Angular Identity Management

Déploiement avec Docker

Nous allons déployer l'application Angular avec Docker sous nginx.

Étapes de déploiement

Pour faire le déploiement avec Docker, nous allons :

1. Créer un répertoire docker
2. Générer l'application
3. Écrire le fichier Dockerfile
4. Écrire le fichier de configuration de nginx
5. Démarrer le container

Création d'un répertoire docker

Nous allons créer un répertoire pour centraliser tous les fichiers nécessaires à la mise en place de l'application avec Docker.
L'arborescence sera la suivante :

```
image                    ← appelé ainsi car nous allons créer une image docker et non un fichier compose
--- target
    --- dist              ← contient le résultat du build de l'application
    nginx.conf           ← fichier de configuration du serveur web nginx
    Dockerfile           ← image pour docker
    makeWSDockerImage.sh      ← script de déploiement de l'image dans docker
```

Générer l'application

Avec Angular, React et bien d'autres framework JS, pour construire une application il suffit de la "builder".
Placez-vous dans le répertoire racine et exécutez la commande suivante : `yarn build`
La génération va créer un répertoire dist contenant les fichiers javascript nécessaire au déploiement sur un serveur web.

Vous pouvez "builder" votre application en mode:

- développement : `yarn build` ou `ng build`
- production : `ng build --prod`

N'oubliez pas que les variables d'environnements sont différentes:

- build "normal" : fichier "src/environments/environments.ts"
- build production : fichier "src/environments/environments.prod.ts"

Rappel : Lors d'un build production, le fichier "environments.ts" est remplacé (écrasé) par le fichier "environments.prod.ts".

Copiez le contenu du répertoire "dist" du projet Angular dans le répertoire dist de docker.

Ecrire le fichier Dockerfile

Le fichier Dockerfile va permettre de

- copier les fichiers de build de l'application dans le répertoire html de nginx
- copier le fichier de configuration de nginx pour lancer l'application
- lancer nginx

Le Dockerfile sera le suivant :

```
FROM nginx:alpine
COPY target/nginx.conf /etc/nginx/conf.d/default.conf
COPY target/dist/users-management /usr/share/nginx/html


CMD ["nginx", "-g", "daemon off;"]
```

Ecrire le fichier de configuration de nginx

Le fichier de configuration de nginx sera le suivant :

```
server {
  listen 80;

# Type your domain name below
  server_name usersmgnt.kilroy.lan;

# Always serve index.html for any request
  location / {
    # Set path
    root /usr/share/nginx/html;
    index  index.html;

    #try_files $uri /index.html;
    try_files $uri $uri/ /index.html?$args;
  }

# Do not cache sw.js, required for offline-first updates.
  location /sw.js {
      add_header Cache-Control "no-cache";
      proxy_cache_bypass $http_pragma;
      proxy_cache_revalidate on;
      expires off;
      access_log off;
  }
}
```

Démarrer le container

Pour démarrer le container il faut

- Créer l'image : `docker build -t ...`
- Lancer le container : `docker run -p ...`

Pour ce faire, nous allon sécrire le fichier batch (linux) suivant nommé "`makeWSDockerImage.sh`":

`#!/bin/bash`

```
echo "Arret des containers"
# Liste des containers en cours nommé identity_mgmt
LISTIDS=$(docker ps -aqf "name=identity_mgmt")
# Si la liste n'est pas vide ...
if [ ! -z $LISTIDS ]
then
  # ... on arrête le container
  docker container stop $(docker ps -aqf "name=identity_mgmt")
fi

echo "Suppression des containers"
# Suppression du container si il existe
LISTIDS=$(docker ps -aqf "name=identity_mgmt")
if [ ! -z $LISTIDS ]
then
  docker container rm $(docker ps -aqf "name=identity_mgmt")
fi

echo "Suppression de l'image"
# Suppression de l'image
LISTIDS=$(docker images -q kilroy/identity_mgmt)
if [ ! -z $LISTIDS ]
then
  docker rmi $(docker images -q kilroy/identity_mgmt)
fi

echo "Création de l'image"
# Créer l'image Docker
docker build -t identity_mgmt .

echo "Démarrage du container"
docker run -p 4201:80 --detach --restart always --name identity_mgmt identity_mgmt
```

Vous devez rendre le fichier exécutable : `chmod 755 makeWSDockerImage.sh`
Puis l'exécuter : `./makeWSDockerImage.sh`

Liens

https://angular.io/start

https://blog.angular.io/

https://blog.angular-university.io/

https://guide-angular.wishtack.io/

https://openclassrooms.com/fr/courses/4668271-developpez-des-applications-web-avec-angular

https://www.typescriptlang.org/docs/home.html

https://blog.soat.fr/

Parent Child Two way binding
https://medium.com/@preethi.s/angular-custom-two-way-data-binding-3e618309d6c7

Organisation par module
Mise en place de la sécurité
https://angular.io/guide/route

Login:
https://loiane.com/2017/08/angular-hide-navbar-login-page/

Dialog:
https://blog.angular-university.io/angular-material-dialog/

Angular Material
https://material.angular.io/

https://medium.com/@ismapro/first-steps-with-angular-7-with-angular-cli-and-angular-material-d69f55d8ac51

https://www.positronx.io/create-angular-material-8-custom-theme/

https://akveo.github.io/ngx-admin/

https://auth0.com/blog/creating-beautiful-apps-with-angular-material/