COSC349 Assignment 1: effecting the portable building and deployment of software applications using virtualisation

Submission information

Due date: Monday, 31st August 2020, at 11:59 PM.

Weight: This assignment is worth 20% of the mark for the paper.

Participation: You are permitted to work individually or in pairs.

How you will make submissions: An assignment on the COSC349 Blackboard section will collect the following two pieces of information:

1. the URL to a git repository—or even better, to a specific git commit, git branch or git tag—that contains the version of your project's files that you want to be marked;
2. a project report in PDF format.

All students must make their own personal submission within Blackboard, however members of pairs will both submit exactly the same material.

Requirements:

- You are expected to work on your assignment within a git repository. The easiest way to acquire URLs that reach your repository is to push your work to a (free) cloud-based git service, such as GitHub, Bitbucket, GitLab, or Altitude (a GitLab server run locally within the Department of Computer Science that you can access). The aforementioned cloud services allow you to create private repositories.
- You must invite teaching staff to collaborate on your repository, so that they can access the URL that you submit. Inviting David is sufficient: he is user `dme26` on GitHub, Bitbucket, and GitLab. Otherwise you can send an email invitation to `dme@cs.otago.ac.nz`.
- Do not apply git history-rewriting operations to your repository: it is important that the markers can see how your commits completed your assignment, and how you worked on the assignment over time.
- Your repository must include all of the files required to build your application, other than stable, publicly-available, external resources downloaded automatically during your build process. For example you do *not* need to include Vagrant box files, or data that can be downloaded from cloud-based repository servers (e.g., GitHub, etc.), but be careful to test that such material is being successfully fetched by your build process just before you declare your project complete.
- Your repository should be set up so as to allow a new developer to join your project, and to rapidly get up to speed regarding how to use and extend your work. Ideally your repository will not contain too many indicators that it is aiming to satisfy a university assignment.

- Your project report is where you can communicate with your markers, explaining aspects of your project that might not be reflected clearly within your repository, such as unsuccessful attempts that you might have made to use particular technologies. The marking guide, below, indicates a number of points that your report must cover.

Late submissions: The latest commit date of the git branch, tag, commit, or repository reached by the URL that you submit will be taken as your submission timestamp. COSC349 applies a late submissions policy typical of other COSC papers. Late submissions will incur a 10% penalty per working day, rolling over at 11:59 PM. Submissions that are more than five days overdue will not be accepted.

Learning objectives

The aims of the assignment include the following learning objectives:

- to demonstrate that you understand how virtualisation can usefully underpin application development; (This is also particularly true of application development for cloud computing.)
- to employ, in practice, fundamental concepts in cloud-ready software development, such as automated provisioning of virtual machines;
- to carry out research to discover open source projects and materials that you can use within your application development work; and
- to deliver software that is well tested and documented.

Problem Description

For this assignment you should design and develop an application whose build process relies on virtualisation. Ideally, your application should operate through coordination of multiple virtual machines (VMs).

However, this assignment is not focused on the functionality of your application: the assignment is focused on how you *build* your application, and in particular **how you facilitate other developers potentially modifying, building and running your application**. Your use of virtualisation should allow a developer to build and run your application even if they check out your project's git repository on a different host operating system from yours.

The assignment has been designed with use of Vagrant in mind, although use of other tools available on the Computer Science Lab F computers, such as VirtualBox (i.e., used directly, rather than via Vagrant), is acceptable too.

The easiest way for a virtualised application to provide an interface to (non-developer) users is through the use of web technologies. Because this is not a web technology paper, marking will not focus on the details of your web implementation. Any other form of interface is acceptable too, provided that it does not require your users to have advanced computing qualifications to use it.

You are welcome to use others' code within software components and as a starting point for your assignment, but you still need to build an application of your own design. Any use of others' resources must be attributed by you in your in-code documentation and your project report. Further, the history of commits in your git repository will be examined to determine what, and how much code and configuration you added into your project.

A developer must be able to modify your application by changing source files in their clone of your git repository, and then be able to rebuild their instance of your application on their virtualisation host.

Example applications

If you cannot think of an application to build, some suggestions are listed below. You may want to consider developing a project that you can present to future potential employers. Also, given that the assignment does not focus on the application itself, you are welcome to rework and extend material that you have completed for other assignments within Computer Science or Information Science papers, for example.

- A world timezone converter: one VM could run a web interface, another VM could run a database storing users' specific display preferences, and a third VM could provide tools for batch querying of timezone conversions.
- A customised editor of some sort: e.g., shared note-taking tools, or an assistant for recording minutes of meetings. Different VMs could run a web interface for the editor, a database server, and a tool for producing reports, or calendar files of due dates for items.
- A site for visualising data: one VM could provide the interface users use to upload their data and retrieve their results. Another VM could store the users' data in a database or other storage system. A third VM could provide administrative functionality.

Rough marking guide

The assignment is worth 20% of your COSC349 mark but will be marked out of 100. A breakdown of marks is included below. This is not a strict marking scheme, as having one has been found to tend to disadvantage students. For example, if you are unable to achieve one of the points in the marking breakdown below, you may be able to gain compensatory marks by explaining in your project report what you intended to do, what problem arose, and what compromise you reached. Also, it may be considered that you have gone well beyond what was expected for part of the assignment, and should be awarded additional marks for a section, thus helping compensate for some minor problems elsewhere.

20% Your application should be built using at least three VMs that interact.

- In your project report, explain the purpose of each of your VMs, why they are separate VMs, and how they interact.
- Your application needs to be of your design, even if you use others' open-source components to help build it.

10% The build system for your application should minimise use of large downloads that are specific to this application. Widely-available, popular, open-source disk images and other open source materials can (and should) be used within your build system.

- In your project report, explain the expected approximate download volumes for both the first build, and for subsequent builds.

20% After starting your application's build process, your application should build and start, unattended.

- Your project report can explain why manual interaction is required in your build process, if you cannot achieve entirely unattended building.
- Your repository should default to preloading test data for demonstration purposes into your application, rather than requiring significant amounts of manual interaction with your application's user interface before useful results are seen.

10% In your project report, describe briefly how a user should use your application. You may instead include a link to a screencast showing use of your application, provided that it is shorter than five minutes in duration.

20% In your project report or elsewhere in your repository, highlight at least two useful types of modification or extension a developer can make to your application's code, and explain how to rebuild and rerun the application after they have made these changes.

20% The way in which you incrementally developed and debugged your application is clear, from examining the commit history in your git repository.