Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра автоматики та управління в технічних системах

Лабораторна робота №3

з дисципліни «Інтелектуальні вбудовані системи»

Виконав:

студент групи ІА-02

Гузь Богдан

Перевірив:

Нікольський С. С.

Київ 2024

**Постановка задачі**

Потрібно реалізувати частину Hub. Cервіс який займається

накопиченням отриманих даних від Agent/EdgeDataLogic та подальшим зберіганням оброблених даних в БД. Перед збереженням дані потрібно накопичити (наприклад 10 записів), так як такі дані простіше аналізувати і запис в БД множини даних буде швидшим.

**Хід роботи:**

1.Створення віртуального середовища. Пакет virtualenv повинен бути встановлений в системі.

python -m venv venv

2. Активація середовища

source ./venv/bin/activate # linux

.\venv\Scripts\activate.bat # windows

3. Встановимо необхідні залежності(бібліотеки)

pip install fastapi

pip install "uvicorn[standard]"

pip install SQLAlchemy

pip install psycopg2

pip install pydantic

4. Збережемо залежності в requirements.txt

pip freeze > requirements.txt

5.Реалізуємо StoreApiAdapter. Необхідно імплементувати запит до

StoreApi для збереження даних. А саме зробити post запит на

‘{store\_host}/processed\_agent\_data’ з списком елементів

ProcessedAgentData.

*import json*

*import logging*

*from typing import List*

*import requests*

*from app.entities.processed\_agent\_data import ProcessedAgentData*

*from app.interfaces.store\_gateway import StoreGateway*

*class StoreApiAdapter(StoreGateway):*

*def \_\_init\_\_(self, api\_base\_url):*

*self.api\_base\_url = api\_base\_url*

*def save\_data(self, processed\_agent\_data\_batch: List[ProcessedAgentData]) -> bool:*

*try:*

*data = [data.dict() for data in processed\_agent\_data\_batch]*

*for item in data:*

*item['agent\_data']['timestamp'] = item['agent\_data']['timestamp'].isoformat()*

*response = requests.post(f"{self.api\_base\_url}/processed\_agent\_data", json=data)*

*if response.ok:*

*logging.info("Data saved successfully")*

*return True*

*else:*

*logging.error(f"Failed to save data. Status code: {response.status\_code}")*

*return False*

*except Exception as e:*

*logging.error(f"Error saving data: {e}")*

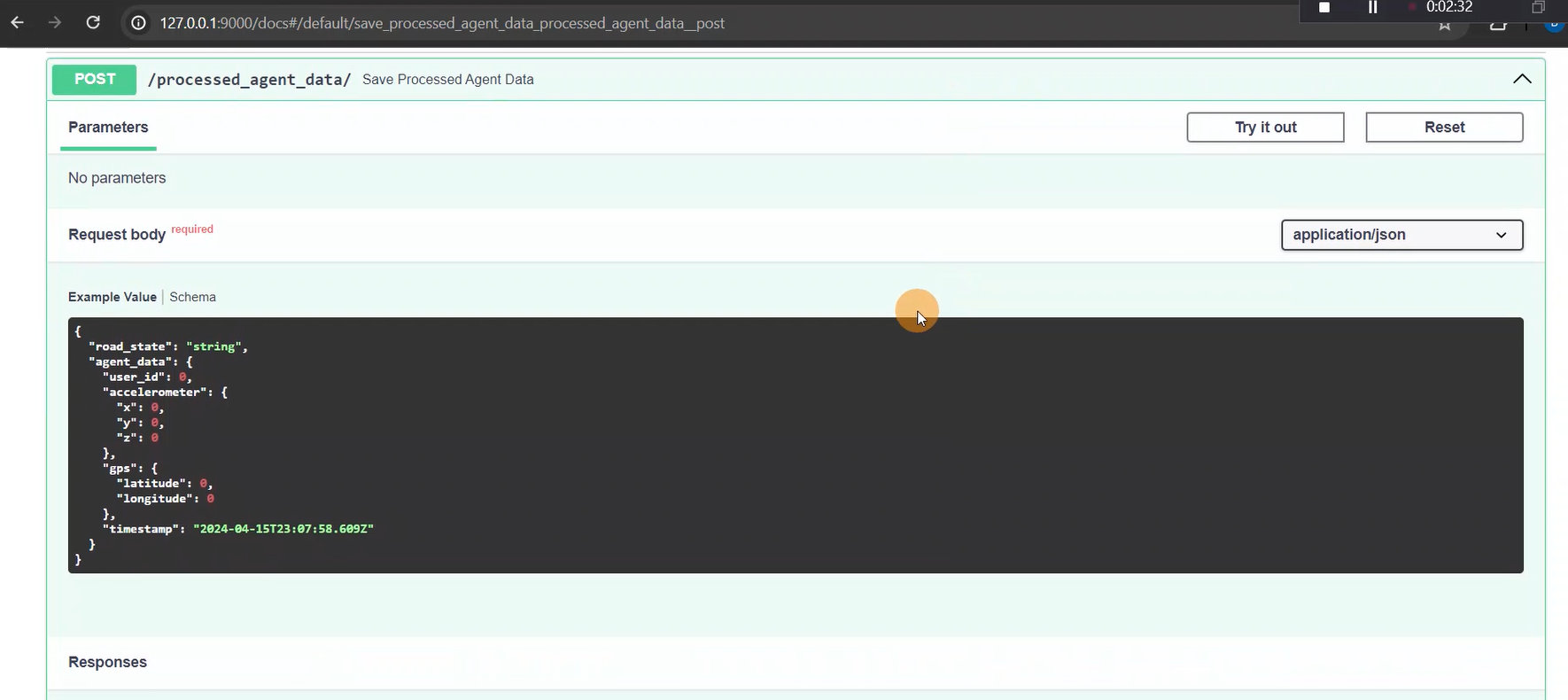
*return False*

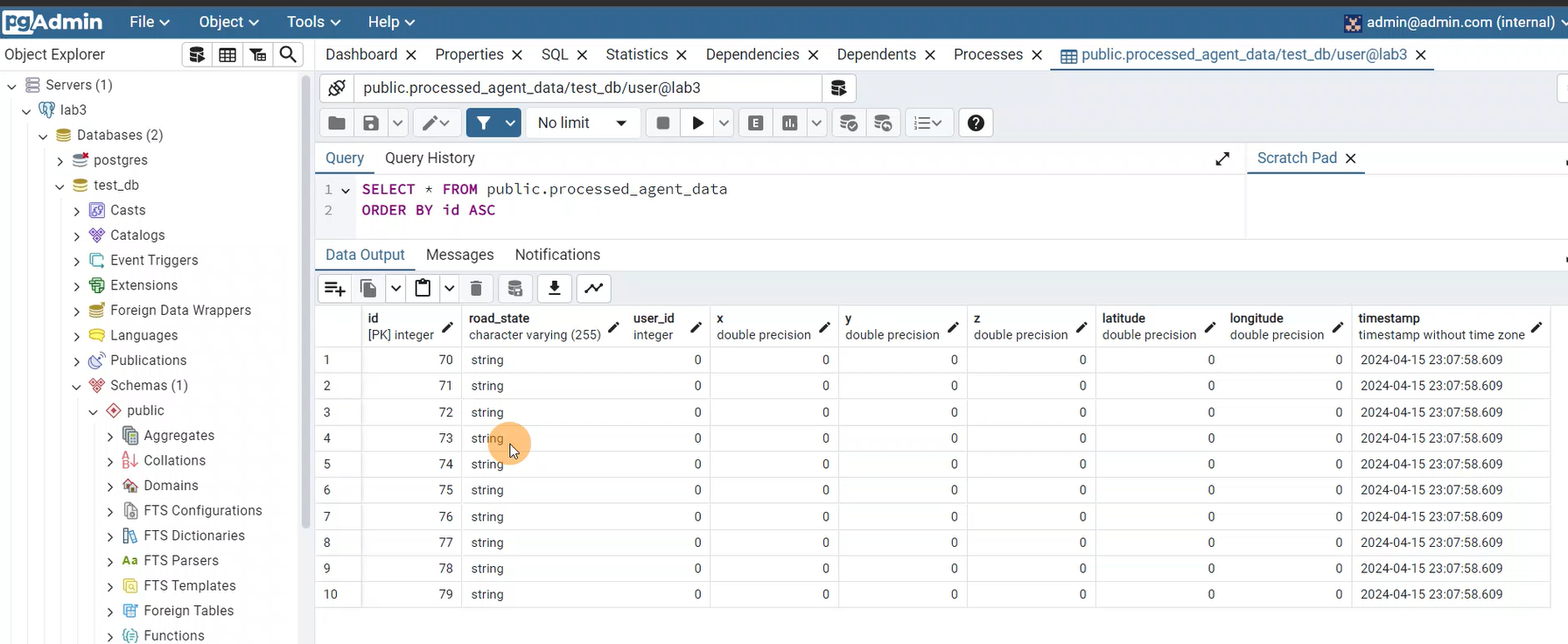
6.Запуск програму в docker

cd docker

docker-compose up --build

1. Перевірка працездатності через Swagger та PgAdmin.





(Хоча в config.py BATCH\_SIZE =20,для спрощення в цьому конкретному випадку,BATCH\_SIZE =10)

Оскільки додалося 10 елементів,накопичення данних працює.

Також,оскільки елементи додаються в БД ,то рost запит на‘{store\_host}/processed\_agent\_data’ з списком елементів ProcessedAgentData теж працює

Очікувані результати в протоколі

* Посилання на git репозиторій з проектом
* Реалізація StoreApiAdapter

*import json*

*import logging*

*from typing import List*

*import requests*

*from app.entities.processed\_agent\_data import ProcessedAgentData*

*from app.interfaces.store\_gateway import StoreGateway*

*class StoreApiAdapter(StoreGateway):*

*def \_\_init\_\_(self, api\_base\_url):*

*self.api\_base\_url = api\_base\_url*

*def save\_data(self, processed\_agent\_data\_batch: List[ProcessedAgentData]) -> bool:*

*try:*

*data = [data.dict() for data in processed\_agent\_data\_batch]*

*for item in data:*

*item['agent\_data']['timestamp'] = item['agent\_data']['timestamp'].isoformat()*

*response = requests.post(f"{self.api\_base\_url}/processed\_agent\_data", json=data)*

*if response.ok:*

*logging.info("Data saved successfully")*

*return True*

*else:*

*logging.error(f"Failed to save data. Status code: {response.status\_code}")*

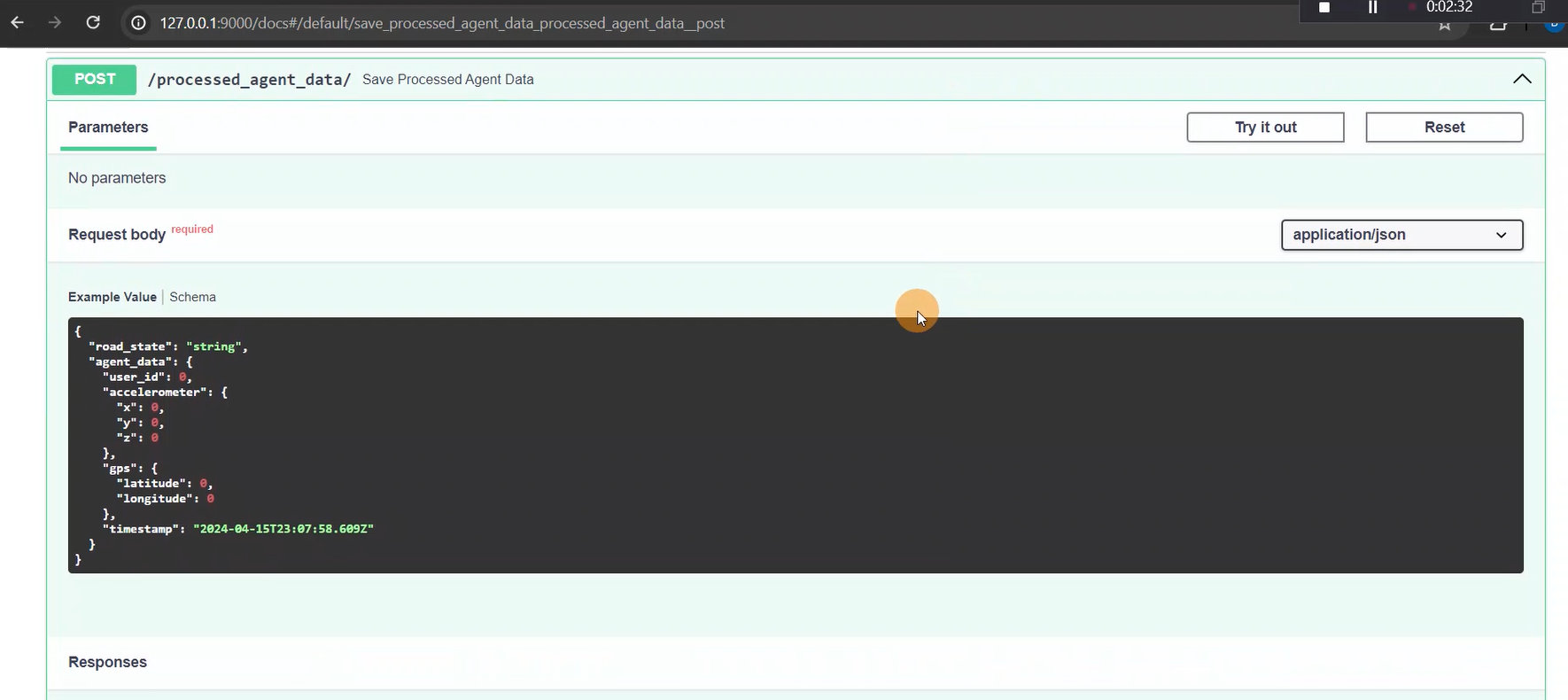
*return False*

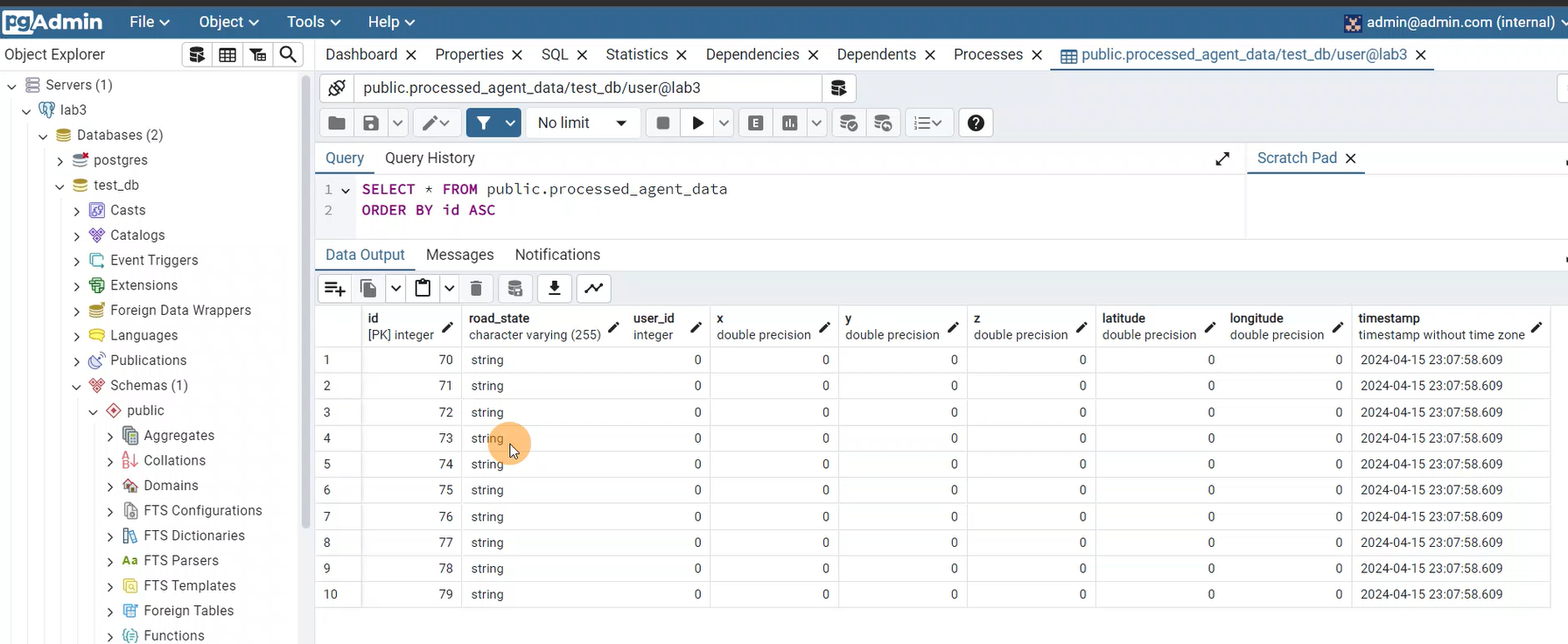
*except Exception as e:*

*logging.error(f"Error saving data: {e}")*

*return False*

* Показ результатів роботи в Swagger, PgAdmin





* Висновки

В даній лабораторній роботі було реалізовано частину Hub. Cервіс який займається накопиченням отриманих даних від Agent/EdgeDataLogic та подальшим зберіганням оброблених даних в БД. Перед збереженням даних,було розроблено накопичення даних (наприклад 10 записів) і запис в БД множини даних.

Реалізували StoreApiAdapter ,а саме зробили post запит на

‘{store\_host}/processed\_agent\_data’ з списком елементів

ProcessedAgentData.