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 пере**клас**ти, ***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 повертати 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**, **Pyton** 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

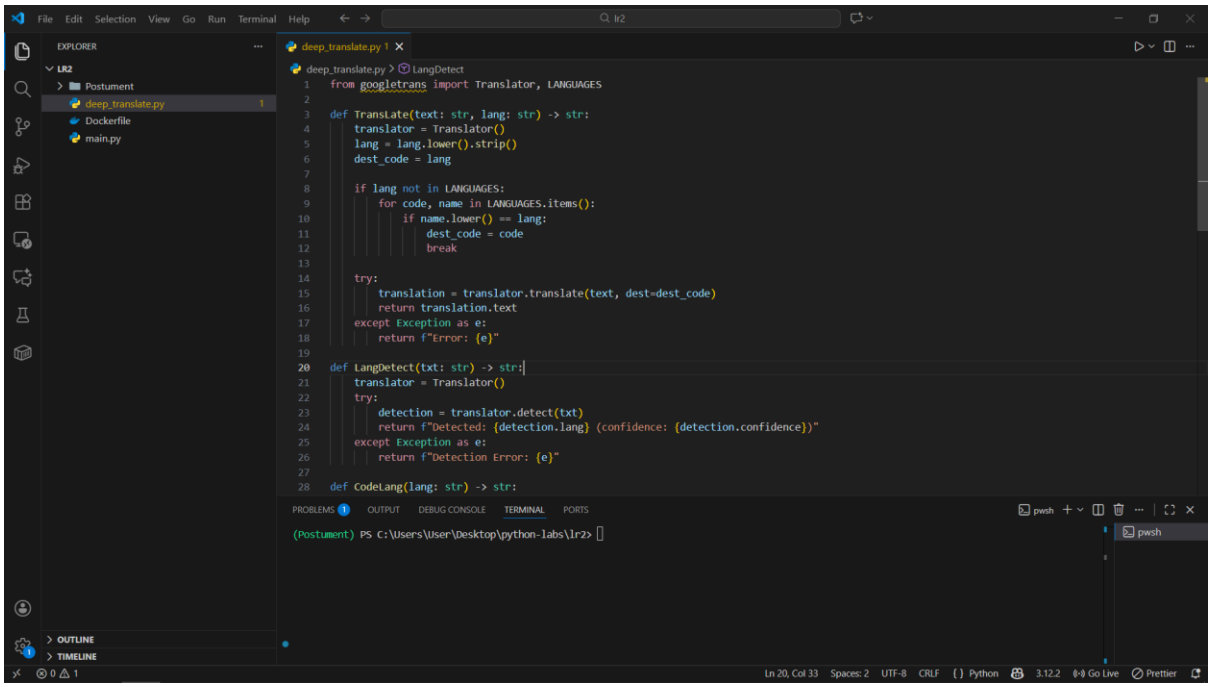

Рис 1.1 – Скріншот середовища VS Code з кодом програми

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("--- ЛОКАЛЬНИЙ 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}")
```

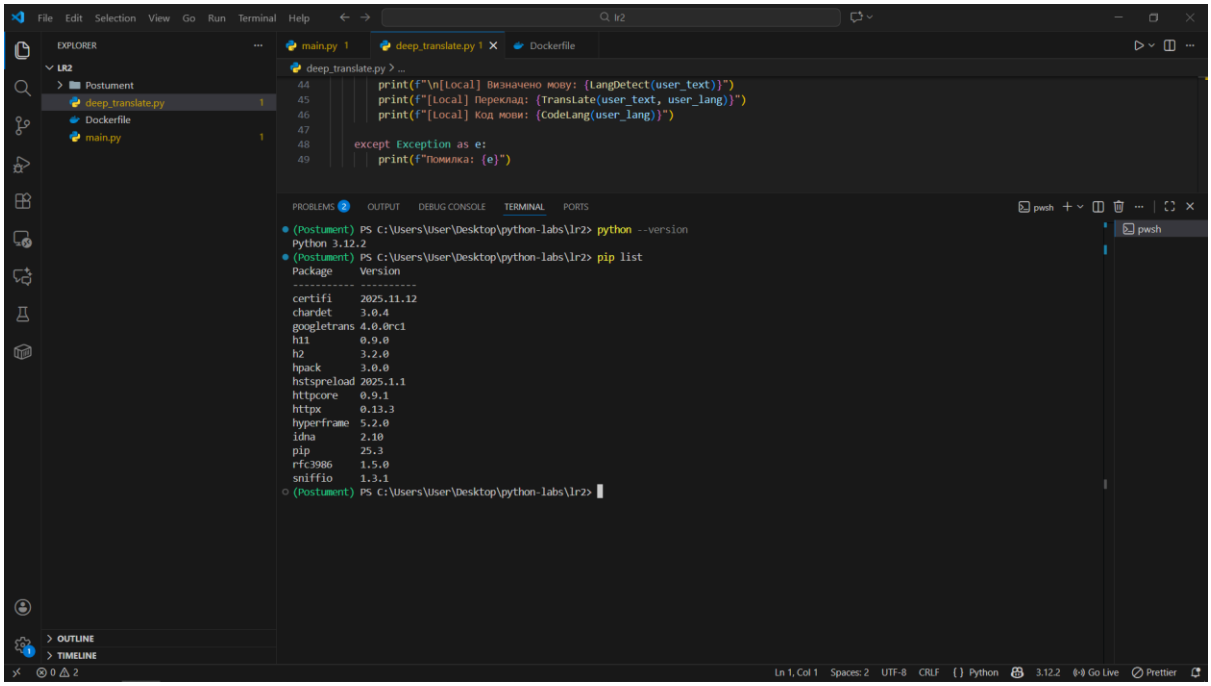

Рис. 2 – Скріншот з версією Python та списком встановлених пакетів

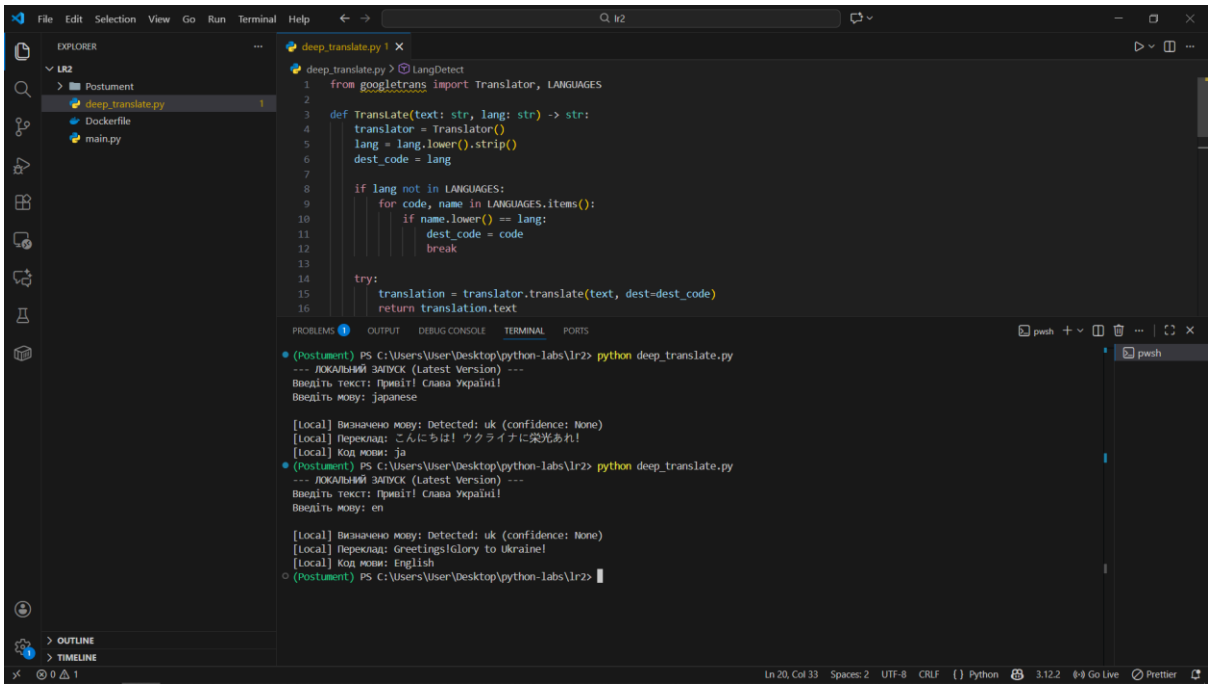

Рис. 3 – Результат виконання програми з пункту 2

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 прапорцем -it")
    except Exception as e:
        print(f"System Error: {e}")
```

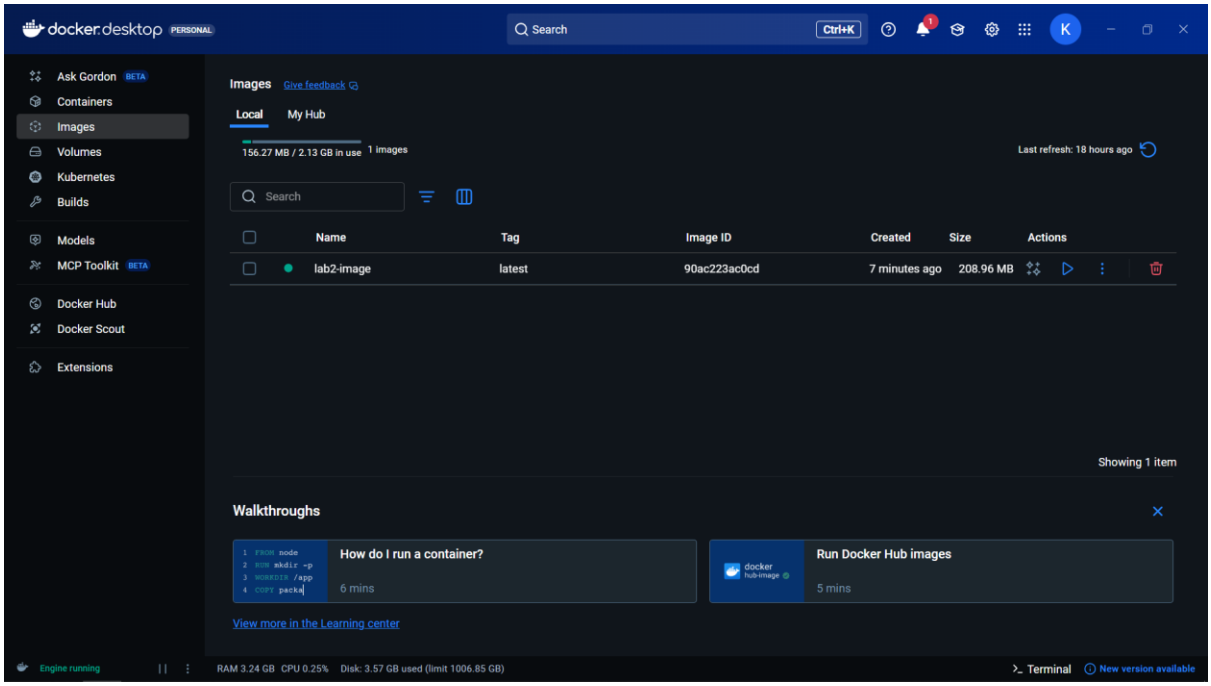

Рис. 4 – Створений docker образ

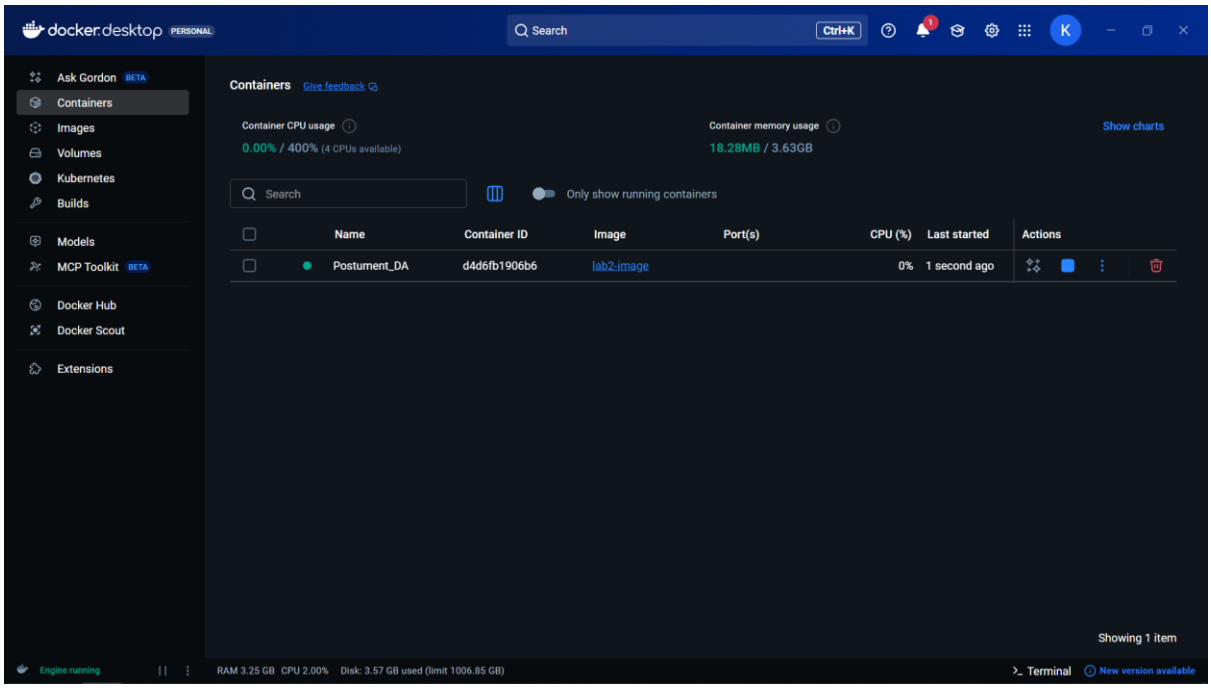

Рис 5 – Створений docker контейнер

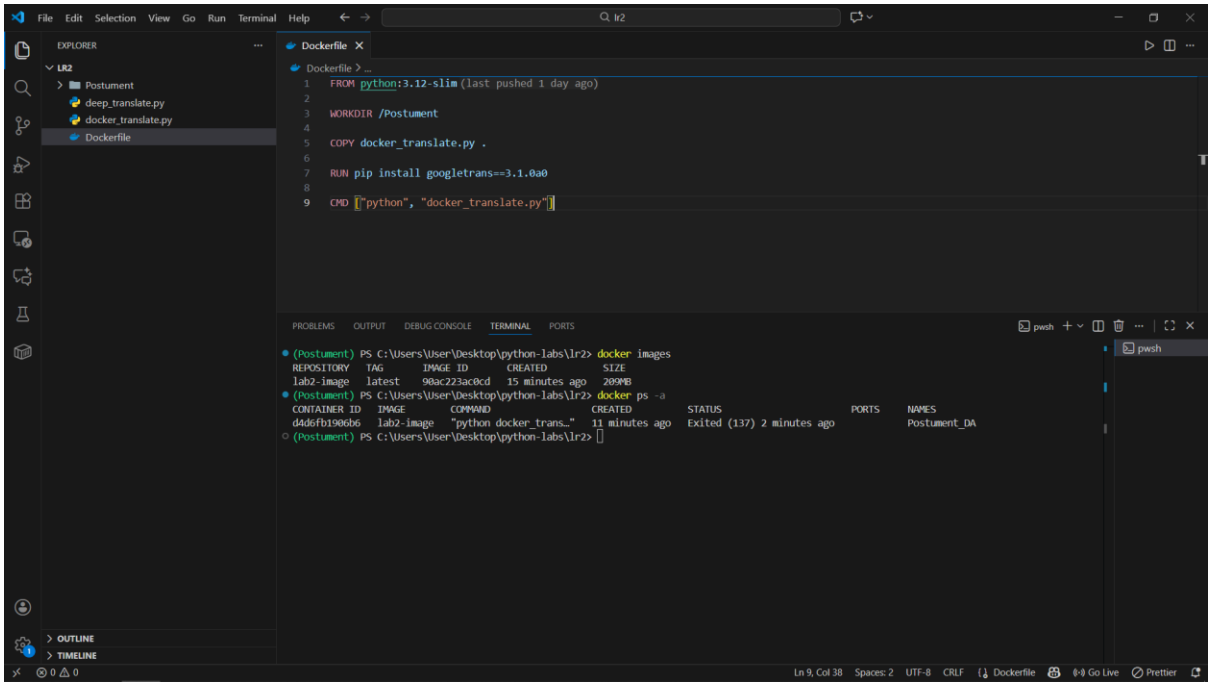

Рис. 6 – Інформація про образ та контейнер в терміналі

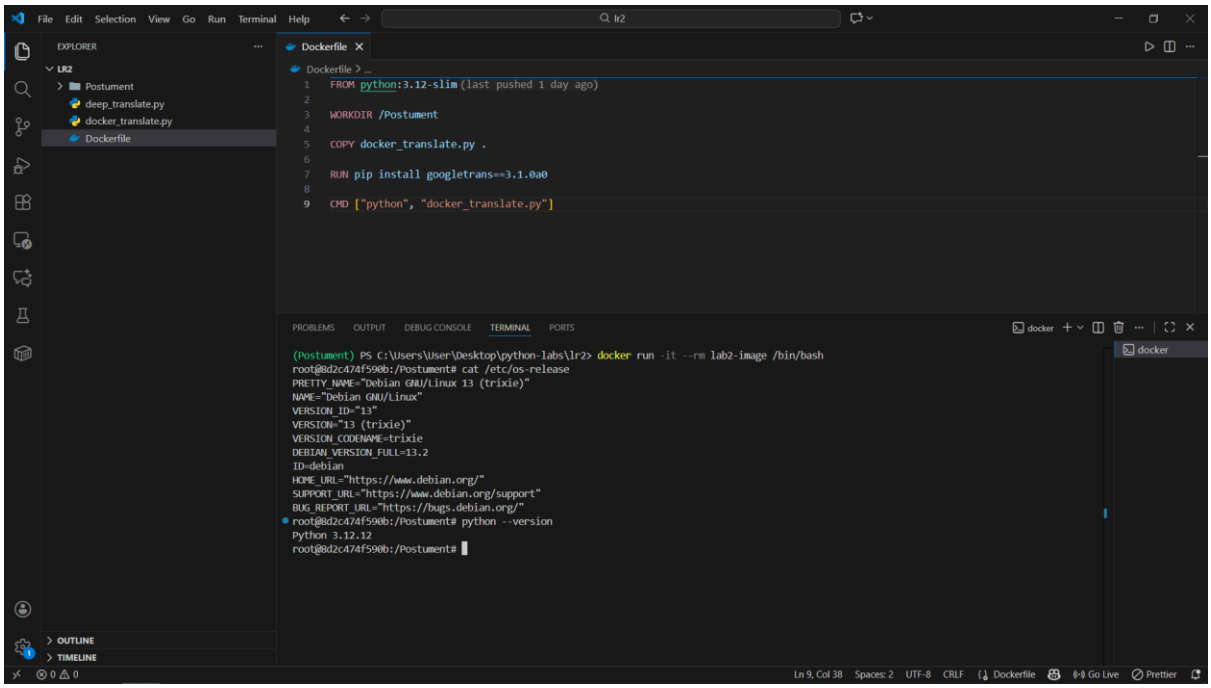

Рис. 7 – Інформація про версію Python та Linux в запущеному контейнері

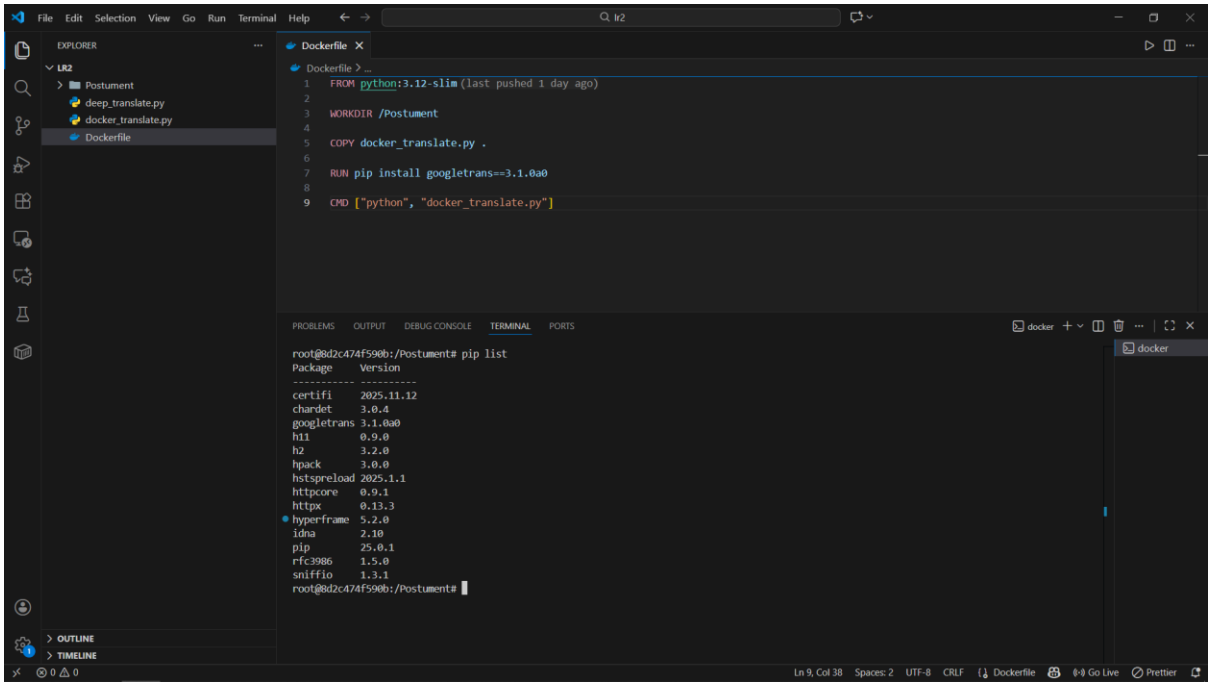

Рис. 8 – Список всіх встановлених пакетів в контейнері

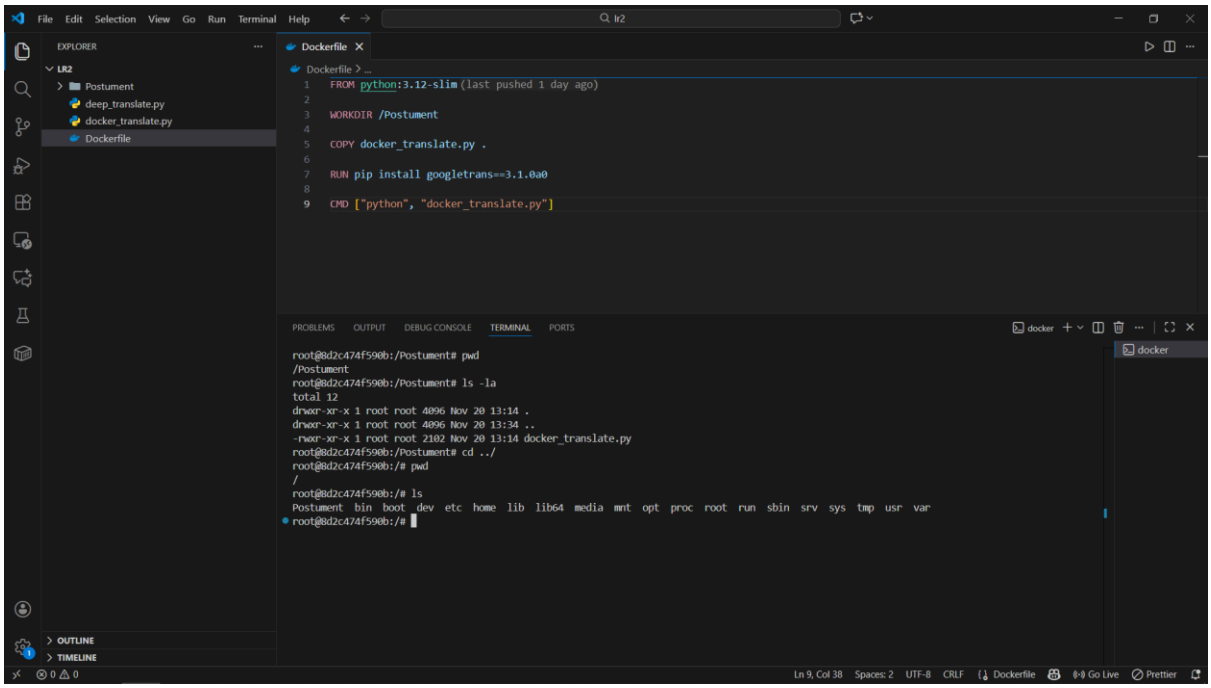

Рис. 9 – Файлова структура контейнеру

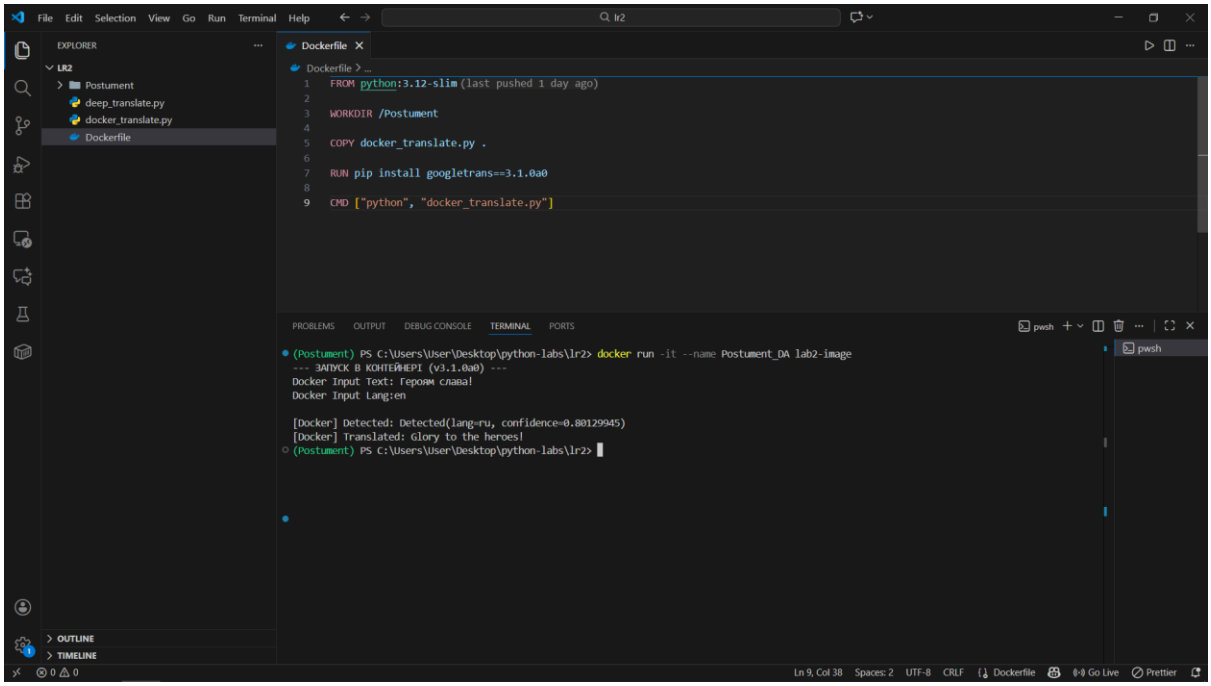

Рис. 10 – Виконання програми з пункту 5 в контейнері

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.