# Docker Compose
# IN720 Virtualisation

## Introduction

In this lab we will create a service using a three different containers managed with Docker Compose. To begin, create a project directory named `lab_4.2`

## 1 Create a Flask application container

Make a directory `flaskapp` to serve as a build context for the Python/Flask application image. You can just reuse the context you used in lab 4.1, with a two changes:

1. Replace the Flask application code from last time with the files `app.py` and `requirements.txt` you will find in the `week04/lab-4.2-resources` directory.
2. Change the `ENTRYPOINT` value to `python app.py`.

## 2 Create an nginx proxy container

In this case you can simply reuse the nginx container from lab 4.1 with no changes. It would, however, be a good idea to rename the image to something else, like "`proxy`".

## 3 Create a compose file

Finally., create the `docker-compose.yml` file. In it, you will need directives to

1. Build and run the `flaskapp` container, using the `app` network.
2. Build and run the `proxy` container, using the app network and mapping its port 80 to a port on the host.
3. Run a `redis` container, also attached to the app network. It's not necessary to set up a special image for this. You can just use the standard `redis` image from Dockerhub.
4. Create the app network.

Once this is done, you can build any images you need with the `docker-compose build` command. You will also need to run this any time you make changes in your build contexts to update your images. Start the whole thing running with the `docker-compose up` command, and shut it down when you're done with `docker-compose down`.