Analyse du dev.eneotransportation.com et déploiement de dev.lewootract.com

- Connection au backend d'eneo:
  - Login: root
  - Ip address: 161.97.155.250
  - Password: b357U78QMqA9u
- Connection a la platforme en ligne:
  - Login: africacustom
  - Password: africacustom

Pour déployer la solution eneotransportation, Une fois rendu dans le serveur, on clone le dépôt bitbucket (ce dépôt sur bitbucket c'est le repository eneotransportation de lewootract), on installe tous les dépendance de chaque fichier, on complète éventuellement les variables d'environnement (on créer les fichier **.env** correspondant). Dans le backend, on lance tous les micro-services **(pm2 start pm2-apps-dev.json)**, et pour chaque micro-services on installe les dépendance, on complète les **.env**.

Après avoir lancée ces micro-services, si certain renvoyé le statut errored, ou tout simplement se sont arrêtés, il faut se renseigner sur la cause de cet arrêt en consultant les logs de ce dernier (**pm2 logs nom du micro-service**)

Note importante : le package pm2 est installé à partir de **npm** (npm install -g pm2) ou **yarn** (yarn add -g pm2) pour la gestion des micro services.

Un micro service est spécifique pour une tache bien précise. Les micros services sont des applications complètement indépendantes qui quand on les ressemble il construise une application entière.

root@vmi593450:~/apps/eneotransportation/Backend/services/api-stat-service# ls (liste le répertoire)

root@vmi593450:~/apps/eneotransportation/Backend/services/api-stat-service# ls -a (liste le répertoire avec les fichier cachés)

- Avec le pm2 on est capable de consulter le log d'un micro service, comment il est entrain de communiquer (commande : **pm2 logs main-service** permet de consulter les logs du micro-service main-service)

Les logs sont consultés par ce qu'il y a des micros-services qui peuvent s'arrêter de fonctionner

- La commande **pm2** peut prendre plusieurs types de paramètres :
  - **pm2 logs main-service** -> vérifié le logs des mains services, généralement en cas de disfonctionnement
  - **pm2 stop main-service** -> arrêté le main-service
  - **pm2 status main-service** -> ceci affichera le statut du main service actuellement c'est stopped
  - **pm2 list** pour tout lister les micro-service
  - Noter que main-service peut être représenté par n'importe quel autre service.
  - **grep –ar port */.env** : recherche l'élément **port** dans tous les fichiers ayant l'extension **.env**

La définition de la structure des micros services de base se trouvent dans le fichier **pm2-apps-dev.json** du répertoire Backend de eneo tracking lorsqu'on ouvre ce fichier lorsque le nbres d'instances est en précision alors on à ce nbre d'instance au cas contraire ou par défaut c'est une instance. Cat **pm2-apps-dev.json** nous donne plus d'information sur ce contenu. C'est donc ces instances qui s'affichent lorsqu'on lance la commande **pm2 status**

Redémarrer un micro service :

Exp : opération intervient généralement lorsque le micro service a été modifié

root@vmi605670:~/apps/eneotransportation/Backend/services# pm2 nom du micro_service(exp :**restart api-stat-service**

Déploiement de dev.lewootrack.com

Il est question de reconstituer totalement le serveur à partir de zéro

Network : ping dev.lewootrack.com
User: root
Ip: 161.97.175.75
Pwd: S9J8KU6r3bLdep7

Acces bitbuker:

Login: as.contributor@africasystems.com
Password: As@contributor

Comment démarrer, supprimer, redémarrer une image et un container, comment télécharger une image

Dans la suite de cet exercice, nous répondrons à ces questions.

Parlons de graphql (apprendre du nodejs)

En effet nous avons commencé à configurer notre serveur pour commencer on a besoin de **bitvise** (télécharger en ligne) et des accès pour accéder à la machine distante ou au serveur

Une fois cela fait on crée le dossier **apps**, on se positionne dessus et on clone le dépôt distant dessus, l'adresse de clonage ressemble à ceci : git clone https://AsContributor@bitbucket.org/lewootrack/eneotransportation.git et un mot de passe : ATBBCUk5S8dD3Td2uP5qvJwqmXA9BBFB0C25

Installation du node_module :

Une fois cela fait on accède à eneotransportation et on accède au dossier admin-backend et backend puis

npm install pour installer les dépendances du node module

Installation par yarn dans des appli front-end

Une fois cela fait on rentre et on accède à admin-frontend puis **yarn** pour installer des dépendances, mais il peut y avoir erreur sur la version du node module. Une fois termine on rentre et on accède au dossier **frontend** puis yarn pour installer également des dépendances. (si la version de nodejs n'est pas bonne, on peut installer le gestionnaire de version de nodejs (nvm) puis telecharger la bonne version en utilisant **nvm Install nouvelle_version**

*Nb : les installations se font dans des dossiers qui ont un fichier **package.json***

building appli :

Puis il faut builder l'application commande de building : **sh docker-build.sh**, ce fichier est dispo dans le dossier courant d'eneotransportation. En effet le dossier sur lequel on se place doit contenir ce fichier docker-build.sh, on build l'application frontend pour compresser en un fichier index… mais il n'est pas nécessaire de builder une application backend.

Node_micro_services :

Apres le building on se positionne sur dossier contenant un fichier **.json** et on installe le node module via npm Install exp : **root@vmi605670:~/apps/eneotransportation/Backend/services/api-stat-service# npm install (pour installer ces dépendances très facilement).**

Lancer tous les micros services :

Sur le dossier Backend : root@vmi605670:~/apps/eneotransportation/Backend# **pm2 start** pm2-apps-dev.json. Après cette étape, les services ne sont pas encore tous online, il faut les démarrer pour ce fait on le fait en block avec la commande précédente.

Une fois cette étape fait on vérifie l'état des micro service : **pm2 list**

Micro services problématiques :

Pour les micro services qui ne sont pas en ligne, on vérifie leurs **logs** exp : **pm2 logs microservice_name** pour essayer de capter l'erreur et le résoudre. A cette étape nous n'avons pas encore fait une activation de serveur donc les problèmes qui sont liés à l'activation des serveurs peuvent être déposés de cote pour plutard mais dans la plupart des temps, cette erreur provient de l'absence du fichier **.env** car des variables d'environnement s'y trouvent.

Dans notre cas nous avons identifié le **.env** du micro service correspondant dans la plateforme eneotransportation et nous avons copie ce contenu dans la plateforme test que nous sommes en train de construire. Pour créer un fichier dans bitwise et sur le répertoire du micro service correspondant on tape la commande suivante : **nano nom_fichier(exp .env)** ***>entrer> coller contenu > ctrl o pour sauver a un nom de fichier(.env) > entrer>ctrl + x(pour fermer l'environnement.***

Les micros services les plus problématiques

- main service - live-pipeline service - afcm service

Le fichier frondend doit également être builder, ceci rend compact l'ensemble de nos fichiers

PASSONS A DOCKER

L'usage de docker est fait pour vitaliser des serveurs, c'est un gestionnaire de container, il est basé sur des images serveur, il faut télécharger et installer une image pour la virtualiser, il est utile pour tester des appli, il permet de faire des tests en local afin de déployer si tout fonctionne correctement.

- installation de docker :
  sudo apt-get install docker-ce docker-ce-cli containerd.io
  - lancer docker : **sudo systemctl start docker**
  - **docker --v:** avoir la version de docker en installe
  - **docker ps** : pour lister tous les containers
  - **docker ps --a :** pour visualiser même le container éteint
  - **docker rm --f id_container** : supprime un container d'id id_container
  - **docker images** : montre l'image construite
  - **docker images --a** : montre des images intermédiaires
  - **docker containers ls** : liste un ensemble de de container
  - **docker stop id_container** : pour stopper un container
  - **docker start id_container** : démarrer le containeur en spécification
  - **docker rm id_container** : supprimer un container
- création des containers et images :
  - mysql /*installation et dockerisation de l'image mysql
    __ install mysql client: **apt install mysql-client-core-5.7**
    __create an image: **docker run --name admin-mysql -p 3306:3306 -v /var/lib/mysql:/var/lib/mysql -e "MYSQL_ROOT_PASSWORD=afric@dmin "-d mysql**

    __ executer le container: **docker exec -it mon-mysql mysql -u root –p**

__pwd: **afric@dmin**

__ pour visualizer les tables: **SHOW DATABASES;**

__pour visualiser les table : **USE nom-bd;**

SHOW TABLES;

NB : les commandes se saisissent sur une seule ligne

- postgres /*installation et dockerisation de l'image postgres
  __install postgres :
  sudo apt install postgresql
  __ image postgres:

  docker run --name timescaledb -p 5432:5432 -v /var/lib/postgresql/data:/var/postgresql/data -e "POSTGRES_PASSWORD=afric@tracking2021" timescale/timescaledb:latest-pg9.6

  __to connect: **psql -U postgres -h '0.0.0.0'**
  __pwd **: afric@tracking2021**
  __ installation du client postgres
  apt install postgresql-client-common
  sudo apt-get install postgresql-client

  __lister les bases de donnees : **\l**

  __se connecter a une base de donnee : **\c** db_name

  __voir des tables : **\dt**

- redis /*installation et dockerisation de l'image redis
  __ sudo apt install redis-tools

  __ docker run -p 6379:6379 --name redis -d redis:latest --requirepass "afric@sys@123"

- mino /*utilisé pour la création des **BROKERS**

  __docker run -d -p 9000:9000 --name minio -e "MINIO_ACCESS_KEY=minio_storage" -e "MINIO_SECRET_KEY=AfricaMinioStorage@2021" -v /minio/data:/data -v /minio/config:/root/.minio minio/minio server --address ":9000" /data

CLOUD STORAGE LAYER :

Il permet de regrouper des informations déjà trop anciennes dans la bd pour éviter de la saturer

- Deploy the archival storage server on docker:
  __ Docker run -d -p 9000:9000 --name minio -e " MINIO_ACCESS_KEY=minio_storage " -e " MINIO_SECRET_KEY=AfricaMinioStorage@2021 " -v /minio/data:/data -v /minio/config:/root/.minio minio/minio server --address ":9000 " /data
- Deploying deux brokers:
  __ docker run -p 1883:1883 -e "DOCKER_VERNEMQ_ACCEPT_EULA=yes" -e "DOCKER_VERNEMQ_ALLOW_ANONYMOUS=on" --name mqtt-broker-1 -d vernemq/vernemq

__ docker run -p 1884:1884 -e "DOCKER_VERNEMQ_ACCEPT_EULA=yes" -e "DOCKER_VERNEMQ_ALLOW_ANONYMOUS=on" --name mqtt-broker-2 -d vernemq/vernem

BROKER

Sur ce dernier on a la possibilité de souscrire a des topics, et quand une entité publique des informations sur ces topics tous ceux ayant souscrit reçoivent ces informations, ils nous permettent de communiquer avec des équipements ou traceurs en effet les traceurs et nos équipements public sur des topics et la plateforme en commun et lorsque les équipements communiquent des infos sur le topic, nous recevons également ces informations puis on les parses avec l'algorithme et on les enregistre dans notre base de données. Dans notre cas nous créons deux brokers : **mqtt-broker-1**, **mqtt broker-2** (voir : commandes de créations ci-dessus).

Alors on peut tester dans un navigateur : **161.97.175.75 : 9000** puis on entre les paramètres de connexions qu'on a spécifié dans les paramètres de connexions.

Lorsque je lance la commande **yarn** à la fin on a une erreur comme quoi node incompatible, on ajoute a cette commande l'option --ignore-scripts

Pour copier le contenu du .env : on ouvre enoetransportation déjà fonctionnel puis on sélectionne le contenu à copier et ça se fait automatiquement puis on revient dans l'interface de colle puis click droit et ça colle auto puis crtl+O pr save et ctrl+x pr quitter. Surtout n'oublie jamais le fichier **.env** du dossier frontend

CLOUD STORAGE LAYER

Stockage dans le cloud(buket) utiliser pour soulager des bases de données, il contient des données anciens stoker les bases de données devenues trop grand sur le. Il faut créer un conteneur pour lui ici on a mino : la commande est la suivante :

NGINX

Permet la configuration du serveur local et c'est un service, command de redémarrage : service nginx restart

Télécharger : sudo apt-get install nginx puis configure cd /etc/nginx/site-available

On peut accéder à son répertoire : /etc/nginx/ puis cat dev.lewootrack.com