Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної підготовки

випускна кваліфікаційна роботи бакалавра

<<СИСТЕМА УПРАВЛІННЯ КОНТЕНТОМ ЕЛЕКТРОННОЇ КОМЕРЦІЇ (за прикладом сервісу Amazon)>>

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»
Виконавці проекту

№	П.І.Б.	Ролі	Група
1	Семенюк Олег Олегович	Backend	КН-П-201
2	Пшеничний Олександр Сергійович	Backend	КН-П-201
3	Вацько Ірина Олександрівна	Frontend	КН-П-201
4	Рибальченко Денис Леонідович	Frontend	КН-П-201
5	Гетманцева Вікторія Юріївна	Designer	КН-Д-201
6	Гарбуз Дар'я Вячеславівна	Designer	КН-Д-202
7	Петров Всеволід Едуардович	DevOps	КН-А-201

Автор звіту
_____ Пшеничний Олександр Сергійович

Одеса – 2024

АНОТАЦІЯ

УДК 004.9
Д-30

Пшеничний О.С. Розробка інтернет магазину. Клон Amazon: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп'ютерні науки». - Одеса: ОТУШ, 2024 - 13 с.

Робота присвячена дослідженню, розробці прототипу інтернет магазину. Основні функції системи включають у собі: відображення товарів, категорій товарів, реєстрацію користувачів, формування замовлень тощо.

Актуальність данної роботи визначається популярністю інтернет магазинів у сучасному світі, повсемісне використання інтернет магазинів як основного виду заробітку серед усіх верств населення. Особливу увагу звертає на себе той факт, що більшість інтернет магазинів (на далі, магазин, e-commerce shop тощо) являються копією одного іншого, мають спільний функціонал, що зосереджений на зручності використання. Це дає нам розуміння того, що варто звертати увагу на охайність верстки (на далі page layout, лейаут) сторінок, оптимізацію магазину та відсутність проблем з оптимізацією, зручність використання веб-сторінок (на далі, сторінки) на мобільних пристроях.

Об'єктом дослідження у рамках даної роботи виступає аналіз найпопулярніших інтернет магазинів світу (такі як Allegro, Ebay.ua тощо) та створення прототипу на базі проаналізованих веб-сайтів. У якості задач дослідження було встановлене наступне:

- Провести аналіз сторінок на функціонал/лейаут/метрики оптимізації.
- Визначити фреймворки та концепції (Vue.js, Bootstrap тощо), які є найбільш вживані у верстці сторінок та дозволяють відображувати великі масиви інформації.
- Розробити головні сторінки та протестувати їх на різних пристроях (персональний ком'ютер, мобільні телефони, планшети тощо).

При складанні звіту використано посилання на друковані та електронні джерела інформації.

ЗМІСТ

Вступ

В останні декілька років інтернет-комерція України стрімко розвивається завдяки Закону України ”Про електронну комерцію” у 2015 році[1]. Інтернет комерція є одним з найприбутковіших видів бізнесу. Обсяг ринку електронної комерції в США досяг приблизно 1.11 трильйона доларів США у 2023 році. Прогнозується, що ринок буде зростати зі швидкістю 14.7% між 2024 і 2032 роками, досягаючи вартості 3.85 трильйона доларів США до 2032 року. Одним з найприбутковіших інтернет-бізнесів є Amazon.com, Inc.[2].

В якості предметної області було взято інтернет-магазин Amazon.com, Inc.як його розклад на такі складові як функціональність, дизайн, адаптивність тощо.

Основні складові Amazon’у та інших схожих магазинів, з точки зору фронтенд розробки, (на далі, front-end) мають у собі адаптивність сторінок, легкість у зміні дизайну (кольорової гами, іконок, елементів на головній сторінці тощо) та серверної частини(на далі, back-end) для зберігання та обробки даних(зберігання до бази даних, авторизація та автентифікація тощо).

Далі у цьому документі розглядаються аналіз системи, проектування архітектури, розробка, тестування та майбутня підтримка цифрової платформи бронювання. Також будуть обговорені проблеми, спірні питання, їх вирішення та реалізація, а також проведений аналіз командної роботи.

Технічне завдання до проекту представлено у наступному розділі під назвою "Технічне завдання до проекту". У рамках даної роботи розглядається серверна частина платформи, а також елементи клієнтської сторони веб-додатку. Висновки демонструють результати, отримані під час тестування цієї частини проекту. Загальні практичні та логічні підсумки є сукупністю звітів усіх учасників проекту

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЄКТУ

ІНФОРМАЦІЙНА СИСТЕМА УПРАВЛІННЯ ЖИТТЄВИМ ЦИКЛОМ ЕЛЕКТРОННОГО МАРКЕТИНГУ
(за зразком Amazon)

Загальний опис.
Програмне забезпечення (ПЗ) призначене для створення розподіленої системи управління контентом, який призначений для електронної комерції. Функції системи мають забезпечити повний життєвий цикл управління контентом: створення, актуалізація, модифікація та знищення. Головна особливість системи полягає у тому, що вона передбачає механізми узгодження контента з віддаленими джерелами даних , а також управління замовленнями, їх оформленням, статусом, а також зберіганням та переданням їх користувачу або зовнішнім системам (платіжна система, служба доставки, тощо).

Призначення:
ПЗ орієнтується на створення контейнеру управління інформаційними одиницями комерційного призначення. Значна увага приділяється маркетинговим інструментам, що дозволяють проводити дослідження ефективності роботи системи, зокрема, зворотному зв'язку між сторонами комерційної взаємодії. ПЗ створюється за прикладом мережевого магазину, 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 комерційних одиниць (або їх груп), вибірка з найбільш популярного та новітнього контенту, а також нових дописів (відгуків), зовнішня реклама. Головна сторінка обмежує функції системи для незареєстрованих користувачів лише переглядом, інші

функції надаються у разі авторизації користувача. Має бути забезпечений пошук контенту, групування та фільтрування результатів за різними ознаками, наявними у знайденому контенті. Для об'ємних результатів має бути забезпечена пагінація.

Реєстрація користувачів:
Користувач ПЗ має мати змогу зареєструватись за допомогою авторизаційних даних (наприклад електронної пошти та паролю). При реєстрації обов'язково зазначається засіб оперативного контактування (ЗОК) - номер телефону, електронна пошта тощо. На тестовому етапі, за умови ускладнення замовлення послуг телефонії чи СМС, можливе використання лише електронної пошти. Засіб оперативного контактування (ЗОК) при реєстрації підлягає верифікації через введення коду з надісланого листа на введену пошту користувачем. 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.
Також передбачається наявність у системі користувачів з додатковими правами: адміністраторів. Реєстрація кореневого адміністратора системи має бути реалізоване при інсталяції системи, наступним користувачам змінюють права наявні адміністратори.

Особистий кабінет користувача:
Основне робоче місце зареєстрованого користувача ПЗ. Повинен забезпечити усі необхідні засоби для

- Управління власними замовленнями: створення, видалення, перегляд усіх замовлень з можливістю більш детального огляду
- Встановлення, зміна та вилучення персональних даних, у т.ч. платіжного засобу, зміна чи відновлення авторизаційних даних (паролю)
- Перегляд поточної активності: перелік відзначеного контенту (кошик, wishlist)

- Формування зворотного зв'язку: надання оцінок, складання відгуків та рекомендацій

Також засобами особистого кабінету має бути передбачено можливість зміни персональних даних.

Зворотній зв'язок
Користувач має змогу встановити власну оцінку (за пятибальною шкалою), або відгук для довільного контенту, щодо якого зафіксована його активність (замовлення). Засоби пошуку, фільтрування та групування мають забезпечити врахування рейтингу контенту.

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 створені користувачами дані (відгуки) мають в обов'язковому порядку пройти погодження (модерацію). Якщо контент порушує норми, кореневий адміністратор має можливість видаляти подібний контент.

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 - додаток, доступний з веб-простору (сайт), що забезпечує візуальний інтерфейс користувачів усіх категорій
.
Усі модулі ПЗ (окрім візуальної моделі Mockup) повинні обов'язково включати до свого складу засоби

- розгортання на новому пристрої (інсталяції) із встановленням початкових таблиць у БД

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 зразок:

- root
    - Backend
    - Frontend
    - Mockup (вихідні файли або посилання на зовнішній проект)
    - DevOps

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

База даних - MySQL
Backend - ASP .NET Core Web-API.
Frontend - JS фреймворк Next.js на основі бібліотеки React
Mockup - визначається ментором з дизайну
DevOps - GitLab, Docker, MySQL, WireGuard VPN

Команда проєкту

Для виконання поставлених завдань передбачається робота команди розробників окремо за кожним з модулів проекту.
Мінімальна команда проекту - чотири учасники, у такій команді кожен з учасників відповідає за одну з частин: Backend, Frontend, Mockup, DevOps.
Дозволяється включення до команди більшої кількості учасників, у такому разі нові учасники будуть розподілені між існуючих частин (Mockup, Backend, Frontend, DevOps).

Звітність

За результатами виконання завдань із створення ПЗ кожен з учасників команди проекту 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 проекту, 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 командую було обрано - SPA (Single Page Application) як головним архітектурним стилем сервісу. Основа програмної архітектури складається з таких само відокремлених частин:

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

Дизайн — одна з ключових частин у розробці проекту, завдяки якій створюється не лише візуальна частина, красива картинка та стилістика, але також продумується структура, логіка та вивчається цільова аудиторія, що допомагає правильно розставити акценти та привернути увагу на певні елементи.

Під час створення відеоролику для інтернет маркетингу, використовувалися такі інструменти, як:

- Adobe Illustrator для створення графічних елементів
- Adobe Lightroom для корекції кольорів фотографій
- Adobe InDesign для створення брендбука
- Figma для верстки сторінок, що підтримує як і стаціонарну, так і мобільну версію, а також для створення UI Kit, карти сайту, цільової аудиторії та інше
- Notion для аналізу конкурентів на ринку
- Adobe After Effect і Adobe Premiere Pro для створення промо ролика, відео презентації.

Завдання, які необхідно реалізувати для успішного виконання цієї частини:

1. Дослідження. Воно дає нам зрозуміти, які переваги та недоліки є у конкурентів, що можна оптимізувати, щоб задовольнити потреби користувачів та оптимізувати використання функціоналу платформи, яка наша цільова аудиторія, на що варто звернути увагу при реалізації проекту, який шлях проходить користувач, перебуваючи на сайті.

2. Карта сайту. Вона дозволяє побачити зв'язки між сторінками, їхню ієрархію, пріоритетність, а також випадки, при яких відкриватиметься та чи інша сторінка.

3. Wireframe. Каркас веб-сайту, або інакше - схема, план, що дає уявлення структури проекту, а саме: схематичне розташування елементів на сторінці та якої вони будуть форми та розміру.

4. Дизайн системи. У цьому пункті створюється логотип, правила використання, колірні рішення, модульна сітка, захисне поле, фірмові кольори, шрифти, графічні елементи, іконки, UI Kit.

5. Макети. Роблять наголос на більш реалістичний дизайн. Визначивши базовий макет і структуру за допомогою каркаса, можна перейти до додавання детальних візуальних елементів, таких як кольори, зображення, шрифти і типографіка.

Frontend - є односторінковим додатком (SPA), що дозволяє досягти максимальної інтерактивності і зробити сайт зручним для користувача, але також має у своєму складі окремий сервер, який рендерить сторінки коли це необхідно, для досягнення найкращого користувацького досвіду та продуктивності роботи сайту у різних сценаріях. Реалізовано за допомогою бібліотеки React та фреймворка Next.js.
Основні цілі, яких необхідно досягти для успішного виконання цієї частини:

- Узгоджений дизайн - більшість елементів мають бути стандартизовані, мати однаковий зовнішній вигляд, колірну схему, стани, тощо.
- Взаємодія з сервером - відповіді з серверу повинні кешуватись для зменшення часу очікування при виконанні однакових запитів, але також повинні оновлюватись після певних дій для відображення актуальних даних.
- Адаптивність - сайт має адаптуватись під різні розміри екранів, зберігаючи оригінальний дизайн, але перебудовуючи його під конкретний пристрій для зручності використання. Особлива увага приділяється мобільним пристроям, тому що значна частина користувачів буде користуватися саме телефонами.
- Користувацький досвід (UX) - якість користувацького досвіду повинна бути висока, чого можна досягти за допомогою таких елементів як: кешування запитів, відображення стану завантаження даних, виділення елементів з якими взаємодіє користувач, зручний одноманітний інтерфейс, наявність версій сайту під різні пристрої.

Backend це невід'ємна частина системи, яка реалізує основний функціонал сайту. Серверна частина розроблена за допомогою фреймворку ASP .NET Core. Backend реалізує REST Api що дозволяє частині frontend звертатися до серверу за допомогою HTTP-запитів. Кожен запит обробляється, валідується, в залежності від результату валідації робляться наступні дії. Якщо результат валідації не задовільний, сервер повертає у відповідь на запит, структуру, яка описує кожну помилку. Задовільний результат передає запит у відповідний обробник, кожен обробник виконує свою роботу з базою даних що включає, створення, оновлення, видалення та читання даних.

DevOps відіграє важливу роль у розробці даного проєкту. Його функції включають реалізацію постійної інтеграції та безперервного розгортання елементів додатка на платформі GitLab. Ми

використовуємо приватні сервери нашого університету з підключенням VPN WireGuard для створення інфраструктури, що дає нам змогу забезпечити надійну та масштабовану роботу проєкту.

Ми також використовуємо сервіси Prometheus і Grafana для створення моніторингу системи серверів, як-от GitLab і сервер Prod, на який відбувається розгортання проєкту, що дає нам змогу слідкувати за споживанням ресурсів, аналізувати дані та вчасно виявляти будь-які проблеми або незвичні відхилення в роботі системи.

РОЗДІЛ 2

РЕАЛІЗАЦІЯ СЕРВЕРНОЇ ЧАСТИНИ

Серверна частина складається з:
- RESTful-API з використанням платформи розробки ASP.NET від Microsoft;
- Бази даних в основі MYSQL;
- Мови програмування: C#, SQL;
- Середовища розробки: Visual Studio, MySQL Workbench;
- Використанні технології: Entity Framework Core, LINQ;
- Встановлені додаткові пакети: AutoMapper, AWSSDK.S3, JwtBearer;
- Тестування: Postman

Сервер реалізує REST API, який дозволяє виконувати стандартні HTTP операції (GET, POST, PUT, DELETE) для роботи з даними.

Сервер реалізує дві частини:

1. Основна частина:

- Реалізує весь необхідний функціонал для покупця, більшість з якого доступна лише для авторизованих користувачів.
- Авторизація на сайті реалізована за допомогою JWT-токенів, які зберігаються у базі даних. Це необхідно для деактивації токенів у випадку виходу користувача зі свого профілю, а також для перевірки валідності токенів.
- Створення нового акаунту складається з кількох етапів: вказівка електронної пошти та пароля, підтвердження пошти, надання прізвища та імені.
- Авторизовані користувачі можуть додавати товари у кошик або список бажань, замовляти товари, залишати відгуки та оцінювати товари, а також ставити вподобання іншим відгукам.
- Всі користувачі можуть переглядати товари, читати відгуки та дивитися рейтинг товарів за допомогою фільтрів чи пошуку.

Адміністративна частина:

2. Адміністративна частина:
    - Реалізує функціонал для адміністратора сайту, включаючи створення нових категорій, наповнення сайту товарами, редагування та видалення товарів.
    - Адміністратори можуть переглядати та видаляти відгуки користувачів, видаляти користувачів, змінювати їм ролі та відновлювати акаунти.

Для архітектури серверної частини був використаний стиль CQRS, який відділяє операції читання від операцій запису, що дозволяє легко масштабувати проект. Для реалізації CQRS використовувався додатковий пакет MediatR.

З самого початку вся архітектура серверної частини була спроектована з використанням патернів та принципів проектування для створення конкурентоспроможного продукту, який можна масштабувати та підтримувати надалі без проблем.

Використовувалися принципи SOLID, надаючи перевагу агрегації з композицією замість успадкування, мислили інтерфейсами, а не класами. База даних була нормалізована за допомогою перших трьох форм нормалізації, використовувалася MySQL, а для маніпулювання з даними – Entity Framework Core з мовою запитів LINQ.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ АДМІНІСТРАТИВНОЇ ЧАСТИНИ

Адміністративна частина знаходиться в одному проєкті разом з с клієнтською частиною.

Адміністративна частина потрібна для керування контентом на сайті. Адміністратор може створювати, редагувати або видаляти категорії для товарів, також відповідає за створення товарів їх видаленні та редагування.

Для кожної категорії під час її створення можна задати правила для товарів які будуть обов’язковими для створення товару. Також під час створення категорії треба задати картинку. Картинки мають відповідати вимогам сайту(розмір, кількість тощо). Обов’язковим також потрібно задати опис для категорії, завдяки тому що категорії створює адміністратор ми можемо не враховувати фактор людини(не цензурна лексика, політичні або релігійні приниження то що)

Основний контент сайту це товари. Модерацією товару займається адміністратор. Для створення товару треба обов’язково вказати категорію. Товар буде неможливо створити якщо не вказати характеристики які були задані в категорії як обов’язкові, це було створено для того щоб товари з однієї категорії мали базові однакові

характеристики та через таке архітектурне рішення можно абстрагуватись від створення товару без характеристик або з невірними характеристиками , характеристики також можна додати в разі потреби. Для кожного товару можно задати від 1 до 10 фотографій. В товару є опис, ціна, рейтинг, відгуки, код. Рейтинг товару можна залишити за шкалою від 1 до 5. Також при створенні товару або його редагуванні можна поставити знижку на цей товар. Описів для товару 2 це короткий опис який можно побачити на карточці товару та детальний опис яких може користувач подивитись під час його перегляду.

Всі картинки на сайті зберігаються на S3 бакеті, за допомогою AWSSDK.S3 був реалізований сервіс який дозволяє завантажувати зображення на бакет. Для кожного з них зберігається посилання у базі даних. Картинки можна додавати, видаляти, або змінювати на нові.

Для взаємодії з категоріями та товарами були створені ендпоінти кожен з яких для адміністративної частини захищений, тільки адміністратор сайту може звернутись до них. Ендпоінти були створені за принципом CRUD(створення, отримання, редагування, видалення)

На сайті є можливість створити новий акаунт покупця, для авторизації користувачів був впроваджений JWT-токен. Токен має свій час роботи, але є деякі ситуації, наприклад якщо користувач вийшов зі свого профілю, токен необхідно деактивувати, якщо змінити сам токен то він вже по суті буде новим. А старим токеном ще можна користуватись поки його строк дії не завершився. Для запобігання цієї ситуації, токени зберігаються в БД, для валідації токену застосовується ряд перевірок наприклад: перевірка валідності токену, перевірка токену у базі чи не деактивований він, отримати користувача за токеном, та перевірити чи не видалений це користувач, чи взагалі існує цей користувач.
Усі паролі користувачів шифруються за допомогою хеш-функції bcrypt, яка кожен раз генерує унікальний хеш, це означає якщо у декількох користувачів повторюються паролі, у кожного все одно

буде свій хеш. Такий підхід ускладнює процес злому паролю аналізуючи кожен хеш.

Висновок

На початку проєкту наша команда провела ретельний аналіз і обговорила всі можливості сайту. Було багато ідей, але деякі функції вирішили реалізувати пізніше, наприклад, комбінації для товарів з різними характеристиками і сервіс для продавців, щоб вони могли створювати свої товари та переглядати статистику.

Ми використали архітектурний патерн CQRS, щоб легко масштабувати проєкт, розділяючи операції на команди або запити. Це дозволяє кожній частині серверу працювати незалежно, не впливаючи на вже існуючий функціонал.

Зараз у нас є додаток, який дозволяє переглядати товари, шукати їх, фільтрувати за характеристиками, читати відгуки та дивитися рейтинги. Авторизовані користувачі можуть зберігати товари в списках бажань, додавати їх до кошика, робити замовлення та скасовувати їх. Вони також можуть змінювати дані свого профілю та видаляти акаунт. Для авторизації ми використовуємо JWT-токени, а паролі шифруються за допомогою хеш-функції bcrypt, що забезпечує високу безпеку.

В процесі розробки ми використали сучасні технології, що підвищують продуктивність і допомагають створити масштабовану систему. Серверна частина реалізована на ASP.NET Core, а для роботи з базою даних ми використовували Entity Framework Core з LINQ. База даних спроектована на Microsoft SQL Server.

Однією з найбільших складностей було відсутність системи проектування, що потребувало значних зусиль для її інтеграції в існуюче застосування. Це важливо враховувати на початкових етапах розробки, щоб уникнути проблем у майбутньому.

Ми обрали RESTful архітектурний стиль, а клієнтську частину реалізували за допомогою SPA-стилю. API забезпечує безпечне отримання та відправку даних через HTTP. База даних спроектована для швидкого доступу до даних.

Висновки показують, що використання сучасних технологій і правильне проектування дозволили створити продукт, який можна легко масштабувати та підтримувати. Принципи SOLID допомогли нам створити добре структурований код. Конфігурація API централізована у файлах JSON, що спрощує подальші зміни та розробку.

Таким чином, наш додаток надає широкі можливості для користувачів і адміністраторів, забезпечуючи високу продуктивність, безпеку та масштабованість.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

- Мова програмування C#
- ASP .NET Core
- Entity Framework Core
- EF LINQ
- CQRS
- MediatR
- Auto-Mapper
- FluentValidation
- AWSSDK
- Postman
- MySQL
- ChatGPT
- GitHub
- Git docs
- Stack Overflow
- Slugify
- NewtonsoftJSON
- YouTube
- Medium
- Microsoft Learn