AREP Lab 5

Orlando Antonio Gelves Kerguelen - orlandoagk

August 2020

1 Introduction

In this project we build a simple message's repository with Java and Redis (NoSQL DB), this project is containerized with Docker and have a docker-compose.yml thats help you to run all the containers easily, we use 3 images to run the whole project; the redis original image, a image that have the LogService App that provide the interaction with the DB and the LoadBalancer that choose to what container send the petition (this have a facade to see in a browser the messages)

In this project you have the docker-compose, the Dockerfile of each Image that I build (this is in the corresponding folder). You don't need to build the Image you can pull it from my Dockerhub repository (orlandogk10)

2 Meanings

- Maven: Is a build automation tool used primarily for Java projects.

- Java: Is a programming language that is class-based, object-oriented, and designed to have as few implementation dependencies as possible

- Web Server: Is server software, or hardware dedicated to running this software, that can satisfy client requests on the World Wide Web.

- Informatic Resource: are defined as the data and information used by an organization

- AWS: Amazon Web Services is a cloud solution offered by Amazon

- Redis: Is a NoSQL DB, normally we use this DB to manage the Cache

- Docker: Is a project that we use to have a automate deploying of a APP, with this we can containerizing a APP
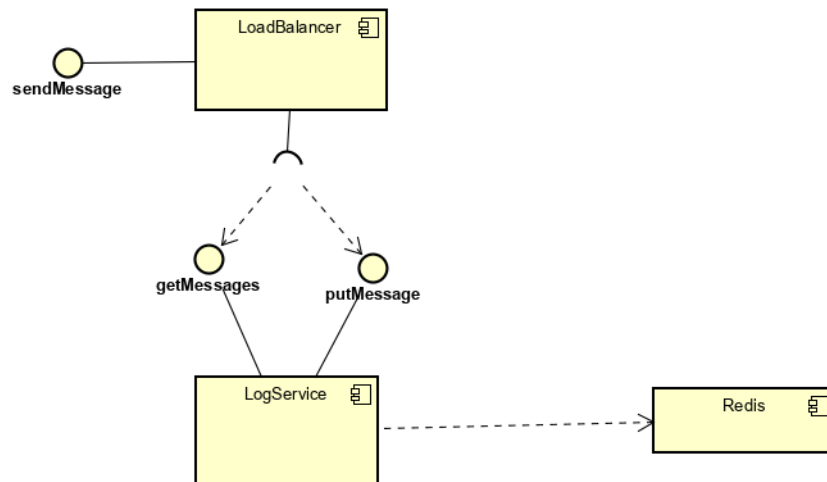
[1]

Figure 1: Component Model

3   Design

Here I present the component diagram of the architecture, and the two Class Model that we implement to give a solution to the AREP 5 Laboratory

We have 3 instances in the project, the first one is a Facade-LoadBalancer that have a Web interface that help you to publish and read messages, this instance use a Round Robin algorithm to choose the instance that needs to response the petition to call the API of the other server I use Unirest, here I have a big problem because the character underline is not supported in Unirest URL and the names of the containers use it, to solve this I use a translate of Name to IP.

The second instance is a Log Service that offer 2 API (/getMessages and /putMessage), this APP is in charged to connect the DB (Redis) with the LoadBalancer to offer the API's we use Spark

The last instance is a Redis DB with a password, we use this to persist the data of the messages

All the instances are deployed using Docker containers and all are located in a unique Virtual Machine in AWS (Linux AWS), the only port that is open in the security group is the 80 (http) that is used by the facade-loadbalancer
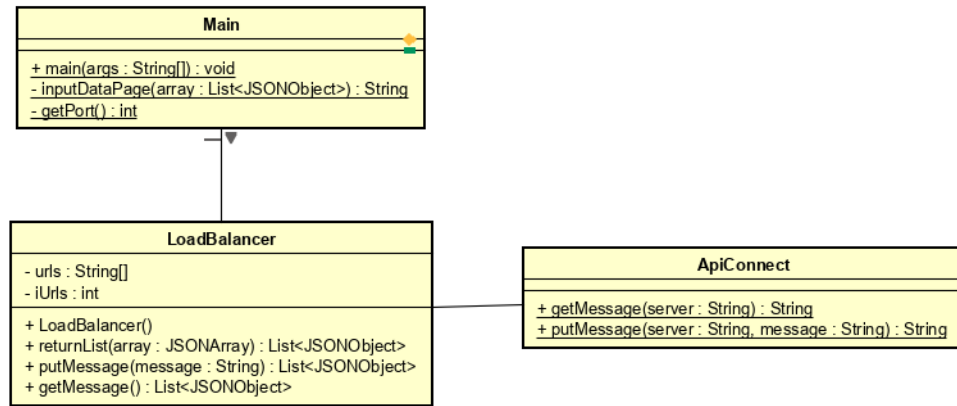
Main

+ main(args : String[]) : void
- inputDataPage(array : List<JSONObject>) : String
- getPort() : int

LoadBalancer

- urls : String[]
- iUrls : int

+ LoadBalancer()
+ returnList(array : JSONArray) : List<JSONObject>
+ putMessage(message : String) : List<JSONObject>
+ getMessage() : List<JSONObject>

ApiConnect

+ getMessage(server : String) : String
+ putMessage(server : String, message : String) : String

Figure 2: Load Balancer Class Model

Jedis

- jedis

Main

+ main(args : String[]) : void
- getPort() : int

Persistence

+ Persistence()
+ getEnvironment() : String
+ putMessage(message : String) : void
+ getMessages(nMessages : int) : List<JSONObject>
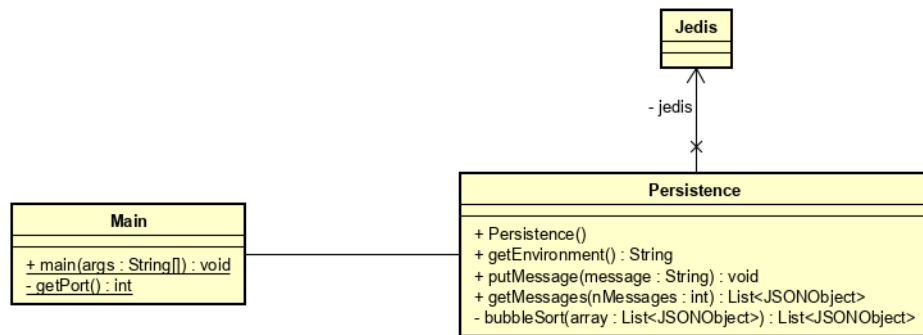- bubbleSort(array : List<JSONObject>) : List<JSONObject>

Figure 3: Log Service Class Model

3

4 Execution

4.1 Code

To run the server you need to follow the next instructions

- git clone https://github.com/orlandoagk/AREP5-AllApps.git

- cd AREP5-AllApps

- sudo docker-compose up -d –scale logservice=3

4.2 Test

- mvn test

5 Conclusion

I develop a little micro-service architecture using Docker Container and deploying that in AWS, I learn how to connect to a NoSQL BD using a Java client in this case I use Redis (NoSQL DB) and Jedis (Java Client), I probe in the experiment a auto-scaling tool like docker-compose and all is managed from AWS to have a great disponibility and offer the service to the world

References

[1] *Wikipedia.*