Containerization with docker

Foreword

In previous sessions which we looked at programming languages, I mentioned one of the best practices in developing any package is to create an environment for it, such as using `virtualenv` in Python. This makes sure the packages you need for your project will not interfer with other projects you are working on.

Containerization shares the same idea, but take it further. In this course, we have been sharing code with others through `GitHub`. However, when it comes to deploying your code, i.e. making your application available to the public as a service, we probably don't want to share the codebase either out of security concerns or because we don't want to burden the user with the need to install all the dependencies. This is where containerization comes in. Containerizing your application means you package your application and all its dependencies into a single image, which can be easily run on another machine. Furthermore, the user won't need to modify their environment to run your application, and they don't have to worry about using your application would interfere with other applications they are running.

In this lab, we will learn how to containerize an application using Docker, a popular containerization tool. Docker has an excellent 101 tutorial, so here we are just going to give the essential commands to get the tutorial running.

Instruction for getting the tutorial running

Once you have Docker installed on your machine, you can run the following command to get the tutorial running:

```
docker run -d -p 80:80 docker/getting-started
```

Once you have this command going, you can navigate to `http://localhost` in your browser to see the tutorial running. Then we will go through the tutorial from there.