**Exercise 3 – Flight Scanner Microservices**

**Objective**

Practice container-based development, packaging and shipment.

**Overview**

Modify the application from exercise 2 so that each process will run on separate containers and communicate with each other via **named pipe**.

**Detailed description**

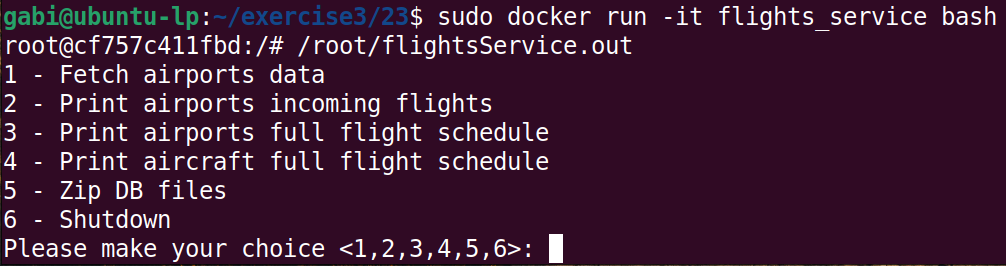
Create 2 docker images:

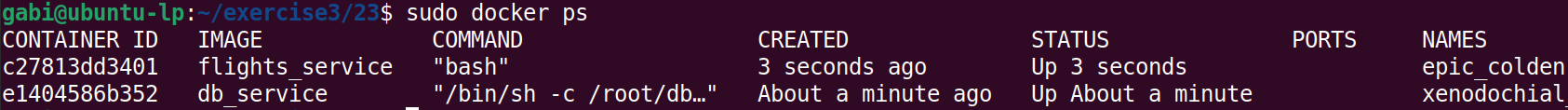
1. dbService - Shall contain the “child process” application and create a named pipe in a predefined mounted path(e.g. /tmp/flights\_pipe) and wait for opcodes from flightService app.
2. FlightsService – Shall contain the “main process” application that will process the user commands, send opcode to dbService and print the received data.

Notes:

1. dbService container will always run before main
2. dbService container should be able to work in the background, meaning in a non-interactive mode(without -it option)
3. Shutdown should result in closure of both containers
4. Modify “Update DB” option to “Fetch airports data” from the internet, this option should receive a list of airports IOCAs separated by space, keep using your script just modify the code to pass the airports to it
5. Make sure each of the containers can be restarted and the whole application will resume running properly
6. Handle errors!!

**Examples**





**Submission**

Upload your 2 images to docker hub, tutorial: <https://docs.docker.com/docker-hub/>

In your MAMA submission files:

1. A README file with your IDs and additional remarks if needed.
2. A bash script to initialize the whole setup, pull and run both containers.