IES GONZALO
NAZARENO

GESTIÓN Y ANÁLISIS DE APLICACIONES EN KUBERNETES

MÉTRICAS, COBERTURA DE CÓDIGO E IC / DC

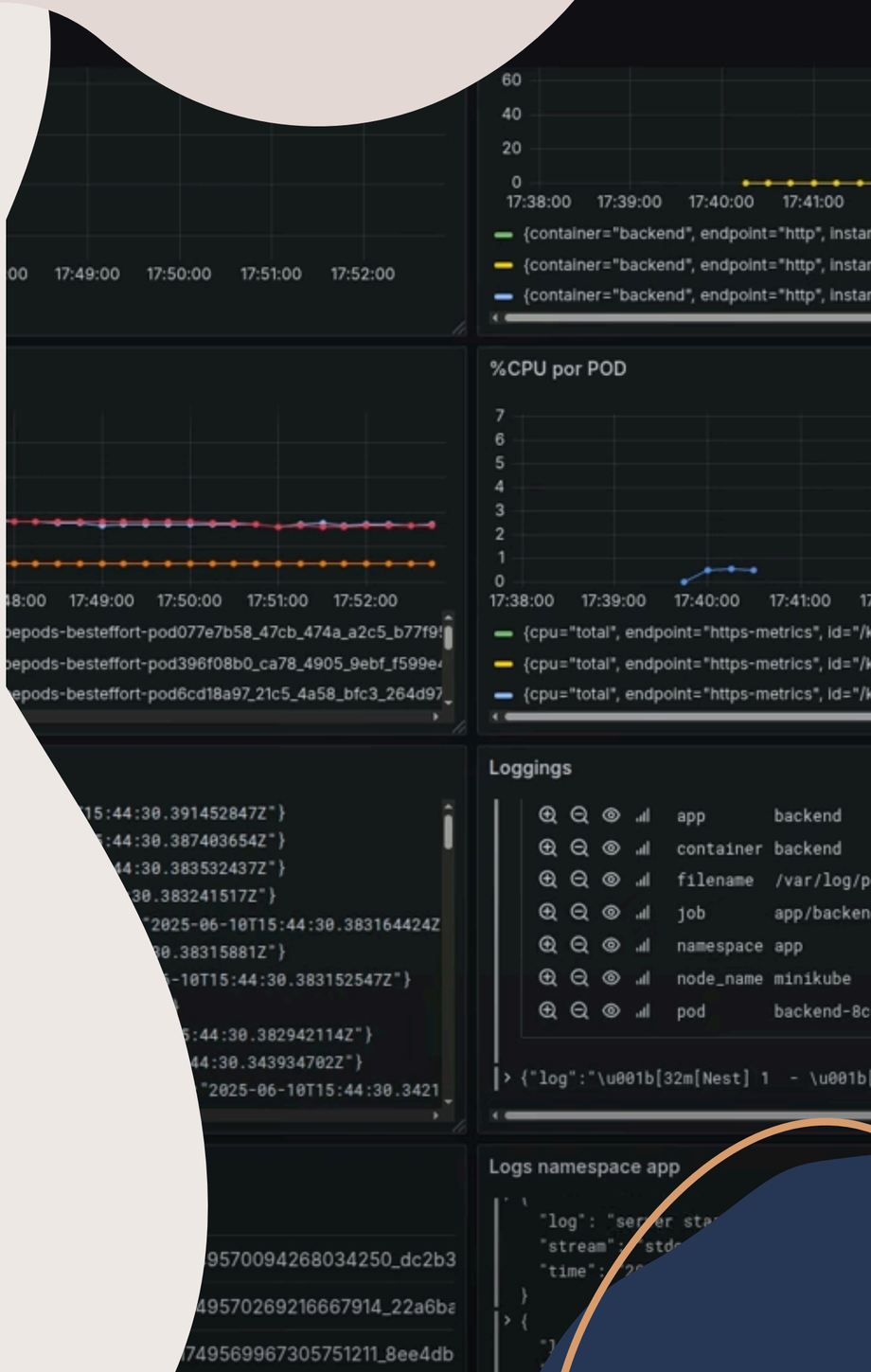
RAÚL HERRERA RUIZ

ÍNDICE

INTRODUCCIÓN
AL PROYECTO

En este proyecto podremos ver el despliegue de una aplicación con test unitarios y cobertura de código mediante github actions.

También veremos un stack de observabilidad desplegado en kubernetes para dicha aplicación.



FUNDAMENTOS TEÓRICOS

ORQUESTACIÓN Y DESPLIGUE

El crecimiento de las aplicaciones basadas en contenedores ha impulsado la necesidad de sistemas que automaticen su gestión, escalado y despliegue. La orquestación y la integración continua permiten mantener entornos consistentes y actualizados de forma eficiente.

OBSERVABILIDAD

En sistemas complejos, ya no basta con saber si algo “funciona”; es clave entender cómo lo hace. La observabilidad permite recolectar, analizar y visualizar información clave del sistema para anticiparse a fallos críticos de la aplicación.

TESTS Y COBERTURA DE CÓDIGO

Asegurar la calidad del software requiere comprobar no solo que funciona, sino que está bien construido. Los tests unitarios y el análisis de cobertura permiten validar el comportamiento del código y detectar posibles errores.

SOFTWARE
USADO

1. DOCKER : EL ESTÁNDAR PARA CONTENEDORES
2. KUBERNETES : EL ORQUESTADOR POR ANTONOMASIA
3. PROMETHEUS : LA SOLUCIÓN PARA LAS MÉTRICAS
4. LOKI : GESTIÓN DE LOGS
5. GRAFANA : VISUALIZACIÓN DE MÉTRICA Y LOGS
6. JEST : FRAMEWORK DE TESTING
7. SONARQUBE : CONTROL DE COBERTURA DE CÓDIGO
8. GITHUB ACTION : AUTOMATIZACIÓN
9. CLOUDFLARE : EXPOSICIÓN DE LOCAL A PÚBLICO

EL ESCENARIO

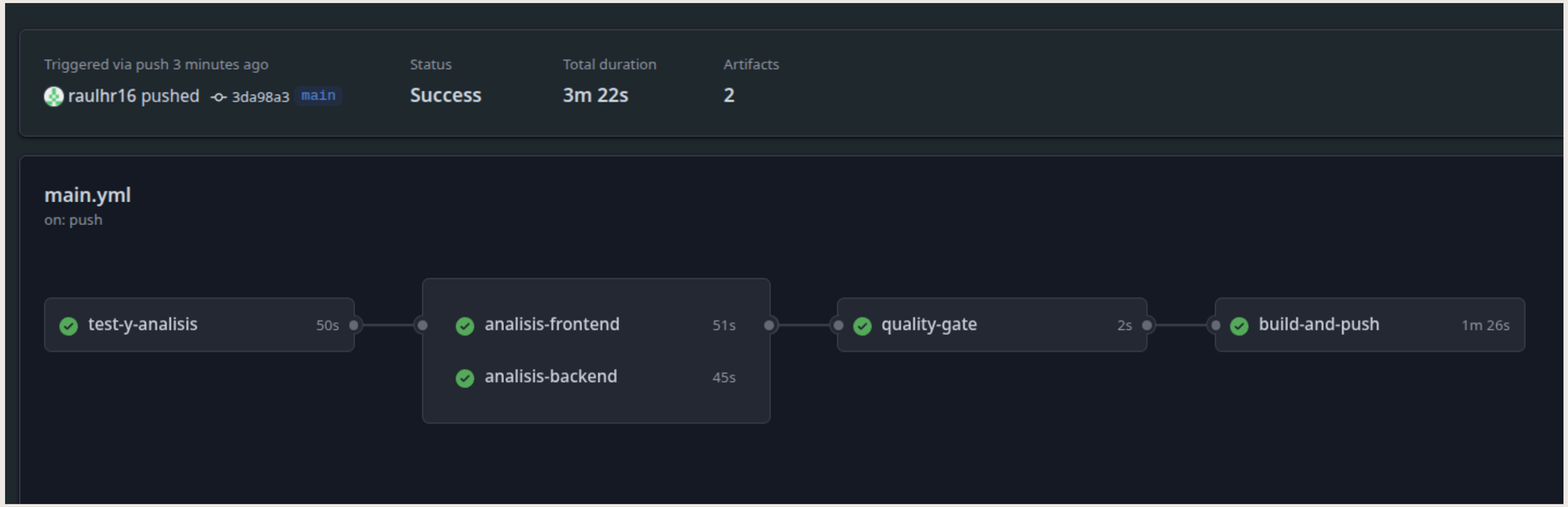
ENTORNO

En nuestro escenario contamos con una aplicación desplegada en kubernetes dentro del namespace app, dicha aplicación pasa tests unitarios con Jest y cobertura de código con SonarQube, el cual también tenemos desplegado en k8s dentro del namespace sonarqube. Una vez Github Actions ha corroborado que todo pasa bien sube dos imágenes a docker hub con un nuevo tag, una del frontend y otra del backend de nuestra app.

Luego tenemos un stack de monitoreo dentro de kubernetes con Prometheus, Loki y Grafana, dicho stack se encuentra dentro del namespace monitoreo, este stack nos sirve para comprobar las métricas y logs de la app.

MUESTRAS

GITHUB ACTIONS



MUESTRAS

SONAR

☆ app-back-raulhr PUBLIC — Passed

Last analysis: 3 hours ago • **561** Lines of Code • TypeScript

Security	Reliability	Maintainability	Hotspots Reviewed	Coverage	Duplications
A 0	A 0	A 8	E 0.0%	100%	0.0%

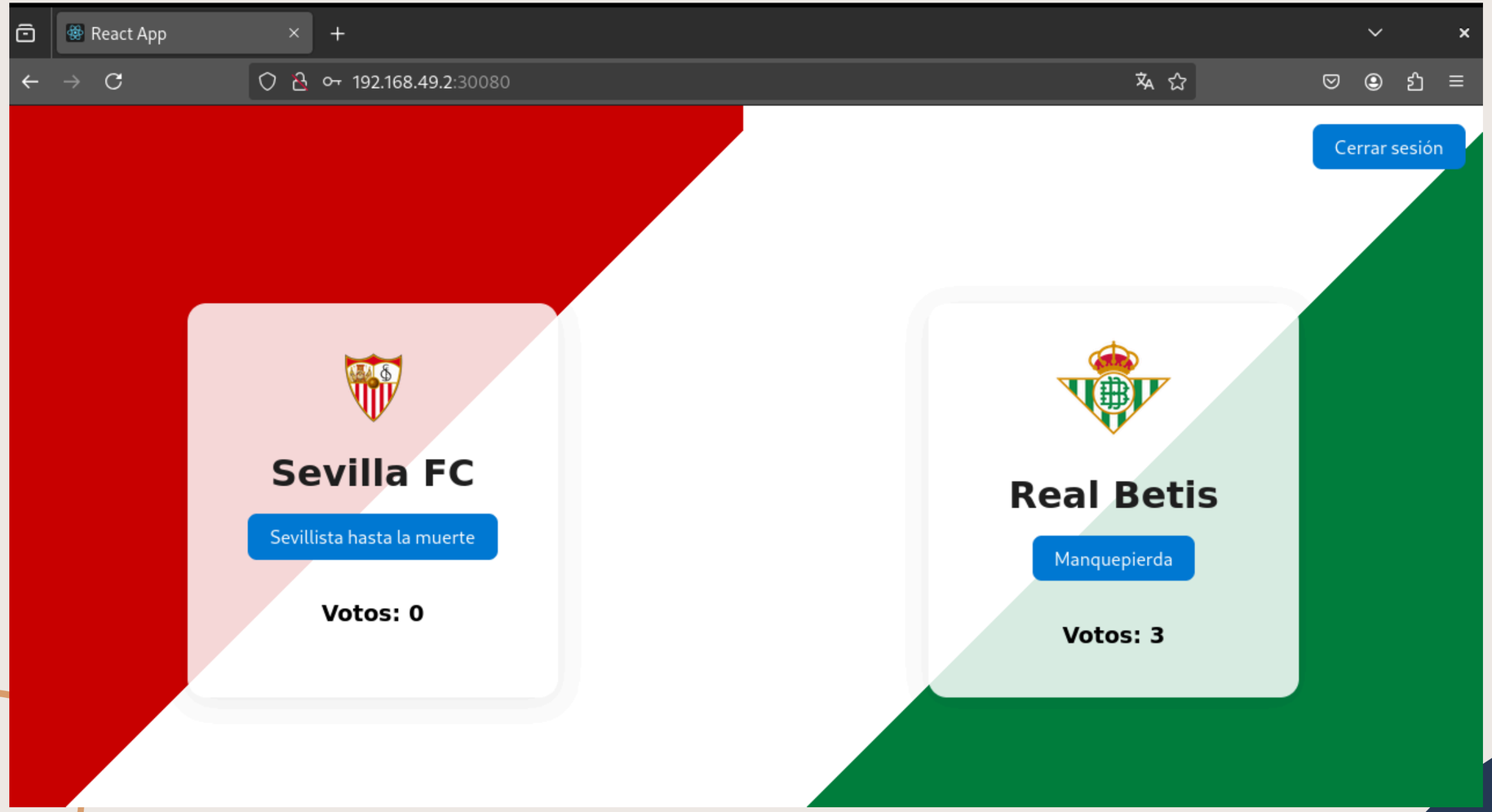
☆ app-front-raulhr PUBLIC — Passed

Last analysis: 3 hours ago • **359** Lines of Code • JavaScript, CSS

Security	Reliability	Maintainability	Hotspots Reviewed	Coverage	Duplications
A 0	B 6	A 5	A —	97.0%	0.0%

MUESTRAS

APP



MUESTRAS

GRAFANA

CONCLUSIÓN

Este proyecto me ha servido para entender mejor herramientas que ya uso, y también para profundizar sobre otras nuevas. Por ejemplo, apenas conocía SonarQube y me ha sorprendido lo útil que es para detectar errores en el código, sobre todo en temas de seguridad y calidad.

También me ha venido muy bien profundizar en el despliegue continuo y en cómo automatizarlo con GitHub Actions, ya que hoy en día es algo muy usado y próximamente tengo una migración a github actions en mi trabajo.

Por último, he aprendido la importancia de los logs y las métricas, ya que ayudan a detectar problemas antes de que afecten al sistema y permiten mantenerlo estable.

IES GONZALO
NAZARENO

MUCHAS GRACIAS

Por estos buenos dos años.

¡He aprendido mucho!

raulhr16