Taller de Modularización con Virtualización e Introducción a Docker y a AWS

Daniel Felipe Walteros Trujillo

26 de Febrero del 2021

Profesor:
Luis Daniel Benavides Navarro

Arquitecturas Empresariales

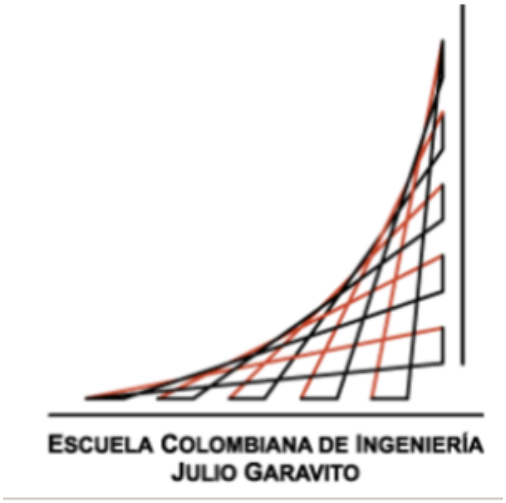


Tabla de Contenido

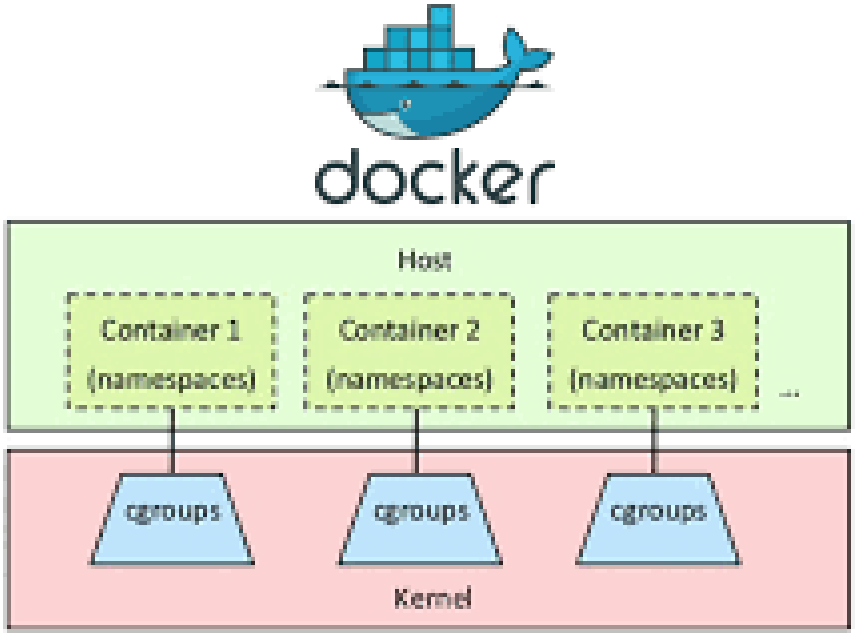
1 Prerrequisitos

Para el desarrollo del programa se utilizó Maven como una herramienta para la gestión del ciclo de vida del software, el código fue desarrollado con el lenguaje de programación Java, por lo tanto para su ejecución se requiere:

- Java versión 8 o superior.
- Maven versión 3.5 o superior.
- Docker versión 19 o superior.

2 Introducción

La tecnología Docker usa el kernel de Linux y sus funciones, como Cgroups y namespaces, para segregar los procesos, de modo que puedan ejecutarse de manera independiente. El propósito de los contenedores es esta independencia: la capacidad de ejecutar varios procesos y aplicaciones por separado para hacer un mejor uso de su infraestructura y, al mismo tiempo, conservar la seguridad que tendría con sistemas separados [6].



Amazon Web Services (AWS) es la plataforma en la nube más adoptada y completa en el mundo, que ofrece más de 200 servicios integrales de centros de datos a nivel global. Millones de clientes, incluso las empresas emergentes que crecen más rápido, las compañías más grandes y los organismos gubernamentales líderes, están usando AWS para reducir los costos, aumentar su agilidad e innovar de forma más rápida [2].

Amazon Elastic Compute Cloud (Amazon EC2) es un servicio web que proporciona capacidad informática en la nube segura y de tamaño modificable. Está diseñado para simplificar el uso de la informática en la nube a escala web para los desarrolladores. La sencilla interfaz de servicios web de Amazon EC2 permite obtener y configurar capacidad con una fricción mínima. Proporciona un control completo sobre los recursos informáticos y puede ejecutarse en el entorno informático acreditado de Amazon [1].

MongoDB es una base de datos distribuida, basada en documentos y de uso general que ha sido diseñada para desarrolladores de aplicaciones modernas y para la era de la nube. Ninguna otra ofrece un nivel de productividad de uso tan alto [4].

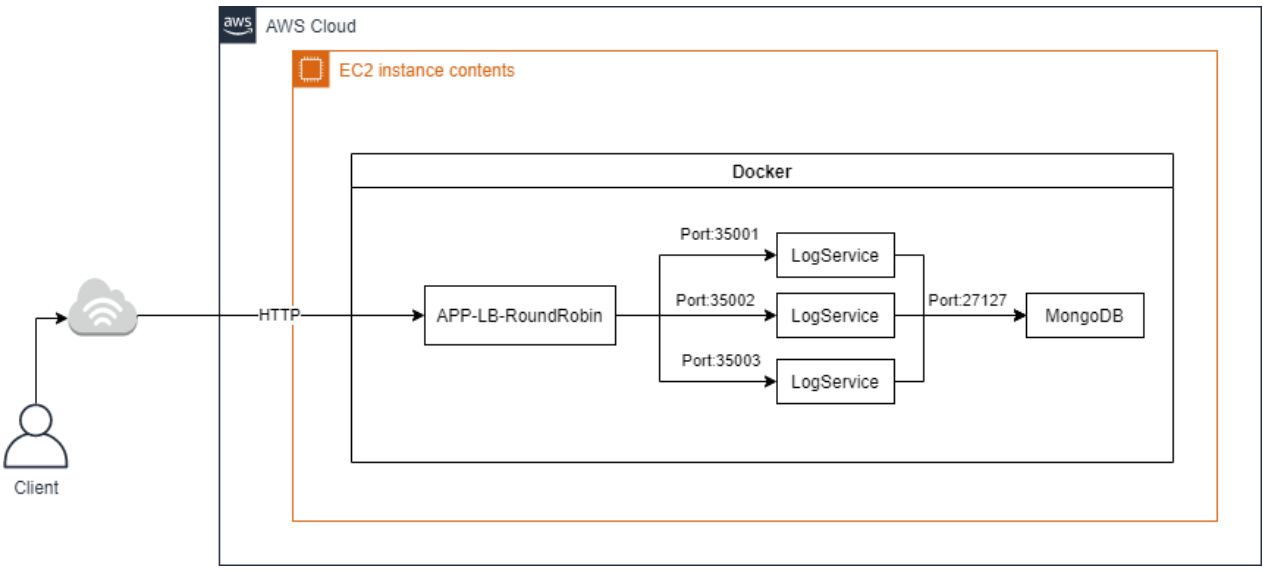
Round-robin es un método para seleccionar todos los abstractos en un grupo de manera equitativa y en un orden racional, normalmente comenzando por el primer elemento de la lista hasta llegar al último y empezando de nuevo desde el primer elemento [8].

Postman es una plataforma de colaboración para el desarrollo de APIs. Las características de Postman simplifican cada paso de la creación de una API y agilizan la colaboración para que pueda crear mejores APIs, más rápido [5].

Este ejercicio consiste en el desarrollo de una aplicación distribuida dentro de un contenedor Docker que se conecta a una MongoDB y reparte la carga por medio del algorimto Round Robin, se desplegó en AWS usando EC2.

3 Diseño

3.1 Diagrama de Despliegue



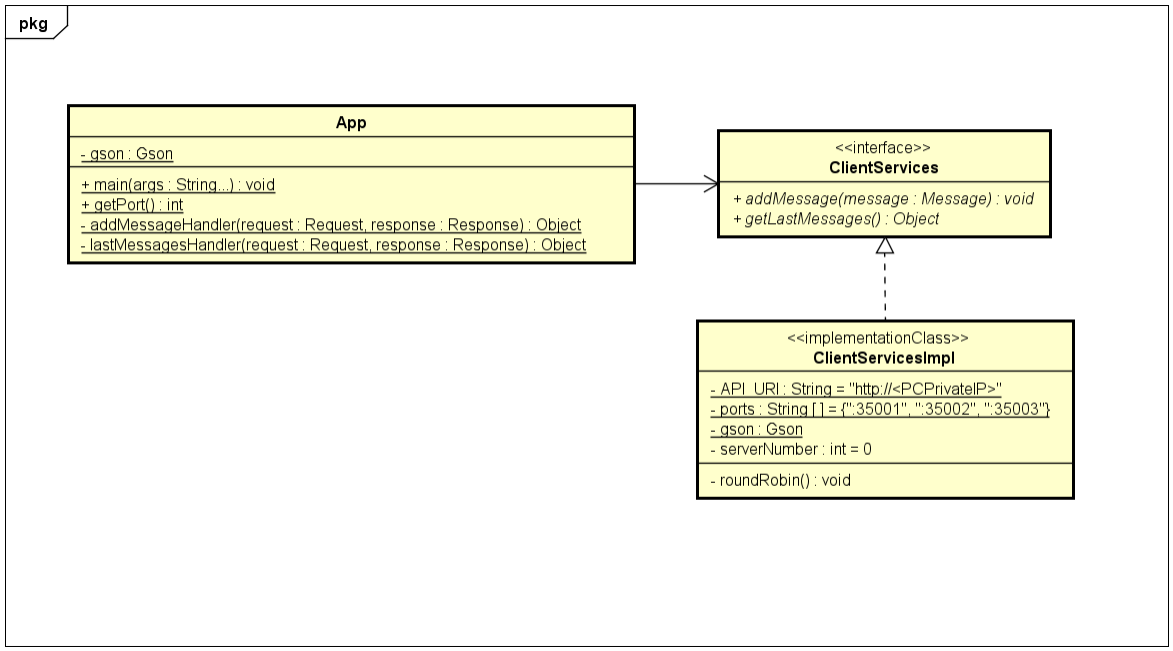
El servicio MongoDB es una instancia de MongoDB[4] corriendo en un container de docker

en una máquina virtual de EC2[1].

LogService es un servicio REST que recibe una cadena, la almacena en la base de datos y responde en un objeto JSON con las 10 ultimas cadenas almacenadas en la base de datos y la fecha en que fueron almacenadas.

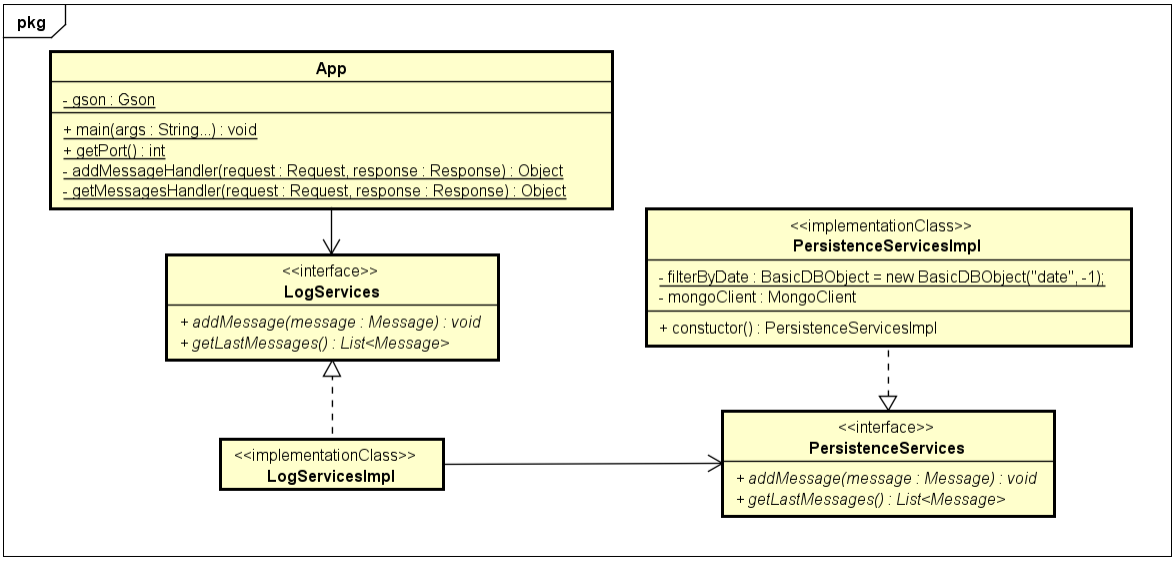
La aplicación web APP-LB-RoundRobin está compuesta por un cliente web y al menos un servicio REST. El cliente web tiene un campo y un botón y cada vez que el usuario envía un mensaje, este se lo envía al servicio REST y actualiza la pantalla con la información que este le regresa en formato JSON. El servicio REST recibe la cadena e implementa un algoritmo de balanceo de cargas de Round Robin[8], delegando el procesamiento del mensaje y el retorno de la respuesta a cada una de las tres instancias del servicio LogService.

3.2 Diagrama de Clases APP-LB-RoundRobin



El programa APP-LB-RoundRobin utiliza la interfaz ClientServices para distribuir la carga de añadir y consultar mensajes, la implementación de esta interfaz realiza por medio de una URI y una lista de puertos quemada la repartición de esta carga entre diversos nodos Log Service utilizando el algoritmo roundRobin[8].

3.3 Diagrama de Clases LogService

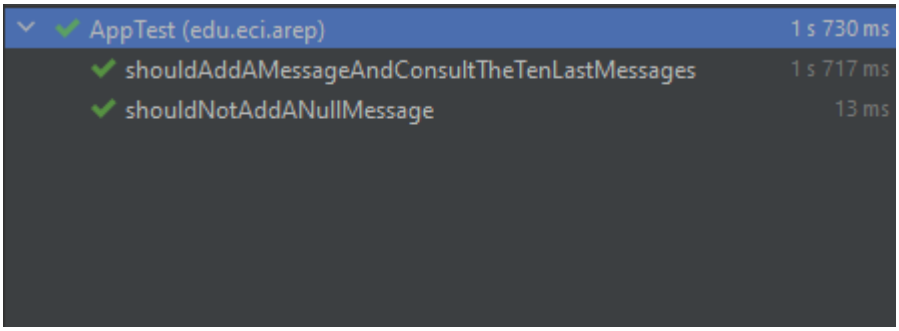


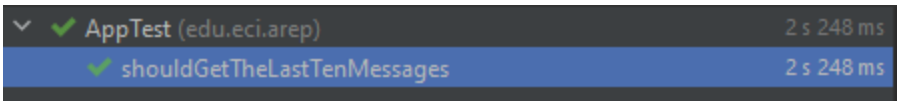
El programa LogService utiliza la interfaz LogServices para agregar y consultar mensajes, la implementación de esta clase maneja las posibles excepciones y a su vez se conecta con la interfaz PersistenceServices, la implementación de esta otra interfaz se conecta a la base de datos MongoDB por medio de un MongoClient[3] y consulta los ultimos diez mensajes por medio de un filtro de clase BasicDBObject[7].

4 Pruebas

El programa fue probado con tres pruebas unitarias de JUnit donde se contemplaron los siguientes casos:

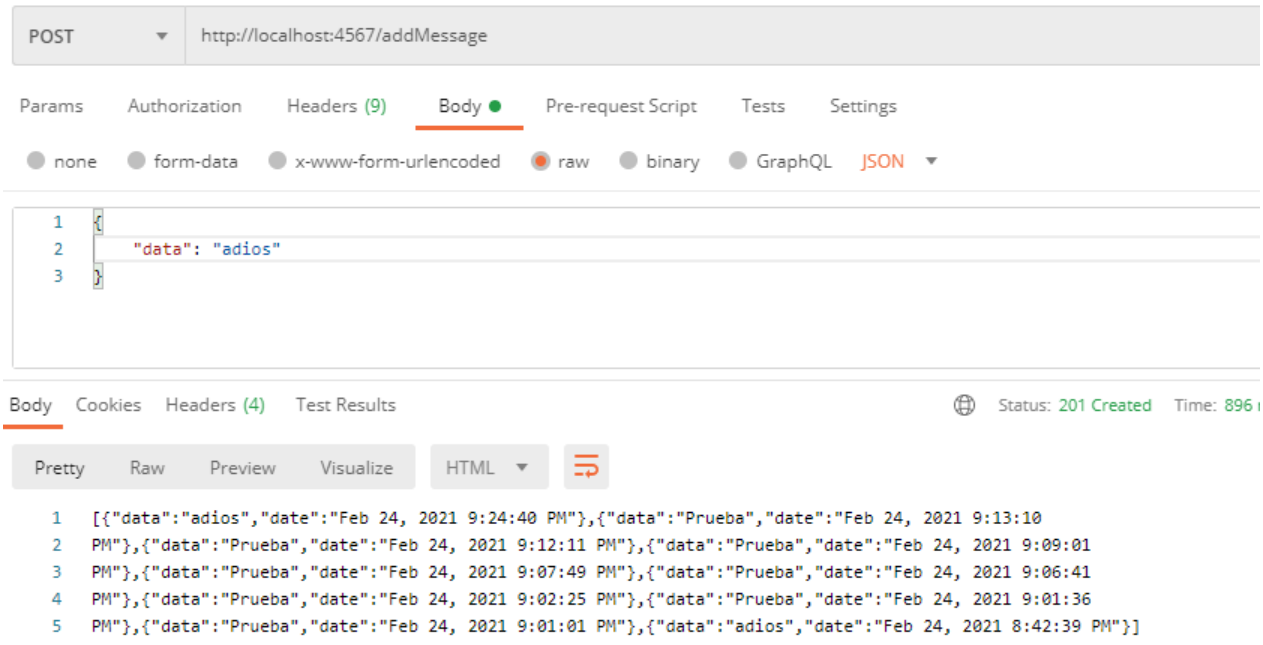
- Agregar un mensaje nuevo.
- Agregar un mensaje con contenido nulo.
- Consultar los ultimos diez mensajes.





4.1 Prueba Manual Local

Al principio se realizaron pruebas realizando peticiones al servicio por medio de Postman[5].



Después se revisaron los logs de los dos servicios para validar su funcionamiento.

- Log Service Logs:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Message: adios created on MongoDB at: Wed Feb 24 21:24:40 COT 2021
Last Ten Messages Selected From MongoDB at: Wed Feb 24 20:42:39 COT 2021
Message: hola created on MongoDB at: Wed Feb 24 21:25:29 COT 2021
Last Ten Messages Selected From MongoDB at: Wed Feb 24 21:01:01 COT 2021
Message: mensaje de prueba created on MongoDB at: Wed Feb 24 21:25:39 COT 2021
Last Ten Messages Selected From MongoDB at: Wed Feb 24 21:01:36 COT 2021
Message: prueba created on MongoDB at: Wed Feb 24 21:25:50 COT 2021
Last Ten Messages Selected From MongoDB at: Wed Feb 24 21:02:25 COT 2021
```

- Round Robin Logs:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
POST Request to: http://192.168.1.20:35001/addMessage made at: 2021-02-25T14:21:54.880
GET Request to: http://192.168.1.20:35002/messages made at: 2021-02-25T14:21:54.971
POST Request to: http://192.168.1.20:35003/addMessage made at: 2021-02-25T14:22:11.193
GET Request to: http://192.168.1.20:35001/messages made at: 2021-02-25T14:22:11.268
POST Request to: http://192.168.1.20:35002/addMessage made at: 2021-02-25T14:23:01.360
GET Request to: http://192.168.1.20:35003/messages made at: 2021-02-25T14:23:01.415
POST Request to: http://192.168.1.20:35001/addMessage made at: 2021-02-25T14:23:07.300
GET Request to: http://192.168.1.20:35002/messages made at: 2021-02-25T14:23:07.362
POST Request to: http://192.168.1.20:35003/addMessage made at: 2021-02-25T14:23:12.719
GET Request to: http://192.168.1.20:35001/messages made at: 2021-02-25T14:23:12.784
POST Request to: http://192.168.1.20:35002/addMessage made at: 2021-02-25T14:23:16.429
GET Request to: http://192.168.1.20:35003/messages made at: 2021-02-25T14:23:16.492
```

4.2 Prueba Manual En AWS

Se revisaron los logs de los dos servicios para validar su funcionamiento.

- Log Service Logs:

```
[ec2-user@ip-172-31-81-62 ~]$ docker logs arep-lab5_web_2
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Last Ten Messages Selected From MongoDB at: 2021-02-26T11:38:16.892
Message: de created on MongoDB at: Fri Feb 26 11:38:29 COT 2021
Last Ten Messages Selected From MongoDB at: 2021-02-26T11:38:38.300
Message: es created on MongoDB at: Fri Feb 26 11:38:51 COT 2021
Last Ten Messages Selected From MongoDB at: 2021-02-26T11:38:55.826
```

- Round Robin Logs:

```
[ec2-user@ip-172-31-81-62 AREP-Lab5]$ docker logs -f loadbalancer
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
POST Request to: http://ip-172-31-81-62.ec2.internal:35001/addMessage made at: 2021-02-26T15:48:37.880
GET Request to: http://ip-172-31-81-62.ec2.internal:35002/messages made at: 2021-02-26T15:48:38.219
POST Request to: http://ip-172-31-81-62.ec2.internal:35003/addMessage made at: 2021-02-26T15:48:52.859
GET Request to: http://ip-172-31-81-62.ec2.internal:35001/messages made at: 2021-02-26T15:48:52.989
POST Request to: http://ip-172-31-81-62.ec2.internal:35002/addMessage made at: 2021-02-26T15:48:54.912
GET Request to: http://ip-172-31-81-62.ec2.internal:35003/messages made at: 2021-02-26T15:48:55.051
POST Request to: http://ip-172-31-81-62.ec2.internal:35001/addMessage made at: 2021-02-26T15:49:21.319
GET Request to: http://ip-172-31-81-62.ec2.internal:35002/messages made at: 2021-02-26T15:49:21.432
POST Request to: http://ip-172-31-81-62.ec2.internal:35003/addMessage made at: 2021-02-26T15:49:24.297
GET Request to: http://ip-172-31-81-62.ec2.internal:35001/messages made at: 2021-02-26T15:49:24.478
POST Request to: http://ip-172-31-81-62.ec2.internal:35002/addMessage made at: 2021-02-26T15:49:26.854
GET Request to: http://ip-172-31-81-62.ec2.internal:35003/messages made at: 2021-02-26T15:49:26.955
```

5 Conclusiones

- El programa almacena y consulta efectivamente mensajes desde la MongoDB dentro del contenedor de Docker.

- El programa distribuye la carga entre los nodos de Log Service por medio del algoritmo Round Robin, esto permite mayor cantidad de peticiones en una menor cantidad de tiempo.
- El uso de interfaces para generalizar comportamientos dentro de la aplicación permite la extensión o cambio de código sin la necesidad de alterar múltiples archivos.
- El despliegue de una aplicación web por medio de EC2 permite el uso comercial de la misma en todos los sitios que tengan conexión a Internet sin incurrir en altos costos por administración y soporte de servidores, esto se debe a que esta plataforma utiliza tecnologías en la nube.
- El desarrollo en contenedores por medio de Spring nos permite trasladar toda la arquitectura de la aplicación a cualquier dispositivo, y su ejecución se mantiene solo con la mínimia información necesaria para funcionar, realizando el mayor uso efectivo de los recursos informaticos.

6 Referencias

[1] Amazon. *Amazon EC2*. URL: https://aws.amazon.com/es/ec2/?ec2-whats-new.sort-by=item.additionalFields.postDateTime&ec2-whats-new.sort-order=desc. (entered: 26-02-2021).

[2] Amazon. *Informática en la nube con AWS*. URL: https://aws.amazon.com/es/what-is-aws/. (entered: 26-02-2021).

[3] Mongo. *Getting Started with MongoDB and Java: Part I*. URL: https://www.mongodb.com/blog/post/getting-started-with-mongodb-and-java-part-i. (entered: 21-10-2020).

[4] Mongo. *Mongo*. URL: https://www.mongodb.com/es. (entered: 26-02-2021).

[5] Postman. *Postman*. URL: https://www.postman.com/. (entered: 26-02-2021).

[6] Redhat. *¿Qué es DOCKER?* URL: https://www.redhat.com/es/topics/containers/what-is-docker. (entered: 26-02-2021).

[7] Shanika Wickramasinghe. *MongoDB Sorting*. URL: https://www.bmc.com/blogs/mongodb-sorting/. (entered: 26-11-2020).

[8] Wikipedia. *Planificación Round-robin*. URL: https://en.wikipedia.org/wiki/Round-robin_scheduling. (entered: 7-12-2020).