COSC349 - Assignment 1

Jack Royal - 6974461

August/September 2019

This is the report section for the COSC349 Assignment 1. All work can be found on my Github
https://github.com/jackroyalgit/COSC349_Assignment1.git

1 Brief Description of Application

The assignment focuses on creating multiple virtual machines (VMs) that interact with one another. The application is very simple in which the user is greeted by a form which contains a number of sections that require input from the user, this data is then inserted into the database. The second part of the application queries the DB and returns information about data that has been inputed in the previous section.

1.1 Your application should be built using at least three VMs that interact

For this assignment i decided to use Vagrant to create a virtualised application. My virtualised application comprises of three VMs, two of the VMs are running web serves specifically Apache web servers while the third VM is running a MySQL database. The process i followed for my development was to first combine all aspects of the application into one virtual machine and from that extend each feature to seperate VMs. From this i then seperated the VMs incrementally until i had all three VMs. First of all i seperated the MySQL database into its own VM which was given the IP: 192.168.33.11 this meant that i now had to connect the websever to this IP rather than the localhost of 127.0.0.1. After this was succesfull the webserver was then seperated into two parts. The first part was the form that inserted user input into the database. The second part was the server that queried the database. Both parts connected to the database server separately through the host IP: 192.168.33.11 and then connected to the database using PDO in PHP. The query server and the website server also link to each other to allow easy access for the user to go between the two servers. The IP's used for these connections were 192.168.33.10 and 192.168.33.12 respectively.

1.2 Explain the expected approximate download for first build and for subsequent builds

The first build really only needs to download the VM box (Linux distro Xenial 64 bit) which will be initially downloaded and then is stored locally for the rest of the development process and this will be reused anytime a VM uses the same box file. This file is around 270mb. The other time spent in setting up the VMs is in the provisioning of the VMs such as the setting up of the MySQL database and the configuration of the Apache servers along with the configuration of port forwarding and other network related configs. I have also seperated my provisioning into seperate provisioning scripts instead of just including the provisioning within the Vagrantfile. This allows for easier provisioning for each of the VMs and cleans up the Vagrantfile to only include network information.

1.3 How to use my application

My application is very simple and intuitive to use (assuming you have Vagrant and Virtual Box installed)

1. Clone the respository from the link `https://github.com/jackroyalgit/COSC349_Assignment1.git`
2. Make sure you are in the cloned repository on your local machine from this you can use the command vagrant up which will initialise the Vagrantfile and the corresponding provisioning that it contains
3. Once the provisioning has all been completed you should be able to connect to the application in your chosen web browser through the address `http://192.168.33.10`
4. At the above address you should see the survey form which you can fill out and then follow the link at the bottom of the page to see the results queried from the database server that the two other VMs are connected to

1.4 Modifications/Extensions to code and restarting the application

Obviously as my application is quite simple there are many improvements or modifications that could be made that can better my current application.

1. One modification that could be made is to add an administration aspect and seperate this into its own VM which will allow you to control the database i.e some way of manipulating tables using CREATE or ALTER statements that can only be controlled by a administration

of the application. This would require a login authentication system in which if an administrator logged into the application they would get redirected to a different VM than if just a regular user logged in and they would get directed to the user side of the application. So you would effectively have a client VM and a administration VM that would obviously present different things. This would allow for much easier manipulation of the database and tables contained in the database. As the administration side and client side have been seperated out into different VMs the Vagrantfile will have to be reprovisioned in order to update the changes in IP and general other additions to each of the VMs. To do this you can use the command *vagrant provision* which will reprovision the entire contents of the Vagrantfile or use *vagrant provision [vm-name]* which will just provision the virtual machine specified.

2. To build on the previous change a further extension that could be made is adding some sort of user registration which controls what kind of survey a user receives. For example the initial welcome page of the application would either invite a user to log in or create an account. If a user created an account they could list what type of surveys they were interested in receiving (potentially a drop down with several options) and their email address and whenever a survey was created under a certain section this would automatically be emailed to the users who had selected that category of survey as one of interest. Furthermore the administration side of the application could have some sort of survey template builder in which you could declare what kind of questions you ask in the survey and this would automatically create a specific table for that survey in the database which would collect all the responses. This would allow for easier creations of surveys and a slightly more automated administration side of the application. Again like above the process of restarting the application would be similar in which you would provision the specific aspects of which VM you changed.

1.5 References

- David Eyers, vagrant-multivm, (2019), GitLab Repository, https://altitude.otago.ac.nz/cosc349/vagrant-multivm

- Neeraj Agarwal, php-data-object, (2018), Article, https://www.formget.com/php-data-object/