

Національний Університет Біоресурсів і Природокористування України
Факультет інформаційних технологій

Програмування Python

Лабораторна робота №2

Виконав

Студент групи ІПЗ-23008бск

Постумент Денис

Київ 2025

Тема: Технології контейнеризації Python-додатків: основи роботи з Docker

Мета роботи: ознайомитися з технологіями контейнеризації Python-додатків на базі Docker; закріпити навички створення та використання віртуальних оточень, розробки Python-програм з використанням зовнішніх бібліотек, а також навчитися адаптувати та розгортати програми у контейнерному середовищі для забезпечення портативності та відтворюваності.

Текст завдання:

1. Створити **віртуальне оточення** (ім'я - прізвище студента) з **Python** версії не нижче **3.13.1**.

2. В цьому оточенні створити програму на мові Python для перекладу тексту за допомогою

модуля **googletrans** (<https://pypi.org/project/googletrans/>)

В програмі реалізувати три функції: **Translate(str, lang)**, **LangDetect(txt)**, **CodeLang(lang)**.

2.1. **Translate(str, lang)** - **функція** для перекладу тексту. Тут **str** – текстовий рядок, який необхідно перевести, **lang** – текстовий параметр, який визначає мову, на яку необхідно перекласти текст **str**.

Параметр **lang** може бути або назвою мови (формат: English або english), або ISO-639 code мови (формат: en або En або EN) (дивись таблицю мов <https://cloud.google.com/translate/docs/languages>). Функція повинна повернати текстовий рядок – переклад на вибрану мову, або повідомлення про помилку.

2.2. **LangDetect(txt)** - функція, яка повертає мову тексту **txt** і його **confidence**.

2.3. **CodeLang(lang)** - функція, яка повертає код мови (відповідно до **таблиці**), якщо в параметрі **lang** міститься назва мови, або повертає назву мови, якщо в параметрі **lang** міститься її код.

2.4. Використав створені функції, написати програму для перекладу тексту на задану користувачем мову.

3. На платформі Docker створити контейнер (ім'я - прізвище студента та ініціали) з наступним складом: **ОС Linux, Python** версії **3.12** або нижче.

4. В корні файлової системи контейнера створити папку (ім'я - прізвище студента)

5. Переписати програму із пункту 2

використав **модуль googletrans==3.1.0a0** і скопіювати в контейнер в

створену папку. (Функціонал і структура програми як описано в пункті 2)

6. Запустити програму із контейнера.

7. Завантажити проект на GitHub, попередньо додавши до файлу .gitignore всі технічні папки та файли.

Хід роботи

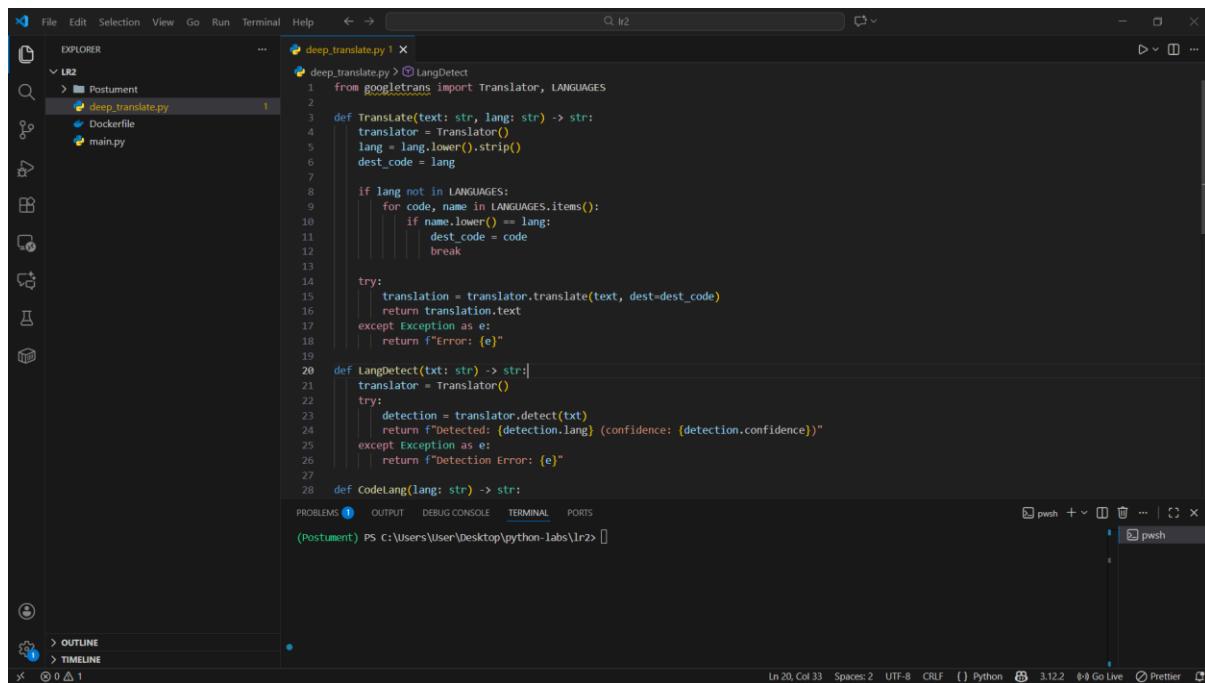


Рис 1.1 – Скріншот середовища VS Code з кодом програми

Повний код програми з пункту 2:

```
from googletrans import Translator, LANGUAGES
```

```
def Translate(text: str, lang: str) -> str:
```

```
    translator = Translator()
```

```
    lang = lang.lower().strip()
```

```
    dest_code = lang
```

```
    if lang not in LANGUAGES:
```

```
for code, name in LANGUAGES.items():

    if name.lower() == lang:

        dest_code = code

        break

try:

    translation = translator.translate(text, dest=dest_code)

    return translation.text

except Exception as e:

    return f"Error: {e}"
```

```
def LangDetect(txt: str) -> str:

    translator = Translator()

    try:

        detection = translator.detect(txt)

        return f"Detected: {detection.lang} (confidence: {detection.confidence})"

    except Exception as e:

        return f"Detection Error: {e}"
```

```
def CodeLang(lang: str) -> str:

    lang = lang.lower().strip()

    if lang in LANGUAGES:

        return LANGUAGES[lang].capitalize()
```

```
for code, name in LANGUAGES.items():

    if name.lower() == lang:

        return code

    return "Unknown Language"

if __name__ == "__main__":

    print("--- ЛОКАЛЬНЫЙ ЗАПУСК (Latest Version) ---")

    try:

        user_text = input("Введіть текст: ").strip()

        user_lang = input("Введіть мову: ").strip()

        print(f"\n[Local] Визначено мову: {LangDetect(user_text)}")

        print(f"[Local] Переклад: {TransLate(user_text, user_lang)}")

        print(f"[Local] Код мови: {CodeLang(user_lang)}")

    except Exception as e:

        print(f"Помилка: {e}")
```

The screenshot shows the Visual Studio Code interface with the Python extension installed. The Explorer sidebar shows a project named 'Postument' containing files: 'deep_translate.py', 'Dockerfile', and 'main.py'. The 'main.py' file is open in the editor, displaying code for deep language detection and translation. The 'TERMINAL' tab shows the output of running Python and pip commands. The status bar at the bottom indicates the file is 3.12.2 and the code is in Python.

```
File Edit Selection View Go Run Terminal Help < > Q lr2
EXPLORER Postument ...
main.py 1 deep_translate.py 1 Dockerfile
44     print(f"\n[Local] Визначено мову: {LangDetect(user_text)}")
45     print(f"[Local] Переклад: {Translate(user_text, user_lang)}")
46     print(f"[Local] Код мови: {codeLang(user_lang)}")
47
48 except Exception as e:
49     print(f"Помилка: {e}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• (Postument) PS C:\Users\User\Desktop\python-labs\lr> python --version
Python 3.12.2
• (Postument) PS C:\Users\User\Desktop\python-labs\lr> pip list
Package Version
certifi 2025.11.12
chardet 3.0.4
googletrans 4.0.0rc1
h11 0.9.0
h2 3.2.0
hpack 3.0.0
httplib2 2025.1.1
httpcore 0.9.1
httpx 0.13.3
hyperframe 5.2.0
idna 2.10
pip 25.3
rfc3986 1.5.0
sniffio 1.3.1
o (Postument) PS C:\Users\User\Desktop\python-labs\lr>

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF () Python 3.12.2 ⓘ Go Live ⚙ Prettier ⌂
```

Рис. 2 – Скріншот з версією Python та списком встановлених пакетів

The screenshot shows the Visual Studio Code interface after running the program. The terminal output shows the execution of 'python deep_translate.py', followed by user input 'Введіть текст: Привіт! Слава Україні!' and 'Введіть мову: japanese'. The program then outputs the detected language and the translated text. The status bar at the bottom indicates the file is 3.12.2 and the code is in Python.

```
File Edit Selection View Go Run Terminal Help < > Q lr2
EXPLORER Postument ...
deep_translate.py 1
deep_translate.py > LangDetect
1 from googletrans import Translator, LANGUAGES
2
3 def Translate(text: str, lang: str) -> str:
4     translator = Translator()
5     lang = lang.lower().strip()
6     dest_code = lang
7
8     if lang not in LANGUAGES:
9         for code, name in LANGUAGES.items():
10             if name.lower() == lang:
11                 dest_code = code
12                 break
13
14     try:
15         translation = translator.translate(text, dest=dest_code)
16     return translation.text

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
• (Postument) PS C:\Users\User\Desktop\python-labs\lr> python deep_translate.py
--- ЛОКАЛЬНИЙ ЗАПУСК (Latest Version) ---
Введіть текст: Привіт! Слава Україні!
Введіть мову: japanese

[local] Визначено мову: Detected: uk (confidence: None)
[local] Переклад: こんにちは！ ウクライナに榮光あれ！
[local] Код мови: ja
• (Postument) PS C:\Users\User\Desktop\python-labs\lr> python deep_translate.py
--- МОДУЛЬНИЙ ЗАПУСК (Latest Version) ---
Введіть текст: Привіт! Слава Україні!
Введіть мову: en

[local] Визначено мову: Detected: uk (confidence: None)
[local] Переклад: Greetings! Glory to Ukraine!
[local] Код мови: English
o (Postument) PS C:\Users\User\Desktop\python-labs\lr>

Ln 20, Col 33 Spaces: 2 UTF-8 CRLF () Python 3.12.2 ⓘ Go Live ⚙ Prettier ⌂
```

Рис. 3 – Результат виконання програми з пункту 2

Повний код програми з пункту 5:

```
from googletrans import Translator, LANGUAGES
```

```
try:
```

```
    from googletrans import LANGCODES
```

```
except ImportError:
```

```
    LANGCODES = {v.lower(): k for k, v in LANGUAGES.items()}
```

```
def Translate(text: str, lang: str) -> str:
```

```
    translator = Translator()
```

```
    lang = lang.lower().strip()
```

```
    dest_code = None
```

```
    if lang in LANGUAGES:
```

```
        dest_code = lang
```

```
    elif lang in LANGCODES:
```

```
        dest_code = LANGCODES[lang]
```

```
    else:
```

```
        for code, name in LANGUAGES.items():
```

```
            if name.lower() == lang:
```

```
                dest_code = code
```

```
                break
```

```
    if dest_code is None:
```

```
        dest_code = lang
```

```
try:  
    translation = translator.translate(text, dest=dest_code)  
    return translation.text  
except Exception as e:  
    return f"Docker Error: {e} (Target code was: {dest_code})"
```

```
def LangDetect(txt: str) -> str:  
    translator = Translator()  
    try:  
        detection = translator.detect(txt)  
        return f"Detected(lang={detection.lang},  
confidence={detection.confidence})"  
    except Exception as e:  
        return f"Error: {e}"
```

```
def CodeLang(lang: str) -> str:  
    lang = lang.lower().strip()  
    if lang in LANGUAGES:  
        return LANGUAGES[lang].capitalize()  
    try:  
        from googletrans import LANGCODES  
        if lang in LANGCODES:  
            return LANGCODES[lang]
```

```
except:  
    pass  
  
for code, name in LANGUAGES.items():  
    if name.lower() == lang:  
        return code  
    return "Not Found"  
  
if __name__ == "__main__":  
    print("--- ЗАПУСК В КОНТЕЙНЕРІ (v3.1.0a0) ---")  
  
    try:  
        user_text = input("Docker Input Text: ").strip()  
        user_lang = input("Docker Input Lang: ").strip()  
  
        print(f"\n[Docker] Detected: {LangDetect(user_text)}")  
        print(f"[Docker] Translated: {TransLate(user_text, user_lang)}")  
  
    except EOFError:  
        print("Помилка: Запустіть контейнер з пропорцем -it")  
    except Exception as e:  
        print(f"System Error: {e}")
```

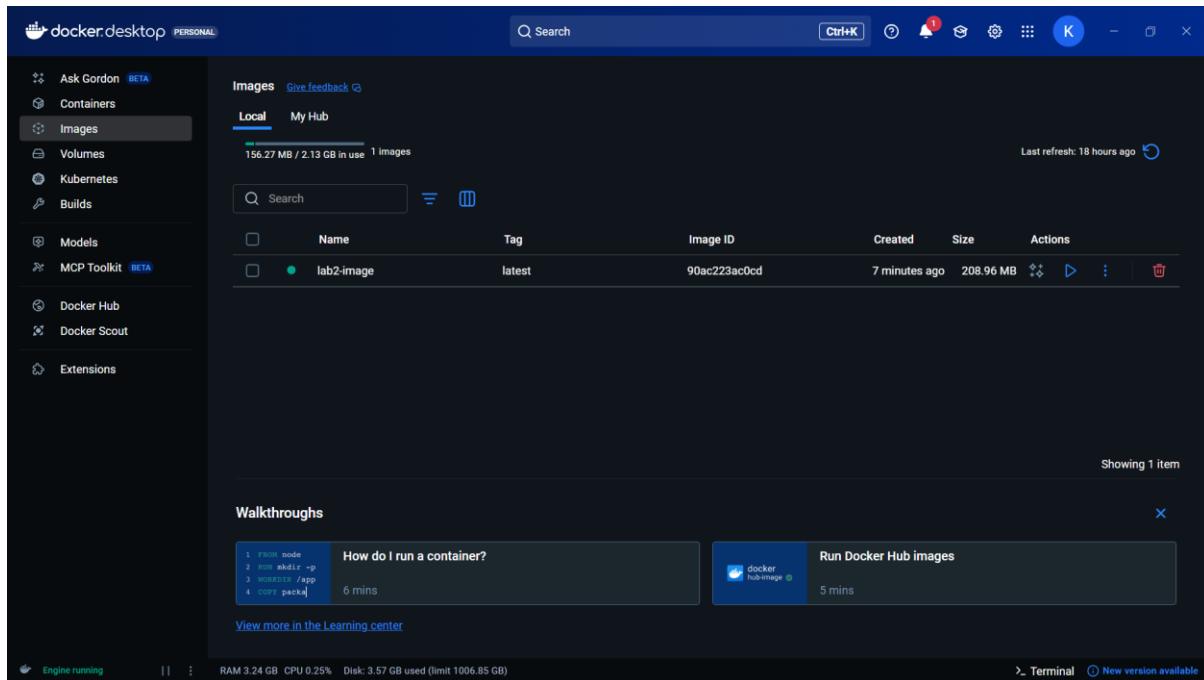


Рис. 4 – Створений docker образ

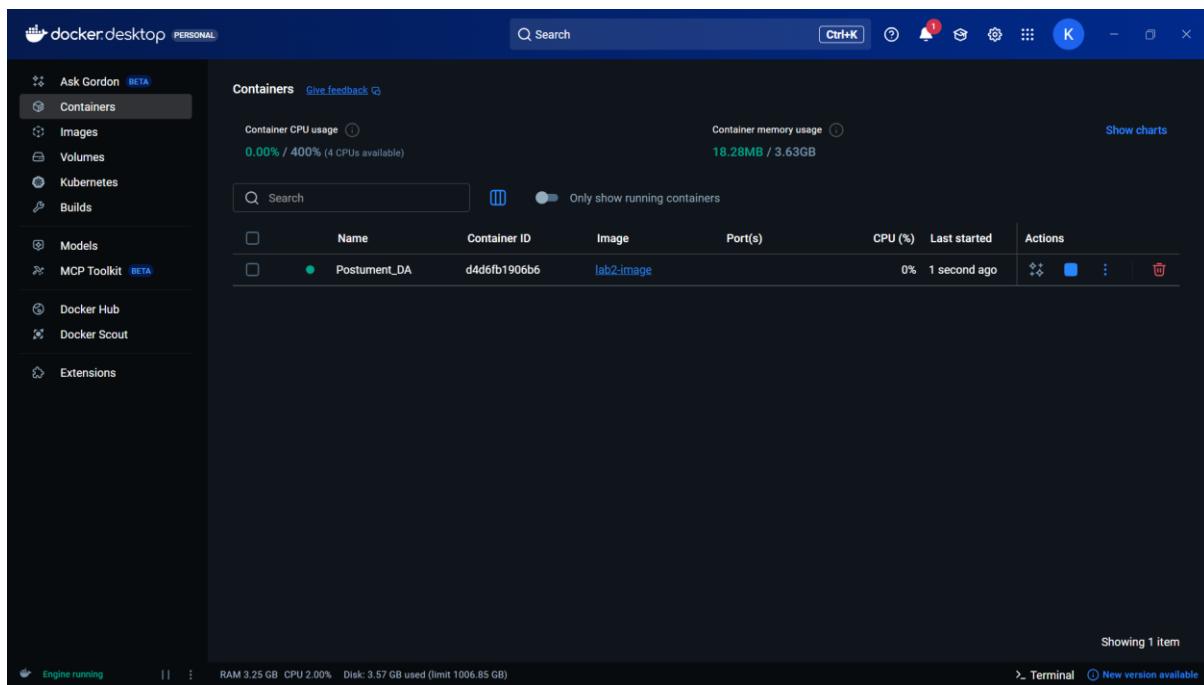


Рис 5 – Створений docker контейнер

The screenshot shows the Visual Studio Code interface with the Docker extension. The Explorer sidebar on the left shows a project named 'LR2' containing files 'Postument', 'deep_translate.py', 'docker_translate.py', and 'Dockerfile'. The main area displays the contents of the 'Dockerfile'. Below the code editor is a terminal window titled 'psh' showing the following command-line session:

```
(Postument) PS C:\Users\User\Desktop\python-labs\lr> docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
lab2-image latest 90ac223ac0cd 15 minutes ago 209MB
● (Postument) PS C:\Users\User\Desktop\python-labs\lr> docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
d4defbf19e6b6 lab2-image "python docker_trans..." 11 minutes ago Exited (137) 2 minutes ago Postument_DA
```

Рис. 6 – Інформація про образ та контейнер в терміналі

This screenshot is similar to Figure 6, showing the same VS Code environment. The terminal window is now titled 'docker' and displays the output of running a Docker container:

```
(Postument) PS C:\Users\User\Desktop\python-labs\lr> docker run -it --rm lab2-image /bin/bash
root@8d2c474f590b:/Postument# cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 13 (trixie)"
NAME="Debian GNU/Linux"
VERSION_ID="13"
VERSION="13 (trixie)"
VERSION_CODENAME="trixie"
DEBIAN_VERSION_FULL=13.2
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
root@8d2c474f590b:/Postument# python --version
Python 3.12.12
root@8d2c474f590b:/Postument#
```

Рис. 7 – Інформація про версію Python та Linux в запущеному контейнері

The screenshot shows the Visual Studio Code interface with the Docker extension. In the Explorer sidebar, there is a folder named 'LR2' containing files 'Postument', 'deep_translate.py', 'docker_translate.py', and 'Dockerfile'. The 'Dockerfile' tab is selected in the top navigation bar. The terminal tab shows the command 'pip list' output:

```
root@8d2c474f590b:/Postument# pip list
Package           Version
-----
certifi          2025.11.12
chardet          3.0.4
googletrans      3.1.0@0
h11              0.9.0
h2              3.2.0
hpack             3.0.0
hstspreload     2025.1.1
httpcore         0.9.1
httpx             0.13.3
hyperframe       5.2.0
idna              2.10
pip              25.0.1
rfc3986          1.5.0
sniffio          1.3.1
root@8d2c474f590b:/Postument#
```

Рис. 8 – Список всіх встановлених пакетів в контейнері

The screenshot shows the Visual Studio Code interface with the Docker extension. In the Explorer sidebar, there is a folder named 'LR2' containing files 'Postument', 'deep_translate.py', 'docker_translate.py', and 'Dockerfile'. The 'Dockerfile' tab is selected in the top navigation bar. The terminal tab shows the command 'ls -la' output:

```
root@8d2c474f590b:/Postument# pwd
/Postument
root@8d2c474f590b:/Postument# ls -la
total 12
drwxr-xr-x 1 root root 4896 Nov 29 13:14 .
drwxr-xr-x 1 root root 4896 Nov 29 13:14 ..
-rw-r--r-- 1 root root 2182 Nov 29 13:14 docker_translate.py
root@8d2c474f590b:/Postument# cd ..
root@8d2c474f590b:# pwd
/
root@8d2c474f590b:# ls
Postument bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@8d2c474f590b:#
```

Рис. 9 – Файлова структура контейнеру

The screenshot shows the Visual Studio Code interface. In the Explorer sidebar, there is a folder named 'LR2' containing files 'Postument', 'deep_translate.py', 'docker_translate.py', and 'Dockerfile'. The 'Dockerfile' tab is selected, displaying the following code:

```
FROM python:3.12-slim (last pushed 1 day ago)
WORKDIR /Postument
COPY docker_translate.py .
RUN pip install googletrans==3.1.0a0
CMD ["python", "docker_translate.py"]
```

In the bottom right corner of the terminal window, there is a 'push' button.

Рис. 10 – Виконання програми з пункту 5 в контейнері

Посилання на гітхаб: <https://github.com/DenysSheppard/python-labs/tree/master/lr2>

Висновки

У ході роботи було досягнуто мети ознайомлення та практичного застосування технологій контейнеризації Python-додатків на базі Docker. Успішно створено та протестовано Python-програму для перекладу тексту, яка використовує модуль googletrans, спочатку у віртуальному оточенні, а потім адаптовано та розгорнуто у Docker-контейнері. Це підтвердило ключову перевагу контейнеризації: забезпечення портативності, відтворюваності та ізоляції програмного середовища незалежно від операційної системи хоста, що є критично важливим для сучасної мікросервісної розробки.