

# VPROFILE PROJECT SETUP

## Prerequisite

1. Oracle VM Virtualbox
2. Vagrant
3. Vagrant plugins
  - a. vagrant plugin install vagrant-hostmanager
  - b. vagrant plugin install vagrant-vbguest
4. Git bash or equivalent editor

## VM SETUP

1. Clone source code.
2. Cd into the repository.
3. Switch to local-setup branch.
4. cd into vagrant/Manual\_provisioning.

Bring up vm's

*\$ vagrant up*

NOTE: Bringing up all the vm's may take a long time based on various factors.  
If vm setup stops in the middle run "vagrant up" command again.

INFO: All the vm's hostname and /etc/hosts file entries will be automatically updated.

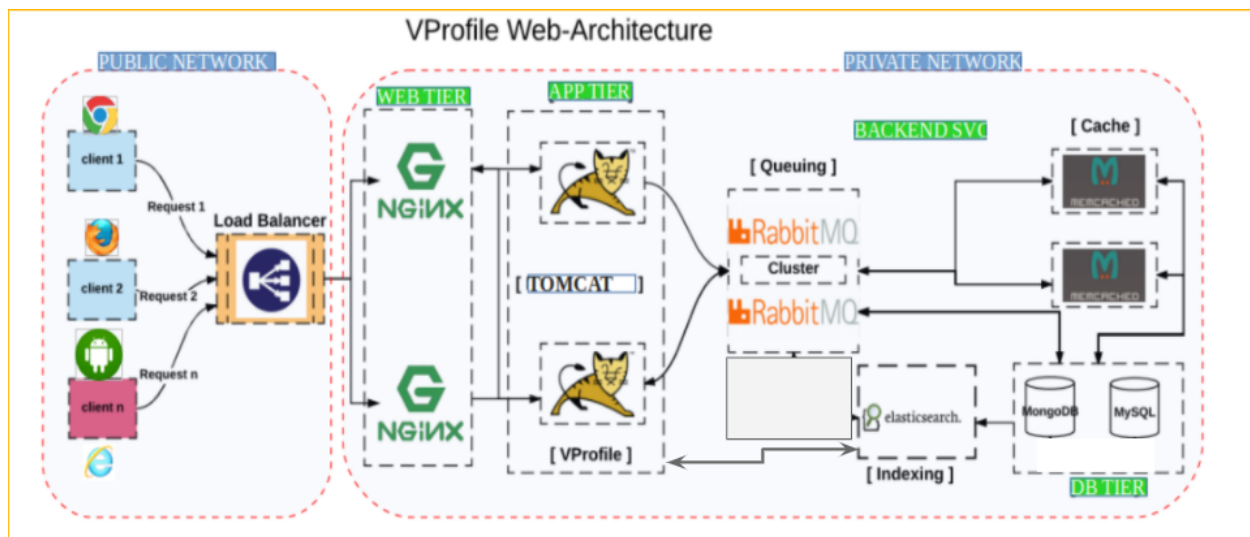
# PROVISIONING

## Services

1. Nginx:  
Web Service
2. Tomcat  
Application Server
3. RabbitMQ  
Broker/Queueing Agent
4. Memcache  
DB Caching
5. ElasticSearch  
Indexing/Search service
6. MySQL  
SQL Database

Setup should be done in below mentioned order

1. MySQL (Database SVC)
2. Memcache (DB Caching SVC)
3. RabbitMQ (Broker/Queue SVC)
4. Tomcat (Application SVC)
5. Nginx (Web SVC)



## MYSQL Setup

Login to the db vm

```
$ vagrant ssh db01
```

Verify Hosts entry, if entries missing update the it with IP and hostnames

```
# cat /etc/hosts
```

Set DB password Variable

```
# DATABASE_PASS='admin123'
```

Update OS with latest patches

```
# yum update -y
```

Set Repository

```
# yum install epel-release -y
```

Install Maria DB Package

```
# yum install git mariadb-server -y
```

Starting & enabling mariadb-server

```
# systemctl start mariadb
```

```
# systemctl enable mariadb
```

RUN mysql secure installation script.

```
# mysql_secure_installation
```

**NOTE:** Set db root password as same as you set in the variable above  
(DATABASE\_PASS='admin123')

```
Set root password? [Y/n] Y
```

```
New password:
```

```
Re-enter new password:
```

```
Password updated successfully!
```

```
Reloading privilege tables..
```

```
... Success!
```

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

```
Remove anonymous users? [Y/n] Y
```

... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n

... skipping.

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] Y

- Dropping test database...

... Success!

- Removing privileges on test database...

... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] Y

... Success!

## Download Source code

```
# git clone -b local-setup https://github.com/devopshydclub/vprofile-project.git
# cd vprofile-project
```

## Set DB name and users.

```
# mysql -u root -p"$DATABASE_PASS" -e "create database accounts"
# mysql -u root -p"$DATABASE_PASS" -e "grant all privileges on accounts.* TO
'admin'@'app01' identified by 'admin123' "
# mysql -u root -p"$DATABASE_PASS" accounts < src/main/resources/db_backup.sql
# mysql -u root -p"$DATABASE_PASS" -e "FLUSH PRIVILEGES"
```

## Restart mariadb-server

```
# systemctl restart mariadb
```

## Starting the firewall and allowing the mariadb to access from port no. 3306

```
# systemctl start firewalld
# systemctl enable firewalld
# firewall-cmd --get-active-zones
# firewall-cmd --zone=public --add-port=3306/tcp --permanent
# firewall-cmd --reload
# systemctl restart mariadb
```

## MEMCACHE SETUP

Install, start & enable memcache on port 11211

```
#yum install epel-release -y
#yum install memcached -y
#systemctl start memcached
#systemctl enable memcached
#systemctl status memcached
#memcached -p 11211 -U 11111 -u memcached -d
```

Starting the firewall and allowing the port 11211 to access memcache

```
# systemctl enable firewalld
# systemctl start firewalld
# systemctl status firewalld
# firewall-cmd --add-port=11211/tcp --permanent
# firewall-cmd --reload
# memcached -p 11211 -U 11111 -u memcache -d
```

## RABBITMQ SETUP

Login to the RabbitMQ vm

```
$ vagrant ssh rmq01
```

Verify Hosts entry, if entries missing update the it with IP and hostnames

```
# cat /etc/hosts
```

Update OS with latest patches

```
# yum update -y
```

Set EPEL Repository

```
# yum install epel-release -y
```

Install Dependencies

```
# yum install socat -y
```

```
# yum install wget -y
```

Create Repo file for Erlang package

Create below mentioned file.

```
vi /etc/yum.repos.d/rabbitmq_erlang.repo
```

Update the file with below mentioned content

```
[rabbitmq_erlang]
name=rabbitmq_erlang
baseurl=https://packagecloud.io/rabbitmq/erlang/el/7/$basearch
repo_gpgcheck=1
gpgcheck=1
enabled=1
# PackageCloud's repository key and RabbitMQ package signing key
gpgkey=https://packagecloud.io/rabbitmq/erlang/gpgkey
        https://dl.bintray.com/rabbitmq/Keys/rabbitmq-release-signing-key.asc
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
metadata_expire=300

[rabbitmq_erlang-source]
name=rabbitmq_erlang-source
baseurl=https://packagecloud.io/rabbitmq/erlang/el/7/SRPMS
```

```
repo_gpgcheck=1
gpgcheck=0
enabled=1
# PackageCloud's repository key and RabbitMQ package signing key
gpgkey=https://packagecloud.io/rabbitmq/erlang/gpgkey
        https://dl.bintray.com/rabbitmq/Keys/rabbitmq-release-signing-key.asc
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
metadata_expire=300
```

## Install Erlang

```
# yum install erlang -y
```

Run below mentioned command to setup rabbitmq-server repo file.

```
# curl -s
https://packagecloud.io/install/repositories/rabbitmq/rabbitmq-server/script.rpm.sh | sudo
bash
```

## Install Rabbitmq Server

```
# yum install rabbitmq-server -y
```

## Start & Enable RabbitMQ

```
# systemctl start rabbitmq-server
# systemctl enable rabbitmq-server
# systemctl status rabbitmq-server
```

## Config Changes

```
# echo "[{rabbit, [{loopback_users, []}]}]." > /etc/rabbitmq/rabbitmq.config
```

## Add user test in rabbitmq

```
# rabbitmqctl add_user test test
```

## Set admin privileges to rabbit user

```
# rabbitmqctl set_user_tags test administrator
```

Restart RabbitMQ service

```
# systemctl restart rabbitmq-server
```

Enabling the firewall and allowing port 25672 to access the rabbitmq permanently

```
# systemctl start firewalld
```

```
# systemctl enable firewalld
```

```
# firewall-cmd --get-active-zones
```

```
# firewall-cmd --zone=public --add-port=25672/tcp --permanent
```

```
# firewall-cmd --reload
```

## **TOMCAT SETUP**

Login to the tomcat vm

```
$ vagrant ssh app01
```

Verify Hosts entry, if entries missing update the it with IP and hostnames

```
# cat /etc/hosts
```



Update OS with latest patches

```
# yum update -y
```

Set Repository

```
# yum install epel-release -y
```

Install Dependencies

```
# yum install java-1.8.0-openjdk -y
```

```
# yum install git maven wget -y
```

Change dir to /tmp

```
# cd /tmp/
```

Download & Tomcat Package

```
# wget https://archive.apache.org/dist/tomcat/tomcat-8/v8.5.37/bin/apache-tomcat-8.5.37.tar.gz  
-O tomcatbin.tar.gz
```

*Optional Step, Below variables are used in “Copy data to tomcat home dir” step. You can use the tomcat directory name also in the source.*

```
# EXTOUT=`tar xzvf tomcatbin.tar.gz`  
# TOMDIR=`echo $EXTOUT | cut -d '/' -f1`
```

Add tomcat user

```
# useradd --home-dir /usr/local/tomcat8 --shell /sbin/nologin tomcat
```

Copy data to tomcat home dir

```
# rsync -avzh /tmp/$TOMDIR /usr/local/tomcat8/
```

Make tomcat user owner of tomcat home dir

```
# chown -R tomcat.tomcat /usr/local/tomcat8
```

Setup systemd for tomcat

Update file with following content.

```
vi /etc/systemd/system/tomcat.service
```

[Unit]

Description=Tomcat

After=network.target

[Service]

User=tomcat

WorkingDirectory=/usr/local/tomcat8

Environment=JRE\_HOME=/usr/lib/jvm/jre

Environment=JAVA\_HOME=/usr/lib/jvm/jre

Environment=CATALINA\_HOME=/usr/local/tomcat8

Environment=CATALINE\_BASE=/usr/local/tomcat8

ExecStart=/usr/local/tomcat8/bin/catalina.sh run

ExecStop=/usr/local/tomcat8/bin/shutdown.sh

SyslogIdentifier=tomcat-%i

[Install]

WantedBy=multi-user.target

*# systemctl daemon-reload*

*# systemctl start tomcat*

*# systemctl enable tomcat*

Enabling the firewall and allowing port 8080 to access the tomcat

*# systemctl start firewalld*

*# systemctl enable firewalld*

*# firewall-cmd --get-active-zones*

*# firewall-cmd --zone=public --add-port=8080/tcp --permanent*

*# firewall-cmd --reload*

## **CODE BUILD & DEPLOY (app01)**

Download Source code

```
# git clone -b local-setup https://github.com/devopshydclub/vprofile-project.git
```

Update configuration

```
# cd vprofile-project
```

```
# vim src/main/resources/application.properties
```

```
# Update file with backend server details
```

Build code

*Run below command inside the repository (vprofile-project)*

```
# mvn install
```

Deploy artifact

```
# systemctl stop tomcat
```

```
# sleep 120
```

```
# rm -rf /usr/local/tomcat8/webapps/ROOT*
```

```
# cp target/vprofile-v2.war /usr/local/tomcat8/webapps/ROOT.war
```

```
# systemctl start tomcat
```

```
# sleep 300
```

```
# chown tomcat.tomcat /usr/local/tomcat8/webapps -R
```

```
# systemctl restart tomcat
```

## NGINX SETUP

Login to the Nginx vm

```
$ vagrant ssh web01
```

Verify Hosts entry, if entries missing update the it with IP and hostnames

```
# cat /etc/hosts
```

Update OS with latest patches

```
# apt update
```

```
# apt upgrade
```

Install nginx

```
# apt install nginx -y
```

Create Nginx conf file with below content

```
# vi /etc/nginx/sites-available/vproapp
```

```
upstream vproapp {  
    server app01:8080;  
}  
server {  
    listen 80;  
    location / {  
        proxy_pass http://vproapp;  
    }  
}
```

Remove default nginx conf

```
# rm -rf /etc/nginx/sites-enabled/default
```

Create link to activate website

```
# ln -s /etc/nginx/sites-available/vproapp /etc/nginx/sites-enabled/vproapp
```

Restart Nginx

```
# systemctl restart nginx
```