

Group 45 DCSG2003 Oblig 1

Raphael Storm Larsen, Sara Stentvedt Luggenes

Task 1

Instances

Instance ID =

Filter

Launch Instance

Delete Instances

More Actions ▾

Displaying 4 items

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	balancer	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.132.36, 10.212.169.121	gx1.1c1r	manager	Active	nova	None	Running	5 hours, 8 minutes	Create Snapshot ▾
<input type="checkbox"/>	ww2	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.131.231	gx1.1c1r	manager	Active	nova	None	Running	5 hours, 11 minutes	Create Snapshot ▾
<input type="checkbox"/>	ww1	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.131.189	gx1.1c1r	manager	Active	nova	None	Running	5 hours, 12 minutes	Create Snapshot ▾
<input type="checkbox"/>	manager	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.130.103, 10.212.173.5	gx1.2c2r	laptop	Active	nova	None	Running	6 days, 20 hours	Create Snapshot ▾

Displaying 4 items

Task 2

Floating IPs

Floating IP Address =

Filter

Allocate IP To Project

Release Floating IPs

Displaying 2 items

<input type="checkbox"/>	IP Address	Description	Mapped Fixed IP Address	Pool	Status	Actions
<input type="checkbox"/>	10.212.173.5	manager project	manager 192.168.130.103	ntnu-internal	Active	Disassociate ▾
<input type="checkbox"/>	10.212.169.121	balancer	balancer 192.168.132.36	ntnu-internal	Active	Disassociate ▾

Displaying 2 items

Task 3

To install apache and php on the two webserver, the following commands is executed as the root user.

```
# Installing apache webserver
apt-get update
```

```
apt-get install apache2
# Installing php and mysql support
apt-get install libapache2-mod-php php-mysql php-pgsql net-tools
apt-get install mysql-client
```

At the time of writing this documentation we no longer have the logs after installing apache and php, but the snippet below shows that the apache webserver is installed and running on webserver 1. The same has been completed on webserver 2.

```
ubuntu@ww1:~$ service apache2 status
• apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset:
   enabled)
   Active: active (running) since Tue 2023-01-17 10:16:31 UTC; 5h 2min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 10363 ExecStart=/usr/sbin/apachectl start (code=exited,
   status=0/SUCCESS)
   Main PID: 10368 (apache2)
     Tasks: 7 (limit: 1116)
    Memory: 14.6M
       CPU: 2.043s
    CGroup: /system.slice/apache2.service
           └─10368 /usr/sbin/apache2 -k start
           └─10369 /usr/sbin/apache2 -k start
           └─10370 /usr/sbin/apache2 -k start
           └─10371 /usr/sbin/apache2 -k start
           └─10372 /usr/sbin/apache2 -k start
           └─10373 /usr/sbin/apache2 -k start
           └─10439 /usr/sbin/apache2 -k start

Jan 17 10:16:31 ww1 systemd[1]: Starting The Apache HTTP Server...
Jan 17 10:16:31 ww1 systemd[1]: Started The Apache HTTP Server.
```

Another option is to use `ps aux | grep apache` to check if apache2 webserver is running.

To confirm that apache2 is listening to port 80, use `netstat -anltp`. If everything went right, the output should be a single line listed as "*apache2*".

Task 4

Using the manager VM, we can cURL the HTML from `/var/www/html/index.html` on ww1 using internal openstack IP addresses.

```
ubuntu@manager:~$ curl http://192.168.131.189
<!DOCTYPE html>
<html>
  <head>
    <title>Gruppe 45 - WWW1</title>
  </head>
```

```
<body style="background-color: yellow; font-family: Arial, Helvetica, sans-serif;display:flex;flex-flow:column;align-items: center;">
  <h1>Gult er kult!</h1>
  <h3>Fra Gruppe 45 www1</h3>
</body>
</html>
```

And the same for www2:

```
ubuntu@manager:~$ curl http://192.168.131.231
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Gruppe 45 - WWW2</title>
  </head>
  <body style="color:white;background-color: blue; font-family: Arial, Helvetica, sans-serif;display:flex;flex-flow:column;align-items: center;">
    <h1>Blått er r  tt!</h1>
    <h3>Fra Gruppe 45 www2</h3>
  </body>
</html>
```

Task 5

To install haproxy on the *balancer* VM, we use the following commands:

```
sudo add-apt-repository -y ppa:vbernat/haproxy-1.8
apt-get update
apt-get install haproxy socat
```

We then use the command `nano /etc/haproxy/haproxy.cfg` to open the haproxy config file. This is our current config file:

```
global
  log /dev/log      local0
  log /dev/log      local1 notice
  chroot /var/lib/haproxy
  stats socket /run/haproxy/admin.sock mode 660 level admin expose-fd
listeners
  stats timeout 30s
  user haproxy
  group haproxy
  daemon

  # Default SSL material locations
  ca-base /etc/ssl/certs
```

```

    crt-base /etc/ssl/private

    # See: https://ssl-config.mozilla.org/#server=haproxy&server-
    version=2.0.3&config=intermediate
    ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-
    GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-
    CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
    AES256-GCM-SHA384
    ssl-default-bind-ciphersuites
    TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
    ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
    log      global
    mode     http
    option    httplog
    option    dontlognull
    timeout  connect 5000
    timeout  client  50000
    timeout  server  50000
    errorfile 400 /etc/haproxy/errors/400.http
    errorfile 403 /etc/haproxy/errors/403.http
    errorfile 408 /etc/haproxy/errors/408.http
    errorfile 500 /etc/haproxy/errors/500.http
    errorfile 502 /etc/haproxy/errors/502.http
    errorfile 503 /etc/haproxy/errors/503.http
    errorfile 504 /etc/haproxy/errors/504.http

frontend AETA
    bind *:80
    mode http
    default_backend nodes

backend nodes
    mode http
    balance roundrobin
    server ww1 192.168.131.189:80 check weight 75
    server ww2 192.168.131.231:80 check weight 25

listen stats
    bind *:1936
    stats enable
    stats uri /
    stats hide-version
    stats auth raphael:boing
    stats auth saraslu:boing

```

Other useful commands are:

```

# To check for syntax errors in the config file
haproxy -c -f /etc/haproxy/haproxy.cfg
# To perform actions on the haproxy service

```

```
service haproxy start
service haproxy restart
service haproxy status
```

Challenge: Canary Releases

Using the *weight* keyword at the end of the server lines in the haproxy config file, we can allocate the amount of traffic to either server. Right now, 75% of traffic is routed to server 1, while 25% is routed to server 2. This was changed back to 50-50 before launching the bookface service.

Security Groups

We have opened port 22, 80 and 1936 in the openstack default security groups. This is to allow for http, ssh and access to the statistics report site.

Displaying 7 items

Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description
Egress	IPv4	Any	Any	0.0.0.0/0	-	-
Egress	IPv6	Any	Any	::/0	-	-
Ingress	IPv4	Any	Any	-	default	-
Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0	-	SSH
Ingress	IPv4	TCP	80 (HTTP)	0.0.0.0/0	-	HTTP
Ingress	IPv4	TCP	1936	0.0.0.0/0	-	Balancer Status
Ingress	IPv6	Any	Any	-	default	-

Displaying 7 items

In the browser

Now that the load balancer is online, we can access the website on our personal computers while connected to NTNU vpn. The blue website is only shown on every 4th refresh.





Task 7

A-D) Service Account

We retrieve the password and username for our service account, log on to openstack and download the openrc script.

```
Username: DCSG2003_V23_group45_service
Password: lATJUZekWli6
```

We then send the script to our manager VM using this command:

```
scp DCSG2003_V23_group45-openrc.sh ubuntu@10.212.173.5:~/
```

On the VM, we change the password in the file to correspond to the password of our service account.

```
ubuntu@manager:~$ cat ./DCSG2003_V23_group45-openrc.sh
#!/usr/bin/env bash
# To use an OpenStack cloud you need to authenticate against the Identity
# service named keystone, which returns a **Token** and **Service Catalog**.
# The catalog contains the endpoints for all services the user/tenant has
# access to - such as Compute, Image Service, Identity, Object Storage, Block
# Storage, and Networking (code-named nova, glance, keystone, swift,
# cinder, and neutron).
#
# *NOTE*: Using the 3 *Identity API* does not necessarily mean any other
# OpenStack API is version 3. For example, your cloud provider may implement
```

```
# Image API v1.1, Block Storage API v2, and Compute API v2.0. OS_AUTH_URL is
# only for the Identity API served through keystone.
export OS_AUTH_URL=https://api.skyhigh.iik.ntnu.no:5000
# With the addition of Keystone we have standardized on the term **project**
# as the entity that owns the resources.
export OS_PROJECT_ID=4661480d37154198b27092c5c15a765c
export OS_PROJECT_NAME="DCSG2003_V23_group45"
export OS_USER_DOMAIN_NAME="Default"
if [ -z "$OS_USER_DOMAIN_NAME" ]; then unset OS_USER_DOMAIN_NAME; fi
export OS_PROJECT_DOMAIN_ID="cb782810849b4ce8bce7f078cc193b19"
if [ -z "$OS_PROJECT_DOMAIN_ID" ]; then unset OS_PROJECT_DOMAIN_ID; fi
# unset v2.0 items in case set
unset OS_TENANT_ID
unset OS_TENANT_NAME
# In addition to the owning entity (tenant), OpenStack stores the entity
# performing the action as the **user**.
export OS_USERNAME="DCSG2003_V23_group45_service"
# With Keystone you pass the keystone password.
# echo "Please enter your OpenStack Password for project $OS_PROJECT_NAME as user
$OS_USERNAME: "
# read -sr OS_PASSWORD_INPUT
export OS_PASSWORD="lATJUZekWli6"
# If your configuration has multiple regions, we set that information here.
# OS_REGION_NAME is optional and only valid in certain environments.
export OS_REGION_NAME="SkyHiGh"
# Don't leave a blank variable, unset it if it was empty
if [ -z "$OS_REGION_NAME" ]; then unset OS_REGION_NAME; fi
export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
```

E-G) Openstack Client

We add this line to the end of the .bashrc file in the user and root user home directories. This will automatically run the script on the start of every session.

```
source /home/ubuntu/DCSG2003_V23_group45-openrc.sh
```

We run the script, and install the OpenStack client.

```
sudo apt-get update
sudo apt-get install python3-openstackclient
```

To make sure the installation was successful, we run the `openstack server list` command.

```
ubuntu@manager:~$ openstack server list
+-----+-----+-----+-----+
-----+-----+-----+-----+
```

ID	Name	Status	Networks
Image	Flavor		
-----+-----+-----+-----			
09422067-d0a3-4fe6-845e-84d66022a442	balancer	ACTIVE	
imt3003=10.212.169.121, 192.168.132.36	Ubuntu Server 22.04 LTS (Jammy Jellyfish)		
amd64 gx1.1c1r			
d4af6822-0c05-4935-a834-285b42cdd51f	ww2	ACTIVE	
imt3003=192.168.131.231	Ubuntu Server 22.04 LTS (Jammy Jellyfish)		
amd64 gx1.1c1r			
7530e313-4683-4d76-9a4e-bdb91f49cb8e	ww1	ACTIVE	
imt3003=192.168.131.189	Ubuntu Server 22.04 LTS (Jammy Jellyfish)		
amd64 gx1.1c1r			
781264fd-f153-45d2-bee5-980703505d55	manager	ACTIVE	imt3003=10.212.173.5, 192.168.130.103
Ubuntu Server 22.04 LTS (Jammy Jellyfish)			amd64 gx1.2c2r
-----+-----+-----+-----			
-----+-----+-----+-----			

Creating the database and launching bookface service

Create the virtual machine instance for the database

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
<input type="checkbox"/>	db1	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.130.219	gx3.4c4r	manager	Active	nova	None	Running	53 minutes	Create Snapshot
<input type="checkbox"/>	balancer	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.132.36, 10.212.169.121	gx1.1c1r	manager	Active	nova	None	Running	1 week	Create Snapshot
<input type="checkbox"/>	ww2	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.131.231	gx1.1c1r	manager	Active	nova	None	Running	1 week	Create Snapshot
<input type="checkbox"/>	ww1	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.131.189	gx1.1c1r	manager	Active	nova	None	Running	1 week	Create Snapshot
<input type="checkbox"/>	manager	Ubuntu Server 22.04 LTS (Jammy Jellyfish) amd64	192.168.130.103, 10.212.173.5	gx1.2c2r	laptop	Active	nova	None	Running	1 week, 6 days	Create Snapshot

Downloading the database

```
wget https://binaries.cockroachdb.com/cockroach-v22.2.3.linux-amd64.tgz
tar xzf cockroach-v22.2.3.linux-amd64.tgz
cp cockroach-v22.2.3.linux-amd64/cockroach /usr/local/bin
```

To make sure the code above works, we run the command `cockroach version`. This should give an output that verify the program is installed on the machine

We then make a directory at the top of the hierarchy(right below the root), with the name bfdata. Each CockroachDB node contains at least one store, specified when the node starts, which is where the cockroach process reads and writes its data on disk. In our case, bfdata will be this store.

Starting the database

```
cockroach start --insecure --store=/bfdata --listen-addr=0.0.0.0:26257 --http-addr=0.0.0.0:8080 --background --join=localhost:26257
```

Initializing the database

We only have to do this once. We chose username **AETAadmin** for our Cockroach database.

```
cockroach init --insecure --host=localhost:26257
```

To see whether the database is up and running, we use the command `ps aux | grep cockroach`, which should give a line as output corresponding to the running database.

When this is complete, we need to log in and create the user, tables and permissions. We use the cockroach command to open an SQL-session.

```
cockroach sql --insecure --host=localhost:26257

#In SQL shell, creates database and defines a user.
CREATE DATABASE bf;
CREATE USER AETAadmin;
GRANT ALL ON DATABASE bf TO AETAadmin;

#In SQL shell, these commands will set up the tables.
USE bf;
CREATE table users ( userID INT PRIMARY KEY DEFAULT unique_rowid(), name
STRING(50), picture STRING(300), status STRING(10), posts INT, comments INT,
lastPostDate TIMESTAMP DEFAULT NOW(), createDate TIMESTAMP DEFAULT NOW());
CREATE table posts ( postID INT PRIMARY KEY DEFAULT unique_rowid(), userID INT,
text STRING(300), name STRING(150), image STRING(32), postDate TIMESTAMP DEFAULT
NOW());
CREATE table comments ( commentID INT PRIMARY KEY DEFAULT unique_rowid(), postID
INT, userID INT, text STRING(300), postDate TIMESTAMP DEFAULT NOW());
CREATE table pictures ( pictureID STRING(300), picture BYTES );
GRANT SELECT,UPDATE,INSERT on TABLE bf.* to AETAadmin;
```

Viewing the cockroach dashboard

We use FoxyProxy to view the CockroachDB dashboard in the browser on our private computers. FoxyProxy can be installed as an extension directly on Google Chrome. After installing, we add a proxy for localhost with port 5000.

FoxyProxy - Proxy settings

General

Proxy Details

URL Patterns

Direct internet connection (no proxy)

Manual Proxy Configuration

Help! Where are settings for HTTP, SSL, FTP, Gopher, and SOCKS?

Host or IP AddresslocalhostPort5000

SOCKS proxy?

SOCKS v4/4a

SOCKS v5

Save Login Credentials

Authentication

Username

Password

Password - again

Proxy mode: Use proxy localhost:5000 for all URLs

Proxies

Enabled	Color	Proxy Name	Proxy Notes	Host or IP Address	Port	SOCKS proxy?	SOCKS Version	Auto PAC URL	
<div></div>	<div></div>	localhost:5000		localhost	5000	<div></div>	5		<div>Move Up</div> <div>Move Down</div>
<div></div>	<div></div>	Default	These are the settings that are used when no patterns match an URL				5		<div>Add New Proxy</div> <div>Edit Selection</div> <div>Copy Selection</div> <div>Delete Selection</div>

Import your proxies from FoxyProxy on Mozilla Firefox or from another computer.

Please Donate

Buy Proxy Service

By using port 5000 with FoxyProxy while connected to the manager-vm over ssh, we can connect to the Cockroach dashboard by using the internal IP address on the database VM. When we connect to the manager with SSH, we must use this option for it to work with FoxyProxy: `ssh -D 5000 ubuntu@10.212.173.5`.

Use proxies based on their pre-defined patterns and priorities

Use proxy localhost:5000 for all URLs

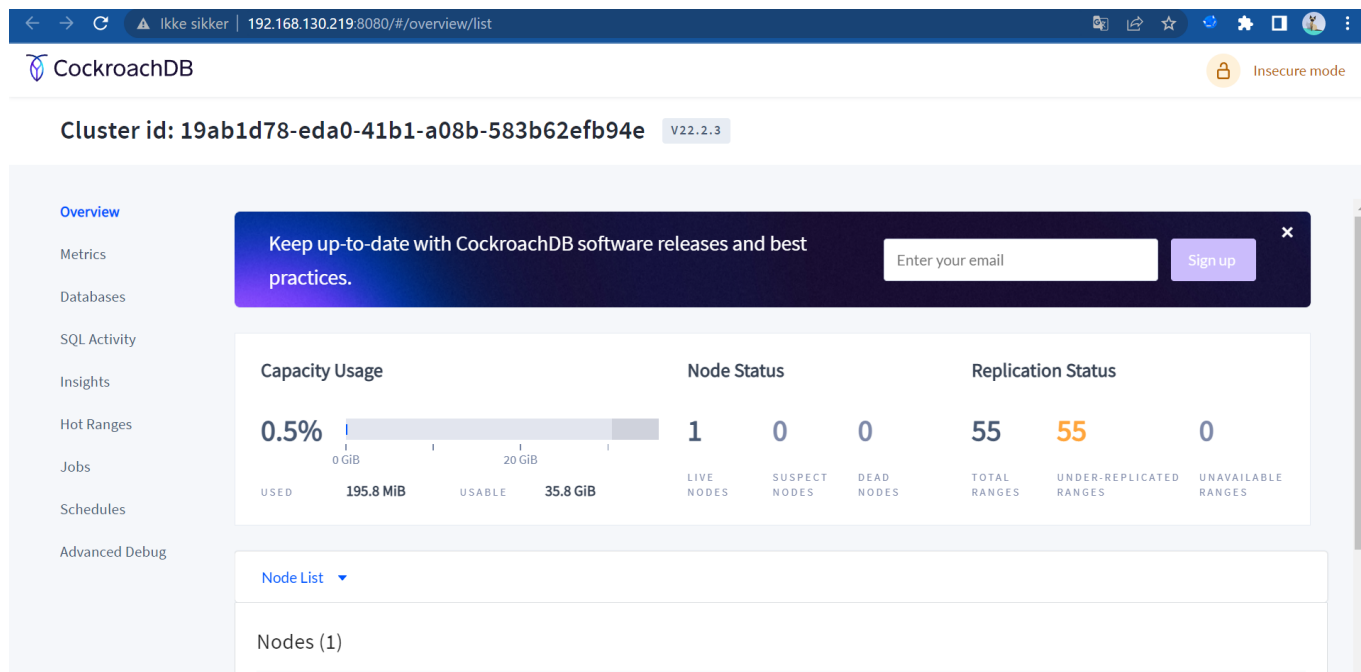
Use proxy Default for all URLs

Disable FoxyProxy

Options

We can then connect to `http://192.168.130.219:8080` in the browser, where the ip address is the database's internal address. The cockroach dashboard use port 8080, which we add at the end of the address.

10 / 14



Setting up the webserver for bookface

The steps below are completed on both webserver. The additional packs *php-pgsql* and *net-tools* required for this step was also added to the previous installation documentation under the *Task 3* subtitle. The commands below are thus redundant.

```
apt-get update
apt-get install apache2 libapache2-mod-php php-pgsql net-tools
```

Next, download the bookface-code from the git-repository, and cd into the bookface directory. We used the */home/ubuntu/* home folder as the destination for the repository.

```
git clone https://github.com/hioa-cs/bookface.git
cd bookface
```

Remove the default index.html file in the html folder and move the main PHP files into apache's document root:

```
rm /var/www/html/index.html
cp code/* /var/www/html/
```

Next, we make a copy of the example configuration file in the bookface directory with the name *config.php*, and put it in the */var/html/* directory:

```
cp config.php /var/www/html/config.php
```

Then we can edit this config-file:

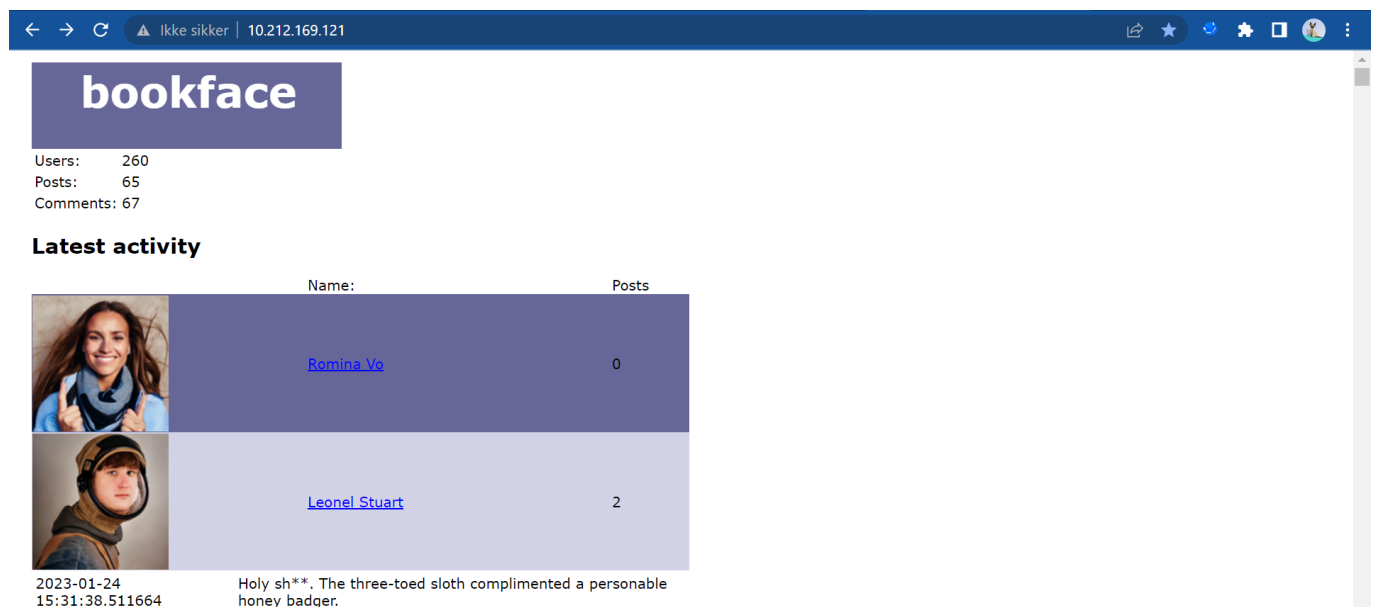
```
nano /var/www/html/config.php
```

The configuration file for bookface now looks like this, where dbhost is the ip address for the database(db2), and webhost is the ip address for the loadbalancer.

```
<?php
$dbhost = "192.168.130.246";
$dbport = "26257";
$db = "bf";
$dbuser = "AETAadmin";
$dbpasswd = '';
$webhost = '10.212.169.121';
$weburl = 'http://' . $webhost ;
$frontpage_limit = "1000";
?>
```

Testing the setup

While connected to the NTNU network, we can enter the loadbalancers floating ip address into the browser, and verify that BookFace is indeed running:



Additions

How to remove users with no images:

First, go to your database and start the console:

```
cockroach sql --insecure --host=localhost:26257
```

Inside the console run each of these commands (each of these is a single long line):

```
use bf;

delete from comments where exists (select postid from posts where postid =
comments.postid and exists ( select * from users where posts.userid = users.userid
and not exists ( select * from pictures where pictureid = users.picture)));

delete from comments where exists ( select * from users where comments.userid =
users.userid and not exists ( select * from pictures where pictureid =
users.picture));

delete from posts where exists ( select * from users where posts.userid =
users.userid and not exists ( select * from pictures where pictureid =
users.picture));

delete from users where not exists ( select * from pictures where pictureid =
users.picture);
```

Regarding db1/db2

The db1 virtual machine was created using a gx3.4c4r flavor, as was proposed in the tutorial. However, upon reaching about 500 users our database would start crashing due to a shortage of memory. Even with a script that automatically restarted cockroach we could barely get 50% uptime. So instead of wasting more time trying to improve the situation, we decided to instead scale vertically. We created a new database: db2, in the sx3.8c16r flavor. With 4 times the memory of db1, we are now well above 1000 users with a 97,5% uptime. Not ideal but drastically improved. This is why you will see the ip of both db1 and db2 in this report. Db1 is currently shut down.

Useful notes on databases

- This is the error-code that occurs when the database is offline: *"SQLSTATE[08006] [7] connection to server at "192.168.130.219", port 26257 failed: ERROR: server is not accepting clients, try another node"*
- The database log files can be found at [/bfdata/logs#](#)

Our .bashrc file additions

The following lines are added at the end of the `~/ .bashrc` file. This makes our aliases persist over multiple sessions.

```
source /home/ubuntu/DCSG2003_V23_group45-openrc.sh

# Aliaser for å koble til webservere
alias www1="ssh ubuntu@192.168.131.189"
alias www2="ssh ubuntu@192.168.131.231"
alias balancer="ssh ubuntu@192.168.132.36"
alias db1="ssh ubuntu@192.168.130.219"
alias db2="ssh ubuntu@192.168.130.246"
```

Crontab

We utilize crontab to automatically check the status of our database. We use `crontab -e` while on the root user to edit the file.

```
# Every 15 minutes, this script is executed. The scripts checks if cockroach is  
down, in which case it is restarted.  
0,15,30,45 * * * * bash /home/ubuntu/checkDbStatus.sh
```