МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ

Циклова комісія (кафедра) комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

ВЕБСАЙТ ДЛЯ ОРГАНІЗАЦІЇ ТА УПРАВЛІННЯ СПОРТИВНИМИ ЗМАГАННЯМИ

Виконав: студент групи 1П-21

Спеціальності

121 Інженерія програмного забезпечення

Антон ГОЛЕЦЬ

Керівник:

Вікторія НЕМЧЕНКО

Черкаси 2025

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

Кафедра комп'ютерної інженерії та інформаційних технологій

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри КІ та ІТ

_____ Владислав ХОТУНОВ
(підпис)
«_____»_____2024р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Гольцю Антону Вікторовичу

1. Тема випускної роботи «Вебсайт для організації та управління спортивними змаганнями»

Керівник роботи _Немченко Вікторія Юріївна, викладач другої категорії

затверджені наказом закладу вищої освіти від "07" жовтня 2024 року №68у.
2. Строк подання студентом випускної роботи 02.06.2025р.
3. Вихідні дані до випускної роботи потреба в інструменті для організації спортивних заходів; відсутність універсальних вебплатформ; необхідність зберігання інформації про учасників, змагання, результати та забезпечення доступу до неї; вимоги до створення сучасного вебінтерфейсу з базою даних, засобами авторизації та можливістю створення/публікації змагань.
4. Зміст випускної роботи (перелік питань, які потрібно розробити) огляд існуючих рішень для організації спортивних змагань, визначення вимог до вебсайту для управління спортивними змаганнями, розробка архітектури вебдодатку, дизайн інтерфейсу користувача, реалізація функціональності управління змаганнями.
5. Дата видачі завдання _02.10.2024р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Терміни виконання етапів	Примітка про виконання з підписами керівника і студента
1	Вступ	14.10.2024	
2	Розділ 1. АНАЛІЗ ТЕМАТИКИ ТА ПОСТАНОВКА ЗАДАЧІ	9.01.2025	
3	Розділ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ВЕБСАЙТУ	10.03.2025	
4	Розділ 3. ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ	28.04.2025	
5	Висновки	12.05.2025	
6	Оформлення випускної роботи (чистовий варіант)	26.05.2025	
7	Здача випускної роботи на кафедру для рецензування (за 14 днів до захисту)	30.05.2025	
8	Перевірка випускної роботи на наявність ознак плагіату (за 10 днів до захисту)	02.06.2025	
9	Подання випускної роботи на затвердження завідувачу кафедри (за 7 днів до захисту)	10.06.2025	

Студент _____ (підпис) Антон ГОЛЕЦЬ

Керівник роботи _____ (підпис) Вікторія НЕМЧЕНКО

АНОТАЦІЯ

Кваліфікаційна робота «Розробка вебсайту для організації та управління спортивними змаганнями» присвячений процес створення вебсайту здебільшого для морського багатоборства та триатлону.

Метою кваліфікаційної роботи є надання користувачам можливості переглядати та створювати змагання, реєструватися як учасники, а також організаторам змагань створювати та керувати спортивними заходами. Описати структуру сайту, основні технології розробки, архітектуру бази даних і результати тестування. Отримати вебсайт, щоб він був базовою платформою для подальшої розробки та удосконалення для кращого проведення спортивних змагань.

Організація спортивних змагань вимагає значних адміністративних ресурсів, особливо при організації багатьох заходів одночасно, тому вебсайт для організації та управління спортивними змаганнями зможе допомогти вирішити проблему.

Створення вебплатформи дозволить автоматизувати процеси управління змаганнями з морського багатоборства та триатлона, сприятиме ефективності в організації спортивних заходів.

ANNOTATION

The qualification work "Development of a website for the organization and management of sports competitions" is devoted to the process of creating a website mainly for marine all-around and triathlon.

The purpose of the qualification work is to provide users with the ability to view and create competitions, register as participants, as well as competition organizers to create and manage sports events. Describe the structure of the site, basic development technologies, database architecture and testing results. Get the website to be a basic platform for further development and improvement for better sports competitions.

The organization of sports competitions requires significant administrative resources, especially when organizing many events at the same time, so a website for organizing and managing sports competitions can help solve the problem.

The creation of a web platform will allow you to automate the processes of managing marine all-around and triathlon competitions, and will contribute to the efficiency of organizing sports events.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ІТ – інформаційні технології.

БД – база даних.

ОС – операційна система.

ПЗ – програмне забезпечення.

API (application programming interface, програмний інтерфейс додатка*)* – опис методів та можливостей взаємодії однієї програми з іншою.

AI (Artificial Intelligence) – штучний інтелект.

ЗМІСТ

ВСТУП 3

РОЗДІЛ 1. АНАЛІЗ ТЕМАТИКИ ТА ПОСТАНОВКА ЗАДАЧІ 4

1.1. Огляд існуючих рішень для організації спортивних змагань 4

1.2 Визначення вимог вебсайту для управління спортивними змаганнями . 10

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ВЕБСАЙТУ 11

2.1. Розгляд архітектури вебдодатку 11

2.2. Дизайн інтерфейсу 13

2.3. Налаштування середовища розробки 24

РОЗДІЛ 3. ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ 28

3.2. Аналіз результатів тестування 33

3.3. Рекомендації щодо впровадження та використання 34

ВИСНОВКИ 35

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 36

ДОДАТКИ 38

ВСТУП

Задачі, які вирішуватимуться: створення та керування змаганнями з видів спорту як біг, плавання, велоспорт, триатлон, вітрила та веслування; учасники змагань зможуть подивитися свої результати на вебсайті.

Ступінь дослідження проблеми: на сьогодні існує багато досліджень у сфері автоматизації спортивних процесів, проте більшість з них орієнтовані не на всі види спорту. Менш вивченими залишаються змагання, які потребують доступних та гнучких інструментів: парус та веслування.

Актуальність теми: організація спортивних змагань вимагає значних адміністративних ресурсів, особливо при організації багатьох подій одночасно. Онлайн-рішення, які є наразі, для управління змаганнями не підходять для української аудиторії користувачів, так як не мають ні локалізації, ні місць проведень в Україні. Створення вебплатформи дозволить автоматизувати процеси управління змаганнями з морського багатоборства та триатлона, сприятиме ефективності в організації спортивних заходів.

Мета проєкту: розробити вебсайт, який дозволить організаторам змагань зручно та легко створювати змагання з різних видів спорту, переглядати та вносити результати учасників, дозволить учасникам переглядати свої результати та інших учасників змагань.

Об’єкт дослідження: організація спортивних змагань, включаючи реєстрацію учасників, планування подій, фіксацію результатів і їхньої публікації.

Предмет дослідження: автоматизована система управління спортивними змаганнями як вебплатформи.

Апробація роботи: часткові результати кваліфікаційної роботи були апробовані на XVII Студентській науково-практичній конференції студентів, аспірантів та молодих вчених «ІТенденції розвитку ІТ-технологій в Україні» [ ].

РОЗДІЛ 1.
АНАЛІЗ ТЕМАТИКИ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Огляд існуючих рішень для організації спортивних змагань

Для організації спортивних змагань, необхідні ефективні інструменти для управління учасниками, розкладами та результатами. На сьогодні існує багато програмних рішень, які допомагають автоматизувати ці процеси але деякі з них не підтримують українську мову та не мають можливості реєстрації користувачів.

1.1.1 Платформи для організації змагань з бігу

Run Ukraine – організовує наймасштабніші бігові події України: Київський марафон, Recruit Run, забіги на 10 км тощо (https://runukraine.org).

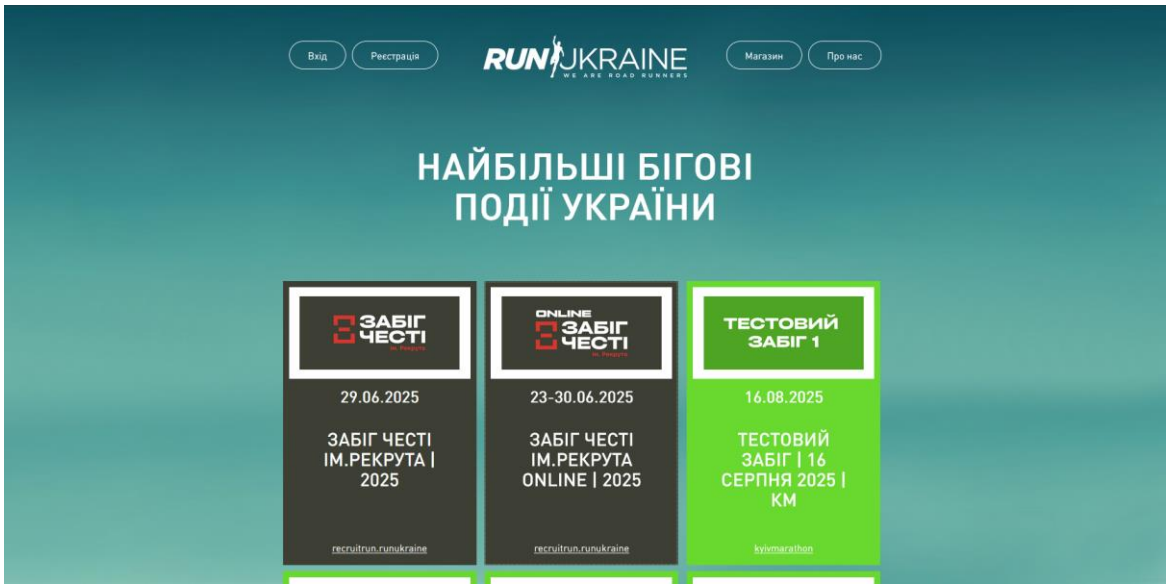

Рисунок 1.1 – Головна сторінка сайту «Run Ukraine»

Недоліки: на сайті немає персоналізованого профілю або кабінету (користувач не може зберегти історію участі або бачити ключову інформацію про попередні реєстрації).

ВсеПробіги – календар бігових подій з можливістю реєстрації. Є фільтр за містами, видами дистанцій (https://vseprobegi.org).

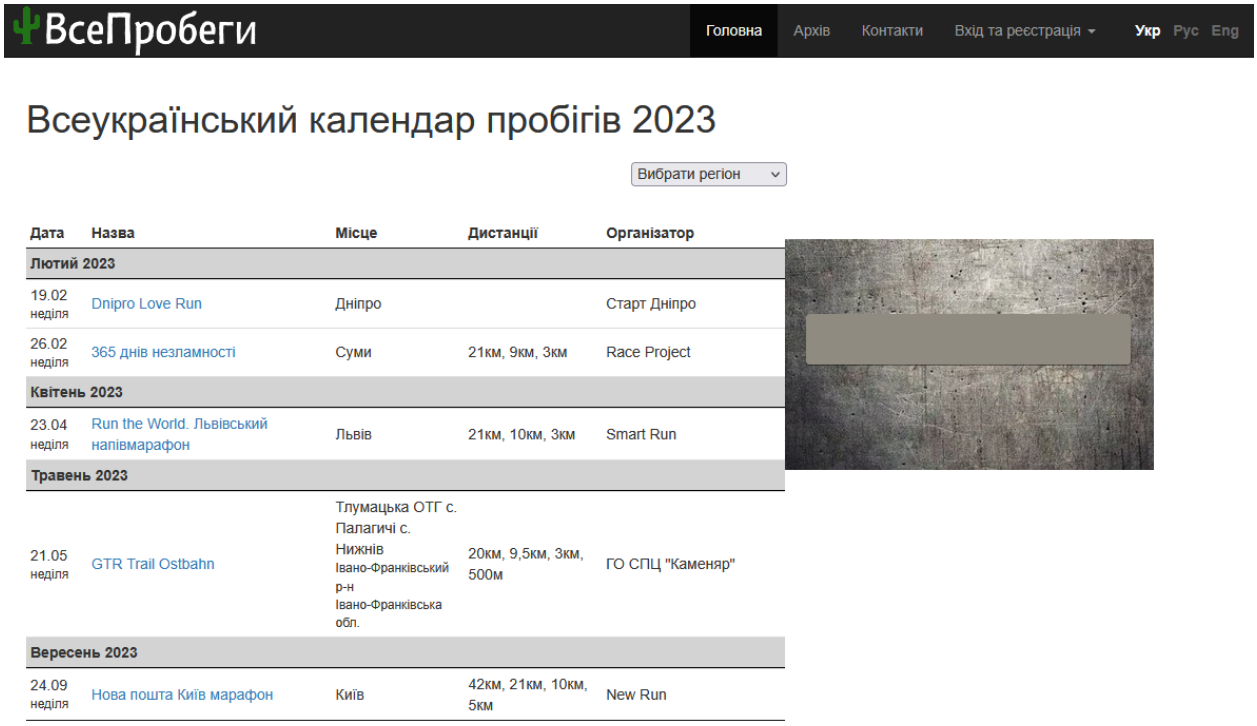

Рисунок 1.2 – Головна сторінка сайту «ВсеПробіги»

Недоліки: вебсайт, скоріш за все, не підтримується, орієнтований лише на Дніпропетровську, Івано-Франківську, Київську, Львівську та Сумську області, відсутня можливість реєстрації учасника на вебсайті, лише перегляд інформації про час, місце і деталі змагань.

1.1.2 Платформи для організації змагань з плавання

Федерація плавання України - призначений для публікації новин про українське плавання (змагання, результати, оголошення), інформації про внески, документи, нормативи (usf.org.ua).

Недоліки: містить лише новини та повідомлення (без функціоналу для керування змаганнями), відсутність профілю, немає пошуку.

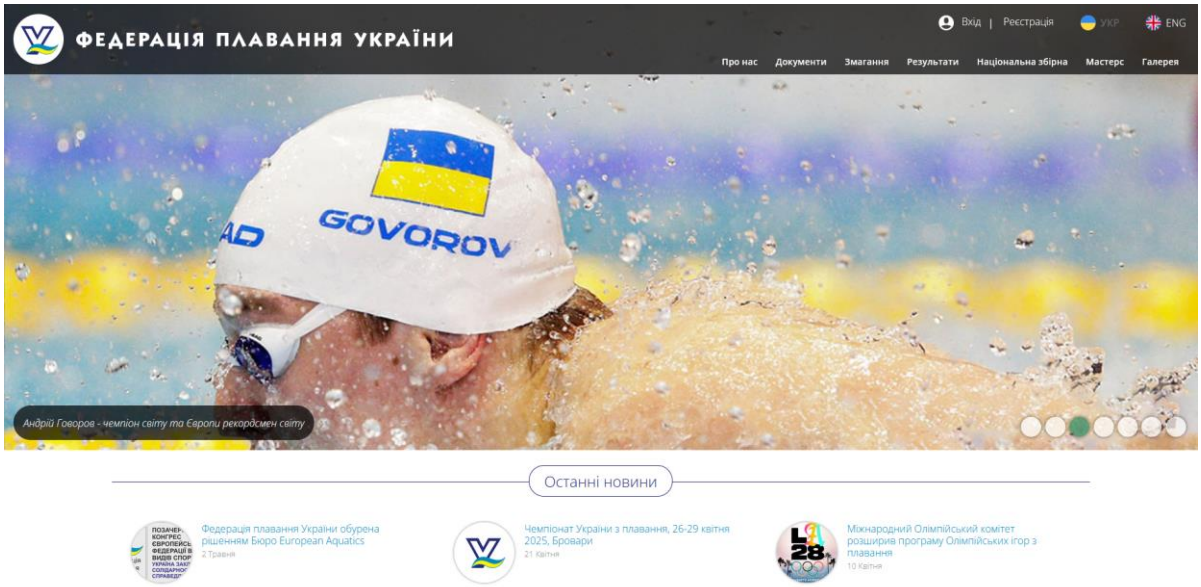

Рисунок 1.3 – Головна сторінка сайту «Федерація плавання України»

SwimMeet Software - система для управління змаганнями з плавання, що дозволяє вести хронометраж і облік результатів (https://swim-meet.com).

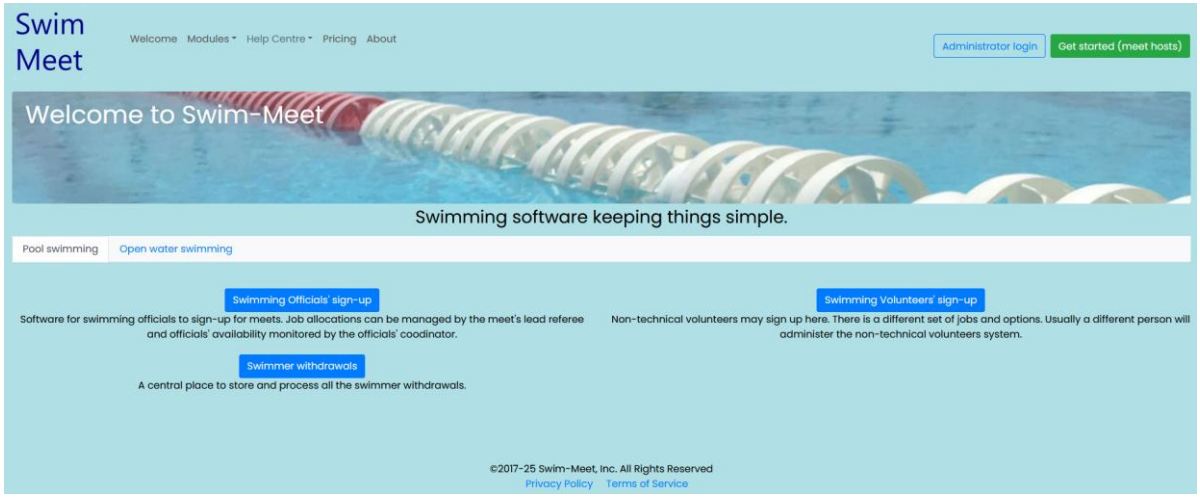

Рисунок 1.4 – Головна сторінка сайту «SwimMeet Software»

Недоліки: має не дуже зручний та зрозумілий інтерфейс, немає української локалізації, немає реєстрації учасника, не орієнтований на широку аудиторію, взагалі немає функцій для зручного пошуку (фільтрів), має платну модель для організаторів, що ускладнює доступ для некомерційних організацій.

1.1.3 Платформи для організації змагань по веслуванню

World Rowing – офіційний сайт міжнародної федерації академічного веслування (https://worldrowing.com).

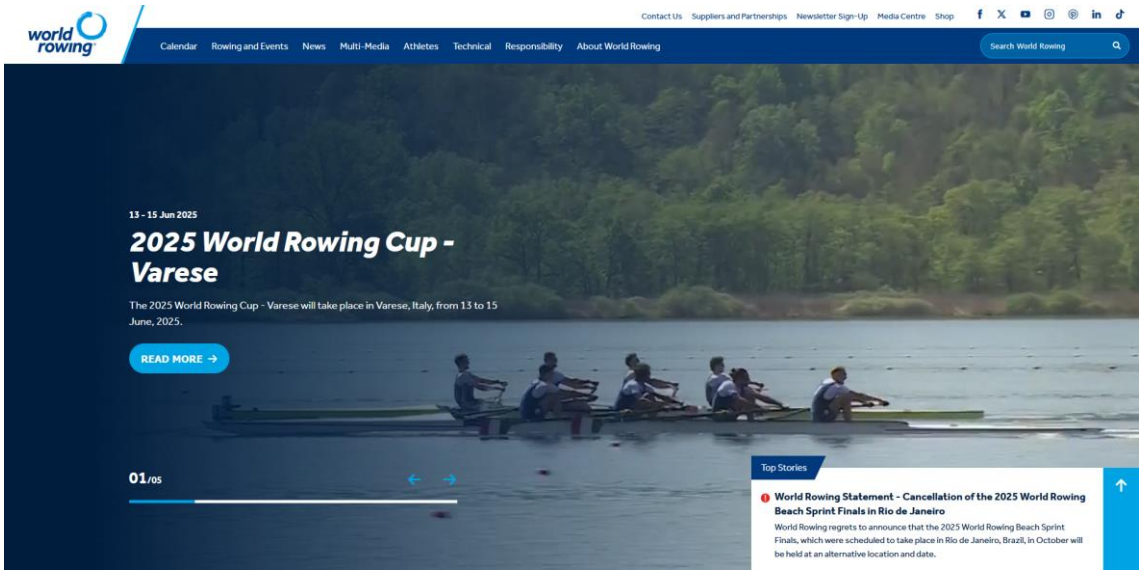

Рисунок 1.5 – Головна сторінка сайту «World Rowing»

Недоліки: немає української локалізації, незручна навігація (користувачі часто скаржаться, що важко знайти відео або результат, особливо на мобільних пристроях.

1.1.4 Платформи для організації змагань з вітрильного спорту

World Sailing - міжнародний календар змагань, клубні та олімпійські події з вітрильного спорту (https://www.sailing.org/).

Рисунок 1.6 – Головна сторінка сайту «World Sailing»

1.1.5 Платформи для організації змагань з триатлону

Let's Do This - платформа для пошуку та реєстрації на триатлон, також платформа включає окремо біг, марафони та велоспорт. Має мапу, на якій зображено точне місце розташування змагання (https://www.letsdothis.com/us).

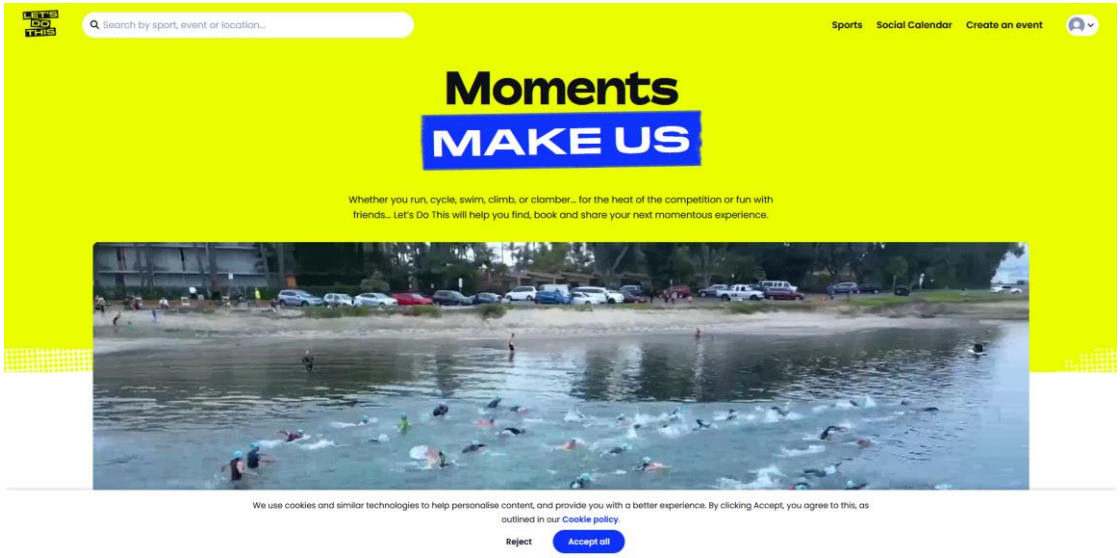

Рисунок 1.7 – Головна сторінка сайту «Let's Do This»

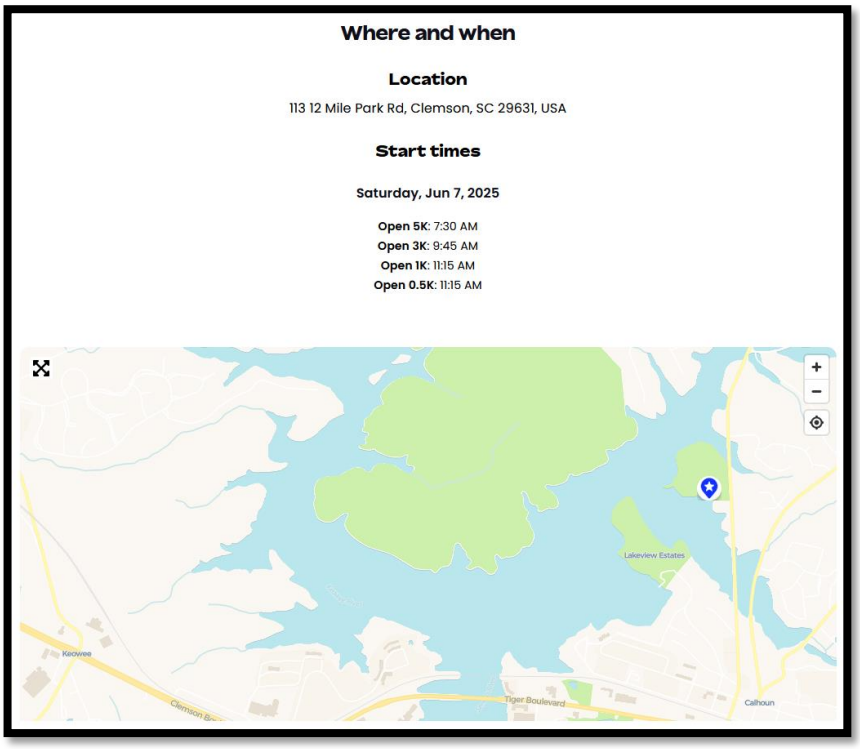

Рисунок 1.8 – Сторінка заходу з плавання

Недоліки: на цій платформі відсутня можливість реєстрації учасника на змагання, користувачі можуть лише переглядати інформацію про час і місце проведення заходів.

1.1.6 Переваги та недоліки всіх платформ

Переваги: автоматизовані основні процеси (реєстрація, формування розкладу, облік результатів), доступність для користувачів завдяки онлайн-платформам, інтеграція з мобільними додатками та соціальними мережами.

Недоліки: деякі вебсайти складні у використанні та потребують навчання, підтримка деяких вебсайтів припинена, відсутність української локалізації та підтримки специфічних вимог для регіональних змагань.

1.2 Визначення вимог вебсайту для управління спортивними змаганнями

Для створення вебсайту, який буде інтуїтивно зрозумілий і дозволить організаторам змагань автоматизувати основні процеси, були визначені функціональні та нефункціональні вимоги. Функціональні вимоги визначатимуть набір можливостей, які вебсайт буде забезпечувати для користувачів, а нефункціональні - стосуються якості роботи вебсайту, його продуктивності та безпеки.

1.2.1 Функціональні вимоги

- Реєстрація та авторизація користувачів: підтримка кількох ролей (організатор, учасник), вхід через email/пароль або соцмережі.
- Управління змаганнями: створення та налаштування змагання (вид спорту, назва, місце проведення і т.д.), додавання команд та учасників.
- Ведення статистики та результатів: внесення результатів змагань, відображення результатів, створення профілів команд та учасників.
- Адміністративна панель: управління користувачами та їхніми правами доступу, аналітика щодо активності користувачів та переглядів сайту.

1.2.2 Нефункціональні вимоги

- Зручність та адаптивність: інтуїтивно зрозумілий інтерфейс, швидкий доступ до ключової інформації.
- Безпека: захист персональних даних користувачів, автоматичне резервне копіювання бази даних.

РОЗДІЛ 2.

ПРОЄКТУВАННЯ ТА РОЗРОБКА ВЕБСАЙТУ

2.1. Розгляд архітектури вебдодатку

Для реалізації вебсайту було обрано стек технологій React.js, Node.js (Express) та MongoDB. Такий вибір обумовлено рядом причин:

- React.js – популярна бібліотека для створення інтерфейсів, яка дозволяє будувати швидкі та динамічні вебсторінки. Вона має велику спільноту, добре документована та активно підтримується. Завдяки React стало можливим реалізувати компонентну структуру інтерфейсу, що спрощує розробку, тестування та масштабування проєкту.
- Node.js та Express.js – сучасний інструмент для створення серверної частини на мові JavaScript. Використання Node.js дозволяє розробляти як клієнтську, так і серверну частину однією мовою програмування, що спрощує зв’язок між частинами системи. Express.js як фреймворк забезпечує зручне створення REST API, необхідного для взаємодії з клієнтською частиною.
- MongoDB – документоорієнтована база даних, яка ідеально підходить для зберігання структурованих об'єктів у форматі JSON. Вона дозволяє легко працювати з даними без необхідності суворого визначення схеми. Це дозволяє швидко розгортати проєкт і гнучко змінювати структуру даних у процесі розробки.

2.1.1 Взаємодія компонентів (реєстрація користувача на вебсайт)

1. Користувач відкриває вебсайт і проходить реєстрацію.
2. Вводить особисті дані та натискає кнопку «зареєструватись», тим самим відправляє дані на сервер.
3. Сервер перевіряє коректність даних та створює нового користувача в базі даних.

4. Повертає відповідь про успішну реєстрацію.

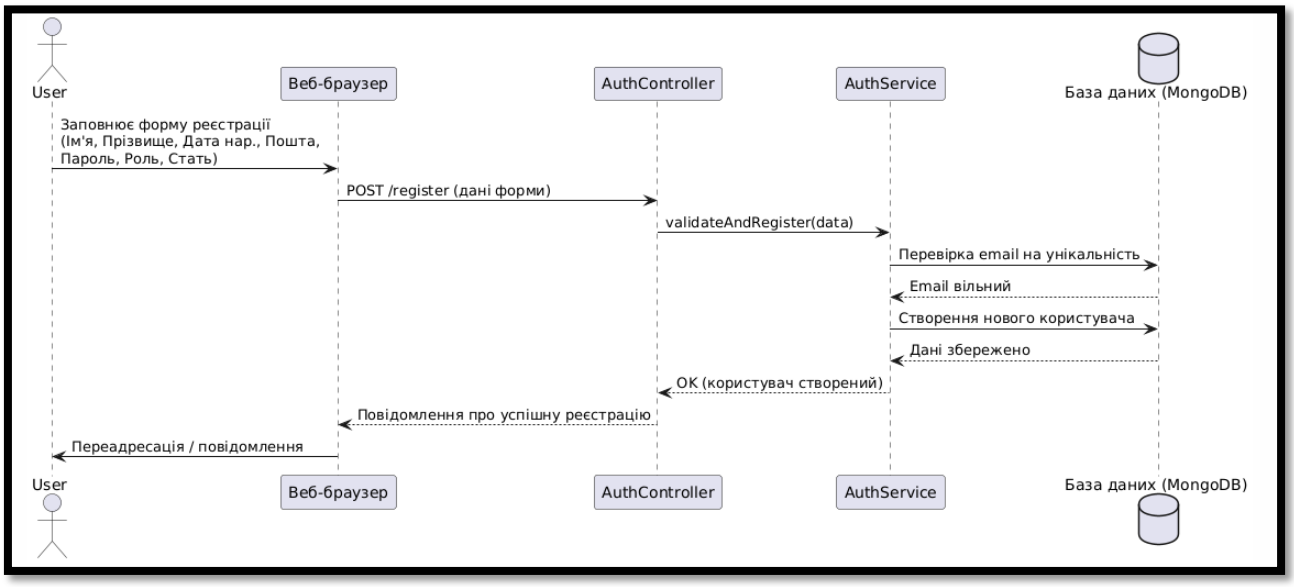

Рисунок 2.1 - Діаграма взаємодії при реєстрації користувача на вебсайт

2.1.2 Взаємодія компонентів (кнопка створення змагання)

Після реєстрації чи входу, користувач отримує доступ до створення змагання.

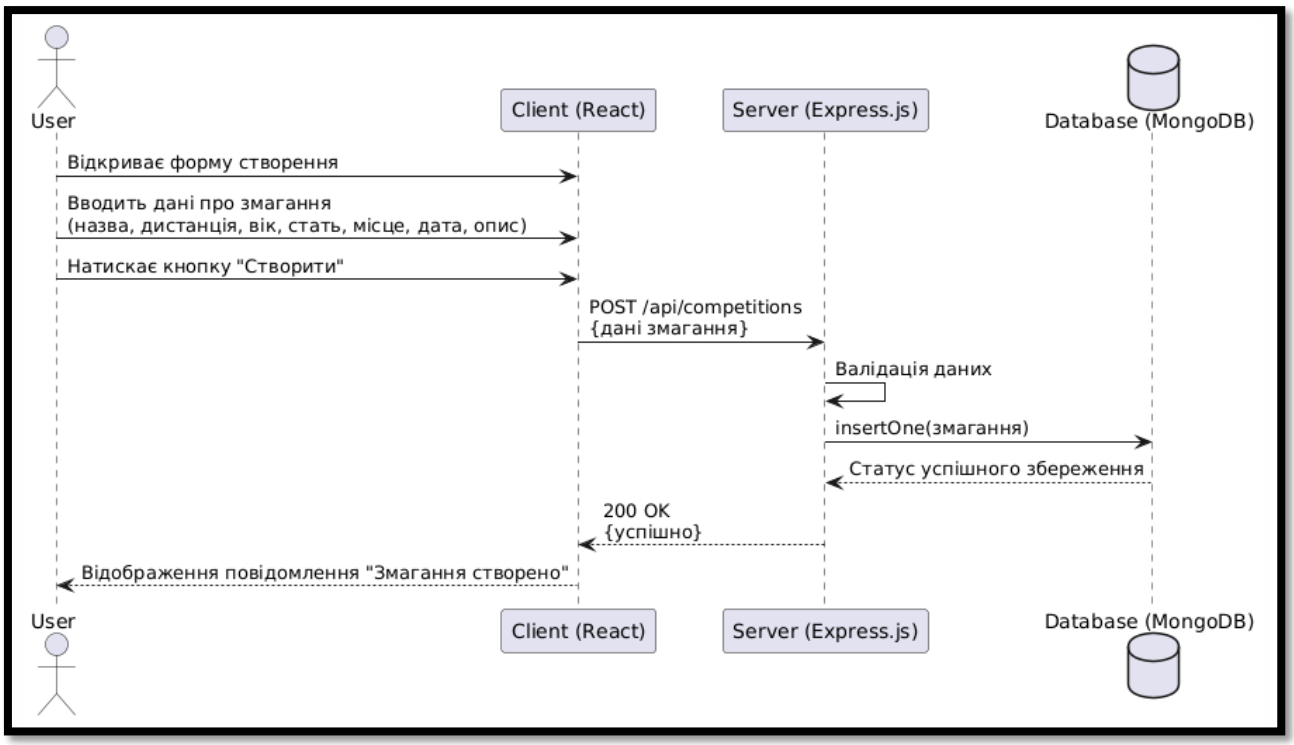

Рисунок 2.2 – Діаграма взаємодії користувача з кнопкою створення змагання

2.1.3 Взаємодія компонентів (реєстрація користувача на змагання)

1. Користувач відкриває сторінку змагання.
2. Сервер віддає дані про змагання з БД.
3. Користувач натискає кнопку реєстрації.
4. Дані про участь записуються в таблицю реєстрацій.
5. Повертається підтвердження реєстрації.

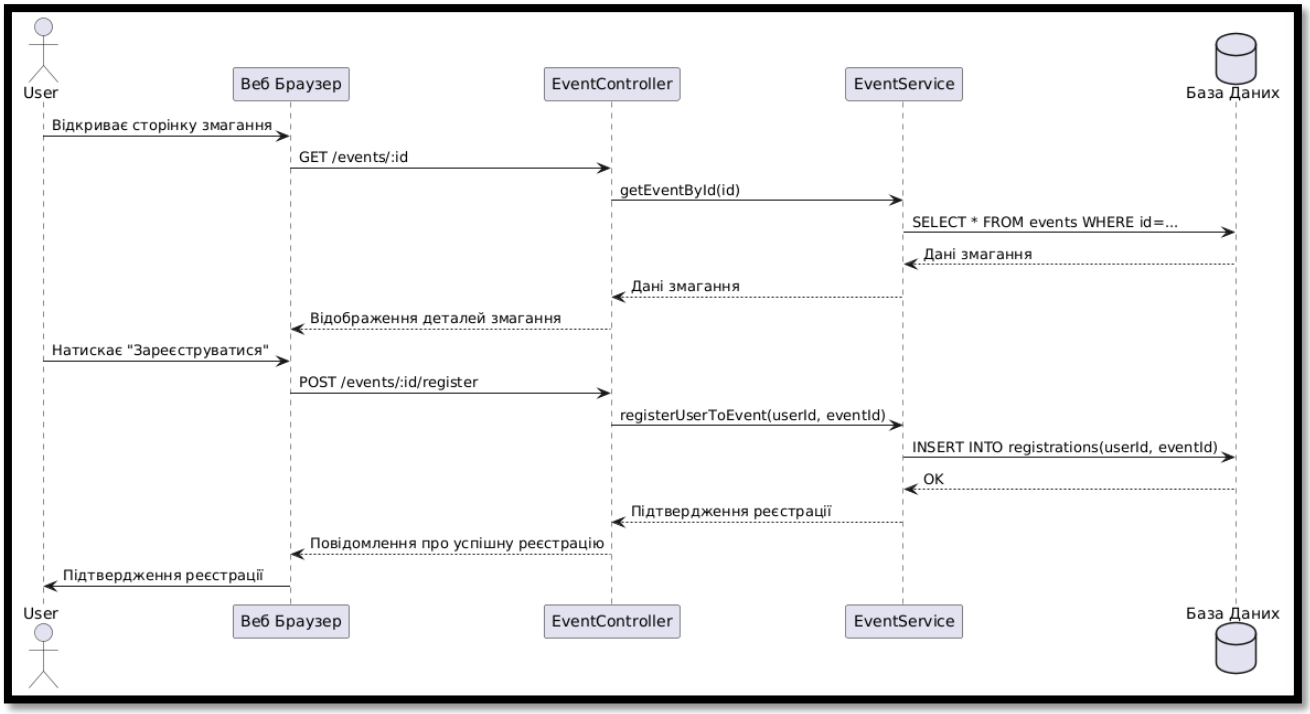

Рисунок 2.3 – Діаграма реєстрації користувача на змагання

2.2. Дизайн інтерфейсу

Інтерфейс орієнтований на роботу з персонального комп’ютера, що дозволяє максимально ефективно використовувати екранний простір. Увага під час розробки приділялася логічній побудові навігації, простоті доступу до ключових функцій та зрозумілості для кінцевого користувача. Проєктування інтерфейсу здійснювалося за допомогою онлайн-сервісу Figma, що дозволило створити інтерактивні макети та швидко вносити правки. Було розроблено такі сторінки: сторінка реєстрації та входу користувача, головна сторінка, сторінка

змагань та їхнього створення, профіль користувача, сторінка результатів та сторінка введення результатів учасників.

2.2.1 Сторінки реєстрації та входу

Перед тим як перейти на головну сторінку вебсайту, необхідно зареєструватись (рис. 2.4) або увійти (рис. 2.5).

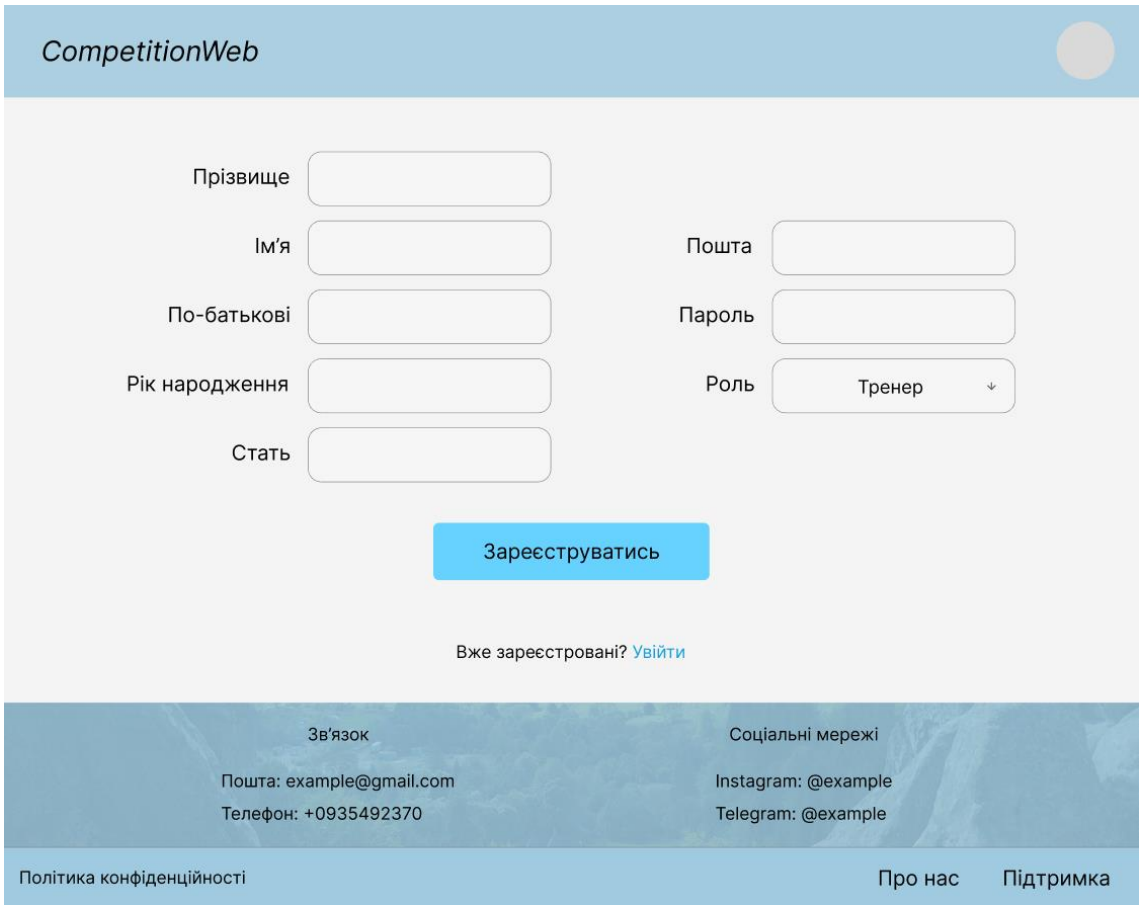

Рисунок 2.4 Сторінка реєстрації

На сторінці реєстрації треба обов'язково заповнити всі поля. Також на сторінці реєстрації є опція «Роль», в якій потрібно вибрати ким буде являтись користувач (учасник або тренер). Після реєстрації на вебсайті, змінити інформацію про себе не можна (лише фото профілю), тому дані треба вводити одразу правильні.

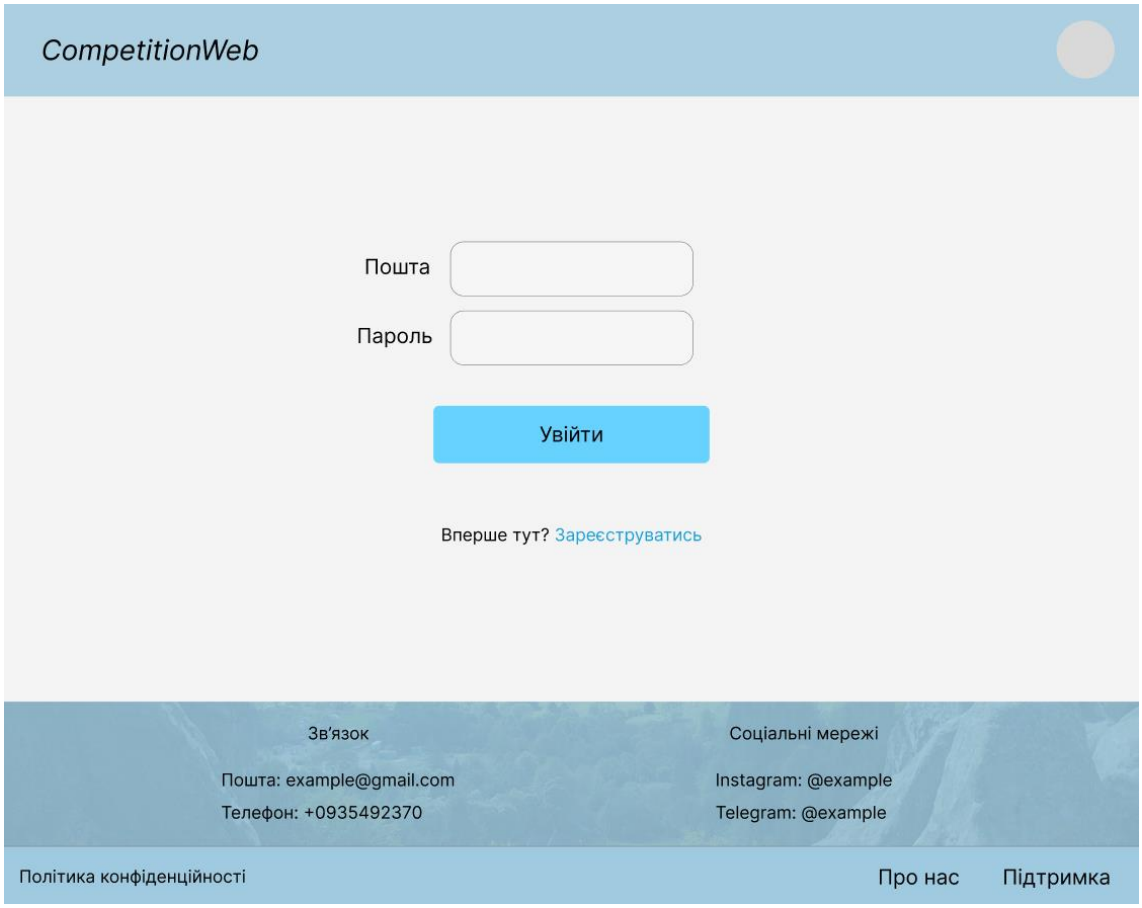

Рисунок 2.5 – Сторінка входу

2.2.2 Головна сторінка та footer

На головній сторінці можна: створити своє змагання, перейти у свій профіль, обрати вид спорту для перегляду змагань та зареєструватись. Створені змагання доступні усім користувачам вебсайту. Також є фільтр, завдяки якому можна легше знайти потрібне змагання, наприклад змагання з потрібною дистанцією або віком.

Рисунок 2.6 – Footer для всіх сторінок вебсайту

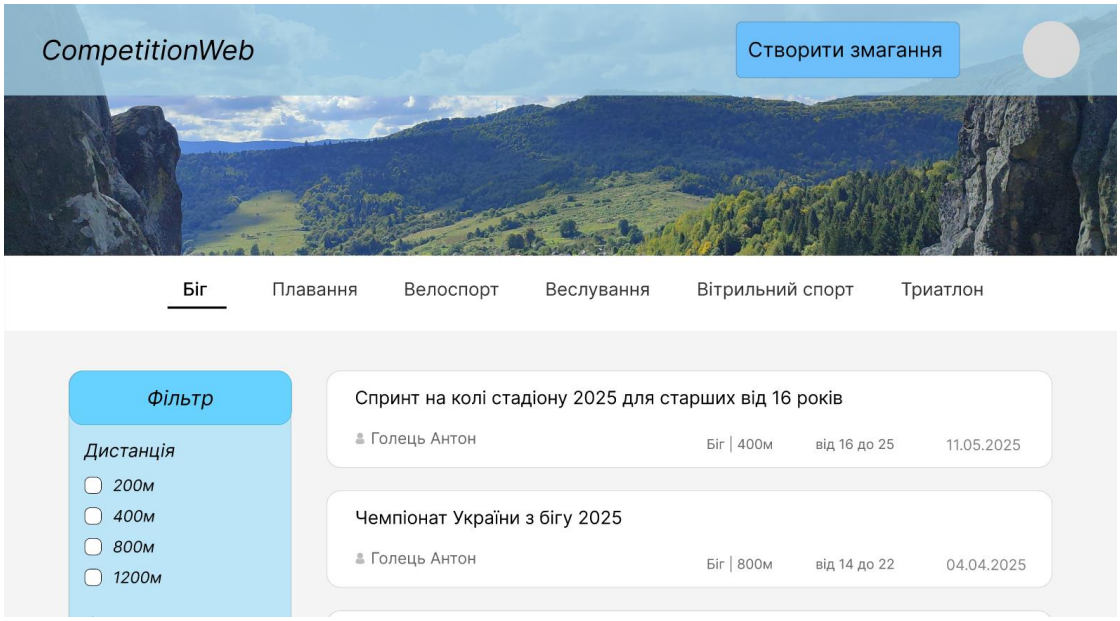

Рисунок 2.7 – Головна сторінка

2.2.3 Сторінки створення змагань

При натисненні кнопки «Створити змагання» на головній сторінці, відкривається сторінка вибору змагання (рис. 2.8). На цій сторінці доступні кнопки з різними видами спорту, при натисненні на які відкриваються для кожного виду спорту індивідуальні опції для створення змагання.

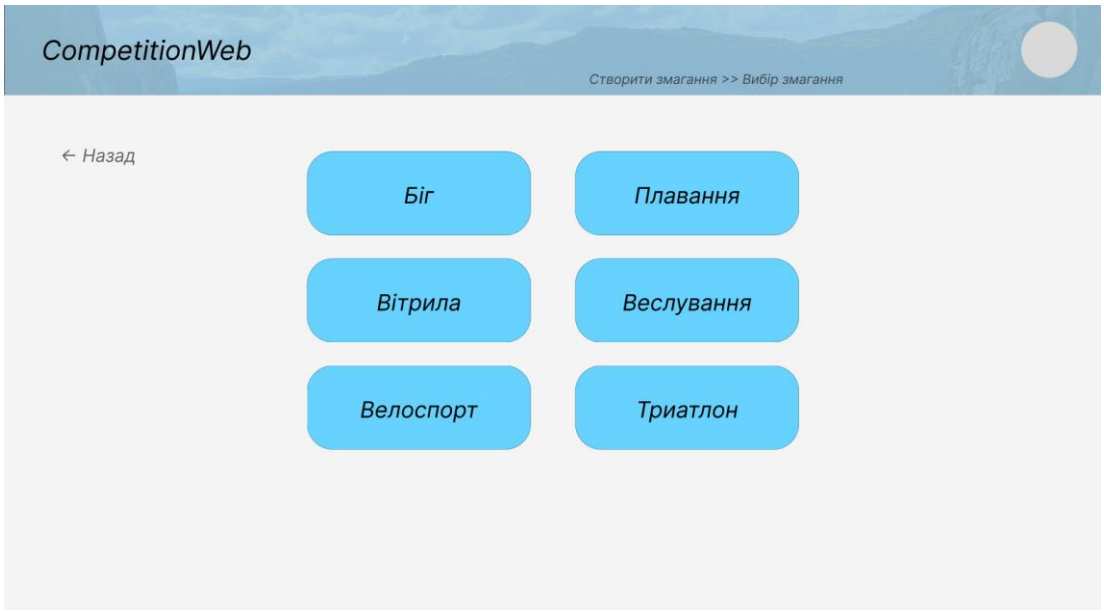

Рисунок 2.8 – Сторінка вибору змагання

На сторінці вибору змагання при натисненні на кнопку «Біг», відкриваються опції для виду спорту з бігу (рис. 2.9).

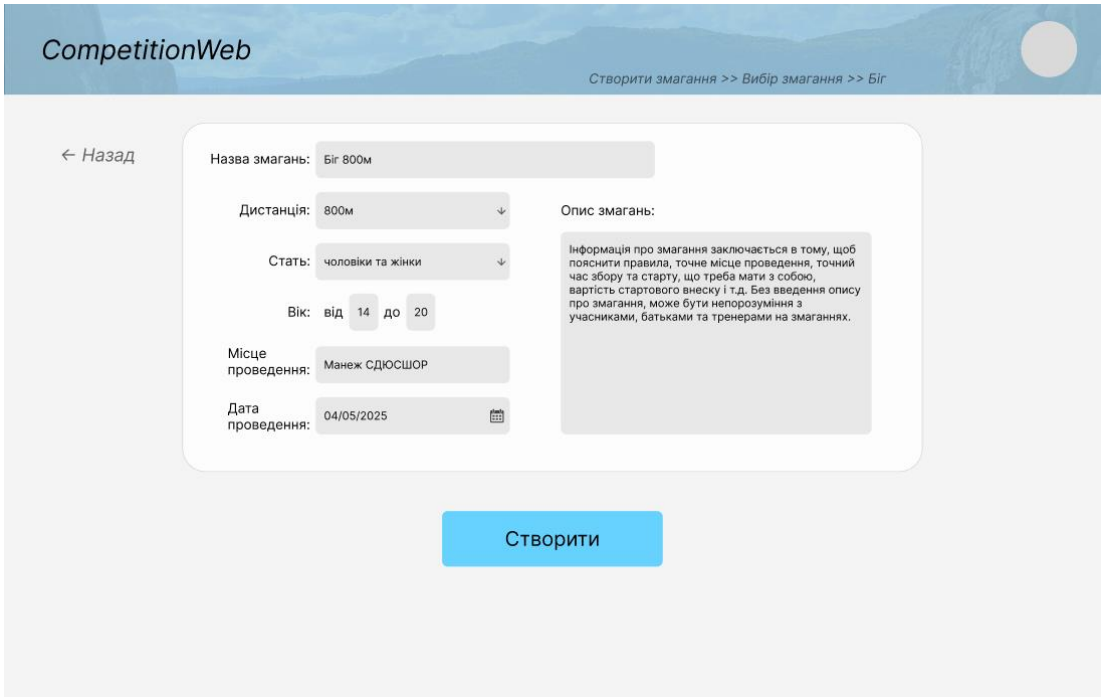

Рисунок 2.9 – Опції для створення змагання для бігу

Для створення змагання, необхідно: вказати назву змагань, вибрати дистанцію, стать учасників, вік, місце проведення та опис змагань. Після натиснення кнопки «Створити», користувача автоматично перенаправляє на сторінку профілю з щойно створеним ним змаганням (рис. 2.10).

2.2.4 Сторінка профілю

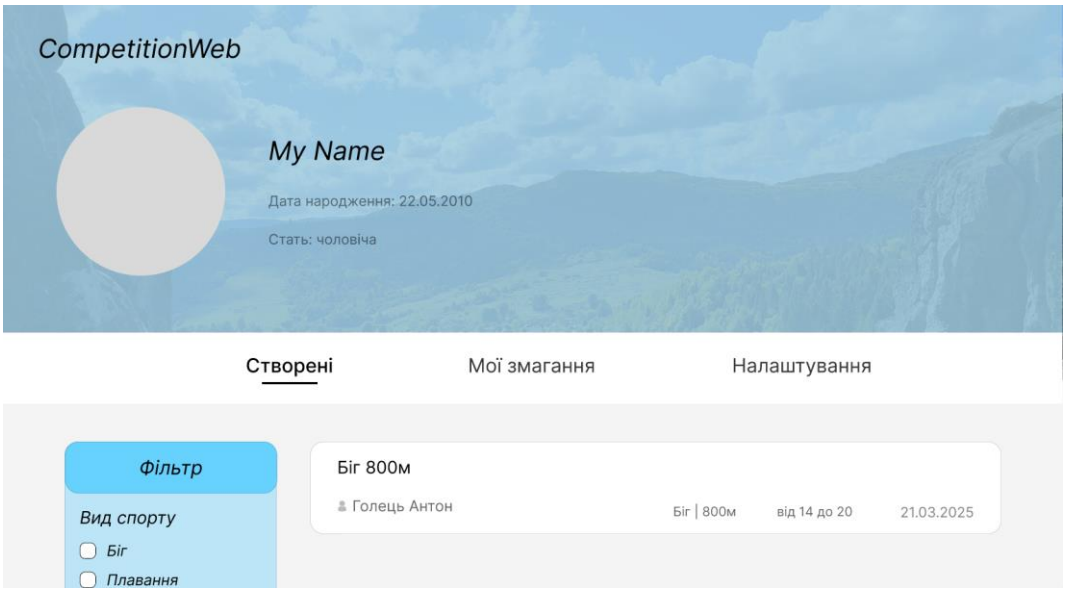

Рисунок 2.10 – Сторінка профілю. Створені користувачем змагання

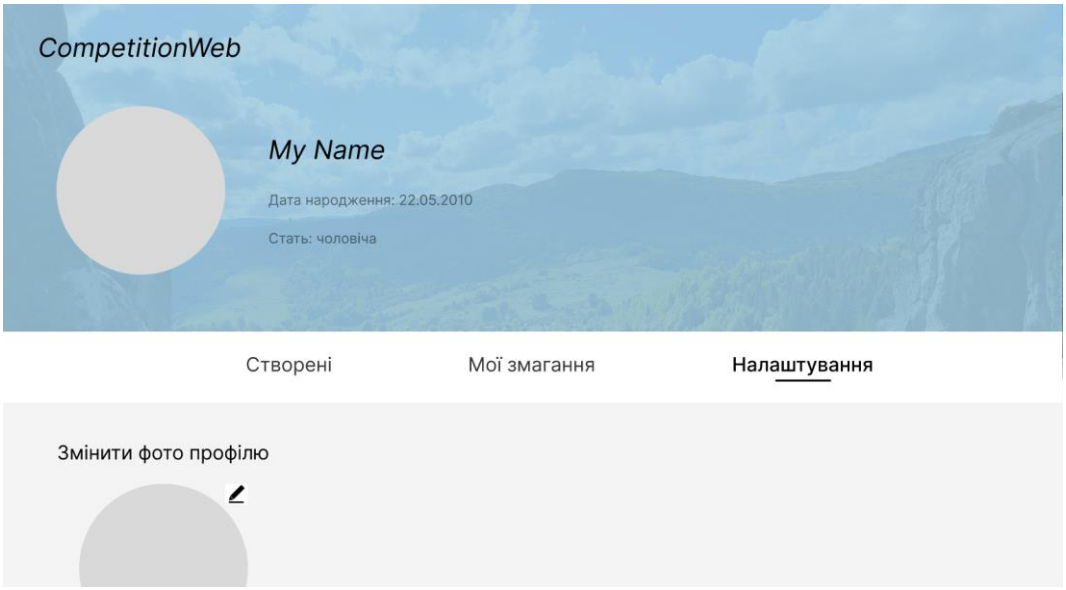

Рисунок 2.11 – Сторінка профілю. Налаштування профілю

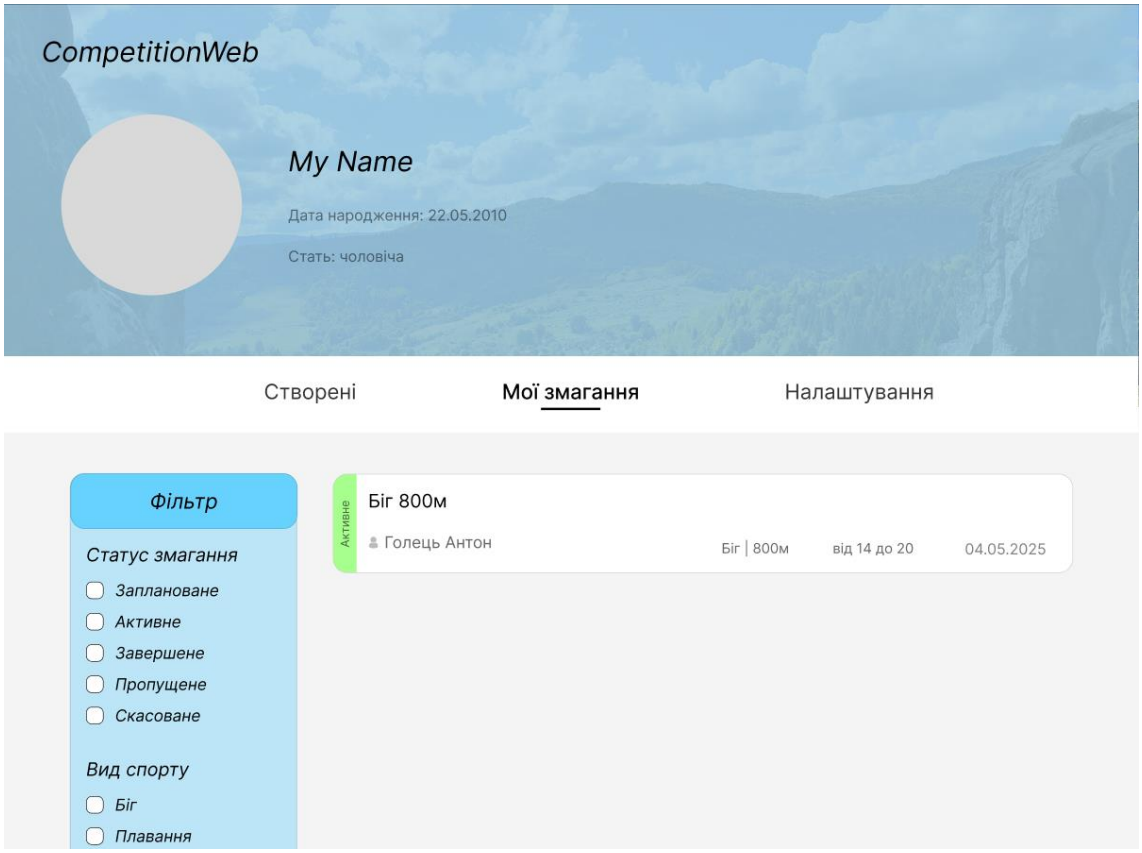

Рисунок 2.12 – Сторінка профілю. Список змагань, у яких користувач бере або буде брати участь

У фільтрі «Статус змагання» означає:

- Заплановане: змагання, яке ще не почалося
- Активне: змагання, яке проводиться зараз
- Завершене: змагання, яке закінчилось
- Пропущене: змагання, яке закінчилось і в ньому не брали участь
- Скасоване: змагання, яке відмінилось з якихось причин

Статус змагання відображається лише тоді, коли користувач зареєструвався на це змагання.

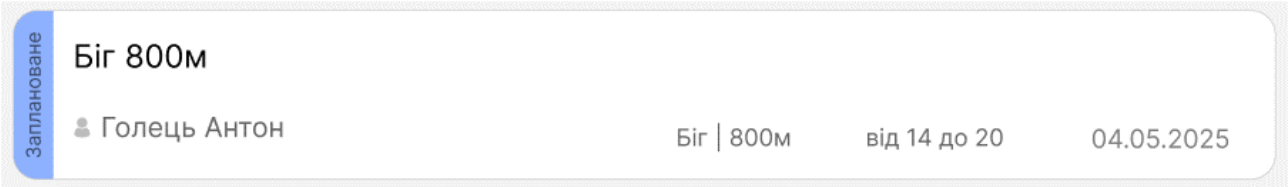

Рисунок 2.13 – Загальний вигляд запланованого змагання

Рисунок 2.14 – Загальний вигляд активного змагання

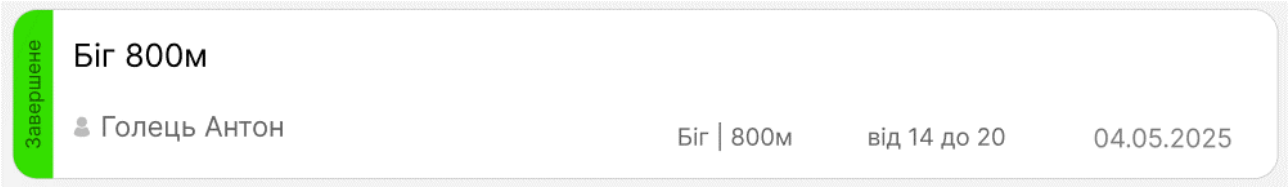

Рисунок 2.15 – Загальний вигляд завершеного змагання

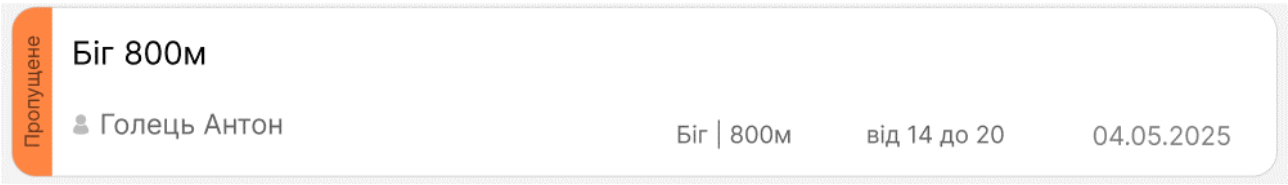

Рисунок 2.16 – Загальний вигляд пропущеного змагання

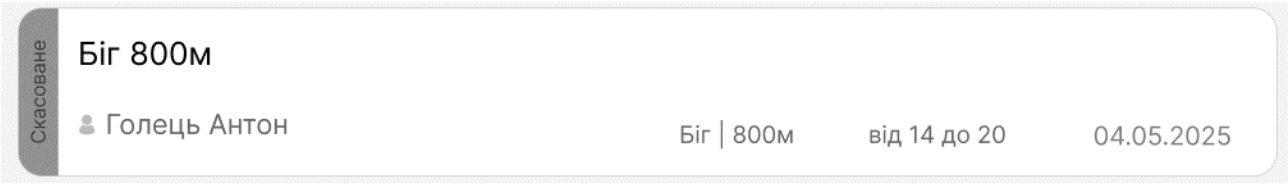

Рисунок 2.17 – Загальний вигляд скасованого змагання

2.2.5 Сторінка створеного змагання

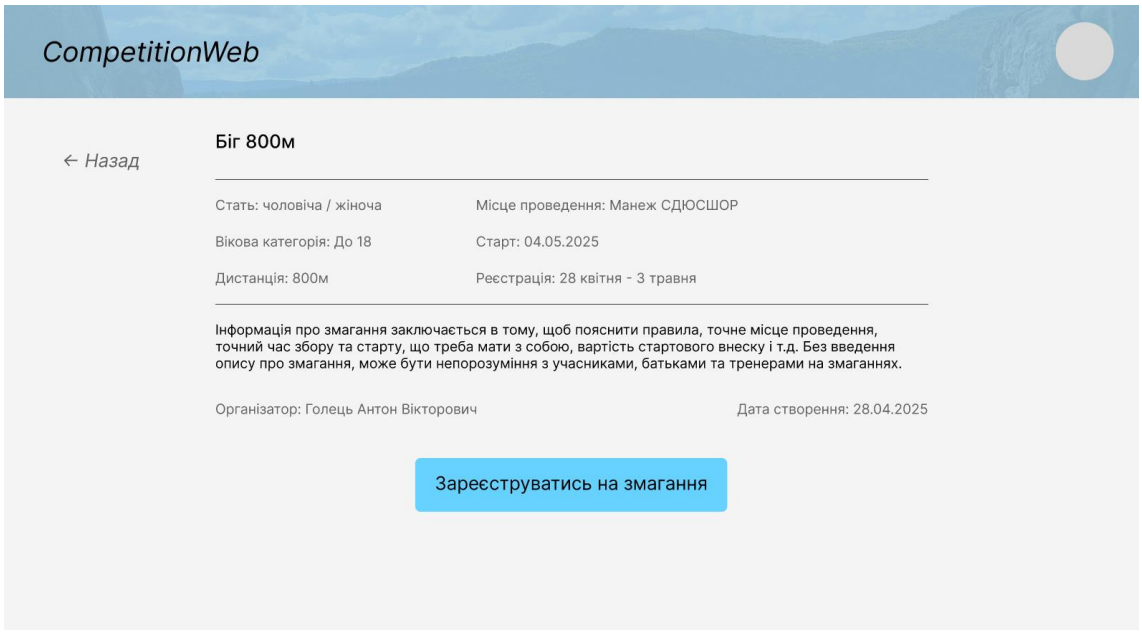

Рисунок 2.18 – Загальний вигляд в не зареєстрованому змаганні

При натисненні кнопки «Зареєструватись на змагання», перекидає у профіль з зареєстрованим змаганням.

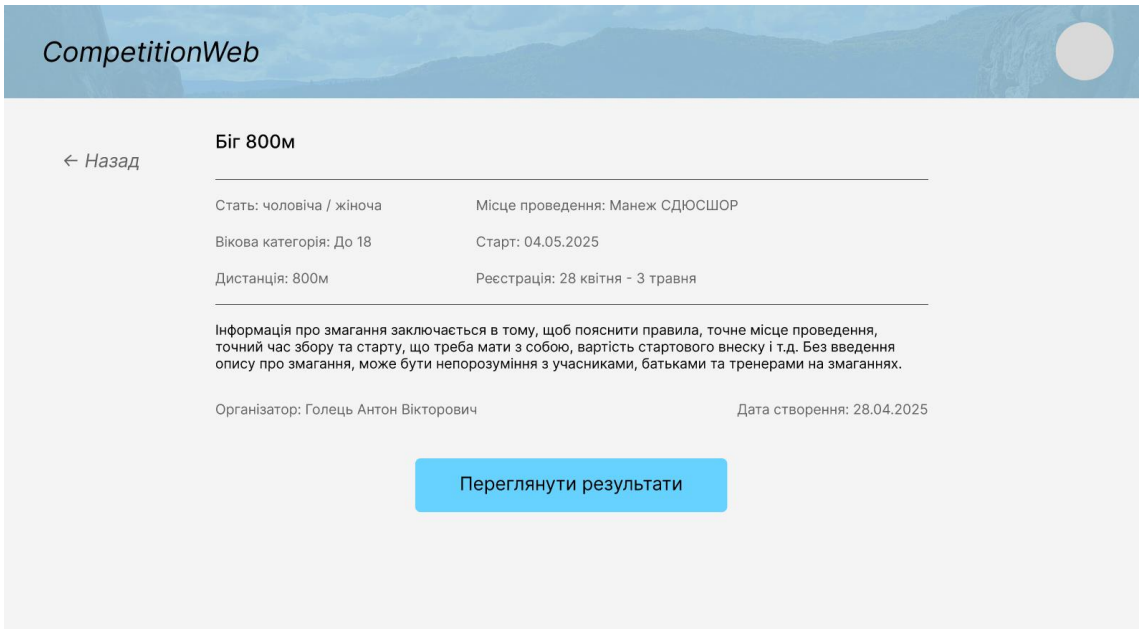

Рисунок 2.19 – Загальний вигляд в зареєстрованому змаганні іншого організатора

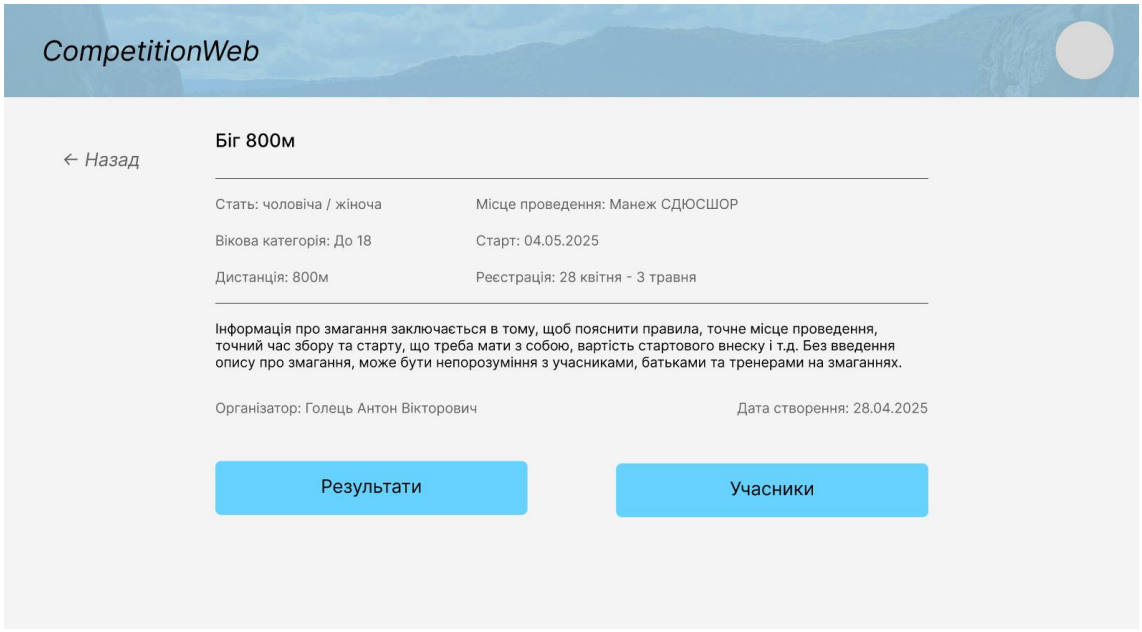

Рисунок 2.20 – Загальний вигляд в своєму створеному та зареєстрованому змаганні

При натисненні кнопки «Результати», відкриваються результати змагань (рис. 2.21). Ця кнопка доступна всім але натискається тільки тоді, коли організатор опублікував результат хоча б одного учасника.

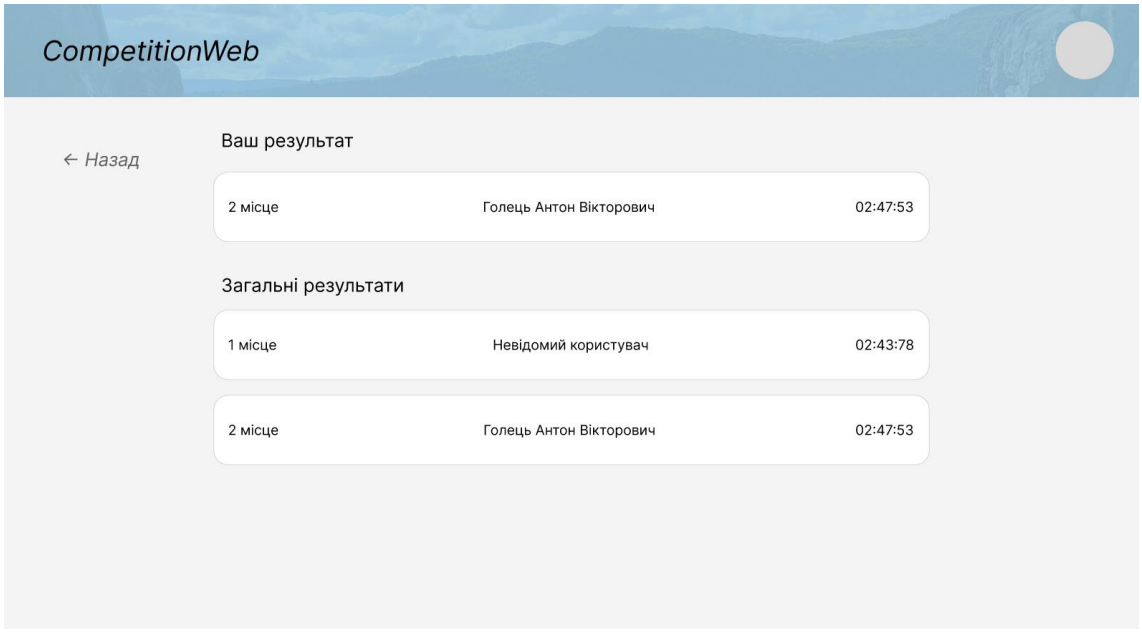

Рисунок 2.21 – Загальний вигляд результатів змагань

При натисненні кнопки «Учасники», відкривається сторінка з зареєстрованими на змагання учасниками (рис. 2.22), кнопка доступна тільки організатору. На цій сторінці можна переглянути усіх учасників змагання, додавати результати змагання до кожного учасника під час змагань або після них.

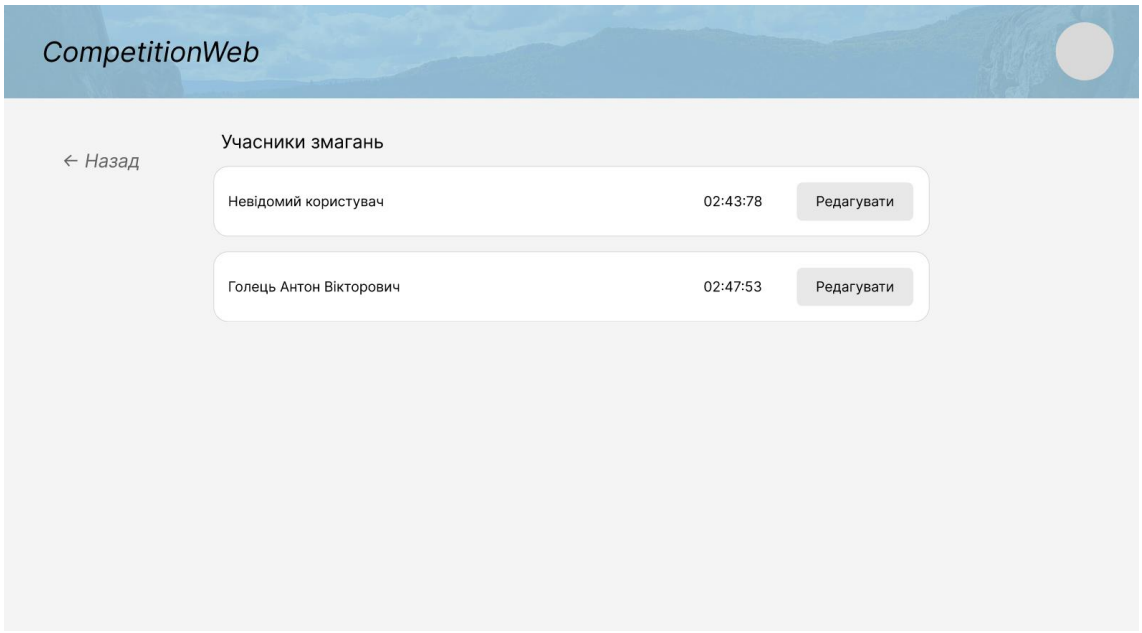

Рисунок 2.22 – Загальний вигляд учасників змагань

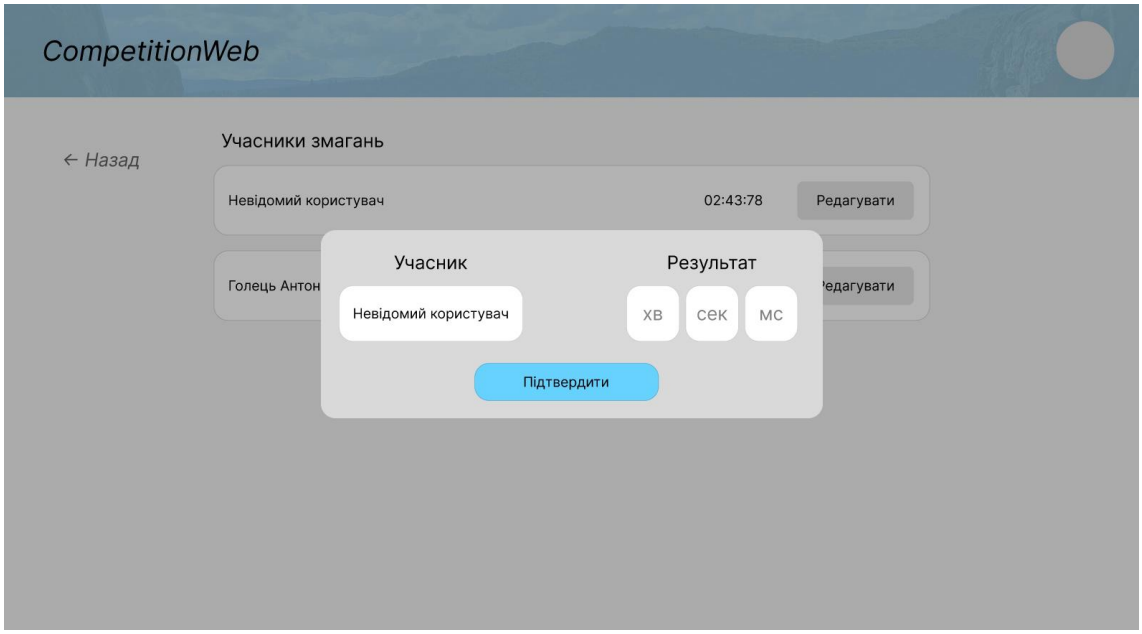

Рисунок 2.23 – Введення результату учасника

2.3. Налаштування середовища розробки

Для реалізації функціональності вебсайту було налаштовано серверну частину (back-end) на основі Node.js з Express, клієнтську частину (front-end) з використанням React та базу даних MongoDB.

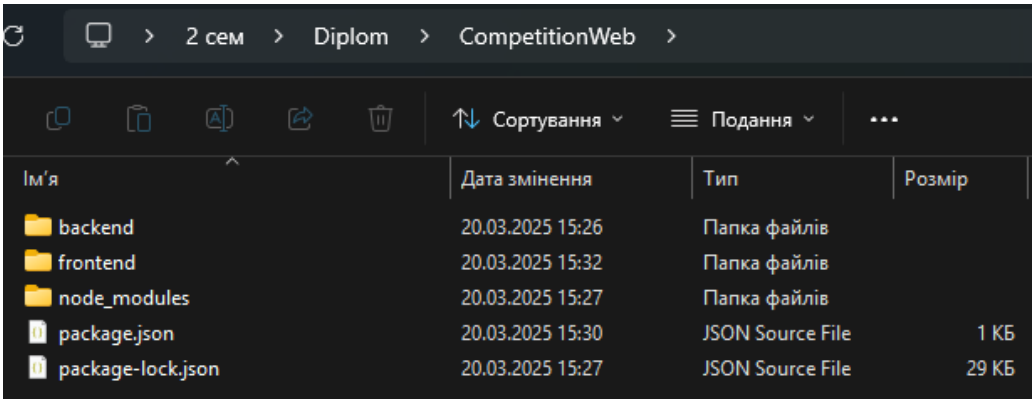

Рисунок 2.25 – Створення окремої папки для проєкту

2.3.1 Налаштування серверної частини (back-end)

```
C:\Users\Anton\Desktop\sports-management-app\back-end>npm init -y
Wrote to C:\Users\Anton\Desktop\sports-management-app\back-end\package.json:

{
  "name": "back-end",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}



C:\Users\Anton\Desktop\sports-management-app\back-end>npm install express mongoose cors dotenv bcryptjs jsonwebtoken

added 107 packages, and audited 108 packages in 5s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Users\Anton\Desktop\sports-management-app\back-end>npm install nodemon --save-dev

added 28 packages, and audited 136 packages in 2s

20 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Рисунок 2.26 – Ініціалізація back-end, встановлення необхідних бібліотек, встановлення nodemon

Були встановлені наступні бібліотеки:

- express – вебфреймворк для Node.js;
- mongoose – обробка запитів до MongoDB;
- cors – дозвіл запитів з фронтенду;
- dotenv – змінні середовища;
- bcryptjs – хешування паролів;
- jsonwebtoken – створення та перевірка токенів авторизації;
- nodemon – для автоматичного перезапуску сервера при зміні файлів.

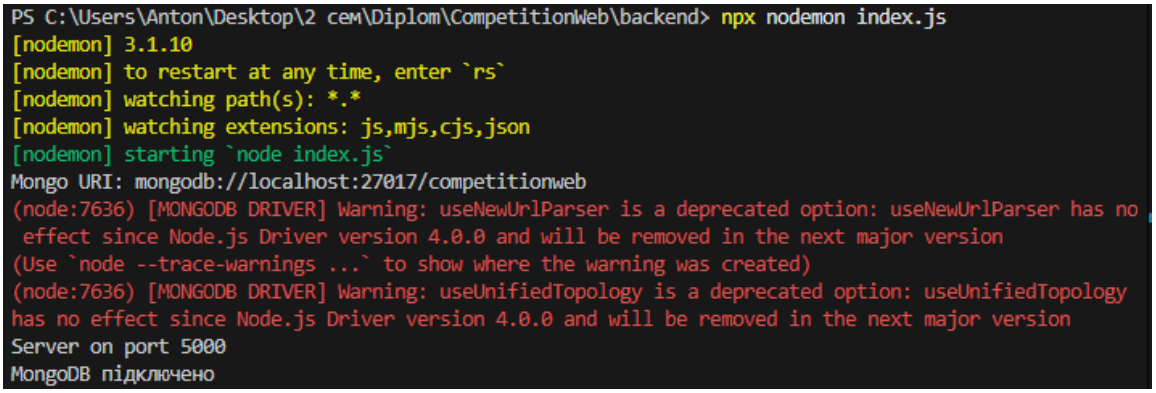

Рисунок 2.27 – Запуск сервера

2.3.2 Налаштування клієнтської частини (front-end)

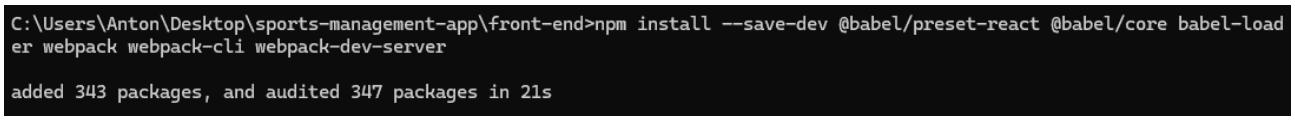

Рисунок 2.28 – Встановлення бібліотек для компіляції JSX та сучасного JavaScript

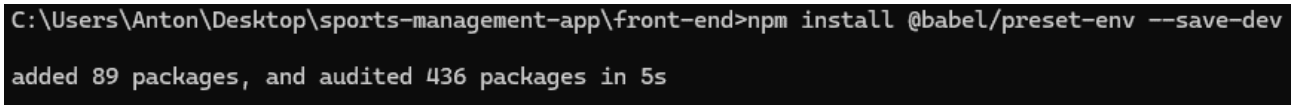

Рисунок 2.29 – Встановлення бібліотеки для навігації по сторінках без перезавантаження сайту та підтримки нових стандартів JS

```
C:\Users\Anton\Desktop\sports-management-app\front-end>npm install react-router-dom axios

added 14 packages, and audited 450 packages in 4s

61 packages are looking for funding
  run `npm fund` for details
```

Рисунок 2.30 – Встановлення бібліотеки для виконання HTTP-запитів до back-end API

```
C:\Users\Anton\Desktop\sports-management-app\front-end>npm install --save-dev style-loader css-loader

added 14 packages, and audited 494 packages in 7s

75 packages are looking for funding
  run `npm fund` for details
```

Рисунок 2.31 – Встановлення бібліотеки для підтримки імпорту CSS-файлів у JavaScript-коді

```
PS C:\Users\Anton\Desktop\2 сем\Diplom\CompetitionWeb\frontend> npm start

> frontend@0.1.0 start
> react-scripts start

(node:5408) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddl
eware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use `node --trace-deprecation ...` to show where the warning was created)
(node:5408) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMid
dleware' option is deprecated. Please use the 'setupMiddlewares' option.
Starting the development server...
Compiled successfully!

You can now view frontend in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.56.1:3000
```

Рисунок 2.32 – Запуск клієнта

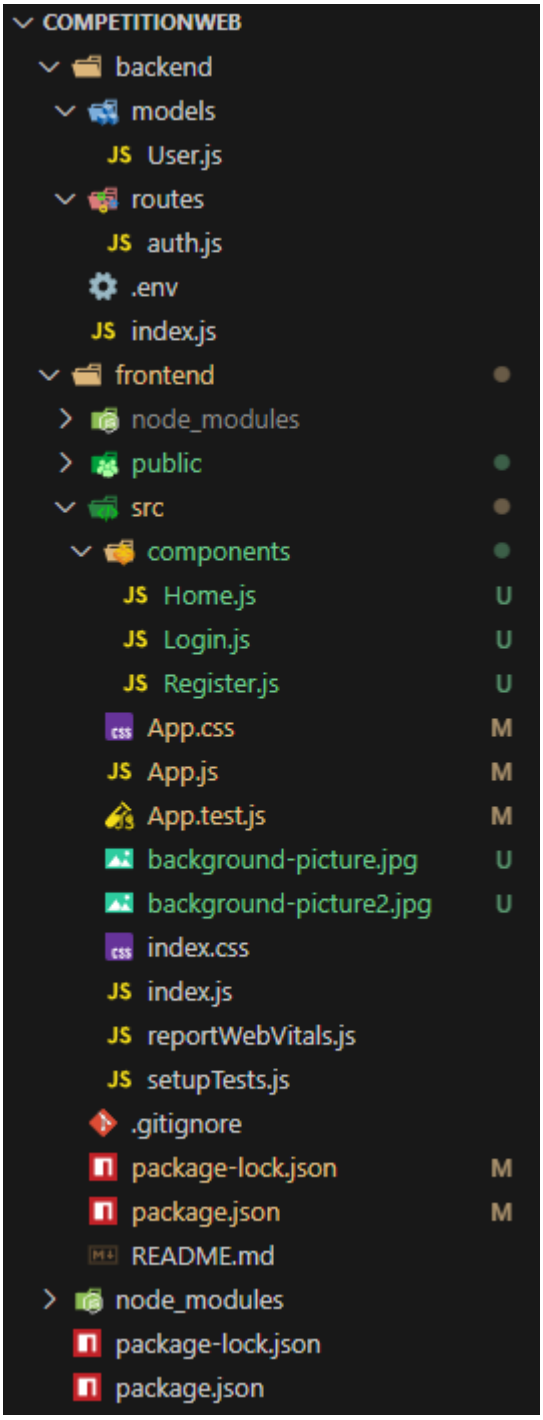

Рисунок 2.33 – Структура проєкту у середовищі розробки VS Code

Весь лістинг вихідного коду, що реалізує функціональність вебсайту, наведено у додатках (див. Додатки А–В).

РОЗДІЛ 3.

ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ

3.1. Тестування функціональності вебсайту

На даному етапі розробки вебсайту було виконано тестування функціональності на основі прототипів і макетів у середовищі Figma. Проєкт ще не має повністю реалізованої функціональної частини, однак структура, логіка переходів між сторінками та інтерфейс уже сформовані.

Мета тестування – перевірити відповідність інтерфейсу заявленому функціоналу, а також описати очікувану поведінку компонентів при їх реалізації. Основним методом було імітаційне функціональне тестування за допомогою тест-кейсів, які відображають реальні сценарії взаємодії користувача з вебзастосунком.

Таблиця 3.1 – Тест-кейси імітаційного тестування

№	Назва тесту	Вхідні дані	Очікуваний результат	Фактичний результат
1	Реєстрація нового користувача	Ім’я, прізвище, дата нар., email, пароль, стать	Користувач створений, повідомлення успіху	Успішно
2	Реєстрація з вже існуючою поштою	Email, який вже існує	Повідомлення про помилку	Успішно
3	Вхід з неправильним паролем	Вірна пошта + неправильний пароль	Повідомлення “Невірний пароль”	Успішно
4	Створення змагання	Назва, дата, місце проведення, тип	Змагання збережено	Успішно
5	Перегляд результатів змагання	Змагання з доданими учасниками	Виводиться таблиця результатів	Успішно
6	Спроба створити змагання без назви	Пусте поле "назва"	Виводиться повідомлення про обов’язкове поле	Успішно

Приклад реєстрації користувача на вебсайт (рис. 3.1) та автоматичне внесення його даних в mongoDB (рис. 3.3)

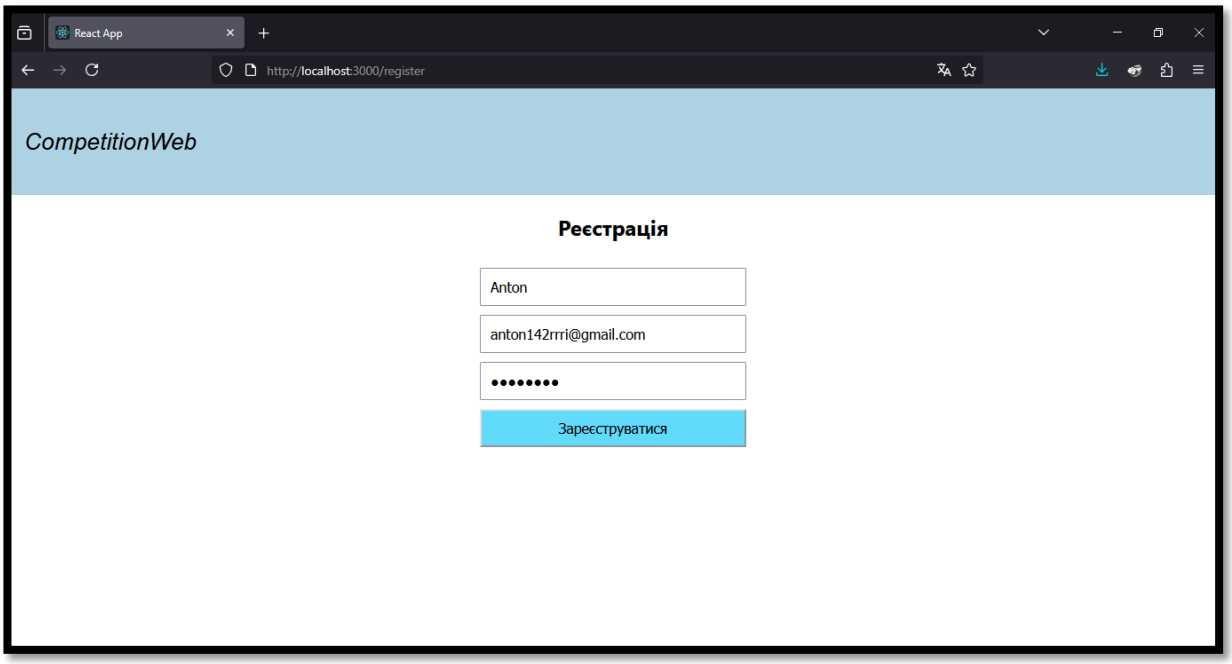

Рисунок 3.1 – Реєстрація на вебсайт

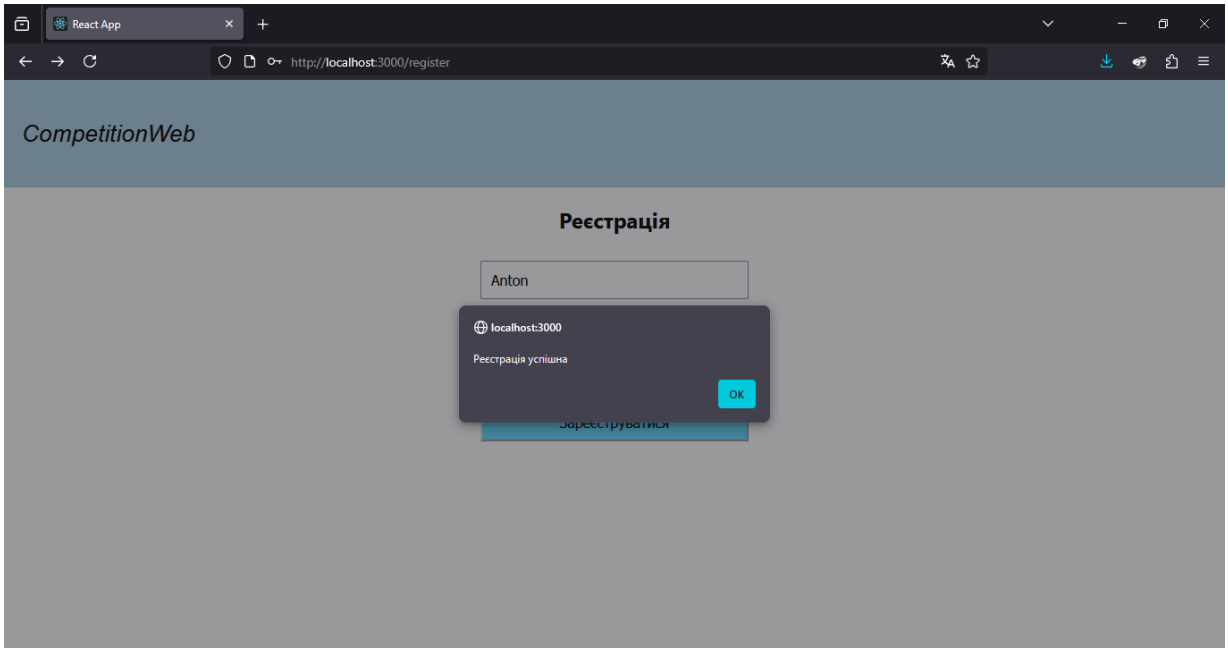

Рисунок 3.2 – Успішна реєстрація на вебсайт»

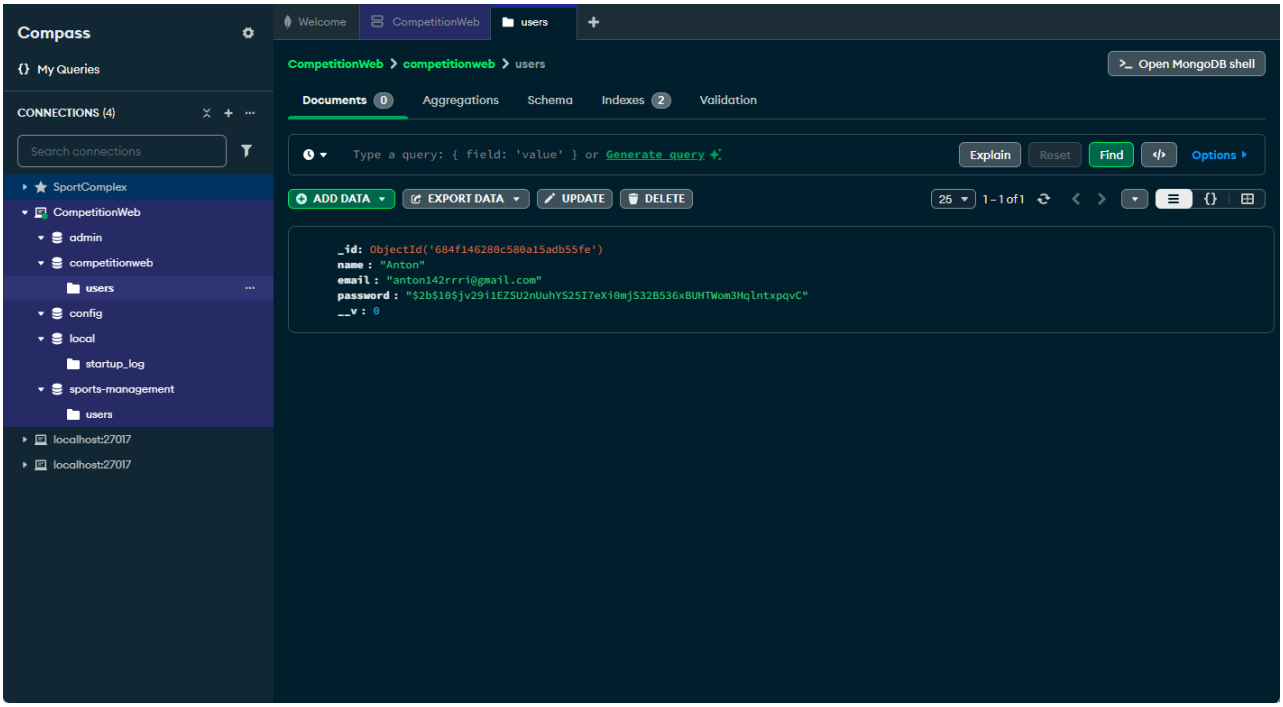

Рисунок 3.3 – Занесені дані зареєстрованого користувача в базу даних

Якщо увійти з даними які внесені в базу даних (рис. 3.4), виведе alert з успішним входом (рис. 3.5), але якщо ввести дані, які не було занесено в базу даних, виведе alert з помилкою (рис. 3.6).

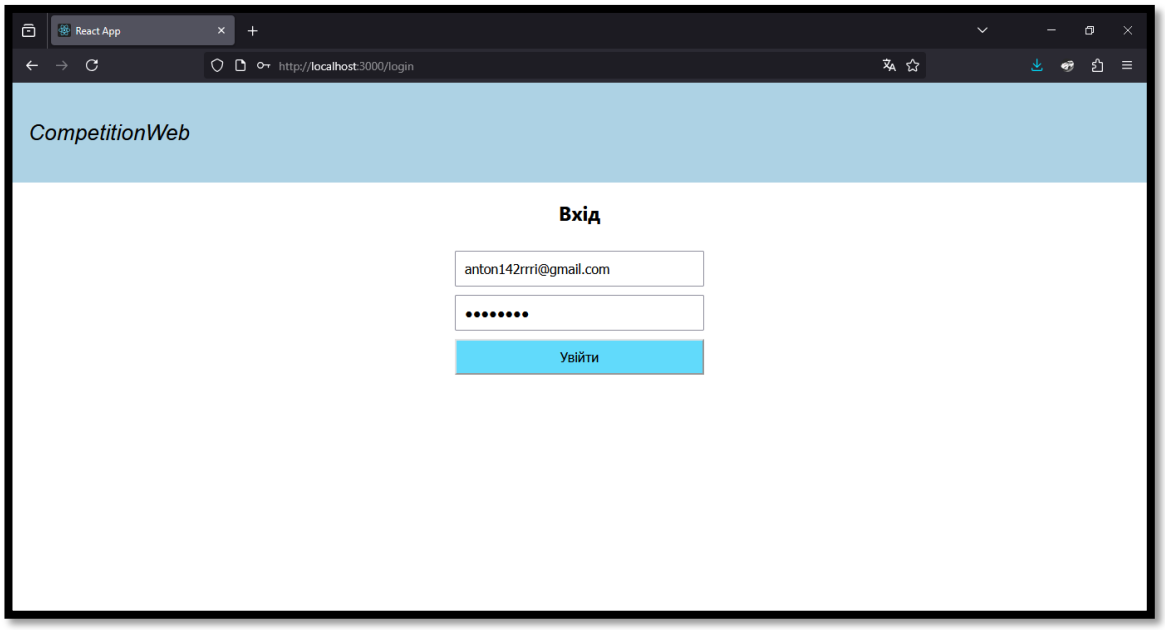

Рисунок 3.4 – Вхід на вебсайт з даними, які занесені в БД

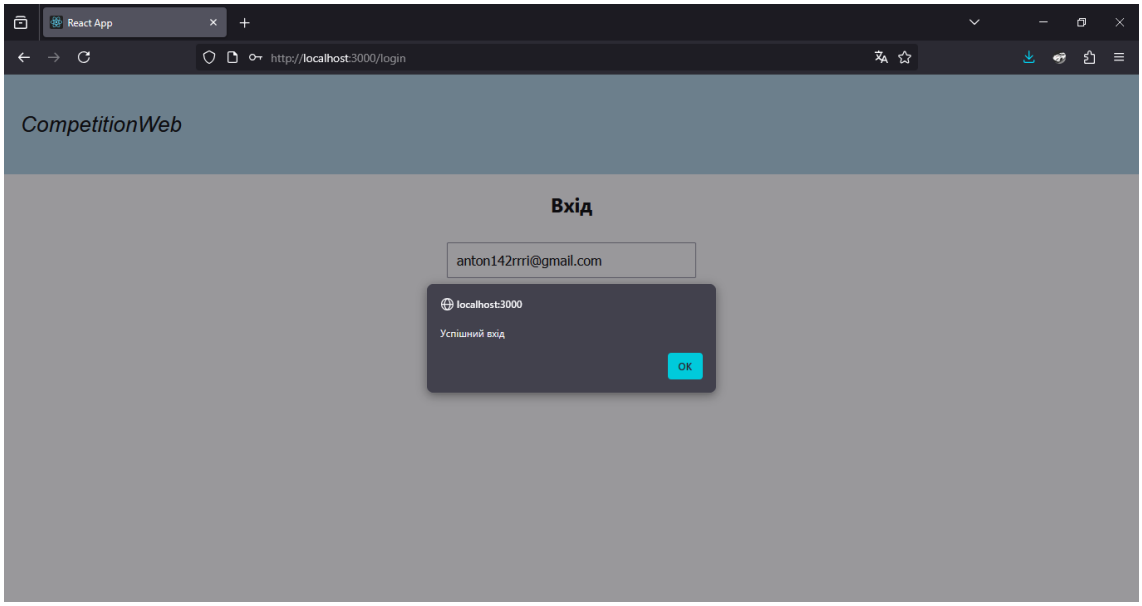

Рисунок 3.5 – Успішний вхід на вебсайт

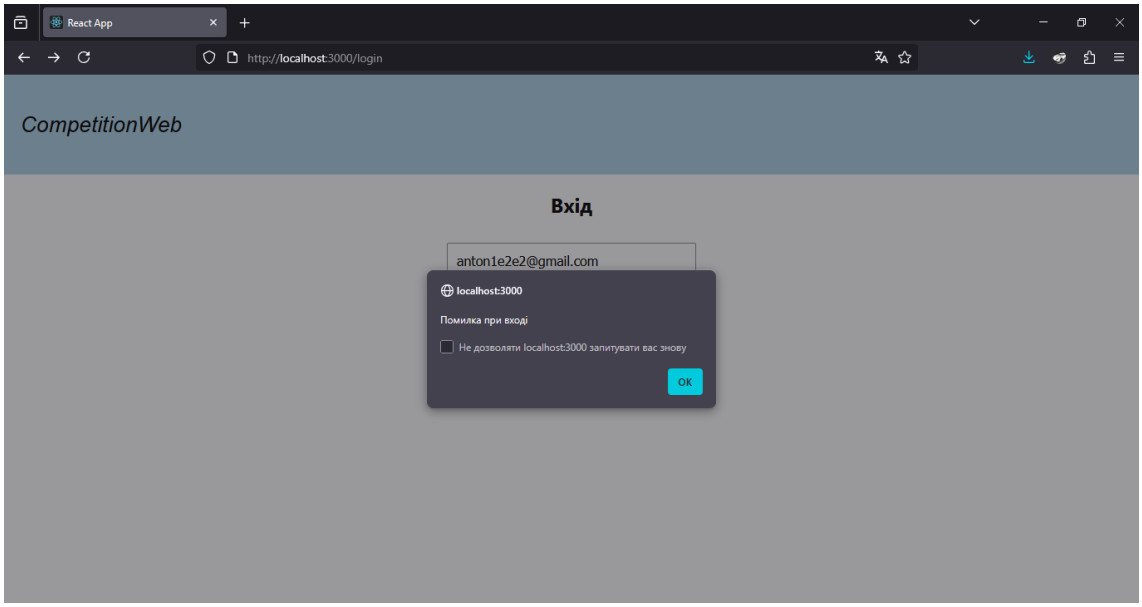

Рисунок 3.6 – Помилка входу на вебсайт

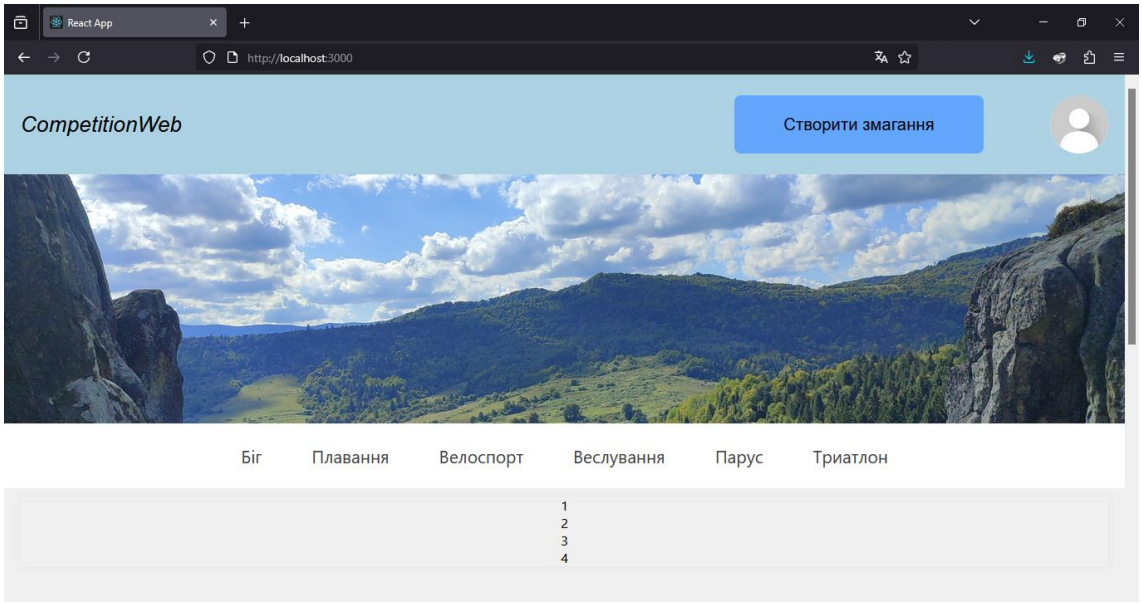

Рисунок 3.7 – Головна сторінка вебсайту

Для тестування було змодельовано відповідь сервера за допомогою Jest (лістинг 3.1).

```
import React from 'react';
import { render, screen, waitFor } from '@testing-library/react';
import App from './App';

// Мокаємо fetch, щоб імітувати відповідь бекенду
beforeEach(() => {
  global.fetch = jest.fn(() =>
    Promise.resolve({
      json: () => Promise.resolve({ message: "Тестове повідомлення" }),
    })
  );
});

test('відображає повідомлення з бекенду', async () => {
  render(<App />);

  // Перевірка, що показується Loading data
  expect(screen.getByText(/Loading data/i)).toBeInTheDocument();
```

```
  // Повинно з'явитися повідомлення з fetch
  const messageElement = await waitFor(() =>
    screen.getByText("Тестове повідомлення")
  );

  expect(messageElement).toBeInTheDocument();
});
```

Лістинг 3.1 Файл App.test.js. Перевірка на отримання та відображення даних з бекенду

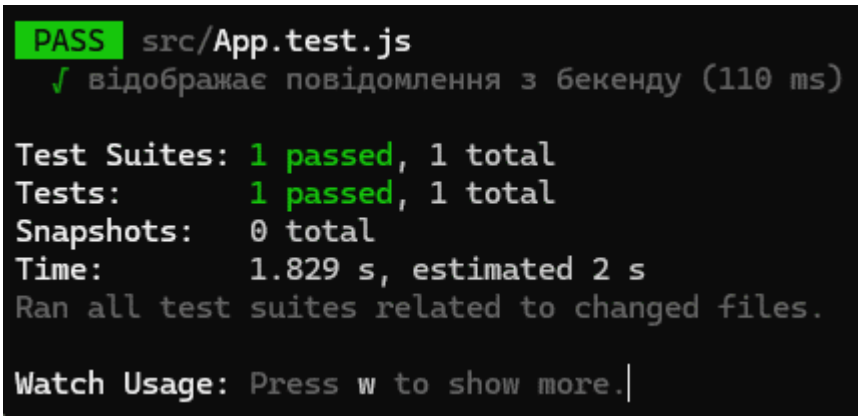

Рисунок 3.8 – Результат виконання тесту

3.2. Аналіз результатів тестування

На основі проведеного тестування макетів та описаних сценаріїв можна зробити висновок, що запланована функціональність вебзастосунку повністю відповідає його призначенню. Було проаналізовано логіку переходів між сторінками, коректність структури інтерфейсу, а також передбачено реакцію системи на можливі помилки з боку користувача (некоректний пароль, порожні поля, дублікати пошти тощо). Окрему увагу приділено розмежуванню прав доступу: лише організатор має можливість редагувати результати змагань, тоді як інші користувачі можуть лише переглядати опубліковану інформацію.

3.3. Рекомендації щодо впровадження та використання

Розроблений вебзастосунок може бути ефективно використаний для організації та управління спортивними змаганнями на будь-якому рівні: початковому або професійному. Одним із пріоритетних напрямів використання системи є морське багатоборство - складний багатофункціональний вид спорту, що поєднує декілька дисциплін, таких як біг, плавання, веслування, вітрильний спорт. Враховуючи наявність різноманітних категорій, заліків та змінної кількості учасників, запропонований вебсайт може значно спростити процес реєстрації, формування груп та підрахунку результатів. Вебсайт може бути розгорнутий як на локальному сервері для внутрішніх потреб клубу чи федерації, так і на загальнодоступному хостингу для публічного доступу спортсменів і глядачів. Застосунок доцільно впроваджувати в морські багатоборства, триатлон, змагання з веслування та плавання, а також інші подібні заходи, які вимагають зручного цифрового інструменту для обліку, координації та інформування учасників.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було реалізовано вебдодаток для організації спортивних змагань з використанням стеку технологій React, Node.js та MongoDB. Вебсайт дозволяє створювати змагання, переглядати категорії видів спорту та забезпечує базову інтеграцію між фронтендом і бекендом. Під час реалізації були вивчені особливості налаштування середовища розробки, роутінгу, обміну даними між компонентами та серверною частиною. Здійснено часткове тестування функціональності, проведено візуалізацію інтерфейсу в Figma. Попри те, що частина функціоналу залишилася на рівні макетів та заготовок, проєкт демонструє цілісну архітектуру вебсистеми і є придатним для подальшого розвитку, інтеграції нових функцій і використання в реальних умовах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Голець А.В., Немченко В.Ю.Розробка веб-платформи для організації та управління спортивними змаганнями. Матеріали XVII Студентської науково-практичної конференції студентів, аспірантів та молодих вчених за тематикою «Тенденції розвитку ІТ-технологій в Україні». 26-27 березня 2025 р., Черкаси, Україна. Черкаси : Черкаський державний фаховий бізнес-коледж, 2025. С. 122-123.

2. Матеріали з HTML, CSS, JavaScript - навчальний веб-сайт W3Schools. - Режим доступу: https://www.w3schools.com/ (дата звернення: 23.02.2025).

3. Node.js Documentation - офіційна документація платформи Node.js. - Режим доступу: https://nodejs.org/en/docs/ (дата звернення: 04.03.2025).

4. Express.js Guide - офіційний посібник по фреймворку Express.js для Node.js. - Режим доступу: https://expressjs.com/ (дата звернення: 04.03.2025).

5. MongoDB Manual - офіційна документація системи управління базами даних MongoDB. - Режим доступу: https://www.mongodb.com/docs/manual/ (дата звернення: 05.03.2025).

6. React Router Documentation - офіційна документація бібліотеки React Router для React.js. - Режим доступу: https://reactrouter.com/ (дата звернення: 06.03.2025).

7. Документація до бібліотеки Axios - офіційний сайт бібліотеки для HTTP-запитів Axios. - Режим доступу: https://axios-http.com/ (дата звернення: 08.03.2025).

8. Run Ukraine - офіційний сайт організаторів бігових заходів в Україні. - Режим доступу: https://runukraine.org (дата звернення: 12.05.2025).

9. ВсеПробіги - інформаційний портал про пробіги та спортивні події в Україні. - Режим доступу: https://vseprobegi.org (дата звернення: 12.05.2025).

10. Федерація плавання України - офіційний сайт федерації плавання України. -Режим доступу: usf.org.ua (дата звернення: 12.05.2025).

11. SwimMeet Software - платформа для організації змагань з плавання. - Режим доступу: https://swim-meet.com (дата звернення: 12.05.2025).

12. World Rowing - офіційний сайт міжнародної федерації академічного веслування. - Режим доступу: https://worldrowing.com (дата звернення: 12.05.2025).

13. World Sailing - офіційний сайт міжнародної федерації парусного спорту. - Режим доступу: https://www.sailing.org/ (дата звернення: 12.05.2025).

14. Let's Do This - платформа для пошуку і реєстрації на спортивні заходи по всьому світу. - Режим доступу: https://www.letsdothis.com/us (дата звернення: 12.05.2025).

15. Stack Overflow - популярний ресурс для програмістів з відповідями на технічні питання. - Режим доступу: https://stackoverflow.com/ (дата звернення: 12.05.2025).

ДОДАТКИ

ДОДАТОК А. Серверна частина

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const dotenv = require('dotenv');

dotenv.config();

const app = express();
const PORT = process.env.PORT || 5000;

mongoose.connect(process.env.MONGO_URI, {
  useNewUrlParser: true,
  useUnifiedTopology: true
})
.then(() => console.log("MongoDB підключено"))
.catch((err) => console.error("Помилка MongoDB:", err));

app.use(cors());
app.use(express.json());

const authRoutes = require('./routes/auth');
app.use('/api/auth', authRoutes);

app.get('/api', (req, res) => {
  res.json({ message: "Сервер працює" });
});

app.listen(PORT, () => {
  console.log(`Server on port ${PORT}`);
});
```

Лістинг А.1 «index.js»

```
const mongoose = require('mongoose');

const UserSchema = new mongoose.Schema({
  name: { type:String, required: true },
  email: { type: String, required: true, unique: true },
  password: { type:String, required: true },
});

module.exports = mongoose.model('User', UserSchema);
```

Лістинг А.2 «User.js»

```
const express = require('express');
const router = express.Router();
const bcrypt = require('bcryptjs');
const User = require('../models/user');

// Реєстрація
router.post('/register', async (req, res) => {
  const { name, email, password } = req.body;

  try {
    const existingUser = await User.findOne({ email });
    if (existingUser) return res.status(400).json({ message: 'Користувач з такою поштою
вже існує' });

    const hashedPassword = await bcrypt.hash(password, 10);
    const newUser = new User({ name, email, password: hashedPassword });

    await newUser.save();
    res.status(201).json({ message: 'Реєстрація успішна' });
  } catch (err) {
    res.status(500).json({ message: 'Помилка при реєстрації' });
  }
});

// Вхід
router.post('/login', async (req, res) => {
```

```
  const { email, password } = req.body;

  try {
    const user = await User.findOne({ email });
    if (!user) return res.status(400).json({ message: 'Користувача не знайдено' });

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch) return res.status(400).json({ message: 'Невірний пароль' });

    res.json({ message: `Вхід успішний, вітаємо ${user.name}` });
  } catch (err) {
    res.status(500).json({ message: 'Помилка при вході' });
  }
});

module.exports = router;
```

Лістинг А.3 «auth.js»

ДОДАТОК В. Клієнтська частина

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';

function Register() {
  const navigate = useNavigate();
  const [formData, setFormData] = useState({
    name: '', email: '', password: ''
  });

  const handleChange = e => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const res = await fetch('/api/auth/register', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(formData)
      });
      const data = await res.json();
      alert(data.message);
      if (res.ok) navigate('/');
    } catch (err) {
      alert("Помилка при реєстрації");
    }
  };

  return (
    <div className="form-container">
      <h2>Реєстрація</h2>
      <form onSubmit={handleSubmit}>
        <input name="name" placeholder="Ім’я" onChange={handleChange} required />
```

```
        <input name="email" type="email" placeholder="Email" onChange={handleChange}
required />
        <input name="password" type="password" placeholder="Пароль"
onChange={handleChange} required />
        <button type="submit">Зареєструватися</button>
      </form>
    </div>
  );
}

export default Register;
```

Лістинг В.1 «Register.js»

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';

function Login() {
  const navigate = useNavigate();
  const [formData, setFormData] = useState({
    email: '', password: ''
  });

  const handleChange = e => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const res = await fetch('/api/auth/login', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(formData)
      });
      const data = await res.json();
      alert(data.message);
      if (res.ok) navigate('/');
```

```
    } catch (err) {
      alert("Помилка при вході");
    }
  };

  return (
    <div className="form-container">
      <h2>Вхід</h2>
      <form onSubmit={handleSubmit}>
        <input name="email" type="email" placeholder="Email" onChange={handleChange}
required />
        <input name="password" type="password" placeholder="Пароль"
onChange={handleChange} required />
        <button type="submit">Увійти</button>
      </form>
    </div>
  );
}

export default Login;
```

Лістинг В.2 «Login.js»

```
import React, { useState, useEffect } from 'react';
import { Link } from 'react-router-dom';

function Home() {
  const [data, setData] = useState(null);

  useEffect(() => {
    fetch('/api')
      .then(response => response.json())
      .then(response => setData(response.message));
  }, []);

  return (
    <div className="App">
      <header className="App-header">
```

```
        <div className="headerName"><i>CompetitionWeb</i></div>
        <div className="button-and-avatar">
          <Link to="/register" className="button-create-competition">Реєстрація</Link>
          <Link to="/login" className="button-create-competition">Вхід</Link>
        </div>
        { !data ? 'Loading data...' : data }
      </header>
    </div>
  );
}

export default Home;
```

Лістинг В.3 «Home.js»