Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

З лабораторної роботи №5 з дисципліни

«Технології розробки вбудованих IoT системи»

„MapView”

**Перевірив**

(прізвище, ім'я, по батькові)

Київ 2023

1. Посилання на git repo: <https://github.com/faaant/iot>
2. Реалізація класу MapViewApp

import asyncio

from logging import Logger

from datasource import Datasource

from kivy.app import App

from kivy\_garden.mapview import MapMarker, MapView

from kivy.clock import Clock

from lineMapLayer import LineMapLayer

class MapViewApp(App):

    def \_\_init\_\_(self, \*\*kwargs):

        super().\_\_init\_\_()

        self.startPoint = (50.45034509664691, 30.5246114730835)

        self.badRoadPoints = []

    def on\_start(self):

        """

        Встановлює необхідні маркери, викликає функцію для оновлення мапи

        """

        self.map\_layer = LineMapLayer()

        self.map\_view.add\_layer(self.map\_layer, mode="scatter")

        self.car = MapMarker(

            lat=self.startPoint[0],

            lon=self.startPoint[1],

            source="images/car.png",

        )

        self.map\_view.add\_marker(self.car)

        self.datasource = Datasource(1)

        Clock.schedule\_interval(self.update, 1)

    def update(self, \*args):

        """

        Викликається регулярно для оновлення мапи

        """

        points = self.datasource.get\_new\_points()

        if len(points) == 0:

            return

        for point in points:

            if (point[2] != 'POTHOLE' and len(self.badRoadPoints) > 0):

                self.set\_pothole\_marker(self.badRoadPoints[int(len(self.badRoadPoints) / 2)])

                self.badRoadPoints = []

            if (point[2] == 'POTHOLE'):

                self.badRoadPoints.append(point)

            if (point[2] == 'BUMP'):

                self.set\_bump\_marker(point)

            self.map\_layer.add\_point((point[0], point[1]))

        self.update\_car\_marker(points[-1])

    def update\_car\_marker(self, point):

        """

        Оновлює відображення маркера машини на мапі

        :param point: GPS координати

        """

        self.map\_view.remove\_marker(self.car)

        self.car.lat = point[0]

        self.car.lon = point[1]

        self.map\_view.add\_marker(self.car)

    def set\_pothole\_marker(self, point):

        """

        Встановлює маркер для ями

        :param point: GPS координати

        """

        self.map\_view.add\_marker(MapMarker(

            lat=point[0],

            lon=point[1],

            source="images/pothole.png",

        ))

    def set\_bump\_marker(self, point):

        """

        Встановлює маркер для лежачого поліцейського

        :param point: GPS координати

        """

        self.map\_view.add\_marker(MapMarker(

            lat=point[0],

            lon=point[1],

            source="images/bump.png",

        ))

    def build(self):

        """

        Ініціалізує мапу MapView(zoom, lat, lon)

        :return: мапу

        """

        self.map\_view = MapView(

            zoom=15,

            lat=self.startPoint[0],

            lon=self.startPoint[1],

        )

        return self.map\_view

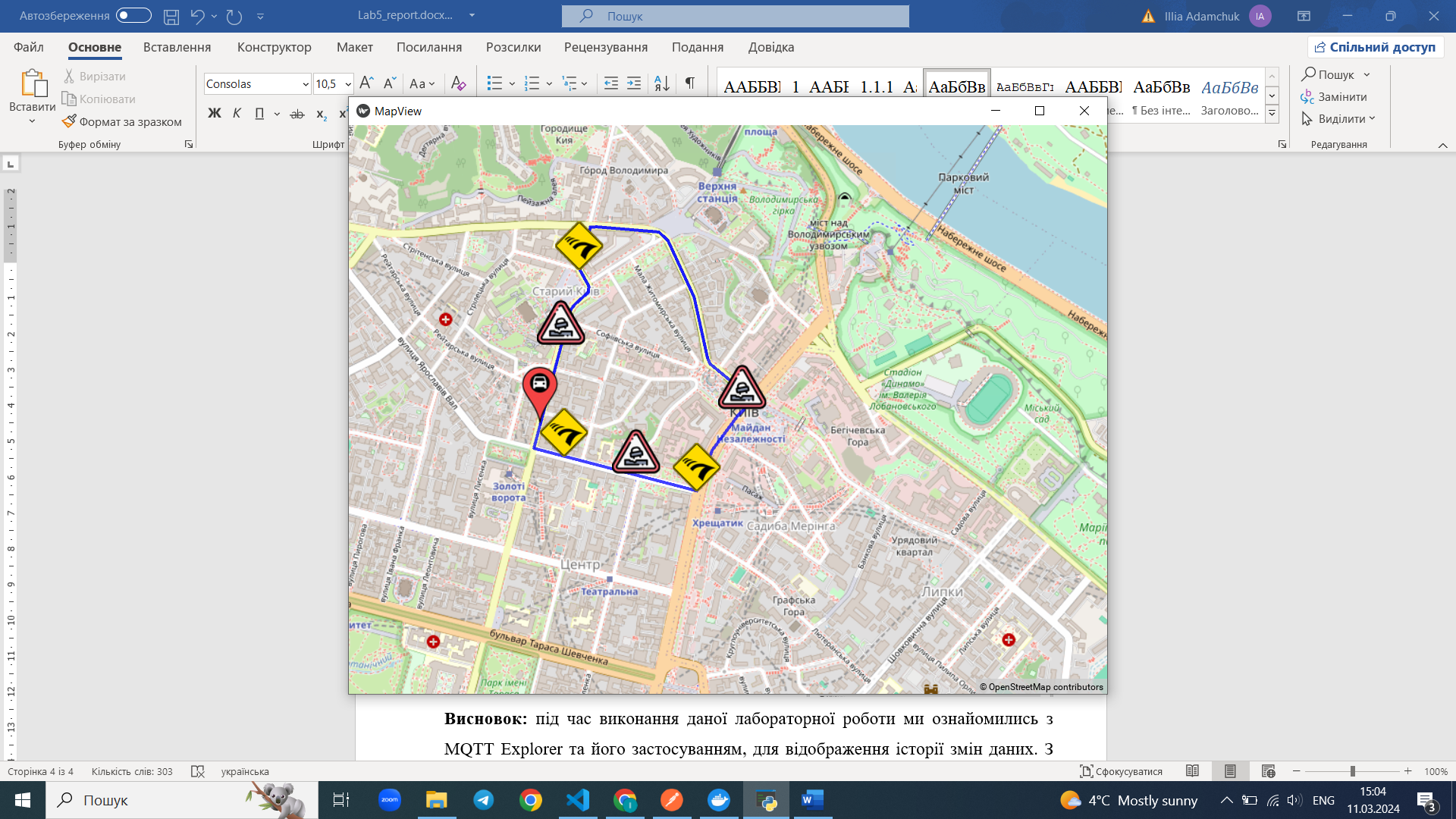
if \_\_name\_\_ == "\_\_main\_\_":

    loop = asyncio.get\_event\_loop()

    loop.run\_until\_complete(MapViewApp().async\_run(async\_lib="asyncio"))

    loop.close()

1. Результати запуску з попередніми лабораторними



**Висновок:** під час виконання даної лабораторної роботи ми об’єднали всі частини проекту та виконали їх налаштування, щоб вони працювали як одна система. Було реалізовано систему аналізу якості дорожнього покриття, з відображенням ямок та лежачих поліцейських.