Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія управління знижками на товарні позиції для мобільних додатків»

Виконав: студент 2-го курсу, групи 2КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Манченко С.В.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. КН
_____ Богач.І.В.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: д.т.н., професор каф. АІІТ
_____ Квєтний Р.Н.
(прізвище та ініціали)
«07.» 12 2023 р.

Допущено до захисту
Завідувач кафедри КН
_____ д.т.н., проф. Яровий А.А.
(прізвище та ініціали)
«____»_____2023 р.

Вінниця ВНТУ - 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д.т.н., проф. Яровий А.А.
29. 08 2023 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Манченко Сергію Віталійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна технологія управління знижками на товарні позиції для мобільних додатків

керівник роботи к.т.н., доцент кафедри КН Богач І.В.
затверджені наказом вищого навчального закладу від "18" 09 2023 року №147
2. Строк подання студентом роботи 13. 09. 2023 року

3. Вихідні дані до роботи:
Середня швидкість відповіді серверу до 1000мс, об'єктно-орієнтована мова програмування, браузерне середовище роботи додатку, мінімальна кількість символів у назві знижки – 3 од., мінімальна кількість типів знижок – 5 од.
4. Зміст текстової частини:
Вступ, аналіз існуючих інформаційних технологій управління знижками, розробка інформаційної технології управління знижками на товарні позиції для мобільних додатків, програмна реалізація інформаційної технології управління знижками на товарні позиції для мобільних додатків, економічна частина, висновки, перелік використаних джерел, додатки
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
Алгоритм роботи веб-додатку управління знижками на товарні позиції для мобільних додатків, інтерфейс користувача, UML-діаграми роботи частин веб-додатку, результати тестування веб-додатку управління знижками на товарні позиції для мобільних додатків.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціалита посада консультанта	Підпис, дата	
		завдання видав	виконання прийняв
1-3	Богач І.В., к.т.н., доц. каф. КН	29.08.23	20.11.23
4	Ратушняк О.Г. к. е. н., доц. каф. ЕПВМ	21.10.23	29.10.23

7. Дата видачі завдання 29.08. 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз інформаційних технологій управління знижками на товарні позиції	1.09.23 - 07.09.23	
2	Розробка інформаційної технології управління знижками на товарні позиції	08.09.23 - 20.09.23	
3	Програмна реалізація та тестування інформаційної технології управління знижками на товарні позиції	21.09.23 - 20.10.23	
4	Підготовка економічної частини	21.10.23 – 29.10.23	
5	Апробація та/або впровадження результатів дослідження	30.10.23 - 05.11.23	
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	06.11.23 - 10.11.23	

Студент _____ Манченко С.В.

Керівник роботи _____ Богач І.В.

АНОТАЦІЯ

УДК 004.42

Манченко С. В. Інформаційна технологія управління знижками на товарні позиції для мобільних додатків. Магістерська кваліфікаційна робота зі спеціальності 122 «Комп'ютерні науки», освітня програма «Системи штучного інтелекту». Вінниця: ВНТУ, 2023. 107 с.

Укр. мовою. Бібліогр.: 24 назв; рис.: 36; табл. 8.

Дана магістерська кваліфікаційна робота присвячена розширенню можливостей інформаційної технології для управління знижками на товарні позиції у мобільних додатках. У контексті зростаючого значення мобільних платформ у сфері торгівлі та ринку споживачів, ця робота вивчає можливості та переваги використання інформаційних технологій для ефективного управління знижками. Робота розглядає такі ключові аспекти: Аналіз сучасних технологій управління знижками та їх переваги; Розробка веб-додатку з використанням JavaScript та React.js для зручного управління товарними позиціями та знижками; Оптимізація продуктивності для забезпечення швидкого та ефективного взаємодії користувачів з додатком; Проведення тестів та валідація функціональності. Економічна оцінка витрат та користі від впровадження нової системи.

Ця робота спрямована на розробку інноваційної інформаційної технології, яка дозволить ефективно управляти знижками на товарні позиції у мобільних додатках, сприяючи поліпшенню досвіду користувачів та оптимізації ділових процесів для підприємств.

Ключові слова: Інформаційна технологія, Управління знижками, JavaScript, Мобільні додатки, Оптимізація продуктивності.

ABSTRACT

Manchenko S.V. Information technology for managing discounts on product positions for mobile applications. Master`s thesis in the specialty 122 «Computer sciences», educational program «Artificial intelligence systems. Vinnytsia: VNTU, 2023. 107 p.

In Ukrainian language. Bibliographer: 24 titles; fig .: 36; table 8.

This master's qualification work is dedicated to expanding the possibilities of an information technology for managing discounts on product items within mobile applications. In the context of the growing significance of mobile platforms in the realm of commerce and consumer markets, this work examines the possibilities and advantages of employing information technology for efficient discount management. The work encompasses the following key aspects: An analysis of contemporary discount management technologies and their benefits; The development of a web application utilizing JavaScript and React.js for convenient product item and discount management; Performance optimization to ensure swift and effective user interaction with the application; Testing and functionality validation; An economic evaluation of costs and benefits resulting from the implementation of the new system.

This work is aimed at the development of an innovative information technology that enables the efficient management of discounts on product items within mobile applications, contributing to improved user experiences and the optimization of business processes for enterprises.

Keywords: Information Technology, Discount Management, JavaScript, Mobile Applications, Performance Optimization.

ЗМІСТ

ВСТУП

Актуальність теми дослідження. Управління знижками на товарні позиції для мобільних додатках стає все більш актуальною проблемою в сучасному бізнес-середовищі. З ростом популярності мобільних платформ та збільшенням конкуренції в цьому секторі, підприємства шукають шляхи поліпшення взаємодії зі споживачами та стимулювання продажів через ефективне управління знижками.

Аналіз сучасних підходів до управління знижками підкреслює необхідність інноваційних технологічних рішень для оптимізації процесів, пов'язаних із знижками та пропозиціями для клієнтів. Інформаційні технології можуть значно полегшити цей процес, забезпечуючи зручний доступ до актуальних знижок, зменшуючи трудомісткість управління та надаючи інструменти для аналізу та взаємодії зі споживачами.

Зростання мобільних додатків у сфері торгівлі та послуг робить інформаційну технологію управління знижками на товарні позиції ключовим елементом конкурентоспроможності. Інтеграція цих технологій дозволить бізнесу швидше реагувати на зміни на ринку та задовольняти очікування клієнтів у реальному часі.

Таким чином, актуальність цього дослідження полягає в необхідності розробки та впровадження інформаційної технології, яка допоможе підприємствам ефективно управляти знижками на товарні позиції у мобільних додатках, забезпечуючи підвищення конкурентоспроможності та задоволення потреб сучасних споживачів. Надалі, подальший розвиток та вдосконалення таких технологій відкриває значний потенціал для покращення якості обслуговування клієнтів та підтримки бізнес-процесів [1].

Зв'язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напряму наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету та плану наукової та навчально-методичної роботи кафедри.

Мета та завдання досліджень. Метою магістерської кваліфікаційної роботи є розширення функціональних можливостей програмного додатку інформаційної технології для ефективного управління знижками на товарні позиції в мобільних додатках.

Для досягнення даної мети, передбачається реалізувати наступні завдання:

1. Провести аналіз сучасних технологій управління знижками на товарні позиції та їх переваг.
2. Обгрунтувати вибір інструментів технічної реалізації та виконати проектування інформаційної технології для управління знижками на товарні позиції для мобільних додатків.
3. Виконати програмну реалізацію та провести тестування інформаційної технології для управління та відстеження знижок на товарні позиції.
4. Зробити економічну оцінку витрат та користі від впровадження нової системи управління знижками на товарні позиції для мобільних додатках.

Дослідження та реалізація цих завдань спрямовані на досягнення мети розробки інформаційної технології, яка полегшить управління знижками на товарні позиції в мобільних додатках, забезпечуючи покращення користувацького досвіду та підтримку бізнес-процесів для підприємств.

Об’єкт дослідження. Об’єктом дослідження є процес управління знижками на товарні позиції для мобільних додатків..

Предметом дослідження. Предметом дослідження є інструменти та технології, які використовуються для створення та функціонування компонентів веб-додатку, що дозволяє управляти знижками на товарні позиції в мобільних додатках.

Методи дослідження. У роботі використані такі методи наукових досліджень: аналіз сучасних технологій та практик управління знижками для мобільних додатках з метою визначення кращих практик та проблем, які потрібно вирішити; вивчення аналогічних програмних рішень та сервісів, використовуваних для управління знижками на товарні позиції для мобільних додатках, для виявлення корисних властивостей та інноваційних підходів;

дослідження алгоритмів та методів організації знижок на товарні позиції для мобільних додатках з метою створення ефективного та універсального адаптивного алгоритму; аналіз проблем та вимог, пов'язаних з організацією та управлінням знижками на товарні позиції для мобільних додатках, для забезпечення належної функціональності та зручності використання; проведення експериментів із програмним забезпеченням для перевірки та оцінки розроблених рішень та алгоритмів; використання об'єктно-орієнтованого програмування для реалізації програмного забезпечення для управління знижками в мобільних додатках.

Наукова новизна одержаних результатів.

Вперше запропоновано інформаційну технологію для управління знижками на товарні позиції в мобільних додатках, що відрізняється від існуючих гнучким підходом до розробки на основі моделі композитної архітектури, що забезпечило розширення функціональних можливостей інформаційної технології та дозволило суттєво спростити процес планування та управління знижками на товарні позиції для мобільних додатків, що робить її більш доступною та зручною для користувачів.

Практичне значення одержаних результатів полягає в наступному.

1. Розроблено алгоритм інформаційної технології для управління знижками на товарні позиції в мобільних додатках на основі композитної архітектури.

2. Розроблено програмний засіб для для управління знижками на товарні позиції в мобільних додатках на основі композитної архітектури з використанням React.

Розроблена інформаційна технологія сприяє покращенню процесів управління знижками на товарні позиції в мобільних додатках та підвищує їх ефективність.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, теоретичні положення в магістерській роботі базуються на чіткому та обґрунтованому

визначенні завдань та цілей дослідження. Перевірка та аналіз роботи програмного забезпечення, розробленого в рамках дослідження, дозволяють підтвердити вірність та функціональність теоретичних положень.

Особистий внесок здобувача. Усі результати, що наведені у магістерській кваліфікаційній роботі, тримані самостійно. У працях, які написано у співавторстві, здобувачу належать: аналіз процесу побудови веб-додатку з використанням React [2].

Апробація результатів. Основні результати досліджень було апробовано на Міжнародній науково-практичній Інтернет-конференції студентів, аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи-2024» (МН-2024). (Вінниця, 2024 р.) [2]

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано тези доповіді на науково-технічній конференції [2].

1 АНАЛІЗ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ УПРАВЛІННЯ ЗНИЖКАМИ НА ТОВАРНІ ПОЗИЦІЇ ДЛЯ МОБІЛЬНИХ ДОДАТКІВ

1.1 Огляд існуючих інформаційних технологій управління знижками

Сучасне комерційне середовище вимагає ефективних інструментів для привертання уваги клієнтів і збільшення продажів, зокрема через надання знижок та промоційних пропозицій. Особливу актуальність ця проблема набуває в мобільних додатках, які стали невід'ємною частиною нашого повсякденного життя. У розділі 1.1 проводиться огляд існуючих інформаційних технологій, які використовуються для управління знижками на товарні позиції в мобільних додатках.

Опис основних інструментів та технологій:

1. QR-коди та промоційні коди: використання QR-кодів та промоційних кодів стало поширеним методом для надання знижок в мобільних додатках. Користувачі можуть сканувати ці коди, отримувати доступ до акційних цін, безкоштовних товарів або інших привілеїв [3]. Приклад QR-коде наведено на рисунку 1.1.

Рисунок 1.1 – Приклад QR-коду

2. Мобільні додатки для збільшення продажів (рисунок 1.2): деякі мобільні додатки розроблені спеціально для просування товарів та послуг через знижки. Вони надають можливість користувачам переглядати акційні пропозиції, отримувати сповіщення про знижки та здійснювати покупки через додаток.

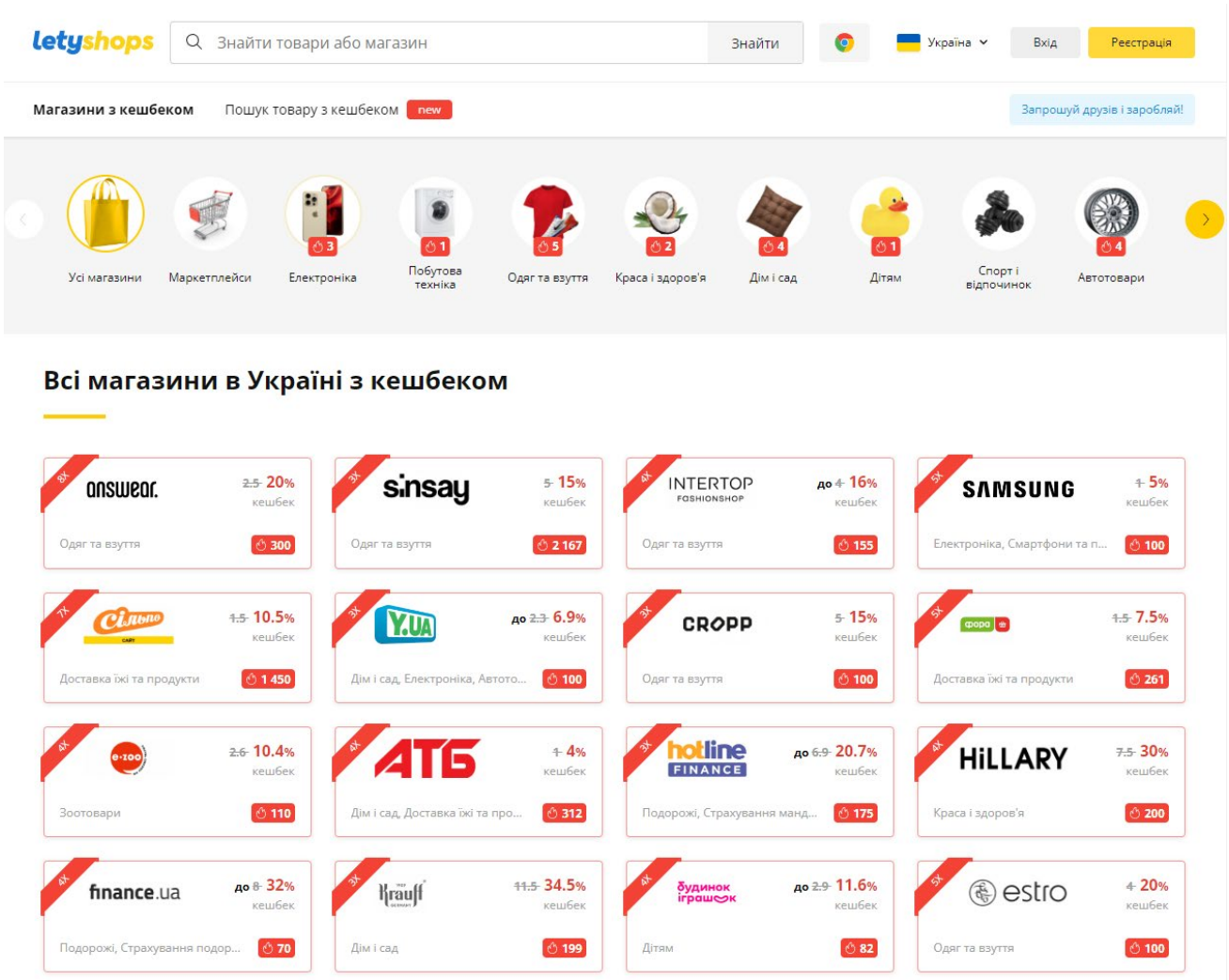

Рисунок 1.2 – Приклад веб-сайту з подібними послугами

3. Системи лояльності: використання програм лояльності дозволяє накопичувати бонуси або отримувати знижки для постійних клієнтів. Ці програми можуть бути інтегровані в мобільні додатки, що сприяє залученню та утриманню клієнтів [4]. Приклад додатку наведено на рисунку 1.3.

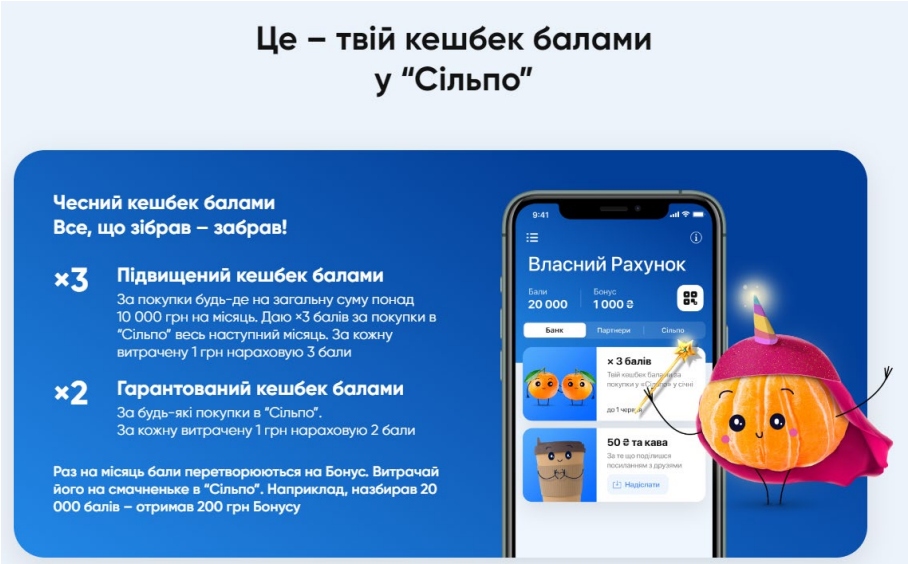

Рисунок 1.3 – Приклад мобільного додатку для отримання кешбеку за покупки

Визначення переваг і недоліків існуючих технологій:

Переваги:

1. Зручність для клієнтів: використання мобільних додатків та QR-кодів дозволяє клієнтам з легкістю користуватися знижками та промоціями.

2. Ефективність: мобільні додатки можуть надавати персоналізовані знижки та рекомендації, що сприяє підвищенню продажів.

3. Можливість аналізу: за допомогою інформаційних технологій можна збирати та аналізувати дані про користувачів та їх поведінку, що допомагає оптимізувати стратегії управління знижками.

Недоліки:

1. Конкуренція: зростання кількості мобільних додатків та пропозицій призводить до жорсткої конкуренції в цьому сегменті ринку.

2. Питання безпеки: використання мобільних технологій може створювати питання щодо захисту особистих даних клієнтів та безпеки операцій.

3. Необхідність постійного оновлення: технології швидко розвиваються, і компанії повинні постійно оновлювати свої системи для відповідності останнім тенденціям та потребам користувачів.

Окрім цих методів є ще ряд інструментів та технологій, які використовуються для управління знижками та промоціями в мобільних додатках. Один з найефективніших та широко застосовуваних підходів - це використання купонів.

Купони є популярним інструментом управління знижками та промоціями в мобільних додатках [5]. Вони представляють собою унікальні коди, які користувачі можуть використовувати для отримання певних переваг або знижок при покупці товарів чи послуг. Основні характеристики та переваги використання купонів в мобільних додатках включають:

1. Персоналізація: мобільні додатки можуть генерувати купони, враховуючи індивідуальні вподобання та покупкову історію користувачів, що робить пропозиції більш привабливими та цікавими для них.

2. Ефективність: купони дозволяють привернути увагу користувачів та стимулюють їх до покупок, що може призвести до збільшення обсягу продажів.

3. Можливість аналізу: інформація про використання купонів, така як час, місце та типи знижок, може бути зібрана та використана для подальшого аналізу та оптимізації стратегії управління знижками.

4. Взаємодія зі зв'язаними послугами: купони можуть бути інтегровані з іншими функціями мобільного додатка, такими як системи лояльності або оплата, що розширює можливості користувачів та створює інтегровану екосистему.

5. Можливість обміну: деякі мобільні додатки дозволяють користувачам обмінювати купони або подарункові картки, що стимулює їх використання та залучає нових клієнтів.

Але окрім перегав також є і недоліки:

1. Витрати: видача купонів може призвести до фінансових витрат, оскільки знижка або подарункова карта може вплинути на прибуток компанії.

2. Активність користувачів: не всі користувачі, які отримали купони, використовують їх. Часто це залежить від привабливості пропозиції та інших факторів.

3. Контроль над використанням: контроль за правильним використанням купонів та запобігання їх недозволеному поширенню може бути важливою задачею.

Купони становлять ефективний інструмент управління знижками та приваблennям клієнтів в мобільних додатках, і їх використання має бути уважно розглянуте в рамках розробки інформаційної технології для управління знижками.

Проведемо порівняльний аналіз існуючих мобільних додатків та рішень, що пропонують знижки та акції, а також аналізуємо їхню функціональність, інтерфейс та відгуки користувачів

Порівняльний аналіз мобільних додатків для знижок та акцій:

1. Dosh - це додаток, який надає можливість користувачам отримувати грошові повернення за покупки в магазинах та ресторанах [6]. Додаток автоматично відстежує покупки та зараховує гроші на баланс користувача. Dosh відзначається зручним інтерфейсом та автоматизованим процесом отримання грошових повернень, інтерфейс наведено на рисунку 1.4. Користувачі можуть легко перевіряти свій баланс та отримувати гроші на свій банківський рахунок. Користувачі високо оцінюють Dosh за можливість отримання грошових повернень без зайвих зусиль. За загальними ідеями мобільний додаток схожий на український мобільний банк – Monobank, концепція кешбеку з кожної покупки.

2. RetailMeNot - це додаток, який надає користувачам доступ до купонів та знижок в різних магазинах та ресторанах [7]. Користувачі можуть переглядати актуальні пропозиції та заощаджувати гроші. Додаток RetailMeNot має велику базу купонів і акцій, і користувачі можуть легко знайти знижки у близьких магазинах. Інтерфейс досить інтуїтивно зрозумілий наведено на рисунку 1.5. Багато користувачів задоволені великою кількістю доступних купонів та акцій в RetailMeNot. Цей додаток теж має схожу концепцію з українським продуктом, а саме LetyShops, веб-додаток має базу партнерів і пропозицій з якими співпрацює,

а користувач активувавши купон, отримає для себе невелику знижку та кешбек з покупки.

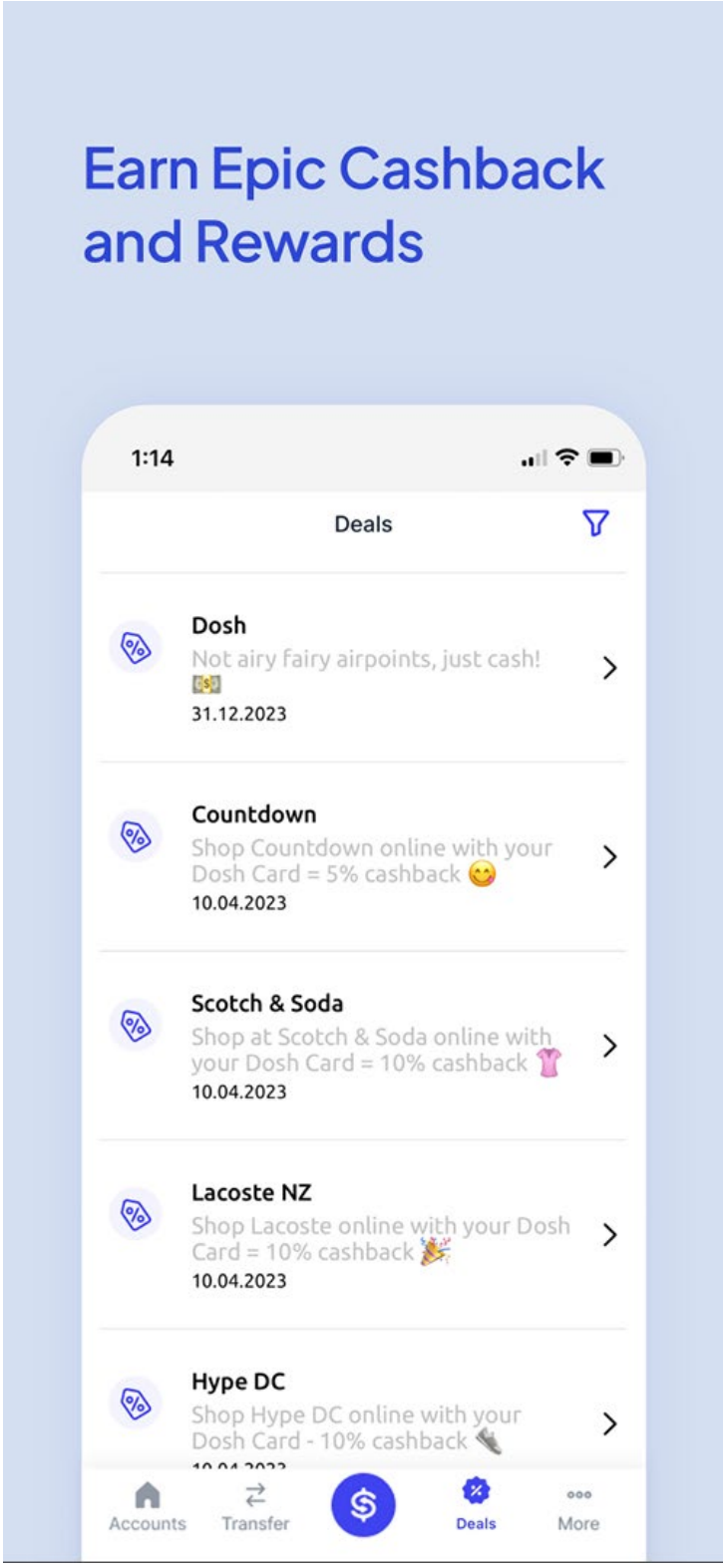

Рисунок 1.4 – Інтерфейс мобільного додатку Dosh

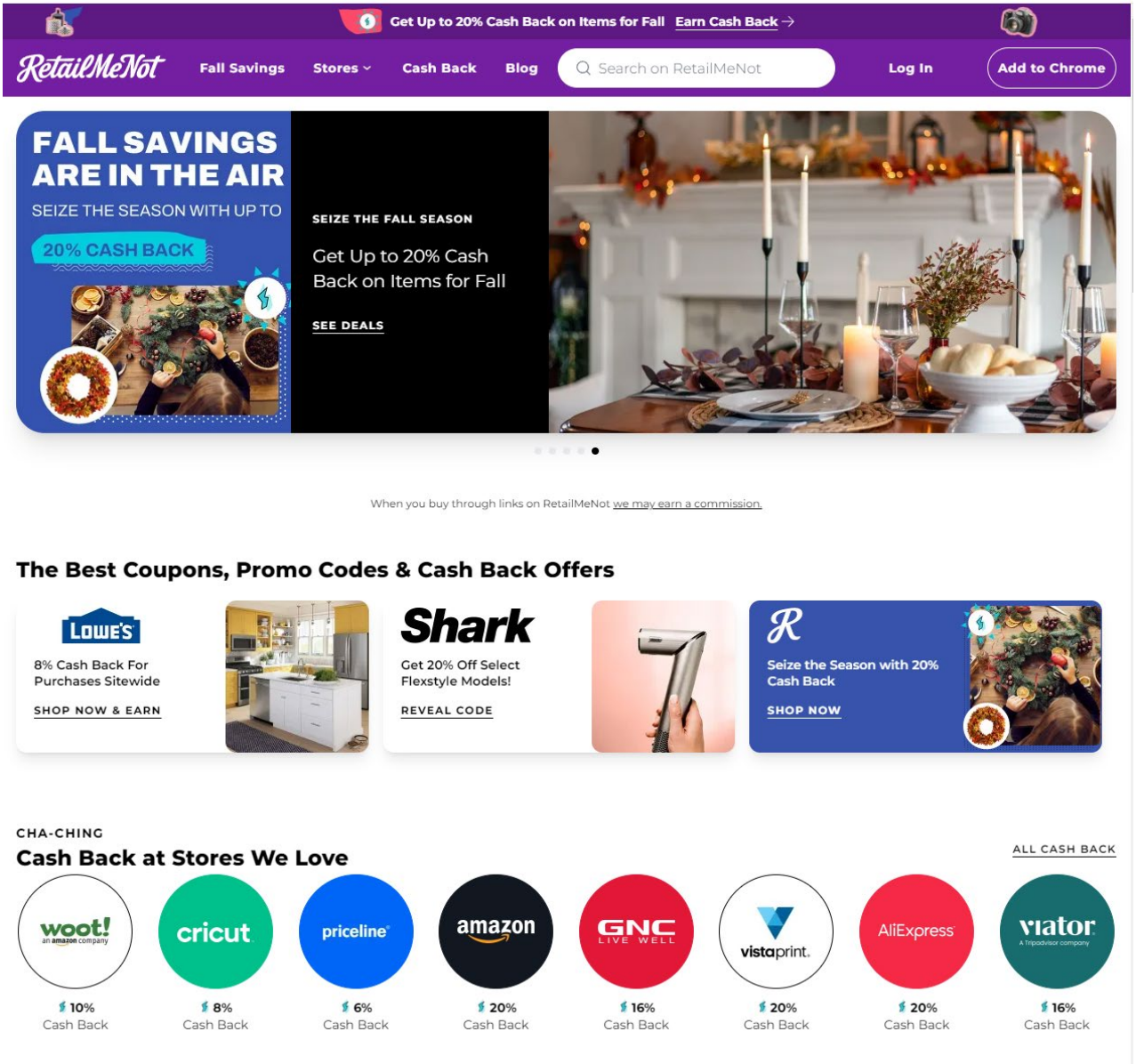

Рисунок 1.5 – Інтерфейс веб-додатку RetailMeNot

3. Honey - це розширення для веб-браузерів та мобільних додатків, яке шукає та застосовує автоматичні купони та знижки під час онлайн-покупок [8]. Honey відзначається своєю легкістю використання, оскільки він автоматично застосовує доступні купони та знижки під час онлайн-покупок. Honey також отримує позитивні відгуки за ефективність та спрощення процесу отримання знижок.

4. Shopify — це платформа електронної комерції, яка дозволяє підприємцям створювати власні онлайн-магазини та ефективно управляти продажами [9]. Shopify пропонує низку інструментів для обробки замовлень,

взаємодії з клієнтами, відстеження запасів та інші функціональності, важливі для успішного ведення електронного бізнесу. Shopify об’єднує всі ваші комерційні дії на одній платформі з одним адміністратором для доступу до всього: від замовлень до аналітики та каналів продажів. За допомогою Shopify власники бізнесу можуть створювати й налаштовувати онлайн-магазин і продавати в багатьох місцях, зокрема в Інтернеті, на мобільних пристроях і особисто через звичайні локації та спливаючі магазини. Замовлення та інформацію про клієнтів можна синхронізувати в кількох каналах, від соціальних мереж до онлайн-ринків, завдяки потужній інтеграції та керувати всім в одному місці. Shopify є абсолютно безпечним і розміщеним у хмарі, що означає, що користувач може безпечно отримати доступ до нього з будь-якого сумісного пристрою з підключенням до Інтернету. Це надає можливість вести свій бізнес з будь-якого місця. Щодо системи знижок в Shopify платформа надає можливості для налаштування різних видів знижок і промокодів. Основні можливості знижок у Shopify включають:

1. Процентні знижки: Встановлення фіксованого відсотка знижки на всі або певні товарні позиції.
2. Фіксовані знижки: Визначення конкретної суми знижки на замовлення або окремі товари.
3. Безкоштовна доставка: Надання можливості безкоштовної доставки для певних замовлень або товарів.
4. Промокоди: Генерація унікальних промокодів, які клієнти можуть використовувати при оформленні покупки для отримання знижки.
5. Знижки на кількість: Встановлення знижок в залежності від кількості придбаних товарів.
6. Знижки на товари для певних груп клієнтів: Можливість налаштування спеціальних знижок для певних клієнтів або груп клієнтів.
7. Знижки на сезонні продажі та акції: Можливість автоматично встановлювати та відключати знижки на певні товари або всі асортименти товарів під час сезонних розпродажів або акцій.

Один з прикладів процентної знижки, яка застосовується при купівлі користувачем певного товару, зменшуючи його ціну, наведено на рисунку 1.6. Знижка може використовуватись в залежності від додаткових налаштувань адміністратором, також разом зі знижкою можливо використати промокод, якщо це дозволено налаштуваннями.

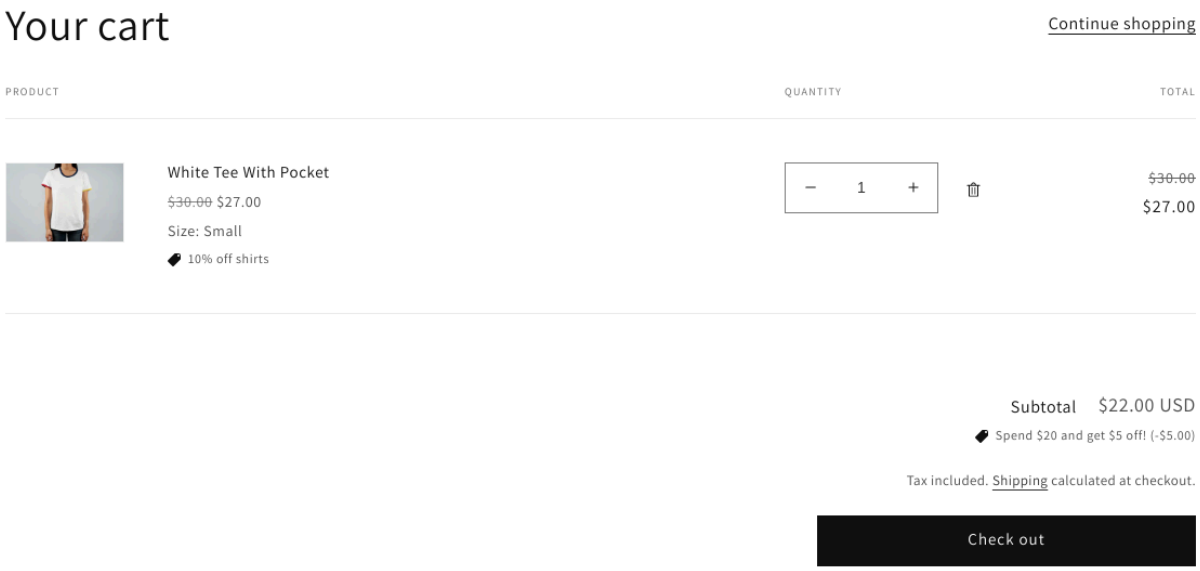

Рисунок 1.6 – Інтерфейс веб-додатку Shopify

Отже проведено загальний огляд існуючих інформаційних технологій управління знижками, який вказує на актуальність і важливість подальшого дослідження та розробки інформаційної технології для управління знижками на товарні позиції в мобільних додатках

1.2 Аналіз потреб та вимог користувачів мобільних додатків

Важливим етапом у розробці інформаційної технології для управління знижками для користувачів мобільних додатків є ретельний аналіз потреб та вимог користувачів мобільних додатків. Цей аналіз є ключовим для створення інструменту, який відповідає очікуванням і сприяє задоволенню користувачів.

Вивчення потреб цільової аудиторії щодо отримання знижок та акцій в мобільних додатках. Для ефективного управління знижками на товарні позиції в мобільних додатках необхідно ретельно дослідити, які саме знижки та акції є найбільш привабливими для цільової аудиторії. Це включає в себе вивчення споживчих звичок, сезонних попитів, популярних категорій товарів та багато інших аспектів. Основною метою цього дослідження є встановлення того, що саме важливо користувачам, коли мова йде про знижки та пропозиції, і які види акцій найбільш важливі для їхнього задоволення. Визначення очікувань адміністраторів від інтерфейсу та функціональності. Побажання та очікування адміністраторів веб-додатку для управління знижками стосовно інтерфейсу та функціональності додатка грають важливу роль у процесі розробки. Аналізуючи їх, ми здатні створити інтерфейс, який є зручним та легким у використанні, а також надати функціональність, яка відповідає потребам користувачів. Очікування щодо вигляду, функцій, швидкодії та безпеки додатка допомагають побудувати досконалий продукт, який задовольняє як адміністраторів веб-додатку, які використовують його для своїх потреб, так і користувачів та робить процес отримання знижок максимально комфортним для обох сторін.

1.3 Технічні можливості та обмеження

Розробка інформаційної технології для управління знижками на товарні позиції в мобільних додатках передбачає ретельний аналіз технічних можливостей та обмежень, які стосуються мобільних платформ та операційних систем.

Аналіз технічних характеристик мобільних платформ та операційних систем для реалізації знижок. Першим кроком є вивчення технічних особливостей популярних мобільних платформ, таких як Android та iOS, а також операційних систем, які вони підтримують. Це включає аналіз можливостей створення та впровадження знижок у додатках, підтримки мобільних платежів, інтеграції з різними платформами соціальних мереж і багато інших аспектів.

Важливо з'ясувати, які саме функції можуть бути реалізовані на кожній платформі та як це вплине на користувачів.

Android - це одна з найпоширеніших мобільних платформ у світі, відома своєю різноманітністю пристроїв і відкритістю для розробників [10]. Розробники мають можливість створювати різноманітні додатки, включаючи ті, які пов'язані зі знижками та акціями. Для реалізації знижок у додатках для Android можна використовувати Google Play Billing Library для обробки мобільних платежів та Android's In-App Purchase API для впровадження системи оплати.

Як операційна система, забезпечує розробникам багатий функціонал для інтеграції знижок. Вона підтримує різні методи платежів, включаючи кредитні карти, платіжні системи та інші. Також Android має широкий набір інструментів для роботи з користувачами, включаючи сповіщення, роботу з GPS, камерами тощо.

Операційна система iOS використовується на пристроях, вироблених компанією Apple, і відрізняється своєю стабільністю та безпекою [11]. Якщо ваша цільова аудиторія в основному використовує продукцію Apple, вам слід розглянути розробку додатка для iOS. Для реалізації знижок у додатках для iOS можна використовувати Apple's StoreKit Framework та інструменти для обробки платежів в магазині App Store. iOS, у свою чергу, славиться своєю високою безпекою та якістю. Інструменти розробки Apple дозволяють створювати додатки, які легко інтегруються з іншими пристроями та послугами Apple. Впровадження знижок на платформі iOS може бути безпечним та надійним.

Завдяки аналізу технічних характеристик мобільних платформ і операційних систем, розробники матимуть змогу визначити, які конкретні можливості кожної платформи можуть бути використані для реалізації системи знижок та як забезпечити максимальний комфорт для користувачів під час отримання та використання знижок та акцій у мобільних додатках.

1.4 Постановка задачі і формулювання вимог до інформаційної технології управління знижками на товарні позиції для мобільних додатків

У контексті зростаючого інтересу до мобільних додатків та цифрової торгівлі, ефективне управління знижками на товарні позиції стає ключовим аспектом для привертання та утримання користувачів. Оскільки більшість систем для управління знижками веб-орієнтованими та їхні дані оновлюються у реальному часі, правильним рішенням буде розробити інформаційну технологію у вигляді веб додатку, який в свою чергу буде підтримувати всі основні функції браузера та буде працювати незалежно від операційної системи користувача, також за допомогою сучасних інструментів розробки додаток буде кросплатформеним, тобто буде працювати як з ПК так і з мобільного телефону.

Постановка задачі та формулювання вимог до інформаційної технології управління знижками в мобільних додатках визначається наступним чином:

1. Інтеграція із системою управління продажами: забезпечення безперебійної інтеграції з основною системою управління продажами для автоматичного визначення доступних знижок та їх відображення у мобільному додатку.

2. Гнучка система налаштувань знижок: створення інтуїтивного інтерфейсу для адміністраторів, де вони можуть встановлювати та налаштовувати різноманітні види знижок, такі як фіксовані знижки, акційні та сезонні пропозиції.

3. Персоналізовані пропозиції для користувачів: можливість створення персоналізованих знижок на основі покупкової історії, вподобань та поведінки користувача для підвищення лояльності та покращення користувацького досвіду.

4. Управління строками дії знижок: розробка механізму для автоматичного визначення строків дії знижок, а також можливість попереднього налаштування часових рамок для акцій та спеціальних пропозицій.

5. Аналітика та звітність: включення в систему інструментів аналізу ефективності знижок, зокрема, збору даних щодо реакції користувачів, конверсії та впливу на обсяги продажів.

6. Захист від зловживань: розробка заходів для попередження можливих зловживань системою знижок, включаючи обмеження на кількість використань та визначення правил для предотвращення фроду.

7. Мобільність та кросплатформенність: забезпечення функціональності системи на різних мобільних платформах (Android, iOS) та адаптація інтерфейсу для зручного використання на різних пристроях.

1.5 Висновок до розділу 1

В першому розділі роботи проведено аналіз інформаційних технологій, Визначено, які технології на сьогоднішній день є найбільш актуальними та популярними. Вивчені потреби цільової аудиторії, яка має інтерес до отримання знижок та акцій через мобільні додатки. Також було визначено очікування користувачів щодо інтерфейсу та функціональності таких додатків. Проаналізовані технічні характеристики мобільних платформ та операційних систем, які впливають на реалізацію інформаційної технології для управління знижками. Також визначено технічні обмеження, які можуть вплинути на процес розробки та впровадження інформаційної технології. Загальний аналіз перших чотирьох розділів вказує на актуальність і важливість розробки інформаційної технології для управління знижками на товарні позиції в мобільних додатках. Даний дослідницький проект спрямований на задоволення потреб користувачів та надання їм зручного та ефективного інструменту для отримання знижок та акцій в мобільних додатках. Дані з першого розділу послужать підставою для подальших досліджень і розробки інформаційної технології.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ ЗНИЖКАМИ НА ТОВАРНІ ПОЗИЦІЇ

2.1 Опис загальної архітектури інформаційної технології

У цьому розділі буде детально розглянуто загальну структуру і архітектуру інформаційної технології для управління знижками на товарні позиції в мобільних додатках.

Ця інформаційна технологія буде базуватися на розподіленій архітектурі, що включатиме в себе серверну та клієнтську частини. Вона буде взаємодіяти з мобільними додатками користувачів, дозволяючи їм отримувати знижки та акції на товари. Основні компоненти системи включатимуть:

1. Серверну частину: ця частина відповідає за обробку запитів від мобільних додатків користувачів. Вона забезпечуватиме роботу з базою даних, аналіз і управління знижками та акціями, а також взаємодію з іншими інформаційними системами.

2. Клієнтську частину: клієнтська частина буде представлена мобільним веб-додатком, який адміністратори використовуватимуть. Він забезпечуватиме зручний інтерфейс для адміністраторів.

В цьому випадку система матиме обмежену архітектуру, спрощену до необхідних функцій адміністрування та роботи зі знижками. Адміністратор буде мати можливість створювати купони на знижки для конкретних товарних позицій, вказуючи необхідні умови та параметри, такі як розмір знижки та термін дії, зможете відстежувати та управляти створеними купонами, переглядати їх та змінювати їх параметри за потреби, матимете можливість видаляти купони, які вже неактуальні або необхідні.

Розробка додатку буде з використанням React, Redux і TypeScript це є відмінним вибором і забезпечить високу якість та продуктивність додатку.

Розглянемо кожну з цих технологій окремо та їх переваги:

1. React – це бібліотека для створення користувацьких інтерфейсів. Вона відома своєю декларативністю, швидкістю та ефективністю. У порівнянні з іншими технологіями на ринку, React є новою технологією. Джордан Волке, інженер-програміст Facebook, заснував бібліотеку в 2011 році, давши їй життя. Подібні до XHP, простому фреймворку HTML-компонентів для PHP, впливають на React. Новинна стрічка React була його дебютним додатком у 2011 році. Пізніше Instagram підбирає його та включає в свою платформу [12]. Популярність React сьогодні затьмарила популярність усіх інших інтерфейсних фреймворків розробки. Ось чому:

- Просте створення динамічних додатків: React полегшує створення динамічних веб-додатків, оскільки вимагає менше кодування та пропонує більше функціональних можливостей, на відміну від JavaScript, де кодування часто стає складним дуже швидко.

- Покращена продуктивність: React використовує Virtual DOM, завдяки чому швидше створюються веб-додатки. Віртуальний DOM порівнює попередні стани компонентів і оновлює лише ті елементи в реальному DOM, які були змінені, замість того, щоб оновлювати всі компоненти знову, як це роблять звичайні веб-програми.

- Багаторазові компоненти: компоненти є будівельними блоками будь-якої програми React, і одна програма зазвичай складається з кількох компонентів. Ці компоненти мають свою логіку та елементи керування, і їх можна повторно використовувати в усій програмі, що, у свою чергу, значно скорочує час розробки програми.

- Односпрямований потік даних: React слідує за односпрямованим потоком даних. Це означає, що під час розробки програми React розробники часто вкладають дочірні компоненти в батьківські компоненти. Оскільки дані передаються в одному напрямку, стає легше виправляти помилки та знати, де виникає проблема в програмі в даний момент.

- Невелика крива навчання: React легко освоїти, оскільки він здебільшого поєднує базові концепції HTML і JavaScript з деякими корисними доповненнями. Однак, як і у випадку з іншими інструментами та фреймворками, вам доведеться витратити деякий час, щоб правильно зрозуміти бібліотеку React.

- Його можна використовувати для розробки як веб-, так і мобільних додатків: ми вже знаємо, що React використовується для розробки веб-додатків, але це ще не все, що він може зробити. Існує фреймворк під назвою React Native, похідний від самого React, який надзвичайно популярний і використовується для створення красивих мобільних додатків. Отже, насправді React можна використовувати для створення як веб-, так і мобільних додатків.

- Спеціальні інструменти для легкого налагодження: Facebook випустив розширення Chrome, яке можна використовувати для налагодження програм React. Це робить процес налагодження веб-додатків React швидшим і простішим.

- JSX — це синтаксичне розширення JavaScript. Це термін, який використовується в React для опису того, як має виглядати інтерфейс користувача. Ви можете писати структури HTML у той самий файл, що й код JavaScript, використовуючи JSX [13].

Наведені вище причини з лишком виправдовують популярність бібліотеки React і чому її приймає велика кількість організацій і компаній. Тепер давайте познайомимося з функціями React.

React допомагає створювати компоненти і реюзабельні інтерфейси, спрощуючи процес розробки та підтримки. Швидкість розробки: react допомагає створювати компоненти швидше та ефективніше завдяки своїй декларативній природі. Redux дозволяє ефективно керувати станом додатку, що полегшує розширення та збереження коду. Схема прикладу структури компонентів показано на рисунку 2.1.

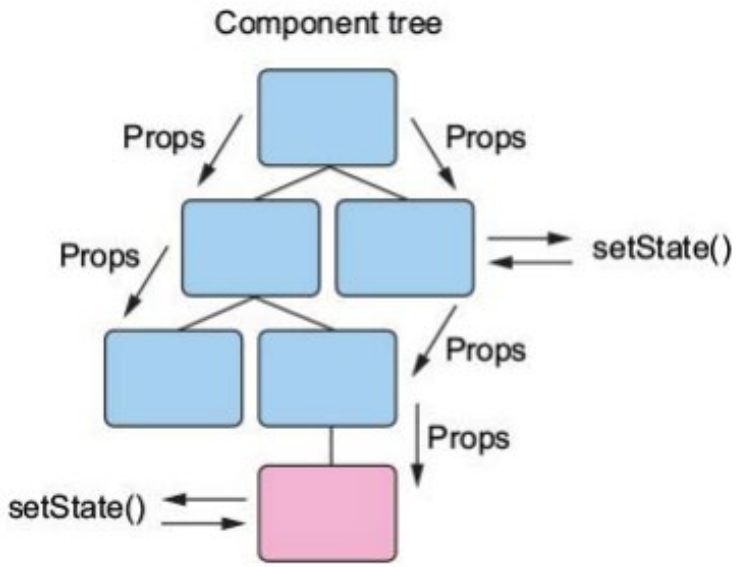

Рисунок 2.1 – Схема компонентів «React»

2. Redux – це бібліотека для керування станом додатку. Вона надає засоби для ефективного управління станом додатку та створення централізованих зберігань даних [14]. Redux особливо корисний великим додаткам та додаткам зі складними структурами даних. Redux робить управління станом простим та масштабованим, що особливо важливо у великих проектах. Хоча він здебільшого використовується як інструмент керування станом із React, ви можете використовувати Redux із будь-яким іншим фреймворком чи бібліотекою JavaScript. Він легкий — 2 КБ (включаючи залежності), тож вам не потрібно турбуватися про те, що він збільшить розмір ресурсу вашої програми. Redux дозволяє вам керувати станом програми в одному місці та робити зміни в програмі більш передбачуваними та відстежуваними, що полегшує розуміння змін, які відбуваються у вашій програмі. Але всі ці переваги супроводжуються низкою проблем. Деякі розробники стверджують, що Redux представляє непотрібний шаблон, потенційно ускладнюючи те, що інакше є простими завданнями. Однак це залежить від архітектурних рішень проекту. Redux використовується для підтримки та оновлення даних у ваших програмах для спільного використання кількома компонентами, залишаючись незалежними від компонентів, зображено на рисунку 2.2.

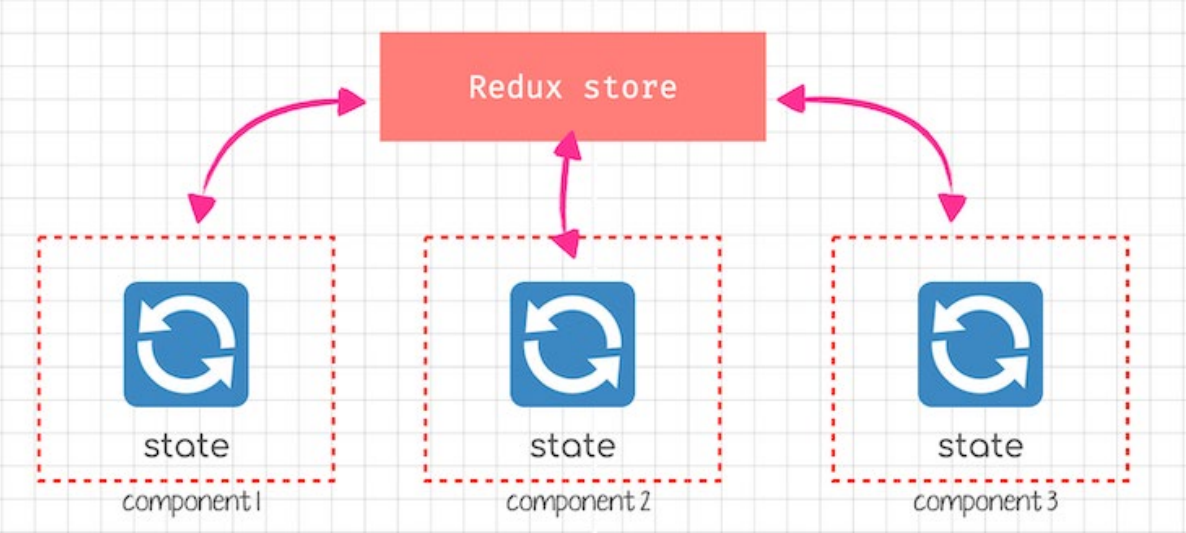

Рисунок 2.2 – Схема сховища «Redux»

Якщо дані потрібно передати від батьківського до дочірнього в глибині дерева, це все ще можна зробити за допомогою утиліт React, таких як Context. Але коли справа доходить до спільного використання стану між компонентами на одному рівні, Redux є неминучим варіантом.

Це тому, що React дозволяє лише односпрямований потік даних. Це означає, що дані не можуть бути надіслані від дитини до батьків; вона повинна текти вниз від батьків до дитини. Ця модель мислення дуже добре працює з Redux, де ми не можемо безпосередньо змінити стан. Замість цього ми надсилаємо дії, спрямовані на зміну стану, а потім окремо спостерігаємо за результатами змін стану.

Більшість бібліотек, таких як React і Angular, створено таким чином, щоб компоненти могли внутрішньо керувати своїм станом без потреби у зовнішній бібліотеці чи інструменті. Це добре працює для додатків з невеликою кількістю компонентів, але в міру того, як додаток зростає, керування спільним станом між компонентами стає складним.

Ось простий приклад компонента входу в React. Вхідні дані компонента входу впливають на те, що відображається його однорідним компонентом, компонентом status:

```
import React, { useState } from 'react';

const App = () => {
  const [count, setCount] = useState(0);

  const handleIncrement = (incrementBy) => {
    const numberToIncrement = incrementBy || 1;
    setCount(count + numberToIncrement)
  }

  return (
    <div>
      <BigCountDisplay count={count} />
      <CounterButton onIcrement={handleIncrement} />
    </div>
  );
}

export default App;
```

А тепер уявіть, що станеться, коли стан потрібно розділити між компонентами, які знаходяться далеко один від одного в дереві компонентів. По суті, стан потрібно буде підняти до найближчого батьківського компонента і вгору, доки він не дійде до найближчого спільного предка обох компонентів, яким потрібен стан, а потім він передається. Це робить стан компоненту складним для підтримки та менш передбачуваним. Зрозуміло, що управління станом стає безладним, оскільки додаток стає складнішим.

Схема структури сховища при використанні бібліотеки Redux зображена на рисунку 2.3.

3. TypeScript - це додаткова опція до JavaScript, яка додає сильну типізацію до мови. Він дозволяє визначити типи для змінних, функцій та об'єктів, що полегшує виявлення помилок та покращує автокомплітацію у вашій розробці. TypeScript особливо корисний при роботі над великими проектами, де точність та безпека є важливими [15]. TypeScript допомагає уникнути багатьох типових помилок на етапі розробки і покращує роботу з кодом.

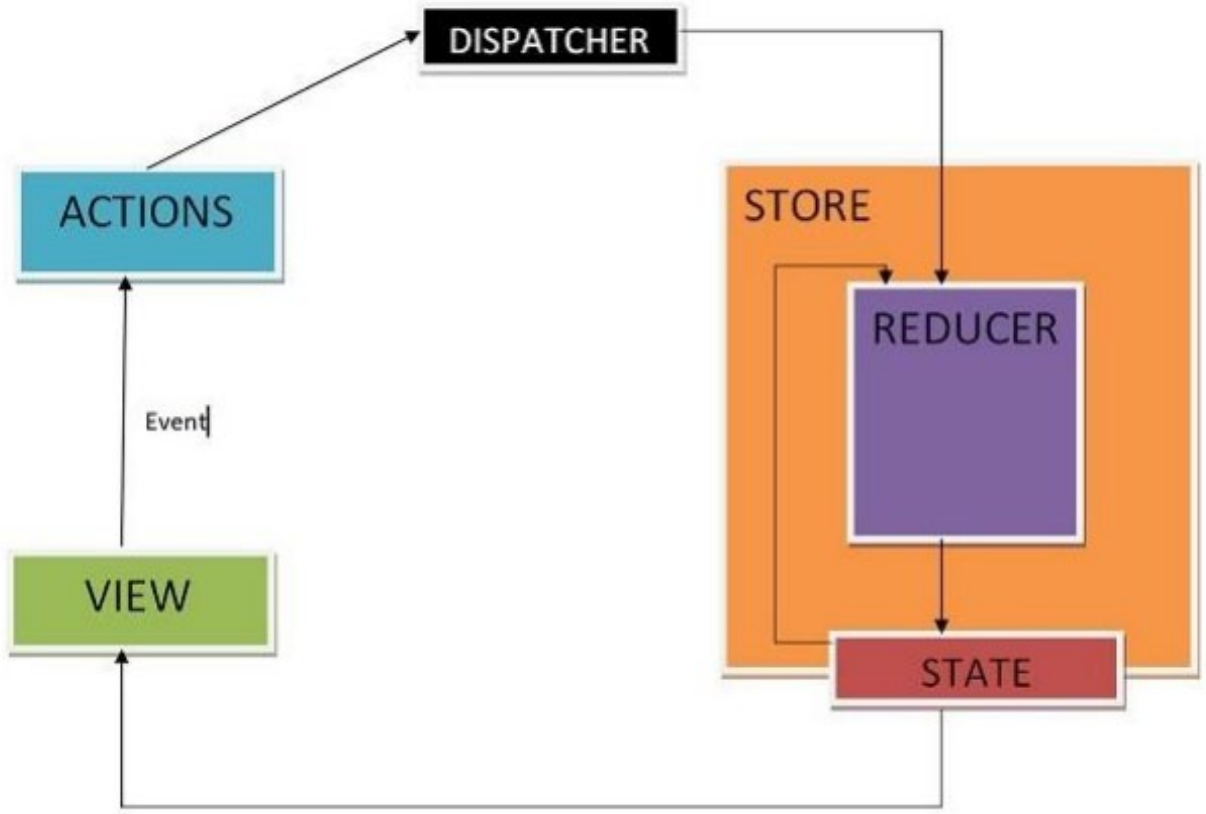

Рисунок 2.3 – Схема архітектури «Redux»

TypeScript був створений Microsoft і випущений у 2012 році після двох років розробки. Він був створений для забезпечення додаткової статичної перевірки типу, яка була б особливо корисною під час розробки великомасштабних програм. Цікаво, що одна з причин, чому Microsoft розробила TypeScript, полягала в тому, що їхні внутрішні команди мали проблеми з масштабуванням JavaScript для власних проектів Microsoft, зокрема команди, яка працювала над Bing Maps [16].

JavaScript типізується динамічно. Таким чином, програми, написані на JavaScript, не знають типу даних змінної, доки цій змінній не буде присвоєно значення під час виконання. Змінну можна перепризначити або примусово призначити значення іншого типу без проблем або попереджень. Це може призвести до помилок, які часто не помічаються, особливо у великих програмах.

TypeScript, з іншого боку, використовує статичний тип. Змінним можна надати тип, коли вони оголошені. TypeScript перевірятиме типи під час

компіляції та видаватиме помилку, якщо змінній буде надано значення іншого типу. Однак помилка не перешкоджає виконанню коду. Код усе одно буде скомпільовано у звичайний JavaScript і працюватиме нормально. Таким чином, TypeScript схожий на «перевірку правопису» для вашого коду. Це повідомить вам, коли щось виглядає не так, але це не змінить роботу вашого коду.

Нижче ми призначаємо рядок типу параметру функції, зображено на рисунку 2.4.

```
function askForFood(food: string) {
console.log(`May I please have ${food}?`);
}
```

Рисунок 2.4 – Приклад функції з описаним типом даних

Якщо ми викличемо askForFood(11), ми отримаємо цю помилку під час компіляції: Аргумент типу «11» не можна призначити параметру типу «рядок».

TypeScript також повідомить вам, якщо функцію було викликано з неочікуваною кількістю аргументів. Якщо ми викличемо askForFood(), ми отримаємо повідомлення про помилку: Очікувався 1 аргумент, але отримано 0.

Ми можемо використовувати інтерфейси для надання типів властивостям об’єкта. Ми використовуємо інтерфейс, коли створюємо об’єкти, які повинні мати властивості, що збігаються з описаними в інтерфейсі, що зображено на рисунку 2.5.

Під час компіляції наведеного вище коду ми побачимо помилку: Типи властивості 'graduationYear' несумісні. Тип «рядок» не можна призначити типу «число».

Окрім відображення помилок під час компіляції, певні IDE, наприклад Visual Studio Code, відображатимуть помилки розробнику під час введення коду. Це робить виправлення простих необережних помилок набагато швидшим і легшим.

```
interface Student {
  name: string,
  major: string
  graduationYear: number
}

const bestStudent: Student = {
  name: 'Anne Smith',
  major: 'History',
  graduationYear: '2018'    //notice we are using a string
}
```

Рисунок 2.5 – Приклад описаного інтерфейсу

TypeScript базується на JavaScript, щоб ефективно заповнити прогалини та надати розробникам кращі інструменти будь-якого масштабу. Порівняно з конкурентами, TypeScript простіший і ефективніший.

Отже, загальна архітектура буде орієнтована на високу продуктивність, з урахуванням потреб для розробки з використанням сучасних інструментів, що покращать можливі процеси. Розробка такої архітектури враховуватиме технічні особливості мобільних платформ та операційних систем для оптимальної функціональності та ефективності веб-додатку

2.2 Проектування інтерфейсу веб-додатку для адміністратора

З урахуванням широкого поширення Інтернету в розробці веб-додатків сьогодні виникає складність на рівні розробки веб-додатків, подібно до прикладних програм. Особливо важливим та ресурсомістким етапом є проектування інтерфейсу користувача (UI). Зараз кількість інтернет-користувачів по всьому світу становить приблизно 1 мільярд, тому бізнес та інші галузі вимагають від продуктів максимальної продуктивності, функціональності та зручності. Завдання, пов'язані із створенням якісного інтерфейсу користувача, займають близько 70% від загального обсягу робіт.

Люди іноді плутають ці два, але UI насправді є спеціалізованою підмножиною UX. За словами Х’юго Реймонда, дизайнера Figma, привабливий

інтерфейс користувача закладає основу для позитивного загального досвіду роботи з цифровим продуктом або веб-сайтом. «Ефективний дизайн інтерфейсу користувача поєднує зручність використання та інтерактивний дизайн для створення емоційного зв’язку між користувачами та продуктами», — пояснює він [17].

Щоб створити привабливий інтерфейс користувача, дизайнери враховують ці чотири ключові елементи:

Макет сторінки. В ідеалі організація веб-сторінки або екрана мобільного додатка має здаватися користувачам інтуїтивно зрозумілою. Але щоб організувати це таким чином, дизайнери інтерфейсу користувача повинні прийняти десятки добре зважених рішень — від положення заголовка до кількості білого простору.

Колірна схема і вибір шрифту. Дизайнери інтерфейсу користувача ретельно обирають кольори та шрифти в інтерфейсі цифрового продукту для узгодженості, доступності та відповідності бренду.

Інтерактивні елементи. Від дизайну кнопок до спадних меню, дизайнери інтерфейсу користувача стилізують екрани цифрових продуктів, щоб зробити роботу користувача інтуїтивно зрозумілою.

Точність каркаса та прототипу. UX-дизайнери часто створюють базові каркаси та прототипи. Дизайнери інтерфейсу користувача можуть допомогти перетворити їх у високоточні, функціональні, інтерактивні макети продукту.

У цифровому дизайні користувальницький інтерфейс (UI) стосується інтерактивності, зовнішнього вигляду та відчуття від екрана продукту або веб-сторінки, тоді як користувальницький досвід (UX) охоплює загальний досвід роботи користувача з продуктом або веб-сайтом. Читайте далі, щоб дізнатися, що потрібно для розробки привабливого інтерфейсу користувача та створення UX, що запам’ятовується.

UX-дизайн — це більше, ніж здається на перший погляд в інтерфейсі користувача. Процес дизайну UX включає дослідження ринку, розробку каркасу, тестування прототипу та міжфункціональну співпрацю.

Є п’ять важливих кроків до успішного дизайну UX:

Дослідження споживачів і конкурентів. Щоб забезпечити позитивний досвід користувача, UX-дизайнери повинні розуміти свою цільову аудиторію. Завдяки дослідженню UX вони дізнаються, що подобається їхнім користувачам, з якими проблемами й проблемами вони стикаються та як вони поводяться в Інтернеті чи під час використання програми чи програмного забезпечення. UX-дизайнери також можуть виконувати аналіз конкурентів за допомогою шаблону аналізу SWOT, щоб визначити нішу своєї продукції.

UX-дизайнери часто консолідують результати досліджень користувачів у покупців або користувачів, які є детальним описом типів цільової аудиторії.

Інформаційна архітектура. Коли дизайнери UX зрозуміють потреби та поведінку користувачів, вони зможуть створити інформаційну архітектуру (IA) для свого продукту чи сайту. Дизайнери використовують IA як візуальний план, що окреслює важливу навігацію, ієрархію вмісту, функції та взаємодії.

Каркаси та прототипи. Завдяки начерку IA дизайнери UX можуть почати перетворювати ідеї на реальні моделі, такі як каркаси та прототипи. Команди використовують ці докази концепції, щоб перевірити ідеї, визначити вимоги та встановити пріоритети функцій. Онлайн-прототипи Figma сприяють співпраці між дизайнерами, розробниками та власниками продуктів, об’єднуючи всіх разом для створення більш чутливого, доступного, корисного та привабливого кінцевого продукту.

Якість будь-якого інтерфейсу визначається наскільки ефективно взаємодіє користувач із системою. Незважаючи на наявність безлічі методологій та підходів у галузі дизайну інтерфейсів, існують певні актуальні проблеми:

1. Забезпечення відповідності інтерфейсу предметної області: початкова робота зі збору інформації для створення продукту може виявитися недостатньою і займати значну кількість часу, що призводить до збільшення тривалості та складності розробки. Крім того, аналітики та розробники не завжди є експертами в конкретній предметній області, тож інформація, зібрана на початку проекту, може втратити актуальність.

2. Відсутність упорядкованості та формалізації вимог до продукту: специфікації вимог для програмних продуктів повинні бути адекватними та однозначними, щоб кожна особа, що бере участь у розробці веб-додатку, розуміла їхнє значення та мету. Не всі користувачі завжди точно знають, що вони хочуть від продукту, іноді їм потрібно випробувати додаток, щоб зрозуміти свої потреби. Цю проблему можна вирішити за допомогою створення "живого" прототипу, який показує взаємодію елементів інтерфейсу в анімованому форматі.

3. Гнучкість інтерфейсів користувача: деякі інтерфейси дозволяють користувачам налаштовувати їх під свої потреби (додавати, видаляти функці її, змінювати порядок тощо), що сприяє покращенню користувацького досвіду. Проте, багато додатків обмежують можливість налаштувань, що може призвести до втрати актуальності. Наприклад, операційні системи часто надають користувачам можливість налаштовувати робочий стіл, додавати необхідні програми, але багато веб-додатків не надають таких можливостей, що знижує їхню привабливість для користувачів.

Розпочнемо з проектування інтерфейсу веб-додатку, який використовуватиметься адміністратором системи для створення та керування купонами. Цей інтерфейс має бути зручним, інтуітивним, та функціональним, щоб адміністратор міг ефективно виконувати свої обов'язки щодо керування купонами та забезпечення користувачів необхідними знижками. Інтерфейс для адміністратора повинен бути досить простим та інтуїтивним для того, щоб адміністратор міг без зайвих зусиль розбиратися в усіх функціях та можливостях панелі керування. Чітка структура та легкий доступ до основних опцій дозволять мінімізувати час, який адміністратор витрачає на виконання необхідних завдань. Інтерфейс адміністратора повинен включати всі необхідні функціональність та опції для керування купонами. Це включає в себе можливість створення, редагування, видалення купонів, а також встановлення різних параметрів, таких як термін дії, обмеження та умови використання. Дизайн інтерфейсу повинен відповідати загальному стилю та бренду веб-додатку. Розроблений веб-дизайн

головної сторінки містить все необхідне для адміністратора, зручний і інтуїтивний дизайн полегшує роботу з веб-додатком, розроблений шаблон дизайну інформаційної технології зображено на рисунку 2.6.

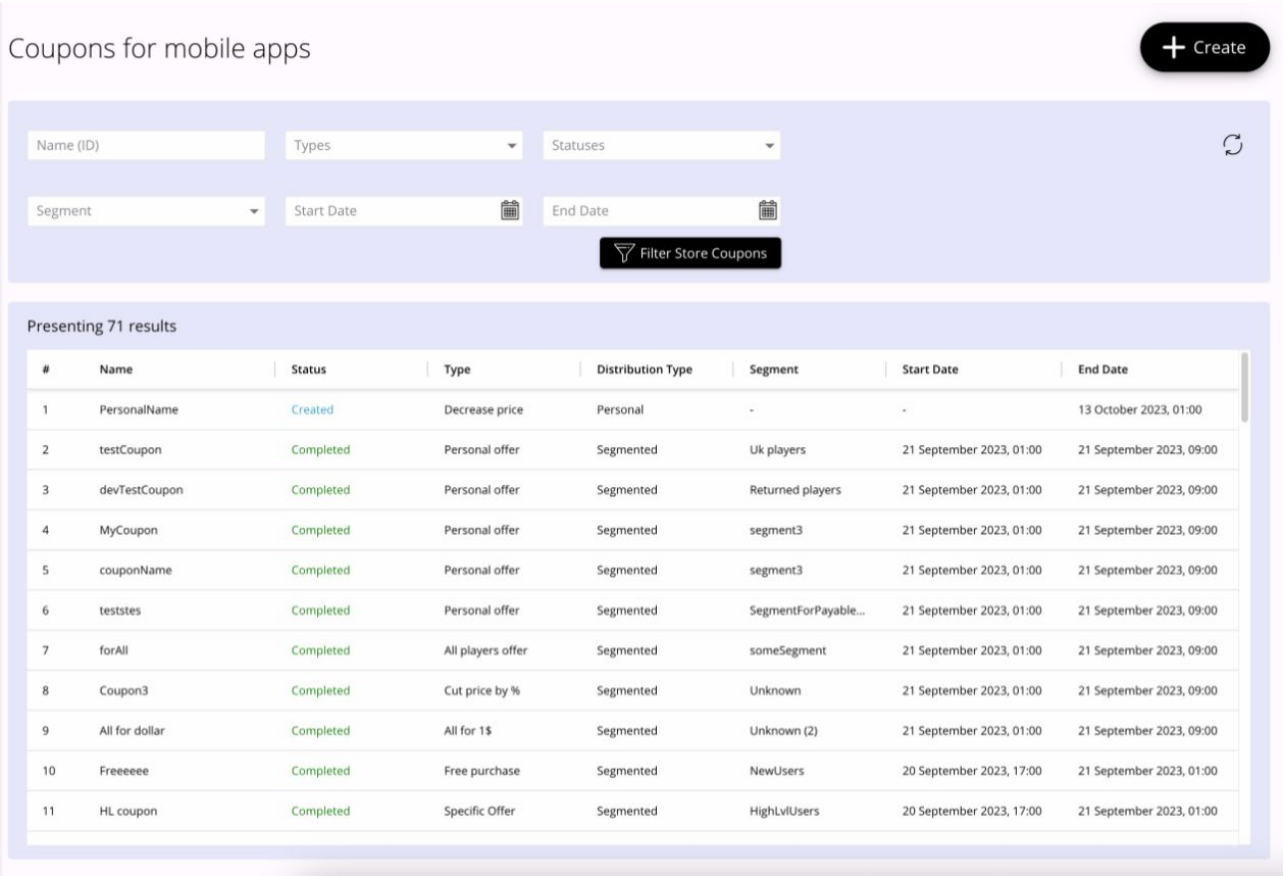

Рисунок 2.6 – Інтерфейс головної сторінки адміністратора

Отже, проектування якісного інтерфейсу користувача є складним завданням, яке вимагає ретельного аналізу предметної області, чітких та адекватних специфікацій вимог, а також розгляду гнучкості інтерфейсів для забезпечення максимального задоволення користувачів від взаємодії з системою

2.3 Розробка алгоритмів управління знижками та створенням купонів на товарні одиниці

Архітектура веб-додатків — це структура високого рівня, яка визначає спосіб роботи, ефективності та масштабування вашого продукту та бізнесу. У

наші дні на етапі вибору архітектури веб-додатків ви часто губитеся серед безлічі варіантів, доступних на ринку розробки програмного забезпечення. Чим більше з'являється нових імен і трендів, тим важче стає прийняти рішення. Isomorph, Progressive Web app, SPA чи SSR – яка архітектура сучасної веб-програми для вас найкраща та які критерії використовувати для оцінки?

Спочатку давайте визначимо, що таке веб-додаток. Це клієнт-серверна програма, де є браузер (клієнт) і веб-сервер. Логіка веб-додатку розподілена між сервером і клієнтом, є канал для обміну інформацією і сховище даних, розташоване локально або в хмарі.

Веб-додатки з'явилися на етапі еволюції веб-сайтів і, дійсно, мають багато спільного. Факторами, які відрізняють веб-сайт від веб-додатку, є інтерактивність, інтеграція та автентифікація, зображено на рисунку 2.7.

Сучасні веб-програми все ще використовують концепцію 3-рівневої архітектури, яка розділяє програми на рівень презентації, рівень програми та рівень даних.

У 3-рівневій архітектурі веб-додатків кожен рівень працює на власній інфраструктурі та може розроблятися паралельно різними командами [18]. Така структура дозволяє оновлювати та масштабувати кожен рівень за потреби, не впливаючи на інші рівні, зображено на рисунку 2.8.

Щоб допомогти вам зрозуміти, чи дійсно запропонований підхід до продукту відповідає потребам бізнесу, оцінимо сучасні типи архітектури веб-додатків відповідно до критеріїв, які вважаємо найважливішими для бізнесу, тобто продуктивність, інтерфейс користувача, пошукова оптимізація, можливість зв'язування та швидкість реалізації. на стороні розвитку.

Server Side Rendering (SSR). Говорячи про основні веб-принципи, ми зазвичай маємо на увазі архітектуру клієнт-сервер. Клієнт запитує вміст із сервера, де розташована бізнес-логіка та база даних. Використовуючи простий JavaScript, статична веб-сторінка надсилає запит до служби (можливо, API). Служба повертає дані та відображає клієнту HTML-сторінку.

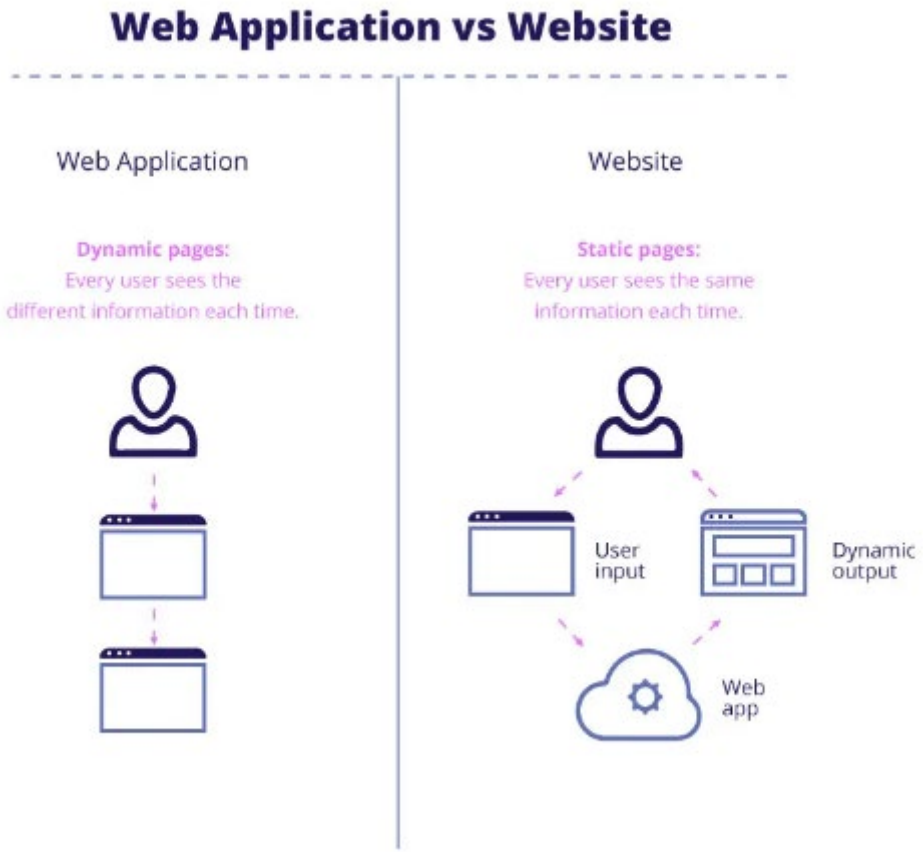

Рисунок 2.7 – Різниця між веб-сайтом та веб-додатком

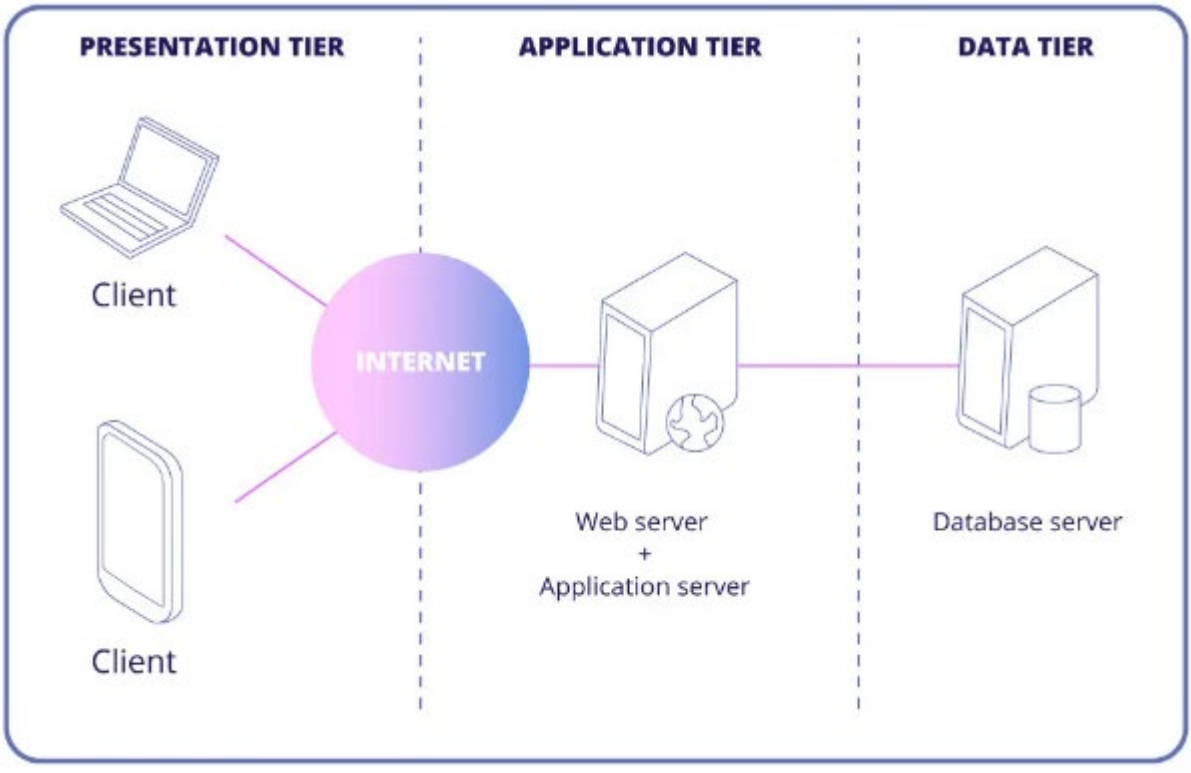

Рисунок 2.8 – 3-рівнева архітектура веб-додатків

Якщо програма відображається на стороні сервера, вміст отримується із сервера та передається в браузер для відображення користувачеві. Якщо сторінка HTML відображається на стороні сервера, користувач повинен перейти до сторінки, перш ніж браузер отримає сторінку з сервера. Це означає, що для відображення вмісту користувачеві потрібно більше часу. Для кешування вмісту сторінки ця схема часто постачається з Nginx, веб-сервером, який також можна використовувати як поштовий проксі та балансувальник навантаження.

Плюси мінуси. Той факт, що HTML відтворюється на сервері, забезпечує низку переваг, таких як SEO, можливість зв'язування та миттєве завантаження першої сторінки. Візуалізація на стороні сервера працює, коли JS вимкнено в браузері. Оскільки код обробляється на сервері, ніяких особливих вимог до браузера не висувається, це дозволяє миттєво виявляти помилки. Однак SSR не може обробляти важкі запити до сервера (повторювані HTML, CSS), що призводить до повільного рендерингу під час завантаження сервера або повного перезавантаження сторінки. Але справжньою ахіллесовою п'ятою цього базового типу архітектури веб-додатків є погана взаємодія з кінцевим користувачем і неможливість створити повноцінний інтерфейс користувача. Іншими словами, SSR є простим і економічно ефективним шляхом, якщо вам потрібно створити простий веб-сайт [18]. Реалізація цього типу архітектури можлива з будь-якою мовою програмування та серверною частиною, приклад архітектури зображено на рисунку 2.9.

Static Site Generation (SSG). У процесі генерації статичного сайту використовується генератор, який автоматизує кодування окремих HTML-сторінок, створюючи їх із шаблонів. Вибираючи статичну генерацію сайту, ви отримуєте простий статичний веб-сайт, розташований на CDN або будь-якому сервері, який містить уже згенеровану HTML-сторінку, яка буде надана користувачам за запитом. Тому не потрібно генерувати його щоразу, коли хтось відвідує веб-сайт – сервер просто надсилає вже наявні дані через API.

Плюси мінуси. Перш за все, цей підхід підходить лише для веб-сайтів. Крім того, вміст згенерованих сторінок веб-сайту не змінюється, якщо ви не додаєте

нові дані чи компоненти. Це означає, що вам доведеться повністю генерувати веб-сайт, як тільки ви захочете додати новий вміст. Це один із головних недоліків, який серйозно обмежує бізнес-випадки, до яких він застосовний.

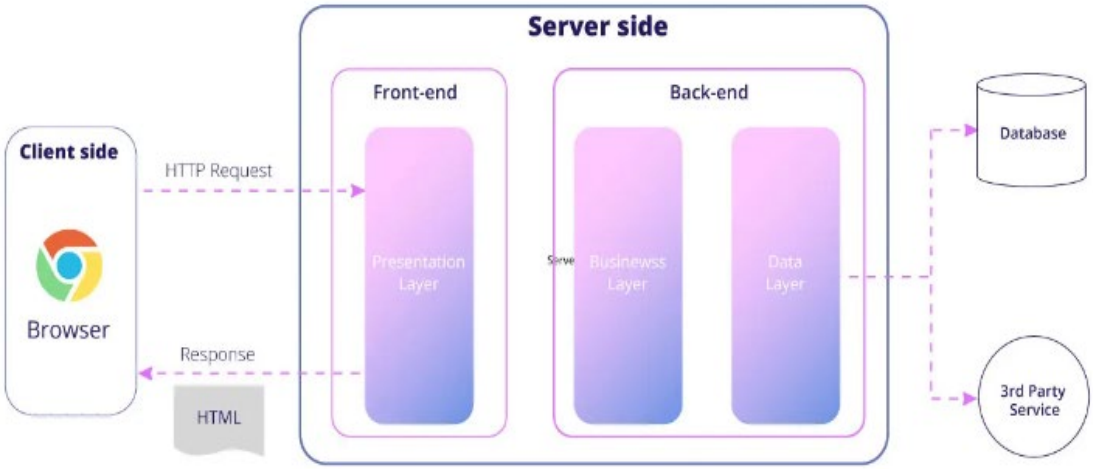

Рисунок 2.9 – Архітектура SSR додатку

Серед переваг, однак, висока швидкість статичного вмісту, який доставляється через CDN. Також у SSG усі серверні операції та робота з базою даних реалізуються через незалежний від сайту API. Цей варіант простий і тому виключно доступний для реалізації, приклад такої архітектури зображено на рисунку 2.10.

Single Page Application (SPA). SPA — це тип веб-програми, яка працює в браузері. Не потрібно перезавантажувати сторінку, коли потрібно відобразити нові дані. Цей тип архітектури веб-додатків широко використовується в нашому повсякденному житті: Facebook, Gmail, Google Maps, GitHub і Twitter – усі вони є односторінковими додатками [18].

Плюси мінуси. На відміну від SSR і SSG, SPA дозволяє створити інтерактивну веб-програму. Він використовує API для зв’язку із сервером. Ця архітектура хороша для легкого масштабування продукту. Крім того, якщо потрібен мобільний додаток, не потрібно докладати додаткових зусиль для

розробки API – мобільний додаток може використовувати той самий API, що й Web.

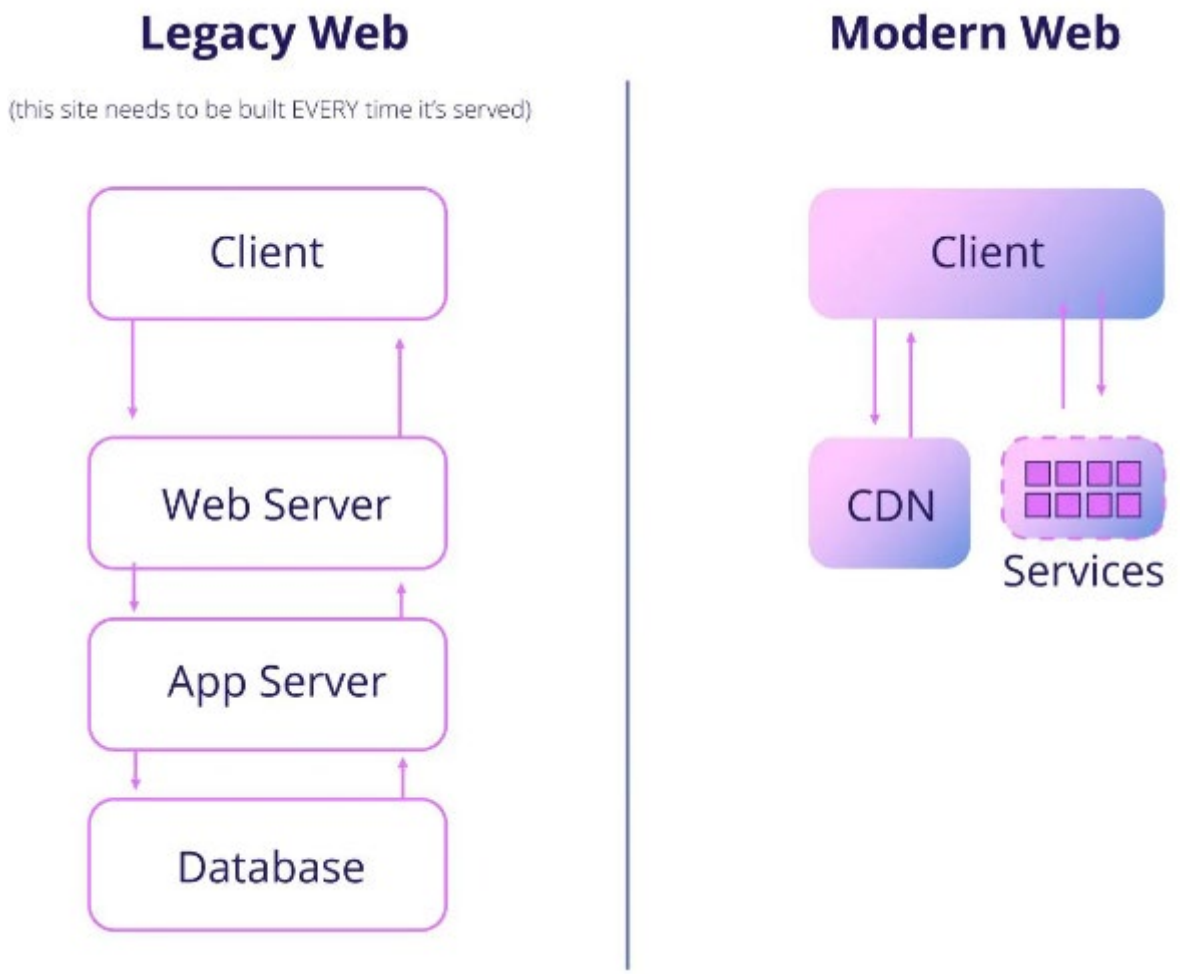

Рисунок 2.10 – Архітектура SSG додатку

SPA забезпечує швидку візуалізацію після повного завантаження програми у браузері та створює програмне забезпечення для кінцевого користувача з високою чутливістю. У той же час це «вбиває» ваше SEO і обмежує зв’язуваність, оскільки реалізація такого функціоналу потребуватиме додаткових зусиль. Серед інших недоліків – тривалий час, необхідний для першого завантаження, погана маршрутизація та обмежена підтримка застарілих браузерів. Будучи досить дорогим типом веб-архітектури, SPA підходить для створення адаптивного інтерфейсу користувача для користувачів B2C, приклад архітектури зображено на рисунку 2.11.

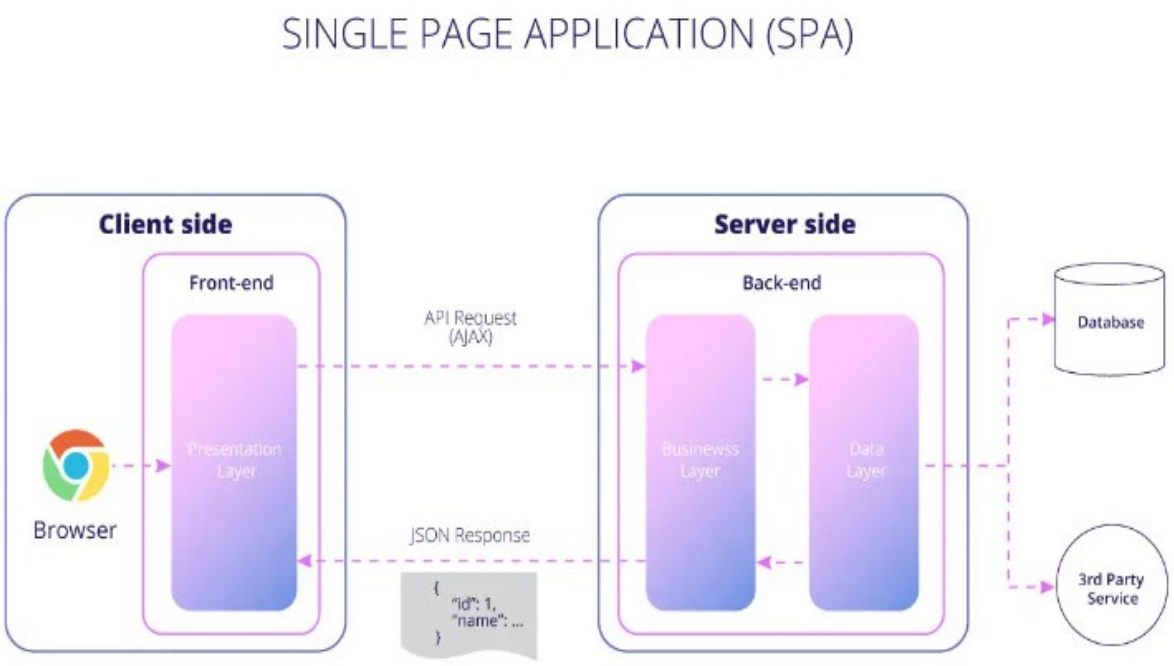

Рисунок 2.11 – Архітектура SPA додатку

Progressive Web App (PWA). Здається, останні кілька років усі говорять про розробку PWA. Що в них такого особливого і чи справді вони вирішують усі питання розробки веб-додатків?

Прогресивна архітектура веб-програми використовує логіку односторінкової веб-програми з деякими службами, які запускаються далі у браузері. Це означає, що основним моментом, який слід взяти до уваги, є те, що і браузер, і ОС повинні підтримувати цей набір стандартів.

Для кінцевого користувача прогресивна веб-програма фізично означає спливаюче вікно з пропозицією додати програму на екрані запуску (не в браузері, а на екрані вашої операційної системи), коли він відвідує веб-сайт. Якщо користувач погоджується, програма автоматично додається на пристрій.

Впровадження PWA дозволяє вашій веб-програмі підтримувати роботу в автономному режимі, фонову синхронізацію та push-сповіщення. Це відкриває доступ до функціональних можливостей, які раніше потребували рідної програми. При цьому, вибираючи для свого проекту архітектуру PWA, потрібно враховувати, що більшість функцій недоступні на iOS. Варто аналізувати кожен конкретний бізнес-кейс.

Плюси мінуси. Прогресивна архітектура веб-додатків підтримується Windows, Android та iOS (для iOS офлайн-режим вимкнено). Розробники можуть додавати оновлення до веб-програми віддалено. PWA є безпечним, оскільки використовує HTTPS. У той же час кінцеві користувачі можуть встановити PWA, навіть не відвідуючи Play Market або App Store. Серед недоліків цього типу архітектури – необхідність вибору браузера та ОС, які його повністю підтримують, приклад архітектури зображено на рисунку 2.12.

Такі параметри, як PWA, не підходять для стартапів. Це досить ефективно, якщо бізнес стабільний, а власник продукту знає, хто його кінцеві користувачі та який досвід вони очікують (наприклад, більшість із них користувачі Android). Хоча підтримка прогресивних веб-додатків не така широка. Нещодавно Firefox (який все ще займає 6,3% ринку США) припинив підтримку PWA», що доводить, що цей тип веб-архітектури все ще нестабільний.

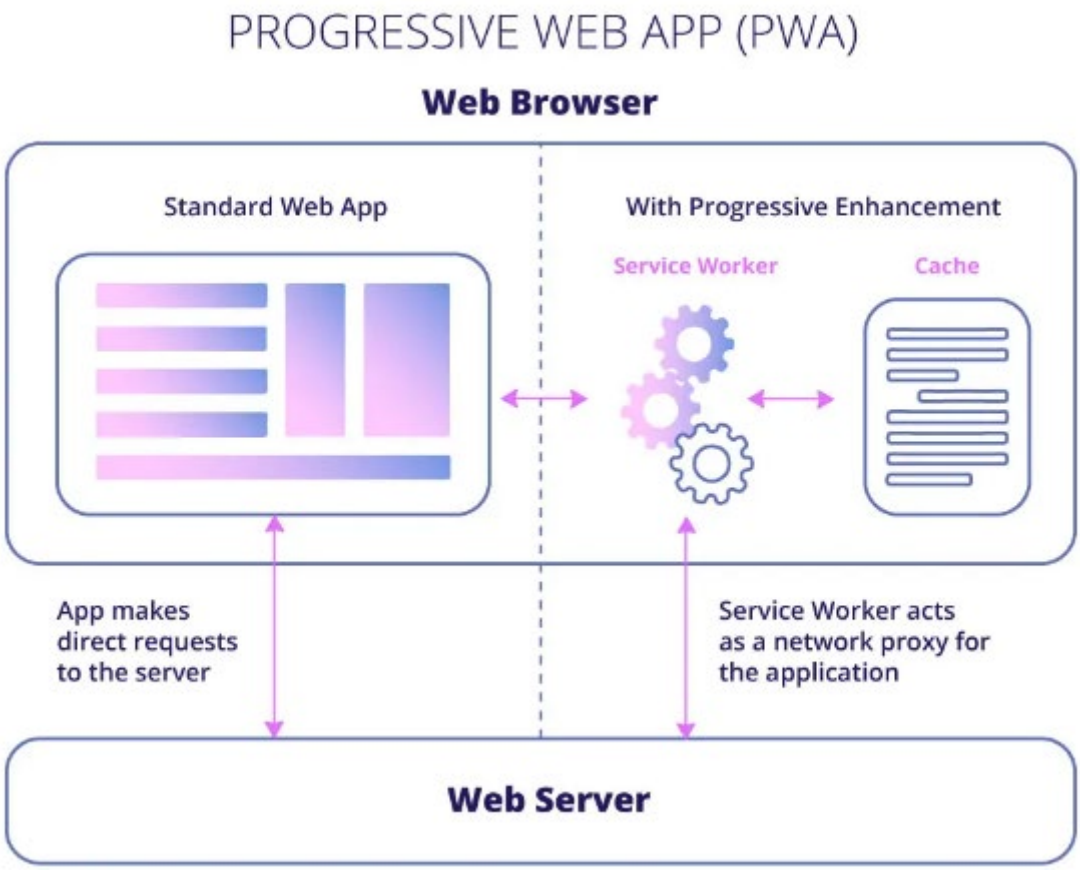

Рисунок 2.12 – Архітектура PWA додатку

Isomorphic. Ще одна сучасна архітектура веб-додатків називається ізоморфною. Це тип програми JavaScript, яка може працювати як на стороні клієнта, так і на стороні сервера. Спочатку клієнт завантажує HTML, де програма JavaScript завантажується у браузер, а потім програма починає працювати як SPA [18].

На відміну від SPA, тут перший рендер відбувається на сервері. Іншими словами, коли користувач вводить адресу веб-програми, сервер спочатку повністю відтворює HTML (пакети JS). Це дозволяє веб-сканерам Google перевіряти веб-сторінки та забезпечувати роботу пошукової оптимізації вашої веб-програми. У результаті компанія отримує вже відрендерену сторінку та пакети JS веб-додатку SPA.(рис.2.13).

Плюси мінуси. На відміну від рендерингу на стороні сервера, ізоморфна веб-архітектура забезпечує швидке оновлення даних, швидкість реагування та численні параметри UI/UX. Це забезпечує швидший рендеринг під час завантаження сервера, оскільки оброблений код передається клієнту. На відміну від візуалізації на стороні клієнта, це забезпечує миттєве відображення у веб-переглядачі, зручну маршрутизацію, пошукову оптимізацію та можливість зв’язування. Єдиним недоліком цього типу архітектури веб-додатків є те, що він повністю підтримується лише JavaScript. Найчастіше це означає, що технологічний стек для вибору обмежений фреймворками та інструментами на основі JS.

Micro Front-End. Серед інших принципів дизайну веб-додатків ми виділяємо мікроінтерфейс — підхід, який базується на декомпозиції зовнішнього додатка на окремі «мікропрограми», що працюють разом. Для кінцевого користувача всі вони розташовані на одній сторінці. Цей тип архітектури веб-програм є модульним, що означає, що сторінки та віджети є повністю незалежними програмами. При такому підході розробка та розгортання йдуть паралельно [18]. Але в той же час структура ускладнює ваш додаток і спричиняє дублювання коду, зображено на рисунку 2.14.

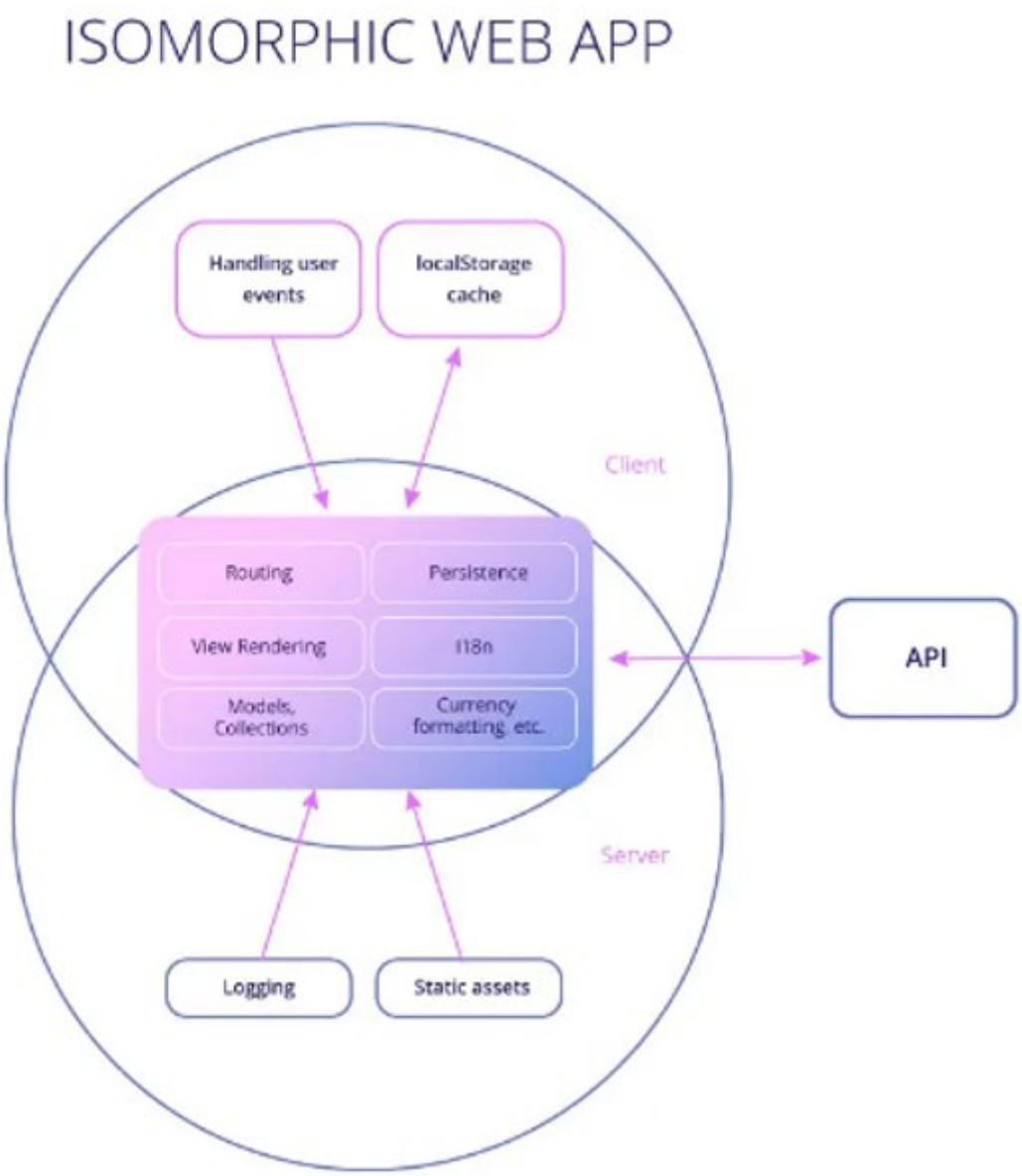

Рисунок 2.13 – Архітектура Isomorphic додатку

Беззаперечною перевагою архітектури мікроінтерфейсу є її масштабованість. Це підходить для корпоративних клієнтів, оскільки часто розробка масивних частин програмного забезпечення розподіляється між кількома командами. Мікросервіси відображаються як на задньому, так і на передньому плані, і їх можна масштабувати відповідно до ваших команд і навантаження, що працює в хмарі.

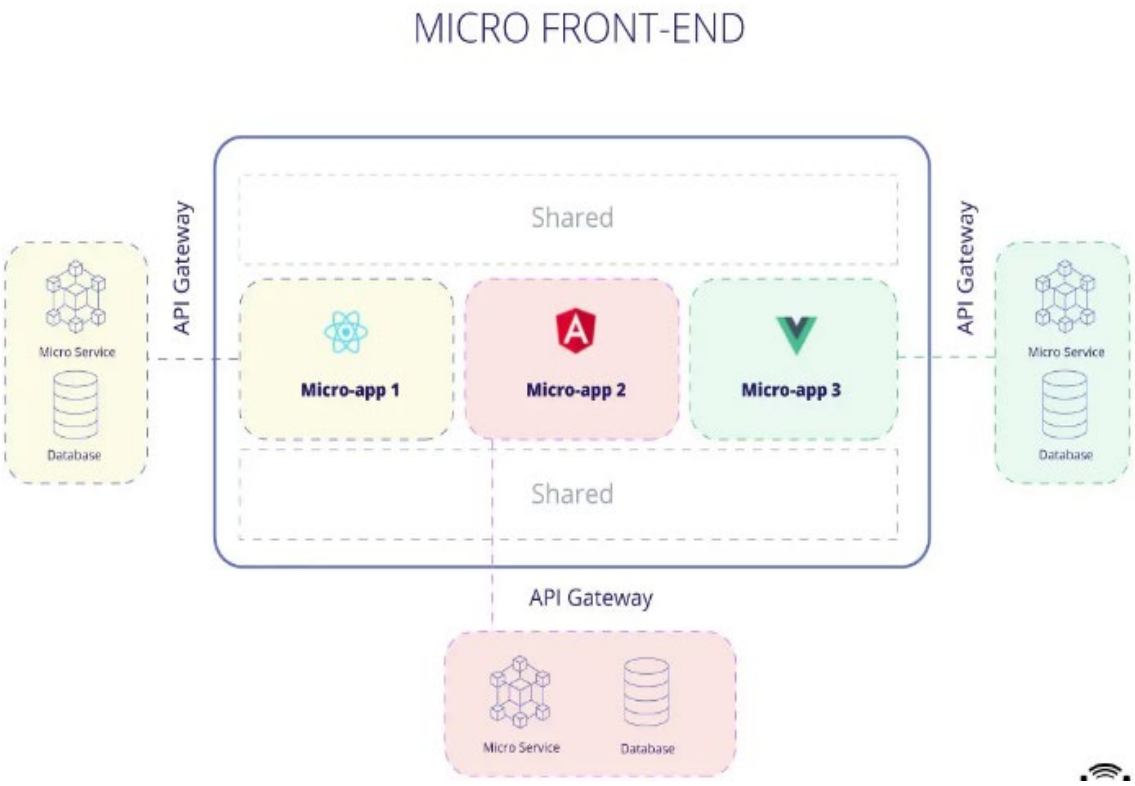

Рисунок 2.14 – Архітектура Micro front-end додатку

Node.JS and Web Front-End. Ця концепція була вперше сформульована в 2013 році Ніколасом С. Закасом і зараз використовується для складних веб-рішень.

«Відокремлення внутрішнього рівня інтерфейсу користувача від внутрішньої бізнес-логіки має сенс у більшій веб-архітектурі. Чому інженерам інтерфейсу має бути важливо, яка мова на стороні сервера потрібна для виконання критично важливих для бізнесу функцій? Чому це рішення має витікати на рівень внутрішнього інтерфейсу користувача? Потреби фронтенду принципово відрізняються від потреб бекенда», — так Закас пояснює головний принцип розробки цього типу веб-додатків [18].

Цей тип архітектури веб-додатків складається з сервера, створеного за допомогою Node.js, і рівня інтерфейсу користувача. Крім того, сервер бізнес-логіки може бути написаний будь-якою мовою (розглянемо PHP як приклад) і використовувати API для зв’язку з сервером.

Насправді новий підхід до веб-інтерфейсу дозволяє нам використовувати всі переваги типу ізоморфної архітектури, SSR або SPA, API для мобільних пристроїв і можливості зв’язування.

На відміну від ізоморфних веб-програм, немає обмежень щодо мови чи платформи бізнес-логіки.

Хоча слід зазначити, що розробка веб-програми, яка використовує нову логіку веб-інтерфейсу, займає більше часу, ніж розробка SSG або SSR. Щоб заощадити час на розробку, ми пропонуємо використовувати фреймворки Next.js і Nuxt.js.

Концепція нового веб-інтерфейсу підходить, коли йдеться про розробку ізоморфної веб-програми, де вже існує сервер API.

Приклад архітектури такого додатку зображено на рисунку 2.15

.

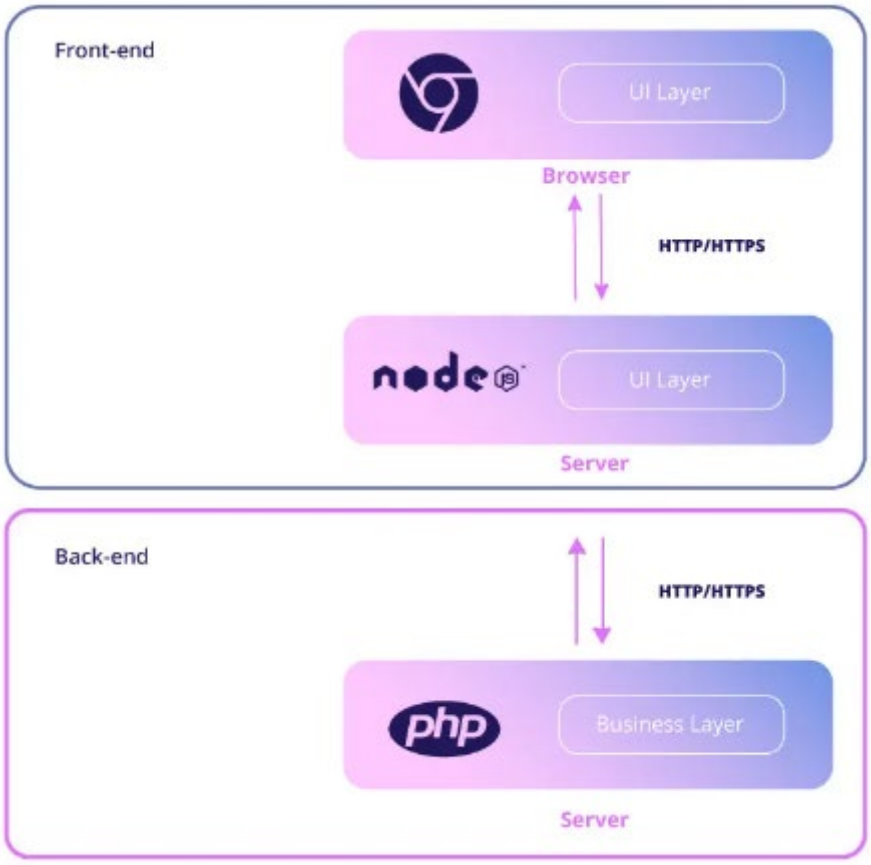

Рисунок 2.15 – Приклад архітектури веб додатку, з відокремленим сервером

Cloud Architecture. Говорячи про архітектуру веб-додатку, коротко розглянемо його серверну частину. Хмарна архітектура передбачає, що серверами керує хмарний провайдер, наприклад Amazon AWS, Azure або Google Cloud. Крім розміщення серверів на своїй стороні, ці провайдери пропонують комплекс послуг, які дозволяють створювати веб-додатки, розміщені та керовані в хмарі [18].

Дуже важливо вибрати правильну технологію для розробки хмарних веб-додатків. Популярним вибором є Python, який дозволяє автоматизувати завдання, керувати хмарними ресурсами та використовувати хмарні сервіси, які пропонують різні постачальники.

Безсерверні послуги AWS є одним із найпопулярніших хмарних рішень, які використовуються для впровадження таких популярних шаблонів, як мікросервіси, мобільні серверні програми та односторінкові програми. Наведена нижче схема дає розуміння того, як веб-сервіси AWS можна використовувати для створення веб-додатку за допомогою логіки 3-рівневої архітектури, яку ми пояснювали раніше, приклад архітектури зображено на рисунку 2.16.

Отже дослідивши різні можливі архітектури веб-додатків було обрано SPA архітектуру, яка надає всі необхідні можливості для створення та підтримки інформаційної технології.

Але не менш важливою частиною є розробка алгоритмів управління знижками та створенням купонів на товарні одиниці є важливим етапом у створенні інформаційної технології для управління знижками в веб-додатку. Ефективний алгоритм управління знижками дозволяє оптимізувати процес пропонування знижок і акцій, що полегшує користувачам доступ до вигідних пропозицій.

Для розробки алгоритмів управління знижками та створенням купонів необхідно враховувати кілька ключових аспектів:

1. Аналіз потреб користувачів: перш за все, важливо ретельно вивчити потреби та очікування цільової аудиторії. Які види знижок вони бажають отримувати? Які обмеження чи умови вони готові приймати для активації

знижок? Цей аналіз допоможе створити знижкові пропозиції, які справді привертають користувачів.

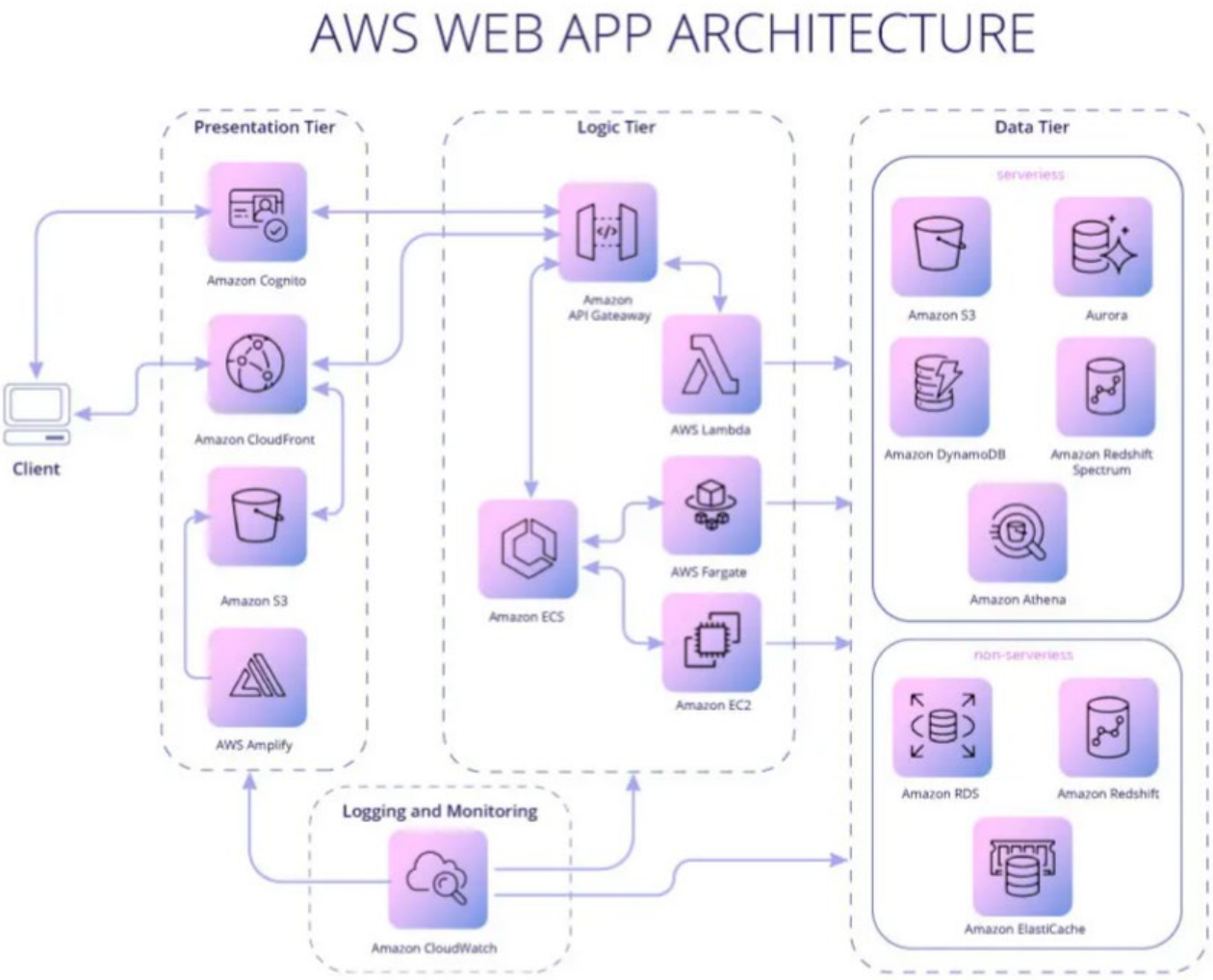

Рисунок 2.16 – Приклад архітектури додатку з використання AWS

2. Створення алгоритмів генерації купонів: визначення алгоритмів створення та розповсюдження купонів є ключовим. Кожен купон повинен бути унікальним, і його генерація повинна бути безпечною. Алгоритми мають враховувати фактори, такі як строк дії купона, обмеження на кількість використань, типи знижок тощо.

3. Управління станом знижок: знижки можуть мати різні стани, такі як активні, неактивні, завершені тощо. Розробка алгоритмів управління цими станами дозволяє ефективно контролювати процес використання знижок.

4. Взаємодія з базою даних: реалізація алгоритмів вимагає ефективної роботи з базою даних, де зберігаються інформація про знижки, купони та їх використання. Оптимізована робота з базою даних допомагає підвищити продуктивність системи.

Важливо враховувати, що розробка алгоритмів повинна бути гнучкою, оскільки потреби користувачів і умови можуть змінюватися. Також, безпека даних та конфіденційність користувачів мають бути високого рівня, оскільки це стосується особисто. Отже на основі цього розроблено алгоритм роботи користувацької частини інформаційної технології, на кожен запит, алгоритм буде перевіряти необхідні дані і в залежності від результату виконувати певні дії, що забезпечать надійність у роботі всієї системи, алгоритм зображено на рисунку В1 в Додатку В.

2.4 Висновок до розділу 2

У другому розділі було розглянуто ключові аспекти розробки інформаційної технології для управління знижками на товарні позиції. В даному розділі досліджено ключові особливості у розробці UI/UX, проаналізовано, які типи архітектур існують, їхні переваги та недоліки обрано загальну архітектуру розроблюваної інформаційної системи та визначено клієнтську частину, яку розроблятимемо з використанням React, Redux і TypeScript. Важливим аспектом є відзначення того, що розробка інформаційної технології для створення та управління знижками відбуватиметься у середовищі React. Ця технологія дозволить створити динамічний інтерфейс, що надасть адміністраторам зручну можливість управління всіма функціями веб-додатку.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ ЗНИЖКАМИ НА ТОВАРНІ ПОЗИЦІЇ

3.1 Обґрунтування вибору мови та середовища програмування для реалізації веб-додатку

JavaScript та TypeScript є основними мовами програмування, які використовуються у розробці веб-додатків. JavaScript - це мова програмування високого рівня, яка широко підтримується браузерами та використовується для створення динамічного веб-змісту. TypeScript, з іншого боку, є суперсетом JavaScript, який додає статичну типізацію, що допомагає виявляти та уникати потенційних помилок на етапі розробки. TypeScript забезпечує більшу стабільність та читабельність коду.

JavaScript - це високорівнева мова програмування, яка використовується для створення динамічних та інтерактивних веб-сайтів [19]. Вона має декілька ключових особливостей:

1. JavaScript є динамічною мовою, що означає, що типи змінних визначаються під час виконання програми. Це дає більшу гнучкість, але може також призвести до помилок на етапі виконання.

2. JavaScript виконується безпосередньо в браузері користувача, і він використовується для надання додаткової функціональності веб-сторінкам. Він може взаємодіяти з DOM (Document Object Model), змінювати вміст сторінок та обробляти події користувача.

3. JavaScript є однією з основних мов для розробки веб-додатків і має велику підтримку в браузерах. Це означає, що практично будь-який браузер може виконувати JavaScript-код.

Простіше кажучи, JavaScript є популярною мовою сценаріїв для додавання інтерактивних функцій та іншого динамічного веб-вмісту на веб-сторінки. Добре відомі приклади використання JavaScript включають заповнювані форми, слайд-шоу фотогалереї та анімаційну графіку.

JavaScript — це останній рівень функціональності на високоінтерактивних веб-сайтах. HTML забезпечує базову структуру сторінки. CSS — це модна особливість вашого веб-сайту — вона визначає стиль вашого сайту. Тоді JavaScript додає хвилювання. Вивчаючи JavaScript, важливо розуміти взаємозв'язок між HTML, CSS і JavaScript, а також те, як вони поєднуються у відображенні веб-сайту.

Відмінності між Java і JavaScript. По-перше, важливо зазначити, що Java і JavaScript не пов'язані, незважаючи на спільний термін «Java». І Java, і JavaScript є мовами для розробки веб-сторінок і веб-додатків. Однак вони мають чіткі відмінності.

Об'єктно-орієнтоване програмування: Java є об'єктно-орієнтованою мовою програмування. JavaScript — це об'єктно-орієнтована мова сценаріїв.

Синтаксис: синтаксис JavaScript не такий формальний або структурований, як Java. Таким чином, це простіше для більшості користувачів.

Компіляція: Java є скомпільованою мовою, тоді як JavaScript є інтерпретованою мовою, яка інтерпретується рядок за рядком під час виконання; скомпільовані мови, як правило, швидше, але інтерпретовані мови, як правило, більш гнучкі.

Середовище: ви можете використовувати Java-програми практично в будь-якому середовищі, запускаючи їх у віртуальних машинах або браузерах; JavaScript призначений лише для браузерів.

Використання пам'яті: Java потребує більше пам'яті, ніж JavaScript; це робить JavaScript кращим для веб-сторінок і веб-додатків.

Чи безпечний JavaScript? Хоча JavaScript широко поширений і використовується для веб-розробки, він має добре відомі вразливості. Однією з найпоширеніших кібератак, викликаних уразливістю JavaScript, є атака міжсайтового сценарію (XSS). Кіберзлочинці використовують XSS-атаки, щоб отримати доступ до ідентифікаційної інформації та викрасти її. Щоб мінімізувати експлойти, дуже важливо тестувати та переглядати свій код під час розробки. Такі методи тестування, як статичне й динамічне тестування безпеки

додатків (SAST і DAST), допомагають виявити вразливі місця на всіх етапах життєвого циклу розробки програмного забезпечення. За словами аналітиків безпеки Cloud Defense, SAST перевіряє ваш код на порушення правил безпеки та порівнює виявлені вразливості між вихідною та цільовою гілками. Ви отримаєте сповіщення, якщо на залежності вашого проекту впливають нещодавно виявлені вразливості.

JavaScript DOM. DOM, або об’єктна модель документа, діє як інтерфейс між мовою програмування, такою як JavaScript, і основним документом, зокрема документами HTML і XML. DOM – це стандарт W3C (World Wide Web Consortium), який визначається як «платформа та нейтральний щодо мови інтерфейс, який дозволяє програмам і сценаріям динамічно отримувати доступ до вмісту, структури та стилю документа та оновлювати їх». Документи складаються з набору окремих елементів і властивостей (тексту, кнопок, посилань тощо).

Основні компоненти JavaScript. Як і в інших мовах програмування, JavaScript використовує змінні для визначення місць зберігання даних. Змінні можуть бути глобальними (доступними для будь-якої функції в коді) або локальними, також відомими як блочні (доступні лише в блоці, де вони оголошені). Змінні можуть містити або фіксовані значення (константи, відомі як літерали), або змінні значення. JavaScript має особливий синтаксис для оголошення (створення) констант і змінних і присвоєння їм значень.

Однією з популярних інтегрованих середовищ розробки для веб-розробки є WebStorm. Це комерційний продукт, розроблений компанією JetBrains, і відомий своєю потужною функціональністю та інструментами для розробки веб-додатків. WebStorm підтримує мови JavaScript та TypeScript, а також інші технології, які використовуються в сучасних веб-додатках [20].

IDE надає ряд корисних функцій, таких як автодоповнення коду, відстеження помилок, відлагодження та підтримку систем контролю версій. Всі ці можливості роблять WebStorm відмінним вибором для розробки веб-додатків на JavaScript та TypeScript.

Інтелектуальний редактор коду WebStorm забезпечує функції автозаповнення, які передбачають, що ви намагаєтеся ввести, тим самим збільшуючи швидкість кодування. Крім того, він автоматично перевіряє ваш код на наявність помилок і надає швидкі виправлення, приклад зображено на рисунку 3.1.

```
// Example in JavaScript
var person = {
    name: "Alice",
    age: 25
};

console.log(person.na); // WebStorm will suggest 'name' as you type 'na'.
```

Рисунок 3.1 – Приклад коду, який буде перевірено

WebStorm інтегровано з такими популярними системами контролю версій, як Git, GitHub, SVN, Mercurial і Perforce. Це дозволяє розробникам виконувати різноманітні завдання, пов’язані з VCS, у IDE, спрощуючи співпрацю та керування версіями.

Вбудований дебаггер WebStorm дозволяє дебажити код JavaScript, запущений у Node.js або браузері. Ви можете встановлювати точки зупину, проходити через код і оцінювати вирази – і все це, не виходячи з IDE.

WebStorm інтегрується з такими популярними інструментами збирання, як Grunt, Gulp, Webpack, і менеджерами пакетів, такими як npm і yarn. Це забезпечує спрощений процес розробки, дозволяючи вам залишатися в IDE для більшості завдань.

Після встановлення WebStorm дозволяє налаштувати ваше середовище. Ви можете вибрати колірну тему, налаштувати розмір шрифту та вибрати плагіни, щоб покращити свій досвід розробки. Ринок WebStorm має безліч плагінів, починаючи від тем і закінчуючи інструментами продуктивності.

WebStorm – неймовірно потужний інструмент для веб-розробки. Він оптимізує процес кодування та підвищує продуктивність завдяки інтелектуальній допомозі в коді, вбудованому налагодженню та бездоганній інтеграції з системами контролю версій та інструментами збірки. Незалежно від того, чи ви тільки починаєте роботу, чи є досвідченим веб-розробником, WebStorm є важливим доповненням до вашого інструментарію.

Для ведення контролю над версіями файлів додатку та належної організації структури репозиторія, було використано систему контролю версій "GitHub." Git - це безкоштовна система управління версіями з відкритим вихідним кодом, яку створив Лінус Торвальдс у 2005 році. На відміну від попередніх систем управління версіями, таких як SVN і CVS, Git є розподіленою системою: кожен розробник має доступ до локальної копії повної історії репозиторію коду. Це означає, що початковий клонування репозиторію може займати більше часу, але наступні операції, такі як коміти, аналіз змін, об'єднання гілок та перегляд історії, виконуються набагато швидше та ефективніше [21].

За усіма файлами які додані слідкує інструмент під назвою git, файли можуть мати різні стани які зображено на рисунку 3.2.

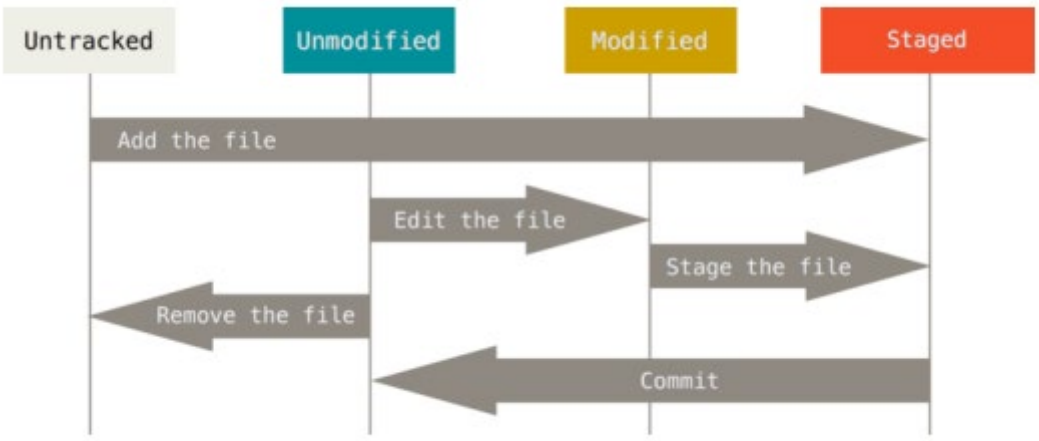

Рисунок 3.2 – цикл станів файлів у системі GIT

3.2 Розробка UML-діаграми активації знижки користувачем на товарну одиницю

У данній частині роботи розглянуто розробку UML-діаграми активації знижки на товарну одиницю. Ця діаграма є ключовим інструментом в аналізі та проектуванні системи управління знижками та створенням купонів для товарних позицій. Вона дозволяє відобразити послідовність операцій та взаємодію об'єктів у системі під час активації знижок.

UML (Unified Modeling Language) - це стандартизована мова для візуалізації, специфікації, конструювання та документування програмних систем. Діаграма активації (або діаграма послідовності) - це один із типів діаграм UML, який моделює послідовність виконання операцій та взаємодію об'єктів у системі. У контексті розробки системи управління знижками, діаграма активації допомагає відобразити, яким чином активується знижка на конкретну товарну позицію в системі.

На діаграмі активації можуть бути представлені такі основні елементи:

1. Об'єкти: це конкретні екземпляри класів (наприклад, користувачі, товарні позиції, купони тощо), які взаємодіють під час активації знижки.

2. Лінії: вони вказують послідовність виконання операцій. Лінії пов'язують об'єкти та вказують, хто і коли виконує певну операцію.

3. Повідомлення: це взаємодії між об'єктами, які відображаються у вигляді стрілок. Повідомлення описують операції, які виконуються об'єктами під час активації знижки.

4. Фрагменти: діаграма може містити фрагменти, такі як умовні оператори (if-else), цикли (for, while) тощо, які дозволяють моделювати альтернативні та повторювані варіанти взаємодії.

Розробка UML-діаграми активації для системи управління знижками допомагає зрозуміти, як проходить процес активації знижки на товарну одиницю, яка послідовність операцій та взаємодій об'єктів. Ця діаграма служить

основою для подальшої розробки та реалізації системи. UML-діаграми активації купона зображена на рис. 3.3.

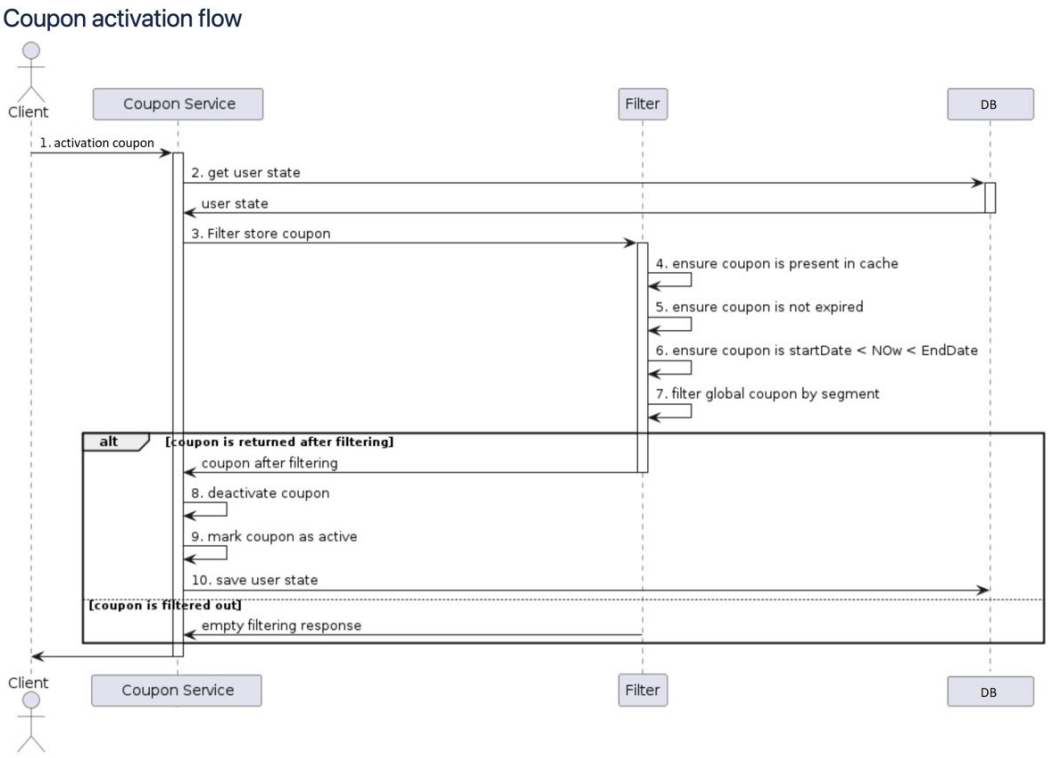

Рисунок 3.3 – UML-діаграма активації купона користувачем

3.3 Програмна реалізація веб-додатку управління знижками на товарні позиції

Для надійної роботи додатку було реалізовано певна кількість API. Вони є необхідними в реалізації роботи додатку з різною функціональністю для користувача. API були реалізовані за допомогою мови програмування JavaSript та програмної платформи для створення серверних додатків Node.js та разом використанням фреймворку Nest.js. Для комфортної взаємодії з користувачем було реалізовано функціонал, де користувач заходить в додаток працює з ним і при різних подіях надсилаються запити на сервер, де вони будуть опрацьовані і після результат буде повернено до користувача, також якщо запит містить якусь певну інформацію користувача він буде містити в собі також додаткову

інформацію потрібну додатку для розпізнання користувача і що саме він надсилає цей запит. На основі цього всього створено певні API:

1. ("/api/segments") – отримати список користувачів системи розбитих по сегментах;

2. ("/api/search") – отримати весь список знижок;

3. ("/api/create") – створити нову знижку;

4. ("/api/delete") – видалити знижку;

5. ("/api/upload") – для завантаження картинок і інших ассетів;

6. ("/api/status") – для зміни статусу знижки;

7. ("/api/view") – для перегляду знижки;

8. ("/api/edit") – для редагування знижки;

9. ("api/password-change") – змінити пароль доступу користувача;

10. ("api/search?*") – знайти конкретну знижку.

Приклад роботи з сервером зображено на рис. 3.4.

Доступ до певних API можуть мати лише додатки які було додано в білий список, аби лише розроблений додаток міг працювати з сервером повноцінно, отримувати основну інформацію, чи надсилати дані про користувача:

```
const whitelist = [
  "https://localhost:3001",
];
```

Для передачі даних використовується протокол HTTPS. Це поширений інтернет-протокол, при якому обмін даними між браузером користувача та сайтом відкритий. Комп'ютер користувача надсилає запит на сервер, а сервер видає результат – сторінку, форму реєстрації тощо.

Відвідувач, гортаючи сторінки сайту, вступає із сервером в обмін даними. Поки він просто читає і дивиться картинки, це відносно безпечно, хоча існує ймовірність, що сайт заражений вірусами, і, клацнувши щось шкідливе, ми отримаємо тривалі проблеми [22]. По-справжньому ризиковано без захисту там, де йде збирання особистих та фінансових даних. Якщо користувач реєструється на сайті, вводячи ім'я, телефон, номери кредитних карток, або якщо він платить

за покупки онлайн, зростає можливість фішингу. Це означає, що між браузером та сайтом вклиниться зловмисник та отримає інформацію, яка для нього не призначалася. Найчастіше люди про це вже щось чули, тому можуть відмовитись від покупки на сайті, маркованому як ненадійний.

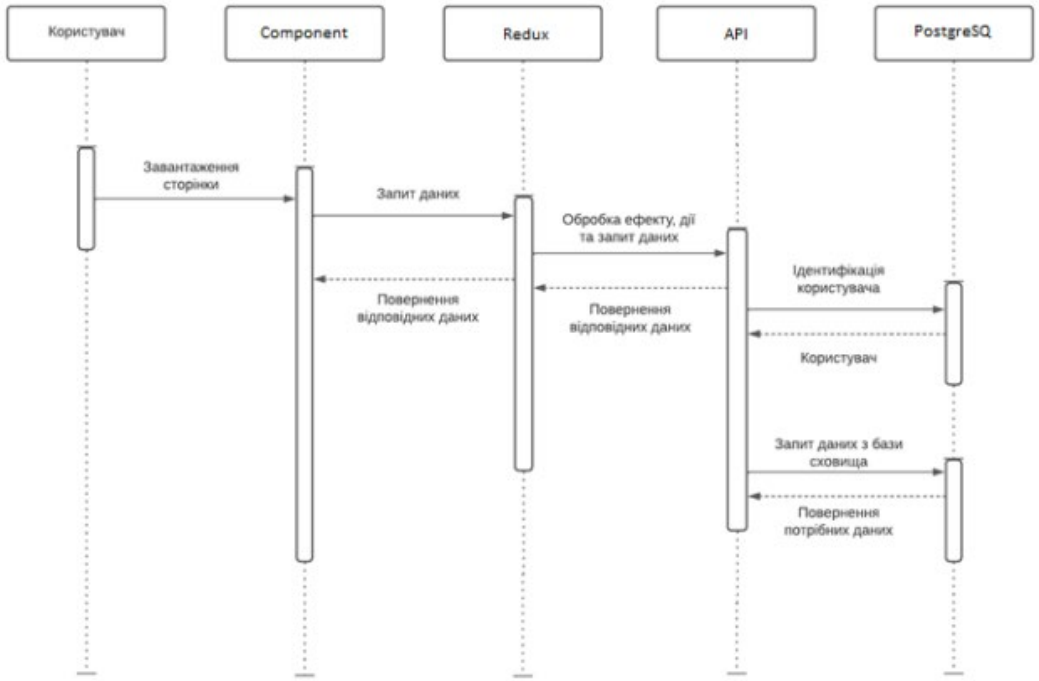

Рисунок 3.4 – UML діаграма роботи веб-додатку

Протокол https – це розширення звичайного http, яке дозволяє перевести взаємодію у закритий режим із шифруванням – найчастіше SSL. Зазвичай потрібно отримати сертифікат безпеки, встановити його та перекласти сайт на https. Це займає деякий час, але дозволяє убезпечити своїх клієнтів. Існують сертифікати різних видів. Одні дозволяють захистити лише сам сайт та невелику кількість піддоменів, інші – з опцією Wildcard – покривають велику кількість сайтів та дзеркал. Існують сертифікати з найпростішою перевіркою, яка підтверджує лише факт володіння доменом і займає до 15 хвилин. Інші варіанти передбачають розширену перевірку протягом кількох тижнів, підтвердження реального існування компанії та дають більш надійний захист.

Надалі при вході на сайт та обміні даними браузер користувача отримує підтвердження, що персональна інформація дійсно йде лише до власників конкретного сайту. Користувач цього моменту не помічає, все відбувається лише на рівні "браузер-сервер". Навіть найпростіші SSL-сертифікати, без яких захист обміну даними неможливий, дають можливість встановити на сайт статичну печатку довіри. Вона показує, що ресурс захищений. І хоча це лише візуальна позначка, вона дає відчуття безпеки.

Клієнтський додаток відправляє запит до API, при цьому він надає необхідні параметри для запиту та ідентифікує себе, потім сервер підтверджує або спростовує, що користувач дійсно існує і тоді йому надається необхідна інформація як успішна так і неуспішна. Прикладом запиту з додатку може бути наступний код програми, даний потрібний для отримання інформації про користувача:

```
const GetCoupon = (id, token) =>{
  const headers = new axios.headers({
    `Bearer ${token}`,
  });
  return axios.get(`${GET_COUPON_BY_ID}/${id}`, {
    headers,
  });
}
```

Для розробки було використані різноманітні інструменти та засоби, які підвищують надійність роботи серверу і клієнту.

Веб-інтерфейс додатку був написаний самостійно з використання певних бібліотек та інструментів за допомогою мови гіпертекстової розмітки та каскадних стилів, а також з використанням препроцессорів. Такий підхід до розробки надає можливість легкого створення стилів та перевикористання їх і легке та швидше написання стилів, які в кінцевому результаті створюють один файл з усіма стилями, а також допомагають вирішити проблеми з можливою колізією стилів.

Для того щоб зв’язувати разом компоненти додатку між собою і передавати дані між клієнтом і сервером який спілкується з базою даних та використовувались додаткові інструменти та бібліотеки.

Структура компоненту складається з двух основних файлів. Перший файл це основний скрипт для компоненту, він реалізує всі операції над даними що має виконувати даний компонент. Даний файл має характерний формат .js.

Приклад коду для компонента наведений нижче:

```
const Main = () => {
  return (
    <MainContainer>
      <Routers />
    </MainContainer>
  );
};
export default Main;

ReactDOM.render(
    <ProductProvider>
        <Router>
            <Main />
        </Router>
    </ProductProvider>,
    document.getElementById('root')
);
```

Інтерфейс додатку підтримує багато різних можливостей, які були додані в додаток, але є доволі важливими для користувача. На рисунку 3.5 зображено головний компонент веб-додатку який бачить адміністратор веб-сайту.

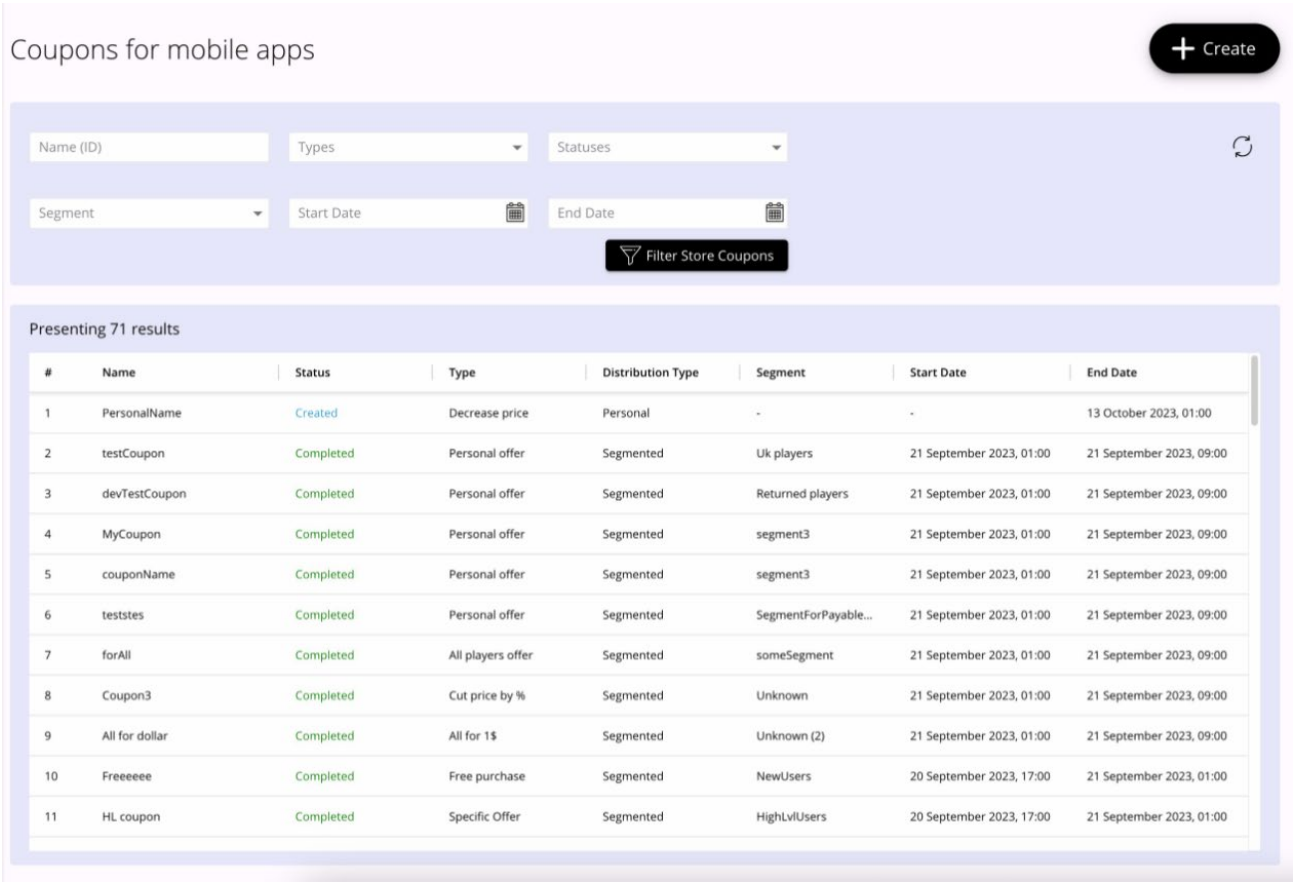

Рисунок 3.5 – Головний екран веб-додатку

На рисунку 3.6 зображено компонент створення нової знижки для користувача. Данний компонент містить в собі всі необхідні поля для опрацювання, це назва знижки або ж купона, яка далі буде опрацьовуватись адміністратором, або буде виведена користувачу мобільного додатку, тип знижки, тип може бути «безкоштовно», тобто для всіх користувачів або для певного списку можна зробити певний товар безкоштовний, або наприклад тип для зменшення ціни у відсотках, і тоді кінцева ціна буде зменшена на певний відсоток, після можна обрати чи купон буде персональний чи сегментованим, якщо він персональний, то застосується для певного користувача, а якщо сегментований, то для певної кількості користувачів, які входять в певний сегмент, який задається пізніше, також можна вказати кінцеву дату, при певних типах купону можна також обрати дату початку дії його, або ж просто вказати як довго він триватиме і обрати одиницю часу.

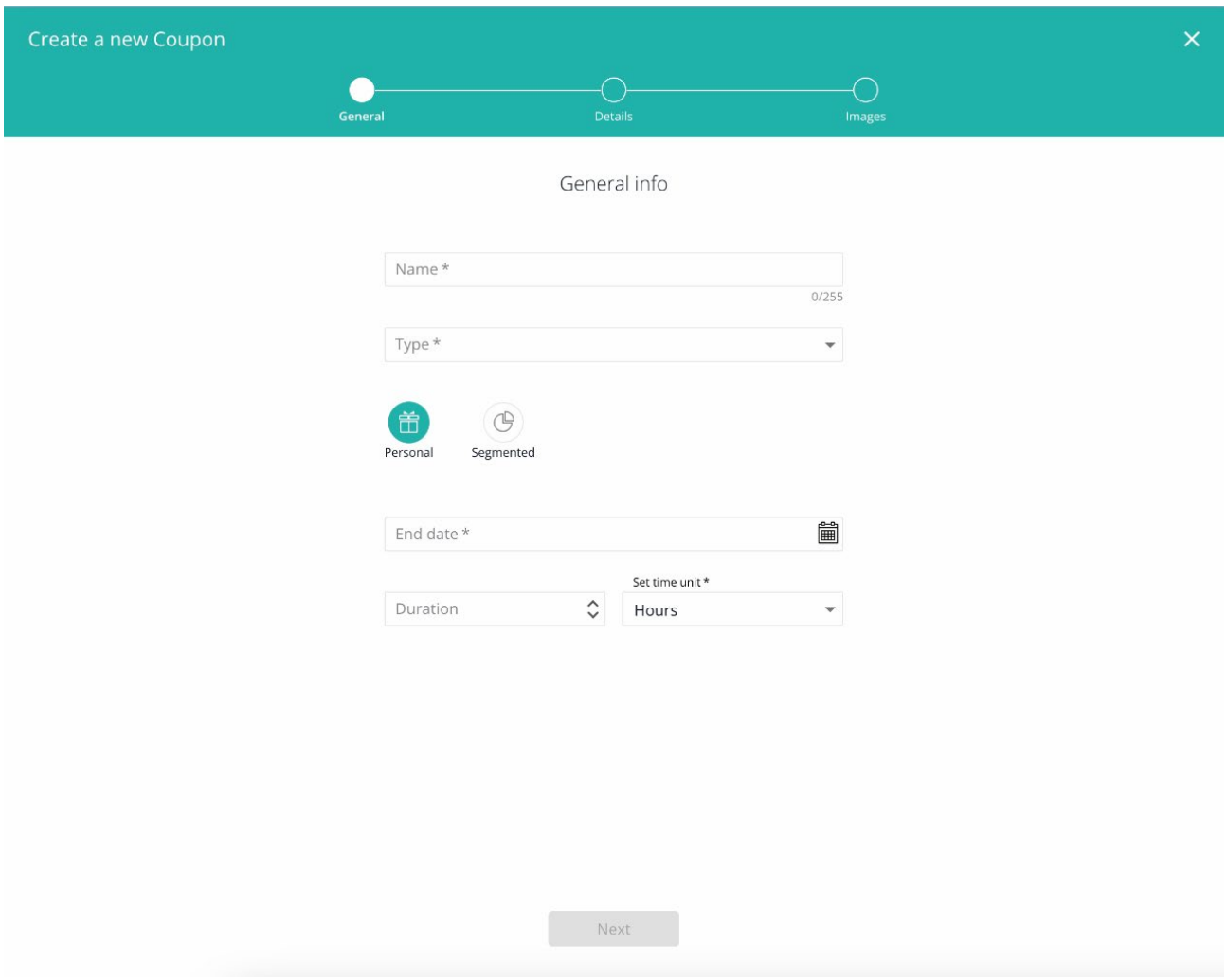

Рисунок 3.6 – Компонент створення нової знижки

На рисунку 3.7 зображено наступний крок створення купону, він містить назву типу купона, який було обрано на попередньому вікні, назву товарної одиниці для якої буде застосовано знижку, так як купон має тип фіксованої ціни, це означатиме, що товарна одиниця буде для всіх доступною з однією ціною, і в кінці також обирається сегмент користувачів які будуть мати змогу скористатися цією пропозицією. Окрім цього є ще додаткові поля, які відповідають за додаткову інформацію про знижку, як наприклад поле End date, яке відповдає за час коли знижка стане неактивною, або ж можливо вказати час, скільки годин, днів або тижнів буде діяти знижка на певний товар, завдяки цьому дуже легко конфігурувати різні параметри у одному вікні, без необхідності мати декілька вкладок браузера, тощо.

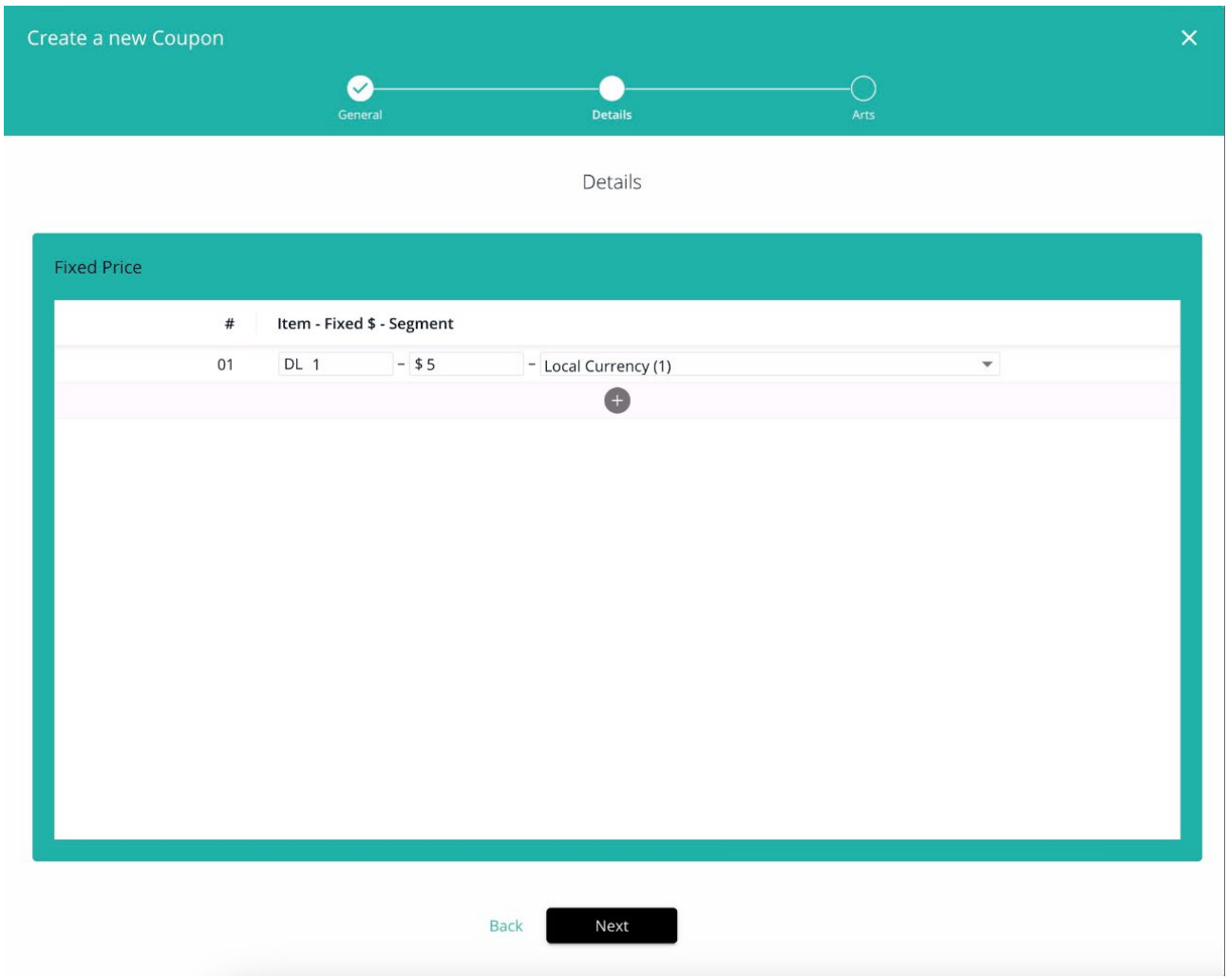

Рисунок 3.7 – Наступний крок створення знижки на товарну одиницю

І рисунку 3.8 зображено останній крок для створення знижки, це завантаження зображення, тобто воно може бути використано у мобільному додатку аби повідомити або показати користувачу, що є певна знижка і допомогти його направити на потрібний товар. Також на цьому вікні є певна додаткова інформація яка буде конфігуруватись користувачем при необхідності, як наприклад фіксована ціна у долларах, це поле може містити додаткову валідацію, як мінімальну ціну так і максимальну, щоб при помилці користувач не вказав занадто малу або високу ціну і додатково є можливість вказати певну вибірку користувачів для яких буде діяти знижка, цю вибірку можна створити самому задаючи певні параметри і налаштовуючи правила за якими користувачі будуть обиратись, або просто увімкнути, що знижка буде дійсною які всіх кінцевих користувачів.

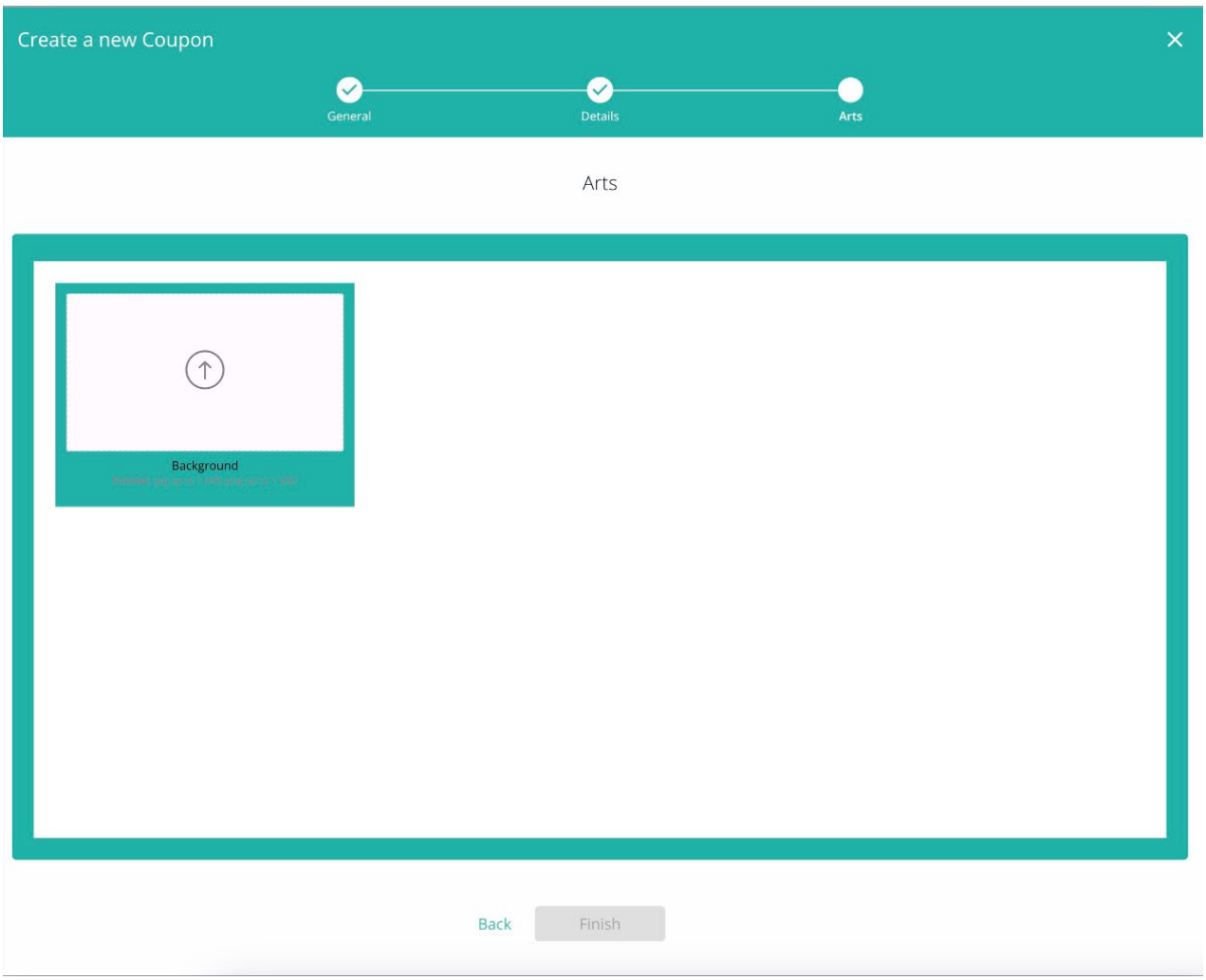

Рисунок 3.8 – Останній крок для створення знижки

В результаті буде створено новий купон або ж знижку на товарну одиницю, яку потім зможу використовувати системи для інтегровані з цим веб-додатком у своїх мобільний додатках, наприклад: магазинах, іграх, тощо.

Новий запис про знижку після створення буде відображено на головному екрані веб-додатку для управління знижками, і буде містити коротку інформацію про себе, що зображено на рисунку 3.9.

На рисунку можна побачити назву яку було задано для знижки, її статус «Created» так як вона лише створена, і для активації її потрібно перевести в наступний статус, відредагувати при наявності помилок або видалити. Тип самої знижки, для яких користувачів вона створена і її кінцеву дату дії.

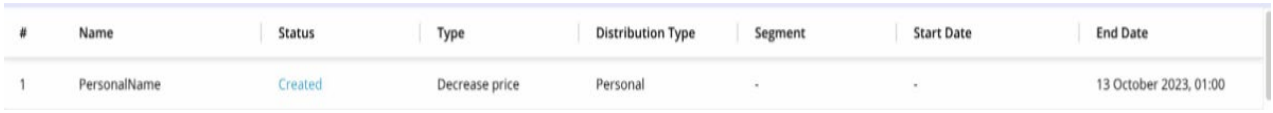

Рисунок 3.9 – Створена знижка

Отже у цій частині було детально розглянуто архітектуру додатку, реалізацію його компонентів, створено користувацький інтерфейс веб-додатку та описані особливості взаємодії з сервером.

3.4 Порівняння інформаційної технології для управління знижками на товарні позиції для мобільних додатків з аналогами

У сучасному цифровому бізнесі конкуренція на ринку мобільних додатків надто висока, і ефективне управління знижками стає ключовим елементом для залучення та утримання клієнтів. Декілька інформаційних технологій пропонують різноманітні рішення для цієї задачі, і їх порівняльний аналіз допомагає визначити найкращий підхід для досягнення бізнес-цілей, порівняння з існуючими аналогами зображено у таблиці 3.1.

Таблиця 3.1 – Порівняння з існуючими аналогами

Назва	Плюси	Недоліки
1	2	3
Shopify Discount	1. Інтегрована з системою управління продажами Shopify. 2. Гнучка система налаштувань знижок. 3. Підтримка різних типів знижок та промокодів.	1. Високі витрати на користування для невеликих бізнесів. 2. Низька кількість типів знижок.

Продовження табл. 3.1

1	2	3
WooCommerce Discounts	1. Широкий спектр додаткових плагінів для розширення функціоналу. 2. Безкоштовна основна версія для користувачів WordPress.	1. Складніше налаштування порівняно з нашим веб-додатком 2. Залежність від сторонніх розширень для деяких функцій.
Magento Discounts	1. Висока масштабованість для великих електронних комерцій. 2. Розширені можливості кастомізації	1. Складність встановлення та підтримки. 2. Вимагає додаткових ресурсів для ефективної роботи.
Розроблена інформаційна технологія	1. Великий список готових типів знижок 2. Гнучкість у користуванні і налаштуванні 3. Зручний і зрозумілий веб-інтерфейс 4. Можливість розширення функціоналу	1. Необхідність інтегруватись з додатком, який буде використовувати функціонал.

Кожна інформаційна технологія має свої переваги та недоліки, і вибір залежить від конкретних потреб бізнесу. Shopify вирізняється своєю інтегрованістю та зручністю, в той час, як розроблена система, може бути

вигідним варіантом для користувачів, і надає більше можливостей для кастомізації та легке внесення доробок та розширення існуючого функціоналу.

Але всеодно при виборі потрібно врахувати розмір бізнесу, бюджету та рівень технічної експертизи. Це порівняння слугує основним рішенням щодо вибору технології для управління знижками. Після аналізу різних інформаційних технологій для управління знижками на товарні позиції в мобільних додатках, можна виділити основні недоліки кожної системи.

Кожну проблему можна вирішити і зробити інформаційну технологію, яка не матиме недоліків інших систем.

При проблемі високих витрат на користування для невеликих бізнесів рішенням для цього може бути впровадження модульної системи тарифів для врахування потреб різних розмірів бізнесів. Введення базового безкоштовного плану для початківців.

При проблемі залежності від сторонніх розширень для деяких функцій, може бути вирішено за допомогою створення інтуїтивного інтерфейсу для спрощення процесу налаштування знижок. Розширення власних можливостей для уникнення сторонніх розширень.

Якщо ж проблема стосується складності встановлення та підтримки, то на цей випадок може бути рішення у вигляді розробки інструкцій та онлайн-ресурсів для полегшення процесу встановлення та налаштування. Оптимізація кодової бази для зменшення навантаження на ресурси.

Загальні переваги нашої технології це легкість встановлення та використання для різних розмірів бізнесу, максимально інтуїтивний інтерфейс для адміністраторів, оптимізована робота з мінімальними витратами на обслуговування, гнучкі налаштування для врахування унікальних потреб кожного бізнесу, постійне оновлення та оптимізація на основі зворотного зв'язку користувачів.

Цей підхід до оптимізації інформаційної технології для управління знижками спрямований на створення належного балансу між функціональністю,

вартістю та зручністю використання для різних бізнесів, щоб надати максимальну вартість клієнтам та користувачам.

3.5 Тестування веб-додатку для управління знижками на товарні позиції для мобільних додатків

Тестування – процес дослідження з метою отримання інформації про якість продукту. Процес перевірки відповідності заявлених до продукту вимог та реально реалізованої функціональності, що здійснюється шляхом спостереження за його роботою у штучно створених ситуаціях та на обмеженому наборі тестів, обраних певним чином.

Тоді як баг – це різниця між тим що було очікувано і що отримано. Для тестування створеного додатку були написані три типи тестів: unit тести, Інтеграційні тести, E2E тести.

Unit-тестування (модульне тестування) – процес у програмуванні, що дозволяє перевірити на коректність окремі модулі вихідного коду програми, функції чи методи.

Інтеграційне тестування – це тип тестування, у якому програмні модулі поєднуються логічно і тестуються як група. Як правило, програмний продукт складається з кількох програмних модулів, написаних різними програмістами. Метою нашого тестування є виявлення багів при взаємодії між цими програмними модулями.

End-to-end тестування це процес тестування програми на всіх рівнях – починаючи з фронтенду та закінчуючи бекендом, включаючи інтерфейс та кінцеві точки. Виконання End-to-end тестування гарантує, що програма перевірена на основі користувацьких сценаріїв, які допомагають контролювати та уникати ризиків. Ці види тестів утворюють певну піраміду тестування, що зображено на рисунку 3.10.

React-Testing-Library – це легке рішення для тестування веб-сторінок шляхом запитів і взаємодії з вузлами DOM (незалежно від того, моделюється за

допомогою JSDOM/Jest або в браузері). Основні утиліти, які він надає, включають запити DOM щодо вузлів способом, подібним до того, як користувач знаходить елементи на сторінці. Таким чином бібліотека допомагає переконатися, що ваші тести дають вам впевненість у тому, що ваша програма працюватиме, коли її використовує реальний користувач [23].

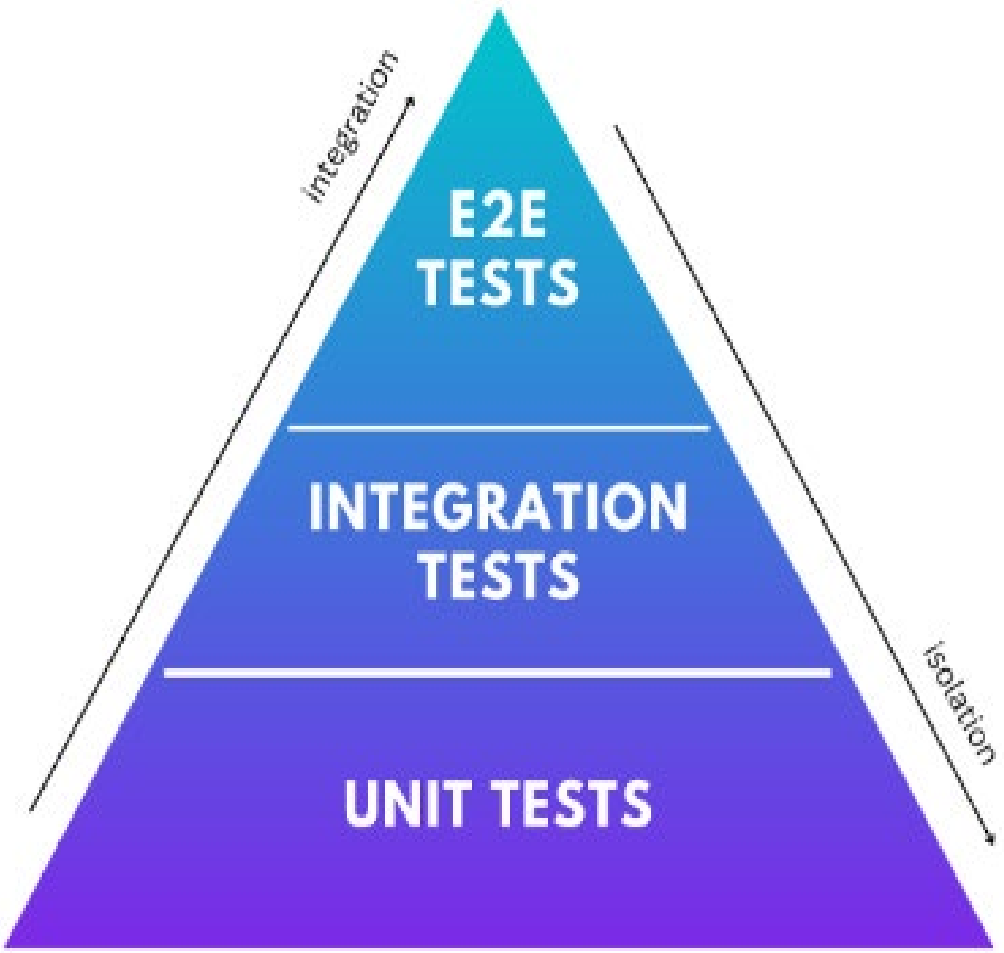

Рисунок 3.10 – Піраміда тестування

Отримання надійного тестового покриття є обов’язковим для надійності веб-програми. Jest – програма для виконання тестів JavaScript, яка надає ресурси для написання та виконання тестів. Бібліотека React Testing Library пропонує набір помічників для тестування, які структурують ваші тести на основі взаємодії користувача, а не деталей реалізації компонентів.

Як Jest, так і React Testing Library постачаються попередньо запакованим із додатком Create React App і дотримуються керівного принципу, згідно з яким програми для тестування повинні нагадувати те, як програмне забезпечення буде використовуватися.

Основна ідея впровадження тестування React у вашій програмі полягає в тому, щоб переконатися, що вона працює належним чином. Однак ширша ідея проведення тестів вашого додатка React полягає в тому, щоб зробити його менш схильним до помилок і забезпечити хорошу взаємодію з користувачем.

Крім того, програми зазвичай часто оновлюються; тестування гарантує, що програма не зламається, а код залишається надійним.

Основна бібліотека була загорнута, щоб забезпечити ергономічні API для кількох фреймворків, включаючи React, Angular і Vue. Також є плагін для використання запитів бібліотеки тестування для наскрізних тестів у Cypress і реалізація для React Native.

Модульні тести проводяться виключно для перевірки окремого компонента без втручання будь-якого іншого компонента. Тому кожен компонент, який ви проводите для модульних тестів, має бути незалежним і не покладатися на будь-який інший компонент або мати будь-яку залежність.

Щоб блок-тест залишався зосередженим, ви повинні тестувати одну річ за раз і перевіряти певну поведінку компонента.

Ви повинні витратити деякий час на написання назв тесту, щоб виявити та виправити проблеми, якщо тест не вдасться. Ви повинні використовувати описові назви, які вказують, що ви тестуєте в компоненті та який має бути очікуваний результат.

Використовуйте конвеєри CI/CD для автоматизації тестів, гарантуючи постійну роботу програмного забезпечення.

Приклад написання тесту наведено на рисунку 3.11.

```
// __tests__/fetch.test.js
import React from 'react'
import {rest} from 'msw'
import {setupServer} from 'msw/node'
import {render, fireEvent, waitFor, screen} from '@testing-library/react'
import '@testing-library/jest-dom'
import Fetch from '../fetch'

const server = setupServer(
  rest.get('/greeting', (req, res, ctx) => {
    return res(ctx.json({greeting: 'hello there'}))
  }),
)

beforeAll(() => server.listen())
afterEach(() => server.resetHandlers())
afterAll(() => server.close())

test('loads and displays greeting', async () => {
  render(<Fetch url="/greeting" />)

  fireEvent.click(screen.getByText('Load Greeting'))

  await waitFor(() => screen.getByRole('heading'))

  expect(screen.getByRole('heading')).toHaveTextContent('hello there')
  expect(screen.getByRole('button')).toBeDisabled()
})

test('handles server error', async () => {
  server.use(
    rest.get('/greeting', (req, res, ctx) => {
      return res(ctx.status(500))
    }),
  )

  render(<Fetch url="/greeting" />)

  fireEvent.click(screen.getByText('Load Greeting'))

  await waitFor(() => screen.getByRole('alert'))

  expect(screen.getByRole('alert')).toHaveTextContent('Oops, failed to fetch!')
  expect(screen.getByRole('button')).not.toBeDisabled()
})
```

Рисунок 3.11 – Простий тест написаний за допомогою RTL

Якщо потрібно створити mock функцію чи щось інше для тесту це можна легко створити за допомогою можливостей RTL як наведено на рисунку 3.12.

```
// declare which API requests to mock
const server = setupServer(
  // capture "GET /greeting" requests
  rest.get('/greeting', (req, res, ctx) => {
    // respond using a mocked JSON body
    return res(ctx.json({greeting: 'hello there'}))
  }),
)

// establish API mocking before all tests
beforeAll(() => server.listen())
// reset any request handlers that are declared as a part of our tests
// (i.e. for testing one-time error scenarios)
afterEach(() => server.resetHandlers())
// clean up once the tests are done
afterAll(() => server.close())

// ...

test('handles server error', async () => {
  server.use(
    // override the initial "GET /greeting" request handler
    // to return a 500 Server Error
    rest.get('/greeting', (req, res, ctx) => {
      return res(ctx.status(500))
    }),
  )

  // ...
})
```

Рисунок 3.12 - Використання mock функцій для тесту

Для проекту були створені unit-test для тестування атомарної функціональності додатку окремо, в штучно створеному середовищі. приклад наведено на рисунку 3.13.

```
describe( name: 'List',  fn: function () : void  {
    let wrapper: ReactWrapper;

    afterAll( fn: () => wrapper.unmount());

    it( name: 'should render component',  fn: function () : void  {
        prepareTestStoreCouponsPageResult(store);
        prepareTestOperators(store);

        wrapper = mountWithTheme(
            <Provider store={store}>
                <List/>
            </Provider>,
        );

        expect(wrapper.text()).toBe( expected: 'Presenting 3 resultsImport');
    });
});
```

Рисунок 3.13 – Один із unit тестів створених для веб-додатку

Всього для покриття всіх створених частин веб-додатку було створено декілька сотень тестів, які покривають всіх необхідні модулі роботи веб-додатку, інтеграцію з сервером, коректне відображення данних тощо, результат тестування наведено на рисунку 3.14.

```
Test Suites: 72 passed, 72 total
Tests:       396 passed, 396 total
Snapshots:   72 passed, 72 total
Time:        42.322 s
Ran all test suites related to changed files.
```

Рисунок 3.14 – Успішне проходження всіх тестів веб-додатку, та перевірки на правильне відображення необхідних данних

3.6 Висновок до розділу 3

У цьому розділі було представлено детальну інформацію щодо програмної реалізації інформаційної технології для управління знижками на товарні позиції у веб-додатку. Було розглянуто вибір мови та середовища програмування, такі як JavaScript і TypeScript, і визначено їх важливість для розробки сучасного веб-додатку. Докладно описано модель активації знижок, що використовується в системі та надає загальний вигляд на процес взаємодії адміністраторів із знижками на товарні позиції. Було розглянуто важливі компоненти реалізації, такі як використання React та Redux, взаємодія з сервером через RESTful API, використання бази даних, а також важливість вибору WebStorm IDE для зручної розробки. Створено зручний та сучасний користувацький інтерфейс та показані основні компоненти для роботи з додатком. Проведено процес тестування створеної системи, що включає в себе верифікацію функціональності та стабільності додатку. Усі ці аспекти разом створюють комплексний вигляд на розробку та реалізацію інформаційної технології управління знижками на товарні позиції, підкреслюючи важливість правильного вибору технологій, грамотного програмування та тестування для створення надійного та ефективного веб-додатку.

Тобто мета магістерської кваліфікаційної роботи досягнута – створено веб-додаток для управління знижками на товарні одиниці для мобільних додатків, адміністратор має доступ до всіх необхідних функцій, що актуальні і потрібні для управління однією із складових екосистем додатків, або ж для роботи з мобільними додатками які інтегровані з системою.

4 ЕКОНОМІЧНА ЧАСТИНА

Для успішного впровадження науково-технічної розробки важливо, щоб вона відповідала сучасним вимогам науково-технічного прогресу та ураховувала економічні аспекти. Оцінка економічної ефективності результатів науково-дослідної роботи є ключовою частиною цього процесу. Дослідження, яке представлене в магістерській роботі та присвячене розробці та аналізу "Інформаційної технології управління знижками на товарні позиції для мобільних додатків", відноситься до науково-технічних робіт, спрямованих на виведення на ринок. Виведення на ринок може бути визначене під час виконання самої роботи і розглядається як комерціалізація науково-технічної розробки. Цей напрямок розглядається як пріоритетний, оскільки отримані результати можуть бути корисними для різних зацікавлених сторін, приносячи економічні вигоди. Однак для успішної реалізації цього процесу важливо здійснити пошук зацікавленого інвестора, який виявить інтерес до втілення даного проекту, та переконати його в обґрунтованості вкладання інвестицій у цю розробку. З цією метою були визначені такі етапи виконання робіт:

1. Проведення комерційного аудиту науково-технічної розробки, включаючи визначення науково-технічного рівня та комерційного потенціалу.
2. Розрахунок витрат на реалізацію науково-технічної розробки.
3. Проведення розрахунку економічної ефективності впровадження та комерціалізації науково-технічної розробки для потенційного інвестора, а також обґрунтування економічної доцільності комерціалізації з точки зору інвестора.

4.1 Комерційний та технологічний аудит науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" є розширення функціональних можливостей програмного забезпечення для захисту файлів.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1 [24].

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження табл. 4.1

Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці. Для оцінки науково-технічного рівня і комерційного потенціалу розробки експертами було запрошено трьох незалежних експертів з Вінницького національного технічного університету, кафедри «Комп'ютерні науки»: Арсенюк Ігор Ростиславович, к.т.н., доцент; Барабан Сергій Володимирович, к.т.н., доцент, Белзецький Руслан Станіславович, к.т.н., доцент.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	Арсенюк Ігор Ростиславович	Барабан Сергій Володимирович	Белзецький Руслан Станіславович
	Бали:		
1. Технічна здійсненність концепції	3	2	3
2. Ринкові переваги (наявність аналогів)	4	3	4
3. Ринкові переваги (ціна продукту)	2	2	3
4. Ринкові переваги (технічні властивості)	4	4	4
5. Ринкові переваги (експлуатаційні витрати)	2	2	3
6. Ринкові перспективи (розмір ринку)	1	2	2
7. Ринкові перспективи (конкуренція)	3	3	3
8. Практична здійсненність (наявність фахівців)	4	2	2
9. Практична здійсненність (наявність фінансів)	2	2	2
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4

Продовження табл. 4.2

11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	4	4	4
Сума балів	$СБ_1$=37	$СБ_2$=34	$СБ_3$=38
Середньоарифметична сума балів $СБ_c$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{34 + 37 + 38}{3} = 36$		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3 [24].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" становить 36 балів, що, відповідно до таблиці 4.3 рівень комерційного потенціалу розробки вище середнього, що свідчить про комерційну важливість проведення даних досліджень.

Результатом магістерської кваліфікаційної роботи є веб-додаток, які корисні власникам (розробникам) мобільних додатків.

В якості аналога для розробки було обрано Shopify. Основними недоліками аналога є те, що він працює зі знижками лише для магазині створених за допомогою Shopify платформи і немає змоги керувати знижками для власних

продуктів створених поза платформою Shopify. Також до недоліків можна віднести те, що Shopify платформа містить малу кількість інструментів для управління знижками які не можуть існувати поза платформою Shopify і вони є жорстко прив’язані до цієї платформи.

У розробці дана проблема вирішується за допомогою того, що веб-додаток може існувати незалежно і немає жорстких обмежень по створенню знижок для будь-якого додатку який інтегрувався і працює разом з системою. Також система випереджає аналог за такими параметрами як гнучкість системи і легке внесення змін для подальшого налаштування під потреби кінцевого клієнта.

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему "Інформаційної технології управління знижками на товарні позиції для мобільних додатків", під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [24]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (4.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=21 дні.

$$З_о = 38000 \cdot 42 / 21 = 72545 \; грн.$$

Проведені розрахунки зведемо до таблиці 4.4.

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	38000	1727,3	42	72545
Інженер-програміст	46000	2090,9	63	131727
Тестувальник	30400	1381,8	10	13818
Всього				218091

Основна заробітна плата робітників. Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Інформаційна система підтримки підприємств малого бізнесу у сфері послуг» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i \,, \tag{4.2}$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} \,, \tag{4.3}$$

де $М_М$ – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=2379,00 грн;

$К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [24];

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати. Результат наведено у таблиці 4.5.

$Т_р$ – середнє число робочих днів в місяці, приблизно $Т_р$ = 21 дн;

$t_{зм}$ – тривалість зміни, год.

$$С_1 = 6500{,}00 \cdot 1{,}10 \cdot 1{,}65 \,/\, (21 \cdot 8) = 72{,}4 \text{ грн.}$$

$$З_{р1} = 72{,}4 \cdot 8{,}00 = 579{,}1 \text{ грн.}$$

Таблиця 4.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
1.Підготовчі	8	2	72,4	579,1
2.Інсталяція ПО	2	5	111,9	223,7
3.Проведення досліджень	8	3	88,8	710,7
4.Тестування системи	2	4	98,7	197,4
Всього				1710,9

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \qquad (4.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (218091+ 1710,9) · 11 / 100% = 24178,2 грн.

Відрахування на соціальні заходи. Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (218091+ 1710,9+24178,2) · 22 / 100% = 53675 грн.

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків".

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (4.6)$$

де $Н_j$ – норма витрат матеріалу *j*-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу *j*-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів *j*-го найменування, кг;

$Ц_{вj}$ – вартість відходів *j*-го найменування, грн/кг.

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 шт, грн	Норма витрат, шт	Вартість витраченого матеріалу, грн
Папір А4	185	1	185
Ручка	36	1	36
Диск оптичний CD	18	1	18
Flesh-пам'ять 64	350	1	350
Всього			589
З врахуванням коефіцієнта транспортування			647,9

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" відсутні.

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення в роботі відсутні.

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення. Програмні засоби, які були використанні при написанні магістерської роботи є безкоштовними.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.7)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$$A_{обл} = (35000 \cdot 1) / (2 \cdot 12) = 1458{,}33 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці 4.7.

Таблиця 4.7 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Комп'ютер	35000	2	1	1458,33
Приміщення лабораторії	300000	20	1	1250,00
Всього				2708,33

Витрати на силову електроенергію ($B_е$) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.8)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_е$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_е = 7,5$ грн;

$К_{вп і}$ – коефіцієнт, що враховує використання потужності, $К_{вп і} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$В_е = 0,25 \cdot 300 \cdot 7,5 \cdot 0,5 / 0,8 = 351,56 \text{ грн.}$$

До статті «Службові відрядження» дослідної роботи на тему "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (4.9)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св} = 20\%$.

$$В_{св} = (218091 + 1710,9) \cdot 20 / 100\% = 43960,36 \text{ грн.}$$

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» відсутні.

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.10)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів} = 50\%$.

$$І_в = (218091 + 1710{,}9) \cdot 50 / 100\% = 109900{,}9 \text{ грн.}$$

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.11)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв} = 100\%$.

$В_{нзв} = (218091 + 1710{,}9) \cdot 100 / 100\% = 219801{,}8$ грн.

Витрати на проведення науково-дослідної роботи на тему "Інформаційної технології управління знижками на товарні позиції для мобільних додатків". розраховуємо як суму всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.12)$$

$В_{заг}$=218091+1710,9+24178,2+53675+647,9+2708,33+351,56+43960,36 +109900,9+219801,8 =675026,46 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.13)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,5.

ЗВ = 675026,46 / 0,5 = 1350052,92 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 1 особа

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 15000,00 грн;

$\pm\Delta Ц_о$ – зміна вартості послуги від впровадження результатів, приймемо зростання на 1000,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 3-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [24]:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.14)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2021 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо ρ =40%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (1 \cdot 1000 + 15000 \cdot 800) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 2186750$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (1 \cdot 1000 + 15000 \cdot (800 + 700)) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 4100836$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (1 \cdot 1000 + 15000 \cdot (800 + 700 + 600)) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 5740770,4$ грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.15)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=18%;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 2186750 /(1+0,18)^1+4100836 /(1+0,18)^2+5740770,4 /(1+0,18)^3=$$
$$= 8007751,68 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \quad (4.16)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 1350052,92 грн.

$$PV = k_{інв} \cdot ЗВ = 2 \cdot 1350052{,}92 = 2700105{,}84 \text{ грн.}$$

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \tag{4.17}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 8007751,68 грн;

PV – теперішня вартість початкових інвестицій 2700105,84 грн.

$E_{абс} = ПП - PV = 8007751{,}68 - 2700105{,}84 = 5307645{,}84$ грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.18}$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, грн;

PV – теперішня вартість початкових інвестицій, грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+5307645{,}84 / 2700105{,}84)^{1/3} - 1 = 0{,}70.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \tag{4.19}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,1+0,25 = 0,35 < 0,70 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{4.20}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 0,70 = 1,4 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

4.4 Висновок до розділу 4

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків" становить 36 балів, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

Також термін окупності становить 1,4 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою "Інформаційної технології управління знижками на товарні позиції для мобільних додатків.

ВИСНОВКИ

При виконанні магістерської кваліфікаційної роботи розв’язано задачу розробки інформаційної технології та програмного забезпечення управління знижками на товарні позиції для мобільних додатків.

В першому розділі роботи проведено аналіз інформаційних технологій, пов'язаних з управлінням знижками на товарні позиції в мобільних додатках. Проведено докладний огляд основних інструментів та технологій, що використовуються для управління знижками в мобільних додатках. Визначено, які технології на сьогоднішній день є найбільш актуальними та популярними. Вивчені потреби цільової аудиторії, яка має інтерес до отримання знижок та акцій через мобільні додатки. Також було визначено очікування користувачів щодо інтерфейсу та функціональності таких додатків. Проаналізовані технічні характеристики мобільних платформ та операційних систем, які впливають на реалізацію інформаційної технології для управління знижками.

У другому розділі було розглянуто ключові аспекти розробки інформаційної технології для управління знижками на товарні позиції. В даному розділі проаналізовано загальну архітектуру розроблюваної інформаційної системи та визначено клієнтську частину, яку розроблятимемо з використанням React, Redux і TypeScript. Важливим аспектом є відзначення того, що розробка веб-додатку для створення та управління знижками відбуватиметься у середовищі React.

У цьому розділі було представлено детальну інформацію щодо програмної реалізації інформаційної технології для управління знижками на товарні позиції у веб-додатку. Було розглянуто вибір мови та середовища програмування, такі як JavaScript і TypeScript, і визначено їх важливість для розробки сучасного веб-додатку. Докладно описано модель активації знижок, що використовується в системі та надає загальний вигляд на процес взаємодії адміністраторів із знижками на товарні позиції. Було розглянуто важливі компоненти реалізації, такі як використання React та Redux, взаємодія з сервером через RESTful API,

використання бази даних. Створено зручний та сучасний користувацький інтерфейс та показані основні компоненти для роботи з додатком. Проведено процес тестування створеної системи, що включає в себе верифікацію функціональності та стабільності додатку. Усі ці аспекти разом створюють комплексний вигляд на розробку та реалізацію інформаційної технології управління знижками на товарні позиції.

Розроблене програмне забезпечення забезпечує:

- швидку відповідь від серверу приблизно 100мс, використану об’єктно-орієнтовану мову програмування JavaScript, обмеження в мінімальній кількості назві знижки у 3 символа та кількість типів знижок 7.

- покращення ефективності підприємств і брендів, які використовують цю технологію, шляхом залучення більше клієнтів та підвищення конверсії завдяки зручним інструментам управління знижками;

- забезпечує підвищений рівень ітеракцій між споживачами та підприємствами, що сприяє покращенню відносин та побудові лояльності клієнтів;

- допомагає оптимізувати рекламні кампанії, знижуючи витрати на рекламу та підвищуючи прибутковість.

- спрощує процесу планування та управління знижками на товарні позиції дл мобільних додатках, що робить його більш доступним та зручним для користувачів.

У четвертому розділі було виконано розрахунок витрат на розробку та виготовлення нового технічного рішення, сума яких складає 1350052,92 грн гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який потенційно може отримати виробник від реалізації розробленого технічного рішення, знайдено термін окупності витрат виробника та економічний ефект для споживача при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розробка у виробництві та використання дешевша за аналог і є висококонкурентоспроможною. Період окупності складе близько 1,4 роки.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Discrount Maganement. URL: https://dealhub.io/glossary/discount-management/ (дата звернення: 02.09.2023)

2. Манченко С. В., Богач І. В. АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ НА МОВІ ПРОГРАМУВАННЯ JAVASCRIPT З ВИКОРИСТАННЯМ REACT SPA [Електронний ресурс] // Матеріали Всеукраїнської науково-практичної інтернет-конференції «Молодь в науці: дослідження, проблеми, перспективи (МН-2023)» : збірник матеріалів. – Вінниця: ВНТУ, 2023. - Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2024/paper/view/19615

3. QR code (quick response code). URL: https://www.techtarget.com/whatis/definition/QR-code-quick-response-code (дата звернення: 03.09.2023)

4. Що таке програма лояльності та як їх використовувати. URL: https://tranzzo.com/uk-ua/blog/shcho-take-prohrama-loialnosti-ta-yak-yii-vykorystovuvaty (дата звернення 03.09.2023)

5. Купони і сертифікати. URL: https://incust.com/ua/knowledgebase/articles/coupon_and_certificate/ (дата звернення 03.09.2023)

6. Dosh. URL: https://www.dosh.com/ (дата звернення 03.09.2023)

7. RetailMeNot URL: https://www.retailmenot.com/ (дата звернення 03.09.2023)

8. Honey URL: https://www.joinhoney.com/ (дата звернення 03.09.2023)

9. What is Shopify and How does it work? URL: https://www.shopify.com/blog/what-is-shopify (дата звернення 03.09.2023)

10. What is Android. URL: https://www.android.com/intl/en_en/what-is-android/ (дата звернення 04.09.2023)

11. What is iOS. URL: https://www.socialpilot.co/social-media-terms/what-is-ios (дата звернення 04.09.2023)

12. What is React.js. URL: https://blog.hubspot.com/website/react-js (дата звернення 10.09.2023)

13. React JSX. URL: https://www.w3schools.com/react/react_jsx.asp (дата звернення 10.09.2023)

14. Introduction to Redux: URL: https://blog.logrocket.com/understanding-redux-tutorial-examples/#introduction-redux (дата звернення 10.09.2023)

15. TypeScript. URL: https://www.typescriptlang.org/ (дата звернення 10.09.2023)

16. TypeScript: What is it & when is it useful. URL: https://medium.com/front-end-weekly/typescript-what-is-it-when-is-it-useful-c4c41b5c4ae7 (дата звернення 10.09.2023)

17. What is the difference between UI and UX. URL: https://www.figma.com/resource-library/difference-between-ui-and-ux/ (дата звернення 17.09.2023)

18. Web Application Architecture in 2023: Moving in the Right Direction. URL: https://mobidev.biz/blog/web-application-architecture-types (дата звернення 20.09.2023)

19. What is JavaScript. URL: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript (дата звернення 22.09.2023)

20. About WebStorm. URL: https://www.componentsource.com/product/webstorm/about (дата звернення 23.09.2023)

21. Git Documentation. URL: https://git-scm.com/doc. (дата звернення 24.09.2023)

22. HTTP/S. URL: https://uk.wikipedia.org/wiki/HTTPS (дата звернення 25.09.2023)

23. The different types of software testing 2020. URL: http://surl.li/cflzv (дата звернення 26.09.2023)

24. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

Додаток А (обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія управління знижками на товарні позиції для мобільних додатків

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 87,4% Схожість 12,6%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Манченко С.В.

Керівник роботи Богач І.В.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програми

```
import React from 'react';
import {Switch,Route} from "react-router-dom";
import './App.css';
import "bootstrap/dist/css/bootstrap.min.css";
import Navbar from "./components/Navbar";
import ProductList from "./components/ProductList";
import Details from "./components/Details";
import Cart from "./components/Cart";
import Default from "./components/Default";
import Modal from './components/Modal';

function App() {
  return (
    <React.Fragment>
      <Navbar />
      <Switch>
        <Route exact path="/" component={ProductList} />
        <Route path="/details" component={Details} />
        <Route path="/cart" component={Cart} />
        <Route component={Default} />
      </Switch>
      <Modal />
    </React.Fragment>
  );
}

export default App;

import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import { BrowserRouter as Router } from "react-router-dom";
import {ProductProvider} from './context';
import * as serviceWorker from './serviceWorker';

ReactDOM.render(
    <ProductProvider>
        <Router>
            <App />
        </Router>
```

```
    </ProductProvider>,
     document.getElementById('root'));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();

import React, { Component } from 'react'

export default class Default extends Component {
    render() {
        console.log(this.props);
        return (
            <div className="container">
                <div className="row">
                    <div className="col-10 mx-auto text-center text-title">
                        <h1>error</h1>
                        <h1 className="display-3">404</h1>
                        <h2>page not found</h2>
                        <h3>the requested URL
                            <span className="text-danger">
                              {this.props.location.pathname}
                            </span>{""}
                            was not found</h3>
                    </div>
                </div>
            </div>
        )
    }
}

import React, { Component } from 'react'
import {Link} from 'react-router-dom';
import logo from '../logo.svg';
import styled from 'styled-components';
import {ButtonContainer} from './Button';

export default class Navbar extends Component {
    render() {
        return (
            <NavWrapper className="navbar nav-bar-expand-sm navbar-dark px-sm-5">
               <Link to='/'>
                  <img src={logo} alt="store" className="navbar-brand" />
               </Link>
               <ul className="navbar-nav align-items-center">
```

```
                <li className="nav-item ml-5">
                    <Link to="/" className="nav-link">
                        Products
                    </Link>
                </li>
            </ul>
            <Link to="/cart" className="ml-auto">
                <ButtonContainer>
                    <i className="fas fa-cart-plus">my cart</i>
                </ButtonContainer>
            </Link>
        </NavWrapper>
    )
  }
}
const NavWrapper = styled.nav`
background:var(--mainBlue);
.nav-link{
    color:var(--mainWhite) !important;
    font-size:1.3 rem;
    text-transform:capitalize;
}
`;

import React, { Component } from 'react';
import Product from "./Product";
import Title from "./Title";
import {ProductConsumer} from '../context';

export default class ProductList extends Component {
    render() {
        return (
            <React.Fragment>
                <div className="py-5">
                    <div className="container">
                    <Title name="our" title="products"/>
                        <div className="row">
                          <ProductConsumer>
                              {value=>{
                                return value.products.map(product =>{
                                    return <Product key={product.id} product={product} />;
                                });
                              }}
                          </ProductConsumer>
                        </div>
                        </div>
```

```
                    </div>
                </React.Fragment>
            );
        }
    }

import React from 'react'

export default function Title({name , title}) {
    return (
        <div className="row">
            <div className="col-10 mx-auto my-2 text-center text-title">
                <h1 className="text-capitalize font-weight-bold">
                    {name}  <strong className="text-blue">{title}</strong>
                </h1>
            </div>
        </div>
    )
}

import styled from 'styled-components';

export const ButtonContainer = styled.button`
text-transform:capitalize;
font-size:1.4rem;
background:transparent;
border:0.05rem solid var(--lightBlue);
border-color:${props => props.cart ? "var(--mainYellow)" :"var(--lightBlue)"};
border-radius:0.5rem;
color:${props => props.cart ? "var(--mainYellow)" : "var(--lightBlue)"};
padding:0.2rem 0.5rem;
cursor:pointer;
margin:0.2rem 0.5rem 0.2rem 0;
transition:all 0.5s ease-in-out;
&:hover{
    background:${props => props.cart ? "var(--mainYellow)" : "var(--lightBlue)"};
    color:var(--mainBlue);
}
:focus{
    outline:none;
}
`;
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ УПРАВЛІННЯ ЗНИЖКАМИ НА ТОВАРНІ ПОЗИЦІЇ ДЛЯ МОБІЛЬНИХ ДОДАТКІВ

Виконав: студент 2-го курсу,
групи 2КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Манченко С.В.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. КН
_____ Богач І.В.
(прізвище та ініціали)
«07» 12 2023 р.

Вінниця ВНТУ - 2023 рік

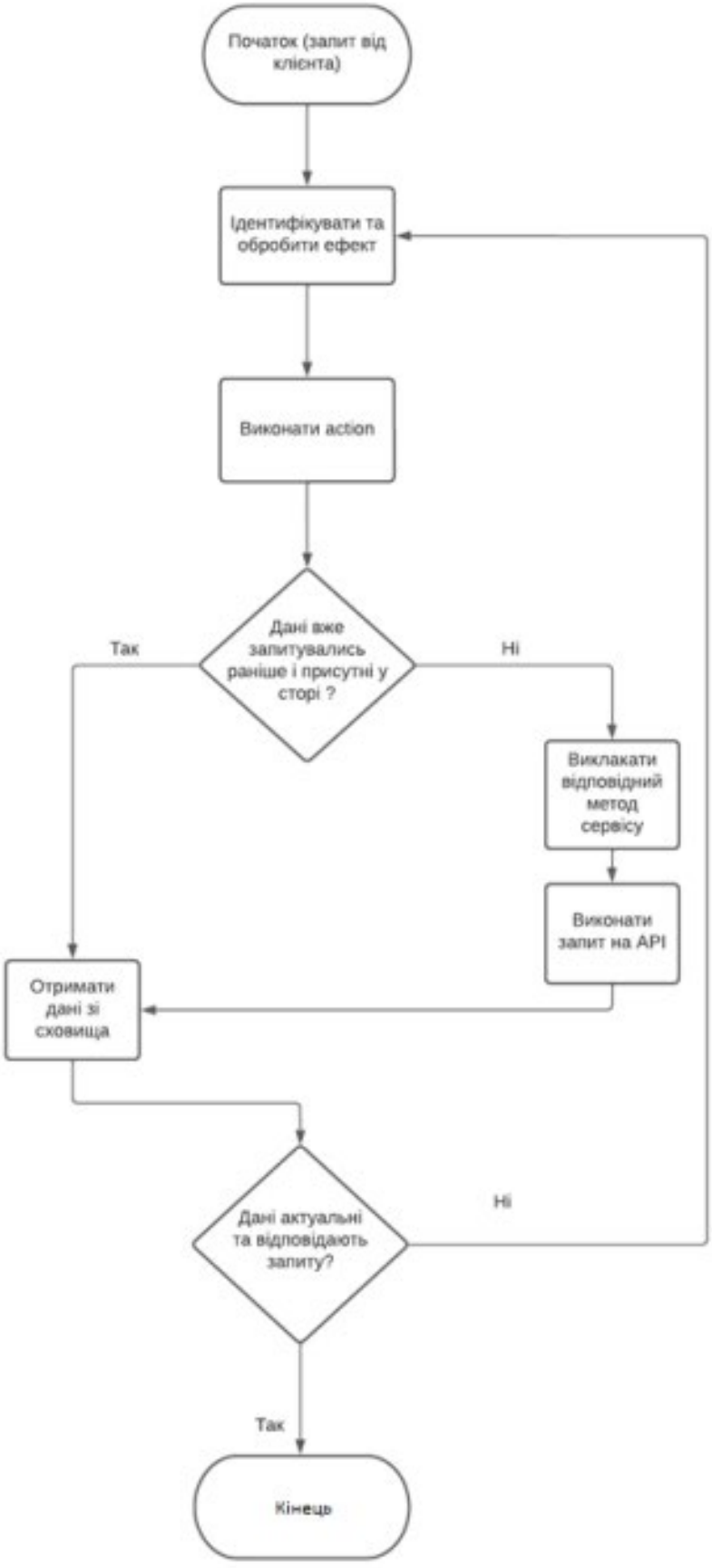

Рисунок В.1 – Алгоритм інформаційної технології роботи

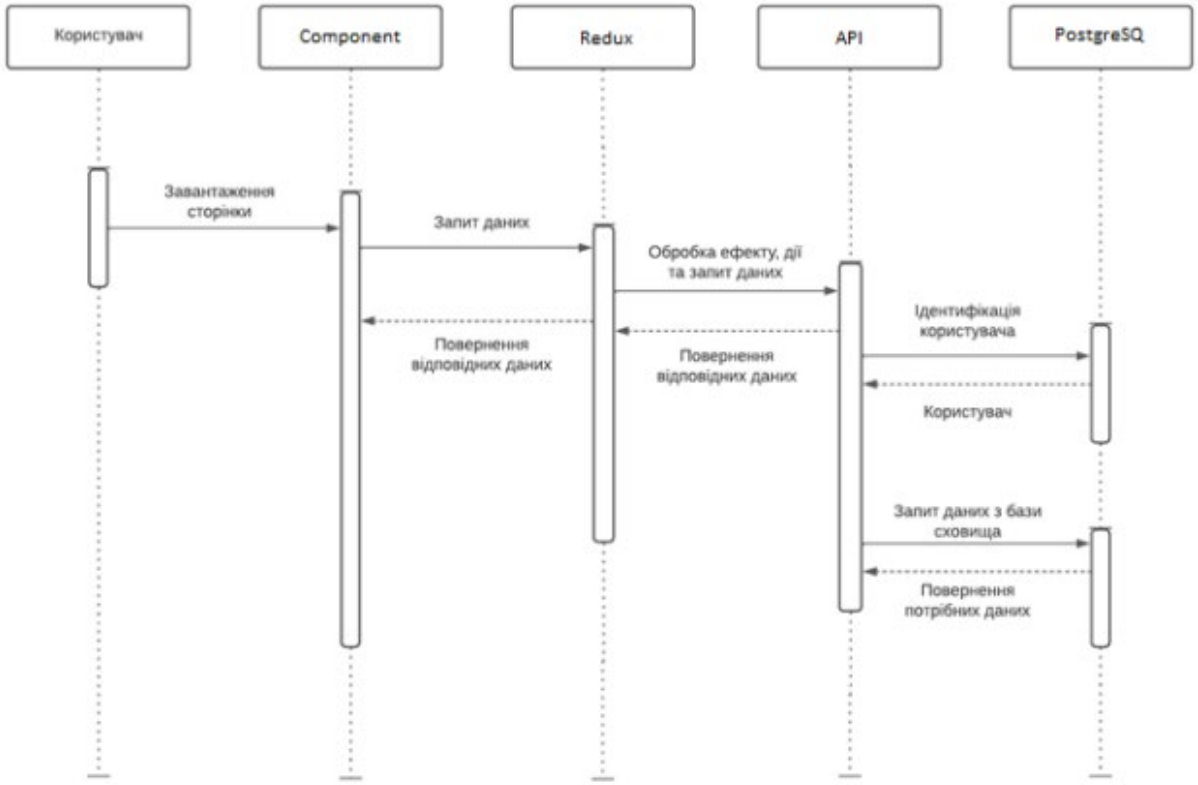

Рисунок В.2 – UML діаграма роботи веб-додатку

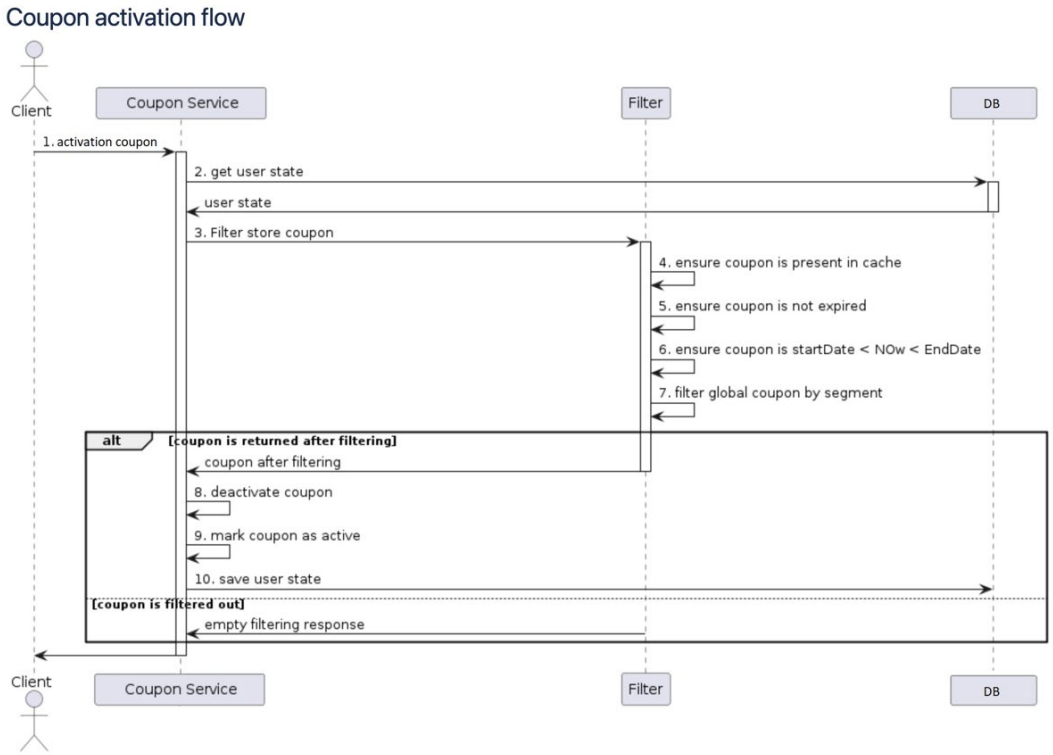

Рисунок В.3 – UML-діаграма активації купона користувачем

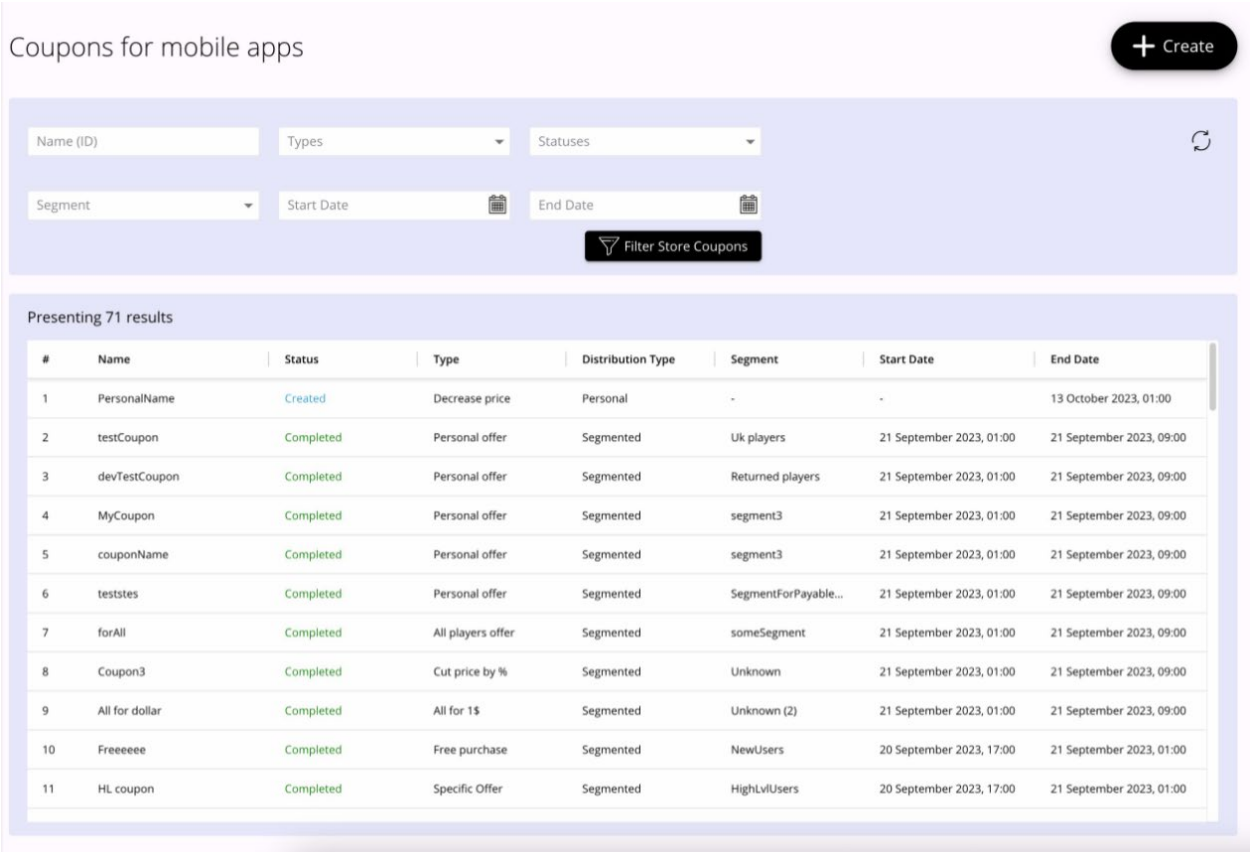

Рисунок В.4 – Головна сторінка інформаційної технології

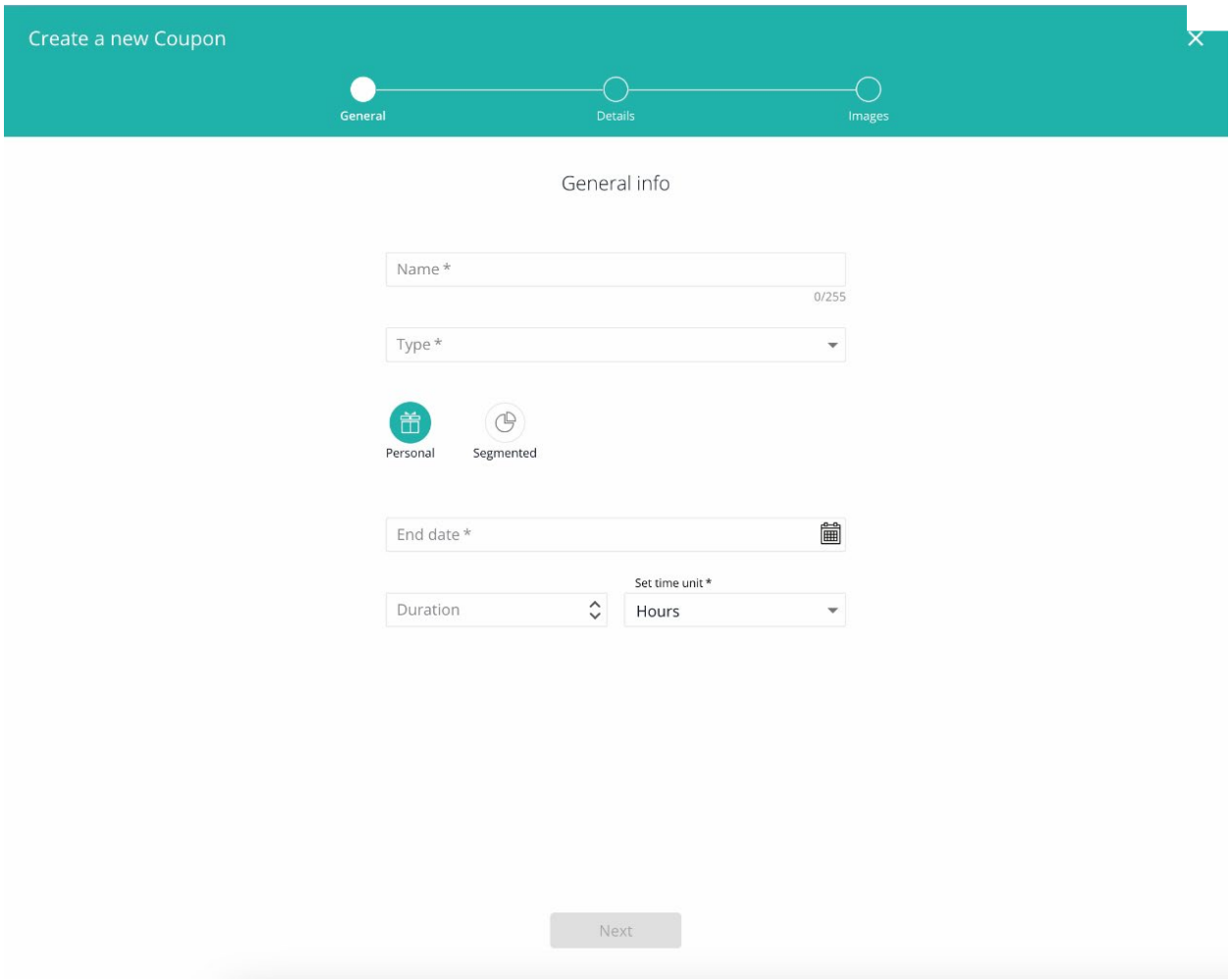

Рисунок В.5 – Робоче вікно створення знижки

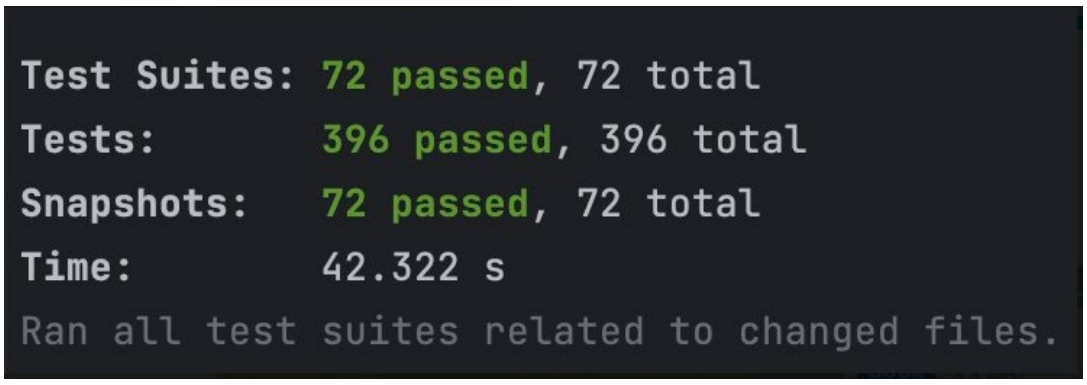

Рисунок В.6 – Результат тестування інформаційної технології

Додаток Г (довідниковий)

Інструкція користувача

Для авторизації та переходу на сторінку користувача, необхідно:

1. Перейти на головну сторінку за посиланням – http://domain:3000/
2. Клацнути на іконку з зображенням користувача.
3. Заповнити всі поля.
4. Нажати кнопку входу.

Для того, щоб створити знижку, необхідно:

1. Перейти на головну сторінку системи за посиланням – http://domain:3000/
2. Нажати лівою кнопкою миші на пошук на кнопку Create.
3. Заповнити необхідні поля
4. Нажати кнопку Next
5. Заповнити необхідні поля
6. Нажати кнопку Finish