May 2018

Oblig 2 & 3 Report

DATS2410 Computer Networking and Cloud Computing

Table of Contents

List of Figures

Introduction

This group work consists of setting up a cloud-based web application architecture using LEMP stack in OpenStack (ALTO), as well as developing and deploying a simple application in the cloud. The cloud-based application has a load balancer, three web servers, a database proxy and three database servers. This is illustrated in the figure below in task 1. The naming conventions provided in the task description was used to name the VMs with our group number included.

The load balancer is of m1.1GB flavor, and all other VMs are of m1.512MB flavor as requested. The remaining setup was also performed as requested in the task description:

OS: Ubuntu 16.04

Web server: Nginx v1.10.3

Load balancer: HAProxy v1.6.3

Database server: MariaDB v10.2.14

Server-side programming: PHP v7.0.28

Database proxy: MariaDB MaxScale v2.2.2

The whole work of this oblig can be divided into 10 tasks, which includes implementing in ALTO cloud and providing details in the submitted report. These will be presented in this report together with the problem description and useful screenshots. The task description is written in italic to distinguish it from the answers provided by us.

Task 1 - VM Setup

This task consists of creating a ssh key (datsXX - key) and a security group (datsXX - security), creating VMs with desired flavor, doing minimal required common configurations in VMs such as naming hosts, and setup locales. From the security point of view, only the required outside access (such as ssh, web, etc.) to the VMs must be given. This means you should open the required ports only in the security group.

Task 1.1

The report should provide an architecture diagram of your cloud setup, where all the VMs are labelled with your host names, IPs, port numbers.

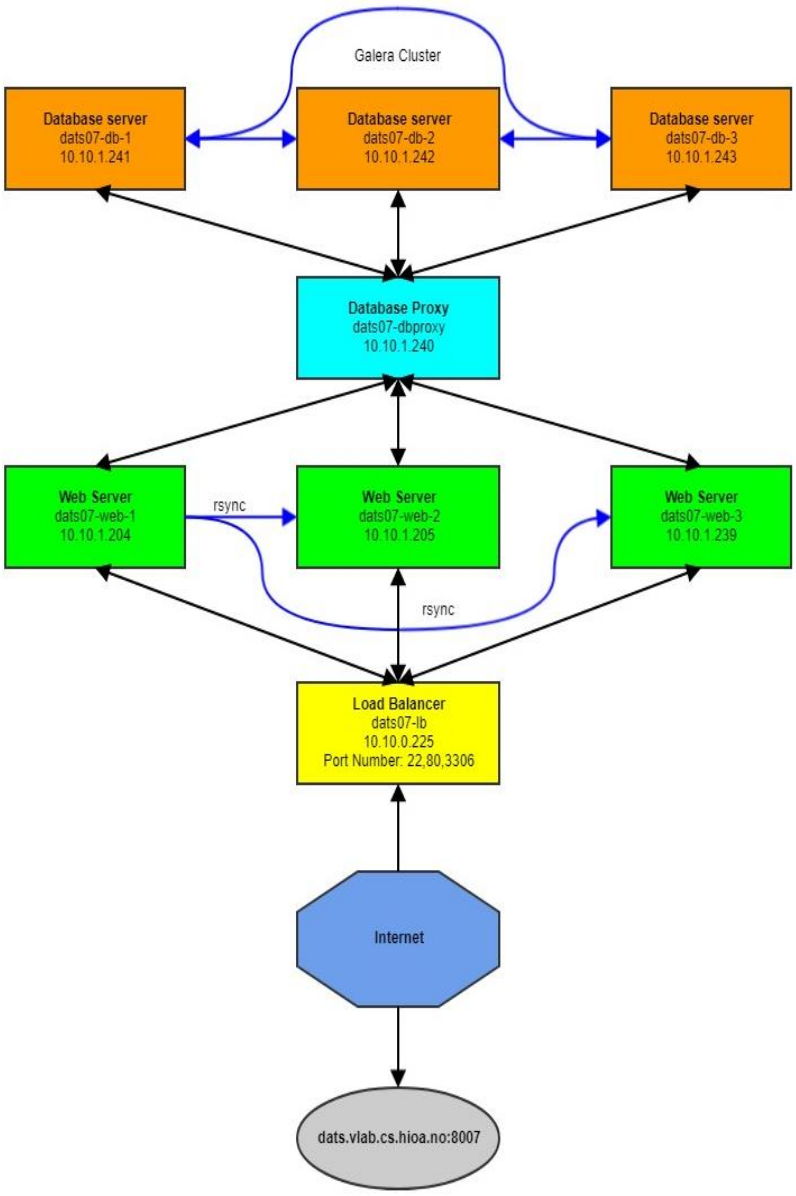


Figure 1: Architecture diagram of the cloud setup

Task 2.2

The report should provide screenshots from ALTO showing network diagram, created ssh key, security group and security group rules. Label security group rules indicating purpose of the rules.

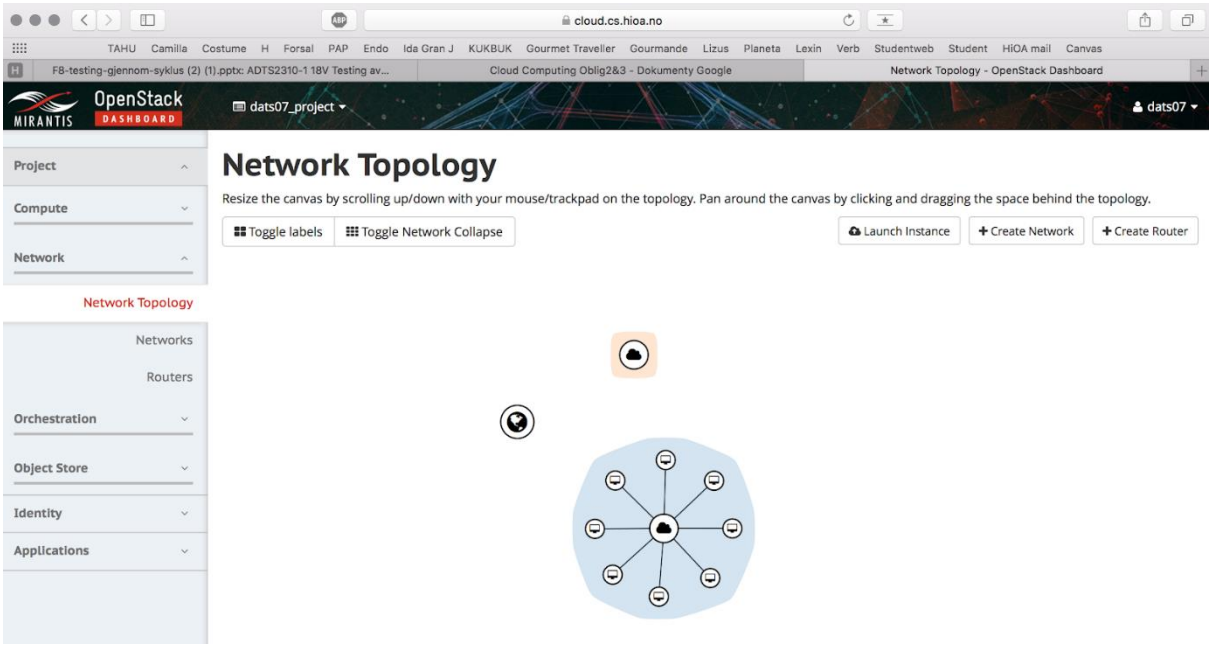


Figure 2:Screenshot from ALTO showing Network Topology

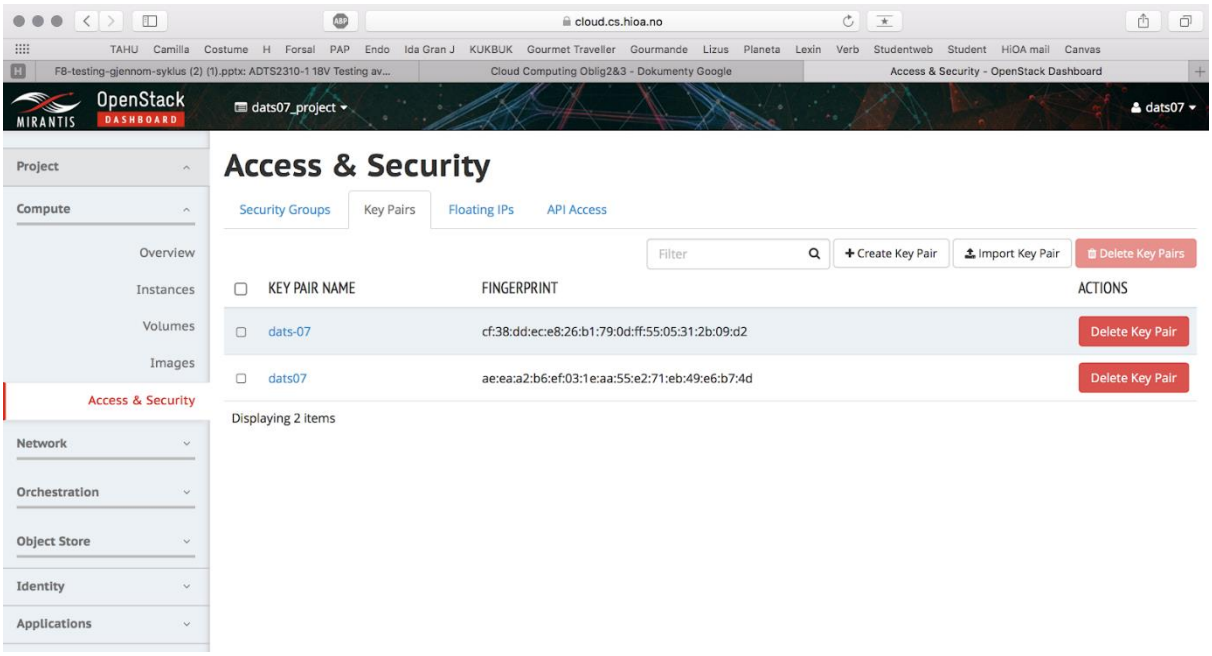


Figure 3: Screenshot from ALTO showing SSH Key

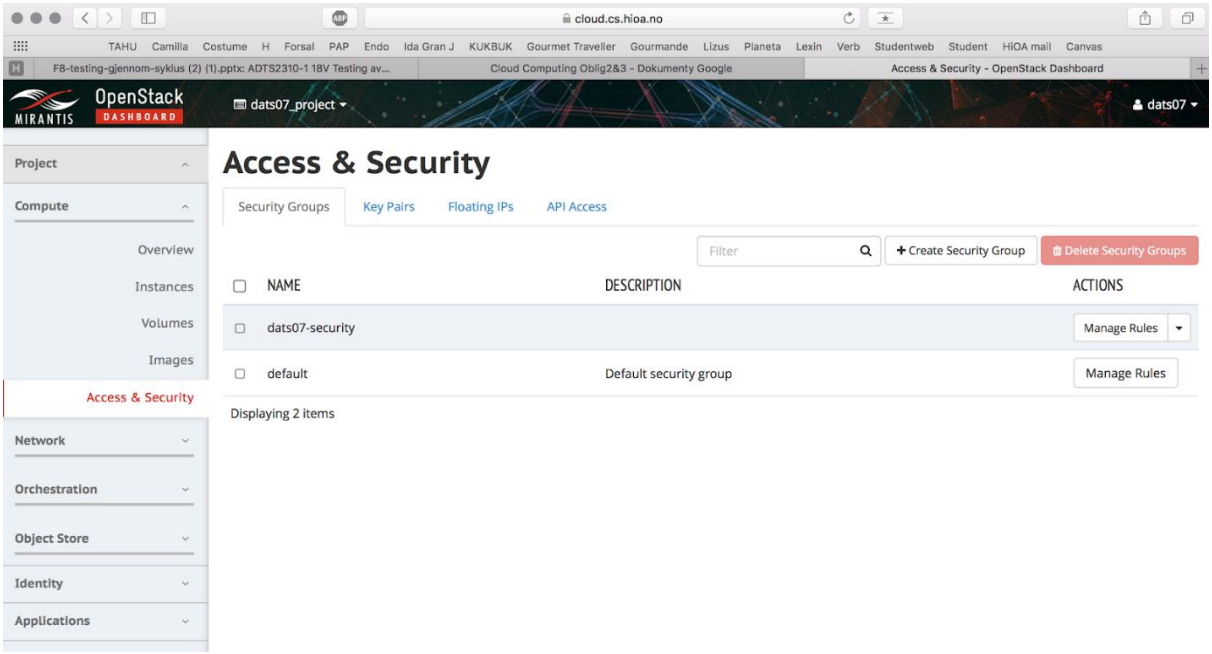


Figure 4: Screenshot from ALTO showing security group

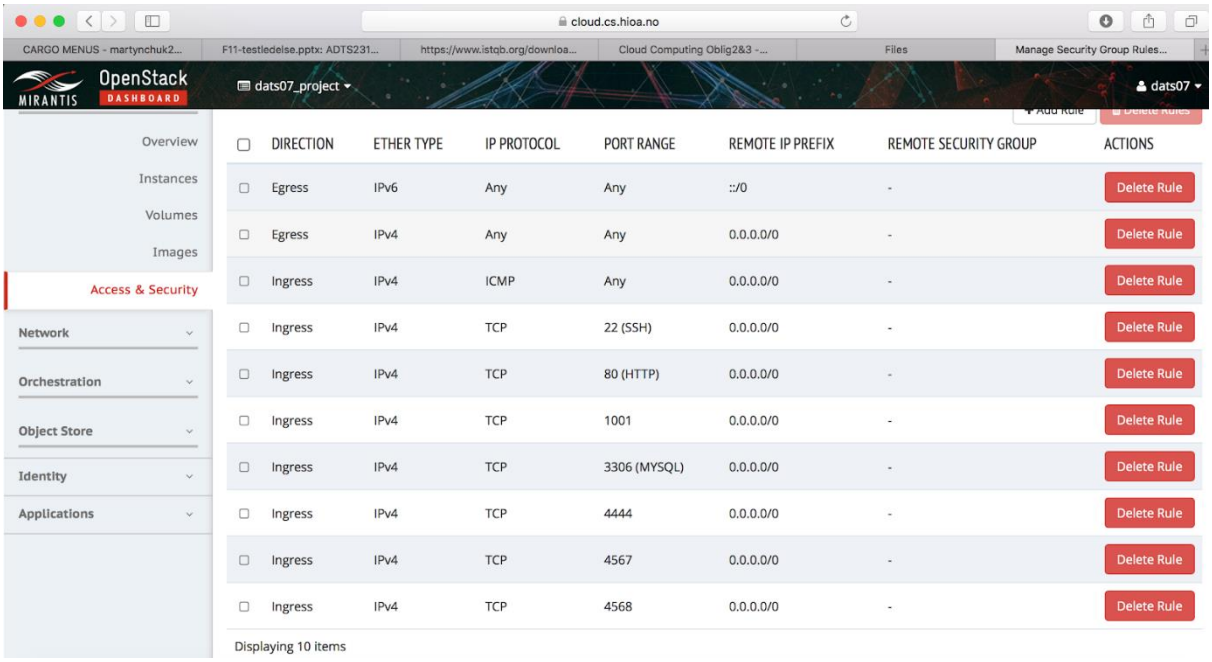


Figure 5:Screenshot from ALTO showing security group rules

Port 22 allow SSH access to Linux instances from IPv4 IP addresses in our network.

Port 80 allow HTTP access from IPv4 addresses.

Port 3306 allow MySQL access to instances in the specified security group.

Port 4444 is used for State Snapshot Transfer (State Snapshot Transfer is a full data copy from one node (donor) to the joining node (joiner)).

Port 4567 is used for Galera Cluster.

Port 4568 is used for Incremental State Transfer (In an Incremental State Transfer (IST) a node only receives the missing write-sets and catch up with the group by replaying them)

Port 1001 is used for having access to check status/monitor haproxy.

Task 1.3

The report should provide a table listing the VMs with these information: VM name, IP, flavor, software you installed in the VM. Also, provide a screenshot of the VMs created.

VM Name	IP	Flavor	Software installed
dats07-lb	10.10.0.225	m1.1GB	HAProxy(v1.6.3)
dats07-web-1	10.10.1.204	m1.512MB	nginx (v1.10.3), MariaDB(v10.2.14), php (v7.0.28), php-fpm, php-mysql, rsync v3.1.1
dats07-web-2	10.10.1.205	m1.512MB	nginx (v1.10.3), MariaDB(v10.2.14), php (v7.0.28), php-fpm, php-mysql, rsync (v3.1.1)
dats07-web-3	10.10.1.239	m1.512MB	nginx (v1.10.3), MariaDB(v10.2.14), php (v7.0.28), php-fpm, php-mysql, rsync (v3.1.1)
dats07-proxy	10.10.1.240	m1.512MB	MariaDB(v10.2.14), Maxscale(v2.2.2)
dats07-db-1	10.10.1.241	m1.512MB	MariaDB(v10.2.14)
dats07-db-2	10.10.1.242	m1.512MB	MariaDB(v10.2.14)
dats07-db-3	10.10.1.243	m1.512MB	MariaDB(v10.2.14)

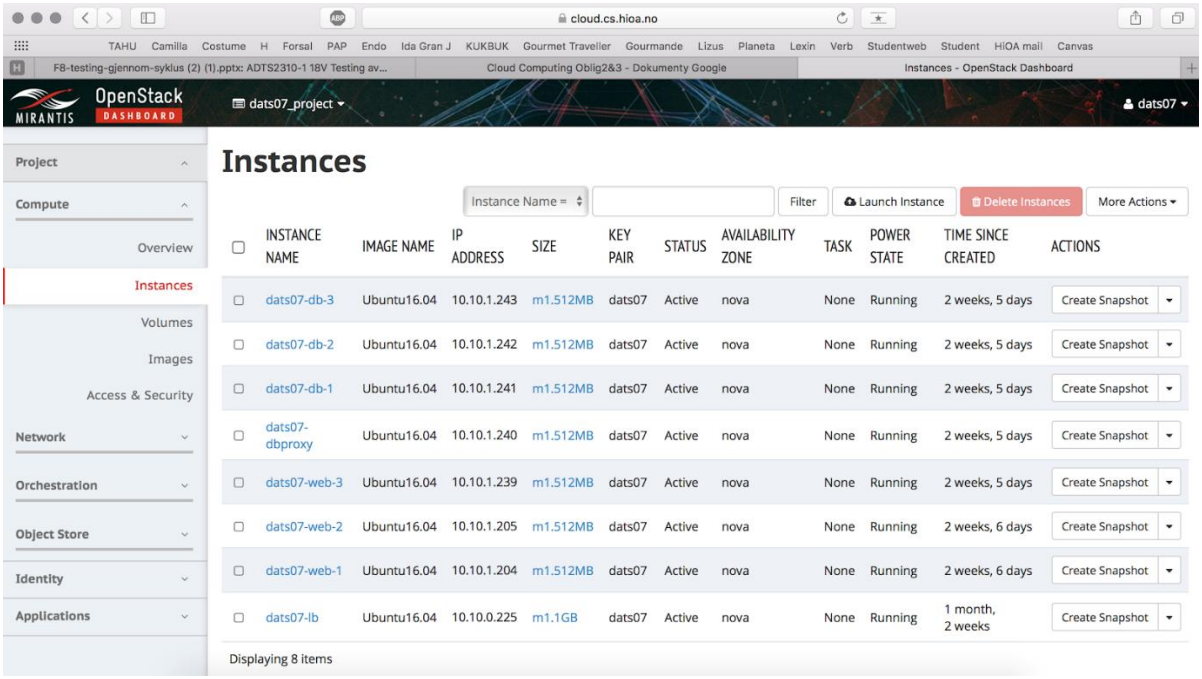


Figure 6:Screenshot from ALTO showing the VMs created

Task 1.4

The report should provide a screenshot of the VM names you defined in /etc/hosts of one of the servers (say, datsXX- lb). Give short names to your servers here, such as lb, web1, web2, web3, db1, db2, db3, and maxscale and use names in your configurations instead of hardcoded IPs.

```
ira — ubuntu@lb: ~ — ssh dats07@dats.vlab.cs.hioa.no — 80×24
0 updates are security updates.


*** System restart required ***
Last login: Mon Apr 30 12:01:21 2018 from 10.10.0.14
ubuntu@lb:~$ which nginx
ubuntu@lb:~$ cat /etc/hosts
127.0.0.1 localhost
10.10.0.225 lb
10.10.1.204 web1
10.10.1.205 web2
10.10.1.239 web3
10.10.1.240 maxscale
10.10.1.241 db1
10.10.1.242 db2
10.10.1.243 db3
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
ubuntu@lb:~$
```

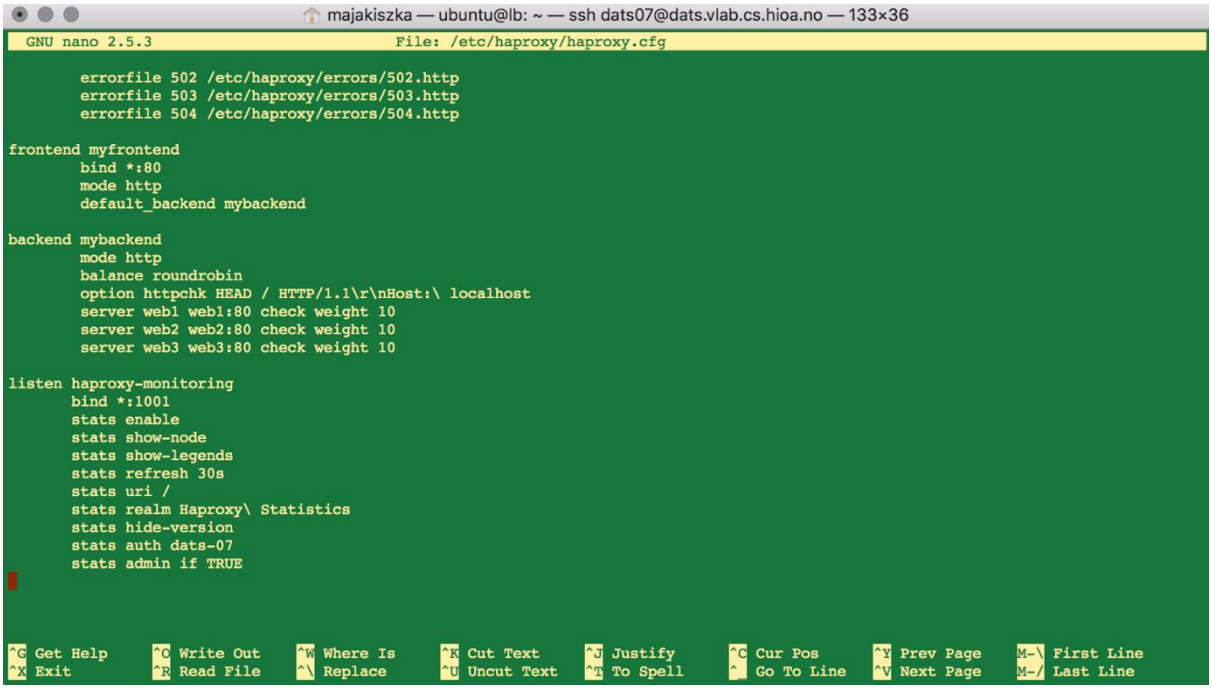
Figure 7:Screenshot of VM names defined in /etc/hosts, here on Load Balancer

Task 2 - Load Balancer (LB) Setup

Setup load balancer using HAProxy so that user requests are distributed to the three web servers in a round robin basis with equal weights.

Task 2.1

Provide screenshots of the HAProxy configurations you have done in the load balancer.



Task 2.2

Show test results confirming working load balancer, i.e. alternately switching to the 3 web servers.

The simple PHP file (myphp.php) used to test that the load balancer is working is as follows:

<?php

echo "Server IP: " . $_SERVER['SERVER_ADDR'];

?>

Below are screenshots from the test.

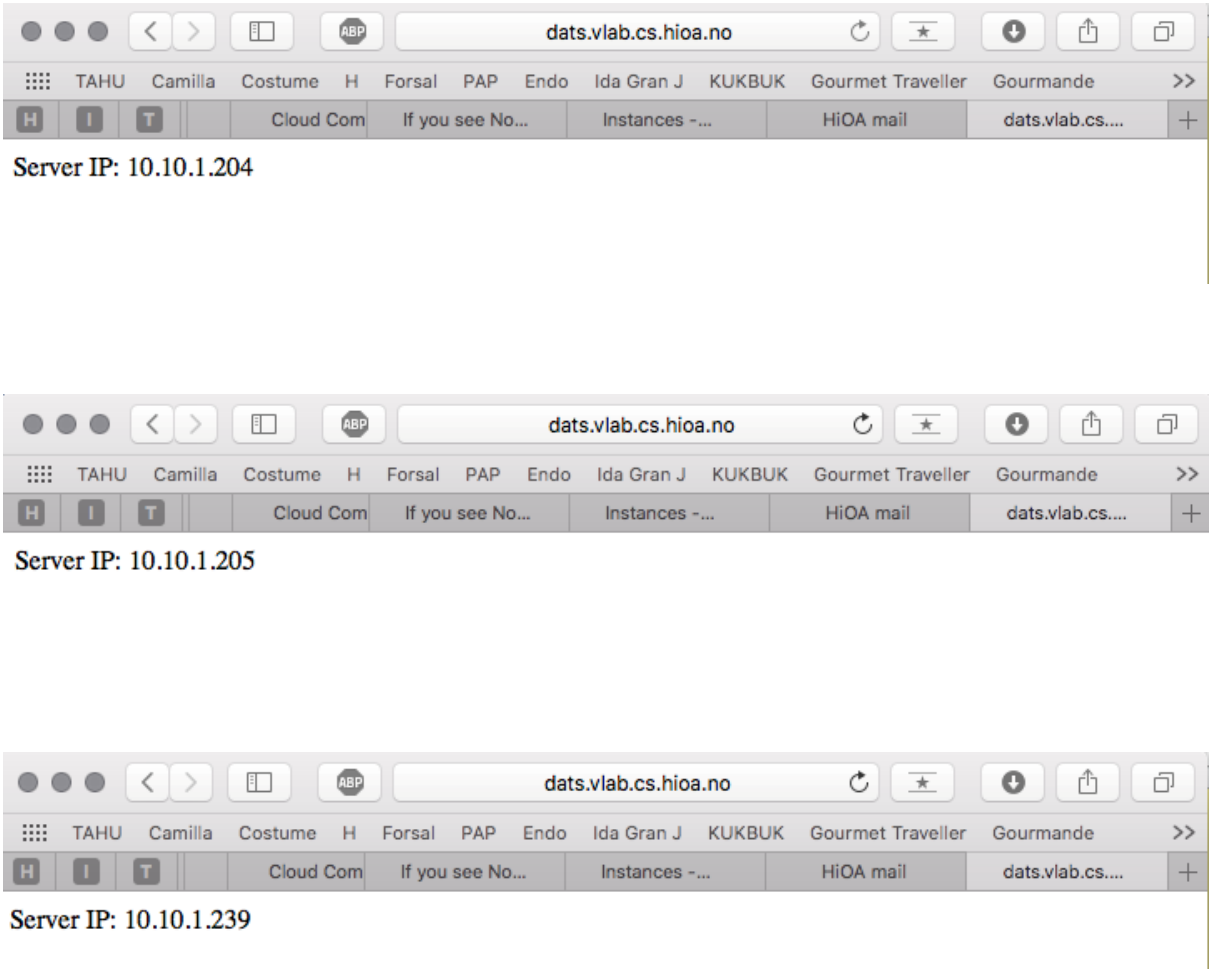


Figure 8: Screenshots from http://dats.vlab.cs.hioa.no:8007/myphp.php

Task 3 - Web Server Setup

Setup all the web servers using Nginx with the support for dynamic web development with PHP and MariaDB and give a proper ownership and permission to the webroot folder for 'ubuntu' user.

Task 3.1

Provide a screenshot of Nginx configuration you updated to enable PHP support in datsXX- web-1.

WEB1

```
ira — ubuntu@web1: ~ — ssh dats07@dats.vlab.cs.hioa.no — 80×24
ff02::3 ip6-allhosts
ubuntu@lb:~$ exit
logout
Connection to 10.10.0.225 closed.
dats07@dats-master:~$ ssh -i dats-07 ubuntu@10.10.1.204
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-121-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

107 packages can be updated.
0 updates are security updates.


Last login: Wed May  2 09:30:51 2018 from 10.10.0.14
ubuntu@web1:~$ users
ubuntu ubuntu
ubuntu@web1:~$ groups
ubuntu adm dialout cdrom floppy sudo audio dip www-data video plugdev netdev lxd
ubuntu@web1:~$
```

```
ubuntu@web1:~$ namei -m /var/www/
f: /var/www/
 drwxr-xr-x /
 drwxr-xr-x var
 drwxrwxr-x www
ubuntu@web1:~$
```

```
ubuntu@web1:~$ ls -l /var/www/
total 4
drwxr-xr-x 2 root root 4096 Apr 13 11:00 html
ubuntu@web1:~$
```

Figure 9: Screenshot of ownership configuration of Webroot folder in web1

WEB2

```
ubuntu@web2:~$ users
ubuntu
ubuntu@web2:~$ namei -m /var/www/
f: /var/www/
 drwxr-xr-x /
 drwxr-xr-x var
 drwxrwxr-x www
ubuntu@web2:~$ ls -l /var/www/
total 4
drwxrwxr-x 2 www-data www-data 4096 Apr 27 09:39 html
ubuntu@web2:~$
```

Figure 10: Screenshot of ownership configuration of Webroot folder in web2

WEB3

```
ubuntu@web3:~$ users
ubuntu
ubuntu@web3:~$ namei -m /var/www/
f: /var/www/
 drwxr-xr-x /
 drwxr-xr-x var
 drwxrwxr-x www
ubuntu@web3:~$ ls -l /var/www/
total 4
drwxrwxr-x 2 www-data www-data 4096 Apr 25 14:39
ubuntu@web3:~$
```

Figure 11: Screenshot of ownership configuration of Webroot folder in web3

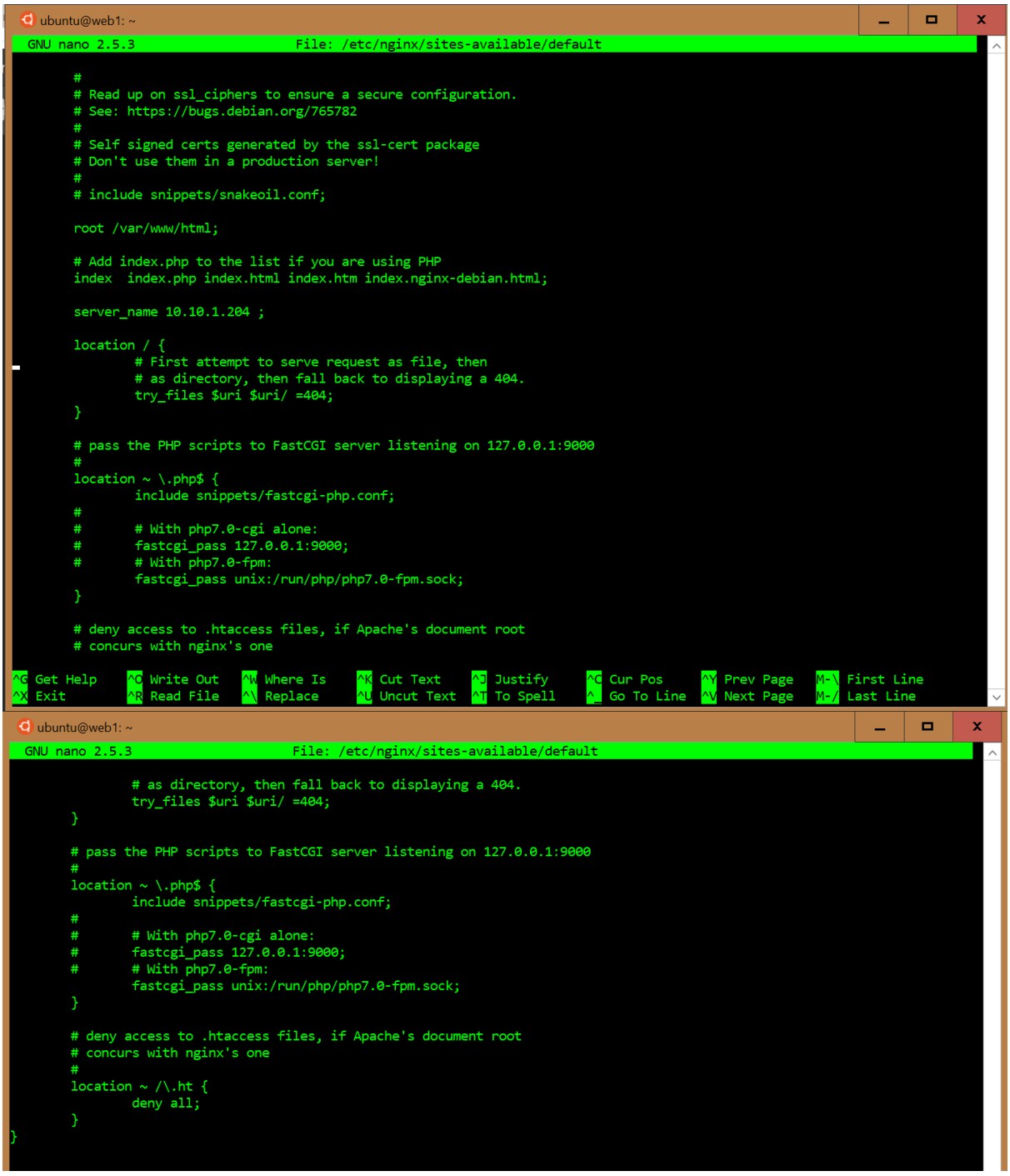


Figure 12: Screenshot of nginx configurations

Task 3.2

Setup a simple web deployment mechanism, where datsXX-web-1 is considered as the primary web server and whenever an updated web application is deployed to this server, the application is synchronized (pushed) to the other web servers automatically in every 2 minutes (using rsync and crontab). Provide a screenshot of the crontab showing the rsync command you used for synchronizing the web servers.

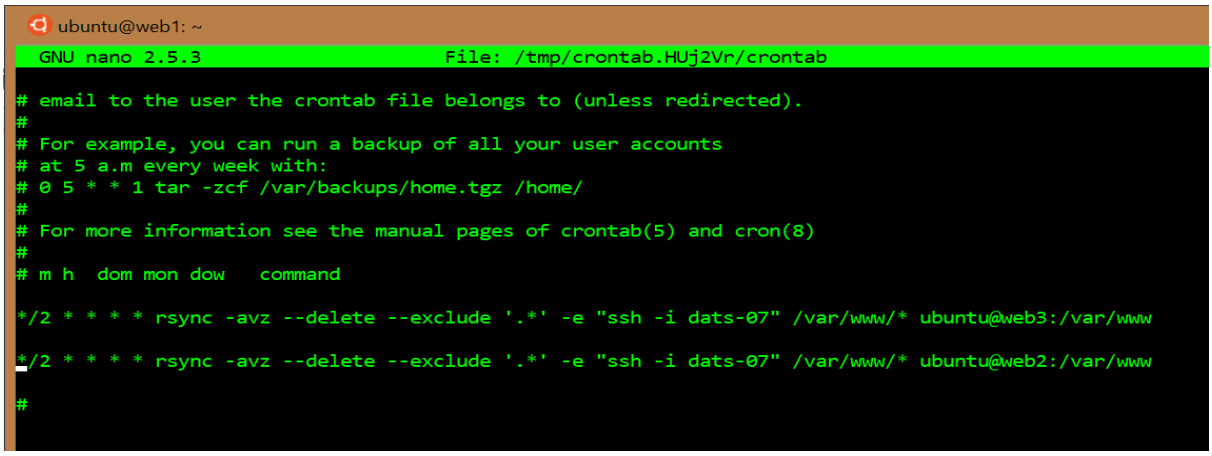


Figure 13: Screenshot of crontab configured to push changes from web1 to web2 and web3 every 2 minutes

Task 4 - HA Database Setup

Setup a cluster-based high availability database with Galera cluster of three MariaDB database servers (nodes) and a MaxScale database proxy.

Task 4.1

Provide screenshots of configurations of the 3 database servers in the Galera cluster and the MaxScale proxy server.

```
[galera]
# Mandatory settings
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
wsrep_cluster_address=gcomm://db1, db2, db3
binlog_format=row
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
wsrep_sst_method=rsync
#
# Allow server to accept connections on all interfaces.
#
bind-address=0.0.0.0
#
# Optional setting
#wsrep_slave_threads=1
#innodb_flush_log_at_trx_commit=0

[mysqldump]
quick
quote-names
max_allowed_packet      = 16M

[mysql]
#no-auto-rehash # faster start of mysql but no tab completion

[isamchk]
key_buffer              = 16M

#
# * IMPORTANT: Additional settings that can override those from
#   The files must end with '.cnf', otherwise they'll be ignore
#
!includedir /etc/mysql/conf.d/
ubuntu@db1:~$
```

```
# * Galera-related settings
#
[galera]
# Mandatory settings
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
wsrep_cluster_address=gcomm://db1, db2, db3
binlog_format=row
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
wsrep_sst_method=rsync
#
# Allow server to accept connections on all interfaces
#
bind-address=0.0.0.0
#
# Optional setting
#wsrep_slave_threads=1
#innodb_flush_log_at_trx_commit=0

[mysqldump]
quick
quote-names
max_allowed_packet      = 16M

[mysql]
#no-auto-rehash # faster start of mysql but no tab com

[isamchk]
key_buffer              = 16M

#
# * IMPORTANT: Additional settings that can override t
#   The files must end with '.cnf', otherwise they'll
#
!includedir /etc/mysql/conf.d/
ubuntu@db2:~$
```

```
# * Galera-related settings
#
[galera]
# Mandatory settings
wsrep_on=ON
wsrep_provider=/usr/lib/galera/libgalera_smm.so
wsrep_cluster_address=gcomm://db1, db2, db3
binlog_format=row
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
wsrep_sst_method=rsync
#
# Allow server to accept connections on all interfac
#
bind-address=0.0.0.0
#
# Optional setting
#wsrep_slave_threads=1
#innodb_flush_log_at_trx_commit=0

[mysqldump]
quick
quote-names
max_allowed_packet      = 16M

[mysql]
#no-auto-rehash # faster start of mysql but no tab c

[isamchk]
key_buffer              = 16M

#
# * IMPORTANT: Additional settings that can overrid
#   The files must end with '.cnf', otherwise they'l
#
!includedir /etc/mysql/conf.d/
ubuntu@db3:~$
```

```
ubuntu@db1:~$ sudo cat /etc/mysql/debian.cnf
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
[mysql_upgrade]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
basedir  = /usr
ubuntu@db2:~$ sudo cat /etc/mysql/debian.cnf
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
[mysql_upgrade]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
basedir  = /usr
ubuntu@db3:~$ sudo cat /etc/mysql/debian.cnf
# Automatically generated for Debian scripts. DO NOT TOUCH!
[client]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
[mysql_upgrade]
host     = localhost
user     = debian-sys-maint
password = YDnJ3vBAy8QlwtA9
socket   = /var/run/mysqld/mysqld.sock
basedir  = /usr
ubuntu@db1:~$ sudo cat /var/lib/mysql/grastate.dat
# GALERA saved state
version: 2.1
uuid:    3940fd82-447c-11e8-bf00-2b6ee5acbbe5
seqno:   -1
safe_to_bootstrap: 1
```

Figure 14: Screenshots of configurations of the 3 database servers in the Galera cluster

```
[maxscale]
threads = 4

[Galera-Monitor]
type = monitor
module = galeramon
servers = db1, db2, db3
user = maxscaleuser
passwd = iramaja
monitor_interval = 10000
disable_master_failback = 1
```

```
[Read-Write-Service]
type = service
router = readwritesplit
servers = db1, db2, db3
user = maxscaleuser
passwd = iramaja
max_slave_connections = 1
router_options = slave_selection_criteria=LEAST_GLOBAL_CONNECTIONS,master_failure_mode=error_on_write
weightby = serv_weight
enable_root_user = true
```

```
[Read-Write-Listener]
type = listener
service = Read-Write-Service
protocol = MySQLClient
port = 3306
```

```
[MaxAdmin-Listener]
type = listener
service = MaxAdmin-Service
protocol = maxscaled
socket = default

[db1]
type = server
address = 10.10.1.241
port = 3306
protocol = MySQLBackend
serv_weight = 2

[db2]
type = server
address = 10.10.1.242
port = 3306
protocol = MySQLBackend
serv_weight = 1

[db3]
type = server
address = 10.10.1.243
port = 3306
protocol = MySQLBackend
serv_weight = 1
```

Figure 15: Screenshots of configurations of the MaxScale proxy server.

Task 4.2

Provide a screenshot confirming Galera cluster size to 3.

```
ubuntu@db3:~$ mysql -u root -e "SHOW STATUS LIKE 'wsrep_cluster_size'"
+--------------------+-------+
| Variable_name      | Value |
+--------------------+-------+
| wsrep_cluster_size | 3     |
+--------------------+-------+
```

Figure 16: Screenshot confirming Galera cluster size to 3

Task 4.3

Also, show the list of servers from the command maxadmin list servers in MaxScale.

```
ubuntu@maxscale:~$ sudo maxadmin list servers
Servers.
-------------------+-----------------+-------+-------------+--------------------
Server             | Address         | Port  | Connections | Status
-------------------+-----------------+-------+-------------+--------------------
db1                | 10.10.1.241     |  3306 |           0 | Master, Synced, Running
db2                | 10.10.1.242     |  3306 |           0 | Slave, Synced, Running
db3                | 10.10.1.243     |  3306 |           0 | Slave, Synced, Running
-------------------+-----------------+-------+-------------+--------------------
```

Figure 17: Screenshot of list of servers in MaxScale

Task 5 - Developing Web Application

Develop a very simple dynamic database-driven web application for 'DATS2410 Students' using PHP and MariaDB and deploy it in the web servers.

The application should provide management of student information (minimally: student id, name, email, study program). Based on the knowledge and skills learnt from the database course you had, design a relational database 'studentinfo' to accommodate the required information. You are free to add more information to make the system as realistic as possible, but it won't add any extra points.

The application should have web pages for a basic CRUD (Create, Retrieve [just list all], Update and Delete) operations on student information from the web. There should be a main index.php page from where one can navigate to other pages. Each web page should have a footer showing the hostname or IP of the web server used at a given time.

Use only standard HTML and PHP (and CSS if you want, but not required) to develop the application. Do not use JavaScript and any third-party frameworks!

Task 5.1

Give an ER diagram of the database design.

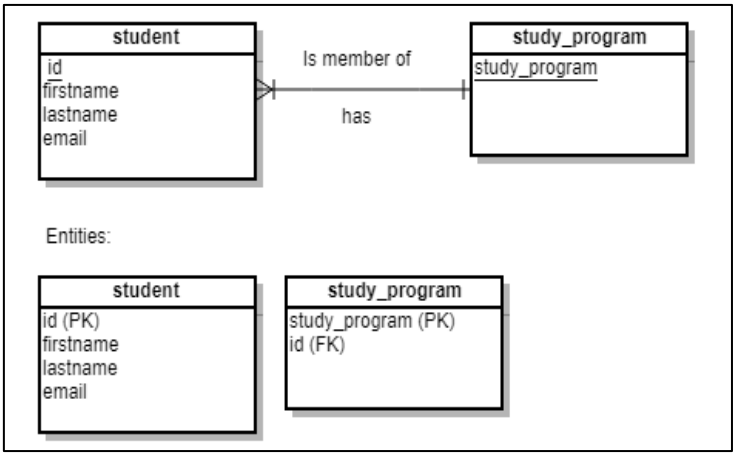


Figure 18: ER Diagram of the database design

**We assume that one student can take only <u>one</u> study program, and that a study program has to have at least one student in order to be made active*

Task 5.2

Show the functionalities of your application by providing relevant screenshots.

Page for registration:

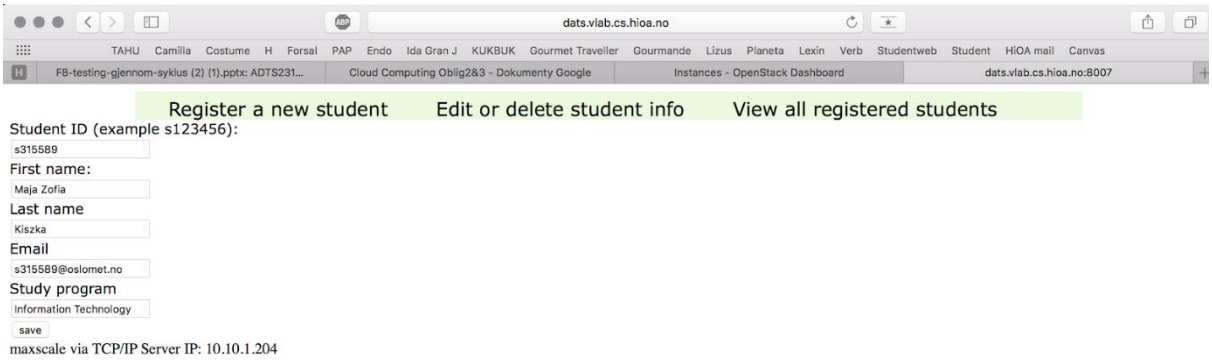


Figure 19:Screenshot of application front page, for registration of students

Page that lists all registered students, and result from the database query showing saved changes

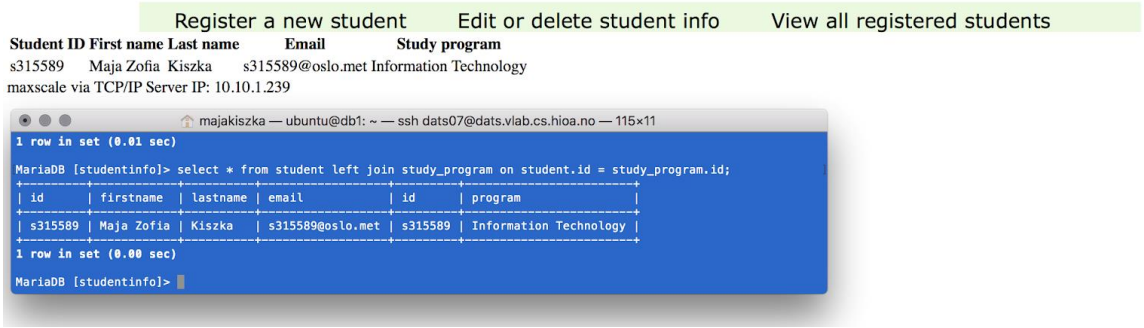


Figure 20: Screenshot of application page that lists registered students, plus database query results

The edit and delete page, and result from database query showing changes in the database

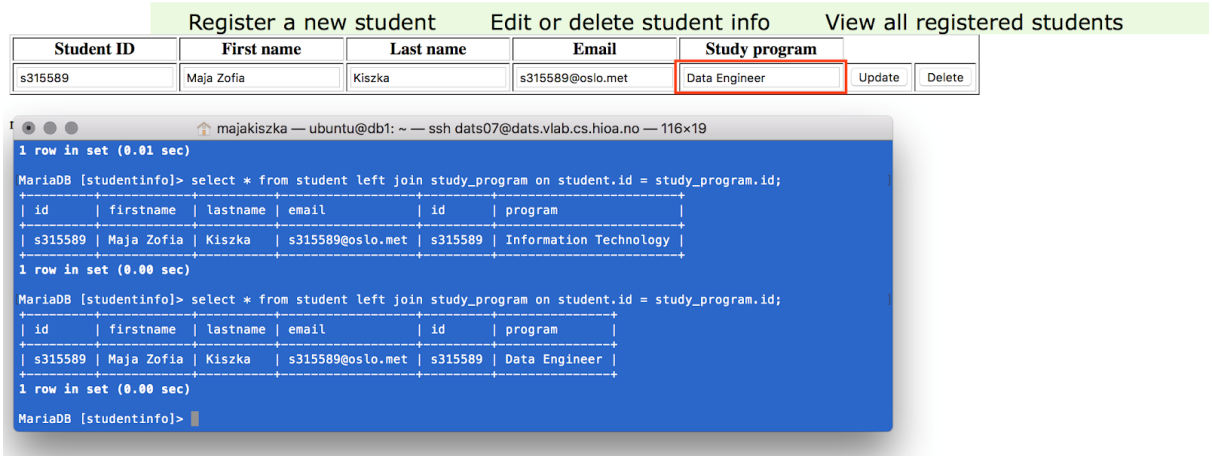


Figure 21:Screenshot of application edit and delete page, plus database query results

Result on the website and the database after pressing delete button

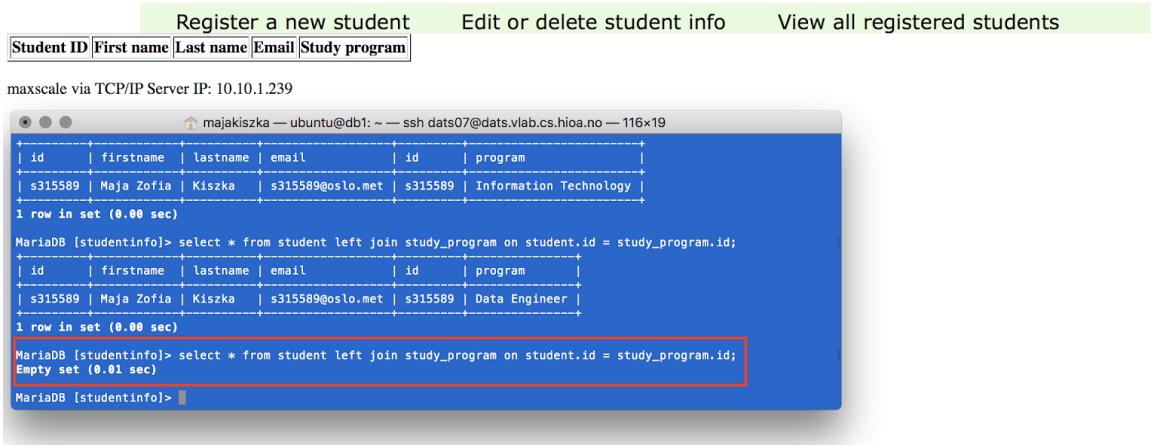


Figure 22:Screenshot of application page after pressing delete button, plus database query results

Task 6 - HAProxy Monitor Setup

Setup HAProxy for monitoring status of the web servers from web address dats.vlab.cs.hioa.no:10XX.

Task 6.1

Provide screenshots of the HAProxy configuration (for monitoring) and resulting monitoring web page.

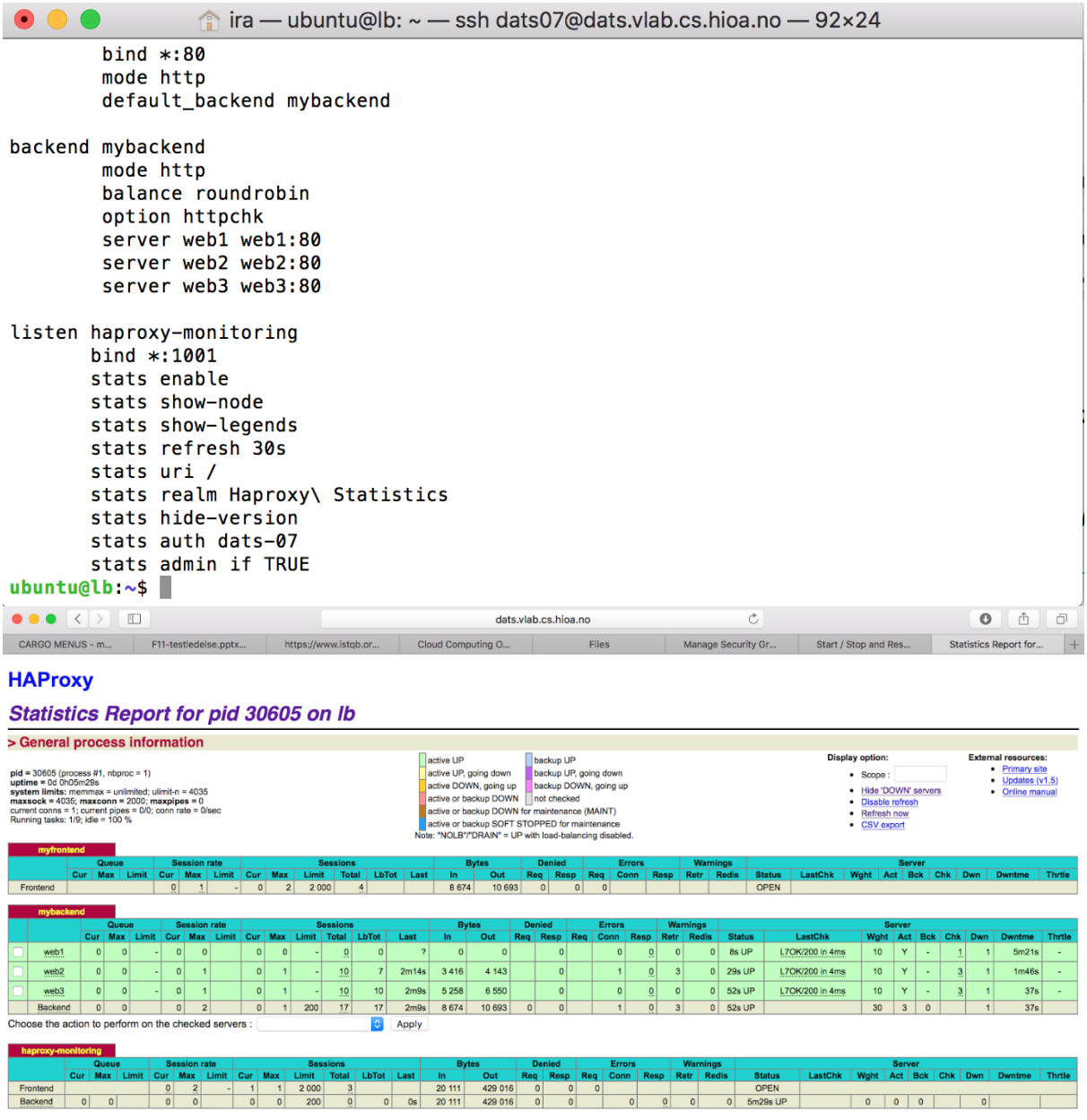


Figure 23: Screenshots of the HAProxy configuration and resulting monitoring web page.

Task 7 - I/O Benchmark Test

Do an active file I/O benchmark test [at least 3 times in each test] for two different machines (say, any two different laptop computers, labelled as Comp1, Comp2) using Bonnie++, and create a HTML graph showing comparative results that can be viewed from the web: dats.vlab.cs.hioa.no:80xx/bout-graph.html.

1. *Provide a table showing comparative results and screenshot(s) of the resulting graph.*
2. *From the comparative results, explain in brief what can you say about the performance of the two machines.*

In this task, we chose to combine subtask 1 and 2 to make it easier for the reader to understand the discussions. This way the discussions regarding each graph is located directly above the graph in question. We performed this test on two of our personal computers. One of them was a Mac (Comp2) and the other a Windows machine(comp1).

- The command that we used on Mac: /usr/local/sbin/bonnie++ -d /tmp -n 5 -m Comp2 -b -x 10 > bout.dat
- The command that we used on Windows PC: bonnie++ -d /tmp -n 5 -m comp1 -b -x 10 > bout.dat

Block IO

The higher values in Block IO test (in kB/sec), the better. It is apparent from the results that Comp2 is performing better. For instance, we see that Comp2 performs three times better than Comp1 with regards to sequential block output.

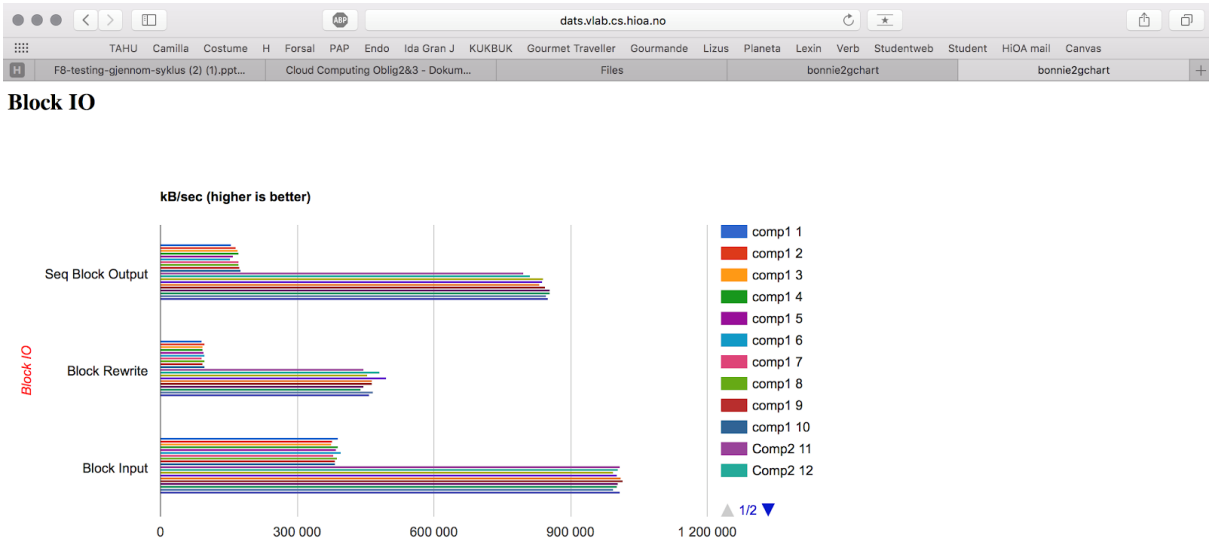


Figure 24:Screenshot of resulting graph from Block IIO test

File Metadata

Here, higher values are better. We see from the graphs that the results are varying slightly in each test, and that Comp2 is actually not represented here. This is because the resulting file shows "+++" indicating that the particular benchmark completes too quickly. Based on that, we can conclude that Comp2 is the best performer regarding file metadata.

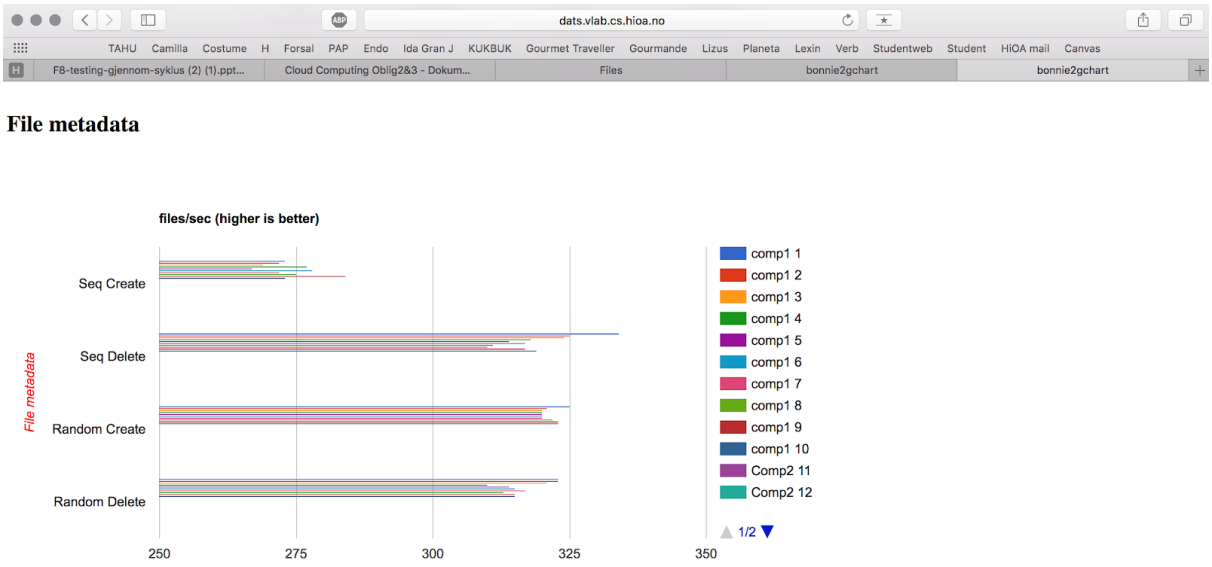


Figure 25: Screenshot of resulting graph from File Metadata test

Block IO CPU

This graph illustrates CPU usage in percent (%). Thus, it is best to have as low values as possible. We see from the graph that the two units get even test results when testing sequential block output CPU, block rewrite CPU and block input CPU. What is clear is that it requires a larger portion of the available CPU than for instance block input. As we can see from the graph the CPU usage is several times higher for both Comp1 and Comp2 in this test. We can further conclude that Comp1 is the winner in the three other test with a much lower CPU usage percentage than Comp2.

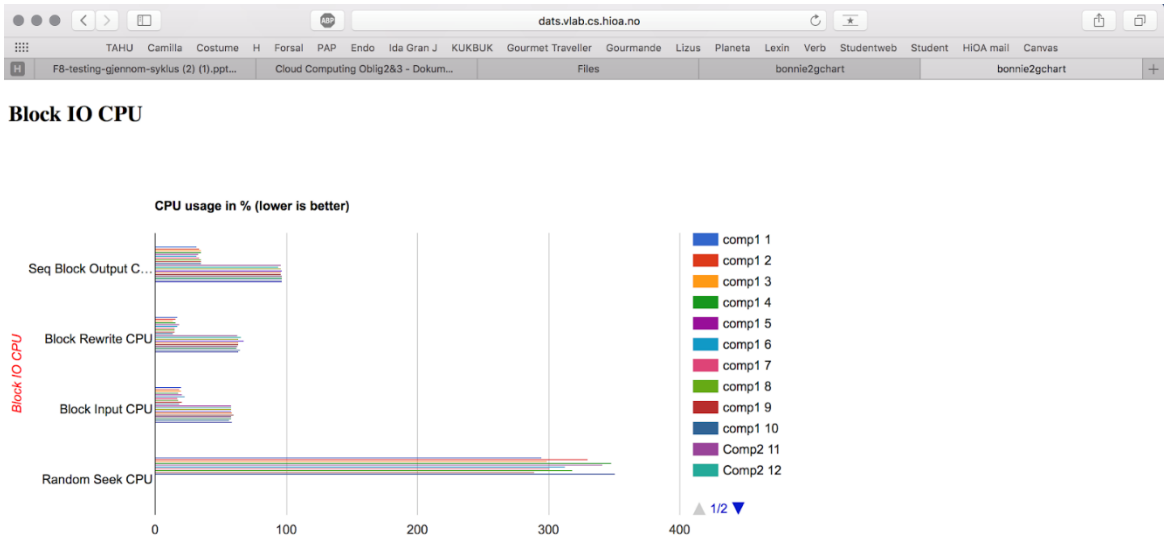


Figure 26: Screenshot of resulting graph from Block IO CPU test

Seq and Random CPU

This graph also illustrates CPU usage in percent (%), and low values are preferable. In this test we also did not get any results for Comp2. The resulting file after running bonnie++ displayed the result using "+++", indicating that the particular benchmark completed too quickly. Comp2 is thus the best performing as it managed to complete the benchmark that quickly.

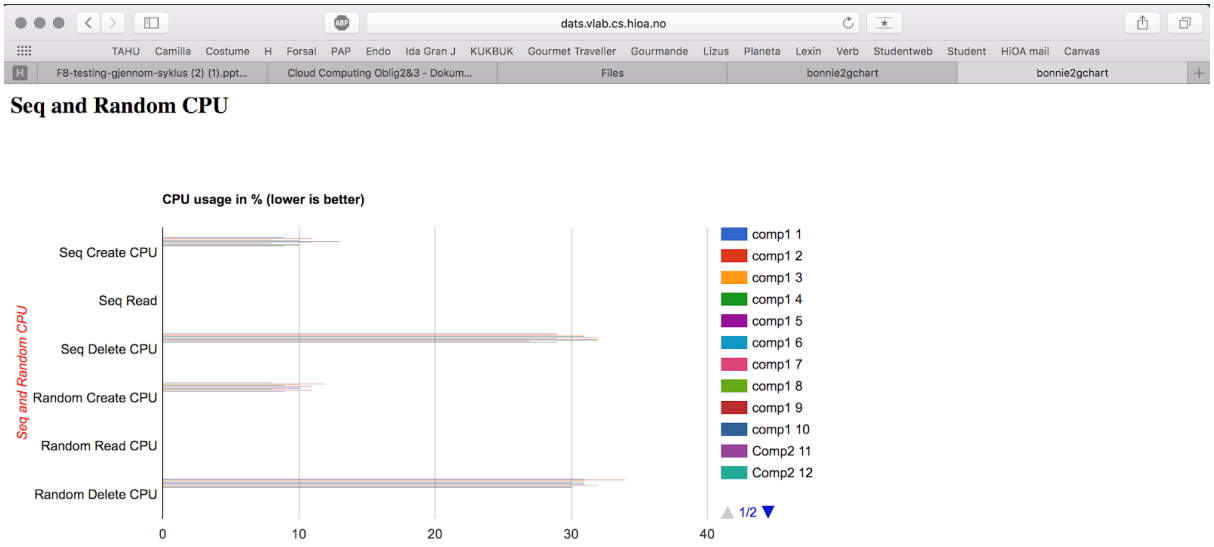


Figure 27:Screenshot of resulting graph from CPU usage test

Block IO Latency

The results of these benchmarks are shown in nanoseconds. Thus, lower values are better than higher ones. It is obvious from the graphs that Comp2 is the best performing. The results from Comp2 are so low that they are barely visible on the graph, while Comp1 spent much more time. Comp2 is particularly superior regarding Latency block output, where Comp1 in the worst case got a result on 6,000,000,000 nanoseconds while Comp2 was only 18,326,000.

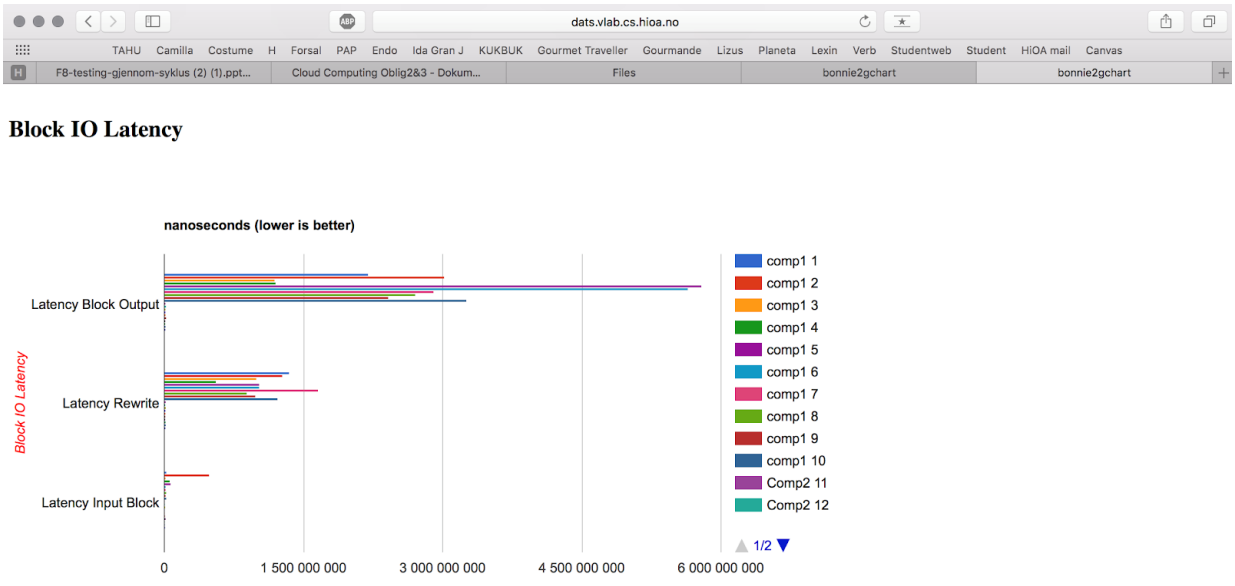


Figure 28: Screenshot of resulting graph from Block IO Latency test

File Metadata Latency

The results of these benchmarks are also shown in nanoseconds. Thus, lower values are better than higher ones. In the categories Latency seq create, latency seq delete and latency random create, Comp2 is the superior one. Noe results are displayed for Comp2 in these categories as it completed the benchmark to fast. in the category latency random delete, the results for Comp2 is listed once. All other 9 tests for Comp2 was completed too fast, and it should therefore be considered the best performing in this category as well.

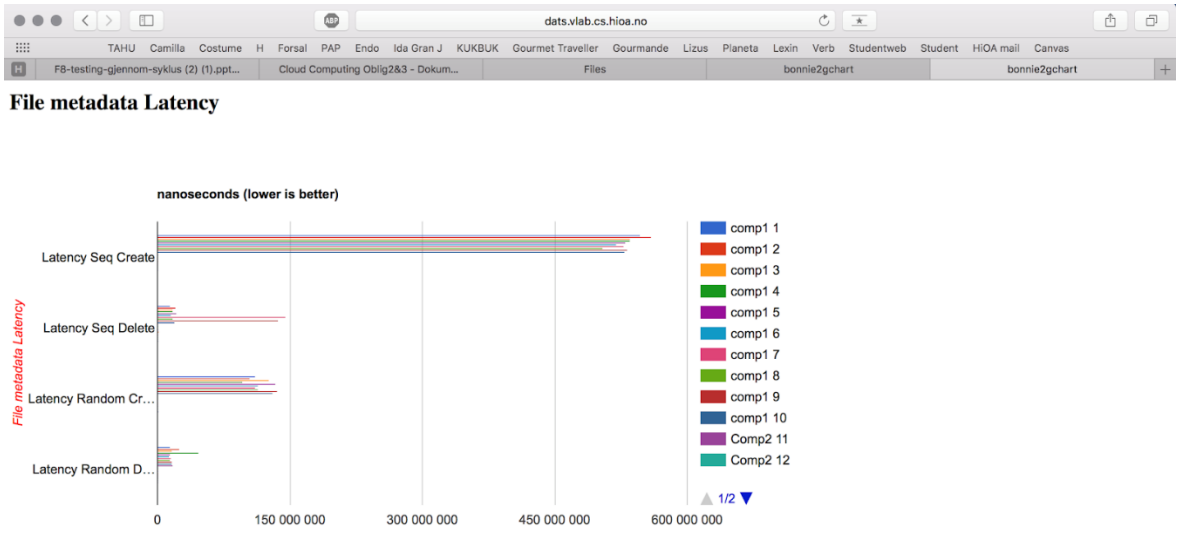


Figure 29: Screenshot of resulting graph from File metadata Latency test

File Metadata (read) Latency

The results of these benchmarks are also shown in nanoseconds. Thus, lower values are better than higher ones. In this test we get results listed for both machines, but it is apparent that Comp2 performs better than Comp1 in that its results are always somewhat lower than for Comp2.

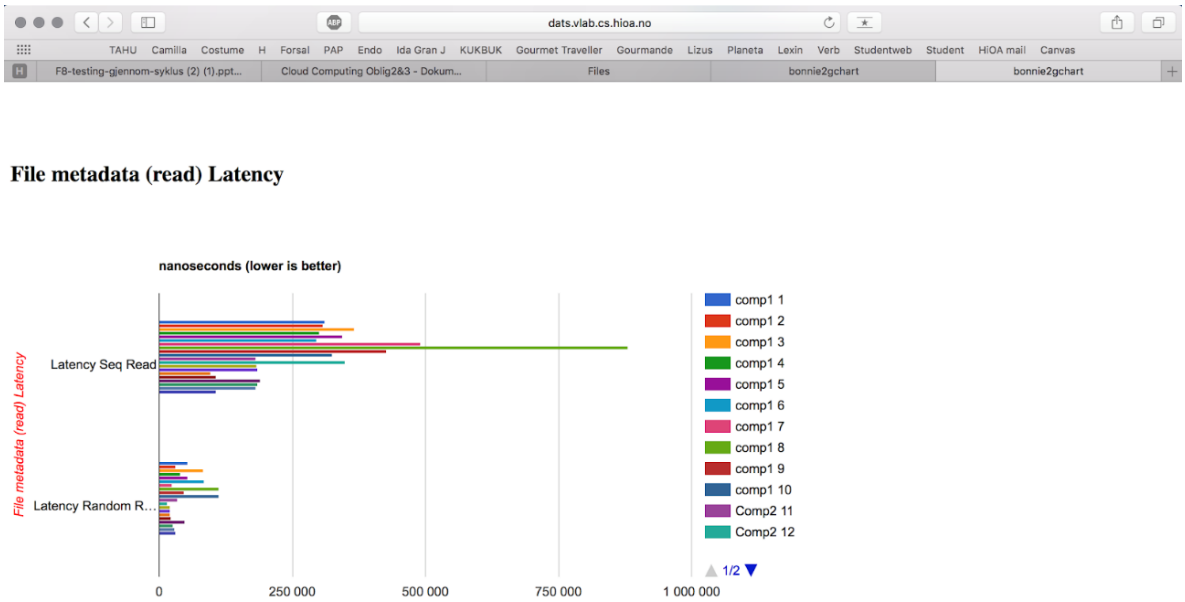
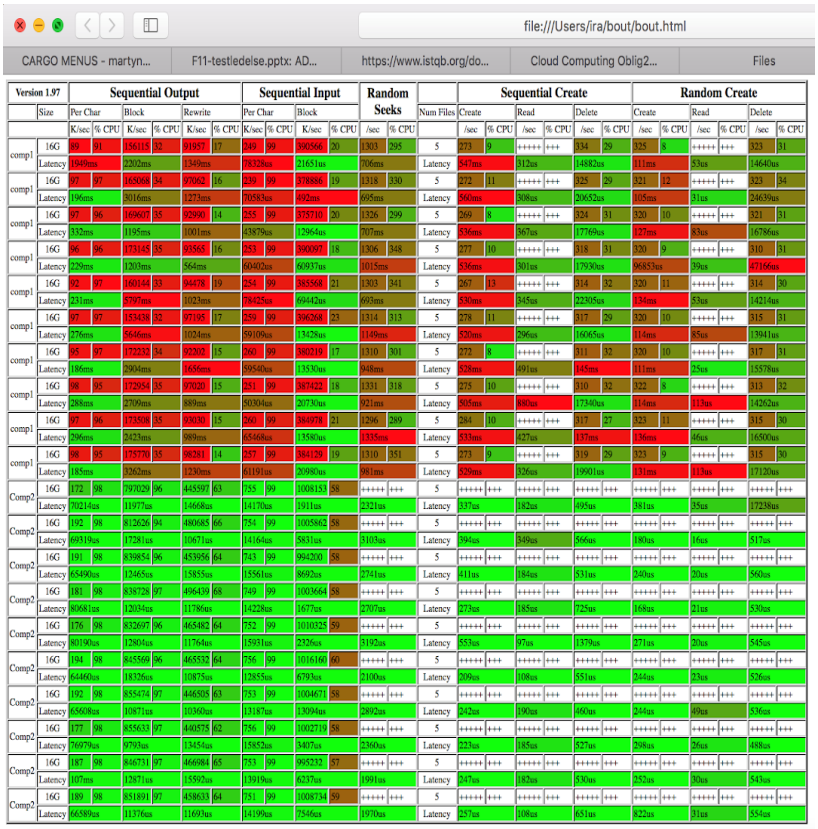


Figure 30: Screenshot of resulting graph from File Metadata (read) Latency test



Version 1.97		Sequential Output						Sequential Input				Random Seeks		Sequential Create						Random Create					
	Size	Per Char		Block		Rewrite		Per Char		Block				Create		Read		Delete		Create		Read		Delete	
		K/sec	% CPU	K/sec	% CPU	K/sec	% CPU	K/sec	% CPU	K/sec	% CPU	/sec	% CPU	Num Files	/sec	% CPU	/sec	% CPU	/sec	% CPU	/sec	% CPU	/sec	% CPU	/sec	% CPU
comp1	16G	88	91	456115	52	91957	17	249	99	990566	20	1303	295	5	273	9	+++++	+++	334	29	325	8	+++++	+++	323	31
	Latency	1949ms		2202ms		1349ms		78328us		21651us		706ms		Latency	547ms		512us		14882us		111ms		83us		14640us	
comp1	16G	97	97	163068	34	97062	16	239	99	578886	19	1318	330	5	272	11	+++++	+++	325	29	321	12	+++++	+++	323	34
	Latency	196ms		3016ms		1273ms		70583us		692ms		695ms		Latency	560ms		808us		20652us		105ms		31us		24639us	
comp1	16G	97	96	169607	35	92990	14	255	99	975710	20	1326	299	5	269	8	+++++	+++	324	31	320	10	+++++	+++	321	31
	Latency	332ms		1195ms		1001ms		43879us		7296us		707ms		Latency	536ms		567us		17769us		127ms		83us		16786us	
comp1	16G	96	96	173145	35	93565	16	253	99	990097	18	1306	368	5	277	10	+++++	+++	318	31	320	9	+++++	+++	310	31
	Latency	229ms		1203ms		564ms		60402us		60937us		1015ms		Latency	536ms		301us		17930us		66853us		39us		47166us	
comp1	16G	92	97	160144	33	94478	19	254	99	985568	21	1303	341	5	267	13	+++++	+++	314	32	320	11	+++++	+++	314	30
	Latency	231ms		3797ms		1023ms		78425us		6942us		693ms		Latency	530ms		845us		22305us		134ms		53us		14214us	
comp1	16G	97	97	153438	32	97195	17	259	99	896268	23	1314	313	5	278	11	+++++	+++	317	29	320	10	+++++	+++	315	31
	Latency	276ms		5616ms		1024ms		59109us		13428us		1149ms		Latency	520ms		296us		16065us		114ms		85us		13941us	
comp1	16G	95	97	172232	34	92202	15	260	99	880219	17	1310	301	5	272	8	+++++	+++	311	32	320	10	+++++	+++	317	31
	Latency	186ms		2004ms		1656ms		69540us		7353us		948ms		Latency	528ms		691us		8459us		111ms		25us		15578us	
comp1	16G	98	95	172954	35	97021	15	251	99	887422	18	1331	318	5	275	10	+++++	+++	310	32	322	8	+++++	+++	313	32
	Latency	268ms		2709ms		889ms		50304us		20730us		921ms		Latency	505ms		880us		17340us		114ms		113us		14262us	
comp1	16G	97	96	173508	35	93030	15	260	99	884978	21	1296	289	5	284	10	+++++	+++	317	27	323	11	+++++	+++	315	30
	Latency	296ms		2423ms		989ms		65468us		13580us		1335ms		Latency	533ms		427us		137ms		136ms		86us		16500us	
comp1	16G	98	95	175770	35	98281	14	257	99	884129	19	1310	351	5	273	9	+++++	+++	319	29	323	9	+++++	+++	315	30
	Latency	185ms		3262ms		1230ms		61191us		20980us		98ms		Latency	529ms		326us		19901us		131ms		113us		17120us	
Comp2	16G	172	98	797029	96	445597	63	755	99	1008153	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	70214us		11977us		14668us		14170us		1911us		2321us		Latency	337us		182us		695us		381us		35us		17238us	
Comp2	16G	192	98	812626	93	480085	66	754	99	1005862	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	69319us		17281us		10671us		14166us		5831us		3103us		Latency	396us		349us		566us		180us		16us		517us	
Comp2	16G	191	98	839854	96	453956	64	743	99	994200	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	65490us		12465us		13855us		15561us		8692us		2741us		Latency	411us		184us		531us		240us		20us		560us	
Comp2	16G	181	98	838728	97	496439	68	749	99	1003664	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	86681us		12034us		11786us		14228us		1677us		2707us		Latency	273us		185us		725us		168us		21us		530us	
Comp2	16G	176	98	832697	96	465482	64	752	99	1010525	59	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	80190us		12804us		11764us		15931us		2326us		3192us		Latency	553us		97us		1379us		271us		20us		545us	
Comp2	16G	194	98	845569	96	465532	64	756	99	1016160	60	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	64460us		18336us		10875us		12855us		6793us		2100us		Latency	209us		108us		551us		344us		23us		526us	
Comp2	16G	192	98	855474	97	466505	63	753	99	1004671	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	65608us		10871us		10360us		13187us		13094us		2892us		Latency	242us		190us		460us		244us		49us		536us	
Comp2	16G	177	98	855633	97	440575	62	756	99	1002719	58	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	76979us		9793us		13454us		15852us		3407us		2360us		Latency	223us		185us		527us		298us		26us		458us	
Comp2	16G	187	98	846731	97	466984	65	753	99	995232	57	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	107ms		12871us		15597us		13919us		6237us		1991us		Latency	247us		182us		530us		252us		30us		543us	
Comp2	16G	189	98	851891	97	458633	64	751	99	1008734	59	+++++	+++	5	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++	+++++	+++
	Latency	66589us		11376us		11693us		14190us		7546us		1970us		Latency	257us		108us		651us		822us		31us		554us	

Figure 31: Screenshot of the resulting graph from I/O Benchmark Test

Task 8 - Questions and Answer

Wherever relevant, provide screenshots supporting your answers in each scenario given in the following task descriptions.

Task 8.1

(i) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when one web server is down? Give screenshots from HAProxy monitoring pages.

When one server is down, for the user nothing changes, because the Load Balancer will know that this particular server is not active, and it will redirect the traffic to two other servers, and by that the website will be displayed correctly.

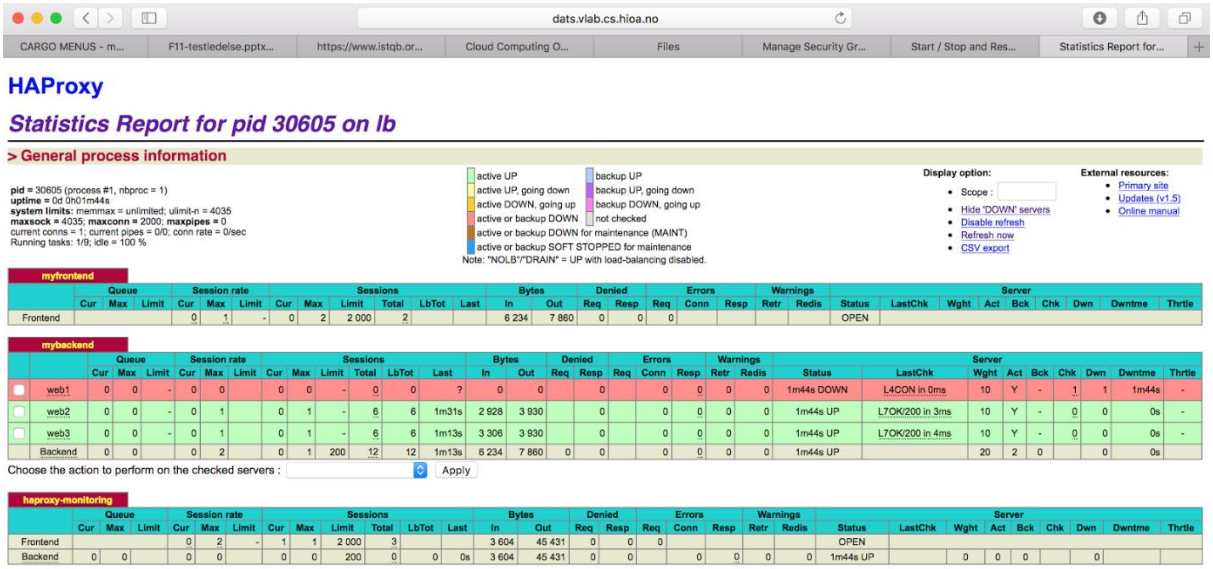


Figure 32: Screenshot of HAProxy monitoring pages when one web server is down

(ii) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when two web servers are down? Give screenshots from HAProxy monitoring pages.

When two out of three servers are not running, all the workload will be now redirected to the last server, and the user will still be able to view the content of the website.

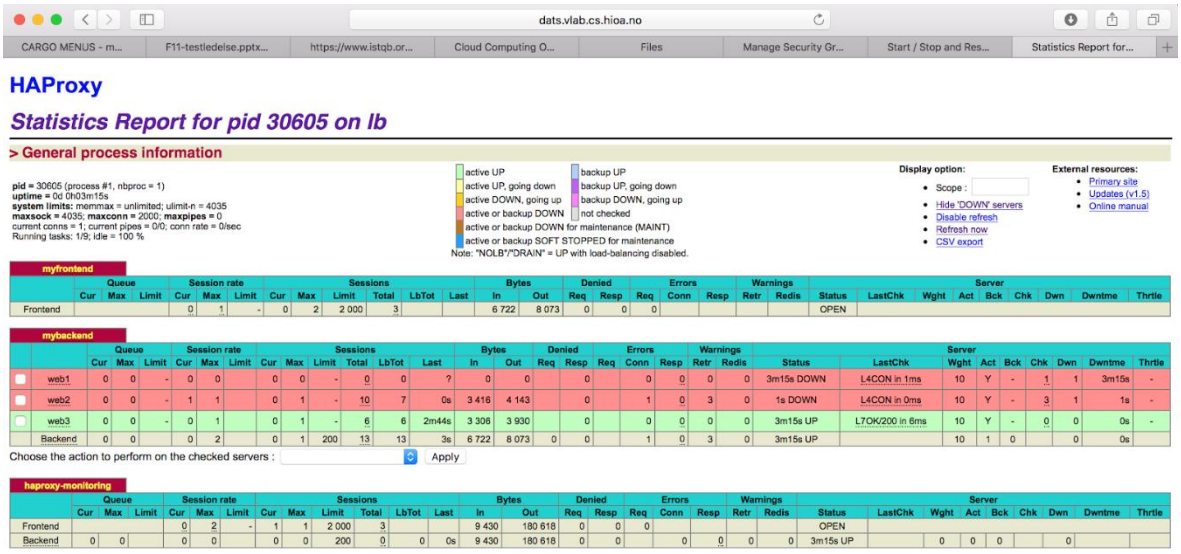


Figure 33: Screenshot of HAProxy monitoring pages when two web servers are down

(iii) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when all the three web servers are down? Give screenshots from HAProxy monitoring pages.

When all three servers are down, the website content will not be displayed at all. User will be prompted with the information in the second picture below.

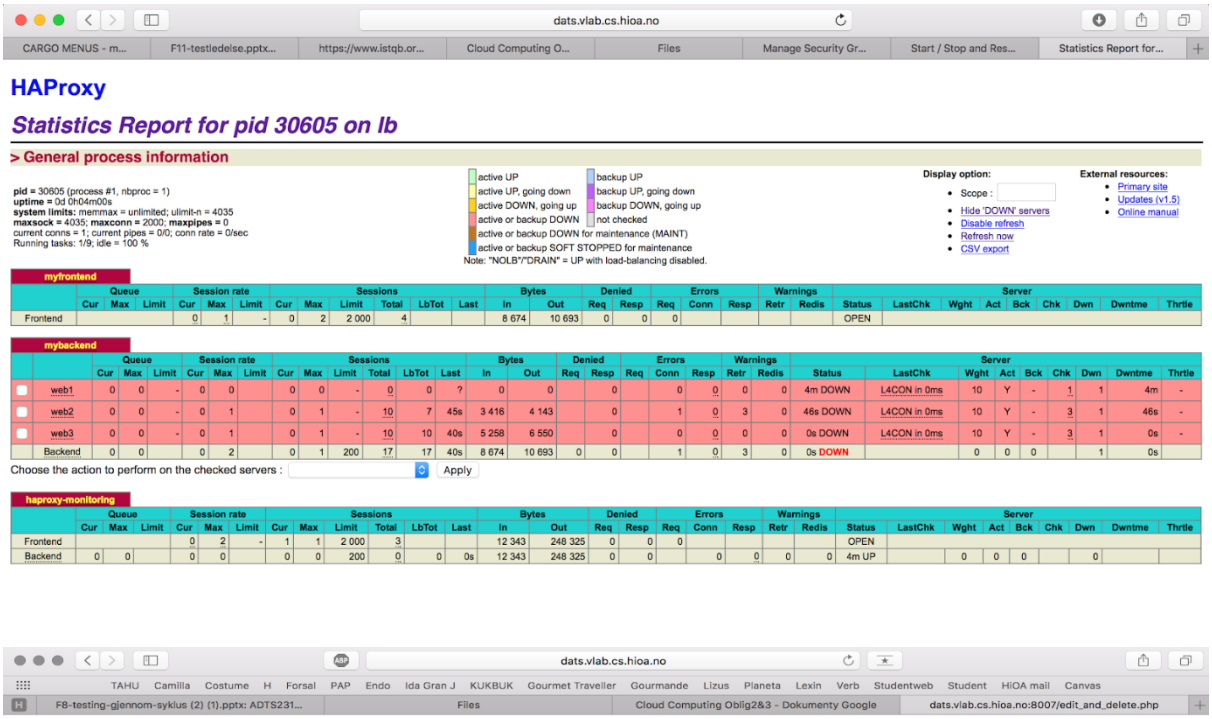


Figure 34: Screenshot of HAProxy monitoring pages when all web servers are down

Task 8.2

(i) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when one database server is down?

When one of the database servers are down, for example db1, one will still be able to use application, because two other database servers still work. But we can see that db2 is master now instead of db1.

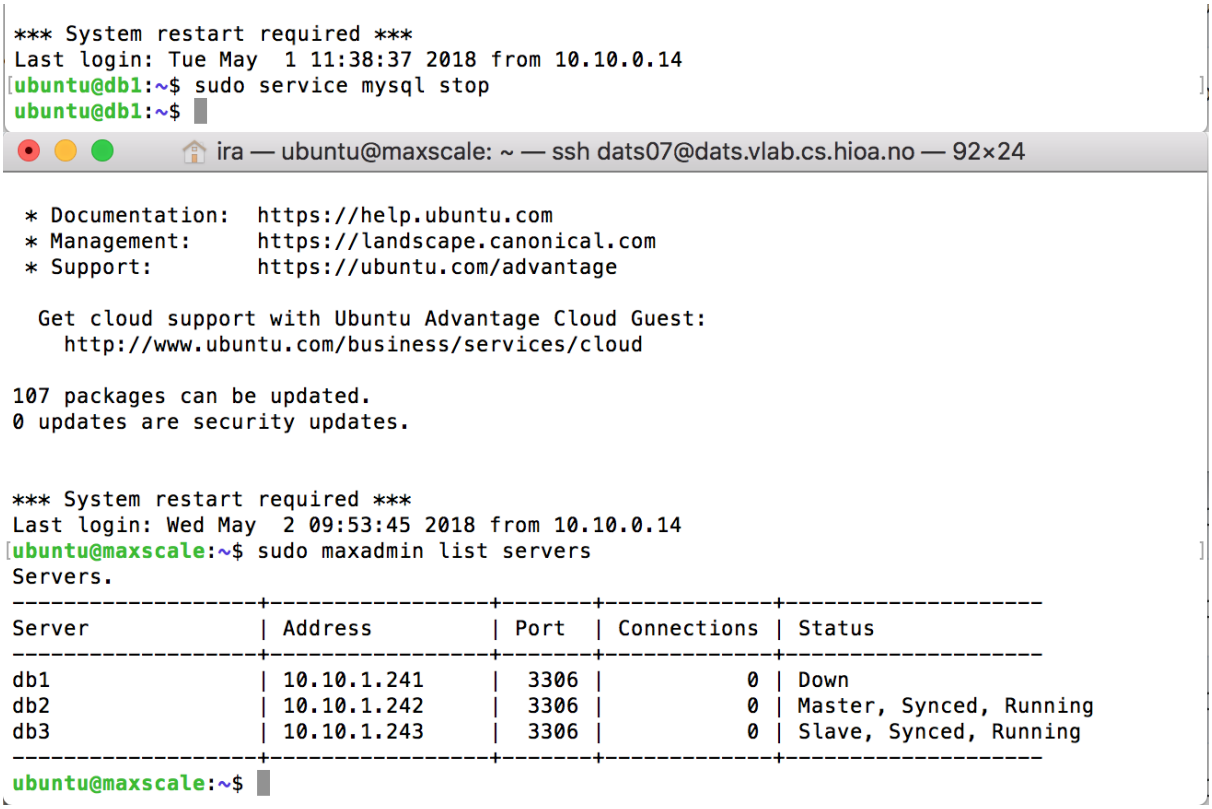
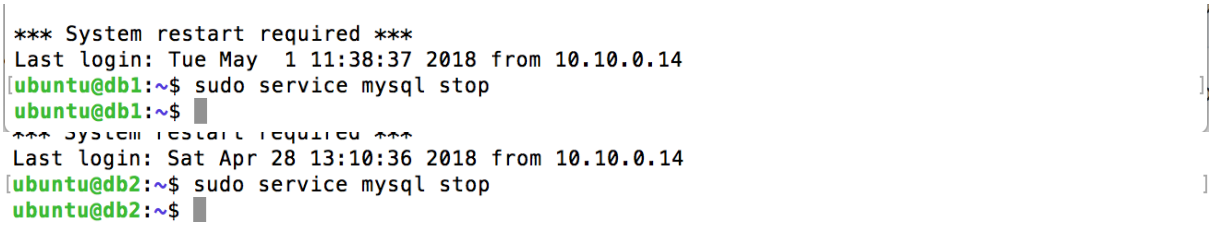


Figure 35: Screenshot of list of servers from maxscale when one database server is down

(ii) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when two database servers are down?

When two database servers are down, one will still be able to use app and retrieve, view and create database queries, because there is still one database server which still works, and it will become master.



```
ubuntu@maxscale:~$ sudo maxadmin list servers
Servers.
-------------------+-----------------+-------+-------------+--------------------
Server             | Address         | Port  | Connections | Status
-------------------+-----------------+-------+-------------+--------------------
db1                | 10.10.1.241     |  3306 |           0 | Down
db2                | 10.10.1.242     |  3306 |           0 | Down
db3                | 10.10.1.243     |  3306 |           0 | Master, Synced, Running
-------------------+-----------------+-------+-------------+--------------------
ubuntu@maxscale:~$
```

Figure 36: Screenshot of list of servers from maxscale when two database servers are down

(iii) What happens (wherever relevant, provide screenshots supporting your answers in each scenario) when all of the database servers are down?

When all of the database servers are down one can still use an app, but there won't be any connection database, so you can't retrieve, create or view information from database.

```
*** System restart required ***
Last login: Tue May  1 11:38:37 2018 from 10.10.0.14
ubuntu@db1:~$ sudo service mysql stop
ubuntu@db1:~$
*** System restart required ***
Last login: Sat Apr 28 13:10:36 2018 from 10.10.0.14
ubuntu@db2:~$ sudo service mysql stop
ubuntu@db2:~$
Last login: Fri Apr 27 09:39:29 2018 from 10.10.0.14
ubuntu@db3:~$ sudo service mysql stop
ubuntu@db3:~$
ubuntu@maxscale:~$ sudo maxadmin list servers
Servers.
-------------------+-----------------+-------+-------------+--------------------
Server             | Address         | Port  | Connections | Status
-------------------+-----------------+-------+-------------+--------------------
db1                | 10.10.1.241     |  3306 |           0 | Down
db2                | 10.10.1.242     |  3306 |           0 | Down
db3                | 10.10.1.243     |  3306 |           0 | Down
-------------------+-----------------+-------+-------------+--------------------
ubuntu@maxscale:~$
```

Figure 37: Screenshot of list of servers from maxscale when all database servers are down

Task 8.3

What happens when the load balancer is crashed? What could be a solution to address the problem?

When Load Balancer crashes, then all the service will stop because it stops distributing the workload.

A good solution to fix that issue would be to have more than one Highly Available Load Balancers that are clustered together. For example, two Load Balancers, where one is always active, and the other one is passive. Then when one of them crashes, the other one (passive) will figure out that the first LB is down, and the second LB will become now active. The connection between the two (or more) clustered Load Balancers occurs very fast, so that whenever one of them is down, the other one picks up that information immediately and starts to take incoming traffic and redirects it accordingly.

Task 9 - Working in a group

Provide following details reflecting how well you worked as a group:

How you worked in your group? [How often did you meet, how tasks were distributed among your group members, whether you managed to make everyone participate and known about all the tasks (not just what s/he did), ...]

The group worked as a team throughout the entire oblig-work. For the first half of the project time we met once a week to work together on the oblig. For the last half of the project time we met several times a week. The different tasks were performed by all the group members together and we made sure that everyone was part of all the tasks. This way we secured that all group members gained knowledge on all parts of the oblig and thus learned about the different topics.

State who did what in terms of concrete tasks and provide contribution of the individual members in percentage (100% as the reference for the one who contributed the most).

All tasks in this oblig were performed by a united group. We used one computer for performing the tasks, sometimes two or three simultaneously if appropriate. When working on one computer, one or two computers were available for googling questions and checking results from the servers etc. We therefore find it most suitable to reference 100 % contribution for all group members on all tasks.

Problems faced while working with your group (if any).

The group did not meet any major difficulties while working in our group.

Task 10 - Self-evaluation

Evaluate your own submission by filling up the table below. It has two parts: your section-wise expected points (out of the given full points in brackets), and comments.Comments should be given point-wise whether you have done what has been asked properly (+), or not done or if any issues/weaknesses (-), and any other comments worth mentioning (*).

Oblig	Task (Full Points)	Expected Points	Comment
2	**1 VM Setup (10)**	9	+We believe VM Setup was performed correctly, as we did thoroughly the Lab prior to this oblig -May have left out some installed software in the table
	2 LB Setup (10)	10	+We believe we set up load balancer as it was required +We believe all required screenshots are included -It always can be some misunderstanding according to which screenshots should be taken
	3 Web Server Setup(15)	15	+We are certain that the Web Servers were set up correctly +We believe all required screenshots are included +Crontab works as expected
	4 Ha DB setup (15)	15	+We believe configurations are correct and tests show that it works properly
3	**5 Web Application (20)**	19	+Based on experience from previous courses like Databases and Web Programming, we are certain that we've fulfilled this exercise correctly -However Database could have been designed to a bigger extent, nevertheless, it fulfills requested criteria
	6 HAProxy Monitor setup (8)	8	+Configured monitoring according lecture notes, everything works as expected, tested it while we're turning web servers off.
	7 I/O Benchmark test (8)	8	+ We performed 10 tests even though the minimum requirement was three. We therefore have a good basis for our comparison

			- Had some small problems because of different OSs. But fixed everything.
	8 Q & A (6)	6	+ It was easy to answer questions, since we could test it in reality, i.e turning web servers and database servers down.
	9 Working in group (3)	3	+Working in the group was pleasant, and without any problems. We've discussed the collaboration between us and we were unanimously satisfied with it
	10 Self-evaluation (2)	2	+We took time to perform this self-evaluation in detail
	Report and submission quality (3)	3	+We believe that report is written in a clear, structured and readable way
	Total Points [group score] (100)	98	All members: 100 % participation