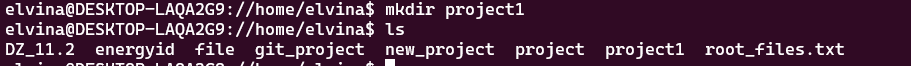
**Задание 1. Создание виртуального окружения**

Для реализации задания используйте встроенный инструмент управления виртуальными окружениями venv, выполнив команду python -m venv.

**Выполните следующие шаги:**

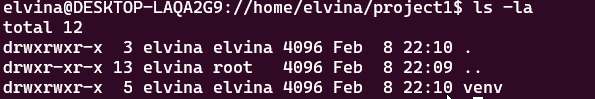
1. Создайте папку под проект project1.

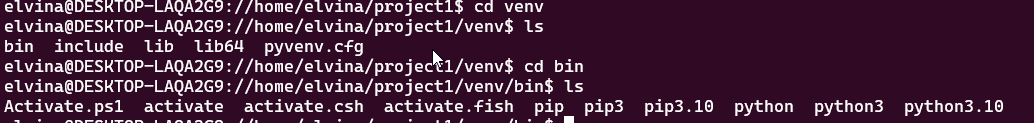


1. Создайте в project1 виртуальное окружение venv.









В некоторых версиях venv вместо папки Scripts внутри виртуального окружения могут создаваться папки bin. Учитывайте это при написании пути до файла активации окружения.

Пример с папкой bin

new\_env

├── bin

│ ├── activate

│ …

│ ├── pip

│ …

│ └── python3 -> …/python3

├── lib

│ └── python3.8

│ └── site-packages

│ ├── pip

│ │ ├── …

│ …

│ └── setuptools-44.0.0.dist-info

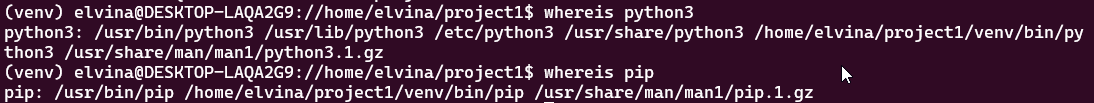
│ ├── …

…

1. Активируйте виртуальное окружение.

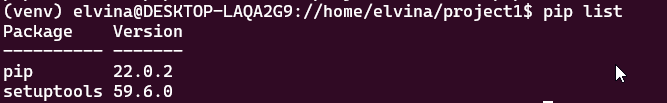


1. Выведите в консоль расположение интерпретатора python и менеджера pip, используемых для работы с этим виртуальным окружением.



Для этого используйте команды gcm pip (PowerShell), whereis pip (Linux), where pip (CMD).

1. Выведите список пакетов, установленных в виртуальном окружении проекта project1.



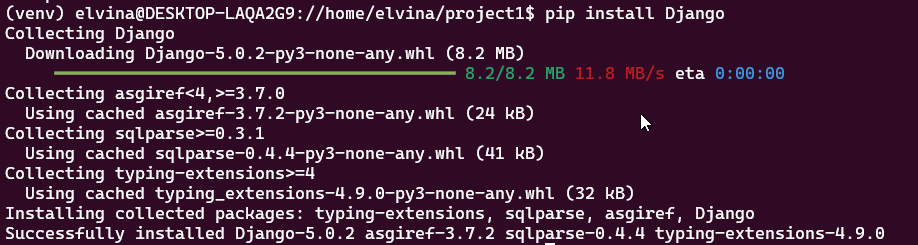
1. Сделайте скриншот, который подтверждает, что вы успешно выполнили задание.

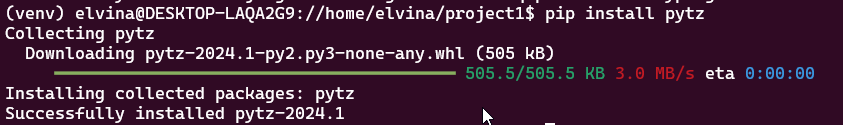
**Задание 2. Настройка виртуального окружения**

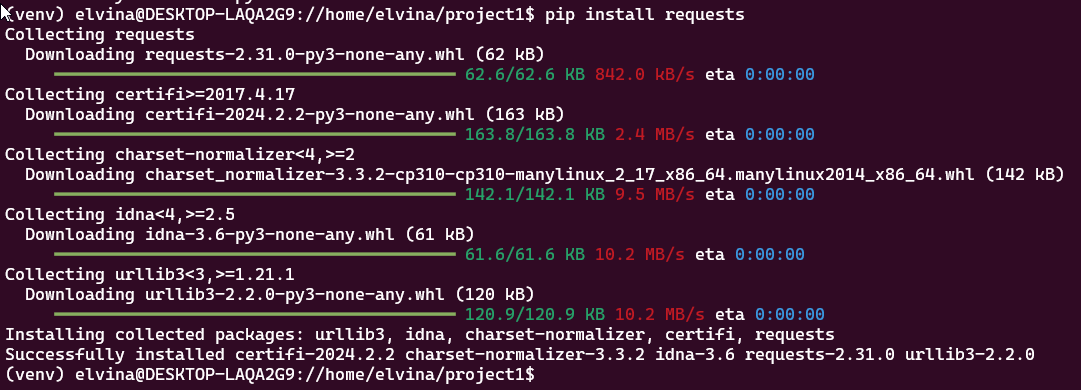
Продолжение задания 1. Установите зависимости в проект.

**Выполните следующие шаги:**

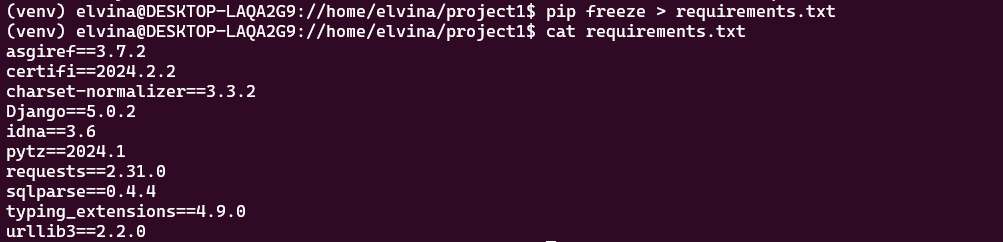
1. Установите следующие библиотеки:
   * django
   * pytz
   * requests







1. Создайте файл requirements.txt, содержащий список библиотек с версиями, установленными для проекта project1.



1. Выйдите из виртуального окружения.



1. Сделайте скриншот, который подтверждает, что вы успешно выполнили задание.

Если появилось сообщение, что доступна новая версия pip, то сделайте его обновление через команду python -m pip install --upgrade pip.

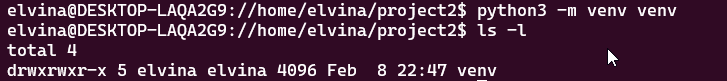
**Задание 3. Воссоздание виртуального окружения**

Продолжение задания 2. Используя список установленных библиотек из project1, настройте окружение для project2.

**Выполните следующие шаги:**

1. Создайте директорию project2 и виртуальное окружение venv.

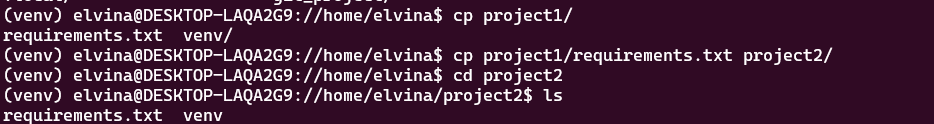




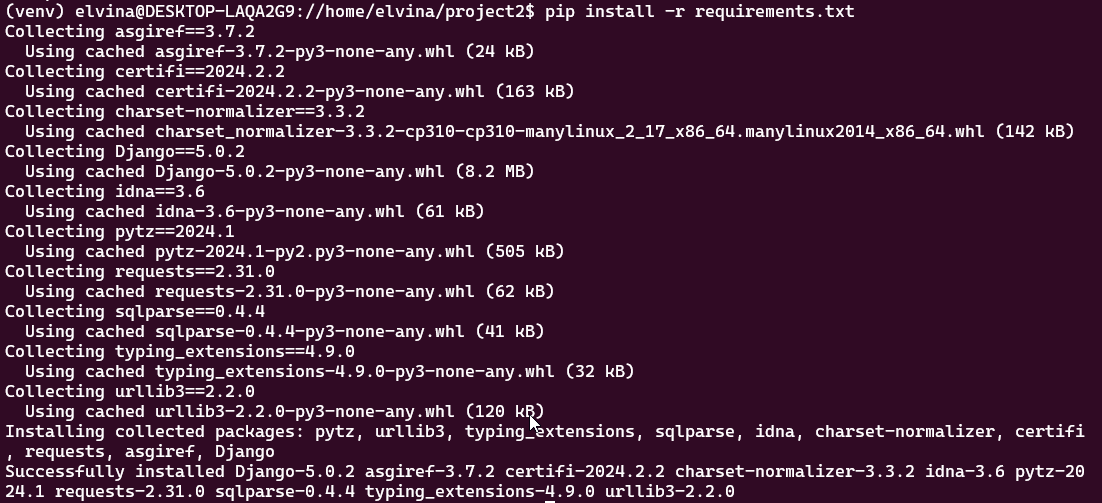
1. Активируйте виртуальное окружение.



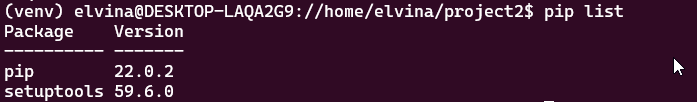
1. Скопируйте файл с зависимостями requirements.txt из project1 в project2.



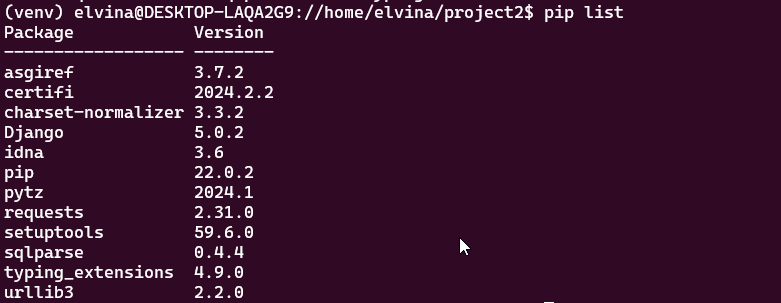
1. Установите записанные в этом файле библиотеки.



До установки из файла:



После установки из файла:

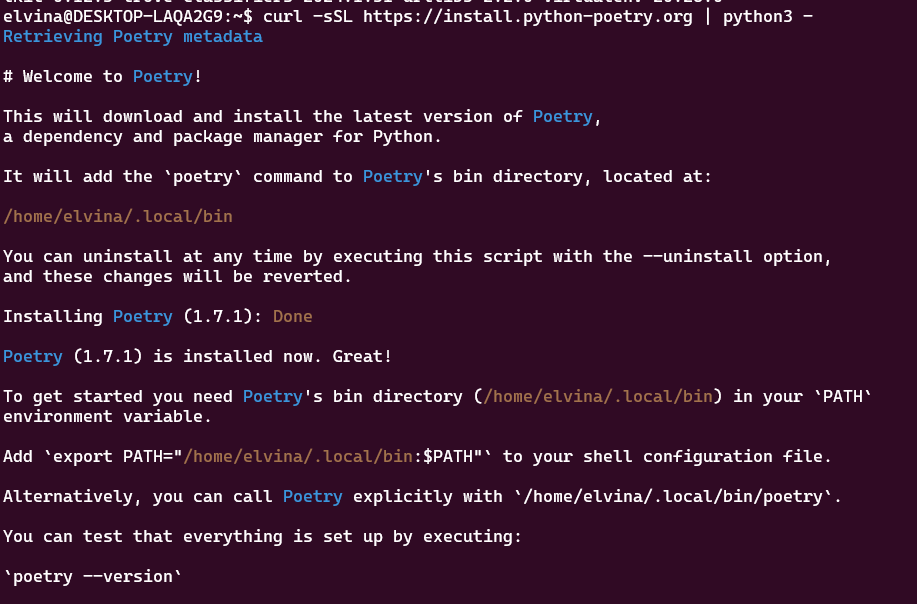


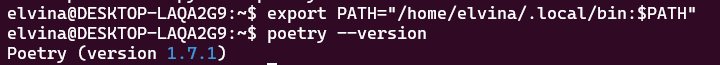
1. Сделайте скриншот, который подтверждает, что вы успешно выполнили задание.

**Задание 4. Poetry. Управление зависимостями**

Создайте виртуальное окружение для проекта, используя Poetry. Установите нужные библиотеки, в том числе и те, которые используются только в процессе разработки и на продакшен не идут.

Установка **Poetry**





**Выполните следующие шаги:**

1. Создайте новый проект project3 с помощью poetry new. У вас получится такая структура:

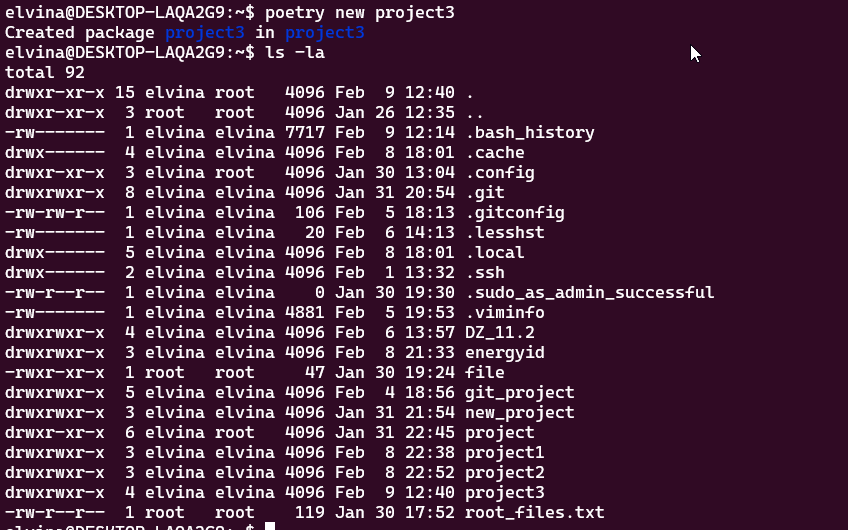
project3

|- project3

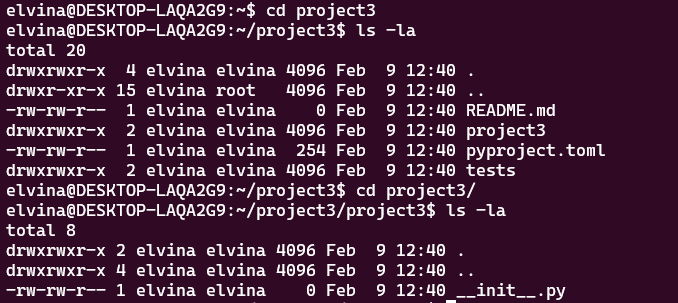
| |- \_\_init\_\_.py

|- tests

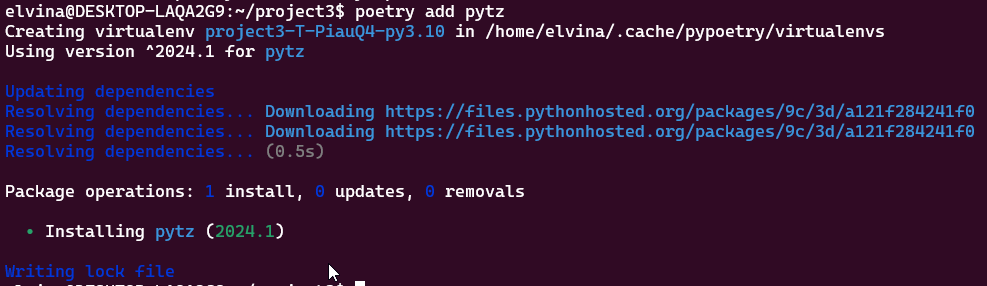
| |- \_\_init\_\_.py

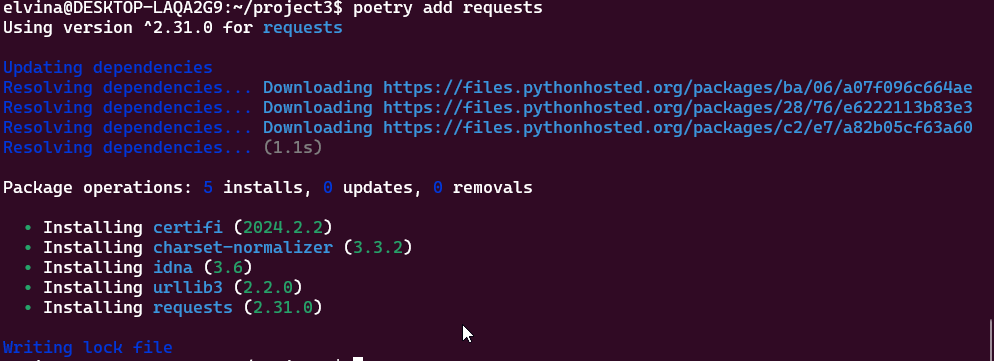


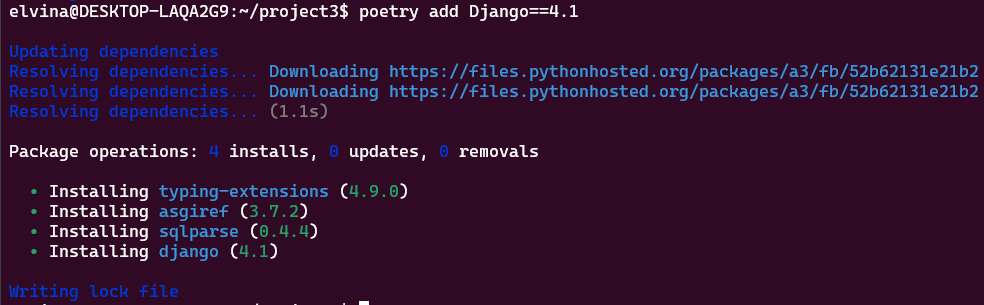
Проверяем, что структура соответствует:



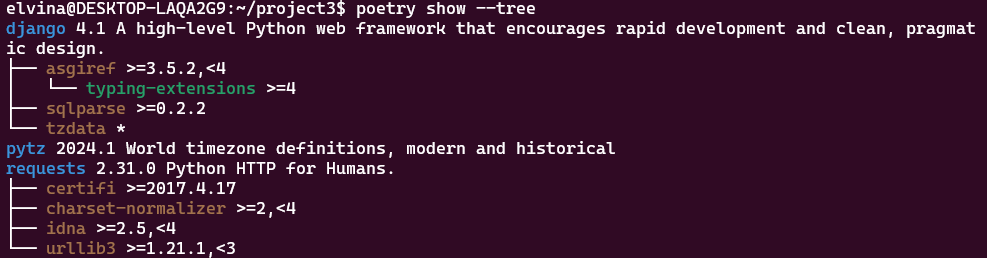
1. Установите следующие библиотеки, используя Poetry:
   * pytz
   * requests
   * django==4.1

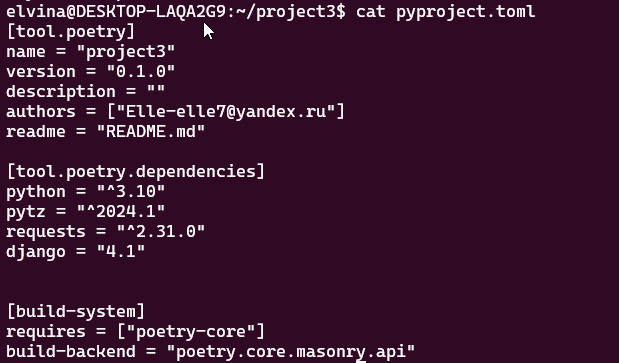




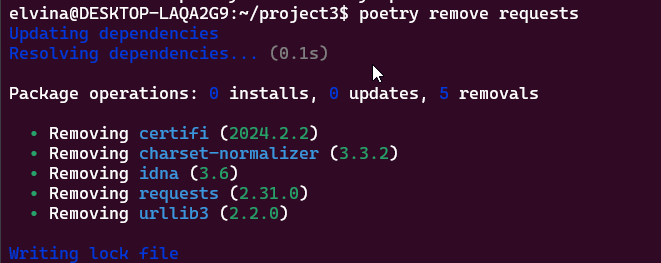


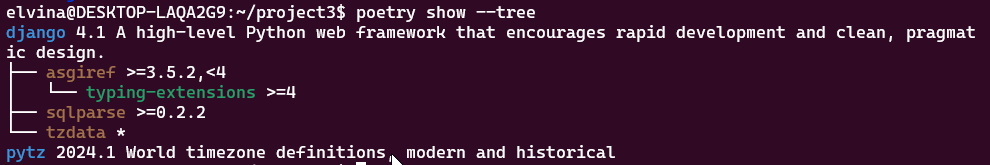
1. Через команду poetry show --tree посмотрите установленные зависимости и зависимости зависимостей. Также посмотрите файл pyproject.toml.



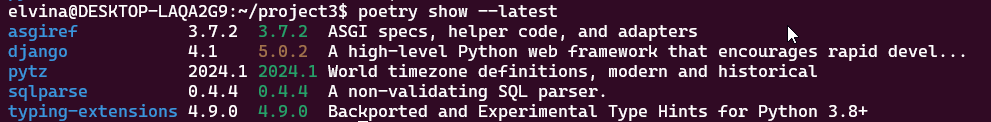


1. Удалите библиотеку requests. Посмотрите, как изменился вывод команды poetry show --tree.

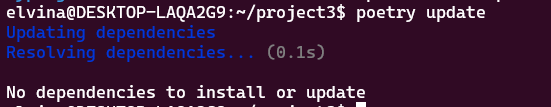




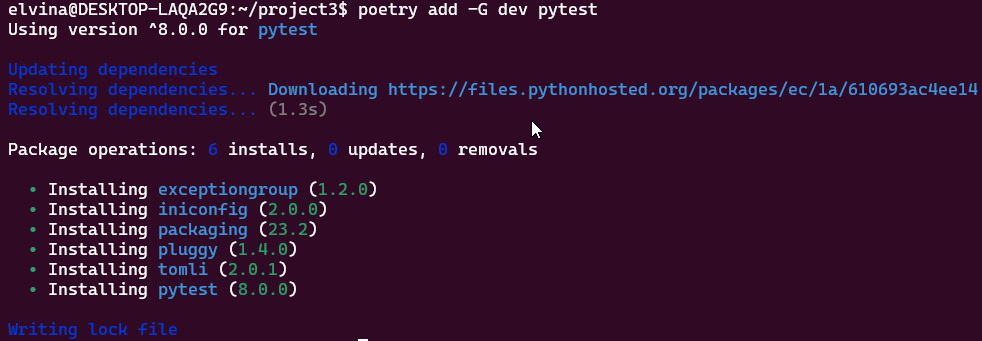
1. Посмотрите, насколько актуальны версии библиотек, установленных для проекта.

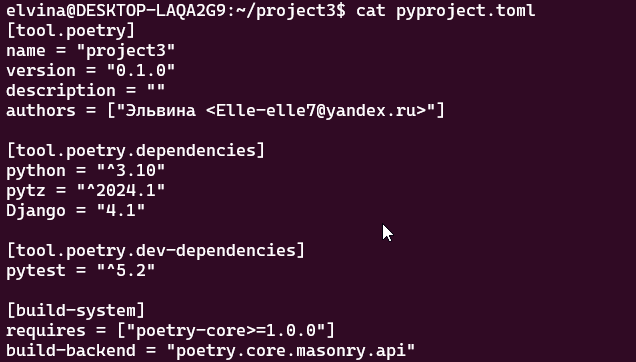


Обновим Django:



1. Установите библиотеку pytest в группу зависимостей dev. Убедитесь, что в pyproject.toml библиотека pytest находится в соответствующем разделе.





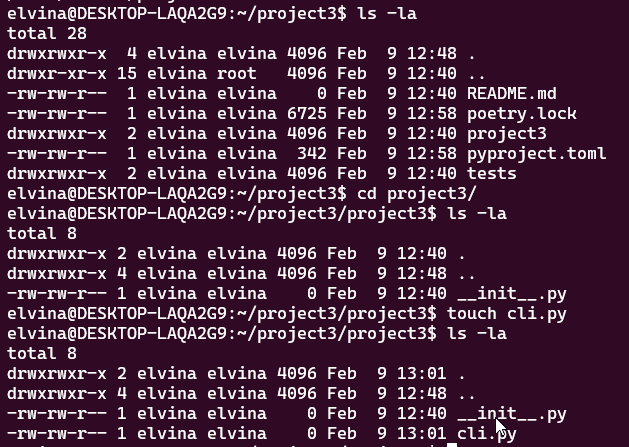
1. Сделайте скриншот, который подтверждает, что вы успешно выполнили задание.

**Задание 5. Poetry. Добавление скрипта**

Зачастую удобно иметь быстрый доступ к запуску некоторой функции. Вы можете дать команду, по которой Poetry будет запускать такую функцию.

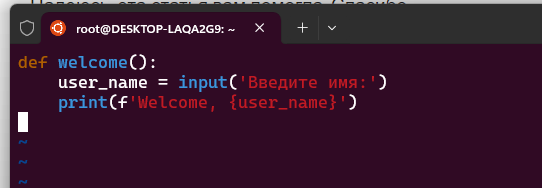
**Выполните следующие шаги:**

1. Создайте файл cli.py в project3/project3.

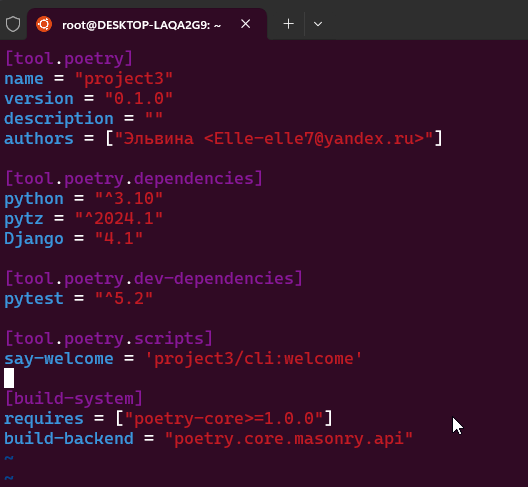


1. В cli.py реализуйте функцию welcome, которая запрашивает имя пользователя и выводит приветственное сообщение с этим именем.

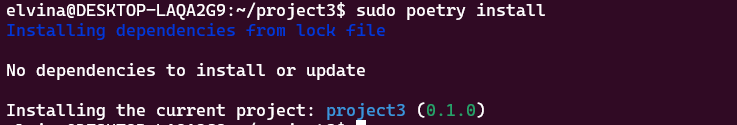




1. Пропишите запуск этой функции в разделе [tool.poetry.scripts] файла pyproject.toml, дав название скрипту, например say-welcome.



1. Выполните команду poetry install, чтобы команда say-welcome стала доступна в виртуальном окружении.



1. Запустите скрипт say-welcome, введите свое имя и получите приветственное сообщение:

$ poetry run say-welcome

Ivan # Ввод пользователя

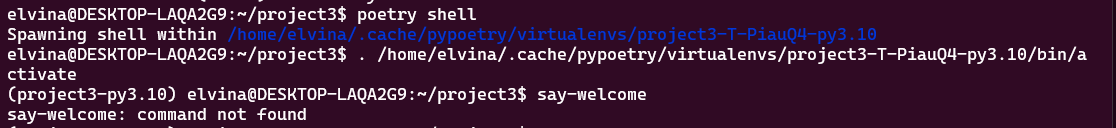
Welcome, Ivan!

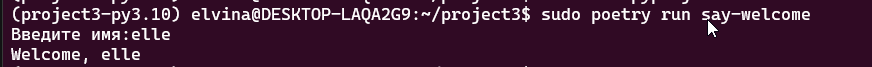
Если вы вошли в виртуальное окружение, выполнив poetry shell:

$ say-welcome

Ivan # Ввод пользователя

Welcome, Ivan!





1. Сделайте скриншот, который подтверждает, что вы успешно выполнили задание.