

Práctica Microservicios y Contenedores

Separe el documento de la siguiente forma, esto para dejar al principio una breve descripción del servicio, mostrar las rutas expuestas y un ejemplo de las llamadas a cada uno de los servicios

1. Descripción del servicio
2. Prueba general del servicio
3. Descripción del entorno de desarrollo y Pruebas en el entorno de desarrollo
5. Migrando a Docker y cambios con respecto a las pruebas con el IDE
6. Migrando a Kubernetes con OpenShift OKD

1. Descripción del servicio

Las rutas expuestas corresponden a un CRUD para un documento que esta descrito con la siguiente clase de Java, esto pretende ser una representación de Álbumes musicales

```
@Document("album")
public class Album {

    @JsonIgnoreProperties(ignoreUnknown = true)
    private String id;
    private String nombre;
    private String artista;
    private LocalDate fechaDeSalida;
    private Integer idArtista;
```

Para hacer las operaciones en la base de datos se expusieron las siguientes rutas, sus parametros y sus posibles respuestas se describen en la siguiente seccion

```
/registrarAlbum
/modificarAlbum
/borrarAlbum/{idAlbum}
/buscarAlbum/{idAlbum}
/obtenerTodos
```

2. Prueba general del servicio

Ruta: `"/registrarAlbum"` Metodo: `POST`

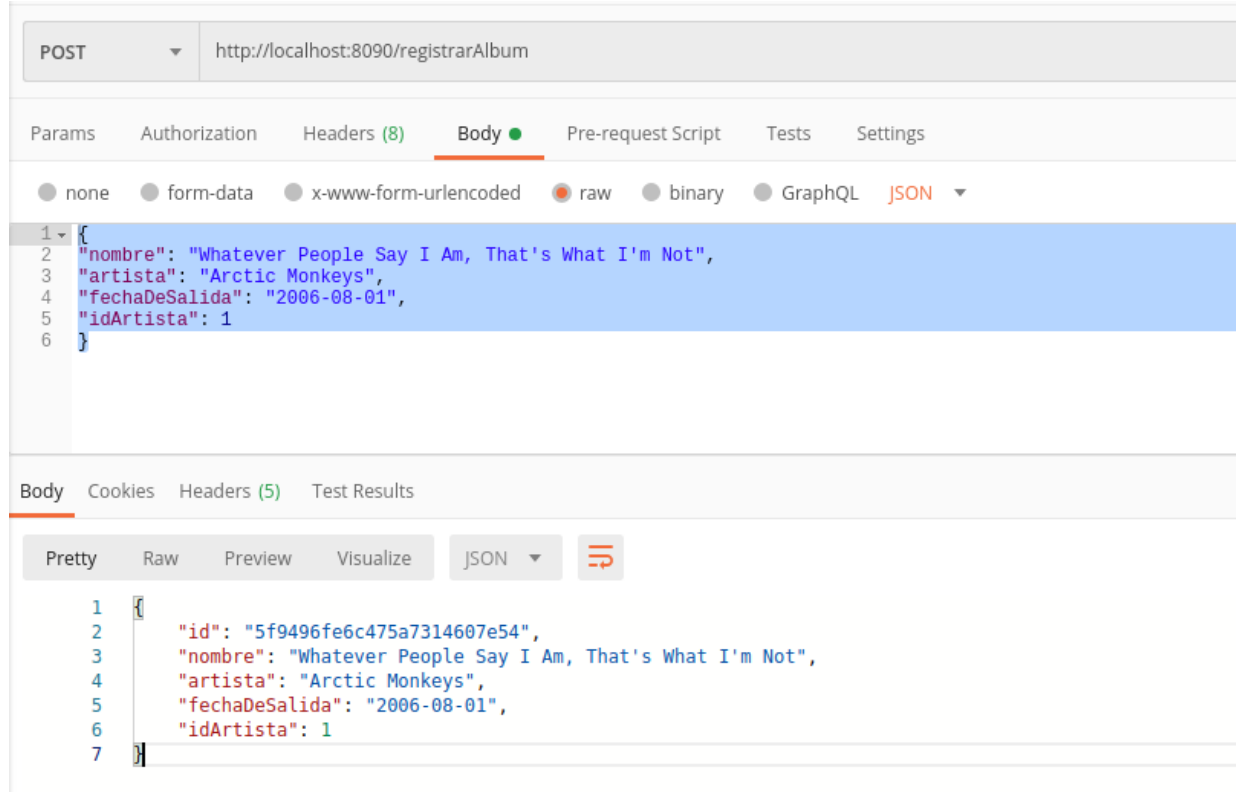
Esta ruta tiene la intención de agregar nuevos objetos a la base de datos, requiere que el cuerpo de la solicitud lleve una representación en JSON del documento Album y como respuesta regresa el nuevo documento agregando el ID que se genera al guardarlo en la base de datos

Formato esperado del JSON en la solicitud

```
{
  "nombre": "Whatever People Say I Am, That's What I'm Not",
  "artista": "Arctic Monkeys",
  "fechaDeSalida": "2006-08-01",
  "idArtista": 1
}
```

Formato esperado de la respuesta

```
{
  "id": "5f9496fe6c475a7314607e54",
  "nombre": "Whatever People Say I Am, That's What I'm Not",
  "artista": "Arctic Monkeys",
  "fechaDeSalida": "2006-08-01",
  "idArtista": 1
}
```



Ruta: "/modificarAlbum" Metodo: PUT

Esta ruta modifica Documentos existentes en la base de datos, muy parecido al servicio anterior espera un JSON que represente al documento Album, regresa el documento extraído de la base de datos con los cambios ya aplicados

Ejemplo del JSON esperado

```
{
  "id": "5f9496fd6c475a7314607e53",
  "nombre": "Franz Ferdinand",
  "artista": "Franz Ferdinand",
  "fechaDeSalida": "2002-08-01",
  "idArtista": 2
}
```

Respuesta en caso de que el documento exista en la base de datos:

The screenshot shows a REST client interface with the following details:

- Method:** PUT
- URL:** http://localhost:8090/modificarAlbum
- Body Type:** JSON
- Request Body:**

```
{
  "id": "5f9496fd6c475a7314607e53",
  "nombre": "Franz Ferdinand",
  "artista": "Franz Ferdinand",
  "fechaDeSalida": "2002-08-01",
  "idArtista": 2
}
```
- Response Body:**

```
{
  "id": "5f9496fd6c475a7314607e53",
  "nombre": "Franz F.",
  "artista": "Franz Ferdinand",
  "fechaDeSalida": "2002-08-01",
  "idArtista": 2
}
```

En caso contrario regresa el código de respuesta 204 el cual indica que no encontró contenido

The screenshot shows the same REST client interface, but with a different response:

- Method:** PUT
- URL:** http://localhost:8090/modificarAlbum
- Body Type:** JSON
- Request Body:**

```
{
  "id": "5f9496fd6c475a73146",
  "nombre": "Franz Ferdinand",
  "artista": "Franz Ferdinand",
  "fechaDeSalida": "2002-08-01",
  "idArtista": 2
}
```
- Status:** 204 No Content

Ruta: /borrarAlbum/{idAlbum} Metodo: DELETE

Esta ruta elimina el documento que corresponda al ID indicado en la propia ruta del servicio, en caso de encontrarlo solo regresa un código 200 y un mensaje que indica que eliminó el documento, en caso contrario regresa el código de respuesta 204

DELETE

http://localhost:8090/borrarAlbum/5f9496fd6c475a7314607e53

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (5)

Test Results

Status: 200 OK Time: 10 ms

Pretty

Raw

Preview

Visualize

Text

1 Objeto Eliminado

Segunda solicitud, el documento ya no existe y regresa un 204

DELETE

http://localhost:8090/borrarAlbum/5f9496fd6c475a7314607e53

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (3)

Test Results

Status: 204 No Content Time: 4 ms

Ruta: “/buscarAlbum/{idAlbum}” Metodo: GET

Esta ruta regresa un JSON que representa el documento cuyo ID coincide con el id ingresado en la ruta, al igual que los ejemplos anteriores en caso de no encontrarlo regresa un código 204

Ejemplo de la respuesta:

```
{
  "id": "5f9496f96c475a7314607e51",
  "nombre": "Whatever People Say I Am, That's What I'm Not",
  "artista": "Arctic Monkeys",
  "fechaDeSalida": "2006-08-01",
  "idArtista": 1
}
```

The screenshot shows a REST client interface with the following components:

- Request Bar:** Method: GET, URL: `http://localhost:8090/buscarAlbum/5f9496f96c475a7314607e51`
- Request Tabs:** Params, Authorization, Headers (6), Body, Pre-request Script, Tests, Settings. The 'Params' tab is selected.
- Query Params Table:**

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Response Bar:** Status: 200 OK, Time: [blank]
- Response Tabs:** Body, Cookies, Headers (5), Test Results. The 'Body' tab is selected.
- Response Format:** Pretty, Raw, Preview, Visualize. The 'Pretty' tab is selected, and the format is set to JSON.
- Response Body:**

```
1 {
2   "id": "5f9496f96c475a7314607e51",
3   "nombre": "Whatever People Say I Am, That's What I'm Not",
4   "artista": "Arctic Monkeys",
5   "fechaDeSalida": "2006-08-01",
6   "idArtista": 1
7 }
```

Ruta: “/obtenerTodos” Metodo: GET

Este servicio solo regresa una lista de todos los documentos en la base de datos, no espera parámetros en la ruta o la solicitud.

```
GET http://localhost:8090/obtenerTodos

Pretty Raw Preview Visualize JSON

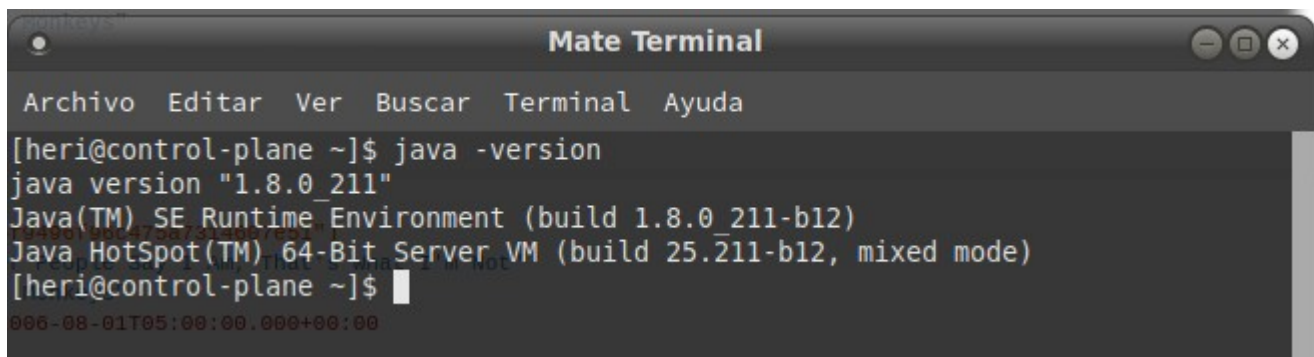
1  [
2    {
3      "id": "5f9496f96c475a7314607e51",
4      "nombre": "Whatever People Say I Am, That's What I'm Not",
5      "artista": "Arctic Monkeys",
6      "fechaDeSalida": "2006-08-01",
7      "idArtista": 1
8    },
9    {
10     "id": "5f94b9646c475a7314607e75",
11     "nombre": "Franz Ferdinand",
12     "artista": "Franz Ferdinand",
13     "fechaDeSalida": "2002-08-01",
14     "idArtista": 2
15   }
16 ]
```


3. Descripción del entorno de desarrollo y Pruebas en el entorno de desarrollo

La aplicación se desarrolló con Java utilizando Spring, las dependencias requeridas inicialmente fueron Boot, Web y Data-Mongo. Hay dependencias que se agregaron posteriormente pero se detallarán a medida que se vayan agregando.

El IDE seleccionado fue la versión de Eclipse que viene con los plugins de Spring (STS)

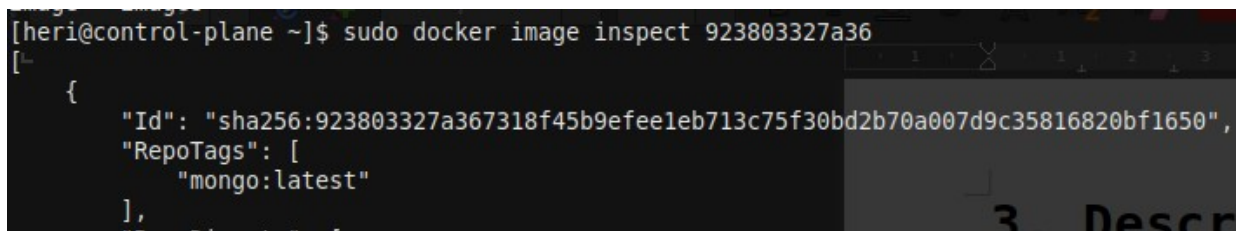
Versión de Java



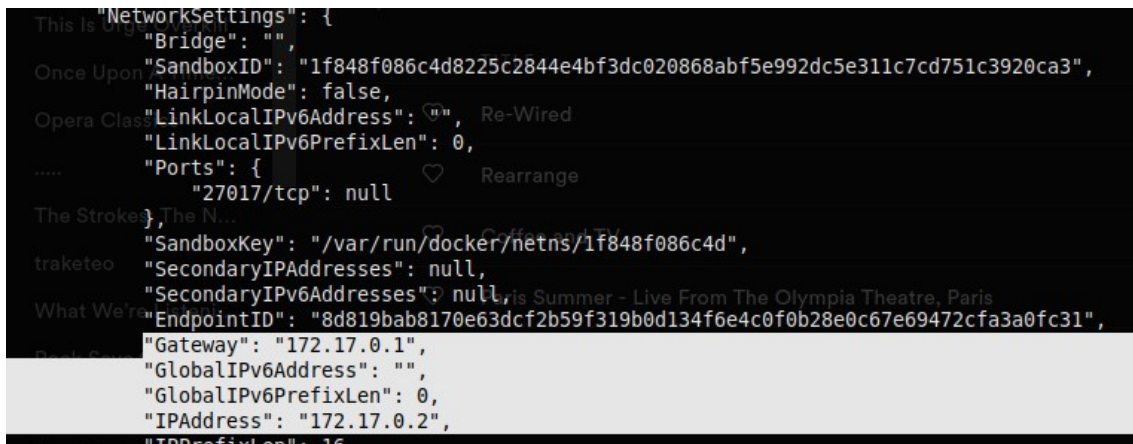
```
Mate Terminal
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
[heri@control-plane ~]$ java -version
java version "1.8.0_211"
Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.211-b12, mixed mode)
[heri@control-plane ~]$
```

La base de datos es MongoDB, actualmente me encuentro tratando de ganar experiencia con Mongo no hay una razón en concreto relacionada a sus prestaciones para dicha decisión.

Para términos de prueba la base de datos no se encuentra instalada localmente, todas las operaciones se llevaron a cabo en un contenedor que está disponible en Docker Hub



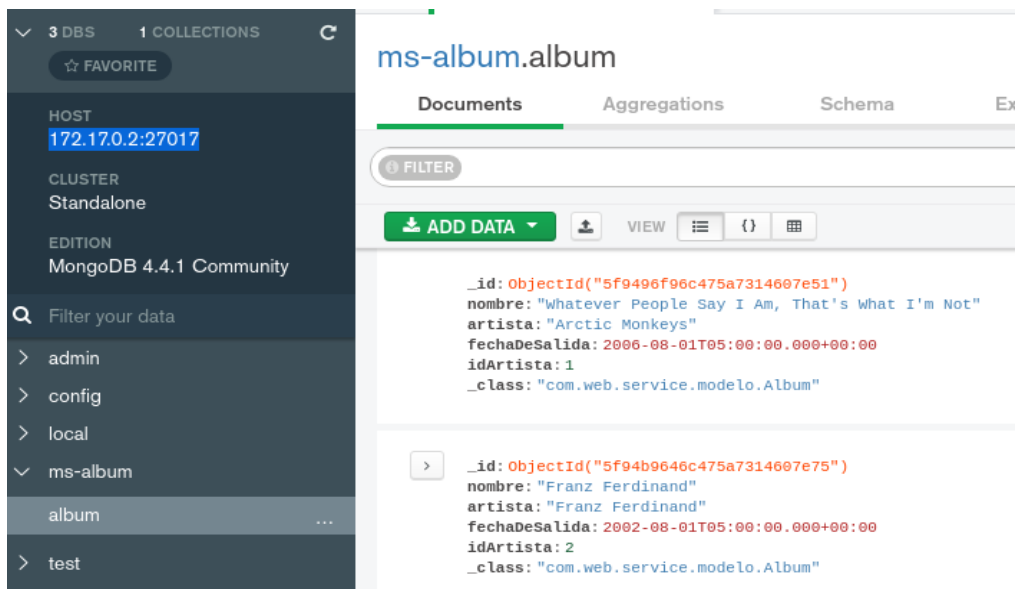
```
[heri@control-plane ~]$ sudo docker image inspect 923803327a36
[{"Id": "sha256:923803327a367318f45b9efee1eb713c75f30bd2b70a007d9c35816820bf1650",
  "RepoTags": [
    "mongo:latest"
  ],
  "ParentId": null,
  "NetworkSettings": {
    "Bridge": "",
    "SandboxID": "1f848f086c4d8225c2844e4bf3dc020868abf5e992dc5e311c7cd751c3920ca3",
    "HairpinMode": false,
    "LinkLocalIPv6Address": "",
    "LinkLocalIPv6PrefixLen": 0,
    "Ports": {
      "27017/tcp": null
    },
    "SandboxKey": "/var/run/docker/netns/1f848f086c4d",
    "SecondaryIPAddresses": null,
    "SecondaryIPv6Addresses": null,
    "EndpointID": "8d819bab8170e63dcf2b59f319b0d134f6e4c0f0b28e0c67e69472cfa3a0fc31",
    "Gateway": "172.17.0.1",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "IPAddress": "172.17.0.2",
    "IPPrefixLen": 16
  }
}]
```



```
"NetworkSettings": {
  "Bridge": "",
  "SandboxID": "1f848f086c4d8225c2844e4bf3dc020868abf5e992dc5e311c7cd751c3920ca3",
  "HairpinMode": false,
  "LinkLocalIPv6Address": "",
  "LinkLocalIPv6PrefixLen": 0,
  "Ports": {
    "27017/tcp": null
  },
  "SandboxKey": "/var/run/docker/netns/1f848f086c4d",
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "8d819bab8170e63dcf2b59f319b0d134f6e4c0f0b28e0c67e69472cfa3a0fc31",
  "Gateway": "172.17.0.1",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "172.17.0.2",
  "IPPrefixLen": 16
}
```

Use compass para verificar el acceso a la base y revisar que los cambios desde el proyecto estuvieran correctos

Con respecto al sistema operativo, todo se encuentra corriendo sobre Fedora 30 y las pruebas a los servicios se hicieron con Postman



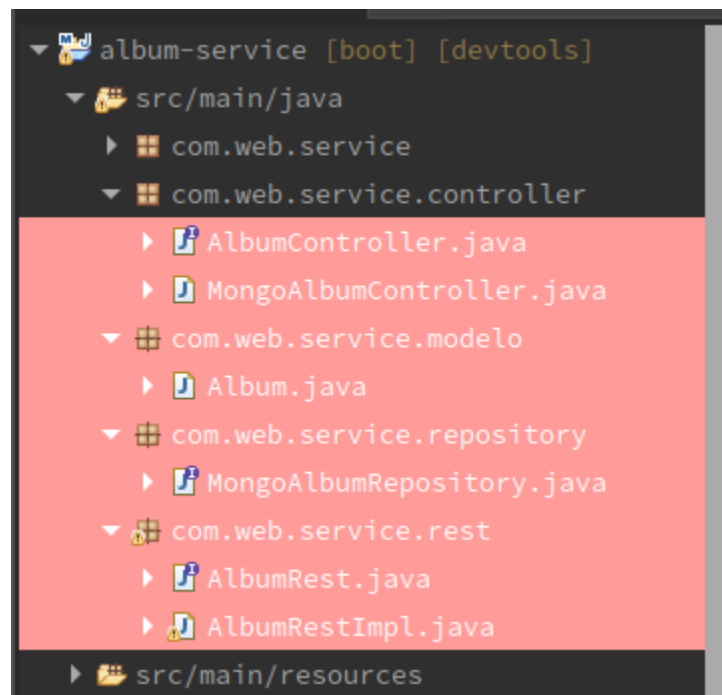
Estructura de la aplicación

El paquete **modelo** contiene la clase de representación para el documento en la base de datos.

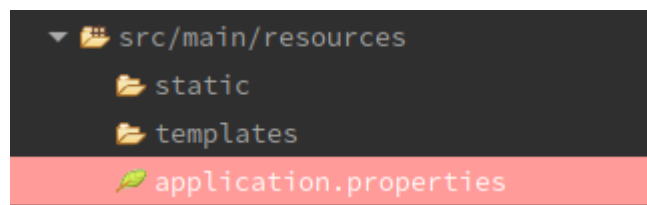
El paquete **repository** contiene la definición de los métodos para las operaciones en la base de datos.

El paquete **controller** contiene la definición y la implementación de los métodos encargados de comunicar los repositorios con los métodos que exponen los servicios

Por último el paquete **rest** contiene los métodos que exponen los servicios.



Los ajustes de la aplicación se encuentran definidos en el archivo properties



```
#puerto indicado en la practica  
server.port=8090
```

```
#datos para la conexion a la base  
spring.data.mongodb.host=172.17.0.2  
spring.data.mongodb.port=27017  
spring.data.mongodb.database=ms-album
```

La aplicación no registra errores y corre en el puerto solicitado para la práctica

```

    ____  __
   / ___/ /_  __
  / _  / __/ / 
 /___/_/____/_/

:: Spring Boot ::      (v2.3.4.RELEASE)


2020-10-24 19:40:20.068 INFO 6009 --- [ restartedMain] com.web.service.AlbumServiceApplication : Starting AlbumServiceApplication on control-plane.minikube.internal with PID 6009 (/home/her
2020-10-24 19:40:20.069 INFO 6009 --- [ restartedMain] com.web.service.AlbumServiceApplication : No active profile set, falling back to default profiles: default
2020-10-24 19:40:20.051 INFO 6009 --- [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
2020-10-24 19:40:20.053 INFO 6009 --- [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 1ms. Found 1 MongoDB repository interfaces.
2020-10-24 19:40:20.096 INFO 6009 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8090 (http)
2020-10-24 19:40:20.096 INFO 6009 --- [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-24 19:40:20.096 INFO 6009 --- [ restartedMain] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-24 19:40:20.098 INFO 6009 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Initializing Spring embedded WebApplicationContext
2020-10-24 19:40:20.098 INFO 6009 --- [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 88 ms
2020-10-24 19:40:20.166 INFO 6009 --- [ restartedMain] org.mongodb.driver.cluster : Cluster created with settings {hosts=[172.17.0.2:27017], mode=SINGLE, requiredClusterType=UN
2020-10-24 19:40:20.167 INFO 6009 --- [72.17.0.2:27017] org.mongodb.driver.connection : Opened connection [connectionId{localValue:58, serverValue:81}] to 172.17.0.2:27017
2020-10-24 19:40:20.167 INFO 6009 --- [72.17.0.2:27017] org.mongodb.driver.cluster : Monitor thread successfully connected to server with description ServerDescription{address=}
2020-10-24 19:40:20.114 INFO 6009 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2020-10-24 19:40:20.150 INFO 6009 --- [ restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-24 19:40:20.171 INFO 6009 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8090 (http) with context path '/'
2020-10-24 19:40:20.173 INFO 6009 --- [ restartedMain] com.web.service.AlbumServiceApplication : Started AlbumServiceApplication in 0.173 seconds (JVM running for 14263.644)
2020-10-24 19:40:20.174 INFO 6009 --- [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged

```

5. Migrando a Docker y cambios con respecto a las pruebas con el IDE

Instalación de Docker:

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
[heri@control-plane ~]$ sudo docker version
Client: Docker Engine - Community
Version:          19.03.12
API version:      1.40
Go version:       go1.13.10
Git commit:       48a66213fe
Built:            Mon Jun 22 15:46:41 2020
OS/Arch:          linux/amd64
Experimental:     false

Server: Docker Engine - Community
Engine:
Version:          19.03.12
API version:      1.40 (minimum version 1.12)
Go version:       go1.13.10
Git commit:       48a66213fe
Built:            Mon Jun 22 15:45:03 2020
OS/Arch:          linux/amd64
Experimental:     false
containerd:
Version:          1.2.13
GitCommit:        7ad184331fa3e55e52b890ea95e65ba581ae3429
runc:
Version:          1.0.0-rc10
GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
docker-init:
Version:          0.18.0
GitCommit:        fec3683
```


Comprobando la aplicación desde el .jar generado por maven

```
[heri@control-plane target]$ java -jar album-service-0.0.1-SNAPSHOT.jar
[Spring Boot :: (v2.3.4.RELEASE)]
2020-10-24 20:25:05.437 INFO 25498 --- [main] com.web.service.AlbumServiceApplication : Starting AlbumServiceApplication v0.0.1-SNAPSHOT on control-plane.minikube.internal with P
-tool-suite-4-4.5.1.RELEASE/album-service/target/album-service-0.0.1-SNAPSHOT.jar started by heri in /home/heri/Documents/workspace-spring-tool-suite-4-4.5.1.RELEASE/album-service/target)
2020-10-24 20:25:05.439 INFO 25498 --- [main] com.web.service.AlbumServiceApplication : No active profile set, falling back to default profiles: default
2020-10-24 20:25:05.791 INFO 25498 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
2020-10-24 20:25:05.824 INFO 25498 --- [main] s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 29ms. Found 1 MongoDB repository interfaces.
2020-10-24 20:25:06.292 INFO 25498 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8090 (http)
2020-10-24 20:25:06.302 INFO 25498 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-24 20:25:06.303 INFO 25498 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-24 20:25:06.353 INFO 25498 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-24 20:25:06.353 INFO 25498 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 870 ms
2020-10-24 20:25:06.455 INFO 25498 --- [main] org.mongodb.driver.cluster : Cluster created with settings {hosts=[172.17.0.2:27017], mode=SINGLE, requiredClusterType=
2020-10-24 20:25:06.495 INFO 25498 --- [72.17.0.2:27017] org.mongodb.driver.connection : Opened connection [connectionId{localValue:1, serverValue:84}] to 172.17.0.2:27017
2020-10-24 20:25:06.499 INFO 25498 --- [72.17.0.2:27017] org.mongodb.driver.cluster : Monitor thread successfully connected to server with description ServerDescription{address
CTED, ok=true, minWireVersion=0, maxWireVersion=9, maxDocumentSize=16777216, logicalSessionTimeoutMinutes=30, roundTripTimeNanos=1958011}
2020-10-24 20:25:06.799 INFO 25498 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-24 20:25:06.994 INFO 25498 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8090 (http) with context path ''
2020-10-24 20:25:06.999 INFO 25498 --- [main] com.web.service.AlbumServiceApplication : Started AlbumServiceApplication in 1.83 seconds (JVM running for 2.168)
2020-10-24 20:25:40.891 INFO 25498 --- [nio-8090-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2020-10-24 20:25:40.891 INFO 25498 --- [nio-8090-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2020-10-24 20:25:40.896 INFO 25498 --- [nio-8090-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 5 ms
2020-10-24 20:25:40.956 INFO 25498 --- [nio-8090-exec-1] org.mongodb.driver.connection : Opened connection [connectionId{localValue:2, serverValue:85}] to 172.17.0.2:27017
```

Generando la imagen

```
[heri@control-plane workspace-spring-tool-suite-4-4.5.1.RELEASE]$ sudo docker build -t msalbum:1 album-service/
[sudo] password for heri:
Sending build context to Docker daemon 22.52MB
Step 1/4 : FROM openjdk:8-jdk-alpine
--> a3562aa0b991
Step 2/4 : ARG JAR_FILE=target/*.jar
--> Using cache
--> 498be6305577
Step 3/4 : COPY ${JAR_FILE} app.jar
--> 619bed00835e
Step 4/4 : ENTRYPOINT ["java","-jar","/app.jar"]
--> Running in d596f7118793
Removing intermediate container d596f7118793
--> e59f4ac5e42b
Successfully built e59f4ac5e42b
Successfully tagged msalbum:1
```

Revisando que exista en el registro

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
msalbum	1	e59f4ac5e42b	About a minute ago	127MB
openshift/origin-control-plane	15:45:03 2020 v3.11	b31a04df54dc	34 hours ago	833MB
openshift/origin-hypershifter	v3.11	d7478e0a7b08	34 hours ago	550MB
openshift/origin-hyperkube	v3.11	2ef8f40bcb24	34 hours ago	510MB
openshift/origin-control-plane	<none>	0779443cfd7	2 weeks ago	833MB
openshift/origin-hyperkube	<none>	2330d8e2bbc7	2 weeks ago	510MB
openshift/origin-hypershifter	31fa3e55e52b890ea95e	dc29bd0bbc6d	2 weeks ago	550MB
boot-ejemplo	latest	f458c8327bed	2 weeks ago	121MB
172.30.1.1:5000/demo-wm/demo-wm	latest	f755f8733144	2 weeks ago	326MB
172.30.1.1:5000/myproject/nginx-exif	5b3839f4323d	b79762ba751d	2 weeks ago	324MB
openshift/origin-node	v3.11	d893143c3500	3 weeks ago	1.17GB
openshift/origin-control-plane	<none>	6ee8aac7bba6	3 weeks ago	833MB
openshift/origin-haproxy-router	v3.11	2d44eb806d1d	3 weeks ago	412MB
openshift/origin-docker-builder	v3.11	022453764762	3 weeks ago	463MB
openshift/origin-deployer	v3.11	a3662f343243	3 weeks ago	384MB
openshift/origin-hypershifter	<none>	a2d150544720	3 weeks ago	550MB
openshift/origin-hyperkube	<none>	8aa01bf4c44f	3 weeks ago	510MB

levantando el contenedor

```
[heri@control-plane workspace-spring-tool-suite-4-4.5.1.RELEASE]$ sudo docker run e59f4ac5e42b

:: Spring Boot ::
(v2.3.4.RELEASE)

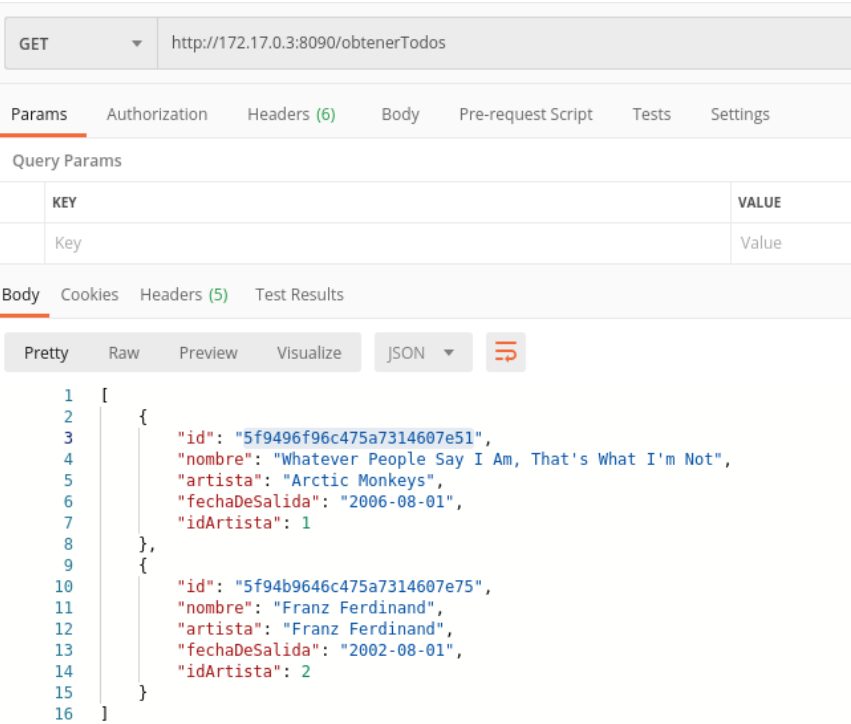
2020-10-25 01:51:35.444 INFO 1 --- [main] com.web.service.AlbumServiceApplication : Starting AlbumServiceApplication v0.0.1-SNAPSHOT on e93381789e6b with PID 1
2020-10-25 01:51:35.446 INFO 1 --- [main] com.web.service.AlbumServiceApplication : No active profile set, falling back to default profiles: default
2020-10-25 01:51:35.861 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data MongoDB repositories in DEFAULT mode.
2020-10-25 01:51:35.928 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 62ms. Found 1 MongoDB repository
2020-10-25 01:51:36.470 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8090 (http)
2020-10-25 01:51:36.483 INFO 1 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-25 01:51:36.483 INFO 1 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-25 01:51:36.541 INFO 1 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-25 01:51:36.541 INFO 1 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1026 ms
2020-10-25 01:51:36.744 INFO 1 --- [72.17.0.2:27017] org.mongodb.driver.connection : Cluster created with settings {hosts=[172.17.0.2:27017], mode=SINGLE, require
2020-10-25 01:51:36.749 INFO 1 --- [72.17.0.2:27017] org.mongodb.driver.connection : Opened connection [connectionId(localValue:1, serverValue:86)] to 172.17.0.2
2020-10-25 01:51:36.749 INFO 1 --- [72.17.0.2:27017] org.mongodb.driver.connection : Monitor thread successfully connected to server with description ServerDescr
, ok=true, minWireVersion=0, maxWireVersion=9, maxDocumentSize=16777216, logicalSessionTimeoutMinutes=30, roundTripTimeNanos=3347219}
2020-10-25 01:51:37.139 INFO 1 --- [main] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-25 01:51:37.292 INFO 1 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8090 (http) with context path ''
2020-10-25 01:51:37.299 INFO 1 --- [main] com.web.service.AlbumServiceApplication : Started AlbumServiceApplication in 2.2 seconds (JVM running for 2.759)
```

En este punto se encuentran corriendo dos contenedores, uno con la aplicación y otro más con la base de datos, no hay un mapeo de puertos, se omite para poder probar la aplicación directo desde el contenedor

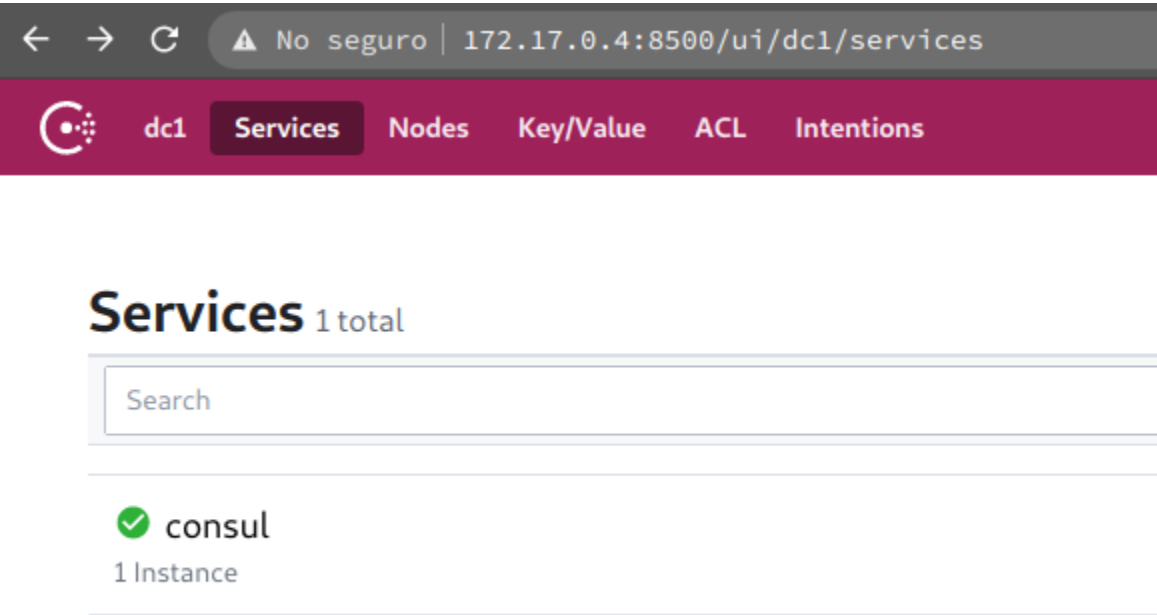
```
[heri@control-plane workspace-spring-tool-suite-4-4.5.1.RELEASE]$ sudo docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS	NAMES	
5cd4388a2800	e59f4ac5e42b	"java -jar /app.jar"	7 seconds ago
Up 6 seconds		thirsty_darwin	
a6b11fa2935c	923803327a36	"docker-entrypoint.s..."	8 hours ago
Up 8 hours	27017/tcp	stoic_grothendieck	

Lanzando una nueva solicitud desde Postman al contenedor para revisar que se encuentre funcionando correctamente



Se agregó una nueva imagen que despliega una instancia de Consul, el cual funciona como un servidor de nombres, esto para tener disponibles las validaciones que hace al estado del servicio y posteriormente exponerlo como un microservicio



Cambios al proyecto para funcinar con Consul

Se agregó la dependencia **cloud.consul** y se agregaron las siguientes propiedades en el archivo properties

```
#valores requeridos para registrar con consul
spring.cloud.consul.host=172.17.0.4
spring.cloud.consul.port=8500
spring.application.name=ms-album
spring.cloud.consul.discovery.instance-id=${spring.application.name}:${random.value}
```

Debido a que no se agregó spring-actuator los servicios para consultar el estado del api aún no se encuentran disponibles

6. Migrando a Kubernetes con OpenShift OKD

Proceso de importación

1. Login en el registro Docker interno de OKD

```
[heri@control-plane ~]$ sudo docker login -u developer -p W-RYB4YYWwVAcOJMjeN94J2WM_i
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
```

2. Crear la referencia dentro del registro de OKD

```
[heri@control-plane ~]$ sudo docker tag msalbum:v3 172.30.1.1:5000/openshift/java-app
```


3. Hacer el push de la imagen dentro de OKD

```
[heri@control-plane ~]$ sudo docker push 172.30.1.1:5000/openshift/java-app
The push refers to repository [172.30.1.1:5000/openshift/java-app]
80db9e2dece1: Pushed
ceaf9e1ebef5: Pushed
9b9b7f3d56a0: Pushed
f1b5933fe4b5: Pushed
latest: digest: sha256:33d0c14b00c8661e3c9643ccd2c81c70d38aa05bc2517f45385b1a3dfe7caf9f size: 1159
```

Posterior a estos pasos, crear un proyecto y hacer el deploy se obtiene lo siguiente

The screenshot shows the OpenShift OKD web console interface. On the left is a sidebar with navigation links: Overview, Applications, Builds, Resources, Storage, Monitoring, and Catalog. The main panel displays the deployment details for 'java-app-1', which was created an hour ago. The deployment is associated with the 'java-app' application and the 'openshift.io/deployment-config.name' configuration. The status is 'Active', and the deployment reason is 'config change'. The selectors are 'app=java-app', 'deployment=java-app-1', and 'deploymentconfig=java-app'. The replicas section shows '2 current / 2 desired'. A circular progress indicator shows '2 pods'. Below the deployment details, there is a 'Template' section with a warning: 'Container java-app does not have health checks to ensure your application is running correctly. Add Health Checks'. At the bottom, the 'Containers' section lists the 'java-app' container with the image 'openshift/java-app'.

En Consul podemos ver que ambos pods se registraron como instancias diferentes del servicio


 dc1 Services Nodes Key/Value ACL Intentions


[All Services](#)


ms-album


Instances Intentions Routing Tags


ms-album-190a66646aaa994a7ebb2f7b9008296f

 All service checks failing


 All node checks passing


 462748513ca8


 java-app-1-x6lsw:8090


 secure=false


ms-album-71d1fae0abffec153f66b902780549ed

 All service checks failing

 All node checks passing

 462748513ca8

 java-app-1-f2jhd:8090

 secure=false

Request Individual a cada uno de los Pods

Primer Pod

java-app-1-f2jhd created an hour ago

app java-app deployment java-app-1 deploymentconfig java-app

Details Environment Logs Terminal Events

Status

Status: Running

Deployment: java-app, #1

IP: 172.17.0.13

Node: localhost (192.168.100.23)

Restart Policy: Always

Container java-app

State: Running since Oct 24, 2020 11:58:31 PM

Ready: true

Restart Count: 0

Untitled Request

GET http://172.17.0.13:8090/obtenerTodos

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY VALUE

Key Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   {
3     "id": "5f9496f96c475a7314607e51",
4     "nombre": "Whatever People Say I Am, That's What I'm Not",
5     "artista": "Arctic Monkeys",
6     "fechaDeSalida": "2006-08-01",
7     "idArtista": 1
8   },
9   {
10    "id": "5f94b9646c475a7314607e75",
11    "nombre": "Franz Ferdinand",
12    "artista": "Franz Ferdinand",
13    "fechaDeSalida": "2002-08-01",
14    "idArtista": 2
15  }
16 }
```

Segundo Pod

Pods > java-app-1-x6lsw

java-app-1-x6lsw created an hour ago

app java-app deployment java-app-1 deploymentconfig java-app

Details Environment Logs Terminal Events

Status

Status: Running

Deployment: java-app, #1

IP: 172.17.0.14

Node: localhost (192.168.100.23)

Restart Policy: Always

Container java-app

State: Running since Oct 24, 2020 11:53:30 PM

Ready: true

Restart Count: 0

Untitled Request

GET http://172.17.0.14:8090/buscarAlbum/5f9496f96c475a7314607e51

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY VALUE

Key Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "5f9496f96c475a7314607e51",
3   "nombre": "Whatever People Say I Am, That's What I'm Not",
4   "artista": "Arctic Monkeys",
5   "fechaDeSalida": "2006-08-01",
6   "idArtista": 1
7 }
```

Al reducir las réplicas a uno, Consul solo conserva la instancia que corresponde al Pod que sigue corriendo

java-app-1

created an hour ago

app

java-app

openshift.io/deployment-config.name

java-app

Details

Environment

Logs

Events

Status:

Deployment Config:

Status Reason:

Selectors:

Replicas:

🔄 Active

java-app

config change

app=java-app
deployment=java-app-1
deploymentconfig=java-app

1 current / 1 desired

Template

Container java-app does not have health checks to ensure your application is running correctly. [Add Health Checks](#)

dc1

Services

Nodes

Key/Value

ACL

Intentions

< All Services

ms-album

Instances

Intentions

Routing

Tags

ms-album-190a66646aaa994a7ebb2f7b9008296f

All service checks failing

All node checks passing

462748513ca8

java-app-1-x6lsw:8090

secure=false

Agregando Spring-Actuator

tras agregar la siguiente dependencia, se agregan las rutas para revisar el estado del servicio

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

GET http://localhost:8090/actuator

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "_links": {
3     "self": {
4       "href": "http://localhost:8090/actuator",
5       "templated": false
6     },
7     "health": {
8       "href": "http://localhost:8090/actuator/health",
9       "templated": false
10    },
11    "health-path": {
12      "href": "http://localhost:8090/actuator/health/{*path}",
13      "templated": true
14    },
15    "info": {
16      "href": "http://localhost:8090/actuator/info",
17      "templated": false
18    }
19  }
20 }
```

GET http://localhost:8090/actuator/health

Params Authorization Headers (6) Body Pre-request Script

Query Params

KEY
Key

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "status": "UP"
3 }
```