

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## **ПОЯСНЮВАЛЬНА ЗАПИСКА**

до кваліфікаційної роботи освітнього ступеня «бакалавр»  
за спеціальністю 121 «Інженерія програмного забезпечення»  
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Сайт тренажерного залу»**

Виконав студент групи ПЗ-20-2

Гарбар Дмитрій Сергійович

Керівник роботи:

Єфремов Юрій Миколайович

Рецензент:

Дунєв Сергій Сергійович

Житомир – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

**«ЗАТВЕРДЖУЮ»**

Зав. кафедри інженерії  
програмного забезпечення

Тетяна Вакалюк

**«26» лютого 2024 р.**

**ЗАВДАННЯ**  
**на кваліфікаційну роботу**

Здобувач вищої освіти: **Гарбар Дмитрій Сергійович**

Керівник роботи: **Єфремов Юрій Миколайович**

Тема роботи: **«Сайт тренажерного залу»**,

затверджена Наказом закладу вищої освіти від **«23» лютого 2024 р., №74/с**

Вихідні дані для роботи: **Сайт тренажерного залу**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів,  
що їх стосуються:

| Розділ | Консультант  | Завдання видав | Завдання прийняв |
|--------|--------------|----------------|------------------|
| 1      | Єфремов Ю.М. | 27.02.2024р.   | 27.02.2024р.     |
| 2      | Єфремов Ю.М. | 27.02.2024р.   | 27.02.2024р.     |
| 3      | Єфремов Ю.М. | 27.02.2024р.   | 27.02.2024р.     |

## РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з сайту тренажерного залу та пояснювальної записки. Пояснювальна записка до випускної роботи містить 59 сторінок, 45 ілюстрацій та 12 таблиць.

Метою роботи є реалізація зручного додатку для спортивного залу звикористанням актуальних технологій та інтеграцією з базою даних для зберіганням даних.

Сайт побудований на монолітній архітектурі , що забезпечує простоту моніторингу та налагодження проекту. Для тестування та поліпшення продуктивності ми можемо використовувати стандартні інструменти розробки.

Додаток має зручний, лаконічний інтерфейс, який дозволяє користувачам легко обирати тренерів, групи, взаємодіяти з абонементом.

**КЛЮЧОВІ СЛОВА:** АБОНЕМЕНТ, МОНОЛІТ, САЙТ, ТРЕНАЖЕРНИЙ ЗАЛ, ТРЕНЕР

|           |              |          |        |      |                        |                                            |      |         |
|-----------|--------------|----------|--------|------|------------------------|--------------------------------------------|------|---------|
|           |              |          |        |      | ІПЗ.КР.Б-121-24-ПЗ     |                                            |      |         |
| Змн       | Арк.         | № докум. | Підпис | Дата |                        |                                            |      |         |
| Розроб.   | Гарбар Д.С.  |          |        |      | Сайт тренажерного залу | Літ.                                       | Арк. | Аркушів |
| Керівник  | Єфремов Ю.М. |          |        |      |                        |                                            | 4    | 78      |
|           |              |          |        |      |                        | Житомирська політехніка,<br>група ІПЗ-20-2 |      |         |
| Рецензент | Дунев С.С.   |          |        |      |                        |                                            |      |         |
| Зав. каф. | Вакалюк Т.А. |          |        |      |                        |                                            |      |         |

## ABSTRACT

The Bachelor's graduation qualification work consists of a web application for a gym and an explanatory note. The explanatory note for the graduation work contains 59 pages, 45 illustrations, and 12 tables.

The purpose of the work is to implement a convenient application for the gym using current technologies and integrating with a database for data storage.

The site is built on a monolithic architecture, which ensures simplicity in monitoring and debugging the project. Standard development tools can be used for testing and performance improvement.

The application has a convenient, concise interface that allows users to easily choose trainers, groups, and interact with subscriptions.

**KEYWORDS:** SUBSCRIPTION, MONOLITH, PLATFORM, GYM, TRAINER.

|           |              |          |        |      |                        |                                            |      |         |
|-----------|--------------|----------|--------|------|------------------------|--------------------------------------------|------|---------|
|           |              |          |        |      | ІПЗ.КР.Б-121-24-ПЗ     |                                            |      |         |
|           |              |          |        |      |                        |                                            |      |         |
| Змн       | Арк.         | № докум. | Підпис | Дата | Сайт тренажерного залу | Літ.                                       | Арк. | Аркушів |
| Розроб.   | Гарбар Д.С.  |          |        |      |                        |                                            | 4    | 78      |
| Керівник  | Єфремов Ю.М. |          |        |      |                        |                                            |      |         |
| Рецензент | Дунев С.С.   |          |        |      |                        | Житомирська політехніка,<br>група ІПЗ-20-2 |      |         |
| Зав. каф. | Вакалюк Т.А. |          |        |      |                        |                                            |      |         |

## ЗМІСТ

|                                                                                       |           |
|---------------------------------------------------------------------------------------|-----------|
| РЕФЕРАТ .....                                                                         | 3         |
| ABSTRACT .....                                                                        | 4         |
| ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....                                                        | 6         |
| ВСТУП .....                                                                           | 7         |
| <b>РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ САЙТУ</b>                                  |           |
| <b>ТРЕНАЖЕРНОГО ЗАЛУ .....</b>                                                        | <b>9</b>  |
| 1.1 Постановка задачі .....                                                           | 9         |
| 1.2 Аналіз аналогів програмного продукту.....                                         | 10        |
| 1.3 Визначення варіантів використання та об'єктно-орієнтованої структури системи..... | 14        |
| Висновки до першого розділу .....                                                     | 18        |
| <b>РОЗДІЛ 2. ПРОЕКТУВАННЯ САЙТУ СПОРТИВНОГО ЗАЛУ .....</b>                            | <b>19</b> |
| 2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи..... | 19        |
| 2.2 Розробка бази даних системи .....                                                 | 25        |
| 2.3 Проектування та реалізація алгоритмів роботи системи.....                         | 27        |
| 2.4 Реалізація функціоналу сайту спортивного залу.....                                | 28        |
| Висновки до другого розділу: .....                                                    | 37        |
| <b>РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З САЙТОМ</b>                                 |           |
| <b>ТРЕНАЖЕРНОГО ЗАЛУ .....</b>                                                        | <b>38</b> |
| 3.1 Проектування інтерфейсу обробки даних.....                                        | 38        |
| 3.2 Структура інтерфейсу сайту тренажерного залу.....                                 | 39        |
| 3.3 Тестування сайту .....                                                            | 47        |
| Висновки до третього розділу .....                                                    | 54        |
| <b>ВИСНОВКИ.....</b>                                                                  | <b>56</b> |
| <b>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....</b>                                                | <b>57</b> |
| <b>ДОДАТКИ.....</b>                                                                   | <b>60</b> |

## ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API – прикладний програмний інтерфейс

БД – База даних

СУБД – система управління базами даних

IDE – Інтегроване середовище розробки

MVC – схема розділення даних додатків і керуючої логіки на три окремі компоненти Model-View-Controller

VIP – Very Important Person. Людина яка має привілегії.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 6    |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

## ВСТУП

**Актуальність теми.** В сучасному світі майже неможливо уявити життя без веб-технологій. Кожна компанія, магазин, університет мають власні сайти, адже це дуже важлива складова, навіть для маленьких компаній. Сайт спрощує доступ до інформації, або послуги певної компанії чи магазину. Наприклад, просто зайшовши на сайт, ми можемо дізнатися адресу магазину, контактні дані, ціни, види послуг та зробити інтернет-замовлення. Сайт тренажерного залу закриває перераховані задачі для конкретного закладу. Набагато легше зайти на сайт і ознайомитись з профілями тренерів, щоб обрати найкращого для себе.

**Метою** випускної кваліфікаційної роботи бакалавра є розробка сайту тренажерного залу з використанням сучасних технологій веб-розробки. Цей сайт буде включати інтерактивні функції, такі як онлайн-розклад тренувань, реєстрація та бронювання місць, перегляд інформації про тренерів та тренування, а також можливість взаємодії з користувачами через коментарі та відгуки.

Для розробки сайту тренажерного залу нам необхідно:

1. Провести детальний аналіз ринку: дослідити функціональні можливості та недоліки існуючих сайтів тренажерних залів. Це допоможе виявити, які функції потребують вдосконалення та які нові можливості можуть бути корисними для користувачів.
2. Вибрати оптимальні технології: визначити найефективніші інструменти для розробки фронтенду та бекенду, а також обрати відповідну базу даних. Це дозволить забезпечити надійну, стабільну та продуктивну роботу сайту.
3. Розробити сайт на основі зібраних даних: створити зручний користувацький інтерфейс, впровадити ключові функції, такі як розклад тренувань, реєстрація та бронювання місць, перегляд профілів

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 7    |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

тренерів та тренувань. Особлива увага буде приділена забезпеченню безпеки даних користувачів.

4. Оптимізувати і тестувати сайт: використати сучасні методи оптимізації для підвищення швидкодії та масштабованості сайту. Провести комплексне тестування всіх функціональних можливостей для гарантування стабільної та надійної роботи системи.

**Об'єктом дослідження** є процес створення сайту тренажерного залу з нуля до повного функціоналу, використовуючи популярні технології веб-розробки. Дослідження охоплює всі етапи розробки: від початкового планування та проектування архітектури системи до написання коду та тестування. Процес включає розробку адаптивного інтерфейсу користувача, реалізацію функціональності для взаємодії з користувачами, включаючи реєстрацію, бронювання місць на тренування, перегляд розкладів та відгуків, а також забезпечення безпеки даних та швидкодії системи.

**Предметом дослідження** є можливості і актуальність стеку MERN(MongoDB, Express, React, Node).

Досліджується складність розробки та використання як окремих функцій, так і їх сукупність в рамках описаного стеку на основі сайту тренажерного залу.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 8    |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

# РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ САЙТУ ТРЕНАЖЕРНОГО ЗАЛУ

## 1.1 Постановка задачі

**Метою** випускної кваліфікаційної роботи бакалавра є розробка сайту тренажерного залу з використанням сучасних технологій.

Тренажерні зали були популярні весь час. Перші зали з'явилися близько 19 століття. В той час у залів не було якихось спеціалізацій. Люди займалися базовою фізкультурою, проте вони займалися в одному приміщенні, що єднало і змушувало людей змагатися в різних дисциплінах.

На даний момент існує багато спортзалів різних спеціалізацій зі своїми тренажерами, які підходять для вузьких напрямлень. Найефектнішим видом спорту, який з'явився через тренажерні зали це Бодібілдинг. Найкращим прикладом впливу залу на цей спорт є Ронні Колеман – найвідоміший бодібілдер у світі, володар 8 медалей “Містер Олімпія”.

Наш сайт спеціалізується на фітнесі, тобто тренування та вправи, направлені на підтримку рівня життя. Додаток повинен мати декілька важливих функцій, щоб конкурувати з іншими сайтами:

- Управління тренуваннями: створення і відстеження індивідуальних та групових тренувальних програм для клієнтів, включаючи графіки занять, реєстрацію на заняття та моніторинг прогресу.
- Облік клієнтів: автоматизоване ведення бази даних клієнтів, включаючи реєстрацію нових членів, оновлення інформації про існуючих клієнтів, а також управління абонементом.
- Фінансовий облік: управління фінансовими транзакціями, включаючи облік платежів за абонементи, витрат на обслуговування та закупівлю обладнання.
- Аналітика та звітність: збір та аналіз даних про відвідуваність, ефективність тренувань та фінансові показники, що дозволяє приймати обґрунтовані управлінські рішення.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 9    |

В результаті маємо створити сайт на якому будь-який користувач зможе створити акаунт для відвідування спортзалу, отримати максимально детальну інформацію про вид тренувань, тренерів, наявні абонементи.

Розіб'ємо розробку на етапи:

1. Розробка дизайну сайту

- Створення базового стилю для всього додатку
- Створення різних UI компонентів
- Персоналізований дизайн для сторінок

2. Побудова архітектури БД

3. Планування серверної частини

4. Розробка основного функціоналу

- Авторизація користувача
- Панель адміністратора
- Створення та редагування груп
- Створення та редагування курсів
- Створення та реалізація абонементів

5. Поєднання бекенду і фронтенду

6. Тестування додатку

В результаті дотримання плану маємо сайт тренажерного залу який дозволяє записатись в зал на основі отриманої інформації про заняття, тренерів, абонементи, локацію.

## 1.2 Аналіз аналогів програмного продукту.

При розробці сайту тренажерного залу, потрібно провести детальний аналіз конкурентів. Проведемо дослідження залів різної величини, різних країн, щоб обрати найкращі функції та функції, яких немає на більшості сайтів, та побудимось більшості поганих та недоречних функцій, які заважають роботі сайту, унеможлиблюють користування інтерфейсом, або погано впливають на користувацький досвід. Нами було обрано чотири популярних тренажерних залів.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 10   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

Одинм з сайтів є SportLife(рис. 1.1).

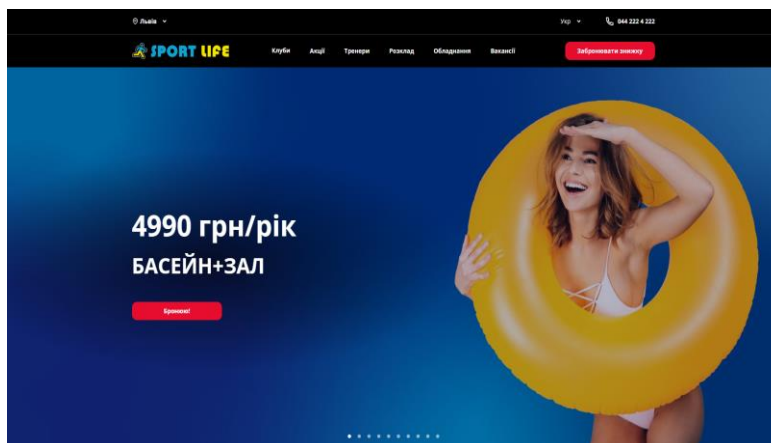


Рис. 1.1. Сайт тренажерного залу SportLife[1]

Наведений сайт є однією з найкращих в м. Житомир. Їх сайт надає користувачу всю потрібну інформацію.

Основними перевагами SportLife є:

- Сучасний приємний дизайн
- Великий вибір абонементів
- Детальна інформація про тренерів
- Наявні акційні пропозиції

З недоліків можна визначити:

- Відсутність інформації про групи, якими керує тренер
- Перевантажений інтерфейс
- Відсутність освітньої інформації для самостійного тренування

ArenaGym – досить популярний зал, який обирають за різноманіття і якість тренажерів(рис. 1.2).

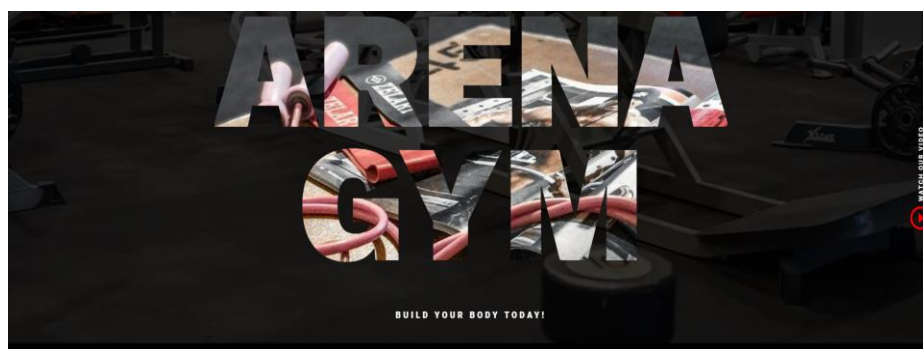


Рис. 1.2. Сайт тренажерного залу ArenaGym[2]

Перевагами цього сайту є:

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 11   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

- Цікаві дизайнерські рішення

Проте сайт має низку недоліків:

- Не інформативна головна сторінка
- Наявність російської мови
- Відсутність інформації про тренерів, групи, розклад занять
- Відсутня інформація про абонементи чи інші послуги

Gold's Gym – Дуже популярний світовий бренд спортивних залів. Зал має багату історію та традиції(рис. 1.3).

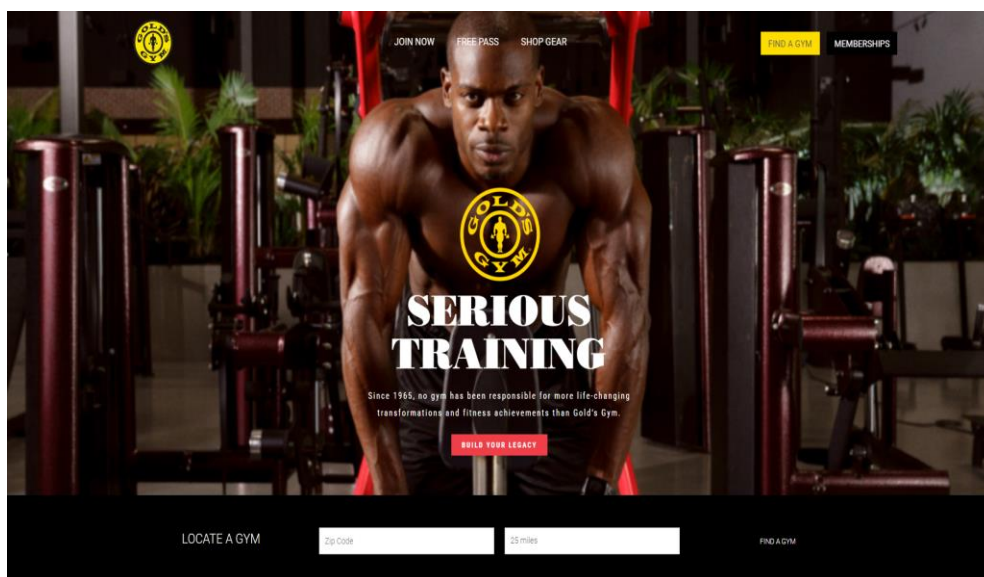


Рис. 1.3. Сайт тренажерного залу Gold's Gym[3]

Сайт має декілька переваг серед конкурентів:

- Акційні пропозиції
- Можливість знайти найближчий до нас зал
- Тематичний магазин

З недоліків можна зазначити:

- Мінімалістичний дизайн
- Відсутність інформації для клієнта

Lime Fitness – спортивний зал в м. Житомир. Дуже інформативний сайт з усім функціоналом потрібним для користувача(рис.1.4)

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 12   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

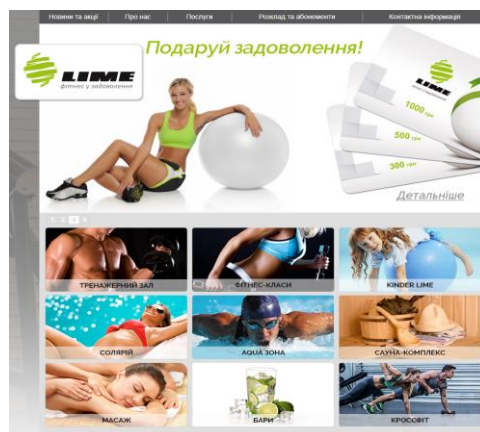


Рис. 1.4. Сайт тренажерного залу Lime Fitness[4]

Серед переваг:

- Інформативний дизайн сайту
- Наявність розкладу
- Новини та акції

Щодо недоліків:

- Застарілий інтерфейс
- Відсутність інформації про тренерів

На основі проведених нами досліджень можемо скласти порівняльну таблицю сайтів за деякими пунктами(табл. 1.1).

Таблиця 1.1

Таблиця порівняння сайтів тренажерних залів

| Характеристика                       | SportLife | ArenaGym | Gold's Gym | Lime Fitness |
|--------------------------------------|-----------|----------|------------|--------------|
| Сучасність дизайну                   | +         | +-       | -          | -            |
| Наявність інформації про тренерів    | +         | -        | -          | -            |
| Акції                                | +         | -        | +          | +            |
| Наявність абонементів                | +         | -        | +          | +            |
| Зручність інтерфейсу                 | +         | -        | -          | +            |
| Додаткова інформація для користувача | +         | -        | +          | -            |
| Додаткові особливості сайту          | -         | +        | +          | -            |

Переглянувши сайти залів конкурентів, можна зазначити основні відмінності: деякі сайти дозволяють вивчати всю інформацію безпосередньо перед походом в зал. Інші дають базову інформацію для заохочення. Основний функціонал всіх сайтів однаковий, відрізняються лише якісь особливості дизайну.

Враховуючи інформацію яку ми отримали в ході дослідження, було вирішено створити сайт який буде відповідати сукупності функцій наведених залів та введення власних функцій. Серед основних: Авторизація, присвоєння ролей, Списки груп, тренувань, тренрів та детальна інформація про них. Сайт який ми реалізуємо має бути сучасним, функціональним. Доцільний дизайн та інтуїтивно зрозумілий інтерфейс допоможуть користувачам отримати легкий доступ до інформації на рахунок будь-яких питань, можливість керування своїм профілем, зручність контакту з тренерами.

### **1.3 Визначення варіантів використання та об'єктно-орієнтованої структури системи.**

Додаток буде побудований на монолітній архітектурі. Серверна частина буде написана з використанням сучасного фреймворку для JavaScript – NodeJS.

Для фронтенду буде використано фреймворк ReactJS – відкрита JavaScript бібліотека для створення інтерфейсів користувача, яка покликана вирішувати проблеми часткового оновлення вмісту вебсторінки, з якими стикаються в розробці односторінкових застосунків. Базою даних було обрано MongoDB.

**Node.js** — це сайт для виконання JavaScript-коду на сервері, заснована на рушію V8 від Chrome. Вона набрала популярності завдяки своїм унікальним можливостям та архітектурі, що дозволяють створювати масштабовані та ефективні мережеві додатки.

Сильні сторони NodeJS:

*Висока продуктивність.* Node.js використовує асинхронну, подієво-орієнтовану архітектуру, що дозволяє обробляти багато одночасних запитів без

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 14   |

блокування вводу-виводу. Це робить його ідеальним для створення реальних додатків, таких як чати та ігри, які потребують швидкого оновлення даних [5] [6].

*Масштабованість.* Завдяки легкій та модульній архітектурі Node.js легко масштабувати додатки як горизонтально, так і вертикально. Це дозволяє додавати більше мікросервісів у вже існуючі додатки, забезпечуючи їхню гнучкість та стійкість [7].

*Використання єдиної мови програмування.* Node.js дозволяє розробникам використовувати JavaScript як на стороні клієнта, так і на стороні сервера. Це спрощує процес розробки, оскільки розробники можуть використовувати одну мову для всього проекту, що також сприяє швидшому навчанню та адаптації нових членів команди [7].

*Підтримка численних інструментів.* Node.js має розширену підтримку різних інструментів, таких як npm (Node Package Manager), що дозволяє легко тестувати та управляти залежностями проекту [7].

Node.js є потужним інструментом для розробки сучасних сайтів, завдяки його високій продуктивності, масштабованості, можливості використовувати єдину мову програмування на клієнті та сервері, а також активній спільноті та підтримці численних інструментів.

**React JS**, або просто React, є однією з найпопулярніших бібліотек для створення користувацьких інтерфейсів на JavaScript. Розроблена та підтримувана Facebook, вона вперше була представлена у 2013 році.

Переваги React JS[8][9][10]:

*Компонентний підхід:* React використовує компонентний підхід до розробки, що означає, що користувацький інтерфейс складається з невеликих, повторно використовуваних компонентів. Кожен компонент управляє своїм власним станом та логікою, що сприяє легкості управління кодом та підвищує його повторну використання. Це полегшує розробку та тестування великих додатків, роблячи їх більш організованими та зручними для підтримки.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 15   |

*Віртуальний DOM:* React використовує віртуальний DOM, який є копією реального DOM. Це дозволяє React ефективно оновлювати користувацький інтерфейс, порівнюючи зміни у віртуальному DOM з реальним DOM та мінімізуючи кількість прямих маніпуляцій з ним. Такий підхід значно покращує продуктивність додатків, особливо тих, які мають складні та динамічні інтерфейси.

*Однонаправлений потік даних.* React підтримує однонаправлений потік даних, що спрощує розуміння та відстеження стану додатка. Це допомагає уникнути непередбачуваних змін стану та полегшує налагодження коду. Компоненти отримують дані через пропси та можуть викликати зміни стану за допомогою функцій, переданих їм від батьківських компонентів.

*Потужна екосистема та інструменти.* React має велику екосистему бібліотек та інструментів, таких як React Router для маршрутизації, Redux для управління станом та багато інших. Крім того, React DevTools дозволяє розробникам зручно налагоджувати та аналізувати React додатки.

React JS забезпечує високу продуктивність, зручність у розробці та підтримці додатків, а також має потужну екосистему та активну спільноту. Це робить його чудовим вибором для створення сучасних сайтів.

**Express.js** є одним із найпопулярніших веб-фреймворків для Node.js. Розроблений для спрощення процесу створення серверних додатків, він надає легковагову та гнучку інфраструктуру для побудови сайтів та API.

Переваги Express.js[11][12]:

*Простота та гнучкість.* Express має мінімалістичний підхід до дизайну, що робить його дуже простим у використанні. Він не нав'язує правил або структур, дозволяючи нам обирати та налаштовувати необхідні модулі та середовища.

*Широкий спектр Middleware.* Однією з переваг Express є підтримка middleware – функцій, які виконуються під час обробки запитів. Існує велика кількість готових до використання middleware, що дозволяє розширювати функціональність додатків.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 16   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

*Висока продуктивність.* Завдяки своїй легковаговій природі та ефективній обробці запитів, Express забезпечує високу продуктивність серверних додатків. Використовуючи асинхронну природу Node.js, Express може обробляти великий обсяг одночасних запитів без значних витрат на ресурсах.

*Підтримка REST API.* Express ідеально підходить для створення RESTful API. Завдяки простій обробці маршрутів та запитів, розробники можуть швидко та ефективно створювати масштабовані та надійні API для своїх додатків.

*Легка інтеграція з фронтед-фреймворками.* Express чудово працює з різноманітними фронтед-фреймворками, такими як React, Angular, Vue.js. Це робить його ідеальним вибором для побудови повного стеку додатків, забезпечуючи зручний бекенд для фронтед-компонентів.

Express.js забезпечує легковагову та гнучкий сайт для розробки серверних додатків, що робить його одним з найкращих виборів для створення сучасних сайтів та API.

**MongoDB** – популярна база даних NoSQL, яка надає розробникам можливість зберігати та обробляти великі обсяги даних в гнучкий та ефективний спосіб. MongoDB була створена компанією MongoDB Inc. (раніше відомою як 10gen). Вона стала одним із ключових інструментів для розробників завдяки своїм численним перевагам.

MongoDB використовує документо-орієнтовану модель даних, де дані зберігаються у форматі BSON (бінарна версія JSON). Це дозволяє зберігати складні, вкладені структури даних, що робить MongoDB дуже гнучкою у порівнянні з реляційними базами даних. Замість жорстких схем, які обмежують типи та структуру даних, MongoDB дозволяє зберігати документи з різними структурами в одній колекції.

Завдяки своїй архітектурі MongoDB забезпечує високу швидкість запису та читання даних. Вона оптимізована для роботи з великими наборами даних і може ефективно обробляти запити завдяки індексам, які підтримують різні типи пошуку, включаючи текстовий та геопросторовий.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 17   |

MongoDB має велику кількість сторонніх інструментів та бібліотек, які дозволяють інтегрувати її з різними мовами програмування та фреймворками. Крім того, MongoDB Inc. надає потужну комерційну підтримку та хмарне рішення MongoDB Atlas, що спрощує керування базою даних.

База даних має потужну систему агрегації, яка дозволяє виконувати складні операції обробки даних на сервері. Це включає групування, сортування, фільтрацію та обчислення, що дозволяє розробникам ефективно аналізувати та обробляти дані без необхідності витягувати їх на клієнтську сторону. [13][14]

Підсумовуючи, MongoDB забезпечує гнучкість, продуктивність та масштабованість, що робить її чудовим вибором для сучасних сайтів та великих даних. Завдяки своїм численним перевагам вона стала однією з найпопулярніших баз даних серед розробників по всьому світу.

### **Висновки до першого розділу**

В першому розділі був сформований набір основних технологій і засобів реалізації для розробки сайту. Для створення фронтенду було обрано React, що забезпечує високий рівень інтерактивності та зручність для користувачів. Для бекенду було вирішено використовувати Node.js та Express, які дозволяють створити масштабовану та продуктивну серверну частину. MongoDB була обрана як основна база даних завдяки її гнучкості та можливості роботи з великими обсягами даних. Цей набір технологій дозволить створити надійний, функціональний та сучасний сайт тренажерного залу, який буде відповідати всім встановленим вимогам та очікуванням користувачів.

## РОЗДІЛ 2. ПРОЕКТУВАННЯ САЙТУ СПОРТИВНОГО ЗАЛУ

### 2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи.

Сайт спортивного залу надає можливість користувачу створити акаунт в мережі залу, оглянути інформацію про групи, їх розклад, відповідальних за групи тренерів. Після огляду можна зв'язатись з потрібним тренером і запросити доступ до потрібної групи. Якщо користувач не хоче займатись з тренером для нього є курси по технікам та комплексам тренувань. Користувачі поділяються на дві групи: з абонементом, або без нього. Користувач з абонементом отримує всі курси безкоштовно. Для наглядної взаємодії між елементами додатку було створено Діаграму варіантів використання (рис 2.1).

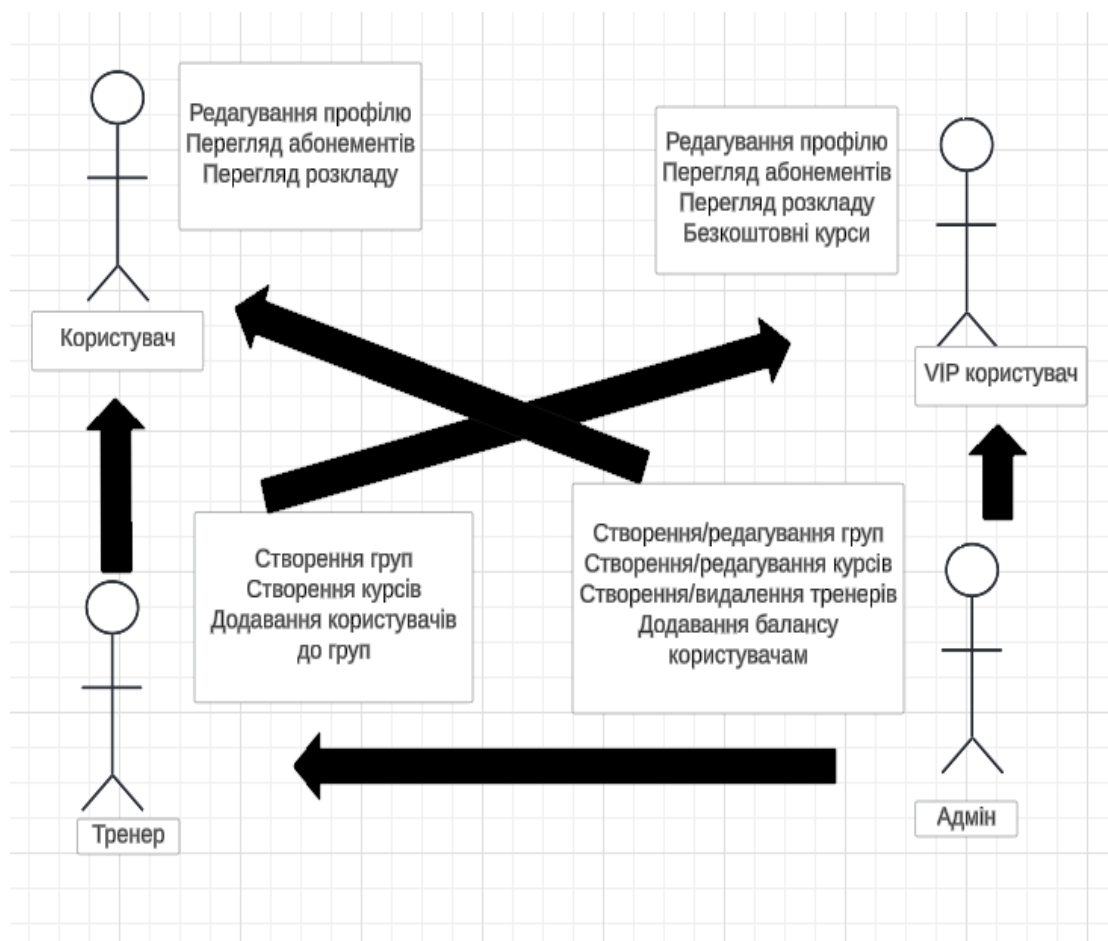


Рис.2.1. Діаграма варіантів використання

Опис діаграми варіантів використання наведено в таблицях 2.1 – 2.8.

Таблиця 2.1

Опис прецеденту «Створення/редагування груп»

| Назва варіанту використання | Створення/редагування груп                                                            |
|-----------------------------|---------------------------------------------------------------------------------------|
| Основні актори              | Адмін, тренер                                                                         |
| Короткий опис               | Адмін або тренер може створити або редагувати групу, в яку може додавати користувачів |

Таблиця 2.2

Опис прецеденту «Створення/редагування курсів»

| Назва варіанту використання | Створення/редагування курсів                                                           |
|-----------------------------|----------------------------------------------------------------------------------------|
| Основні актори              | Адмін, тренер, користувач                                                              |
| Короткий опис               | Адмін або тренер може створити або редагувати курс, доступний для покупки користувачем |

Таблиця 2.3

Опис прецеденту «Створення/видалення тренерів»

| Назва варіанту використання | Створення/видалення тренерів                                       |
|-----------------------------|--------------------------------------------------------------------|
| Основні актори              | Адмін, тренер                                                      |
| Короткий опис               | Адмін може Надавати або забирати права тренера різним користувачам |

Таблиця 2.4

Опис прецеденту «Додавання користувачів до груп»

| Назва варіанту використання | Додавання користувачів до груп                   |
|-----------------------------|--------------------------------------------------|
| Основні актори              | Тренер, користувач                               |
| Короткий опис               | Тренер може додавати учасників до створених груп |

Таблиця 2.5

## Опис прецеденту «Редагування профілю»

|                             |                                                               |
|-----------------------------|---------------------------------------------------------------|
| Назва варіанту використання | Редагування профілю                                           |
| Основні актори              | Авторизований користувач                                      |
| Короткий опис               | Користувач може змінювати дані, які він вводив при реєстрації |

Таблиця 2.6

## Опис прецеденту «Перегляд абонементів»

|                             |                                                    |
|-----------------------------|----------------------------------------------------|
| Назва варіанту використання | Перегляд абонементів                               |
| Основні актори              | Авторизований користувач                           |
| Короткий опис               | Користувач може переглядати та купувати абонементи |

Таблиця 2.7

## Опис прецеденту «Перегляд розкладу»

|                             |                                                        |
|-----------------------------|--------------------------------------------------------|
| Назва варіанту використання | Перегляд розкладу                                      |
| Основні актори              | Авторизований користувач                               |
| Короткий опис               | Користувач може переглядати розклад тренувань для груп |

Таблиця 2.8

## Опис прецеденту «Безкоштовні курси»

|                             |                                                                       |
|-----------------------------|-----------------------------------------------------------------------|
| Назва варіанту використання | Безкоштовні курси                                                     |
| Основні актори              | VIP користувач                                                        |
| Короткий опис               | За наявності VIP абонементу, користувач отримує всі курси безкоштовно |

Основною ціллю створення проекту є надання зручного та ефективного інструменту для користувачів тренажерних залів, який допоможе їм досягати своїх фітнес-цілей, підтримувати мотивацію та організованість. Метою проекту є розробка сучасного сайту, що забезпечує легкий доступ до інформації про тренування, тренерів, розклади занять та управління абонементом. Це включає

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 21   |

створення інтуїтивного інтерфейсу, який дозволяє користувачам швидко знаходити потрібну інформацію, записуватися на тренування, слідкувати за своїм прогресом та взаємодіяти з іншими учасниками спільноти. Інтеграція з базою даних забезпечить надійне зберігання та обробку даних, що дозволить користувачам отримувати персоналізовані рекомендації та підтримку у досягненні їхніх фітнес-цілей.

**Віомги користувачів:**

**Адміністратор:**

1. Надання ролей для тренерів
2. Слідкування за користувачами
3. Створення/видалення груп
4. Створення/видалення курсів

**Користувач:**

1. Авторизація
2. Перегляд курсів
3. Перегляд груп
4. Перегляд/придбання абонементу
5. Контакт з тренером

**Тренер:**

1. Створення/редагування груп
2. Створення/редагування курсів
3. Додавання користувачів до групи

**Функціональні вимоги:**

- Додаток має забезпечувати авторизацію зі збереженням токenu, хешування паролю
- Редагування профілю користувача.
- Створення та редагування тренувальних програм.
- Запис на групові тренування.
- Відображення розкладу занять у зручному форматі.
- Додавання та редагування профілів тренерів.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 22   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

- Призначення тренерів на заняття.

### **Нефункціональні Вимоги:**

#### **1. Час Відповіді Системи:**

- Максимальний час відповіді системи на запит користувача - не більше 2 секунд.

#### **2. Стійкість до Навантаження:**

- Система повинна бути стійкою до навантаження та здатною обслуговувати до 1000 активних користувачів одночасно.

#### **3. Безпека Даних:**

- Захист особистих та фінансових даних користувачів за допомогою шифрування та інших заходів безпеки.

- велика кількість користувачів мають змогу переглядати вебресурс одночасно.

### **Системні вимоги**

#### **1. Вимоги до середовища виконання.**

Сайт повинна задовільняти вимогам на комп'ютері, що знаходиться в наступній конфігурації:

- 4 ГБ оперативної пам'яті
- Процесор з тактовою частотою від 1 GHz
- Операційна система будь-яка, що підтримує використання інтернету
- Браузер
- HTML, JS, SQLITE3

#### **2. Вимоги до СУБД та доступу до даних**

- У ядрі системи повинна бути представлена СУБД реляційного доступу.

#### **3. Вимоги до серверу інформаційної системи:**

- Підтримка сервером технологій NodeJS.

Аналіз функціональних вимог дозволив виділити наступні сутності, що забезпечать програмну реалізацію додатку. На рисунку 2.2. наведено Діаграму

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 23   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

класів, яка використовуються для його роботи.

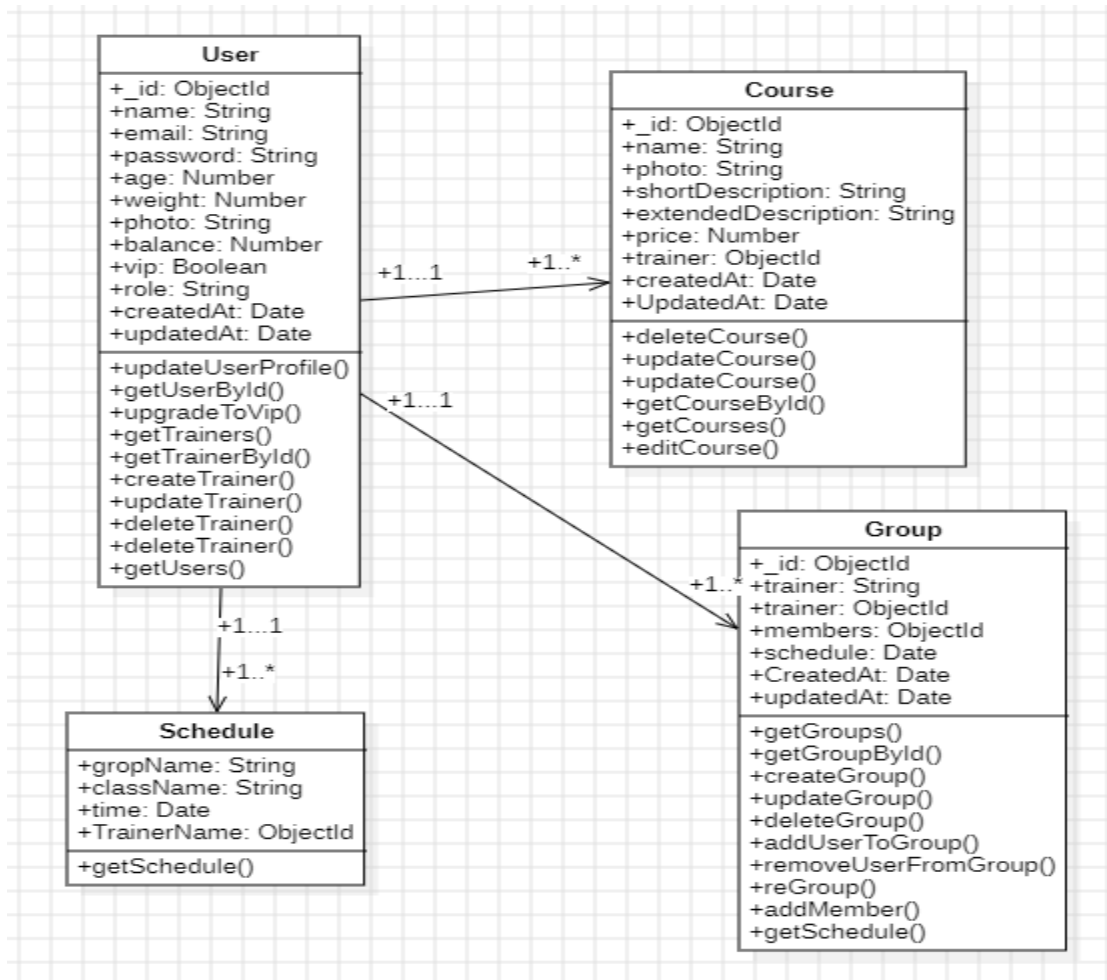


Рис. 2.2. Діаграма класів

Опишемо діаграму класів(рис.2.2).

User – клас, що визначає користувача

- updateUserProfile() – відповідає за оновлення профілю
- upgradeToVip() – Відповідає за зміну статусу абонементу
- getUserById() – відповідає за конкретного користувача
- createTrainer() – відповідає за створення тренера
- updateTrainer() – відповідає за зміну інформації про тренера
- deleteTrainer() – відповідає за видалення тренера
- getTrainerById() – відповідає за конкретного тренера
- getTrainers() – відповідає за всіх тренерів
- getUsers() – відповідає за всіх користувачів

Schedule – клас, який відповідає за створення розкладу

- `getSchedule()` – відповідає за отримання розкладу

Course – клас який відповідає за курси

- `deleteCourse()` – відповідає за видалення курсу
- `updateCourse()` – відповідає за редагування курсу
- `createCourse()` – відповідає за створення курсу
- `getCourseById()` – відповідає за отримання конкретного курсу
- `getCourses()` – відповідає за отримання всіх курсів
- `editCourse()` – відповідає за редагування курсу

Group – клас який відповідає за групи

- `getGroups()` – відповідає за отримання всіх курсів
- `getGroupById()` – відповідає за отримання конкретного курсу
- `createGroup()` – відповідає за створення групи
- `updateGroup()` – відповідає за редагування групи
- `deleteGroup()` – відповідає за видалення групи
- `addUserToGroup()` – відповідає за додавання користувача до групи
- `removeUserFromGroup()` – відповідає за видалення користувача

## 2.2 Розробка бази даних системи

Для зберігання даних нами було вирішено обрати базу даних MongoDB. MongoDB – документо-орієнтована система керування базами даних яка не потребує опису схеми таблиць. MongoDB підтримує зберігання документів в JSON-подібному форматі, має досить гнучку мову для формування запитів, може створювати індекси для різних збережених атрибутів.

База даних складається з чотирьох таблиць:

- User – таблиця, яка зберігає дані користувачів;
- Group – таблиця, яка зберігає дані про групи;
- Course – таблиця, яка зберігає дані про курси;
- Schedule – таблиця, яка зберігає дані про розклад;

Опис кожної з таблиць бази даних наведений у таблицях 2.9 – 2.12.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 25   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

Таблиця 2.9

Опис таблиці User

| Поле      | Тип      | Призначення                             |
|-----------|----------|-----------------------------------------|
| _id       | ObjectId | Унікальний ідентифікатор                |
| name      | String   | Ім'я користувача                        |
| email     | String   | Електронна пошта                        |
| password  | String   | Хешований пароль                        |
| age       | Number   | Вік користувача                         |
| weight    | Number   | Вага користувача                        |
| photo     | String   | URL фото                                |
| balance   | Number   | Баланс користувача                      |
| vip       | Boolean  | Статус VIP                              |
| role      | String   | Роль користувача (user, trainer, admin) |
| createdAt | Date     | Дата створення                          |
| updatedAt | Date     | Дата оновлення                          |

Таблиця 2.10

Опис таблиці Course

| Поле                | Тип      | Призначення                 |
|---------------------|----------|-----------------------------|
| _id                 | ObjectId | Унікальний ідентифікатор    |
| name                | String   | Назва курсу                 |
| photo               | String   | URL фото                    |
| shortDescription    | String   | Короткий опис               |
| extendedDescription | String   | Розширений опис             |
| price               | Number   | Ціна курсу                  |
| trainer             | ObjectId | Посилання на тренера (User) |
| createdAt           | Date     | Дата створення              |
| updatedAt           | Date     | Дата оновлення              |

Таблиця 2.11

Опис таблиці Group

| Поле      | Тип      | Призначення                 |
|-----------|----------|-----------------------------|
| _id       | ObjectId | Унікальний ідентифікатор    |
| name      | String   | Назва групи                 |
| trainer   | ObjectId | Посилання на тренера (User) |
| members   | ObjectId | Масив користувачів (User)   |
| schedule  | Date     | Дата та час тренування      |
| createdAt | Date     | Дата створення              |
| updatedAt | Date     | Дата оновлення              |

Таблиця 2.12

Опис таблиці Schedule

| Поле        | Тип      | Призначення                 |
|-------------|----------|-----------------------------|
| _id         | ObjectId | Унікальний ідентифікатор    |
| groupName   | String   | Назва групи                 |
| className   | String   | Назва курсу                 |
| time        | String   | Час                         |
| day         | String   | День                        |
| trainerName | ObjectId | Посилання на тренера (User) |

### 2.3 Проектування та реалізація алгоритмів роботи системи

Сайт є простим для користувача, має весь основний функціонал сайту, що робить навігацію використання функціоналу простим та легким для юзера. Алгоритм роботи для різних ролей детально представлений нижче(рис.2.3).

Якщо авторизований адміністратор:

- після аторизації адмін має доступ до адміністративної панелі;
- адміністратор може міняти ролі користувачів;
- адміністратор має можливість створювати нові групи, які потім можна редагувати або видалити;
- адмін може редагувати баланс будь якого користувача;

- адміністратор може створювати, редагувати та видаляти курси;

Якщо авторизований тренер:

- тренер може створювати, редагувати видаляти курси які були створені ним;
- тренер може створювати групи, додавати користувачів, міняти деталі груп, змінювати дату та час проведення;
- тренер може переглядати розклад;

Якщо зареєстрований користувач:

- користувач має можливість змінювати свій профіль;
- користувач має можливість купувати курси;
- користувач може оформити абонемент та отримати деякі привілеїї;
- користувач може контактувати з тренером;

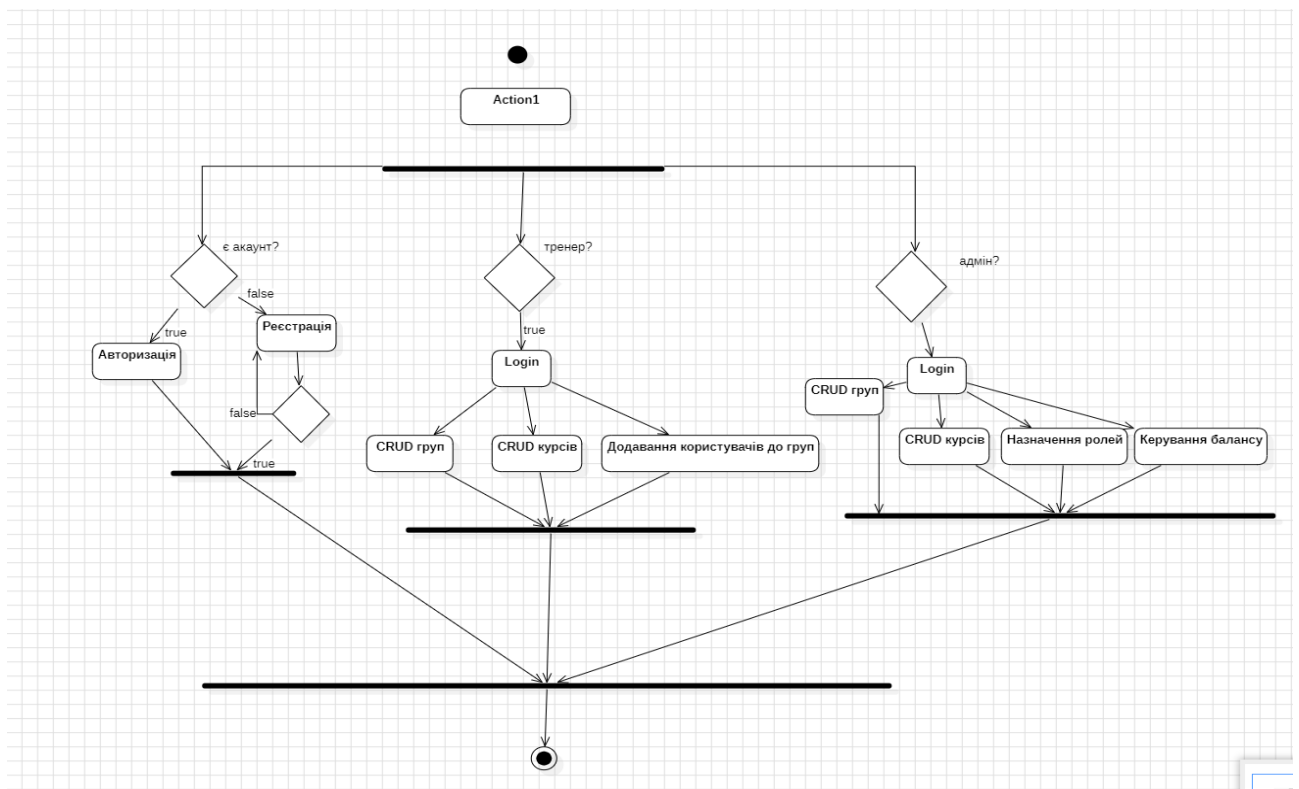


Рис.2.3. Діаграма активності

## 2.4 Реалізація функціоналу сайту спортивного залу

Розпишемо основні етапи взаємодії з сайтом. Першим етапом взаємодії є реєстрація користувача який прийшов на сайт(рис.2.4).

```
const registerUser = asyncHandler(async (req, res) => {
```

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 28   |

```

const { name, email, password } = req.body;
const userExists = await User.findOne({ email });
if (userExists) {
    res.status(400);
    throw new Error('User already exists');}
const user = await User.create({
    name,
    email,
    password,
});if (user) {
    res.status(201).json({
        _id: user._id,
        name: user.name,
        email: user.email,
        role: user.role,
        token: generateToken(user._id),
        vipStatus: user.vipStatus
    });
} else {
    res.status(400);
    throw new Error('Invalid user data');}}});

```

Рис.2.5. Функція реєстрації

Після реєстрації користувач має можливість придбати абонемент (рис. 2.5), який відкриває додаткові переваги. Після успішної покупки абонементу користувач автоматично отримує статус VIP. Для VIP користувачів усі курси стають безкоштовними, що надає їм необмежений доступ до навчальних матеріалів та тренувальних програм. Це стимулює користувачів активно користуватися сайтом та покращувати свої навички без додаткових витрат. Така система заохочення не тільки підвищує задоволеність користувачів, але й сприяє їхньому довготривалому залученню та лояльності до сайту.

```

const upgradeToVip = asyncHandler(async (req, res) => {
    const { userId, amount } = req.body;
    if (user) {
        if (user.balance >= amount) {
            user.balance -= amount;
            user.vipStatus = true;
            const updatedUser = await user.save();

```

|      |      |          |        |      |
|------|------|----------|--------|------|
|      |      |          |        |      |
| Змн. | Арк. | № докум. | Підпис | Дата |

```

        res.json(updatedUser);
    } else {throw new Error('Недостатньо коштів на балансі');
    }} else {
    throw new Error('Користувача не знайдено');}}});

```

Рис.2.5. Функція придбання VIP

Розглянемо можливості адміністратора на прикладі функцій, реалізованих на бекенді. Адміністратор може створювати курси які прирівнюються до тренерських курсів. Можна визначити відповідального тренера, який зможе редагувати та відповідати за цей курс(рис.2.6).

```

const createCourse = asyncHandler(async (req, res) => {
const { name, photo, price, shortDescription, extendedDescription } = req.body;
const course = new Course({
    name,
    photo,
    price,
    shortDescription,
    extendedDescription,
    trainer: req.user._id,
});
const createdCourse = await course.save();
res.status(201).json(createdCourse);
});

```

Рис.2.6. Створення курсу

Після створення курсу на сторінці адміністратора, курс можна тільки видалити(рис.2.7).

```

const deleteCourse = asyncHandler(async (req, res) => {
const course = await Course.findById(req.params.id);

if (course) {
    await course.remove();
    res.json({ message: 'Course removed' });
} else {
    res.status(404);
    throw new Error('Course not found');
}
});

```

### Рис.2.7. Видалення курсу

Адміністратор може міняти ролі для користувачів на тренера, адміна чи користувача(рис.2.8). В залежності від ролі, авторизованому користувачу буде видно різні варіанти сайту. Функція потрібна для коректного працювання сайту, назанчення нових тренерів, адмінів, також їх звільнення. Роль користувача міняється шляхом переписування даних в БД.

```
const updateUserRole = asyncHandler(async (req, res) => {
  const user = await User.findById(req.params.id);
  if (user) {
    user.role = req.body.role || user.role;
    const updatedUser = await user.save();
    res.json(updatedUser);
  } else {
    res.status(404);
    throw new Error('User not found');
  }
});
```

### Рис.2.8. Зміна ролі

Адміністратор має можливість створити пусту групу(Рис.2.9). В адмінпанелі користувач може ввести назву групи, опис і відповідального тренера. Після натискання кнопки відправлення, функція приймає ці дані і записує їх в базу даних, інші поля записуються автоматично. Ці поля потім міняє відповідальний тренер(рис2.10). Тренер може додати користувачів до груп за які він відповідає. Функція редагує поле members, записуючи масив користувачів в базу даних.(рис.2.11).

```
const createGroup = asyncHandler(async (req, res) => {
  const { name, description, trainer } = req.body;
  const group = new Group({ name, description, trainer });
  const createdGroup = await group.save();
  res.status(201).json(createdGroup);
});
```

### Рис.2.9. Створення групи

```
const reGroup = asyncHandler(async (req, res) => {
  const { name, date, time } = req.body;
  const group = await Group.findById(req.params.id);
  if (group) {
    group.name = name;
```

|      |      |          |        |      |
|------|------|----------|--------|------|
|      |      |          |        |      |
| Змн. | Арк. | № докум. | Підпис | Дата |

```

group.date = date;
group.time = time;
const updatedGroup = await group.save();
res.json(updatedGroup);
} else {
res.status(404);
throw new Error('Group not found');
}});

```

Рис.2.10. Редагування групи

```

const addMember = asyncHandler(async (req, res) => {
const { email } = req.body;
const user = await User.findOne({ email });
const group = await Group.findById(req.params.id);
if (user && group) {
if (group.members.includes(user._id)) {
res.status(400);
throw new Error('User already in group');
}
group.members.push(user._id);
await group.save();
res.json(group);
} else {
res.status(404);
throw new Error('User or Group not found');
}});

```

Рис.2.11. Додавання користувачів до групи

Адміністратор має можливість редагувати баланс користувача, що є важливою функцією для управління фінансовими аспектами сайту. Ця функція дозволяє адміністратору отримати користувача за його ідентифікатором, виділити поле балансу та перезаписати його відповідно до необхідних змін. Це може бути корисно для коригування помилок, нарахування бонусів, обробки повернень або внесення інших фінансових коригувань.

Можливість редагування балансу також сприяє демонстрації можливостей покупок на сайті, що покращує користувацький досвід і сприяє залученню нових клієнтів. Крім того, ця функція дозволяє створювати точну

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 32   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

статистику фінансових операцій, що є важливим для аналізу доходів та планування майбутнього розвитку сайту. (рис.2.12).

```
const plusBalance = asyncHandler(async (req, res) => {
  const { userId, amount } = req.body;
  const user = await User.findById(userId);
  user.balance += amount;
  await user.save();
  res.status(200).json({ message: 'Balance added successfully', balance:
    user.balance });
});
```

Рис.2.12. Редагування балансу

Таким чином було описано основний функціонал для адміністрування. Це дозволяє власнику якісно адмініструвати сайт, керувати тренерами і користувачами. Так само ми опишемо функціонал тренера.

Функції тренера відповідають за якісне наповнення сайту і керування користувачами.

*Функція створення курсу* (рис. 2.13) приймає введені дані від користувача. Вона створює новий об'єкт курсу, присвоює йому унікальний ідентифікатор та записує введені дані в цей об'єкт. Після цього об'єкт курсу зберігається у відповідній базі даних для подальшого використання.

*Функція редагування курсу* (рис.2.14). Функція відповідає за зміну деталей курсу, його наповнення та ціни. Редагувати курс може тільки той тренер, який створив курс, або якщо його назначив адміністратор. Функція приймає об'єкт курсу за ідентифікатором, поля якого може редагувати тренер.

*Функція отримання розкладу* (рис. 2.15) повертає масив об'єктів груп. Клієнтська частина отримує цей масив, сортує об'єкти за датою та відображає користувачу у зручному форматі. Це дозволяє користувачам легко ознайомитися з майбутніми тренуваннями та планувати свій графік.

*Функція створення групи* (рис.2.16). Функція приймає введені дані, створює новий об'єкт і присвоює йому унікальний ідентифікатор. Після цього вона записує введені дані в об'єкт, зберігаючи його у відповідній базі даних.

*Функція редагування групи* (рис.2.17). Функція відповідає за зміну деталей групи, включаючи користувачів, відповідального тренера, дату та час проведення тренування. Редагувати групу може тільки той тренер, який створив групу, забезпечуючи контроль над своїми тренувальними програмами. Функція приймає об'єкт групи за ідентифікатором, дозволяючи тренеру редагувати відповідні поля для актуалізації інформації. Це забезпечує гнучкість та точність в управлінні груповими тренуваннями, покращуючи організацію та планування занять.

```
const createCourse = asyncHandler(async (req, res) => {
  const { name, photo, price, shortDescription, extendedDescription } = req.body;
  const course = new Course({
    name,
    photo,
    price,
    shortDescription,
    extendedDescription,
    trainer: req.user._id,
  });
  const createdCourse = await course.save();
  res.status(201).json(createdCourse);
});
```

Рис.2.13. Створення курсу

```
const updateCourse = asyncHandler(async (req, res) => {
  const { name, description } = req.body;

  const course = await Course.findById(req.params.id);

  if (course) {
    course.name = name || course.name;
    course.description = description || course.description;

    const updatedCourse = await course.save();
    res.json(updatedCourse);
  } else {
```

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 34   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

```

        res.status(404);
        throw new Error('Course not found');
    }
});

```

Рис.2.14. Редагування курсу

```

const getSchedule = asyncHandler(async (req, res) => {
    const schedule = await Schedule.find({});
    res.json(schedule);
});

```

Рис.2.15. Отримання розкладу

```

const createGroup = asyncHandler(async (req, res) => {
    const { name, date, time } = req.body;

    const group = new Group({
        name,
        date,
        time,
        trainer: req.user._id,
    });

    const createdGroup = await group.save();
    res.status(201).json(createdGroup);
});

```

Рис.2.16. Створення групи

```

const reGroup = asyncHandler(async (req, res) => {
    const { name, date, time } = req.body;
    const group = await Group.findById(req.params.id);
    if (group) {
        group.name = name;
        group.date = date;
        group.time = time;
        const updatedGroup = await group.save();
        res.json(updatedGroup);
    } else {
        res.status(404);
        throw new Error('Group not found');
    }
});

```

Рис.2.17. Редагування групи

В кінцевому результаті, маємо низку функцій які створюють і взаємодіють з багатьма об'єктами. Функціонал забезпечує коректну роботу сайту і простоту використання сайту користувачами. Виходячи з цього, було створено діаграму компонентів спортивного залу де визначені основні компоненти додатку(рис. 2.18).

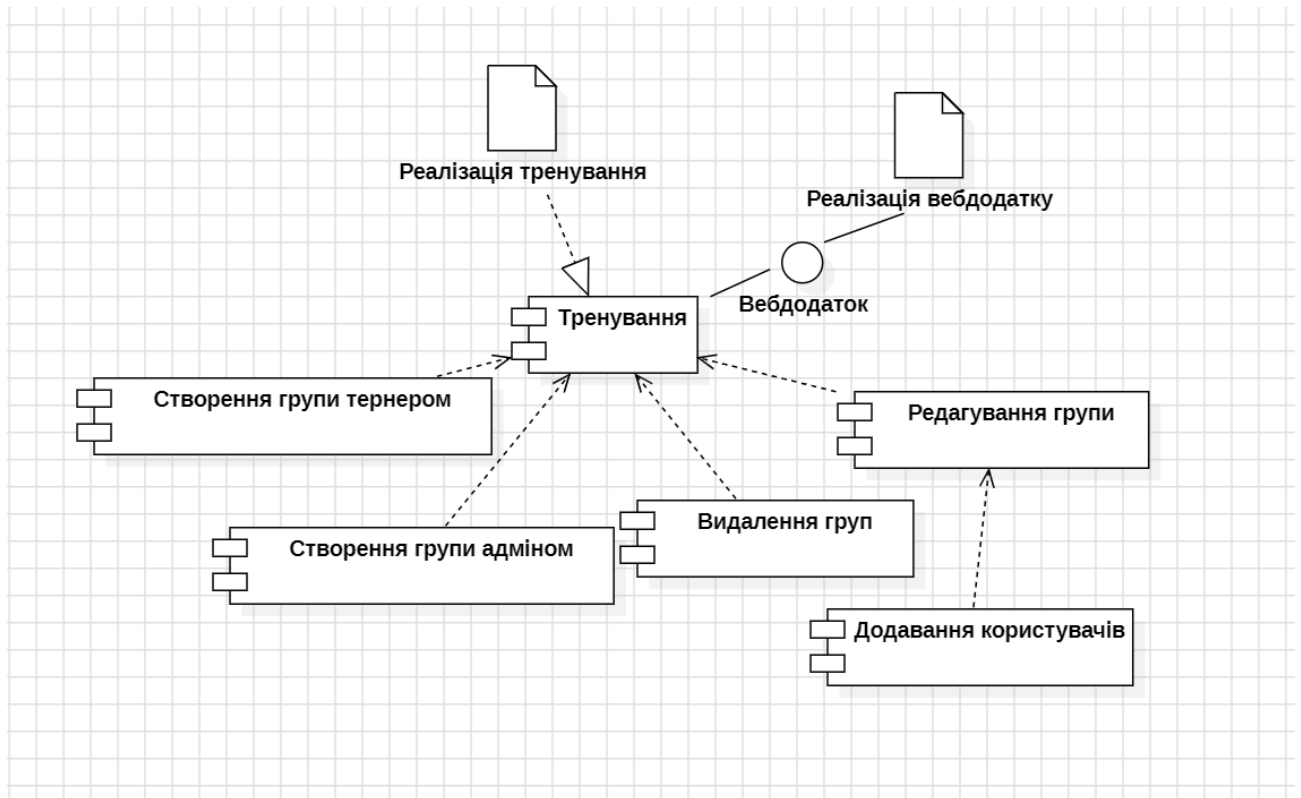


Рис.2.18. Діаграма компонентів

На діаграмі бачимо такі компоненти:

- тренування – компонент який відповідає за тренування
- створення групи тренером – компонент за допомогою якого тренер може створювати групу
- створення групи адміном – компонент за допомогою якого адміністратор може створювати групу
- редагування групи – компонент за допомогою якого тренер може редагувати групу
- додавання користувачів – компонент за допомогою якого тренер може додавати користувачів до групи

- видалення груп – компонент за допомогою якого тренер або адмін видаляє групу.

### **Висновки до другого розділу:**

В другому розділі був проведений детальний аналіз більшості сценаріїв використання програмного продукту відповідно до вимог програмного забезпечення. Виходячи з цієї інформації була спроектована структура сайту. Проектування включало в себе моделювання структури проекту та взаємодії між їх складовими.

Також було розроблено структуру бази даних для зберігання значень та взаємодії між собою та серверною частиною, таких як користувачі, курси, групи, розклад.

Був наведений опис одного з основних компонентів додатку – створення груп, їх редагування та видалення.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 37   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

## РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З САЙТОМ ТРЕНАЖЕРНОГО ЗАЛУ

### 3.1 Проектування інтерфейсу обробки даних

Налаштування додатку вимагає розгортання всіх його компонентів та налаштування як кожного елементу окремо, так і загальної системи в цілому.

Для опису архітектури сайту було створено діаграму розгортання(рис.3.1).

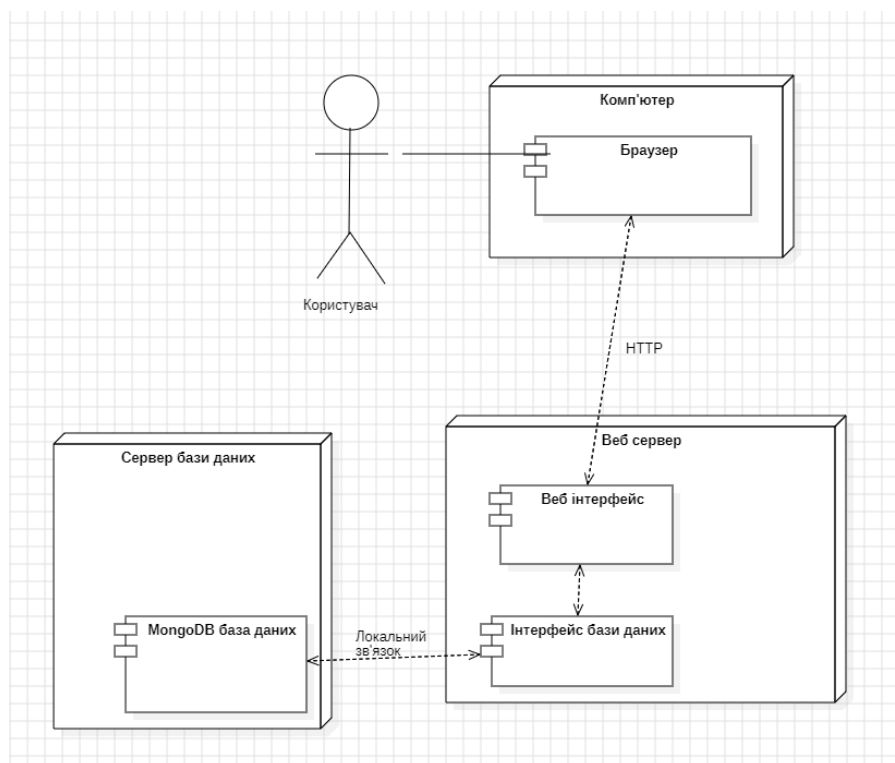


Рис. 3.1. Діаграма розгортання

Опис процесу розгортання.

- Сервер бази даних – розгортання серверної частини проекту з базою даних;
  - MongoDB – база даних яка буде розгортатись;
- Веб сервер – розгортання клієнтської частини додатку;
  - Веб інтерфейс – інтерфейс який буде бачити користувач при використанні сайту;
  - Інтерфейс бази даних – інтерфейс адміна для детальної інформації про стан бази даних;

- Комп'ютер – пристрій за допомогою якого користувач має доступ до сайту;

### 3.2 Структура інтерфейсу сайту тренажерного залу

Взаємодія користувачів відбувається через сайт. На головній сторінці користувач може побачити вітальне повідомлення, яке запрошує до Тренажерного залу та інформує про можливість знайти найкращих тренерів, групові заняття, курси та багато іншого. Є декілька кнопок для навігації: "Розклад", "Тренери", "Групи" та "Курси". Також розміщено мотиваційний текст: "Ніколи не зупиняйся на півдорозі до своєї мети, навіть коли здається, що сили закінчуються. Пам'ятай, що найбільший прогрес приходить через подолання найважчих перешкод." Користувач може легко орієнтуватися по сайту завдяки чіткій структурі та зрозумілим навігаційним елементам(рис.3.2).

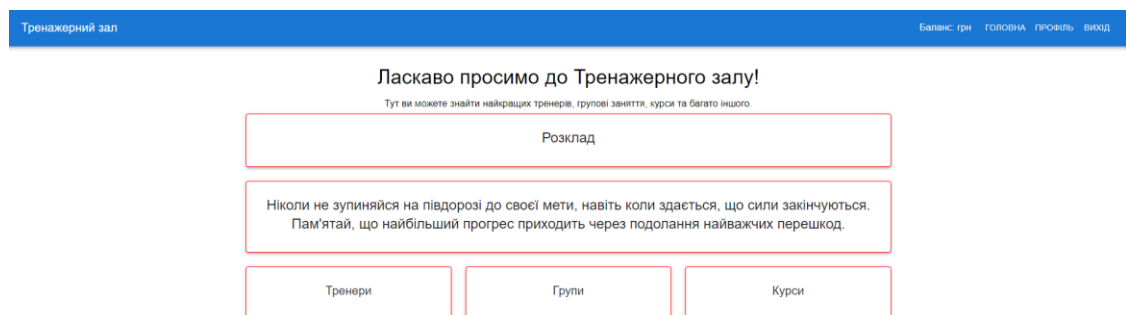


Рис. 3.2. Головна сторінка сайту

Кнопка "Тренери" на головній сторінці сайту надає користувачам доступ до списку кваліфікованих тренерів, які працюють у тренажерному залі. Натиснувши на цю кнопку, користувач може ознайомитися з детальною інформацією про кожного тренера, включаючи їх досвід, спеціалізацію, досягнення та розклад тренувань. Це допомагає обрати тренера, який найкраще

відповідає їхнім потребам та цілям, забезпечуючи персоналізований підхід до тренувань(рис.3.3).

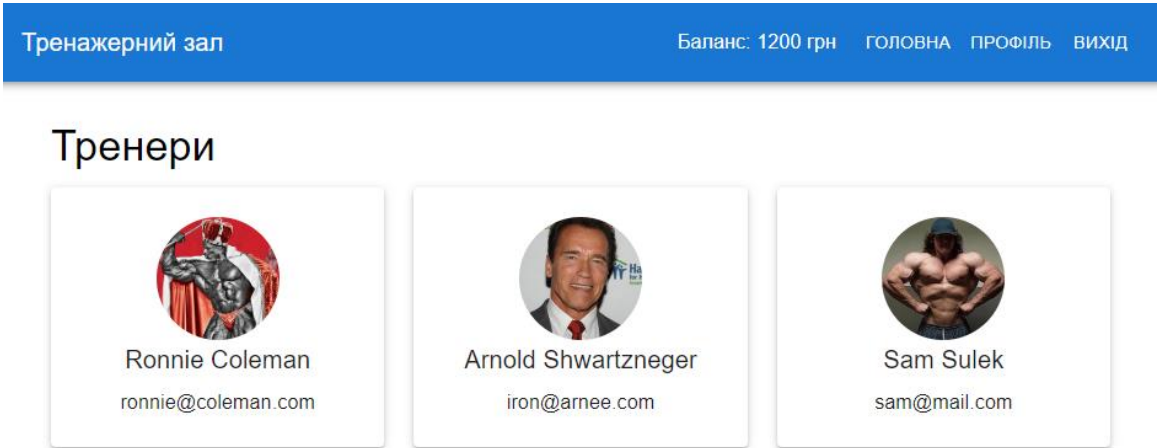


Рис. 3.2. Тренери

Кнопка "Групи" на головній сторінці сайту дозволяє користувачам ознайомитися з різноманітними груповими заняттями, які пропонує тренажерний зал. Натиснувши на цю кнопку, користувач може переглянути розклад групових занять(рис.3.4), дізнатися про різні типи тренувань, такі як йога, аеробіка, силові тренування та інші, а також ознайомитися з описами та рівнями складності кожного заняття. Це допомагає користувачам обрати групу, яка відповідає їхнім інтересам та рівню фізичної підготовки, сприяючи ефективному та цікавому тренуванню в компанії однодумців(рис.3.3).

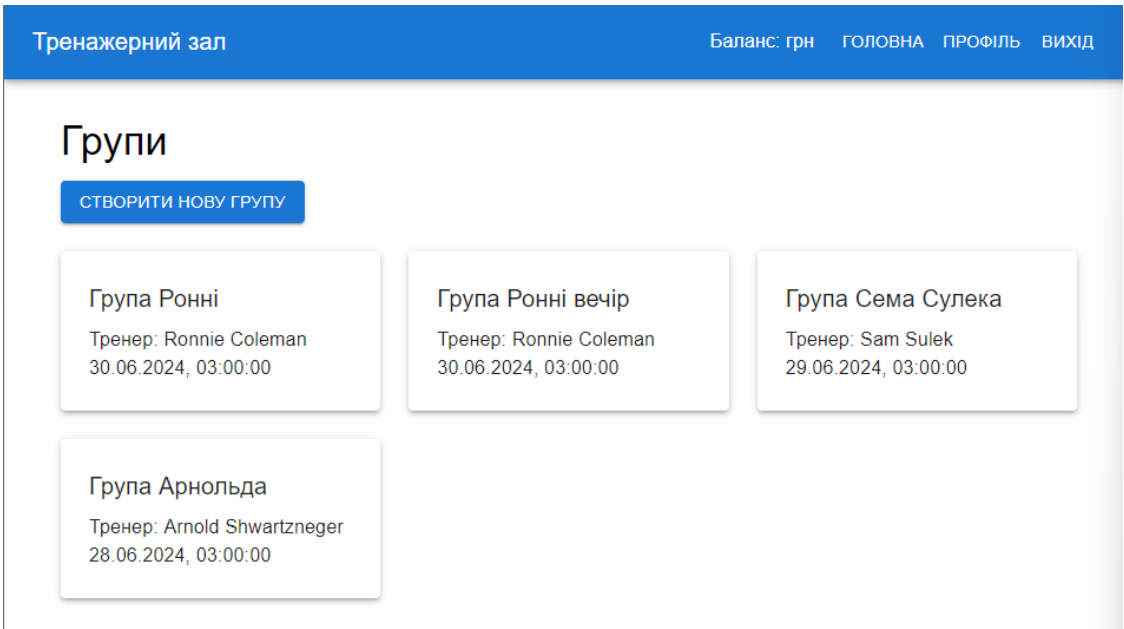


Рис.3.3. Сторінка груп

Розклад

Група Арнольда

28.06.2024 о 13:25

Тренер: Arnold Shwartzneger

ПЕРЕГЛЯНУТИ ГРУПУ

Група Сема Сулека

29.06.2024 о 14:25

Тренер: Sam Sulek

ПЕРЕГЛЯНУТИ ГРУПУ

Група Ронні

30.06.2024 о 16:00

Тренер: Ronnie Coleman

ПЕРЕГЛЯНУТИ ГРУПУ

Група Ронні вечір

30.06.2024 о 17:10

Тренер: Ronnie Coleman

ПЕРЕГЛЯНУТИ ГРУПУ

< 1 >

Рис.3.4. Сторінка розкладу

Кнопка "Курси" на головній сторінці сайту відкриває доступ до повного списку навчальних курсів, які пропонує тренажерний зал. Натиснувши на цю кнопку, користувачі можуть переглядати детальні описи курсів, які охоплюють різні аспекти здорового способу життя, включаючи правильне харчування, техніку виконання вправ, плани тренувань та програми для різних рівнів підготовки. Кожен курс дозволяє користувачам вибрати оптимальний курс для досягнення своїх фітнес-цілей та покращення загального самопочуття(рис.3.5).

Курси

СТВОРИТИ КУРС

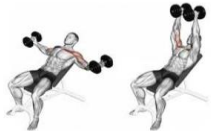


FullBody тренування

Fullbody тренування — це комплекс вправ, спрямованих на опрацювання всіх основних груп м'язів за одне заняття.

200 грн

ПРИДБАТИ



Спліт(Понеділок)

Спліт-тренування в понеділок зазвичай фокусується на окремих групах м'язів.

150 грн

ПРИДБАТИ



Спліт(Середа)

Спліт-тренування у середу може бути спрямоване на опрацювання м'язів ніг.

150 грн

ПРИДБАТИ



Спліт(П'ятниця)

Спліт-тренування у п'ятницю може бути присвячене м'язам спини та щіцепів.

150 грн

ПРИДБАТИ

Рис.3.5. сторінка курсів



### FullBody тренування

FullBody тренування - це комплексне тренування, яке охоплює всі основні групи м'язів тіла. Це тренування дозволяє досягти максимального ефекту від тренування, оскільки всі м'язи працюють одночасно. Тренування проводиться в тренажерному залі, де є всі необхідні обладнання для тренування. Тренування проводиться в групі, що дозволяє отримувати підтримку та мотивацію від тренера та інших учасників. Тренування проводиться впродовж 60 хвилин, що дозволяє досягти максимального ефекту від тренування. Тренування проводиться впродовж 60 хвилин, що дозволяє досягти максимального ефекту від тренування. Тренування проводиться впродовж 60 хвилин, що дозволяє досягти максимального ефекту від тренування.

Ціна: 200 грн

Автор:

ПРИДБАТИ КУРС

РЕДАГУВАТИ КУРС

Рис.3.6. Приклад курсу

Тренер у нашому тренажерному залі має можливість створювати групи для різних видів тренувань та додавати туди учасників. Це дозволяє тренерам організовувати ефективні групові заняття, спрямовані на різні рівні підготовки та специфічні цілі учасників. Кожна група може мати власний розклад, програму тренувань та індивідуальний підхід до кожного учасника. Це сприяє кращій взаємодії між тренером і учасниками, забезпечуючи високий рівень мотивації та підтримки в процесі тренувань(рис.3.7).

### Створити нову групу

|                |                      |
|----------------|----------------------|
| Назва *        | Група Ронні Колемана |
| Дата *         | 29.06.2024           |
| Час *          | 15:00                |
| СТВОРИТИ ГРУПУ |                      |

Рис.3.7. Створення групи

Тренер має можливість створювати курси, що дозволяє пропонувати учасникам структуровані та цілеспрямовані програми тренувань. Кожен курс розробляється з урахуванням конкретних цілей, таких як підвищення фізичної витривалості, зниження ваги, нарощування м'язової маси або поліпшення

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 42   |

техніки виконання вправ. Тренери можуть включати в курс детальні інструкції, плани харчування та рекомендації щодо відновлення. Це забезпечує комплексний підхід до тренувань, дозволяючи учасникам отримувати максимум користі від занять і досягати своїх фітнес-цілей під професійним керівництвом(рис.3.8).

#### Створення курсу

Файл не выбран

Рис.3.8. Створення курсу

#### Редагувати курс

Файл не выбран



Рис.3.9. Редагування курсу

Редагування профілю дозволяє користувачам оновлювати та змінювати особисту інформацію, забезпечуючи актуальність і точність даних. Користувачі можуть змінювати своє ім'я, електронну пошту, пароль, завантажувати нові фотографії профілю та додавати контактну інформацію. Крім того, в профілі

можна вказувати інтереси, фітнес-цілі та іншу інформацію, що допоможе тренерам краще розуміти потреби і вподобання кожного користувача. Така функціональність сприяє персоналізованому підходу до тренувань і покращує взаємодію в рамках тренажерного залу або онлайн-сайту(рис.3.10).

Рис.3.10. Редагування профілю

Адміністратор має можливість змінювати ролі користувачів, що є важливим аспектом управління тренажерним залом або сайтом. Завдяки цій функції адміністратор може призначати нових тренерів, модераторів або інших адміністраторів, забезпечуючи гнучке керування ресурсами та розподіл завдань. Зміна ролей дозволяє адаптувати структуру сайту до поточних потреб, розширювати функціонал для певних користувачів та забезпечувати належний контроль за всіма аспектами діяльності. Це сприяє ефективному управлінню, підтримці високого рівня обслуговування та безпеки на сайті.

Крім того, можливість змінювати ролі користувачів допомагає в підтримці дисципліни та порядку. Адміністратор може понижувати роль користувача у разі порушення правил або неадекватної поведінки, що забезпечує дотримання встановлених норм і стандартів. Це також дозволяє оперативно реагувати на зміни в команді та швидко інтегрувати нових співробітників або тренерів. Такий підхід забезпечує динамічне та адаптивне управління сайтом, що є ключовим для її успішного функціонування та розвитку(рис. 3.11).

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 44   |

### Адмін панель

Зміна ролей користувачів

|                                          |                                         |
|------------------------------------------|-----------------------------------------|
| <div>dim1231</div> <div>Користувач</div> | <div>фівфів</div> <div>Користувач</div> |
| <div>test2</div> <div>Користувач</div>   | <div>Ronnie</div> <div>Адмін</div>      |
| <div>123</div> <div>Користувач</div>     | <div>123</div> <div>Користувач</div>    |

Рис.3.11. Зміна ролей користувача

Адміністратор має можливість створювати та видаляти групи, що є важливим інструментом для організації та управління тренувальними програмами на сайті. Завдяки цій функції адміністратор може легко створювати нові групи для різних типів тренувань, встановлювати розклад занять, додавати тренерів та учасників. Крім того, адміністратор може налаштовувати параметри груп, такі як максимальна кількість учасників, рівень складності та специфічні цілі тренувань, що сприяє ефективній організації тренувального процесу.

Видалення груп також є важливою функцією для підтримки порядку та актуальності на сайті. Адміністратор може видаляти групи, які вже не є актуальними або більше не відповідають потребам користувачів, що допомагає уникнути перевантаження та плутанини. Це також дозволяє вчасно оновлювати тренувальні програми та забезпечувати користувачів лише актуальною та корисною інформацією. Завдяки можливості створювати та видаляти групи, адміністратор може гнучко керувати тренувальним процесом, забезпечуючи ефективну організацію занять та високий рівень обслуговування(рис. 3.12).

Управління групами

СТВОРИТИ ГРУПУ

|                                                     |                                                           |                                                        |
|-----------------------------------------------------|-----------------------------------------------------------|--------------------------------------------------------|
| <div>Група Ронні</div> <div>ВИДАЛИТИ ГРУПУ</div>    | <div>Група Ронні вечір</div> <div>ВИДАЛИТИ ГРУПУ</div>    | <div>Група Сема Сулека</div> <div>ВИДАЛИТИ ГРУПУ</div> |
| <div>Група Арнольда</div> <div>ВИДАЛИТИ ГРУПУ</div> | <div>Група Ронні Колемана</div> <div>ВИДАЛИТИ ГРУПУ</div> |                                                        |

Рис.3.12. Зміна ролей користувача

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 45   |

Адміністратор має можливість створювати та видаляти курси, що є ключовим інструментом для управління навчальними програмами на сайті. Завдяки цій функції адміністратор може розробляти нові курси, додаючи до них докладні описи, відеоуроки, плани занять та інші навчальні матеріали. Це дозволяє адаптувати навчальні програми до потреб користувачів, забезпечуючи їх різноманітними і корисними ресурсами. Крім того, адміністратор може призначати тренерів для ведення курсів, встановлювати розклад та контролювати процес навчання, що сприяє підвищенню якості освітніх послуг на сайті.

Видалення курсів є не менш важливою функцією, яка дозволяє підтримувати актуальність і якість навчальних програм. Адміністратор може видаляти курси, які стали застарілими або не відповідають поточним вимогам і стандартам. Це допомагає уникнути плутанини та забезпечити користувачам доступ лише до найактуальніших і найефективніших навчальних матеріалів. Завдяки можливості створювати та видаляти курси, адміністратор може гнучко керувати освітнім контентом, забезпечуючи високу якість навчання та задоволення потреб користувачів(рис. 3.13).

Управління курсами

СТВОРИТИ КУРС

FullBody тренування

Ціна: 200 грн

ВИДАЛИТИ КУРС

Спліт(Понеділок)

Ціна: 150 грн

ВИДАЛИТИ КУРС

Спліт(Середа)

Ціна: 150 грн

ВИДАЛИТИ КУРС

Спліт(П'ятниця)

Ціна: 150 грн

ВИДАЛИТИ КУРС

Рис.3.13. Зміна ролей користувача

Адміністратор має можливість відслідковувати статистику покупки курсів і абонементів, що є важливим інструментом для аналізу та планування розвитку сайту. Завдяки цій функції адміністратор може отримувати детальну інформацію про кількість проданих курсів та абонементів, визначати

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 46   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

популярність конкретних програм та тренувань, а також відстежувати фінансові показники. Це дозволяє приймати обґрунтовані рішення щодо покращення контенту, маркетингових стратегій та оптимізації цінової політики. Такий підхід сприяє підвищенню задоволеності користувачів і забезпечує стабільний ріст сайту(рис.3.14).

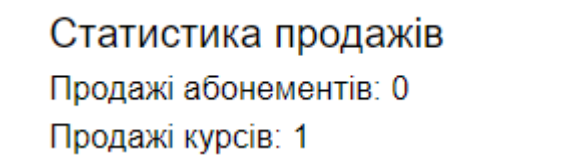


Рис.3.14. Статистика

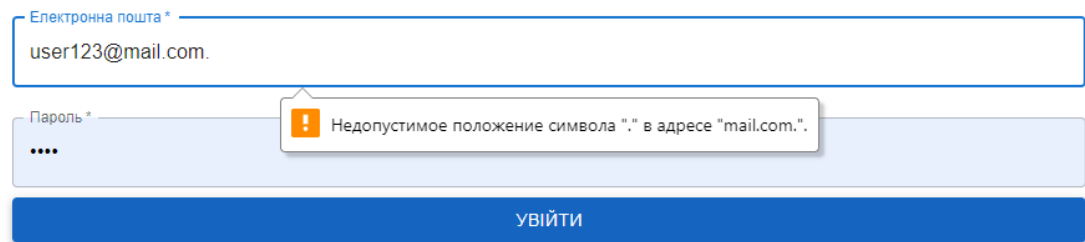
### 3.3 Тестування сайту

Тестування може відбуватися вручну, або автоматично. Ручне тестування передбачає безпосереднє виконання тестових сценаріїв та аналіз результатів користувачем, що дозволяє виявити проблеми в реальному часі та оцінити зручність використання інтерфейсу. Це ефективний спосіб для виявлення помилок, які можуть бути пропущені автоматизованими тестами, оскільки людина може звертати увагу на несподівані нюанси взаємодії з системою. Водночас автоматизоване тестування дозволяє швидко перевіряти велику кількість сценаріїв, зменшуючи ризик людської помилки та підвищуючи ефективність повторюваних тестів. Однак, в даному випадку буде використовуватися ручний аналіз для забезпечення максимальної точності та адаптивності до різних ситуацій.

Перш за все, перевіримо сторінку реєстрації. Важливо переконатися, що всі необхідні поля (ім'я користувача, електронна пошта, пароль) присутні і правильно позначені. Тестування включає введення коректних даних для успішної реєстрації та перевірку обробки помилок при введенні некоректних даних (наприклад, неправильний формат електронної пошти(рис.3.15-3.16), або занадто короткий пароль). Також слід перевірити функціональність кнопки "Зареєструватися" та переконатися, що після успішної реєстрації користувач

перенаправляється на правильну сторінку, а в разі помилок - отримує відповідні повідомлення.

## Авторизація



Електронна пошта \*

user123@mail.com.

Пароль \*

....

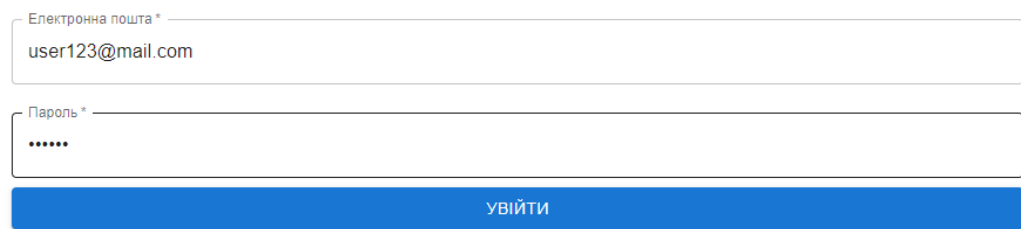
Недопустимое положение символа "." в адресе "mail.com."

УВІЙТИ

Рис.3.15. Тестування поля пошти

## Авторизація

Invalid email or password



Електронна пошта \*

user123@mail.com

Пароль \*

.....

УВІЙТИ

Рис.3.16. Тестування правильності запитів

Для тестування сторінки профілю слід перевірити кілька ключових аспектів. Спочатку слід переконатися, що всі поля форми профілю коректно відображають інформацію користувача. Це включає такі поля, як ім'я, вік, вага та можливість завантаження файлу(рис.3.17-3.19). Важливо перевірити, що дані, введені під час реєстрації, правильно відображаються на сторінці профілю. Також варто перевірити коректність міток та підписів до полів, а також переконатися, що обов'язкові поля позначені зірочкою (\*) і виділяються, якщо вони залишаються порожніми.

Далі слід протестувати функціональність редагування профілю. Введення нових даних у поля повинно правильно оновлювати інформацію про користувача після натискання кнопки "Зберегти зміни". Важливо перевірити, що всі обов'язкові поля повинні бути заповнені, інакше з'явиться відповідне повідомлення про помилку, як показано на рисунках. Необхідно також

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 48   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

перевірити, що повідомлення про помилку відображаються зрозумілою мовою для користувача і містять вказівки щодо виправлення помилок. Крім цього, слід перевірити, чи правильно працює завантаження файлів: чи файл обирається, завантажувється, і чи з'являється відповідне повідомлення або попередній перегляд завантаженого файлу.

The screenshot shows a web form for editing a user profile. At the top left is a blue button labeled "РЕДАГУВАТИ". Below it are four input fields: "Ім'я \*" containing "123", "Вік \*" which is empty, "Вага \*" which is empty, and a file upload field with a "Вибрати файл" button and the text "Файл не выбран". A red tooltip with an exclamation mark icon and the text "Заполните это поле." points to the empty "Вік \*" field. At the bottom is a wide blue button labeled "ЗБЕРЕГТИ ЗМІНИ".

Рис.3.17. Перевірка полів профілю

This screenshot shows the same profile form as Figure 3.17, but with some data entered. The "Ім'я \*" field contains "123" and the "Вік \*" field contains "23". The "Вага \*" field is still empty. The file upload field remains unchanged. The red tooltip "Заполните это поле." is still present over the empty "Вік \*" field. The "РЕДАГУВАТИ" and "ЗБЕРЕГТИ ЗМІНИ" buttons are also visible.

Рис.3.18. Перевірка полів профілю

This screenshot shows the profile form with all fields filled out. The "Ім'я \*" field contains "123", "Вік \*" contains "23", and "Вага \*" contains "23". The file upload field still shows "Вибрати файл" and "Файл не выбран". The red tooltip "Заполните это поле." is now gone. The "РЕДАГУВАТИ" button is at the top, and the "ЗБЕРЕГТИ ЗМІНИ" button is at the bottom.

Рис.3.19. Перевірка полів профілю

Для перевірки полів створення курсу необхідно забезпечити правильність введення даних у всі обов'язкові поля. Спершу переконаємося, що всі поля форми присутні та відображаються правильно. Основні поля, які слід перевірити, включають назву курсу, опис, категорію, вартість та інші необхідні параметри(рис.3.20-3.22). Кожне з цих полів повинно мати відповідну мітку та підпис, які допоможуть користувачеві зрозуміти, що саме потрібно вводити. Крім того, обов'язкові поля повинні бути чітко позначені, наприклад, за допомогою зірочки (\*), а форма має забезпечувати коректну валідацію цих полів.

Далі слід перевірити, як система обробляє помилки при введенні некоректних даних. Введення неправильного формату даних або залишення обов'язкових полів порожніми повинно призводити до появи зрозумілих повідомлень про помилки. Наприклад, якщо залишити поле "назва курсу" порожнім, система має видати повідомлення про те, що це поле обов'язкове для заповнення. Аналогічно, для поля "вартість" має бути перевірка на введення числових значень. Після виправлення помилок і повторного введення даних, система повинна дозволяти створення курсу, зберігати його у базі даних і відображати у відповідному розділі сайту.

#### Створення курсу

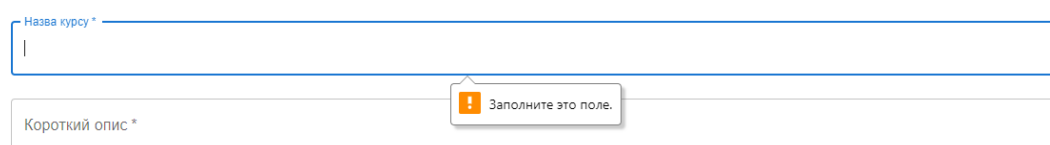


Рис.3.20. Перевірка полів профілю

#### Створення курсу

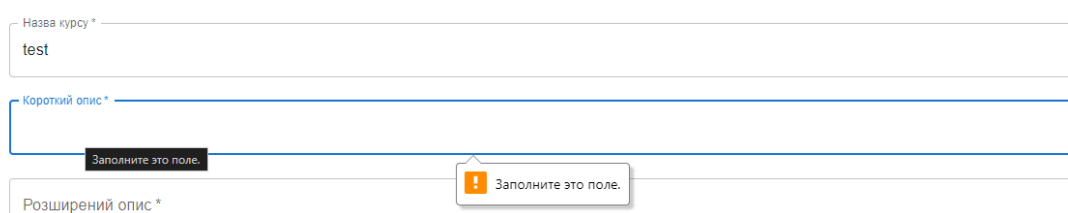


Рис.3.21. Перевірка полів профілю

Розширений опис \*

!

Заполните это поле.

Рис.3.22. Перевірка полів профілю

При тестуванні функціональності покупки курсу необхідно переконатися, що система правильно перевіряє баланс користувача перед завершенням транзакції(рис.3.23). Спершу перевіримо, чи відображається поточний баланс користувача на сторінці покупки курсу. Це допоможе користувачеві відразу зрозуміти, чи вистачає в нього коштів для покупки.

Після цього спробуємо здійснити покупку курсу. Якщо баланс користувача достатній, система повинна дозволити завершити транзакцію, зняти відповідну суму з балансу та оновити інформацію про курс у профілі користувача, відображаючи його як придбаний(рис.3.24-3.27). Важливо перевірити, що користувач отримує повідомлення про успішну покупку. Якщо ж баланс недостатній, система повинна відмовити в покупці і показати користувачеві зрозуміле повідомлення про те, що недостатньо коштів, з пропозицією поповнити баланс. Перевірка повинна включати сценарії з точною сумою коштів, трохи менше і набагато менше, щоб переконатися, що система правильно реагує в різних ситуаціях.

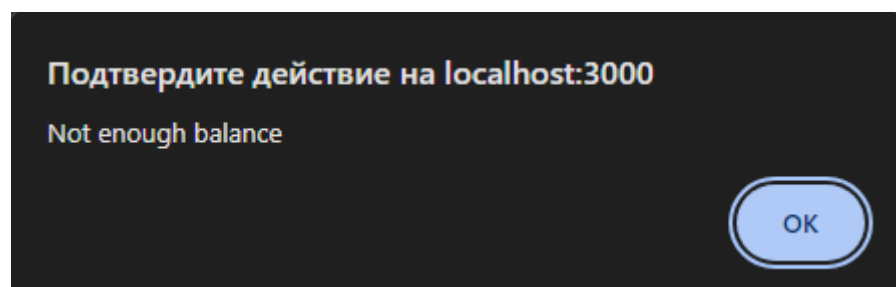


Рис.3.23. Помилка(недостатній баланс)



Спліт(П'ятниця)

Спліт-тренування у п'ятницю може бути присвячене м'язам спини та біцепсів, що дозволяє зосередитися на зміцненні та розвитку цих важливих груп м'язів. Тренування спини допомагає покращити поставу, збільшити силу і витривалість, а також сприяє загальному балансу тіла. Робота над біцепсами додає об'єму рукам і покращує їхню функціональність. Спліт-тренування дозволяє детально опрацювати кожну м'язову групу, забезпечуючи їй достатнє навантаження і час на відновлення. До основних вправ для спліт-тренування спини та біцепсів у п'ятницю можна віднести: 1. Підтягування: базова вправа для розвитку верхньої частини спини, біцепсів та плечей. 2. Тяга штанги в нахилі: ефективна для опрацювання м'язів середньої та нижньої частини спини. 3. Тяга гантелі до поясу: ізолювана вправа для кожної сторони спини окремо. 4. Підйом штанги на біцепс: класична вправа для розвитку біцепсів. 5. Концентрований підйом гантелі на біцепс: ізолювана вправа, яка дозволяє ретельно опрацювати біцепси. Ці вправи допоможуть комплексно навантажити м'язи спини та біцепсів, сприяти їхньому гармонійному розвитку та зміцненню. Регулярні тренування за таким планом допоможуть досягти значних результатів у побудові міцного та збалансованого тіла.

Ціна: 150 грн

Автор:

ПРИДБАТИ КУРС

РЕДАГУВАТИ КУРС

Рис.3.24. Непридбаний курс

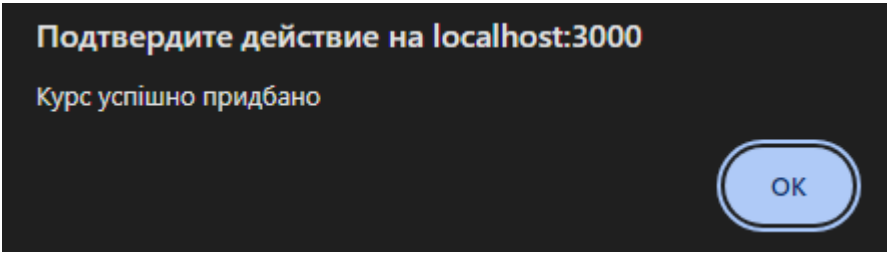


Рис.3.25. Придбаний курс



Спліт(П'ятниця)

Спліт-тренування у п'ятницю може бути присвячене м'язам спини та біцепсів, що дозволяє зосередитися на зміцненні та розвитку цих важливих груп м'язів. Тренування спини допомагає покращити поставу, збільшити силу і витривалість, а також сприяє загальному балансу тіла. Робота над біцепсами додає об'єму рукам і покращує їхню функціональність. Спліт-тренування дозволяє детально опрацювати кожну м'язову групу, забезпечуючи їй достатнє навантаження і час на відновлення. До основних вправ для спліт-тренування спини та біцепсів у п'ятницю можна віднести: 1. Підтягування: базова вправа для розвитку верхньої частини спини, біцепсів та плечей. 2. Тяга штанги в нахилі: ефективна для опрацювання м'язів середньої та нижньої частини спини. 3. Тяга гантелі до поясу: ізолювана вправа для кожної сторони спини окремо. 4. Підйом штанги на біцепс: класична вправа для розвитку біцепсів. 5. Концентрований підйом гантелі на біцепс: ізолювана вправа, яка дозволяє ретельно опрацювати біцепси. Ці вправи допоможуть комплексно навантажити м'язи спини та біцепсів, сприяти їхньому гармонійному розвитку та зміцненню. Регулярні тренування за таким планом допоможуть досягти значних результатів у побудові міцного та збалансованого тіла.

Ціна: 150 грн

Автор:

РЕДАГУВАТИ КУРС

Рис.3.26. Придбаний курс



### Спліт(П'ятниця)

Спліт-тренування у п'ятницю може бути присвячене м'язам спини та біцепсів.

150 грн

ВИ ВЖЕ КУПИЛИ ЦЕЙ КУРС

Рис.3.27. Придбаний курс

Для перевірки функціональності редагування групи та додавання туди користувачів необхідно спочатку переконатися, що існує можливість доступу до форми редагування групи. Перевіримо, чи правильно відображаються поточні дані групи, такі як назва, опис та список учасників. Важливо переконатися, що ці поля можна редагувати, і що система дозволяє зберігати зміни після редагування. Після внесення змін, система повинна оновити інформацію про групу, відобразити нові дані і повідомити користувача про успішне оновлення.

Далі перевіримо функціональність додавання користувачів до групи. Спершу переконаємося, що в інтерфейсі доступна опція додавання нових користувачів. При виборі цієї опції система повинна надати список доступних користувачів, які можна додати до групи. Після вибору користувачів і підтвердження додавання, система має оновити список учасників групи і відобразити нових членів. Важливо перевірити, чи отримує користувач відповідне повідомлення про успішне додавання нових учасників і чи оновлюється інформація в режимі реального часу. Тестування повинно включати також перевірку обробки помилок, наприклад, спроба додати неіснуючого користувача або користувача, який вже є членом групи(рис.3.28, 3.29).

Учасники

|                                |                           |                          |
|--------------------------------|---------------------------|--------------------------|
| dima1231<br>user5@mail.com7123 | фівфів<br>user5@mail.com7 | test2<br>user5@mail.com2 |
|--------------------------------|---------------------------|--------------------------|

РЕДАГУВАТИ ГРУПУ

Додати учасника за email \*

user5@mail.com7123

ДОДАТИ УЧАСНИКА

Рис.3.28. Спроба додати користувача який є в групі

Користувач вже є учасником групи

Рис.3.29. Помилка

### Висновки до третього розділу

В третьому розділі було наведено детальну схему розгортання сайту спортивного залу. Кожен структурний елемент системи був описаний, а також продемонстровано його взаємодію з іншими частинами сайту. Це дозволило зрозуміти архітектуру системи та взаємозв'язки між її компонентами. Далі було надано опис інтерфейсу розробленого програмного продукту. Були розглянуті всі функціональні можливості додатку, залежно від ролі користувача. Кожен тип користувача отримав свої унікальні функції та можливості в системі. Це допомагає кожному користувачу зорієнтуватись у функціоналі додатку та використовувати його відповідно до своїх потреб.

Окрім того, було продемонстровано, як працює статистика для адміністрації та звичайних користувачів. Була показана різниця між ними та продемонстровано, що конкретно відображають різні діаграми та статистичні дані. Адміністраторам надається детальна інформація про активність відвідувачів, використання обладнання, популярність тренувальних програм та розклад тренерів. Звичайні користувачі можуть бачити свій прогрес, кількість відвідувань, спожиті калорії та персональні досягнення.

Також було детально описано, як користувачі можуть реєструватися на тренування, бронювати обладнання та користуватися іншими послугами залу

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 54   |

через сайт. Кожен користувач має можливість переглядати доступний розклад занять, записуватись на групові тренування або бронювати час для персональних тренувань. Інтерфейс сайту є зручним і інтуїтивно зрозумілим, що сприяє легкому використанню усіх функцій сайту.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 55   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

## ВИСНОВКИ

Під час виконання кваліфікаційної роботи бакалавра було спроектовано та реалізовано сайт для спортивного залу. Проведено аналіз предметної галузі та порівняння різних сайтів, що пропонують послуги спортивних залів та тренувань. На основі цього аналізу були визначені основні вимоги до сайту для спортивного залу. В процесі дослідження була розроблена архітектура, підходи та основні технології реалізації сайту.

Проектування сайту включало аналіз варіантів використання програмного продукту та проектування структури, включаючи моделювання взаємодії між складовими частинами. Також було спроектовано базу даних для зберігання інформації про користувачів, тренерів, розклад занять та обладнання. Особлива увага приділялась інтеграції з системами оплати та контролю доступу.

Описано алгоритм роботи з додатком та послідовні пункти варіантів його використання. Розглянуто основний функціонал, включаючи реєстрацію користувачів, створення та редагування профілів, бронювання обладнання та запис на тренування. Також описано функції для тренерів та адміністрації з управління розкладом та статистикою відвідувань.

Також була надана детальна схема розгортання сайту для спортивного залу, описано взаємодію структурних елементів системи та наведено опис інтерфейсу розробленого програмного продукту для різних типів користувачів. Показано роботу статистики для адміністрації та звичайних користувачів, що надає інформацію про активність відвідувачів.

Архітектура сайту побудована на базі сучасних вебтехнологій, що гарантує надійність та масштабованість системи. Додаток має інтуїтивний інтерфейс, який дозволяє користувачам легко реєструватися, бронювати обладнання та записуватися на тренування. Інтерфейс сайту є зручним і інтуїтивно зрозумілим, що сприяє легкому використанню усіх функцій сайту та задовольняє потреби різних категорій користувачів.

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 56   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ArenaGym [Електронний ресурс] – Режим доступу до ресурсу:  
<https://arenagym.com.ua/>
2. SportLife [Електронний ресурс] – Режим доступу до ресурсу:  
<https://sportlife.ua/uk/>
3. Gold's Gym [Електронний ресурс] – Режим доступу до ресурсу:  
<https://www.goldsgym.com/>
4. Lime Fitness [Електронний ресурс] – Режим доступу до ресурсу:  
<https://lime-fitness.com/main/>
5. Sharma, S. Learning MERN Stack. Birmingham: Packt Publishing Ltd, 2020. 340 p.
6. Kumar, N. MERN Quick Start Guide: Build web applications with MongoDB, Express.js, React, and Node. Birmingham: Packt Publishing Ltd, 2018. 192 p.
7. Siddharth, S., Waqar, F. Hands-On MERN Stack Web Development. Birmingham: Packt Publishing Ltd, 2019. 300 p.
8. Beck, K. Test-Driven Development: By Example. Boston: Addison-Wesley Professional, 2002. 240 p.
9. Meszaros, G. xUnit Test Patterns: Refactoring Test Code. Boston: Addison-Wesley Professional, 2007. 944 p.
10. Hungria, F. Testing JavaScript Applications. Birmingham: Packt Publishing Ltd, 2020. 538 p.
11. Ford, N. Functional and Reactive Domain Modeling. Boston: Addison-Wesley Professional, 2010. 392 p.
12. Sillito, J. JavaScript Testing Recipes: Using Mocha, Chai, and Sinon. Boston: Addison-Wesley Professional, 2013. 262 p.
13. Kosmaczewski, P. JavaScript Testing with Jasmine: JavaScript Behavior-Driven Development. Birmingham: Packt Publishing Ltd, 2016. 182 p.
14. O'Brien, T. Building React and Node.js Applications with Database Integration. Birmingham: Packt Publishing Ltd, 2021. 372 p.

15. Robinson, A. Mastering JavaScript Functional Programming. Birmingham: Packt Publishing Ltd, 2019. 324 p.
16. Myers, G.J. The Art of Software Testing. 3rd ed. Hoboken: John Wiley & Sons, 2011. 320 p.
17. MongoDB [Электронный ресурс] – Режим доступа до ресурсу: <https://docs.mongodb.com>
18. Express.js [Электронный ресурс] – Режим доступа до ресурсу: <https://expressjs.com/en/4x/api.html>
19. React Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://reactjs.org/docs/getting-started.html>
20. Node.js Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://nodejs.org/en/docs/>
21. Webpack Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://webpack.js.org/concepts/>
22. Redux Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://redux.js.org/introduction/getting-started>
23. TypeScript Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://www.typescriptlang.org/docs/>
24. Mongoose Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://mongoosejs.com/docs/guide.html>
25. React Hook Form Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://react-hook-form.com/get-started>
26. Tailwind CSS Documentation [Электронный ресурс] – Режим доступа до ресурсу: <https://tailwindcss.com/docs>
27. Elliott, E. Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Modern JS Libraries. Sebastopol: O'Reilly Media, 2014. 254 p.
28. Searls, J. The BDD Books: Cucumber. Leanpub, 2017. 120 p.
29. Liu, A. Express in Action: Writing, Building, and Testing Node.js Applications. Greenwich: Manning Publications, 2016. 375 p.

30. Wieruch, R. The Road to React: Your Journey to Master Plain Yet Pragmatic React. Independently Published, 2019. 254 p.
31. Eden, D. Learning React: Functional Web Development with React and Redux. Sebastopol: O'Reilly Media, 2017. 358 p.
32. Sams, J. Practical Node.js: Building Real-World Scalable Web Apps. 2nd ed. New York: Apress, 2016. 340 p.
33. Herron, A. React Design Patterns and Best Practices. Birmingham: Packt Publishing Ltd, 2018. 330 p.
34. Casciaro, A. Node.js Design Patterns. 2nd ed. Birmingham: Packt Publishing Ltd, 2016. 526 p.
35. Bhargava, V. Pro Express.js: Mastering the Node.js Framework. New York: Apress, 2014. 273 p.
36. Laverde, F. Hands-On JavaScript High Performance. Birmingham: Packt Publishing Ltd, 2019. 498 p.

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 59   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

# ДОДАТКИ

## Додаток А

### CreateCoursePage

```
import React, { useState } from 'react';
import { Container, Box, Typography, TextField, Button } from '@mui/material';
import { useNavigate } from 'react-router-dom';

const CreateCoursePage = () => {
  const [name, setName] = useState('');
  const [photo, setPhoto] = useState('');
  const [price, setPrice] = useState('');
  const [shortDescription, setShortDescription] = useState('');
  const [extendedDescription, setExtendedDescription] = useState('');
  const [error, setError] = useState(null);
  const navigate = useNavigate();

  const handlePhotoChange = (e) => {
    const file = e.target.files[0];
    const reader = new FileReader();
    reader.onloadend = () => {
      setPhoto(reader.result);
    };
    if (file) {
      reader.readAsDataURL(file);
    }
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    const token = localStorage.getItem('token');
    const response = await fetch('http://localhost:5000/api/courses', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${token}`,
      },
    });
  };
};
```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 60   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

    },
    body: JSON.stringify({ name, shortDescription, extendedDescription, photo,
price })),
  });

  if (response.ok) {
    alert('Курс створено успішно');
    navigate('/courses');
  } else {
    const data = await response.json();
    alert(data.message || 'Помилка при створенні курсу');
  }
};

return (
  <Container>
    <Box mt={4}>
      <Typography variant="h4" component="h1" gutterBottom>
        Створення курсу
      </Typography>
      <form onSubmit={handleSubmit}>
        <TextField
          label="Назва курсу"
          fullWidth
          margin="normal"
          value={name}
          onChange={(e) => setName(e.target.value)}
          required
        />
        <TextField
          label="Короткий опис"
          fullWidth
          margin="normal"
          value={shortDescription}
          onChange={(e) => setShortDescription(e.target.value)}
          required
        />
        <TextField
          label="Розширений опис"

```

```

        fullWidth
        margin="normal"
        value={extendedDescription}
        onChange={(e) => setExtendedDescription(e.target.value)}
        required
        multiline
        rows={4}
      />
      <TextField
        label="Фото"
        type="file"
        fullWidth
        margin="normal"
        onChange={handlePhotoChange}
      />
      <TextField
        label="Ціна"
        fullWidth
        margin="normal"
        type="number"
        value={price}
        onChange={(e) => setPrice(e.target.value)}
        required
      />
      <Button type="submit" variant="contained" color="primary" sx={{ mt: 2 }}>
        Створити курс
      </Button>
    </form>
  </Box>
</Container>
);
};

```

```
export default CreateCoursePage;
```

## CreateGroupPage

```

import React, { useState } from 'react';
import { Container, Box, Typography, TextField, Button } from '@mui/material';
import { useNavigate } from 'react-router-dom';

```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 62   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

const CreateGroupPage = () => {
  const [name, setName] = useState('');
  const [date, setDate] = useState('');
  const [time, setTime] = useState('');
  const [error, setError] = useState(null);
  const navigate = useNavigate();

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const response = await fetch('http://localhost:5000/api/groups', {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json',
          Authorization: `Bearer ${localStorage.getItem('token')}`,
        },
        body: JSON.stringify({ name, date, time }),
      });

      const data = await response.json();

      if (response.ok) {
        navigate('/groups');
      } else {
        setError(data.message);
      }
    } catch (err) {
      setError('Сталася помилка. Спробуйте ще раз.');
```

```

<form onSubmit={handleSubmit}>
  <TextField
    label="Назва"
    type="text"
    fullWidth
    margin="normal"
    value={name}
    onChange={(e) => setName(e.target.value)}
    required
  />
  <TextField
    label="Дата"
    type="date"
    fullWidth
    margin="normal"
    value={date}
    onChange={(e) => setDate(e.target.value)}
    required
  />
  <TextField
    label="Час"
    type="time"
    fullWidth
    margin="normal"
    value={time}
    onChange={(e) => setTime(e.target.value)}
    required
  />
  <Button type="submit" variant="contained" color="primary" fullWidth>
    Створити групу
  </Button>
</form>
</Box>
</Container>
);
};

export default CreateGroupPage;

```

## UserProfilePage

```
import React, { useState, useEffect } from 'react';
import { Container, Box, Typography, TextField, Button, Avatar } from
 '@mui/material';

const ProfilePage = () => {
  const [user, setUser] = useState(null);
  const [editing, setEditing] = useState(false);
  const [name, setName] = useState('');
  const [age, setAge] = useState('');
  const [weight, setWeight] = useState('');
  const [photo, setPhoto] = useState('');
  const [error, setError] = useState(null);

  useEffect(() => {
    const userInfo = JSON.parse(localStorage.getItem('userInfo'));
    const token = localStorage.getItem('token');

    console.log('userInfo:', userInfo);
    console.log('token:', token);

    if (!userInfo || !userInfo._id || !token) {
      setError('Користувач не авторизований або токен не знайдено');
      return;
    }

    fetchUserDetails(userInfo._id, token);
  }, []);

  const fetchUserDetails = async (userId, token) => {
    try {
      const response = await fetch(`http://localhost:5000/api/users/${userId}`, {
        headers: {
          'Authorization': `Bearer ${token}`,
        },
      });
      const data = await response.json();
      if (response.ok) {
```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 65   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

        setUser(data);
        setName(data.name);
        setAge(data.age);
        setWeight(data.weight);
        setPhoto(data.photo);
    } else {
        setError(data.message);
    }
} catch (err) {
    console.error('Error fetching user details:', err);
    setError('Сталася помилка. Спробуйте ще раз.');
```

```

};

const handlePhotoChange = (e) => {
    const file = e.target.files[0];
    const reader = new FileReader();
    reader.onloadend = () => {
        setPhoto(reader.result);
    };
    if (file) {
        reader.readAsDataURL(file);
    }
};

const handleSubmit = async (e) => {
    e.preventDefault();
    const token = localStorage.getItem('token');
    const userInfo = JSON.parse(localStorage.getItem('userInfo'));

    console.log('Submitting profile update with token:', token);
    console.log('userInfo:', userInfo);

    if (!token || !userInfo || !userInfo._id) {
        setError('Не вдалося отримати токен авторизації або інформацію про користувача');
        return;
    }
};

```

```

try {
  const response = await fetch(`http://localhost:5000/api/users/profile`, {
    method: 'PUT',
    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`,
    },
    body: JSON.stringify({ name, age, weight, photo }),
  });
  const data = await response.json();
  if (response.ok) {
    setUser(data);
    setEditing(false);
    localStorage.setItem('userInfo', JSON.stringify(data)); // Оновлення
userInfo в localStorage
    localStorage.setItem('token', data.token); // Оновлення токена в
localStorage
  } else {
    setError(data.message);
  }
} catch (err) {
  console.error('Error updating user profile:', err);
  setError('Сталася помилка. Спробуйте ще раз.');
```

```

}
if (error) {
  return <Typography color="error">{error}</Typography>;
}

```

```

if (!user) {
  return <Typography>Завантаження...</Typography>;
}

```

```

return (
  <Container>
    <Box mt={4}>
      <Typography variant="h4" component="h1" gutterBottom>
        Профіль

```

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                    | 67   |

```

</Typography>
<Avatar alt={name} src={photo} sx={{ width: 100, height: 100 }} />
{!photo && <Typography>Фото за замовчуванням</Typography>}
<Typography variant="h6">{name}</Typography>
<Typography variant="body1">{user.email}</Typography> {/* Відображення
електронної пошти */}
<Typography variant="body1">Вік: {age}</Typography>
<Typography variant="body1">Вага: {weight}</Typography>
<Typography variant="body1" sx={{ color: user.vipStatus ? 'green' : 'red'
}}>
    {user.vipStatus ? 'VIP Абонемент активний' : 'VIP Абонемент не активний'}
</Typography>
<Button variant="contained" color="primary" onClick={() =>
setEditing(true)}>
    Редагувати
</Button>
{editing && (
    <form onSubmit={handleSubmit}>
        <TextField
            label="Ім'я"
            type="text"
            fullWidth
            margin="normal"
            value={name}
            onChange={(e) => setName(e.target.value)}
            required
        />
        <TextField
            label="Вік"
            type="number"
            fullWidth
            margin="normal"
            value={age}
            onChange={(e) => setAge(e.target.value)}
            required
        />
        <TextField
            label="Вага"
            type="number"

```

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 68   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

```

        fullWidth
        margin="normal"
        value={weight}
        onChange={(e) => setWeight(e.target.value)}
        required
      />
      <TextField
        label="Фото"
        type="file"
        fullWidth
        margin="normal"
        onChange={handlePhotoChange}
      />
      {photo && <img src={photo} alt="Фото" style={{ maxHeight: '150px' }}
    />}

    <Button type="submit" variant="contained" color="primary" fullWidth>
      Зберегти зміни
    </Button>
  </form>
)}
</Box>
</Container>
);
};

export default ProfilePage;

```

## CourseListPage

```

import React, { useEffect, useState } from 'react';
import { Container, Box, Typography, Grid, Paper, Button } from '@mui/material';
import { useNavigate } from 'react-router-dom';

const CoursesPage = () => {
  const [courses, setCourses] = useState([]);
  const [purchasedCourses, setPurchasedCourses] = useState([]);
  const userInfo = JSON.parse(localStorage.getItem('userInfo'));

```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 69   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

const navigate = useNavigate();

useEffect(() => {
  fetchCourses();
  if (userInfo) {
    fetchPurchasedCourses();
  }
}, []);

const fetchCourses = async () => {
  const response = await fetch('http://localhost:5000/api/courses');
  const data = await response.json();
  setCourses(Array.isArray(data) ? data : []);
};

const fetchPurchasedCourses = async () => {
  const token = localStorage.getItem('token');
  const response = await
fetch('http://localhost:5000/api/payment/purchases/courses', {
  headers: {
    'Authorization': `Bearer ${token}`,
  },
});
  const data = await response.json();
  if (Array.isArray(data)) {
    setPurchasedCourses(data.map(purchase => purchase.course._id));
  } else {
    setPurchasedCourses([]);
  }
};

const handlePurchaseCourse = async (courseId) => {
  const token = localStorage.getItem('token');
  const response = await fetch('http://localhost:5000/api/payment/purchase-
course', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`,

```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 70   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

    },
    body: JSON.stringify({ courseId }),
  });

  const data = await response.json();

  if (response.ok) {
    fetchPurchasedCourses();
    alert('Курс придбано успішно');
  } else {
    alert(data.message || 'Помилка при покупці курсу');
  }
};

const handleClick = (courseId) => {
  navigate(`/courses/${courseId}`);
};

const handleCreateCourse = () => {
  navigate('/create-course');
};

return (
  <Container>
    <Box mt={4}>
      <Typography variant="h4" component="h1" gutterBottom>
        Курси
      </Typography>
      {userInfo && userInfo.role === 'trainer' && (
        <Button variant="contained" color="primary" onClick={handleCreateCourse}
sx={{ mb: 2 }}>
          Створити курс
        </Button>
      )}
      <Grid container spacing={3}>
        {courses.map((course) => (
          <Grid item xs={12} sm={6} md={4} key={course._id}>
            <Paper elevation={3} sx={{ padding: 3 }}>
              <img

```

```

        src={course.photo}
        alt={course.name}
        style={{ width: '100%', height: '200px', objectFit: 'cover' }}
        onClick={() => handleCourseClick(course._id)}
      />
      <Typography variant="h6" gutterBottom>
        {course.name}
      </Typography>
      <Typography variant="body2" gutterBottom>
        {course.shortDescription}
      </Typography>
      <Typography variant="body1" color="primary">
        {course.price} грн
      </Typography>
      {userInfo && purchasedCourses.includes(course._id) ? (
        <Button variant="contained" color="primary" fullWidth disabled>
          Ви вже купили цей курс
        </Button>
      ) : (
        <Button variant="contained" color="primary" fullWidth onClick={() =>
=> handlePurchaseCourse(course._id)}>
          Придбати
        </Button>
      )}
    </Paper>
  </Grid>
  )})
</Grid>
</Box>
</Container>
);
};

```

```
export default CoursesPage;
```

GroupListPage

```

import React, { useEffect, useState } from 'react';
import { Container, Box, Typography, Grid, Paper, Button } from '@mui/material';

```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
|      |      |          |        |      |                     | 72   |
| Змн. | Арк. | № докум. | Підпис | Дата |                     |      |

```

import { useNavigate } from 'react-router-dom';

const GroupsPage = () => {
  const [groups, setGroups] = useState([]);
  const navigate = useNavigate();
  const userInfo = JSON.parse(localStorage.getItem('userInfo'));

  useEffect(() => {
    const fetchGroups = async () => {
      const response = await fetch('http://localhost:5000/api/groups');
      const data = await response.json();
      setGroups(data);
    };

    fetchGroups();
  }, []);

  const handleGroupClick = (groupId) => {
    navigate(`/groups/${groupId}`);
  };

  return (
    <Container>
      <Box mt={4}>
        <Typography variant="h4" component="h1" gutterBottom>
          Групи
        </Typography>
        {userInfo && userInfo.role === 'trainer' && (
          <Button
            variant="contained"
            color="primary"
            onClick={() => navigate('/create-group')}
            sx={{ mb: 3 }}
          >
            Створити нову групу
          </Button>
        )}
        <Grid container spacing={3}>
          {groups.map((group) => (

```

```

        <Grid item xs={12} sm={6} md={4} key={group._id}>
          <Paper elevation={3} sx={{ padding: 3 }} onClick={() =>
handleGroupClick(group._id)} style={{ cursor: 'pointer' }}>
            <Typography variant="h6" gutterBottom>
              {group.name}
            </Typography>
            <Typography variant="body1">
              Тренер: {group.trainer.name}
            </Typography>
            <Typography variant="body1">
              {new Date(group.date).toLocaleString('uk-UA')}
            </Typography>
          </Paper>
        </Grid>
      )})
    </Grid>
  </Box>
</Container>
);
};

export default GroupsPage;

```

## HomePage

```

import React from 'react';
import { Container, Box, Typography, Grid, Paper } from '@mui/material';
import { useNavigate } from 'react-router-dom';

const HomePage = () => {
  const navigate = useNavigate();

  const handleNavigation = (path) => {
    navigate(path);
  };

  return (
    <Container>
      <Box mt={4} textAlign="center">

```

|      |      |          |        |      |                     |      |
|------|------|----------|--------|------|---------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ІІЗ | Арк. |
| Змн. | Арк. | № докум. | Підпис | Дата |                     | 74   |

```
<Typography variant="h4" component="h1" gutterBottom>
```

Ласкаво просимо до Тренажерного залу!

```
</Typography>
```

```
<Typography variant="body1" gutterBottom>
```

Тут ви можете знайти найкращих тренерів, групові заняття, курси та багато іншого.

```
</Typography>
```

```
<Grid container spacing={3} justifyContent="center">
```

```
<Grid item xs={12}>
```

```
<Paper
```

```
  onClick={() => handleNavigation('/schedule')}>
```

```
    elevation={3}
```

```
    sx={{ padding: 3, cursor: 'pointer', border: '1px solid red' }}
```

```
>
```

```
<Typography variant="h5" gutterBottom>
```

Розклад

```
</Typography>
```

```
</Paper>
```

```
</Grid>
```

```
<Grid item xs={12}>
```

```
<Paper
```

```
  elevation={3}
```

```
  sx={{ padding: 3, border: '1px solid red' }}
```

```
>
```

```
<Typography variant="h5" gutterBottom>
```

Ніколи не зупиняйся на півдорозі до своєї мети, навіть коли здається, що сили закінчуються. Пам'ятай, що найбільший прогрес приходить через подолання найважчих перешкод.

```
</Typography>
```

```
</Paper>
```

```
</Grid>
```

```
<Grid item xs={12} sm={4}>
```

```
<Paper
```

```
  onClick={() => handleNavigation('/trainers')}>
```

```
    elevation={3}
```

```
    sx={{ padding: 3, cursor: 'pointer', border: '1px solid red' }}
```

```
>
```

```
<Typography variant="h6" gutterBottom>
```

Тренери

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІІЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 75   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |

```

        </Typography>
      </Paper>
    </Grid>
    <Grid item xs={12} sm={4}>
      <Paper
        onClick={() => handleNavigation('/groups')}
        elevation={3}
        sx={{ padding: 3, cursor: 'pointer', border: '1px solid red' }}
      >
        <Typography variant="h6" gutterBottom>
          Групи
        </Typography>
      </Paper>
    </Grid>
    <Grid item xs={12} sm={4}>
      <Paper
        onClick={() => handleNavigation('/courses')}
        elevation={3}
        sx={{ padding: 3, cursor: 'pointer', border: '1px solid red' }}
      >
        <Typography variant="h6" gutterBottom>
          Курси
        </Typography>
      </Paper>
    </Grid>
  </Grid>
</Box>
</Container>
);
};

export default HomePage;

TrainerListPage

import React, { useEffect, useState } from 'react';
import { Container, Box, Typography, Grid, Paper, Avatar } from '@mui/material';

const TrainersPage = () => {

```

```

const [trainers, setTrainers] = useState([]);

useEffect(() => {
  const fetchTrainers = async () => {
    const response = await fetch('http://localhost:5000/api/trainers');
    const data = await response.json();
    setTrainers(data);
  };

  fetchTrainers();
}, []);

return (
  <Container>
    <Box mt={4}>
      <Typography variant="h4" component="h1" gutterBottom>
        Тренери
      </Typography>
      <Grid container spacing={3}>
        {trainers.map((trainer) => (
          <Grid item xs={12} sm={6} md={4} key={trainer._id}>
            <Paper elevation={3} sx={{ padding: 3, textAlign: 'center' }}>
              <Avatar alt={trainer.name} src={trainer.photo} sx={{ width: 100,
height: 100, margin: '0 auto' }} />
              <Typography variant="h6" gutterBottom>
                {trainer.name}
              </Typography>
              <Typography variant="body1">
                {trainer.email}
              </Typography>
            </Paper>
          </Grid>
        ))}
      </Grid>
    </Box>
  </Container>
);
};

```

```
export default TrainersPage;
```

|      |      |          |        |      |                    |      |
|------|------|----------|--------|------|--------------------|------|
|      |      |          |        |      | ІПЗ.КР.Б-121-24-ПЗ | Арк. |
|      |      |          |        |      |                    | 78   |
| Змн. | Арк. | № докум. | Підпис | Дата |                    |      |