Задание:

1. Установить Docker в виртуальную машину(или в VDS) – делаю по инструкции [Установка и использование Docker в Ubuntu 20.04](https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04-ru%20)

Обновляем существующий список пакетов:

**sudo apt update**

Устанавливаем пакеты, которые позволяют apt использовать пакеты через HTTPS:

**sudo apt install apt-transport-https ca-certificates curl software-properties-common**

Добавляем ключ GPG для официального репозитория Docker:

**curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**

Добавляем репозиторий Docker в источники APT:

**sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"**

Обновляем базу данных пакетов из добавленного репозитория:

**sudo apt update**

Проверяем установку:

**apt-cache policy docker-ce**

Вывод:



Установим Docker:

**sudo apt install docker-ce**

Проверим, что он запущен:

**sudo systemctl status docker** - Ок.

Проверим установку контейнеров:

 - контейнер установился

 - Ок.

**Устанавливаем docker compose** по инструкции [Как установить Docker Compose на Ubuntu 18.04](https://www.digitalocean.com/community/tutorials/how-to-install-docker-compose-on-ubuntu-18-04)

Проверяем [текущую версию](https://github.com/docker/compose/releases) и обновляемся:



(обратить внимание на буковку v в номере версии!)

Даем разрешение на запуск:



Проверяем:  ответ:  - Ок.

**Настроить набор контейнеров через docker compose по инструкции по ссылке**: [Установка WordPress с помощью Docker Compose](https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-docker-compose-ru).

*Часть с настройкой certbot и HTTPS опустить, если у вас нет настоящего домена и белого IP.*

Создаем в домашней папке директорию wordpress: 

Создаем в этой папке папку с настройками nginx: 

Создаем файл настроек: 

Вносим настройки:

server {

listen 80;

listen [::]:80;

server\_name test.loc www.test.loc;

index index.php index.html index.htm;

root /var/www/html;

location ~ /.well-known/acme-challenge {

allow all;

root /var/www/html;

}

location / {

try\_files $uri $uri/ /index.php$is\_args$args;

}

location ~ \.php$ {

try\_files $uri =404;

fastcgi\_split\_path\_info ^(.+\.php)(/.+)$;

fastcgi\_pass wordpress:9000;

fastcgi\_index index.php;

include fastcgi\_params;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param PATH\_INFO $fastcgi\_path\_info;

}

location ~ /\.ht {

deny all;

}

location = /favicon.ico {

log\_not\_found off; access\_log off;

}

location = /robots.txt {

log\_not\_found off; access\_log off; allow all;

}

location ~\* \.(css|gif|ico|jpeg|jpg|js|png)$ {

expires max;

log\_not\_found off;

}

}

Создаем файл с настройками базы данных:

MYSQL\_ROOT\_PASSWORD=test

MYSQL\_USER=admin

MYSQL\_PASSWORD=test

Вносим файл .env в файл с настройками игнора для docker:

.env

.git

docker-compose.yml

.dockerignore

Создаем файл настроек для docker-compose:



Определяем в нем службы (настройки работы контейнеров):

version: '3'

services:

db:

image: mysql:8.0

container\_name: db

restart: unless-stopped

env\_file: .env

environment:

- MYSQL\_DATABASE=wordpress

volumes:

- dbdata:/var/lib/mysql

command: '--default-authentication-plugin=mysql\_native\_password'

networks:

- app-network

wordpress:

depends\_on:

- db

image: wordpress:5.1.1-fpm-alpine

container\_name: wordpress

restart: unless-stopped

env\_file: .env

environment:

- WORDPRESS\_DB\_HOST=db:3306

- WORDPRESS\_DB\_USER=$MYSQL\_USER

- WORDPRESS\_DB\_PASSWORD=$MYSQL\_PASSWORD

- WORDPRESS\_DB\_NAME=wordpress

volumes:

- wordpress:/var/www/html

networks:

- app-network

webserver:

depends\_on:

- wordpress

image: nginx:1.15.12-alpine

container\_name: webserver

restart: unless-stopped

ports:

- "80:80"

volumes:

- wordpress:/var/www/html

- ./nginx-conf:/etc/nginx/conf.d

networks:

- app-network

volumes:

wordpress:

dbdata:

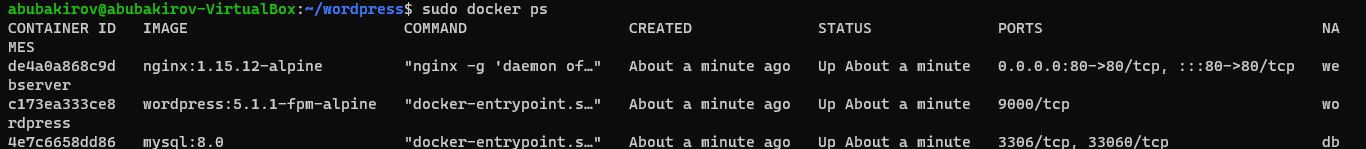
networks:

app-network:

driver: bridge

Запускаем контейнеры: 

Проверяем:



Ок.

Далее, через браузер заходим по адресу 10.0.1.12/test.loc – видим приглашение wordpress, заканчиваем настройку.

2. \* Запустить два контейнера, связанные одной сетью (используя документацию). Первый контейнер БД (например, образ mariadb:10.8), второй контейнер — phpmyadmin. Получить доступ к БД в первом контейнере через второй контейнер (веб-интерфейс phpmyadmin).

Создаем новую папку database и переходим в нее: 

Создаем файл настроек для docker-compose:

Определяем в нем службы (настройки работы контейнеров):

version: '3'

services:

db:

image: mariadb:10.8

container\_name: mariadb

environment:

MARIADB\_ROOT\_PASSWORD: my\_secret\_password

MARIADB\_DATABASE: app\_db

MARIADB\_USER: db\_user

MARIADB\_PASSWORD: db\_user\_pass

ports:

- "6033:3306"

volumes:

- dbdata:/var/lib/mariadb

phpmyadmin:

image: phpmyadmin/phpmyadmin

container\_name: pma

links:

- db

environment:

PMA\_HOST: mariadb

PMA\_PORT: 3306

PMA\_ARBITRARY: 1

restart: always

ports:

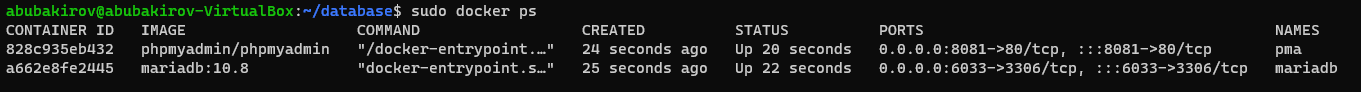
- 8081:80

volumes:

dbdata:

Запускаем контейнеры: 

Проверяем:



Далее, через браузер заходим по адресу 10.0.1.12:8081 – видим приглашение phpmuadmin, вводим сервер – db, user - db\_user, PASSWORD: db\_user\_pass, заканчиваем настройку.