1. Section 1:

* Kiểm tra chắc chắn đã cài python, docker trên máy tính
* Tạo folder chứa project, mở bằng VS Code

Mở terminal -> cài virtual enviroment của python trên vs code để chạy vs docker:

pip install virtualenv

tạo virtual enviroment trong folder hiện tại, đặt tên là venv:

python -m virtualenv venv

sau khi tạo xong venv, mở tab terminnal mới để vào trong venv



Note: để vào bên trong venv, terminal tự động chạy dòng lệnh



Có thể xảy ra lỗi

….venv\Scripts\activate.ps1 cannot be loaded because running scripts is disabled on this system.

Xem tại

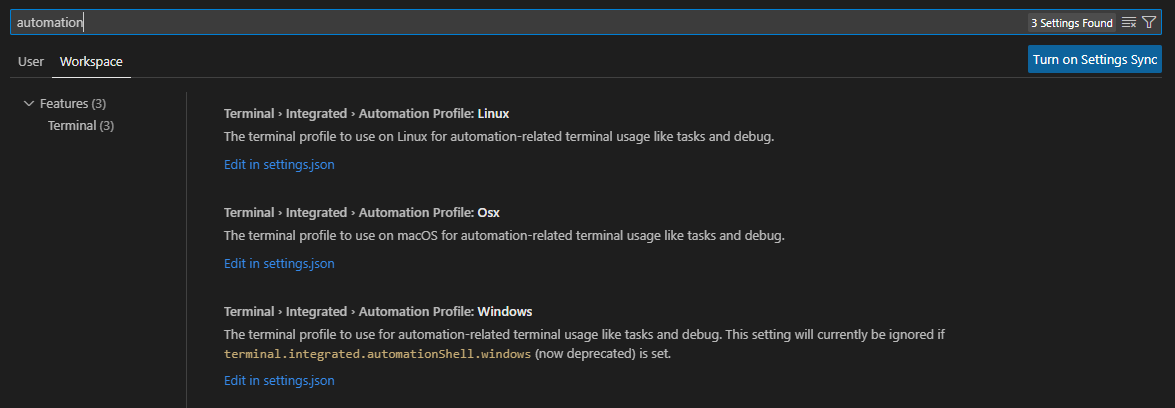
https://stackoverflow.com/questions/56199111/visual-studio-code-cmd-error-cannot-be-loaded-because-running-scripts-is-disabl

Do execution policies không cho phép. Pypass bằng cách:

Cách 1. open PowerShell as Administrator and paste this.

Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser

Cách 2. VS Code: vào File -> Prefrerences -> Settings, search “automation” và mở tab Workspace



Chọn edit in setting.json đúng hđh đang sử dụng. Paste đoạn code

"terminal.integrated.profiles.windows": {

"PowerShell": {

"source": "PowerShell",

"icon": "terminal-powershell",

"args": ["-ExecutionPolicy", "Bypass"]

}

},

"terminal.integrated.defaultProfile.windows": "PowerShell",

Sau đó khởi động lại VS Code

* Cài thư viện trong venv

pip install psycopg2-binary

pip install asyncpg sqlalchemy~=2.0

* Setting up postgresql with docker

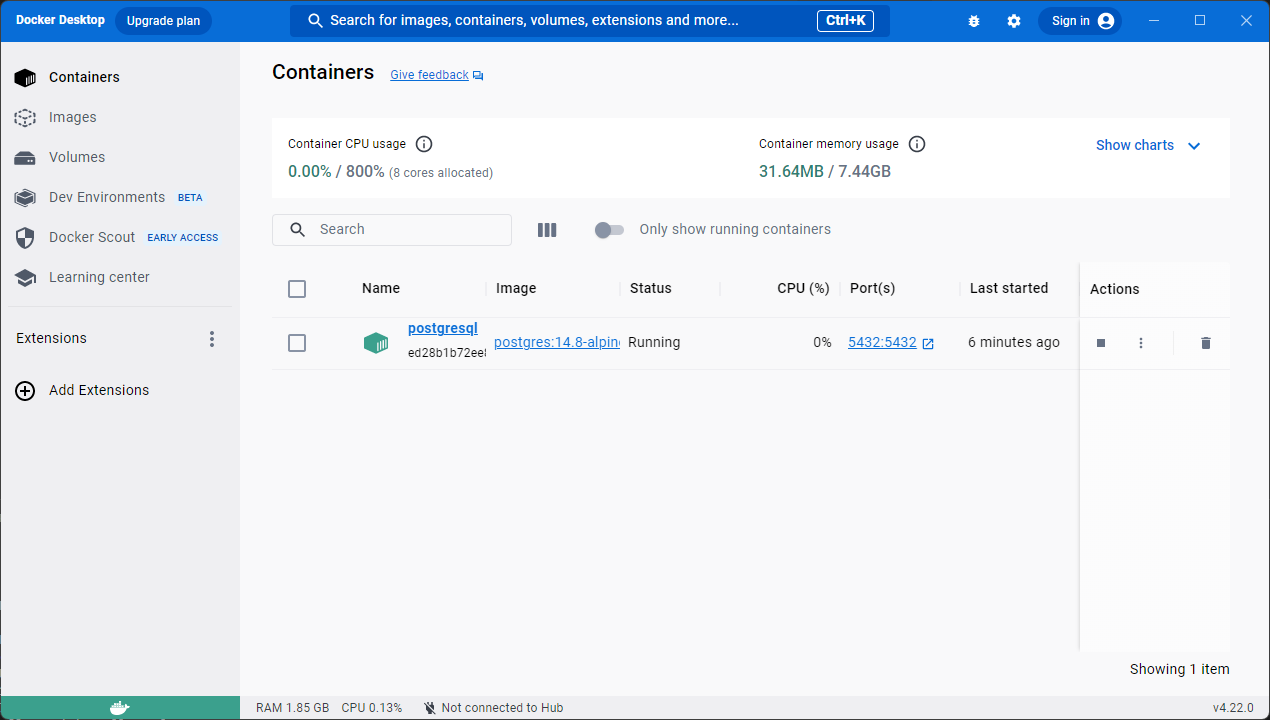
Gõ docker để kiểm tra đã cài docker chưa, engine đang running hay k

Tạo container:

docker run --name postgresql -e POSTGRES\_PASSWORD=[password tùy chọn] -e POSTGRES\_USER=[Tên user] -e POSTGRES\_DB=[Tên DB] -p 5432:5432 postgres:[version postgresql muốn tạo]-alpine

(alpine : lightweight distribution)

Mở docker desktop để kiểm tra đã tạo container thành công chưa



\*Connection pools

Server applications cần đáp ứng được request độc lập từ nhiều client. Nếu đơn giản, Server app khi sử dụng DB sẽ tạo mới một connection đến DB cho mỗi request-> k thể scale bởi vì tại một thời điểm chỉ có thể mở một lượng giới hạn connection nếu không hệ thống sẽ bị treo. Việc close connection cũng tốn tài nguyên của cả applications và cả DB.

-> patterns về connection pools ra đời:

Hoạt động:

1 App sẽ implements một connection pool mở n connections đến Database. Có cơ chế đánh dấu connections là "available" hay "in use". Khi gọi connect() thì 1 connection sẽ được lấy ra khỏi pool (được đánh dấu là "in use" và trả về 1 connection cho request). Khi close() thì nó sẽ được put lại pool (chứ không thực sự đóng lại).

Ngăn connections leaking:

Connection pools có config max connection có thể mở tại 1 thời điểm. Leak connection sẽ làm cho lời gọi connect() bị treo mãi mãi (connection pools sẽ từ chối mở thêm connection mới và đợi đến khi một connection "available")

-> Để ngăn leak hãy đảm bảo gọi close() khi không sử dụng

Note: If running postgresql both on docker and PC, make sure they are not using use the same port.

1. Session 2

Create table

**with** session\_pool() **as** session:

query **=** text("""

CREATE TABLE users

(

telegram\_id BIGINT PRIMARY KEY,

full\_name VARCHAR(255) NOT NULL,

username VARCHAR(255),

language\_code VARCHAR(255) NOT NULL,

created\_at TIMESTAMP DEFAULT NOW(),

referrer\_id BIGINT,

FOREIGN KEY (referrer\_id)

REFERENCES users (telegram\_id)

ON DELETE SET NULL

);

""")

session**.**execute(query)

*# and commit the changes*

session**.**commit()

Add some records and try to print them

**with** session\_pool() **as** session:

insert\_query **=** text("""

INSERT INTO users (telegram\_id, full\_name, username, language\_code, referrer\_id)

VALUES (1, 'John Doe', 'johndoe', 'en', NULL),

(2, 'Jane Doe', 'janedoe', 'en', 1);

""")

session**.**execute(insert\_query)

session**.**commit()

select\_query **=** text("""

SELECT \* FROM users;

""")

result **=** session**.**execute(select\_query)

**for** row **in** result:

print(row)

Create SQLAlchemy table classes:

-Create a class that inherits from the declarative base class

-To create columns in the table, need to create new attributes and assign them. Since ver 2.0, you can use mapped\_column function, mapped type annotations for column types

-SQL data types: import specific object from the sqlalchemy modulem

-Primary key: pass the primary\_key argument to the column

-null constrain: pass the nullabe=True argument to the column

-Default value: pass the server\_default argument to the column

-Functions in sqlalchemy: use the sqlalchemy.func module

-nameing a table: always specify the \_ \_table\_ \_ attribute