**ЛАБОРАТОРНАЯ РАБОТА №7**

**DOCKER: ОСНОВЫ РАБОТЫ С КОНТЕЙНЕРАМИ**

**Цель работы:** освоить основные принципы работы с Docker, научиться создавать, запускать и управлять контейнерами, а также работать с Docker образами.

**ЗАДАНИЯ К ЛАБОРАТОРНОЙ РАБОТЕ**

1. ***Установка Docker***

* Установите Docker на вашу систему.
* Проверьте корректность установки командой docker --version.

Docker – это платформа для контейнеризации, которая позволяет упаковывать приложения и их зависимости в изолированные контейнеры. Чтобы начать работу с Docker, сначала нужно установить его. Установка дает возможность использовать инструменты Docker для создания, запуска и управления контейнерами.

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 1.1* ***– Проверьте корректность установки командой docker --version***

1. ***Работа с образами***

* Загрузите образ nginx из Docker Hub.
* Запустите контейнер на основе этого образа и откройте веб страницу Nginx в браузере.

Работа с образами в Docker включает скачивание готовых образов приложений из репозитория Docker Hub и их использование для запуска контейнеров. Например, вы можете использовать образ веб-сервера, такого как Nginx, чтобы быстро развернуть его и протестировать работу прямо в вашем браузере.

**Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

*Рисунок 2.1* ***– Образ nginx, загруженный в локальный репозиторий из DockerHub***

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

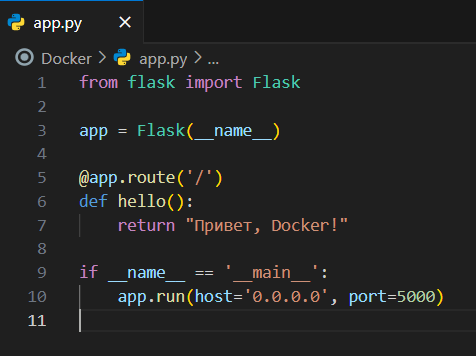
Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 2.2* ***– Запуск контейнер на основе образа и открытие веб-страницы Nginx в браузере***

1. ***Создание пользовательского образа***

* Напишите Dockerfile для создания образа с простым веб приложением на Python (например, Flask).
* Соберите образ и запустите контейнер.

Создание пользовательского образа в Docker позволяет настроить контейнер под ваши потребности, например, добавить собственное веб-приложение. Для этого нужно описать инструкции для сборки в специальном файле – Dockerfile, собрать образ и затем использовать его для запуска контейнера. Это полезно для разработки и тестирования уникальных приложений.

****

*Рисунок 3.1* ***– Веб-приложение на Python (Flask)***

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 3.2* ***– Dockerfile для создания образа веб-приложения***

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 3.3* ***– Создание образа на основе Dockerfile***

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 3.4* ***– Создание и запуск контейнера на основе образа flask\_app***

Изображение выглядит как снимок экрана, программное обеспечение, текст, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 3.5* ***– Веб-приложение из контейнера***

1. ***Работа с томами (Volumes)***

* Создайте том и подключите его к контейнеру для сохранения данных.
* Убедитесь, что данные сохраняются после перезапуска контейнера.

Тома (volumes) в Docker используются для хранения данных, которые сохраняются даже после остановки или удаления контейнера. Это позволяет обеспечить долговременное хранение данных и их доступность между перезапусками контейнеров. Подключение томов упрощает управление данными и их сохранность в процессе работы приложения.

Изображение выглядит как текст, Шрифт, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 4.1* ***– Создание тома и подключение его к контейнеру для сохранения данных***

**Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.**

*Рисунок 4.2 –* **Сохранение данных и проверка их наличия**

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 4.3 –* ***Проверка сохранения данных после перезапуска контейнера***

1. ***Сеть в Docker***

* Создайте два контейнера (например, веб-сервер и базу данных).
* Настройте сеть для взаимодействия между ними.

Docker позволяет контейнерам взаимодействовать друг с другом через сеть. Взаимодействие контейнеров можно настроить, например, для создания общей сети между веб-сервером и базой данных. Это важно для построения многокомпонентных приложений, где каждая часть (контейнер) выполняет свои задачи и обменивается данными с другими компонентами.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.1 –* ***Проверка сохранения данных после перезапуска контейнера***

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.2 –* ***Работа с БД***

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.3 –* ***Работа с БД***

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.4 –* ***Проверка наличия сети и подключение к ней контейнеров***

Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.5 –* ***Проверка подключения контейнеров к сети***

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.6 –* ***Новый код для взаимодействия с БД***

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.7 –* ***Новый код для взаимодействия с БД***

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.8 –* ***Изменение Dockerfile, перезапись образа и пересоздание контейнера***

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 5.9 –* ***Веб-страница на основе контейнеров***

1. ***Docker Compose***

* Напишите файл docker-compose.yml для развёртывания многоконтейнерного приложения (например, веб-сервер + БД).
* Запустите приложение с помощью Docker Compose.

Docker Compose – это инструмент для упрощённого управления многоконтейнерными приложениями. Используя файл docker-compose.yml, можно описать конфигурацию всех необходимых сервисов (например, веб-сервер и базу данных) и запустить их одной командой. Это делает развертывание сложных приложений удобным и быстрым.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 6.1 –* ***Файл docker-compose.yml***

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

*Рисунок 6.2 –* ***Запуск приложение с помощью Docker Compose***

1. ***Очистка ресурсов***

* Удалите все созданные контейнеры, образы и тома.

Очистка ресурсов в Docker включает удаление всех созданных контейнеров, образов и томов, которые больше не нужны. Это позволяет освободить место и поддерживать систему в чистоте. Регулярная очистка помогает избежать накопления ненужных ресурсов и упрощает управление Docker-средой.

**Использовался следующий список:**

***Команда для удаления контейнеров:***

docker rm -f $(docker ps -aq)

***Команда для удаления образов:***

docker rmi -f $(docker images -q)

***Команда для удаления томов:***

docker volume rm $(docker volume ls -q)

***Команда для удаления сетей:***

docker network rm my\_network