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

Кафедра інформаційних систем та технологій

Лабораторна робота №1

з дисципліни «Технології розробки вбудованих систем IoT»

Виконав:
студент групи ІС-12
Канупа Максим

Перевірив:
Каплунов А. В.

Київ – 2025

1) Git repo: https://github.com/Aristocrab/Labs

2) FileDatasource

```python
import config
from csv import reader
from datetime import datetime
from domain.aggregated_data import AggregatedData
from domain.accelerometer import Accelerometer
from domain.parking import Parking
from domain.gps import Gps

class FileDatasource:
    def __init__(self, filename: str) -> None:
        self.filename = filename
        self.row_counter = 0
        self.file_content = []

    def read(self):
        # infinite read
        self.row_counter %= len(self.file_content)
        data = self.file_content[self.row_counter]
        # read next row
        self.row_counter += 1

        return data

    def startReading(self, *args, **kwargs):
        with open(self.filename, "r") as file:
            self.file_content = list(reader(file))[1:]

    def stopReading(self, *args, **kwargs):
        self.row_counter = 0

class AccelerometerFileDatasource(FileDatasource):
    def __init__(self, accelerometer_filename: str) -> None:
        super().__init__(accelerometer_filename)

    def read(self) -> Accelerometer:
        data = super().read()
        return Accelerometer(
```

```
            x=int(data[0]),
            y=int(data[1]),
            z=int(data[2]),
        )

class GpsFileDatasource(FileDatasource):
    def __init__(self, gps_filename: str) -> None:
        super().__init__(gps_filename)

    def read(self) -> Gps:
        data = super().read()
        return Gps(
            latitude=float(data[0]),
            longitude=float(data[1]),
        )

class ParkingFileDatasource(FileDatasource):
    def __init__(self, parking_filename: str) -> None:
        super().__init__(parking_filename)

    def read(self) -> Parking:
        data = super().read()

        return Parking(
            empty_count=int(data[0]),
            gps=Gps(
                longitude=float(data[1]),
                latitude=float(data[2])
            )
        )


class AggregatedDatasource:
    def __init__(self, accelerometer_filename: str,
gps_filename: str) -> None:
        self.accelerometerDatasource =
AccelerometerFileDatasource(accelerometer_filename=accelerome
ter_filename)
        self.gpsDatasource =
GpsFileDatasource(gps_filename=gps_filename)

    def read(self):
```

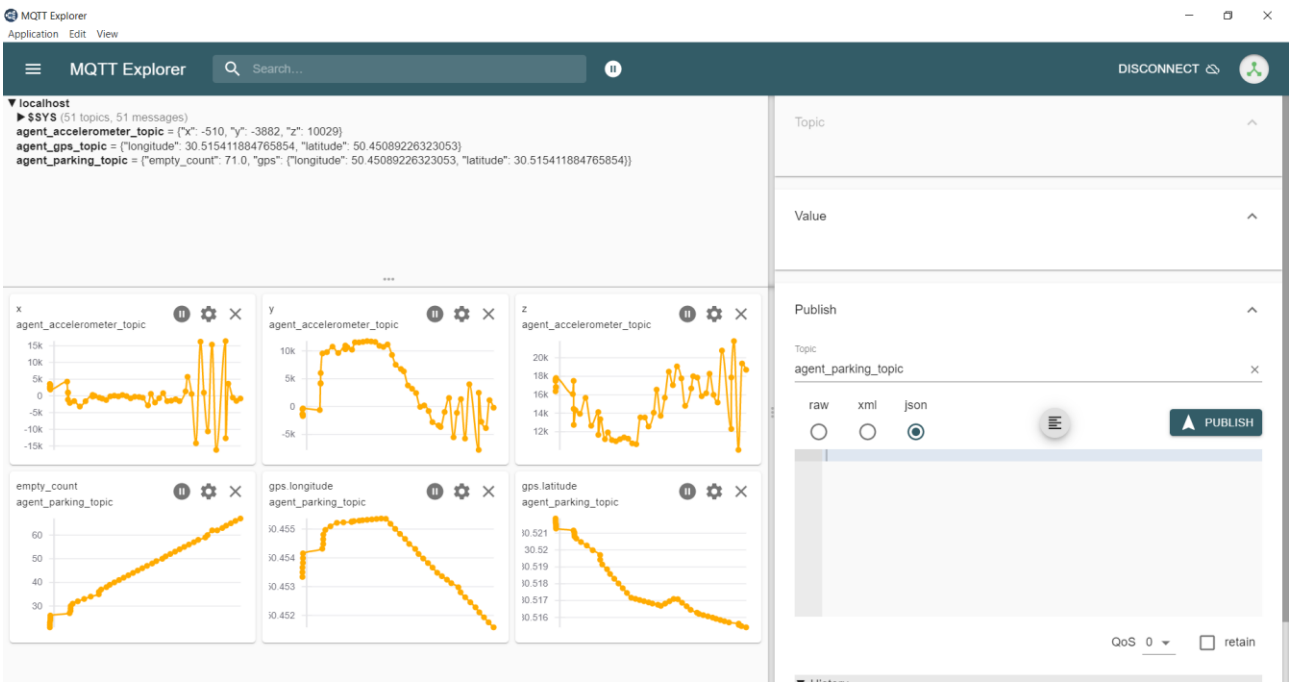
```
        acceletometerData =
self.accelerometerDatasource.read()
        gpsData = self.gpsDatasource.read()

        return
AggregatedData(accelerometer=acceletometerData, gps=gpsData,
timestamp=datetime.now(), user_id=config.USER_ID)

    def startReading(self, *args, **kwargs):
        self.accelerometerDatasource.startReading()
        self.gpsDatasource.startReading()

    def stopReading(self, *args, **kwargs):
        self.accelerometerDatasource.stopReading()
        self.gpsDatasource.stopReading()
```

3) MQTT Explorer



Висновок:

Під час виконання даної лабораторної роботи проведено ознайомлення з MQTT Explorer та його можливостями для відстеження змін даних. Реалізовано зчитування інформації з файлів та передавання її до MQTT-клієнта під різними ключами.