# CS352 - Assignment 3: SelfieLessActs on Containers

**Learnings**

Learnt how to setup and use docker containers. Could understand the benefit of using docker images. Publishing and pulling images to a commonly accessible network is also something that came in quite handy when trying to clone our local machine containers onto the instance. Had to add a few network rules to ensure proper working on the instance.

We originally planned to create a third container containing our database service. However, upon more thought we realised that it would beat the point of isolating these APIs into separate microservices. Down the line, if we needed to add more such containers for replication, it would have shown up as a problem. Thus, we split the database and made it such that there is no foreign key constraint. Further, to ensure that only users who exist are to upload new acts, we made sure to call an API from within the uploadAct API on the container for acts.

**Any other observations/challenges/comments**

Found it particularly annoying that docker containers once created cannot have changes to their network configuration in terms of their ports and mappings. Hence to add/change the port related information we resorted to creating a new container with the required changes.

Found it time consuming to install the most naive packages like MariaDB on an image as light as alpine. Would have preferred to use an ubuntu image.

We were originally using docker’s backend network service to establish connection between the two containers. We learnt that this is not detected by the script being used to test our assignment. We switched to using the public IP to communicate, and are aware that this works only on the AWS instances as AWS has its instances on DNS,allowing its own IP address to be found by the container which has no regard for the other container, despite being on the same physical system.