|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА − Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИППО)**

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

по дисциплине «Разработка серверных частей интернет ресурсов»

**Практическое занятие № 1**

|  |  |  |
| --- | --- | --- |
| Студент группы ИНБО-01-17 | ИКБО-20-21 Сидоров С.Д. | (подпись) |
| Преподаватель | Благирев М.М. | (подпись) |

# Теоретическое введение

Одна из главных причин, почему Docker стал популярным инструментом в backend разработке, заключается в его способности к оптимизации процесса развертывания приложений на многих серверах. Далее ниже перечислено несколько пунктов, из-за которых Docker становится все более востребованным инструментом в backend разработке:

Изолированность. Docker контейнеры создают изолированные среды, которые обеспечивает изоляцию приложения от других приложений на том же хосте. Контейнеры обычно содержать только те зависимости и пакеты, которые им необходимы для работы. Изолированные контейнеры помогают обезопасить приложение и защитить его от конфликтов, связанных с непредсказуемыми версиями библиотек и зависимостей. К тому же контейнеры гарантируют, что приложение не повлияет на другие приложения на хосте;

Переносимость. Docker контейнеры обеспечивают переносимость вашего приложения между разными серверами и окружениями. Это означает, что вы можете разработать приложение на своем локальном компьютере и затем развернуть его в любой другой среде, где установлен Docker. Контейнеры позволяют выводить приложение быстрее на рынок и сокращать время релиза;

Быстрое развертывание. С Docker можно быстро создавать и развертывать новые контейнеры с вашим приложением. Вы можете использовать готовые образы, которые содержат все необходимые зависимости и пакеты, что позволяет избежать дополнительной установки и настройки. Это позволяет разрабатывать и улучшать приложение более быстро и эффективно;

Удобная локальная разработка. Docker позволяет запускать виртуальные контейнеры на вашем локальном компьютере, что облегчает локальную разработку и отладку вашего приложения. Вы можете использовать Docker для разработки на любой платформе, в том числе Windows, Mac OS и Linux. Docker контейнеры позволяют разрабатывать приложение локально и тестировать его в клиентском окружении.

Основными компонентами докера является:

docker daemon — сердце докера. Это демон, работающий на хост-машине, и умеющий сохранять с удалённого репозитория и загружать на него образы, запускать из них контейнеры, следить за запущенными контейнерами, собирать логи и настраивать сеть между контейнерами (а с версии 0.8 и между машинами). А еще именно демон создает образы контейнеров, хоть и может показаться, что это делает docker-client.

docker — это консольная утилита для управления docker-демоном по HTTP. Она устроена очень просто и работает предельно быстро. Вопреки заблуждению, управлять демоном докера можно откуда угодно, а не только с той же машины. В сборке нового образа консольная утилита docker принимает пассивное участие: архивирует локальную папку в tar.gz и передает по сети docker-daemon, который и делает всю работу. Именно из-за передачи контекста демону по сети, лучше собирать тяжелые образы локально.

docker Hub централизованно хранит образы контейнеров. Когда вы пишете “docker run ruby”, docker скачивает самый свежий образ с ruby именно из публичного репозитория. Изначально хаба не было, его добавили уже после очевидного успеха первых двух частей.

Docker-compose — это инструмент для управления множеством контейнеров Docker или же инструмент оркистрации контейнеров, который позволяет определять и запускать несколько контейнеров Docker в единой среде. Docker-compose позволяет автоматизировать процесс создания, запуска, остановки и удаления контейнеров, упрощая процесс развертывания много контейнерных приложений.

Docker Compose позволяет определить настройки контейнера, включающие сетевую конфигурацию, зависимости, переменные среды и другие параметры. Он позволяет создавать и запускать несколько контейнеров, связанных между собой, как единое приложение. Docker Compose позволяет легко масштабировать и управлять несколькими контейнерами в единой среде, что делает его инструментом для много контейнерного развертывания.

Docker Compose используется в различных сценариях, таких как развертывание микросервисов, тестирование, разработка и многое другое. Он позволяет определять и управлять множеством контейнеров из одного файла YAML, что облегчает управление и обеспечивает более простую конфигурацию и управление контейнерами. Docker Compose также позволяет создавать собственные настраиваемые образы Docker, которые можно переносить между различными окружениями.

Таким образом, Docker Compose - это средство автоматизации в управлении множеством контейнеров Docker, которое облегчает процесс создания, запуска и управления многоконтейнерными приложениями.

Для полноценной работоспособности конфигурации нужны: операционная система, Веб-сервер, язык программирования и База данных, но база данных будет устанавливаться в следующей практике. Из всего этого следует идея технологии LAMP — акроним, обозначающий набор (комплекс) серверного программного обеспечения, широко используемый в интернете. LAMP назван по первым буквам входящих в его состав компонентов:

Linux — операционная система Linux;

Apache — веб-сервер;

MariaDB / MySQL — СУБД;

PHP — язык программирования, используемый для создания веб-приложений (помимо PHP могут подразумеваться другие языки, такие как Perl и Python).

Содержание данного набора может варьироваться в зависимости от задач и технического задания, например:

LEMP — Nginx вместо Apache (Nginx читается Engine-X)

LNMP — другой вариант названия 'Nginx вместо Apache'

LLSMP - Linux, LiteSpeed, MySQL, PHP

BAMP — BSD вместо Linux

MAMP — Mac OS X вместо Linux.

SAMP — Solaris вместо Linux

WAMP — Microsoft Windows вместо Linux

WASP — Windows, Apache, SQL Server и PHP

WIMP — Windows, IIS, MySQL и PHP

PAMP — Personal Apache MySQL PHP — набор серверов для платформы S60. Специфика платформы накладывает свой отпечаток на работу комплекса. Так, в частности, модули PHP получают и возвращают строки только в кодировке UTF-8.

FNMP — FreeBSD и Nginx вместо Linux и Apache.

XAMPP — кроссплатформенная сборка веб-сервера, X (любая из четырёх операционных систем), Apache, MySQL, PHP, Perl

Но в данной работе вам необходимо установить Apache.

Apache — это самый популярный веб-сервер, который запускается на миллионах веб-сайтов во всем мире. Apache является свободной и открытой программой, которая доступна для распространения и использования бесплатно.

Основной целью Apache является обслуживание веб-страниц пользователям, обрабатывая запросы от веб-браузеров и отображая содержимое веб-страниц в ответ на эти запросы.

В дополнение к основным функциям веб-сервера, Apache также предоставляет множество расширенных функций, приложений и библиотек, которые могут быть использованы веб-разработчиками для создания динамических веб-сайтов и приложений. Apache также поддерживает различные языки программирования, такие как PHP и Perl, которые могут быть использованы для создания динамических веб-страниц.

# Задание

Необходимо создать конфигурацию docker-compose для веб-сервера.

Docker контейнер должен включать:

* ОС Linux
* Веб-сервер Apache

Docker-compose:

* Docker-контейнер, созданный ранее
* Примонтированные тома
* Настройки портов

# Выполнение работы

Для выполнения практической работы были созданы два конфигурационных файла docker-compos.yml и Dockerfile. Содержимое файла Dockerfile представлено на листинге 1.

Листинг 1 – содержание Dockerfile

FROM php:7.4-apache

RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli

WORKDIR /var/www/html

COPY ./index.php /var/www/html

COPY ./style.css /var/www/html

EXPOSE 80

Содержимое файла docker-compose.yml представлено на листинге 2.

Листинг 2 – содержание docker-compose.yml

version: '3'

services:

   db:

      image: mysql

      environment:

         MYSQL\_USERNAME: user

         MYSQL\_PASSWORD: password

         MYSQL\_DATABASE: appDB

         MYSQL\_ROOT\_PASSWORD: tmp

      volumes:

         - ./init.sql:/docker-entrypoint-initdb.d/init.sql

   server:

      build:

         context: ./

         dockerfile: dockerfile

      ports:

         -  '8080:80'

      volumes:

         -  ./:/var/www/html

      links:

         -  db

volumes:

   db\_data:

С помощью содержимого Dockerfile устанавливается и запускается расширение для php mysqli, а также устанавливается apache, а также перебрасываются файлы index.php и style.css.

С помощью содержимого docker-compose.yml монтируется образ mysql. В данном образе задаются переменные окружения для работы с базой данных mysql. Также, пробрасывается файл init.sql в директорию /docker-entrypoint-initdb.d. Для запуска во время старта образа mysql. Создается служба server, которая содержит в себе Dockerfile, пробрасывает серверное содержимое, прикрепленное к заданию практической работы, а также обозначает порты, на которых работает данная система.

Результаты сборки и функционирования представлены на рис. 1-3.

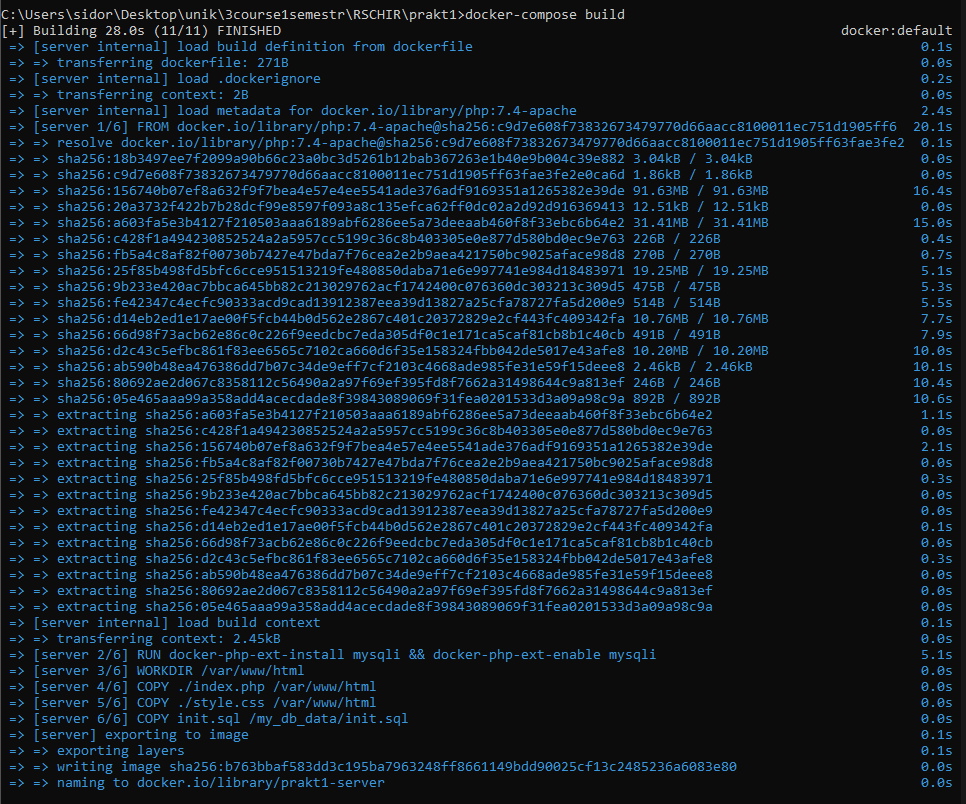


Рисунок 1 - Результат выполнения команды docker-compose build

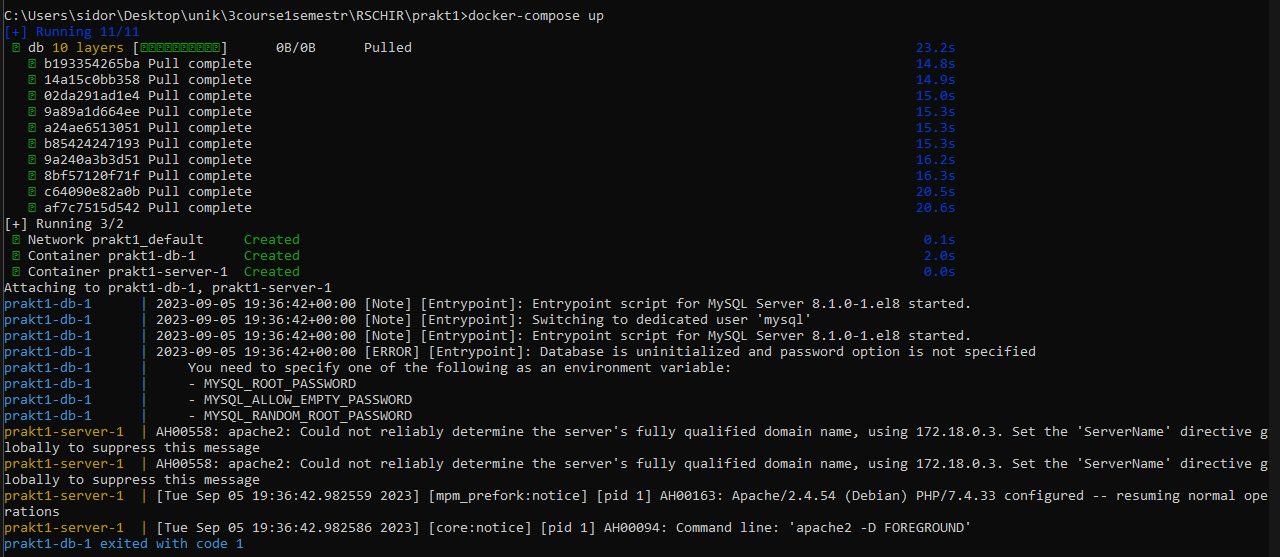


Рисунок 2 - Результат выполнения команды docker-compose up

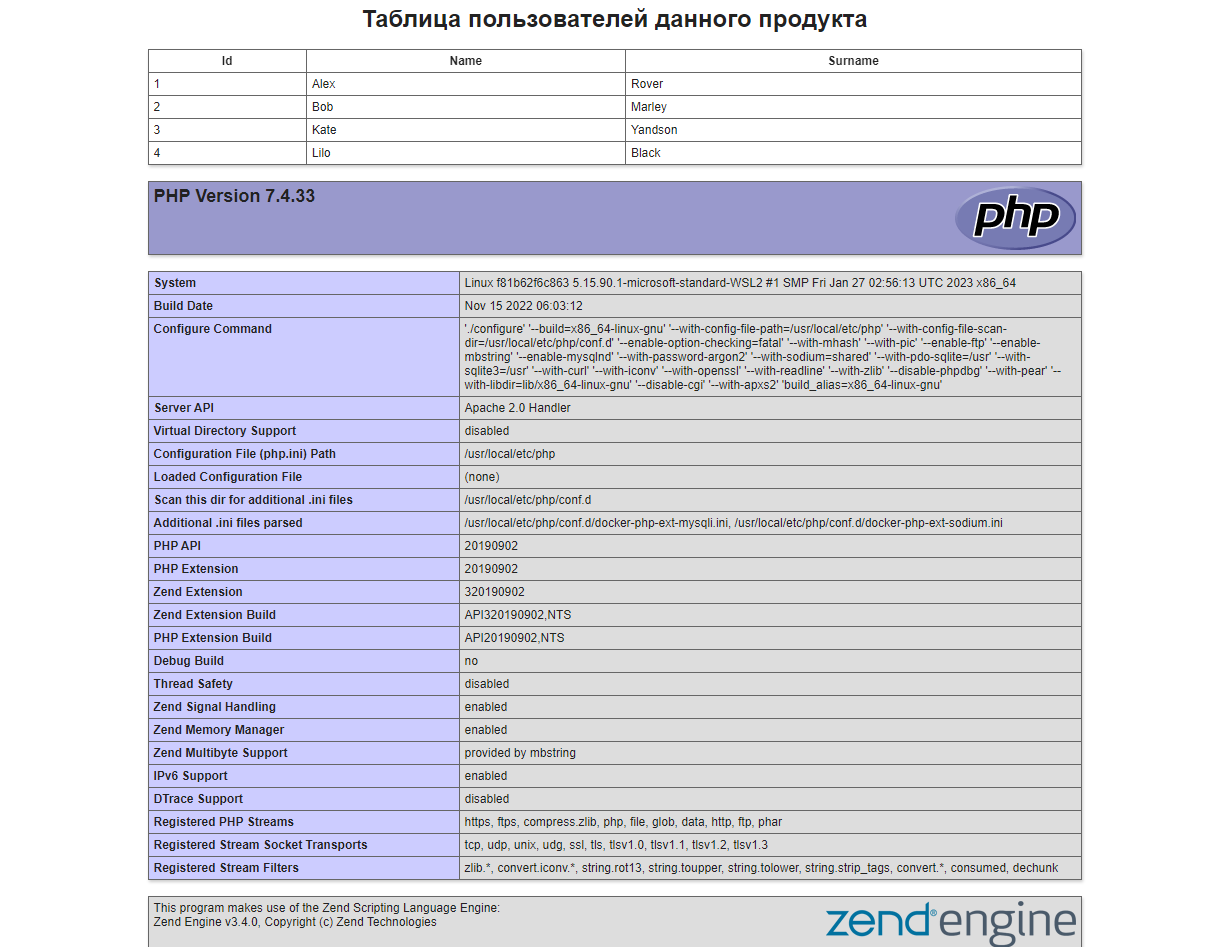


Рисунок 3 - Сгенерированная страница

**Вывод**

В ходе выполнения практической работы был создан конфигураций docker-compose для веб-сервера. Был произведен корректный запуск приложенного php скрипта генерации страницы с характеристиками веб-сервера и его работа на созданном веб-сервере.

**Ответы на вопросы**

1. Что такое apache и для чего он используется? Apache – открытое кросс-платформенное обеспечение, которое используется для размещения и поддержки веб-сервера.

2. Что такое оркестрация контейнеров и для чего это нужно? Оркестрация – программная технология, позволяющая автоматически управлять контейнерами, это нужно для разработки современных облачных приложений.

3. Что такое сервер? Сервер – устройство или программное обеспечение, способное принимать запросы и отвечать на них.

**Список использованных источников и литературы:**

1. Видео “Введение в Докер” на английском языке от создателя: [Introduction to Docker](https://www.youtube.com/watch?v=Q5POuMHxW-0) (https://www.youtube.com/watch?v=Q5POuMHxW-0)
2. Статья о назначении докера простыми словами: <https://habr.com/ru/post/309556/>
3. Более сложная и подробная статья про докер: <https://habr.com/ru/post/277699/>
4. Хорошая статья с пингвинами для прочтения после туториала по докеру: <https://habr.com/ru/post/250469/>
5. Официальная документация докера: <https://docs.docker.com/>
6. Статья о конкретном опыте использования докер контейнеров: <https://habr.com/ru/post/247969/>
7. Туториал по докеру: <https://badcode.ru/docker-tutorial-dlia-novichkov-rassmatrivaiem-docker-tak-iesli-by-on-byl-ighrovoi-pristavkoi/>
8. Туториал по докеру с Хабра: <https://habr.com/ru/post/310460/>
9. Шпаргалка с командами Docker: <https://habr.com/ru/company/flant/blog/336654/>
10. Ссылка на скачивание докера с официального сайте: <https://www.docker.com/products/docker-desktop>
11. Отличная статья про dockerfile: <https://habr.com/ru/company/ruvds/blog/439980/>
12. Установка и настройка PHP: <https://www.php.net/manual/ru/install.php>
13. Настройка среды PhpStorm и полезные фичи: <https://habr.com/ru/post/282003/>
14. Про docker compose: <https://habr.com/ru/company/ruvds/blog/450312/>
15. Docker hub: <https://hub.docker.com/>