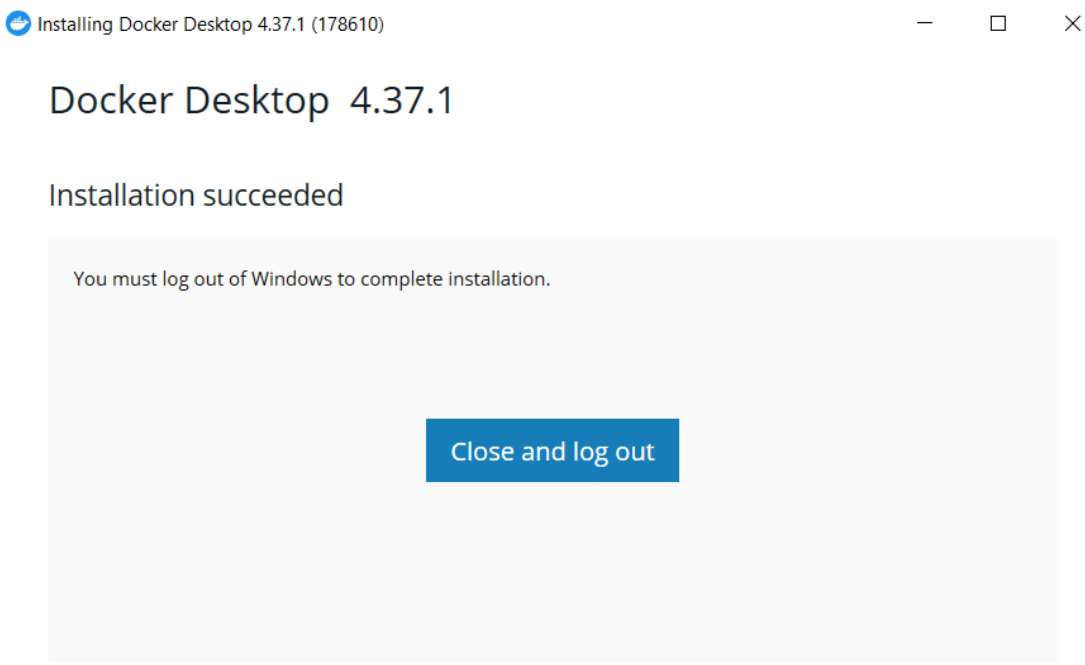
Homework_Lesson22_Docker_3

Цель: получить практический опыт написания Dockerfile, развертывания приложений с использованием Docker-compose/
Задание 1: Создание Dockerfile для приложения веб-сервера. Вам необходимо написать Dockerfile для создания контейнера с приложением веб-сервера на основе образа Ubuntu 20.04. Приложение должно быть запущено на порту 8080 и должно отдавать статические файлы из каталога /app/static.
Шаги, которые необходимо выполнить:
1. Создайте новый файл Dockerfile в пустой директории на вашем локальном компьютере.
2. Напишите инструкцию FROM, которая указывает базовый образ Ubuntu 20.04.
3. Установите необходимые зависимости с помощью инструкции RUN. Установите пакеты nginx и curl, а также создайте каталог /app/static.
4. Скопируйте файл конфигурации nginx из вашего локального каталога внутрь контейнера с помощью инструкции COPY.
5. Скопируйте статические файлы из каталога /app/static на вашем локальном компьютере внутрь контейнера с помощью инструкции COPY.
6. Используйте инструкцию EXPOSE для открытия порта 8080.
7. Используйте инструкцию CMD для запуска команды nginx с указаниемпути к файлу конфигурации, который вы скопировали на шаге 4.
8. Сохраните файл Dockerfile и соберите образ с помощью команды dockerbuild.
9. Запустите контейнер из образа с помощью команды docker run и проверьте, что веб-сервер отдает статические файлы из каталога /app/static на порту 8080.
Задание 2 – развертывание приложения с помощью Docker-compose
Шаги, которые необходимо выполнить:
1. Создайте новый файл docker-compose.yml в пустой директории на вашем локальном компьютере.
2. Напишите инструкцию version в версии 3.
3. Определите сервис для базы данных PostgreSQL. Назовите его "db". Используйте образ postgres:latest, задайте переменные окружения POSTGRES_USER, POSTGRES_PASSWORD и POSTGRES_DB для установки пользовательского имени, пароля и имени базы данных соответственно.
4. Определите сервис для веб-сервера на основе образа NGINX. Назовите его "web". Используйте образ nginx:latest. Определите порт, на котором должен работать сервер, с помощью инструкции ports. Задайте путь к файлам конфигурации NGINX внутри контейнера, используя инструкцию volumes.
5.* Определите ссылку на сервис базы данных в сервисе веб-сервера. Используйте инструкцию links.
6. Сохраните файл docker-compose.yml и запустите приложение с помощью команды docker-compose up.
7. Проверьте, что приложение работает, перейдя в браузере на localhost:80.

Выполнение первого задания:

Установка Docker Desktop и интеграция на WSL.

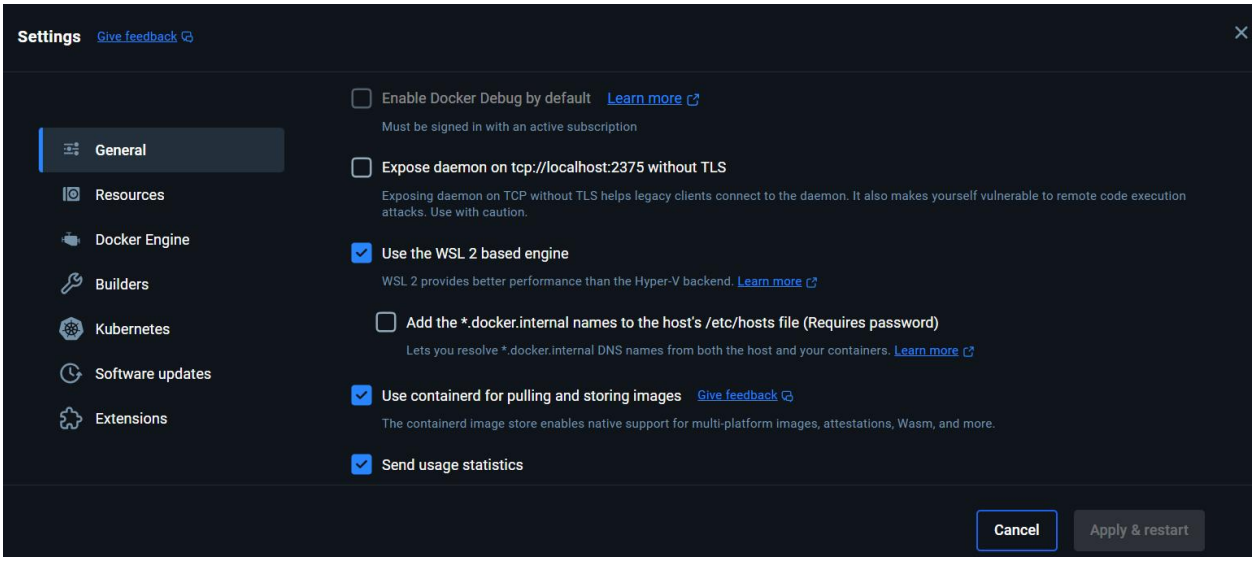
1) Скачиваем Docker Desktop и следуюм инструкциям по установке https://docs.docker.com/desktop/features/wsl/#download.
После успешной установки выдаст окно о завершении и попросит перезагрузить Windows:



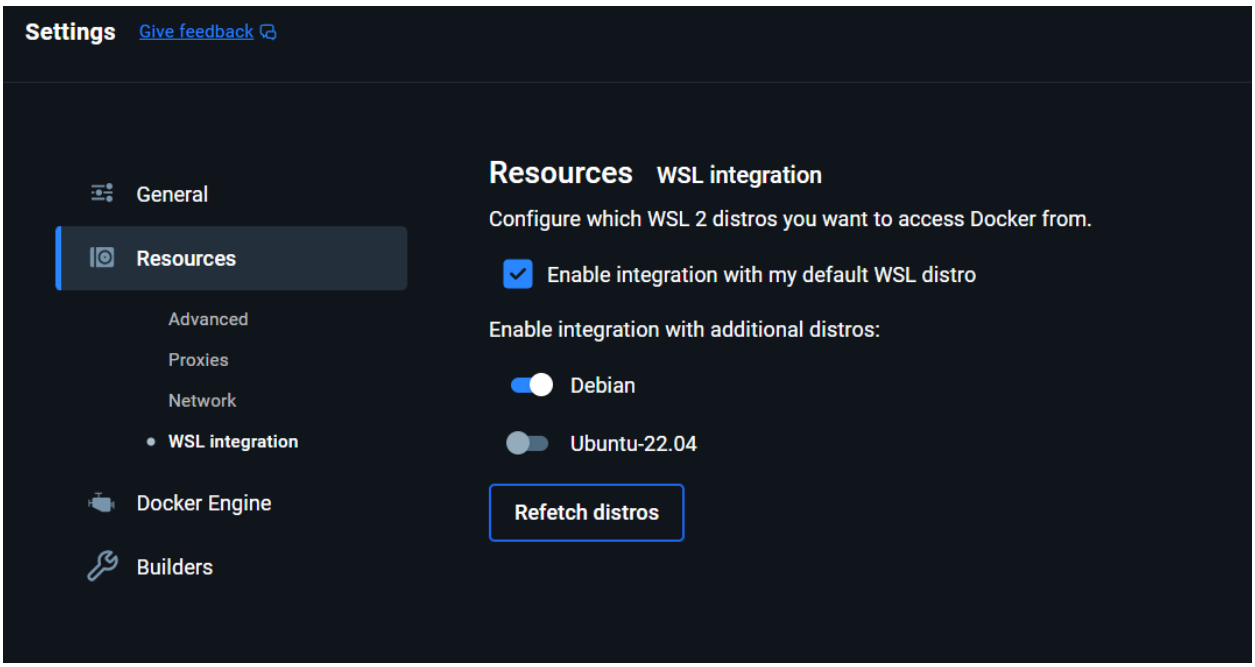
Успешная установка Docker Desktop

2) После перезагрузки Windows, нужно запустить Docker Desktop и перейти в настройки:

3) Поставить флажок “Use the WSL2 based engine”:



4) Выбрать из установленных дистрибутивов WSL2, которые необходимо включить интеграцию Docker, перейдя в раздел Resources -> WSL integretion:



5) После, нужно убедиться, что Docker установлен. Для этого открываем в терминале дистрибутив WSL и вводим следующие команды:

$ docker --version

6) Проверим правильность работы установки, выполнив встроенный образ Docker:

$ docker run hello-world

```
makarov@DESKTOP-UG6J7T7:~$ docker --version
Docker version 27.5.1, build 9f9e405
makarov@DESKTOP-UG6J7T7:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Download complete
Digest: sha256:d715f14f9eca81473d9112df50457893aa4d099adeb4729f679006bf5ea12407
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

makarov@DESKTOP-UG6J7T7:~$
```

$ sudo apt install curl software-properties-common ca-certificates apt-transport-https -y \\ установка 4 необходимых пакетов для Docker
$ sudo curl -f -s -S -L https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add - \\ импортирование GPG ключей для верификации подписей ПО
$ add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb_release -cs) stable" \\ добавление репозитория Docker для Debian
$ sudo apt-get update -y \\обновление индексов пакетов

$ sudo apt-get install docker-ce docker-ce-cli -y \\ установка Docker