REPORT P1

AggieStack - version 0.2

CSCE 678 Distributed Systems and Cloud Computing

Fall 2018

Submitted by

Ashima Sharma: 426008924

Nikhil Gupta: 526003001

Index

Project Repository

https://github.tamu.edu/nikhil-gupta/678-18-c

Contributors:

Ashima Sharma
Nikhil Gupta

Tools Used

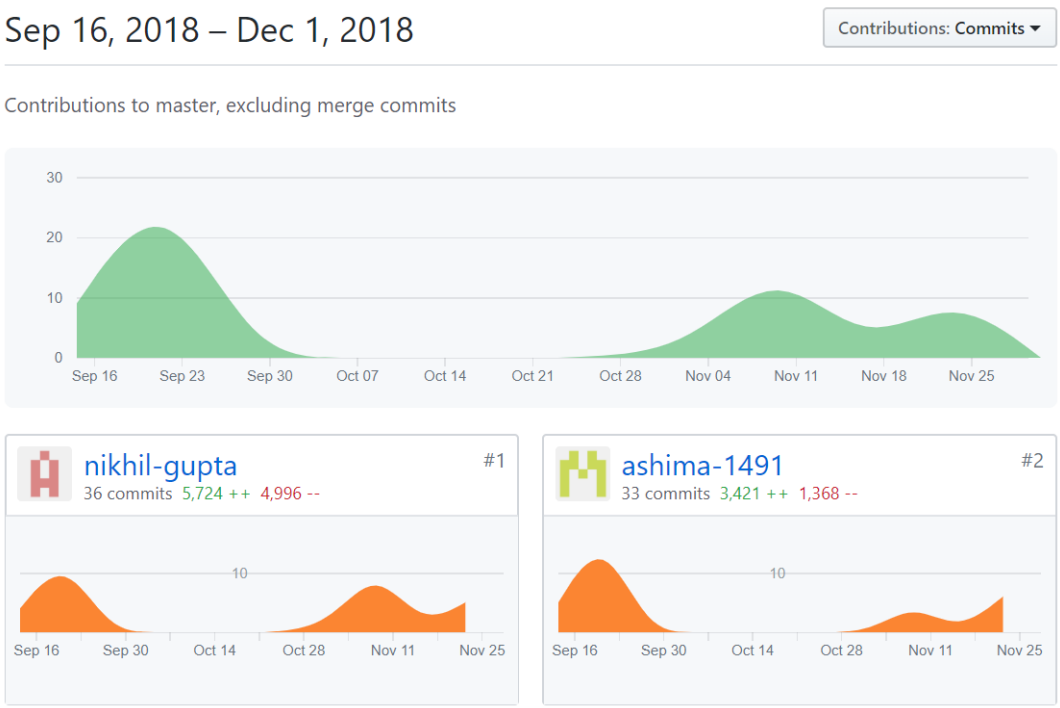
IDE: Eclipse with PyDev plugin

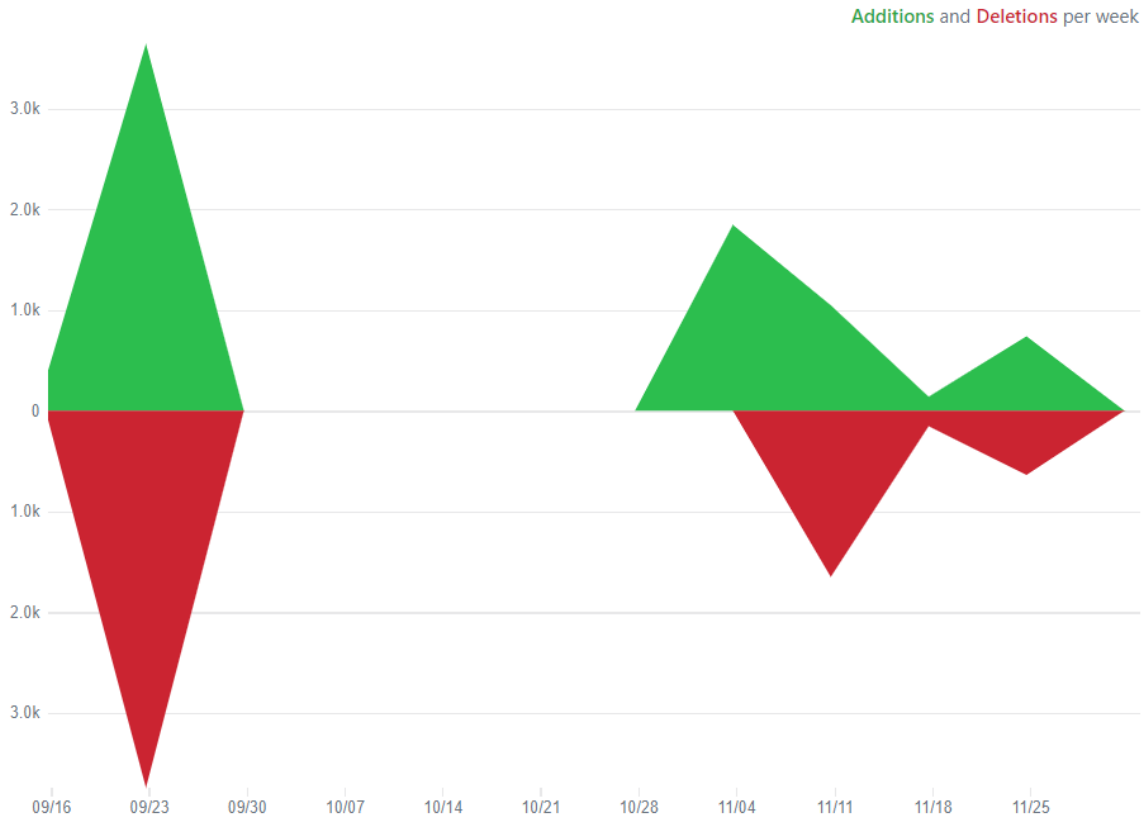
Version Control: GitBash

Language: Python

Database: MongoDB Cloud

Timeline





We made this project within a span of 8 days, putting around 4 hours each day by each person in the project.

Git Logs

commit 416484ae9a4ec1eeb3d7b9fdf08f268c0200612c (HEAD -> master, origin/master, origin/HEAD)

Merge: 979cdcf 832e59d

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 22:18:39 2018 -0600

log file deleted for merge_conflicts

commit 979cdcf31e7a35a527d17ed881aac2e09c12c823

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 22:17:37 2018 -0600

show images corrected

commit 832e59db931d813f8709ac7df286a47d25128953

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sat Dec 1 22:04:44 2018 -0600

enhanced error checking functionality

commit b916becbaee2739aaa96f96a0c0817d60c249942

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sat Dec 1 19:33:30 2018 -0600

removed useless files

commit 2339be3476890afe1aac555965afe95cb6c2659f

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 19:32:59 2018 -0600

    check if machine exist or not

commit 93bf27724ff024877b3fe4c2799fab0a9e68c880

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 16:53:12 2018 -0600

    remove command changed to rem

commit 9dce8e28f71db0bbfdbf1d6108f9d403a867318e

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 16:50:18 2018 -0600

    error message on remove if instance hosted

commit e53db61adcd780db65704f0bb2e7d7b703d57c4f

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sat Dec 1 16:37:01 2018 -0600

    remove machine modified

commit b80ca578d56cfe83a4397759f6d3c5a262010dfc

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Wed Nov 28 20:28:52 2018 -0600

    Part C command added

commit 8924241217d5ef733f9b7a8638a744d8a69c48f7

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Wed Nov 28 18:27:13 2018 -0600

    remove command

commit ead599402c051c28c590eaab49fb27ebbfd7d8b0

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 27 22:55:04 2018 -0600

    adding logger functionality version 3

commit dff8d48702d36ec8250ff51e3276b916ad01c87d

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 27 20:59:12 2018 -0600

    adding logger functionality version 2

commit d89c25814354c49428c377475ed4bfdd6098ddc1

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Mon Nov 19 16:10:11 2018 -0600

adding logger functionality version 1

commit ad53898629f8d3cacce594491df7690bf35069e3

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 13 17:57:16 2018 -0600

correcting logic in evacuate function

commit 6cda47f741e8ddc81ac2cbadf8209146979d35d2

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 13 17:30:04 2018 -0600

adding aggiestack admin add command

commit e42e6689f744d860a19f3c61815bd3d632ea291e

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 13 16:56:58 2018 -0600

adding admin remove command

commit f49a0423af2112920c870202e825de1048d1f3f0

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Tue Nov 13 15:15:59 2018 -0600

adding evacuate command

commit 8a7e404e68f84e85c65c643373e5269fd7bdb2d3

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Mon Nov 12 23:01:19 2018 -0600

    added status crud operation

commit 164358052af49509a2e89def4b874fc9f8f1514c

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Mon Nov 12 22:42:12 2018 -0600

    added status of machines

commit 4c326bda409f3cc50e452f10161f22ce746d899c

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Mon Nov 12 22:37:00 2018 -0600

    added status of machines

commit 5ca2de1a921ec30cb95f63feb8c07b78dcd4e962

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Mon Nov 12 22:28:03 2018 -0600

    rack_status healthy added

commit 057afbe471c978b350973086e3b2b3676bb52a31

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sun Nov 11 22:47:59 2018 -0600

    added show images

commit a47fb9448d2fdd80550748d5b616dcb8b72ca12e

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sun Nov 11 21:56:18 2018 -0600

    count_rack

commit df8bb24068f765e70584b6a32df8f10002133153

Merge: 75f087a 23341f7

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sun Nov 11 21:54:12 2018 -0600

    fixed

commit 75f087aced9eff99d740818ea190cbda4e1ec340

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sun Nov 11 21:52:08 2018 -0600

    show hardwre

commit 23341f7a945efc97f2001321524808fc1819ac7f

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sun Nov 11 21:51:04 2018 -0600

    added count functionality

commit 7f767329c6d6767009349e10c5ecfd2953cec455

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sun Nov 11 21:42:03 2018 -0600

    added config-images functionality

commit 50cdc044ae8d226c3ab5fdf5e3d2ef1b07ab5851

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Sun Nov 11 21:14:42 2018 -0600

    deleted log file

commit 3ebfa0830de87d74ec3b8378019637f51086ebec

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sun Nov 11 21:34:04 2018 -0600

    admin config hardware modified

commit a8f290d4c3c95d820e646dce479b0da6e8ca0602

Author: ashima1491 <sharma.ashima91@gmail.com>

Date: Sun Nov 11 21:13:24 2018 -0600

PartB harware_rack

commit 24ab862ee67598d13ecaea48e89c1ead000462c8

Author: nikhil-gupta <nikhil.gupta@tamu.edu>

Date: Fri Nov 9 18:18:14 2018 -0600

Add files via upload

commit c47d20d8b0f0b1c04aef17c5d7dc88ae88223185

Author: Nikhil Gupta <niksgupta36@gmail.com>

Date: Fri Nov 9 17:52:23 2018 -0600

added crud operations

Architecture

Design and Structure

As mentioned in the P0 report we designed the CLI app and found a standard approach of parsing doc string using “Docopt” and creating commands. We have used MongoDB cloud as a database to store the data from config files. We selected a database over the file system as a storage medium, to keep our application flexible enough so that we can add more functions in the future, hence more scalable. Also, having a database makes the CRUD operations easier.

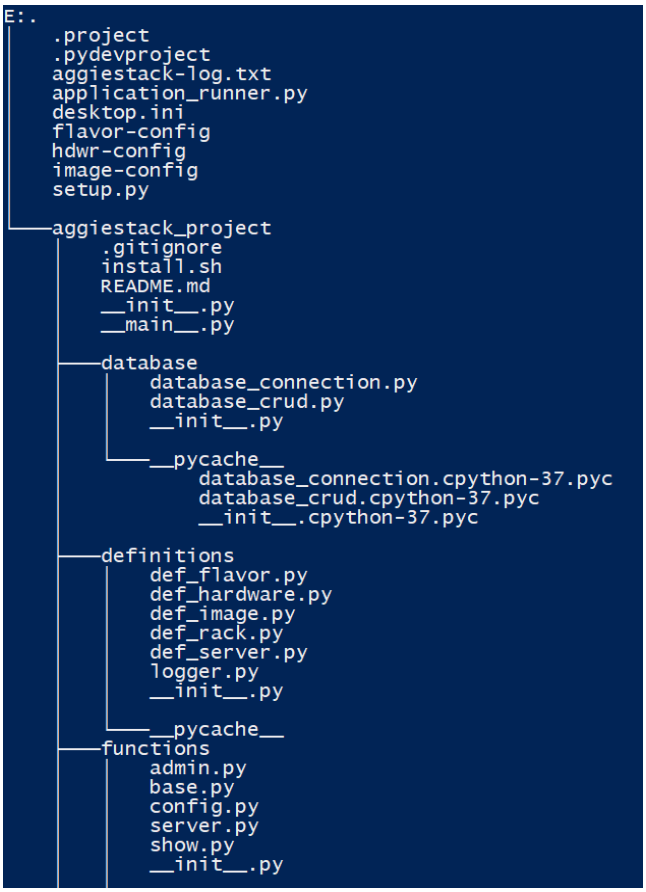
- **The data files**: hdwr-config, flavor-config, image-config have been kept the root of the project so that when the user gives the config command, the file can be read. **Please note these files have type ‘file’ and not ‘text’.**
- **Handling Duplicity**: If a config file is loaded multiple times, the old duplicate entries will be deleted from the database, and new entries will be inserted from the file.
- **Handling Dynamic configuration for the machine**: The RAM, num of disks and number of virtual CPU is stored into two sets of columns: “original” and “current”. The values for “original”

will remain intact and the values for “current” will be updated in future. Consequently, admin can see the “current” and “original” values. But a regular user can only see the original i.e. default configuration.

- **Rack_collection** has been added to store all racks; **instance_collection** is added to store all instances created on physical machines
- Racks and machines have a **status** to determine whether a rack/machine is **‘sick’ or ‘healthy’** which is used as an underlying concept in evacuation of racks and allocation of instances
- Whenever an instance is allocated on a machine, the machine’s **configuration** (RAM, CPU, Disk) is updated
- In **Placement Algorithm**, we allocate the instance on the first ‘healthy’ machine on a ‘healthy’ rack in order of iteration which meets the *can_host* criteria from P0

File system architecture

Below is the tree structure of the files inside our Project named 678-18-c. Our root folder 678-18-c contains P1 folder.



Database Collections

test.flavor_collection

COLLECTION SIZE 364B TOTAL DOCUMENTS 4 INDEXES TOTAL SIZE 16KB

Find Indexes

FILTER {"filter":"example"}

QUERY RESULTS 1-4 OF 4


```
_id: ObjectId("5bfdd5af67071c2ed0bd3494")
flavor_name: "small"
RAM: "1"
num_Disks: "1"
num_Vcpus: "1"
```


test.image_collection

COLLECTION SIZE 325B TOTAL DOCUMENTS 3 INDEXES TOTAL SIZE 16KB

Find Indexes

FILTER {"filter":"example"}

QUERY RESULTS 1-3 OF 3


```
_id: ObjectId("5bfdd5a667071c36b0148c8b")
image_name: "linux-ubuntu"
image_size: "128"
path: "/images/linux-ubuntu-v1.0.img"
```

test.instance_collection

COLLECTION SIZE 1.71KB TOTAL DOCUMENTS 15 INDEXES TOTAL SIZE 36KB

Find Indexes

FILTER {"filter":"example"}

QUERY RESULTS 1-15 OF 15


```
_id: ObjectId("5bff3c3767071c075ca993e4")
machine_name: "k1"
flavor_name: "small"
image_name: "linux"
instance_name: "niks"
```


test.machine_collection

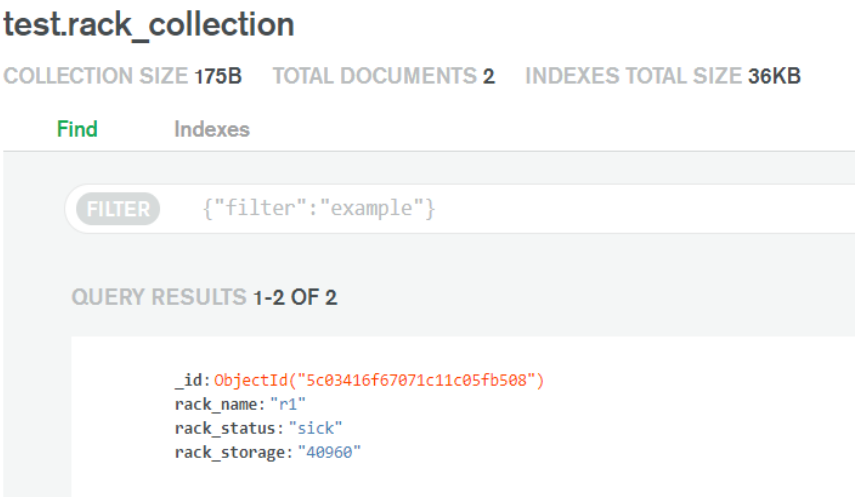
COLLECTION SIZE 2.44KB TOTAL DOCUMENTS 10 INDEXES TOTAL SIZE 36KB

Find Indexes

FILTER {"filter":"example"}

QUERY RESULTS 1-10 OF 10


```
_id: ObjectId("5c03416c67071c11c05fb4fd")
hardware_name: "m1"
Current RAM: 15
Current num_Disks: 7
Current num_Vcpus: 3
machine_status: "sick"
rack_name: "r1"
ip: "128.0.0.1"
Original RAM: "16"
Original num_Disks: "8"
Original num_Vcpus: "4"
```



Logging

As per the requirements of the project, we have a log file aggiestack-log.txt inside folder P1/aggiestack, that includes all the activity performed in the CLI.

The log will have the following properties:

- Stores any command typed on CLI.
- Since we have put try-except functions in our code, the log will store most of the exceptions or error messages.
- The log will catch errors such as invalid filename, non-existing configuration files, and invalid format in the files
- The log file will have the following format:
    a. Command entered by the user
    b. The output generated by the command entered in CLI
    c. Status - SUCCESS/FAILURE

Log File Screenshots

Status: SUCCESS →

```
##########################################################################

Command: aggiestack  admin evacuate r2

Updated rack data inserted into database
Updated hardware status inserted into database
Status : SUCCESS
```

Status: SUCCESS →

```
##########################################################################

Command: aggiestack  admin rem m1

Machine cannot be removed as an instance is already hosted on it
Status : SUCCESS
```

Status: FAILURE →

```
##########################################################################

Command: aggiestack  config --hardware hdwr-conf

File not found (Invalid path)
Status : FAILURE
```

Status: FAILURE →

```
######################################################################

Command: aggiestack  server create --image linux --flavors small ash

Image does not match with the image in storage server! Please try again with the correct image name

Status : FAILED
```

Commands

PART A

1. aggiestack server create --image IMAGE --flavor FLAVOR_NAME INSTANCE_NAME

An instance is created successfully, if image name and flavor name match with the ones in storage servers.

Step 1. We check the first machine with '**healthy'** status in *machine_collection*, if it can host the instance, then it is created on that machine, else we iterate through the next 'healthy' machines.

Step 2. We check the flavor RAM, nDisks, and nVcpus and if can_host is True, then we add it to *instance_collection* and subtract the RAM, nDisks, and nVcpus from the hosting machine's RAM, nDisks and nVcpus.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack server create --image linux-ubuntu --flavors small instance1
Updated data inserted into database
Data inserted into database
Server created!
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

2. aggiestack server delete INSTANCE_NAME

The given instance will be deleted if it exists in *instance_collection*. We also updated the machine RAM, nVcpus, and nDisks in *machine_collection* after deleting the instance hosted on it.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack server delete instance1
Updated data inserted into database
Instance deleted : instance1
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

3. aggiestack server list

Lists all instances running (name, image, flavor).

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack server list
['instance_name: instance1', 'flavor_name: small', 'image_name: linux-ubuntu']
['instance_name: instance2', 'flavor_name: small', 'image_name: linux-ubuntu']
['instance_name: instance3', 'flavor_name: large', 'image_name: linux-ubuntu']
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

4. aggiestack admin show hardware

For each physical machine in the cloud, it lists how much memory, nDisks, and nVcpus are currently free. It also shows the resources available.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack admin show hardware
**** Current Hardware available on the server ****
hardware_name  :  m1
Current RAM  :  14
Current num_Disks  :  6
Current num_Vcpus  :  2
ip  :  128.0.0.1
Original RAM  :  16
Original num_Disks  :  8
Original num_Vcpus  :  4


hardware_name  :  m2
Current RAM  :  0
Current num_Disks  :  30
Current num_Vcpus  :  0
ip  :  128.0.0.2
Original RAM  :  16
Original num_Disks  :  32
Original num_Vcpus  :  4


hardware_name  :  m3
Current RAM  :  16
Current num_Disks  :  16
Current num_Vcpus  :  4
ip  :  128.0.0.3
Original RAM  :  16
Original num_Disks  :  16
Original num_Vcpus  :  4


hardware_name  :  m4
Current RAM  :  16
Current num_Disks  :  8
Current num_Vcpus  :  4
ip  :  128.0.0.4
Original RAM  :  16
Original num_Disks  :  8
```

5. aggiestack admin show instances

For each instance currently running, it shows which physical server the instance is running.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack admin show instances
['machine_name: m1', 'instance_name: instance1']
['machine_name: m1', 'instance_name: instance2']
['machine_name: m2', 'instance_name: instance3']
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

PART B

1. aggiestack admin evacuate RACK_NAME

This command marks the rack as 'sick' in rack_collection and consequently marks all the machines hosted on that rack as 'sick' as well. Now, all the instances hosted on 'sick' machines will be transferred to 'healthy machines' as per the algorithm implemented previously.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack admin evacuate r1
Updated rack data inserted into database
Updated hardware status inserted into database
Updated hardware status inserted into database
Updated hardware status inserted into database
Updated hardware status inserted into database
Updated hardware status inserted into database
Updated hardware status inserted into database
Updated data inserted into database
Updated data inserted into database
Updated data inserted into database
Updated data inserted into database
Updated data inserted into database
Updated data inserted into database
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

2. aggiestack admin rem MACHINE

This command removes/deletes the machine from machine_collection from the DB. If it hosts instance, then the CLI command will output an error.

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack admin rem m1
Machine deleted : m1
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

3. aggiestack admin add -mem MEM -disk NUM_DISKS -vcpus VCPUs -ip IP -rack RACK_NAME MACHINE

This command adds a machine to the rack (existing racks, if not present creates a rack in rack_collection).

```
PS E:\Eclipse Projects\678-18-c\P1\aggiestack> aggiestack admin add -mem 512 -disk 4 -vcpus 8 -ip 128.0.0.2 -rack r3 m9
Machine data inserted into database
PS E:\Eclipse Projects\678-18-c\P1\aggiestack>
```

PART C

Data Structure Changes

As the racks have a storage server where images can be cached, and the same image can be cached on different racks, there can be two ways of storing this in the database.

- The *rack_collection* can have a list of images OR
- The i*mage_collection* can have a list of racks

As in our API, we have methods like aggiestack show images and aggiestack admin show imagecaches RACK_NAME, the first data structure would be more suitable.

Initially, the *rack_collection* had: {rack_name, rack_status, rack_storage}

```
_id: ObjectId("5bff3c2c67071c441808cffa")
rack_name: "r1"
rack_status: "healthy"
rack_storage: "40960"
```

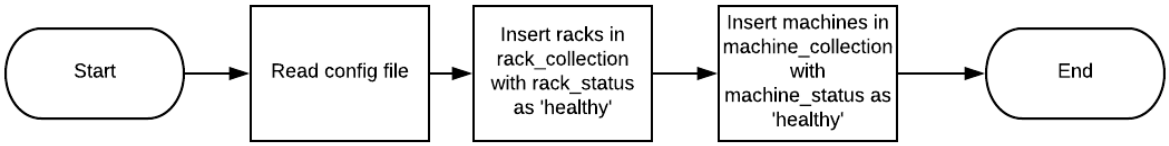
To implement Part C, *rack_collection* will have: {rack_name, rack_status, rack_storage, rack_storage_available, images}

The *rack_collection* will look like this:

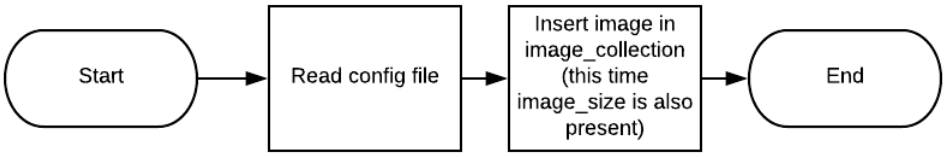
```
{"_id":
{"$oid":"5bff3c2c67071c441808cffa"},
"rack_name":"r1",
"rack_status":"healthy",
"rack_storage":"40960",
"rack_storage_available": "40000";
"images": ["image_1", "image_2"]
}
```

APIs that are impacted:

- **aggiestack config --hardware hdwr-config.txt (Implemented)**

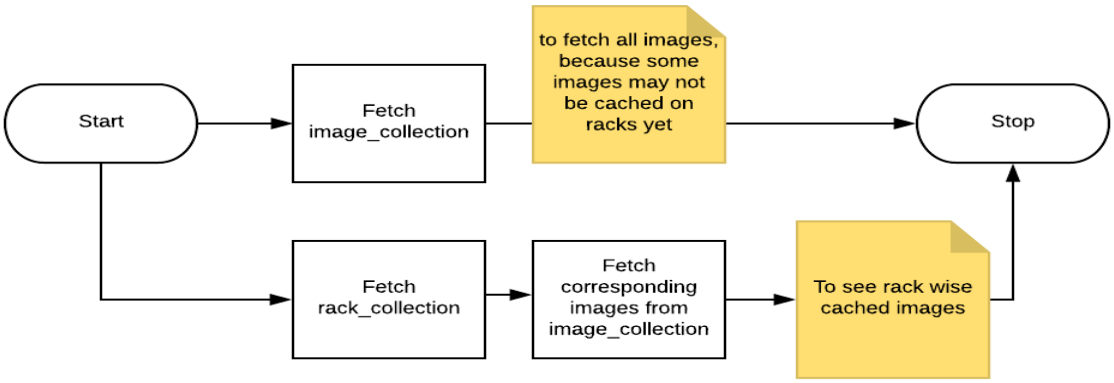


- **aggiestack config --images image-config.txt (Implemented)**

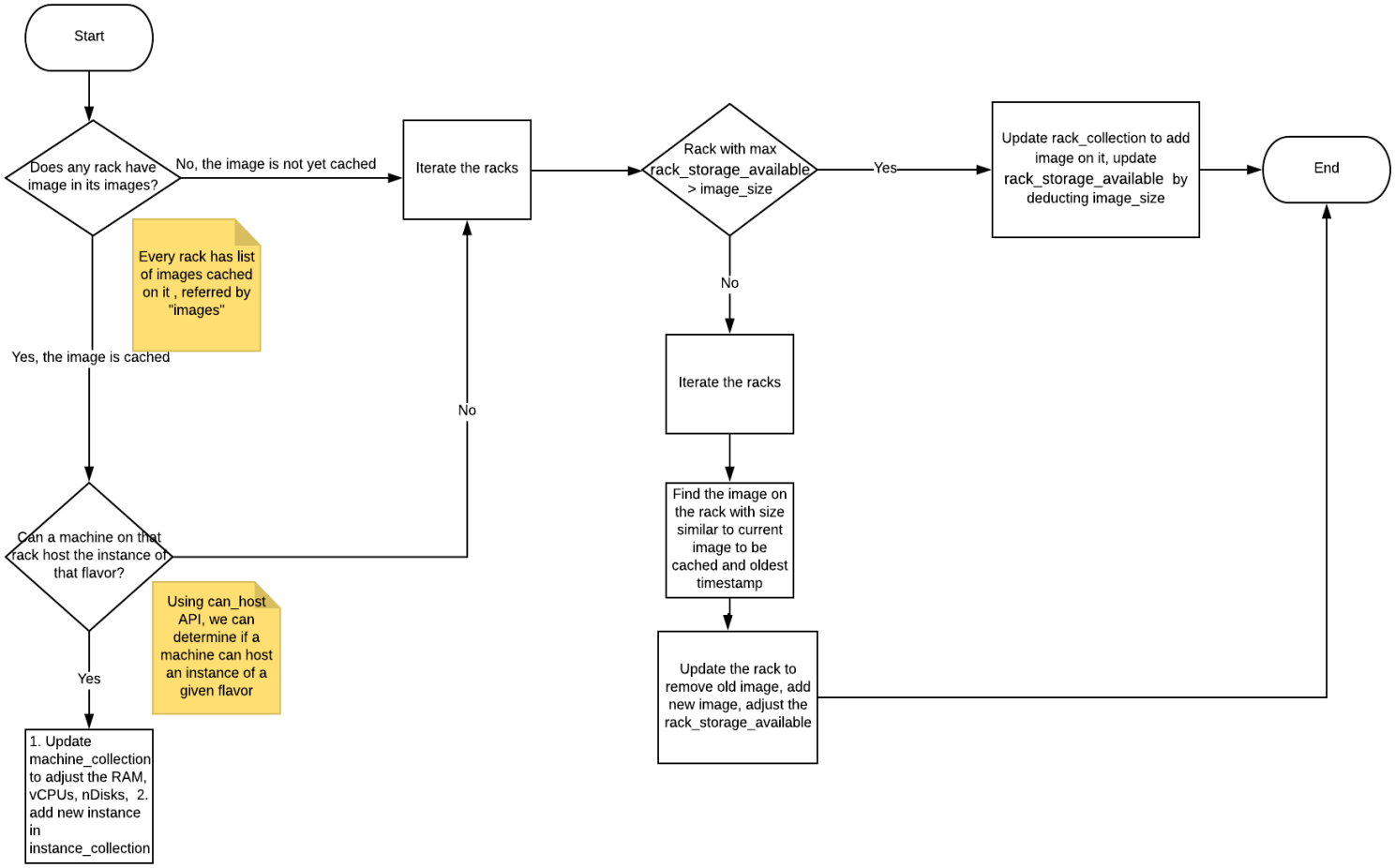


- **aggiestack show hardware (Implemented)**

- **aggiestack show images**



- **aggiestack server create --image IMAGE --flavor FLAVOR_NAME INSTANCE_NAME**



- **aggiestack admin show imagecaches RACK_NAME**

- **aggiestack config --flavors flavor-config.txt** (Same as before, Implemented)

- **aggiestack server delete INSTANCE_NAME** (Same as before, Implemented)
- **aggiestack server list** (Same as before, Implemented)
- **aggiestack admin show hardware** (Same as before, Implemented)
- **aggiestack admin show instances** (Same as before, Implemented)

ESTIMATION:

A) Time required for Implementation

- In Part C, 6 out of 11 API commands need to be upgraded over previous version. Assuming a single command requires on an average 4 hours to implement, approximate time needed for 6 commands would be 6*4= 24 hours.

B) Time required for Testing

- In Part C, all 11 API commands need to be tested to verify regression, code breakage, integration. Assuming a single command requires on an average 2 hours to test, approximate time needed for 11 commands would be 11*2= 22 hours.

HOW TO RUN THE PROJECT

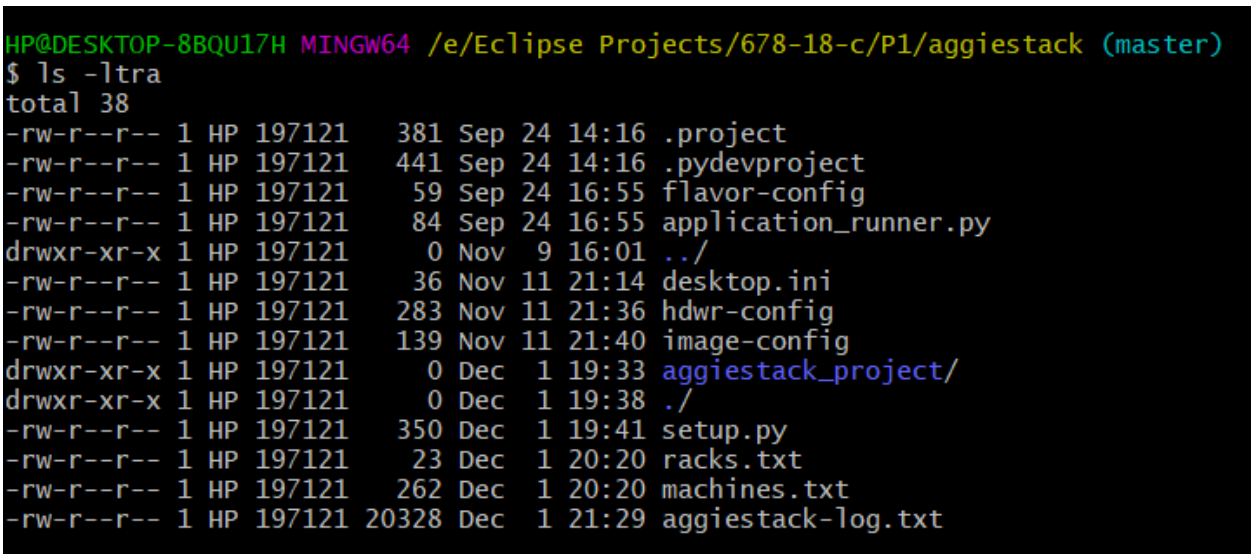
Note: Our project works fine on Windows machine. We did not test this on a Mac machine due to non-availability of the same.

A) Running via pip install command

- Download and unzip the application
- Navigate to path: \678-18-c\P1\aggiestack
- Run the command: **pip install .**
- The above command will install our project
- You might have to install some python modules such as dnspython, pymongo etc.

- Now run any of the following commands:
    i. aggiestack config --hardware <filename>
    ii. aggiestack config --images <filename>
    iii. aggiestack config --flavors <filename>
    iv. aggiestack admin show hardware
    v. aggiestack admin rem <machinename>
    vi. aggiestack admin can_host <machinename> <flavor>
    vii. aggiestack show hardware
    viii. aggiestack show images
    ix. aggiestack show flavors
    x. aggiestack show all
    xi. aggiestack server create --image <imagename> --flavors <flavor> <instancename>
    xii. aggiestack server delete <instancename>
    xiii. aggiestack server list
    xiv. aggiestack admin show instances
    xv. aggiestack admin evacuate <rackname>
    xvi. aggiestack admin add -mem <RAM> -disk <NUM_DISKS> -vcpus <VCPUs> -ip <IP> -rack <rackname> <machinename>

- For the commands that require <filename>, the files have been put in the path- \678-18-c\P1\aggiestack\ as shown in the screenshot given below:



- You can see the logs in the log aggiestack-log.txt stored at location path: \678-18-c\P1\aggiestack

Important Note: The log file will be created where we run the command. So, it's advisable to run the command in the path \678-18-c\P1\aggiestack so it appends all CLI activities in the same log file.

B) Running locally

In case you do not want to install our CLI, you can also locally run the application and test the commands

- Download and unzip the application
- Open command prompt or git bash
- Navigate to path: \678-18-c\P1\aggiestack
- Run the command **python application_runner.py <aggiestack cli command without keyword aggiestack>**

For example:

```
python application_runner.py config --hardware hdwr-config

python application_runner.py show hardware

python application_runner.py admin show hardware
```