Manual for the Replication of MrJinIT.com Server

Jin Cherng Chong (33170193)

School of Engineering and Information Technology
Murdoch University
12/05/2018

Table of Contents

Abstract:

This report provides systematic documentation on how I created and customised my own EC2 server. The report should be considered a resource guide that aims to teach students the basics of setting up an EC2 server, WordPress, and a personal web-based git repository (Gogs).

Website name: mrjinit.com
Private Ipv4 elastic: 52.76.195.180
Public DNS: ec2-52-76-195-180.ap-southeast-1.compute.amazonaws.com
(If you are trying to ping it you shouldn't be able to:
https://serverfault.com/questions/511738/why-cant-i-ping-my-freshly-set-up-amazon-web-service-ec2-instance)

1.0 Introduction

This report provides step by step documentation on how I created and customised my own EC2 server. The report should be considered a resource guide that aims to teach students the basics of setting up an EC2 server, WordPress, and a personal web-based git repository (Gogs). My personal WordPress serves as a resume outlining my qualities and knowledge while my web-based git repository is intended for the storage of my university assignments and notes. My WordPress website has a button which, when clicked, redirects to my Git.

This report assumes that readers will have an Amazon EC2 server with the following requirements-

Minimum:

Server: Amazon EC2
OS: Ubuntu Server 16.04 LTS (HVM), SSD Volume Type
Processor: Intel Xeon at 2.5GHZ with 1 vCPUs
Memory: 1 GiB
Storage: 8 GiB

The report also assumes that the reader has the following software installed on their computer-

Putty
PuttyGen
Google Chrome

For my personal EC2 server, the amount of storage I have is 20 GiB but this amount of storage is not necessary. The operating system I used in conjunction with the Amazon EC2

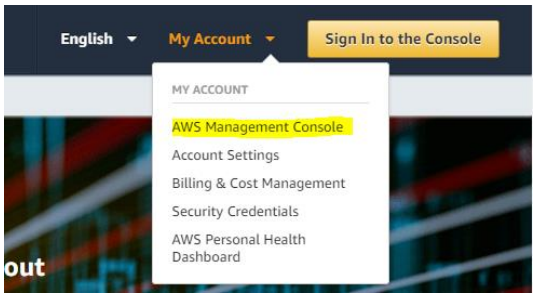
operating system is Windows 10 and the domain registrar I refer to in the report is Namesilo.com.

2.0 Amazon EC2 & DNS

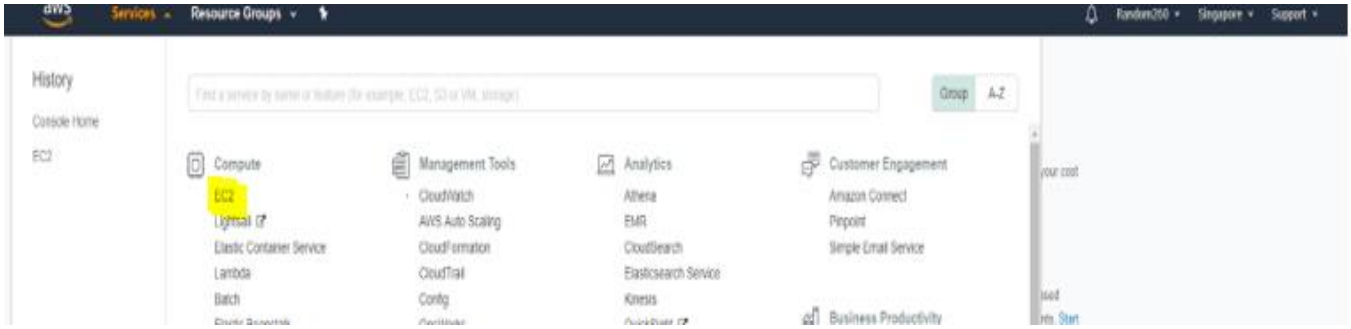
Amazon EC2 is a cloud-based server offered in the Amazon Webserver market place [1]. For small to medium business, it serves as a cheaper alternative to buying and maintaining their own server [2]. A company that is purchasing a cloud server only really must consider the renting cost of the server, while a company that wants to own their own server needs to consider additional factors such as the environment where the server is stored and whether they need to hire workers to maintain the server. These additional factors will drastically affect the total cost of ownership for a server. A domain name system (DNS) is a system that translates the domain name to the IP address identifying a device [3]. The domain name can be purchased from registrars such as- Namesilo and Crazydomains [4].

2.1 Creating Amazon EC2 Instance

To start, make sure you are logged into your Amazon Webserver Account and are in the AWS Management Console: https://aws.amazon.com/ec2/



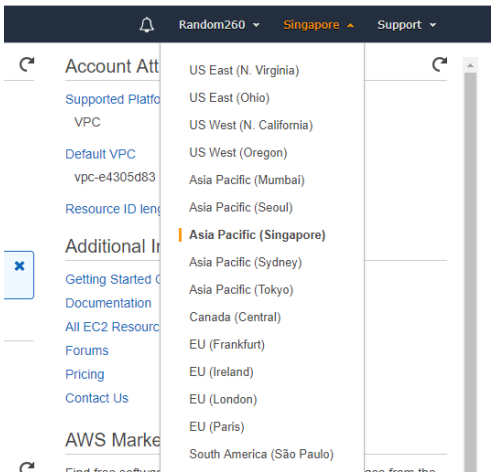
Once you are in the AWS Management console
Click on the 'Services' tab
Click on the 'EC2' option

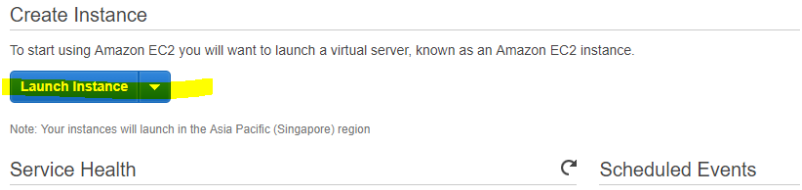


You should now see a ‘Launch Instance’ option.
This option allows you to deploy and configure an Amazon EC2 Server.

The Amazon EC2 will deploy in a default location. For me, the default location is in Singapore and it is my optimal location. The optimal server location is often determined by figuring out the purpose of your server. If you want to run a gaming server and most of your player base is in Texas, then your server should be closer Texas. Having a gaming server closer to your player base means that data will be transferred quicker from the gaming server to the player base computers and vice versa [5]. The result of quicker transfer of data means lower ping and better performance for the players [5]. However, the optimal location for this server will be the location closest to you. The reason for this is to minimise the time it will take for us to upload our files to the Gitlab.

Click ‘(The default location)’
Select your optimal location
Click ‘Launch Instance’





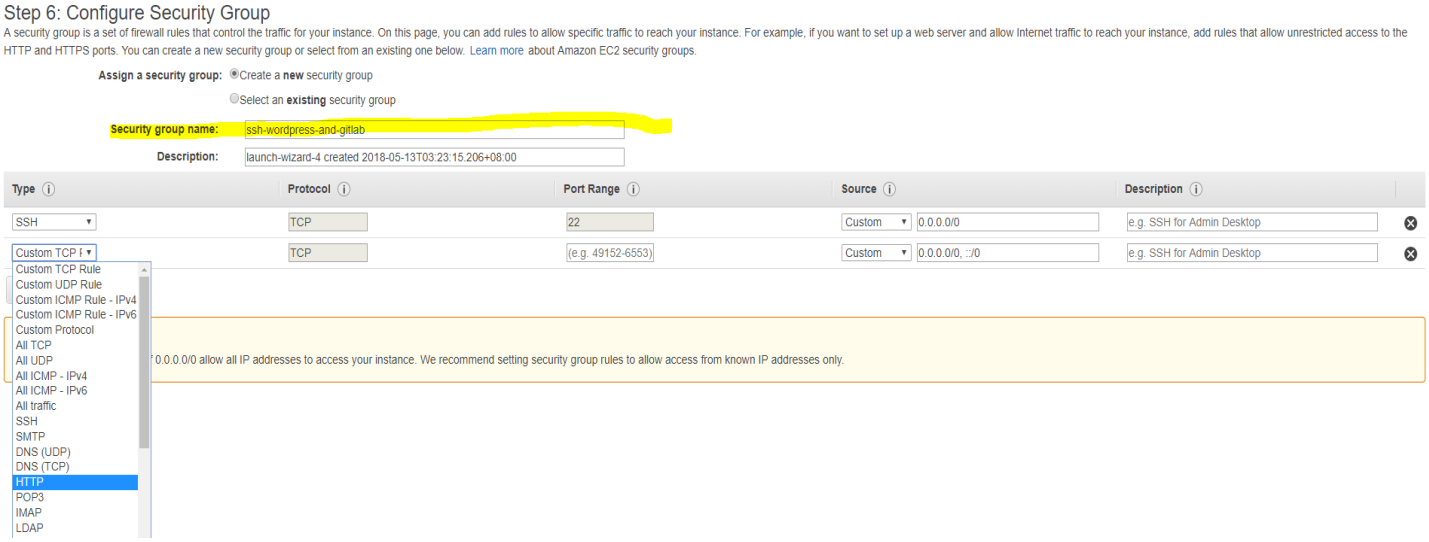
Now, it is time to configure our Amazon EC2. You will be confronted with 7 configurations steps- choose AMI, instance type, storage, tags, security groups, and review.

Refer back to page 2 of this report to figure out which 'Amazon Machine Image' and 'Instance Type' to choose
Leave 'Configure instance' settings as the default
Refer back to page 2 of this report to work out the amount of 'storage' to add
Leave 'Add Tags' settings as the default

For 'Configuring Security Groups' step

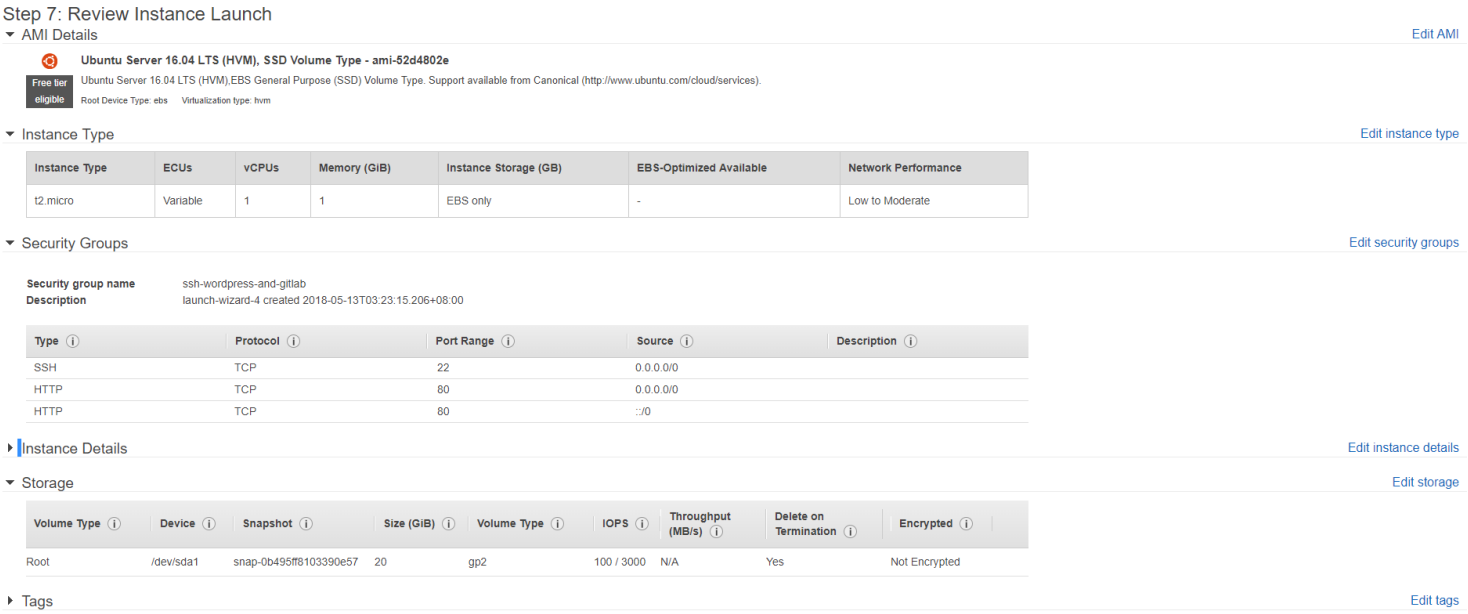
Type ssh-wordpress-and-gitlab in 'Security group name:' textbox
Click 'Add Rule'
Click 'Custom TCP Rule' menu in the 'type' column
Select 'HTTP'

By selecting HTTP, it allows web requests to be accepted by our server. This is necessary for our WordPress website.



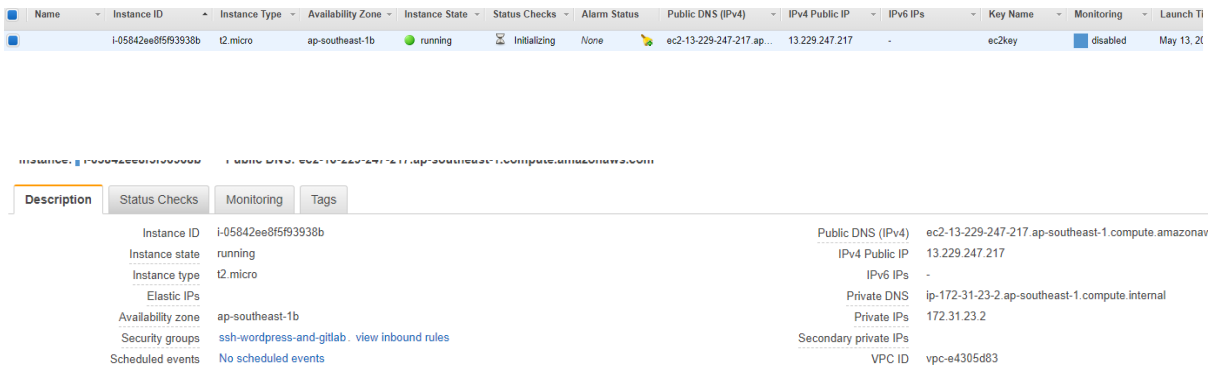
Click 'Review and launch'

This is how my server is configured; it should look very similar to the server you just configured.



Finally:

Click 'Launch' if you are happy
Select 'Create a new key pair'
Type ec2key or whatever name you want to call key in 'Key pair name' textbox
Click 'Download Key Pair'
Click 'Launch Instances'
Click 'View Instances'



Congratulations! Your Amazon EC2 Server should be set up perfectly and you are now ready to begin the installation of WordPress and Gitlab.

IMPORTANT TO ACCESS EC2 SERVER:

Refer to either Windows or Linux link depending on operating system you are using in conjunction to your EC2 server

Windows: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html
Linux:
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.html

2.2 Setting up server DNS

This section is where you will be setting up your website's DNS service. We will be installing apache2 to help do this. This section is part of the installation of a WordPress website.

Before we begin make sure you are connected an Amazon EC2 server through a ssh client such as Putty. Also, just to be safe, update all your packages to their latest versions. The latest packages of software tend to have fewer vulnerabilities [6].

Type 'sudo apt update' in ssh client. Then type 'y' when prompted
Type 'sudo apt upgrade' in ssh client. Then type 'y' when prompted

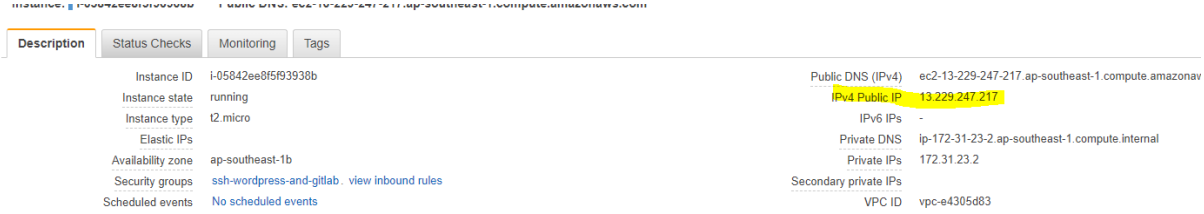
Once upgraded, we need to install apache2.

Type 'sudo apt install apache2' then type 'y' when prompted

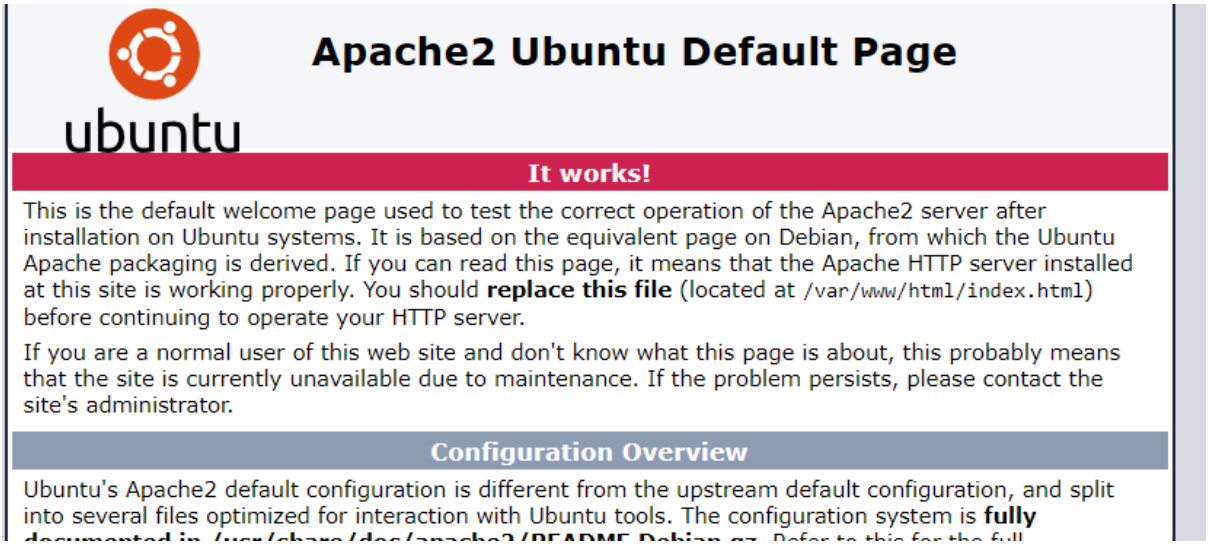
To check whether the webserver was configured and installed correctly try to go to the website (not from your server web browser). The web addresses for the apache2 web server is the Amazon EC2 'IPv4 Public IP'.

This can be found in the Amazon EC2 dashboard website:

Click 'Instances'
Refer to the IPv4 Public IP at the bottom of the website



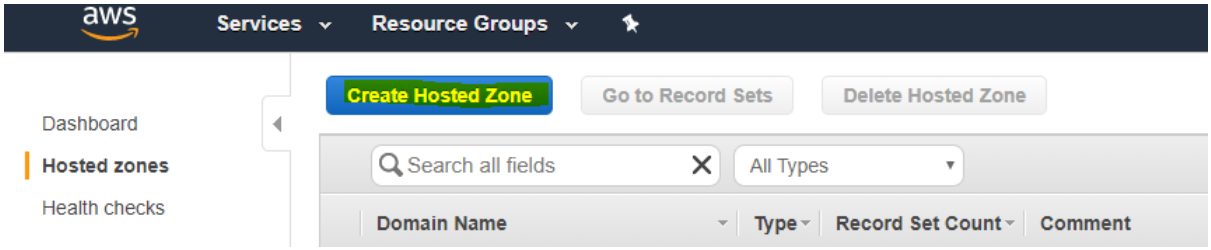
Open your web browser on your main computer and
Copy and paste the 'Ipv4 Public IP' in the web browser's URL search box and check if you see a website called 'Apache2 Ubuntu Default Page'*



*If the above does not appear then when you tried to go to the apache2 web addresses from a web browser, you most likely made a mistake along the way. The possible reasons for this may include: not adding a HTTP rule (page 4), firewall issues, mistyping, or forgetting to type commands. Try typing in the ssh client 'sudo ufw disable' to turn off the firewall.
Now, it is time to assign a domain name to the web address. A domain registrar allows for the purchase of a domain name. The one registrar I will be referring to is namesilo.com.

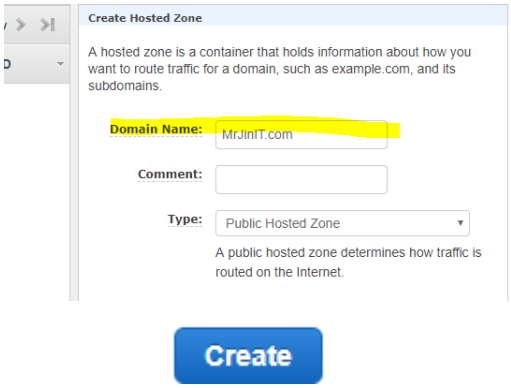
From the Amazon EC2 dashboard

Click 'Services' tab
Click 'Route 53' option found below
Select 'DNS management'
Click 'Create Hosted Zone' button twice



On the right-hand side of the website:

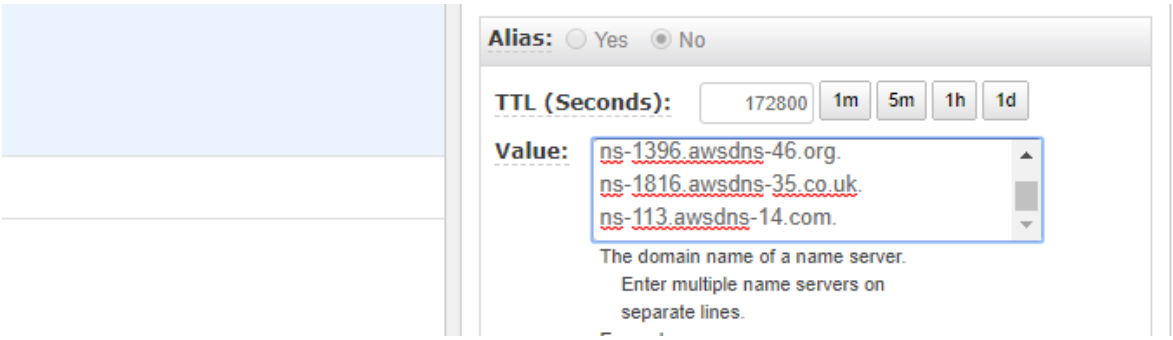
Enter your domain name in 'Domain Name:' textbox
Click 'Create'



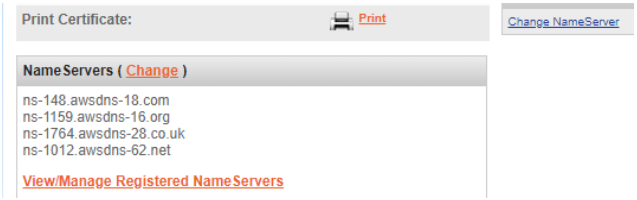
Click on the dot on the left of 'Domain Name'
Select 'Go to Record Sets'
Select on the dot on the left of 'Name'

On the right, you can see the 'value' There should be four domain names. What you need to do now is copy these four 'values' and go to your registrar and enter them in the Name Server configuration fields. This process will slightly differ from registrar to registrar. However, I will show you how I did it through Name silo.

Copy 'value' from record set



Go to Registrar (Namesilo.com)
Sign in
Click 'Manage My Domains'
Click on the domain so you get to 'Domain Console'
Select 'Change' on the right of Nameservers heading



Paste all 4 "value" from record set into the 'NameServer:' textbox
Press 'submit'

There is an issue with this method because a restart to the EC2 may cause the IP to change, thus the DNS may not point to the server address. To circumvent this, use an elastic IP address. So on the EC2 dashboard:

Click 'Elastic IPs' then 'Allocate new address'
Click 'Action' and 'Associate with instance'

3.0 WordPress

In this section, I will be explaining how to install, set up WordPress, and add features. Before we can actually install WordPress, there are a number of software packages we need to install these include: php7, MySQL, and WordPress [7].

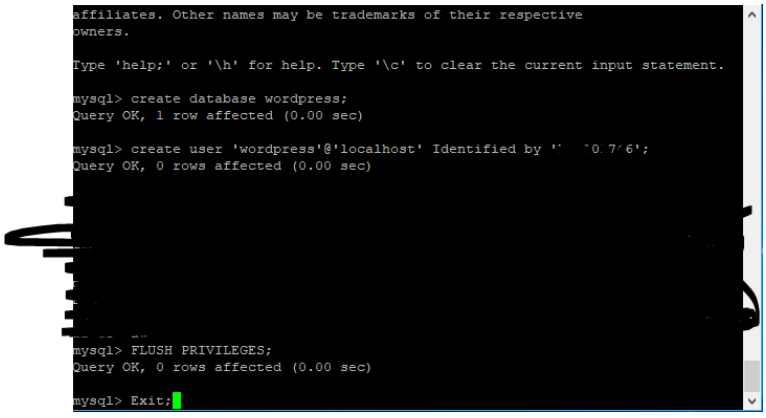
3.1 Setting up WordPress

We need to remove the index.html because when we install WordPress we should use the WordPress index.html

```
sudo rm /var/www/index.html
```

Then we should install and configure MySQL to work with WordPress. The package **MySQL** is a database management system, which is needed for WordPress. The database for WordPress will also be created:

```
sudo apt-get install mysql-server php7.0-mysql
Enter password of mysql
mysql –u root –p
Type create database [enter database name here];
Type create user '[Enter username]'@'localhost' Identified by '[Enter WordPress password]';
Type GRANT ALL PRIVILEGES ON [database name].* TO '[Username entered above]'@'localhost;
Enter FLUSH PRIVILEGES;
Type Exit;
```



The next step is installing **php7.0**

```
sudo apt-get install php7.0 libapache2-mod-php7.0 php7.0-cgi
```

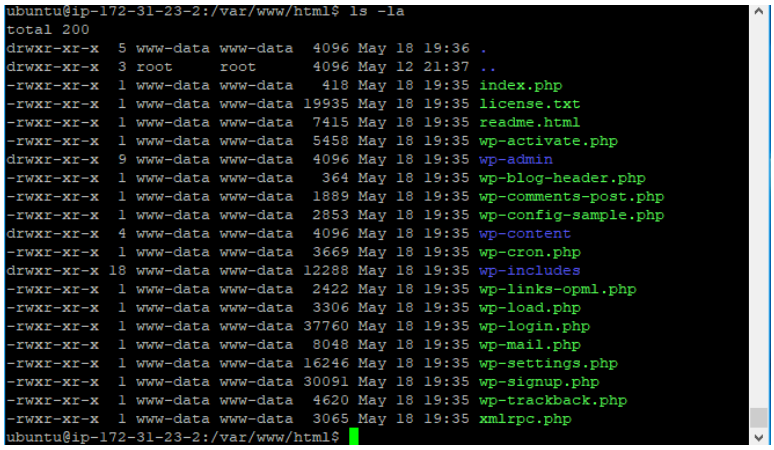
After installing php7.0, we will need to install WordPress and modify one of the files to link it to the newly created database. The first step is going to the /var/www/html directory. Since it is in the public web directory it can be stored and accessed by anyone with permission.

```
Type cd /var/www/html
Type sudo wget https://wordpress.org/latest.zip
Type sudo apt-get unzip
```

Type sudo unzip latest.zip
Enter sudo rm latest.zip
Type sudo cp –r wordpress/* /var/www/html
Finally *type* sudo rm –r wordpress

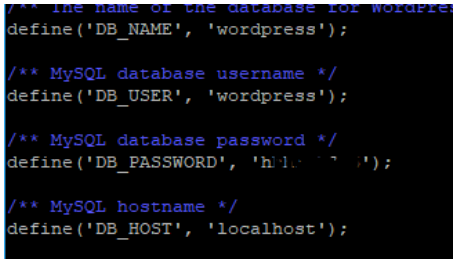
It is also important to change the permissions of the files and folders.

Type sudo chown –r www-data:www-data /var/www/html/
Type sudo chmod -R 755 /var/www/html/

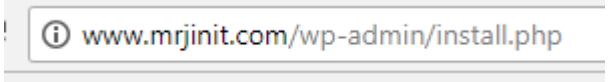


The last couple of steps would be modifying the configuration file to link to the database created earlier. We want to rename the file as well since the current name has the word ‘sample’ in it [7].

Enter sudo mv wp-config-sample.php wp-config.php
Type sudo nano wp-config
Change Dn_name, DB_usr and DB_Password with configurations typed earlier
Type sudo service apache2 restart
Type sudo service mysql restart



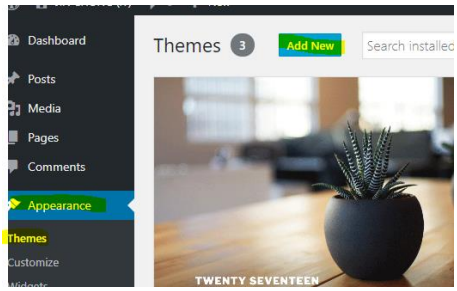
Now check if everything is installed correctly by going to either the server domain name or server IP address. Then follow the prompts to complete installation.



3.2 WordPress Features

One of the important features is learning how to **install a theme.** A theme is a collection of files that are designed to work together in order to produce a graphical user interface for a weblog [8]. WordPress allows the installation of free themes or paid themes at different price points. The parent theme I used for my website is Genesis Framework, while the child theme is beautiful pro theme.

Go to Dashboard
Hover mouse over menu 'Appearance'
Click 'Themes'
Click 'Add new'
Click 'Upload theme and Choose File'
Choose and Upload zip file with the theme. (It can be located in any directory on server)
Click 'Activate'

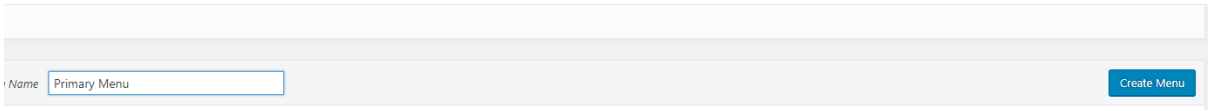


Another Important feature is creating **a category menu**:

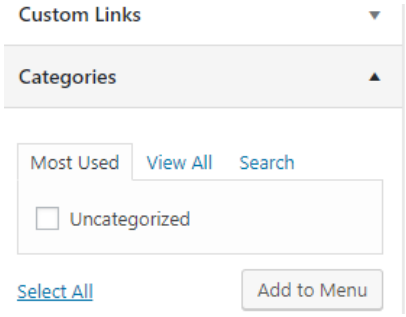
A custom-link menu function is when a user clicks on the menu and automatically goes to the URL. The URL may be a website, or it may be the path to download a file. Another menu type offered is categories menu. This menu allows for submenus. Finally, when a post menu is

created and clicked it will redirect to the URL of the post created. Below are screenshots showing how to create a category menu.

Go to Dashboard
Hover over 'Appearances'
Click 'Menu'
State the 'Menu Name' and click create menu



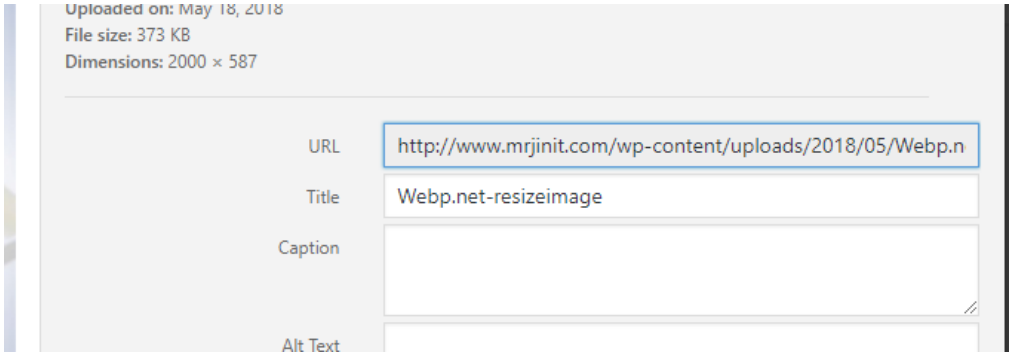
Click 'Categories' on the left-hand side



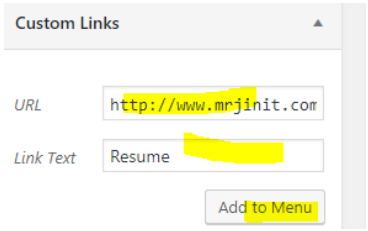
Check 'Uncategorized'
Click 'Add to menu'
Expand 'Uncategorized' and change label to ideal name of menu for it is: GitLab
Check 'Primary Navigation Menu' (allows the menu to appear on home page at top)
Save menu

The next feature I have implemented on my website was **creating a menu and linking to an item** on my server. (Custom Links)

Go to Dashboard
Click 'Media' and 'Upload New Media'
Click on Uploaded Media
Copy 'URL'

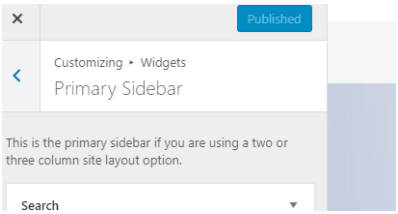


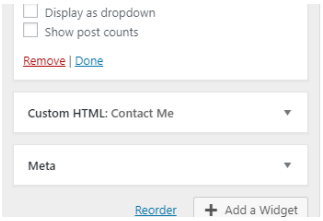
Go to Dashboard
Hover over 'Appearances'
Click 'Menu'
Expand 'Custom Links' and change URL to the URL copied
Change 'Link text' to name of menu and Add to Menu then Save Menu



The Third feature is **adding widgets**:

The changing of widgets is quite simple.
Open 'Customisation' tab on the home page
Click 'Widget'
Select 'Primary side bar'
Then you can add or delete widgets on the right-hand side





The fourth feature is **changing the background**

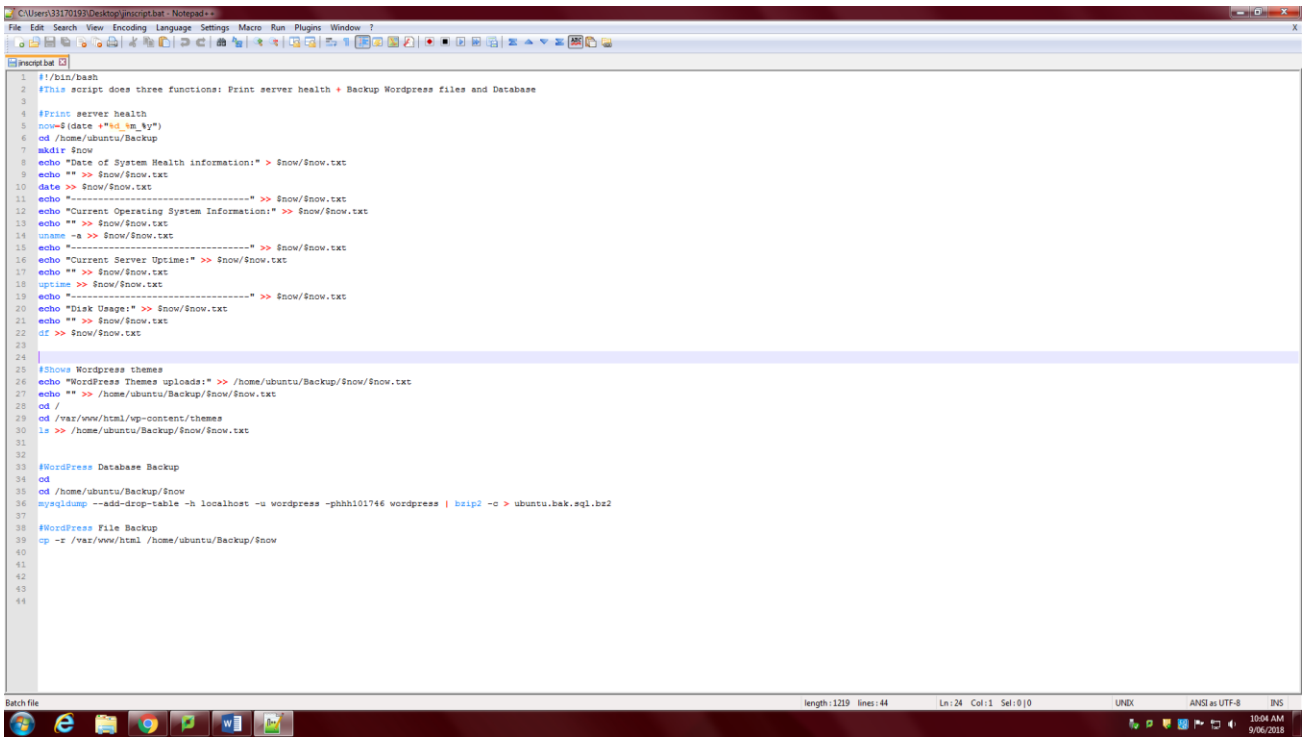
Go to WordPress dashboard
Click 'Media' and then 'Add new' to upload background image
Click on the image to get URL
Click on 'Editor' found in 'Appearances' menu
Replace

```
body {
	background-color: #fff;
	color: #666;
	font-family: 'Lato', sans-serif;
	font-size: 18px;
	font-weight: 300;
	line-height: 1.625;
}
```

```
body {
	background-image: url('http://www.mrjinit.com/wp-content/uploads/2018/05/3147.jpg');
	font-family: 'Lato', sans-serif;
	font-size: 18px;
	font-weight: 300;
	line-height: 1.625;
}
```

4.0 Server Scripting

Link to output of my script: **https://www.youtube.com/watch?v=xhD-nwLwlZ4**



```
#!/bin/bash
#This script does three functions: Print server health + Backup Wordpress files and Database

#Print server health
now=$(date +"%d_%m_%y")
cd /home/ubuntu/Backup
mkdir $now
echo "Date of System Health information:" > $now/$now.txt
echo "" >> $now/$now.txt
date >> $now/$now.txt
echo "---------------------------------" >> $now/$now.txt
echo "Current Operating System Information:" >> $now/$now.txt
echo "" >> $now/$now.txt
uname -a >> $now/$now.txt
echo "---------------------------------" >> $now/$now.txt
echo "Current Server Uptime:" >> $now/$now.txt
echo "" >> $now/$now.txt
uptime >> $now/$now.txt
echo "---------------------------------" >> $now/$now.txt
echo "Disk Usage:" >> $now/$now.txt
echo "" >> $now/$now.txt
df >> $now/$now.txt

#Shows Wordpress themes
echo "WordPress Themes uploads:" >> /home/ubuntu/Backup/$now/$now.txt
echo "" >> /home/ubuntu/Backup/$now/$now.txt
cd /
cd /var/www/html/wp-content/themes
ls >> /home/ubuntu/Backup/$now/$now.txt


#WordPress Database Backup
cd
cd /home/ubuntu/Backup/$now
mysqldump --add-drop-table -h localhost -u wordpress -phhh101746 wordpress | bzip2 -c > ubuntu.bak.sql.bz2

#WordPress File Backup
cp -r /var/www/html /home/ubuntu/Backup/$now
```

(Zoom in to see more clearly. (not actually a bat file did it so it highlights things))

The above is my server scripting code, which I configured to run once every day. This script has three main objectives: printing different types of server information and storing it in a text file, backing up the MySQL database in a zip file, and backing up all the WordPress files. All the three files- system information text file, MySQL zip and WordPress files- are stored in a directory with the name of the directory being the date. This script is useful for me since I plan to migrate my WordPress website to a more cost efficient virtual private server in the future. I

also use this script a lot while I am modifying my website. All it takes is a simple 'jinscript' command and everything is backed up, like how with Microsoft Word you would press ctrl + s to save every so often.

The commands I used for the script are commonly recognised commands. The echo command will output a string. For my script, the string would be written in a text file, with the date as the txt file's name. The additional commands implemented are also added to text file created earlier.

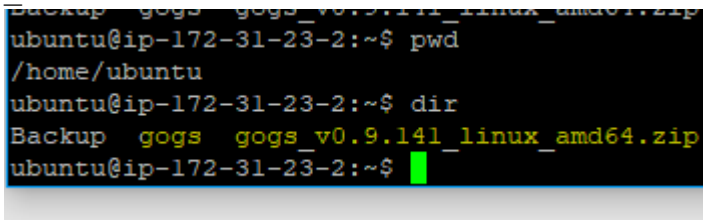
5.0 Gog Git Service

For my server, I decided to go with a lightweight alternative to Gitlab/Github. The Amazon Ec2 instance does not support Gitlab since the minimum specifications needed is 2GB of ram while our instance is 512mb[x]. Please understand that the git repository you are about to install is intended for your use only. This manual will not support additional user registrations on your hosted Git service. However, it is possible to run this Git service for many other users.

5.1 Setting Up and Installing Gog

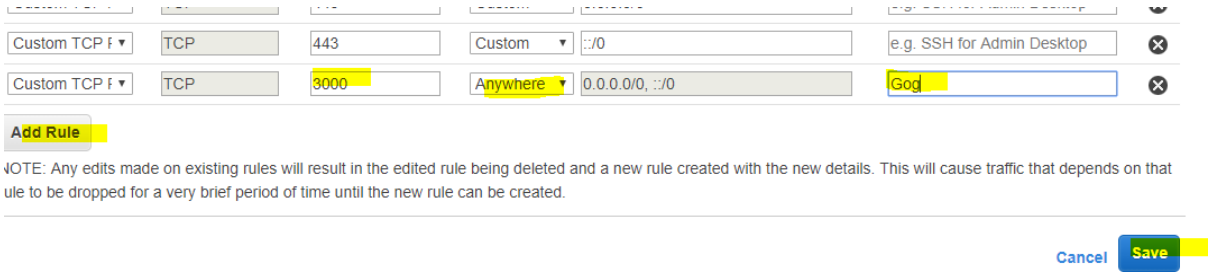
Firstly, download the Gog repository from https://dl.gogs.io/ [9]. Also create database for Gog:

Type cd /home/ubuntu
Type wget https://dl.gogs.io/gogs_v0.9.141_linux_amd64.zip
Type unzip gogs_v0.9.141_linux_amd64.zip
Type mysql –u root –p
Type CREATE DATABASE gogs CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
Type \q



The next step is to open an inbound port in order for Gogs to work. The port we will be opening is port 3000.

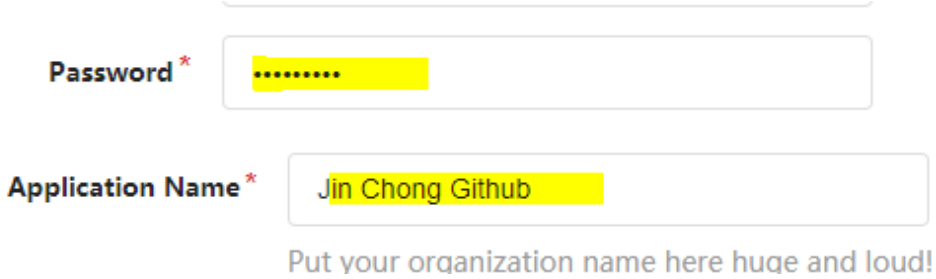
Go to Amazon EC2 dashboard
Click 'Security Groups' option on the left
Click on the group name you are using
Click 'Inbound' bottom located at bottom.
Click 'Edit'
Click 'Add Rule'
Type '3000' in port range
Click 'Custom' in source column and change to 'Anywhere' then click save



Then we will need to run Gogs and set it up:

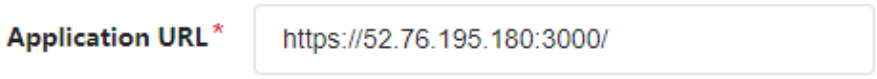
Type cd /home/ubuntu/gogs
Type ./gogs web
Go to Firefox and enter IP_ address:3000
Fill in the details

Please note below are the properties I modified for my server. You should also modify it to suit you.



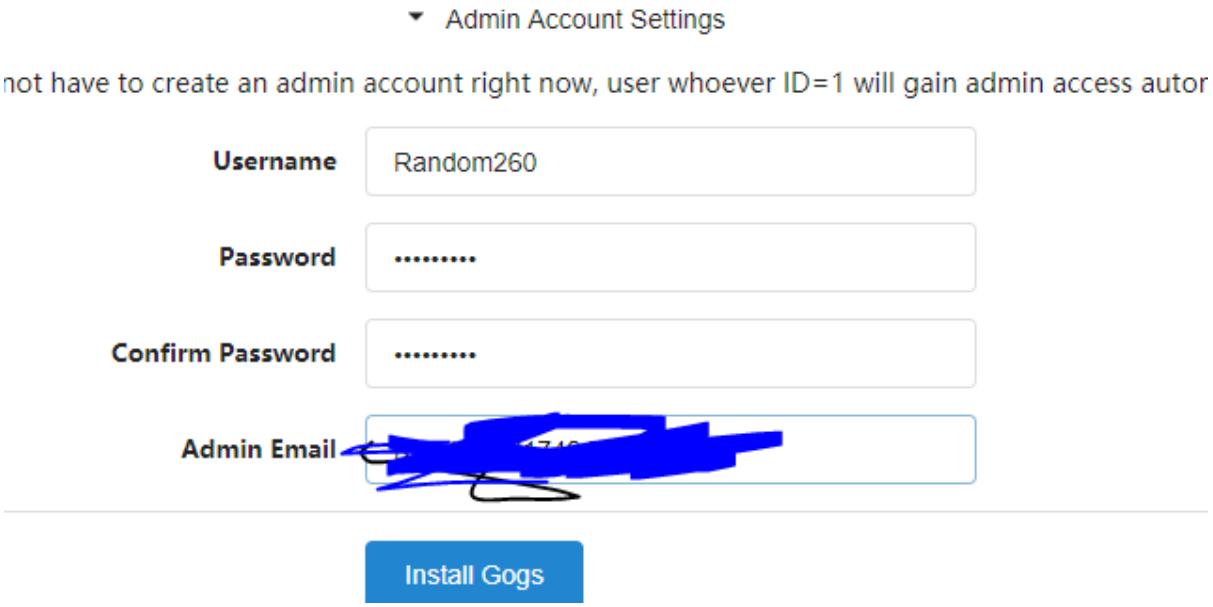
(Application name is your name of git hub)

(Should be your Amazon EC2 IP address.)

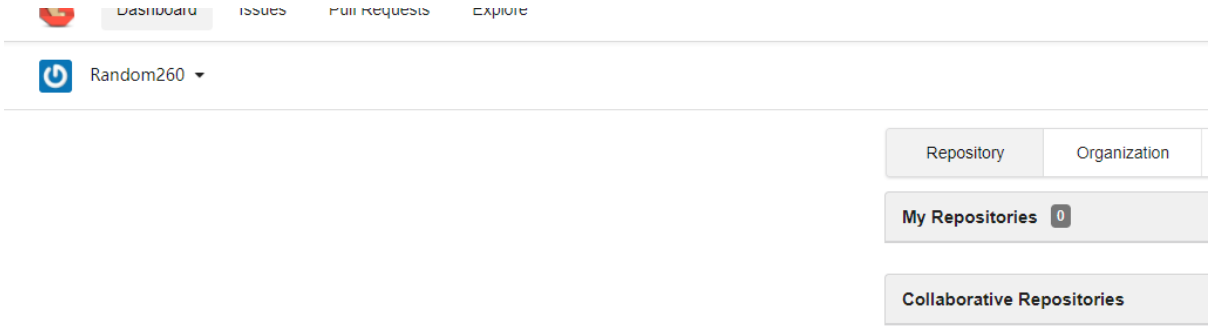


(Should be your IP address and port)

Then the last step is to create your account where you will upload content and *Click* 'Install'



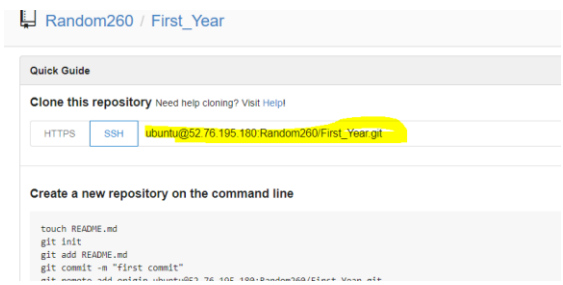
Well done, you've just installed your own Gog! It should look like this:



5.2 Creating Repository

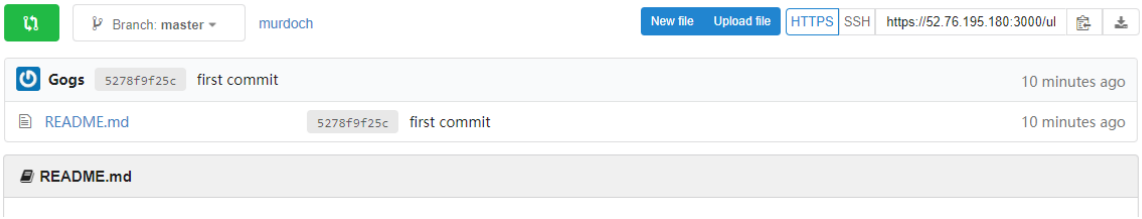
The process to upload and create repositories is quite simple it will be done through ssh.

Go to your dashboard
Click '+' button
Fill out the 'repository name' then create



Go ssh and log in
Type mkdir [foldername]
Cd [foldername path]
Type git init
Type git clone [local path of repository created above]
Type touch README.md
Type git add README.md
Type git commit –m "commit message"
Type git remote add origin [local path of initial created repository]
Type git push origin master

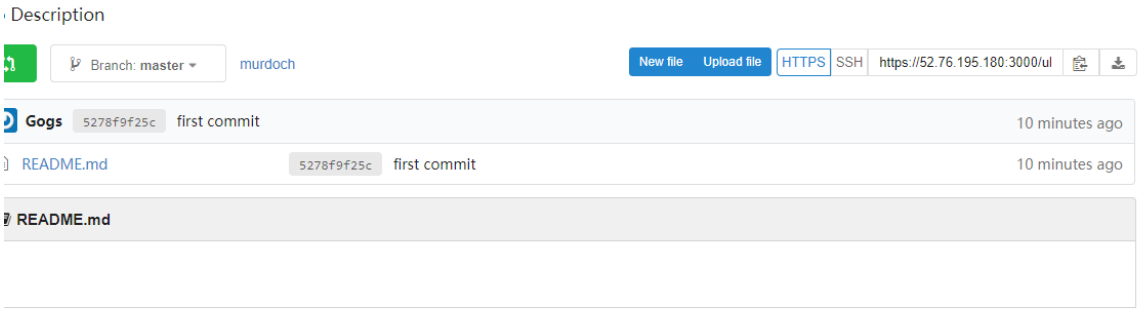
Upon completion of those steps, you can finally see a readme file in repository



5.3 Uploading Files and folders:

There are two different ways of uploading files and folders on your personal git repository. The first method is doing it through the graphical user interface of your git repository website and the second is through the ssh client. I will briefly go over the two methods.

Graphical user interface method:



So, to create a file click → 'New file' while
upload file click → 'Upload file'

SSH method:

There are three essential commands used all the time to upload fields and folders [10]. This Manual assumes you know how to upload files to your server**.** The general rule is: in order to

upload a file to Git repository you must add it → commit it → push it. In addition, it is important to know that to upload folders you need to have at least one file in it and the steps for uploading files and folders are the same.

Go to ssh and be in the directory you created before you cloned
Type git add [filename]
Type git commit –m "[commit message in the quotation marks]"
Type git push origin master
(The above steps apply to folders as well. The difference is a file must be in the folder and we do not add the folder, we add the file in it, so essentially the same steps are applied.)

6.0 Conclusion

I hope you found this manual useful. This report aimed to assist you in your understanding of git repositories and WordPress by providing you with information on how I created my git repository and WordPress website.

7.0 References

[1]"What Is Amazon EC2? - Amazon Elastic Compute Cloud", *Docs.aws.amazon.com*, 2018. [Online]. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html. [Accessed: 09- Jun- 2018].

[2]"Why Move To The Cloud? 10 Benefits Of Cloud Computing", *Salesforce UK Blog*, 2018. [Online]. Available: https://www.salesforce.com/uk/blog/2015/11/why-move-to-the-cloud-10-benefits-of-cloud-computing.html. [Accessed: 09- Jun- 2018].

[3]J. Grubb, "The reason your League of Legends' ping is high: Comcast, Verizon, and AT&T", *VentureBeat*, 2018. [Online]. Available: https://venturebeat.com/2014/12/31/league-of-legends-servers-are-not-over-capacity-so-heres-why-your-ping-stinks/. [Accessed: 09- Jun- 2018].

[4]"Cheap Domain Names-Domain Name Registration-Transfer-Renewal: NameSilo", *Namesilo.com*, 2018. [Online]. Available: https://www.namesilo.com/. [Accessed: 09- Jun- 2018].

[5]T. Davis and →, "How to Install WordPress & LAMP on Ubuntu 16.04 (Vultr) - TechWombat", *TechWombat*, 2018. [Online]. Available: https://techwombat.com/how-to-install-wordpress-lamp-on-ubuntu-16-04-vps/. [Accessed: 09- Jun- 2018].

[6]M. Saxe, "5 Reasons Why It's Important To Update Your Software Regularly - Saxons Blog", *Saxons Blog*, 2018. [Online]. Available: http://www.saxonsgroup.com.au/blog/tech/5-reasons-important-update-software-regularly/. [Accessed: 09- Jun- 2018].

[7]"How To Install WordPress with LAMP on Ubuntu 16.04 | DigitalOcean", *Digitalocean.com*, 2018. [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-lamp-on-ubuntu-16-04. [Accessed: 09- Jun- 2018].

[8]"Using Themes « WordPress Codex", *Codex.wordpress.org*, 2018. [Online]. Available: https://codex.wordpress.org/Themes. [Accessed: 09- Jun- 2018].

[9]"How To Install Gogs On Ubuntu 16.04 – Ken Favors.com", *Kenfavors.com*, 2018. [Online]. Available: https://kenfavors.com/code/how-to-install-gogs-on-ubuntu-16-04/. [Accessed: 09- Jun- 2018].

[10]"git - the simple guide - no deep shit!", *Rogerdudler.github.io*, 2018. [Online]. Available: http://rogerdudler.github.io/git-guide/. [Accessed: 09- Jun- 2018].