Лабораторна робота №4

Тема: Контроль версій, віртуальне середовище та бази даних

Мета роботи: навчитися запускати віртуальне середовище та встановлювати залежності локально на прикладі роботи з базою даних.

ТЕОРЕТИЧНА ЧАСТИНА

Віртуальне середовище дозволяє встановлювати локально дистрибутив Python + pip та всі інші пакети для проекту, не впливаючи на пакети встановлені глобально. Це особливо важливо при роботі з лінукс подібними системами, які по замовчуванню вже включають в себе дистрибутив Python та деякі пакети. Вони можуть конфліктувати з новими пакетами, які ви будете інсталювати для розроблюваного проекту.

В цій роботоі ви використажте новий прийом try… exept - читажмо тут https://www.w3schools.com/python/python_try_except.asp

* розгляньте можливість встановлення на свій персональний комп'ютер ОС Linux в якості другої чи основної системи.

ПОРЯДОК ВИКОНАННЯ

Частина 1 - до 80 балів

№	Linux	Опис	Windows

1	`python3 -m pip install --user virtualenv`
	
	`Для деяких випадків: apt-get install python3-venv`
		Інсталюємо віртуальне середовище.	`py -m pip install --user virtualenv`

2	`python3 -m venv myproject`	Переходимо в папку та створюємо віртуальне середовище. Створивши папку 'myproject' - перейдіть в неї і проаналізуйте вміст. pyvenv.cfg - яка інформація у файлі?	`py -m venv myproject`
3	`source myproject/bin/activate`	Активуємо віртуальне середовище.	`.\myproject\Scripts\activate`
4	`which python`	Перевіряємо чи ми використовуємо інтерпритатор з віртуального середовища. Він лежатиме в створеній папці.	`where python`

5	`deactivate`	Вийти з середовища. Крок 3 - активувати.	`deactivate`
6	`python3 -m pip install requests`	Встановити залежності (пакети необхідні для роботи з HTTP).	`py -m pip install requests`
7	`python3 -m pip install --upgrade requests`	Оновити пакети (не виконувати).	`py -m pip install --upgrade requests`
8	`python3 -m pip install -r requirements.txt`	Встановити пакети із файлу (не виконувати).	`py -m pip install -r requirements.txt`

9	`python3 -m pip freeze`
	
	`python3 -m pip freeze > requirements.txt`
		Зафіксувати версію пакетів (залежності) для проекту (не виконувати).
		У конкретний файл теж можна зберегти їх.
			`py -m pip freeze`
			
			`py -m pip freeze > requirements.txt`

10	https://www.digitalocean.com/community/tutorials/how-to-install-the-latest-mysql-on-debian-10	Встановлюємо MySQL server. Не встановлюйте пароль.	https://dev.mysql.com/downloads/mysql/

Тепер створимо додаток app.py який використаємо для роботи з базою даних.
У віртуальному середовищі виконайте команду *code app.py,* це запустить текстовий редактор VSCode з вже відкритим файлом app.py.

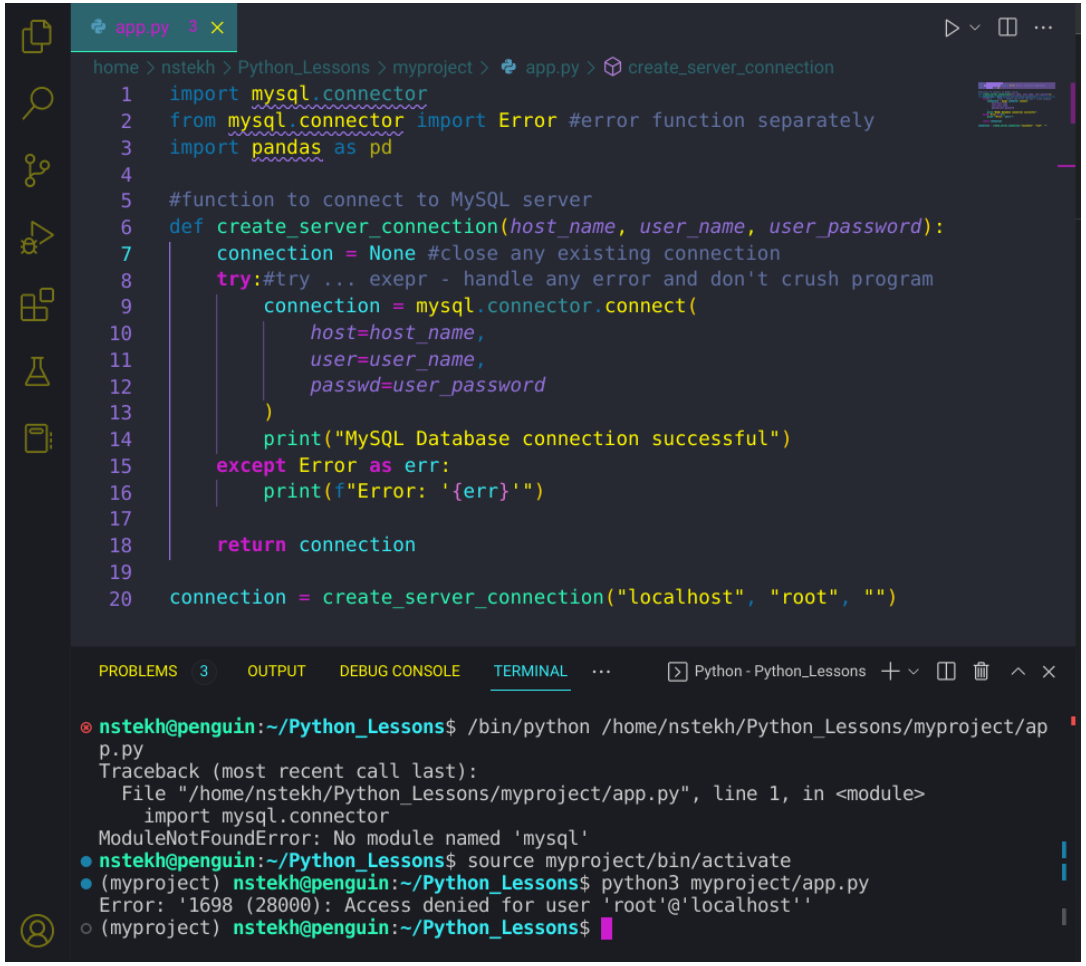
Встановлюємо пакети.

Бібліотека для роботи з БД:
pip install mysql-connector-python

А ще ось ця:
pip install pandas

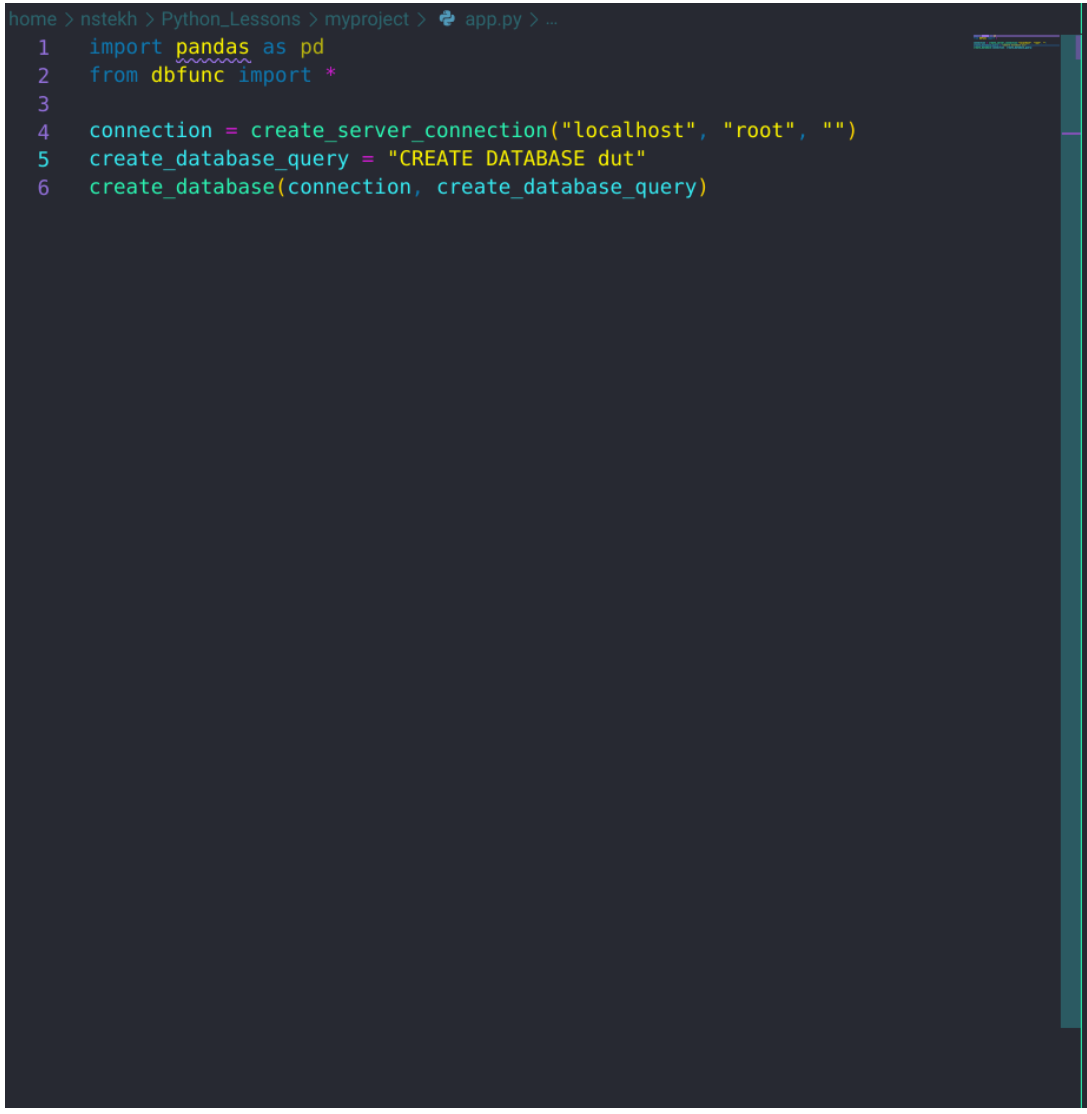
Переходимо до файлу **app.py.** Введіть такий же лістинг програми. Зверніть уваги на помилки які виникають в теміналі. Поки не активоване віртуальне середовище - встановлені пакети інтерпретатор не бачить. Щоб обрати інтерпретатор віртуального середовища у VSCode - потрібно вручну прописати до нього шлях в налаштуваннях. Розібратися в цьому вам допоможе відео (не інсталюйте pyenv):

Linux: https://www.youtube.com/watch?v=1Zgo8M9yUtM&t=77s
Mac: https://www.youtube.com/watch?v=31WU0Dhw4sk&t=8s



Windows: https://www.youtube.com/watch?v=HTx18uyyHw8

Помилка з базою даних (як у мене) виправляжться так:
https://stackoverflow.com/questions/39281594/error-1698-28000-access-denied-for-user-rootlocalhost
Маємо отримати вивід: *MySQL Database connection successful*
Далі створюємо нову базу даних. Виносимо усі функції в новий файл. Підключаємо його через імпорт. У файлі у нас будуть створені функціх для підключення до MySQL, створення БД та підключення до БД.



Зверніть увагу. Вперше програма створить базу даних ‘dut’ успішно, а при повторному виконанні повідомить про те що база даних існує.

Частина 2 - до 90 балів:

Створіть функцію, яка буде виконувати запити до бази даних з програми у вигляді рядків (string). За її допомогою можна буде створити таблицю. Створіть таблицю з інформацією про студентів. Заповніть таблицю даними.

```
def execute_query(connection, query):
    cursor = connection.cursor()
    try:
        cursor.execute(query)
        connection.commit()
        print("Query successful")
    except Error as err:
        print(f"Error: '{err}'")
```

```
create_student_table = """
CREATE TABLE student (
  student_id INT PRIMARY KEY,
  first_name VARCHAR(40) NOT NULL,
  last_name VARCHAR(40) NOT NULL,
  phone_no VARCHAR(20)
  );
 """

connection =
create_db_connection("localhost", "root", "",
"dut")
execute_query(connection, pop_students)
```

```
pop_studets = """
INSERT INTO student VALUES
(1,  'James', 'Smith','+491774553676'),
(2, 'Stefanie',  'Martin', '+491234567890'),
(3, 'Steve', 'Wang', '+447840921333'),
(4, 'Friederike', 'Müller-Rossi','+492345678901'),
(5, 'Isobel', 'Ivanova', '+491772635467');
"""

connection = create_db_connection("localhost", "root", "", "dut")
execute_query(connection, pop_students)
```

Створіть функцію, яка буде читати дані з таблиці.

```
def read_query(connection, query):
    cursor = connection.cursor()
    result = None
    try:
        cursor.execute(query)
        result = cursor.fetchall()
        return result
    except Error as err:
        print(f"Error: '{err}'")
```

```
q1 = """
SELECT *
FROM students;
"""

connection = create_db_connection("localhost", "root", pw, db)
results = read_query(connection, q1)

for result in results:
  print(result)
```

Зафіксуйте версії пакетів командою **pip freeze > requirements.txt**

Які пакети з файлу requirements.txt не встановлені в глобальній версії Python?

Частина 3 - до 90 балів

Написати просенький парсер з Вікіпедії, заповнити таблицю.

Частина 3 - до 100 балів
Інсталювати менеджер залежностей https://pypi.org/project/poetry/.
Виконати всі дії аналогічно з pip.