Контейнеризация

Урок 4. Dockerfile и слои

Задания

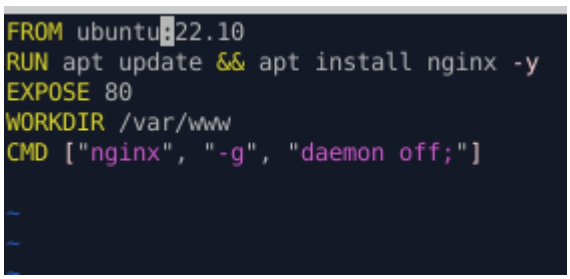
1. Повторить то, что было на семинаре
2. ** Самостоятельно на основе голой ОС (alpine, ubuntu, debian) сделать образ MariaDB (создать Dockerfile) и подключиться к контейнеру пхпмайадмином.

Решение

1. Задача

Создаем *Dockerfile* в папке проекта *nginx*

```
vim Dockerfile
```



Запускаем в папке проекта сборку образа *gb-nginx*

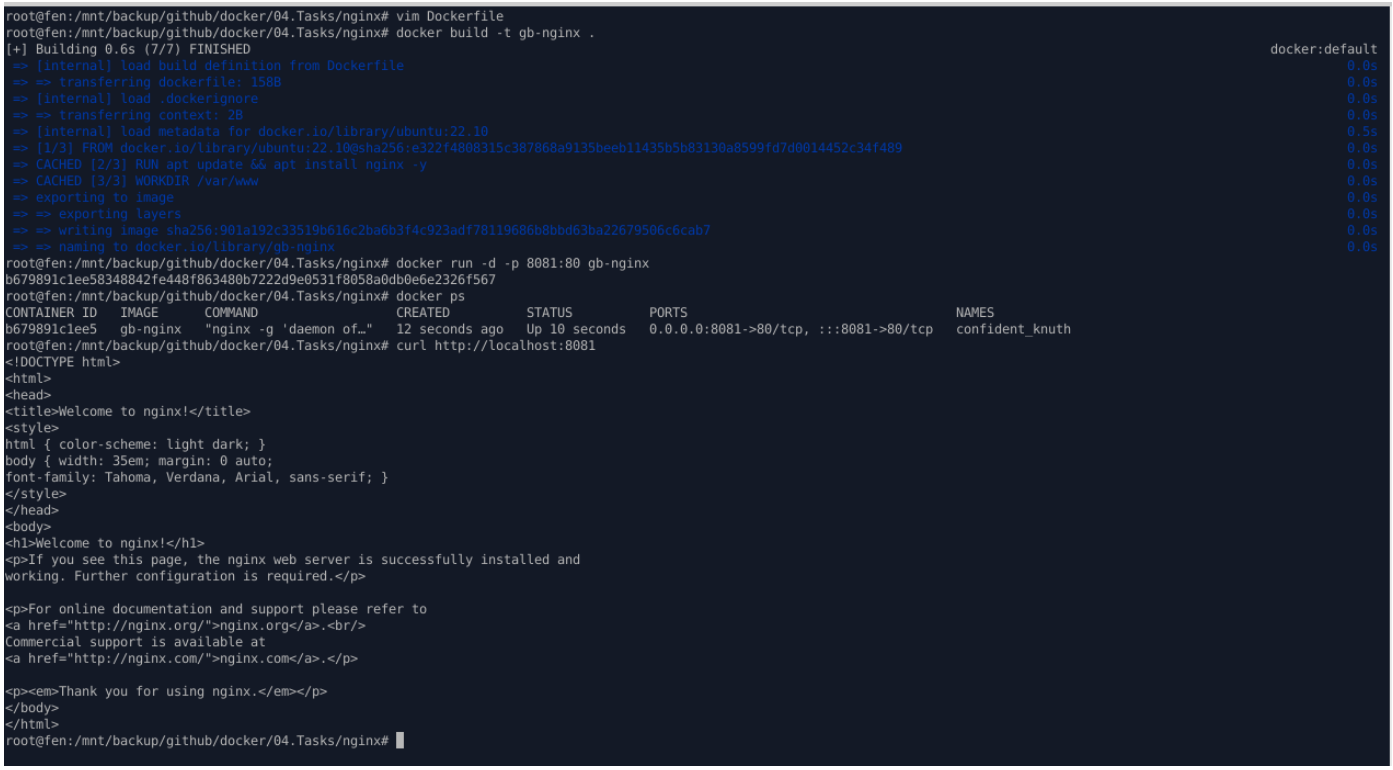
```
docker build -t gb-nginx .
```

Запускаем контейнер на основе образа *gb-nginx* с пробросам порта 80 контейнера на порт 8081 реальной машины

```
docker run -d -p 8081:80 gb-nginx
```

Проверку работы веб-сервера выполним с помощью утилиты *curl*

```
curl http://localhost:8081
```



Виден ответ веб-сервера с *html* страницей *nginx*

2. Задача

Для реализации проекта создам образ включающий сразу три службы:

- **mariadb**
- **php-fpm**
- **nginx**

На веб-сервере будет расположена система управления БД *phpmyadmin*

В папке проекта *maria* для веб-сервера создадим конфиг, включающий работу через *php-fpm*

```
vim nginx.conf
```

```
server {
listen 80;
server_name _;

root /var/www/html/phpmyadmin;

index index.php index.html index.htm;

location / {
try_files $uri $uri/ =404;
}

location ~ \.php$ {
include snippets/fastcgi-php.conf;
fastcgi_pass unix:/run/php/php8.1-fpm.sock;
}
}
~
~
~
```

В той же папке создадим конфиг для системы *phpmyadmin*

```
vim config.inc.php
```

```
<?php
// Настройки аутентификации
$cfg['Servers'][1]['auth_type'] = 'cookie';
$cfg['Servers'][1]['user'] = 'phpmyadmin';
$cfg['Servers'][1]['password'] = 'phpmyadmin';

// Настройки базы данных
$cfg['Servers'][1]['host'] = 'localhost';
$cfg['Servers'][1]['connect_type'] = 'tcp';
$cfg['Servers'][1]['compress'] = false;
$cfg['Servers'][1]['AllowNoPassword'] = false;

// Настройки безопасности
$cfg['blowfish_secret'] = 'secret'; // Секретная фраза для шифрования cookie
$cfg['UploadDir'] = '';
$cfg['SaveDir'] = '';
?>

~
~
~
```

Третьим файлом в папке проекта будет *Dockerfile*

```
vim Dockerfile
```

```
FROM ubuntu:22.04

# Обновляем пакеты и устанавливаем необходимые зависимости
RUN export DEBIAN_FRONTEND=noninteractive && \
    apt-get update && apt-get install -y \
    nginx php-fpm php-mysql php-mbstring php-xml php-zip php-curl \
    mariadb-server mariadb-client wget unzip && \
    rm -rf /var/lib/apt/lists/*

# Копируем конфигурационный файл nginx в контейнер
COPY nginx.conf /etc/nginx/sites-available/default

# Скачиваем и распаковываем архив с phpMyAdmin в папку /var/www/html
RUN wget https://files.phpmyadmin.net/phpMyAdmin/5.1.1/phpMyAdmin-5.1.1-all-languages.zip -O /tmp/phpmyadmin.zip && \
unzip -q /tmp/phpmyadmin.zip -d /var/www/html && \
mv /var/www/html/phpMyAdmin-5.1.1-all-languages /var/www/html/phpmyadmin && \
rm /tmp/phpmyadmin.zip

# Копируем конфигурационный файл phpMyAdmin в контейнер
COPY config.inc.php /var/www/html/phpmyadmin/config.inc.php

# Создаем базу данных и пользователя для phpMyAdmin
RUN service mariadb start && \
mysql -e "CREATE DATABASE gb;" && \
mysql -e "CREATE USER 'geekbrains'@'localhost' IDENTIFIED BY 'test1234';" && \
mysql -e "GRANT ALL PRIVILEGES ON gb.* TO 'geekbrains'@'localhost';" && \
mysql -e "FLUSH PRIVILEGES;" && \
service mariadb stop

# Открываем 80 порт для внешнего доступа
EXPOSE 80

# Запускаем сервисы nginx, php-fpm и mysql при старте контейнера
CMD service mariadb start && service php8.1-fpm start && nginx -g "daemon off;"
```

Сборка образа *gb-maria*

```
docker build -t gb-maria .
```

Запуск контейнера на основе образа *gb-maria* с пробросом порта 80 контейнера на порт 8080 хостовой ОС

```
docker run -d -p 8080:80 gb-maria
```

Заходим с помощью браузера, подключившись к хостовому порту 8080

После успешной авторизации создаем таблицу *students* в базе *gb* и заносим данные.