

**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра Автоматизированных систем управления

**ЛАБОРАТОРНАЯ РАБОТА №5**

по дисциплине «Операционная система Linux»

Настройка среды Docker

Студентка

Пехова А.А.

Группа ПИ-19

Руководитель

Кургасов В.В.

К.П.Н.

Липецк 2021

Оглавление	
<u>Цель работы</u> .....	2
<u>Задание кафедры</u> .....	3
<u>Ход работы</u> .....	4
<u>Вывод</u> .....	16
<u>Контрольные вопросы</u> .....	17

### Цель работы

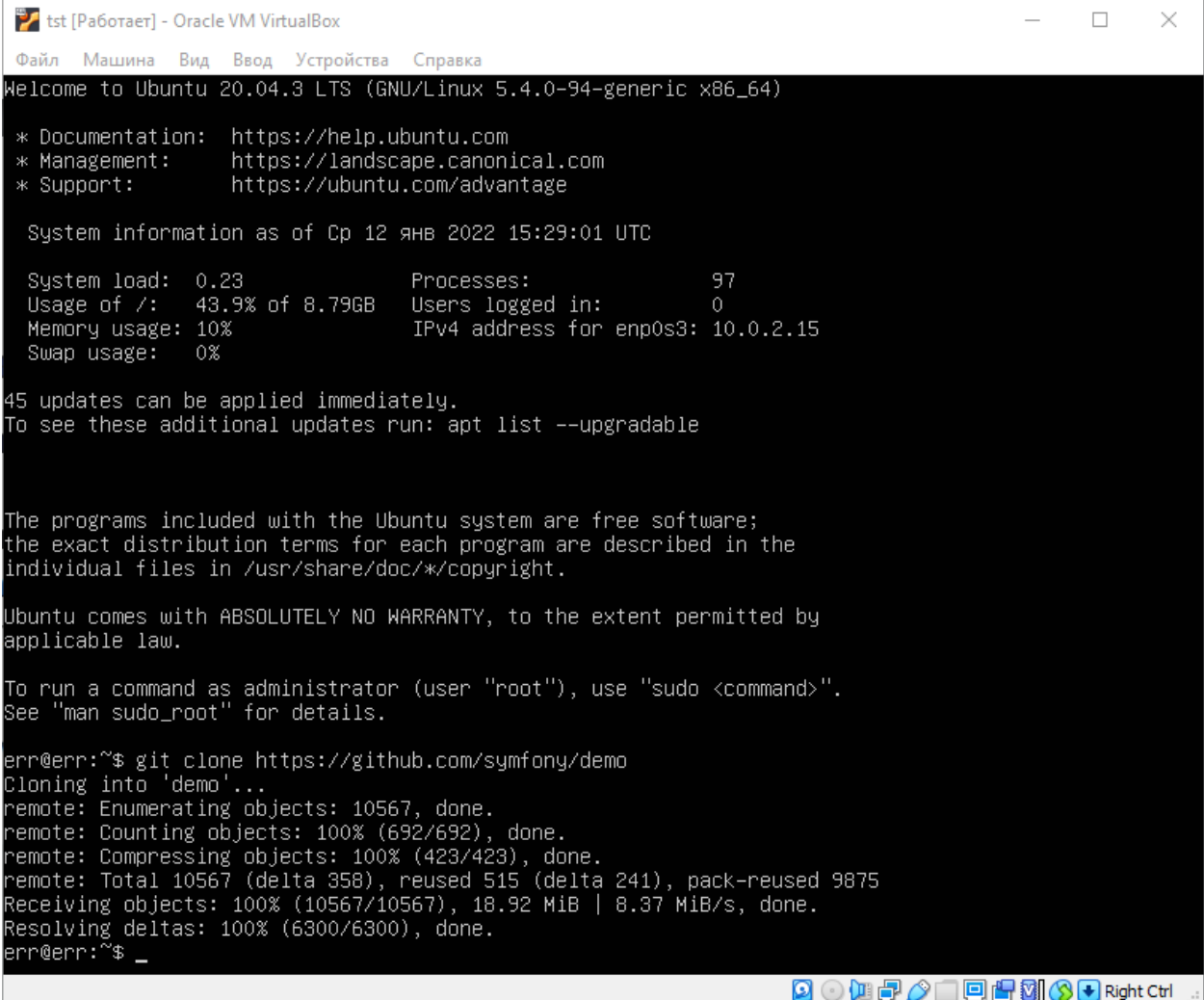
Изучить современные методы разработки ПО в динамических и распределенных средах на примере контейнеров Docker.

### Задание кафедры

Изучить основные этапы работы с контейнерами Docker. С помощью docker-compose на своем компьютере поднять сборку nginx+php-fpm+postgres, продемонстрировать ее работоспособность, запустив внутри контейнера мини-проект на symfony (ссылка на github).

## Ход работы

Для начала установим docker, docker-compose, composer, symfony, драйвер pdo\_postgresql, проведем настройку окружения и склонируем репозиторий проекта demo.



```
tst [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-94-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of Ср 12 янв 2022 15:29:01 UTC

System load:  0.23           Processes:           97
Usage of /:   43.9% of 8.79GB Users logged in:      0
Memory usage: 10%           IPv4 address for enp0s3: 10.0.2.15
Swap usage:   0%

45 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

err@err:~$ git clone https://github.com/symfony/demo
Cloning into 'demo'...
remote: Enumerating objects: 10567, done.
remote: Counting objects: 100% (692/692), done.
remote: Compressing objects: 100% (423/423), done.
remote: Total 10567 (delta 358), reused 515 (delta 241), pack-reused 9875
Receiving objects: 100% (10567/10567), 18.92 MiB | 8.37 MiB/s, done.
Resolving deltas: 100% (6300/6300), done.
err@err:~$ _
```

Создадим БД в postgres.

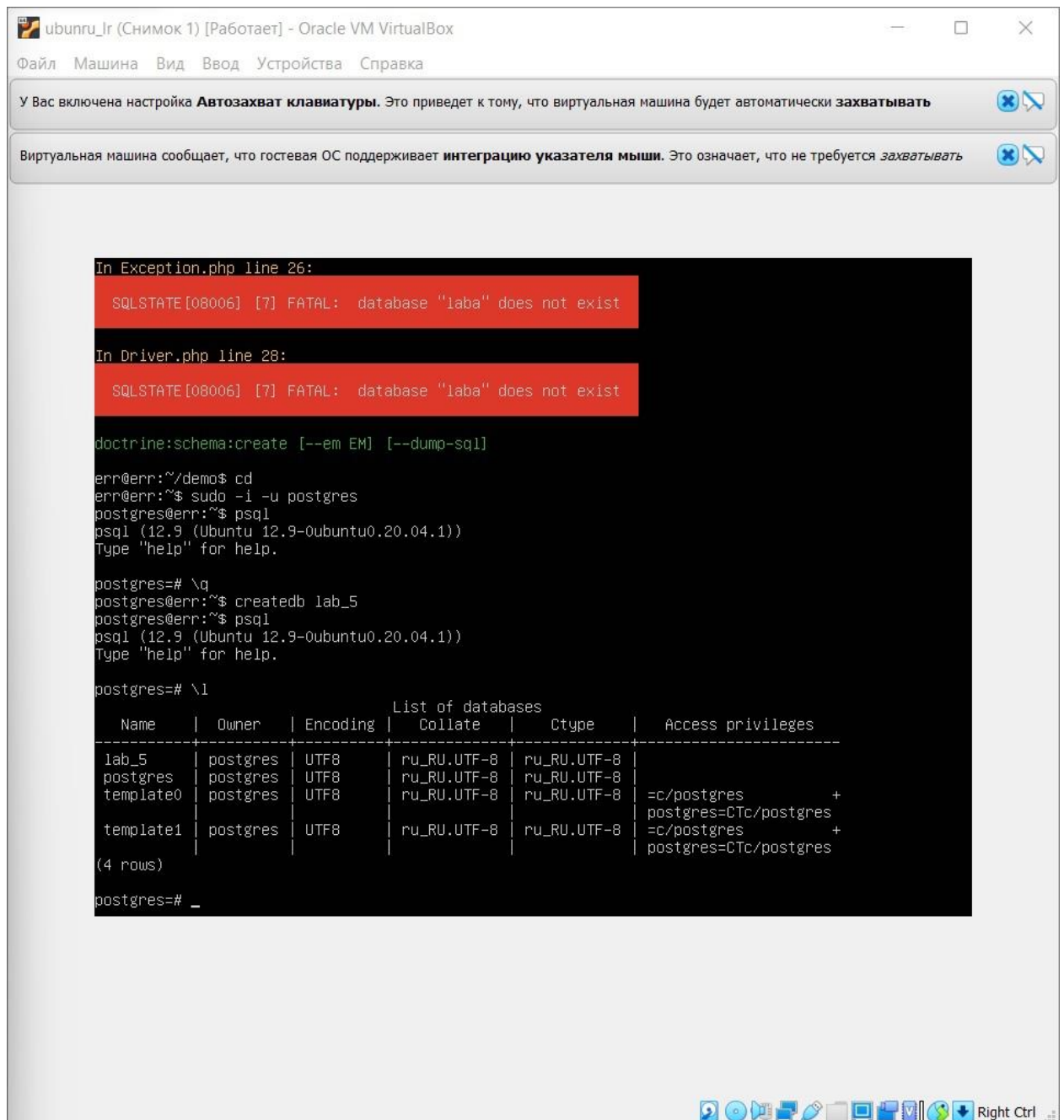
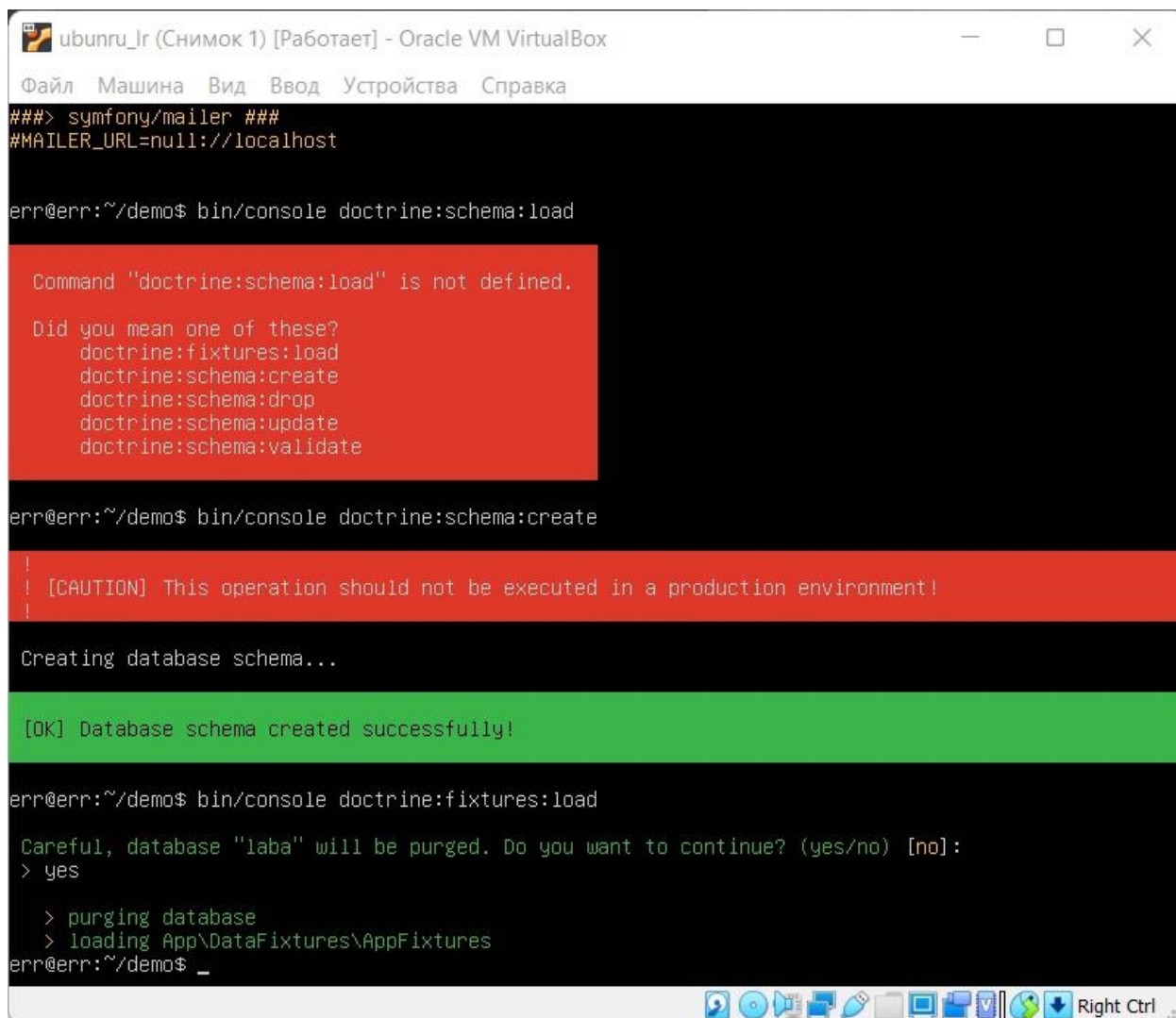


Рисунок 2 – Создание БД

Перейдем в папку проекта, создадим таблицы в БД и загрузим данные, используя команды:

- php bin/console doctrine:schema:create
- php bin/console doctrine:fixtures:load



```
ubuntu_l_r (Снимок 1) [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
###> symfony/mailer ###
#MAILER_URL=null://localhost

err@err:~/demo$ bin/console doctrine:schema:load

Command "doctrine:schema:load" is not defined.

Did you mean one of these?
doctrine:fixtures:load
doctrine:schema:create
doctrine:schema:drop
doctrine:schema:update
doctrine:schema:validate

err@err:~/demo$ bin/console doctrine:schema:create

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

Creating database schema...

[OK] Database schema created successfully!

err@err:~/demo$ bin/console doctrine:fixtures:load

Careful, database "laba" will be purged. Do you want to continue? (yes/no) [no]:
> yes

    > purging database
    > loading App\DataFixtures\AppFixtures
err@err:~/demo$ _
```

Рисунок 3 – Создание таблиц и их заполнение

Далее установим nginx и настроим файлы проекта.

Содержимое файла demo/.env:

```
###> symfony/framework-bundle ###
```

```
APP_ENV=dev
```

```
APP_SECRET=743df4115e7e1dea13b473da07c09fe6
```

```
###< symfony/framework-bundle ###
```

```
###> doctrine/doctrine-bundle ###
```

```
DATABASE_URL="postgresql://postgres:password@127.0.0.1:15432/lab_5?serverVersion=13&charset=utf8"
```

```
###< doctrine/doctrine-bundle ###
```

```
###> nelmio/cors-bundle ###
```

```
CORS_ALLOW_ORIGIN='^https?:/(localhost|127\.\0\.\0\1)(:[0-9]+)?$'
```

```
###< nelmio/cors-bundle ###
```

Содержимое файла docker/.env:

```
###> symfony/framework-bundle ###
```

```
APP_ENV=dev
```

```
APP_SECRET=743df4115e7e1dea13b473da07c09fe6
```

```
###< symfony/framework-bundle ###
```

```
###> doctrine/doctrine-bundle ###
```

```
DATABASE_URL="postgresql://postgres:password@db:5432/lab_5?serverVersion=13&charset=utf8"
```

```
###< doctrine/doctrine-bundle ###
```

```
###> nelmio/cors-bundle ###
```

```
CORS_ALLOW_ORIGIN='^https?:/(localhost|127\.\0\.\0\1)(:[0-9]+)?$'
```

```
###< nelmio/cors-bundle ###
```

Содержимое файла docker/docker-compose.yml:

```
version: '3.8'
```

```
services:
```

```
php-fpm:
```

```
container_name: php-fpm
```

```
build:
```

```
context: ./php-fpm
```



depends\_on:

- db

environment:

- APP\_ENV=\${APP\_ENV}
- APP\_SECRET=\${APP\_SECRET}
- DATABASE\_URL=\${DATABASE\_URL}

volumes:

- ../../demo:/var/www

nginx:

container\_name: nginx

build:

context: ./nginx

volumes:

- ../../demo:/var/www
- ./nginx/nginx.conf:/etc/nginx/nginx.conf
- ./nginx/sites:/etc/nginx/sites-available
- ./nginx/conf.d:/etc/nginx/conf.d
- ./logs:/var/log

depends\_on:

- php-fpm

ports:

- "80:80"

- "443:443"

db:

container\_name: db

image: postgres:12

restart: always

environment:

POSTGRES\_USER: postgres

POSTGRES\_PASSWORD: password

POSTGRES\_DB: dbtest

ports:

- "15432:5432"

volumes:

- ./pg-data:/var/lib/postgresql/data

Содержимое файла docker/nginx/Dockerfile:

FROM nginx:alpine

WORKDIR /var/www

CMD ["nginx"]

EXPOSE 80 443

Содержимое файла docker/php-fpm/Dockerfile:

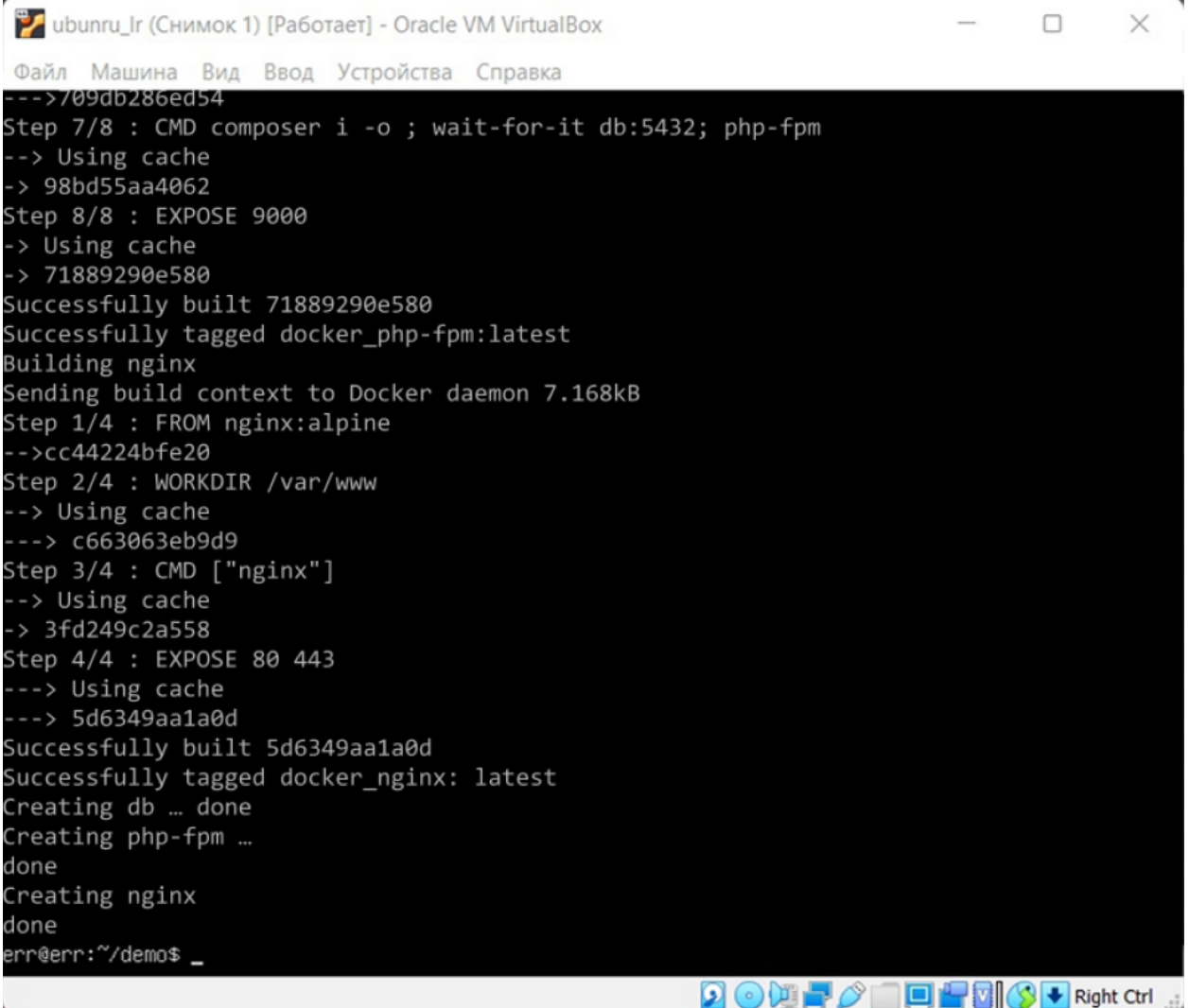
FROM php:8.0-fpm

COPY wait-for-it.sh /usr/bin/wait-for-it

RUN chmod +x /usr/bin/wait-for-it

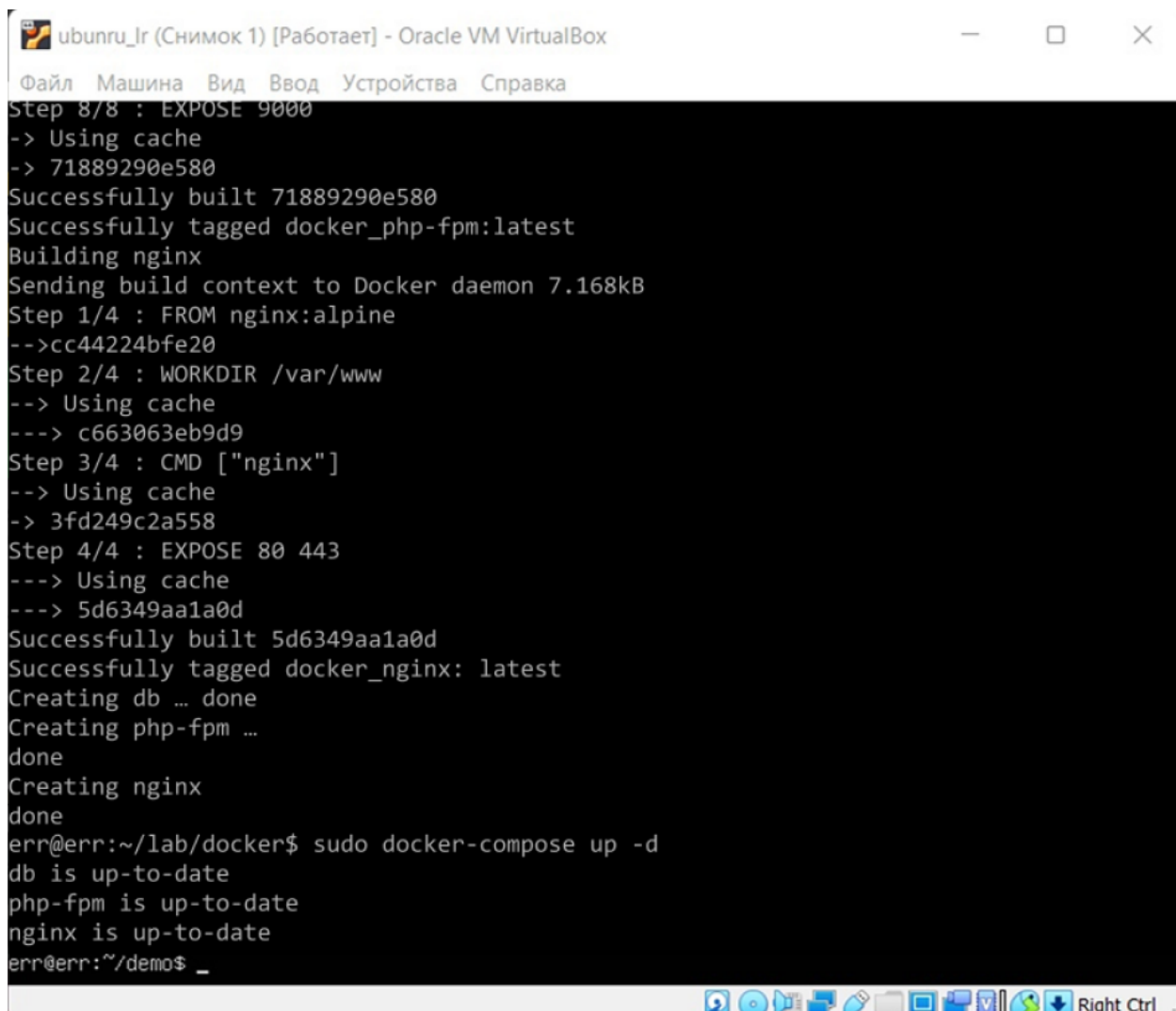
```
RUN apt-get update && \
apt-get install -y --no-install-recommends libssl-dev zlib1g-dev
curl git unzip netcat libxml2-dev libpq-dev libzip-dev && \
pecl install apcu && \
docker-php-ext-configure pgsql -with-pgsql=/usr/local/pgsql && \
docker-php-ext-install -j$(nproc) zip opcache intl pdo_pgsql pgsql
&& \
docker-php-ext-enable apcu pdo_pgsql sodium && \
apt-get clean && rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
COPY --from=composer /usr/bin/composer /usr/bin/composer
WORKDIR /var/www
CMD composer i -o ; wait-for-it db:5432; php-fpm
```

## EXPOSE 9000



```
ubunru_lr (Снимок 1) [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
-->709db286ed54
Step 7/8 : CMD composer i -o ; wait-for-it db:5432; php-fpm
--> Using cache
-> 98bd55aa4062
Step 8/8 : EXPOSE 9000
-> Using cache
-> 71889290e580
Successfully built 71889290e580
Successfully tagged docker_php-fpm:latest
Building nginx
Sending build context to Docker daemon 7.168kB
Step 1/4 : FROM nginx:alpine
-->cc44224bfe20
Step 2/4 : WORKDIR /var/www
--> Using cache
---> c663063eb9d9
Step 3/4 : CMD ["nginx"]
--> Using cache
-> 3fd249c2a558
Step 4/4 : EXPOSE 80 443
---> Using cache
---> 5d6349aa1a0d
Successfully built 5d6349aa1a0d
Successfully tagged docker_nginx: latest
Creating db ... done
Creating php-fpm ...
done
Creating nginx
done
err@err:~/demo$ _
```

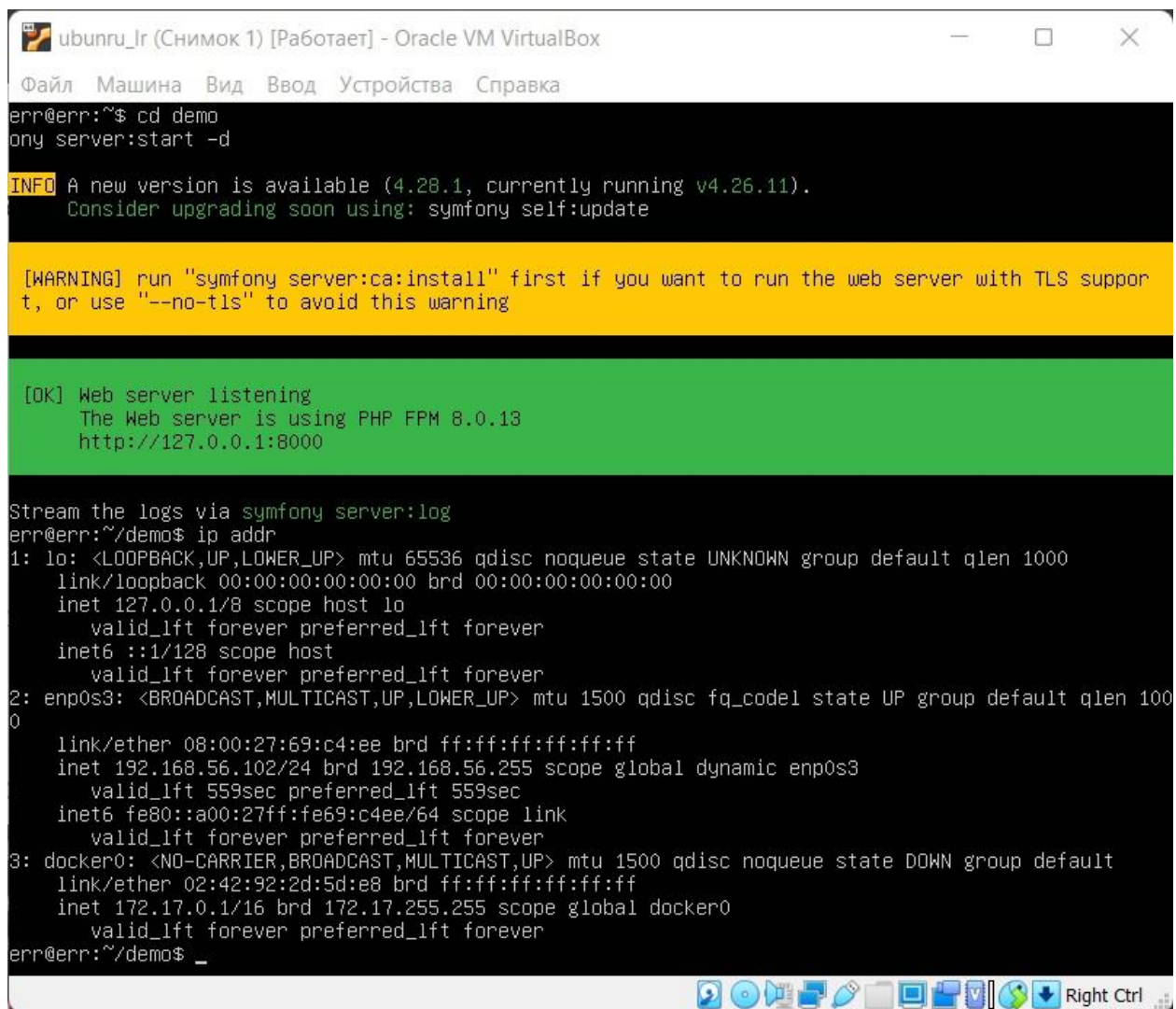
Рисунок 4 – Сборка образа



```
ubunru_lr (Снимок 1) [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Step 8/8 : EXPOSE 9000
-> Using cache
-> 71889290e580
Successfully built 71889290e580
Successfully tagged docker_php-fpm:latest
Building nginx
Sending build context to Docker daemon 7.168kB
Step 1/4 : FROM nginx:alpine
-->cc44224bfe20
Step 2/4 : WORKDIR /var/www
--> Using cache
---> c663063eb9d9
Step 3/4 : CMD ["nginx"]
--> Using cache
-> 3fd249c2a558
Step 4/4 : EXPOSE 80 443
---> Using cache
---> 5d6349aa1a0d
Successfully built 5d6349aa1a0d
Successfully tagged docker_nginx: latest
Creating db ... done
Creating php-fpm ...
done
Creating nginx
done
err@err:~/lab/docker$ sudo docker-compose up -d
db is up-to-date
php-fpm is up-to-date
nginx is up-to-date
err@err:~/demo$ _
```

Рисунок 5 – Инициализация БД

Настроим сеть, поменяв NAT на Host-only adapter, в дополнительных настройках указав Allow VM. И запустим проект командой `symfony server:start -d`.



```
ubuntu_l_r (Снимок 1) [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
err@err:~$ cd demo
ony server:start -d

INFO  A new version is available (4.28.1, currently running v4.26.11).
      Consider upgrading soon using: symfony self:update

[WARNING] run "symfony server:ca:install" first if you want to run the web server with TLS support, or use "--no-tls" to avoid this warning

[OK] Web server listening
      The Web server is using PHP FPM 8.0.13
      http://127.0.0.1:8000

Stream the logs via symfony server:log
err@err:~/demo$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:69:c4:ee brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global dynamic enp0s3
        valid_lft 559sec preferred_lft 559sec
    inet6 fe80::a00:27ff:fe69:c4ee/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:92:2d:5d:e8 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever
err@err:~/demo$ _
```

Рисунок 5 – Инициализация БД

Зайдем в браузер на нужный нам адрес.

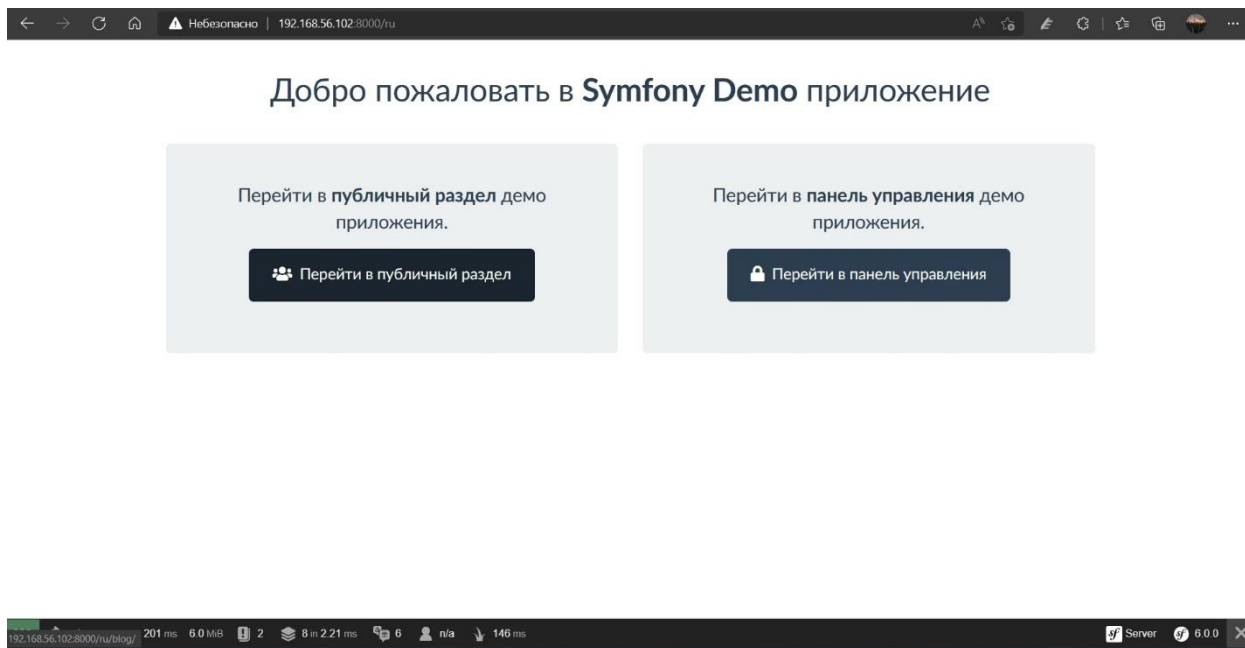


Рисунок 6 – Вход на страницу

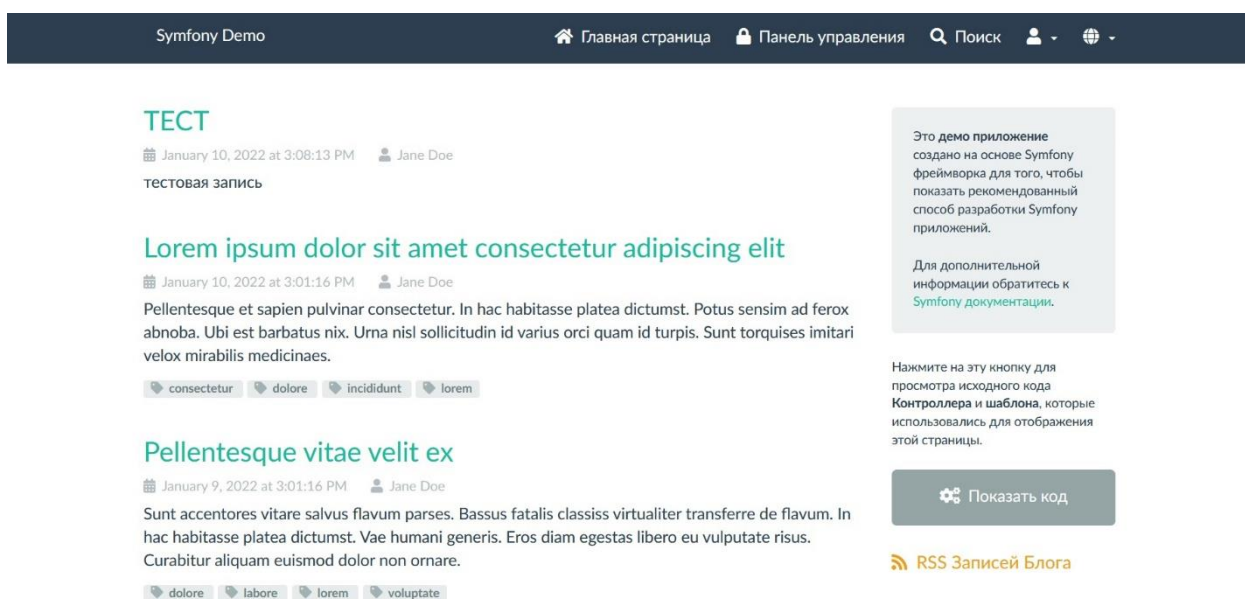


Рисунок 7 – Добавление записи



## Добро пожаловать

Добро пожаловать в знаменитую пятиминутную установку WordPress! Просто заполните поля — и вперёд, к использованию самой мощной и гибкой персональной платформы для публикаций в мире!

## Требуется информация

Пожалуйста, укажите следующую информацию. Не переживайте, потом вы всегда сможете изменить эти настройки.

Название сайта

Имя пользователя

Имя пользователя может содержать только латинские буквы, пробелы, подчёркивания, дефисы, точки и символ @.

Пароль



**Важно:** Этот пароль понадобится вам для входа. Сохраните его в надёжном месте.

Подтвердите  
пароль

☒ Разрешить использование слабого пароля.

Ваш e-mail

Внимательно проверьте адрес электронной почты, перед тем как продолжить.

Видимость для  
поисковых систем

☒ Попросить поисковые системы не индексировать сайт  
Будет ли учитываться этот запрос — зависит от поисковых систем.

Установить WordPress

Рисунок 8 – Настройка Wordpress



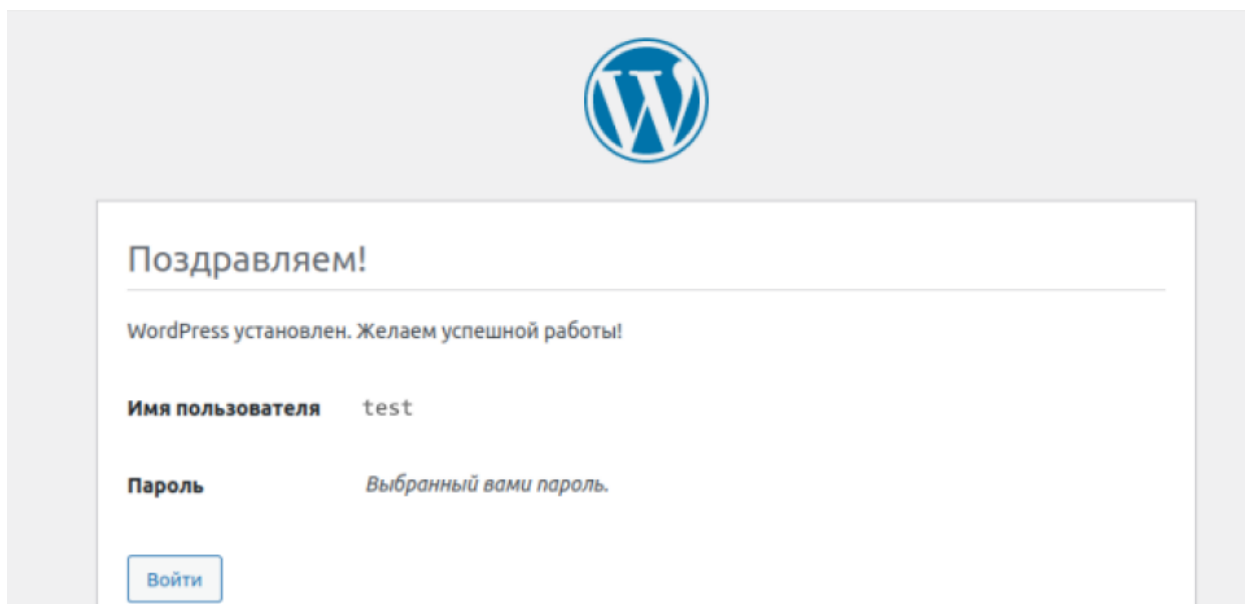


Рисунок 9 – Завершение настройки Wordpress

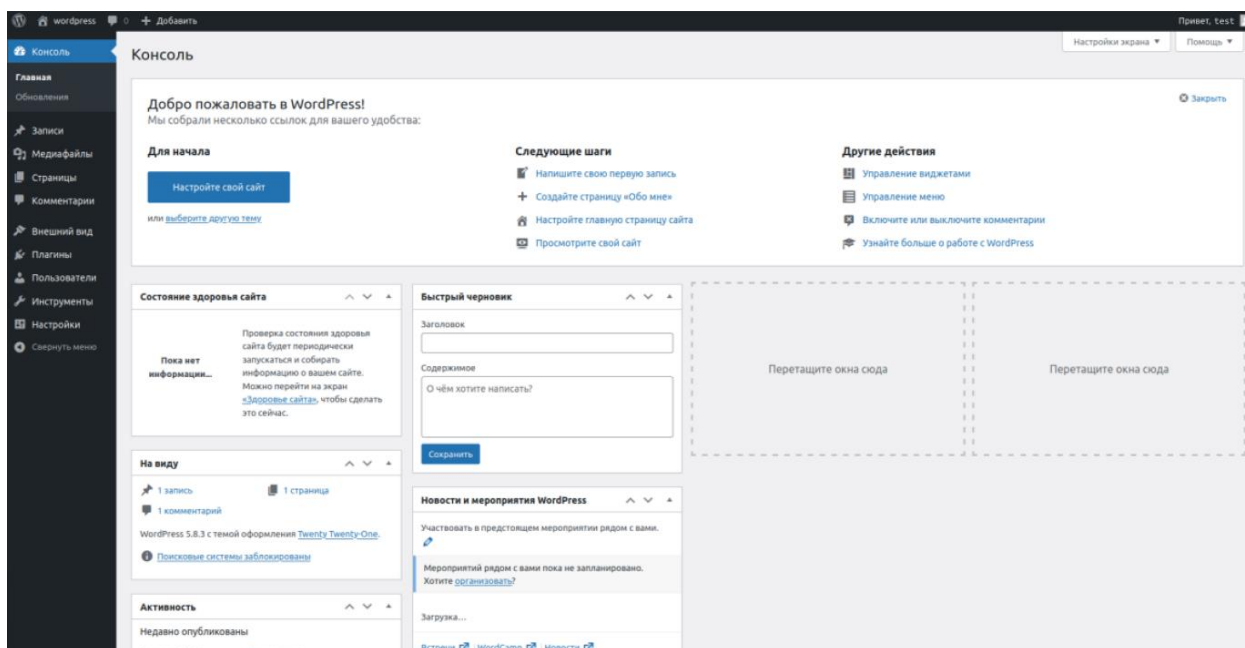


Рисунок 9 – Главная страница Wordpress

## Вывод

В ходе выполнения лабораторной работы были изучены современные методы разработки ПО в динамических и распределенных средах на примере контейнеров Docker.

## Контрольные вопросы

1. Назовите отличия использования контейнеров по сравнению с виртуализацией.
  - A. Меньшие накладные расходы на инфраструктуру
2. Назовите основные компоненты Docker.
  - B. Контейнеры
3. Какие технологии используются для работы с контейнерами?
  - C. Контрольные группы (cgroups)
4. Найдите соответствие между компонентом и его описанием:
  - образы – доступные только для чтения шаблоны приложений;
  - контейнеры – изолированные при помощи технологий операционной системы пользовательские окружения, в которых выполняются приложения;
  - реестры (репозитории) – сетевые хранилища образов.
5. В чем отличие контейнеров от виртуализации?

Виртуальная машина – программная и/или аппаратная система, эмулирующая аппаратное обеспечение некоторой целевой и исполняющая программы для гостевой платформы на платформе-хозяине (хосте) или виртуализирующая некоторую платформу и создающая на ней среды, изолирующие друг от друга программы и даже операционные системы. Виртуальные машины запускают на физических машинах, используя гипервизор.

В отличие от виртуальной машины, обеспечивающей аппаратную виртуализацию, контейнер обеспечивает виртуализацию на уровне операционной системы с помощью абстрагирования пользовательского пространства.

В целом контейнеры выглядят как виртуальные машины. Например, у них есть изолированное пространство для запуска приложений, они позволяют выполнять команды с правами суперпользователя, имеют частный сетевой

интерфейс и IP-адрес, пользовательские маршруты и правила межсетевого экрана и т. д.

Одна большая разница между контейнерами и виртуальными машинами в том, что контейнеры разделяют ядро хоста с другими контейнерами.

6. Перечислите основные команды утилиты Docker с их кратким описанием.

- `docker ps` — показывает список запущенных контейнеров;
- `docker pull` — скачать определённый образ или набор образов (репозиторий);
- `docker build` — эта команда собирает образ Docker из Dockerfile и «контекста»;
- `docker run` — запускает контейнер, на основе указанного образа;
- `docker logs` — эта команда используется для просмотра логов указанного контейнера;
- `docker volume ls` — показывает список томов, которые являются предпочитаемым механизмом для сохранения данных, генерируемых и используемых контейнерами Docker;
- `docker rm` — удаляет один и более контейнеров;
- `docker rmi` — удаляет один и более образов;
- `docker stop` — останавливает один и более контейнеров;
- `docker exec -it ...` - выполняет команду в определенном контейнере

7. Каким образом осуществляется поиск образов контейнеров?

Сначала проверяется локальный репозиторий на наличия нужного контейнера, если он не найден локально, то поиск производится в репозитории

Docker Hub.

8. Каким образом осуществляется запуск контейнера?

Для запуска контейнера его необходимо изначально создать из образа, поэтому изначально контейнер собирается с помощью команды `docker build`, а

уже затем запускается с помощью команды `docker run`.

#### 9. Что значит управлять состоянием контейнеров?

Это означает, что в любой момент времени есть возможность запустить, остановить или выполнить команды внутри контейнера.

15

#### 10. Как изолировать контейнер?

Контейнеры уже по сути своей являются изолированными единицами, поэтому достаточно без ошибок сконфигурировать файлы `Dockerfile` и/или `docker-compose.yml`.

#### 11. Опишите последовательность создания новых образов, назначение `Dockerfile`?

Производится выбор основы для нового образа на Docker Hub, далее производится конфигурация `Dockerfile`, где описываются все необходимые пакеты, файлы, команды и т.п.

`Dockerfile` — это текстовый файл с инструкциями, необходимыми для создания образа контейнера. Эти инструкции включают идентификацию существующего образа, используемого в качестве основы, команды, выполняемые в процессе создания образа, и команду, которая будет выполняться при развертывании новых экземпляров этого образа контейнера.

#### 12. Возможно ли работать с контейнерами Docker без одноименного движка?

Да, если использовать Kubernetes

#### 13. Опишите назначение системы оркестрации контейнеров Kubernetes. Перечислите основные объекты Kubernetes.

Kubernetes — открытое программное обеспечение для автоматизации развёртывания, масштабирования контейнеризированных приложений и управления ими. Поддерживает основные технологии контейнеризации, включая Docker, rkt, также возможна поддержка технологий аппаратной

виртуализации.

— Nodes: Нода это машина в кластере Kubernetes.

— Pods: Pod это группа контейнеров с общими разделами, запускаемых как единое целое.

— Replication Controllers: replication controller гарантирует, что определенное количество «реплик» pod'ы будут запущены в любой момент времени.

— Services: Сервис в Kubernetes – это абстракция, которая определяет логический объединённый набор pod и политику доступа к ним.

— Volumes: Volume(раздел) это директория, возможно, с данными в ней, которая доступна в контейнере.

— Labels: Label'ы это пары ключ/значение которые прикрепляются к объектам, например pod'ам. Label'ы могут быть использованы для создания и выбора наборов объектов.

— Kubectl Command Line Interface: kubectl интерфейс командной строки для управления Kubernetes.