Національний технічний університет України «КПІ ім. Ігоря Сікорського» Факультет Інформатики та Обчислювальної Техніки



Кафедра інформаційних систем та технологій

Лабораторна робота №2

з дисципліни «Технології розробки вбудованих систем IoT»

Виконав:

студент групи IC-12 Канупа Максим

Перевірив:

Каплунов А. В.

- 1) Git repo: https://github.com/Aristocrab/Labs
- 2) CRUD-операції

```
# FastAPI CRUDL endpoints
@app.post("/processed_agent_data/")
async def create_processed_agent_data(data:
List[ProcessedAgentData]):
    query_values = []
    for item in data:
        value = {
            "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,
            "user_id": item.agent_data.user_id,
        }
        query_values.append(value)
    query =
processed_agent_data.insert().values(query_values)
    # Execute the query within a transaction
    with engine.begin() as conn:
        conn.execute(query)
    # Send data to subscribers
    for item in data:
        await
send_data_to_subscribers(item.agent_data.user_id, [{**d,
"timestamp": d["timestamp"].isoformat()} for d in
query_values])
@app.get(
    "/processed_agent_data/{processed_agent_data_id}",
    response_model=ProcessedAgentDataInDB,
def read_processed_agent_data(processed_agent_data_id: int):
```

```
# Construct a query to select data from the
processed_agent_data table based on the provided ID
    query = select(processed_agent_data).where(
        processed_agent_data.c.id == processed_agent_data_id
    # Execute the query and retrieve the data
    with engine.connect() as conn:
        result = conn.execute(query)
        data = result.fetchone()
        if data is None:
            raise HTTPException(status_code=404, detail="Data
not found")
    return data
@app.get("/processed_agent_data/",
response_model=list[ProcessedAgentDataInDB])
def list_processed_agent_data():
    # Construct a query to select data from the
processed_agent_data table
    query = select(processed_agent_data)
    # Execute the query and retrieve the data
    with engine.connect() as conn:
        result = conn.execute(query)
        data_list = result.fetchall()
    return data_list
@app.put(
    "/processed_agent_data/{processed_agent_data_id}",
    response_model=ProcessedAgentDataInDB,
def update_processed_agent_data(processed_agent_data_id: int,
data: ProcessedAgentData):
    # Construct the update query
    query = processed_agent_data.update().where(
        processed_agent_data.c.id == processed_agent_data_id
    ).values(
        road_state=data.road_state,
        x=data.agent_data.accelerometer.x,
```

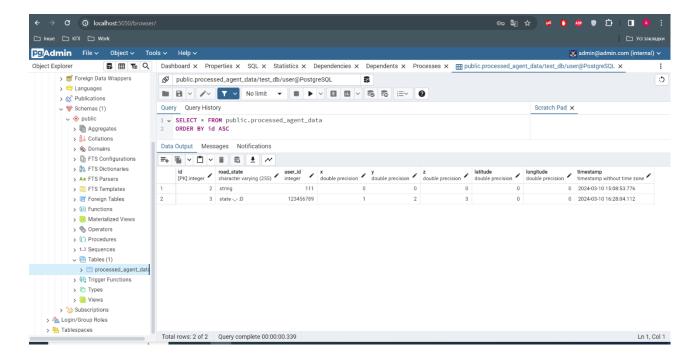
```
y=data.agent_data.accelerometer.y,
        z=data.agent_data.accelerometer.z,
        latitude=data.agent_data.gps.latitude,
        longitude=data.agent_data.gps.longitude,
        timestamp=data.agent_data.timestamp,
        user_id=data.agent_data.user_id,
    )
    # Execute the query within a transaction
    with engine.begin() as conn:
        result = conn.execute(query)
        # Check if any rows were affected (updated)
        if result.rowcount == 0:
            raise HTTPException(status_code=404, detail="Data
not found")
    return read_processed_agent_data(processed_agent_data_id)
@app.delete(
    "/processed_agent_data/{processed_agent_data_id}",
    response_model=ProcessedAgentDataInDB,
def delete_processed_agent_data(processed_agent_data_id:
int):
    processed_agent_data_to_be_removed =
read_processed_agent_data(processed_agent_data_id);
    # Construct the delete query
    query = processed_agent_data.delete().where(
        processed_agent_data.c.id == processed_agent_data_id
    )
    with engine.begin() as conn:
        result = conn.execute(query)
        # Check if any rows were affected (deleted)
        if result.rowcount == 0:
            raise HTTPException(status_code=404, detail="Data
not found")
```

3) Swagger ta PgAdmin

POST: Swagger:

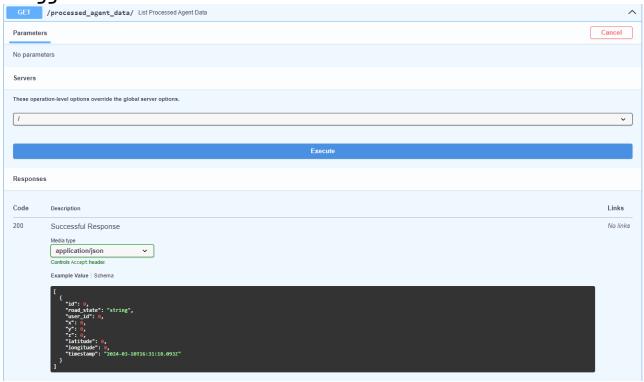
```
POST /processed_agent_data/ Create Processed Agent Data
   Request body required
                                                                                                                                                                                                                                                                                             application/json
   [
    "road_state": "state -- :D",
    "agent_data": {
        "user_id": 12345789,
        "accelerometer": {
        "x": 1,
        "y": 2,
        "z": 3
    }.
               },
"timestamp": "2024-03-10T16:28:04.112Z"
Server response
                                      length: 4
-type: application/json
un_10 Mar 2024 16:29:10 GMT
```

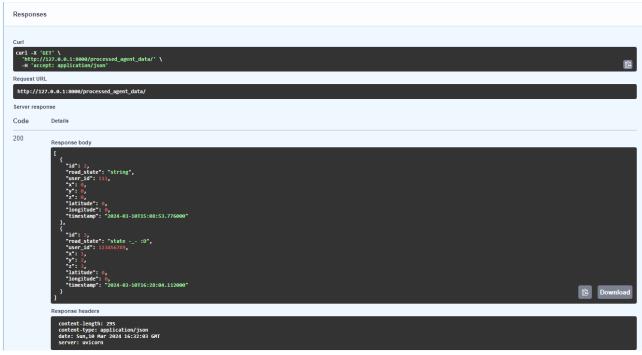
Результат у pgAdmin:



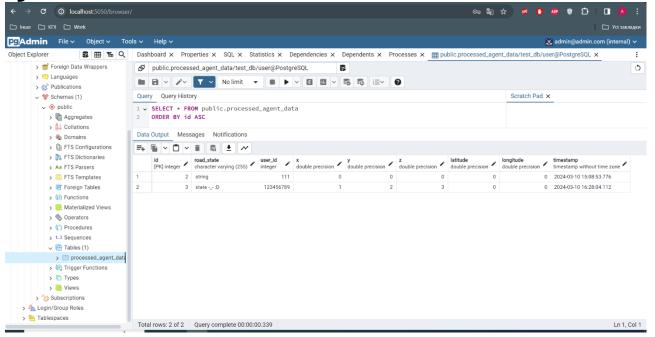
GET:

Swagger:





PgAdmin:



GET: ID

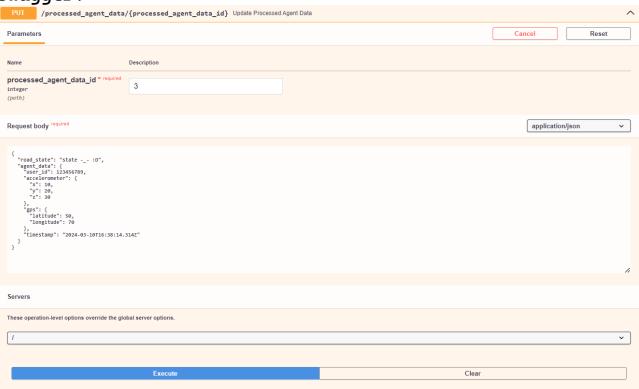
Swagger:

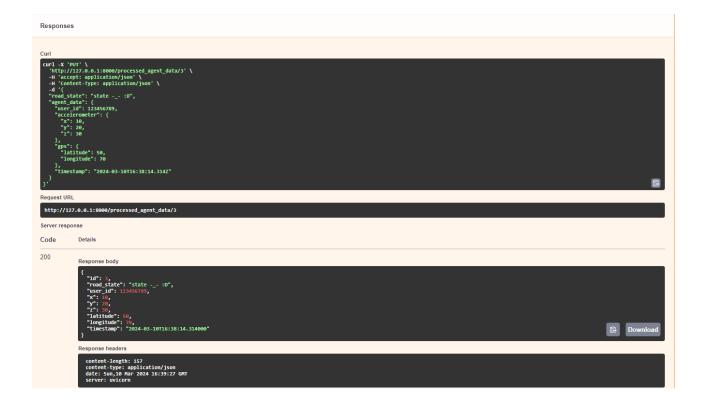
Juagger .	
GET /processed_agent_data/{processed_agent_data_id} Read Processed Agent Data	^
Parameters	Cancel
Name Description	
processed_agent_data_id * required integer (path)	
Servers	
These operation-level options override the global server options.	
	~
Execute	



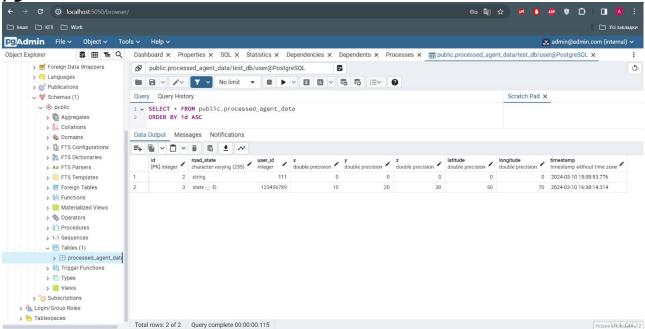
UPDATE:

Swagger:



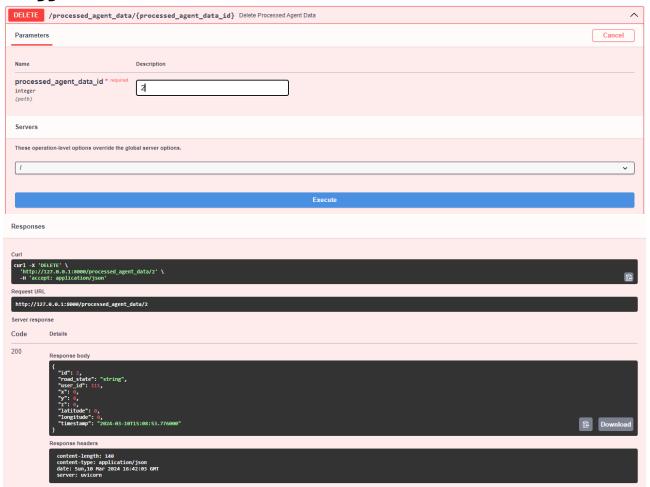


pgAdmin:

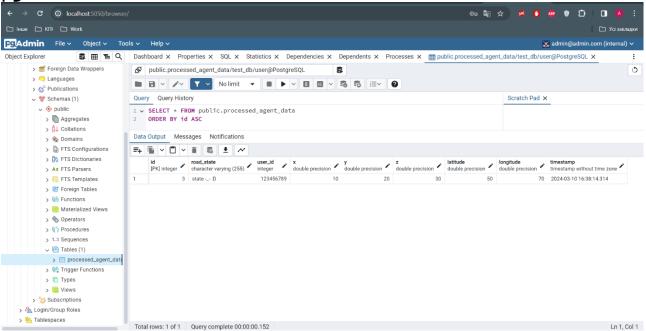


DELETE:

Swagger:



pgAdmin:



Висновок:

Під час виконання даної лабораторної роботи проведено ознайомлення з поняттям CRUD та його застосуванням. Реалізовано CRUD-операції для endpoint-a processed_agent_data мовою Python.