ДВНЗ «Донецький національний технічний університет»

Факультет комп'ютерно-інформаційних технологій та автоматизації

Кафедра прикладної математики та інформатики

«До захисту допущено»
Завідувачка кафедри ПМІ

_____ Ольга ДМИТРІЄВА
(підпис) (ім'я та прізвище)

«18» червня 2022 р.

Випускна кваліфікаційна робота бакалавра

на тему «Підтримка А / В тестування елементів інтерфейсу сайтів інтернет-магазинів»

Спецчастина «Розробка модуля А / В тестування для сайту інтернет-магазину на мові Python та JavaScript»

Виконав: студент 4 курсу, групи КН-18
(шифр групи)

спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

Скринника Д. В. _____
(прізвище та ініціали) (підпис)

Керівник проф. каф. ПМІ, д.т.н., проф. Євген БАШКОВ _____
(посада, науковий ступінь, вчене звання, ім'я та прізвище) (підпис)

Рецензент доц. каф. КІ, к.т.н., доц. Юлія ДІКОВА _____
(посада, науковий ступінь, вчене звання, ім'я та прізвище) (підпис)

Нормоконтроль:

_____ к.т.н., доц. Ірина НАЗАРОВА
(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Луцьк – 2022 р.

ДВНЗ «Донецький національний технічний університет»

Факультет комп’ютерно-інформаційних технологій та автоматизації
Кафедра прикладної математики та інформатики
Освітній ступінь бакалавр
Спеціальність 122 Комп’ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ:
Завідувачка кафедри ПМІ

_____/Ольга ДМИТРІЄВА
6.05.2022 року

З А В Д А Н Н Я
НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ
Скриннику Данііілу Віталійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Підтримка A / B тестування елементів інтерфейсу сайтів інтернет-магазинів
Спецчастина Розробка модуля A / B тестування для сайту інтернет-магазину на мові Python та JavaScript
керівник роботи Башков Є. О., д.т.н., проф.
(ім’я та прізвище, науковий ступінь, вчене звання)

затверджені наказом від « 06 » травня 20 22 року № 180
2. Строк подання студентом роботи 18 червня 2022 року
3. Вихідні дані до роботи результати переддипломної практики
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) аналіз методів A / B тестування елементів інтерфейсу сайтів; 2) огляд інструментів A / B тестування; 3) проектування web-додатку; 4) розробка web-додатку інтернет-магазину; 4) робота користувача з web-додатком.
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 6 травня 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Ознайомлення із аналізом призначення, характеристик та параметрів, типових підходів до A / B тестування	09.05.22-15.05.22	
2	Створення дизайну сторінок web-додатку	16.05.22-22.05.22	
3	Створення програмної частини клієнту	23.05.22-29.05.22	
4	Створення модуля A / B тестування	30.05.22-03.06.22	
5	Оформлення пояснювальної записки та опис створеної системи. Проходження нормоконтролю	04.06.22-07.06.22	
6	Перевірка пояснювальної записки антиплагіатною системою. Оформлення презентації до роботи, графічного матеріалу та рецензування роботи	08.06.22-21.06.22	
7	Захист роботи	23.06.22	

Студент _____ Скринник Д. В.
(підпис) (ім'я та прізвище)

Керівник роботи _____ Башков Є.О.
(підпис) (ім'я та прізвище)

АНОТАЦІЯ

Скринник Даніїл Віталійович. Модуль A/B тестування для сайту інтернет-магазину. Випускна кваліфікаційна робота на здобуття освітньо-кваліфікаційного рівня «бакалавр» за спеціальністю 122 «Комп'ютерні науки». – ДВНЗ ДонНТУ, Покровськ, 2022.

Об'єктом дослідження є процеси A / B -тестування елементів інтерфейсів сайту в рамках реалізації цифрової маркетингової стратегії.

Предмет дослідження це маркетинговий метод A / B -тестування для оцінки ефективності веб-сайту.

Мета роботи це створення модуля з інструментами A / B -тестування сайту в рамках реалізації цифрової маркетингової стратегії інтернет- магазину.

Методи розробки базуються на інструменті маркетингового дослідження A / B тестування сайту; розробка програмного модуля, використовуючі фреймворк Flask для вебдодатків, створений з використанням мови програмування Python.

Проаналізовано особливості методів A/B тестування. Змодельовано та реалізовано web-додаток інтернет-магазину з модулем A / B тестування елементів сайту. Описано можливості взаємодії користувача з інтернет-сайтом.

Наукова новизна полягає у створенні гнучкого модуля A / B тестування, що адаптований для використання на сторінці інтернет-магазину.

Практичне значення полягає в полегшенні оцінки ефективності змін сторінки інтернет-магазина для збільшення результативності продажів.

Ключові слова: A / B тестування, мова Python, фреймворк Flask, інтернет-магазин, web-додаток.

ABSTRACT

Skrynnik Daniil Vitalievich. testing module for the online store's website. Final qualification work for the educational qualification level "Bachelor" in specialty 122 "Computer Science". - SHEl Donntu, Pokrovsk, 2022.

The object of research is the processes of A / B testing of site interface elements as part of the implementation of a digital marketing strategy.

The subject of the research is a marketing method of A / B testing to evaluate the effectiveness of a website.

The purpose of this work is to create a module with A / B site testing tools as part of the implementation of the digital marketing strategy of an online store.

Development methods are based on the market research tool A / B site testing; software module development using the Flask framework for web applications created using the Python programming language.

Features of A / B testing methods are analyzed. The web-application of the online store with the module A / B testing of site elements was modeled and implemented. The possibilities of user interaction with the Internet site are described.

The scientific novelty lies in the creation of a flexible A / B testing module adapted for use on the online store page.

The practical significance is to facilitate the evaluation of the effectiveness of changes to the online store page in order to increase sales performance.

Keywords: testing, Python language, Flask framework, online store, web-application.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

CTR – Click-Through Rate

CR – Conversion Rate

ARPU – Average Revenue Per User

AOV – Average Order Value

UML – Unified Modeling Language

IDE – Integrated Development Environment

HTML – HyperText Markup Language

CSS – Cascading Style Sheets

БД – База даних

ВСТУП

Розвиток інтернет-технологій змусив бізнес перейти в онлайн режим. В сучасних умовах, компанії, які не мають інтернет-магазину, програють конкуренцію й втрачають велику частку ринку. Проте наявністю веб-сайту вже нікого не здивуєш. Перші місця у видачі пошукових систем заслуговують лише найкращі рішення у сфері веб-розробки. Для того щоб краще розуміти побажання користувача, необхідно використовувати A / B тестування.

Ця тема досить актуальна, оскільки сервіси в Інтернеті розвиваються та вдосконалюються, а бізнес завжди конкурує за поширення на ринкі та намагається збільшити коефіцієнт конверсії своїх сайтів.

Мета роботи це створення модуля з інструментами A / B -тестування сайту в рамках реалізації цифрової маркетингової стратегії інтернет-магазину.

Зважаючи на визначену мету потрібно виконати наступні завдання:

1) ознайомитись із аналізом призначення, характеристик та параметрів, типових підходів до A / B тестування;

2) розробити дизайн та сторінки інтернет-магазину;

3) розробити web додаток інтернет-магазину;

4) розробити панель адміністратора з можливістю проведення A / B тестування та перегляду статистики.

Статистична частина сайту створена на основі маркетингового дослідження A / B тестування, web додаток створений на основі фрейморку Flask, що базується на мові програмування Python, для верстки сайту було використано мову розмітки HTML та створено дизайн сайту за допомогою мови стилю CSS.

Обґрунтованість розробки базується на створеному web-додатку, що виконує функції інтернет-магазину та має вбудований модуль A / B тестування з можливістю перегляду статистичних даних.

1 МЕТОД А/В ТЕСТУВАННЯ ЕЛЕМЕНТІВ ІНТЕРФЕЙСУ САЙТІВ: ПРИЗНАЧЕННЯ ТА ЗАГАЛЬНІ ХАРАКТЕРИСТИКИ І ПАРАМЕТРИ

В даний час в різних (роздрібних, медійних, новинних, інтернет-рекламних) компаніях спостерігається основна тенденція надавати клієнтам максимально швидкий доступ до контенту, який вони, швидше за все, будуть використовувати. Кращий підхід для цього та інших типів сервісів або рішень по оптимізації користувацького інтерфейсу полягає в постійному внесенні змін в пропоновані сервіси або інтерфейси із застосуванням визначеного індикатора для вимірювання того, яка зміна дає найкраще очікуване значення індикатора. У той же час компанії, які пропонують послуги електронної комерції через свій веб-сайт або додаток, постійно оптимізують процес покупки, щоб зменшити зусилля клієнтів по завершенню процесу онлайн-покупки і забезпечити розміщення замовлення якомога більшою кількістю клієнтів. Цей тип експериментів широко відомий як -тестування.

А / В тестування відноситься до рандомизированному процесу експериментів, в якому дві або більше версії змінної (веб-сторінка, елемент сторінки і т. д.) показуються різним сегментам відвідувачів веб-сайту одночасно, щоб визначити, яка версія надає максимальний вплив і впливає на бізнес-показники. А / В-тестування використовується як для надання більш персоналізованих послуг, так і для оптимізації процесу покупки в електронній комерції.

1.1 Загальні етапи А / В тестування

Кожен тест слід структурувати наступним чином:

1. постановка цілі: для чого проводиться саме цей експеримент, на яке питання потрібно отримати відповідь, що потрібно перевірити, якою метрикою необхідно вимірювати результат;

2. формулювання гіпотез для досягнення мети, підтвердивши/спростувавши які буде вирішено поставлене завдання;

3. конструювання та проведення експерименту для перевірки гіпотез, збір даних і розрахунок тих критеріїв, які необхідні для валідування поставлених гіпотез;

4. перевірка підтвердження гіпотез, створення висновків на рівні поставлених завдань і впровадження результатів [1].

1.2 Вибір метрик для оцінки ефективності A / B тестування

Вибір показника є однією з найважливіших частин -тесту, оскільки цей показник буде використовуватися для вимірювання продуктивності продукту або функції для експериментальних і контрольних груп і буде використовуватися для визначення наявності статистично значущої різниці між цими двома групами.

Для вимірювання ефективності сайту використовуються різні види метрик. Для дискретних метрик, які також називають біноміальними метриками, можливі тільки два значення 0 і 1. Основні дескретні метрики наведені нижче [2].

Коефіцієнт клікабельності (CTR), використовується для врахування загальної кількості переглядів або сеансів. Коефіцієнт клікабельності розраховується за наступною формулою:

$$CTR = \frac{tc*100\%}{tc+tv} \quad (1.1)$$

де CTR – коефіцієнт клікабельності;

tc – кількіст людей, що натискають на сторінку (рекламу);

tr – кількість людей які переглядають сторінку (рекламу).

Коефіцієнт конверсії (CR) – це процес збільшення відсотка користувачів або відвідувачів веб-сайту для здійснення бажаної дії (наприклад, покупки

продукту або залишення контактних даних). Коефіцієнт конверсії розраховується за наступною формулою:

$$CR = \frac{c}{n} * 100\% \quad (1.2)$$

де CR – коефіцієнт конверсії;

c – загальна кількість конверсій;

n – загальна кількість людей, що відвідали ваш сайт.

Показник відмов являє собою відсоток відвідувачів, які заходять на сайт і потім виходять, а не продовжують переглядати інші сторінки на тому ж сайті й визначається за наступною формулою:

$$R_b = \frac{T_v}{T_e} \quad (1.3)$$

Де R_b – показник відмов;

T_v – загальна кількість відвідувачів, які переглядають тільки одну сторінку;

T_e – загальна кількість записів на сторінці.

Іншим видом метрик є безперервні метрики або небіноміальні метрики які можуть приймати безперервні значення та не обмежені набором двох дискретних станів. Нижче наведено основні безперервні метрики:

– середній дохід на користувача (ARPU), – показник, який використовується в основному споживчими комунікаціями, цифровими медіа і мережевими компаніями, визначений як загальний дохід, поділений на кількість передплатників;

– середня вартість замовлення (AOV) відстежує середню суму, що витрачається кожного разу, коли клієнт розміщує замовлення на веб-сайті або в мобільному додатку [2].

1.3 Формування гіпотез для А / В тестування

Гіпотеза – це передбачення, яке створюється перед проведенням експерименту. У ній чітко йдеться, що змінюється, який, буде результат. Провівши експеримент гіпотеза доводиться або спростуєься.

Потім слід виділити дві гіпотези, які допоможуть зрозуміти, чи те, що відбувається на сторінці, результатом внесених до її роботи змін:

- нульова гіпотеза припускає, що результати версій А і В не сильно відрізняються, а всі відмінності є випадковими, цю гіпотезу слід спростувати;
- альтернативна гіпотеза полягає в тому, що А відрізняється від В, і необхідно зробити висновок про її істинність [3].

1.4 Вибір гіпотез для А / В тестування

Для створення гіпотези необхідно правильно обрати елемент сайту, який необхідно тестувати. Нижче наведено основні гіпотези, що найчастіше досліджуються в ході А/В-тестування.

CTA (call to action або заклик до дії) – це фрагмент тексту або елемент (наприклад кнопка), що закликає відвідувача зробити потрібну дію, наприклад, залишити заявку або підписатися на розсилку, приклади кнопок заклику до дії зображено на рисунку 1.1.

На головній сторінці сайту і сторінці замовлення користувач звертає увагу на ряд чинників:

- вибір при реєстрації;
- користуватися сайтом як авторизований користувач або як гість;
- кількість полів при реєстрації;
- лід-форми;
- сторінка замовлення;
- варіанти оплати і доставки.

У інтернет-магазині найчастіше тестуються наступні елементи:

- розподіл продукції в категорії товарів;
- кількість термінів при описі товару;

– текст, розбитий на кілька блоків;
– навігацію по сайту на кожній сторінці сайту;
– кнопки заклику до цільових дій [3].

Рисунок 1.1 – Приклади кнопок заклику до дії

Джерело [4]

1.5 Обчислення розміру вибірки A / B тестування

Обчислення необхідного розміру вибірки для A / B -тесту необхідне для проведення експерименту з правильною потужністю. Занадто мало зразків (користувачів) – є ризик не побачити ефект, занадто багато – експеримент буде проводиться занадто довго.

Оскільки -тестування схильне випадковості, нам потрібно обмежити два типи помилок:

1. помилка типу I – це ймовірність того, що при відсутності експериментального ефекту ми все одно знайдемо статистичну різницю через випадковість, дана ймовірність називається рівнем значущості (α) і зазвичай встановлюється рівним 0.05 (або 5% імовірності);

2. помилка типу II – це ймовірність (β) того, що реальний ефект не покаже значних результатів.

Також необхідно ввести поняття потужності ($1-\beta$) - це ймовірність того, що реальний ефект дасть значні результати. Зазвичай ми встановлюємо потужність на 0,8 (80%) і β= 0,2 (20%) [5].

Зміни, які необхідно побачити з заданими порогами помилок I і II типів розраховуються за допомогою мінімального очікуваного ефекту конверсії (MDE).

Розрахунок необхідної кількості користувачів для проведення експеременту для заданих параметрів конверсії та вірогідності помилок типу I та типу II зазначено на формулі 1.4.

$$N = \left(\frac{Qnorm\left(1-\frac{a}{2}\right) * \sqrt{Pc*(1-Pc*2)} - Qnorm(\beta)}{Dmin} \right)^2 \qquad (1.4)$$

де N – необхідна кількість користувачі;

a – ймовірність помилки типу I;

Qnorm (1-a/2) – нормальне розподілення для помилки типу I;

Pc – коефіцієнт конверсії в даний час;

Qnorm (β) – нормальне розподілення для помилки типу II;

Dmin – очікуване підвищення конверсії [5].

1.6 Сегментування користувачів для A / B тестування

Сегментування – це ключ до отримання вигоди від результатів A / B тестування. Незважаючи на те, що B може втратити A в загальних результатах, варіація може перемогти оригінал сторінки в певних сегментах (органічний трафік, переходи з Facebook, мобільний трафік тощо).

Існує величезна кількість даних які використовується для сегментації:

- тип браузера;
- тип джерела;
- мобільний або настільний комп'ютер або пристрій;
- зареєстровані і вийшли з системи відвідувачі;
- PPC / SEM-кампанії;
- географічні регіони (місто, штат/провінція, країна);
- нові та постійні відвідувачі;

- нові та повторні покупці;
- просунуті користувачі проти випадкових відвідувачів;
- чоловіки проти жінок;
- віковий діапазон;
- нові і вже представлені ліди;
- типи планів або рівні програми лояльності;
- поточні, потенційні та колишні передплатники;
- ролі (якщо, наприклад, сайт пропонує ролі покупця і продавця) [6].

2 ОГЛЯД ІНСТРУМЕНТІВ А / В ТЕСТУВАННЯ

Для проведення А / В тестування найчастіше використовуються готові рішення, які мають багато налаштунків для створення гіпотез та тестування будь-яких елементів сайту. Дані інструменти також мають вбудовані метрики для аналізу результатів тестування.

2.1 Інструмент Google Optimize

Optimize від Google – це один з найпопулярніших сервісів, що призначений для спліт-тестування. Система підключається до сайту і дозволяє проводити експерименти з різними варіантами відображення функціональних елементів та контенту. Також Google Optimize дозволить користуватись даними з Google Analytics, які допоможуть знайти найбільш зручний варіант для користувачів, інтерфейс зображений на рисунку 2.1.

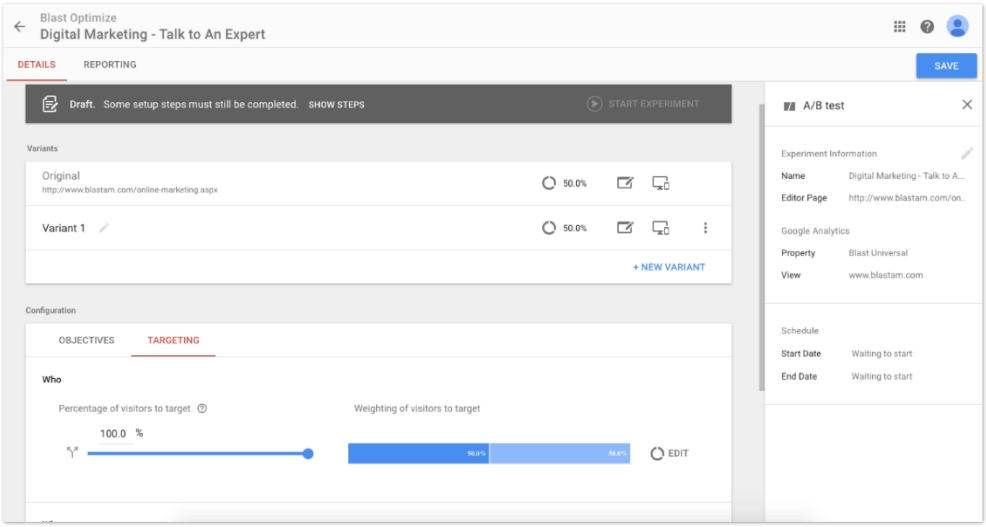


Рисунок 2.1 – Інтерфейс інструменту Google Optimize

Джерело [7]

Робота над створенням А/Б тесту в Google Optimize складається з кількох етапів:

- Створення варіантів для тесту, що будуть представлені користувачеві.

- Окреслення цілей та параметрів для визначення варіанту-переможця. Серед них число переглядів варіантів, тривалість сесії, конверсії, показник відмов і т.д.
- Визначення аудиторії, що буде брати участь в тестуванні та розподіл трафіку між варіантами сторінок.

При визначенні частки аудиторії, яка братиме участь в експерименті, варто розглянути варіант розподілу трафіку між половиною користувачів або взяти 20% і залучити їх до тесту.

Окрім класичних A/B тестів у сервісі Optimize доступний запуск мультитестів та редірект-тестів. Мультиваріантне тестування призначене для перевірки ефективності мінливих елементів та різноманіття комбінацій. Редірект-тест призначений для сторінок з різним дизайном та URL адресами.

Після проведення тесту в Google Optimize існує можливість ознайомитись зі звітом та проаналізувати результати. Найчастіше у звітах зустрічаються такі терміни, на які варто звернути увагу:

- Improvement –вірогідний діапазон для коефіцієнтів конверсій.
- Conversions – число активних сесій з конверсіями.
- Conversion Rate – передбачений середній коефіцієнт конверсій.
- Probability to be best – вірогідність того, що варіант перевершив усі інші.
- Probability to beat baseline – варіант з більшим коефіцієнтом конверсій, ніж оригінал.

Визначення кращого варіанту відбувається з попереднім розрахунком імовірності згідно з Байесовою статистикою. Тобто можливо дізнатись варіант, який стане ймовірним переможцем ще до початку тесту [8].

2.2 Інструмент Optimizely

Optimizely – це експериментальна платформа, яка допомагає розробникам створювати і запускати тести на веб-сайтах. Це робиться для того, щоб домогтися валідації зусиль щодо оптимізації коефіцієнта конверсії. В Optimizely можливо створювати різні оптимізовані експерименти

для своїх веб-сторінок, використовуючи динамічні або базові функції, що поставляються разом з сервісом, інтерфейс Optimizely зображено на рисунку 2.2.

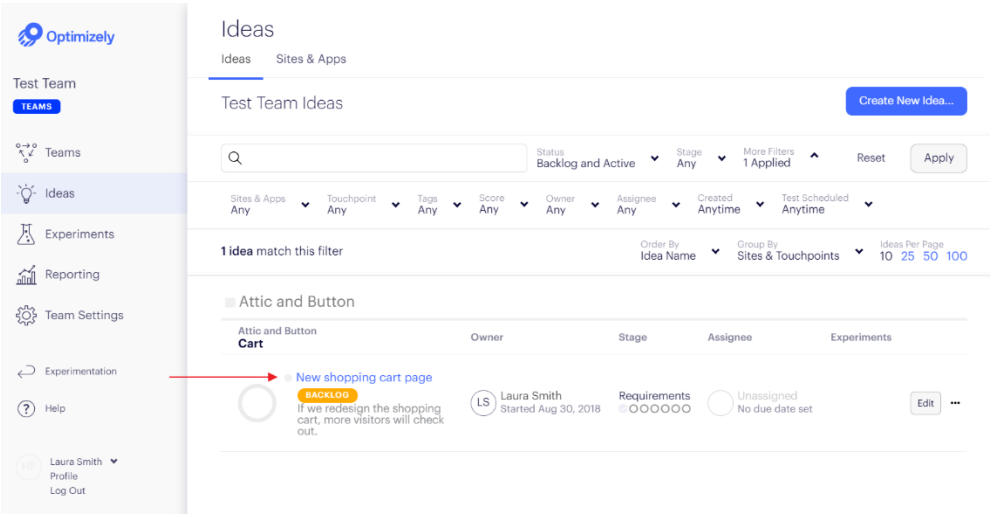


Рисунок 2.2 – Інтерфейс інструменту Optimizely

Джерело [9]

Нижче наведено основні переваги сервісу Optimizely:

– Optimizely обслуговує найрізноманітніші аудиторії: має простий у інтерпретації редактор для початку роботи, візуальний редактор Optimizely може бачити і редагувати сторінку саме так, як необхідно, щоб її бачили користувачі;

– існує можливість налаштування експериментів, використовуючи сторінку, подія, теги, аудиторію і різні інші функції, щоб створити, націлити і оцінити вплив споживчого досвіду;

– Optimzely включає в себе такі опції, як користувальницький таргетинг, пріоритизація аудиторії, шаблони, візуальний редактор, інструменти розробника і API [10].

Робота над створенням А/Б тесту в Optimizely складається з кількох етапів:

1. потрібно скопіювати і вставити один рядок в JavaScript код сайту:

– необхідно перети на панель управління Optimizely, звідти перейти в налаштування та вибрати реалізацію;

– звідти, виберати той фрагмент сайту, який необхідно вставити на сайт, натиснувши на значок з трьома крапками;

– натиснути кнопку скопіювати фрагмент коду;

2. далі необхідно налаштувати основні параметри дослідження:

– встановити експериментальний розподіл трафіку – частка загального трафіку для включення в експеримент, що зазначена у відсотках;

– встановити варіаційні ключі і розподіл трафіку – різні шляхи коду, з якими ви хочете поекспериментувати;

– додати події, які відстежуються з допомогою Optimizely SDK, в якості показників для вимірювання впливу;

3. далі проводеться A / B тестування: заходить на сторінку і якщо тест готовий до застосування, користувача буде згруповано або в контрольну групу, або в тестову групу;

4. Optimizely завантажує фрагментом, після чого почне вносити необхідні зміни, необхідні для зміни контрольної версії в варіацію, якщо користувач потрапляє в тестову групу, поки сторінка завантажується, Optimizely перезаписує сторінку до того, як вона завершить завантаження;

5. створюється звіт, і де можливо проаналізувати результати після закінчення запропонованого періоду часу [11].

3 ПРОЕКТУВАННЯ WEB-ДОДАТКУ

Для створення web-додатку було спроектовано логічну модель за допомогою спеціалізованих методів дослідження. Для більш кращого розуміння проєктованого web-додатку було створено декілька діаграм UML, що необхідні для: опису взаємодії користувача з системою, планування необхідних функцій, планування необхідних програмних компонентів, їх взаємодії. Також досліджено проектування бази даних, що буде використана при реалізації web-додатку.

3.1 Моделювання UML-діаграм

Уніфікована мова моделювання (UML) – це мова, яка використовується в області програмної інженерії, що представляє компоненти концепцій об'єктно-орієнтованого програмування. UML це загальний спосіб визначення всієї архітектури або структури програмного забезпечення. В об'єктно-орієнтованому програмуванні вирішуються складні алгоритми і взаємодію з ними, розглядаючи їх як об'єкти або сутності, що виконують певні завдання. Завданнями можуть бути взаємодія з іншими об'єктами, передача даних від одного об'єкта до іншого, маніпулювання іншими об'єктами. Одне програмне забезпечення може містити сотні або навіть тисячі об'єктів. Таким чином, UML надає спосіб подання та відстеження цих об'єктів на діаграмі, щоб спроектувати архітектуру програмного забезпечення [12].

3.1.1 Побудова UML-діаграми компонентів

Діаграма компонентів використовується для моделювання статичного подання реалізації системи, тобто компонентів, які допомагають створювати функціональні можливості системи. Вона може показати деталі реалізації і допомагає гарантувати, що запланована розробка охоплює всі аспекти проекту. Діаграми компонентів дають візуальне уявлення про організацію

різних компонентів системи і їх залежності та відносини один з одним, допомагають детально моделювати реалізацію і, перевіряти чи відповідає запланована розробка всім вимогам. [13].

Діаграма компонентів складається з наступних складових:

1) компонент (будівельний блок або логічний блок системи, інкапсулює вміст сутності, що є типом класу, але з більш високою абстракцією, ніж класи);

2) інтерфейс (показує, як компоненти з'єднані і взаємодіють один з одним, описує групу операцій який клас, компонент або підсистема може надати іншому класу, компоненту або підсистемі, для виконання ним своїх функцій);

3) взаємозв’язок (пов'язують один компонент з іншим, де стрілка показує напрямок залежності) [13].

Діаграма компонентів web-додатку інтернет-магазину складається з наступних компонентів:

- Main.py (програмний код клієнту web-додатку);
- Database.py (програмний код, що створює сховище бази даних);
- database.db (сховище бази даних, що зберігає дані web-додатку);
- profileHome.html (сторінка профіля користувача);
- admin.html (головна сторінка панелі адміністатора);
- adminLogin.html (сторінка авторизації у панель адміністратора);
- adminregister.html (сторінка реєстрації у панель адміністратора);
- add.html (сторінка з шаблоном додавання нового товара);
- categories.html (сторінка для додавання та видалення категорій);
- static.html (сторінка для показу статистики з A / B тестування);
- items.html (сторінка для додавання та видалення товарів);
- cart.html (сторінка корзини);
- changePassword.html (сторінка зміни паролю користувача);
- checkout.html (сторінка виведення чеку);
- displayCategory.html (сторінка з товарами з конкретої категорії);

– editProfile.html (сторінка з редагування даних користувача);

– found.html (сторінка, що виводить знайдені товари);

– home.html (головна сторінка інтернет-магазину);

– login.html (сторінка аторизації користувача);

– productDescription.html (сторінка опису товару);

– register.html (сторінка реєстрації користувача).

На рисунку 3.1 зображено діаграму компонентів web-додатку інтернет-магазину з взаємозв’язками між вище зазначеними компонентами.

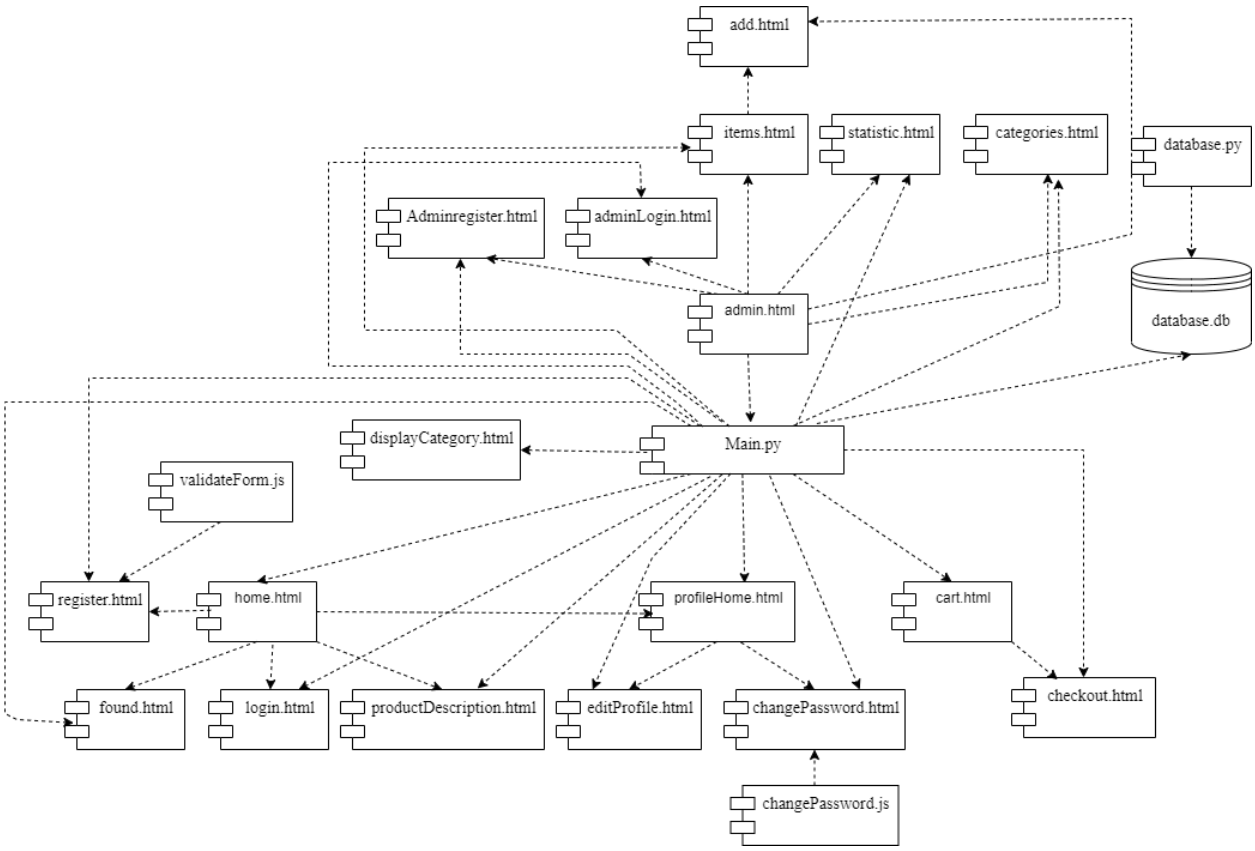


Рисунок 3.1 – Діаграма компонентів web-додатку інтернет-магазину

Джерело: авторська розробка

3.1.2 Побудова UML-діаграми варіантів використання

UML діаграма варіантів використання (діаграма прецедентів) – це спосіб моделювання системних вимог для розроблюваної нової програмної системи. Вона показує очікуваний вигляд і реакцію на дизайн, які бачить кінцевий

користувач. Діаграма прецедентів – це раціональний метод показати поведінку розроблюваної системи з точки зору передбачуваних користувачів, використовуючи візуалізацію зовні видимої поведінки даної системи. Крім того діаграми варіантів використання допомагає визначити внутрішні або зовнішні фактори, які можуть вплинути на робочий процес системи.

Діаграма компонентів складається з декількох компонентів:

1) актори (зовнішні користувачі, які взаємодіють з системою: людина, організація або зовнішня система, яка взаємодіє з аналізованим додатком);

2) прецедент (являє собою дію, і тому імена повинні починатися з дієслова) [14];

3) відносини, що викиконують різні функції: показують зв'язок між актором і прецедентом, узагальнення акторів, розширення або включення прецендента іншим пецендентом [14].

Діаграма варіантів використання web-додатку інтернет-магазину, що зображена на рисунку 3.2 вміщує декілька акторів: користувач, клієнт, адміністратор. Також зображено відносини акторів з прецендентами.

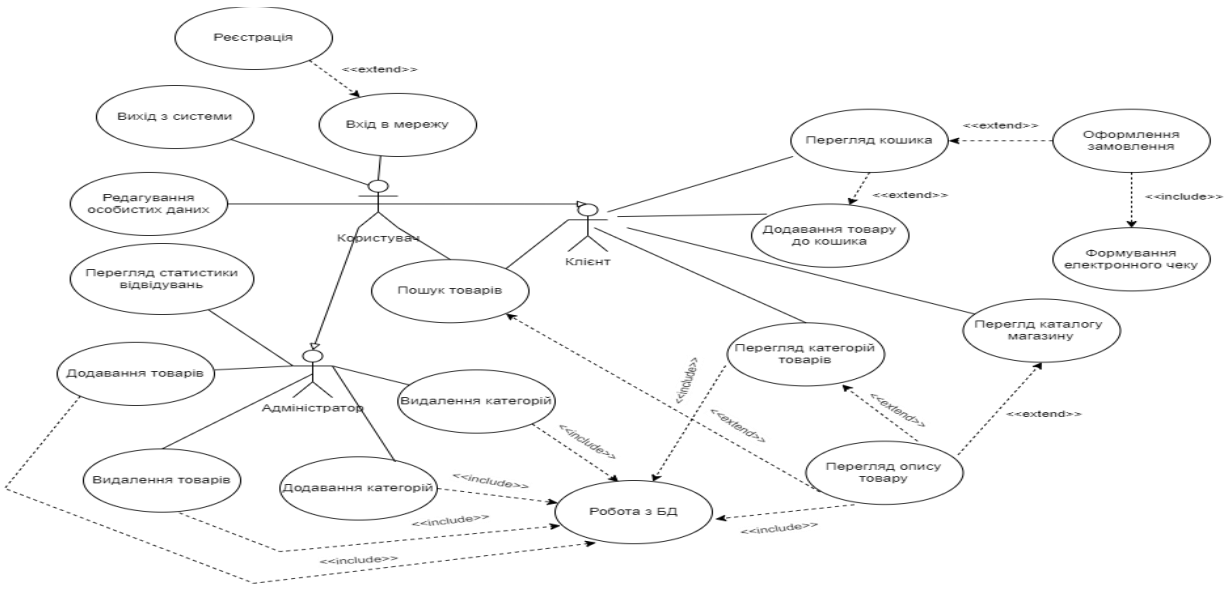


Рисунок 3.2 – Діаграма варіантів використання web-додатку інтернет-магазину

Джерело: авторська розробка

3.2 Моделювання бази даних

Після проектування моделі web-додатку, було спроектовано модель бази даних, для цього використано діаграму відносин сутностей (ER), що являє собою тип структурної діаграми для використання при проектуванні баз даних. ER діаграма містить різні символи і з'єднувачі, які візуалізують дві важливі інформації: основні об'єкти в області дії системи і взаємозв'язку між цими об'єктами.

Основні компоненти ER діаграми:

– сутність – це визначена річ або концепція в системі, подія, в контексті баз даних сутністю є таблиця;

– атрибут – це властивість або характеристика сутності, яка його містить, атрибут має ім'я, що описує властивість, тип атрибуту;

– первинний ключ – це особливий вид атрибута сутності, який однозначно визначає запис в таблиці бази даних;

– зовнішній ключ – це посилання на первинний ключ у таблиці, використовується для ідентифікації відносин між сутностями;

– відносини – це зв'язок між двома сутностями означає, що ці дві сутності якимось чином пов'язані один з одним, потужність відносин визначає можливу кількість входжень в одному об'єкті, який пов'язаний з кількістю входжень в іншому.

Для web-додатку інтернет-магазину визначено наступні сутності:

– products (каталог товарів в магазині);

– categories (категорії товарів в магазині);

– orders (список замовлень в магазині);

– roles (ролі користувачів магазина);

– users (дані аккаунтів користувачів магазину);

– kart (товари, що знаходяться в кошику покупця).

Для кожної сутності були визначені атрибути, що вказують тип даних, ім’я атрибута, його призначення (табл. 3.1-3.6).

Таблиця 3.1 – Атрибути сутності «users»

Назва атрибута	Тип атрибута	Призначення атрибута
userId	integer	первинний ключ – id користувача
password	text	пароль користувача
email	text	Email користувача
firstName	text	ім’я користувача
LastName	text	прізвище користувача
address1	text	поле для введення адреси проживання користувача
address2	text	продовження поля для введення адреси користувача
zipcode	text	індекс адреси користувача
city	text	місто проживання користувача
state	text	область проживання користувача
country	text	країна проживання користувача
phone	text	номер телефону користувача
roleId	integer	вторинний ключ – роль користувача

Таблиця 3.2 – атрибути сутності «orders»

Назва атрибута	Тип атрибута	Призначення атрибута
roleId	integer	вторинний ключ – користувач, що придбав товар
productId	text	вторинний ключ – назва товару

Таблиця 3.3 – атрибути сутності «kart»

Назва атрибута	Тип атрибута	Призначення атрибута
1	2	3
userId	integer	вторинний ключ – користувач, що використовує корзину

Продовження таблиці 3.3

1	2	3
productId	integer	вторинний ключ – назва товару, що добавлено в корзину

Таблиця 3.4 – атрибути сутності «products»

Назва атрибута	Тип атрибута	Призначення атрибута
productId	integer	первинний ключ товару
name	text	назва товару
description	text	опис товару
image	text	назва зображення обкладинки
stock	integer	кількість товару в наявності
categoryId	integer	вторинний ключ – назва типу категорії товару
author	text	автор товару
price	integer	ціна товару

Таблиця 3.5 – атрибути сутності «categories»

Назва атрибута	Тип атрибута	Призначення атрибута
categorytId	integer	первинний ключ категорії
name	text	назва категорії товарів

Таблиця 3.6 – атрибути сутності «roles»

Назва атрибута	Тип атрибута	Призначення атрибута
categorytId	integer	первинний ключ категорії
name	text	назва категорії товарів

На рисунку 3.3 зображено ER діаграму бази даних web-додатку інтернет-магазину, де зазначені відносини між сутностями та показана потужність цих відносин.

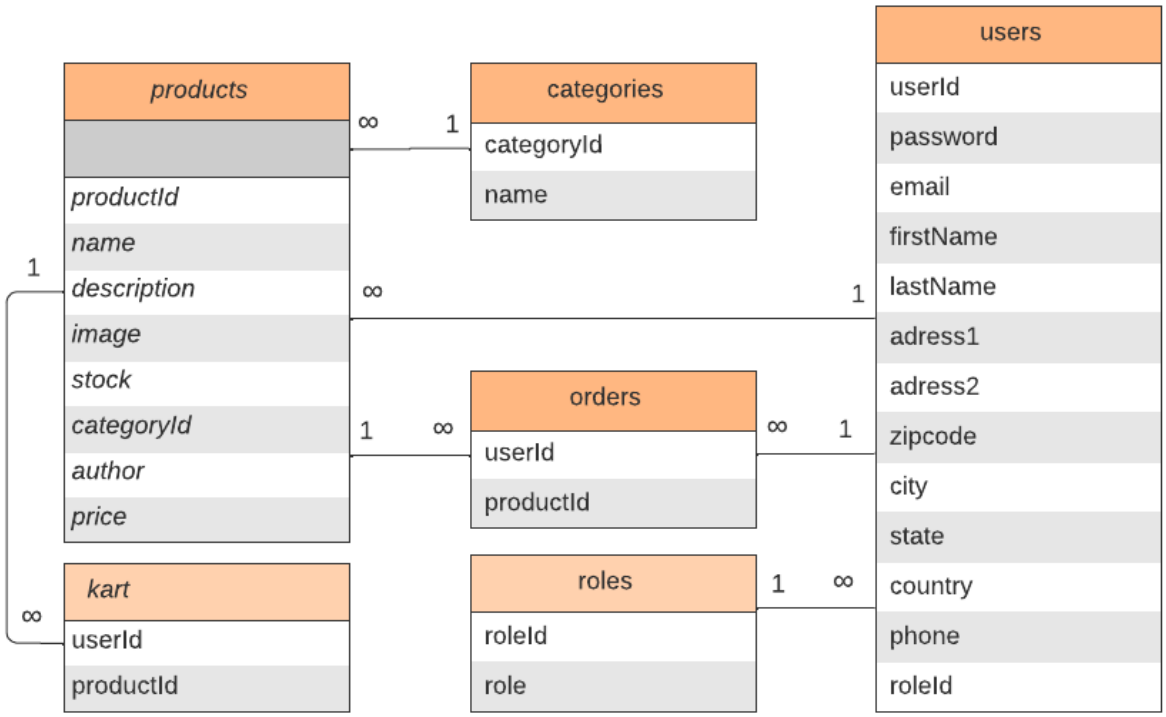


Рисунок 3.3 – ER діаграма бази даних web-додатку інтернет-магазину

Джерело: авторська розробка

4 РОЗРОБКА WEB-ДОДАТКУ ІНТЕРНЕТ-МАГАЗИНУ

4.1 Огляд засобів розробки

Для написання програмної частини web-додатку і його компіляції необхідно використовувати IDE, що включає редактор і компілятор, які використовуються для написання програми.

Використання IDE спрощує написання програмного додатку, інтерпретує, введений текст та пропонує відповідні ключові слова IDE допомагає розрізняти клас і метод, виділяючи її різними кольорами. Якщо виникає яка-небудь помилка, IDE показує попередження і пропозиції у вікні виводу для її усунення [15].

Для створення сторінок web-додатку використовувався звичайний текстовий редактор з підсвіткою коду.

4.1.1 Середовище розробки PyCharm

PyCharm – це гібридна платформа, розроблена компанією JetBrains як IDE для Python. Він зазвичай використовується для розробки додатків на Python [15].

Нижче наведено основні функції, що надає PyCharm:

– інтелектуальний редактор коду, що складається з колірних схем для ключових слів, класів і функцій, надає функцію автозаповнення та інструкції по завершенню коду;

– рефакторинг в PyCharm дозволяє розробникам покращувати внутрішню структуру без зміни зовнішньої продуктивності коду, швидко вносити зміни як в локальні, так і в глобальні змінні;

– PyCharm підтримує популярні веб-фреймворки, такі як Django та Flask;

– PyCharm підтримує наукові бібліотеки Python, такі як Matplotlib, NumPy і Anaconda, ці наукові бібліотеки допомагають у створенні проектів у галузі науки про дані та машинного навчання. [15].

Вигляд інтерфейсу IDE Pycharm: редактору коду з підсвіткою ключових слів зображено на рисунку 4.1.

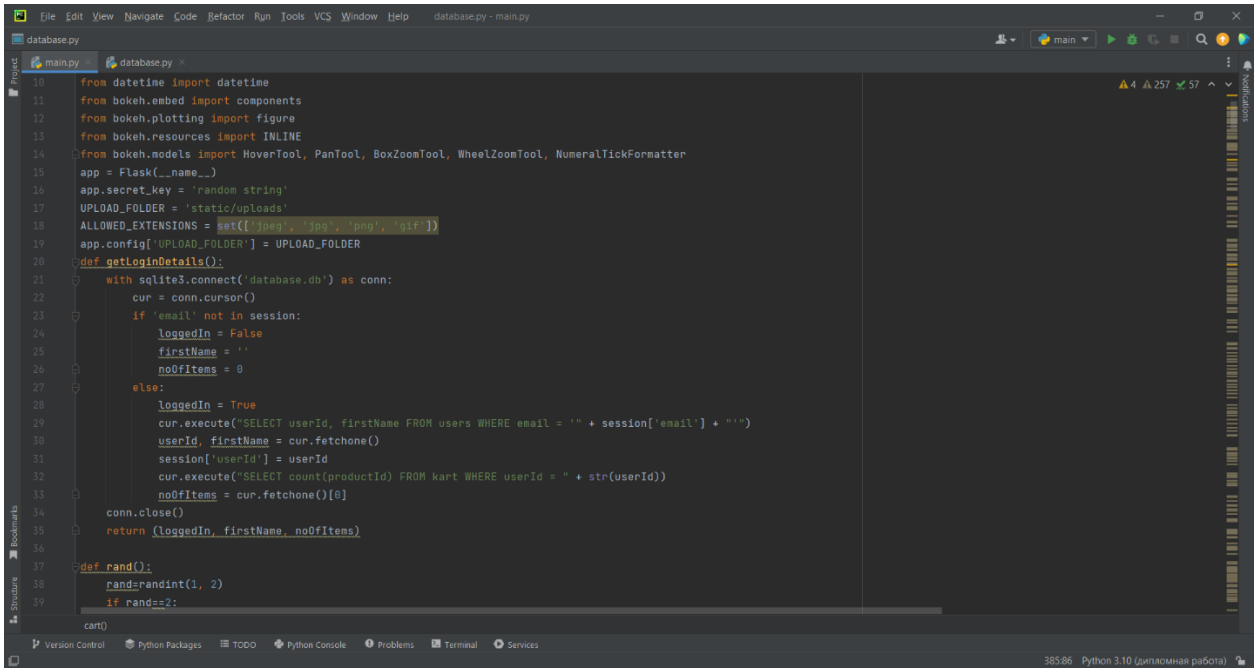


Рисунок 4.1 – Вигляд інтерфейсу IDE Pycharm

Джерело: авторська розробка

4.1.2 Текстовий редактор Notepad++

Notepad++ – це рішення на базі Windows, яке допомагає розробникам додатків створювати і редагувати вихідні коди на декількох мовах програмування, таких як Python, Matlab, Ruby і Verilog. Функціональність сеансів дозволяє членам команди створювати кілька активних файлів і зберігати інформацію, таку як поточні файли, закладки, вибрані елементи і мови. Notepad++ також дозволяє знаходити і замінювати текст в декількох файлах на основі декількох методологій, таких як інкрементний пошук, Пошук на основі діалогів і пошук без діалогів. Модуль автоматизації завдань дозволяє записувати і повторно використовувати послідовності дій з редагування

документів. Крім того, вбудований графічний інтерфейс користувача (GUI) дозволяє керівникам редагувати файли конфігурації, налаштовувати контекстні меню, додавати ключові слова, планувати резервне копіювання даних і створювати поєднання клавіш [16].

Вигляд інтерфейсу редактора Notepad++ зображено на рисунку 4.2.

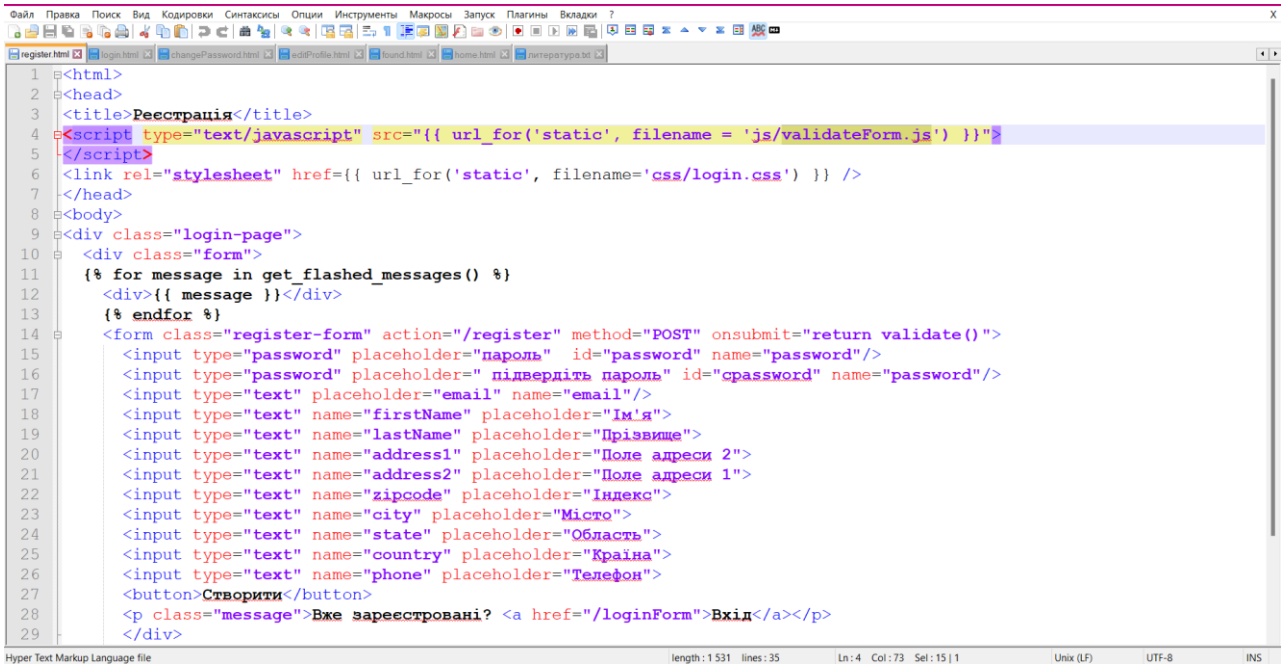


Рисунок 4.2 – Вигляд інтерфейсу редактора Notepad++

Джерело: авторська розробка

4.2 Огляд мов програмування

4.2.1 Мова програмування Python

В даний час Python користується великим попитом. Він широко використовується в індустрії розробки програмного забезпечення. Для цього є кілька причин:

– Python це високорівнева об'єктно-орієнтована мова програмування;

– швидка розробка додатків: завдяки лаконічну коду і дослівному синтаксисусу, розробка додатків прискорюється, простота коду допомагає скоротити час і витрати на розробку;

– динамічний машинописний текст: Python має високорівневі вбудовані структури даних, змішані з динамічним машинописним текстом, є можливість зручної прив'язки типів даних.

До унікальних особливостей, які роблять Python найпоширенішою мовою серед спільноти розробників відносять:

– підтримку можливості повторного використання коду і модульність;

– швидкість циклу редагування-перевірки-налагодження;

– власний відладчик, написаний на самому Python, що надає більшої чутливості до відладки програми;

– Python включає в себе безліч сторонніх компонентів, присутніх в індексі пакетів Python (PyPI) [16].

4.2.1 Мова програмування JavaScript

JavaScript – це текстова мова програмування, що використовується як на стороні клієнта, так і на стороні сервера, й дозволяє інтерактивно проектувати веб-сайти. У той час як HTML і CSS надають веб-сторінці структуру і стиль, JavaScript забезпечує інтерактивні елементи. При цьому кожна змінна має ім'я, тип і значення. JavaScript використовується для різних додатків в Інтернеті, таких як автоматично оновлювані новинні стрічки, форми, функції пошуку і інші інтерактивні функції. Майже всі інтерактивні функції, які можна знайти на веб-сторінках, створюються за допомогою JavaScript. Оскільки він підтримується всіма основними браузерами, JS широко використовується в розробці інтерфейсів.

JavaScript може використовуватися як на стороні клієнта, так і на стороні сервера. JavaScript на стороні клієнта робить можливим маніпулювання браузером і веб-сторінками. Він запускає скрипт в браузері користувача і є однією з найбільш поширених форм використання JavaScript. Серверний JavaScript працює не в браузері, а на сервері. Це дозволяє запобігти багато проблем SEO, які може викликати JavaScript. Тим не менш, він також займає багато серверних ресурсів. Гібридний рендеринг відображає важливі ресурси

на стороні сервера і менш важливий контент на стороні клієнта. Це розподіляє навантаження між браузером клієнта і сервера, в той час як сканери можуть миттєво бачити важливий контент [17].

4.3 Огляд мов розмітки та стилів

4.3.1 Мова розмітки HTML

HTML – це заснований на тексті спосіб опису структурованого вмісту, що міститься в HTML-файлі. Для опису вмісту веб-сторінки використовуються теги, які показують форматований вивід тексту, тип відображуваного вмісту на веб-сторінці. HTML файл містить певний синтаксис, що показує комп'ютеру та веб-серверу, що він знаходиться у форматі HTML і повинен бути прочитаний як web-сторінка.

HTML є офіційною рекомендацією консорціуму всесвітньої павутини (W3C) і, як правило, дотримується всіма основними веб-браузерами, включаючи як настільні, так і мобільні веб-браузери. HTML5 є останньою версією специфікації [18].

4.3.1 Мова стилю сторінок CSS

Каскадні таблиці стилів (CSS) – це мова програмування, яка дозволяє визначати дизайн електронних документів. Прості інструкції, представлені в чітких вихідних кодах, дозволяють налаштовувати елементи веб-сторінки, такі як макет, колір і типографіка.

Технологію CSS використовують для позначення поділу вмісту веб-сторінок і способу його представлення, що дає безліч переваг, таких як:

– можливість представити підсумковий документ в різних стилях (екранний, Голосовий, друкований);

– можливість мати адаптивний веб-сайт;

– не робити файли занадто важкими;

– визначати візуальний стиль всього веб-сайту;

– можливість забезпечувати більшу гнучкість і контроль в специфікаціях веб-сайту;

– спрощує створення сторінки [19].

4.4 Огляд Веб-фраймворку Flask

Flask – це полегшена платформа веб-додатків з інтерфейсом шлюзу веб-сервера (WSGI), що написана на Python. Це дає розробнику широкий вибір при розробці веб-додатків, необхідні інструменти для створення і розгортання веб-додатків. Він не нав'язує ніяких залежностей і не надає фіксовану структуру проекту, як інші веб-фреймворки Python, такі як Django. Це додає переваг, роблячи його доступним для використання багатьма розробниками. Веб-сайт може бути для чого завгодно, але, тим не менш, він дає розробникам можливість використовувати деякі розширення, надані спільнотою, що дозволяє додати більше функціональності в веб-додаток. Для Flask створено багато розширень для: аутентифікації користувачів, обробки завантажених файлів, створення панелі адміністратора.

Нижче наведені деякі з переваг використання Flask:

– простота у використанні: фреймворк Flask простий для розуміння, що робить його ідеальним для початківців;

– дуже гнучкий: більшість компонентів можуть бути змінені;

– тестування: він дозволяє проводити модульне тестування завдяки інтегрованій підтримці, вбудованому серверу розробки, швидкому відладчику і диспетчеризації запитів restful [20].

4.5 Огляд системи управління базою данних sqlite3

SQLite – це бібліотека в процесі, яка реалізує автономний, бессерверний, транзакційний механізм бази даних SQL з нульовою конфігурацією. Код для SQLite знаходиться в суспільному надбанні і, таким чином, вільний для використання в будь-яких цілях, комерційних чи приватних.

Серед характерних відмінностей SQLite можна виділити:

– база даних SQLite – це один звичайний дисковий файл, який може бути розташований у будь-якому місці ієрархії каталогів;

– формат файлу SQLite є кросплатформним: файл бази даних, записаний на одній машині, може бути використаний на іншій машині з іншою архітектурою;

– SQLite дозволяє користувачеві зберігати будь-яке значення будь-якого типу даних в будь-якому стовпці незалежно від оголошеного типу цього стовпця.

4.6 Реалізація клієнта сайту

4.6.1 Створення дизайну сайту

Від того як виглядає головна сторінка інтернет-магазину залежить, чи продовжить користувач знайомство з брендом і, як наслідок, рівень продажів. Важливим є не тільки дизайн, а й структура – наявність і коректний порядок потрібних блоків позитивно впливають на позиції в пошуковій видачі.

Структура розроблюваного сайту складається з декількох частин: хедера, елементів навігації, контента. Також для більш наочного розмежовування частин сайту використовується закруглення країв. Для виводу тексту було обрано декілька шрифтів, що використовують гротексний напис:

– Akzidenz Grotesk BQ Medium;

– Open Sans;

– Seaweed Script.

Кольором елементів навігації було обрано темно-зелений. Він транслює свіжість, аромат і цілющу вологість рослинного світу. Цей колір робить мінімалістичний дизайн більш живим. В якості основного кольору було обрано нейтральний жовто-зелений, що чудово поєднується з темно-зеленим.

Хедер, тобто верхня частина сайту, вигляд якого зображений на рисунку 4.3, відображається майже на всіх сторінках сайту, тому в нього вбудовано основні елементи навігації сторінки:

– логотип сайту, що повертає на головну сторінку;
– кнопка та поле пошуку;
– кнопка авторизації;
– віджет корзини (один з варіантів розташування);
– поле з інформацією: контакти з магазином та графік роботи.

Рисунок 4.3 – Вигляд хедру сайту інтернет-магазину

Джерело: авторська розробка

Контент займає більшу частину сторінки й складається з двох частин: списку жанрів та каталогу товарів.

Список жанрів, що зображено на рисунку 4.4, оформлено у вигляді маркованого списку, також додана анімація при наведенні курсором до конкретного жанру.

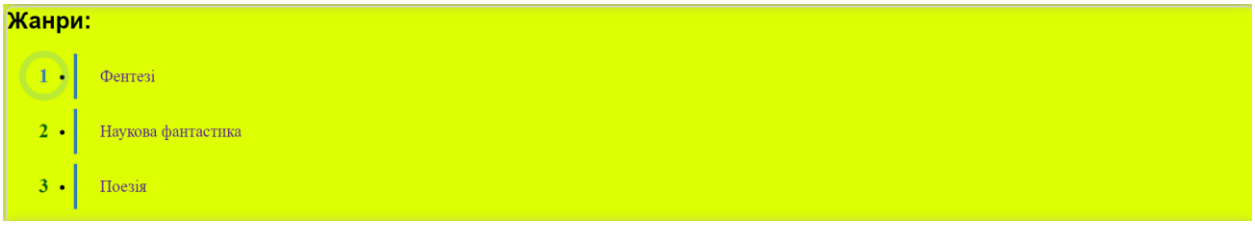


Рисунок 4.4 – Вигляд списку жанрів

Джерело: авторська розробка

Каталог складається з карток товарів (рис. 4.5), що включає три частини: назву, обкладинку, ціну з можливістю додати товар до корзини. Якщо натиснути на обкладинку, то відкриється детальний опис книги. Для більш наочного відображення назви використовувалися тіні, також до обкладинки додано зелене підсвічування.

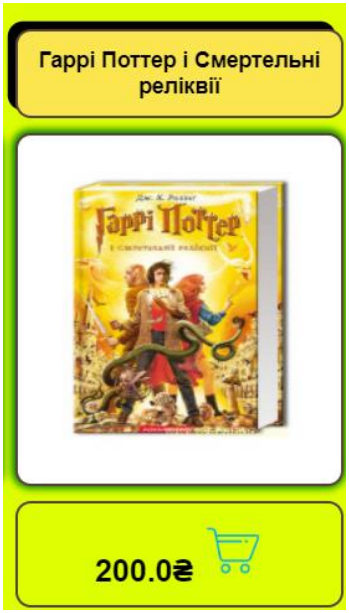


Рисунок 4.5 – Картка товару

Джерело: авторська розробка

Для елементу віджета корзини використовується A / B тестування, тому існує два дизайну цього елементу: традиційний з розташування у футері (рис. 4.3) та інший варіант – круглий віджет, що закріплений внизу зліва сторінки. Другий варіант корзини зображено на рисунку 4.6.

Рисунок 4.6 – Круглий варіант віджета корзини

Джерело: авторська розробка

Дизайн сторінки адміністратора дещо відрізняється від сторінок користувача: панель з елементами управління знаходяться вертикально зліва та при наведенні на окремий елемент відбувається підсвічування червоним кольором. Вигляд цієї панелі зазначено на рисунку 4.7

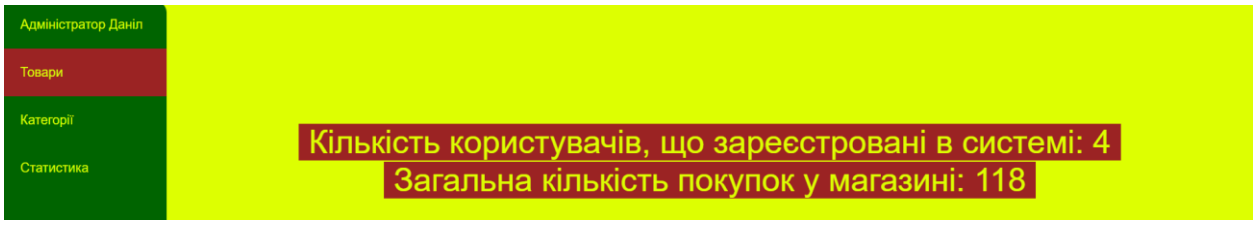


Рисунок 4.7 – Дизайн панелі адміністратора

Джерело: авторська розробка

4.6.1 Розробка програмної реалізації сайту

Для створення програмного клієнта сайту було обрано фреймворк Flask, створено об’єкт типу Flask – WSGI-додаток (WSGI – це інтерфейс для взаємодії сервера з фреймворком) котрий показує, що сервер передає всі отримані запити екземпляру для подальшої обробки. Також для роботи web-додатку необхідно сховище даних, тому перед запуском серверу Flask, створено файл, що генерує відповідні таблиці, що зазначені у розділі 3.2. Для роботи сайту достатньо один раз згенерувати сховище даних.

Для взаємодії функцій зі сторінками web-додатку використовуються маршрути, що прикріпляються до шаблонів URL-адрес програми. В Python існує декоратор який Flask надає для легкого призначення маршруту, тобто URL-адреси відповідної функції, наприклад @app.route("/") призначується до головної сторінки інтернет-магазину. Якщо користувач переходить на певну сторінку web-додатку, то в залежності від маршрута сторінки за допомогою декоратора визивається відповідна функція, що генерує сторінку. Дана функція повертає відповідний HTML шаблон та передає в нього необхідні дані, які попередньо були згенеровані: витягнуті з бази даних, обчислені функцією. Для обчислень іноді необхідно передати дані з шаблону до функції: у коді HTML форми визивається необхідна функція, де за допомогою функції request.form() передається вміст форми. Також більшість web-сторінок взаємодіють з сховищем бази даних, для цього викоистовуються запроси, які витаскують з необхідної таблиці дані, що необхідно відобажати на сторінці,

використовуючи оператор запиту SELECT. Для сторінки пошуку використовується оператор запиту LIKE, котрий знаходить збіг у назві або його частину у відповідному полі таблиці. Якщо ж на сторінку не потрібно передавати дані, то замість повернення шаблону сторінки використовується функція redirect(), що посилається на URL-адресу, котра генерує необхідну сторінку.

Також слід розглянути поняття сесії, що використовується у цьому web-додатку. Сесія необхідна для зберігання інформації, що відноситься до користувача, коли він взаємодіє з web-додатком. Для кожного користувача створюється нова сесія. Для зберігання цієї інформації використовуються файли cookie – невеликі фрагменти даних, що зберігаються на комп'ютері веб-браузером. У cookie файл записуються дані про авторизацію: email, id користувача. Ці дані зберігаються у словнику і можуть бути використані у будь-якій функції. Якщо користувач вже увійшов у систему, то у cookie записується email користувача. Надалі, виконується перевірка чи записані дані про авторизацію, якщо вони присутні, то сторінка з авторизацією не виводиться, в іншому випадку, наприклад, неавторизований користувач при додаванні товару у корзину побачить поле авторизації у систему.

У даному web-додатку cookie файли використовуються також для проведення A / B тестування – в них зберігаються відповідні статистичні дані.

Дані, збережені для сесії, є тимчасовими, так як в кінцевому підсумку термін дії сервісу закінчиться, після чого всі cookie файли зтираються.

Окремо слід розглянути процес авторизація та реєстрації, оскільки це досить важлива частина взаємодії з користувачем. Якщо користувач ввів персональні дані до сторінки реєстрації, то необхідно перевірити, що це новий користувач: для цього виконується запит у БД до таблиці користувачів: якщо email вже існує, то користувачеві виводиться повідомлення з помилкою. Вивід повідомлення у форму реалізовано за допомогою миттєвих повідомлень Flash, що є вбудованими у фремйворк Flask. У полі реєстрації є перевірка на повторне введення створюваного паролю користувача, для цього

використовується JavaScript, у коді якого перевіряється, що паролі однакові, при неспівпадінні паролю виводиться спливаюче вікно, про неправильність введених даних. Аналогічна схема використовується для сторінки зміни паролю. Для збереження приватності користувача пароль, що записується у БД шифрується. Шифрування відбувається за допомогою бібліотеки hashlib: пароль хешується алгоритмом md5 та переводиться до шістнадцяткової систем числення й у такому вигляді записується до БД. Оскільки на сайті інтернет-магазину також є панель адміністратора, то виникла необхідність розділити користувачів на декілька ролей, тому під час реєстрації, з сторінки користувача, у БД записується також його роль, в даному випадку як користувача.

Для авторизації використовується функція, котра перевіряє існування користувача з такими даними. Виконується запит у БД, де порівнюється значення email та паролю. Оскільки пароль у БД зашифрований, то спочатку виконується його розшифрування, значення якого потім порівнюється з введеним значенням. При невідповідності даних користувачеві виводиться попередження, про неправильність введених даних.

4.7 Створення панелі адміністрування

Панель адміністратора є важливою частиною web-додатку, оскільки вона значно спрощує роботу з каталогом товарів й позбавляє необхідності редагування бази даних напряму через файл. Також у панелі адміністратора розроблено панель з виведенням статистики використання ресурсу.

Оскільки доступ до панелі адміністратора необхідно надавати тільки працівникам інтернет-магазину, то при вході до панелі виконується перевірка авторизації користувача. Окрім перевірки даних аккаунту, також для входу необхідно ввести дані акаунту, у якого роль це адміністратор. Для цього у коді сторінки авторизації виконується запит до БД, для перевірки ролі користувача, якщо роль не підходить, то виводиться повідомлення про помилку. Інші дані перевіряються аналогічно до звичайної авторизації. Якщо ж адміністратор

хоче створити новий аккаунт, то при збереженні даних з панелі адміністратора в якості ролі зберігається відповідна роль.

На головній сторінці панелі адміністратора виводяться, статистичні дані: кількість зареєстрованих користувачів та кількість покупок, для цього використовується запит у БД, де використовується оператор запиту COUNT, що повертає кількість полів у таблицях користувачів та транзакцій.

Наступним елементом навігації є каталог товарів: на сторінку виводяться дані вже існуючих товарів, що витягуються з БД за допомогою оператор запиту COUNT. На цій сторінці також є посилання, до сторінки реєстрації. Окрім текстових даних про книгу, форму необхідно завантажувати файл обкладинки, для фільтрації неправильних форматів файлів, створено масив з дозволеними розширеннями зображень. Збережене зображення переміщається до каталогу з зображеннями, що знаходиться у папці з сервером. Для більшої зручності було створено випадаючий список з вибором жанру товару. Також будь-який товар можливо видалити, для цього при натисканні на відповідну функції виконується функція, яка створює запит у БД з використанням оператору запиту DELETE, даний оператор видаляє необхідний товар з таблиці.

Аналогічно створено елемент навігації з категоріями товарів: налаштовано перегляд категорій, додавання категорій, їх видалення.

Останнім елементом навігації є статистика, у цьому розділі можливо налаштовувати експеримент з проведення A / B тестування. Для того, щоб розпочати проведення експерименту з A / B тестування необхідно розрахувати мінімальну вибірку користувачів, що розраховується за формулою 1.4, яка наведена у розділі 1.5. Для введення потужності та рівня значущості використано повзунки, з кроком в 1 відсоток. Для виведення поточного значення використано JavaScript, який при перетягуванні повзунка змінює значення відсотку та повертає його до сторінки.

Також для початку експерименту необхідно введення даних початку й кінця проведення експерименту, для цього використано спеціалізовану форму.

Після натискання кнопки, що відповідає про початок експерименту відбувається збереження цих даний у файл формату csv, який є досить зручним форматом для зберігання табличних даних. Опис налаштування виводу результатів тестування наведено у розділі 4.8.

4.8 Створення модуля A / B тестування

В кожній сесії, які створюють користувачі можливо виведення обох версій інтерфейсу: основного та тестового. Варіант який користувач побачить при вході в web-додаток генерується випадково з ймовірністю 50% для кожного варіанту. Далі значення варіанту інтерфейсу додається до cookie файлу, що зберігається для кожної сесії окремо. На кожній сторінці відбувається перевірка для виводу правильного варіанту інтерфейсу.

Найважливішою частиною A / B тестування є збір даних про дії користувача. Для зберігання цих даних використовуються cookie файли: якщо користувач увійшов у систему зберігається id користувача, при проведенні транзакції зберігається помітка, що користувач зробив покупку, також зберігається дата й година відвідування сайту й номер сесії. Оскільки неможливо перевірити коли користувач припинить користуватися web-додатком, то дані про сесію записуються кожний раз коли користувач заходить на головну сторінку. Ці дані для більш простого маніпулювання зберігаються у файл формату csv, з розділенням на стовбці. Збір даних припиняється, якщо досягнута необхідна дата кінця експерименту.

Якщо адміністратор заходить на сторінку зі статистикою експерименту, то відбудеться аналіз зібраних даних. Для сортування статистичних даних використано бібліотеку Pandas. Дані зчитуються з csv файлу й записуються у фрейм даних Pandas: структуру, що містить двовимірні дані і відповідні їм мітки. Оскільки користувач міг декілька раз виконувати однакові дії, то в

статистичних даних багато дублікатів, для їх видалення використано функцію Pandas drop_duplicates(), що залишає перше входження екземпляру. Також необхідно знайти найповніші дані сесії, для цього у функцію Pandas drop_duplicates() додано сортування за полем з id сесії: видаляються всі записи з однаковим id, окрім останнього, що є найповнішим. На останньому етапі необхідно відсортувати записи за датою: для кожної дати сформовано окремий підмасив, що знаходяться у фреймі даних.

Далі необхідно за формулою 1.2 розрахувати рівень конверсії кожної дати: для тестового та основного варіантів. Рівень конверсії для кожної дати та сама дата записується у відповідні масиви.

На наступному етапі на основі отриманих даних рендериться графік: для його відображення було обрано бібліотеку bokeh. Дана бібліотека добре встраюється на шаблон сайту й має багато зручних інструментів: зміну масштабу, збереження зображення графіку, закріплення орієнтації, перегляд координат точок графіку. В якості координати X виводиться дата запису даних, координата Y виводить конверсію. Окремо на сайті виводиться поточні дані для елементів, що тестуються: конверсія, кількість покупок та відвідувань версій сайту. Варто зазначити, що при малій кількості отриманих статистичних даних, графік та загальні дані експерименту не можливо відобразити, тому виводиться тільки порожня координатна площина.

5 РОБОТА КОРИСТУВАЧА З WEB-САЙТОМ

Для роботи з сайтом у користувача має буди будь-який сучасний браузер з підтримкою необхідних web-технологій, наприклад Google Chrome.

5.1 Інструкція з використання сайту користувачем

На головній сторінці інтернет-магазину, що зображено на рисунку 5.1, неавторизований користувач може виконати обмежену кількість дій: переглянути каталог, авторизуватися, виконати пошук товару.

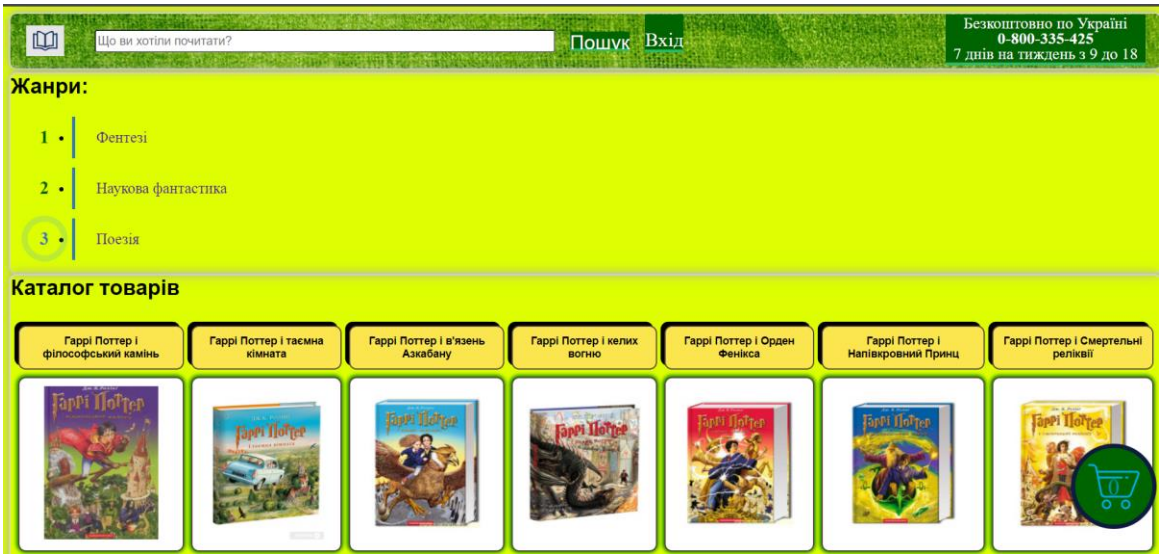


Рисунок 5.1 – Вигляд головної сторінки інтернет-магазину

Джерело: авторська розробка

Для входу в аккаунт необхідно натиснути зверху на панелі навігації кнопку «вхід», після чого з’явиться поле авторизації, що зображено на рисунку 5.2. Для успішної авторизації необхідно ввести email та пароль користувача та натиснути кнопку «вхід».

Якщо користувачу необхідно створити новий акаунт, потрібно натиснути кнопку «створити акаунт», після чого відкривається сторінка з формою реєстрації, яка зображена на рисунку 5.3. В формі необхідно ввести персональні дані: пароль, персональні дані, email, ім’я, прізвище, адресу,

індекс, місто, область, країну, телефон. Далі необхідно кнопку підтвердження, після чого вібувається перенаправлення на сторінку авторизації.

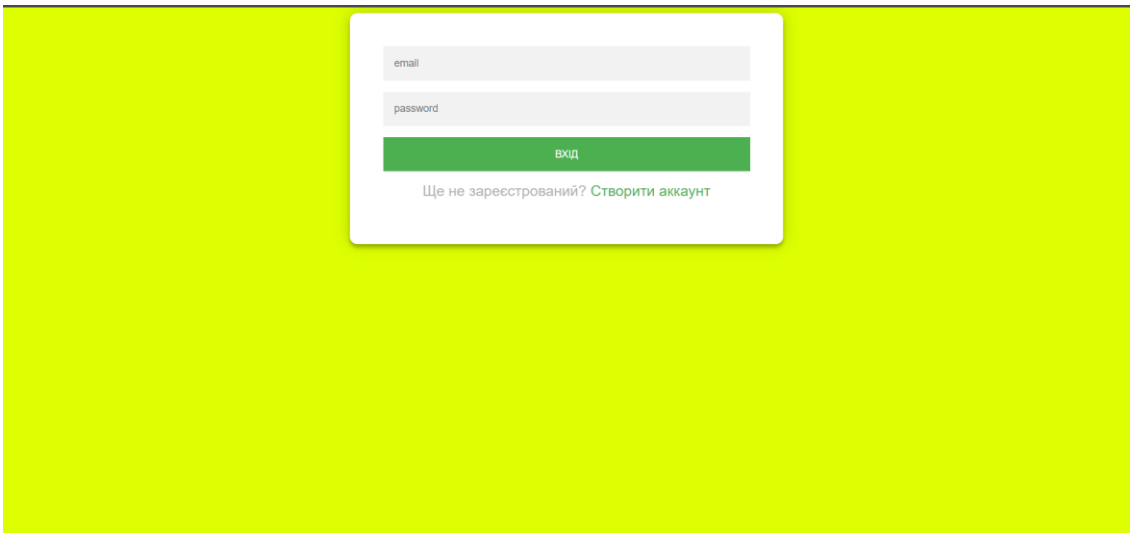


Рисунок 5.2 – Поле авторизації

Джерело: авторська розробка

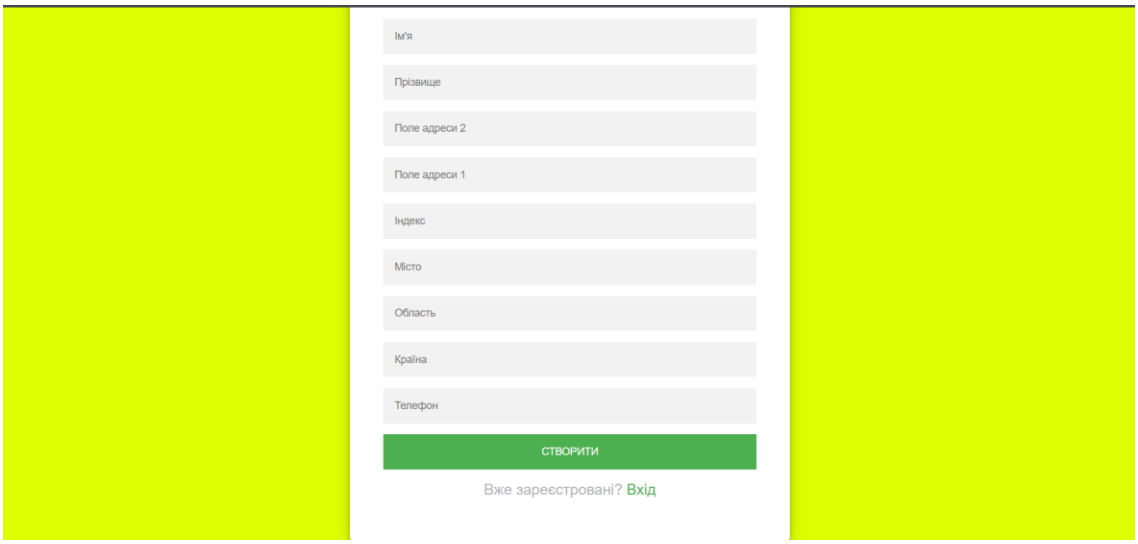


Рисунок 5.3 – Поле реєстрації

Джерело: авторська розробка

Для пошуку у каталозі необхідно у поле пошуку написати назву товару або його частину й натиснути кнопку «пошук», після чого відкривається сторінка зі знайденими товарами. Приклад сторінки зі знайденими товарами зображено на рисунку 5.4.

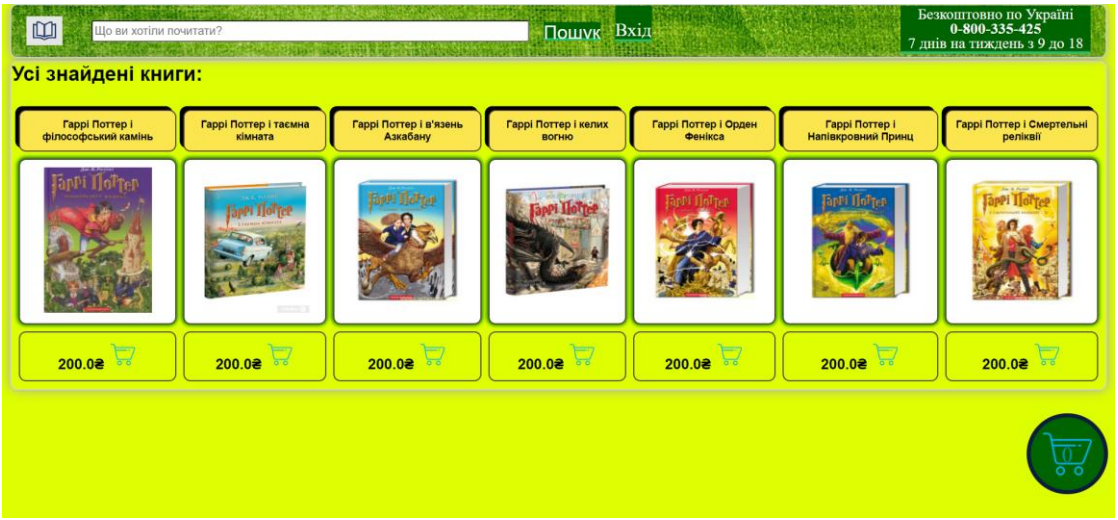


Рисунок 5.4 – Сторінка зі знайденими товарами

Джерело: авторська розробка

Якщо користувачу необхідно переглянути книги вибраного жанру, необхідно у маркованому списку натиснути на цей жанр, після чого відбувається перенаправлення до сторінки з списком відповідних книг. Приклад знайдених книг з жанром «Наукова фантастика» зображено на рисунку 5.5.

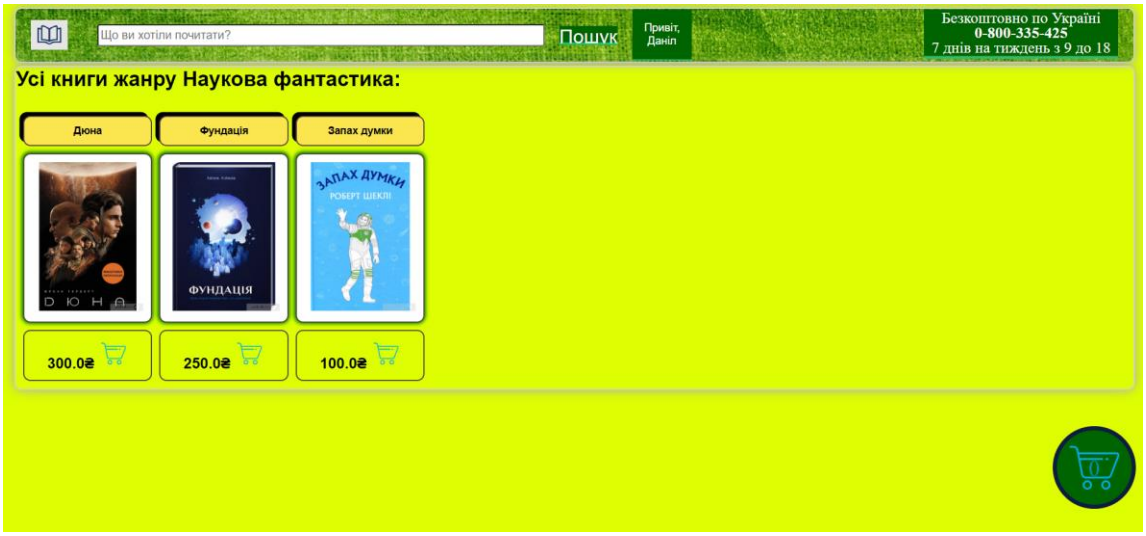


Рисунок 5.5 – Сторінка з книгами жанру «Наукова фантастика»

Джерело: авторська розробка

Для перегляду опису товару необхідно натиснути на обкладинку, після чого відбувається перенаправлення до сторінки опису товару (рис 5.6). На даній сторінці показана наступна інформація: автор, ціна кількість книг, опис книги. Також існує можливість додати товар до корзини.

Рисунок 5.6 – Сторінка опису товару

Джерело: авторська розробка

При необхідності у користувача є можливість змінити персональні дані, для цього необхідно натиснути на кнопку з ім’ям користувача, після чого виводиться випадаючий список, де є можливість вийти з акаунта, або перейти до персонального профілю. На сторінці профіля можливо змінити дані профіля або змінити пароль (рис. 5.7).

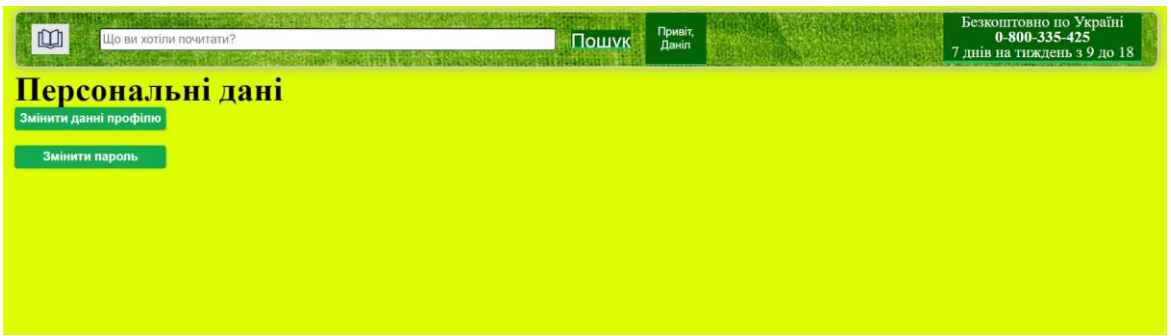


Рисунок 5.7 – Сторінка персонального профіля

Джерело: авторська розробка

Якщо користувач натисне кнопку «змінити пароль», відкриється сторінка зміни паролю (рис. 5.8), де необхідно ввести старий та новий пароль, після чого відбудеться перенаправлення до сторінки профілю.

Також користувач має змогу змінити персональні дані, якщо натисне кнопку «змінити персональні дані». Далі відкриється сторінка з формою для зміни даних (рис. 5.9), де відображаються старі дані, після чого користувач може змінити будь-яке поле. Для збереження даних необхідно натиснути кнопку «зберегти».

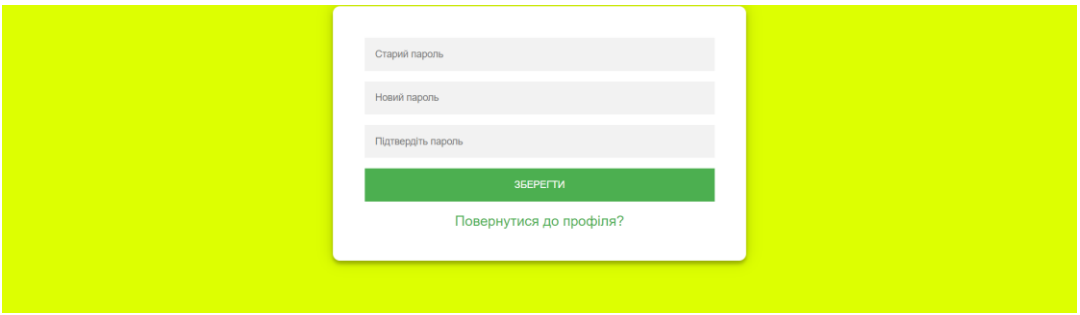


Рисунок 5.8 – Форма зміни паролю

Джерело: авторська розробка

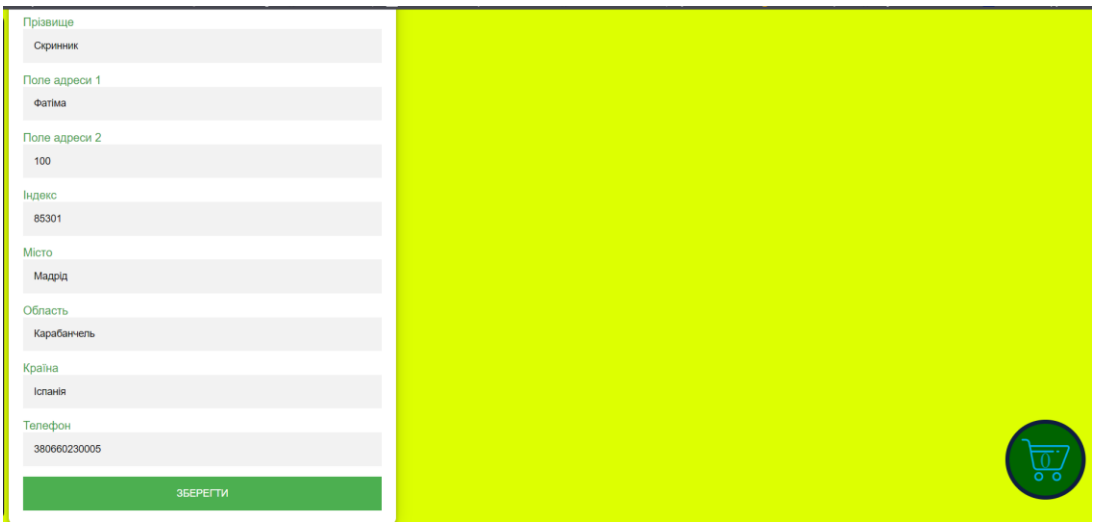


Рисунок 5.9 – Форма для зміни даних користувача

Джерело: авторська розробка

Для перегляду вмісту корзини авторизованому користувачеві необхідно натиснути на кнопку корзини, після чого відбувається перенаправлення до вмісту корзини, який зображено на рисунку 5.10. Користувач має можливість видалення товару з корзини, для цього необхідно натиснути відповідну кнопку. При необхідності можливо сплатити товари, натиснувши кнопку «до сплати». Після завершення користувач побачить сторінку з результатами транзакції (рис. 5.11), також на цій сторінці є посилання до головної сторінки.

Рисунок 5.10 – Вміст корзини

Джерело: авторська розробка

Рисунок 5.11 – Сторінка з результатами транзакції

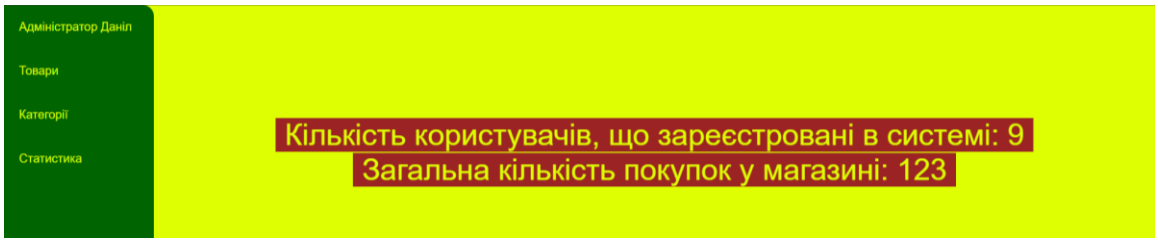
Джерело: авторська розробка

5.2 Інструкція з використання панелі адміністратора

На головній сторінці панелі адміністратора виводяться дані з статистики користування сайту (рис 5.12). З цієї сторінки через панелі навігації можливо перейти до інших сторінок для модерації ресурсу: редагування товарів та категорій, зібрання статистичних даних.

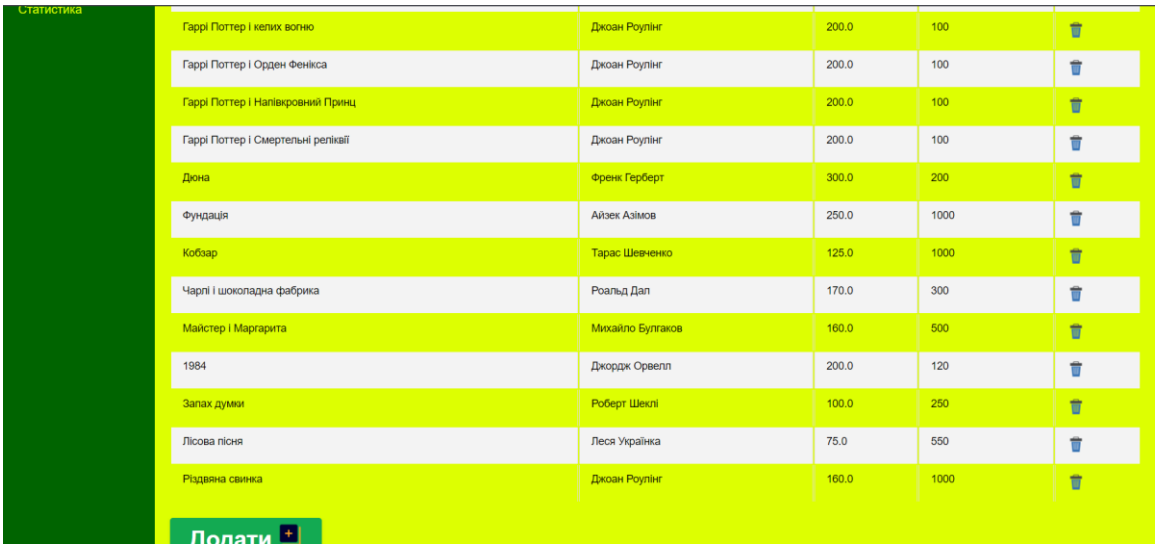
При необхідності редагувати товари каталогу, використовується вкладка «товари», вигляд якої зображено на рисунку 5.13. В даній вкладці можливо

видалити товар: для цього необхідно натиснути на кнопку у вигляді корзини, що розташована поряд з відповідним товаром. Якщо необхідно додати новий товар, то користувач повинний натиснути на кнопку «додати», далі відбувається перенаправлення до форми, де необхідно ввести дані створюваного товару. Після завершення вводу даних необхідно натиснути кнопку збереження даних. Вигляд даної форми зображено на рисунку 5.14.



Рисунокс 5.12 – Сторінка з даними статистики користування сайту

Джерело: авторська розробка



Рисунокс 5.13 – Вкладка редагування товарів каталогу

Джерело: авторська розробка

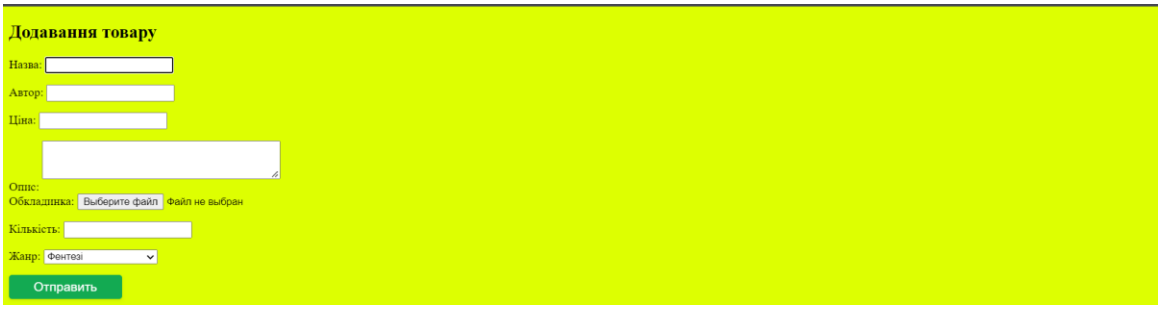


Рисунок 5.14 – Форма додавання нового товару

Джерело: авторська розробка

Аналогічно при необхідності взаємодії з жанрами каталогу адміністратор повинен перейти до вкладки «категорії» В цій вкладці адміністратор може видалити каталог або додати новий, для чого необхідно ввести назву каталогу у відповідне поле й натиснути кнопку зберегти. Вигляд сторінки взаємодії з каталого м зображено на рисунку 5.15.

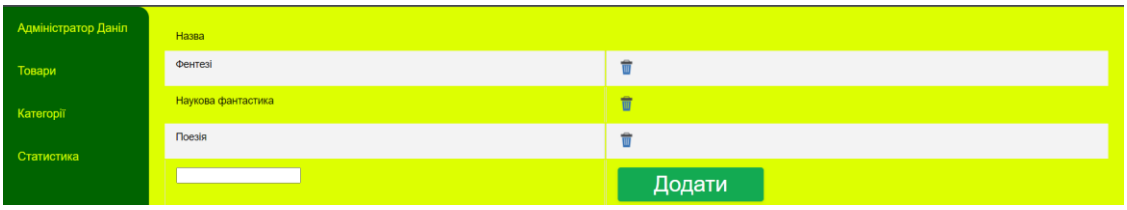


Рисунок 5.15 – Вкладка редагування категорій товарів

Джерело: авторська розробка

У останній вкладці «Статистика» користувач може провести A / B тестування, для цього необхідно ввести необхідні дані для початку експерименту: потужність, рівень значущості, поточний та необхідний коефіцієнт конверсії, дати початку та кінця експерименту. Для введення потужності та рівня значущості необхідно у повзунку вибрати необхдідні значення відсотків, для введення дат необхідно у випадаючому списку вибрати необхідну дату. Налаштування експерименту зазначено на рисунку 5.16.

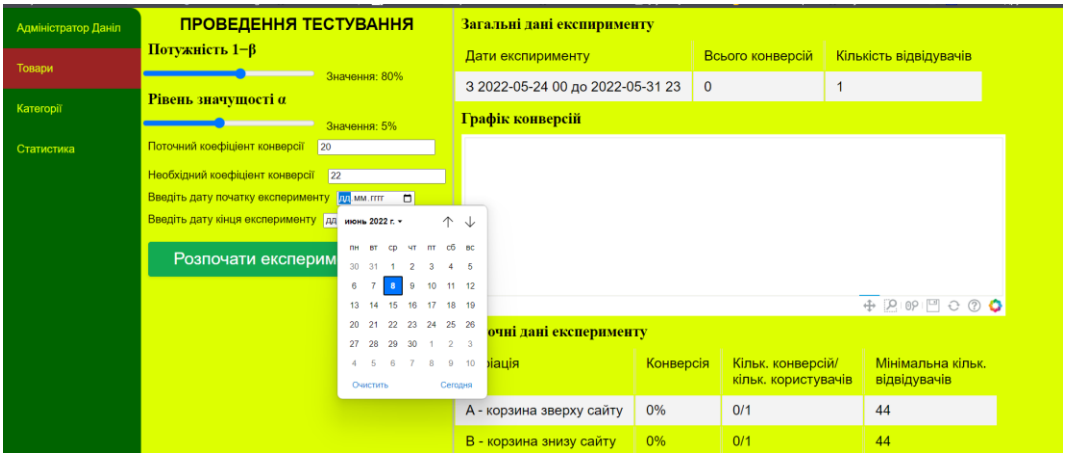


Рисунок 5.16 – Налаштування проведення експерименту

Джерело: авторська розробка

Після натискання кнопки розпочати експеримент користувач у правій частині екрану побачить поточні деталі з проведення тестування (рис. 5.17).

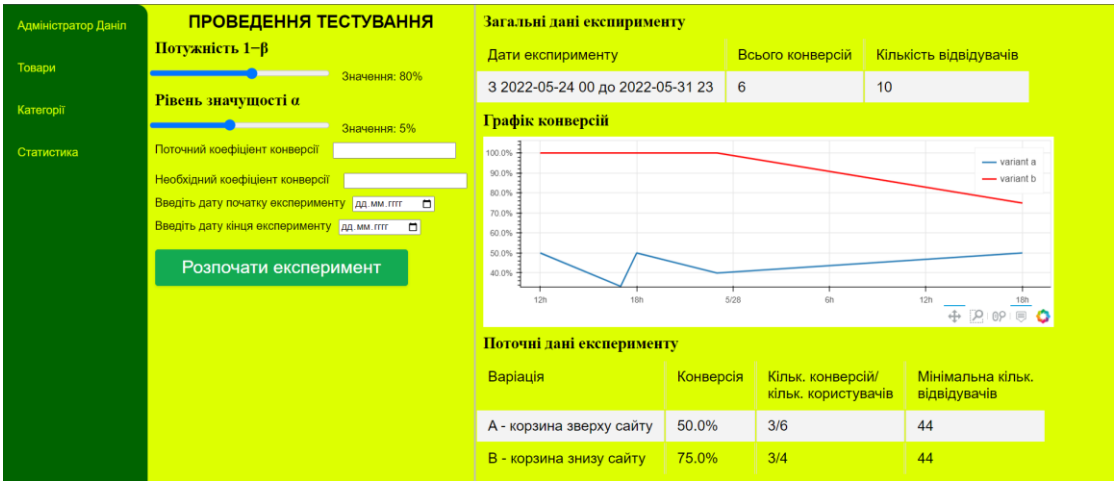


Рисунок 5.18 – Перегляд результат експерименту

Джерело: авторська розробка

5.3 Тестування додатку A / B тестування

Для тесту роботи алгоритму A / B тестування створено експеримент, з наступними даними: потужність – 80%, рівень значущості – 5%, датою початку експерименту вибрано 24.05.22, кінця – 31.05.22.

Для проведення тестування в зазначені дати, було здійснено декілька відвідувань сайту у різні частини дня. Оскільки у cookie файлах браузеру до

його закриття зберігаються дані сесії з фіксованим типом елементу інтерфейсу, що використовується у експерименті, тому для тестування різних версій цього елементу необхідно використовувати режим інкогніто браузера. У цьому режимі браузер не пам'ятає дані сесії, тому кожне нове вікно створює нову сесію сайту з різним варіантом елементу, що тестується.

Після завершення експерименту було сформовано графік з рівнями конверсій за датами, який зображено на рисунку 5.19. З цього графіку видно, що переміг варіант інтерфейсу «корзина зверху сайту», оскільки остаточний рівень конверсії цього елименту сягає 56%, коли конверсія варіанту інтерфейсу «корзина знизу сайту» є дещо меншою: 41%.

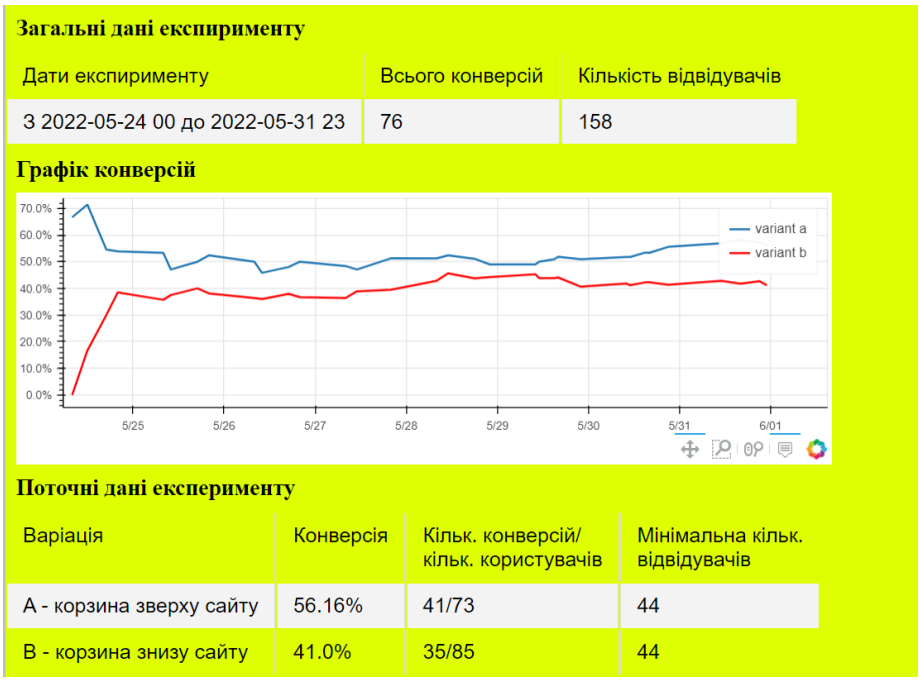


Рисунок 5.19 – Графік конверсій за датами

Джерело: авторська розробка

ВИСНОВКИ

У ході виконання дипломної роботи було розроблено web-додаток інтернет-магазину з вбудованим модулем A / B тестування. Проаналізовано призначення, характеристики та параметри проведення експерименту з A / B тестування, розгянуті типові підходи до A / B тестування. Також проведено аналіз існуючих сервісів з A / B тестування.

Було спроектовано логічну та фізичну модель web-додатку. Для цього створено декілька UML діаграм, де описано взаємодію користувача системою, сплановано необхідність створення програмних компонентів та описано їх взаємодію. Також розроблено базу даних: описано поля таблиць та їх призначення. Розроблено ER-діаграму взаємодії таблиць між собою.

У процесі розробки було реалізовано усі необхідні підсистеми веб-додатку. Створено панель адміністратора, в якій можливо редагувати каталог товарів, переглядати та проводити експерименти з A / B тестування. В звіті детально розглянуто деталі взаємодії клієнта web-додатку з користувачем. Розібрано особливості проведення A / B тестування: збір статистики під час тестування, аналіз зібраних даних, їх вивід на сторінку панелі адміністратора.

Основною перевагою розробленого сервісу, є досить простий процес проведення A / B тестування. Так як користувач налаштовує тестування не в окремому сервісі, а за допомогою панелі адміністратора, у деяких випадках це може бути набагато зручніше, і при цьому легко дає можливість провести A / B тестування ввівши лише необхідні параметри експерименту, з мінімальними знаннями мов програмування для вибору елементів тестування.

Під час експлуатації web-додатку не використовуючи дорогі професіональні сервіси з A / B тестування. Також цей модуль є вбудованим в клієнт сайту, що дозволяє при необхідності запускати тест за короткий проміжок часу, не гаючи час на громіздкі налаштування. Великою перевагою

є те, що для проведення тестування використовується невеликий локальний код, що не сповільнює роботу всього web-додатку.

З головної переваги витікає й недолік сервісу – порівняно невеликий функціонал в зрівнянні з професійними сервісами типу Google Analytics. Але метою дипломної роботи ставилась створення простого web-додатку інтернет-магазину з базовими елементами статистики. В подальшому даний додаток необхідно покращити: додати тестування для певних груп користувачів, наприклад з різних країн, або провести розмежовування для пристроїв. Також необхідно додати можливість проведення тестування декількох елементів інтерфейсу паралельно. Потрібно розглянути можливість покращити алгоритм A / B тестування, вибрати більш досконалі методи дослідження.

ПЕРЕЛІК ПОСИЛАНЬ

1. A / B Testing: A Complete Guide to Statistical Testing [Електронний ресурс]. – Режим доступу: https://towardsdatascience.com/a-b-testing-a-complete-guide-to-statistical-testing-e3f1db140499

2. A / B Testing: An Actionable Step-By-Step Guide For CRO [Електронний ресурс]. – Режим доступу: https://academy.pagefly.io/guide/ab-testing-guide-for-cro/

3. CTA (Call to Action): формування ефективного заклику до дії [Електронний ресурс]. – Режим доступу: https://www.interkassa.com/blog/cta-call-to-action-formirovanie-effektivnogo-prizyva-k-deystviyu/

4. Як створити CTA за допомогою -тестування [Електронний ресурс]. – Режим доступу: https://blog.mailup.com/2017/09/ab-test-cta/

5. Intuition Behind A / B Sample Sizing [Електронний ресурс]. – Режим доступу https://towardsdatascience.com/intuition-behind-a-b-sample-sizing-cb7e9c4fb992

6. Великий гайд з A / B тестування [Електронний ресурс]. – Режим доступу: https://habr.com/ru/company/boodet_online/blog/498688/

7. Questions on Enterprise Google Analytics 360 Suite [Електронний ресурс]. – Режим доступу: https://www.blastanalytics.com/blog/google-analytics-360-top-5-question

8. A / B тестування Google Optimize [Електронний ресурс]. – Режим доступу: https://outsourcing.team/uk/blog/ppc/vse-pro-a-b-testuvannya/

9. Collaborate on ideas with Program Management [Електронний ресурс]. – Режим доступу: https://support.optimizely.com/hc/en-us/articles/4410288455053-Collaborate-on-ideas-with-Program-Management

10. Optimizely: Things You Need to Know Before You Opt for It [Електронний ресурс]. – Режим доступу: https://www.brillmark.com/optimizely-things-you-need-to-know-before-you-opt-for-it-in-2020/

11. Run A / B tests It [Електронний ресурс]. – Режим доступу: https://docs.developers.optimizely.com/full-stack/docs/run-a-b-tests

12. What is UML | Unified Modeling Language [Електронний ресурс]. – Режим доступу: https://www.edrawsoft.com/what-is-uml-diagram.html?gclid=CjwKCAjwv-GUBhAzEiwASUMm4tAk0lQs_5ZLGF_8cdJu4RUNeTDxN_5EvXZHTO2UODo-uEqfKbg1ERoC7VgQAvD_BwE

13. Component Diagrams [Електронний ресурс]. – Режим доступу: https://www.edrawsoft.com/uml-component.html

14. UML Use Case Diagram [Електронний ресурс]. – Режим доступу: https://www.edrawsoft.com/uml-use-case.html

15. What is PyCharm? [Електронний ресурс]. – Режим доступу: https://intellipaat.com/blog/what-is-pycharm/

16. About Notepad++ [Електронний ресурс]. – Режим доступу: https://www.softwareadvice.com/app-development/notepad-profile/

17. About JavaScript [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

18. What is HTML – Definition and Meaning of Hypertext Markup [Електронний ресурс]. – Режим доступу: Language https://www.freecodecamp.org/news/what-is-html-definition-and-meaning/

19. What is CSS? [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS

20. Introduction to Micro Web Framework Flask [Електронний ресурс]. – Режим доступу: https://medium.com/featurepreneur/introduction-to-micro-web-framework-flask-78de9289270b

Додаток А

Зауваження нормконтролера

Таблиця А.1 – Зауваження нормоконтролера

Сторінка	Позначення	Зміст зауваження
Зміст		Назви розділів прописними, нема назв додатків
С.10-11, 14,17-18		Невірні переліки
ПЗ		Формул и- змінні в тексті та в формулі однакові повинні бути , в редакторі формул
		Нещільне заповнення сторінок ПЗ

Додаток Б

Код клієнту web-додатку інтернет-магазину

Вміст файлу main.py:

```
from flask import *
import sqlite3, hashlib, os
from werkzeug.utils import secure_filename
from random import *
from numpy import*
from scipy.stats import norm
import csv
import pandas as pd
from csv import writer, DictWriter
from datetime import datetime
from bokeh.embed import components
from bokeh.plotting import figure
from bokeh.resources import INLINE
from bokeh.models import HoverTool, PanTool, BoxZoomTool, WheelZoomTool,
NumeralTickFormatter
app = Flask(__name__)
app.secret_key = 'random string'
UPLOAD_FOLDER = 'static/uploads'
ALLOWED_EXTENSIONS = set(['jpeg', 'jpg', 'png', 'gif'])
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
def getLoginDetails():
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        if 'email' not in session:
            loggedIn = False
            firstName = ''
            noOfItems = 0
        else:
            loggedIn = True
            cur.execute("SELECT userId, firstName FROM users WHERE email = '" + session['email']
+ "'")
            userId, firstName = cur.fetchone()
            session['userId'] = userId
            cur.execute("SELECT count(productId) FROM kart WHERE userId = " + str(userId))
            noOfItems = cur.fetchone()[0]
    conn.close()
    return (loggedIn, firstName, noOfItems
def rand():
    rand=randint(1, 2)
    if rand==2:
        return True
```

```
    return False
@app.route("/")
def root():
    if "ab" not in session:
        ab = rand()
        session["ab"]=ab
    if "userId" not in session:
        session['userId']=0
    if abparameters():
        writestatic()
    loggedIn, firstName, noOfItems = getLoginDetails()
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute('SELECT productId, name, price, description, image, stock FROM products')
        itemData = cur.fetchall()
        cur.execute('SELECT categoryId, name FROM categories')
        categoryData = cur.fetchall()
    itemData = parse(itemData)
    return render_template('home.html', itemData=itemData, loggedIn=loggedIn,
firstName=firstName, noOfItems=noOfItems,
                           categoryData=categoryData, ab=session["ab"])
@app.route("/add")
def addForm():
   with sqlite3.connect('database.db') as conn:
       cur = conn.cursor()
       cur.execute("SELECT categoryId, name FROM categories")
       categories = cur.fetchall()
   conn.close()
   return render_template('add.html', categories=categories)
@app.route("/addItem", methods=["GET", "POST"])
def addItem():
    if request.method == "POST":
        name = request.form['name']
        price = float(request.form['price'])
        description = request.form['description']
        stock = int(request.form['stock'])
        categoryId = int(request.form['category'])
        author = request.form['author']
        # Uploading image procedure
        image = request.files['image']
        if image and allowed_file(image.filename):
            filename = secure_filename(image.filename)
            image.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
        else:
            return redirect(url_for('addItem'))
        imagename = filename
        with sqlite3.connect('database.db') as conn:
            try:
                cur = conn.cursor()
                cur.execute(
                    '''INSERT INTO products (name, price, description, image, stock, categoryId,
author) VALUES (?, ?, ?, ?, ?, ?, ?)''',
                    (name, price, description, imagename, stock, categoryId, author))
                conn.commit()
```

```
            except:
                conn.rollback()
        conn.close()
        return redirect(url_for('Items'))
@app.route("/admin/items")
def Items():
    name= getLoginDetails()[1]
    if 'email' not in session or getrole(session["email"]) != 2:
        return render_template('Adminlogin.html', error='')
    else:
        with sqlite3.connect('database.db') as conn:
            cur = conn.cursor()
            cur.execute('SELECT productId, name, author, price, stock FROM products')
            data = cur.fetchall()
        conn.close()
        return render_template('items.html', data=data, name=name)
@app.route("/removeItem")
def removeItem():
    productId = request.args.get('productId')
    with sqlite3.connect('database.db') as conn:
        try:
            cur = conn.cursor()
            cur.execute('DELETE FROM products WHERE productID = ' + productId)
            conn.commit()
        except:
            conn.rollback()
    conn.close()
    return redirect(url_for('Items'))
@app.route("/admin")
def admin():
    name= getLoginDetails()[1]
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute("SELECT COUNT(*) as count FROM users")
        countusers = cur.fetchone()[0]
        cur.execute("SELECT COUNT(*) as count FROM orders")
        countorders = cur.fetchone()[0]
    if 'email' not in session or getrole(session["email"]) != 2:
        return render_template('Adminlogin.html', error='')
    else:
        return render_template('admin.html', name=name,countusers=countusers,
countorders=countorders)
@app.route("/AdminloginForm")
def AdminloginForm():
    if 'email' in session:
        return redirect(url_for('admin'))
    else:
        return render_template('adminLogin.html', error='')
def getrole(email):
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute("SELECT roleId FROM users WHERE email = '" + str(email) + "'")
        roleId = cur.fetchone()[0]
    return roleId
```

```
@app.route("/Adminlogin", methods=['POST', 'GET'])
def Adminlogin():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        if not is_valid(email, password):
            error = 'Invalid UserId / Password'
            flash("Невірний email або пароль")
            return render_template('adminLogin.html', error=error)
        if  getrole(email) == 2:
            session['email'] = email
            print("@@@@")
            return redirect(url_for('admin'))
        else:
            error = 'Invalid role'
            flash("Акаунт не має доступу")
            return render_template('adminLogin.html', error=error)
@app.route("/Adminregister", methods=['GET', 'POST'])
def Adminregister():
    if request.method == 'POST':
        # Parse form data
        password = request.form['password']
        email = str(request.form['email'])
        firstName = request.form['firstName']
        lastName = request.form['lastName']
        address1 = request.form['address1']
        address2 = request.form['address2']
        zipcode = request.form['zipcode']
        city = request.form['city']
        state = request.form['state']
        country = request.form['country']
        phone = request.form['phone']
        if checkUser(email):
            flash("Користувач з таким email вже існує")
            return render_template("Adminregister.html")
        else:
            with sqlite3.connect('database.db') as con:
                try:
                    cur = con.cursor()
                    cur.execute(
                        'INSERT INTO users (password, email, firstName, lastName, address1,
address2, zipcode, city, state, country, phone, roleId) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?)',
                        (
                        hashlib.md5(password.encode()).hexdigest(), email, firstName, lastName,
address1, address2, zipcode,
                        city, state, country, phone, 2))
                    con.commit()
                except:
                    con.rollback()
            con.close()
            return render_template("Adminlogin.html")
@app.route("/AdminregisterationForm")
def AdminregistrationForm():
    return render_template("Adminregister.html")
```

```
@app.route("/displayCategory")
def displayCategory():
    loggedIn, firstName, noOfItems = getLoginDetails()
    categoryId = request.args.get("categoryId")
    try:
        with sqlite3.connect('database.db') as conn:
            cur = conn.cursor()
            cur.execute(
                "SELECT products.productId, products.name, products.price, products.image,
categories.name FROM products, categories WHERE products.categoryId =
categories.categoryId AND categories.categoryId = " + categoryId)
            data = cur.fetchall()
        conn.close()
        categoryName = data[0][4]
        data = parse(data)
    except:
        return render_template('displayCategory.html', data=data, loggedIn=loggedIn,
firstName=firstName,
                               noOfItems=noOfItems, ab=session["ab"])
    return render_template('displayCategory.html', data=data, loggedIn=loggedIn,
firstName=firstName,
                           noOfItems=noOfItems, categoryName=categoryName, ab=session["ab"])
@app.route("/account/profile")
def profileHome():
    if 'email' not in session:
        return redirect(url_for('root'))
    loggedIn, firstName, noOfItems = getLoginDetails()
    return render_template("profileHome.html", loggedIn=loggedIn, firstName=firstName,
noOfItems=noOfItems, ab=session["ab"])
@app.route("/account/profile/edit")
def editProfile():
    if 'email' not in session:
        return redirect(url_for('root'))
    loggedIn, firstName, noOfItems = getLoginDetails()
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute(
            "SELECT userId, email, firstName, lastName, address1, address2, zipcode, city, state,
country, phone FROM users WHERE email = '" +
            session['email'] + "'")
        profileData = cur.fetchone()
    conn.close()
    return render_template("editProfile.html", profileData=profileData, loggedIn=loggedIn,
firstName=firstName,
                           noOfItems=noOfItems, ab=session["ab"])
@app.route("/account/profile/changePassword", methods=["GET", "POST"])
def changePassword():
    if 'email' not in session:
        return redirect(url_for('loginForm'))
    if request.method == "POST":
        oldPassword = request.form['oldpassword']
        oldPassword = hashlib.md5(oldPassword.encode()).hexdigest()
        newPassword = request.form['newpassword']
        newPassword = hashlib.md5(newPassword.encode()).hexdigest()
```

```
        with sqlite3.connect('database.db') as conn:
            cur = conn.cursor()
            cur.execute("SELECT userId, password FROM users WHERE email = '" + session['email']
+ "'")
            userId, password = cur.fetchone()
            if (password == oldPassword):
                try:
                    cur.execute("UPDATE users SET password = ? WHERE userId = ?", (newPassword,
userId))
                    conn.commit()
                    msg = "Changed successfully"
                except:
                    conn.rollback()
                    msg = "Failed"
                return render_template("ProfileHome.html", msg=msg)
            else:
                msg = "Wrong password"
                flash("Невірний пароль")
        conn.close()
        return render_template("changePassword.html", msg=msg)
    else:
        return render_template("changePassword.html")
@app.route("/updateProfile", methods=["GET", "POST"])
def updateProfile():
    if request.method == 'POST':
        email = request.form['email']
        firstName = request.form['firstName']
        lastName = request.form['lastName']
        address1 = request.form['address1']
        address2 = request.form['address2']
        zipcode = request.form['zipcode']
        city = request.form['city']
        state = request.form['state']
        country = request.form['country']
        phone = request.form['phone']
        with sqlite3.connect('database.db') as con:
            try:
                cur = con.cursor()
                cur.execute(
                    'UPDATE users SET firstName = ?, lastName = ?, address1 = ?, address2 = ?,
zipcode = ?, city = ?, state = ?, country = ?, phone = ? WHERE email = ?',
                    (firstName, lastName, address1, address2, zipcode, city, state, country, phone,
email))
                con.commit()
            except:
                con.rollback()
        con.close()
        return redirect(url_for('editProfile'))
@app.route("/loginForm")
def loginForm():
    if 'email' in session:
        return redirect(url_for('root'))
    else:
        return render_template('login.html', error='')
```

```
@app.route("/login", methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        if is_valid(email, password):
            session['email'] = email
            return redirect(url_for('root'))
        else:
            error = 'Invalid UserId / Password'
            flash("Невірний email або пароль")
            if abparameters():
                writestatic()
            return render_template('login.html', error=error)
@app.route("/productDescription")
def productDescription():
    loggedIn, firstName, noOfItems = getLoginDetails()
    productId = request.args.get('productId')
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute(
            'SELECT productId, name, price, description, image, stock, author    FROM products
WHERE productId = ' + productId)
        productData = cur.fetchone()
    conn.close()
    return  render_template("productDescription.html",  data=productData,  loggedIn=loggedIn,
firstName=firstName,
                            noOfItems=noOfItems)
@app.route("/addToCart")
def addToCart():
    if 'email' not in session:
        return redirect(url_for('loginForm'))
    else:
        productId = int(request.args.get('productId'))
        with sqlite3.connect('database.db') as conn:
            cur = conn.cursor()
            cur.execute("SELECT userId FROM users WHERE email = '" + session['email'] + "'")
            userId = cur.fetchone()[0]
            try:
                cur.execute("INSERT  INTO  kart  (userId,  productId)  VALUES  (?,  ?)",  (userId,
productId))
                conn.commit()
            except:
                conn.rollback()
        conn.close()
        return redirect(url_for('root'))
@app.route("/cart")
def cart():
    if 'email' not in session:
        return redirect(url_for('loginForm'))
    loggedIn, firstName, noOfItems = getLoginDetails()
    email = session['email']
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
```

```
        cur.execute("SELECT userId FROM users WHERE email = '" + email + "'")
        userId = cur.fetchone()[0]
        cur.execute(
            "SELECT products.productId, products.name, products.price, products.image,
products.author FROM products, kart  WHERE products.productId = kart.productId AND
kart.userId = " + str(
                userId))
        products = cur.fetchall()
    totalPrice = 0
    for row in products:
        totalPrice += row[2]
    return render_template("cart.html", products=products, totalPrice=totalPrice,
loggedIn=loggedIn,
                           firstName=firstName, noOfItems=noOfItems, ab=session["ab"])
@app.route("/removeFromCart")
def removeFromCart():
    if 'email' not in session:
        return redirect(url_for('loginForm'))
    email = session['email']
    productId = int(request.args.get('productId'))
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute("SELECT userId FROM users WHERE email = '" + email + "'")
        userId = cur.fetchone()[0]
        try:
            cur.execute("DELETE FROM kart WHERE userId = " + str(userId) + " AND productId =
" + str(productId))
            conn.commit()
        except:
            conn.rollback()
    conn.close()
    return redirect(url_for('root'))
@app.route("/logout")
def logout():
    session.pop('email', None)
    return redirect(url_for('root'))
def is_valid(email, password):
    con = sqlite3.connect('database.db')
    cur = con.cursor()
    cur.execute('SELECT email, password FROM users')
    data = cur.fetchall()
    for row in data:
        if row[0] == email and row[1] == hashlib.md5(password.encode()).hexdigest():
            return True
    return False
@app.route("/checkout", methods=['GET', 'POST'])
def payment():
    if 'email' not in session:
        return redirect(url_for('loginForm'))
    loggedIn, firstName, noOfItems = getLoginDetails()
    email = session['email']
    with sqlite3.connect('database.db') as conn:
        cur = conn.cursor()
        cur.execute("SELECT userId FROM users WHERE email = '" + email + "'")
```

```
        userId = cur.fetchone()[0]
        cur.execute(
            "SELECT products.productId, products.name, products.price, products.image FROM
products, kart WHERE products.productId = kart.productId AND kart.userId = " + str(
                userId))
        products = cur.fetchall()
    totalPrice = 0
    for row in products:
        totalPrice += row[2]
        cur.execute("INSERT INTO Orders (userId, productId) VALUES (?, ?)", (userId, row[0]))
    cur.execute("DELETE FROM kart WHERE userId = " + str(userId))
    conn.commit()
    session['payd']=True
    writestatic()
    return render_template("checkout.html", products=products, totalPrice=totalPrice,
loggedIn=loggedIn,
                           firstName=firstName, noOfItems=noOfItems)
def checkUser(email):
    with sqlite3.connect('database.db') as conn:
        cursor = conn.cursor()
        cursor.execute(
            "SELECT email  FROM users  WHERE users.email='" + email + "'")
        data = cursor.fetchall()
    if len(data)>0:
        return True
    return False
@app.route("/register", methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        # Parse form data
        password = request.form['password']
        email = str(request.form['email'])
        firstName = request.form['firstName']
        lastName = request.form['lastName']
        address1 = request.form['address1']
        address2 = request.form['address2']
        zipcode = request.form['zipcode']
        city = request.form['city']
        state = request.form['state']
        country = request.form['country']
        phone = request.form['phone']
        if checkUser(email):
            flash("Користувач з таким email вже існує")
            return render_template("register.html")
        else:
            with sqlite3.connect('database.db') as con:
                try:
                    cur = con.cursor()
                    cur.execute(
                        'INSERT INTO users (password, email, firstName, lastName, address1,
address2, zipcode, city, state, country, phone, roleId) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,?)',
                        (
                        hashlib.md5(password.encode()).hexdigest(), email, firstName, lastName,
address1, address2, zipcode,
```

```
                    city, state, country, phone, 1))
                con.commit()
                msg = "Registered Successfully"
            except:
                con.rollback()
                msg = "Error occured"
        con.close()
        return render_template("login.html", error=msg)
@app.route("/registerationForm")
def registrationForm():
    return render_template("register.html")
def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1] in ALLOWED_EXTENSIONS
def parse(data):
    ans = []
    i = 0
    while i < len(data):
        curr = []
        for j in range(7):
            if i >= len(data):
                break
            curr.append(data[i])
            i += 1
        ans.append(curr)
    return ans
@app.route('/handle_data', methods=['POST'])
def search():
    data=[]
    loggedIn, firstName, noOfItems = getLoginDetails()
    found = str(request.form['foundName'])
    if len(found)<2:
        return render_template('found.html', data=data, loggedIn=loggedIn,
firstName=firstName, noOfItems=noOfItems)
    with sqlite3.connect('database.db') as conn:
        cursor = conn.cursor()
        cursor.execute( "SELECT  productId, products.name, products.price, products.image FROM
products  WHERE products.name LIKE '%" + found +"%'")
        data = cursor.fetchall()
    data = parse(data)
    conn.close()
    return render_template('found.html', data=data, loggedIn=loggedIn, firstName=firstName,
noOfItems=noOfItems, ab=session["ab"])
@app.route("/admin/categories")
def Categories():
    name = getLoginDetails()[1]
    if 'email' not in session or getrole(session["email"]) != 2:
        return render_template('Adminlogin.html', error='')
    else:
        with sqlite3.connect('database.db') as conn:
            cur = conn.cursor()
            cur.execute('SELECT categoryId, name FROM categories')
            data = cur.fetchall()
        conn.close()
```

```
        return render_template('categories.html', data=data, name=name)
@app.route("/removeCategory")
def removeCategory():
    categoryId = request.args.get('categoryId')
    with sqlite3.connect('database.db') as conn:
        try:
            cur = conn.cursor()
            cur.execute('DELETE FROM categories WHERE categoryID = ' + categoryId)
            conn.commit()
        except:
            conn.rollback()
    conn.close()
    return redirect(url_for('Categories'))
@app.route("/addCategory", methods=['GET', 'POST'])
def addCategory():
        name = str(request.form.get('name'))
        with sqlite3.connect('database.db') as conn:
            try:
                cur = conn.cursor()
                cur.execute("INSERT INTO categories (name) VALUES (?)", ( name, ))
                conn.commit()
            except:
                conn.rollback()
        conn.close()
        return redirect(url_for('Categories'))
@app.route("/admin/statistic", methods=['GET', 'POST'])
def statistic():
    name = getLoginDetails()[1]
    data = csvwrite('expdata.csv')
    data1 = csvwrite('userstatistic.csv')
    size = int(float(data[0][2]))
    frame = pd.DataFrame(data1, columns=['sessionId', 'userId', 'ab','payd','date'])
    frame=frame.drop_duplicates(keep='first')
    frame=frame.drop_duplicates(subset=['sessionId'],keep='last')
    if len(frame)<=1:
        plot = figure(plot_height=200, sizing_mode='scale_width', x_axis_type="datetime",
                      toolbar_location="below", name=name)
        script, div = components(plot)
        js_resources = INLINE.render_js()
        css_resources = INLINE.render_css()
        return render_template('statistic.html', size=size, script=script, div=div,
js_resources=js_resources,
                               css_resources=css_resources, convA=0, convB=0,
                               a=0, b=0, s1=1, s2=1, sall=1, conv=0, start=data[0][0], end=data[0][1],
name=name)
    frame = [pd.DataFrame(y) for x, y in frame.groupby('date', as_index=False)]
    xa=[]
    xb=[]
    y=[]
    a=0
    s1=0
    b=0
    s2=0
    try:
```

```
        for i in frame:
                print(i)
                a += len(i[(i['ab']=='True') & (i['payd']=='True')])
                s1 += len(i[(i['ab']=='True')])
                b += len(i[(i['ab']=='False') & (i['payd']=='True')])
                s2 += len(i[(i['ab'] == 'False')])
                print(s1)
                print(s2)
                xa.append(a/s1)
                xb.append(b/s2)
                y.append(max(pd.to_datetime(i['date'])))
        y=list(y)
        tools = [
           PanTool(),
           BoxZoomTool(),
           WheelZoomTool(),
           HoverTool(tooltips=[('conversion', '$y{%0}')] )
           ]
        plot = figure(plot_height=200, sizing_mode='scale_width', x_axis_type="datetime",
tools=tools, toolbar_location="below")
        plot.line(y, xa, line_width=2, legend_label = "variant a")
        plot.line(y, xb, line_width=2, line_color="red", legend_label="variant b")
        plot.yaxis[0].formatter = NumeralTickFormatter(format='0.0%')
        script, div = components(plot)
        js_resources = INLINE.render_js()
        css_resources = INLINE.render_css()
    except Exception as e:
        print(e)
        plot = figure(plot_height=200, sizing_mode='scale_width', x_axis_type="datetime",
                    toolbar_location="below")
        plot.yaxis[0].formatter = NumeralTickFormatter(format='0.0%')
        script, div = components(plot)
        js_resources = INLINE.render_js()
        css_resources = INLINE.render_css()
        return render_template('statistic.html', size=size, script=script, div=div,
js_resources=js_resources,
                               css_resources=css_resources, convA=0, convB=0,
                               a=0, b=0, s1=0, s2=0, sall=0, conv=0, start=data[0][0], end=data[0][1])
    return render_template('statistic.html', size=size, script=script,div=div,
js_resources=js_resources,
        css_resources=css_resources, convA=round(100*xa[-1], 2), convB=100*round(xb[-1], 2),
a=a, b=b, s1=s1, s2=s2, sall=s1+s2,conv=a+b,start=data[0][0],end=data[0][1], name=name)
@app.route("/calcsize", methods=['GET', 'POST'])
def sizecalc():
    alpha = int(request.form['significance'])/100  # Type 1 error
    beta = (100-int(request.form['power']))/100  # Type 2 error
    Qalpha = norm.ppf(1 - alpha / 2)  # quantile value
    Qbeta = norm.ppf(beta)  # quantile value
    Pc = int(request.form['Pc'])/100  # Base CTR for Control
    Dmin = int(request.form['Dmin'])/100  # Min value of practical significance
    Pt = Pc + Dmin  # CTR for Treatment (Null hypothesis for Beta)
    sd1 = sqrt(2 * Pc * (1 - Pc))
    sd2 = sqrt((Pc) * (1 - Pc) + Pt * (1 - Pt))
    N = ceil(((Qalpha * sd1 - Qbeta * sd2) / (Dmin)) ** 2)
```

```
    os.remove('userstatistic.csv')
    with open('expdata.csv', 'w', newline='') as csvfile:
        exp = ['start', 'end', "size"]
        writer = DictWriter(csvfile, fieldnames=exp)
        writer.writeheader()
        writer.writerow({'start': request.form['start']+' 00', 'end': request.form['end']+' 23',
'size':N})
    return redirect(url_for('statistic'))
def abparameters():
    data = csvwrite('expdata.csv')
    end = datetime.strptime(data[0][1], '%Y-%m-%d %H' )
    current_datetime = datetime.now()
    if current_datetime>end:
        return False
    return True
def csvwrite(name):
    try:
        count=0
        data = []
        with open(name, newline='') as File:
            reader = csv.reader(File)
            for row in reader:
                if count !=0:
                    data += [row]
                count += 1
    except:
        return []
    return data
def writestatic():
    if 'payd' not in session:
        session['payd'] = False
    data=[]
    session['date'] = datetime.now().strftime('%Y-%m-%d %H')
    lastsession=csvwrite("userstatistic.csv")
    if lastsession != []:
        if 'sesId' not in session:
            session['sesId']=int(lastsession[-1][0])+1
    else:
        session['sesId'] = 0
    data.append(session['sesId'])
    data.append(session['userId'])
    data.append(session['ab'])
    data.append(session['payd'])
    data.append(session['date'])
    with open('userstatistic.csv', 'a', newline='') as f_object:
        writer_object = writer(f_object)
        writer_object.writerow(data)
if __name__ == '__main__':
    app.run(debug=True)
```

Вміст файлу database.py:

```
import sqlite3
#Open database
conn = sqlite3.connect('database.db')
#Create table
```

```
conn.execute('''CREATE TABLE users
                (userId INTEGER PRIMARY KEY,
                password TEXT,
                email TEXT,
                firstName TEXT,
                lastName TEXT,
                address1 TEXT,
                address2 TEXT,
                zipcode TEXT,
                city TEXT,
                state TEXT,
                country TEXT,
                phone TEXT
                )''')
conn.execute('''CREATE TABLE products
                (productId INTEGER PRIMARY KEY,
                name TEXT,
                price REAL,
                description TEXT,
                image TEXT,
                stock INTEGER,
                categoryId INTEGER,
        author TEXT,
                FOREIGN KEY(categoryId) REFERENCES categories(categoryId)
                )''')
conn.execute('''CREATE TABLE kart
                (userId INTEGER,
                productId INTEGER,
                FOREIGN KEY(userId) REFERENCES users(userId),
                FOREIGN KEY(productId) REFERENCES products(productId)
                )''')
conn.execute('''CREATE TABLE categories
                (categoryId INTEGER PRIMARY KEY,
                name TEXT
                )''')
conn.execute('''CREATE TABLE orders
                (userId INTEGER,
                productId INTEGER,
                FOREIGN KEY(userId) REFERENCES users(userId),
                FOREIGN KEY(productId) REFERENCES products(productId)
                )''')
conn.close()
```

Вміст файлу changePassword.js:

```
function validate() {
    var pass = document.getElementById("newpassword").value;
    var cpass = document.getElementById("cpassword").value;
    if (pass == cpass) {
        return true;
    } else {
        alert("Паролі не співпадають!");
        return false;
    }
}
```

Вміст файлу validateForm.js:

```
function validate() {
    var pass = document.getElementById("password").value;
    var cpass = document.getElementById("cpassword").value;
    if (pass == cpass) {
        return true;
    } else {
        alert("Паролі не співпадають");
        return false;
    }
}
```

Вміст файлу add.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Admin</title>
<link rel="stylesheet"  href={{url_for('static', filename="css/add.css")}} />
</head>
<body>
<h2>Додавання товару</h2>
<form action="/addItem" method="POST" enctype="multipart/form-data">
        Назва: <input type="text" name="name" class="add"><br>
        Автор: <input type="text" name="author" class="add"><br>
        Ціна: <input type="text" name="price" class="add"><br>
        Опис: <textarea name="description" rows=3 cols="40" class="add"></textarea><br>
        Обкладинка: <input type="file" name="image" class="add"><br>
        Кількість: <input type="text" name="stock" class="add"><br>
        Жанр: <select name="category" class="add">
                {% for row in categories %}
                        <option value="{{row[0]}}">{{row[1]}}</option>
                {% endfor %}
        </select><br>
        <input type="submit" class="remove">
</form>
</body>
</html>
```

Вміст файлу admin.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Панель адмінстратора</title>
    <meta name="robots" content="noindex" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-
scale=1.5" />
    <link rel="stylesheet"  href={{url_for('static', filename="css/admin.css")}} />
  </head>
  <body>
    <div  id="pgside">
                <div class="admin">
                <a href="/admin" >
       <i class="txt">Адміністратор {{name}}</i></a></div>

      <a href="/admin/items" class="current">
        <i class="txt">Товари</i>
      </a>
```

```
      <a href="/admin/categories">
        <i class="txt">Категорії</i>
      </a>
      <a href="/admin/statistic">
        <i class="txt">Статистика</i>
      </a>
    </div>
    <div id="pgmain">
          <div class="plate">
  <p class="script"><span>Кількість користувачів, що зареєстровані в системі: {{countusers}}</span></p>
  <p class="script"><span>Загальна кількість покупок у магазині: {{countorders}}</span></p>
</div>
    </div>
  </body>
</html>
```

Вміст файлу AdminLogin.html:

```
<html>
<head>
<title> Вхід </title>
<link rel="stylesheet" href={{ url_for('static', filename='css/login.css') }} />
</head>
<body>
<div class="login-page">
  <div class="form">
    {% for message in get_flashed_messages() %}
    <div>{{ message }}</div>
          {% endfor %}
    <form class="login-form" action="/Adminlogin" method="POST">
      <input type="text" placeholder="email" name="email"/>
      <input type="password" placeholder="password" name="password"/>
      <button>Вхід</button>
      <p class="message">Ще не зареєстрований? <a href="/AdminregisterationForm">Створити аккаунт </a></p>
    </form>
  </div>
</div>
</body>
</html>
```

Вміст файлу Adminregister.html:

```
<html>
<head>
<title>Реєстрація</title>
<script type="text/javascript" src="{{ url_for('static', filename = 'js/validateForm.js') }}">
</script>
<link rel="stylesheet" href={{ url_for('static', filename='css/login.css') }} />
</head>
<body>
<div class="login-page">
  <div class="form">
  {% for message in get_flashed_messages() %}
    <div>{{ message }}</div>
          {% endfor %}
```

```
   <form class="register-form" action="/Adminregister" method="POST" onsubmit="return validate()">
     <input type="password" placeholder="пароль" id="password" name="password"/>
          <input type="password" placeholder=" підвердіть пароль" id="cpassword" name="password"/>
     <input type="text" placeholder="email" name="email"/>
          <input type="text" name="firstName" placeholder="Ім'я">
          <input type="text" name="lastName" placeholder="Прізвище">
          <input type="text" name="address1" placeholder="Поле адреси 2">
          <input type="text" name="address2" placeholder="Поле адреси 1">
     <input type="text" name="zipcode" placeholder="Індекс">
     <input type="text" name="city" placeholder="Місто">
     <input type="text" name="state" placeholder="Область">
     <input type="text" name="country" placeholder="Країна">
     <input type="text" name="phone" placeholder="Телефон">
     <button>Створити</button>
     <p class="message">Вже зареєстровані? <a href="/AdminloginForm">Вхід</a></p>
          </div>
</body>
</html>
```

Вміст файлу cart.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Товарів у кошику: {{noOfItems}}</title>
<link rel="stylesheet" href={{url_for('static', filename='css/cart.css')}} />
<link rel="stylesheet" href={{url_for('static', filename='css/topStyle.css') }} />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
        <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
                <div id="signInButton">
                        <a class="link" href="/loginForm">Вхід</a>
                </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Твій профіль</a>
                        <hr>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
```

```
            {% if ab %}
        <div id="kart">
                {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}
id="cartIcon" />
                Кошик {{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                Кошик {{noOfItems}}
            {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" ">                    {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div id="cartItems">
        <div id="header">
        <span id="product">Товарів у кошику: {{noOfItems}} </span><span
id="Price">Ціна</span>
        </div>
        <div id="tableItems">
            {% for row in products %}
            <div class="Item">
                <hr id="seperator">
                <div id="itemImage">
                <a href="/productDescription?productId={{row[0]}}">
                    <img src={{url_for('static', filename='uploads/'+row[3])}}
id="image"/>
                </a>
                </div>
                <div id="itemName">
                    <span id="itemNameTag"><nobr>{{row[1]}}</nobr></span>
                    <span id="itemNameTag">{{row[4]}}</span>
                    <div class="remove" ><a
href="/removeFromCart?productId={{row[0]}}">Видалити</a></div>
                </div>
                        <div id="itemPrice">
                    ₴{{row[2]}}
```

```
                    </div>
                </div>
                {% endfor %}
                <div id="total">
                <hr id="seperator">
<div class="avd_div"><a href="/checkout" class="but_sad_g_3 green_sad">До сплати: ₴{{totalPrice}}</a></div>
                </div>
        </div>
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу categories.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Панель адмінстратора</title>
    <meta name="robots" content="noindex" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.5" />
    <link rel="stylesheet"  href={{url_for('static', filename="css/admin.css")}} />
         <link rel="stylesheet"  href={{url_for('static', filename="css/items.css")}} />
  </head>
  <body>
    <div  id="pgside">
                <div class="admin">
                <a href="/admin" >
       <i class="txt">Адміністратор {{name}}</i></a></div>
      <a href="/admin/items" class="current">
       <i class="txt">Товари</i>
      </a>
      <a href="/admin/categories">
       <i class="txt">Категорії</i>
      </a>
      <a href="/admin/statistic">
       <i class="txt">Статистика</i>
      </a>
    </div>
    <div id="pgmain">
      <table class="table">
<tr>
<td>Назва</td>
</tr>
                                {% for row in data %}
                                <tr id="productName">
                            <td id="BookName">
                                {{row[1]}}
                            </td>
                            <td id="BookName">
                                <a          href="/removeCategory?categoryId={{row[0]}}" class="addToCart"><img src={{url_for('static', filename='images/remove.png')}}/></a>
                            </td>
```

```
                    </tr>
                    {% endfor %}
                    <tr>
                     <form  action="{{ url_for('addCategory') }}" method="post" >
                    <td><input type="text" name="name" class="add"
method="POST"></td>
                    <td> <input type="submit" class="remove" value="Додати"></td>
                    </form>
                    </tr>
</table>
   </div>
  </body>
</html>
```

Вміст файлу changePassword.html:

```
<html>
<head>
<title>Зміна паролю</title>
<script src={{ url_for('static', filename='js/changePassword.js') }}></script>
<link rel="stylesheet" href={{ url_for('static', filename='css/login.css') }} />
</head>
<body>
<div class="login-page">
  <div class="form">
     {% for message in get_flashed_messages() %}
   <div>{{ message }}</div>
        {% endfor %}
<form action={{ url_for('changePassword') }} method="POST" onsubmit="return validate()">
        <input type="password" name="oldpassword" placeholder="Старий пароль">
        <input type="password" name="newpassword" id="newpassword" placeholder="Новий
пароль">
        <input type="password" name="cpassword" id="cpassword" placeholder="Підтвердіть
пароль">
        <button>Зберегти</button>
</form>
<p class="message"><a href="{{ url_for('profileHome') }}">Повернутися до
профіля?</a></p>
</div>
</div>
</body>
</html>
```

Вміст файлу checkout.html:

```
<!DOCTYPE HTML>
<html>
<head>
<link rel="stylesheet" href={{url_for('static', filename='css/cart.css')}} />
        <title>Чек</title>
</head>
<body>
        <div id="cartItems">
                <div id="header">
                <h2 id="product">Товари, що ви купили</h2>
        </div>
        <div id="tableItems">
```

```
                {% for row in products %}
                <div>
                        <hr id="seperator">
                        <div id="checkoutName">
                                <span id="itemNameTag">{{row[1]}}</span><br>
                        </div>
                        <div id="checkoutPrice">
                                ${{row[2]}}
                        </div>
                </div>
                {% endfor %}
                <hr id="seperator">
                <div id="total">
                        <span id="subtotal">Загальна сума</span> : ₴{{totalPrice}}
                        <span if="deliverydate"></span>
                </div>
                <div class="remove"><a href="/">Головна сторінка</div>
        </div>
</div>
</body>
</html>
```

Вміст файлу displayCategory:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Категорія: {{categoryName}}</title>
<link rel="stylesheet" href={{ url_for('static', filename='css/home.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css') }} />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Твій профіль</a>
                        <hr>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
```

```
        {% endif %}
        {% if ab %}
        <div id="kart">
                {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}
id="cartIcon" />
                Кошик {{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                Кошик {{noOfItems}}
            {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" ">                {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div class="displayCategory">
        <h2>Усі книги жанру {{categoryName}}:</h2>
        {% for itemData in data %}
        <table>
            <tr id="productName">
                {% for row in itemData %}
                <td id="BookName">
                    {{row[1]}}
                </td>
                {% endfor %}
            </tr>
            <tr id="productImage">
                {% for row in itemData %}
                <td id="BookImage">
                    <a href="/productDescription?productId={{row[0]}}">
                        <img src={{ url_for('static', filename='uploads/' + row[3])
}} id="itemImage" />
                    </a>
                </td>
                {% endfor %}
            </tr>
            <tr id="productPrice">
```

```
                {% for row in itemData %}
            <td id="BookPrice">
                                {{row[2]}}₴
                                <a href="/addToCart?productId={{row[0]}}"
class="addToCart"><img src={{url_for('static', filename='images/shoppingCart.png')}}
id="cartIcon" /></a>
            </td>
                {% endfor %}
            </tr>
        </table>
        {% endfor %}
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу editProfile.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Зміна даних профіля </title>
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/edit.css') }} />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <div>
        <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли
почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        </div>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Твій профіль</a>
                        <hr>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
        {% if ab %}
        <div id="kart">
```

```
                {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}
id="cartIcon" />
                Кошик {{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                Кошик {{noOfItems}}
            {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" ">                {% if noOfItems %}
            <a class="link" href="/cart">
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            </a>
            {% else %}
            <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                <br>{{noOfItems}}
            {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div class="display">
<div class="login-page">
<div class="form">
<div class="header-h1"><h1 >Зміна даних профіля</h1></div>
        <form action="/updateProfile" method="POST">
            <label>email</label>
            <input type="email" name="email" value={{profileData[1]}}
readonly="readonly" >
            <label>Ім'я</label>
            <input type="text" name="firstName" value={{profileData[2]}}>
            <label>Прізвище</label>
            <input type="text" name="lastName" value={{profileData[3]}}>
            <label>Поле адреси 1</label>
            <input type="text" name="address1" value={{profileData[4]}}>
            <label>Поле адреси 2</label>
            <input type="text" name="address2" value={{profileData[5]}}>
            <label>Індекс</label>
            <input type="text" name="zipcode" value={{profileData[6]}}>
            <label>Місто</label>
            <input type="text" name="city" value={{profileData[7]}}>
            <label>Область</label>
            <input type="text" name="state" value={{profileData[8]}}>
            <label>Країна</label>
            <input type="text" name="country" value={{profileData[9]}}>
            <label>Телефон</label>
```

```
                <input type="text" name="phone" value={{profileData[10]}}>
                <button>Зберегти</button>
        </form>
</div>
</div>
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу found.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Знайдені книги</title>
<link rel="stylesheet" href={{ url_for('static', filename='css/home.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css') }} />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Твій профіль</a>
                        <hr>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
        {% if ab %}
        <div id="kart">
                        {% if noOfItems %}
                <a class="link" href="/cart">
                        <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon" />
                        Кошик {{noOfItems}}
                </a>
                {% else %}
```

```
                <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                        Кошик {{noOfItems}}
                {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" ">                        {% if noOfItems %}
                <a class="link" href="/cart">
                        <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                        <br>{{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                        <br>{{noOfItems}}
                {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>

<div class="displayCategory">
        <h2>Усі знайдені книги:</h2>
        {% for itemData in data %}
        <table>
                <tr id="productName">
                        {% for row in itemData %}
                        <td id="BookName">
                                {{row[1]}}
                        </td>
                        {% endfor %}
                </tr>
                <tr id="productImage">
                        {% for row in itemData %}
                        <td id="BookImage">
                                <a href="/productDescription?productId={{row[0]}}">
                                        <img src={{ url_for('static', filename='uploads/' + row[3])
}} id="itemImage" />
                                </a>
                        </td>
                        {% endfor %}
                </tr>
                <tr id="productPrice">
                        {% for row in itemData %}
                <td id="BookPrice">
                                        {{row[2]}}₴
                                        <a                href="/addToCart?productId={{row[0]}}"
class="addToCart"><img      src={{url_for('static',      filename='images/shoppingCart.png')}}
id="cartIcon" /></a>
                </td>
                        {% endfor %}
                </tr>
```

```
        </table>
        {% endfor %}
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу home.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Інтернет-магазин "Буквик"</title>
<link rel="stylesheet" href={{ url_for('static', filename='css/home.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css') }} />
<link rel="shortcut icon" href="favicon.ico" type="image/x-icon">
<link href="/favicon.ico" rel="icon" type="image/x-icon" />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
         <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Твій профіль</a>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
        {% if ab %}
        <div id="kart">
                        {% if noOfItems %}
                <a class="link" href="/cart">
                        <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon" />
                        Кошик {{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon" />
                        Кошик {{noOfItems}}
```

```
                {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" ">                        {% if noOfItems %}
                <a class="link" href="/cart">
                        <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                        <br>{{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                        <br>{{noOfItems}}
                {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div class="display">
        <div class="displayCategory">
                <h2>Жанри: </h2>
                <ul>
                        {% for row in categoryData %}
                        <li><a
href="/displayCategory?categoryId={{row[0]}}">{{row[1]}}</a></li>
                        {% endfor %}
                </ul>
        </div>
        <div class="Items" >
                <h2>Каталог товарів</h2>
                {% for data in itemData%}
                <table >
                        <tr id="productName">
                                {% for row in data %}
                                <td id="BookName">
                                        {{row[1]}}
                                </td>
                                {% endfor %}
                        </tr>
                        <tr id="productImage">
                                {% for row in data %}
                                <td id="BookImage">
                                        <a href="/productDescription?productId={{row[0]}}">
                                                <img  src={{  url_for('static', filename='uploads/' +
row[4]) }} id="itemImage" />
                                        </a>
                                </td>
                                {% endfor %}
                        </tr>
                        <tr id="productPrice">
                                {% for row in data %}
                                <td id="BookPrice">
                                        {{row[2]}}₴
```

```
                                    <a                  href="/addToCart?productId={{row[0]}}"
class="addToCart"><img      src={{url_for('static',      filename='images/shoppingCart.png')}}
id="cartIcon" /></a>
                                </td>
                                {% endfor %}
                            </tr>
                        </table>
                        {% endfor %}
                </div>
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу items.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Панель адмінстратора</title>
    <meta name="robots" content="noindex" />
    <meta    name="viewport"    content="width=device-width,    initial-scale=1.0,    maximum-
scale=1.5" />
    <link rel="stylesheet"  href={{url_for('static', filename="css/admin.css")}} />
         <link rel="stylesheet"  href={{url_for('static', filename="css/items.css")}} />
  </head>
  <body>
    <div  id="pgside">
                <div class="admin">
                <a href="/admin" >
       <i class="txt">Адміністратор {{name}}</i></a></div>

      <a href="/admin/items" class="current">
        <i class="txt">Товари</i>
      </a>
      <a href="/admin/categories">
        <i class="txt">Категорії</i>
      </a>
      <a href="/admin/statistic">
        <i class="txt">Статистика</i>
      </a>
    </div>
    <div id="pgmain">
      <table class="table">
<tr>
<td>Назва</td>
<td>Автор</td>
<td>Ціна</td>
<td>Кількість</td>
</tr>
                                {% for row in data %}
                                <tr id="productName">
                            <td id="BookName">
                                {{row[1]}}
                            </td>
```

```
                                <td id="BookName">
                                        {{row[2]}}
                                </td>
                                <td id="BookName">
                                        {{row[3]}}
                                </td>
                                <td id="BookName">
                                        {{row[4]}}
                                </td>
                                <td id="BookName">
                                        <a href="/removeItem?productId={{row[0]}}"
class="addToCart"><img src={{url_for('static', filename='images/remove.png')}}/></a>
                                </td>
                        </tr>
                        {% endfor %}
</table>
<div class="remove" ><a href="/add">Додати <img src={{url_for('static',
filename='images/add.png')}}></a></div>
    </div>
  </body>
</html>
```

Вміст файлу login.html:

```
<html>
<head>
<title> Вхід </title>
<link rel="stylesheet" href={{ url_for('static', filename='css/login.css') }} />
</head>
<body>
<div class="login-page">
  <div class="form">
    {% for message in get_flashed_messages() %}
    <div>{{ message }}</div>
        {% endfor %}
    <form class="login-form" action="/login" method="POST">
      <input type="text" placeholder="email" name="email"/>
      <input type="password" placeholder="password" name="password"/>
      <button>Вхід</button>
      <p class="message">Ще не зареєстрований? <a href="/registerationForm">Створити
аккаунт </a></p>
    </form>
  </div>
</div>
</body>
</html>
```

Вміст файлу ProductDescription.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Усе про книжку "{{data[1]}}"</title>
<link rel="stylesheet" href={{url_for('static', filename='css/productDescription.css')}} />
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css')}} />
</head>
<body>
```

```
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <form action="{{ url_for('search') }}" method="post">
   <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли
почитати?">
   <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
                        <a href="/account/profile">Мій профіль</a>
                        <hr>
                        <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
        <div id="kart">
                        {% if noOfItems %}
                <a class="link" href="/cart">
                        <img      src={{url_for('static',      filename='images/shoppingCart.png')}}
id="cartIcon" />
                        Кошик {{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                        Кошик {{noOfItems}}
                {% endif %}
        </div>
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div id="display">
                <h1>{{data[1]}}</h1>
        <div id="productImage">
                <img          src={{url_for('static',          filename='uploads/'+data[4])          }}
id="productImage"/>
        </div>
        <div id="productDescription">
                <h2>Опис</h2>
                <table id="descriptionTable">
                        <tr>
```

```
                        <td>Автор</td>
                        <td>{{data[6]}}</td>
                    </tr>
                    <tr>
                        <td>Ціна</td>
                        <td>{{data[2]}}₴</td>
                    </tr>
                    <tr>
                        <td>Залишилось книг</td>
                        <td>{{data[5]}}</td>
                    </tr>
                </table>
        </div>
    <h2>Усе про книжку</h2>
        <div id="description">
        <p>{{data[3]}}</p></div>
         <div class="Cart">
                <a                href="/addToCart?productId={{request.args.get('productId')}}"
class="addToCart">Додати у кошик</a>
        </div>
</div>
 </div>
    </div>
</body>
</html>
```

Вміст файлу ProfileHome.html:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Твій профіль</title>
<link rel="stylesheet" href={{ url_for('static', filename='css/profileHome.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/topStyle.css') }} />
<link rel="stylesheet" href={{ url_for('static', filename='css/home.css') }} />
</head>
<body>
<div class="page-container">
<div class="page-container-inner">
<div id="title">
        <a href="/">
                <img id="logo" src= {{ url_for('static', filename='images/logo.png') }} />
        </a>
        <form action="{{ url_for('search') }}" method="post">
    <input type="search" name="foundName" id="searchBox" placeholder="Що ви хотіли
почитати?">
    <input type="submit" id="searchButton" value="Пошук">
</form>
        {% if not loggedIn %}
        <div id="signInButton">
                <a class="link" href="/loginForm">Вхід</a>
        </div>
        {% else %}
        <div class="dropdown">
                <button class="dropbtn">Привіт, <br>{{firstName}}</button>
                <div class="dropdown-content">
```

```
                    <a href="/account/profile">Твій профіль</a>
                    <hr>
                    <a href="/logout">Вихід</a>
                </div>
        </div>
        {% endif %}
        {% if ab %}
        <div id="kart">
                    {% if noOfItems %}
                <a class="link" href="/cart">
                    <img src={{url_for('static', filename='images/shoppingCart.png')}}
id="cartIcon" />
                    Кошик {{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}} id="cartIcon"
/>
                    Кошик {{noOfItems}}
                {% endif %}
        </div>
        {% else %}
        <div class="fixedbut" onclick="location.href='/o-nas/'">                    {%              if
noOfItems %}
                <a class="link" href="/cart">
                    <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                    <br>{{noOfItems}}
                </a>
                {% else %}
                <img src={{url_for('static', filename='images/shoppingCart.png')}}  />
                    <br>{{noOfItems}}
                {% endif %}</div>
        {% endif %}
        <div class="number">
        <p>Безкоштовно по Україні</p>
        <p style="font-weight: bold">0-800-335-425<p/>
        7 днів на тиждень з 9 до 18</p>
        </div>
</div>
<div class="display">
        <h1>Персональні дані</h1>
        <div          class="remove"><a          href="/account/profile/edit">Змінити          данні
профілю</a></div>
        <div          class="remove"><a          href="/account/profile/changePassword">Змінити
пароль</a></div>
</div>
 </div>
   </div>
</body>
</html>
```

Вміст файлу register.html:

```
<html>
<head>
<title>Реєстрація</title>
<script type="text/javascript" src="{{ url_for('static', filename = 'js/validateForm.js') }}">
```

```
</script>
<link rel="stylesheet" href={{ url_for('static', filename='css/login.css') }} />
</head>
<body>
<div class="login-page">
  <div class="form">
  {% for message in get_flashed_messages() %}
    <div>{{ message }}</div>
        {% endfor %}
   <form class="register-form" action="/register" method="POST" onsubmit="return validate()">
     <input type="password" placeholder="пароль"  id="password" name="password"/>
          <input type="password" placeholder=" підвердіть пароль" id="cpassword" name="password"/>
     <input type="text" placeholder="email" name="email"/>
          <input type="text" name="firstName" placeholder="Ім'я">
          <input type="text" name="lastName" placeholder="Прізвище">
          <input type="text" name="address1" placeholder="Поле адреси 2">
          <input type="text" name="address2" placeholder="Поле адреси 1">
     <input type="text" name="zipcode" placeholder="Індекс">
     <input type="text" name="city" placeholder="Місто">
     <input type="text" name="state" placeholder="Область">
     <input type="text" name="country" placeholder="Країна">
     <input type="text" name="phone" placeholder="Телефон">
     <button>Створити</button>
     <p class="message">Вже зареєстровані? <a href="/loginForm">Вхід</a></p>
          </div>
</body>
</html>
```

Вміст файлу statistic.html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Панель адмінстратора</title>
    <meta name="robots" content="noindex" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.5" />
    <link rel="stylesheet"  href={{url_for('static', filename="css/admin.css")}} />
         <link rel="stylesheet"  href={{url_for('static', filename="css/statistic.css")}} />
         {{ script|indent(4)|safe }}
         {{ js_resources|indent(4)|safe }}
   {{ css_resources|indent(4)|safe }}
  </head>
  <body>
     <div  id="pgside">
                <div class="admin">
                <a href="/admin" >
       <i class="txt">Адміністратор {{name}}</i></a></div>
     <a href="/admin/items" class="current">
       <i class="txt">Товари</i>
     </a>
     <a href="/admin/categories">
       <i class="txt">Категорії</i>
     </a>
```

```
      <a href="/admin/statistic">
        <i class="txt">Статистика</i>
      </a>
    </div>
  <div class="staticvalues">
  <div>
  <h2>ПРОВЕДЕННЯ ТЕСТУВАННЯ</h2>
   <form class="register-form" action="{{ url_for('sizecalc') }}" method="POST" >
   <h4>Потужність 1−β</h4>
  <div class="range-slider-power">
    <label class="slider" for="slider-input">
    <input name="power" class="slider_input-power" id="slider-input-power" type="range"
min="60" max="95" value="80">
    <span class="slider_label-power">
      Значення: <output class="slider_output-power" for="slider-input"></output>%
    </span>
  </label>
         </div>
         <h4>Рівень значущості α</h4>
<div class="range-slider-significance">
    <label class="slider" for="slider-input">
    <input name="significance" class="slider_input-significance" id="slider-input-significance"
type="range" min="1" max="10" value="5">
    <span class="slider_label-significance">
      Значення: <output class="slider_output-significance" for="slider-input"></output>%
    </span>
  </label>
</div>
<span>Поточний коефіцієнт конверсії</span><input type="text" name="Pc"
class="add"><br>
<span>Необхідний коефіцієнт конверсії<span><input type="text" name="Dmin"
class="add"><br>
<!-- <h4>Необхідний розмір вибірки для кожної варіації: {{size}}</h4> -->
<span>Введіть дату початку експерименту</span><input type="date" name="start"
class="date"><br>
<span>Введіть дату кінця експерименту</span><input type="date" name="end"
class="date">
<div>
<input type="submit" class="remove" value="Розпочати експеримент">
</div>
</form>
</div>
<script type="text/javascript">
const slider = document.querySelector('.range-slider-power');
const input = document.querySelector('.slider_input-power');
const output = document.querySelector('.slider_output-power');
const slider1 = document.querySelector('.range-slider-significance');
const input1 = document.querySelector('.slider_input-significance');
const output1 = document.querySelector('.slider_output-significance');
document.addEventListener('DOMContentLoaded', updateValue);
document.addEventListener('DOMContentLoaded', updateValue1);
input.addEventListener('input', updateValue);
input1.addEventListener('input', updateValue1);
function updateValue() {
```

```
  slider.style.setProperty('--value', input.value);
  output.value = input.value;
}
function updateValue1() {
  slider1.style.setProperty('--value', input1.value);
  output1.value = input1.value;
}
</script>
  </div>
  <div>
  <h4>Загальні дані експерименту</h4>
    <table class="table">
<tr>
<td>Дати експерименту</td>
<td>Всього конверсій</td>
<td>Кількість відвідувачів</td>
</tr>
<tr>
<td>3 {{start}} до {{end}}</td>
<td>{{conv}}</td>
<td>{{sall}}</td>
</tr>
</table>
<h4>Графік конверсій</h4>
    <div class="graph">
{{div|safe}}
</div>
<h4>Поточні дані експерименту</h4>
  <table class="table">
  <tr id="productName">
  <td>Варіація</td>
  <td>Конверсія</td>
  <td>Кільк. конверсій/ <br>
  кільк. користувачів</td>
  <td>Мінімальна кільк. <br>
  відвідувачів
  </td>
  </tr>
  <tr id="productName">
  <td>A - корзина зверху сайту</td>
  <td>{{convA}}%</td>
  <td>{{a}}/{{s1}}</td>
  <td>{{size}}</td>
  </tr>
  <tr>
  <td>B - корзина знизу сайту</td>
  <td>{{convB}}%</td>
  <td>{{b}}/{{s2}}</td>
  <td>{{size}}</td>
  </tr>
  </table>
  </div>
  </body>
```

Вміст файлу add.css:

```
body{
	background-color: #ddff01;
}
.add {
	margin-bottom: 15px;
}
.remove {
  width: 150px;
  background-color: #13aa52;
  border: 1px solid #13aa52;
  border-radius: 4px;
  box-shadow: rgba(0, 0, 0, .1) 0 2px 4px 0;
  box-sizing: border-box;
  color: #fff;
  cursor: pointer;
  font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-serif;
  font-size: 17px;
  font-weight: 400;
  outline: none;
  outline: 0;
  padding: 5px 5px;
  text-align: center;
  transform: translateY(0);
  transition: transform 150ms, box-shadow 150ms;
  -webkit-user-select: none;
}
.remove:hover {
  box-shadow: rgba(0, 0, 0, .15) 0 3px 9px 0;
  transform: translateY(-2px);
}
.remove a{
	color:#fff;
	text-decoration:none;
	font-weight:bold;
}
```

Вміст файлу admin.css:

```
* {
  font-family: arial, sans-serif;
  box-sizing: border-box;
}
body {
  display: flex;
  min-height: 100vh;
  padding: 0; margin: 0;
  background-color: #ddff01;
}
.admin a:hover{
	border-radius: 0px 15px 0px 0px;
}
#pgside {
  width: 200px;
  transition: width 0.2s;
  background: DarkGreen;
  float: left;
```

```
  position: relative;
  border-radius: 0px 15px 15px 0px;
}

#pgside #pguser {
  display: flex;
  align-items: center;
  padding: 10px 5px;
}

#pgside a {
  color: #ddff01;
  display: block;
  padding: 20px;
  width: 100%;
  text-decoration: none;
  cursor: pointer;
}
#pgside a:hover { background: #9b2323; }

#pgside i.txt { font-style: normal; }
#pgmain {
  flex-grow: 1;
  padding: 20px;
  background-color: #ddff01;
}
@import url(https://fonts.googleapis.com/css?family=Arvo:700);
@import url(https://fonts.googleapis.com/css?family=Seaweed+Script);
.plate {
  width: 100%;
  margin: 10% auto;
}
.script {
  font-family: "Seaweed Script";
  color: #ddff01;
  text-align: center;
  font-size: 40px;
  position: relative;
  margin:0;
}
.script span {
  background-color: #9b2323;
  padding: 0 0.3em;
}
.script:before {
  content:"";
  display: block;
  position: absolute;
  z-index:-1;
  top: 50%;
  width: 100%;
  border-bottom: 3px solid #fff;
}
```

Вміст файлу cart.css:

```
#tableItems {
	margin-left: 20px;
	margin-right: 20px;
	margin-top: 10px;
	margin-bottom: 10px;
	width: 100%;
	display: flex;
	flex-direction: column;
}
body{
	background-color: #cdff01;
}
#itemName {
	margin-left: 5px;
	margin-right: 5px;
	margin-top: 40px;
	margin-bottom: 5px;
	height: 60px;
	width: 200px;
	display: flex;
	flex-direction: column;
	float: left;
}
#titleName {
	width: 200px;
	float: left;
}
#Price{
	float: left;
	font-size: 22px;
	margin-bottom: 5px;
	vertical-align: bottom;
	display:inline-block;
	margin-top: 12px;
}
#product{
	float: left;
	font-size: 35px;
	margin-right: 175px;
	margin-bottom: 5px;
	display:inline-block;
	vertical-align: bottom;
}
#titlePrice {
	float: left;
}
#itemPrice {
	margin-left: 5px;
	margin-right: 5px;
	margin-top: 50px;
	margin-bottom: 5px;
	display: inline-block;
	margin-left: 10%;
	float: left;
```

```
        font-family: 'Source Sans Pro', sans-serif;
}
#image {
        height: 150px;
        width: 100px;
}
#itemImage{
        margin-left: 5px;
        margin-right: 5px;
        margin-top: 5px;
        margin-bottom: 5px;
        display: inline-block;
        float: left;
}
#seperator {
        margin: 0px;
        max-width: 35%;
}
#header{
        float:left;
}
#total {
        width: 100%;
}

#itemNameTag {
        font-weight: bold;
        margin-bottom: 5px;
        font-size: 17px;
        font-family: 'Source Sans Pro', sans-serif;
}
#header{
        margin-left: 20px;
}
#subtotal {
        font-weight: bold;
        font-size: 20px;
}
.avd_div{
 text-align:center;
 display:block;
 float: left;
 margin-top: 20px;
 margin-bottom: 20px;
 }
.but_sad_g_3{
          text-decoration:none;
          color:#fff;
          padding:12px 20px;
          font-size:18px;
          font-weight:bold;
          position:relative;
          border-bottom:1px solid rgba(255,255,255,0.2);
          text-shadow:0 1px 1px #555;
```

```
        border-radius:5px 5px 35px 5px;
       }
.but_sad_g_3:active{
top:7px;
box-shadow: 0 2px 0 #393939, 0px 4px 4px rgba(0,0,0,0.4), 0px 2px 5px rgba(0,0,0,0.2) inset;
 }
 .green_sad{
        background-color:DarkGreen;
        box-shadow:0 5px 5px #508530, 0 9px 0 #385e25, 0 9px 10px rgba(0,0,0,0.4), 0 2px 9px rgba(255,255,255,0.2) inset, 0 -2px 9px rgba(0,0,0,0.2) inset;}
 .green_sad:hover{
        -webkit-box-shadow:0 5px 5px #508530, 0 9px 0 #385e25, 0px 9px 10px rgba(0,0,0,0.4), 0px 2px 15px rgba(255,255,255,0.4) inset, 0 -2px 9px rgba(0,0,0,0.2) inset;
        -moz-box-shadow:0 5px 5px #508530, 0 9px 0 #385e25, 0px 9px 10px rgba(0,0,0,0.4), 0px 2px 15px rgba(255,255,255,0.4) inset, 0 -2px 9px rgba(0,0,0,0.2) inset;
        box-shadow:0 5px 5px #508530, 0 9px 0 #385e25, 0px 9px 10px rgba(0,0,0,0.4), 0px 2px 15px rgba(255,255,255,0.4) inset, 0 -2px 9px rgba(0,0,0,0.2) inset;
        }
 .green_sad:active{
        -webkit-box-shadow:0 2px 2px #385e25, 0px 4px 4px rgba(0,0,0,0.4), 0px 2px 5px rgba(0,0,0,0.2) inset;
        -moz-box-shadow:0 2px 0 #385e25, 0px 4px 4px rgba(0,0,0,0.4), 0px 2px 5px rgba(0,0,0,0.2) inset;
        box-shadow:0 2px 0 #385e25, 0px 4px 4px rgba(0,0,0,0.4), 0px 2px 5px rgba(0,0,0,0.2) inset;
        }
.remove {
  width: 120px;
  background-color: #13aa52;
  border: 1px solid #13aa52;
  border-radius: 4px;
  box-shadow: rgba(0, 0, 0, .1) 0 2px 4px 0;
  box-sizing: border-box;
  color: #fff;
  cursor: pointer;
  font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-serif;
  font-size: 16px;
  font-weight: 400;
  outline: none;
  outline: 0;
  padding: 5px 5px;
  text-align: center;
  transform: translateY(0);
  transition: transform 150ms, box-shadow 150ms;
  -webkit-user-select: none;
}

.remove:hover {
  box-shadow: rgba(0, 0, 0, .15) 0 3px 9px 0;
  transform: translateY(-2px);
}

.remove a{
        color:#fff;
```

```
        text-decoration:none;
        font-weight:bold;
}

.checkoutPrice{
        margin-left: 5px;
        margin-right: 5px;
        margin-top: 50px;
        margin-bottom: 5px;
        display: inline-block;
        margin-left: 10%;
        float: left;
        font-family: 'Source Sans Pro', sans-serif;
}

#CheckoutName {
        margin-left: 5px;
        margin-right: 5px;
        margin-bottom: 5px;
        height: 60px;
        width: 200px;
        display: flex;
        flex-direction: column;
        float: left;
}
```

Вміст файлу edit.css:

```
.form {
  position: relative;
  z-index: 1;
  background: #FFFFFF;
  max-width: 500px;
  padding: 20px;
  text-align: center;
  box-shadow: 0 0 20px 0 rgba(0, 0, 0, 0.2), 0 5px 5px 0 rgba(0, 0, 0, 0.24);
  border-radius: 10px;
  float: left;
}
.form input {
  font-family: "Roboto", sans-serif;
  outline: 0;
  background: #f2f2f2;
  width: 100%;
  border: 0;
  margin: 0 0 15px;
  padding: 15px;
  box-sizing: border-box;
  font-size: 14px;
}
.form button {
  font-family: "Roboto", sans-serif;
  text-transform: uppercase;
  outline: 0;
  background: #4CAF50;
```

```
  width: 100%;
  border: 0;
  padding: 15px;
  color: #FFFFFF;
  font-size: 14px;
  -webkit-transition: all 0.3 ease;
  transition: all 0.3 ease;
  cursor: pointer;
}
.form button:hover,.form button:active,.form button:focus {
  background: #43A047;
}
.form .message {
  margin: 15px 0 0;
  color: #b3b3b3;
  font-size: 20px;
}
.form .message a {
  color: #4CAF50;
  text-decoration: none;
}
body{
        background-color: #ddff01;
}
label{
        float: left;
        font-family: "Roboto", sans-serif;
        color: #43A047;
}
.header-h1 {
    position: relative;
    text-align: center;
        margin-bottom: 1rem;
        margin-top: -1.2rem;
        padding: 0rem;
  }
.header-h1 h1 {
    position: relative;
    display: inline-block;
    background: #fff;
    margin-bottom: 0;
    padding: 0.5rem 1rem;
    border-bottom: .125rem solid #a5d6a7;
    font-size: 1.25rem;
    text-transform: uppercase;
    color: #4caf50;
  }
  .header-h1 h1::before {
    content: "";
    position: absolute;
    left: 50%;
    bottom: -1.25rem;
    transform: translateX(-1.25rem);
    border-top: 1.25rem solid #a5d6a7;
```

```
    border-left: 1.25rem solid transparent;
    border-right: 1.25rem solid transparent;
  }
  .header-h1 h1::after {
    content: "";
    position: absolute;
    left: 50%;
    bottom: -1.125rem;
    transform: translateX(-1.25rem);
    border-top: 1.25rem solid #fff;
    border-left: 1.25rem solid transparent;
    border-right: 1.25rem solid transparent;
  }
```

Вміст файлу home.css:

```
#itemImage {
        height: 200px;
        width: 150px;
}
body{
        background-color: #ddff01;
}
a{
    text-decoration: none;
}
h1 { color: black;
         font-family: 'Trocchi', serif;
         font-size: 45px;
         font-weight: bold;
         line-height: 48px; margin: 0; }

h2 { color: black;
font-family: 'Source Sans Pro', sans-serif;
font-size: 28px;
line-height: 32px;
font-weight: bold;
margin: 0 0 24px; }
td{
        padding-left: 5px;
        padding-right: 5px;
        border-radius: 10px;
        vertical-align:middle;
}
}
#productName {
        text-align: center;
        font-weight: bold;
}
#BookName{
        background: white;
    box-shadow: -5px -5px;
    background: #fce650;
}
#productPrice {
        text-align: center;
```

```
        vertical-align:middle;
}
table {
border-spacing: 0 10px;
font-family: 'Open Sans', sans-serif;
font-weight: bold;
vertical-align:middle;
}
#BookImage{
        box-shadow: 0 0 5px 2px green;
        background: white;
}
#BookPrice{
  font-size: 20px;
  vertical-align:middle;
}
.displayCategory ul li {
        font-size: 20px;
}
.displayCategory ul{
        padding-left:0px;
}
.displayCategory {
    padding:0;
    list-style: none;
    counter-reset: li;
        display: inline-block;
        width: 100%;
        border: 2px solid #D4D4D4; /* Ширина и цвет границ*/
    border-radius: 10px; /* Радиус границ*/
    box-shadow: 0 0 15px #A9A9A9;

}
.Items{         display: inline-block;
        width: 100%;
        border: 2px solid #D4D4D4; /* Ширина и цвет границ*/
    border-radius: 10px; /* Радиус границ*/
    box-shadow: 0 0 15px #A9A9A9;}
.displayCategory li {
    position: relative;
    border-left: 4px solid #337AB7;
    padding:16px 20px 16px 28px;
    margin:12px 0 12px 80px;
    -webkit-transition-duration: 0.3s;
    transition-duration: 0.3s;
}
.displayCategory li:before {
    line-height: 32px;
    position: absolute;
    top: 10px;
    left:-80px;
    width:80px;
    text-align:center;
    font-size: 24px;
```

```
    font-weight: bold;
    color: DarkGreen;
    counter-increment: li;
    content: counter(li);
    -webkit-transition-duration: 0.3s;
    transition-duration: 0.3s;
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}
.displayCategory li:hover:before {
    color: #337AB7;
}
.displayCategory li:after {
    position: absolute;
    top: 26px;
    left: -40px;
    width: 60px;
    height: 60px;
    border: 8px solid #3399FF;
    border-radius: 50%;
    content: '';
    opacity: 0;
    -webkit-transition: -webkit-transform 0.3s, opacity 0.3s;
    -moz-transition: -moz-transform 0.3s, opacity 0.3s;
    transition: transform 0.3s, opacity 0.3s;
    -webkit-transform: translateX(-50%) translateY(-50%) scale(0.1);
    -moz-transform: translateX(-50%) translateY(-50%) scale(0.1);
    transform: translateX(-50%) translateY(-50%) scale(0.1);
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}
.displayCategory li:hover:after {
    opacity: 0.2;
    -webkit-transform: translateX(-50%) translateY(-50%) scale(1);
    -moz-transform: translateX(-50%) translateY(-50%) scale(1);
    transform: translateX(-50%) translateY(-50%) scale(1);
}
table {
border-spacing: 10px 10px;
font-family: 'Open Sans', sans-serif;
font-weight: bold;
border-radius: 0 10px;
}
th {
padding: 10px 20px;
background: #56433D;
color: #F9C941;
border-right: 2px solid;
border-left: 2px solid;
font-size: 0.9em;
}
th:last-child {
```

```
border-right: none;
}
td {
vertical-align: middle;
padding: 10px;
font-size: 14px;
text-align: center;
border-top: 2px solid #56433D;
border-bottom: 2px solid #56433D;
border-right: 2px solid #56433D;
border-left: 2px solid #56433D;
}
```

Вміст файлу items.css:

```
.table {
        width: 100%;
        border: none;
        margin-bottom: 20px;
}
.table tbody td {
        text-align: left;
        border-left: 1px solid #ddd;
        border-right: 1px solid #ddd;
        padding: 10px 15px;
        font-size: 14px;
        vertical-align: top;
}
.table thead tr :first-child, .table tbody tr td:first-child {
        border-left: none;
}
.table thead tr:last-child, .table tbody tr td:last-child {
        border-right: none;
}
.table tbody tr:nth-child(even){
        background: #f3f3f3;
}
.remove {
  width: 200px;
  background-color: #13aa52;
  border: 1px solid #13aa52;
  border-radius: 4px;
  box-shadow: rgba(0, 0, 0, .1) 0 2px 4px 0;
  box-sizing: border-box;
  color: #fff;
  cursor: pointer;
  font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-serif;
  font-size: 30px;
  font-weight: 400;
  outline: none;
  outline: 0;
  padding: 5px 5px;
  text-align: center;
  transform: translateY(0);
  transition: transform 150ms, box-shadow 150ms;
  -webkit-user-select: none;
```

```
}
.remove:hover {
  box-shadow: rgba(0, 0, 0, .15) 0 3px 9px 0;
  transform: translateY(-2px);
}
.remove a{
        color:#fff;
        text-decoration:none;
        font-weight:bold;
}
```

Вміст файлу login.css:

```
login-page {
  width: 500px;
  padding: 8% 0 0;
  margin: auto;
}
.form {
  position: relative;
  z-index: 1;
  background: #FFFFFF;
  max-width: 500px;
  margin: 0 auto 100px;
  padding: 45px;
  text-align: center;
  box-shadow: 0 0 20px 0 rgba(0, 0, 0, 0.2), 0 5px 5px 0 rgba(0, 0, 0, 0.24);
  border-radius: 10px;
}
.form input {
  font-family: "Roboto", sans-serif;
  outline: 0;
  background: #f2f2f2;
  width: 100%;
  border: 0;
  margin: 0 0 15px;
  padding: 15px;
  box-sizing: border-box;
  font-size: 14px;
}
.form button {
  font-family: "Roboto", sans-serif;
  text-transform: uppercase;
  outline: 0;
  background: #4CAF50;
  width: 100%;
  border: 0;
  padding: 15px;
  color: #FFFFFF;
  font-size: 14px;
  -webkit-transition: all 0.3 ease;
  transition: all 0.3 ease;
  cursor: pointer;
}
.form button:hover,.form button:active,.form button:focus {
  background: #43A047;
```

```
}
.form .message {
  margin: 15px 0 0;
  color: #b3b3b3;
  font-size: 20px;
}
.form .message a {
  color: #4CAF50;
  text-decoration: none;
}
body {
  background-color: #ddff01;
  font-family: "Roboto", sans-serif;
  -webkit-font-smoothing: antialiased;
}
```

Вміст файлу ProductDescription.css:

```
#display {
        margin-top: 20px;
        margin-left: 20px;
        margin-right: 20px;
        margin-bottom: 20px;
        border-radius: 10px; /* Радиус границ*/
}
body{
background-color: #cdff01;}
.addToCart {
font-size: 14px;
color: rgb(255, 255, 255);
font-weight: bold;
text-transform: uppercase;
vertical-align: top;
display: inline-block;
background-color: rgb(96, 166, 69);
padding: 0 30px;
border-radius: 3px;
}
h2 {
margin-bottom: 0px;
margin-top: 0px;
}
h1 { color: black;
         font-family: 'Trocchi', serif;
         font-size: 45px;
         font-weight: bold;
         line-height: 48px;  }
a.addToCart {
  font-weight: 700;
  color: white;
  text-decoration: none;
  padding: .8em 1em calc(.8em + 3px);
  border-radius: 3px;
  background: rgb(64,199,129);
  box-shadow: 0 -3px rgb(53,167,110) inset;
  transition: 0.2s;
```

```
}
a.addToCart:hover { background: rgb(53, 167, 110); }
a.addToCart:active {
  background: rgb(33,147,90);
  box-shadow: 0 3px rgb(33,147,90) inset;
}
#Cart{
        float: top;
        position: relative;
}
#productImage {
        height: 250px;
        width: 200px;
        margin-right: 20px;
        display: block;
        float: left;
        box-shadow: 0 0 5px 2px green;
        background: white;
        border-radius: 10px;
}
#description p { color: #333; font-family: 'Muli', sans-serif; margin-top: 0px;  }
#productDescription {
        display: inline;
        font-size: 19px;
        border-radius: 10px;
}

#descriptionTable td {
        width: 150px;
}
#description{
        border:2px solid #3d73a9;
        border-radius: 0 20px 0 0px;
        margin-bottom: 20px;
}
#addToCart {
        font-size: 20px;
}
table {
border-spacing: 10px 10px;
font-family: 'Open Sans', sans-serif;
font-weight: bold;
border-radius: 0 10px;
}
th {
padding: 10px 20px;
background: #56433D;
color: #F9C941;
border-right: 2px solid;
border-left: 2px solid;
font-size: 0.9em;
}
th:last-child {
border-right: none;
```

```
}
td {
vertical-align: middle;
padding: 10px;
font-size: 14px;
text-align: center;
border-top: 2px solid #56433D;
border-bottom: 2px solid #56433D;
border-right: 2px solid #56433D;
border-left: 2px solid #56433D;
border-radius: 20px 0px 0px 0px;
box-shadow: -5px -5px;
}
td:last-child{
        border-radius: 0px 0px 0px 0px;
}
```

Вміст файлу ProfileHome.css:

```
.remove {
  width: 200px;
  background-color: #13aa52;
  border: 1px solid #13aa52;
  border-radius: 4px;
  box-shadow: rgba(0, 0, 0, .1) 0 2px 4px 0;
  box-sizing: border-box;
  color: #fff;
  cursor: pointer;
  margin-bottom: 20px;
  font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-serif;
  font-size: 16px;
  font-weight: 400;
  outline: none;
  outline: 0;
  padding: 5px 5px;
  text-align: center;
  transform: translateY(0);
  transition: transform 150ms, box-shadow 150ms;
  -webkit-user-select: none;
}
.remove:hover {
  box-shadow: rgba(0, 0, 0, .15) 0 3px 9px 0;
  transform: translateY(-2px);
}
.remove a{
        color:#fff;
        text-decoration:none;
        font-weight:bold; }
```

Вміст файлу statistic.html:

```
input[type=range] {
    width: 250px;
    height:20px;
}
h4, h2,h5, span{

margin: 10px;
```

```
}
h2{
        text-align: center;
}
h4{font-family: 'Trocchi', serif;
         font-size: 22px; }
input[type="range"]::-webkit-slider-thumb {
    width: 10px;
    height: 26px;
}
.staticvalues{
 min-height:100%;
 min-width: 400px;
border-color: rgb(190,190,190);
border-right-style: solid;
float: left;
}
.remove {
  width: 350px;
  height: 50px;
  /* float: left; */
  background-color: #13aa52;
  border: 1px solid #13aa52;
  border-radius: 4px;
  box-shadow: rgba(0, 0, 0, .1) 0 2px 4px 0;
  box-sizing: border-box;
  color: #fff;
  cursor: pointer;
  font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-serif;
  font-size: 25px;
  font-weight: 400;
  outline: none;
  outline: 0;
  padding: 5px 5px;
  text-align: center;
  transform: translateY(0);
  transition: transform 150ms, box-shadow 150ms;
  -webkit-user-select: none;
  margin: 10px;
}
.remove:hover {
  box-shadow: rgba(0, 0, 0, .15) 0 3px 9px 0;
  transform: translateY(-2px);
}
.remove a{
        color:#fff;
        text-decoration:none;
        font-weight:bold;
}
.add{
        margin: 10px;
}
.date{
        margin-bottom: 10px; }
```

```
.experiment{
border-top: solid;
border-color: rgb(190,190,190);}
.graph{
        float: left;
        min-widthwidth: 870px;
        margin-left:10px;
        margin-bottom:10px;
        margin-right:5px; }
.table tbody td {
        text-align: left;
        border-left: 1px solid #ddd;
        border-right: 1px solid #ddd;
        padding: 10px 15px;
        font-size: 20px;
        font-family: "Akzidenz Grotesk BQ Medium", -apple-system, BlinkMacSystemFont, sans-
serif;
        vertical-align: top; }
.table thead tr :first-child, .table tbody tr td:first-child {
        border-left: none; }
.table thead tr:last-child, .table tbody tr td:last-child {
        border-right: none; }
.table tbody tr:nth-child(even){
        background: #f3f3f3; }
```

Вміст файлу topStyle.css:

```
#logo {
        height: 40px;
        width: 50px;
        margin-left: 20px;
        margin-top: 15px;
        margin-bottom: 10px;
        margin-right: 20px;
        float: left; }
#title {
        background: url("fon.jpg");
        display: inline-block;
        width: 100%;
        border: 2px solid #D4D4D4; /* Ширина и цвет границ*/
   border-radius: 10px; /* Радиус границ*/
   box-shadow: 0 0 15px #A9A9A9; }
#searchButton {
  position: relative;
  vertical-align: top;
  padding: 0;
  font-size: calc(15px + (30 - 15) * ((100vw - 500px) / (1920 - 500)));
  color: white;
  text-align: center;
  text-shadow: 0 1px 2px rgba(0, 0, 0, 0.25);
  background:DarkGreen;
  border: 0;
  border-bottom: 2px solid #28be68;
  cursor: pointer;
  -webkit-box-shadow: inset 0 -2px #28be68;
```

```
  box-shadow: inset 0 -2px #28be68; }
#searchButton:active {
  top: 1px;
  outline: none;
  -webkit-box-shadow: none;
  box-shadow: none; }
#searchBox {
        height: 30px;
        width: 40%;
        margin-left: 20px;
        margin-top: 20px;
        margin-bottom: 20px;
        margin-right: 20px;
        float: left;
        font-size: 1em;
        border-radius: 3px; }
#searchButton {
        height: 25px;
        margin-top: 23px;
        margin-bottom: 20px;
        margin-right: 20px;
        float: left; }
.dropbtn {
        background-color: DarkGreen;
        color: white;
        padding: 16px;
        font-size: 15px;
        border: none;
        cursor: pointer; }
.dropdown {
        position: relative;
        float: left;
        color:DarkGreen;
        background: DarkGreen;
    font-size: 22px;
    color: white;
    text-align: center;
    text-shadow: 0 1px 2px rgba(0, 0, 0, 0.25);
    border: 0;
    border-bottom: 2px solid #28be68;
    cursor: pointer;
    -webkit-box-shadow: inset 0 -2px #28be68;
    box-shadow: inset 0 -2px #28be68; }
.dropdown-content  {
        display: none;
        position: absolute;
        background-color: #f9f9f9;
        min-width: 70px;
        box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
        border-radius: 0px 0px 10px 10px; }
.dropdown-content a {
        font-size: 14px;
        color:DarkGreen;
        padding: 12px 16px;
```

```
        text-decoration: none;
        display: block; }
.dropdown-content a:hover {background-color: #f1f1f1}
.dropdown:hover .dropdown-content {
        display: block; }
.dropdown:hover .dropbtn {
        background-color: gray; }
#signInButton {
        margin-bottom: 20px;
    float: left;
        height: 48px;
        display: flex;
    align-items: flex-end;
        text-align: bottom;
        color:DarkGreen;
        background: DarkGreen;
        font-size: calc(15px + (30 - 15) * ((100vw - 500px) / (1920 - 500)));
    color: white;
    text-shadow: 0 1px 2px rgba(0, 0, 0, 0.25);
    border-bottom: 2px solid #28be68;
    -webkit-box-shadow: inset 0 -2px #28be68;
    box-shadow: inset 0 -2px #28be68; }
a.link:active, /* активная/посещенная ссылка */
a.link:hover,  /* при наведении */
a.link {
  text-decoration: none;
  color: white; }
#kart {
        margin-left: 20px;
        float: left;
        text-align: center;
        position: relative;
        color:DarkGreen;
        background: DarkGreen;
        font-size: calc(10px + (30 - 15) * ((100vw - 500px) / (1920 - 500)));
    color: white;
    text-shadow: 0 1px 2px rgba(0, 0, 0, 0.25);
    border: 0;
    border-bottom: 2px solid #28be68;
    -webkit-box-shadow: inset 0 -2px #28be68;
    box-shadow: inset 0 -2px #28be68; }
#cartIcon {
        height: 40px;
        width: 40px;
        text-align: center; }
.number {
        float: right;
        margin-right: 20px;
        margin-left:20px;
        text-align: center;
        color:DarkGreen;
        background: DarkGreen;
        font-size: calc(10px + (30 - 15) * ((100vw - 500px) / (1920 - 500)));
    color: white;
```

```
    text-shadow: 0 1px 2px rgba(0, 0, 0, 0.25);
    border-bottom: 2px solid #28be68;
    -webkit-box-shadow: inset 0 -2px #28be68;
    box-shadow: inset 0 -2px #28be68; }
.number p{
        margin:5px; padding:5px; line-height:5px; }
div.page-container {
  max-width: 75%;
  min-width: 1505px;
  margin: 0 auto;
  padding: 0; }
div.page-container-inner {
  min-width: 1505px;
  max-width: 2000px;
  margin: 0 auto;
  padding: 0; }
.fixedbut { position: fixed;
 bottom: 35px;
 right: 10px;
 border-radius: 50%;
 display: block;
 width:100px;
 height:100px;
 background:DarkGreen;
 text-decoration: none;
 text-align: center;
 color: #0096a3;
 border: 0.1875em solid #0F1C3F;
 font-size: 30px;}
.fixedbut img{
     position: absolute;
    margin: auto;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
        width: 80px;
        height: 80px; }
.fixedbut:hover { background: #222; }
```

Додаток В

Матеріали презентації

Слайд 1 – Тема дослідження

Слайд 2 – Наукова новизна, практичне значення

об’єкт та предмет дослідження

Слайд 3 – Мета та завдання роботи

Слайд 4 – Етапи проведення А / В тестування

Слайд 5 – Вибір метрик оцінки ефективності А / В тестування

Слайд 6 – Вибір гіпотез для А / В тестування

Слайд 7 – Кнопки заклику до дії (cta)

Слайд 8 – Розрахунок необхідної кількості користувачів

Слайд 9 – Моделювання структури компонентів web-додатку

Слайд 10 – Моделювання взаємодії користувачів з web-додатком

Слайд 11 – Моделювання бази даних web-додатку

Слайду 12 – Приклад сторінки web-додатку

Слайду 13 – Приклад проведення експерименту А/В тестування

Слайд 14 – Висновки