ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Кафедра інженерії програмного забезпечення

**КУРСОВИЙ ПРОЄКТ**

з дисципліни

«Вебтехнології»

на тему: Розробка сервісу для читання і покупки книг

**КПІПЗ.250129.01.04.ПЗ**

Галузь знань . 12 «Інформаційні технології» .

Спеціальність . 121 «Інженерія програмного забезпечення»

(шифр, назва)

Студента І . курсу, група . ІПЗс-25-1. \_\_\_\_\_\_\_\_\_\_ Дмитро ДОСКОЧ

(шифр) (підпис) (ініціали, прізвище)

Керівник . доцент, канд.техн.наук.. \_\_\_\_\_\_\_\_\_\_\_ Оксана ЯШИНА

(посада, вчене звання, науковий ступінь) (підпис) (ініціали, прізвище)

Кількість балів \_\_\_\_\_\_\_

Оцінка за шкалою:

національною\_\_\_\_\_\_\_\_\_/ЄКТС\_\_\_\_

Члени комісії: \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали, прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали, прізвище)

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (ініціали, прізвище)

Хмельницький 2025

*Змін.*

*Аркуш*

*№ докум.*

*Підпис*

*Дата*

*Розробив*

*Перевірив*

*Затвер.*

*Н.контр.*

*Літ*

*Аркуш*

*Аркушів*

**ЗМІСТ**

[**ВСТУП** 5](#_Toc212098644)

[**РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ СЕРВІСУ ДЛЯ ЧИТАННЯ ТА ПОКУПКИ КНИГ** 7](#_Toc212098645)

[1.1 Призначення розроблюваного продукту та проблеми предметної області 7](#_Toc212098646)

[1.2 Огляд існуючих рішень, методів та засобів розробки 7](#_Toc212098647)

[1.3 Ролі, функціональні та нефункціональні вимоги 14](#_Toc212098648)

[1.4 Постановка задачі 17](#_Toc212098649)

[**РОЗДІЛ 2 ПРОЄКТУВАННЯ СТРУКТУРИ ТА КОМПОНЕНТІВ ПРОЄКТУ** 19](#_Toc212098650)

[2.1 Проєктування архітектури 19](#_Toc212098651)

[2.2 Проєктування бази даних 21](#_Toc212098652)

[2.3 Проєктування інтерфейсу 25](#_Toc212098653)

[2.4 Вибір стеку технологій 28](#_Toc212098654)

ABSTRACT

Course project “Service for reading and purchasing books.”

Author: Doskoch D.S.

Supervisor: Yashina O.M.

Volume – pages, figures, tables, appendices, sources.

Keywords:

The purpose of the course project is to develop a service for reading and purchasing books that will allow users to purchase and immediately read electronic versions of books on the web resource, automatically translate books into the language they need using the translator built into the site, and enable novice authors to find their audience.

The practical significance of the results obtained lies in the creation of a convenient service for reading and purchasing e-books directly on the web resource, without the need to perform unnecessary actions. In addition, some books will be free to read, which will simplify users' access to literature. The integrated translator will remove the language barrier and allow users to read various publications that are not localized for the reader. Support for new authors will enable them to find an audience.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(дата) (підпис)

**ВСТУП**

Тема курсового проєкту полягає у створенні вебсервісу, який дозволить користувачам читати та купувати книги на одному ресурсі.

Об’єктом дослідження є електронні сервіси для читання та покупки книг.

Предметом дослідження є функціональні можливості сервісів та необхідності користувачів у процесі читання електронних книг

Сьогодні кожен аспект суспільства так чи інакше підлягає цифровізації, від державних послуг до роботи. Звісно така сфера як література, не могла залишитись осторонь від цього, зараз дуже активно набуває популярності читання саме електронних книг, адже це і зручно, і дешевше ніж читання фізичних видань, та створює більшу гнучкість, від самостійного вибору мови, до вибору того чи іншого перекладу.

Через більшу зручність та меншу вартість саме сервіси для покупки та читання електронних книг набувають все більшої популярності, як серед читачів, які зацікавлені у цьому через зручність, так і серед авторів, які можуть на таких сервісах знайти свою початкову аудиторію, та отримати реальні відгуки про свої твори, без необхідності попередньо шукати видавця, який погодиться видати його книгу.

У зв’язку з великим попитом ми маємо надати користувачам необхідний для них ключовий функціонал, який буде реалізовано у нашому сервісі. Основним функціоналом сервісу будуть такі функції: можливість придбати книгу, читати книгу відразу на вебресурсі, перекласти на необхідну для користувача мову за допомогою вбудованого перекладача, якщо такого перекладу не існує, також функціонал для авторів, який дозволить виставляти свої твори на сайт, та отримувати статистику по читачам.

Більшість цих функцій в тій чи іншій мірі також присутні і на інших сайтах схожої направленості, але загалом вони там реалізовані лише частково, і сукупно цих функцій немає в жодному українському аналогу, і в майже усіх іноземних.

Отже, розробка сервісу такої направленості є актуальною та доцільною під сучасні реалії, не тільки в Україні, а й при локалізації та грамотному піарі з можливістю потрапити ще й на іноземний ринок. Також варто зазначити відносно малу кількість конкурентів, адже зачасту такі послуги надають лише частково книжкові магазини, які все ж в більшості орієнтовані на продаж саме фізичних видань, і електронні видання для них є лише субпродуктом.

**РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ СЕРВІСУ ДЛЯ ЧИТАННЯ ТА ПОКУПКИ КНИГ**

1.1 Призначення розроблюваного продукту та проблеми предметної області

Сучасна сфера літератури активно переходить в фазу цифровізації. Все більше читачів віддають перевагу саме електронним книгам над фізичними виданнями, адже це зручніше, дешевше, а також гнучкіше і дозволяє читачам в будь-який час мати доступ до книг. Це постійно збільшує попит на сервіси для онлайн читання книг на платній чи безоплатній основі.

Основним призначенням сайту є створення єдиної системи, яка б вирішувала три доволі суттєві проблеми на ринку, які в повній мірі не можуть бути вирішені існуючими рішеннями, лише частково і не на всіх ресурсах:

* комфортне інтегроване читання, адже більшість комерційних ресурсів лише надають можливість після покупки завантажити файл в тому чи іншому форматі.
* підтримка молодих авторів, нажаль зачасту маловідомі або тільки починаючі письменники не з’являються на сайтах по продажу електронних книг, і тому мають постійно відвідувати видавців, щоб спробувати знайти свою аудиторію.
* усунення мовних бар’єрів, адже зачасту якщо переклад і застосовується, то лише для перекладу друкованих видань.

Відсутність цих факторів негативно впливає на усіх учасників літературного процесу, як авторів, які не можуть без ризику знайти свою аудиторію, так і на читачів, які явно обмеженні у виборі, та не зручному читанні.

1.2 Огляд існуючих рішень, методів та засобів розробки

Почнемо аналіз з такого сайту як KSD [1], який є інтернет магазином, для електронних та фізичних видань (Рисунок 1.1):

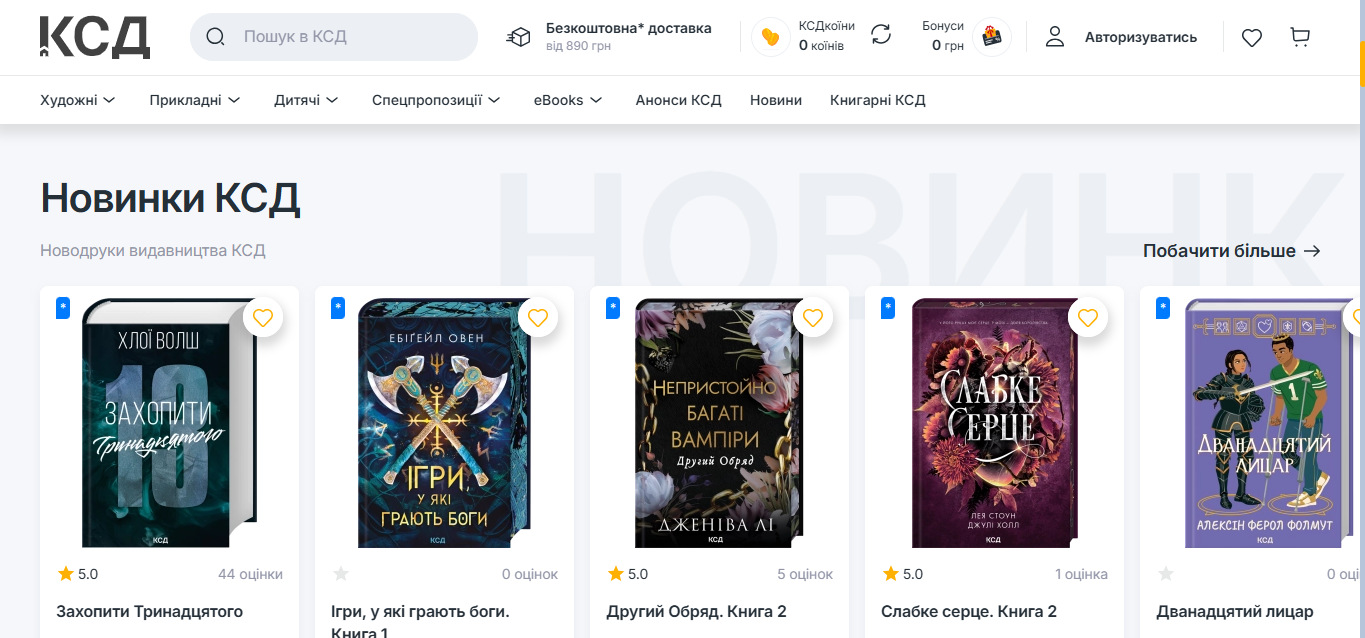


Рис. 1.1 – Вітальня сторінка сайту KSD

Отже, як ми можемо побачити там реалізовано все подібно до того сервіса який ми будемо розробляти, а саме те що неавторизований користувач так само може переглядати книги, інформацію про них, та відгуки, і може авторизуватись для більшої інформації, варто зазначити що книгарня продає і електронні видання (рисунок 1.2):

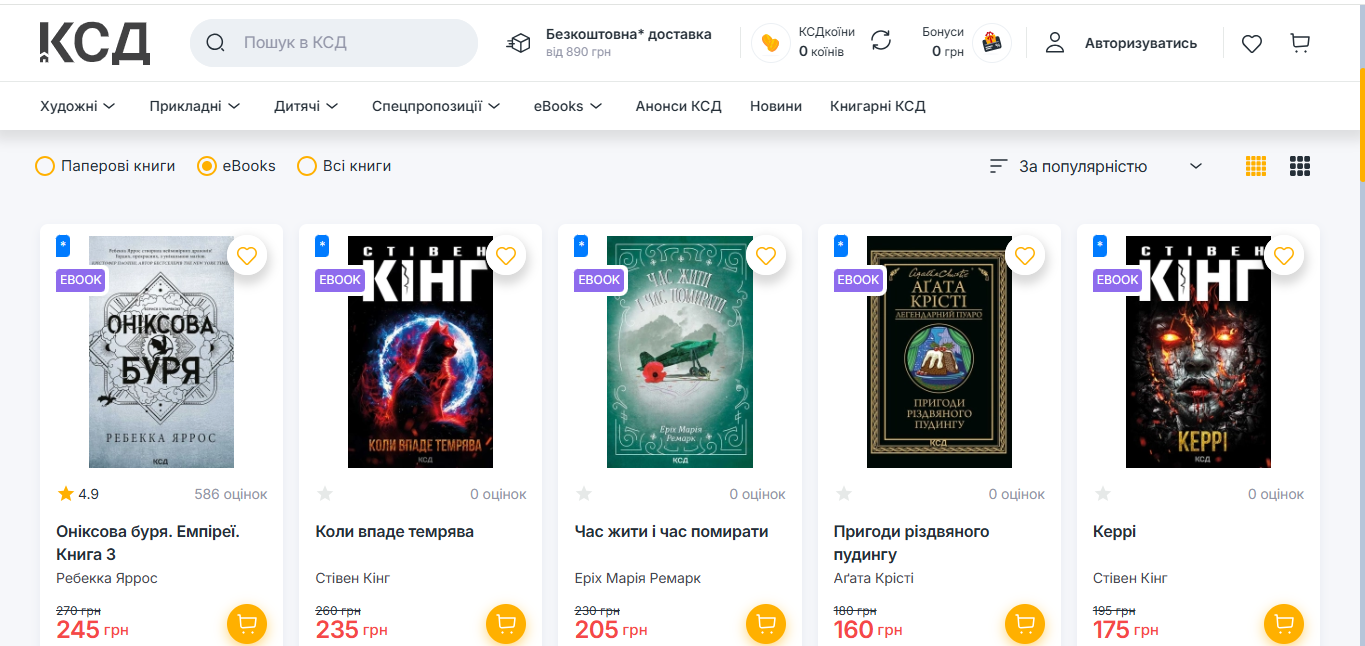


Рис. 1.2 – Електронні примірники на сайті KSD

Але тут електронні книги є лише субпродуктом, і відповідно у такій варіації є лише деяка частина книжок, з усіх доступних на сайті. Отже, ми маємо подібний до нашого сайт, але з головною відмінністю у вигляді орієнтації на фізичні видання.

Тому перейдемо до наступного сайту, який займається комерцією книг, саме у електронному форматі Epub [2](Рисунок 1.3):

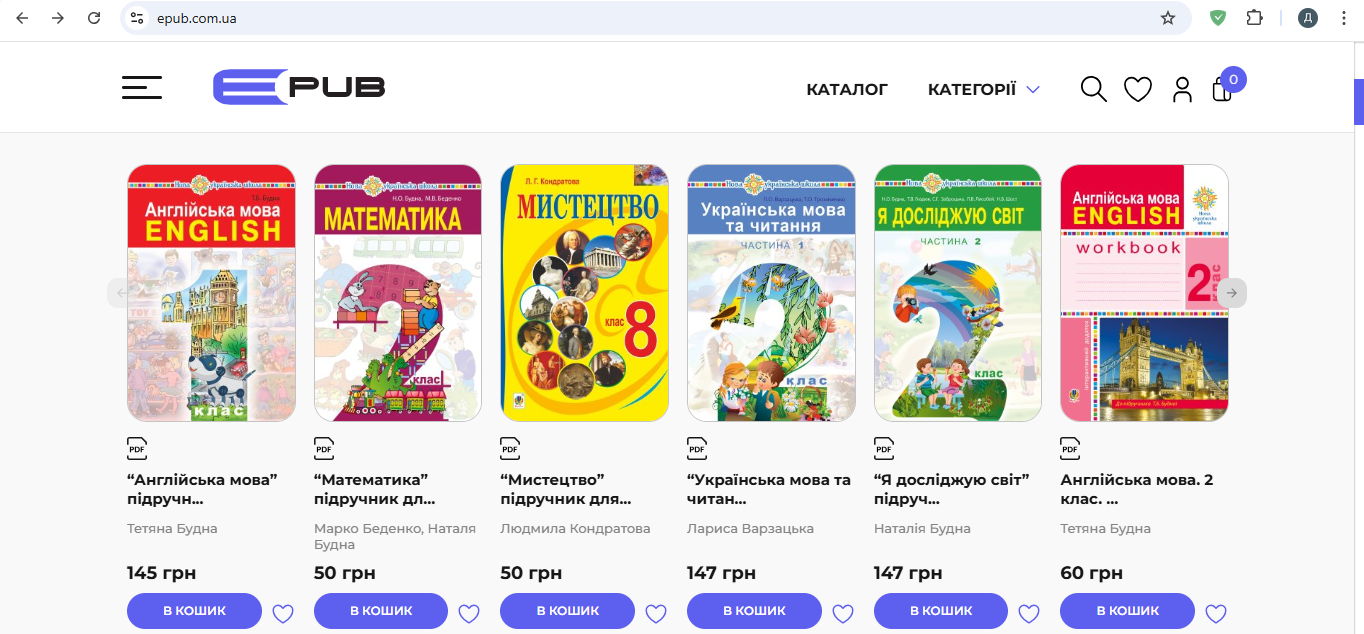


Рис. 1.3 – Головна сторінка сайту з продажу електронних книг Epub

Проаналізуємо можливості які представляє даний сайт і які буде представляти наш сервіс. Пройдемо по відмінностям, перш за все на даному ресурсі немає можливості прочитати книгу безпосередньо там, при покупці користувач лише отримає можливість завантажити файл з вмістом відповідної книги. Також на даному сайті відсутня можливість перекладу, і якщо користувач хоче, то йому необхідно самому шукати сторонній ресурс та перекладати, і третьою відмінністю є неможливість завантажити свою книгу. Варто відзначити доволі продуману і інтуїтивно зрозумілу систему фільтрації, що дозволить швидко знайти книгу з саме шуканого сайту(Рисунок 1.4):

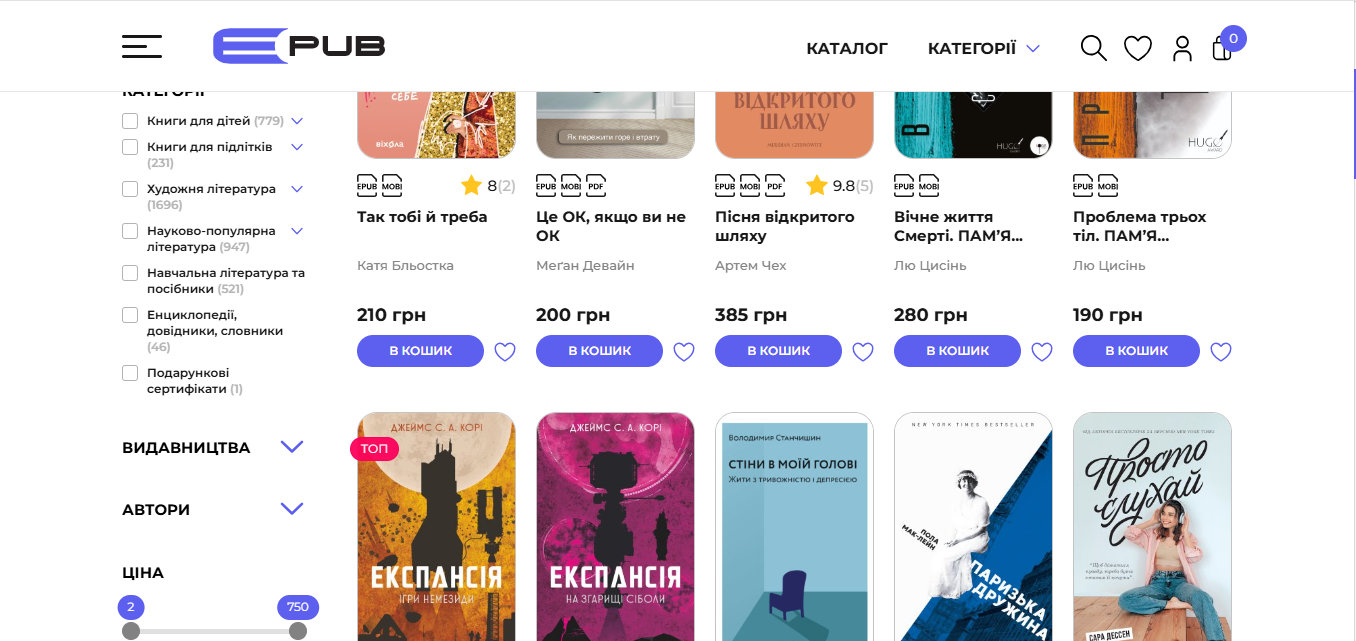


Рис. 1.4 – Фільтрація доступна на ресурсі Epub

Отже, як ми можемо побачити що у даного ресурсу доволі багато схожого з розроблюваним нами, але є доволі багато відмінних моментів, які дозволять уніфікувати наш сайт від прямого конкурента з комерції електронних книг. Розглянемо ще один вебресурс який також займається продажем книг. Це буде вебресурс Книжкова Майстерня [3](Рисунок 1.5):

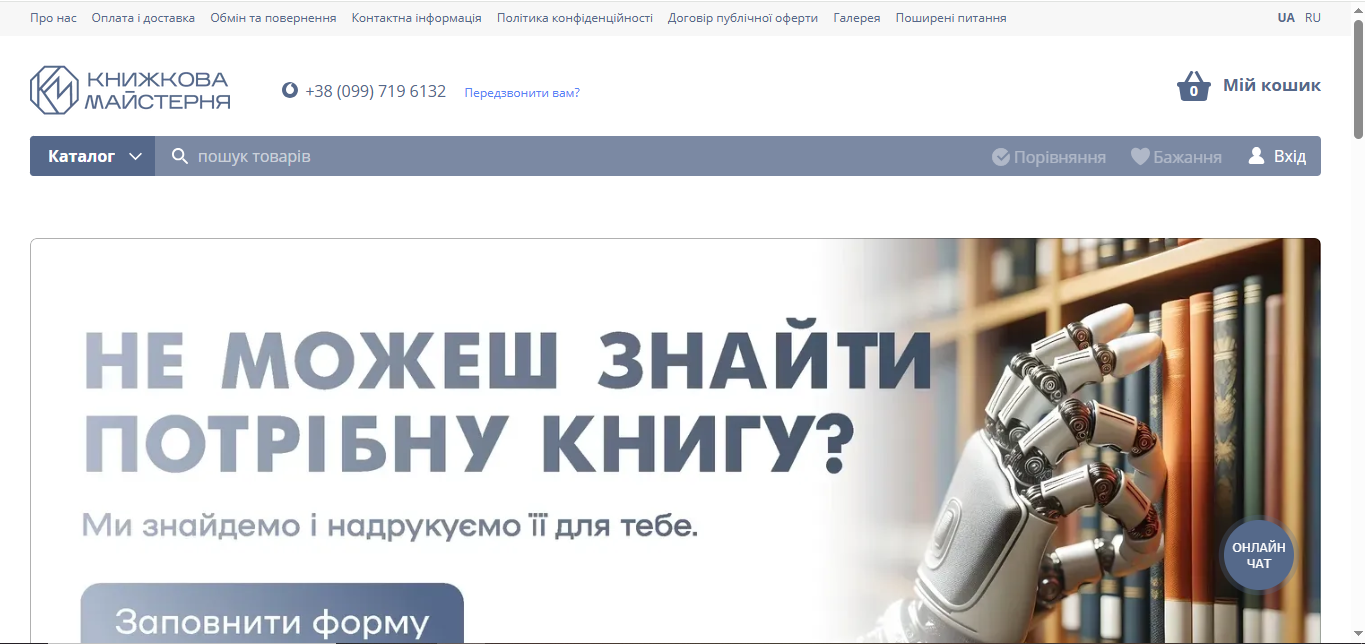


Рис. 1.5 – Головна сторінка сайту Книжкова Майстерня

Не зважаючи на те що цей вебсайт займається поширенням саме фізичних видань, вирішили взяти його на аналіз тому що у нашого вебресурсу з ним доволі багато схожих аспектів. Наприклад автоматичний переклад творів, який повністю недоступний користувачу через продаж саме фізичних видань, і може бути лише частково наданий для оцінки якості перекладу. Також є можливість поспілкуватись з підтримкою сайту, писати і переглядати відгуки до книг, а також сформувати список бажаних для покупки книг. Дуже цікавою особливістю цього сайту також є можливість запропонувати напряму книгу іншого автора для друку, надавши або посилання на оригінал, або завантаживши саму книгу у прийнятному форматі(Рисунок 1.6):

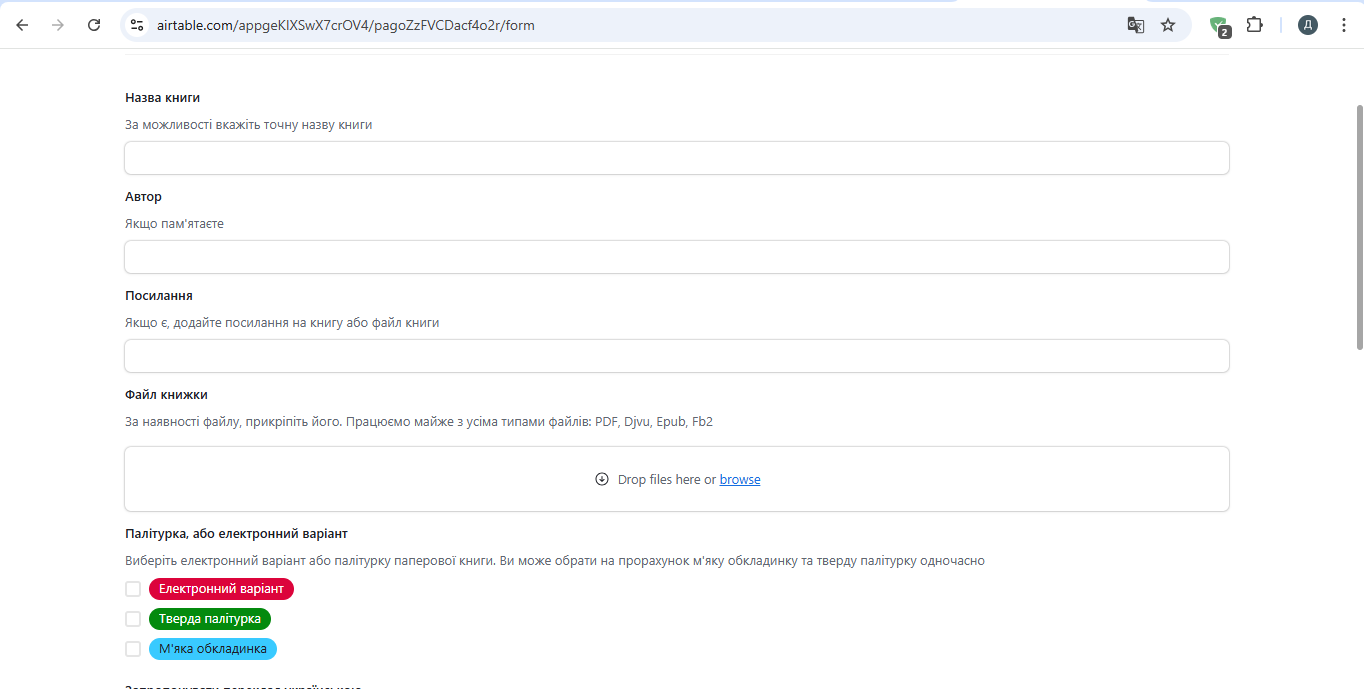


Рис. 1.6 – Форма для пропозиції певної книги на сайті Книжкова Майстерня

Переглянемо наступний вебсайт, а саме КнигарняЄ[4](Рисунок 1.7):

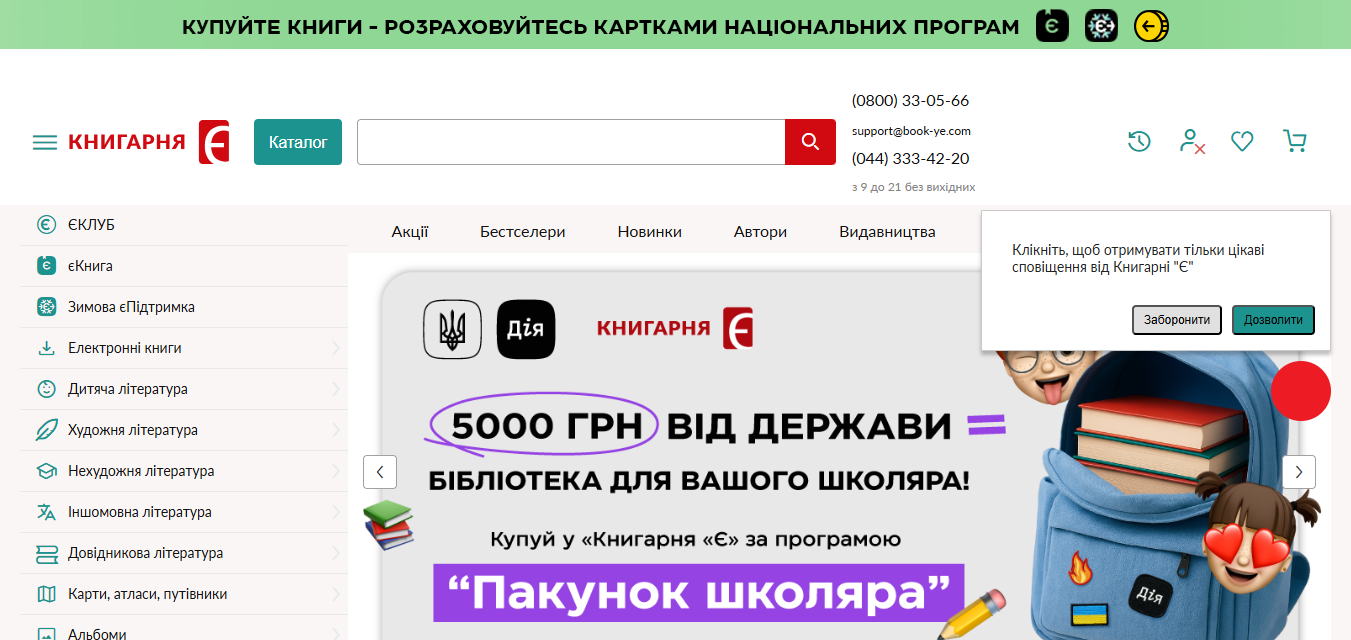


Рис. 1.7 – Вітальна сторінка сайту КнигарняЄ

На цьому ресурсі нас зустрічає доволі не привабливий дизайн, який є дратівлим для користувача, постійними вспливаючими вікнами з пропозиціями, що зараз вважається застарілим, і від чого більшість сайтів наразі активно відмовляються. На сайті присутній продаж електронних книг, який реалізовано шляхом покупки файлу книги, яку потрібно відкрити додатком для певного формату і лише тоді читати. Дуже вдалою є функція прочитати уривок , яка надає користувачу скановану частину книги, щоб дізнатись чи зацікавить вона його (Рисунок 1.8):

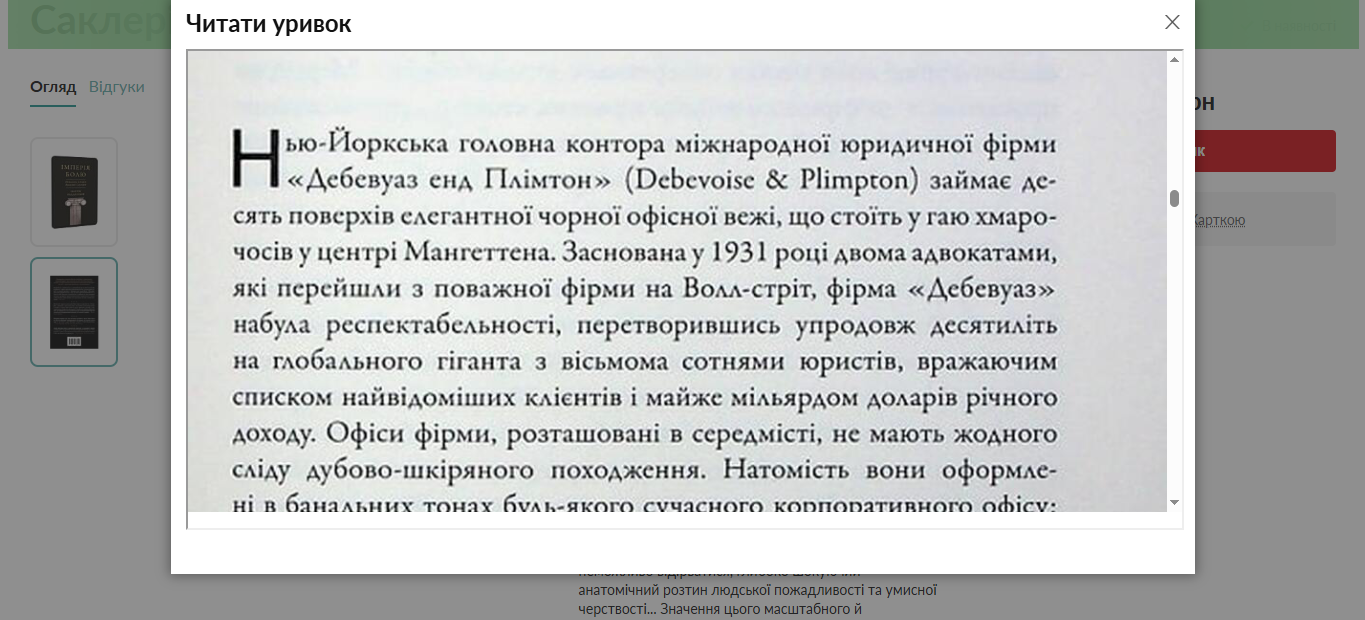


Рис. 1.8 – Уривок книги на сайті КнигарняЄ

Інших особливого функціоналу на сайті немає, і він є цілком стандартним для будь-якого інтернет магазину.

Перейдемо до останнього магазину літератури, вибраного для порівняння, а саме до сайту Академкнига[5](Рисунок 1.9):

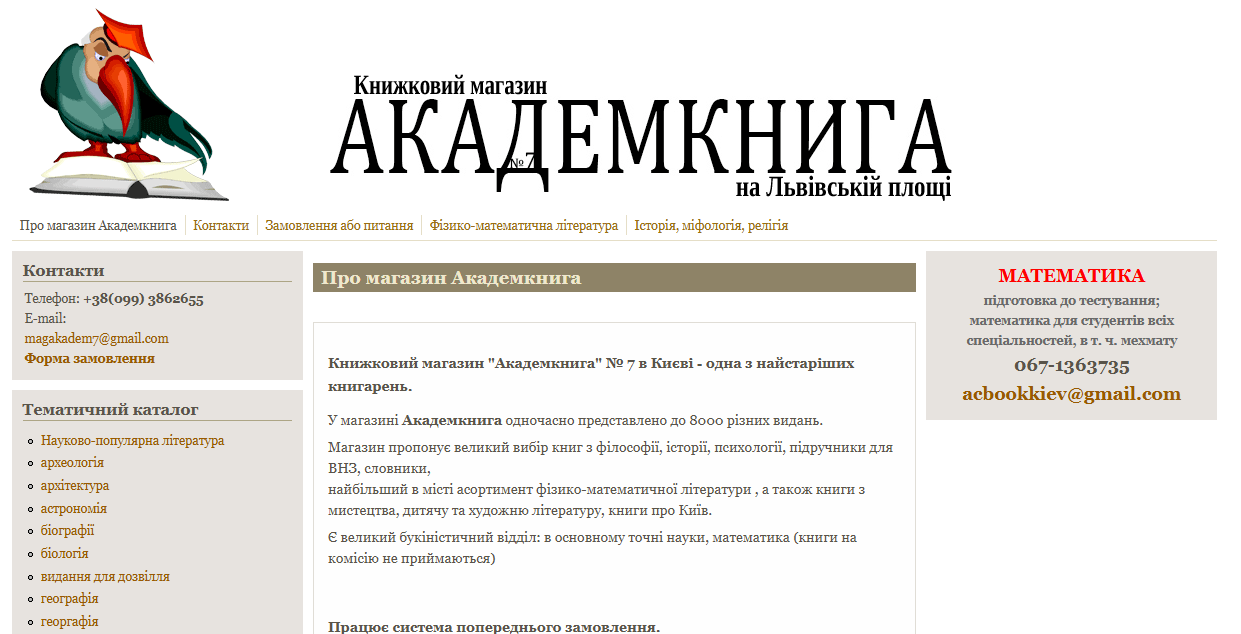


Рис. 1.9 – Головна сторінка сайту Академкнига

Після дослідження цього магазину в очі кидається застарілий інтерфейс, зроблений двома різними мовами, відсутність явного продажу, лише форма для того щоб лишити замовлення(Рисунок 1.10) та виключно мала кількість інформації про самі книги, які пропонує магазин(Рисунок 1.11):

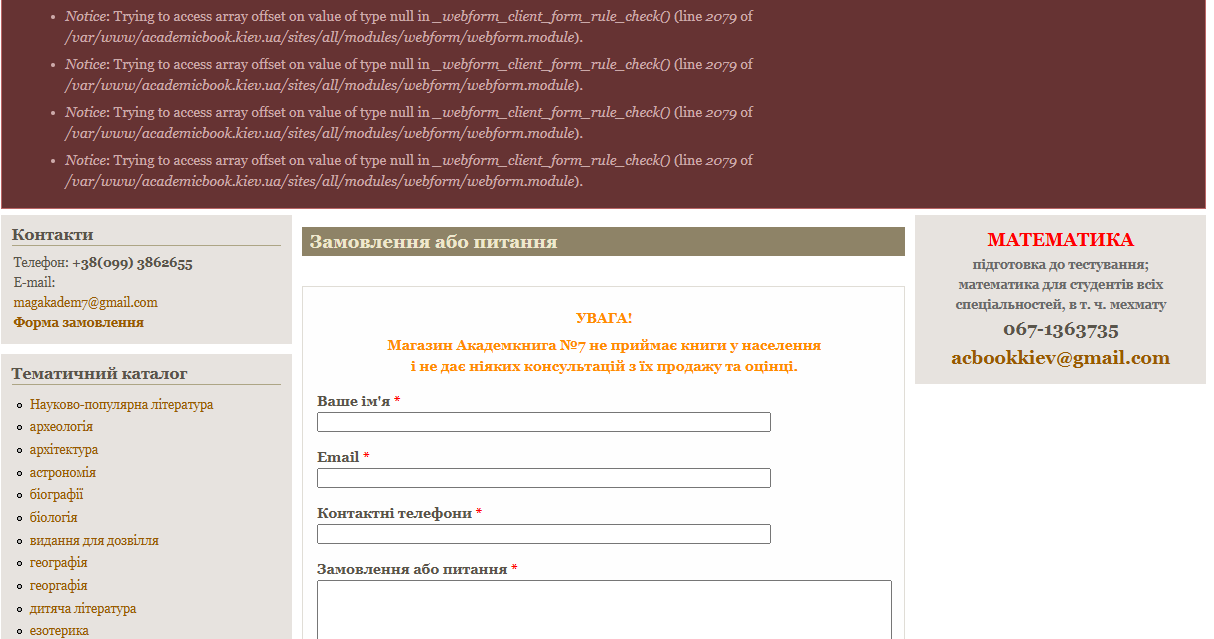


Рис. 1.10 – Форма для замовлення у магазины Академкнига



Рис. 1.11 – Обрана книга на сайті магазину Академкнига

Крім показаного на рисунках, нажаль більше ніяких можливостей сайт не представляє, лише можливість також дізнатись контакти у вигляді робочого телефону та електронної пошти.

Отже проаналізувавши загалом 5 різних сайтів однієї сфери з нашим ресурсом, ми побачили що хоча на деяких з них є частково риси схожі на наш ресурс, але немає сайту який зміг би об’єднати весь функціонал нашого розроблюваного сервісу разом.

Тому наш ресурс вийде на ринок цілком конкурентоспроможним, забезпечивши користувачів усіма необхідними функціями, без необхідності переходити на сторонні сайти для виконання певних з них, через обмежений функціонал ресурсу.

Визначивши призначення сайту, усі необхідні для його функціонування ролі та їхні можливості, та проаналізувавши подібні до нашого ресурсу аналоги, можемо перейти до складання технічного завдання для нашого розроблюваного ресурсу.

1.3 Ролі, функціональні та нефункціональні вимоги

Тепер нам необхідно визначити усі вимоги до нашого додатку щоб, знати що і як реалізовувати.

Почнемо з функціональних вимог, розділимо їх по ролям користувачів, які їх будуть виконувати, зробимо це за допомогою таблиці, що дозволить нам краще структурувати нашу інформацію та краще її сприймати на наступних кроках розробки:

|  |  |  |
| --- | --- | --- |
| **Користувач** | **Функція** | **Опис** |
| Неавторизований користувач | Авторизація | Авторизація на сайті за уведеними даними |
| Загальне для всіх користувачів | Читання літератури | Можливість читати літературу на сайті |
| Адміністратор | Керування користувачами | Адміністратор повинен мати змогу різним чином керувати користувачами, від блокування і видалення облікового запису за порушення, до внесення певної зміни, на випадок якщо користувач хоче це зробити, але має певні труднощі з цим |
| Адміністратор | Перевірка книг на цензуру та помилки | Адміністратору необхідно могти переглядати книгу перед тим як виставити її у загальний доступ на сайті |
| Адміністратор | Перевіряти метрики сайту | Перевіряти статистику по читачах, по доходах з книги, по зацікавленості автором |
| Адміністратор | Видалення\приховання книги | Можливість видалити або приховати книгу у зв’язку з різними обставинами |
| Адміністратор і користувач(автор) | Редагування змісту книг | Редагування описів, обкладинки та додавання розділів(для деяких творів) |
| Користувач | Реєстрація | Можливість зареєструвати обліковий запис |
| Користувач | Відновлення пароля | Можливість відновити пароль до облікового запису |
| Користувач | Придбання книг | Купувати різні книги |
| Користувач | Списки бажаного | Створювати списки бажаних для покупки книг |
| Користувач | Оцінка та коментарі | Можливість оцінювати та ділитись коментарями щодо книг |
| Користувач | Переклад | Автоматичний переклад за допомогою інтегрованого АРІ |
| Користувач | Підтримка | Можливість написати у підтримку щодо якогось питання і отримати відповідь |
| Користувач | Знижки | Використання знижок |
| Користувач(Автор) | Виставлення власних творів | Може виставляти власний твір на сайті для пошуку своєї аудиторії або отримання доходу\рекомендацій |
| Користувач(Автор) | Статистика | Збирання статистики щодо усієї активності щодо додатку |
| Користувач(Автор) | Отримання коштів | Можливість отримання коштів за продаж власних електронних книг |

Таблиця 1.1 – Функціональні вимоги нашого вебсайту

Перейдемо до нефункціональних вимог нашого додатку, для кращого сприйняття розділимо їх по 5 стандартних типах:

1. Вимоги до застосування;
2. вимоги до надійності;
3. вимоги до продуктивності;
4. вимоги до обслуговування;
5. вимоги до конструктивних обмежень.

Також візуалізуємо за допомогою таблиць наші вимоги до кожного з цих типів, які будуть реалізовуватись на нашому сайті під час розробки.

Почнемо з вимог до застосування, usability англійською :

|  |  |
| --- | --- |
| Вимога | Опис та значення |
| Зрозумілість інтерфейсу | Буде орієнтація на максимально швидке розуміння користувачем як користуватись сайтом без допомоги |
| Час на адаптацію | Час на розуміння користувачем усіх основних інструментів сайту не має перевищувати 5 хвилин, щоб користувач не вирішив закрити сайт |
| Видні елементи керування | Робитимуться усі можливі елементи інтерфейсу максимально підкресленими дизайном, щоб користувач їх не пропускав |

Таблиця 1.2 – Вимоги до застосування

Перейдемо до наступного типу вимог, а саме до вимог до надійності, так само їх візуалізуємо за допомогою таблиці для кращого сприйняття:

|  |  |
| --- | --- |
| Вимога | Опис та значення |
| Робота без збоїв | Буде реалізовуватись максимально можлива стабільність сайту, з мінімум можливих збоїв, розділення фронтенду та бекенду допоможе з цим |
| Дані користувачів збережуться навіть при збої | Через збереження даних на окремій базі даних MySQL, навіть при можливих збоях дані збережуться |
| Повідомлення при помилці | Яка б помилка не була, користувачу має показати що це за помилка і можливу причину |

Таблиця 1.3 – Вимоги до надійності

Далі перейдемо до наступних вимог, до вимог до продуктивності, де ми зазначимо вимоги до швидкодії нашого сайту під час використання користувачем:

|  |  |
| --- | --- |
| Вимога | Опис та значення |
| Час завантаження | Необхідно буде зробити щоб на звичайних сторінках сайту час завантаження був до трьох секунд, а на сторінці для читання до п’яти секунд |
| Пошук книги | Пошук книги на стороні MySQL сервера не може тривати довше однієї секунди |
| Підтримка багатьох користувачів | Сайт повинен витримувати навантаження багатьох користувачів, вибраний стек технологій цілком дозволить витримувати близько 20 користувачів без ніяких налаштувань |

Таблиця 1.4 – Вимоги до продуктивності

Перейдемо до четвертого типу нефункціональних вимог, до вимог до обслуговування:

|  |  |
| --- | --- |
| Вимога | Опис та значення |
| Розділення коду | Увесь код має бути розділений на модулі, обрана архітектура уже передбачає базове розділення і роботу вже з урахуванням цього |
| Дружелюбність до додавання нових функцій | Проєкт має бути дружелюбним до додавання нових функцій і не повинен вимагати зміну більше ніж двох модулів при цьому |
| Документування змін | Усі зміни в проєкті мають бути задокументовані, чому сприятимуть логи у проєкті а також, історія репозиторію на гітхаб, що допоможе швидше зрозуміти усе починаючому працювати над проєктом розробнику |

Таблиця 1.5 – Вимоги до обслуговування

Перейдемо до наступної, останньої категорії нефункціональних вимог, а саме до вимог до конструктивних обмежень, так само візуалізуємо за допомогою таблиці:

|  |  |
| --- | --- |
| Вимога | Опис та значення |
| Вимоги до система | Система буде розділена на дві складові, бекенд та фронтенд, бекенд буде розроблено за допомогою ASP.NET Core, а фронтенд за допомогою React |
| База даних | Базою даних проєкту буде база даних MySQL |
| Сумісність | Сайт повинен працювати корректно на усіх можливих десктопних браузерах, а також коректно працювати з телефону |

Продовження таблиці 6

1.4 Постановка задачі

Отже, метою курсового проєкту є розробка веб-застосунку, що дозволить користувачам купувати та читати книги, який спробує вирішити виявленні проблеми в ході аналітики в розділі 1.1.

На основі детального аналізу уже існуючих рішень у розділі 1.2, було встановлено, що всі проаналізовані комерційні платформи реалізують лише частковий функціонал і в більшості є орієнтовані на продаж саме фізичних видань, не надаючи нормальних інструментів для зручного онлайн читання.

Виходячи з усіх знайдених недоліків, головна мета проєкту полягає у створенні конкуренто спроможного на ринку, продукту, який буде відповідати функціональним та не функціональним вимогам з розділу 1.3.

Для досягнення мети необхідно зробити наступне:

1. Розробити логічну структуру сайту дружньою до користувача, що дасть змогу легкого керування користувачам сайту;
2. створити архітектуру бази даних MySQL, яка підтримуватиме зв'язки між сутностями нашого додатку, та дозволить створити тригери, процедури, та івенти, які дозволять більш гнучко керувати базою даних нашого сайту;
3. спроєктувати інтуїтивно зрозумілий та зручний для користувачів дизайн сайту, який дозволить легко читати у браузері;
4. обрати та обгрунтувати самий оптимальний та сучасний стек технологій для роботи нашого додатку, чому цілком відповідають обрані ASP.NET Core як бекенд, та React як фронтенд;
5. розробити детальний опис усіх модулів проєкту, які будуть у додатку, наприклад модуль для читання, для зберігання інформації, та інші.

Розроблюваний програмний продукт є вебсайтом, з клієнт-серверною архітектурою, що буде реалізований за допомогою обраного стеку технологій. Технічне завдання розроблюваного проєкту, що вказує більш детально усі вимоги буде доданий додатком 1.

**РОЗДІЛ 2 ПРОЄКТУВАННЯ СТРУКТУРИ ТА КОМПОНЕНТІВ ПРОЄКТУ**

2.1 Проєктування архітектури

Існує багато різних підходів до розробки проєктів. Кожен краще підходить під певну задачу, під певний стек технологій, під певні бажання замовника чи розробника.

Необхідно продумати найкращу можливу архітектуру для нашого розроблюваного вебсайту.

Після довгих роздумів була обрана клієнт-серверна архітектура, вона є наразі самою популярною на ринку, та ідеально підходить під веб застосунки, адже має наступні переваги:

1. Розділення бізнес логіки і дизайну;
2. краща адаптованість до змін;
3. можливість працювати почергово над усіма модулями вебсайту.

Потім потрібно було обрати тип клієнт-серверної архітектури, наш вибір зупинився на багаторівневій архітектурі, з реалізацією SPA(single page application) та API.

Саме такий вибір технологій дозволяє не лише паралельно працювати відразу над трьома елементами нашого сайту, а й дозволяє легко масштабуватись, адже навіть при повній замінні якого одного з аспектів додатку, інші 2 будуть цілком готовими і не потребуватимуть кардинальних змін, якщо взагалі будуть потребувати.

Проєкт розділений на 3 рівня:

1. рівень представлення;
2. рівень бізнес логіки;
3. рівень доступу до даних.

Розглянемо детальніше кожен з розділів, що він в себе включає, і за що відповідає у роботі нашого вебсайта.

Рівень представлення – відповідає за те що побачить користувач, саме на цьому рівні існує рішення як будуть відображатись елементи вебсайту на екрані користувача. Реалізується цей рівень за допомогою React, який є підходящим під нашу задачу по розробці магазину електронних книг. Цей рівень є самим гнучким, адже навіть критичні зміни в якомусь з модулів, не будуть зачіпати інші завдяки чіткому розділенню кожної функції на модулі і відносну не зв’язаність між ними. Взаємодіє з іншими рівнями за допомогою асинхронних запитів до API, що є нашим рівнем бізнес логіки.

Рівень бізнес логіки – без сумніву є серцем усієї системи вебсайту, адже саме на цьому рівні існують всі операції для роботи з даними, і логіка для них, для цього рівня було обрано ASP.NET Core + Entity framework за високий рівень стабільності, та за легкість у масштабування. Усі основні процеси проходять саме на цьому рівні, від самих простих, як наприклад авторизація, до самих складних, як наприклад надання доступу до текстового вмісту книг. Побудований рівень на принципах RESTful API, а саме на використанні стандартних HTTP методів.

Рівень доступу до даних – цей рівень характеризується базою даних та представляє собою саму головну частину вебсайту, адже зберігає в собі усі дані сайту та користувачів. Для взаємодії з даним рівнем та рівнем бізнес логіки використовується Entity Framework що значно підвищує швидкість роботи додатку з базою данних.

На рисунках 2.1 і 2.2 зображено схематичне зображення архітектури:

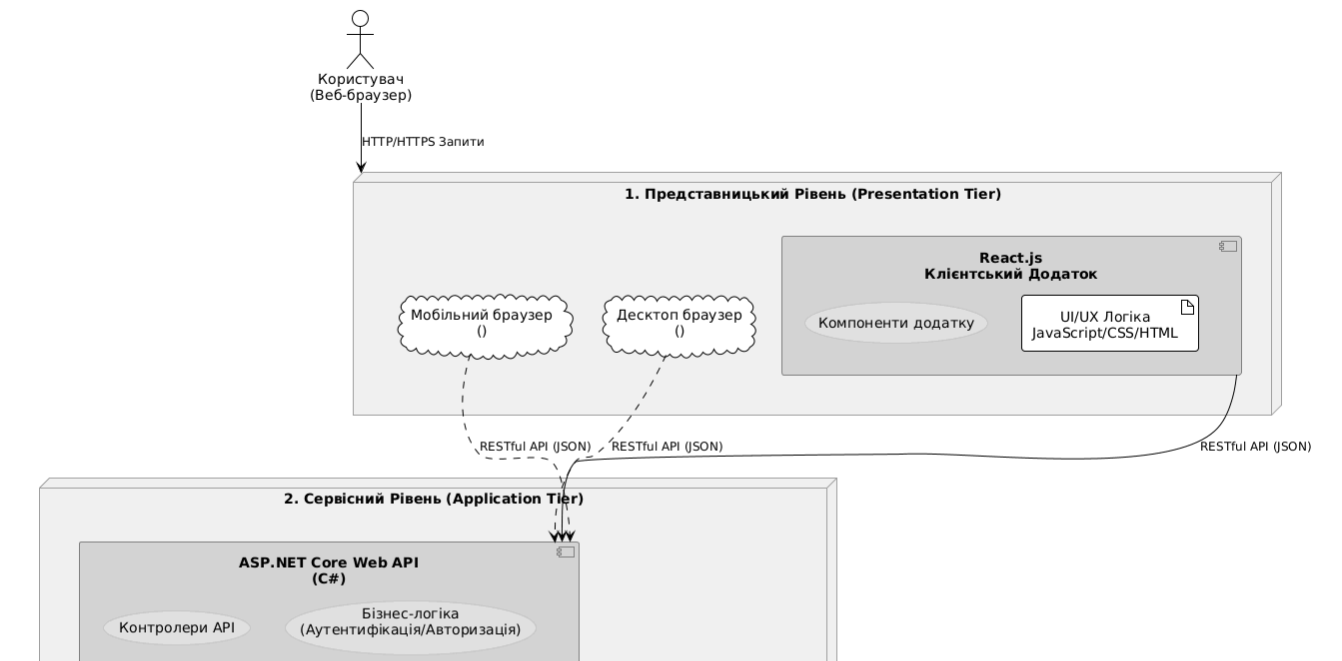


Рис 2.1 – Діаграма архітектури вебсайту 1

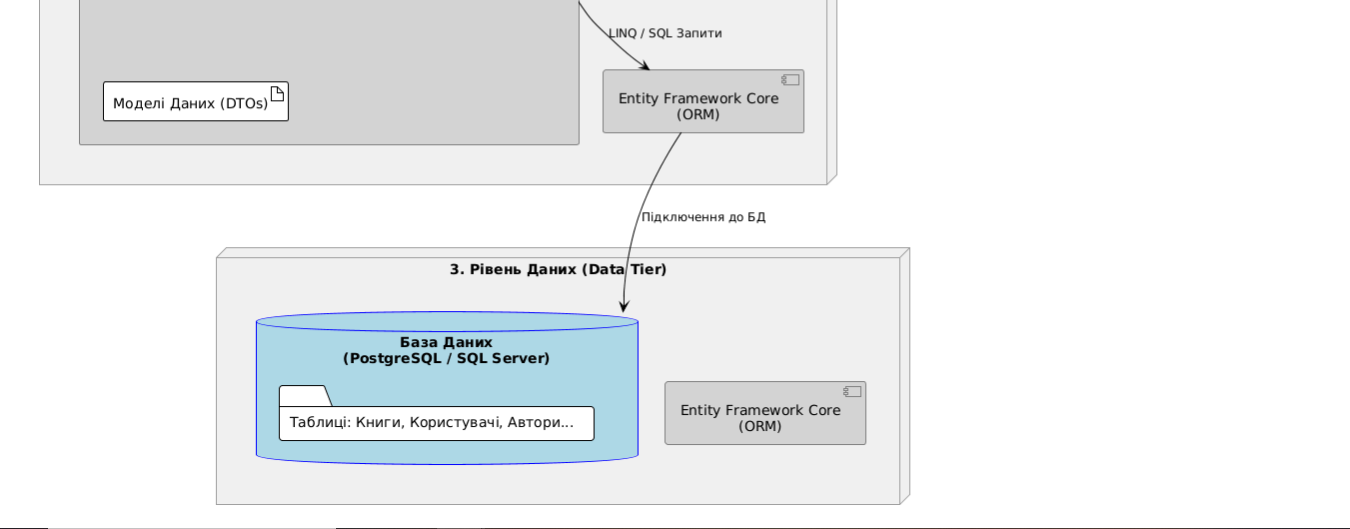


Рис 2.2 – Діграма архітектури вебсайту 2

Отже, завдяки чіткому розділенню рівнів архітектура горизонтально масштабована, а це означає що у разі збільшення користувачів, ми зможемо проводити дії над базою даних, або над бізнес логікою, не чіпаючи клієнтську частину, а це гарантує високу доступність сервісу.

2.2 Проєктування бази даних

База даних беззаперечно є основою будь-якого додатку, адже саме навколо неї будується увесь наш додаток.

Під час вибору бази даних для проєкту, було вибрано реляційну базу даних, адже це допоможе відразу підготувати сталу структуру бази даних, для взаємодії з якою вже і буде сторюватись додаток.

Перш за все необхідно визначити які сутності будуть у нашому проєкті, щоб створити для кожної з них таблицю, а також допоміжні таблиці, які будуть забезпечувати нашому додатку правильне функціонування, і забезпечувати міцний зв’язок між таблицями.

Першою сутністю нашого додатку буде user, користувач, який матиме наступні поля:

1. Id(int);
2. Mail(string);
3. Login(string);
4. Password(string);
5. RoleID(FK);
6. Registration date(datetime);
7. Last login date(datetime);
8. Avatar(url);
9. Status(string);
10. Language(string).

Перейдемо до наступної сутності, до книги, book, у якої будуть присутні наступні поля:

1. Id(int);
2. Title(string);
3. AuthorID(FK);
4. CategoryID(FK);
5. Price(decimal);
6. DiscountID(FK);
7. Bookfile(file);
8. Filename(string);
9. Description(string).

Перейдемо до третьої, заключної основної сутності, у нашій базі даних, а саме до замовлення, Order:

1. Id(int);
2. UserID(FK);
3. Orderdate(datetime);
4. Totalamount(decimal);
5. Status(string).

Також, для функціонування системи необхідно буде створити також додаткові таблиці у базі даних.

Почнемо з усіх необхідних таблиць, крім трьох основних для покупки книги, першою таблицею буде Orderitem, замовлений прдемет:

1. Id(int);
2. OrderID(FK);
3. BookID(FK);
4. Priceatpurchase(decimal);
5. Discountapplied(decimal);
6. Finalprice(decimal).

Далі таблиця авторів, author, щоб користувачі могли бачити автора книги, і наприклад натиснувши на його ім’я могли знайти усі його книги:

1. Id(int);
2. Name(string);
3. Biography(string).

Також у нашій базі даних буде ще багато різних допоміжних таблиць для роботи додатку, наприклад Category з полями:

1. Id(int);
2. Name(string);
3. Description(string).

Таблиця розділів Chapter, адже деякі типи літератури не виходять повністю:

1. Id(int);
2. BookID(FK);
3. ChapterNumber(double);
4. Title(string);
5. Content(text).

Також ще такі другорядні таблиці як:

1. Discount;
2. Ratingandcomment;
3. Wishlist;
4. Readingprogress;
5. Userbooks;
6. Notification.

Отже, коли ми визначили і створили усі необхідні нам сутності, опишемо зв’язки між ними.

Почнемо з самої головної сутності, User, вона формує багато зв’язків з усіма іншими елементами системи, наприклад багато до одного з таблицею Role, адже у одніїє ролі може бути багато власників.

Користувач також буде мати прив’язку багато до багатьох до знижок, адже в одного користувача може бути декілька запропонованих персональних акцій, один до багатьох через окрему таблицю UserBook, користувач по ID буде прив’язаний до книг, які він має придбані, також відповідно зв’язком один до багатьох користувач має зв’язок з таблицею коментарів і оцінок, так як він може залишити багато оцінок різним книгам.

Перейдемо до зв’язків для реалізацї торгівлі та проведення транзакцій, Система купівлі книг вимагає чіткої структури, реалізованої через делегацію умов зв’язку з книгами, які будуть купуватись через просту сутність OrderItem. Таблиця Order має зв'язок один до багатьох з таблицею OrderItem. Кожне замовлення складається з однієї або декількох позицій, які деталізуються у таблиці OrderItem. Кожен запис у OrderItem містить зовнішній ключ OrderID, що прив'язує його до конкретного замовлення.

Далі, кожен замовлений предмет у таблиці OrderItem сам по собі пов'язаний з однією конкретною книгою через зовнішній ключ BookID. Це створює зв'язок один до багатьох між Book та OrderItem, оскільки одна книга може бути продана у складі багатьох різних замовлень. Такий двосторонній механізм гарантує фіксацію історії продажів, цін та знижок на момент транзакції.

Також розберемо для прикладу декілька другорядних зв’язків які теж повинні бути для вдалого загального функціонування нашого сайту:

Таблиця прогресу читання, яка відповідає за збереження прогресу читання користувача має зв’язок з юзером та з певною книгою, що дозволить користувачу з комфортом повертатись до читання з того місця, де він закінчив читати у попередній сесії.

Таблиця списків бажаного, що прив’язана до користувача зв’язком багато до одного, що дозволить користувачу створювати власні добірки книг, які він планує потім придбати.

2.3 Проєктування інтерфейсу

Дизайн інтерфейсу як і вебсайту, так і десктопного додатку є одним із головних факторів його успіху на ринку, адже який би функціональний продукт не був, якщо він буде не зручний користувачу та не красивий, користувач банально не стане його використовувати.

Отже для нашого вебсайту з продажу електронних книг необхідно розробити сучасний дизайн, який був би стильним і водночас дозволив би користувачу інтуїтивно розуміти як користуватись сайтом.

Було вирішено працювати у стилі мінімалістичному, зараз дуже популярному за свою зручність, а саме flat дизайну, який передбачає у собі відсутність об’ємних форм, анімацій, максимальну простоту дизайну для найшвидшої адаптації користувача.

Враховуючи обраний нами стиль дизайну, кольорова схема додатку буде виконана у білих, пустельних тонах, що заспокоює та не настільки сильно навантажує зір як використання темних або яскравих кольорових схем.

Фонові поверхні будуть зроблені білим кольором, акцентним буде виступати м’яко жовтий колір, заспокоюючий користувача, а колір тексту буде стандартним, чорного кольору для найкращого сприйняття, також буде можливість змінити колір сайту на темну тему, що зробить текст білим, на чорних тонів фоні.

Стандартний текст буде шрифту Times New Roman, а розмір шрифтів буде підібрано відповідно до тексту в книгах:

1. заголовки: 26 рх;
2. підзаголовки: 18 рх;
3. основний текст: 14 рх.

Також звісно ми у рідері надамо змогу користувачу самостійно обирати шрифт та розмір тексту, відповідно до його бажаних потреб.

Для кращої візуалізації було прийнято рішення створити макет того, як буде виглядати рідер на вебсайті(рисунок 2.3):

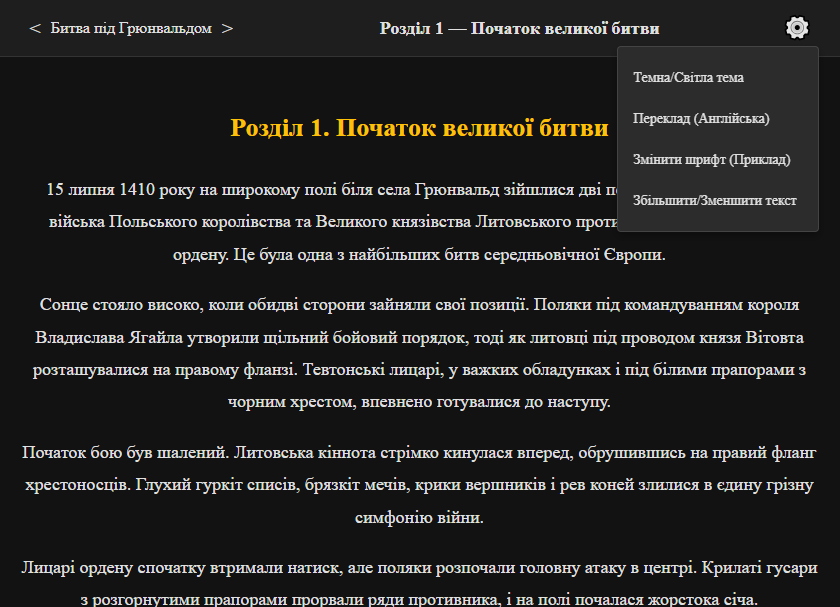


Рис. 2.3 – Макет рідера вебсайту у темній темі

Перейдемо далі до каталогу книг, фільтр було вирішено розмістити ліворуч від книг, щоб користувач завжди міг його налаштувати, і не шукав його, також на макеті(Рисунок 2.4) занадто нерівна пропорція, але після додавання книг, та розробки реального рішення, буде адаптація під різні екрани, і пропорція приблизно 20% на фільтри літератури буде застосована:

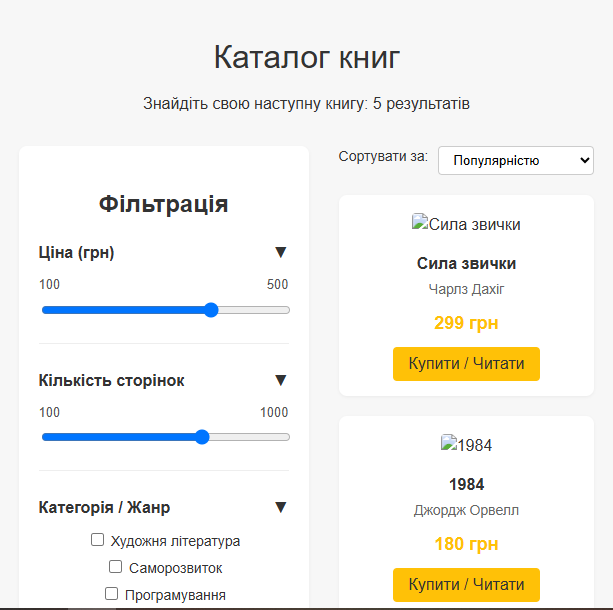


Рис. 2.4 – Каталог на розроблюваному вебсайті

Далі реалізуємо макет сторінки вибраного товару, адже вона теж неймовірно важлива у нашому додатку(рисунок 2.5):

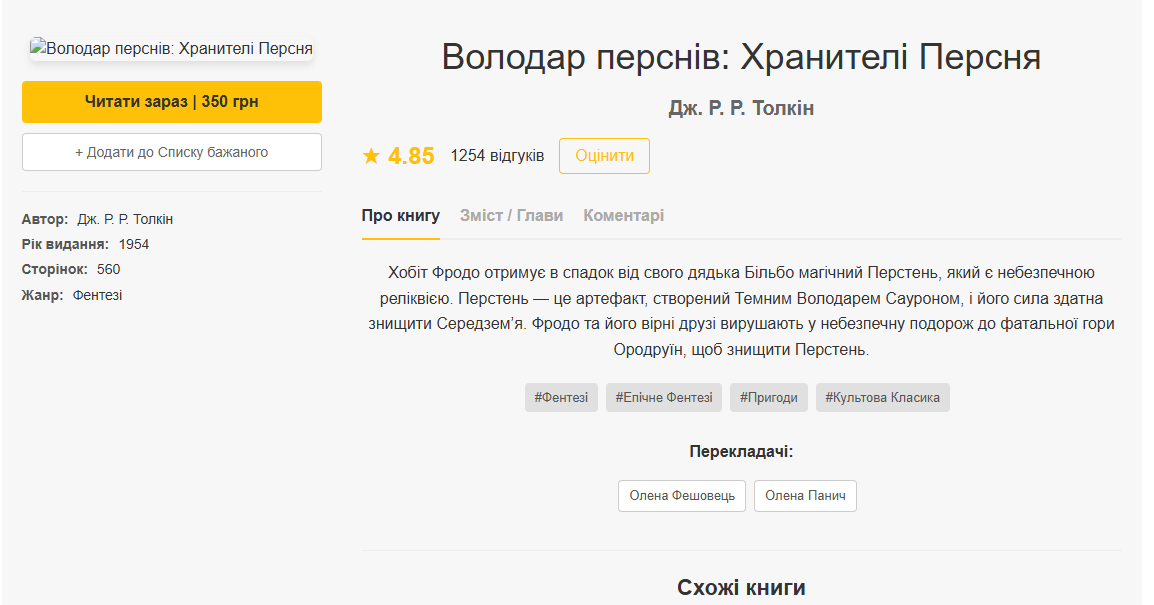


Рис. 2.5 – Макет вибраного одного товару на вебсайті

Також створимо макет для сторінки автора, де будуть видні всі його книги та коротка біографія(рисунок 2.6):

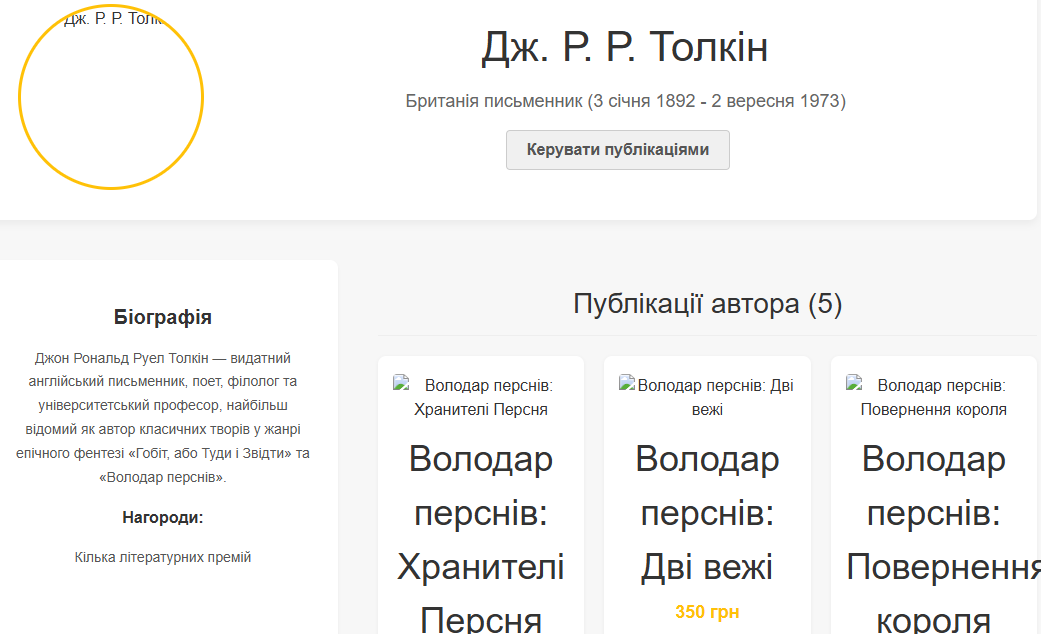


Рис. 2.6 – Макет сторінки письменника

2.4 Вибір стеку технологій

У наш час існує дуже багато можливих технологій, якими можна виконати поставлену нами собі задачу, це стосується усіх аспектів додатку, як бекенду, так і фронтенду, і навіть бази даних.

Почнемо з бази даних, адже саме база даних це серце проєкту, біля якого вже створюється бекенд та фронтенд для роботи з цими даними.

Для кращої наглядності зобразимо відмінності різних баз даних, щоб було зрозуміліше чому була обрана саме база даних MySQL.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерій | MySQL(обрано) | PostgreSQL | MSSQL | MongoDB |
| Тип СУБД та модель | Реляційна | Реляційна | Реляційна | Документна |
| Вартість | Open source | Open source | Платна ліцензія | Open source |
| Продуктивність | Висока | Висока, але повільніша ніж у інших реляційних | Висока | Висока |
| Спільнота та сумісність | Велика спільнота, найкраща сумісність | Велика спільнота, важка інтеграція | Мала спільнота,сумісність, обмежена лише оточенням Microsoft | Велика спільнота, низька сумісність |

Таблиця 2.2 – Порівняння баз даних

Отже, як ми бачимо з порівняння у таблиці 2.х, нашому додатку найкраще підходить MySQL, адже вона сумісна з усіма іншими технологіями, не має проблем з певними видами запитів, також реляційна структура дозволить зарання спланувати і налаштувати багато опцій, попередивши наперед багато можливих помилок.

Після вибору бази даних для нашого додатку, ми можемо перейти до вибору бекенду, що є неймовірно важливим етапом, адже він не лише визначить напряму який потім буде фронтенд, а й більшість можливостей для проекту, у плані масштабованості, стабільності та багато чого ще, гарантує саме цей вибір технології, знову вдамося до візуалізації варіантів таблицею та розглянемо чому саме був обраний ASP.NET як бекенд технологія для нашого додатку.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерій | ASP.NET Core | Laravel | Node.js | Gin |
| Продуктивність | Надвисока | Висока | Висока | Надвисока |
| Безпека та типізація | Найвищий рівень | Середній/високий рівень | Високий рівень | Високий рівень |
| Розробка і інтеграція АРІ | Ідеальне створення АРІ | Добре підходить для монолітної архітектури(blade) | Чудове створення АРІ | Швидке розгортання |
| Масштабованість | Чудово масштабується та підтримує мікросервіси | Високий рівень | Високий рівень | Високий рівень |

Таблиця 2.3 – Порівняння можливих бекенд рішень

Отже, аналізуючи таблицю 2.3, ми можемо дійти до висновку що для даного проєкту ASP.NET Core є ідеальним вибором, завдяки тому що він або на рівні з іншими можливими фреймоворками, або краще них виконає задачу по створенню MVC архітектури для нашого додатку, що позитивно відобразиться на усіх можливих аспектах нашої серверної частини додатку, і дозволить легше виконати усі нефункціональні вимоги.

Вибравши базу даних та бекенд для нашого додатку, перейдемо до фронтенду, який також є дуже важливою частиною нашого додатку, адже відповідатиме за можливості, які ми реалізуємо для очей користувача, і як вони будуть інтегруватись з нашою серверною частиною на ASP.NET.

Для вибору знову використаємо візуалізацію за допомогою таблиці, щоб порівняти усі можливі варіанти.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерій | React(Обрано) | Vue.js | Angular | Svelte |
| Архітектура та інтерактивність | SPA | SPA | MPA/SPA | SPA |
| Спільнота та екосистема | Найбільша і найактивніша спільнота | Велика спільнота | Підтримується Google, але більше для корпоративних проєктів | Менша спільнота, наразі, але активно зростає |
| Складність | Низький поріг входу | Дуже простий для початківців | Високий поріг входу | Низький поріг входу |
| Продуктивність | Висока | Висока | Середня | Висока |

Таблиця 2.4 – Порівняння фронтенд рішень

Отже, як ми можемо дізнатись з таблиці 2.4 React є цілком конкуренто спроможний, адже не поступається іншим фреймворкам, і враховуючи попередній досвід роботи, було прийняте рішення обрати саме його. Також для кращої взаємодії під час роботи по об’єднанню фронтенду та бекенду буде використовуватись інструмент для документації Swagger.

Також для контролю технічними процесами буде використовуватись систему контролю версій GitHub.

Висновки

.

Список використаних джерел

1. KSD. Офіційний вебсайт книгарні. – Режим доступу: https://www.ksd.ua.
2. Epub. Вебсайт магазин по продажу електронних книг – Режим доступу: https://www.epub.com.ua.
3. Wikipedia Діаграма діяльності [електронний ресурс]. - Режим доступу: <https://uk.wikipedia.org/wiki/Діаграма_діяльності>
4. Мова програмування C# 9 і платформа .NET 5: основні принципи і практики програмування, том 1, 10-е видання
5. ДСТУ 3008-95 "Документація. Звіти в сфері науки і техніки. Структура і правила оформлення" – Державний стандарт України.
6. Вермулен, М. (2018). **Принципи, Практики та Шаблони Впровадження Залежностей**. Manning Publications.
7. Мартін, Р. (2019). Чиста архітектура. Мистецтво розробки програмного забезпечення
8. Річардс, М. (2023). Сучасний підхід до програмної архітектури.Складні компроміси