Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)
бакалавра
(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СЛІДКУВАННЯ ЗА БАСКЕТБОЛЬНИМИ ЗМАГАННЯМИ
SOFTWARE FOR MONITORING BASKETBALL COMPETITIONS

Виконав: студент(ка) 4 курсу, групи КНТ-217
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Морозов В.М.
(прізвище та ініціали)

Керівник Колпакова Т.О.
(прізвище та ініціали)

Рецензент Голуб Т.В.
(прізвище та ініціали)

2021

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІІРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____С.О. Субботін
"____" _____2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Морозова Владислава Михайловича
(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення слідкування за баскетбольними змаганнями
Software for Monitoring Basketball Competitions

керівник проєкту (роботи) Колпакова Тетяна Олексіївна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від "30" березня 2021 року № 103
2. Строк подання студентом проєкту (роботи) 17.05.2021
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Розробка програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Колпакова Т.О., доцент		
Нормоконтролер	Андреєв М.О., асистент		

7. Дата видачі завдання “ 17 ” березня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5-6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи	9 тиждень	

Студент(ка) _____ Морозов В.М.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи) _____ Колпакова Т.О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 80 с., 26 рис., 2 табл., 3 дод., 12 джерел.

БАСКЕТБОЛЬНІ ЗМАГАННЯ, БАСКЕТБОЛЬНІ КОМАНДИ, ПОКАЗНИКИ ГРАВЦІВ, ВЕБЗАСТОСУНОК, БАЗА ДАНИХ, ASP.NET.

Об’єкт дослідження – процес накопичення результатів баскетбольних змагань. Предмет дослідження – програмне забезпечення слідкування за баскетбольними змаганнями.

Мета роботи – розроблення програмного забезпечення слідкування за баскетбольними змаганнями для забезпечення індивідуальної траєкторії такого слідкування.

Матеріали, методи та технічні засоби: мова програмування C#, технологія розробки вебзастосунків ASP.NET, система керування базами даних Microsoft SQL Server.

Результати. Виконано проєктування програмного забезпечення, включаючи моделювання роботи адміністратора, авторизованого і неавторизованого користувачів з програмою, проєктування бази даних, визначення архітектури програми. Виконано реалізацію програмного забезпеченння, яке відрізняється від існуючих аналогів можливістю індивідуалізації перегляду показників виступу гравців і команд.

Висновки. Створений вебзастосунок надає можливість перегляду результатів баскетбольних змагань, окремих матчів, статистичних показників виступу гравців у цих матчах, узагальнених показників гравців та команд за виступ у сезоні турніру, організовувати індивідуальну траєкторію перегляду результатів виступу команд і гравців, створюючи переліки уподобаних гравців, команд і змагань.

Галузь використання. Інформаційні бази баскетбольних змагань.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 80 pages, 26 figures, 2 tables, 3 appendixes, 12 sources.

BASKETBALL COMPETITIONS, BASKETBALL TEAMS, INDICATORS OF PLAYERS, WEB APPLICATION, DATABASE, ASP.NET.

The object of research is the process of accumulating the results of basketball competitions. The subject of research is software for monitoring basketball competitions.

The purpose of the work is to develop software for monitoring basketball competitions to ensure the individual trajectory of such monitoring.

Materials, methods and technical means: C# programming language, ASP.NET web application development technology, Microsoft SQL Server database management system.

Results. Software design was performed, including modeling of administrator, authorized and unauthorized users with software, database design, definition of program architecture. The implementation of software that differs from existing analogues by the ability to individualize the performance of players and teams was performed.

Conclusions. The created web application provides an opportunity to view the results of basketball competitions, individual matches, statistics of player performance in these matches, generalized indicators of players and teams for performance in the tournament season, organize individual trajectory of monitoring of teams and players, creating lists of favorite players and teams.

Field of use. Information databases of basketball competitions.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

NBA – National Basketball Association;

PER – Player Efficiency Rating;

PPP – Points Per Possession.

ВСТУП

За оцінками [1] баскетбол – третій за популярністю вид спорту в світі на даний момент. Кількість фанатів оцінюється кількістю від 2 до 3 мільярдів людей: більшість прихильників знаходиться в США, Канаді, Китаї, Японії та на Філіпінах [1]. Ця величезна база прихильників звичайно була здобута багато в чому завдяки популярності National Basketball Association (NBA).

У баскетболі, як і в будь-якому іншому сучасному виді спорту, кількість статистичних показників активно збільшується. Це є одним з інструментів привертання уваги до змагань, залучення більшої аудиторії до перегляду матчів. Сучасним поколінням недостатньо просто дивитися матч, вони вимагають більшого занурення в процес, подібності до комп'ютерних ігор. Статистичні показники дозволяють створювати такі подібності. Звичайно, що статистика в сучасному світі може надаватися користувачам тільки через інформаційні технології. У даній роботі потрібно виділити особливості такого програмного забезпечення, що відсутні в існуючих рішеннях і будуть сприяти створенню індивідуальної траєкторії роботи зі статистикою.

Метою робота було визначено розроблення програмного забезпечення слідкування за баскетбольними змаганнями для забезпечення індивідуальної траєкторії такого слідкування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Статистичні показники баскетбольних змагань

Оскільки будь-який турнір визначається кількістю команд, які беруть в ньому участь, а також матчами між ними, то основним питанням, яке має бути чітко визначено для вивчення предметної області, є індивідуальна статистика виступу гравців.

Проаналізуємо поняття, якими визначено показники індивідуальних виступів гравців:

– повернення: мета в баскетболі – успішно виконувати удари, але без м’яча команда не може це робити, ось чому статистика повернення така ж важлива, як і статистика очок за гру, окрім того існує така статистика як відсоток захищених повернень, щоб з’ясувати, наскільки кваліфікованим є гравець у відборах, потрібно підсумувати кількість пропущених ударів, які не вийшли за рамки та не призвели до фолу, потім обчислити відсоток випадків, коли окремий гравець заволодів м'ячем в результаті цих пропущених ударів, не враховуючи порушення правил [2];

– асистування: гравець допомагає іншому гравцеві в спробі закинути м’яч у кошик, різні баскетбольні ліги по-різному визначають і обчислюють передачі, наприклад, у NBA передача відбувається, коли гравець передає м'яч іншому гравцеві, який безпосередньо закидає в кошик його, на рівні коледжів NCAA визначає це як будь-яку передачу, яка сприяє голам, незалежно від того, скільки гравців торкається м'яча, перш ніж той закидає його в сітку, при цьому відсоток передач – це оцінка відсотка польових голів товариша по команді, яким гравець допомагав, перебуваючи на паркеті [2];

– рейтинг ефективності гравця або Player Efficiency Rating (PER) – це популяризований колишнім гуру аналітики телеканалу ESPN Джоном Холлінгером показник, який став найбільш часто використовуваною вдосконаленою метрикою, показник направлений на те, щоб виміряти продуктивність гравця за хвилину, він підсумовує всі позитивні внески

гравця до своєї команди, віднімаючи негативні статистичні бали, він також враховує як темп, так і час гри, щоб полегшити порівняння гравців між собою, при цьому в формулу не входить багато захисних статистичних даних, наприклад, відбори та блокування, хоча вони і не обов'язково співвідносяться з хорошим захистом [3];

– частина виграшів оцінює внесок окремого гравця в загальний виграш його команди: розраховує наступальні виграшні частки, обчислюючи граничні очки гравця з його отриманих очків та наступальних володінь м’ячем та розділивши їх на граничні очки за виграш, захисні виграші розраховуються шляхом обчислення граничного захисту гравця від його оборонного рейтингу та ділення його на граничні очки за виграш, далі ці показники складаються, щоб отримати загальну частку виграшу, при цьому гравці, які втрачають час через травму, отримують нижчий рейтинг, це потужний інструмент для оцінки впливу гравців на напад, захист та загальний успіх своєї команди [3];

– ефективність наступу та оборони: виступ у баскетболі багато в чому залежить від ефективності, максимізація набраних балів та мінімізація очок, дозволених за кожне володіння, важливіша за загальні підсумки, на підсумки впливають такі змінні, як темп або кількість володінь, які команда отримує в грі, ефективність наступу та оборони регулюється відповідно до темпу, обчислюється кількість набраних очок на основі володіння, щоб цифри були легкими для сприйняття, їх повідомляють на 100 володінь, тому вони схожі на показники кількості очок за гру [2]-[3];

– бали за володіння або Points Per Possession (PPP): оцінюючи ефективність кидків, недостатньо просто поглянути на традиційний відсоток голів, адже цей показник однаково враховує двоочкові і триочкові кидки, хоча триочкові приносять вирішальний додатковий бал, також не враховуються штрафні кидки, при цьому два найефективніших способи забити гол у баскетболі – це триочкові та штрафні кидки, справжній відсоток кидків враховує їх обох, враховуючи спроби штрафних кидків та

використовуючи загальну кількість набраних очок замість загальної кількості голів, справжній відсоток може застосовуватися і до команд, але, дивлячись на команди, прийнято відокремлювати штрафні кидки, натомість слід дивитись на ефективний відсоток голів, який вимірює ефективність кидків команди тільки з гри [3]-[4];

– частка штрафних кидків – це спроби штрафних кидків, розділені на спроби кидків з поля, це показник ефективності наступу, адже штрафні кидки – один із найефективніших способів забити гол, поряд із триочковими, чим більше штрафних кидків буде виконуватися за кожну спробу кидка, тим ефективнішим буде наступ, цей показник може бути розрахований як для гравців, так і для команд (де він може бути розрахований як для нападу, так і для захисту) і є корисним показником для аналізу в будь-якому випадку [2];

– відсоток використання оцінює відсоток володіння команди, що використовується гравцем, коли він знаходився на майданчику, при цьому володіння зазвичай визначається як спроба атаки, фолу або повернення, загалом, ефективність знижується із збільшенням використання, однією з визначальних характеристик суперзірки є те, що такий гравець може використовувати велику кількість володінь з мінімальним зниженням ефективності [3]-[5];

– частка відборів: замість загального числа відборів, частка відборів обчислює відсоток доступних відборів, які команда або гравець захоплює, це пристосовується до таких змінних, як темп (команди, які мають більше володінь, матимуть більше пострілів), окрім того даний показник можна додатково розділити на частку відборів у нападі та в захисті, для команди частка відборів показує процентну різницю між відборами, які отримує одна команда, порівняно з опонентами [3];

– частка асистувань, повернень, блоків: усі ці показники передають інформацію подібним чином, тому їх можна об’єднати, знову ж таки, це статистика часток, яка регулює темп і час гри, і їх можна обчислити як для окремих людей, так і для команд, частка асистувань оцінює відсоток

польових голів товариша по команді, яким гравець допоміг, перебуваючи на паркеті, частка повернень оцінює відсоток володінь суперника, які закінчувались поверненням гравця, частка блоків оцінює відсоток спроб кидків суперника, які гравець блокує [3], [5];

– розділення за присутністю: за обчислення цього показника фіксується результативність команди (наступальна та захисна ефективність, частка повернень тощо), коли певний гравець знаходиться на майданчику або поза ним, виступ одного гравця стосовно іншого гравця, який знаходиться на майданчику або поза ним, також можна відстежувати, хоча ці дані знайти набагато складніше, ця статистика допомагає продемонструвати вплив гравця на різні аспекти команди [3]-[4];

– склади команди: подібно до попереднього розділення, продуктивність певних складів команди також можна відстежувати, дані складу можна відсортувати за групами від п’яти, чотирьох, трьох та двох гравців, визначення того, як використовуються різні склади, ілюструє, які групи найкраще функціонують у команді [2]-[5].

1.2 Програмні засоби слідкування за баскетбольними змаганнями

Одним з рушіїв зокрема і тих показників баскетбольної статистики, які були розглянуті в попередньому підрозділі, є телеканал ESPN [6], тому саме з його вебсайту (рис. 1.1) в частині, присвяченій баскетболу і потрібно розпочати аналіз існуючих програмних засобів для даної сфери.

Баскетбольний розділ вебсайту присвячений NBA і дозволяє переглянути стосовно ліги розклад ігор, результати за календарем, турнірну таблицю, лідерів за статистичними показниками серед всіх гравців ліги або заданої команди, відобразити дані за командою, включаючи статистику за її гравцями, розклад ігор, склад, травмованих гравців, окрім того за кожним гравцем можна відкрити його індивідуальну сторінку, що включає загальні статистичні показники за всіма іграми, за окремими останніми іграми.

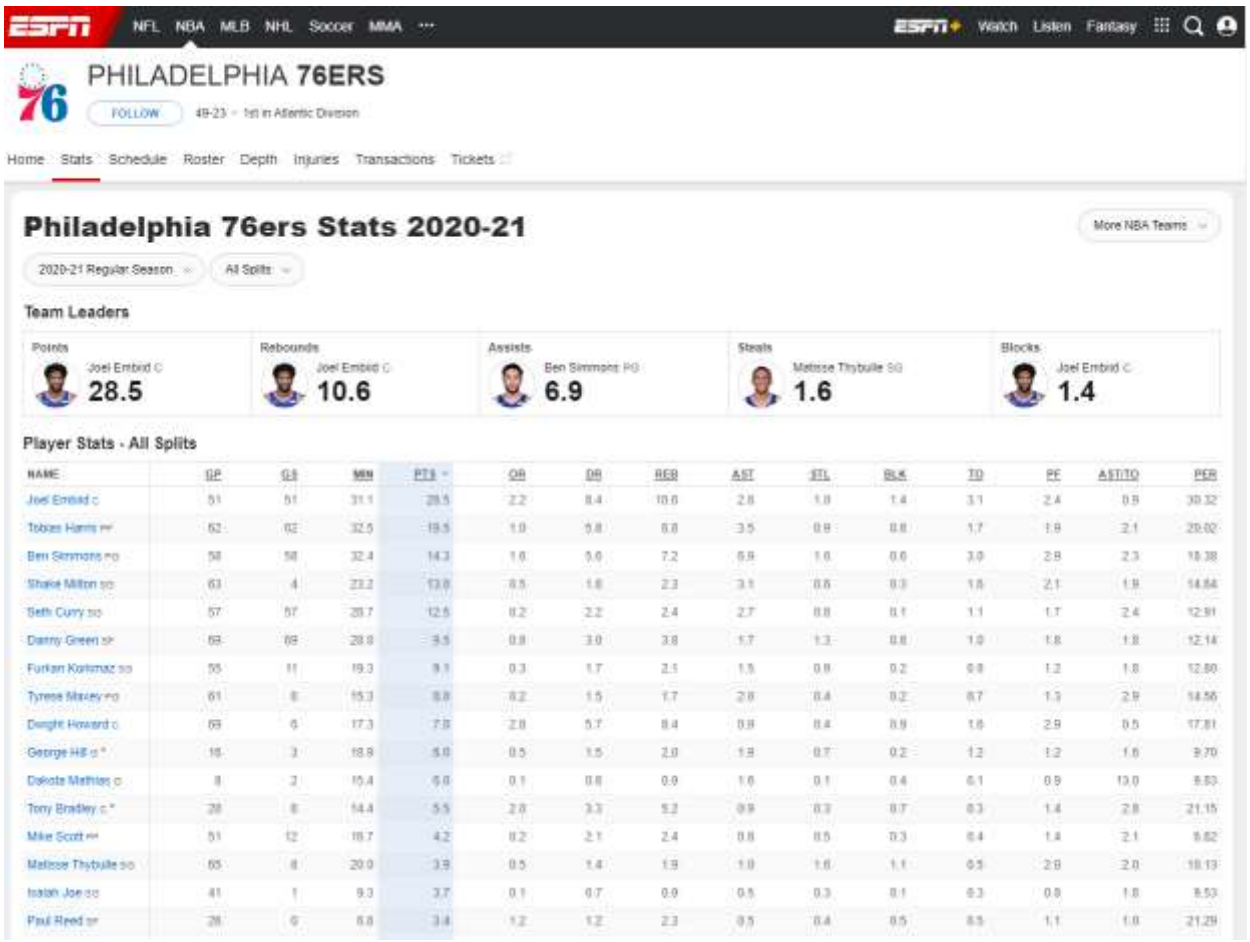


Рисунок 1.1 – Вебсайт ESPN

Для авторизованих користувачів функціональність вебсайту стосовно результатів майже ніяк не змінюється, адже з реєстрацією користувач отримує можливості оформлення підписок на засоби телеканалу і подібні засоби.

Basketball Reference [7] (рис. 1.2) є вебсайтом, який спрямований саме на баскетбольну статистику та відслідковування результатів баскетбольних змагань.

Вебсайт дозволяє переглянути інформацію про гравців, команди, статистику попередніх сезонів, лідерів за різними статистичними показниками за різні сезони, при чому за кожним показником відображається тільки один гравець, результати виступів команд. Варто відзначити, що будь-які засоби, що стосуються не окремої команди, а усіх команд ліги разом більше направлені на підсумовування даних за всіма сезонами.

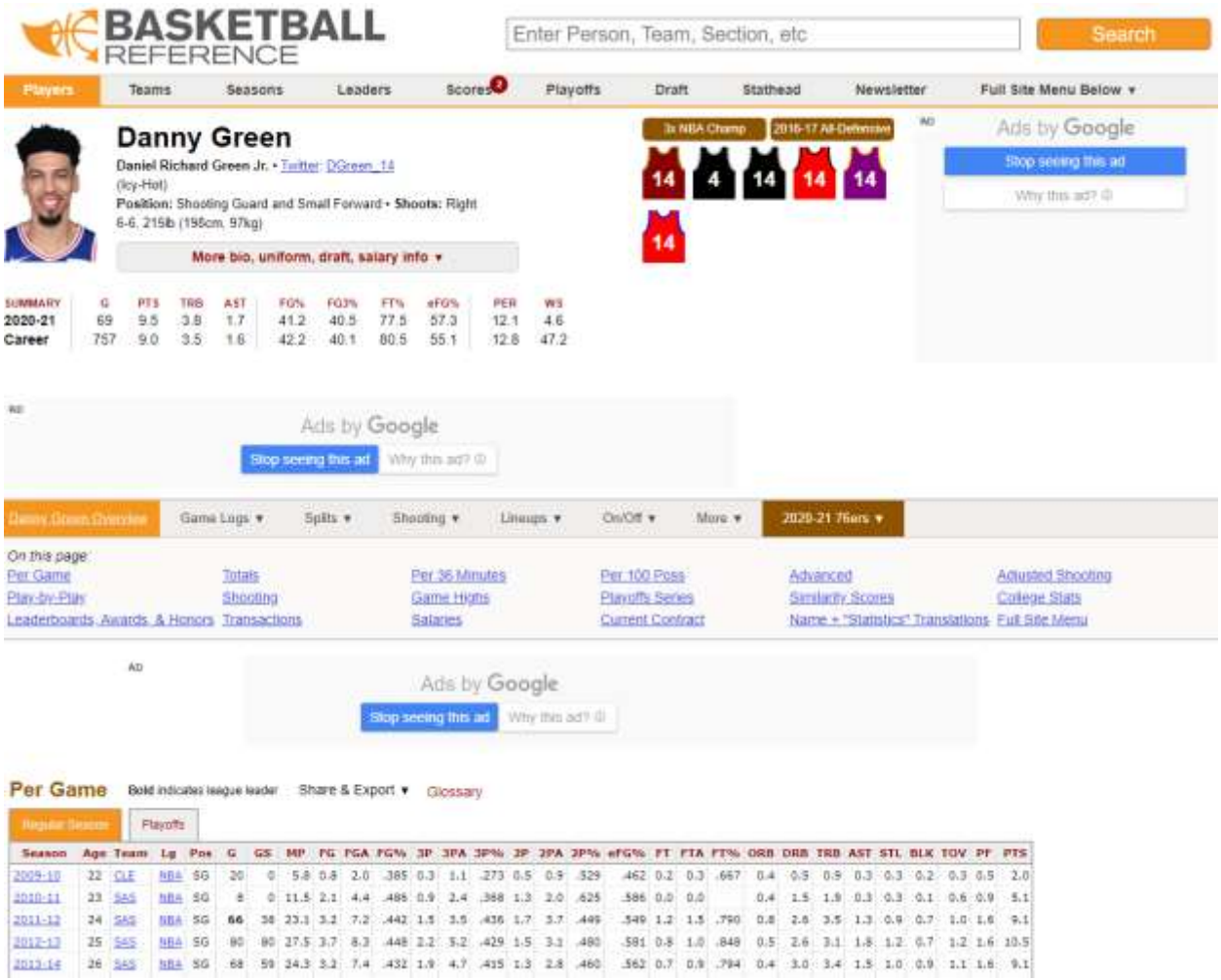


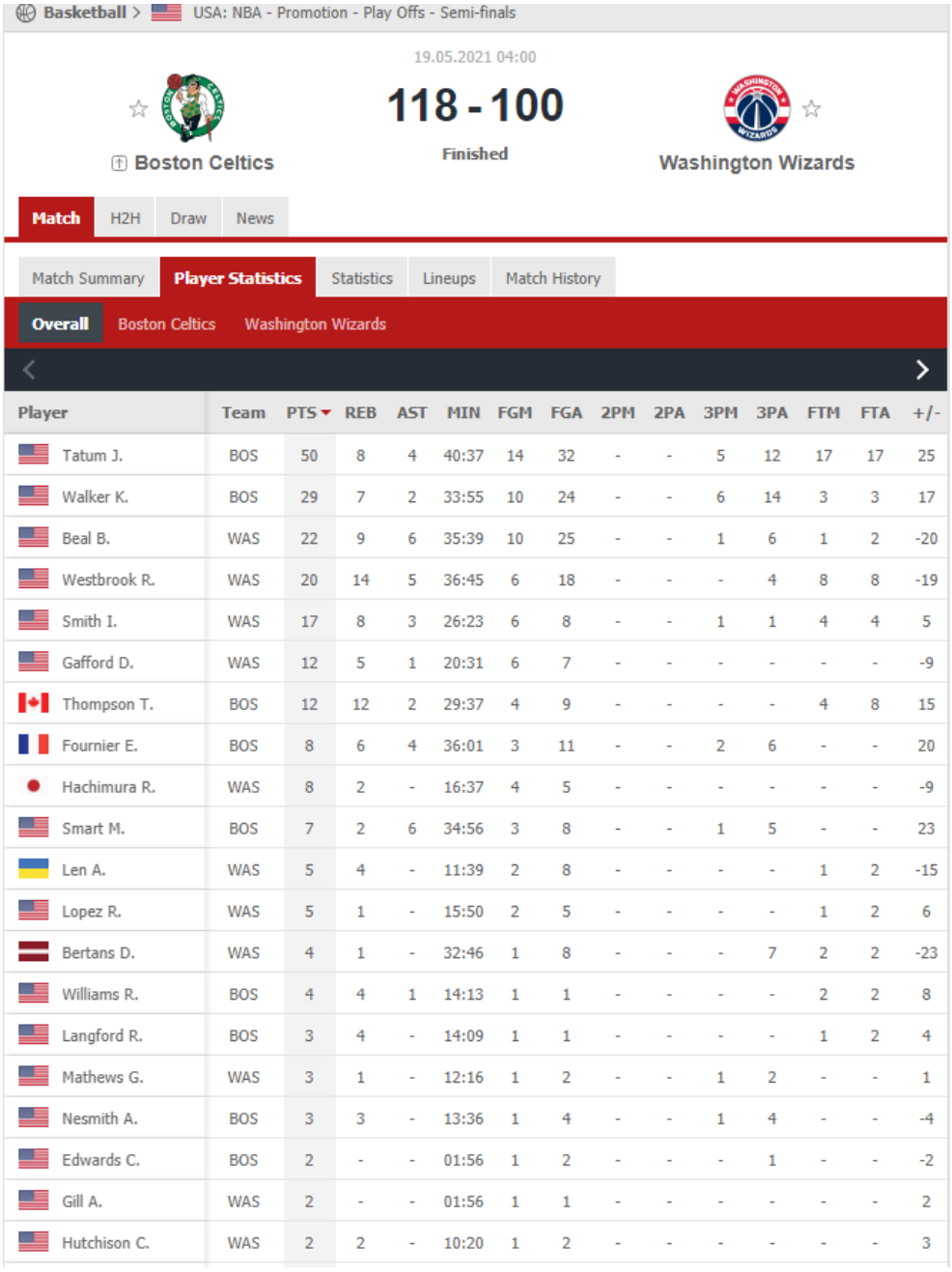
Рисунок 1.2 – Вебсайт Basketball Reference

Негативною особливістю цього рішення є велика кількість реклами, що під час роботи з будь-якою сторінкою займає більше чверті корисного простору.

Вебсайт Scoreboard [8] (рис. 1.3) також надає доступ до баскетбольної статистики, але основним напрямком цієї статистики є як раз результати змагань. Тобто основними сутностями, з якими ведеться робота, є змагання, матч та команда.

Програма дозволяє користувачам зареєструватися і працювати з нею, отримуючи персональні засоби. Ці персональні засоби дещо обмежені. Можна підписатися на змагання, на команду або на окремий матч, додаючи їх до улюблених, це дозволить швидше переходити до інформації про відповідно змагання, команду або матч. Тобто на сторінці користувача буде

відображатися перелік таких команд і це дозволить вирішити проблему навігації по вебсайту. При цьому не буде надано одразу статистичні показники. Тобто така можливість вирішує головним чином проблему навігації по вебсайту.



Basketball > USA: NBA - Promotion - Play Offs - Semi-finals

19.05.2021 04:00

Boston Celtics **118 - 100** Washington Wizards

Finished

Match | H2H | Draw | News

Match Summary | Player Statistics | Statistics | Lineups | Match History

Overall | Boston Celtics | Washington Wizards

Player	Team	PTS	REB	AST	MIN	FGM	FGA	2PM	2PA	3PM	3PA	FTM	FTA	+/-
Tatum J.	BOS	50	8	4	40:37	14	32	-	-	5	12	17	17	25
Walker K.	BOS	29	7	2	33:55	10	24	-	-	6	14	3	3	17
Beal B.	WAS	22	9	6	35:39	10	25	-	-	1	6	1	2	-20
Westbrook R.	WAS	20	14	5	36:45	6	18	-	-	-	4	8	8	-19
Smith I.	WAS	17	8	3	26:23	6	8	-	-	1	1	4	4	5
Gafford D.	WAS	12	5	1	20:31	6	7	-	-	-	-	-	-	-9
Thompson T.	BOS	12	12	2	29:37	4	9	-	-	-	-	4	8	15
Fournier E.	BOS	8	6	4	36:01	3	11	-	-	2	6	-	-	20
Hachimura R.	WAS	8	2	-	16:37	4	5	-	-	-	-	-	-	-9
Smart M.	BOS	7	2	6	34:56	3	8	-	-	1	5	-	-	23
Len A.	WAS	5	4	-	11:39	2	8	-	-	-	-	1	2	-15
Lopez R.	WAS	5	1	-	15:50	2	5	-	-	-	-	1	2	6
Bertans D.	WAS	4	1	-	32:46	1	8	-	-	-	7	2	2	-23
Williams R.	BOS	4	4	1	14:13	1	1	-	-	-	-	2	2	8
Langford R.	BOS	3	4	-	14:09	1	1	-	-	-	-	1	2	4
Mathews G.	WAS	3	1	-	12:16	1	2	-	-	1	2	-	-	1
Nesmith A.	BOS	3	3	-	13:36	1	4	-	-	1	4	-	-	-4
Edwards C.	BOS	2	-	-	01:56	1	2	-	-	-	1	-	-	-2
Gill A.	WAS	2	-	-	01:56	1	1	-	-	-	-	-	-	2
Hutchison C.	WAS	2	2	-	10:20	1	2	-	-	-	-	-	-	3

Рисунок 1.3 – Вебсайт Scoreboard

Основним недоліком даного рішення є відсутність можливості працювати з окремими баскетболістами. Тобто за ними можна достатньо

інформативно отримувати дані про їх виступ в конкретному матчі, але для цього потрібно обрати змагання, матч, а тоді всередині відповідної сторінки буде відображено необхідні дані. У той же час окремої сторінки гравця не існує, відповідно побачити загальну статистику баскетболіста за всі матчі сезону, за роки неможливо.

Proballers [9] (рис. 1.4) є ще одним програмним аналогом.

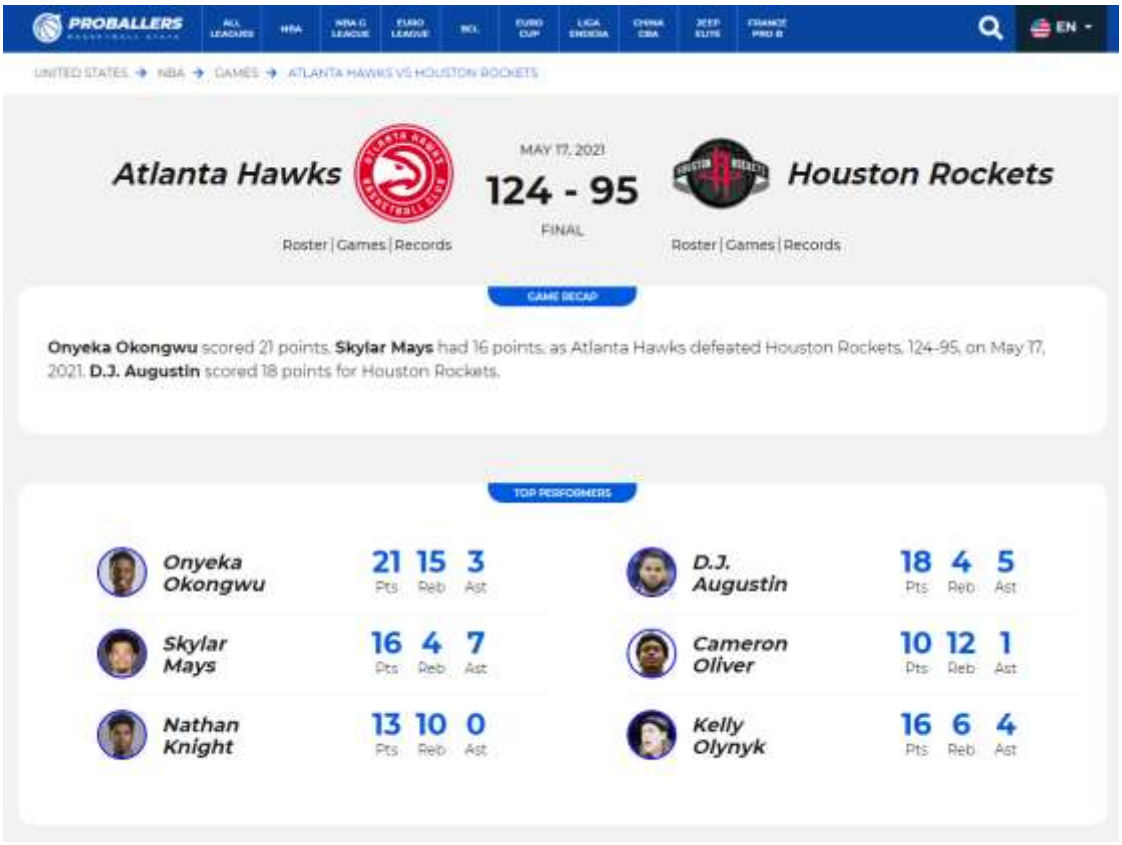


Рисунок 1.4 – Вебсайт Proballers

Proballers має подібну функціональність, але розширені засоби включають особливе представлення інформації. Наприклад, за кожним гравцем доступний ще й текстовий опис його виступу в грі.

Спільною рисою для всіх розглянутих засобів є схильність до табличного представлення даних.

1.3 Порівняння аналогів

Проведений аналіз існуючих програмних засобів продемонстрував, що всі вони надають засоби статистики змагань, деякі з них не мають засобів відслідковування загальної статистики за гравцем, але при цьому всі вони подібні в тому, що не надають можливість зручного відслідковування за потрібною статистикою, створюючи індивідуальну траєкторію такого відслідковування. Тобто користувачі зазвичай слідкують за окремими виконавцями, командами, а лише Basketball Reference надає можливість сформувати перелік команд, але цей перелік тільки пришвидшує подальший перехід до потрібної сторінки, тобто за результатами не формується сторінка під користувача зі статистикою одразу.

Порівняння аналогів виконано в табл. 1.1.

Таблиця 1.1 – Порівняння аналогів

	Basketball Reference	Scoreboard	Розробка
Відслідковування результатів змагань	Так	Так	Так
Відслідковування статистики команд	Так	Так	Так
Відслідковування статистики гравців	Так	Ні	Так
Статистика вибраних гравців та команд	Ні	Ні	Так

1.4 Висновки за розділом 1

Завдання дипломної кваліфікаційної роботи:

– аналіз предметної області;

– проєктування програмного забезпечення;

– реалізація і тестування програмного забезпечення.

Проведений аналіз предметної області дозволив визначити відомі статистичні показники виступу гравців у баскетбольних змаганнях, проаналізувати існуючі вебсайти баскетбольної статистики. Проведений аналіз аналогів показав, що існуючі рішення не мають індивідуальної траєкторії відслідковування статистики. На створення програмної реалізації, що підтримує таку функціональну особливості, направлена дана робота.

2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

2.1 Функціональна вимоги до програмного забезпечення

Підтримка роботи програми має забезпечуватися двома сторонами: адміністратором і звичайними користувачами. Звичайні користувачі можуть бути неавторизованими і авторизованими.

Адміністратор може виконувати внесення даних щодо баскетбольних змагань в систему і керувати ними за необхідності. Тобто його відповідальність – наповнення системи даними.

У результаті проведеного аналізу предметної області функціональними вимогами до програми стосовно адміністратора були визначені наступні:

– додавання баскетбольних команд;

– редагування даних баскетбольних команд;

– внесення складу баскетбольної команди;

– керування складом баскетбольної команди;

– редагування даних гравців;

– додавання баскетбольного змагання;

– внесення ігор баскетбольного змагання;

– редагування внесених ігор;

– внесення результатів баскетбольних матчів;

– внесення статистики баскетбольного матчу;

– редагування внесених даних матчів.

За даними функціональними вимогами було виконано моделювання роботи адміністратора з програмою (рис. 2.1).

Неавторизований користувач може тільки переглядати інформацію про баскетбольні змагання, що подається в різній формі, використовуючи її для аналізу. Ніяких засобів індивідуалізації неавторизованому користувачу не надається.

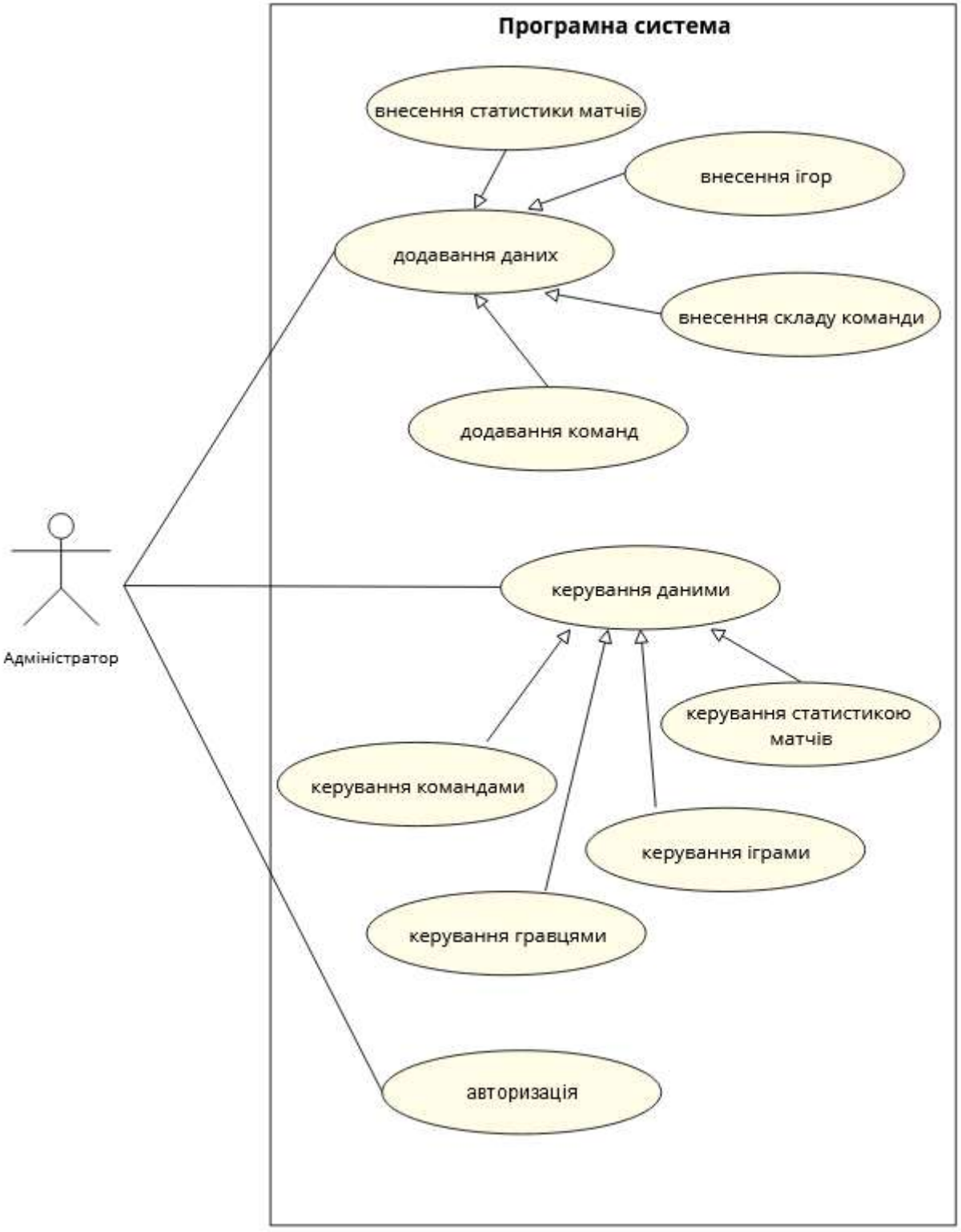


Рисунок 2.1 – Діаграма прецедентів адміністратора

Для неавторизованого користувача було виконано моделювання роботи з програмною системою (рис. 2.2).

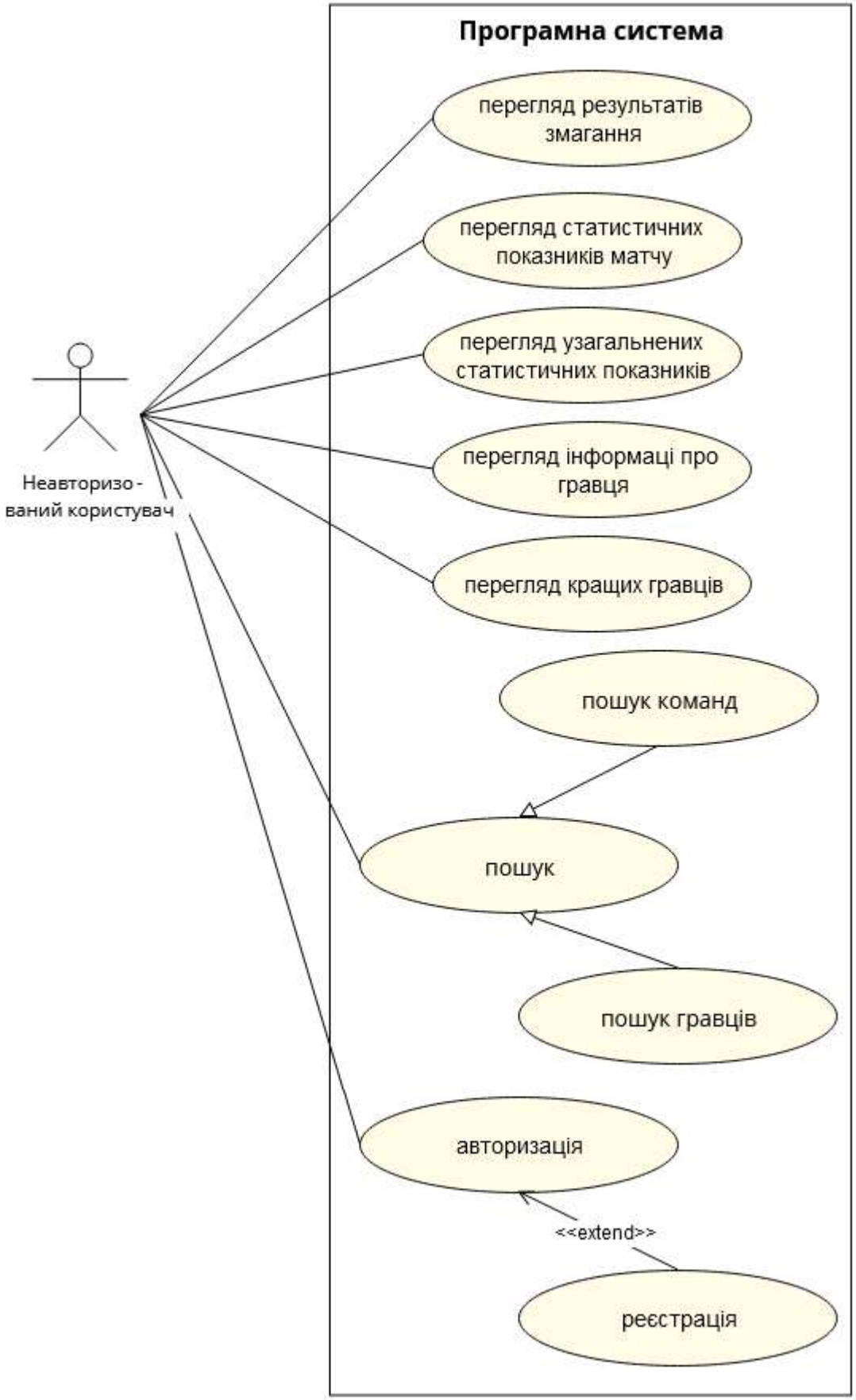


Рисунок 2.2 – Діаграма прецедентів неавторизованого користувача

У результаті проведеного аналізу предметної області функціональними вимогами до програми стосовно звичайних користувачів були визначені наступні:

– перегляд таблиці змагання;

– перегляд результатів матчів змагання;

– перегляд статистичних показників команд у матчі;

– перегляд статистичних показників гравців у матчі;

– перегляд інформації про гравця;

– перегляд узагальненої статистики гравця за змаганнями;

– пошук гравців;

– пошук команд;

– перегляд узагальненої статистики команди за змаганнями;

– перегляд кращих гравців за статистичними показниками за турніром;

– перегляд кращих гравців за статистичними показниками у команді;

– авторизація;

– реєстрація користувачів;

– визначення улюблених команд:

– визначення улюблених гравців;

– визначення улюблених змагань;

– керування переліком улюблених команд:

– керування переліком улюблених гравців;

– керування переліком улюблених турнірів;

– відображення статистики улюблених гравців;

– відображення статистики улюблених команд;

– керування профілем користувача.

Для авторизованого користувача було виконано моделювання роботи з програмною системою (рис. 2.3). Авторизований користувач може реалізовувати індивідуальну траєкторію роботи над статистикою шляхом уподобання команд, гравців та змагань і перегляду їх статистичних даних.

Рисунок 2.3 – Діаграма прецедентів авторизованого користувача

2.2 Вибір технології розробки

Як було проаналізовано в першому розділі, усі існуючі рішення мають щонайменше вебзастосунок для підтримки своєї роботи.

У роботі було проаналізовано технології розробки вебзастосунків, що включають вебфреймворки ASP.NET [10] та Django. Результати вибору технології веброзробки зведено до табл. 2.1.

Таблиця 2.1 – Вибір технології веброзробки

	ASP.NET	Django
Розповсюдженість	Перевага на ринку в усіх категоріях	Програє в усіх категоріях
Об’єктно-реляційне відображення	Так	Так
Обробка багатопотокових запитів	Так	Ні
Мова програмування	C#, Visual Basic.NET, J#	Python

У результаті було обрано для використання технологію ASP.NET, оскільки вона використовується більшою кількістю проєктів, використовується великими проєктами, тобто загалом можна говорити про те, що це рішення надійне і на практиці при роботі з великою кількістю користувачів не має викликати проблем. Окрім того ASP.NET дозволяє обробляти багатопотокові запити користувачів, тобто працювати з великою кількістю запитів у різних процесах, що в порівнянні з Django дозволяє пришвидшити роботу вебзастосунку.

Перевагами технології ASP.NET є:

– відокремлення проблеми: ASP.NET слідує архітектурі Model-View-Controller, яка дозволяє відокремити введення, оброблення даних та виведення в програмі, ця трирівнева архітектура має взаємопов’язані деталі та може обробляти конкретні аспекти розробки програмних застосунків;

– скорочує час кодування: фреймворкова технологія дуже допомагає зменшити час кодування, особливо коли розробляються великі програми, існують різні типи оглядів коду, тому у розробника немає шансів написати поганий код, огляди коду допомагають поліпшити якість коду;

– складається з деяких нестандартних особливостей: ASP.NET забезпечує підвищену продуктивність та масштабованість, також має такі функції, як компіляція в процесі, раннє прив'язування, нативна оптимізація та послуги кешування, і вони також служать для підвищення продуктивності на декілька рівнів вище;

– великий набір інструментів: фреймворк має багатий набір інструментів завдяки вбудованому середовищу розробки Visual Studio, цей набір інструментів виконує роль дуже важливої основи побудови та допомагає розробнику створювати програми дуже швидко;

– забезпечення потужності та гнучкості: мова фреймворку базується на загальномовному середовищі виконання, тому всі розробники вебзастосунків можуть користатися гнучкістю та потужністю всієї платформи, фреймворк не залежить від мови, тому можна вибрати мову для своєї програми або розділити її на декілька мов;

– простота: кожне завдання можна виконати легко, навіть складне, загальномовне середовище виконання робить процес розробки простим, з такими послугами, як збір сміття та автоматичний підрахунок посилань, можна створювати користувацькі інтерфейси, які можуть розділяти логіку програми та код презентації;

– настроюваність та розширюваність: добре продумана архітектура фреймворку є головною допомогою для розробників, можна легко розширити або замінити підкомпонент середовища виконання ASP.NET за допомогою власних підготовлених на замовлених компонентів, впровадити їх стало достатньо просто;

– безпека: можна розробляти захищені програми за допомогою вбудованих функцій автентифікації Windows та конфігурації для кожного застосунка;

– керованість: відмінна керованість функції фреймворку забезпечується завдяки його ієрархічній системі конфігурації на основі тексту, оскільки ці конфігурації включені як звичайні текстові дані, можна просто скористатися інструментами місцевого адміністрування, щоб застосувати нові налаштування, це значно полегшує завдання без перезапуску сервера або необхідності розгортання окремо або заміни запущеного скомпільованого коду;

– перевага постійного моніторингу: постійний моніторинг – особливість ASP.NET, не потрібно турбуватися про стан програм, компонентів та самих сторінок, програма стежить за будь-якими незаконними подіями, і якщо щось трапиться, вона негайно вступає в дію, перезапускаючи себе;

– міжплатформна міграція: мова фреймворку забезпечує легку міжплатформенну міграцію, конфігурацію та послуги розгортання [11].

В якості мови програмування було використано мову C#, оскільки це сучасна об’єктно-орієнтовна мова програмування, яка дозволяє зокрема створювати і варіанти рішень для мобільних застосунків, що є позитивною характеристикою.

2.3 Проєктування бази даних

У ролі системи керування базами даних було використано Microsoft SQL Server, що дозволило забезпечити найвищий можливий рівень для вбудованого використання таких рішень, адже C#, ASP.NET, Microsoft SQL Server є добре відтестованою зв’язкою на масштабних програмних застосунках.

За результатами проєктування бази даних було виділено наступні основні сутності.

Сутність User визначає користувача, який працює з програмою.

Сутність Competition визначає змагання, в якому беруть участь баскетбольні команди. У виконаному огляді деякі застосунки є направленими на одне конкретне змагання, найчастіше NBA, а є такі, які дозволяють збирати статистику з різних змагань. Рішення, що розробляється, підтримує ідею збору статистики з різних змагань одночасно. Тому було введено дану сутність.

Сутність CompetitionSeason є конкретним сезоном змагання, в якому беруть участь конкретні команди.

Відповідні таблиці Competition і CompetitionSeason зв’язані зв’язком один-до-багатьох, оскільки кожне змагання відбувається кожен рік, відповідно сезонів може бути багато, але кожен сезон відповідає одному конкретному змаганню.

Сутність Team визначає безпосередньо команду, що бере участь в турнірах.

Сутність Player визначає гравця, який є учасником команди, бере участь в іграх змагань.

Сутність Game визначає конкретну гру, що відбувається в межах відповідного змагання.

Сутність PlayerGame характеризує участь конкретного гравця в конкретній грі. Саме тут і мають зберігатися статистичні дані. Це статистичні дані одного гравця в одній грі.

Оскільки кожен гравець приймає участь в багатьох іграх, то зв’язок між таблицями Player і PlayerGame один-до-багатьох (таблиця PlayerGame визначає участь конкретного гравця).

Оскільки у кожній грі приймає участь багато гравців, то зв’язок між таблицями Player і PlayerGame один-до-багатьох.

Сутність Trainer визначає тренера команди.

Таблиці Game і Trainer пов’язані зв’язком один-до-одного двічі, оскільки в кожній конкретній грі командою господарів керує один тренер і аналогічно командою гостей керує також один тренер. Такий спосіб визначено, оскільки протягом сезону команда може мати декілька тренерів.

Сутність PlayerCompetitionSeason визначає узагальнену статистику про участь гравця в конкретному сезоні змагання.

Таблиці Player і PlayerCompetitionSeason зв’язані зв’язком один-до-багатьох, оскільки кожен гравець протягом кар’єри може брати участь в декількох сезонах.

Таблиці Team і PlayerCompetitionSeason зв’язані зв’язком один-до-багатьох, оскільки кожна команда протягом одного сезону має багато гравців, при цьому кожен гравець бере участь протягом сезону в одній команді тільки. Але в наступних сезонах гравець може змінити команду, тому в таблиці PlayerCompetitionSeason зберігається зовнішнім ключем саме команда.

Сутність TeamCompetitionSeason визначає узагальнену статистику про участь цілої команди в конкретному сезоні змагання.

Таблиця UserTeam визначає команди, уподобані користувачем, і введена для реалізації зв’язку багато-до-багатьох між таблицями User і Team, адже кожен користувач може уподобати багато команд, а кожна команда може бути уподобана одночасно багатьма користувачами.

Таблиця UserPlayer визначає ігри, уподобані користувачем, і введена для реалізації зв’язку багато-до-багатьох між таблицями User і Player, адже кожен користувач може уподобати багато гравців, а кожен гравець може бути уподобаний одночасно багатьма користувачами.

Таблиця UserCompetition визначає змагання, уподобані користувачем, і введена для реалізації зв’язку багато-до-багатьох між таблицями User і Competition, адже кожен користувач програми може уподобати багато змагань, а кожне змагання може бути уподобано одночасно багатьма користувачами.

У результаті була побудована схема бази даних (рис. 2.4).

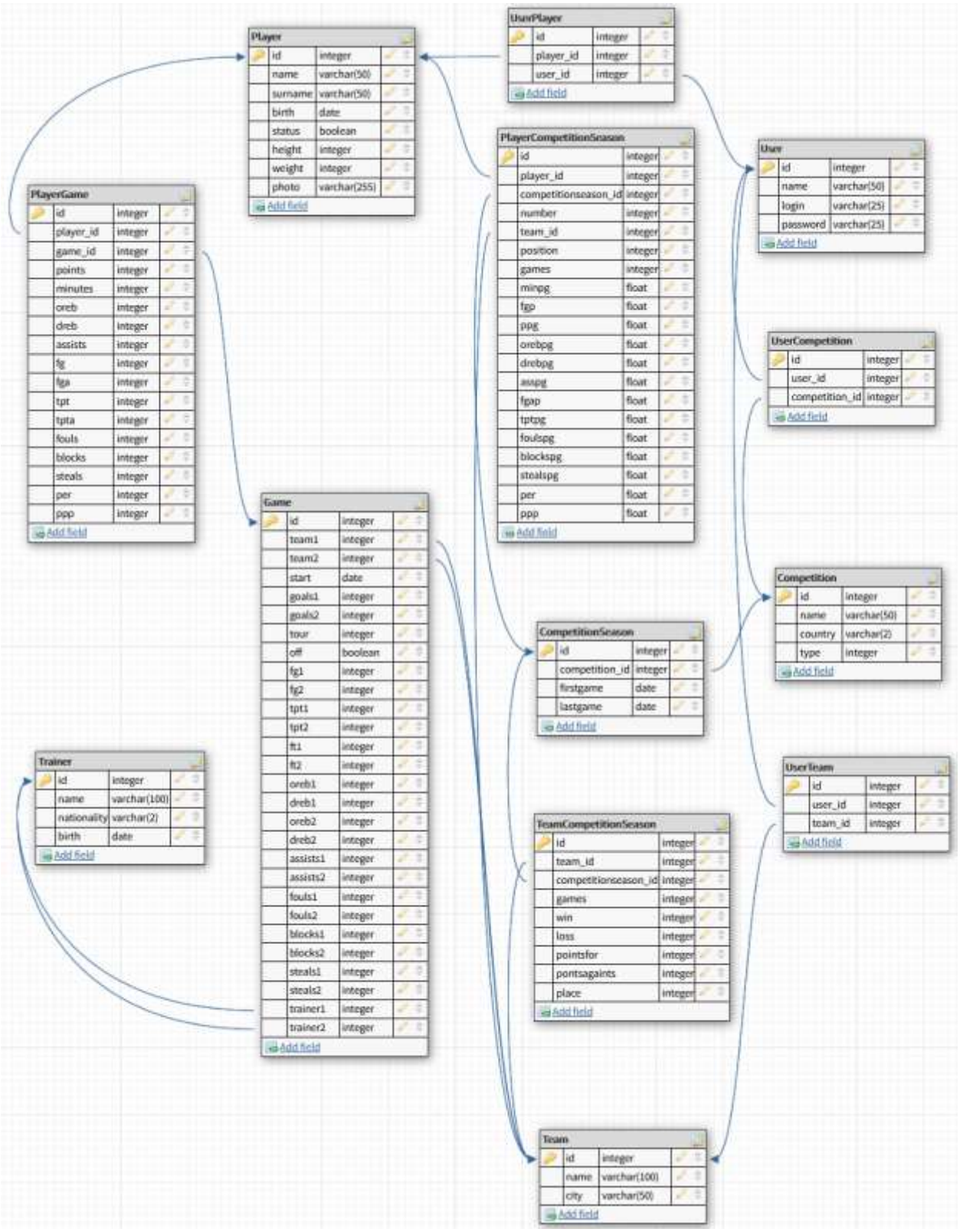


Рисунок 2.4 – Схема бази даних

2.4 Архітектура програмного забезпечення

У відповідності з описаною в підрозділі 2.2 логікою використання технології ASP.NET і її архітектурою було визначено архітектуру застосунку (рис. 2.5).

Рисунок 2.5 – Архітектура програми

Архітектура програми побудована на тришаровому принципі. Презентаційний шар розроблено окремо для звичайного користувача і окремо для адміністратора у вигляді адміністративної панелі.

Презентаційний шар взаємодіє з шаром бізнес-логіки, який реалізує основні принципи роботи в предметній області, в той час як шар доступу до даних, до якого звертається шар бізнес-логіки, визначає принципи роботи з даними в предметній області. Шар доступу до даних за допомогою об’єктно-реляційного відображення звертається до бази даних.

2.5 Висновки за розділом 2

У даному розділі визначено функціональні вимоги для роботи неавторизованого, авторизованого користувачів та адміністратора з програмою, проведено моделювання взаємодії цих користувачів з програмою у вигляді діаграм прецедентів, обрано мову програмування C# та технологію веброзробки ASP.NET для розроблення програмного забезпечення, виконано проєктування бази даних, в результатів чого створено схему бази даних, визначено архітектуру вебзастосунку на основі трьох основних шарів, що відділяють роботу з даними від представлення цих даних користувачу.

3 РОЗРОБКА ПРОГРАМИ

3.1 Алгоритм функціонування програмного забезпечення

Логіка роботи програми побудована на 2 окремих режимах. Перший режим потрібний адміністратору, другий – користувачу. Оскільки в першому режимі роботи відбувається над групами даних, тобто над їх редагування, додаванням, то логіку буде описано детальніше на прикладі роботи звичайного користувача з програмою.

Після початку роботи програми користувачу відображаються результати змагання в останньому сезоні NBA. Ці дані відображаються за замовчуванням.

Коли ці дані відображені, користувач може обрати наступний варіант режиму роботи. Якщо він виконує вибір змагання, то йому на цій же сторінці відображаються результати змагання в останньому сезоні змагання. Якщо користувач обере один з сезонів змагання, то йому буде відображено результати змагання в заданому сезоні.

Користувач може з переліків, які йому відображаються, обрати команду або гравця і відобразити по ним повні дані.

За необхідності користувач може обрати можливість пошуку гравців і перейти на відповідну окрему сторінку. Така ж окрема сторінка доступна користувачу з меню для пошуку команд. Пошук відбувається між всіма командами, що беруть участь у всіх змаганнях.

Якщо користувач авторизований, то йому буде доступна можливість переходу до сторінки профілю користувача з можливістю зміни його. Також авторизований користувач може відобразити індивідуальну статистику, що формується на основі його переліків уподобань.

Для виконання описаних дій, якщо користувач неавторизований, він може авторизуватися. Або зайти в інший обліковий запис, якщо авторизований.

Відповідну схему функціонування програми приведено на рис. 3.1.

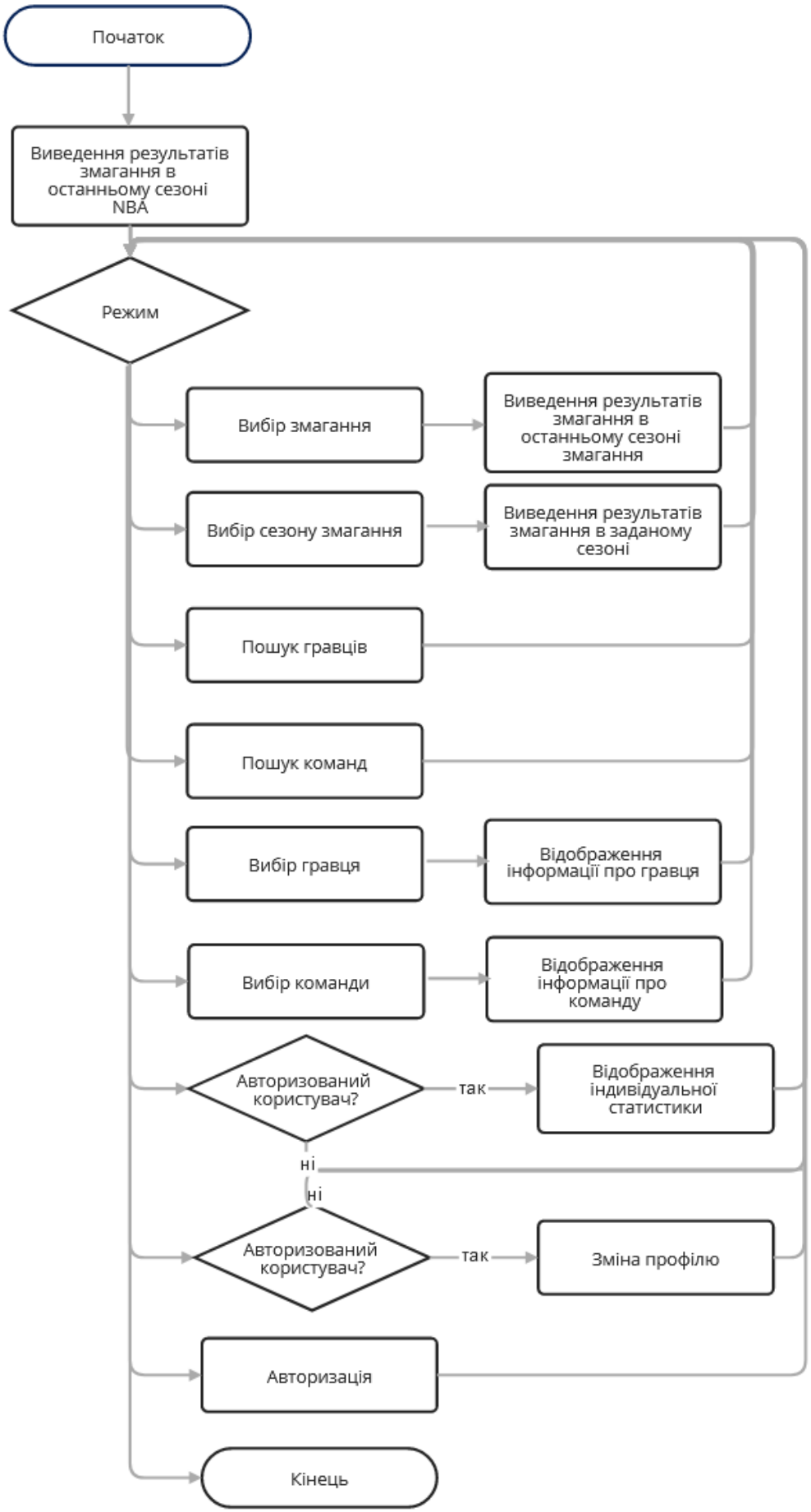


Рисунок 3.1 – Схема функціонування програми для звичайного користувача

3.2 Опис прийнятих під час реалізації рішень

Для реалізації шару роботи з даними було реалізовано відповідні класи.

Клас гравця Player включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер гравця;

– public string Name – ім’я гравця;

– public string Surname – прізвище гравця;

– public DateTime Birth – дата народження гравця;

– public bool Status – стан гравця, який використовується для позначення гравців, які вже завершили кар’єру;

– public int Height – зріст гравця;

– public int Weight – вага гравця;

– public string Photo – фотографія гравця;

– public string StatusStr – доступний тільки на отримання значення, визначає текстове представлення стану гравця;

– public string BirthStr –доступна тільки на отримання значення дата народження гравця в текстовому форматі;

– public virtual ICollection<PlayerGame> PlayerGames – колекція ігор гравця;

– public virtual ICollection<PlayerCompetitionSeason> PlayerCompetitionSeasons – колекція узагальнених статистик гравця за виступами в різних сезонах змагань.

Клас змагання Competition включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер змагання;

– public string Name – назва змагання;

– public string Country – країна, де відбувається змагання;

– public int Type – тип роботи зі статистикою змагання (мінімальний,

скорочений або повний – визначає набір статистичних показників, за допомогою яких фіксується виступ гравця і команди в грі);

– public string TypeStr – текстове представлення типу роботи, доступне тільки для читання;

– public virtual ICollection<CompetitionSeason> CompetitionSeasons – колекція сезонів організації змагання.

Клас результатів участі команди в сезоні змагання TeamCompetitionSeason включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер змагання;

– public int TeamId – номер команди;

– public int CompetitionseasonId – номер сезону змагання;

– public int Games – кількість ігор, проведених командою в сезоні даного змагання;

– public int Win – кількість перемог в цих іграх;

– public int Loss – кількість поразок в цих іграх;

– public int Pointsfor – сумарна кількість набраних очок у всіх цих матчах;

– public int Pontsagaints – сумарна кількість очок, набраних іншими командами, у всіх цих матчах;

– public int Place – поточне або остаточне місце команди в змаганні;

– public virtual Team Team – об’єкт команди, що зберігає повну інформацію про команду, номер якої представлено в атрибуті TeamId;

– public virtual CompetitionSeason CompetitionSeason – об’єкт сезону змагання, що зберігає повну інформацію про сезон змагання, номер якого представлено в атрибуті CompetitionseasonId.

Клас гри Game включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер гри;

– public int Team1Id – номер команди господарів у грі;

– public int Team2Id – номер команди гостей у грі;

– public DateTime Start – дата гри;

– public int Goals1 – кількість очок, здобутих першою командою у грі;

– public int Goals2 – кількість очок, здобутих другою командою у грі;

– public int Tour – номер туру змагання, в межах якого відбувається гра;

– public int Off – показник рівня гри (гра в групі або плейофф);

– public int Fg1 – кількість очок, набраних з гри командою господарів;

– public int Fg2 – кількість очок, набраних з гри командою гостей;

– public int Tpt1 – кількість триочкових кидків команди господарів;

– public int Tpt2 – кількість триочкових кидків команди гостей;

– public int Ft1 – кількість штрафних кидків команди господарів;

– public int Ft2 – кількість штрафних кидків команди гостей;

– public int Oreb1 – кількість відборів у нападі команди господарів;

– public int Dreb1 – кількість відборів у захисті команди господарів;

– public int Oreb2 – кількість відборів у нападі команди гостей;

– public int Dreb2 – кількість відборів у захисті команди гостей;

– public int Assists1 – кількість асистувань команди господарів;

– public int Assists2 – кількість асистувань команди гостей;

– public int Fouls1 – кількість порушень правил команди господарів;

– public int Fouls2 – кількість порушень правил команди гостей;

– public int Blocks1 – кількість блокувань команди господарів;

– public int Blocks2 – кількість блокувань команди гостей;

– public int Steals1 – кількість втрат команди господарів;

– public int Steals2 – кількість втрат команди гостей;

– public int Trainer1Id – номер тренера команди господарів;

– public int Trainer2Id – номер тренера команди гостей;

– public virtual Trainer Trainer1 – об’єкт, який представляє тренера команди господарів;

– public virtual Trainer Trainer2 – об’єкт, який представляє тренера

команди гостей;

– public virtual Team Team1 – об’єкт, який представляє команду господарів;

– public virtual Team Team2 – об’єкт, який представляє команду гостей;

– public string PlayDateStr – рядкове представлення дати гри;

– public string OffStr – рядкове представлення рівня гри;

– public virtual ICollection<PlayerGame> PlayerGames – колекція гравців, що взяли участь в грі.

Клас статистики виступу гравця в грі PlayerGame включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер записи щодо участі гравця в грі;

– public int PlayerId – номер гравця;

– public int GameId – номер гри;

– public int Points – кількість очок, здобута гравцем у грі;

– public int Minutes – кількість хвилин, проведених на паркеті гравцем у грі;

– public int Oreb – кількість відборів у нападі, виконаних гравцем;

– public int Dreb – кількість відборів у захисті, виконаних гравцем;

– public int Assists – кількість асистувань, виконаних гравцем;

– public int Fg – кількість очок, набраних з гри;

– public int Fga – кількість очок, які потенційно міг би набрати гравець за свою гру;

– public int Tpt – кількість триочкових кидків гравця;

– public int Tpta – кількість триочкових, які потенційно міг би набрати гравець за свою гру;

– public int Fouls – кількість порушень правил гравця протягом гри;

– public int Blocks – кількість блокувань, виконаних гравцем протягом гри;

– public int Steals – кількість втрат гравця протягом гри;

– public int Per – показник PER гравця протягом гри;

– public int Ppp – показник PPP гравця протягом гри;

– public virtual Player Player – об’єкт гравця;

– public virtual Game Game – об’єкт гри.

Клас узагальненої статистики гравця за сезон в змаганні PlayerCompetitionSeason включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер запису про статистику виступів гравця в сезоні;

– public int PlayerId – номер гравця;

– public int CompetitionseasonId – номер сезону змагання;

– public int number – номер, під яким грає гравець в даному сезоні;

– public int TeamId – команда, за яку виступає гравець у сезоні;

– public int Position – позиція гравця на паркеті в сезоні;

– public int Games – кількість ігор, проведених гравцем за сезон;

– public float Minpg – кількість хвилин, яку в середньому проводив гравець на полі протягом всіх ігор сезону;

– public float Fgp – кількість очок з гри, яку в середньому заробляв гравець на полі протягом всіх ігор сезону;

– public float Ppg – кількість очок, яку в середньому заробляв гравець на полі протягом всіх ігор сезону;

– public float Orebpg – кількість відборів у нападі, виконаних гравцем в середньому за гру;

– public float Drebpg – кількість відборів у захисті, виконаних гравцем в середньому за гру;

– public float Asspg – кількість асистувань, виконаних гравцем в середньому за гру протягом сезону;

– public float Fgap – кількість потенційних очок в середньому за гру сезону;

– public float Tptpg – кількість триочкових, виконаних в середньому за гру сезону;

– public float Foulspg – кількість порушень правил, виконаних гравцем в середньому за гру протягом сезону;

– public float Blockspg – кількість блокувань, виконаних гравцем в середньому за гру протягом сезону;

– public float Stealspg – кількість втрат гравця за одну гру сезону в середньому;

– public float Per – показник PER гравця за всі ігри сезону;

– public float Ppp – показник PPP гравця за всі ігри сезону;

– public string MinpgStr – текстове представлення округленого значення кількості хвилин, яку в середньому проводив гравець на полі протягом всіх ігор сезону;

– public string FgpStr – текстове представлення округленого значення кількості очок з гри, яку в середньому заробляв гравець на полі протягом всіх ігор сезону;

– public string PpgStr – текстове представлення округленого значення кількості очок, яку в середньому заробляв гравець на полі протягом всіх ігор сезону;

– public string OrebpgStr – текстове представлення округленого значення кількості відборів у нападі, виконаних гравцем в середньому за гру;

– public string DrebpgStr – текстове представлення округленого значення кількості відборів у захисті, виконаних гравцем в середньому за гру;

– public string AsspgStr – текстове представлення округленого значення кількості асистувань, виконаних гравцем в середньому за гру протягом сезону;

– public string FgapStr – текстове представлення округленого значення кількості потенційних очок в середньому за гру сезону;

– public string TptpgStr – текстове представлення округленого значення кількості триочкових, виконаних в середньому за гру сезону;

– public string FoulspgStr – текстове представлення округленого

значення кількості порушень правил, виконаних гравцем в середньому за гру протягом сезону;

– public string BlockspgStr – текстове представлення округленого значення кількості блокувань, виконаних гравцем в середньому за гру протягом сезону;

– public string StealspgStr – текстове представлення округленого значення кількості втрат гравця за одну гру сезону в середньому;

– public string PerStr – текстове представлення округленого значення показника PER гравця за всі ігри сезону;

– public string PppStr – текстове представлення округленого значення показника PPP гравця за всі ігри сезону;

– public virtual Team Team – об’єкт команди, за яку виступає гравець;

– public virtual Player Player – об’єкт гравця;

– public virtual ICollection<CompetitionSeason> CompetitionSeasons – сезони змагань.

Клас сезону змагання CompetitionSeason включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер сезону змагання;

– public int CompetitionId – номер змагання;

– public DateTime Firstgame – дата першої гри сезону;

– public DateTime Lastgame –дата останньої гри сезону;

– public string FirstgameStr –текстове представлення дати першої гри сезону;

– public string LastgameStr –текстове представлення дати останньої гри сезону;

– public string Name – назва сезону, що складається з пари «рік початку – рік завершення» сезону;

– public virtual Competition Competition – об’єкт, що містить повну інформацію про змагання;

– public virtual PlayerCompetitionSeason PlayerCompetitionSeasons –

колекція загальної статистики участі всіх гравців у сезоні.

Клас користувача User включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер користувача;

– public string name – ім’я користувача;

– public string login – логін користувача;

– public string password – пароль користувача;

– public virtual ICollection<Competition> Competitions – змагання, уподобані користувачем;

– public virtual ICollection<Team> Teams – команди, уподобані користувачем;

– public virtual ICollection<Player> Players – гравці, уподобані користувачем.

Клас команди Team включає такі атрибути, для яких визначено методи для встановлення і отримання значень:

– public int Id – номер користувача;

– public string name – назва команди;

– public string city – місто, яке представляє команда.

Розроблені класи контролерів:

– CompetitionController – контролер вебсторінок, пов’язаних зі змаганнями і їх сезонами;

– GameController – контролер вебсторінок, пов’язаних з іграми, участю в них команд і гравців;

– PlayerController – контролер вебсторінок, пов’язаних з гравцями;

– UserController – контролер вебсторінок, пов’язаних з авторизованим користувачем;

– TeamController – контролер вебсторінок, пов’язаних з командами і їх участю в змаганнях.

Для кожного контролера було створено відповідний набір вебсторінок на основі шаблону [12], що надають можливості відображення і керування взаємодією за відповідними наборами сценаріїв.

3.3 Висновки за розділом 3

У даному розділі представлено алгоритм функціонування програми в режимі роботи звичайних користувачів, в тому числі авторизованих і неавторизованих, описано реалізовані класи і їх структуру.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення програми

Програма призначена для надання користувачам інформації про результати і статистику баскетбольних змагань з можливістю отримання індивідуальної траєкторії такого перегляду. Програма надає можливість перегляду результатів баскетбольних змагань, окремих матчів, статистичних показників виступу гравців у цих матчах, узагальнених показників гравців та команд за виступ у сезоні турніру, організовувати індивідуальну траєкторію перегляду результатів виступу команд і гравців, створюючи переліки уподобаних гравців, команді змагань.

4.2 Умови виконання програми

Апаратні вимоги до сервера:

– жорсткий диск з вільним простором – 1 Гб;

– оперативна пам’ять – 4 Гб;

– процесор: 4 ядра, 2,8 ГГц частота.

Клієнт представлений браузером користувача. Для виконання вимог до нього потрібний встановлений сучасний браузер з доступом до мережі інтернет.

4.3 Виконання програми

Програма має два окремі інтерфейси. Доступ до них реалізується окремо.

Панель адміністратора доступна за іменем вебсервера з додаванням «/admin» після цього імені. В результаті відкривається сторінка авторизації.

Після успішної авторизації користувачу відкривається сторінка роботи над змаганнями (рис. 4.1).

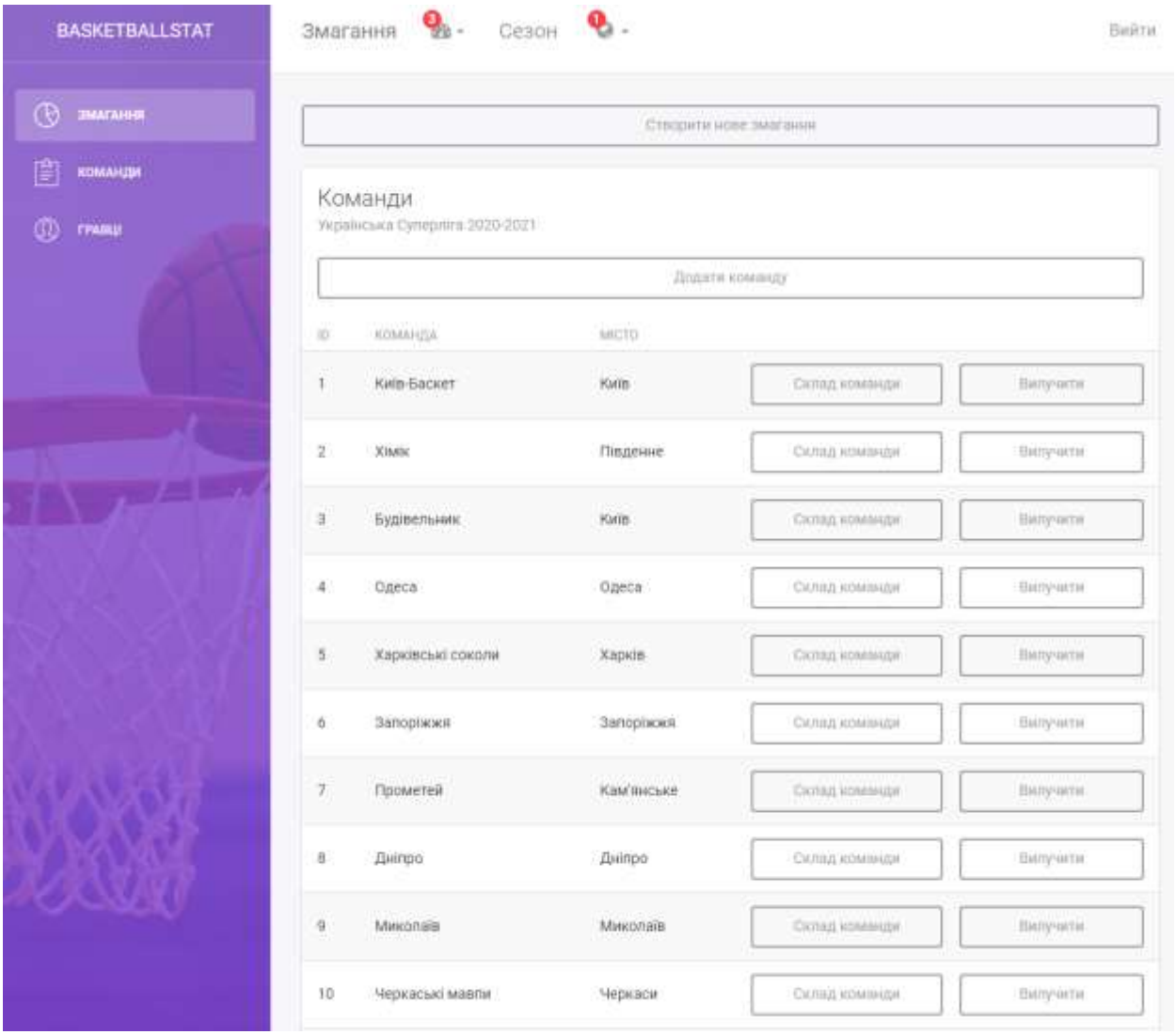


Рисунок 4.1 – Панель адміністратора при роботі зі змаганням

На даній сторінці адміністратору доступна можливість створити нове змагання. Для цього потрібно натиснути на кнопку «Створити нове змагання», тоді відкриється сторінка внесення інформації про нове змагання. Після успішного створення змагання воно буде доступне у заголовку цієї сторінки в переліку «Змагання». Даний перелік містить назви всіх змагань, доступних для роботи. Для роботи з відповідним змаганням потрібно обрати його з переліку. Число біля іконки вказує на загальну кількість доступних змагань в застосунку на даний момент. Після вибору змагання потрібно

обрати сезон проведення змагання. Сезон також доступний через випадаючий перелік. Число на іконці вказує на загальну кількість сезонів обраного змагання. Після вибору потрібного сезону (за замочуванням вибирається останній) на сторінці відображається набір команд і ігор обраного сезону обраного змагання.

На відповідній картці нижче на сторінці доступний перелік команд, що беруть участь в змаганні в даному сезону. Для того щоб додати команду, потрібно натиснути на кнопку «Додати команду», тоді відкриється окрема сторінка, де можна буде приєднати іншу команду до турніру. Важливо відзначити, що це не створення команди, а тільки додавання існуючої команди до змагання.

Нижче на картці виведено перелік усіх команд, які вже було додано до змагання. За кожною командою виводиться її назва та місто базування. Для кожної команди можна натиснути на кнопку «Склад команди» у відповідному цій команді рядку. За натисканням на цю кнопку відкривається сторінка керування складом команди. На цій сторінці можна змінювати гравців, які виступають за команду в даному сезоні.

Для кожної команди доступна також кнопка «Вилучити», що дозволяє за необхідності вилучити команду з проведення змагання в цьому сезоні, якщо раніше її було помилково додано.

Нижче даної картки розташовано наступну, на якій задається календар ігор. Тут також можна або додати нову гру, або редагувати існуючі. Всі ігри згруповані за турами. Таким чином їх зручніше сприймати. Окрім того для кожної гри доступна кнопка «Статистика», що дозволяє на окремій сторінці задати результат проведення гри, статистичні показники виступу всіх гравців, які потім будуть доступні користувачам через їхній інтерфейс.

Адміністратор вводить тільки показники за грою (загальний результат і показники окремих гравців), на основі цього автоматично розраховуються показники виступу команди в грі, а також виступу команди та гравців у сезоні.

В адміністративній панелі доступне меню, яке включає наступні пункти:

– «Змагання» – пункт, який дозволяє керувати змаганнями;

– «Команди» – пункт, який дозволяє створювати команди, які потім можна буде додавати до змагань;

– «Гравці» – пункт, який створює гравців та тренерів, яких потім можна буде додавати до команд.

У заголовку в крайній правій позиції розташовано пункт «Вийти» для того щоб вийти з облікового запису адміністратора.

Для звичайного користувача доступні наступні пункти меню:

– «Змагання», що дозволяє переглядати результати потрібного змагання, статистичні показники ігор змагання, виступу гравців в цих іграх;

– «Пошук», що дозволяє виконувати пошук команд і гравців;

– «Профіль», що дозволяє керувати даними власного профіля користувача (змінювати пароль);

– «Улюблені», що дозволяє керувати переліком улюблених змагань, команд і гравців.

На сторінці «Змагання» користувачу виводяться дані змагання. Для визначення змагання в заголовку сторінки користувач може обрати одне з доступних змагань та задати потрібний з доступних сезонів.

Після вибору потрібного змагання та сезону відображається таблиця змагання з даними за кожною командою (рис. 4.2):

– місцем команди;

– назвою команди;

– кількість ігор команди в сезоні;

– кількість перемог у сезоні;

– кількість програшів у сезоні;

– кількість очок, набраних у сезоні командою та її суперником;

– різниця набраних очок.

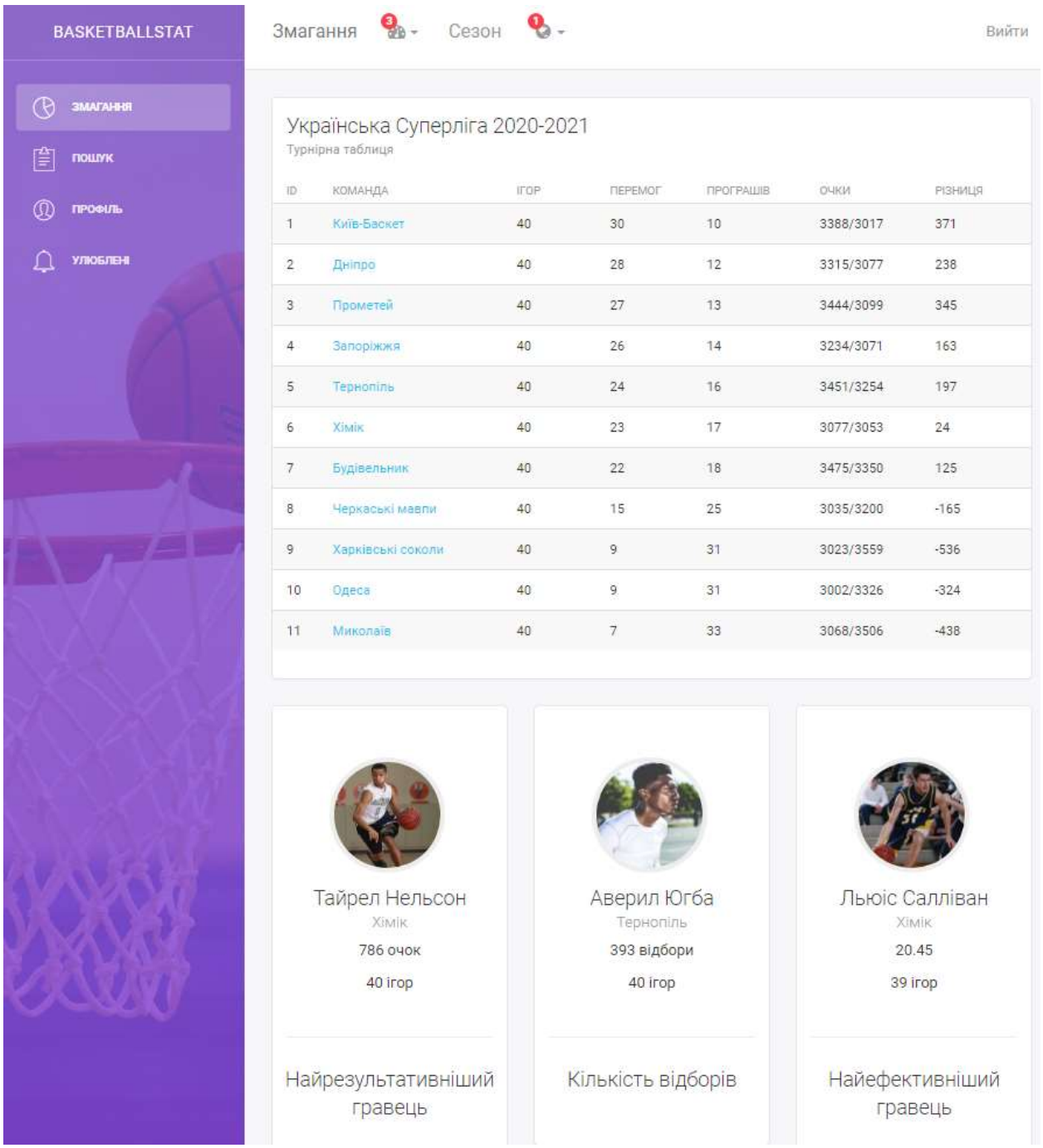


Рисунок 4.2 – Перегляд статистики змагання користувачем

Для переходу на сторінку команди (рис. 4.3) потрібно натиснути на назву команди в таблиці.

Нижче таблиці змагання розташовано картки з найкращими гравцями змагання. На кожній картці відображається найкращий гравець за одним з показників. Якщо натиснути на ім’я гравця, то можна перейти до сторінки

гравця, де буде виведено індивідуальні дані. За обраним сезоном при цьому виводиться узагальнена статистика по гравцю, а також дані по кожній грі.

На цій сторінці нижче розташовано також перелік усіх ігор змагання, звідки можна перейти до сторінки з показниками обраної гри.

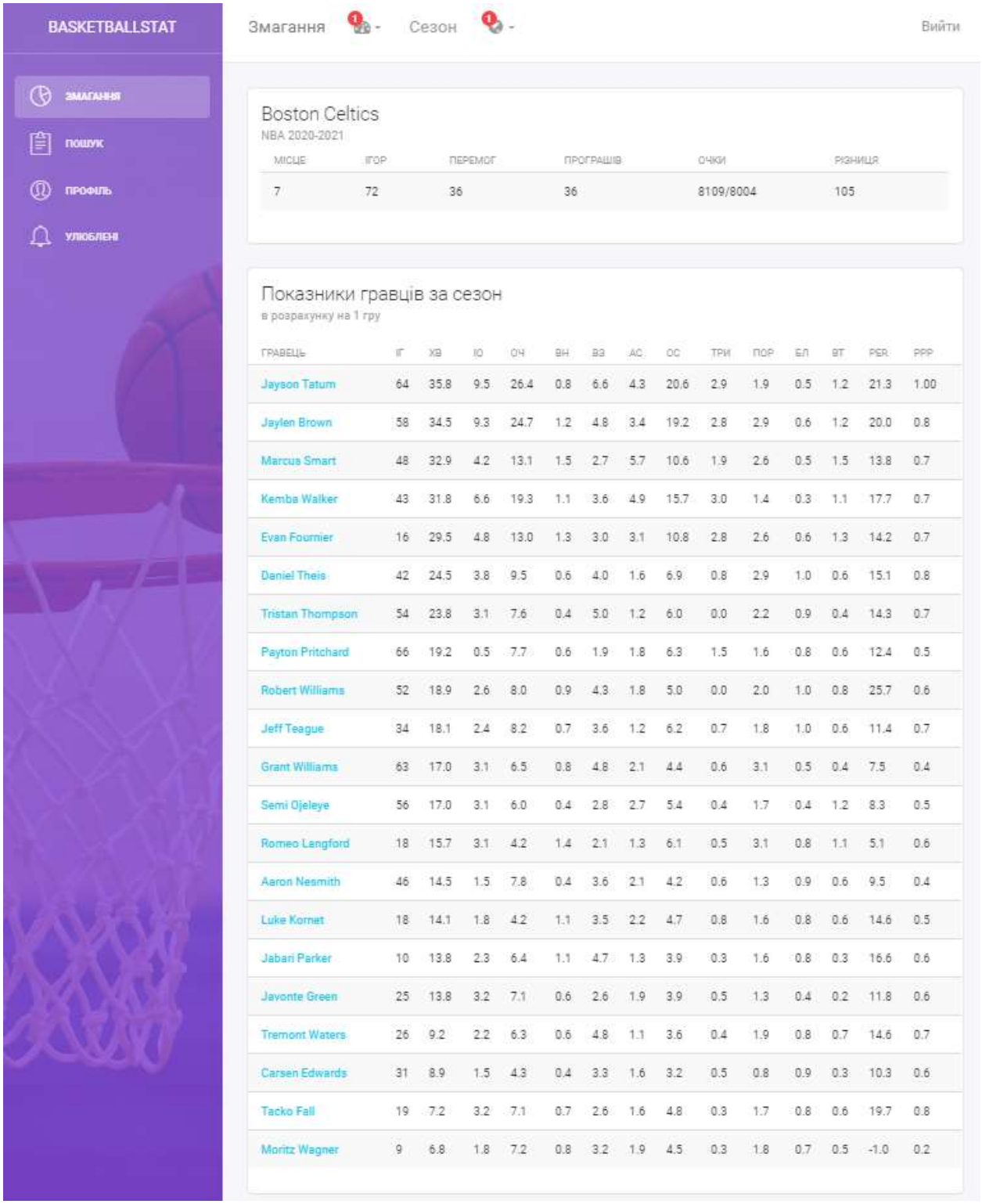


Рисунок 4.3 – Сторінка команди

На сторінці команди можна обрати потрібне змагання і його сезон. У цьому змаганні відображається спочатку результат команди загальний, потім узагальнені показники усіх гравців команди у даному сезоні, а нижче таблиця з переліком усіх ігор, в яких брала команда участь.

Сторінка «Улюблені» дозволяє керувати переліком власних уподобань (рис. 4.4).

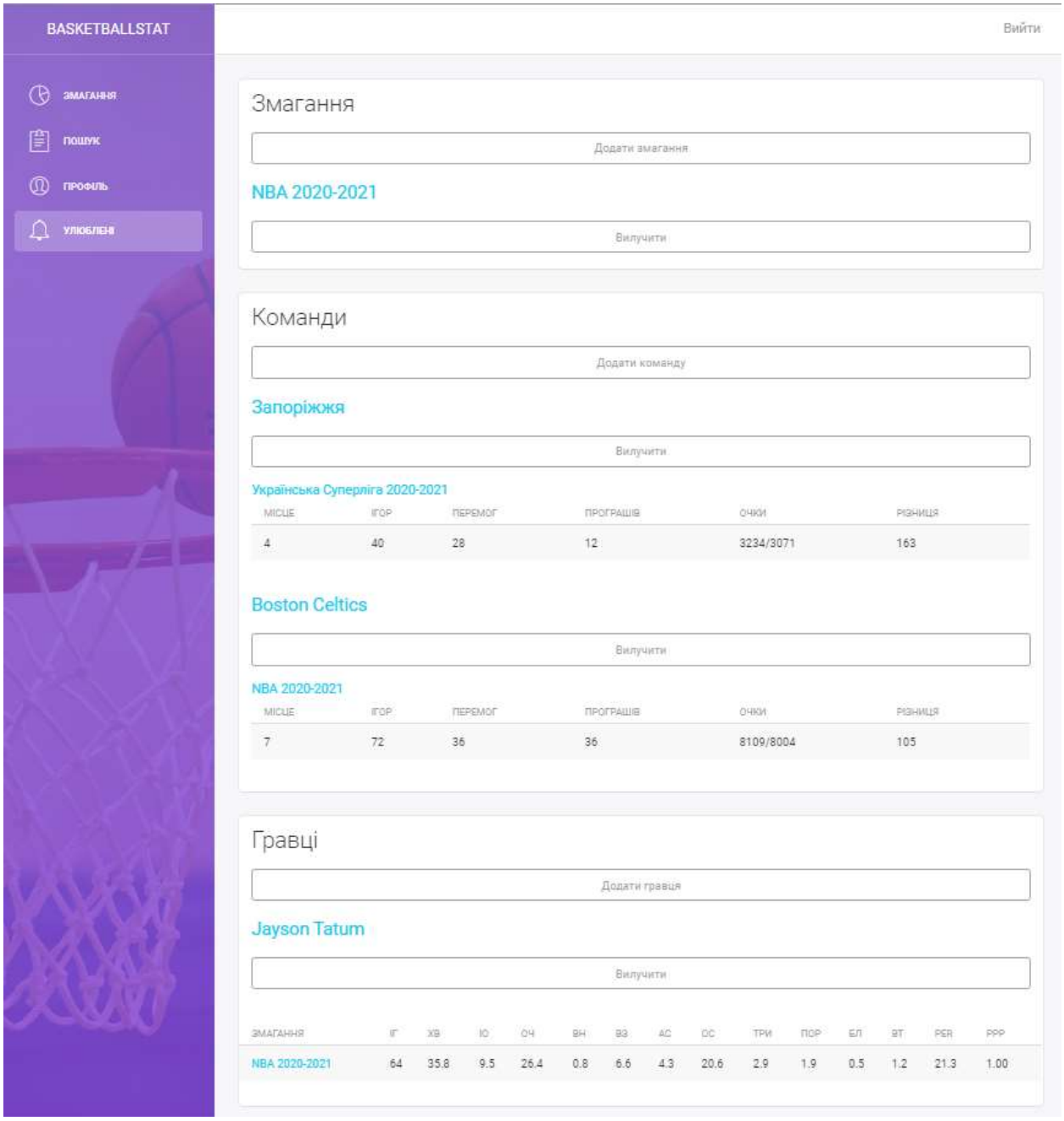


Рисунок 4.4 – Індивідуальна траєкторія роботи з програмою

На даній сторінці відображаються три картки. Перша картка – улюблені змагання, друга – улюблені команди, третя – улюблені гравці. На кожній картці присутня кнопка «Додати», що дозволяє додати до улюблених відповідне змагання, команду або гравця відповідно. За кожним елементом можна натиснути на його назву і перейти до сторінки деталізації змагання, команди або гравця. За допомогою кнопки «Вилучити» можна вилучити з переліку улюблених змагання, команду або гравця. За командами і гравцями відображаються також всі змагання останнього сезону, в яких ця команда або гравець беруть участь.

ВИСНОВКИ

За результатами виконання дипломної кваліфікаційної роботи виконано:

– аналіз предметної області, який дозволив визначити відомі статистичні показники виступу гравців у баскетбольних змаганнях, проаналізувати існуючі вебсайти баскетбольної статистики, за результатами чого виявлено, що існуючі рішення не мають індивідуальної траєкторії відслідковування статистики;

– проєктування програмного забезпечення, під час якого визначено функціональні вимоги для роботи неавторизованого, авторизованого користувачів та адміністратора з програмою, проведено моделювання взаємодії цих користувачів з програмою у вигляді діаграм прецедентів, обрано мову програмування C# та технологію веброзробки ASP.NET для розроблення програмного забезпечення, виконано проєктування бази даних, в результатів чого створено схему бази даних, визначено архітектуру вебзастосунку на основі трьох основних шарів, що відділяють роботу з даними від представлення цих даних користувачу;

– реалізацію і тестуванню програмного забезпечення, що дозволило створити вебзастосунок, який надає можливість перегляду результатів баскетбольних змагань, окремих матчів, статистичних показників виступу гравців у цих матчах, узагальнених показників гравців та команд за виступ у сезоні турніру, організовувати індивідуальну траєкторію перегляду результатів виступу команд і гравців, створюючи переліки уподобаних гравців, команді змагань.

Завдання дипломної кваліфікаційної роботи в результаті повністю виконано.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Top 10 Most Popular Sposts in The World [Electronic resource]. – Access mode : https://sportsshow.net/top-10-most-popular-sports-in-the-world/

2. Khan, E. Advanced NBA Stats for Dummies: How to Understand the New Hoops Math [Electronic resource] / E. Khan. – Access mode : https://bleacherreport.com/articles/1813902-advanced-nba-stats-for-dummies-how-to-understand-the-new-hoops-math

3. The Numbers Advantage [Electronic resource] : How to Use Statistics as a Tool in Basketball. – Access mode : https://www.ussportscamps.com/tips/basketball/how-to-use-statistics-as-a-tool-in-basketball

4. Haefner, J. 9 Stats That Every Serious Basketball Coach Should Track [Electronic resource] / J. Haefner. – Access mode : https://www.breakthroughbasketball.com/stats/9_stats_basketball_coach_should_track.html

5. Taylor, B. Nylon Calculus [Electronic resource] : What's the best advanced stat? / B. Taylor. – Access mode : https://fansided.com/2019/01/08/nylon-calculus-best-advanced-stat/

6. NBA - National Basketball Association Teams, Scores, Stats, News, Standings, Rumors – ESPN [Electronic resource]. – Access mode : https://www.espn.com/nba/

7. Basketball Statistics and History | Basketball-Reference.com [Electronic resource]. – Access mode : https://www.basketball-reference.com/

8. Scoreboard.com – Live MLB scores, NFL, NBA, NHL, Golf and Tennis Scores [Electronic resource]. – Access mode : https://www.scoreboard.com/ru/

9. Basketball Stats for Players, Teams, Leagues, worldwide | Proballers [Electronic resource]. – Access mode : https://www.proballers.com/

10. Gutsch, J. Customizing ASP.NET Core 5.0 [Text] : Turn the right

screws in ASP.NET Core to get the most out of the framework / J. Gutsch, D. Bowden. – Birmingham : Packt Publishing, 2021. – 160 p.

11. Advantages and Disadvantages of ASP.NET | Software Developer India [Electronic resource]. – Access mode : https://www.software-developer-india.com/advantages-and-disadvantages-of-asp-net/

12. Light Bootstrap Dashboard [Electronic resource] : Free Bootstrap 4 Admin Template @ Creative Tim. – Access mode : https://www.creative-tim.com/product/light-bootstrap-dashboard

ДОДАТОК А

Технічне завдання

Вступ

Вебсайти баскетбольної статистики є достатньо відокремленими рішеннями. Найчастіше такі рішення мають функціональність тільки перегляду результатів і статистики і не завжди мають навіть можливість авторизації, відповідно це не сприяє постійному залученню користувача. З іншого боку таке відокремлення призводить до того, що існуючі функції не піддаються індивідуалізації. Необхідно знайти можливості індивідуалізації статистичних функцій для тіснішого залучення користувачів до роботи програми.

А.1 Підстави для розробки

Підставою для розробки було завдання на дипломну кваліфікаційну роботу за темою «Програмне забезпечення слідкування за баскетбольними змаганнями» на основі наказу № 103 від 30 березня 2021 р. за Національним університетом «Запорізька політехніка».

А.2 Призначення розробки

Програма призначена для надання користувачам інформації про результати і статистику баскетбольних змагань з можливістю отримання індивідуальної траєкторії такого перегляду.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Вимоги до функціональних характеристик складаються з наступних:

– додавання баскетбольних команд;

– редагування даних баскетбольних команд;

– внесення складу баскетбольної команди;

– керування складом баскетбольної команди;

– редагування даних гравців;

– додавання баскетбольного змагання;

– внесення ігор баскетбольного змагання;

– редагування внесених ігор;

– внесення результатів баскетбольних матчів;

– внесення статистики баскетбольного матчу;

– редагування внесених даних матчів;

– перегляд таблиці змагання;

– перегляд результатів матчів змагання;

– перегляд статистичних показників команд у матчі;

– перегляд статистичних показників гравців у матчі;

– перегляд інформації про гравця;

– перегляд узагальненої статистики гравця за змаганнями;

– пошук гравців;

– пошук команд;

– перегляд узагальненої статистики команди за змаганнями;

– перегляд кращих гравців за статистичними показниками за турніром;

– перегляд кращих гравців за статистичними показниками у команді;

– авторизація;

– реєстрація користувачів;

– визначення улюблених команд:

– визначення улюблених гравців;

– визначення улюблених змагань;

– керування переліком улюблених команд:

– керування переліком улюблених гравців;

– керування переліком улюблених турнірів;

– відображення статистики улюблених гравців;

– відображення статистики улюблених команд;

– керування профілем користувача.

А.3.2 Вимоги до надійності

Внесення даних має відбуватися тільки через адміністративну панель, доступ до якої має відбуватися за окремою адресою. Доступ звичайних користувачів, авторизованих і неавторизованих, має реалізовуватися через вебсервер стандартно. Користувачі повинні мати можливість тільки додавати власні записи, які ніяк не впливають на дані щодо самих баскетбольних змагань.

А.3.3 Умови експлуатації

Для експлуатації програми потрібна підтримка зі сторони менеджера. Ним може бути відповідальна особа з правами адміністратора щодо роботи з системою. Менеджерів може бути декілька для підтримки зручності роботи. Внесення менеджерів у базу даних має відбуватися системним адміністратором.

А.3.4 Вимоги до складу та параметрів технічних засобів

Апаратні вимоги до сервера:

– жорсткий диск з вільним простором – 1 Гб;

– оперативна пам’ять – 4 Гб;

– процесор: 4 ядра, 2,8 ГГц частота.

Клієнт представлений браузером користувача. Для виконання вимог до нього потрібний встановлений сучасний браузер з доступом до мережі інтернет.

А.4 Порядок контролю та приймання

Контроль за виконанням дипломної кваліфікаційної роботи покладається на керівника. Остаточне рішення приймається екзаменаційною комісією під час захисту роботи.

ДОДАТОК Б

Текст програми

Б.1 Текст файла Player.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Threading.Tasks;

namespace BasketballStat.Models
{
    public class Player
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id
        {get; set; }
        public string Name
        {get; set; }
        public string Surname
        {get; set; }
        public DateTime Birth
        {get; set; }
        public bool Status
        {get; set; }
        public int Height
        {get; set; }
        public int Weight
        {get; set; }
        public string Photo
        {get; set; }
        [NotMapped]
        public string StatusStr
        {
            get
            {
                if (this.Status == true)
                {
                    return "активний";
                }
                else
                {
                    return "не виступає";
                }
```

```
                }
            }
           [NotMapped]
            public string BirthStr
            {
                get
                {
                    return this.Birth == null ? "не вказано" :
((DateTime)this.Birth).ToString("dd.MM.yyyy");
                }
            }
            public virtual ICollection<PlayerGame> PlayerGames
            {get; set; }
            public   virtual   ICollection<PlayerCompetitionSeason>
PlayerCompetitionSeasons
            {get; set; }
        }
    }
```

Б.2 Текст файла Competition.cs

```
    using System;
    using System.Collections.Generic;
    using System.ComponentModel.DataAnnotations;
    using System.ComponentModel.DataAnnotations.Schema;
    using System.Linq;
    using System.Threading.Tasks;

    namespace BasketballStat.Models
    {
        public class Competition
        {
            [Key]
            [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            public int Id
            {get; set; }
            public string Name
            {get; set; }
            public string Country
            {get; set; }
            public int Type
            {get; set; }
            [NotMapped]
```

```
            public string TypeStr
            {
                get
                {
                    if (this.Type == 0)
                    {
                        return "мінімальний";
                    }
                    else if (this.Type == 1)
                    {
                        return "скорочений";
                    }
                    else
                    {
                        return "повний";
                    }
                }
            }
            public       virtual       ICollection<CompetitionSeason>
CompetitionSeasons
            {get; set; }
        }
    }
```

Б.3 Текст файла TeamCompetitionSeason.cs

```
    using System;
    using System.Collections.Generic;
    using System.ComponentModel.DataAnnotations;
    using System.ComponentModel.DataAnnotations.Schema;
    using System.Linq;
    using System.Threading.Tasks;

    namespace BasketballStat.Models
    {
        public class TeamCompetitionSeason
        {
            [Key]
            [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            public int Id
            {get; set; }
            public int TeamId
            {get; set; }
            public int CompetitionseasonId
```

```
        {get; set; }
        public int Games
        {get; set; }
        public int Win
        {get; set; }
        public int Loss
        {get; set; }
        public int Pointsfor
        {get; set; }
        public int Pontsagaints
        {get; set; }
        public int Place
        {get; set; }
        [NotMapped]
        public int Diff
        {
            get
            {
                return this.Pointsfor – this.Pontsagaints;
            }
        }
        public virtual Team Team
        {get; set; }
        public virtual CompetitionSeason CompetitionSeason
        {get; set; }
    }
}
```

Б.4 Текст файла Game.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Threading.Tasks;

namespace BasketballStat.Models
{
    public class Game
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
```

```
public int Id
{get; set; }
public int Team1Id
{get; set; }
public int Team2Id
{get; set; }
public DateTime Start
{get; set; }
public int Goals1
{get; set; }
public int Goals2
{get; set; }
[NotMapped]
public int Diff
{
    get
    {
        return this.Goals1– this.Goals2;
    }
}
public int Tour
{get; set; }
public int Off
{get; set; }
public int Fg1
{get; set; }
public int Fg2
{get; set; }
public int Tpt1
{get; set; }
public int Tpt2
{get; set; }
public int Ft1
{get; set; }
public int Ft2
{get; set; }
public int Oreb1
{get; set; }
public int Dreb1
{get; set; }
public int Oreb2
{get; set; }
public int Dreb2
{get; set; }
public int Assists1
```

```
                {get; set; }
                public int Assists2
                {get; set; }
                public int Fouls1
                {get; set; }
                public int Fouls2
                {get; set; }
                public int Blocks1
                {get; set; }
                public int Blocks2
                {get; set; }
                public int Steals1
                {get; set; }
                public int Steals2
                {get; set; }
                public int Trainer1Id
                {get; set; }
                public int Trainer2Id
                {get; set; }
                public virtual Trainer Trainer1
                {get; set; }
                public virtual Trainer Trainer2
                {get; set; }
                public virtual Team Team1
                {get; set; }
                public virtual Team Team2
                {get; set; }
                [NotMapped]
                public string StartStr
                {
                    get
                    {
                        return this.Start == null ? "не вказано" :
((DateTime)this.Start).ToString("dd.MM.yyyy");
                    }
                }
                [NotMapped]
                public string OffStr
                {
                    get
                    {
                        if (this.Off == 0)
                        {
                            return "група";
                        }
```

```
                else if (this.Off == 1)
                {
                    return "плейофф";
                }
            }
        }
        public virtual ICollection<PlayerGame> PlayerGames
        {get; set; }
    }
}
```

Б.5 Текст файла PlayerGame.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Threading.Tasks;

namespace BasketballStat.Models
{
    public class PlayerGame
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id
        {get; set; }
        public int PlayerId
        {get; set; }
        public int GameId
        {get; set; }
        public int Points
        {get; set; }
        public int Minutes
        {get; set; }
        public int Oreb
        {get; set; }
        public int Dreb
        {get; set; }
        public int Assists
        {get; set; }
        public int Fg
```

```
        {get; set; }
        public int Fga
        {get; set; }
        public int Tpt
        {get; set; }
        public int Tpta
        {get; set; }
        public int Fouls
        {get; set; }
        public int Blocks
        {get; set; }
        public int Steals
        {get; set; }
        public int Per
        {get; set; }
        public int Ppp
        {get; set; }
        public virtual Player Player
        {get; set; }
        public virtual Game Game
        {get; set; }
    }
}
```

Б.6 Текст файла PlayerCompetitionSeason.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
using System.Linq;
using System.Threading.Tasks;

namespace BasketballStat.Models
{
    public class PlayerCompetitionSeason
    {
        [Key]
        [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
        public int Id
        {get; set; }
        public int PlayerId
```

```
{get; set; }
public int CompetitionseasonId
{get; set; }
public int number
{get; set; }
public int TeamId
{get; set; }
public int Position
{get; set; }
public int Games
{get; set; }
public float Minpg
{get; set; }
public float Fgp
{get; set; }
public float Ppg
{get; set; }
public float Orebpg
{get; set; }
public float Drebpg
{get; set; }
public float Asspg
{get; set; }
public float Fgap
{get; set; }
public float Tptpg
{get; set; }
public float Foulspg
{get; set; }
public float Blockspg
{get; set; }
public float Stealspg
{get; set; }
public float Per
{get; set; }
public float Ppp
{get; set; }
[NotMapped]
public string MinpgStr
{
    get
    {
        return Math.Round(this.Minpg, 1);
    }
}
```

```
[NotMapped]
public string FgpStr
{
    get
    {
        return Math.Round(this.Fgp, 1);
    }
}
[NotMapped]
public string PpgStr
{
    get
    {
        return Math.Round(this.Ppg, 1);
    }
}
[NotMapped]
public string OrebpgStr
{
    get
    {
        return Math.Round(this.Orebpg, 1);
    }
}
[NotMapped]
public string DrebpgStr
{
    get
    {
        return Math.Round(this.Drebpg, 1);
    }
}
[NotMapped]
public string AsspgStr
{
    get
    {
        return Math.Round(this.Asspg, 1);
    }
}
[NotMapped]
public string FgapStr
{
    get
    {
```

```
        return Math.Round(this.Fgap, 1);
    }
}
[NotMapped]
public string TptpgStr
{
    get
    {
        return Math.Round(this.Tptpg, 1);
    }
}
[NotMapped]
public string FoulspgStr
{
    get
    {
        return Math.Round(this.Foulspg, 1);
    }
}
[NotMapped]
public string BlockspgStr
{
    get
    {
        return Math.Round(this.Blockspg, 1);
    }
}
[NotMapped]
public string StealspgStr
{
    get
    {
        return Math.Round(this.Stealspg, 1);
    }
}
[NotMapped]
public string PerStr
{
    get
    {
        return Math.Round(this.Per, 1);
    }
}
[NotMapped]
public string PppStr
```

```
            {
                get
                {
                    return Math.Round(this.Ppp, 1);
                }
            }
            public virtual Team Team
            {get; set; }
            public virtual Player Player
            {get; set; }
            public       virtual       ICollection<CompetitionSeason>
CompetitionSeasons
            {get; set; }
        }
    }
```

Б.7 Текст файла CompetitionSeason.cs

```
    using System;
    using System.Collections.Generic;
    using System.ComponentModel.DataAnnotations;
    using System.ComponentModel.DataAnnotations.Schema;
    using System.Linq;
    using System.Threading.Tasks;

    namespace BasketballStat.Models
    {
        public class CompetitionSeason
        {
            [Key]
            [DatabaseGenerated(DatabaseGeneratedOption.Identity)]
            public int Id
            {get; set; }
            public int CompetitionId
            {get; set; }
            public DateTime Firstgame
            {get; set; }
            public DateTime Lastgame
            {get; set; }
            [NotMapped]
            public string FirstgameStr
            {
```

```
                get
                {
                    return this.Firstgame == null ? "не вказано" :
((DateTime)this.Firstgame).ToString("dd.MM.yyyy");
                }
            }
            [NotMapped]
            public string LastgameStr
            {
                get
                {
                    return this.Lastgame == null ? "не вказано" :
((DateTime)this.Lastgame).ToString("dd.MM.yyyy");
                }
            }
            [NotMapped]
            public string Name
            {
                get
                {
                    return
((DateTime)this.Firstgame).ToString("yyyy")       +       "-"       +
((DateTime)this.Lastgame).ToString("yyyy");
                }
            }
            public virtual Competition Competition
            {get; set; }
            public          virtual          PlayerCompetitionSeason
PlayerCompetitionSeasons
            {get; set; }
        }
    }
```

Б.8 Текст файла BasketballContext.cs

```
using BasketballStat.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace BasketballStat.Data
```

```
      {
          public class BasketballContext : DbContext
          {
              public DbSet<Player> Players
              {get; set; }
              public DbSet<Competition> Competitions
              {get; set; }
              public                        DbSet<TeamCompetitionSeason>
TeamCompetitionSeasons
              {get; set; }
              public DbSet<Game> Games
              {get; set; }
              public DbSet<PlayerGame> PlayerGames
              {get; set; }
              public DbSet<User> Users
              {get; set; }
              public                      DbSet<PlayerCompetitionSeason>
PlayerCompetitionSeasons
              {get; set; }
              protected   override   void   OnModelCreating(ModelBuilder
mbuild)
              {
                  mbuild.Entity<Player>().ToTable("Player");
                  mbuild.Entity<Competition>().ToTable("Competition");

mbuild.Entity<TeamCompetitionSeason>().ToTable("TeamCompetitionSeason"
);
                  mbuild.Entity<Game>().ToTable("Game");
                  mbuild.Entity<PlayerGame>().ToTable("PlayerGame");
                  mbuild.Entity<User>().ToTable("User");

mbuild.Entity<PlayerCompetitionSeason>().ToTable("PlayerCompetitionSea
son");
              }
      }
```

ДОДАТОК В

Слайди презентації

Рисунок В.1 – Слайд № 1

Рисунок В.2 – Слайд № 2

	Basketball Reference	Scoreboard	Розробка
Відслідковування результатів змагань	Так	Так	Так
Відслідковування статистики команд	Так	Так	Так
Відслідковування статистики гравців	Так	Ні	Так
Статистика вибраних гравців та команд	Ні	Ні	Так

Рисунок В.3 – Слайд № 3

Рисунок В.4 – Слайд № 4

Рисунок В.5 – Слайд № 5

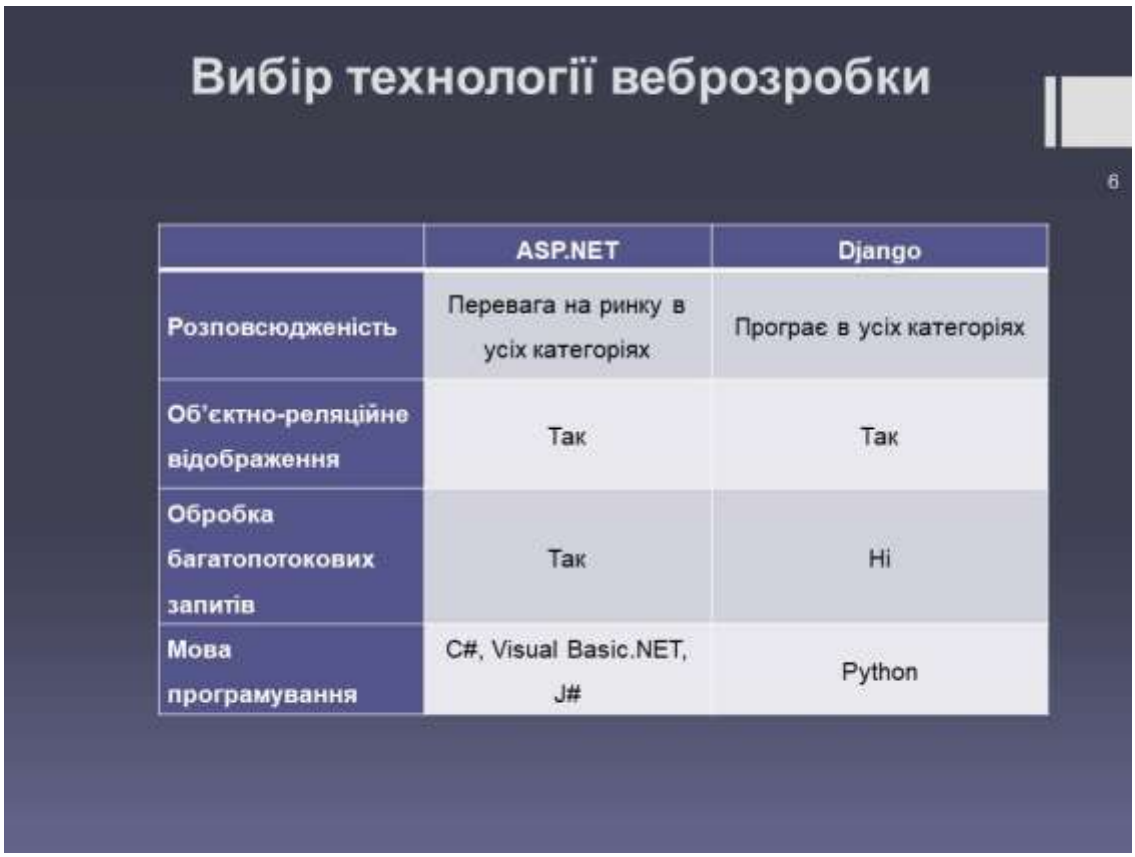


Рисунок В.6 – Слайд № 6

Рисунок В.7 – Слайд № 7

Рисунок В.8 – Слайд № 8

Рисунок В.9 – Слайд № 9

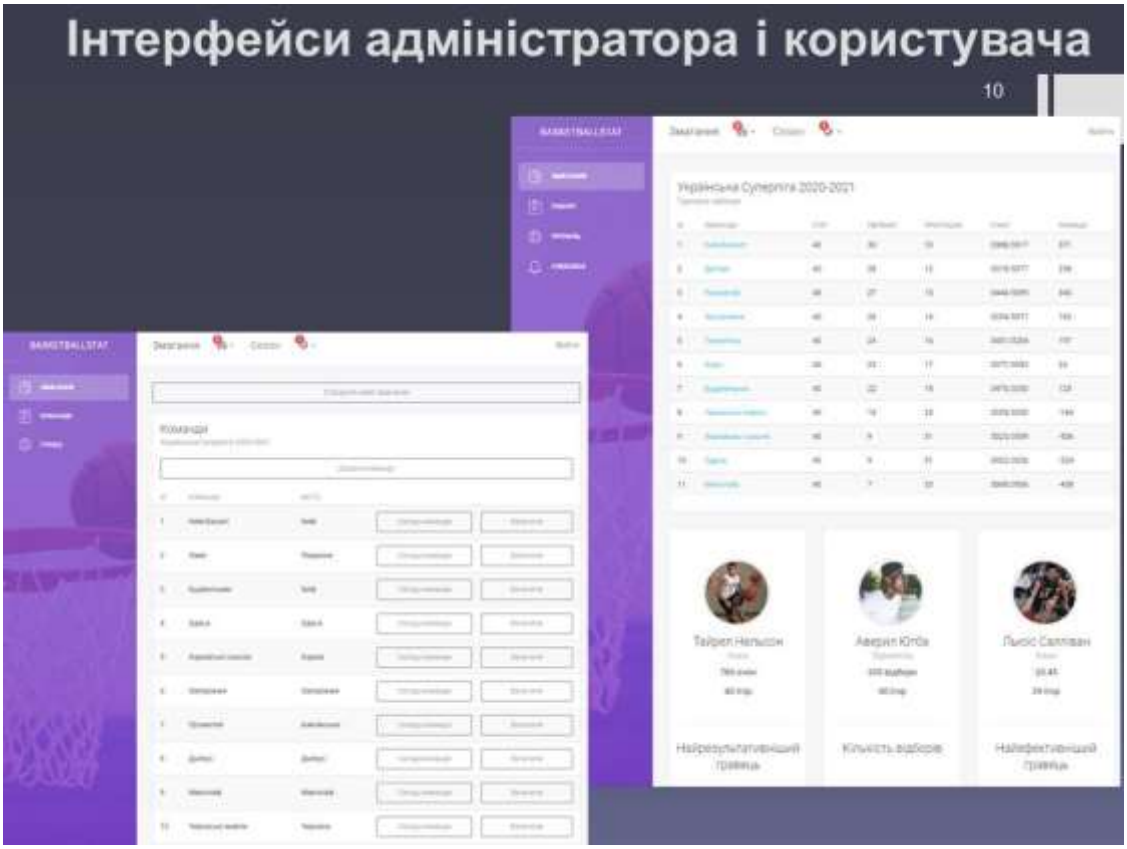


Рисунок В.10 – Слайд № 10

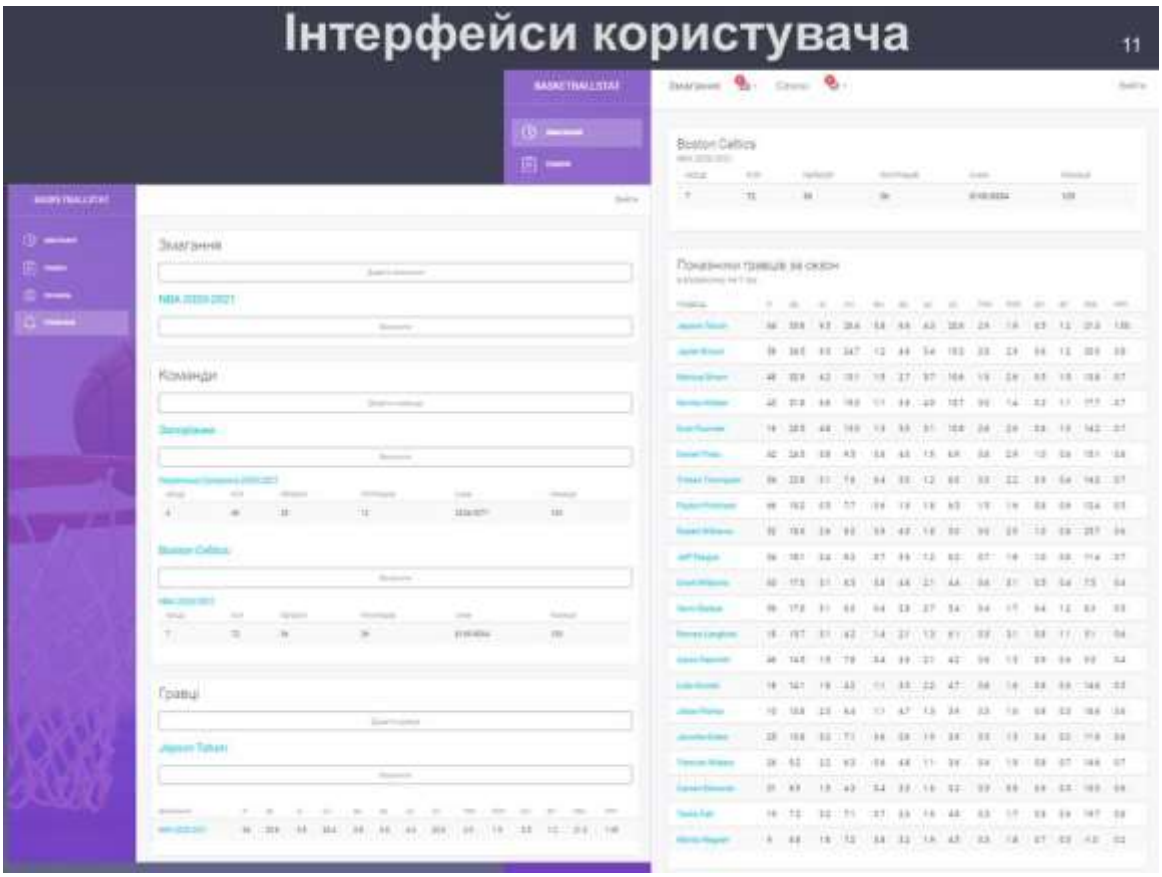


Рисунок В.11 – Слайд № 11

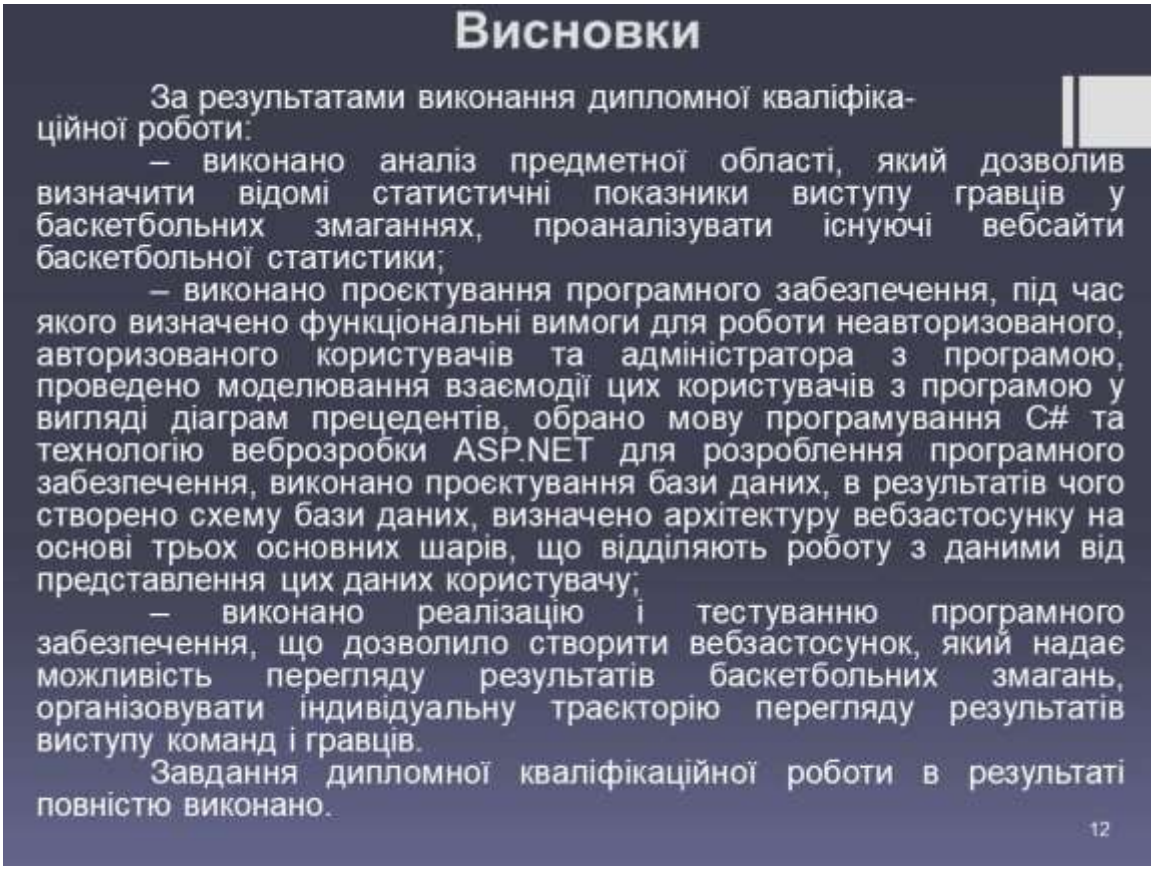


Рисунок В.12 – Слайд № 12