## Steps based off screenshots (in /snaps folder)

### The words in brackets indicate the snapshot the step is referring to

### Requires the PUTTY application package

# The Virtual Linux Amazon Instance

1. Prior to being able to use the Amazon Web Services, it is necessary to create a new (or use an existing) generic Amazon account. It is a requirement of the AWS sign-up process to link this account as the services will be tied to it. The sign-up process does not provide you with full access to the AWS services right away, as it is seen as a kind of request form for which you will need to wait on a response from and/or keep in contact with Amazon themselves regards if and when they allow you to use these services.
2. It may take some weeks, though when the AWS are linked to your account there is a “free tier” program available for usewhich can greatly help minimise the costs when using (in this case) virtual machines. Options which are eligible for this are indicated clearly on each step of the instance creation process, so there is strong assurance that the right choice is being made cost-wise. When you are ready to create a new virtual machine, simply click Launch Instance on the top-left (EC2-1).
3. For the purposes of this project, Amazon Linux 64-bit should be selected (indicated as part of the Free-Tier program) (EC2-2). All selections in the following steps will assure that the Free Tier program is being applied wherever available.
4. The 1GB t2 micro memory instance type should be selected (EC2-3).
5. Leave Instance Details as default, as any changes here could bring about complications or uncertainty in the future when dealing with network details and such (EC2-4).
6. Change the root storage disk size to 30 gb. This should be more than enough space for the simple needs we require, and is of course also eligible for the Free Tier (EC2-5).
7. At this point you are able to name or “Tag” your instance. There are several other tags available to set here though the only one to be concerned with for our purposes is the Name tag. I have called my instance “vlinux” but any other simple name will also be appropriate (EC2-6).
8. Choose the “Select an existing Security Group” radio button, and then select the only group there (named default). (EC2-7).
9. Proceed to launch the instance and create a new key pair, at the same time assigning it a name. Don’t forget to click Download Key Pair here! You can’t download it anywhere else or call up this dialog box again. (EC2-8& EC2-9).
10. Launch this instance, and then click Volumes under the ELASTIC BLOCK STORE category on the left column. Click Create Volume on the top-left corner of this window. (EC2-10).
11. For this volume, choose the same availability zone as the current root drive is allocated to. (EC2-11).
12. Right click this new volume, and click Attach Volume. (EC2-12).
13. Click on the instance field, it should show your instance id as the only item in the list that pops up. Note the Device field value here, this is important. (EC2-13).
14. Open PuttyGen, load in the private key pem file you just downloaded before. Make sure the file types to look for dropdown is set to All Files. If you have unfortunately forgotten to download this pem file from step 9, you will have to re-create the instance.. (EC2-14).
15. Decide whether you want to use a passphrase or not. For a simple project like this one, I did not think it was necessary. (EC2-15).
16. Generate a ppk PUTTY Private Key file from the pem file, and you will be prompted to choose a location to save this key to, I suggest saving it to the same place as the pem file.
17. Back on the instance list on the EC2 page, right click your instance and click Connect (EC2-16).
18. Observe the example line here which demonstrates how to connect to the instance. The first part of this line will be handled by PUTTY itself, we are only interested in the address starting with ec2-user. (EC2-17)
19. Open Putty and feel free to save this as a new session; enter from the “ec2-user….” part into the Host Name field, then click on Data under the Connection category on the left column, and enter ec2-user into the Auto-login username. Open this connection and it should connect.
20. Regards working with the extra EBS volume, you will first need to make a new file system with the location of the drive (sdf in my case, also displayed in the details of the drive when clicked on in the AWS window)
21. In the terminal, type “mkfs /dev/sdf” where sdf is the device name field from step 12.

Make a new directory which maps to this drive, I called mine “bigdata”.

* mkdir /mnt/bigdata
* Mount /dev/sdf /mnt/bigdata

Check disk space using the “df” command.

Verify that the 100gb extra EBS volume has been mounted on the directory you specified above. You will notice this in the same output from the “df” command, as it also displays the size of each drive as well as the directory they are mounted on.(EC2-18).

1. To confirm the drive can be written to, a simple test can be performed. You can change the working directory to the one which the drive is mounted on, list its contents, write the current date to a new file called “x”, output the contents of this file and then list the contents of the folder. The screenshot outlines how to do this. (EC2-19).

# The MYSQL package

1. After connecting to the Linux instance, type “yum list mysql” to show all available packages related to mysql. The screenshot only shows one, ignore the “Excluded Packages” message. (MYSQL-1)
2. Type sudo –s to grant root permissions (logging in as root), then type “yum install (package name)”, in this example “yum install mysql.noarch”. Type “Y” to confirm, then press enter (MYSQL-2)
3. Do the same for the “phpMyAdmin.noarch” package (MYSQL-3), and then we will be able to proceed with setting up the phpMyAdmin management environment.

# PostGres

Initially when trying to create a database table using the phpPgAdmin web management page at (Virtual Amazon IP)/phpPgAdmin, an error will show relating to a column named “spclocation” not existing. Following the simple instructions at <http://frustratedtech.com/post/39936405077/fix-cpanel-phppgadmin-to-work-with-postgres9x> replacing the generic path names etc with the actual values, this issue was fixed and tables could be created from that point.

to list files installed by package 'phpPgAdmin':

rpm -ql phpPgAdmin