RAILS APP DEPLOY TO AWS WITH PASSENGER+NGINX+POSTGRESQL

1.SET UP AWS

2. CREATE SECURITY GROUP
In the EC2 Dashboard, click Security Groups, then Create Security Group.
Enter name
Enter Description
Rules to add

    HTTP
    SSH
    Custom TCP Rule: Port 3000 (if using rails development environment)

Click create

3.CREATE IAM ROLE

4.CREATE EC2 INSTANCE

5.CREATE ELASTIC IP

6.UPDATE PEM PERMISSIONS IN LOCAL MACHINE TO CHMOD 400

7.SSH INTO THE INSTANCE - `ssh -i <location/of/pem/file> ubuntu@<elastic_ip>`

8.SETUP USER IN INSTANCE

```
sudo apt-get update
sudo apt-get upgrade -y
sudo dpkg-reconfigure tzdata #to set time zone
```

#ADD A ANOTHER USER FOR YOUR APP IF YOU WANT. MUST COPY PUBLIC KEY TO THIS USERS AUTHORIZED_KEYS FILE ALSO

#SET UP A FIRE WALL IF YOU WANT

9.INSTALL DEPENDENCIES

```
sudo apt-get install git-core curl zlib1g-dev build-essential libssl-dev libreadline-dev libyaml-dev libsqlite3-dev sqlite3 libxml2-dev libxslt1-dev libcurl4-openssl-dev python-software-properties libffi-dev openssl bison -y
```

10.DO NOT INSTALL GEM DOCS:

```
echo "gem: --no-ri --no-rdoc" > ~/.gemrc
```

11.INSTALL NODE.JS

```
sudo add-apt-repository ppa:chris-lea/node.js
sudo apt-get update
sudo apt-get install nodejs -y
```

12.INSTALL POSTGRES

```
sudo apt-get install postgresql postgresql-contrib libpq-dev -y
```

#IF YOU SEE SOME ERROR AS CANNOT CREATE CLUSTORS OR A WARNING LIKE “The locale requested by the environment is invalid” THEN DO FOLLOWING. SOMETIMES YOU MAY NEED TO SWITCH TO SUPER USER (sudo su #to switch to super user) AND BACKTO NORMAL USER AND DO FOLLOWING IN BOTH ENVIORENMENTS.

```
export LANGUAGE="en_US.UTF-8"
echo 'LANGUAGE="en_US.UTF-8"' >> /etc/default/locale
echo 'LC_ALL="en_US.UTF-8"' >> /etc/default/locale
```

or

```
export LANG="en_US.utf8"
export LANGUAGE="en_US.utf8"
export LC_ALL="en_US.utf8"
```

#WHEN TYPE “locale -a” OR “psql -V” YOU SHOULD NOT GET ANY EERORS OR WARNINGS. THIS IS DUE TO SOME ERRORS IN UBUNTU 14.04

13.INSTALL RUBY WITH RBENV (MAY TAKE FEW MINUITES)

```
cd
git clone git://github.com/sstephenson/rbenv.git .rbenv
echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >> ~/.bashrc
echo 'eval "$(rbenv init -)"' >> ~/.bashrc
source ~/.bashrc

git clone git://github.com/sstephenson/ruby-build.git ~/.rbenv/plugins/ruby-build
echo 'export PATH="$HOME/.rbenv/plugins/rubybuild/bin:$PATH"' >> ~/.bashrc
source ~/.bashrc

git clone https://github.com/sstephenson/rbenv-gem-rehash.git ~/.rbenv/plugins/rbenv-gem-rehash
rbenv install 2.3.3 #HERE INSTALL YOUR APP CREATE RUBY VERSION
rbenv global 2.3.3 #HERE ALSO SAME, MAKE IT GLOBAL RUBY VERSION
ruby -v
```

14.INSTALL BUNDLER

```
gem install bundler
rbenv rehash
```

15.INSTALL RAILS

```
gem install rails
```

16.CREATE A DATABASE USER IN YOUR AWS INSTANCE (STILL AT AWS INSTACE :))

```
sudo -u postgres createuser -s <user> #IF YOU DID NOT CREATE A SEPERATE USER FOR YOUR APP
THEN                                              #MOST PROBABLY HERE <user> SHOULD BE ubuntu
```

17. BUILD YOUR APP (I AM SURE YOU HAVE DONE IT ALLREADY ;)). # THIS IS DONE IN LOCAL MACHINE. MAKE YOUR APP DATABASE IN POSTGRESQL (rails new app_name -d postgresql) . MAKE config/secrtes.yml FILE .gitignore. PUSH YOUR CODE TO YOUR REMOTE REPOSITORY

SERVER SETUP ###### SO BACK AGAIN IN AWS INSTANCE :O

18.CREATE APP FOLDER OR BEST WAY IS GO TO HOME'S CURRENT USER DIRECOTRY (TYPING cd ONLY FROM ANY WHERE BRINGS YOU BACK TO YOUR HOME'S CURRENT USER DIRECTORY) AND CLONE YOUR APP FROM REMOTE REPO (git clone https://github.com/***/***)

19. INSTALL THE APT KEY

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 561F9B9CAC40B2F7
```

20.CREATE AN APT SOURCE FILE

```
sudo vim /etc/apt/sources.list.d/passenger.list
```

21.INSERT TO THAT FILE FOLLOWING

```
deb https://oss-binaries.phusionpassenger.com/apt/passenger trusty main
```

#WHEN USING vim YOU MAY NEED TO KNOW SOME KEYS. TO GO TO INSERT(TYPING) MODE PRESS KEY #FROM a/A/i/I/o/O. TO GOTO COMMAND ENTERING MODE PRESS esc(escape key). TO SAVE AND EXIT, GO #TO COMMAND ENTERING MODE AND TYPE :wq AND PRESS ENTER. TO GO WITHOUT SAVING PRESS :q!.

22.IF YOU WANT YOU CAN CHANGE OWNER PERMISSIONS OF THE ABOVE FILE

```
sudo chown root: /etc/apt/sources.list.d/passenger.list
sudo chmod 600 /etc/apt/sources.list.d/passenger.list
```

23.UPDATE APT CACHE

```
sudo apt-get update
```

24.UNINSTALL OS INSTALLED NGINX IF ANY

```
sudo apt-get purge nginx nginx-full nginx-light nginx-naxsi nginx-common
sudo rm -rf /etc/nginx
```

24.INSTALL NGINX AND PASSENGER

```
sudo apt-get install nginx-extras passenger -y
```

25.UPDATE NGINX CONFIG FILE

```
sudo vim /etc/nginx/nginx.conf
```

26.UNCOMMENT FOLLOWING LINE, IF NOT EXIST ADD IT

```
include /etc/nginx/passenger.conf;
```

27.SAVE AND EXIT

28.OPEN DEFAULT SERVER BLOCK

```
sudo vim /etc/nginx/sites-available/default
```

29.COMMENT ALL THERE (SPECIALLY FOLLOWING TWO)

```
listen 80 default_server;
listen [::]:80 default_server ipv6only=on;
```

30.CREATE PROPER SERVER CONFIG FILE

```
sudo vim /etc/nginx/sites-available/<appname>
```

31.ADD FOLLOWING TO THAT FILE

```
server {
        listen 80 default_server;
        server_name <dns_or_elastic_or_normal_ip>
        passenger_enabled on;
        # passenger_app_env development; # ONLY IF IN DEVELOPMENT MODE
        root /home/<user>/<appname>/current/public;
}
```

32.CREATE SYM LINK TO SITES-ENABLED

```
sudo ln -s /etc/nginx/sites-available/<appname> /etc/nginx/sites-enabled/<appname>
```

33.RELOAD AND RESTART NGINX SO CHANGES TAKE EFFECT

```
sudo nginx -s reload
sudo service nginx restart
```

34.VALIDATE INSTALLATION OF NGINX AND PASSENGER

```
rvmsudo passenger-config validate-install  #SOME TIME JUST sudo ALSO WORKS
rvmsudo passenger-memory-stats
```

33.CONFIGURE APP FOR DEPLOYMENT.

```
#YOU CAN CHOSE CAPISTRANO FOR EASY DEPLOYMENT. BUT I WILL NOT DO THAT. I WILL GIT
CLONE FROM AWS INSTANCE AND DEPLOY. SO NO CAPISTRANO HERE.
```

34.ADD PASSENGER GEM TO GEM FILE

```
gem 'passenger'
```

35.THEN BUNDLE INSTALL

```
bundle install
```

36.GENERATE SECRET

```
rake secret
#COPY THE OUTPUT
```

37.COPY THE PUTPUT TO config/secrets.yml

```
secret_key_base: <secret_you_copied>
```

38.REMOVE (COMMENT) DB USERNAME AND PASSWORD FROM config/database.yml

```
# username: <appname>
# password: <%= ENV['<APPNAME>_DATABASE_PASSWORD'] %>
```

39.COMMIT AND PUSH YOUR CODE TO REMOTE REPOSITORY

```
git add .
git commit -m “ready for deployment”
git push origin branch_name
```

```
###############BACK IN AWS INSTANCE #################
```

40.GO TO HOME CURRENT USER DIRECTORY

```
cd
```

41.CLONE YOUR APP FROM REMOTE REPOSITORY

```
git clone https://github.com/***/***
```

42.GO TO THAT APP DIRECTORY

```
cd <appname>
```

43.CHECKOUT TO NEEDED BRANCH IF NEEDED

```
git checkout branch_name
```

44.INSTALL DEPENDENCIES

```
bundle install
```

45.CREATE DATABASES

```
bundle exec rake db:create:all # TO CREATE ALL ENVIORENMENT DATABASES.
Bundle exec rake db:migrate:all
```

46.START PASSENGER

```
bundle exec passenger start
#IF NEEDED TO START IN PRODUCTION ENVIORENMENT USE FOLLOWING
passenger start -a 0.0.0.0 -p 3000 -d -e production
#IF NEEDED YOU CAN START PASSENGER IN DIFFERENT PORT ALSO AS BELOW
bundle exec passenger start -p 3001
```

47.IF YOU WANT TO STOP SERVER, GO TO APP DIRECTORY AND

```
passenger stop
```

48.UPDATE

#AS WE DID NOT USE CAPISTRANO, WE HAVE REMOVE APP DIRECTORY AND MUST CLONE IT BACK AGAIN. THE DATA IN DATABASES ARE NOT DESTROYED DOING THIS.

```
$rm -rf <appname>
git clone https://github.com/***/***
cd <appname>
git checkout <branch_name> #ONLY IF NEEDED
bundle install
bundle exec passenger start #TO START SERVER AGAIN AFTER CLONING UPDATES
```