Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ КУПІВЛІ КОМП'ЮТЕРНИХ ІГОР
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-20-3
Терещенко О.О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Кіношенко Д.К.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Терещенку Олександру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка вебзастосунку для купівлі комп'ютерних ігор

затверджена наказом університету від 20 травня 2024 року № 464 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 25 травня 2024 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, бібліотека, мови програмування Golang, JavaScript, відкрита JavaScript бібліотека React, документо-орієнтована система керування базами даних з відкритим вихідним кодом MongoDB, хмарний сервіс для зберігання даних Cloud Storage for Firebase, середовище розробки Visual Code.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд існуючих вебзастосунків для купівлі відеоігор.
2. Аналіз найліпших рішень серед вебзастосунків.
3. Вивчення потрібних підходів та рішень реалізації вебзастосунків.
4. Розробка мікросервісів для функціонування вебзастосунку для купівлі відеоігор.
5. Розробка вебзастосунку для купівлі відеоігор.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність вебзастосунків для купівлі цифрових продуктів, постановка задачі, зображення прикладів або результатів тестування програми, лістинги коду.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.04.2024	
2	Аналіз завдання, підбір літератури	08.04.24-12.04.24	
3	Аналіз літератури з досліджуваної проблеми	12.04.24-18.04.24	
4	Аналіз технічних засобів	19.04.24-23.04.24	
5	Розробка мікросервісів	27.04.24-16.05.24	
6	Програмна реалізація	17.05.24-24.05.24	
7	Оформлення пояснювальної записки	24.05.24-26.05.24	
8	Перевірка на плагіат	28.05.24	
9	Рецензування	29.05.24	
10	Підготовка презентації та доповіді	30.05.24-09.06.24	
11	Занесення роботи в електронний архів	10.06.24	
12	Попередній захист кваліфікаційної роботи	10.06.24	

Дата видачі завдання 8 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ ст. викл. Кіношенко Д.К.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 65 с., 30 рис., 31 джерело.

ОНЛАЙН МАГАЗИН, ВЕБЗАСТОСУНОК, ВЕБРОЗРОБКА, АНАЛІТИКА ДАНИХ, ПЕРСОНАЛІЗАЦІЯ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ, КОМП'ЮТЕРНІ ІГРИ, GOLANG, JAVASCRIPT, REACT, MONGODB.

Об'єктом роботи є вебзастосунок для купівлі комп'ютерних ігор гравцями на вебсайті та можливостями взаємодіяти з рецензіями на ігри.

Метою роботи є розробка вебзастосунку де гравці можуть купити ігри, лишати рецензії та отримувати персоналізовану статистику та рекомендації. А розробники отримують напряму відгуки від гравців. Видавці та розробники мають можливість додавати ігри до вебсайту.

Був проведений аналіз інший вебзастосунків по продажу ігор, для розуміння хороший та кепських рішень, а також навіть критично невдалих. Завдяки чому при створені вебсайту було використано найліпші рішення для залучення покупців.

У результаті роботи зроблено вебзастосунок для купівлі комп'ютерних ігор з ліпшим функціоналом для покупців.

ONLINE STORE, WEB PROJECTLICATION, WEB DEVELOPMENT, DATA ANALYTICS, PERSONALIZATION, RECOMMENDER SYSTEMS, COMPUTER GAMES, GOLANG, JAVASCRIPT, REACT, MONGODB.

The object of the work is a web projectlication for the purchase of computer games by players on a website and the ability to interact with game reviews.

The aim is to develop a web projectlication where players can buy games, leave reviews, and receive personalized statistics and recommendations. And developers receive direct feedback from players. Publishers and developers can add games to the website.

We analyzed other web projectlications for selling games to understand good and bad solutions, as well as even critically unsuccessful ones. Thanks to this, the best solutions for attracting customers were used when creating the website.

As a result, we created a web projectlication for buying computer games with better functionality for customers.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМОВЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

EA – Electronic Arts (електронне мистецтво)

EGS – Epic Games Store (магазин епічних ігор)

GOG – Good Old Games (старі добрі ігри)

VS – Visual Code (візуальний код)

БД – база даних

UI – User Interface (користувацький інтерфейс)

PC – Personal Computer (персональний комп’ютер)

NoSQL – Not only Structured Query Language (не тільки структурована мова запитів)

JSX – JavaScript XML

HTML – HyperText Markup Language (мова розмітки гіпертексту)

XML – EXtensible Markup Language (розширювана мова розмітки)

ВСТУП

21 століття це епоха економік третинного сектору, а це в свою чергу види діяльності що відносяться до сфери послуг. Комп'ютерні ігри як раз входять в цю сферу. Люди люблять в комфорті приємно проводити свій час та готові за це віддавати гроші, цей ринок відносно молодий, якщо взагалі говорити за цифрові копії, то йому трошки більше 20 років, а саме 22 роки. Вже 22 березня 2002 року появився перший магазин по продажу ігор через інтернет. За 2 десятки років цей магазин став найбільшим та найпопулярнішим в свої сфері, але через його монструозні розміри він часто надто повільно працює та має багато рудиментарних елементів які на нинішній день зайві, та навіть шкідливі для продуктивності застосунку. Також появлялися нові магазини, але вони не враховували найліпші сторони найбільшого гравця на ринку, тому не є такими успішними, хоча запит від покупців на ліпші альтернативи існує. Для цього й був розроблений вебзастосунок для купівлі комп'ютерних ігор.

Сервіс для найкращої взаємодії з користувачем має взяти найліпші рішення від інших вебсайтів, а також мати свою систему персоналізації. Щоби полегшувати для покупців вибір ігор. Аналіз даних буде відбувався на діях покупця. Які ігри він купує, які в тих іграх переважають жанри, що цікаво друзям гравця, які продукти найбільше зацікавили гравця, але також потрібно розуміти що могли розізлити чи розчарувати користувача. Так як вебзастосунок цілиться на його комфорт, щоби він хотів залишатися на платформі, а отже продовжував купляти ігри саме тут та закликати інших людей до вебзастосунку для купівлі комп'ютерних ігор.

Розробка вебзастосунку є актуальна та корисна в нинішню добу, бо ринок продажу цифрових копій відеоігор молодий та неперенасичений. А отже в час економік послуг та Інтернету потрібно вливатися в цю сферу, заки час це дозволяє.

1 АНАЛІЗ ІСНУЮЧИХ ЗАСТОСУНКІВ ДЛЯ КУПІВЛІ ТА ПУБІЛКАЦІЇ КОМП'ЮТЕРНИХ ІГОР

1.1 Нинішня ситуація на ринку цифрових копій комп'ютерних ігор

Ринок комп'ютерних ігор вже вдосталь старий, але цифрові копії почали продавати відносно недавно, ще 20 років тому не було можливості купити ігри в Інтернеті, але за малий проміжок часу після запуску вебсайтів та застосунків по продажу цифрових копій ситуація різко почала змінюватися не на користь фізичних копій (рис. 1.1). Але на нинішній день майже всі фізичні копії продаються на консолях. Але їх ринок менш цікавий для української аудиторії, бо головна платформа в Україні це PC. Та якщо на платформах консолей є повна монополія від видавця ігрових станцій, то на платформі PC такого немає й там може розвиватися любий новий інтернет магазин, бо покупців вдосталь, що на українському ринку, та й тим паче на світовому [1].

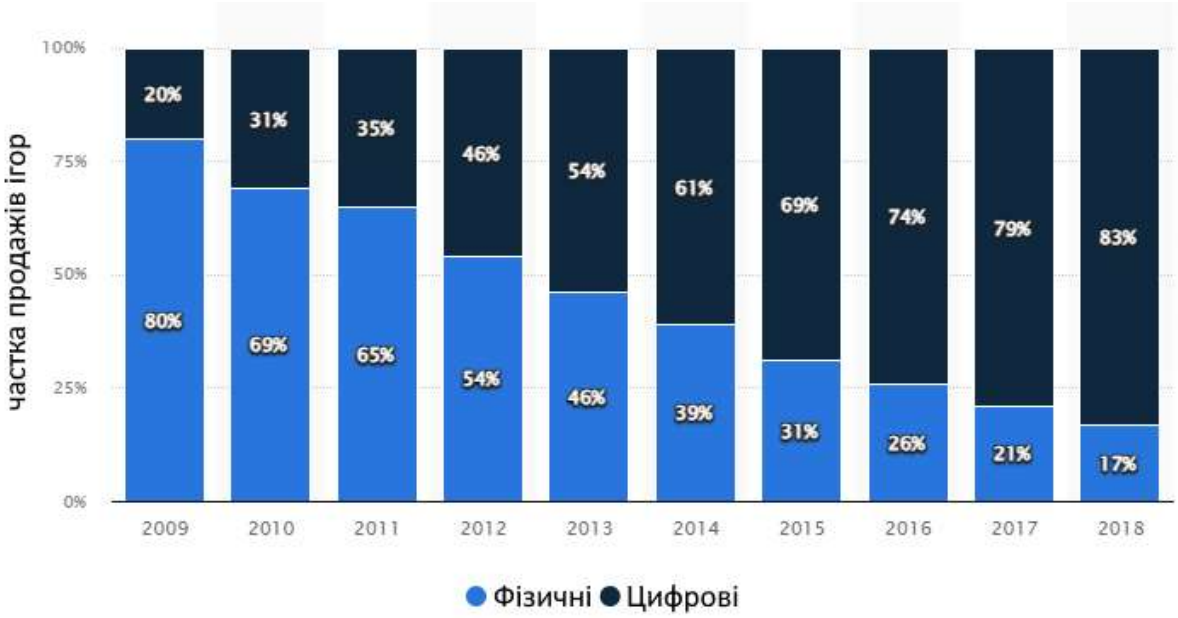


Рисунок 1.1 – Діаграма зображення частки продажів ігор в фізичних та цифрових копіях [2]

Це створює конкурентну атмосферу, яка сприяє появі нових пропозицій і змушує існуючі магазини постійно вдосконалюватися. Прикладом таких змагань є історія Epic Games Store, він відомий своєю агресивною стратегією залучення користувачів, де пропонує безкоштовні ігри та вигідні умови для розробників. Його вихід на ринок змусив інших гравців, зокрема такого як Steam, поліпшити свої послуги та адаптуватися до мінливого конкурентного середовища. Такий підхід змусив обидві компанії постійно вдосконалюватися і пропонувати більш вигідні умови користувачам і розробникам, а новим магазинам враховувати ці події. Розуміти які помилки були допущені зі сторони Epic Games й як Valve реагували на зміни в ринку.

1.2 Найбільш популярні вебсервіси для купівлі комп'ютерних ігор

Для розуміння що потрібно покупцю повинні дослідити найбільших гравців на ринку та зрозуміти чому вони такі та з'ясувати недоліки їх крамниць. Зараз існує 3 великих незалежних вебмагазинів по продажу комп'ютерних ігор та 3 чималі від видавців. Перша категорія має багато різних відео ігор від різних видавців, незалежних розробників, інді компаній чи взагалі самостійних програмістів в одну людину, але й існує друга категорія яка тісно пов'язана саме лише з видавцем.

Ще пару років тому кожна студія які належало пару франшиз мали свої магазини як то до слова Bethesda, але через не прибутковість, яка була спричинена малої кількістю новинок, хоча би щорічно випускати хоч якісь відеоігри, магазин було закрито [3]. Тому такі вебзастосунки можуть тримати лише видавці які постійно випускають нові тайтли. Такий варіант зрозуміло не підходе вебсервісу для покупок комп'ютерних ігор, але це не заважає розглянути рішення тих магазинів щодо UI та функціоналу що надається користувачу. Бо то є великі магазини з своїм інколи унікальним функціоналом та рішеннями в інтерфейсі. Найбільшими видавцями, які можуть тримати свої

крамниці є Ubisoft [4] (рис. 1.2) та EA [5] (рис. 1.3). Завдяки своїм можливостям випускати регулярно ігри та їх масштабності, їх сервіси є прибутковими та мають доцільність в своєму існуванні.

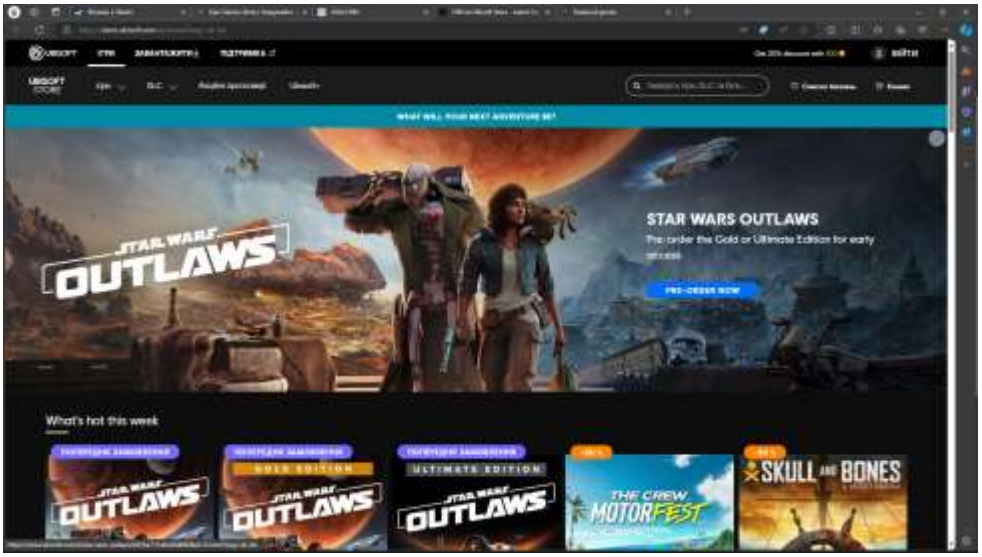


Рисунок 1.2 – Приклад вебзастосунку для купівлі комп'ютерних відеоігор від Ubisoft

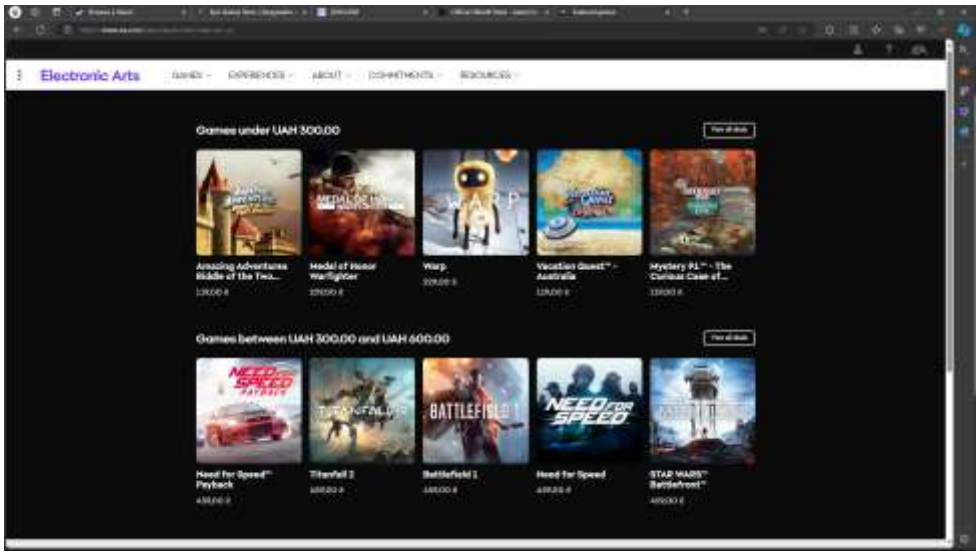


Рисунок 1.3 - Приклад вебзастосунку для купівлі комп'ютерних відеоігор від EA

Що стосується незалежних магазинів, вони також належать студіям які випускають ігри, але на їх платформах продаються й комп'ютерні бавки й від

інших розробників. Що розширює кількість продукції в їх крамницях до тисячі, а випадку найбільшого гравця до сотні тисяч ігор. За інформацією на 1 квартал 2024 року існує:

– Steam (рис. 1.4), заснований у 2004 році, є найбільшим онлайн-магазином ігор у світі, з понад 150 мільйонами активних користувачів, та піковими показниками водночас онлайн 31,9 мільйона користувачів в листопаді 2022 та 33 мільйона в січні 2023 року [6]. Також вебзастосунок пропонує просто захмарне число ігор, там доступно понад 300000 ігор [7]. Це ігри абсолютно різних жанрів та якості, але на кожний продукт находиться свій покупець. З переваг саме функціоналу платформи від Valve для активних гравців є розділ «Спільнота», де покупці годні спілкуватися, ділитися досягненнями та обговорювати ігри [8]. Але найголовнішим плюсом є систематичні знижки, як для певних жанрів ігор, так загальні для всіх ігор в крамниці;

Рисунок 1.4 – Приклад вебзастосунку для купівлі комп’ютерних відеоігор Steam

– Epic Games Store (рис. 1.5), заснований у 2018 році, «головний» конкурент першого сервісу, його витворила компанія Epic Games. Він

пропонує конкурентні ціни, безкоштовні ігри щотижня для користувачів магазину та ліпші умови для розробників, викуплення ексклюзивних прав на продаж комп'ютерних забав, але це не допомагає. Проблематика магазину від розробників Fortnite, що їх магазин окрім ігор більше нічого не може запропонувати. Так як він вийшов пізніше Steam, а отже йому потрібно було в нього відбирати аудиторію покупців, але в нього не було відповідного функціоналу в застосунку, навіть досягнень які так обожнюють геймери, як раз та соціальна група яка найбільше купує ігри [9];

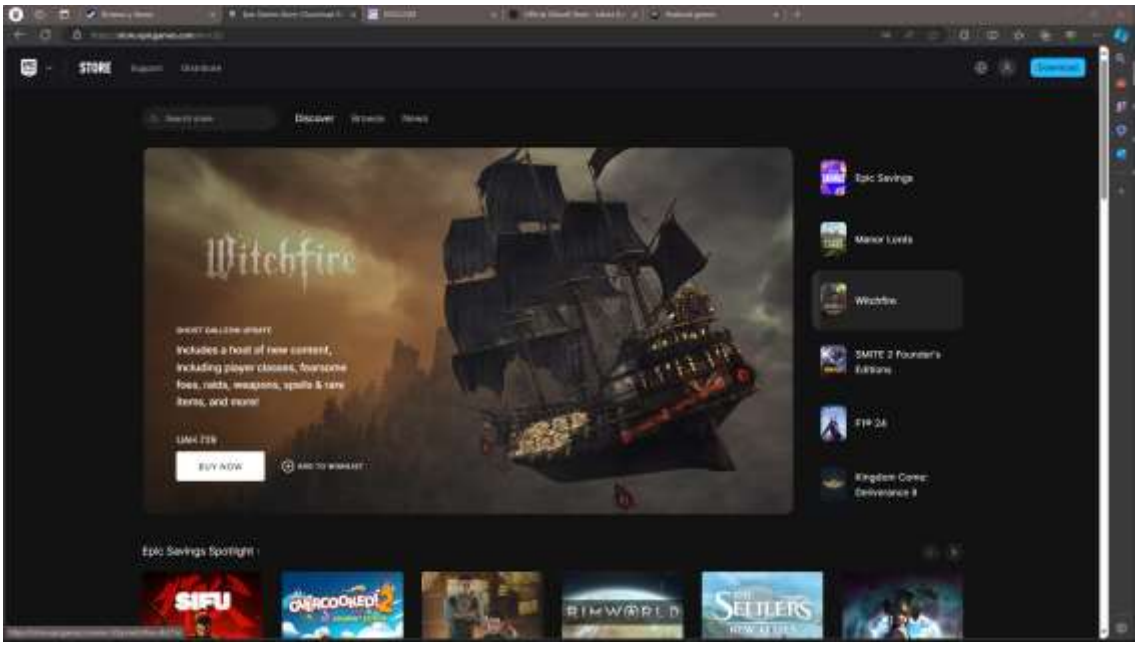


Рисунок 1.5 – Приклад вебзастосунку для купівлі комп'ютерних відеоігор Epic Games Store

– GOG (рис. 1.6), заснований у 2002 році, спеціалізується на продажу цифрових версій по більшості, вже визнаної класики відеоігор без DRM. Тут можна знайти багато ігор 80 та 90 років 20 сторіччя, яких не має навіть в найбільшому цифровому магазині Steam, але тут є й більш нові відеозабави, тільки як зазначалося вище без захисту від піратства. Взагалі це головна спеціалізація крамниці від CD Project Red, ігри без захисту. Деякі антипіратські системи захисту сильно навантажують персональні комп'ютери чи ноутбуки гравців. Тому існує попит на такі ігри без такого захисту. Окрім

того часто до цифрової копій гри йдуть й додаткові матеріали, як то саундтреки чи артбуки, що додає цінності покупці [10].

Рисунок 1.6 – Приклад вебзастосунку для купівлі комп’ютерних відеоігор Epic Games Store

Завдяки цим гігантам, на яких потрібно рівнятися в створенні вебзастосунку для купівлі комп’ютерних ігор, можна взнати найліпші рішення які подобаються гравцям.

1.3 Аналіз найуспішнішого вебсервісу «Steam»

Розглядаючи найбільш вдатні вебзастосунки для покупки комп’ютерних ігор, не годен не визнати Steam як лідера цієї галузі. Його успіх та популярність пояснюються кількома факторами:

– широкий спектр соціальних функцій для користувачів, включаючи можливість обговорювати ігри, обмінюватися думками та відгуками з іншими геймерами;

– можливість створення груп та спільнот, де гравці можуть обмінюватися досвідом та рекомендаціями;

– система досягнень та ігрова статистика, яка надає користувачу додаткову мотивацію для гри;

– розгалужена система категорій та фільтрів, які допомагають користувачам швидко знаходити ігри за жанрами, ціною, рейтингом та іншими параметрами. Це зроблено з метою максимального комфорту та ефективності під час пошуку;

– красиві сторінки користувачів, де є деяка інформація за них, які гри мають, список друзів та інше;

– сторінка «Список бажаного», де покупці залишають ігри які планують купити в майбутньому;

– розділ «Рекомендації» (рис. 1.7), де система аналізує історію покупок та геймплей користувача, пропонуючи персоналізовані рекомендації для майбутніх покупок;

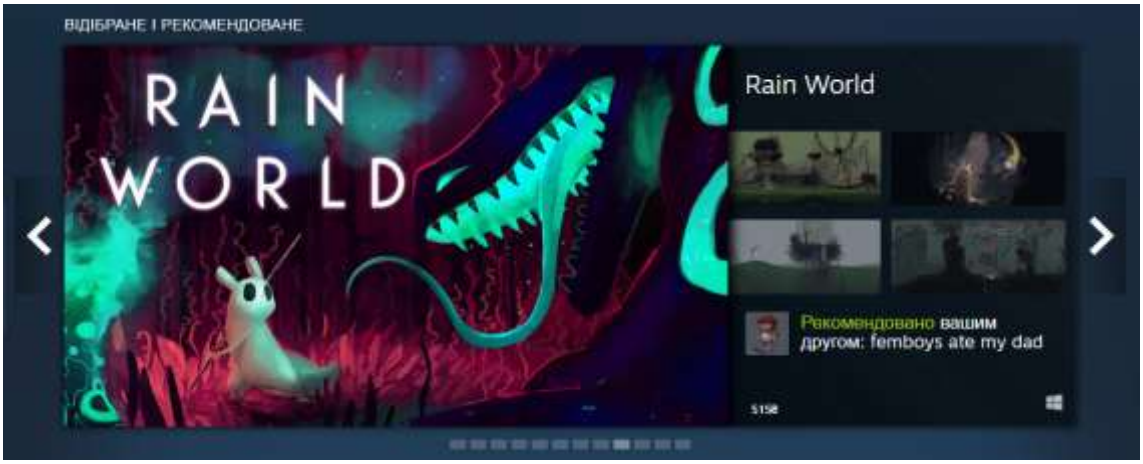


Рисунок 1.7 – Приклад розділу «Рекомендації» в Steam

– розумна система пошуку, яка дозволяє найти ігри не записуючи повну їхню назву;

– розпродажі які видно зразу при вході на головну сторінку вебзастосунка;

– безпека. Вона є всьому, від авторизація до вебзастосунку, до транзакцій. Для входу для ліпшої безпечності можна використовувати двофакторну автенфікацію, чи то через пошту, чи окремий застосунок. Шифрування даних, де всі дані, що передаються між клієнтом Steam та серверами, шифруються для захисту від несанкціонованого доступу. Також існує система відгуків, щоби самі користувачі мали змогу повідомляти про підозрілі чи незаконні дії від інших користувачів;

– сторінка «Діяльність», де можна відслідковувати активність друзів, щ окупив, що добавив до бажаного, з ким почав дружити;

– налаштування бібліотеки, де користувач може організовувати свій список комп'ютерних забав, так він захоче в можливостях функціоналу вебзастоснку;

– коротка статистика для користувачів, як то які зараз ігри в топах продажів, процентна кількість позитивних та негативних відгуків, оцінка від різних видань, також розгорнута для видавців та розробників кількість проданих копій, в який час, з якою мовою інтерфейсі, з найбільш подібними уподобаннями груп покупців та інше;

– детальна сторінка кожної гри, де представлена вся необхідна інформація про продукт: опис гри, відгуки користувачів, системні вимоги, відео та зображення. Це дозволяє покупцям отримати повну картину перед покупкою.

Щодо сторінки гри потрібно розписати детальніше, бо це один з найголовніших елементів для вебзастосунку, що в Steam, EGS, GOG, EA project, Ubisoft Connect, Microsoft store чи Battle.net всюди вигляд сторінки майже однаковий. Це галерея де є зображення, або навіть відео гри. Коротка інформація за гру, хто розробив, хто видавець, жанри гри (рис. 1.8). Звісно є кнопка для купівлі гри, яку добре видно завдяки контрастному вибору кольорів, щоби покупець її точно видіти зобачив, великий розгорнутий опис гри, хто з друзів має цю гру чи бажає. А також мови які гра підтримує та

системні вимоги мінімальні та максимальні, але деякі ігри вказують середні (рис. 1.9).

Рисунок 1.8 – Приклад сторінки гри в Steam

Рисунок 1.9 – Приклад сторінки гри в Steam

З сторінки гри є доступ до сторінки рецензій. Що є хорошим рішенням, так як гравців зразу годні отримати інформацію про гру від інших користувачів. Існує взаємодія з рецензіями від інших геймерів, як вподобайка чи підтвердження кумедності допису, що дає натхнення для написання покупцями цифрових продуктів більше відгуків, а отже підняття активності щодо гри.

Таким чином, обираючи Steam як основу для свого проєкту, планується використовувати його успішні рішення та функціонал для створення зручного

та привабливого вебзастосунку для купівлі комп'ютерних ігор. Його інноваційні підходи та глибокий розуміння потреб геймерської спільноти стануть в нагоді для реалізації ідей та привернення широкої аудиторії до проєкту. Також потрібно зазначити, що будуть використані рішення від інших магазинів, які можуть бути ліпше аніж в розглянуті в цьому розділі крамниці, бо вона не ідеальна, а потрібно вибирати найліпше.

Недоліки є й потрібно їх не повторювати. Steam на свої платформі має багато ігор, що звісно перевага, але дуже багато з них сумнівної якості й потрібно контролювати, що публікувати, як то робить EGS та GOG. Також крамниця від Valve надзвичайно велика й стара вже, що провокує повільну роботу, тому потрібно використовувати мікросервісну та інші рішення для швидкодії сайту. Старий дизайн, хоча в останніх обновленнях це помалому виправляється ситуація, але інші крамниці виглядають куди ліпше та сучасніше, що візуально надає їм перевагу.

1.4 Постановка задачі

Підсумовуючи, розробка вебзастосунку для купівлі комп'ютерних ігор є досить актуальний рішенням. Тому є потреба в розробці сервісу на цю тематику, який годен був би відповідати нинішнім стандартам індустрії сфери продажу цифрових копій відеоігор, яка буде враховувала сучасні тенденції в UI, безпеці та функціоналі сайту.

Об'єктом роботи є вебзастосунок для купівлі комп'ютерних ігор гравцями на вебсайті та можливостями взаємодіяти з рецензіями на ігри.

Метою роботи є розробка вебзастосунку де гравці можуть купити ігри, лишати рецензії та отримувати персоналізовану статистику та рекомендації. А розробники отримують напряму відгуки від гравців. Видавці та розробники мають можливість додавати ігри до вебсайту.

Для досягнення мети необхідно вирішити такі завдання:

– аналіз ситуації на ринку цифрових копій комп’ютерних ігор;

– аналіз вебсервісів які вже існують на ринку;

– дослідити переваги та недоліки інших сервсів;

– реалізувати різні соціальні функції для користувачів. Можливість добавлятися в друзі, видіти їх активіність. Реагувати на чужі рецензії ігор. Надати можливість реагувати на порушення правил спільноти;

– реалізувати алгоритм вираховування тенденцій купівлі ігор на платформі;

– реалізувати систему та алгоритми рекомендацій для користувача;

– реалізувати алгоритми підрахування статистичних даних;

– реалізувати зручний, сучасний та інтуїтивний інтерфейс для зручності покупців;

– реалізувати розумну систему пошуку та фільтрів;

– реалізувати систему знижок на ігри;

– реалізувати систему розпродажів;

– реалізувати бібліотеку ігор користувачів;

– реалізувати сторінки користувачів;

– реалізувати сторінки ігор;

– побудувати архітектуру застосунку на мікросервісні архітектурі задля швидкодії;

– побудувати систему добавлення, редагування та видалення ігор з плтаформи вебзастосунку;

– побудувати надійну систему безпеки, щоби унеможливити злому сторінок користувачів;

– побудувати систему рецензій від користувачів про відеогру;

– наповнити сторінки ігор всією потрібною інформацію про бавку, яка потрібна користувачам.

2 ОБҐРУНТУВАННЯ ПРОГРАМНИХ РІШЕНЬ ПРОЄКТУ ВЕБЗАСТОСУНКУ

2.1 Вибір програмних рішень для Back-end

Вибір програмних рішень для Back-end частини проєкту вебзастосунка є критично важливим етапом розробки, який визначає надійність та швидкодію внутрішньої сторони. Роздивившись цей аспект ближче, можна зрозуміти, як вибір правильних інструментів може вплинути на функціональність, продуктивність, безпеку та масштабованість системи.

По-перше, вибір програмного забезпечення для Back-end має вирішальне значення для забезпечення потрібної функціональності. Він відповідає за обробку даних, бізнес-логіку та взаємодію з базою даних, тому важливо обрати такі інструменти, які забезпечать необхідні можливості для виконання цих завдань ефективно та надійно.

По-друге, вибір правильних програмних рішень для Back-end впливає на продуктивність системи. Ефективність обробки запитів, аналіз даних [11], швидкість відповіді та масштабованість є важливими рішеннями, які потрібно враховувати при виборі технологій Back-end.

По-третє, безпека є одним з ключових аспектів при виборі програмних рішень для Back-end. Забезпечення захисту конфіденційності, цілісності та доступності даних важливо для будь-якого вебзастосунку. Тому важливо обрати такі інструменти, які надають потрібні засоби для аутентифікації, авторизації, шифрування даних та захисту від вразливостей.

Останнє, масштабованість системи, воно є одним з важливіших аспектів, який потрібно враховувати при виборі програмних рішень для back-end. Вебзастосунки можуть зростати в розмірі та обсязі з часом, тому важливо обрати технології, які дозволять легко масштабувати систему за потреби. До прикладу, використовуючи мікросервісну архітектуру або контейнеризації може спростити процес масштабування та розгортання системи.

Отже, вибір програмних рішень для back-end є важливим та відповідальним на майбутнє етапом у розробці вебзастосунка, який може вплинути на всі аспекти його функціональності, продуктивності, безпеки та масштабованості. Правильний вибір технологій дозволить створити надійний, ефективний та безпечний продукт, який задовольнить потреби користувачів та забезпечить успішну реалізацію кваліфікаційної роботи.

2.1.1 Мікросервісна архітектура

Насамперед, про мікросервісну архітектуру. Вона собою являє підхід до розробки програмного забезпечення, де застосунок розбивається на малесенькі незалежні мікросервіси, де кожен виконує певні обмежені функції та має власний код та базу даних. Що надає їм швидкодію та незалежність від інших частин проєкта. Також вони можуть бути розгорнуті, масштабовані чи оновленні незалежно від інших.

Мікросервісна архітектура неймовірно корисна для вебзастосунків таких як онлайн крамниці:

– гнучкість та масштабованість системи. Коли великий монолітний застосунок стає важко масштабованим або різко збільшується навантаження, мікросервіси дозволяють масштабувати лише ті частини системи, які потребують більшої потужності, залишаючи інші незмінними;

– швидкості розробки та впровадження нових функцій. Кожен сервіс може розроблятися та впроваджуватися незалежно, що дозволяє розробнику працювати над окремими частинами застосунку без необхідності робити зміни в інших сервісах. Це спрощує тестування та випробування нових функцій і дозволяє швидко реагувати на зміни в вимогах користувачів або ринкових умовах;

– надійність системи. Якщо один сервіс відмовить або стане недоступним, інші можуть продовжувати працювати нормально. Це дозволяє

підтримувати доступність та продуктивність системи навіть у разі виникнення проблем з окремими сервісами.

У контексті онлайн крамниці мікросервісна архітектура забезпечує можливість ефективно керувати різноманітністю функцій, які надаються користувачам. Словом, окремі сервіси можуть відповідати за обробку замовлень, авторизацію, обробку платежів, рекомендації товарів, оброблення фото [12] та інші аспекти роботи вебзастосунка. Це дозволяє створити гнучку та ефективну систему, яка буде швидко адаптуватися до змін у вимогах ринку та потреб користувачів.

В кваліфікаційної роботі це більш аніж застосовано, проєкт розбитий на багато малесеньких сервісів, які незалежні між собою, але й взаємодіять для функціонування вебзастосунка. До прикладу, є окремі сервіси для авторизації, завантаження даних про гру на сайт, даних за користувача, рецензії, також для додавання чи редагування даних об’єктів та звісно оплати.

2.1.2 Golang

Go (або Golang) – то є мова програмування, яка була розроблена в Google і вперше представлена суспільству в 2009 році. Вона відома своєю простотою, швидкістю та ефективністю у використанні пам’яті.

У веброзробці Go є популярним завдяки своїй швидкості та здатності ефективно обробляти багато запитів. Для вебзастосунків таких як онлайн крамниці, де швидкість та масштабованість грають важливу роль, використання Go є найліпшим рішенням. Він дозволяє обробляти великий потік запитів і забезпечує відмінну продуктивність, що особливо важливо для вебзастосунків з великою кількістю користувачів та транзакцій.

Однією з основних переваг Go для мікросервісної архітектури є його простота у використанні та швидкість розгортання нових сервісів. А як зазначалося вище, мікросервіси часто реалізують в окремі функціональні

блоки застосунку, й Golang ідеально підходить для цієї цілі. Він має чистий синтаксис та простий набір технологій, що дозволяє розробникам швидко створювати та розгортати нові сервіси.

Крім того, Go відомий своєю здатністю працювати з багатопоточністю та паралельною обробкою. Це особливо важливо, коли сервіс може обробляти свої запити швидко та водночас. Особливо таке важливо в вебзастунках онлайн крамниць, де на знижки появляється фест велика кількість покупців цифрових товарів. Ще однією важливою характеристикою Go є його здатність легко інтегруватися з іншими мовами програмування та сервісами. Це робить його ідеальним вибором для вебзастосунків типу онлайн крамниці, де може використовуватися разом з іншими технологіями, такими як бази даних, вебсервери та фронтенд-технології.

В кваліфікаційної роботі ці переваги також закладенні. Використовується багатопочність для ще більш швидкої взаємодії, щоби користувачі мали ліпші враження, а отже відповідати стандартам ринку.

2.1.3 Контейнеризація

У випадку використання мікросервісної архітектури, де застосунок розбивається на невеликі самостійні сервіси, контейнеризація стає незамінною. Кожен мікросервіс може бути упакований в окремий контейнер, що дозволяє незалежно масштабувати, розгортати та керувати кожним з них окремо. Контейнери забезпечують стабільність та цілісність середовища виконання, що спрощує розробку та управління мікросервісами. Словом, так виглядає код для Docker в кваліфікаційній роботі.

Лістинг 2.1 Dockerfile React частини проєкту:

```
FROM node:alpine
WORKDIR /project
```

COPY package.json .

RUN npm install

COPY . .

EXPOSE 4200

CMD ["npm", "start"]

Запишемо алгоритм дій у вигляді кроків:

Крок 1. FROM: цей рядок вказує Docker на використання базового образу «node:alpine» для побудови нового образу. «node:alpine» – це легковажний образ, що містить в собі середовище Node.js, побудоване на базі Linux, яке забезпечує мінімальний розмір і швидкість.

Крок 2. WORKDIR: ця інструкція встановлює робочу директорію контейнера на «/project», що означає, що всі наступні команди будуть виконуватися в цій директорії.

Крок 3. COPY: цей рядок копіює файл «package.json» з локального контексту в директорію «/project» всередині контейнера. «.» означає поточну директорію, тобто копіювання файлу «package.json» з локальної машини в «/project» у контейнері.

Крок 4. RUN: ця команда виконується в процесі збірки образу. Вона встановлює всі залежності, зазначені у файлі «package.json», за допомогою менеджера пакетів npm.

Крок 5. COPY . .: цей рядок копіює всі файли та директорії з локального контексту будівництва (у тому числі файли проєкту React) в поточну робочу директорію контейнера «/project».

Крок 6. EXPOSE: ця інструкція вказує Docker на те, що контейнер прослуховуватиме порт «4200». Це означає, що контейнер буде доступний на порту «4200» для з'єднань з інших контейнерів або зовнішніх джерел.

Крок 7. CMD: ця команда вказує Docker запустити застосунок за допомогою команди «npm start» при старті контейнера.

Також потрібно навести ще приклад для Docker файла сервіса. Різниця між ними невелика, але все таки існує й її потрібно показати для розуміння побудови проєкта. Але буде приведено в приклад лише для одного сервісу, бо вже між різними мікросервісами різниця не така вже й значуща.

Лістинг 2.2 Dockerfile сервіса авторизації:

```
FROM golang:alpine AS builder
WORKDIR /project
COPY . .
RUN go build -o auth .

FROM alpine:latest
WORKDIR /project
COPY --from=builder /project/auth .
EXPOSE 8080
CMD ["./auth"]
```

У цих прикладах Dockerfile описує кожний контейнер окремо. Перший Dockerfile відповідає за React проєкт, а другий за мікросервіс авторизації. Також для інших мікросервісів існують свої Docker файли, які утворюють контейнери. Де кожен контейнер містить необхідні залежності та конфігурацію для запуску відповідного сервісу.

Контейнеризація через Docker дозволяє забезпечити ізоляцію, незалежність та стабільність кожного компонента системи, що робить його ідеальним інструментом для мікросервісної архітектури.

Підсумовуючи програмні рішення для Back-end є насамперед важливими саме в аспектах гнучкості, або ж масштабованості, швидкодії, продуктивності вебзастосунка. Для цього було очевидно вибрано найліпші рішення, як мову програмування Golang для реалізації мікросервісної архітектури та Docker для керування Linux-контейнерів.

2.2 Вибір програмних рішень для Front-end

Вибір програмних рішень для Front-end є важливою складовою будь-якого вебзастосунку, оскільки від них залежить користувацький досвід, ефективність розробки та можливість майбутнього розширення функціональності. Розглянемо деякі ключові аспекти вибору програмних рішень для Front-end та їх вплив на успішність проєкту.

Перш за все, важливо визначити тип програмних рішень, які найбільш підходять для конкретного проєкту. Це можуть бути фреймворки, бібліотеки або навіть різноманітні інструменти для розробки інтерфейсу. Фреймворки, такі як Vue (рис. 2.1 (а)), Angular (рис. 2.1 (б)) або React (рис. 2.1 (в)), який насправді є UI-бібліотека, надають широкі можливості для розробки вебзастосунків з багатофункціональним інтерфейсом, а бібліотеки, наприклад, Redux або MobX, допомагають в управлінні станом застосунку [13].

(а) (б) (в)

Рисунок 2.1 – Логотипи найпопулярніших фреймврорків та бібліотек: (а) Vue; (б) Angular; (в) React

Вибір конкретного фреймворку або бібліотеки також може залежати від потреб проєкту в реалізації певних функцій. Наприклад, якщо потрібно реалізувати динамічний інтерфейс з великою кількістю взаємодії, React може бути ліпшим вибором завдяки своїй швидкості та ефективності в управлінні DOM.

Крім того, важливо враховувати екосистему та спільноту навколо обраних програмних рішень. Наявність активної та підтримуваної спільноти може значно полегшити процес розробки, забезпечуючи доступ до документації, плагінів, шаблонів та інших корисних ресурсів. Також наявність розширень та інтеграцій з іншими інструментами розробки також може вплинути на вибір конкретного фреймворку чи бібліотеки.

Важливо враховувати майбутню масштабованість та гнучкість проєкту при виборі програмних рішень для Front-end. Використання технологій, які підтримують модульність та добре масштабуються, може спростити розвиток проєкту в майбутньому. Наприклад, використання компонентного підходу у React дозволяє розділити інтерфейс на невеликі, незалежні компоненти, що полегшує розробку, тестування та підтримку застосунку з часом.

2.2.1 JavaScript

JavaScript є однією з найбільш популярних мов програмування, особливо в сфері веброзробки. Він широко використовується для створення динамічних інтерактивних вебсторінок та вебзастосунків, оскільки відкриває широкі можливості для реалізації різноманітних функцій та ефектів.

Обрання JavaScript для Front-end розробки є логічним кроком. Його широке поширення та підтримка забезпечують доступність великої кількості ресурсів, документації та спільноти, що спрощує розробку та підтримку проєкту. Крім того, JavaScript є мовою програмування, яка підтримується браузерами без необхідності встановлення додаткових плагінів чи програм. Що собою являє безперечну перевагу.

Однак, якщо йдеться про вибір конкретної технології для розробки інтерфейсу, JSX відображається як потужний інструмент у наборі програмних рішень. JSX є розширенням для JavaScript, яке дозволяє використовувати HTML-подібний синтаксис для опису структури користувацького інтерфейсу.

Це робить код більш читабельним та зрозумілим, особливо для розробників, які знайомі з HTML та CSS.

Використання JSX (рис. 2.2) також дозволяє вбудовувати JavaScript в розмітку, що полегшує маніпулювання даними та станом застосунку. Це забезпечує більшу гнучкість у розробці інтерактивних компонентів та взаємодії з користувачем. Крім того, JSX підтримується більшістю сучасних фреймворків та бібліотек, таких як React, що робить його чудовим вибором для розробки Front-end.

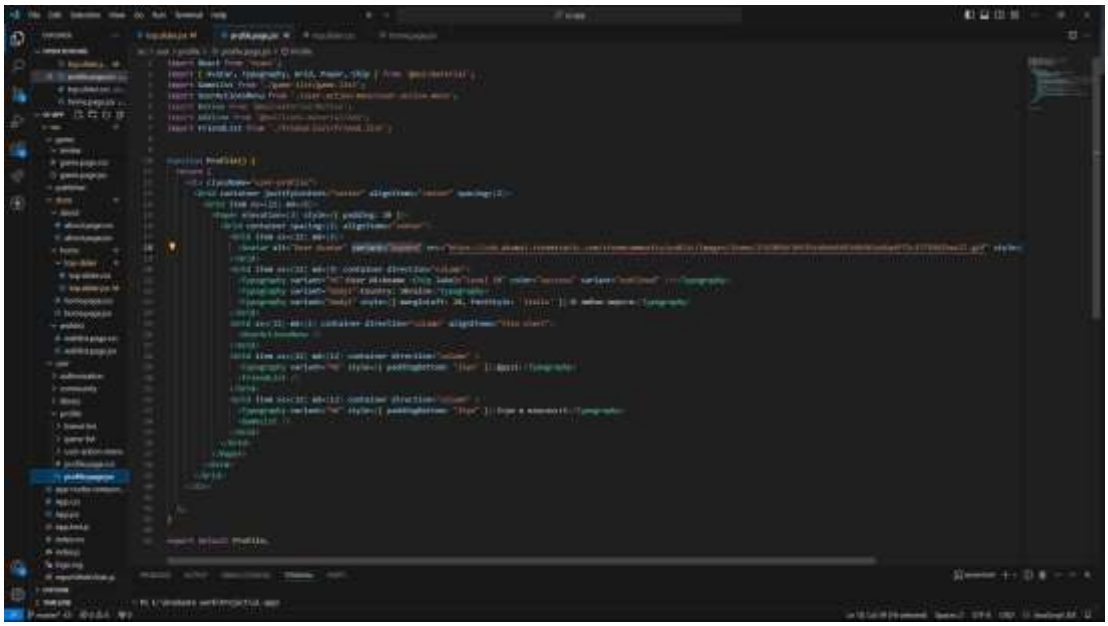


Рисунок 2.2 – Приклад JSX коду

Окрім зручності в написанні коду, використання JSX також дозволяє використовувати переваги сучасних інструментів розробки, таких як перевірка типів та рефакторинг коду. Його динамічний характер дозволяє створювати динамічні та інтерактивні інтерфейси з меншими зусиллями.

Обираючи JavaScript та JSX для розробки Front-end вебзастосунка, отримуються потужні інструменти, які дозволяють створювати ефективні та динамічні користувацькі інтерфейси з меншими зусиллями. Їх широка підтримка та зручність у використанні роблять їх ідеальним вибором для розробки сучасних вебзастосунків.

2.2.2 React

React яв ляє собою бібліотека JavaScript, яка використовується для розробки інтерфейсів користувача. Вона набула великої популярності завдяки своїй ефективності, гнучкості та простоті використання. Для онлайн-крамниць React являється найліпшим вибором з кількох причин.

React дозволяє програмувати складні користувацькі інтерфейси з використанням компонентної архітектури. Це означає, що окремі частини інтерфейсу можуть бути розроблені та підтримані окремо, що спрощує процес розробки та управління кодом. Вебзастосунок виду онлайн-крамниці може містити багато різноманітних елементів, таких як списки товарів, фільтри, кошики покупок, тощо. React дозволяє ефективно управляти всіма цими компонентами.

Надається можливість для створення динамічних інтерфейсів, які реагують на зміни в даних та взаємодію з користувачем без перезавантаження сторінки. Це дозволяє створювати більш інтерактивні та приємні для користувача застосунки. У випадку онлайн-крамниці, це може включати оновлення списку товарів у реальному часу, кошика покупок та інші елементи, що змінюються під час взаємодії користувача з застосунком.

Екосистема та спільнота React є дуже активними, що означає наявність великої кількості сторонніх бібліотек, компонентів та інструментів, які можна використовувати для розширення функціональності застосунку. Це робить React дуже гнучким та розширюваним інструментом для будь-якого типу проєкту.

Всі ці фактори роблять React ідеальним вибором для розробки вебзастосунків таких як онлайн-крамниця. Він дозволяє розробникам створювати сучасні та ефективні застосунки з високою продуктивністю та масштабованістю. Тому потрібно роздивитися як це можливо застосувати в проєкті. До прикладу в кваліфікаційні роботі є сторінка користувача (рис. 2.3). На які є багато різних компонентів, як сама сторінка, а також список ігор,

список друзів та інше. Також використано динамічний інтерфейс, що дозволяє без загрузок переміщатися по сайту. Завдяки активні спільноті існує MUI, який надає змогу спростити написання коду та використовувати готові рішення для тривіальних задач. Ну й безумовно спрощує роботу з дизайном.

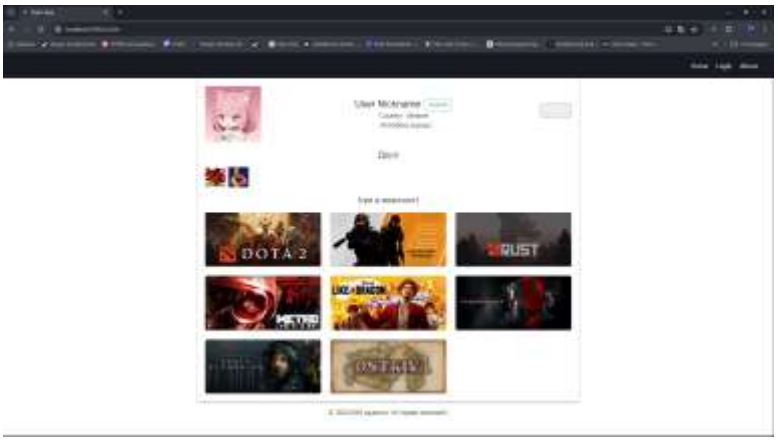


Рисунок 2.3 – Вигляд сторінки користувача

2.3 Вибір програмних рішень для баз даних

Вибір програмних рішень для баз даних є ключовим етапом при розробці вебзастосунка, оскільки від цього залежить ефективність, надійність та продуктивність системи.

По-перше, важливо враховувати тип даних, з якими потрібно буде працювати. Наприклад, для вебзастосунків типу онлайн-крамниці є важливим зберігання великого обсягу товарів та інформації про них, така як назва, опис, ціна, розробник, видавець, зображення гри, жанри, мови, тощо. Також може знадобитися зберігання інформації про користувачів, їхні замовлення та історію покупок. Вибір бази даних, яка найкраще підходить для зберігання цих даних, може значно полегшити роботу над проєктом та забезпечити оптимальну продуктивність системи.

По-друге, важливо враховувати вимоги до продуктивності та масштабованості системи. Наприклад, якщо вебзастосунок, має велику

кількість активних користувачів та великі обсяги даних, важливо обрати базу даних, яка забезпечить швидкий доступ до даних та ефективне їхнє зберігання. Також важливо мати можливість легко масштабувати базу даних відповідно до зростання обсягів даних та навантаження на систему. А так як любий магазин самого початку має цілитися на чим більші аудиторію покупців, то це також неймовірно важливий аспект.

По-третє, безпека даних є одним з найважливіших аспектів при розробці вебзастосунків. Обраний тип бази даних повинен забезпечувати високий рівень захисту даних від несанкціонованого доступу та зловживання. Для цього може бути важливим використання різноманітних механізмів шифрування, аутентифікації та авторизації користувачів, а також регулярне оновлення та моніторинг системи з метою виявлення та усунення потенційних загроз безпеці.

MongoDB є однією з найпопулярніших NoSQL баз даних, відомою своєю гнучкістю та продуктивністю. Вибір MongoDB як основного сховища даних для вебзастосунку такого як онлайн-крамниця принесе багато переваг завдяки її особливостям та доступним програмним рішенням.

Використання документної моделі даних, де дані зберігаються у вигляді документів JSON-подібного формату, BSON (рис. 2.4). Це дозволяє зберігати складні та вкладені структури даних у закодовані формі, що значно спрощує роботу з ними. Для онлайн-крамниці це може бути особливо корисним, оскільки дозволяє легко зберігати інформацію про відеоігри, замовлення та користувачів без необхідності нормалізації даних, як у реляційних базах даних.

```
8c 00 00 00                         # Розмір документа (140 байт)
02 6c 6f 67 69 6e 00 06 00 00 00 75 73 65 72 35 00  # Поле login: "user5"
02 65 6d 61 69 6c 00 13 00 00 00 75 73 65 72 35 40 65 78 61 6d 70 6c 65 2e 63 6f 6d 00  # Поле email: "user5@example.com"
02 70 61 73 73 77 6f 72 64 00 0a 00 00 00 70 61 73 73 77 6f 72 64 35 00  # Поле password: "password5"
02 63 6f 75 6e 74 72 79 00 04 00 00 00 55 53 41 00  # Поле country: "USA"
02 66 75 6c 6c 5f 6e 61 6d 65 00 01 00 00 00 00  # Поле full_name: ""
02 70 6f 73 74 63 6f 64 65 00 06 00 00 00 35 34 33 32 31 00  # Поле postcode: "54321"
02 67 61 6d 65 5f 6e 61 6d 65 00 0c 00 00 00 75 73 65 72 32 5f 67 61 6d 65 72 00  # Поле game_name: "user2_gamer"
02 75 73 65 72 5f 69 64 00 24 00 00 00 30 35 35 38 66 39 38 37 2d 38 38 63 37 2d 34 35 32 34 2d 38 34 37 63 2d 36 38 61 36
# Поле user_id: "0558f987-88c7-4524-847c-68a60a8affbc"
00  # Кінець документа
```

Рисунок 2.4 – Приклад моделі User в BSON форматі

Один з ключових аспектів MongoDB – це масштабованість. А як зазначається протягом всієї кваліфікаційної роботи, це майже що найважливіший елемент для вебзастосунків. База даних підтримує горизонтальне масштабування через збереження записів даних на декількох машинах, що дозволяє розподіляти дані між кількома серверами. Це дуже важливо для великих вебзастосунків, які мають обробляти значні обсяги даних та забезпечувати високу продуктивність навіть при великому навантаженні.

Іншою важливою перевагою MongoDB є гнучкість у схемі даних. Можливо додавати нові поля до документів без потреби оновлювати всю базу даних, що робить процес розробки більш гнучким та швидким. Це дозволяє швидко адаптуватися до змін у вимогах бізнесу та зменшити час на оновлення структури даних.

Щодо програмних рішень, для MongoDB доступний широкий спектр інструментів та бібліотек. Один з ключових інструментів є MongoDB Atlas, це хмарна платформа, яка забезпечує повністю керовану базою даних інфраструктуру. Atlas автоматизує налаштування, обслуговування, моніторинг та резервне копіювання бази даних, що дозволяє зосередитися на розробці кваліфікаційної роботи, а не на адмініструванні БД.

2.4 Вибір програмних рішень для всієї програми

Вибір правильних принципів і підходів до написання коду є фундаментально важливим для успіху будь-якого програмного проєкту. Це забезпечує читабельність, підтримуваність та розширюваність коду. Крім того, дотримання цих принципів допомагає мінімізувати кількість помилок та сприяє ефективній роботі. Застосування найкращих практик програмування дозволяє створювати гнучкі та надійні системи, які будуть легко адаптуватися до змін.

2.4.1 SOLID

Одним із основних принципів є SOLID. SOLID є акронімом, що включає в себе п’ять ключових принципів об’єктно-орієнтованого програмування:

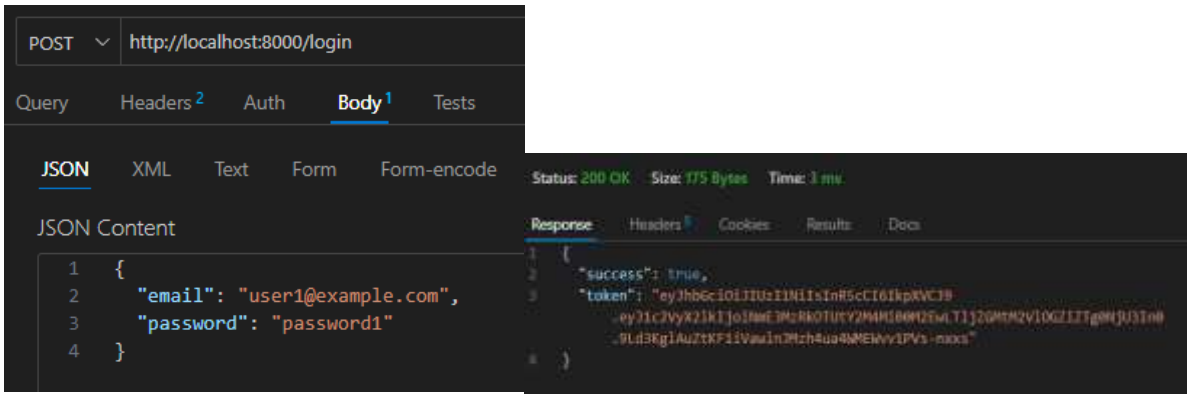
- принцип єдиної відповідальності (Single Responsibility Principle);
- принцип відкритості/закритості (Open/Closed Principle);
- принцип підстановки Ліскова (Liskov Substitution Principle);
- принцип поділу інтерфейсу (Interface Segregation Principle);
- принцип інверсії залежностей (Dependency Inversion Principle).

Ці принципи допомагають структурувати код таким чином, щоб кожен клас мав тільки одну причину для зміни, дозволяли розширювати функціональність без змін існуючого коду, забезпечували можливість заміни базових класів підкласами без порушення логіки програми, розділяли великі інтерфейси на вузькі, та зменшували залежність між модулями.

2.4.2 REST API

REST API, або репрезентативна передача стану, є архітектурним стилем для створення вебсервісів. Використання REST API дозволяє створювати вебзастосунки, які є легко масштабованими та інтегрованими з іншими сервісами. Ключові принципи REST включають використання HTTP-методів (GET, POST, PUT, DELETE) для маніпулювання ресурсами, представлення ресурсів у вигляді уніфікованих URL, та безстанове з’єднання між клієнтом і сервером. Дотримання цих принципів забезпечує простоту, гнучкість і високу продуктивність системи, що є критично важливим для сучасних вебзастосунків. Словом, в кваліфікаційні роботі є авторизація користувача, де на мікросервіс приходе JSON форма з даними які вводить користувач в форму логіна (рис. 2.5 (а)), а на сервер вже зашифровано в токені приходить

унікальний ідентифікатор користувача (рис. 2.5 (б)), щоби вже сайт годен був через нього отримувати потрібні дані через інші запити інформацію.



(а) (б)

Рисунок 2.5 – Приклад POST-запита для авторизації:
(а) POST запит на сервіс; (б) дані які відправляє сервіс на сайт

2.4.3 Чистий код

Чистий код (Clean Code) – це підхід, що акцентує увагу на написанні коду, який є зрозумілим для інших розробників, легким у підтримці та тестуванні. Чистий код повинен бути простим і прямолінійним, кожна функція має виконувати лише одну дію, а імена змінних і функцій повинні бути зрозумілими та самодокументуючими. Писання чистого коду також включає видалення зайвого коду, уникнення дублювання, та організацію коду в логічні модулі. Цей підхід допомагає зменшити технічний борг і полегшує процес оновлення та підтримки програмного забезпечення в майбутньому [14]. Дотримання принципів чистого коду сприяє підвищенню продуктивності програми, оскільки можливо швидше зрозуміти і модифікувати існуючий код. Впровадження цього підходу також покращує якість кінцевого продукту, роблячи його більш надійним та менш схильним до помилок.

2.4.4 Асинхронність

Асинхронність у програмуванні дозволяє виконувати декілька операцій одночасно, не блокуючи основний потік виконання. Це особливо важливо в кваліфікаційні роботі, де необхідно обробляти численні запити від користувачів, працювати з базою даних, взаємодіяти з іншими сервісами тощо. Асинхронність забезпечує ефективне використання ресурсів і підвищує продуктивність системи.

Прикладом, може бути асинхронний метод для відправки даних про гру на сайт з мікросервіса. Асинхронність у поданому коді реалізована за допомогою горутин і каналів, що є ключовими концепціями мови Go для побудови асинхронних програм. Горутини – це легкі потоки виконання, які дозволяють виконувати функції паралельно, не блокуючи основний потік програми [15]. Функція sendGameData запускається у новій горутині за допомогою ключового слова go, що забезпечує асинхронне виконання цієї функції. Основний потік продовжує виконання, не чекаючи завершення функції sendGameData.

Лістинг 2.3 Асинхронна функція надсилання даних на мові Golang:

```
func sendGameData(game Game, ch chan<- bool) {
    gameData, err := json.Marshal(game)
    if err != nil {
        log.Printf("Error marshaling game data: %v", err)
        ch <- false
        return    }
    url := "https://example.com/api/games"
    req, err := http.NewRequest("POST", url, bytes.NewBuffer(gameData))
    if err != nil {
        log.Printf("Error creating request: %v", err)
        ch <- false
```

```
        return
    }
    req.Header.Set("Content-Type", "application/json")
    client := &http.Client{Timeout: 10 * time.Second}
    resp, err := client.Do(req)
    if err != nil {
        log.Printf("Error sending request: %v", err)
        ch <- false
        return
    }
    defer resp.Body.Close()
    if resp.StatusCode != http.StatusOK {
        log.Printf("Unexpected status code: %d", resp.StatusCode)
        ch <- false
        return
    }
    log.Println("Game data sent successfully")
    ch <- true
}
```

Якщо роздивлятися асинхронність на стороні Front-end, то в прикладу можна привести знову ту саму ситуацію, але вже з отриманням даних на сайт. В JSX асинхронні операції зазвичай виконуються за допомогою таких конструкцій, як fetch, axios або інших бібліотек для здійснення запитів на сервер. Ці операції можуть відбуватися в фоновому режимі, тобто без блокування інших операцій в застосунку. Так як JS не підтримує загальноприйняту асинхронність, то в нього існує своя. Де програма чекає, доки запит не завершиться, і тільки після цього вона продовжить свою роботу. Це дозволяє користувачеві відправляти та отримувати дані, не чекаючи на

завершення операцій, що може поліпшити швидкодію та відзивчивість застосунку.

Лістинг 2.4 Асинхронна функція приймання даних на мові JavaScript:

```
const fetchGameData = async () => {
  try {
    const response = await fetch('https://example.com/api/games', {
      method: 'GET',
      headers: {
        'Content-Type': 'application/json',
      },
    });
    if (!response.ok) {
      throw new Error(`HTTP error! Status: ${response.status}`);  }
    const data = await response.json();
    setGameData(data);
  } catch (err)
{
    setError(err.message);
  } finally
{
    setLoading(false);
  } };
```

У вищенаведеному прикладі асинхронність реалізована за допомогою fetch, який виконує запит на сервер асинхронно та не блокує виконання інших частин програми. Після завершення запиту відбувається оновлення стану компонента, що призводить до перерендерингу інтерфейсу користувача з урахуванням отриманих даних.

2.4.5 Модульність

Модульність для вебзастосунка являє собою ключовий аспект для поліпшення структури та підтримки коду на рівні компонентів та функцій. Розбиття програми на невеликі, незалежні модулі дозволяє легше розробляти, тестувати та розширювати програму в майбутньому.

Кожен модуль може виконувати певну функціональність, яка може бути наново використана в різних частинах застосунку. До прикладу цього випадку, в проєкті є сторінка користувача, яка розбита на пару модулів (рис. 2.6). Де окрім головного елементу є додаткові модулі, як список друзів, список ігор та меню взаємодії з користувачем. Словом, список друзів також використовується на сторінці гри, бо вигляд вивід даних там подібний, але вони різні. Тому використовується окремий модуль в який залежно від ситуації передаються різні масиви даних, якщо це сторінка користувача, то його список друзів, якщо сторінка гри, то список друзів що мають цю гру.

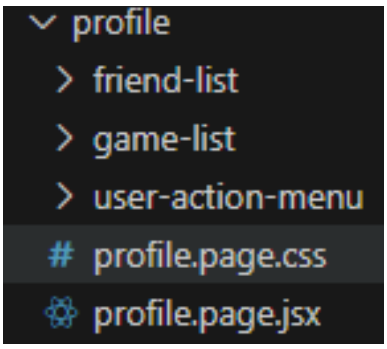


Рисунок 2.6 – Приклад директорій яка розбита на окремі модулі

Модульність допомагає покращити перевикористання коду, що було зазначено в прикладі раніше, оскільки окремі модулі можуть бути легко відокремлені та використані в інших проєктах або частинах програми. Це допомагає зменшити дублювання коду.

3 РОЗРОБКА ВЕБЗАСТОСНУКУ

3.1 Вирішення програмних рішень баз даних

Для збереження мультимедійних файлів було використано Firebase Cloud Storage, який дозволяє зберігати великий обсяг даних, в випадку кваліфікаційної роботи це зображення та відео для ігор, а також знімки для аватарок користувачів.

Що стосується текстових даних, то в попередньому розділі було зазначено, що найліпший вибір для БД то є MongoDB, а отже тепер потрібно розглянути як встановити, налаштувати та створити потрібні бази даних для проєкта.

Насамперед потрібно завантажити локальний дистрибутив MongoDB, але це не є обов'язково, так як існує хмарне рішення, MongoDB Atlas. Яке може дати поміч випадку зміни пристрою на якому ся вела робота. Для цього потрібно реєстрація на платформі, а після цього можна створювати новий кластер.

3.1.1 Створення баз даних

Наступний крок – налаштування бази даних та колекцій. У MongoDB дані організовуються в бази даних, які містять колекції документів. Колекції аналогічні таблицям у реляційних базах даних, а документи – це записи. У консолі MongoDB або через MongoDB Atlas створюю базу даних для вебкрамниці, до прикладу візьму базу для мікросервісів що взаємодіяють з інформацією користувачів, «users_store», а в ній кілька основних колекцій: «users», «users_libraries», «orders», «wishlist», «users_review», «friends».

Далі необхідно визначити структуру даних для кожної колекції. Наприклад, колекція «users» містить такі поля, як:

– login – назвисько користувача для входу до крамниці;

– email – електрона пошта для реєстрації та відновлення доступу до акаунта в випадку втрати пароля;

– password – пароль;

– country – країна проживання користувача;

– full_name – повне ім’я користувача;

– postcode – поштовий індекс користувача;

– game_name – ігрове назвисько користувача, яке буде відображатися в його профілі;

– user_id – унікальний ідентифікатор користувача, для взаємодії з сервісами.

Решта колекцій виконують функцію для «зв’язування» користувачів з іграми, друзями, рецензіями, замовленнями. Таблиця «users_libraries» записує до себе унікальний «user_id» та «game_id», для того щоби відображати бібліотеку куплених ігор покупців вебзастунка. Колекція «orders» записує до себе час, унікальний ідентифікатор користувача, яку саме бавку. А вже «wishlist» існує для збереження бажаного користувача, як він захоче купити гру пізніше [16]. Для рецензій також існує своя колекція, це «users_review» де вказані ідентифікатори користувача та ідентифікатор рецензії, так як рецензент може мати багато рецензії, то тут буде існувати зв’язок один до багатьох, як у випадку бібліотеки ігор, бажаного та замовлень. Ну й для функціоналу системи друзів існує зв’язуюча таблиця «friends», де вже ситуація один до одного.

Також існують окремі бази даних для ігор та транзакцій, для ліпшої швидкодії вебзастунку. Де формування даних подібне, але мікросервісна архітектура надає перевагу немонолітним базам даних. Але потрібно зазначити, що відеоігри, «games_store», в себе включають такі таблиці: «game», «publisher», «developer», «developer_publisher», «game_developer», «game_publisher». А транзакції невеличкі та мають лише одну колекцію, де зберігаються дані за замовлення цифрового продукту.

3.1.2 Налаштування безпеки

Безпека бази даних є надзвичайно важливою, особливо для вебзастосунків, які оперують чутливою інформацією користувачів, до прикладу купівля товарів через інтернет з вказування своїх кредитних даних. MongoDB забезпечує кілька рівнів захисту для гарантування цілісності та конфіденційності даних. Основні механізми безпеки, які пропонує MongoDB:

- аутифікація;
- авторизація;
- шифрування даних;
- аудит;
- моніторинг.

Лістинг 3.1 Приклад налаштування аутентифікації та авторизації:

```
use admin
db.createUser(
  {
    user: "siteAdmin",
    pwd: "securePassword123",
    roles: [ { role: "userAdminAnyDatabase", db: "admin" }, { role: "readWriteAnyDatabase", db: "admin" } ]
  }
)
use myDatabase
db.createUser({
    user: "appUser",
    pwd: "anotherSecurePassword456",
    roles: [ { role: "readWrite", db: "myDatabase" } ]
  }
)
```

Спершу відбувається аутентифікація, яка відбувається за паролем. Потім авторизація, де визначається, що дозволено робити аутентифікованим користувачам. MongoDB реалізує модель на основі ролей (RBAC – Role-Based Access Control), де кожному користувачу призначаються певні ролі, що визначають його права доступу. На щастя база даних сама забезпечує шифрування інформації, яку містить. Як в стані пошуку, де дані, що зберігаються на диску, шифруються за допомогою механізму WiredTiger з можливістю використовувати власні ключі шифрування. А під час передачі використовується TLS/SSL для забезпечення захищеного каналу зв'язку між клієнтами та серверами MongoDB. Також сховище даних підтримує механізм аудиту, що дозволяє відстежувати дії користувачів та операції в базі даних. Що допомагає виявляти та розслідувати підозрілі дії або конфігураційні проблеми. Аудит-лог може зберігати інформацію про успішні та невдалі спроби аутентифікації, операції CRUD (створення, читання, оновлення, видалення), а також інші адміністративні дії. А для моніторингу використовується MongoDB Atlas, де є графіки продуктивності, оповіщення про проблеми та інструменти для оптимізації запитів.

3.2 Вирішення програмних рішень Back-end

Для вирішення програмних рішень Back-end використовується програмне середовище Visual Studio Code та мова програмування Golang. Насамперед потрібно перейти на офіційний сайт від Microsoft для текстовий редактор, завантажити його звітам, встановити і після цього можна його налаштувати для роботи з Golang. Так як в звичайні версії застосунка не має підтримки цієї мови програмування, то потрібно встановити розширення від самих же розробників мови програмування, яке надає змогу компілятору видіти помилки в коді, воно називається «Go». Для цього потрібно зайти на

сторінку розширень та в пошуку вести назву потрібного розширення (рис. 3.1).

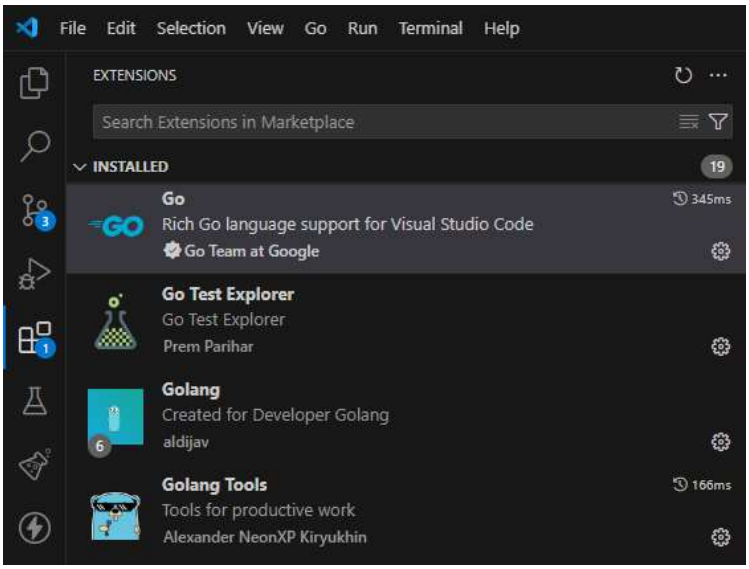


Рисунок 3.1 – Розширення для мови Go в VS Code

Після чого можна завантажувати сам Golang. Що також просто робиться з офіційного сайту. Як буде встановлено мову програмування можна буде приступати до самих мікросервісів.

3.2.1 Мікросервіс для авторизації

Мікросервіс для авторизації є важливим компонентом вебзастосунка, що забезпечує функціонал логіну та реєстрації користувачів. Це важлива частина будь-якого онлайн-сервісу, оскільки вона відповідає за безпеку та контроль доступу користувачів до системи і прив’язку ігор до певного користувача, якщо він їх купляв. Мікросервіс авторизації, зокрема, виконує такі завдання, як зберігання даних користувачів, перевірка облікових записів, управління сесіями та забезпечення захисту від несанкціонованого доступу.

Розробка такого мікросервісу включає використання безпечних методів зберігання паролів, наприклад, хешування за допомогою алгоритмів bcrypt. Це

гарантує, що навіть у випадку компрометації бази даних, зловмисники не зможуть легко отримати доступ до паролів користувачів. Під час реєстрації новий користувач вводить свої дані, які потім проходять валідацію та зберігаються в базі даних. Для забезпечення зручності використання та безпеки, до даних користувачів додаваються унікальні токени, що дозволяють відстежувати сесії та зберігати стан авторизації.

Логін процес включає перевірку введених даних з тими, що зберігаються в базі. Якщо введені дані коректні, користувач отримує доступ до системи і йому надається сесійний токен (рис. 3.2), який використовується для подальшої автентифікації запитів. Це дозволяє уникнути повторного введення логіну і паролю для кожної дії користувача на сайті, підвищуючи загальну зручність використання.

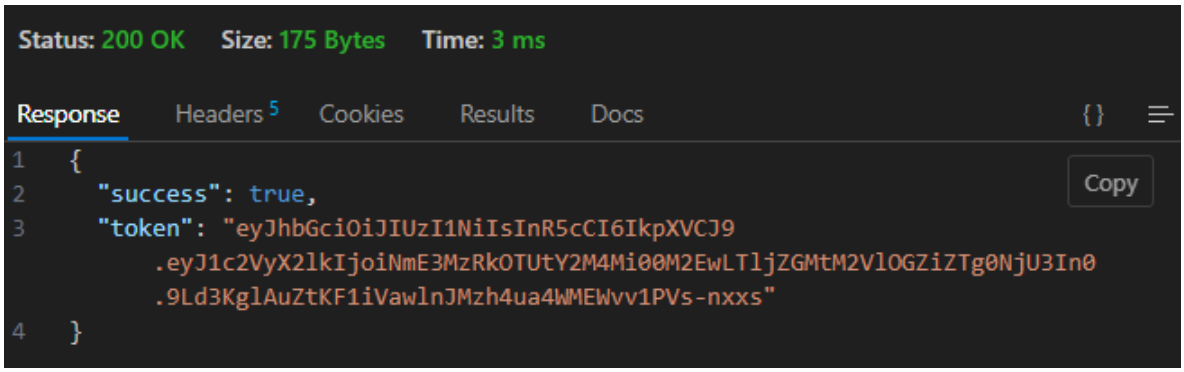


Рисунок 3.2 – Приклад запиту логіну з сесійним токеном

Реєстрація користувачів в свою чергу у мікросервісі є процесом, що забезпечує створення нових облікових записів, дозволяючи новим користувачам отримати доступ до функціоналу вебзастосунка. Коли користувач надсилає свої дані, такі як ім’я, email, пароль та інші необхідні відомості, ці дані спочатку проходять через процес валідації, щоб забезпечити правильність та повноту інформації. Після успішного проходження валідації пароль користувача хешується з використанням безпечного алгоритму, що запобігає можливим витокам даних. Далі дані зберігаються в базі даних, створюючи новий запис користувача. У відповідь новому користувачу

надіслано підтвердження реєстрації через email, що додає додатковий рівень безпеки і впевненості у правильності введених даних. Це не лише покращує захист від несанкціонованого доступу, але й забезпечує більш персоналізований підхід до кожного користувача, закладаючи основи для подальшого безпечного та комфортного користування вебзастосунком.

3.2.2 Мікросервіс для транзакцій

Що стосується мікросервіса для транзакцій то він певне один з найважливіших у функціонуванні онлайн-крамниці, забезпечуючи безпечний та ефективний процес обробки платежів. Цей мікросервіс відповідає за всі фінансові операції між користувачами та платформою, включаючи покупки, повернення коштів та управління платіжними методами. Коли користувач ініціює покупку, мікросервіс отримує запит, перевіряє наявність необхідних даних та валідність платіжної інформації.

Після успішного проведення платежу мікросервіс оновлює статус замовлення в базі даних, забезпечуючи миттєве відображення змін для користувача. У разі виникнення помилок або проблем з обробкою платежу, мікросервіс надає відповідні повідомлення та інструкції для користувача щодо подальших дій. Важливим аспектом роботи цього мікросервісу є його інтеграція з іншими компонентами системи, такими як мікросервіси авторизації, користувачів та ігор, що дозволяє забезпечити цілісність та узгодженість даних у всій системі.

Безпека є одним із найважливіших аспектів мікросервісу для транзакцій. Всі платіжні дані шифруються та передаються через захищені канали зв’язку, що запобігає несанкціонованому доступу та зловживанням. Крім того, мікросервіс забезпечує відповідність стандартам PCI DSS, що гарантує високий рівень захисту платіжних даних. Регулярні перевірки та моніторинг

транзакцій допомагають виявляти та запобігати можливим шахрайським діям, забезпечуючи довіру користувачів до платформи.

3.2.3 Мікросервіс для CRUD

Сервіс для CRUD (створення, читання, оновлення та видалення) є важливою складовою будь-якого вебзастосунку, оскільки він дозволяє адміністраторам та розробникам ефективно керувати вмістом сайту та даними про ігри. У цьому сервісі існує функціонал для адмінів та розробників/видавців, причому перші мають можливість редагувати будь-яку інформацію на сайті, тоді як другі мають доступ лише до ігор, які належать їм.

Для адміністраторів функціонал CRUD означає повний контроль над контентом та налаштуваннями вебзастосунку. Вони можуть додавати нові ігри, редагувати існуючі, видаляти застарілі, керувати користувачами та виконувати інші адміністративні завдання. Цей доступ є критично важливим для забезпечення актуальності та правильності інформації на сайті.

З іншого боку, розробники та видавці мають доступ лише до ігор, які вони створили або видавці, яким належать. Вони можуть оновлювати інформацію про свої ігри, додавати нові випуски, виправляти помилки та взагалі керувати контентом [17-20], який вони пропонують користувачам. Це дозволяє їм забезпечувати якісну підтримку своїх ігор та ефективно взаємодіяти зі своєю аудиторією.

3.2.4 Мікросервіс для рецензій

Сервіс для рецензій дозволяє користувачам ділитися своїми думками та враженнями від ігор з іншими. Цей мікросервіс працює в асинхронному

режимі, що дозволяє ефективно обробляти великий потік даних та забезпечує швидкий доступ до рецензій користувачів.

Асинхронний режим роботи є ключовим аспектом сервісу для рецензій. Він дозволяє мікросервісу ефективно опрацьовувати велику кількість запитів на додавання та вивід рецензій, забезпечуючи при цьому плавну та безперебійну роботу. Коли користувач додає рецензію, запит на її збереження обробляється асинхронно, що дозволяє йому продовжувати взаємодіяти з іншими функціями вебзастосунку без затримок.

Мікросервіс для рецензій просто записує та виводить рецензії про ігри. Користувачі можуть додавати свої власні рецензії, вказуючи свої враження від гри, її переваги та недоліки. Крім того, вони можуть переглядати рецензії інших користувачів, що дозволяє їм отримувати об’єктивну інформацію про ігри перед покупкою.

3.2.5 Мікросервіс для обробки статистики

Мікросервіс для статистики надає можливість аналізу та візуалізації різноманітних даних про активність користувачів та функціонування системи.

Основні функції мікросервісу включають аналіз та обробку різноманітних даних. Завдяки різним запитам користувачі можуть отримувати різноманітні статистичні дані, такі як загальна статистика по магазину, де враховуються всі ігри які є в крамниці, статистика гравця, де враховуються бавки лише в власності покупця, статистика видавця або розробника, де враховуються комп’ютерні ігри цих програмістів [21-23].

Однією з ключових функцій мікросервісу є можливість генерації звітів та візуалізація даних у вигляді графіків, діаграм, таблиць та інших візуалізацій вже на Front-end частині для приємно розуміння користувач.

3.2.6 Сервіс для даних про користувача

Сервіс для даних про користувача забезпечує користувачів необхідною функціональністю та зручним інтерфейсом для управління своїм профілем, як то зміна знімки на сайті [24]. Він відповідає за відображення та оновлення інформації на сторінці користувача, включаючи персональні дані, список друзів, бібліотеку ігор, бажані покупки та рецензії. Коли користувач заходить на свою сторінку, сервіс завантажує та відображає актуальну інформацію, забезпечуючи швидкий доступ до всіх необхідних функцій.

Для початку розглянемо функціонал добавлення в друзі, він дозволяє користувачам розширювати своє коло спілкування, шукаючи та додаючи інших гравців у свій список друзів. Цей сервіс підтримує надсилання запитів на дружбу, підтвердження та відхилення запитів, а також управління існуючими друзями. Коли користувач додає нового друга, запит асинхронно надсилається на сервер, де обробляється і, в разі успішного підтвердження, новий друг додається до списку. Це забезпечує швидку взаємодію і безперебійну роботу без блокування інтерфейсу.

Коли майбутній покупець додає гру до списку бажаного – це собою являє ще одну важливу функцію, яка дозволяє користувачам відстежувати ігри, які вони планують придбати у майбутньому. Коли користувач додає гру до списку бажаного, асинхронний запит надсилається до сервера, де ця інформація зберігається у базі даних. Це забезпечує миттєве оновлення списку бажаного без затримок у роботі сайту.

Процес купівлі гри є складнішим і вимагає інтеграції з мікросервісом транзакцій. Коли користувач вирішує придбати гру, сервіс ініціює асинхронний процес транзакції, який включає перевірку платіжної інформації, обробку платежу та підтвердження успішності транзакції. Після успішної обробки транзакції, гра автоматично додається до бібліотеки користувача. Цей процес також включає оновлення інформації про користувача і його ігрову

бібліотеку у базі даних, що дозволяє користувачам миттєво бачити придбану гру у своєму профілі.

Сервіс також підтримує функціонал написання рецензій та виставлення оцінок іграм. Користувачі можуть залишати свої відгуки про придбані ігри, допомагаючи іншим гравцям приймати рішення про покупку. Коли користувач пише рецензію або виставляє оцінку, асинхронний запит надсилається на сервер для збереження цих даних у базі. Це гарантує, що сайт працює швидко і ефективно, без зайвих затримок.

Важливість асинхронності в цих методах не можна переоцінити. Всі запити на сервер виконуються у фоновому режимі, що дозволяє користувачам продовжувати роботу на сайті без переривань. Це забезпечує плавний і швидкий користувацький досвід, зменшуючи час очікування і підвищуючи загальну продуктивність вебзастосунку.

3.2.7 Сервіс для даних про гру

Сервіс для даних про гру, інформаційно є найважливішим компонентом онлайн-крамниці, оскільки саме через нього користувачі отримують детальну інформацію про ігри, розробників, видавців і франшизи. Це великий і складний сервіс, який потребує ретельного опрацювання та інтеграції з іншими мікросервісами, аби забезпечити повноцінну і безперебійну роботу вебзастосунку.

Вивід інформації про гру є ключовою функцією цього сервісу. Користувачі можуть переглядати опис гри, рецензії, оцінки, які друзі мають гру в наявності, скріншоти, відео та інші медіафайли [25-29], які допомагають їм приймати рішення про покупку. Інформація про гру включає також технічні характеристики, такі як системні вимоги, жанри, локалізації і дату випуску. Все це робить сервіс для даних про гру невід’ємною частиною користувацького досвіду.

Розробник та видавець є важливою частиною інформації про гру. Ці дані допомагають користувачам зрозуміти, хто стоїть за створенням та розповсюдженням гри, що може бути важливим фактором при прийнятті рішення про покупку. Наприклад, відомі розробники та видавці можуть мати великий вплив на рішення користувачів завдяки своїй репутації і попереднім успіхам. Сервіс дозволяє користувачам переглядати інші ігри від тих самих розробників і видавців, що може сприяти збільшенню продажів.

Франшиза – ще один важливий аспект, який враховується в сервісі для даних про гру. Ігри, що належать до відомих франшиз, можуть мати великий вплив на зацікавленість користувачів. Інформація про франшизу допомагає користувачам знаходити ігри, які є частиною серії, що вони вже люблять. Це включає в себе перегляд попередніх та майбутніх релізів у межах однієї франшизи, що забезпечує послідовний і зручний досвід для користувачів.

Асинхронність є критичним елементом в роботі цього сервісу. Всі запити на отримання інформації про гру виконуються асинхронно, що дозволяє зменшити час очікування і підвищити продуктивність вебзастосунку. Коли користувач запитує інформацію про гру, асинхронний запит надсилається на сервер, де обробляється і повертається відповідь. Це дозволяє користувачам продовжувати взаємодіяти з сайтом без затримок, забезпечуючи плавний та безперебійний досвід.

На Golang є різні технології для реалізації цього сервісу. Однією з ключових є Gorilla Mux, яка забезпечує потужну маршрутизацію запитів і дозволяє легко налаштовувати маршрути для отримання інформації про гру. Крім того, можна використовувати бібліотеки для роботи з JSON, такі як encoding/json, які дозволяють ефективно серіалізувати та десеріалізувати дані.

Цей сервіс також інтегрується з іншими мікросервісами, такими як сервіс для інформації про користувачів, який забезпечує дані з ким друже користувач, щоби вивести чи є в них ця гра та чи рекомендують вони цей цифровий продукт, а також мікросервіс транзакцій, який обробляє покупки та забезпечує оновлення інформації про гру в режимі реального часу. Це робить

сервіс для даних про гру важливим елементом екосистеми онлайн-крамниці, що забезпечує користувачів повною та актуальною інформацією для прийняття рішень про покупку.

3.3 Вирішення програмних рішень Front-end

Першочергово, користувача зустрічає головна сторінка вебзастосунку, яка відіграє ключову роль у привернені його уваги. На головній сторінці розміщуються різноманітні блоки та рекомендації, які стимулюють користувача до активних дій. Один із таких блоків – «Топ 10 ігор по знижках» (рис. 3.3). Цей блок привертає увагу користувача, оскільки пропонує переглянути найбільш вигідні пропозиції на ринку.

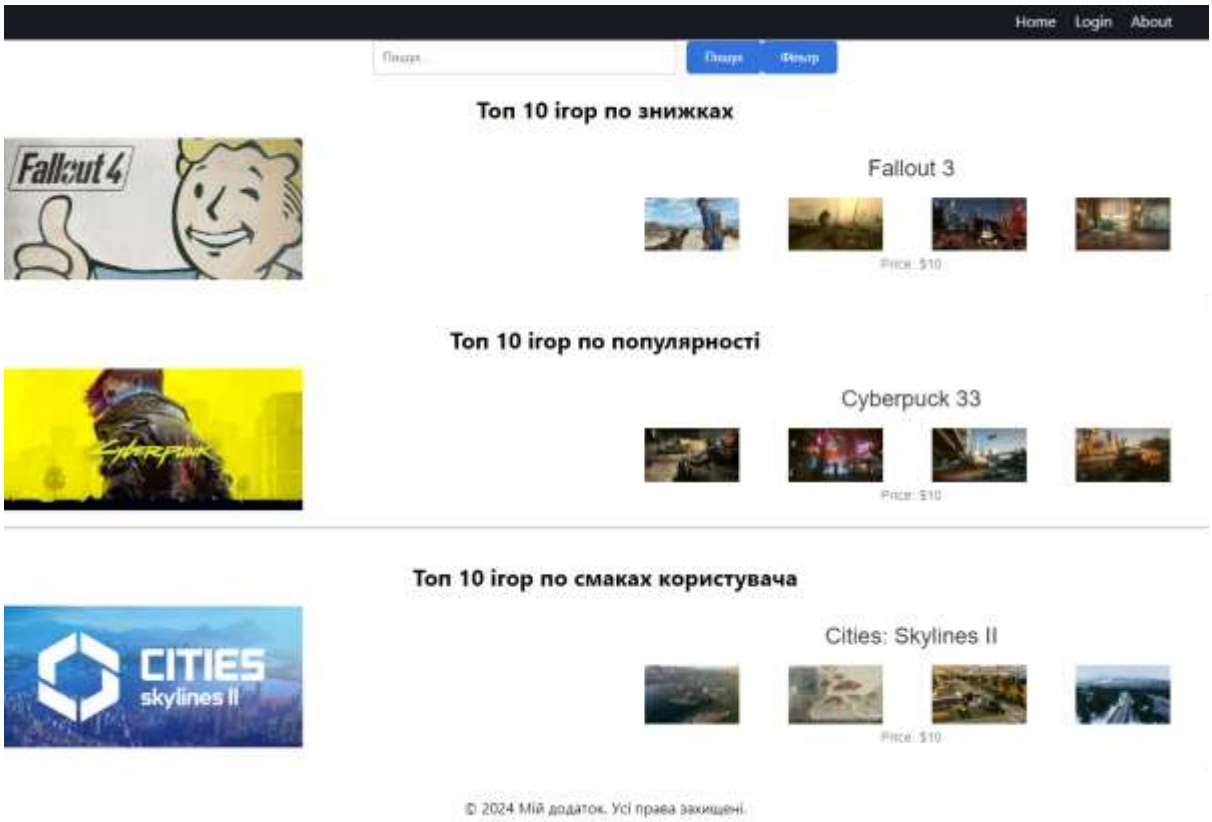


Рисунок 3.3 – Приклад головної сторінки сайту

Цей блок заснований на даних про найбільші знижки або акції на ігри, які пропонуються на платформі. Використання Swiper дає змогу створити карусель, де користувач може швидко прогортати найкращі пропозиції та знижки. Наприклад, в такій каруселі кожна карточка містить назву гри, її ціну, декілька фотографій та короткий опис. Користувач може переглянути цікаву гру, отримати інформацію про неї та зробити вибір за допомогою вбудованих посилань для переходу на сторінку з детальною інформацією або для покупки.

Крім того, на головній сторінці може бути представлений блок «Топ 10 ігор за популярністю», який базується на рейтингу користувачів або кількості продажів. Цей блок дозволяє користувачам дізнатися про найпопулярніші ігри на платформі та зробити свій вибір серед них.

Також користувачі завжди зацікавлені в отриманні рекомендацій, що відповідають їх вподобанням. Тому ще одним важливим блоком на головній сторінці є «Топ 10 ігор по смаках користувача». Цей блок використовує алгоритми рекомендацій на основі попередніх покупок, переглядів або оцінок користувача, щоб запропонувати гравцям ігри, які найбільш відповідають їхнім інтересам.

З домашньої сторінки зразу же доступний пошук (рис. 3.4).

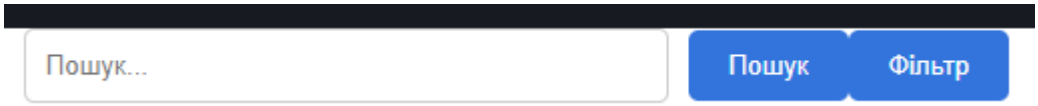


Рисунок 3.4 – Пошук на головні сторінці

Пошук в онлайн крамниці відіграє ключову роль у забезпеченні користувачам зручного та швидкого доступу до бажаних продуктів. Щоб забезпечити ефективність та зручність пошуку, використовується розумний пошук, який базується на різноманітних критеріях, таких як назва гри, розробник, жанр, рейтинг та інші.

Наприклад, якщо користувач вводить запит «Yaku» у поле пошуку, розумний пошук може автоматично врахувати можливі орфографічні помилки

або запропонувати варіанти запиту, щоб забезпечити точність результатів. Пошук також може враховувати схожість запиту з наявними назвами гри, щоб забезпечити повний охоплення результатів.

У застосунок до базового пошуку, користувачам надані фільтри, які дозволяють точніше налаштувати результати. Наприклад, можна встановити фільтр за жанром, який дозволить користувачам швидко знайти ігри в певній категорії, такі як «Action». Фільтри можуть також включати критерії, такі як платформа, рейтинг, ціновий діапазон тощо (рис. 3.5).

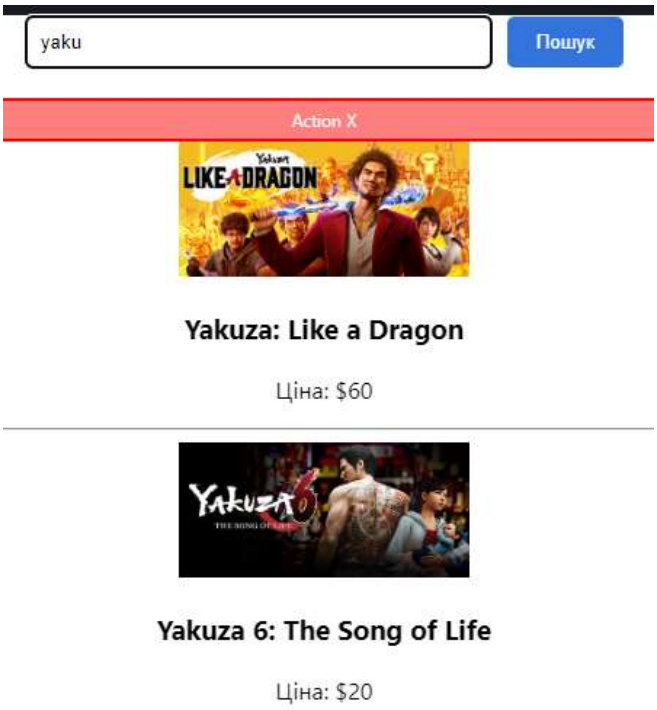


Рисунок 3.5 – Пошук на головні сторінці

Розумний пошук та фільтри допомагають користувачам ефективно орієнтуватися в великому обсязі ігор та знаходити ті, які найбільше відповідають їхнім потребам та вподобанням. Це підвищує задоволення користувачів від використання платформи та сприяє зростанню продажів, що безумовно важливо для онлайн крамниці.

Після пошуку зразу же переходимо на сторінку гри. Де буде детально викладена вся інформація про гру. Зразу нас зустрічає Велика галерея, яка реалізована за допомогою Swiper, забезпечує користувачам можливість

перегляду зображень та відео гри. Це дозволяє користувачам краще оцінити графічні особливості та геймплей, що є важливим для прийняття рішення про покупку.

На сторінці гри також міститься короткий опис гри, який дає загальне уявлення про її суть та основні особливості. Є інформація про видавця та розробника та жанри гри. Крім того, відразу видно ціну гри та дату випуску, що допомагає користувачам приймати швидше рішення про покупку (рис. 3.6).

Рисунок 3.6 – Пошук на головні сторінці

Присутність кнопок купівлі та додавання гри до списку бажаного робить процес придбання максимально зручним і простим. Користувачі можуть швидко переходити до оплати чи додавати гру до списку бажаного без зайвих турбот та перешкод.

Опис гри на сторінці відображається як важлива частина інформації, вона займає левову частку місця на сторінці гри. Ці дані допомагають користувачам отримати повний обсяг знань про обрану гру. Цей розділ містить детальні відомості про сюжет, геймплей та унікальні особливості гри.

Користувачі можуть отримати важливу інформацію про ігровий процес, механіку та функціональні можливості, щоб краще зрозуміти, чи вони зацікавлені у грі.

Також на сторінці гри відображається наявність гри у списку друзів користувача чи в їх списку бажаного (рис. 3.7 (а)). Це дозволяє спілкуватися з друзями про обрану гру, обмінюватися враженнями та рекомендаціями, що підвищує соціальну взаємодію між гравцями.

Щодо рейтингу гри, він може складатися з декількох складових, таких як оцінка магазину, відгуки друзів та віковий рейтинг (рис. 3.7 (б)). Оцінка магазину відображає загальний рейтинг гри серед користувачів магазину, відгуки друзів вказують на думку близьких осіб, а віковий рейтинг дозволяє батькам чи опікунам зробити інформований вибір, чи підходить гра для їхніх дітей (рис. 3.7 (в)). Ця інформація допомагає користувачам зробити обґрунтоване рішення про придбання гри.

(а) (б) (в)

Рисунок 3.7 – Пошук на головні сторінці:
(а) наявність гри в друзів; (б) оцінки; (в) віковий рейтинг

Рецензії є важлива частина інформації на сторінці гри, яка дозволяє користувачам отримати суб’єктивну думку про якість та особливості гри від інших гравців. Рецензії можуть містити різноманітну інформацію, таку як опис геймплею, графіка, сюжету, а також переваги та недоліки гри (рис. 3.8).

Додавання рецензій на сторінці гри створює можливість спілкування між гравцями та обміну враженнями та думками про гру. Це допомагає іншим користувачам отримати більш повний огляд гри.

Рисунок 3.8 – Рецензія від покупця

Користувачі також можуть залишати коментарі та оцінки під рецензіями, що дозволяє розширити обговорення та надати додаткову інформацію про думку спільноти щодо гри.

Також на сторінці зазначенні системні вимоги, щоби гравців годні були зрозуміти чи підходе їхній персональний комп'ютер під гру (рис. 3.9).

Рисунок 3.9 – Рецензія від покупця

На сторінці логіну користувачеві відкривається можливість увійти до свого особистого облікового запису. Це важлива процедура, що дозволяє впізнати та аутентифікувати користувача перед наданням доступу до ширшого функціоналу крамниці, аніж просто перегляд інформації про ігри. На цій сторінці розташована форма, в яку користувач повинен ввести свої дані, а саме логін та пароль.

Наявність трьох кнопок визначає зручність та функціональність цієї сторінки. Перша кнопка, «Увійти», дозволяє користувачеві підтвердити свою ідентичність та отримати доступ до особистого облікового запису, коли він

вже ввів свої дані. Якщо користувач ще не має облікового запису, він може натиснути кнопку «Створити акаунт», що перенаправить його на сторінку реєстрації. Якщо користувач забув свій пароль, він може скористатися третьою кнопкою, «Забули пароль?», яка відправить його на сторінку відновлення паролю, де він зможе відновити доступ до свого облікового запису (рис. 3.10).

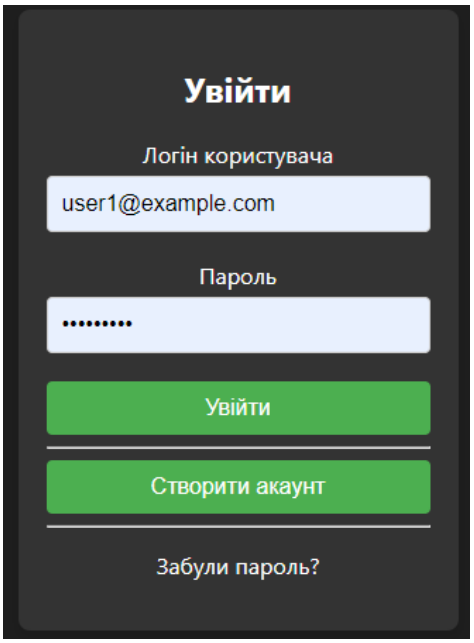
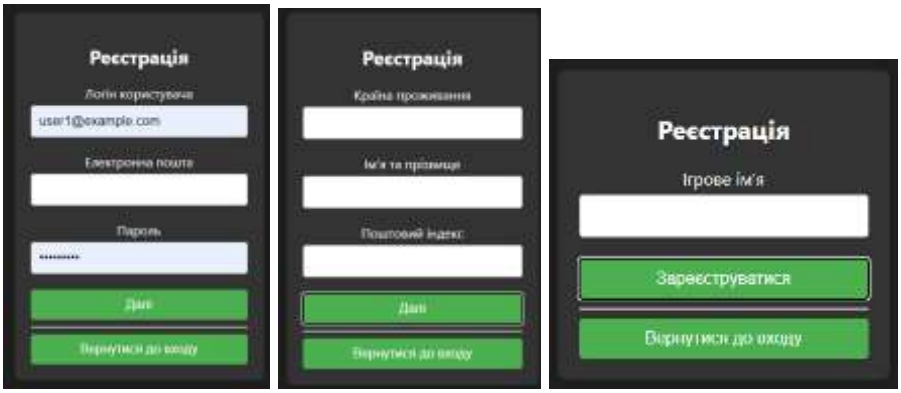


Рисунок 3.10 – Приклад форми логіна

В свою чергу реєстрація відбувається покроково (рис. 3.11). На першому етапі реєстрації користувачу пропонується ввести основні дані для створення облікового запису. Він вводить свій логін, електронну пошту та обирає надійний пароль для майбутнього входу на сайт (рис. 3.11 (а)).

Після успішного завершення першого етапу, користувач переходить до наступного кроку, де йому пропонується ввести додаткові особисті дані. Тут він вказує країну проживання, поштовий індекс та повне ім’я (рис. 3.11 (б)).

Останнім етапом процесу реєстрації є вибір ігрового назвиська, який буде використовуватися для ігрової діяльності на сайті. Користувач обирає унікальне ім’я, що буде ідентифікувати його в онлайн іграх та взаємодії з іншими користувачами (рис. 3.11 (в)).



(а) (б) (в)

Рисунок 3.11 – По крокова реєстрація:
(а) логін, пошта та пароль; (б) країна, ПІБ та індекс; (в) ім’я на сайті

Звісно після успішних перевірок на кожному кроці [30], користувач годен створити свій обліковий запис в вебзастосунку та отримує повний функціонал крамниці.

Сторінка користувача то особистий простір, де кожен зареєстрований користувач може переглянути та керувати своїми особистими даними та діями на сайті. На першій частині (рис. 3.12) сторінки розміщена загальна інформація про користувача, така як ім’я, адреса електронної пошти, дата реєстрації тощо. Додатково можуть бути відображені деталі про країну проживання, вік та інші дані.

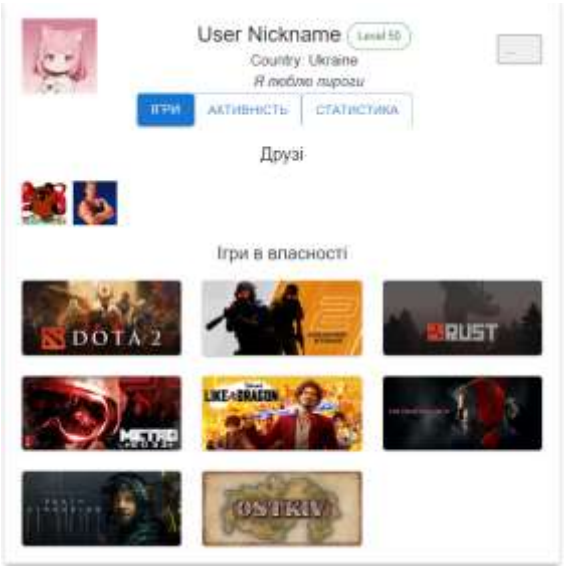


Рисунок 3.12 – Сторінка користувача, коли обраний вигляд ігри

Другій компоненти сторінки (рис. 3.13) можна зобачити натиснувши кнопку активність, там показана історія дій користувача на сайті, такі як нові друзі чи ігри, оцінки ігор, залишені рецензії тощо. Це допомагає користувачеві або його друзям відстежувати його активність та спілкування з іншими користувачами.

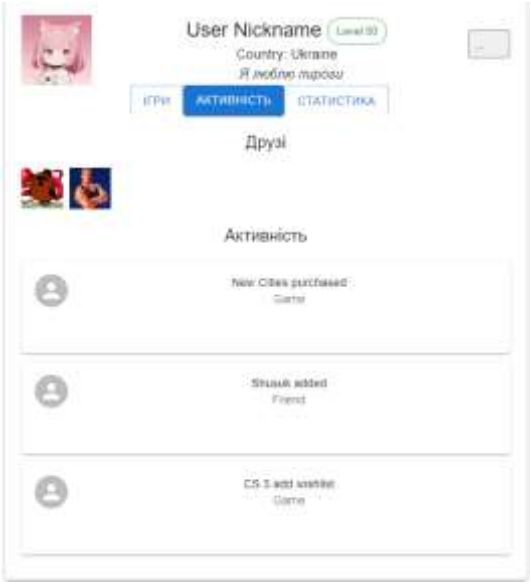


Рисунок 3.13 – Сторінка користувача, коли обраний вигляд активність

Третій компонент на сторінці користувача (рис. 3.14) – це розділ, який відображає статистику про улюблені жанри та франшизи користувача. В цьому розділі показано декілька важливих моментів, які можуть бути цікаві для користувача та його друзів.

В цьому компоненті показується загальна кількість ігор у власності користувача, а також кількість позитивних та негативних рецензій, що він залишив на сайті. Це дозволяє користувачеві швидко переглянути свою активність та ставлення до ігор. Крім того, у цьому розділі вказується улюблений жанр та франшиза користувача. Ця інформація допомагає користувачеві краще розуміти свої ігрові вподобання та може бути корисною при виборі нових ігор.

Найцікавішою частиною цього компоненту є два графіки, які відображають розподіл ігор за жанрами та франшизами. У кожному графіку

показано «пиріг», де кожен сегмент представляє певний жанр або франшизу, а його розмір відповідає кількості ігор у цьому жанрі або франшизі. Такий візуальний підхід дозволяє швидко оцінити основні смаки гравця. Крім того, ці графіки надають можливість детально аналізувати власну ігрову колекцію та виявляти переваги в різних категоріях.

Рисунок 3.14 – Сторінка користувача, коли обраний вигляд статистика

Також існує функціонал додавання в друзі, він дозволяє спростити процес спілкування та взаємодії між користувачами. На сторінці користувача є доступна кнопка, яка додає в друзі. Ця функція сприяє формуванню спільноти та обміну ігровим досвідом серед користувачів платформи.

Коли користувач натискає на кнопку «Додати в друзі», відбувається відправлення запиту на додавання в друзі до відповідного користувача. Після відправлення запиту, відправник очікує підтвердження від одержувача. Якщо користувач, до якого надійшов запит, погоджується на дружбу, то вони автоматично стають друзями.

Також існують функції перегляду списку друзів, видалення користувача зі списку друзів.

На сторінці видавця (рис. 3.15) презентується коротка, але змістовна інформація про даного видавця. Перш за все, вказується назва видавця, щоб користувач міг легко ідентифікувати його. Далі, ми надаємо кількість ігор, які були видані цим видавцем, щоб дати уявлення про його активність та діяльність на ринку. Також важливою інформацією є кількість покупців, які вже придбали ігри цього видавця, що може служити показником його популярності та впливу.

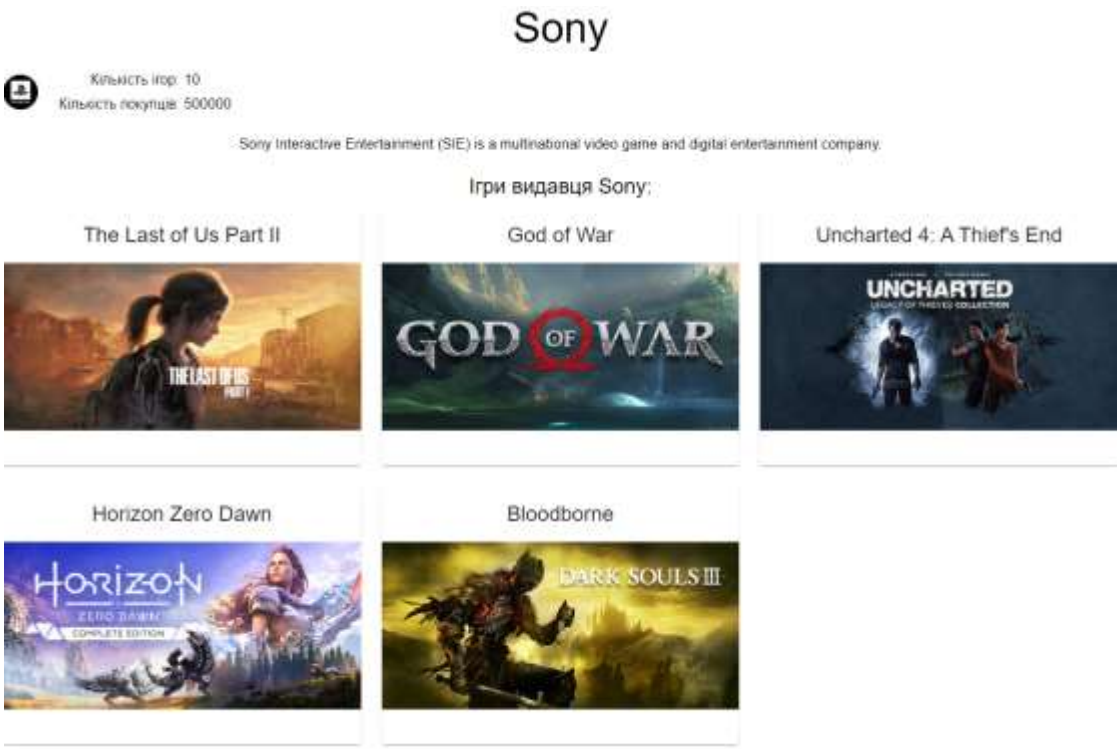


Рисунок 3.15 – Сторінка видавця

Окремою частиною сторінки є перелік ігор, які були видані даною компанією. Кожна гра супроводжується назвою, щоб користувач міг швидко зрозуміти, що цікавить його. Це допомагає створити повну картину про продукцію даного видавця та дозволяє користувачам легко знайти ігри, які вони можуть зацікавитися.

ВИСНОВКИ

У рамках даної кваліфікаційної роботи був розроблений вебзастосунок для купівлі комп'ютерних ігор. Для досягнення цієї мети були розглянуті та проаналізовані методи роботи з великої кількістю даних, аналіз даних.

Виконано всі поставлені задачі, а саме:

– проведений аналіз ситуації на ринку цифрових копій комп'ютерних ігор, вже існуючих вебсервісів, а також досліджено їхні переваги та недоліки;

– реалізовано весь заявлений функціонал. Користувачі можуть переглядати доступний каталог ігор та придбати бажані позиції. Реалізовано систему рекомендацій, яка аналізує попередні покупки користувача та пропонує подібні ігри для покупки. Алгоритми підрахунку статистичних даних надають користувачам інформацію про популярність та рейтинг ігор. Створено зручний та інтуїтивний інтерфейс, що дозволяє користувачам легко знаходити потрібні ігри та здійснювати покупки. Реалізована система пошуку та фільтрів, що допомагає користувачам швидко знаходити ігри за різними параметрами. Впроваджено систему знижок та розпродажів для приваблення користувачів. Розроблено бібліотеку ігор, де користувачі можуть зберігати свої придбані та вибрані ігри. Створено особисті сторінки користувачів та сторінки ігор, де можна знайти додаткову інформацію та опис продуктів;

– побудовані мікросервісна архітектура для швидкодії та маштабованості проєкта. CRUD для наповнення продуктами крамниці. Надійна система безпеки та система рецензій від користувачів про відеогру;

– сторінки ігор має цілковите наповнення інформацією про бавку, яка потрібна користувачам.

Результати роботи апробовано у вигляді 1 тези доповідей під час 28-го Міжнародного молодіжного форуму «РАДІОЕЛЕКТРОНІКА ТА МОЛОДЬ У XXI СТОЛІТТІ» [31].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Statista: video game revenue worldwide 2022-2032, by platform / J. Clement. – 2024. URL: https://www.statista.com/statistics/379682/games-revenue-category/ (дата звернення 15.05.2024).

2. C. Digital and physical game sales in the U.S. 2009-2018, by format / J. Clement. – 2022. URL: https://www.statista.com/statistics/190225/digital-and-physical-game-sales-in-the-us-since-2009/ (дата звернення 15.05.2024).

3. Bethesda.net – digital video game store. URL: https://help.bethesda.net/#en/answer/35008 (дата звернення 15.05.2024).

4. Official Ubisoft Store – game store by Ubisoft. URL: https://store.ubisoft.com/ie/home?lang=uk-UA (дата звернення 15.05.2024).

5. EA Play – game store by Electronic Arts Inc. URL: https://www.ea.com/sales/deals?setLocale=en-us (дата звернення 15.05.2024).

6. ITC.ua: Нові рекорди Steam. URL: https://itc.ua/ua/novini/novi-rekordy-steam-ponad-33-mln-korystuvachiv-onlajn-ta-ponad-10-mln-aktyvnyh-gravtsiv/ (дата звернення 15.05.2024).

7. SteamDB. URL: https://steamdb.info/ (дата звернення 15.05.2024).

8. Steam – digital store by Valve Inc. URL: https://store.steampowered.com/ (дата звернення 15.05.2024).

9. Epic Games Store. URL: https://store.epicgames.com/en-US/?lang=en-US (дата звернення 15.05.2024).

10. GOG. URL: https://www.gog.com/en/ (дата звернення 15.05.2024).

11. Гороховатський, В. О., & Творошенко, І. С. (2022). Аналіз багатовимірних даних за описом у формі множини компонент.

12. Ibrahim Daradkeh, Y., Gorokhovatskyi, V., Tvoroshenko, I., & Al-Dhaifallah, M. (2022). Classification of Images Based on a System of Hierarchical Features. Computers, Materials & Continua, 72(1).

13. Танянський, О., & Руденко, Д. (2018). Порівняльний аналіз популярних JavaScript-фреймворків та бібліотек для front-end розробки.

14. McConnell, S. (2004). Code Complete. Microsoft Press.

15. Golang Goroutines. Google Inc. URL: https://go.dev/tour/concurrency/1 (дата звернення 15.05.2024).

16. Гороховатський, В. О., & Творошенко, І. С. (2021). Методи інтелектуального аналізу та оброблення даних: навч. посібник.

17. Yevstratov, M., Lyubchenko, V., Amer, A. J., & Lyashenko, V. (2024). Color correction of the input image as an element of improving the quality of its visualization.

18. Kinoshenko, D., Mashtalir, V., & Yegorova, E. (2010). Block-Diagonal Forms of Distance Matrices for Partition Based Image Retrieval. In Pattern Recognition Recent Advances. IntechOpen.

19. Гороховатський, В., & Метелев, В. (2021). Редукція структурного опису зображення на основі критерію інформативності. EDITORIAL BOARD, 424.

20. Gorokhovatskyi, V. O., Tvoroshenko, I. S., & Peredrii, O. O. (2020). Image classification method modification based on model of logic processing of bit description weights vector. Telecommunications and Radio Engineering, 79(1).

21. Tvoroshenko, I., & Gorokhovatskyi, V. (2022). The Application of Hybrid Intelligence Systems for Dynamic Data Analysis.

22. Шафроненко, А., Бодянський, Є., & Плісс, І. (2022). Нечіткі методи інтелектуального аналізу даних.

23. Ситніков, Д. Е., Ситнікова, П. Е., Тітов, С. В., & Тітова, О. В. (2021). Фільтрація результуючого набору асоціативних правил з точки зору оцінки цікавості. Системи обробки інформації, (1 (164)), 83-88.

24. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

25. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2023). Statistical data analysis models for determining the relevance of structural image descriptions. IEEE Access, 11, 126938-126949.

26. Chupikov, A., Kinoshenko, D., Mashtalir, V., & Shcherbinin, K. (2006, July). Image retrieval with segmentation-based query. In International Workshop on Adaptive Multimedia Retrieval (pp. 207-221). Berlin, Heidelberg: Springer Berlin Heidelberg.

27. Tvoroshenko, I., Pomazan, V., Gorokhovatskyi, V., & Kobylin, O. (2023). Application of video data classification models using convolutional neural networks.

28. Kinoshenko, D., Mashtalir, S., Stephan, A., & Vinarski, V. (1993). Neural Network Segmentation Of Video Via Time Series Analysis. INFORMATION THEORIES & APPLICATIONS, 232.

29. Машталір, С. В. (2024). Video fragment processing by Ky Fan norm. Прикладні аспекти інформаційних технологій, 7(1), 59-68.

30. Iryna, T., & Heorhii, M. (2021). TO THE QUESTION OF ANALYSIS OF EXISTING MECHANISMS OF WEB APPLICATION TESTING. Problems of modern science and practice, 1, 403.

31. Терещенко О.О., Кіношенко Д.К. (2024). АНАЛІЗ ПРОДАЖІВ ДЛЯ ВИЗНАЧЕННЯ ПОПУЛЯРНИХ ІГОР НА РИНКУ. 28-ий міжнародний молодіжний форум «РАДІОЕЛЕКТРОНІКА І МОЛОДЬ У XXI СТОЛІТТІ».