PLAN DE PRUEBAS MICROSERVICIOS

20 OCTUBRE 2023

3IT – PROGRAMA EUREKA TEAMS
MENTORES: JACOB VEGA – JORGE ANDREWARTHA
TRAINEE: FRANCISCA MADARIAGA

3IT.

1. Introducción

1.1 Propósito

Garantizar la calidad en las funcionalidades de microservicios con una entidad principal y secundaria realizada con el lenguaje de programación Java, el framework Spring Boot, servidor Eureka para registro y localización, y API Gateway para la recepción de llamadas y derivación de servicios.

1.2 Objetivos del plan

El siguiente plan comprende una serie de procesos que se llevarán a cabo para la correcta verificación y validación de los microservicios, con el objeto de:

- Hallar defectos y fallas en el producto
- Validar las necesidades del usuario
- Verificar los requerimientos
- Determinar y evaluar la calidad del producto
- Describir y especificar los tipos de pruebas que se implementaran
- Definir y describir tareas necesarias para el proceso de pruebas
- Indicar herramientas necesarias para las pruebas

1.3 Alcance del Testing

Este plan de pruebas cubre las principales funcionalidades de los microservicios correspondientes a entidades principales y secundarias (Artista y álbum), las cuales incluyen operaciones separadas de un CRUD (Crear, Leer, Eliminar y Actualizar). Se consideran aspectos funcionales y no funcionales.

1.4 Criterios de Entrada

Para dar inicio a la fase de pruebas, es necesario que se cumplan los siguientes criterios:

- El desarrollador a cargo (Luis Fuentes), ha liberado el proyecto para realizar las pruebas.
- El desarrollador confirma que el entregable se encuentra funcional para el ambiente de pruebas.
- El desarrollador entrega la documentación relacionada con la funcionalidad de la versión del sistema liberado.
- Los requerimientos definidos cumplen con los criterios de salidas especificados

- El Analista QA tiene en su poder todas las herramientas necesarias para ejecutar el sistema a probar.

1.5 Criterios de Salida

Los siguientes criterios deben ser satisfechos para considerar los casos de prueba como pasados:

- No existen fallas, paradas o procesos culminados de forma inesperada.
- El probador ha ejecutado todas las pruebas planeadas para cada ambiente.
- El desarrollador ha resuelto todos los defectos que deben ser depurados.
- Se realizan pruebas de regresión y confirmación.
- El dueño del producto aprueba que el producto satisface las expectativas impuestas por el usuario.

2. Configuración de ambiente de pruebas

2.1 Tecnologías, herramientas y software de proyecto

- Eclipse IDE for Java Developers v4.29.0
- Java Development Kit 21
- Postman v9.4
- MySQL Workbench v8.0.34
- Spring Data JPA v3.1.4
- Lombok v 1.18.30
- Spring Web v5.3.27
- Spring Boot v3.1.4
- Spring Cloud Starter Gateway v2.2.9
- Spring Cloud Netflix Eureka Server v4.0.3

2.2 Tecnologías, herramientas y software para las pruebas

- Pruebas manuales
- Postman v10
- Test Link v1.9.20

2.3 Datos de entrada

Artista (Entidad principal)

- Nombre Artista

- Primer nombre
- Primer Apellido
- Fecha de nacimiento
- Edad actual

Álbum (Entidad secundaria)

- Nombre del álbum
- Total de canciones
- Duración del álbum

3. Pruebas del sistema

Módulos a probar:

- Registro de artista
- Visualización de todos los artistas registrados
- Visualización de artista en específico
- Eliminación de artista
- Registro de álbum
- Visualización de todos los álbumes registrados
- Visualización de álbum en específico
- Eliminación de álbum

Cada caso de prueba creado debe ser documentado por el Analista QA (acciones a tomar, datos a usar y resultados esperados). La documentación debe ser lo suficientemente clara y concisa para que cualquier otro probador pueda ejecutarlo.

3.1 Estrategias de las pruebas del sistema

Este análisis identificó áreas importantes que están sujetas a fallas o son una prioridad para el negocio. Estas áreas requieren datos de prueba nuevos o actualizados, casos de pruebas, entre otras tareas.

- Preparar conjunto de datos para los requisitos de las pruebas de rendimiento.
- No solo escenarios ficticios sino datos reales que cumplen con la entrada esperada actual.
- Se crearán casos de prueba manuales para la aplicación móvil que cubrirán cualquier característica funcional.

- Se deben crear datos de prueba para enfatizar cualquier condición compleja requerida por la aplicación.

3.2 Pruebas en Operación Normal

Módulos a probar:

Artista:

- Registro de artistas
- Eliminar registro de artistas por ID
- Ver registro específico con ID
- Ver todos los registros de artistas

Álbum:

- Registro de álbumes
- Eliminar registro de álbumes por ID
- Ver registro específico con ID
- Ver todos los registros de álbumes

Funcionalidad Método POST (Artista y álbum)

Se ingresa el Endpoint proporcionado por el desarrollador. Al entregar los datos de entrada en el apartado de body (cuerpo) y enviar la solicitud “POST”, se espera que el sistema entregue una respuesta “201”, correspondiente a “Created” (Creado)”.

Funcionalidad Método DELETE (Artista y álbum)

Se ingresa el Endpoint proporcionado por el desarrollador, sumándole el ID del registro que se desea eliminar. Se procede a enviar la solicitud “DELETE” y esperar la respuesta “200” correspondiente a “OK”, es decir, una solicitud exitosa.

Funcionalidad Método GET ID (Artista y álbum)

Se ingresa el Endpoint proporcionado por el desarrollador., sumándole el ID del registro que se desea ver. Se procede a enviar la solicitud “GET” y esperar la respuesta “200” correspondiente a “OK”, es decir, una solicitud exitosa. En el apartado de Response de Postman, se podrá ver los datos del registro especificado.

Funcionalidad Método GET ALL (Artista y álbum)

Se ingresa el Endpoint proporcionado por el desarrollador. Se procede a enviar la solicitud “GET” y esperar la respuesta “200” correspondiente a “OK”, es decir, una solicitud exitosa. En el apartado de Response de Postman, se podrá ver una lista con los datos de los registros almacenados.

Suponiendo que las funcionalidades anteriores corresponden a un “Happy Path”, se crearán diversos casos de prueba a partir de modificaciones que se realizarán en los datos de entrada (validación de datos como nulos, caracteres permitidos, formato, longitud, etc.) Además de la verificación de la correcta creación, eliminación y visualización de datos. Se estimará si son necesarias la realización de pruebas no funcionales.

3.1 Pruebas en Condiciones de Excepción:

Errores de entrada de datos: Enviar datos de entrada incorrectos o mal formateados y verificar cómo responden los microservicios.

Plataforma y entorno: Ejecución de pruebas en diferentes entornos de prueba para observar el comportamiento de los microservicios de manera consistente en diferentes sistemas operativos y configuraciones.

Seguridad: Observar cómo los microservicios responden a diversos ataques simulados de seguridad SQL.

Red y comunicación: Realizar pruebas que simulen interrupciones de red o tiempos de espera agotados al intentar la comunicación con los microservicios.

Integración de terceros: Probar la interacción con servicios externos a los microservicios, observando el manejo de respuestas inesperadas o errores de esos servicios externos.

Rendimiento: Ejecutar pruebas de carga que evalúen la respuesta de los microservicios a cargas inesperadas o picos de tráfico.

3.2 Criterios de Éxito de Pruebas:

Respuestas correctas: Los datos obtenidos deben coincidir con los datos esperados según el caso de prueba especificado. Un ejemplo es obtener la lista de registro de artistas al enviar una solicitud GET.

Códigos de estado HTTP: Al obtener las respuestas de los microservicios, estas deben devolver los códigos de estado HTTP esperados para cada solicitud.

Tiempo de respuesta: Las respuestas que entregan los microservicios deben tener un tiempo aceptable.

Cumplimiento de requisitos: La funcionalidad de los microservicios se comporta según lo esperado y cumple requisitos funcionales.

Manejo de errores: Se manejan de forma correcta las excepciones y devuelve los mensajes de error descriptivos para casa problema.

3.3 Criterios de Fracaso de Pruebas:

Respuestas incorrectas: Las respuestas de los microservicios no coinciden con los datos esperados o contienen errores.

Códigos de estado HTTP incorrectos: Se obtienen códigos HTTP incorrectos o incoherentes con lo esperado.

Tiempo de respuestas inaceptables: El tiempo de respuesta excede a los límites aceptables o es excesivamente lento.

Errores en casos de prueba: Casos de prueba propuestos fallan, indicando que los microservicios no se comportan según lo esperado.

3.3 Entregables

El documento para observar los resultados de la fase de pruebas, contendrá toda la información recabada al ejecutar las pruebas, considerando posibles arreglos, oportunidades de mejoras a futuro, informes de defectos, áreas críticas del sistema, etc.

Los casos de prueba con sus resultados y métricas serán especificados a partir del software “TestLink”.

4. Tareas

4.1 Actividades

La secuencia de actividades para probar el sistema es:

1. **Planificación:** Se definen los objetivos de prueba según las expectativas del usuario, considerando los posibles riesgos que se obtengan al probar los microservicios.
2. **Monitoreo y control:** Ocurre durante todo el cronograma de pruebas, principalmente, para establecer posibles acciones correctivas a los microservicios.

3. **Análisis de prueba:** De acuerdo al nivel de prueba en que se encuentre el aplicativo, se definen las bases de prueba de acuerdo a la especificación de requisitos.
4. **Diseño de pruebas:** Se priorizan casos de prueba y conjunto de pruebas, se definen los datos necesarios atingentes a los microservicios, se especifica el entorno de prueba que se utilizará y las herramientas necesarias.
5. **Implementación de pruebas:** Se crean y organizan las suites de prueba de acuerdo a las funcionalidades de los microservicios.
6. **Ejecución de pruebas:** Se ejecutan las suites de prueba especificadas, se registran resultados del funcionamiento de los microservicios, se evalúa si es necesario realizar más pruebas, comparación de resultados, informar defectos y repetir pruebas para verificar correcciones.
7. **Compleción de pruebas:** Recopilación de datos, evaluando si los resultados cumplen los objetivos propuestos. Evaluar si se requieren más pruebas. Documentar sobre la aceptación del producto.

4.2 Responsabilidades

Responsabilidades del Grupo de Desarrollo

- Ejecutar las pruebas unitarias
- Ejecutar y probar la integración de bajo nivel
- Depuración del código
- Entregar documentación con especificaciones del producto
- Responder inquietudes a probador

Responsabilidades del Grupo de Testing

- Planificar las pruebas del sistema
- Configurar el ambiente de prueba
- Ejecutar las pruebas del sistema
- Reporte de defectos
- Redacción de informes de resultados y mejoras

Responsabilidades de la Gerencia

- Proveer recursos
- Aceptación final y aprobación de la liberación del producto

4.3 Planificación

Semana	Actividades de prueba
Semana 16 Octubre 2023	• Planificación y diseño de plan de pruebas • Elaboración de documento de registro de casos de pruebas
Semana 23 Octubre 2023	• Redacción de casos de pruebas, ejecución y registro de resultados • Informe de defectos a desarrollador • Pruebas de regresión y confirmación
Semana 30 Octubre 2023	• Evaluación sobre si es necesario pruebas adicionales • Informe de resultados para cumplimiento de objetivos