

Національний університет «Одеська політехніка»
Інститут комп'ютерних систем
Кафедра інформаційних систем

КУРСОВА РОБОТА

з дисципліни «Об'єктно-орієнтоване програмування»

Тема: «Розробка веб-додатка для відстеження тренувань та їх візуалізації на
мапі»

Студентка __2__ курсу AI-225 групи
Спеціальності 122 – «Комп'ютерні науки»

_____ Аврамова Р. Р.

(прізвище та ініціали)

Керівник ст.викл. к.т.н. Годовіченко М.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____

Оцінка: ECTS _____

Члени комісії

_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)
_____	_____
(підпис)	(прізвище та ініціали)

Одеса – 2024

ЗМІСТ

ЗАВДАННЯ НА КУРСОВУ РОБОТУ	3
ВСТУП	5
1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT	6
1.1 Теоретичні відомості	6
2 ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ	8
2.1 Програмна реалізація ООП	8
3 ІНСТРУКЦІЯ КОРИСТУВАЧА	12
3.1 Інструкція користувача	12
ВИСНОВКИ.....	16
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	17
ДОДАТОК А.....	17

Національний університет «Одеська політехніка»

Інститут комп'ютерних систем

Кафедра інформаційних систем

ЗАВДАННЯ НА КУРСОВУ РОБОТУ

1. Розробка інтерфейсу користувача, включаючи форму для додавання нових тренувань.
2. Інтеграція з картографічною бібліотекою для відображення тренувань на мапі.
3. Реалізація геолокації для визначення поточного місцезнаходження користувача.
4. Збереження тренувань у локальному сховищі браузера для подальшого відображення та відновлення.
5. Обробка подій користувача, таких як кліки на тренуваннях для переходу до відповідного місця на мапі.
6. Розрахунок додаткових параметрів тренувань, таких як швидкість чи темп.
7. Валідація введених даних користувачем для уникнення некоректних записів тренувань.
8. Оптимізація відображення тренувань на мапі для забезпечення ефективної роботи додатка.
9. Розгортання додатка на веб-сервері для загального доступу.

АНОТАЦІЯ

Mapty - це веб-додаток, призначений для відстеження тренувань та їх візуалізації на мапі. Додаток дозволяє користувачам додавати нові тренування (біг чи їзда на велосипеді) з вказанням дистанції, тривалості та інших параметрів. Тренування відображаються на мапі, дозволяючи користувачам переглядати їх місцеположення та деталі. Додаток також забезпечує можливість збереження тренувань у локальному сховищі браузера для подальшого відображення та відновлення при перезавантаженні сторінки. Взаємодія з користувачем реалізована через інтерактивний інтерфейс, який дозволяє додавати, переглядати та редагувати тренування.

ABSTRACT

This paper presents an overview of the Mapty web application, which enables users to track their workouts and visualize them on a map interface. Mapty follows a simpler structure that prioritizes functionality and code simplicity.

The application consists of three main components: the Model, View, and Controller. The Model handles workout data representation, including details like workout type, duration, distance, and location coordinates. The View component renders workout data onto the user interface, presenting it in a visually appealing manner on the map interface. The Controller module manages user inputs, coordinates data flow between the Model and View, and ensures a seamless user experience.

While Mapty does not strictly adhere to MVC principles, it leverages Object-Oriented Programming (OOP) principles in JavaScript to enhance code organization, readability, and maintainability. This approach simplifies code management, promotes reusability, and facilitates the implementation of new features.

Overall, Mapty provides users with a straightforward yet effective solution for tracking workouts and visualizing them on a map interface, demonstrating the practical application of Object-Oriented Programming principles in web development.

ВСТУП

Мета нашої курсової роботи полягає у засвоєнні та поглибленні знань, набутих під час вивчення курсу "Об'єктно-орієнтоване програмування", а також у розвитку навичок вибору оптимальних способів представлення даних та удосконаленні технік тестування та налагодження програм. Було використано JavaScript для створення веб-додатку Marty.

Важливу роль у створеному проекті відіграють динамічні структури даних, які дозволяють ефективно зберігати та опрацьовувати інформацію в залежності від потреб користувачів. Використано різноманітні об'єкти та масиви для зберігання даних про тренування та іншу інформацію, що забезпечує зручний та легкий доступ до неї.

Підхід до програмування базується на об'єктно-орієнтованому підході, де динамічні структури даних реалізовані у вигляді класів та об'єктів. Це допомагає створити чітку та організовану структуру програми та спрощує роботу з даними. Важливі етапи розробки веб-додатку Marty включають аналіз вимог, проектування архітектури, програмування, тестування та впровадження продукту. Після впровадження необхідно відслідковувати зворотний зв'язок від користувачів та вчасно вносити необхідні зміни. Також важливо правильно документувати програмний продукт для подальшої підтримки та розвитку.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО КЛАСИ ТА ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ В JAVASCRIPT

1.1 Теоретичні відомості

Об'єктно-орієнтоване програмування (ООП) є одним із найпоширеніших підходів у сучасному програмуванні, а JavaScript, який використовується для створення інтерактивних та динамічних веб-додатків, не є винятком. ООП дозволяє створювати код, який є більш структурованим, повторно використовуваним та підтримуваним. Одними з основних концепцій ООП є класи та об'єкти.

– Класи - це шаблони або візитівки для створення об'єктів. Вони описують структуру та поведінку об'єктів, які будуть створені на їх основі.

У JavaScript класи можна створювати за допомогою ключового слова `'class'`. Вони можуть містити властивості та методи, які характеризують об'єкти цього класу.

Приклад створення класу в JavaScript:

```
class Car {  
  constructor(make, model, year) {  
    this.make = make;  
    this.model = model;  
    this.year = year;  
  }  
  
  getInfo() {  
    return `${this.make} ${this.model} (${this.year})`;  
  }  
}
```

– Об'єкти - це екземпляри класів, створені за допомогою конструкторів класів. Вони представляють конкретні екземпляри даних, які відповідають певному класу.

У JavaScript об'єкти можна створювати за допомогою ключового слова `'new'`, викликаючи конструктор класу.

Приклад створення об'єкта в JavaScript:

```
const myCar = new Car('Toyota', 'Corolla', 2020);
```

Переваги об'єктно-орієнтованого програмування:

- Модульність: Програма поділяється на окремі класи та об'єкти, що полегшує управління та розуміння коду.
- Повторне використання коду: Класи можна використовувати для створення нових об'єктів з аналогічною функціональністю без необхідності повторного написання коду.
- Розширюваність: Нові класи та функціональність можуть бути легко додані та розширені, не змінюючи вже існуючий код.

Використання класів та об'єктів у JavaScript дозволяє розробникам створювати більш структуровані, масштабовані та підтримувані програми, що є ключовими для успішного створення складних веб-додатків.

2 ПРОГРАМНА РЕАЛІЗАЦІЯ ООП У ПРОЕКТІ

2.1 Програмна реалізація ООП

У нашому проєкті, ми використовуємо принципи ООП для створення структурованого та ефективного коду, який дозволяє керувати динамічними структурами даних та забезпечує зручний інтерфейс для користувачів. Основними аспектами програмної реалізації ООП у нашому проєкті є використання класів та об'єктів для представлення різних об'єктів нашого додатку та їх взаємодії.

Класи:

- У Марті ми використовуємо класи для моделювання різних типів тренувань, таких як біг або велосипед, а також для представлення різних елементів інтерфейсу користувача, таких як форма вводу даних та списки тренувань.
- Наприклад, у нашому коді ми можемо мати класи Workout, Running та Cycling, які відповідають за моделювання тренувань різних типів.

Об'єкти:

- За допомогою класів ми створюємо об'єкти, які представляють конкретні екземпляри тренувань або інтерфейсні елементи.
- Наприклад, можемо створити об'єкт конкретного тренування, такий як "біг півгодини", який буде екземпляром класу Running.

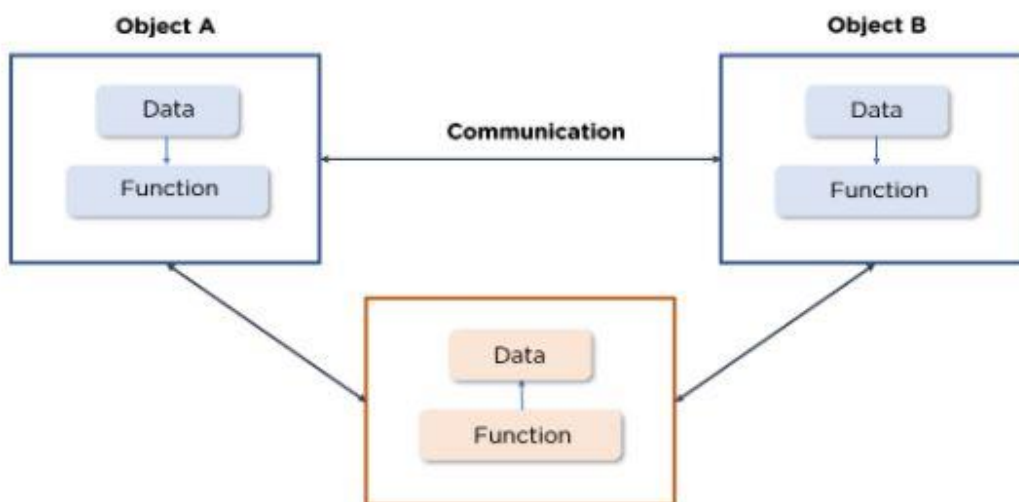


Рисунок 2.1 – Basic concept of classes in Javascript

Опис структури проекту Marty.

Javascript:

1. `_getPosition()`: Ця функція використовується для отримання поточного місцезнаходження користувача за допомогою геолокації. Якщо браузер підтримує геолокацію, вона викликає метод `_loadMap()` для завантаження карти з поточними координатами користувача.

2. `_loadMap(position)`: Ця функція завантажує карту на сторінку з використанням отриманих координат користувача. Вона встановлює вид карти та додає обробник подій для можливості додавання тренувань на карту.

3. `_showForm(mapE)`: Ця функція відображає форму для введення даних про тренування при натисканні на карті. Параметр `mapE` містить дані події клікання на карті.

4. `_hideForm()`: Ця функція приховує форму після введення даних про тренування та очищує всі поля форми.

5. `_toggleElevationField()`: Ця функція перемикає відображення поля для введення висоти під час вибору типу тренування. Якщо обрано тип "біг", поле для введення висоти стає прихованим, а для велосипеда - видимим.

6. `_newWorkout(e)`: Ця функція викликається при натисканні на кнопку "Зберегти" у формі для введення даних про тренування. Вона перевіряє введені

дані, створює новий об'єкт тренування типу Running або Cycling в залежності від обраного типу тренування, додає його до масиву тренувань, відображає на карті та у списку тренувань, та зберігає в локальне сховище.

7. `_renderWorkoutMarker(workout)`: Ця функція створює маркер на карті для позначення місця проведення тренування. Вона призначає маркеру відповідну підказку та відкриває її для користувача.

8. `_renderWorkout(workout)`: Ця функція відображає інформацію про тренування в списку на сторінці. Вона генерує HTML-код для нового елемента списку на основі даних про тренування та додає його до HTML-структури.

9. `_moveToPopup(e)`: Ця функція переміщує вид карті до місця проведення конкретного тренування при натисканні на його елемент у списку. Вона знаходить відповідне тренування за його ідентифікатором та переміщує вид карті до його координат.

10. `_setLocalStorage()`: Ця функція зберігає дані про всі тренування в локальне сховище браузера у форматі JSON.

11. `_getLocalStorage()`: Ця функція завантажує дані про тренування з локального сховища браузера та відображає їх на сторінці при завантаженні додатку.

12. `reset()`: Ця функція очищує локальне сховище браузера від даних про тренування та перезавантажує сторінку.

CSS:

Стилі, наведені вище, визначають зовнішній вигляд веб-додатку і його компонентів. Ось пояснення кожного блоку стилів:

1. `:root`: Визначення кореневих змінних для кольорів, які використовуються в усьому додатку.

2. `*`: Стилзація для всіх елементів, зняття відступів та відступів, встановлення `box-sizing` на `border-box`.

3. `html`: Встановлення базового розміру шрифту для зручного використання `rem`.

4. `body`: Загальні стилі для тіла документа, такі як шрифт, колір тексту, фон та інші.
5. `a:link`, `a:visited`: Стилi для посилань.
6. `sidebar`: Стилi для бiчного меню, такі як фон, відступи та кольори тексту.
7. `workouts`: Стилi для списку тренувань, з встановленням висоти, прокрутки та відображення scrollbar.
8. `workout`: Стилi для окремих елементів списку тренувань, такі як фон, округлення кутів та відступи.
9. `form`: Стилi для форми введення даних про тренування, такі як фон, округлення кутів та відступи.
10. `form.hidden`: Стилi для прихованої форми з використанням зміщення та зміни висоти.
11. `form__row`: Стилi для рядка в формі з встановленням відступів та вирівнювання елементів.
12. `form__label`, `.form__input`: Стилi для міток та введення в формі.
13. `#map`: Стилi для відображення карти, встановлення фону та вирівнювання.
14. `.leaflet-popup`: Стилi для впливаючих вікон на карті, з встановленням фону, колірiв та округлення кутів.

3 ІНСТРУКЦІЯ КОРИСТУВАЧА

3.1 Інструкція користувача

Першочергово, після переходу на сайт у користувача з'являється запит на дозвіл розташування, що в подальшому забезпечить відображення мапи та взаємодія з нею (рис. 3.1).

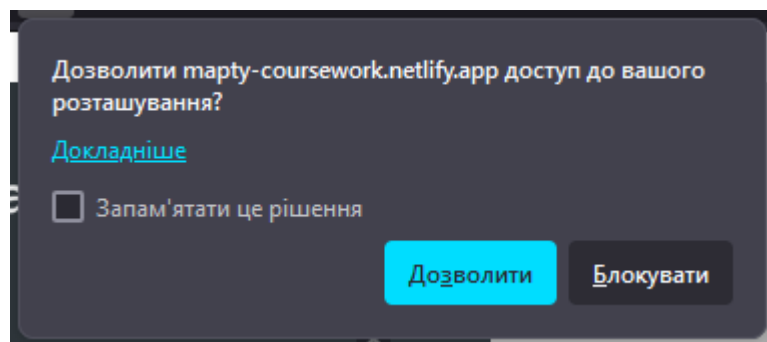


Рисунок 3.1 – Вебсайт запрошує дозвіл на розташування

Після дозволу користувач бачить початковий екран нашого вебсайту з картою, де відображається район проживання користувача (рис. 3.2).

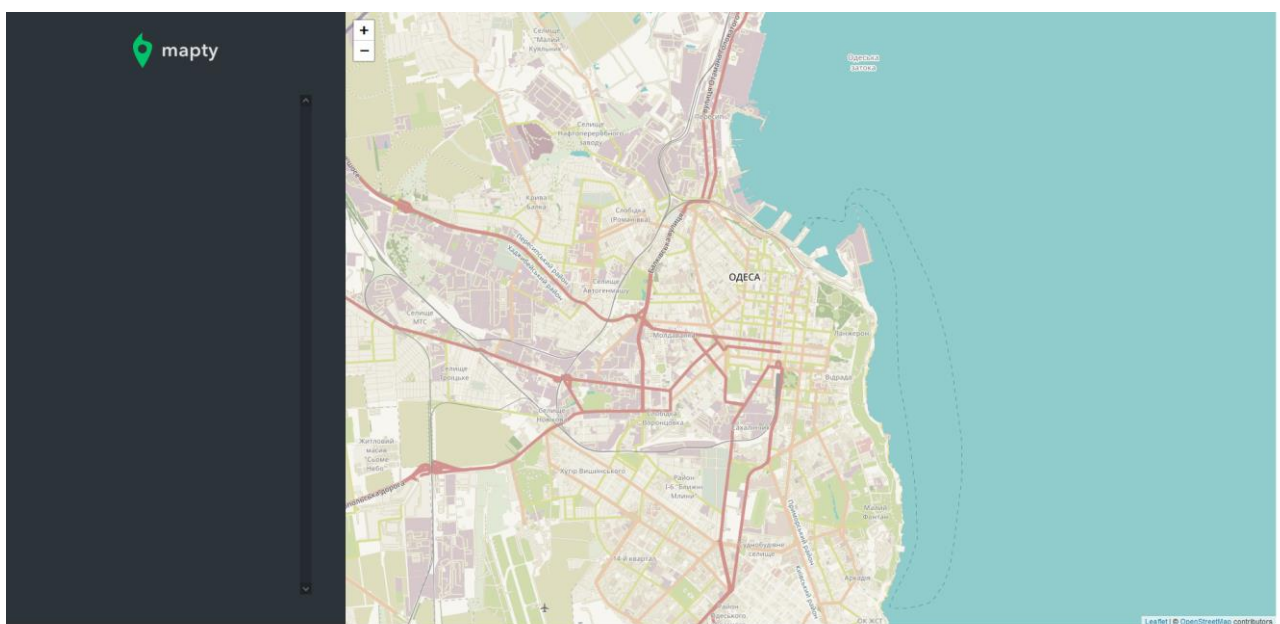


Рисунок 3.2 – Початковий вид вебсайту

Після кліку на карту з'являється форма для заповнення, яка дозволить залишити мітку в обраному місці (рис 3.3).

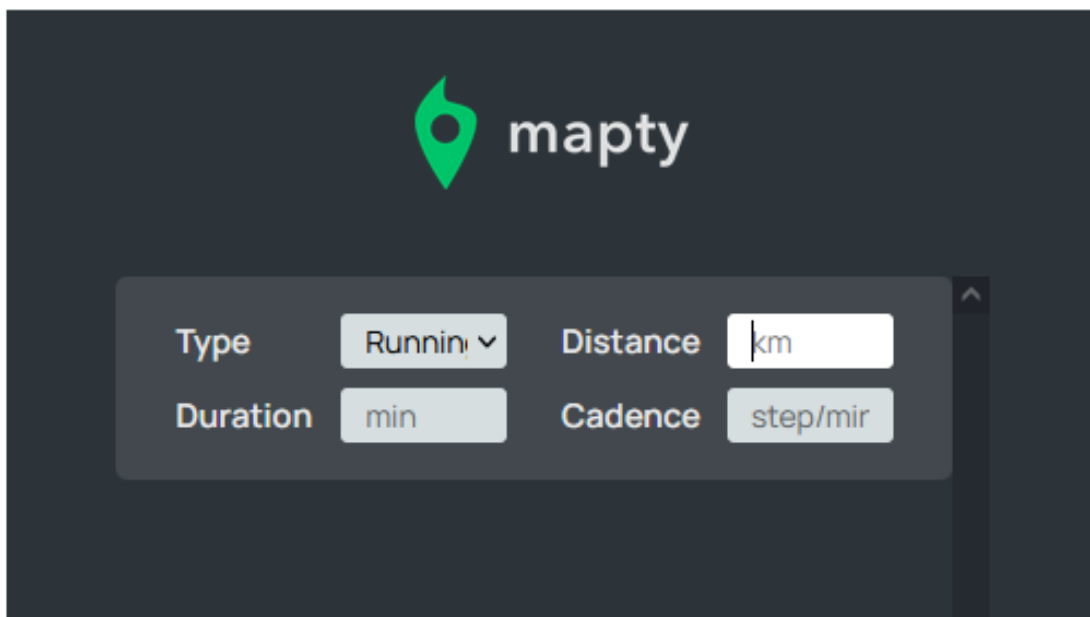
The image shows a dark-themed user interface for 'mapty'. At the top center is a green location pin icon followed by the text 'mapty'. Below this is a light gray rounded rectangle containing four input fields. The first row has 'Type' with a dropdown menu showing 'Running' and a right-pointing arrow, and 'Distance' with a text input field containing 'km'. The second row has 'Duration' with a text input field containing 'min', and 'Cadence' with a text input field containing 'step/mir'. A small upward-pointing arrow is visible on the right side of the form.

Рисунок 3.3 – Форма для заповнення тренування

На формі розміщені опція типу тренування (Running або Cycling) та поля для введення значень (рис. 3.4).

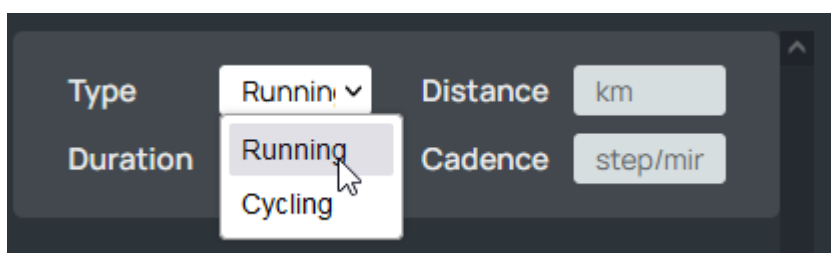
This image is a close-up of the 'Type' dropdown menu from the previous screenshot. The dropdown is open, showing two options: 'Running' and 'Cycling'. A mouse cursor is hovering over the 'Running' option, which is highlighted with a light gray background. The 'Cycling' option is below it. The rest of the form fields ('Distance', 'Duration', 'Cadence') are partially visible in the background.

Рисунок 3.4 – Користувач може обрати тип тренування

Після заповнення форми користувач може побачити свою мітку на мапі (рис. 3.5).

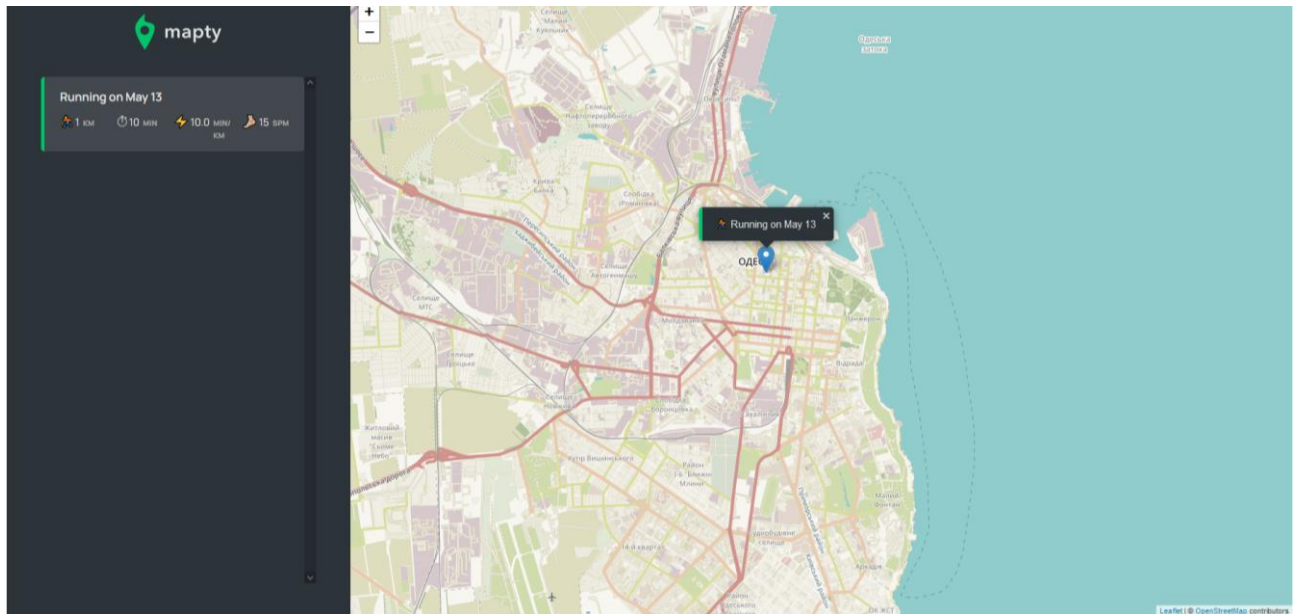


Рисунок 3.5 – Створена мітка на мапі

Користувач може додавати будь-яку бажану кількість міток з різними даними, і інтерфейс зручно відображає відмінності міток за допомогою кольорів, емодзі і безпосередь датах на мітках (рис. 3.6).

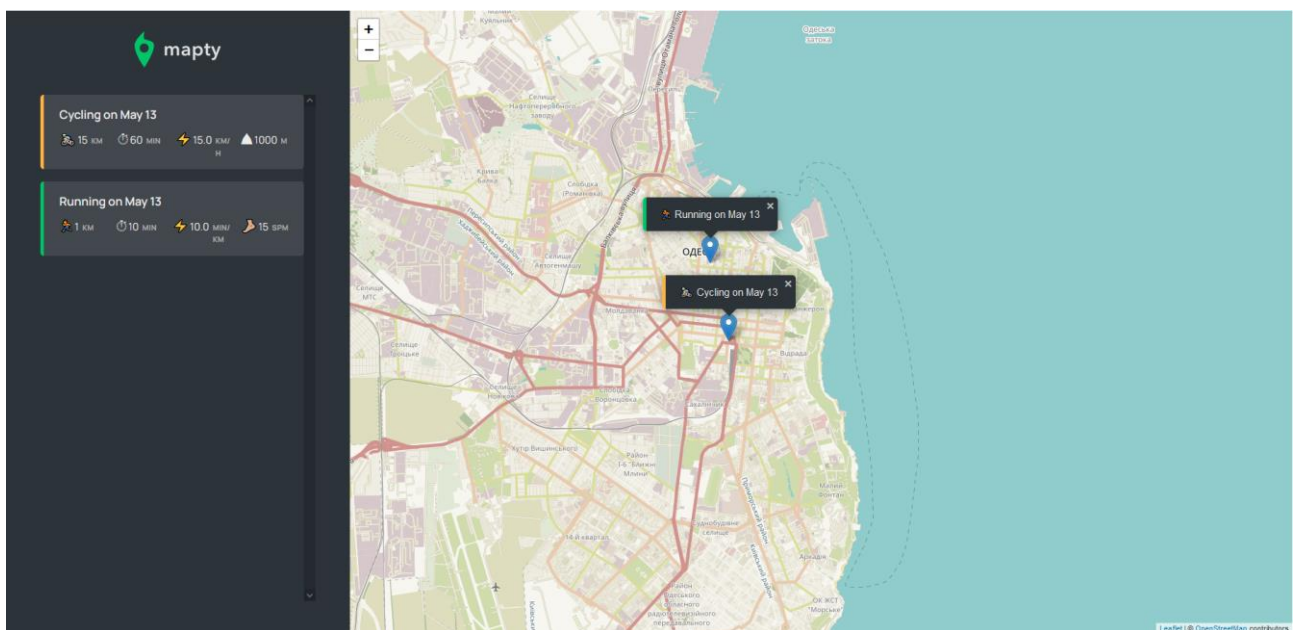


Рисунок 3.6 – Користувач може відрізнати свої типи тренування по кольору, емодзі та даті

Користувач може натискати на тренування на панелі зліва та карта відобразить у якій місцевості це тренування відбувалось, переносячи користувача до цієї мітки. Надпис над міткою можна приховати, натиснувши на іконку хрестика (рис. 3.7).

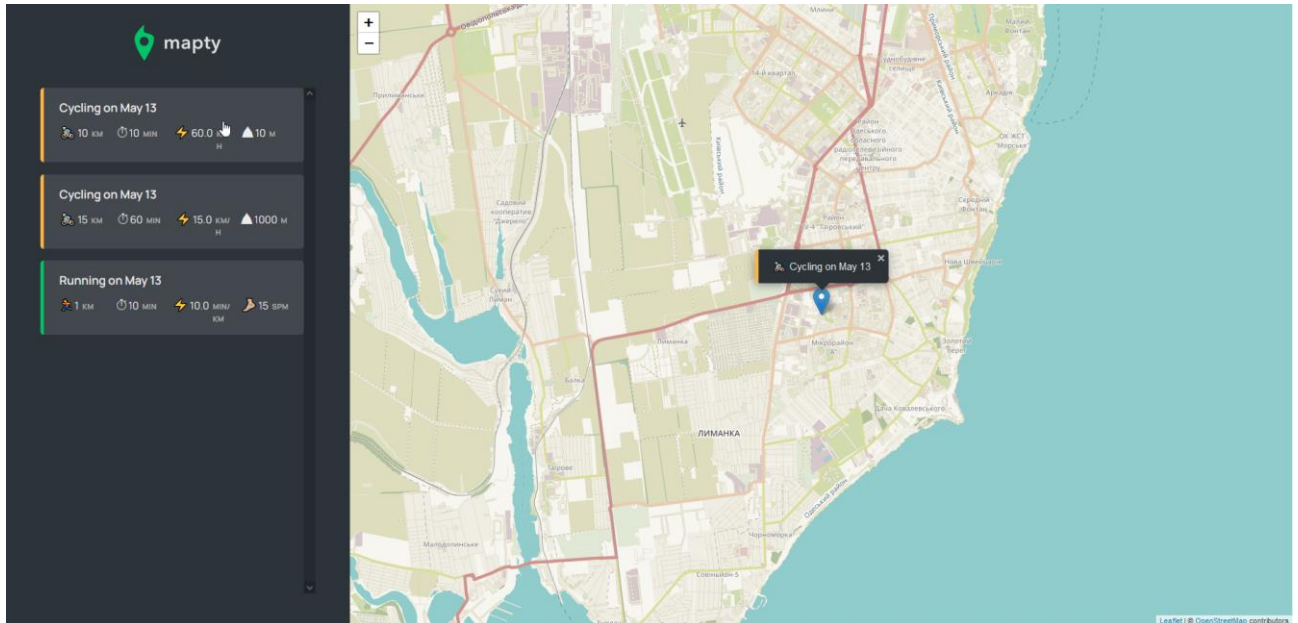


Рисунок 3.7 – Відображення та переміщення між створеними мітками на мапі

При бажанні користувач може збільшувати чи зменшувати карту (рис. 3.8).

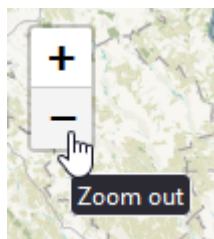


Рисунок 3.8 – Збільшення та зменшення карти

ВИСНОВКИ

Проект Marpu відображає нашу здатність до створення функціональних веб-додатків з допомогою JavaScript. У процесі розробки ми створили набір функцій, які дозволяють користувачам легко додавати нові тренування, вибираючи між бігом та велосипедом, вказуючи дистанцію, тривалість, темп бігу або швидкість їзди. Ми також реалізували інтерактивну карту, яка відображає всі тренування користувача. Ця функціональність забезпечує зручну взаємодію з додатком та можливість візуального відстеження прогресу.

Крім того, ми провели оптимізацію коду для покращення швидкодії веб-сайту. Це включало в себе використання ефективних алгоритмів обробки даних, мінімізацію зайвих запитів до сервера та оптимізацію завантаження ресурсів. Також застосували кешування для прискорення завантаження сторінок та покращення користувацького досвіду. Ці оптимізації дозволили зменшити час завантаження сторінок і покращити реактивність веб-додатку, що сприяє зручній та ефективній роботі користувачів з ним.

Перевірка швидкодії вебсайту за допомогою Chrome LightHouse:

Можемо побачити, що доступність нижче 90, причиною цього є API карти, завантаження додаткових ресурсів, які використовуються для відображення карт, може призвести до збільшення часу завантаження сторінки, а також вплинути на швидкодію та продуктивність веб-сайту. Попри це, інші показники високі, що показує гарну роботу вебсайту (рис. 3.9).

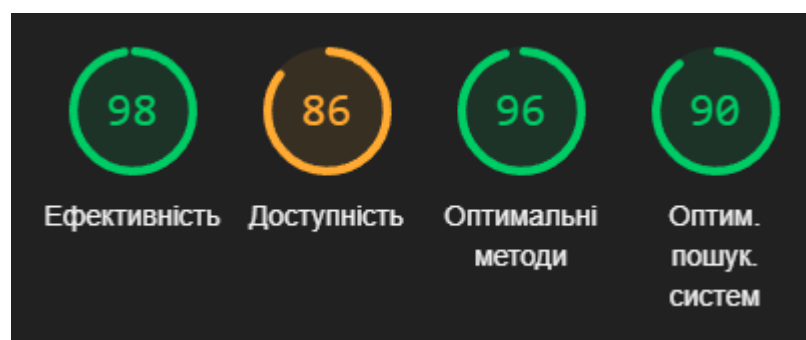


Рисунок 3.9 – Швидкодія створеного вебсайту

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Leaflet Map, URL: <https://leafletjs.com/>
2. OOP Js, URL: <https://www.simplilearn.com/tutorials/javascript-tutorial/oop-in-javascript>
3. MDN Documentation(Classes), URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Classes>
4. CodeCamp Blog about OOP JS , URL: <https://www.freecodecamp.org/news/object-oriented-programming-javascript/>

ДОДАТОК А

Код програми:

<https://github.com/asssslay/Mapty>