Table of Contents

Creating an Ambience Cluster for High Availability

Ambience is fully HA compliant. All jobs can run on any node of the cluster and all data are replicated across the cluster.

In case of a node failure, jobs on the failed node are automatically run on other nodes.

When the failed node recovers, it automatically joins the cluster again.

Configuring the Cluster

Note: I use Debian 7 for all my servers. Adapt the commands to whichever OS you are using.

Let's say we have three nodes A,B,C with the following IP addresses:

A – 199.168.117.23

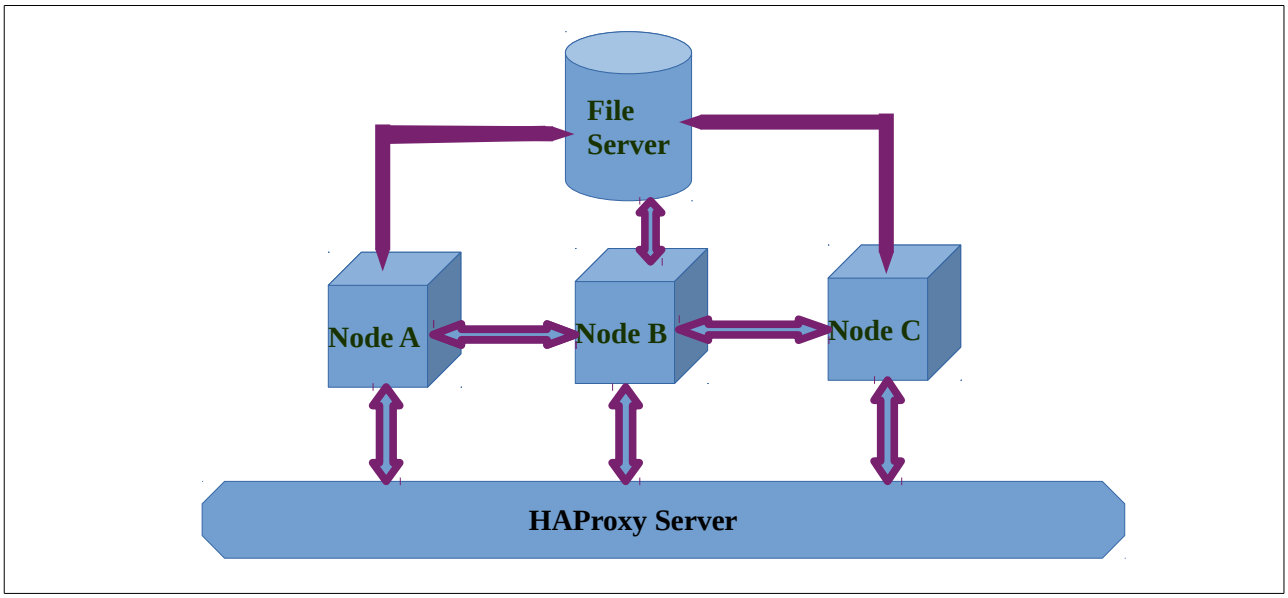
B – 199.168.117.24

C – 199.168.117.30

All three nodes should be running the recommended version of Java (currently **Java 8**) on them. This is mandatory to run Ambience 4.2.

There is also a file server (199.168.117.59) that contains the Dacapo data folder shared between these three nodes.

Finally, a HAProxy server (199.168.117.60) load balances the web serving of these three nodes.



Installing Ambience on the Three Nodes

1. Login to the three nodes and create a new user called **ambience**:

adduser ambience

2. Logout and login to the three nodes as user **ambience** and unzip the Ambience software.

For e.g.

unzip elx-ambience-4.2.0.zip

3. Navigate to the **etc/** folder within the Ambience structure and edit **application.conf**

On each server, change **client="localhost"** to the IP address of the server. For e.g. **client=199.168.117.23**

4. On each server, change:

seed-nodes = [

"akka.tcp://elx-ambience@"${host}":"${seed-port}

]

to:

seed-nodes = [

"akka.tcp://elx-ambience@199.168.117.23:"${seed-port},

"akka.tcp://elx-ambience@199.168.117.24:"${seed-port},

"akka.tcp://elx-ambience@199.168.117.30:"${seed-port}

]

Now you have a cluster of three nodes. Each node can connect to the others. In case a node or two

goes down, the other will still run. When the failed nodes come back online, they will automatically join the cluster.

Note: You can change the order of the seed-nodes. However, the seed-nodes configuration should be identical on all the servers.

5. Save the configuration.

Configuring the File Server for the Dacapo Data Folder

Note: You can use any storage method such as NFS, S3, Samba etc for this.

I use sshfs.

1. Create data mount points on the three nodes.

Login to each node as user **ambience** and create a folder called **dacapo** in **/home/ambience**

cd /home/ambience

mkdir dacapo

2. Login to the file server and install sshfs as **root**.

apt-get install sshfs

3. Create a new user called **ambience**. Never operate as root.

Run:

adduser ambience

to add a new user called ambience.

4. Make sure that the **fuse** kernel module is loaded:

lsmod | grep fuse

The command should return output similar to the following :

lsmod | grep fuse
fuse 62012 1

If the **fuse** module is not loaded, load it:

modprobe fuse

5. Add user **ambience** to the fuse group:

adduser ambience fuse

6. Change to user **ambience**:

su ambience

7. Navigate to the **ambience** home directory:

cd /home/ambience

8. Make a folder called **dacapo** on the fileserver under the **/home/ambience** folder.

mkdir dacapo

9. Change back to the **root** user:

exit

10. Create a public/private keypair to use to mount the filesystem on the nodes:

ssh-keygen

Do not enter a passphrase otherwise mounting will not work without human interaction. Simply hit ENTER when prompted for a passphrase.

11. Copy the generated keygen to all the three nodes.

ssh-copy-id -i $HOME/.ssh/id_rsa.pub ambience@199.168.117.23

ssh-copy-id -i $HOME/.ssh/id_rsa.pub ambience@199.168.117.24

ssh-copy-id -i $HOME/.ssh/id_rsa.pub ambience@199.168.117.30

12. Mount this dacapo folder as **/home/ambience/dacapo** on each of the three nodes.

sshfs -o idmap=user ambience@199.168.117.23:/home/ambience/dacapo /home/ambience/dacapo -o nonempty

sshfs -o idmap=user ambience@199.168.117.24:/home/ambience/dacapo /home/ambience/dacapo -o nonempty

sshfs -o idmap=user ambience@199.168.117.30:/home/ambience/dacapo /home/ambience/dacapo -o nonempty

13. To automatically mount at boot, add these commands to **/etc/rc.local**.

14. Login to each of the nodes as user **ambience**, create the data folder in the Ambience folder if it does not exist and link the mounted dacapo folder as the Ambience dacapo folder.

cd /home/ambience/ElixirAmbience

mkdir data

cd data

ln -s /home/ambience/dacapo dacapo

Ambience initialization needs to be done only once at startup by the first node listed in the seed node configuration.

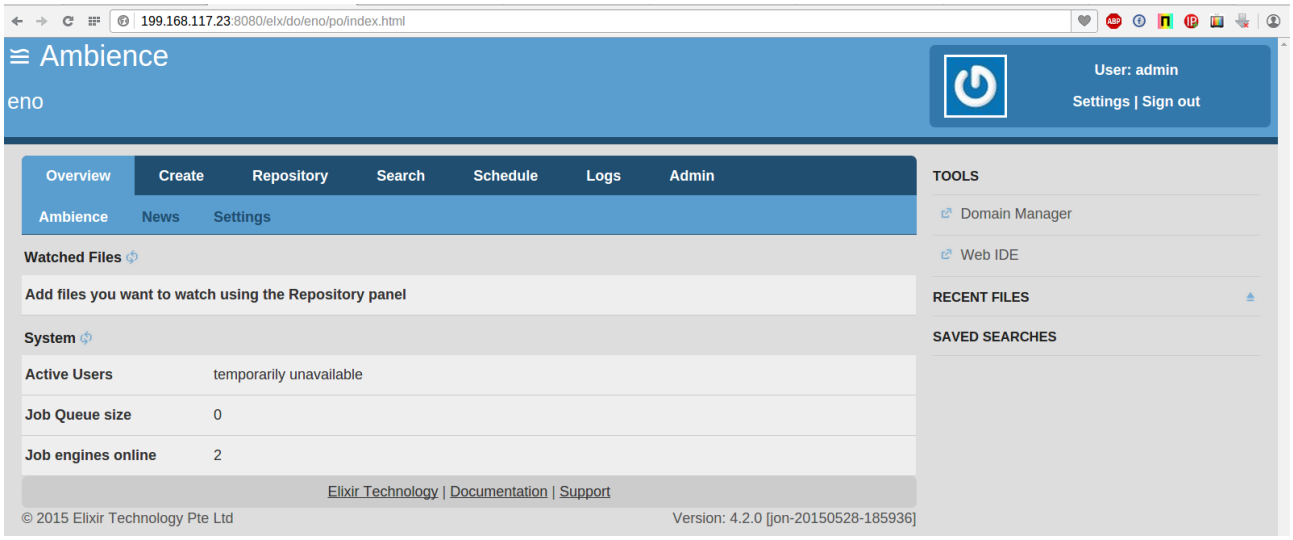
We recommend that you delete the **init** directory inside the Ambience folder structure from all the other seed nodes, to prevent problems wth initalization when two nodes start up simultaneously.

Testing the Cluster

1. Always ***start the first node*** mentioned in the seed-node configuration. This is important as it initializes the cluster correctly.

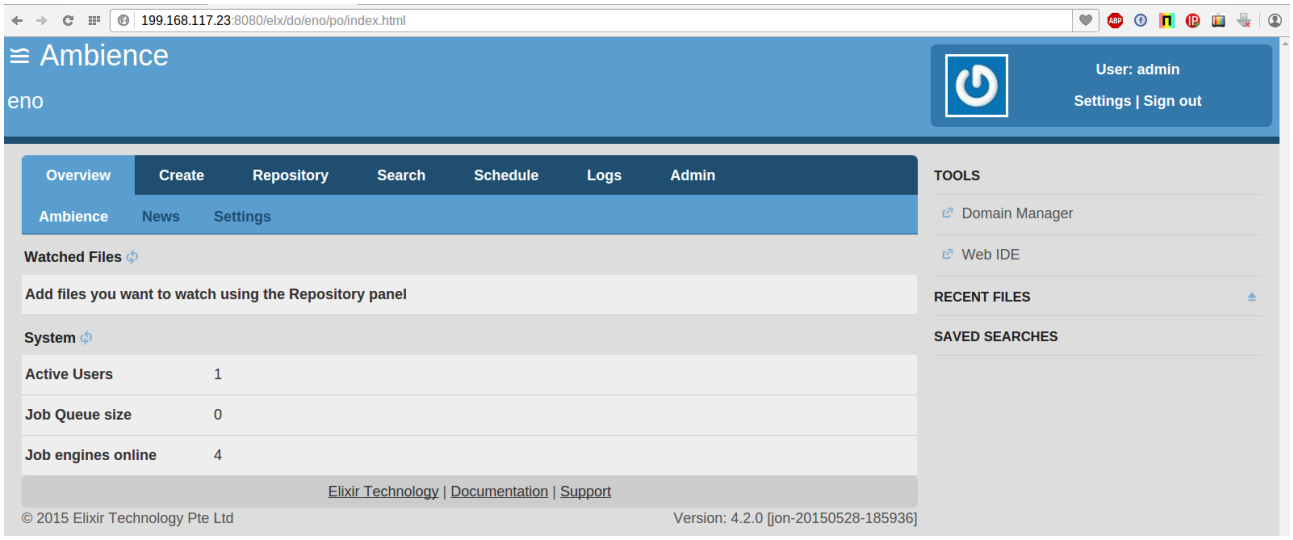
2. Login to Ambience using the node IP and port 8080 (default). The default domain is **eno**, username is **admin** and password is **sa**.

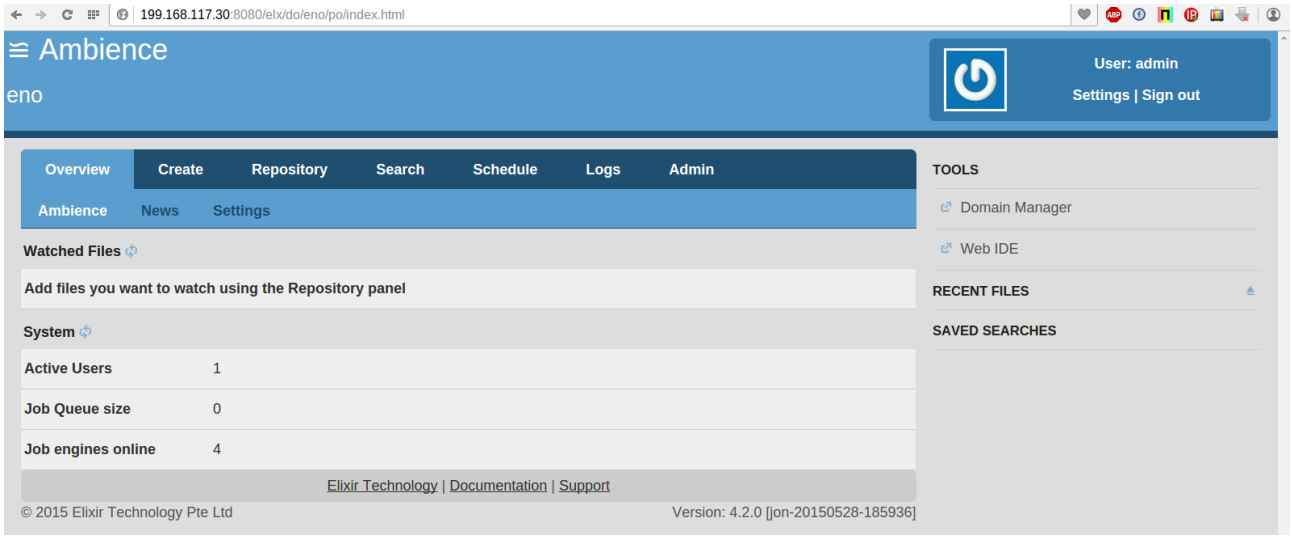
3. Note the number of Job Engines. It should be 2 (By default 2 Job Engines are started for each Ambience instance).



4. Next, start up Ambience on any of the other two servers.

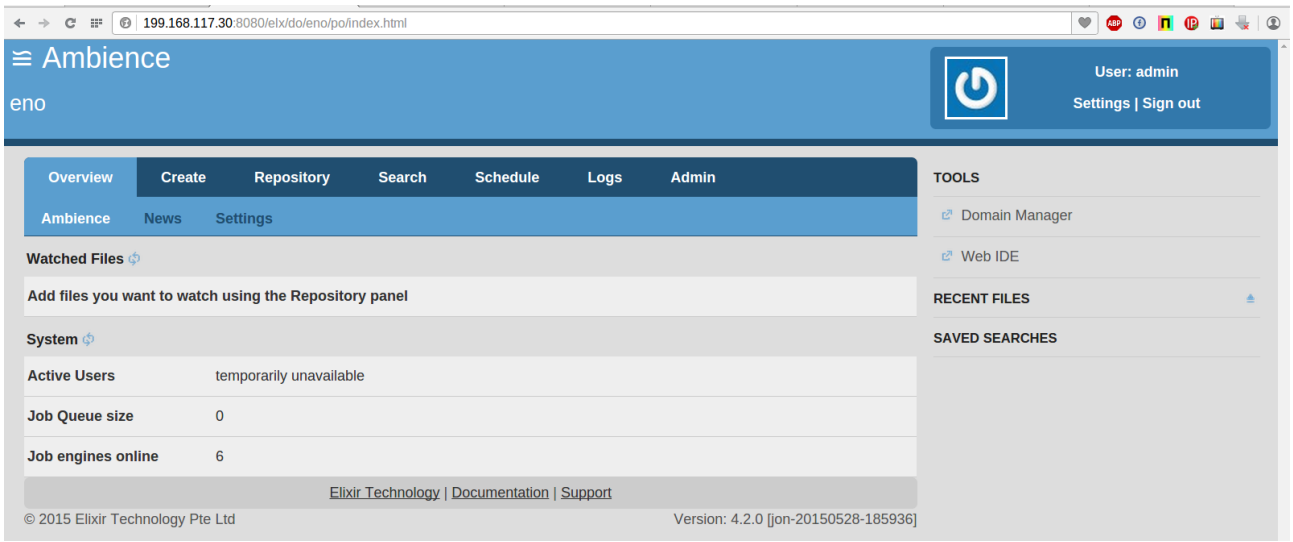
5. Wait a minute and view the number of Job Engines in any of the running Ambience instances. The Job Engines count should now be 4.



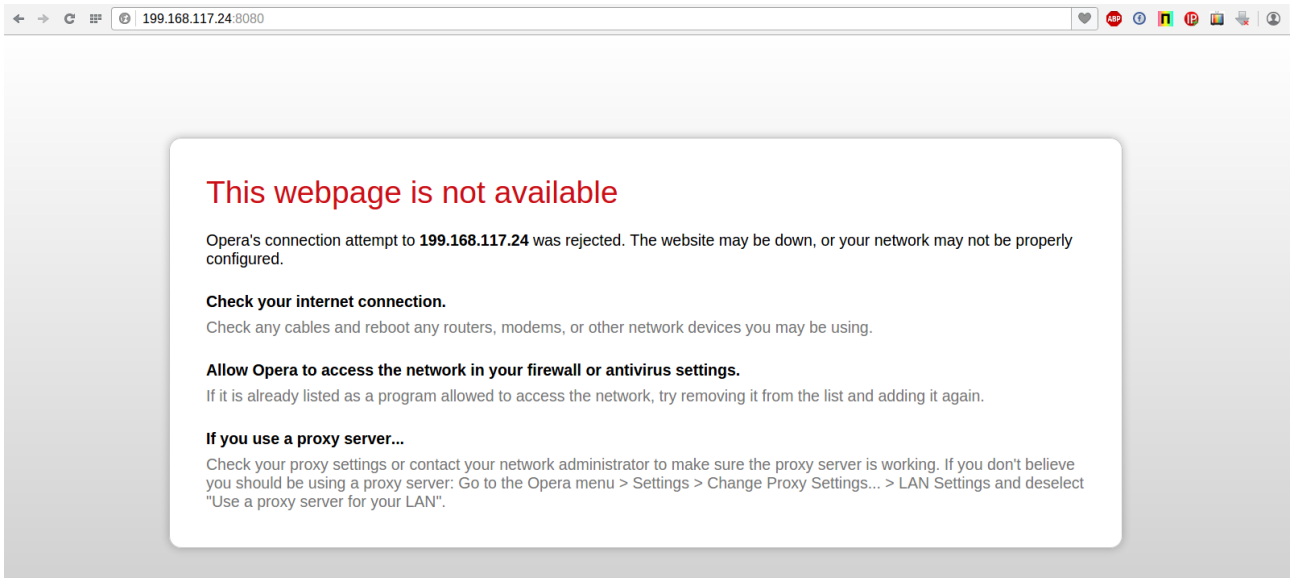


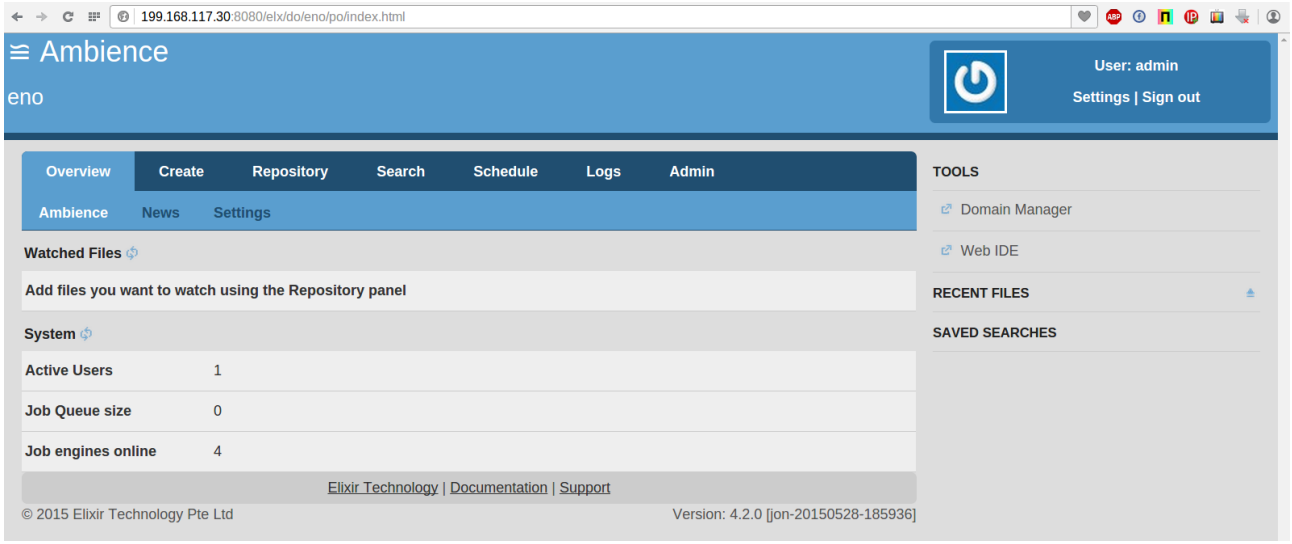
6. Now start the remaining Ambience instance.

7. Wait a minute and view the number of Job Engines in any of the running Ambience instances. It should now be 6.

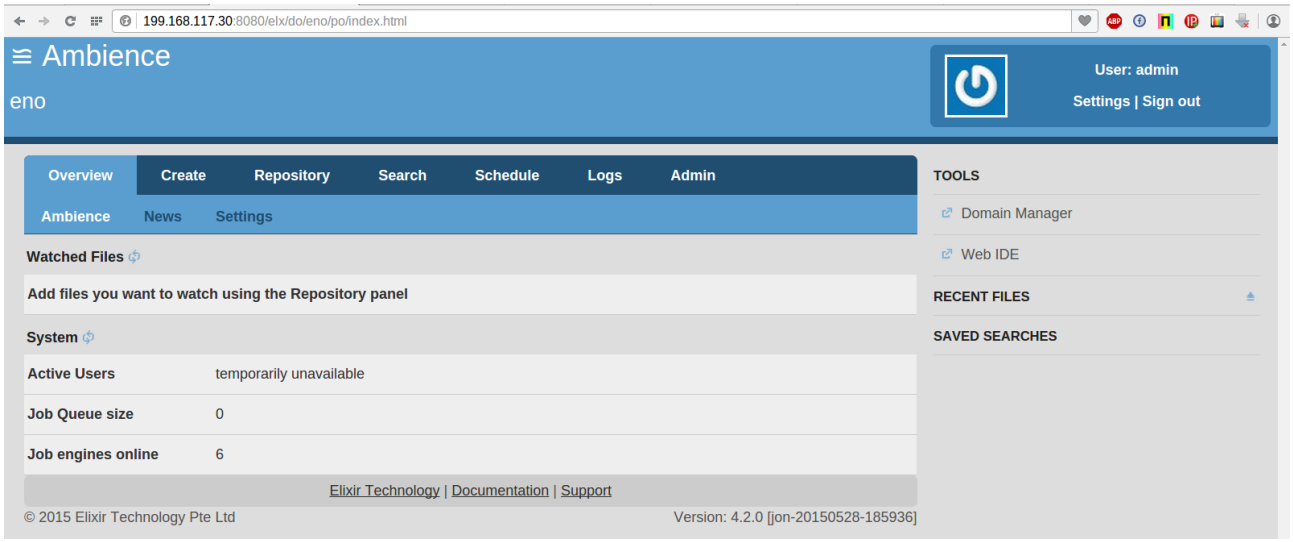


8. Stop any Ambience instance (for example: **199.168.117.24**) to simulate a failure. The Job Engine count should drop to 4.





9. Start the stopped Ambience instance (**199.168.117.24** in our example) to simulate recovery. The Job Engine count should increase to 6 again.



The above examples serve to show you how simple it is to configure Ambience nodes as a cluster, for redundancy.

Load Balancing Ambience Web Servers

Configuring HAProxy

HAProxy (High Availability Proxy) is an open source load balancer which can load balance any TCP service. It is particularly suited for HTTP load balancing as it supports session persistence and layer 7 processing.

Installation:

1. On the load balancer, run the following commands as root:

***echo deb http://httpredir.debian.org/debian wheezy-backports main | ***

tee /etc/apt/sources.list.d/backports.list

apt-get update

apt-get install haproxy -t wheezy-backports

Configuring HAProxy

1. Edit **`/etc/default/haproxy`**

and set:

ENABLED=1

2. To start HAProxy on boot, run:

update-rc.d haproxy defaults

3. Move the default configuration file and create a new one.

```
mv /etc/haproxy/haproxy.cfg{,.original}
```

4. Create a new file: **/etc/haproxy/haproxy.cfg** with the following contents:

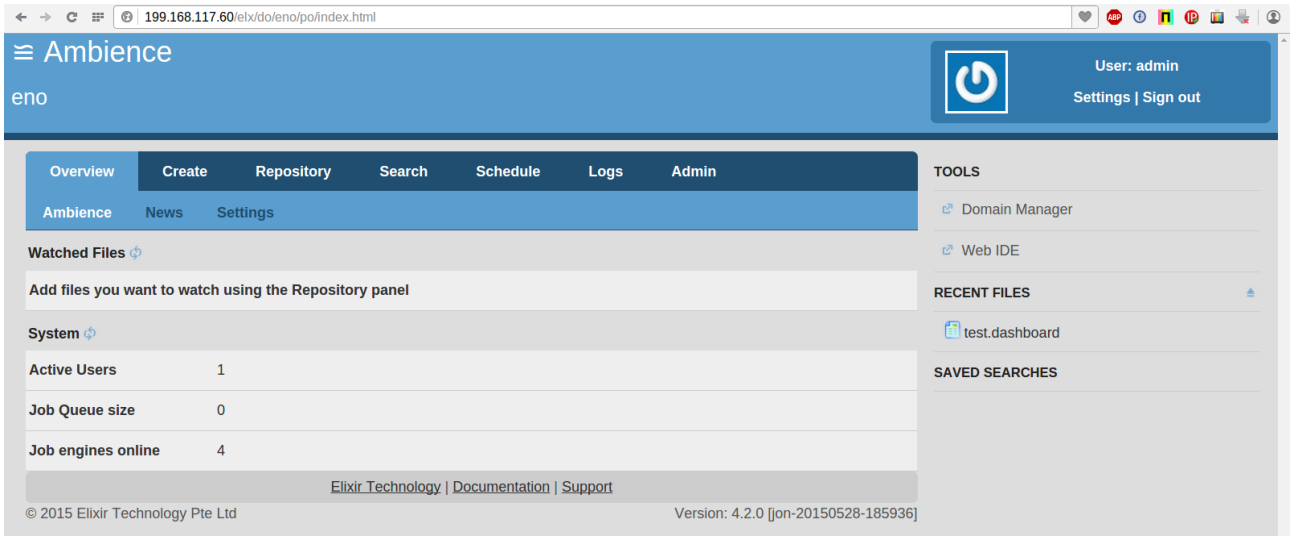
```
global
   log 127.0.0.1 local0 notice
   maxconn 2000
   user haproxy
   group haproxy

defaults
  log global
  mode http
  option httplog
  option dontlognull
  retries 3
  option redispatch
  timeout connect 5000
  timeout client 10000
  timeout server 10000


listen haproxy 199.168.117.60:80
  mode http
  balance leastconn
  option http-server-close
  timeout http-keep-alive 3000
  option forwardfor
  cookie SRVNAME insert
  server node1 199.168.117.23:8080 cookie N1 check
  server node2 199.168.117.24:8080 cookie N2 check
  server node3 199.168.117.30:8080 cookie N3 check
```

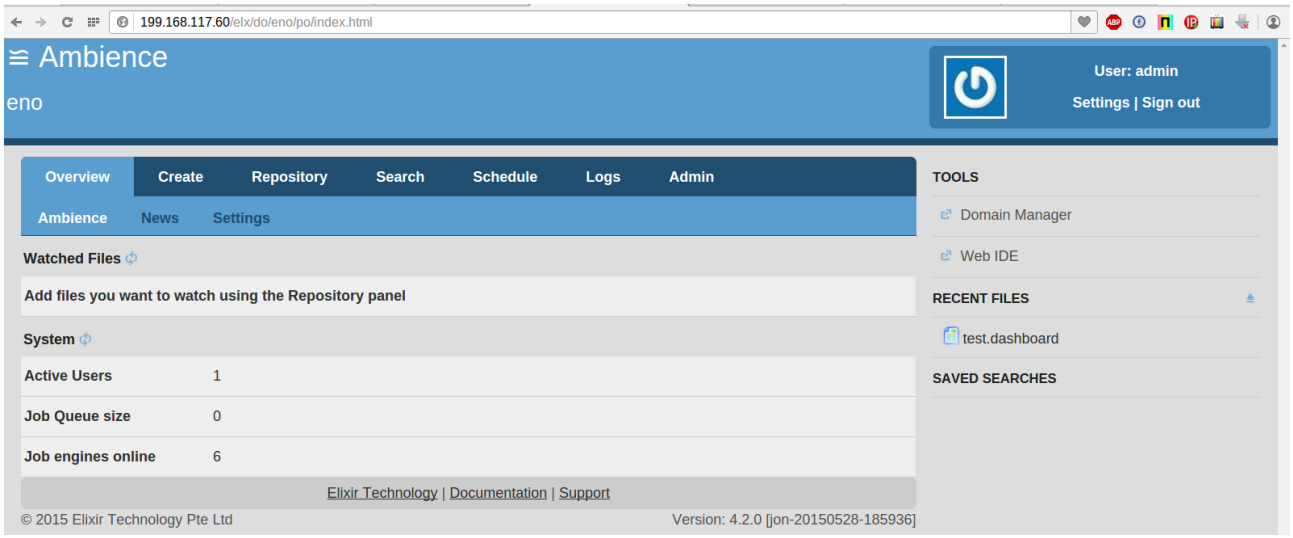
5. Start HAProxy:

```
service haproxy start
```

Now start Ambience on your nodes and access it with your HAProxy IP on port 80.

An example with my HAProxy loadbalancer (199.168.117.60) is shown below.





Use the powerful, easy to use cluster for your needs.