Student ID: 5264897

Assignment 1 Report

Overview

This repository demonstrates three Virtual machines running an application. It runs a container, loads up a database and displays the contents of that database on a web home. Each step in creating the repository in documented using git. The majority of the code came from code used in my COSC349 labs. And all files and images used are publicly available.

Contents

This assignment contains a Vagrantfile which provision three virtual machines running Ubuntu Xenial, each with a unique purpose. The three purposes are: A web server, a SQL database server, and a server which runs Docker. The web server displays the contents of the database on Apache via localhost:8080. The server running Docker pulls a container from my Docker account and runs the contents which displays a quirky message in the console of that machine.

The other files are: a setup file for the database which contains SQL code which initialises a relational database with a default set of data; a website configuration file which specifies the location of the contents of the website; a www directory containing a php file which will display the contents of the database in a table when ran, a gitattributes file which ensures the project run smoothly on Windows; a gitignore file which ignores extraneous files when adding the repository; and a pdf which explains the whole thing to a marker.

Function

Outlined briefly in Overview and Contents, the function of this project is: to demonstrate a proof of concept, to explain the project, to get enough marks to live, and maybe to learn something[1]. The function of this application is to store small to moderate amount of (relational) records, provide data processing (presumably to the records) and display the records on a web home (preferably queryably). In actuality, the current iteration of this project struggles to do any of this, however it could be extended to do so.

Creation

I began by initialising Vagrant and git. I added and committed my unadulterated Vagrantfile with appropriate comments. I incrementally added functionality to my repository creating a web server by copying David Eyers's Vagrantfiles in the relevant COSC349 labs. I hand-copied each line of his files. I tested the code regularly, trying to understand each step of the process and fixing mistakes I made in the process. I made four commits and I pushed all of these to github.

Then the holidays arrived and I was locked out of the Computer Science Building and I had to work on my assignment remotely.

My I was able to pull my files off github. After initial confusion installing software I managed to run Vagrant and modify my projects files, however I found pushing files to be a greater challenge. I resolved challenge by creating a new repository which did not include my first few commits, since I was unable to master merging separate repositories (out of fear of destroying my hard work and out of laziness, but mostly laziness.) So that concludes why I am missing my few couple of git commits.

As I scaled up to two VMs, the next challenge was overcoming the slow testing cycle as my computer struggled under the weight of the world, two VMs and Google Chrome. This slow cycle was additionally compounded by being away from my work environment, a 50th birthday party and seasonal laziness. As a result the development cycle slowed down to 1 commit per every so many whenevers. Nevertheless, total development of the project remained at projected estimates, as putting in the dedication and very confused research were correctly predicted to be misspent instead on learning about Norweagen.

As I scaled up to three VMs things looked dire. Fortunately, the holidays were over, and that meant two things! It meant this assignment was due and I had access to lab computers (which I have a pavlovian association of work with). So basically, I went into Varsity and got to work on puzzling out how Docker works. And I made a few more commits. Then I realised that I had no idea how to put files on SQL. So I decided that that was enough work on my project and I knew based on previous experience that the writing would take all evening before the quality and readability fell. Half-baked tangents where abound. And lots of conjunctions cropped up in the text. Overall, the readability of the report was suffering, and the accuracy of exact meaning was lacking. Regretfully, Docker processing Python code did not make it into the final draft, despite working fine in a container playground; this is in part due to design paralysis in deciding where the python code would go and also in part due to spending close to two hours on encapsulating python code into an echo statement on a vim editor of a Dockerfile. Eventually I decided that the equivalent of a hullo would be fine.

At this stage I commented the Vagrantfile and everything went fine. I wrote my report as happy as Larry, in which everything went swimmingly. And no editing was required.

The build reliably takes 6 min 45 sec to build.

Extension

The application could be extended to include automatic IP address allocation to avoid address conflicts of new Virtual Machines. Javascript could be enabled in the web home which could allow for querying the database. A separate server could be made to handle new data using Docker for data proccessing. Presumably data will come from connected web services and

would track things like user data or store scientific data or something data-ery which lends itself to being stored in a relational database. Data entry could be handled through Javascript or internally via Python implemented with Docker containers. Data processing will be Python making scatter plots (very advanced), or R working out F values for scientific data (and another container could reject the null hypotheses creating another statistic to record).

In order for the VMs to communicate with each other they could use a shared folder, or they could establish a secure connection with each other, or they could use Dropbox, or (unwisely) they could be set to public visibility and anyone could alter them! Alternatively the VMs could use their private network to communicate, but at this point all computer science jargon looks like Greek to me. The SQL password could be more secure, using a salted hash or something.

Conclusion

So basically my application doesn't do much and if I had better time management I could have extended its functionality but quite a bit, and all my technical knowledge with any of these technologies is limited to what I could pick up in the labs. Maybe I'm being pessimistic and maybe my mental health isn't as good as it could be, but I learnt some stuff in my frantic final effort to finish. The core lesson is time management is important and I have seen just about every permutation of the word destory.

1. Although this is an unlikely outcome, it is likely enough to not be ruled out with Bayesian Inference within a sensible margin of error[2].
2. The definition for sensible margins of error is exactly a million-to-one[3].
3. Because traditionally, one has to say "it's a million-to-one chance, but it might just work!" and it will succeed nine out of ten times[4].
4. Guards! Guards! (Terry Parchett, 1989)