Fase 4 - Activitat 7.3: Eina docker compose i fitxer docker-compose.yml

0- Identificació del grup i activitat:

Curs: ASIX2
Projecte: PJ9 DevOps i Cloud Computing
Fase: 4
Activitat: 7.3
Grup/Individual: Individual
Membres/Alumne:

1- Introducció i objectius de l'activitat 7.3

a) Comprendre el propòsit del fitxer **.dockerignore**
b) Comprendre el funcionament i propòsit de l'ordre **docker compose**
c) Comprendre el propòsit del fitxer **docker-compose.yml**
d) Tenir una breu introducció al format **YAML**
e) Realitzar una activitat avaluativa

2- LLegeix-me: host, .dockerignore, docker compose, docker-compose.yml i format YAML en poques paraules

a) Quan tractem el tema dels contenidors amb Docker, el **host** és l'equip informàtic sobre el qual es creen i s'executen els contenidors. En el nostre cas és normalment una màquina virtual creada amb Vagrant però també pot ser un equip físic.

a) Dins d'un fitxer **.dockerignore** escriurem la llista de fitxers i directories que NO volem que s'afegeixin a una imatge de contenidor quan fem un build. Això permet ajustar la mida de les imatges de contenidors, crear-les més ràpidament i augmentar la seguretat. Per més informació llegeix aquest link.

b) L'eina **docker compose** permet llegir la configuració que es troba dins d'un arxiu de nom **docker-compose.yml** i a continuació amb una simple instrucció i de manera ràpida i automàtica:

- Crear un servei que permet crear i posar en marxa un o més contenidors que treballen conjuntament per poder desplegar una aplicació. Si cal, també permet crear primer les imatges dels contenidors necessaris per fer funcionar el servei.
- Crear volums per compartir dades entre contenidors i la màquina host dels contenidors, o també entre els contenidors amb uns altres.
- Crear una xarxa de contenidors perquè els contenidors es puguin comunicar entre ells d'una manera ràpida i eficient que si ho haguessin de fer a través de la màquina host dels contenidors.

Amb l'eina **docker compose** es molt fàcil desplegar des d'una aplicació sencilla que només té un contenidor fins a una aplicació complexa amb múltiples contenidors sense que calgui res més que el fitxer **docker-compose.yml** i executar una instrucció molt senzilla.

En comparació a fer la feina manualment, **docker compose** és més ràpid, més fàcil d'utilitzar, més confiable perquè redueix la probabilitat de cometre un error i més compartible. Per més informació llegeix aquest link.

c) Allò que diferència un tècnic qualificat d'un simple usuari de l'eina **docker compose** és la seva capacitat per crear un fitxer **docker-compose.yml**. Aquest fitxer utilitza el nou format **YAML** per definir els contenidors, imatges, volums, xarxes i dependències entre contenidors que són necessàries per desplegar una aplicació. El format **YAML** com més va més s'utilitza per crear arxius de configuració de programes i serveis, definir estructures de dades, etc..., i hauria de ser conegut per qualsevol administrador o desenvolupador. Per més informació llegeix aquest link.

d) Respecte del format **YAML**:

- Els fitxer que utilitzen YAML són de texte, llegible pels humans i tenen l'extensió **.yml** o **.yaml**.
- Igual que python les sagnies són obligatories per indicar que un paràmetre està dins d'un altre.
- Les sagnies han de ser sempre iguals i només es poder utilitzar espais en blanc (mai tabuladors).
- Utilitza mapejat per associar un paràmetre amb el seu valor. Per exemple → **image: ipizza:1.0**
- Per veure altres característiques, llegeix aquest link.

2- Modificació de l'aplicació. Crea un dipòsit local i remot per pujar l'aplicació i el seu entorn de treball amb contenidors Docker i Vagrant.

a) Dins de la **màquina física**, accedeix a la carpeta **pj9 → f4 → a7 → pj9f4a7.1 → codis → pizzeria → app** i a continuació, modifica el fitxer **pizza.php** de la següent manera:

- Darrera de la línia **2**, afegeix: **$IVA=21;**
- La nova linia **5** serà: **$pvp_public=$pvp_public*$IVA;**
- La nova linia **6** serà: **echo "El preu de la pizza bàsica és $pvp_public € (IVA inclòs)
";**
- Esborra la nova línia **7** amb el codi **exit(0);**
- Esborra la nova línia **9** amb el codi **$IVA=21;**

b) Dins de la **màquina física**, i dins de la carpeta **pj9 → f4 → a7 → pj9f4a7.1**:

- Fes un **add** del continguts del directori **pj9f4a7.1** (exceptuant el indicats a **.gitignore**). Executa la següent ordre: **git add** *
- Fes un **commit** del continguts del directori **pj9f4a7.1** dins del dipòsit local. El missatge del commit serà: *"Versió 1.2 del projecte pizzeria"*.

c) Comprova que s'han pujat al dipòsit local:

- La nova versió de **pizza.php**.

Per fer la comprovació, executa: **git show --pretty="" --name-only**

d) Sincronitza el dipòsit local amb el dipòsit remot dins de Github.

e) Refresca la web del dipòsit per assegurar-te que s'ha pujat la nova versió del projecte.

3- Creant un fitxer .dockerignore per ignorar fitxers i directoris durant un build

a) Dins de la **màquina física**, i dins de la carpeta **pj9 → f4 → a7 → pj9f4a7.1 → codis → pizzeria**:

- Crea un fitxer de nom **.dockerignore**
- Obre el fitxer **.dockerignore**.
- Fes que el seu contingut sigui:

```
.dockerignore
docker-compose.yml
```

b) Torna a **pj9 → f4 → a7 → pj9f4a7.1** i fes un **add** del continguts del directori **pj9f4a7.1** inclós el nou fitxer **.dockerignore** executant: **git add .**

c) Fes un nou **commit** del continguts del directori **pj9f4a7.1**. El missatge del commit serà: *"Versió 1.2 del nou projecte pizzeria amb .dockerignore"*.

d) Comprova que s'ha pujat **.dockerignore** al dipòsit local executant: **git show --pretty="" --name-only**

e) Puja la nova versió del projecte al dipòsit remot i refresca la web del dipòsit remot per assegurar-te que tots les noves versions i el nous fitxers del projecte s'han pujat.

4- Creant un fitxer docker-compose.yml mínim per iniciar l'aplicació

a) Dins de la **màquina física**, i dins de la carpeta **pj9 → f4 → a8 → pj9f4a7.1 → codis → pizzeria** i crea un fitxer de nom **docker-compose.yml**.

b) Fes que el contingut de **docker-compose.yml** sigui:

```
services:
    webphp_pizzeria:
        image: ipizza:1.2
        build: .
        container_name: cpizza1.2
        ports:
            - "80:80"
#
# NOMS DE SERVEIS, IMATGES I CONTENIDORS.
# 1- webphp_pizzeria → Etiqueta identificadora del servei
# 2- ipizza:1.0 -> Nom i versió de la imatge que crearem
# 3- cpizza1.2 -> Nom del contenidor que crearem
#
# PORTS EXPOSATS
# 1- ports:
#     - "80:80"  # Ports del contenidor que exposarem a la màquina virtual
#
# OBJECTIUS DEL FITXER docker-compose.yml:
# 1- Posa en marxa el servei webphp_pizzeria necessari per fer anar l'aplicació. Aquest
# servei utilitza un contenidor. Quan es faci un docker compose up es posarà en marxa
# aquest servei i els contenidor que necessita.
# 2- Crea una imatge de contenidor de nom ipizza:1.2
# 3- Crea un contenidor de nom cpizza1.2 que utilitzant la imatge ipizza:1.2
# 4- El contenidor exposa el seu port 80 al port 80 de la màquina virtual
```

b) orna a **pj9 → f4 → a7 → pj9f4a7.1** i fes un **add** del continguts del directori **pj9f4a7.1** executant l'ordre: **git add .**

c) Fes un nou **commit** del continguts del directori **pj9f4a7.1**. El missatge del commit serà: *"Versió 1.2 del nou projecte pizzeria - amb docker-compose.yml"*.

d) Comprova que s'ha pujat al dipòsit local el fitxer **docker-compose.yml**.

e) Puja la nova versió del projecte al dipòsit remot i refresca la web del dipòsit remot per assegurar-te que tots les noves versions i el nous fitxers del projecte s'han pujat.

5- Crea la imatge i posa en marxa el contenidor amb l'aplicació utilitzant docker-compose

a) Dins de la **màquina virtual**, accedeix a la carpeta **~/codis/pizzeria** i comprova que tens els següents arxius:

```
vagrant@pj9f4a71-dacomo:~/codis/pizzeria$ ls -a
.  ..  .dockerignore  Dockerfile  app  docker-compose.yml
```

b) A continuació, executa: **docker compose up -d**

c) Si tot ha ana bé, el sistema mostrarà el missatge: ✔ Container cpizza1.2 Started

d) Comprova que s'ha creat:
- La imatge **ipizza:1.2**
- El contenidor **cpizza1.2**

e) Accedeix des d'un navegador de la màquina física a l'aplicació i comprova que funciona.

6- Aturant i esborrant completament l'aplicació.

a) Dins de la **màquina virtual**, accedeix a la carpeta **~/codis/pizzeria** i executa: **docker compose down**

b) Si tot ha ana bé, el sistema mostrarà el missatge: ✔ Container cpizza1.2 Removed

c) Comprova que s'ha esborrat el contenidor **cpizza1.2** però que la imatge **ipizza:1.2** encara existeix.

d) Esborra la imatge **ipizza:1.2** executant: **docker rmi ipizza:1.2**

e) Comprova que la imatge ja no existeix executant: **docker images**

Lliurament de l'activitat

a) Comprovacions dins de la màquina virtual creada amb Vagrant:
- Mostra els fitxers de la carpeta **~/codis/pizzeria**.
- Comprova que no existeix la imatge **ipizza:1.2** ni el contenidor **cpizza1.2**.
- Amb **docker compose** posa en marxa l'aplicació i comprova que es crea la imatge i el contenidor.

b) Des de la màquina física:
- Mostra dins del teu compte de **Github** el dipòsit **pj9f4a7.1** i el seu contingut.
- Mostra des de Github el contigut de **.dockerignore** i **docker-compose.yml**
- Accedeix a l'aplicació i comprova que funciona.

c) Des de la màquina virtual, atura l'aplicació i comprova que l'aplicació deixa de funcionar.

d) Data límit per obtenir el 100% de la nota: **dijous 27-10-24** a les **19.10**. Posteriorment la nosta serà menor.