## Enterprise Sales Engineer - Denmark

**Hiring Exercise**

**Prerequisites - Setup The Environment**

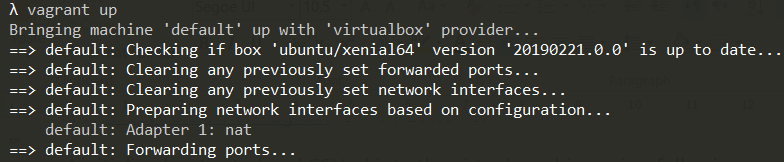
**Vagrant**

Downloaded and installed [VirtualBox 6.0](https://www.virtualbox.org/wiki/Downloads) on Windows 10 for OS X hosts. I then downloaded and installed the [latest version of Vagrant](https://www.vagrantup.com/downloads.html) for Windows 10.

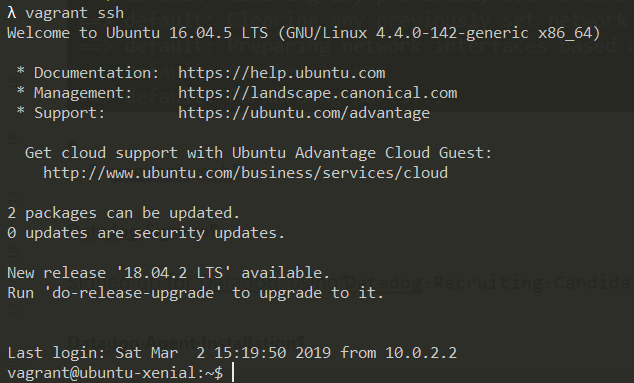
I initialized, activated and SSHed into the virtual machine using following commands on my terminal:

$ vagrant init ubuntu/xenial64

$ vagrant up



$ vagrant ssh

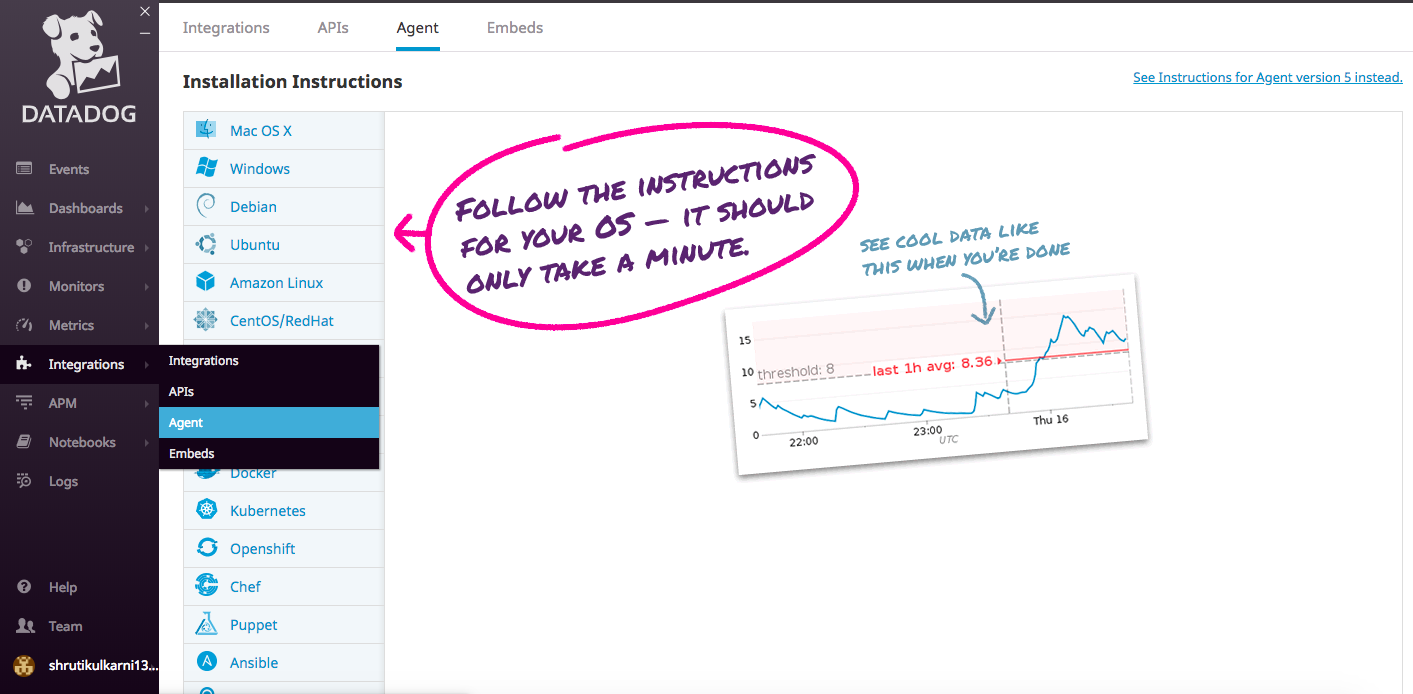


**Datadog Sign up**

Signed up for [Datadog](https://app.datadoghq.com/signup), using “Datadog Recruiting Candidate” in the Company field.

**Datadog Agent Installation**

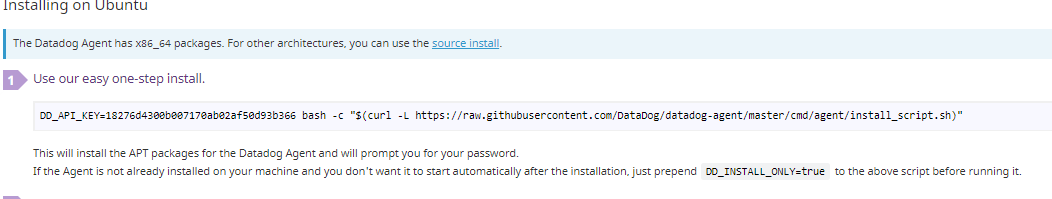
Navigated to the “Integrations” tab on the Datadog webapp and selected “ Agent” option:

[](https://github.com/Shrutiku/hiring-engineers/blob/Shruti_Kulkarni_Solutions_Engineer/screenshots/installation%20instructions.png)

Selected Ubuntu, which is my platform, and followed the 1-step installation instructions:

The following command was used on terminal to install the datadog-agent:

$ DD\_API\_KEY=18276d4300b007170ab02af50d93b366 bash -c "$(curl -L https://raw.githubusercontent.com/DataDog/datadog-agent/master/cmd/agent/ install\_script.sh)"



**Collecting Metrics**

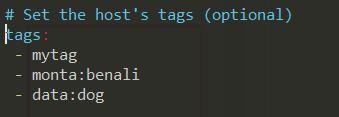
*Add tags in the Agent config file and show us a screenshot of your host and its tags on the Host Map page in Datadog.*

According to the descritiption on [Datadog site](https://docs.datadoghq.com/developers/write_agent_check/?tab=agentv6), I have to edit the “datadog.yaml” file to create the tags

To be able to make changes to the “datadog.yaml”  I had to use “sudo” admin privileges. This was done by initially navigating back to the vagrant root directory and then accessing “datadog.yaml” file from the datadog-agent directory. The commands used are as follows:

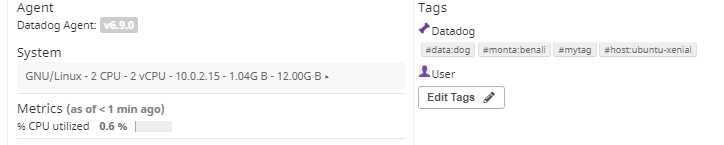


I opened the “datadog.yaml”file in my terminal and enabled and added tags to the file. See picture below::



I then closed the file, saving the changes and restarted the agent:

$ sudo service datadog-agent restart

To observe the added changes to the tags, you access the Datadog agent UI on-line, navigate to Host Map under “Infrastructure” tab, and you will see the updated tags. See picture below.

.

*Install a database on your machine (MongoDB, MySQL, or PostgreSQL) and then install the respective Datadog integration for that database.*

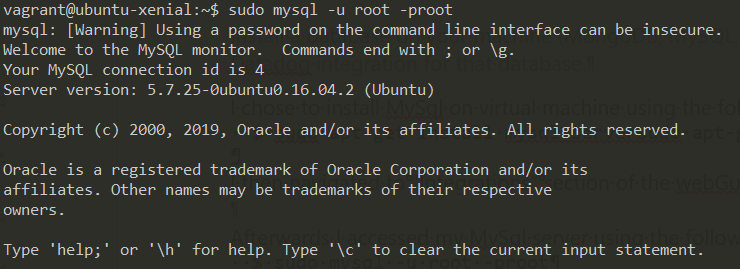
I chose to install MySql on virtual machine using the following command

$ sudo apt-get install mysql-serversudo apt-get update

I then navigated to “Integrations” section of the webGui and enabled MySql application.

Afterwards I accessed my MySql server using the following command:

$ sudo mysql -u root -proot

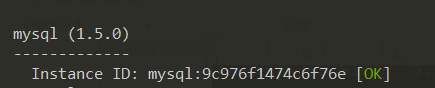


I then created a user on MySql with the proper permissions for the Datadog agent by following the instruction on [Datadog website for MySql Integration](https://docs.datadoghq.com/integrations/mysql/#data-collected) and restarted the agent:

$ sudo service datadog-agent restart

Then I executed the “Agent status” command to verify that the integration went well by running the following command:

$ sudo datadog-agent status



*Create a custom Agent check that submits a metric named my\_metric with a random value between 0 and 1000.*

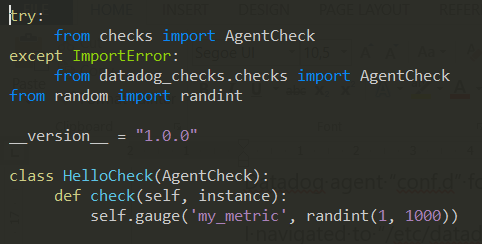
Following instruction on Datadog site “ [writing an Agent check](https://docs.datadoghq.com/developers/write_agent_check/?tab=agentv6)“, I created two files; “my\_metric.yaml” in the Datadog agent “conf.d” folder and “my\_metrci.py” in the “checks.d” folder

I navigated to “/etc/datadog-agent/checks.d” directory and created a file called [my\_metric.py](https://github.com/Shrutiku/hiring-engineers/blob/Shruti_Kulkarni_Solutions_Engineer/files/my_metric.py)

$ cd /etc/datadog-agent/checks.d

$ sudo vi my\_metric.py

And added the commands, as can be seen below



I then navigated to “/etc/datadog-agent/conf.d” and created the “my\_metric.yaml”  file there:

$ cd /etc/datadog-agent/conf.d

$ sudo nano my\_metric.yaml

Saved the following code in “my\_metric.yaml”

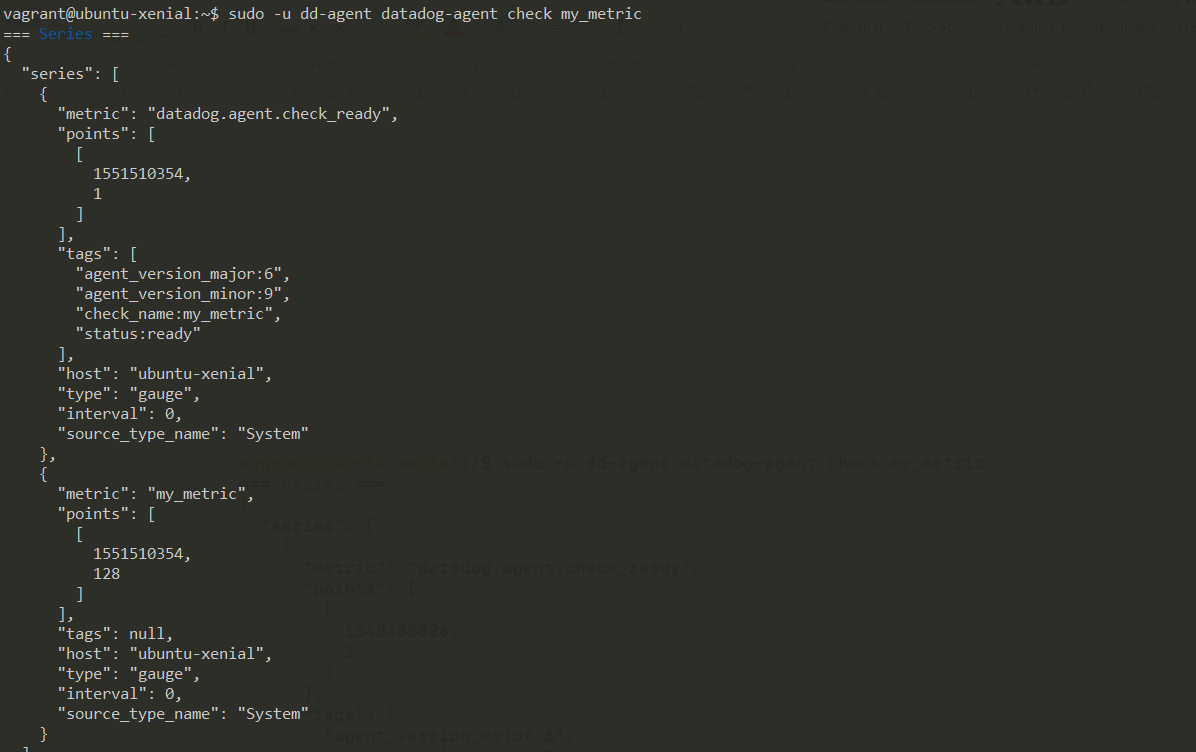


Restarted the Agent for the changes to reflect and checked the agent status:

$ sudo service datadog-agent restart

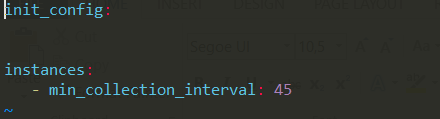
$ sudo service datadog-agent status

And then I ran a check on the configuration of the metric:



*Change your check's collection interval so that it only submits the metric once every 45 seconds.*

I opened the ”[my\_metric.yaml](https://github.com/Shrutiku/hiring-engineers/blob/Shruti_Kulkarni_Solutions_Engineer/files/my_metric.yaml)” file again to make the following changes before saving it again:



Restarted the Agent for the changes to reflect:

$ sudo service datadog-agent restart

*Bonus Question Can you change the collection interval without modifying the Python check file you created?*

Yes, the collection interval can be changed by editing the “my\_metric.yaml” file as made above

**Visualizing Data**

*Utilize the Datadog API to create a Timeboard that contains:*

* *Your custom metric scoped over your host.*
* *Any metric from the Integration on your Database with the anomaly function applied.*
* *Your custom metric with the rollup function applied to sum up all the points for the past hour into one bucket Please be sure, when submitting your hiring challenge, to include the script that you've used to create this Timeboard.*

I started by installing “pip” and Datadog :

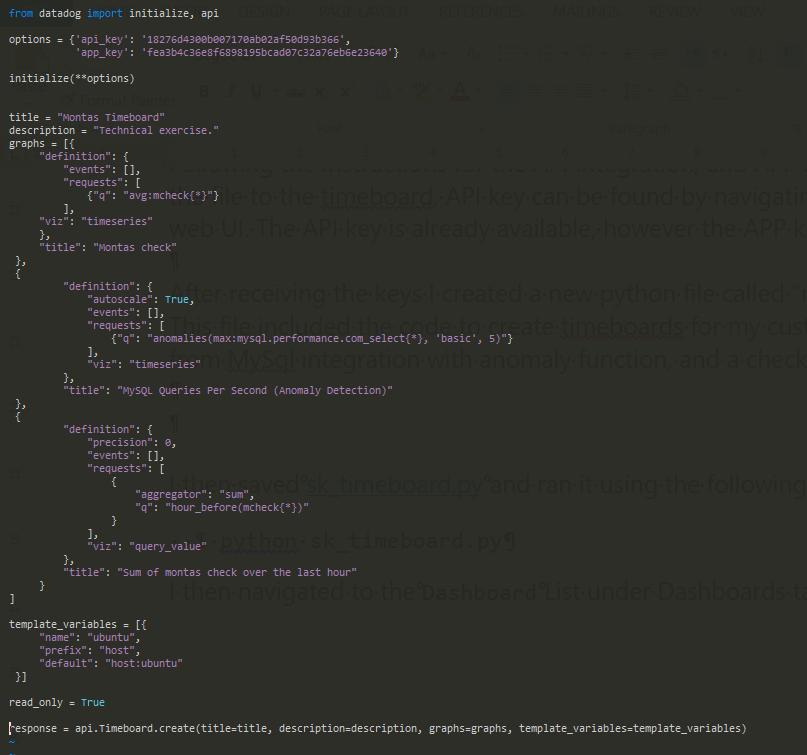
$ sudo apt-get install python-pip

$ pip install datadog

Following the instructions for the API integration, and APP key and API key is required to be able to create the file to the timeboard. API key can be found by navigating to APIs under Integrations option of Datadog web UI. The API key is already available, however the APP key has to be generated

After receiving the keys I created a new python file called “mytimeboard.py” in the Datadog agent directory.

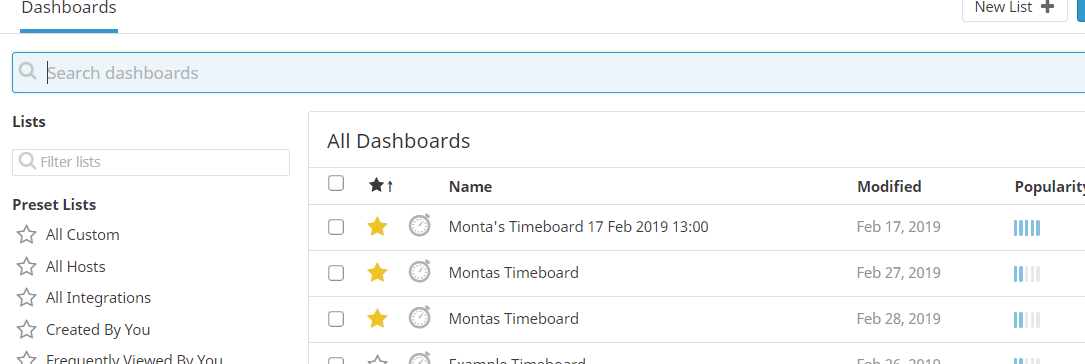
This file included the code to create timeboards for my custom metric scoped over the host, any metric from MySql integration with anomaly function, and a check of over the last hour



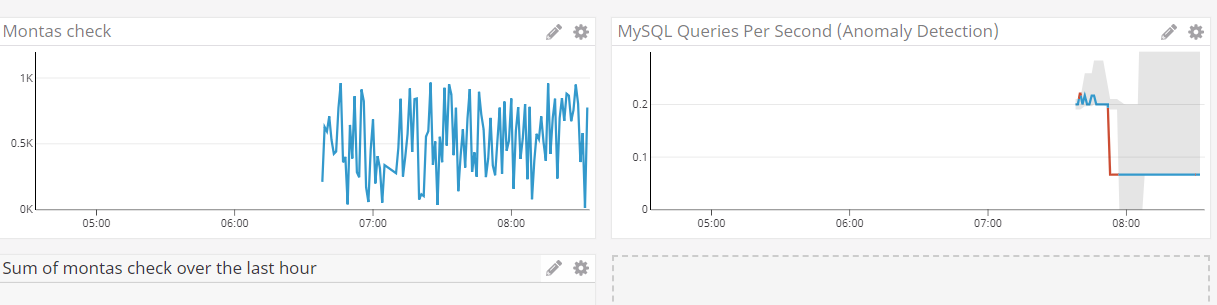
Saving “mytimboard.py” and ran it using the following terminal command:

$ python mytimeboard.py

I then navigated to the Dashboard List under Dashboards tab on the web UI:



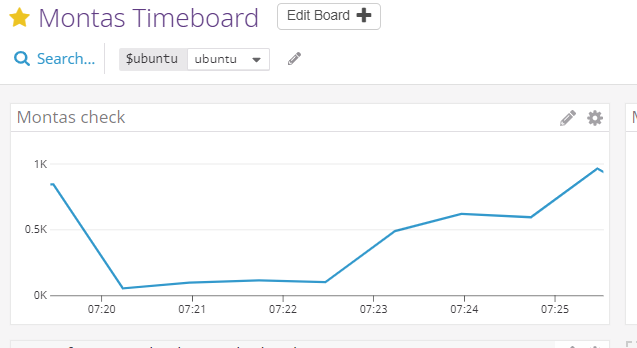
I could see that the timeboard was created successfully in the list and clicked into it to see graphs created for the metric created earlier.



*Once this is created, access the Dashboard from your Dashboard List in the UI:*

* *Set the Timeboard's timeframe to the past 5 minutes*
* *Take a snapshot of this graph and use the @ notation to send it to yourself.*

To set timeboard’s timeframe to past 5 minutes, I manually dragged the graph such that I could see the data for past 5 minutes:



I then clicked the small camera icon at the top of one of the metric as below, and added my name with @ notation in the comments, to send the snap to my email address, and the e-mail received is: 

.

* *Bonus Question: What is the Anomaly graph displaying?*

The anomaly graph is tracking whether the metric is behaving differently than the average and indicates the deviation in red to simplify the view of the anomalies over time

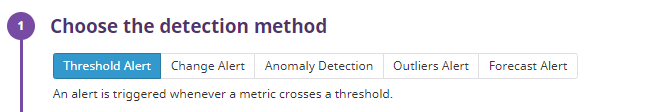
**Monitoring Data**

*Since you’ve already caught your test metric going above 800 once, you don’t want to have to continually watch this dashboard to be alerted when it goes above 800 again. So let’s make life easier by creating a monitor. Create a new Metric Monitor that watches the average of your custom metric (my\_metric) and will alert if it’s above the following values over the past 5 minutes:*

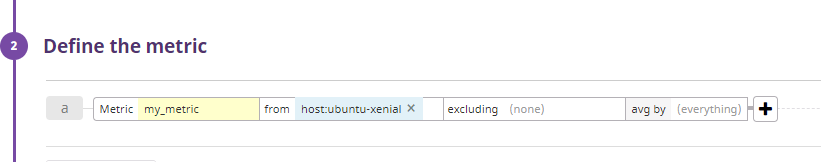
* *Warning threshold of 500*
* *Alerting threshold of 800*
* *And also ensure that it will notify you if there is No Data for this query over the past 10m.*

I accessed the Web UI and navigated to the “Monitors” section and chose “New Monitors”. Then I selected:

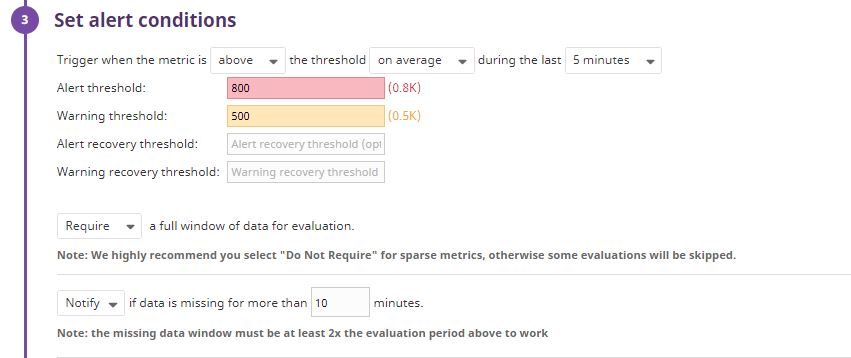
* “threshold method”



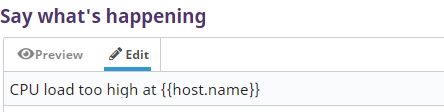
* Chose “my\_metric” in “Define the Metric)



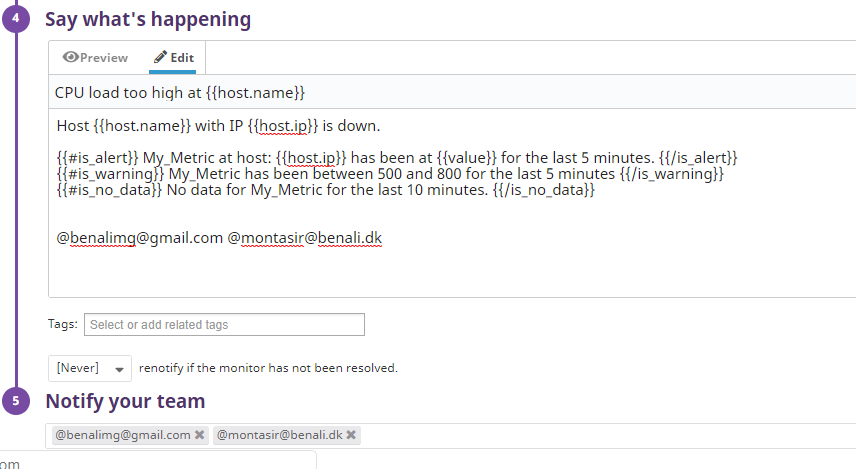
* Set the alert conditions:
  + Above, average, and 5 min
  + Alert threshold was set to 800
  + Warning threshold to 500
  + Selected “notify” if data is missing for 10 mins



* Added a name for the notification: “CPU load too high at {{host.name}}”

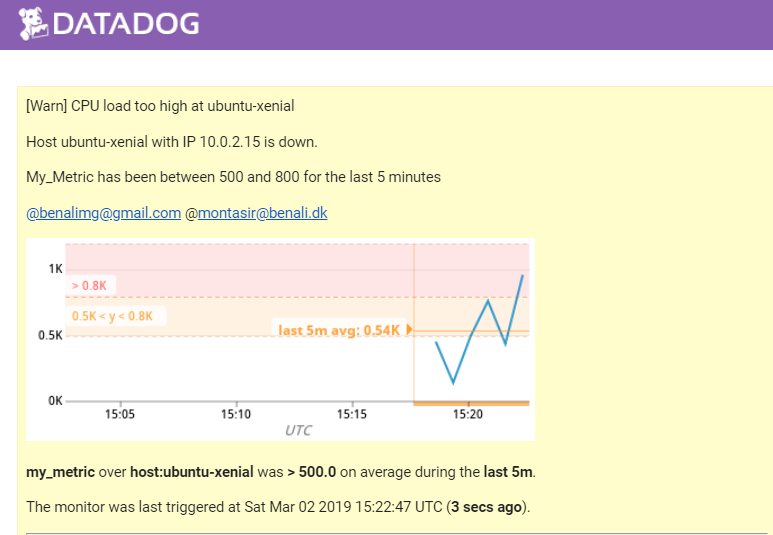


* Added the notifications configuration and the e-mail to forward them

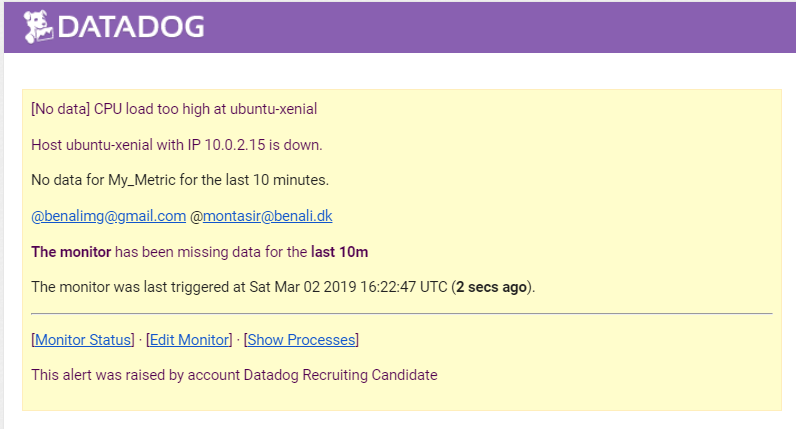


The notifications received are found below.

**Trigger: Warning – threshold too high**



**Trigger: No data**

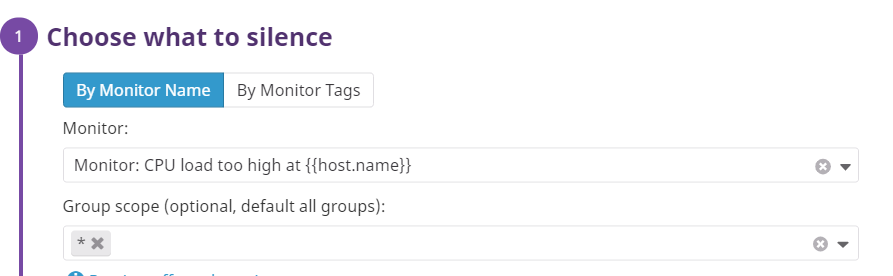


*Bonus Question: Since this monitor is going to alert pretty often, you don’t want to be alerted when you are out of the office. Set up two scheduled downtimes for this monitor:*

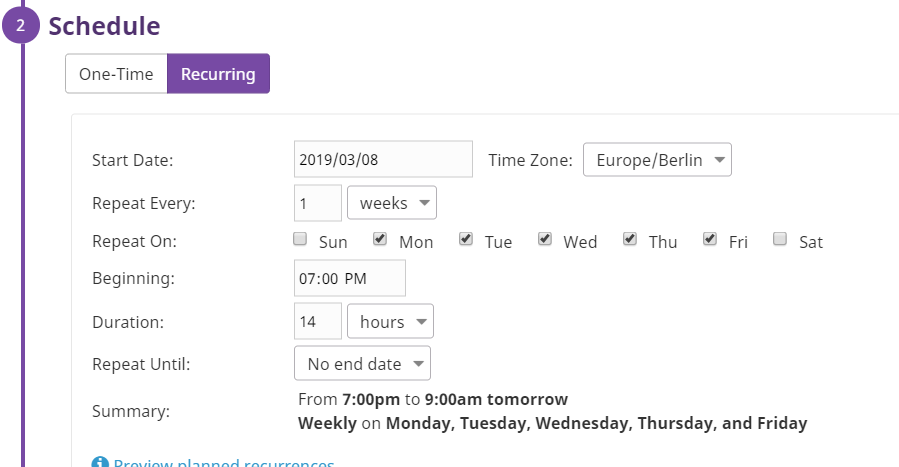
* *One that silences it from 7pm to 9am daily on M-F,*
* *And one that silences it all day on Sat-Sun.*
* *Make sure that your email is notified when you schedule the downtime and take a screenshot of that notification.*

I accessed the Web UI and navigated to the “Monitors” section and chose “Manage Monitors” and then “Manage downtime” at the top of the page. Then I selected:

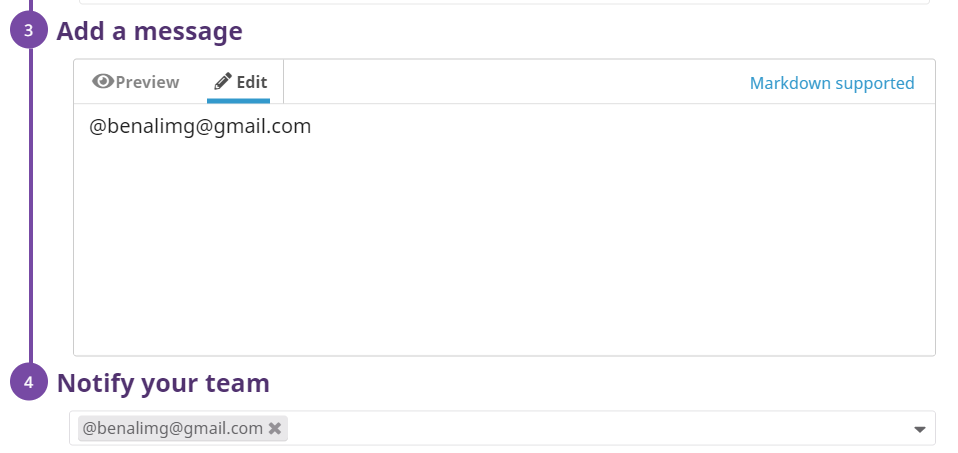
* By monitor name and the monitor



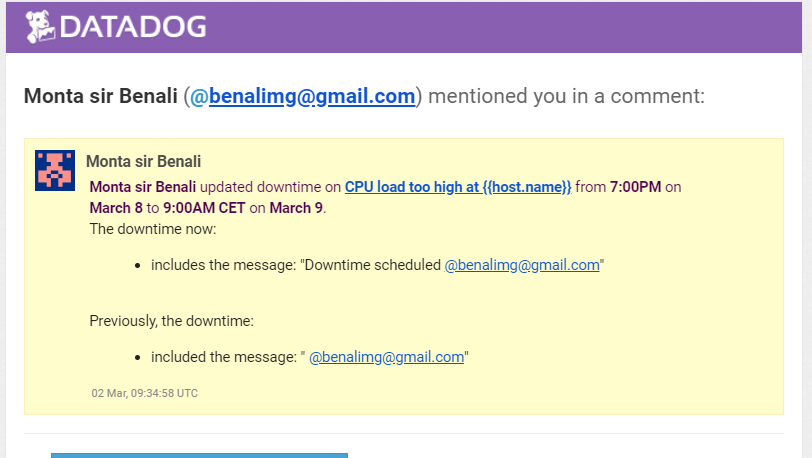
* Selected “recurring”:
  + Added the start date
  + Repeat every week
  + Selected the days
  + Start at 7:00 pm and 14 hours up front with no end date



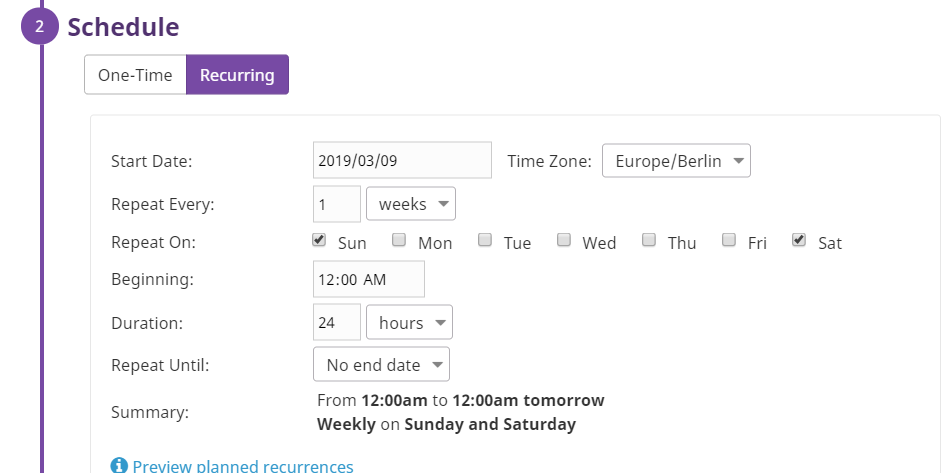
* + Added the e-mail for notification



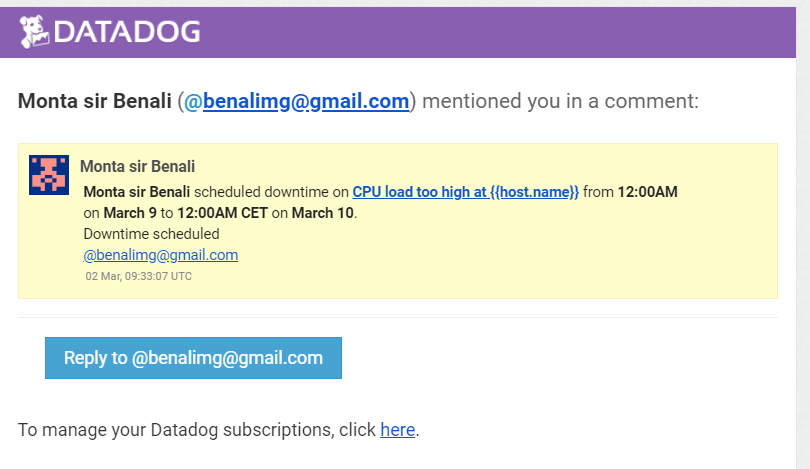
I then received an email confirmation for the weekday downtime scheduled:



By following the same procedure as above, I made schedule for the weekend.



I then received an email confirmation for the weekend downtime scheduled:



**Collecting APM Data**

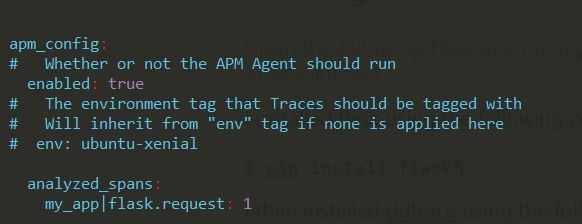
*Given the following Flask app (or any Python/Ruby/Go app of your choice) instrument this using Datadog’s APM solution:*

I installed flask using the following command:

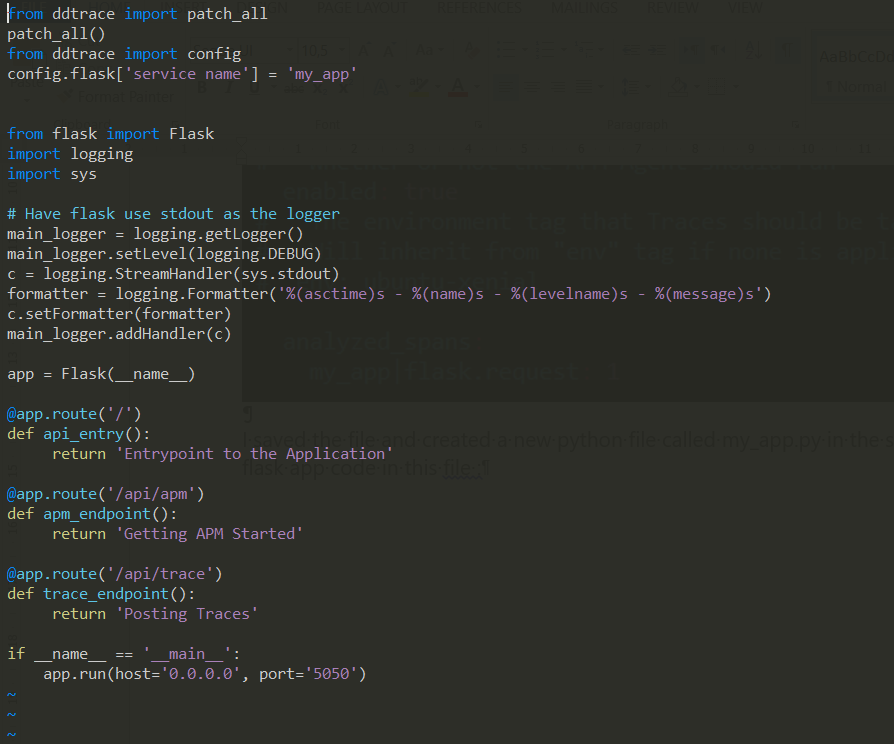
$ pip install flask

I then installed ddtrace using the following command:

$ pip install ddtrace

I then edited the “datadog.yaml” file in the”datadog-agent” directory to enable trace collection for the trace agent. I updated the “datadog.yaml” as can be seen in the following picuter

I saved the file and created a new python file called my\_app.py in the same directory. I added the given flask app code in this file :



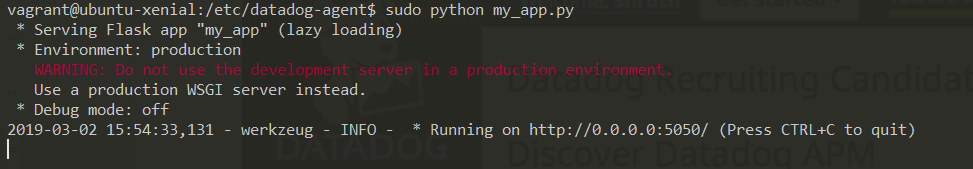
I then restarted the datadog-agent:

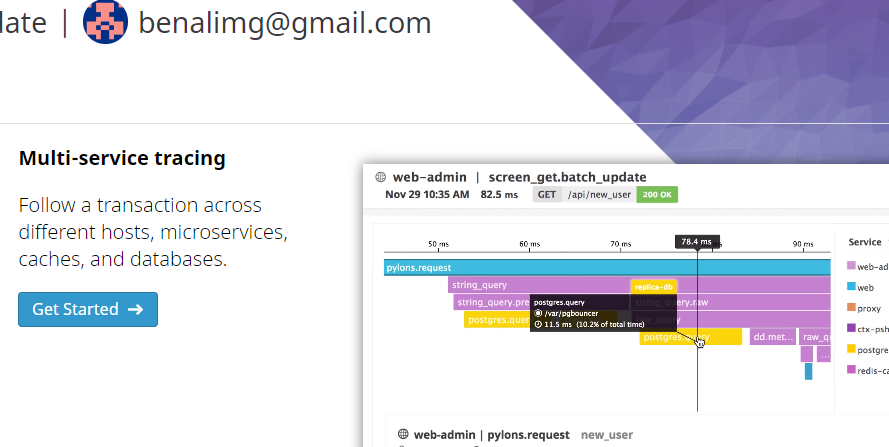
sudo service datadog-agent restart

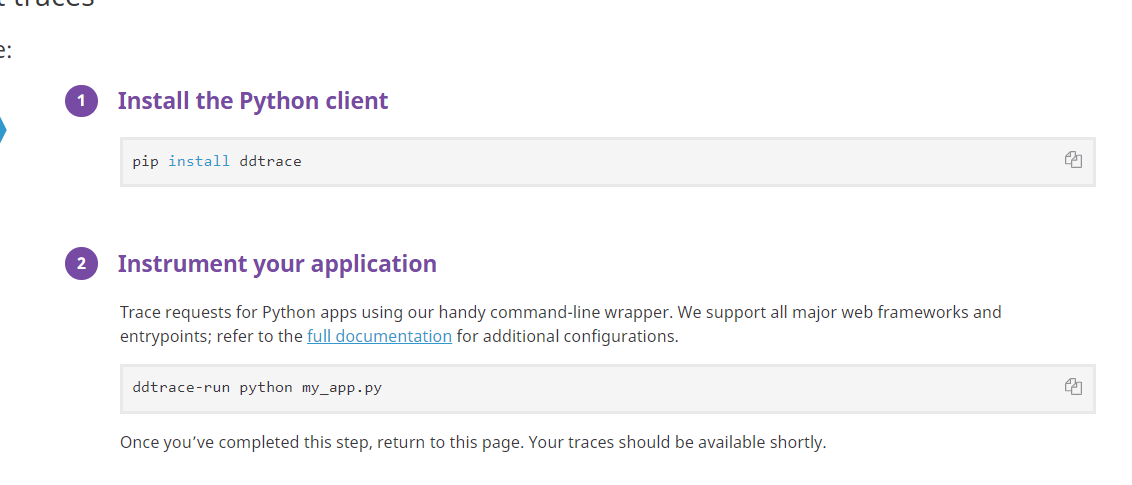
I then ran “my\_app.py” file using ddtrace:

ddtrace-run python my\_app.py

The below screenshot shows the result of the ddtrace-run, however the result is not as expected. I was not able to get the APM up and running.







I tried to solve this issue by using different updates and changes to my\_app.py file with no success. I went through all the pages on the datadoghq.com pages with no luck. I event tried to uninstall the datadog agent and restart the entire process with no luck.

*Bonus Question: What is the difference between a Service and a Resource?*

Basically a service is a group of different resources e.g. a service can be a web app service and the rsource could be an action for this service

*Provide a link and a screenshot of a Dashboard with both APM and Infrastructure Metrics. Please include your fully instrumented app in your submission, as well.*

My flask app can be found [here](https://github.com/Shrutiku/hiring-engineers/blob/Shruti_Kulkarni_Solutions_Engineer/files/app.py).

**Final Question**

*Datadog has been used in a lot of creative ways in the past. We’ve written some blog posts about using Datadog to monitor the NYC Subway System, Pokemon Go, and even office restroom availability! Is there anything creative you would use Datadog for?*

There are two area, where I can see this application being used.

* 1. Professional athletes. If one can monitor their conditions during different form of conditions and exercise, one can get a clear view of, when a person is peaking and when his performance is being reduced
  2. Video service monitoring. Today; the likes of Netflix and Amazon are spending millions of dollars in building the proper networks and streaming solutions. I believe that Datadog would be a perfect fit for such applications

Additionally, many multibillion dollar companies (Cisco, Microsoft etc) in cloud based applications, which makes the Datadog product very interesting for the IoT market

**Feedback**

It was interesting doing this exercise. I had to refresh old linux knowledge to succeed with this exercise. I think this is a great way of introducing the product to new hires, which will help in deciding whether this is a product that you believe in, and would like to sell and introduce to the market