Fichier AILENEI\_ROBERT Examen AdminInfra

# Explications document

Pour chaque question, mettez ici si le résultat attendu est atteint ou pas (OK/KO) c’est-à-dire si cela est fonctionnel sur votre VM.

Si cela est demandé dans l’énoncé, ajoutez des explications (commandes, scripts commentés, fichiers de configuration commentés, capture d’écran, …)

Si le résultat est KO, ajoutez quelques explications, un état des lieux (où vous en êtes dans la résolution de la question). N’hésitez pas à expliquer/commenter ce qui pourrait sembler obscur à l’évaluateur. L’objectif de ce fichier est de faire comprendre aux évaluateurs ce que vous avez fait et surtout ce que vous avez tenté de faire quand cela ne fonctionne pas.

# Question 1

Résultat attendu : OK

Commande pour afficher la page HTML : awslocal s3 cp s3://exam/index.html -

Explications éventuelles : installation de aws et awslocal.

creation du fichier main.tf qui utilise localstack et aws et creation du fichier index.html qui contient Ex admininfra2025. Le bucket est cree dans le fichier main.tf. donc pas besoin de le recreer. Utilisation de terraform init et terraform apply pour lancer le processus. On verifie si tout marche avec awslocal s3 ls s3://exam pour voir le contenu du bucket. On voit bien le fichier index.html. on regarde a l’interieur avec : awslocal s3 cp s3://exam/index.html -

# Question 2

Faire tourner le site localement - Résultat attendu : OK

Instructions pour faire tourner le site localement :

Il faut installer node et npm afin de lancer le site localement. sudo apt install nodejs npm -y

On verifie l installation : node -v

npm -v

on installe les dependances avec npm install

on demarre le server avec npm start

on teste le site avec curl http://localhost:3000

Explications éventuelles :

Reverse proxy avec SSL (sans page des exoplanètes avec DB) - Résultat attendu : OK

Explications éventuelles :

On cree un virtual host ou on precise le certificat ssl autosigné et le reverse proxy. Nous faisons la liaison dans le fichier hosts. Si je lance le server avec npm start sur le port 3000 et que dans un autre terminal je fais lynx <https://exopl> je suis bin redirigé sur la page des exoplanetes.

Site fonctionnel totalement (avec la page des exoplanètes DB) - Résultat attendu : OK

Explications :

Installation de postgresQL avec sudo apt install postgresql postgresql-contrib -y

On s’assure que le service postgreSQL est bien installe et qu il fonctionne avec :

sudo systemctl start postgresql

sudo systemctl enable postgresql

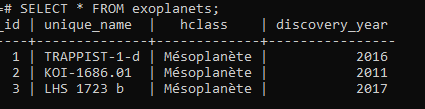
nous nous connectons à l’interface utilisateur de psql avec sudo -u postgres sql (donc le user est postgres)

nous voyons un entry de commande, nous allons créer la database qu’on a besoin pour notre site.

Je cree la database exoplanets avec la commande CREATE DATABASE exoplanets; \c exoplanets. Le \c me connecte a la db exoplanets avec mon user postgres. Je quitte le psql avec \q pour import le fichier d’initialisation de db dans psql. Je le fais avec cette commande : sudo -u postgres psql -d exoplanets -f /var/www/exoplanets/initdb/createdb.sql

Je rerentre dans le psql avec mon user postgres pour verifier si les donnees ont bien été importees.

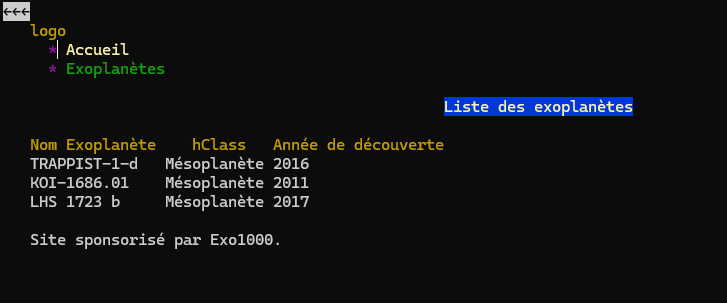
Je fais cela : \c exoplanets pour me connecter a la db exoplanets

Et SELECT \* FROM exoplanets me montre bien 3 planetes : 

Le pghost devrait etre localhost vu que le serveur et la db roulent sur la meme machine, donc j ai mis pghost : localhost. Maintenant il y a un probleme de mot de passe :

je pense que le superuser n’a pas le meme mot de passe que ce qu on a dans le config.js du site. Du coup on va devoir changer son mdp en ipl. On ouvre le terminal et on change le mdp avec cette commande : ALTER USER postgres PASSWORD 'ipl';

Maintenant ca marche, on voit bien les planetes de la db sur la page exoplanetes :



# Question 3

Résultat attendu : OK

Explications :

1 ere demande : nous allons créer un alias dans ~/.bashrc comme suit :

alias bn="sudo docker compose up" , ou bn sera la commande qui va remplacer docker compose up.

Il faut recharger le fichier : source ~/.bashrc

J’ai utilisé un alias pour la premiere demande car ca permet de lier une commande a un alias plus court , comme il l a demandé.

2 eme demande : dans le folder dezipe, je cree un docker-compose.yml

Donc root@lab2LWUZV:/home/ipl/exam/qst3/siteBonnesNouvelles# nano docker-compose.yml

Voici ce qu’il y a dedans :

services:

web:

image: php:7.4-apache

ports:

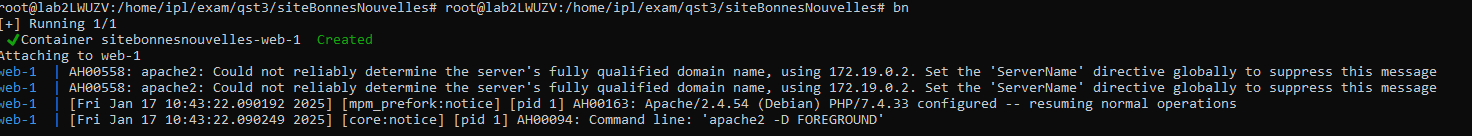
- "8080:80"

volumes:

- .:/var/www/html

J’ai choisi un montage de volume car Le montage de volume permet un développement rapide sans reconstruction d’image.

Maintenant , dans le meme folder que le docker-compose.yml, on ecrit bn dans la console :



Le container est bien cree. Nous avons besoin au total de 3 terminaux : 1 pour lancer le container, un pour faire un lynx dessus, et 1 pour modifier le texte afin de voir si le CTRL – R marche bien.

Dans le premier terminal, on se connecte a la vm, on cree son alias dans ~/.bashrc comme precise plus haut et on va dans le folder ou se trouve le docker-compose.yml . on ecrit le alias (bn) et le container est lancé. Ne pas oublier de faire source ~/.bashrc !

Dans un deuxieme terminal, nous faisons un lynx pour acceder au site :

Lynx http://localhost:8080

On peut voir le header Un site de bonnes nouvelles :



Dans le troisieme terminal, il nous faut les permissions de modification de ces documents, le plus simple est de se mettre en root avec la commande su.

Une fois en mode root, nous allons modifier le header pour verifier que on peut voir les changements juste apres un reload (CTRL-R).

On va faire un nano du header.php et remplacer Bonnes par Mauvaises.

Apres un CTRL-R sur le deuxieme terminal, nous voyons que le changement a bien eu lieu :

