

**Київський національний університет імені Тараса Шевченка**  
**Факультет комп'ютерних наук та кібернетики**

**Звіт**

з дисципліни «Інструментальні середовища та технології  
програмування»  
на тему: «Лабораторна робота. Волейбольні чемпіонати»

студентки 2-го курсу, групи К-26  
спеціальності 122 «Комп'ютерні науки»  
Халваші Катерини  
Керівник Омельчук Л.Л.

## РЕФЕРАТ

Обсяг роботи 37 сторінок, 21 ілюстрація, 10 таблиць, 1 додаток

С#, ASP NET, SQL SERVER, ФУТБОЛ, ЧЕМПІОНАТ, ТАБЛИЦІ, ЗВ'ЯЗОК ТАБЛИЦЬ, АВТОРИЗАЦІЯ, ГРАФІКИ, ІМПОРТ, ЕКСПОРТ, КОРИСТУВАЧ, РІВНІ ДОСТУПУ, ВІЗУАЛІЗАЦІЯ

**Об'єкт:** розробка веб-сервісу, який візуалізує базу даних волейбольних чемпіонатів.

**Мета роботи:** розробка продукту для візуалізації бази даних волейбольних чемпіонатів з використанням технології ASP NET. Реалізувати всі зв'язки, можливість імпорту та експорту та візуалізації даних за допомогою таблиць та вбудованих діаграм.

**Методи розроблення:** аналіз існуючих проблем, проектування майбутньої архітектури програми, покрокова розробка з розбиттям на підзадачі. Інструменти розроблення: безкоштовне середовище розробки Visual Studio та SQL Server для БД, мова програмування С#. Було використано технологію ASP.NET CORE MVC.

**Результати роботи:** виконано загальний огляд проектування архітектури веб-додатків. Було досліджено та застосовано кросплатформенну технологію ASP.NET Core. Розроблено додаток для додавання, редагування, видалення, візуалізації бази даних компаній, також експорту і імпорту інформації певних таблиць.

# Зміст

<u>СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ</u>	4
<u>ВСТУП</u>	5
<u>РОЗДІЛ 1. ОГЛЯД НАЯВНИХ НА РИНКУ СИСТЕМ</u>	7
<u>РОЗДІЛ 2. ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ</u>	10
<u>РОЗДІЛ 3. ПРИЗНАЧЕННЯ ТА ЦІЛІ СТВОРЕННЯ СИСТЕМИ</u>	15
<u>3.1 Призначення системи</u>	15
<u>3.2 Цілі створення системи</u>	16
<u>3.3 Вимоги до системи</u>	17
<u>3.3.1 Вимоги до системи в цілому</u>	17
<u>3.3.2 Вимоги до функцій, які використовуються системою</u>	18
<u>3.4 Технічні вимоги</u>	19
<u>РОЗДІЛ 4. ОПИС ОРГАНІЗАЦІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ</u>	20
<u>4.1 Логічна структура бази даних</u>	20
<u>РОЗДІЛ 5. РЕАЛІЗАЦІЯ СИСТЕМИ</u>	24
<u>РОЗДІЛ 6. ІНСТРУКЦІЯ КОРИСТУВАЧА</u>	30
<u>ВИСНОВКИ</u>	36
<u>СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ</u>	37
<u>Додаток А</u>	38

## СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

EF – Entity Framework.

IDE - Integrated Development Environment

ASP.NET – Active Server Pages

MVC - Model–View–Controller

SQL - Structured query language

БД – База даних

СУБД - Система управління базами даних

PostgreSQL - об'єктно-реляційна система  
керування базами даних (СКБД)

VS – Visual Studio

.xlsx - вид zip-архіву, що містить електронну таблицю

## Вступ

**Актуальність теми.** Волейбол являється найпопулярнішим видом спорту, яким захоплюються сотні мільйонів людей. Щоденно волейбольні матчі переглядають велика кількість вболівальників. Усі фанати слідкують за статистикою Чемпіонатів, Команд, Збірних. В зв'язку з розвитком технологій усі розваги повинні підлаштовуватись під глядача, для його зручності. Потрібна зручна платформа для розміщення інформації та зручного перегляду для вболівальника, яка економить час та сили. Через це розробка веб застосунку для відстеження волейбольних чемпіонатів є актуальною та корисною задачею.

**Методи дослідження.** Методи дослідження включають в себе вивчення літератури, аналіз існуючих веб застосунків для відстеження футбольних чемпіонатів та використання сучасних технологій C#, Visual Studio 2022 та ASP.NET Core.

**Основні цілі та завдання.** Основними цілями та завданнями проекту були розробка веб застосунку, який забезпечуватиме користувачів зручний та швидкий доступ до інформації про волейбольні чемпіонати. Для досягнення цих цілей було запропоновано створення веб застосунку, який може зберігати інформацію та надавати її користувачу у зручному виді. Це все працює спільно з базою даних, яка включає в себе такі таблиці, як Чемпіонати, Матчі, Стадіони, Команди, Гравці та проміжну таблицю Забиті Голи.

**Переваги та застосування.** Однією з ключових переваг даного застосунку є можливість зберігання та відображення даних про волейбольні чемпіонати, команди, гравців, матчі та стадіони. Крім того, вона дозволяє зберігати дані про забиті голи та авторів цих голів у кожному матчі. Це робить її корисною для волейбольних організацій, тренерів та волейбольних фанатів.

**Можливий розвиток.** Одним з можливих напрямів подальшого розвитку даної лабораторної роботи може бути розширення функціональності застосунку. Наприклад, можна додати функцію аналізу гри команди, детальну статистику гравців та команд, а також можливість відстежувати інформацію про трансфери гравців між командами.

## Розділ 1. Огляд наявних на ринку систем

На ринку існує багато волейбольних систем, що забезпечують управління волейбольними клубами і організацію волейбольних турнірів та чемпіонатів. Деякі з них - Volleyball Manager, FIFA Manager, Pro Evolution Soccer і т.д.

Розглянемо деякі з них:

- Однією з найбільш популярних систем для ведення футбольних статистик є програмне забезпечення "Volleyball Manager". Ця система дозволяє вести статистику про гравців, команди, матчі, стадіони та інші футбольні показники. "Volleyball Manager" є платним програмним забезпеченням та призначений в основному для тренерів та менеджерів команд.
- Іншою популярною системою є веб-сайт "SofaScore Volleyball", який надає користувачам доступ до статистики матчів та команд. Сайт містить інформацію про склади команд, статистику по гравцях, історію матчів та інші показники. "Volleyball" є безкоштовним і доступним для використання всім користувачам.
- Також існують інші системи для ведення футбольної статистики, такі як "Opta Sports", "StatsBomb" та "InStat Volleyball". Ці системи є спеціалізованими на аналізі гри та даних, що дозволяє клубам, тренерам та менеджерам команд зробити більш обґрунтовані рішення при підготовці до матчів та під час їх проведення.

Основні переваги цих систем полягають у глибокому аналізі волейбольних матчів та здатності використовувати велику кількість статистичних даних, щоб допомогти тренерам і керівникам клубів приймати рішення щодо складу команди, тренувань, трансферів тощо.

Однак, є декілька недоліків таких систем. Перш за все, вони зазвичай дуже дорогі і складні у використанні. Крім того, вони зазвичай зосереджені на аналізі матчів та статистичних даних, а не на управлінні волейбольними клубами і організацією турнірів.

У порівнянні з конкуруючими системами, мій проєкт має головний плюс це простота виконання та отриманий результат, який уже дає змогу роботи користувачам. З іншого - додаток більш зосереджений на управлінні волейбольними клубами і організацією турнірів. Він забезпечує зручний інтерфейс та функціональність для додавання, редагування та видалення даних про команди, стадіони, матчі, гравців і чемпіонати. Мій проєкт відрізняється від інших систем тим, що він не тільки забезпечує ведення статистики матчів та команд, але також інтегрує усі ці дані в один веб-додаток. Він також містить інформацію про гравців та стадіони, що дозволяє клубам та менеджерам більш повноцінно аналізувати гру своїх команд. Крім того, він дозволяє проводити різні типи турнірів і чемпіонатів, що дає можливість користувачам детально налаштувати їх параметри.

Крім того, мій проєкт використовує технології ASP.NET Core та C#, що забезпечує більші можливості для розширення та покращення функціоналу в майбутньому. Мій проєкт також може бути доступним для різних платформ, таких як Windows, Linux та Mac OS, що дає можливість користувачам вибрати оптимальну для них платформу.

Загалом, мій проєкт має багато переваг порівняно перед такими застосунками як ESPN або Soccerway. Зокрема, мій проєкт пропонує більше можливостей для користувачів, швидкість та ефективність роботи, відкритий вихідний код та можливості для розширення та покращення функціоналу.



## Розділ 2. Огляд використаних технологій

**Мова програмування. C#** - це сучасна об'єктно-орієнтована мова програмування, розроблена компанією Microsoft. Вона була створена з метою розробки програмного забезпечення для платформи .NET, а також для розробки застосунків на платформі Windows. Синтаксис C# близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато від своїх попередників — мов C++, Object Pascal, Модула і Smalltalk — C#, спираючись на практику їхнього використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, мова C#, на відміну від C++, не передбачає множинне успадкування класів.

### Переваги C#:

- Об'єктно-орієнтований підхід дозволяє розробляти складні програми з великою кількістю функцій і класів.
- Має сучасну синтаксичну конструкцію, що дозволяє писати чистий і лаконічний код, що підвищує читабельність і розуміння коду.
- C# має дуже велику кількість вбудованих бібліотек, що забезпечує широкий спектр можливостей для розробки різноманітних додатків.
- Мова має вбудовану підтримку мульти потокового програмування, що дозволяє писати більш швидкодіючі та ефективні застосунки.
- Розробники C# можуть використовувати Visual Studio, інтегроване середовище розробки, що забезпечує зручний і продуктивний процес розробки.

### **Недоліки C#:**

- Платформо-залежність може стати проблемою, якщо потрібно розробляти застосунки на іншій платформі.
- В деяких випадках розробка на C# може бути більш складною та тривалою, особливо для початківців, порівняно з іншими мовами програмування.

### **Особливості використання C# у веб-розробці:**

- C# є основною мовою програмування для розробки веб-додатків на платформі ASP.NET.
- Завдяки вбудованій підтримці мульти потокового програмування, C# є особливо підходящою мовою для написання серверної логіки веб-додатків [5].

Отже, C# є дуже гарним вибором для виконання цієї лабораторної роботи, так як містить потрібну кількість бібліотек і технологій, що забезпечують легшу розробку, безпечне функціонування застосунку та шляхи розширення в майбутньому.

**Фреймворк.** ASP.NET є фреймворком веб-розробки, розробленим компанією Microsoft. Він дозволяє розробникам створювати динамічні веб-сторінки, використовуючи мови програмування .NET, такі як C# та VB.NET. ASP.NET включає в себе велику кількість компонентів та інструментів для підтримки розробки веб-додатків, в тому числі бібліотеки класів, вбудовані контролери та інші інструменти для створення динамічного контенту на веб-сторінках [1].

### **Переваги ASP.NET:**

- Безпека: ASP.NET включає в себе різноманітні заходи безпеки для захисту веб-додатків від атак, таких як SQL-ін'єкції та XSS.
- Швидкість: завдяки своїй архітектурі та кешуванню, ASP.NET може обробляти великий обсяг запитів швидко та ефективно.

- Масштабованість: ASP.NET розроблено з урахуванням потреб в масштабуванні веб-додатків, що дозволяє розробникам легко масштабувати додатки, якщо це необхідно.
- Інструменти розробки: ASP.NET має велику кількість інструментів розробки, таких як Visual Studio, що дозволяє розробникам створювати додатки швидко та ефективно.
- Кросплатформеність: з випуском .NET Core, ASP.NET став кросплатформовим фреймворком, що дозволяє розробникам розробляти веб-додатки на Windows, macOS та Linux.

#### **Недоліки ASP.NET:**

- Вартість: для використання деяких функцій ASP.NET, таких як Visual Studio, потрібно платити.
- Велика складність: через велику кількість компонентів та інструментів, які включені в ASP.NET, розробка веб-додатків може бути дещо складною та вимагати більше часу та зусиль для навчання. Наприклад, використання ASP.NET потребує знання C# або іншої мови програмування .NET, а також знання HTML, CSS та JavaScript для фронтенд-розробки.
- Залежність від Microsoft та замкнутість: ASP.NET є фреймворком, створеним компанією Microsoft, тому він може бути залежним від інших продуктів Microsoft. Крім того, певні аспекти фреймворку можуть бути замкненими, що обмежує можливості вільного розширення та модифікації [2].

Отже, хоч і цей фреймворк має певну кількість недоліків, він дуже добре підходить для реалізації задуманого проекту, в силу швидкості та можливості розширення ми маємо проект який уже готовий до використання та є дуже перспективним в майбутньому завдяки інструментам ASP.NET, які допоможуть його розширити у всіх напрямках.

**Система управління базами даних.** Веб-ресурси містять величезну кількість даних від облікових записів користувачів до контенту, опублікованого на сторінках. Те саме стосується «хмарних» додатків на кшталт CRM, програм для бухгалтерії, складського обліку та ін. Скрізь використовується один спосіб зберігання інформації – база даних. І цією базою потрібно якось керувати. В нашому випадку була використана PostgreSQL. Її ми і розглянемо.

**PostgreSQL Server** - — об'єктно-реляційна система керування базами даних (СКБД). Є альтернативою як комерційним СКБД (Oracle Database, Microsoft SQL Server, IBM DB2 та інші), так і СКБД з відкритим кодом (MySQL, Firebird, SQLite). Порівняно з іншими проектами з відкритим кодом, такими як Apache, FreeBSD або MySQL, PostgreSQL не контролюється якоюсь однією компанією, її розробка можлива завдяки співпраці багатьох людей та компаній, які хочуть використовувати цю СКБД та впроваджувати у неї найновіші досягнення. Сервер PostgreSQL написаний на мові С. Зазвичай розповсюджується у вигляді набору текстових файлів із початковим кодом.

**SQL Server** — це основа платформи обробки даних Microsoft, яка надає надійну та стійку продуктивність (у тому числі завдяки технологіям обробки даних у пам'яті) та допомагає швидше отримати цінну інформацію з будь-яких даних, розташованих як у локальному середовищі, так і в хмарі.

#### **Переваги PostgreSQL:**

- Потужні та надійні механізми транзакцій та реплікації;
- розширювана система вбудованих мов програмування та підтримка завантаження C-сумісних модулів;
- Успадкування;
- Легка розширюваність.

#### **Недоліки PostgreSQL:**

- У найпростіших операціях читання продуктивність цієї СУБД поступається іншим;
- Також недоліком можна вважати складність системи, її налаштування.

Отже, в нашому випадку ця база даних нам ідеально підходить, для повного функціонування проекту. Завдяки гарним настановам лектора налаштування не викликало проблем, тому цей недолік не вплинув на кінцевий результат роботи.

## **Розділ 3. Призначення і цілі створення системи**

### **3.1 Призначення системи**

Призначенням проекту "Волейбольні чемпіонати" є автоматизація процесу будь-якої взаємодії з інформацією пов'язаною з волейбольними чемпіонатами, матчами, командами та їх гравцями і стадіонами, що дає можливість забезпечити швидкий доступ до інформації вболівальникам, та легку взаємодію з базою для організаторів.

Лабораторна робота передбачає:

- аналіз методів, методик і моделей, що застосовуються для розв'язання задач відображення та поновлення інформації по волейбольним заходам;
- аналіз наявних програмних засобів в галузі спортивного менеджменту та автоматизації організації спортивних заходів;
- проектування та програмну реалізацію системи "Волейбольні чемпіонати", яка буде забезпечувати зручний та ефективний спосіб управління інформації про проведенні матчі, нові команди та гравців, зміни в турнірній таблиці та результативність команд;
- апробацію розробленої системи "Волейбольні чемпіонати" на реальних волейбольних заходах для оцінки її ефективності та можливості вдосконалення.

### **3.2 Цілі створення системи**

**Система "Волейбольні чемпіонати" створюється з метою:**

- забезпечення збору, обробки та аналізу інформації про волейбольні чемпіонати та команди, що в них беруть участь;
- підвищення ефективності поширення інформації про останні футбольні новини для вболівальників;

- можливість проаналізувати статистику гравців та команд для підвищення їх ефективності та визначення найкращих результатів.

Також система призначена для забезпечення доступу користувачів до актуальної та достовірної інформації про волейбольні змагання та команди, а також для автоматизації процесу підготовки та проведення змагань.

Наведемо USE-CASE діаграма проекту ([див. додаток А.](#))

### **3.3. Вимоги до системи**

#### **3.3.1 Вимоги до системи в цілому**

Система "Волейбольні чемпіонати" повинна відповідати наступним вимогам до структури та функціонування:

1. Бути доступною для користувачів з різних пристроїв та операційних систем.
2. Забезпечувати збір та збереження інформації про волейбольні матчі, команди, гравців, розклади та результати чемпіонатів.
3. Мати можливість редагування даних про команди, гравців, матчі та результати відповідальними особами.
4. Забезпечувати можливість перегляду інформації про матчі, команди та гравців для користувачів.
5. Бути стійкою до витоку даних та використання системи від несанкціонованих осіб.
6. Мати можливість генерувати звіти та статистику про результати чемпіонатів.
7. Бути простою та зрозумілою для використання користувачами.

В системі передбачається виділити наступні функціональні підсистеми:

- Адміністративна підсистема, призначена для управління даними про футбольні матчі, команди, гравців, розклади та результати чемпіонатів.
- Підсистема користувача, призначена для перегляду інформації про матчі, команди та гравців, пошуку інформації, генерації звітів та статистики.

### 3.3.2. Вимоги до функцій, які виконуються системою

**Таблиця 1. Підсистема адміністратора.**

Функція	Задача
Керування роботою користувачів системи	Реєстрацію нових користувачів, редагування, видалення та блокування аккаунтів користувачів
Керування підсистемою користувачів	Створення, редагування та видалення користувачів, налаштування їх прав доступу та відстеження їх дій в системі
Управління базою даних	Створення, редагування та видалення інформації в таблицях. Доступ до перегляду діаграм.
Робота з файлом	Можливість експорту та імпорту даних файла формату .xlsx(EXCEL таблиця)

**Таблиця 2. Підсистема користувача**

Функція	Задача
Вхід до системи	Аутентифікація та авторизація користувача в системі



Перегляд усієї доступної інформації	Надання користувачеві можливості переглядати інформацію про команди та гравців, їх статистику та іншу важливу інформацію
Експорт в файл	Можливість експорту даних в файл формату .xlsx(EXCEL таблиця)

### 3.4 Технічні вимоги

Браузери: Сайт повинен коректно відображатися в інтернет-браузерах MS Internet Explorer 5.5 і вище, Mozilla 1.7 і вище, Opera 7.54 і вище.

На довільні некоректні дії користувача, пов'язані з введенням невірних даних, не заповненням обов'язкових полів введення в формах та інші, які можуть бути оброблені системою, генеруються відповідні повідомлення про помилки англійською мовою, в межах загального дизайну сайту.

Операційна система: WINDOWS 7, 8, 10, 11

Джерелом даних для Системи повинна бути інформаційна система SQL Server.

## Розділ 4. Опис організації інформаційної бази

### 4.1. Логічна структура бази даних

Збереженням даних для проекту займається інформаційна система SQL Server від Microsoft.

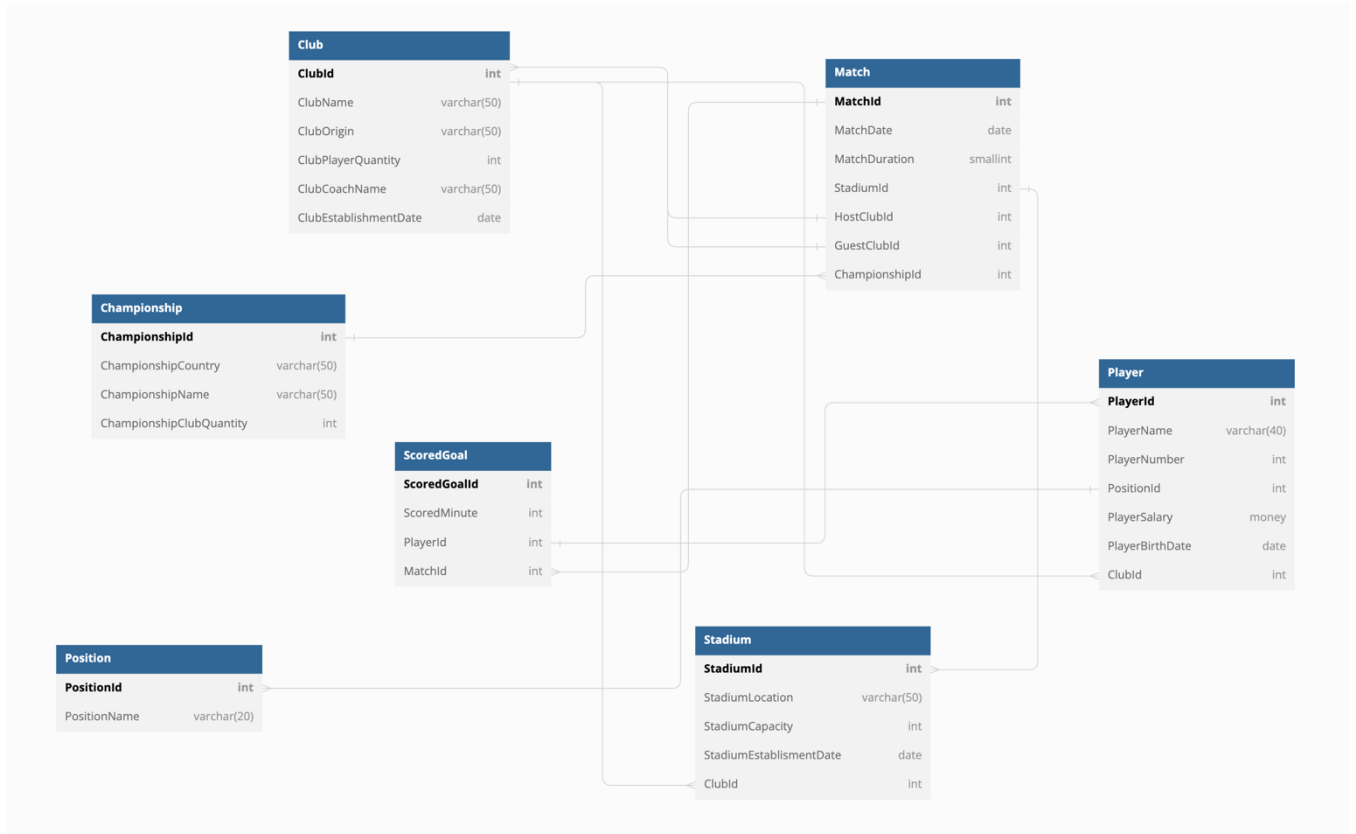


Рисунок 2 – Діаграма бази даних «PostgreSQL»

Наведу перелік усіх таблиць та їх призначення для більшого розуміння задумки проекту

Таблиця 3. Опис бази даних

Номер	Таблиця	Опис
1	Championship	Таблиця для збереження всіх чемпіонатів, які поділяють проведення матчів на різні розділи
2	Match	Таблиця для збереження всієї статистики проведеного матчу
3	Stadium	Таблиця для збереження інформації про

		стадіон певної команди
4	Club	Таблиця для збереження всієї інформації пов'язаної з командою
5	Player	Таблиця для збереження всієї інформації пов'язаної з гравцем
6	Position	Таблиця, яка зберігає назви можливих позицій для гравця
7	ScoredGoal	Проміжна таблиця, утворена для збереження інформації про голи в матчі

#### 4.2. Опис таблиць

Нижче описані усі поля відповідних таблиць для розуміння їх призначення

**Таблиця 4. Опис класу “Championship”**

Атрибут	Тип	Опис
ChampionshipId	integer	Ідентифікатор чемпіонату
ChampionshipCountry	nvarchar(50)	Країна чемпіонату
ChampionshipName	nvarchar(50)	Назва чемпіонату

**Таблиця 5. Опис класу “Match”**

Атрибут	Тип	Опис
MatchId	integer	Ідентифікатор матчу
MatchDate	date	Дата проведення матчу
MatchDuration	integer	Тривалість матчу
StadiumId	integer	Ідентифікатор стадіону на якому проводився матч
HostClubId	integer	Ідентифікатор команди хазяїв
GuestClubId	integer	Ідентифікатор команди гостей
ChampionshipId	integer	Ідентифікатор чемпіонату до якого відноситься матч

**Таблиця 6. Опис класу “Stadium”**

Атрибут	Тип	Опис
StadiumId	integer	Ідентифікатор стадіону
StadiumLoaction	nvarchar(50)	Місцезнаходження стадіону
StadiumCapacity	integer	Місткість стадіону
StadiumEstablishment Date	date	Дата заснування стадіону
ClubId	integer	Команда власник стадіону

**Таблиця 7. Опис класу “Club”**

Атрибут	Тип	Опис
ClubId	integer	Ідентифікатор команди
ClubName	nvarchar(50)	Назва команди
ClubOrigin	nvarchar(50)	Походження команди
ClubPlayerQuantity	integer	Кількість гравців в команді
ClubCoachName	nvarchar(50)	Головний тренер команди
ClubEstablishmentDate	date	Дата заснування команди

**Таблиця 8. Опис класу “Player”**

Атрибут	Тип	Опис
PlayerId	integer	Ідентифікатор гравця
PlayerName	nvarchar(40)	Ім'я гравця
PlayerNumber	integer	Ігровий номер гравця
PositionId	integer	Ідентифікатор позиції на якій гравець грає
PlayerSalary	money	Заробітна плата гравця
PlayerBirthDate	date	Дата народження гравця

ClubId	date	Ідентифікатор команди за яку гравець виступає
--------	------	---

**Таблиця 9. Опис класу “Position”**

Атрибут	Тип	Опис
PositionId	integer	Ідентифікатор позиції
Position	nvarchar(20)	Назва позиції

**Таблиця 10. Опис класу “ScoredGoal”**

Атрибут	Тип	Опис
ScoredGoalId	integer	Ідентифікатор забитого гола
ScoredMinute	integer	Хвилина забитого гола
PlayerId	integer	Ідентифікатор гравця який відзначився
MatchId	integer	Ідентифікатор матчу в якому був забитий гол

## Розділ 5. Реалізація системи

Model-View-Controller (MVC) - це популярний підхід до розробки веб-додатків, який дозволяє розділити логіку додатку на три взаємодіючі компоненти: модель, представлення та контролер.

**Загальна структура ASP.NET додатку включає наступні компоненти:**

- **Модель даних:** Відображає структуру даних, які будуть використовуватись в додатку. Модель може бути створена за допомогою Entity Framework або вручну.
- **Контролери:** Оброблюють запити від користувача і взаємодіють з моделлю та представленням. Контролери можуть мати багато дій, кожна з яких відповідає за обробку конкретного запиту.
- **Представлення:** Відображає дані на веб-сторінках і забезпечує інтерфейс для взаємодії з користувачем. Представлення може бути створене за допомогою HTML, CSS та JavaScript.
- **Маршрутизатор:** Відповідає за маршрутизацію запитів до правильного контролера та дій.
- **Пакети NuGet:** Містять додаткові бібліотеки, які можна використовувати в додатку. Наприклад, бібліотеки для роботи з базами даних або для розробки інтерфейсу користувача.

Ці компоненти співпрацюють, щоб забезпечити роботу додатку та його взаємодію з користувачем через веб-браузер. В результаті створюється динамічний та масштабований веб-додаток, який може виконувати різні завдання в залежності від потреб користувачів.

Модель відповідає за представлення бізнес-логіки додатку та містить у собі всю необхідну інформацію про даний додаток. Цей компонент містить у собі дані та методи для їх обробки.

Представлення відповідає за те, як дані представлені користувачам. Цей компонент генерує відповідні HTML-сторінки, які користувач може бачити та з якими він може взаємодіяти.

Контролер відповідає за обробку запитів користувачів та взаємодію з моделлю та представленням. Цей компонент обробляє запити, взаємодіє з моделлю для отримання необхідних даних та передає їх до представлення для відображення користувачу.

Основна ідея підходу MVC полягає в тому, що кожен компонент має відповідальність лише за свої власні функції, що дозволяє забезпечити більшу модульність, розширюваність та тестованість додатку. Крім того, підхід MVC дозволяє зменшити залежність між різними компонентами додатку, що полегшує його розробку та підтримку.

MVC є досить популярним підходом до розробки веб-додатків та використовується в багатьох веб-фреймворках, включаючи ASP.NET MVC, Ruby on Rails та Django.

Якщо виразити своїми словами, при старті роботи програми, головний файл Program.cs посилається на велику кількість методів для правильного функціонування і відображення веб застосунку, проте найбільше нас цікавить момент зазначений на скріншоті знизу.

```
var builder = WebApplication.CreateBuilder(args);  
  
// Add services to the container.  
builder.Services.AddControllersWithViews();  
  
builder.Services.AddDbContext<FootballBdContext>(option => option.UseSqlServer(  
    builder.Configuration.GetConnectionString("DefaultConnection")  
));  
builder.Services.AddControllersWithViews();  
  
builder.Services.AddDbContext<IdentityContext>(option => option.UseSqlServer(  
    builder.Configuration.GetConnectionString("IdentityConnection")  
));  
builder.Services.AddControllersWithViews();  
  
builder.Services.AddIdentity<User, IdentityRole>().AddEntityFrameworkStores<IdentityContext>();  
  
var app = builder.Build();
```

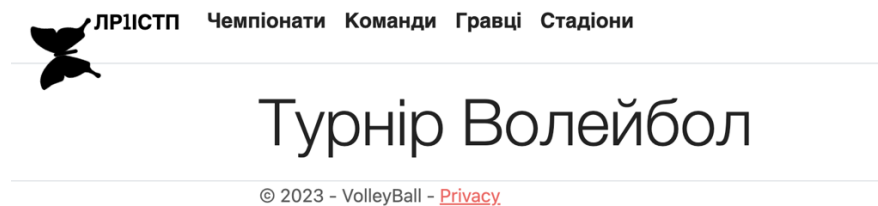
Рисунок 3 – код з файлу Program.cs

Це є відправною точкою до основної роботи програми, які представлені у всьому звіті. Після додавання наших контекстів, вони, в свою чергу, запускають роботу усіх контролерів, які використовують



існуючі моделі. Початковий екран буде відображено відповідно до вказаного в файлі `_Layout`, в нашому випадку ми попадемо на сторінку авторизації користувача. Усе це доповнюється підключенням до бази даних і копіюванням інформації на наші екрани. Після того як база підключена, вся система повністю готова до роботи.

Коли програма повністю запущена, вона очікує будь яку дію зі сторони користувача. Покажу наглядно процес роботи програми.



**Рисунок 4 – Початкова сторінка**

Користувач натискає кнопку чемпіонати. Програма реєструє ввід користувача і працює відповідно до вказаних розробником дій.

```
<li class="nav-item">
  <a class="nav-link text-dark" asp-area="" asp-controller="Championships" asp-action="Index">Чемпіонати</a>
</li>
```

**Рисунок 5 – Код з файлу `_Layout.cshtml`**

А саме в файлі `_Layout` натята кнопка активує метод `Index` контролеру `Championships`.

```
public async Task<IActionResult> Index()
{
    return View(await _context.Championships.ToListAsync());
}
```

**Рисунок 6 – Метод `Index` контролер `Championship`**

Метод індекст вертає `View`, яке в параметрах має контекст Чемпіонату, який зводиться до типу `List`.

Детальніше про контекст. Це `partial class`, який зберігає типи відповідно до моделей

```
23 references
public virtual DbSet<Championship> Championships { get; set; }
```

**Рисунок 7 – Тип `Championship` в класі `FootballDbContext`**

У прикладі використано тип Championship. Можна переглянути його детально.

```
20 references
public partial class Championship
{
    16 references
    public int ChampionshipId { get; set; }
    [Display(Name = "Країна")]
    15 references
    public string ChampionshipCountry { get; set; }
    [Display(Name = "Назва")]
    23 references
    public string ChampionshipName { get; set; }
    [Display(Name = "Кількість команд")]
    [Range(5, 50, ErrorMessage = "Чемпіонат може містити від 5 до 50 команд")]
    12 references
    public int ChampionshipClubQuantity { get; set; }

    3 references
    public virtual ICollection<Match> Matches { get; } = new List<Match>();
}
```

Рисунок 8 – Клас Championship

Він знаходиться в папці з моделями і містить усі поля, які зберігає відповідна таблиця у нашій базі даних. Ще в ньому присутні деякі обмеження до відповідного поля, як от до кількості команд в чемпіонаті, яка не може бути менше 5 і не може перевищувати кількість в 50 команд. Тепер можна повернутися до методу Index, він вертає вигляд нашої сторінки, який зберігається в файлі Views -> Championships-> Index.cshtml.

Саме тут реалізовано вигляд усієї сторінки.

```
@foreach (var item in Model) {
    <tr>
        <td>
            @Html.DisplayFor(modelItem => item.ChampionshipCountry)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.ChampionshipName)
        </td>
        <td>
            @Html.DisplayFor(modelItem => item.ChampionshipClubQuantity)
        </td>
        <td>
            <a asp-action="Details" class="btn btn-primary mt-1 mb-1" style="font-size:12px; padding:5px 7px; border-radius: 10px" asp-route-
            @if (User.IsInRole("admin"))
            {
                <a asp-action="Edit" class="btn btn-info mt-1 mb-1" style="font-size:12px; padding:5px 7px; border-radius: 10px" asp-route-id="@item.C
            }
            <a asp-action="Delete" class="btn btn-danger mt-1 mb-1" style="font-size:12px; padding:5px 7px; border-radius: 10px" asp-route-id="@item.C
            }
        </td>
    </tr>
}
```

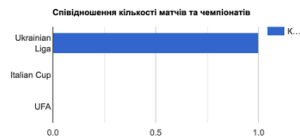
Рисунок 9 – Шматок коду з файлу Index.cshtml

На скріншоті вище видно перелік усіх полів, які беруться з бази даних і виводяться на сторінку користувачу.

## Чемпіонати

Додати новий чемпіонат

Країна	Назва	Кількість команд	
Italy	Italian Cup	8	<a href="#">Матчі</a> <a href="#">Редагувати</a> <a href="#">Видалити</a>
Ukraine	Ukrainian Liga	6	<a href="#">Матчі</a> <a href="#">Редагувати</a> <a href="#">Видалити</a>
France	UFA	8	<a href="#">Матчі</a> <a href="#">Редагувати</a> <a href="#">Видалити</a>



© 2023 - VolleyBall - [Privacy](#)

### Рисунок 10 – Сторінка з чемпіонатами

Також, на картинці видно відображення діаграми, яка збирає інформацію з бази даних і представляє її в такому форматі, для кращої орієнтації користувача в застосунку.

В цьому описі була наведена базова робота програми на прикладі старту роботи програми і звернення до таблиці Championship. Інші таблиці містять схожий функціонал, проте через їх пов'язаність складність систем збільшується, тому покрокова робота програми може бути довшою відповідно до роботи методів, які були прописані розробником. Після авторизації, користувач буде мати права юзера тобто зможе тільки переглядати наявну інформацію. Зверху присутній навбар для кращої орієнтації в застосунку.



Перейдемо на сторінку з інформацією про всі команди

# Команди

Оберіть будь ласка файл для завантаження

Завантажити Excel-файл:  Файл не выбран

Завантажити інформацію про команди у Excel

Назва	Походження	Кількість гравців	Тренер	Дата заснування	
Manchester City	Great Britian	5	Gvardiola	12.03.1894	<input type="button" value="Список гравців"/> <input type="button" value="Редагувати"/> <input type="button" value="Видалити"/>
Olympiacos	Greece	4	Daniel Castellani	18.10.1925	<input type="button" value="Список гравців"/> <input type="button" value="Редагувати"/> <input type="button" value="Видалити"/>

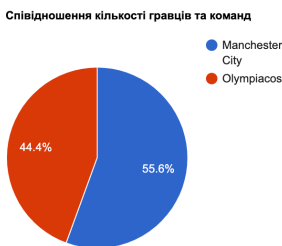


Рисунок 11 – Сторінка з командами.

## КОМАНДИ

### ОБЕРІТЬ ФАЙЛ ДЛЯ ЗАВАНТАЖЕННЯ

Завантажити Excel-файл:  Файл не вибран

### ЗАВАНТАЖИТИ ІНФОРМАЦІЮ ПРО КОМАНДИ

НАЗВА	ПОХОДЖЕННЯ	КІЛЬКІСТЬ ГРАВЦІВ	ТРЕНЕР	ДАТА ЗАСНУВАННЯ	
club4	origin4	1	coach4	28.02.2023	<input type="button" value="СПИСОК ГРАВЦІВ"/> <input type="button" value="РЕДАГУВАТИ"/> <input type="button" value="ВИДАЛИТИ"/>
club6	origin6	3	coach6	27.03.2022	<input type="button" value="СПИСОК ГРАВЦІВ"/> <input type="button" value="РЕДАГУВАТИ"/> <input type="button" value="ВИДАЛИТИ"/>
club7	origin5	1	coach7	28.02.2023	<input type="button" value="СПИСОК ГРАВЦІВ"/> <input type="button" value="РЕДАГУВАТИ"/>

Рисунок 12 – Сторінка з командами, вигляд користувача

Наглядно видно різницю, користувач має право на редагування, видалення, додавання та імпорт файлу.

### СТВОРЕННЯ КОМАНДИ

Назва

Походження

Тренер

Дата заснування

### РЕДАГУВАННЯ КОМАНД CLUB

Назва

Походження

Тренер


Дата заснування

Рисунок 13 – Додавання нової команди. Редагування існуючої

Функціонал доволі простий та зрозумілий. У випадку редагування, просто виправляємо потрібні поля і зберігаємо дані. У випадку додавання вводимо дані до відповідних полей, після цього натискаємо “**Перейти до створення стадіону**”, там нас очікує така сама сторінка, тільки з полями, які властиві для стадіону. Саме така система запроваджена, через те що команда не може існувати без стадіону. При додаванні деяких інших таблиць, можна зустріти схожу механіку. Вона впроваджена мною через неможливість існування одних полей без інших, це робить роботу програми більш безпечно і стабільною. Також потрібно зазначити що як при створенні, так і при редагування і імпорту файлу, всі поля повинні пройти валідацію прописану розробником. Наприклад при додаванні матчу, дата його проведення не може передувати даті створення команди, або при додаванні команди в чемпіонат, вона не може брати участь в інших чемпіонатах.

## ДОДАВАННЯ МАТЧУ

Дата матчу

08.02.2018 

Дата проведення матчу не може передувати даті створення команд

Тривалість

96

Команда хазяїв

club4

Команда гостей


club6

СТВОРИТИ

НАЗАД ДО СПИСКУ

## ДОДАВАННЯ МАТЧУ

Дата матчу

02.05.2023 

Тривалість

98

Вказана команда уже бере участь в іншому чемпіонаті

Команда хазяїв

club7

Команда гостей

club4

СТВОРИТИ

НАЗАД ДО СПИСКУ

Рисунок 14 – Помилка валідації даних

Об'єм валідації набагато більший ніж вказано вище, але не має сенсу перечисляти усе, в ході роботи користувач зрозуміє усе сам.

Імпорт даних виглядає наступним чином. Користувач натискає кнопку **“Виберіть файл”**, після цього відкривається вікно провідника, де він може вибрати відповідний файл (Якщо розширення файлу не відповідає розширенню .xlsx, то файл не буде розпізнано і він не пройде валідацію). Після того як файл вибрано, натискаємо **“Завантажити”**

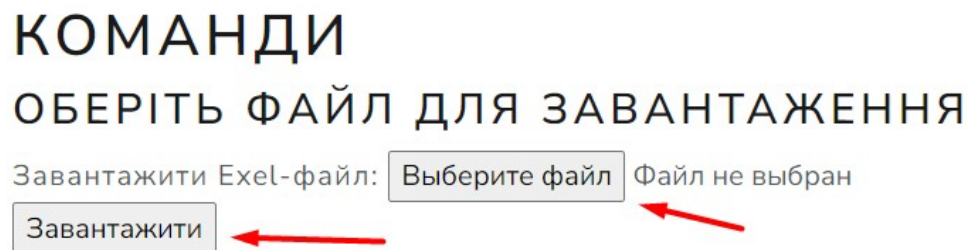


Рисунок 15 – Вкладка завантажити

Переглянемо всі можливості юзера, які доступні для нього. Отже юзеру доступні вкладки, які містять деталі про поле на сторінці якого він перебуває. Наприклад він може переглянути **“Список гравців”** відповідної команди.

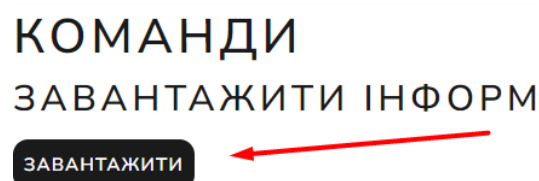
НАЗВА	ПОХОДЖЕННЯ	КІЛЬКІСТЬ ГРАВЦІВ	ТРЕНЕР	ДАТА ЗАСНУВАННЯ	
club4	origin4	1	coach4	28.02.2023	<b>СПИСОК ГРАВЦІВ</b>

Рисунок 16 – Кнопка “Список гравців”

ГРАВЕЦЬ КОМАНДИ CLUB4					
ІМ'Я	НОМЕР	ПОЗИЦІЯ	ЗАРПЛАТА	РІК НАРОДЖЕННЯ	КОМАНДА
player4	32	ПівЗахисник	3123,00	28.11.2004	club4

Рисунок 17 – Сторінка списку гравців команди

Також юзер може експортувати усю наявну інформацію в файл з розширенням .xlsx, для збереження в офлайн форматі і перегляду інформації.





**Рисунок 18 – Кнопка “Экспорту” в файл**

A1															
ClubId															
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	ClubId	ClubName	ClubOrigin	ClubCoach	ClubEstabl	StadiumLo	StadiumCa	StadiumEs	PlayerNam	PlayerNum	PositionN	PlayerSala	PlayerBirthDate		
2	88	club4	origin4	coach4	28.02.2021	location4	31242	28.02.2021	player4	32	Півзахисн	3123,0000	28.11.2004		
3	90	club6	origin6	coach6	27.03.2021	location6	13241	16.02.2021	player6,pl	21,32,51	Воротар	23141,000	17.02.2004,17.06.2004,24.06.2004		
4	91	club7	origin5	coach7	28.02.2021	location7	1232	20.03.2021	player7	2	Воротар	7575,0000	08.02.2005		

**Рисунок 19 – Видяг експортованого файлу**

Також важливо зазначити, що будь який користувач має діаграму, яка допомагає йому підсумувати інформацію збережену в базі.

Співвідношення кількості гравців та команд

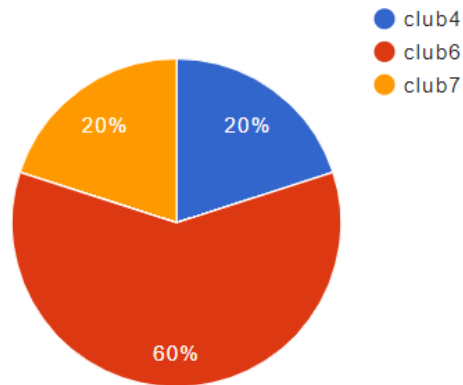


Рисунок 20 – Вигляд експортованого файлу

Отже, вище був описаний весь головний базовий функціонал програми на прикладі сторінки з командами. Робота з усіма іншими сторінками схожа і не має кардинальних відмінностей, тому усі інші тонкощі користувач може вивчити під час роботи з веб застосунком. А завдяки простоті і вузько направленості програми, це не буде складною задачею.

## **Висновки**

У ході розробки проекту були проведені аналіз потреб користувачів, визначені функціональні та нефункціональні вимоги до системи, розроблена структура та функціональність системи з використанням підходу Model-View-Controller (MVC).

Проект успішно реалізував функції збору, обробки та аналізу інформації про футбольні чемпіонати, а також забезпечив можливість проаналізувати результати та статистику чемпіонатів. Були розроблені функціональні підсистеми для адміністратора та користувача, що забезпечують можливість редагування та видалення інформації про користувачів, формування та візуалізацію звітів про користувачів системи, а також можливість введення інформації про себе та перегляду результатів чемпіонатів.

За результатами роботи над проектом можна зробити висновок, що використання підходу Model-View-Controller (MVC) дозволило ефективно розробити функціональність та структуру системи, що відповідає вимогам користувачів та може бути успішно застосована в практичних цілях. Для подальшого розвитку проекту можна розглянути можливість додавання нових функцій та модулів, покращення інтерфейсу користувача та оптимізацію швидкості роботи системи.

Підсумовуючи усе вище сказане, я можу сказати, що навчився базовим аспектам роботи з таким фреймворком як ASP.NET. Всі ці знання були застосовані при розробці власного проекту який уже готовий до роботи і має не дуже поширений, проте точний функціонал. Він уже готовий до використання іншими людьми для завантаження і перегляду інформації. Проект має усе для розширення та більш стійкого функціоналу в майбутньому

## Список використаних джерел

1. Microsoft. (2021). About ASP.NET Core [Електронний ресурс] - Режим доступу до ресурсу: <https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-5.0>
2. Venkatesan, V. (2019). Advantages and Disadvantages of ASP.NET [Електронний ресурс] - Режим доступу до ресурсу: <https://www.c-sharpcorner.com/article/advantages-and-disadvantages-of-asp-net/>
3. " PostgreSQL: a closer look at the object-relational database management system" Режим доступу до ресурсу: <https://www.ionos.com/digitalguide/server/know-how/postgresql/>
4. "C Sharp" Wikipedia [Електронний ресурс] - режим доступу до ресурсу: [https://uk.wikipedia.org/wiki/C\\_Sharp](https://uk.wikipedia.org/wiki/C_Sharp)

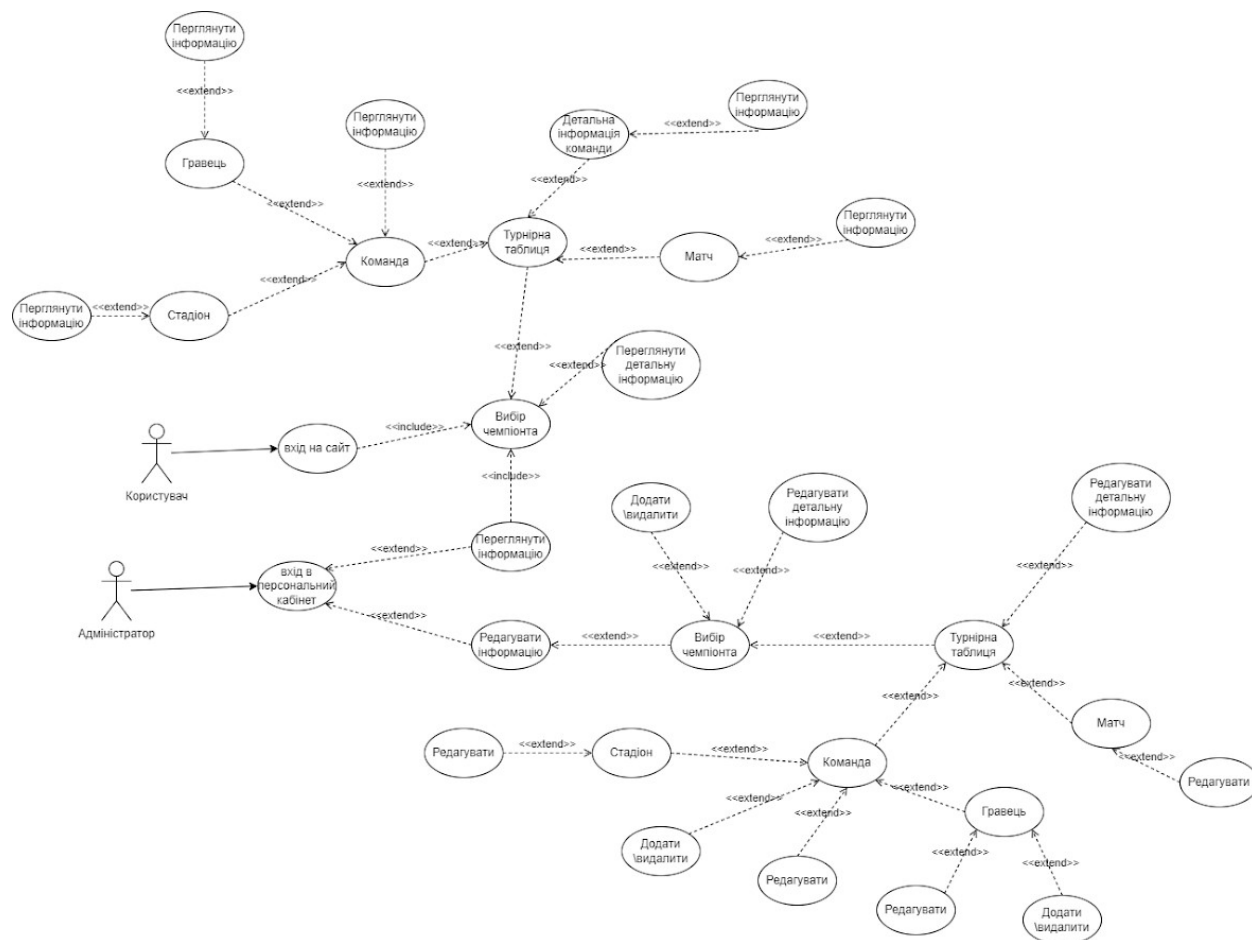


Рисунок 1 – USE-CASE діаграма