Приватний заклад вищої освіти

Одеський технологічний університет «ШАГ»

Кафедра інформаційних технологій та фундаментальної підготовки

Випускна кваліфікаційна роботи бакалавра

«Інформаційна система поширення та актуалізації програмних ресурсів (за прикладом сервісу Steam)»

на здобуття бакалаврського рівня вищої освіти

зі спеціальності «122 Комп'ютерні науки»

Виконавці проекту

№	П.І.Б.	Група
1	Лінчевський Борислав Ігорович	КН-П-201
2	Тарасенко Степан Сергійович	КН-П-201
3	Павлова Анастасія Олексіївна	КН-А-201
4	Ринкова Ніка Владиславівна	КН-Д-201
5	Олєйнік Катерина Андріївна	КН-Д-202

Автор звіту

_____ Лінчевський Борислав Ігорович

Одеса – 2024

АНОТАЦІЯ

УДК 004.9

Д-30
Лінчевський Б. І. Платформа онлайн-маркетплейсу ігрових додатків: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності “122 Комп’ютерні науки”. - Одеса: ОТУШ, 2024 -13 с.

Робота присвячена аналізу, розробці та тестуванню онлайн-маркетплейсу ігрових додатків. Головна функцією цієї платформи полягає в наданні користувачам можливості купувати продукти, розміщені на маркетплейсі їхніми власниками, з можливістю для кожного користувача виступати як у ролі клієнта, так і у ролі власника продукту. Другорядною функцією онлайн-платформи є надання можливості користувачам додавати інших клієнтів у список "друзів", об'єднуватися в спільноти та обмінюватися повідомленнями за допомогою чатів. Платформа поділяється на велику кількість частин, кожна з яких має свою функціональність. Основними функціями цих частин є:

Авторизація
Автентифікація
Реєстрація
Додавання продуктів
Додавання користувачів у “друзі”
Об’єднання у спільноти
Додавання категорій для продуктів
Додавання медіафайлів
Додавання коментарів до медіафайлів
Додавання коментарів до сторінки користувача
Додавання продуктів у список “бажаних”
Додавання продуктів до користувацьких колекцій

Релевантність проекту аналізується через реалізацію актуальних функцій, які реалізовані розробниками задля спрощення процесу розміщення продуктів на маркетплейсі та придбання цих продуктів користувачами. Зручність у використанні сервісу також у способі реєстрації через підтвердження акаунту через повідомлення на пошту.

Найбільшу кількість уваги було приділено організації архітектури бази даних та організації пошуку продуктів та додавання їх в користувацькі колекції. Це дозволяє максимально швидко знаходити продукти на сервісі та додавати куплені продукти в користувацькі колекції, що також полегшує організацію цих продуктів.

Актуальність сервісу також ґрунтується на впровадженні надійної системи безпеки для захисту особистих персональних даних. В цю систему входить шифрування вхідних та вихідних персональних даних, таких як пароль, також впровадження системи JWT та покладання його в cookie сайту, а сам JWT існує тільки одну годину, після вичерпання котрої цей JWT треба перегенерувати. Сам JWT шифрується за алгоритмом HmacSha256. Пароль шифрується за алгоритмом SHA384. Також впроваджена система ідентифікації елементів в базі даних за допомогою Guid, що дає змогу уникнути небажаних запитів з використанням SQL-ін’єкцій.

Об’єктом дослідження у рамках даної роботи виступає алгоритм пошуку продуктів з використанням категорій, котрі були вказані в продукті розробником, що спрощує пошук елементів на маркетплейсі. У якості задач дослідження було встановлено наступне:

- проаналізувати предметну область для організації архітектури бази даних на базі MySQL
- проаналізувати різні варіанти реалізації взаємодії клієнтської та серверної частини за допомогою Single Page Application
- розробити програмний інтерфейс (API), впровадивши безпечне отримання вхідних даних тільки від авторизованого користувача, використовувавши JWT за допомогою протокола HTTP
- розробити алгоритм отримання запитів з клієнтської частини на серверну та надсилання відповідей з серверної частини на клієнтську
- реалізувати взаємодію серверної частини з базою даних, використовуючи Entity Framework та додавання міграцій(Migrations) для внесення змін до архітектури бази даних
- впровадити в серверну частину засоби захисту для забезпечення цілісності користувацьких даних

Методи дослідження, що використані для досягнення поставлених задач, утворені за допомогою методів тестування програмного забезпечення,

гнучких методологій розробки, моделювання архітектури за допомогою принципів SOLID та теорії баз даних.

Окрему уваги треба звернути на те, що командна робота була організована за допомогою Agile-методологій, це дозволило нам гнучко розподілити обов’язки між членами команди та найбільш ефективно реалізовувати поставлені задачі, та мати постійний зворотній зв’язок з менторами для того, щоб редагувати код та виправляти помилки у виконаній роботі. Щоб використовувати цю методологію, наша команда використовувала багатоплатформну систему управління проектами Trello, що дозволило нам організувати та впровадити Agile методології, зробити роботу команди більш організованою, контрольованою та максимально ефективною. Також у процесі розробки back-end та front-end розробники користувались методологією екстремального програмування, що дозволило швидше впроваджувати певний функціонал, котрий вимагав додаткової реалізації. Для прискорення розробки також використовували методологію CI\CD, що дозволило всім учасникам команди реалізувати непреривну інтеграцію, пришвидшити пошук дефектів, підвищити продуктивність розробки та найбільш швидко випускати білди проекту.

Також для розробки програмного забезпечення команда розробників використовувала систему контролю версій Git, та інструмент, котрий її реалізує - GitLab. Це забезпечило нам більш гнучкий обмін актуальними версіями проекту і дало можливість скасовувати внесені зміни у разі необхідності. Також це підвищило безпеку та цілісність проекту, так як доступ до GitLab можливий виключно з використанням стороннього VPN “Tailscale” з особистою конфігурацією.

При складанні звіту використано 12 посилань на електронні джерела інформації

ЗМІСТ

ВСТУП

Сучасне життя цілком та повністю складається з інформаційних технологій, через що все більше та більше людей використовують онлайн сервіси в повсякденному житті, в роботі та у відпочинку. З кожним днем з’являється величезна кількість онлайн-сервісів з наданням різноманітних послуг. Спираючись на стратегію розвитку інформаційних технологій Міністерства цифрової трансформації України, наша команда створила онлайн-маркетплейс, який представляє собою цифровий інтернет-магазин для купування ігрових додатків.

Всі члени команди вважають, що інвестиції у технології, які мають надавати людям більш спрощені для користувачів варіанти придбання продуктів, а для власників інтуїтивно зрозумілий інтерфейс та інструментарій для розміщення своїх особистих продуктів на онлайн-маркетплейсах. Через це наш сервіс пропонує велику кількість способів полегшення купівлі та продажу ігрових додатків без клопоту.

Також платформа допомагає виробникам просувати свої продукти більш гнучко, використовуючи категорії, котрими можуть скористатись користувачі маркетплейсу для знаходження товарів, котрі співпадають з їхніми пошуковими запитами.

Далі у цьому документі розглянуто аналіз предметної області, проектування архітектури проекту та бази даних, розробку проекту, тестування та майбутню підтримку онлайн-маркетплейсу з купівлі та продажу ігрових додатків.

У наступному розділі під назвою “Технічне завдання до проекту” описане вичерпне технічне завдання. У рамках даної роботи розглядається серверна частина платформи, архітектура бази даних, а також елементи клієнтської частини Web-застосунку. Висновки показують результати, які були отримані при тестуванні всіх частин проекту. Загальні практичні та логічні підсумки є сукупністю звітів всіх розробників проекту.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЕКТУ

Інформаційна система поширення та актуалізації програмних ресурсів (за прикладом сервісу Steam)

Загальний опис.
Програмне забезпечення (ПЗ) призначене для створення мережевої системи яка дозволяє шукати, встановлювати, оновлювати та оцінювати програмні ресурси. Функції системи мають забезпечити повний життєвий цикл управління контентом: створення, актуалізація, модифікація та знищення, а також встановлення скорингових параметрів (оцінок) наявного контенту. Головна особливість системи полягає у тому, що наявні у неї інформаційні одиниці мають бути захищені авторським правом у відповідності до умов, що їх висуває власник ПЗ.

Призначення:
ПЗ орієнтується на створення онлайн системи управління інформаційними одиницями на основі аналізу їх поточної версії. Визначальною функцією системи є можливість встановлення обраного ПЗ та контролю його актуального стану. ПЗ створюється за прикладом сервісу Steam, у той же час має складатись за принципом універсальності, мати змогу змінити цільове призначення.

Вимоги до функціоналу. Обов'язкова частина
Головна сторінка
Стартовий інтерфейс ПЗ (головна сторінка або активність) має за мету популяризацію системи. На ньому має бути забезпечено відображення популярних показників як-то загальна кількість користувачів та інформаційних одиниць, вибірка з найбільш популярного контенту та нових дописів, зовнішня реклама. Значний акцент має бути зроблений на детальному пошуку контенту за принципом таргетування. Головна сторінка обмежує функції системи для незареєстрованих користувачів лише переглядом та пошуком, інші функції надаються у разі авторизації користувача.

Реєстрація користувачів:
Користувач ПЗ має змогу зареєструватись як за допомогою авторизаційних даних (логіну/телефону/E-mail/паролю), так і за

допомогою облікових записів популярних соціальних мереж (Google, Facebook, тощо).
При реєстрації обов'язково зазначається засіб оперативного контактування (ЗОК) - телефон та електронна пошта. На тестовому етапі, за умови ускладнення замовлення послуг телефонії чи СМС, можливе використання лише електронної пошти. Засіб оперативного контактування при реєстрації підлягає верифікації через відповідь на тестовий виклик чи надісланий лист, або введення коду, пересланого через повідомлення довільного типу. Рекомендується використовувати ЗОК у якості авторизаційного логіну користувача.
За умови що користувач забув пароль, ЗОК має бути використаний для його відновлення. Схема відновлення паролю жорстко не регламентується, рекомендовано вживання тимчасових паролів (one time passwords, OTP), але допускаються й інші схеми.
Також передбачається наявність у системі окремих користувачів з додатковими правами: модераторів та адміністраторів. Реєстрація кореневого адміністратора системи має бути реалізоване при інсталяції системи, наступним користувачам змінюють права наявні адміністратори.
Користувачі системи поділяються на надавачів та споживачів послуг. При реєстрації надавача послуги вимагається додаткова перевірка його прав та ліцензій на можливість надання послуг (розміщення ПЗ) чи ведення електронної комерції. На тестовому етапі достатньо лише факту надсилання користувачеві запиту на відповідні документи, надісланого через ЗОК.
Для користувачів усіх категорій вимагається формальна валідація усіх реєстраційних даних (відповідність загальновживаним або власним шаблонам).

Особисті кабінети користувачів:
Особисті кабінети різних груп користувачів - надавачів та споживачів послуг - мають забезпечувати різну функціональність. Дозволяється на початковому етапі (до впровадження механізмів перевірки ліцензій надавачів послуг) функції надавача послуг делегувати користувачам з підвищеним статусом керування (адміністраторам) через що кабінет надавача послуги може виконуватись як панель адміністрування контента.

Особистий кабінет надавача послуги має забезпечити необхідні засоби для

- Управління даними та метаданими (пропозицій та їх описів, цільових груп, тощо)
- Створення, редагування, видалення (блокування), перегляд розміщених пропозицій.
- Перегляд поточної активності щодо власних пропозицій (коментарів, оцінок, історії завантажень)
- Контактування з користувачами-споживачами, що розміщують замовлення на контент, за допомогою ЗОК або через внутрішні засоби системи (чат або мессенджер)

Особистий кабінет споживача послуги має забезпечити необхідні засоби для

- Встановлення належності до цільових груп (вибір із заданого переліку)
- Пошуку контенту інших користувачів за метаданими, групування та фільтрування результатів за різними ознаками, наявними у знайденому контенті, а також за цільовою групою. Для об'ємних результатів має бути забезпечена пагінація.
- Контактування з іншими користувачами-надавачами послуг, що розміщують пропозиції, за допомогою ЗОК або через внутрішні засоби системи (чат або мессенджер)

Також засобами особистих кабінетів має бути передбачено можливість

- Зміни персональних даних, зокрема, контактних засобів. У разі зміни важливих даних, що раніше проходили верифікацію, вимагається їх повторна верифікація.
- Зворотного зв'язку - встановлення власної оцінки (лайк або бал) для спожитої або наданої послуги, написання відгуку (коментаря) або рекомендації за фактом взаємодії.

Вимоги до функціоналу. Опціональна частина

Модерація

Для забезпечення дотримання загальноприйнятих норм спілкування, а також з метою недопущення публікування контенту, який містить заборонені компоненти (ненормативні висловлювання, заклики до протиправних дії, приниження чи дискримінація, тощо), усі інформаційні одиниці, а також створені користувачами метадані

(маркери) мають в обов'язковому порядку пройти погодження (модерацію). Право надати погодження належить кореневому адміністратору системи, а також модераторам. При погодженні інформаційна одиниця має зберігати ідентифікатор користувача, який надав погодження.

Привілейовані функції
З метою покращення економічної ефективності проєкту передбачається наявність додаткових функцій системи, що доступні лише за умови платної підписки (або внутрішніх досягнень) користувачів. До таких функцій можна віднести окремі групи контенту, що є доступними тільки для привілейованих користувачів, відображення ЗОК інших користувачів (для прямого контактування), розширені засоби фільтрації та групування. Також можливе встановлення обмеження щодо кількості опублікованих одиниць контенту або замовлень.

Архітектура
ПЗ має бути спроектоване за клієнт-сервеною розподіленою архітектурою з чітко відокремленими структурними вузлами системи:
Mockup - візуальна модель інтерфейсу користувача із зазначенням зв'язків і переходів між різними частинами інтерфейсу.
Backend - серверне програмне забезпечення, що забезпечує збереження та оброблення поточних даних, взаємодію з базою даних. Надає програмний інтерфейс взаємодії (API) з іншими вузлами.
Web App - сайт або застосунок, доступний з веб-простору, що забезпечує візуальний інтерфейс користувачів усіх категорій
Mobile App (опціонально) - застосунок, що встановлюється на мобільних пристроях (смартфон, планшет). Дана частина виконується за наявності у команді проєкту трьох і більше програмістів.

Усі модулі ПЗ (окрім візуальної моделі Mockup) повинні обов'язково включати до свого складу засоби

- тестування функціональності та модульні тести,
- розгортання на новому пристрої (інсталяції) із встановленням початкових таблиць у БД та кореневого користувача-адміністратора

Технології

Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контроля версій (вибір конкретної системи узгоджується групою проекту). Структура файлів та каталогів повинна відбивати архітектуру ПЗ, на кшталт:

- root
    - Backend
    - Web
    - Mobile
    - Mockup (вихідні файли або посилання на зовнішній проєкт)

Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології створення:

Backend - .NET або Java, або NodeJS технології з хмарним розміщенням (Azure, AWS, Google Cloud),

Web - серверні технології (ASP, Razor Pages, JSP) або React чи Angular

Mobile - Java, або Kotlin, або Flutter, або React Native

Mockup - визначається ментором з дизайну

Команда проєкту

Для виконання поставлених завдань передбачається робота команди розробників окремо за кожним з модулів проєкту.

Мінімальна команда проєкту - два учасники, у такому складі один учасник виконує модуль Backend та програмну частину Web, другий - модель Mockup та розміткову частину Web.

Рекомендована команда проєкту - три учасники, у такій команді кожен з учасників відповідає за одну з частин: Mockup, Backend, Web.

Дозволяється включення до команди більшої кількості учасників, у такому разі модуль Mobile App набуває обов'язкової реалізації. Також модуль Mockup може бути поділений на моделі Web та Mobile частин.

Звітність

За результатами виконання завдань із створення ПЗ кожен з учасників команди проєкту складає власний (індивідуальний) звіт, що складається з практичної частини та автореферату. Практична частина є прямим результатом роботи і подається у вигляді посилання на репозиторій (чи інший URL) або у вигляді архіву на довільному

носієві даних. Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU http://www.gnu.org/prep/standards/, у тому числі з додержанням вимог до коментування та документування коду.
Автореферат містить описову частину, у якій розкриваються основні концепти проєкту, деталізуються завдання до виконавця, встановлюється актуальність роботи, її об'єкт та предмет, обґрунтовується вибір технологій та відзначається шлях виконання та ступінь досягнення поставлених задач, посилання на джерела інформації.

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

Згідно з технічним завданням (ТЗ) нашою командою було проаналізовано всі види архітектурних рішень програмного забезпечення. Через свою продуктивність та зручність використання для користувача платформи, а також сучасністю розробки, нашою командою було обрано SPA (Single Page Application), як головним архітектурним стилем сервісу. Основа програмної архітектури складається з таких самовідокремлених частин:

- Design;
- Frontend;
- Backend;
- DevOps.

Частина “Design” представляє собою набір готових Web-сторінок та набір прототипів до Web-сторінок з урахуванням кожного елементу користувацького інтерфейсу. Учасником нашої команди, котрий відповідав за частину UI, були підібрані:

- Основна гама кольорів
- UI-kit для усіх контролів інтерфейсу

Під час реалізації кожної частини візуального інтерфейсу, були задіяні такі підходи як:

- UI\UX дизайн
- Колористика
- Теорія Web-дизайну

На деяких Web-сторінках та на навігаційній панелі були застосовані анімації з урахуванням продуктивності Web-браузера та швидкого відгуку від дій користувача для забезпечення максимально комфортного користування платформою.

Основна частина “Front-end” розробки була реалізована за допомогою бібліотеки React, основними мовами програмування котрою є:

- TypeScript
- JavaScript

А також мовами розмітки та стилей, такими як:

- HTML
- CSS

Для передачі користувальницьких даних від клієнтської частини до серверної наша команда використовувала мережеві запити за допомогою технології “fetch” з використанням протоколу HTTP.

Серверна частина “Back-end” була розроблена за допомогою технології платформи Microsoft ASP.NET Core та архітектурного стилю RESTful, основною мовою програмування котрої являється C#. Для перебійного зберігання персональних даних користувачів, виробників та даних платформи було використано My SQL Server. Проаналізувавши та вивчивши предметну область роботи, була обрана ця СУБД (система управління базами даних) через її швидкість, сумісність з підходом та архітектурою з кожною частиною проекту. Використання My SQL Server дозволило нам використовувати ORM (Object Relational Mapping) Entity Framework Core, котра розроблена Microsoft, що спростило розробку та взаємодію з базою даних через підтримку мови запитів LINQ (Language Integrated Query), що є однією з найпопулярніших та найефективніших технологій розробників ПЗ (програмного забезпечення) для створення SQL-запитів до бази даних. У якості серверу було реалізовано та впроваджено RESTful-API, який приймає мережеві запити з клієнтської частини на серверну за допомогою протоколу HTTP, оброблює вхідні дані, відправляє запити до бази даних, та відправляє відповіді на клієнт з обробленими даними за допомогою текстового формату JSON. Так як мережеві запити відправлялись з клієнтської частини, що означає, що надсилались вони з іншого домену, були налаштовані CORS (Cross-Origin Resource Sharing) політики для забезпечення безпечної та постійної взаємодії “Front-end” частини з “Back-end” частинами проекту. Також був впроваджений JWT (JSON Web Token) задля реалізації авторизації та автентифікації користувача, що забороняє надсилати запити неавтентифікованим користувачам на серверну частину проекту. У якості сервісу з встановленням зовнішніх пакетів була використана система NuGet від Microsoft. Також всі конфігураційні параметри були розподілені у текстовому форматі JSON в окремих файлах, з яких, за потребою, використовується інформація. Головним приципом проектування архітектури серверу було використання патернів проектування, а також принципів проектування класів SOLID.

Адміністративна частина “DevOps” була реалізована на базі локальних серверів, на котрих розташовувались GitLab та MySQL Server, також на локальних серверах були налаштовані pipeline’и та CI\CD, що дозволило організувати запуск runner’ів для віддаленої та незалежної перевірки та тестування клієнтської та серверної частини.

Програмна частина платформи зберігається у приватному хмарному зберіговищі репозиторії веб-сервісу GitLab за посиланням GitLab, також на публічного хмарному зберіговищі репозиторії веб-сервісу GitHub за посиланням GitHub.

РОЗДІЛ 2

РЕАЛІЗАЦІЯ СЕРВЕРНОЇ ЧАСТИНИ

Серверна частина складається з:

- RESTful-API з використанням платформи розробки ASP.NET Core від Microsoft;
- База даних в основі MySQL;
- Мови програмування C#, Transact-SQL;
- Середовища розробки: Microsoft Visual Studio, MySQL Workbench;
- Використані технології: Entity Framework Core, LINQ
- Встановлені додаткові пакети: BCrypt.Next-Next, Microsoft.AspNetCore.Authentication.JwtBearer, Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore, Microsoft.AspNetCore.Identity.EntityFrameworkCore, Microsoft.AspNetCore.Identity.UI, Microsoft.EntityFrameworkCore, Microsoft.EntityFrameworkCore.SqlServer, Microsoft.EntityFrameworkCore.Tools, Microsoft.VisualStudio.Azure.Containers.Tools.Targets.

Вся архітектура серверної частини була спроектована, використовуючи патерни та принципи проектування для того, щоб на фіналі продукт був конкурентоспроможний з можливістю масштабування та подальшої підтримки без проблем зі сторони розробників. Для цього наша команда при розробці використовувала принципи SOLID для проектування класів, щоб мінімальне зв’язування між класами. При проектуванні бази даних були втілені перші три нормальні форми.

Конфігурація API спроектована та розподілена у файлах з текстовим форматом JSON для того, щоб змінюючи один параметр - зміни застосовувались одразу в усьому проекті.

Усі персональні та системні дані зберігаються у базі даних, котра реалізована за допомогою MySQL. База даних була спроектована за першими трьома принципами нормалізації баз даних. Для взаємодії із базою на серверній частині використовуєтсья Entity Framework Core з мовою запитів LINQ від Microsoft. Це пришвидшує розробку та підвищує продуктивність системи. Всі запити до БД (бази даних) написані власноруч з врахуванням зв’язків “один до багатьох” та “багато до багатьох”. При додаванні елементу до БД для нього за допомогою Guid (Globally Unique Identifier) генерується унікальний

ідентифікатор, за котрим потім можна елемент знайти та отримати із БД.

Впроваджуючи медіафайли ми в БД зберігаємо виключно шлях до них, тому що база не може зберігати файли. Самі медіа зберігаються на окремому онлайн зберіговищі.

Також на платформі реалізовані системи реєстрації та авторизації, які забезпечують надійний захист особистих даних користувача. При реєстрації пароль хешується за алгоритмом SHA384. Був обраний саме цей алгоритм через більш продуктивний метод шифрування та для унеможливлення “взламу” особистих даних користувачів. Після реєстрації на вказану користувачем пошту приходить повідомлення з шестизначним кодом та проханням підтвердити пошту на сайті.

На головній сторінці маркетплейсу можна побачити ігрові додатки, котрі були записані до БД, відправлені через запит на сервер до клієнта.

Також реалізований елемент розміщення ігрового застосунку на сайті. Вказується назва, розробник, видавництво, ціна, можливість додавання медіафайлів (скріншотів та відеороликів), додавання категорій. Після викладання гри на маркетплейс користувачі мають можливість написати коментар до гри, та поставити позитивні реакції.

Впроваджена система досягнень, котрі відображаються у форматі значків на сторінці користувача. Досягнення здобуваються за придбані ігри та кількість користувачів у “друзях”.

Додана можливість чатингу між користувачами. Будь який користувач може надіслати повідомлення іншому користувачу, котрий знаходиться в його “друзях”, після чого у них створиться їхній особистий чат, в котрому вони можуть обмінюватись повідомленнями.

Для кожної гри на етапі створення можна додати спільноту, де будуть розміщені всі актуальні новини, дописи розробників, або видавництв. Звичайні користувачі під дописами можуть залишати коментарі.

Також кожний верифікований користувач може створити свою особисту спільноту, куди він може викладати власні дописи.

З приводу коментарів. Вони можуть бути дописані на сторінці ігрового застосунку, на сторінці користувача, під постами у спільнотах та під медіафайлами. Коментарі можуть залишати всі верифіковані користувачі.

Коли користувач придбає ігровий застосунок - він потрапляє до його особистої бібліотеки, де він може подивитись всі придбані застосунки.

Всі особисті ігрові застосунки можна додати до власноруч створених категорій, котрі можна змінювати, додавати туди ігри, вилучати ігри та видаляти категорії.

Також користувач може змінювати свої персональні дані на свої особистій сторінці користувача.

Також впроваджена система “бажаних” ігрових застосунків для кожного користувача. Користувач може помічати ігри на сайті як “бажані”, і вони будуть покладатись в окремий список бажаних ігор, котрий відображається на окремій сторінці.

Спроектована база даних була створена завдяки методу “code first”. Це допомагає зробити зміни ще швидше та прискорити розробку проекту.

РОЗДІЛ 3
АДМІНІСТРАТИВНА ЧАСТИНА ОНЛАЙН-ПЛАТФОРМИ

Платформа має два види користувачів: Виробники та звичайні користувачі. У виробників є можливість розмістити свій власний продукт на маркетплейсі, котрий потім зможуть купувати звичайні користувачі. Щоб користувач став виробником - при реєстрації треба вказати те, що ви реєструєтесь як виробник. Для того щоб користуватись маркетплейсом - треба увійти під своїми обліковими даними. Після того, як ви ввели пошту та пароль - він відправляється на серверну частину, звіряється з паролем, котрий знаходиться в БД, і якщо пароль вірний - користувач отримає JW-token, котрий дозволяє знаходитись та користуватись сайтом.

До реалізованих функцій виробника\адміністратора можна віднести:

- Додавання гри
- Додавання дописів до спільноти гри
- Змінювати ціну гри
- Додавання DLC до гри
- Додавання новин

Усе це було зроблено для максимального комфорту власників платформи та її користувачів. Кожен крок був максимально продуманим та зваженим. Також цей розділ був реалізований так, щоб у майбутньому можна було без зусиль додати нову функціональність

ВИСНОВКИ

Проаналізовані різні способи реалізації взаємодії “Front-end” клієнтських додатків з “Back-end” серверними додатками, а також врахована складність алгоритмів при розробці. В результаті аналізу вдалось визначити, що сучасні технології підвищують продуктивність та прискорюють розробку застосунків. І в результаті аналізу було вирішено використовувати SPA-стиль розробки в якості клієнтської частини та API з серверної частини. Загальним архітектурним стилем був RESTful.

Був розроблений програмний інтерфейс (API), який реалізує безпечне та коректне отримання та відправлення даних за допомогою протоколу HTTP. API було створено на платформі ASP.NET Core з використанням Entity Framework Core у якості ORM системи для взаємодії з базою даних. Це дозволило використовувати мову запитів LINQ, щоб зробити їх максимально гнучкими та оптимізованими.

Спроектовано масштабовану та ефективну БД для швидкого доступу до данних. База даних була обрана у стилі MySQL.

Основним способом захисту даних при зберіганні до БД був обраний криптографічний алгоритм шифрування SHA386. Таке рішення було прийнято після глибокого аналізу предметної області. Також було вжито засоби підтвердження через верифікацію по електронній пошті.

Систему було протестовано шляхом перевірки усіх функціональностей на різних пристроях. Результати випробування та тестування підтвердили загальну працездатність розробленої системи.

ПЕРЕЛІК ВИКОРИСТАНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ СЕРВЕРНОЇ ЧАСТИНИ

№ п.п.	**Ресурс**	**Ступінь запозичення**
1.	ASP.NET Core; Розробник: Microsoft; Документація: посилання	Використана у якості платформи для розробки серверної частини
2.	Entity Framework Core; Розробник: Microsoft; Документація: посилання	Використана як ORM-технологія
3.	Мова програмування C#; Розробник: Microsoft; Документація: посилання	Основна мова для реалізації RESTful API
4.	BCrypt; Розробник: ChrisMcKee; Документація: посилання	Пакет використаний як сервіс для шифрування паролю
5.	Microsoft.AspNetCore.Authentication.JwtBearer; Розробник: Microsoft Документація: посилання	Пакет використаний для впровадження безпечної авторизації та автентифікації
6.	Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore; Розробник: Microsoft; Документація: посилання	Пакет використаний для діагностики та перевірки міграцій, створених Entity Framework
7.	MySQL Workbench; Розробник: Oracle Документація: посилання	Додаток використаний для перевірки бази даних та додавання “синтетичної” інформації за допомогою запитів

8.	Visual Studio; Розробник: Microsoft; Документація: посилання	Додаток використаний в якості середи написання коду та запуску серверної частини
9.	Postman; Розробник: Postman Inc. Документація: посилання	Додаток використаний в якості перевірки надсилання запитів та отримання відповідей з серверної частини
10.	Декларативна мова програмування Transact-SQL; Розробник: Microsoft Документація: посилання	Мова використана для маніпуляції даними в базі даних
11.	Декларативна мова запитів LINQ; Розробник: Microsoft Документація: посилання	Мова використана для маніпуляції даними в базі даних
12.	ChatGPT; Розробник: OpenAI; Документація: посилання	Мовна модель генеративного штучного інтелекту була використана для формування “синтетичної” інформації для бази даних та додавання цієї інформації