**Форма № 24**

Міністерство освіти і науки України

**Національний університет «Запорізька політехніка»**

Інститут інформатики та радіоелектроніки

Факультет комп’ютерних наук і технологій

(повне найменування інституту, факультету)

Кафедра програмних засобів

(повне найменування кафедри )

**Пояснювальна записка**

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ВІДСТЕЖЕННЯ КОРИСТУВАЦЬКОГО СПИСКУ ЧИТАННЯ. DEVELOPMENT OF A WEB APPLICATION FOR TRACKING A USER'S READING LIST

Виконав: студент(ка)4 курсу, групи КНТ-210

Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Індик А.А.

(прізвище та ініціали)

Керівник Федорончак Т.В.

(прізвище та ініціали)

Рецензент Поляков М.О.

(прізвище та ініціали)

2024

**Форма № 25**

Міністерство освіти і науки України

**Національний університет «Запорізька політехніка»**

( повне найменування закладу вищої освіти)

Інститут, факультет ІІРЕ, ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти бакалавр

Спеціальність 122 Комп’ютерні науки

(код і найменування)

Освітня програма (спеціалізація) Комп’ютерні науки

(назва освітньої програми (спеціалізації)

**ЗАТВЕРДЖУЮ**

**Завідувач кафедри ПЗ, д.т.н, проф.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С.О. Субботін**

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_року

**З А В Д А Н Н Я**

**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)**

Індика Андрія Андрійовича

(прізвище, ім’я, по батькові)

1. Тема проєкту (роботи) Розробка вебзастосунку для відстеження користувацького списку читання. Development of a web application for tracking a user's reading list

Керівник проєкту (роботи) Федорончак Тетяна Василівна, к.т.н., доцент ,

( прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від 0 квітня 2024 року № 23

2. Строк подання студентом проєкту (роботи) 0 травня 2024 року\_\_\_\_\_\_\_\_\_\_\_

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Розробка архітектури програми. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація, тестування та експериментальне дослідження програми. 5. Технічне завдання. 6. Текст програми.

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада  консультанта | Підпис, дата | |
| завдання видав | прийняв  виконане завдання |
| 1-5 Основна частина | Федорончак Т.В., доцент |  |  |
| Нормоконтроль | Липовець М.В., асистент |  |  |

7. Дата видачі завдання 15 березня 2024 р.

#### **КАЛЕНДАРНИЙ ПЛАН**

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

**Студент (ка) \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

( підпис ) (прізвище та ініціали)

**Керівник проєкту (роботи) \_\_\_\_\_\_\_\_\_\_\_\_\_** Федорончак Т.В.

( підпис ) (прізвище та ініціали)

**РЕФЕРАТ**

Пояснювальна записка до випускної кваліфікаційної роботи бакалавра: 22 с., 2 табл., 12 рис., 3 дод., 11 джерел.

СПИСОК ЧИТАННЯ, ВЕБЗАСТОСУНОК, ВІДСТЕЖЕННЯ, ЛІТЕРАТУРА, JAVASCRIPT.

Об’єкт дослідження – засоби для управління списком літератури.

Предмет дослідження – вебзастосунки для створення та відслідковування власного списку читання.

Мета роботи – створення вебзастосунка для покращення слідкування за власним списком читання, зберіганням прочитаної літератури, запланованої та літератури, що читається в поточний момент, зручний пошук та сортування цієї літератури.

Матеріали, методи та технічні засоби: мови розмітки сторінки HTML та стилів CSS, мова програмування JavaScript, бібліотека React на її основі, персональний комп’ютер з процесором Intel Core I7-3770 під управлінням ОС Microsoft Windows 10, вебпереглядач на основі Chromium версії 124.

Результати. Створено вебзастосунок, що керується мовою JavaScript, який дає користувачу можливість керувати списком читання.

Висновки. Розроблено систему керування особистим списком читання користувача, до якої можна додавати нову літературу, інформацію про неї, розподіляти її за станами читання та категоріями.

Галузь використання – покращення та структуризація хобі читання, створення списку прочитаного, відслідковування поточної зайнятості та планування читання майбутньої літератури.

**ABSTRACT**

Explanatory note to the final qualification work of a bachelor: 22 p., 2 tables, 12 figures, 3 appendixes, 11 sources.

READING LIST, WEB APPLICATION, TRACKING, LITERATURE, JAVASCRIPT.

Object of research is tools for managing the reading list.

Subject of research is Web applications for creating and tracking your own reading list.

Purpose is to create a Web application to improve the tracking of your own reading list, storage of read literature, planned and currently read literature, convenient search and sorting of this literature.

Materials, methods, and equipment: HTML and CSS page markup languages, JavaScript programming language, React library based on it, a personal computer with an Intel Core I7-3770 processor running Microsoft Windows 10, a web browser based on Chromium version 124.

Results. A JavaScript-based web application that allows the user to manage the reading list has been created.

Conclusions. A system for managing a user's personal reading list, to which new literature can be added, information about it, and distributed by reading status and category has been developed.

The field of application is improving and structuring reading hobbies, creating a reading list, tracking current employment and planning the reading of future literature.

ЗМІСТ

[ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК 7](#_Toc166020869)

[ВСТУП 8](#_Toc166020870)

[1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 9](#_Toc166020871)

[1.1 Предметна область 9](#_Toc166020872)

[1.2 Огляд аналогів 11](#_Toc166020873)

[1.3 Постановка завдання 24](#_Toc166020874)

[1.4 Висновки з розділу 26](#_Toc166020875)

[2 РОЗРОБКА АРХІТЕКТУРИ ПРОЄКТУ 27](#_Toc166020876)

[2.1 Вибір інструментів розробки 27](#_Toc166020877)

[ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 41](#_Toc166020878)

# ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

|  |  |
| --- | --- |
| HTML | – HyperText Markup Language |
| CSS | – Cascading Style Sheets |
| JSX | – JavaScript XML |
| DOM | – Document Object Model |
| npm | – Node Package Manager |
| ГБ | – Гігабайт |
| ОЗП | – Оперативний Запам'ятовуючий Пристрій |
| ОС | – Операційна Система |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# ВСТУП

Актуальність цієї роботи полягає в тому, що книги вже десятки сотень років являються головними джерелами знань та інформації, що написані одними людьми для читання іншими. І хоч форма представлення письмової інформації змінювалася з роками, починаючи свій шлях з глиняних табличок, папірусу, пергаменту, дійшла до звичного нам паперу, з яким людство паралельно починає використовувати електроні книги та аудіокниги, які можна читати на будь-якому носії інформації, який призначений для її відтворення, це може бути телефон, комп’ютер або навіть окремий девайс для читання, який і називається електронна книга. Це показує нам, що хоч ми і відходимо від звичного сприйняття паперових книг, самі книги не зникають і ніколи не зникнуть. Папір замінюється планшетами, бібліотеки – інтернет-репозиторіями. В світі, де актуальність книг безперечна і це навряд чи зміниться у майбутньому, необхідні засоби для зручного контролю того, що вже прочитано, та що стоїть у черзі для прочитання у майбутньому.

Метою цієї роботи є створення програмного забезпечення у вигляді вебзастосунку, основною ціллю якого буде спрощення процесу відслідковування списку читання користувача, допомога в його структуризації та підбитті підсумків читання.

Задача цієї роботи полягає у розробці проєкту для відстеження користувацького списку читання, створення його зовнішньої складової, його програмна реалізація у вигляді вебзастосунку, побудованого на мові програмування JavaScript та його бібліотеці React.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Предметна область

Після винаходу стародавніми цивілізаціями систем запису люди почали використовувати для письма майже все, на чому можна писати — глиняні таблички, кору дерева, листи металу тощо. Перші книги писали на листях, корі, полотні, шовку, глиняних табличках, шкірі і папірусі [1].

У Стародавньому Єгипті для запису, з часів Першої Династії, використовувався папірус (вид паперу, зробленого із стебел однойменної рослини). Окремі листи папірусу, для зручності зберігання, склеювалися в сувої [1].

У школах, бухгалтерії і для нотаток зазвичай використовувалися воскові таблички. Вони мали перевагу багаторазового використання: віск можна було розплавити і наносити новий текст. Зв'язування таких табличок — можливий попередник сучасних книг [1].

У перших книгах для сторінок використовували пергамент або веленевий папір (шкіру теляти). Обкладинки були зроблені з деревини і покриті шкірою [1].

Як і багато інших середньовічних винаходів, перший папір був виготовлений у Китаї, в 200 р. до н. е. і досяг Європи через мусульманські території. Спочатку його робили з тканини, але промислова революція дозволила почати робити папір з більш дешевого матеріалу — целюлози [1].

Тільки у 1880-х папір отримав широке розповсюдження. На початку 1800-х стали популярні парові друкарські машини [1].

Багато років, починаючи з XV століття, були витрачені на поліпшення друкарських машин і прийняття свободи слова, поступово знижуючи рівень цензури [1].

У 2010 році, за оцінками Google, було близько 130 мільйонів унікальних книг у світі [2].

Зараз текст книги можна відтворити і зберігати у вигляді мікрофільму та в електронному вигляді на відповідному носії інформації [1].

Книги існували століттями як джерело інформації та знань, відтворюючи важливі тексти та твори, створені людьми для використання та аналізу. Від глиняних табличок, папірусу та пергаменту до звичайного паперу. Водночас книги не втратили своєї актуальності, навіть коли електронні книги та аудіокниги стали доступними на різноманітних носіях, включаючи мобільні телефони, комп'ютери та спеціалізовані пристрої для читання.

Перш за все, книги залишаються невичерпним джерелом знань та інформації, створених однією людиною для сприйняття і розуміння іншою. Книги є основою для передачі знань і культурних цінностей від покоління до покоління і забезпечують безперервний розвиток суспільства.

По-друге, книги є свідками еволюції людської думки та ідей, відображаючи розмаїття культур, світоглядів і способів мислення. Вони є важливими реліквіями нашої культурної спадщини, сприяють діалогу між сьогоденням і минулим та допомагають зберегти пам'ять про наше коріння і цінності.

По-третє, книги є фундаментальними інструментами для критичного мислення, самоосвіти та саморозвитку. Книги стимулюють мислення, допомагають розібратися в складних проблемах і знайти можливі рішення, надихають на досягнення.

Таким чином, книги залишаються актуальним і необхідним джерелом знань, мудрості та натхнення в сучасному світі. Важливість книги полягає в комунікації, яка сприяє збереженню та передачі культурної спадщини, особистісному зростанню та безперервному розвитку суспільства.

Розробка ефективних інструментів для структурування літературних творів важлива для забезпечення організованості та послідовності в процесі сприйняття та аналізу інформації, яку ми споживаємо. Історія літератури багата на розмаїття жанрів, галузей, тем і авторів, тому можливість вести списки літератури є важливим компонентом для кожного читача.

Інструменти управління списком літератури допоможуть не загубитися у величезному морі знань, які може запропонувати людство. Завдяки цим інструментам ми можемо структурувати свій читацький досвід, розставляючи пріоритети та плануючи майбутні роботи.

Особливо важливим є можливість формувати особисті враження та висновки про прочитані твори. Зберігаючи ці враження, користувач може періодично повертатися до них, щоб оновити пам'ять про свої знання та розуміння прочитаних робіт.

Тому використання інструментів для контролю та організації списку читання є важливим елементом у сьогоднішньому дедалі різноманітнішому інформаційному світі. Інструменти допомагають нам не лише зберігати та структурувати знання, але й активно використовувати їх для розвитку та збагачення нашого світогляду.

## 1.2 Огляд аналогів

Перед створенням проєкту на тему відстеження списку читання, обов’язковим кроком є огляд аналогічних успішних проєктів.

Перше, на що варто звернути увагу, – це функціональність інших платформ. До сильних сторін можна віднести інтуїтивно зрозумілий інтерфейс, можливість додавання різних форматів контенту та ефективну систему фільтрації та сортування. До слабких сторін можна віднести обмежену можливість ділитися списками літератури, нестабільну роботу мобільних додатків і погану інтеграцію з іншими сервісами.

По-друге, важливо порівняти доступні функції з тими, що пропонують різні альтернативи. Якщо в одному проєкті відсутня функція, яка є стандартною на інших схожих платформах, це може бути гарною можливістю покращити функціонал. Наприклад, якщо проєкти пропонують можливість залишати коментарі або оцінювати літературу, то це один з елементів, який потенційно може привернути увагу майбутніх користувачів. Якщо ж інші проєкти з якоїсь причини не мають певних функцій, треба визначити, наскільки вони необхідні та чи впливають на зацікавленість у підсумку.

Першим проєктом для порівняння і, напевно, фаворитом серед всіх оберемо вебсайт Goodreads.

На рис. 1.1 зображено логотип вебсайту Goodreads.



Рисунок 1.1 – Логотип вебсайту Goodreads

Goodreads – це американський вебсайт, який слугує платформою соціального каталогу книг, надаючи користувачам можливість шукати та переглядати різні книги, анотації, уривки та рецензії в базі даних Goodreads. Він використовується для створення особистих бібліотек і списків для читання, а також для того, щоб користувачі могли відстежувати свої літературні інтереси і плани [3].

Крім того, користувачі можуть створювати власні групи для обговорення книг, обміну рекомендаціями та думками, голосування, ведення блогів та участі в різноманітних дискусіях. Ці можливості роблять Goodreads не лише інструментом для ведення особистої історії читання, але й платформою для спілкування та обміну думками з іншими книголюбами [3].

Загалом, Goodreads – це не лише місце для пошуку нових книг, але й соціальна спільнота, де книголюби з усього світу можуть ділитися своїми думками про прочитані книги, отримувати рекомендації та взаємодіяти з іншими читачами [3].

Ідея Goodreads народилася з особистого досвіду, який один із співзасновників, Отіс Чендлер, отримав, переглядаючи книжкову полицю свого друга. Його вразила ідея об'єднати цей досвід перегляду книг і створити онлайн-простір, де люди могли б ділитися своїми враженнями від прочитаних книжок [3].

Місія Goodreads полягає в тому, щоб «дати можливість людям по всьому світу знаходити і ділитися своїми улюбленими книгами... щоб покращити досвід читання та навчання в усьому світі». Ця місія відображена в особливостях і функціоналі платформи, які спрямовані на полегшення пошуку книг і спільне обговорення літературних творів [3].

«Проблема відкриття», на якій наголошує Goodreads, є нагальною в цифрову епоху, і Goodreads спрощує цей процес, допомагаючи користувачам знаходити книги, які вони хочуть прочитати, і сприяючи обміну ідеями та пропозиціями між читачами [3].

Відкриття книг. На сайті Goodreads користувачі можуть додавати книги на свої книжкові полиці, оцінювати книги, бачити, що читають їхні друзі та автори, брати участь у дискусіях та групах на різні теми, а також отримувати рекомендації щодо майбутніх книг на основі вже прочитаних. Додавши друзів до свого профілю, користувачі можуть переглядати книжкові полиці та рецензії своїх друзів, а також коментувати сторінки своїх друзів. Goodreads пропонує систему оцінювання від однієї до п'яти зірок, з можливістю написання рецензії. На сайті є стандартні книжкові полиці, такі як "Книги, які я прочитав", "Книги, які я прочитав" і "Книги, які я хочу прочитати", а також можливість для користувачів створювати власні полиці, щоб класифікувати свої книги за категоріями [4].

Доступ до контенту. Користувачі Goodreads можуть читати і слухати прев'ю книг на сайті за допомогою Kindle Cloud Reader або Audible. Крім того, Goodreads пропонує вікторини, головоломки, цитати, списки книг і безкоштовні подарунки. Користувачі отримують регулярну розсилку з новими книгами, книжковими рекомендаціями, інтерв'ю з авторами та віршами. Якщо користувач пише твір, він потрапляє на сторінку профілю автора, а також публікується в авторському блозі. Goodreads також організовує офлайн-події, такі як обмін книжками та літературні паб-кроули [4].

Взаємодія з користувачами. Сайт заохочує взаємодію між читачами та авторами через інтерв'ю, розіграші, авторські блоги та інформацію в профілі. Також є спеціальний розділ для авторів про те, як просувати свої твори на Goodreads.com, що допомагає їм досягти своєї цільової аудиторії. У 2011 році "17 000 авторів", серед яких Джеймс Паттерсон і Маргарет Етвуд, використовували Goodreads для реклами своїх творів. Користувачі можуть додавати один одного в друзі та обмінюватися відгуками, повідомленнями, рекомендаціями щодо книг, повідомленнями тощо. Goodreads присутній у Facebook, Pinterest, Twitter та інших соціальних мережах Якщо ви підключите свій обліковий запис до соціальних мереж, таких як Facebook, ви зможете імпортувати контакти зі своїх сторінок у соціальних мережах до Goodreads і розширити список друзів Goodreads. Існують також налаштування, які дозволяють Goodreads автоматично публікувати публікації в соціальних мережах, наприклад, сповіщати друзів у Facebook про прочитані користувачем книги або про оцінки, які користувач поставив книгам [4].

Goodreads має кілька переваг як інструмент для запису прочитаного. По-перше, він має велику кількість користувачів та інтегрований з Facebook, що дозволяє користувачам швидко знаходити своїх друзів на платформі. Це створює можливості для обміну пропозиціями та думками про книги. Це також полегшує пошук нових книг для читання на основі інтересів друзів та книг, які вони зараз читають або планують прочитати. Goodreads також сприяє взаємодії та обговоренню літературних творів, дозволяючи користувачам розпочинати розмови на основі коментарів своїх друзів. Нарешті, платформа дозволяє користувачам відстежувати книги, які вони хочуть прочитати, але не мають, а також книги у власній бібліотеці [5].

Однак Goodreads має кілька недоліків. По-перше, він не може відстежувати інші матеріали, такі як відеоігри чи фільми. Це обмежує його функціональність для тих, хто хоче керувати різними культурними інтересами на одній платформі. Крім того, відсутність налаштувань приватності може створити проблеми для тих, хто хоче контролювати доступ до своїх даних і рецензій. Крім того, Goodreads не пов'язаний з місцевими бібліотеками, тому необхідно вжити спеціальних заходів для перевірки наявності матеріалів у бібліотеках. Нарешті, дизайн платформи можна вважати незручним і неефективним для використання на мобільних пристроях, що ускладнює її використання на ходу і в мобільних форматах [5].

На рис. 1.2 зображено інтерфейс вебсайту Goodreads.

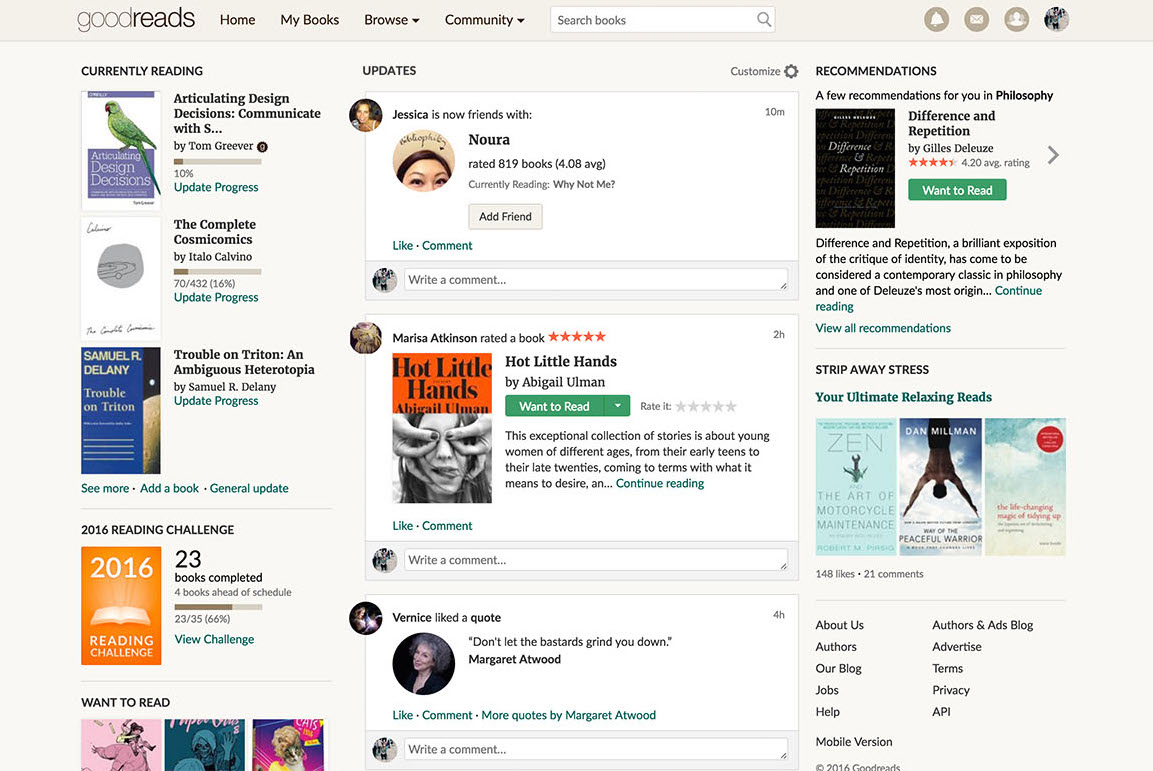


Рисунок 1.2 – Інтерфейс вебсайту Goodreads [6]

Наступним на розгляд проєктом і, напевно, головним конкурентом Goodreads є The StoryGraph.

На рис. 1.3 зображено логотип вебсайту The StoryGraph.

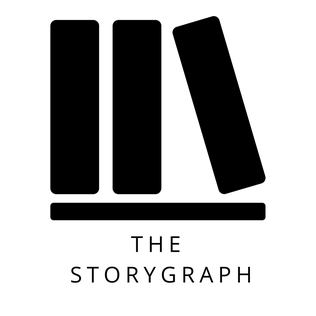


Рисунок 1.3 – Логотип вебсайту The StoryGraph [7]

TheStoryGraph – це вебплатформа соціального каталогу книжок, яка конкурує з Goodreads, яким володіє Amazon. StoryGraph набув популярності після того, як Book Riot опублікував статтю про переваги платформи, такі як персоналізовані рекомендації, настроювані опції оцінювання та незалежність від Amazon. StoryGraph використовує безкоштовну модель з деякими функціями, доступними лише через платну підписку [7].

У порівнянні з Goodreads, StoryGraph пропонує користувачам більш персоналізовані рекомендації та рейтинги. Його незалежність від Amazon також приваблює тих, хто хоче уникнути "амазонізації" книжкової індустрії і віддає перевагу альтернативним платформам. Однак, як і у випадку з багатьма безкоштовними сервісами, деякі функції доступні лише за умови сплати передплати [7].

У 2019 році програмістка Надя Одунайо запустила StoryGraph, спочатку як сайд-проєкт для відстеження книг. На основі коментарів користувачів Goodreads та інших читачів Одунайо розробила персоналізовану систему книжкових рекомендацій і зосередилася на її впровадженні на платформі [7].

Після запитів користувачів Goodreads та інших форумів Одунайо зрозуміла, що потрібен більш персоналізований підхід до книжкових рекомендацій. На платформі StoryGraph вона активно розробляє та впроваджує різні системи, спрямовані на створення персоналізованих книжкових рекомендацій, щоб кожен читач міг знайти книгу, яка відповідає його вподобанням [7].

Порівняння з Goodreads. The StoryGraph у багатьох відношеннях схожий на Goodreads, також будучи вебплатформою, призначеною для читачів з метою відстеження прочитаних книг і використання метаданих для створення профілів книг. На відміну від Goodreads, StoryGraph зосереджується на відстеженні книжкових колекцій і не має соціальної домашньої сторінки, як у Goodreads. Платформа StoryGraph дозволяє читачам реєструватися та оцінювати книги, спілкуватися з друзями на сторінках спільноти сайту та ставити читацькі завдання. Платформа надає книжкові рекомендації на основі аналізу читацьких звичок користувачів. Прокручуючи сторінку профілю, користувачі можуть оцінити свою онлайн-бібліотеку, класифікуючи її за настроєм, темпом, тривалістю, жанром і рейтингом. Ця функція може бути оновлена за щомісячну плату і пропонує більш розширену статистику; на відміну від Goodreads, StoryGraph пропонує можливість давати книгам півзірки або чверть зірки [7].

Тепер давайте розглянемо переваги The StoryGraph як інструменту для запису читання.

The StoryGraph пропонує унікальну систему каталогу на основі настрою, яка дозволяє знайти потрібну книгу відповідно до вашого настрою. Це дозволяє знайти потрібний літературний твір для будь-якого дня або настрою, спрощуючи процес пошуку нових творів [5].

Варто також відзначити, що StoryGraph дозволяє користувачам знаходити твори, яких немає в їхній бібліотеці. Це розширює кругозір і дозволяє відстежувати не лише придбані книги, а й ті, які ви плануєте прочитати в майбутньому [5].

Платформа також надає статистику про ваш стиль читання, ваш настрій і швидкість, з якою ви любите читати. Це дозволяє краще зрозуміти ваші читацькі вподобання та швидше знаходити нові та цікаві книги [5].

Витончений дизайн StoryGraph доступний як на ПК, так і на мобільних пристроях, що робить його простим у використанні в будь-який час і в будь-якому місці.

Ще однією важливою особливістю є те, що ви можете отримувати рекомендації щодо наступної книги з вашого особистого списку "до прочитання". Це сприяє безперервному розвитку вашого списку читання і забезпечує постійний потік нової літератури [5].

StoryGraph також може відстежувати популярні онлайн-челенджі з читання, такі як POPSUGAR та Book Riot. Це заохочує до активної участі в літературних челенджах і допомагає розширити читацький досвід [5].

Оглянемо недоліки StoryGraph: На відміну від Goodreads, StoryGraph не може відстежувати нічого, окрім книг, наприклад, відеоігри чи фільми. Це може бути обмеженням для тих, хто хоче відстежувати різні аспекти свого культурного життя на одній платформі [5].

Також немає налаштувань конфіденційності, що може бути проблемою для тих, хто хоче контролювати доступ до своїх даних і відгуків [5].

Крім того, StoryGraph не пов'язаний з каталогом місцевої бібліотеки, тому необхідно вжити спеціальних заходів, щоб перевірити наявність матеріалів у бібліотеці [5].

На рис. 1.4 зображено інтерфейс вебсайту The StoryGraph.

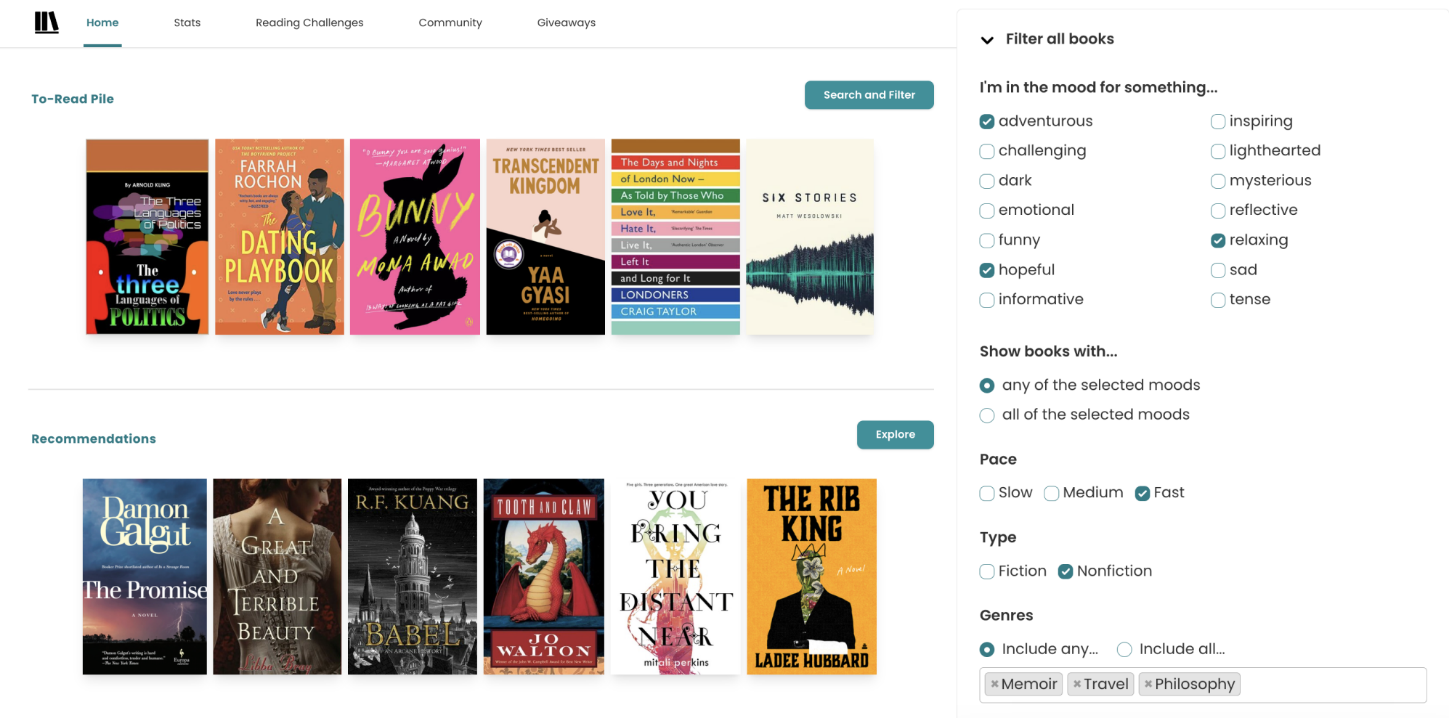


Рисунок 1.4 – Інтерфейс вебсайту The StoryGraph [8]

Останнім в огляді альтернатив розглянемо вебзастосунок LibraryThing.

На рис. 1.5 зображено логотип вебсайту The StoryGraph.

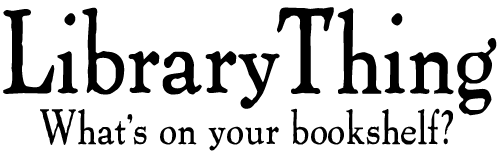


Рисунок 1.5 – Логотип вебсайту The StoryGraph [9]

LibraryThing – це вебзастосунок соціального каталогу, призначений для зберігання та обміну книжковими каталогами та різноманітними метаданими про книги. Його використовують автори, окремі користувачі, бібліотеки та видавці [9].

Заснований у Портленді, штат Мен, LibraryThing був розроблений Тімом Сполдінгом і запущений 29 серпня 2005 року за моделлю вільної підписки. Натомість, керуючись можливостями каталогізації та фінансовими викликами, спричиненими пандемією COVID-19, вони зосередилися на створенні набору продуктів для академічних бібліотек і дотрималися обіцянки не використовувати рекламу для зареєстрованих користувачів. 8 березня 2020 року сервіс став "повністю безкоштовним". Станом на лютий 2021 року він налічував 2 600 000 користувачів і колекцію з понад 155 000 000 книг, дані надані Amazon та тисячами бібліотек, що використовують протокол каталогу Z39.50 [9].

Можливості. Основна функція LibraryThing – каталогізація книг, фільмів, музики та інших медіа шляхом передачі даних з бібліотек і шести магазинів Amazon.com через з'єднання Z39.50. Бібліотечні джерела надаватимуть LibraryThing записи у форматі Dublin Core та MARC. Користувачі можуть імпортувати інформацію з понад 2000 бібліотек, включаючи Британську бібліотеку, Національний каталог Канади, Бібліотеку Конгресу, Національну бібліотеку Австралії та Єльський університет. Якщо записів немає в жодному з цих джерел, інформацію про книгу можна ввести вручну, використовуючи чистий бланк [9].

Кожен твір може мати різні видання, переклади, друковані та аудіоверсії. Учасники заохочуються додавати загальнодоступні рецензії, описи, загальні відомості та іншу інформацію про твір. Рейтинги, колекції та теги допомагають у категоризації. Також заохочуються дискусії на форумі [9].

Теми класифікуються за допомогою Десяткової системи класифікації Мелвілла. Вона базується на неавторському виданні класифікації Дьюї 1922 року, стандартному написанні назв розділів (на відміну від оригінальних назв, які були написані відповідно до рекомендацій реформи правопису Дьюї) та модифікаціях сучасної термінології [9].

Соціальні функції LibraryThing порівнювали з менеджером закладок Del.icio.us та музичним сервісом Last.fm. Подібні сайти-каталоги книг включають aNobii, BookLikes, Goodreads, Libib, Shelfari (зараз об'єднаний з Goodreads) та weRead [9].

У 2016 році LibraryThing запустила TinyCat, OPAC, призначений для каталогізації до 20 000 одиниць зберігання та для пересувних бібліотек. TinyCat заохочується для невеликих незалежних бібліотек, таких як школи, громадські центри, релігійні установи, академічні підрозділи та індивідуальні користувачі [9].

LibraryThing дозволяє користувачам шукати книги, обирати улюблені та додавати їх до особистої бібліотеки. На головній сторінці є меню "Додати книгу", щоб спростити цей процес. Крім того, користувачі можуть вибрати джерело, з якого шукати книги. Наприклад, вони можуть обрати Бібліотеку Конгресу США або Amazon. Платформа має понад 700 джерел даних, які можна відфільтрувати за мовою та країною [10].

Сайт має відведену сторінку для "статистики/мемів". Вона відображає кількість книг у вашій бібліотеці, статистику за оригінальними мовами (наприклад, кількість книг, які спочатку були написані французькою, іспанською, давньогрецькою тощо), стовпчикову діаграму середніх оцінок, стовпчикові діаграми років видання вашої бібліотеки та інше. Також вона відображає всі серії у вашій бібліотеці й показує, чи є ваша серія "завершеною". На сторінці статистики також доступні такі можливості [10]:

* можливість перегляду книг на основі подій у них, персонажів, місць. Наприклад, можна знайти книги, які містять події, що відбуваються в певному місці, такому як "Аляска".
* можливість перегляду статистики за авторами, таку як статистика за статевою приналежністю, статусом життя (живий/померлий) та національності, розкладеної за секторами кругової діаграми.

На сайті спостерігається значне використання текстового матеріалу, що стосується як домашньої сторінки, так і інших сторінок ресурсу. Цей контент охоплює різноманітні аспекти, такі як презентація авторів, інформація для рецензентів, розіграші та інше. Проте для деяких користувачів ці дані можуть бути неактуальними та відволікати увагу від основного завдання. Крім того, звернення до текстового вмісту без належного оформлення може створювати враження застарілості та неадаптованості до сучасних стандартів дизайну [10].

Хоча існують можливості для налаштування домашньої сторінки, слід враховувати, що процес цього налаштування може вимагати часу та зусиль від користувача. Тому, незважаючи на наявність таких можливостей, складність використання вебсайту може впливати на загальний досвід користувача та вимагати додаткової уваги з боку користувача для його оптимізації [10].

Система сортування відгуків на платформі LibraryThing відкриває користувачам можливості здійснення більш ефективного пошуку та оцінки літературних творів. Можливість сортування за мовою дозволяє зосередитися на відгуках у власній мові, сприяючи зручності взаємодії з платформою для користувачів з різних культурних та мовних середовищ. Крім того, функція сортування за часом публікації відгуків надає можливість користувачам отримувати актуальну інформацію про нові літературні твори та тренди [11].

Сортування відгуків за кількістю "голосів" дозволяє виділити найбільш важливі та впливові відгуки, що сприяє збагаченню користувачів авторитетною інформацією. З урахуванням меншої кількості користувачів на платформі, відгуки можуть бути менш масовими, проте це може забезпечити більшу увагу до кожного відгуку та більш індивідуалізований підхід до оцінки книги [11].

Можливість встановлення півзіркових оцінок відображає важливість точності та деталізації висловлення вражень від прочитаного, що може бути вирішальним у виборі подальшого читання. Також варто відзначити, що відгуки на платформі LibraryThing відомі своєю об'єктивністю та стильовою збалансованістю, що може забезпечити користувачам достовірну та повну інформацію при прийнятті рішення про вибір книг для читання [11].

LibraryThing – це інструмент для серйозних користувачів, які прагнуть мати докладний, об'єктивний та розгорнутий каталог книг. Обговорення та групи на платформі відзначаються вищим рівнем інтелектуальності, і взаємодія з авторами менш виражена, що може розглядатися як позитивний або негативний аспект, залежно від вподобань користувача. LibraryThing сприймається як інструмент для читачів, а не як платформа для реклами та просування видавців і авторів. Його інтерфейс менш інтуїтивний, менш привабливий з естетичної точки зору та менш зручний для користування порівняно з іншими аналогічними сервісами. Проте, його система статистики та тегів є більш розвиненою. LibraryThing розроблено для тих, хто надає перевагу розміщенню своєї фізичної бібліотеки на перше місце, а спілкування з іншими читачами - на друге [11].

На рис. 1.6 зображено інтерфейс вебсайту LibraryThing.

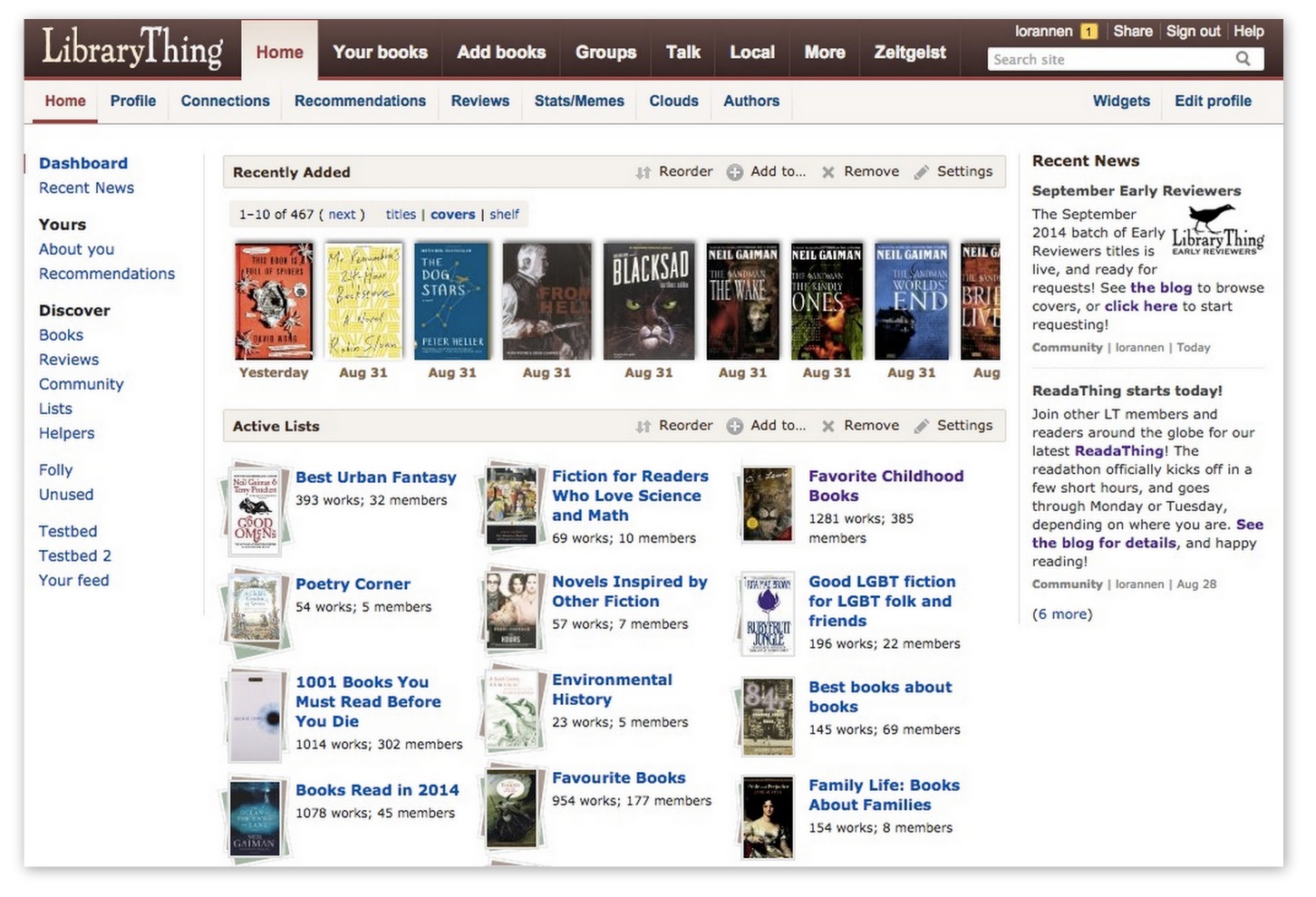


Рисунок 1.6 – Інтерфейс вебсайту LibraryThing [12]

В заключення, порівняємо три розглянуті сервіси, випишемо їх функціональні та інші параметри і спробуємо звести їх у порівняльну таблицю 1.1.

Виходячи з таблиці можна зробити висновки, що Goodreads представляє собою найпопулярніший та найзвичніший варіант з середнім функціоналом та оформленням. The StoryGraph є наймолодшим та найсучаснішим з усіх, втілює креативні нові ідеї те з’являється у новітньому легкому інтерфейсі, система настроїв книг виділяється з поміж кандидатів. LibraryThing, хоч і є менш популярним за інших, має застарілий дизайн, незрозумілий інтерфейс, але представляє найширший функціонал з усіх, а також дає доступ до оцінки різного виду контенту.

Таблиця 1.1 – Порівняння альтернатив.

|  |  |  |  |
| --- | --- | --- | --- |
| **Параметр** | **Goodreads** | **The StoryGraph** | **LibraryThing** |
| **1** | **2** | **3** | **4** |
| Інтерфейс | Класичне стримане оформлення, але є простір для модернізації. | Сучасний, не навантажений | Застарілий, місцями не має оформлення |
| Оцінювання | П’ятизірковий варіант оцінювання, що може сковувати читачів. | Оцінювання п’ятизіркове, але є можливість ставити половину та чверть зірки. | П’ятизіркове оцінювання з можливістю ставити половину зірки. |
| Фільтрування | Пошук не має фільтрів, але після введення запиту можна буде обрати за яким критерієм шукати (книги, групи, цитати, люди, назва або автор). | Пропонуються фільтри за настроєм книги, її темпом, переліком жанрів, трьома типами розмірів та датою публікації. | Пошук не має початкових фільтрів, але результати запиту можна відфільтрувати за типом речі, що шукають. |
| Додаткові можливості пошуку | Після введення запиту, пропонуються певні результати на нього, а також, на основі цих результатів, користувачу пропонуються полиці жанрів, де він може знайти схожі роботи. | Додаткових можливостей немає, але це компенсується багатими фільтрами і великою системою настрою книг. | Сервіс має унікальну систему тегів, які кожен користувач сам додає до кожної книги. Є можливість шукати книги за цими тегами. |
| Статистика | Надається статистика про середній рейтинг прочитаних книг, кількість книг та сторінок на рік. | Надається велика кількість статистики у вигляді діаграм, а саме настроїв книг, темпу, кількості сторінок, типу книг, жанрів, форматів, авторів, мовам, графіку кількості книг та сторінок та оцінкам | Надається широка статистика з діаграмами по типу книг, жанрам, інформації про авторів (живі/мертві, стать), власній оцінці, мові, обкладинці та її кольору. |
| Рецензія | Надається можливість поставити оцінку, написати ємну рецензію та призначити книгу на певні власні полички за категоріями. Рецензії можна сортувати та фільтрувати за мовами. | Пропонує коротку вижимку думок читача, його оцінку. Також читач проходить коротке опитування для того, щоб сформувати категорії під які підпадає книга, її настрої. | Користувачі не обмежені в обсязі рецензії, призначається оцінка та користувач обирає теги до книги. Рецензії можна сортувати та фільтрувати за мовами. |
| Додавання нових книг | Потрібно створювати запит на додавання книги, якщо її неможливо зайти на сервісі. | Є можливість ручного додавання книги, яку потім можуть затвердити. | Є можливість додати книгу за номером або вручну, якщо він відсутній. |
| Тип контенту | Книги | Книги | Переважно книги, але є можливість додавати та шукати відео та аудіо, фільми та пісні. |

## 1.3 Постановка завдання

Після вивчення предметної області та аналізу наявних успішних реалізацій для вирішення її проблем, сформулюємо технічне завдання.

Метою цієї роботи є створення програмного забезпечення у вигляді вебзастосунку, основною ціллю якого буде спрощення процесу відслідковування списку читання користувача, допомога в його структуризації та підбитті підсумків читання.

Основними функціональними вимогами до проєкту є:

* додавання нових книг до читацької бази;
* додавання та редагування будь-якої інформації про створені книги;
* розподілення книг за списками, які позначають поточний стан її читання;
* пошук книг у списках за назвою, автором або описом;
* фільтр пошуку за жанрами або тегами, які міститимуть книги;
* додавання рецензії на книгу, можливість редагування рецензії;
* можливість оцінити книгу при перенесенні її до стану прочитаних.

Проєкт повинен бути розроблений з можливістю подальшої модернізації та масштабування.

При створенні інтерфейсу важливо врахувати потреби користувача і переконатися, що користувачеві комфортно працювати з інтерфейсом. Це означає, що інтерфейс повинен бути не тільки простим і зрозумілим, але й ефективним. Він повинен дозволяти користувачам легко виконувати необхідні дії, не заважаючи їм зосередитися на основній меті використання програми або системи.

Занадто простий інтерфейс може справляти враження застарілого або не мати достатньої функціональності. Тому важливо знайти баланс між простотою і достатнім рівнем функціональності та дизайну, щоб він виглядав сучасним і привабливим для користувачів.

Однак також важливо не перевантажувати інтерфейс додатковими деталями, які можуть відволікати увагу користувача від основного завдання. Чистий і простий дизайн допомагає користувачеві зосередитися на важливих діях та інформації.

Проєкт у вигляді вебзастосунку має бути представлений у вигляді HTML-сторінки, оформлений за допомогою CSS. Функціонал застосунку має бути реалізований у скрипт-файлах мовою JavaScript з використанням бібліотеки React і її інструментів. Конкретніше, компоненти на HTML-сторінках будуть створюватися мовою JSX, що поєднує в собі об’єкти, які відображуються та логічні або функціональні елементи Для створення та розгортання проєкту повинен використовуватись пакетний менеджер npm, який вимагає встановлення середовища виконання Node.js.

Технічними вимогами для підтримання проєкту є:

* операційна система Windows 10;
* процесор Intel Core I7-3770;
* ОЗП ємністю 16 ГБ;
* будь-який вебпереглядач на основі Chromium версії 124;
* npm версії 10.5.0;
* node версії 20.12.2;
* редактор початкового коду Visual Studio Code.

Працездатність проєкту, можливість його покращення гарантується на заявлених технічних вимогах, але цілком можлива з іншими складовими, но новіших або старіших версіях та системах.

У склад програмної документації повинні входити такі елементи як керівництва розробника та користувача, опис програмного забезпечення та описи основних програмних модулів, тексти реалізації цих модулів.

Тестування повинно бути виконано з оглядом на те, що всі сценарії, які будуть надані користувачу в керівництві повинні виконуватись без нарікань з боку застосунку. Будь які помилки, які користувач може допустити під час цих сценаріїв повинні зрозуміло пояснюватися користувачу застосунком.

## 1.4 Висновки з розділу

У цьому розділі була розглянута предметна область розроблюваного проєкту, а також проблеми, які має на меті вирішити цей проєкт. Досліджено актуальність даної роботи як у минулому, так і у майбутньому.

Для більш глибокого розуміння контексту були розглянуті три аналогічні, успішні проєкти. Проведений аналіз їх функціоналу, виділено сильні та слабкі сторони кожного з них. В результаті цього дослідження отримано базове розуміння для подальшого розвитку та вдосконалення розроблюваного проєкту.

Було створено технічне завдання до проєкту, в якому описано основні функціональні можливості, які повинні бути реалізовані в проєкті, вимоги до інтерфейсу, складові компоненти та програмне забезпечення, яке необхідно для редагування проєкту, його подальшої модифікації, вказані системні специфікації з якими створюється проєкт, описані складові програмної документації.

# 2 РОЗРОБКА АРХІТЕКТУРИ ПРОЄКТУ

## 2.1 Вибір інструментів розробки

Як мову розробки функціональної частини вебзастосунку було обрано JavaScript.

JavaScript, у своєму статусі високорівневої скриптової мови програмування, широко використовується у веброзробці для створення динамічних інтерактивних вебзастосунків і сайтів. Завдяки своїй гнучкості і потужності, JavaScript забезпечує функціональність і зручність вебсторінок для користувачів [13].

Основні застосування JavaScript включають анімацію, яка прикрашає вебсторінки та забезпечує їх привабливість для відвідувачів, а також валідацію форм, яка гарантує правильність введених даних користувачем. Більш того, JavaScript може динамічно оновлювати вміст сторінки без перезавантаження, забезпечуючи плавний і неперервний перегляд [13].

Ще однією ключовою можливістю JavaScript є можливість реагувати на дії користувача, такі як наведення курсору миші на елемент або натискання на кнопку. Це дозволяє створювати інтерактивні елементи, які взаємодіють з користувачем, змінюючи свій вигляд або поведінку у реальному часі [13].

За опитуваннями розробників, JavaScript є не тільки найпопулярнішою мовою для веброзробки, але найпопулярнішою мовою в цілому серед всіх мов програмування. Це підтверджує дослідження Лайонела Вейлшері, що охоплює тренди мов програмування на 2023 рік. Статистика мов зі статті зображена на рис. 2.1.

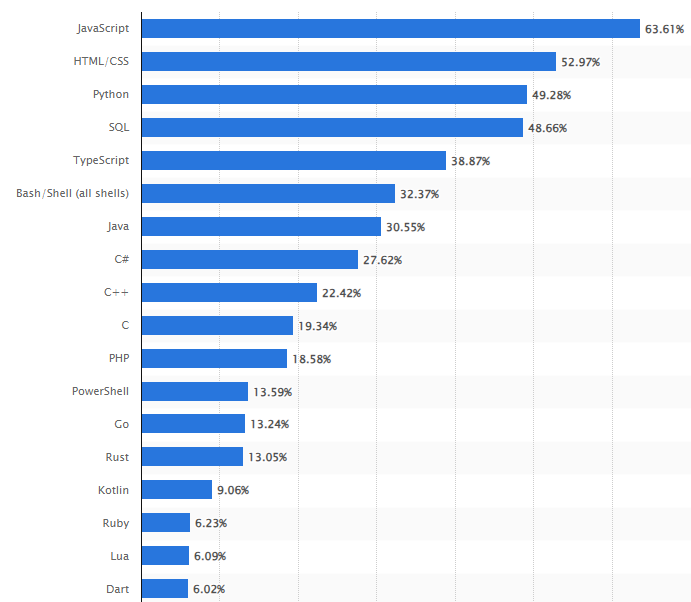


Рисунок 2.1 – Найпоширеніші мови програмування серед розробників у всьому світі станом на 2023 рік [14]

Діаграма показує, що майже 64% опитуваних зазначили JavaScript, як свою основну мову програмування.

Якщо не брати до уваги мови HTML/CSS, що прийнято вважати мовами розмітки, а не програмування, а також Python, який, хоч і використовується для веброзробки, найбільше застосовується в датааналітиці, то найпопулярнішою мовою після JavaScript у веброзробці є TypeScript з майже 39% відповідей серед опитуваних.

Мова TypeScript є відкритою мовою програмування, розробленою для створення великих і складних застосунків. TypeScript був створений одним із великих технологічних гігантів – компанією Microsoft – у 2012 році. Основною метою створення TypeScript було полегшення розробки та підтримки великих проєктів програмного забезпечення [15].

Основна особливість TypeScript полягає в тому, що він є синтаксичним розширенням мови JavaScript, але при цьому додає статичну типізацію. Це дозволяє розробникам забезпечити більшу стабільність та безпеку великих програм, виявляючи помилки на етапі компіляції, що сприяє вирішенню проблем на ранніх етапах розробки [15].

Однією з ключових переваг TypeScript є можливість використання сучасних функціональних можливостей, таких як асинхронне програмування та робота з промісами, що спрощує розробку складних асинхронних застосунків [15].

Завдяки поєднанню можливостей JavaScript та переваг статичної типізації, TypeScript став популярним інструментом серед розробників для створення надійних та масштабованих програмних продуктів [15].

Порівнюючи TypeScript з JavaScript, варто зазначити, що TypeScript надає всі можливості JavaScript, а також додаткові можливості та функції, які полегшують розробку складних програмних продуктів. Оскільки TypeScript базується на JavaScript, весь коректний код JavaScript також є коректним у TypeScript, що робить TypeScript надмножиною JavaScript [15].

Можливість інтеграції з існуючим кодом JavaScript робить TypeScript привабливим варіантом для розширення функціональності існуючих проєктів. Крім того, TypeScript має додаткову функцію статичної типізації, що дозволяє виявляти помилки на етапі розробки, підвищуючи надійність та підтримку програмного забезпечення [15].

З використанням TypeScript розробник отримує ряд переваг. Уникнення прихованих помилок, таких як помилка "undefined" не є функцією. Код може бути перероблений зі збереженням його функціональності. Орієнтація в складних системах більше не є проблемою. За результатами досліджень, TypeScript виявляє понад 15% всіх помилок у JavaScript [15].

Динамічне типування часто спричиняє помилки, що знижують ефективність розробника та можуть ускладнити розробку через збільшення витрат на додавання нового коду [15].

Отже, через відсутність типів та перевірок помилок на етапі компіляції, JavaScript може бути невдалим вибором для серверного коду в організаціях та великих кодових базах [15].

TypeScript містить різні базові типи, такі як число, масив, кортеж, булевий тип, рядок і багато інших. Деякі з цих типів недоступні в JavaScript [15].

TypeScript підтримує як неявні, так і явні типи. Якщо ви не пишете типи явно, компілятор увімкне виведення типів, щоб визначити типи, які ви використовуєте [15].

З іншого боку, написання їх явно надає переваги, такі як допомога іншим членам команди, які читають ваш код, і перевірка того, що те, що бачите ви, відповідає тому, що бачить компілятор [15].

Перевірка типів під час компіляції. У ванільному JavaScript перевірка типів виконується під час виконання. Однак це призводить до збільшення часу виконання, якого можна уникнути, виконуючи перевірку під час компіляції [15].

TypeScript чудово працює у великих проєктах або коли багато розробників працюють разом, завдяки своїм сильним системам типів, які допомагають у виявленні помилок на етапі компіляції і полегшують спільну роботу над кодом. Однак, у невеликих проєктах з невеликою кількістю коду використання TypeScript може бути зайвим, оскільки воно може ускладнити процес розробки, додавши додаткову складність та затримки у випуску продукту. У таких випадках, використання JavaScript може бути більш виправданим, дозволяючи швидше розробляти та випускати програмне забезпечення [15].

Зведемо інформацію про ці мови програмування у таблицю 2.1, порівняємо їх параметри.

Таблиця 2.1 – Порівняння JavaScript та TypeScript [15]

|  |  |  |
| --- | --- | --- |
| **Параметр** | **JavaScript** | **TypeScript** |
| **1** | **2** | **3** |
| Типізація | Сильна типізація. TypeScript підтримує як статичну, так і динамічну типізацію. | Слабка типізація. JavaScript підтримує лише динамічну типізацію. |
| Компіляція | TypeScript потребує компіляції. | JavaScript не потребує компіляції. |
| Прив'язка даних | TypeScript використовує такі поняття, як інтерфейси та типи для визначення даних, що використовуються. | У JavaScript таких понять немає. |
| Крива навчання | TypeScript має жорстку криву навчання. Крім того, він потребує знань зі створення сценаріїв. | Це проста у вивченні і гнучка мова для написання вебскриптів. |
| Спільнота | TypeScript має меншу спільноту розробників програмного забезпечення. | JavaScript має велику спільноту розробників програмного забезпечення. |
| Створення прототипів | Функція прототипування доступна в TypeScript. | JS не підтримує прототипування. |

Використання JavaScript для дипломного проєкту має свої переваги. Перш за все, це відповідає розміру проєкту. Для невеликого обсягу роботи, яким зазвичай є дипломна робота, TypeScript може виявитися занадто потужним і складним в управлінні. JavaScript, з іншого боку, забезпечує простоту і швидкість розробки завдяки своїй прямій і лаконічній синтаксичній структурі.

Крім того, використання JavaScript дозволяє використовувати широкий спектр доступних бібліотек і фреймворків. Це дає можливість розширити функціональність проєкту і використовувати вже наявні рішення для різноманітних задач, що спрощує процес розробки і зменшує час витрачений на написання коду.

Не останнім чинником є широка підтримка JavaScript у спільноті розробників. Завдяки своїй популярності, JavaScript має велику кількість активних розробників, які можуть надати підтримку, поради та відповіді на питання, що виникають під час роботи над проєктом.

Для створення вебзастосунку, обрання однієї лише мови програмування недостатньо для досягнення задовільного результату. Велику долю розробки займає також фреймворк, на якому побудовано застосунок.

В якості фреймворку для нашого проєкту обираємо React, який хоч формально і є бібліотекою до JavaScript, має функціонал дуже близький до альтернативних фреймворків і є найпопулярнішим варіантом для веброзробки.

React — JavaScript-бібліотека, яка застосовується для розробки UI (призначених для користувача інтерфейсів), та відповідає за те, як ваш Web-застосунок буде виглядати. У реакті відкритий вихідний код, створений корпорацією Facebook, яка активно його розвиває разом зі спільнотою. Зокрема, бібліотека використовується в React Native, технології для створення нативних та гібридних застосунків для iOS та Android [16].

React, як гнучка декларативна бібліотека, ніяк не обмежує вас у дизайні Web-застосунку. Якщо вам потрібно створити унікальний користувальницький інтерфейс, то реакт — найкращий вибір. Розробникам потрібно буде тільки описати, як кожна частина інтерфейсу виглядає в різних станах, а бібліотека буде міняти їх при отриманні потрібної команди [16].

React робить сайт не тільки красивим, але і продуктивним. За це відповідає віртуальний DOM, важлива перевага бібліотеки. З ним сайт набуває можливість оновлювати тільки ті частини сторінки, які потрібно змінити. У пам’яті Web-застосунки зберігається структура попередньої версії, що дає змогу порівняти її з новим станом інтерфейсу. Так само важливий рендеринг на стороні сервера. На практиці, швидкість завантаження компонентів вашого Web-застосунку не буде залежати від потужності того пристрою, на якому користувач його відкрив. До того ж Реакт підходить для деяких нестандартних завдань, крім перетворення HTML-коду в зображення в браузері. З його допомогою можна зображати динамічні графіки або зробити застосунок ізоморфним — щоби сервер міг рендерити сторінки для первинного завантаження й користувач бачив контент, а не заставку «Завантаження» [16].

React можна використовувати в Web-застосунках будь-якого масштабу, але саме в невеликих односторінкових застосунках із перспективою росту він показує себе максимально добре. Зокрема, для стартапів його вибирають через легку окупність. До того ж сайти з React високо оцінюються пошуковими ботами Google. Їм, як і користувачам, подобається зручний інтерфейс і досвід, який він надає [16].

Та все ж є вагомі причини, чому саме React обирають для створення вебзастосунків.

JavaScript XML — розширений синтаксис, із яким працювати простіше, а знаходити помилки легше. Для розробників які знають JavaScript і HTML, а це легкі та популярні мови, об’єднання їх в JSX робить компоненти простими для розуміння, а отже, ніяких проблем із доопрацюванням, тестуванням і підтримкою [16].

Код, написаний з React, просто тестувати та можна використовувати повторно. Декларативні подання (одностороння прив’язка даних) роблять його максимально передбачуваним, у порівнянні з іншими фреймворками, і зменшують кількість помилок під час налагодження. При цьому, розробники можуть створювати інкапсульовані компоненти з власним станом і об’єднувати їх у складні інтерфейси. Логіка JavaScript не обмежена шаблонами - дає змогу легко передавати дані між компонентами з будь-якої частини програми [16].

Творці реакта дають розробникам використовувати рівно стільки можливостей бібліотеки, скільки потрібно, і там, де це потрібно. React можна використовувати на сервері, наприклад з Node.js, але також і для розробки мобільних застосунків з React Native. Крім того, можна просто додати кілька динамічних віджетів на HTML-сторінку, щоби зробити інтерфейс більш «живим» і сучасним. Якщо в майбутньому захочеться використовувати більше можливостей React, розширення його присутності не потребує багато часу й серйозних змін у коді сторінки [16].

React є найпопулярнішим JavaScript-фрейморків. За опитуванням, його обирає понад 80% відсотків розробників. На рис. 2.2 зображено найпоширеніші JavaScript-фреймворки станом на 2022 рік.

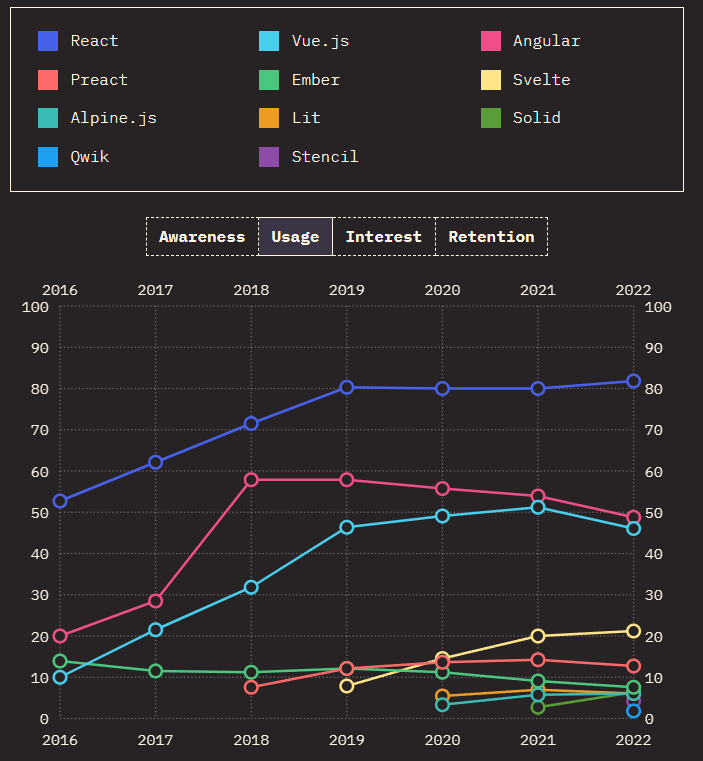


Рисунок 2.2 – Найпоширеніші JavaScript-фреймворки станом на 2022 рік [17]

З рисунку можна побачити, що основним конкурентом React є фреймворк Angular, який має популярність в районі 50%. Angular пропонує часто схожі, а інколи і зовсім інші рішення при створенні вебсайтів.

Говорячи простою мовою, Angular це інструмент для розробки інтерактивних вебсайтів. Він являє собою потужний фронтенд фреймворк і пропонує розробникам всі необхідні компоненти для швидкої побудови складних сайтів і вебдодатків [18].

Angular був винайдений розробником Mishko Hevery, який на той момент працював у Google. Він подумав, що знайшов хороший спосіб допомогти вебдизайнерам швидко створювати прототипи додатків використовуючи HTML. Він поринув у роботу і через якийсь час у нього вийшло те, що зараз називають AngularJS. Керівництво компанії Google побачило потенціал цієї ідеї, і вона стала офіційним проєктом компанії і до роботи над ним підключилася велика кількість веброзробників [18].

AngularJS швидко став дуже популярним інструментом і на ньому розроблено безліч сайтів. Але ще раніше, ніж фреймворк досяг свого піку популярності веброзробники, які працюють над ним, зрозуміли, що він потребує істотних змін, так з’явився Angular 2. Головні відмінності від першої версії, це висока стандартизація, використання більш сучасних [18].

Angular має дуже високу відповідність стандартам, він підтримує стандарт JavaScript ES6 та модулі. Розробка багатомовних і доступних сайтів використовуючи цей фреймворк легка і швидка [18].

Однією з головних переваг Angular перед аналогічними інструментами такого роду є його продуктивність. Можливо, він більш складний у вивченні і поступається чимось іншим, але продуктивність поза конкуренцією [18].

Популярність Angular серед інженерів стабільно тримається на високому рівні з його виникнення. Це означає, що є багато фахівців, які знають цей фреймворк. Його прямими конкурентами є React і Vue, яким Angular нічим не поступається, а в чомусь перевершує [18].

Один із ключових аспектів, який слід враховувати під час порівняння Angular і React, – це ступінь їхньої складності. Angular, будучи повноцінним фреймворком, має крутішу криву навчання, особливо для новачків у веброзробці. У той час як React, зосереджений на компонентах, надає більш плавний старт для нових розробників [19].

Angular є повноцінним фреймворком, в той час як React – це бібліотека для створення інтерфейсів. Це вказує на різницю у підходах до розробки: Angular надає більше готових рішень, в той час як React дає більшу свободу вибору інструментів [19].

Щодо двостороннього зв’язування даних, Angular пропонує автоматичну синхронізацію між даними інтерфейсу та коду. У React такого немає, і для забезпечення такого зв’язування потрібно використовувати додаткові інструменти [19].

Обидва фреймворки базуються на компонентному підході до розробки, де компоненти використовуються для створення користувацьких елементів. Проте у Angular компоненти відіграють важливу роль в організації структури додатка, тоді як у React компоненти стають основою для створення користувацького інтерфейсу з використанням JSX [19].

Щодо початку роботи, Angular може вважатися більш складним у освоєнні через потрібність налаштування та велику кількість концепцій. У React стартувати простіше, адже JSX дозволяє швидко створювати компоненти [19].

Екосистема Angular включає великий набір інструментів і бібліотек для різних завдань. У React також є розширена екосистема, але деякі функціональності можуть вимагати використання додаткових бібліотек [19].

Що стосується роутінгу та управління станом, Angular надає вбудовані інструменти для маршрутизації та управління станом. У React для цих завдань часто використовують сторонні бібліотеки [19].

На останок, розмір і продуктивність також різняться. Angular зазвичай має більший розмір через включені інструменти, але його продуктивність хороша, особливо у великих додатках. У React розмір менший, і фреймворк працює швидко та плавно, зокрема на невеликих і середніх проєктах [19].

Як бачимо, у кожного з фреймворків є свої відмінності та неповторний стиль. Angular – потужний і повноцінний, хороший для великих проєктів і досвідчених команд. React – легкий, гнучкий і швидкий, ідеально підходить для менших і середніх проєктів, а також для команд, які вважають за краще працювати з компонентами [19].

Зведемо розглянуті параметри у порівняльну таблицю 2.2.

Таблиця 2.2 – Порівняння Angular та React [19]

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Angular** | **React** |
| **1** | **2** | **3** |
| Тип | Фреймворк | Бібліотека |
| Двостороннє зв’язування даних | Автоматична синхронізація даних інтерфейсу та коду | Необхідно використовувати додаткові інструменти |
| Компонентний підхід | Використовуються компоненти для створення користувацьких елементів | Компоненти – основа створення користувацького інтерфейсу |
| Складність початку роботи | Потрібен певний час для освоєння і налаштування | Простий старт і швидке створення компонентів з JSX |
| Розширена екосистема | Великий набір інструментів і бібліотек для різних завдань | Деякі функціональності вимагають додаткових бібліотек |
| Роутінг і управління станом | Вбудовані інструменти для маршрутизації та управління станом | Потрібно використовувати сторонні бібліотеки для роутингу та управління станом |
| Розмір і продуктивність | Більший розмір через включені інструменти, хороша продуктивність у великих додатках | Менший розмір, швидка і плавна робота, особливо на невеликих і середніх проєктах |
| Суспільство та підтримка | Велика й активна спільнота розробників і хороша підтримка | Величезна й активна спільнота, безліч рішень і допомоги |

Отже, у підсумку можна сказати, що React – найпопулярніша бібліотека JavaScript. Опитування розробників Stack Overflow, статистика пошуку Google, GitHub, усе показує постійне зростання попиту на саму технологію й розробників. Гнучкий, легкий і продуктивний React використовується Facebook та багатьма іншими відомими компаніями від Netflix, де він адаптує контент під телевізори з невеликої продуктивності, до Instagram, де відповідає, навіть за геолокацію. [16]

Пакетний менеджер npm та виконувач пакетів npx використовуватимуться для початкової генерації та подальшої побудови вебзастосунку, підключенні бібліотек до нього, тестуванні. Їх встановлення відбувається разом із середовищем виконання Node.js.

Node.js - це потужна платформа, яка протягом кількох років змінила уявлення про розробку вебдодатків. Її ключова ідея полягає в тому, щоб зробити вже популярний JavaScript мовою загального призначення, придатною як для фронтенду, так і для бекенду [20].

Основний принцип Node.js – модель асинхронного програмування. Це означає, що розробники можуть створювати ефективні та чуйні додатки, спроможні одночасно обробляти безліч запитів. Node.js також відомий своєю легкістю та швидкістю роботи, що дозволяє ефективно використовувати ресурси сервера [20].

Node.js став однією з найпопулярніших платформ для створення вебсерверів, API, мікросервісів та багато іншого [20].

Однією з головних особливостей Node.js є асинхронна та подієво-орієнтована модель програмування. Замість того, щоб блокувати виконання коду під час очікування операцій введення-виведення, Node.js використовує неблокувальні операції, дозволяючи додаткам одночасно обробляти безліч запитів. Це робить Node.js дуже ефективним для створення масштабованих та чуйних серверних додатків [20].

Якщо говорити про переваги Node.js, то варто згадати його екосистему модулів, яка представлена вбудованим менеджером пакетів npm. Npm — це один з найбільших репозиторіїв програмного забезпечення, де розробники можуть знайти та використовувати тисячі готових модулів та бібліотек для прискорення процесу розробки [20].

Node.js також чудово інтегрується з іншими технологіями веброзробки, на кшталт Express.js — популярним фреймворком для створення вебдодатків на Node.js. Express.js забезпечує набір інструментів та функціонал, які роблять створення вебдодатків швидким та зручним [20].

Проте Node.js не обмежується виключно розробкою вебсерверів. Можна написати на Node.js різноманітні додатки, включно з мережевими серверами, мікросервісами, інструментами командного рядка й навіть настільними додатками. Завдяки його гнучкості та потужним можливостям, Node.js став однією з найпопулярніших платформ для розробки серверної частини додатків [20].

Node.js має низку унікальних характеристик, які роблять його привабливим для розробників. Ось деякі з ключових особливостей Node.js:

Висока швидкість виконання. Node.js заснований на рушії JavaScript V8 від Google, який забезпечує швидку та ефективну обробку коду. Це дозволяє Node.js виконувати операції швидко, що особливо важливо для створення високопродуктивних серверних додатків [20].

Універсальність та гнучкість. Node.js дозволяє використовувати JavaScript як мову програмування для розробки серверних та клієнтських додатків. Це дозволяє розробникам використовувати одну мову та ділитися кодом між серверною та клієнтською стороною. Завдяки цій універсальності, Node.js полегшує розробку повноцінних вебдодатків та спрощує командну взаємодію [20].

Безліч модулів та бібліотек. Node.js має велику екосистему модулів та бібліотек, доступних через вбудований менеджер пакетів npm. Розробники можуть використовувати ці модулі та бібліотеки для розширення функціональності своїх додатків та прискорення розробки. Завдяки величезній спільноті розробників, яка підтримує npm, можна знайти рішення практично для будь-якого завдання [20].

Сумісність з декількома платформами. Node.js може бути запущений на різних операційних системах, таких як Windows, MacOS та Linux. Це робить його універсальним та забезпечує розробникам свободу вибору платформи розгортання своїх додатків. Крім того, Node.js також сумісний з контейнеризацією та хмарними платформами, що спрощує розгортання та масштабування додатків [20].

Розглядання альтернатив Node.js не є необхідним у цій роботі, оскільки Node.js має занадто широке призначення, як для фронтенд, так і для бекенд розробки. Його альтернативами виступали б окремі мови та окремі фреймворки цих мов, головне призначення яких була б підтримка серверної частини вебзастосунка, що не є основною темою проєкту.

## 2.2 Створення структурної схеми вебзастосунку

Перш ніж перейти до активної фази розробки вебзастосунку, важливо створити каркас, базову основу, яка визначає його структуру та функціональність. Структурні схеми, які ще називають вайрфреймами, допомагають зрозуміти взаємозв'язки між різними елементами вебзастосунку та організувати їх у найкращий спосіб. Цей етап важливий, оскільки він закладає основу для подальшої роботи і дозволяє розробникам зосередитися і полегшити робочий процес. Структурні схеми діють як візуальна інтерпретація концепції вебзастосунку, забезпечуючи єдність у сприйнятті завдань і допомагаючи уникнути несподіваних труднощів на більш пізніх етапах розробки.

Вайрфрейм — це приблизний та навіть грубий ескіз структури виробу. Він дозволяє швидко отримати уявлення про те, як він буде працювати, не витрачаючи багато часу на його створення. Він дозволяє швидко зафіксувати концепцію та орієнтувати команду без значного витрати часу на детальне виконання. У суті, вайрфрейм є орієнтиром для розробників і схемою продукту — каркасом, на який потім буде навішано "м'ясо" [21].

Створення вайрфрейму передбачає визначення цілей, цільової аудиторії та аналіз конкурентів. Для простих проєктів цього може вистачити, але для складних необхідно мати технічне завдання з переліком потрібних функцій [21].

Вайрфрейм може бути представлений у формі карти шляху користувача для однієї або всіх існуючих сторінок, при цьому рівень деталізації може варіюватися. Наприклад, низькодеталізований вайрфрейм можна швидко намалювати навіть на етапі брейншторму, проте для його розуміння без додаткової пояснювальної презентації може знадобитися автор. З іншого боку, високодеталізований вайрфрейм вже є більш докладним планом з підписами та поясненнями, і може бути показаний клієнту без додаткових пояснень [21].

Час створення вайрфрейму залежить від його рівня деталізації: низькодеталізований можна створити за кілька хвилин, у той час як високодеталізований може зайняти 1-2 дні [21].

Створення вайрфрейму можна провести на папері з олівцем для низькодеталізованого варіанту, або в будь-якому графічному редакторі для високодеталізованого. На сьогоднішній день існують численні сервіси для створення вайрфреймів, такі як Ninjamock або Figma [21].

Для створення вайрфрейму використовуватимемо сервіс Ninjamock, що безкоштовно надає достатній функціонал для цілей цієї роботи.

Усі вайрфрейми повинні реалізовувати раніше оголошені вимоги до застосунку, тому при створенні каркасів необхідно звертати на це найбільше уваги.

Почнемо створення з головної сторінки застосунку. За ідеєю проєкту, книга, як об’єкт, з яким взаємодіє користувач може мати три стани, а саме прочитана, читається зараз, або її читання заплановано на майбутнє. У застосунку усі книги повинні бути першочергово розподілені за списками, кожен з яких відповідає одному зі станів. Доступ до кожного цих списків повинен знаходитись на головній сторінці.

Окрім цього, сторінка повинна містити поле, за допомогою якого можна було б здійснювати пошук книг у цих списках за назвою, або автором. Сторінка повинна містити фільтри для елементів, що були знайдені пошуком, основні з них це жанри, сторінки, оцінка та наявність рецензії на книгу. Фільтрів може бути більше та вони можуть змінитися в процесі роботи, але цих буде цілком достатньо для представлення функціоналу фільтрів в цілому.

На сторінці списків, окрім самого перегляду книг у списках, повинна бути можливість змінювати їх приналежність до конкретного списку, а також елемент інтерфейсу, який би дозволяв додавати нові.

Додамо на сторінку елементи, що виконували б усі вище зазначені функції. Результат зображено на рис. 2.3.

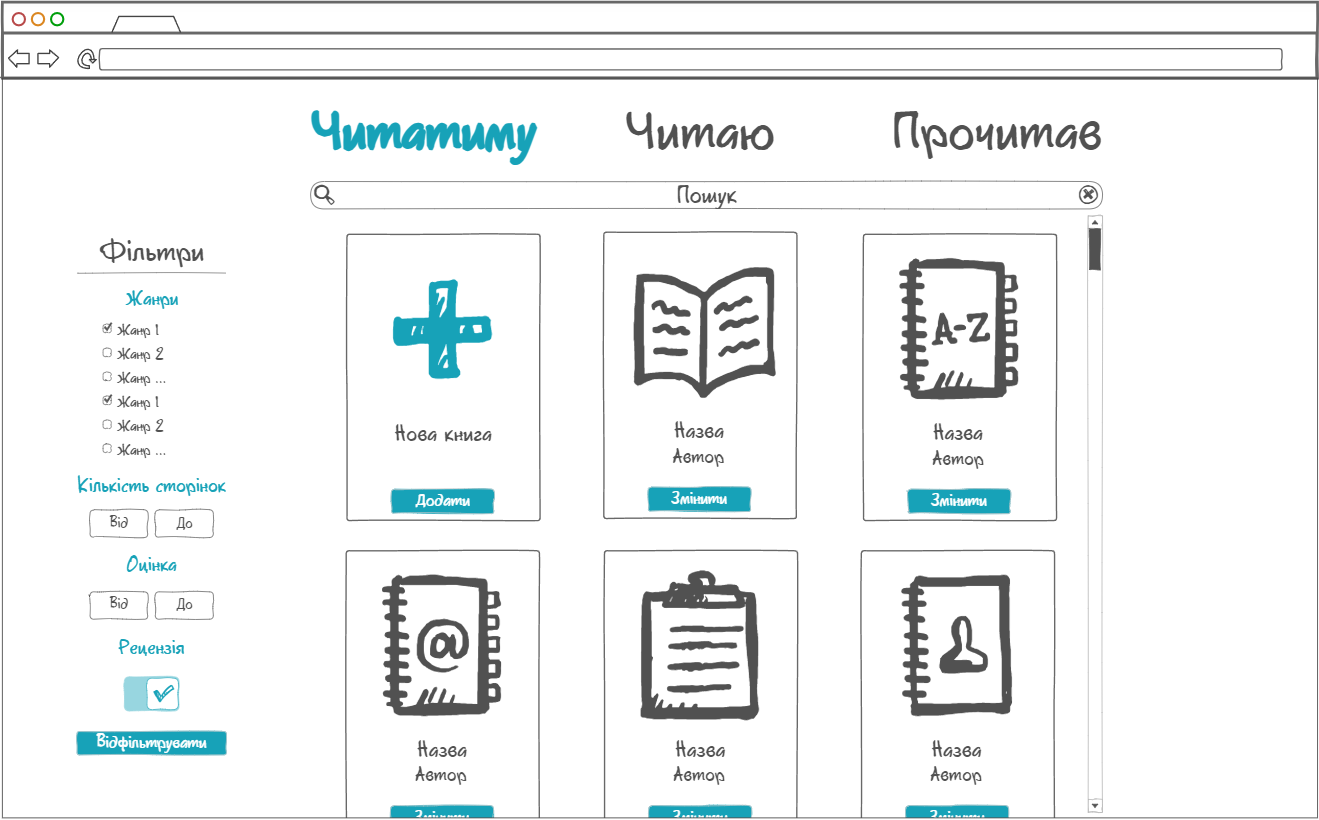


Рисунок 2.3 – Вайрфрейм сторінки списків

Після планування всього можливого функціоналу головної сторінки і додання його через елементи, необхідно спланувати інші сторінки, які б реалізовували незадіяний функціонал, що зазначено у вимогах, або логічно б продовжували вже описаний.

Наступною сторінкою, за таким принципом, створимо сторінку додання нової книги. Її ж можна буде використати пізніше для редагування інформації про вже існуючі книги.

Перехід на цю сторінку відбуватиметься, коли користувач захоче додати нову книгу, будучи на сторінці списків і натисне на елемент, який відповідає за створення.

На сторінці додавання нової книги для заповнення повинні бути доступні усі поля, які взагалі може мати книга як об’єкт, це і її назва, автор, жанри, які може містити книга, кількість сторінок, опис.

Надання книзі оцінки на етапі створення має сенс не в усіх випадках, бо користувач не має змоги оцінити книгу, яку він додає до списку тих, які він тільки бажає прочитати. Це поле є опціональним і доється для того випадку, якщо користувач захоче додати книгу яку він вже прочитав і хоче виставити їй оцінку без проведення декількох додаткових дій переходячи по різним сторінкам проєкту. Окрім того, якщо використовувати цю форму повторно в редагуванні вже існуючих книг, які вже можуть мати оцінку, це рішення має ще більше сенсу.

Відповідно до того, що книга може мати оцінку в зазначених випадках, на етапі додання нової книги одразу впроваджуємо зручне для майбутнього користувача рішення вибору списку до якого додається книга, замість того щоб повертатися до списків, змінювати бажаний список і додавати до нього.

Додавання зображення іде окремим блоком, де надається можливість завантажити зображення обкладинки, побачити його вигляд на сторінці, або видалити, якщо користувача не влаштовує результат, який він бачить.

Результат описаного функціоналу у вигляді об’єктів на сторінці наведено на рис. 2.4.

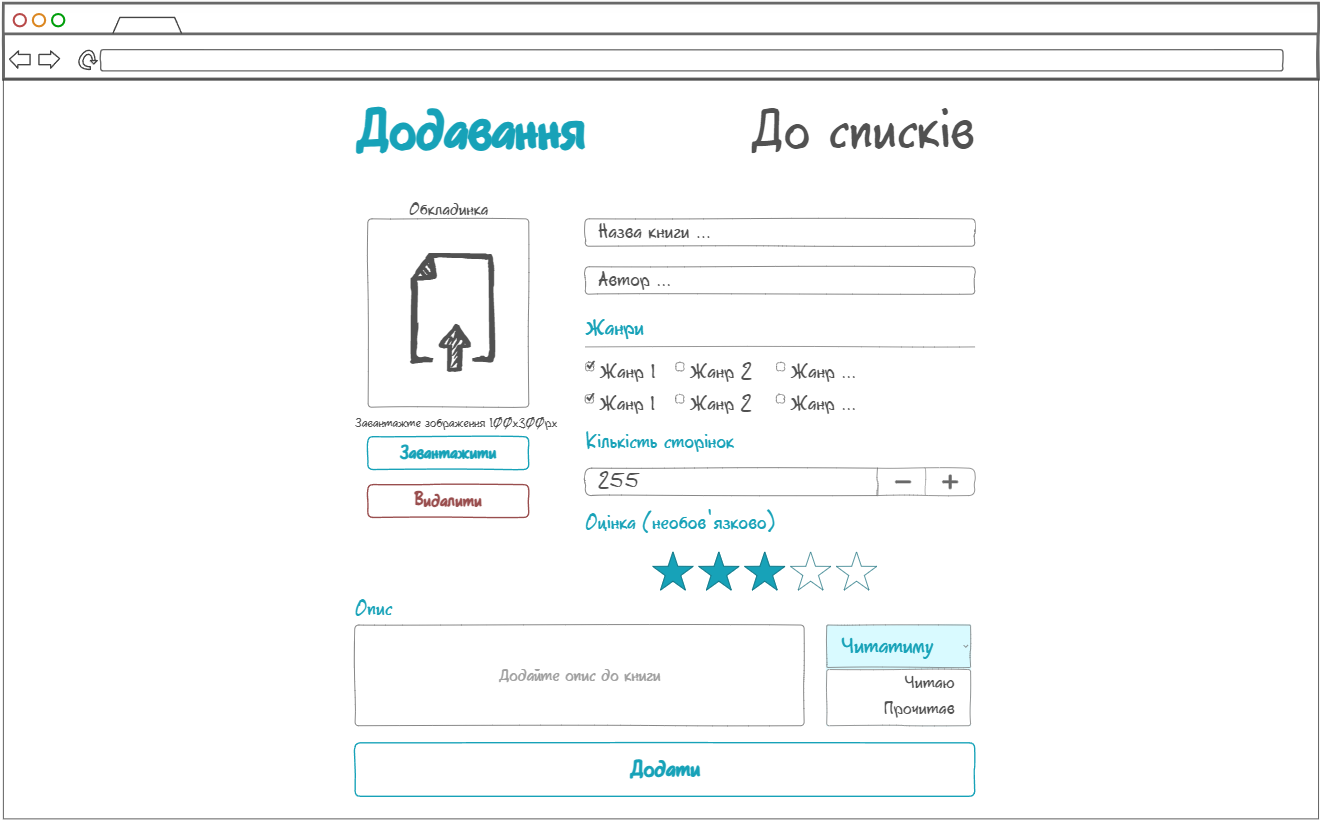


Рисунок 2.4 – Вайрфрейм сторінки нової книги

Повертаючись до головної сторінки, наступна взаємодія, яку може провести користувач – переглянути детальну інформацію про будь яку книгу зі списків.

На сторінці з інформацією про конкретну книгу повинна бути наявна вся інформація, під яку ми створювали поля під час проєктування сторінки створення нової книги.

Як і раніше, користувачу буде набагато зручніше завжди мати можливість змінювати приналежність книги до стану на будь-якій сторінці, де він бачить книгу або інформацію про неї, це прискорить взаємодію з застосунком. Додаємо селектор і на цю сторінку.

В тому ж блоці, що і селектор, додаємо не менш важливі функції редагування та видалення книги, що переглядається. Якщо з видаленням все зрозуміло, воно повинно прибирати усі записи про цю книгу в застосунку, то редагування, в свою чергу, повинно переносити користувача до сторінки створення книги, за тим винятком, що в цьому випадку ця сторінка буде сторінкою редагування і її поля вже будуть заповнені даними з книги, яку користувач обрав для редагування.

Після всіх полів, що вже були розглянуті, ця сторінка міститиме дещо нове поле – «Рецензія». Це поле не додається при створенні нової книги, а залишається порожнім, користувач взаємодіє з ним тільки на цій сторінці, а саме редагувати його вміст. Поле містить оцінку, яку можна було бачити і раніше, і текст, що містить думки користувача про книгу, а не її загальний опис.

Вайрфрейм побудованої сторінки зображено на рис. 2.5.

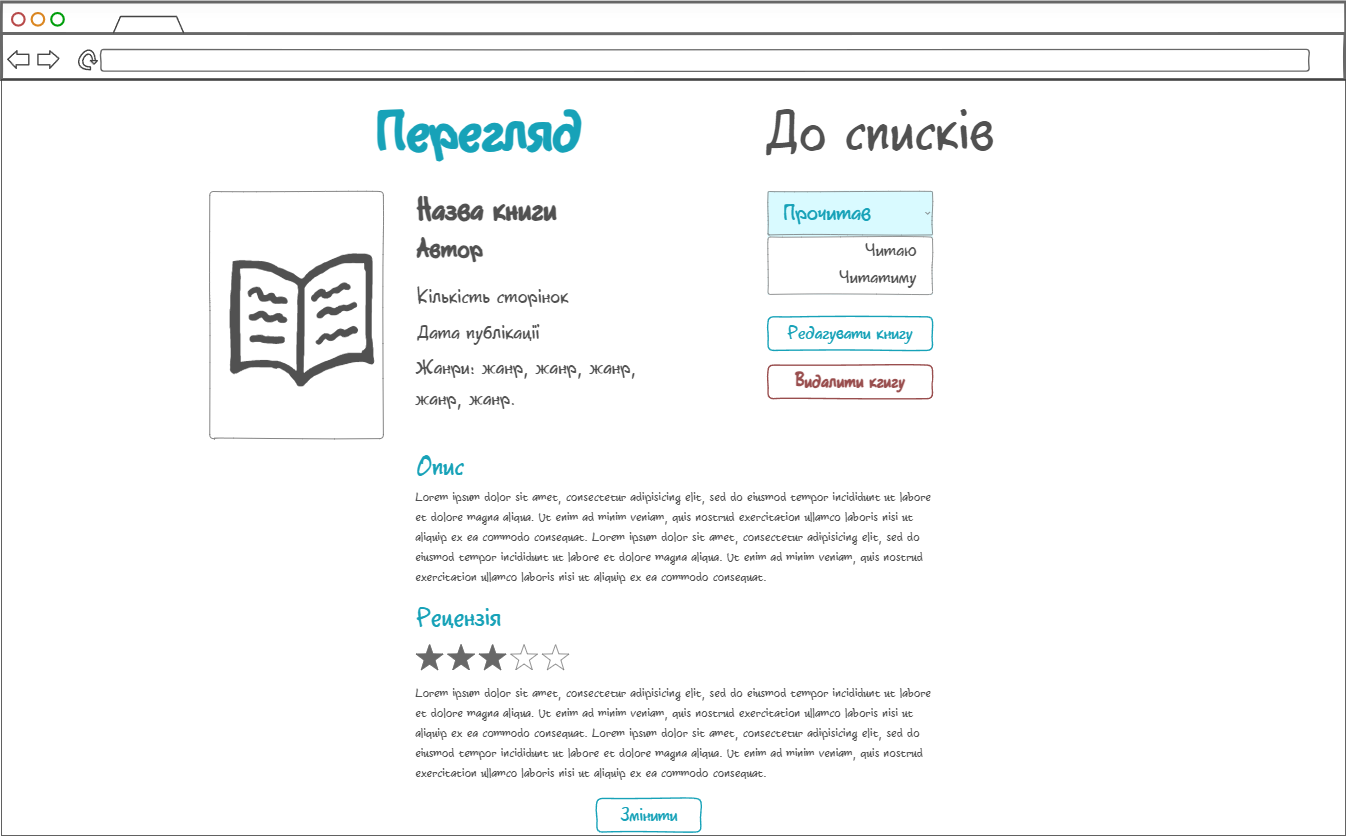


Рисунок 2.5 – Вайрфрейм сторінки інформації про книгу

Відходячи від головної сторінки, функціонал якої на даний момент було представлено на інших сторінках, що ще може змінитися вже на етапі не планування, а реалізації, переходимо до реалізації функціоналу інших сторінок, а саме сторінки інформації про книгу.

На сторінці перегляду інформації про конкретну книгу було описано таке поле як «Рецензія». Її додавання або редагування ще не було представлено на жодній сторінці проєкту, лише такий її елемент як оцінка можна було встановити раніше. Додамо ці функції на нову сторінку, яка буде повністю призначена цьому.

Сторінка повинна містити найголовнішу інформацію про книгу, щоб користувач не заплутався та не забув, до якої роботи він взагалі пише відгук.

Окрім цього, на сторінці повинен бути селектор вибору оцінки та поле для введення великого об’єму тексту для розгорнутої думки.

Розроблений вайрфрейм для рецензії зображено на рис. 2.6.

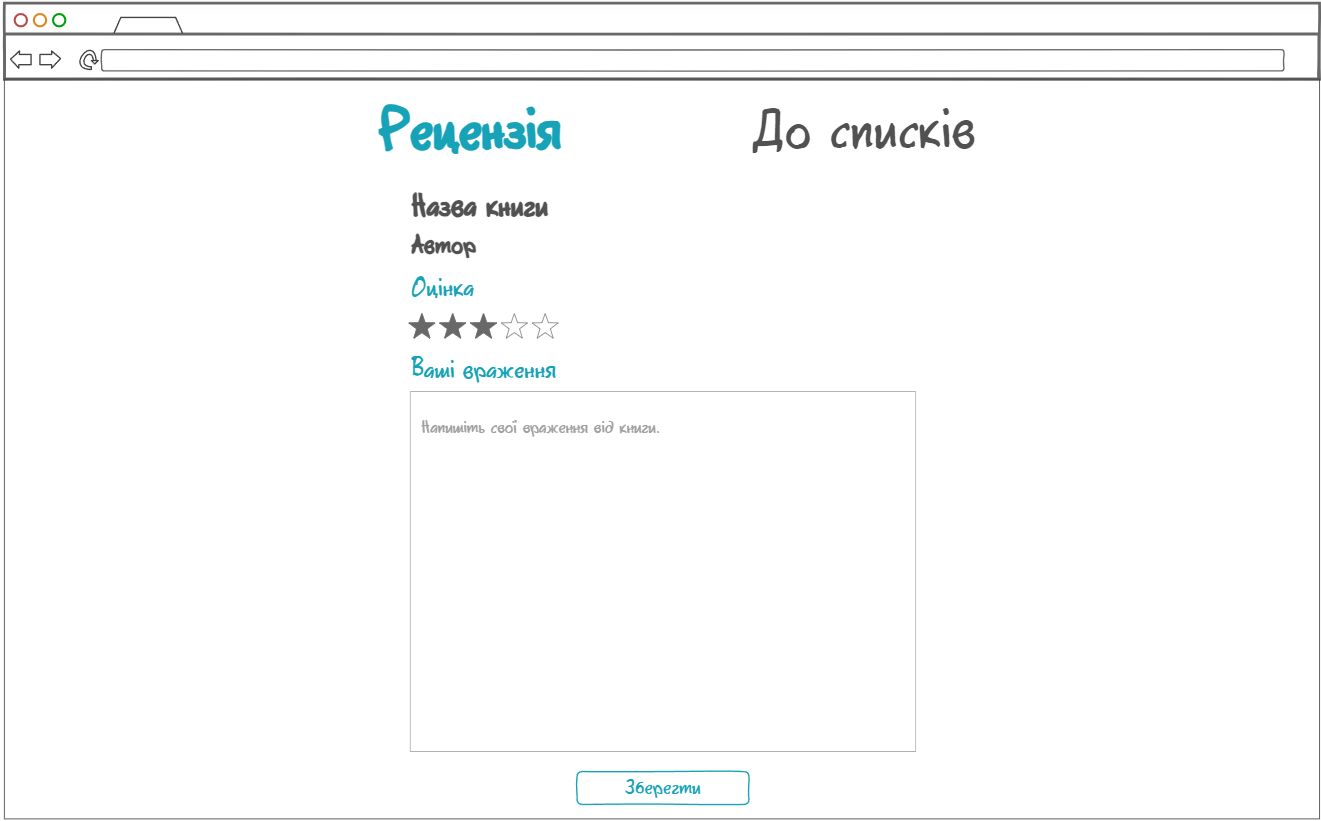


Рисунок 2.6 – Вайрфрейм редагування рецензії

Розроблені вайрфрейми допомогли скласти приблизний зовнішній вигляд вебзастосунка та знайти місце для усіх його функціональних можливостей, які були зазначені у вимогах.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Книга. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Книга (дата звернення: 30.03.2024).
2. Taycher L. Books of the world, stand up and be counted! All 129,864,880 of you. *Inside Google Books*. URL: https://booksearch.blogspot.com/2010/08/books-of-world-stand-up-and-be-counted.html (дата звернення: 30.03.2024).
3. Goodreads. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Goodreads (дата звернення: 01.04.2024).
4. Goodreads. *Wikipedia, the free encyclopedia*. URL: https://en.wikipedia.org/wiki/Goodreads (дата звернення: 01.04.2024).
5. Canton Connections. Comparing Reading Trackers. *Canton Public Library*. URL: https://www.cantonpl.org/blogs/post/comparing-reading-trackers/ (дата звернення: 01.04.2024).
6. Pinchuk M. New Goodreads Homepage Now Rolling Out to Everyone! Goodreads. URL: https://www.goodreads.com/blog/show/670-new-goodreads-homepage-now-rolling-out-to-everyone (дата звернення: 01.04.2024).
7. The StoryGraph. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/The\_StoryGraph (дата звернення: 01.04.2024).
8. The StoryGraph. URL: https://www.thestorygraph.com/ (дата звернення: 01.04.2024).
9. LibraryThing. Wikipedia, the free encyclopedia. URL: https://en.wikipedia.org/wiki/LibraryThing (дата звернення: 01.04.2024).
10. Nelson A. Goodreads v. LibraryThing- Part One. BOOK RIOT. URL: https://bookriot.com/goodreads-v-librarything-part-one/ (дата звернення: 01.04.2024).
11. Nelson A. Goodreads v. LibraryThing: Part Two. BOOK RIOT. URL: https://bookriot.com/goodreads-v-librarything-part-two/ (дата звернення: 01.04.2024).
12. Show Overview. LibraryThing.com. URL: https://www.librarything.com/quickstart.php (дата звернення: 02.04.2024).
13. Що таке JavaScript і для чого він потрібен. GoIT Global. URL: https://goit.global/ua/articles/shcho-take-javascript-i-dlia-choho-vin-potriben/ (дата звернення: 05.04.2024).
14. Vailshery L. S. Most used languages among software developers globally 2023. Statista. URL: https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/ (дата звернення: 05.04.2024).
15. Raval N. TypeScript vs JavaScript: Which One Is Better to Choose? Radixweb. URL: https://radixweb.com/blog/typescript-vs-javascript (дата звернення: 05.04.2024).
16. Що таке Реакт? web-developer.in.ua. URL: https://web-developer.in.ua/assets/articles/react/react-introduction/react-introduction.html (дата звернення: 16.04.2024).
17. State of JavaScript 2022: Front-end Frameworks. stateofjs. URL: https://2022.stateofjs.com/en-US/libraries/front-end-frameworks/ (дата звернення: 05.04.2024).
18. Angular для інтерактивних вебсайтів. Web-developer.in.ua. URL: https://web-developer.in.ua/assets/articles/angular/%D0%B0ngular-interactive/%D0%B0ngular-interactive.html (дата звернення: 16.04.2024).
19. Angular vs React: як обрати відповідний фреймворк під проєкт. FoxmindEd. URL: https://foxminded.ua/angular-vs-react/ (дата звернення: 18.04.2024).
20. Все що потрібно знати про Node.js: особливості, характеристики, область застосування. Wezom. URL: https://wezom.com.ua/ua/blog/vse-chto-nuzhno-znat-o-nodejs (дата звернення: 18.04.2024)
21. Романенко Ю. Не соромно запитати: що таке вайрфрейм, мокап і прототип. SKVOT. URL: https://skvot.io/uk/blog/ne-soromno-zapitati-shcho-take-vayrfreym-mokap-i-prototip (дата звернення: 08.04.2024).