

UNIVERSIDAD CENTRAL DEL ECUADOR

FACULTAD DE CIENCIAS APLICADAS

INGENIERÍA EN SISTEMAS DE INFORMACIÓN



PROGRAMACIÓN DISTRIBUIDA

DOCENTE: ING. DIEGO PINTO

SEMESTRE: NOVENO

ESTUDIANTE:

- **BRYAN CHOLANGO**
- **ELVIS HERRERA**

FECHA: 04 AGOSTO 2023

2023 - 2023

Tema: Evaluación (Aplicación Distribuida Tolerante a fallos)

Contenido

1.	Introducción	3
2.	Objetivos	3
3.	Desarrollo	3
3.1.	Arquitectura del Sistema Escogido.....	3
3.2.	Descripción del Sistema Escogido	4
3.2.1.	App-authors	4
3.2.2.	App-books	8
3.3.	Ambiente de Pruebas.....	12
3.4.	Implementación de los servicios.....	13
3.4.1.	app-authors.....	14
3.4.2.	app-books.....	14
3.4.3.	app-books consume app-authors	16
3.5.	Pruebas Realizadas.....	16
3.5.1.	Pruebas Funcionales	16
3.5.2.	Pruebas de Tolerancia a Fallos	16
3.6.	Resultados de las Pruebas.....	17
3.6.1.	Pruebas Funcionales	17
3.6.2.	Pruebas de Tolerancia a Fallos	18
4.	Conclusiones	20
5.	Referencias.....	21

1. Introducción

En los últimos años, se ha tenido un crecimiento significativo en el interés y la adopción de enfoques de sistemas distribuidos en el ámbito del desarrollo de software. Los sistemas distribuidos son un paradigma en el que los componentes de software interconectados colaboran y trabajan juntos para lograr objetivos comunes, a pesar de operar en nodos separados dentro de una red.

Un enfoque de sistemas distribuidos implica diseñar y construir aplicaciones o servicios de software de manera que puedan operar en múltiples nodos o máquinas interconectadas. Esto contrasta con las arquitecturas monolíticas en las que una aplicación se ejecuta en una sola máquina. En lugar de un único y gran bloque de código, los sistemas distribuidos se componen de múltiples componentes interconectados que pueden estar distribuidos geográficamente.

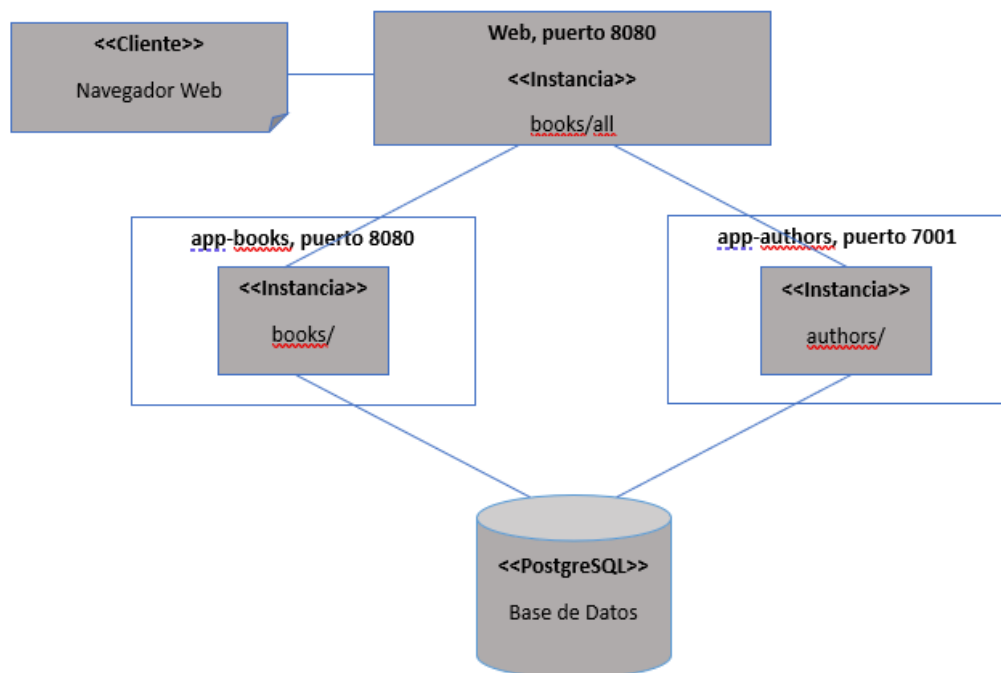
Los sistemas distribuidos pueden utilizar diversos modelos de comunicación, como llamadas remotas, servicios REST, mensajería asíncrona y comunicación a través de servicios web.

2. Objetivos

- Consultar un programa que tenga las características de un sistema distribuido.
- Evaluar y verificar el funcionamiento de un sistema que sea distribuido.
- Aprender la funcionalidad y características que debe tener un sistema distribuido tolerante a fallos.

3. Desarrollo

3.1. Arquitectura del Sistema Escogido



3.2. Descripción del Sistema Escogido

3.2.1. App-authors

3.2.1.1. Lenguaje de Programación (Java)

El lenguaje de programación utilizado en esta aplicación es Java con modulo SDK 17.



Java es un lenguaje de programación de alto nivel y orientado a objetos. Fue diseñado para ser portátil, lo que significa que las aplicaciones escritas en Java pueden ejecutarse en diferentes plataformas sin necesidad de modificaciones significativas. Java se ha vuelto muy popular en el desarrollo de software empresarial, aplicaciones web, aplicaciones móviles (Android) y sistemas embebidos debido a su combinación de características como la portabilidad, la seguridad y la robustez. Las principales características de Java incluyen:

- **Orientación a objetos:** Java es un lenguaje de programación orientado a objetos, lo que significa que organiza el código en clases y objetos, promoviendo la modularidad y reutilización del código.
- **Portabilidad:** Las aplicaciones Java se compilan en un formato intermedio llamado "bytecode", que puede ser ejecutado por la Máquina Virtual Java (JVM) en cualquier plataforma que tenga una implementación de JVM disponible.
- **Seguridad:** Java proporciona mecanismos de seguridad integrados, como la gestión de permisos y la ejecución de código en un entorno controlado.
- **Gestión automática de memoria:** Java utiliza un sistema de recolección de basura para administrar automáticamente la memoria asignada a los objetos, lo que ayuda a prevenir fugas de memoria y simplifica el proceso de programación.
- **Amplia biblioteca estándar:** Java incluye una amplia gama de bibliotecas estándar que ofrecen funcionalidades para tareas comunes, como manipulación de cadenas, entrada/salida, manejo de excepciones y más.

3.2.1.2. Herramienta de Construcción (Gradle)

La herramienta de construcción utilizada en esta aplicación es Gradle en su version 7.2.



Gradle es una herramienta de construcción y automatización de tareas que se utiliza en el desarrollo de software en Java y otros lenguajes. Su principal función es facilitar la gestión del ciclo de vida de construcción y despliegue de proyectos, simplificando tareas repetitivas y permitiendo a los desarrolladores centrarse en la escritura de código y en la lógica de la aplicación. Aquí hay algunas formas en las que Gradle se utiliza en el contexto del desarrollo de aplicaciones Java:

- **Gestión de Dependencias:** Gradle permite administrar fácilmente las dependencias del proyecto, como bibliotecas y frameworks externos. Puede descargar automáticamente las dependencias necesarias y garantizar que las versiones sean compatibles.
- **Compilación:** Gradle se encarga de compilar el código fuente de Java en archivos bytecode ejecutables. Configura y ejecuta el proceso de compilación de manera eficiente.
- **Pruebas y Ejecución:** Puedes definir tareas para ejecutar pruebas automatizadas y verificar la calidad y la integridad del código. También puedes configurar tareas para ejecutar la aplicación Java construida.
- **Empaquetado:** Gradle puede empaquetar la aplicación en diferentes formatos, como JAR (Java Archive) o WAR (Web Archive), listos para su despliegue en diferentes entornos.
- **Generación de Documentación:** Puede generar documentación automáticamente, como documentación de API, a partir de comentarios en el código fuente.
- **Integración con Entornos de Desarrollo:** Gradle se integra con IDEs populares como IntelliJ IDEA, Eclipse y Android Studio, lo que facilita la configuración del entorno de desarrollo.
- **Gestión de Propiedades y Configuraciones:** Puedes gestionar configuraciones específicas del proyecto, como propiedades, variables de entorno y opciones de construcción, de manera centralizada en archivos de configuración de Gradle.

3.2.1.3. Base de Datos (PostgreSQL)

La base de datos utilizada en esta aplicación es PostgreSQL 14 con su pgAdmin 4.



PostgreSQL, también conocido como Postgres, es un sistema de gestión de bases de datos relacional de código abierto que ofrece una amplia gama de características y capacidades como:

- **Fiabilidad y Durabilidad:** PostgreSQL se enfoca en la integridad de los datos y la confiabilidad. Utiliza transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para garantizar que las operaciones sean seguras y confiables, incluso en caso de fallas del sistema.
- **Soporte para Tipos de Datos Avanzados:** Además de los tipos de datos básicos como enteros y cadenas de texto, PostgreSQL ofrece tipos de datos avanzados como matrices, JSON, hstore (almacenamiento clave-valor), geometrías y más. También permite definir tus propios tipos de datos personalizados.
- **Lenguaje de Consulta Complejo:** PostgreSQL admite consultas SQL potentes y complejas, incluidas subconsultas, uniones, agregaciones, ventanas y consultas recursivas.
- **Extensiones y Funcionalidades Adicionales:** Puedes extender las capacidades de PostgreSQL mediante la instalación de extensiones. Estas extensiones proporcionan funciones adicionales, como soporte para búsqueda de texto completo, análisis espacial avanzado, manipulación de datos temporales y más.
- **Integridad Referencial y Restricciones:** PostgreSQL admite restricciones de integridad referencial, como claves primarias, claves foráneas y restricciones únicas, para garantizar la coherencia de los datos en la base de datos.

- **Soporte para Procedimientos Almacenados y Funciones:** PostgreSQL permite definir funciones almacenadas en varios lenguajes, incluidos SQL, PL/pgSQL (similar a PL/SQL), Python, Java y otros.
- **Replicación y Alta Disponibilidad:** PostgreSQL ofrece opciones de replicación, como la replicación síncrona y asíncrona, para garantizar la alta disponibilidad y la redundancia de los datos.
- **Seguridad Avanzada:** PostgreSQL permite autenticación a nivel de host y de usuario, control de acceso basado en roles y privilegios granulares para garantizar la seguridad de los datos.

3.2.1.4. Protocolo de Red (HTTP)

El protocolo de red que utilizara la aplicación para la transmisión de datos es HTTP, con formato JSON para el intercambio de información entre aplicaciones.

<http://>

El Protocolo de Transferencia de Hipertexto (HTTP, por sus siglas en inglés) es un protocolo de red utilizado para transferir datos en la World Wide Web (WWW). Algunas de las características más importantes de HTTP son:

- **Basado en texto:** HTTP es un protocolo de texto simple que utiliza mensajes de solicitud y respuesta para transferir datos entre servidores web y navegadores.
- **Cliente-servidor:** HTTP sigue el modelo cliente-servidor, en el que el cliente (como un navegador web) envía una solicitud al servidor, y el servidor responde con los datos solicitados.
- **Sin estado:** HTTP es un protocolo sin estado, lo que significa que no guarda información sobre las solicitudes previas del cliente. Cada solicitud se trata como una nueva transacción independiente.
- **Métodos de solicitud:** HTTP admite varios métodos de solicitud, como GET, POST, PUT y DELETE, que se utilizan para realizar acciones específicas en el servidor web.
- **Códigos de estado:** HTTP utiliza códigos de estado para indicar el resultado de una solicitud. Los códigos de estado incluyen 200 OK (solicitud exitosa), 404 Not Found (recurso no encontrado) y 500 Internal Server Error (error interno del servidor).
- **Seguridad:** HTTP no proporciona cifrado o autenticación, lo que significa que los datos pueden ser interceptados o manipulados. Sin embargo, HTTPS (HTTP seguro) utiliza SSL/TLS para cifrar y autenticar los datos transferidos, proporcionando una mayor seguridad.
- **Extensible:** HTTP es un protocolo extensible que permite la creación de nuevas funciones y características a través de extensiones y complementos.

3.2.1.5. Editor de Código (IntelliJ IDEA)

El editor en donde se realizó esta aplicación es IntelliJ IDEA 2022.3.2



Es un entorno de desarrollo integrado (IDE) ampliamente utilizado para la programación en diversos lenguajes, pero principalmente se asocia con el desarrollo de aplicaciones en Java. Aquí tienes algunas de las características clave de IntelliJ IDEA:

- **Productividad Mejorada:** IntelliJ IDEA se destaca por su enfoque en aumentar la productividad del desarrollador. Ofrece sugerencias de código inteligentes, completado automático, corrección de errores en tiempo real y refactorización avanzada.
- **Soporte para Múltiples Lenguajes:** Aunque es conocido por su excelente soporte para Java, IntelliJ IDEA también admite una variedad de lenguajes de programación, como Kotlin, Scala, Groovy, JavaScript, TypeScript, HTML, CSS, entre otros.
- **Integración con Frameworks y Tecnologías:** Proporciona una integración profunda con diversos frameworks y tecnologías, como Spring, Hibernate, Android, JUnit y más, lo que agiliza el desarrollo de aplicaciones basadas en estas tecnologías.
- **Análisis de Código:** IntelliJ IDEA realiza análisis estático de código en tiempo real para ayudar a identificar posibles problemas y errores antes de la ejecución. También ofrece herramientas para la detección de duplicados y la mejora de la calidad del código.
- **Gestión de Dependencias:** Facilita la gestión de dependencias de proyectos mediante integración con sistemas como Maven y Gradle, lo que permite agregar, actualizar y administrar bibliotecas externas.
- **Diseño de Interfaz Gráfica:** Para aplicaciones de escritorio y móviles, IntelliJ IDEA ofrece herramientas visuales para diseñar interfaces de usuario, lo que acelera el desarrollo de la interfaz gráfica.
- **Extensibilidad:** IntelliJ IDEA es altamente extensible. Los usuarios pueden agregar plug-ins y extensiones personalizadas para adaptar el entorno a sus necesidades específicas.
- **Integración con Herramientas de Desarrollo Frontend:** Para aplicaciones web, IntelliJ IDEA se integra con herramientas populares de desarrollo frontend como JavaScript, TypeScript, Angular, React, Vue.js y más.

3.2.1.6. Framework (Quarkus)

En esta aplicación se utilizó el framework Quarkus en su versión 2.16.1.Final.



Quarkus es un framework de desarrollo de aplicaciones Java diseñado para aplicaciones nativas de la nube y microservicios, con un enfoque en el alto rendimiento, el inicio rápido y la eficiencia en el uso de recursos. Aquí tienes algunas de las características más destacadas de Quarkus:

- **Inicio Rápido:** Quarkus se destaca por su arranque rápido. Esto significa que las aplicaciones pueden iniciar en milisegundos en lugar de segundos, lo que es esencial para cargas de trabajo en la nube y microservicios.
- **Bajo Consumo de Memoria:** Quarkus optimiza el uso de memoria, lo que permite que las aplicaciones se ejecuten eficientemente en entornos con recursos limitados.
- **Quarkus Extensions:** Quarkus ofrece una variedad de extensiones que facilitan la integración con tecnologías y frameworks populares, como Hibernate, RESTEasy, Apache

Kafka, y más. Estas extensiones aceleran el desarrollo al proporcionar configuraciones y características listas para usar.

- **Desarrollo Reactivo:** Quarkus admite la programación reactiva y ofrece características para construir aplicaciones reactivas que pueden manejar cargas de trabajo asincrónicas y de alto rendimiento.
- **Soporte para Inyección de Dependencias:** Quarkus utiliza CDI (Contexts and Dependency Injection) para administrar la inyección de dependencias y el ciclo de vida de los componentes.
- **Integración con Jakarta EE:** Quarkus es compatible con Jakarta EE (anteriormente conocido como Java EE), lo que permite a los desarrolladores aprovechar su experiencia en Java EE para crear aplicaciones modernas y eficientes.
- **Desarrollo de Extensiones Personalizadas:** Puedes crear tus propias extensiones personalizadas de Quarkus para satisfacer necesidades específicas de tu proyecto.
- **Integración con Herramientas de Desarrollo:** Quarkus se integra con IDEs populares como IntelliJ IDEA, Eclipse y Visual Studio Code para facilitar el desarrollo y la depuración.
- **Compatibilidad con Kubernetes y Orquestación de Contenedores:** Quarkus se integra con Kubernetes y otras herramientas de orquestación de contenedores, lo que facilita el despliegue y la administración de aplicaciones en entornos de contenedores.
- **APIs y Servicios Web:** Quarkus admite la creación de APIs y servicios web utilizando tecnologías como JAX-RS (Java API for RESTful Web Services) y más.

3.2.2. App-books

3.2.2.1. Lenguaje de Programación (Java)

El lenguaje de programación utilizado en esta aplicación es Java con modulo SDK 17.



Java es un lenguaje de programación de alto nivel y orientado a objetos. Fue diseñado para ser portátil, lo que significa que las aplicaciones escritas en Java pueden ejecutarse en diferentes plataformas sin necesidad de modificaciones significativas. Java se ha vuelto muy popular en el desarrollo de software empresarial, aplicaciones web, aplicaciones móviles (Android) y sistemas embebidos debido a su combinación de características como la portabilidad, la seguridad y la robustez. Las principales características de Java incluyen:

- **Orientación a objetos:** Java es un lenguaje de programación orientado a objetos, lo que significa que organiza el código en clases y objetos, promoviendo la modularidad y reutilización del código.
- **Portabilidad:** Las aplicaciones Java se compilan en un formato intermedio llamado "bytecode", que puede ser ejecutado por la Máquina Virtual Java (JVM) en cualquier plataforma que tenga una implementación de JVM disponible.
- **Seguridad:** Java proporciona mecanismos de seguridad integrados, como la gestión de permisos y la ejecución de código en un entorno controlado.

- **Gestión automática de memoria:** Java utiliza un sistema de recolección de basura para administrar automáticamente la memoria asignada a los objetos, lo que ayuda a prevenir fugas de memoria y simplifica el proceso de programación.
- **Amplia biblioteca estándar:** Java incluye una amplia gama de bibliotecas estándar que ofrecen funcionalidades para tareas comunes, como manipulación de cadenas, entrada/salida, manejo de excepciones y más.

3.2.2.2. Herramienta de Construcción (Gradle)

La herramienta de construcción utilizada en esta aplicación es Gradle en su versión 7.2.



Gradle es una herramienta de construcción y automatización de tareas que se utiliza en el desarrollo de software en Java y otros lenguajes. Su principal función es facilitar la gestión del ciclo de vida de construcción y despliegue de proyectos, simplificando tareas repetitivas y permitiendo a los desarrolladores centrarse en la escritura de código y en la lógica de la aplicación. Aquí hay algunas formas en las que Gradle se utiliza en el contexto del desarrollo de aplicaciones Java:

- **Gestión de Dependencias:** Gradle permite administrar fácilmente las dependencias del proyecto, como bibliotecas y frameworks externos. Puede descargar automáticamente las dependencias necesarias y garantizar que las versiones sean compatibles.
- **Compilación:** Gradle se encarga de compilar el código fuente de Java en archivos bytecode ejecutables. Configura y ejecuta el proceso de compilación de manera eficiente.
- **Pruebas y Ejecución:** Puedes definir tareas para ejecutar pruebas automatizadas y verificar la calidad y la integridad del código. También puedes configurar tareas para ejecutar la aplicación Java construida.
- **Empaquetado:** Gradle puede empaquetar la aplicación en diferentes formatos, como JAR (Java Archive) o WAR (Web Archive), listos para su despliegue en diferentes entornos.
- **Generación de Documentación:** Puede generar documentación automáticamente, como documentación de API, a partir de comentarios en el código fuente.
- **Integración con Entornos de Desarrollo:** Gradle se integra con IDEs populares como IntelliJ IDEA, Eclipse y Android Studio, lo que facilita la configuración del entorno de desarrollo.
- **Gestión de Propiedades y Configuraciones:** Puedes gestionar configuraciones específicas del proyecto, como propiedades, variables de entorno y opciones de construcción, de manera centralizada en archivos de configuración de Gradle.

3.2.2.3. Base de Datos (PostgreSQL)

La base de datos utilizada en esta aplicación es PostgreSQL 14 con su pgAdmin 4.



PostgreSQL, también conocido como Postgres, es un sistema de gestión de bases de datos relacional de código abierto que ofrece una amplia gama de características y capacidades como:

- **Fiabilidad y Durabilidad:** PostgreSQL se enfoca en la integridad de los datos y la confiabilidad. Utiliza transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) para garantizar que las operaciones sean seguras y confiables, incluso en caso de fallas del sistema.
- **Soporte para Tipos de Datos Avanzados:** Además de los tipos de datos básicos como enteros y cadenas de texto, PostgreSQL ofrece tipos de datos avanzados como matrices, JSON, hstore (almacenamiento clave-valor), geometrías y más. También permite definir tus propios tipos de datos personalizados.
- **Lenguaje de Consulta Complejo:** PostgreSQL admite consultas SQL potentes y complejas, incluidas subconsultas, uniones, agregaciones, ventanas y consultas recursivas.
- **Extensiones y Funcionalidades Adicionales:** Puedes extender las capacidades de PostgreSQL mediante la instalación de extensiones. Estas extensiones proporcionan funciones adicionales, como soporte para búsqueda de texto completo, análisis espacial avanzado, manipulación de datos temporales y más.
- **Integridad Referencial y Restricciones:** PostgreSQL admite restricciones de integridad referencial, como claves primarias, claves foráneas y restricciones únicas, para garantizar la coherencia de los datos en la base de datos.
- **Soporte para Procedimientos Almacenados y Funciones:** PostgreSQL permite definir funciones almacenadas en varios lenguajes, incluidos SQL, PL/pgSQL (similar a PL/SQL), Python, Java y otros.
- **Replicación y Alta Disponibilidad:** PostgreSQL ofrece opciones de replicación, como la replicación síncrona y asíncrona, para garantizar la alta disponibilidad y la redundancia de los datos.
- **Seguridad Avanzada:** PostgreSQL permite autenticación a nivel de host y de usuario, control de acceso basado en roles y privilegios granulares para garantizar la seguridad de los datos.

3.2.2.4. Protocolo de Red (HTTP)

El protocolo de red que utilizara la aplicación para la transmisión de datos es HTTP, con formato JSON para el intercambio de información entre aplicaciones.



El Protocolo de Transferencia de Hipertexto (HTTP, por sus siglas en inglés) es un protocolo de red utilizado para transferir datos en la World Wide Web (WWW). Algunas de las características más importantes de HTTP son:

- **Basado en texto:** HTTP es un protocolo de texto simple que utiliza mensajes de solicitud y respuesta para transferir datos entre servidores web y navegadores.
- **Cliente-servidor:** HTTP sigue el modelo cliente-servidor, en el que el cliente (como un navegador web) envía una solicitud al servidor, y el servidor responde con los datos solicitados.

- **Sin estado:** HTTP es un protocolo sin estado, lo que significa que no guarda información sobre las solicitudes previas del cliente. Cada solicitud se trata como una nueva transacción independiente.
- **Métodos de solicitud:** HTTP admite varios métodos de solicitud, como GET, POST, PUT y DELETE, que se utilizan para realizar acciones específicas en el servidor web.
- **Códigos de estado:** HTTP utiliza códigos de estado para indicar el resultado de una solicitud. Los códigos de estado incluyen 200 OK (solicitud exitosa), 404 Not Found (recurso no encontrado) y 500 Internal Server Error (error interno del servidor).
- **Seguridad:** HTTP no proporciona cifrado o autenticación, lo que significa que los datos pueden ser interceptados o manipulados. Sin embargo, HTTPS (HTTP seguro) utiliza SSL/TLS para cifrar y autenticar los datos transferidos, proporcionando una mayor seguridad.
- **Extensible:** HTTP es un protocolo extensible que permite la creación de nuevas funciones y características a través de extensiones y complementos.

3.2.2.5. Editor de Código (IntelliJ IDEA)

El editor en donde se realizó esta aplicación es IntelliJ IDEA 2022.3.2



Es un entorno de desarrollo integrado (IDE) ampliamente utilizado para la programación en diversos lenguajes, pero principalmente se asocia con el desarrollo de aplicaciones en Java. Aquí tienes algunas de las características clave de IntelliJ IDEA:

- **Productividad Mejorada:** IntelliJ IDEA se destaca por su enfoque en aumentar la productividad del desarrollador. Ofrece sugerencias de código inteligentes, completado automático, corrección de errores en tiempo real y refactorización avanzada.
- **Soporte para Múltiples Lenguajes:** Aunque es conocido por su excelente soporte para Java, IntelliJ IDEA también admite una variedad de lenguajes de programación, como Kotlin, Scala, Groovy, JavaScript, TypeScript, HTML, CSS, entre otros.
- **Integración con Frameworks y Tecnologías:** Proporciona una integración profunda con diversos frameworks y tecnologías, como Spring, Hibernate, Android, JUnit y más, lo que agiliza el desarrollo de aplicaciones basadas en estas tecnologías.
- **Análisis de Código:** IntelliJ IDEA realiza análisis estático de código en tiempo real para ayudar a identificar posibles problemas y errores antes de la ejecución. También ofrece herramientas para la detección de duplicados y la mejora de la calidad del código.
- **Gestión de Dependencias:** Facilita la gestión de dependencias de proyectos mediante integración con sistemas como Maven y Gradle, lo que permite agregar, actualizar y administrar bibliotecas externas.
- **Diseño de Interfaz Gráfica:** Para aplicaciones de escritorio y móviles, IntelliJ IDEA ofrece herramientas visuales para diseñar interfaces de usuario, lo que acelera el desarrollo de la interfaz gráfica.
- **Extensibilidad:** IntelliJ IDEA es altamente extensible. Los usuarios pueden agregar plug-ins y extensiones personalizadas para adaptar el entorno a sus necesidades específicas.

- **Integración con Herramientas de Desarrollo Frontend:** Para aplicaciones web, IntelliJ IDEA se integra con herramientas populares de desarrollo frontend como JavaScript, TypeScript, Angular, React, Vue.js y más.

3.2.2.6. Framework (Helidon Microprofile)

En esta aplicación se utilizó el framework Helidon en su versión 3.1.0.



Helidon es un framework de desarrollo de aplicaciones Java diseñado específicamente para la creación de microservicios y aplicaciones en la nube. Es una iniciativa de código abierto de Oracle y ofrece dos variantes principales: Helidon SE (Helidon Small Edition) y Helidon MP (Helidon MicroProfile).

Algunas características de Helidon MP (MicroProfile):

- **Conformidad con MicroProfile:** Helidon MP es compatible con el proyecto Eclipse MicroProfile, que es un conjunto de especificaciones para la creación de microservicios en Java. Esto permite utilizar estándares conocidos y ampliamente adoptados en el desarrollo de microservicios.
- **Integración con CDI:** Helidon MP se integra con CDI (Contexts and Dependency Injection), lo que facilita la inyección de dependencias y la creación de componentes.
- **Gestión de Configuración:** Ofrece herramientas para administrar la configuración de la aplicación, como propiedades y variables de entorno.
- **Metrics y Monitorización:** Proporciona soporte para métricas y monitorización, lo que permite recopilar datos y estadísticas sobre el rendimiento y la salud de la aplicación.
- **Cliente REST:** Helidon MP incluye un cliente REST para realizar llamadas a servicios externos utilizando HTTP.
- **Fault Tolerance:** Ofrece características de tolerancia a fallos, como el manejo de fallbacks y circuit breakers, para mejorar la resiliencia de las aplicaciones.

3.3. Ambiente de Pruebas

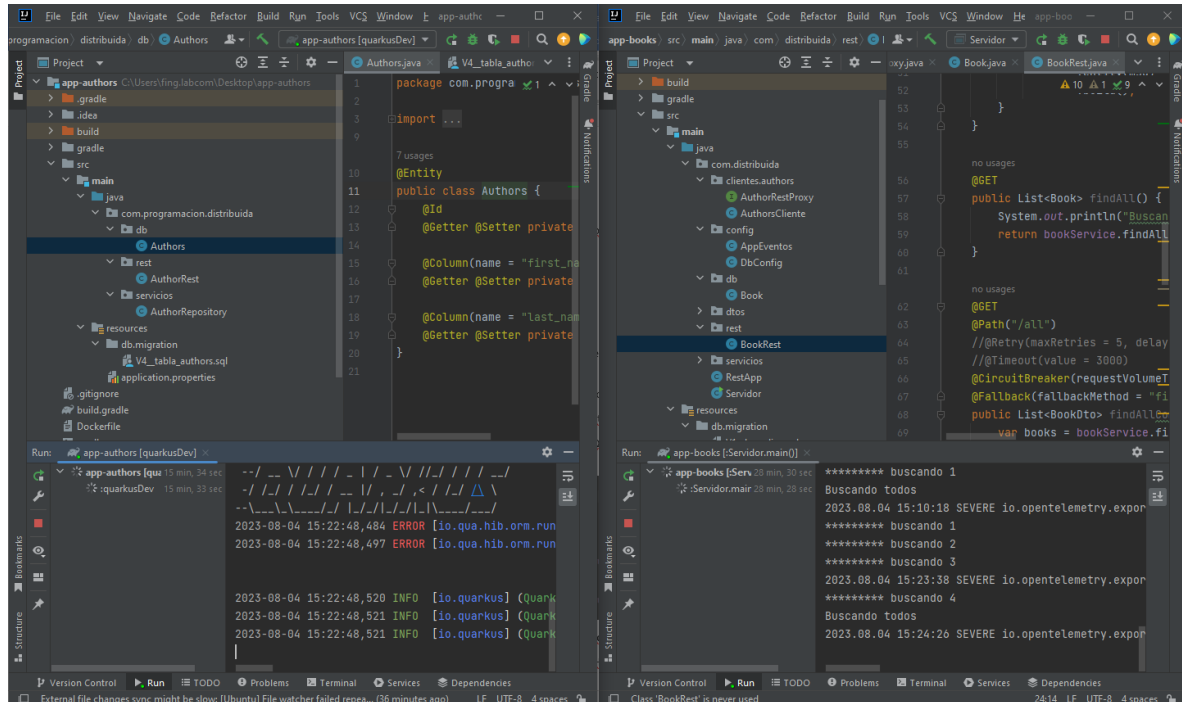
Sistema Operativo	Windows 10	
Java JDK	Versión 17 o 17.0.6	
Gradle	Versión 7.2	
Framework	Quarkus	Helidon
Base de Datos	PostgreSQL	
Software editor de código	IntelliJ IDEA	
Comunicación	REST	

3.4. Implementación de los servicios

Para este ejemplo se empleará un sistema hecho en cursos anteriores con la siguiente característica:

- Tenemos dos servicios el cual app_books consume al servicio de app_authors.
- Si el servicio de app_authors la aplicación app_books no sufrirá pérdida o caída de su servicio.
- Las dos aplicaciones son servicios individuales no depende ninguno del otro.

En la izquierda tenemos el servicio de autores y a la derecha el servicio de libros:



Los dos servicios realizan el CRUD necesario para el funcionamiento, este se le puede llamar mediante los servicios REST ya que este sistema no contiene una interfaz gráfica.

Método	Path	Función
GET	/books/{id}	buscar un libro por ID
GET	/books	listar todos los libros
POST	/books	Insertar un libro
PUT/PATCH	/books/{id}	Actualizar un libro
DELETE	/books/{id}	Eliminar un libro
GET	/authors/{id}	buscar un autor por ID
GET	/authors	listar todos los autores
POST	/authors	Insertar un autor
PUT/PATCH	/authors/{id}	Actualizar un autor
DELETE	/authors/{id}	Eliminar un autor

3.4.1. app-authors

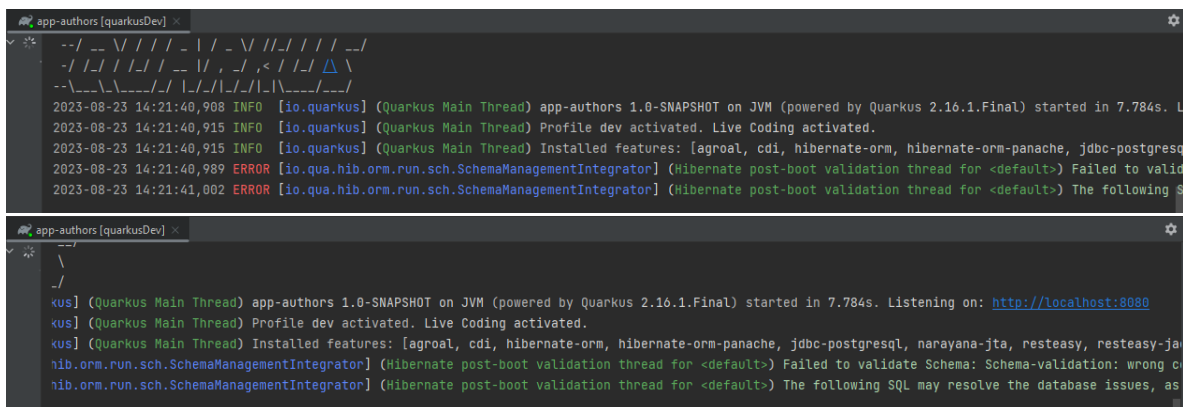
Es un sistema creado en Java Gradle, con servidor Quarkus y con base de datos en PostgreSQL la cual tiene los siguientes atributos en su Tabla de Authors:

```
@Entity
public class Authors {
    @Id
    @Getter @Setter private Long id;

    @Column(name = "first_name")
    @Getter @Setter private String firstName;

    @Column(name = "last_name")
    @Getter @Setter private String lastName;
}
```

Este servicio o aplicación corre en el puerto 8080 predeterminadamente, como se muestra a continuación:



```
app-authors [quarkusDev]
--/ -- \ / / / - / / - \ / / / / / --/
-/ / / / / / - \ / / / / / / / / \
--\ \ \ \ \ / / / / / / / / \ \ \ \
2023-08-23 14:21:40,908 INFO [io.quarkus] (Quarkus Main Thread) app-authors 1.0-SNAPSHOT on JVM (powered by Quarkus 2.16.1.Final) started in 7.784s. L
2023-08-23 14:21:40,915 INFO [io.quarkus] (Quarkus Main Thread) Profile dev activated. Live Coding activated.
2023-08-23 14:21:40,915 INFO [io.quarkus] (Quarkus Main Thread) Installed features: [agroal, cdi, hibernate-orm, hibernate-orm-panache, jdbc-postgresq
2023-08-23 14:21:40,989 ERROR [io.qua.hib.orm.run.sch.SchemaManagementIntegrator] (Hibernate post-boot validation thread for <default>) Failed to valid
2023-08-23 14:21:41,002 ERROR [io.qua.hib.orm.run.sch.SchemaManagementIntegrator] (Hibernate post-boot validation thread for <default>) The following S

app-authors [quarkusDev]
--/ -- \ / / / - / / - \ / / / / / --/
-/ / / / / / - \ / / / / / / / / \
--\ \ \ \ \ / / / / / / / / \ \ \ \
[Quarkus Main Thread] app-authors 1.0-SNAPSHOT on JVM (powered by Quarkus 2.16.1.Final) started in 7.784s. Listening on: http://localhost:8080
[Quarkus Main Thread] Profile dev activated. Live Coding activated.
[Quarkus Main Thread] Installed features: [agroal, cdi, hibernate-orm, hibernate-orm-panache, jdbc-postgresql, narayana-jta, resteasy, resteasy-jax
hib.orm.run.sch.SchemaManagementIntegrator] (Hibernate post-boot validation thread for <default>) Failed to validate Schema: Schema-validation: wrong c
hib.orm.run.sch.SchemaManagementIntegrator] (Hibernate post-boot validation thread for <default>) The following SQL may resolve the database issues, as
```

Y se accede a la lista de todos los autores que se encuentran en la base de datos mediante la siguiente dirección: <http://localhost:8080/authors>



```
[
  {
    id: 1,
    firstName: "nombre1",
    lastName: "ape1"
  },
  {
    id: 2,
    firstName: "nombre2",
    lastName: "ape2"
  }
]
```

3.4.2. app-books

Es un sistema creado en Java Gradle, con servidor Helidon y con base de datos en PostgreSQL la cual tiene los siguientes atributos en su tabla Book.

```

22 usages
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Book {
    private Integer id;
    private String isbn;
    private String title;
    private String author;
    private Double price;
}

```

Este servicio o aplicación corre en el puerto 7001 predeterminadamente, como se muestra a continuación:

```

app-books [Server:main0]
2023.08.23 14:29:55 INFORMATION org.flywaydb.core.internal.database.base.BaseDatabaseType Thread[main,5,main]: Database: jdbc:postgresql://127.0.0.1:54
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbValidate Thread[main,5,main]: Successfully validated 5 migrations (execution time
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Current version of schema "public": 4
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Schema "public" is up to date. No migration necessary
2023.08.23 14:29:56 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Registering JAX-RS Application: HelidonMP
2023.08.23 14:29:57 INFORMATION io.helidon.webserver.NettyWebServer Thread[nioEventLoopGroup-2-1,10,main]: Channel 'default' started: [id: 0xe361e208,
2023.08.23 14:29:57 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Server started on http://localhost:7001 (and all
2023.08.23 14:29:57 INFORMATION io.helidon.common.HelidonFeatures Thread[features-thread,5,main]: Helidon MP 3.1.0 features: [CDI, Config, Db Client, F

```

Y se accede a la lista de todos los libros que se encuentran en la base de datos mediante la siguiente dirección: <http://localhost:7001/books>

```

localhost:7001/books
[
  {
    author: "author1",
    id: 1,
    isbn: "11-11",
    price: 20.0,
    title: "title1"
  },
  {
    author: "author2",
    id: 2,
    isbn: "22-22",
    price: 20.0,
    title: "title2"
  }
]

```

En este servicio se implemento la tolerancia a fallos ya que app-books consume a la app-authors para poder relacionar datos en la cual se implemento Fault Tolerance de la siguiente manera:

- Si el servicio de autores no está listo o este se encuentra caído la aplicación de libros **no deja de funcionar**.
- La información que se mostrará en pantalla será la información de los libros sin los autores.

En la tolerancia a fallos si el método /all no está disponible directamente le mandamos al método findAll() el cual busca todos los libros, pero no relaciona sus libros con sus respectivos Autores.

```

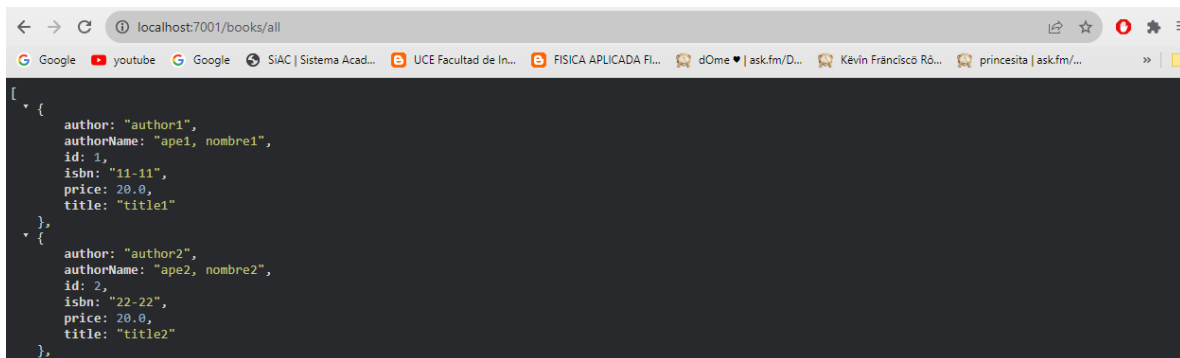
no usages
@GET
@Path("/all")
//@Retry(maxRetries = 5, delay = 400L)
//@Timeout(value = 3000)
@CircuitBreaker(requestVolumeThreshold = 4, failureRatio = 0.5, delay = 2000)
@Fallback(fallbackMethod = "findAll")
public List<BookDto> findAllCompleto() throws Exception {
    var books = bookService.findAll();
}

no usages
@GET
public List<Book> findAll() {
    System.out.println("Buscando todos");
    return bookService.findAll();
}

```

3.4.3. app-books consume app-authors

Para poder acceder a la lista de todos los libros con sus respectivos autores que se encuentran en la base de datos mediante la siguiente dirección: <http://localhost:7001/books/all>



```

[
  {
    author: "author1",
    authorName: "ape1, nombre1",
    id: 1,
    isbn: "11-11",
    price: 20.0,
    title: "title1"
  },
  {
    author: "author2",
    authorName: "ape2, nombre2",
    id: 2,
    isbn: "22-22",
    price: 20.0,
    title: "title2"
  }
]

```

3.5. Pruebas Realizadas

3.5.1. Pruebas Funcionales

- Se realizaron pruebas para verificar que los servicios de Libros y Autores funcionan correctamente y responden a las solicitudes esperadas.
- Se comprobó que el servicio de Libros puede consumir datos del servicio de Autores de manera adecuada y utilizarlos en las operaciones relacionadas con los libros.
- Se evaluaron diferentes escenarios, como crear, leer, actualizar y eliminar registros de libros y autores para asegurarse de que todas las operaciones funcionan correctamente estos se los realizo mediante la aplicación IntelliJ IDEA la cual nos permite realizar el consumo de los métodos GET/POST/PUT/DELETE.

3.5.2. Pruebas de Tolerancia a Fallos

- Se realizaron pruebas para simular escenarios de fallos, como caída del servicio de la app-authors o errores en la red.

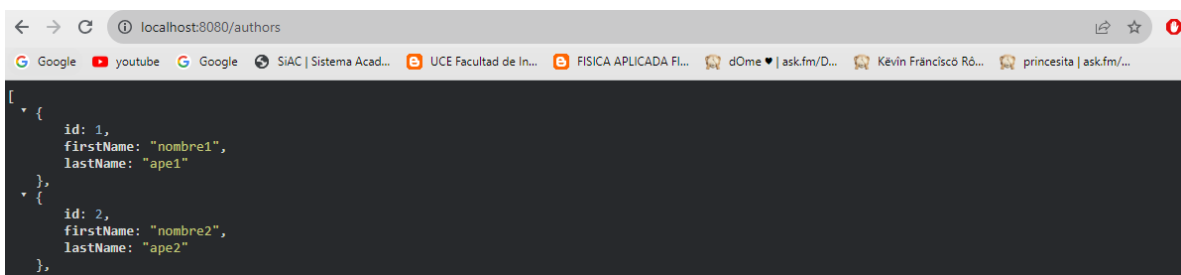
- Se verificó si el sistema es capaz de recuperarse y continuar funcionando correctamente después de que se haya resuelto el problema.

3.6. Resultados de las Pruebas

3.6.1. Pruebas Funcionales

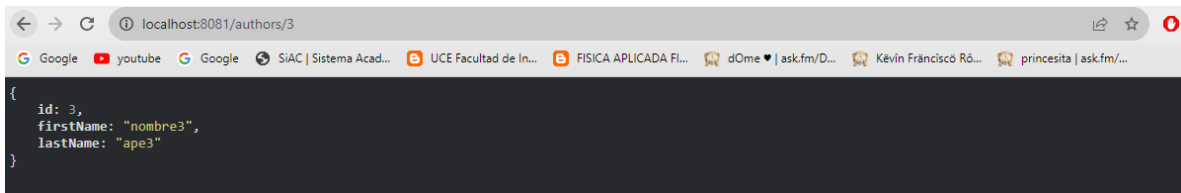
- Todas las pruebas funcionales fueron exitosas. Los servicios de app-books y app-authors respondieron adecuadamente a las solicitudes y las operaciones CRUD se realizaron sin problemas.

En las siguientes imágenes se mostrará algunas pruebas funcionales realizadas del CRUD de cada app-authors y app-books:



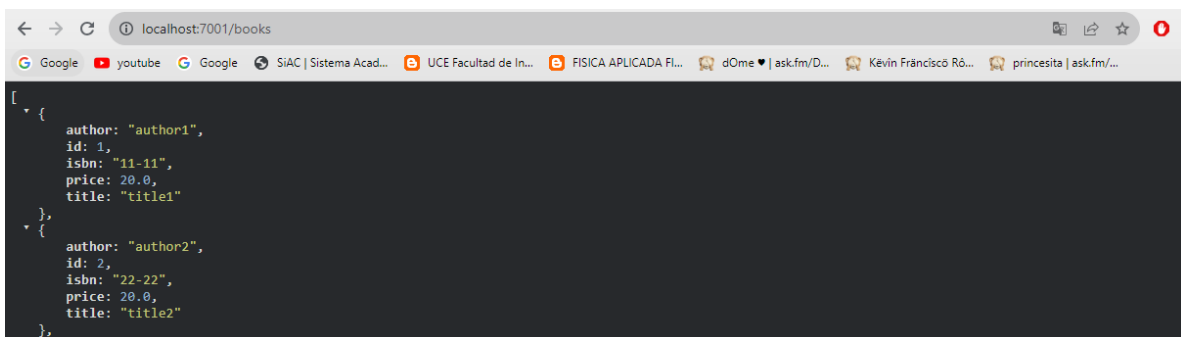
```
[
  {
    id: 1,
    firstName: "nombre1",
    lastName: "ape1"
  },
  {
    id: 2,
    firstName: "nombre2",
    lastName: "ape2"
  }
]
```

En la imagen anterior podemos observar que el método listar todos los autores funciona correctamente.



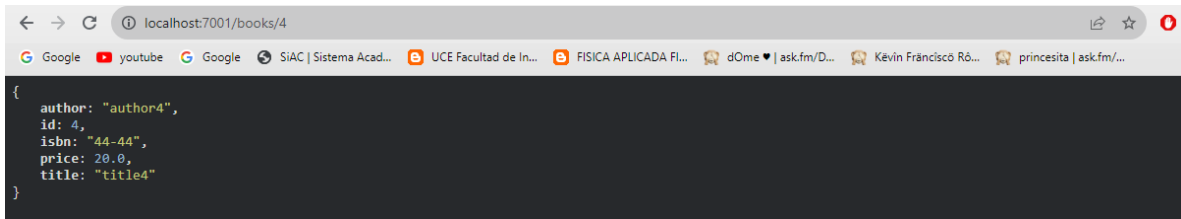
```
{
  id: 3,
  firstName: "nombre3",
  lastName: "ape3"
}
```

En la imagen anterior podemos observar que el método buscar el autor por id=3 funciona correctamente.



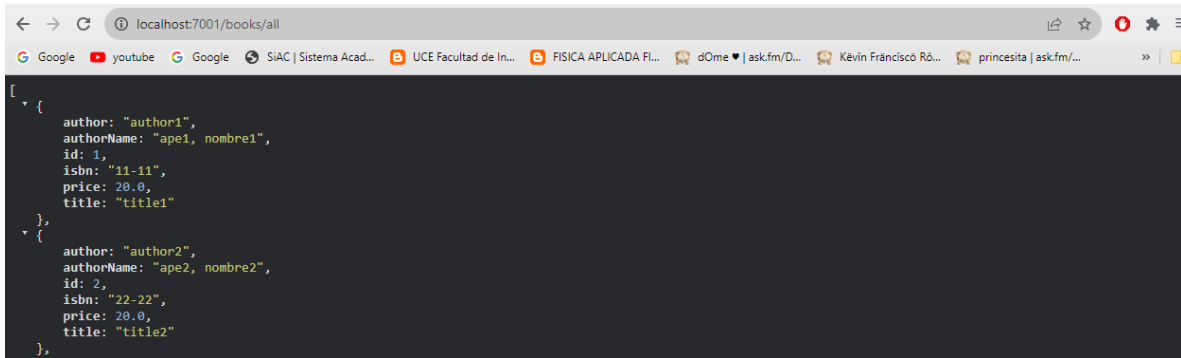
```
[
  {
    author: "author1",
    id: 1,
    isbn: "11-11",
    price: 20.0,
    title: "title1"
  },
  {
    author: "author2",
    id: 2,
    isbn: "22-22",
    price: 20.0,
    title: "title2"
  }
]
```

En la imagen anterior podemos observar que el método listar todos los libros funciona correctamente.



En la imagen anterior podemos observar que el método buscar el libro por id=4 funciona correctamente.

- La integración entre los servicios de app-books y app-authors funcionó correctamente, y los datos de los autores fueron consumidos adecuadamente en las operaciones de libros.



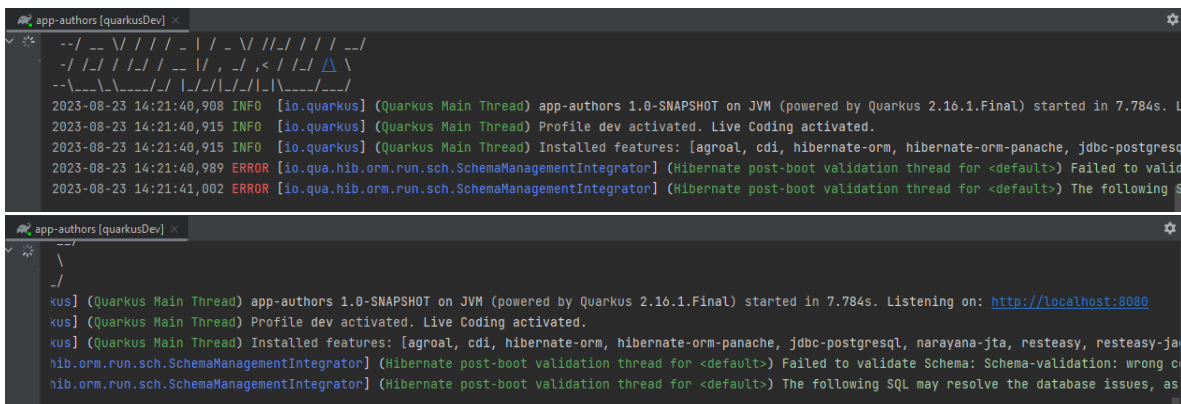
En la imagen podemos observar que la app-books consume correctamente a la app-authors lo que nos da una prueba exitosa de funcionalidad.

3.6.2. Pruebas de Tolerancia a Fallos

El sistema demostró una buena tolerancia a fallos. En escenarios simulados de caída del servicio app-authors, el servicio app-books fue capaz de recuperarse y continuar funcionando después de que el servicio app-authors estuviera disponible nuevamente.

En las siguientes imágenes observaremos un ejemplo de la prueba de tolerancia a fallos que se realizó a nuestro sistema distribuido constituido por app-books y app-authors.

app-authors



app-books

```
app-books [Server.main0] <
2023.08.23 14:29:55 INFORMATION com.zaxxer.hikari.HikariDataSource: HikariPool-1 - Start completed.
2023.08.23 14:29:55 INFORMATION org.flywaydb.core.internal.database.base.BaseDatabaseType Thread[main,5,main]: Database: jdbc:postgresql://127.0.0.1:54
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbValidate Thread[main,5,main]: Successfully validated 5 migrations (execution time
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Current version of schema "public": 4
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Schema "public" is up to date. No migration necessary
2023.08.23 14:29:56 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Registering JAX-RS Application: HelidonMP
2023.08.23 14:29:57 INFORMATION io.helidon.webserver.NettyWebServer Thread[nioEventLoopGroup-2-1,10,main]: Channel '@default' started: [id: 0xe361e208,
2023.08.23 14:29:57 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Server started on http://localhost:7001 (and all
2023.08.23 14:29:57 INFORMATION io.helidon.common.HelidonFeatures Thread[features-thread,5,main]: Helidon MP 3.1.0 features: [CDI, Config, Db Client, F
```

app-books consume app-authors

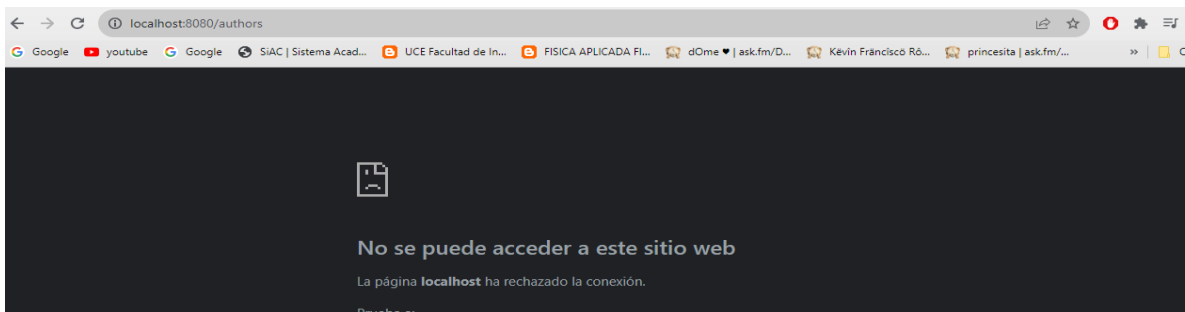
```
localhost:7001/books/all
[
  {
    author: "author1",
    authorName: "ape1, nombre1",
    id: 1,
    isbn: "11-11",
    price: 20.0,
    title: "title1"
  },
  {
    author: "author2",
    authorName: "ape2, nombre2",
    id: 2,
    isbn: "22-22",
    price: 20.0,
    title: "title2"
  }
]
```

Ahora vamos a simular la caída del servicio app-authors:

app-authors

```
app-authors [quarkusDev] <
Execution failed for task ':quarkusDev'.
> Build cancelled while executing task ':quarkusDev'

* Try:
> Run with --stacktrace option to get the stack trace.
> Run with --info or --debug option to get more log output.
> Run with --scan to get full insights.
```

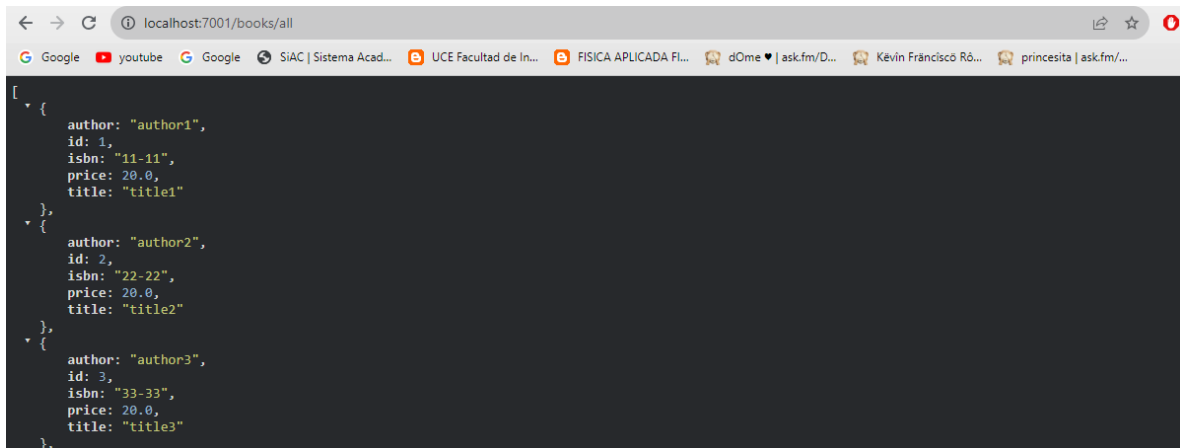


Podemos observar que no existe conexión al app-authors.

app-books

```
app-books [Server.main0] <
2023.08.23 14:29:55 INFORMATION com.zaxxer.hikari.HikariDataSource: HikariPool-1 - Start completed.
2023.08.23 14:29:55 INFORMATION org.flywaydb.core.internal.database.base.BaseDatabaseType Thread[main,5,main]: Database: jdbc:postgresql://127.0.0.1:54
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbValidate Thread[main,5,main]: Successfully validated 5 migrations (execution time
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Current version of schema "public": 4
2023.08.23 14:29:56 INFORMATION org.flywaydb.core.internal.command.DbMigrate Thread[main,5,main]: Schema "public" is up to date. No migration necessary
2023.08.23 14:29:56 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Registering JAX-RS Application: HelidonMP
2023.08.23 14:29:57 INFORMATION io.helidon.webserver.NettyWebServer Thread[nioEventLoopGroup-2-1,10,main]: Channel '@default' started: [id: 0xe361e208,
2023.08.23 14:29:57 INFORMATION io.helidon.microprofile.server.ServerCdiExtension Thread[main,5,main]: Server started on http://localhost:7001 (and all
2023.08.23 14:29:57 INFORMATION io.helidon.common.HelidonFeatures Thread[features-thread,5,main]: Helidon MP 3.1.0 features: [CDI, Config, Db Client, F
```

app-books consume app-authors



```
[
  {
    author: "author1",
    id: 1,
    isbn: "11-11",
    price: 20.0,
    title: "title1"
  },
  {
    author: "author2",
    id: 2,
    isbn: "22-22",
    price: 20.0,
    title: "title2"
  },
  {
    author: "author3",
    id: 3,
    isbn: "33-33",
    price: 20.0,
    title: "title3"
  }
]
```

Podemos observar que el servicio app-books no se cae, aunque ahora ya no muestra los datos de autores los cuales consume del servicio app-authors.

4. Conclusiones

- Los sistemas distribuidos son altamente eficientes, ya que, si un servicio falla, el sistema en su conjunto puede seguir funcionando sin interrupciones significativas. Esta capacidad de mantener la operación a pesar de las fallas individuales es fundamental para garantizar la continuidad y disponibilidad de las aplicaciones y servicios en entornos distribuidos.
- Es fundamental en un sistema distribuido dividir la lógica del negocio en diversos servicios independientes. Esta división nos permite una gestión más eficiente, escalabilidad selectiva y la capacidad de realizar actualizaciones y mejoras de manera más específica, contribuyendo en última instancia a la agilidad y flexibilidad del sistema en su conjunto, así como pudimos observar en nuestros servicios app-books y app-authors.
- En los sistemas distribuidos, existe la flexibilidad de utilizar diversos lenguajes de programación en su creación. Permitiéndonos elegir las herramientas más adecuadas para cada componente del sistema, optimizando la eficiencia y el rendimiento en función de las necesidades específicas de cada parte del sistema distribuido.
- Implementar tolerancia (Fault Tolerance) a fallos en un método es de suma importancia para garantizar que un sistema no se vea afectado por fallos en la conexión con otros componentes. Esta estrategia contribuye a mantener la integridad y continuidad del sistema, permitiendo que las aplicaciones sigan funcionando de manera confiable incluso en situaciones adversas, lo que es esencial para brindar una experiencia de usuario sin interrupciones.

5. Referencias

- What is Fault Tolerance and How it Works? | vSphere | VMware. (2022, August 9). VMware. <https://www.vmware.com/es/products/vsphere/fault-tolerance.html>
- IntelliJ IDEA: el IDE líder para Java y Kotlin. (2021, June 1). JetBrains. <https://www.jetbrains.com/es-es/idea/>
- Gil, J. G. (2023, May 25). 8 características más importantes de PostgreSQL. OpenWebinars.net. <https://openwebinars.net/blog/caracteristicas-importantes-de-postgresql/>
- Tutorial de Java - Características de Java. <http://www.itlp.edu.mx/web/java/Tutorial%20de%20Java/Intro/carac.html>
- Muradas, Y. (2023, April 14). Qué es Gradle: La herramienta para ser más productivo desarrollando. OpenWebinars.net. <https://openwebinars.net/blog/que-es-gradle/>
- Develop a microservices-based RESTful Java application. (2022, April 21). Oracle Help Center. <https://docs.oracle.com/es/solutions/develop-microservice-java-app/index.html#GUID-7B47D70F-9DD2-46FC-8E3D-5C6F8C775E54>
- Quarkus - Supersonic subatomic java. <https://es.quarkus.io/#:~:text=Caracter%C3%ADsticas%20de%20Quarkus&text=Un%20tiempo%20de%20arranque%20incre%C3%ADblemente,orquestaci%C3%B3n%20de%20contenedores%20como%20Kubernetes>
- Harnish, B. (2021). Qué es HTTPS: la guía definitiva para entender cómo funciona. Semrush Blog. https://es.semrush.com/blog/que-es-https/?kw=&cmp=LM_SRCH_DSA_Blog_ES&label=dsa_pagefeed&Network=g&Device=c&utm_content=641222104705&kwid=dsa-1929298977083&cmpid=19249322807&agpid=145221535620&BU=Core&extid=64565394121&adpos=&gclid=Cj0KCQjw3JanBhCPARIsAJpXTx7aRKNzclgGK8a--Jy9zZhQQ12MMLPvYWjeiTdoclk6jHt0J0gV7EaAuR6EALw_wcB