Capstone Project
Name: Juan Li
CnetID: juanmiracle

website: https://juanmiracle.ucmpcs.org

1. instance tag

a. Instance(s) running the web application: `juanmiracle-capstone-webserver`

b. Instance(s) running the annotator: `juanmiracle-capstone-annotator`

c. An instance running the utility scripts: `juanmiracle-Utilities`

d. `ELB: juanmiracle-elb`

2. Full source code in a .zip file. Include:

The full code is divided into three parts (folder) with the major file:

(1) gas_web_server: `mpcs_app.py, view`

(2) gas_annotator: `jobs_runner.py, run.py, anno_config.py`

(3) UTILITIES:

```
results_notify.py: send notification to email when job is completed
glacier_updown.py: archive and restore job in glacier
```

3. This is the flow we will use to test your GAS:

1. Register a new user. **`mpcs_app.py`**
2. Get email confirmation notice; click to confirm new user registration. **`mpcs_app.py`**
3. Login using the new username. **`mpcs_app.py`**
4. Run one or more annotations. **`jobs_runner.py`**
5. Receive notification email when annotation(s) is(are) complete. **`results_notify.py`**
6. View a list of annotation jobs. **`mpcs_app.py`**
7. Click on a job in the list and view the details. **`mpcs_app.py`**
8. Click on a link to download the input/results files, and view the log file (this should work for all Free users right after results are received). **`mpcs_app.py`**
9. Revisit the list of annotation jobs, after allowing at least two hours to pass. Click on a link to download the input/results files (this should not work; I should be asked to upgrade, ideally just by displaying an upgrade link instead of a file link); I should still be able to view the log file.
10. View the user's profile; click the link to upgrade to a Premium user.
11. Provide the credit card info and upgrade to a Premium user.
12. View the jobs list again.
13. Click on a link to download the input/results files, and view the log file (this should all work now since it's a Premium user).

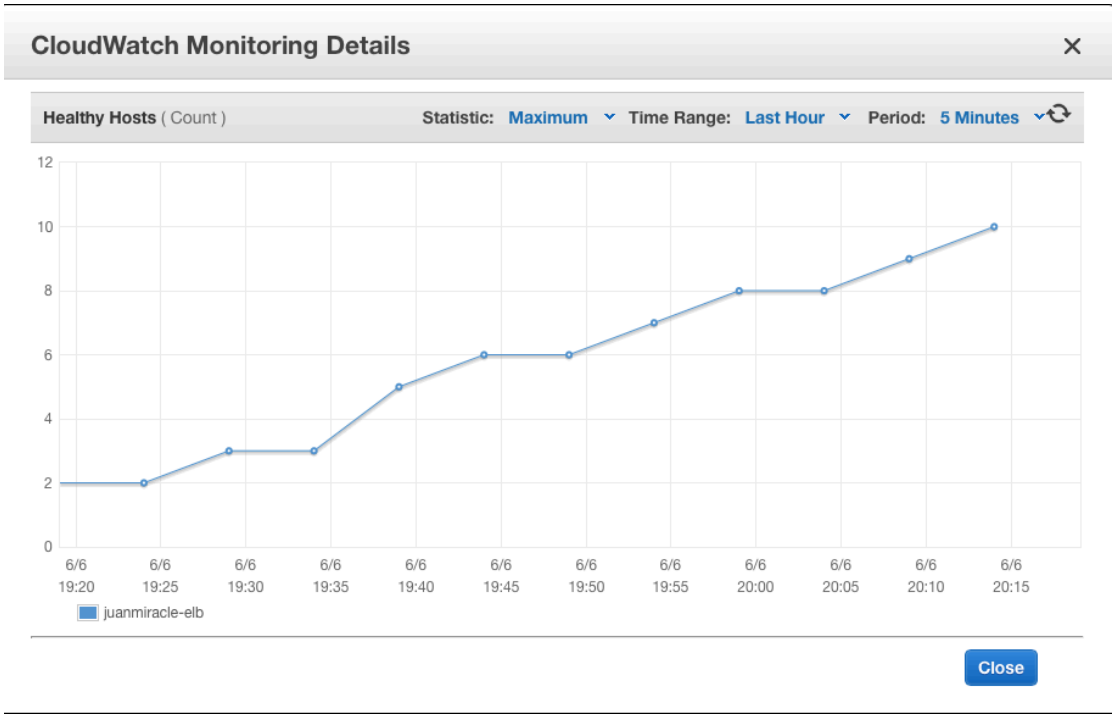
credit card: 378282246310005
eg: name on card: test
CVC: 123
month: 08
year: 2018

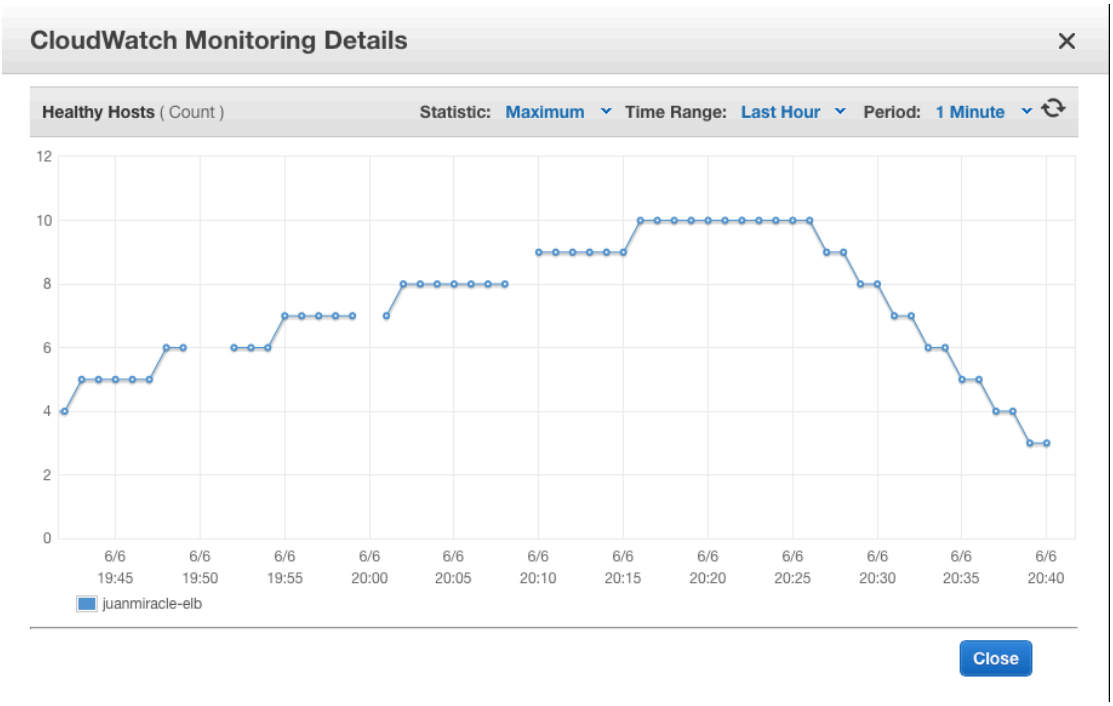
We will check the following in the AWS environment:

- Uniquely identified Input, output, log files are stored in the correct S3 buckets.
- Correctly updated items are stored in the database for each job.
- Archived Free user files are stored in the gas-archive bucket (storage class set to Glacier).
- You have correctly configured two notification topics (one for job requests, one for job completions) and subscribed the corresponding message queues to their topic.
- You have a running load balancer with two instances in service.
- You have two Auto Scaling groups, each with 2 instances: one for the web app and another for the annotator.
- All your instances are properly tagged by your auto scalers.
- You have separate instance(s) running the results notifier and the file archiver scripts.

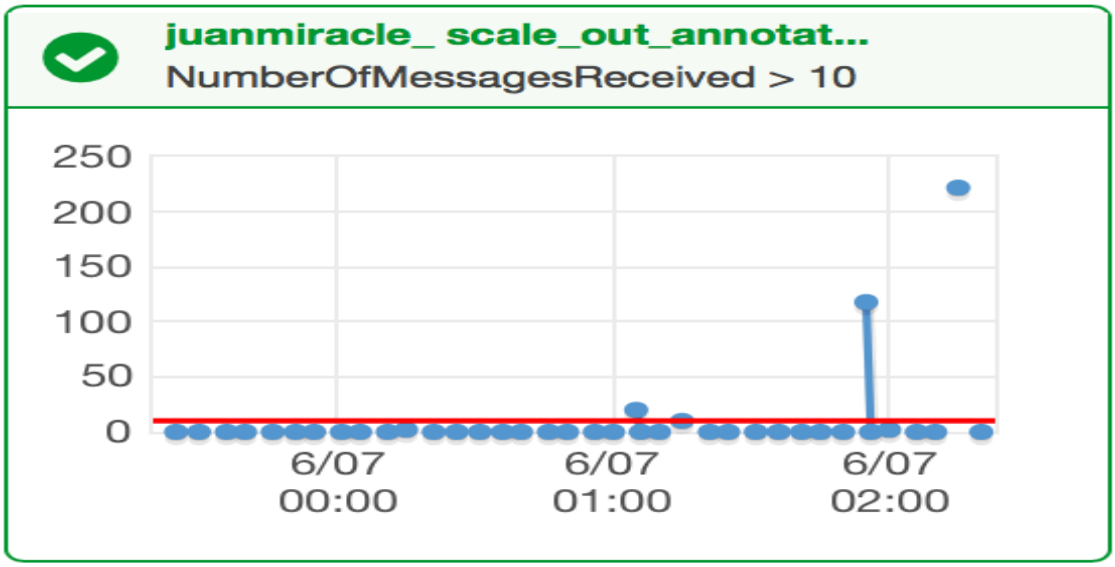
4. Your screenshots from the “bees” exercise.

```
beeTest.py : attack bee
deleteSQS: delete the attack sqs
```





We will use Da Bees to test the scaling policies of your web auto scaling group. We expect that it will scale up and down under load without exceeding the maximum of 10 instances or going below the minimum of 2 instances. We will use manual loading to test the annotator auto scaling group if you have not provided a script as part of exercise #14.



Name	Launch Configuration	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health Check Grac
juanmiracle--auto-scaler-annotator	juanmiracle-launch-con...	3	3	2	10	us-east-1c	300	300
juanmiracle-auto-scaler	juanmiracle-launch-config	2	2	2	10	us-east-1c	300	300