Міністерство освіти і науки України

Національний технічний університет України «КПІ ім. Ігоря Сікорського»

Факультет Інформатики та Обчислювальної Техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №2

З дисципліни «Інтелектуальні вбудовані системи»

Виконав:

Студент групи ІО-04

Ленков Н.О.

Перевірив:

Нікольський С. С.

Київ 2024

**Посилання на git репозиторій проекту:**

https://github.com/LENKOVDEV/IES

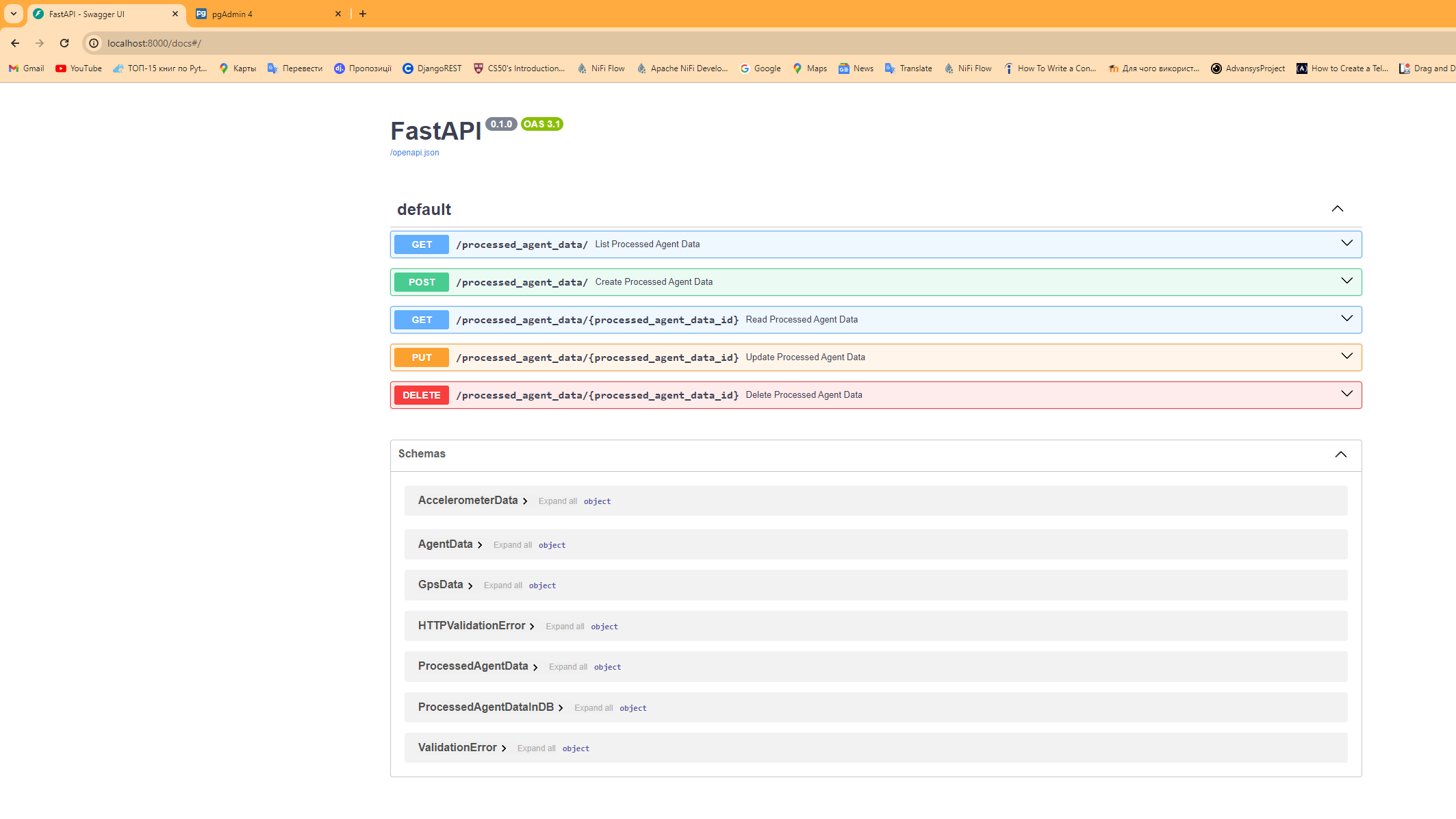
**Код програми:**

*main.py*

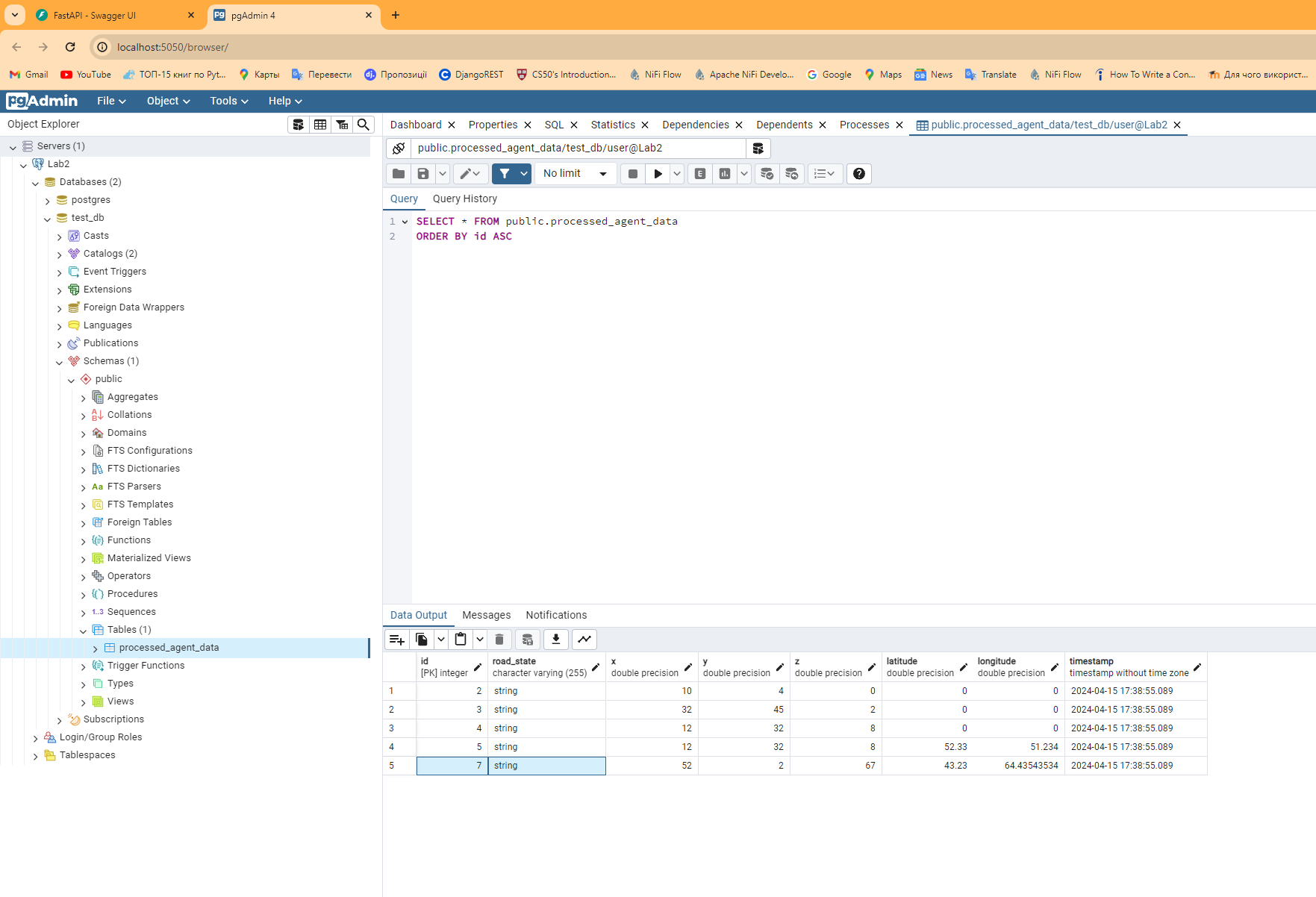
# FastAPI CRUDL endpoints  
@app.post("/processed\_agent\_data/")  
async def create\_processed\_agent\_data(data: List[ProcessedAgentData]):  
 db = SessionLocal()  
 try:  
 for item in data:  
 query = insert(processed\_agent\_data).values(  
 road\_state=item.road\_state,  
 x=item.agent\_data.accelerometer.x,  
 y=item.agent\_data.accelerometer.y,  
 z=item.agent\_data.accelerometer.z,  
 latitude=item.agent\_data.gps.latitude,  
 longitude=item.agent\_data.gps.longitude,  
 timestamp=item.agent\_data.timestamp  
 )  
 db.execute(query)  
 await send\_data\_to\_subscribers(item.json())  
 db.commit()  
 finally:  
 db.close()  
  
@app.get(  
 "/processed\_agent\_data/{processed\_agent\_data\_id}",  
 response\_model=ProcessedAgentDataInDB,  
)  
def read\_processed\_agent\_data(processed\_agent\_data\_id: int):  
 db = SessionLocal()  
 try:  
 query = select(processed\_agent\_data).where(processed\_agent\_data.c.id == processed\_agent\_data\_id)  
 data = db.execute(query).fetchone()  
 return data  
 finally:  
 db.close()  
  
@app.get("/processed\_agent\_data/", response\_model=List[ProcessedAgentDataInDB])  
def list\_processed\_agent\_data():  
 db = SessionLocal()  
 try:  
 query = select(processed\_agent\_data)  
 data = db.execute(query).fetchall()  
 return data  
 finally:  
 db.close()  
  
@app.put(  
 "/processed\_agent\_data/{processed\_agent\_data\_id}",  
 response\_model=ProcessedAgentDataInDB,  
)  
def update\_processed\_agent\_data(processed\_agent\_data\_id: int, data: ProcessedAgentData):  
 db = SessionLocal()  
 try:  
 update\_query = (  
 update(processed\_agent\_data)  
 .where(processed\_agent\_data.c.id == processed\_agent\_data\_id)  
 .values(  
 road\_state=data.road\_state,  
 x=data.agent\_data.accelerometer.x,  
 y=data.agent\_data.accelerometer.y,  
 z=data.agent\_data.accelerometer.z,  
 latitude=data.agent\_data.gps.latitude,  
 longitude=data.agent\_data.gps.longitude,  
 timestamp=data.agent\_data.timestamp  
 )  
 )  
 db.execute(update\_query)  
 db.commit()  
 updated\_data = db.execute(select(processed\_agent\_data).where(processed\_agent\_data.c.id == processed\_agent\_data\_id)).fetchone()  
 return updated\_data  
 finally:  
 db.close()  
  
@app.delete(  
 "/processed\_agent\_data/{processed\_agent\_data\_id}",  
 response\_model=ProcessedAgentDataInDB,  
)  
def delete\_processed\_agent\_data(processed\_agent\_data\_id: int):  
 db = SessionLocal()  
 try:  
 data\_to\_delete = db.execute(select(processed\_agent\_data).where(processed\_agent\_data.c.id == processed\_agent\_data\_id)).fetchone()  
 delete\_query = delete(processed\_agent\_data).where(processed\_agent\_data.c.id == processed\_agent\_data\_id)  
 db.execute(delete\_query)  
 db.commit()  
 return data\_to\_delete  
 finally:  
 db.close()

**Результат виконання:**

**Перевірка працездатності в Swagger (http://localhost:8000/docs)**

****

**Перевірка працездатності в PgAdmin (http://localhost:5050/browser/)**



**Висновок:**

Протягом лабораторної роботи ми внесли удосконалення у компонент системи, відомий як "Store", який задіяний у процесах доступу та архівації даних. Основним досягненням стало створення API, яке дозволяє користувачам через графічний інтерфейс взаємодіяти з даними в базі даних. Ця лабораторна робота надала нам цінний досвід використання таких інструментів, як Docker, PgAdmin та PostgreSQL. Окрім того, завдяки практичній роботі ми покращили свої навички у написанні об'єктно-орієнтованого коду на Python, що є важливим аспектом розвитку наших професійних компетенцій у галузі програмування та управління даними.