Міністерство освіти і науки України

Національний технічний університет України  
"Київський політехнічний інститут імені Ігоря Сікорського"

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

Лабораторна робота №9.2 з дисципліни «Операційна система UNIX»

Створення контейнерів Python Django

Група ТР-25

Колісніченко Анни

Перевірила д.т.н., проф. Левченко Л.О.

КИЇВ 2023

**Мета роботи**: – ознайомитися та набути навичок:

- написання скрипта Dockerfile для створення контейнеру;

- встановлення Docker Compose в Ubuntu 22.04;

- створення контейнерів (сервісів): Django+PostgreSQL для розроблення додатку.

**Теоретична частина**

Для створення контейнерів необхідно вміти працювати з **Docker Compose** та **Dockerfile**.

**Dockerfile та синтаксис для їх створення**

**Dockerfile** - скрипт, який дозволяє автоматизувати процес побудови контейнерів шляхом виконання відповідних команд (дій) в base образі для формування нового образу.

Усі подібні файли починаються з позначення **FROM** так як і процес побудови нового контейнера, далі йдуть різні методи, команди, аргументи або умови, після застосування яких створиться Docker контейнер.

Розглянемо синтаксис Dockerfile. В Докер файлах міститься два типи основних блоків - коментарі та команди з аргументами. Причому для всіх команд передбачається певний порядок.

Нижче наведено типовий приклад синтаксису, де перший рядок є коментарем, а другий - командою.

*# Print «Hello from User!»*

*RUN echo «Hello from User!!»*

**Завдання**

1. Описати хід виконання поставлених завдань, надаючи покроковий знімок екрану (screenshot).
2. Висновки по роботі.

**Хід роботи**

1. Встановимо останню версію Docker Compose для управління додатками з декількома контейнерами.
2. Перевіряємо поточну версію, за необхідності оновимо її за допомогою наступної команди:

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

1. Встановимо дозвіл (+х – виконання /rwx), тобто зробимо файл виконуваним:



1. Перевіримо чи установка пройшла успішно за допомогою перевірки версії:



1. Запуск контейнера за допомогою Docker Compose
2. Створимо директорію для файлу YAML і перейдемо в неї:

Зображення, що містить текст, Шрифт, знімок екрана, Графіка

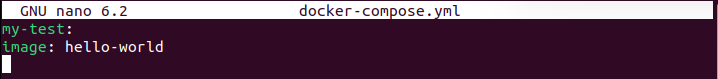
Автоматично згенерований опис

1. Створимо в цій директорії файл YAML:

Зображення, що містить текст, знімок екрана, монітор, програмне забезпечення

Автоматично згенерований опис

1. Помістимо у файл docker-compose.yml наступні дані, збережемо його і закриємо текстовий редактор:



Перший рядок файлу YAML використовується в якості частини імені контейнера.

Другий рядок вказує, який образ використовується для створення контейнера.

1. Запускаємо образ hello-world командою docker-compose up (це аналог команди docker run) вона буде шукати локальний за вказаним іменем. Після цього можна зберегти і закрити файл.

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

1. Ми можемо вручну переглянути образи в нашій системі за допомогою команди docker images:

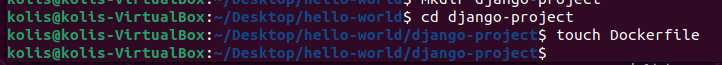
Зображення, що містить текст, знімок екрана, Шрифт, ряд

Автоматично згенерований опис

1. Починаємо виконувати завдання.
2. Створимо порожній каталог для проекту:



1. Створимо новий файл під назвою у своєму проекті:



1. Додамо наступний вміст у файл Dockerfile:

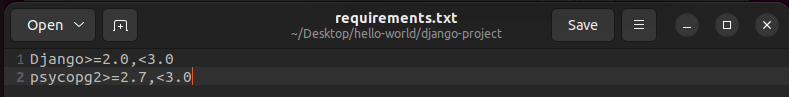
Зображення, що містить текст, знімок екрана, Мультимедійне програмне забезпечення, Шрифт

Автоматично згенерований опис

1. Створимо файл вимог requirements.txt у нашому каталозі проекту.



1. Додамо у файл.



1. Створимо файл з назвою docker-compose.yml у нашому каталозі проекту



Файл Docker-compose.yml описує сервіси, які створюють наш додаток.

1. Додамо у файл docker-compose.yml наступну конфігурацію.

Зображення, що містить текст, знімок екрана, програмне забезпечення, Мультимедійне програмне забезпечення

Автоматично згенерований опис

Цей файл визначає два сервіси: db-сервіс та веб-сервіс.

1. Створимо проект для запуску Django
2. Створимо проект Django, виконуючи команду run для docker-compose наступним чином.

Зображення, що містить текст, знімок екрана

Автоматично згенерований опис

Команда вказує Compose запустити django-admin startproject composeexample в контейнері, використовуючи образ та конфігурацію веб-сервіса. Оскільки веб-образу поки ще не існує, Compose створює його з поточного каталогу, як вказано у збірці: рядок в docker-compose.yml.

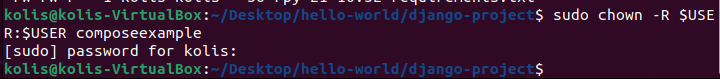
Після того, як побудовано образ веб-сервіса, Compose запускає його та виконує команду django-admin startproject у контейнері. Ця команда вказує Django створити набір файлів і каталогів, що представляють проект Django.

1. Після завершення команди Docker-Compose, складемо список вмісту нашого проекту.

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

1. Оскільки Linux при роботі з Docker створені файли django-admin належать користувачу root (це відбувається тому, що контейнер працює від імені користувача root), тому треба змінити власника на нові файли:



1. Підключимо базу даних PostgreSQL
2. Відредагуємо файл composeexample/settings.py у своєму каталозі проектів. Замінимо DATABASES = ... на наступне:

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

1. Запускаємо проект. Спочатку створюється контейнер бази даних, від якого залежить web проект. Після бази даних створюється та запускається сервер Django.

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

1. Перевіряємо роботу серверу. Сайт локального серверу успішно відкривається.

Зображення, що містить текст, знімок екрана, програмне забезпечення, Веб-сторінка

Автоматично згенерований опис

**Висновок:** Під час виконання лабораторної роботи №9.2 я набула навичок написання скрипта Dockerfile для створення контейнеру, встановлення Docker Compose в Ubuntu 22.04 та створення контейнерів (сервісів): Django+PostgreSQL для розроблення додатку.