МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2021 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

ІНТЕРНЕТ-МАГАЗИН КНИЖКОВОЇ ПРОДУКЦІЇ З МОДУЛЕМ ОПТИМІЗАЦІЇ ПРОПОЗИЦІЙ ТОВАРУ ВІДВІДУВАЧАМ

Спеціальність 122 «Комп'ютерні наук»

122 – БКР – 401.1810108

Виконав студент 4-го курсу, групи 401
_____ Дарій А.М.
«22» червня 2022 р.

Керівник: канд. техн. наук, доцент
_____ Кошовий В.В.
«22» червня 2022 р.

Миколаїв – 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко
«____» _____20____р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Видано студенту групи 401 факультету комп'ютерних наук

Дарію Артему Миколайовичу.
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи
Інтернет-магазин книжкової продукції з модулем оптимізації пропозицій товару відвідувачам.

Керівник роботи ст.викладач Кошовий Віталій Володимирович.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затв. наказом Ректора ЧНУ ім. Петра Могили від «___» ____ 20__ р. № _____

2. Строк представлення кваліфікаційної роботи студентом «__» _____ 20__р.

3. Вхідні (початкові) дані до роботи:_____
_____
_____
_____.

Очікуваний результат роботи: Розроблений інтернет-магазин з модулем рекомендацій товару.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки
аналіз сучасного стану задачі вибору технології;
огляд існуючих методів;
експертне оцінювання технологій;
порівняльний аналіз результатів застосування обраних методів для.

5. Перелік графічних матеріалів презентація.

6. Завдання до спеціальної частини
Захист від іонізуючих випромінювань.

7. Консультанти:

Розділ	Прізвище, ініціали та посада консультанта	Підпис

Керівник роботи
ст.викладач Кошовий Віталій Володимирович
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання _____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання «____» _____ 202__ р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: ІНТЕРНЕТ-МАГАЗИН КНИЖКОВОЇ ПРОДУКЦІЇ З МОДУЛЕМ ОПТИМІЗАЦІЇ ПРОПОЗИЦІЙ ТОВАРУ ВІДВІДУВАЧАМ .

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Оформлення індивідуального завдання	03.05.2022	04.05.2022	виконано
2.	Ознайомлення з Робочою програмою переддипломної практики і правилами оформлення звіту	10.05.2022	17.05.2022	виконано
3	Складання плану звіту	16.05.2022	19.05.202	виконано
4	Підбір матеріалів по звіту – постановка задачі і аналіз існуючих аналогів (Розділ 1 ДР)	20.05.2022	20.05.2022	виконано
5	Аналіз існуючих аналогів по темі ДР – переваг та недоліків	23.05.2022	27.05.2022	виконано
6	Розробка постановки задачі по темі ДР	28.05.2022	29.05.2022	виконано
7	Узгодження матеріалів переддипломної практики з керівником ДР	29.05.2022	30.05.2022	виконано
8	Оформлення звіту по переддипломній практиці	29.05.2022	30.05.2022	виконано
9	Здача заліку по переддипломній практиці	29.05.2022	30.05.2022	виконано
10	Доповнення Розділу 3 та 4	11.06.2022	15.06.2022	виконано
11	Коригування оформлення БКР	19.06.2022	22.06.2022	виконано
12	Збір рецензій та відгуків	22.06.2022	23.06.2022	виконано

Розробив студент Дарій А.М _____
(прізвище та ініціали) (*підпис*)

Керівник роботи ст.викладач Кошовий Віталій Володимирович
(*ступень, звання, прізвище, ім'я, по батькові)* _____
(підпис)

«____» _____ 2022 р.

АНОТАЦІЯ

бакалаврської кваліфікаційної роботи
студента групи 401 ЧНУ ім. Петра Могили
Дарія Артема Миколайовича
на тему:

ІНТЕРНЕТ-МАГАЗИН КНИЖКОВОЇ ПРОДУКЦІЇ З МОДУЛЕМ ОПТИМІЗАЦІЇ ПРОПОЗИЦІЙ ТОВАРУ ВІДВІДУВАЧАМ

Мета роботи: Створення інтернет-магазину книжкової продукції з модулем оптимізації пропозицій товару відвідувачам.

Об'єкт дослідження: створення та налаштування роботи інтернет-магазину книжкової продукції з модулем оптимізації пропозицій товару відвідувачам.

Предмет дослідження: методи оптимізації пропозицій.

Ціль: створити модуль для оптимізації пропозицій.

Методи дослідження: Фреймворк Laravel,.

Розділи кваліфікаційної роботи:

1. Аналіз предметної області.
2. Моделі, методи та інформаційні технології для вирішення поставленої задачі.
3. Моделювання та проектування інформаційної системи
4. Програмна реалізація та розробка документації
5. Охорона праці

Сторінок – 78, таблиці - 4, рисунків – 45, посилань – 9, додатків – 0.

Ключові слова: Фреймворк, Laravel, php, MySQL, База Даних, сайт.

ABSTRACT

bachelor's qualification work
student group 401 CHMNU im. Petra Mogili
Dariy Artem Mykolayovych
on the topic:

INTERNET STORE OF BOOK PRODUCTS WITH MODULE OF OPTIMIZATION OF PRODUCT OFFERS TO VISITORS

Purpose of the work: Creating an online bookstore with a module for optimizing product offers to visitors.

Object of study: creation and adjustment of work of online store of book production with the module of optimization of offers of the goods to visitors.

Subject of research: methods of optimization of offers.

Goal: create a module for optimizing proposals.

Follow-up methods.

Distributed qualification work:

1. Analysis of the subject area.
2. Models, methods and information technologies to solve the problem.
3. Modeling and design of information system
4. Software implementation and development of documentation
5. Labor protection

Pages - 78, tables - 4, drawings - 45, posting - 9, supplements - 0.

Key words: Framework, Laravel, php, MySQL, Database, website.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗ – програмне забезпечення

БКР – бакалаврська кваліфікаційна робота

БД – База Даних

СУБД - Система управління базами даних

Web – всесвітнє павутиння

HighLoad - високонавантажені web-застосунки, з великою кількістю запитів на секунду

URL – Уніфікований покажчик ресурсу (Uniform Resource Locator)

Пояснювальна записка

до кваліфікаційної роботи

на тему:

ІНТЕРНЕТ-МАГАЗИН КНИЖКОВОЇ ПРОДУКЦІЇ З МОДУЛЕМ ОПТИМІЗАЦІЇ ПРОПОЗИЦІЙ ТОВАРУ ВІДВІДУВАЧАМ

Спеціальність 122 «Комп’ютерні науки»

122 – БКР – 401.1810108

Виконав студент 4-го курсу, групи 401

_____ Дарій А.М.

«20» червня 2022 р.

Керівник: канд. техн. наук, доцент

_____Кошовий В.В.

«20» червня 2022 р.

Миколаїв – 2022

ВСТУП

Починаючи з XX століття, ми все більше і більше починаємо впроваджувати сучасні технології в наше життя. Багато сфер та областей виробництва автоматизуються.

Щороку у сфері автоматизації окремих сфер діяльності людини відбуваються значні зміни, які у свою чергу ведуть до змін у свідомості людей. Відчувши, що використання автоматизованих засобів багато разів полегшує роботу і при цьому тільки покращує її якість, досить важко відмовитися від придбання та впровадження комп'ютера.

Системи автоматизації торговельної діяльност у останні роки повільно, але вірно займають своє місце у цій сфері діяльності людей. Автоматизація обліку наявних та відпущених товарів, розрахунків та створення звітів у багато разів підвищують ефективність та якість роботи, значно полегшують працю працівників.

З поширенням мережі Інтернет виникли «електронні магазини», що торгують різними товарами. Порівняно із звичайними магазинами вони мають безліч переваг, які сприяють зростанню доходів у цій сфері торгівлі. Особливо сприяє зростанню доходів індивідуальні пропозиції товарів користувачу.

Ціль бакалаврської кваліфікаційної роботи полягає в тому, щоб провести дослідження анатомії рекомендаційних систем та рекомендаційних алгоритмів, провести дослідження наявних на ринку рекомендаційних систем та виявити їх вади. І як результат, розробити інтернет-магазин книжкової продукції з модулем рекомендацій, який буде базуватись на досліджених уже створених системах, алгоритмах та буде створюватись за допомогою отриманих знань.

Для досягнення мети використовуватимуться:

1) Знання про рекомендаційні системи;
2) Знання про алгоритми рекомендацій;
3) Орієнтація у роботі з Laravel Framework;

4) орієнтація у мові програмування PHP та орієнтація у роботі з Базами Даних.

5) допоміжні знання про Web-бібліотеки такі як jQuery, twitter-bootstrap, Swiper.js та інші.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз сфери продажу книг в мережі інтернет.

Інтернет-магазин це онлайн-майданчик, на якому ведеться купівля-продаж товарів та послуг. З її допомогою користувачі мережі можуть сформувати замовлення, вибрати зручний для них спосіб оплати та доставки замовлення. Саме поняття «інтернет-магазин» включає безліч складових. Серед них прийнято виділяти створення інформаційного наповнення, каталог товарів, систему авторизації та обробки замовлень, збирачів даних та підключення платіжних систем.

Інтернет-магазини мають власну специфіку. Визначається вона такими особливостями їх роботи: цілодобовий доступ до замовлення товарів та послуг через сайт, широка географічна доступність, відсутність прив'язки до конкретного приміщення, можливість демонстрації всіх товарів на одній вітрині, а також показу тих позицій, які тільки очікуються, щодо низьких витрат на персонал, можливість фокусування на конкретній аудиторії та ін.

Аудиторія інтернет-магазинів постійно розширюється. Перш за все це пов'язано з тим, що кількість інтернет-користувачів з кожним роком зростає. Водночас зростає і кількість інтернет-магазинів. Купувати онлайн стає дедалі вигідніше та зручніше.

Купівля товарів та послуг через інтернет-магазини має чимало переваг. Основними серед них вважаються економія часу, гнучкість цінової політики, низькі ціни, великий асортимент товарів та послуг, їх доступність та простота доставки. Крім того, купуючи товар у тому чи іншому магазині, потенційний покупець завжди може знайти відгуки про його роботу та продукцію, що їм пропонується. Таким чином, процес купівельного вибору стає більш усвідомленим та обґрунтованим.

Книжкова торгівля в сучасному світі, і в сучасній Україні зокрема є надзвичайно важливим сегментом, що має дуже серйозне соціальне значення, адже книги багато в чому визначають культурний та духовний стан суспільства. З економічної точки зору надання торгових послуг з продажу книг, газет, інших періодичних видань є важливим не тільки як один з сегментів найважливішої галузі національної економіки, що динамічно розвивається, але і як надзвичайно цікавий у науковому плані об'єкт дослідження, що дозволяє вивчити і в подальшому екстраполювати виявлені закономірності на деякі області та сектори.

На Заході, тобто на ринку Європи та Америки, продаж книг через Інтернет, мабуть, головна інновація в книжковому видавничому бізнесі за останні десятиліття. Інтернет-продаж книг в Україні, нажаль останній час не дотягував до цього рівня (за різними оцінками, на них припадає 1–2% ринку). Причини тому – нерозвиненість електронних платежів, всезагальне піратство в мережі, слабка правова база, низька комп'ютеризація країни, відсутність налагоджених мереж доставки книг, бідність населення, відносна дешевизна паперових видань, культурні традиції. Проте в українському віртуальному просторі існує кілька великих магазинів які займаються онлайн продажем різної продукції, у тому числі і книжкової. Також існують книжкові магазини, які працюють уже не перший рік і заявляють про свою успішність. Проте більшість учасників ринку не вважає зміщення попиту в Інтернет визначальною тенденцією. Також не можна забувати, що значна конкуренція та невисока вартість книг призводить до низької, а часто негативної рентабельності перших етапів роботи книжкових віртуальних магазинів. Але, незважаючи на це, будь-який книжковий інтернет-магазин знаходив свого покупця, що безпосередньо залежало і до сьогоднішнього дня залежить від якості опрацювання сайту. Така ситуація з інтернет-магазинами була до недавніх пір.

Актуальність створення інтернет-магазинів набула досить високого рівня з початком світової пандемії. У цей період часу були досить сильні обмеження через що маленькі бізнеси, накшталт книжкових магазинів, які і так були не

досить популярним місцем перебування людей геть знелюдніли. І якщо бізнес не переорієнтовувався в режим онлайн, він дуже швидко втрачав прибуток, про його існування поступово забували люди і в решті решт бізнес становився банкрутом.

1.2 Огляд та аналіз наявних аналогів інтернет-магазинів

Як було зазначено вище, в Україні існує кілька великих магазинів які займаються онлайн продажем різної продукції, у тому числі і книжкової, а також виключно книжкові магазини. Я буду спиратись не тільки на те, що пропонується в Українському просторі, але і зроблю аналіз Західних інтернет магазинів.

Rozetka

Інтернет-магазин Rozetka на сьогоднішній день є найпопулярнішим магазином України. Такого статусу магазину вдалося досягти завдяки максимально широкому асортименту, розумній ціновій політиці та відмінному сервісу.

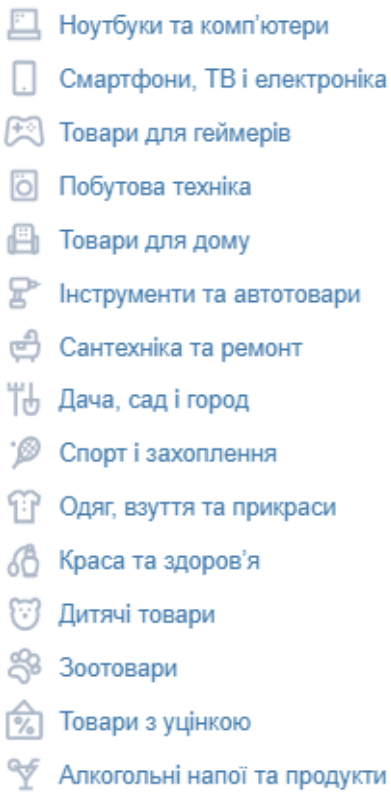


Рисунок 1.1 – Каталог категорії магазину ROZETKA

Це дуже зручно з точки зору того, що в цьому інтернет-магазині ми можемо знайти майже усіх категорій, які тільки може спасти нам на думку. Але у цьому і є основний недолік цього магазину. Він не спеціалізується в якійсь певній області, тому, навіть якщо у магазині присутня потрібна нам категорія, ми можемо не знайти ту саму модель, або у моєму випадку книгу, яка потрібна.

Prom

Prom це другий за розміром маркетплейс в Україні. Так само як і магазин розетка Prom налічує велику кількість категорій товарів від тисячі підприємців зі всієї України і так само як і ROZETKA магазин не спеціалізується в якійсь певній області, тому на сайті можна знайти товари майже усіх категорій, але все одно можна не знайти те що треба.

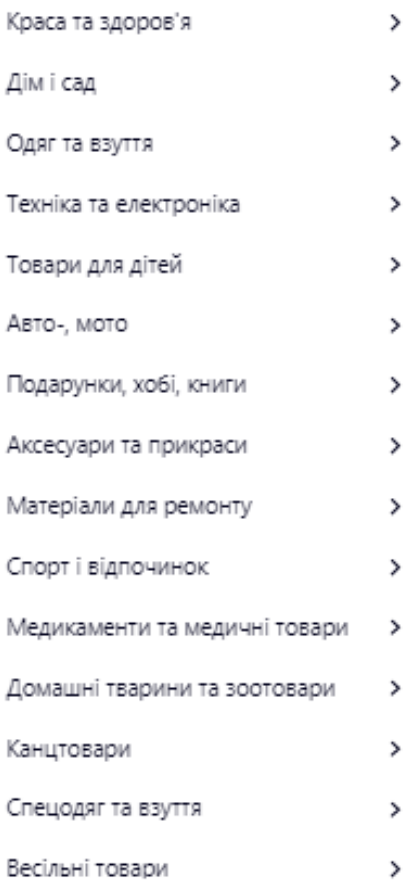


Рисунок 1.2 – Каталог категорії магазину Prom.ua

Yakaboo

Yakaboo – найбільший книжковий інтернет магазин в Україні. Як і у попередньо зазначених магазинах Prom і ROZETKA, у цьому магазині є дуже багато категорій товарів, за однією відмінністю – усі товари це книжки.

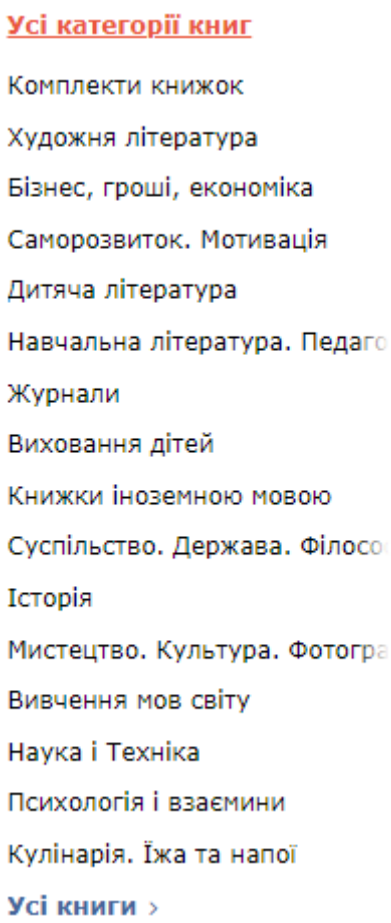


Рисунок 1.3 – Каталог книжок магазину Yakaboo

Враховуючи те, що магазин спеціалізований то буде легше знайти щось до вподоби, просто тому що асортимент дуже різноманітний. Наприклад коли я купував собі книги, я почав шукати на ROZETKA, тому що цей бренд знає кожний, та я врахував величезний асортимент, але нажаль я не знайшов те що мене цікавило і знайшов на спеціалізованому сайті Yakaboo. Yakaboo має в своєму розпорядженні модуль який відповідає за пропозицію товарів користувачам. Для цього потрібно буди авторизованим на сайті. Цей модуль робить пропозиції ґрунтуючись на тому, що користувач дивився востаннє, а саме робить пропозиції книг з тої ж самої категорії, в якій знаходиться нещодавно переглянута вами книга.

Також я хочу звернути увагу на деякі Західні магазини. Наприклад **americanbookstore.pl** та **buch24.de.**

Спершу хочеться звернути увагу саме на americanbookstore. Досить вузько спеціалізований магазин. По-перше магазин польський. По-друге уся література англійською мовою. На перший погляд нічого такого, адже у кожній країні є люди які вивчають іноземні мови і для практики читають різну літературу. Але все ж таки заради успішності магазину і прибутку це не дуже логічне рішення. У цього інтернет магазину присутній модуль для пропозицій товарів. Він рекомендує товари зареєстрованим користувачам ґрунтуючись на тому, що користувач дивився востаннє, як і на Yakaboo.

Що стосується інтернет-магазина buch24, то тут картина значно гірша. Тут також є рекомендації, та вони не зроблені відповідно до кожного користувача. Тут вони загальні, наприклад книги що мають найбільший успіх у продажах, або категорії, що рекомендуються амінами сайту, тощо.

1.3 Постановка задачі

Враховуючи всю актуальність цього питання, можна затвердити, що об'єктом дослідження є алгоритми рекомендацій, як напрямок у сфері розробки не тільки сайтів для розміщення товарів на продаж, а ще й сайтів спрямованих на надання контенту користувачам . Відповідно, предметом дослідження є створення та налаштування роботи модуля пропозицій товарів для інтернет-магазину книжкової продукції.

Ціль: аналіз вподобань користувача з подальшою можливістю використовувати ці данні для коректних пропозицій саме цьому користувачу, що збільшує шанси на придбання товару.

Для досягнення мети використовуватимуться:

1) знання анатомії рекомендаційних систем;
2) орієнтація у мові програмування PHP, JavaScript;
3) знання бібліотеки jQuery;
4) знання Laravel Framework.

Для виконання цього завдання повинні бути описані *вимоги проекту*.

Призначення — модуль буде створений для оптимізації пропозицій товарів користувачам.

Кордони проекту — аналіз вподобань окремого користувача для вдалої пропозиції товару для нього.

Класи та характеристики користувачів — усвідомленими користувачами будуть адміністратори та власники сайтів, які будуть використовувати модуль для збільшення продажів, тобто свого прибутку, а неусвідомленими будуть користувачі, які будуть використовувати цей модуль для успішного обрання товару для себе.

Висновки до розділу 1

В ході написання першого розділу був проведений аналіз сфери інтернет-торгівлі. Було виявлено, що сегмент книжкової торгівлі мережі Інтернет як на західному ринку так і в Україні є надзвичайно важливим сегментом, що має дуже серйозне соціальне значення, адже книги багато в чому визначають культурний та духовний стан суспільства. Також була сформована думка, чому сфера інтернет-торгівлі стає ще більш актуальнішою з кожним днем. Був проведений аналіз великих магазинів України та їх порівняння. Був проведений аналіз книжкових магазинів як українських так і західних для того, щоб можна було виявити слабкі місця. Також була поставлена задача і сформована ціль роботи. І на останок було визначено пул технологій для вирішення задачі.

2 МОДЕЛІ, МЕТОДИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ДЛЯ ВИРІШЕННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1 Методи для вирішення задачі

Анатомія рекомендаційної системи

Завдання рекомендаційної системи – проінформувати користувача про товар, який може бути найцікавіший у даний момент часу. Клієнт отримує інформацію, і залежно від відсотка відповідності вподобанням користувача, зростає вірогідність успішного замовлення користувачем, а сервіс заробляє на наданні якісних послуг. Хоча послуги - це не обов'язково прямий продаж запропонованого товару. Сервіс також може заробляти на комісійних або просто збільшувати лояльність користувачів, яка потім виливається у рекламні та інші доходи.

Залежно від моделі бізнесу рекомендації можуть бути його основою, як, наприклад, у TripAdvisor, а можуть бути просто зручним додатковим сервісом (як, наприклад, в якомусь інтернет-магазині одягу), покликаним покращити Customer Experience та зробити навігацію за каталогом. зручною.

Персоналізація онлайн-маркетингу – очевидний тренд останнього десятиліття. За оцінками McKinsey, 35% виручки Amazon або 75% Netflix припадає саме на рекомендовані товари і цей відсоток, ймовірно, зростатиме. Рекомендаційні системи – це те, що запропонувати клієнту, щоб зробити його щасливим.

Щоб проілюструвати всю різноманітність рекомендаційних сервісів, наведу список основних характеристик, за допомогою яких можна описати будь-яку рекомендаційну систему.

Предмет рекомендації – що рекомендується.

Тут велика різноманітність – це можуть бути товари (Amazon, Ozon), статті (Arxiv.org), новини (Surfingbird, Яндекс.Дзен), зображення (500px), відео (YouTube, Netflix), люди (Linkedin, LonelyPlanet), музика (Last.fm, Pandora), плейлисти та інше. Загалом, рекомендувати можна будь-що.

Ціль рекомендації – навіщо рекомендується.

Наприклад: покупка, інформування, навчання, заклад контактів.

Контекст рекомендації – що користувач у цей момент робить.

Наприклад: дивиться товари, слухає музику, спілкується з людьми.

Джерело рекомендації – хто рекомендує:

- аудиторія (середній рейтинг);
- схожі за інтересами користувачі;
- експертне співтовариство (буває, коли йдеться про складний товар, такий, як, наприклад, вино).

Ступінь персоналізації.

Неперсональні рекомендації – коли вам рекомендують те саме, що всім іншим. Вони допускають таргетинг по регіону або часу, але не враховують ваші особисті уподобання.

Більш просунутий варіант – коли рекомендації використовують дані вашої поточної сесії. Ви подивилися кілька товарів, і унизу сторінки вам пропонуються схожі.

Персональні рекомендації використовують всю доступну інформацію про клієнта, зокрема історію його покупок.

Прозорість.

Люди більше за все довіряють рекомендації, якщо розуміють, як саме її було отримано. Так менший ризик нарватися на «несумлінні» системи, які просувають товар за гроші або ставлять дорожчі товари вище в рейтингу. Крім того, хороша рекомендаційна система сама повинна вміти боротися з купленими відгуками та накрутками продавців.

Маніпуляції до речі бувають і ненавмисними. Наприклад, коли виходить новий блокбастер, насамперед на нього йдуть фанати, відповідно, першу пару тижнів рейтинг може бути сильно завищений.

Формат рекомендації.

Це може бути вікно, що спливає, відсортований список що з'являється в певному розділі сайту, стрічка внизу екрана або щось ще.

Алгоритми.

Незважаючи на безліч існуючих алгоритмів, вони зводяться до кількох базових підходів, які будуть описані далі. До найбільш класичних відносяться алгоритми Summary-based (неперсональні), Content-based (моделі, засновані на описі товару), Collaborative Filtering (колаборативна фільтрація), Matrix Factorization (методи засновані на матричному розкладанні) та деякі інші.

Неперсоналізовані рекомендації

Почнемо з неперсоналізованих рекомендацій, оскільки вони найпростіші у реалізації. Вони потенційний інтерес користувача визначають просто середнім рейтингом товару: «Всім подобається – значить сподобається і вам». За цим принципом працює більшість сервісів, коли користувач не авторизується в системі, наприклад той же TripAdvisor.

Показуватись рекомендації можуть по-різному – як банер збоку від опису товару (Amazon), як результат запиту, відсортований за певним параметром (TripAdvisor), або ще якось.

Рейтинг товару також може зображуватись різними способами. Це можуть бути зірочки поруч із товаром, кількість лайків, різниця позитивних та негативних голосів (як зазвичай роблять на форумах), частка високих оцінок або взагалі гістограма оцінок. Гістограми – найбільш інформативний спосіб, але вони мають один мінус – їх складно порівнювати між собою або сортувати, коли потрібно вивести товари списком.

Проблема холодного старту

Холодний старт – це типова ситуація, коли ще не накопичено достатньо даних для коректної роботи рекомендаційної системи (наприклад, коли товар новий або просто його дуже рідко купують). Якщо середній рейтинг порахований за оцінками всього трьох користувачів (igor92, xyz_111 та oleg_s), така оцінка не буде достовірною, і користувачі це розуміють. Часто у таких ситуаціях рейтинги штучно коригують.

Перший спосіб показувати не середнє значення, а згладжене середнє (Damped Mean). Сенс такий: при малій кількості оцінок рейтинг, що

відображається, більше тяжіє до якогось безпечного «середнього» показника, а як тільки набирається достатня кількість нових оцінок, «усереднювальне» коригування перестає діяти.

Інший підхід – розраховувати за кожним рейтингом інтервали достовірності (confidence Intervals). Математично, чим більше оцінок, тим менша варіація середнього і, отже, більша впевненість у його коректності. А як рейтинг можна виводити, наприклад, нижню межу інтервалу (Low CI Bound). При цьому зрозуміло, що така система буде досить консервативною, з тенденцією до заниження оцінок щодо нових товарів (якщо, звичайно, це не хіт).

Content-based рекомендації

Персональні рекомендації передбачають максимальне використання інформації про самого користувача, насамперед про його попередні покупки. Одним з перших з'явився підхід content-based filtering. У межах цього підходу опис товару (content) зіставляється з інтересами користувача, отриманими з попередніх оцінок. Що товар цим інтересам відповідає, то вище оцінюється потенційна зацікавленість користувача. Очевидна вимога тут – у всіх товарів у каталозі має бути опис.

Історично предметом Content-based рекомендацій найчастіше були товари з неструктурованим описом: фільми, книги, статті. Такими ознаками можуть бути, наприклад, текстові описи, рецензії, склад акторів та інше. Однак ніщо не заважає використовувати і звичайні числові чи категоріальні ознаки.

Неструктуровані ознаки описуються типовим тексту способом – векторами у просторі слів (Vector-Space model). Кожен елемент такого вектора – ознака, що потенційно характеризує інтерес користувача. Аналогічно, продукт – вектор у тому самому просторі.

У міру взаємодії користувача із системою (скажімо, він купує фільми), векторні описи придбаних ним товарів об'єднуються (сумуються та нормалізуються) у єдиний вектор і, таким чином, формується вектор його

інтересів. Далі досить знайти товар, опис якого найближче до вектора інтересів, тобто. вирішити задачу пошуку n найближчих сусідів.

Не всі елементи однаково значущі: наприклад, союзні слова, очевидно, не мають жодного корисного навантаження. Тому щодо кількості співпадаючих елементів у двох векторах все виміри потрібно попередньо зважувати з їхньої значимості. Це завдання вирішує добре відоме в Text Mining перетворення TF-IDF, яке призначає більшу вагу більш рідкісним інтересам. Збіг таких інтересів має більше значення щодо близькості двох векторів, ніж збіг популярних.

Content-based фільтрація майже повністю повторює механізм query-document matching, який використовується у пошукових системах типу Яндекс та Google. Відмінність лише у формі пошукового запиту — це вектор, який описує інтереси користувача, а там — ключові слова запитуваного документа. Коли пошукові системи стали додавати персоналізацію, відмінність стерлося ще більше.

Колаборативна фільтрація (user-based)

Цей клас систем почав активно розвиватися у 90-ті роки. У рамках підходу рекомендації генеруються виходячи з інтересів інших схожих користувачів. Такі рекомендації є результатом «колаборації» багатьох користувачів. Звідси й назва методу.

Класична реалізація алгоритму заснована на принципі найближчих сусідів. На пальцях – для кожного користувача шукаємо найбільш схожих на нього (в термінах переваг) і доповнюємо інформацію про користувача відомими даними по його сусідам. Так, наприклад, якщо відомо, що ваші сусіди за інтересами у захваті від роману «Гаррі Поттер», а ви його з якоїсь причини ще не читали, це чудова нагода запропонувати вам цю книжку для читання.

"Схожість" - в даному випадку синонім "кореляції" інтересів і може вважатися безліччю способів (крім кореляції Пірсона, є ще косинусне відстань, є відстань Жаккара, відстань Хеммінга та ін.).

У класичної реалізації алгоритму є один явний мінус - він погано застосовується на практиці через квадратичну складність. Справді, як і будь-

який метод найближчого сусіда, він вимагає розрахунку всіх попарних відстаней між користувачами (а користувачів можуть бути мільйони). Неважко порахувати, що складність розрахунку матриці відстаней буде $O(n^2m)$, де n – число користувачів, а m – число товарів. При мільйоні користувачів для зберігання матриці відстаней у сирому вигляді потрібно мінімум 4TB.

Ця проблема може бути вирішена покупкою високопродуктивного заліза. Але якщо підходити з розумом, то краще ввести коригування в алгоритм:

1. оновлювати відстані не при кожній покупці, а батчами (наприклад, щодня),
2. не перераховувати матрицю відстаней повністю, а оновлювати її інкрементально,
3. зробити вибір на користь ітеративних та наближених алгоритмів (наприклад ALS).

Для того щоб алгоритм був ефективний, важливо, щоб виконувалося кілька припущень:

1. Вподобання людей не змінюється з часом (або змінюється, але для всіх однаково).
2. Якщо уподобання людей збігаються, то вони збігаються у всьому.

Наприклад, якщо два клієнти віддають перевагу книгам з певної категорії, то автори їм теж подобаються однакові. Якщо це не так, то у пари клієнтів цілком можуть збігатися переваги в книжках, а політичні погляди бути прямо протилежними — алгоритм буде менш ефективним.

Околиця користувача у просторі переваг (його сусіди), яку ми аналізуватимемо для генерації нових рекомендацій, можна обирати по-різному. Ми можемо працювати взагалі з усіма користувачами системи, можемо задати якийсь поріг близькості, можемо вибрати кілька сусідів випадковим чином або брати найбільш схожих сусідів (це найбільш популярний підхід).

Автори з MovieLens як оптимальну кількість сусідів наводять цифри в 30-50 сусідів для фільмів і 25-100 для довільних рекомендацій. Тут зрозуміло, що якщо візьмемо занадто багато сусідів, то отримаємо більше ймовірність

випадкового шуму. І навпаки, якщо візьмемо дуже мало, то отримаємо точніші рекомендації, але меншу кількість товарів можна рекомендувати.

Колаборативна фільтрація (Item-based)

Підхід Item-based є природною альтернативою класичному підходу User-based, описаному вище, і майже повністю його повторює, за винятком одного моменту - застосовується до транспонованої матриці переваг. Тобто. шукає близькі товари, а чи не користувачів.

Нагадаю, колаборативна фільтрація користувача (user-based CF) шукає для кожного клієнта групу найбільш схожих на нього (у термінах попередніх покупок) клієнтів і усереднює їх переваги. Ці усереднені переваги і є рекомендаціями для користувача. У випадку з товарною колаборативною фільтрацією (item-based CF) найближчі сусіди шукаються на безлічі товарів — стовпців матриці переваг. І усереднення відбувається саме з них.

Справді, якщо продукти змістовно схожі, то, швидше за все, вони або одночасно подобаються, або одночасно не подобаються. Тому коли ми бачимо, що у двох товарів оцінки дуже корелюють, це може говорити про те, що це товари-аналоги.

Переваги Item-based перед User-based:

1. Коли користувачів багато (майже завжди), завдання пошуку найближчого сусіда стає погано обчислюваним. Наприклад, для 1 млн користувачів потрібно розрахувати та зберігати $\frac{1}{2}10^6 * 10^6 \sim$ 500 млрд відстаней. Якщо відстань кодувати 8 байтами, це виходить 4TB для однієї матриці відстаней. Якщо ми робимо Item-based, то складність обчислень знижується з $O(N^2 n)$ до $O(n^2 N)$, а матриця відстаней має розмірність не (1 млн на 1 млн) а, наприклад, (100 на 100) за кількістю товарів.
2. Оцінка близькості товарів набагато точніша, ніж оцінка близькості користувачів. Це прямий наслідок того, що користувачів зазвичай набагато більше, ніж товарів і, отже, стандартна помилка при

розрахунку кореляції товарів там істотно менше. У нас просто більше інформації, щоб зробити висновок.

3. У user-based варіанті опису користувачів, як правило, сильно розріджені (товарів багато, оцінок мало). З одного боку, це допомагає оптимізувати розрахунок — ми перемножуємо тільки ті елементи, де є перетин. Але з іншого боку — скільки сусідів не бери, список товарів, які можна порекомендувати, виходить дуже невеликим.
4. Уподобання користувача можуть змінюватися з часом, але опис товарів штука набагато стійкіша.

В іншому алгоритм майже повністю повторює User-based варіант: та ж косинусна відстань як основна міра близькості, та сама необхідність нормалізації даних. Число сусідніх товарів N зазвичай вибирають у районі 20.

Через те, що кореляція продуктів вважається на більшій кількості спостережень, не так критично перераховувати її після кожної нової оцінки і можна це робити періодично в батчевому режимі.

При використанні item-based підходу рекомендації мають тенденцію бути більш консервативними. Справді, розкид рекомендацій виходить меншою і отже меншою є можливість показати нестандартні товари.

Якщо в матриці переваг як рейтинг ми використовуємо перегляд опису товару, то рекомендовані товари швидше за все будуть аналогами - товарами, які часто дивляться разом. Якщо ж рейтинги в матриці переваг ми розраховуємо на підставі покупок, то, швидше за все, рекомендовані товари будуть аксесуарами — товарами, які часто купують разом.

2.2 Технології для розробки системи

HTML (від англ. HyperText Markup Language - «мова гіпертекстової розмітки») - стандартизовані мову розмітки документів у Всесвітній павутині. Більшість web-сторінок містять опис розмітки на мові HTML (або XHTML). Мова HTML інтерпретується браузерами; отриманий в результаті інтерпретації форматований текст відображається на екрані монітора комп'ютера або

мобільного пристрою. Інформація на сайті, спосіб її подання та оформлення залежать виключно від розробника і тих цілей, які він перед собою ставить.

Мова гіпертекстової розмітки HTML був розроблений британським вченим Тімом Бернерс-Лі приблизно в 1986-1991 роках в стінах ЦЕРНу в Женеві в Швейцарії. HTML створювався як мова для обміну науковою і технічною документацією, придатний для використання людьми, які не є фахівцями в області верстки. HTML успішно справлявся з проблемою складності SGML шляхом визначення невеликого набору структурних і семантичних елементів - дескрипторів. Дескриптори також часто називають «тегами».

На даний момент випущено п'ять версій мови. Перша версія була розроблена між 1986 і 1991 роками, а остання (5.2) - в 2017. Спочатку він повинен був стати незалежним від будь-яких платформ - відображатися скрізь однаково. Але цього не сталося, тому що у користувачів росли вимоги до мультимедіа.

Як підсумок, код інтерпретується по-різному не тільки на різних пристроях, а й в різних браузерах. Це не погано, тому що вимоги відрізняються: користувач, який заходив на сайт з телефону, готовий до обмеженого функціоналу і мінімалістичному дизайні, а власник ПК часто хоче більшого.

У п'ятій редакції HTML став трохи більш незалежним від інших мов. Раніше перевірка правильного заповнення форм була виключно на JS, а тепер частина завдань можна перекласти на HTML.

Наприклад, можна вказати тип даних, які повинні вводитися в поле. Якщо користувач спробує ввести щось не те, у нього не вийде. Самі ж поля стали більш доброзичливими: користувач може вибрати дату, час і навіть колір, може вказати число з інтервалу, коректну адресу електронної пошти або посилання на сайті.

З'явився вбудований плеєр - тепер не потрібно підключати Adobe Flash Player або інші плагіни. Візуальна частина перетягування елементів тепер реалізується на HTML і CSS, якщо додати атрибут draggable.

CSS (Cascading Style Sheets, каскадні таблиці стилів) - мова опису зовнішнього вигляду HTML-документа. Це одна з базових технологій в сучасному інтернеті. Практично жоден сайт не обходиться без CSS, тому HTML і CSS діють в єдиній зв'язці.

Мовою HTML ми створюємо розмічений текст - документ з гіперпосиланнями, таблицями, маркованими списками, різними зображеннями шрифтів, заголовками, підзаголовками і так далі. Отримуємо «простирадло» тексту з таблицями та ілюстраціями. Інтернет винайшли вчені, і для них такий стан речей було прийнятним. Але все змінилося, коли WWW пішов в маси і свої сторінки почали створювати прості користувачі, які хотіли індивідуальності і самовираження, а також комерційні компанії зі своїми корпоративними стандартами оформлення. Загалом, web-сторінок знадобилося індивідуальне оформлення: стиль.

Тому беремо HTML-основу - і підключаємо до неї стиль CSS. За допомогою CSS красиво оформляємо існуючий текст, тобто прописуємо унікальні властивості елементів HTML.

JavaScript - це мова скриптів, на якому тримається весь frontend web-розробки. Він дозволяє перехоплювати події і виконувати різні дії. Наприклад, користувач клікнув по якійсь кнопці - спрацювала подія click. І, зв'язавши з кліком, ми можемо виконати потрібну нам функцію - відкрити модальне вікно або змінити колір елемента.

JavaScript використовується для того, щоб робити сторінки інтерактивними, тобто дати користувачеві можливість взаємодіяти з елементами. Коли сторінки можуть реагувати на якісь дії, це робить їх цікавіше. Якщо, звичайно, не намішано багато несмаку.

На JavaScript можна працювати з анімацією, проводити обчислення, малювати на полотні (canvas), створювати cookies та багато іншого.

Одна з найпопулярніших бібліотек - це jQuery. Її варто вивчити бо загалом це бібліотека з реалізацією функцій, які зустрічаються у житті розробників дуже часто, і попросту не хочеться витрачати час на кодування

простих задач, які ти нещодавно вирішив і вони в більшості своїй схожі один на одну на 98%, ще й розібратися зможе навіть новачок.

Зараз PHP - одна з найпопулярніших мов web-розробки. Майже весь Facebook написаний на PHP. З інших прикладів - WordPress, Wikipedia, Yahoo і Tumblr.

PHP - це скриптова (сценарна) мова загального призначення. На сценарних мовах пишуть сценарії або скрипти - програми, які автоматизують деякі завдання . Скрипти допомагають уникнути помилок, заощадити час користувача і змінити програму, не боячись, що все інше перестане працювати. На відміну від більшості мов, скриптовій мові не потрібна компіляція, і вони використовуються в основному для невеликих рутинних задач.

PHP створювався для web-розробки, і для цього він в основному і використовується, причому в основному для backend-розробки, тобто розробки серверної частини сайту. На ньому часто пишуть динамічні сторінки і невеликі web-програми. PHP - популярна мова для backend-розробки, у нього простий синтаксис і його легко вчити.

SQL - проста мова програмування, яка має небагато команд і якій може навчитися будь-хто. Розшифровується як Structured Query Language - мова структурованих запитів, яка була розроблена для роботи з БД, а саме, щоб отримувати/додавати/змінювати дані, мати можливість обробляти великі масиви інформації та швидко отримувати структуровану та згруповану інформацію. Є багато варіантів мови SQL, але у них всі основні команди майже однакові. Також існує багато СУБД, але основними з них є: Microsoft Access, Microsoft SQL Server, MySQL, Oracle SQL, IBM DB2 SQL, PostgreSQL та Sybase Adaptive Server SQL. Щоб працювати з SQL кодом, нам знадобиться одна з перерахованих вище СУБД. Я обрав MySQL.

MySQL - вільна реляційна система управління базами даних. Розробку та підтримку MySQL здійснює корпорація Oracle, яка отримала права на торгову марку разом із поглиненою Sun Microsystems, яка раніше придбала шведську компанію MySQL AB. Продукт розповсюджується як під GNU General Public

License, так і під власною комерційною ліцензією. Крім цього, розробники створюють функціональність на замовлення ліцензійних користувачів. Саме завдяки такому замовленню майже в ранніх версіях з'явився механізм реплікації.

MySQL є рішенням для малих та середніх додатків. Входить до складу серверів WAMP, AppServ, LAMP та в портативні зборки серверів Денвер, XAMPP, VertrigoServ, Open Server Panel. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або віддалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції лише на рівні окремих записів. Понад те, СУБД MySQL постачається зі спеціальним типом таблиць EXAMPLE, демонструючим принципи створення нових типів таблиць. Завдяки відкритій архітектурі та GPL-ліцензування, в СУБД MySQL постійно з'являються нові типи таблиць.

Lavarel - це web-фреймворк PHP з відкритим вихідним кодом для розробки web-додатків на основі Symfony, які слідують за архітектурою модель - представлення - контролер (MVC). Він пропонує модульну пакувальну систему із спеціальним менеджером залежностей. Laravel також надає своїм користувачам кілька способів доступу до реляційних баз даних, а також утиліт для обслуговування та розгортання програм. Laravel має ліцензію MIT і вихідний код розміщено на GitHub.

Висновки до розділу 2

В ході написання другого розділу був проведений аналіз методів для вирішення задачі. Було проведено дослідження анатомії рекомендацій, що це таке, які цілі у рекомендацій, що може буди джерелом, та було досліджено декілька алгоритмів для рекомендацій, а саме Summary-based (неперсональні), Content-based (моделі, засновані на описі товару), Collaborative Filtering

(колаборативна фільтрація), Matrix Factorization (методи засновані на матричному розкладанні) та деякі інші. Засновуючись на проведеному аналізі було вирішено скористатись Колаборативною фільтрацією (user-based). Цей клас систем почав активно розвиватися у 90-ті роки. У рамках підходу рекомендації генеруються виходячи з інтересів інших схожих користувачів. Такі рекомендації є результатом «колаборації» багатьох користувачів. Звідси й назва методу. Також були досліджені методи реалізації задачі. Проведоно дослідження мов програмування для серверу, для клієнтської сторони, тобто те що бачить користувач на сторінках та програмування і управління базою даних. Було дослідження реляційних систем управління базами даних. Проведено вивчення і дослідження існуючих СУБД. В результаті був приблизно сформований список того, що знадобиться для розробки системи та був обраний алгоритм для створення на його основі модулю рекомендацій.

3 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Порівняння фреймворків для серверної розробки.

PHP — одна з найпопулярніших мов програмування. Його активно використовують великі проекти, наприклад, Facebook та «ВКонтакті». На PHP написані найпопулярніші системи управління контентом (CMS), у тому числі WordPress. На цьому движку працює близько третини всіх сайтів в інтернеті та близько 60% сайтів на CMS.

PHP розвивається, а версія 8 зробила цю мову стабільною. У такому разі, навіщо потрібні фреймворки і чому їх часто використовують замість нативної мови? Ось кілька причин:

1. Фреймворки прискорюють розробку. Наприклад, PHP-фреймворк позбавляє вас необхідності писати запити до баз даних. У фреймворках реалізовані базові функції CRUD, які необхідні роботи з базами даних.
2. Написані на фреймворках програми легко масштабуються.
3. Підтримувати легше проект на фреймворку, ніж на нативному PHP. Код фреймворків короткий, тому з ним легко працювати.
4. У PHP-фреймворках використовується модель MVC. Вона спрощує розробку.
5. Програми на фреймворках краще захищені, ніж програми на чистому PHP.
6. У фреймворках реалізовано принцип DRY. Це дозволяє розробникам писати менше за код.

Нативний PHP дозволяє робити програми. Але перерахованих вище переваг достатньо, щоб звернути увагу на фреймворки.

Як вибрати PHP-фреймворк

Вибрати фреймворк допоможуть відповіді на такі запитання:

1. Яка функціональність у фреймворку, чи відповідає вона потребам проекту?

2. Наскільки складно вивчати вибраний фреймворк?
3. Чи легко масштабувати проект, створений за допомогою цього фреймворку?
4. Наскільки активно розвивається інструмент?
5. Чи є гарантована довгострокова підтримка (LTS)?
6. Чи є активна спільнота?

Symfony, Laravel та Yii2

Перед зануренням у деталі коротко розглянемо основні особливості найпопулярніших PHP-фреймворків. Це Symfony, Laravel та Yii2.

Symfony

Symfony є набір PHP-компонентів, які підходять для повторного використання. Фреймворк дозволяє робити масштабовані та продуктивні програми. API Symfony інтегрується зі сторонніми програмами, а також із інструментами для фронтенд-розробки, наприклад, Angular JS. Symfony використовують багато популярних проектів, наприклад, Drupal і phpBB.

Laravel

Станом на середину лютого 2022 року це найпопулярніший PHP-фреймворк у світі. Поточна стабільна версія - 9. Популярність Laravel підкреслює наступний факт: багато хостерів пропонують спеціальні рішення для додатків, створених за допомогою цього фреймворку.

Yii2

Yii був представлений у 2008 році. Це безпечний, швидкий та продуктивний фреймворк для розробки web-додатків. Поточна версія - 2.0.19.

У Yii2 використовується пакетний менеджер Composer для керування залежностями. Завдяки лінивому завантаженню Yii2 вважається найшвидшим PHP-фреймворком.

Ще одна особливість Yii2 – інтеграція з jQuery. Завдяки цьому фронтенд-розробникам зручно працювати з програмами, створеними на Yii2. Як і в Symfony, Yii2 використовуються готові компоненти. Це пришвидшує розробку.

Шаблонізатори

Шаблонізатори прискорюють розробку та спрощують створення фронтенду програми. Наприклад, за допомогою шаблонизаторів вирішується завдання автоматичного екранування HTML.

Symfony

У Symfony за замовчуванням використовується Twig. Це обробник шаблонів, який дозволяє писати чистий код та розширює можливості нативного PHP. Наприклад, Twig спрощує створення послідовностей, що екранують.

Laravel

У цьому фреймворку застосовується шаблонизатор Blade. Він дозволяє використовувати код PHP у уявленнях. Blade практично не впливає на швидкість роботи програм, оскільки уявлення зберігаються в окремих файлах з розширенням .blade.php. Код уявлень перетворюється на нативний PHP.

Yii2

Цей фреймворк не використовує стандартні сторонні шаблонизатори. Але розробник може вибирати інструменти в залежності від розв'язуваних завдань. До рекомендованих шаблонізаторів входять Twig і Smarty.

Проміжний висновок: за цим критерієм чистого переможця немає. Усі фреймворки підтримують роботу з шаблонизаторами, що прискорює розробку фронтенду додатків. Невелику перевагу має Yii2, тому що в ньому не використовується будь-який шаблонізатор за замовчуванням.

Встановлення

Кожен фреймворк підтримує кілька варіантів встановлення. Наприклад, Symfony, Laravel та Yii2 можна встановити за допомогою пакетного менеджера Composer. Усі фреймворки після встановлення дозволяють працювати з шаблонною програмою.

Проміжний висновок: за критерієм простоти встановлення переможців немає, кожен із трьох інструментів легко встановлювати.

Швидкість розробки

Symfony вважається надійним фреймворком, за яким стоїть чисельна та активна спільнота. Laravel швидко розвивається та утримує перше місце у

списку найпопулярніших фреймворків. Yii2 забезпечує продуктивність додатків.

Якщо вам потрібно якнайшвидше створити веб-додаток, і ви ніколи не працювали з PHP-фреймворками, вибирайте Laravel. Його найпростіше вивчати, і в мережі найбільше посібників саме по Laravel.

Продуктивність

На думку багатьох людей, які займались дослідженнями продуктивності фареймворків, найпродуктивнішим фреймворком є Yii2. Це оптимальний вибір для створення високонавантажених програм.

Продуктивність Laravel – дискутабельне питання. За цим критерієм він поступається Yii2 і Symfony. Проте в мережі можна знайти багато рекомендацій щодо прискорення програм на Laravel.

Чому Laravel фаворит?

Інформативна документація

Почати потрібно з особливості Laravel, з якою неминуче стикаються всі розробники при освоєнні нової технології. Це документація Laravel, яка, на думку великої кількості розробників, є дуже гарною та структурованою. Це також додає популярності цьому двигуну серед розробників.

Особливо було приємно вивчати Laravel docs після спроб подружитися з Yii2, у якого мануали є якоюсь суцільною розповіддю щодо створення сайту на базі фреймворку з поодинокими згадками деяких конструкцій та принципів їх роботи.

У Laravel документації таких проблем немає – кожній конструкції та процесу присвячена окрема стаття.

Оскільки у даного PHP фреймворку маса послідовників по всьому світу, то в мережі можна знайти безліч різних спільнот та користувальницьких перекладів статей.

MVC структура коду

Структура коду Laravel framework відповідає популярному патерну проектування MVC, тобто. у ньому можна назвати моделі (models), уявлення

(views) і контролери (controllers).

Даний шаблон проектування зарекомендував себе як перевірене часом розв'язання ефективної структури додатків, насамперед у сфері веб, що дозволяє відокремити логіку програми від його візуальної частини.

MVC дозволяє робити код більш читабельним, а процес розробки комфортним, розмежовуючи роботу frontend- та backend-розробників.

Artisan

Artisan – це консоль Laravel, в арсеналі команд якої є робота з міграціями, контролерами та моделями, авторизацією та іншими базовими компонентами фреймворку.

Свого роду контроль версій для структури таблиць БД. Кожен файл міграції містить структуру таблиць, або зміни її структури. Тобто. Процес створення нових сутностей БД в Laravel фреймворку являє собою створення міграції та запуск її за допомогою спеціальних консольних команд artisan. Також у наявності фреймворку є безліч методів для маніпуляцій міграціями: відкат окремих, скидання всіх і т.д.

```php
public function up()
{
    Schema::create( table: "users", function (Blueprint $table){

        $table->id( column: "id");
        $table->string( column: "email")->unique( indexName: "email");
        $table->string( column: "password");
        $table->string( column: "phone")->unique( indexName: "phone");
        $table->string( column: "name");
        $table->string( column: "surname");
        $table->string( column: "country");
        $table->string( column: "region");
        $table->string( column: "city");
        $table->foreignId( column: "role_id")->default( value: 1)
            ->references( column: "id")->on( table: "roles")->onDelete( action: "cascade");
        $table->foreignId( column: "status")->default( value: 1)
            ->references( column: "id")->on( table: "statuses")->onDelete( action: "cascade");
        $table->string( column: "remember_token",  length: 100)->nullable( value: true);
        $table->timestamps();

    });
}
```

Рисунок 3.1 – код з файлу міграції таблиці користувачів.

Twitter Bootstrap та jQuery з коробки

Після встановлення Laravel фреймворку в розпорядженні розробника знаходяться файли app.js і app.css, які являють собою скомпоновані та мінімізовані jQuery та BootStrap останніх версій на момент виходу релізу Laravel.

Отже, підключати їх вручну ще раз немає сенсу. Якщо вас, звичайно, влаштує подібне використання даних пакетів і ви не волієте використовувати будь-які збирачі пакетів, наприклад, WebPack.

Для роботи з ним у Laravel framework, до речі, є спеціальний інструмент.

Laravel Mix

Даний пакет є надбудовою над згаданим WebPack, що дозволяє розділяти css і js код на окремі модулі, конфігурувати їх використання, налаштовувати мініфікацію та використання css-препроцесорів (sass, less, stylus і т.д.).

Реєстрація та аутентифікація

З коробки Laravel надає механізм реєстрації та авторизації користувачів, що полегшує життя розробникам, дозволяючи не винаходити чергові велосипеди.

Валідатори

Згадайте часи, коли код був написан на чистому PHP і робилася перевірка на існування в БД значення, що вводиться з форми, шляхом ручного запиту в основу для уникнення дублювання.

Laravel валідатори – це конструкції, що дозволяють проводити перевірку даних на основі різноманітних готових правил. Також Laravel дозволяє створювати власні правила, повідомлення про помилки та індивідуальні валідатори загалом.

Eloquent ORM

ORM – це технологія програмування, покликана полегшити програмістам роботу з БД шляхом надання методів API для типових операцій (вибірка, додавання, оновлення, видалення тощо.).

Реалізацій ORM існує безліч, але автори Laravel і тут заморочилися,

придумавши свою.

Механізм черг

У Laravel з коробки є інструменти організації черг процесів (наприклад, для масового відправлення email). Ця функція є незамінною для HighLoad-проектів, т.к. дозволяє розвантажити сервер від постійної роботи.

Інтерфейс для Cron задач

Laravel надає набір методів для створення та управління задачами, що виконуються за допомогою планувальника задач Cron.

Зручний дебаггінг коду та тестування

Реалізується наявністю debug panel, спеціальної функції dd() для виведення даних на екран (аналог PHP-конструкції echo ‘<pre>’; print_r($var); die();) та докладним логом програми.

Що стосується тестування, то в Laravel механізм написання юніт-тестів з використанням популярного тестувального фреймворку PHPUnit доступний із коробки.

Крім того, в процесі тестування працездатності ресурсів у Laravel фреймворку є можливості емуляції відвідування сторінок сайту та різних дій (натискання на посилання, кнопки, введення тексту тощо) завдяки використанню компонентів Symfony.

Можливість розширення базового функціоналу

Оскільки фреймворк - це просто набір інструментів для ефективного написання коду, то всі банальні функції, які в CMS, наприклад, додаються в два рахунки шляхом установки готових модулів, у разі використання фреймворків доводиться щоразу писати руками. Щоб уникнути цієї рутинної роботи, розробники Laravel ввели можливість розширення функціоналу базового додатку за рахунок встановлення пакетів, які є аналогами модулів для CMS.

Кешування з коробки

Ще одна корисна фіча, без якої неможлива технологія повноцінного HighLoad-ресурсу. Причому, кешування в Laravel є за допомогою різних технологій: Redis, MemCached і т.д. за допомогою відповідних драйверів та

пакетів. За замовчуванням доступний драйвер кешування файлу, завдяки якому закешована інформація зберігатиметься у файловій системі.

Зручний механізм роутингу

Маніпуляції з URL, доступними на сайті, в Laravel неймовірно прості та зручні. Все, що потрібно зробити для додавання Laravel маршрутів – це відредагувати файл routes/web.php. Найпростіше додавання нового роуту виглядає так:

```
Route::get('/', function () {
    return view('welcome');
});
```

Цей код виводитиме на екран зміст файлу resources/views/welcome.blade.php при переході в корінь(головна сторінка) сайту.

Також для маршрутів можна вказувати методи контролерів і функції користувача, застосовуючи до них MiddleWare - ще одну фішку Laravel, що є прошарок між маршрутом і дією при його виконанні.

Висновок який можна зробити проанализувавши інформацію:

Laravel зручне середовище розробки, оскільки має велику екосистему і велике спільнота як англомовна, так і україномовна. У чатах та форумах завжди можна знайти відповіді на тему можливостей фреймворку або вирішення типових завдань. Також він дозволяє закривати більшість типового функціоналу сайту: реєстрація, авторизація (як базова, так і через соцмережі), ролей користувачів, нотифікації тощо.

Створено вже понад 50 000 пакетів із готовими рішеннями, що значно прискорює та спрощує роботу розробника. Все що залишається – це сконцентруватися лише на особливостях проекту. Це сприяє досить високій швидкості розробки.

3.2 Проектування інформаційної системи

Проектування — це процес визначення архітектури, компонентів, інтерфейсів та інших характеристик системи або її частини (ISO 24765).

Результатом проектування є проект - цілісна сукупність моделей, властивостей або характеристик, описаних у формі, придатній для реалізації системи.

Проектування, поруч із аналізом вимог, є частиною великий стадії життєвого циклу системи, званої визначенням системи (англ. system definition). Результати цієї стадії є вхідний інформацією стадії реалізації (втілення) системи (англ. system realization).

Проектування системи спрямоване на уявлення системи, що відповідає передбаченій меті, принципам та задумам; воно включає оцінку та прийняття рішень щодо вибору таких компонентів системи, які відповідають її архітектурі та укладаються в запропоновані обмеження.

Жоден сайт не може бути зробленим без сторінок. Інтернет магазин, як правило складається мінімум с декількох сторінок. Усе залежить від масштабів сайту. Для масштабу цього проекту далі буде наведено структуру сайту.

Сайт буде складатися з таких сторінок як:

1. Головна сторінка
2. Сторінка окремої книги
3. Сторінка з результатами пошуку
4. Сторінка с книгами окремого автору
5. Сторінка для відправлення повідомлень
6. Панель адміністратора.

Панель адміністратора також складається з декількох сторінок:

1. Сторінка зі списком усіх користувачів
2. Сторінка з усіма книгами
3. Сторінка для додавання нової книги
4. Сторінка для редагування книги
5. Сторінка для переглядів повідомлень
6. Сторінка для додавання нових категорій
7. Сторінка для редагування вже існуючих категорій

Все це не може працювати без бази даних. Тому для проекту планується створити базу даних яка буде містити потрібні нам дані.

Уся логіка буде виконуватись в контролерах. Для кожної сторінки будуть свої методи, які будуть розподілені в окремих контролерах, які відповідатимуть за певні розділи сайту.

Висновки до розділу 3

В ході написання третього розділу був проведений аналіз та порівняння характеристик фреймворків для серверної сторони сайту, з ціллю визначити кращий варіант якій підійде для успішного виконання поставленої задачі. В ході дослідження фреймворки оцінувалися за низкою критеріїв, за деякими з критеріїв переможця було визначити важко, але він все ж таки знайшовся. Це фрейморк Laravel. Також було спроектовано і коротко розписано з чого буде складатися сайт, які він матиме сторінки і відповідно функціонал.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА РОЗРОБКА ДОКУМЕНТАЦІЇ

4.1 Опис програмної реалізації

Було вирішено розпочати розрубку із створення бази даних і заповнення її тестовими даними, щоб створенні потім сторінки не були пустими і відображали потрібну нам інформацію.

Для роботи з СУБД MySQL було завантажено і встановлено на ПК Open Server Panel с офіційного сайту. Open Server Panel — це портативне програмне середовище, створене спеціально для веб-розробників з урахуванням їх рекомендацій та побажань.

Даний програмний комплекс включає ретельно підібраний набір серверного програмного забезпечення, а так само неймовірно зручну і продуману керуючу утиліту, яка володіє потужними можливостями з адміністрування і налаштування всіх доступних компонентів.

В це програмне середовище за замовчування входить phpMyAdmin — веб-застосунок з відкритим кодом, написаний мовою PHP і який є веб-інтерфейсом для адміністрування СУБД MySQL.

Для роботи з Laravel, була створенна база даних з рекомендованим типом кодування

Рис 4.1 – Ім'я БД та кодування

У базі даних знаходиться декілька таблиць, кожна з яких називається відповідно до того, які данні зберігаються в цій таблиці.

Перша таблиця яка була створена це таблиця ролей. Вона містить у собі лише 2 поля: найменування ролі та ідентифікатор цієї ролі. Ця таблиця призначена для того, щоб користувачам, які будуть реєструватися на сайті були надані ролі для не допуску певних користувачів до певних розділів сайту. Наприклад звичайний користувач не має прав на вхід до адміністративної панелі з можливостями редагувати інформацію про книги та читати

повідомлення. Для цього власник сайту сам призначить адміністраторів.

Другою була створена таблиця статусів. В неї така ж структура як і в таблиці ролей: ідентифікатор і найменування статусу. Їх усього два, які відповідають на питання користувач активний чи ні. Якщо виражатись більш зрозуміло, то вони позначають чи заблокован користувач на сайті. Наприклад можуть заблокувати за нецензурні коментарі або за спам.

Далі була створена таблиця для зберігання даних користувачів яка має таку структуру:

- Ідентифікатор
- Електронна адреса
- Номер телефону
- Ім’я
- Прізвище
- Країна
- Регіон
- Місто
- Ідентифікатор ролі (якій посилається на таблицю ролей)
- Ідентифікатор статусу (якій посилається на таблицю статусів)

Наступною була створена таблиця категорій. Вона булла створена для зберігання даних про категорії, які використовуються для

Також потрібно створити таблицю, яка буде зберігати дані про книги.

Вона міститиме у собі ідентифікатор книги, посилання на категорію, до якої відноситься книга, назву, короткий опис книги, автора книги, ціну, кількість переглядів, та дату коли книга була додана до бази даних.

Також була створена таблиця для зберігання повідомлень, які користувачі мають можливість залишити на окремій сторінці через спеціальну форму. Як і у всіх інших таблицях, ця містить поле ідентифікатор. Також у цій таблиці зберігається електронна адреса, телефон, ім’я та повідомлення користувача.

Була створена і таблиця для зберігання коментарів. Вона містить у собі

ідентифікатор коментаря, посилання на ідентифікатор книги до якої був написаний цей коментар, посилання на ідентифікатор користувача, якій залишив цей коментар, ім’я користувача, його номер телефону, електрону адресу та коментар якій він залишив.

І наостанок, була створена таблиця для зберігання даних про вподобання користувачі. Таблиця містить у собі ідентифікатор запису, посилання на ідентифікатор користувача и посилання на ідентифікатор книги, яку вподобав користувач.

Завдяки можливостям фреймворку, не обов’язково створювати базу вручну у за стосунку phpMyAdmin. Для цього можна використати файли міграцій, де прописується код для створення таблиць.

```
public function up()
{
    Schema::create( table: 'categories', function (Blueprint $table) {
        $table->id();
        $table->string( column: 'name')->unique( indexName: 'name');
        $table->string( column: 'img_path')->default( value: '/');
        $table->timestamps();
    });
}
```

Рисунок 4.1 – Приклад файлу за інструкціями для міграції.

У цих файлах і були прописані усі інструкції для міграцій у базу даних. Але щоб усе правильно і нормально працювало потрібно налаштувати файл .env з параметрами, який використовує Laravel, зокрема для з’єднання з БД. Він знаходиться в корні згенерованого проекту Laravel.

```
DB_CONNECTION=mysq
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=larave
DB_USERNAME=root
DB_PASSWORD=
```

Рисунок 4.2 – Інформація про БД з файлу .env

Розробку слід продовжити зі створення сторінок, паралельно створюючи

методи в контролерах для коректної роботи і відображення даних на сторінках. Перша сторінка яку слід розробляти, це головна сторінка, або як її ще називають – корінь сайту, або домашня сторінка. Розберемось що буде відображатись на головній сторінці. По перше на сторінці будуть секції, які повторюються і на інших сторінках. Це хедер та футер сайту. У хедері знаходиться логотип, поле для введення тексту запиту для пошуку і посилання на головну сторінку, сторінку для авторизації і сторінку для відправки повідомлень.

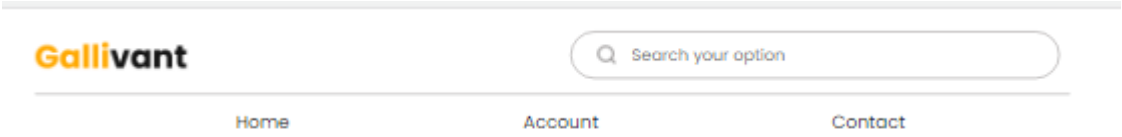


Рисунок 4.2 – Хедер сайту

Далі за хедером розташований слайдер зі списком категорій.

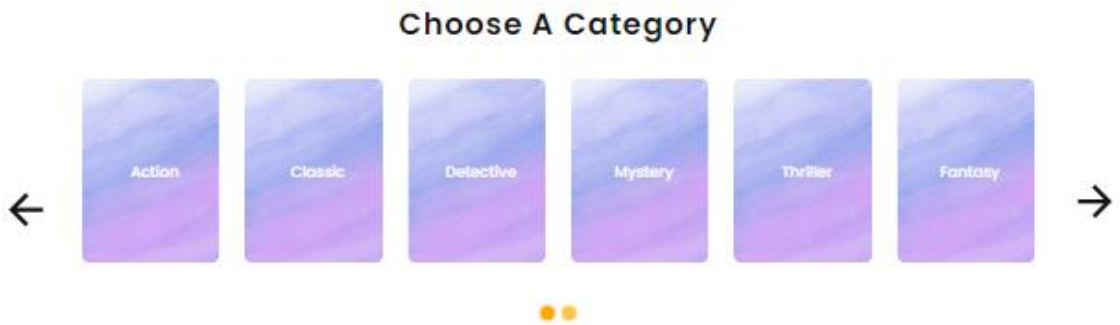


Рисунок 4.3 – Слайдер зі списком категорій.

Для використання слайдеру була підключена бібліотека Swiper

Swiper - це найсучасніший безкоштовний сенсорний слайдер для мобільних пристроїв з апаратним прискоренням переходів і приголомшливою власною поведінкою. Він призначений для використання на мобільних веб-сайтах, мобільних веб-додатках та мобільних нативних/гібридних програмах. Swiper сумісний не з усіма платформами, це сучасний сенсорний слайдер, орієнтований тільки на сучасні програми/платформи, щоб забезпечити найкращий досвід та простоту.

Для відображення категорій було створено модель Category через яку можно звертатись до таблиці категорій. Laravel включає Eloquent, об'єктно-реляційний перетворювач (ORM), який робить взаємодію з вашою базою даних

приємною. У разі використання Eloquent кожна таблиця бази даних має відповідну «Модель», яка використовується для взаємодії з цією таблицею. Крім вилучення записів з таблиці бази даних, моделі Eloquent також дозволяють вставляти, оновлювати та видаляти записи з таблиці.

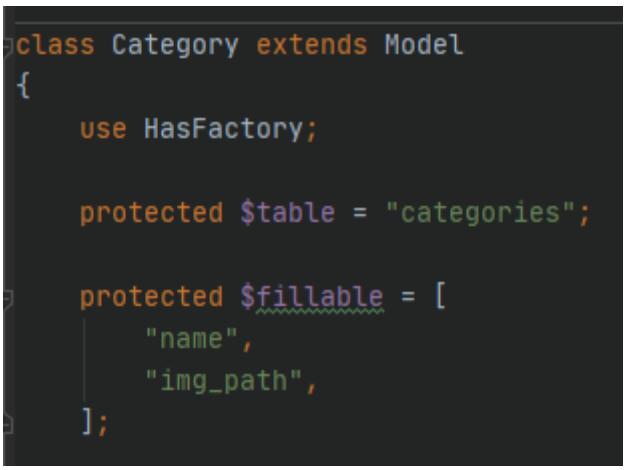


Рисунок 4.4 – Приклад моделі Category

Для відображення категорій ми звернулись до контролера якій відповідає за головну сторінку, у ньому є виклик сервісу в якому є звернення до моделі Category з цілю отримати всі дані.

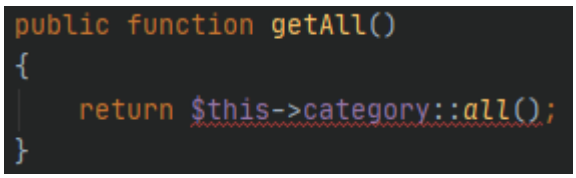


Рисунок 4.5 – Метод для отримання усіх категорій

І відображається головна сторінка із змінною в яку записані категорії. Для того щоб вони були відображені потрібно у шаблоні сторінки перебрати зміну яка прийшла з контролера і в потрібні нам місця вставити дані із змінної.

```
<div class="swiper-wrapper">
    @foreach($categories as $category)
        <div class="swiper-slide">
            <a href="#">
                <div class="slider-card">
                    <div class="slider-img">
                        <a href="{{route('category', ['id' => $category->id])}}">
                            <img src="{{asset('storage/'.$category->img_path)}}"
                                 alt="Category">
                        </a>
                    </div>
                    <div class="centered">{{__($category->name)}}</div>
                </div>
            </a>
        </div>
    @endforeach
</div>
```

Рисунок 4.6 – Відображення категорій на головній сторінці

Кожна з карток окремої категорії веде перекидає користувача на сторінку з книгами із цієї категорії за кліком по картці. Як можна побачити на цій конструкції в атрибуті якій відповідає за посилання на яке буде направлено користувача написана функція яку надає фреймворк. Вона переміщає нас за вказаним маршрутом використовуючи ім’я маршруту який прописано у файлі routes/web.php з параметром – ідентифікатор категорії, завдяки якому ми можемо взяти з бази даних усі пости які належать до цієї категорії.

```
<a href="{{route('category', ['id' => $category->id])}}">
```

Рисунок 4.7 – Посилання на сторінку категорії

Якщо перейти на сторінку, наприклад, категорії «Детективи», то буде відображена ось така сторінка:

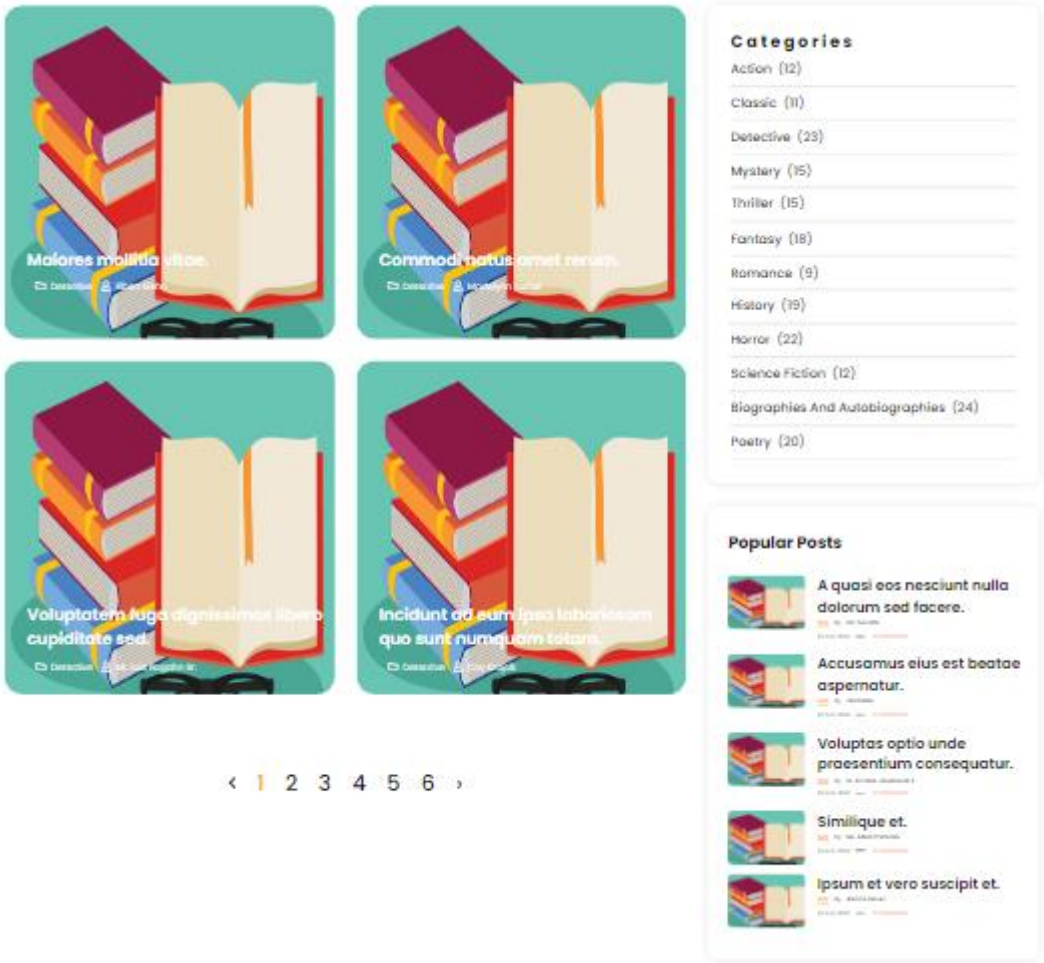


Рисунок 4.8 – Сторінка книжок за категорією

Тут ми можемо побачити книжки з цієї категорії за пагінацією. Пагінація надається фреймворком з коробки і її дуже просто використовувати. Також можна побачити список з усіма категоріями і кількістю книжок в кожній категорії. І список популярних постів цієї категорії, якій відображається тільки для неавторизованих користувачів.

Для того щоб вивести на сторінку усі книжки з пагінацією ми, як і з категоріями звертаємось до відповідної моделі передаючи параметр – ідентифікатор категорії, у цьому випадку до моделі Book, і робимо запит $this->book::where("category_id", $id)->paginate(4), де в таблиці book шукаємо кніжки, у яких ідентифікатор категорії (category_id) відповідає переданому параметру ідентифікатора категорії ($id).

```
@foreach($books as $key => $book)
    <div class="col-lg-6 col-md-10 ">
        <div class="random">
            <div class="random__post">
                <div class="random__card">
                    <a href="{{route('single', ["id" => $book->id])}}">
                        <img src="{{asset("storage/".$book->img_path)}}" alt=""></a>
                </div>
                <div class="random__text">
                    <div class="random__title">{{$book->title}}</div>
                    <div class="random__data">
                        <div class="random__category-name">
                            <a href="{{route('category', ['id' => $book->category_id])}}">
                                <i class="far fa-folder icon-folder-search"></i>
                                {{$book->category->name}}</a>
                        </div>
                        <div class="random__author">
                            <a href="{{route('author', ['author' => $book->author])}}">
                                <i class="far fa-user icon-user-search"></i>{{$book->author}}
                            </a>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
@endforeach
```


Рисунок 4.9 – Відображення книжок на сторінці

```
{{ $books->links() }}
```

Рисунок 4.10 – Команда для відображення пагінації

Наступний запит поверне п’ять популярних книжок, які будть відображатись у сайдбарі з правого боку для усіх неавторизованих користувачів.

$this->book::where("category_id", $id)->orderBy("views", "desc")->limit(5)->get();

Рисунок 4.11 – Популярні книги у сайд барі

Логіка для отримання популярних книжок дуже проста. При відвідуванні сторінки книжки, тобто при кожному виклику методу, який відповідає за відображення цієї сторінки поле в таблиці книжок, яке відповідає за кількість переглядів простим набором команд інкрементується, тобто збільшується на один перегляд. І потім книжки сортуються за переглядами від найбільшого до найменшого і з гори списку обирається потрібна кількість. Book::where("id", $book_id)->first()->increment("views", 1);

І також у сайд барі відображен список усіх категорій з кількістю книжок у кожній з них. Для цього також була використана модель Category, але тепер були використанні відношення Eloquent які надає нам фреймворк.

Таблиці бази даних часто пов'язані один з одним. Наприклад, повідомлення в блозі може мати багато коментарів або замовлення може бути пов'язане з користувачем, який його розмістив. Eloquent спрощує управління цими відносинами та роботу з ними та підтримує безліч спільних відносин. Відносини Eloquent визначаються як методи у класах моделей Eloquent. Оскільки відносини також служать потужними розробниками запитів, визначення відносин як методів забезпечує потужні можливості створення ланцюжків методів і запитів.

Для цього в класах моделей Category і Book потрібно написати певні методи. В моделі Category написана наступна функція, яка вказує на те, що ця модель має багато книжок.

```
public function books()
    {
        return $this->hasMany(Book::class);
    }
```

А в моделі Book написана функція, яка вказує на те, що кожна книжка належить до певної категорії, і може бути відстежень за зовнішнім ключем category_id за таблиці books в базі даних.

```
public function category()
    {
        return $this->belongsTo(Category::class, "category_id");
    }
```

```
<div class="categories__list">
    @foreach($categories as $category)
        <div class="category-item">
            <div class="category-item__name">
                <a href="{{route('category', ['id' => $category->id])}}">{{$category->name}}</a>
            </div>
            <div class="category-item__posts">({{$category->books->count()}})</div>
        </div>
        <div class="category__hr"></div>
    @endforeach
</div>
```

Рисунок 4.12 – Відображення категорій у сайд барі

Хочу звернути увагу на цей код: {{$category->books->count()}}. Це і є ті самі відношення між моделями. Категорія з моделі Category викликає метод books щоб з бази даних дістати усі книги цієї категорії і підрахувати їх кількість.

Рисунок 4.12 – Список категорій у сайд барі з підрахованою кількістю книжок з цієї категорії.

Повертаючись на головну сторінку, також можна побачити топ 15 книжок одразу після слайдеру з категоріями.

Рисунок 4.12 – Топ 15 книжок.

Ці пости були взяті із бази даних через звернення до моделі Book де шукалися книжки, відсортовані за переглядами від більшого до меншого і забиралися за бази даних 15 перших книжок.

$this->book::orderBy("views", "desc")->limit(15)->get();

На всіх сторінках сайту є однаковий футер. Footer – футер, він же підвал сайту – це блок у нижній частині сторінки, куди виносять корисну, але не

першорядну інформацію. Як приклади можна навести:

1. Дані про копірайт – (c) Ім'я компанії 2012
2. Карта сайту
3. Дублювання основних пунктів навігації – Головна, Про нас, Контакти…
4. Хмара основних тегів сайту
5. Назва команд або компаній, які розробляли сайт, з посиланням на їх контактні дані або їхню сторінку в інтернеті

Не варто перевантажувати футер зайвими даними – більшість користувачів, швидше за все, не зверне на них уваги, а пошукові системи можуть розцінити це як методи чорного просування та застосувати до сайту санкції, прибравши його з пошукової видачі. Основні традиції із заповнення футера вже склалися, і більшість користувачів звикли бачити там перелічені вище пункти; не варто їх розчаровувати.

В моєму випадку футер виглядає ось так

Рисунок 4.13 – Футер сайту

Перейшовши на сторінку з контактною формою можна побачити контактну форму, яка має два вигляди: один призначений для відвідувачів сайту, які не мають аккаунту, і для авторизованих користувачів.

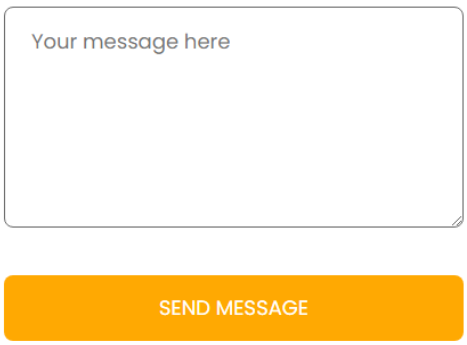


Рисунок 4.14 – Контактна форма для авторизованих користувачів

Рисунок 4.15– Контактна форма для не авторизованих користувачів

Відображати якусь із цих форм на сторінці дуже просто. У шаблонізаторі Blade фреймворку Laravel дуже зручно вставляти код в розмытку сторінки. Завдяки классу Auth, який выдповідає за аутентифікацію і має корисні методи, можна написати умовну конструкцію прямо посеред розмітки, і якщо користувач аутентифікований – відображається відповідна форма, якщо ні, то інша.

```
@if(Auth::check())
    <div class="form-input">
        <textarea required="required" id="inputMessage"
                  class="form-message" type="text" name="message"
                  placeholder="Your message here"></textarea>
    </div>
@else
    <div class="form-input">
        <input id="inputName" class="form-name" type="text" name="name"
               placeholder="Full Name">
    </div>
```

Рисунок 4.16 – Відображення контактної форми на сторінці

Дані з контактної форми відправляються на відповідний контролер за допомогою бібліотеки jQuery та AJAX і відправляється на відповідний контроллер для обробки, валідації та додавання до БД. Валідація у Laravel дуже

зручна. За допомогую вже існуючих правил валідації, можна первірити надіслані у контролер дані, наприклад на тип даних які прийшли, перевірити чи достатньо даних відправлено і чи відправлені поля з позначкою «обов'язково». Якщо валідація пройдена до дані можна записувати у бд і повертати повідомлення про успішну операцію. Якщо ні, то можна повернути текст помилки і вивести його користувачеві, щоб він виправи деякі поля, які не відповідають вимогам і спробував ще раз.

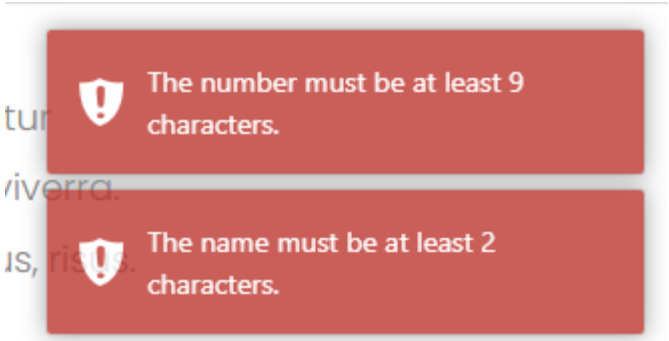


Рисунок 4.17 – Повідомлення про помилку

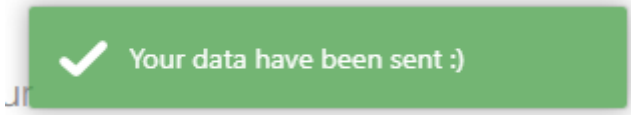


Рисунок 4.18 – Повідомлення про успішне відправлення даних до БД

```
$validated = $request->validate([
    "name" => "required|string|min:2",
    "email" => "required|string|min:12",
    "number" => "required||max:13|min:9",
    "message" => "required|string|min:3",
]);
```

Рисунок 4.19 – Валідація даних перед відправкою до БД

Як можна побачити на рисунку вище, дані які були введені у поля з контактнї форми, передаються в контролер і проходять перевірку. Вони усі повинні буди заповнені, і мають мінімальний набір символів для відправки, бо наприклад ім'я не може складатися лише з однієї літери.

У випадку з авторизованими користувачами, де у контактної форми лише одне поле для вводу повідомлення, дані про користувача беруться з активної сесії за допомогою класу Auth і методу який обирає користувача, і потім ці дані

також ідуть у контролер. Auth::user() дістає дані про користувача з сесії. І потім повідомлення також валідується перед записом у базу даних.

Перейдемо на сторінку авторизації. Вона дуже проста і звична. Потрібно ввести вашу електрону адресу яку ви вказала при реєстрації і свій пароль. Вони будуть передані у контролер, буде проведена валідація, для захисту від SQL Injection. Використання SQL-коду (англ. SQL Injection) — один із поширених засобів злому сайтів і програм, які працюють із базами даних, заснований на впровадженні запит довільного SQL-коду.

Впровадження SQL, залежно від типу використовуваної СУБД та умов впровадження, може дати можливість атакуючому виконати довільний запит до бази даних (наприклад, прочитати вміст будь-яких таблиць, видалити, змінити або додати дані), отримати можливість читання та/або запису локальних файлів та виконання довільних команд на сервері, що атакується.

Атака типу застосування SQL може бути можлива через некоректну обробку вхідних даних, що використовуються в SQL-запитах. Якщо валідація пройдена, робиться запит у БД з пошуком користувача, якому відповідають ввідні дані. Якщо все добре, стартує сесія користувача.

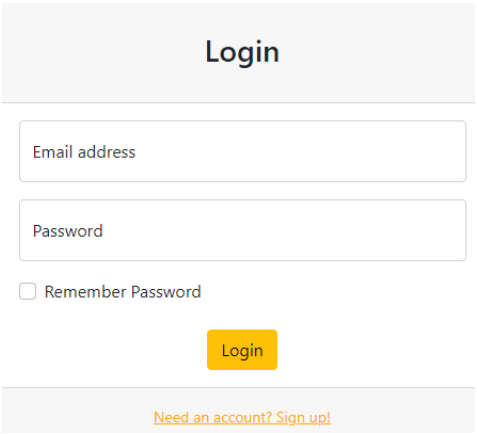


Рисунок 4.20 – Форма аутентифікації користувача

Якщо ж у вас ще немає зареєстрованого аканту, то нижче форми є посилання на форму реєстрації.

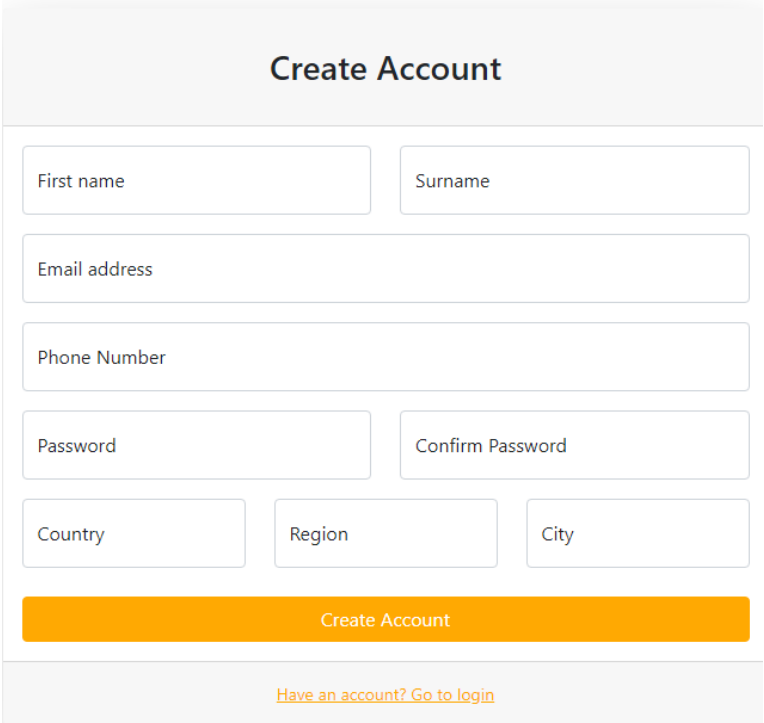


Рисунок 4.21 – Форма реєстрації користувача

Як і у випадку з формою авторизації, для реєстрації також є метод у контролері який відповідає за валідацію даних, захист від SQL Injection, та перевірку на унікальність, щоб не створити 2 користувача з однаковими електронними адресами. У випадку успішної валідації даних, и запису нового користувача у БД, новий користувач відразу буде авторизован і буде запущено його сесію.

За замовчуванням при реєстрації новим користувачам буде надаватись роль звичайного користувача. Вони не матимуть доступу до адміністративної панелі і будуть при авторизації переміщатись на головну сторінку. У випадку, якщо власником сайту або одним із адміністраторів вам було надано роль адміністратора, то після авторизації ви потрапите у панель адміністратора, де ви маєте можливість перевіряти усі повідомлення, які були залишені через форму повідомлень, додавати нові книжки, редагувати інформацію про старі книжки, або видаляти їх. Також маєте можливість редагувати користувачів, наприклад змінювати їх ролі, або блокувати за певні порушення, передбачені власником сайту.

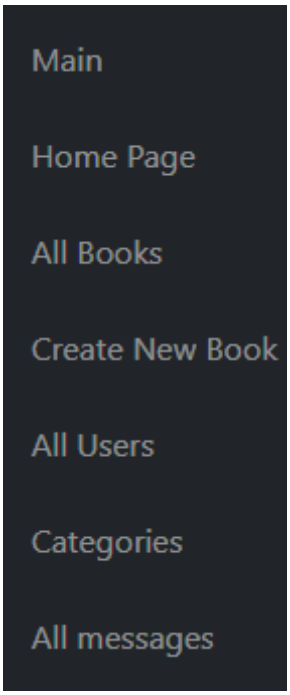


Рисунок 4.22 – Список сторінок в адміністративній панелі.

Коротко пройдемося по сторінкам. Main – це посилання на головну сторінку адміністративної панелі. Home Page – це посилання на головну сторінку сайту, або як ще кажуть – корінь сайту. All Books - це сторінка де відображається список усіх книг з двома кнопками, які відповідають за посилання на сторінку редагування книги, і на її видалення. За кожну функцію відповідає окремий метод в контролері.

Title	Author	Price	Description		
A quasi eos nesciunt nulla dolorum sed facere.	Ian Turcotte	23.9$	Nihil perspiciatis quod adipisci doloremque. Deserunt ab aperiam accusantium ea. Quaerat unde architecto ut voluptas et. Molestiae occaecati magnam voluptas.	Edit	Delete
Accusamus eius est beatae aspernatur.	Zita Ratke	23.1$	Reiciendis dicta autem veniam iure est laborum consequuntur. Magnam error qui perferendis cupiditate non. Sed officia asperiores sequi alias aut reprehenderit.	Edit	Delete
Voluptas optio unde praesentium consequatur.	Dr. Annette Jakubowski II	9.5$	Voluptatum assumenda quas ipsum excepturi culpa in. Minima magnam sapiente aut optio harum modi et. Expedita sit dolores quibusdam consequatur deserunt vitae.	Edit	Delete
Similique et.	Ms. Adela Prohaska	13$	Vitae ut aliquid repellat nostrum. Est dolores eveniet incidunt odio ut. Suscipit cum tempora assumenda.	Edit	Delete

Рисунок 4.23 – Сторінка з усіма книгами.

Create New Book посилає нас на сторінку з формою для додавання нової книги. Вона містить у собі такі поля як назва книги, автор книги, ціна книги, та список за категоріями звідки потрібно вибрати відповідну до книги категорію, та також вибрати зображення яке буде відображатись на сторінках сайту.

All Users посилає нас на список з усіма користувачами. Там також є кнопка, яка посилає нас на сторінку редагування. На цій сторінці як раз і можна надавати користувачеві роль, або змінювати його статус.

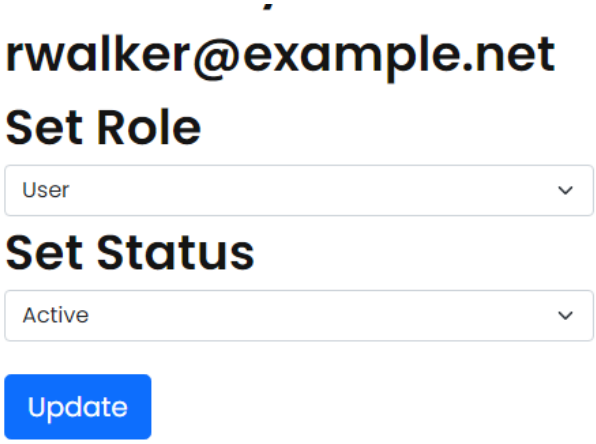


Рисунок 4.23 – Сторінка редагування користувача.

Categories – це сторінка зі списком усіх категорій, також має посилання на редагування та кнопку створення нової категорії, яка в свою чергу веде на сторінку з формою для створення категорії.

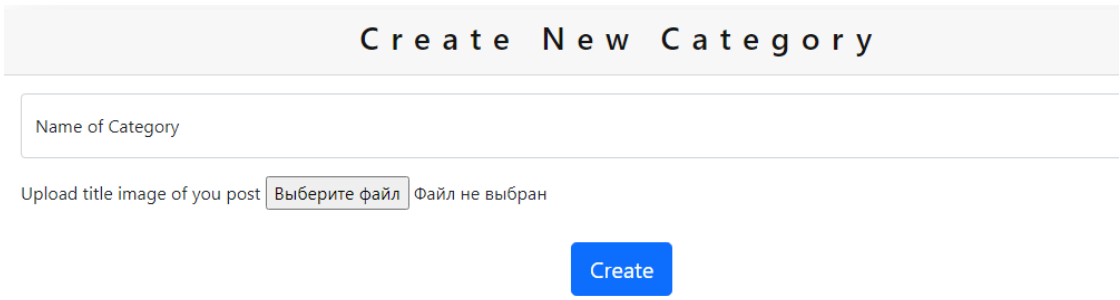


Рисунок 4.24 – Сторінка створення нової категорії

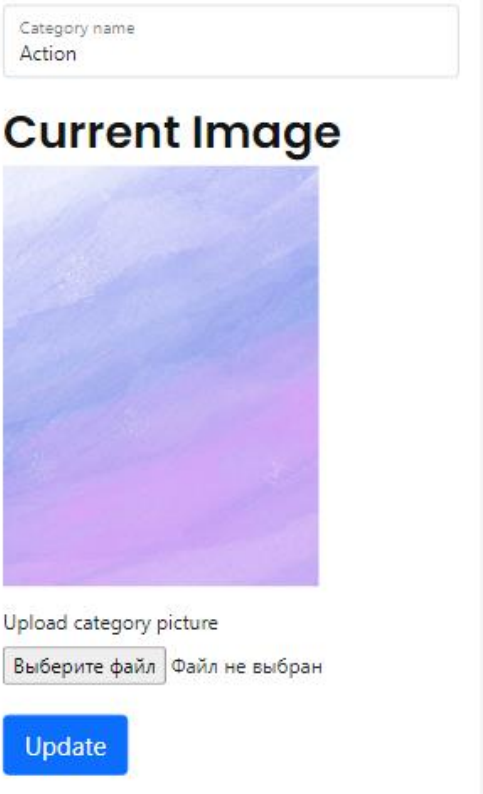


Рисунок 4.24 – Сторінка редагування вже існуючої категорії

All messages – це сторінка де можна переглянути повідомлення від користувачів.

ID	Name	Email	Message	Departure Date
1	Артем	dariy.artiem@gmail.com	hELLO!	2022-06-23 06:57:20

Рисунок 4.24 – Сторінка з повідомленнями

Усі дані при створенні категорій, або додавання нових книжок чи їх редагування, також проходять валідацію у контролерах. Для отримання даних про книжки, використовується модель Book і створюється запит на усі книжки Book::all().

```
@foreach($result as $element)
    <tr>
        <td>{{$element->title}}</td>
        <td>{{$element->author}}</td>
        <td>{{$element->price}}$</td>
        <td>{{$element->description}}</td>
        <td>
            <form action="{{route('books.edit', ['id' => $element->id])}}">
                <button class="btn btn-primary">{{__("Edit")}}</button>
            </form>
        </td>
        <td>
            <form method="POST" action="{{route('books.delete', ['id' => $element->id])}}">
                @csrf
                @method('DELETE')
                <button class="btn btn-danger">{{__("Delete")}}</button>
            </form>
        </td>
    </tr>
@endforeach
```

Рисунок 4.25 – Вивід усіх книжок в адміністративній панелі.

Як можна побачити є маршрути, якими передаються ідентифікатори окремо обраної книжки і форма за натисканням кнопки відправляє нас на сторінку редагування книги, де можна виправити назву, ціну або автора. Також є кнопка для видалення. Вона викликає метод контролера який видаляє відповідний запис у базі даних.

На сайті також реалізовано пошук за назвою книги. У шапці на кожній сторінці є форма для пошуку.

Рисунок 4.26 – Форма пошуку

У це поле для пошуку користувач вводить назву твору, який хоче знайти. Дані відправляються у контролер, проводиться валідація даних і якщо все добре, контролер звертається до бази даних через відповідну модель и робить запит який повинен повернути усе що має в назві введені дані, і повертає вісім результатів на сторінку з пагінацією.

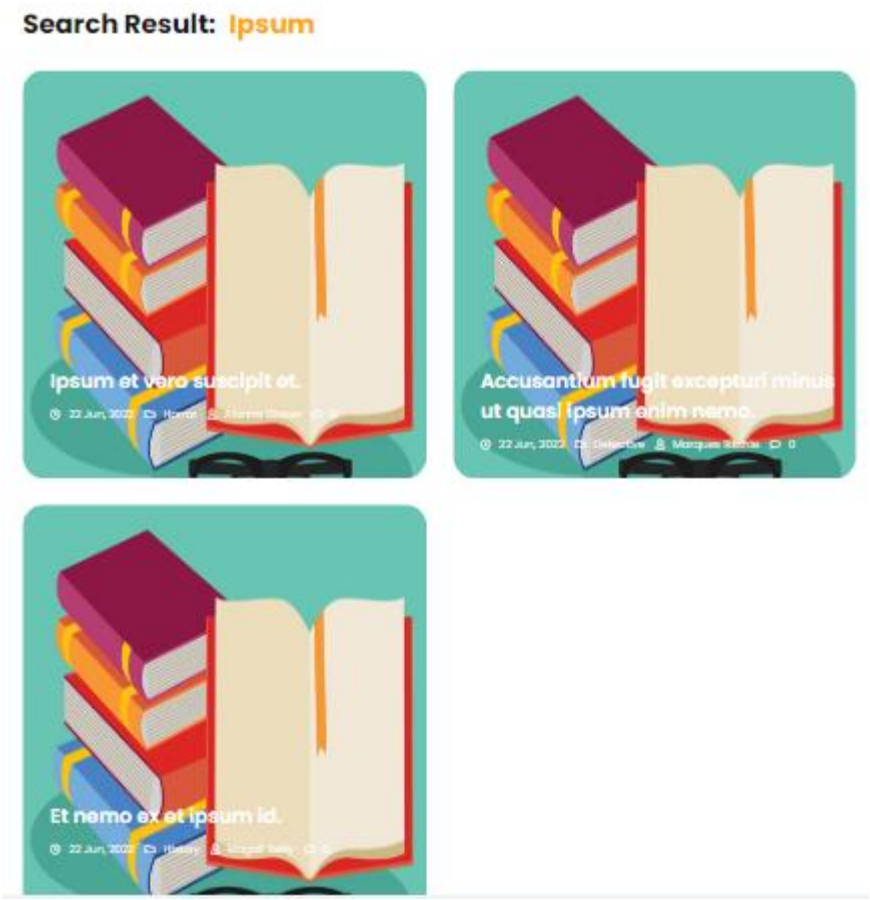


Рисунок 4.26 – Вивід усіх книжок які містять у своїй назві введений запит.

На рисунку вище зображено лише 3 книги, це свідчить про те, що в базі даних тільки 3 книжки було знайдено за нашим запитом.

Залишилося розглянути ще декілька моментів. Наступна сторінка яка буде розглянута це сторінка окремої книги. Там розміщена уся інформація про книгу, її назва, ціна, автор, категорія книги та короткий опис цієї книги. До кожної книги користувачі сайту можуть залишити коментарі, за таким же зразком як і повідомлення. Якщо користувач не авторизован, то йому відображається розширена форма для вводу коментаря. А якщо користувач навпаки пройшов авторизацію, то форма складається лише с поля для вводу коментаря. Дані які вводаться також проходять валідацію перед відправкою у базу даних.

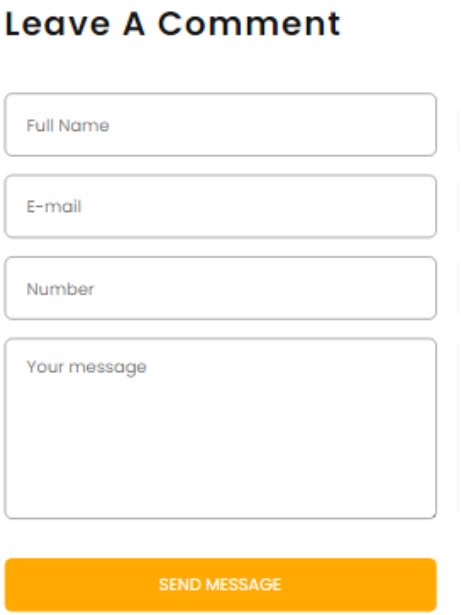


Рисунок 4.27 – Форма для додавання коментаря для не авторизованих користувачів

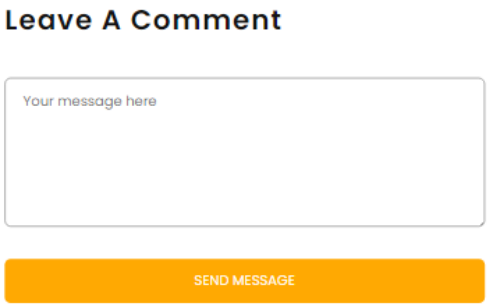


Рисунок 4.28 – Форма для додавання коментаря для авторизованих користувачів

Інформація про книгу, яка відображається на сторінці – це посилання, за якими можна перейти на інші сторінки. Можна перейти на сторінку категорії із постами цієї категорії і можна перейти на сторінку з постами автора. Потрібно лише натиснути на текст

Рисунок 4.29 – Посилання на категорію і на автора

Неавторизовані користувачі на сторінках можуть побачити лише популярні пости. Якщо ж користувач має обліковий запис і авторизован, то для нього відкривається певна можливість. Він може додати до бази даних запис про ту книгку, яка йому сподобалась. Для цього на сторінці кожної книги

проводиться перевірка, чи авторизован користувач. Якщо він авторизована то відображається кнопка яка відповідає за 2 дії: Додати до улюблених, або видалити з улюблених. Яка кнопка буде відображатись залежить від того, чи додали є запис про вподобання певної книжки у базі даних.

Рисунок 4.30 – Кнопка для додавання книги до улюблених

Рисунок 4.30 – Кнопка для видалення книги з улюблених

Якщо ви авторизований користувач і в базі даних є запис про вашу улюблену книгу, то для вас буде працювати модуль рекомендацій. У другому розділі були розглянуті методи для рекомендацій. Вибір був зосереджений на колаборативній фільтрації (user-based), яка робить рекомендація базуючись на найближчих сусідах за інтересам. Тобто метод шукає користувачів, яким подобається щось, що вподобали ви, і пропонує вам те що також подобається цим користувачам. Написаний метод обирає останній за датою запис про вподобання користувача, потім метод звертається до бази даних і шукає користувачів, які також вподобали вибрану книгу. Якщо пошук вдачний, і було знайдено користувачів які теж додали книгу до улюблених, то метод знову звертається до бази даних, і шукає книги які ці користувачі додавали до

улюблених і сортує за датою, тобто шукає найсвіжіші записи. Створює масив з п’ятьма книгами і виводить їх у сайдбар.

Рисунок 4.31 – Результат успішної роботи модуля рекомендацій.

Якщо виявилось так, що модуль не знайшов ніяких записів про користувачів, які додавали книгу до улюблених, то метод просто поверне з бази даних п’ять найпопулярніших постів.

Рисунок 4.31 – Результат невдалої роботи модуля рекомендацій.

Висновки до розділу 4

Можна зробити висновок, що фреймвор досить легкий в опануванні і його можливості легко реалізовувати: у кількох рядках коду можна створити відноси між таблицями бази даних, створити методи які будуть працювати з відносинами і отримувати потрібну нам інформацію без великих запитів мовою

sql. Дуже просто проводиться аутентификація користувачів, і досить легко і зрозуміло проводиться валідація даних що відправляється до бази даних.

В процесі написання четвертого розділу було опановано багато можливостей фреймворку і як результат, розроблен сайт книжкового магазину з модулем рекомендацій товару користувачам.

Спеціальний розділ

ОХОРОНА ПРАЦІ

до кваліфікаційної роботи

на тему:

СТВОРЕННЯ ІНТЕРНЕТ-МАГАЗИНУ КНИЖКОВОЇ ПРОДУКЦІЇ З МОДУЛЕМ ОПТИМІЗАЦІЇ ПРОПОЗИЦІЙ ТОВАРУ ВІДВІДУВАЧАМ

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 401.1810108

Виконав студент 4-го курсу, групи 401
_____*Дарій А.М.*
(підпис, ініціали та прізвище)
«___» _____ 2022 р.

Консультант, старший викладач
(наук. ступінь, вчене звання)
_____*Макарова О. В.*
(підпис, ініціали та прізвище)
«___» _____ 2022 р.

Миколаїв – 2022

ОХОРОНА ПРАЦІ

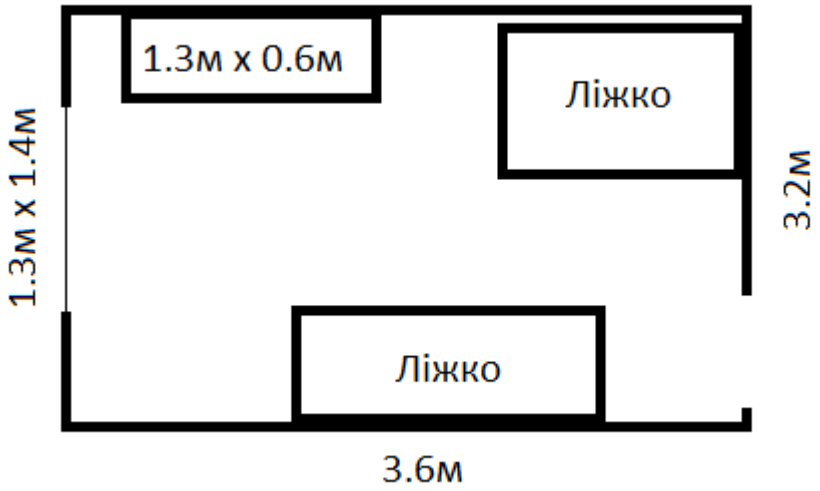
5.1 Вступ

Охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності. Основним законодавчим документом є Закон України «Про охорону праці», який визначає основні положення щодо реалізації конституційного права працівників на охорону їх життя і здоров'я у процесі трудової діяльності, на належні, безпечні і здорові умови праці, регулює за участю відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні.

Тема дипломної роботи - створення інтернет-магазину книжкової продукції з модулем оптимізації пропозицій товару відвідувачам. Виконується аналіз умов приміщення, в якому виконувалась робота та організації, в якій будуть використовуватися результати дипломної роботи. Це робиться для того, щоб уникнути неприємних ситуацій зі службами цивільної безпеки.

Основними шкідливими факторами на робочому місці користувача ПК є: електромагнітні випромінювання, ризик ураження електричним струмом; підвищений або знижений рівень освітлення; несприятливі параметри мікроклімату; напруга зору, уваги; фізичні перевантаження через тривале перебування у фіксованій позі.

5.2 Характеристика приміщення



Малюнок 4.1. План приміщення до 24.02.2022

Розробка програми велася у приміщенні, план якого приведено на малюнку 4.1.

Приміщення має одностороннє природне освітлення та загальне штучне освітлення.

Ширина приміщення 3,2 м, довжина – 3,6 м, висота стелі – 2,7 м. Кількість робочих місць – одне. Приміщення знаходиться на шостому поверсі дев’ятиповерхової будівлі. Площа – 11,52 м^2, об’єм – 31,1 м^3.

Виходячи з цього отримано дані наведені в таблиці 1. Нормативні значення згідно з Таблиці 4.1:

Таблиця 4.1.

Параметр	Норма	Фактичні параметри
Площа, S	Не менше ніж 6 м^2	11,52 м^2
Об’єм, V	Не менше ніж 20 м^3	31,1 м^3

Показники відповідають вимогам нормативних документів.

5.3 Мікрокліматичні умови

Оптимальні та фактичні параметри мікроклімату наведені у таблиці 4.2 про нормативні значення мікроклімату.

Таблиця 4.2. Нормативні значення мікроклімату

Період року	Параметр	Оптимальний	Фактичний
Теплий	Температура	23 – 25	28
	Вологість	60 – 40	50
	Швидкість повітря	< 0,1 м/с	
Холодний	Температура	22 – 24	22
	Вологість	60 – 40	40
	Швидкість повітря	< 0,1 м/с	

Значення відносної вологості повітря в холодний період року знаходиться на межі допустимих значень, доцільно використовувати в цей період зволожувачі повітря.

Температура повітря в теплий період року виходить за межі допустимих значень, для забезпечення вимог [101] необхідно встановлення кондиціонера.

5.4 Освітлення

Норми освітленості регламентуються у «ДСТУ EN 12464-1:2016 Світло та освітлення. Освітлення робочих місць. Частина 1. Внутрішні робочі місця».

Згідно з робота відноситься до розряду IIIв. При загальному освітленні показник освітлення робочої поверхні має бути у межах 200 – 400 лк.

У даному приміщенні використовується природне, штучне та змішане освітлення.

Джерелом природного освітлення є вікно шириною 1,3 м і висотою 1,4 м.

Джерелом штучного освітлення є світильник з трьома світлодіодними лампами потужністю 80 Вт, світловий потік 2150 лм.

Для визначення освітленості робочої зони скористаємось методом світлового потоку.

Формула світлового потоку має вигляд:

$$\Phi = \frac{EkSZ}{N\eta} \quad (1)$$

де Ф – світловий потік, Лм;

E – освітленість робочого місця, Лк;

k – коефіцієнт запасу, що враховує зменшення світлового потоку ламп у процесі експлуатації, *k* = 1,2;

Z – коефіцієнт нерівномірності, *Z* = 1,1;

S – площа приміщення, м^2;

N – кількість ламп;

η – коефіцієнт використання світлового потоку.

Звідси освітленість на робочому місці дорівнює:

$$E = \frac{\Phi N\eta}{kSZ} \quad (2)$$

Для визначення коефіцієнту використання світлового потоку η потрібно розрахувати індекс приміщення *i* за формулою 3:

$$i = \frac{S}{h\,(A+B)} \quad (3)$$

де *S* – площа приміщення, *S* = 11,52 м^2;

h – висота світильників над робочою поверхнею, м;

A - ширина приміщення, *A* = 3.2 м;

B - довжина приміщення, *B* = 3,6 м.

Висота світильників над робочою поверхнею знаходиться за формулою 4:

$$h = H - h_{\text{св}} - h_{\text{рп}} \quad (4)$$

де Н – висота приміщення, м;

$h_{\text{св}}$ – висота світильника, м;

$h_{\text{рп}}$ – висота робочої поверхні, м.

$$h = 2{,}7 - 0{,}3 - 0{,}8 = 1{,}6 \text{ м}$$

$$i = \frac{8{,}75}{1{,}6\,(3{,}2+3{,}6)} = 0{,}8$$

За індексом приміщення та коефіцієнтами світлового потоку від стелі – 70%, стін – 50%, підлоги – 30 % визначаємо значення коефіцієнту використання світлового потоку η = 0,5.

Підставимо всі значення у формулу 2 для визначення освітленості:

$$E = \frac{2150 \cdot 5 \cdot 0{,}51}{1{,}2 \cdot 8{,}75 \cdot 1{,}1} = 474 \text{ Лк}$$

Освітленість на робочому місці становить 474 лк, що відповідає вимогам для IIIв розряду зорових робіт.

5.5 Рівень шуму на робочому місці

Джерелом шуму в приміщенні є комп'ютер. Згідно технічній документації шум кулера у блоці живлення має рівень 10-15 дБ, кулера процесора – 10-15 дБ, загальний рівень шуму комп'ютера 25-30 дБ. Беручи до уваги незначний рівень шуму інших компонентів комп'ютера та незначний рівень фонового шуму іншого обладнання, сумарний рівень звукового забруднення у приміщені не перевищує 50 дБ.

Згідно з [103] допустимий рівень звуку на робочому місці має бути не вище ніж 50 дБ. Отже рівень звукового забруднення не перевищує норму.

Може бути перевищено рівень шуму з причини шумних сусідів, крику дітей надворі, шуму близьких поруч тощо. Тому робоче місце може бути змінено, але є ризик, що умови на тимчасовому робочому місці не будуть відповідати [100], [101], [102], [103].

5.6 Випромінювання

В приміщенні відсутні джерела інфрачервоного, ультрафіолетового та електромагнітного випромінювання. В моніторі комп'ютера використовується рідкокристалічна матриця з світлодіодною підсвіткою, що не створює значного електромагнітного випромінювання.

До того ж в роботі використовуються захисні окуляри проти шкідливого випромінювання. Ще періодичний відпочинок для очей може поліпшити стан очей.

5.7 Ергономіка робочого місця

Робоче місце працівника з ПЕОМ має відповідати вимогам [100].

Висота робочої поверхні стола має бути у межах 680-800 мм (фактичний розмір 750 мм), рекомендована ширина стола 600-1400 мм (фактичний розмір 1500 мм), рекомендована глибина 800-1000 мм (фактично 700 мм). Простір для ніг заввишки не менше ніж 600 мм (фактично 700 мм), завширшки не менше ніж 500 мм (фактично 700 мм), завглибшки не менше ніж 650 мм (фактично 700 мм). Крісло є підйомно-поворотним, має підлокітники та можливість регулювання за висотою, кутом нахилу сидіння та спинки.

Таким чином за ергономічними показниками робоче місце відповідає нормативним вимогам.

5.8 Ризики для виробничої організації

Для обробки замовлень або регістрації на сайті користувачем буде надаватись контактна та деяка особиста інформація.

Тому слід зазначити, що за документом [104], не допускається збирання, зберігання, використання і поширення конфіденційної інформації про особу без її згоди, крім випадків, визначених законом, і лише в інтересах національної безпеки, економічного добробуту та прав людини.

З цього випливає, що дані, які можуть зберігатись в організації, не повинні використовуватися в особливих цілях.

5.9 Висновки.

Аналіз умов праці в розглянутому робочому приміщенні показав, що умови праці з ПЕОМ відповідають вимогам [100,101,102,103], оскільки площа

та об’єм не менше нормативних значень, рівні шуму, вібрації і загазованості не перевищують нормативних значень.

Рівень освітлення робочого місця відповідає нормам.

Для підтримання параметрів мікроклімату в приміщенні встановлено радіатор водяної системи центрального опалення, що складається з 8 секцій.

Ергономіка робочого місця і режим зорової роботи задовольняють вимогам і сприяють зниженню втоми.

ВИСНОВОК

У ході виконання БКР було створено сайт інтернет-магазину книжкової продукції і реалізовано модуль рекомендацій товарів користувачам завдяки фреймворку Laravel.

Було запропоновано демоверсію сайту, яка дозволяє інтуїтивно просто використовувати у своїх цілях можливості які надає розроблена система.

В першому розділі ясно, що проблематика, яка була зачеплена, вкрай необхідна сьогодні і також у наступних роках, коли починається епоха комп'ютеризації абсолютно всіх сфер життя та діяльності. Немає жодних сумнівів, що подібні продукти будуть розвиватися все більше і більше, і рекомендаційні системи будуть становитись більш досконалим.

В другому розділі був проведений аналіз методів для вирішення задачі. Було проведено дослідження анатомії рекомендацій, що це таке, які цілі у рекомендацій, що може буди джерелом, та було досліджено декілька алгоритмів для рекомендацій, а саме Summary-based (неперсональні), Content-based (моделі, засновані на описі товару), Collaborative Filtering (колаборативна фільтрація), Matrix Factorization (методи засновані на матричному розкладанні) та деякі інші. Засновуючись на проведеному аналізі було вирішено скористатись Колаборативною фільтрацією (user-based). Проведоно дослідження мов програмування для серверу, для клієнтської сторони, тобто те що бачить користувач на сторінках та програмування і управління базою даних. Було невеличке дослідження реляційних систем управління базами даних. Проведено вивчення і дослідження існуючих СУБД.

У третьому розділі був проведений аналіз та порівняння характеристик фреймворків для серверної сторони сайту, з ціллю визначити кращий варіант якій підійде для успішного виконання поставленої задачі. В ході дослідження фреймворки оцінувалися за низкою критеріїв, за деякими з критеріїв переможця було визначити важко, але він все ж таки знайшовся. Також було спроектовано і коротко розписано з чого буде складатися сайт, які він матиме сторінки і відповідно функціонал.

У четвертому розділі було продемонстровано реалізацію завдання БКР. Було детально розписано про всі сторінки сайту, як вони наповнюються інформацією, як контролери працюють з базою даних і описан принцип роботи алгоритму рекомендацій.

Під час виконання БКР було навчено працювати в екстремальних умовах, вирішувати питання та завдання, з якими раніше не доводилося стикатися. Окрім технічних навичок, які доводилося наскільки можна вдосконалювати під час виконання роботи, були розвинені такі якості, як відповідальність, терпіння та посидючість. Завдяки вимогам, що були описані в охороні праці, проблему посидючості та терпіння було вирішено успішно, оскільки умови, в яких виконувалася БКР, сприяла підтримці фізичного та ментального здоров'я.

Виходячи з цього, тема БКР була висвітлена досконало з багатьох сторін. Важливість вирішення проблеми цієї роботи також було доведено.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин ДСанПІН 3.3.2.007-98. – К.: Постанова Головного державного санітарного лікаря України, 1998. - № 7.
2. Санітарні норми мікроклімату виробничих приміщень : ДСН 3.3.6.042-99. – К., 2000.- С.16
3. Природне і штучне освітлення : ДБН В.2.5-28:2006. – К. : Міністерство будівництва, архітектури та житлово-комунального господарства України, 2006. – 68 с. – (Національні стандарти України).
4. Санітарні норми виробничого шуму, ультразвуку та інфразвуку. Державні санітарні норми. ДСН 3.3.6.037-99. – К., 1999.
5. [Електронний ресурс] Документація до фреймворку Laravel
- https://laravel.com/docs/9.x
6. [Електронний ресурс] https://www.interviewbit.com/blog/php-frameworks/
7. [Електронний ресурс] https://scand.com/company/blog/top-5-php-frameworks-for-web-development/
8. [Електронний ресурс] https://devdojo.com/amyparker/top-8-php-frameworks-for-2022
9. [Електронний ресурс] https://kinsta.com/blog/php-frameworks/
10. C.C. Aggarwal: Recommender Systems: The Textbook – Springer, 2016.
11. D. Jannach, M. Zanker, A. Felfernig, G. Friedrich: Recommender Systems: An
12. Introduction – Cambridge University Press, 2011.
13. F. Ricci, L. Rokach, B. Shapira (eds.): Recommender Systems Handbook, 2nd ed. – Springer, 2015.
14. K. Falk: Practical Recommender Systems – Manning Publications Co., 2019.
15. Felfernig, L. Boratto, M. Stettinger, M. Tkacic: Group Recommender Systems: An Introduction, Springer Briefs in Electrical and Computer Engineering – Springer, 2018.

16.E. Negre: Information and Recommender Systems, Advances in Information Systems Set,
17.G. Kembellec, G. Chartron, I. Saleh (eds.): Recommender Systems – Wiley, 2014.
18.1. J. A. Konstan Recommender systems: from algorithms to user experience / J. A. Konstan J. A. //
19.User Modeling and User-Adapted Interaction. – 2012 – Vol. 22. – No. 1–2. – P. 101–123. 2. Schafer J. B.
20.E-Commerce Recommendation Applications / J. B. Schafer J. B., J. A. Konstan, J. Riedl // Data Mining
21. Recommendation algorithms for e-commerce / B. Sarwar, G. Karypis, J. Konstan, J. Riedl // In
22.Vol. 4571. – 2007. – P. 548–562. 7. Su X., Khoshgoftaar T. M. A survey of collaborative filtering
23.Recommendation systems: Principles, methods and evaluation / F. O. Isinkaye, Y. O. Folajimi,