Державний вищий навчальний заклад
“Прикарпатський національний університет імені Василя Стефаника”
Кафедра інформаційних технологій

УДК 004

ДИПЛОМНИЙ ПРОЕКТ

Тема: Розробка бекенд частини для веб-платформи з проведення спортивних змагань

Спеціальність 121 Інженерія програмного забезпечення
код і назва спеціальності

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДП.ІПЗ-16.ПЗ

(позначення)

Рецензент
доцент Лазарович І.М.
(посада) (підпис) (дата) (розшифровка підпису)

Студент
ІПЗ-41 Машталер Т.Т.
(шифр групи) (підпис) (дата) (розшифровка підпису)

Нормоконтролер
доцент Лазарович І.М.
(посада) (підпис) (дата) (розшифровка підпису)

Керівник дипломного проекту
ст.викладач Пікуляк М.В.
(посада) (підпис) (дата) (розшифровка підпису)

Допускається до захисту

Завідувач кафедри

доцент Козленко М.І.
(посада) (підпис) (дата) (розшифровка підпису)

2020
(рік)

Державний вищий навчальний заклад
«Прикарпатський національний університет імені Василя Стефаника»
Факультет математики та інформатики Кафедра інформаційних технологій
Спеціальність 121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Завідувач кафедри Козленко М.І.
_____
„_____”_____ 20___р.

ЗАВДАННЯ
НА ВИКОНАННЯ ДИПЛОМНОГО ПРОЕКТУ

Студенту Машталеру Тарасу Тарасовичу
(прізвище, ім’я, по батькові студента)

1. Тема проекту. Розробка бекенд частини для веб-платформи з проведення спортивних змагань
затверджена розпорядженням по факультету математики та інформатики від „25” жовтня 2019р.№7

2. Термін здачі студентом закінченого проекту 22 травня 2020р

3. Вихідні дані до дипломного проекту: зразки жеребкування, дизайн, технологія розробки серверної частини Laravel

4. Зміст пояснювальної записки (перелік питань, що їх належить опрацювати)
1.Аналіз предметної області системи організації змагань
2.Моделювання та формування вимог програмного засобу
3.Програмна реалізація бекенд частини
4. Бізнес-план розробки вебплатформи

5. Перелік графічного матеріалу (з точним забезпеченням обов’язкових креслень) Титульний аркуш, Мета створення системи з організації проведення спортивних змагань, Огляд сфери фізичного виховання та існуючих програмних засобів, Моделювання та аналіз програмного забезпечення, Архітектура бекенд частини, Програмна реалізація бекенд частини, Економічна частина розробки, Висновки

6. Дата видачі завдання 11.09.2019

Керівник _____ Пікуляк М.В.
(підпис) (розшифровка підпису)

Завдання прийняв до виконання _____ Машталер Т.Т.
(підпис) (розшифровка підпису)

КАЛЕНДАРНИЙ ПЛАН

Номер і назва етапів дипломного проекту	Термін виконання етапів проекту	Примітка
1.Аналіз предметної області системи організації змагань	02.12.2019	Виконав
2.Моделювання та формування вимог програмного засобу	15.02.2020	Виконав
3.Програмна реалізація бекенд частини	17.04.2020	Виконав
4.Бізнес-план розробки вебплатформи	11.05.2020	Виконав
5.Оформлення пояснювальної записки	18.05.2020	Виконав

Студент _____ Машталер Т.Т.
(підпис) (розшифровка підпису)

Керівник проекту _____ Пікуляк М.В.
(підпис) (розшифровка підпису)

РЕФЕРАТ

Пояснювальна записка: 84 сторінки (без додатків), 33 рисунки, 26 таблиць, 23 джерела, 1 додаток на 30 сторінках.

Ключові слова: WEB-ПЛАТФОРМА, ОРГАНІЗАЦІЯ ЗМАГАННЯ, ЖЕРЕБКУВАННЯ, РЕЗУЛЬТАТИ ЗМАГАННЯ, PHP.

Об`єктом дослідження є процеси автоматизації організації та проведення змагань.

Предметом дослідження є методи та засоби побудови бекенд частини проведення жеребкування спортивних змагань.

Мета роботи: розробка бекенд частини, застосування якої дозволяє забезпечити клієнт серверну взаємодію та реалізувати логічну частину роботи веб ресурсу.

Стислий опис тексту пояснювальної записки:

Для організації баз даних використано Microsoft SQL server Management Studio 2016.

Одержані результати полягають в розробці бекенд частини для веб платформи з проведення спортивних змагань, що дозволить переглядати результати змагання та формувати сітку жеребкування автоматично.

ABSTRACT

Explanatory note: 84 pages (without appendix), 33 figures, 26 tables, 23 references, 1 appendix on 30 pages.

Keywords: WEB-PLATFORM, COMPETITION ORGANIZATION, DRAW, COMPETITION RESULTS, PHP.

The object of research is the automation of the process of organizing and conducting competitions.

The subject of the study are the methods and means of building a backend part of the draw of sports competitions.

Purpose: to develop a backend part of a web platform for sports competitions to automate the process of organizing sports fights and to automate the draw process.

Brief description of the explanatory note:

Development methods are based on PHP technology in the text code editor Sublime Text 3. Microsoft SQL server Management Studio 2016 was used to organize databases.

The obtained results are the development of a backend part for a web platform for sports competitions, which will allow you to view the results of the competition and form a grid of draws automatically.

ЗМІСТ

ВСТУП

У сучасних умовах проведення спортивних змагань, формування графіку боїв та жеребкування учасників спортивних змагань це довгий і трудомісткий процес. На сьогодні актуальною задачею є оптимізація процесів жеребкування та формування графіків боїв. Тому ми пропонуємо вирішити цю проблему, розробивши веб-платформу для проведення спортивних змагань, яка буде виконувати поставлене завдання та буде зручна у використанні.

Даний дипломний проект буде дозволяти тренерам спортивних організацій створювати змагання, де вони зможуть додавати вікові та вагові групи учасників змагань, та додавати чи видаляти учасників. В результаті тренери матимуть змогу автоматично сформувати списки жеребкування та проведення боїв, також будуть відображатись статистичні дані по кожному з учасників спортивних змагань.

Для стабільної роботи, системі потрібен процесор з частотою від 2 ГГц, 1 Гб оперативної пам’яті та каналом інтернету від 10 Мбіт/сек. Back-end частина дипломної роботи реалізована на мові програмування PHP. Це сервлет обгортка для взаємодії з бекендом. На бекенді використовується фреймворк Laravel, як СУБД було обрано MySQL оскільки вона безкоштовна та дає одні з найкращих технічних показників по швидкодії, масштабованості, переносності, зв’язності та простоті використання.

Об’єктом даного проекту є процеси автоматизації організації та проведення змагань, а предметом є методи та засоби побудови бекенд частини програмного продукту з проведення жеребкування спортивних змагань.

Мета проекту – розробка бекенд частини, застосування якої дозволяє забезпечити клієнт серверну взаємодію та реалізувати логічну частину роботи веб ресурсу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ СИСТЕМИ ОРГАНІЗАЦІЇ ЗМАГАНЬ

1.1 Коротка характеристика об’єкту управління

1.1.1 Напрям діяльності

Відвідування спортивної секції є запорукою здоров’я та підтримки відмінної фізичної форми на роки. Спортивна інфраструктура надає безліч можливостей занять спортом і для аматорів, і для професійних спортсменів. Багато колишніх професійних спортсменів надалі пов’язують своє життя із спортом, тренуючи молоде покоління.

Різноманітні спортивні секції та спеціальні школи направляють зусилля на формування та розвиток хорошого відношення до спорту, розпочинаючи з шахів та закінчуючи культуризмом. Основною метою діяльності офіційних спортивних організацій є проведення спортивних змагань на всеукраїнському рівні та на рівні університету.

Для себе можна обрати будь-який вид спортивних занять, розпочинаючи з екстремальних, завершуючи щоденною ранковою гімнастикою. Також можна відвідувати спортивні клуби, спортзали, фітнес-центри, тренажерні зали або басейн, займатися з тренером чи обрати групові заняття. Періодичне відвідування спортивних установ (тренажерний зал, спортзал, басейн і т.д.) або самостійне виконання фізичних вправ дозволяє не тільки виробити самодисципліну, але й сприяють всебічному розвитку, гартують і оздоровлюють організм. Заняття спортом призначені зберегти здоров’я, надати заряд бадьорості і гарний настрій.

Об’єктом управління є процес роботи ГО ЦЕНТР ПІДГОТОВКИ ЧЕМПІОНІВ "ЗОЛОТИЙ ДРАКОН". В даному центрі діє спортивна секція з тхеквондо та проводяться змагання з цього виду спортивних єдиноборств.

Діяльність даного центру з підготовки чемпіонів здійснюється за трьома напрямками:

- розвиток спорту вищих досягнень (підготовка спортсменів високого класу);
- розвиток масових видів спорту (підготовлення спортсменів до участі у внутрішних та зовнішних спортивних заходах);
- впровадження та поширення нових розділів східних єдиноборств.

1.1.2 Схема організаційної структури управління

Організаційна структура управління – це впорядкований набір взаємопов’язаних елементів, які перебувають у стабільній взаємозв'язку один з одним, забезпечуючи їх розвиток та функціонування в цілому [1,2].

Організаційна структура управління націлена на встановлення чітких зв’язків між окремими секторами організації, розподіл прав та обов'язків між ними.

Зв’язки між елементами структури управління бувають:

- вертикальними, коли відбувається взаємодія між керівником і підлеглим (наприклад, зв’язок між директором фірми і керуючим структурним підрозділом);
- горизонтальними, коли відбувається взаємодія рівноправних елементів (наприклад, зв’язку між керуючими структурними підрозділами одного рівня).

Види відносин всередині організації аналогічні типу побудови структури її управління і діляться на:

- лінійні відносини – це відносини між керівником і його підлеглими;

- функціональні відносини – це відносини фахівця, який уповноважений виконувати ту чи іншу функцію в рамках всієї організації, з іншими членами організації;
- відносини управлінського апарату, даний тип відносин має місце у разі подання будь-яких прав і повноважень. Посадові обов'язки при цьому полягають у наданні рекомендації, рад.

У Центрі підготовки чемпіонів «Золотий Дракон» працюють тренери з різними розрядами, а саме: майстер спорту України з тхеквондо ВТФ, кандидат в майстри спорту, та з І розрядом. Звання та розряди з тхеквондо присвоюються за виконання норм і вимог Єдиної спортивної класифікації України (ЄСКУ). Норми і вимоги ЄСКУ залежать від версії тхеквондо, дисципліни, в якій виступав спортсмен, і зайнятого підсумкового місця. В окремих випадках існують вимоги щодо проведення певної кількості боїв, поєдинків, єдиноборств для отримання спортивного звання або розряду.

Спортивні звання та спортивні розряди з тхеквондо присвоюються спортсменам за віковими групами:

- без обмеження верхньої межі віку – чоловіки, жінки;
- з обмеженням верхньої межі віку – кадети та кадетки, юніори та дівчата.

Щоб отримати спортивний розряд потрібно досягнути віку – 10 років.

Тренери, котрі мають дані звання відповідають за спортивні змагання та підготовку учасників до них. Також вони формують відвідуваність учасників спортивної секції, яка належать даному центру підготовки чемпіонів та стежать за їх успіхами.

Для кращого розуміння ієрархії об'єкту управління та спортивних розрядів тренерів, на рисунку 1.1. представлено діаграму з видами спортивних звань і розрядів.

Рисунок 1.1 – Структура спортивних звань та розрядів

Керівником ГО Центр підготовки чемпіонів «Золотий дракон» є президент центру, він же головний тренер. Усі працівники тренувального центру чемпіонів підкоряються йому. У президента є ряд тренерів з різними спортивними званнями та званнями, а тренери в свою чергу вже мають спортсменів у формі учасників змагань. Тож навчальний центр чемпіонів має вертикальні зв’язки, а також усі різновиди відносин.

1.2 Опис предметної області організації змагань

1.2.1 Опис функціоналу системи

Цифровізація сфер діяльності людини залежить від масштабу та структури того, що необхідно комп'ютеризувати. Головною ціллю є створення єдиної інформаційної структури, що б давала можливість людям легко і правильно ввести інформацію в систему управління, швидко знайти необхідну інформацію та зручно обмінюватися нею. Для досягнення мети будь-які процеси взаємодії людей стають зручними, зрозумілими та більш контрольованими, займаючи на порядок менше затраченого часу. Тому гроші, що витрачаються на комп'ютеризацію, необхідно обґрунтувати саме тими цілями, досягнення яких зробить той чи інший процес максимально ефективними [3].

Предметною областю дипломної роботи є автоматизація процесу організації спортивних змагань та генерації сітки жеребкування, яка пришвидшить процес внесення нових змагань спортивної організації, формування та перегляд графіку змагань. Веб-платформа розроблена з розрахунку на різні рівні управління роботою веб-сайту:

- можливість додавання та редагування даних;
- забезпечення єдиного підходу до порядку організації інформації;
- можливість постійного оперативного впливу на процес організації змагань;
- значне підвищення доступу до інформації по змаганнях.

При роботі з веб-платформою для організації спортивних змагань зазвичай існує два базові актори: Адмін – той хто створює змагання та сітки жеребкування, та Тренер – той хто має можливість зареєструватися на сайті та додавати чи видаляти учасників змагань. Також є актор – Користувач, якому

доступна функція перегляду списку змагань та учасників, які беруть участь у змаганнях.

Відповідно існує такий список функцій:

- реєстрація Адміна;
- реєстрація тренера;
- створення змагання;
- видалення змагання;
- редагування змагання;
- відображення списку змагань;
- створення нового учасника;
- редагування профілю учасника;
- видалення учасника;
- перегляд списку учасників змагання;
- створення вікових груп;
- створення вагових груп;
- побудова сітки жеребкування;
- створення та перегяд результатів змагання у різних форматах;
- друк сітки жеребкування конкретної категорії або всіх.

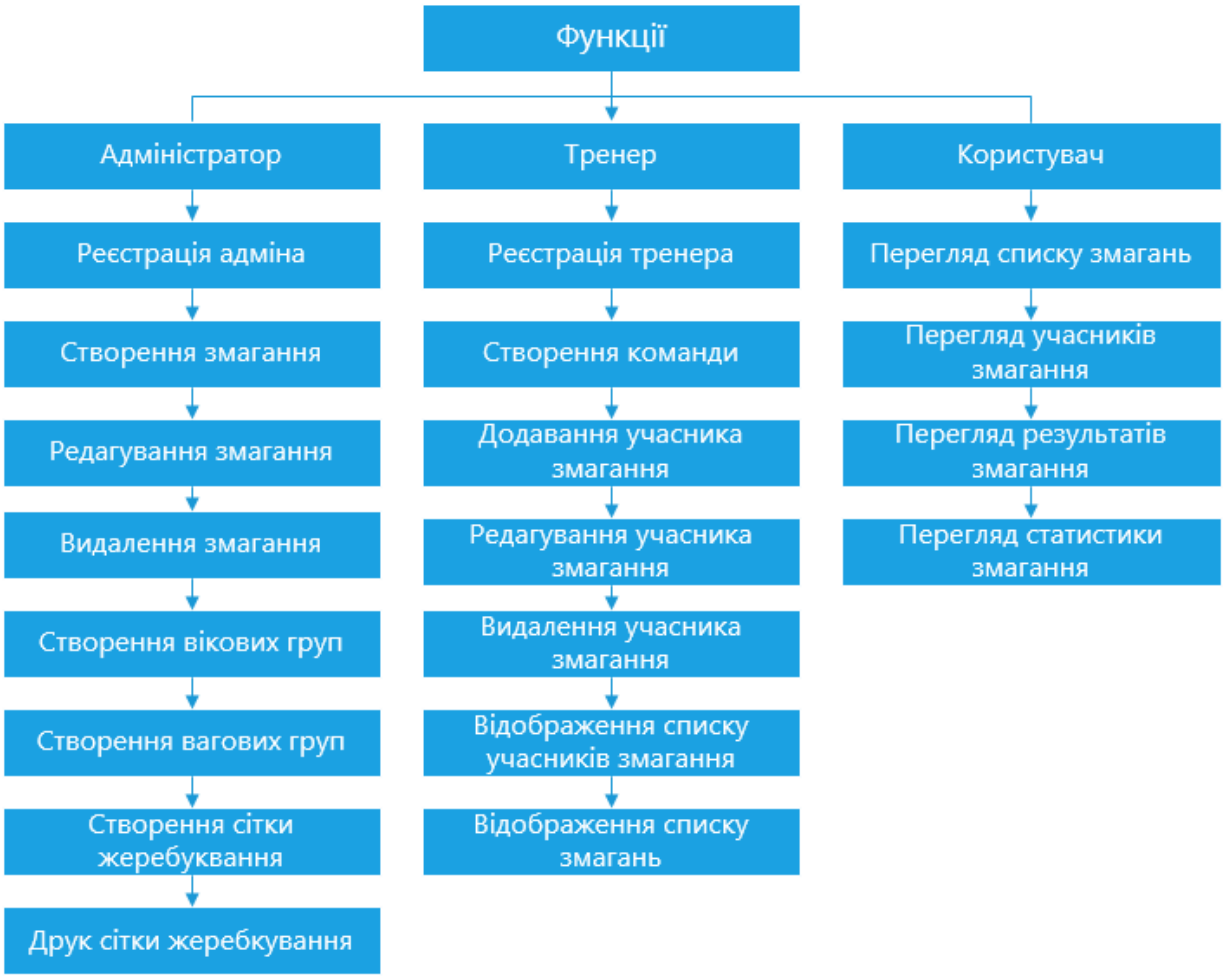

Рисунок 1.2 – Дерево функції для веб-платформи з організації спортивних змагань

1.2.2 Опис бізнес-процесів

Щоб повністю представити функціональність програми, виділено основні бізнес-процеси, які представлені на рисунку 1.3.

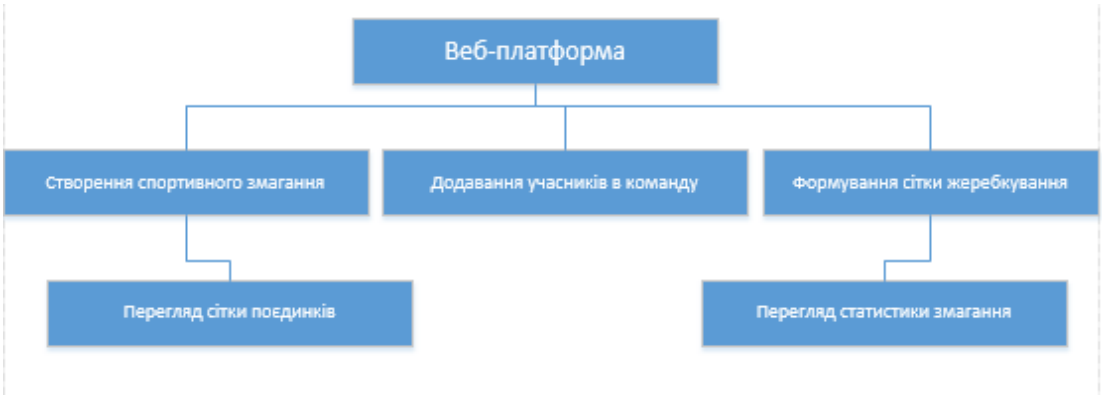

Рисунок 1.3 – Діаграма бізнес-процесів розроблюваної веб-платформи

Наступним кроком, ми детальніше розглянемо вище представлені бізнес-процеси. На рисунку 1.4. зображено процес створення спортивного змагання.

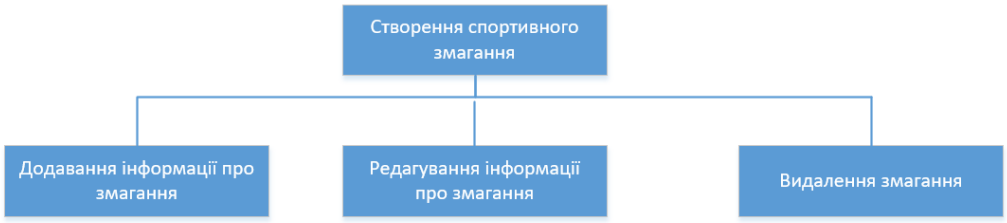

Рисунок 1.4 – Діаграма бізнес-процесу створення спортивного змагання

Перше, що потрібно зробити адміністратору - це вибрати вкладку «Змагання», щоб мати можливість переглядати інформацію про існуючі змагання та додавати новий конкурс для подальшого додавання його до бази даних. Створення спортивного змагання поділяється на такі особливості:

- додавання інформації про змагання - адміністратор повинен ввести відповідну інформацію у полі для створення;
- редагування інформації про змагання - адміністратор має можливість змінити інформацію в існуючому конкурсі;
- видалення змагання – адміністратор має можливість видалити змагання.

Деталі бізнес-процесу з створення спортивного змагання адміністратором наведено в таблиці 1.1.

Таблиця 1.1 – Деталі бізнес-процесу з створення спортивного змагання

Назва характеристики	**Значення характеристики**
Ім’я бізнес-процесу	Створення спортивного змагання
Основні учасники	Адміністратор
Вхідна подія	Запит на проведення змагань
Вхідні документи	Інформація про дату змагань, місце змагань, вид спорту, кількість даянгів, ліміт учасників ,програма змагань, головний суддя,головний секретар, вікові категорії.
Вихідна подія	Перегляд сформованого спортивного змагання
Вихідні документи	Сформована інформація про змагання
Клієнт бізнес-процесу	Тренер, учасник змагання

На рисунку 1.5. показано діаграму функцій процесу формування сітки жеребкування (виконання завдання).

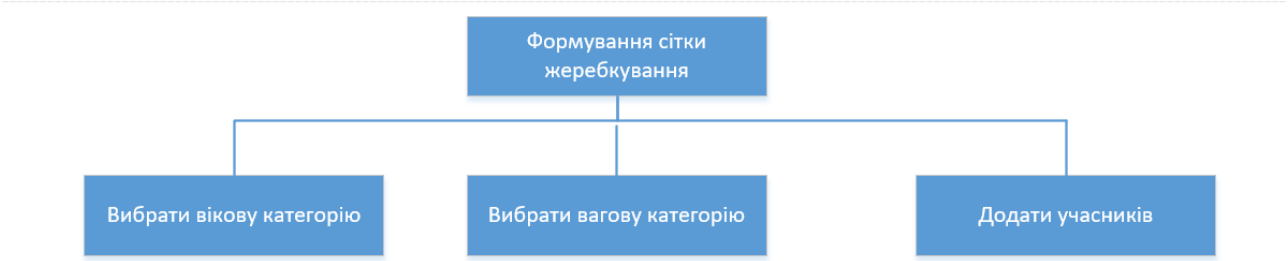

Рисунок 1.5 – Діаграма бізнес-процесу формування сітки жеребкування

Наступне, що повинен зробити адміністратор – створити сітку для спортивних змагань. Це найважливіший процес, оскільки він формує мережу боїв.

Формування сітки поділяється на такі особливості:

- вибір вікової категорії - це вибір доступних вікових категорій учасників;

- вибір вагової категорії - це вибір доступних вагових категорій учасників змагань;

- додавання учасників - додайте учасників до змагань та вибраних вікових та вагових категорій.

Характеристика бізнес-процесу формування сітки лотів адміністратором наведена в таблиці 1.2.

Таблиця 1.2 – Характеристика бізнес-процесу формування сітки лотів

Назва характеристики	**Значення характеристики**
Ім’я бізнес-процесу	Формування сітки жеребкування
Основні учасники	Адміністратор
Вхідна подія	Запит на проведення змагання
Вхідні документи	Інформація про вікові та вагові категорії учасників змагання
Вихідна подія	Перегляд сітки жеребкування учасниками змагань
Вихідні документи	Оновлена сітка проведення боїв
Клієнт бізнес-процесу	Тренер,учасники змагань

На рисунку 1.6 показана схема функцій процесу перегляду сітки матчів обраного змагання спортивного центру для тренувань чемпіонів. Сітка боїв буде доступна за 1 день до боїв, це залежить від кількості даянгів (рингів).

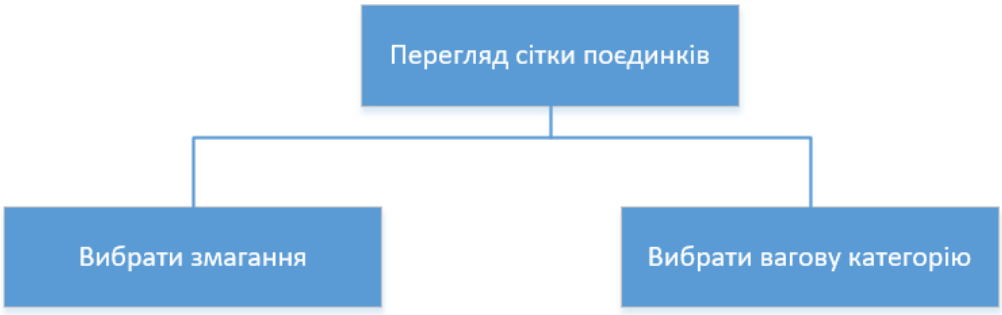

Рисунок 1.6 – Діаграма бізнес-процесу формування сітки жеребкування

Процес перегляду сітки боїв є останнім кроком і служить для легкого перегляду сітки боїв для користувачів веб-платформи.

Характеристика бізнес-процесу перегляду сітки поєдинків наведена в таблиці 1.3.

Таблиця 1.3 – Характеристика бізнес-процесу перегляду сітки поєдинків

Назва характеристики	**Значення характеристики**
Ім’я бізнес-процесу	Перегляд сітки поєдинків
Основні учасники	Користувач, адміністратор,тренер
Вхідна подія	Запит на здійснення перегляду
Вхідні документи	Запит на перегляд
Вихідна подія	Відображення графіка по необхідній віковій категорії
Вихідні документи	Актуальна сітка поєдинків
Клієнт бізнес-процесу	Процес додавання учасників змагання

На рисунку 1.7. показано діаграму функцій процесу додавання учасника в команду.

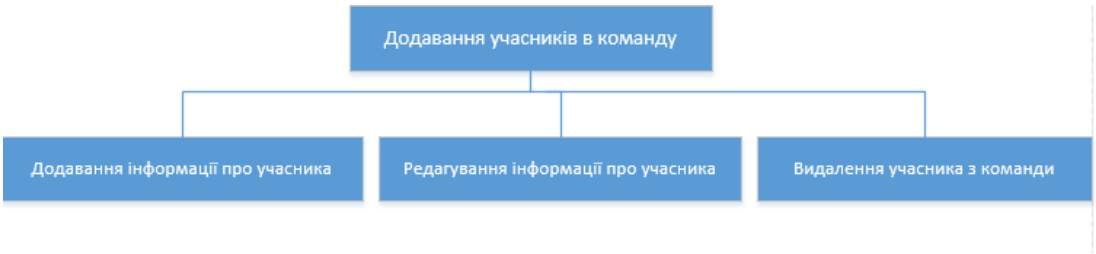

Рисунок 1.7 – Діаграма бізнес-процесу додавання учасників в команду

Перше, що тренер повинен зробити, - це вибрати вкладку «Команда», щоб мати змогу додати нового конкурента для подальшого додавання його до бази даних.

Додавання учасника до команди поділяється на такі особливості:

- додавання інформації про учасника;

- тренер повинен заповнити форму відповідними полями для введення інформації;

- редагування інформації про учасника - тренер або адміністратор має можливість змінити інформацію в існуючому профілі учасника;

- зняття змагань - тренер чи адміністратор мають можливість відсторонити учасника.

Характеристика бізнес-процесу створення спортивних змагань адміністратором наведена в таблиці 1.4.

Таблиця 1.4 – Характеристика бізнес-процесу додавання учасника в команду

Назва характеристики	**Значення характеристики**
Ім’я бізнес-процесу	Додавання учасника в команду
Основні учасники	адміністратор,тренер
Вхідна подія	Запит на створення команди
Вхідні документи	Інформація про учасника
Вихідна подія	Відображення профілю нового учасника команди
Вихідні документи	Профіль нового учасника команди
Клієнт бізнес-процесу	Користувач

На рисунку 1.8 представлена схема функцій процесу перегляду статистики вибраних змагань спортивного центру для тренувань чемпіонів.

Рисунок 1.8 – Діаграма бізнес-процесу перегляду статистики змагання

Характеристику бізнес-процесу перегляду статистики змагання наведено в таблиці 1.5.

Таблиця 1.5 – Характеристика бізнес-процесу перегляду статистики змагання

Назва характеристики	**Значення характеристики**
Ім’я бізнес-процесу	Перегляд статистики змагання
Основні учасники	Користувач, адміністратор,тренер
Вхідна подія	Запит на здійснення перегляду
Вхідні документи	Запит на перегляд
Вихідна подія	Відображення статистики по вибраному змаганню
Вихідні документи	Статистика по змаганню
Клієнт бізнес-процесу	Користувач

Процес автоматизації роботи центру підготовки чемпіонів «Золотий Дракон» сприяє підтриманню бази даних в актуальному стані і отриманню статистично достовірної інформації. Це має значно підвищувати швидкість створення та доступ до інформації щодо графіку проведення змагань та створення сітки поєдинків і дає можливість постійного оперативного впливу на процес управління.

Веб-платформа має зручний інтерфейс, що надає змогу легко та швидко працювати в її середовищі. Основним користувачем системи є тренери та президент спортивного центру, які відповідальні за створення графіку. Всі вище представлені функції позбавляють працівників спортивного центру роботи по заповненню документів, плутанини в паперовій документації та помилок.

1.3 Огляд та аналіз існуючих програмних засобів, що реалізують функції предметної області

1.3.1 Аналіз функціоналу та інтерфейсу існуючих програмних засобів

На сьогодні є достатньо веб-систем, за допомогою яких можна здійснити планування будь-яких подій. Проте систем за допомогою яких можна здійснити планування спортивних поєдинків майже немає [5].

Першим, розглянуто веб-сайт «Федерація Тхеквондо ВТФ України», який можна завантажити за посилання https://ukr.tkdo.events/tournaments/. Головна сторінка цього сайту представлена на рисунку 1.9.

Ця програма має дуже простий та зручний інтерфейс, завдяки якому користувач може легко переходити в наступних кроках. Для перегляду учасників конкурсу та перегляду результатів конкурсу користувач повинен зареєструватися. Реєстрація також доступна лише для тренерів. Тому, щоб використовувати додаткові функції, потрібно зареєструватися як тренер.

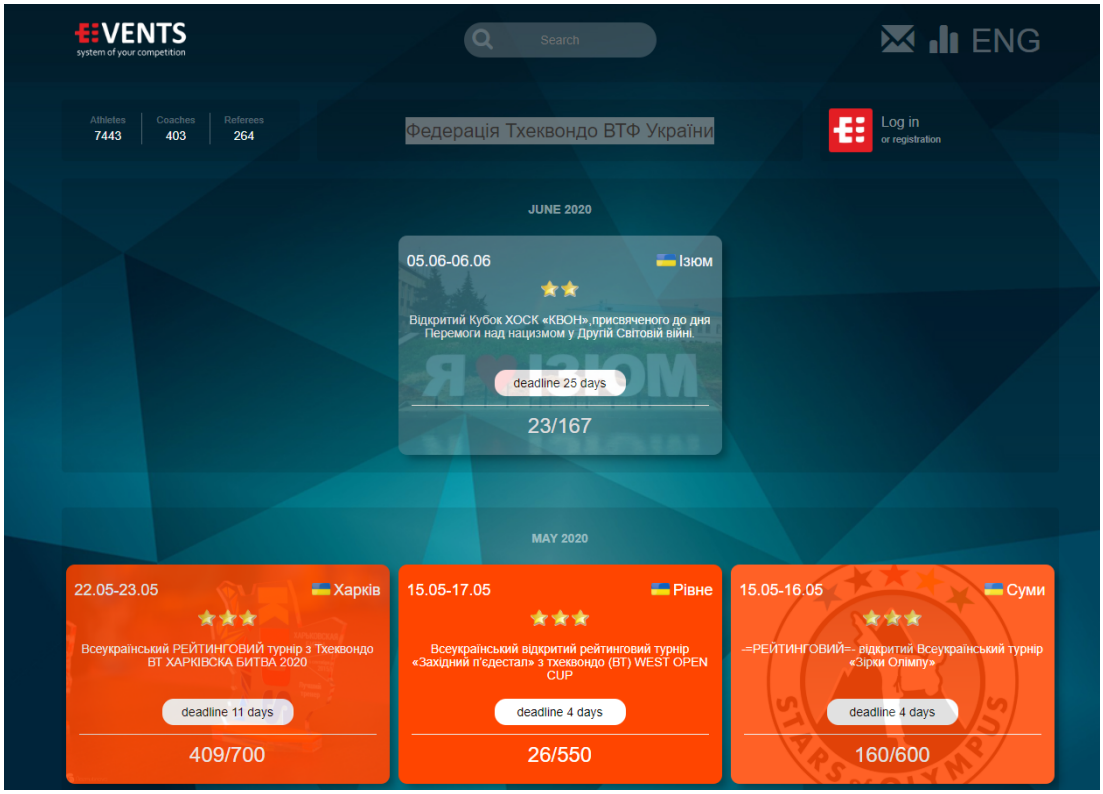

Рисунок 1.9 – Головна сторінка сайту «Федерація Тхеквондо ВТФ України»

Дана програмна система має наступні функції:

- історія спортивних змагань;
- створення спортивних змагань;
- формування статистики перемог для кожної команди;
- можливість перегляду результатів змагань.

Всі ці функції також будуть доступні у розробленій нами веб-платформі. Перевагою розробленої веб-платформи над конкурентом буде автоматична генерація сітки жеребкування та автоматичне формування списку боїв, з можливістю друку.

Також, для порівняння, розглянемо сторінку «Учасники». Вона представлена на рисунку 1.10.

Рисунок 1.10 – Сторінка сайту «Учасники»

Сторінку списку команд наведено на рисунку 1.11., а сторінку статистики результатів змагань показано наведено на рисунку 1.12.

Хоч даний продукт і має велику кількість функцій, вони доступні лише зареєстрованому користувачеві.

Здійснивши аналіз альтернатив, створено таблицю, де відображається порівняльна характеристика сайту-аналогу.

Оглянувши та зробивши аналіз схожих сайтів можна дійти до висновку, що вище перелічені програмні системи є хорошими для організації спортивних змагань. Кожна програма функціонує досить добре. Але, після огляду функцій кожного програмного продукту, можна побачити, що обидві системи різняться і вони орієнтовані тільки на зареєстрованих користувачів.

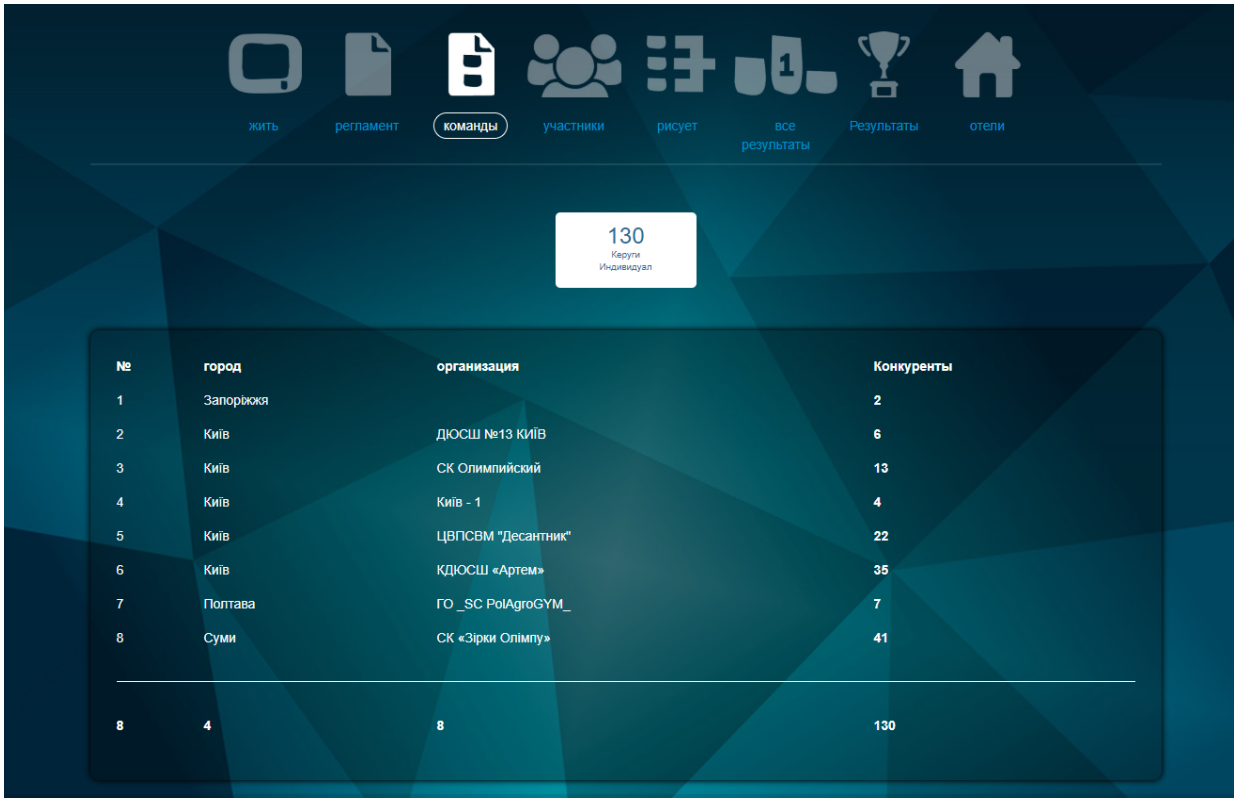

Рисунок 1.11 – Сторінка сайту «Команди»

Рисунок 1.12 – Сторінка сайту «Регламент»

Таблиця 1.6 – Порівняльна характеристика програмних продуктів

Назва параметра	**Програмний продукт**	
	Сайт організації змагань Федерація Тхеквондо ВТФ в Україні	**Сайт Центру підготовки чемпіонів «Золотий Дракон»**
Зручний, сучасний інтерфейс	–	+
Розмежування прав доступу	+	+
Формування списку проведення боїв	–	+
Результати змагання	+	+
Статистика по змаганню	–	+
Відображення новин, розкладу тощо	+	+
Можливість роботи у мережі	+	+
Безкоштовна версія програмного продукту	+	+
Пошук	+	+

Окрім цього функція створення сітки жеребкування доступна лише на створеній нами веб-платформі, а в представленому аналогу її немає. Отож веб-платформа, яка розроблена нами, є унікальною так як аналоги не виконують функцій генерації сітки жеребкування, тому є менш ефективними у організації спортивних змагань.

2 МОДЕЛЮВАННЯ ТА ФОРМУВАННЯ ВИМОГ ПРОГРАМНОГО ЗАСОБУ

2.1 Моделювання та аналіз програмного забезпечення

У таблиці 2.1. подано глосарій основних використовуваних термінів при створенні веб-платформи створення спортивних змагань.

Таблиця 2.1 – Глосарій основних термінів

Термін	**Опис терміну**
1. Основні поняття та категорії предметної області та проекту	
Спортивний центр	Спортивний центр для підготовки чемпіонів з тхеквондо
Тренер	Керівник спортивної секції в спортивному центрі
Секція	Періодичні відвідування спортивного центру групою під керівництвом тренера протягом певного періоду часу
Графік проведення боїв	Сукупність елементів, що становлять систематизацію боїв
Інтерфейс	Засіб зручної взаємодії користувача з інформаційною системою, що дозволяє взаємодіяти з системою з високою ефективність
2. Користувачі системи	
Тренер	Людина, яка використовує систему організації змагань, створює змагання, команди, додає або видаляє учасників змагань
Користувач	Людина, яка хоче переглянути список змагань, результати змагань або статистику команди
Адміністратор	Користувач системи, який здійснює формування сітки жеребкування, також має доступ до всіх функцій Тренера

Продовження таблиці 2.1

3. Вхідні та вихідні документи	
Графік проведення боїв	Документ, створений адміністратором, який вказує послідовність боїв з урахуванням вікових та вагових категорій.
База даних	Впорядкований набір логічно взаємопов'язаних даних, які поділяються та розроблені для задоволення інформаційних потреб користувачів

Діаграма варіантів використання є початковим концептуальним поданням або концептуальною моделлю системи в процесі її проектування і розробки [6,22]. Також, описує функціональне призначення системи або те, що система буде робити в процесі свого функціонування.

Безліч систем має поділ на категорії користувачів. Тому кожна категорія користувачів представляється окремою діючою особою (актором).

При аналізі роботи системи були виділені наступні дійові особи і розроблені варіанти використання:

- користувач – переглядає результати змагань, переглядає список змагань і список команд;
- тренер – реєструється на сайті, авторизується, якщо вже зареєстрований, створює змагання, команди, переглядає результати змагань;
- адміністратор – веде управління сайтом, вносить зміни в БД та на сайт, керує користувачами, створює графік проведення боїв.

Кожен варіант використання визначає послідовність дій, які повинні бути виконані спроектованою системою при взаємодії з відповідним актором. Діаграма варіантів використання може доповнюватися пояснювальним текстом, який розкриває зміст або семантику її компонентів. Такий пояснювальний текст має назву примітки або сценарій.

Для даної веб-системи, виходячи з потреб дійових осіб, можна виділити наступні варіанти використання:

- авторизація;
- реєстрація;
- адміністрування ресурсу і БД;
- створення змагання;
- редагування змагання;
- генерація сітки жеребкування;
- перегляд результатів змагання;
- отримання статистики змагання;
- збереження результатів змагання;
- ztворення команди;
- редагування команди;
- додавання учасників змагання;
- редагування профілю учасника змагання;
- видалення учасника змагання.

На діаграмі варіантів використання показано взаємодії між усіма дійовими особами і варіантами використання. Діаграма повинна показувати, які дійові особи ініціюють варіанти використання, а також повинна відображати, коли дійові особи отримують інформацію від варіантів використання.

Основні взаємодії між діючими особами і варіантами використання задаються за допомогою зв’язку у вигляді простої стрілки. Напрямок стрілки показує, хто ініціює зв’язок (завжди дійова особа) і який варіант використання відправляє інформацію дійовій особі.

Діаграма прецедентів представлена на рисунку 2.1

Рисунок 2.1 – Діаграма прецедентів

Наступним кроком буде опис прецедентів. Варіанти використання представленні у таблицях 2.2-2.5. Варіант використання «Реєстрація» подано у таблиці 2.2.

Таблиця 2.2 –Опис варіанту використання «Реєстрація»

Контекст використання	**Реєстрація**
Дійові особи	Не зареєстрований користувач(тренер)
Передумова	Користувач знаходиться на сторінці реєстрації
Тригер	Відкрита сторінка для реєстрації
Сценарій	1.Заповнюємо поле «Введіть логін»; 2.Заповнюємо поле «Введіть пароль»; 3.Заповнюємо поле «Введіть пароль ще раз»; 4.Натискаємо кнопку «Продовжити» 5. Заповнити інформацію «Персональні дані» 6. Заповнити інформацію «Організація"
Пост-умова	Користувач зареєстрований

Варіант використання «Авторизація» подано у таблиці 2.3.

Таблиця 2.3 – Опис варіанту використання «Авторизація»

Контекст використання	**Авторизація**
Дійові особи	Не авторизований тренер, адміністратор
Передумова	Користувач знаходиться на сторінці авторизації
Тригер	Відкрита сторінка для авторизації
Сценарій	1.Заповнюємо поле «Введіть логін»; 2.Заповнюємо поле «Введіть пароль»; 3.Натискаємо кнопку «Увійти» 4. Увійти за допомогою соц.мереж Facebook або Instagram
Пост-умова	Тренер, адміністратор авторизований

Варіант використання «Перегляд результатів змагання» подано у таблиці 2.4.

Таблиця 2.4 – Опис варіанту використання «Перегляд результатів змагання»

Контекст використання	**Перегляд результатів змагання**
Дійові особи	Користувач, адміністратор, гість
Передумова	Користувач знаходиться на головній сторінці
Тригер	Відкрита головна сторінка
Сценарій	1.Клік на зображення змагання 2. Перейти на вкладку «Результати»
Пост-умова	Відкрита головна сторінка

Варіант використання «Створення команди» подано у таблиці 2.5.

Таблиця 2.5 – Опис варіанту використання «Створення команди»

Контекст використання	**Внесення студентських даних**
Дійові особи	Тренер
Передумова	Авторизація Тренера
Тригер	Відкрита сторінка команди
Сценарій	1. Відкриваємо профіль тренера; 2. Натискаємо кнопку “Додати учасника»; 3. Додаємо необхідного учасника; 4. Вносимо дані про учасника.
Пост-умова	Збереження даних в БД

Для представлення варіантів використання було здійснено опис таблиць баз даних системи для показу основної функціональності системи та вимоги до неї.

Рисунок 2.2 – Таблиця даних «Змагання»

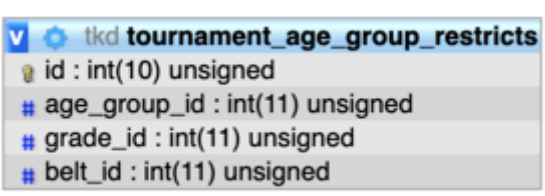

Рисунок 2.3 – Таблиця даних «Вікова група»

Рисунок 2.4 – Таблиця, що визначає зв’язок між учасником і змаганням

У таблиці зображено зв’язок між змаганнями, атлетом, віковою групою та ваговою групою. Адже спортсмен кожного разу може брати участь у інший віковій та ваговій групі, так як він стає старшим та важчим.

Рисунок 2.5 – Таблиця генерації сітки жеребкування

У таблиці показано зв’язок між змаганнями, днем змагань, рингом, віковою групою, ваговою групою та порядком (базуючись на певному дні, рингу, віку та вазі).

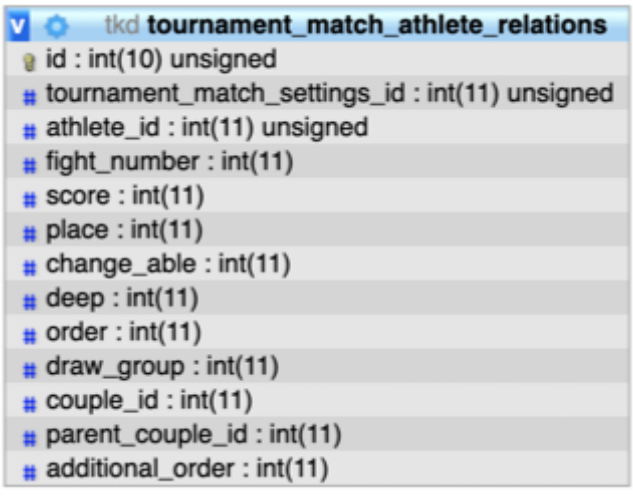

Рисунок 2.6 – Таблиця для відображення сітки жеребкування

Таблиця, яка використовується для збереження та відображення сітки:

- зв’язок між налаштуванням змагань (описано вище)
- атлетом
- номер поєдинку
- кількістю набраних балів
- місцем, яке посів спортсмен

Отже, ми навели діаграму варіантів використання для акторів системи та здійснили їх розкадровку для наочного показу функцій, які реалізовано в бекенд частині веб-платформі.

2.2 Архітектура програмного забезпечення

2.2.1 Опис архітектури програмного забезпечення

Для розробки додатку було обрано трирівневу архітектуру системи або модель прикладного сервера (AS – Application Server). У цій моделі реалізовано трирівневу систему розподілу функцій (Рисунок 2.7) [7,8].

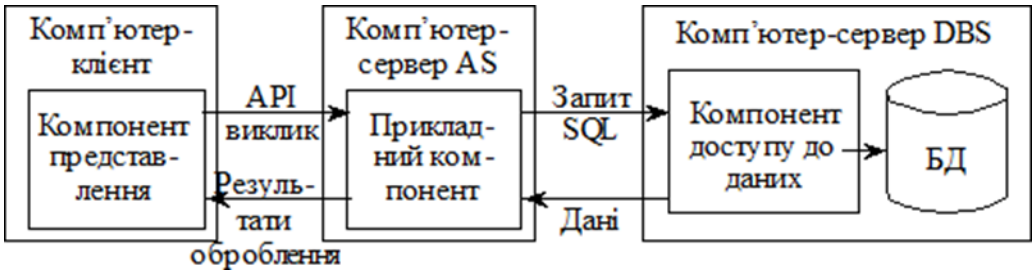

Рисунок 2.7 – Модель трирівневої архітектури

Першим рівнем є комп’ютер клієнта, де розміщений користувацький інтерфейс та функції локального редагування та логіка для перевірки даних. Також системі необхідно взаємодіяти з мережею. Тобто на комп’ютері-клієнті виконуються функції першої групи, що відповідають за інтерфейс із

користувачем. Звертаючись до компонента, що розміщений на сервері, комп’ютер виступає клієнтом додатку.

Другий рівень – це сервер додатків, який є ознакою трирівневої архітектури клієнт-сервер. Основне його призначення – зберігання та виконання бізнес-правил. Він реалізований як група процедур, які виконують прикладні функції та називається прикладним сервером або Application Server. Розділення прикладної логіки у інший архітектурний рівень дозволяє реалізувати її поширеними мовами програмування (C, C++, C#, Cobol, php), які мають великі переваги з мовами роботи із БД, оскільки вони – достатньо спеціалізовані. Розмежування застосованої логіки збільшує незалежність функціональних компонентів одного рівня від змін або вдосконалень компонентів іншого [9,11].

Третій рівень – сервер бази даних. Він відповідає за зберігання та підтримку даних, включаючи також їх узгоджене перетворення, попередження несанкціонованого чи некоректного коригування БД, створення резервних копій. Тобто він забезпечує зберігання інформації в БД.

Для опису бізнес логіки і наочного представлення бекенд компонентів веб-платформи використано UML-діаграму класів (Рисунок 2.8)

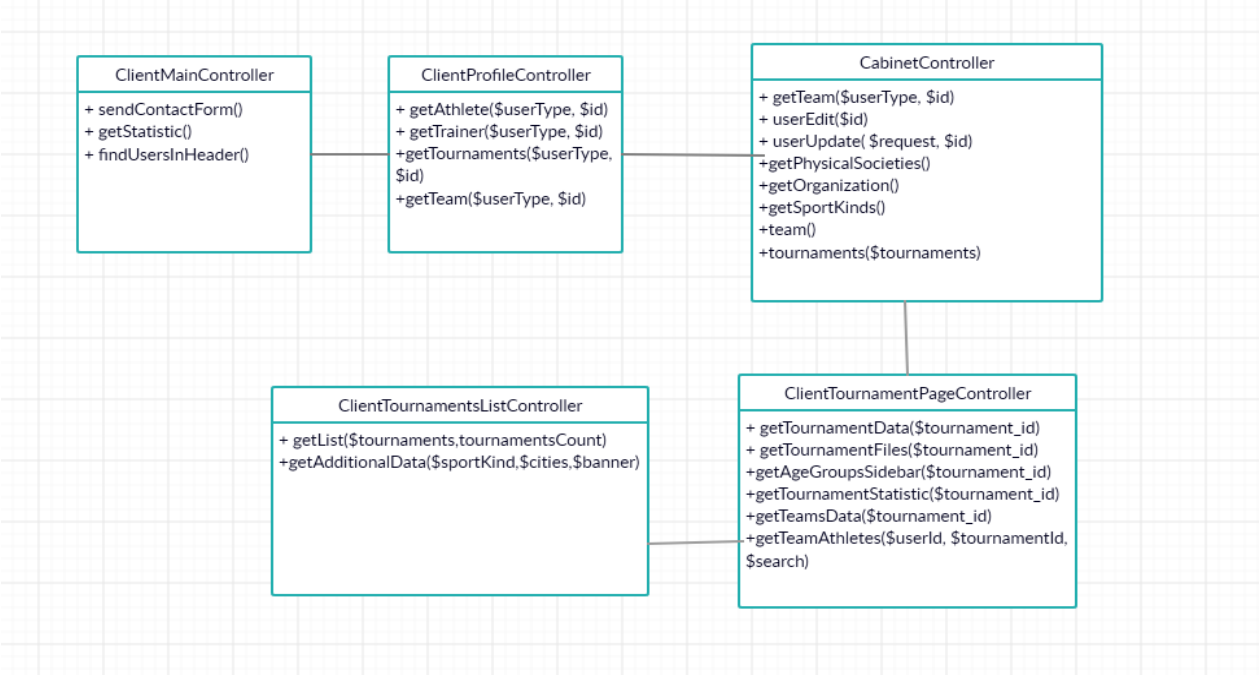

Рисунок 2.8 – Діаграма класів бекенд частини веб-платформи

На даній діаграмі представлені прототипи основних класів бекенд частини веб-платформи. Описано функції додавання, редагування, видалення, змагання та внесення нових учасників змагання в команди.

Функціональну структуру системи можна розділити на наступні модулі:

модуль обробки інформації про тренерів:

- реєстрація;
- авторизація.

модуль обробки інформації про змагання:

- додавання змагання;
- додавання команди;
- додавання учасника змагання від команди;
- редагування команди;
- редагування профілю учасника;
- формування графіку проведення боїв;
- перегляд результатів змагання.

Для того,щоб краще зрозуміти процес функціонування системи, побудовано діаграми активності для кожного модуля.

Діаграму активності авторизації тренера зображено на рисунку.2.9.

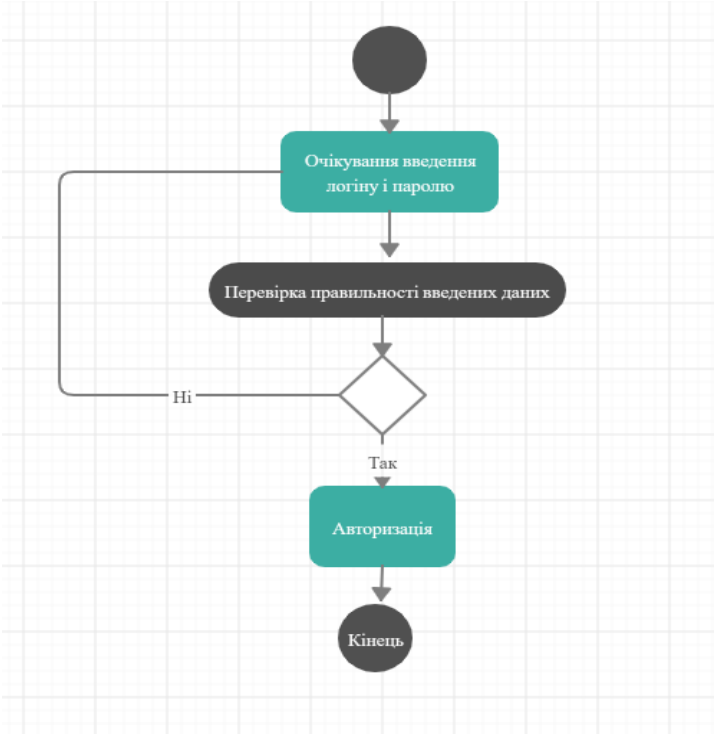

Рисунок 2.9 – Діаграма активності авторизації тренера

Діаграму активності реєстрації тренера зображено на рисунку 2.10.

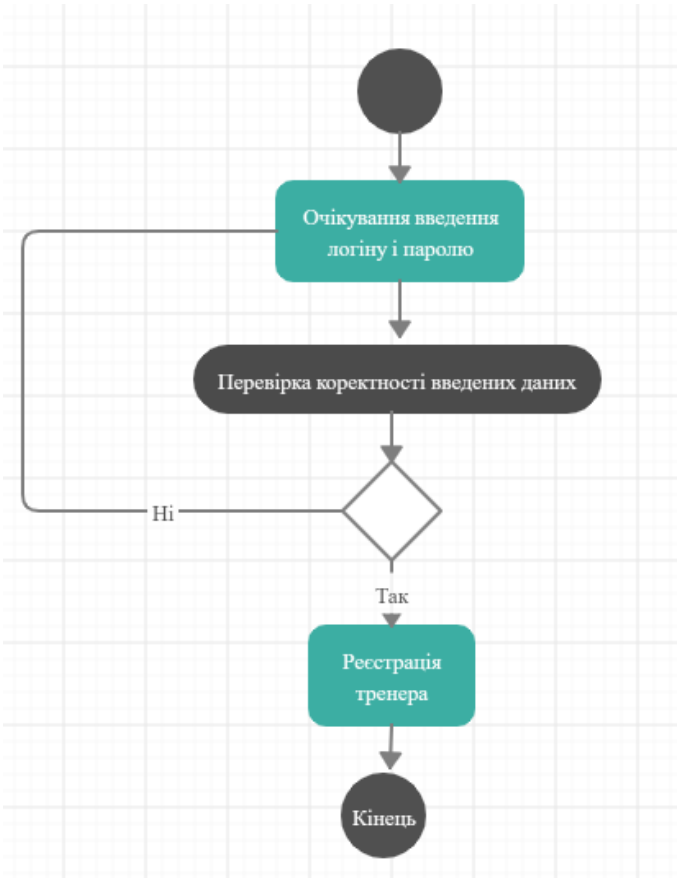

Рисунок 2.10 – Діаграма активності реєстрації тренера

Діаграму активності для модуля обробки інформації про змагання на рисунку 2.11.

Рисунок 2.11 – Діаграма активності для модуля обробки інформації про змагання

2.3 Специфікація функціональних та нефункціональних вимог

Провівши розкадровку, можна виокремити основні функціональні вимоги системи, які представлений у таблиці 2.6., та нефункціональні – у таблиці 2.7.

Таблиця 2.6 – Специфікація функціональних вимог

Ідентифікатори вимоги	**Назва вимоги**	**Атрибут вимог**		
		Пріоритет	**Складність**	**Контакт**
1	Авторизація	Обов’язкове	Середня	Адміністратор Тренер
2	Перегляд Результатів змагань	Обов’язкове	Середня	Адміністратор Користувач Тренер
3	Реєстрація	Обов’язкове	Середня	Адміністратор Тренер
4	Формування списку боїв	Обов’язкове	Висока	Адміністратор
5	Внесення даних нового змагання	Обов’язкове	Висока	Адміністратор
6	Додавання учасника змагання	Обов’язкове	Середня	Адміністратор Тренер
7	Редагування змагання	Рекомендоване	Складна	Адміністратор
8	Редагування профілю учасника змагань	Рекомендоване	Середня	Адміністратор Тренер

Таблиця 2.7 – Специфікація нефункціональних вимог

Ідентифікатор вимоги	Назва вимоги	Атрибути вимог		
		Пріоритет	Складність	Контакт
1. Застосовність				
1.1	Основні вимоги застосовності нової системи відносно інших систем, які знають користувачі	Рекомендована	Низька	Адміністратор, Тренер
1.2	Вимоги по відповідальност і стандартам графічного інтерфейсу користувача	Обов’язкова	Низька	Адміністратор, Тренер, користувач
1.3	Час, необхідний для навчання звичайних і досвідчених користувачів	Рекомендована	Середня	Адміністратор, Тренер
2. Надійність				
2.1	Доступність	Обов’язкова	Середня	Адміністратор, користувач
2.2	Середній час безвідмовної роботи	Обов’язкова	Середня	Адміністратор, Тренер
2.3	Точність	Обов’язкова	Середня	Адміністратор Тренер
3. Робочі характеристики				
3.1	Використання ресурсів	Рекомендована	Середня	Адміністратор Тренер
4. Проектні обмеження				
4.1	Вимоги до технології програмування	Рекомендована	Середня	Адміністратор Тренер

Значення нефункціональних вимог:

- необхідний час для навчання постійних користувачів – 30 хвилин;
- необхідний час для навчання досвідчених користувачів – 15 хвилин;
- основні вимоги щодо застосування нової системи для інших відомих користувачам систем
- всі функції системи легко виконувати, а структура програми не відрізняється від існуючих аналогів;
- вимоги щодо відповідності загальним стандартам застосовності та стандартам графічного інтерфейсу користувача – програма повинна коректно працювати у всіх браузерах останніх версій;
- доступність – час, що був витрачений на обслуговування системи, повинен не перевищувати 3% від загального часу роботи програми;
- середній час роботи – 3 години.

Ресурси, які використовуються – системні вимоги [11]:

- об’єм HDD: > 20 Мб;
- об’єм RAM: > 512 Мб;
- відео пам’ять: > 128 Мб;
- процесор: > 1 ГГц;
- клавіатура, маніпулятор миша;
- доступ до інтернет;
- веб браузер.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ БЕКЕНД ЧАСТИНИ

3.1 Програмна реалізація бекенд частини веб-платформи

Для написання оптимізованого та безпомилкового коду розробнику веб-ресурсу необхідно вибрати відповідне програмне забезпечення, яке розділяється на дві категорії: текстові редактори та інтегровані середовища розробки. Текстовий редактор – це комп'ютерна програма, яка призначена для створення та редагування текстових даних. До популярних текстових редакторів відносяться: Sublime Text 3, Visual Studio Code, Atom, Brackets, Notepad++, Vim, TextMate [10,20].

Інтегроване середовище розробки (IDE) – це комплексна комп'ютерна програма для розробки програмного забезпечення, що включає в себе наступні складові: текстовий редактор, компілятор чи інтерпретатор, а також інструменти для автоматизації процесів написання та відлагодження програм. До популярних IDE відносяться: Jetbrains Webstorm, Adobe Dreamweaver, Microsoft Visual Studio, NetBeans, IntelliJ IDEA. Вірно підібране програмне забезпечення підвищує ефективність написання та редагування коду, а також запобігає уникненню різноманітних помилок.

Сьогодні розробники мають широкий вибір того, якою мовою чи технологією користуватися для створення сайту. Для реалізації бекенд частини бакалаврської роботи було вибрано мову програмування PHP та фреймворк Laravel.

Ієрархічний поділ процесу розробки зумовлений складністю та багаторівневістю сучасних веб-ресурсів. Він складається із двох частин: клієнтська (front-end) та серверна (back-end) розробки. В даній роботі ми розгладаємо тільки бекенд частину розробленої платформи.

Back-end розробка – це набір апаратно-програмних засобів, за допомогою яких реалізується логіка роботи ресурсу. Тобто, це те, що приховано від очей

користувача і відбувається поза його браузером та комп’ютером. Яскравий приклад back-end розробки: коли користувач вводить запит на сторінці пошуковика та натискає клавішу Enter, робота front-end закінчується і починається back-end. Запит відправляється на сервер, де розташовані алгоритми пошуку. Сервер – це більш потужний комп’ютер, що виконує певні функції. Він зберігає дані та відповідає на запити користувачів. Щойно на моніторі з’являється інформація, яку шукав користувач, знову відбувається повернення в зону front-end. По суті, back-end – це процес об’єднання сервера з користувачем за допомогою баз даних, API та операційних систем [12].

Back-end розробник може застосовувати різноманітні інструменти, що доступні на сервері. Як правило, в його розпорядженні наявні розмаїті мови програмування, зокрема PHP, Java та Python. Також для back-end розробки використовуються різні системи управління базами даних: MySQL, MongoDB, Cassandra тощо. Самою поширеною комбінацією в back-end розробці є поєднання PHP з MySQL (Рисунок 3.1). Залежно від цілей, back-end розробник може виконувати наступні функції: забезпечувати кібербезпеку ресурсу та даних користувачів, створювати та інтегрувати бази даних, налаштовувати технології резервного копіювання та відновлення .

Технологія Personal Home Page (PHP) набула дуже широкого поширення завдяки своїй вільності та підтримці найпопулярніших платформ. Сторінки PHP виглядають як звичайні HTML-сторінки, на яких можна використовувати спеціальні теги типу <? Php> і <?>. Рядки програмного коду спеціальною мовою скриптів PHP вставляються між тегами [13,14].

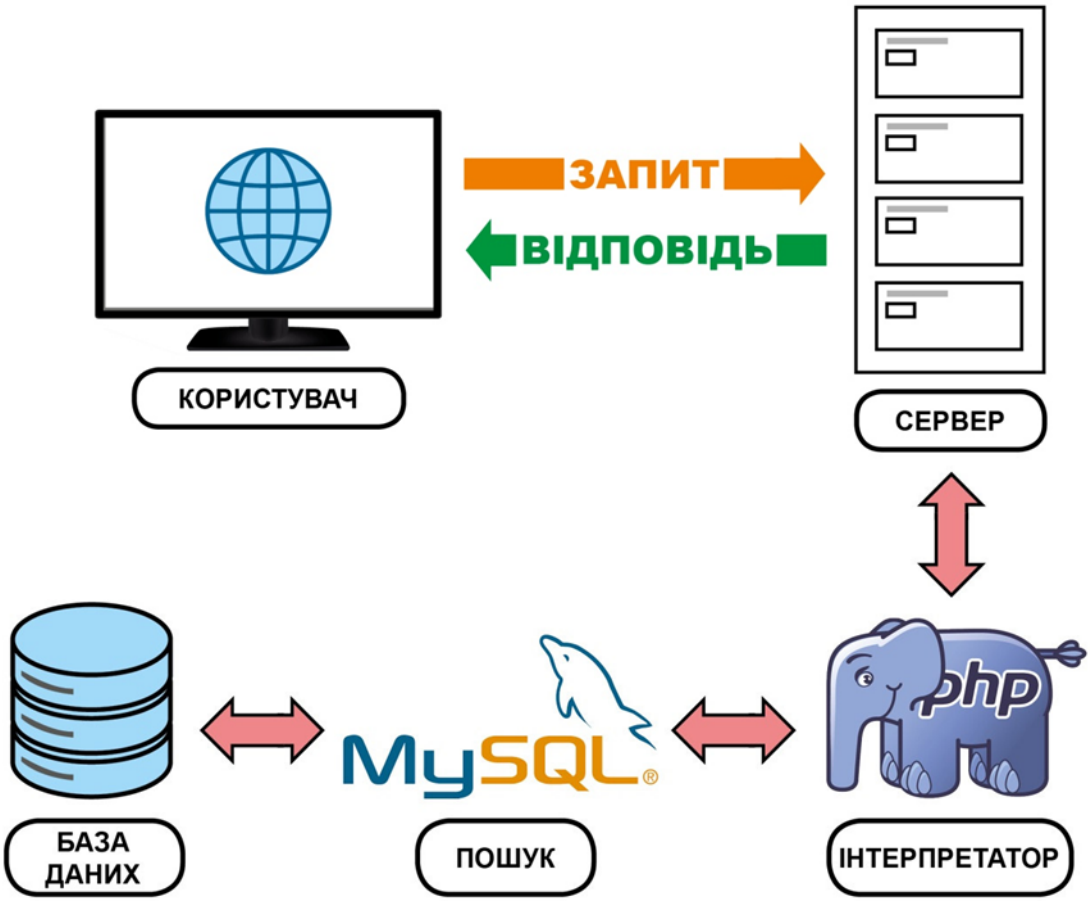

Рисунок 3.1 – Делегування відповідальності в back-end

Набагато швидше і без помилок, дозволив розробникам PHP писати програми, принцип роботи шаблонів, властивий традиційним CGI - програмам, що виводять вміст HTML у вихідний потік. Сьогодні діапазон систем на основі шаблонів коливається від простих зразків сторінок бази даних до великих програм електронної комерції на основі XML [17].

Розглянемо основні переваги та недоліки платформи:

- використовувана мова PHP проста та зручна, але не повністю орієнтована на об'єкти;

- для PHP є великі бібліотеки, а також багато вбудованих функцій для вирішення різноманітних проблем;

- PHP не має власних засобів масштабування, всі можливості кластеризації повністю доручені веб-серверу та розробникам;

- можливості інтеграції обмежені включенням модулів та використанням зовнішніх функцій, що не відповідає сучасним вимогам.

Фреймворк Laravel – це вільна PHP-програма з відкритим кодом з відкритим кодом, створена Тейлором Отвелл і розроблена для розробки веб-додатків відповідно до шаблону контролера перегляду моделі (MVC).

Деякі з особливостей Laravel – це модульна система упаковки із спеціалізованим менеджером залежності композитора, різні способи доступу до реляційних баз даних, утиліти, що допомагають при розгортанні та технічному обслуговуванні додатків, і його спрямованість на синтаксичний цукор [15].

Станом на березень 2019 року, Laravel рахується одним з найпопулярніших PHP фреймворком, разом з Nette, Symfony2, CodeIgniter, Yii2 й іншими фреймворками.

Сирий код Laravel'а розміщується на GitHub і надається під ліцензією MIT.

При виконанні дипломної роботи для створення веб-ресурсу кафедри доцільніше використовувати текстовий редактор, оскільки на відміну від IDE, він працює набагато швидше (відсутня велика кількість специфічні функції, що завантажують апаратне забезпечення), має логічний і простий інтерфейс, краще перевірений при роботі з невеликими проектами (10-20 файлів).

Під час вибору текстового редактору необхідно враховувати наступні основні критерії :

- підсвічування синтаксису (виділення початку та закінчення окремого логічного блоку певним кольором сприяє візуальному пошуку та робить код зручним для читання та корегування);
- вимоги до апаратного забезпечення (мінімальні системні вимоги для комфортної та швидкої роботи);

- автоматичні відступи (перенесення рядка програмою автоматичним виставленням рівня вкладеності для покращення орієнтації в структурі коду);
- автодоповнювання (аналіз програмою написаного код та пропозиція варіантів подальшого продовження);
- персоналізація інтерфейсу;
- розділення на робочі зони (можливість розділення робочої області редактора на необхідну кількість секторів);
- міні-мапа (відображення структури файлу та зручна навігація по документу);
- інтеграція додатків та модулів (покращення функціональності за допомогою додавання плагінів);
- гарячі клавіші (наявність певних комбінацій клавіш для того, швидко виконувати відповідні дії та можливість налаштування розробника).

Як середовище розробки було вибрано кросплатформовий редактор Sublime Text 3.

Він швидкий і багатий функціоналом, для практично кожної мови програмування. Підтримує декілька виділень, згортання коду, макроси, проекти та інше. Також можливо повноекранне редагування, яке виглядає чудово на великих моніторах. Запускається на Linux, Windows і OSX. Цей редактор надається з необмеженим тестовим періодом.

- концептуальні переваги редактора;
- швидка навігація (Goto Anything);
- множинні виділення (Multiple Selections);
- розподілене редагування (Split Editing);
- перемикання між проектами на льоту (Instant Project Switch);
- збірка програм (Build System);
- можливість розширення функціональності через Plugin API;
- кросплатформеність.

Особлива цінність програми – в можливості підключення плагінів і пакетів. Через модуль управління пакетами і плагінами Package Control можна знайти і додати (а також без проблем видалити, оновити, активувати або деактивувати) безліч необхідного в роботі – і перевірку орфографії, і розширення спектра даних, що відображаються, і елементи, що дозволяють більше "заточити" програму під розробку, і підгонку середовища під звичні стилі роботи (наприклад, робота з Emacs) і під звичні користувачу набори гарячих клавіш, додавання фрагментів для роботи з HTML5 або грамотної роботи з HTML-тегами. [20].

Проаналізувавши найпопулярніші СУБД для розробки бази даних було обрано СУБД MySQL, оскільки вона безкоштовна, відносно швидка і проста у використанні [16].

СУБД MySQL – є однією з надійніших, найшвидших та найвідоміших зі всього сімейства існуючих СУБД. Причиною є правила її розповсюдження, оскільки вона є безкоштовна і розповсюджується разом зі своїми початковими текстами. Іншою причиною є те, що MySQL доволі швидка СУБД. Postgresql, наприклад, також розповсюджується під подібною ліцензією, але вона не набула такого широкого поширення. Одна з причин – це помітна повільність. Отже, ціна і продуктивність є двома головними причинами популярності MYSQL.

MySQL написаний для близько десяти видів операційних систем. Це і FREEBSD, OPENBSD, MACOS, OS/2, SUNOS, WinXP і Linux. Сьогодні

MySQL особливо поширена на платформах Linux і Windows. Причому на останніх зустрічається набагато рідше.

Принцип роботи бази даних MySQL аналогічний принципу роботи будь-якої бази даних, яка використовує SQL (Structured Query Language) як мову команди для створення та видалення баз даних, таблиць, поповнення таблиць даними, вибірки даних.

Характеристики СУБД MySQL:

- написаний на мовах C і C++;
- протестований на широкому спектрі різних компіляторів;
- працює на безлічі різних платформ;
- для забезпечення переносимості використовує інструменти GNU;
- доступні API-інтерфейси для C, C++, PHP, Python , Java, Ruby;
- повністю багато потоковий з використанням потоків ядра. Може працювати в багатопроцесорних системах;
- забезпечує транзакційний і нетранзакційний механізми зберігання;
- використовує дуже швидкі дискові таблиці (MYLSAM) із стисненням індексів;
- дуже швидка система розподілу пам’яті, заснована на потоках;
- код MySQL протестований за допомогою інструментів пошуку витоку пам’яті;
- доступна як окрема програма, яку можна використовувати в клієнт-серверному середовищу. Мережева зв’язність СУБД MySQL:

клієнти можуть підключатися до сервера MYSQL, за допомогою TCP / IP-розеток на будь-якій платформі. У системах Windows клієнти можуть підключатися за допомогою названих каналів.

- інтерфейс ODBC / CONNECTOR дозволяє MySQL підтримувати програми, які використовують з'єднання ODBC. Наприклад, ви можете використовувати MS Access для підключення до сервера MySQL. Клієнтське програмне забезпечення може працювати в Windows або UNIX. Доступні початкові тексти інтерфейсу CONNECTOR / ODBC. Всі функції ODBC підтримуються, як і багато інших.інтерфейс CONNECTOR/JDBC дозволяє MYSQL взаємодіяти з клієнтськими програмами на Java, в яких використовуються JDBC- підключення.
- клієнтське програмне забезпечення може працювати в Windows або UNIX.

Доступні початкові тексти інтерфейсу CONNECTOR / JDBC.Для програмування для PHP на сервері реалізована програмна сторінка для створення баз даних та таблиць phpMyAdmin. Phpmyadmin – це інструмент, написаний на PHP, який дозволяє адмініструвати базу даних MYSQL. В реалізованому інструменті можливо запускати команди SQL, працювати з полями (додавати, редагувати, видаляти), працювати з таблицями (створювати, змінювати), створювати додаткові бази даних, і багато що інше.

Розглянемо детальні основні класи та функції бекенд частини веб-платформи.

Існує 5 основних класів ClientMainController, ClientProfileController, ClientTournamentPageController,ClientTournamentListController,CabinetController, тобто класів в яких немає полів, лише методи. Детальніше ці класи описані на рисунках 3.2-3.6.

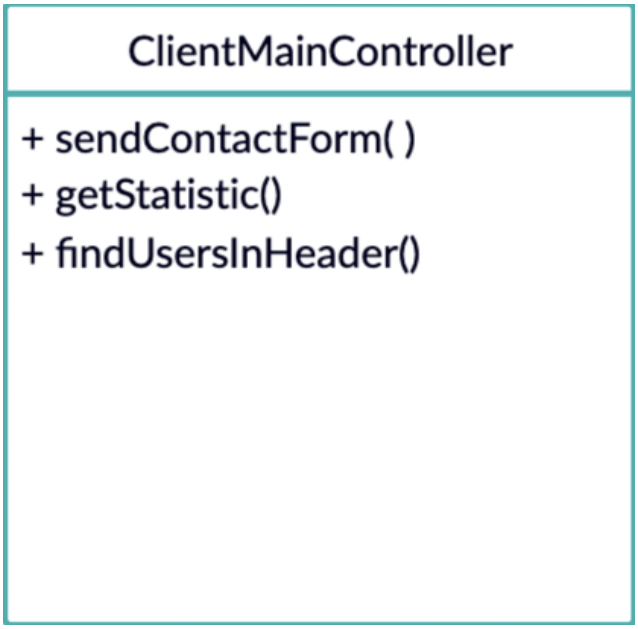

Рисунок 3.2 – Клас ClientMainController

Даний клас використовується для різних сторінок та містить загальні методи:

- sendContactForm() – використовується для відправлення контактної форми;
- getStatistic() – використовується для відображення статистики на головній сторінці;
- findUsersInHeader() – використовується для пошуку спортсменів у шапці сайту

Контролер ClientProfileController використовується для повернення інформації на сторінку профілю (адміністратор або тренер).

- getAthlete($userType, $id) – приймає тип та ідентифікатор та повертає інформацію про профіль атлета, яку ми бачимо на сторінці;
- getTrainer($userType, $id) – приймає тип та ідентифікатор та повертає інформацію про профіль тренера, яку ми бачимо на сторінці;
- getTournaments($userType, $id) – приймає тип та ідентифікатор та повертає щоденник останніх змагань на профілі атлета;
- getTeam($userType, $id) – приймає тип та ідентифікатор та повертає учасників команди тренера на профілі тренера.

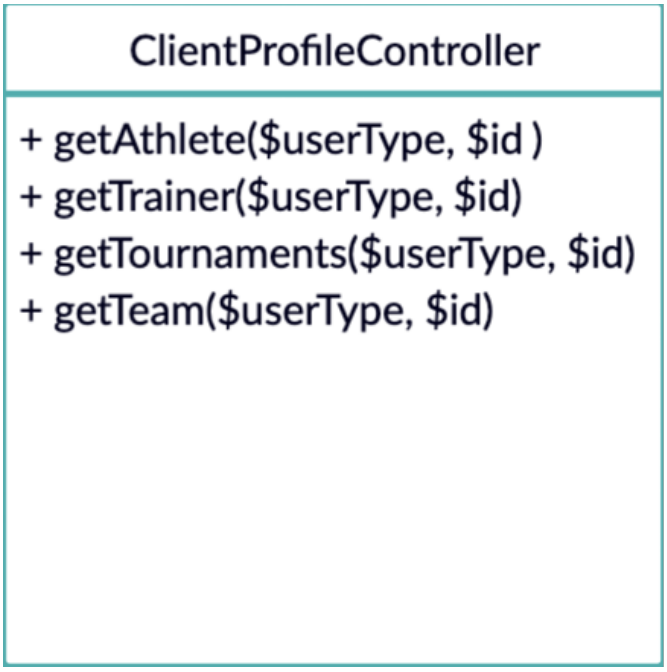

Рисунок 3.3 – Клас ClientProfileController

Рисунок 3.4 – Клас ClientTournamentPageController

Контролер ClientTournamentPageController опрацьовує всі запити на сторінці змагань.

- getTournamentData($tournament_id) – приймає тип та ідентифікатор та повертає дані про змагання;
- getTournamentFiles($tournament_id);
- getAgeGroupsSidebar($tournament_id);
- getTournamentStatistic($tournament_id) – приймає тип та ідентифікатор та статистику по змаганню;
- getTeamsData($tournament_id) – приймає тип та ідентифікатор та повертає дані про учасників команди;
- getTeamAthletes($userId, $tournamentId, $search).

Рисунок 3.5 – Клас ClientTournamentListController

- getList($tournament_id) – приймає ідентифікатор змагання та пошукові параметри (можна на сторінці всі взяти). Робимо запит до бази даних та отримуємо відповідні змагання, які відповідають всім переданим критерієм;
- getAdditionalData($tournament_id) – метод використовується для отримання даних фільтрів. Створений для того, щоб відправити з фронтенду запит синхронно з getList() і відповідно повернути користувачу сторінку швидше. Даний метод повертає наступні дані: види спорту, міста, фото банера.

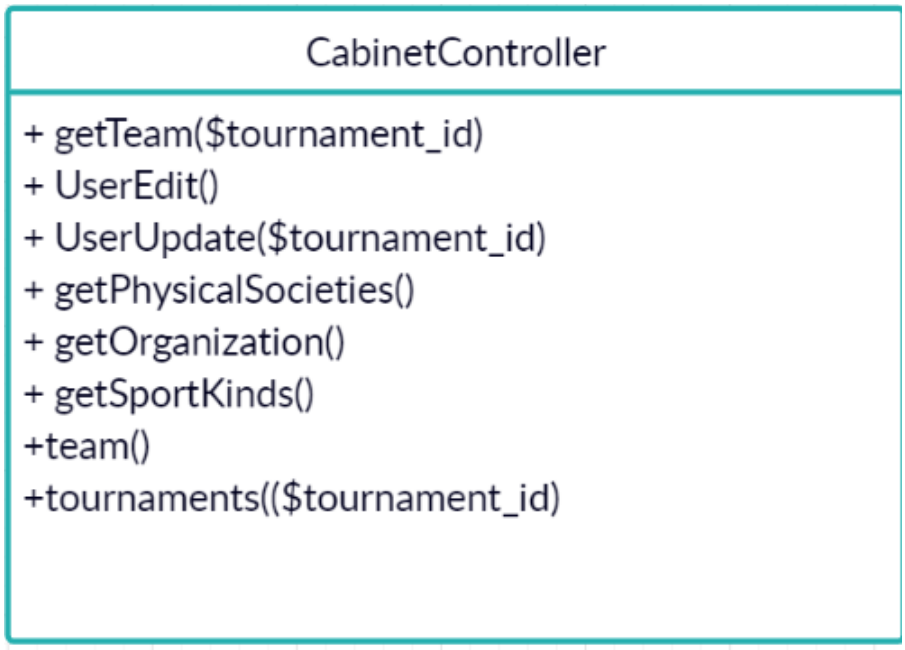

Рисунок 3.6 – Клас CabinetController

- getTeam($tournament_id) – приймає ідентифікатор тренера/користувача та пошуковий параметр - ім’я спортсмена. Робить запит до бази даних та отримує список спортсменів, які відповідають всім переданим критерієм;
- userEdit() – використовується для завантаження сторінки для редагування тренера;
- userUpdate() – використовується для обробки та збереження даних при редагування тренера (свого профілю). Отримує та обробляє відповідно всі дані, які ми бачимо на фронтенді;
- getphysicalSocieties() – отримуємо список всіх видів ФСТ для селекту, який відображається на сторінці редагування тренера/профілю;
- getOrganization() – отримуємо список всіх видів організацій для селекту, який відображається на сторінці редагування тренера/профілю.
- getSportKinds() – отримуємо список всіх видів спорту для селекту, який відображається на сторінці редагування тренера/профілю.
- team() - використовується для відображення сторінки команди.

- tournaments() – використовується для відображення та отримання даних для сторінки змагань. А також, отримуємо міста окремим запитом, щоб відобразити у списку.

Однією з важливих функцій веб-платформи є можливість автоматично згенерувати сітку жеребкування та створити список боїв. Для цього було розроблено клас SetMatchAthleteRelation() з використанням мови програмування PHP. index() – метод є початковий у записку жеребкування. Метод містить три цикли. Спочатку, ми отримуємо всі змагання, після цього для кожного змагання отримуємо кількість днів.

Після, ми отримуємо для кожного дня – кількість рингів, які будуть доступні під час дня. І тоді ми отримуємо спортсмени, які приєднані до кожного з рингу і передаємо їх до алгоритму жеребкування. Саме, така послідовність потрібна для того, щоб дотримуватися правильної нумерації поєдинків. Клас SetMatchAthleteRelation() описано в наступному фрагменті програми:

```
    public function index($tournamentId = null, $weight_id = null)
     {
      $this->tournamentId = $tournamentId;
      $this->weightId = $weight_id;

      $this->deleteOldData();

      foreach ($this->getTournaments() as $this->tournament) {
       foreach ($this->getTournamentDays() as $this->day){
```

```
        foreach ($this->getTournamentRings() as $this->ring){
          $this->setMatchRelation();

          $this->deep = 0;
         }
        }
       }
      }
```

setAthletesCouple() – даний метод отримує список спортсменів, та обробляє їх. Результатом обробки є масив пар спортсменів в залежності від їхньої кількості (у випадку непарної). Опис методу setAthletesCouple() наведений нижче:

```
    private function setAthletesCouple()
     {
      if($this->deep == 0){
       $this->setAthletesCoupleFirst();
       return;
      }

      $dividedAthletes = [];
      $this->athletesCouple = [];

      foreach ($this->athletes as $key => $athlete) {
       $dividedAthletes
        [$athlete['weight_id']][$athlete['draw_group']][]
= $athlete;
```

```
        }

        foreach ($dividedAthletes as $weightKey =>
$weightDivide) {
         if(count($weightDivide) > 1){
          $check = 1;
          $counter = 1;
          $couple = 0;

          for($i = 0; $i < count($weightDivide); $i++){

if(count($weightDivide[array_keys($weightDivide)[$i]])  >
1)
            $check = 0;
          }

          if($check){
           foreach ($weightDivide as $key => $value) {
            if($couple == 0){
             $weightDivide[$key][0] = $value[0];
             $couple = $key;
            } else{
             $weightDivide[$couple][1] = $value[0];
             unset($weightDivide[$key]);
             $couple = 0;
            }

           }
          } else{
```

```
          foreach ($weightDivide as $key => $value) {
           if(count($value) == 1)
            unset($weightDivide[$key]);
          }
         }
        }

        foreach ($weightDivide as $drawNumberKey =>
$drawNumber) {
         $drawGroup = &$this->athletesCouple
          [$weightKey][$drawNumberKey];
         $couple = 1;

         foreach ($drawNumber as $key => $athlete){
          if(
           ($athlete['deep'] == 0 && $this->deep == 1)
           || ($athlete['deep'] >= 1 && $this->deep > 1)
          ){
           if($couple == 1){
            $drawGroup[$key][0] = $athlete;
            $couple = 0;
           } else{
            if($athlete['id']   ==   $drawNumber[$key   -
1]['id']){
             $drawGroup[$key - 1][1] = $athlete;
             $couple = 1;
            } else{
             $drawGroup[$key][0] = $athlete;
            }
```

```
            }
           } else{
            $drawGroup[$key][0] = $athlete;
            $drawGroup[$key][1] = $athlete;
           }
          }

          if(count($drawGroup) > 1){
           end($drawGroup);
           $key = key($drawGroup);

           if(count($drawGroup[$key]) == 1){
            $drawGroup[$key][] = $drawGroup[$key][0];
           }
          }
         }
        }
       }
```

divideSimilarAthletes – метод використовується для того, щоб розділити спортсменів з однієї команди у максимально далекі кінці. Щоб поєдинок відбувся тільки у фіналі, а не на початку. Метод divideSimilarAthletes() описаний в наступному фрагменті програми:

```
    private function divideSimilarAthletes($athletes)
{
 $athletesCount = [];
 $trainerAthletesGroup = [];
 $generalOrder = [];
 $maxCount = 0;
 $dividedAthletes = [];
```

```
  foreach ($athletes as $key => $athlete) {
   if(!array_key_exists($athlete['trainer_id'],
$athletesCount))
    $athletesCount[$athlete['trainer_id']] = 0;

   $athletesCount[$athlete['trainer_id']]++;
   $trainerAthletesGroup[$athlete['trainer_id']][]      =
$athlete;
  }

  arsort($athletesCount);

  foreach ($athletesCount as $key => $athleteCount) {
   $generalOrder = array_merge(
    $generalOrder,
    $trainerAthletesGroup[$key]
   );
  }

  $maxCount                                              =
$athletesCount[array_keys($athletesCount)[0]];

  for($i = 0; $i < $maxCount; $i++){
   for($j  =  $i;  $j  <  count($generalOrder);  $j  +=
$maxCount){
    $dividedAthletes[] = $generalOrder[$j];
   }
  }

  return $dividedAthletes;
 }
```

saveAthletesCoupleInDatabaseFirst() – збереження пар у базу даних та викликання функції для оброки наступних пар. Метод saveAthletesCoupleInDatabaseFirst() описаний нижче:

```
private function saveAthletesCoupleInDatabaseFirst()
 {
  foreach   ($this->athletesCouple   as   $groupKey   =>
$groupDivide) {
```

```
   foreach ($groupDivide as $weightKey => $weightDivide)
{
    foreach    ($weightDivide    as    $drawNumberKey    =>
$drawNumber) {
     $drawNumber = array_reverse($drawNumber);

     foreach ($drawNumber as $key => $couple) {
      $fight_number = $this->getFightNumberByRing();

      $coupleId = TournamentMatchAthleteRelation
       ::orderBy('id', 'DESC')
       ->first();

      $coupleId = $coupleId != null
       ? $coupleId->id
       : 0;

      $couple[0]['couple_id'] = $coupleId;
      $couple[1]['couple_id'] = $coupleId;

      if($couple[1]['athlete_id']      ==      0      ||
$couple[0]['deep'] == 1){
       $couple[0]['fight_number'] = 0;
       $couple[1]['fight_number'] = 0;
      } else{
       $couple[0]['fight_number'] = $fight_number;
       $couple[1]['fight_number'] = $fight_number;
      }

      TournamentMatchAthleteRelation::create($couple[0]);
      TournamentMatchAthleteRelation::create($couple[1]);
     }
    }
   }
  }

  $this->deep++;

  $this->setMatchRelation();
 }
```

3.2 Тестування розробленої системи

Тестуватися буде веб-платформа для організації спортивних змагань. Під тестування підпаде реєстрація користувачів, створення змагання, додавання учасника команди та формулювання статистики.

Враховуючи застосування системи та її рівень важливості, буде проводитися лише мануальне тестування функціоналу.

Нижче наведено таблицю з тесткейсами та результатом їх виконання.

Таблиця 3.1 – Звіт про мануальне тестування

№	Передумови	Дії	Результат	Очікуваний результат
1	2	3	4	5
1	Відкрита будь-яка сторінка без авторизації.	1. У рядок вводиться адрес веб-сайту та будь-який URN.	Відбувається переадресація на сторінку переадресації.	Відбувається переадресація на сторінку переадресації.
2	Відкрита сторінка реєстрації.	1. Введення даних реєстрації. 2. Клік на кнопку «зареєструватися».	Повідомлення про успішну реєстрацію.	Повідомлення про успішну реєстрацію та перехід до авторизації.
3	Відкрита сторінка авторизації.	1. Клік на кнопку входу. 2. Введення коректного логіну та паролю студента. 3. Клік на кнопку “авторизуватись”.	Переадресація на головну сторінку.	Переадресація на головну сторінку.

Продовження таблиці 3.1

1	2	3	4	5
4	Відкрита сторінка Авторизації	1.Клік на кнопку входу. 2.Введення коректного логіну та пустого паролю. 3.Клік на кнопку “авторизуватись”.	Оновлення сторінки авторизації без повідомлень	Повідомлення у невірному логіні або паролю
5	Увійти як тренер, відкрита сторінка створення змагань	1.Клік на кнопку створення нового змагання. 2.Введення даних про нове змагання 3.Клік на кнопку “створити змагання”.	Оновлення сторінки, у списку додається нове змагання	Оновлення сторінки, у списку змагань додається нове змагання
6	Увійти як тренер, відкрита сторінка змагань	1. Клік на кнопку «Більше інформації» по змаганню	Відображається вся статистика обраного змагання	Відображається вся статистика змагання
7	Увійти як користувач	1. На сторінці веб-сайту натиснути на змагання 2. Натиснути на кнопку «Більше інформації»	Відображається статистика з усіма результатами змагання	Відображається статистика з усіма результатами змагання

При запуску веб-ресурсу завантажується головна сторінка сайту (див. Рисунок 3.6.)

Рисунок 3.6. – Головна сторінка веб-платформи

Щоб увійти у систему під власним іменем, необхідно спершу зареєструватися, реєстрація доступна лише тренерам. Для цього необхідно натиснути «Увійти» в верхньому правому кутку вікна на панелі навігації. Після входження в систему, результати роботи користувача будуть зберігатися в базі даних.(рис 3.7.)

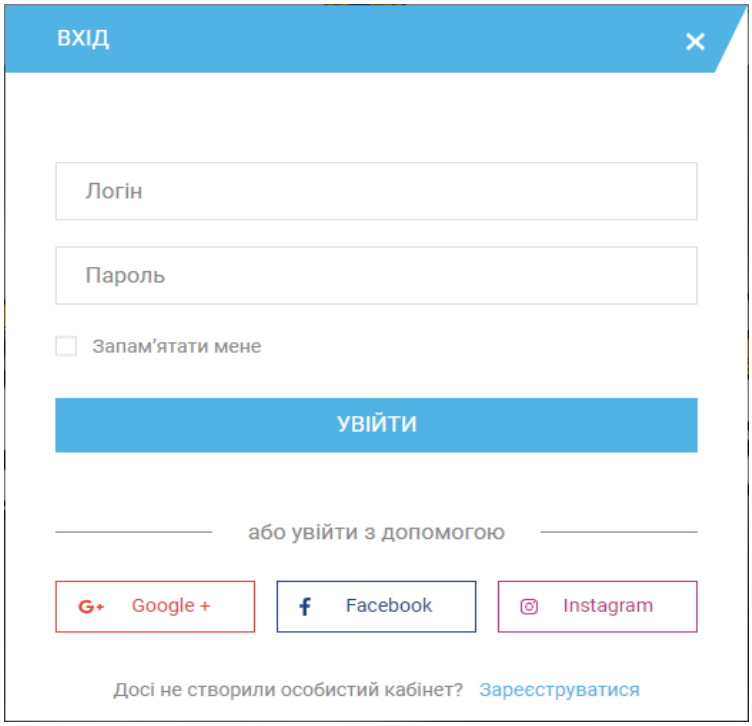

Рисунок 3.7. – Вхід в систему

Після входу в систему як тренер, будуть доступні функції додавання учасників змагання в команду, також видаляти учасника та редагувати профіль учасника.

Додавати нове змагання може тільки Адмін, тренеру ця функція недоступна.

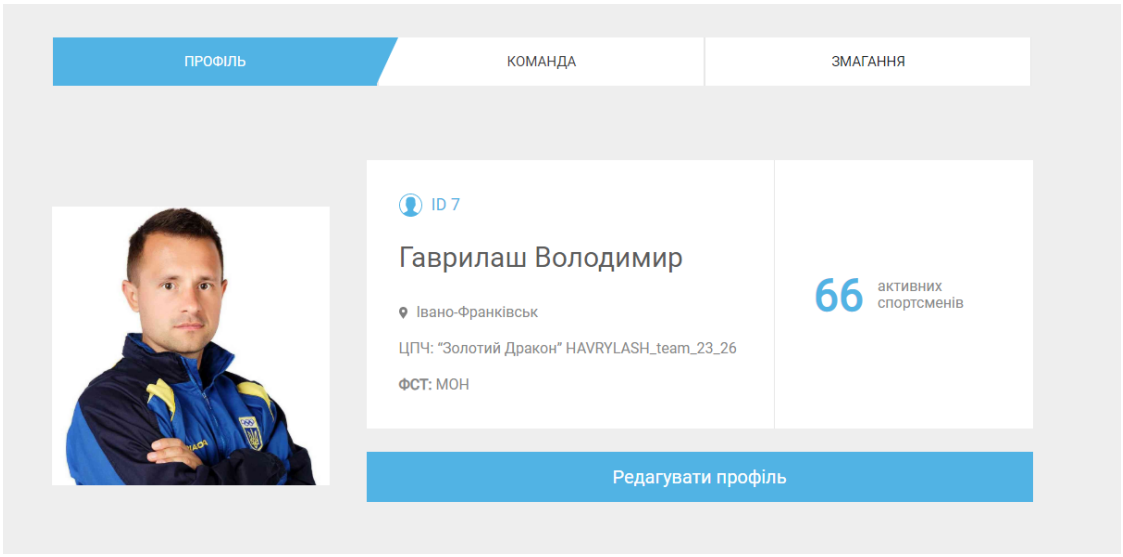

Рисунок 3.8. – Профіль тренера

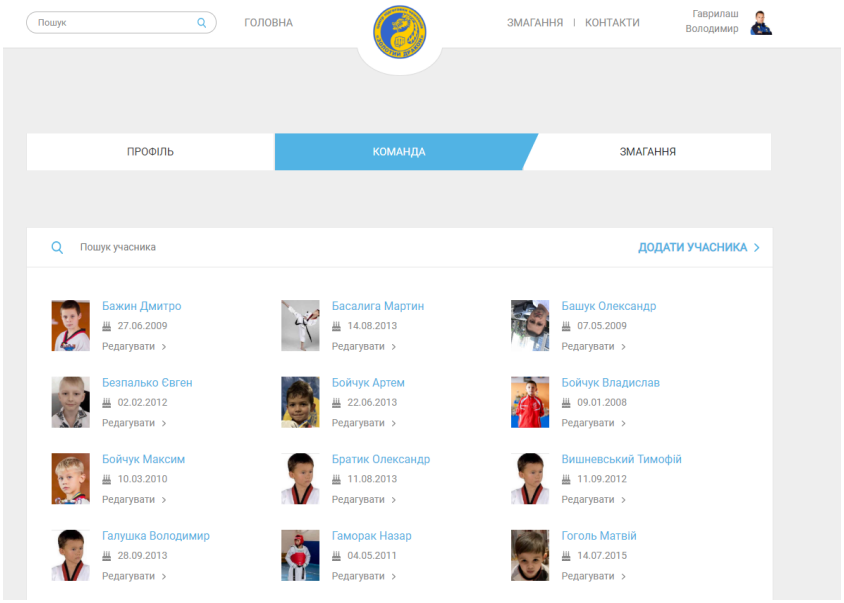

Рисунок 3.9. – Вкладка «Змагання» у профілі тренера

4 БІЗНЕС-ПЛАН РОЗРОБКИ ВЕБ-ПЛАТФОРМИ

4.1 Аналіз ринку збуту запуску проекту

Для початку проаналізуємо зміст ідеї веб-платформи, її можливі напрямки застосування, чим запропонована веб-платформа відрізняється від існуючих аналогів, а також основні вигоди, які може отримати користувач товару. Результати аналізу представлені у таблиці 4.1 [21].

Таблиця 4.1. Опис ідеї проекту

Зміст ідеї	**Напрямки застосування**	**Вигоди для користувача**
Автоматичне формування списку боїв та генерація сітки жеребкування	1. Веб-сайти спортивних організацій	Автоматична генерація списків боїв
	2. Проведення спортивних змагань	Автоматизація процесу жеребкування та створення списку боїв

Дослідження риинкових можливостей, які можна використати під час впровадження проекту на ринок, та потенційних ринкових загроз, які можуть зашкодити запуску проекту, дозволяє спланувати напрями розвитку проекту із урахуванням ситуації ринкового середовища, потреб потенційних користувачів та пропозицій платформ-конкурентів. Для цього спочатку проводиться аналіз попиту (таблиця 4.2).

Таблиця 4.2. Попередня характеристика потенційного ринку проекту

Показники стану ринку	**Характеристика**
Загальна потреба в продукції	Необхідна, але офіційних запитів на розробку немає через брак фінансування
Потенційні обсяги випуску в натуральних показниках за рік часу	До 100 копій
Вартість за одиницю продукції	350$
Річні обсяги у вартісних показниках	3000 – 5000$
Динаміка ринку (якісна оцінка)	Зростає
Наявність обмежень для входу	Доступу до правдивої статистичної інформації.
Показники стану ринку	Характеристика
Специфічні вимоги до стандартизації та сертифікації	Для ПЗ відсутні. Для коректної роботи - використання стандартів ISO 9126 та ISO 25010
Середня норма рентабельності в галузі (або по ринку)	87%

За попереднім оцінювання ринок не здається достатньо привабливим для входження. Але при проведенні аналізу поточного стану ринку виявлено підвищений інтерес до сфери фізичної культури.

Пошук ринкових можливостей, що можна використати під час виходу продукту на ринок, та потенційниї загроз, що можуть загрожувати реалізації проекту, дозволяє подумати про різні напрями розвитку проекту із урахуванням ситуації на ринку, потреб потенційних користувачів та пропозицій платформ-конкурентів.

В подальшому визначаються потенційні групи клієнтів та сторюється орієнтовний перелік вимог до товару для кожної групи (таблиця 4.3).

Таблиця 4.3. Характеристика потенційних клієнтів проекту

Потреба, що формує ринок	**Цільова аудиторія**	**Особливості поведінки споживачів**	**Вимоги споживачів до товару**
Автоматизація процесу проведення. спортивних змагань	Тренери спортивних центів	Розробники займаються написанням програм, які не завжди: відповідають стандартам (за стильовою характеристикою, наприклад), не завжди достатньо оптимізовані, що впливає на подальше життя створених програмних продуктів – виникають проблеми, недоліки та конфлікти. Тривале вирішення проблем несумісності або критичних помилок ПЗ.	– доступна ціна; – зручність простота використання; – мобільність
	Учасники спортивних змагань	Основною метою створити програмний продукт та випустити його на ринок, власники програмних розробок націлені на основні завдання такі, як: швидше створити ПП і якомога вигідніше продати (більше копій, вища ціна). Тому важливо мати й можливість контролю.	– зручність і простота використання

Таблиця 4.4. Фактори загроз

Фактор	**Зміст загрози**	**Можлива реакція компанії**
Поява конкурентів	Можлива поява конкурентів, які спроможуться створити більш якісний продукт. Можлива поява більш дешевих продуктів	Зменшення ціни з підвищенням якості при цьому, розробка удосконалень, розширення асортименту (додавання нових можливостей, нового функціоналу) збільшення точності розпізнавання, підвищення швидкості розпізнавання, зменшення тренувальної вибірки зображень
Зміни тенденцій ринку	Можлива ситуація, в якій з'явиться більш досконала програмна система від конкурентів, які значно довше на ринку.	Можлива така ситуація. Але можливості вирішення найпростіші - розробка нових сучасних необхідних удосконалень, тобто додання або заміна старого функціоналу на можливості розрахунку нових параметрів
Зниження репутації компанії	Можлива ситуація, коли конкуренти спроможуться на більший попит	Зміна партнерів, заключення нових контрактів, проведення рекламних та промо-акцій
Економічний спад	Відсутність попиту на товар компанії через економічну складову	Збільшення обсягів продажів, зменшення ціни; зміна цільової аудиторії

Таблиця 4.5. Фактори можливостей

Фактор	**Зміст загрози**	**Можлива реакція компанії**
Невелика кількість конкурентів	На ринку на сьогоднішній день дуже не значна кількість конкурентів, їх програмні продукти в переважній більшості вузько спеціалізовані	Розповсюджувати створений продукт, розвивати його можливості
Відповідні тенденцій ринку	ІТ-ринок на сьогоднішній день потребує, а відповідно і надає всі можливості для впровадження систем, які надаватимуть користувачам можливість розпізнавання інформаційних вкидань	Розповсюджувати створений продукт, розвивати його можливості
Можливість побудови власної репутації	Новий «гравець» на ринку має всі можливості для побудови власної репутації з «чистого листка»	Пошук замовників, можливих покупців створеного продукту, розширення бази замовників. Зарекомендувати себе, як надійну компанію. Можливо на вигідних умовах співпраці

Таблиця 4.6. Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	**У чому проявляється дана характеристика**	**Вплив на діяльність підприємства (можливі дії компанії)**
Тип конкуренції	Чиста Залежить від кількості конкурентів та якості надання ними послуг у порівнянні з послугами компанії	Покращення власного продукту через зниження ціни та підвищення якості
За рівнем конкурентної боротьби	Локальна Конкуренція на вітчизняному ринку	На вітчизняному ринку конкурентів не виявлено, а тому компанія має можливість встановлення власної бажаної ціни, та наробляти клієнтську базу. Перспектива – вихід на міжнародний рівень
За галузевою ознакою	Внутрішньогалузева Продукт націлений лише на конкретну сферу діяльності	Немає можливостей та сенсу розширювати функціонал за межі ІТ-сфери, але існує багато варіантів розвиватись всередині неї
Конкуренція за видами товарів	Марки-конкуренти	Зниження ціни, розширення функціональних,

Продовження таблиці 4.6

	Створений товар може мати конкурентів, які пропонують аналогічний товар	встановлення в державних закладах охорони здоров'я (задля популяризації методу)
За характером конкурентних переваг	Цінова Важливо за скільки продається товар, та скільки з нього прибутку	Можливе підвищення ціни на нові розробки, зниження на старі версії для заохочення покупців у порівнянні з цінами конкурентів
За інтенсивністю	Марочна Можуть з'являтись конкуренти	На ринку цільової аудиторії поки що конкурентів не виявлено. Але при виході на міжнародний ринок потрібно рекламувати кращий функціонал створеного продукту, встановлювати конкурентоспроможні ціни, та доводити свою надійність

Таблиця 4.7. Обґрунтування факторів конкурентоспроможності

Фактор конкурентоспроможності	**Обґрунтування**
Невелика кількість конкурентів на ринку	На вітчизняному ринку, на який для старту націлена розроблена система, конкурентів немає
Доступність створеного продукту (програмно)	Немає жорстких системних вимог, програма буде працювати навіть на застарілих ПК
Легкість і простота використання	Зручний зрозумілий інтерфейс, створені довідка та інструкція для користувача
Відсутня потреба у постійному супроводі	Не потребує супроводу спеціалістів і постійних доробок з боку розробника
Підключення до мережі Інтернет	Потрібен доступ до Інтернету, як у всіх аналогах
Додаткові компоненти	Немає необхідності встановлення додаткових компонентів, на відміну від деяких аналогів, які не працюють без АПК

SWOT-аналіз (Strength, Weak, Troubles, Opportunities) є фінальним етапом аналізу можливостей впровадження проекту (таблиця 4.8) на основі виділених загроз, можливостей, та сильних і слабких сторін [21].

Базуючися на SWOT-аналізу створюються альтернативи ринкової поведінки (перелік заходів) для виведення проекту на ринок та оптимальний час їх ринкової реалізації, враховуючи потенційні проекти конкурентів, що також можуть бути виведені на ринок у той же час.

Альтернативи, що були визначені у процесі, аналізуються з точки зору ймовірності отримання ресурсів та строків (таблиця 4.9).

Таблиця 4.8. SWOT-аналіз проекту

Сильні сторони (S):	Слабкі сторони (W):
– невелика кількість працівників; – молодий і перспективний колектив; – гнучка політика керівництва; – інноваційні технології	– недостатньо оборотних коштів; – відсутність репутації компанії;
Можливості (O):	Загрози (T):
– додаткові послуги; – вихід на нові ринки; – розширення клієнтської бази; – співробітництво з іншими компаніями	– падіння економіки – вихід на ринок нових конкурентів; – зниження репутації; – зміни тенденцій попиту;

Таблиця 4.9. Альтернативи ринкового впровадження стартап-проекту

Альтернатива ринкової поведінки	**Ймовірність отримання ресурсів**	**Строки реалізації**
Вихід на нові ринки	Пошук інвесторів	1-6 місяців
Розширення виробничої лінії	Пошук інвесторів	Після виходу на ринок основного продукту, до 6 місяців

Отож, спочатку потрібно вивести на основний ринок розроблену систему, а вже потім шукати можливості розширення програмного функціоналу для користувачів.

Кошторис витрат – це об’єднаний план виробничо-фінансової діяльності, що включає усі витрати підприємства на плановий період.

Кошторис на розробку бекенд частини веб-платформи передбачає такі основні витрати:

1) на основну заробітну плату;
2) на додаткову заробітну плату;
3) нарахування на заробітну плату;
4) зношення приміщень, інвентару та обладнання, що використовувались в процесі роботи;
5) на оренду;
6) витрати на техніку та предмети, які використовувалися для розробки рішень;
7) на комплектуючі;
8) на силову електроенергію;
9) інші витрати.

Основна заробітна плата бекенд розробників розраховується за формулою 4.1.

$$З_о = \frac{М}{Т_р} * t, \quad (4.1)$$

де М – місячний посадовий оклад конкретного розробника з бекенд частини, грн.;

Тр – кількість робочих днів у місяці, дні (21…23);

t – число робочих днів розробника.

Над програмним додатком працювали один бекенд розробник та науковий керівник. Місячний оклад бекенд розробника складає близько 15000 грн.. Місячний оклад наукового керівника, який обіймає посаду доцента кафедри – близько 9500 грн.

Близько двох місяців велися роботи з проектування архітектури, розробки та тестування. Керівник проводив консультації один раз в тиждень.

Основна заробітна плата бекенд розробника $З_{оР}$ та керівника $З_{оК}$ складає:

$$З_{оР} = \frac{15000}{22} * 44 = 30000 \text{ грн}$$

$$З_{оК} = \frac{9500}{22} * 9 = 3886 \text{ грн}$$

Кошторис витрат на основну заробітну плату наведено у таблиці 4.1.

Таблиця 4.10 – Кошторис витрат на основну заробітну плату

Назва посади	**Місячний оклад, грн.**	**Денна оплата, грн.**	**Кількість днів роботи**	**Сума витрат на заробітну плату, грн.**	**Примітка**
Бекенд розробник	15000	681,8	44	30000	
Науковий керівник	9500	431,8	9	3886	Консультація раз на тиждень
Всього				33886	

Додаткова заробітна плата працівників розраховується від основної заробітної плати у розмірі 10-12% (формула 4.2). Тоді додаткова заробітна плата складає:

$$З_{Д} = \frac{З_{о} * 10...12\%}{100\%}, \quad (4.2)$$

$$З_{Д} = \frac{33886 * 10}{100} = 3388{,}6 \text{ грн}$$

Відрахування на соціальні заходи здійснюється від суми всіх витрат на оплату праці робітників, зайнятих безпосередньо виробництвом продукції.

До статті нарахування включають відрахування на державне соціальне страхування у вигляді єдиного соціального внеску.

Відрахування, яке здійснюється за нормативами, встановлюється на державному рівні відповідно до класів професійного ризику виробництва. Для видання програмного забезпечення ЄСВ становить 22% (формула 4.3).

$$В_{СЗ} = \frac{22\% * (З_о + З_д)}{100\%}, \quad (4.3)$$

$$В_{СЗ} = \frac{22 * (33886 + 3388{,}6)}{100} = 8200 \text{ грн}$$

Амортизаційні відрахування по кожному пункту розраховуються прямолінійним методом (формула 4.4).

$$А = \frac{Ц}{Т_К} * \frac{Т}{12} \quad (4.4)$$

де Ц – балансова вартість пункту, що входить до підрахунку, грн.;

Тк – термін корисного використання обладнання (для ЕОМ від 2-х років);

Т – термін використання (місяці).

У роботі використовувались: один персональний комп'ютер з операційною системою Windows 10. Підставивши відповідні значення у формулу 4 отримаємо амортизаційні відрахування.

$$А = \frac{10000}{6} * \frac{2}{12} = 277{,}8 \text{ грн}$$

Витрати на матеріали визначаються за формулою 4.5. У зв'язку з особливостями збуту розробленого програмного забезпечення витрати на носії з інсталятором програмного забезпечення відсутні. До уваги беруться витратні канцелярські матеріали.

$$М = \sum_i^n Н_i Ц_i К_i - \sum_i^n В_i Ц_{Вi} \quad (4.5)$$

де $Н_i$ – витрати матеріалу і-го найменування;

$Ц_i$ – вартість матеріалу і-го найменування;

$К_i$ – коефіцієнт транспортних витрат;

$В_i$ – маса відходів матеріалу і-го найменування;

$Ц_{Вi}$ – ціна відходів матеріалу і-го найменування.

Підставивши відповідні значення у формулу 4.5, отримаємо величину витрат на матеріали:

$$М = 80 * 1 + 1{,}1 + 110 * 1 * 1{,}1 + 5 * 4 * 1{,}1 = 231 \text{ грн}$$

Витрати на матеріали зведено до таблиці 4.2.

Таблиця 4.2 – Витратні матеріали

Назва	**Ціна, грн.**	**Кількість, шт.**	**Коефіцієнт транспортних витрат**	**Вартість витраченого матеріалу, грн.**
Блок паперу А4	80	1	1,1	88
Тонер чорний	110	1		121
Ручка кулькова синя	5	4		22
Всього				231

Загальні витрати є сумою всіх попередніх статей витрат.

$$В = З_о + З_Д + В_{СЗ} + А + М \quad (4.6)$$

$$В = 33886 + 3388{,}6 + 8200 + 277{,}8 + 231 = 45983 \text{ грн}$$

Отже, кошторис витрат на розробку бекенд частини програмного продукту становить 45983 грн.

Програмний продукт є незвичний товаром, який відрізняється від продуктів звичного для нас матеріального виробництва. Специфіка у формуванні витрат на збут та виробництво програмних продуктів, а не на виробництво і відтворення.

Немає потреби розраховувати вартість носія з програмним продуктом, так як продукт буде працювати через мережу Інтернет. Витрати на інтелектуальну власність, також потрібно включити (формула 4.7).

$$I_B = K * I_p \quad (4.7)$$

де К – коефіцієнт, що включає відповідні нарахування на заробітну плату, (k = 1,22, 22% ЄСВ);

I_p – отримані кошти від продажу однієї копії продукту.

Запланована вартість від продажу кожної копії програмного продукту, становить 5 000грн. Тоді витрати на інтелектуальну власність становлять:

$$I_B = 1{,}22 * 5000 = 6100 \text{ грн}$$

Також, потрібно включити кошти, які були витраченя на збут продукту. Дані витрати дорівнюють вартості дистрибуції у Windows Store та становить 20% від ціни на продукт. Результати розрахунків всіх секторів виробничої собівартості продемонстровані у таблиці 4.3.

Таблиця 4.3 – Собівартість програмного продукту

Сектор калькуляції	**Витрати, грн.**
Інтелектуальна власність	6100
Інші витрати	40
Всього	6140

Розрахуємо вартість реалізації бекенд частини програмного продукту. Нижню межу ціни реалізації товару розраховують за формулою:

$$Ц_{НМ} = S_В(1 + \frac{Р}{100})(1 + \frac{\alpha_{ПДВ}}{100}) \quad (4.8)$$

де $S_В$ – виробнича собівартість продукту, грн.;

Р – рентабельність, % (Р=30…60%);

$\alpha_{ПДВ}$ – податок на додану вартість, % (20%).

Нижня межа ціни програмного продукту становить:

$$Ц_{НМ} = 6140\left(1 + \frac{50}{100}\right)\left(1 + \frac{20}{100}\right) = 11052 \text{ грн}$$

Верхню межу розраховуємо за формулою:

$$Ц_{ВМ} = Ц_{НМ} * К_{ВЯ} \quad (4.9)$$

де $К_{ВЯ}$ – відносний коефіцієнт якості ($К_{ВЯ}$. = 1.36).

Підставимо відповідні значення у формулу 4.9:

$$Ц_{ВМ} = 11052 * 1{,}36 = 15030 \text{ грн}$$

Тоді ціну реалізації можна прийняти у розмірі 15030 грн.

Аналітично критичний об’єм виробництва можна розрахувати за формулою 4.10.

$$Q_K = \frac{\text{ПВ}}{\text{Ц}_{\text{НМ}} - \text{ЗМ}} \quad (4.10)$$

де ПВ – постійні витрати, які ми прирівнюємо до витрат на створення та підтримку;

ЗМ – виробничі витрати або змінні [20].

$$Q_K = \frac{45983}{11052 - 6140} = 9{,}3 \approx 10 \text{ копій}$$

Отже, необхідно продати 10 копій коду бекенд частини програмного продукту для того, щоб досягти точки беззбитковості.

ВИСНОВКИ

Результатом бакалаврської роботи є створений сайт організації змагань спортивного центру «Золотий Дракон». Даний ресурс буле доступний усім користувачам персональних комп'ютерів чи ноутбуків з активним підключенням до мережі Інтернет.

У першому розділі проведено аналіз існуючих систем по організації та проведенню спортивних змагань, в результаті якого встановлено необхідність додавання у програму наступних компонентів: формування команди, створення змагання та генерація сітки жеребкування.

В другому розділі розроблену трирівневу архітектуру запропонованої системи, що дозволяє описати процеси клієнт-серверної взаємодії (модель прецедентів, яка дозволяє описати взаємодію).

В третьому розділі виконано програмну реалізацію з використанням мови програмування PHP та фреймворком Laravel, що дозволило реалізувати функціональність системи та виконати поставлені практичні завдання.

Проведено детальний опис вимог до програмного забезпечення та характеристик сервера на якому система буде ефективно функціонувати.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

REFERENCES

1. Золотов. С.Ю. Проектирование информационных систем: Учебное методическое пособие. – Томск: ТМЦДО, 2006 – 34с.
2. СІТ 2:2018. Стандарт кафедри інформаційних технологій. Дипломний проект. Вимоги до змісту та оформлення [Чинний від 2018-09-03]. Вид. офіц. Івано-Франківськ, 2018. 43 с.
3. I. Lazarowych and J. Nikolaychuk, "Theory and methods of digital streams randomization in telecommunicational systems," Modern Problems of Radio Engineering, Telecommunications and Computer Science (IEEE Cat. No.02EX542), Lviv-Slavsko, Ukraine, 2002, pp. 265-, doi: 10.1109/TCSET.2002.1015956.
4. Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. – 2-е изд., перераб. и доп.– М.: Финансы и статистика, 2006. – 544 с: ил.
5. Основы системного анализа и проектирования АСУ./ под. ред.А.А.Павлова. – К: Вища школа, 1991.
6. Побудова діаграми варіантів використання – URL: http://moodle.ipo.kpi.ua/.
7. Перегудов Ф.И, Тарасенко Ф.П. Введение в системный анализ. М: Высшая школа, 1992.
8. Системный анализ и структуры управления./ под ред. В.Г.Щорина – М.: Знание, 1975.
9. Шарапов О.Д., Терехов Л.М., Сіднєв С.П. Системний аналіз. К.: Вища школа, 1983.

10. Бібліотека MSDN – цінне джерело інформації для розробників, які використовують засоби, продукти, технології та служби корпорації Майкрософт – URL: https://docs.microsoft.com/ru-ru/
11. Камер Дуглас Компьютерные сети и Internet / Дуглас Камер – М.: Вильямс, 2007. – 640 с.
12. Котеров Д.В. и Кастарев А.Ф. «PHP 5 в подлиннике. Наиболее полное руководство» 2005г. Изд.:BHV. – 1104 с.
13. PHP Language Manual: Type Juggling – URL: https://www.php.net/manual/en/language.types.type-juggling.php
14. The laravel Open Source Project on Open Hub: Languages Page – URL: https://www.openhub.net/p/laravel/analyses/latest/languages_summary
15. PHP, MySQL и другие веб-технологии[Электронный ресурс]: техническая документация по работе с PHP и MySQL – PHP: PHP.SU 2006 - 2011. –URL: http://www.phpfaq.ru/sessions - 17.05.2011.
16. Microsoft SQL Server – URL: http://uk.wikipedia.org/wiki/Microsoft_SQL_Server.
17. Andi Gutmans, Stig Bakken, Derick Rethans. PHP5 Power Programming. — Prentice Hall, 2005. — 704 с.
18. Джамса К. Эффективный самоучитель по креативному Web – дизайну. Перевод с англ. / Крис Джамса, Конрад Кинг, Энди Андерсон – М.: ДиаСофтЮП, 2005. – 672 с.
19. Розенсон И. А. Основы теории дизайна: учебник для вузов / И. А. Розенсон – СПб.: Издательство «Питер», 2010. – 219 с.
20. Фрейн Б. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств / Бен Фрейн – СПб.: Питер, 2014. – 304 с.
21. Кривда В.І., Кривда О.В., Нараєвський С.В. Можливості удосконалення методики SWOT-аналізу / В.І. Кривда, О.В. Кривда, С.В. Нараєвський // Економіко-математичне моделювання соціально-економічних систем ; зб. наук.праць МННЦ ІТіС. – 2007. – № 12. – С. 74–77.

22. Буч Г. Язык UML. Руководство пользователя / Гради Буч, Джеймс Рамбо, Ивар Якобсон – М.: ДМК Пресс, 2007. – 496 с.
23. Гарретт Дж. Веб-дизайн: книга Джесса Гарретта. Элементы опыта взаимодействия / Дж. Гарретт – М.: Символ-Плюс, 2008. – 192 с.

ДОДАТОК А

ЛІСТИНГ БЕКЕНД ЧАСТИНИ ВЕБ-ПЛАТФОРМИ

```
<?php

namespace App\Classes;

use App\Models\Tournament;
use DateTime;

class SetTournamentStatus{

 private $kindDates = [
  1 => 'register_from',
  2 => 'register_to',
  3 => 'date_from',
  4 => 'date_to'
 ];
 private $tournaments = [];
 private $activeTournament = [];
 private $activeStatus = [];
 private $tournamentId = null;

 public function index($tournamentId = null)
 {
  $this->tournamentId = $tournamentId;

  $this->getTournaments();

  $this->setTournamentStatus();
 }

 private function setTournamentStatus()
 {
  $date = date("Y-m-d");

  foreach ($this->tournaments as $key => $tournament) {
   $this->activeTournament = $tournament;

   foreach($this->kindDates as $key => $kind){

    if($tournament[$kind] > $date){
     $currentTime = new DateTime($date);
```

```
        $tournamentTime = new DateTime($tournament[$kind]);

        $this->activeStatus = [
         'status'      => $key,
         'status_days' => $currentTime
          ->diff($tournamentTime)
          ->days
        ];

        $this->saveStatusInDatabase();

        break;
       }

       if($key + 1 == count($this->kindDates)){
        $this->activeStatus = [
         'status'      => 5,
         'status_days' => 0
        ];

        $this->saveStatusInDatabase();
       }
     }
    }
   }

   private function saveStatusInDatabase()
   {
    Tournament
     ::where('id', $this->activeTournament['id'])
     ->update($this->activeStatus);
   }

   private function getTournaments()
   {
    $this->tournaments = Tournament
     ::getTournamnetListForStatus($this->tournamentId);
   }

}

<?php
```

```
namespace App\Classes;

use App\Models\TournamentMatchSettings;
use App\Models\TournamentMatchAthleteRelation;
use App\Models\Tournament;
use DateTime;

class SetMatchAthleteRelation{

 private $tournament = [];
 private $athletesCouple = [];
 private $athletes = [];
 private $day;
 private $ring;
 private $deep = 0;
 private $tournamentId = null;
 private $weightId = null;
 private $firstSortingDraws = [];
 private $alphabet = [];
 private $drawRange = [
  1, 2, 4, 8, 16, 32, 64, 128, 256
 ];
 private $countDrawsTypes = [
  32  => 2,
  64  => 4,
  128 => 8
 ];

 public function __construct()
 {
  $this->alphabet = range('A', 'Z');
 }

 public function test(){
  $this->index(request('tournament_id'));
 }

 public function index($tournamentId = null, $weight_id =
null)
 {
  $this->tournamentId = $tournamentId;
  $this->weightId = $weight_id;
```

```
  $this->deleteOldData();

  foreach ($this->getTournaments() as $this->tournament)
{
   foreach ($this->getTournamentDays() as $this->day){
    foreach ($this->getTournamentRings() as $this->ring){
     $this->setMatchRelation();

     $this->deep = 0;
    }
   }
  }
 }

 private function deleteOldData()
 {
  if($this->tournamentId)
   TournamentMatchAthleteRelation
    ::leftJoin(
     'tournament_match_settings',
     'tournament_match_settings.id',

'tournament_match_athlete_relations.tournament_match_sett
ings_id'
    )
    ->where('tournament_match_settings.tournament_id',
$this->tournamentId)
    ->delete();
  else
   TournamentMatchAthleteRelation::truncate();
 }

 private function setMatchRelation()
 {
  $this->getTournamentAthletesByRing();

  if(count($this->athletes) != 0){

   $this->setAthletesCouple();
   $this->saveAthletesCoupleInDatabase();
  }
 }
```

```
 private function saveAthletesCoupleInDatabase()
 {
  if($this->deep == 0){
   $this->saveAthletesCoupleInDatabaseFirst();
   return;
  }

  foreach ($this->athletesCouple as $weightKey =>
$weightGroup){
   foreach ($weightGroup as $drawKey => $drawGroup){
    foreach ($drawGroup as $key => $couple) {
     $coupleId = TournamentMatchAthleteRelation
      ::orderBy('id', 'DESC')
      ->first();

     $coupleId = $coupleId != null
      ? $coupleId->id
      : 0;

     $data = [
      'tournament_match_settings_id' => $couple[0]['id'],
      'deep'                         => $this->deep,
      'order'                        => 1,
      'couple_id'          => $coupleId,
      'parent_couple_id'       => $couple[0]['couple_id'],
      'draw_group'         => $drawKey,
      'place'           => $couple[0]['place'] - 1,
      'additional_order'       =>
$couple[0]['additional_order']
     ];

     if($this->deep == 6)
      $data['draw_group'] = 2;

     $data['fight_number'] = $this-
>getFightNumberByRing();

     if($this->deep == 1 && $couple[0]['deep'] == 1){
      TournamentMatchAthleteRelation
       ::where('couple_id', $couple[0]['couple_id'])
       ->update([
        'fight_number' => $data['fight_number'],
        'place' => $couple[0]['place'] - 1
```

```
        ]);

      } else{
       if(count($couple) == 1){
        $data['fight_number'] = 0;


        TournamentMatchAthleteRelation::create($data);
       } else {
        if($couple[0]['fight_number'] == 0)
         $data['athlete_id'] = $couple[0]['athlete_id'];
        else
         $data['athlete_id'] = 0;

        TournamentMatchAthleteRelation::create($data);

        if(
         $couple[1]['fight_number'] == 0
         && $couple[0]['fight_number'] != 0
        )
         $data['athlete_id'] = $couple[1]['athlete_id'];
        else
         $data['athlete_id'] = 0;

        $data['parent_couple_id'] =
$couple[1]['couple_id'];
        $data['order'] = 2;

        TournamentMatchAthleteRelation::create($data);
       }
      }
     }
    }
   }

   $this->deep++;

   $this->setMatchRelation();
  }

  private function saveAthletesCoupleInDatabaseFirst()
  {
```

```
  foreach ($this->athletesCouple as $groupKey =>
$groupDivide) {
   foreach ($groupDivide as $weightKey => $weightDivide)
{
    foreach ($weightDivide as $drawNumberKey =>
$drawNumber) {
     $drawNumber = array_reverse($drawNumber);

     foreach ($drawNumber as $key => $couple) {
      $fight_number = $this->getFightNumberByRing();

      $coupleId = TournamentMatchAthleteRelation
       ::orderBy('id', 'DESC')
       ->first();

      $coupleId = $coupleId != null
       ? $coupleId->id
       : 0;

      $couple[0]['couple_id'] = $coupleId;
      $couple[1]['couple_id'] = $coupleId;

      if($couple[1]['athlete_id'] == 0 ||
$couple[0]['deep'] == 1){
       $couple[0]['fight_number'] = 0;
       $couple[1]['fight_number'] = 0;
      } else{
       $couple[0]['fight_number'] = $fight_number;
       $couple[1]['fight_number'] = $fight_number;
      }

      TournamentMatchAthleteRelation::create($couple[0]);
      TournamentMatchAthleteRelation::create($couple[1]);
     }
    }
   }
  }

  $this->deep++;

  $this->setMatchRelation();
 }
```

```
 private function saveOneParticipant($weightGroup)
 {
  $athlete = $weightGroup[0];

  $coupleId = TournamentMatchAthleteRelation
   ::orderBy('id', 'DESC')
   ->first();

  $coupleId = $coupleId != null
   ? $coupleId->id
   : 0;

  $data = [
   'athlete_id'                    =>
$athlete['athlete_id'],
   'tournament_match_settings_id' => $athlete['id'],
   'draw_group'                    => 1,
   'change_able'                   => 0,
   'fight_number'                  => 0,
   'deep'                          => 2,
   'order'                         => 1,
   'couple_id'         => $coupleId,
   'place'          => 1
  ];

  TournamentMatchAthleteRelation::create($data);
 }

 private function setAthletesCouple()
 {
  if($this->deep == 0){
   $this->setAthletesCoupleFirst();
   return;
  }

  $dividedAthletes = [];
  $this->athletesCouple = [];

  foreach ($this->athletes as $key => $athlete) {
   $dividedAthletes
    [$athlete['weight_id']][$athlete['draw_group']][] =
$athlete;
  }
```

```
  foreach ($dividedAthletes as $weightKey =>
$weightDivide) {
   if(count($weightDivide) > 1){
    $check = 1;
    $counter = 1;
    $couple = 0;

    for($i = 0; $i < count($weightDivide); $i++){

if(count($weightDivide[array_keys($weightDivide)[$i]]) >
1)
      $check = 0;
    }

    if($check){
     foreach ($weightDivide as $key => $value) {
      if($couple == 0){
       $weightDivide[$key][0] = $value[0];
       $couple = $key;
      } else{
       $weightDivide[$couple][1] = $value[0];
       unset($weightDivide[$key]);
       $couple = 0;
      }

     }
    } else{
     foreach ($weightDivide as $key => $value) {
      if(count($value) == 1)
       unset($weightDivide[$key]);
     }
    }
   }

   foreach ($weightDivide as $drawNumberKey =>
$drawNumber) {
    $drawGroup = &$this->athletesCouple
     [$weightKey][$drawNumberKey];
    $couple = 1;

    foreach ($drawNumber as $key => $athlete){
     if(
```

```
      ($athlete['deep'] == 0 && $this->deep == 1)
      || ($athlete['deep'] >= 1 && $this->deep > 1)
     ){
      if($couple == 1){
       $drawGroup[$key][0] = $athlete;
       $couple = 0;
      } else{
       if($athlete['id'] == $drawNumber[$key - 1]['id']){
        $drawGroup[$key - 1][1] = $athlete;
        $couple = 1;
       } else{
        $drawGroup[$key][0] = $athlete;
       }
      }
     } else{
      $drawGroup[$key][0] = $athlete;
      $drawGroup[$key][1] = $athlete;
     }
    }

    if(count($drawGroup) > 1){
     end($drawGroup);
     $key = key($drawGroup);

     if(count($drawGroup[$key]) == 1){
      $drawGroup[$key][] = $drawGroup[$key][0];
     }
    }
   }
  }
 }

 private function setAthletesCoupleFirst()
 {
  $dividedAthletes = [];
  $this->athletesCouple = [];

  foreach ($this->athletes as $key => $athlete) {
   $dividedAthletes
    [$athlete['age_group_id']][$athlete['weight_id']][] =
$athlete;
  }
```

```
  foreach ($dividedAthletes as $groupKey => $groupDivide)
{
   foreach ($groupDivide as $weightKey => $weightDivide)
{
    if(count($weightDivide) == 1 && $this->deep == 0){
     $this->saveOneParticipant($weightDivide);

     continue;
    }

    $place = $this-
>getPlaceByCount(count($weightDivide));

    $weightDivide = $this-
>divideSimilarAthletes($weightDivide);

    $weightDivide = $this->divideDraws($weightDivide);

    $this->drawCounts[$weightKey] = count($weightDivide);

    foreach ($weightDivide as $drawNumberKey =>
$drawNumber) {
     $minCount = null;
     $maxCount = null;
     $couple = 1;
     $data = [];
     $drawNumberCount = count($drawNumber);

     for($i = 0; $i < count($this->drawRange); $i++){
      if(
       $drawNumberCount > $this->drawRange[$i]
       && $drawNumberCount < @$this->drawRange[$i + 1]
      ){
       $minCount = $this->drawRange[$i];
       $maxCount = $this->drawRange[$i + 1];
      }
     }

     $deepFirstCount = 0;
     $deepZeroCount = $drawNumberCount;

     if($maxCount !== null){
      $deepFirstCount = $maxCount - $drawNumberCount;
```

```
      $deepZeroCount = $drawNumberCount -
$deepFirstCount;
     }

     $dividedDraws = [];
     $direction = 0;

     $this->dividedAthletesByCouple(
      $drawNumberCount,
      $deepFirstCount,
      $deepZeroCount,
      $direction
     );

     ksort($this->firstSortingDraws);

     $athleteCounter = 0;
     $coupleCounter = 0;

     $weightGroup = &$this->athletesCouple
      [$groupKey][$weightKey][$drawNumberKey];

     for($i = 0; $i < count($this->firstSortingDraws);
$i++){
      $key = array_keys($this->firstSortingDraws)[$i];
      $element = $this->firstSortingDraws[$key];

      if($element['count'] == 1){
       $athlete = $drawNumber[$athleteCounter];
       $athleteCounter++;
       $deep = 0;

       if(
        count($weightDivide) == 2
        && $drawNumberKey == 1
        && count($weightDivide[0]) == 17
        && count($weightDivide[1]) == 16
       )
        $deep = 1;

       $weightGroup[$coupleCounter][0] = [
        'athlete_id' => $athlete['athlete_id'],
        'tournament_match_settings_id' => $athlete['id'],
```

```
    'draw_group' => $drawNumberKey + 1,
    'change_able' => 1,
    'place' => $place,
    'deep' => $deep,
    // 'order' => $element['direction'] == 1 ? 1 : 2,
    'order' => 1,
    'additional_order' => $i
   ];
   $weightGroup[$coupleCounter][1] = [
    'athlete_id' => 0,
    'tournament_match_settings_id' => $athlete['id'],
    'draw_group' => $drawNumberKey + 1,
    'change_able' => 1,
    'place' => $place,
    'deep' => $deep,
    // 'order' => $element['direction'] == 2 ? 1 : 2,
    'order' => 2,
    'additional_order' => $i
   ];
   $coupleCounter++;
  } else{
   $athlete = $drawNumber[$athleteCounter];
   $athleteCounter++;
   $deep = $element['deep'];

   if(
    count($weightDivide) == 2
    && $drawNumberKey == 1
    && count($weightDivide[0]) == 17
    && count($weightDivide[1]) == 16
   )
    $deep += 1;

   $weightGroup[$coupleCounter][0] = [
    'athlete_id' => $athlete['athlete_id'],
    'tournament_match_settings_id' => $athlete['id'],
    'draw_group' => $drawNumberKey + 1,
    'change_able' => 1,
    'place' => $place,
    'deep' => $deep,
    'order' => 1,
    'additional_order' => $i
   ];
```

```
        $athlete = $drawNumber[$athleteCounter];
        $athleteCounter++;

        $weightGroup[$coupleCounter][1] = [
         'athlete_id' => $athlete['athlete_id'],
         'tournament_match_settings_id' => $athlete['id'],
         'draw_group' => $drawNumberKey + 1,
         'change_able' => 1,
         'place' => $place,
         'deep' => $deep,
         'order' => 2,
         'additional_order' => $i
        ];

        $coupleCounter++;
       }
      }

      $this->firstSortingDraws = [];

      krsort($weightGroup);
     }
    }
   }

}

private function dividedAthletesByCouple(
 $athletesCount,
 $deepFirstCount,
 $deepZeroCount,
 $direction,
 $orderString = '',
 $counter = 0
){
 if($athletesCount > 2){
  if($this->checkOnEvenNumber($athletesCount)){
   $dividedCount = $athletesCount / 2;

   $this->dividedAthletesByCouple(
    $athletesCount / 2,
    $deepFirstCount / 2,
```

```
     $deepZeroCount / 2,
     1,
     $orderString . $this->alphabet[$counter] . $this-
>alphabet[0],
     $counter + 1
    );

    $this->dividedAthletesByCouple(
     $athletesCount / 2,
     $deepFirstCount / 2,
     $deepZeroCount / 2,
     2,
     $orderString . $this->alphabet[$counter] . $this-
>alphabet[1],
     $counter + 1
    );
   } else{
    $athletesEvenNumber = $this
     ->getEvenIntNumberByDivide($athletesCount);
    $athletesOddNumber = $athletesCount -
$athletesEvenNumber;

    $deepFirstEvenCount = $this
     ->getEvenIntNumberByDivide($deepFirstCount);
    $deepFirstOddCount = $deepFirstCount -
$deepFirstEvenCount;

    $deepZeroEvenCount = $this
     ->getEvenIntNumberByDivide($deepZeroCount);
    $deepZeroOddCount = $deepZeroCount -
$deepZeroEvenCount;

    if($direction == 1 || $direction == 0){
     $athletesDirectionEvenNumber = 2;
     $athletesDirectionOddNumber = 1;
     $athletesOrderNumberOdd = $orderString
      . $this->alphabet[$counter]
      . $this->alphabet[0];
     $athletesOrderNumberEven = $orderString
      . $this->alphabet[$counter]
      . $this->alphabet[1];
    } else{
     $athletesDirectionEvenNumber = 1;
```

```
    $athletesDirectionOddNumber = 2;
    $athletesOrderNumberOdd = $orderString
     . $this->alphabet[$counter]
     . $this->alphabet[1];
    $athletesOrderNumberEven = $orderString
     . $this->alphabet[$counter]
     . $this->alphabet[0];
   }

   $this->dividedAthletesByCouple(
    $athletesOddNumber,
    $deepFirstOddCount,
    $deepZeroOddCount,
    $athletesDirectionOddNumber,
    $athletesOrderNumberOdd,
    $counter + 1
   );
   $this->dividedAthletesByCouple(
    $athletesEvenNumber,
    $deepFirstEvenCount,
    $deepZeroEvenCount,
    $athletesDirectionEvenNumber,
    $athletesOrderNumberEven,
    $counter + 1
   );
  }
 } else{
  if($deepFirstCount == 0)
   $deep = 0;
  else
   $deep = 1;

  $this->firstSortingDraws[$orderString] = [
   'count' => $athletesCount,
   'deep' => $deep,
   'direction' => $direction,
  ];
 }
}

private function checkOnEvenNumber($number)
{
 if($number % 2 == 0)
```

```
    return true;
   else
    return false;
  }

  private function getEvenIntNumberByDivide($divided)
  {
   $dividedNumber = round($divided / 2, 0);

   if($dividedNumber % 2 == 0)
    return $dividedNumber;
   else
    return $divided - $dividedNumber;
  }

  private function divideDraws($weightDivide)
  {
   $countChunks = 1;
   $sizeInChunk = null;

   foreach ($this->countDrawsTypes as $key => $value) {
    if(
     $key < count($weightDivide)
     && $key * 2 > count($weightDivide)
    )
     $countChunks = $value;
   }

   $sizeInChunk = ceil(count($weightDivide) /
$countChunks);

   return array_chunk(
    $weightDivide,
    $sizeInChunk
   );
  }

  private function getFightNumberByRing()
  {
   return
TournamentMatchAthleteRelation::getMaxFigthNumberByRing(
    $this->tournament['id'], $this->day, $this->ring
   ) + 1;
```

```
 }

 private function getTournamentAthletesByRing()
 {
  $this->athletes = [];

  $parameters = [$this->tournament['id'], $this->day,
$this->ring];

  if($this->deep > 1)
   $parameters[] = $this->deep;

  switch($this->deep){
   case 0:
    $this->athletes = TournamentMatchSettings
     ::getAthletesByRing(...$parameters);
    break;
   default:
    $this->athletes = TournamentMatchAthleteRelation
     ::getAthletesByDeep(...$parameters);
  }
 }

 private function getPlaceByCount($count)
 {
  $places = [
   1 => 1,
   2 => 2,
   3 => 4,
   4 => 8,
   5 => 16,
   6 => 32,
   7 => 64,
   8 => 128
  ];

  $place = 0;

  for($i = 1; $i <= count($places); $i++){
   if(
    $count > $places[$i]
    && $count <= @$places[$i + 1]
   ){
```

```
    $place = array_keys($places)[$i - 1];

    break;
   }
  }

  return $place;
 }

 private function divideSimilarAthletes($athletes)
 {
  $athletesCount = [];
  $trainerAthletesGroup = [];
  $generalOrder = [];
  $maxCount = 0;
  $dividedAthletes = [];

  foreach ($athletes as $key => $athlete) {
   if(!array_key_exists($athlete['trainer_id'],
$athletesCount))
    $athletesCount[$athlete['trainer_id']] = 0;

   $athletesCount[$athlete['trainer_id']]++;
   $trainerAthletesGroup[$athlete['trainer_id']][] =
$athlete;
  }

  arsort($athletesCount);

  foreach ($athletesCount as $key => $athleteCount) {
   $generalOrder = array_merge(
    $generalOrder,
    $trainerAthletesGroup[$key]
   );
  }

  $maxCount =
$athletesCount[array_keys($athletesCount)[0]];

  for($i = 0; $i < $maxCount; $i++){
   for($j = $i; $j < count($generalOrder); $j +=
$maxCount){
    $dividedAthletes[] = $generalOrder[$j];
```

```
    }
   }

   return $dividedAthletes;
  }

  private function getTournamentRings()
  {
   return range(1, $this->tournament['rings_count']);
  }

  private function getTournamentDays()
  {
   $dateFrom = new DateTime($this-
>tournament['date_from']);
   $dateTo = new DateTime($this->tournament['date_to']);

   $difference = $dateFrom->diff($dateTo)->d + 1;

   return range(1, $difference);
  }

  private function getTournaments()
  {
   return Tournament::getDrawTournaments($this-
>tournamentId);
  }

}
<?php

namespace App\Http\Controllers\Client;

use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use App\Models\Tournament;
use App\Models\TournamentFiles;
use App\Models\TournamentAthlete;
use App\Models\AgeGroup;
use App\Models\Athlete as AtheleteModel;
use App\Models\TournamentAgeGroupWeight;
use App\Models\TournamentMatchAthleteRelation;
use App\Models\TournamentMatchSettings;
```

```
use App\Models\TournamentDrawShow;
use Storage;

class ClientTournamentPageController extends Controller
{

  public function __construct()
  {
    parent::__construct();
  }

  //GENERAL PAGE

  public function index($id)
  {
    return view('client.tournament-page.reglament',
compact('id'));
  }

  public function getTournamentData()
  {
    $tournament =
Tournament::getTournamentById(request('id'));

    return response()->json($tournament, 200);
  }

  public function getTournamentFiles()
  {
    $files =
TournamentFiles::getFilesById(request('id'));

    return response()->json($files, 200);
  }

  public function getAgeGroupsSidebar()
  {
    $ageGroups =
AgeGroup::getAgeGroupsSidebar(request('id'));

    return response()->json($ageGroups, 200);
  }
```

```
  public function getTournamentStatistic()
  {
    $statistic =
Tournament::getTournamentStatistic(request('id'));

    return response()->json($statistic, 200);
  }

  //TEAMS PAGE

  public function teams($id)
  {
    return view('client.tournament-page.teams',
compact('id'));
  }

  public function getTeamsData()
  {
    $teams = TournamentAthlete::getTeams(request('id'));

    return response()->json($teams, 200);
  }

  public function getTeamAthletes()
  {
    $athletes = AtheleteModel::getTeamAtheltes(
      request('userId'),
      request('tournamentId'),
      request('search')
    );

    // $athletes = TournamentAthlete::getTeamAtheltes(
    //  request('userId'),
    //  request('tournamentId'),
    //  request('search')
    // );

    return response()->json($athletes, 200);
  }

  //PARTICIPANT PAGE

  public function participants($id)
```

```
  {
    return view('client.tournament-page.participant',
compact('id'));
  }

  public function getAgeGroups()
  {
    $ageGroups = AgeGroup
      ::where('age_groups.tournament_id', request('id'))
      ->orderBy('date_from', 'DESC')
      ->get();

    return response()->json($ageGroups, 200);
  }

  public function getAgeGroupsDraw()
  {
    $ageGroups = AgeGroup
      ::selectRaw('DISTINCT age_groups.id, age_groups.*')
      ->leftJoin(
        'tournament_match_settings',
        'tournament_match_settings.age_group_id',
        'age_groups.id'
      )
      ->leftJoin(
        'tournament_draw_shows',
        function($query){
          $query
            ->on(
              'tournament_draw_shows.tournament_id',
              'tournament_match_settings.tournament_id'
            )
            ->on(
              'tournament_draw_shows.day',
              'tournament_match_settings.day'
            );
        }
      )
      ->leftJoin(
        'tournament_match_athlete_relations as relation',
        'relation.tournament_match_settings_id',
        'tournament_match_settings.id'
      )
```

```
      ->where(function($query){
        if(auth()->user() && auth()->user()->id !=
Tournament::find(request('id'))->creator_id)
          $query->where('tournament_draw_shows.show', 1);
      })
      ->where('relation.id', '!=', null)
      ->where('age_groups.tournament_id', request('id'))
      ->orderBy('date_from', 'DESC')
      ->get();

    return response()->json($ageGroups, 200);
  }

  public function getWeights()
  {
    $weights =
TournamentAgeGroupWeight::getTournamentWeightsByGroupId(
      request('ageGroupId')
    );

    return response()->json($weights, 200);
  }

  public function getWeightsDraw()
  {
    $weights =
TournamentAgeGroupWeight::getDrawWeightsByGroupId(
      request('ageGroupId')
    );

    return response()->json($weights, 200);
  }

  public function getOwnParticipantForAge()
  {
    $participants =
AtheleteModel::getOwnParticipantForAge(
      auth()->user()->id,
      request('data'),
      request('search')
    );

    return response()->json($participants, 200);
```

```
  }

  public function addParticipantToTournament()
  {
    $data = request('data');

    $participantsIds = $data['ids'];
    $generalData = $data['generalData'];

    foreach ($participantsIds as $key => $participantsId)
{

      TournamentAthlete::insert([
        'tournament_id' => $generalData['tournamentId'],
        'athlete_id' => $participantsId,
        'age_group_id' => $generalData['ageGroupId'],
        'weight_id' => $generalData['weightId']
      ]);

    }
  }

  public function getTournamentParticipants()
  {
    $data = request('data');
    $search = request('search');
    $athletes =
TournamentAthlete::getTournamentParticipants($data,
$search);

    return response()->json($athletes, 200);
  }

  public function deleteOwnParticipant()
  {
    TournamentAthlete::destroy(request('id'));
  }

  public function deleteOwnParticipantByAthelete()
  {
    TournamentAthlete
      ::where('tournament_id', request('tournament'))
      ->where('athlete_id', request('id'))
```

```
      ->delete();
  }

  //DRAW PAGE
  public function draw($id)
  {
    return view('client.tournament-page.draw',
compact('id'));
  }

  public function getTournamentDraw()
  {
    $athletes =
TournamentMatchAthleteRelation::getDrawByWeight(
      request('data')['weight_id']
    );

    $drawInf = TournamentMatchSettings::getDrawInf(
      request('data')['weight_id']
    );

    $data = [
      'athletes' => $athletes,
      'drawInf'  => $drawInf
    ];

    return response()->json($data, 200);
  }

  public function getTournamentAllDraw()
  {
    $athletes =
TournamentMatchAthleteRelation::getDrawsByTournament(
      request('tournament_id')
    );
    $athletesGroup = [];
    $drawsInfGroup = [];

    foreach($athletes as $athlete){
      $athletesGroup[$athlete['weight_id']][] = $athlete;
    }
```

```
    $drawsInf =
TournamentMatchSettings::getTournamentDrawInf(
      request('tournament_id')
    );

    foreach($drawsInf as $drawInf){
      $drawsInfGroup[$drawInf['weight_id']] = $drawInf;
    }

    $data = [
      'athletes' => $athletesGroup,
      'drawInf'  => $drawsInfGroup
    ];

    return response()->json($data, 200);
  }

  public function setScoreData()
  {
    $scores = request('data');

    foreach ($scores as $key => $score) {
      TournamentMatchAthleteRelation
        ::where('id', $key)
        ->update([
          'score' => $score
        ]);
    }

    $this->setWinnerAthletes();
  }

  public function setWinnerAthletes()
  {
    $athletes =
TournamentMatchAthleteRelation::getWinnerAthletesByScore(
);

    foreach ($athletes as $key => $athlete) {
      TournamentMatchAthleteRelation
        ::where('parent_couple_id',
$athlete['couple_id'])
        ->update([
```

```
            'athlete_id' => $athlete['athlete_id']
          ]);
      }
    }

    public function getRightFilterData()
    {
      $data =
TournamentMatchAthleteRelation::getResultFilterData(
        request('tournamentId'),
        request('leftFilterId')
      );

      return response()->json($data, 200);
    }

    public function results($id)
    {
      return view('client.tournament-page.results',
compact('id'));
    }

    public function getTournamentResults()
    {
      $results =
TournamentMatchAthleteRelation::getTournamentResults(
        request('id'),
        request('filterData')
      );

      return response()->json($results, 200);
    }

    public function changeParticipant()
    {
      $participant = request('athletes');

      $checkAthlete1 = TournamentMatchAthleteRelation
        ::where('couple_id', $participant[1]['couple_id'])
        ->where('order', 2)
        ->first();

      $checkAthlete2 = TournamentMatchAthleteRelation
```

```
      ::where('couple_id', $participant[2]['couple_id'])
      ->where('order', 2)
      ->first();

    if($checkAthlete1 == null ||
$checkAthlete1['athlete_id'] == 0)
      TournamentMatchAthleteRelation
        ::where('parent_couple_id',
$participant[1]['couple_id'])
        ->where('order', 1)
        ->update([
          'athlete_id' => $participant[2]['athlete_id']
        ]);

    // dd($checkAthlete1);

    if($checkAthlete2 == null ||
$checkAthlete2['athlete_id'] == 0)
      TournamentMatchAthleteRelation
        ::where('parent_couple_id',
$participant[2]['couple_id'])
        ->where('order', 1)
        ->update([
          'athlete_id' => $participant[1]['athlete_id']
        ]);

    TournamentMatchAthleteRelation
      ::where('id', $participant[1]['relationId'])
      ->update([
        'athlete_id' => $participant[2]['athlete_id']
      ]);

    TournamentMatchAthleteRelation
      ::where('id', $participant[2]['relationId'])
      ->update([
        'athlete_id' => $participant[1]['athlete_id']
      ]);

    return response(null, 200);
  }

  //ADDITIONAL METHOD
```

```
  public function getCountParticipant()
  {
    $count = TournamentAthlete
      ::where('tournament_id', request('id'))
      ->count();
    $limit = Tournament
      ::where('id', request('id'))
      ->first()
      ->participants_limit;

    $available = $limit - $count;

    return response()->json($available, 200);
  }

  public function checkOnDrawDisabled()
  {
    $id = request('id');

    $drawShow = TournamentDrawShow
      ::where('tournament_id', $id)
      ->where('show', 1)
      ->count();
    $drawCount = TournamentMatchAthleteRelation
      ::from('tournament_match_athlete_relations as
relation')
      ->leftJoin(
        'tournament_match_settings as settings',
        'settings.id',
        'relation.tournament_match_settings_id'
      )
      ->where('settings.tournament_id', $id)
      ->count();

    $check = $drawShow >= 1 && $drawCount >= 1;

    return response()->json($check, 200);
  }

}
```