# Cloud Computing Applications and Services
Containers

2023

## Docker

Docker is the most widely-known container technology. Containers are intended to be a loosely isolated (when compared to Virtual Machine's isolation levels) and lightweight environment to run applications and services. Along this exercise guide, we will go through the steps of configuring and deploying the Swap application on top of the Docker platform. For this, we will be deploying two Docker containers. One will serve the database management system and the other will serve the remaining Swap services.

Docker documentation is available at: `https://docs.docker.com`.

## Tasks

### VM Setup

1. Create 1 VM (*server1*). You may use one of the Vagrantfiles from the previous guides.
2. Install Docker (`https://docs.docker.com/engine/install/ubuntu/`).
3. Add your user to the *Docker* group to avoid using *sudo* all the time (`https://docs.docker.com/engine/install/linux-postinstall/`).
4. Run the sample application (`https://docs.docker.com/get-started/02_our_app/`).
5. Access the application from your browser (`http://<SERVER1_IP>:3000`).
6. Understand the commands `docker {image, ps, exec, stop, start, kill, rm, logs}`. E.g., understand and explore the command: `docker exec -ti <CONTAINER_ID> /bin/sh`

### Docker Network

Similar to the network adapter created and configured for the virtual machines in previous guides, let us set up a docker network for connecting our containers. In Docker, creating such a network will also allow us to use hostnames to refer to each container, thus simplifying our setup.

1. Create a network using the following command: `docker network create <NETWORK_NAME>`
2. Use the `docker network list` and `docker network inspect` commands to check details about the new network.

### Database Container

Let us start by setting up a MySQL database engine in its own Docker container. For this step, we will use an already prepared image.

1. Pull the official MySQL Docker image (`mysql:latest`).
   Explore the `docker image pull <IMAGE_NAME>` command to pull the image.
   Explore the image's documentation at `https://hub.docker.com/_/mysql`.

2. Create a Docker container with a MySQL installation:

```
docker run --name <DB_CONTAINER_NAME> --net <NETWORK_NAME> -p 3306:3306 -d \
  -e MYSQL_USER=<DB_USERNAME> -e MYSQL_PASSWORD=<DB_PASSWORD> \
  -e MYSQL_DATABASE=<DB_DATABASE> -e MYSQL_ALLOW_EMPTY_PASSWORD=true \
  mysql:latest
```

3. Check that MySQL service is running and well configured:
   - check MySQL logs: `docker logs <DB_CONTAINER_NAME>`
   - connect to the container with `docker exec -ti <DB_CONTAINER_NAME> /bin/sh` and verify if the Swap database was successfully created: `mysql -u<DB_USERNAME> \ -p<DB_PASSWORD> -e 'show databases';`

**Swap container**

Now, let us set up the actual Swap app. Please refer to Guide 1 and your previous Swap installation experience for the next tasks.

1. Explore the Dockerfile reference documentation at `https://docs.docker.com/engine/reference/builder/`
2. Explore the Dockerfile instructions (e.g., `FROM`, `RUN`, `WORKDIR`, `COPY`, `EXPOSE`, `CMD`) and set up a Dockerfile to build a Docker image for the Swap app.

   Such app should use the MySQL installation configured earlier as its database engine. Please note that you should configure the environment accordingly (*.env* configuration file).

   Use Ubuntu 20.04 as the base image for the Dockerfile (`FROM` instruction).

   **Note:** The tasks for migrating and seeding the database should be ignored in this phase.
3. Use the command `docker build` to build the Swap Docker image.

   ```
   docker build . -t <SWAP_IMAGE_NAME>
   ```
4. Use the command `docker run` to run the Swap container.

   ```
   docker run --net <NETWORK_NAME> -p 8000:8000 \
     --name <SWAP_CONTAINER_NAME> <SWAP_IMAGE_NAME>
   ```

   **Note:** This is only an example, you should add any other additional container flags that your image may require.
5. Test your setup by accessing the app through a web browser (`http://<SERVER1_IP>:8000`).
6. Ensure that the variables `<DB_HOST>`, `<DB_DATABASE>`, `<DB_USERNAME>`, and `<DB_PASSWORD>` defined at Swap's *.env* configuration file are dynamically set when running the Swap container and are not hard-coded on the Dockerfile (e.g., explore the Dockerfile `ENV` instruction).

# Extras

1. Modify the Swap Docker image to include the database migration and seed tasks. Note that the database seed task should run optionally (i.e., only when the user asks).
2. Setup Redis in a separate container and add such component to the Swap deployment.
3. Build an Ansible playbook for automating the installation and configuration of Docker.

# Learning outcomes

Hands-on experience with software container technology (Docker). Deployment of an application on top of a container platform.