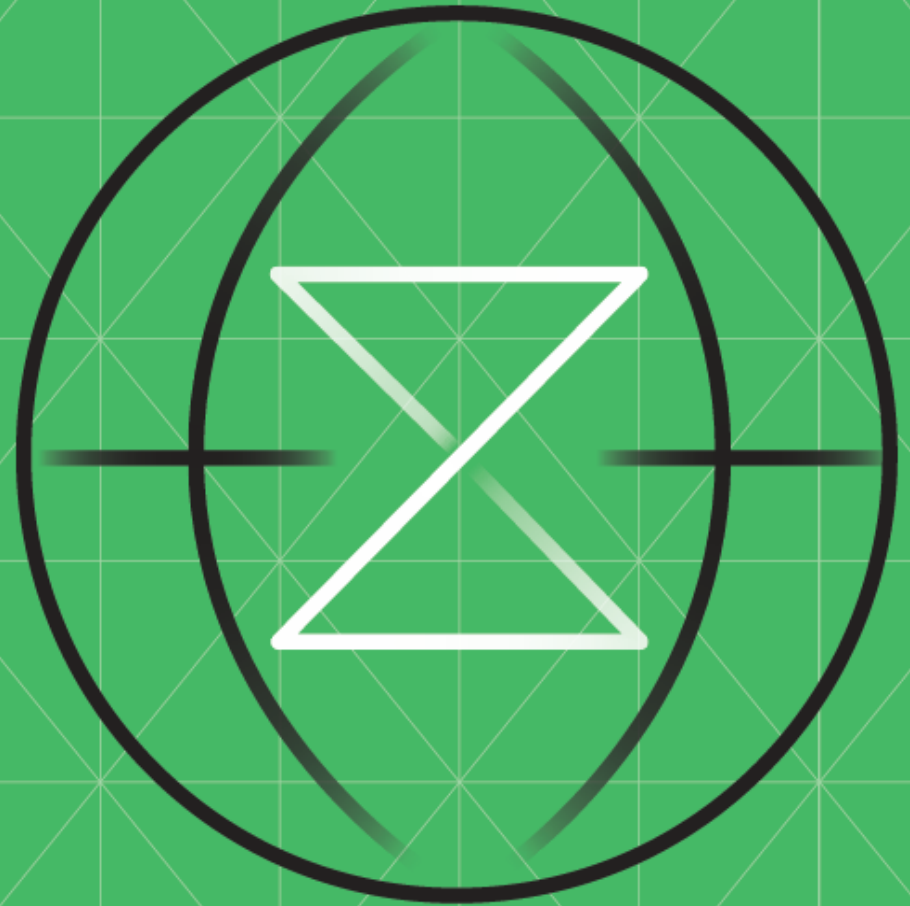


Docker

Can you Contain yourself ??

- [ELEVATE YOUR MAINFRAME EXPERIENCE WITH DOCKER](#)
- [1 INSTALL DOCKER IN VSCODE](#)
- [2 INSTALL DOCKER DESKTOP](#)
- [3 FOLLOW THE PROMPTS](#)
- [4 TAKE THE TOUR \(OPTIONAL\)](#)
- [5 LOAD UP AND IMAGE](#)
- [6 UP AND RUNNING, JUST LIKE THAT!](#)
- [7 REMOTE DEVELOPMENT EXTENSION](#)
- [8 LAUNCH REMOTE EXPLORER](#)
- [9 OPEN ROOT](#)



ELEVATE YOUR MAINFRAME EXPERIENCE WITH DOCKER

A growing trend in the IT world is the adoption of containers - packaging self-contained applications, and all the necessary supporting code libraries and modules, into lightweight “runtimes” that can be executed as tasks within the host operating system.

The container approach originally started in linux but support to run containers has been added to Windows and MacOS. The most popular (and free!) tool for creating, managing and running containers for development and testing is Docker.

You will be using Docker to learn the basics of containers; this will give you enough experience to understand what the enterprise-grade container platforms, like Kubernetes and Red Hat Openshift, are doing “under the covers”.

The Challenge

By now, you know how the deep-down mechanics of z/OS work. You know how to allocate a data set from Zowe, through VSCode, through USS and ZOAU; you’re at the point where you just want things to happen as quickly and smoothly as possible, and you are about to experiment with Docker, an open-source platform to create, deploy and manage virtualized application containers.

Before You Begin

You’ll need some essential tools in place -

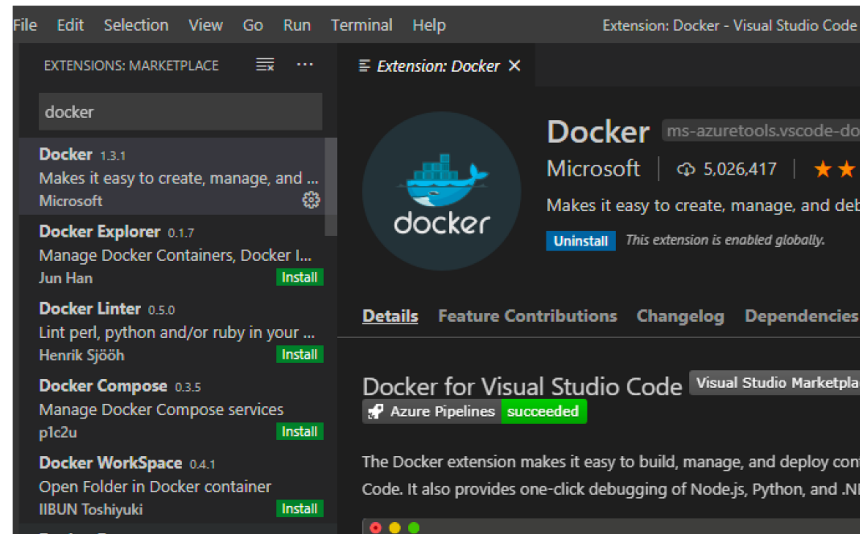
- Docker
- a couple of VS Code extensions.

This will require some installation and setup of software on your workstation and in VSCode - it will all be worth it!

Investment

Steps	Duration
9	60 minutes

1 INSTALL DOCKER IN VSCODE



In VSCode, select “Extensions” and search for Docker. There are lots of Docker-related extensions, but you can start with just the plain old “Docker”.

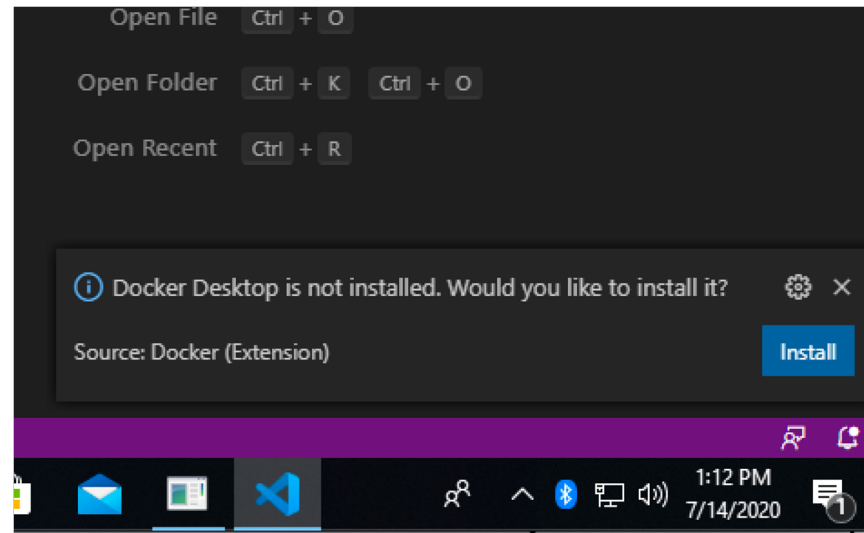
This will give VSCode the ability to work with Docker images and containers. The install should be pretty quick; but there will be a bit more for you to do ...

Go into the new Docker icon that was created on the left side in VSCode and you may notice that all of the boxes have little yellow “warning” triangles in them, because Docker Desktop isn’t installed yet. (Unless, of course, you already *do* have Docker Desktop installed!)

If you see the warnings, you can fix that in the next step.

Docker | 230901-1117

2 INSTALL DOCKER DESKTOP

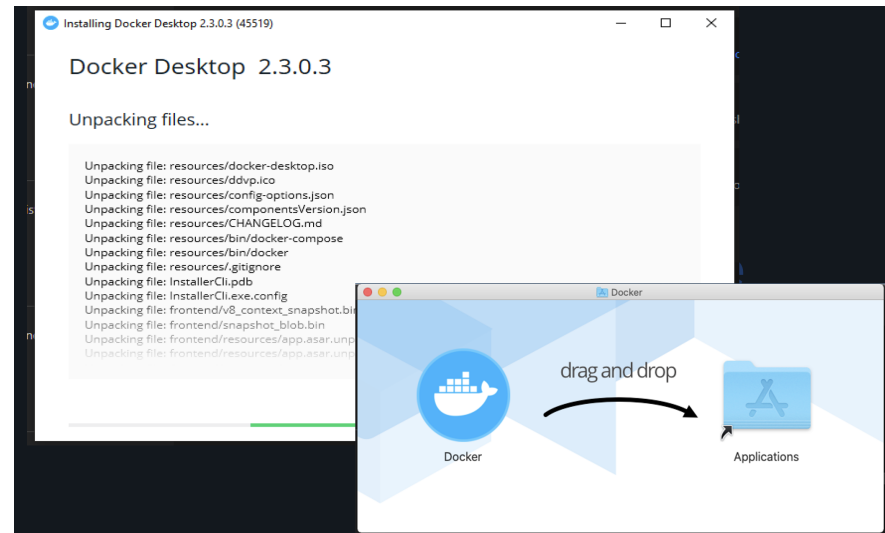


If this message pops up, click "Install".

If you missed it, or still have a warning that Docker Desktop isn't installed/running, then head over to [Docker Desktop website](#) and start the download for your workstation/laptop.

This is a more complicated install, and might take up to 30 minutes depending on your computer speed and internet connection.

3 FOLLOW THE PROMPTS



You are going to have to use your best judgment here for a while.

Installation should be fairly straightforward.

- On Windows, you'll run an installer and reboot.
- On MacOS, you'll drag the Docker icon into Applications.

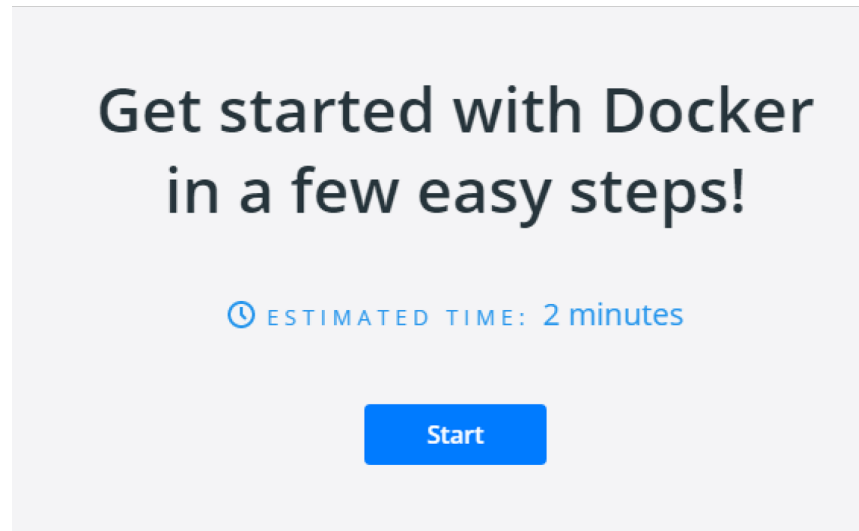
A reboot may be required. The installation process will tell you if this is needed.

After the installation finishes, launch Docker Desktop and follow the prompts.

4 TAKE THE TOUR (OPTIONAL)

Do you want to play around with a sample container and try out a few commands before getting started?

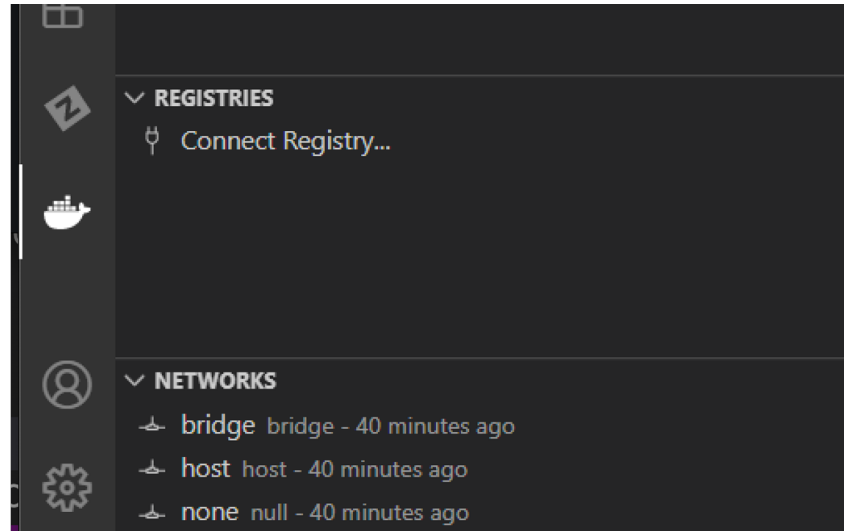
There's a built-in demo that might help you get some of the Docker concepts.



Already done, or feeling like you're pretty solid on Docker? Then move along to the next step, but make sure to keep Docker Desktop running.

5 LOAD UP AND IMAGE

If you relaunch VSCode, click on the Docker icon on the left side (it's that iconic whale with the containers on its back).



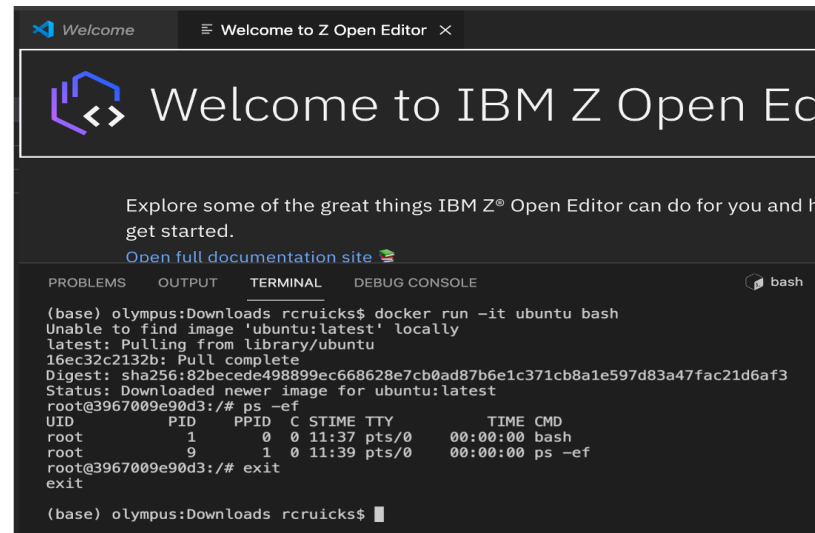
You should see no warning triangles, but instead something similar to the screenshot above.

Go into your Terminal and enter this command to run a docker image as a container, and start an interactive shell inside the running container. Think of this like a tiny linux system running on your laptop.

```
docker run -it ubuntu bash
```


6 UP AND RUNNING, JUST LIKE THAT!

Since you probably don't already have this "ubuntu" image, Docker Desktop will need to download all of the pieces for you.



```

Welcome
Welcome to Z Open Editor x

Welcome to IBM Z Open Editor

Explore some of the great things IBM Z® Open Editor can do for you and how to get started.
Open full documentation site

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE bash
(base) olympus:Downloads rcrucks$ docker run -it ubuntu bash
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
16ec32c2132b: Pull complete
Digest: sha256:82becede498899ec668628e7cb0ad87b6e1c371cb8a1e597d83a47fac21d6af3
Status: Downloaded newer image for ubuntu:latest
root@3967009e90d3:/# ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1        0  0  11:37 pts/0    00:00:00 bash
root           9        1  0  11:39 pts/0    00:00:00 ps -ef
root@3967009e90d3:/# exit
exit
(base) olympus:Downloads rcrucks$
```

When the download has completed, you will notice that you now have:

1. A container
2. An image
3. The terminal prompt is a window into the docker container

This can be a little mind-bending at first. The container is acting like a virtual machine running on your computer, but it's really just the parts you will need to run Ansible.

This will be made a little more clear by installing one more VSCode extension, in the next step.

Docker | 230901-1117

I CAN'T GET DOCKER RUNNING. DO I NEED IT?

Docker is a technology which allows solutions that run in a standardized environment (basically, anything that runs Docker) to be picked up and made to run anywhere else.

This means that every little file, library, piece of code, even full software packages and operating systems can be packaged into a Docker Container and made available for distribution. This set of challenges are made a little bit easier by making the Ansible environment pre-packaged in a Docker Container.

If you can't get it installed, don't want to, or want to keep your options open, you can install and use Docker on a Linux image like the one you created in the Linux-based challenges.

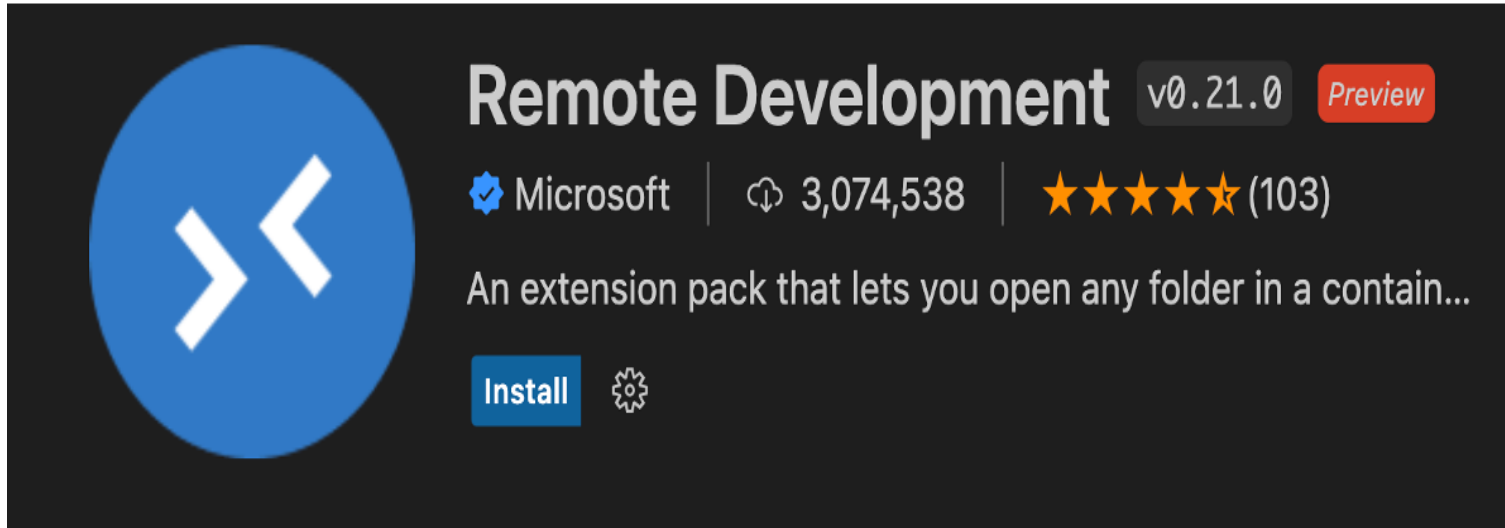
Or bypass using Docker altogether, and [install Ansible](#) directly on your laptop operating systems.

For these challenges, however, we will be using the VSCode+Docker Desktop method, as it should work for most people, and provide you with useful experience with Docker.

Docker | 230901-1117

7 REMOTE DEVELOPMENT EXTENSION

In the VSCode Extensions screen, search for and install the “Remote Development” extension suite.



Then open the VSCode Preferences/Settings menu, search for **DOCKER_HOST**, set/update to use the local docker socket, and save:

`"unix:///private/var/run/docker.sock"` (MacOS)

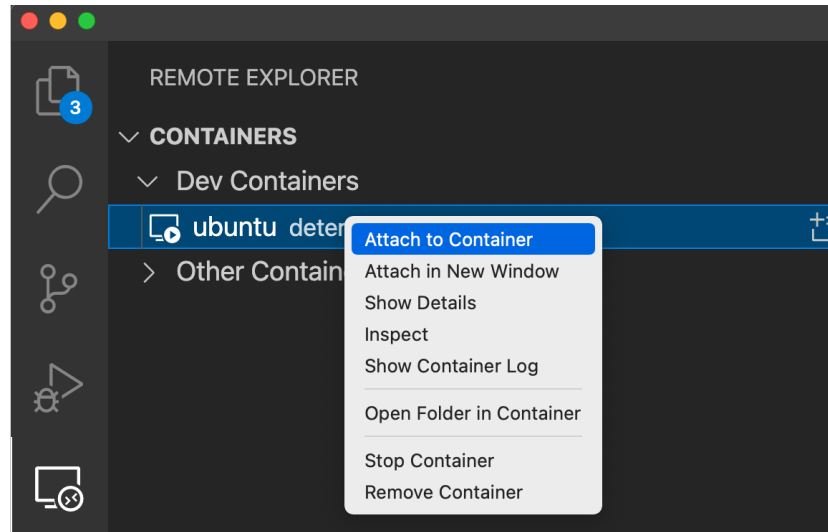
`"unix:///var/run/docker.sock"` (Windows, Linux)

Once updated, you should restart VSCode to apply all the updates.

Docker | 230901-1117

8 LAUNCH REMOTE EXPLORER

When you have restarted VSCode, find the new icon on the left side for “Remote Explorer”, and use that to find the Ubuntu container which you just pulled down started running. (You may probably need to redo the command from Step 5 since when you restarted VSCode, the terminal session will have ended)



Right-click on the Ubuntu container, and select “Attach to Container”.

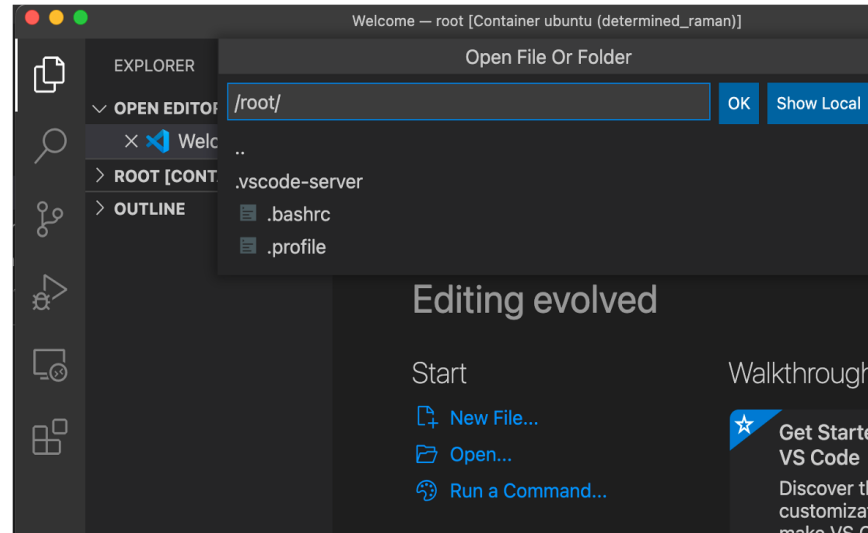
This will launch a new VSCode window, which gives you a view into what’s happening inside this container image.

This is the container view, *not* your computer’s regular filesystem.

Notice the green box at the bottom of the window showing you what the view is, and also notice that the Zowe icon is not there.

9 OPEN ROOT

Click on the Explorer button, then click "Open ...", and enter `"/root/"` as the folder you want to open.



You should then see a listing of files like the one in the screenshot above.

Now you are ready to tackle all sorts of container-based activities – you can use this setup to get hands-on experience with Ansible.

As usual, you can mark this challenge complete by submitting the **CHKADOCK** job in **ZXP.PUBLIC.JCL**

WHY ALL THE PREP WORK?

Getting a program to work once is good. Getting it to work hundreds, if not thousands of times after that, the exact same way, takes careful orchestration.

In the past, that meant that a person, or a team of people, had to manually configure a whole group of systems, and carefully set them up so that a program could run on them. They would manage every step of the process, and doubling the amount of programs running meant everyone had to work twice as hard.

That's the old way!

Docker containers, and orchestration that happens in solutions like Ansible and Kubernetes, means that scaling up in capability doesn't mean adding complexity at every step. It's an important revolution in Enterprise IT that not only produces more capable solutions, but frees up staff to work on other tasks - like coming up with the next greatest thing, instead of setting up yet another system.

Nice job - let's recap	Next up ...
<p>You have set it all up, so you might as well use it. The Ansible challenges depend on the use of docker containers to build and execute “playbooks” - you should be all set to tackle these now.</p>	<p>Launch the ANS1 Ansible introduction challenge</p>