**Dockerizing other components**   
  
Based on the instructions provided in the previous four exercises implement the deployment of the following components:

* **employee-service**
  + Create a Dockerfile similar to the one done for authorisation-service
  + Modify port as 8092 and mysql connection URL in application.properties
  + Modify docker-compose.yml with inclusion of employee-service linking to payroll-mysql
  + Run docker compose and test a REST API service in employee-service to verify if the service works end to end
* **eureka-discovery-service**
  + Create a Dockerfile similar to the one done for authorisation-service
  + Modify port as 8093 and mysql connection URL in application.properties
  + Run docker compose and test if the registry service gets hosted
* **zuul-gateway-service**
  + Create a Dockerfile similar to the one done for authorisation-service
  + Modify port as 8094
  + Modify docker-compose.yml with inclusion of zuul-gateway-service linking to authentication-service and employee-service
  + Run docker compose and test a REST API service in authentication-service and employee-service to verify if the service works end to end
* **angular-learn**
  + Modify the REST API URL in the angular application refering to zuul-gateway-service
  + Execute the angular build which creates the dist folder
  + Create a Dockerfile that copies dist folder content to nginx container and starts the server (Reference: <https://hub.docker.com/_/nginx>)
  + Include docker-compose.yml with a new service named angular-service that runs the Dockerfile and links with zuul-gateway-service
  + Run docoker compose and test the angular application to see if it works end to end