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

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №1**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Студент группы** ИКБО-01-20 Железнов Даниил Алексеевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы**  Волков М.Ю.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Москва 2022

**ОГЛАВЛЕНИЕ**

[Цель работы 3](#_Toc113123156)

[Ход работы 3](#_Toc113123157)

[Вывод 5](#_Toc113123158)

[Ответы на вопросы к практической работе 6](#_Toc113123159)

[Ссылка на удалённый репозиторий проекта 7](#_Toc113123160)

[Список использованной литературы 8](#_Toc113123161)

# Цель работы

Научиться создавать серверное программное обеспечение. Получить навыки конфигурации Docker контейнера, PHP, базы данных

# Ход работы

Создадим два Dockerfile, для серверной части и базы данных, конфигурацию последней поместим в отдельную директорию вместе с init.sql

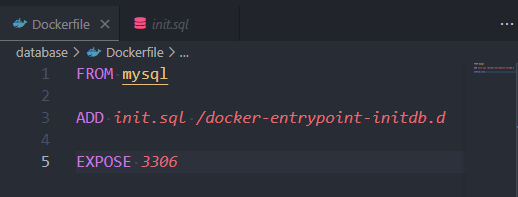


Рисунок 1 – Dockerfile для конфигурации MySQL

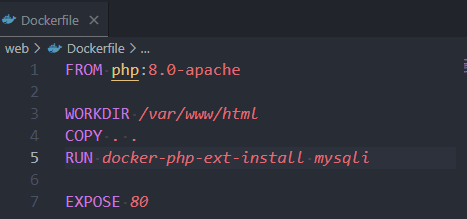


Рисунок 2 – Dockerfile с конфигурацией PHP и Apache

Далее напишем docker-compose.yaml файл, в котором укажем основную конфигурацию создаваемых контейнеров

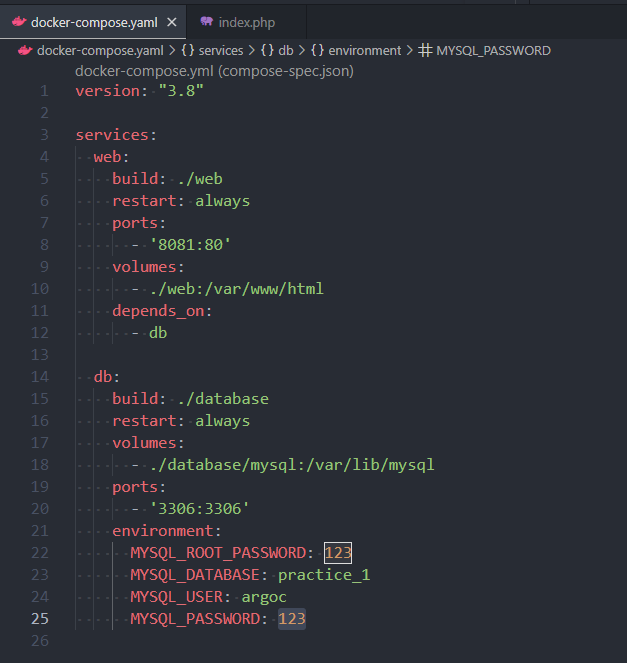


Рисунок 3 – конфигурация docker-compose

Так как мы установили порт 3306 для MySQL, изменим вызов конструктора mysqli в файле index.php, добавив в его параметры проброшенный порт следующим образом:



Рисунок 4 – изменения в index.php

Запустим веб-сервер с помощью команды docker-compose up –build и удостоверимся в его работоспособности

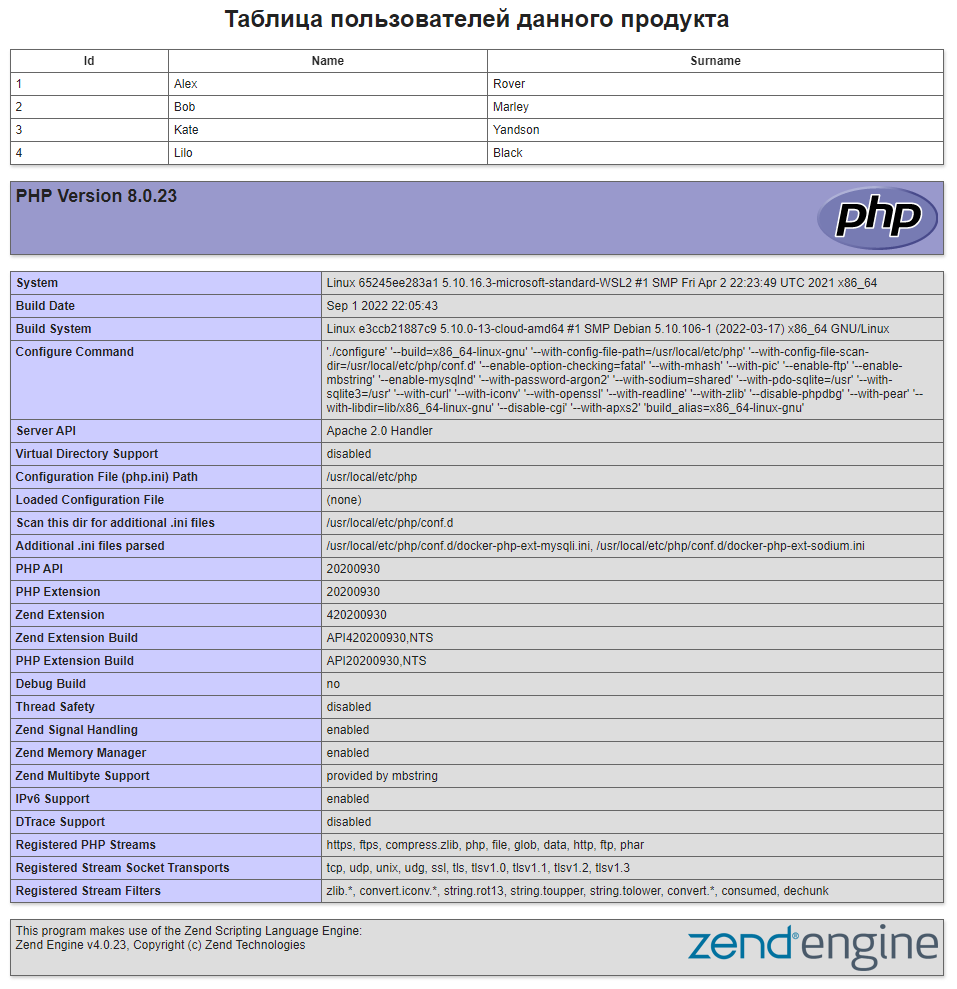


Рисунок 5 – результат работы скрипта

# Вывод

В ходе практической работы были получены навыки развёртывания PHP и Apache в Docker контейнере, а также подключения MySQL.

# Ответы на вопросы к практической работе

1. Сервер – обрабатывает данные и возвращает ответ на запрос. Клиент – запрашивающая данные сторона, предоставляющая пользователю, как правило, интерфейс
2. База данных — совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных
3. API – программный интерфейс приложения. Описание способа взаимодействия различных приложений
4. Сервис – процесс, предоставляющий услуги. Может работать на сервере и передавать данные по сети.
5. Клиент-серверная архитектура – архитектура, при которой вычислительная нагрузка распределены между устройствами клиентов и серверами
6. Виды сервисов – веб-сервисы, почтовые сервисы, файлообменники
7. Масштабируемость – распределение нагрузки на обособленные компоненты, увеличение нагрузки и производительности сервера
8. Протокол передачи данных – набор определённых правил или соглашений интерфейса логического уровня, который определяет обмен данными между различными программами
9. Тонкий клиент – большая часть вычислений выполняется на стороне сервера, от клиента требуется стабильное соединение. Толстый клиент – большая часть вычислений проходит на стороне клиента, может работать автономно
10. Паттерн MVC: общие тезисы. MVC – это способ организации кода, при котором выделяются компоненты, модель данных приложения, представление и контроллер. Эти компоненты независимы, модификации одного из них, не влияют на друга, что прощает процесс разработки приложения.
11. Паттерн MVC: Model-View-Presenter. Производный от MVC паттерн проектирования. Отличие от MVC в том, что элемент Presenter берёт на себя обработку событий пользовательского интерфейса, в то время как в MVC за это отвечает представление.
12. Паттерн MVC: Model-View-View Model. ModelView – модель представления, обёртка над моделью, преобразованная для прямого использования представлению, содержит команды, которые использует представление для работы с моделью.
13. Паттерн MVC: Model-View-Controller. Model – модель, отвечает за данные приложения, View – представление, отвечает за отображение данных пользователю, Controller – контроллер, управляет работой приложения и его жизненного цикла.
14. Docker: общие тезисы и определения. Docker – программное обеспечение для развёртывание приложений средствами контейнеризации. Базовый образ – основа приложения, состоящий из неизменяемых слоёв окружения, на основе которого собираются контейнеры.
15. Dockerfile – файл конфигурации сборки контейнера, в нём обязательно указывается базовый образ, а также может содержаться последовательность команд для создания контейнера
16. Docker Compose – инструмент для конфигурации и управлением связями между контейнерами и виртуальными томами. Конфигурация задаётся в файле docker-compose.yaml. Также представлен CLI интерфейс для развёртывания приложения
17. LAMP – часто используемый набор технологий для создания веб-сервера. Linux, Apache, MySQL/MariaDB, PHP.

# Ссылка на удалённый репозиторий проекта

Успешно выполненное задание было выложено в открытом репозитории по ссылке <https://github.com/DsArmor/development_server>

# Список использованной литературы

1. Документация языка PHP [Электронный ресурс]. – URL: <https://www.php.net/docs.php> (дата последнего обращения: 02.09.2022)
2. Medium blog. Статья «Apache Airflow in 10 minutes» [Электронный ресурс]. – URL: (дата последнего обращения: 02.09.2022)