

❖ DOKUMENTASI

I. INSTALASI AMBARI DENGAN CENTOS 6

A. Persiapan dan Instalasi Tools :

- WinSCP
- PuTTY
- VMWare
- Sebelum menginstall Ambari, diperlukan melakukan instalasi terhadap 3 buah mesin sistem operasi CentOS 6.4 pada VMware, dengan rincian :
 - 1 mesin / 1 node sebagai server (master), dengan jumlah memory minimal 40 GB.
 - 2 mesin / 2 node sebagai host (node1, node2), dengan jumlah memory masing-masing minimal 30 GB.

B. Instalasi Apache Ambari 2.4.0

Bagian I – Persiapan

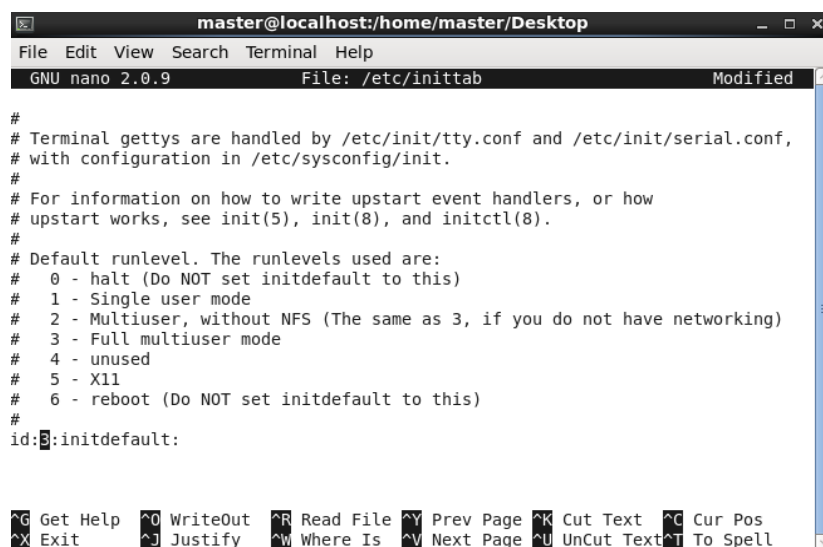
1. Jalankan ketiga mesin CentOS yang telah dibuat sebelumnya, lalu bukalah terminal dan masuk sebagai root dengan perintah “su”, lalu isikan password yang telah dibuat sebelumnya.

```
[master@localhost Desktop]$ su  
Password:
```

2. Kemudian lakukan konfigurasi pada file inittab, pada mesin master, node1, dan node2. Dengan perintah sebagai berikut :

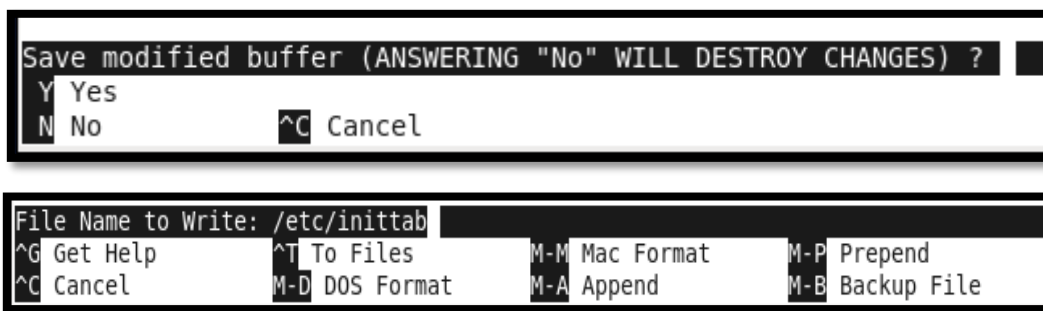
```
[root@localhost Desktop]# nano /etc/inittab
```

3. Lalu ubahlah id runlevel menjadi 3 (Full multiuser mode).



```
master@localhost:/home/master/Desktop  
File Edit View Search Terminal Help  
GNU nano 2.0.9 File: /etc/inittab Modified  
  
#  
# Terminal gettys are handled by /etc/init/tty.conf and /etc/init/serial.conf,  
# with configuration in /etc/sysconfig/init.  
#  
# For information on how to write upstart event handlers, or how  
# upstart works, see init(5), init(8), and initctl(8).  
#  
# Default runlevel. The runlevels used are:  
# 0 - halt (Do NOT set initdefault to this)  
# 1 - Single user mode  
# 2 - Multiuser, without NFS (The same as 3, if you do not have networking)  
# 3 - Full multiuser mode  
# 4 - unused  
# 5 - X11  
# 6 - reboot (Do NOT set initdefault to this)  
#  
id:3:initdefault:  
  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

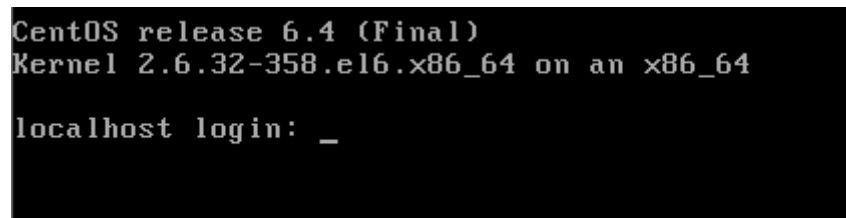
4. Kemudian simpanlah konfigurasi file inittab dengan menekan Ctrl+X.



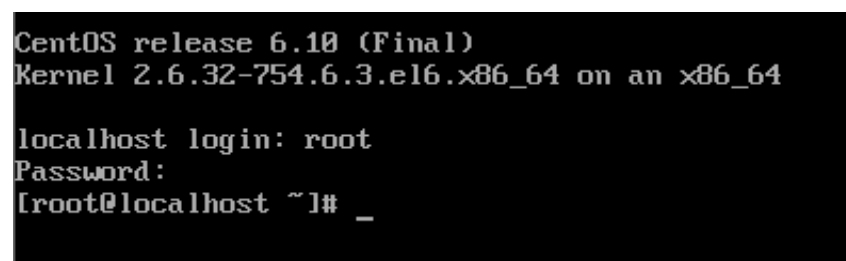
5. Kemudian lakukan reboot dengan perintah “reboot”, pada ketiga mesin tersebut.

```
[root@localhost Desktop]# reboot|
```

6. Jika sudah selesai melakukan reboot, maka tampilan pada node master, node1, dan node2 akan seperti pada gambar berikut :



7. Lalu lakukan login sebagai root pada ketiga mesin.



8. Kemudian lakukan pengecekan alamat IP pada ketiga mesin, dengan perintah ifconfig :

- a. Pengecekan alamat IP pada mesin master, didapat memiliki alamat IP yaitu : **192.168.150.132.**

```
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:8A:98:7D
          inet addr:192.168.150.132  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe8a:987d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:182 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13695 (13.3 KiB)  TX bytes:2836 (2.7 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

- b. Pengecekan alamat IP pada mesin node, didapat memiliki alamat IP yaitu : **192.168.150.136.**

```
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:3C:D1:18
          inet addr:192.168.150.136  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe3c:d118/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:153 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11109 (10.8 KiB)  TX bytes:2764 (2.6 KiB)

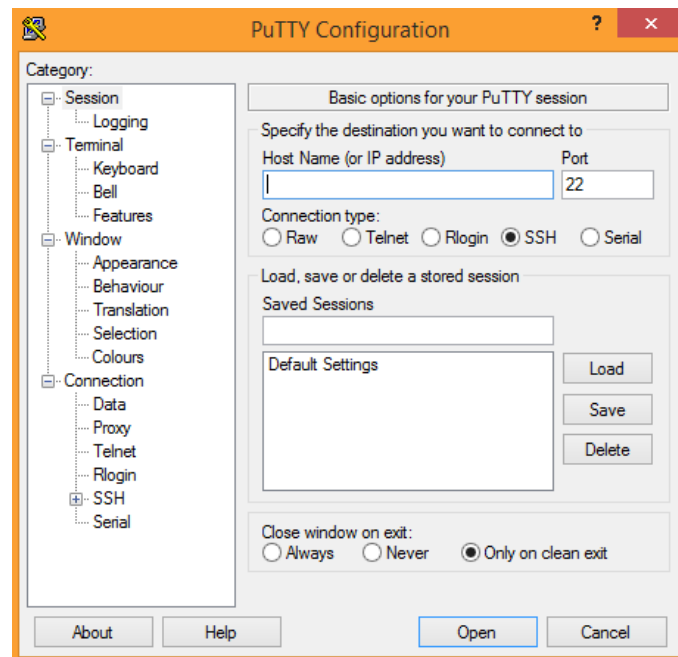
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

- c. Pengecekan alamat IP pada mesin node2, didapat memiliki alamat IP yaitu : **192.168.150.137.**

```
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:43:A1:62
          inet addr:192.168.150.137  Bcast:192.168.150.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe43:a162/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:135 errors:0 dropped:0 overruns:0 frame:0
          TX packets:25 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9364 (9.1 KiB)  TX bytes:2868 (2.8 KiB)

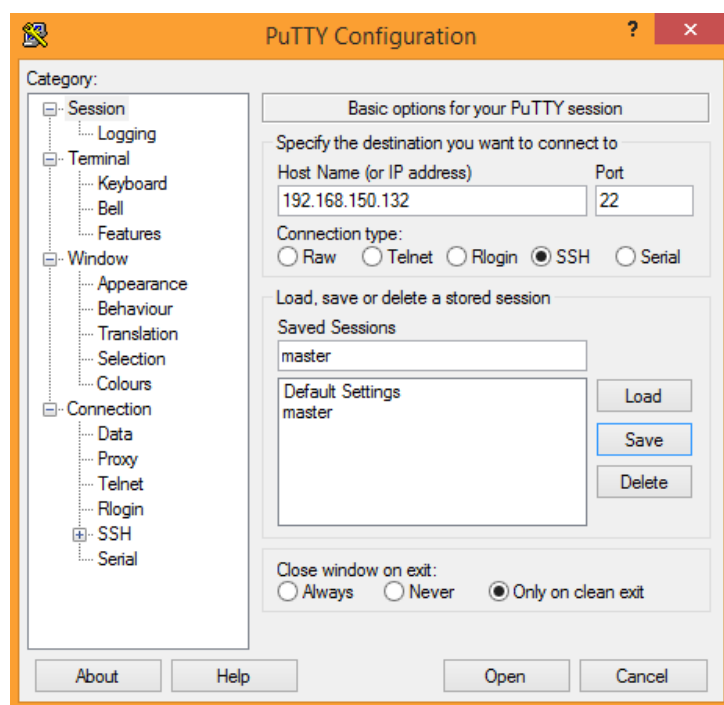
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

9. Setelah mendapatkan alamat IP ketiga mesin, jalankan aplikasi PuTTY.

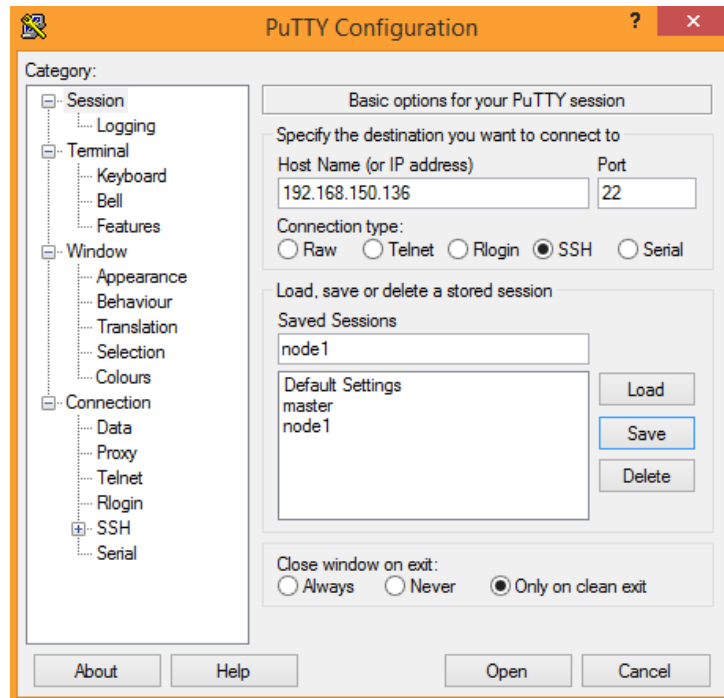


10. Daftarkan alamat IP dari ketiga mesin tersebut :

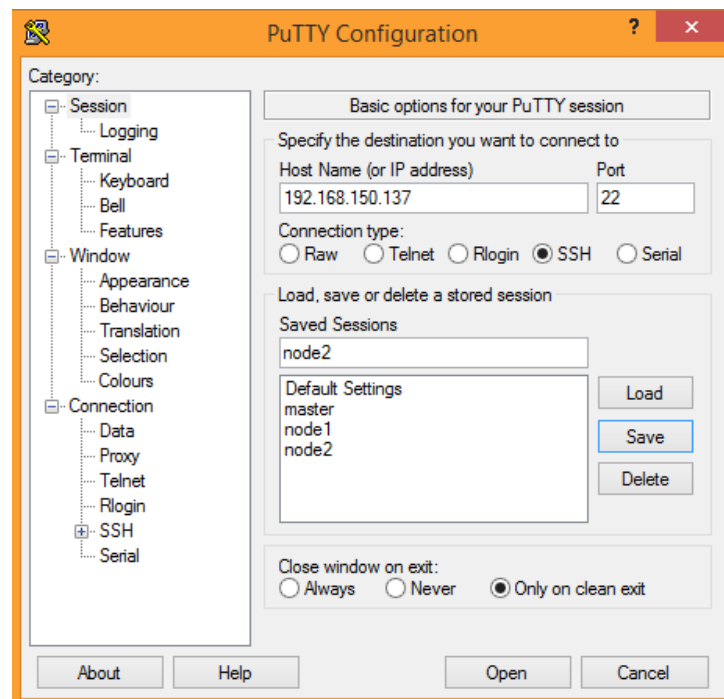
- a. Daftarkan alamat IP dari mesin master yaitu 192.168.150.132, kemudian Save, dan klik Open.



- b. Daftarkan alamat IP dari mesin node1 yaitu 192.168.150.136, kemudian Save, dan klik Open.



- c. Daftarkan alamat IP dari mesin master yaitu 192.168.150.137, kemudian Save, dan klik Open.



11. Jika semua mesin pada puTTY sudah dijalankan, maka lakukan login kembali sebagai root pada ketiga session yang ada :

```
login as: root
root@192.168.150.132's password:
Last login: Tue Nov  6 03:41:35 2018
[root@localhost ~]#
```

```
login as: root
root@192.168.150.136's password:
Last login: Tue Nov  6 03:43:03 2018
[root@localhost ~]#
```

```
login as: root
root@192.168.150.137's password:
Last login: Tue Nov  6 03:43:10 2018
[root@localhost ~]#
```

12. Lalu bukalah file network dengan perintah :

```
[root@localhost ~]# nano /etc/sysconfig/network
```

13. Lakukan konfigurasi pada file network di semua mesin :

- a. Pada mesin master, berilah nama HOSTNAME=master.hadoop.com

```
GNU nano 2.0.9      File: /etc/sysconfig/network      Modified
NETWORKING=yes
HOSTNAME=master.hadoop.com
```

- b. Pada mesin node1, berilah nama HOSTNAME=node1.hadoop.com

```
GNU nano 2.0.9      File: /etc/sysconfig/network      Modified
NETWORKING=yes
HOSTNAME=node1.hadoop.com
```

- c. Pada mesin node2, berilah nama HOSTNAME=node2.hadoop.com

```
GNU nano 2.0.9      File: /etc/sysconfig/network      Modified
NETWORKING=yes
HOSTNAME=node2.hadoop.com
```

Jika sudah melakukan konfigurasi nama HOSTNAME, lalu keluar dengan perintah Ctrl+X, kemudian lakukan penyimpanan dengan menekan “Y”.

14. Setelah itu lakukan, reboot pada ketiga mesin tersebut :

```
[root@localhost ~]# reboot
```

15. Jalankan kembali aplikasi puTTY dan lakukan login kembali sebagai root pada ketiga mesin.

- a. Maka dapat dilihat bahwa nama hostname dari mesin master telah berubah menjadi @master.

```
login as: root
root@192.168.150.132's password:
Last login: Tue Nov  6 04:03:57 2018 from 192.168.150.1
[root@master ~]#
```

- b. Maka dapat dilihat bahwa nama hostname dari mesin node1 telah berubah menjadi @node1

```
login as: root
root@192.168.150.136's password:
Last login: Tue Nov  6 04:01:51 2018 from 192.168.150.1
[root@node1 ~]#
```

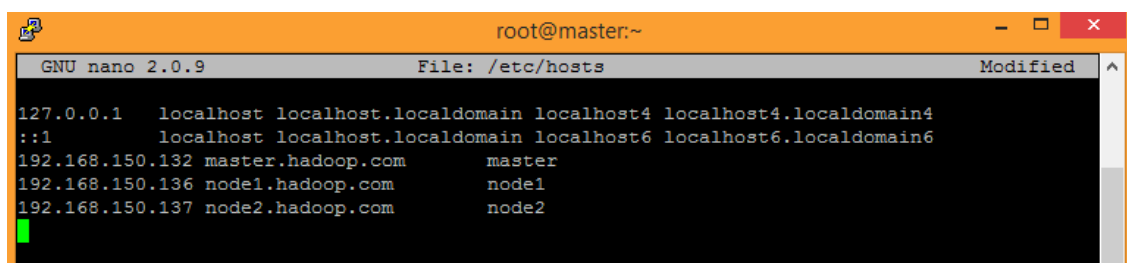
- c. Maka dapat dilihat bahwa nama hostname dari mesin node2 telah berubah menjadi @node2

```
login as: root
root@192.168.150.137's password:
Last login: Tue Nov  6 04:02:02 2018 from 192.168.150.1
[root@node2 ~]#
```

16. Setelah itu bukalah file hosts untuk mendaftarkan host antara mesin, supaya dapat saling terhubung, dengan sebagai berikut :

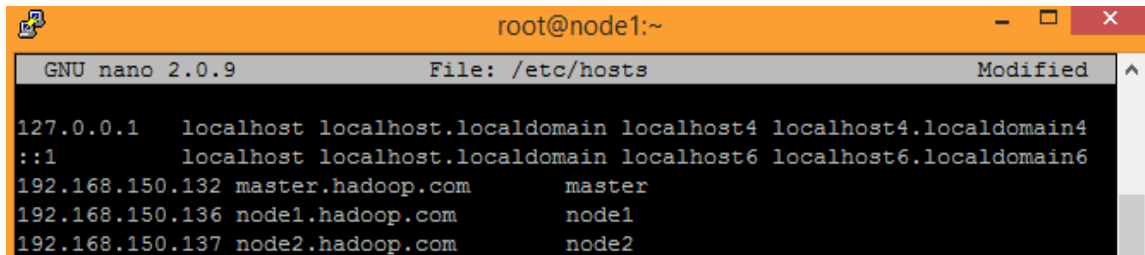
- a. Pada mesin master, daftarkanlah host dengan nama hostname dan alamat IP masing-masing mesin yang ada.

```
[root@master ~]# nano /etc/hosts
```



- b. Pada mesin node1, daftarkanlah host dengan nama hostname dan alamat IP masing-masing mesin yang ada.

```
[root@node1 ~]# nano /etc/hosts
```

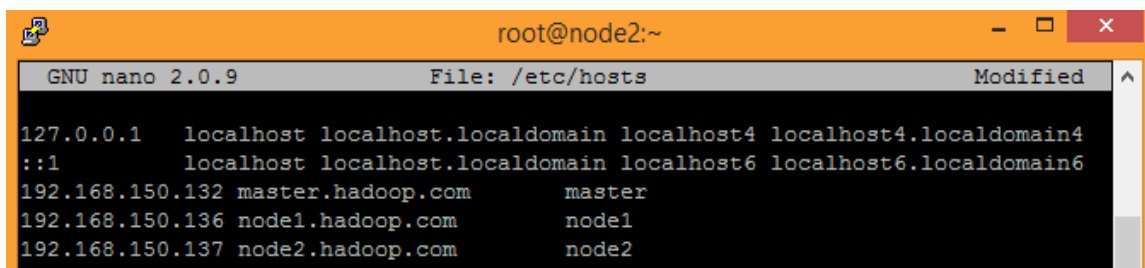


The screenshot shows a terminal window titled 'root@node1:~' with the nano 2.0.9 editor open to the file /etc/hosts. The file contains the following entries:

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.150.132 master.hadoop.com      master
192.168.150.136 node1.hadoop.com       node1
192.168.150.137 node2.hadoop.com       node2
```

- c. Pada mesin node2, daftarkanlah host dengan nama hostname dan alamat IP masing-masing mesin yang ada.

```
[root@node2 ~]# nano /etc/hosts
```



The screenshot shows a terminal window titled 'root@node2:~' with the nano 2.0.9 editor open to the file /etc/hosts. The file contains the following entries:

```
127.0.0.1    localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.150.132 master.hadoop.com      master
192.168.150.136 node1.hadoop.com       node1
192.168.150.137 node2.hadoop.com       node2
```

17. Untuk memastikan ketiga mesin sudah saling terhubung maka dapat dilakukan pengecekan dengan melakukan ping antara satu mesin dengan mesin yang lain.

- a. Melakukan ping dari mesin master.

```
[root@master ~]# ping node1
PING node1.hadoop.com (192.168.150.136) 56(84) bytes of data.
64 bytes from node1.hadoop.com (192.168.150.136): icmp_seq=1 ttl=64 time=1.44 ms
64 bytes from node1.hadoop.com (192.168.150.136): icmp_seq=2 ttl=64 time=3.66 ms
64 bytes from node1.hadoop.com (192.168.150.136): icmp_seq=3 ttl=64 time=0.677 ms
^C
--- node1.hadoop.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2392ms
rtt min/avg/max/mdev = 0.677/1.926/3.662/1.267 ms
[root@master ~]# ping node2
PING node2.hadoop.com (192.168.150.137) 56(84) bytes of data.
64 bytes from node2.hadoop.com (192.168.150.137): icmp_seq=1 ttl=64 time=3.65 ms
64 bytes from node2.hadoop.com (192.168.150.137): icmp_seq=2 ttl=64 time=0.640 ms
^C
--- node2.hadoop.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.640/2.149/3.659/1.510 ms
```


- b. Melakukan ping dari mesin node1.

```
[root@node1 ~]# ping master
PING master.hadoop.com (192.168.150.132) 56(84) bytes of data.
64 bytes from master.hadoop.com (192.168.150.132): icmp_seq=1 ttl=64 time=0.795 ms
64 bytes from master.hadoop.com (192.168.150.132): icmp_seq=2 ttl=64 time=0.837 ms
^C
--- master.hadoop.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1855ms
rtt min/avg/max/mdev = 0.795/0.816/0.837/0.021 ms
[root@node1 ~]# ping node2
PING node2.hadoop.com (192.168.150.137) 56(84) bytes of data.
64 bytes from node2.hadoop.com (192.168.150.137): icmp_seq=1 ttl=64 time=2.20 ms
64 bytes from node2.hadoop.com (192.168.150.137): icmp_seq=2 ttl=64 time=0.644 ms
^C
--- node2.hadoop.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1988ms
rtt min/avg/max/mdev = 0.644/1.426/2.208/0.782 ms
```

- c. Melakukan ping dari mesin node2.

```
[root@node2 ~]# ping master
PING master.hadoop.com (192.168.150.132) 56(84) bytes of data.
64 bytes from master.hadoop.com (192.168.150.132): icmp_seq=1 ttl=64 time=0.758 ms
64 bytes from master.hadoop.com (192.168.150.132): icmp_seq=2 ttl=64 time=0.669 ms
64 bytes from master.hadoop.com (192.168.150.132): icmp_seq=3 ttl=64 time=0.669 ms
^C
--- master.hadoop.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2068ms
rtt min/avg/max/mdev = 0.669/0.698/0.758/0.051 ms
[root@node2 ~]# ping node1
PING node1.hadoop.com (192.168.150.136) 56(84) bytes of data.
64 bytes from node1.hadoop.com (192.168.150.136): icmp_seq=1 ttl=64 time=0.646 ms
64 bytes from node1.hadoop.com (192.168.150.136): icmp_seq=2 ttl=64 time=1.34 ms
^C
--- node1.hadoop.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1893ms
rtt min/avg/max/mdev = 0.646/0.997/1.348/0.351 ms
```

18. Jika semua mesin sudah saling terhubung, kemudian salinlah text berikut :

```
echo "echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled" >> /etc/rc.local
echo "echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag" >> /etc/rc.local
echo "vm.swappiness = 10" >> /etc/sysctl.conf
chkconfig iptables off
chkconfig ip6tables off
sed -i 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config
yum -y install ntp
chkconfig ntpd on
```

Lalu tuliskan perintah di atas pada terminal pada ketiga mesin yaitu master, node1, dan node2. Seperti pada gambar berikut :

```
[root@master ~]# echo "echo never > /sys/kernel/mm/redhat_transparent_hugepage/enabled" >> /etc/rc.local
[root@master ~]# echo "echo never > /sys/kernel/mm/redhat_transparent_hugepage/defrag" >> /etc/rc.local
[root@master ~]# echo "vm.swappiness = 10" >> /etc/sysctl.conf
[root@master ~]# chkconfig iptables off
[root@master ~]# chkconfig ip6tables off
[root@master ~]# sed -i 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config
[root@master ~]# yum -y install ntp
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Install Process
Loading mirror speeds from cached hostfile
* base: mirror.biznetgio.com
* extras: mirror.biznetgio.com
* updates: mirror.biznetgio.com
Package ntp-4.2.6p5-12.el6.centos.2.x86_64 already installed and latest version
Nothing to do
```

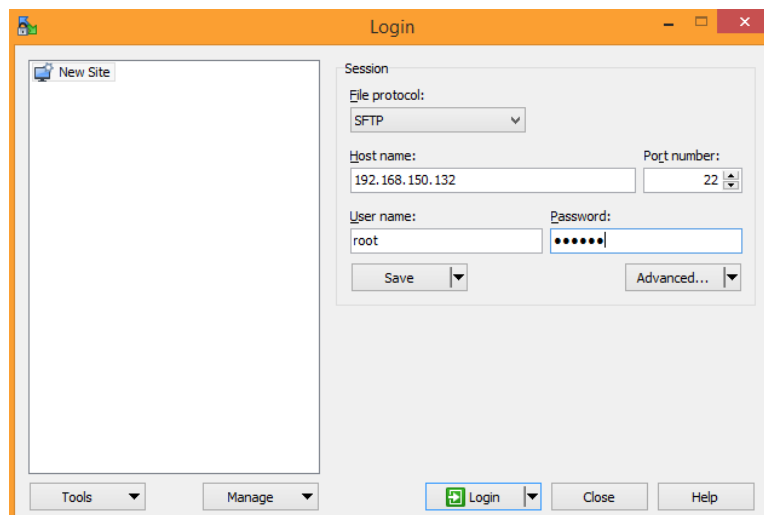
19. Download file JDK pada tautan berikut :

<https://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html>

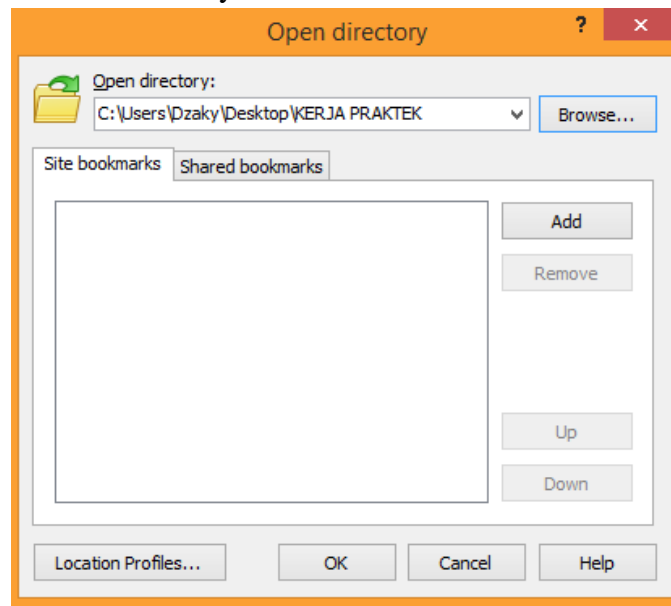
Java SE Development Kit 8u181		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	72.95 MB	jdk-8u181-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.89 MB	jdk-8u181-linux-arm64-vfp-hflt.tar.gz
Linux x86	165.06 MB	jdk-8u181-linux-i586.rpm
Linux x86	179.87 MB	jdk-8u181-linux-i586.tar.gz
Linux x64	162.15 MB	jdk-8u181-linux-x64.rpm
Linux x64	177.05 MB	jdk-8u181-linux-x64.tar.gz
Mac OS X x64	242.83 MB	jdk-8u181-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	133.17 MB	jdk-8u181-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	94.34 MB	jdk-8u181-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	133.83 MB	jdk-8u181-solaris-x64.tar.Z
Solaris x64	92.11 MB	jdk-8u181-solaris-x64.tar.gz
Windows x86	194.41 MB	jdk-8u181-windows-i586.exe
Windows x64	202.73 MB	jdk-8u181-windows-x64.exe

20. Apabila proses download telah selesai, bukalah aplikasi winSCP untuk menyalin file JDK yang telah didownload sebelumnya pada ketiga mesin yang digunakan yaitu master, node1, dan node2. Dengan langkah-langkah sebagai berikut :

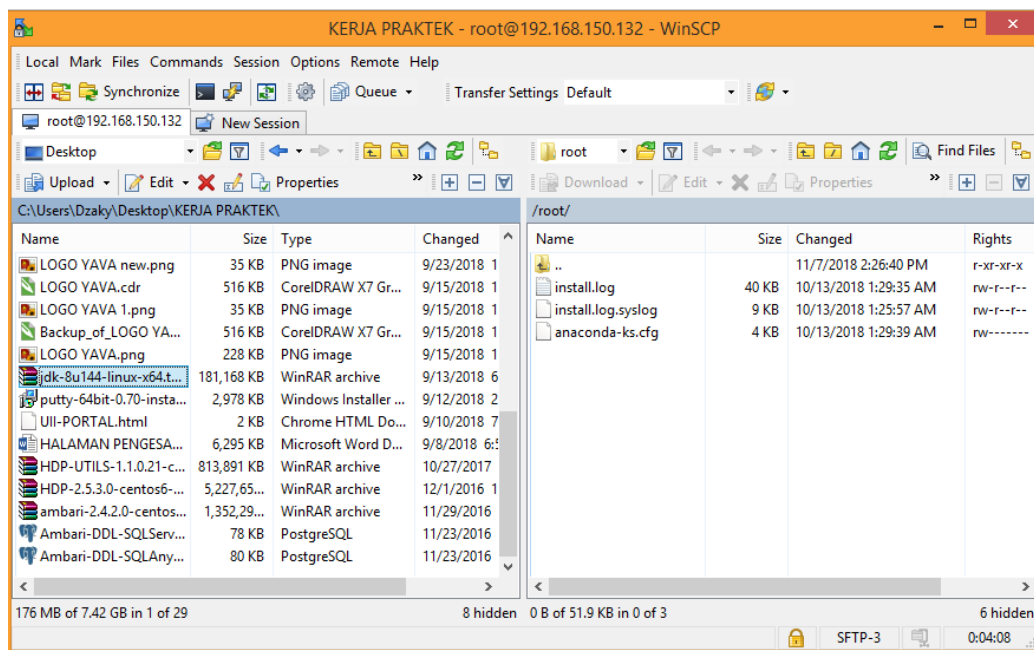
- masukkan alamat IP mesin master beserta username dan password yang digunakan, seperti pada gambar berikut :



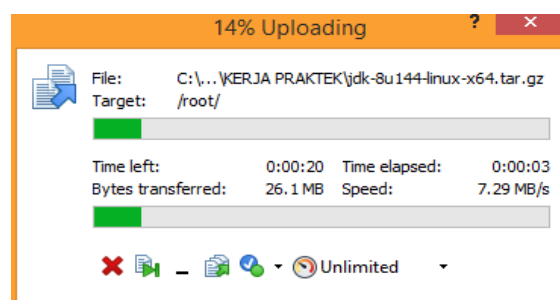
- b. Lalu bukalah folder atau direktori tempat dimana file JDK yang telah didownload sebelumnya.



“Klik Browese > Pilih Folder tempat dimana file JDK disimpan > Klik OK”



- c. Lakukan penyalinan file JDK kepada folder /root/ pada mesin master.



- d. Setelah selesai melakukan penyalinan file, kita dapat melihat file yang sudah disalin pada direktori /root/ di mesin master dengan perintah ls, seperti pada gambar di bawah :

```
[root@master ~]# ls
anaconda-ks.cfg  install.log  install.log.syslog  jdk-8u144-linux-x64.tar.gz
```

- e. Kemudian buatlah direktori dengan perintah mkdir -p /usr/java/

```
[root@master ~]# mkdir -p /usr/java/
```

- f. Lakukan unzip atau ekstrak file jdk dan letakan hasil ekstrak file tersebut pada direktori /usr/java/ yang telah dibuat dengan perintah berikut :

```
[root@master ~]# tar zxvf jdk-8u144-linux-x64.tar.gz -C /usr/java/
```

- g. Lakukan hal yang sama pada mesin node1 dan node2.

21. Jika pada semua mesin selesai melakukan ekstraksi file JDK, maka kita dapat melihat isi dari direktori /usr/java/, seperti pada gambar berikut :

```
[root@master ~]# ls /usr/java/jdk1.8.0_144/bin/
appletviewer  javac      javaws     jinfo      jsadebugd   orbd       serialver
ControlPanel  javadoc    jcmd       jjs        jstack      pack200    servertool
extcheck      javafxpackager  jconsole   jmap       jstat       policytool  tnameserv
idlj          javah      jcontrol   jmc        jstatd      rmic       unpack200
jar           javap      jdbc        jmc.ini    jvisualvm   rmid       wsgen
jarsigner     javapackager  jdeps      jps        keytool     rmiregistry wsimport
java          java-rmi.cgi  jhat       jrunscript native2ascii schemagen   xjc
```

Bagian II - Instalasi Ambari

1. Download file https://docs.hortonworks.com/HDPDocuments/Ambari-2.6.2.0/bk_ambari-installation/content/hdp_25_repositories.html

HDP 2.5 Repositories

OS	Version Number	Repository Name	Format	URL
RedHat 6 CentOS 6 Oracle Linux 6	HDP-2.5.3.0	HDP	Version Definition File (VDF)	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.5.3.0/HDP-2.5.3.0-37.xml
			Base URL	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.5.3.0
			Repo File	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.5.3.0/hdp.repo
			Tarball md5 asc	http://public-repo-1.hortonworks.com/HDP/centos6/2.x/updates/2.5.3.0/HDP-2.5.3.0-centos6-rpm.tar.gz
		HDP-UTILS	Base URL	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos6
			Tarball md5 asc	http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.21/repos/centos6/HDP-UTILS-1.1.0.21-centos6.tar.gz

2. Kemudian lakukanlah instalasi httpd jika belum pernah menginstall sebelumnya, dengan perintah berikut :

```
[root@master ~]# yum -y install httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Install Process
Loading mirror speeds from cached hostfile
* base: mirror.biznetgio.com
* extras: mirror.biznetgio.com
* updates: mirror.biznetgio.com
```

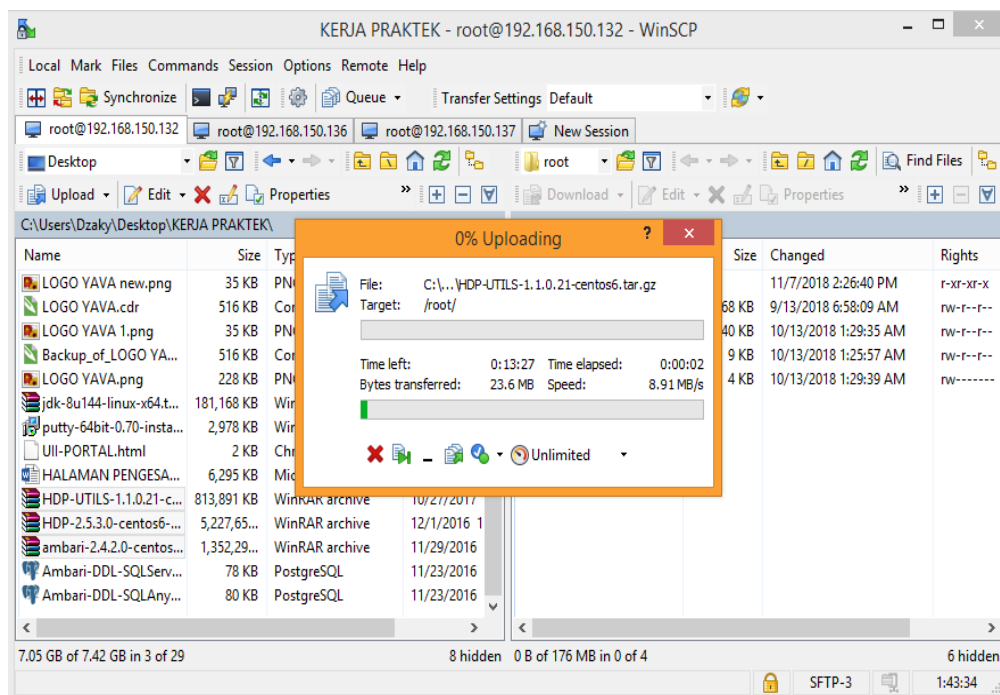
Kemudian ketikkan perintah berikut :

```
[root@master ~]# chkconfig httpd on
```

3. Setelah itu jalankan service httpd dengan perintah berikut :

```
[root@master ~]# service httpd start
Starting httpd: [ OK ]
```

4. Salin 2 file yang telah di download pada direktori /root/ di ketiga mesin yaitu, master, node1 dan node2.



Kemudian Lakukan pengecekan file yang telah disalin dengan perintah ls, untuk memastikan file sudah berhasil disalin :

```
[root@master ~]# ls
ambari-2.4.2.0-centos6.tar.gz  HDP-UTILS-1.1.0.21-centos6.tar.gz  jdk-8u144-linux-x64.tar.gz
anaconda-ks.cfg              install.log
HDP-2.5.3.0-centos6-rpm.tar.gz  install.log.syslog
```

5. Lakukan ekstrak file ambari-2.4.2.0 ke dalam folder /var/www/html/

```
[root@master ~]# tar zxvf ambari-2.4.2.0-centos6.tar.gz -C /var/www/html/
```

6. Kemudian lakukan ekstrasi file HDP-2.5.3.0, ke dalam folder /var/www/html/.

```
[root@master ~]# ls
ambari-2.4.2.0-centos6.tar.gz  HDP-UTILS-1.1.0.21-centos6.tar.gz  jdk-8u144-linux-x64.tar.gz
anaconda-ks.cfg               install.log
HDP-2.5.3.0-centos6-rpm.tar.gz  install.log.syslog
[root@master ~]# tar zxvf HDP-2.5.3.0-centos6-rpm.tar.gz -C /var/www/html/
```

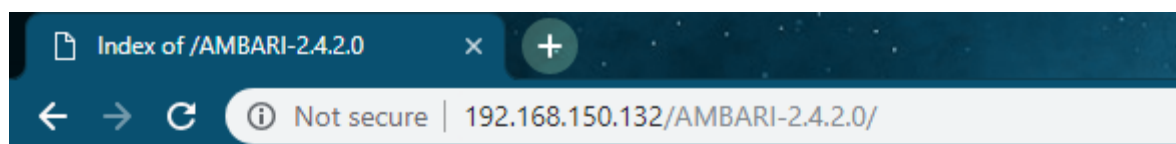
7. Buatlah folder “HDP-UTILS” dengan perintah mkdir, pada path /var/www/html/, kemudian lakukan ekstrasi file HDP-UTILS-1.1.0.21-centos6.tar.gz seperti gambar berikut :

```
[root@master ~]# tar zxvf HDP-UTILS-1.1.0.21-centos6.tar.gz -C /var/www/html/HDP-UTILS
epel-release-6-8.noarch.rpm
extjs/
extjs/extjs-2.2-1.noarch.rpm
fping/
```

8. Kemudian lakukan reboot :

```
[root@master ~]# reboot
```

9. Setelah selesai reboot, bukalah browser lewat komputer utama anda dan ketikkan url dengan alamat IP mesin master seperti gambar berikut :



Index of /AMBARI-2.4.2.0

Name	Last modified	Size	Description
Parent Directory		-	
OSL.txt	22-Nov-2016 23:55	530K	
centos6/	22-Nov-2016 23:55	-	
setup_repo.sh	22-Nov-2016 23:55	5.6K	

Apache/2.2.15 (CentOS) Server at 192.168.150.132 Port 80

Lalu salinlah path url berikut : <http://192.168.150.132/AMBARI-2.4.2.0/centos6/2.4.2.0-136/>.

10. Gantilah direktori pada terminal mesin master,

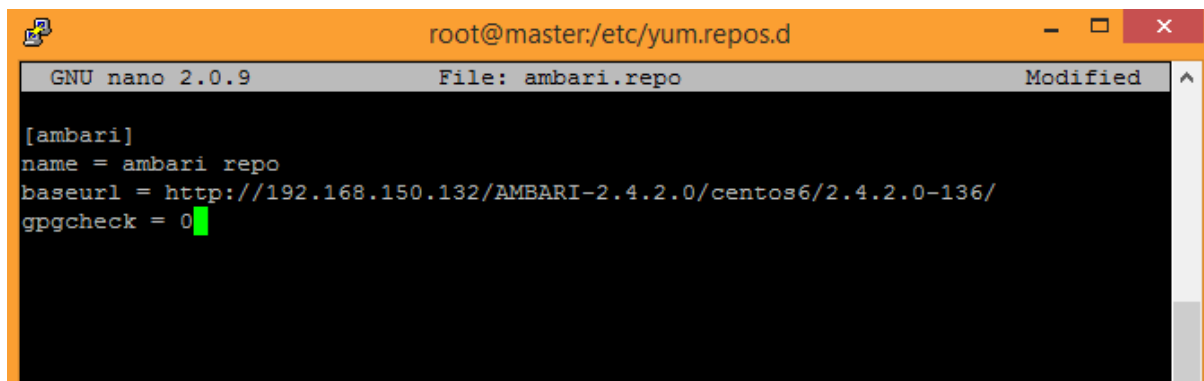
```
[root@master ~]# cd /etc/yum.repos.d/
[root@master yum.repos.d]#
```

11. Lalu di dalam direktori tersebut, buatlah file dengan nama ambari.repo dengan perintah nano.

```
[root@master yum.repos.d]# nano ambari.repo
```

```
[ambari]
name = ambari repo
baseurl = http://[Alamat IP Master]/AMBARI-2.4.2.0/centos6/2.4.2.0-136/
gpgcheck = 0
```

Kemudian ketiklah tulisan pada kotak di atas dan di dalam file ambari.repo tersebut, seperti pada gambar di bawah dan sesuaikan dengan alamat IP mesin master/servernya, kemudian simpan.



```
root@master:/etc/yum.repos.d
GNU nano 2.0.9      File: ambari.repo      Modified
[ambari]
name = ambari repo
baseurl = http://192.168.150.132/AMBARI-2.4.2.0/centos6/2.4.2.0-136/
gpgcheck = 0
```

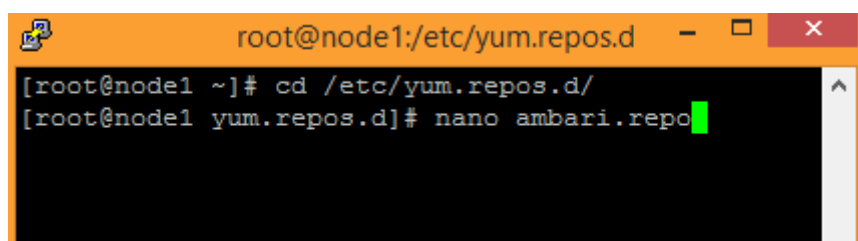
12. Lalu ketiklah perintah yum clean all :

```
[root@master yum.repos.d]# yum clean all
Loaded plugins: fastestmirror, refresh-packagekit, security
Cleaning repos: ambari base extras updates
Cleaning up Everything
Cleaning up list of fastest mirrors
```

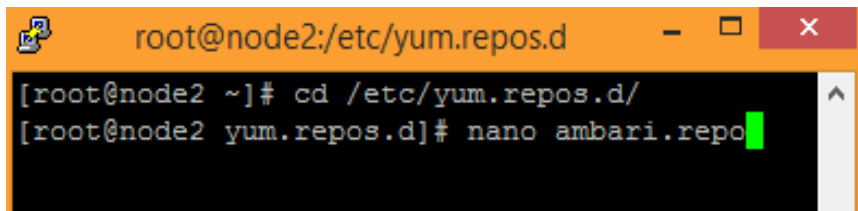
13. Ketiklah perintah yum makecache :

```
[root@master yum.repos.d]# yum makecache
Loaded plugins: fastestmirror, refresh-packagekit, security
Determining fastest mirrors
 * base: kartolo.sby.datautama.net.id
 * extras: kartolo.sby.datautama.net.id
 * updates: kartolo.sby.datautama.net.id
ambari                               | 2.9 kB    00:00
ambari/filelists_db                  | 139 kB    00:00
ambari/primary_db                    | 8.3 kB    00:00
ambari/other_db                      | 1.3 kB    00:00
base                                  | 3.7 kB    00:00
base/group_gz                        | 242 kB    00:00
```

14. Buatlah file ambari.repo kembali, pada mesin node1 dan node2 di direktori /etc/yum.repos.d/ :



```
root@node1:/etc/yum.repos.d
[root@node1 ~]# cd /etc/yum.repos.d/
[root@node1 yum.repos.d]# nano ambari.repo
```

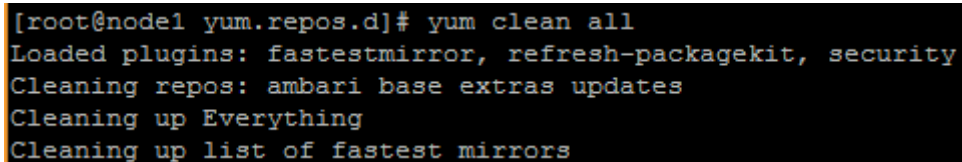



```
root@node2:/etc/yum.repos.d
[root@node2 ~]# cd /etc/yum.repos.d/
[root@node2 yum.repos.d]# nano ambari.repo
```

Lalu di dalam file ambari.repo tersebut, salinlan teks di bawah, kemudian simpan.

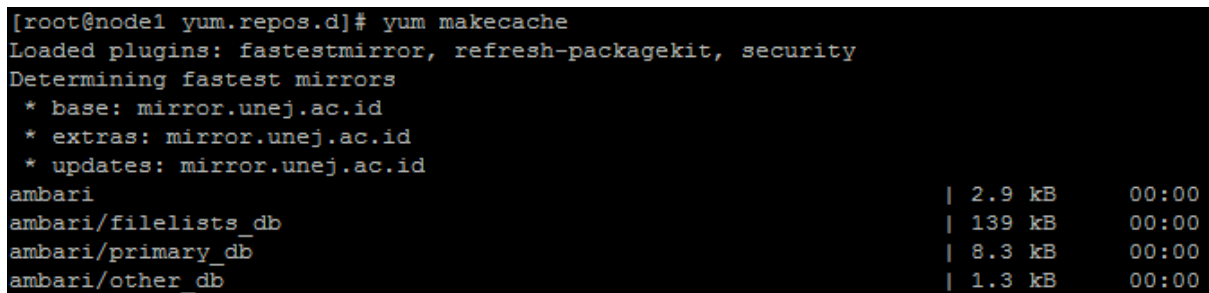
```
[ambari]
name = ambari repo
baseurl = http://[Alamat IP Master]/AMBARI-2.4.2.0/centos6/2.4.2.0-136/
gpgcheck = 0
```

Kemudian, ketikkan perintah yum clean all pada terminal mesin node1 dan node2 :



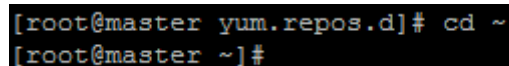
```
[root@node1 yum.repos.d]# yum clean all
Loaded plugins: fastestmirror, refresh-packagekit, security
Cleaning repos: ambari base extras updates
Cleaning up Everything
Cleaning up list of fastest mirrors
```

Jika proses clean all sudah selesai, maka ketiklah perintah yum makecache pada terminal mesin node1 dan node2 :



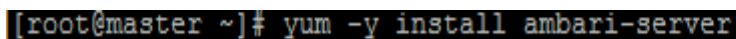
```
[root@node1 yum.repos.d]# yum makecache
Loaded plugins: fastestmirror, refresh-packagekit, security
Determining fastest mirrors
 * base: mirror.unej.ac.id
 * extras: mirror.unej.ac.id
 * updates: mirror.unej.ac.id
ambari | 2.9 kB 00:00
ambari/filelists_db | 139 kB 00:00
ambari/primary_db | 8.3 kB 00:00
ambari/other_db | 1.3 kB 00:00
```

15. Kemudian pada terminal mesin master atau server, ketikkan perintah `cd ~` untuk kembali ke direktori `/root/` :



```
[root@master yum.repos.d]# cd ~
[root@master ~]#
```

16. Lakukan instalasi ambari-server hanya pada mesin master dengan perintah seperti pada gambar di bawah, dan tungguilah beberapa saat hingga proses instalasi selesai.



```
[root@master ~]# yum -y install ambari-server
```


17. Apabila proses instalasi ambari-server sudah selesai, maka langkah selanjutnya adalah melakukan instalasi ambari-agent pada mesin master, node1 dan node2, dengan perintah seperti pada gambar berikut di semua mesin :

```
[root@master ~]# yum -y install ambari-agent
```

```
[root@node1 yum.repos.d]# cd ~  
[root@node1 ~]# yum -y install ambari-agent
```

```
[root@node2 yum.repos.d]# cd ~  
[root@node2 ~]# yum -y install ambari-agent
```

18. Lakukan konfigurasi pada file ambari-agent.ini di mesin master, node1 dan node2.

```
[root@node1 ~]# nano /etc/ambari-agent/conf/ambari-agent.ini
```

```
[root@node2 ~]# nano /etc/ambari-agent/conf/ambari-agent.ini
```

19. Lalu ubahlah isi hostname dengan hostname milik mesin master karena sebagai server, yang memiliki hostname yaitu master.hadoop.com

```
GNU nano 2.0.9      File: /etc/ambari-agent/conf/ambari-agent.ini      Modified  
  
# Unless required by applicable law or agreed to in writing, software  
# distributed under the License is distributed on an "AS IS" BASIS,  
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
# See the License for the specific  
  
[server]  
hostname=master.hadoop.com  
url_port=8440  
secured_url_port=8441
```

20. Lalu jalankan setup ambari-server.

```
[root@master ~]# ambari-server setup  
Using python /usr/bin/python  
Setup ambari-server
```

Pada saat setup, langkah pertama adalah pengecekan SELinux, untuk defaultnya maka klik enter atau pilih n :

```
Using python /usr/bin/python  
Setup ambari-server  
Checking SELinux...  
SELinux status is 'disabled'  
Customize user account for ambari-server daemon [y/n] (n)?  
Adjusting ambari-server permissions and ownership...  
Checking firewall status...
```

Kemudian pilih pilihan (1) atau enter, untuk memilih JDK secara default.

```
Checking JDK...
[1] Oracle JDK 1.8 + Java Cryptography Extension (JCE) Policy Files 8
[2] Oracle JDK 1.7 + Java Cryptography Extension (JCE) Policy Files 7
[3] Custom JDK
=====
Enter choice (1):
To download the Oracle JDK and the Java Cryptography Extension (JCE) Policy Files you must
accept the license terms found at http://www.oracle.com/technetwork/java/javase/terms/1
license/index.html and not accepting will cancel the Ambari Server setup and you must inst
all the JDK and JCE files manually.
```

Kemudian ketik y atau enter, untuk accept agreement.

```
Do you accept the Oracle Binary Code License Agreement [y/n] (y)?
Downloading JDK from http://public-repo-1.hortonworks.com/ARTIFACTS/jdk-8u77-linux-x64.tar.gz to /var/lib/amb
ari-server/resources/jdk-8u77-linux-x64.tar.gz
```

Klik n, untuk memilih database yang digunakan secara default yaitu PostgreSQL.

```
Enter advanced database configuration [y/n] (n)?
Configuring database...
Default properties detected. Using built-in database.
Configuring ambari database...
Checking PostgreSQL...
Running initdb: This may take up to a minute.
Initializing database: [ OK ]

About to start PostgreSQL
Configuring local database...
Connecting to local database...connection timed out...retrying (1)
Connecting to local database...done.
Configuring PostgreSQL...
Restarting PostgreSQL
Extracting system views...
.....ambari-admin-2.4.2.0.136.jar
.....
Adjusting ambari-server permissions and ownership...
Ambari Server 'setup' completed successfully.
```

21. Jalankan ambari-server pada terminal master dengan perintah seperti berikut :

```
[root@master ~]# service ambari-server start
```

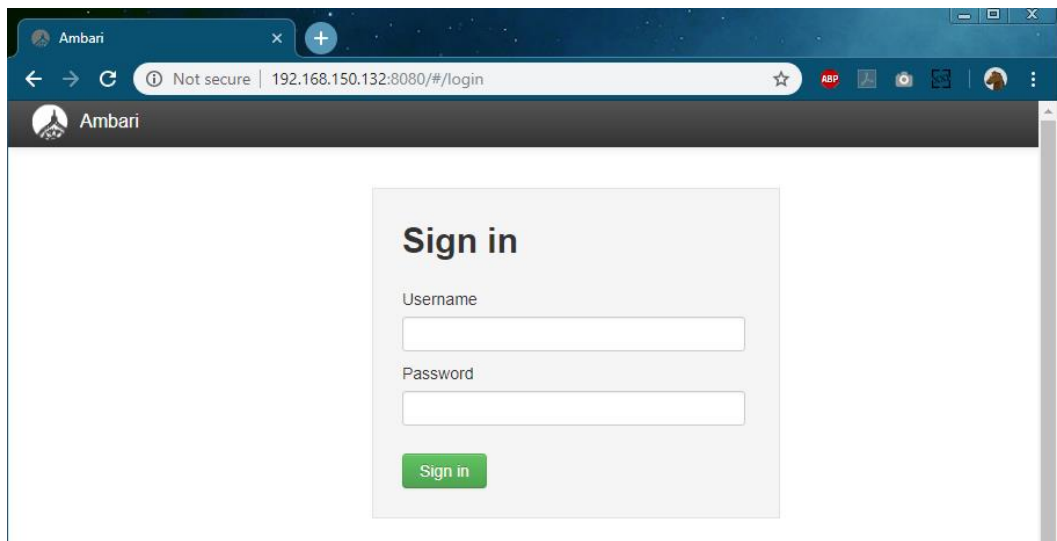
22. Jalankan ambari-agent pada terminal di mesin master, node1, dan node2 dengan perintah “service ambari-agent start” :

```
[root@master ~]# service ambari-agent start
```

```
[root@node1 ~]# service ambari-agent start
```

```
[root@node2 ~]# service ambari-agent start
```

23. Setelah semua ambari-agent dan ambari-server dijalankan, bukalah browser dan masukkan alamat IP dari mesin master yang terinstall dengan ambari-server, seperti pada gambar berikut :



Dengan demikian, cara untuk menginstall proses setup ambari-server dan ambari-agent.

Bagian III – Membuat Cluster

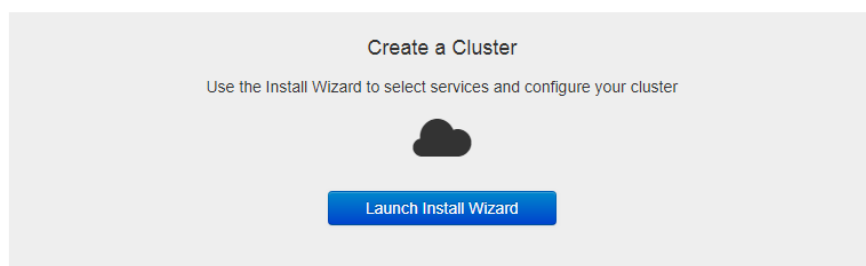
1. Bukalah browser lalu login dengan user yang sudah terbuat pada saat melakukan setup ambari di proses sebelumnya. Masukkan username "admin" dan password "admin". Kemudian klik "Sign in"



2. Untuk membuat cluster baru, klik tombol "Launch Install Wizard".

Welcome to Apache Ambari

Provision a cluster, manage who can access the cluster, and customize views for Ambari users.



3. Isikan nama cluster yang dibuat misalkan “java_01”, kemudian klik next.

Get Started

This wizard will walk you through the cluster installation process. First, start by naming your new cluster.

Name your cluster [Learn more](#)

java_01

Next →

4. Gunakanlah versi HDP-2.5.

Select Version

Select the software version and method of delivery for your cluster. Using a Public Repository requires Internet connectivity. Using a Local Repository requires you have configured the software in a repository available in your network.

HDP-2.5

HDP-2.5.3.0




Kemudian klik menggunakan “local repository”, dan remove atau hapus sistem operasi selain redhat6 (karena dalam dokumentasi ini menggunakan OS CentOS 6). Lalu masukkan Base URL, sesuai dengan dimana kita telah meengstrak file HDP dan HDP-UTILS sebelumnya. Kemudian klik next.

☐ Use Public Repository

☒ Use Local Repository

Repositories

Provide Base URLs for the Operating Systems you are configuring.

OS	Name	Base URL	
redhat6	HDP-2.5	<input type="text" value="http://192.168.150.132/HDP/centos6/"/>	 
	HDP-UTILS-1.1.0.21	<input type="text" value="http://192.168.150.132/HDP-UTILS/"/>	 

☐ Skip Repository Base URL validation (Advanced) ?

☐ Use RedHat Satellite/Spacewalk ?

← Back

Next →

5. Pada tahapan Install Options, isilah target hosts berdasarkan hostname yang sudah dikonfigurasi sebelumnya pada mesin master, node1, dan node2.

Install Options

Enter the list of hosts to be included in the cluster and provide your SSH key.

Target Hosts

Enter a list of hosts using the Fully Qualified Domain Name (FQDN), one per line. Or use [Pattern Expressions](#)

master.hadoop.com
node1.hadoop.com
node2.hadoop.com

Karena pada proses pembuatan cluster ini juga belum memiliki SSH, maka pilih “Perform manual registration...” lalu klik Register and Confirm →.

Host Registration Information

☐ Provide your [SSH Private Key](#) to automatically register hosts

No file chosen

SSH User Account

SSH Port Number

☒ Perform [manual registration](#) on hosts and do not use SSH

Karena pada proses sebelumnya juga sudah melakukan instalasi ambari-agent maka langsung saja klik OK.

Before You Proceed ✕

You must install Ambari Agents on each host you want to manage before you proceed.

6. Maka tungguilah beberapa saat hingga semua Host sudah terdaftar, kemudian klik Next.

Confirm Hosts

Registering your hosts.
Please confirm the host list and remove any hosts that you do not want to include in the cluster.

	Host	Progress	Status	Action
<input type="checkbox"/>	master.hadoop.com	<div></div>	Success	<input type="button" value="Remove"/>
<input type="checkbox"/>	node1.hadoop.com	<div></div>	Success	<input type="button" value="Remove"/>
<input type="checkbox"/>	node2.hadoop.com	<div></div>	Success	<input type="button" value="Remove"/>

Show: **All (3)** | [Installing \(0\)](#) | [Registering \(0\)](#) | [Success \(3\)](#) | [Fail \(0\)](#)

Show: 25 1 - 3 of 3

Please wait while the hosts are being checked for potential problems...

7. Lalu pilihlah services yang akan diinstal, misalkan beberapa service seperti pada gambar berikut :

Choose Services

Choose which services you want to install on your cluster.

<input type="checkbox"/> Service	Version	Description
<input checked="" type="checkbox"/> HDFS	2.7.3	Apache Hadoop Distributed File System
<input checked="" type="checkbox"/> YARN + MapReduce2	2.7.3	Apache Hadoop NextGen MapReduce (YARN)
<input type="checkbox"/> Tez	0.7.0	Tez is the next generation Hadoop Query Processing framework written on top of YARN.
<input type="checkbox"/> Hive	1.2.1000	Data warehouse system for ad-hoc queries & analysis of large datasets and table & storage management service
<input type="checkbox"/> HBase	1.1.2	A Non-relational distributed database, plus Phoenix, a high performance SQL layer for low latency applications.
<input type="checkbox"/> Pig	0.16.0	Scripting platform for analyzing large datasets
<input type="checkbox"/> Sqoop	1.4.6	Tool for transferring bulk data between Apache Hadoop and structured data stores such as relational databases
<input type="checkbox"/> Oozie	4.2.0	System for workflow coordination and execution of Apache Hadoop jobs. This also includes the installation of the optional Oozie Web Console which relies on and will install the ExtJS Library .
<input checked="" type="checkbox"/> ZooKeeper	3.4.6	Centralized service which provides highly reliable distributed coordination

Jika sudah memilih services, maka klik Next.

8. Klik next apabila tidak ingin melakukan perubahan.

Assign Masters

Assign master components to hosts you want to run them on.

NameNode:	master.hadoop.com (3.7 GB, 4 c ▾)	master.hadoop.com (3.7 GB, 4 cores) NameNode ZooKeeper Server
SNameNode:	node1.hadoop.com (3.7 GB, 4 c ▾)	
App Timeline Server:	node1.hadoop.com (3.7 GB, 4 c ▾)	node1.hadoop.com (3.7 GB, 4 cores) SNameNode App Timeline Server ResourceManager History Server ZooKeeper Server
ResourceManager:	node1.hadoop.com (3.7 GB, 4 c ▾)	
History Server:	node1.hadoop.com (3.7 GB, 4 c ▾)	
ZooKeeper Server:	master.hadoop.com (3.7 GB, 4 c ▾) -	node2.hadoop.com (3.7 GB, 4 cores) ZooKeeper Server
ZooKeeper Server:	node2.hadoop.com (3.7 GB, 4 c ▾) -	
ZooKeeper Server:	node1.hadoop.com (3.7 GB, 4 c ▾) -	

← Back

Next →

9. Klik next saja, apabila tidak ingin melakukan pengaturan terkait pengaturan Slaves dan Clients.

Assign Slaves and Clients

Assign slave and client components to hosts you want to run them on.
Hosts that are assigned master components are shown with *.
"Client" will install HDFS Client, YARN Client, MapReduce2 Client and ZooKeeper Client.

Host	all none	all none	all none	all none
master.hadoop.com *	<input checked="" type="checkbox"/> DataNode	<input type="checkbox"/> NFSGateway	<input checked="" type="checkbox"/> NodeManager	<input checked="" type="checkbox"/> Client
node1.hadoop.com *	<input checked="" type="checkbox"/> DataNode	<input type="checkbox"/> NFSGateway	<input checked="" type="checkbox"/> NodeManager	<input type="checkbox"/> Client
node2.hadoop.com *	<input checked="" type="checkbox"/> DataNode	<input type="checkbox"/> NFSGateway	<input checked="" type="checkbox"/> NodeManager	<input checked="" type="checkbox"/> Client

Show: 25 1 - 3 of 3

← Back

Next →

10. Lakukan kustomisasi jika diperlukan, apabila tidak maka klik Next.

Customize Services

We have come up with recommended configurations for the services you selected. Customize them as you see fit.

HDFS YARN MapReduce2 ZooKeeper Misc

Group Default (3) Manage Config Groups

Filter...

Settings Advanced

NameNode

NameNode directories

/hadoop/hdfs/namenode

NameNode Java heap size



NameNode Server threads



Minimum replicated blocks %



DataNode

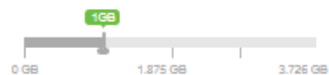
DataNode directories

/hadoop/hdfs/data

DataNode failed disk tolerance



DataNode maximum Java heap size



DataNode max data transfer threads



11. Berikut adalah hasil Review dari hasil proses pembuatan Cluster, kemudian klik Deploy untuk memulai proses penerapan Services pada Cluster yang dibuat.

Review

Please review the configuration before installation

Admin Name : admin

Cluster Name : java_01

Total Hosts : 3 (3 new)

Repositories:

- redhat6 (HDP-2.5):
http://192.168.150.132/HDP/centos6/
- redhat6 (HDP-UTILS-1.1.0.21):
http://192.168.150.132/HDP-UTILS/

Services:

HDFS

- DataNode : 3 hosts
- NameNode : master.hadoop.com
- NFSGateway : 0 host
- SNameNode : node1.hadoop.com

YARN + MapReduce2

- App Timeline Server : node1.hadoop.com
- NodeManager : 3 hosts
- ResourceManager : node1.hadoop.com

ZooKeeper

- Server : 3 hosts

[← Back](#) [Print](#) [Deploy →](#)

12. Tunggulah beberapa saat untuk melakukan instalasi services terhadap host yang terdaftar, jika sudah selesai maka klik next.

Install, Start and Test

Please wait while the selected services are installed and started.

97 % overall

Show: All (3) | In Progress (1) | Warning (0) | Success (2) | Fail (0)

Host	Status	Message
master.hadoop.com	92% 	Executing MapReduce2 Service Check
node1.hadoop.com	100% 	Success
node2.hadoop.com	100% 	Success

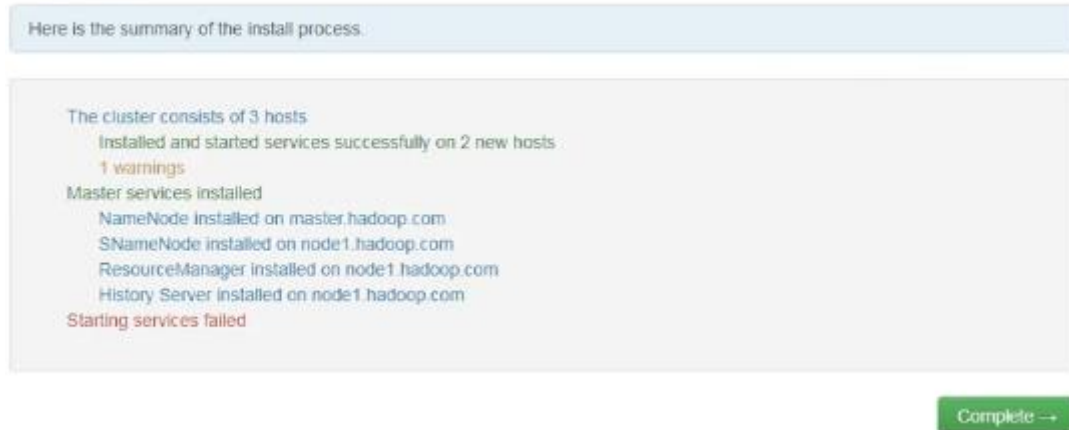
3 of 3 hosts showing - [Show All](#)

Show: 25 1 - 3 of 3 

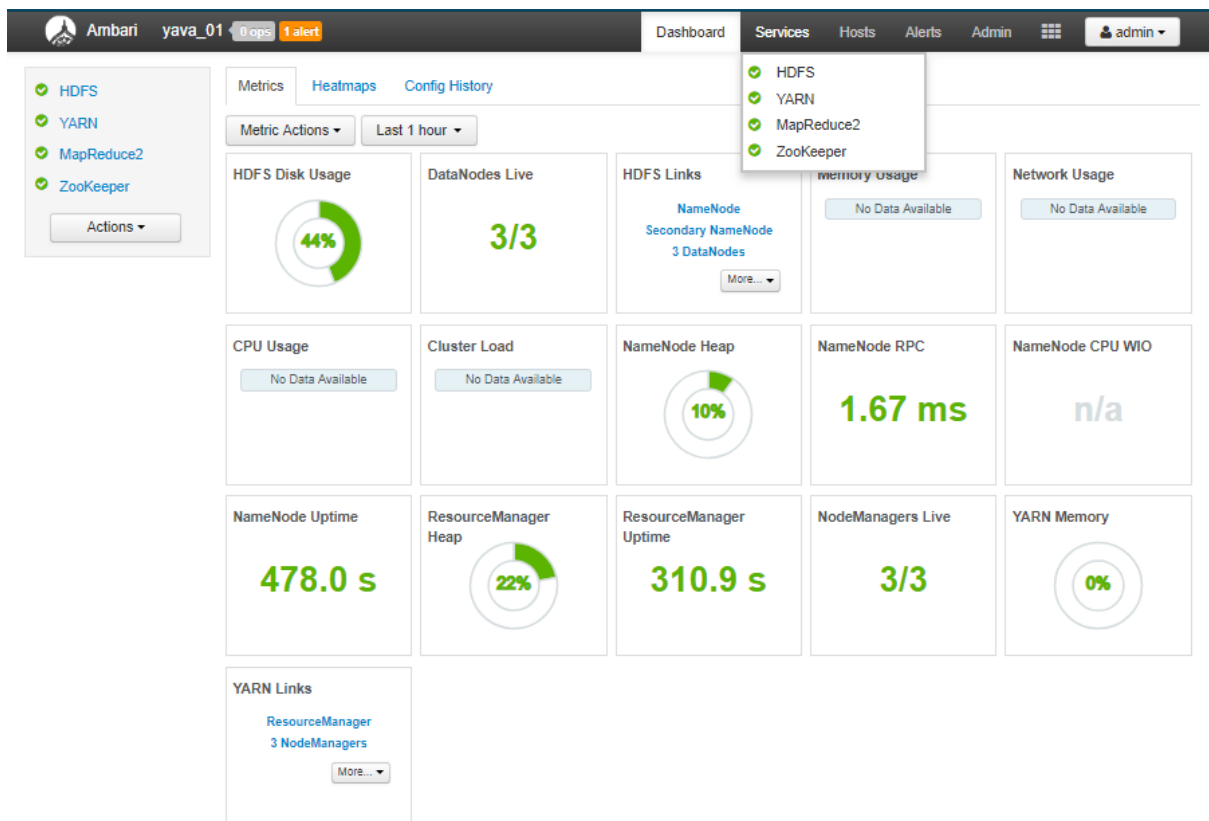
[Next →](#)

13. Berikut tampilan Summary atau kesimpulan dari hasil proses instalasi.

Summary



14. Setelah proses pembuatan Cluster telah selesai, berikut tampilan Dashboard dari Ambari beserta grafik yang ditampilkan berdasarkan Services yang telah diinstall.



Referensi :

https://www.youtube.com/watch?v=T4Ktg6yZ3Q&list=PLY-V_O-O7h4dwTtMk77X6JO-AcENbSBr

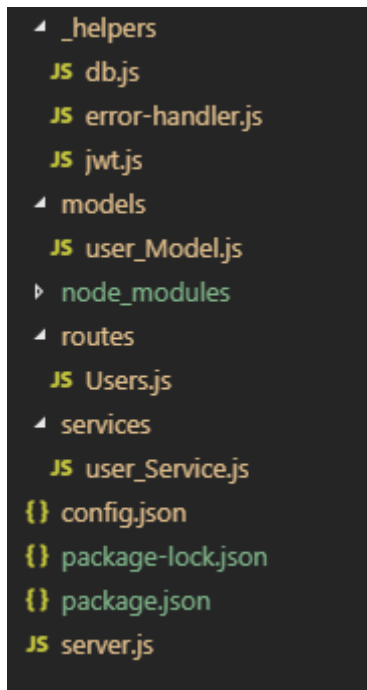
<https://docs.hortonworks.com/HDPDocuments/Ambari/Ambari-2.4.2.0/index.html>

II. PERANCANGAN REST API LOGIN & USER

A. Persiapan Tools :

- Visual Studio Code (Text Editor)
- NPM v.6.4.1
- Express JS (Framework)
- MongoDB (Database)
- Postman (API Tester)

1. Struktur Direktori Back-End / Server :



2. Persiapan :

1. **Bukalah command prompt (cmd) apabila menggunakan Windows.**
2. Buatlah folder dengan nama server : **mkdir server**
3. Lalu masuk ke dalam folder tersebut : **cd server**
4. Buatlah folder-folder berikut di dalam folder server : **_helpers, models, routes, services**
5. Ketik perintah berikut pada command prompt : **npm install —save**
6. Lakukan instalasi modul-modul yang diperlukan : **bcryptjs, body-parser, cors, express, express-jwt, jsonwebtoken, mongodb, mongoose, nodemon, rootpath**

3. Membuat models :

1. Buatlah file config.js pada folder Server /.
Dan isilah file tersebut dengan sintaks berikut, untuk membuat koneksi dengan mongoDB dan membuat database dengan nama java-app :

```
{  
  "connectionString": "mongodb://localhost:27017/java-app",  
  "secret": "secret"  
}
```

2. Buatlah file user_Model.js pada direktori /Models
3. Berikut isi dari file user_Model.js :

```
const mongoose = require('mongoose')  
const Schema = mongoose.Schema  
  
// Membuat schema users  
const schema = new Schema({  
  username: { type: String, unique: true, required: true },  
// hash (password)  
  hash: { type: String, required: true },  
  createdAt: { type: Date, default: Date.now }  
})  
  
schema.set('toJSON', { virtuals: true })  
  
// Eksport model users  
module.exports = mongoose.model('User', schema)
```

B. Fungsi API Login & User :

a. Register / Create User dengan metode POST

<http://localhost:3000/users/register>

Contoh :

The screenshot shows a REST client interface with a POST request to `http://localhost:3000/users/register`. The request body is `x-www-form-urlencoded` with the following data:

KEY	VALUE	DESCRIPTION
username	admin	
password	admin	

The response status is **200 OK** with a time of **492 ms** and size of **272 B**. The response body is `"User Berhasil Didaftarkan!"`.

Berhasil ditambah pada database :

The screenshot shows a MongoDB database interface. The `users` collection contains one document:

Key	Value	Type
<code>_id</code>	<code>ObjectId("5be854559fe1c84a583a2919")</code>	Object
<code>username</code>	<code>admin</code>	String
<code>createdAt</code>	<code>2018-11-11 16:09:57.587Z</code>	Date
<code>hash</code>	<code>\$2a\$10\$4mVG5/ipFLqhKrKt07miuRfCeAcrlVoGXQe1PsXKg7L5JlVj...</code>	String
<code>_v</code>	<code>0</code>	Int32

b. Authenticate User dengan metode POST

<http://localhost:3000/users/authenticate>

Contoh :

The screenshot shows a REST client interface with a POST request to `http://localhost:3000/users/authenticate`. The request body is `x-www-form-urlencoded` with the following data:

KEY	VALUE	DESCRIPTION
username	admin	
password	admin	

The response status is **200 OK** with a time of **288 ms** and size of **508 B**. The response body is a JSON object:

```
{
  "_id": "5be854559fe1c84a583a2919",
  "username": "admin",
  "createdAt": "2018-11-11T16:09:57.587Z",
  "_v": 0,
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI1YmU4NTQ1NT1mZTFjODRhNTgzVTI5MTkiLCJpYXQ1OjE1NDU5NTI5Nj19.YjVkhW1_2XZF6fxgZFQaVxuTQp6GFUU-ffAi3hUoXPg"
}
```

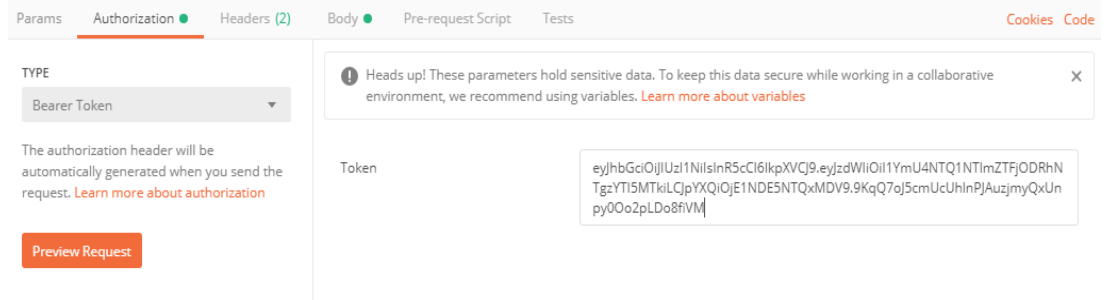
Pada hasil proses authenticate yang berhasil, akan menghasilkan token yang dapat digunakan untuk mendapatkan “authorization”. Sehingga dengan token tersebut kita dapat menggunakan request yang lain seperti untuk melihat data seluruh user (`/getAll`), melihat data user berdasarkan ID (`/getByID`), menghapus data user (`/_delete`), dan memperbaharui data user (`/update`)

- c. Melihat seluruh data user / getAll dengan metode GET

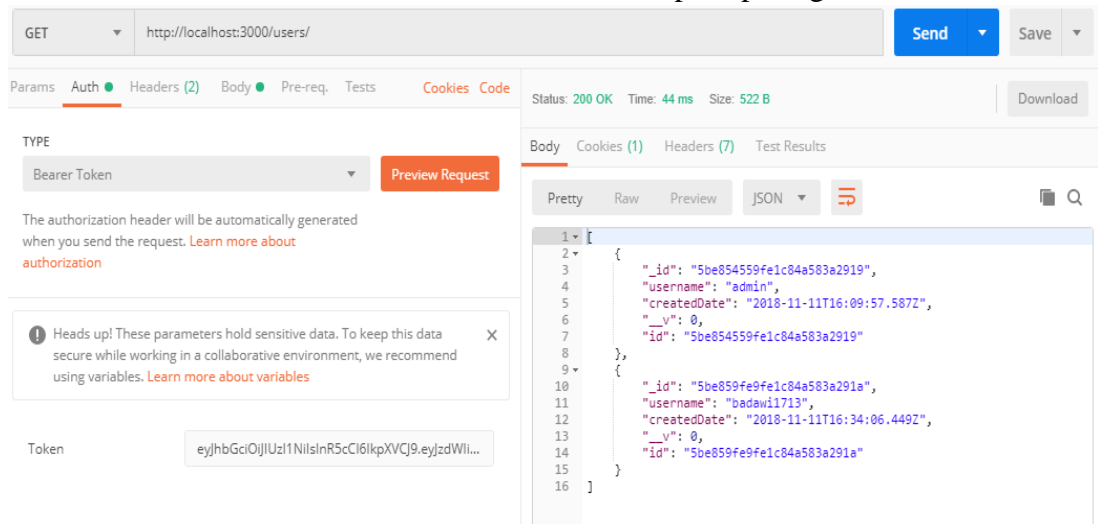
<http://localhost:3000/users/>

Contoh :

- Setelah melakukan /authenticate, maka kita akan mendapatkan token :
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI1YmU4NTQ1NTlmZTFjODRhNTgzYTl5MTkiLCJpYXQiOiE1NDE5NTQxMDV9.9KqQ7oJ5cmUcUhl
nPJAUzjmyQxUnpy0Oo2pLDo8fiVM
- Kemudian klik menu Authorization pada POSTMAN, setelah itu klik TYPE dan pilih metode Bearer Token, dan masukkan token tersebut untuk mendapatkan akses.



- Dan masukkan url untuk melihat semua data user seperti pada gambar berikut :



- d. Melihat data user berdasarkan `_id` / `getById` dengan metode GET

<http://localhost:3000/users/:id>

Contoh :

▼ (2) ObjectId("5be859fe9fe1c84a583a291a")	{ 5 fields }	Object
_id	ObjectId("5be859fe9fe1c84a583a291a")	ObjectId
username	badawi1713	String
createdAt	2018-11-11 16:34:06.449Z	Date
hash	\$2a\$10\$2b/jLEbUCII35ThKJFm2hONMyFq\$TOjTyrFcbrLI8fPi5.UWg...	String
_v	0	Int32

Salinlah ID untuk username badawi1713, kemudian letakkan pada url di aplikasi Postman :

GET <http://localhost:3000/users/5be859fe9fe1c84a583a291a> Send Save

Params Auth Headers (2) Body Pre-req. Tests Cookies Code

TYPE: Bearer Token Preview Request

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [Learn more about variables](#)

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWiiOi...

Status: 200 OK Time: 51 ms Size: 384 B Download

Body Cookies (1) Headers (7) Test Results

Pretty Raw Preview JSON

```
1 {
2   "_id": "5be859fe9fe1c84a583a291a",
3   "username": "badawi1713",
4   "createdAt": "2018-11-11T16:34:06.449Z",
5   "_v": 0,
6   "id": "5be859fe9fe1c84a583a291a"
7 }
```

- e. Update user berdasarkan `_id` / `update` dengan metode PUT

<http://localhost:3000/users/:id>

Contoh :

▼ (2) ObjectId("5be859fe9fe1c84a583a291a")	{ 5 fields }	Object
_id	ObjectId("5be859fe9fe1c84a583a291a")	ObjectId
username	badawi1713	String
createdAt	2018-11-11 16:34:06.449Z	Date
hash	\$2a\$10\$2b/jLEbUCII35ThKJFm2hONMyFq\$TOjTyrFcbrLI8fPi5.UWg...	String
_v	0	Int32

Salinlah ID untuk username badawi1713, kemudian letakkan pada url di aplikasi Postman ubahlah metode request menjadi PUT, dan masukkan data username atau password yang akan diubah pada BODY :

PUT <http://localhost:3000/users/5be859fe9fe1c84a583a291a> Send Save

Params Auth Headers (2) Body Pre-req. Tests Cookies Code

none form-data x-www-form-urlencoded raw binary

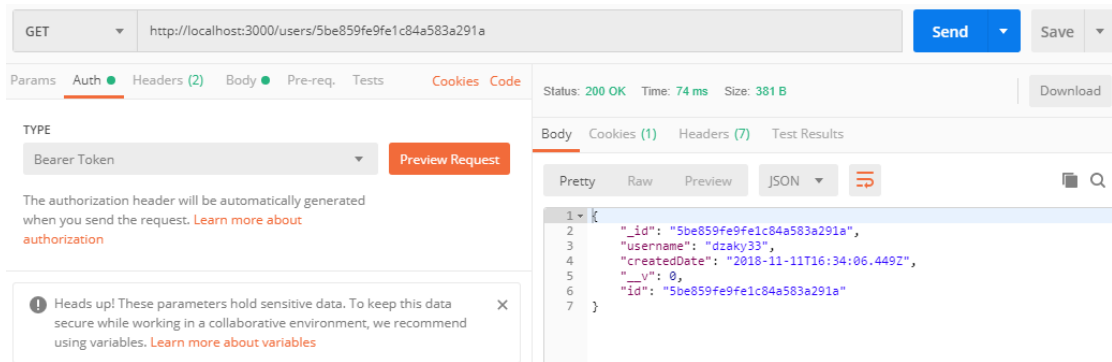
KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	username	dzaky33		
<input checked="" type="checkbox"/>	password	okokok		
Key	Value	Description		

Status: 200 OK Time: 672 ms Size: 272 B Download

Body Cookies (1) Headers (7) Test Results

Pretty Raw Preview JSON

```
1 "Data User Berhasil Diubah!"
```



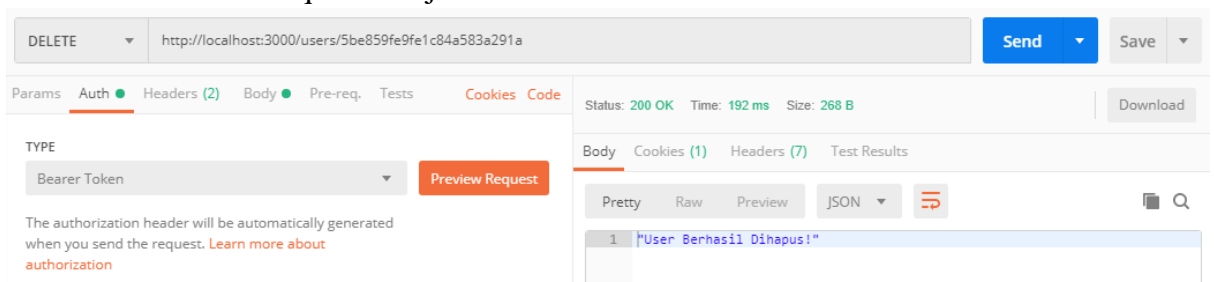
***Username sudah berubah.**

- f. Menghapus data user berdasarkan ID / _delete menggunakan metode DELETE
<http://localhost:5000/users/:id>

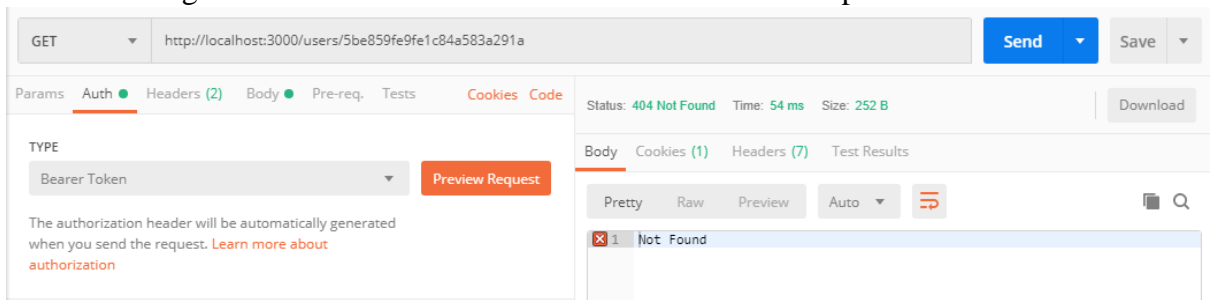
Contoh :

(2) ObjectId("5be859fe9fe1c84a583a291a")	{ 5 fields }	Object
_id	ObjectId("5be859fe9fe1c84a583a291a")	ObjectId
username	badawi1713	String
createdAt	2018-11-11 16:34:06.449Z	Date
hash	\$2a\$10\$2b/jLEbUCI35ThKJFm2hONMyFqSTOjTyrFcbriL8fPi5.UWg...	String
_v	0	Int32

Salinlah ID untuk username dzaky33, kemudian letakkan pada url di aplikasi Postman, dan ubahlah metode request menjadi DELETE :



Data user dengan id **"5be859fe9fe1c84a583a291a"** berhasil dihapus dari database.



Referensi :

<https://www.youtube.com/watch?reload=9&v=yJchGDVZFTA>

<http://jasonwatmore.com/post/2018/06/14/nodejs-mongodb-simple-api-for-authentication-registration-and-user-management>

Source code : <https://github.com/badawi1713/server>