Arquitectura de Software en la Práctica
Obligatorio 2

Objetivos

Desarrollar los conceptos que definen la computación en la nube mediante la adopción de técnicas de construcción de software como servicio, con foco en arquitecturas de microservicios orientados a eventos, desde el desarrollo hasta su despliegue en un entorno de producción.

Problema

Luego del éxito obtenido con el despliegue de la aplicación *Coupons* en los primeros clientes, su empresa encuentra dificultades al escalar el producto. A partir de un análisis detallado, descubre que existen los siguientes puntos de tensión en la arquitectura:

- Al acceder a los reportes la *performance* de evaluación de promociones y administración de promociones se ve degradada.
- En horas pico la *performance* global del sistema de ve perjudicada debido a la elevada cantidad de peticiones sobre el *endpoint* de evaluación.

Por otra parte, su empresa consiguió un cliente del *Fortune 100*. Si bien esto significa un gran paso para la empresa se debe realizar modificaciones para atenerse a sus requerimientos.

Desarrollo

El proyecto tecnológico a desarrollar se deberá ajustar a las características de una aplicación *cloud native*. Sin embargo, dada la escasa familiaridad del equipo con tecnologías *cloud* se deberán mantener los esfuerzos iniciales de *DevOps* y orquestación de servicios a un mínimo, motivo por el cual se decide utilizar un servicio *PaaS* que permita facilitar las iteraciones rápidas y automatizadas.

Requerimientos funcionales

Además de los requerimientos funcionales solicitados para la primera iteración (primera entrega) se pide:

RF9. Gestión de usuarios

Actor: Usuario administrador

El sistema debe permitir la posibilidad de gestionar múltiples usuarios administrativos y usuarios de tipo financiero. Entonces, el sistema debe soportar los siguientes tipos de usuarios:

- Usuario administrador: Con los mismos permisos de la primera iteración.
- Usuario de organización: Se conservan los permisos de la primera iteración con excepción de poder visualizar los distintos reportes.
- Usuario de finanzas: Solo puede ver el listado de promociones y acceder al reporte de uso.

De la misma manera que para la primera iteración los nuevos miembros son invitados a través de un link vía correo electrónico.

RF10. SDK de evaluación de promociones

Actor: Sistema externo

Se deberá disponibilizar un SDK (lenguaje a elección) que permita consultar por la operación de evaluación de promociones (*RF8*). En caso de que no se pueda conectar con el servicio de *Coupons* o el tiempo de respuesta supere los **1000 ms** el SDK debe retornar "no aplica" al evaluar una promoción.

RF11. Límite de uso de cupón

Actor: Sistema

El sistema debe permitir tener una opción para reutilizar un mismo codigo de promocion. Es decir, cuántas veces puede ser utilizado por distintos usuarios.

Si la cantidad de utilizaciones supera el límite de uso entonces el sistema debe retornar un mensaje de error.

RF12. Expiración de promociones

Actor: Sistema

Debido a la dificultad operativa que conlleva desactivar las promociones manualmente cuando la organización ya no desea mantener una promoción activa se solicita que estos sean desactivados automáticamente.

El sistema deberá notificar a los usuarios administrativos de la organización vía correo electrónico un día antes de que la promoción expire y también en el momento de que se haga efectivo.

RF13. Agregar cupones a promoción

Actor: Usuario administrador

El sistema debe permitir agregar nuevos cupones cuando la promoción sea del tipo "cupón". Adicionalmente el sistema debe validar que no se hayan generado previamente.

De estos nuevos cupones se puede especificar una fecha de vencimiento distinta (*RF14*) y no mayor a la fecha de expiración de una promoción (si tiene). Las reglas de otorgamiento de un cupón continúan siendo las mismas y no se pueden alterar.

RF14. Vencimiento de cupones

Actor: Sistema

Con la finalidad de incentivar la utilización de los cupones el sistema debe permitir configurar el vencimiento de los cupones de una promoción. En el caso de que un usuario intente utilizar un cupón por fuera de su validez el sistema debe retornar un mensaje de error especial indicando el motivo.

RF15. Reporte demográfico de usuarios

Actor: Usuario administrador y usuario de organización

El sistema deberá soportar un reporte demográfico de usuarios de la organización. Este reporte es segmentado por promoción y debe desplegar la siguiente información:

- Usuarios de una ciudad determinada
- Usuarios por país
- Usuarios por grupo de edad (18-25, 25-40, 40-60, 60+)

La recolección de información se realiza al momento de evaluar una promoción por medio de los siguientes campos opcionales: *city* (IATA), *county* (ISO 3166-1), *birth_date* (mes/dia/año), por lo que el reporte se genera si y solo si esta información es provista.

Requerimientos no funcionales

Además de los requerimientos no funcionales solicitados para la primera iteración (primera entrega) adicionalmente se pide:

RNF8. Monitoriabilidad

Para facilitar la detección e identificación de fallas, se deberá monitorear las siguientes métricas de los distintos servicios:

- Peticiones por minuto
- Tiempos de respuesta de distintos endpoints

RNF9. Independencia de aplicaciones

Debido a que el área de tecnología de la compañía se divide en 3 equipos de desarrollo de 5 personas cada uno se requiere dividir la aplicación *Coupons* en un mínimo de 3 microservicios (sin un máximo) para aprobar el obligatorio. Algunos de estos equipos tienen experiencia en otras tecnologías diferentes a *Ruby on Rails*, por lo tanto al menos uno de los microservicios deberá ser desarrollado con otro *stack* tecnológico.

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.

Adicionalmente deberá explicar cómo separo y llegó del sistema existente a la arquitectura presentada y cómo logró cumplir con los distintos principios de una arquitectura de microservicios.

Justificaciones de diseño

Se pide un detalle de las tácticas aplicadas para conseguir los requerimientos no funcionales 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.

Entrega

La entrega podrá realizarse hasta el día **28 de noviembre** a las **21 horas** a través de gestion.ort.edu.uy. La entrega incluye:

- URI para el acceso web público a las instancias en producción del sistema *Coupons*.
- Dirección de los repositorios privados en *Github*.
- Cualquier conjunto de credenciales adicionales que sean necesarias para el uso de *Coupons*.