**Ruby on Rails Cloud configuration**

**Pre-Requirment**

1. Cloud Server IP Address
2. Username
3. Password or Key pairs
4. Putty Client , PuttyGen
5. FileZilla

**Application Details**

1. Ruby on Rails Application

**Version** : Ruby 1.9.3 , Rails 3.2.13 ,Gem 1.8.24

**CMS** : Refinerycms

**Steps To Configure the Ruby on rails Environment in cloud**

**Creation of Amazon Account**

Create an account in amazon. The below details are related to Amazon EC2 instances. We have to create EC2 instance to deploy our ruby on rails. After creating account , login into amazon control panel , the control panel helps to create instance in cloud. Consider here we are having Amazon EC2 instance for Learnguild.com

1. **Generate Public and Private keys Using PuttyGen**

After creating amazon instance you can able to access the server instance through SSH. But we need to use public and private keys for authentication. From your instance you can download [**.pem**] file. We can create public and private keys from .pem file using puttyGen

2. Connect to the Remote Server using ip via **SSH using Putty Client**. Use the generated keys for the authentication

Putty is a client tool used to connect remote linux server through SSH[Secure Shell]

3. Enter the username in putty and connect it

4. Now we can run commands through Putty Client

The following steps remove all the existing ruby-rails packages from the system. By following the steps you install all the updated packages.

5. Run **dpkg -S $(which ruby)** - It will show the details of installed ruby version in remote server

6. Run **dpkg -S $(which rails)** - It will show the details of installed rails version

7. It is better to remove all the installed ruby and rails gem. So Run

**sudo apt-get autoremove** - It will remove all the gems and make the system ready for fresh install

8. Run **sudo** **apt-get update –** It will update the system kernel

9. Run **sudo** **apt-get upgrade -** It will upgrade the system kernel

10. Since the kernel update, must reboot the server before going to install Ruby on rails

Run **sudo reboot**

11. Now we can install Ruby Before that run

Run **sudo apt-get install curl -**  It checks the system has update curl

12. Using RVM we going to install ROR

Run **sudo curl -L https://get.rvm.io | bash -s stable**

13. Now we need to change the RVM user mode

Run **sudo groupadd rvm** – This Command add a new user group

Run **sudo usermod -a -G rvm ubuntu -** This Command change the user mode

14. Now Just Logout and Again Login to the Putty

15. Run **rvm requirements -** It Install all the rvm requirements

16. Now We going to install Ruby on rails by mention the version in rvm

Run **rvm install 1.9.3**

These will take some time to finish

17. Make these RVM as default one

Run **rvm use 1.9.3 –default**

18. Next we going to install RubyGem

Run **rvm rubygems current -** It will install all the Latest gems.

19. After Install Ruby and Ruby gem Install Rails

Run **gem install rails --version=3.2.13 -** This is the basic command to install Rails 3.2.13

20. Now we need Sqlite gem for Database configuration

Run **gem install sqlite3-ruby -** This command install sqlite gem required for the database

21. Now we need to install apache gem to configure the apache server

Run **sudo apt-get install apache2-mpm-worker -** This command install apache gems needed to runb ruby on rails in apache

22. After installing Apache, Make sure it runs properly by entering the ip with port number in browser

Eg: <http://50.198.233.211:8080>

The apache displays the default page “It Works”

23. Now It our need to install more apache modules

Run **sudo apt-get install apache2-threaded-dev libcurl4-openssl-dev**

24. The deployment of ruby on rails with apache is easy when we use passenger tool

Passenger is a deployment tool which acts as a middleware between Ruby on Rails and the Apache Server. It is the preferred option to integrate with apache

Run **gem install passenger -** This will install the passenger gems

25. Now add a module to Configure the passenger with apache server

Run **passenger-install-apache2-module -** This install a apache plug helps to make configure between passanger and apache

26. After finishing the above steps. Create a file under named

File path & name:  **/etc/apache2/conf.d/passenger.conf**

Content

**LoadModule passenger\_module /home/ubuntu/.rvm/gems/ruby-1.9.3-p429/gems/passenger-4.0.2/libout/apache2/mod\_passenger.so**

**<IfModule mod\_passenger.c>**

**PassengerRoot /home/ubuntu/.rvm /gems/ruby-1.9.3-p429/gems/passenger-4.0.2**

**PassengerDefaultRuby /home/ubuntu/.rvm /wrappers/ruby-1.9.3-p429/ruby**

**</IfModule>**

27. Now move the application to remote server using Filezilla

First convert the application folder into .zip format

Second Connect the server through filezilla

Third Place the .zip Application file under **/home/ubuntu --** It is better use the same location but not necessary we can place it in any where

28.Now navigate to the application file using putty client

Unzip the file using

Make sure your server have .zip package. If not Run **sudo apt-get install zip**

**sudo** **unzip** **/home/ubuntu/Applicationfilename.zip**

29. Change the folder name like [www.learnguild.com](http://www.learnguild.com) using below command

Run **mv Application\_folder** [**www.learnguild.com**](http://www.learnguild.com)

30. Add Virtual host file

31. Create /var/ruby/www.learnguild.com file

32. Create a symlink

Run **ln -s /var/ruby/www.learnguild.com /home/ubuntu/learnguild**

33. Create a file for /**etc/apache2/sites-available/www.learnguild.com/**

34. Restart the Apache

Run **sudo service apache2 restart**

35. Now navigate to **/etc/apache2/sites-enabled**

Run **delete the file 000-default**

36. Run **ln ../sites-available/www.learnguild.com 000-www.learnguild.com**

Apache supports running different websites on the same server, you keep all the config files for each website on /etc/apache2/sites-available and it needs the active ones to be linked to /etc/apache2/sites-enabled

37. Restart apache

38.Open /**etc/apache2/sites-available/www.learnguild.com** and Change

RailsEnv from **Development** to **Production**

39. Make sure the content in the passeneger.confi

**<VirtualHost \*:80>**

**ServerAdmin example@gmail.com**

**ServerName www.yourhost.com**

**DocumentRoot /somewhere/public # <-- be sure to point to 'public'!**

**<Directory /somewhere/public>**

**AllowOverride all # <-- relax Apache security settings**

**Options -MultiViews # <-- MultiViews must be turned off**

**</Directory>**

**</VirtualHost>**

40. Run **chown -R www-data.www-data /var/ruby/www.learnguild.com –** It helps to change the permission mode of the folder

41. Now Run **gem install bundler** – it will help to re install everything in proper manner.

42. Navigate to **/var/ruby/www.learnguild.com**  and Run **bundle install –deployment**

43. Run the command **chmod ugo+w -R /var/ruby/www.learnguild.com** – it is used to set permission

41. Make sure the git is installed **sudo apt-get install git**

42. use **chmod -R ugo+w**

43. Restart Apache - **sudo service apache2 restart**

44. Now open the browser and access through your ip address. The website homepage will display

45. Finally one more gem needed to install for handling images

**sudo apt-get imagemagick**

The above command will install the imagematricks package our linux server

46. Now install the gem to integrate imagematricks with our ROR

Run **gem install mimemagic**

47**.** Now restart the server **sudo service apache2 restart**

48**.** Login into the refinery and make sure the images are generated internally. If it is not try refreshing the pages. Then images will display.

49. We can access only through by giving the ip address of the cloud. To change this to URL we need a domain from domain service provider.

50. Login to your domain control panel and mention the cloud ip as redirect ip.

You can also get help from hosting support team

Here we have a domain named <http://www.learnguild.com> from Godaddy.com

Open

**Addition Linux commands:**

**>cd foldername -** It is used for navigation to specific folder

>**cd ..**  - Moving one folder back

>**vi Filename.txt -** This is used to edit the config file or application file in linux server itself. Vi is an editor used to update file it in linux os

>**ls -** It shows the list of file in folder

> **ln -s /home/source\_location/destination\_location**

The above command helps us to create symblink. In windows we call it as shortcut. In linux it is Symblink

**S**till having any trouble in loading, just check whether all the gits are installed properly and check all are available

**References:**

<https://www.digitalocean.com/community/articles/how-to-install-ruby-on-rails-on-ubuntu-12-04-lts-precise-pangolin-with-rvm>

<http://www.modrails.com/documentation/Users%20guide%20Apache.html>