МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри Є. О. Давиденко

«____» _____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Розробка програмного забезпечення інтернет-сервісу аналізу власних досягнень

Спеціальність 121 «Інженерія програмного забезпечення

121 – КРБ – 408.21910803

Виконав студент 4-го курсу, групи 408 _____*І. В. Белевят*

«___» червня 2023 р.

Керівник канд. техн. наук, ст. викладач_____ *К. О. Кірей*

«___» червня 2023 р.

Консультант канд. техн. наук, доцент _____ *А. О. Алексєєва*

«___» червня 2023 р.

м. Миколаїв – 2023 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри інженерії програмного
забезпечення, канд.техн.наук, доцент,
_____Є.О. Давиденко
«____»_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи бакалавра

Видано студенту групи 408 факультету комп'ютерних наук

Белевята Іллі Вікторовича.
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

Вебзастосунок аналізу власних досягнень.

Затверджена наказом по ЧНУ від «17» березня 2023 р. № 60

2. Строк представлення кваліфікаційної роботи «26» червня 2023р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Вхідні дані до роботи − функціональні та нефункціональні вимоги до вебзастосунку аналізу власних досягнень. Результат – функціонуючий вебзастосунку аналізу власних досягнень

4. Перелік питань, що підлягають розробці.

Провести аналіз сучасних автоматизованих систем та ресурсів орієнтованих на аналіз власних досягнень. Визначити їх сильні та слабкі сторони. Розробити специфікацію вимог до програмного забезпечення інтернет-сервісу аналізу власних досягнень. Проєктування та моделювання програмного забезпечення інтернет-сервісу аналізу власних досягнень.

Розробка функціональних модулів ПЗ та проєктування зручного та інтуїтивно зрозумілого інтерфейсу. Тестування ПЗ.

5. Перелік графічних матеріалів:

Презентація.

6. Завдання до спеціальної частини

Дослідження питань охорони праці, які безпосередньо пов’язані з діяльністю розробника програмного забезпечення.

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Алексєєва А.О., канд. техн. наук., доцент	Кафедра екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Завдання прийнято до виконання

Виконав студент 4-го курсу, групи 408 _____*І. В. Белевят*

«___» червня 2023 р.

Керівник канд. техн. наук, ст. викладач_____ *К. О. Кірей*

«___» червня 2023 р.

Дата видачі завдання « » 20 р

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: «Вебзастосунок аналізу власних досягнень».

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРБ	26.10.2022	27.10.2022	Виконано
2.	Огляд літератури за темою роботи	30.10.2022	02.11.2022	Виконано
3.	Складання календарного плану КРБ	20.01.2023	22.01.2022	Виконано
4.	Аналіз предметної області	23.01.2023	25.01.2023	Виконано
5.	Розробка проектних рішень	25.01.2023	26.01.2023	Виконано
6.	Моделювання та конструювання ПЗ	26.02.2023	28.02.2023	Виконано
7.	Кодування, тестування та апробація розробленого ПЗ	28.02.2023	04.03.2023	Виконано
8.	Розробка спеціальної частини з охорони праці	05.03.2023	10.03.2023	Виконано
9.	Відгук керівника КРБ	20.05.2023	21.05.2023	Виконано
6.	Оформлення КРБ та презентації	10.05.2023	22.05.2023	Виконано
7.	Попередній захист	25.05.2023	25.05.2023	Виконано
8.	Рецензування	26.05.2023	18.06.2023	Виконано
10.	Завершення оформлення КРБ та презентації	19.06.2023	21.06.2023	Виконано
14.	Захист кваліфікаційної роботи	24.06.2022	26.06.2023	

Виконав студент 4-го курсу, групи 408 _____*І. В. Белевят*

«___» червня 2023 р.

Керівник канд. техн. наук, ст. викладач_____ *К. О. Кірей*

«___» червня 2023 р.

АНОТАЦІЯ

до кваліфікаційної роботи бакалавра

«Вебзастосунок аналізу власних досягнень»

Студент 408 гр.: Белевят Ілля Вікторович

Керівник: ст. викладач Кірей Катерина Олександрівна

Метою роботи є підвищення ефективності досягнення особистої мети шляхом автоматизації процесів аналізу та відбору досягнень в різних сферах життя, розробки плану дій для покращення результатів в майбутньому.

Об’єктом дослідження є організація та управління власними досягненнями.

Предметом дослідження є метод організації управління власними досягненнями з використанням вебтехнологій.

В роботі обґрунтовується актуальність дослідження та ставиться проблема, мета та завдання проекту, визначається об'єкт та предмет дослідження.

При розробці проекту була розкрита актуальність дослідження в обраному напрямку, поставлені цілі та завдання дослідження, визначені об'єкт та предмет дослідження, обґрунтовані основні проектні рішення та вказана його теоретична та практична значущість.

У першому розділі було проведено системний аналіз для визначення основних функціональних та нефункціональних вимог до аналізу власних досягнень. Далі в розділі розробки специфікації вимог до програмного забезпечення була сформульована постановка задачі та конкретизовані вимоги до функціональності вебзастосунку, його інтерфейсу та забезпечення безпеки.

У наступному розділі були розроблені проектні рішення, що забезпечують виконання специфікації вимог до програмного забезпечення. Описано структуру основних класових систем. Також була описана архітектура системи та взаємодія між її компонентами.

У останніх розділах була представлена виконана робота з кодування, тестування та внесення коректив до розробленого програмного забезпечення.

Була надана інформація про використані технології та інструменти розробки, а також процес тестування та виявлені проблеми під час розробки.

Кваліфікаційна робота бакалавра «Вебзастосунок аналізу власних досягнень» складається зі 88 сторінок, включаючи 28 ілюстрацій та 20 джерелв переліку посилань. У роботі було використано різноманітні джерела, включаючи наукові статті, книги та Інтернет-джерела. Процес розробки програмного забезпечення включав в себе використання таких технологій, як React. Робота містить детальні описи проектних рішень та класів системи, а також опис процесу тестування та виявлення проблем під час розробки.

Отже, робота містить детальний опис проекту інтернет-сервісу аналізу власних досягнень включаючи аналіз вимог, проєктування системи, розробку та тестування програмного забезпечення. Результатом роботи є функціональний та безпечний вебзастосунок.

Ключові слова: інтернет-сервіс, програмне забезпечення, аналіз досягнень.

ABSTRACT

to the qualification work of a bachelor

"Web application for personal achievements analysis"

Student 408 group: Beleviat Illia Viktorovych

Supervisor: Senior Lecturer Kirei Kateryna Oleksandrivna

The purpose of the work is to increase the efficiency of achieving personal goals through automation of the processes of analysis and selection of achievements in various spheres of life, development of an action plan for improving results in the future.

The object of research is the organization and management of personal achievements.

The subject of research is the method of organizing the management of personal achievements using web technologies.

The relevance of the research is substantiated and the problem, goal, and tasks of the project are formulated, the object and subject of the research are determined. The project's relevance in the chosen direction was revealed during its development, research goals and tasks were set, the object and subject of the research were determined, the main project decisions were substantiated, and its theoretical and practical significance was indicated.

In the first chapter, a systematic analysis was conducted to determine the main functional and non-functional requirements for analyzing personal achievements. Then, in the specification development chapter, the task was formulated and the requirements for the functionality of the web application, its interface, and security were specified.

In the next chapter, design decisions were developed to ensure compliance with the software requirements specification. The structure of the main class systems was described, as well as the architecture of the system and the interaction between its components.

Last chapters presented the work on coding, testing, and making corrections to the developed software. Information was provided on the technologies and

development tools used, as well as the testing process and problems encountered during development.

The Bachelor's thesis "Web Application for Personal Achievements Analysis" consists of 88 pages, including 28 figures and 20 sources. Various sources were used in the work, including scientific articles, books, and internet sources. The software development process included the use of technologies such as React. The work contains detailed descriptions of project decisions and system classes, as well as a description of the testing process and problem detection during development.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

UI – User Interface

SQL – Structured Query Language

БД – База даних

ПЗ – Програмне Забезпечення

API – Application Programming Interface

ВСТУП

Нині більшість людей прагнуть досягти успіху в різних сферах життя – навчанні, роботі, спорті, творчості тощо. Однак, не завжди вдається досягнути поставлених цілей, іноді навіть здається, що нічого не вдається досягти. Тому було вирішено провести аналіз власних досягнень, щоб з'ясувати, чому ми досягаємо успіху в деяких сферах життя, а в інших – ні. Це дослідження має на меті дослідити різні методики оцінки власних досягнень, з'ясувати, які фактори впливають на наші успіхи і невдачі та як можна використовувати цю інформацію для поліпшення своєї продуктивності. Для цього розглянуто різні аспекти аналізу власних досягнень, а також звернута увага на внутрішню мотивацію та важливість налагодження здорового співвідношення з успіхом та невдачами.

Сервіс корисний для будь-якої людини, яка прагне досягти успіху в житті та хоче зрозуміти, які кроки потрібно зробити для досягнення поставлених цілей. Було надано інструментарій, який допоможе знайти сильні та слабкі сторони власного досягнення, щоб зробити кроки для покращення своїх результатів.

Для розробки програмного забезпечення використано вихідні дані, такі як вимоги та специфікації користувачів, галузеві тенденції та дослідження цілей. Крім того, використано принципи та методології проєктування програмного забезпечення, щоб забезпечити надійність, масштабованість та зручність системи.

Об’єкт дослідження: організація та управління власними досягненнями.

Предмет дослідження: метод організації управління власними досягненнями з використанням вебтехнологій.

Мета роботи: підвищення ефективності досягнення особистої мети шляхом автоматизації процесів аналізу та відбору досягнень в різних сферах життя, розробки плану дій для покращення результатів в майбутньому.

Для досягнення цієї мети необхідно вирішити наступні завдання:

1.Провести аналіз сучасних автоматизованих систем та ресурсів орієнтованих на аналіз власних досягнень. Визначити їх сильні та слабкі сторони.

2.Специфікація вимог до програмного забезпечення інтернет-сервісу аналізу власних досягнень.

3.Проєктування та моделювання програмного забезпечення інтернет-сервісу аналізу власних досягнень.

4.Розробка функціональних модулів ПЗ та проєктування зручного та інтуїтивно зрозумілого інтерфейсу

4.1) Впровадити серверну частину застосунку для обробки запитів.

4.2) Впровадити систему облікових записів користувачів, яка дозволить користувачам керувати своїми аккаунтами.

4.3) Розробити систему управління та модерації контенту, щоб гарантувати, що він залишається безпечним та надійним для всіх користувачів.

5.Тестування та апробація ПЗ.

Потреба в цій роботі обґрунтована декількома причинами. По-перше, в сучасному світі успіх залежить не лише від знань та навичок людини, але і від здатності реалізовувати свій потенціал та досягати поставлених цілей. Аналіз власних досягнень дозволяє кожній людині правильно оцінити свої здібності, визначити свої сильні та слабкі сторони та знайти шляхи покращення. По-друге, в світі бізнесу та кар'єрного росту, аналіз власних досягнень допомагає людині визначити свої цілі та стати більш ефективним у досягненні саме цих цілей. Це допомагає збільшити рівень самооцінки та впевненості в собі, що є ключовим фактором для успіху. По-третє, аналіз власних досягнень може допомогти кожній людині знайти своє місце в житті та визначити, які кроки потрібно зробити для досягнення своєї певної гармонії в житті. Це може допомогти знайти більше задоволення від життя та збільшити рівень щастя.

Отже, вибір теми "аналіз власних досягнень" є досить обґрунтованим, оскільки актуальність аналізу власних досягнень полягає у впливі на особистий розвиток, самомотивації, виявленні сильних сторін та слабкостей, розвитку самосвідомості і плануванні майбутніх кроків. Це корисний інструмент для досягнення успіху у різних сферах життя.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ. СПЕЦИФІКАЦІЯ ВИМОГ

1.1 Опис предметної галузі

Розробка онлайн-сервісу аналізу власних досягнень - складне завдання, яке вимагає глибокого розуміння різних предметних областей. Успіх сервісу залежить від того, наскільки добре він спроектований, наскільки він простий у використанні і наскільки ефективно задовольняє потреби користувачів.

Одним з важливих аспектів розробки є розуміння користувацького досвіду. Добре продуманий інтерфейс може полегшити користувачам пошук потрібної інформації, тоді як погано розроблений інтерфейс може призвести до розчарування та відтоку користувачів[7].

Для того, щоб створити зручний і привабливий UI, UX-дизайнери повинні мати глибоке розуміння поведінки, вподобань і потреб користувачів. Вони повинні досліджувати та аналізувати дані користувачів, щоб отримати уявлення про те, що користувачі шукають і як вони взаємодіють з веб-сайтом.

Ще одним важливим аспектом є управління базами даних. Добре розроблена база даних може ефективно зберігати, організовувати та отримувати обсяги даних. База даних повинна підтримувати різні функції, такі як пошук і фільтрація, додавання , редагування та видалення постів, авторизацію користувачів[5].

Таким чином, розробка онлайн-сервісу аналізу власних досягнень вимагає глибокого розуміння дизайну користувацького досвіду, управління базами даних та управління контентом. Впровадивши ці елементи в дизайн, можна створити успішну платформу, яка задовольнить потреби користувачів.

1.2 Огляд та аналіз аналогів

На ринку є пропозиції щодо готових систем підтримки процесу досягнення мети. Найбільш цікавими з них є: Smart Progress, 24Goals, Habit list. Для кожного визначено наступні характеристики:

Назви: Smart Progress, 24Goals, Habit list.

Розробники: Smart Progress: розроблено компанією Smart Progress Inc, 24Goals: розроблено компанією 24Goals Inc, Habit list: розроблено компанією Habit list Inc.

Архітектура: Усі вони побудовані як трирівневі вебдодатки: зовнішній рівень, який обробляє користувацький інтерфейс і взаємодію з користувачем, середній рівень , який обробляє бізнес-логіку та дані, і внутрішній рівень, який обробляє зберігання та пошук даних. Така архітектура дає змогу вебсайту обробляти великий обсяг трафіку і забезпечує масштабованість.

Мова реалізації: Реалізовані з використанням комбінації технологій, включно з HTML, CSS, JavaScript і серверними мовами, як-от Java, C# або Python.

Перелік функцій:

Запис та аналіз власних досягнень.

Пошук і перегляд досягнень інших людей за місцезнаходженням, категорією та часом.

Можливість створювати та публікувати оголошення про свою ціль та кроки, які будуть зроблені щоб досягти її.

Зв'язуйтеся з іншими користувачами через вебсайт, щоб мати комунікацію з однодумцями.

Можливість фільтрувати цілі за різними критеріями.

Можливість зберігати пошукові запити та отримувати сповіщення про появу нових, що відповідають вашим критеріям.

Аналіз переваг та недоліків даного ПЗ:

Переваги:

Широко використовувана і добре налагоджена платформа для встановлення власної цілі.

Має велику базу користувачів та широкий вибір різних цілей

Має простий і зручний інтерфейс

Безкоштовне використання для більшості оголошень

Недоліки:

Обмежений функціонал пошуку.

Відсутність детальної інформації щодо досягнень інших людей.

Відсутність чіткої системи оцінювання

Недостатній контроль якості та достовірності інформації, яку розміщують користувачі на сайті

джерело інформації (вебсайт);

Smart Progress: https://smartprogress.do/

24Goals: https://24goalsapp.com/

Habit list: https://habitlist.com/

На рисунках 1.1 – 1.3 наведено головні сторінки аналогів.

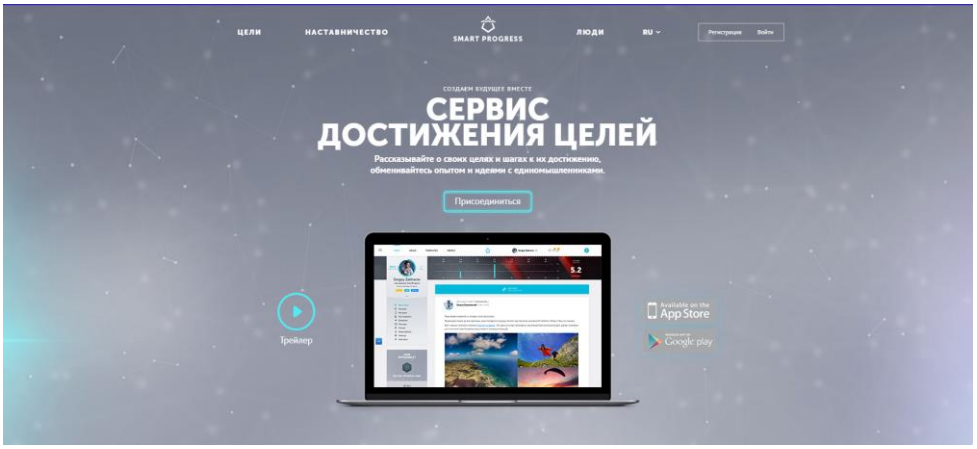

Рисунок 1.1 – Головна сторінка Smart Progress

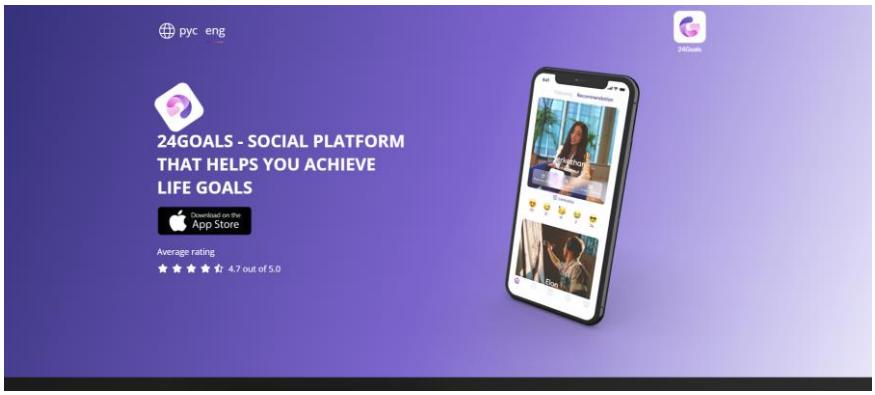

Рисунок 1.2 – Головна сторінка 24Goals

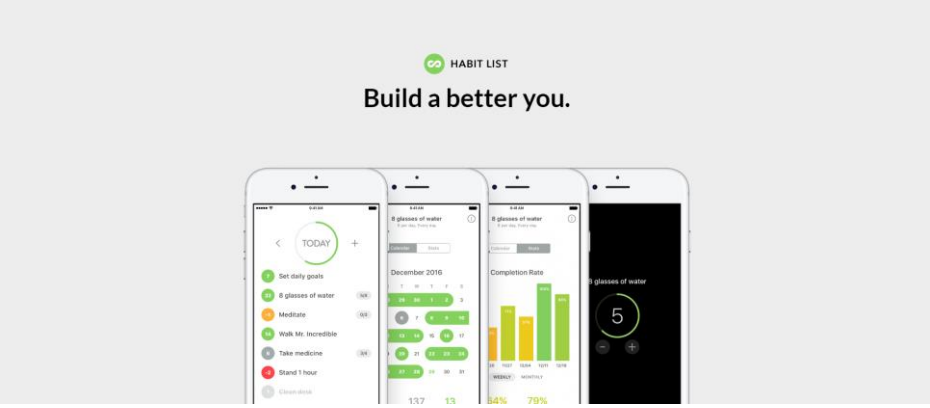

Рисунок 1.3 – Головна сторінка Habit list

Аналіз системи, що розробляється

- призначення ПЗ;

Вебсайт для аналізу власних досягнень – це онлайн-платформа, яка дозволяє користувачам розміщувати оголошення про свої досягнення та цілі, а іншим користувачі надає можливість залишати коментарі та поради. Ці вебсайти зазвичай містять такі функції, як пошукові фільтри, списки постів з детальною інформацією, а також контактні форми для зв'язку.

- опис структури системи (схема)

Структура системи інтернет-сервісу з аналізу власних досягнень, як правило, включає наступні компоненти:

Інтерфейс користувача: Це інтерфейс системи, який включає в себе вебсторінки та форми, з якими користувачі взаємодіють при користуванні сайтом.

База даних: Це внутрішня частина системи, яка зберігає інформацію про досягнення та користувачів, інші дані.

Функціонал пошуку та фільтрації: Цей компонент дозволить користувачам шукати досягнення інших користувачів за різними критеріями, такими як тип, час та популярність.

Система облікових записів користувачів: Цей компонент відповідатиме за створення та управління обліковими записами користувачів, а також дозволить користувачам переглядати свої цілі та керувати ними.

Внутрішня система: Вона відповідатиме за закулісні операції, такі як обробка високого трафіку, безпека та масштабованість.

Система управління контентом: Цей компонент відповідатиме за модерацію сайту та гарантуватиме, що він залишається безпечним і надійним для всіх користувачів.

- засоби апаратної та програмної реалізації

Апаратні та програмні засоби реалізації інтернет-магазину з купівлі-продажу транспортних засобів залежать від конкретних вимог проекту, але деякі популярні інструменти можуть бути використані:

React.Js: Бібліотека JavaScript для створення користувацьких інтерфейсів, React.Js зазвичай використовується для створення адаптивних, високопродуктивних вебзастосунках[9].

FireBase: Надає розробникам доступ до різноманітних інструментів, що допомагають в роботі з базами даних, аутентифікацією користувачів, зберіганням файлів, аналітикою додатків, пуш-сповіщеннями та багатьма іншими функціями. Firebase пропонує безкоштовні та платні плани, які залежать від використовуваних функцій та кількості користувачів додатку[10].

Node.js/Express: Node.js – це середовище виконання JavaScript, яке дозволяє розробникам створювати внутрішні сервіси за допомогою JavaScript, тоді як Express – це популярний фреймворк для вебзастосунків для Node.js. Разом вони можуть бути використані для створення потужного та ефективного бекенду для інтернет-сервісу[18].

1.3 Специфікація вимог

1 ПРИЗНАЧЕННЯ ТА МЕЖІ ПРОЄКТУ

1.1 Призначення системи, для якої розробляється програмне забезпечення.

Програмне забезпечення розробляється для системи аналізу власних досягнень.

1.2 Межі проєкту ПЗ

Межами програмного проєкту є створення зручного та ефективного вебзастосунку для аналізу власних досягнень.

2. ЗАГАЛЬНИЙ ОПИС

2.1 Сфера застосування

Додаток дозволить користувачам переглядати та шукати досягнення користувачів, додавати власні досягнення, переглядати інформацію про конкретні досягнення, залишати коментарі та керувати інформацією про свій обліковий запис.

2.2 Характеристики користувачів

Користувачами додатку будуть звичайні люди.

2.3 Загальна структура і склад системи

Система складатиметься з вебінтерфейсу користувача, бази даних для зберігання інформації про досягнення та користувачів, а також внутрішньої системи для обробки даних та зв'язку між інтерфейсом та базою даних.

2.4 Загальні обмеження

Система не буде включати в себе функції управління аккаунтами.

3 ФУНКЦІЇ СИСТЕМИ

3.2 Система облікових записів користувачів

3.2.1 Опис функції:

Система облікових записів користувачів дозволить користувачам створювати та отримувати доступ до системи управління досягненнями .

3.2.2 Вхідна та вихідна інформація:

Користувачі вводять інформацію про свій обліковий запис і отримують доступ для створення досягнень.

3.2.3 Функціональні вимоги:

Система повинна надійно зберігати та керувати інформацією про облікові записи користувачів, дозволяти легко створювати облікові записи та керувати ними.

3.1 Функції пошуку та фільтрації

3.1.1 Опис функції:

Функція пошуку та фільтрації дозволить користувачам шукати потрібні їм пости.

3.1.2 Вхідна та вихідна інформація:

Користувач вводить критерії пошуку та отримує список відповідних досягнень

3.1.3 Функціональні вимоги:

Система повинна забезпечувати точний пошук і фільтрацію бази даних з інформацією про досягнення користувачів, а також відображати результати пошуку у зрозумілій та зручній для користувача формі.

3.3 Система управління досягненнями

3.3.1 Опис функції

Система управління досягненнями дозволить користувачам створювати, редагувати, видаляти свої досягнення.

3.3.2 Вхідна та вихідна інформація:

Користувач додає інформацію про свої досягнення, данні пройшовши перевірку зберігаються у БД, пост публікується у загальному списку.

3.3.3 Функціональні вимоги:

Система повинна надійно зберігати та керувати інформацією про досягнення користувачів, дозволяти легко створювати пости та керувати ними, а також відображати у загальному списку та у власному списку користувача.

4 ВИМОГИ ДО ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Джерела та зміст вхідної інформації (даних):

Система буде спиратися на інформацію про досягнення, що вводиться користувачами.

4.2 Вимоги до методів організації, зберігання та підтримки інформації:

Система повинна мати безпечну та надійну базу даних для зберігання та організації інформації про досягнення та користувачів, з регулярним резервним копіюванням та процедурами технічного обслуговування.

5 ВИМОГИ ДО АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

Для роботи системи потрібні сервери з достатньою обчислювальною потужністю та обсягом пам'яті для обробки великих обсягів трафіку та зберігання даних.

6 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

6.1 Архітектура програмного забезпечення:

Система повинна мати масштабовану та ефективну програмну архітектуру з чітким розподілом між інтерфейсом користувача, базою даних та внутрішніми системами.

6.2 Програмне забезпечення системи:

Система повинна використовувати надійне та сучасне системне програмне забезпечення, з регулярними оновленнями та виправленнями безпеки.

6.3 Мережеве програмне забезпечення:

Система повинна використовувати безпечне та надійне мережеве програмне забезпечення для зв'язку між інтерфейсом користувача та внутрішніми системами.

6.4 Програмне забезпечення для ведення інформаційної бази:

Система повинна мати програмне забезпечення для управління та ведення бази даних досягнень та інформації про користувачів.

6.5 Мова та технологія розробки програмного забезпечення

Програмне забезпечення для вебзастосунку буде розроблено з використанням сучасного та надійного програмного стеку. Інтерфейс

користувача буде побудований з використанням React.js, популярної та ефективної бібліотеки JavaScript для побудови користувацьких інтерфейсів. Внутрішній сервер буде побудований з використанням Node.js та фреймворку Express, які забезпечують гнучку та масштабовану платформу для створення вебзастосунків. База даних була обрана FireBase, яка пропонує високу масштабованість та продуктивність.

ВИМОГИ ДО ЗОВНІШНІХ ІНТЕРФЕЙСІВ

7.1 Інтерфейс користувача

Інтерфейс користувача для вебзастосунку аналізу власних досягнень повинен бути інтуїтивно зрозумілим, зручним та швидким у користуванні. Він включатиме функцію пошуку та фільтрації, що дозволить користувачам знаходити потрібні їм пости. Інтерфейс також матиме систему облікових записів, яка дозволить користувачам створювати та керувати своїми обліковими записами, додавати досягнення та керувати ними. Користувацький інтерфейс буде доступний з настільних пристроїв і буде розроблений таким чином, щоб забезпечити однаковий користувацький досвід на всіх платформах.

7.2 Апаратний інтерфейс

Система вебзастосунку аналізу власних досягнень буде розроблена для роботи на стандартному апаратному забезпеченні та операційних системах. Користувачі не мають жодних специфічних вимог до апаратного забезпечення, оскільки система буде доступна через веббраузер на будь-якому пристрої з доступом до Інтернету.

7.3 Інтерфейс програмного забезпечення

Програмний інтерфейс системи буде заснований на RESTful API (інтерфейс прикладного програмування). Цей API дозволить користувацькому інтерфейсу взаємодіяти з внутрішнім сервером, обмінюючись даними та запитами. API буде побудовано з використанням Node.js та фреймворку Express, які забезпечують гнучку та масштабовану платформу для побудови вебінтерфейсів.

7.4 Протокол зв'язку

Вебзастосунок буде використовувати протоколи HTTP (Hypertext Transfer Protocol) та HTTPS (HTTP Secure) для забезпечення безпечного зв'язку між клієнтом та сервером. Протокол HTTPS буде використовуватися для шифрування всіх даних, що передаються через Інтернет, гарантуючи, що конфіденційна інформація, така як паролі користувачів, буде захищена від підслуховування та втручання.

8 ВЛАСТИВОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

8.1 Доступність

Будь-який користувач який забажає користуватись цим програмним застосунком зможе це зробити.

8.2 Супроводжуваність

Система буде розроблена таким чином, щоб її було легко обслуговувати та оновлювати. Кодова база буде добре задокументована та структурована, з чітким розподілом обов'язків між різними модулями та компонентами. Система буде спроектована таким чином, щоб забезпечити легке розгортання оновлень та нових функцій, не спричиняючи перебоїв у роботі користувачів або системи в цілому.

8.3 Переносимість

Система буде розроблена таким чином, щоб бути портативною, тобто її можна було розгортати на різних платформах та інфраструктурах без значних модифікацій. Система буде побудована з використанням відкритих стандартів і технологій, що гарантує її роботу на будь-якому обладнанні та операційній системі, які їх підтримують.

8.4 Продуктивність

Система буде розроблена з високою продуктивністю, що гарантує, що вона зможе обробляти великі обсяги трафіку та даних без уповільнення або збоїв. Система буде оптимізована для швидкості та ефективності, з

кешуванням та іншими методами, що використовуються для мінімізації часу відгуку на запити та інші операції.

8.5 Надійність

Надійність – це здатність системи виконувати свої функції послідовно і без помилок протягом тривалого часу. У контексті вебзастосунку аналізу власних досягнень надійність має вирішальне значення для того, щоб користувачі могли довіряти системі і покладатися на її точне виконання своїх функцій. Система повинна бути спроектована таким чином, щоб мінімізувати кількість помилок і забезпечити правильну реєстрацію.

Для забезпечення надійності в системі повинні бути передбачені заходи для коректної обробки помилок та винятків. Це включає надання користувачам інформативних повідомлень про помилки, реєстрацію помилок з метою налагодження та забезпечення механізмів автоматичного відновлення, де це можливо.

Крім того, система повинна бути спроектована таким чином, щоб справлятися з великим трафіком і масштабуватися для задоволення потреб зростаючої кількості користувачів. Цього можна досягти завдяки використанню хмарних хостингових платформ, балансувальників навантаження та інших заходів масштабування.

8.6 Безпека

Система повинна передбачати заходи для захисту даних користувачів, включаючи шифрування конфіденційних даних. Доступ до системи повинен контролюватися за допомогою механізмів автентифікації та авторизації користувачів, таких як ім'я користувача та пароль, контроль доступу на основі ролей.

Для запобігання таким атакам, як міжсайтовий скриптинг (XSS) ,система повинна бути розроблена з урахуванням найкращих практик безпеки, таких як перевірка та обробка вхідних даних, а також підготовлені твердження для запитів до бази даних.

Нарешті, система повинна регулярно тестуватися і перевірятися на наявність вразливостей безпеки, а виправлення та оновлення безпеки повинні застосовуватися оперативно, щоб забезпечити захист системи від нових загроз.

Висновки до розділу 1

Отже, в першому розділі проведено аналіз предметної сфери розробки програмного забезпечення для вебзастосунку аналізу власних досягнень. Дослідження дозволило визначити позитивні та негативні сторони різних аналогів та проаналізувати їх ефективність у певних сценаріях використання.

З метою розробки програмного забезпечення для інтернет-сервісу аналізу власних досягнень була розроблена специфікація вимог, яка містить ключові особливості та функціональні можливості, які повинна мати система. Такі вимоги дозволяють визначити напрямок процесу розробки та забезпечують задоволення потреб користувачів.

2 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-СЕРВІСУ АНАЛІЗУ ВЛАСНИХ ДОСЯГНЕНЬ

2.1 Діаграма прецедентів

Діаграма використання - це вид діаграми в UML (Unified Modeling Language), який показує взаємодію між системою та її користувачами або іншими системами, що взаємодіють з нею. Ця діаграма надає графічне зображення того, як користувачі використовують функціональні можливості системи або як система взаємодіє з іншими системами.

Діаграма використання складається з акторів та їх взаємодій з системою. Актор - це роль, яку відіграє людина або система в процесі взаємодії з системою, наприклад, користувач, клієнт, інший комп'ютер або інша система. Взаємодія відображається у вигляді стрілок, які вказують на те, які дії виконуються між акторами та системою.

Діаграма використання може бути використана для аналізу потреб користувачів та визначення функціональних можливостей системи. Вона також може допомогти у розробці системи, оскільки вона надає зрозуміле графічне зображення того, як система взаємодіє з користувачами та іншими системами. Діаграма використання є важливим інструментом у процесі аналізу та розробки програмного забезпечення, тому вона широко використовується в сфері інформаційних технологій.

Складено глосарій проєкту (див. Таблицю-2.1)

Таблиця 2.1 – глосарій проєкту

Адміністратор (Administrator)	Спеціаліст, який контролює роботу системи
Користувач (AnyUser)	Покупець, продавець або адміністратор якому потрібно авторизуватися у системі.
Модератор(Moderator)	Спеціаліст, який керує списками користувачів

Кінець таблиці 2.1

Загальний список постів (General list of posts)	Список усіх постів що є на сайті.
Пост (post)	Сторінка, що містить конкретний опис досягнення та хто його зробив.
Фільтр (filter)	Інструмент що надає можливість знайти пост.

На рисунку 2.1 наведено діаграму використання застосунку.

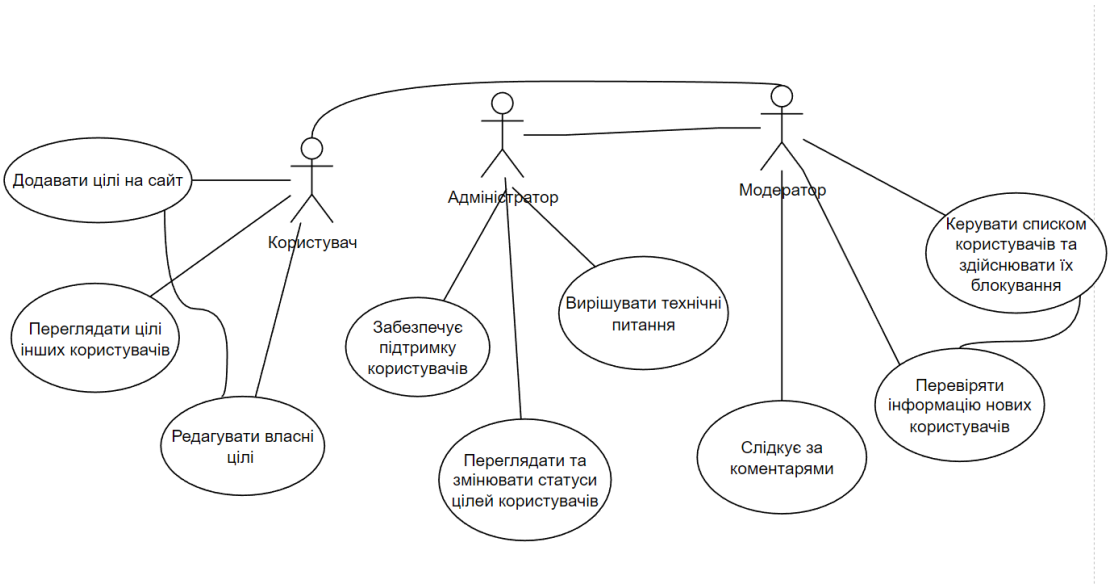

Рисунок 2.1 – Діаграма варіантів використання

Опис дійових осіб

Користувач (AnyUser) – входить в систему (ідентифікація)

Модератор(Moderator) – керує списками користувачів, здійснює блокування.

Адміністратор (Administrator) – має доступ до всіх оголошень та аккаунтів користувачів, може вирішити технічні питання.

Варіант використання "Вхід в систему"

Передумови: відсутні.

Пост умови: якщо варіант використання виконаний успішно, користувач входить в систему. В іншому випадку стан системи не змінюється.

На таблицях 2.2 – 2.4 наведено перебіг варіанту використання “ Вхід в систему’’.

Таблиця 2.2 – Головний розділ сценарію виконання варіанта використання "Вхід в систему"

Варіант використання	Вхід в систему
Актори	Користувач
Короткий опис	Описує вхід користувача в систему управління досягненнями
Мета	Увійти до системи
Тип	Базовий
Посилання на інші варіанти використання	

Таблиця 2.3 – Типовий хід подій сценарію виконання варіанта використання «Вхід в систему»

Дії актора	**Відгук системи**
1. Користувач хоче увійти в систему	2. Система запитує ім'я користувача реєстрації курсів
3. Користувач вводить ім'я і пароль Виняток №1. Користувач вводить невірне ім'я або пароль	

Таблиця 2.4 – Винятки сценарію виконання варіанта використання "Вхід в систему"

Дії актора	**Відгук системи**
Виняток №1. Користувач вводить невірне ім'я або пароль	

Кінець таблиці 2.4

	4. Система відображає інформацію про введення неправильного імені та / або пароля.
5. Клієнт вводить ім'я і пароль заново або відмовляється від входу в систему	

Варіант використання "Управління досягненнями"

Передумови: користувач повинен бути зареєстрований і володіти необхідними правами.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені продавцем, будуть збережені системою.

На таблицях 2.5 – 2.6 наведено перебіг варіанту використання “ Управління досягненнями’’.

Таблиця 2.5 – Головний розділ сценарію виконання варіанта використання «Управління досягненнями»

Варіант використання	Управління оголошеннями
Актори	Користувач
Короткий опис	Додавання, редагування та видалення посту
Мета	Додати/редагувати/видалити досягнення
Тип	Базовий
Посилання на інші варіанти використання	Включає в себе варіанти: • додавання досягнення • редагування досягнення • видалення досягнення

Таблиця 2.6 – Типовий хід подій сценарію використання «Управління досягненнями»

Дії актора	**Відгук системи**
1. Користувач хоче почати роботу з досягненнями	2. Система запрошує необхідну дію (створити досягнення, відредагувати існуюче, видалити)
3. Продавець обирає дію.	4. Система виконує одну дію з переліку.

Варіант використання "Створення досягнення "

Передумови: користувач вибрав дію.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені користувачем, будуть збережені системою. В іншому випадку стан системи не змінюється.

На таблицях 2.7 – 2.9 наведено перебіг варіанту використання “ Створення оголошення’’.

Таблиця 2.7 – Головний розділ сценарію виконання варіанта використання «Створення досягнення»

Варіант використання	Створення досягнення
Актори	Користувач
Короткий опис	Створення користувачем досягнення
Мета	Створити пост
Тип	Підлеглий
Посилання на інші варіанти використання	Виконується в рамках варіанту використання «Управління досягненнями»

Таблиця 2.8 – Типовий хід подій сценарію виконання варіанта використання «Створення досягнення»

Дії актора	**Відгук системи**
3. Користувач обирає дію	4. Система виводить форму для створення досягнення
5. Користувач заповнює усі необхідні поля 6. Користувач підтверджує зберігання. Виняток №1. Користувач помилився при введені інформації	7. Система створює пост і зберігає його.

Таблиця 2.9 – Винятки сценарію виконання варіанта використання «Створення досягнення»

Дії актора	Відгук системи
Виняток №1. Користувач помилився при введені інформації	
6. Користувач підтверджує зберігання.	7. Система попереджає про помилку, збереження не відбувається.
8. Користувач повертається до пункту 5 типового ходу подій	

Варіант використання "Редагування досягнення"

Передумови: користувач вибрав дію.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені користувачем, будуть збережені системою. В іншому випадку стан системи не змінюється.

На таблицях 2.10 – 2.12 наведено перебіг варіанту використання “Редагування досягнення’’.

Таблиця 2.10 – Головний розділ сценарію виконання варіанта використання «Редагування досягнення»

Варіант використання	Редагування досягнення
Актори	Користувач
Короткий опис	Редагування користувачем досягнення
Мета	Редагування пост
Тип	Підлеглий
Посилання на інші варіанти використання	Виконується в рамках варіанту використання «Управління досягненнями»

Таблиця 2.11 – Типовий хід подій сценарію виконання варіанта використання «Редагування досягнення»

Дії актора	**Відгук системи**
3.Користувач обирає дію	4. Система відображає список постів користувача
5.Обирає потрібний пост для редагування.	6.Система відображає форму в якій дані що готові до редагування.
7. Користувач змінив інформацію 8. Підтвердив зміни Виняток №1 Користувач ввів дані що не підлягають валідації	9. Успішно зберігло дані.

Таблиця 2.12 – Винятки сценарію виконання варіанта використання "Редагування тексту"

Дії актора	**Відгук системи**
Виняток №1 Користувач ввів дані що не підлягають валідації	

Кінець таблиці 2.12

7. Користувач змінив інформацію 8. Підтвердив зміни	9. Система повідомляє про помилку
10. Користувач повертається до пункту 7	

Варіант використання "Видалення досягнення"

Передумови: користувач вибрав дію.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені користувачем, будуть збережені системою. В іншому випадку стан системи не змінюється.

На таблицях 2.13 – 2.15 наведено перебіг варіанту використання “Видалення досягнення.

Таблиця 2.13 – Головний розділ сценарію виконання варіанта використання «Редагування досягнення»

Варіант використання	Видалення досягнення
Актори	Користувач
Короткий опис	Видалення користувачем досягнення
Мета	Видалити пост
Тип	Підлеглий
Посилання на інші варіанти використання	Виконується в рамках варіанту використання «Управління досягненнями»

Таблиця 2.14 – Типовий хід подій сценарію виконання варіанта використання «Видалення досягнення»

Дії актора	Відгук системи
3. Користувач обирає дію Виняток №1 У цього користувача немає постів	4. Система відображає список постів користувача
5.Обирає потрібний пост для видалення.	6.Система запрошує підтвердження.

Таблиця 2.15 – Винятки сценарію виконання варіанта використання « Видалення досягнення»

Дії актора	Відгук системи
Виняток №1 У цього користувача немає оголошень	
3. Користувач обирає дію	4. Система відображає пустий список досягнень
5. Користувач повертається до 3 пункту	

Варіант використання «Фільтрування списку досягнень»

Передумови: відсутні.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені користувачем, будуть відображені системою. В іншому випадку стан системи не змінюється.

На таблицях 2.16 – 2.18 наведено перебіг варіанту використання “ Фільтрування списку досягнень.

Таблиця 2.16 – Головний розділ сценарію виконання варіанта використання «Фільтрування списку досягнень»

Варіант використання	Фільтрування списку досягнень
Актори	Користувач

Кінець таблиці 2.16

Короткий опис	Фільтрування користувачем постів
Мета	Відфільтрувати досягнення
Тип	Базовий

Таблиця 2.17 – Типовий хід подій сценарію виконання варіанта використання «Фільтрування списку досягнень»

Дії актора	**Відгук системи**
1.Покупець обирає головний список досягнень	2.Ситема відображає усе наявне
3. Відкриває фільтр та вносить критерії Виняток №1 Вказує модель якої немає на сайті	4. Відображає список згідно критеріїв.

Таблиця 2.18 – Винятки сценарію виконання варіанта використання " Фільтрування списку досягнень"

Дії актора	**Відгук системи**
Виняток №1 Вказує критерій якого немає на сайті	
4. Відкриває фільтр та вносить критерії	5. Система повідомляє що такого критерію немає.
6. Користувач повертається до 3 пункту	

Варіант використання «Видалення користувача та його даних»

Передумови: користувач ідентифікований як адміністратор.

Пост умови: якщо варіант використання виконаний успішно, зміни, внесені адміністратором, будуть відображені системою. В іншому випадку стан системи не змінюється.

На таблицях 2.19 – 2.21 наведено перебіг варіанту використання “Видалення користувача та його даних’’.

Таблиця 2.19 – Головний розділ сценарію виконання варіанта використання «Видалення користувача та його даних»

Варіант використання	**Фільтрування списку авто**
Актори	Адміністратор
Короткий опис	Видалення адміністратором користувача
Мета	Видалити дані про користувача
Тип	Базовий
Посилання на інші варіанти використання	

Таблиця 2.20 – Типовий хід подій сценарію виконання варіанта використання «Видалення користувача та його даних»

Дії актора	**Відгук системи**
1. Адмін відкриває панель адміністрування користувачами	2.Система відображає список користувачів
3. Адмін обирає дію	4. Система відображає кнопки видалити та блокування користувача
5.Обирає видалити користувача та його оголошення Виняток №1 Адміністратор помиляється та блокує користувача замість видалення	6. Система запитує підтвердження
7. Підтверджує дії	8. Успішно видаляє дані

Таблиця 2.21 – Винятки сценарію виконання варіанта використання " Видалення користувача та його даних "

Дії актора	**Відгук системи**
Виняток №1 Адміністратор помиляється та блокує користувача замість видалення	
5.Обирає заблокувати користувача	6 Система запитує підтвердження та повідомляє про дію що зробив адміністратор.
7. Адміністратор помічає це та повертається до пункту 3.	

Варіанти використання системи дали розуміння того як саме система відповідає на дії користувача та які головні сценарії має.

2.2 Проєктування інтерфейсу веб застосунку

Успіх у веб-застосунку залежить від багатьох факторів, а одним з найважливіших є користувачі. Якщо веб-застосунок не зручний та не привабливий для користувачів, вони можуть швидко втратити інтерес і перейти до конкурентів. Тому важливо забезпечити, щоб інтерфейс був інтуїтивно зрозумілим та легким у використанні.

Розробка відповідного інтерфейсу веб-застосунку вимагає ретельного планування та дослідження. Розробники повинні враховувати потреби користувачів, щоб створити інтерфейс, який буде максимально корисним та зручним.

Крім того, інтерфейс повинен бути адаптивним до різних пристроїв та розмірів екранів, оскільки користувачі можуть використовувати веб-застосунок на різних пристроях, включаючи смартфони, планшети та комп'ютери.

Добре розроблений інтерфейс також може допомогти забезпечити безпеку веб-застосунку, наприклад, за допомогою простого та безпечного інтерфейсу авторизації користувачів. Це дозволяє зменшити ризик несанкціонованого доступу та інших кібератак.

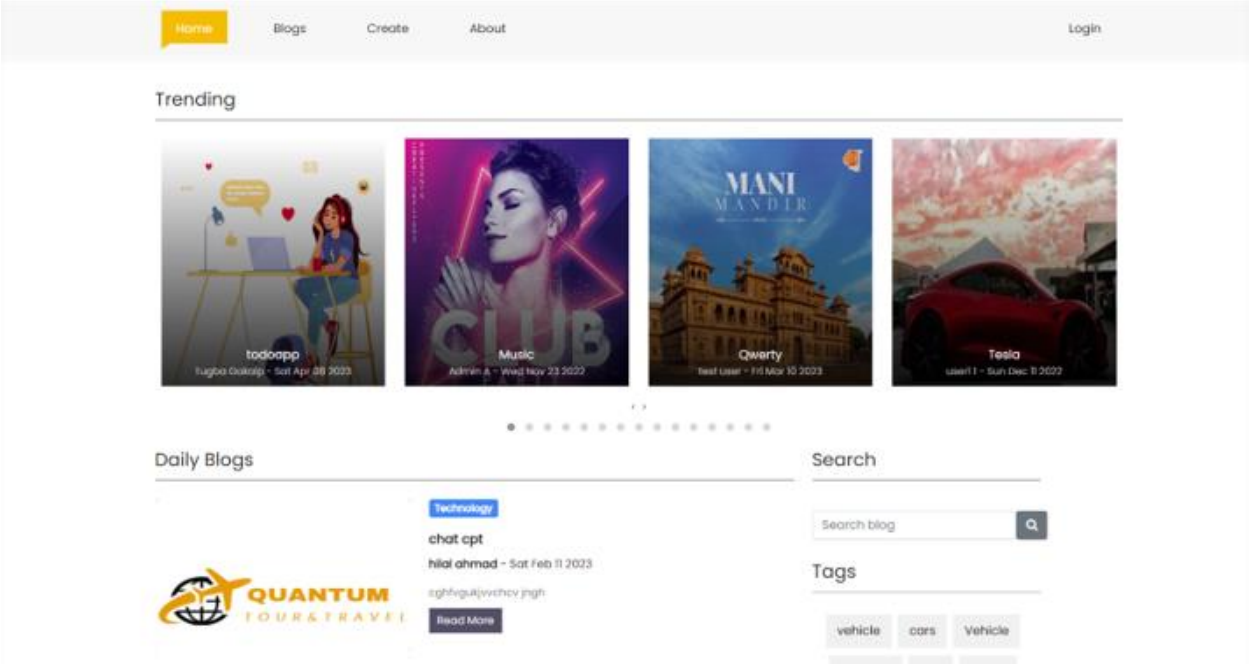

Рисунок 2.2 – Домашня сторінка

Ця домашня сторінка (див. рисунок 2.2) здається дуже зручною та легкою в користуванні. Навігаційна панель містить всі необхідні розділи сайту, що допомагає користувачеві швидко знайти потрібну інформацію. У тілі сайту є різні інформаційні блоки, які доповнюють загальний зміст сторінки. Ці блоки містять інформацію про головні позиції сайту, мету сайту та основну інформацію для користувача.

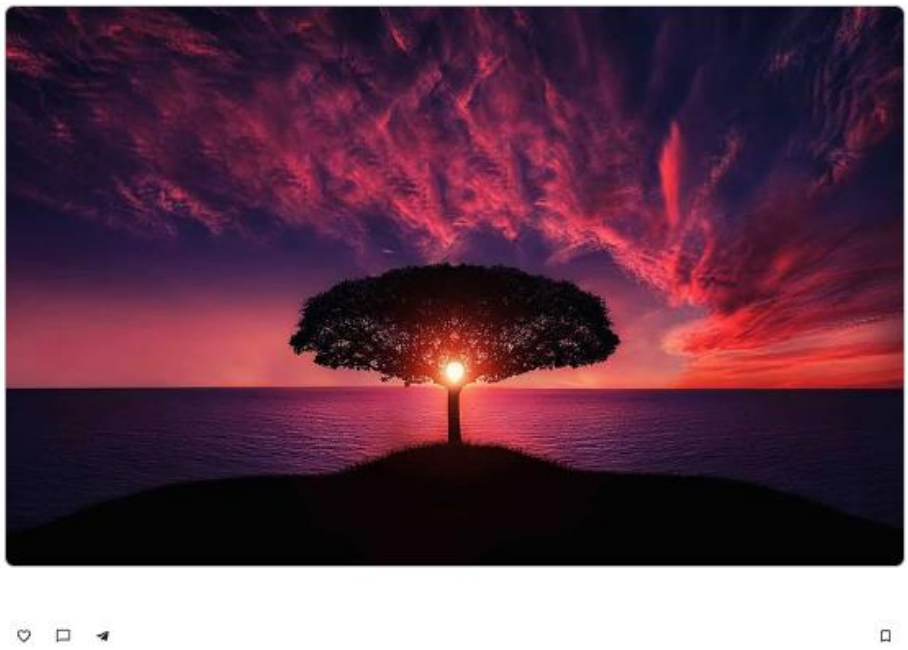

Рисунок 2.3 – Сторінка додавання посту

На сторінці додавання посту (див. рисунок 2.3) зображено пост з фотографією. У нижній частині відображений додатковий функціонал, де користувачі можуть залишити коментарі або оцінити пост. Сторінка створення постів є дуже важливою, оскільки саме на цій сторінці користувач може створити свій пост з досягненнями на сайті. Якщо дана сторінка розроблена правильно і користувач зможе легко розібратися як додавати власні пости, то це збільшить кількість постів, що розміщуються на сайті.

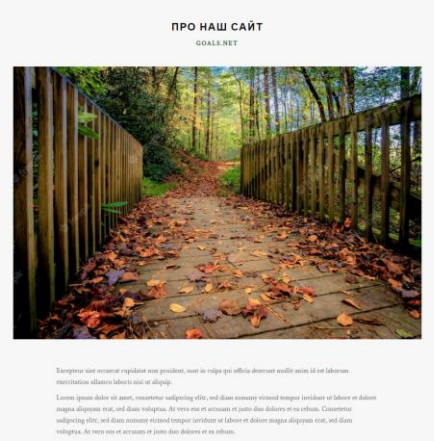

Рисунок 2.4 – Сторінка про сайт

На сторінці про сайт (див. рисунок 2.4) розташована основна інформація про розробників сайту та його основна ціль. Також було створено форму зворотного зв’язку, щоб користувачі могли запитати необхідну їм інформацію у адміністратора.

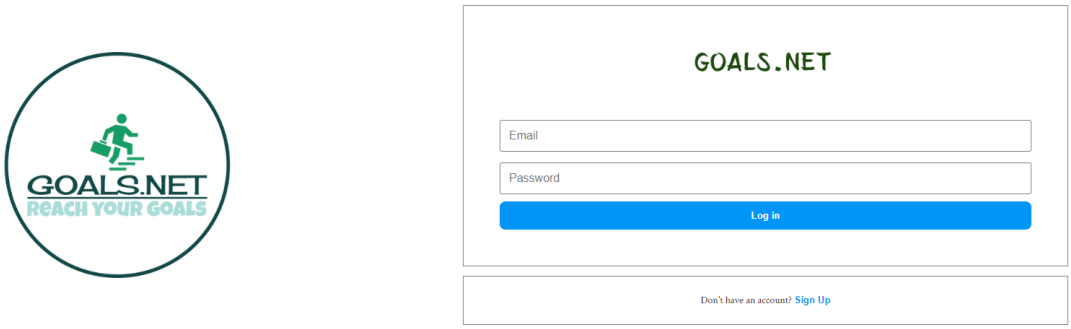

Рисунок 2.5 – Логін форма

На логін формі (див. рисунок 2.5) розташовані поля для вводу електронної пошти та паролю. Також були зроблені допоміжні інструменти такі як “ Sign Up” що зареєструє користувача у разі потреби. У низу форми кнопка під твердження.

2.3 Візуальна карта веб застосунку

Візуальна карта веб-застосунку є інструментом, який дозволяє візуалізувати архітектуру та функціональність веб-застосунку. Вона зображає структуру веб-застосунку та взаємозв'язки між різними його елементами.

Візуальна карта допомагає розробникам та дизайнерам краще зрозуміти структуру веб-застосунку та взаємозв'язки між його елементами. Це дозволяє створювати більш логічну та просту структуру, яка покращує користувацький досвід та зручність використання веб-застосунку. На рисунку 2.6 наведено візуальну карту веб застосунку.

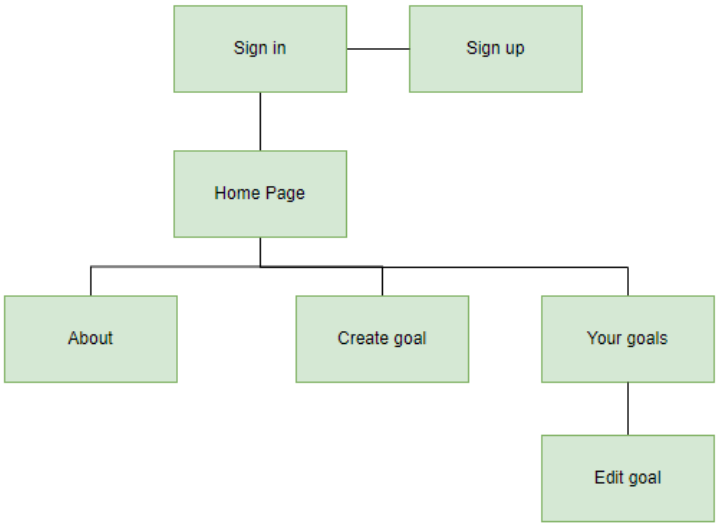

Рисунок 2.6 – Візуальна карта веб застосунку

На візуальній карті зображено переходи між сторінками на веб сайті. Так користувач може зайти на головну сторінку лише за умови, що він увійшов до свого аккаунту або зареєстрував новий. Після входу він може користуватись повним функціоналом сайту.

2.4 Проєктування бази даних

Розробляючи онлайн-сервіс аналізу власних досягнень, стало зрозуміло, що база даних є критично важливим компонентом, який зберігатиме та отримуватиме дані про пости та користувачів, а також полегшуватиме взаємозв'язок між ними. Щоб гарантувати, що база даних буде ефективною та безпечною для зберігання та пошуку даних, дотримано суворого процесу проєктування бази даних[5].

По-перше, визначено об'єкти, про які потрібно було зберігати дані. Основними сутностями були пости та користувачі. Для постів визначено такі атрибути, як назва, досягнення, основна ціль, картинка. Для користувачів визначено такі атрибути, як ім'я, адреса електронної пошти, та пароль.

Далі обрано Firebase як систему управління базами даних, оскільки вона дозволяє розробнику не відволікатися на створення backend. І це спрощує і прискорює створення додатків, дає можливість повністю зосередитися саме на інтерфейсі і досвіді[10].

Для постів створено колекцію " Firestore", яка зберігала документи з такими атрибутами, як назва, досягнення, основна ціль, картинка. Для користувачів створено колекцію "Authentication", в якій зберігалися документи з такими атрибутами, як ім'я, адреса електронної пошти та пароль.

На рисунку 2.8 зображено створену базу даних Firebase.

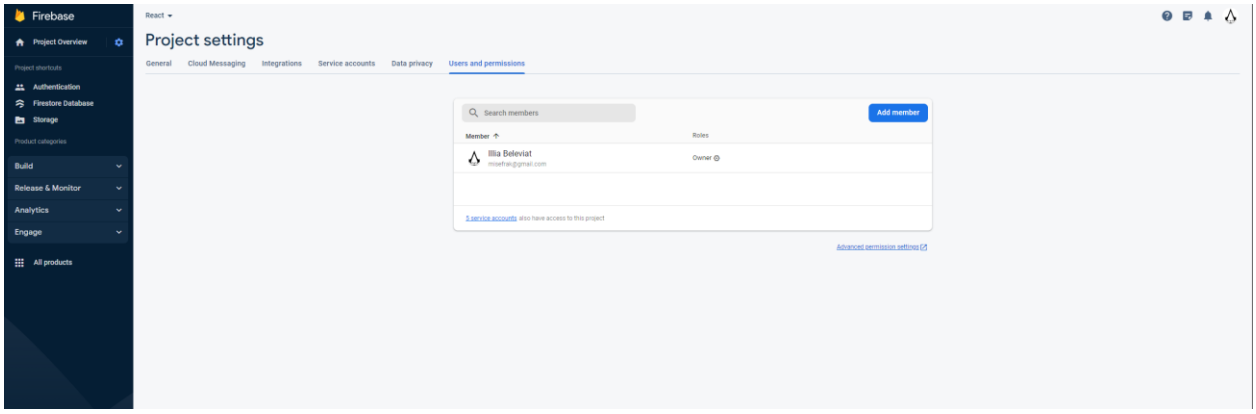

Рисунок 2.8 – база даних Firebase.

Насамкінець, процес проєктування бази даних мав вирішальне значення для успіху онлайн-сервісу аналізу власних досягнень. Використовуючи Firebase і дотримуючись суворого процесу, спроектовано базу даних, яка була ефективною та безпечною.

Висновки до розділу 2

У цьому розділі було проведено аналіз потреб користувачів та складено Use case для кожного з акторів, включаючи їх дії на сайті. На основі цього були створені макети вебсайту та карта сайту, що допомагають візуалізувати структуру сайту та навігацію для користувачів. Також, було описано проєктування бази даних для зберігання інформації про користувачів та їх дії на сайті.

За допомогою цих етапів, визначено потреби користувачів та їх очікування від сайту, а також була створена ефективна структура сайту та бази даних, щоб забезпечує зручний та логічний досвід користувачів. Це допоможе підвищити задоволеність користувачів та покращити ефективність сайту.

3 ВИБІР ЗАСОБІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНТЕРНЕТ-СЕРВІСУ АНАЛІЗУ ВЛАСНИХ ДОСЯГНЕНЬ

3.1 Вибір технологій розробки клієнтської частини вебзастосунку

Вибір технологій для розробки клієнтської частини веб-застосунку є критичним етапом в процесі створення сучасного програмного рішення. Клієнтська частина веб-застосунку є тим, що бачить і взаємодіє з користувачем, тому вона відіграє важливу роль у створенні позитивного користувацького досвіду та задоволення потреб клієнтів.

Залежно від потреб і обмежень проекту, можна мати різноманітні вимоги до клієнтської частини веб-застосунку. Це можуть бути вимоги до швидкості та продуктивності, розширюваності та підтримки, кросплатформенності або можливості розробки мобільних додатків.

Технологічний стек для розробки клієнтської частини веб-застосунку постійно розширюється та розвивається, і є багато варіантів для вибору. Важливо ретельно проаналізувати свої потреби, врахувати обмеження проекту та визначити найкращі технології, які відповідають загальним вимогам.

Надалі буде розглянуто декілька популярних технологій, які можуть бути використані для розробки клієнтської частини веб-застосунків. Буде проаналізовано їх переваги, обмеження та випадки використання. Це допоможе зробити осмислений вибір і забезпечити успіх проекту.

На рисунку 3.1 наведено статистику популярності фроненд-фреймворків.

Рисунок 3.1 – Найпопулярніші фронтенд-фреймворки.

Вибір правильних технологій для розробки клієнтської частини веб-застосунку є ключовим етапом у створенні успішного та конкурентоспроможного програмного рішення. Клієнтська частина є основним інтерфейсом взаємодії з користувачем, тому правильний вибір технологій гарантує позитивний користувацький досвід та задоволення потреб клієнтів.

Вибір відповідних технологій включає ретельний аналіз вимог проекту, розгляд можливостей та обмежень. Різні технології мають свої переваги та особливості, які можуть впливати на швидкість, продуктивність, масштабованість та зручність розробки. Вибір правильних технологій дозволяє ефективно втілити функціональність, забезпечити гнучкість та майбутню розширюваність веб-застосунку.

Додатково, важливо врахувати тренди та інновації у сфері веб-розробки, оскільки технологічний ландшафт постійно змінюється. Правильний вибір технологій забезпечує конкурентні переваги, дозволяє створювати сучасні та функціональні веб-застосунки, які задовольняють потреби користувачів і прогресують разом з технологічним розвитком.

Перелік доступних технологій розробки клієнтської частини вебзастосунку дуже широкий, проте серед найпопулярніших можна виділити такі фреймворки та бібліотеки.

React є одним з найпопулярніших фронтенд фреймворків, який широко використовується для розробки сучасних веб-застосунків. Він вирізняється своєю компонентною архітектурою та високою продуктивністю, що дозволяє розробникам будувати складні та інтерактивні інтерфейси користувача[2].

React був розроблений компанією Facebook і вперше випущений у 2013 році. Він здобув широку популярність серед розробників завдяки своїй простоті, швидкості та гнучкості. React використовує JSX (розширений

синтаксис JavaScript), який дозволяє розробникам комбінувати HTML-подібний код з JavaScript для створення компонентів.

Одним з ключових принципів React є "однокерневий" потік даних, де зміни у стані компонентів призводять до оновлення відображення на екрані. Це дозволяє легше відслідковувати та керувати станом додатка. Крім того, React пропонує використання віртуального DOM (Document Object Model), що сприяє ефективній реактивності та оновленню інтерфейсу при зміні даних[8].

React також має велику спільноту розробників, що сприяє наявності багатьох готових бібліотек, компонентів та інструментів розробки. Це дозволяє розробникам швидше будувати функціональність інтерфейсу та скорочує час розробки[15].

Усе це робить React потужним інструментом для створення веб-застосунків різної складності, від простих статичних сайтів до складних односторінкових додатків з багатим функціоналом.

На рисунку 3.2 наведено головну сторінку React.

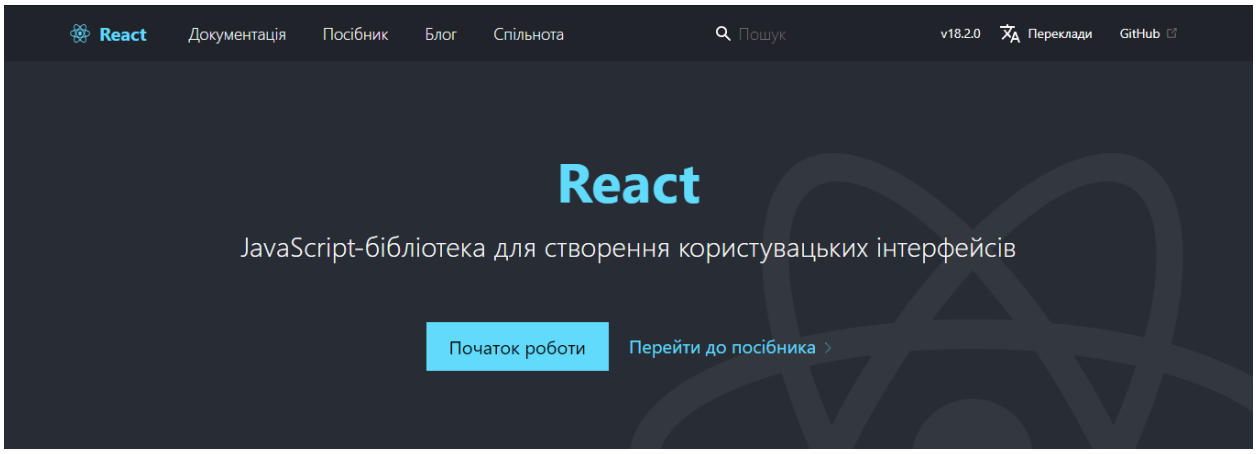

Рисунок 3.2 – Головна сторінка React

Vue.js є одним з найпопулярніших фронтенд фреймворків, який набуває все більшої популярності серед розробників. Він пропонує простоту використання та гнучкість для створення сучасних веб-застосунків.

Vue.js був розроблений Еваном Ю і перший раз випущений у 2014 році. Його основна ідея полягає в тому, щоб поєднати найкращі аспекти інших

фреймворків, таких як React та Angular, і створити простий, але потужний інструмент для розробки.

Одним з ключових понять Vue.js є "реактивність". Він використовує спеціальний об'єкт "Vue" для слідкування за змінами даних та автоматичного оновлення відображення. Це дозволяє забезпечити швидку та ефективну відповідь на зміни даних користувача.

Vue.js також пропонує компонентну архітектуру, яка дозволяє розбити веб-застосунок на незалежні компоненти. Це спрощує розробку та підтримку коду, оскільки компоненти можна повторно використовувати та розширювати. Крім того, Vue.js має привабливу документацію та широку спільноту розробників, що сприяє доступності ресурсів та підтримці для розробників.

Завдяки своїй простоті та легкості вивчення, Vue.js підходить як для початківців у веб-розробці, так і для досвідчених розробників. Він забезпечує зручну роботу з даними та відображенням, а також широкий набір додаткових можливостей, таких як маршрутизація, керування станом та анімації.

Vue.js продовжує зростати у популярності і стає все більш привабливим вибором для розробників, які шукають ефективний та простий у використанні фронтенд фреймворк для своїх проектів.

На рисунку 3.3 наведено головну сторінку Vue.js.

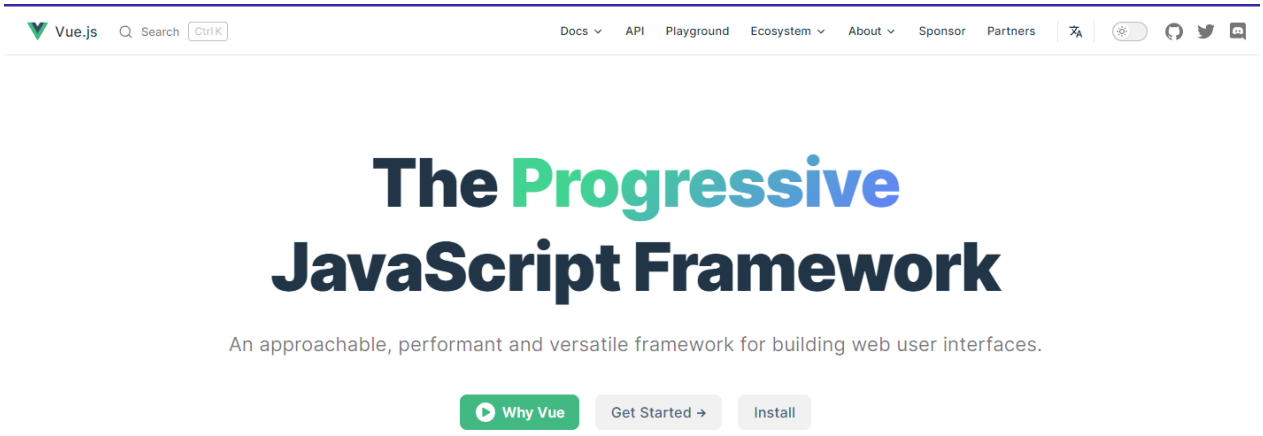

Рисунок 3.3 – Головна сторінка Vue.js

Angular є одним з провідних фронтенд фреймворків, який надає повноцінні можливості для розробки веб-застосунків. Він визначається своєю структурованістю, потужними інструментами та готовими рішеннями для складних проектів.

Розроблений компанією Google, Angular з'явився на світ у 2010 році під назвою AngularJS і отримав широку популярність. Згодом була представлена нова версія - Angular (без суфікса "JS"), яка має значні вдосконалення та покращену продуктивність.

Angular пропонує повністю розширюваний і прогресивний фреймворк для створення веб-застосунків. Він базується на TypeScript, мові програмування, яка надає статичну типізацію та більшу безпеку кодування. Angular також пропонує широкі можливості маршрутизації, обробки подій, управління станом та взаємодії з сервером.

Одним з ключових аспектів Angular є його модульна структура, яка дозволяє розбити додаток на компоненти, сервіси та модулі. Це спрощує розробку, підтримку та тестування коду, а також забезпечує повторне використання компонентів. Angular також надає потужні інструменти для створення складних форм, валідації та обробки даних.

За допомогою Angular, розробники можуть будувати розширені веб-застосунки з багатофункціональним інтерфейсом та високою продуктивністю. Він також має велику спільноту розробників, що сприяє доступності документації, плагінів та інструментів розробки.

Angular підходить для великих та складних проектів, де необхідно використовувати широкий спектр функціональності та забезпечити швидку та ефективну розробку. Він є рішенням для команд розробників, які шукають розширюваність, масштабованість та гнучкість.

На рисунку 3.4 наведено головну сторінку Angular.

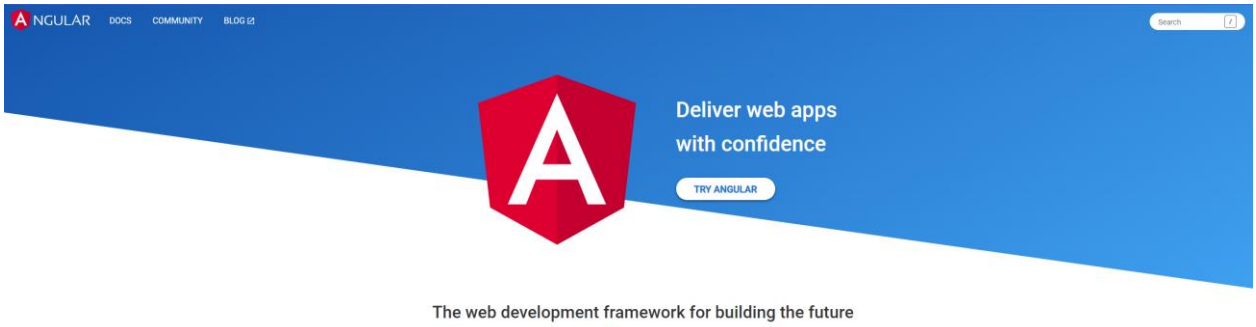

Рисунок 3.4 – Головна сторінка Angular

Розглянемо переваги та недоліки трьох популярних фронтенд фреймворків: React, Vue.js та Angular.

React:

Переваги:

1) Висока швидкість та продуктивність завдяки використанню віртуального DOM.
2) Масштабованість та гнучкість у розробці складних додатків.
3) Широка спільнота розробників та багато готових компонентів і бібліотек.

Недоліки:

1) Вимагає додаткових бібліотек та інструментів для реалізації повноцінного фронтенд стеку.
2) Висока крутизна навчання для початківців.
3) Вимагає багато ручного коду для реалізації складних функцій.

Vue.js:

Переваги:

1) Простота вивчення та використання, особливо для початківців.
2) Реактивність даних, що полегшує керування станом додатку.
3) Широкий вибір готових рішень та додаткових бібліотек.

Недоліки:

1) Обмежені можливості для розробки великих та складних проектів порівняно з Angular.

2) Менша спільнота розробників порівняно з React та Angular.
3) Відсутність широкої підтримки великих корпорацій порівняно з Angular.

Angular:

Переваги:

1) Повноцінний фреймворк з готовими рішеннями для розробки великих та складних додатків.
2) Структурованість та модульність, що полегшує розробку та підтримку коду.
3) Велика спільнота розробників та підтримка від Google.

Недоліки:

1) Висока крутизна навчання для початківців та більша складність у порівнянні з React та Vue.js.
2) Великий розмір фреймворку, що може впливати на швидкість завантаження додатку.
3) Потребує більше обов'язкових вимог, таких як використання TypeScript та строга структура проекту.

Варто зазначити, що переваги та недоліки можуть бути контекстуальними, залежно від потреб проекту та рівня володіння розробником конкретним фреймворком.

3.2 Вибір бази даних для вебзастосунку

Вибір бази даних є одним із важливих етапів при розробці веб-застосунків. База даних відіграє ключову роль у зберіганні, організації та отриманні даних, які використовуються в додатку. Це може бути структурована реляційна база даних або нереляційна база даних, така як документ-орієнтована, стовпцева, ключ-значення або графова база даних[5].

Вибір правильної бази даних залежить від конкретних потреб проекту, обсягу даних, вимог до швидкодії, доступності, масштабованості та інших факторів. Кожен тип бази даних має свої переваги та обмеження, і важливо

зрозуміти їх, щоб забезпечити ефективну та надійну роботу вашого веб-застосунку[10].

Реляційні бази даних (наприклад, MySQL, PostgreSQL) використовують таблиці зі зв'язками між ними, що дозволяє забезпечити структурованість та цілісність даних. Нереляційні бази даних, з іншого боку, надають більшу гнучкість та швидкодію, особливо при роботі з великими обсягами неструктурованих даних.

При виборі бази даних варто враховувати також фактори, як масштабованість системи, вимоги до резервного копіювання та відновлення даних, підтримка транзакцій, аналітичні можливості та вартість розгортання та підтримки системи. Зрозуміння особливостей різних типів баз даних допоможе зробити обґрунтований вибір і забезпечити оптимальну роботу вашого веб-застосунку.

На рисунку 3.5 наведено основні популярні бази даних.

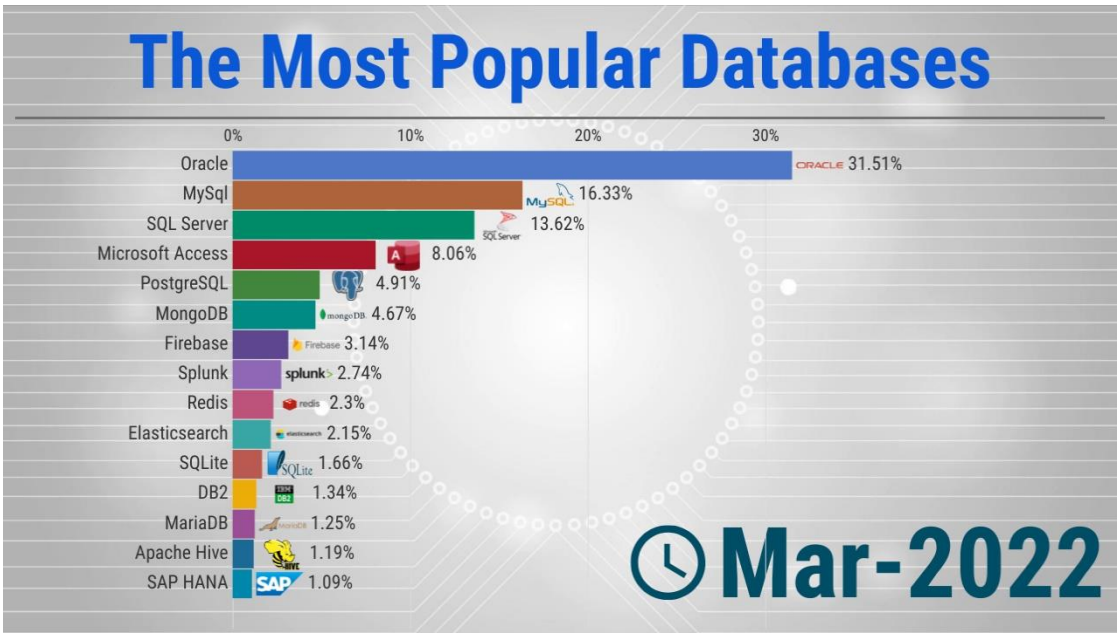

Рисунок 3.5 – Популярні бази даних за 2022р.

SQL і NoSQL є двома основними підходами до організації та управління базами даних. Вони мають суттєві відмінності у своїх моделях даних, схемах, запитах та використанні.

SQL бази даних, такі як MySQL, PostgreSQL, Oracle, використовують реляційну модель даних. Основні особливості SQL включають:

1) Табличну організацію даних зі структурою, що складається з таблиць, рядків та стовпців.
2) Використання SQL мови для створення, зміни та операцій над даними за допомогою запитів, таких як SELECT, INSERT, UPDATE, DELETE.
3) Встановлення схеми даних, яка описує структуру таблиць, включаючи типи даних, обмеження та зв'язки між таблицями.
4) Забезпечення цілісності даних за допомогою використання зв'язків (foreign keys) та правил цілісності.

NoSQL бази даних, такі як MongoDB, Cassandra, Redis, використовують нереляційну модель даних. Основні особливості NoSQL включають:

1) Гнучку структуру даних, що дозволяє зберігати неструктуровані та поліморфні дані.
2) Відсутність жорсткої схеми даних, що дозволяє змінювати структуру даних без перестановки всіх записів.
3) Використання різних моделей даних, таких як документ-орієнтовані, стовпцеві, ключ-значення, графові, для зберігання різноманітних типів даних та задоволення різних вимог.

Одним з ключових відмінностей між SQL і NoSQL є підхід до масштабування. SQL бази даних зазвичай використовують вертикальне масштабування (збільшення обсягу ресурсів на окремому сервері), тоді як NoSQL бази даних надають горизонтальне масштабування (розподілення даних на кілька серверів).

Обираючи між SQL та NoSQL, важливо враховувати потреби проекту, тип даних, гнучкість схеми, швидкодію, масштабованість та інші фактори. Кожен підхід має свої переваги та обмеження, і вибір залежить від конкретного використання та вимог вашого веб-застосунку.

На рисунку 3.6 наведено різницю між SQL та NoSQl.

Рисунок 3.6 – Різниця між SQL та NoSQL.

Висновки до розділу 3

Після аналізу характеристик та функціональних можливостей кожної з технологій, було визначено, що React є найкращим вибором для розробки клієнтської частини веб-застосунку. Однією з основних переваг React є його компонентний підхід, що дозволяє розбити інтерфейс на невеликі і повторно використовувані компоненти. Це спрощує розробку, підтримку та розширення коду, а також сприяє структурованості та чистоті проекту. Крім того, велика кількість сторонніх бібліотек, інструментів та розширень для React дозволяють розробникам швидко розширювати функціональність своїх проектів та використовувати найновіші розробки у веб-розробці. Враховуючи всі ці фактори, можна зробити висновок, що React є привабливим вибором для розробки веб-застосунків. Він надає зручний інструментарій для створення модульного, ефективного та масштабованого інтерфейсу, а також забезпечує широкі можливості для співпраці та підтримки завдяки активній спільноті розробників.

Обгрунтованість обрання Firebase для невеликого проекту полягає у його здатності швидко розпочати роботу з базою даних та іншими сервісами.

Firebase - це інтегрована платформа, яка надає інструменти для розробки веб та мобільних додатків зі зручним інтерфейсом та готовими сервісами. Загалом, Firebase забезпечує простоту, швидкість та готовість до використання сервіси, що робить його привабливим вибором для невеликих проектів, де потрібно швидко розпочати роботу та мати доступ до потужних функцій.

Таким чином, вибір React.js та FireBase для розробки веб-застосунку є обґрунтованим і забезпечує швидку та ефективну розробку, масштабування та підтримку застосунку.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕРНЕТ-СЕРВІСУ АНАЛІЗУ ВЛАСНИХ ДОСЯГНЕНЬ

4.1 Архітектура системи

Архітектура системи інтернет-сервісу аналізу власних досягнень є важливим компонентом розробки сучасних додатків, що дозволяють користувачам оцінювати, вимірювати та аналізувати свої досягнення у різних аспектах життя. Цей тип сервісу стає все популярнішим, оскільки люди шукають способи поліпшити своє саморозвиток, здоров'я, навчання та роботу.

Успішна реалізація такого сервісу вимагає ретельного планування, розуміння потреб користувачів, аналізу вимог до системи, вибору відповідних технологій, створення зручного інтерфейсу, налагодження алгоритмів обробки даних та забезпечення захисту конфіденційності особистої інформації користувачів.

В результаті, користувачі отримують можливість систематичного відстеження свого прогресу, виявлення тенденцій та слабких місць, що сприяє зміцненню мотивації, установленню нових цілей та досягненню особистого росту та розвитку.

Загальна архітектурна модель, слугує основою для розробки системи аналізу власних досягнень, проте конкретна реалізація може варіюватися залежно від потреб та вимог конкретного проекту.

Архітектура системи інтернет-сервісу аналізу власних досягнень є ключовим елементом розробки і реалізації проекту. Вона визначає структуру та організацію компонентів системи, їх взаємозв'язок та функціональність. Цей підрозділ розділу "Програмна реалізація" присвячений детальному опису архітектури системи, зокрема фронтенду та бази даних.

4.2 Фронтенд та його функціонал

Фронтенд є клієнтською частиною системи, з якою взаємодіє користувач. Він відповідає за відображення інтерфейсу та забезпечення зручного та ефективного взаємодії з системою. На рисунку 4.1 наведено вигляд головної сторінки.

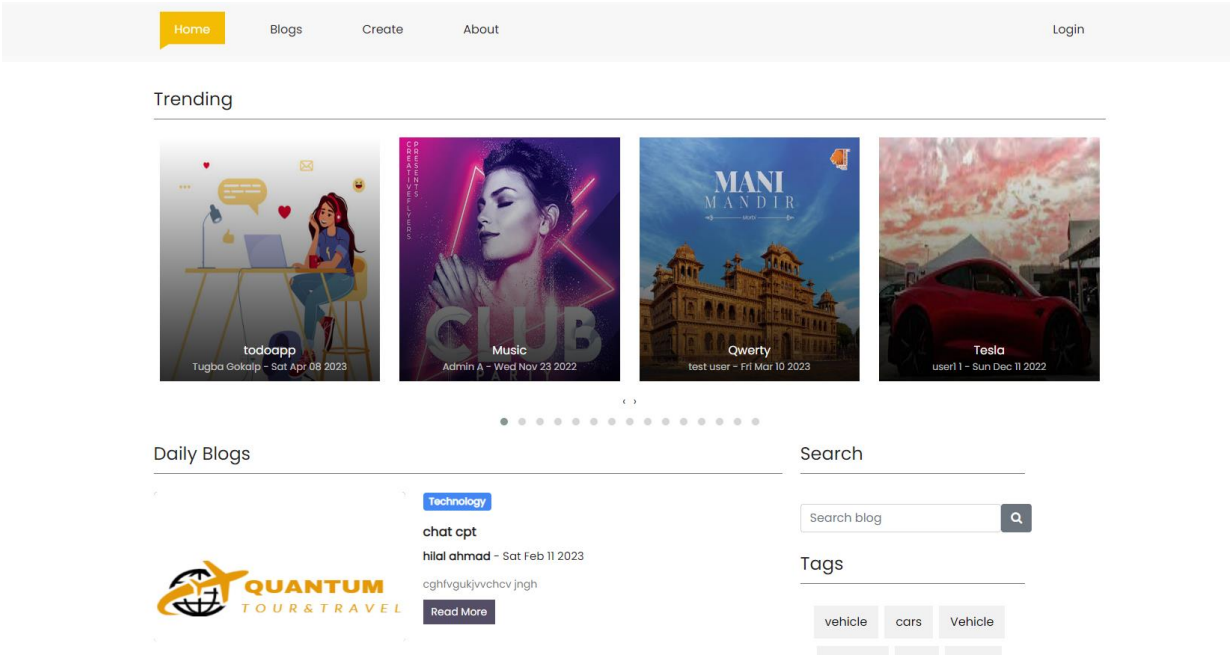

Рисунок 4.1 – Головна сторінка

При розробці дизайну фронтенда системи було використано заголовок (хедер) з панеллю навігації. Цей елемент дизайну забезпечує зручну навігацію користувача по системі та містить навігаційне меню. Панель навігації містить пункти меню, які дозволяють користувачеві переходити на різні сторінки сайту;

У тілі головної сторінки була реалізована система популярних постів, які розділяються за спеціальними тегами.

В загальному списку постів, який доступний користувачам, реалізована можливість перегляду кожного посту з коротким описом та фотографією. Крім того, у цьому списку також присутня панель з фільтром, яка дозволяє користувачам швидко знайти пости за певними критеріями.

Функціональна панель фільтра дозволяє користувачам відфільтрувати пости за різними тегами, такими як спорт, технології, здоров'я та інші характеристики. Користувачі можуть встановити певні значення тегів та отримати список постів, які відповідають їх вимогам. На рисунку 4.2 наведено роботу фільтру.

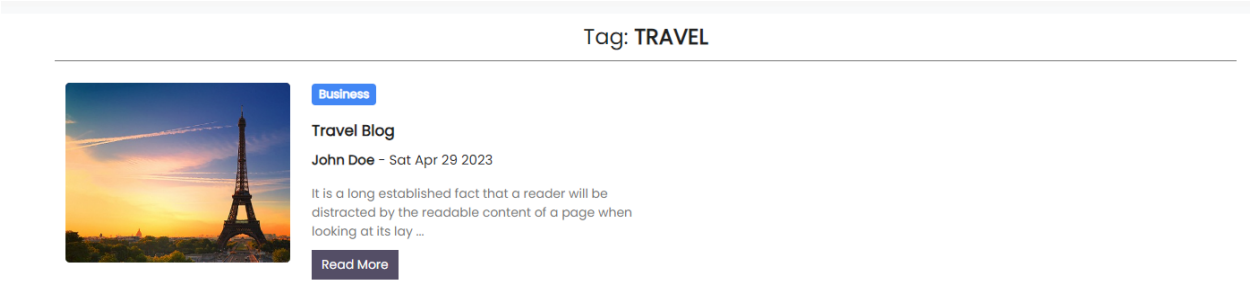

Рисунок 4.2 – Сторінка використання тегів

На сторінці Blogs виставляються останні завантажені пости на сайт. Показується лише основна інформація про пост, а саме тег, назва, дата та короткий опис. На рисунку 4.3 наведено вигляд такої сторінки.

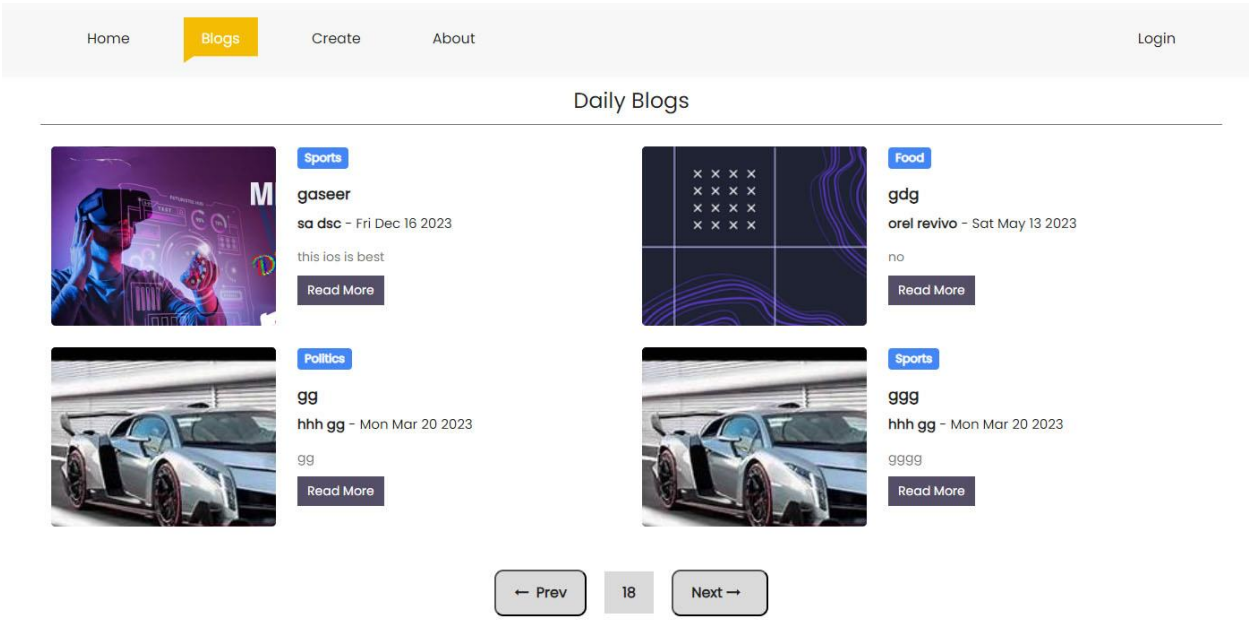

Рисунок 4.3 – Сторінка Blogs

Для реєстрації користувачів створено відповідну сторінку на якій можна створити обліковий запис що має права звичайного користувача та який може редагувати контент який сам і завантажив на сайт. На рисунку 4.4 наведено зображення сторінки реєстрації.

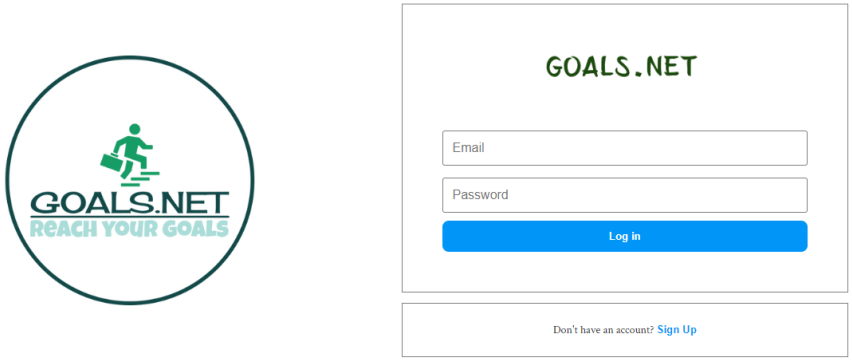

Рисунок 4.4 – Сторінка реєстрації

На сторінці About показана основна інформація про сайт. На доданок була створена панель зворотного зв’язку для користувачів, щоб вони могли зв’язатись з адміністрацію сайту. На рисунку 4.5 наведено панель адміністратора.

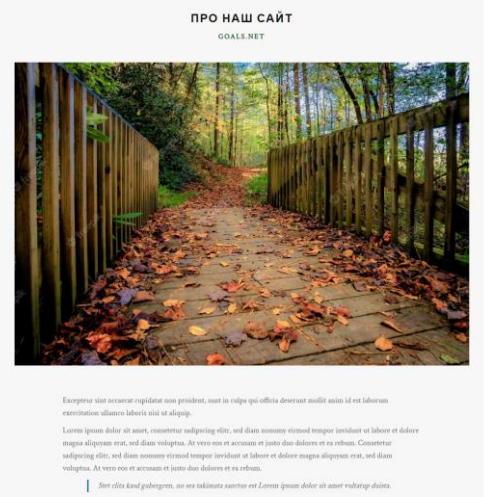

Рисунок 4.5 – Сторінка About

4.3 Діаграма класів

Діаграма класів є одним із видів діаграм, які використовуються в об'єктно-орієнтованому програмуванні для візуалізації структури класів та взаємозв'язків між ними. Вона надає графічне представлення класів, їх властивостей, методів та зв'язків між об'єктами.

Наведено опис класів фронтенду.

App.js головний компонент React програми, що включає імпорти, компоненти, хуки та функції для створення веб-сайту "Goals.net".

AddEditBlog.js React-компонент для форми створення постів та їх редагування, що використовує бібліотеку Formik для валідації форми, а також компоненти з React-Bootstrap для інтерфейсу форми.

Detail.js використовується для відображення інформації про пости на сторінці. Отримує дані з сервера за допомогою HTTP-запитів і відображає їх, включаючи зображення.

BlogSection.js відображає список постів та тегів для пошуку. Містить стейти та функції для обробки фільтрів та відображення відфільтрованого списку постів.

LogInPage.js є компонентом для сторінки входу користувача. Він містить форму з полями для електронної пошти та пароля, а також кнопку для входу. Компонент також здійснює валідацію полів форми з використанням Yup та відображає повідомлення про помилки.

LogInUser.js є компонентом для входу користувача, який виконує перевірку облікових даних та аутентифікацію на сервері.

CreateUser.js є компонентом для створення нового користувача, який виконує перевірку унікальності електронної пошти та зберігає дані користувача на сервері.

SignUpPage.js є компонентом сторінки реєстрації нового користувача. Включає форму з полями для введення інформації, включаючи ім'я, прізвище,

електронну пошту та пароль. Застосовує валідацію полів форми з використанням Yup і відображає повідомлення про помилки. На рисунку 4.6 наведено діаграму класів фронденту.

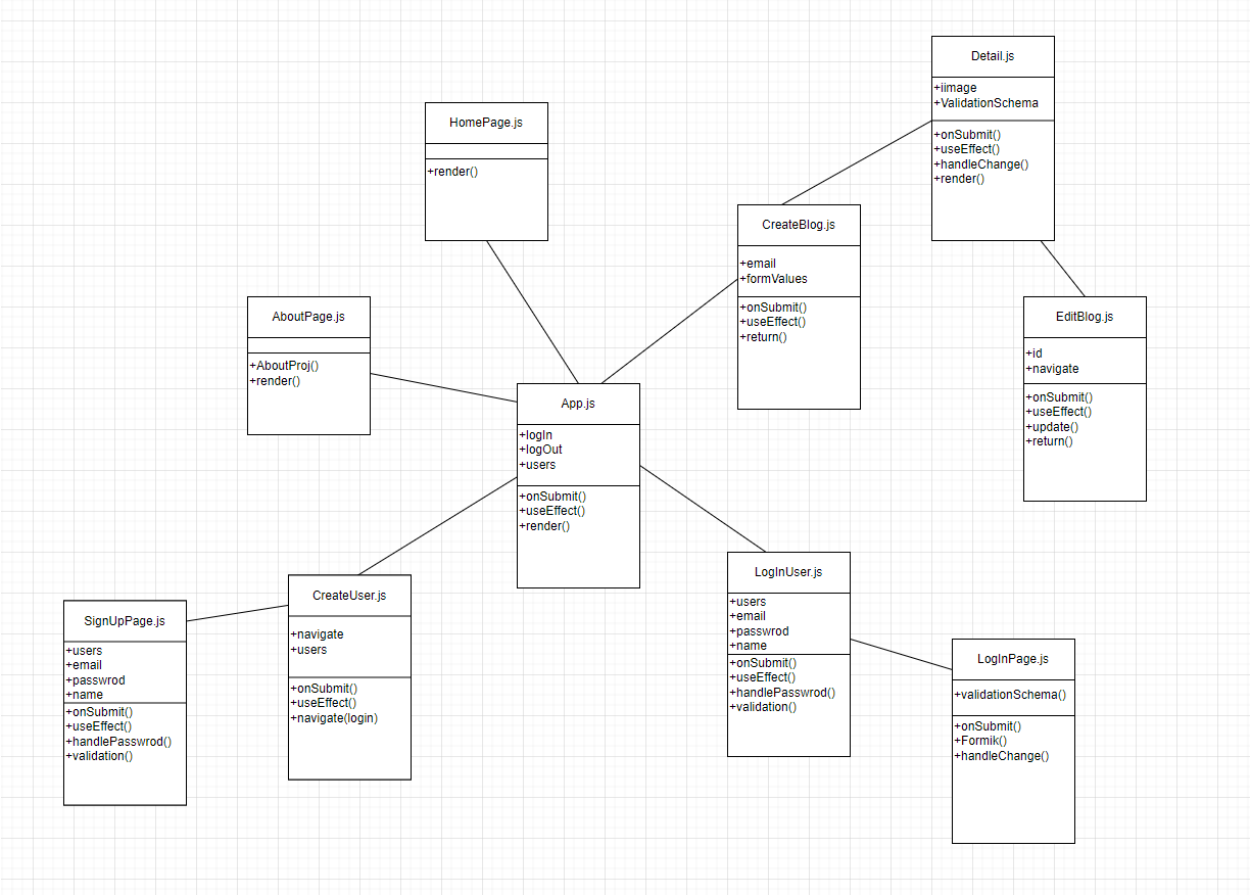

Рисунок 4.6 – Діаграма класів фронденту

Діаграми класів можуть бути використані для аналізу коду. Вони можуть допомогти виявити залежності між класами, виявити непотрібні або некоректні залежності, а також виявити можливі шляхи для оптимізації коду. Аналіз діаграм класів може бути особливо корисним при великих проектах зі складною архітектурою.

4.4 Тестування веб-застосунку

4.4.1 Мета тестування

Мета тестування веб-застосунку для аналізу власних досягнень полягає в перевірці належної функціональності компонентів та їх взаємодії.

4.4.2 Передумови

Даний веб-застосунок є платформою для аналізу власних досягнень, що надає користувачам можливість розмістити пости про себе або знайти потрібний пост за допомогою пошуку та фільтрації.

4.4.3 Вимоги до застосунку

— Система повинна дозволяти користувачам шукати пости за різними тегами, такими як спорт, транспорт, медицина та інші;

— Результати пошуку повинні бути точними та відображатися у зрозумілій та зручній для користувача формі;

— Система повинна надійно зберігати та керувати інформацією про облікові записи користувачів;

— Користувачі повинні мати можливість легко створювати облікові записи та керувати ними;

— Система повинна надавати доступ для створення постів користувачам;

— Система повинна надійно зберігати та керувати інформацією про пости користувачів;

— Користувачі повинні мати можливість створювати, редагувати та видаляти свої пости;

— Пости повинні відображатися у загальному списку та у власному списку користувача;

4.4.4 Стратегія тестування

Процес ad-hoc тестування веб-застосунку розподілено на два етапи для забезпечення якості та коректності роботи.

Функціональне тестування, перевірка основних функцій та сценаріїв застосунку для виявлення помилок.

Тестування інтерфейсу користувача, перевірка зручності та логічності інтерфейсу, включаючи навігацію, взаємодію з елементами та відповідність дизайну та вимогам щодо користувацького інтерфейсу.

Ці етапи допомагають забезпечити якість та надійність веб-застосунку, виявити та усунути помилки та недоліки, а також забезпечити зручність та безпеку для користувачів.

4.4.5 Етапи тестування

Етап функціонального тестування.

Ціль, впевнитися що користувач може використовувати основні функції застосунку в процесі реального використання. Перевірити ступінь зручності програми для користувача за основними критеріями та власного сприйняття.

Процес проходження етапу. Емуляція повного циклу використання програми: перегляд постів, використання фільтру, створення посту, створення облікового запису, редагування облікового запису. Аналіз зручності роботи користувача з програмою на основі виникаючих потреб.

Результатом етапу є те що всі вказані функції працюють належним чином.

Етап тестування користувацького інтерфейсу.

Ціль, виявити помилки в роботі користувацького інтерфейсу застосунку, шляхом реалізації своєрідних сценаріїв що в теорії можуть спричини помилку в роботі.

Процес проходження етапу. Введено некоректні дані у числові поля вводу, спостереження за реакцією програми та аналіз поведінки при обробці помилкових даних. Створені сценарії з помилками та винятками, проведення дій у неправильній послідовності або провокування ситуацій, що можуть призвести до помилок. Спостереження за реакцією програми, отримання повідомлень про помилки та аналіз їх змісту.

Після проведення тестування було отримано позитивні результати. Програма продемонструвала стабільну реакцію на некоректні дані, а також виявила та обробила помилки і винятки відповідно до передбачених сценаріїв.

4.4.6 Ресурси

Беручи до уваги, що подібні рішення вже існують достатньо давно з чого можна зробити висновки що основні функції протестовані та їх робота відрегульована, що знижує риск некоректної роботи програми. Достатньо буде лише одного ad-hoc тестувальника.

4.4.7 Висновки з тестування

Після проведення функціонального тестування та тестування користувацького інтерфейсу було отримано позитивні результати. Всі основні функції веб-застосунку для аналізу власних досягнень засобів працюють належним чином. Результати пошуку відображаються точно та зручно для користувача, інформація про облікові записи користувачів та оголошення надійно зберігається та керується системою. Користувачам надається можливість створювати, редагувати та видаляти свої оголошення, а адміністратор може редагувати дані користувачів та їх облікові записи.

Тестування користувацького інтерфейсу також показало позитивні результати. Програма стабільно реагує на некоректні дані та виявляє та обробляє помилки та винятки згідно з передбаченими сценаріями.

Загалом, веб-застосунок для аналізу власних досягнень успішно пройшов тестування і відповідає вимогам, що ставляться до нього.

4.5 Інструкції користувача веб-застосунку

Функції пошуку та фільтрації:

— На головній сторінці системи знайдіть розділ "Blogs";

— Оберіть теги для пошуку, такі як спорт, медицина, транспорт та технології;

— Натисніть на кнопку "Застосувати фільтр";

— Система відобразить результати пошуку у зрозумілій та зручній формі;

Система облікових записів користувачів:

— Зайдіть до системи та знайдіть розділ "Login";

— Якщо у вас вже є обліковий запис, натисніть на "Login" та введіть свої дані для входу. Якщо ви новий користувач, натисніть "SignUp" та заповніть необхідну інформацію;

— Перевірте правильність введених даних та натисніть на кнопку "Login" або "SignUp";

— Після успішного входу в обліковий запис ви матимете доступ до системи управління постами та інших функцій;

Система управління постами:

— У вашому обліковому записі знайдіть розділ "Create";

— Заповніть всю необхідну інформацію про ваш пост, включаючи ім’я, теги, опис та фото;

— Перевірте інформацію, щоб вона була коректною, та натисніть на кнопку "Submit", щоб зберегти зміни;

Висновок до розділу 4

Була представлена детальна архітектура системи інтернет-сервісу аналізу власних досягнень, з фокусом на функціоналі фронтенду та його основних компонентах. Описано інтерфейс, а також компоненти, які використовувалися. Для перевірки функціоналу та інтерфейсу системи була використана стратегія ad-hoc тестування, яка включала функціональне та тестування інтерфейсу користувача. В результаті тестування було підтверджено, що застосунок успішно відповідає вимогам, включаючи функції пошуку, створення постів та керування обліковими записами користувачів. Користувачам також були надані інструкції з використання веб-застосунку.

ВИСНОВКИ

У даній роботі проведено аналіз предметної сфери розробки програмного забезпечення для інтернет-сервісу аналізу власних досягнень. Було проаналізовано сучасні аналоги інтернет-сервісів та сформульовано специфікацію вимог до програмного забезпечення такого проекту.

У розділах було проведено моделювання та проєктування програмного забезпечення інтернет-сервісу аналізу власних досягнень. Для цього було розроблено діаграму прецедентів, проєктування інтерфейсу веб-застосунку, візуальну карту веб-застосунку та проєктування бази даних.

Розроблено функціональні модулі програмного забезпечення і проєктовано зручний та інтуїтивно зрозумілий інтерфейс. Була впроваджена функція пошуку та фільтрації. Була впроваджена система облікових записів користувачів. Для забезпечення безпеки та надійності контенту була розроблена система управління та модерації. Проведено тестування програмного забезпечення. Були проведені різноманітні тести, включаючи функціональні тести, тести на відповідність вимогам. Застосунок готовий до експлуатації і забезпечує користувачам зручний та надійний інтерфейс для аналізу власних досягнень.

Загальний висновок полягає в тому, що розробка програмного забезпечення інтернет-сервісу аналізу власних досягнень є складним процесом, який потребує детального аналізу та проєктування. Для досягнення успіху в цій галузі, необхідно враховувати потреби користувачів та забезпечувати зручний та простий у використанні інтерфейс. Також важливо правильно спроектувати базу даних, щоб забезпечити ефективне зберігання та обробку інформації про користувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. "Learning React: A Hands-On Guide to Building Web Applications Using React and Redux" by Kirupa Chinnathambi, published by Addison-Wesley Professional in 2017. 322 pages.

2. "React Native: Building Mobile Apps with JavaScript" by Bonnie Eisenman, published by O'Reilly Media in 2017. 342 pages.

3. "Fullstack React: The Complete Guide to ReactJS and Friends" by Anthony Accomazzo, Nate Murray, and Ari Lerner, published by Fullstack.io in 2017. 800 pages.

4. "React Design Patterns and Best Practices" by Michele Bertoli, published by Packt Publishing in 2017. 344 pages.

5. "Firebase for Web Development" by Salman Ul Haq, published by Packt Publishing in 2016. 156 pages.

6. Офіційна документація React. URL : https://reactjs.org/docs/getting-started.html.(Last accessed: 04.04.2023).

7. "Learning React: A Hands-On Guide to Building Web Applications Using React and Redux" by Kirupa Chinnathambi, published by Addison-Wesley Professional in 2017. 240 pages.

8. "React: Up & Running: Building Web Applications" by Stoyan Stefanov, published by O'Reilly Media in 2016. 250 pages.

9. "React.js Essentials" by Artemij Fedosejev, published by Packt Publishing in 2015. 152 pages.

10. "Firebase Essentials: Get to Know Firebase Real-Time Database and Its Authentication Features" by Neil Smyth, published by CreateSpace Independent Publishing Platform in 2016. 138 pages.

11. "The Road to React: Your journey to master plain yet pragmatic React.js" by R. Virle, Leanpub, 2020. 272 pages.

12. "Learning React: A Hands-On Guide to Building Web Applications Using React and Redux" by Nixon R. Boston: Addison-Wesley Professional, 2018. 304p.

13. "React and Express: Full Stack Web Development with Node.js and React" by Banks A., Porcello E. Sebastopol, O'Reilly Media, 2019. 450p.

14. "React for Beginners: A Practical Guide to Building User Interfaces with React" by Bos W., Toronto: Self-p, 2018. 250p.

15. "Understanding React and Redux: A Comprehensive Guide to Modern Web Development" by Rascia T., Chicago: Self-published, 2019. 280p.

16. "Responsive Web Design: Building Flexible Websites for a Multi-Device World" by Marcotte E., New York: A Book Apart, 2016. 180p.

17. "React Design Patterns and Best Practices: Build easy to scale modular applications using the most powerful components and design patterns" by M. Bertoli, Packt Publishing, 2017, 440 pages.

18. "React Cookbook: Create dynamic web apps with React using Redux, Webpack, Node.js, and GraphQL" by C. Roldán, Packt Publishing, 2020, 422 pages.

19. "Fullstack React: The Complete Guide to ReactJS and Friends" by A. Accomazzo, N. Murray, Fullstack.io, 2020, 810 pages.

20. "Mastering React: Master the art of building modern web applications using React" by A. Horton, Packt Publishing, 2016, 404 pages.

21. "React for Real: Frontend Code, Untangled" by L. Fischer, Leanpub, 2021, 444 pages.