



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЛИПЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет автоматизации и информатики
Кафедра автоматизированных систем управления

Лабораторная работа № 5
по дисциплине “Linux”
«Контейнеризация»»»

Студент АС-21-1

(подпись, дата)

Болдырев М.Р

Руководитель

доцент

(подпись, дата)

Кургасов В.В

Липецк 2023

Цель работы:

изучить современные методы разработки ПО в динамических и распределенных средах на примере контейнеров Docker.

Задачи:

Изучить теоретический материал и выполнить предложенные практические задания.

В результате необходимо:

- Знать назначение и возможности Docker;
- Знать особенности установки и настройки Docker;
- Владеть инструментом для определения и запуска многоконтейнерных приложений

Docker – Docker Compose

Ход работы:

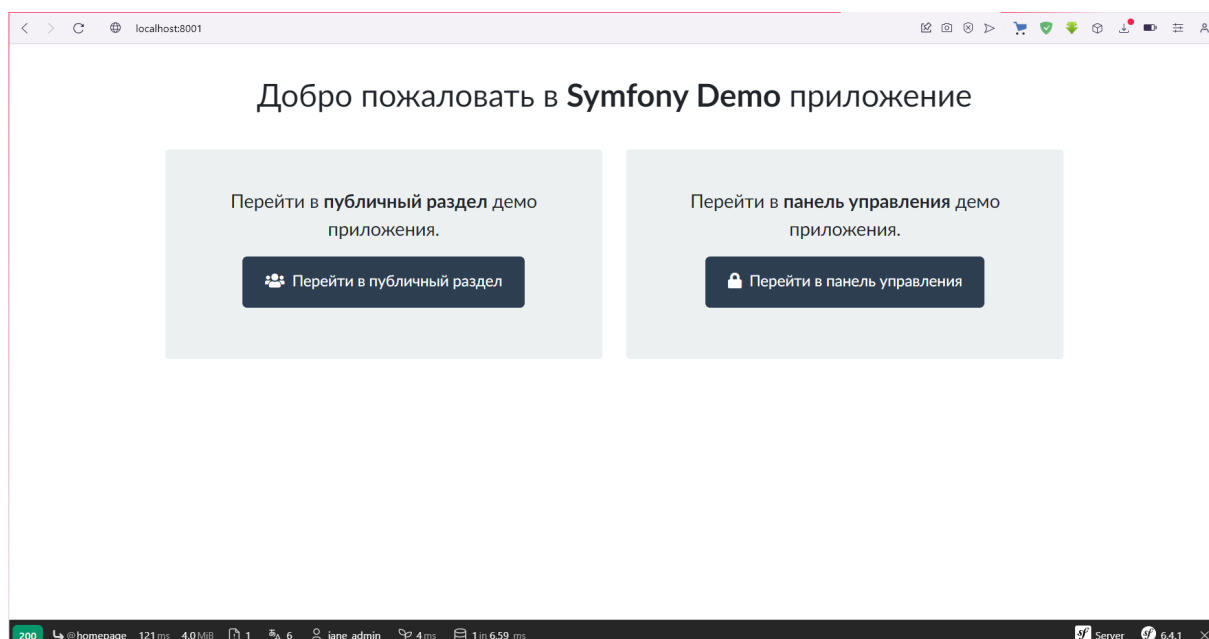
Для начала выполним «git clone https://github.com/symfony/demo» и перейдем в директорию “demo”. Здесь мы запустим наш сервер.

```
root@maxbo: /home/maxim_boldyrev/demo
root@maxbo:/home/maxim_boldyrev/demo# symfony server:start
Following Web Server log file (/root/.symfony5/log/6e8c2ec6f54b1055d1b37e553103e492e8653597.log)
Following PHP log file (/root/.symfony5/log/6e8c2ec6f54b1055d1b37e553103e492e8653597/7daf403c7589f4927632ed3b6af762a992f09b78.log)
WARNING the current dir requires PHP 8.1.0 (composer.json from current dir: /home/maxim_boldyrev/d
emo/composer.json), but this version is not available: fallback to 8.1

[WARNING] The local web server is optimized for local development and MUST never be used in a p
roduction setup.

[OK] Web server listening
The Web server is using PHP CLI 8.1.2
https://127.0.0.1:8001

[Web Server ] Dec 18 16:07:10 |DEBUG | PHP   Reloading PHP versions
[Web Server ] Dec 18 16:07:10 |WARN  | PHP   the current dir requires PHP 8.1.0 (composer.json f
rom current dir: /home/maxim_boldyrev/demo/composer.json), but this version is not available: fall
back to 8.1
[Web Server ] Dec 18 16:07:10 |DEBUG | PHP   Using PHP version 8.1.2 (from composer.json from cu
```



Установим docker и docker-compose. Как результат имеем:

```
root@maxbo: /home/maxim_boldyrev/demo
root@maxbo:/home/maxim_boldyrev/demo# docker info
Client: Docker Engine - Community
Version: 24.0.7
Context: default
Debug Mode: false
Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version: v0.11.2
    Path: /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version: v2.21.0
    Path: /usr/libexec/docker/cli-plugins/docker-compose
Server:
Containers: 3
  Running: 3
  Paused: 0
  Stopped: 0
Images: 17
Server Version: 24.0.7
Storage Driver: overlay2
Backing Filesystem: extfs

root@maxbo:/home/maxim_boldyrev/demo# docker-compose version
docker-compose version 1.25.0-rc4, build 8f3c9c58
docker-py version: 4.1.0
CPython version: 3.7.4
OpenSSL version: OpenSSL 1.1.0l 10 Sep 2019
root@maxbo:/home/maxim_boldyrev/demo#
```

Теперь начнем работу с проектом. Я создаю отдельный проект командой `symfony new —demo symfony`. У нас автоматически создается тот же набор файлов, что и при пуле с гита. На фото ниже представлен уже готовый проект, тем не менее суть остается той же.

```
root@maxbo: /home/maxim_boldyrev/laba/symfony
root@maxbo:/home/maxim_boldyrev/laba# cd symfony/
root@maxbo:/home/maxim_boldyrev/laba/symfony# ls
assets          data            package.json    README.md      vendor
bin             docker          package-lock.json  src            webpack.config.js
composer.json   docker-compose.yml  php-fpm         symfony.lock   yarn.lock
composer.lock   lr.db_backup.sql  phpstan-baseline.neon  templates
conf.d          migrations       phpstan.neon.dist  tests
config          nginx            phpunit.xml.dist  translations
CONTRIBUTING.md nginx.conf        public           var
root@maxbo:/home/maxim_boldyrev/laba/symfony#
```

Для начала я изменю файл .env, и помещу туда нашу базу данных PostgreSQL.

```
root@maxbo: /home/maxim_boldyrev/laba/symfony
APP_SECRET=2ca64f8d83b9e89f5f19d672841d6bb8
#TRUSTED_PROXIES=127.0.0.0/8,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
#TRUSTED_HOSTS='^(localhost|example\..com)$'
###< symfony/framework-bundle ###

###> doctrine/doctrine-bundle ###
# Format described at https://www.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/
configuration.html#connecting-using-a-url
# IMPORTANT: You MUST configure your server version, either here or in config/packages/doctrine.ya
ml
DATABASE_URL="postgresql://postgres:12345@127.0.0.1:5432/lr_db"
# DATABASE_URL="mysql://app:!ChangeMe!@127.0.0.1:3306/app?serverVersion=8&charset=utf8mb4"
# DATABASE_URL="postgresql://app:!ChangeMe!@127.0.0.1:5432/app?serverVersion=15&charset=utf8"
###< doctrine/doctrine-bundle ###

###> symfony/mailer ###
# MAILER_DSN=null://null
###< symfony/mailer ###
".env" 35L, 1700B 27,1 Внизу
```

Теперь начнем работу с Docker. Я планирую написать docker-compose файл, и по папкам директории сделать несколько Dockerfile под каждый из образов. Docker-compose.yml файл выглядит следующем образом:

```
root@maxbo: /home/maxim_boldyrev/laba/symfony
version: '3.0'
services:
  db:
    container_name: db
    image: postgres:12
    restart: always
    environment:
      POSTGRES_PASSWORD: 12345
      POSTGRES_DB: lr_db
    ports:
      - "5432:5432"
    volumes:
      - ./data/postgresql:/var/lib/postgresql/data
      - ./lr_db_backup.sql:/docker-entrypoint-initdb.d/lr_db_backup.sql
  php-fpm:
    container_name: php-fpm
    build:
      context: ./php-fpm
    depends_on:
      - db
    volumes:
      - ../../symfony/:/var/www
  nginx:
    image: nginx
    container_name: nginx
    depends_on:
      - php-fpm
    ports:
      - "8080:80"
      - "443:443"
~
~
~
~
24,16 Весь
```

Dockerfile для nginx:

```
root@maxbo: /home/maxim_boldyrev/laba/symfony/nginx
FROM nginx:alpine
WORKDIR /var/www
EXPOSE 80 443
CMD ["nginx", "-g", "daemon off;"]
```

4,34 Весь

Dockerfile для php-fpm

```
root@maxbo: /home/maxim_boldyrev/laba/symfony/php-fpm
FROM php:8.2-fpm
RUN apt-get update && \
    apt-get install -y --no-install-recommends libssl-dev zlib-dev curl git unzip libxml2-dev libpq-dev lib
zip-dev && \
    pecl install apcu && \
    docker-php-ext-configure pgsql -with-pgsql=/usr/local/pgsql && \
    docker-php-ext-install -j$(nproc) zip opcache intl pdo_pgsql pgsql && \
    docker-php-ext-enable apcu pdo pgsql sodium && \
    apt-get clean && rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
COPY --from=composer:latest /usr/bin/composer /usr/bin/composer
WORKDIR /var/www
CMD composer install -o; php-fpm
```

"Dockerfile" 12L, 581B 3,83 Весь

Теперь создаем схему для базы данных

```
root@maxbo:/home/maxim_boldyrev/laba/symfony# php bin/console doctrine:schema:create

! [CAUTION] This operation should not be executed in a production environment!

Creating database schema...

[OK] Database schema created successfully!
```


docker-compose up:

```
root@maxbo: /home/maxim_boldyrev/laba/symfony/php-fpm
root@maxbo:/home/maxim_boldyrev/laba/symfony/php-fpm# docker-compose up -d
Creating network "symfony_default" with the default driver
Creating db ... done
Creating php-fpm ... done
Creating nginx ... done
root@maxbo:/home/maxim_boldyrev/laba/symfony/php-fpm#
```


Открываем проект в браузере:

Добро пожаловать в **Symfony Demo** приложение

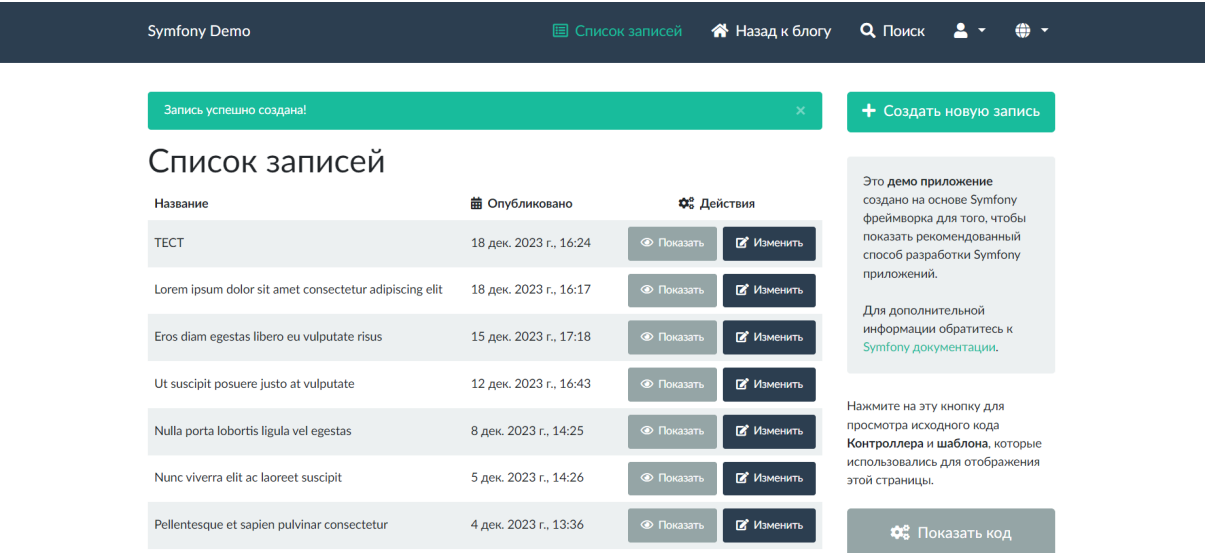
Перейти в публичный раздел демо приложения.

 Перейти в публичный раздел

Перейти в панель управления демо приложения.

 Перейти в панель управления

Добавление записей:

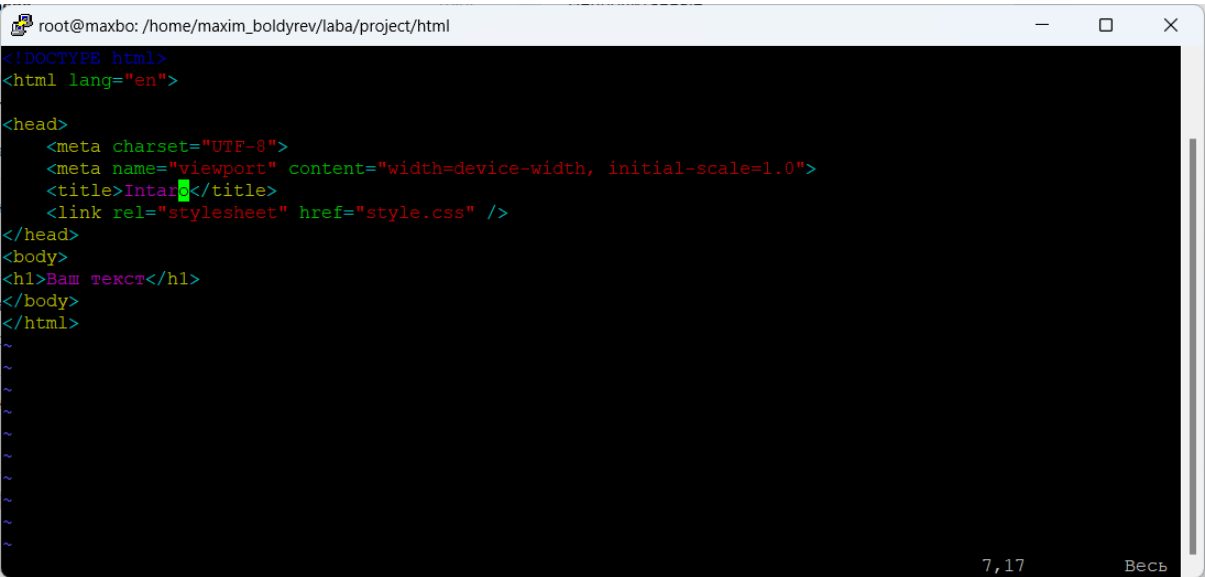


Часть 2:

Шаг 1: Установка nginx.



Шаг 2. Создадим файл index.html. Выведем сообщение: Ваш текст.



Также начнем писать docker-compose файл для работы с nginx.


```
root@maxbo: /home/maxim_boldyrev/laba/project
version: '3.0'

services:
  nginx:
    image: nginx
    ports:
      - "8080:80"
    volumes:
      - ./html:/usr/share/nginx/html
```

"docker-compose.yml" 10L, 132B 8,17 Весь

В результате получаем

Ваш текст

Шаг 3.

Docker-compose файл

```
version: '3.0'
services:
  nginx:
    image: nginx
    environment:
      - VIRTUAL_HOST=site.local
    depends_on:
      - php
    volumes:
      - ./html:/usr/share/nginx/html
    networks:
      - frontend
      - backend

  php:
    build:
      context: ./php
    volumes:
      - ./docker/php/php.ini:/usr/local/etc/php/php.ini
      - ./html:/var/www/html/
    networks:
      - backend

  mysql:
    image: mysql:5.7
    volumes:
      - ./mysql/data:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=root
    networks:
      - backend

  phpmyadmin:
    image: phpmyadmin/phpmyadmin:latest
    environment:
      - VIRTUAL_HOST=phpmyadmin.local
      - PMA_HOST=mysql
      - PMA_USER=root
      - PMA_PASSWORD=root
    networks:
      - frontend
      - backend

networks:
  frontend:
    external:
      name: proxy_proxy
  backend:
```

Dockerfile для php

```
FROM php:8.2-fpm

RUN apt-get update && apt-get install -y \
    libzip-dev \
    zip \
    && docker-php-ext-configure zip \
    && docker-php-ext-install zip \
    && docker-php-ext-install mysqli

COPY --from=composer:latest /usr/bin/composer /usr/bin/composer

WORKDIR /var/www/html
```

Docker-compose для proxy

```
version: '3.0'

services:
  proxy:
    image: jwilder/nginx-proxy
    ports:
      - 80:80
    volumes:
      - /var/run/docker.sock:/tmp/docker.sock:ro
    networks:
      - proxy

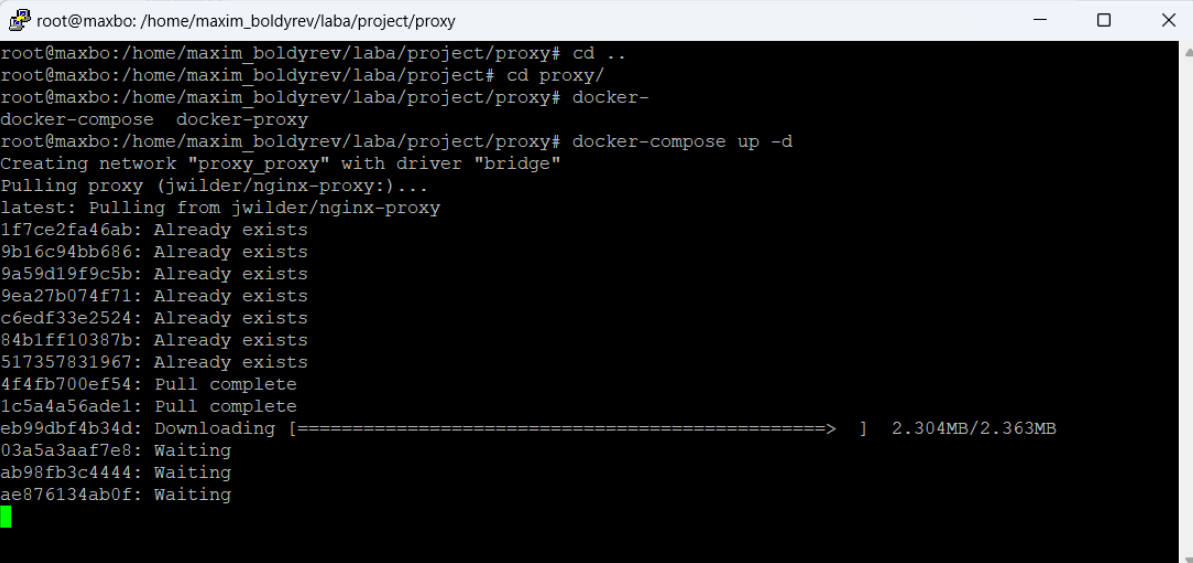
networks:
  proxy:
    driver: bridge
```

index.php

```
<?php
//phpinfo();
$link = mysqli_connect('mysql', 'root', 'root');
if (!$link) {
    die('Ошибка соединения: ' . mysqli_error());
}

echo 'Успешно соединено';
mysqli_close($link);
```

docker-compose up

A terminal window with a title bar showing the path /home/maxim_boldyrev/laba/project/proxy. The terminal output shows the user navigating to the proxy directory and running docker-compose up -d. It creates a network named 'proxy proxy' and pulls the jwilder/nginx-proxy image. The pull progress shows several layers already existing and one layer being downloaded (2.304MB/2.363MB). Several containers are shown in a 'Waiting' state.

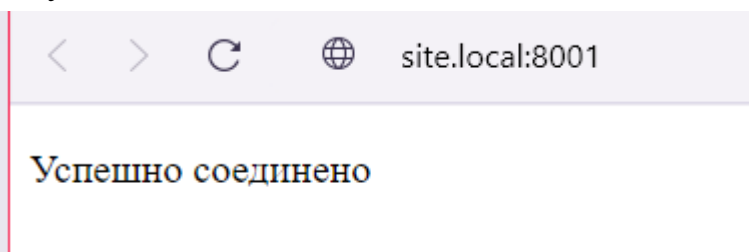
```
root@maxbo: /home/maxim_boldyrev/laba/project/proxy
root@maxbo:/home/maxim_boldyrev/laba/project/proxy# cd ..
root@maxbo:/home/maxim_boldyrev/laba/project# cd proxy/
root@maxbo:/home/maxim_boldyrev/laba/project/proxy# docker-
docker-compose  docker-proxy
root@maxbo:/home/maxim_boldyrev/laba/project/proxy# docker-compose up -d
Creating network "proxy proxy" with driver "bridge"
Pulling proxy (jwilder/nginx-proxy:...)
latest: Pulling from jwilder/nginx-proxy
1f7ce2fa46ab: Already exists
9b16c94bb686: Already exists
9a59d19f9c5b: Already exists
9ea27b074f71: Already exists
c6edf33e2524: Already exists
84b1ff10387b: Already exists
517357831967: Already exists
4f4fb700ef54: Pull complete
1c5a4a56ade1: Pull complete
eb99dbf4b34d: Downloading [=====> ] 2.304MB/2.363MB
03a5a3aaf7e8: Waiting
ab98fb3c4444: Waiting
ae876134ab0f: Waiting
```

docker network

```
root@maxbo: /home/maxim_boldyrev/laba/project/proxy
9b16c94bb686: Already exists
9a59d19f9c5b: Already exists
9ea27b074f71: Already exists
c6edf33e2524: Already exists
84b1ff10387b: Already exists
517357831967: Already exists
4f4fb700ef54: Pull complete
1c5a4a56ade1: Pull complete
eb99dbf4b34d: Pull complete
03a5a3aaf7e8: Pull complete
ab98fb3c4444: Pull complete
ae876134ab0f: Pull complete
Digest: sha256:09541d391fceb70d6d671e1d2a2370db883fdb8d28d09248fdb6e9b7cae70029
Status: Downloaded newer image for jwilder/nginx-proxy:latest
Creating proxy_proxy_1 ... done
root@maxbo:/home/maxim_boldyrev/laba/project/proxy# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
5ce4a5265f6c        bridge              bridge              local
fcb63af54570        demo_default        bridge              local
f88438eed888        host                host                local
02ce7c15baef        none                null                local
2c7bfb6c71d4        project_default     bridge              local
ad0f7fa93f66        proxy_proxy         bridge              local
root@maxbo:/home/maxim_boldyrev/laba/project/proxy#
```

```
root@maxbo: /home/maxim_boldyrev/laba/project
root@maxbo:/home/maxim_boldyrev/laba/project# docker-compose up -d
ERROR: build path /home/maxim_boldyrev/laba/project/docker/php either does not exist, is not accessible, or is not a valid URL.
root@maxbo:/home/maxim_boldyrev/laba/project# vi docker-compose.yml
root@maxbo:/home/maxim_boldyrev/laba/project# docker-compose up -d
ERROR: build path /home/maxim_boldyrev/laba/project/project/php either does not exist, is not accessible, or is not a valid URL.
root@maxbo:/home/maxim_boldyrev/laba/project# vi docker-compose.yml
root@maxbo:/home/maxim_boldyrev/laba/project# docker-compose up -d
Creating network "project_backend" with the default driver
Building php
Step 1/5 : FROM php:7.3.2-fpm
7.3.2-fpm: Pulling from library/php
f7e2b70d04ae: Downloading [=====>] 8.956MB/22.5MB
744aedb7995c: Download complete
07afe22f8a58: Downloading [==>] 1.618MB/67.45MB
c7bf4f31c4a4: Download complete
f6f33e903915: Waiting
b114f0d30d48: Waiting
eec1339ad6f3: Waiting
e43480cddc87: Waiting
d80e51d1df6: Waiting
f57165e6872a: Waiting
```

Результат:



Шаг №4.

Для запуска Wordpress на уже готовом нашем сервере, добавим папку wordpress и допишем пару моментов в docker-compose.yml, добавив туда загрузку wordpress.

```
version: "3"
services:
  nginx:
    image: nginx
    environment:
      - VIRTUAL_HOST=site.local
    depends_on:
      - php
    ports:
      - "8082:80"
    volumes:
      - ./wordpress:/var/www/html/
      - ./docker/nginx/conf.d/default.conf:/etc/nginx/conf.d/default.conf
    networks:
      - frontend
      - backend
  php:
    build:
      context:
        ./docker/php
    volumes:
      - ./docker/php/php.ini:/usr/local/etc/php/php.ini
      - ./wordpress:/var/www/html/
    networks:
      - backend
  mysql:
    image: mysql:5.7
    volumes:
      - ./docker/mysql/data:/var/lib/mysql
    environment:
      - MYSQL_ROOT_PASSWORD=root
      - MYSQL_DATABASE=wordpress
    networks:
      - backend
  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    ports:
      - "8083:80"
    environment:
      - VIRTUAL_HOST=phpmyadmin.local
      - PMA_HOST=mysql
      - PMA_USER=root
      - MYSQL_ROOT_PASSWORD=root
```

```

networks:
- frontend
- backend
wordpress:
depends_on:
26
- mysql
image: wordpress:5.1.1-fpm-alpine
container_name: wordpress
restart: unless-stopped
env_file: .env
environment:
- WORDPRESS_DB_HOST=mysql:3306
- WORDPRESS_DB_USER=root
- WORDPRESS_DB_PASSWORD=root
- WORDPRESS_DB_NAME=wordpress
volumes:
- ./wordpress/:/var/www/html/
networks:
- frontend
- backend
networks:
frontend:
external:
name: proxy_proxy
backend:

```

Результат:

