

БАКАЛАВРСЬКА РОБОТА

БР. ІІ - 15.00.00.000 ІЗ

Група ІІ-22-4

Федорчук Максим

2026

Івано-Франківський національний технічний університет нафти і газу

Факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Федорчук Максим Олегович

(прізвище, ім'я, по батькові)

УДК 004.942

(індекс)

БАКАЛАВРСЬКА РОБОТА

Розробка веб-додатку для аналізу та візуалізації спортивної статистики

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Робота містить результати власних досліджень, використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело:

Здобувач освітнього ступеня **Федорчук М.О.**

(підпис, ініціали та прізвище здобувача)

Науковий керівник **Романишин Тарас Любомирович, к.т.н, доцент**

(підпис, прізвище, ім'я, по батькові, науковий ступінь, вчене звання керівника)

Допущено до захисту

Завідувач кафедри

доц. Бандура В.В.

(посада)

(підпис) (дата) (ініціали та прі-

звище)

Івано-Франківський національний технічний університет нафти і газу
Інститут, факультет інформаційних технологій
Кафедра інженерії програмного забезпечення
Ступінь вищої освіти бакалавр
Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедрою _____ **ІПЗ**
доцент. _____ **В.В. Бандура**

“ _____ ” _____ 2026 р.

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТОВІ

Федорчук Максим Олегович

(прізвище, ім'я, по-батькові)

1. Тема проекту (роботи) "Розробка веб-додатку для аналізу та візуалізації спортивної статистики"

керівник проекту (роботи) Романишин Тарас Любомирович, к.т.н, доцент

затверджені наказом вищого навчального закладу від “ 21 ” квітня 2026 р. № 179/7

2. Строк подання студентом проекту (роботи) 10 червня 2026 р.

3. Вихідні дані до проекту (роботи) Результати і матеріали отримані під час проходження перекваліфікаційної практики

4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)

1. Аналіз вимог до програмного забезпечення

2. Аналіз вимог і постановка задачі

3. Проектування системи

4. Реалізація проекту

5. Тестування та впровадження

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

1 Визначення різних концепцій даних (рис. 1.1, ст. 17)

2 Процес пошуку знань з баз даних(рис. 1.3, ст. 19)

4. Архітектура мікроядра з плагінами (рис.1.12, ст. 33)

4 Модель даних для статистичної системи (рис. 3.1, ст. 61)

5 Дизайн та архітектура статистичної системи (рис. 3.3, ст. 69)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

2. Дата видачі завдання 2026 р.

Керівник

_____ (підпис)

Завдання прийняв до виконання

_____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту	Примітка
1	Визначення та обґрунтування теми роботи	17.01.2026	виконано
2	Огляд існуючих концепцій, рішень та сервісів в даній області	21.02.2026	виконано
3	Побудова моделі або алгоритму власного рішення	01.03.2026	виконано
4	Документування реалізації власного оригінального рішення вибраними засобами	21.04.2026	виконано
5	Оформлення пояснювальної записки бакалаврської роботи	02.05.2026	виконано

Студент

_____ Федорчук М.О.
(підпис)

Керівник роботи

_____ Романишин Т.Л.
(підпис)

АНОТАЦІЯ

Бакалаврська робота містить 87 сторінок, 26 рисунків, 2 таблиці, список використаних джерел із 40 найменувань.

Метою роботи є дослідження методів аналізу та візуалізації спортивної статистики, а також розробка веб-додатку для збору, обробки, аналізу та графічного представлення спортивних даних.

Об'єкт дослідження: веб-додатки для аналізу та візуалізації спортивної статистики.

Предмет дослідження: методи аналізу спортивних даних, технології веб-розробки, інструменти візуалізації статистичної інформації та підходи до побудови інтерактивних аналітичних систем.

Результати дослідження: проведено аналіз сучасних підходів до спортивної аналітики, моделювання та візуалізації спортивної статистики, реалізовано веб-додаток для інтерактивного аналізу спортивних показників.

У першому розділі розглянуто теоретичні основи спортивної статистики, методи аналізу спортивних даних та сучасні технології веб-розробки для аналітичних систем.

У другому розділі досліджено методи та інструменти розробки веб-додатків для спортивної аналітики, проведено порівняння архітектурних підходів і технологічних стеків, а також обґрунтовано вибір інструментів.

У третьому розділі розглянуто проєктування моделей даних, реалізацію веб-додатку, інтеграцію сучасних веб-технологій, тестування системи та оцінку ефективності її роботи.

Висновок: використання сучасних веб-технологій та інструментів візуалізації даних дозволяє створити ефективний веб-додаток для аналізу спортивної статистики для взаємодії з даними.

КЛЮЧОВІ СЛОВА: СПОРТИВНА АНАЛІТИКА, ВЕБ-ДОДАТОК, ВІЗУАЛІЗАЦІЯ ДАНИХ, СПОРТИВНА СТАТИСТИКА, АНАЛІЗ ДАНИХ, JAVASCRIPT, API, ІНТЕРАКТИВНІ ГРАФІКИ, ВЕБ-РОЗРОБКА.

ANNOTATION

The thesis consists of 87 pages, 26 figures, 2 table, and a reference list containing 40 sources.

The aim of the work is to study methods of sports statistics analysis and visualization, as well as to develop a web application for collecting, processing, analyzing, and graphically presenting sports data using modern web technologies.

Research object: web applications for sports statistics analysis and visualization.

Research subject: methods of sports data analysis, web development technologies, statistical data visualization tools, and approaches to building interactive analytical systems.

Research results: modern approaches to sports analytics were analyzed, methods of modeling and visualizing sports statistics were studied, architectural solutions and technology stacks were compared, and a web application for interactive analysis of sports indicators was developed.

The first section describes the theoretical foundations of sports statistics, methods of sports data analysis, and modern web development technologies for analytical systems.

The second section analyzes methods and tools for developing web applications for sports analytics, compares architectural approaches and technology stacks, and justifies the choice of implementation tools for the system.

The third section covers data model design, web application implementation, integration of modern web technologies, system testing, and evaluation of system performance efficiency.

Conclusion: the use of modern web technologies and data visualization tools makes it possible to create an effective web application for sports statistics analysis.

KEY WORDS: SPORTS ANALYTICS, WEB APPLICATION, DATA VISUALIZATION, SPORTS STATISTICS, DATA ANALYSIS, JAVASCRIPT, API, INTERACTIVE CHARTS, WEB DEVELOPMENT.

ЗМІСТ

ВСТУП.....	8
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ ВЕБ-ДОДАТКІВ ДЛЯ СПОРТИВНОЇ АНАЛІТИКИ	10
1.1 Основні принципи спортивної статистики та аналітики даних	10
1.2 Методи аналізу спортивних даних та моделювання статистичної інформації	14
1.3 Сучасні технології веб-розробки для систем спортивної аналітики	31
1.4 Висновок по розділу	49
РОЗДІЛ 2. МЕТОДИ ТА ІНСТРУМЕНТИ РОЗРОБКИ ВЕБ-ДОДАТКУ .	50
2.1 Теоретичне дослідження методів візуалізації та аналізу спортивної статистики	50
2.2 Порівняння архітектурних підходів і технологічних стеків веб-розробки ..	54
2.3 Вибір інструментів та реалізація архітектури веб-додатку	56
2.4 Висновок по розділу	61
РОЗДІЛ 3. РОЗРОБКА, ІНТЕГРАЦІЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ ВЕБ-ДОДАТКУ	63
3.1 Проектування моделей даних та структури спортивної статистики	63
3.2 Реалізація веб-додатку та інтеграція сучасних веб-технологій	68
3.3 Оцінка ефективності веб-додатку, результати тестування та перспективи розвитку системи	80
3.4 Висновок по розділу	90
ВИСНОВКИ	91
СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА	93

					БР.ІІ –15.00.00.000 ІІЗ						
Змн.	Арк.	№ докум.	Підпис	Дата	Розробка веб-додатку для аналізу та візуалізації спортивної статистики			Літ.	Арк.	Акрушіє	
Розроб.		Федорчук М.О.									
Перевір.		Романишин Т.Л.								7	
Реценз.		Корнута В.А.						ІФНТУНГ ІІ-22-3			
Н. Контр.		Піх М.М.									
Затверд.		Бандура В. В.									
					Пояснювальна записка						

ВСТУП

У сучасному світі спортивна аналітика стає важливою складовою професійного спорту, кіберспорту, спортивного менеджменту та медіаіндустрії. Постійне зростання обсягів статистичних даних про результати матчів, індивідуальні показники спортсменів, командні рейтинги та аналітичні метрики вимагає використання сучасних інформаційних систем для їх збору, обробки, аналізу та візуалізації. Розвиток веб-технологій, інтерактивних інтерфейсів та інструментів аналізу даних дозволяє створювати веб-додатки, які забезпечують швидкий доступ до статистичної інформації, її наочне представлення та підтримку прийняття рішень у спортивній сфері.

Актуальність теми зумовлена необхідністю створення ефективного веб-додатку для аналізу та візуалізації спортивної статистики, який забезпечуватиме інтерактивне представлення даних, швидкий аналіз показників та зручний доступ до статистичної інформації. Існуючі спортивні платформи часто мають обмеження щодо функціональності, швидкодії або можливостей візуалізації даних, а також можуть бути складними для адаптації під потреби конкретних користувачів. У сучасних умовах цифровізації спорту та активного використання аналітики для прогнозування результатів і оцінювання ефективності спортсменів виникає потреба у створенні гнучких і масштабованих веб-систем, здатних працювати з великими обсягами спортивних даних.

Метою роботи є розробка веб-додатку для аналізу та візуалізації спортивної статистики, який забезпечує збір, обробку, аналіз та графічне представлення спортивних даних із використанням сучасних веб-технологій та інтерактивних засобів візуалізації.

Завданнями дослідження є аналіз предметної області та сучасних підходів до спортивної аналітики, дослідження методів аналізу та моделювання спортивних даних, визначення функціональних і нефункціональних вимог до веб-додатку, порівняння архітектурних рішень та технологічних стеків, проєктування структури системи та моделей даних, реалізація основних модулів веб-додатку, інтеграція засобів візуалізації статистики, тестування системи та оцінка ефективності її роботи.

					БР.ІІ - 15.00.00.000 ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

Об’єктом дослідження є процеси розробки веб-систем для аналізу та візуалізації спортивної статистики, що включають збір, зберігання, обробку, аналіз та відображення спортивних даних.

Предметом дослідження є методи аналізу спортивних даних, підходи до побудови інтерактивних веб-інтерфейсів, технології веб-розробки та інструменти візуалізації статистичної інформації.

Методи дослідження включають аналіз наукових джерел для вивчення сучасних методів спортивної аналітики, порівняльний аналіз веб-технологій та архітектурних підходів, моделювання структури даних і бізнес-логіки системи, експериментальний метод для реалізації та тестування веб-додатку, а також статистичний аналіз отриманих результатів.

Розроблений веб-додаток передбачатиме можливість перегляду спортивної статистики, аналізу результатів команд і спортсменів, інтерактивної візуалізації даних за допомогою графіків та діаграм, а також фільтрації та пошуку статистичної інформації. Особлива увага приділятиметься швидкодії системи, масштабованості, зручності інтерфейсу та інтерактивності засобів візуалізації, що сприятиме ефективному аналізу спортивних даних та підвищенню якості прийняття рішень у спортивній сфері.

Бакалаврська робота містить 87 сторінку, 26 рисунків, 2 таблиці, 3 розділи, список використаних джерел із 40 найменувань.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						10
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ РОЗРОБКИ ВЕБ-ДОДАТКІВ ДЛЯ СПОРТИВНОЇ АНАЛІТИКИ

1.1 Основні принципи спортивної статистики та аналітики даних

Спорт і технології багато в чому пов'язані між собою. Протягом історії технології сприяли постійному розвитку спортивного спорядження, дозволяючи спортсменам досягати кращих результатів, підвищувати функціональність і виводити змагання на новий рівень [1]. Це не лише покращило сам процес тренування завдяки кращим та досконалішим інструментам, але й створило можливість отримувати кращі вимірювання та точніший зворотний зв'язок щодо результатів.

Розвиток інформаційних технологій досяг прогресу, і деякі важливі переваги щодо збору спортивної статистики були покращені. Спортивна статистика є великим активом і допомагає вже давно [2] не лише для отримання резюме спортивної гри чи сезону з точки зору очок чи голів. Вона також може бути корисною для спортивних клубів або окремих спортсменів, щоб спростити прийняття рішень на багатьох різних рівнях, наприклад, для необхідності задрафтувати швидкого футболіста до клубу або визначити область на баскетбольному полі, де команда набирає найбільше очок. Завдяки сучасним інформаційним технологіям можна зібрати повний обсяг статистики. Великі обсяги даних можуть бути основою для аналізу важливих організаційних питань або до рівня деталізації певних матчевих ситуацій, щоб побачити, що потрібно змінити для досягнення найкращого результату. Це також може запропонувати аспекти продуктивності, які раніше були невідомі, або більш складний аналіз суперників. Іншими словами, краща та детальніша спортивна статистика – це те, що може бути корисним для багатьох спортивно-орієнтованих організацій та ситуацій.

Збір даних під час спортивних подій є вирішальною частиною забезпечення цінних даних. При вирішенні проблеми збору даних про спортивні ігри можна використовувати багато методів, враховуючи важливість надійних

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						11
Змн.	Арк.	№ докум.	Підпис	Дата		

методів. Один із підходів полягає в тому, щоб прикріпити GPS-пристрої до гравців для збору їхнього місцезнаходження та іншої інформації. Багато різних великих спортивних ліг використовують цей метод, наприклад, у НБА (Національна баскетбольна асоціація), де вони прикріпили GPS-трекери до баскетболістів для відстеження пересування гравців та їхньої кидкової активності [3]. Прем'єр-ліга, найвищий дивізіон футболу в Англії, також використовує методи GPS [4], і навіть нижчі дивізіони в регбі, такі як Регбійний союз в Англії, намагалися використовувати GPS [5].

Інший поширений підхід полягає у використанні застосунку, за допомогою якого дані вводяться в систему вручну. У SHL (Шведська хокейна ліга) під час матчів використовується застосунок, наданий Statnet [6], де статистики повідомляють дані в систему під час матчу [7]. SSL (Svenska Superligan), найкраща ліга флорболу у Швеції, взагалі не використовує жодних застосунків, окрім ручки та паперу, під час матчів, а потім статистики вручну вводять їх після матчів. Тому, залежно від підходу до збору даних, використовуються різні методи.

Коли дані збираються, тренери та гравці використовують різні програми для поглибленого аналізу всіх подій, що відбулися в грі. У прикладі НБА програма, підключена до GPS-пристроїв, надає теплову карту ефективності кидків кожного гравця залежно від його положення на полі [3]. Представлення даних має велике значення, щоб зробити їх цінними для тренерів, але дані потрібно обробляти таким чином, щоб, наприклад, допомогти тренерам приймати цінні рішення. Різні види аналітики даних та моделі даних є корисними та допоможуть у таких ситуаціях.

Такі види збору даних повинні бути правильно оцінені бізнесом, щоб статистика оброблялася професійно та була захищена. *Sports Editing Sweden AB* [8] – це компанія, яка визнає важливість спортивної статистики. Компанія зосереджується на наданні послуг та технічних рішень для спортивних ліг та їхніх уболівальників. Вони співпрацюють з деякими з найбільших спортивних ліг Швеції, такими як SHL та SSL. *Sports Editing* – це компанія, яка охоплює технології, веб-розробку та спорт з метою допомогти спортивним лігам у їхній комунікації та

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						12
Змн.	Арк.	№ докум.	Підпис	Дата		

маркетингу. Вони також співпрацюють з постачальниками спортивної статистики ліг, щоб представляти дані спортивної статистики на веб-сайтах та інших платформах. Коли справа доходить до збору даних, Sports Editing співпрацює із зовнішніми сторонами, і основна увага приділяється обробці та представленню даних. Sports Editing також надає веб-редакторів та спортивних журналістів, які висвітлюють новини на пропонованих сайтах, щоб надати повний продукт для спортивних організацій.

Багато найбільших спортивних ліг інвестували в передові та дорогі системи статистичної звітності. Деякі спортивні ліги не мають або не можуть дозволити собі спеціалізованого постачальника спортивної статистики, тому Sports Editing хоче розширити спектр своїх послуг шляхом розробки власної статистичної програми, яка могла б задовольнити основні потреби спортивних звітів. Це дозволило б меншим лігам збирати статистику та полегшило б їм подальший перехід до спеціалізованого постачальника статистики, такого як Statnet.

Ця робота розглядає розробку цього застосунку, який слугуватиме доповненням до високотехнологічних додатків, що пропонуються спеціалізованими постачальниками. «Спортивне редагування» зацікавлене в розробці статистичної системи, яка повинна мати змогу обробляти дані спортивних репортерів інтуїтивно зрозумілим та ефективним способом для кількох різних видів спорту.

Заявлено, що застосунок буде веб-застосунком, але передбачуваний формат не визначено. Нова система статистики має забезпечити експорт багатьох різних розширених даних та статистичних даних [8]. Для надання відповідної статистики знадобляться різні види аналітики даних, а також шукається гарне графічне представлення, щоб зробити висновки на основі даних більш зручними для користувача.

Нова статистична програма потребує розробки як на стороні серверної частини, так і на стороні фронтенду. На стороні сервера потрібні розумні способи обробки статистичних даних, що надходять до системи, а також обробка та обчислення даних для надання іншої релевантної статистики. Оскільки це веб-додаток, можна використовувати багато різних серверних фреймворків та

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						13
Змн.	Арк.	№ докум.	Підпис	Дата		

технологій баз даних, а також порівняти та оцінити, які технології потрібно вибрати належним чином. Буде проведено загальну теоретичну оцінку спортивної аналітики та деякі дослідження щодо візуалізації даних.

Щодо фронтенду, метою є розробка прототипу веб-застосунку, який мав би взаємодіяти зі статистичною системою. Він має використовуватися як інструмент для звітів про матчі, так і для відображення статистики минулих матчів та її належної візуалізації. Інструмент звітів про матчі призначений для використання журналістами та статистиками для повідомлення про ігрові події шляхом взаємодії з застосунком у графічному та інтуїтивно зрозумілому вигляді. Інструмент звітів про матчі буде певною мірою оцінено та, сподіваємося, призведе до легшого та ефективнішого збору спортивної статистики [9].

Будуть деякі обмеження щодо прототипу веб-застосунку, який буде розроблено. Sports Editing значною мірою використовує середовище Linux, програми та фреймворки з відкритим кодом для своєї розробки. Це виключає деякі досить популярні фреймворки, такі як .NET-фреймворк від Microsoft, які не вписуються в цей програмний стек. Sports Editing вже встановив певний дизайн та архітектуру, і відповідно до вибору технології буде проведено розробку серверної частини статистики, яка буде синхронізована з архітектурою системи, розгортанням та додатками компанії.

Деякі технології баз даних також будуть розглянуті на відповідність потребам програми, але не так глибоко, як у звіті про повне висвітлення, і основна увага буде зосереджена на більш усталених та часто використовуваних технологіях баз даних [10]. Той самий стандарт буде присутній щодо зовнішнього вигляду та графічного дизайну прототипу, оскільки головною метою не є отримання остаточної версії, але будуть обговорені деякі дизайнерські ідеї. Прототип має служити способом тестування статистичної системи та допомогти оцінити, чи спрощує вона збір даних під час матчів.

Види спорту, які система охоплюватиме, обмежені командними видами спорту, зосереджуючись на основних видах спорту, що цікавлять спортивну редакцію. Їхні основні знання стосуються хокею та флорболу, оскільки вони

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дата		

встановили ділові контакти з такими лігами.

Буде реалізовано задуману статистичну програму відповідно до одного програмного стеку та архітектурного проекту. Теорія та попередня оцінка фреймворків призведуть до вибору методів, і буде реалізовано одну реалізацію програми. Кілька версій не будуть розроблені відповідно до різних програмних стеків, і порівняння не розглядатиме реалізовані системи. Буде просто теоретичне порівняння, а не практичне.

Ще однією особливістю статистичної системи є те, що статистичні моделі та математична аналітика будуть виконуватися на базовому статистичному рівні. Метою програми не є робити великі висновки щодо складних математичних моделей, наприклад, для розрахунку наступної покупки гравця або прогнозування подій у наступній грі. Програма використовуватиме підхід збору, розрахунку та представлення спортивної статистики належним чином, що допоможе залученим сторонам отримати знання або зробити висновки з гри чи сезону. Аналітика буде реалізована певною мірою, але на базовому рівні.

Система генеруватиме певний обсяг даних, але не зі швидкістю, що визначена як Великих даних [11]. Однак, це все одно буде розглянуто в теорії, оскільки багато великих спортивних ліг генерують дані зі швидкістю, яку можна вважати визначеною як Великих даних. Тому дуже важливо порушити цю тему під час проведення огляду літератури в галузі спортивної аналітики. Це також може бути актуальним для майбутньої роботи статистичної програми.

1.2 Методи аналізу спортивних даних та моделювання статистичної інформації

Розробка програми спортивної статистики відповідає різним аспектам, як теоретичним, так і технічним. Наступні розділи забезпечують теоретичну основу роботи у вигляді різних розділів, що стосуються спортивної аналітики та веб-розробки.

Для збору даних, моделювання їх відповідно до різних моделей даних та

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						15
Змн.	Арк.	№ докум.	Підпис	Дата		

візуалізації у зрозумілий спосіб дослідження літератури має бути різноманітним та обширним [12]. Спортивна аналітика розглядається разом із вступом до загального визначення аналітики даних згідно з визначеннями науки про дані. Обробка даних у поєднанні технологій розробки потребує теоретичного обґрунтування разом із коротким вивченням моделювання даних та математичних методів.

Оскільки веб-розробка постійно розвивається, теоретично представлено належний вступ до деяких сучасних фреймворків. Різні компоненти системи також описані більш детально, такі як технології баз даних, фронтенд- та бекенд-технології. Також є розділ більш архітектурного характеру, який розглядає огляд структури системи відповідно до набору архітектур та дизайну. Розділи, що стосуються вступу до різних програмних стеків та сучасних веб-технологій, також включені в теорію. Візуалізація даних також буде коротко розглянута теоретично з рекомендаціями та парадигмами для правильного відображення даних.

Спортивна аналітика

Потреба в статистиці для правильного аналізу спортивної гри є критично важливою [2]. Оскільки спортивна аналітика має концепцію використання спортивних даних для створення цінної статистики для аналізу, існує потреба у належних моделях даних. Існують різні підходи до обробки спортивних даних у поєднанні з аналітикою даних для створення статистики та іншої корисної інформації. Спортивна аналітика також потребує отримання правильних даних з матчів для аналізу. Для збору даних про матчі застосовуються різні технології, що забезпечують великі обсяги даних, багато з яких можна побудувати багато детальної статистики. Великий збір спортивних даних потім сприяє аналізу та прийняттю рішень у спортивних іграх [9].

Спортивна аналітика – це концепція обробки спортивних даних за допомогою аналітичних методологій, що допомагає робити цінні висновки [10]. Аналітика застосовується для визначення переваг у заняттях спортом. Висновки повинні бути отримані на основі усталених методів аналізу даних, відповідно до математичних моделей, які оцінила спортивна індустрія та використовуються

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

певним чином. Щоб мати змогу виконувати такий аналіз, необхідно уточнити визначення аналітики даних.

Дані в контексті спортивної аналітики не слід змішувати зі статистикою. Дані – це необроблений сирий елемент з інформацією, яка має бути основою для подальшої аналітики, а не прямим відображенням спортивної статистики. Оскільки існує багато термінів, пов'язаних з даними, необхідно навести деякі визначення. Деякі терміни, які часто згадуються в контексті аналітики, залежно один від одного, – це *Data Mining* (інтелектуальний аналіз даних), *Data Analytics* (аналіз даних) та *Data Analysis* (аналіз даних), які всі є частиною дослідницької галузі *Data Science* (див. рисунок 1.1).

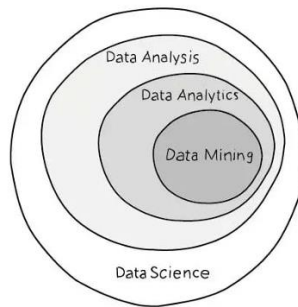


Рисунок 1.1 - Визначення різних концепцій даних

Інтелектуальний аналіз даних

Не слід плутати *Data Mining* з простим процесом збору необроблених даних. Цю концепцію слід радше розглядати як збір даних та накопичення знань за допомогою техніки пошуку закономірностей у зібраних даних [12]. Весь етап *Data Mining* також можна описати як етап, на якому ви намагаєтесь знайти взаємозв'язки в даних, щоб представити їх у більш зрозумілому вигляді [13]. Процес під назвою «*Процес Data Mining*» описує *Data Mining* більш детально та поділяє його на три кроки: збір даних, вилучення ознак та очищення даних, а також аналітична обробка та алгоритми [13] (див. рисунок 1.2).

Перший крок – це збір даних у певній формі. Це може бути що завгодно: від реєстрації взаємодії користувача з системою до надсилання даних з апаратних датчиків. Оскільки це залежить від контексту того, які дані збираються,

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						17
Змн.	Арк.	№ докум.	Підпис	Дата		

методи збору даних різняться та застосовуються на рівні програми. На другому кроці, де активується вилучення та очищення даних, дані позначаються метаданими за допомогою різних методів, щоб їх можна було класифікувати певною мірою [14]. Немає промислових чи теоретичних стандартів щодо того, як це виконується, але метою є забезпечення зрозумілої структури набору даних. Частина очищення полягає головним чином у виправленні помилок у даних, відповідно до того, що є релевантним, і дані перетворюються у потрібний формат. Третій і останній етап – це початок аналітичної частини, коли до набору даних застосовується певний метод. Використані алгоритми повинні надавати зрозумілі та читабельні результати, щоб аналітики могли виконувати свою роботу.

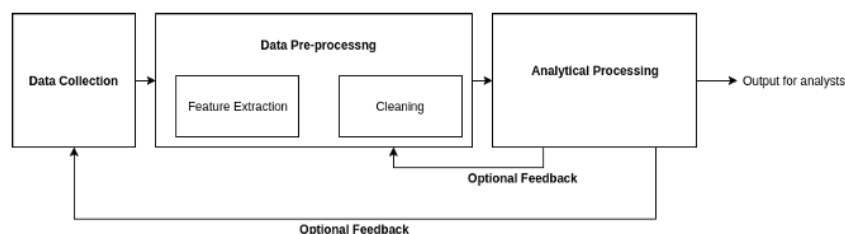


Рисунок 1.2 - Процес інтелектуального аналізу даних

Цей процес відповідає набору методів, що мають подібний характер. Подібний, але дещо більш базовий, підхід до згаданих методів data mining є підходом Блома. Він стверджує, що статистичне дослідження є певним чином фазою з чотирьох кроків: *планування, збір даних, обробка даних та представлення* [15]. Різні кроки згадуються в ширшому сенсі, але можуть бути пов'язані з data mining. Оскільки це головний фактор під час проведення статистичних досліджень, існує потреба в якісній обробці. Однак це відповідає статистичним теоріям, які дуже корисні для моделювання даних.

Інтелектуальний аналіз даних, як зазначено в моделі інтелектуального аналізу даних, може бути достатнім для багатьох контекстів, де потрібно аналізувати дані, але коли обсяг даних збільшується, може стати в пригоді певний вид аналізу даних. Варто зазначити, що існує багато різних визначень того, де закінчується процес інтелектуального аналізу даних і починається аналіз даних,

					БР.ІП - 15.00.00.000 ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

оскільки третій крок моделі інтелектуального аналізу даних можна порівняти з аналітикою даних. Обидва кроки мають на меті застосувати алгоритм, який повинен надавати факти. Ці концепції перетинаються, оскільки певною мірою виконується певний вид аналітики. Інтелектуальний аналіз даних також визначається як один із кроків у так званому *процесі KDD* [16].

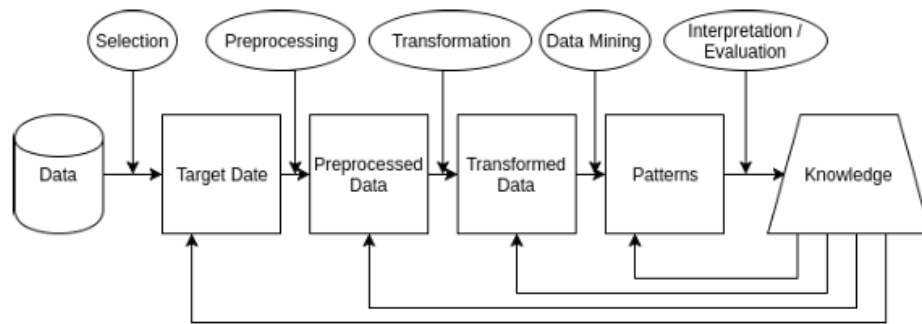


Рисунок 1.3 - Процес пошуку знань з баз даних

Іншим підходом до отримання аналітичних результатів є процес KDD. Цей процес є аббревіатурою від Knowledge Discovery from Databases (виявлення знань з баз даних) і головним чином є методом, заснованим у дев'яностих роках як відхід від більш ручних підходів до створення знань з набору даних [17] (див. рис. 1.3) . Раніше в процесі виконуються кроки для уточнення даних для досягнення широких закономірностей.

Процес починається з визначення даних, які необхідно проаналізувати, та вибору підмножини даних. Далі виконується певний вид попередньої обробки, яка очищує дані разом з ідентифікацією відсутніх властивостей. Пізніше деякі дані скорочуються для підвищення ефективності відповідно до мети. Дані також перетворюються на бажаний формат, наприклад, на різні фрейми або класи. Під час виконання методів інтелектуального аналізу даних розрізняються певні шаблони. Шаблон у контексті інтелектуального аналізу даних полягає в класифікації очищених даних відповідно до певного методу, щоб зіставити дані для певних цілей. Одним із поширених методів класифікації є кластеризація даних там, де

вони корелюють один з одним (див. рис.1.4) .

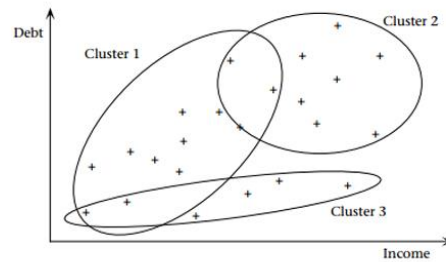


Рисунок 1.4 - Кластеризація даних за допомогою методу Data Mining

Різні кластери можуть бути корисними для різних питань, що потребують відповідей, і це надає структурі перекриваючих висновків для витягу з даних. Існує більше методів інтелектуального аналізу даних, які прийняті для процесу KDD, таких як регресія та підсумовування. Можна використовувати як лінійну, так і нелінійну версії регресії, де лінійна просто створює навчальну лінійну функцію, яка представляє кореляції (див. рис.1.5). Підсумовування - це натомість функція, яка намагається позначити підмножини даних певною характеристикою [17] [18].

Процес KDD нагадує пізніші визначення процесу Data Mining. Однак, є також дослідження, які стверджують, що Data Mining можна розглядати як окремий метод, який не обов'язково пов'язаний з процесом KDD [16].

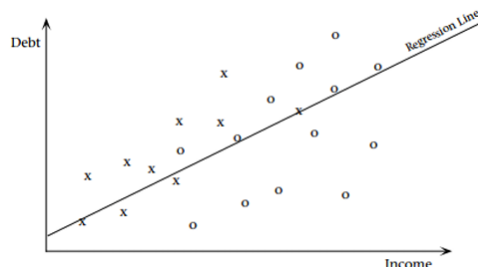


Рисунок 1.5 - Лінійна регресія, отримана за допомогою методу інтелектуального аналізу даних

Аналіз даних

Визначення аналітики даних є досить лінійним, але певною мірою варіює

					БР.ІП - 15.00.00.000 ПЗ	Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

ться залежно від науки про дані. Згідно з більшістю науки про дані, термін «аналітика даних» можна розглядати як певний програмний додаток, що використовується для автоматизації бажаних інтуїтивних знань щодо набору даних, щоб знаходити факти з даних відповідно до конкретних питань [19]. Він походить з таких галузей досліджень, як статистика, машинне навчання та штучний інтелект, і має на меті виявлення залежностей того, що система надає від різних вхідних параметрів [20]. Можуть бути використані деякі методи з інших концепцій, таких як моделі інтелектуального аналізу даних, але також алгоритми для спостереження даних у режимі реального часу можуть призвести до успішної аналітики. Ще одне базове визначення аналітики даних полягає в тому, що це автоматизований алгоритмічний процес для пошуку певних специфічних кореляцій у даних [11]. Його також можна розглядати як інструмент пошуку цінних тенденцій, прихованих для загального аналізу, які не залежать від попередніх гіпотез. Останній крок моделі інтелектуального аналізу даних можна порівняти з тим, що вона має на увазі під аналітикою даних, оскільки обидва поняття повинні знаходити закономірності в даних. Для пошуку закономірностей можна використовувати багато методів, таких як так звана *дослідницька аналітика даних*, яка використовується для виявлення зв'язків, доступних у наборі даних, які були невідомі або нечітко сформульовані [19].

Коротше кажучи, відмінність між аналітикою даних та інтелектуальним аналізом даних може полягати в тому, що якщо ви знаєте питання, на яке хочете отримати відповідь, або логіку, яку потрібно перевірити, вам слід виконати аналітику даних. Коли ви мало знаєте про дані та не маєте конкретного питання, на яке потрібно відповісти, вам слід почати з інтелектуального аналізу даних.

Аналіз даних та інтелектуальний аналіз даних широко використовуються в багатьох галузях досліджень, таких як соціальні мережі та бізнес. У соціальних мережах багато даних потрібно структурувати, а користувачі пов'язані один з одним різними способами [20]. Усі взаємовідносини в соціальних мережах є складними, і постійно відбувається взаємодія. Аналіз даних може допомогти відповісти, наприклад, на те, як мережі розвиваються з часом, а також часто

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						21
Змн.	Арк.	№ докум.	Підпис	Дата		

зустрічається в бізнес-контекстах, щоб полегшити інвестиції компанії. Його часто називають бізнес-аналітикою, і його метою є виявлення змін у галузі, на які компанія повинна реагувати найкращим чином. Процес інтелектуального аналізу даних також використовується в бізнес-контекстах, наприклад, як основа для вимірювання результату певної дії [21].

Завдяки можливостям інформатики, аналітика даних може поєднати всі ці галузі досліджень, щоб показати знання, приховані у великій масі даних. Деякі мови програмування більше підходять для виконання релевантної та ефективної аналітики. Статистична мова *R* особливо підходить для складних статистичних обчислень [22]. *MATLAB* також є передовим програмним забезпеченням, призначеним для математичних та статистичних обчислень, серед інших галузей [23]. Щодо інших мов, які передбачають одну з цих згаданих спеціалізованих мов, *Python* згадується як потужна мова для створення цінного аналізу даних, здатна імпортувати бібліотеки *R* для обчислень [24].

Аналіз даних

Аналіз даних, який не слід плутати з аналітикою даних, – це момент, з якого починається взаємодія людини зі структурованими даними. Попередні кроки повинні призвести до зрозумілої основи, яка підтримує прийняття рішень. Коротко кажучи, взаємодія людини з аналітикою здійснюється для того, щоб зробити певні висновки з досліджуваних даних. Це допомагає постійно розвивати не лише контекст, на якому вона повинна базуватися, але й ширшу картину того, чого може допомогти досягти наука про дані.

Аналізовані дані можна вивести для якісного та кількісного аналізу. Кількісний аналіз даних проводиться на основі більш числових або градуйованих наборів даних для конкретних моделей. Навіть якщо це взаємодія людини з даними, деякі комп'ютерні інструменти, що використовуються для аналізу, можуть бути ефективними [25]. Сподіваємося, що більш якісний набір інформації слід проаналізувати для генерування нових гіпотез, які можна буде додатково перевірити [26].

Великі дані

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дата		

Одним із модних слів у галузі науки про дані є великі дані (Big Data). Оскільки швидкість передачі даних зросла в останні роки, потреба в ефективній аналітиці великих даних стала дедалі важливішою. Зростання кількості інтелектуальних пристроїв призвело до вибухового зростання швидкості передачі даних, і зі збільшенням кількості датчиків та взаємодій у суспільстві необхідно впроваджувати деякі інтелектуальні рішення [27]. Вплинуло не лише зростання кількості пристроїв, але й поведінка користувачів. Така технологія, як позиціонування, генерує збільшену кількість даних, і існує багато областей, таких як бізнес-дані, дані зображень та дані промислових процесів [27]. Поведінка користувачів генерує набагато більше даних, що, за деякими прогнозами, очікується, що до 2020 року вона перевищить 240 екзабайт на день [28]. Очевидно, що існує потреба в підходах, які можуть обробляти ці великі обсяги даних, оскільки звичайної обробки даних буде недостатньо.

Під час проведення спортивної аналітики існує багато різних точок зору, які можна використовувати як відправну точку. Одним з важливих моментів є визначення даних, які потрібно обробляти. Багато програм, пов'язаних зі спортом, генерують великі обсяги даних, що створює потребу в структурованому дизайні. Великі дані мають потенціал зробити будь-яку систему розумнішою, і термін «розумні дані» може краще описувати потенціал того, як можна покращити бізнес [9]. Для структурування великих даних у розумні дані вводиться так звана *модель SMART*. *SMART розшифровується як:*

- S: Стратегія
- M: Вимірювання показників та даних
- A: Застосуйте аналітику
- R: Результати звіту
- T: Моделі перетворення

Спочатку необхідно чітко визначити стратегію, щоб розрізнити, який намір має бути досягнутий з даними. Слід витратити час на точність під час визначення стратегії. Другий етап - це перевірка даних, де відбувається збір та визначення джерел даних. Дуже важливо знати оцінку даних, оскільки вона багато

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дата		

говорить про контекст, у якому вони збираються. З огляду на це, аналітика застосовується для вилучення висновків та знань з даних, які передаються зацікавленим сторонам системи на наступному етапі. Нарешті, буде присутній етап людського аналізу, щоб визначити, які частини потрібно трансформувати та які зміни будуть внесені для покращення бізнесу [9].

З такою великою кількістю даних може бути неможливо зберегти всі дані та проаналізувати їх у повільному темпі, що може спричинити деякі проблеми зі зберіганням. Однак проблеми зі зберіганням даних незначні, якщо порівняти їх з багатьма можливостями, які відкриваються завдяки цій техніці, тим не менш у спортивній аналітиці. Більш детальні висновки можна зробити, оскільки обсяг даних збільшився, а отже, наука про дані та розпізнавання образів стають ще важливішою галуззю досліджень [29].

Наука про дані та спортивна аналітика

Як уже згадувалося, існує кореляція між спортивною аналітикою та наукою про дані. Аналіз даних вже проводився на багатьох рівнях у спорті, наприклад, НБА, НФЛ та MLS [30]. Одним із прикладів є НБА, яка надає теплові карти точності кидків [3]. Легко знайти гравця, який набрав найбільше очок, але важче виміряти, хто дійсно є найкращим стрільцем у НБА відповідно до позиції та ситуації. Згідно з деякими визначеннями того, що вважається хорошим стрільцем, стверджується, що Стів Неш на певній позиції є найефективнішим бомбардиром, а не той, хто очолює звичайну таблицю результатів. Таку статистику можна отримати за допомогою сучасних інструментів збору даних.

Хокейна ліга НХЛ, яка є одним із видів спорту з найменш розвиненою аналітикою, вже спробувала аналітику як на рівні гравців, так і на рівні команд, щоб проаналізувати, що потрібно покращити. Це може вплинути на те, як слід тренувати команди та яких гравців шукати в складі. Спортивна аналітика НХЛ розвивається, і Барнс вважає, що настав час лізі змінити свою структуру.

Інтелектуальний аналіз даних (Data Mining) вже добре зарекомендував себе у спорті, і для його структурування було розроблено деякі фреймворки. Одним із прикладів таких фреймворків є фреймворк, описаний у книзі «

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		

Інтелектуальний аналіз даних в елітному спорті: огляд та фреймворк». Його метою є підтримка рішень, а також пошук кореляцій між тим, які події в грі мали найбільший вплив певною мірою, де атрибути даних визначаються як показники ефективності. Вони визначають процес інтелектуального аналізу даних відповідно до так званої *ієрархії мудрості*, де знання отримуються шляхом перетворення необроблених даних на інформацію за допомогою логічних та математичних закономірностей даних. Коли ці знання аналізуються тренером, гравцем або будь-якою іншою особою, пов'язаною зі спортом, вони перетворюються на спортивну мудрість.

У випадку спорту аналіз повинен враховувати ключові атрибути, які можуть призвести до успіху команди, наприклад, певні події матчу або тактики. Він поділяє діапазон результатів на прямі (прості змінні) та складні (приховані висновки), а обробку даних – на п'ять категорій:

- Фільтрація: Категоризує дані.
- Перетворення формату: Перетворює дані у потрібний формат.
- Вилучення: Створює нові альтернативні дані з відформатованих даних (наприклад, рейтинг часу тощо).
- Структурне перетворення: Перетворює на вимірювані дані.
- Описове перетворення: Перетворює результати таким чином, щоб вони описували та діяли доречніше.

Коли ці кроки зроблені, встановлюється певний зв'язок між тим, які події призводять до успіху відповідно до критеріїв перемоги, наприклад, ви виграєте гру в хокей, якщо виходите вперед з коефіцієнтом 57%. Попередні матчі враховуються в розрахунках, і метою є визначення того, які критерії є найважливішими та які гравці найкраще підходять для їх задоволення [32].

Деякі види спорту важче виміряти, ніж інші, оскільки на різних етапах гри важко оцінити вплив (наприклад, триатлон). Такі змінні, як тривалість та критерії перемоги, важливі для прийняття правильних рішень. У триатлоні фактичний час може мати певне значення, тоді як важливішими можуть бути такі фактори, як те, наскільки спортсмен випереджав інших суперників [32].

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						25
Змн.	Арк.	№ докум.	Підпис	Дата		

Фактичну продуктивність також можна покращити за допомогою аналітики. У книзі «Вимірювання та аналітика спортивної продуктивності» аналітика визначається як моделі, засновані на даних та конкретних дослідницьких питаннях, оскільки це визначення того, що таке хороша продуктивність для окремих осіб та видів спорту. НФЛ є гарним прикладом, де різні позиції в команді активують різні види груп м'язів. Різні типи змінних – це фізичні, психологічні та поведінкові, і все вимірюється для визначення потенціалу покращення продуктивності окремих осіб.

Революція великих даних також вплинула на сферу спортивної аналітики. Вже згадані приклади з НХЛ та НБА залежать від того, що обсяг даних настільки великий, що надає знанням емпіричну основу. Багато великих компаній почали бачити переваги поєднання спортивної аналітики та великих даних для отримання прибутку. На щорічній конференції MIT Sloan: Sports Analytics Conference [34] як IBM, так і Intel виявили свій інтерес, де перша зосередилася на тенісі, а друга - на великих даних у футболі [35]. На SSAC16 відбулася панельна дискусія про те, як великі дані змінили наше сприйняття спорту, і що зростання великих даних - це те, чого вболівальники просто хочуть все більше і більше.

Моделювання даних

Під час створення веб-застосунку на базі даних, де дані потрібно легко вставляти, оновлювати та витягувати, може бути корисним певний тип структури відповідно до того, як виглядатимуть елементи бази даних. Модель «сутність-зв'язок» (ER) - це модель, метою якої є демонстрація загального огляду системи баз даних з висококонцептуальною моделлю як сутностей, так і зв'язків між ними [36].

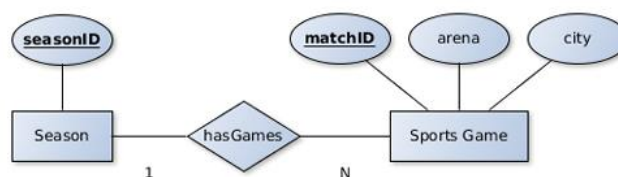


Рисунок 1.6 - Приклад ER-діаграми

Це концептуальне моделювання починається з фази визначення, де потрібно визначити всі сутності, які повинні існувати в базі даних. Сутність може бути подібною до речі або об'єкта, який представляє щось, що ми хочемо зберігати, наприклад, гравця або спортивну гру. До кожної сутності додаються так звані атрибути, які можна розглядати як властивості сутності. Вони додаються, щоб встановити, які елементи має сутність, і щоб представити, чим вона є. Деякі ключові атрибути будуть присутні для кожної сутності та функціонуватимуть як ідентифікація унікального початку сутності, і вони підкреслені на ER-діаграмі. Після визначення сутностей необхідно встановити зв'язки між ними, які представлені у вигляді похилих квадратів, з'єднаних лініями між сутностями. Дуже важливо визначити, скільки зв'язків може мати певна унікальна екземплярація сутності з іншою сутністю, наприклад, гра може мати лише один сезон, тоді як сезон має багато ігор (див. рис. 1.6) . Коли все визначено, можна розпочати реалізацію бази даних.

Модель розширеної сутності-зв'язку (EER) також включає деякі додаткові корисні функції. Підкласи є прикладами розширених можливостей. Пізніше об'єктно-орієнтоване програмування реалізувало подібні функції, де об'єкт успадковує властивості суперкласу. Це могло б, наприклад, зробити обидві сутності, флорбол та хокей, успадкованими від суперкласу спорту [36].

Іншими інструментами, які можна використовувати для моделювання концептуальної моделі даних, є так звана UML (Уніфікована мова моделювання). UML традиційно є моделлю для об'єктно-орієнтованих структурних мов, таких як Java або C++, для визначення класів або об'єктів та того, як вони взаємодіють один з одним у системі. Деякі сучасні технології баз даних мають об'єктно-орієнтовані структури, наприклад, OODBS [37], або структури, що підлягають зв'язку, такі як бази даних на основі документів, наприклад, MongoDB, які також можуть використовувати UML для моделювання баз даних. Версія під назвою UML/P моделює різні класи, інтерфейси та інші елементи, що існують у системі, відповідно до встановленої формули (див. рис. 1.7) .

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

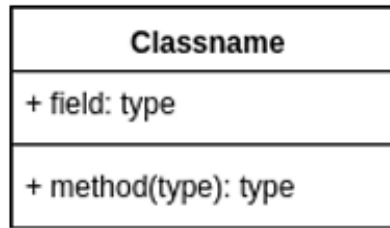


Рисунок 1.7 - Приклад класу згідно з UML

UML може трактувати різні підходи та функції моделювання радше як оголошення методів та змінних, що містяться в класі. Варто зазначити, що існує багато варіантів створення моделей UML, але це досить традиційний та поширений варіант.

Статистична модель

Під час перетворення даних на статистику певні моделі повинні маніпулювати та обчислювати дані, щоб перетворити їх на бажані відповіді. Існують як статистичні, так і ймовірнісні моделі, коли є дані. Ймовірнісні моделі прогнозують, що може статися, тоді як статистика просто впорядковує те, що вже сталося. Статистика створюється шляхом впорядкування даних, узагальнення та подальшого аналізу [39].

Щодо спортивного контексту, існує як базова, так і розширена статистика, яка може бути корисною. Прості суми зіграних ігор, набраних очок та забитих голів часто показують, як команда виступає протягом сезону. Існує два типи результатів, які є важливими для підсумовування спортивних даних, а саме середні значення та медіани. Середні значення обчислюють середнє значення шляхом підсумовування всіх значень, ділячи його на кількість записів. Це дає середнє значення (за певну кількість ігор), яке легко порівнювати з товаришами по команді та гравцями суперників. Медіана - це середнє значення в стеку значень, яке показує, яке значення знаходиться посередині. Це показує половину значень нижче, а іншу половину вище медіани. Іноді ці прості значення більше говорять про ситуацію, ніж про частоту події (див. рівняння 1.1 та 1.2).

$$15X_1 + X_2 + X_3 + X_4 + X_5 \quad (1.1)$$

Рівняння 1.1 Приклад обчислення середнього значення.

$$X_1 = 1, X_2 = 3, X_3 = 5, X_4 = 6, X_5 = 9 \quad (1.2)$$

Рівняння 1.2 Приклад медіани. У цьому прикладі медіана дорівнює 5.

Дані також можна розділити на певні інтервали, які становлять інтерес для спортивної статистики. Прикладом може бути середній час, проведений на льоду хокеїстом протягом сезону, де середній час поділяється на інтервали, наприклад, по п'ять хвилин кожен. Це може дати певне уявлення про те, скільки гравців у команді повинні грати від 20 до 25 хвилин за гру, надаючи інформацію про те, що квота для гравців, які грають двадцять хвилин, заповнена або навпаки, що може призвести до альтернативних тактик драфту (див. рис. 1.8). Іншим прикладом може бути розгляд відсотка кидків у вигляді гістограм у НБА, де різні типи нападників отримують різні відсотки. Ці приклади є висновками, які можна підтвердити спортивною статистикою.

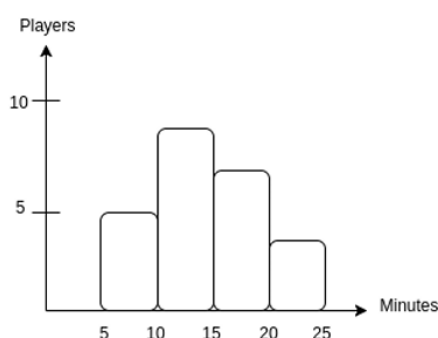


Рисунок 1.8 - Приклад інтервалів, що відображають час перебування хокеїстів на льоду під час матчу

Більш складна статистика, така як визначення кількості ігор, що закінчуються перемогою команди, яка першою забила гол, або яка позиція на полі найкраща для певного гравця, є складнішою та вимагає складніших статистичних моделей. Дисперсії корисні, оскільки результати та досягнення в спорті

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						29
Змн.	Арк.	№ докум.	Підпис	Дата		

різнятися, навіть якщо статистичні моделі встановлювалися для попередніх рекордів. Так зване *стандартне відхилення* описує, наскільки значення відрізняються від середнього значення в наборі значень [2]. Це може бути корисним, оскільки середня кількість забитих голів у грі є певним значенням, але стандартне відхилення показує інтервал, в якому команда забиває голи від середнього значення [30]. Це дозволяє робити більш складні висновки щодо порівняння різних аспектів гри всередині або проти інших команд. Це, разом з іншими більш складними статистичними висновками, може бути корисним, але не входить до предмета цієї роботи.

Візуалізація даних

Під час відображення спортивних даних у графічному та візуально зрозумілому вигляді необхідно використовувати встановлені графіки та діаграми, щоб полегшити збір інформації глядачами. Сьогодні потрібна не лише таблиця підрахунку очок. Оновлення даних у режимі реального часу все частіше відбуваються у найбільших спортивних лігах [40]. Медіаплатформи можуть варіюватися від мобільних пристроїв до телебачення, але існують певні усталені тактики щодо того, як глядачі розуміють спортивні дані незалежно від контексту.

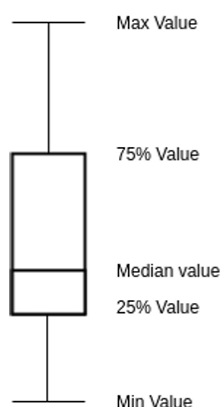


Рисунок 1.9 - Приклад коробчастої діаграми

Гістограми та коробкові діаграми – це два підходи до правильного відображення спортивних даних. Гістограми описують інтервали в наборі даних, створюючи графіки для кількості записів певного типу між двома значеннями.

Ці гістограми представляють наближене значення даних між значеннями, що допомагає зрозуміти набори даних (див., наприклад, рис. 1.8) . Це було успішно розроблено у світі баз даних.

Коробчасті діаграми – це спосіб узагальнення даних, використовуючи медіану для надання інформації про дані. Середня частина коробки – це медіана, а кінці коробки – це значення 25% та 75% набору даних, що становить 50% значень всередині коробки. Дві лінії в кінці – це мінімальне та максимальне значення. Ці компоненти утворюють коробкову діаграму, яка допомагає отримати знання з даних [31].

1.3 Сучасні технології веб-розробки для систем спортивної аналітики

Для розробки веб-застосунків, які повинні збирати та обчислювати дані відповідно до спортивної аналітики, бажано мати гарну архітектуру та дизайн, проте сучасний та ефективний підхід. Дизайн будь-якого застосунку, що збирає дані, не повинен бути надто складним, але все ж надавати детальні дані у зручному для користувача вигляді, оскільки концепція інтерактивного дизайну полягає у створенні функціональних застосунків, орієнтованих на користувача. Застосунок, який належним чином витягує та відображає спортивні дані, повинен містити детальну інформацію, оскільки існує різноманітність ситуацій, у яких дані будуть представлені. Від відображення спортивних даних у ЗМІ, що, можливо, задовольняє шанувальників спорту, перш за все, неофіційною статистикою, до більшої кількості бізнес-даних для організацій [40].

Архітектура та дизайн

Існує кілька шаблонів, рекомендованих для використання під час створення структури застосунку. Їх слід розглядати як парадигми, розроблені з часом програмістами та архітекторами програмного забезпечення. Деякі архітектури зосереджені на легкості з великою гнучкістю, тоді як інші архітектурні проекти зосереджені на забезпеченні найкращої можливої продуктивності. Цей розділ, що стосується різних архітектурних підходів, взято переважно з книги «*Архітек*

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

турні шаблони програмного забезпечення», яка описує деякі сучасні архітектурні шаблони. Варто зазначити, що деякі з цих архітектурних шаблонів можна комбінувати або використовувати на різних рівнях застосунку.

Багатошарова архітектура

Одна з архітектур, яка має рівневу структуру, – це багатошарова архітектура. Вона поділяє різні шари архітектури на довільну кількість шарів, що мають різну функціональність. Однією з найпоширеніших парадигм є структурування її відповідно до чотирьох шарів з назвами: *рівень представлення*, *бізнес-рівень*, *рівень збереження* та *рівень бази даних* (див. рисунок 1.10).

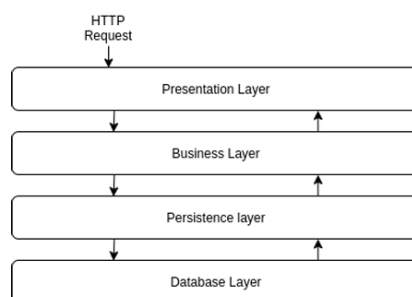


Рисунок 1.10 - Багаторівнева архітектура

Рівень презентації – це рівень, де користувач бачить та взаємодіє з програмою, і його основна функція – відображення даних. Бізнес-рівень взаємодіє з рівнем персистенції, який безпосередньо взаємодіє з базою даних. Останній рівень бази даних має на меті просто зберігати дані. Таким чином, протягом життя HTTP-запит проходитиме через різні рівні далі до рівня даних, а потім повертатиметься до рівня презентації. У контексті веб-застосунку три рівні є поширеним способом структурування, де фронтенд є рівнем презентації, а середній рівень – фреймворками, що спрощують взаємодію з бекенд-рівнем (з моделями баз даних).

Архітектура, керована подіями

Архітектура, керована подіями, використовує підхід обробки запитів залежно від того, які події мають активувати. Існує два основних типи подій: одні відбуваються організовано з централізованим організатором, а інші являють

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						32
Змн.	Арк.	№ докум.	Підпис	Дата		

собою примітивний ланцюговий цикл подій без координації. Перший тип створює подію, яку система ставить у чергу до певного організатора. Організатор делегує події певним відповідним каналам подій (див. рис. 1.11). Ці канали активують необхідні процеси для виконання завдання або реакції на те, що запустила подія.

За допомогою такого підходу можна досягти високої продуктивності системи, надаючи можливість одночасно виконувати багато процесів.

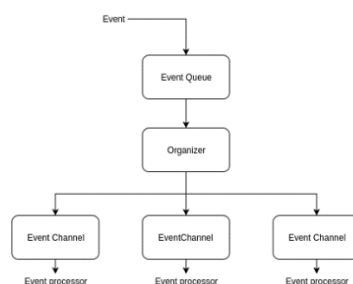
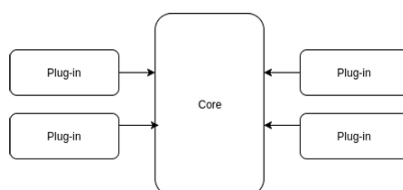


Рисунок 1.11 - Архітектура, керована подіями, з центральним організатором

Друга альтернатива — це радше лінійний підхід, де події безпосередньо передаються до каналу подій, який активує процеси, необхідні для конкретного завдання. Це можна певною мірою скоординувати, коли багато процесів можна активувати одночасно відповідно до каналу.

Архітектура мікроядра (шаблон архітектури плагіна)

Мікроядерна архітектура відокремлює основну функціональність від додаткової у вигляді доданих плагінів (див. рис.1.12). Можна зазначити деяку подібність до операційної системи, де є базова функціональність для взаємодії з комп'ютером та додаткові програми для виконання певних завдань.



Шаблон архітектури мікросервісів

Підхід, що передбачає окрему побудову сервісів без необхідності створювати цілу систему, є основою архітектури мікросервісів (див. рисунок 1.13) Спочатку необхідно сформулювати певне визначення того, що сервіс має намір обробляти. Потім робота розподіляється на певні команди для створення або оновлення незалежних сервісів, які обробляють певну функціональність, та їхнього об'єднання в кінцевому підсумку [32]. Це створює певну гнучкість у розробці, де певні компоненти можуть розроблятися та тестуватися незалежно, тоді як продуктивність може знижуватися, якщо деякі компоненти сервісу повільні у своїй функціональності. Це забезпечує легкий підхід до програмування та дозволяє використовувати деякі функції, які можуть працювати самостійно та використовуватися в інших контекстах.

Запити надходять до певного рівня користувацького інтерфейсу, який обробляє запити та надсилає їх до потрібного сервісу. Один запит може активувати багато сервісів, які координуються на рівні користувацького інтерфейсу, де запити REST (передача репрезентативного стану) є поширеними у веб-розробці. Якщо сервіси є суто внутрішніми для системи, можна розробити певний API (інтерфейс

прикладного програмування), який функціонуватиме як внутрішній рівень користувацького інтерфейсу. Це дозволяє отримувати належну відповідь на запити від зовнішніх користувачів, не знаючи, як вони були оброблені внутрішньо. За допомогою API можна активувати багато сервісів, але

вони повертатимуть належну відповідь користувачеві, який зробив запит [33].

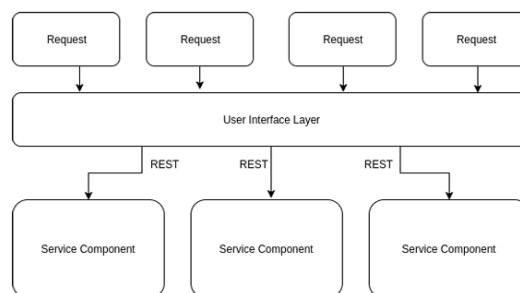


Рисунок 1.13 - Архітектура мікросервісів

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

Космічна архітектура

Просторова архітектура відрізняється від інших підходів. Вона розділяє різні процеси та моделі баз даних для застосунку та обробляє їх за допомогою компонента проміжного програмного забезпечення (див. рис. 1.14). Так звані процесорні блоки містять незалежні функції, які синхронізуються за допомогою проміжного програмного забезпечення [34]. Сучасними прикладами цього типу архітектури є численні хмарні рішення для застосунків, що існують на ринку, де доступ та зміни можуть здійснюватися з різних пристроїв.

Існує розділення блоків обробки, які обробляють різні запити та пізніше синхронізуються з проміжним програмним забезпеченням, яке організовує всі екземпляри. Проміжне програмне забезпечення функціонує як комунікаційний центр, тоді як робота обробляється в багатьох різних місцях. Продуктивність можна покращити шляхом розділення рівня бази даних на багато різних місць, оскільки рівень бази даних часто є вузьким місцем високої продуктивності. Усі запити хочуть отримувати, додавати або змінювати контент з одного місця, але за допомогою просторової архітектури це можна вирішити розумнішим способом. Різні частини проміжного програмного забезпечення можна розділити на *Messaging Grid*, *Data Grid*, *Processing Grid* та *Deployment Manager*.

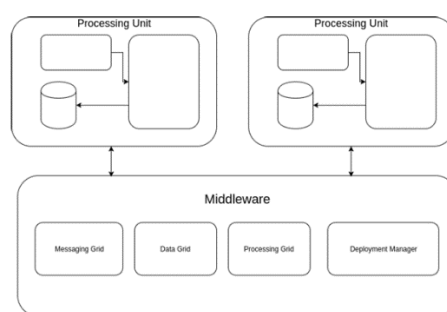


Рисунок 1.14 - Космічна архітектура

Сітка обміну повідомленнями обробляє всі делегування щодо запиту, активує доступні процесорні блоки та активує їх для виконання завдання. Сітка даних обробляє всі функції, пов'язані з базою даних, де реплікація та синхронізація можуть вважатися основними функціональними можливостями. Це дозволяє

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						35
Змн.	Арк.	№ докум.	Підпис	Дата		

рішенням на основі простору інтегрувати процесорні блоки, включені в процес. Сітка обробки натомість є координатором, якщо багато процесорних блоків збираються працювати над одним і тим самим завданням, тоді виконується багаторазова обробка. Розгортання обробляється завданням менеджерів розгортання, які активують та вимикають процеси блоків відповідно до потреби [35].

Стеки програмного забезпечення

У веб-розробці існує безліч комбінацій методів, що стосуються мов програмування, фронтенд-фреймворків та бекенд-фреймворків. Так звані *програмні стеки* – це поширені комбінації методів, які об'єднані разом та добре взаємодіють. Часто згадуються методи операційної системи, веб-сервера та баз даних, але також можуть зустрічатися деякі бажані мови програмування та фреймворки. Деякі із запропонованих стеків можна змінювати за допомогою різних мов програмування або методів баз даних та забезпечувати інші набори методів.

Різні стеки то набувають, то зникають популярності серед розробників. Деякі зі списку в цій роботі – це стеки, які використовувалися в Sports Editing, тоді як інші стеки новіші та сучасніші, ніж ті, що були на момент початку діяльності компанії.

Стек LAMP розшифровується як операційна система Linux, Apache – веб-сервер, MySQL – технологія баз даних та PHP – основна мова для розробки серверного бекенду. Це був популярний вибір серед розробників протягом 2000-х років і утвердився серед розробників. Для веб-розробки PHP є надійним вибором для програмування бекенду, оскільки він певним чином взаємодіє з MySQL. MySQL – це реляційно-орієнтована база даних, де дані зберігаються у вигляді таблиць у базі даних. Дані збираються у вигляді таблиць, з яких певні значення можна витягти за допомогою різних запитів. Реляції пов'язують таблиці разом, створюючи базу даних, повну бажаних залежностей. Доступні й інші функції, такі як реплікація даних.

LEMP – це варіант LAMP, де багато компонентів однакові, але веб-сервер Apache замінено на Nginx. У деяких дослідженнях Nginx вважається швидкою технологією. LEMP має багато спільного з LAMP, і деякі вважають, що це радше

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						36
Змн.	Арк.	№ докум.	Підпис	Дата		

питання смаку щодо техніки веб-сервера. Спортивне редагування використовує певну модифікацію LEMP з деякими додатковими мікросервісами, доданими та розгорнутими WebPack, модулем-пакетом. Webpack не обов'язково є основною частиною стеку рішень LEMP, але застосовний для стеку.

Стек MEAN означає новіший підхід порівняно з LAMP. MEAN розшифровується як MongoDB як технологія баз даних, Express.js як фреймворк веб-застосунків, Angular як фронтенд-фреймворк та Node.js для створення застосунків. Node.js та веб-фреймворк Express.js дійсно тісно пов'язані між собою. MEAN став популярним фреймворком, замінивши багато стеків LAMP у галузі. Головною перевагою використання MEAN є те, що всі компоненти стеку використовують JavaScript як мову програмування, що дозволяє розробникам зосередитися на вивченні однієї мови. Він побудований на звичайній логіці MVC (Model-View-Controller). MEAN також забезпечує підтримку Angular.js, популярного вибору багатьох фронтенд-розробників, які створюють веб-сайти з понад 55 000 зірок на Github. Фреймворк більш детально описаний у розділі «Фронтенд-фреймворки».

Стек MERN - це версія популярного стеку MEAN, де фронтенд-фреймворк Angular замінено фронтенд-фреймворком з відкритим кодом *React*. Як Angular, так і React - це два добре зарекомендували себе фреймворки (на основі JavaScript), які конкурують за увагу веб-розробників. React розроблено Facebook і буде пояснено в розділі про фронтенд-фреймворки.

Розробка бекенду

Бекенд – це базова платформа, з якою взаємодіє додаток, де відбувається обробка запитів та даних. Оскільки дані мають спортивний характер, а деякі функції бекенду, наприклад, для обчислення статистики, деякі поширені фреймворки бекенду потребують оцінки.

Буде представлено три фреймворки, всі з яких використовують різні мови програмування та мають різні масштаби. Express (з Node.js) – це популярний бекенд на основі JavaScript, який працює асинхронно. Flask, з іншого боку, – це легкий варіант для активації Python. Нарешті, Laravel – це PHP-фреймворк, який постачається з багатьма готовими налаштуваннями та функціями.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						37
Змн.	Арк.	№ докум.	Підпис	Дата		

Node.js - це веб-серверна бібліотека для JavaScript, яка дозволяє виконувати асинхронні запити, що дозволяє розробникам виконувати завдання паралельно та є скоріше подієво-орієнтованою мовою. Часто звичайний підхід, коли так звані потоки виконують потік коду в хронічному порядку, був скасований на користь асинхронного програмування. Це дозволяє виконувати багато завдань одночасно та уникати проблем із блокуванням, які може спричинити потокова робота.

Node.js використовує стиль виконання певного коду, керований подіями, якщо спрацює одна конкретна подія (див. рис. 1.15). Це дозволяє виконувати багато подій одночасно, а цикл завершується, коли подій більше не залишається. Звичайний JavaScript виконується таким самим чином.

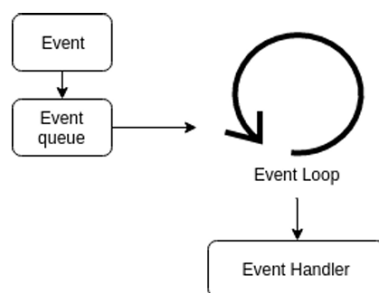


Рисунок 1.15 - Обробник подій згідно з Node.js.

Express - це поширений бекенд-фреймворк для використання техніки Node.js, який надає шаблон для структури веб-сервера. Він генерує код і структуру папок для мінімалістичного веб-застосунку, придатного для початку розробки [36]. Це також дозволяє використовувати деякі функції, наприклад, розширення об'єктів відповідей для відповідності різним критеріям виводу або повнофункціональну систему маршрутизації, де ви, наприклад, керуєте різними URL-адресами для сервера. Це все ще мінімалістичний підхід, де ви додаєте необхідні елементи для певних контекстів.

Flask - це легкий фреймворк на Python для забезпечення функціональності веб-сервера. Оскільки він легкий, вам доведеться створювати всі функції з нуля, і жодна функція не є безкоштовною у Flask, і її можна зберігати в одному

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						38
Змн.	Арк.	№ докум.	Підпис	Дата		

конкретному .ру-файлі. Це забезпечує контроль над обробкою даних, але може призвести до збільшення робочого часу на розробку чогось корисного. Продуктивність, паралельність та гнучкість - це властивості, які вважаються хорошими у фреймворку Flask.

Laravel - це детальний та (в багатьох аспектах) повний бекенд-фреймворк, написаний на PHP. Laravel побудовано на базі PHP- фреймворку *Symfony* , який забезпечує веб-функціональність. Філософія Laravel полягає в тому, щоб надавати інструмент з великою кількістю функцій « з коробки». Остання стабільна версія (Laravel 5.4) постачається з інтегрованою підтримкою фронтенд-фреймворку Vue.js та Bootstrap, а також інших відповідних інструментів, таких як Sass та Webpack (див. «Фронтенд-фреймворки» далі в роботі). Розробку можна розпочати одразу після встановлення, а за допомогою спеціальних команд Laravel це полегшує розробку, наприклад, створення моделей або контролерів за допомогою простої команди. Службу автентифікації можна ввімкнути, з екраном входу та пом'якшенням баз даних, просто за допомогою ще однієї корисної команди. Така інфраструктура економить час розробнику, який витрачає час на створення потрібного застосунку.

Оскільки PHP є усталеним способом налаштування веб-проектів, а Laravel має відомий Model-View-Controller як зразок у своїй структурі, для веб-розробника не так багато сюрпризів. Налаштування маршрутизації, де ви визначаєте маршрути та функції для URL-адрес, легко встановлюються, що дозволяє потрібному проміжному програмному забезпеченню отримувати до них доступ, наприклад, автентифікацію для кенів. Також включені інші функції PHP, такі як інтервали між іменами та власний конструктор запитів Laravel, що спрощує взаємодію з базою даних. Laravel має можливість вибирати, який метод бази даних слід активувати, оскільки MySQL , SQLite та інші методи бази даних включені під час встановлення. Пізніші версії також постачаються з Vue.js як основним інструментом фронтенду, що входить до комплекту встановлення, де компоненти Vue містять код, який може отримувати доступ до даних з бази даних та оновлювати їх [37]. Такий підхід, що дозволяє використовувати багато опцій та

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						39
Змн.	Арк.	№ докум.	Підпис	Дата		

включає підтримку третіх сторін, підходить для розробки прототипів.

Технології баз даних

Під час обробки статистики нам дійсно потрібна належна технологія баз даних, щоб все обробляти правильно. Існує безліч підходів до зберігання даних у базах даних, де SQL став провідною платформою для реляційних баз даних [36]. SQL існує в багатьох версіях, але в цій роботі основна увага буде зосереджена на варіантах з відкритим кодом для згаданої методики, включаючи MySQL та SQLite. Антиполюс до встановленого SQL є NoSQL, який забезпечує більше свободи та менше правил, яких потрібно дотримуватися, наприклад, немає посилань на інші ключові значення. MongoDB розглядається в цій роботі через його популярність та те, що спортивне редагування використовує цю версію NoSQL.

MySQL (Моя структурована мова запитів) – це так звана реляційна база даних, дані якої зберігаються в таблицях. Різні стовпці представляють різні атрибути, а один рядок у таблиці представляє запис із даними. Так звані RDBMS (Реляційні системи керування базами даних) включають концепцію, яка дозволяє різним таблицям з'єднуватися одна з одною, встановлюючи між ними зв'язки. Це забезпечує парадигму в обробці баз даних, де дані зберігаються у вигляді таблиць і доступ до них здійснюється за допомогою так званих *запитів*. Запит подібний до питання до бази даних, де ви отримуєте результуючу таблицю залежно від того, що запитується. Прикладом може бути те, що таблиця для автора може мати зв'язки з таблицями, які представляють книги, якщо книги позначені authorID у таблиці для книг (див. рис. 1.16 та 1.17) .

authorID	name	birthyear
1	Robert Louis Stevenson	1850
2	Joseph Conrad	1857
3	Ryduard Kipling	1865

Рисунок 1.16 - Приклад авторів, що зберігаються як таблиці в MySQL.

Первинний ключ – це концепція, де один з атрибутів, призначених для таблиці, встановлюється як ідентифікатор для певного запису. У прикладі на рисунку 1.16_первинним ключем авторів є *authorID*, який визначає саме цього конкретного автора [38]. Концепція зовнішнього ключа полягає в тому, як з'єднувати таблиці та взаємодіяти одна з одною, де можна посилатися на інший запис в іншій таблиці, де вказано їхній первинний ключ. У прикладі на рисунку 1.17_ми вказуємо *authorID* як посилання (зовнішній ключ) на таблицю авторів, пов'язуючи книги з правильними авторами. Запити можна використовувати для запиту бази даних, наприклад, усіх книг певного автора за його *authorID*.

bookID	authorID	title	genre
1	1	Dr Jekyll & Mr Hyde	Horror
2	1	Treasure Island	Adventure
3	2	Heart of Darkness	Novel
4	3	The Jungle Book	Adventure

Рисунок 1.17 - Приклад книг, що зберігаються як таблиці в MySQL.

Такий спосіб структурування даних дозволяє використовувати інші функції MySQL, такі як JOINS, де ви можете запитувати у бази даних відповіді залежно від кількох умов, або більш примітивні функції, такі як COUNT, де ви просто отримуєте результат певної кількості записів для цього конкретного запиту. Це, разом з іншими функціями, робить MySQL доступною структурою бази даних, яка дозволяє витягувати потрібні дані, якщо дотримуються правил реляції.

SQLite3 - це справді спрощена версія SQL, яка обробляє дані аналогічним чином. Усе зберігається в одному файлі, що дозволяє зменшити обсяг пам'яті, але спростити налаштування. Це не рекомендується для масштабування, оскільки кожен користувач взаємодітиме з файлом бази даних одночасно, навіть якщо це стане швидшим, якщо надати йому більше місця на диску. Усі функції звичайного SQL присутні та можуть бути реалізовані за допомогою такого компактного файлу, що робить його зручним для створення прототипів та невеликих проєктів.

Бази даних NoSQL не передбачають табличної структури реляційних баз даних. Існує велика кількість NoSQL-технологій, де MongoDB є поширеною версією. MongoDB має інший підхід порівняно з добре відомими реляційними базами даних. MongoDB - це скоріше колекція, ніж таблиці. Колекції можуть відрізнятися за структурою і не потребують однакової формули, що дозволяє використовувати динамічний підхід до зберігання даних. Фактичні дані - це скоріше документ, ніж таблиця. Цей підхід може вимагати більше динамічного виділення простору, оскільки документи можна створювати з більшою свободою відповідно до реляційних баз даних, де необхідно дотримуватися правил відповідно до зовнішніх ключів та відношень [39].

Документи в MongoDB містять необхідні дані для цього конкретного контексту, і все зберігається в так званих об'єктах JSON (JavaScript Object Notation). JSON зберігає дані у структурі ключ-значення, де рядок спочатку зберігається як назва змінної, а наступний рядок - як значення, наприклад, `{"variablename" : "value"}`. Об'єкти JSON можуть зберігатися всередині інших об'єктів, що є основою документа. Масиви з об'єктами JSON також можуть зберігатися. Це може бути корисним, наприклад, для зберігання багатьох хокеїстів у хокейній команді (див. рис. 118).

```
_id: ObjectId('59108565d4b9737fa45c0acc')
teamID: "0987678922231"
name: "HV71"
city: "Jönköping"
country: "Sweden"
foundedAtYear: "1971"
sportSeasonID: "78293045832479059837"
arenaID: "324324432453245"
  staff: Object
  players: Array
    0: Object
      _id: "59105812d4b9737fa45c0ab2"
      name: "Simon Önerud"
      position: "Forward"
      shirtnumber: "11"
    1: Object
      _id: "59106dbad4b9737fa45c0ab3"
      name: "Martin Thörnberg"
      position: "Forward"
      shirtnumber: "10"
    2: Object
      _id: "5910771fd4b9737fa45c0abd"
      name: "Mattias Tedenby"
      position: "Forward"
      shirtnumber: "21"
```

Рисунок 1.18 - Приклад хокейної команди, що зберігається в MongoDB.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						42
Змн.	Арк.	№ докум.	Підпис	Дата		

Варто зазначити, що такі об'єкти, як "хокеїсти", можуть містити більше інформації, тоді як команда може зберігати вибрані значення, такі як ім'я, посада та номер футболки, заповнюючи об'єкт команди лише корисною інформацією для цього конкретного контексту.

Розробка фронтенду

Веб-розробка постійно розвивається, і постійно випробовуються нові підходи до вирішення різних проблем у ланцюжку розробки [40]. Кількість фронтенд-фреймворків JavaScript різко зросла за останні роки, оскільки JavaScript є найпопулярнішою мовою для браузера. Необхідність оцінки діапазону всіх фреймворків є важливою для забезпечення обґрунтованого вибору відповідно до доступних досліджень, і те саме стосується бекенд-розробки.

Фронтенд означає те, що бачить користувач і з чим взаємодіє, і здебільшого встановлюється відповідно до так званої парадигми Модель-Вид-Контролер (MVC). Розробка спрощується завдяки розділенню основних даних, що зберігаються в моделях, від того, що ви бачите та з чим взаємодієте, тобто представлень. Контролер - це посередник, який передає вимоги від представлення, оновлюючи або створюючи моделі, що відповідають представленню та тому, що бачить користувач. Цей підхід є звичайною парадигмою у розробці фронтенду, і на даний момент це стандартний спосіб створення інструментів для фронтенду.

Усі фреймворки фронтенду, представлені в роботі, не лише побудовані за допомогою MVC, але й дозволяють реалізувати так зване *реактивне програмування*. Це дозволяє безпосередньо вводити інформацію від користувача в додаток у режимі реального часу. Різні фреймворки по-різному обробляють реактивні вимоги, але всі три фреймворки мають особливість безпосередньо реагувати на взаємодії.

Останніми роками використовують Angular.js як фреймворк. Angular розроблений Google і є фреймворком JavaScript для розробки фронтенду, побудованим на класичному підході Model-View-Controller, де представлення даних та взаємодія розділені. Він створений для забезпечення більш динамічного та інтуїтивного підходу до створення веб-додатків (головним чином, односторінкових

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						43
Змн.	Арк.	№ докум.	Підпис	Дата		

додатків). Варто зазначити, що Angular 2 використовує так званий TypeScript замість JavaScript, який транскompілюється в останній. Це мова, розроблена Microsoft, яка компілює код TypeScript у JavaScript, роблячи його доступним для мультиплатформенного використання.

Angular забезпечує двостороннє зв'язування даних, що означає, що ви можете оновлювати та відображати дані легшим способом. Деякі інші функції, доступні в Angular, полягають у можливості пошуку певних об'єктів DOM (модель об'єкта документа) безпосередньо в коді. DOM означає різні елементи HTML-коду, такі як body-, p- або ul-елементи. Вони структуровані у деревоподібну структуру, що дозволяє Angular зчитувати або оновлювати їх різними способами, наприклад, перебираючи всі записи списку безпосередньо в коді. Angular також має спосіб вмикання фіктивних даних у застосунок, які є свого роду тестовим об'єктом, що вводиться для перевірки поведінки системи.

React - це бібліотека JavaScript, яка використовується для розробки фронтенду. Вона була випущена Facebook та Instagram у 2013 році, щоб кинути виклик парадигмам у звичайних формулах HTML, JavaScript та CSS. Щоб вирішити цю проблему, динамічні набори даних з постійним потоком нових оновлень та зображень, а також відображення цих змін, спричинили деякі проблеми. React відповів на цю проблему, створивши бібліотеку, яка вводить стани (States). Коли дані змінюються або оновлюються, стан замінюється новим, що забезпечує безперебійний досвід відображення оновлених даних перед користувачем. Це дозволяє змінювати дані безпосередньо, не спричиняючи проблем з оновленнями, які виникали б у звичайному фронтенд-стеку. Це більш ефективний підхід до двостороннього зв'язування даних, і React відомий своєю масштабованістю та швидкістю.

Він має підхід Model-View-Controller та має можливість обробляти велику масштабованість, оскільки обидва згадані додатки мають велику кількість зареєстрованих користувачів та багатьох щоденних користувачів. Так званий React Native - це оновлена версія з підходом створення коду, який працює всюди, що дає можливість одночасно створювати мобільні додатки для вебу, Android та iOS

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						44
Змн.	Арк.	№ докум.	Підпис	Дата		

Vue - це сучасний фреймворк JavaScript та найновіший із трьох, розглянутих у дослідженні літератури. Vue був спочатку розроблений автором *Еваном Ю* у 2014 році, і оскільки жодна велика компанія не володіє Vue, він повністю з відкритим вихідним кодом. Спільнота зросла за останні роки, вона має обов'язкову документацію та має 53 700 зірок на Github.

Vue не передбачає двостороннього зв'язування, але розробив надійне одностороннє зв'язування з хорошою продуктивністю. Деяка критика щодо двостороннього зв'язування полягає в тому, що може бути важко відстежувати потік змін даних, і Vue вирішив це за допомогою іншого підходу до оновлення даних. Це спрощує налагодження, оскільки легше відстежувати зміни в даних. Незважаючи на відсутність двостороннього зв'язування, продуктивність Vue є конкурентоспроможною. Стефан Краузе перераховує в таблиці у своєму блозі продуктивність JavaScript Frameworks порівняно один з одним під час призначення, видалення та оновлення даних до елементів DOM, де Vue показує хороші результати серед усіх фреймворків.

Vue дотримується структури, де ви збираєте дані (у вигляді змінних, масивів тощо) в екземплярі Vue, який можна розділити на компоненти. Наступний приклад показує, як оголошення даних здійснюється безпосередньо в оголошенні та як легко отримати доступ до даних за допомогою двох фігурних дужок відповідно до компонента Vue (див. рисунок 1.19). Це зроблено з набагато меншим обсягом коду, ніж у прикладі React.

```
<div id="app">
  {{ text }}
</div>

var app = new Vue({
  el: '#app',
  data: {
    text: 'This is a testing text'
  }
})
```

Рисунок 1.19 - Приклад коду Vue з виводом «Це тестовий текст».

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		45

Не тільки дані можна легко зберігати, але й методи та інші налаштування можна встановлювати відповідно до структури Vue. Тож Vue – чудовий спосіб почати, оскільки вам потрібно лише імпортувати Vue-скрипт безпосередньо в HTML-код, якщо ви просто хочете ввімкнути Vue. У масштабному застосунку це не є прийнятним варіантом, оскільки ви хочете зберігати якомога більше даних для підвищення продуктивності, але почати та навчитися з таким підходом дуже просто. Ви імпортуєте його так само, як і будь-який інший js-файл або .css-файл, але з онлайн-джерела, щоб отримати лише синтаксис, необхідний для створення.

Компілятори JavaScript – це те, що набуло розвитку у фронтенд-розробці, дозволяючи об'єднувати модулі та покращувати продуктивність фронтенд-фреймворків. *Webpack* – це поширене програмне забезпечення для об'єднання модулів в один. Прикладом може бути те, що у вас є багато .js-файлів під час розробки функціонального та адаптивного фронтенд-інтерфейсу, але вони розподілені та залежать один від одного таким чином, що їх важко структурувати. Webpack перетворює всі залежності на статичні ресурси, до яких програма може отримати доступ набагато легше та з більшою продуктивністю. Таким чином, після того, як Webpack зібрав ресурси з різних модулів, час завантаження програми має зменшитися, а ресурси структуруватися легше. Це може бути дуже корисним, оскільки збірку потрібно виконати лише один раз, а коли програма працює, Webpack вже виконав свою роботу. Це може покращити продуктивність для великих проєктів, створюючи більше статичних ресурсів у звичайному JavaScript.

Класичний CSS – це парадигма проектування веб-сайтів зі стилевим кодом, який HTML імпортує для кращого вигляду. CSS – це стара мова, і мало що було зроблено для її революційного вдосконалення в первісному вигляді, але були розроблені так звані *CSS-препроцесори* для спрощення стилізації. Мови *SASS* та *LESS* увійшли до спільноти фронтенду кілька років тому, забезпечуючи такі функції, як збереження значень як змінних та вкладення стилевим кодом для різних компонентів.

SASS має перевагу в тому, що він є валідним CSS у чистому вигляді, що полегшує розуміння новачками. SASS також може виконувати обчислення та

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						46
Змн.	Арк.	№ докум.	Підпис	Дата		

використовувати IF-оператори, що дозволяє, наприклад, легше змінювати колір при натисканні кнопки. LESS також дозволяє використовувати змінні та функції, а також так звані міксини, де набір властивостей можна імпортувати в інший компонент. Структура схожа на Webpack, який імпортує та зчитує різні типи .css-файлів (або .sass та .scss) та перетворює їх на валідні статичні ресурси.

Проектування прототипів

Під час розробки прототипу певна методологія прототипування може спростити процес. Концепція мінімально життєздатного продукту (MVP) є частиною Lean-процесу. Цей підхід до прототипування включає ідентифікацію потреб клієнтів та визначення того, що саме, але пізніше в процесі необхідно встановити MVP. MVP – це прототип, де найбазовіші функції встановлені для цільових клієнтів, забезпечуючи їм достатню цінність, щоб це дало право на розробку застосунку. Останнім кроком є фактичне тестування MVP на клієнті для оцінки реакції та відгуків, чи варто розробляти концепцію.

Для отримання належного прототипу MVP можна використовувати метод, який називається *паперовим прототипуванням*. Цей вид взаємодії полягає в тому, що ви просто малюєте дизайн застосунку на папері з усіма станами першого варіанту MVP, дозволяючи тестувальникам взаємодіяти з ним та надавати відгуки. Це простий і дешевий спосіб провести перше тестування, якщо концепція варта інвестицій.

1.4 Висновок по розділу

У першому розділі розглянуто теоретичні основи розробки веб-додатків для спортивної аналітики. Проаналізовано принципи роботи зі спортивною статистикою, підходи до її обробки та моделювання, а також сучасні технології веб-розробки, що застосовуються у аналітичних системах. Визначено особливості структурування та інтерпретації спортивних даних. Отримані результати формують теоретичну базу для створення веб-додатків для аналізу спортивної статистики.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						47
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. МЕТОДИ ТА ІНСТРУМЕНТИ РОЗРОБКИ ВЕБ-ДОДАТКУ

2.1 Теоретичне дослідження методів візуалізації та аналізу спортивної статистики

Цей розділ охоплює методологію роботи. Він буде в основному розділений на три різні підрозділи. Перший – це *Теоретичне дослідження*, яке описує, як було проведено відповідну літературу та теоретичну підготовку. Другий підрозділ – це *Впровадження*, яке описує, як було виконано реалізацію системи. Він також включає систематичні методології представлення робочого процесу, а також деякі технічні інструменти для спрощення процесу. Третій і останній підрозділ описує оцінку *розробленого* прототипу статистичної програми. У цьому підрозділі буде розглянуто, наскільки ефективним та функціональним є отриманий дизайн програми для досягнення цільової мети – допомогти користувачам збирати та представляти спортивну статистику.

Теоретичне дослідження проводиться для того, щоб надати наукову основу для роботи. Спочатку необхідний підготовчий етап для організації дослідження. Після цього проводиться вивчення літератури, яке включає читання книг, статей та наукових звітів з цієї теми. Перш ніж розпочати етап впровадження, необхідно визначити деякі визначення того, що має бути розроблено, і в цьому випадку необхідно визначити види спорту, які будуть включені до прототипу. Архітектуру застосунку також необхідно визначити перед початком етапу впровадження.

Фаза впровадження – це етап, на якому буде використовуватися кодування та інші систематичні інструменти. Це було проведено відповідно до використання гнучких методів розробки. Зміни, внесені до програмного забезпечення з часом, оброблялися за допомогою певного типу контролю версій. Після завершення розробки розпочався етап оцінювання. Потім прототип був протестований статистичними журналістами під час звіту про матч, де оцінювався дизайн.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						48
Змн.	Арк.	№ докум.	Підпис	Дата		

Підготовка проекту полягає головним чином у визначенні та налаштуванні середовища розробки. Спочатку було здійснено встановлення в компанії, де проводилося дослідження. Компанія Sports Editing Sweden AB була зацікавленою та замовила прототип статистичної програми. Після завершення етапу встановлення було представлено необхідність уточнення того, що входить до проекту. Специфікація вимог, яка перераховувала функціональні та нефункціональні вимоги для кожного завдання, була визначена разом з керівником компанії. Sports Editing вже використовувала передові веб-архітектури, фреймворки та технології для задоволення потреб користувачів у статистичних рішеннях, які вони надають. Також було проведено дослідження веб-методів, що використовуються в компанії, і його додано до порівняння між веб-технологіями.

Літературні дослідження проводяться для вивчення попередніх та поточних досліджень з певної теми, і дуже важливо проводити їх відповідно до належної методики. Алан Брайман стверджує, що «*існуюча література є важливим елементом у будь-якому дослідженні*», і оскільки аналізується багато різноманітної літератури, це була основна частина методу.

Розпочинаючи пошук відповідних джерел, етап планування буде корисним для цього процесу. Також дуже важливо визначити обсяг дослідження та мету, перш ніж проводити пошук відповідної літератури. Це означає, що перед початком пошуку літератури необхідно сформулювати основну мету та концептуалізувати проблему. Щоб визначити обсяг літератури, необхідно визначити ключові слова або рівень деталізації, необхідний для роботи. Коли знайдено статтю, яка здається доречною, буде проведено певну оцінку, наприклад, перевірка кількості цитувань або завантажень може бути важливим аспектом.

Один із методів, що використовується для пошуку відповідної літератури, – це так званий *процес збору інформації*, згідно з книгою Саллі Рамсі «*Як знайти інформацію*». Процес збору інформації поділяється на кроки: *аналіз, визначення, виявлення, оцінка, пошук та доступ*. На етапі аналізу необхідно поставити проблему та визначити, на які дослідницькі питання має відповісти робота. Переходячи до етапу визначення, встановлюються деякі межі та визначається,

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						49
Змн.	Арк.	№ докум.	Підпис	Дата		

яку інформацію потрібно знайти. Коли відомо, яку інформацію потрібно знайти, настає час почати пошук інформації про існуючі дослідження. Щоб виявити різні джерела, ідентифікувати їх та почати визначати, де вони знаходяться, розрізняють такі кроки збору інформації. Щодо деяких джерел може знадобитися певний вид авторизації, а для доступу до певних статей можуть знадобитися додаткові дозволи. Весь процес характеризується оцінкою, і зазначається, що на кожному етапі має бути присутня перспектива релевантності. Рекомендується постійна готовність до проведення оцінювання та оновлень на кожному етапі, що покращить результат збору інформації.

Щодо пошуку онлайн-джерел, встановлено аналогічний процес, де було встановлено важливі ідентифікації понять та ключових слів. Під час визначення пошукових термінів слід проводити оцінку ключових слів за допомогою запропонованих кроків, таких як визначення синонімів, споріднених термінів та альтернативних написань. Модель онлайн-пошуку також перераховує, як обробляти ширші та вужчі терміни, що слід встановити для кожного терміна на етапі планування. Процес онлайн-пошуку – це ітеративний процес, який можна змінювати та повторювати, доки знайдена інформація не задовольнятиме потреби дослідження літератури.

Описано метод, за яким початкова точка веде до більш релевантної літератури. Це метод, який використовується для пошуку релевантної літератури з соціальних досліджень, але може бути релевантним і для технічних термінів. Це більш прямий підхід, що походить від рекомендованої статті з високою релевантністю відповідно до теми (див. рис. 2.1). Це описує методологію, за якою одна релевантна стаття або книга може призвести до іншої. Початок з рекомендованої статті або книги буде оцінюватися відповідно до керівників роботи та пошукової системи зважених алгоритмів для релевантних статей відповідно до пошукового терміна. Коли керівники схвалюють ключову статтю, буде проведено читання літератури, і, відповідно до методу Браймана, будуть вказані деякі примітки та ключові слова. Відповідно до обсягу дослідження будуть згенеровані інші ключові слова, що стосуються пошукового питання. Потім пошук продовжиться за

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						50
Змн.	Арк.	№ докум.	Підпис	Дата		

доступними ресурсами, такими як Бібліотека Університету Лінчепінга, Google Scholar та інші пошукові системи або бібліотеки досліджень. Тепер слід вибрати деякі детально розроблені статті, що мають велику цінність для теми, головним чином шляхом читання анотацій статей або ретельного вивчення. Потім буде прочитано обсяг вибраних статей, і процес повертається до другого етапу, де будуть написані нотатки та зазначені ключові слова. Також слід запланувати певний вид регулярних перевірок новіших публікацій у рамках вивчення літератури.

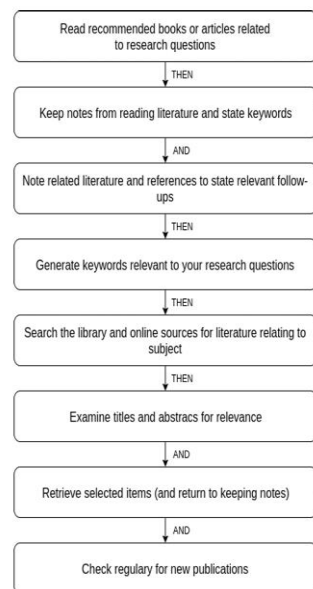


Рисунок 2.1 - Методологія пошуку відповідної літератури.

Було проведено аналіз літератури як з науки про дані, так і з спортивної аналітики, щоб розділити визначення та виявити подібності. Також було досліджено деяку літературу щодо веб-розробки, де основна увага приділялася фронтенд-фреймворкам, бекенд-фреймворкам та повнофункціональним рішенням які є трендовими сьогодні. Було проведено порівняння сучасних фреймворкових рішень, взявши це за основу. Теми, що стосуються спортивної аналітики, розпочалися на великій щорічній конференції MIT Sloan Sports Analytics Conference (SSAC) MIT (Массачусетський технологічний інститут). Щороку вчені з усього світу збираються, щоб обговорити та опублікувати найновіші дослідницькі дані з цієї теми.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						51
Змн.	Арк.	№ докум.	Підпис	Дата		

2.2 Порівняння архітектурних підходів і технологічних стеків веб-розробки

Перед початком впровадження відбувається фаза, на якій визначаються структура та проектування архітектури. Дослідження архітектури програмного забезпечення включає різні підходи до розділення функціональності між різними частинами системи. Існує багато підходів, з певними перевагами та недоліками, для оцінки рекомендованих змінних. Важливість вибору правильної архітектури може вплинути на виконання вимог розробки. Розробили модель для правильного вибору відповідно до проекту архітектури, де деякі аспекти класифікуються.

Атрибути реалізації

- Ремонтопридатність: на якому рівні система може бути працездатною, обробляючи несправності та модифікації.
- Тестованість: Можливість протестувати систему.
- Портативність: Незалежність системи.
- Гнучкість: здатність працювати на різних платформах.
- Повторне використання: рівень використання компонентів в інших контекстах.

- Простота: Як важко це зрозуміти.

Атрибути середовища виконання

- Доступність: рівень охоплення та використання програми.
- Безпека: Рівень безпеки.
- Продуктивність: Як система вимірює час та ресурси для досягнення поставлених цілей.
- Паралелізм: Можливість виконувати багато запитів одночасно.
- Надійність: Зупинка збоїв.
- Масштабованість: Можливість, яку система забезпечує для масштабування, наприклад, кількість користувачів та взаємодій.

Атрибути бізнесу

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						52
Змн.	Арк.	№ докум.	Підпис	Дата		

- Вартість: Скільки коштує система в ресурсах.
- Термін служби: Як довго система служить і залишається сучасною, перш ніж застаріє.

Інші аспекти включають зручність використання та підтримку, що означає, наскільки легко систему обслуговувати та чи легко виправляти помилки. Це також було, серед іншого, включено до методу вибору правильної архітектури для цільового призначення. Також рекомендувалося визначити, які з перелічених аспектів мають найбільше значення для системи, перш ніж оцінювати архітектуру.

Під час вибору веб-технологій, стеків рішень та фреймворків для розробки обговорювалися ті самі пункти оцінки, а висновки щодо цього були обґрунтовані з урахуванням вимог специфікації вимог.

Моделювання даних

Щодо структури зберігання, обчислення та представлення даних, було створено певну концептуальну модель для цих даних. Була побудована високо-концептуальна модель у вигляді ER-моделі (або ER-діаграми) з визначеними сутностями, необхідними для системи спортивної статистики. ER-діаграма представляла функціональні зв'язки даних усієї системи та сутності для кожного з компонентів системи. Це було виконано відповідно до інструкцій книги «*Основи систем баз даних*». Після завершення ER-діаграми вона була впроваджена на етапі впровадження з відповідною технологією баз даних, оціненою для найкращого задоволення потреб статистичної програми.

Існують різні підходи до обробки даних, які можуть бути кращими в контексті спорту. Отже, оскільки статистика потрібна по-різному, необхідно було встановити певне розмежування щодо того, які дані слід експортувати з прототипу. Існувало два основних відправних пункти для розробки статистичної програми, які, можна стверджувати, були важливими для зацікавлених сторін на початку:

- *Статистика спортивних ігор*: Зберігання даних з певного матчу в певному виді спорту з подіями матчу, такими як голи, спроби кидків та пенальті.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						53
Змн.	Арк.	№ докум.	Підпис	Дата		

- *Статистика спортивного сезону:* Огляд багатьох матчів у певному сезоні, де буде розраховано поточну турнірну таблицю та статистику, таку як забиті голи та виграні матчі.

Перший статистичний звіт, який може зацікавити тренерів та самих гравців, стосується конкретного матчу. Це формат введення даних у систему та основа для аналізу даних. Статистик-людина або журналіст вручну вводять дані кожного матчу, використовуючи прототип статистичної програми. Деякі аспекти даних можуть сприяти фактичному тренерському підходу під час гри, тому важливо, щоб дані збиралися та зберігалися під час гри. Тренер може захотіти змінити тактику після перегляду даних першого періоду гри, тому звіт кожного періоду може бути корисним, якщо його впровадити.

Другою відправною точкою для створення статистичної інформації на основі зібраних під час матчів є огляд сезону. Оскільки команда грає в лізі та потребує загального огляду своїх результатів, статистика сезону може бути корисною. На сезонному рівні не вводилися дані, і команда просто збиратиме дані з кожної окремої спортивної гри, щоб мати змогу розрахувати та показати належну статистику для сезону.

2.3 Вибір інструментів та реалізація архітектури веб-додатку

Методологія впровадження – це те, з чого починається фаза розробки. Інструменти для ефективного впровадження використовувалися відповідно до певних методологій розробки. Підходом до розробки була гнучка методологія для планування, огляду роботи та оцінки процесу виконання різних завдань. Амбіція полягає в тому, щоб гнучка розробка забезпечувала контроль під час процесу, оскільки вона відображає завдання з постійною частотою.

Було використано певний вид контролю версій, щоб зміни коду були видимими під час розробки. Це забезпечувало кращий огляд прогресу в розробці, а також залучало керівника, оскільки це міг бути цінний внесок. Іншими аспектами розробки є те, що все було розроблено власними силами компанії та зі

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						54
Змн.	Арк.	№ докум.	Підпис	Дата		

звичайною апаратною підтримкою, використовуючи сервери та офісні приміщення.

Спортивне редагування вже використовує програмний пакет від компанії *Atlassian* для прогресивного виконання розробки. *Atlassian* надає інструменти для різних аспектів планування, кодування та створення. За допомогою програмного забезпечення під назвою *Jira* вони надають програмне забезпечення для надання гнучких інструментів розробки. Внутрішня комунікація здійснюється за допомогою *Atlassian HipChat*, зовнішнє планування здійснюється за допомогою *Atlassian Trello*, а інструментами контролю версій є *Atlassian BitBucket* та *SourceTree*.

Гнучка розробка

Оскільки етап розробки виконуватиметься відповідно до *Agile Development* (гнучких методів розробки), було оцінено версію такого методу для майбутнього проекту. *Agile Development* (гнучка розробка) надає метод для інформування всіх членів команди під час розробки, дозволяючи їм отримувати детальні інструкції щодо того, що вони повинні або можуть робити, а також огляд того, скільки попередніх зважених завдань було виконано.

Існує кілька версій щодо *Agile*-розробки, де *Scrum* можна вважати найпоширенішим. Початкова ідея полягає в тому, щоб декларувати проблему на менші завдання, які потрібно виправити, що призведе до кроку ближче до вирішення проблеми. Завдання зважуються командою за робочими годинами або за сторі-пойнтами, які команда виміряла як певні зусилля. *Scrum* поділяє учасників на *Власника Продукту*, *Scrum Майстра* та *Члена Команди*. Власник Продукту - це той, хто замовляє послуги продукту та інвестує в розробку. *Scrum Майстер* - це людина, яка планує розробку, постійно стежить за прогресом та забезпечує роботу команди на найвищому можливому рівні. Члени команди - це кодери, які повинні вирішувати різні представлені програмні завдання. Спринт - це період часу, протягом якого має бути виконана певна кількість завдань. Так звані Спринт-зустрічі проводяться регулярно (часто два-три тижні), де зазначається, які завдання мають пріоритет і скільки завдань має бути вирішено до наступної

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						55
Змн.	Арк.	№ докум.	Підпис	Дата		

спринт-зустрічі.

Варіантом гнучкої розробки, який багато в чому відповідає Scrum, є метод під назвою *Kanban*. Kanban має інший підхід і є радше постійно суворим способом виконання завдань. Ідея полягає в тому, щоб дозволити лише певній кількості завдань бути активними одночасно, дозволяючи команді зосередитися на меншій кількості завдань, ніж у SCRUM. Чистий Kanban навіть не містить спринт-зустрічей, оскільки він постійно регулюється відповідно до суворої кількості активних завдань. Переваги можна позначити в тому, що це допомагає зосередити увагу розробників і досить легко впроваджується у вже розпочатому бізнесі. Деякими недоліками можна назвати те, що важче отримати уявлення про те, що потрібно зробити.

У спортивному редагуванні використовується Scrum зі спринт-циклами тривалістю два тижні. Шляхом зважування різних завдань балами 1 або більше визначається, скільки з них буде зроблено в спринті. Завдяки пріоритезації беклогу завдань залежно від їхньої релевантності, розробникам легко вибрати відповідне завдання та зробити свій внесок. Короткі зустрічі, які називаються *стендапами*, проводяться щодня у певний час, де розробники встають, обговорюючи, над чим вони працюють і що було зроблено протягом дня. Це допомагає звернутися за допомогою, якщо це необхідно, та звітувати про прогрес.

Оскільки спортивна редакція не є великою компанією, всі члени команди IT-відділу можуть щодня безпосередньо спілкуватися на стендап-сесіях. Керівники компанії постійно отримують інформацію про те, що включає кожен спринт, і відстежують його прогрес за допомогою інструментів програмного забезпечення Atlassian.

Контроль версій

Парадигма веб-розробки полягає в тому, що певний вид контролю версій спрощує етап розробки. Існує безліч сервісів, які дозволяють реалізувати мету контролю версій, тобто контролювати зміни та доданий контент у коді. Сучасні програми контролю версій, такі як GitHub або GitLab, часто надають інструменти для візуалізації того, де і що було змінено, на якому номері рядка коду, таким

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						56
Змн.	Арк.	№ докум.	Підпис	Дата		

чином сприяючи процесу спільної роботи .

GIT є одним з найпоширеніших методів керування версіями, але існують й інші методи. Підхід CVS (Concurrent Version Control System) до керування версіями полягає у відстеженні всіх змін на центральному сервері, де можна знайти подібність до класичної клієнт-серверної архітектури. Subversion є прикладом CVS.

DVCS (Розподілені системи контролю версій) – це радше одноранговий підхід, де користувачі локально додають зміни, які синхронізуються між користувачами через репозиторії. GIT – це популярна система контролю версій, яка є DVCS, а деякі програми – це вже згадані *GitHub*, *GitLab* та *Bitbucket*. Остання використовується компанією Sports Editing, оскільки вона є частиною програмного пакету Atlassian для команд розробників. На етапі розробки цього проекту GitLab використовуватиметься для відстеження прогресу та контролю версій.

Кількісний аналіз даних

Також було проведено впровадження рішення для знання того, як інтерпретувати належний кількісний аналіз даних відповідно до спортивної аналітики та статистичних прескриптивних методологій. Оскільки необхідність належного аналізу даних щодо отримання бажаної статистики з необроблених даних робить це обґрунтованим пунктом методології. Більшість статистичних даних будуть отримані на основі сучасних висновків щодо математичних моделей та алгоритмів у математиці, які довели статистична теорія та теорія ймовірностей у математичній науці [15].

Маючи теоретичне підґрунтя щодо спортивної аналітики та аналізу даних загалом, а також математичних статистичних парадигм, аналіз було виконано після завершення впровадження та коректного збору даних. Частиною етапу впровадження є те, що аналітика даних має бути функцією статистичного застосування, тому розроблений інструмент для експорту коректної статистики з даних був важливою частиною системи.

Методологія проектування прототипів

Метод створення та проектування прототипу відповідав концепції MVP

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						57
Змн.	Арк.	№ докум.	Підпис	Дата		

Lean. Альтернативний процес Lean був присутній, оскільки клієнт та його потреби вже були визначені, що робило перші кроки непотрібними. Натомість це був етап складання списку специфікацій системи разом з потребами компанії. Після того, як цінності та функції для MVP були визначені разом із керівниками спортивного редагування, були створені блок-схеми. Блок-схеми використовувалися для опису можливого потоку взаємодії з прототипом та визначення того, що насправді потрібно розробити в кінцевому підсумку. Блок-схеми не відповідали жодному стандарту, але мали структуру, яка підходила для компанії.

Після встановлення блок-схем розпочнеться проектування прототипу. Потім буде представлена версія паперового прототипування, що дозволить провести мозковий штурм дизайну за кожним паперовим екземпляром [70]. Зацікавлені сторони будуть залучені до ітерацій зворотного зв'язку з керівниками та іншими співробітниками Sports Editing. Пізніше паперова версія буде перетворена на цифровий дизайн, який стане основою дизайну фронтенду застосунку. Варто зазначити, що інструмент звітності про спортивні ігри буде в центрі уваги дизайну фронтенду, оскільки саме з нею взаємодіють спортивні журналісти та статистики.

Однією з головних цілей застосунку є спрощення збору статистики для спортивної гри. Було розпочато етап оцінювання, щоб дослідити, наскільки дизайн застосунку відповідає вимогам. Були проведені якісні інтерв'ю зі спортивними журналістами та іншими спортивно-орієнтованими співробітниками за методом так званих напівструктурованих інтерв'ю.

Різниця між кількісними та якісними інтерв'ю полягає в тому, що перше містить багато чітких запитань зі чіткими відповідями, тоді як друге радше є показником того, що дозволяє респонденту відповідати більш вільно та з більш вільною структурою. Для цих інтерв'ю було обрано якісний підхід, оскільки багато роздумів користувача можуть краще вплинути на елементи дизайну застосунку, якщо він може вільно мислити, а не давати коротких відповідей. Ви можете або застосувати підхід повністю неструктурованого інтерв'ю, де ви просто розмовляєте з респондентом, або поставити лише одне запитання та дати йому

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						58
Змн.	Арк.	№ докум.	Підпис	Дата		

вільно говорити. Інший підхід – це так зване напівструктуроване інтерв'ю. Це свого роду якісне інтерв'ю з основою відкритих запитань, але зі структурою посібника з інтерв'ю. Посібник з інтерв'ю уточнив деякі питання, але інтерв'ю все ще динамічне і може призвести до інших структур. Ось чому напівструктуроване інтерв'ю більше підходить для такого виду оцінювання.

Респонденти прокоментували дизайн ситуації репортажу про матч з хокею. Основою інтерв'ю став посібник з інтерв'ю, який коротко окреслив мету проекту та різні частини програми. Оскільки це якісне дослідження, відповіді могли відрізнятися з кожним інтерв'ю. Було досліджено як загальний дизайн, так і досвід використання програми. Були включені такі питання, які представляли собою посібник з інтерв'ю:

1. Чи дизайн очевидний, і чи вважаєте ви додаток зручним для користувача ?
2. Чи вважаєте ви, що цей застосунок можна використовувати під час спортивної гри для створення відповідної статистики?
3. Чи є щось, що, на вашу думку, могло б покращити дизайн?

Ці питання призводять до узагальнення думок опитаних та того, чи має додаток потенціал для виконання свого призначення. Було опитано дванадцять осіб щодо дизайну додатку . Відгуки та відповіді будуть детальніше обговорені в розділі обговорення звіту.

2.4 Висновок по розділу

У другому розділі розглянуто методи та інструменти розробки веб-додатку для аналізу спортивної статистики. Проведено аналіз підходів до візуалізації даних, порівняно архітектурні рішення та обґрунтовано вибір технологічного стеку для реалізації системи. Визначено ключові інструменти, що забезпечують ефективну обробку та подання спортивної аналітики. У результаті сформовано архітектурну основу веб-додатку.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						59
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 РОЗРОБКА, ІНТЕГРАЦІЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ ВЕБ-ДОДАТКУ

3.1 Проектування моделей даних та структури спортивної статистики

У цьому підрозділі показано результати підготовки та реалізації статистичної програми. У розділах описано різні етапи розробки, які призвели до створення прототипу. Спочатку було проведено детальну презентацію моделей даних, *створених для програми, з візуальним представленням їх за допомогою ER-діаграми. Також розглядаються статистичні моделі, що використовуються для створення відповідної спортивної статистики. Після цього представлено розділ «Порівняння веб-технологій», у якому розглядаються різні технології та причини їх вибору для впровадження.*

Архітектурний підхід до всієї системи пізніше представлено в розділі «*Проектування та архітектура системи*» з компонентами та обраними веб-технологіями, які будуть реалізовані в процесі розробки. Також представлено результуючий дизайн прототипу та варіанти дизайну взаємодії, які призвели до його створення. Нарешті, є розділ, присвячений тематичному дослідженню, де статистики та спортивні журналісти використовували програму, та відгукам, які вони дали про дизайн в інтерв'ю.

Відправною точкою структурування моделей даних було їх візуальне створення за допомогою інструменту візуалізації. Моделі даних були візуалізовані у вигляді ER-діаграми, яка була побудована для реалізації моделей даних. Інструментом, що використовувався для створення ER-діаграми, є програма під назвою уED, написана на Java для створення ER-діаграм або іншого моделювання даних відповідно до інших мов або налаштувань (див. рис. 3.1) .

Модель даних має важливе значення, починаючи з вибору типу бази даних командних видів спорту, з якою користувач взаємодіятиме. Цей вибір визначає, які саме властивості матиме модель спортивних даних. Різними можуть

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						60
Змн.	Арк.	№ докум.	Підпис	Дата		

бути, наприклад, кількість періодів, тривалість періоду, кількість гравців на полі або змінна, яка визначає, чи є це ефективний час чи ні.

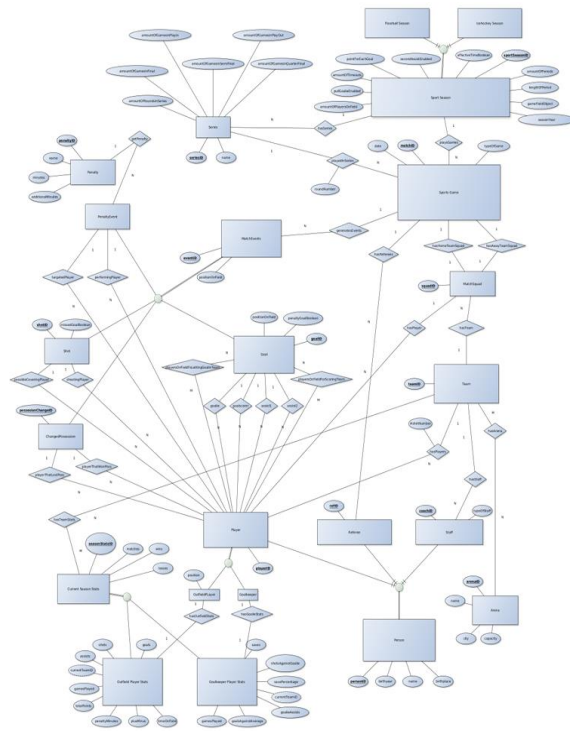


Рисунок 3.1 - Модель даних для статистичної системи.

Модель даних для програми була застосована з урахуванням структури сезону. Це означає, що після вибору спортивних даних, які будуть створені, оновлені або видалені, ініціюється вибір конкретного *спортивного сезону*. Сезон потрібно вказати за допомогою властивостей *серії*, наприклад, як гратимуться

плей-офф, назва серії та кількість раундів у сезоні серії. Сезон передбачає багато *спортивних ігор*, які містять інформацію про багато сутностей. Спортивна гра включає дві *команди*, кожна з яких має *склад матчу*, визначений для вибраної гри, в якому є *гравці*. Спортивні ігри також включають *персонал команд* та *суддів*. Усі такі властивості, як номер особи, ім'я та рік народження.

Спортивні ігри генерують багато *Подій Матчу*, які можуть бути дуже різними. Якою б не була подія, позиція події зберігається. Частою подією в хокеї та флорболі є *Кидки*. У цій сутності такі властивості, як хто гравець, що кидає,

чи був кидок прихований іншим гравцем, чи влучив кидок у ворота чи ні. Те саме стосується сутності *Зміна Володіння*, коли м'яч або шайба втрачається на користь іншої команди. Гравці, які втрачають та приймають м'яч, є властивостями для цієї сутності.

Події, пов'язані зі штрафами, – це події, які отримують правильну інформацію та правильний час відповідно до правил від сутності «Штраф». У списку перелічено гравця, який виконав штраф і отримає його, але також може бути вказана необов'язкова жертва штрафу. Нарешті, присутня сутність «Гол» з багатьма властивостями, оскільки вона безпосередньо впливає на статистику гравців та команд. Зберігається гравець, який забив гол, разом із гравцем, який віддав асист, та необов'язковим другим гравцем, який віддав асист, якщо такий був і гра це дозволяє. Воротар, який стояв разом з усіма гравцями на льоду, також зберігається для статистики плюсів та мінусів. Також присутнє логічне значення для визначення, чи був це гол з пенальті.

У моделі даних гравців часто згадують як членів команди та пов'язаних з подіями матчу. Позиція гравця зберігається на додаток до звичайних успадкованих властивостей особи. Гравці також пов'язані з однією з сутностей статистики поточного сезону залежно від позиції гравця. Польові гравці містять такі властивості, як удари по воротах, ідентифікація поточної команди та відсоток відбитих кидків. Статистика поточного сезону постійно оновлюється.

Структура сезону дозволяє включати гравців до моделі даних та постійно оновлювати поточну статистику за сезон. Це може призвести до створення моделі даних для сезону, де можна зберігати багато сезонів та порівнювати пізніше, якщо це необхідно. Спортивний сезон не прив'язаний до найвищого дивізіону, оскільки серія вирішує, чи це Дивізіон 5, чи Дивізіон 1, що робить модель даних застосовною для будь-якої серії з хокею та флорболу.

Математичні та статистичні моделі

Більшість подій, про які повідомляється в грі, а потім зберігається в документі спортивної гри, орієнтовані на гравців і містять інформацію про положення спортивного поля. Додаток надає як x-, так і y-координати. Знаючи положення

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						62
Змн.	Арк.	№ докум.	Підпис	Дата		

кожного запису, орієнтованого на гравця, різні частини поля можна розділити на зони. Зони можуть відрізнятися за розміром і положенням, оскільки деякі зони мають більш детальні зони. Безпосередньо перед воротарем присутні додаткові зони, оскільки ми хочемо знати, звідки точніше забиваються голи (див. рис.3.2) . Подібний поділ на зони було зроблено для поля для флорболу.

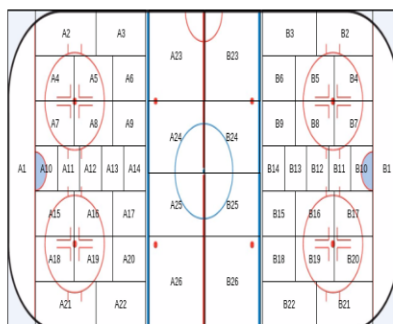


Рисунок 3.2 - Хокейний майданчик, розділений на позиційні зони для певних ігрових ситуацій

Коли нам відомі зони, ми можемо обчислити, яка подія матчу відбувається найчастіше, у цій зоні, та отримати загальну суму для цієї конкретної події шляхом підсумовування всіх подій з усіх зон. Якщо X_i позначає суму певного типу події в зоні i , то всі суми, підсумовані та поділені на загальну кількість зон, отримають середнє значення для події у спортивній грі загалом (див. рівняння 3.1)

$$\frac{1}{52} \sum_{i=1}^{52} X_i \quad (3.1)$$

Знаючи загальну кількість подій певного типу, що з'являються у спортивній грі, ми можемо отримати відсоток від загальної кількості подій, розрахований для кожної зони. Формула для визначення відсотка від загальної кількості подій певного типу, що відбулися в зоні i , - це X_i , поділений на загальну кількість подій (див. рівняння 3.2) є

$$\frac{X_i}{X_{tot}} \quad (3.2)$$

Це може призвести до графічного представлення спортивної гри, де можна відображати кількість подій певного типу для кожної зони або відсоток їх виникнення в зонах. Це також можна фільтрувати, показуючи статистику лише для господарів або гостей. Під час відображення статистики за сезон це також може бути корисним, отримуючи кількість голів, забитих у певній зоні для всієї команди або для конкретного гравця. Користувач фактично встановлює обмеження щодо того, які дані повинні бути частиною функцій середнього та відсоткового значення.

Щодо загальних подій матчів, які також включені до програми, на основі даних з документів матчів робляться прості суми. Наприклад, під час розрахунку кількості тайм-аутів, які команда використовує в середньому, обчислюється та експортується середнє значення, медіана та стандартне відхилення. Оскільки підсумовування є частиною розрахунку на серверній частині, це просто підсумовує кількість записів для певного типу події матчу та експортує бажаний результат для цього конкретного контексту. Це також можна зробити, маючи в базі даних спеціальний документ, який завжди оновлюється останніми сумами команди чи гравця за цей конкретний сезон, що потребує менше обчислень. Недоліком цього може бути те, що конкретні питання до бази даних можуть не охоплюватися цими зведеними документами.

Інші статистичні дані, які можна підсумувати, це загальна кількість голів команди за сезон, загальна кількість голів гравця та фактична таблиця за поточний сезон. Підсумовування подібне до Розрахунку 3.1, але з іншими підсумками та граничними числами, тоді як таблиця сезону містить параметри, які можуть потребувати як підсумовування, так і відсотків, наприклад, відбиті воротарем удари. Коротше кажучи, дуже проста математика підсумовування та обчислення середнього значення та відсотків забезпечує достатньо інструментів для створення необхідної статистики для прототипу спортивної статистики.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						64
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2 Реалізація веб-додатку та інтеграція сучасних веб-технологій

Різні веб-технології в цій роботі будуть порівняні відповідно до встановленої методології. Усі аспекти будуть оцінені як *Добрий*, *Середній* та *Поганий* відповідно до теоретичного дослідження.

Бекенд

Бекенд-фреймворки відрізнялися за своєю природою та забезпечували деякі різні результати (див. табл. 3.1). Ми бачимо, що Flask не отримує таких високих оцінок, як інші бекенд-фреймворки, щодо важливих питань для прототипу як веб-застосунку, таких як простота та можливість повторного використання. Flask є досить низькопродуктивним фреймворком з точки зору розробки прототипів, оскільки більшу частину роботи потрібно виконувати вручну. Також бракувало атрибутів середовища виконання, таких як безпека та надійність, що робить його менш вигідним вибором для програми спортивної статистики.

Таблиця 3.1

Порівняння між бекенд-фреймворками.

Атрибути / Фреймворк	Flask	Express	Laravel
Атрибути впровадження			
Підтримуваність (Maintainability)	Добре	Добре	Добре
Тестованість (Testability)	Середньо	Середньо	Добре
Портативність (Portability)	Добре	Добре	Добре
Гнучкість (Flexibility)	Добре	Добре	Добре
Простота (Simplicity)	Середньо	Середньо	Добре

Продовження таблиці 3.1

Атрибути часу виконання			
Доступність (Availability)	Середньо	Добре	Добре
Безпека (Security)	Середньо	Середньо	Добре
Продуктивність (Performance)	Добре	Добре	Добре
Паралелізм (Concurrency)	Добре	Добре	Добре
Надійність (Reliability)	Середньо	Середньо	Середньо
Масштабованість (Scalability)	Добре	Добре	Добре
Бізнес-атрибути			
Вартість (Cost)	Середня	Середня	Середня
Життєвий цикл (Lifetime)	Середній	Добрий	Добрий

Порівняння Express та Laravel було більш рівною боротьбою, але якщо врахувати важливі для прототипу цінності, такі як безпека "з коробки" з інтегрованою системою автентифікації та простота початку використання вбудованих функцій, це призвело до того, що Laravel був вибором бекенду для програми спортивної статистики, включаючи підтримку фронтенд-фреймворків *Vue.js* та *bootstrap*, компілятора CSS *Sass* та модуля-пакета *Webpack*.

Техніка баз даних

Вибір Laravel як бекенд-фреймворку для застосунку відкрив багато можливостей для роботи з базами даних. З коробки можна було вибирати між MySQL та SQLite лише за допомогою простої конфігурації в Laravel. Щодо потоку спортивних даних у застосунку, продуктивність була більшою проблемою, ніж простота реалізації. Згідно з теоретичним дослідженням, бази даних на основі

документів мають вищу швидкість виконання обчислень, тому для реалізації встановленої NoSQL-бази даних MongoDB було впроваджено сторонню бібліотеку під назвою *jenssegers-mongodb*. Система спортивної статистики хоче легко зберігати, обчислювати та експортувати дані спортивних ігор, тому було обрано документоорієнтований підхід MongoDB.

Фронтенд

Порівняння різних бекенд-фреймворків було цікавим, оскільки вони мали такі різні аспекти та рішення проблем. Цього не передбачали фронтенд-фреймворки, оскільки всі вони є популярними JavaScript-фреймворками працює з чудовою функціональністю (див. таблицю 3.2). У теоретичному дослідженні було важко знайти літературу для порівняння різних фреймворків, але деякі відмінності були виявлені.

Простота є перевагою під час прототипування, а Vue дозволяє обробляти дані безпосередньо в браузері з найменшою кількістю рядків коду. Порівняння з React, який має таку ж продуктивність, вимагало реалізації функціональності стану та більше коду для того ж результату. Те саме стосується Angular, де кращим стеком бекенду був би Express замість Laravel.

Таблиця 3.2

Порівняння фронтенд-фреймворків.

Атрибути / Технологія	Angular	React	Vue
Атрибути впровадження			
Підтримуваність (Maintainability)	Добре	Добре	Добре
Тестованість (Testability)	Добре	Добре	Добре
Портативність (Portability)	Добре	Добре	Добре
Гнучкість (Flexibility)	Добре	Добре	Добре

Продовження таблиці 3.2

Простота (Simplicity)	Середньо	Середньо	Добре
Атрибути часу виконання			
Доступність (Availability)	Добре	Добре	Добре
Безпека (Security)	Середньо	Середньо	Середньо
Продуктивність (Performance)	Добре	Добре	Добре
Паралелізм (Concurrency)	Середньо	Середньо	Добре
Надійність (Reliability)	Середньо	Середньо	Середньо
Масштабованість (Scalability)	Добре	Добре	Добре
Бізнес-атрибути			
Вартість (Cost)	Середня	Середня	Середня
Життєвий цикл (Lifetime)	Добрий	Добрий	Добрий

Laravel включає підтримку Vue безпосередньо "з коробки" з чудовою продуктивністю, що дає Vue велику перевагу в порівнянні. В інших аспектах вони схожі в багатьох випадках, всі з чудовою продуктивністю. Оскільки контекстом цієї роботи є розробка прототипу, Vue став вибором для системи спортивної статистики.

Дизайн та архітектура

Після вибору стеку рішень можна визначити архітектуру. Laravel з MongoDB як методом роботи з базою даних та Vue.js як фронтенд-фреймворком є обраним стеком рішень для розробки. Це забезпечує реалізовану підтримку Apache в Laravel, створюючи просту настройку для роботи бекенду (див. рисунок 3.3) .

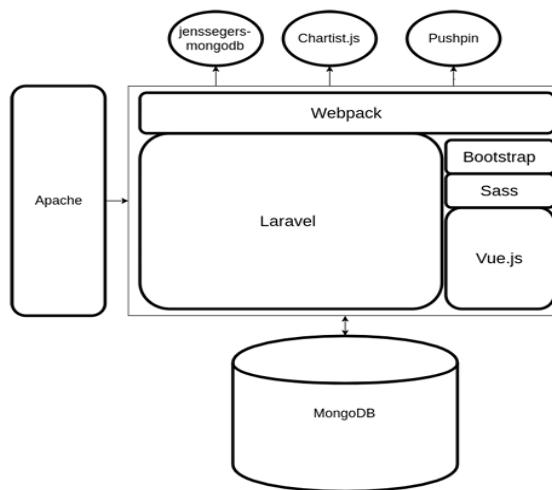


Рисунок 3.3 - Дизайн та архітектура статистичної системи.

Архітектура стеку рішень забезпечує поєднання багаторівневої архітектури та деяких мікросервісів. Laravel в основному побудований на архітектурі MVC, що дозволяє моделям, представленням та контролерам легко працювати разом. MVC можна назвати наступником багаторівневої архітектури, оскільки взаємодія через запити здійснюється на верхньому рівні, рухаючись далі до рівня бази даних, а рух вгору оновлює представлення.

Мікросервіси системи Laravel echo – це те, де легко вмикати більше сторонніх пакетів. Були додані вже згадані Pushpin та jenssegers-mongodb, які мали функції, подібні до мікросервісів, розділяючи роботу між різними зовнішніми компонентами. Також було додано chartist.js для спрощення перегляду експортованих даних. Оскільки багато сторонніх програм вже були включені в систему echo, це не означає нічого, крім того, що вони також є мікросервісами, тому чітко зазначено, що в застосунку спортивної статистики присутня архітектура мікросервісів.

Дизайн прототипу

Отриманий дизайн прототипу відповідно до альтернативних версій Lean та Paper Prototyping призвів до розробки дизайну для спортивної ігрової звітності. По-перше, блок-схеми показали доступні в додатку взаємодії.

Блок-схема призвела до створення дизайну, намальованого на папері та обробленого за допомогою Sports Editing. Остаточний малюнок потрапив до

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						69
Змн.	Арк.	№ докум.	Підпис	Дата		

цифрової версії, яка пізніше призвела до розробки фактичного інтерфейсу статистичного застосунку.

Потік дій

Взаємодія програми для звітів про матчі перелічена у блок-схемі (див. рисунок 3.4) . Спочатку програма пропонує користувачеві визначити, про який вид спорту буде повідомлятися, вибираючи між хокеєм та флорболом. Після встановлення правильних властивостей встановлюється інформація про сезон, серію та матч. Інформація про матч включає арену, суддів та дату, серед інших властивостей. Потім обираються господарі та гості, а потім обираються гравці з команди, які представлятимуть команду у самій грі.

Коли користувач вибере все це, завантажиться інтерфейс програми. Компонентами, що представляють програму, є *Список команд*, *Ігрове поле*, *Стрічка даних* та *Контролер часу* . У Списку команд перелічено всіх гравців, що робить їх доступними для подій.

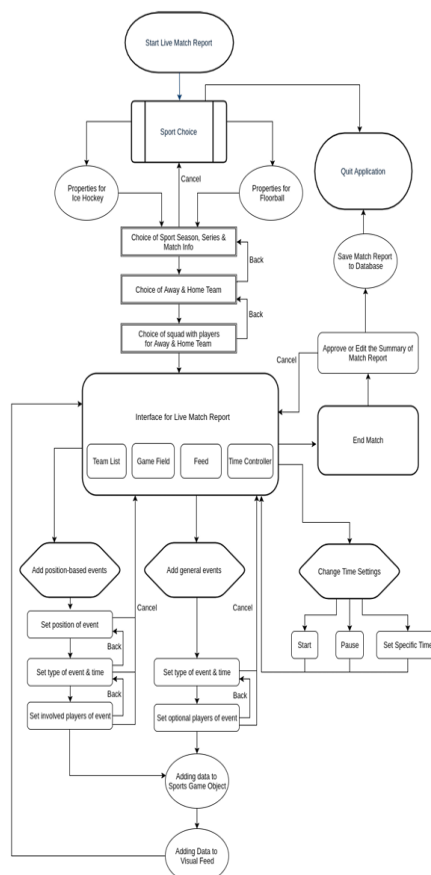


Рисунок 3.4 - Блок-схема програми звітності про матчі.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		70

Коли спрацьовує подія, що залежить від позиції, спочатку потрібно встановити позицію, а потім тип події, яка відбувається і в який час. Після цього вибираються гравці, що беруть участь у події, де кількість залежить від типу події. Ви маєте можливість підтримати або скасувати запис події, якщо бажаєте. Якщо все правильно, запис надсилається до спортивного ігрового об'єкта, який збирає дані, разом з оновленням стрічки інформацією про щойно опублікований запис. Цей цикл є найпоширенішим у застосунку, але є також загальні події, які не залежать від позиції гравців, наприклад, тайм-аут для команди гостей. Тут є подібний цикл, але пропускається крок визначення позиції на полі. Варто зазначити, що залучені гравці можуть бути необов'язковими в цьому випадку, наприклад, перерва в хокеї.

Контролер часу – це компонент, який забезпечує годинник реального часу для гри, яким користувач повинен керувати вручну. Фактор синхронізації реального часу з грою був складним питанням, оскільки він поглинав увагу користувача для правильного виконання. Це призвело до рішення ввімкнути інструмент корекції подій матчу в кінцевому підсумку. Як уже згадувалося, чи можна детально встановити час для події під час її створення?

Коли всі події матчу зареєстровані та гра завершена, застосовується функція завершення матчу. Тепер буде представлено зведення гри, де можна змінити редагування, наприклад, щодо першого автора голу або часу перехресної перевірки. Після виправлення та затвердження звіт про матч зберігається в базі даних як спортивний ігровий об'єкт, після чого програма має закритися.

Концепція дизайну

Після створення блок-схеми розпочався етап проектування застосунку. Оскільки всі компоненти були визначені в блок-схемі, етап проектування розпочався з їх реалізації різними способами. Платформою тут був обраний ПК, оскільки він забезпечував би найдоступніший та найшвидший ввід даних, як за допомогою кліків мишею, так і за допомогою комбінацій клавіш для команд на клавіатурі.

Одним із перших компонентів, розроблених за допомогою мозкового

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						71
Змн.	Арк.	№ докум.	Підпис	Дата		

штурму, було ігрове поле. Майже спочатку було прийнято рішення про впровадження ігрового поля в застосунок у вигляді інтерактивного зображення. Клацання мишею на ігровому полі призводить до отримання координат на полі, збереження зони, до якої воно належить, а потім до розробки специфікації події. Завдяки такому підходу поля для хокею та флорболу у вигляді зображень були реалізовані відповідно до застосунку. Такий підхід робить ігрове поле центральною точкою взаємодії з користувачем, додаючи дані, орієнтовані на позицію.

Списки гравців також були бажаним компонентом, де дизайн включав розмежування як гостей, так і господарів поля на різних сторонах ігрового поля. Дизайн продовжився, і номер футболки та позиція були включені до списків. Ідея полягає в тому, що ви повинні натиснути на гравця після встановлення позиції та конкретної події, що призведе до кліків на гравців, щоб просто пов'язати їх із подією.

Налаштування контролера часу зайняло більше часу та ітерацій. Остаточна концепція дизайну включає більші кнопки відтворення та паузи, оскільки користувач буде багато взаємодіяти з ними та має бути доступною. Їх розмістили поверх програми, поруч із годинником. Початкова інформація про гру, яка була встановлена під час запуску програми, відображається в блоці інформації про гру. Тут відображається статистична інформація під час гри, наприклад, список суддів та місткість арени. Поруч із нею розташована кнопка для додавання загальної події, не прив'язаної до позиції. Вона велика та помітна, оскільки використовуватиметься з великою частотою.

Нарешті, до дизайну додається стрічка даних зі списком усіх записів спортивної гри. Тут час події, тип події та залучені гравці повинні бути перелічені та оновлені в режимі реального часу. Спочатку дизайн передбачав, що стрічка буде великою, але на пізніших ітераціях вона стала меншою, оскільки мала працювати більше як перевірка правильності оновлення запису. Цей паперовий прототип був оцифрований у концептуальний прототип і став основою для фактичного дизайну фронтенду (див. рисунок 3.5) .

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		72

Фактичний дизайн

Прототип концепції було реалізовано з використанням стеку рішень, обраного для веб-застосунку. Дизайн дещо змінився під час впровадження разом з обговореннями та відгуками від Sports Editing, але більша частина вже встановленого дизайну була виконана (див. рисунок 3.6).

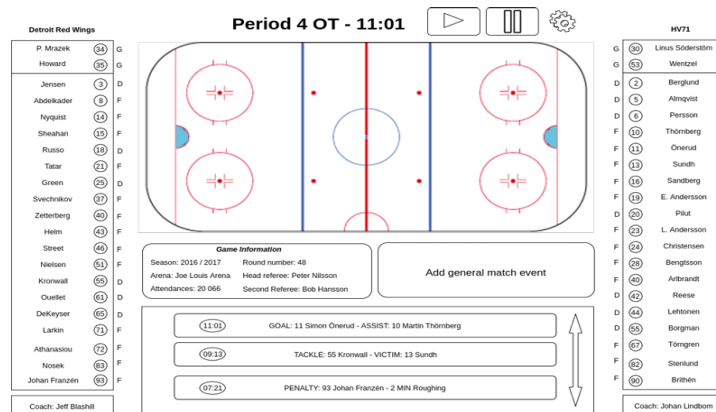


Рисунок 3.5 - Прототип: Концептуальний дизайн.

До ігрового поля було додано слухач, який виявляє клацання миші на зображенні. Це відкривало так званий модальний компонент, який є версією спливаючого вікна в бібліотеці Vue.

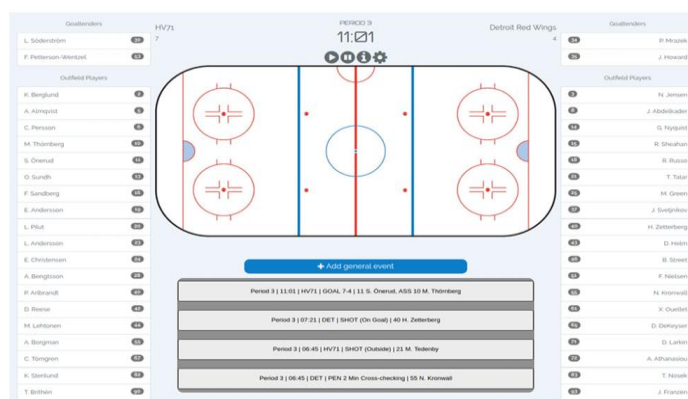


Рисунок 3.6 - Прототип: Фактичний дизайн.

Тут ви вирішуєте, який тип події буде зареєстровано, а потім натискаєте на потрібного гравця або гравців, які брали участь у ньому, за допомогою курс

ора миші. Були реалізовані ярлики до різних подій, що дозволило швидше введення даних. Стрічка постійно оновлюється з кожним новим записом у режимі реального часу, оскільки фреймворк Pushpin забезпечує миттєві оновлення на дисплеї.

Інформація про гру була одним із питань, яке займало багато місця в концептуальному прототипі, тому під час реалізації було обрано більш плавний дизайн. Поруч із контролером часу, який був реалізований там само, де й у концептуальному прототипі, було додано невелику піктограму для інформації, при натисканні на яку з'являється модальне вікно з інформацією про гру. Це заощадило багато місця в центрі програми та зробило дизайн чистішим. Це також розширило та централізувало кнопку для додавання загальних подій у програмі, надаючи їй більшої релевантності в дизайні та легшого доступу. Варто зазначити, що позиції гравців були видалені зі списку гравців, тоді як номери футболок залишилися для ідентифікації.

Оцінка дизайну прототипу

Дизайн прототипу застосунку оцінювали дванадцять осіб, які працюють у спортивному та медійному контексті. Основою інтерв'ю став посібник з інтерв'ю, згаданий у розділі про метод, а напівструктурована форма призвела до чітких та відкритих відповідей. Предметом оцінювання був інструмент для звітності про матчі в застосунку, зосереджуючись на його інтерактивності, дизайні та функціональності.

Перше питання стосувалося того, чи був досвід використання застосунку зручним для користувача. Усі дванадцять опитаних вважали застосунок зручним для користувача. Більшість зазначила, що це був справді зручний і простий у використанні досвід. Більшість зазначила, що простий та зрозумілий підхід був корисним для виконання його мети. Двоє опитаних вважали, що вступ, наданий автором щодо використання застосунку, був необхідним для зручності використання, але погодилися, що всі компоненти застосунку були легкими для розуміння та використання. Один вважав, що це зручний досвід використання, але потребує певного покращення компонентів щодо зворотного зв'язку про кожну

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						74
Змн.	Арк.	№ докум.	Підпис	Дата		

взаємодію, наприклад, маркер на полі, де було встановлено позицію події, та можливість переміщення маркера на нову позицію.

Друге питання стосувалося того, чи вважає респондент, що застосунок може належним чином працювати для створення спортивної статистики. Це питання було найсуворішим за своєю структурою та отримало найспланованіші відповіді. Усі дванадцять опитаних оцінили роботу застосунку в цьому контексті. Усі вони висловили задоволення від використання застосунку для створення спортивних подій. Деякі з опитаних зазначили, що швидкість введення спортивної події була корисною, а наступний крок був легким. Деякі опитані вважали, що використання застосунку є стресовим для однієї людини, і припустили, що в подальшій роботі система дозволить кільком користувачам вводити дані одночасно, зосереджуючись на різних аспектах гри.

Останнє запитання мало більш відкритий характер щодо покращень дизайну або робочого процесу репортажів про матчі. У опитаних було багато думок щодо подальшої роботи та вдосконалення застосунку. Майже половина з них вважала, що більш кольоровий дизайн корисний для застосунку. Деякі пропонували, щоб господарі поля мали певний колір, а гості – інший, що зробило б різницю між командами ще сильнішою. Майже половина опитаних також зазначила, що хотіла б розмістити логотип або футболку команди поруч із годинником, щоб легше ідентифікувати кожну команду візуально. Чотирьом опитаним сподобався більш чистий дизайн, і вони не хотіли б, щоб більше кольорів чи логотипів псували враження. Однак майже всі опитані зазначили, що текст або номери футболок хокеїстів мають бути більшими, щоб їх було легше ідентифікувати та взаємодіяти з ними під час репортажів. Іншими запропонованими подальшими роботами було сортування гравців за іменем, лінією чи позицією, отримання інформації при натисканні на кожного гравця у списку та можливість вибору, чи мають списки гравців містити імена, чи просто блоки з номерами футболок.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						75
Змн.	Арк.	№ докум.	Підпис	Дата		

3.3 Оцінка ефективності веб-додатку, результати тестування та перспективи розвитку системи

У цьому підрозділі представлені обговорення щодо результатів, методу та подальшої роботи. Результат спортивної статистичної системи та процес її створення могли бути покращені певним чином, і вони також дали хороші результати. Метод також аналізується, що дають роздуми про те, що можна було б покращити, і чи була теоретична основа методів міцною. Нарешті, обговорюються деякі подальші дослідження.

Обговорення результатів в основному стосується вибору методів для програми та результату їх реалізації. Обрана архітектура буде коротко обговорена разом з основними проблемами, які створює кількість JavaScript-фреймворків у цій ситуації. Далі обговорюється спортивна аналітика, що надається програмою.

Стек та архітектура програмного забезпечення. Фреймворки для бекенду, згадані та порівняні в роботі, були справді різними та мали унікальні перспективи реалізації веб-проектів. Flask з його легким підходом «зроби сам» на Python, Express з підходом Node.js, що дозволяє використовувати асинхронне програмування на основі подій, і, нарешті, Laravel, який постачається з повним рішенням та включеними функціями. Вибір використання Laravel був головним чином заснований на тому, що це було гарне рішення для початку та створення аплікації одразу. З усіма включеними функціями, починаючи як з фронтенд-проектуювання, так і з бекенд-маршрутизації. Реалізація Bootstrap та Vue.js значно полегшила створення функціональних додатків у найкоротші терміни порівняно з іншими фреймворками. Основу PHP також було легко засвоїти, оскільки вона вже має сформовану базу користувачів, а постійно зростаюча спільнота Laravel полегшує отримання інформації про кожну проблему.

Налаштування Flask легко зайняло б забагато часу та не таку велику винагороду за зусилля, окрім того, що він міг би активувати мову R, яка добре підходить для обчислень. Додатковий час, який знадобився б для створення міцної основи для програми, не був би виправданим. Бібліотека Node.js мала б

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						76
Змн.	Арк.	№ докум.	Підпис	Дата		

працювати, особливо з бекенд-фреймворком Express. Асинхронне програмування могло б спрацювати для програми з хорошим результатом. Laravel був переважно обраний замість Express через комплексне рішення, яке він надає для створення прототипів, але все ж він є гарною альтернативою для масштабування, а завдяки усталеній спільноті веб-розробка стає безпроблемною. Express був протестований на етапі впровадження, але надав мало додаткових функцій для розробки порівняно з Laravel. Програма також краще синхронізувалася з уже існуючими програмами Symfony, які використовувалися для редагування.

Оскільки застосунок містить стрічку, де ви бачите події спортивної гри, що постійно оновлюються, структура подій Node.js могла б бути корисною замість потокового PHP-фреймворку Laravel, який був обраний. Оскільки Laravel може включати сторонні пакети, не було складно реалізувати бібліотеку під назвою *Pushpin*. Ця бібліотека працює подібно до сокетного рішення, яке забезпечує прямий контакт через певний порт, але вона працює локально та дозволяє надсилати дані в режимі реального часу.

Хоча MongoDB не був стандартно включений до Laravel, вибір саме його реалізації дав хороший результат. Концепція отримання повного документа зі спортивної гри забезпечує хороший огляд та хорошу продуктивність під час обчислень, оскільки лише один запит до бази даних дозволяє отримати всі дійсні дані для обчислення. Одним з недоліків було те, що використовувалася стороння реалізація MongoDB з відкритим кодом у Laravel, що призводило до появи відкритих проблем та помилок, які потрібно було виправити. Це не було серйозною проблемою на етапі розробки, але більш надійна реалізація в Laravel могла б заощадити час та зусилля на розробку.

Враховуючи використану архітектуру, Laravel використовує багаторівневу архітектуру з версією мікросервісів. Деяка подієво-керована архітектура може бути застосована до компонента *pushpin*, який доставляє *push*-пакети в режимі реального часу до програми. Це вважається компонентом системи, і тому *Pushpin* можна розглядати як сервісний компонент з архітектури мікросервісів. Це підкріплює тезу, згадану в теорії, що різні види архітектури можна

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						77
Змн.	Арк.	№ докум.	Підпис	Дата		

комбінувати, і не обов'язково виключає інший підхід.

Щодо вибору фронтенду, то точка зору була протилежною. Багато фронтенд-фреймворків, порівняних у роботі, мають схожість і важко розрізнити, який насправді є найкращим вибором. Два з трьох фреймворків працюють з двостороннім зв'язуванням даних, що може полегшити деякі проблеми розробки, водночас ускладнюючи налагодження. Усі три фреймворки популярні в деяких аспектах, і тенденції такі, що багато варіантів можуть працювати. Ці джунгли JavaScript-фреймворків є одночасно позитивними в тому сенсі, що для однієї й тієї ж проблеми пропонується багато різних рішень, тоді як одним негативним аспектом можна сперечатися про те, що розробники плутаються і хочуть опанувати певну мову, яка може застаріти завтра. У цьому випадку вибір використання Vue.js як фронтенд-фреймворку був актуальним і часто зустрічається на даний момент, а також завдяки підтримці, яку він отримав від спільноти Laravel, що дозволило його встановлення. Таке поєднання стеків між фронтенд- та бекенд-фреймворками, що спрощує життя розробникам, на мою думку, слід вважати більшою перевагою, ніж очікувалося. Laravel підтримує інші фронтенд-фреймворки, але для прототипування це є перевагою як для зосередження на створенні застосунку, так і для економії часу, оскільки все включено з самого початку.

Під час вибору та порівняння різних фреймворків не так багато наукових досліджень. Можливо, постійні зміни в бізнесі та спільнотах веб-розробки відбуваються зі швидкістю, за якою університети не можуть встигати. Більшість порівнянь, знайдених у літературі, взяті з інтернет-джерел, що може підірвати достовірність порівняння з науковими статтями. Зрештою, я думаю, що все зводиться до того, які функції підходять для проекту, та до рівня знань розробників певного фреймворку. Однак, є деякі цікаві відмінності в продуктивності, які присутні в результаті, і хоча між фреймворками може бути подібність, варто враховувати одну унікальну річ у фреймворку, вибираючи стек, щоб заощадити зусилля на розробку.

Моделі даних. Моделі даних, створені для програми, підходили для вирішення проблеми, зазначеної на початку. Сезонний підхід вирішив багато

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						78
Змн.	Арк.	№ докум.	Підпис	Дата		

проблем щодо того, як пов'язати гравця з командою в сезоні, який складається з команд з гравцями. Під час створення нового документа сезону він дозволяє скористатися багатьма перевагами MongoDB, порівнюючи та розраховуючи з хорошою продуктивністю. Набір документів сезону може легко охоплювати лігу протягом багатьох років, замість того, щоб створювати модель даних відповідно до виду спорту самого по собі, включаючи багато видів спорту.

Я вважаю, що розмежування командних видів спорту було гарною ідеєю. Модель даних не сильно відрізняється від інших усталених командних видів спорту у Швеції. Sports Editing має ділові контакти з командними видами спорту загалом, що робить моделі даних більш актуальними та гнучкими при додаванні нового командного виду спорту, наприклад, футболу чи гандболу.

Спортивна аналітика. Статистика, розрахована в додатку, дуже проста. Вона використовує лише розрахунки сум, середнього значення та медіани. Незважаючи на наявність простих розрахунків, цінна статистика все одно експортується з системи. Проблему, яку мала компанія зі створенням цінної спортивної статистики, можна вважати вирішеною навіть за допомогою найпростіших методів. Вона розраховується з хорошою продуктивністю, що є перевагою, і забезпечує основну та необхідну статистику для командних видів спорту, таких як хокей на льоду та флорбол.

Коли велика кількість даних зібрана, перетворена на статистику та з неї витягнута певна кількість знань, варто зазначити, що дані корисні, але це не вся правда. Деякі аспекти справді важко виміряти на основі зібраних даних про спортивні ігри, і їх можна відчувати або виміряти лише за допомогою інших інструментів, якщо це взагалі можливо. У цьому також полягає краса людської діяльності у спорті, адже не все можна і не буде виміряно статистикою.

Не всі рішення повинні залежати від аналітики, оскільки в хокеї існує так багато факторів, які неможливо реалізувати в такій системі. Такі терміни, як «хокейне чуття» та «розуміння гри», без абсолютних відповідей чи визначень, все ще є змінними, які згадуються в грі в хокей і можуть впливати на результати команди. Можна стверджувати, що гарне хокейне чуття буде видно в отриманих

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						79
Змн.	Арк.	№ докум.	Підпис	Дата		

данихдесь і з певною швидкістю, але також чітко зазначено, що, хоча технічний прогрес постійно розвивається, неможливо виміряти всі аспекти людської істоти в цифрах та статистиці. Саме це надає спорту перевагу та гостроту, які забезпечує недетермінована природа людини.

Це відповідає аспектам удачі та збігів обставин у спортивній грі. Навіть якщо всі висновки з даних вказують на певний напрямок, немає гарантії, що результат гри буде відповідати цьому конкретному напрямку. Це те, що не слід недооцінювати, і я думаю, що це надає кожній спортивній грі, незалежно від контексту, особливого хвилювання та азарту, яких інакше не було б у такому темпі. Людський підхід до кожної гри є важливим для інтересу вболівальників.

З огляду на це, спортивна статистика надає багато додаткової цінності не лише тренерам та особам, які приймають рішення, а й глядачам. У дискусії зазначається, що вболівальники зараз більш вимогливі та голодні, ніж будь-коли, і що всі ці дані, що надають більше цікавих фактів та статистики про їхню улюблену команду, завжди корисні. Якщо прогресивний розвиток спортивної аналітики надає вболівальникам більше цінності, це має велике значення та мету для цієї галузі досліджень, як і будь-чого іншого.

Відгуки про прототип були переважно позитивними. Багато опитаних зазначили, що застосунок мав гарний дизайн і був простим у використанні, що слід розглядати як позитивну оцінку. Мета була досягнута з дизайном у його поточному стані, і всі зазначили, що використання застосунку було зручним для користувача.

Пропозиції щодо подальшої роботи були конструктивними, і багато запропонованих покращень могли б призвести до кращого продукту. Одним із найцікавіших роздумів респондентів було те, що половина з них хотіла б більш барвистих візуальних елементів, тоді як майже всій іншій половині подобався більш строгий та чистий дизайн. Це розділяє відповіді та може розглядатися як характеристика особистого смаку в оцінюванні. Додаток міг би включати певні варіанти кольорів, щоб задовольнити обидві підмножини в оцінюванні, дозволяючи користувачеві чергувати відповідні кольори.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						80
Змн.	Арк.	№ докум.	Підпис	Дата		

Багато респондентів згадували про більших гравців або номери футболок, з якими можна взаємодіяти та асоціювати події матчів, що слід виправити в іншій ітерації дизайну. Ще одним відображенням була концепція використання блоків замість імен у списках, ідентифікуючи гравців за номерами їхніх футболок. Лише один респондент згадав про цей підхід, але він міг би вирішити проблему збільшення зображення гравця та спрощення взаємодії з ним. Додаток міг би виграти, якби користувач міг вибрати один із підходів.

У цьому підрозділі розглядається обговорення методу, перевірка валідності інструментів моделювання даних, допомога методології щодо розробки прототипу в процесі розробки та можливість покращення інтерв'ю зі спортивними журналістами та статистиками.

Інструментом моделювання даних, який використовувався для створення моделей, були ER-діаграми, створені в програмному забезпеченні під назвою *uEd*. Оскільки результуючий метод баз даних, після порівняння в теоретичному дослідженні, призвів до MongoDB, яка є документоорієнтованою технологією баз даних, інший інструмент міг би покращити результат.

Діаграми ER (та EER) є усталеним інструментом моделювання для створення баз даних у теорії. ER не є реляційною за визначенням, але реалізації, засновані на ній, можуть бути реляційними. MySQL (та SQL загалом) є найпопулярнішим способом реалізації та виконання теоретичних моделей у реальних базах даних згідно з RDBMS [36]. MongoDB не обов'язково є реляційною за своєю структурою, і, можливо, інший інструмент міг би створити більш відповідну теоретичну модель для бази даних. Деякі подальші теоретичні дослідження щодо MongoDB могли б покращити результат. Результат був у будь-якому разі солідним, і його можна було легко інтерпретувати для реляційної бази даних, якщо було потрібно.

Методологія проектування прототипів. Альтернативні версії Lean та паперового прототипування можна вважати гарним поєднанням інструментів для створення прототипів веб-застосунку. Оскільки Lean часто детально обробляє клієнтів, які не знають, чого хочуть, Lean допомагає знайти конкретну

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						81
Змн.	Арк.	№ докум.	Підпис	Дата		

потребу, яку продукт має вирішити. Однак у цьому випадку було створено специфікацію вимог разом із Sports Editing, пропустивши багато непотрібних кроків. Я думаю, що це допомогло так детально визначити, що має включати застосунок, але, можливо, звичайне дослідження Lean, перевірка зацікавлених сторін ринку та їхніх відгуків, могло б покращити результат таким чином, щоб спрямувати прототип у іншому напрямку. Однак робота була поставлена як проблема, сформульована Sports Editing, що призвело до створення застосунку, який був затребуваний. Варто зазначити, що паперове прототипування спрацювало дуже добре в контексті компанії та призвело до деяких ітерацій, які постійно покращували дизайн.

Методологія оцінювання. Методологію використання інтерв'ю для оцінки дизайну та зручності використання можна було б змінити, щоб отримати інші цікаві результати. Якби використовувалася більш сувора анкета, можна було б виключити деякі з більш наочних зворотних зв'язків та пояснень, які надавали розмови в ситуації інтерв'ю щодо подальшої роботи. Анкети можуть призвести до більш правдивих відповідей, навіть якщо вони є більш суворими за своєю структурою.

Щодо рішення про використання інтерв'ю, якісні та напівструктуровані підходи можна вважати успішними. Напівструктуроване інтерв'ю допомогло зосередитися, а також забезпечило гнучкість, дозволяючи респонденту вільно мислити відповідно до дизайнерських рішень та інших аспектів. Такий підхід міг би сприяти отриманню досить чітких відповідей, якби він був зручним для користувача разом із мозковим штурмом пропозицій щодо дизайну.

Подальша робота. Будуть представлені деякі обговорення щодо майбутньої роботи. В основному обговорюються різні варіанти збору даних у майбутньому, а також деякі подальші роботи, які можуть бути природними для розвинутої системи спортивної статистики.

Розробка кращого спортивного спорядження впливає не лише на елітних спортсменів, а й на звичайних людей. Це вплине на соціальні аспекти суспільства [1]. Вже сьогодні багато мобільних тренувальних додатків використовуються для

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						82
Змн.	Арк.	№ докум.	Підпис	Дата		

збору даних про тих, хто займається спортом. Я думаю, що це відкрило більше можливостей для окремих спортсменів, але ускладнило їх для командних видів спорту. Саме тому я вважаю, що подальший розвиток прототипу може бути корисним для звичайних людей у командних видах спорту.

Оскільки технології розвиваються, а GPS-пристрої стають дешевшими, з'являється можливість використовувати такі пристрої у спорті на будь-якому рівні. Спортивна конференція SSAC, де обговорювалося, як великі дані відкривають більше можливостей, у виданні 2016 року майже ніколи не згадувала ручне введення як альтернативу позиціонуванню. Це може бути природним розвитком для системи спортивної статистики, яка надає дані про позиціонування безпосередньо від гравців.

Однак, я думаю, що якщо цей прототип досягне певного прогресу на ринку, необхідність збереження простоти все ще буде прибутковою і може бути чимось вартим інвестицій. У майбутньому, можливо, поєднання GPS-пристроїв на гравцях та ручної реєстрації може бути корисним для більш простого підходу до збору даних спортивних ігор за допомогою веб-застосунку.

У цій роботі чітко було сформульовано обмеження, яке полягає у створенні простої статистичної моделі, що охоплює найбазовіші потреби спортивної статистики. Інші, більш складні статистичні дані можуть бути розраховані для забезпечення більшої підтримки прийняття рішень у спорті.

Першими статистичними показниками, які можна було б додати для підвищення цінності, могли б бути ті, що стосуються стандартного відхилення. Коробчасті діаграми могли б бути гарним візуальним інструментом для їх демонстрації, оскільки всі дані вже є для їх надання. Сторонній фреймворк Chartist.js не ініціює коробкові діаграми з налаштування, але інші сторонні бібліотеки могли б це зробити. Я думаю, що це додало б цінності застосунку. Згадана лінійна регресія та кластеризація даних у теорії могли б бути наступним кроком у більш віддаленому майбутньому для застосунку, забезпечуючи певний результат інтелектуального аналізу даних з отриманням тенденцій та рівня успіху у вигляді гарних графіків.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						83
Змн.	Арк.	№ докум.	Підпис	Дата		

Але як розширити застосування на некомандні види спорту, такі як теніс чи плавання? У цій конкретній версії це не застосовуватиметься і буде не вигідним для використання. Я думаю, що таке розмежування відкриває достатньо контекстів як ринок для спортивного редагування і не обов'язково має розширюватися на індивідуальні види спорту. Однак, застосування спортивної статистики в його поточному стані є надійною альтернативою для нижчих дивізіонів та видів спорту, які не мають багато грошей чи інвесторів.

У цій роботі досліджувалося, як дані спортивних ігор можна обробляти відповідно до моделей даних для належного створення цінної статистики. Отримані моделі даних, створені для застосування спортивної статистики, були придатними. Сезонний підхід, де обраний вид спорту успадковував сезонні властивості, зробив його менш складним та більш керованим, ніж повна спортивна база даних. Модель даних належним чином обробляла дані спортивних ігор та пов'язувала їх із сезонами, що полегшувало надання статистики, необхідної для сезону. Моделі даних, створені з метою забезпечення статистичних систем для сезонів хокею та флорболу, були успішними. Такий підхід спрощує додавання інших видів спорту з визначенням меж командних видів спорту, оскільки необхідні властивості присутні та можуть бути змінені відповідно до контексту іншого виду спорту.

У цій роботі також досліджувалося, як можна розробити веб-застосунок за допомогою сучасних технологій, щоб спростити збір даних про спортивні ігри. Щодо сучасних веб-технологій, було використано програмний стек, придатний для цільового призначення. Фреймворк Laravel підходить для створення прототипів веб-застосунків. Laravel надає все необхідне «з коробки» для створення веб-застосунків, які є одночасно надійними та масштабованими. Додаток спортивної статистики, де користувачі повинні мати можливість повідомляти дані спортивних ігор та виводити спортивну статистику, включено до висновку, що Laravel є сучасним підходом до створення веб-застосунків.

Реалізована в Laravel підтримка фронтенд-фреймворку Vue.js допомагає створити сучасний веб-застосунок «з коробки». Vue.js має все, що може

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						84
Змн.	Арк.	№ докум.	Підпис	Дата		

знадобитися сучасному веб-розробнику, і надає багато корисних можливостей. Інші фронтедн-фреймворки також є хорошими на даний момент і могли б вирішити проблеми для фронтедну спортивної статистики. Однак теоретичне дослідження показало, що Vue.js є надійним вибором для прототипування, а співпраця з бекендом Laravel зробила його вдалим вибором.

MongoDB - це технологія баз даних, яка обробляє спортивні дані, надаючи документи, що містять інформацію про події матчів. Ці документи містять дані, необхідні для відображення та розрахунку статистики. Згаданий підхід моделей даних, пов'язаний зі спортивним сезоном, було легко реалізувати відповідним чином у цій структурі документів, оскільки збір даних призвів до отримання результуючого документа сезону. MongoDB спростив реалізацію, навіть незважаючи на те, що він не був включений до фреймворку Laravel «з коробки».

Існує потреба у швидкому та інтуїтивно зрозумілому генеруванні статистики спортивних ігор. Блок-схема, створена в цій роботі, заклала гарну основу для зручного у використанні прототипу концептуального дизайну. Інструмент звітності про матчі в додатку спортивної статистики отримав хороші результати оцінювання. Отриманий дизайн додатку також був покращений, і, на думку респондентів, його можна розглядати як зручний та функціональний. Подальша робота над дизайном може призвести до створення продукту, який міг би сприяти збору даних про спортивні ігри належним та відповідним чином.

3.4 Висновок по розділу

У третьому розділі реалізовано веб-додаток для аналізу та візуалізації спортивної статистики та проведено оцінку його ефективності. Виконано проектування моделей даних, інтеграцію сучасних веб-технологій та тестування функціональних можливостей системи. Аналіз результатів підтвердив коректність роботи додатку та зручність візуалізації спортивних даних. Отримані результати свідчать про ефективність розробленого рішення та його практичну придатність для спортивної аналітики.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						85
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У ході виконання дипломної роботи було розроблено веб-додаток для аналізу та візуалізації спортивної статистики, який забезпечує збір, обробку, аналіз та інтерактивне представлення спортивних даних із використанням сучасних веб-технологій. Створення системи стало комплексним процесом, спрямованим на розробку функціонального, масштабованого та зручного програмного рішення для роботи зі спортивною аналітикою. У роботі було проаналізовано сучасні підходи до спортивної статистики, досліджено методи аналізу даних і засоби їх візуалізації, а також реалізовано функціональні можливості веб-додатку, що відповідають вимогам швидкодії, інтерактивності та зручності використання.

Роботу розпочато з аналізу предметної області та сучасних систем спортивної аналітики. Було визначено основні функціональні та нефункціональні вимоги до веб-додатку, зокрема необхідність швидкого доступу до статистичних даних, підтримки інтерактивної візуалізації, фільтрації інформації та масштабованості системи. Аналіз існуючих платформ дозволив виявити їх переваги та недоліки, що стало основою для формування структури майбутнього програмного забезпечення.

У процесі дослідження було розглянуто методи аналізу спортивних даних, підходи до моделювання статистичної інформації та засоби її графічного представлення. Значну увагу приділено порівнянню архітектурних рішень і технологічних стеків веб-розробки, що дозволило обрати найбільш ефективний підхід для реалізації системи. Було сформовано структуру веб-додатку, визначено моделі даних та спроектовано логіку взаємодії між клієнтською та серверною частинами системи.

Реалізація веб-додатку охопила створення інтерфейсу користувача, інтеграцію засобів візуалізації статистики та реалізацію механізмів обробки даних. Було розроблено модулі для відображення спортивної статистики у вигляді таблиць, графіків і діаграм, що забезпечують наочне представлення результатів аналізу. Особлива увага приділялася інтерактивності інтерфейсу, зручності навігації та швидкості обробки запитів користувачів.

У процесі тестування системи було проведено перевірку функціональності,

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						86
Змн.	Арк.	№ докум.	Підпис	Дата		

продуктивності та стабільності роботи веб-додатку. Аналіз результатів тестування дозволив виявити окремі проблеми, пов'язані з обробкою великих обсягів статистичних даних, що стало підставою для оптимізації структури запитів та механізмів завантаження інформації. Отримані результати підтвердили працездатність системи та відповідність поставленим вимогам.

Загалом, розроблений веб-додаток відповідає поставленим цілям і дозволяє ефективно виконувати аналіз та візуалізацію спортивної статистики. Перевагами створеного рішення є інтерактивність інтерфейсу, гнучкість структури даних, можливість швидкого аналізу статистичної інформації та зручність візуального представлення результатів. Використання сучасних веб-технологій забезпечує масштабованість системи та можливість її подальшого розвитку.

Водночас система має певні обмеження, пов'язані з продуктивністю при роботі з надвеликими наборами даних та залежністю від зовнішніх джерел спортивної статистики. У майбутньому веб-додаток може бути вдосконалений шляхом інтеграції алгоритмів машинного навчання для прогнозування результатів спортивних подій, підтримки аналітики в реальному часі, адаптації під мобільні платформи та розширення функціоналу візуалізації даних.

					БР.ІІІ - 15.00.00.000 ПЗ	Арк.
						87
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА

1. Angular Documentation. (2025). angular.io. <https://angular.io/docs>

2. React Documentation. (2025). react.dev. <https://react.dev/>

3. Vue.js Guide. (2025). vuejs.org. <https://vuejs.org/guide/introduction.html>

4. Chart.js Documentation. (2025). chartjs.org. <https://www.chartjs.org/docs/latest/>

5. D3.js Data Visualization Library. (2025). d3js.org. <https://d3js.org/>

6. Highcharts Documentation. (2025). highcharts.com. <https://www.highcharts.com/docs>

7. Bootstrap Documentation. (2025). getbootstrap.com. <https://getbootstrap.com/docs/>

8. Tailwind CSS Documentation. (2025). tailwindcss.com. <https://tailwindcss.com/docs>

9. Node.js Documentation. (2025). nodejs.org. <https://nodejs.org/en/docs>

10. Express.js Documentation. (2025). expressjs.com. <https://expressjs.com/>

11. MongoDB Documentation. (2025). mongodb.com. <https://www.mongodb.com/docs/>

12. PostgreSQL Documentation. (2025). postgresql.org. <https://www.postgresql.org/docs/>

13. REST API Design Best Practices. (2024). learn.microsoft.com. <https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design>

14. MDN Web Docs: JavaScript. (2025). developer.mozilla.org. <https://developer.mozilla.org/en-US/docs/Web/JavaScript>

15. TypeScript Documentation. (2025). typescriptlang.org. <https://www.typescriptlang.org/docs/>

16. FIFA Data API Documentation. (2025). fifa.com.
<https://www.fifa.com/fifaplus/en>
17. NBA API Documentation. (2025). nba.com. <https://www.nba.com/stats>
18. ESPN Developer APIs. (2025). espn.com.
<https://www.espn.com/apis/devcenter/docs/>
19. SportsDB API. (2025). thesportsdb.com.
<https://www.thesportsdb.com/api.php>
20. Kaggle Sports Analytics Datasets. (2025). kaggle.com.
<https://www.kaggle.com/datasets>
21. Few, S. (2013). Information Dashboard Design: Displaying Data for At-a-Glance Monitoring. analyticspress.com. <https://www.analyticspress.com/>
22. Ware, C. (2021). Information Visualization: Perception for Design. Morgan Kaufmann.
23. Munzner, T. (2014). Visualization Analysis and Design. CRC Press.
24. Bostock, M. (2024). Data Visualization with D3.js. observablehq.com.
<https://observablehq.com/@d3/learn-d3>
25. Cleveland, W. S. (1993). Visualizing Data. Hobart Press.
26. Fowler, M. (2002). Patterns of Enterprise Application Architecture. martinowler.com. <https://martinfowler.com/eaCatalog/>
27. Freeman, E., & Robson, E. (2021). Head First Design Patterns. oreilly.com.
<https://www.oreilly.com/library/view/head-first-design/9781492077992/>
28. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. pearson.com.
<https://www.pearson.com/store/p/design-patterns-elements-of-reusable-object-oriented/P100000252058>
29. Martin, R. C. (2017). Clean Architecture: A Craftsman's Guide to Software Structure and Design. pearson.com. <https://www.pearson.com/store/p/clean-architecture-a-craftmans-guide-to-software-structure-and-design/P100001465201>
30. Clean Code Principles in .NET. (2023). devblogs.microsoft.com.
<https://devblogs.microsoft.com/dotnet/clean-code-principles/>

					БР.ІІІ - 15.00.00.000 ІІЗ	Арк.
						89
Змн.	Арк.	№ докум.	Підпис	Дата		

31. OWASP Top Ten Security Risks. (2024). owasp.org.
<https://owasp.org/www-project-top-ten/>
32. Cypress End-to-End Testing. (2025). docs.cypress.io.
<https://docs.cypress.io/guides/overview/why-cypress>
33. Selenium Documentation. (2025). selenium.dev.
<https://www.selenium.dev/documentation/>
34. Postman API Testing Guide. (2025). learning.postman.com.
<https://learning.postman.com/docs/>
35. Osherove, R. (2019). The Art of Unit Testing. manning.com.
<https://www.manning.com/books/the-art-of-unit-testing-third-edition>
36. Sports Analytics: A Guide for Coaches, Managers, and Other Decision Makers. (2023). Routledge.
37. Davenport, T., & Harris, J. (2017). Competing on Analytics: The New Science of Winning. Harvard Business Review Press.
38. Alamar, B., & Mehrotra, V. (2011). Sports Analytics: A Guide for Coaches, Managers, and Other Decision Makers. Columbia University Press.
39. Wang, L., & Zhang, Y. (2025). Real-time analytics for interactive sports web applications. ACM Transactions on the Web, 19(1), 1–20.
<https://doi.org/10.1145/3647890> .
40. Chen, J., & Li, X. (2024). Web technologies for sports statistics visualization systems. IEEE Access, 12, 118450–118467.
<https://doi.org/10.1109/ACCESS.2024.3478901>

					БР.ІІІ - 15.00.00.000 ІІЗ	Арк.
						90
Змн.	Арк.	№ докум.	Підпис	Дата		

БІБЛІОГРАФІЧНА ДОВІДКА

Тема бакалаврської роботи: " Розробка веб-додатку для аналізу та візуалізації спортивної статистики"

Обсяг пояснювальної записки: 87 аркушів

Дата закінчення бакалаврської роботи 10 червня 2026р.

Підпис студента _____