ДИПЛОМНА РОБОТА

БАКАЛАВРА

ДРБ. ІП - 85.00.00.00 ПЗ

Група ІП-20-3

Ониськів Юрій Богданович

2024

Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

УДК _____004.4_____

ДИПЛОМНА РОБОТА БАКАЛАВРА

Тема: " Розробка та вдосконалення моделей, методів і алгоритмів побудови ВЕБ-аггрегаторів "

(назва згідно з наказом ректора)

Спеціальність – 121 "Інженерія програмного забезпечення"

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДРБ.ІП - 85.00.00.00 ПЗ

(позначення)

Студент

ІП-20-3 _____ __/ Ониськів Ю.Б. /
(шифр групи) (підпис) (дата)(розшифровка підпису)

Керівник роботи

Консультанти:

асистент /Л.М. Гобир/
(посада) (підпис) (дата) (розшифровка підпису)

з питань нормоконтролю

Допускається до захисту
Завідувач кафедри

асистент /М.М. Піх/
(посада) (підпис) (дата)(розшифровка підпису)

доцент., к.т.н. /В.В. Бандура/
(посада) (підпис) (дата) (розшифровка підпису)

2024
(рік)

Івано-Франківський національний технічний університет нафти і газу

Інститут, факультет інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти бакалавр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
Зав. кафедрою **ІПЗ**
доцент. В.В. Бандура
“ ____ ” _____2024 р.

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ БАКАЛАВРА СТУДЕНТОВІ

Ониськів Юрій Богданович
(прізвище, ім’я, по-батькові)

1. Тема проекту (роботи) "Розробка та вдосконалення моделей, методів і алгоритмів побудови ВЕБ-аггрегаторів"

керівник проекту (роботи) асистент Гобир Лідія Мирославівна

затверджені наказом вищого навчального закладу від **“** 07 ” червня 2024 р. № 347/7

2. Строк подання студентом проекту (роботи) червня 2024 р.

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

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

1 Вступ до проблеми
2 Огляд існуючих концепцій , рішень та сервісів в даній області
3 Побудова моделі або алгоритму власного рішення
4 Документування реалізації власного оригінального рішення вибраними засобами

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

1 Інтерфейс запропонованої діалогової системи (рис. 3.1, ст. 48)
2 Класи, створені для запропонованої моделі онтології (рис. 3.2, ст. 49)
3 Властивості об’єкта, створені для запропонованої моделі онтології (рис. 3.3, ст. 49)
4 Властивості даних, створені для запропонованої моделі онтології (рис. 3.4, ст. 50)
5 Примірники , створені для запропонованої моделі (рис. 3.6, ст. 51)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

2. Дата видачі завдання _____

Керівник _____ Гобир Л.М.
(підпис)

Завдання прийняв до виконання _____ Ониськів Ю.Б.
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту	Примітка
1	Визначення та обгрунтування теми роботи	27.05.2024	виконано
2	Огляд існуючих концепцій, рішень та сервісів в даній області	31.05.2024	виконано
3	Побудова моделі або алгоритму власного рішення	04.06.2024	виконано
4	Документування реалізації власного оригінального рішення вибраними засобами	09.06.2024	виконано
5	Оформлення пояснювальної записки кваліфікаційної роботи	12.06.2024	виконано

Студент _____ Ониськів Ю.Б.
(підпис)

Керівник роботи _____ Гобир Л.М.
(підпис)

АНОТАЦІЯ

Пояснювальна записка дипломного проєкту складається з чотирьох розділів, містить 57 таблиць, 9 рисунків та 10 джерел – загалом 59 сторінки.

Мета: Впровадження і супроводження веб-платформи для написання та перегляду рецензій.

Об'єкт дослідження: Програмне забезпечення для написання та перегляду рецензій на різні продукти та функцію для пропонування випадкового продукту за критеріями користувача. Ця функція підходить для так, хто не знає що обрати. В першому розділі даної роботи було досліджено предметну область. Було проведено аналіз відомих технічних рішень та успішних аналогів.

В другому розділі описано моделювання та конструювання додатку. Була розроблена архітектура застосунку, описані програмні продукти, які були використані.

В третьому розділі було складено тестовий план, визначено об’єкти тестування, визначено підходи до тестування та описано основні задачі тестування, наведено опис впровадження Web-додатку та представлено керівництво користувача.

КЛЮЧОВІ СЛОВА: WEB, ASP NET CORE, MVC, ENTITY FRAMEWORK, THREE-TIER ARCHITECTURE, БАЗА ДАНИХ, VISUAL STUDIO.

ANNOTATION

The explanatory note of the diploma project consists of four sections, contains 57 tables, 9 figures and 10 sources - a total of 59 pages.

The diploma project is devoted to: Web application for writing and viewing reviews of various products.

Purpose: Implementation and maintenance of a web platform for writing and viewing reviews.

Research object: Software for writing and viewing reviews of various products and a function for suggesting a random product based on the user's criteria. This function is suitable for those who do not know what to choose. The subject area was investigated in the first section of this work. An analysis of known technical solutions and successful analogues was carried out.

The second chapter describes the modeling and construction of the application. The application architecture was developed, the software products that were used were described.

In the third section, a test plan was drawn up, test objects were defined, testing approaches were defined and the main tasks of testing were described, a description of the implementation of the Web application was given, and a user guide was presented.

KEYWORDS: WEB, ASP NET CORE, MVC, ENTITY FRAMEWORK, THREE-TIER ARCHITECTURE, DATABASE, VISUAL STUDIO.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 6

ВСТУП 7

РОЗДІЛ 1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 9

1.1 Загальні положення **..... 9**

1.2 Аналіз успішних IT-проектів **..... 10**

1.3 Аналіз вимог до програмного забезпечення **..... 14**

РОЗДІЛ 2. МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 27

2.1 Моделювання та аналіз програмного забезпечення **..... 27**

2.2 Архітектура програмного забезпечення **..... 28**

2.4 Вибір засобів розробки **..... 44**

2.5 Аналіз безпеки даних **..... 46**

РОЗДІЛ 3. АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 48

3.1 Аналіз безпеки даних **..... 47**

3.2 Опис процесів тестування **..... 48**

3.3 Результати тестування **..... 54**

3.4 Розгортання та робота програмного забезпечення **..... 55**

ВИСНОВКИ 57

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА 58

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ТЗ – Технічне завдання

БД – База даних

СУБД - Система управління базами даних

ПЗ - Програмне забезпечення

ВСТУП

У сучасному світі існує дуже багато різних фільмів, серіалів, ігор, книг та інших речей. Але коли ти подивився\прочитав\спробував щось і тобі це не сподобалось - це не дуже приємно і ти витрачаєш час в пусту. Тому не на все коштує витрачати час. Залиш свою думку про цю річ для інших людей. Та наступний раз подивись відгуки про те, що хочеш спробувати. Або якщо не знаєш що обрати, то можна задати параметри які хочеш і тобі випадково запропонують на вибір фільм\серіал\книгу та інше.

На сьогоднішній день користувачам пропонуються багато різних вебплатформ для рецензій. Проаналізував існуючи агрегатори рецензій, не була знайдено платформу у якій будуть можливості подивитись та залишити рецензію на всі типи розважального контенту. Весь контент не подивишся\послухаєш\почитаєш, тому потрібна допомога у виборі.

Актуальність теми: на сьогоднішній день, коли в нашій країні йдуть бойові дії, хочеться трохи відпочити та заспокоїтися подивившись\прочитавши\послухавши і т.д щось. Щоб не витрачати час на те, що тобі може не сподобатися і буде створений мій веб-застосунок.

Мета розробки: впровадження і супроводження веб-платформи для написання та перегляду рецензій.

Завдання розробки:

− розробка сервісу для написання та редагування своїх рецензій, та перегляду рецензій інших користувачів;

− розробка алгоритму вибору фільму\серіалу\музики\книги\гри за заданими користувачем критеріями;

− створення сервісу для керування ролями користувачів (адміністратор, модератор, журналіст, звичайний користувач);

− розробка сервісу для керування категоріями та підкатегоріями; розробка сервісу для додавання продуктів для відповідних категорій;

- розробка сервісу для додавання вподобайок на рецензії інших користувачів;
- розробка сервісу для зміни теми сайту.

РОЗДІЛ 1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Загальні положення

Доступ до веб-додатків на сьогоднішній час має кожна людина майже відусюди та майже весь час. Перевагою веб застосунку те , що він є міжплатформовим сервісом. Тобто не важливо який пристрій використовує користувач, головне - пристрій повинен мати можливість виходити в інтернет та відкривати браузер. Внаслідок цієї універсальності веб-додатки популярні.

Людям властиве оцінкове сприйняття, чим здавна користуються маркетологи у різних цілях. В останні роки багато хто вже не сприймає критиків та оцінки якимось вагомим аргументом на користь реальної якості медіа продукту. Тим не менш, завжди можна спертися на думки «впливових» персон, або просто кивнути на сухі цифри. Для цього і існують так звані агрегатори — сайти, на яких збирають критичні відгуки преси, оцінки ігор та рекомендацій до покупки.

На цей час вплив сайтів-агрегаторів рецензій на економічні показники компаній, що виготовляють фільми\ігри\серіали\книги\музику тощо, стосовно яких пишуться огляди, дуже великий. Деякі компанії навіть ввели систему заохочення своїх співробітників. Суть цієї системи у тому, що сума бонусів для них залежить від оцінок користувачів. На фондових ринках ціни на акції компаній також показали залежність від отриманого рейтингу товарів компанії.

Основна мета агрегатора рецензій - це написання, редагування, перегляд рецензій та пропонування творів на основі заданих параметрів.

Веб платформа повинна бути простою у використанні, мати можливість створювати\редагувати\переглядати рецензії , встановлювати пріоритети для запропонованих творів, мати можливість надання користувачам ролей. Також застосунок буде мати можливість змінювати тему сайту.

Змістовний опис і аналіз предметної області

Агрегатор рецензій — система для зберігання та збирання рецензій та оглядів на різні товари й послуги (фільми, книги, відеоігри тощо). Система

агрегації рецензій надає користувачеві можливість зручно порівнювати різні рецензії на одну і ту саму річ.

Веб-платформа “агрегатор рецензій” повинна показувати продукти за різними категоріями, наприклад фільми, серіали, ігри, музика, книги і т.д. та фільтрувати їх та рецензії за різними параметрами, наприклад за рейтингом або новизною. Також додаток повинен надавати можливість створення та редагування рецензій.

Якщо користувач не знає що обрати, то для нього буде зроблена функція “Мені пощастить”. Ця функція видає за заданими критеріями (або якщо користувач не ввів критерії то і без них) продукт. Як би не видавалась ця функція непотрібною, але є люди для яких важко вибрати щось певне, і вони хочуть відати вибір комусь іншому.

Повинно реалізувати надійність конструкції програмного забезпечення. Це означає, що додаток має підтримувати працездатність протягом заданого проміжку часу та при заданих умовах.

1.2 Аналіз успішних ІТ-проектів

Аналіз відомих технічних рішень

Архітектуру веб-застосунку можна поділити на три частини: Presentation layer (рівень представлення), Business layer (рівень бізнес-логіки) та Data Access layer (рівень доступу до даних). Для кожного шару можна зробити різну реалізацію.

Data Access layer (рівень доступу до даних): іноді званий рівнем бази даних, рівнем доступу до даних або серверною частиною, це місце, де інформація, оброблена додатком, зберігається і управляється. Це може бути система управління реляційною базою даних, такою як PostgreSQL, MySQL, MariaDB, Oracle, DB2, Informix або Microsoft SQL Server, або сервер бази даних NoSQL, такий як Cassandra, CouchDB або MongoDB. Для даного додатку було вирішено обрати MSSQL так, як він дуже добре підходить для застосунків створених на .NET. Для даного веб-сервісу було обрано мову програмування C#

на платформі .NET. Також на цьому рівні для зв’язку з БД я буду використовувати фреймворк Entity Framework Core.

Business layer (рівень бізнес-логіки): містить компоненти, які виконують обробку отриманих даних від рівня представлення, реалізує необхідну логіку веб-застосунку, взаємодіє з базою даних та передає рівню представлення результат обробки. C# на платформі .NET разом з компонентом LINQ було використано для реалізації обробки даних з бази для надання їх на шар представлення, наприклад сортування та фільтрація продуктів для обробки інформації що надійшла з Data Access layer.

Presentation layer (рівень представлення): це інтерфейс користувача та комунікаційний рівень програми, на якій кінцевий користувач взаємодіє з додатком. Основна мета цього рівня – відобразити інформацію та зібрати дані від користувача. Цей рівень верхнього рівня може працювати, наприклад, у веб-браузері як настільна програма або графічний інтерфейс користувача (GUI). Рівні веб-презентацій зазвичай розробляються за допомогою HTML, CSS та JavaScript. Настільні програми можуть бути написані різними мовами залежно від платформи. Для цього рівня було вирішено використовувати ASP.NET Core MVC а також для фронтенду один з фреймворків JavaScript. Є доволі багато різних фреймворків для фронтенду:

AngularJS - це клієнтський JavaScript-фреймворк, що базується на MVC. Це відкритий вихідний код, який підтримується та підтримується Google. AngularJS як "Angular" або "Angular.js" був спочатку випущений у 2009 році з метою покращення процесу розробки та тестування.

React — це бібліотека JavaScript з відкритим кодом для створення зовнішніх інтерфейсів користувача. На відміну від інших бібліотек JavaScript, що надають повноцінну платформу додатків, React орієнтується виключно на створення уявлень додатків через інкапсульовані одиниці (називаються компонентами), які зберігають стан і генерують елементи інтерфейсу користувача. Ви можете розмістити окремий компонент на веб-сторінці або вкласти ієрархії компонентів для створення складного інтерфейсу користувача.

Також є готові бібліотеки для фронтенду, наприклад Bootstrap. Bootstrap – це фреймворк з відкритим відкритим вихідним кодом для створення вебдодатків. Цей фреймворк був створений компанією Twitter у 2011 році. Бібліотека включає в себе CSS шаблони та JavaScript скрипти для роботи з ними. Саме Bootstrap використовується у цьому дипломному проекті.

Аналіз відомих програмних продуктів

На даний час існує доволі велика кількість веб-застосунків агрегаторів рецензій, проте немає застосунку для, де будуть рецензії на всі види творів(фільми, серіали, ігри, музика, книги, комікси, манга, тощо.). Далі наведу приклади існуючих проектів.

Metacritic - найбільш багатогранний агрегатор рецензій. Цей веб-застосунок збирає відгуки про музичні альбоми, відеоігри, фільми і телевізійні шоу. На основі зібраних відгуків вираховується середня числова оцінка. Робота над Metacritic почалась у 1999 році трьома одногрупниками з Університету Південної Кароліни. Джейсон Дейтц подав ідею, а Марк Дойл із сестрою Джулією Дойл Робертс допомогли йому створити сайт, відібрати видання та розробити систему оцінки. Після двох років розробки у січні 2001 року проект був запущений.

Рейтинг на Metacritic є виваженою середньою оцінкою критиків. Деяким виданням надається більша вага «через їх значущість». Персонал Metacritic зазначав, що він не збирається розкривати відносну вагу, привласнену кожному з критиків. Якщо критик не включає в свою рецензію числову оцінку, співробітник Metacritic спробує її вгадати, спираючись на тон і зміст матеріалу, а також на інформацію про переваги рецензента. Для фільмів та музичних альбомів такий метод загалом підходить, хоча іноді й дає збій.



Рисунок 1.1 - Metacritic логотип

Rotten Tomatoes - збирає новини, рецензії та інформацію лише про кінематограф. У 1998 році шанувальник фільмів з Джекі Чаном Сен Дуонг вирішив зібрати всі рецензії на картини з улюбленим актором, опубліковані в США, і придумав сайт, на якому на основі таких рецензій виставлялися б оцінки фільмам. Назву він узяв, згадавши стару традицію закидати тухлими помідорами неугодних артистів.

Дипломовані члени різних асоціацій кінокритиків пишуть свої рецензії, після чого персонал Rotten Tomatoes оцінює огляд як позитивний("свіжий") або негативний("тухлий"). Кожен фільм на Rotten Tomatoes ще має рейтинг глядачів, він формується з оцінок зареєстрованих користувачів сайту.

Агрегатор Metacritic, головний конкурент Rotten Tomatoes, наприклад, влаштований набагато складніше: він надає кожній рецензії свою оцінку, а потім розраховує середній рейтинг. Rotten Tomatoes став дуже популярним агрегатором рецензій насамперед тому, що він простий та зрозумілий, у ньому немає нічого зайвого.

Рисунок 1.2 - Rotten Tomatoes логотип

GameStats - агрегатор рецензій присвячений ігровій індустрії. Створений медіаконгломератом IGN Entertainment у 2002 року. Головним конкурентом для GameStats є Game Rankings який належить CNET.

Щоб відрізнятися від аналогічних веб-сайтів, GameStats включає «GPM», систему, яка розшифровується як «Вимірник популярності ігор». Ця система, серед іншого, відстежує, як часто здійснюється доступ до сторінки гри. Як і

очікувалося, він також включає середню оцінку критиків та середню оцінку фанатів, поєднуючи їх у «GS Score». Однак деякі ігри вже мають рейтинги ще до їх випуску, що ставить під сумнів рейтинги фанатів.

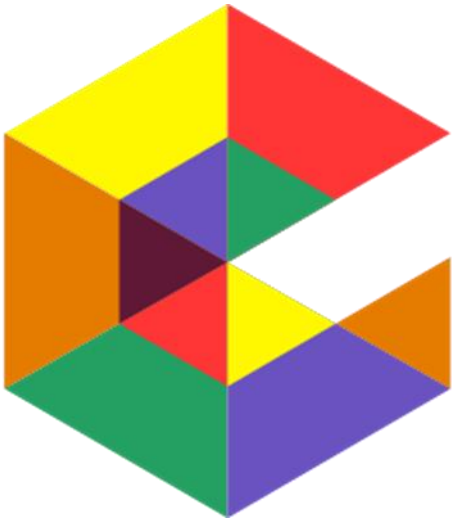


Рисунок 1.3 - GameStats логотип

Хоча це не всі існуючі агрегатори рецензій але вже можна помітити, що на даний час не існує веб-платформи для охоплення всіх типів творів. Саме тому було вирішено написати веб-застосунок який охопить як найбільше типів продуктів.

1.3 Аналіз вимог до програмного забезпечення

Розроблення функціональних вимог

Для даного проєкту сформовані наступні функціональні вимоги:

- створення рецензій;
- редагування рецензій;
- видалення рецензій;
- пошук продуктів; - фільтр продуктів;
- реєстрація користувачів;
- створення нової ролі;
- перегляд рецензій написаних певним користувачем;
- редагування ролей користувачів;

- зміна теми у додатку;
- додавання продуктів;
- редагування продуктів;
- видалення продуктів;
- функція для випадкового пропонування продуктів;
- видалення користувачів;
- фільтр рецензій;
- створення нової категорії;
- видалення ролі;
- видалення категорії;
- створення підкатегорій;
- видалення підкатегорій;
- надання категоріям підкатегорій; - додавання оцінки на рецензію.

Таблиця 1.1 - Варіант використання UC001

Назва	Створення рецензій
Опис	Користувач додатка створює рецензію на продукт
Учасники	Користувач зареєстрований
Передумови	Відкрита сторінка продукту на який буде написана рецензія. Користувач зареєстрований
Після умови	Рецензію буде додано до сторінки продукту
Сценарій	Користувач заходить на сторінку продукту, ставить оцінку та натискає на кнопку «Create review»

Таблиця 1.2 - Варіант використання UC002

Назва	Редагування рецензій
Опис	Користувач додатка редагує рецензію на продукт

Учасники	Звичайний користувач(тільки свою рецензію), Критик(тільки свою рецензію), Модератор(тільки свою рецензію) та Адміністратор(тільки свою рецензію)
Передумови	Відкрита сторінка продукту на який написана рецензія або сторінка з рецензіями користувача, та обрана відповідна рецензія.
Після умови	Рецензію буде відредагована
Сценарій	Користувач заходить на сторінку продукту, змінює свою оцінку та натискає на кнопку «Update review»

Таблиця 1.3 - Варіант використання UC003

Назва	Видалення рецензій
Опис	Користувач додатка видаляє рецензію на продукт
Учасники	Звичайний користувач(тільки свою рецензію), Критик(тільки свою рецензію), Модератор та Адміністратор
Передумови	Відкрита сторінка продукту на який написана рецензія або сторінка з рецензіями користувача, та обрана відповідна рецензія.
Після умови	Рецензію буде видалено
Сценарій	Користувач заходить на сторінку продукту, натискає «Delete review»

Таблиця 1.4 - Варіант використання UC003

Назва	Пошук продуктів
Опис	Додаток шукає за назвою серед усіх продукт.
Учасники	Усі
Передумови	Пошук усіх існуючих продуктів за назвою використовую пошуковий рядок
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям пошуку
Сценарій	У поле пошуку користувач вводить назву або частину назви продукту та натискає «Enter»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.

Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.

Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.

Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.

Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.

Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.
Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

Таблиця 1.5- Варіант використання UC005

Назва	Фільтр продуктів
Опис	Додаток фільтрує продукти за заданими користувачем параметрами
Учасники	Усі
Передумови	Фільтрація усіх існуючих продуктів за заданими користувачем параметрами.

Після умови	Користувачу виводяться усі продукти які задовольняють критеріям фільтрації
Сценарій	Користувач у фільтрі обирає критерії та натискає кнопку «Filter»

Таблиця 1.6 - Варіант використання UC006

Назва	Реєстрація користувача
Опис	Реєстрація користувача
Учасники	Усі
Передумови	Відкрита сторінка реєстрації користувача. Користувач не зареєстрований.
Після умови	Новий користувач зареєстрований
Сценарій	Користувач на сторінці реєстрації вводить валідні дані та натискає кнопку «Register»

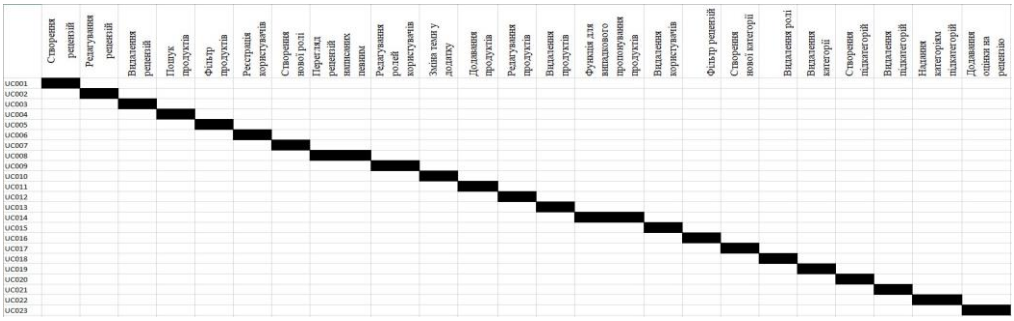


Рисунок1.4 - Матриця залежності між вимогами і варіантами використання

1.4.2 Розроблення нефункціональних вимог

Вимоги до програмної частини:

- застосування має бути розроблено мовою програмування C# використовуючи ASP.NET Core; - застосування має бути розроблено з використанням патерну MVC.

Вимоги до інтерфейсу:

- застосування має відображатись у браузері.

Апаратні та програмні вимоги:

- застосування має має працювати на таких браузерах: Firefox версії 4 і вище, Opera версії 70 і вище, Google Chrome версії 60 і вище.

Операційні вимоги:

- продуктивність:

1) кількість запитів в сервісі на секунду не перевищує 100;

2) час відповіді системи на запит статистичного аналізу не більший ніж 2 секунди;

3) час відповіді системи на запит візуалізації не більший ніж 5 секунд.

1.5 Постановка задачі

Веб-платформа має вирішувати задачі агрегатора рецензій та допомагати користувачам обирати продукт.

Мета створення даної роботи - створення веб-платформи для агрегації рецензій, яка буде охоплювати як найбільше типів продуктів(фільми\серіали\ігри\книги\комікси\манга і т.д.).

Задля того, щоб досягти мети додатком мають вирішуватися такі задачі:

− розробка сервісу для написання та редагування своїх рецензій, та перегляду рецензій інших користувачів;

− розробка алгоритму вибору фільму\серіалу\музики\книги\гри за заданими користувачем критеріями;

− створення сервісу для керування ролями користувачів (адміністратор, модератор, журналіст, звичайний користувач);

− розробка сервісу для керування категоріями та підкатегоріями; розробка сервісу для додавання продуктів для відповідних категорій;

− розробка сервісу для додавання вподобайок на рецензії інших користувачів;

− розробка сервісу для зміни теми сайту.

Висновки до розділу

В розділі 1 проведено:

- аналіз існуючих технічних засобів і відомих рішень;
- аналіз відомих програмних продуктів;
- аналіз вимог до програмного забезпечення – розроблені

функціональні та нефункціональні вимоги.

РОЗДІЛ 2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Моделювання та аналіз програмного забезпечення

Синтаксичне з'єднання можна визначити як залежність двох або більше компонентів програмного забезпечення один на одного через опору, визначених програмістом ідентифікаторів інших компонентів. Знання ідентифікаторів, безумовно, необхідні, якщо компоненти програмного забезпечення мають взаємодію, оскільки це те, що є основою поняття програмного інтерфейсу або API. Наприклад, абонент функції повинен знати ідентифікатор Callee, щоб викликати її. Однак, якби кожна атомна дія в блоці коду (наприклад, додавання) була розкладена на функцію, а відповідний ідентифікатор додає (x, y), блок коду стане занадто сильно насиченим ідентифікаторами. Таким чином, синтаксична зв'язок сильно стосується ідеї розкладання коду на менші, модульні функції, оскільки більший ступінь розкладання призводить до більш високого насичення ідентифікаторів. В ідеалі компонент програмного забезпечення врівноважує розкладання коду на функціональні дзвінки (а отже, і використання ідентифікаторів) інших компонентів з явними блоками коду.

CSS рефакторинг та підтримка інструментів дозволяють підтримувати веб-проекти на високому рівні та забезпечувати їхню продуктивність та розширюваність.

Веб-застосунок був розроблений для браузерів Google Chrome версії 60 і вище, Internet Explorer версії 10 і вище, Firefox версії 4 і вище. Застосування було розроблено використовуючи мову C# на платформі .NET. На рівні представлення було використано ASP.NET Core за патерном MVC. На інших рівнях використовував C# для реалізації бізнес логіки та для взаємодії з базою даних.

У додатку було реалізовано трьох шарову архітектуру. Ця архітектура дозволяє відокремити незалежні модулі та окремо доповнювати їх.

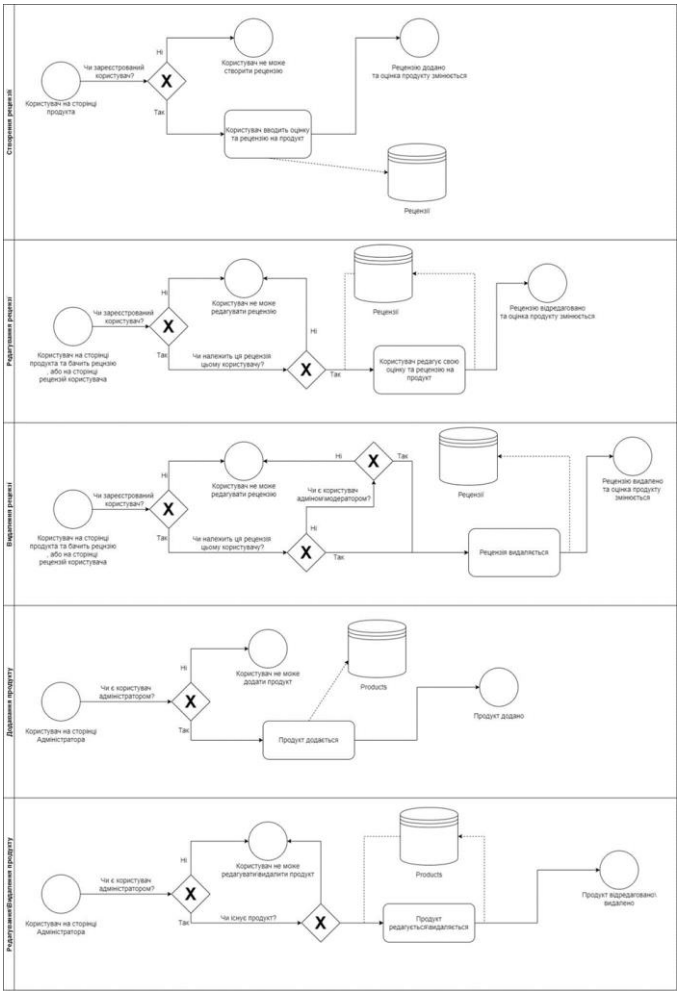


Рисунок 2.1 - Схема бізнес-процесів

2.2 Архітектура програмного забезпечення

Веб-застосунок буде використовувати трьох рівневу архітектуру. Ця архітектура добре зарекомендувала себе. Головні переваги цієї архітектури це :

- велика масштабованість;
- висока безпека;
- висока надійність.

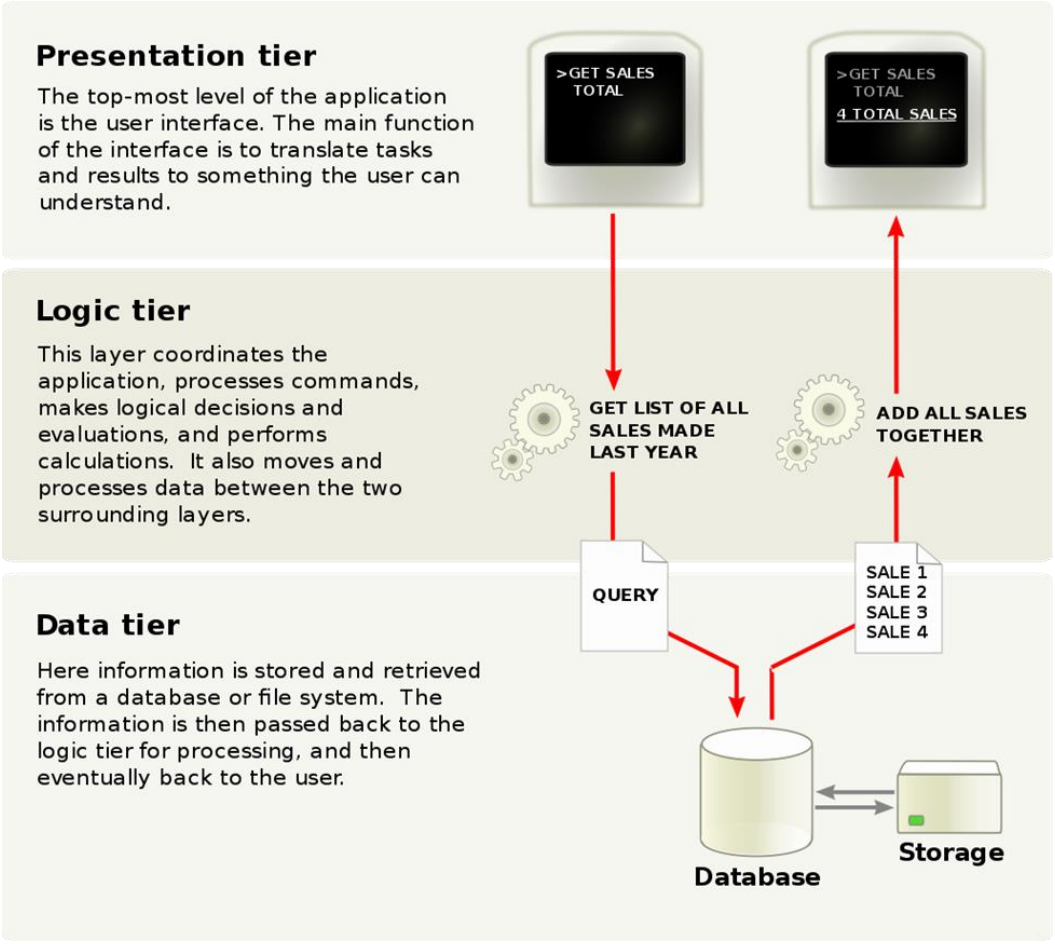


Рисунок 2.2 - Візуальна демонстрація трьох рівневої архітектури

На Data Access layer (рівень доступу до даних) використовується Entity Framework Core. На Business layer (рівень бізнес-логіки) здебільшого використовується C# та LINQ

На Presentation layer (рівень представлення) використовується ASP.NET Core за патерном MVC. Структура архітектури MVC поділяє додаток на три основні групи компонентів: моделі, представлення та контролери. Запити користувачів надсилаються до контролера, який відповідає за роботу з моделлю для виконання дій користувачів та (або) отримання результатів запитів. Контролер вибирає подання для відображення користувачеві з усіма необхідними даними моделі.

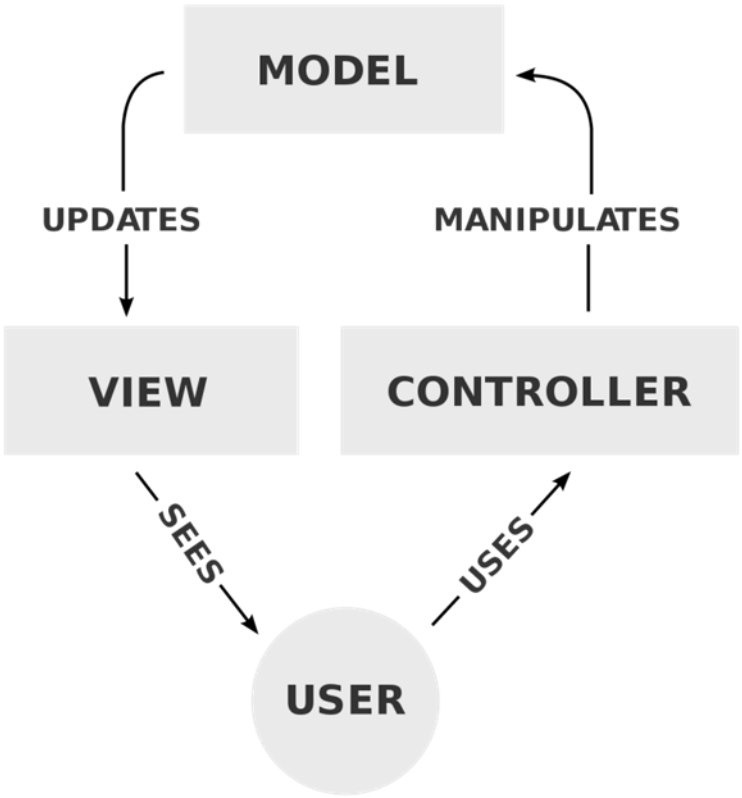


Рисунок 2.3 - Схема зв’язків між компонентами MVC

Діаграма класів та модель БД наведено в документі «Графічний матеріал». Опис структури БД наведено у пункті 2.3. 2.3 Конструювання програмного забезпечення. Наведено методи основних класів:

Клас/Інтерфейс	Рівень	Опис
DefaultContext	Data Access layer	Цей клас наслідує класс DbContext, який в свою чергу є частиною Entity Framework Core. Цей клас потрібен для підключення та роботи з базою даних.
IRepository	Data Access layer	Цей клас є інтерфейсом для роботи з сутностями бази даних
IUnitOfWork	Data Access layer	Цей клас є інтерфейсом для роботи з класами які наслідуються від IRepository
IUserRepository	Data Access layer	Цей клас є інтерфейсом для роботи з сутністю користувача бази даних
IProductService	Business layer	Цей клас є інтерфейсом для реалізації логіки взаємодії з продуктами.
IReviewService	Business layer	Цей клас є інтерфейсом для реалізації логіки взаємодії з рецензіями користувачів.
IUserService	Business layer	Цей клас є інтерфейсом для реалізації логіки взаємодії з користувачами.
ServicesModule	Business layer	Допоміжний клас для налаштування Dependency Injection
ProductsController	Presentation layer	Клас контролер MVC для роботи з продуктами
AdminController	Presentation layer	Клас контролер MVC для роботи з адмін-сторінками
UserController	Presentation layer	Клас контролер MVC для демонстрації користувачів
HomeController	Presentation layer	Стандартний клас контролер MVC

Таблиця 2.2 – Методи основних класів

Клас/Інтерфейс	Метод	Опис
DefaultContext	DefaultContext(DbContextOptio ns<DefaultConte xt> options)	Конструктор налаштування робити взаємодії з базою даних
IRepository<T>	GetAll	Отримати усі записи сутності
IRepository<T>	Get(int id)	Отримати конкретну запис сутності
IRepository<T>	Find(Func<T, bool> predicate)	Отримати усі записи сутності за заданою умовою
IRepository<T>	Create(T item)	Створити нову запис сутності у базі даних
IRepository<T>	Update(T item)	Оновити запис сутності у базі даних
IRepository<T>	Delete(int id)	Видалити запис сутності з бази даних
IUnitOfWork	save	Збереження змін у базі даних
IUserRepository<T, S>	GetAll	Отримати усіх користувачів
IUserRepository<T, S>	Get	Отримати певного користувача
IUserRepository<T, S>	Find	Отримати користувачів за заданою умовою
IUserRepository<T, S>	Update	Оновити певного користувача
IUserRepository<T, S>	Delete	Видалити певного користувача
IUserRepository<T, S>	LogIn	Зайти у систему за певного користувача
IUserRepository<T, S>	AddRolesToUser	Додати ролі певному користувачу
IUserRepository<T, S>	AddRoleToUser	Додати певну роль певному користувачу
IUserRepository<T, S>	CreateNewRole	Створити нову роль

IUserRepository<T, S>	GetAllRoles	Отримати усі ролі
IUserRepository<T, S>	FindRole	Знайти певну роль
IUserRepository<T, S>	DeleteRole	Видалити певну роль
IUserRepository<T, S>	FilterReview	Фільтрує рецензії за заданими параметрами
IUserRepository<T, S>	GetUserRoles	Отримати усі ролі певного користувача
IProductService	CreateCategory	Створення категорії
IProductService	UpdateCategory	Оновлення категорії
IProductService	DeleteCategory	Видалення категорії
IProductService	CreateSubCategory	Створення підкатегорії
IProductService	UpdateSubCategory	Оновлення підкатегорії
IProductService	DeleteSubCategory	Видалення підкатегорії
IProductService	CreateProduct	Створення продукта
IProductService	UpdateProduct	Оновлення продукта
IProductService	DeleteProduct	Видалення продукта
IProductService	GetCategories	Отримання усіх категорій

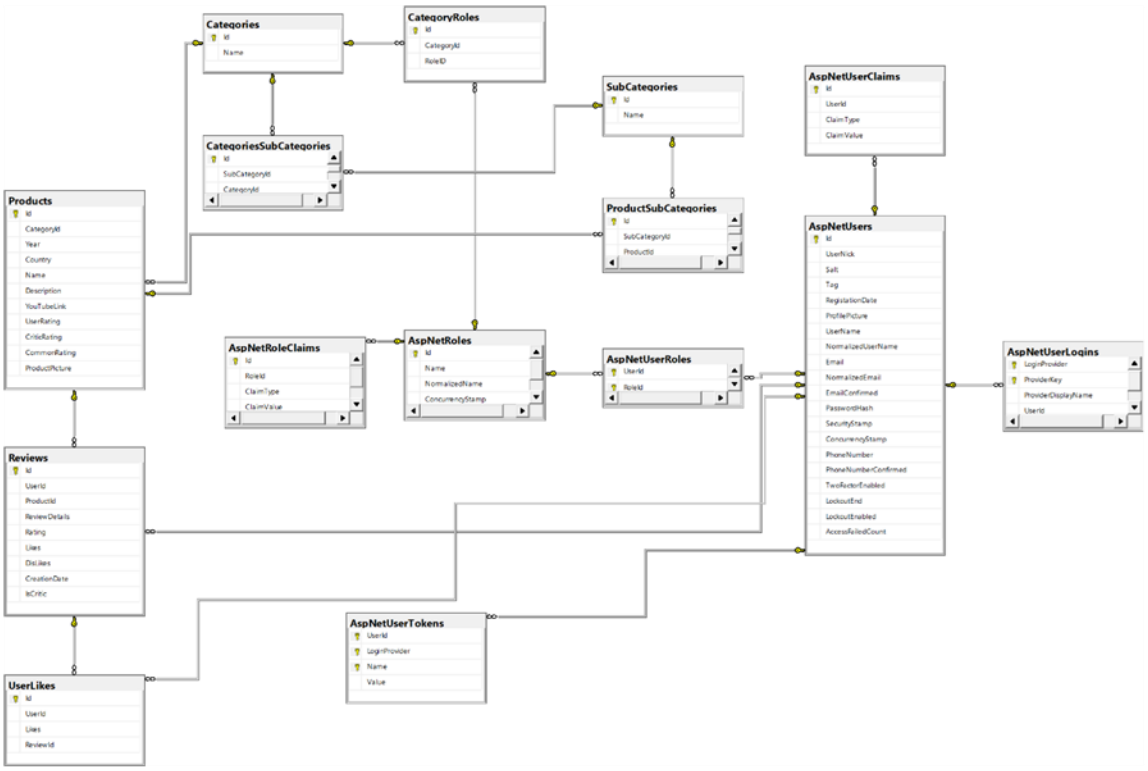
Продовження таблиці 2.2

IProductService	GetAllSubCatFo rCat	Отримання усіх підкатегорій для категорії
IProductService	FindProducts	Отримання продуктів за певним запитом
IProductService	GetAllProducts	Отримання усіх продуктів
IProductService	GetProduct	Отримання певного продукта
IReviewService	CreateReview	Створення рецензії
IReviewService	UpdateReview	Оновлення рецензії
IReviewService	DeleteReview	Видалення рецензії
IReviewService	LikeReview	Поставити вподобайку рецензії
IReviewService	DisLikeReview	Поставити невподобайку рецензії
IReviewService	GetReview	Отримати певну рецензію
IReviewService	GetAllReviewsT oProduct	Отримати усі рецензії для певного продукту
IReviewService	GetAllUserRevie w	Отримати усі рецензії певного користувача
IReviewService	FindUserReview s	Отримати рецензії за певним запитом
IUserService	CreateUser	Створити користувача
IUserService	UpdateUser	Оновити користувача
IUserService	GetUserById	Отримати користувача за id
IUserService	GetUsers	Отримати усіх користувачів
IUserService	FindUsers	Знайти певних користувачів за заданими критеріями
IUserService	DeleteUser	Видалити певного користувача
IUserService	IsUserHasveRole	Перевірка чи є певна роль у певного користувача

Продовження таблиці 2.2

ProductController	Products	Отримання продуктів
ProductController	New	Створення нового продукту
ProductController	Manage	Налаштування певного продукта
ProductController	Delete	Видалення певного продукта
UserController	Register	Реєстрація користувача
UserController	Login	Вхід у систему певним користувачем
UserController	Logout	Вихід з системи
UserController	NewReview	Створення нової рецензії
UserController	EditReview	Редагування рецензії
UserController	DeleteReview	Видалення рецензії
UserController	Like	Ставлення вподобайки певній рецензії
UserController	Dislike	Ставлення невподобайки певній рецензії
UserController	Manage	Налаштування користувача
UserController	User	Отримати дані певного користувача

Наведено таблиці бази даних


Categories
Id
Name
CategoryRoles
Id
CategoryId
RoleID
SubCategories
Id
Name
AspNetUserClaims
Id
UserId
ClaimType
ClaimValue
CategoriesSubCategories
Id
SubCategoryId
CategoryId
Products
Id
CategoryId
Year
Country
Name
Description
YouTubeLink
UserRating
CriticRating
CommonRating
ProductPicture
ProductSubCategories
Id
SubCategoryId
ProductId
AspNetUsers
Id
UserNick
Salt
Tag
RegistrationDate
ProfilePicture
UserName
NormalizedUserName
Email
NormalizedEmail
EmailConfirmed
PasswordHash
SecurityStamp
ConcurrencyStamp
PhoneNumber
PhoneNumberConfirmed
TwoFactorEnabled
LockoutEnd
LockoutEnabled
AccessFailedCount
AspNetRoleClaims
Id
RoleId
ClaimType
ClaimValue
AspNetRoles
Id
Name
NormalizedName
ConcurrencyStamp
AspNetUserRoles
UserId
RoleId
AspNetUserLogins
LoginProvider
ProviderKey
ProviderDisplayName
UserId
Reviews
Id
UserId
ProductId
ReviewDetails
Rating
Likes
Dislikes
CreationDate
IsCritic
AspNetUserTokens
UserId
LoginProvider
Name
Value
UserLikes
Id
UserId
Likes
ReviewId

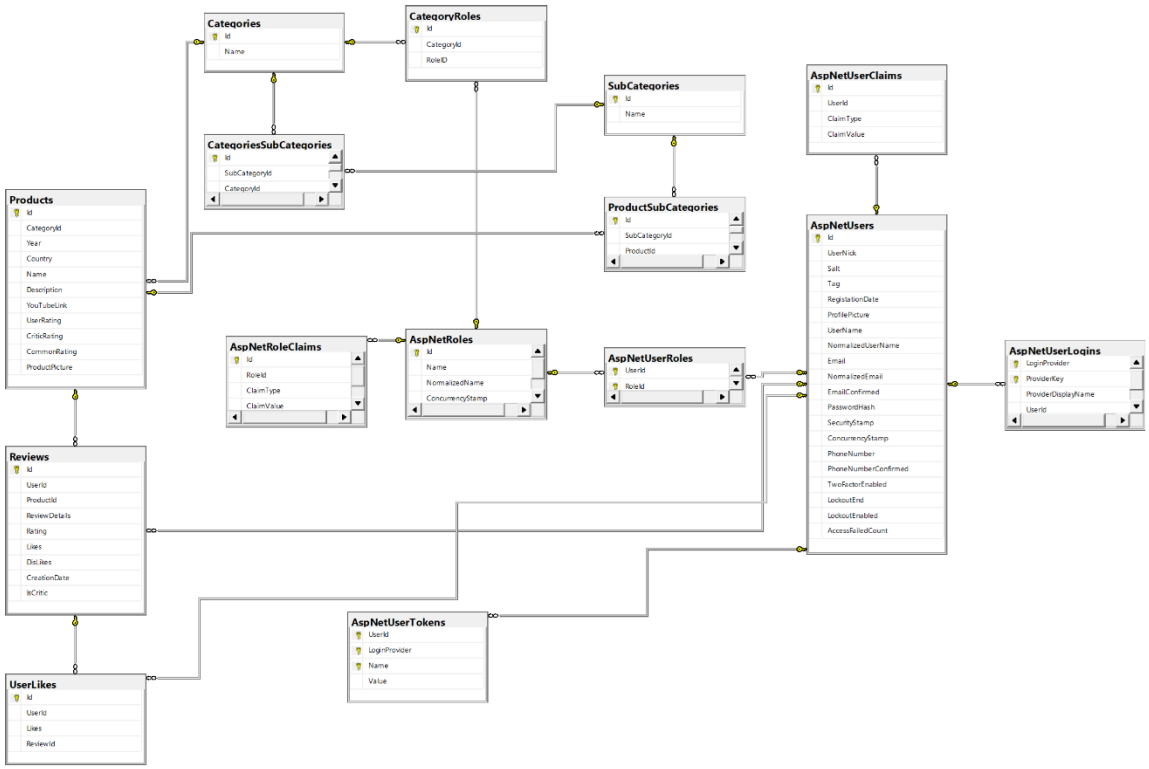


Рисунок 2.4 – Модель Бази Даних

Таблиця 2.3 – Опис таблиць БД

Таблиця	Опис
Products	Таблиця для зберігання даних про продукт
Reviews	Таблиця для зберігання даних про огляди на продукт
UsersLikes	Таблиця для зберігання даних про оцінки користувачів на рецензії інших користувачів
Categories	Таблиця для зберігання даних про категорії
CategoryRoles	Сутність з'єднання, яка пов'язує категорії та ролі.
CategoriesSubCategories	Сутність з'єднання, яка пов'язує категорії та підкатегорії.
SubCategories	Таблиця для зберігання даних про підкатегорії
ProductSubCategories	Сутність з'єднання, яка пов'язує підкатегорії та продукт.

AspNetRoles	Створена автоматично за допомогою ASP NET Core Identity.Таблиця для зберігання даних про ролі.
AspNetRoleClaims	Створена автоматично за допомогою ASP NET Core Identity. Надає твердження(Claims), яке надається всім користувачам у ролі.
AspNetUserRoles	Створена автоматично за допомогою ASP NET Core Identity. Сутність з'єднання, яка пов'язує користувача та роль.
AspNetUsers	Створена автоматично за допомогою ASP NET Core Identity. Таблиця для зберігання даних про користувачів
AspNetUserTokens	Створена автоматично за допомогою ASP NET Core Identity. Таблиця для зберігання даних про маркер автентифікації для користувача.
AspNetUserLogins	Створена автоматично за допомогою ASP NET Core Identity. Таблиця для зберігання даних про стороні входи у систему.
AspNetUserClaims	Створена автоматично за допомогою ASP NET Core Identity. Надає твердження(Claims), яке надається всім користувачам.

Таблиця 2.4 – Структура таблиці Products

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
CategoryId	Ідентифікатор категорії	Int	FK
Year	Рік виготовлення	Int	
Country	Країна виробник	Nvarchar	

Name	Назва продукту	Nvarchar	
Description	Опис продукту	Nvarchar	
YouTubeLink	Посилання на youtube відео	Nvarchar	
UserRating	Рейтинг користувачів	Float	
CriticRating	Рейтинг критиків	Float	
CommonRating	Загальний критиків	Float	
ProductPicture	Зображення продукту	Varbinary	
	Ідентифікатор користувача	Nvarchar	FK

Таблиця 2.5 – Структура таблиці Reviews

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
UserId	Ідентифікатор користувача	Nvarchar	FK
ProductId	Ідентифікатор продукту	Int	FK
ReviewDetalis	Деталі огляду	Nvarchar	
Rating	Рейтинг	Int	
Likes	Кількість вподобайок	Int	
DisLikes	Кількість невподобайок	Int	
CreationDate	Дата створення	Datetime2	
IsCtitic	Чи є автор огляду критиком	Bit	

Таблиця 2.6 – Структура таблиці UsersLikes

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
UsesrId	Ідентифікатор користувача	Nvarchar	FK
Likes	Чи є оцінка вподобайкою або невподобайкою	Bit	
ReviewId	Ідентифікатор рецензії	Int	FK

Таблиця 2.7 – Структура таблиці Categories

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
Name	Назва категорії	Nvarchar	

Таблиця 2.8 – Структура таблиці CategoryRoles

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
CategoryId	Ідентифікатор категорії	Int	FK
RoleID	Ідентифікатор ролі	Nvarchar	FK

Таблиця 2.9 – Структура таблиці CategoriesSubCategories

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
SubCategoryId	Ідентифікатор підкатегорії	Int	FK

CategoryId	Ідентифікатор категорії	Int	FK

Таблиця 2.10 – Структура таблиці SubCategories

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
Name	Назва підкатегорії	Nvarchar	

Таблиці 2.11 – Структура таблиці ProductSubCategories

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Int	PK
SubCategoryId	Ідентифікатор підкатегорії	Int	FK
ProductId	Ідентифікатор продукти	Int	FK

Таблиці 2.12 – Структура таблиці AspNetRoles

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Nvarchar	PK
Name	Назва ролі	Nvarchar	
NormalizedName	Нормалізована назва ролі	Nvarchar	
ConcurrencyStamp	Штамп паралельності	Nvarchar	

Таблиця 2.13 – Структура таблиці AspNetRoleClaims

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	int	PK
RoleId	Ідентифікатор ролі	Nvarchar	FK
ClaimType	Тип твердження(Claim)	Nvarchar	
ClaimValue	Значення твердження(Claim)	Nvarchar	

Таблиця 2.14 – Структура таблиці AspNetUserRoles

Колонка	Опис	Тип даних	Ключ
UserId	Ідентифікатор користувача	Nvarchar	FK
RoleId	Ідентифікатор ролі	Nvarchar	FK

Таблиця 2.15 – Структура таблиці AspNetUsers

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Nvarchar	PK
UserNick	Нік користувача	Nvarchar	
Salt	Сіль для кодування пароля	Nvarchar	
Tag	Тег для кодування пароля	Nvarchar	
RegistrationDate	Дата реєстрації	Datetime2	
ProfilePicture	Зображення користувача	Varbinary	
UserName	Ім’я користувача	Nvarchar	

NormalizedUserName	Нормалізовване ім’я користувача	Nvarchar	
Email	Пошта користувача	Nvarchar	
NormalizedEmail	Нормалізована пошта користувача	Nvarchar	
EmailConfirmed	Підтвердження пошти	Bit	
PasswordHash	Пароль	Nvarchar	
SecurityStamp	Штамп безпеки	Nvarchar	
ConcurrencyStamp	Штамп паралельності	Nvarchar	
PhoneNumber	Номер телефона	Nvarchar	
PhoneNumberConfirmed	Підтвердження номера телефону	Bit	
TwoFactorEnabled	Двох факторна авторизація	Bit	

Таблиця 2.16 – Структура таблиці AspNetUserTokens

Колонка	Опис	Тип даних	Ключ
UserId	Ідентифікатор користувача	Nvarchar	FK
LoginProvider	Постачальник зовнішнього входу	Nvarchar	FK
Name	Ім’я	Nvarchar	
Value	Значення	Nvarchar	

Таблиця 2.17 – Структура таблиці AspNetUserLogins

Колонка	Опис	Тип даних	Ключ
LoginProvider	Постачальник зовнішнього входу	Nvarchar	PK

ProviderKey	Ключ постачальника зовнішнього входу	Nvarchar	
ProviderDisplayName	Ім’я постачальника зовнішнього входу	Nvarchar	
UserId	Ідентифікатор користувача	Nvarchar	FK

Таблиця 2.18 – Структура таблиці AspNetUserClaims

Колонка	Опис	Тип даних	Ключ
Id	Ідентифікатор	Nvarchar	PK
UserId	Ідентифікатор користувача	Nvarchar	FK
ClaimType	Тип твердження(Claim)	Nvarchar	
ClaimValue	Значення твердження(Claim)	Nvarchar	

2.4 Вибір засобів розробки

Для створення веб-застосунку буде використаний ASP.NET Core. ASP.NET Core — це міжплатформна платформа з відкритим вихідним кодом для створення сучасних хмарних додатків, підключених до Інтернету, таких як веб-програми, програми IoT і мобільні серверні програми. Програми ASP.NET Core працюють у .NET, безкоштовному міжплатформному середовищі виконання програм із відкритим кодом. Він був розроблений, щоб забезпечити оптимізовану структуру розробки для програм, які розгортаються в хмарі або виконуються локально. Він складається з модульних компонентів з мінімальними накладними витратами, тож ви зберігаєте гнучкість під час створення своїх рішень. ASP.NET Core надає моєливість розробляти та запускати свої додатки між платформами на Windows, Mac і Linux. Рішення чому був обраний ASP.NET Core випливають з його переваг :

- уніфікована історія для створення веб-інтерфейсу та веб-API;
- Razor Pages робить кодування сценаріїв, орієнтованих на сторінки, простішим і продуктивнішим;

- Blazor дозволяє використовувати C# у браузері разом із JavaScript;
- можливість розробляти та працювати на Windows, macOS і Linux
- рішення є з відкритим вихідним кодом і орієнтована на спільноту,
- інтеграція сучасних, клієнтських фреймворків і робочих процесів розробки,
- вбудована ін’єкція залежностей, Легкий, високопродуктивний і модульний конвеєр запитів HTTP.

Для зв’язку з БД буде використовуватися Entity Framework Core. Entity Framework представляє ORM-технологію (object-relational mapping – відображення даних на реальні об'єкти) від компанії Microsoft для доступу до даних. Entity Framework Core — це легке, розширюване програмне

забезпечення з відкритим кодом, як і всі попередні версії Entity Framework. Це крос-платформне програмне забезпечення, що дозволяє працювати з різними операційними системами, такими як Windows, Mac OS і Linux.

Entity Framework Core автоматично виконує операції створення, читання, оновлення та видалення без необхідності кожного разу писати всі SQL-запити вручну.

Entity Framework Core працює за принципом ORM, що дозволяє відмовитися від необхідності писати весь код доступу, на написання якого розробникам зазвичай доводилося витрачати стільки часу.

Таким чином усі розробники можуть зосередитися на своєму поточному проекті ефективно й ефективно, не турбуючись про написання всіх запитів .NET і SQL.

Також буде розроблений спеціальний алгоритм для пропонування продукту користувачу. Алгоритм також може запропонувати вже оцінені користувачем продукти, але шанс цього залежить від оцінки користувача

2.5 Аналіз безпеки даних

Так як я використовую трьох рівневу архітектуру, то в мене вже є висока безпека. Тому потрібно забезпечити надійність передачі даних через інтернет. Для цього нам потрібно налаштувати протокол https та за можливістю використати для цього сертифікат TLS(transport layer security) 1.3, так як він найкращий на даний час. Для повної безпеки треба ще правильно зберігати дані у базі даних. З важливих даних в нас лише паролі користувачів, тому потрібно забезпечити надійну безпеку.

Спочатку використовуємо функцію SHA-512. Цей алгоритм не дуже надійний але в нього є одна перевага - розмір виводу завжди константного розміру(у нашому випадку 512 біт). Тому якщо хтось у поле паролю внесе велику кількість байтів, то завдяки SHA-512 ми обмежуємо це до 512 байт та передаємо ці байти у більш надійний алгоритм.

Після SHA-512 використовуємо AES-GCM. Для використання AESGCM нам потрібно буде ключ ,тег та сіль (унікальне, випадково згенероване значення). Так як без ключа не вийде розшифрувати пароль, то ключ будемо зберігати у конфігураційному файлі. Сіль та тег не так важливі як ключ, тож вони зберігаються у базі даних разом з зашифрованим паролем.

Висновки до розділу

У розділі було розглянуто:

- архітектуру застосування;
- функціональність застосування;
- безпеку застосування.

РОЗДІЛ 3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1Аналіз безпеки даних

Аналіз якості є невід'ємною частиною життєвого циклу програмного забезпечення. Під час розробки програмного забезпечення тестування є обов'язковим. Аналіз якості програмного забезпечення здійснюється за допомогою оцінки того, на скільки воно відповідає вимогам. Для оцінки якості треба виконати тестування функціональних вимог.

Об'єкти тестування

Для тестування веб-застосунку ми будемо використовувати платформу Windows та Android. Тестування додатка буде проводитись для браузерів Google Chrome версії 60 і вище, Opera GX версії 70 і вище та Firefox версії 4 і вище.

Функції які не обов'язково тестувати

Час за який виконуються запити до бази даних. Швидкодія терфейсу користувача також може залишитись без тестування.

Функції які обов'язково тестувати

- створення рецензій;
- редагування рецензій;
- видалення рецензій;
- реєстрація користувачів;
- створення нової ролі;
- перегляд рецензій написаних певним користувачем;
- редагування ролей користувачів;- додавання продуктів; редагування продуктів;
- видалення продуктів;
- функція для випадкового пропонування продуктів;
- видалення користувачів;
- створення нової категорії;

- видалення підкатегорій;
- додавання оцінки на рецензію.

Методологія проведення тестування

Наш додаток повинен мати якомога більше тестового покриття. Усі функції будуть протестовані на 3 вищеназваних браузерах та на 2 платформах: Windows та Android. У тестів має бути передумова, проміжні дії, очікуваний результат та дійсний результат.

3.2 Опис процесів тестування

Таблиця 3.1 - Створення рецензії

Ідентифікатор тестового кейсу	PO_001
Назва тесту	Створення рецензії
Передумови	Продукт існує. Користувач зареєстрований і увійшов у систему. Користувач не має ролі “Banned”
Проміжні дії	Користувач відкрив сторінку продукту. Користувач заповнив поля для створення рецензії.
Очікуваний результат	Користувач створив нову рецензію. Нова рецензія відображається на сторінці продукту
Дійсний результат	Користувач створив нову рецензію. Нова рецензія відображається на сторінці продукту

Таблиця 3.2 - Редагування рецензії

Ідентифікатор тестового кейсу	PO_002
Назва тесту	Редагування рецензії

Передумови	Рецензія існує. Користувач зареєстрований і увійшов у систему. Користувач є володарем цієї рецензії.
Проміжні дії	Користувач відкрив форму для редагування рецензії.
Очікуваний результат	Користувач відредагував рецензію. Нова рецензія відображається на сторінці продукту
Дійсний результат	Користувач відредагував рецензію. Нова рецензія відображається на сторінці продукту

Таблиця 3.3 - Видалення рецензії

Ідентифікатор тестового кейсу	PO_003
Назва тесту	Видалення рецензії
Передумови	Рецензія існує. Користувач зареєстрований і увійшов у систему. Користувач є володарем цієї рецензії.
Проміжні дії	Користувач відкрив сторінку продукту. Користувач знайшов свою рецензію.
Очікуваний результат	Користувач видалив свою рецензію. Рецензія видалена.
Дійсний результат	Користувач видалив свою рецензію. Рецензія видалена.

Таблиця 3.4 - Реєстрація користувача

Ідентифікатор тестового кейсу	PO_004
Назва тесту	Реєстрація користувача
Передумови	Користувач не зареєстрований
Проміжні дії	Відкрита сторінка реєстрації
Очікуваний результат	Новий користувач зареєстрований
Дійсний результат	Новий користувач зареєстрований

Таблиця 3.5 – Створення нової ролі

Ідентифікатор тестового кейсу	PO_005
Назва тесту	Створення нової ролі
Передумови	Роль ще не існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin”
Проміжні дії	Користувач відкрив сторінку керування ролями. Користувач заповнив поля для створення ролі.
Очікуваний езультат	Користувач створив нову роль. Нова роль відображається на сторінці керування ролями
Дійсний результат	Користувач створив нову роль. Нова роль відображається на сторінці керування ролями

Таблиця 3.6 - Перегляд рецензій написаних певним користувачем

Ідентифікатор тестового кейсу	PO_006
Назва тесту	Перегляд рецензій написаних певним користувачем
Передумови	Користувач існує.
Проміжні дії	Відкрита сторінка профіля користувача.
Очікуваний результат	Користувач бачить усі рецензії написані певним користувачем
Дійсний результат	Користувач бачить усі рецензії написані певним користувачем

Таблиця 3.7 - Редагування ролей користувачів

Ідентифікатор тестового кейсу	PO_007
Назва тесту	Редагування ролей користувачів
Передумови	Користувач існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin”
Проміжні дії	Користувач відкрив сторінку керування ролями користувачів.

Очікуваний результат	Користувач редагує ролі певного користувача
Дійсний результат	Користувач редагує ролі певного користувача

Таблиця 3.8 - Додавання продуктів

Ідентифікатор тестового кейсу	PO_008
Назва тесту	Додавання продуктів
Передумови	Користувач існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування продуктами. Користувач заповнив поля для створення продукту.
Очікуваний результат	Користувач створив новий продукт. Новий продукт відображається на сторінці продуктів
Дійсний результат	Користувач створив новий продукт. Новий продукт відображається на сторінці продуктів

Таблиця 3.9 - Редагування продуктів

Ідентифікатор тестового кейсу	PO_009
Назва тесту	Редагування продуктів
Передумови	Користувач існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування продуктами. Користувач заповнив поля для редагування продукту.
Очікуваний результат	Користувач відредагував продукт. Відредагований продукт відображається на сторінці продуктів
Дійсний результат	Користувач відредагував продукт. Відредагований продукт відображається на сторінці продуктів

Таблиця 3.10 - Видалення продуктів

Ідентифікатор тестового кейсу	PO_010
Назва тесту	Видалення продуктів
Передумови	Користувач існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування продуктами. Користувач натиснув на поля для видалення продукту.
Очікуваний результат	Користувач видалив продукт. Продукт не відображається на сторінці продуктів
Дійсний результат	Користувач видалив продукт. Продукт не відображається на сторінці продуктів

Таблиця 3.11 - Функція для випадкового пропонування продуктів

Ідентифікатор тестового кейсу	PO_011
Назва тесту	Випадкове пропонування продукту
Передумови	Продукт існує.
Проміжні дії	Користувач відкрив сторінку рандомайзера та обрав критерії для нього.
Очікуваний результат	Користувач отримав випадковий продукт за заданими параметрами
Дійсний результат	Користувач отримав випадковий продукт за заданими параметрами

Таблиця 3.12 - Видалення категорії

Ідентифікатор тестового кейсу	PO_012
Назва тесту	Видалення категорії
Передумови	Категорія існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування категоріями. Користувач натиснув на поля для видалення категорії.

Очікуваний результат	Користувач видалив категорію. Категорія не відображається
Дійсний результат	Користувач видалив категорію. Категорія не відображається

Таблиця 3.13 - Створення підкатегорій

Ідентифікатор тестового кейсу	PO_0013
Назва тесту	Створення підкатегорій
Передумови	Підкатегорія не існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування підкатегоріями . Користувач заповнив поля для створення підкатегорії.
Очікуваний результат	Користувач створив нову підкатегорію.
Дійсний результат	Користувач створив нову підкатегорію.

Таблиця 3.14 - Видалення підкатегорій

Ідентифікатор тестового кейсу	PO_014
Назва тесту	Видалення підкатегорій
Передумови	Підкатегорія існує. Користувач зареєстрований і увійшов у систему. Користувач має роль “Admin” або “Moderator”
Проміжні дії	Користувач відкрив керування підкатегоріями. Користувач натиснув на поля для видалення підкатегорії.
Очікуваний результат	Користувач видалив підкатегорію. Підкатегорія не відображається
Дійсний результат	Користувач видалив підкатегорію. Підкатегорія не відображається

Таблиця 3.15 - Додавання оцінки на рецензію

Ідентифікатор тестового кейсу	PO_015
Назва тесту	Додавання оцінки на рецензію
Передумови	Рецензія існує. Користувач зареєстрований і увійшов у систему.
Проміжні дії	Користувач відкрив сторінку продукту. Користувач обрав рецензію для оцінки.
Очікуваний результат	Користувач додав оцінку на рецензію.
Дійсний результат	Користувач додав оцінку на рецензію.

3.3 Результати тестування

Таблиця 3.16 – Результати тестування програмного забезпечення

Ідентифікатор тестового кейсу	Назва тесту	Результат
PO_001	Створення рецензії	Пройдено
PO_002	Редагування рецензії	Пройдено
PO_003	Видалення рецензії	Пройдено
PO_004	Реєстрація користувача	Пройдено
PO_005	Створення нової ролі	Пройдено
PO_006	Перегляд рецензій написаних певним користувачем	Пройдено
PO_007	Редагування ролей користувачів	Пройдено
PO_008	Додавання продуктів	Пройдено
PO_009	Редагування продуктів	Пройдено
PO_010	Видалення продуктів	Пройдено
PO_011	Випадкове пропонування продукту	Пройдено
PO_012	Видалення категорії	Пройдено

PO_013	Створення підкатегорій	Пройдено
PO_014	Видалення підкатегорій	Пройдено
PO_015	Додавання оцінки на рецензію	Пройдено

Висновки до розділу

У даному розділі було протестовані основні функції нашого вебзастосунку. У тестів має була передумова, проміжні дії, очікуваний результат та дійсний результат. Всі тести були успішно пройдено.

3.4Розгортання та робота програмного забезпечення

Так як розроблене програмне забезпечення є веб-додатком то нам потрібно для його розгортання зробити декілька кроків:

- розгорнути базу даних MSSQL на віддаленому сервері;
- розгорнути кожний рівень нашого веб-застосунку на віддаленому сервері, додаткове створення бази даних не потрібно, так як в нас вже налаштована ORM система з міграціями;
- внести всі необхідні дані до бази даних(товари, ролі, ролі користувачів, категорії, підкатегорії).

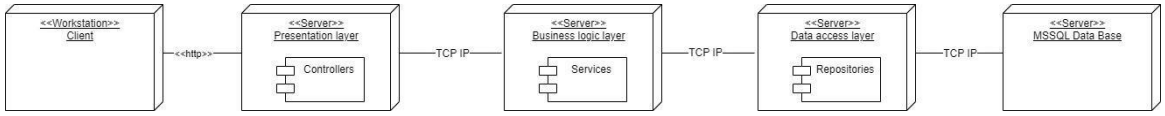


Рисунок 3.1 – Діаграма розгортання

Коли користувач відкриє наш веб-застосунок він матиме змогу користуватися усіма функціями які були розроблені у додатку. Він матиме змогу переглядати рецензії, створювати свої рецензії, оцінювати чужі рецензії.

Також у додатку реалізована рандомайзера продуктів. Для тих користувачів які не можуть вирішити, що спробувати. У цьому рандомайзері також враховуються оцінки цього та інших користувачів.

Керівництво користувача наведено в документі “Керівництво користувача”.

Висновки до розділу

В розділі 4 описано необхідні умови для розгортання розробленого програмного забезпечення та роботи з ним

ВИСНОВКИ

Загальна мета даної роботи полягає у дослідженні моделей та методів визначення характеристик коду в ВЕБ-фреймворках. У вступі зазначено актуальність даної проблеми, а також сформульовано основні завдання дослідження. У першому розділі надано огляд існуючих методів визначення характеристик коду, описано технологію сучасної веб-розробки, базові веб-технології, стандарт CSS та типи зв'язку між HTML та CSS. У другому розділі охарактеризовано методології вдосконалення програмного забезпечення, включаючи методику з'єднання компонентів програмного забезпечення, CSS рефакторинг та підтримку інструментів, а також основні фази емпіричного дослідження проблеми та методи аналізу в емпіричному дослідженні. У третьому розділі описано побудову та підтримку CSS рішень та інструментів рефакторингу для роботи з веб-фреймворками, ідентифікацію веб-рамок зі стилів CSS Code, кореляцію між різними видами коду та результати виконаного дослідження для визначення характеристик коду в ВЕБ-фреймворках. У висновках підведено підсумки дослідження та визначено перспективи подальших досліджень у даній області.

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА

1) Intoduction to Identity on ASP .NET Core [Електронний ресурс] — Режим доступу: Introduction to Identity on ASP.NET Core | Microsoft Learn

2) Overview of ASP.NET Core [Електронний ресурс] — https://learn.microsoft.com/en-us/aspnet/core/introduction-to-aspnetcore?view=aspnetcore-7.0

3) Overview of ASP.NET Core MVC [Електронний ресурс] - https://learn.microsoft.com/en-us/aspnet/core/mvc/overview? view=aspnetcore-7.0

4) Агрегатор рецензій [Електронний ресурс] - https://www.wiki.uk-ua.nina.az/%D0%90%D0%B3%D1%80%D0%B5%D0%B3%D0%B0%D1%82%D0%BE%D1%80_%D1%80%D0%B5%D1%86%D0%B5%D0%BD%D0%B7%D1%96%D0%B9.html

5) Що краще чи найкорисніші агрегатори оцінок [Електронний ресурс] - https://dtf.ru/read/22258-chto-luchshe-ili-samye-poleznye-agregatoryocenok

6) Entity Framework Core [Електронний ресурс]- https://learn.microsoft.com/en-us/ef/core/

7) Advantages and Disadvantages of Three-Tier Architecture in DBMS [Електронний ресурс] - https://www.geeksforgeeks.org/advantages-anddisadvantages-of-three-tier-architecture-in-dbms/

8) Why You Should Use a Review Aggregator [Електронний ресурс] - https://reviewbot.io/blog/reviewaggregator/#:~:text=Review%20aggregators%20are%20platforms%20that%20collect%20reviews%20on%20a%20product,products%2C%20services%2C%20and%20niches

9) Get started with Bootstrap [Електронний ресурс]- https://getbootstrap.com/docs/5.2/getting-started/introduction/

10) Microsoft SQL Server Pros and Cons [Електронний ресурс] - Microsoft SQL Server Pros and Cons | LearnSQL.com

БІБЛІОГРАФІЧНА ДОВІДКА

Тема дипломної роботи бакалавра: " Розробка та вдосконалення моделей, методів і алгоритмів побудови ВЕБ-аггрегаторів "

Обсяг пояснювальної записки: 59 аркушів

Дата закінчення дипломної роботи червня 2024 р.

Підпис студента _____