

May 2018

Oblig 2 & 3 Report

DATS2410 Computer Networking and Cloud Computing

Table of Contents

List of Figures	1
Introduction	2
Task 1 - VM Setup	3
Task 1.1	3
Task 2.2	4
Task 1.3	6
Task 1.4	7
Task 2 - Load Balancer (LB) Setup.....	8
Task 2.1	8
Task 2.2	8
Task 3 - Web Server Setup.....	10
Task 3.1	10
Task 3.2	12
Task 4 - HA Database Setup	14
Task 4.1	14
Task 4.2	20
Task 4.3	20
Task 5 - Developing Web Application.....	21
Task 5.1	21
Task 5.2	22
Task 6 - HAProxy Monitor Setup	24
Task 6.1	24
Task 7 - I/O Benchmark Test	25
Task 8 - Questions and Answer.....	30
Task 8.1	30
Task 8.2	32
Task 8.3	33
Task 9 - Working in a group	34
Task 10 - Self-evaluation	35

List of Figures

Figure 1: Architecture diagram of the cloud setup.....	3
Figure 2:Screenshot from ALTO showing Network Topology.....	4
Figure 3: Screenshot from ALTO showing SSH Key	4
Figure 4: Screenshot from ALTO showing security group	5
Figure 5:Screenshot from ALTO showing security group rules	5
Figure 6:Screenshot from ALTO showing the VMs created.....	7
Figure 7:Screenshot of VM names defined in /etc/hosts, here on Load Balancer.....	7
Figure 8: Screenshots from http://dats.vlab.cs.hioa.no:8007/myphp.php	9
Figure 9: Screenshot of ownership configuration of Webroot folder in web1	10
Figure 10: Screenshot of ownership configuration of Webroot folder in web2	11
Figure 11: Screenshot of ownership configuration of Webroot folder in web3	11
Figure 12: Screenshot of nginx configurations	12
Figure 13: Screenshot of crontab configured to push changes from web1 to web2 and web3 every 2 minutes	13
Figure 14: Screenshots of configurations of the 3 database servers in the Galera cluster.....	17
Figure 15: Screenshots of configurations of the MaxScale proxy server.....	19
Figure 16: Screenshot confirming Galera cluster size to 3.....	20
Figure 17: Screenshot of list of servers in MaxScale	20
Figure 18: ER Diagram of the database design	21
Figure 19:Screenshot of application front page, for registration of students	22
Figure 20: Screenshot of application page that lists registered students, plus database query results ..	22
Figure 21:Screenshot of application edit and delete page, plus database query results	22
Figure 22:Screenshot of application page after pressing delete button, plus database query results	23
Figure 23: Screenshots of the HAProxy configuration and resulting monitoring web page.....	24
Figure 24:Screenshot of resulting graph from Block IIO test	25
Figure 25: Screenshot of resulting graph from File Metadata test	26
Figure 26: Screenshot of resulting graph from Block IO CPU test.....	26
Figure 27:Screenshot of resulting graph from CPU usage test	27
Figure 28: Screenshot of resulting graph from Block IO Latency test.....	27
Figure 29: Screenshot of resulting graph from File metadata Latency test	28
Figure 30: Screenshot of resulting graph from File Metadata (read) Latency test	28
Figure 31: Screenshot of the resulting graph from I/O Benchmark Test	29
Figure 32: Screenshot of HAProxy monitoring pages when one web server is down	30
Figure 33: Screenshot of HAProxy monitoring pages when two web servers are down	31
Figure 34: Screenshot of HAProxy monitoring pages when all web servers are down	31
Figure 35: Screenshot of list of servers from maxscale when one database server is down	32
Figure 36: Screenshot of list of servers from maxscale when two database servers are down	33
Figure 37: Screenshot of list of servers from maxscale when all database servers are down	33

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.

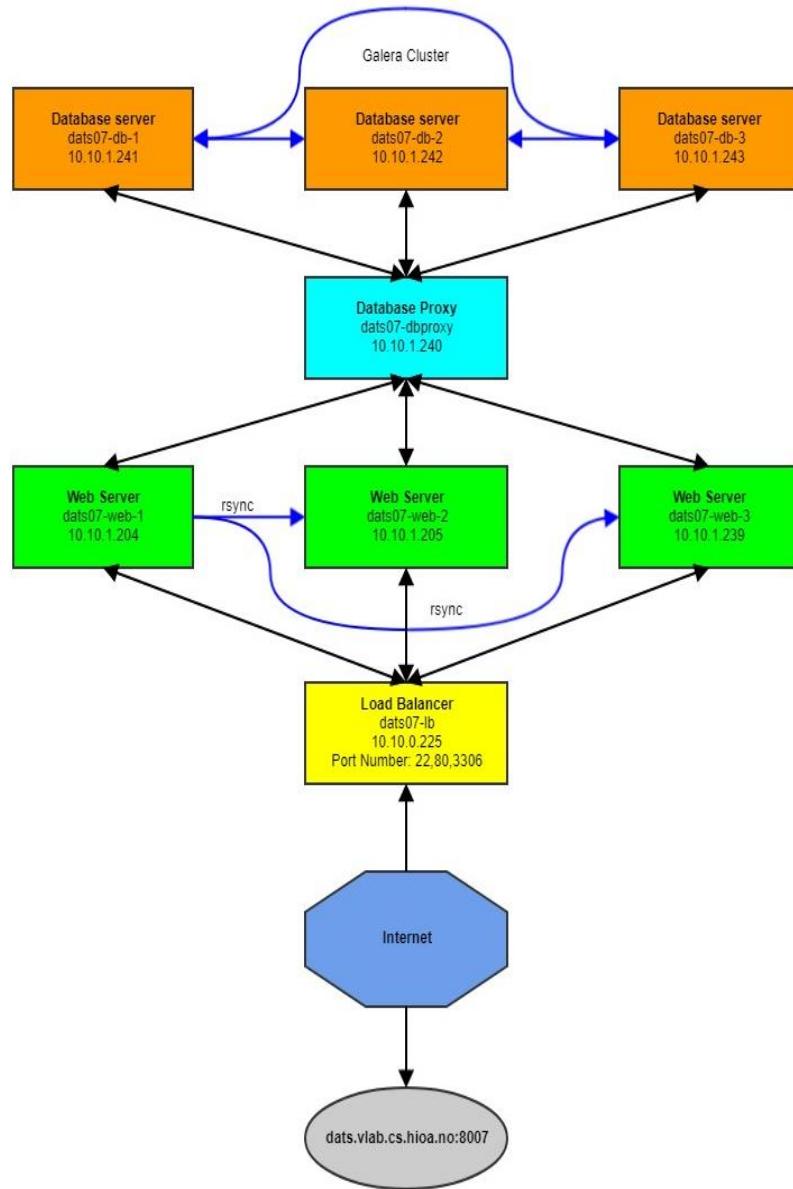


Figure 1: Architecture diagram of the cloud setup

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.

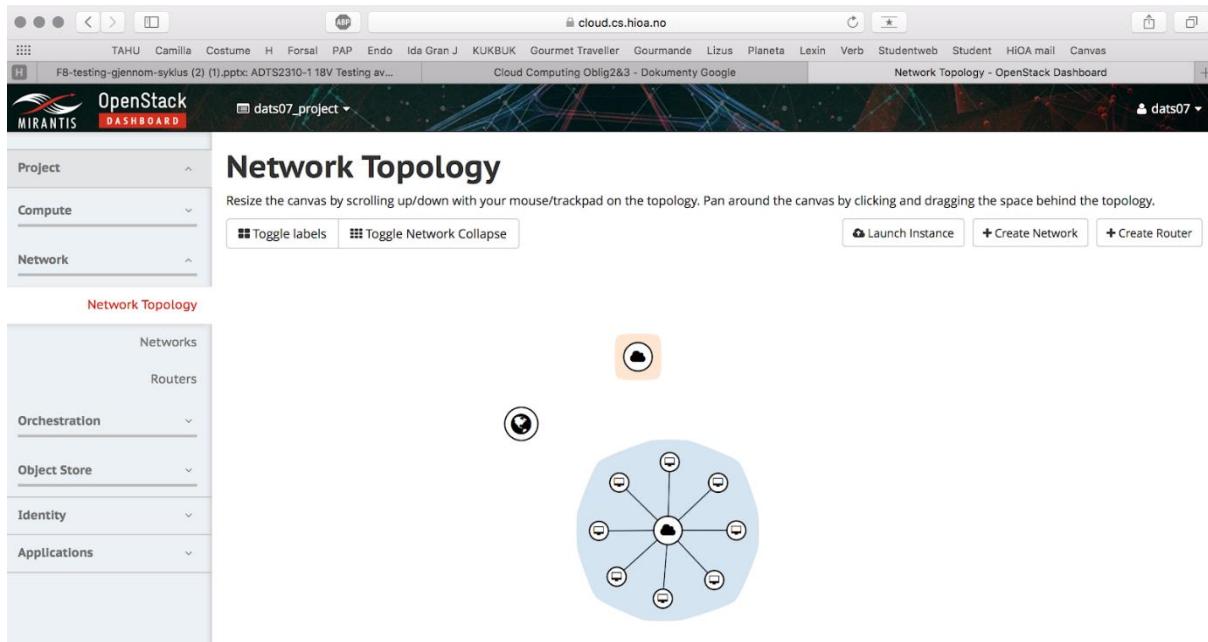


Figure 2: Screenshot from ALTO showing Network Topology

The screenshot shows the 'Access & Security' section of the ALTO dashboard. On the left, a sidebar lists categories: Project, Compute, and Network. Under Network, 'Access & Security' is selected. The main area displays a table of SSH keys. The columns are 'KEY PAIR NAME', 'FINGERPRINT', and 'ACTIONS'. Two entries are listed:

KEY PAIR NAME	FINGERPRINT	ACTIONS
dats-07	cf:38:dd:ec:e8:26:b1:79:0d:ff:55:05:31:2b:09:d2	Delete Key Pair
dats07	ae:ea:a2:b6:ef:03:1e:aa:55:e2:71:eb:49:e6:b7:4d	Delete Key Pair

Below the table, it says 'Displaying 2 items'.

Figure 3: Screenshot from ALTO showing SSH Key

NAME	DESCRIPTION	ACTIONS
dats07-security		Manage Rules
default	Default security group	Manage Rules

Figure 4: Screenshot from ALTO showing security group

DIRECTION	ETHER TYPE	IP PROTOCOL	PORT RANGE	REMOTE IP PREFIX	REMOTE SECURITY GROUP	ACTIONS
Egress	IPv6	Any	Any	::/0	-	<button>Delete Rule</button>
Egress	IPv4	Any	Any	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	ICMP	Any	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	80 (HTTP)	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	1001	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	3306 (MYSQL)	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	4444	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	4567	0.0.0.0/0	-	<button>Delete Rule</button>
Ingress	IPv4	TCP	4568	0.0.0.0/0	-	<button>Delete Rule</button>

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)

INSTANCE NAME	IMAGE NAME	IP ADDRESS	SIZE	KEY PAIR	STATUS	AVAILABILITY ZONE	TASK	POWER STATE	TIME SINCE CREATED	ACTIONS	
dats07-db-3	Ubuntu16.04	10.10.1.243	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 5 days	<button>Create Snapshot</button>
dats07-db-2	Ubuntu16.04	10.10.1.242	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 5 days	<button>Create Snapshot</button>
dats07-db-1	Ubuntu16.04	10.10.1.241	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 5 days	<button>Create Snapshot</button>
dats07-dbproxy	Ubuntu16.04	10.10.1.240	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 5 days	<button>Create Snapshot</button>
dats07-web-3	Ubuntu16.04	10.10.1.239	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 5 days	<button>Create Snapshot</button>
dats07-web-2	Ubuntu16.04	10.10.1.205	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 6 days	<button>Create Snapshot</button>
dats07-web-1	Ubuntu16.04	10.10.1.204	m1.512MB	dats07	Active	nova		None	Running	2 weeks, 6 days	<button>Create Snapshot</button>
dats07-lb	Ubuntu16.04	10.10.0.225	m1.1GB	dats07	Active	nova		None	Running	1 month, 2 weeks	<button>Create Snapshot</button>

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 — 80x24
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:~$ 

```

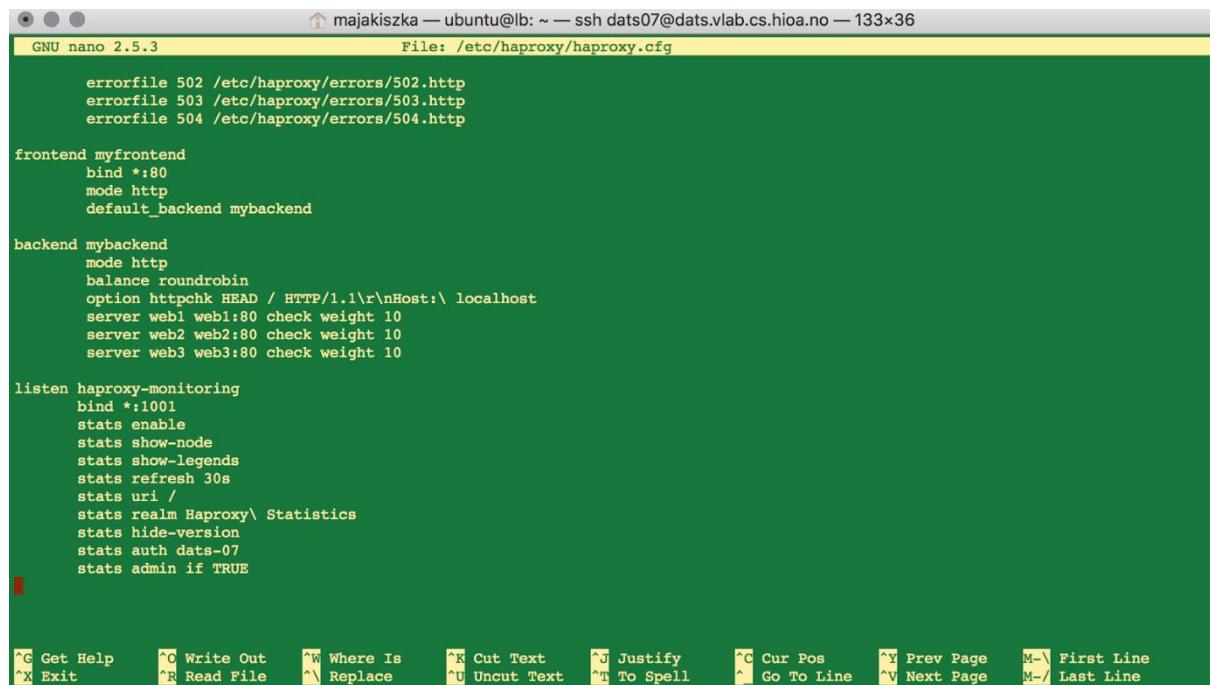
Figure 7: Screenshot of VM names defined in /etc/hosts, here on Load Balancer

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.



The screenshot shows a terminal window with the nano text editor open. The title bar indicates the session is on a 'ubuntu' machine with a connection to 'dats07@datalab.cs.hioa.no'. The file being edited is '/etc/haproxy/haproxy.cfg'. The configuration file contains the following content:

```
errorfile 502 /etc/haproxy/errors/502.http
errorfile 503 /etc/haproxy/errors/503.http
errorfile 504 /etc/haproxy/errors/504.http

frontend myfrontend
    bind *:80
    mode http
    default_backend mybackend

backend mybackend
    mode http
    balance roundrobin
    option httpchk HEAD / HTTP/1.1\r\nHost:\ localhost
    server web1 web1:80 check weight 10
    server web2 web2:80 check weight 10
    server web3 web3:80 check weight 10

listen haproxy-monitoring
    bind *:1001
    stats enable
    stats show-node
    stats show-legends
    stats refresh 30s
    stats uri /
    stats realm Haproxy\ Statistics
    stats hide-version
    stats auth dats-07
    stats admin if TRUE
```

At the bottom of the terminal window, there is a menu bar with various keyboard shortcuts for nano editor commands.

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.

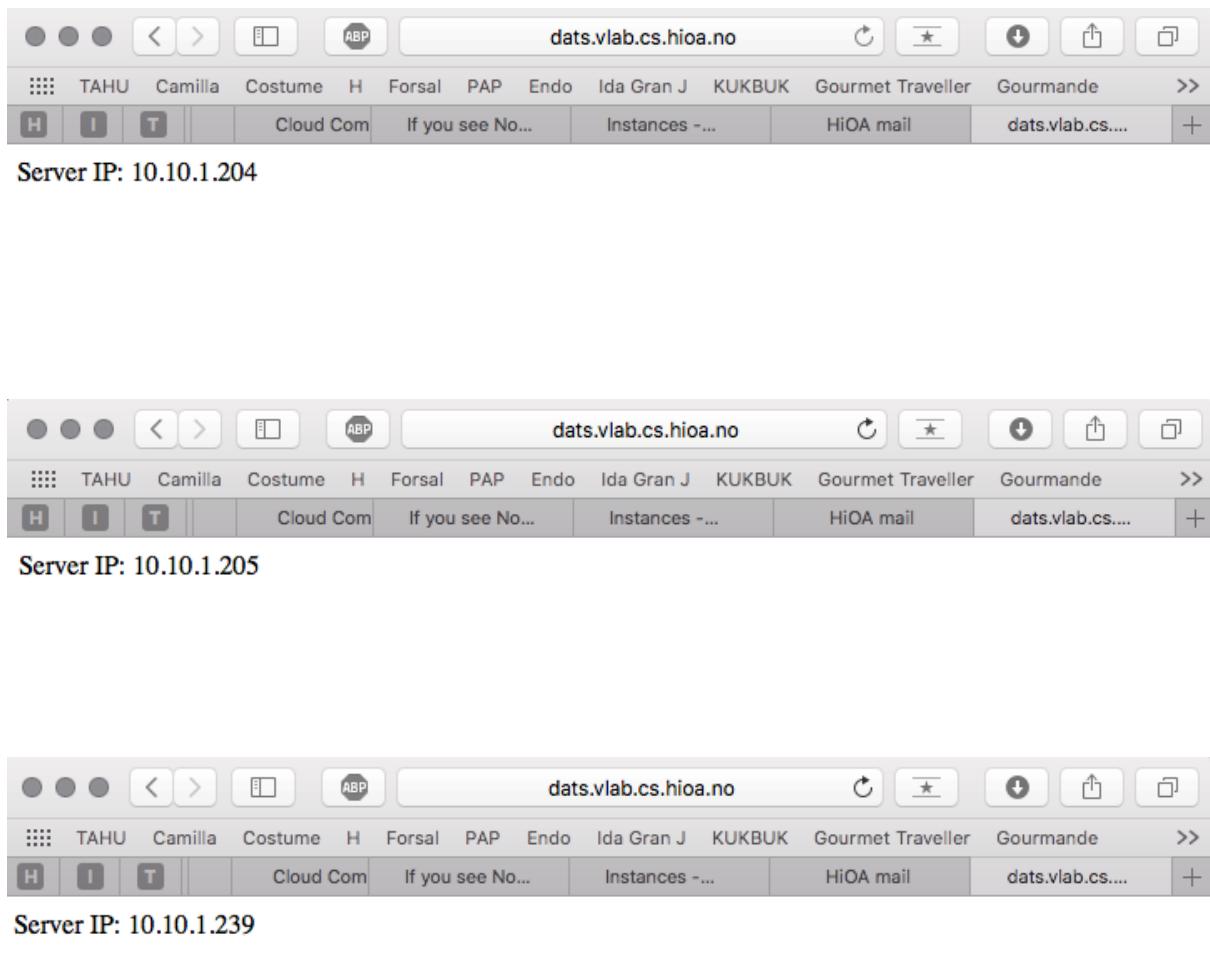


Figure 8: Screenshots from <http://dats.vlab.cs.hioa.no:8007/myphp.php>

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 — 80x24
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 [REDACTED]
[ubuntu@web1:~$ groups
ubuntu adm dialout cdrom floppy sudo audio dip [REDACTED] www-data video plugdev netdev lxd
ubuntu@web1:~$ [REDACTED]

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

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

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 html
ubuntu@web3:~$ ]
```

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

```

ubuntu@web1: ~
GNU nano 2.5.3          File: /etc/nginx/sites-available/default

#
# Read up on ssl_ciphers to ensure a secure configuration.
# See: https://bugs.debian.org/765782
#
# Self signed certs generated by the ssl-cert package
# Don't use them in a production server!
#
# include snippets/snakeoil.conf;

root /var/www/html;

# Add index.php to the list if you are using PHP
index index.php index.html index.nginx-debian.html;

server_name 10.10.1.204;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
#
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    #
    # With php7.0-cgi alone:
    # fastcgi_pass 127.0.0.1:9000;
    # With php7.0-fpm:
    fastcgi_pass unix:/run/php/php7.0-fpm.sock;
}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#

```

GN Get Help **W** Write Out **W** Where Is **C** Cut Text **J** Justify **C** Cur Pos **P** Prev Page **F** First Line
E Exit **R** Read File **R** Replace **U** Uncut Text **T** To Spell **G** Go To Line **N** Next Page **L** Last Line

```

ubuntu@web1: ~
GNU nano 2.5.3          File: /etc/nginx/sites-available/default

        # as directory, then fall back to displaying a 404.
        try_files $uri $uri/ =404;

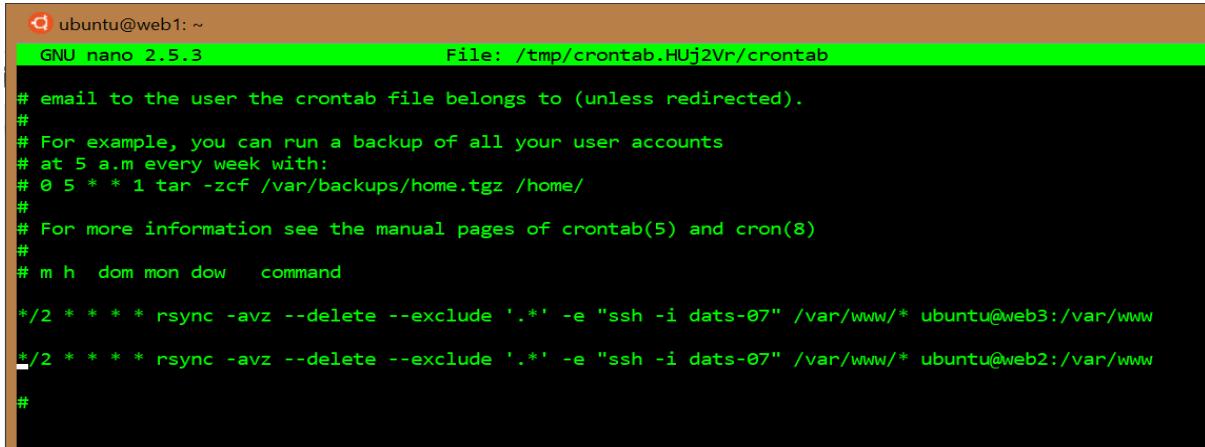
# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
#
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
    #
    # With php7.0-cgi alone:
    # fastcgi_pass 127.0.0.1:9000;
    # With php7.0-fpm:
    fastcgi_pass unix:/run/php/php7.0-fpm.sock;
}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
location ~ /\.ht {
    deny all;
}
}
```

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.



The screenshot shows a terminal window titled "GNU nano 2.5.3" with the file path "File: /tmp/crontab.HUj2Vr/crontab". The content of the crontab file is as follows:

```
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/2 * * * * rsync -avz --delete --exclude '.*' -e "ssh -i dats-07" /var/www/* ubuntu@web3:/var/www
*/2 * * * * rsync -avz --delete --exclude '.*' -e "ssh -i dats-07" /var/www/* ubuntu@web2:/var/www
#
```

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 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 global defaults
#   The files must end with '.cnf', otherwise they'll be ignored
#
!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.

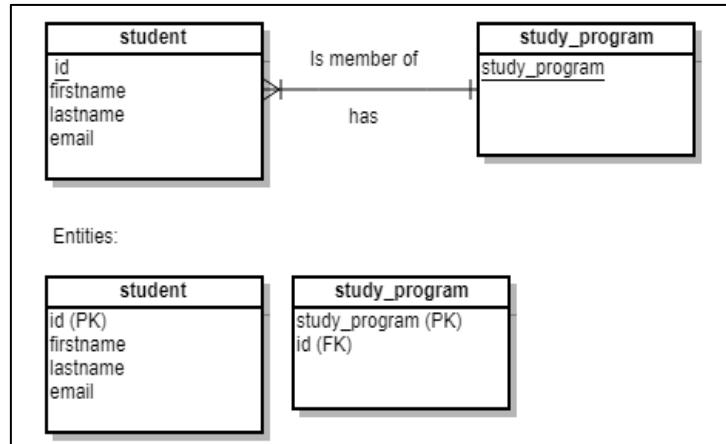


Figure 18: ER Diagram of the database design

*We assume that one student can take only one 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:

The screenshot shows a web browser window titled 'dats.vlab.cs.hioa.no'. The address bar shows 'FB-testing-gjennom-syklus (2) (1).pptx: ADTS231... Cloud Computing Oblig2&3 - Dokumenty Google Instances - OpenStack Dashboard dats.vlab.cs.hioa.no:8007'. The main content area has a green header bar with three buttons: 'Register a new student', 'Edit or delete student info', and 'View all registered students'. Below the header, there is a form for entering student information. The fields are: 'Student ID (example s123456):' with value 's315589', 'First name:' with value 'Maja Zofia', 'Last name:' with value 'Kiszka', 'Email:' with value 's315589@oslo.met', and 'Study program:' with value 'Information Technology'. There is also a 'save' button and a note 'maxscale via TCP/IP Server IP: 10.10.1.204'.

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

The screenshot shows a web browser window with a green header bar. The header buttons are 'Register a new student', 'Edit or delete student info', and 'View all registered students'. Below the header, there is a table showing registered students: 'Student ID First name Last name Email Study program'. One row is shown: 's315589 Maja Zofia Kiszka s315589@oslo.met Information Technology'. Below the table, there is a terminal window with a blue background showing a MySQL database query. The query is: 'MariaDB [studentinfo]> select * from student left join study_program on student.id = study_program.id;'. The result is a table with columns 'id', 'firstname', 'lastname', 'email', 'id', and 'program'. One row is shown: 's315589 | Maja Zofia | Kiszka | s315589@oslo.met | s315589 | Information Technology'. The terminal shows '1 row in set (0.01 sec)' and '1 row in set (0.00 sec)'.

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

The screenshot shows a web browser window with a green header bar. The header buttons are 'Register a new student', 'Edit or delete student info', and 'View all registered students'. Below the header, there is a table for editing student information. The columns are 'Student ID', 'First name', 'Last name', 'Email', and 'Study program'. The 'Study program' field contains 'Data Engineer' and is highlighted with a red border. Below the table, there is a terminal window with a blue background showing a MySQL database query. The query is: 'MariaDB [studentinfo]> select * from student left join study_program on student.id = study_program.id;'. The result is a table with columns 'id', 'firstname', 'lastname', 'email', 'id', and 'program'. One row is shown: 's315589 | Maja Zofia | Kiszka | s315589@oslo.met | s315589 | Information Technology'. The terminal shows '1 row in set (0.01 sec)' and '1 row in set (0.00 sec)'.

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

Result on the website and the database after pressing delete button

The screenshot shows a web application interface with three main tabs at the top: "Register a new student", "Edit or delete student info", and "View all registered students". Below the tabs, there is a form with fields: "Student ID" (s315589), "First name" (Maja), "Last name" (Zofia), "Email" (s315589@oslo.met), and "Study program" (Information Technology). A message below the form says "maxscale via TCP/IP Server IP: 10.10.1.239". To the right of the form is a terminal window titled "majakiszka — ubuntu@db1: ~ — ssh dats07@dat.s.vlab.cs.hioa.no — 116x19". The terminal displays the following SQL queries and their results:

```
mysql> select * from student left join study_program on student.id = study_program.id;
+----+-----+-----+-----+-----+
| id | firstname | lastname | email | program |
+----+-----+-----+-----+-----+
| s315589 | Maja | Zofia | Kiszka | s315589@oslo.met | Information Technology |
+----+-----+-----+-----+-----+
1 row in set (0.00 sec)

MariaDB [studentinfo]> select * from student left join study_program on student.id = study_program.id;
+----+-----+-----+-----+-----+
| id | firstname | lastname | email | program |
+----+-----+-----+-----+-----+
| s315589 | Maja | Zofia | Kiszka | s315589@oslo.met | s315589 | Data Engineer |
+----+-----+-----+-----+-----+
1 row in set (0.00 sec)

MariaDB [studentinfo]> select * from student left join study_program on student.id = study_program.id;
Empty set (0.01 sec)
MariaDB [studentinfo]>
```

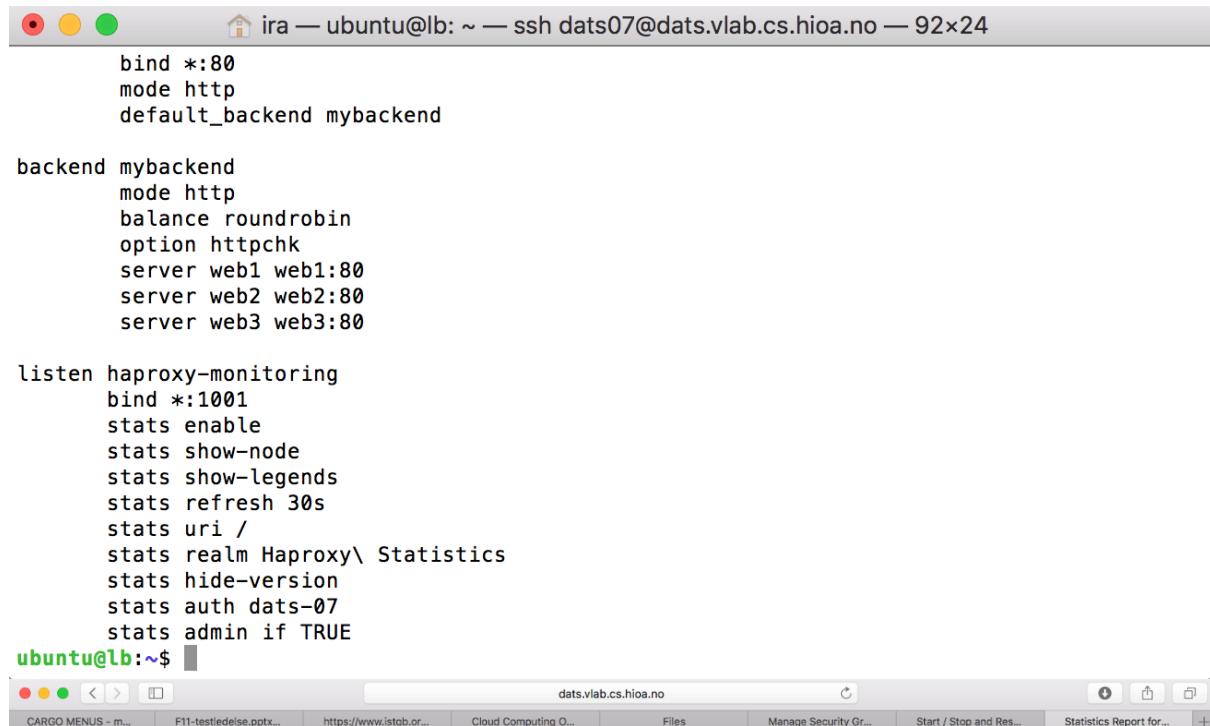
Figure 22: Screenshot of application page after pressing delete button, plus database query results

Task 6 - HAProxy Monitor Setup

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

Task 6.1

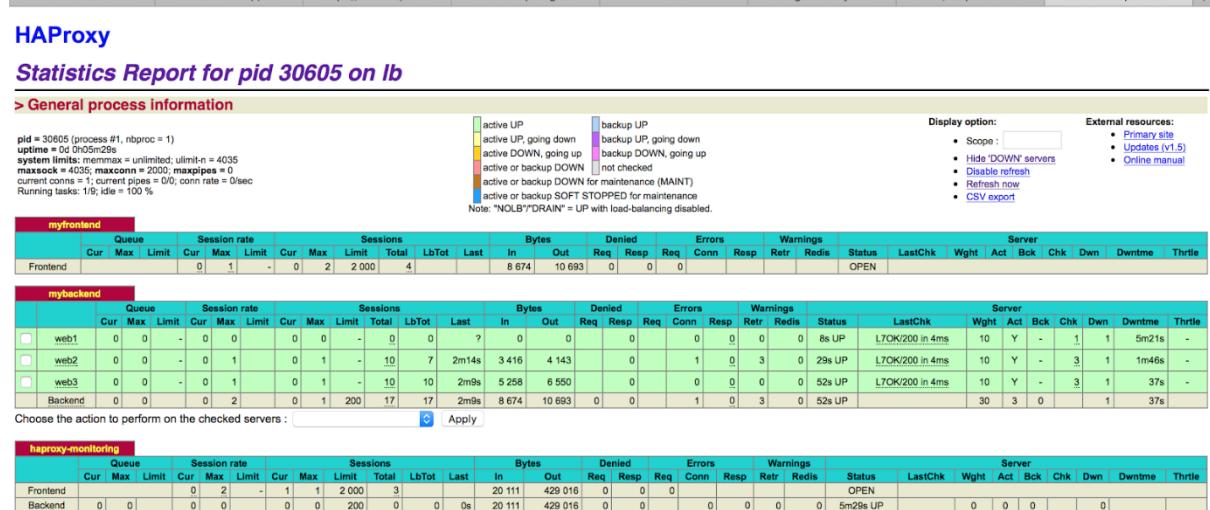
Provide screenshots of the HAProxy configuration (for monitoring) and resulting monitoring web page.



```
ira — ubuntu@lb: ~ — ssh dat.vlab.cs.hioa.no — 92x24
bind *:80
mode http
default_backend mybackend

backend mybackend
    mode http
    balance roundrobin
    option httpchk
    server web1 web1:80
    server web2 web2:80
    server web3 web3:80

listen haproxy-monitoring
    bind *:1001
    stats enable
    stats show-node
    stats show-legends
    stats refresh 30s
    stats uri /
    stats realm Haproxy\ Statistics
    stats hide-version
    stats auth dat.vlab.cs.hioa.no:07
    stats admin if TRUE
ubuntu@lb:~$
```



The screenshot shows the HAProxy Statistics Report for pid 30605 on lb. It includes a legend for server status, a general process information table, and three detailed session tables for myfrontend, mybackend, and haproxy-monitoring.

myfrontend												
	Queue			Session rate			Sessions			Bytes		
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last
Frontend	0	1	-	0	2	2 000	4	8 674	10 693	0	0	0

mybackend												
	Queue			Session rate			Sessions			Bytes		
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last
web1	0	0	-	0	0	0	0	0	-	0	0	7
web2	0	0	-	0	1	0	1	10	7	2m14s	3 416	4 143
web3	0	0	-	0	1	0	1	10	10	2m9s	5 258	6 550
Backend	0	0	0	2	0	1	200	17	17	2m9s	8 674	10 693

haproxy-monitoring												
	Queue			Session rate			Sessions			Bytes		
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last
Frontend	0	2	-	1	1	2 000	3	20 111	429 016	0	0	0
Backend	0	0	0	0	0	200	0	0	0s	20 111	429 016	0

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: dat.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.

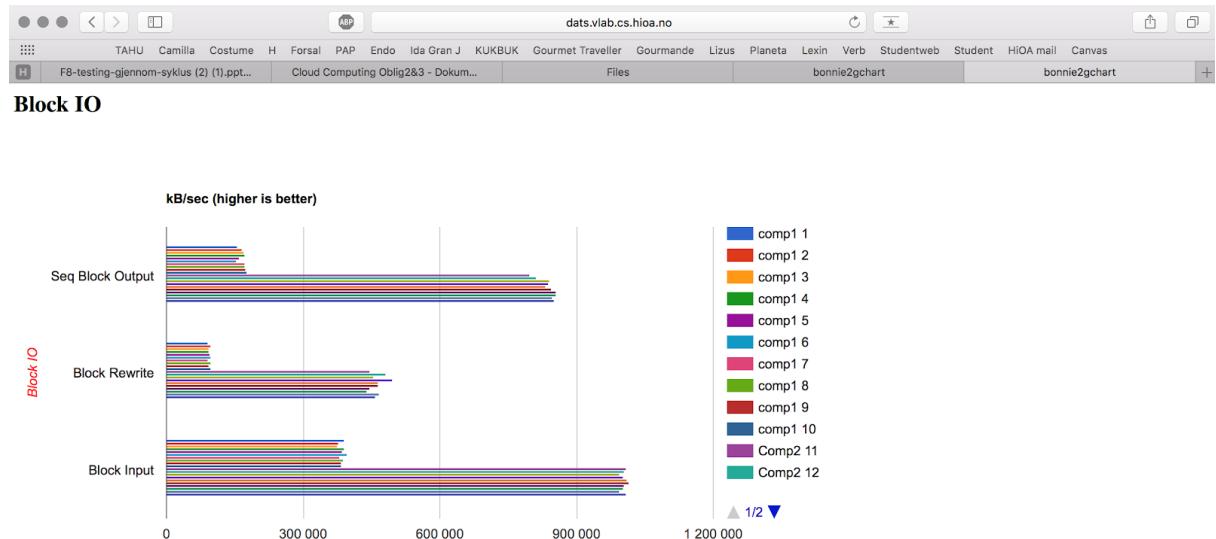


Figure 24: Screenshot of resulting graph from Block IIO test

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.

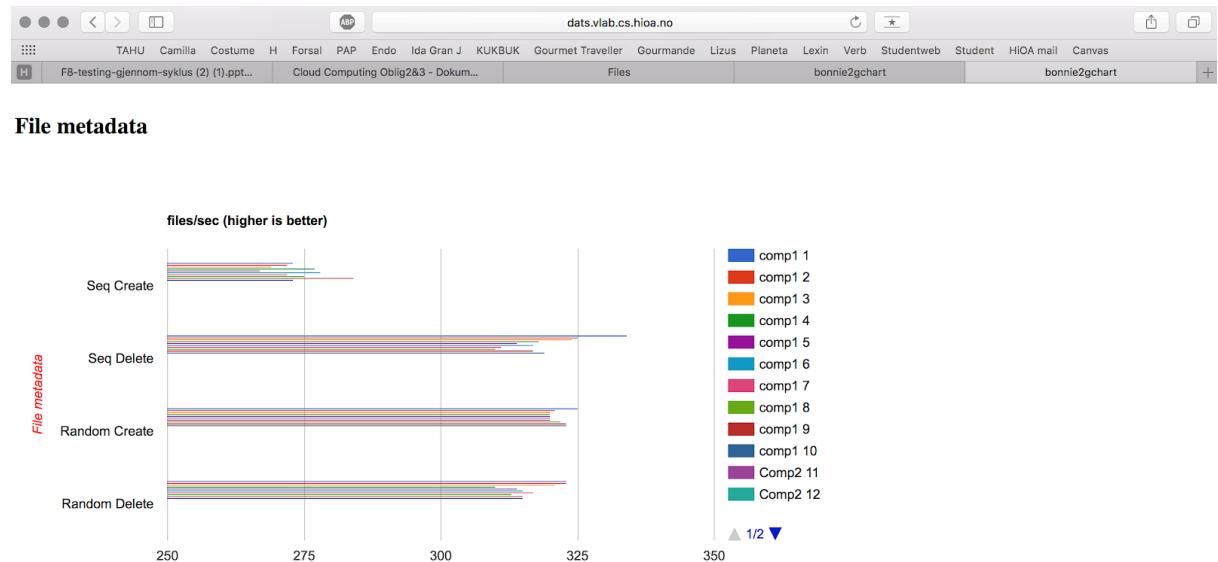


Figure 25: Screenshot of resulting graph from File Metadata test

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.

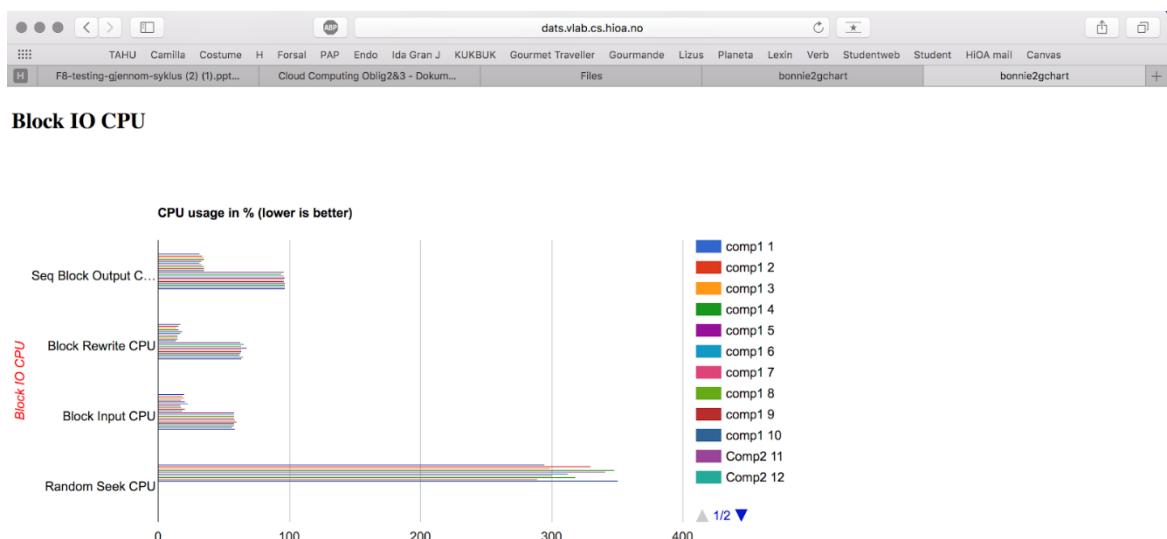


Figure 26: Screenshot of resulting graph from Block IO CPU test

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.

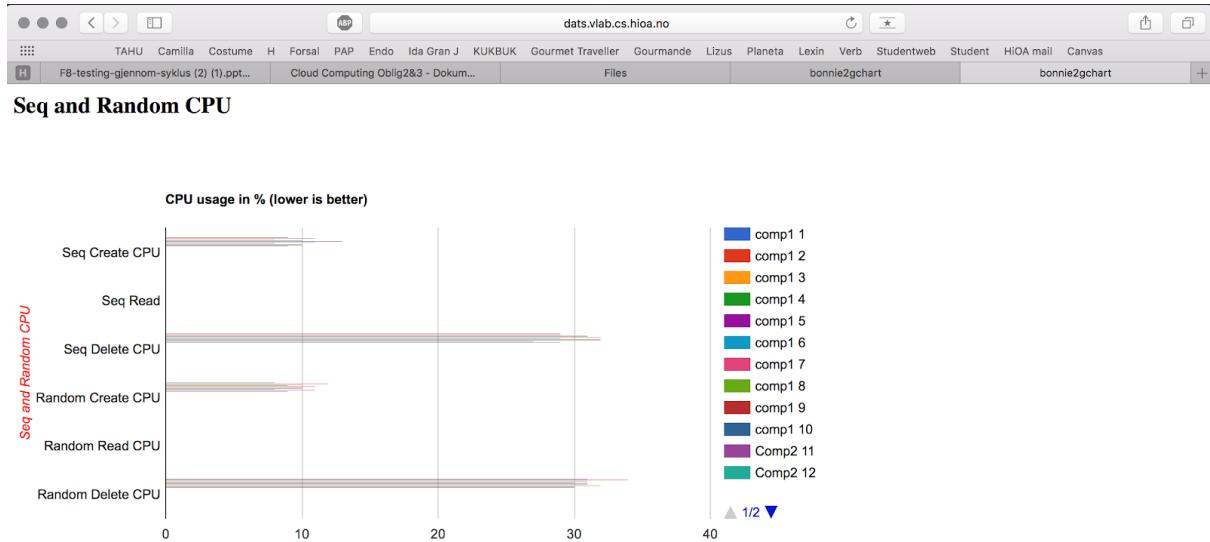


Figure 27: Screenshot of resulting graph from CPU usage test

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.

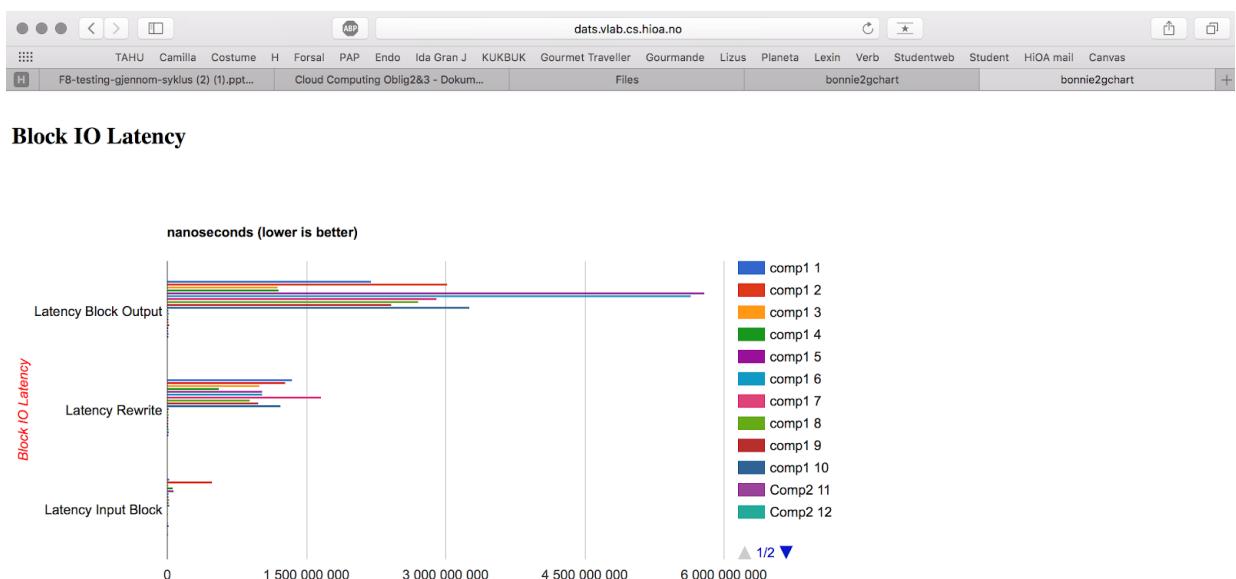


Figure 28: Screenshot of resulting graph from Block IO Latency test

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. No results are displayed for Comp2 in these categories as it completed the benchmark too 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.

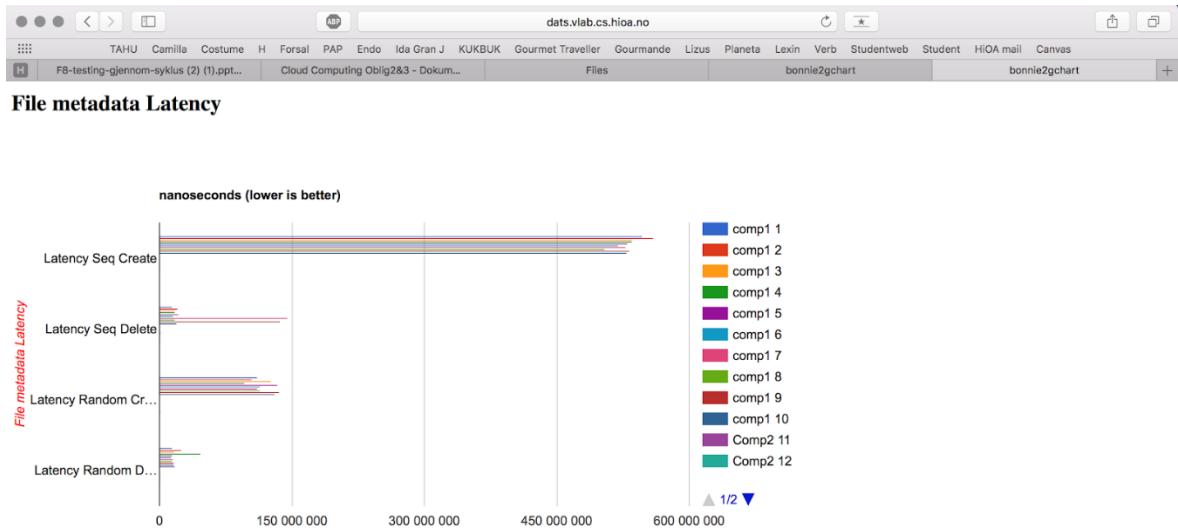


Figure 29: Screenshot of resulting graph from File metadata Latency test

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.

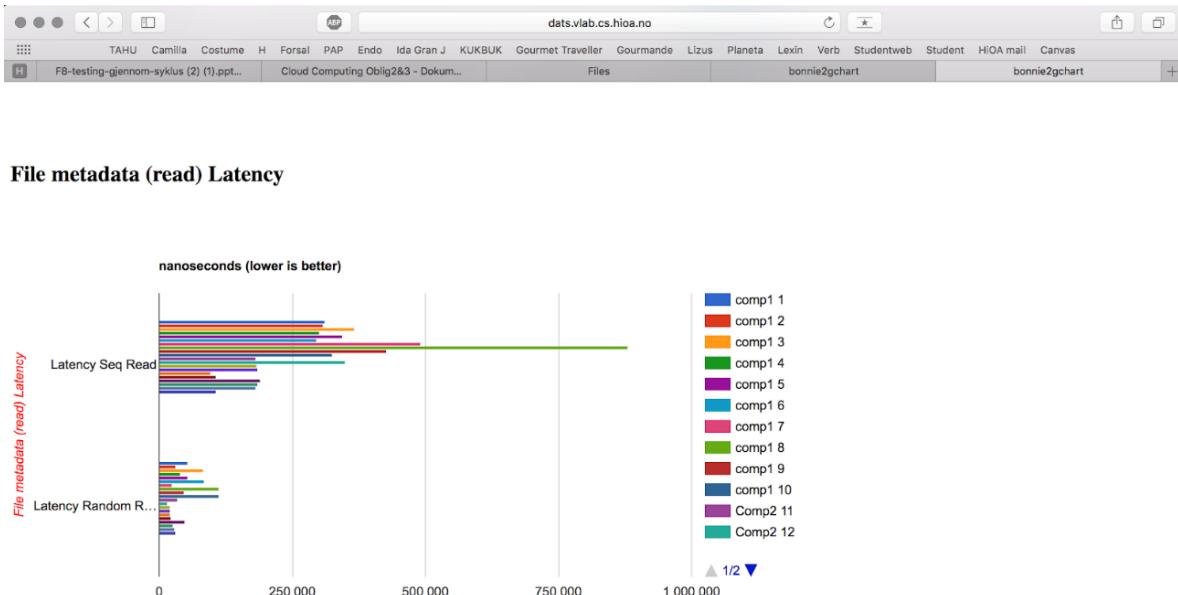


Figure 30: Screenshot of resulting graph from File Metadata (read) Latency test

file:///Users/ira/bout/bout.html

CARGO MENUS - marty...				F11-testledelse.pptx: AD...				https://www.istqb.org/do...				Cloud Computing Oblig2...				Files					
Version 1.97		Sequential Output		Sequential Input		Random Seek		Sequential Create		Random Create		Sequential Output		Sequential Input		Random Seek		Sequential Create		Random Create	
Size	Per Char	Block	Rewrite	Per Char	Block	% CPU	K/sec	Per Char	Block	% CPU	K/sec	Per Char	Block	% CPU	K/sec	Per Char	Block	% CPU	K/sec		
compl1	16G	91	91	136115	27	19377	17	249	99	69946	20	1203	205	5	271	1	+++++ +++	334	29	325	5
	Latency	146ms	232ms	549ms	1832ms	1153ms	76ms	Latency	57ms	12ms	14982ms	11ms	5ms	1640ms				323	31		
compl1	16G	97	97	16306	34	97062	16	239	99	678886	19	1318	330	5	272	11	+++++ +++	325	29	321	12
	Latency	96ms	316ms	127ms	7053ms	929ms	695ms	Latency	590ms	98ms	20652ms	105ms	31ms	2463ms				323	34		
compl1	16G	97	98	10463	35	28949	14	255	99	835710	20	1326	299	5	566	5	+++++ +++	324	31	320	10
	Latency	92ms	195ms	100ms	4387ms	128ms	707ms	Latency	576ms	67ms	17766ms	122ms	8ms	1676ms				321	31		
compl1	16G	96	96	17114	35	9756	16	253	99	869917	18	1306	388	5	277	10	+++++ +++	318	31	320	9
	Latency	98ms	170ms	564ms	6092ms	5697ms	1015ms	Latency	576ms	91ms	1793ms	9ms	96853ms	39ms			310	31			
compl1	16G	92	97	10145	33	84478	19	254	99	885568	21	1303	391	5	597	10	+++++ +++	314	32	320	11
	Latency	21ms	297ms	102ms	7452ms	6942ms	603ms	Latency	570ms	15ms	22301ms	134ms	5ms	421ms				314	39		
compl1	16G	97	97	15348	32	9706	17	259	99	696584	22	1314	313	5	278	11	+++++ +++	317	29	320	10
	Latency	97ms	567ms	104ms	5919ms	1327ms	1149ms	Latency	570ms	96ms	16095ms	116ms	85ms	1941ms				315	31		
compl1	16G	99	97	17223	34	9220	15	260	99	580219	17	1310	301	5	277	8	+++++ +++	311	32	320	10
	Latency	105ms	2994ms	165ms	59540ms	1353ms	108ms	Latency	578ms	91ms	145ms	11ms	25ms	18578ms				317	31		
compl1	16G	93	93	17206	35	9702	15	241	99	651421	18	1311	318	5	273	7	+++++ +++	310	33	322	8
	Latency	98ms	2799ms	109ms	2054ms	1973ms	621ms	Latency	576ms	96ms	1735ms	116ms	11ms	4292ms				313	32		
compl1	16G	97	96	17350	35	9303	15	260	99	848078	21	1296	269	5	280	10	+++++ +++	317	27	323	11
	Latency	99ms	227ms	98ms	6545ms	1359ms	135ms	Latency	573ms	27ms	137ms	136ms	46ms	16505ms				315	30		
compl1	16G	94	95	13728	35	80201	14	357	99	84426	16	1310	301	5	271	8	+++++ +++	310	26	322	9
	Latency	98ms	195ms	226ms	6119ms	8486ms	71ms	Latency	576ms	26ms	1990ms	11ms	11ms	7720ms				315	30		
Comp2	16G	172	98	79302	95	84359	63	735	99	100313	58	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	1021ms	1021ms	14668ms	14170ms	1911ms	2211ms	Latency	573ms	82ms	197ms	38ms	35ms	17228ms							
Comp2	16G	102	98	91263	91	80908	66	752	99	102582	58	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	841ms	1726ms	1671ms	1415ms	8431ms	1010ms	Latency	576ms	66ms	149ms	18ms	16ms	917ms							
Comp2	16G	191	98	83985	95	85958	64	743	99	994290	58	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	849ms	15465ms	15851ms	15851ms	8692ms	271ms	Latency	573ms	84ms	831ms	340ms	20ms	560ms							
Comp2	16G	181	98	83872	97	86469	68	749	99	100364	58	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	868ms	1535ms	1778ms	1622ms	1575ms	2275ms	Latency	573ms	85ms	725ms	168ms	21ms	530ms							
Comp2	16G	176	98	83269	95	86548	64	752	99	808325	59	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	919ms	1580ms	1765ms	1591ms	929ms	3192ms	Latency	573ms	76ms	137ms	271ms	20ms	545ms							
Comp2	16G	104	98	83556	95	86553	64	756	99	801616	60	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	949ms	1875ms	1087ms	1285ms	879ms	2190ms	Latency	576ms	79ms	851ms	265ms	75ms	526ms							
Comp2	16G	98	98	84526	95	84608	63	753	99	104671	59	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	950ms	1037ms	1059ms	1319ms	1094ms	2952ms	Latency	573ms	90ms	106ms	244ms	40ms	536ms							
Comp2	16G	177	98	83563	97	840578	62	756	99	1022719	58	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	997ms	1973ms	1345ms	15852ms	947ms	2360ms	Latency	573ms	88ms	877ms	296ms	26ms	498ms							
Comp2	16G	107	98	83673	97	86694	65	753	99	895732	57	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	95ms	1881ms	1597ms	1397ms	837ms	1091ms	Latency	573ms	92ms	103ms	252ms	30ms	543ms							
Comp2	16G	189	98	83189	97	85863	64	731	99	80834	59	+++++ +++	5	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	+++++ +++	
	Latency	958ms	1137ms	1169ms	1419ms	7346ms	1070ms	Latency	573ms	108ms	681ms	822ms	31ms	536ms							

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.

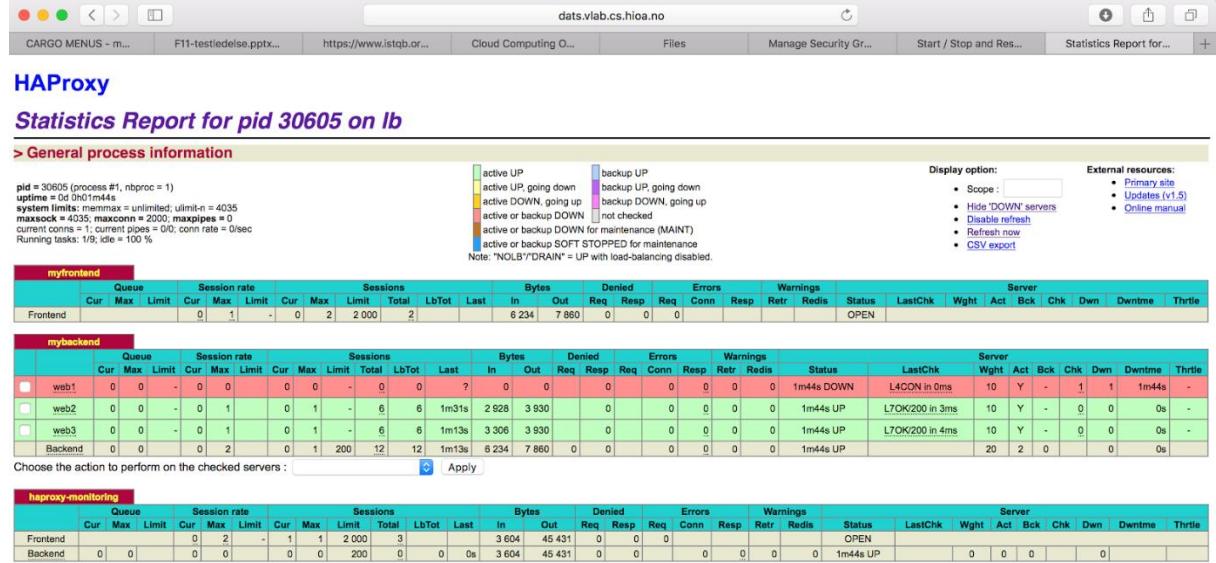


Figure 32: Screenshot of HAProxy monitoring pages when one web server is down

(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.

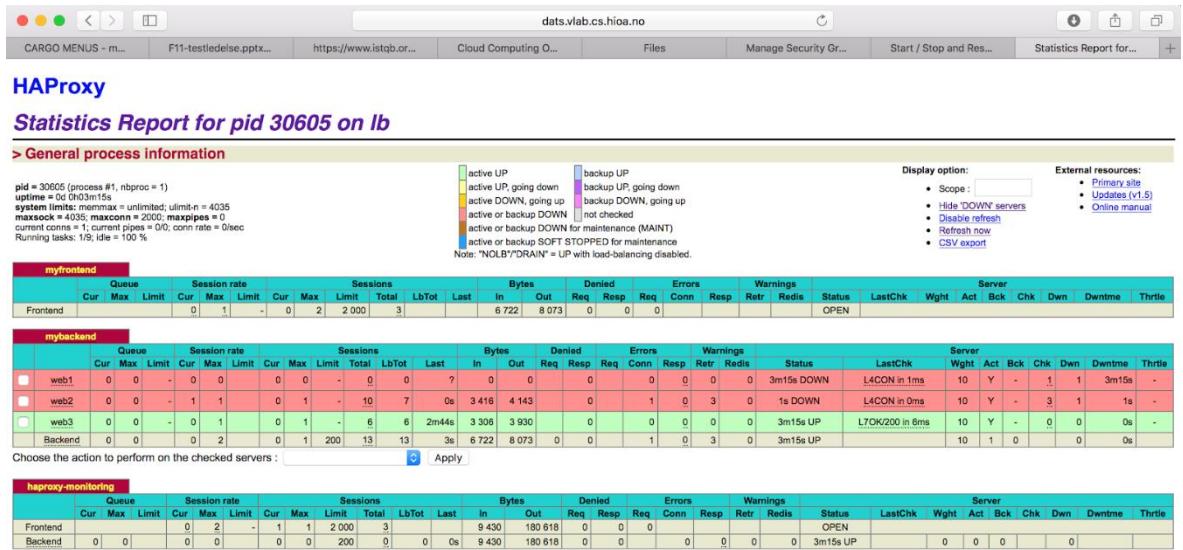


Figure 33: Screenshot of HAProxy monitoring pages when two web servers are down

(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.

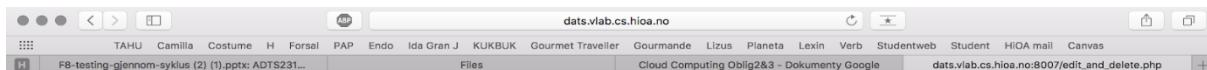
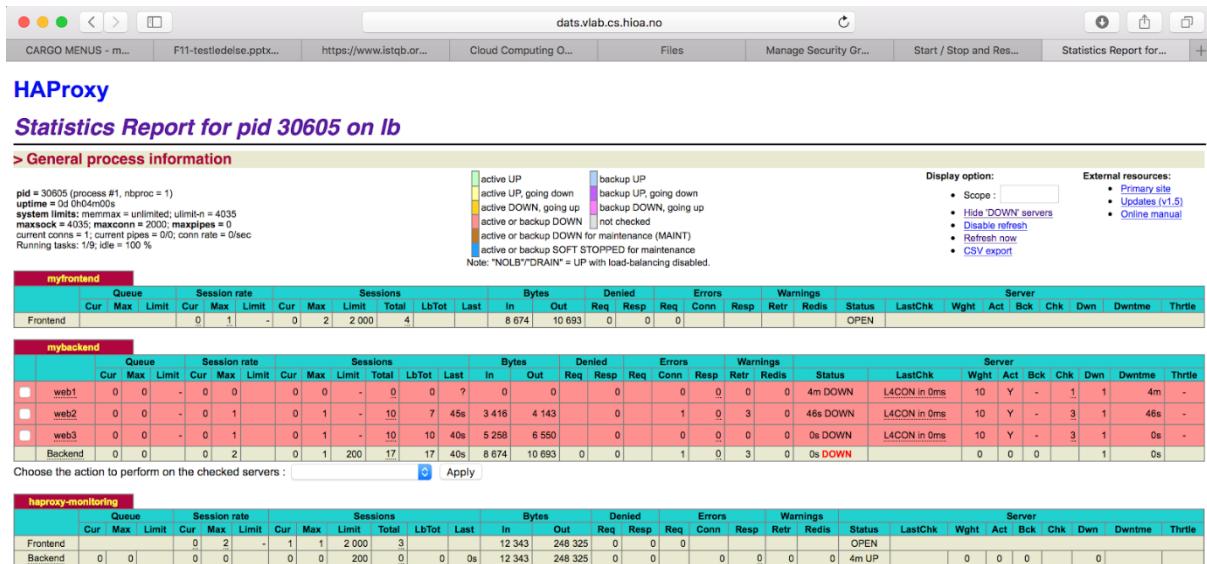


Figure 34: Screenshot of HAProxy monitoring pages when all web servers are down

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.

```
*** 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:~$ ]]

ira — ubuntu@maxscale: ~ — ssh dats07@dats.vlab.cs.hioa.no — 92x24

* 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.

*** System restart required ***
Last login: Wed May  2 09:53:45 2018 from 10.10.0.14
[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 | Master, Synced, Running
db3   | 10.10.1.243  | 3306 |          0 | Slave, Synced, Running
+-----+-----+-----+-----+
[ubuntu@maxscale:~$ ]
```

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.

```
*** 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:~$ ]
```

```
[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 reconfiguration ****
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 09:59:29 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