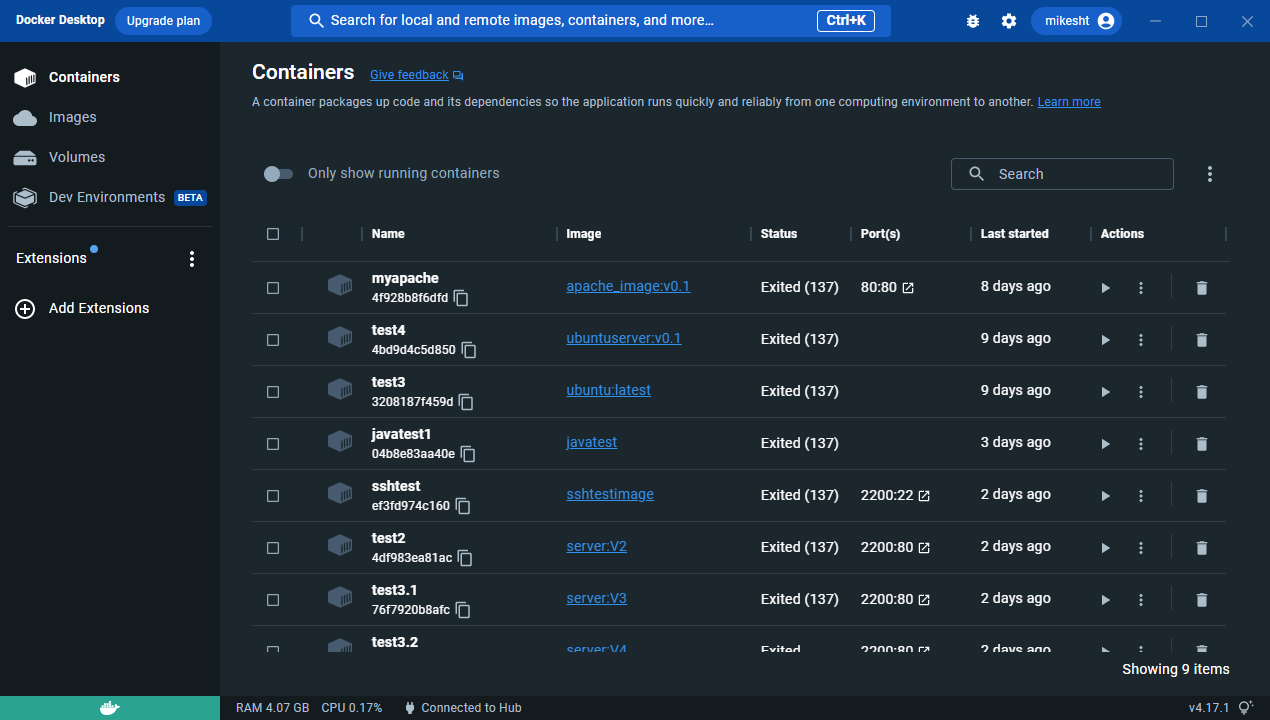
# DevOps Capstone Documentation

## Docker Desktop

Docker is a platform designed to help developers, build, share and run modern applications. It does this by allowing the users to create applications inside isolated containers. A container packages code and all its dependencies into a single unit, thus letting an application run quickly and reliably from one computing environment to another. This makes such applications easily portable between machines and solves the “it works on my machine” problem.

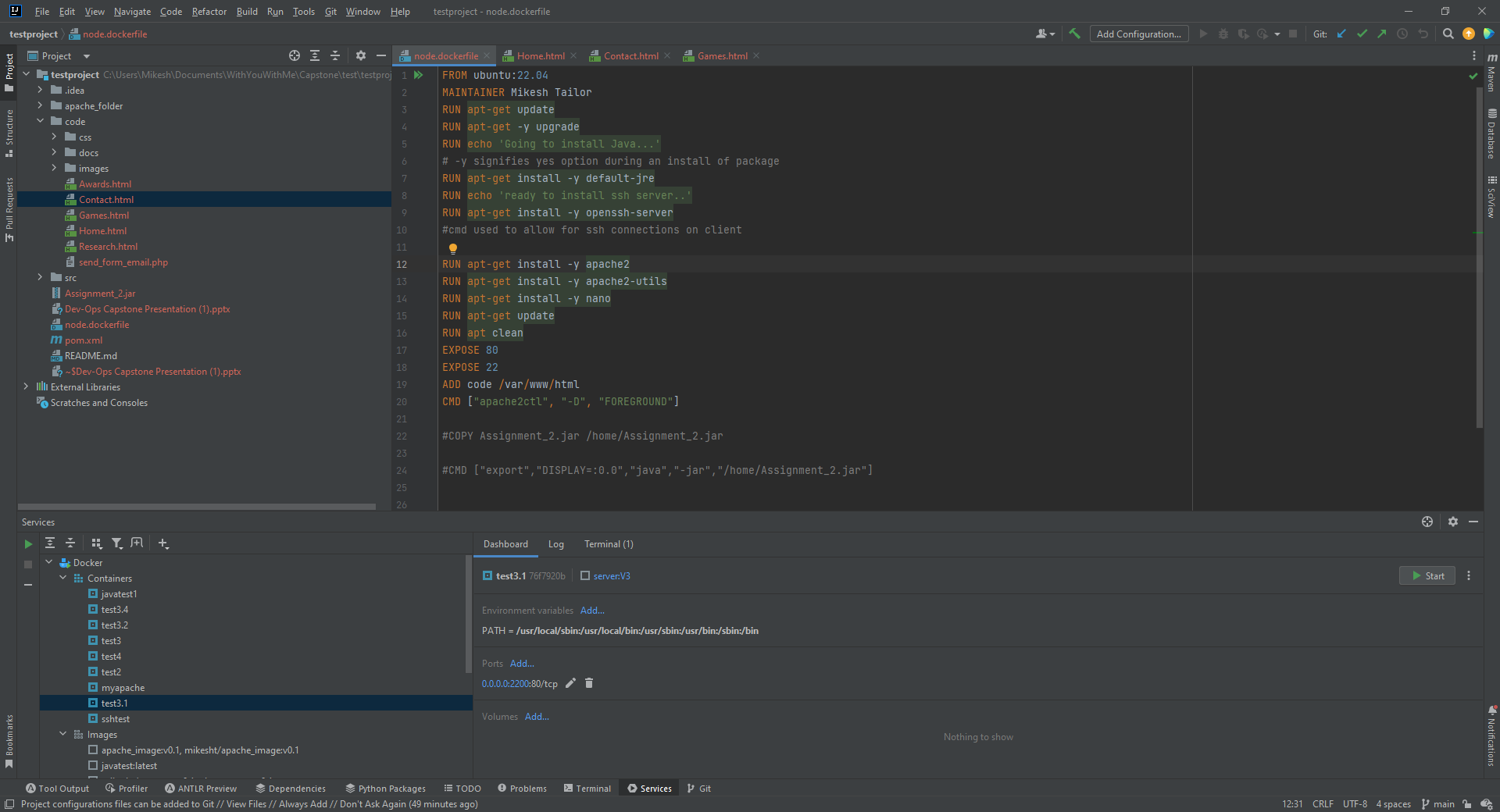


## The Dockerfile

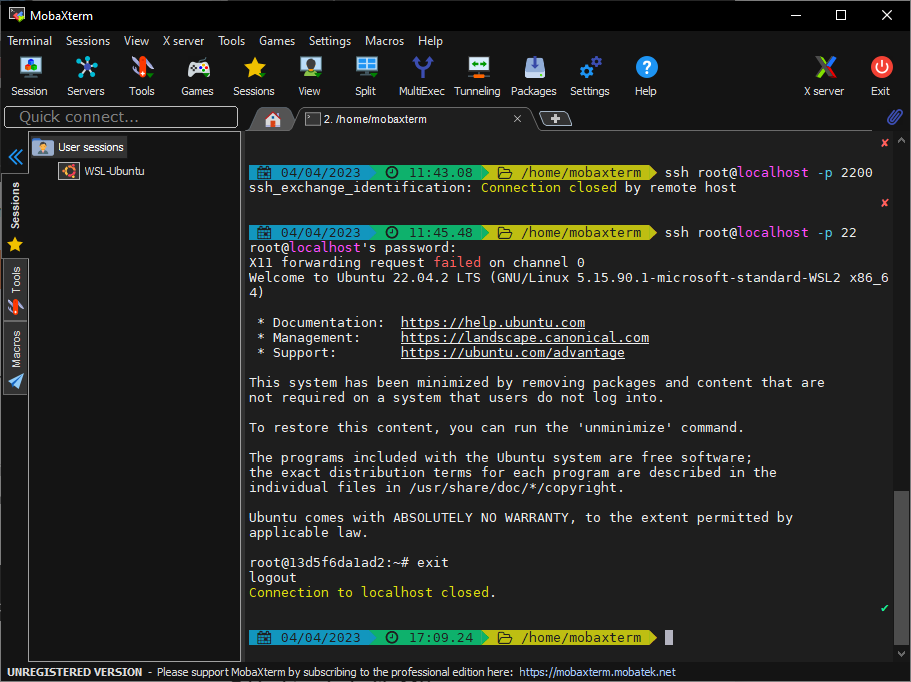
In the appendix at the end of the document is the custom Dockerfile that was created for the Project. It is essentially the Instruction manual for docker when building an image. Within the dockerfile there is a “FROM” command that is used to tell docker what operating system to use as a base, the “RUN” command is then used to install any required packages. This image can then be used to run a container with any packages preinstalled that are required. For this

## IntelliJ

**IntelliJ IDEA** is an integrated development environment (IDE) written in Java for developing computer software written in several languages. It was used in this project as a tool to write the dockerfile and source code. The program also has a docker plugin which would allow the developer to edit, create and push docker images into a registry. It also has the ability to start, stop and interact with the containers from the terminal.



## Mobaxterm

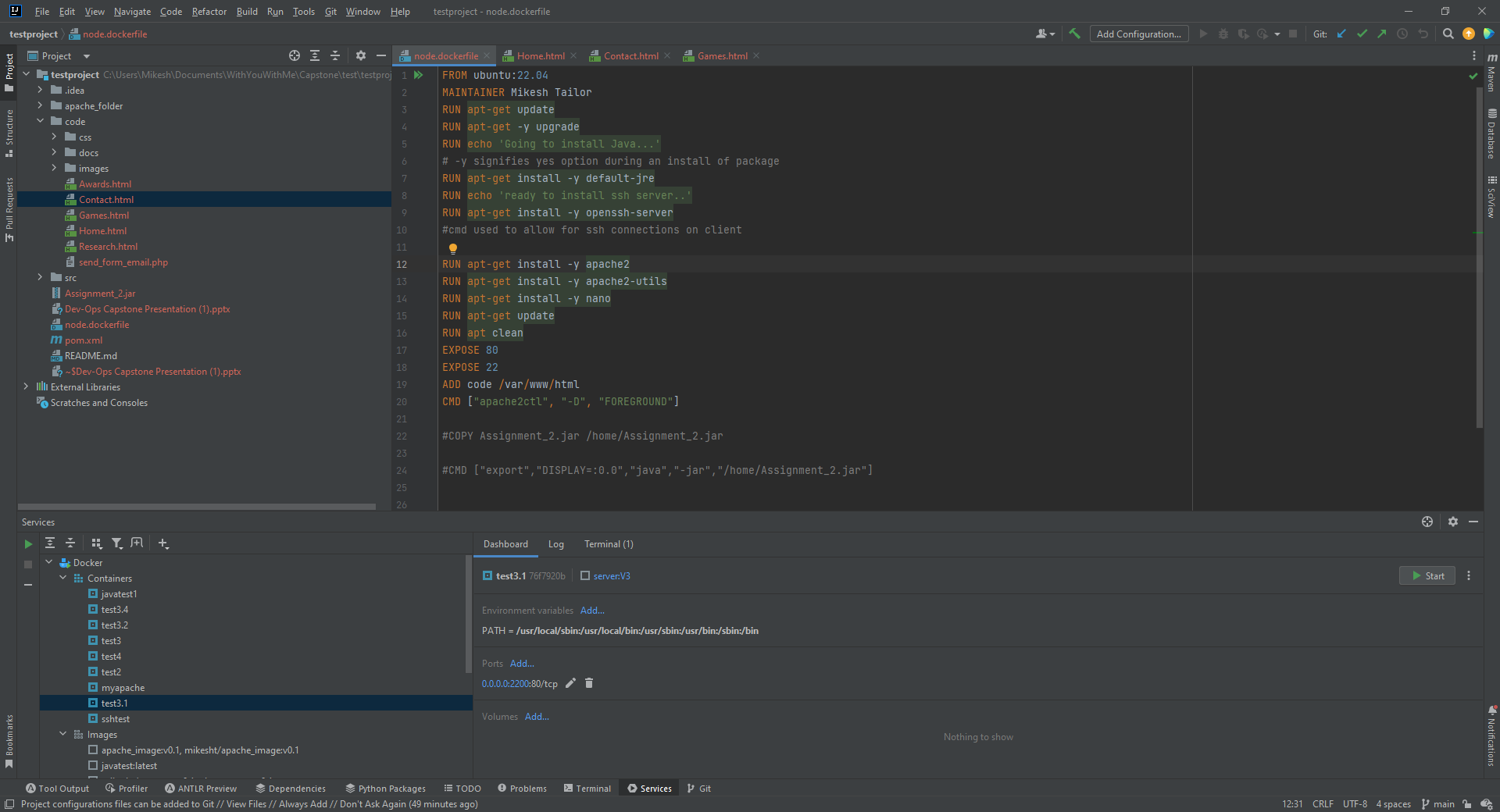
MobaXterm provides all the important **remote network tools** (SSH, X11, RDP, VNC, FTP, MOSH, ...) and **Unix commands** (bash, ls, cat, sed, grep, awk, rsync, ...) to Windows desktop.

It was used in this project as a SSH client in order to connect into the server running on the container.

## Complete Build Guide

In this section there is a detailed step by step guide for the build of the project.

Step 1:



Build a Dockerfile with any packages that you know are essential to speed up any future developments.

Here you can see that the base image is an Ubuntu Server and the packages that are to be preinstalled, including Apache2 webserver.

You can also see the ‘ADD’ command which copies the contents of my source code into the container location provided.

Lastly, the ‘CMD’ command at the end of the file specifies that the web server should run as soon as the container runs.

Below you can see the command you should enter into command line in order to build a docker image from this Dockerfile where ‘-t’ is a tag property of the image and ‘-f’ specifies the Dockerfile name.

docker build -t server:V5 -f node.dockerfile .

Step 2:

Once the docker image has built, the next step is to run a container from that image, which can be done with the following command. Where ‘—name’ is the name of the container and ‘-p’ is the port that the host and container will be listening on, finally ‘-d’ is used so that the container runs detached from the terminal.

docker run --name test3.4 -p 8080:80 -p 22:22 -d server:V5

Once a container is running you then have an isolated environment with your preinstalled packages ready.

Step 3:

As the container is running detached from the terminal a command will have to be entered in order to gain access. This can be seen below, where ‘-it’ stands for interactive.

docker exec -it test3.4 bash

ideally with the SSH package already installed from the docker file the developer should be able to access the container through SSH however, there are some changes that need to be made in the container itself before this will be possible. See below.

Passwd root (set a password as the user will not have one currently set)

nano /etc/ssh/sshd\_config

then you will have to work your way down the configuration file until you get to a “Authentication” heading and type the following below:

PermitRootLogin Yes

After saving the changes to the file a final command can be used to start the SSH service inside the container.

service ssh start

That’s it, SSH will now be available through port 22 of the container.

## References

Docker: <https://www.docker.com/>

IntelliJ: <https://www.jetbrains.com/idea/>

Windows Command Prompt: <https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands>

MobaXterm: <https://mobaxterm.mobatek.net/>

Pluralsight: <https://app.pluralsight.com/course-player?clipId=b7a263b7-2d4a-44f2-a5fa-409541dd0c0e>

## Links

Github: <https://github.com/MikeshTailor/Jerry>

LinkedIn: <https://www.linkedin.com/in/mikesh-tailor-8081ba217/>

## Appendix 1