Arquitectura de Software en la Práctica
Obligatorio 2

Objetivos

Poner en práctica conceptos relacionados a la tecnología de computación en la nube y el diseño de sistemas orientado a microservicios, trabajando sobre un sistema que ya se encuentra en desarrollo.

Introducción

Tras el éxito de Centinela se decide exponerla como plataforma fuera de la integración con EnvíosYa. Actualmente existe interés de 16 compañías más de las cuales 10 son Uruguayas, 3 Argentinas y 3 de Estados Unidos.
Al relevar el probable uso de la plataforma por sus potenciales clientes se sabe que:

- Cada compañía va a registra entre 1000 y 1500 errores por día
- Registran alrededor de 200 usuarios desarrolladores cada compañía
- 7 de las compañías registran un 90% de sus errores entre las 17:00 y 20:00 horas (horario de Uruguay)

Dado que Centinela aspira a ser la plataforma SaaS líder en gestión de errores de software se compromete a:

- SLA de 99.9%
- Ninguna funcionalidad provistas debe afectar la disponibilidad del registro de errores

Desarrollo

El proyecto tecnológico a desarrollar se deberá ajustar a las características de una aplicación *cloud native*. Dado que el equipo posee ahora suficientes conocimientos del dominio del problema a solucionar, **se deberá refactorizar el sistema hacia una arquitectura basada en microservicios.**

Requerimientos funcionales

Se debe cumplir con todos los requerimientos funcionales de la primera entrega y:

RF12. Visualizar consumo

Actor: Usuario administrador

Los usuarios administradores deben poder acceder a una página en la que pueden ver la factura del mes actual, o de meses anteriores. La facturación de Centinela, al ser un SaaS, se basa en un modelo de cobros mensuales y por uso. Se cobrará de la siguiente manera:

- 0.01 USD por error reportado
- 5 USD por usuario de la organización por mes

No es necesario que el sistema realice ningún cobro. Se pide simplemente mostrar el monto que sería facturado para el mes corriente, detallando el cálculo.

RF13. SDK para creación de error

Actor: Sistema externo

Se deberá disponibilizar un SDK (lenguaje a elección) que permita hacer uso de la operación de crear error (RF9 del anterior obligatorio). En caso de que no se pueda conectar con el servicio de Centinela o el tiempo de respuesta supere los 500 ms el SDK debe dar *timeout* y retornar error.

RF14. Recepción de alertas por correo

Actor: Usuario desarrollador y usuario administrador
Los usuarios desarrolladores deben poder configurar sus preferencias en cuanto a qué correos electrónicos recibir. El usuario debe poder activar correos electrónicos para los siguientes eventos:

- Al asignarse un nuevo error:
    - El usuario debe seleccionar si quiere recibirlo inmediatamente.
    - Optar por recibir una alerta en cierto momento del día (al horario que indique) con la cantidad total de errores que ocurrieron durante ese día. Se debe tener en cuenta que tanto el inicio del día como hora de envío de alerta deben pertenecer al mismo huso horario que el usuario desarrollador.

En ambos casos lo hace filtrando por severidad para decidir si recibe las alertas de errores o no.

- Cuando tiene errores asignados de más de 2 días sin resolver, indistintamente su severidad.

Por defecto, las alertas por correo deberán estar apagadas.

RF15. Reporte de asignación de errores

Actor: Usuario administrador

Además del reporte existente de la primera iteración se pide:

- Reporte de top 10 desarrolladores que resolvieron más incidencias asignadas en los últimos 30 días.
- Reporte de incidencias que llevan más de 2 días sin estar asignadas

Requerimientos no funcionales

Se debe cumplir con todos los requerimientos no funcionales de la primera entrega y:

RNF9. Estilo de arquitectura

Se deberá llevar el estilo de arquitectura hacia uno basado en microservicios, especificando fronteras bien definidas entre los mismos, así como las interfaces mediante las cuales se comunicarán. **Cada uno** de estos servicios deberán verse desarrollados y desplegados en aplicaciones diferentes, comunicándose exclusivamente mediante colas de mensaje o llamadas HTTP en tiempo de ejecución. **Al menos uno de los microservicios debe estar en un lenguaje de programación diferente a los demás.**

RNF10. Integración continua

Además de las pruebas de carga, se deberá contar con pruebas automatizadas para al menos tres requerimientos funcionales de su elección en al menos uno de los repositorios.

El o los repositorios involucrados deben correr las pruebas automatizadas del código cuando un nuevo commit se integra a la rama principal.

RNF11. Identificación de fallas

Para facilitar la detección e identificación de fallas, se deberán centralizar y retener los *logs* emitidos por **todo el sistema** en producción por un período mínimo de 24 horas.

RNF12. Desplegabilidad

Se deberá poder desplegar una nueva versión de cualquier microservicio, sin ocasionar *downtime*.

RNF13. Monitoriabilidad

Para facilitar el diagnóstico ante eventuales fallas, se deberá monitorear las siguientes métricas de los distintos servicios:

- Peticiones por minuto
- Tiempos de respuesta de distintos endpoints

Documentación

Además de la solución a implementar, se pide incluir una documentación con los siguientes puntos:

Descripción de la arquitectura

Deberá mostrar las partes que componen al sistema a través de vistas de módulos, componentes y conectores, y despliegue. Deberá asegurarse de mostrar:

- Distribución de todo el sistema en componentes físicos.
- Flujo de información a través de la infraestructura en tiempo de ejecución.

Justificaciones de diseño

Se pide un detalle de las tácticas aplicadas para conseguir los RNFs exigidos por la especificación. Las mismas pueden incluir tanto decisiones explícitamente introducidas en su diseño (esto es, en *userland*), como aquellas contenidas en las soluciones ya ofrecidas por la plataforma o *framework* utilizados.

Descripción del proceso de *deployment*

Se deberán describir los pasos que comprenden el pasaje del sistema en su entorno de desarrollo al entorno de producción (compilación de *assets*, migraciones, ejecución de pruebas, etc.), incluyendo un detalle de las tácticas utilizadas para preservar la disponibilidad del servicio durante este proceso y cómo fueron implementadas.

Manual de uso y ejemplo de utilización de SDK

Adjuntar un breve manual de utilización del SDK y ejemplo de uso del mismo.

Entrega

La entrega podrá realizarse hasta el día 10 de diciembre a las 21 horas a través de gestion.ort.edu.uy. La entrega incluye:

- URI para el acceso web público a todas las instancias en producción del sistema Centinela.
- Dirección de los repositorios privados en Github.
- Link o adjunto de la Documentación requerida para este obligatorio.
- Cualquier conjunto de credenciales adicionales que sean necesarias para el uso de Centinela.
- Todo script de pruebas de Jmeter, Postman, etc.

Datos de prueba

Se solicita que en el ambiente de producción se dejen ya registrados dos usuarios con datos de prueba.

- **Nombre organización:** ORT
- **Usuario administrador:** admin@ort.com
  Contraseña: Password1
- **Usuario desarrollador:** dev@ort.com
  Contraseña: Password1