CSCI 5409 Advanced Topics in Cloud Computing

Assignment-2
Report

Feb. 28, 2019

Submitted By

Vamsi Gamidi

B00834696

VM709690

Section A

I have installed Apache Mesos on my AWS EC2 instance by following the following steps:

- Installed Java8 on Ubuntu 16.04 [1]
  - wget http://download.oracle.com/otn-pub/java/jdk/8u151-b12/e758a0de34e24606bca991d704f6dcbf/jdk-8u151-linux-i586.tar.gz
  - sudo tar -xvzf ~/Downloads/jdk-8u151-linux-x64.tar.gz
- Installed Mesos including zookeeper binaries [2]
  - sudo apt-key adv --keyserver keyserver.ubuntu.com --recv DF7D54CBE56151BF
  - echo "deb http://repos.mesosphere.com/ubuntu xenial main" >> /etc/apt/sources.list.d/mesosphere.list
  - lsb_release -cs
  - apt-get -y install mesos
- Started Apache Mesos master and slave [2]
  - service zookeeper start
  - service mesos-master start
  - service mesos-slave start

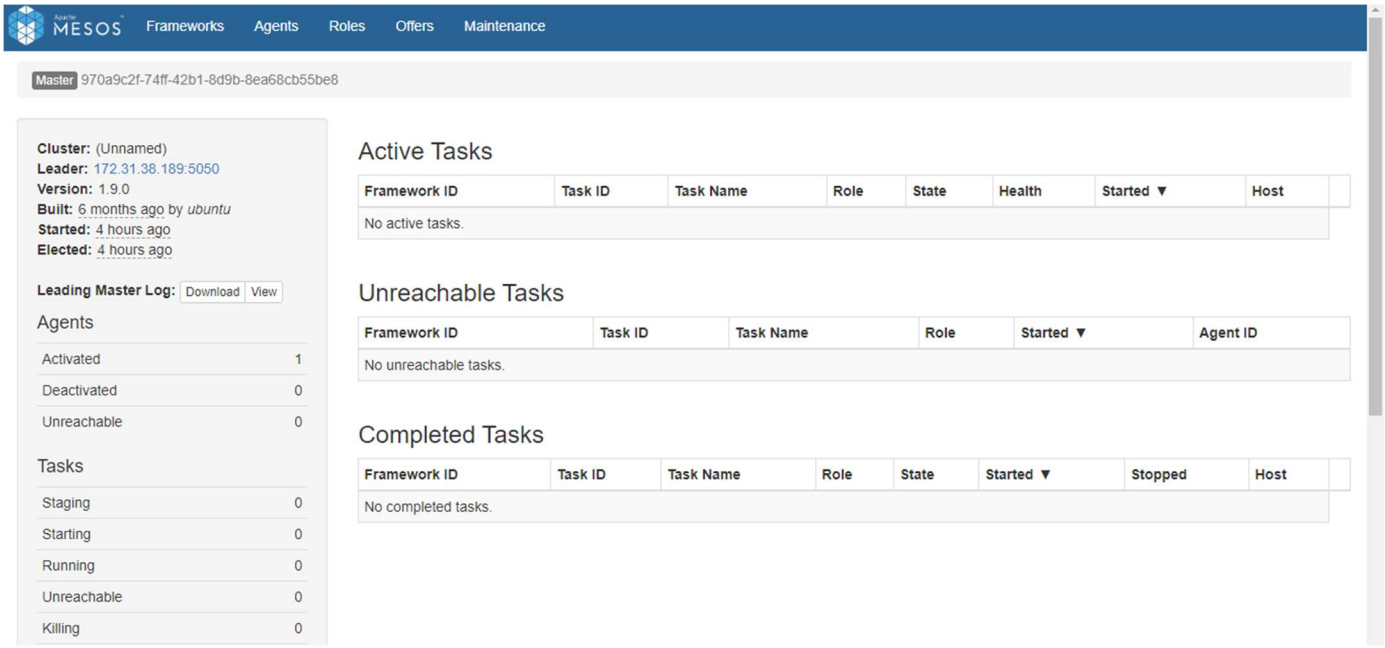


Figure 1: Apache Mesos installation

Section B

Apache Mesos is an open source, widely used cluster-wide resource manager. By using Dominant Resource Fairness (DRF), it provides fair share guarantee for data centers and clouds where resource fairness must be provided for multiple resource types such as memory, disk I/O. The main function of Mesos is to allocate resources to frameworks such as Apache Arora, Mesosphere Marathon, Chronos which in turn schedule tasks within the resources allocated by Mesos. The various factors that affect the fair share of resources are interaction with the resource offer cycles, offer holding period, task arrival rate and task duration. Resource offer cycles represent the number of seconds for which resources from an agent cannot be offered again after the resources were rejected from the same agent. Offer holding period is used to terminate the offer to a framework after a certain amount of time.

First, the Mesos agents sends the information regarding the available resources for launching tasks to Mesos Master in regular intervals. Then the master offers the available resources to the frameworks which are sorted in the order of share of resources received. Frameworks can either accept or decline the resources offered by master. Finally, frameworks allocate the received resources to multiple tasks.

The two important modules in Mesos framework are framework scheduler and framework executor. Framework scheduler takes care of picking offers from the available offers and creating a map of tasks with offers. Framework scheduler users Bin Packing and First Kit scheduling policies. Framework executor will be notified whenever an agent node accepts tasks from frameworks. DRF typically uses two-level scheduling for allocating resources to the frameworks. The master decides the amount of resources to be offered for every offer cycle and the DRF algorithm decides the first level of resource distribution. The framework chooses to accept or decline the resources offered in the second level of scheduling.

Mesos doesn't kill long-running tasks even though they are responsible for starvation of other users. Mesos pools resources from a different set of nodes to present a single view of cluster-wide resources. Master allocates the resource offers and recalculates the dominant share of users to allocate the next available offers in the cluster. Mesos master ignores the demands from the frameworks for resources by assuming the same resource demand for all the frameworks and offers resources to the framework with minimum dominant share of previously offered resources. A priority list will be prepared for every cycle based on the dominant share. The framework with the lowest dominant share will be given the highest priority.

When we compare fairness of Mesos with Marathon and Scylla frameworks, Marathon employs greedy resource consumption policy to obtain more resources even though other framework was waiting for resources which caused 38% reduction in fair resource allocation to Scylla framework. Similarly, in comparison of Aurora and Scylla, Aurora is not able to launch more tasks due to less availability of resources and faces 89% reduction in fairness of resource allocation. To conclude, DRF based allocation in Apache Mesos doesn't produce good fairness of resource allocation for frameworks such as Aurora, Scylla and Marathon.

After researching on Apache Mesos, I have the following insights, the major advantage of Apache Mesos is the ability to manage different types of workloads in different ways based on our requirement. Multi-node applications can benefit a great deal by using Apache Mesos.

Section C

I have installed marathon framework on Mesos cluster by following the below steps:

- curl -O http://downloads.mesosphere.com/marathon/v1.5.1/marathon-1.5.1.tgz [3]
- tar xzf marathon-1.5.1.tgz [3]
- mv marathon-1.5.0-* /usr/share/marathon [3]
- service zookeeper start
- service mesos-master start
- nohup ./marathon --master zk://ip-172-31-38-189.ec2.internal:2181/mesos --zk zk://ip-172-31-38-189.ec2.internal:2181/marathon >> /var/log/marathon.log 2>&1 & [3]
- jps -l [3]

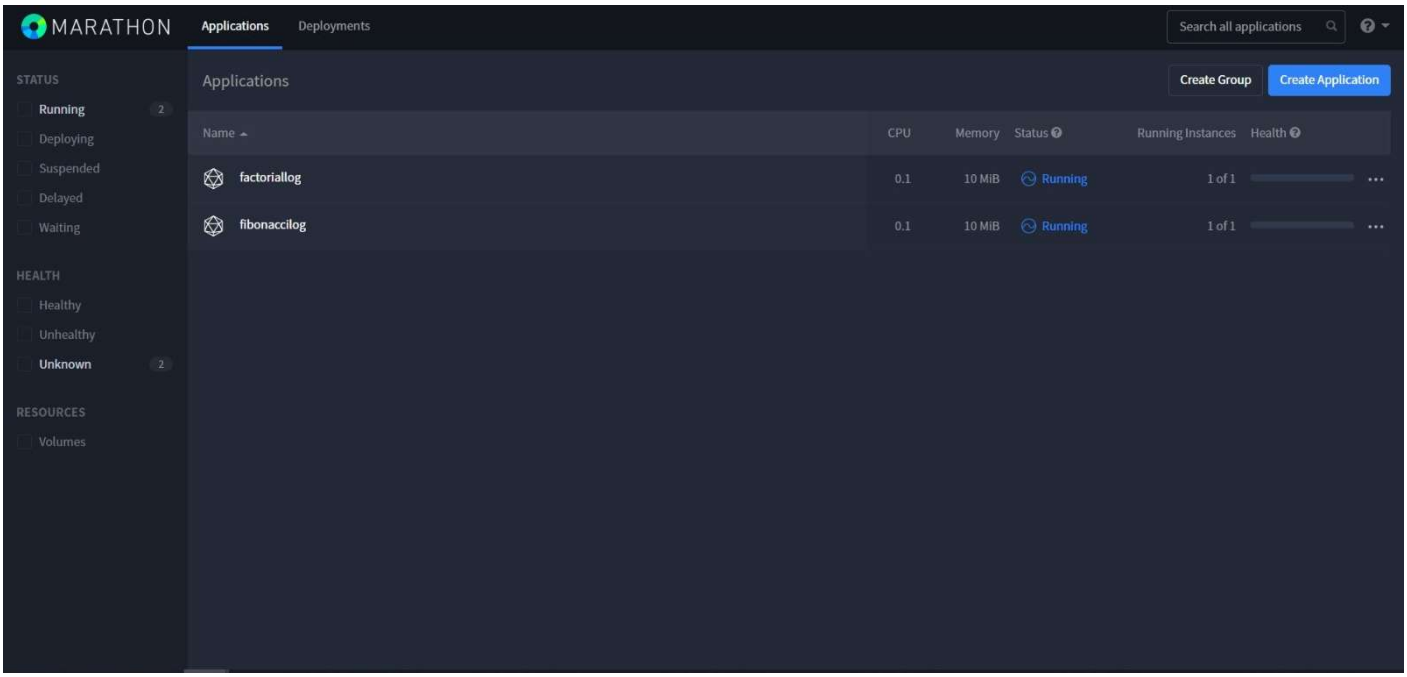


Figure 2: Marathon Installation

Section D

I have simulated the 2-developer behavior by following the below steps:

- created the repository in github
- cloned the project into 2 different folders
- added factorial into first local folder and pushed it to the repository
- pulled the changes into second folder
- added fibonacci file to second local folder and pushed it to the repository
- pulled the changes into first local folder

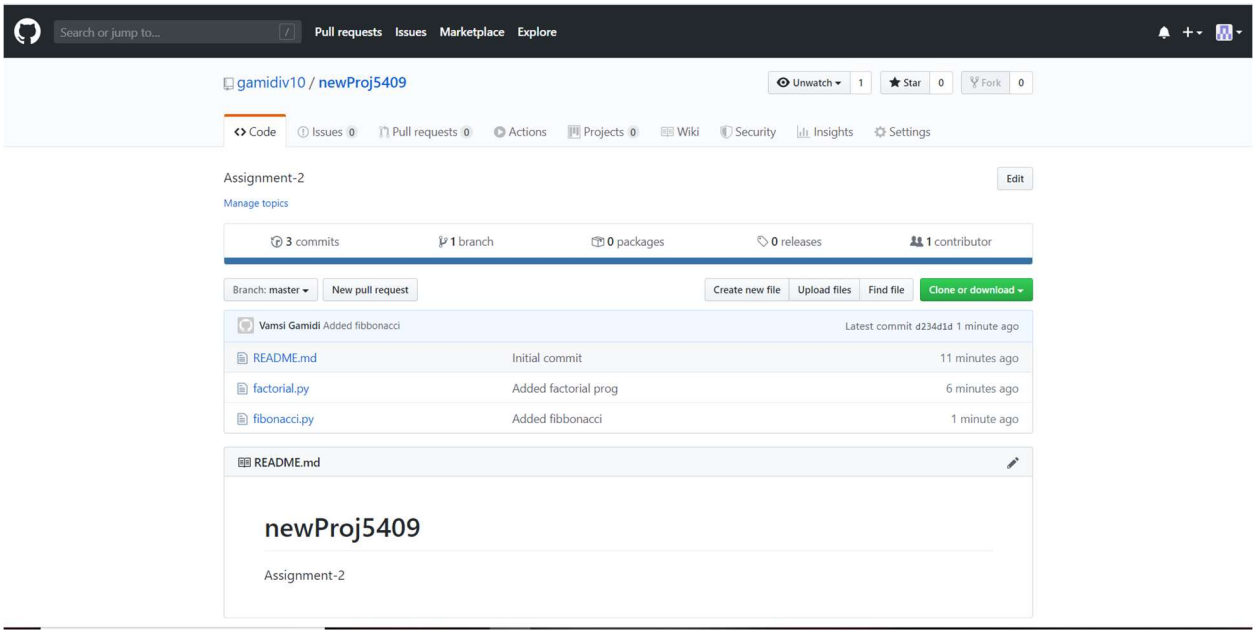
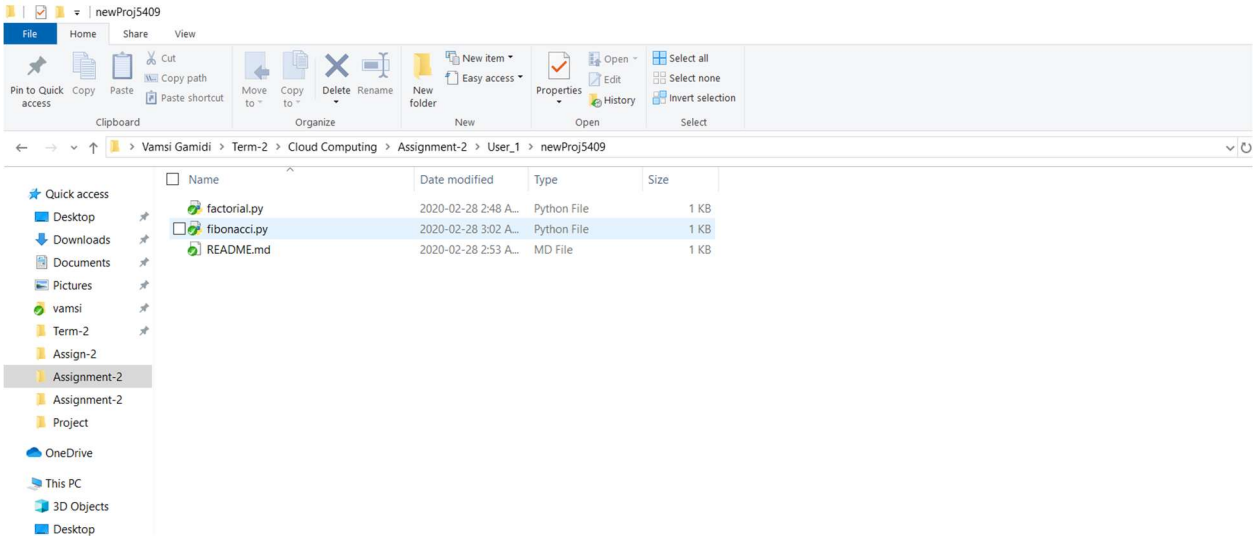


Figure 3: Git Repository

In the below figure, we can see the local folder of User 1.



In the below figure, we can see the local folder of User 2.

Figure 4: User 1 local folder

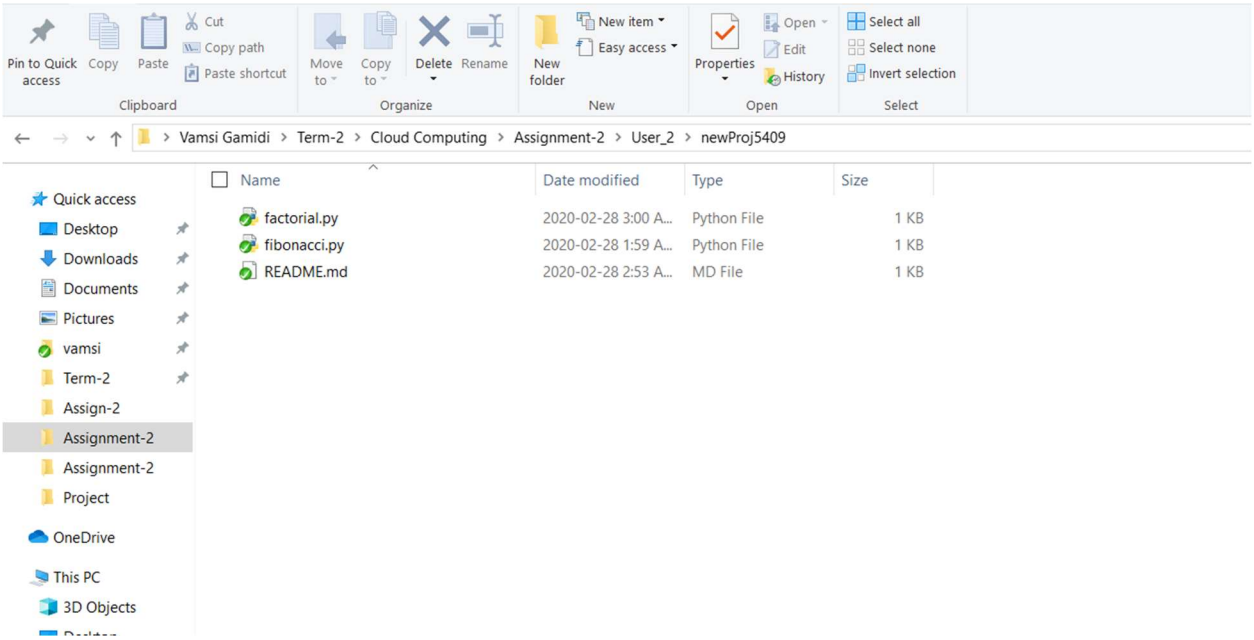


Figure 5: User 2 local folder

All the commits made to the repository can be seen in the below figure.

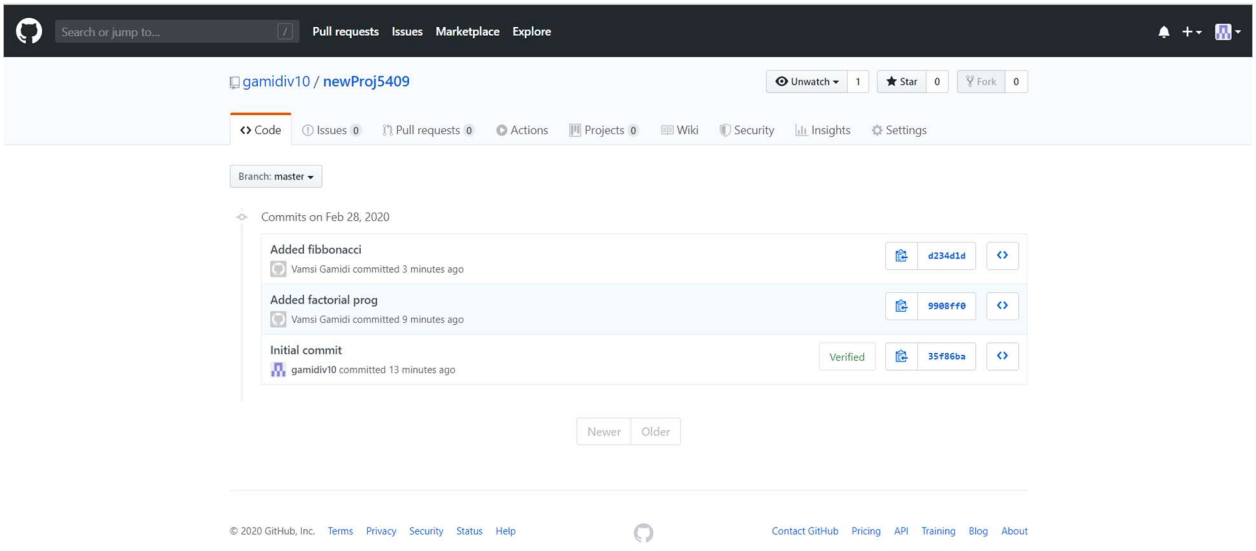


Figure 6: Commits in repository

Section E

I have launched the two tasks, factorial and fibonacci, using marathon framework.

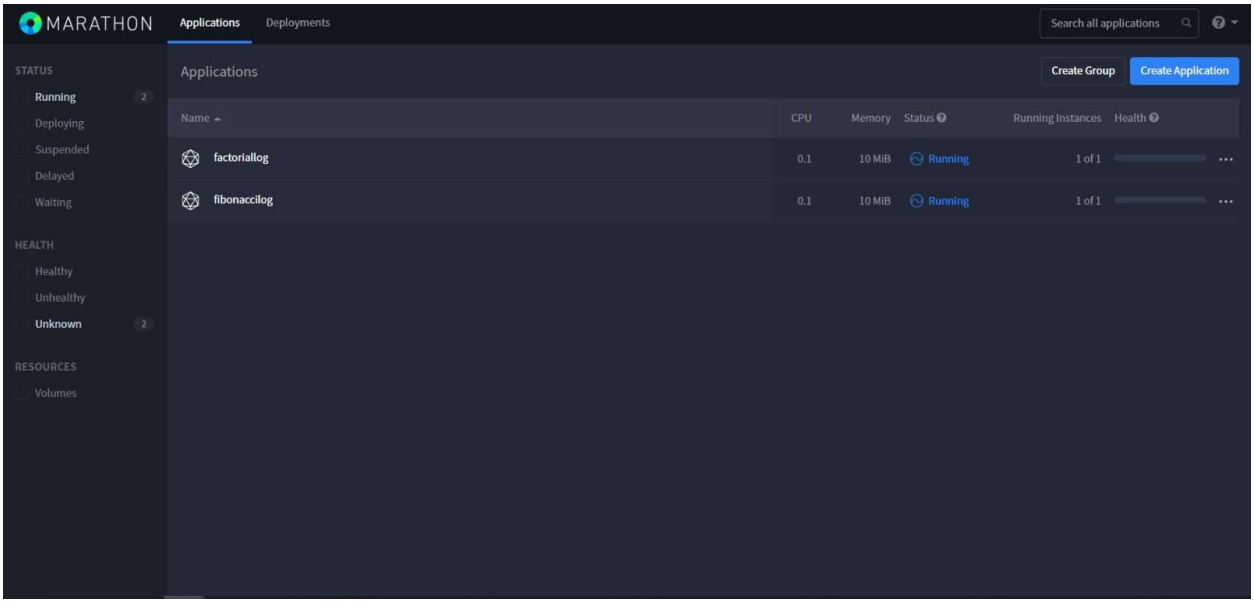


Figure 7: Marathon framework with two tasks

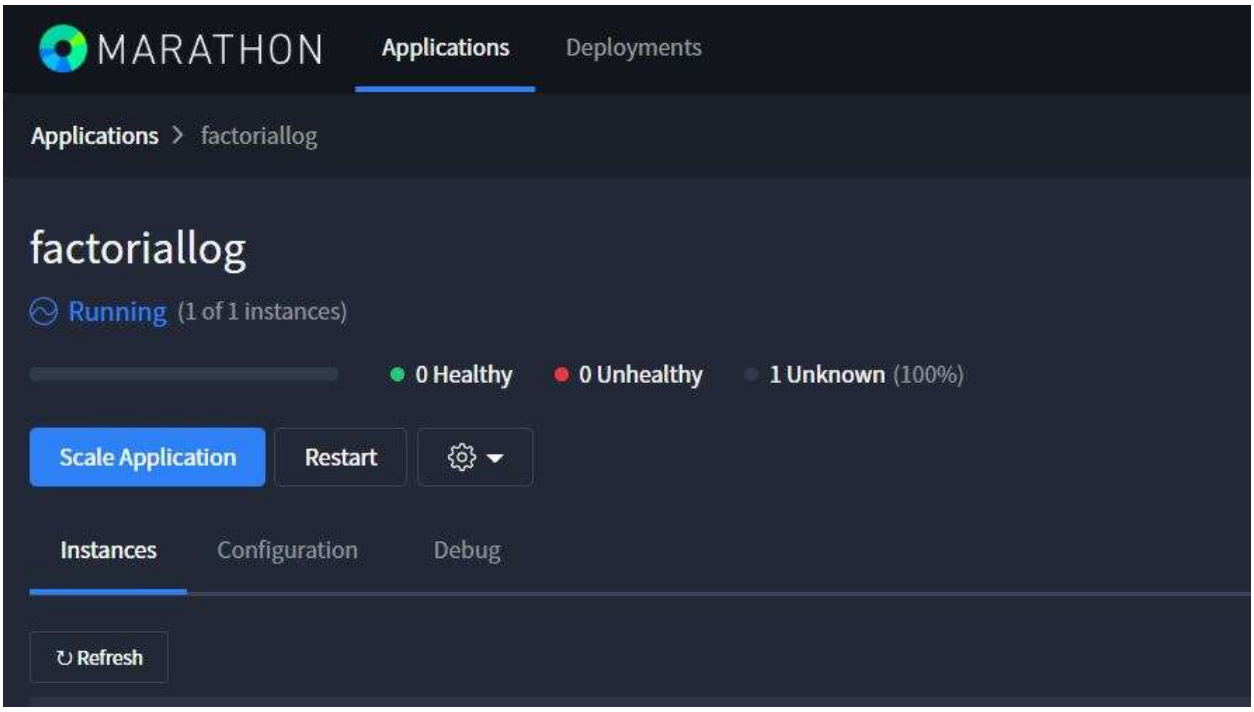


Figure 8: Factorial program task in marathon

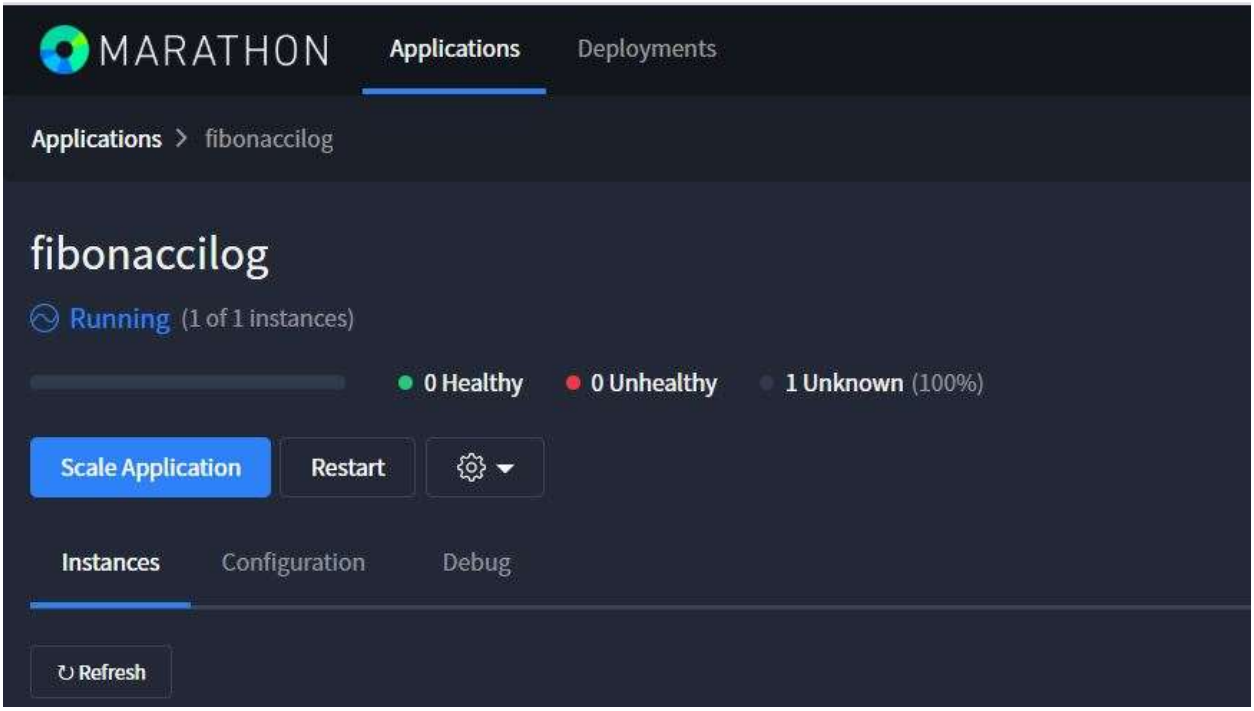


Figure 9: Fibonacci program task in marathon

References

[1] Vultr's 'How to manually install Java 8 on Ubuntu 16.04' [Online]

Available: https://www.vultr.com/docs/how-to-manually-install-java-8-on-ubuntu-16-04 [Accessed on Feb 25, 2020]

[2] Armank's 'Introduction to Apache Mesos' [Online]

Available: https://linuxacademy.com/guide/25034-introduction-to-apache-mesos/ [Accessed on Feb 26, 2020]

[3] Armank's 'Container orchestration using Mesos and Marathon' [Online]

Available: https://linuxacademy.com/guide/28039-container-orchestration-using-mesos-and-marathon/ [Accessed on Feb 26, 2020]

[4] Mesosphere's documentation on 'Marathon Application basic' [Online]

[Accessed on Feb 26, 2020]

[5] Python's documentation on 'timer.timeit' [Online]

Available: https://docs.python.org/3/library/timeit.html [Accessed on Feb 27, 2020]