

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 інсталиувати для розроблюваного проекту.

В цій роботі ви використаєте новий прийом try... except - 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	<code>python3 -m pip install --user virtualenv</code> Для деяких випадків: <code>apt-get install python3-venv</code>	Інсталиємо віртуальне середовище.	<code>py -m pip install --user virtualenv</code>
2	<code>python3 -m venv myproject</code>	Переходимо в папку та створюємо віртуальне середовище. Створивши папку 'myproject' - перейдіть в неї і проаналізуйте вміст. pyvenv.cfg - яка інформація у файлі?	<code>py -m venv myproject</code>
3	<code>source myproject/bin/activate</code>	Активуємо віртуальне середовище.	<code>.\myproject\Scripts\activate</code>
4	<code>which python</code>	Перевіряємо чи ми використовуємо інтерпритатор з віртуального середовища. Він лежатиме в створеній папці.	<code>where python</code>

5	<code>deactivate</code>	Вийти з середовища. Крок 3 - активувати.	<code>deactivate</code>
6	<code>python3 -m pip install requests</code>	Встановити залежності (пакети необхідні для роботи з HTTP).	<code>py -m pip install requests</code>
7	<code>python3 -m pip install --upgrade requests</code>	Оновити пакети (не виконувати).	<code>py -m pip install --upgrade requests</code>
8	<code>python3 -m pip install -r requirements.txt</code>	Встановити пакети із файлу (не виконувати).	<code>py -m pip install -r requirements.txt</code>
9	<code>python3 -m pip freeze</code> <code>python3 -m pip freeze > requirements.txt</code>	Зафіксувати версію пакетів (залежності) для проекту (не виконувати). У конкретний файл теж можна зберегти їх.	<code>py -m pip freeze</code> <code>py -m pip freeze > requirements.txt</code>
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 терміналі. 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 інстальуйте `pyenv`):

Linux: <https://www.youtube.com/watch?v=1Zgo8M9yUtM&t=77s>

Mac: <https://www.youtube.com/watch?v=31WU0Dhw4sk&t=8s>

```

app.py 3 x
home > nstekh > Python_Lessons > myproject > app.py > create_server_connection
1  import mysql.connector
2  from mysql.connector import Error #error function separately
3  import pandas as pd
4
5  #function to connect to MySQL server
6  def create_server_connection(host_name, user_name, user_password):
7      connection = None #close any existing connection
8      try:#try ... expr - handle any error and don't crush program
9          connection = mysql.connector.connect(
10              host=host_name,
11              user=user_name,
12              passwd=user_password
13          )
14          print("MySQL Database connection successful")
15      except Error as err:
16          print(f"Error: '{err}'")
17
18      return connection
19
20  connection = create_server_connection("localhost", "root", "")

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL ... Python - Python_Lessons
nstekh@penguin:~/Python_Lessons$ /bin/python /home/nstekh/Python_Lessons/myproject/ap
p.py
Traceback (most recent call last):
  File "/home/nstekh/Python_Lessons/myproject/app.py", line 1, in <module>
    import mysql.connector
ModuleNotFoundError: No module named 'mysql'
nstekh@penguin:~/Python_Lessons$ source myproject/bin/activate
(myproject) nstekh@penguin:~/Python_Lessons$ python3 myproject/app.py
Error: '1698 (28000): Access denied for user 'root'@'localhost''
(myproject) nstekh@penguin:~/Python_Lessons$

```

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

Помилка з базою даних (як у мене) виправляється 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 функції для підключення до MySQL, створення БД та підключення до БД.

home > nstekh > Python_Lessons > myproject > app.py > ...

```
1 import pandas as pd
2 from dbfunc import *
3
4 connection = create_server_connection("localhost", "root", "")
5 create_database_query = "CREATE DATABASE dut"
6 create_database(connection, create_database_query)
```

home > nstekh > Python_Lessons > myproject > dbfunc.py > create_db_connection

```
1 import mysql.connector
2 from mysql.connector import Error #error function separately
3
4 #function to connect to MySQL server
5 def create_server_connection(host_name, user_name, user_password):
6     connection = None #close any existing connection
7     try:#try ... except - handle any error and don't crush program
8         connection = mysql.connector.connect(
9             host=host_name,
10            user=user_name,
11            passwd=user_password
12        )
13        print("MySQL Database connection successful")
14    except Error as err:
15        print(f"Error: '{err}'")
16    return connection
17
18 #function to connect to database
19 def create_db_connection(host_name, user_name, user_password, db_name):
20     connection = None
21     try:
22         connection = mysql.connector.connect(
23             host=host_name,
24             user=user_name,
25             passwd=user_password,
26             database=db_name
27         )
28         print("MySQL Database connection successful")
29     except Error as err:
30         print(f"Error: '{err}'")
31     return connection
32
33 #function to create db
34 def create_database(connection, query):
35     cursor = connection.cursor()
36     try:
37         cursor.execute(query)
38         print("Database created successfully")
39     except Error as err:
40         print(f"Error: '{err}'")
```

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.

<pre>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}'")</pre>	<pre>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)</pre>	<pre>pop_students = """ 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)</pre>
--	--	---

Створіть функцію, яка буде читати дані з таблиці.

<pre>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}'")</pre>	<pre>q1 = """ SELECT * FROM students; """ connection = create_db_connection("localhost", "root", pw, db) results = read_query(connection, q1) for result in results: print(result)</pre>
--	--

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 рір.