A Report submitted in partial fulfilment of
the regulations governing the award of

the Degree of

BSc (Honours) Computer Networks and Cyber Security

at the University of Northumbria at Newcastle

Project Report

Your Project Title

James Poxon

2020/2021

General Computing Project

Declaration

I declare the following:

1. that the material contained in this dissertation is the end result of my own work and that due acknowledgement has been given in the bibliography and references to ALL sources be they printed, electronic or personal.
2. the Word Count of this Dissertation is ⟨len⟩
   (result of shell command)
3. that unless this dissertation has been confirmed as confidential, I agree to an entire electronic copy or sections of the dissertation to being placed on the eLearning Portal (Blackboard), if deemed appropriate, to allow future students the opportunity to see examples of past dissertations. I understand that if displayed on eLearning Portal it would be made available for no longer than five years and that students would be able to print off copies or download.
4. I agree to my dissertation being submitted to a plagiarism detection service, where it will be stored in a database and compared against work submitted from this or any other School or from other institutions using the service.

   In the event of the service detecting a high degree of similarity between content within the service this will be reported back to my supervisor and second marker, who may decide to undertake further investigation that may ultimately lead to disciplinary actions, should instances of plagiarism be detected.
5. I have read the Northumbria University/Engineering and Environment Policy Statement on Ethics in Research and Consultancy and I confirm that ethical issues have been considered, evaluated and appropriately addressed in this research.

SIGNED: . . . .

Acknowledgements

Abstract

A summary of the entire project. From background to conclusions. I recon on about half a page as the upper end of the summary.

This is an example structure for the Terms-of-Reference and the Dissertation. Along with some notes.

You can start by forking the repository on github `https://github.com/dr-alun-moon/cs-dissertation`. Then you have a working copy of this document as a starting point.

Contents

Chapter 1

Introduction

This document is split into several files. This makes the editing easier, if the file management a little harder.

The two principle means of including sub parts into a main document are the LaTeX commands `\input` and `\include`

1.1 Splitting the Input

The input command simply reads that file in, at that point in the main file. The result is as if the inputed file was pasted in at the point of the command.

The include command is a little more complex. The main point to note is that it forces a new page, then reads in the file. It is good foe content that needs to start a fresh page, such as chapters. As a rule of thumb I put each chapter in an included file.

1.1.1 Some tricks.

The Dissertation uses the book class, that gives the Chapter as the top sectional element, followed by section, subsection, paragraph. The Terms-of-reference uses the article class, which starts at the section level.

By writing the ToR as a driver document that handles all the setting up, and then inputing the tor content from a separate file. When it comes to including the tor as an appendix in the dissertation, you can start a chapter then just input the tor content file.

1.2 Magic comments

Since LaTeX needs to have a certain amount of setup (known as the preamble), this is the file that needs to be build. From a Makefile or command-line, this is simple enough.

Several editors allow you to trigger the build from within their environment. Here the file you are editing is a subpart and not the mater document that needs to be passed to LaTeX. The editor needs to know which `.tex` file is the master document. Some editors recognise a magic comment placed at the top of a sub-file.

This informs the editor that `Dissertation.tex` is the file to build to rebuild the document. I can save changes and just press the key combination set to trigger a LaTeX build, and the editor knows how to do the rest.

Part I

Analysis

Chapter 2

What is Analysis and Containerisation?

Dario Taraborelli, has written a nice page illustrating some of The Beauty of LaTeX illustrating some of the finer points of TeX et.al. typesetting

`TeXample.net` is a site that has many spectacular examples[1] of graphics creation in LaTeX. It mainly illustrates the use of the tikz package.

Two web sites provide forums for questions and answers on latex. LaTeX Community and TeX - LaTeX Stack Exchange

2.1 References and searching

LaTeX and BibTeX provide excellent support for referencing and citations (especially with natbib[2]). Getting your .bib file populated with material can be time consuming. Among the many ways of doing this are:

- Search engines like the University Library allow you to export your saved searched in a bibTeX file
- Google scholar https://scholar.google.co.uk/ have an option to create the BibTeX entry for results.

[1] `http://www.texample.net/tikz/examples/`

[2] see online manual at `http://mirror.ox.ac.uk/sites/ctan.org/macros/latex/contrib/natbib/natbib.pdf` or with the command `texdoc natbib`

- Zotero provides a plugin for Firefox, that extracts information from a web page and exports a bibTeX file. (Great for getting references to Wikipedia, or getting book details from Amazon)

These are great, but sometimes the bibTeX file needs a little post-editing. Usually I end up deleting extraneous fields and escaping TeX characters if needed.

1. Search for articles, books etc, grab the bibTeX file,
2. `\backslash cite{}`and use
3. `\bibliography{save1,litrev,canon}` with
4. `\bibliographystyle{plainnat}` (having `\usepackage{natbib}` in the preamble)...

Harvard referencing of your material... job done

2.2 Writing your dissertation

The most valuable document is the *Project Handbook*, this gives guidance on the structure and contents of the Terms-of-Reference and Dissertation. It also has the marking scheme, which is an important read as this describes what we are looking for when marking the project.

2.2.1 The Logbook

Use your logbook and a project diary, make notes of what you do as you do them. As questions arise, make a note of them at the time, and then you are ready for the weekly meeting.

In the meeting make notes of whet we discuss for the week ahead, and issues that come up that can be looked at later.

Some students in the past have kept a project blog.

2.2.2 The Dissertation

There are several good sites about with advice on how to write clearly. Not all of the advice is directly relevant to writing a B.Sc dissertation in Computer Science. The advice may not even be consistent,

William Stallings [?] maintains a good website of resources for Computer science students.

Chapter 3

What is Analysis and Containerisation?

Dario Taraborelli, has written a nice page illustrating some of The Beauty of LaTeX illustrating some of the finer points of TeX et.al. typesetting

`TeXample.net` is a site that has many spectacular examples[1] of graphics creation in LaTeX. It mainly illustrates the use of the tikz package.

Two web sites provide forums for questions and answers on latex. LaTeX Community and TeX - LaTeX Stack Exchange

3.1 References and searching

LaTeX and BibTeX provide excellent support for referencing and citations (especially with natbib[2]). Getting your .bib file populated with material can be time consuming. Among the many ways of doing this are:

- Search engines like the University Library allow you to export your saved searched in a bibTeX file
- Google scholar https://scholar.google.co.uk/ have an option to create the BibTeX entry for results.

[1] `http://www.texample.net/tikz/examples/`

[2] see online manual at `http://mirror.ox.ac.uk/sites/ctan.org/macros/latex/contrib/natbib/natbib.pdf` or with the command `texdoc natbib`

- Zotero provides a plugin for Firefox, that extracts information from a web page and exports a bibTeX file. (Great for getting references to Wikipedia, or getting book details from Amazon)

These are great, but sometimes the bibTeX file needs a little post-editing. Usually I end up deleting extraneous fields and escaping TeX characters if needed.

1. Search for articles, books etc, grab the bibTeX file,
2. `\backslash cite{}`and use
3. `\bibliography{save1,litrev,canon}` with
4. `\bibliographystyle{plainnat}` (having `\usepackage{natbib}` in the preamble)...

Harvard referencing of your material... job done

3.2 Writing your dissertation

The most valuable document is the *Project Handbook*, this gives guidance on the structure and contents of the Terms-of-Reference and Dissertation. It also has the marking scheme, which is an important read as this describes what we are looking for when marking the project.

3.2.1 The Logbook

Use your logbook and a project diary, make notes of what you do as you do them. As questions arise, make a note of them at the time, and then you are ready for the weekly meeting.

In the meeting make notes of whet we discuss for the week ahead, and issues that come up that can be looked at later.

Some students in the past have kept a project blog.

3.2.2 The Dissertation

There are several good sites about with advice on how to write clearly. Not all of the advice is directly relevant to writing a B.Sc dissertation in Computer Science. The advice may not even be consistent,

William Stallings [?] maintains a good website of resources for Computer science students.

Chapter 4

Tools

4.1 LaTeX

I recommend the TeXlive distribution, which can be downloaded from `https://www.tug.org/texlive/`. There are installation packages for Windows, Linux, and Macs here. For Linux machines there is often a copy of texlive in the package repository.

Figure 4.1: Searching for texlive on Debian/Ubuntu systems

Edit 2018 on my machine at work I've dropped the Ubuntu version of texlive as it didn't pull in all the documentation for the packages (see section 3.7.3)

4.2 Editors

Any text editor will do. I wrote my MSc dissertation in `vi`[1] [2]! I've gone full circle and returned to using `vi` for latex at work and at home.

In practice, many modern editors have a rich set of tools to help with the editing process. From syntax-highlighting, auto-completion, to rebuilding.

[1]`https://en.wikipedia.org/wiki/Vi`

[2]Hardcore Unix use

4.2.1 My setup

Currently I'm using `vim` to edit this document, and `latexmk` to build the pdf. I've two terminals open, in one I have the editor and the other the command

The options are

`-pdf` makes sure a pdf is generated via `pdflatex`

`-pvc` monitors the `.tex` files and reruns the build process if changes are made, it automatically opens and updates the pdf viewer as needed.

`-latexoption=-shell-escape` because I'm using the `minted` package for syntax highlighting, the `-shell-escape` option is passed on to `pdflatex`. Minted uses pygmentize as an external program to perform the highlighting.

4.2.2 Vi/Vim+vimtex

Vim `https://www.vim.org/` the modern incaration of `vi` has an extension package `vimtex` `https://github.com/lervag/vimtex`. I'm finding these more than adequate for my editing needs. Vi is part of the POSIX standard and is either going to be installed or very easy to install on any Unix like system. Windows users have several choises on how to use vim and latex.

4.2.3 Atom

As it is a very extensible editor, there are a number of add-ons that help with latex.

language-latex `https://atom.io/packages/language-latex` which provides syntax highlighting.

latex `https://atom.io/packages/latex` which provides means of compiling latex documents from within the editor.

pdf-view `https://atom.io/packages/pdf-view` which is a PDF viewer. (I'm not sure about this one, I can't tell if it is too much of a strain on the editor)

4.2.4 TeXworks and TeXShop

The TeXShop editor on the Mac inspired TeXworks on Windows. This is a nice little editor with a good pdf previewer built in.

4.2.5 Texmaker

This looks like a nice clean editor for LaTeX. It has usefull pallets of commands.

4.2.6 TeXnicCenter

An editor with a lot of tools. A capable system.

4.2.7 Online

Two online web based editing web-applications are

- ShareLaTeX `https://www.sharelatex.com/`
- Overleaf `https://www.overleaf.com/`

4.3 PDF viewers

Latex generates pdf files out of the box. There are some good PDF viewers about. The ones below integrate nicely with the Atom latex tools.

4.3.1 Skim – OSX

The viewer for Macs `http://skim-app.sourceforge.net/`.

4.3.2 Sumatra PDF – Windows

A good viewer for Windows based machines
`https://www.sumatrapdfreader.org/free-pdf-reader.html`.

A note about Adobe Acrobat on Windows machines. On windows machines Acrobat locks the file when it opens it, this means that pdflatex and tools cannot write to the file to rebuild it.

4.4 SyncTeX integration

In the settings for the latex plugin for Atom, and the various PDF viewers, you'll see settings for something called SyncTeX. This does two useful things.

Firstly it allows the editor to move the document to the position the cursor is at in the text. In Skim and Sumatra the viewer displays a coloured dot corresponding to the position of the cursor.

Second and more useful is allows the PDF viewer to control the cursor position int he editor. Click on a location in the PDF, and the cursor in the editor is moved to the corresponding file and location in the sources.

4.5 GitHub

Treat the documents as code. Put the ToR and Dissertation under version control. It also makes moving your work between university and home easy. Just commit all the changes, push the repository back to the server, then pull the repository at the other end. It also means you have a copy backed up.

Sign up for the Student Developer Pack See `https://education.github.com/` and `https://education.github.com/pack/`. They will give you unlimited private repositories (normally $7/month) while you are a student.

4.6 Perl

Some of the *very* useful tools in texlive, need a Perl installation to work. In Linux and Macs, perl should be there automatically. For windows although texlive does install a minimal Perl set, it isn't enough for the really usefull tools (see 3.7.1 and 3.7.2).

On Windows machines install Perl before TeXlive. Make sure that the `Perl.exe` is in the system PATH before installing TeXlive. My Windows installation of Perl is Strawberry Perl `http://strawberryperl.`

`com/`.

4.7 TeXlive utilities

There are a couple of very utilities that come with TeXlive, they do need a complete Perl installation (see 3.6).

4.7.1 latexmk

Latex needs several passes through to collate and use cross references. It also needs a pass by BibTeX to compile the reference list, and latex passes to put the citations in.

`latexmk` is a make like program that understands latex, it can parse the log files generated and re-run the appropriate tools until a build is complete.

4.7.2 texcount

`texcount` parses the docment, following any inputed or included files, and counts the words used. Being TeX aware it knows how to ignore the markup.

4.7.3 texdoc

Latex is *very* well documented. To find the documentation for a package, which are loaded in the preamble. Use `texdoc` with the package name.

4.8 Listings

There are several packages that layout code listings, complete with syntax highlighting. Their advantage is that they can read in and highlight a source-file in the appropriate language.

4.8.1 minted

I use the `minted` package, it does require the use of an external program. You will need to have Pygments `http://pygments.org/` installed, which depends on having a Python installation `https://www.python.org/`.

In running latex you'll need to enable shell-escape.

Or see the settings for the latex Atom package

4.8.2 listings

Listings is an older package. It doesn't produce coloured output, but it is written in tex, so it has no external dependencies.

Chapter 5

Tools

5.1 LaTeX

I recommend the TeXlive distribution, which can be downloaded from `https://www.tug.org/texlive/`. There are installation packages for Windows, Linux, and Macs here. For Linux machines there is often a copy of texlive in the package repository.

Figure 5.1: Searching for texlive on Debian/Ubuntu systems

Edit 2018 on my machine at work I've dropped the Ubuntu version of texlive as it didn't pull in all the documentation for the packages (see section 3.7.3)

5.2 Editors

Any text editor will do. I wrote my MSc dissertation in `vi`[1] [2]! I've gone full circle and returned to using `vi` for latex at work and at home.

In practice, many modern editors have a rich set of tools to help with the editing process. From syntax-highlighting, auto-completion, to rebuilding.

[1]`https://en.wikipedia.org/wiki/Vi`
[2]Hardcore Unix use

5.2.1 My setup

Currently I'm using `vim` to edit this document, and `latexmk` to build the pdf. I've two terminals open, in one I have the editor and the other the command

The options are

`-pdf` makes sure a pdf is generated via `pdflatex`

`-pvc` monitors the `.tex` files and reruns the build process if changes are made, it automatically opens and updates the pdf viewer as needed.

`-latexoption=-shell-escape` because I'm using the `minted` package for syntax highlighting, the `-shell-escape` option is passed on to `pdflatex`. Minted uses pygmentize as an external program to perform the highlighting.

5.2.2 Vi/Vim+vimtex

Vim `https://www.vim.org/` the modern incaration of `vi` has an extension package `vimtex` `https://github.com/lervag/vimtex`. I'm finding these more than adequate for my editing needs. Vi is part of the POSIX standard and is either going to be installed or very easy to install on any Unix like system. Windows users have several choises on how to use vim and latex.

5.2.3 Atom

As it is a very extensible editor, there are a number of add-ons that help with latex.

language-latex `https://atom.io/packages/language-latex` which provides syntax highlighting.

latex `https://atom.io/packages/latex` which provides means of compiling latex documents from within the editor.

pdf-view `https://atom.io/packages/pdf-view` which is a PDF viewer. (I'm not sure about this one, I can't tell if it is too much of a strain on the editor)

5.2.4 TeXworks and TeXShop

The TeXShop editor on the Mac inspired TeXworks on Windows. This is a nice little editor with a good pdf previewer built in.

5.2.5 Texmaker

This looks like a nice clean editor for LaTeX. It has usefull pallets of commands.

5.2.6 TeXnicCenter

An editor with a lot of tools. A capable system.

5.2.7 Online

Two online web based editing web-applications are

- ShareLaTeX `https://www.sharelatex.com/`
- Overleaf `https://www.overleaf.com/`

5.3 PDF viewers

Latex generates pdf files out of the box. There are some good PDF viewers about. The ones below integrate nicely with the Atom latex tools.

5.3.1 Skim – OSX

The viewer for Macs `http://skim-app.sourceforge.net/`.

5.3.2 Sumatra PDF – Windows

A good viewer for Windows based machines `https://www.sumatrapdfreader.org/free-pdf-reader.html`.

A note about Adobe Acrobat on Windows machines. On windows machines Acrobat locks the file when it opens it, this means that pdflatex and tools cannot write to the file to rebuild it.

5.4 SyncTeX integration

In the settings for the latex plugin for Atom, and the various PDF viewers, you'll see settings for something called SyncTeX. This does two useful things.

Firstly it allows the editor to move the document to the position the cursor is at in the text. In Skim and Sumatra the viewer displays a coloured dot corresponding to the position of the cursor.

Second and more useful is allows the PDF viewer to control the cursor position int he editor. Click on a location in the PDF, and the cursor in the editor is moved to the corresponding file and location in the sources.

5.5 GitHub

Treat the documents as code. Put the ToR and Dissertation under version control. It also makes moving your work between university and home easy. Just commit all the changes, push the repository back to the server, then pull the repository at the other end. It also means you have a copy backed up.

Sign up for the Student Developer Pack See `https://education.github.com/` and `https://education.github.com/pack/`. They will give you unlimited private repositories (normally $7/month) while you are a student.

5.6 Perl

Some of the *very* useful tools in texlive, need a Perl installation to work. In Linux and Macs, perl should be there automatically. For windows although texlive does install a minimal Perl set, it isn't enough for the really usefull tools (see 3.7.1 and 3.7.2).

On Windows machines install Perl before TeXlive. Make sure that the `Perl.exe` is in the system PATH before installing TeXlive. My Windows installation of Perl is Strawberry Perl `http://strawberryperl.`

`com/`.

5.7 TeXlive utilities

There are a couple of very utilities that come with TeXlive, they do need a complete Perl installation (see 3.6).

5.7.1 latexmk

Latex needs several passes through to collate and use cross references. It also needs a pass by BibTeX to compile the reference list, and latex passes to put the citations in.

`latexmk` is a make like program that understands latex, it can parse the log files generated and re-run the appropriate tools until a build is complete.

5.7.2 texcount

`texcount` parses the docment, following any inputed or included files, and counts the words used. Being TeX aware it knows how to ignore the markup.

5.7.3 texdoc

Latex is *very* well documented. To find the documentation for a package, which are loaded in the preamble. Use `texdoc` with the package name.

5.8 Listings

There are several packages that layout code listings, complete with syntax highlighting. Their advantage is that they can read in and highlight a source-file in the appropriate language.

5.8.1 minted

I use the `minted` package, it does require the use of an external program. You will need to have Pygments `http://pygments.org/` installed, which depends on having a Python installation `https://www.python.org/`.

In running latex you'll need to enable shell-escape.

Or see the settings for the latex Atom package

5.8.2 listings

Listings is an older package. It doesn't produce coloured output, but it is written in tex, so it has no external dependencies.

Part II

Synthesis

Chapter 6

Some TeXnical Details

6.1 Structure of the file set

The Terms of Reference and Dissertation are split into several files, to ease editing, and exploit some of LaTeX's capabilities. The structure is relatively *flat* with little hierarchy of directories, directories and sub-directories can be added to simplify some of the structure as the project grows.

The principle files are:

`Makefile` I use `make` still as my build driver. Any automated build system can work with LaTeX files, it just needs to know that PDFs are build from `.tex` files.

`TermsOfReference.tex` This is the main file for creating the *Terms of Reference.* It pulls in the `tor.tex` file, the Gantt chart from `gantt.tex`, and the ethics and risk assessment forms from scanned PDFs. The document is typeset as an `article`.

`Dissertation.tex` Is the main file for creating the dissertation. This pulls in a number of other files as required. It is typeset as a `book`. The sectioning starts as `\chapter` and has `\section` within. This way the terms of reference can be included as a chapter in the appendix. The chapters are each included using `\include`, this allows the chapters to be written as separate files. The difference between

`\include` and `\input` is that `\include` forces a new right-hand page to start (good for starting chapters).

6.1.1 Subsidiary files

`tor.tex` This file is the main *Terms-of-Reference* file. By using `\input` in the `TermsOfReference.tex` document, this gets included and typeset.

`gantt.tex` The Gantt chart is a little complex, so gets put into a separate file, see section 4.1.2.

6.1.2 The Gantt Chart

The Gantt chart in the Terms-of-Reference is drawn with another latex package. It uses an `input` command to pull it into the tor (and dissertation).

Most of the file `Gantt.tex` is the setup of the Gant chart, in order to make it fit in one page. The bottom section is where you can define the tasks. Each bar has a title, a start date, and an end date. Milestones have a title and a date. The `ms` option can be set to left or right to control which side of the milestone mark the text label.

Part III

Evaluation

Part IV

Appendices

Appendix A

Terms of Reference

A.1 Background

Virtualisation has been utilised by a number of industries for a long time now, with first iterations of virtual machines dating back to IBM in the 1960s [?]. System Virtual machines allow an operating system to emulate the function of a full operating system layered on top of a base operating system. Functionally, this allows multiple different logical 'computers' with varying operating systems to run on one physical computer. In most modern implementations, virtualisation requires software (known as a hypervisor) to manage and create the virtual machines.

Whilst I was on a year-long placement provided as part of a sandwich course at my university, I had the chance to work in an IT risk department at a reputable enterprise in Newcastle Upon Tyne. whilst working there I witnessed first hand, infrastructure and operations departments using virtualisation for much of the internally and externally facing server infrastructure, in what was an unwieldy and cumbrous use of scripts to update and install patches and dependencies across a multitude of systems. This resulted in a number of incidents where systems had to be pulled down in order to update the Operating Systems of individual virtual machines. These methods often caused unnecessary down time for systems, resulting in substantial risk for the business. Furthermore, this method often put a substantial stress on the hardware of these systems, with hardware boxes often running at full resource potential under heavy load, and whilst these boxes were designed to withstand these kinds of loads, it still affected the longevity of the hardware.

In recent years there has been research into the use of containers instead of virtual machines for various operations across computing industries [?]. Containers are different from virtual machines in that they run on the base OS, and don't require a secondary emulated operating system to be managed by a hypervisor. This is a benefit as it reduces the resources [?] used by each instance. Overall, containers are a more lightweight and efficient system, that are easier to keep up to date. Whilst research has claim in displaying the benefits of containerisation, much of the server infrastructure of enterprise remains reliant on Virtualisation, not Containerisation.

I have also had the opportunity to partake in gatherings held by CoTech, a network of tech co-operatives that meet semi-regularly to discuss various tech related topics. Whilst I was there, it was discussed that a large portion of these small enterprises used Docker, a system for packaging and deploying containers, to develop applications for their clients.

There is historical research that compares virtualisation and containerisation for other technical applications, for example [?] compares the two methods inside the context of PaaS (Platform as a service), which is similar to co-techs use of Docker. It seems that the application for containerisation has been realised as early as 2014 when it is being applied to the development and hosting of applications, but not as much when talking about the running of wider server infrastructure. This is further supported by research from '451 Research' who in 2017 estimated a compound annual growth rate of 40% for containers in 2020 [?], suggesting a coming paradigm shift from virtualisation to containerisation.

A.2 Proposed Work

I plan to do similar work to the studies I have already mentioned [?], [?], but instead focus this work on my own context and my own interest in Operating Systems and Server Infrastructure. I have experience with running headless servers on virtual machines already, through the Advanced Operating Systems module I did in my second year of study at Northumbria University.

It is important to set a baseline system here, and to best reflect a realistic topology, I will use LAMP (Linux + Apache + MySQL + PHP). I have experience with Ubuntu server, so I will use this version of Linux as the base operating system for my virtual machines, and plan to host a full working topology of servers, including a DNS architecture, a web-server architecture that includes HTTP (Apache) servers, NFS servers and MySQL servers. The reasoning for doing this is to create as realistic a topology as possible, and to ensure a somewhat realistic level of network traffic so that meaningful data can be captured.

For the virtual machines, I will use the virtual machine infrastructure available to me through the university, and for the container infrastructure, I plan on using Docker. This is because Docker is a popular choice among app developers, and is supposed to make container deployment easy. Though, as I have never worked with containers before, if Docker fails to be a good way of providing containers for web-servers, then there are a number of other ways to deploy containers, such as LXC, that I could turn to as a contingency.

I will also need to set a standard for measurement to ensure that my data is meaningful and uses the scientific method. I plant to use tools such as iPerf3 (for network performance) and Sysbench (for hardware performance) across a number of servers. It is important to measure both network and hardware performance (CPU, Memory, Disk, etc) to ensure that described benefits are achieved for the system as a whole. iPerf is an industry standard tool for measuring network performance on Linux systems, whilst Sysbench is a full benchmarking suite for linux that will let me benchmark the CPU, file IO, and importantly, MySQL performance (allowing direct measurement of database performance on the machine that will be hosting the previously mentioned MySQL database. Whilst I have mentioned these benchmarking tools, it is possible that they could have problems with integrating with certain applications or environments, in this case, there is a number of other standard benchmark utilities (Phoronix Test Suite, KDiskMark, UnixBench, etc) available that should give me the flexibility to create measurements. Whilst these utilities all have differing overheads within them (meaning they will use up varying resources to run the benchmark), as long as I use the same benchmark across the same machines I am comparing, this overhead shouldn't effect the measurable performance difference between these machines.

Once data is collected, I will do a comparative analysis of the data to determine if there is a clear improvement as previous research suggests, and if this difference is enough to justify a change in the current best-practice use of virtualisation.

A.3 Aims and Objectives

A.3.1 Aims

To compare and contrast the performance difference and hardware impact between virtualisation and containerisation when running headless servers.

A.3.2 Objectives

Your objective list is a series of measurable objectives, can you tick each one off as *done*? I usually expect between 8 and 12 objectives

1. **Explain the problem domain that encompasses current practices in virtualisation.**
2. **Explanation of the two different methods in practice.**
3. **Determine the software and hardware to be used.**
4. **Design the network infrastructure to be built.**
5. **Learn how to implement and utilise containers using Docker.**
6. **Build the network topology on the virtual machines.**
7. **Build the network topology on the container system.**
8. **Determine a method to scientifically evaluate and determine performance of both systems.**

9. **Accurately measure performance of the two systems.**
10. **Compare and contrast between the findings for both systems to determine improvements or failings.**
11. **Create a recommendation regarding the real-world implementation of containerisation in this use case.**

A.4 Skills

This is where you can cover the skills you have relevant to the project and the new skills you are going to acquire during the project.

1. Advanced Operating Systems 1, see module KF5004.
2. Computer Networking Experience.
3. Computer Technology, see module KF4004.
4. Virtual machine configuration.
5. LAMP topology experience.
6. Docker configuration.

A.5 Resources

I will require access to Virtual Machine infrastructure in order to do build my topology and compare virtual machine performance. I will also require access to a machine that has the same hardware and system resources as those shared by the virtual machines, as I will need to perform my containerisation tests on the same hardware in order to effectively control that variable.

A.5.1 Hardware

I shouldn't require any hardware as long as I can get remote access to the virtual machine infrastructure in a way that allows me to use the same resources for containerisation (as mentioned above).

As contingency, if I cant access this infrastructure, I will still need some way of creating and managing virtualisation. This might be possible on my own hardware, though I'd rather use university infrastructure as this should be more reliable and accessible.

A.5.2 Software

I will need to install Docker, and will need access to benchmarking suites, but these are free and/or open-source.

A.6 Structure and Contents of the Report

Below I will set out the chapters that I will most likely have in my final report.

A.6.1 Report Structure

Abstract & Introduction Here I will set out the background for my project and the reasoning behind the work I will be doing. The Abstract will also summarise the work that has been done along with my findings, subsequent recommendations and conclusions.

What is virtualisation & containerisation An introduction to virtualisation and its current context and usage in today's world of computing. An introduction to containerisation, what it is, how it works, and how it differs to virtualisation. This will form part of my literature review.

Current uses of virtualisation An explanation of modern uses of virtual machines. I will emphasis virtual servers here, and the reasons they are used.

Current uses of containerisation. An explanation of modern usage of containers. I will emphasise application development and other, non-main-sever applications here.

Application of containerisation to server infrastructure Introduce the idea of applying containers to the area I study, with the use-case of server infrastructure. Explain what the implications of this could be. Set out a detailed definition of the problem I want to solve/test.

Brief explanation of how this could be designed, with reference to other similar studies.

Network Infrastructure Design Here I will lay out the infrastructure that I want to implement for my testing and my reasoning for this, along with references (see proposed work).

How to test and measure the system Justify the creation or use of a particular set of benchmarks that I will use against the system for testing. I will also explain here how I will measure and evaluate the two systems.

Building the Virtual System Here I will build the virtual system to work with the topology laid out in the infrastructure chapter. I will make sure to explain how to do it. I will also provide evidence of the system working.

Building the Container System Similar to the previous chapter, I will build the container based system following the topology laid out in the infrastructure design section. I will explain how this is done. (ie, using Docker). (see proposed work). As per the previous chapter, I will provide evidence of this working in practice.

Testing and gathering data Here I will create a test plan for running benchmarks across the parallel servers in order to compare the like for like running of both systems.

Analysis & Comparison I will briefly explain how the data shown was required as an introduction to what the data shows. This is where I will explain in detail the actual results, and what they mean, I won't infer here.

Evaluation of the system A technical evaluation of the system I created. I will highlight and strengths, and failings in my ability to meet the requirements of my implementation. I will also ensure to evidence this accordingly.

Evaluation of the project process Here I will evaluate my own learning, and explain what I would do differently about my approach, should I go on to do the same, or a similar, project in the future. My objectives should be assessed as per the marking scheme laid out in the project handbook.

Conclusion and Recommendations I will first create a balanced conclusion, considering the main points mentioned in my evaluation sections. This should cover the direction the work went in, presenting key findings as a way of getting my opinion on the project across. I will present my opinions on the usage of containers in practical setting (ie, whether I think they should be used instead of virtualisation; where they would/could be used) Furthermore, as my work should have practical implications on enterprise and organisations that use virtualisation, I will aim to create recommendations of further work in order to push this area further in the future.

A.6.2 List of Appendices

My appendices will first include my TOR, Ethics form and Risk Assessment.

Some of the network design documentation may be included if too large for the main body, along with the configurations from the various servers.

I should also include the full and direct results of my benchmarks, as these will almost certainly be too large to display in my main text. (I will instead pull the direct numbers from the benchmarks and reference to the related Appendix).

A.7 Marking Scheme

The marking scheme sets out what criteria we are going to use for the project.

Project Type General Computing.

Project Report *Analysis Chapters*

- What is virtualisation & containerisation.
- Current uses of virtualisation.
- Current uses of containerisation.
- Application of containerisation to server infrastructure.

Synthesis Chapters

- Network Infrastructure Design.
- How to test and measure the system.
- Building the Virtual System.
- Testing and gathering data.
- Analysis & Comparison.

Evaluation Chapters

- Evaluation of the system.
- Evaluation of the project process.
- Conclusion and Recommendations.

Product The 'product' includes the Network Infrastructure design, the configurations across both systems, the virtual system and the container system themselves.

Fitness for Purpose

- Meet requirements identified.
- Application to real world infrastructure.
- Whether the headless servers can communicate effectively on both systems.
- Whether both systems achieved the same goals and outcomes. They should have the same logical topology.

Build Quality

- Requirements specification and analysis.
- Quality of the infrastructure design.
- Configuration quality.
- Benchmarking plan quality.

A.8 Project Plan

A.9 Ethics Form

If you scan the Ethics form on one of the multifunction printers, you can get a pdf copy. This can then be included with the LaTeX command

Assuming of course you have saved the form as `ethics.pdf`

A.10 Risk Assessment Form

Likewise you can scan and include the Risk Assessment Form