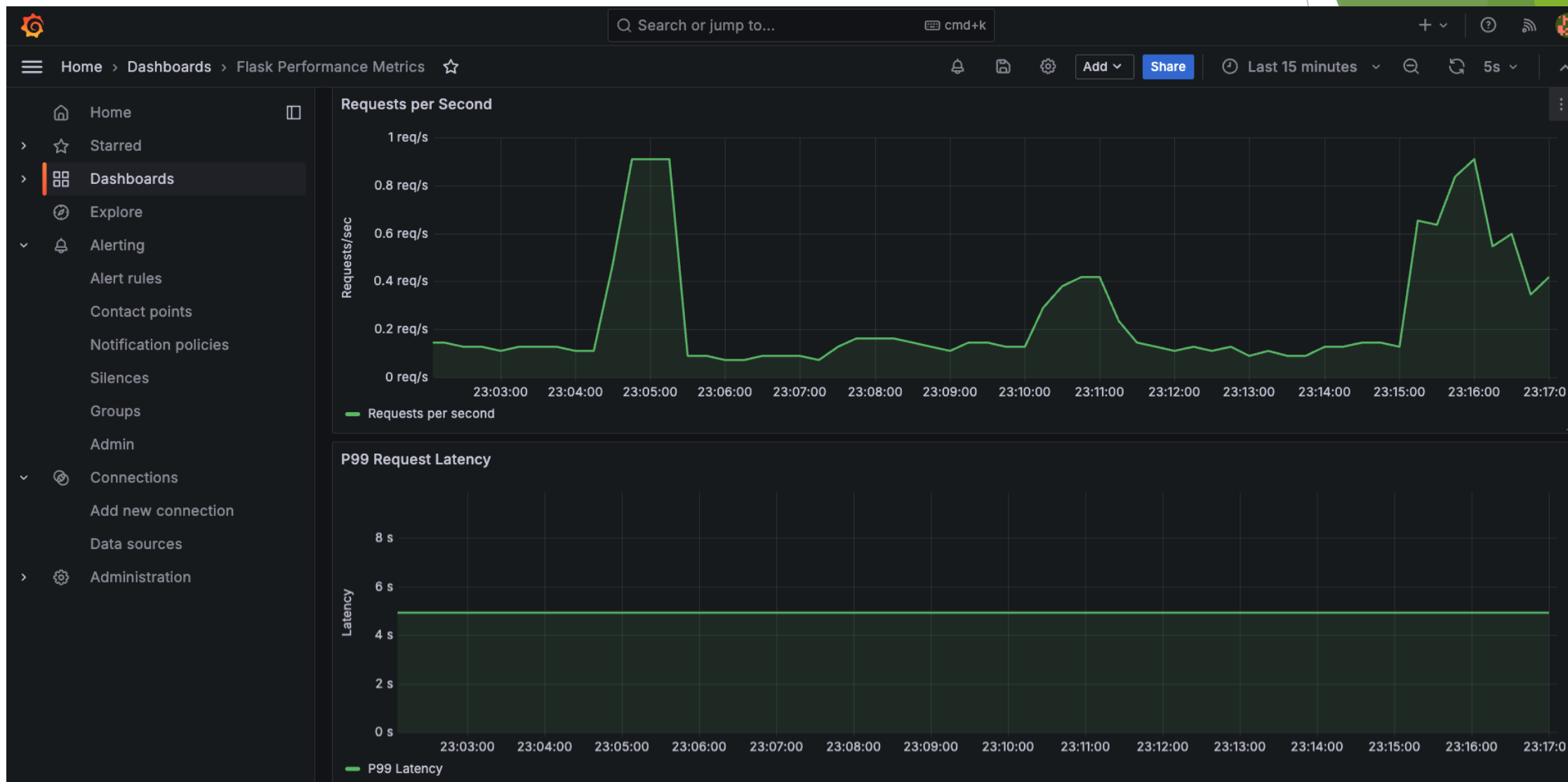
Evaluation time

demo_requests_total{endpoint="random_demo", instance="app:8081", job="demo-flask", method="GET"}	9
demo_requests_total{endpoint="root", instance="app:8081", job="demo-flask", method="GET"}	1

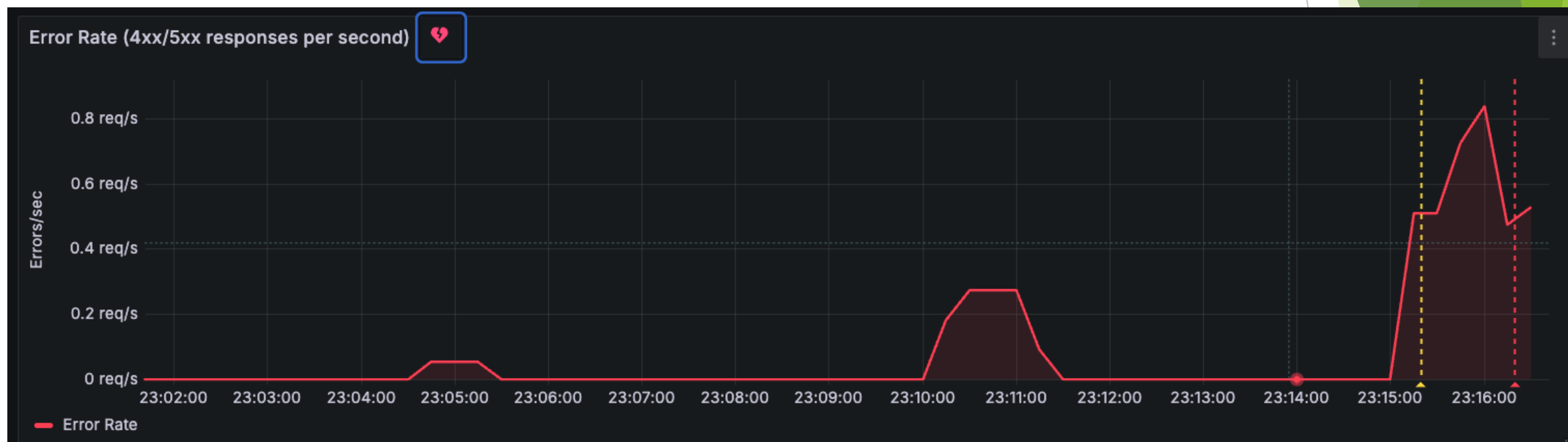
Remove Panel

Add Panel

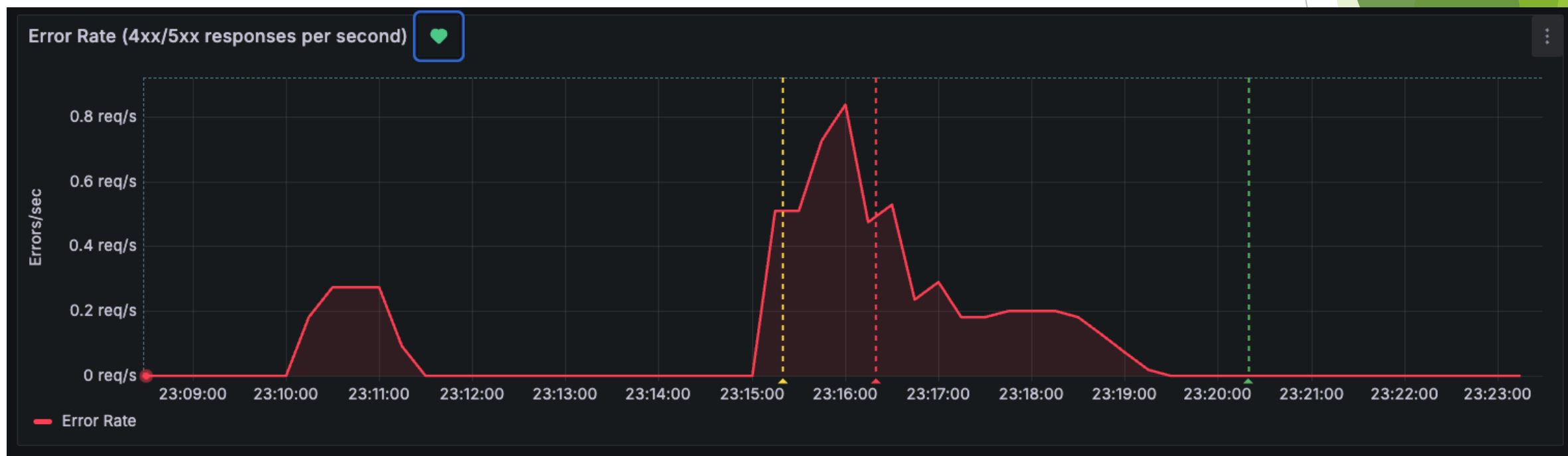
Dashboards en Grafana



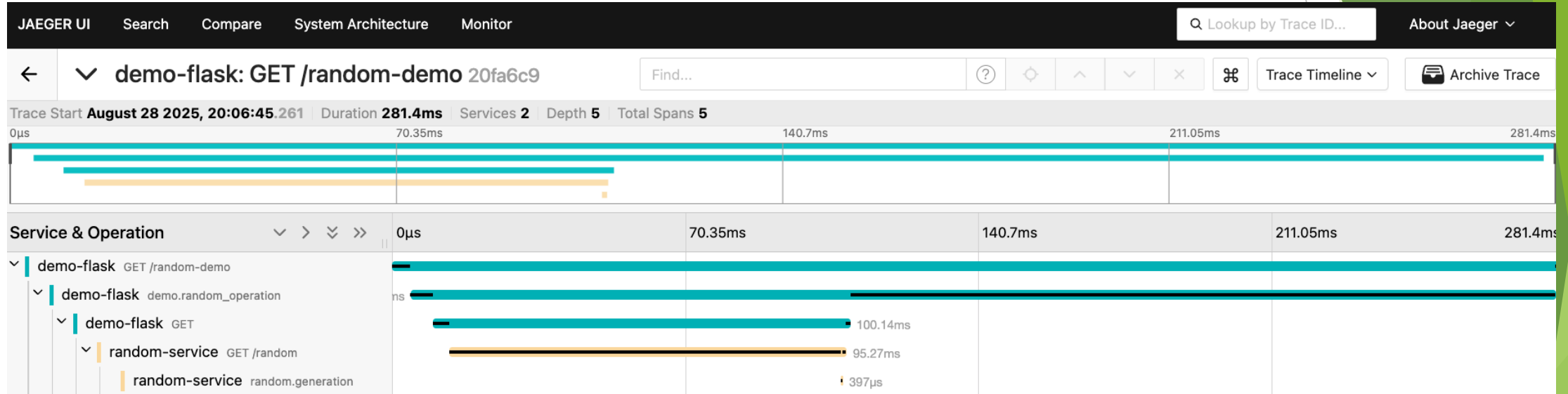
Alertas en Grafana



Alertas en Grafana



Trazas en Jaeger



¿Por dónde empezar?

- ▶ Confirma que tus logs sean estructurados y tengan el contexto correcto
- ▶ Instrumenta métricas técnicas y de negocio
- ▶ Si trabajas con microservicios, propaga traces
- ▶ Define niveles de servicio (SLO)
- ▶ Automatiza alertas y dashboards

Gracias 😊

Contacto

LinkedIn:

<https://www.linkedin.com/in/jcperez91/>

GitHub:

<https://github.com/jcperez>
