Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

ДИПЛОМНИЙ ПРОЕКТ

Інтернет-платформа магазин з продажу взуття
Назва теми

Рівень вищої освіти Перший (бакалаврський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр ДППЗ.180122.01.14.ПЗ

Виконав студент IV курсу група ІПЗ-18-1 _____ В. В. Федоренко
Підпис Ініціали, прізвище

Керівник д-р фіз.-мат. наук професор _____ Л. П. Бедратюк
Науковий ступінь, звання Підпис Ініціали, прізвище

Нормоконтролер канд. пед. наук, доцент _____ Н. І. Праворська
Підпис Ініціали, прізвище

До захисту допускаю:
Завідувач кафедри інженерії
програмного забезпечення _____ Л. П. Бедратюк
Підпис Ініціали, прізвище

2 червня 2022 р.

Хмельницький 2022

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій
Кафедра Інженерії програмного забезпечення
Рівень вищої освіти Перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Л. П. Бедратюк
01 03 2022 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ)

Федоренку Владиславу Вікторовичу
Прізвище, ім`я, по батькові студента

1. Тема проєкту (роботи) Інтернет-платформа магазин з продажу взуття

Керівник проєкту (роботи) Бедратюк Леонід Петрович, д-р фіз.-мат. наук професор
Прізвище, ім`я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 01.03.2022 р. № 18
2. Строк подання студентом проєкту (роботи) на кафедру 01.06.2022 р.
3. Вихідні дані до проєкту (роботи) Матеріали переддипломної практики
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
Дослідження предметної області та постановка задачі, проектування програмного забезпечення, програмна реалізація, тестування програмного забезпечення
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)
Презентаційні матеріали (слайди, 17 шт)

АНОТАЦІЯ

Тема дипломного проекту: «Інтерент-платформа магазин з продажу взуття».

Автор проекту: Федоренко Владислав Вікторович.

Керівник проекту: Бедратюк Леонід Петрович.

Пояснювальна записка: 167 с., 47 рис., 6 табл., 4 дод., 11 джерел.

Графічна частина: 17 презентаційних слайдів.

ЕЛЕКТРОННИЙ МАГАЗИН ВЗУТТЯ, ІНТЕРНЕТ-ПЛАТФОРМА, APACHE Tom Cat, MySQL, Java, Spring Freamwork, Hibernate.

Об'єктом дослідження є підприємство, що займається реалізацією товарів і послуг через мережу Інтернет, що працює у галузі продажу взуття.

Метою проекту є створення системи електронної комерції для реалізації товарів оптичного асортименту через мережу Інтернет.

У дипломному проекті проведено аналіз предметної області та наявного програмного забезпечення, визначені вимоги до системи, розроблена загальна архітектура застосунку, спроектована структура бази даних та структура застосунку. Визначено основні модулі та здійсненна реалізація програмного забезпечення.

Для розробки програмної системи використано мову програмування Java, сервер бази даних MySQL і Web-сервер Apache Tom Cat, фреймоврки Spring Freamwrok, Hibernate.

У результаті проектування здійснена програмна реалізація системи для продажу товарів чоловічого, жнічого взуття.

31.05.22
Дата


Підпис

ВІДОМІСТЬ ДОКУМЕНТІВ

№ рядка	Формат	Позначення документа	Найменування документа	К-сть аркушів	№ екз.	Примітка
			Текстові документи			
1	А4	ДППІЗ.180122.01.14.ПЗ	Пояснювальна записка	167		
2	А4		Завдання на дипломний	1		
			проект			
3	А4		Анотація	1		
			Графічні документи			
4	А4		Презентаційні матеріали	17		

Змн.	Арк.	№ докум.	Підпис	Дата	ДПІПЗ.180122.01.14.ВД		
Виконав		Федоренко В.В.		31.05.2	Інтернет-платформа магазин з продажу взуття	Літ.	Арк. 4 / Акрушів 167
Керівник.		Бедратюк Л.П.		31.05.2			
Н. Контр.		Праворська Н. І.		31.05.2	Відомість документів	ХНУ, ІПЗ-18-1	
Затверд.		Бедратюк Л.П.		31.05.2			

ЗМІСТ

					ДПІПЗ.180122.01.14.ВД	
Змн.	Арк.	№ докум.	Підпис	Дата		
Виконав		Федоренко В.В.		31.05.22	Інтернет-платформа магазин з продажу взуття	Літ. / Арк. 5 / Акрушів 167
Керівник.		Бедратюк Л.П.		31.05.22		
Н. Контр.		Праворська Н.І.		31.05.22	Відомість документів	ХНУ, ІПЗ-18-1
Затверд.		Бедратюк Л.П.		31.05.22		

ПЕРЕЛІК СКОРОЧЕНЬ

БД	–	база даних
ПЗ	–	програмне забезпечення
СУБД	–	Система управління базами даних
ООП	–	об’єктно-орієнтоване програмування
ПЗ	–	програмне забезпечення
JS	–	JavaScript
API	–	Application programming interface
DBS	–	Data Base Server
HTML	–	HyperText Markup Language
HTTP	–	HyperText Transfer Protocol
UML		Unified Modeling Language
CSS	–	Cascading Style Sheets

ВСТУП

Інтернет-торгівля – це процес за допомогою, якого здійснюється товарообіг фізичних і не фізичних товарів, які здійснюються в інтернет платформах, а саме на таких платформах можливо дистанційно находячись в любому місце маюче лише доступ до інтернет мережі оформити замовлення на будь-який вид товару. З кожним роком створюється все більше і більше нових платформ для торгівлі, тому що в Україні обсяг товарообігу збільшується.

На сьогоднішній час завдяки розвитку інформаційних технологій є розвиток в веб-розробці. Завдяки розвитку в цій галузі за для того, щоб задовольнити потреби користувача створюються величезна кількість різноманітних і інтерактивних веб-застосунків, а особливо в торгівельній сфері.

На превеликий жаль величезна кількість підприємців не використовуються інтернет, який так міцно увійшов в наше життя, тому що вони навіть не можуть уявити, як з використанням технологій можна зробити свій бізнес прибутковим.

На теперішній час звичайним користувачам більше не потрібно бігати по магазинам, базарам, щоб придбати собі товар. За допомогою всього лише телефона ми маємо змогу обрати, оплатити та отримати товар. Всі ці можливі маніпуляції економлять нам цілу купу часу. Особливо актуальним онлайн покупки стали в період пандемії COVID-19. До інтернет покупок були залученні навіть ті люди, які ніколи в своєму житті їх не здійснювали, тому що в період пандемії, люди в різних містах, країнах не мали змогу придбати собі навіть звичайне взуття, а завдяки технологіям вони змогли задовольнити свої потрібності.

Можна сказати, що магазину з продажу товару в інтернеті замінили традиційні магазини, але це не так, тому що вони навпаки тільки розширили свою сферу застосування. Підприємці, які не створили свій власний інтернет магазин допустилися величезної стратегічної помилки.

Інтернет-платформи дають неймовірні можливості різним підприємцям, компаніям представити свій товар у електронному вигляді і стисло, а також інформативно надати інформацію про свій товар.

Отож, з кожнім днем і роком ринок інтернет торгівлі збільшується, а також збільшується і конкуренція, тому підприємці розуміють, що потрібно створювати власні інтернет маркетплейси. Через збільшення конкуренції на ринку інтернет-магазинів, потрібно створювати все візуально простіший і зручніший для користувача веб-ресурс, а для продавця більш зручну систему керування, який надає величезну кількість можливостей для придбання товару або фіт беку для продавця про його товар.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Змістовний аналіз предметної області, її структурних та функціональних особливостей

У даний час веб-сайти стали загальнодоступними. І кожен підприємець чи фірма розуміють важливість інтернет-платформ, тому кожна велика чи маленька фірми мають свій веб-портал де вони розміщують свій власний товар або інформації про їхню власну продукцію. Також власники різних підприємства долучають рекламу для збільшення успіху і прибутку в своїй інтернет-платформі.

Суспільство оцінило зручність і швидкість використання сайтів, тому що для того, щоб зробити покупку потрібно всього лише мати комп'ютер чи телефон. З розвиток комп'ютерних технологій дуже швидко прийшла діджиталізація суспільства, а з розвитком інтернету взагалі всі побутові речі можливо зробити в пару натискань.

Через збільшення конкуренції і розвитку технологій крамниці потребують побудови інформаційної системи, який буде містити в собі всі необхідні інструменти для успішного управління бізнесом в теперішній час. А з розвитком Інтернету виросла більша необхідність в створенні сайтів для надання різноманітної інформації.

Сайти можуть бути різними. Можуть бути, як не великі інтернет-платформи на яких відображають не велику інформацію про фірму або її послуги, таким чином з'явились інтернет-каталоги з максимально деталізованою інформацією і характеристики товару, які мають їхнє зображення і ціну. Здебільшого такі сайти створюють для того, щоб користувач міг оглянути товар представлений на фотографії або прочитати детальний опис продукту. Тобто така інтернет-платформа виконує функції рекламного сайту товарів.

Електронний магазин – це автоматизована торгівельна система, що цілодобово працює і забезпечує можливість ведення торгово-облікових операцій,

яка працює за допомогою інформаційної інтернет мережі. Даний вид магазинів подібний до звичайних крамниць, а саме їхніми основними функціями є: демонстрація товару користувачу, обробка замовлення, а також продаж і доставка, товару за вказаною адресую.

Інтернет-магазин містить в собі традиційні і прямі елементи маркетингу. На відмінну від звичайної форми продажу товару інтернет-магазин надає змогу продавати більшу кількість товару і послуг, а також надавати більш стисліше і конкретнішу інформацію про товар, які є необхідними для прийняття рішення про покупку. До того ж , за допомогою сучасних комп'ютерних технологій кожен користувач можу бути персоналізованим до історії змовника, а також товар буде підбиратись відповідно за його смаком.

Базовими елементами в інтернет магазинні є:

- Відображення назви товару;
- Наявність зображень товару;
- Наявність різного варіанту товару (розміру, кольору);
- Ціна;
- Характеристики товару;
- Наявність у продажі.

Одною з проблем в упровадженні інтернет-магазину – поєднання веб-технологій із звичайною троговою діяльністю. У звичайній виді торгування покупці звикли до того, що можуть візуально оцінити товар для того, щоб визначити її якість. В інтернет магазині користувач не має змоги оцінити товар на якість та перевірити зазначенні характеристики. Дуже часто буває таке, що товар доставляють пошкоджений, зазвичай це буває через винну служби доставки, а не продавця.

За останні пару років, чисельність користувачів подобних інтернет-магазинів стає все більшою, а саме продажі зростають, тому торгівельні експерти говорять, що використання веб-технологій у продажі товару є високо актуальною на цей момент. Майже кожен день з'являються різні інтернет магазини, тому що дуже вигідно і зручно продавати товар в інтернеті через те, що не потрібно

орендувати приміщення для продажу свого товару, а користувач сам має змогу зайти на торгівельну площадку і подивись представлений перед ним товар. Все, що потрібно покупцю це всього лише вибрати товар, оформити замовлення, домовитись з постачальником про місце надходження товару і час коли це буде доречно. Невідмінно від звичайних магазинів їхня кількість обмежена, а інтернет магазин працює цілодобово, потрібно всього лише мати доступ до інтернет мережі. Отже інтернет магазини економлять підприємцям гроші і час, а користувачам надають змогу в любому зручному для неї пристрої і часі: вибрати товар, прочитати про нього відгуки і оформити замовлення, що також економить час і сили покупцю.

1.2 Аналіз наявного програмного-технічного забезпечення предметної області

Для аналізу наявного програмно-технічного забезпечення було обрано декілька сайті з схожою тематикою і призначенням.

Одним із веб-сайтів є сайт магазин брендового молодіжного одягу Staff представлений на рисунку 1.1. Staff – це один з найвідоміших молодіжних магазин з продажу одягу й взуття в Україні. Станом на 2022 рік під управлінням знаходяться 22 магазина у 16 містах України. Окрім цього, магазин різко розвивається в напрямку продажу товару в інтернеті, запустивши сайт в 2013 році молодими і креативними людьми. Технології розробки весь час розвиваються і вдосконалюються маючи в команді висококваліфікованих спеціалістів.

Перше за все, що побачить кожен користувач, який перейде на цю інтернет-платформу навіть немаючі досвіду в розробці або дизайні – це зручний та сучасний інтерфейс де користувач може обирати, яка продукція його цікавить чоловіча або жіноча. Зручні розділи продукції де відображаються знижки, новинки, одяг, взуття, аксесуари, а також контактні данні адміністрації сайтів для

швидкого звернення до них за номером телефону. Крім того реалізований чат з менеджером, який весь час знаходить онлайн і до, якого ти також можеш звернутися по будь-яким питанням. Також маємо змогу створити обліковий запис користувача і заповнити всі відповідні поля тоді під-час оформлення замовлення всі контактні дані будь автоматично заповненні, що спрощує і пришвидшує час оформлення замовлення користувачу.

Але навіть в таких сайтах є ціла купа недоліків хоча на перший погляд сайт виглядає ідеально. Товар можливо тільки придбати зі своєї персональної сторінки, що збільшує час покупки товару. Звісно це є не дуже зручно, тому що більшість користувачів не має наміру зберігати свої данні в системі і мають бажання замовити товар через мобільний дзвінок. Тому на мою думку потрібно серйозно підійти до цього питання і покращити свою систему ввівши таку функцію в систему.

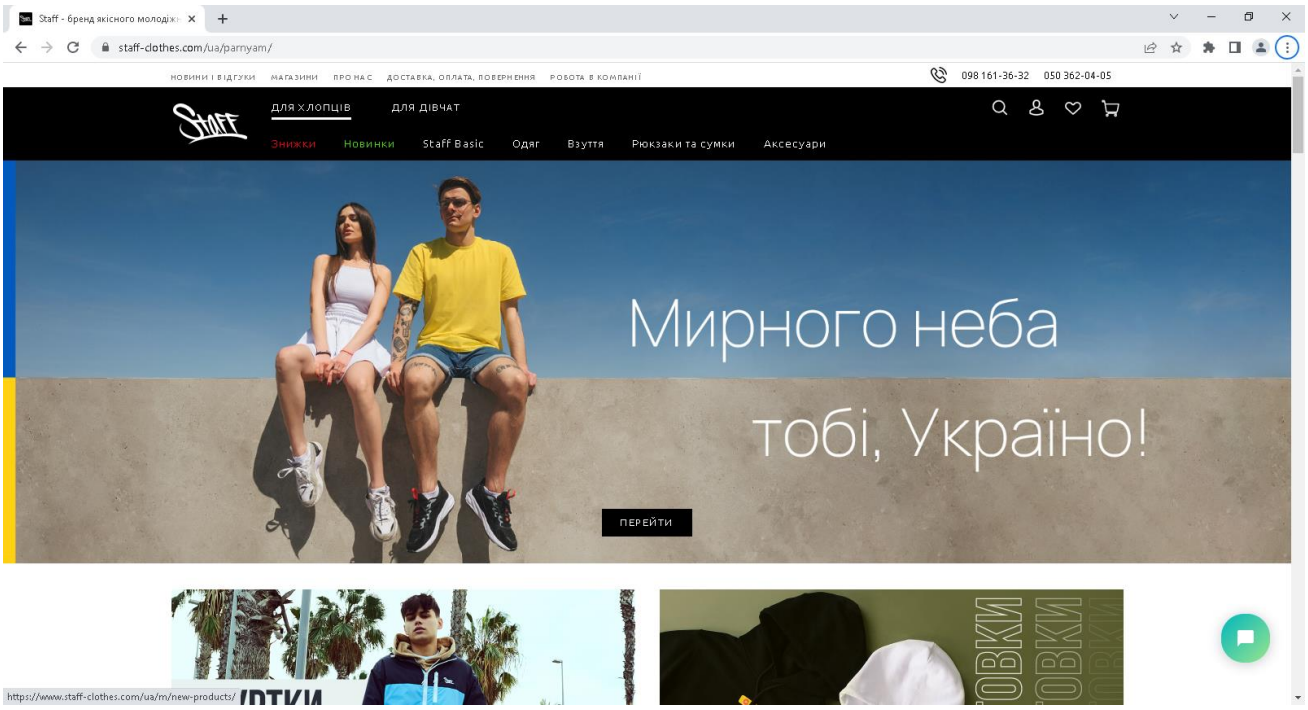


Рисунок 1.1 – Головна сторінка магазину Staff

Sezon – це український магазин чоловічого, жіночого та дитячого взуття, заснована у 2010 році де все почалось у маленькому магазині, який знаходився у місті Чернівці. Магазин взуття Sezon має всього лише два магазини у двох містах: Чернівці і Київ, тому що цей магазин більше спеціалізується на продажі товару через інтернет, тому він створений таким чином аби покупцю було максимально інтуїтивно зрозуміло і зручно знаходитесь на цій інтернет платформі, який представлений на рисунку 1.2.

Інтерфейс сайту інтуїтивний і зрозумілий. Окрім каталогу товару і форми швидкого пошуку є також розділення товару, як для чоловіків, жінок і дітей. Також є контактні телефони завдяки яким користувач може додзвонитись до менеджера і задати йому питання. Крім того власники магазину ведуть власні блоги в соціальних мережах таких як Facebook, Instagram, Twitter і YouTube. Ведення блогу в соціальних мережах є дуже гарною практикою, тому що є змога охватити більше аудиторію. А найголовніше, що на YouTube є відеоролики з товаром, який продають в магазині, так покупець зможе найти і подивитись, як виглядає товар не тільки на фото, але і на відеохостингу.

Одним із найголовніших недоліків даної інтернет платформи це є відсутність приближення фотографії взуття, тому що дуже часто покупець хоче роздивитись фотографію поближче, але немає змоги це зробити. Ще одним недоліком є – це неможливість повернення до початку сторінки одним натиском, потрібно підніматись на сторінку вверх самостійно. Способом вирішення даної проблеми –це створення клавіші швидкого автоповернення вгору. Також є великим мінусом це відсутність зворотного зв’язку з покупцем під час замовлення. Також бувають моменти, коли менеджери по роботі з клієнтами тривалий час не відповідають і не дають відповіді на задане перед ним питанням.

Отже, даний інтернет-магазин є непоганим, але він і має багато недоліків, що потребують рішення.

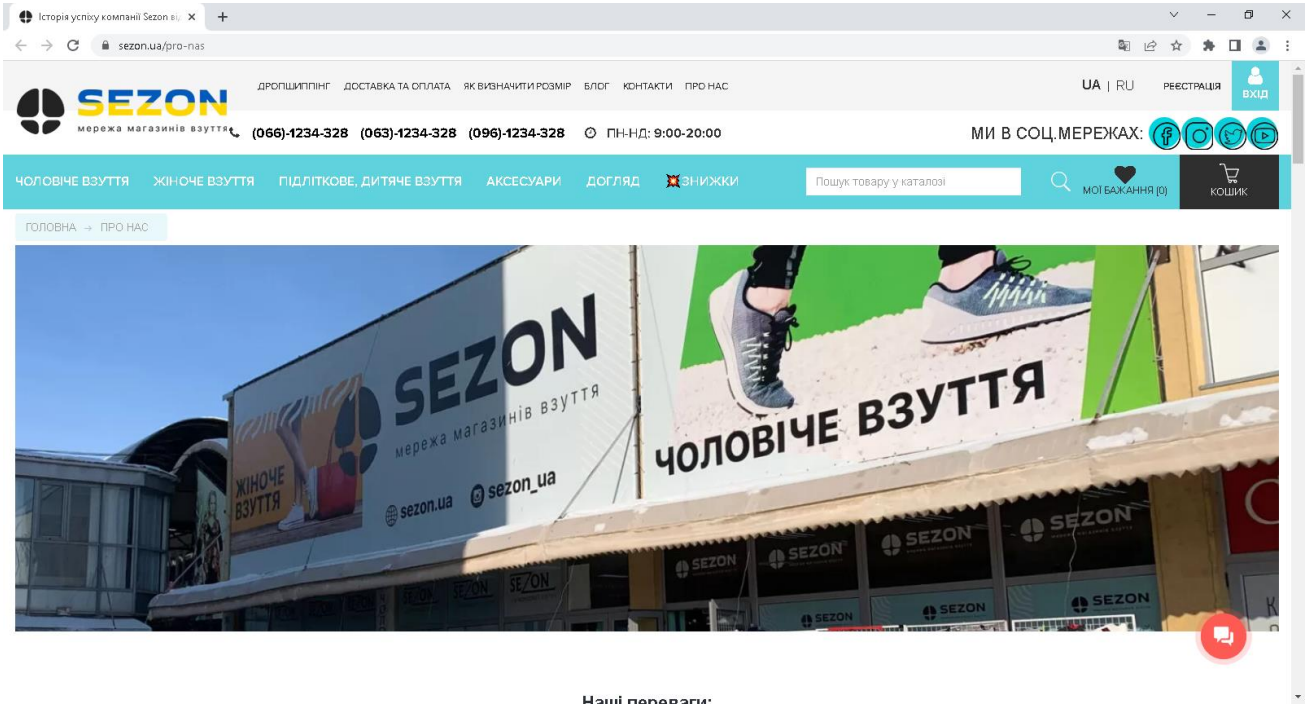


Рисунок 1.2 – Головна сторінка магазину Sezon

Nike – це американська компанія, яка спеціалізується на спортивному одязі та взутті. Даний магазин був заснований у далекому 1964 році. Дана компанія є досить сучасною не дивлячись на її вік з приходом інтернету і технологій веб розробки ця компанія почала орієнтуватись на продажі товару через інтернет, що привело до збільшення її капіталу і росту клієнтів серед прихильників даної продукції. Представлений на рисунку 1.3.

При вході на сайт користувачу дається можливість вибору свого власного регіону, тоді весь текст перекладається на ту мову, яка зручніше користувачу, але при необхідності її можливо змінити, що є дуже зручно, якщо ти українець, але знаходишся в Німеччині і ти не розумієш іноземну мову. Також, якщо заходиш через смартфон і ввімкнена геолокація, мова і сайт автоматично адаптуються під потенційного покупця.

Інтернет-платформа виглядає досить сучасно і мінімалістично. Окрім того зразу випливає форма авторизації або реєстрації, що є досить зручно, тому що це

інтернаціональний магазин і без реєстрації і введені даних буде досить складно замовити товар.

Як у вище зазначених сайтах є також можливість вибрати товар за категоріями, що робить вибір товару набагато легшим.

На жаль, також є дуже багато недоліків. Найголовніше – це немає можливості зв’язатися з менеджерами по роботі з клієнтами, що дуже сильно ускладнює роботу з сайтом для не опитних користувачів. Дана проблема дуже легко вирішується реалізацією маленького віконечка для чатування з менеджером по роботі з клієнтами, але нажаль це є фінансово затратним задоволенням і для того, щоб обслужити всіх клієнтів доведеться переписувати алгоритми роботи сайту, а також наймати величезну кількість працівників, щоб працювало без затримки.

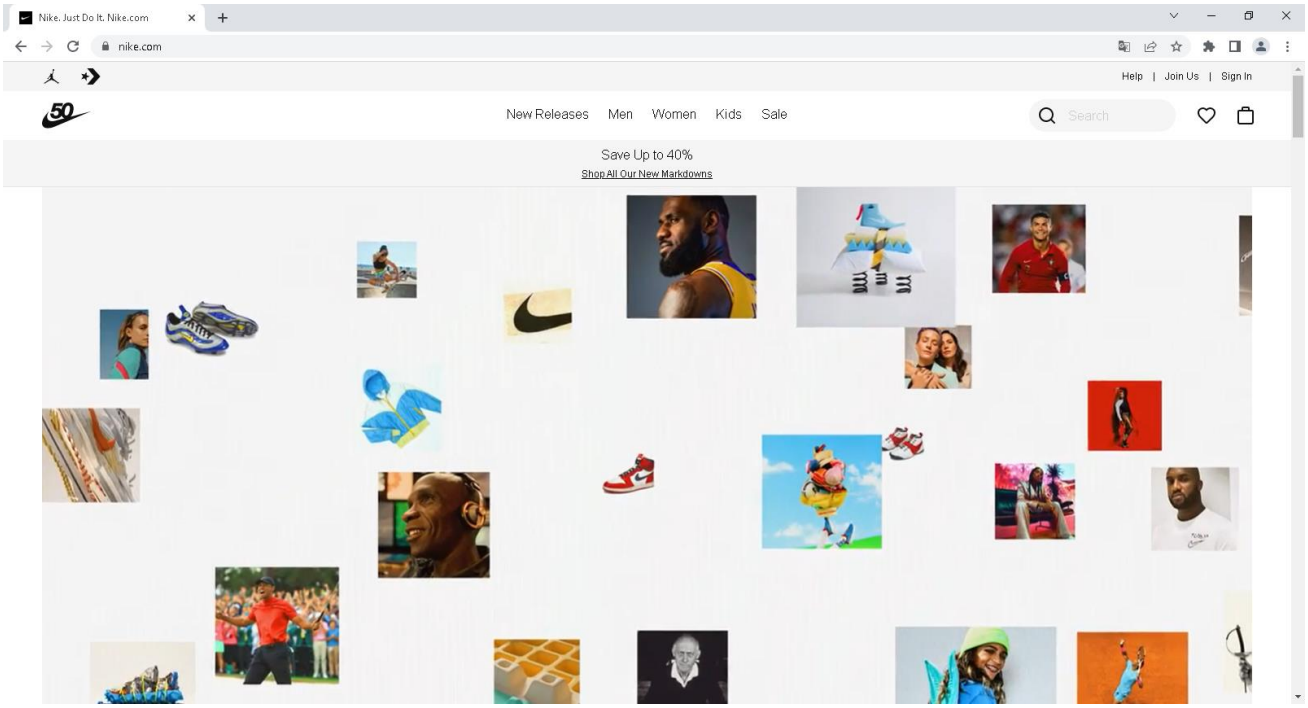


Рисунок 1.3 – Головна сторінка магазину Nike

Intertop – це українська мережа магазинів, яка продає в роздріб взуття і одяг, який був заснований в 1994 році, на 2019 рік даному магазину належать 136 магазинів в 27 містах України. Завдяки технологіям цей магазин почав дуже швидко розвиватись і почала співпрацювати з різними брендами взуття, які продавали свій товар через цей магазин, який представлений на рисунку 1.4.

Перше за все при вході на сайт покупець побачить чудовий і зручний інтерфейс, який дає змогу дуже легко орієнтуватись і шукати товар за каталогом і категоріями. Також є можливість бистрого пошуку, що полегшує користування даним ресурсом, а також можливість зв'язку з менеджером, що є дуже зручно і чудово, тому що він зможе допомогти у будь-якій проблемі.

Натомість, як кожна інтернет-платформа вона має свої недоліки перше це кожен товар має досить малий опис, що визиває сумніви до якості товару. Нажаль немає можливості поставити нічну тему для сайту, що набагато покращило час проведення на сайті, тому що білий і яскравий фон у темну пору доби досить сильно ріже очі, що негативно впливає на здоров'я і самопочуття, але ця проблема досить швидко вирішується де при на тиснені на кнопку просто замінюється фонова частина сайту на більш темнішу.

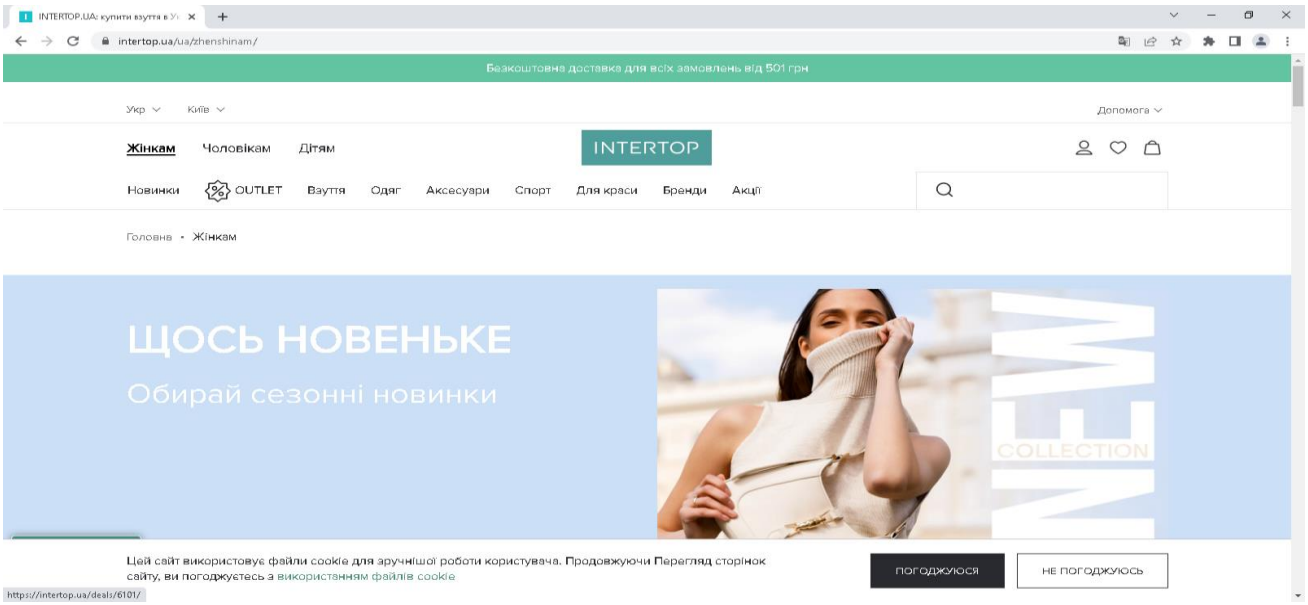


Рисунок 1.4 – Головна сторінка магазину Intertop

Estro – це український магазин взуття для чоловіків та жінок, що власноруч створює власне взуття. Який зображений на рисунку 1.5.

Основна перевага цього магазину це простий і мінімалістичний інтерфейс в якому дуже легко і зручно розібратись. При вході на веб-сторінку ми бачимо розділи, які поділяють чоловіче і жіноче взуття, також сумки і аксесуари, скидки, акції. Також є можливість відправити запитання після чого буде зворотній зв'язок з відповіддю на питання. Маємо можливість здійснити телефонний дзвінок за номерами вказаними вище і здійснити покупку, якщо не має бажання реєструватись. Великим плюсом є вікно швидкого пошуку і реалізований кошик з товаром, який сподобався.

Головним мінусом є те, що немає моментального зворотного зв'язку потрібно очікувати певний час на відповідь. Ця проблема дуже легко вирішується за допомогою влаштування в інтернет-платформу чату з менеджерами по роботі з клієнтами.

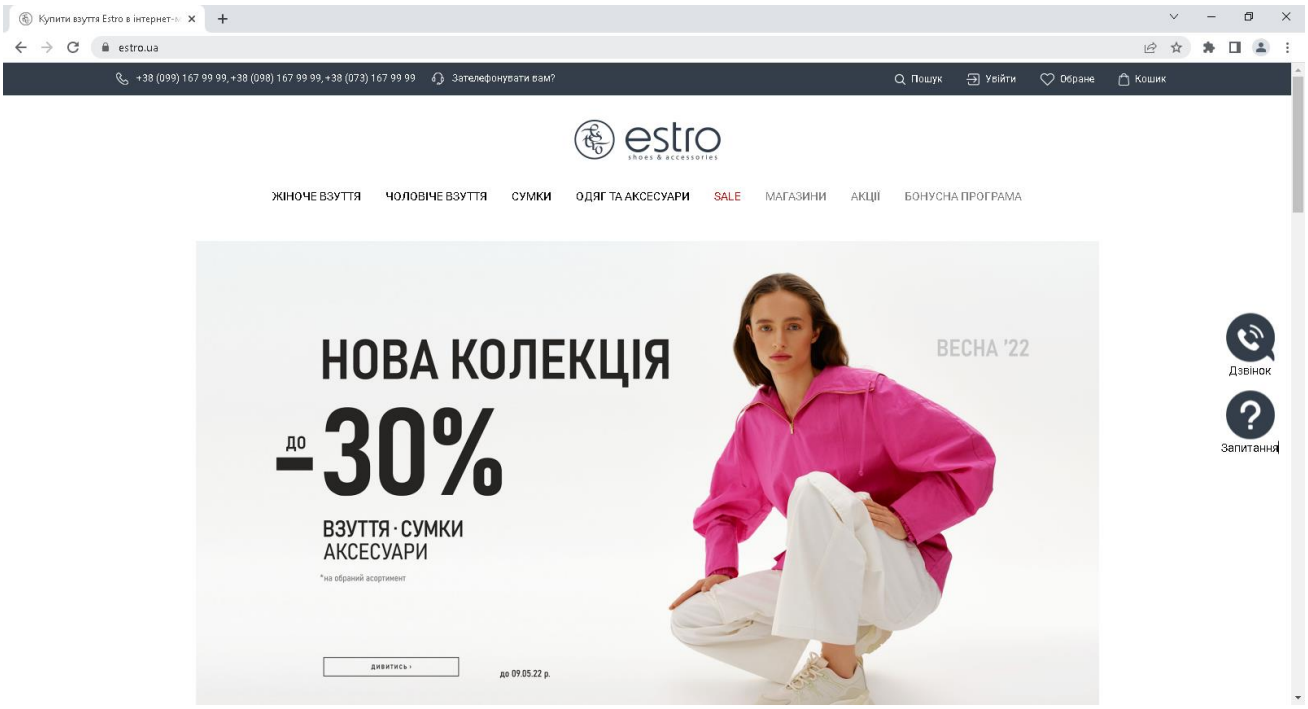


Рисунок 1.5 – Головна сторінка магазину Estro

Хоча на сьогоднішній день створено досить багато різних сайтів подібного плану, але вони всі мають певні недоліки:

– Неактуальність інформації. Кожен день бізнес розвивається і з’являються нові ідеї, які потрібно реалізувати. Майже кожен день змінюється товар на складі і потрібно слідкувати за його кількість. Але нажаль не завжди виходить активізувати інформацію у інтернет-платформі;

– Зручність використання. Самий яскравий приклад це те що користувач не може самостійно розібратись, як оформити замовлення, куди натиснути аби дізнатись всі умови доставки;

– Неквапліва обробка замовлень.

Отже проведемо більш детальний аналіз і розглянемо його таблиці 1.1.

Таблиця 1.1 – аналіз попередньо розглянутих інтернет-платформ для продажу взуття

Критерії	Інтернет-платформа Staff	Інтернет-платформа Sezon	Інтернет-платформа Nike	Інтернет-платформа Intertop	Інтернет-платформа Estro
Відсутність реклами на сайті	+	+	+	-	+
Непотрібна інформація	-	-	-	+	-
Зворотній зв’язок	+	-	-	+	-
Сучасність дизайну	+	+	+	+	+
Можливість збільшувати чи зменшувати фотографії	+	-	+	+	+
Видима цінова політика	+	+	+	+	+

Отже, проаналізувавши всі вище перечисленні інтернет-платформи було виявлено недоліки, які представлені в таблиці 1.1. Тому дана розробка матиме переваги даних сайтів, а також зручний і простий дизайн для легкості роботи користувача .

1.3 Визначення вимог до розробки інтернет-платформи

На сьогоднішній час перед розробкою будь-якого ПЗ ставить безліч різноманітних вимог і завдань: від створення розважальних сайтів, сайт візиток до серйозних бізнес проектів, які потребуються від розробника надійності і безпеки від непотрібного стороннього доступу. Для реалізації потрібно правильно підійти до вибору таких аспектів, як мов програмування на якому буде писатись backend, фреймворки, які можуть спростити нам розробку і доступу до бази даних, які дуже часто використовуються у розробці і стають все актуальнішими на сьогоднішній час.

Звісно на теперішній час є досить багато різних мова програмування і фреймворків, тому кожному програмісту потрібно вибирати мову програмування таку в якій має достатньо знань аби реалізувати проект.

Перед розробкою програмного забезпечення потрібно визначитись з мовою програмування. Тому потрібно розуміти, що для того, щоб розробити інтернет-платформу потрібно володіти такими мовами frontend розробки, як HTML, CSS, JavaScript. Для розробки backend частини сайту було вибрана така мова програмування, як Java з використанням Spring Framework, що значно облегшить нам роботу в створені сайту і доступу до БД. Бази даних, які ми будемо використовувати це MySQL – одна з найпопулярніших реляційних баз даних.

Дані мови програмування були вибрані через свою популярність і надійність. HTML –це мова гіпертекстової розмітки, тобто мова веб браузера за допомогою цієї мови ми зможемо створити каркас інтернет-платформи. CSS – це

мова за допомогою якої ми зможемо стилізувати нашу веб-сторінку, тобто описати зовнішній вигляд нашого документу, написаною мовою розмітки HTML.

JavaScript – це мова програмування, який найчастіше використовується в розробці веб-сайтів. Дана мова програмування допомагає нам управляти сценаріями перегляду сторінки. Наприклад за допомогою JavaScript ми зможемо змінити фоновий колір сторінки не перезавантажуючи її, змінити зображення, перемкнути слайд або відобразити повідомлення.

До робота з базами даних було використано серверну мову програмування MySQL. MySQL – дозволить нам використовувати доступ до даних БД. Звісно ми могли і не використовувати цю реляційну БД, а колекційну таку, як MongoDB або Cassandra, які були створенні дзеркально від реляційних баз даних для спрощення доступу до даних.

Звісно для створення серверної частини веб-додатку буде використано таку мову програмування, як Java з використанням фреймворку Spring Framework, Spring Boot і Spring Data.

Java – це золотий стандарт в веб-розробці в усьому світі. Данна мова програмування широко використовується у всьому світі, що робить її дуже популярною. Також дану мову програмування може працювати на будь-якій платформі, що робить її функціональною і бажаною. Одним з переваг це велика кількість відкритих бібліотек і фреймворків за допомогою, яких можна написати будь-яку комп’ютерну програму, додаток на телефоні або веб-додаток.

Найпопулярнішим фреймворком для розробки інтернет-платформи є Spring, який розділяється на Spring Boot – це фреймворк, який забезпечує більш простий і бистрий спосіб установки, настройки і запуска, як прости так і складних веб-додатків, тому що він потребує мінімальні налаштування для розробки.

Spring Data – це фреймворк, який використовується для спрощення доступу управління даними. Основною його перевагою є те що не потрібно писати код до і після збереження, видалення, пошуку й редагування об’єкта в БД.

Spring Framework – це дуже легкий фреймворк, який використовують, як каркас для фреймворків, оскільки він займається підтримкою інших фреймворків.

В більше широкому смислі даний фреймворк можна класифікувати, як структурний фреймворк для рішення необхідних задач і проблем.

Тому детальний аналіз предметної області, її функціональний, а також і структурних особливостей аналізу наявного програмного забезпечення, вибору мови програмування дозволяє сформувати функціональні та не функціональні вимоги до ПЗ, що розробляється.

До функцій, які будуть розроблені до ПЗ:

– Розробити онлайн магазин, який буде відповідати всім сучасним вимогами, який буде мати зручний і простий мінімалістичний інтерфейс, інтуїтивний для користувача з приємною візуалізацією товару;

– Можливість замовити річ без реєстрації на сайті. Можливість зручного здійснення замовлення, який передбачає можливість вибору бажаного розміру та перегляду стану замовлення;

– Створити зручний пошук товару за категоріями для чоловіків, жінок і розпродаж;

– Контактна інформація для зв’язку з магазином;

– Інформація про магазин;

– Інтегрувати можливість реєстрації для чатування.

У таблиці 1.2 подано опис користувачів інтернет-платформи, а також їхні можливості.

Таблиця 1.2 – Опис користувачів

Користувач	Короткий опис
Авторизований	Має можливість зв’язку з менеджером, можливість додавати товар в корзину і переглядати всі сторінки сайту.
Не авторизований	Даний користувач має обмежені можливості має лише можливість переглядати всі сторінки, але не має можливості зв’язатися з менеджером для цього йому потрібно зареєструватися.

Отже користувацький інтерфейс повинен бути інтуїтивно зрозумілий, повинен представляти структуру розміщеної на ньому інформації, що мати можливість швидко переходити до розділів та сторінок інтернет-платформи.

Так що для розробки було обрано мови програмування такі як HTML, CSS, JavaScript, тому що вони є найпопулярніші і найуживаніші для веб-розробки.

Розробка серверної частини буде відбуватись на такій мові програмування Java з використанням фреймворку Spring, а також БД MySQL.

2 ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Опис реалізації

Розробка інтернет-платформи побудована за принципом клієнтно-срверної архітектури. По суті, клієнт - це покупець, а сервер – це постачальник цих покупок. Обидві сторони репрезентують собою програму: в цій роботі клієнтом є браузер, а робота сервера реалізована мовою програмування Java з використанням фреймворку Spring, який відправляє запити до реляційної БД MySQL.

Клієнт-серверна архітектура працює між собою через інтернет з використанням мережевого протоколу HTTP використовуючи GET і POST запити. Передбачається, що клієнт відправляє запит на серверну частину сайту, а сервер його обробляє і відсилає відповідь користувачу, який використовує оновленні дані у відображеному екрані.

Звісно ж кількість потенційних покупців, які заходять на сайт і взаємодіють з ним залежить від потужності сервера і кількості запитів, які сервер може обробити.

Робота клієнт-серверної архітектури побудована за об’єктно-орієнтованим підходом програмування та принципом проектування SOLID, який використовується для розробки і дизайну програмних системи, які мають бажання тривалий час розвиватися і розширятися. Принцип SOLID дає можливість легко підтримувати систему в працюючому стані. Код розбитий на класи, інтерфейси з методами, які розділені по модулям.

2.2 Моделювання процесу роботи інтернет-платформи

На нульовому рівні процес розглядається, як блок із всіма відповідними робочими та керуючими об’єктами. Дана діаграма відображає усі потрібні дані та

інформацію, яка використовується для замовлення взуття. Діаграма нульового рівня зображена на рисунку 2.1.

Рисунок 2.1 – Контекстна модель

Перша діаграма описує функцію оброки нульового рівня. Тому функціональний блок нульового рівня можна розкласти на набір взаємозалежні під функції, які представлені на рисунку 2.2.

Приклад діаграм використання можу бути виражений низхідним процесом конкретного рівня від найбільш абстрактної і загальної концептуальної моделі діаграм з вихідною системою та логікою відповідно до системи, а потім вже до фізичної моделі.

Суть діаграм полягає в системі представлення у вигляді суб'єктів та сутностей, які взаємодіють між собою та системою. В даному випадку це може бути людина, яка взаємодіє із даною системою зовні. Інакше кажучи, кожен випадок визначає певний набір операцій, які система виконує з користувачами. Приклад діаграми представлений на рисунку 2.3.

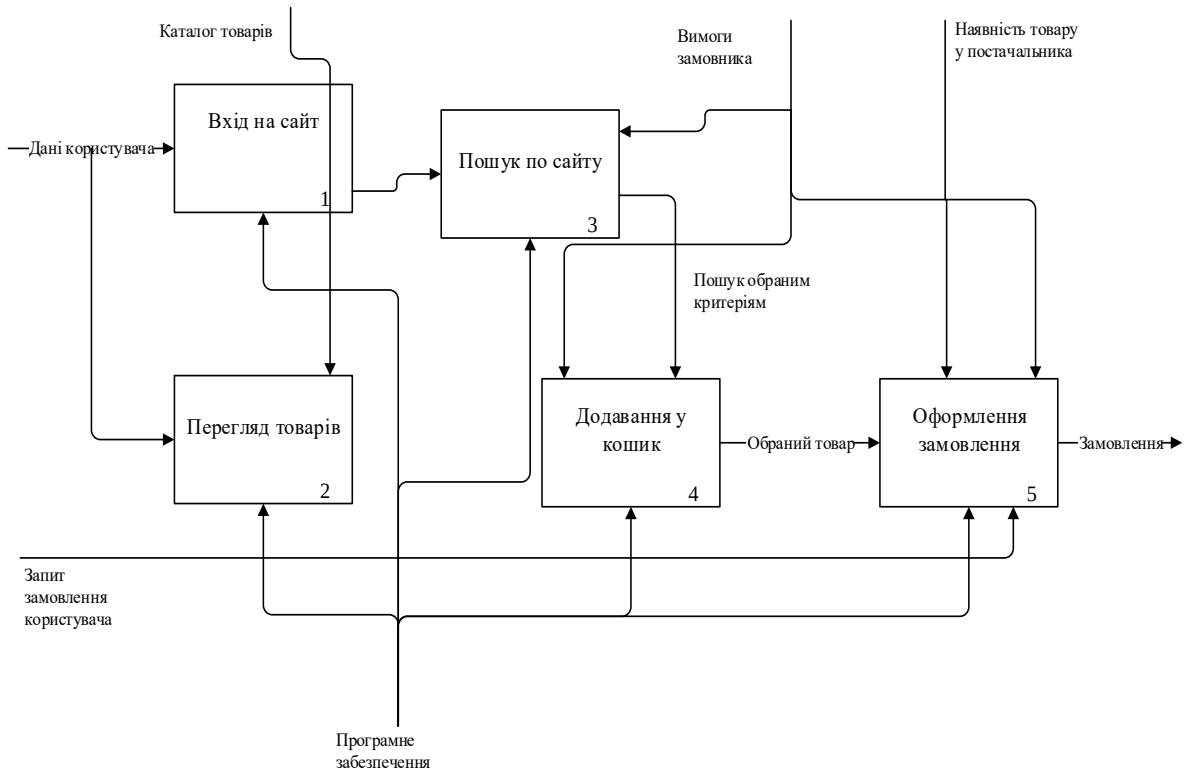


Рисунок 2.2 – Контекстна модель

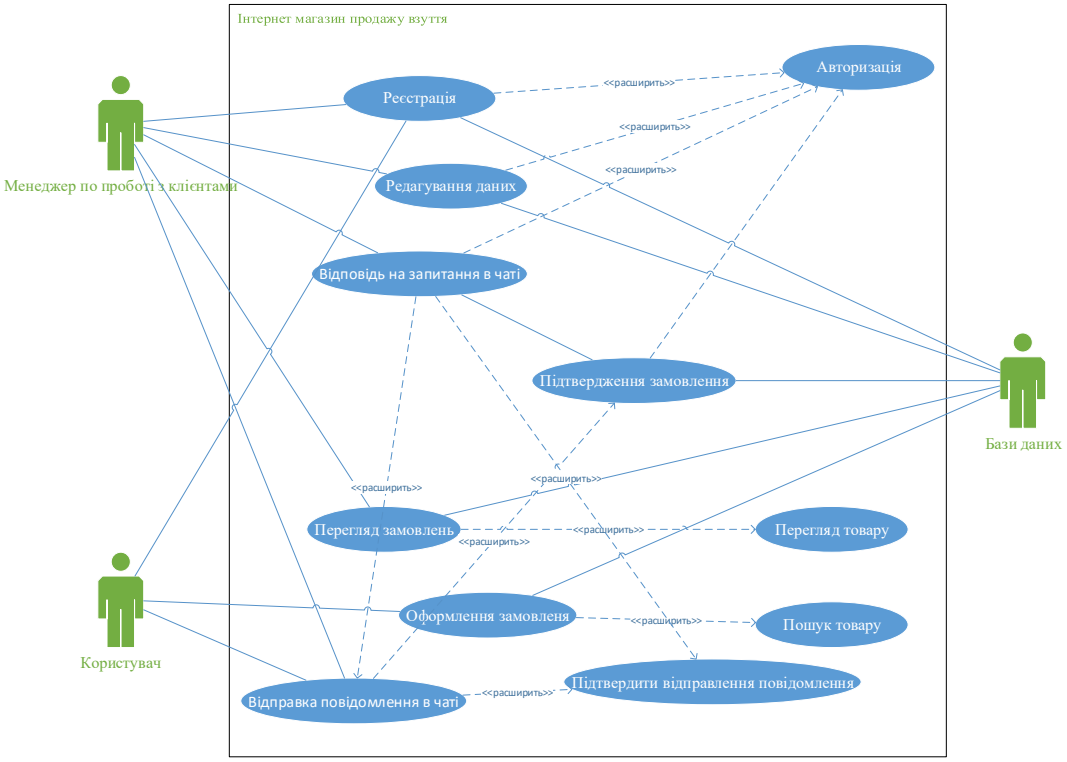


Рисунок 2.3 – Діаграма варіантів використання

2.3 Архітектура проектування та функціональна структура

Архітектура програмного забезпечення – це структура ПЗ, а також обчислювальної системи, що містить в собі програмні компоненти. Від архітектури програмного забезпечення залежить ціна на підтримку та розробка нових фіч. Тобто від архітектури залежить собівартість програмного продукту. А також можливість повторного використання коду, а також і з ним зменшення трудовитрат і часу у подальшій розробці.

Тому будь-який програмний код має взаємозалежність одних частин від інших частин. Тобто класи вимагають наявності інших класів, створені функції, методи і інтерфейси викликають інші. Тобто росте взаємозалежність стає все більшою і більшою. Звісно вимоги до проекту під-час розробки змінюються і іноді потрібно вносити миттєві корективи, тоді код стає складнішим починає ламатись, що спричиняє велику кількість проблем для повторного використання і тестування.

В якості проектування системи дуже часто застосовуються шаблони проектування. Шаблон проектування – це часто застосовувана архітектурна конструкція, що вирішує загальні проблеми під-час розробки й описує значимі рішення.

Звісно ж патерн не є закінченим проектом, який з легкістю можна перетворити в код, це опис або зразок, як вирішити завдання, щоб його можна було використати в різних ситуаціях. Однак використовувати величезну кількість шаблоні у своєму проекті не є актуально, тому що перевищує складність у виконанні. Отже потрібно підходити до вибору шаблону максимально обачно, тому що погані стилі розробки можуть призвести до поломки застосунку або його повільної роботи.

Розробка інтернет-платформ залежить не тільки від знань програміста в програмуванні, в розумінні конструкцій бази даних чи правильному середовищі розробки. Одним з найважливіших знань є знання в шаблонах проектування, які

пришвидшують написання коду, розробку логіки. Одним з найпопулярніших і часто використовуваних шаблонів для створення інтернет-платформ є MVC (Mode-View-Controller) зображений на рисунку 2.1. Це шаблон проектування, який розділяє дані додатку і управління логіки на три частини від користувацького інтерфейсу для модифікації з кожних трьох рівнів:

– Model – це центральний компонент шаблону MVC, що відображає поведінку додатку, не залежачи від інтерфейсу користувача. Зазвичай у цьому компоненті прописані методи за допомогою яких звертаються до баз даних після чого передає іншим компонентам шаблону MVC. Тому результат, отриманий даним компонентом безпосередньо передається в контролер, який має бути представленим у внутрішньому форматі програми;

– View – це компонент, який представляє інформацію користувачу, який відображається. Даний компонент дає змогу користувачу взаємодіяти з функціоналом застосунку. Отже цей компонент є шаблонізатором, який подає інформацію у вигляді HTML на основі будь-яких даних ;

– Controller – це компонент, який одержує дані та обробляє їх відповідно діям користувача після чого оновлює попередній компонент View, завдяки компоненту Model. По суті компонент Controller є мозком всіє програми, який визначає всю логіку. Отже даний модуль повинен стежити за переданими в систему даними на основі введених даних.

Отже для цього проекту було реалізовано MVC шаблон на мові програмування Java з використанням фреймворку Spring. Не дивлячись на вік мови програмування успіх полягає у кросплатформеності цієї мови програмування.

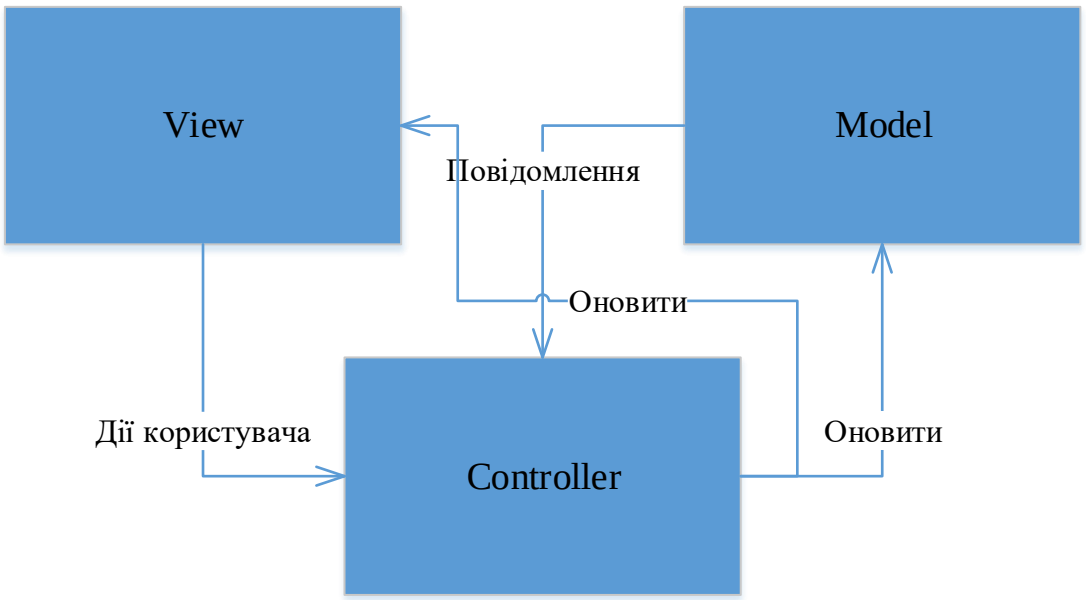


Рисунок 2.4 – Схема роботи шаблону компонентів MVC

В дипломній роботі був розроблений також чат, за допомогою, якого може здійснюватися зв'язок між менеджером по роботі з клієнтами та користувачами. Для такого типу чату найкраще підходить централізоване управління перед повідомлень між відправником і сервером, який зображений на рисунку 2.5.

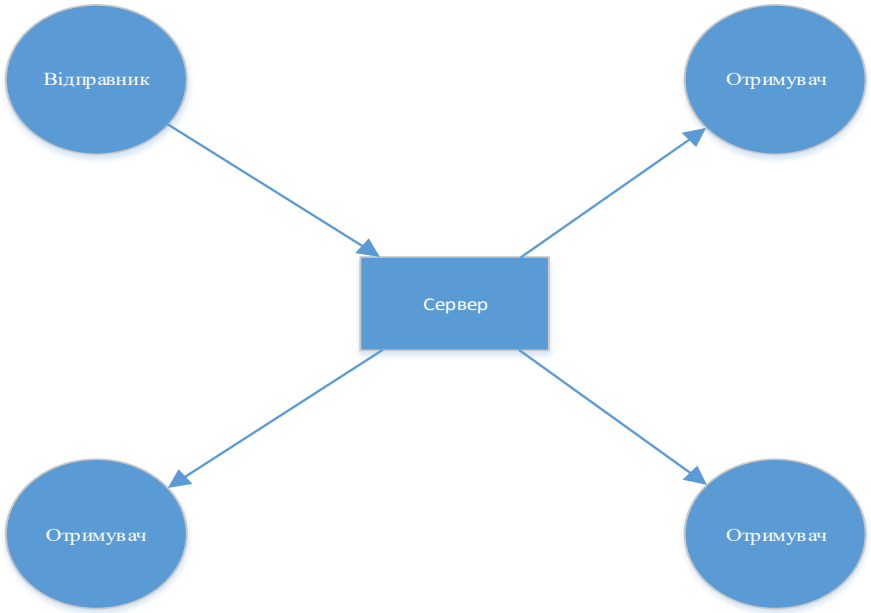


Рисунок 2.5 – Схема логіки управління передачі повідомлень

Даний підхід є інтерпретацією «клієнт – серверної» архітектури. Тобто дана технологія працює над обслуговування запитів між користувачами на сервері, тобто сервер приймає дані та поширює їх між користувачами, які під'єднанні. Продемонстрований на рисунку 2.6.

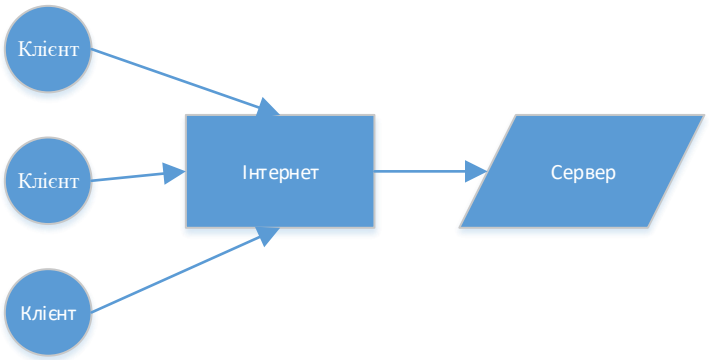


Рисунок 2.6 – Схема логіки клієнт-серверної архітектури

2.4 Модель аналізу програмного забезпечення

За допомогою діаграм варіантів використання (діаграм прецедентів) демонструються основоположні користувачі системи та завдання, які система під-час своє роботи повинна вирішувати. За допомогою діаграм послідовності демонструється послідовність всіх дій для кожного випадку розробки, що є потрібним для досягнення поставленої перед розробкою мети, продемонстрованою на рисунку 2.7 – 2.11.

Діаграма комунікації – діаграма на якій демонструється взаємодія між частинами композитної структури. На відміно від діаграма прецедентів, на діаграмі явно демонструється відношення між об'єктами. На діаграмі комунікацій моделюється взаємодія між інформацією, взятої з діаграм класів послідовності і діаграм прецедентів. До того ж діаграма комунікацій демонструє багато інформації, що і діаграма послідовності , але іншим способи подання інформації.

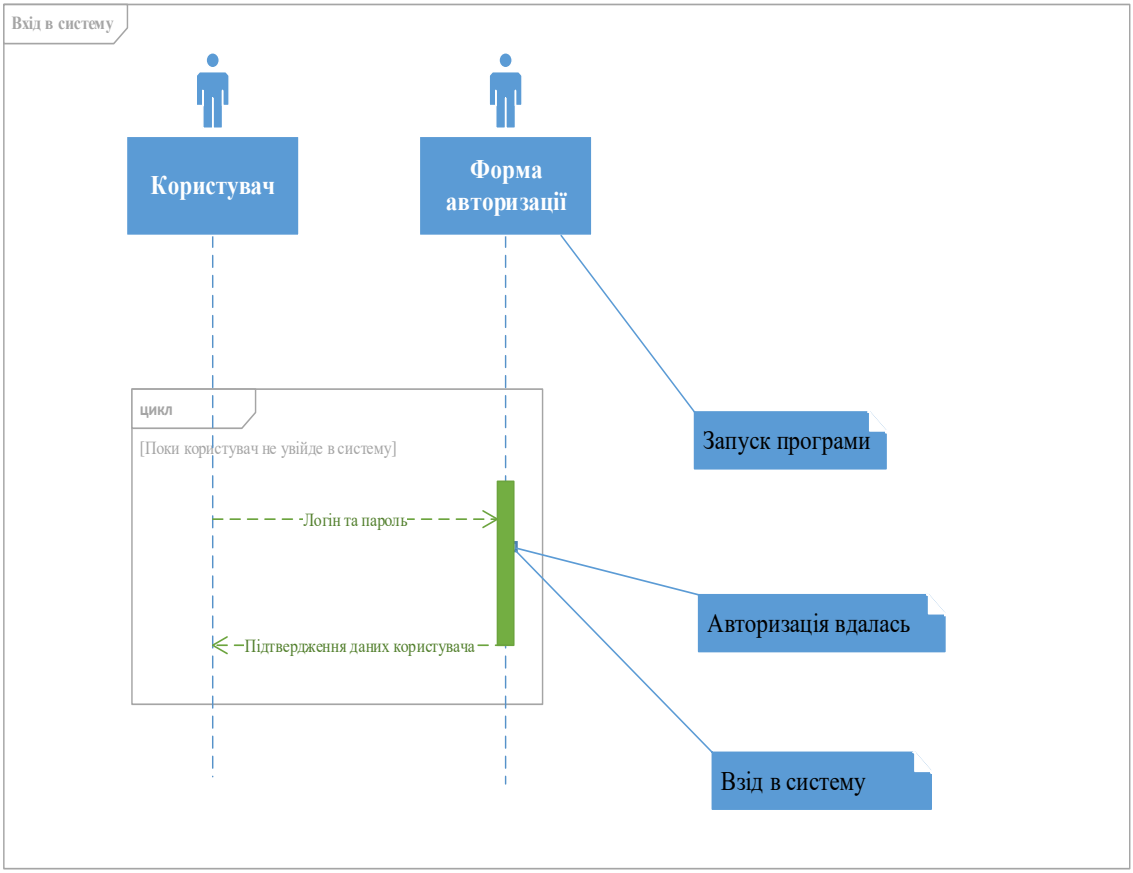


Рисунок 2.7 – Діаграма послідовності входу в систему

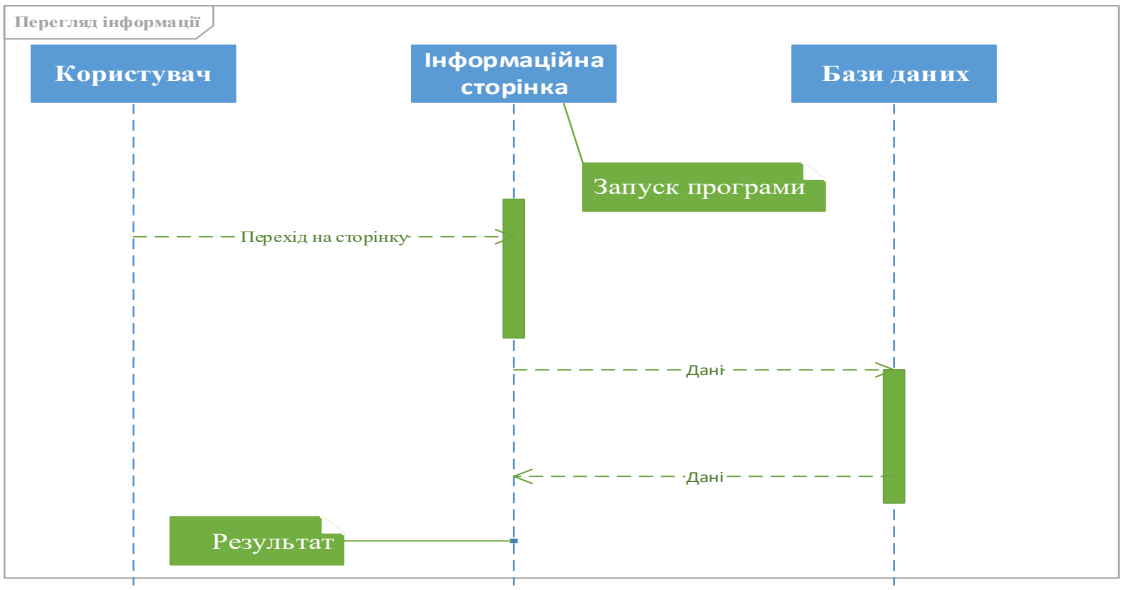


Рисунок 2.8 – Діаграма послідовності перегляду інформації

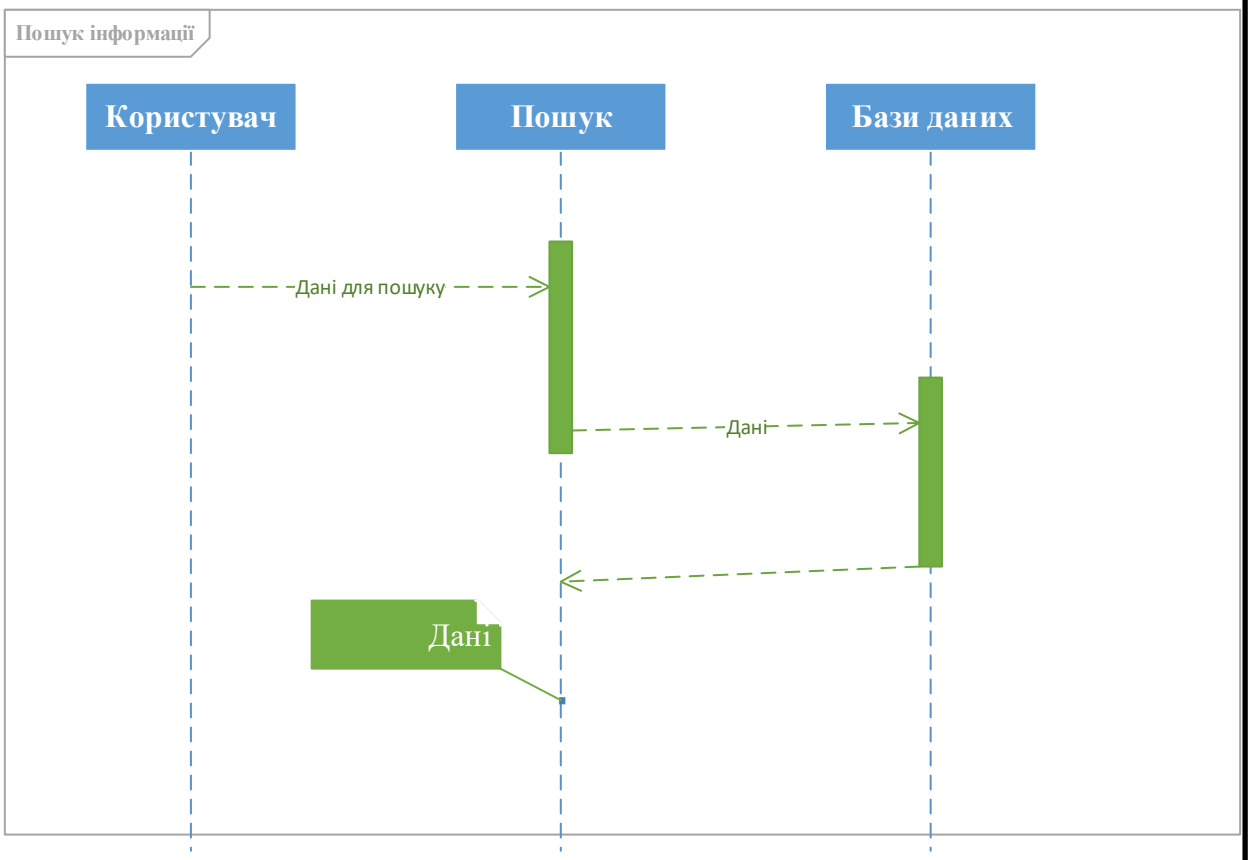


Рисунок 2.9 – Діаграма послідовності пошуку

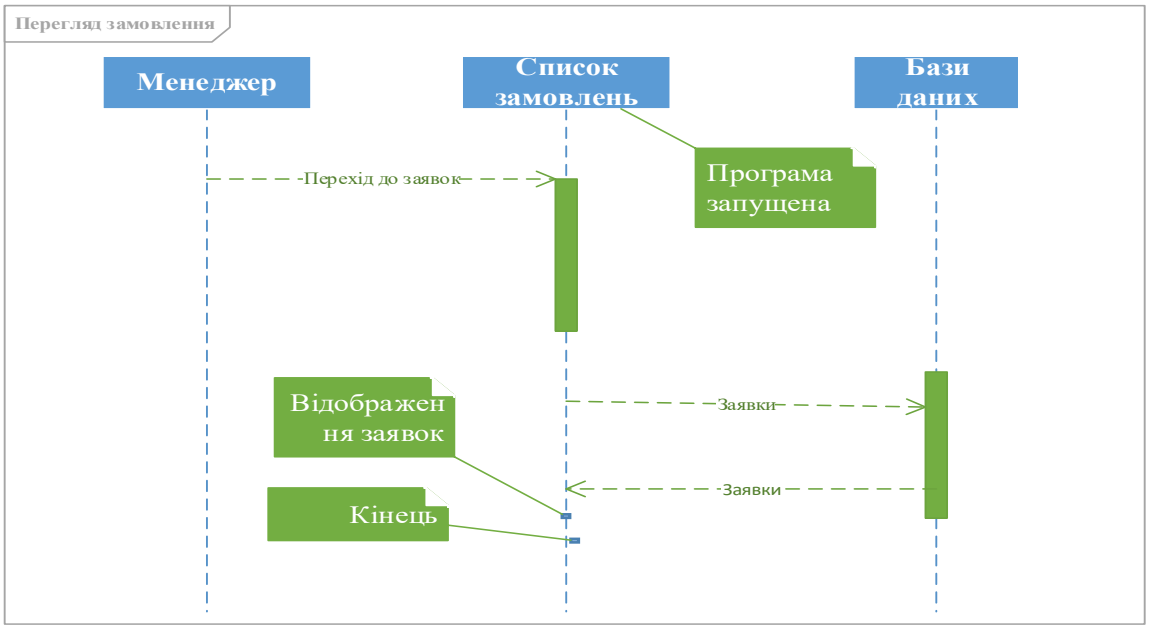


Рисунок 2.10 – Діаграма послідовності перегляду замовлення

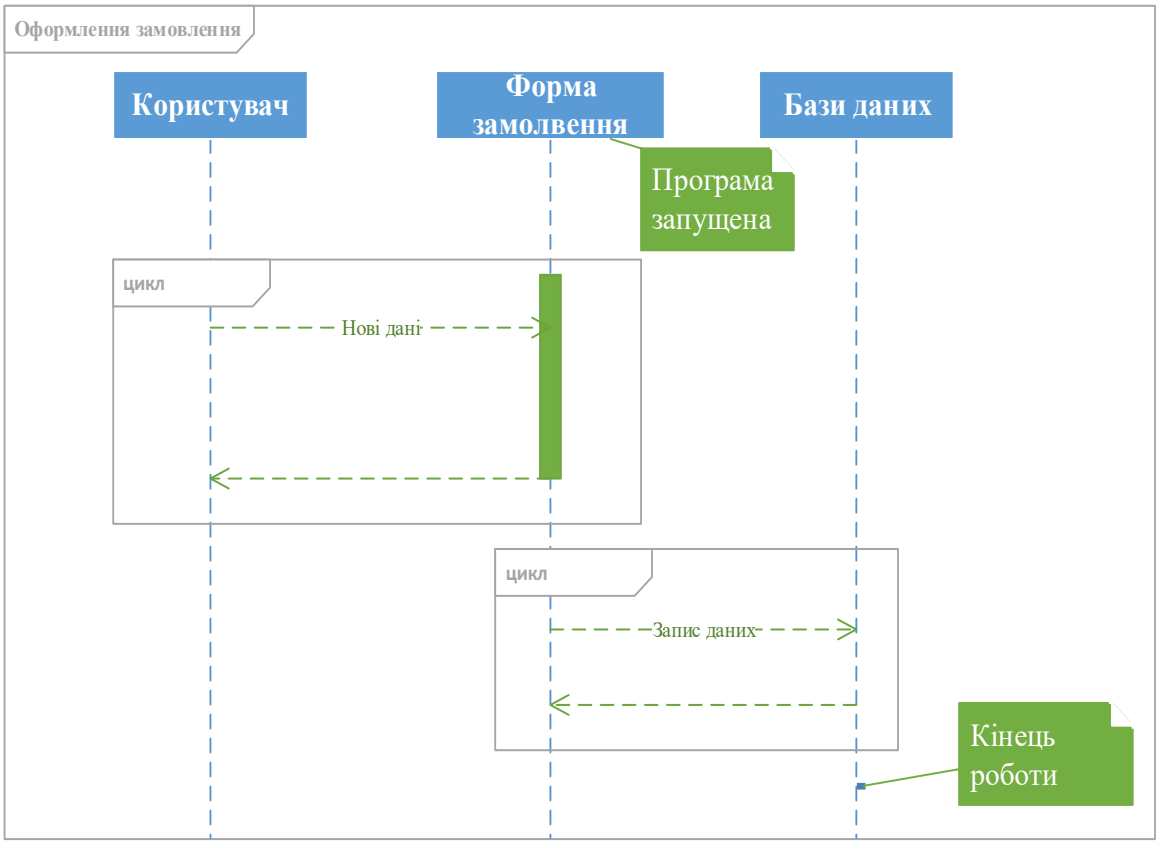


Рисунок 2.11 – Діаграма послідовності оформлення замовлення

2.5 Вибір реляційної системи керування БД

Бази даних – це набір пов’язаних між собою даних, що разом використовуються та призначені для інформаційних потреб користувача. Отже БД – це впорядкований набір різноманітних даних, який в реальному житті схожий на бібліотечну картку або медичну картку.

Бази даних класифікують за різними параметрами такими як:

– Relational Databases – це бази даних, які основані на реляційній моделі відношення, тобто бази даних мають табличний спосіб подання даних, що задаються набором таблиць;

– NoSQL Databases – це бази даних, які відрізняються подання реляційних баз даних. Головною метою розробки подібних баз даних було створення простих за дизайном баз даних, який забезпечував тонкий контроль над доступністю даних.

Велика кількість систем працюють з реляційними базами даних, тому що вони є найбільш незалежними від даних, на відміно від не реляційних баз даних.

Для вибору коректної для створення проекту реляційної бази даних було представлені на вибір такі популярні бази даних: MySQL, PostgreSQL, Maria DB, SQLite, MongoDB, Casandra DB.

Найпопулярніша у всьому світі реляційна база даних MySQL – це система керування базами даних є багато потоковою системою керування реляційних БД, яка була заснована на клієнт-серверній архітектурі, що може надати доступ для великої кількості користувачів. Часто використовують дану базу даних через зручність користування в роботі з різними сайтами та мовами програмування і величезній кількості навчальних матеріалів, які можна знайти на просторах інтернету.

MySQL в даний час є найбільш популярним програмним забезпеченням системи керування базами даних, що використовується для керування реляційною базою даних. Це програмне забезпечення бази даних з відкритим вихідним кодом, яке підтримує компанія Oracle.

MySQL - це програмне забезпечення системи управління реляційними базами даних (RDBMS), яке надає безліч можливостей, а саме:

– Дозволяє нам реалізовувати операції бази даних із таблицями, рядками, стовпцями та індексами;

– Він визначає відносини бази даних як таблиць, також відомих, як відносини;

– Забезпечує цілісний зв'язок між рядками або стовпцями різних таблиць.

– Дозволяє автоматично оновлювати індекси таблиць;

– Використовує безліч SQL-запитів та поєднує корисну інформацію з кількох таблиць для кінцевих користувачів.

MySQL слідує за роботою клієнт-серверної архітектури. Ця модель призначена для кінцевих користувачів, які називають клієнтами, для доступу до ресурсів з центрального комп'ютера, що називається сервером, за допомогою мережевих служб. Тут клієнти роблять запити через графічний інтерфейс користувача (GUI) і сервер видає бажаний результат, як тільки інструкції збігаються. Процес середовища MySQL такий самий, як і модель клієнт-сервер.

PostgreSQL – це об’єктно реляційна система управління база даних. Яка реалізована на Unix like системах. Дана база даних є сильним інструментом контролю логічних механізмів цілісності бази даних. Завдяки можливості зберігати картинки, текст, відео і запис звуки цю базу даних використовують велика кількість компаній у своїх різноманітних потребах.

PostgreSQL використовується для безпечного зберігання даних; підтримуючи найкращі практики та дозволяючи відновлювати їх при обробці запиту.

Maria DB – це система управління реляційними базами даних з відкритими вихідним кодом, яка розроблялась як заміна MySQL. Дана база даних зберігає дані в таблицях де первині ключі і зовнішні використовують для встановлення зв’язку таблицями. Maria DB представляє нам такі можливості:

– СУБД спрощує реалізацію даних з таблицями, стовпами та індексами;

– РСУБД забезпечує цілісність посилань в декількох таблицях;

– Використовується для автоматичного оновлення індексів;

– Використовується для інтерпретації SQL-запитів і операцій під-час маніпулювання або отриманням даних з таблиць.

SQLite – це вбудована система управління реляційними базами даних, тобто це автономний, серверно незалежний механізм бази даних SQL. SQLite можна використовувати безкоштовна для любих комерційних або приватних цілях. Інакше кажучи SQLite – це відкритий не потребуючий конфігурації автономний механізм реляційних баз даних призначений для вбудованих додатків.

SQLite відрізняється від інших баз даних SQL, оскільки в інших баз даних SQL, SQLite не потребує окремого серверного процесу. Він читає і записує

безпосередньо в звичайні файли на диску. Отже база даних SQLite з декількома таблицями, індексами або тригерами зберігається в одному файлі на системному диску.

MongoDB – це не реляційна база даних, а кросплатформова, колекційна тобто для дані зберігаються в документ-орієнтованому бази даних. MongoDB забезпечує високу продуктивність, високу доступність і автоматичне масштабування.

Всі сучасні застосунки потребують велику кількість даних, бистрої розробки функцій, гнучкого розгортання, а старі системи баз даних недостатньо компетентні, тому використовують MongoDB.

Casandra DB – це не реляційна база даних, яка розподіляє і масштабує бази даних. Casandra DB призначена для обробки великих об’ємів даних на більшості стандартних сервері, забезпечуючи високу доступність без відказу.

Casandra DB має розподільну архітектуру, яка має змогу обробляти велику кількість даних. Дані розміщені на різних машинах з декількома факторами реплікації для досягнення високої доступності без відказу.

Отже, порівнявши найпопулярніші бази даних було зроблено вибір в сторону реляційної бази даних MySQL за її доступність, зручність і популярність серед програмістів і користувачів, які мають рішення виникаючих проблем. Також через те, що має підтримку величезної кількості фреймворікв і мов програмування.

2.6 Проектування моделі бази даних

UML (Unified Modeling Language) – це мова програмування, яка створення для об’єктного моделювання програмного забезпечення і бізнес процесів. Вона була створена для широкого використання особливо для визначення візуалізації, проектування та документування систем.

Таблиця «user_mail» представлена двома атрибутами, один з яких також являється первинним ключем. Другий атрибут «пошта користувача» був доданий для того, щоб мати змогу зберігати по спеціальним запитам пошту користувача. Оскільки магазин представляє жіноче та чоловіче взуття – на момент створення до бази внесені ці два значення. Представлені на таблиці 2.1.

Таблиця 2.1 – Таблиця бази даних «user_mail»

Ключ	Ім’я атрибуту	Тип атрибуту	NULL/NOT NULL
PK	Id	AUTOINCREMENT	NOT NULL
	userMail	VARCAHR (111)	NOT NULL

Одна з найголовніших таблиць бази – таблиця «Товар». Вона виступає сховищем товарів з каталогу та інформації про них. Зберігає інформацію про підкатегорію товару. Представлена в таблиці 2.2.

Таблиця 2.2 – Таблиця бази даних «order_product»

Ключ	Ім’я атрибуту	Тип атрибуту	NULL/NOT NULL
PK	id	Integer	NOT NULL
	name	VARCHAR (111)	NOT NULL
	price	VARCHAR (111)	NOT NULL
	size	VARCHAR (111)	NOT NULL

Таблиця «Користувач» зберігає інформацію про зареєстрованого користувача системи. Первинний ключ – автоінкремент, який кожен раз збільшується з додаванням нових користувачів. Атрибут «Роль» є допоміжним атрибутом для розділення користувачів на покупців. Представлено в таблиці 2.3.

Таблиця 2.3 – Таблиця бази даних «usr»

Ключ	Ім’я атрибуту	Тип атрибуту	NULL/NOT NULL
PK	id	Integer	NOT NULL

Кінець таблиці 2.3

	username	VARCHAR (111)	NOT NULL
	firstname	VARCHAR (111)	NOT NULL
	lastname	VARCHAR (111)	NOT NULL
	email	VARCHAR (111)	NOT NULL
	password	VARCHAR (111)	NOT NULL
	active	VARCHAR (111)	NOT NULL
	role		NULL

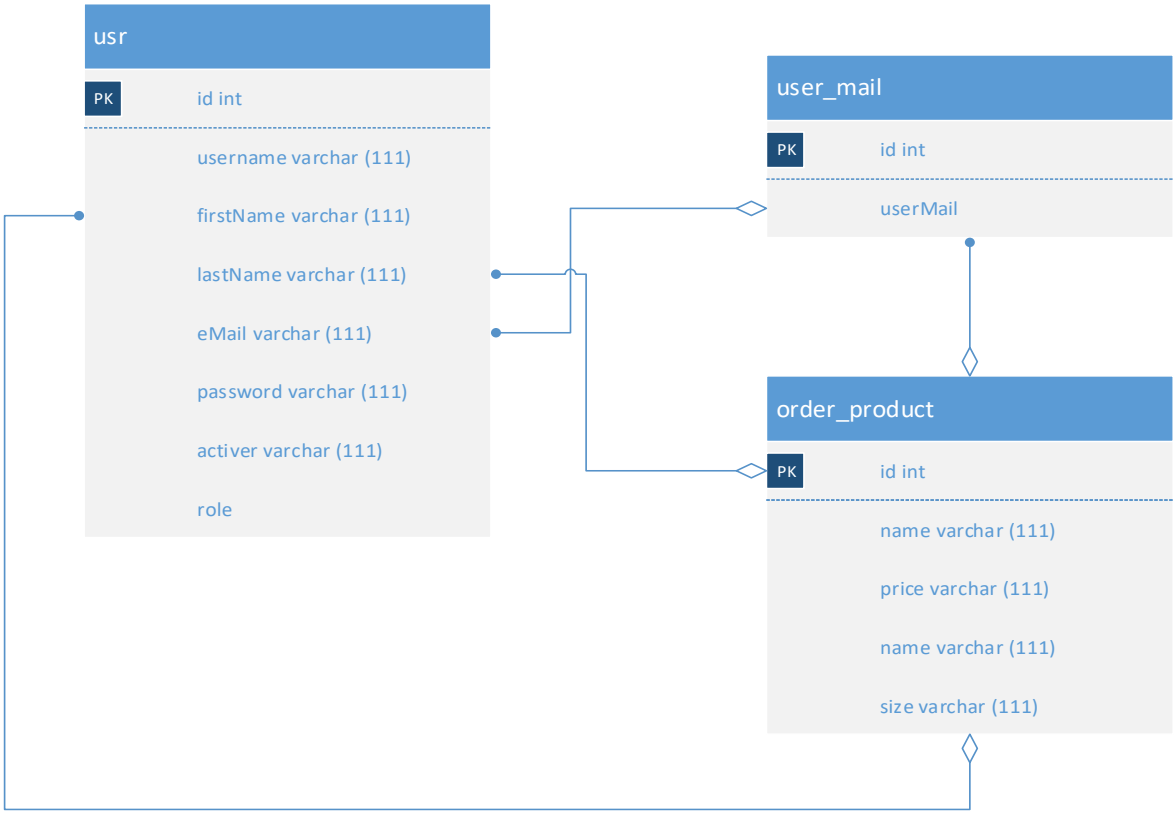


Рисунок 2.12 – Діаграма послідовності оформлення замовлення

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Реалізація розмітки інтернет-платформи

Завдяки таким мовам програмування: HTML, CSS, JavaScript ми створили інтерфейс для користувача. Процес створення інтерфейсу власноруч особливо нічим не відрізняється від того, якщо б ми використовували такий конструктор Bootstrap Studio. Для створення даного проекту не потрібно використовувати такі засоби, тому що інтерфейс не є складним в якому немає великої кількості активних компонентів, тому для розробки було використано Visual Studio Code, як редактор коду.

Розмітка сторінки головної інтернет-платформи (файл index.html) виглядає наступним чином:

```
<!DOCTYPE html>
<html                lang="en"                xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/main_page.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <div class="text">
                Доброго дня аби отримати доступ, до магазину <br>
                вам потрібно авторизуватись. У випадку, якщо <br>
                ви неможете авторизуватись вам потрібно <br>
                зареєструватись. Для того щоб зареєструватись <br>
                нажміть кнопку реєстрація. Для того аби увійти <br>
                нажміть на кнопку авторизуватись.
```

```
            </div>
            <a th:href="@{/main}">
                <input type="button" value="Авторизуватись" class="login">
            </a>
            <a href="/registration">
                <input type="button" value="Зареєструватися" class="registration">
            </a>
        </div>
    </body>
</html>
```

Також для можливості реєстрації було створено сторінку реєстрації (файл registration.html), який виглядає так:

```
<!DOCTYPE html>
<html                lang="en"                xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/registration.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <form     th:action="@{/registration}"     class="popup_registration"
method="post">
                <label>
                    <input type="text" name="firstName">
                    <div class="label_text">
                        Ім'я
                    </div>
                </label>
                <label>
                    <input type="text" name="lastName">
                    <div class="label_text">
```

```
                    Фамілія
                </div>
            </label>
            <label>
                <input type="text" name="eMail">
                <div class="label_text">
                    Пошта
                </div>
            </label>
            <label>
                <input type="text" name="username">
                <div class="label_text">
                    Логін
                </div>
            </label>
            <label>
                <input type="password" name="password">
                <div class="label_text">
                    Пароль
                </div>
            </label>
            <button type="submit">Зареєструватися</button>
        </form>
    </div>
</body>
</html>
```

Також для можливості входу користувача в магазин було створено сторінку реєстрації (файл login.html), який виглядає так:

```
<!DOCTYPE html>
<html                lang="en"                xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/login.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
```

```
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <form th:action="@{/login}" class="popup_registration" method="post">
                <label>
                    <input type="text" name="username"/>
                    <div class="label_text">
                        Логін
                    </div>
                </label>
                <label>
                    <input type="password" name="password"/>
                    <div class="label_text">
                        Пароль
                    </div>
                </label>
                <input type="submit" value="Увійти"/>
            </form>
        </div>
    </body>
</html>
```

Далі наведено сторінку чата (файл chat.html), який виглядає так:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-
scale=1.0">
    <title>Магазин взуття чат</title>
    <link rel="stylesheet" href="/css/chat.css" />
</head>
<body>
<noscript>
    <h2>Sorry! Your browser doesn't support Javascript</h2>
</noscript>

<div id="username-page">
    <div class="username-page-container">
        <h1 class="title">Введіть своє ім'я</h1>
        <form id="usernameForm" name="usernameForm">
```

```
            <div class="form-group">
                <input type="text" id="name" placeholder="Введіть своє ім'я"
autocomplete="off" class="form-control" />
            </div>
            <div class="form-group">
                <button type="submit" class="accent username-submit">Початок
спілкування</button>
            </div>
        </form>
    </div>
</div>

<div id="chat-page" class="hidden">
    <div class="chat-container">
        <div class="chat-header">
            <h2>Менеджер по роботі з клієнтами магазину взуття</h2>
        </div>
        <div class="connecting">
            Підключення...
        </div>
        <ul id="messageArea">
        </ul>
        <form id="messageForm" name="messageForm">
            <div class="form-group">
                <div class="input-group clearfix">
                    <input type="text" id="message" placeholder="Введіть
повідомлення..." autocomplete="off" class="form-control"/>
                    <button type="submit" class="primary">Відправити</button>
                </div>
            </div>
        </form>
    </div>
</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/sockjs-
client/1.1.4/sockjs.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/stomp.js/2.3.3/stomp.min.js"></script
>
<script src="/js/chat.js"></script>
</body>
</html>
```

3.2 Реалізація логіки взаємодії з додатком

Під-час детальної розробки архітектури програмного забезпечення і проектування було вирішено, що вся серверна частина додатку буде працювати на мові програмування Java, з використанням фреймворку Spring Framework. Так було визначено, що працюватиме з реляційною базою даних MySQL.

Основним завданням було розробити інтернет-платформу інтуїтивно зрозумілою для користувача, де буде надаватись структуровано розміщена інформація, та можливість швидко переходити між розділами сторінок та виконувати певні функції.

В першу чергу потрібно налаштувати точку і брокер повідомлень. Все це зроблено в пакеті конфігурацій config. Далі фрагмент коду його реалізації

```
@Configuration
@EnableWebSocketMessageBroker
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws").withSockJS();
    }

    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry) {
        registry.setApplicationDestinationPrefixes("/app");
        registry.enableSimpleBroker("/topic");
    }
}
```

В першому методі ми реєструємо кінцеву точку, яка користувач буде використовувати, щоб підключитись до нашого Websocket – серверу для браузері, який не підтримує Websocket.

STOMP – це Simple Text Oriented Messaging Protocol. Він потрібен для обміну повідомленнями , який задають формат і правила обміну. Справа в тому, що сама по собі WebSocket не дає таких речей (вищого рівня), як відправлення

повідомлень користувачами, підписаним на тему, або відправлення повідомлень вірусу користувачів.

У другому методі configureMessageBroker() ми налаштовуємо брокер повідомлень, який вестиметься повідомленнями від одного клієнта до іншого.

У першому рядку ми говоримо, що повідомлення адресатів (куди відправлені) починаються з "/app", мають бути спрямовані на методи, що займаються обробкою повідомлень.

У другому рядку ми говоримо, що повідомлення, адресатів починається з "/topic", повинні бути направлені в брокер повідомлень. Брокер надсилає всім дітям повідомлення, підписані на тему.

Тепер створимо пакет controller та клас ChatController. У класі ChatController є методи, які відповідають отримання повідомлення від одного клієнта і трансляцію його всім іншим. Додавання користувача та його повідомлення транслюються всім, хто підключений до чату:

```
@Controller
public class ChatController {
    @MessageMapping("/chat.sendMessage")
    @SendTo("/topic/public")
    public ChatMessagePojo sendMessage(@Payload ChatMessagePojo chatMessagePojo) {
        return chatMessagePojo;
    }

    @MessageMapping("/chat.addUser")
    @SendTo("/topic/public")
    public ChatMessagePojo addUser(@Payload ChatMessagePojo chatMessagePojo,
SimpMessageHeaderAccessor headerAccessor) {
        headerAccessor.getSessionAttributes().put("username",
chatMessagePojo.getSender());
        return chatMessagePojo;
    }
}
```

Як ви пам'ятаєте, у конфігурації ми вказали, що всі повідомлення від клієнтів, надіслані на адресу, що починається з /app, будуть перенаправлені у

відповідні методи. Йшлося саме методи, анотовані @MessageMapping. Наприклад, повідомлення, направлене за адресою /app/chat.sendMessage буде перенаправлено метод sendMessage(). Наприклад, повідомлення, направлене за адресою/app/chat.addUser буде перенаправлено в метод addUser().

Тут ми слухаємо події з'єднання з сервером та від'єднання. Це потрібно для того, щоб логувати ці події та передавати в чат на загальний огляд. Так усі бачать, коли хтось заходить у чат і виходить із нього:

```
@Component
public class WebSocketEventListener {
    private static final Logger logger = LoggerFactory.getLogger(WebSocketEventListener.class);
    @Autowired
    private SimpMessageSendingOperations messagingTemplate;

    @EventListener
    public void handleWebSocketConnectListener(SessionConnectedEvent event) {
        logger.info("Отримане нове підключення до веб-сокета!");
    }
    @EventListener
    public void handleWebSocketDisconnectListener(SessionDisconnectEvent event) {
        StompHeaderAccessor headerAccessor = StompHeaderAccessor.wrap(event.getMessage());

        String username = (String) headerAccessor.getSessionAttributes().get("username");
        if (username != null) {
            logger.info("Користувач відключився : " + username);

            ChatMessagePojo chatMessagePojo = new ChatMessagePojo();
            chatMessagePojo.setType(ChatMessagePojo.MessageType.LEAVE);
            chatMessagePojo.setSender(username);

            messagingTemplate.convertAndSend("/topic/public", chatMessagePojo);
        }
    }
}
```

Spring Data надає набір готових реалізацій для створення шару, що забезпечує доступ до БД. Інтерфейс JpaRepository надає набір стандартних методів до роботи з БД:

```
@Repository
public interface UserRepository extends CrudRepository<User, Integer> {
    User findByUsername(String username);
}
```

Просто створивши інтерфейс і успадкувавши JpaRepository, можна виконувати стандартні запити до БД.

Для сторінок, які не обробляються сервером, а просто повертають сторінку, мапінг можна налаштувати в конфігурації. Сторінка login обробляється Spring Security контролером за замовчуванням, тому окремий контролер для неї не потрібен:

```
@Configuration
public class MvcConfig implements WebMvcConfigurer {

    public void addViewControllers(ViewControllerRegistry registry) {
        registry.addViewController("/login").setViewName("login");
    }
}
```

Метод addViewControllers()(який перевизначає метод з тим самим ім'ям у WebMvcConfigurer) додає контролер представлення. Контролер уявлення посилається інше уявлення з ім'ям login.

Наступна конфігурація безпеки гарантує, що користувач зможе бачити головну сторінку, що пройшли перевірку:

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
    @Autowired
    private DataSource dataSource;
```

```
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http

                .authorizeRequests()
                .antMatchers("/", "/registration").permitAll()
                .anyRequest().authenticated()
                .and()
                .formLogin()
                .loginPage("/login")
                .permitAll()
                .and()
                .logout()
                .permitAll();
    }

    @Override
    public void configure(WebSecurity web) {
        web.ignoring()
                .antMatchers(
                        "/css/**", "/fonts/**",
                        "/img/**");
    }

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        auth.jdbcAuthentication()
                .dataSource(dataSource)
                .passwordEncoder(NoOpPasswordEncoder.getInstance())
                .usersByUsernameQuery("select username, password, active from usr
where username=?")
                .authoritiesByUsernameQuery("select u.username, ur.roles from usr
u inner join user_role ur on u.id=ur.user_id where u.username=?");

    }
}
```

Клас WebSecurityConfiganотований @EnableWebSecurity, щоб включити підтримку веб-безпеки Spring Security та забезпечити інтеграцію Spring MVC. Він

також розширює WebSecurityConfigurerAdapteri перевизначає кілька своїх методів, щоб встановити деякі особливості конфігурації веб-безпеки.

Метод configure(HttpSecurity) визначає, які шляхи URL мають бути захищені, а які ні. Зокрема, шляхи /і /registration настроєні так, щоб не вимагати автентифікації. Всі інші шляхи мають бути автентифіковані.

Коли користувач успішно входить до системи, він перенаправляється на раніше запитану сторінку, яка потребує аутентифікації. Існує /login сторінка (вказується loginPage()), і всім дозволено її переглядати.

Метод userDetailsService()налаштовує сховище користувача в пам'яті з одним користувачем. Цьому користувачеві присвоюється ім'я користувача user, пароль password і роль USER.

Також потірбно створити службу реєстрації:

```
@Controller
public class RegistrationController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("/registration")
    public String registration() {
        return "registration";
    }

    @PostMapping("/registration")
    public String addUser(User user, Map<String, Object> model) {
        User userFromDB = userRepository.findByUsername(user.getUsername());
        if (userFromDB != null) {
            return "registration";
        }
        user.setActive(true);
        user.setRoles(Collections.singleton(Role.USER));
        userRepository.save(user);
        return "redirect:/login";
    }
}
```

У результаті було створено веб-додаток, в якому ми можемо керувати доступом користувачів до сторінок сайту застосовуючи ролі. При реєстрації нового користувача додаємо базову для всіх роль User.

3.3. Структура проекту

Після того, яке все розроблено і налагоджено маємо змогу розглянути структуру проекту backend частини, яка зображена рисунку 3.1, а також frontend, яка зображена на рисунку 3.2.

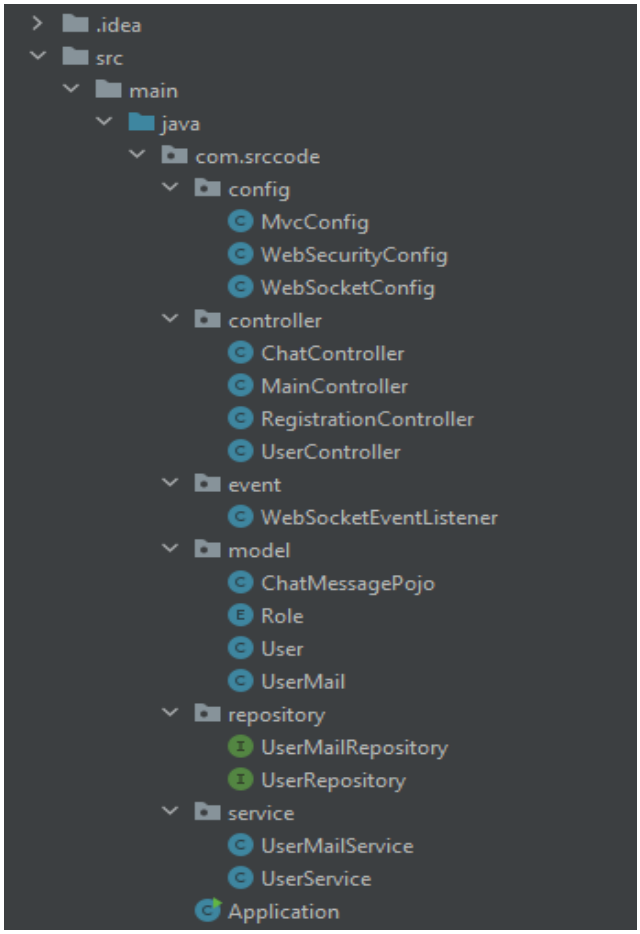


Рисунок 3.1 – Структура backend проекту

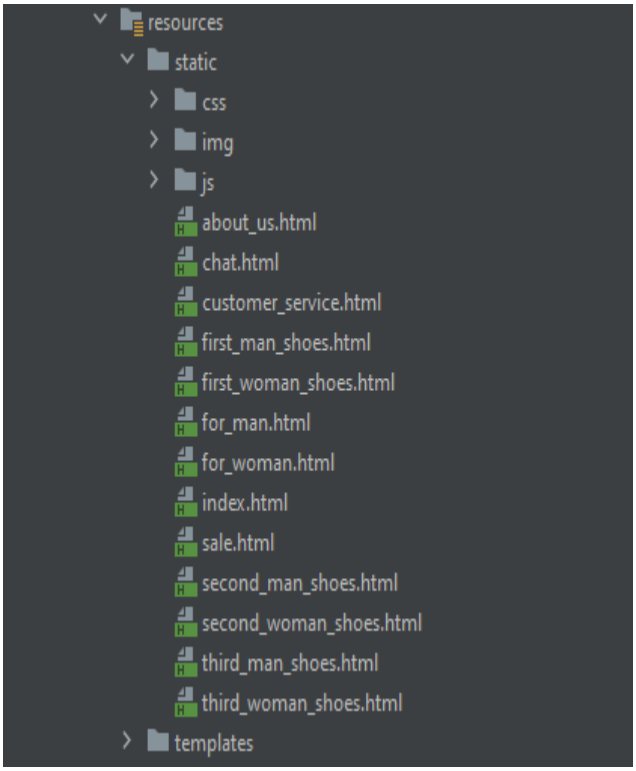


Рисунок 3.2 – Структура frontend проекту

Проект складається з таки каталогів:

com/srccode – каталог, в якому лежить головний клас всього проекту для запуску всіх можливих компонентів.

com/srcocode/config –це каталог, в якому знаходяться всі конфігураційні класи програми, які призначення для коректної роботи тих чи інших модулі програми.

com/srccode/controller – це каталог, в якому знаходяться всі контролери, які відповідають за обробку запитів, модулів і повертає представлення в якості відповідді.

com/srccode/event –це каталог, який відповідає за класи, які обробляють події.

com/srccode/model – це каталог, який містить всі можливі моделі, які ми використовуємо для з'єднання з базою даних MySQL.

com/srccode/model – це каталог, який містить в собі інтерфейси програми, які використовує Java Persistence API для взаємодії з нею.

com/srccode/service – це каталог, який містить сервіси програми, які розробляються самостійно для певних задач, які мають свій власний процес, за допомогою, яких можливо облегшити модель додатку.

resource/static – забезпечує можливість зберігати всі статичні файли, які не будуть оброблятись.

resource/templates – це папка, яка містить в собі файли для відображення в браузері та взаємодії між ними.

Всі ці каталоги було створенні для структурованої роботи програмного забезпечення.

4 ТЕСТУВАННЯ ІНТЕРЕНТ-ПЛАТФОРМИ

4.1 Аналіз методів тестування

Тестування програмного забезпечення є широко застосовуваною технологією, тому що перед розгортанням необхідно протестувати кожне програмне забезпечення.

Тестування програмного забезпечення – це процес визначення правильності програмного забезпечення шляхом розгляду всіх його атрибутів (надійність, масштабованість, переносимість, можливість повторного використання, зручність використання) та оцінки виконання компонентів програмного забезпечення для пошуку програмних помилок, помилок чи дефектів.

Тестування програмного забезпечення забезпечує незалежний погляд і мету програмного забезпечення та дає впевненість у придатності програмного забезпечення. Він включає тестування всіх компонентів необхідних послуг, щоб підтвердити, чи задовольняють вони зазначеним вимогам чи ні. У процесі надається клієнту інформація про якість програмного забезпечення.

Тестування є обов'язковим, тому що це буде небезпечна ситуація, якщо програмне забезпечення вийде з ладу через відсутність тестування. Таким чином, без тестування програмне забезпечення не може бути розгорнуте для кінцевого користувача.

Тестування - це група методів для визначення правильності програми за заздалегідь заданим сценарієм, але тестування не може знайти всі дефекти програми. Основна мета тестування – виявити збої у додатку, щоб їх можна було виявити та виправити. Це не демонструє, що продукт працює належним чином у всіх умовах, а лише те, що він не працює у деяких конкретних умовах.

Тестування забезпечує порівняння, яке порівнює поведінку та стан програмного забезпечення з механізмами, тому що проблема може бути розпізнана механізмом. Механізм може включати попередні версії одного і того

ж зазначеного продукту, сумісні продукти та інтерфейси очікуваного призначення, відповідні стандарти або інші критерії, але не обмежуючись ними.

Тестування включає перевірку коду, а також виконання коду в різних середовищах, умовах, а також всі аспекти перевірки коду. У поточному сценарії розробки програмного забезпечення група тестування може бути відокремлена від групи розробників, щоб інформація, отримана в результаті тестування, могла використовуватися для коригування процесу розробки програмного забезпечення.

Успіх програмного забезпечення залежить від прийняття його цільової аудиторією, простого графічного інтерфейсу користувача, потужного функціонального навантажувального тесту і т. д. Наприклад, аудиторія банківської справи повністю відрізняється від аудиторії відеоігор. Отже, коли організація розробляє програмний продукт, вона може оцінити, чи буде цей програмний продукт корисним для його покупців та іншої аудиторії.

Тестування програмного забезпечення – це процедура впровадження програмного забезпечення або програми для виявлення дефектів чи помилок. При тестуванні програми або програмного забезпечення нам необхідно слідувати деяким принципам, щоб зробити наш продукт вільним від дефектів, і це також допомагає інженерам-випробувачам тестувати програмне забезпечення з їхніми зусиллями та часом. Тут, у цьому розділі, ми дізнаємося про сім основних принципів тестування програмного забезпечення.

Принципи тестування:

– Тестування демонструє наявність дефектів;

– Вичерпне тестування неможливе;

– Раннє тестування;

– Кластеризація дефектів;

– Парадокс пестицидів;

– Тестування залежить від контексту;

– Помилка про відсутність помилок.

Тестування демонструє наявність дефектів при тестуванні ми можемо лише визначити наявність помилок у програмі або програмному забезпеченні. Основна мета проведення тестування — визначити кількість невідомих помилок за допомогою різних методів та технік тестування, оскільки весь тест має бути простежуваним до вимог замовника, а це означає, що потрібно знайти будь-які дефекти, які можуть призвести до невідповідності продукту до вимог. потреби клієнта.

Виконуючи тестування будь-якої програми, ми можемо зменшити кількість помилок, що не означає, що програма не містить дефектів, тому що інколи програмне забезпечення здається вільним від помилок при виконанні кількох типів тестування. Але під час розгортання на робочому сервері якщо кінцевий користувач зіткнеться з тими помилками, які не були виявлені в процесі тестування.

Вичирпне тесутвання неможливе іноді здається дуже складним протестувати всі модулі та їх функції з ефективними та неефективними комбінаціями вхідних даних протягом усього процесу тестування.

Отже, замість проведення вичерпного тестування, оскільки воно вимагає безмежних визначень, більшість важкої роботи виявляється безрезультатною. Таким чином, ми можемо завершити цей тип варіацій відповідно до важливості модулів, тому що термін продукту не дозволяє нам виконувати такі сценарії тестування.

Раннє тестування тут раннє тестування означає, що всі дії з тестування повинні починатися на ранніх стадіях етапу аналізу вимог життєвого циклу розробки програмного забезпечення для виявлення дефектів, тому що якщо ми виявимо помилки на ранній стадії, вони будуть виправлені на початковій стадії, що можуть коштувати нам набагато менше порівняно з тими, що будуть визначені на майбутній стадії процесу тестування.

Для проведення тестування нам знадобляться документи технічного завдання; отже, якщо вимоги визначені неправильно, це можна виправити безпосередньо, а чи не іншому етапі, яким то, можливо етап розробки.

Кластеризація дефектів визначила, що на протязі всього процесу тестування ми можемо виявити кількість помилок, які корелюють із невеликою кількістю модулів. У нас є різні причини цього, наприклад, модулі можуть бути складними; кодуюча частина може бути складною і т.д.

Ці типи програмного забезпечення або програми будуть дотримуватися принципу Парето, який свідчить, що ми можемо визначити, що прибл. Вісімдесят відсотків складності є у 20 відсотках модулів. За допомогою цього ми можемо знайти невизначені модулі, але цей метод має свої труднощі, якщо одні і ті ж тести виконуються регулярно, тому один і той же тест не зможе виявити нові дефекти.

Парадокс пестицидів цей принцип визначив, що якщо ми виконуємо один і той же набір тестів знову і знову протягом певного часу, такі тести не зможуть знайти нові помилки в програмному забезпеченні або додатку. Щоб подолати ці парадокси пестицидів, дуже важливо часто переглядати усі тестові приклади. І нові та різні тести необхідні для реалізації кількох частин програми або програмного забезпечення, що допомагає нам знайти більше помилок.

Тестування — це контекстно-залежний принцип, згідно з яким ми маємо кілька полів, таких як веб-сайти електронної комерції, комерційні веб-сайти тощо, які доступні на ринку. Існує певний спосіб протестувати комерційний сайт, а також веб-сайти електронної комерції, тому що кожен додаток має власні потреби, функції і функціональність. Щоб перевірити цей тип програми, ми скористаємося за допомогою різних видів тестування, різних методів, підходів та кількох методів. Тому тестування залежить від контексту програми.

Після того, як програма буде повністю протестована і перед випуском не буде виявлено жодної помилки, ми можемо сказати, що програма на 99% не містить помилок. Але є ймовірність, що при тестуванні додатка з неправильними вимогами виявлені недоліки та виправлення їх на заданий термін не допоможуть, оскільки тестування проводиться за неправильною специфікацією, яка не стосується вимог замовника. Відсутність помилкової помилки означає, що виявлення та виправлення помилок не допоможе, якщо програма непрактична і не здатна виконати вимоги та потреби клієнта.

Процедура тестування програмного забезпечення також відома як software testing life cycl, яка включає етапи процесу тестування. Процес тестування виконується добре спланованим та систематичним чином. Всі заходи проводяться для покращення якості програмного продукту.

Як відомо, тестування програмного забезпечення – це процес аналізу функціональності програми відповідно до вимог замовника.

Якщо ми хочемо переконатися, що наше програмне забезпечення не містить помилок або стабільно, ми повинні виконувати різні типи тестування програмного забезпечення, тому що тестування є єдиним методом, який робить нашу програму вільною від помилок. Типи тестування зображений на рисунку 4.1.

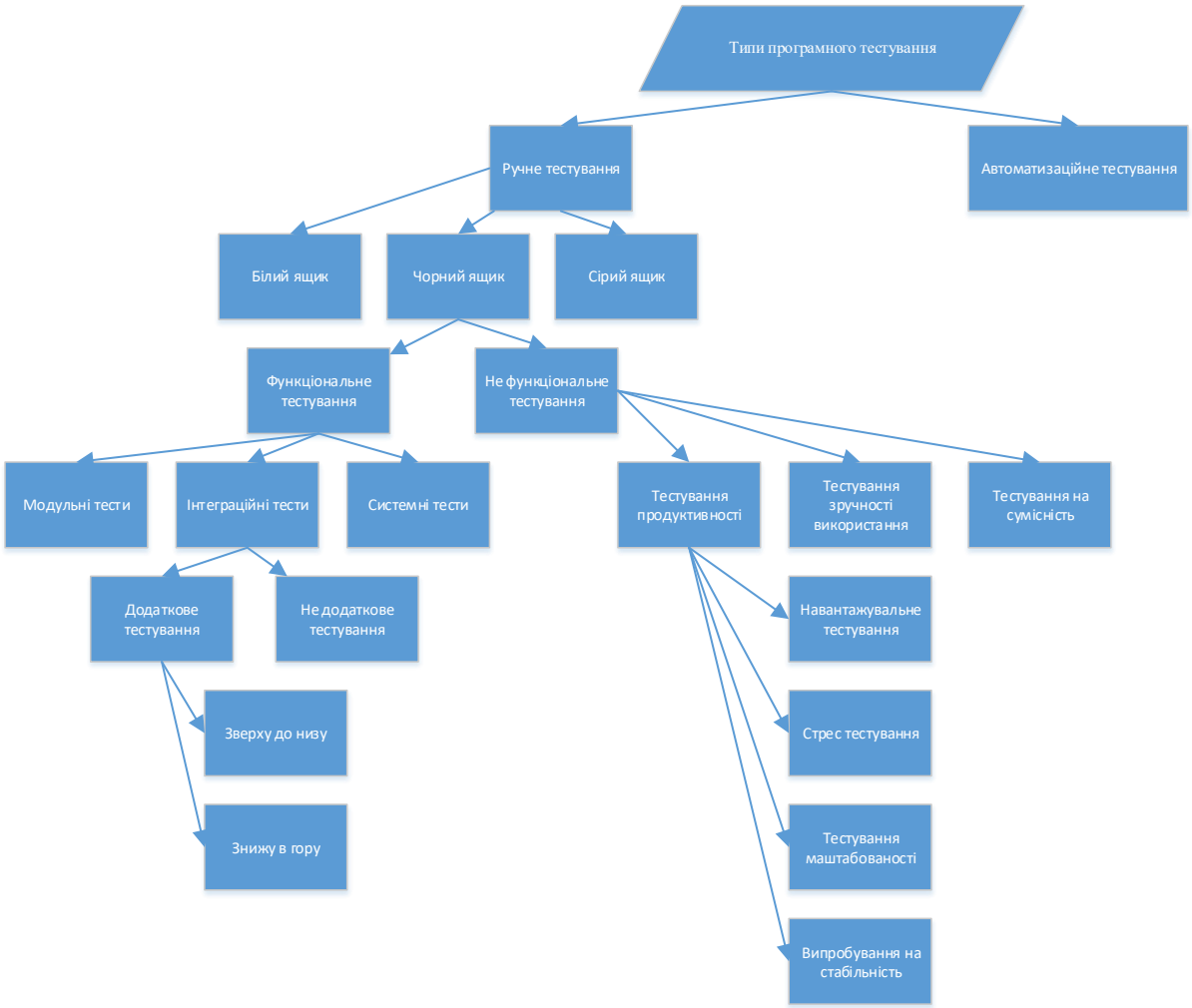


Рисунок 4.1 – Модель типів тестування

Категоризація тестування програмного забезпечення є частиною різних заходів із тестування, таких як стратегія тестування, результати тестування, певна мета тестування тощо. д. А тестування програмного забезпечення це виконання програмного забезпечення для пошуку дефектів.

Щоб почати тестування, у нас має бути вимога, готовий додаток, необхідні ресурси. Щоб зберегти підзвітність, ми маємо призначити відповідний модуль різним інженерам-випробувачам.

Ручне тестування – це процес тестування програмного забезпечення, при якому тестові приклади виконуються вручну без використання будь-яких автоматизованих інструментів. Усі тестові випадки виконуються тестувальником вручну відповідно до точки зору кінцевого користувача. Він гарантує, чи працює програма, як зазначено в документі з вимогами, чи ні. Тестові випадки плануються та реалізуються таким чином, щоб завершити майже 100% програмної програми. Звіти про тест-кейси також генеруються вручну.

Ручне тестування — один із фундаментальних процесів тестування, оскільки він може знайти як видимі, так і приховані дефекти програмного забезпечення. Різниця між очікуваним виходом та виходом, заданим програмним забезпеченням, визначається як дефект. Розробник виправив дефекти та передав тестувальнику для повторного тестування.

Перед автоматичним тестуванням ручне тестування є обов'язковим для кожного нового програмного забезпечення. Це тестування вимагає великих зусиль і часу, але гарантує відсутність помилок у програмному забезпеченні. Ручне тестування вимагає знання методів ручного тестування, але не інструментів автоматичного тестування.

Для ручного тестування застосовуються різні способи. Кожен метод використовується відповідно до своїх критеріїв тестування. Типи ручного тестування наведені нижче:

– Тестування білого ящика;

– Тестування чорного ящика;

– Тестування сірого ящика.

Тестування білого ящика виконується розробником, де вони перевіряють кожну строку коду, перш ніж передати її інженеру-испытателю. Оскільки код видно розробником під час тестування, його також називають тестуванням білого ящика.

Тестування «чорного ящика» виконується інженером-випробувачем, де він може перевірити працездатність додатків або програмного забезпечення відповідно до потреб клієнта. При цьому код не видно при виконанні тестування; вот чому це відомо як тестування чорного ящика.

Тестування «сірого ящика» являє собою поєднання тестування «білого ящика» та «чорного ящика». Це може виповнити людину, яка знала і програмування, і тестування. Якщо один чоловік виконує тестування білого ящика, а також тестування чорного ящика для додатків, це називається тестуванням серого ящика.

Для виконання ручного тестування потрібно:

– По-перше тестувальник вивчає всі документи, пов’язані з програмним збазпеченням, щоб вибрати область тестування;

– Тестувальник аналізує документи, необхідні, щоб охопити всі вимоги з’явленні замовником;

– Тестувальник розробляє тестові приклади відповідно до вимог документа;

– Усі тестові випадки виконуються вручну з використанням тестування чорного ящика та тестування білого ящика;

– Якщо виникають помилки, команда тестування інформує команду розробників виправляє помилки і передає програмне забезпечення командного тестування для повторного тестування.

Основна робота команди збирання полягає в створенні програми або збірки та перетворення мови високого рівня на мову низького рівня.

Переваги ручного тестування:

– Для ручного тестування не потрібні знання програмування під час використання методу чорного ящика;

– Використовується для тестування дизайну графічного інтерфейсу, що динамічно змінюється;

– Тестер взаємодіє з програмним забезпеченням, як реальний користувач, щоб він міг виявити проблеми з зручністю використання та інтерфейсом користувача;

– Це гарантує, програмне забезпечення на сто відсотків не містить помилок;

– Це економічно вигідно;

– Легко навчатися для нових тестувальників.

Недоліки ручного тестування:

– Це вимагає великої кількості людськи ресурсів;

– Це дуже трудомістко;

– Тестер розробляє тестові кейси на основі своїх навичок та досвіду. Немає доказів того, що вони охоплювали всі функції чи ні;

– Тестові випадки не можна використовувати повторно. Необхідно розробляти окремі тестові випадки для кожного нового програмного забезпечення;

– Не передбачає тестування з усіх аспектів тестування;

– Оскільки дві команди працюють разом, іноді буває важко зрозуміти мотиви один одного, в оману

У ручному тестуванні, різних типах тестування, як-от блок, інтеграція, безпека, продуктивність та відстеження помилок, у нас є різні інструменти, такі як Jira , Bugzilla , Mantis, Zap, NUnit, Tessy, LoadRunner, Citrus, SonarQube тощо. Деякі інструменти є відкритими, а деякі є комерційними.

Іншим методом тестування програмного забезпечення є автоматизоване тестування , яке використовує певні специфічні інструменти для виконання

тестових сценаріїв без будь-якого втручання людини. Це найбільш прийнятний спосіб підвищити ефективність, продуктивність та охоплення тестуванням програмного забезпечення.

Нарешті, ми можемо сказати, що це процес, коли інженер-тестувальник повинен бути дуже наполегливим, інноваційним та чуйним.

За допомогою інструменту автоматизованого тестування ми можемо легко підійти до тестових даних, обробити реалізацію тесту та порівняти фактичний результат із очікуваним результатом.

Незважаючи на те, що наразі ми можемо протестувати майже всі програми за допомогою автоматизованого тестування, ручне тестування все одно необхідне, оскільки воно є основою тестування програмного забезпечення.

Під час автоматизованого тестування інженер з автоматизації тестування напише сценарій тестування або використає інструменти тестування автоматизації для виконання програми. З іншого боку, під час ручного тестування інженер-тестувальник напише тестові випадки та впровадить програмне забезпечення на основі письмових тестових випадків.

В автоматизації тестування інженер -тестувальник може виконувати повторювані завдання та інші пов’язані з ними завдання. Під час ручного тестування це виснажливий процес повторювати повторення знову і знову.

Іншими словами, можна сказати, що основна концентрація тестової автоматизації полягає в зміні ручної діяльності людини за допомогою систем або пристроїв.

Процес автоматизованого тестування заощаджує час, оскільки витрачається менше часу на дослідницьке тестування і більше часу на збереження тестових сценаріїв, одночасно покращуючи повне охоплення тестами.

Чому потрібно виконувати автоматизоване тестування:

– У тестуванні програмного забезпечення автоматизоване тестування необхідне для тестування програми, оскільки воно пропонує нам кращу програму з меншими зусиллями та часом;

– Деякі організації досі виконують лише ручне тестування для тестування програми, оскільки ці компанії не повністю обізнані з процесом автоматизованого тестування;

– Знання про автоматичне тестування на виконання процедури автоматизації тестування в процесі розробки додатків;

– Для впровадження автоматизованого тестування на знадобляться досить значні інвестиції ресурсі і грошей;

Ми можемо повторно використовувати тестові сценарії в автоматизованому тестуванні, і нам не потрібно писати нові тестові сценарії знову і знову. Крім того, ми також можемо повторно створити кроки, які детально описані як попередні.

Порівняно з ручним тестуванням, автоматизоване тестування є більш послідовним і набагато швидшим, ніж виконання звичайних монотонних тестів, які не можна пропустити, але можуть викликати помилки при тестуванні вручну.

У автоматизованому тестуванні ми можемо розпочати процес тестування з будь-якої точки світу та в будь-який час. І навіть ми можемо це зробити дистанційно, якщо не маємо багато підходів або можливості їх придбати.

Ми можемо легко виявити критичні помилки на початкових етапах процесу розробки програмного забезпечення, виконавши автоматичне тестування. Це також допомагає нам витрачати менше робочих годин на вирішення цих проблем і зниження витрат.

Щоб реалізувати сценарій тестування автоматизації, нам потрібен інженер з автоматизації тестування, який може написати тестові сценарії для автоматизації наших тестів, замість того, щоб мати кілька людей, які постійно виконують виснажливі ручні тести.

Автоматичне тестування містить три різні методики та підходи, які допоможуть інженеру-тестувальнику підвищити якість програмного продукту:

– Тестування графічного інтерфейсу;

– Керований кодом;

– Платформа автоматизації тестування.

Графічний інтерфейс користувача у цьому підході ми можемо реалізувати це програмне забезпечення або програму, яка містить графічні інтерфейси. Таким чином, інженери з автоматизованого тестування можуть записувати дії користувача та багато разів оцінювати їх.

Методика, керована кодом, є наступною методологією, яка використовується в автоматизованому тестуванні. У цьому методі інженер-тестувальник в основному зосереджується на виконанні тестового прикладу, щоб визначити, чи виконуються деякі частини коду відповідно до заданої вимоги чи ні.

Іншим підходом до автоматизованого тестування є платформа автоматизації тестування . Платформа автоматизації тестування — це набір правил, які використовуються для отримання цінних результатів діяльності автоматизованого тестування.

Процес автоматизованого тестування — це системний підхід до організації та виконання тестових заходів у спосіб, який забезпечує максимальне охоплення тестуванням з обмеженими ресурсами. Структура тесту включає багатоетапний процес, який підтримує необхідні, детальні та взаємопов’язані дії для виконання завдання.

Методології життєвого циклу автоматизованого тестування (ATLM) . На цьому етапі команда з тестування зосереджена на тому, щоб керувати очікуваннями від тесту та з’ясувати потенційні переваги від правильного застосування автоматизованого тестування.

Під час прийняття автоматизованого тестового костюма організації повинні зіткнутися з багатьма проблемами, деякі з них перераховані нижче:

– Для автоматизованого тестування потрібні фахівці з випробувального обладнання, тому першим питанням призначити спеціаліста з випробувальним обладнанням;

– Вибір точного інструменту для тестування певної функції;

– Питання стандартів проектування та розробки при здійсненні автоматизованого процесу тестування;

– Аналіз різноманітним засобів автоматизованого тестування, щоб вибрати найкращий інструмент для автоматизованого тестування;

– Проблема грошей і часу виникає, оскільки споживання грошей і часу є великим на початок тестування.

Вибір інструментів тестування являє собою другий етап методології життєвого циклу автоматизованого тестування (ATLM) . Цей етап керує тестувальником в оцінці та виборі інструменту тестування.

Оскільки інструмент тестування підтримує майже всі вимоги до тестування, тестувальнику все одно потрібно переглянути середовище розробки системи та інші організаційні потреби, а потім скласти список параметрів оцінки інструментів. Інженери-випробувачі оцінюють обладнання на основі наданих критеріїв зразка.

Методології життєвого циклу автоматизованого тестування (ATLM) . Сфера автоматизації включає область тестування програми. Визначення обсягу базується на таких моментах:

– Загальні функціональні можливості програмного застосунку, якими володіє кожна програма;

– Тести автоматизації встановлює діапазон багаторазового використання бізнес-компонентів;

– Тестування автоматизації визначає ступінь повторного використання бізнес-компонентів;

– Додаток має мати особливості для бізнесу та бути технічно можливим;

– Автоматичне тестування передбачає повторення тестових випадків у разі кросбарузерного тестування.

Ця фаза забезпечує загальну стратегію тестування, якою слід добре керувати та змінювати її, якщо потрібно. Щоб забезпечити наявність навичок,

навички тестування окремого члена та всієї команди аналізуються на порівняння з необхідними специфічними навичками для конкретної програмної програми.

Планування та розробка тестів є четвертою і найважливішою фазою Методології життєвого циклу автоматизованого тестування (ATLM), оскільки тут визначено всі стратегії тестування. На цьому етапі визначаються планування тривалих тестових заходів, створення стандартів і рекомендацій, організація необхідної комбінації обладнання, програмного забезпечення та мережі для створення тестового середовища, процедура відстеження дефектів, рекомендації щодо контролю конфігурації тестування та середовища. . Тестер визначає орієнтовні зусилля та вартість для всього проекту. Документи зі стратегією тестування та оцінкою зусиль є результатами цього етапу. Виконання тестового випадку можна розпочати після успішного завершення планування тесту.

Виконання тестового випадку є шостим етапом методології життєвого циклу автоматизованого тестування (ATLM). Це відбувається після успішного завершення планування тесту. На цьому етапі команда тестування визначає дизайн і розробку тесту. Тепер тестові випадки можна виконувати під час тестування продукту. На цьому етапі команда з тестування починає розробку і виконання кейсів за допомогою автоматизованих інструментів. Підготовлені тестові приклади переглядаються колегами-членами групи тестування або керівниками з забезпечення якості.

Під час виконання тестових процедур команда тестування дала доручення дотримуватись графіка виконання. На етапі виконання реалізуються такі стратегії, як інтеграція, прийняття та модульне тестування, які були визначені в плані тестування раніше.

Огляд та оцінка є шостим і останнім етапом життєвого циклу автоматизованого тестування, але дії цього етапу проводяться протягом усього життєвого циклу для підтримки постійного покращення якості. Процес удосконалення здійснюється за допомогою оцінки матриць, перегляду та оцінки діяльності.

Під час огляду експерт зосереджується на тому, чи відповідає конкретний показник критеріям прийнятності чи ні, якщо так, то він готовий до використання у виробництві програмного забезпечення. Він є вичерпним, оскільки тестові випадки охоплюють кожну функцію програми.

Команда тестування проводить власне опитування, щоб дізнатися про потенційну цінність процесу; якщо потенційна вигода є меншою, ніж достатньо, команда тестування може змінити інструмент тестування. Команда також надає зразок форми опитування, щоб запитати відгуки від кінцевого користувача про атрибути та керування програмним продуктом.

Процес автоматизованого тестування має велику кількість переваг для організації. Зазвичай за допомогою автоматизованого тестування команда тестування зазначеного методу перевірки програмного забезпечення може досягти розширеного тестового покриття.

Хоча ми можемо зіткнутися з різними проблемами під час процесу автоматизованого тестування; тому нам потрібен ретельний процес, щоб досягти успішного виконання автоматизованого тестування.

Переваги автоматизованого тестування:

– Автоматичне тестування займає менше часу, ніж ручне;

– Тестер може перевірити реакцію програмного забезпечення, якщо виконання однієї і тієї ж операції повторюється кілька разів;

– Автоматичне тестування забезпечує можливість повторного використання тестових випадків під час тестування різних версій одного і того ж програмного забезпечення;

– Автоматичне тестування є надійним, оскільки усуває приховані помилки шляхом повторного виконання тестових випадків таким же чином;

– Тестування автоматизації є комплексним, оскільки тестові випадки охоплюють кожну функцію програми;

– Не потрібно багато людських ресурсів, замість того, щоб писати тестові приклади та тестувати їх вручну, їм потрібен інженер з автоматизованого тестування, щоб запустити їх;

– Вартість автоматизованого тестування нижча, ніж ручне тестування, оскільки воно вимагає кількох людських ресурсів.

Недоліки автоматизованого тестування

– Для автоматизованого тестування потрібні кваліфіковані тестери високого рівня;

– Для цього потрібні високо якісні інструменти для тестування;

– Коли стикаємось з невдалим тестом, аналіз усієї події ускладняється;

– Тестова обслуговування є дорогим, оскільки необхідне обладнання для тестування ліцензії за високу плату;

– Налагодження є обов’язковим, якщо менш ефективна помилка не була вирішена, це може призвести до фатальних результатів.

Нарешті, можна зробити висновок, що автоматизоване тестування – це методика тестування програмного забезпечення , яка реалізується за допомогою спеціальних програмних засобів автоматизованого тестування.

Це найкращий підхід до виконання набору тестових прикладів, який допомагає нам покращити охоплення тестами, ефективність та швидкість тестування програмного забезпечення.

Вибір інструменту автоматизованого тестування, процесу тестування та команди є важливими аспектами для успішної автоматизації.

Автоматичне тестування сильно залежить від технології, на якій побудовано тестування програми.

Тестовий підхід до обслуговування автоматизації — це етап тестування автоматизації, який виконується, щоб перевірити, чи добре працюють нові функції, додані до програмного забезпечення, чи ні.

Для успішного процесу тестування програмного забезпечення методики ручної роботи та автоматизації йдуть рука об руку. Ми повинні пояснити процес автоматизації, що він використовується для зменшення часу тестування для певних типів тестів.

4.2 Тестування інтернет-платформи

Під-час проведення тесту вальних робіт було проведено ручний варіант тестування, тобто процес тестування програмного забезпечення буде проводитись вручну без використання яких небудь інструментів. Отже розглянемо сценарії використання інтернет-платформи.

Для початку необхідно запустити програму. Після запуску в браузері відкриється інтернет-платформа, яка зображена на рисунку 4.2.

Рисунок 4.2 – Сторінка входу в магазину взуття

На даній сторінці є дві кнопки, які призначення для переходу на сторінку авторизації чи реєстрації користувача, які зображенні на рисунку 4.3 – 4.4.

Рисунок 4.2 – Сторінка авторизації в магазину взуття

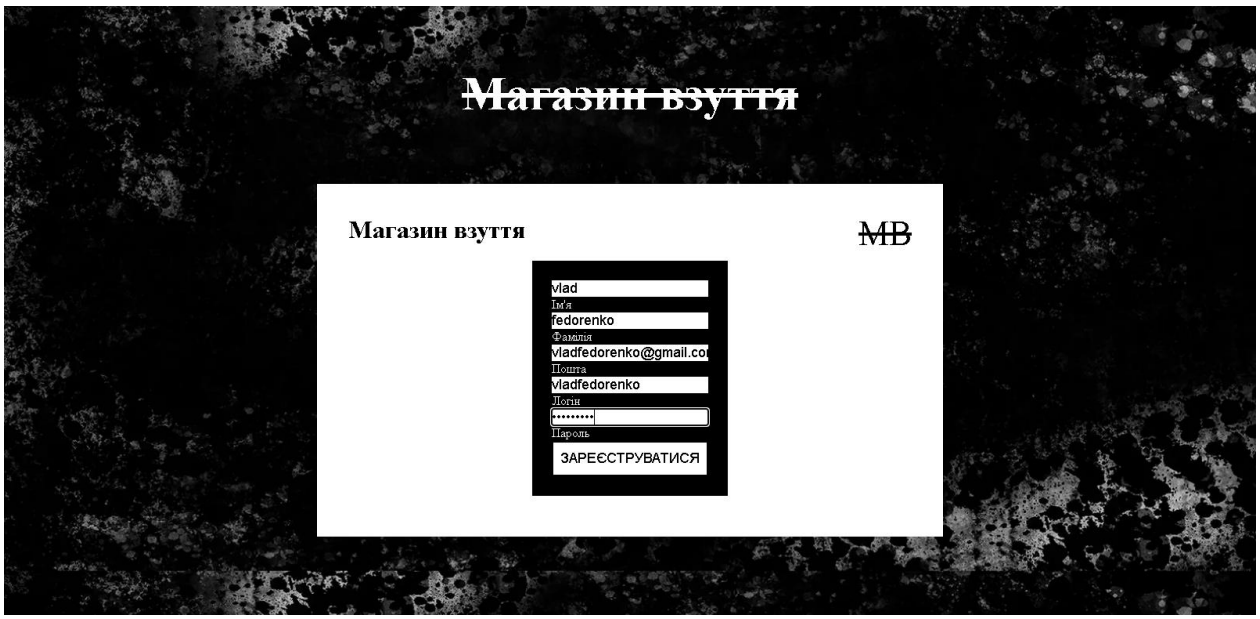


Рисунок 4.3 – Сторінка реєстрації користувача в магазину взуття

Після входу в систему ми автоматично переходимо на головну сторінку інтернет магазину. На головній сторінці має кнопки для переходу між розділами та навігаційне меню, а у разі потреби вийти з магазину взуття, які зображенні на рисунку 4.4.

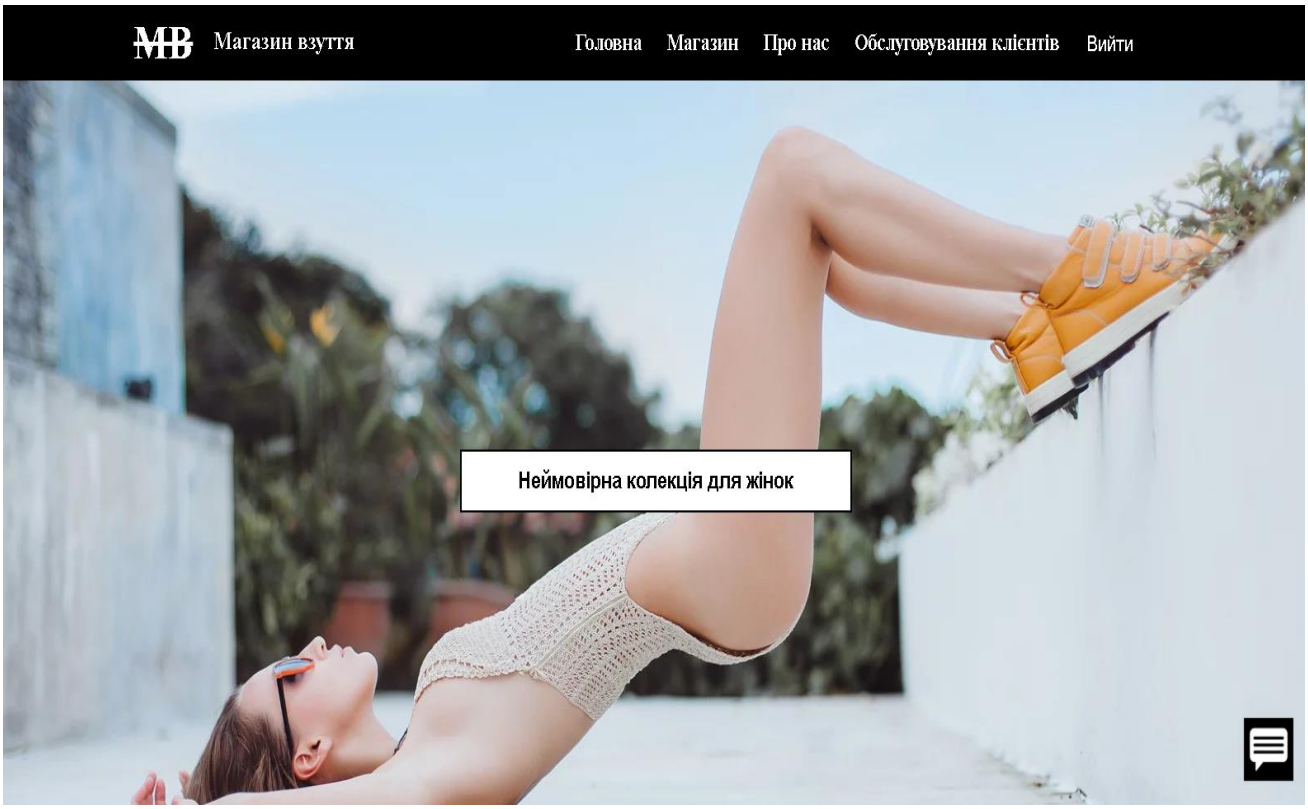


Рисунок 4.4 – Головна сторінка магазину взуття

Зайшовши на головну сторінку ми можемо переходити між розділами в навігаційному меню перейдемо в магазин чоловічого взуття, який зображений на рисунку 4.5.

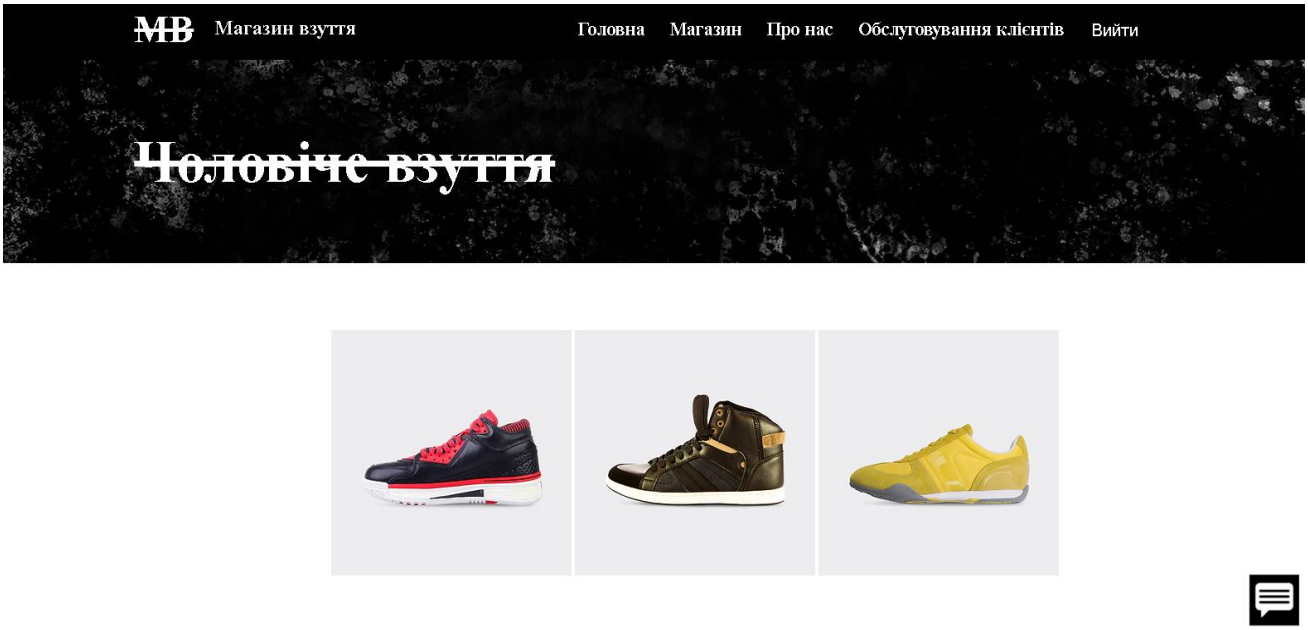


Рисунок 4.5 – Сторінка з чоловічим взуттям

В розділі де продається чоловіче взуття можемо вибрати взуття, яке нам сподобалось після чого відкриється відповідне меню де можемо вибрати колір розмір та придбати. Рисунок 4.6

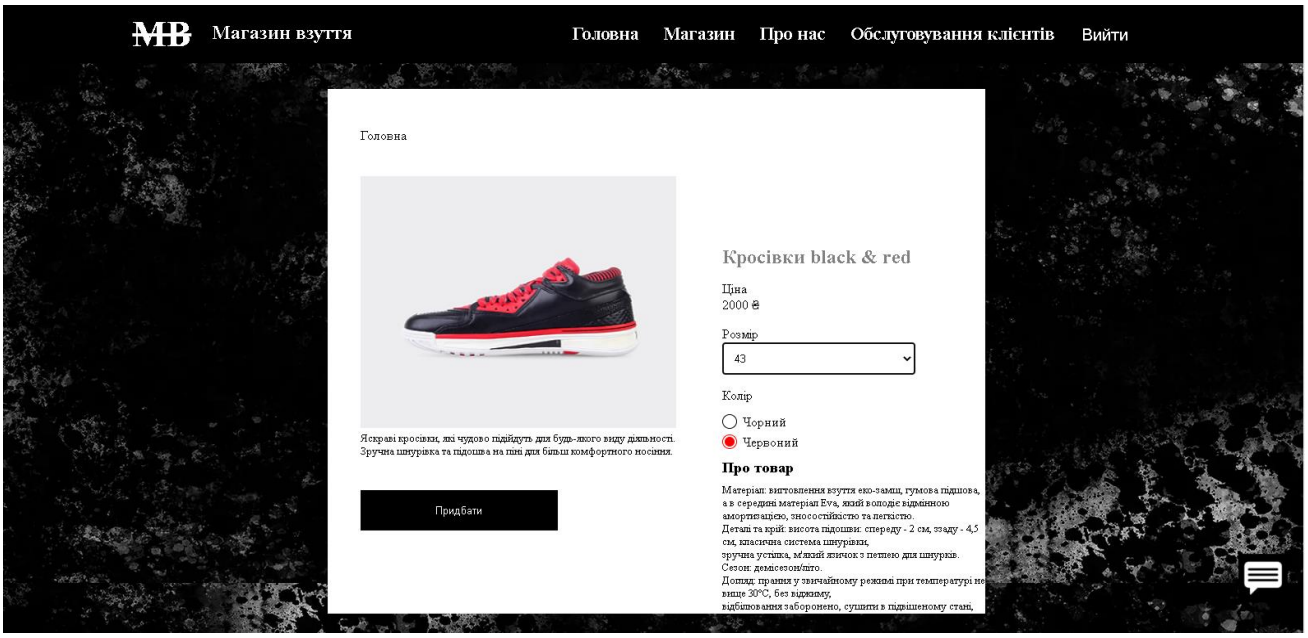


Рисунок 4.6 – Сторінка з взуттям

Також можемо перейти «Про нас», а також «Обслуговування клієнтів», які зображені на рисунку 4.7 – 4.8.

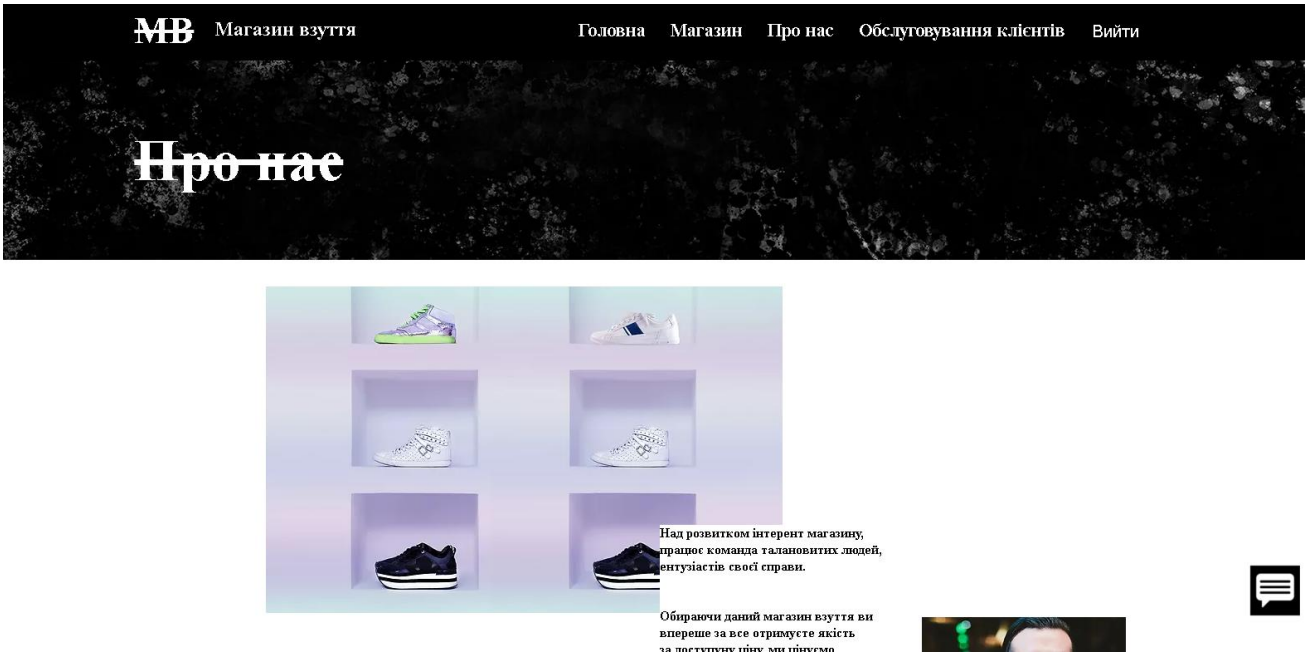


Рисунок 4.7 – Сторінка розділу «Про нас»

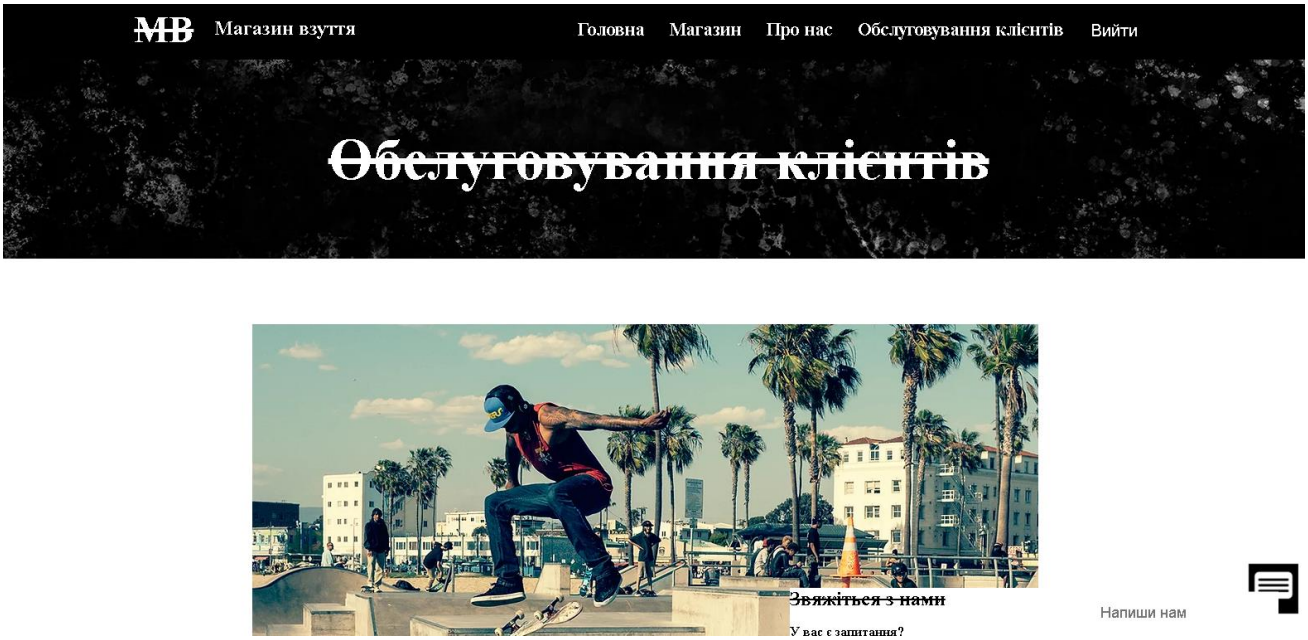


Рисунок 4.8 – Сторінка розділу «обслуговування клієнтів»

Звісно ж, якщо виникнуть, якісь питання користувач має можливість зїднатись з менеджером по роботі з клієнтами на написати йому своє питання. Рисунок 4.9 – 4.10.

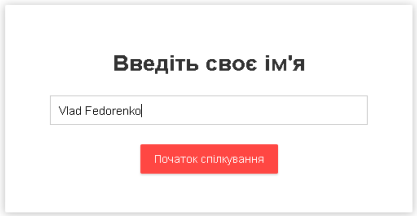


Рисунок 4.9 – Сторінка вводу імені користувача

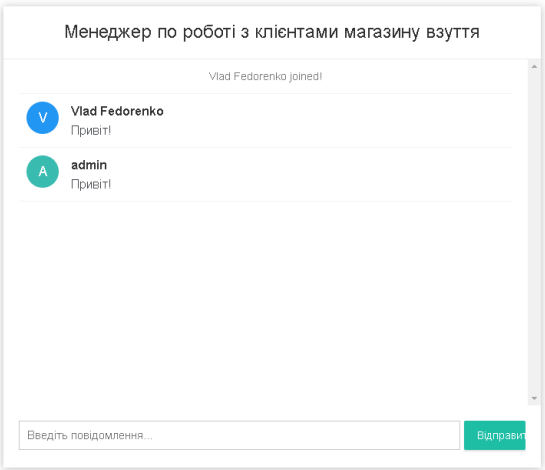


Рисунок 4.10 – Сторінка чату між клієнтом і адміністратором

Саме таким задумувався сценарій роботи інтернет-платформи магазину з продажу взуття.

4.3 Аналіз результатів тестування

Після проведення ручного тестування було розглянуто описи розробленого застосунку, приклади роботи та сценарії роботи додатку з боку користувача. На етапі тестування було проведено тестування роботи всіх сторінок і авторизації, реєстрації в інтернет-платформі, а також можливість з’єднання з менеджером по роботі з клієнтами. Було наведено зображення графічної частини робочої програми.

Як видно з результатів ручного тестування, додаток працює нормально, саме так, як і передбачалося.

Проведення ручного тестування на інших пристроях та браузерах, робота інтернет-платформи повністю задовольняє очікуваним результатам.

Отже, із вказаного вище можна, зробити висновок, що програма працює так як і передбачалось та задовольняє вимоги, які були поставленні в технічному завданні.

ВИСНОВКИ

По закінченню роботи всі задачі були виконані. Було проаналізовано сучасний стан ринку електронних магазинів взуття ї та надані причини, чому підприємцям варто переходити до онлайн продажів. Також було проаналізовано та визначено недоліки сучасних онлайн-магазинів взуття, а також представлено оптимальні рішення цих проблем. В проекті вони всі були враховані та майже всі впроваджені.

Головним завданням проекту є досягнення всіх цілей та виконання завдань проекту, а саме створення інтерет-магазину для продажи взуття. Даний проект надає можливість продавати взуття в мережі інтернет.

В технічному плані були освоєні та обрані сучасні мови веб-програмування, і також оглянуто обраний тип системи керування базами даних. В результаті, обрані технології дійсно гарно працюють та мають всі шанси стати конкурентами вже існуючих розроблених магазинів взуття.

Створений онлайн-магазин можна вже зараз використовувати не тільки в навчальних цілях, але й в комерційних. Окрім цього, завдяки гнучкості програмного коду можна з легкістю продовжувати оснащувати магазин різноманітним функціоналом. Серед перспектив розвитку проекту є розширення функціоналу в вигляді додавання користувацьких відгуків на товар та створення форуму для користувачів. Також можлива автоматизація повернення товару завдяки додаванню потрібних форм. Також, варто додати можливість для менеджера магазину переглядати статистику популярних товарів, витяг інформації про продажі у вигляді таблиць.

Таким чином основними результатами роботи є аналіз та дослідження баз даних, їх типів та систем керування, дослідження існуючих веб-магазинів, сучасних мов веб-програмування, виявлення їх переваг і функцій; та розробка зручного та інтуїтивно зрозумілого онлайн-магазину з врахуванням всіх набутих знань

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. HTML [Електронний ресурс] / Веб сайт HTML – режим доступу до ресурсу: https://devdocs.io/html/. (дата звернення – 10.01.2022).
2. CSS [Електронний ресурс] / CSS – Режим доступу до ресурсу: https://devdocs.io/css/. (дата звернення – 27.01.2022).
3. JavaScript [Електронний ресурс] / JavaScript – Режим доступу до ресурсу: https://javascript.com/. (дата звернення – 15.02.2022).
4. SpringBoot [Електронний ресурс] / SpringBoot – Режим доступу до ресурсу: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/. (дата звернення – 20.02.2022).
5. SpringData [Електронний ресурс] / SpringData – Режим доступу до ресурсу:https://spring.io/spring-data/jpa/docs/current/reference/html/.(дата звернення – 23.02.2022).
6. HiberNate [Електронний ресурс] / HiberNate – Режим доступу до ресурсу: https://hibernate.org/orm/documentation/6.0/.(дата звернення – 05.03.2022).
7. SpringDataJpa [Електронний ресурс] / SpringDataJpa – Режим доступу до ресурсу: https://docs.spring.io/spring-data/jpa/docs/current/regerence/html/. (дата звернення – 05.03.2022).
8. Lombock [Електронний ресурс] / Lombock – Режим доступу до ресурсу: https://projectlombok.org/features/all/. (дата звернення – 07.04.2022).
9. Java[Електронний ресурс] / Java – Режим доступу до ресурсу: https://docs.oracle.com/en/java/. (дата звернення – 15.04.2022).
10. Intellij [Електронний ресурс] / Intellije – Режим доступу до ресурсу: https://jetbrains.com/help/. (дата звернення – 20.02.2022).
11. Maven [Електронний ресурс] / Maven – Режим доступу до ресурсу: https://maven.apache.org/guides/index.html/. (дата звернення – 29.04.2022).

ДОДАТОК А
(обов’язковий)

ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

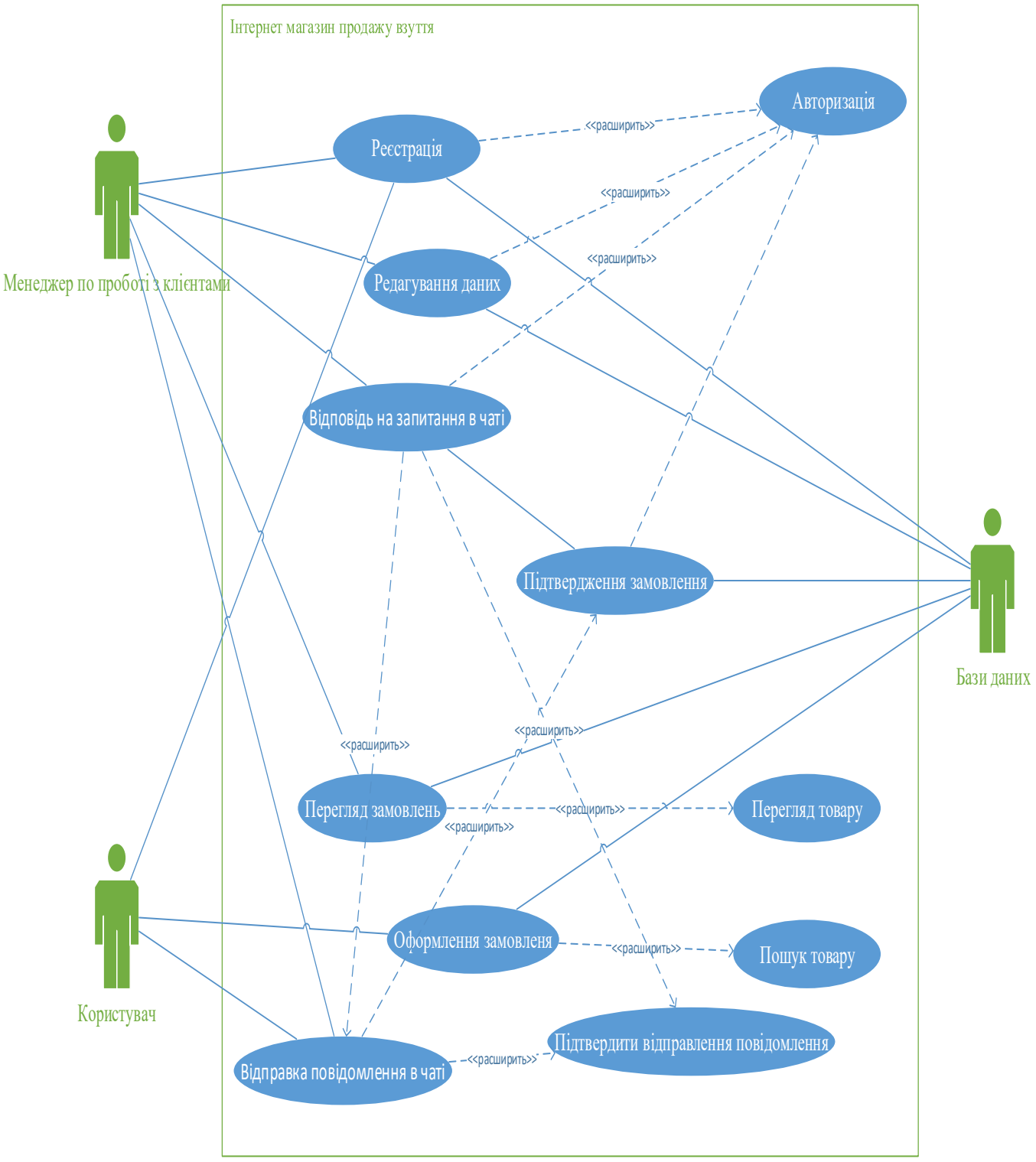


Рисунок А.1 – Діаграма варіантів використання

ДОДАТОК Б

(обов’язковий)

ТЕХНІЧНЕ ЗАВДАННЯ

Введення

Робота виконується в рамках проекту інтернет-платформа магазин з продажу взуття, що дозволяє шукати, переглядати, замовляти товар користувачам.

1 Підстава для розробки

Підставою для розробки є «Завдання на дипломний проект», затверджене завідувачем кафедри інженерії програмного забезпечення.

Найменування розробки: Інтернет-платформа «магазин з продажу взуття».

2 Призначення розробки

2.1 Функціональне призначення

Інтернет-платформа магазин з продажу взуття призначена для перегляду, пошуку, придбання товару.

Користувачами програми є менеджери по роботі з клієнтами, а також звичайні користувачі інтернету.

2.2 Експлуатаційне призначення

Інтернет-платформа повинна використовуватись в браузерах. Кінцевим користувачем виступає люба особа, яка має доступ до інтернету, а також інтернет браузера.

3.3 Вимоги до програми

3.1 Вимоги до функціональних характеристик

Перелік функціональних можливостей, яка надає застосунок:

– Перегляд головної сторінки;

– Відкриття сторінок з товаром для чоловіків та жінок, або товар, який має знижку;

– Перегляд інформації про магазин;

– Можливість зв’язку з менеджером по роботі з клієнтами;

– Оформлення замовлення;

– Вибір кольору взуття;

– Вибір розміру взуття;

– Можливість реєстрації;

– Можливість авторизації.

Вимоги до інтерфейсу:

– Яскраві кольори;

– Мінімалістичний і сучасний вигляд;

– Меню для переходу у розділи, яка знаходиться на шапці інтернет-платформи;

– На головній сторінці має бути вибір товару для жінок, чоловік, а також знижок.

3.2 Вимоги до надійності

Інтернет-платформа має виконувати наступні вимоги:

– Забезпечення збереження інформацій у базі даних;

– Додаток повинен пройти тестування.

3.3 Умови експлуатації та вимоги до технічних засобів

Умови експлуатації повинні відповідати технічним нормам експлуатації персонального комп’ютера, при температурі та відносній вологості навколишнього середовища, визначених для персональної обчислювальної техніки з ГОСТ 15150-19.

Система повинна коректна працювати у веб-браузері де всі навантаження відбувають на серверній частині.

3.4 Вимоги до інформаційної та програмної сумісності

При розробці інтернет-платформи використовувалась така мова програмування, як Java з використання фреймворків Spring Framework, Spring Boot, Spring Data. В якості бази даних використовувалась MySQL, а для спрощення роботи з нею використовувався такий фреймворк, як Spring Data JPA та Hibernate.

3.5 Спеціальні вимоги

Інтернет-платформа повинна мати зручний та інтуїтивний інтерфейс для будь якого користувача.

4 Вимоги до програмної документації

Загальні вимоги до оформлення програмної документації встановлені відповідно до ДСТУ 3009-95, а також єдиній системі програмної документації. Перерахований список документів і їх зміст: Структура програми;

– Текст програми;

– Опис програми ;

– Технічне завдання;

5 Стадії та етапи розробки

Стадії та етапи розробки інтернет-платформи зображені в таблиці Б.1.

Таблиця Б.1 – Стадії та етапи розробки

Стадія розробки	Етапи робіт	Зміст робіт
1	2	3
Технічне завдання 02.01.21 – 31.01.21	Обґрунтування необхідності розробки програми	Коротка характеристика програмного забезпечення; підстава і призначення розробки; вимоги до програмної системи і документація; стад і етапи розробки програми
Ескізний проект 01.02.21 – 26.02.21	Розробка ескізного проекту	Поперечно розробка структури вхідних і вихідних даних; уточнення середовища програмування
Технічний проект 29.02.21 – 19.03.21	Розробка технічного проекту	Уточнення структури вхідних і вихідних даних; розробка структури програми
Робочий проект 20.03.21 – 15.04.21	Розробка програмного забезпечення	Реалізація програмного забезпечення; відлагодження; проведення попереднього тестування
Розробка програмної документації 16.04.21 – 22.04.21	Розробка документації до програмного забезпечення	Розробка необхідної документації, передбаченої технічним завданням
Тестування системи 23.04.21 – 30.04.21	Проведення тестування програмного забезпечення	Розробка методики тестування; проведення основних тестів; коректування програмного забезпечення
Впровадження	Підготовка і передача програми	Підготовка і розгортання програмного забезпечення
Розробка програмної документації 16.04.21 – 22.04.21	Розробка документації до програмного забезпечення	Розробка необхідної документації, передбаченої технічним завданням

6 Порядок контролю приймання

Огляд відбувається фінальним користувачем, підключеною під час тестування системи.

Після завершення сисстема має пройти тестувальні роботи на захист від некоректного введення.

ДОДАТОК В

(обов’язковий)

ПРОГРАМНИЙ КОД

Індекс-файл index.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/main_page.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <div class="text">
                Доброго дня аби отримати доступ, до магазину <br>
                вам потрібно авторизуватись. У випадку, якщо <br>
                ви неможете авторизуватись вам потрібно <br>
                зареєструватись. Для того щоб зареєструватись <br>
                нажміть кнопку реєстрація. Для того аби увійти <br>
                нажміть на кнопку авторизуватись.
            </div>
            <input type="button" value="Авторизуватись" class="login">
            <input type="button" value="Зареєструватися" class="registration">
        </div>
    </body>
</html>
```

Файл login.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/login.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <form action="" class="popup_registration">
                <label>
                    <input type="text" name="text">
                    <div class="label_text">
                        Логін
                    </div>
                </label>
                <label>
                    <input type="password" name="password">
                    <div class="label_text">
                        Пароль
                    </div>
                </label>
                <button type="submit">Увійти</button>
            </form>
```

```
        </div>
    </body>
</html>
```

Файл registration.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" href="css/registration.css">
        <title>Магазин взуття</title>
    </head>
    <body>
        <h1 class="shoes_shop">Магазин взуття</h1>
        <div class="main_page">
            <h1>Магазин взуття</h1>
            <div class="label">MB</div>
            <form action="" class="popup_registration">
                <label>
                    <input type="text" name="text">
                    <div class="label_text">
                        Ім'я
                    </div>
                </label>
                <label>
                    <input type="text" name="text">
                    <div class="label_text">
                        Фамілія
```

```
                    </div>
                </label>
                <label>
                    <input type="text" name="text">
                    <div class="label_text">
                        Пошта
                    </div>
                </label>
                <label>
                    <input type="text" name="text">
                    <div class="label_text">
                        Логін
                    </div>
                </label>
                <label>
                    <input type="password" name="password">
                    <div class="label_text">
                        Пароль
                    </div>
                </label>
                <button type="submit">Зареєструватися</button>
            </form>
        </div>
    </body>
</html>
```

Файл main.html

```
<!DOCTYPE html>
<html                  lang="en"                  xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
```

```
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/main_style.css">
    <title>Магазин взуття</title>
</head>

<body>
    <header >
        <div class="header_logo">
            <a href="/">MB</a>
        </div>
        <div class="header_text">
            <a href="/">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="/">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
                            <li><a href="/forMan">Для чоловіків</a></li>
                            <li><a href="/forWoman">Для жінок</a></li>
                            <li><a href="/sale">Знижки</a></li>
                        </ul>
                    </li>
                    <li> <a href="/aboutUs">Про нас</a></li>
                    <li> <a href="/customerService">Обслуговування клієнтів</a></li>
```

```
                    <li>
                        <form th:action="@{/logout}" method="post" class="exit">
                            <input type="submit" value="Вийти" />
                        </form>
                    </li>
                    <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
                </ul>
            </nav>
        </div>
        <div class="chat">
            <a href="/chat" target="_blank">
                <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
            </a>
        </div>
    </header>
    <main>
        <div class="woman">
            <a href="/forWoman">
                <input type="button" value="Неймовірна колекція для жінок">
            </a>
        </div>
        <img             src="img/shoes/backgorund_sale_shoes.jpg"             alt=""
class="backgroun_sale_image">
        <div class="man">
            <div class="left">
                <a href="/forMan">
                    <img src="img/left_background_man.jpg" alt="">
                </a>
            </div>
            <div class="right">
                <a href="/forMan">
                    <img src="img/right_background_man.jpg" alt="" width="50%">
```

```
            </a>
        </div>
        <a href="/forMan" class="for_man">
            <input type="button" value="Для чоловіків">
        </a>
        <div class="sale">
            <a href="/sale">
                <img src="img/sale_background.jpg" alt="">
            </a>
        </div>
        <a href="/sale" class="for_sale">
            <input type="button" value="Знижки 40%-60%">
        </a>
    </div>
    <div class="shoes">
        <a  href="/firstManShoes">  <img  src="img/shoes/man_first_shoes.jpg"
alt=""></a>
        <a  href="/secondManShoes">  <img  src="img/shoes/man_second_shoes.jpg"
alt=""></a>
        <a  href="/thirdManShoes">  <img  src="img/shoes/man_third_shoes.jpg"
alt=""></a>
        <a  href="/firstWomanShoes">  <img  src="img/shoes/woman_first_shoes.jpg"
alt=""></a>
        <a  href="/secondWomanShoes">  <img  src="img/shoes/woman_second_shoes.jpg"
alt=""></a>
        <a  href="/thirdWomanShoes">  <img  src="img/shoes/woman_third_shoes.jpg"
alt=""></a>

    </div>
    <div class="bg_img">
        <img src="img/footer_background.jpg">
    </div>
```

```
        <div class="footer_logo">
            <a href="/">MB</a>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
            <form th:action="@{/sendMail}" method="post">
                <input type="text" placeholder="Добавити електрону пошту"
name="userMail">
                <button type="submit">OK</button>
            </form>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <form class="write_us_form">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
```

```
        </label>
        <label>
            <input type="text" placeholder="Повідомлення">
        </label>
        <button type="submit">Відправити</button>
    </form>
</main>

<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
```

```
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>
<script src="js/main_script.js"></script>
```

```
</body>


</html>
```

Файл for_man.html

```
<!DOCTYPE html>
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/for_man.css">
    <title>Магазин взуття</title>
</head>


<body>
    <header>
        <div class="header_logo">
            <a href="main.html">MB</a>
        </div>
        <div class="header_text">
            <a href="main.html">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="main.html">Головна</a></li>
                    <li>
                        <a>Магазин</a>
```

```
                        <ul>
                            <li><a href="for_man.html">Для чоловіків</a></li>
                            <li><a href="for_woman.html">Для жінок</a></li>
                            <li><a href="sale.html">Знижки</a></li>
                        </ul>
                    </li>
                    <li> <a href="about_us.html">Про нас</a></li>
                    <li>        <a        href="customer_service.html">Обслуговування
клієнтів</a></li>
                    <li> <a href="#" class="exit">
                        <input type="button" value="Вийти">
                </a></li>
                    <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
                </ul>
            </nav>
        </div>
        <div class="chat">
            <a href="#">
                <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
            </a>
        </div>
    </header>

    <main>
        <h1>Чоловіче взуття</h1>
        <div>
            <div class="square">
                <div class="shoes">
                    <a                href="first_man_shoes.html">                <img
src="img/shoes/man_first_shoes.jpg" alt=""></a>
                    <a              href="second_woman_shoes.html">              <img
src="img/shoes/man_second_shoes.jpg" alt=""></a>
```

```
                    <a              href="third_woman_shoes.html">              <img
src="img/shoes/man_third_shoes.jpg" alt=""></a>
                </div>
            </div>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
            <input type="text" placeholder="Добавити електрону пошту">
            <button type="submit">OK</button>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <form class="write_us_form">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
```

```
        <label>
            <input type="text" placeholder="Повідомлення">
        </label>
        <button type="submit">Відправити</button>
    </form>
</main>

<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
```

```
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>
<script src="js/for_man.js"></script>
</body>
```

```
</html>
```

Файл for_woman.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/for_woman.css">
    <title>Магазин взуття</title>
</head>

<body>
    <header>
        <div class="header_logo">
            <a href="main.html">MB</a>
        </div>
        <div class="header_text">
            <a href="main.html">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="main.html">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
```

```
                        <li><a href="for_man.html">Для чоловіків</a></li>
                        <li><a href="for_woman.html">Для жінок</a></li>
                        <li><a href="sale.html">Знижки</a></li>
                    </ul>
                </li>
                <li> <a href="about_us.html">Про нас</a></li>
                <li>        <a        href="customer_service.html">Обслуговування
клієнтів</a></li>
                <li> <a href="#" class="exit">
                    <input type="button" value="Вийти">
            </a></li>
                <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
            </ul>
        </nav>
    </div>
    <div class="chat">
        <a href="#">
            <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
        </a>
    </div>
</header>

<main>
    <h1>Жіноче взуття</h1>
    <div>
        <div class="square">
            <div class="shoes">
                <a              href="first_woman_shoes.html">              <img
src="img/shoes/woman_first_shoes.jpg" alt=""></a>
                <a              href="second_woman_shoes.html">              <img
src="img/shoes/woman_second_shoes.jpg" alt=""></a>
```

```
                    <a               href="third_woman_shoes.html">               <img
src="img/shoes/woman_third_shoes.jpg" alt=""></a>
                </div>
            </div>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
            <input type="text" placeholder="Добавити електрону пошту">
            <button type="submit">OK</button>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <form class="write_us_form">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
```

```
        <label>
            <input type="text" placeholder="Повідомлення">
        </label>
        <button type="submit">Відправити</button>
    </form>
</main>

<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
```

```
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>
<script src="js/for_woman.js"></script>
</body>
```

```
</html>
```

Файл about_us.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/about_us.css">
    <title>Магазин взуття</title>
</head>

<body>
    <header>
        <div class="header_logo">
            <a href="main.html">MB</a>
        </div>
        <div class="header_text">
            <a href="main.html">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="main.html">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
                            <li><a href="for_man.html">Для чоловіків</a></li>
```

```
                        <li><a href="for_woman.html">Для жінок</a></li>
                        <li><a href="sale.html">Знижки</a></li>
                    </ul>
                </li>
                <li> <a href="about_us.html">Про нас</a></li>
                <li>        <a        href="customer_service.html">Обслуговування
клієнтів</a></li>
                <li> <a href="#" class="exit">
                    <input type="button" value="Вийти">
            </a></li>
                <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
            </ul>
        </nav>
    </div>
    <div class="chat">
        <a href="#">
            <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
        </a>
    </div>
</header>

<main>
    <h1>Про нас</h1>
    <div class="square">
        <img src="img/about_us_shoes.jpg" alt="" class="shoes">
        <div class="about_us_text">
            Над розвитком інтерент магазину, <br>
            працює команда талановитих людей, <br>
            ентузіастів своєї справи. <br>
            <br> <br>
            Обираючи даний магазин взуття ви <br>
            впереше за все отримуєте якість <br>
```

```
            за доступуну ціну. ми цінуємо <br>
            кожного з наших покупців.
            <br> <br>
            Технології розробки та випуску продукції <br>
            безперервно вдосконалюються - наші <br>
            клієнти заслуговують лише найкращого.
        </div>
        <img src="img/about_us_man_photo.jpg" alt="" class="man_photo">
        <h2>Наші магазини</h2>
        <div class="our_shop">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <h3>Графік роботи</h3>
        <div class="work_time">
            Понеділок - П'ятниця: 10:00 - 22:00 <br> <br>
            Субота - Неділя: 10:00 - 20:00
        </div>
    </div>
    <div class="adres">
        Адреса
    </div>
    <div class="write_us">
        Напиши нам
    </div>
    <div class="keep_in_touch">
        Будем на звязку
        <input type="text" placeholder="Добавити електрону пошту">
        <button type="submit">OK</button>
    </div>
```

```
    <div class="adrrees">
        м. Хмельницький,</br>
        вул. Інститутська 11</br>
        vladfedorenko@gmail.ua</br>
        Телефон +380-68-123-45-78</br>
    </div>
    <form class="write_us_form">
        <label>
            <input type="text" placeholder="Напиши нам">
        </label>
        <label>
            <input type="text" placeholder="Електрона пошта">
        </label>
        <label>
            <input type="text" placeholder="Напиши нам">
        </label>
        <label>
            <input type="text" placeholder="Повідомлення">
        </label>
        <button type="submit">Відправити</button>
    </form>
</main>

<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
```

```
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
```

```
                <input type="text" name="text">
                <div class="label_text">
                    Логін
                </div>
            </label>
            <label>
                <input type="password" name="password">
                <div class="label_text">
                    Пароль
                </div>
            </label>
            <button type="submit">Зареєструватися</button>
        </form>
    </div>

    <script src="js/about_us.js"></script>
</body>

</html>
```

Файл customer_service.html

```
<!DOCTYPE html>
<html                  lang="en"                  xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <link rel="stylesheet" href="css/customer_service.css">
    <title>Магазин взуття</title>
</head>

<body>
    <header>
        <div class="header_logo">
            <a href="/main">MB</a>
        </div>
        <div class="header_text">
            <a href="/main">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="/main">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
                            <li><a href="/forMan">Для чоловіків</a></li>
                            <li><a href="/forWoman">Для жінок</a></li>
                            <li><a href="/sale">Знижки</a></li>
                        </ul>
                    </li>
                    <li> <a href="/aboutUs">Про нас</a></li>
                    <li> <a href="/customerService">Обслуговування клієнтів</a></li>
                    <li> <a href="#" class="come_In">Вхід</a></li>
                    <li> <a href="#" class="registration">Реєстрація</a></li>
                    <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
                </ul>
            </nav>
        </div>
```

```
    <div class="chat">
        <a href="/chat" target="_blank">
            <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
        </a>
    </div>
</header>

<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
```

```
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>

<main>
```

```
<h1>Обслуговування клієнтів</h1>
<div class="square">
    <img src="img/customer_service.jpg" alt="">
    <div class="contact_us">
        <h2>Звяжіться з нами</h2>
        У вас є запитання? <br> <br>
        Можливо ви найдете відповідь на FAQ, який знаходиться нижче <br>
        Ми постарались дати відповідь на всі ваші запитання. <br>
        <br>
        Ви також можете написати на пошту vladfedorenko@gmail.ua, <br>
        або просто заповнити форму:
        <form class="write_us_form_first">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Повідомлення">
            </label>
            <button type="submit">Відправити</button>
        </form>
    </div>
</div>
<div class="faq">FAQ</div>>
<div class="faq_square">
    <div class="find_shoes">
        <span>
```

```
            01 <br> <br>
            Я найшов потрібне взуття, але неможу <br>
            вибрати розмір. <br>
        </span>
        <br>
        Порівнюючи довжину стопи та довжину устілки обраного взуття, <br>
        Ви без проблем зможете підібрати відповідний розмір. <br>
        У більшості людей між лівою і правою стопою є невеличка <br>
        різниця в довжині, тому необхідно виміряти дві стопи. <br>
        Розмір взуття завжди підбираємо по стопі, яка має більшу довжину.
    </div>
    <div class="about_product">
        <span>
            02 <br> <br>
            Можете розповісти про продробиці, <br>
            вашої продукції? <br>
        </span>
        <br>
        Наша продукція є одна з самих кращих і якісніших, <br>
        Ви без проблем зможете підібрати понравившейся взуття. <br>
        Служба підтримки завжди вам у цьому допоможе.
    </div>
    <div class="order">
        <span>
            03 <br> <br>
            Моє замовлення вже відправлине? <br>
        </span>
        <br>
        Після замовлення взуття з вами звяжиться менеджер по <br>
        роботі з клієнтами і уточнить стань вашого замовлення, <br>
        а також адресу, якщо цього не відбулося, то ваше замовлення <br>
        на даний момент розглядається.
```

```
            </div>
            <div class="find_oreder">
                <span>
                    04 <br> <br>
                    Чи можу я відслідковувати замовлення?
                </span>
                <br>
                Звісно ви маєте змогу відслідковувати замовлення для <br>
                цього вам потрібно написати номер вашого замовлення менеджеру <br>
                або зателефонувати по гарячому номеру.
            </div>
            <div class="part_shoes">
                <span>
                    05 <br> <br>
                    Мені прийшла тільки частина замовлення. <br>
                </span>
                <br>
                Якщо вам прийшла тільки частина замовлення чи замовлення <br>
                було пошкоджене протягом 14 днів через поштове відділення ви <br>
                можете відправити його назад для заміни, чи повного поверненя <br>
                товару.
            </div>
            <div class="not_this_shoes">
                <span>
                    06 <br> <br>
                    Ви відправили мені не той товар!
                </span>
                <br>
                Якщо вам прийшов не той товар ви можете звернутись до менеджера <br>
                по роботі з клієнтами або повернути товар через поштове відділення
<br>
                за адресою відправника.
```

```
</div>
<div class="about_pay">
    <span>
        07 <br> <br>
        Як оплатити замовлення?
    </span>
    <br>
    Для того, щоб оплатити замовлення вам потрібно перед оплатою, <br>
    перевірити замовлення та прозвести оплату товару на поштовому <br>
    відділу, картою або налічкою.
</div>
<div class="sale">
    <span>
        08 <br> <br>
        Я забув використати купон зі знижкою.
    </span>
    <br>
    Якщо ви забули використати купон зі знижкою ви можете <br>
    його використати звернувшись до менеджерам. <br>
    Якщо, товар вже надійшов ви можете його повернути і <br>
    використати знижку перезамовивши товар.
</div>
<div class="safe_order">
    <span>
        09 <br> <br>
        Замовляти через інтернет безпечно?.
    </span>
    <br>
    Перед тим, як купувати товар в інтерент-магазині прочитайте <br>
    або зверніться до менеджера по роботі з клієнтами за цією <br>
    інформацієї, щоб не стати жертвою. Звісно ж наш магазин надає <br>
    повні гарантіє безпеки замовлення для наших клієнтів відповідно, <br>
```

```
                до законодавства України.
            </div>
            <div class="time_order">
                <span>
                    10 <br> <br>
                    Скілки коштує і скільки часу займає замовлення? <br>
                </span>
                <br>
                Доставка відбувається за ціновою політикою вашої пошти. <br>
                Час протягом, якого товар прибуває до вас в середньому. <br>
                три дні по України, якщо товар замовляється за кордон, то <br>
                час замовлення займає значно більший час.
            </div>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
            <form th:action="@{/sendMail}" method="post">
                <input    type="text"    placeholder="Добавити    електрону    пошту"
name="userMail">
                <button type="submit">OK</button>
            </form>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
```

```
        Телефон +380-68-123-45-78</br>
    </div>
    <form class="write_us_form">
        <label>
            <input type="text" placeholder="Напиши нам">
        </label>
        <label>
            <input type="text" placeholder="Електрона пошта">
        </label>
        <label>
            <input type="text" placeholder="Напиши нам">
        </label>
        <label>
            <input type="text" placeholder="Повідомлення">
        </label>
        <button type="submit">Відправити</button>
    </form>
</main>



<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
```

```
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
```

```
                    Логін
                </div>
            </label>
            <label>
                <input type="password" name="password">
                <div class="label_text">
                    Пароль
                </div>
            </label>
            <button type="submit">Зареєструватися</button>
        </form>
    </div>
    <script src="js/customer_service.js"></script>
</body>

</html>
```

Файл first_man_shoes.html

```
<!DOCTYPE html>
<html                  lang="en"                  xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/first_man_shoes.css">
    <title>Магазин взуття</title>
</head>
```

```
<body>
    <header>
        <div class="header_logo">
            <a href="/main">MB</a>
        </div>
        <div class="header_text">
            <a href="/main">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="/main">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
                            <li><a href="/forMan">Для чоловіків</a></li>
                            <li><a href="/forWoman">Для жінок</a></li>
                            <li><a href="/sale">Знижки</a></li>
                        </ul>
                    </li>
                    <li> <a href="/aboutUs">Про нас</a></li>
                    <li> <a href="/customerService">Обслуговування клієнтів</a></li>
                    <li> <a href="#" class="come_In">Вхід</a></li>
                    <li> <a href="#" class="registration">Реєстрація</a></li>
                    <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
                </ul>
            </nav>
        </div>
        <div class="chat">
            <a href="/chat" target="_blank">
                <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
```

```
            </a>
        </div>
    </header>
    <main>
        <div class="square">
            <a href="/main">
                Головна
            </a>
            <img src="img/shoes/man_first_shoes.jpg" alt="">
            <div class="about_shoes">Яскраві кросівки, які чудово підійдуть для будь-
якого виду діяльності.<br>
                Зручна шнурівка та підошва на піні для більш комфортного носіння.
            </div>
            <h2>Кросівки black & red</h2>
            <div class="prise">
                Ціна <br>
                2000 &#8372;
            </div>
            <div class="size">Розмір</div>
            <select>
                <option value="1">40</option>
                <option value="2">41</option>
                <option value="3">42</option>
                <option value="4">43</option>
                <option value="5">44</option>
                <option value="6">45</option>
            </select>
            <div class="color">
                Колір
            </div>
            <div class="radio">
```

```
                <input     class="radio_input"     name="shoes_color"     type="radio"
id="radio_1">
                <label class="radio_label_first" for="radio_1">Чорний</label>
            </div>
            <div class="radio">
                <input     class="radio_input"     name="shoes_color"     type="radio"
id="radio_2">
                <label class="radio_label_second" for="radio_2">Червоний</label>
            </div>
            <div class="description">
                <h3>Про товар</h3>
                Матеріал: виготовлення взуття еко-замш, гумова підшова, а в середині
матеріал Eva,
                який володіє відмінною амортизацією, зносостійкістю та легкістю. <br>
                Деталі та крій: висота підошви: спереду - 2 см, ззаду - 4,5 см, класична
система шнурівки, <br>
                зручна устілка, м'який язичок з петлею для шнурків. <br>
                Сезон: демісезон/літо. <br>
                Догляд: прання у звичайному режимі при температурі не вище 30°C, без
віджиму, <br>
                відбілювання заборонено, сушити в підвішеному стані, без застосування
штучної сушки.
            </div>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
```

```
            <form th:action="@{/sendMail}" method="post">
                <input    type="text"    placeholder="Добавити    електрону    пошту"
name="userMail">
                <button type="submit">OK</button>
            </form>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <form class="write_us_form">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Повідомлення">
            </label>
            <button type="submit">Відправити</button>
        </form>
    </main>

    <div class="popup_bg_comeIn">
        <form class="popup_comeIn">
            <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
```

```
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
```

```
                <input type="text" name="text">
                <div class="label_text">
                    Логін
                </div>
            </label>
            <label>
                <input type="text" name="text">
                <div class="label_text">
                    Логін
                </div>
            </label>
            <label>
                <input type="password" name="password">
                <div class="label_text">
                    Пароль
                </div>
            </label>
            <button type="submit">Зареєструватися</button>
        </form>
    </div>
    <script src="js/first_man_shoes.js"></script>
</body>

</html>
```

Файл first_woman_shoes.html

```
<!DOCTYPE html>
<html                  lang="en"                  xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/first_woman_shoes.css">
    <title>Магазин взуття</title>
</head>

<body>
    <header>
        <div class="header_logo">
            <a href="/main">MB</a>
        </div>
        <div class="header_text">
            <a href="/main">Магазин взуття</a>
        </div>
        <div class="header_navigation responsive" id="headerNavigation">
            <nav>
                <ul>
                    <li> <a href="/main">Головна</a></li>
                    <li>
                        <a>Магазин</a>
                        <ul>
                            <li><a href="/forMan">Для чоловіків</a></li>
                            <li><a href="/forWoman">Для жінок</a></li>
                            <li><a href="/sale">Знижки</a></li>
                        </ul>
                    </li>
                    <li> <a href="/aboutUs">Про нас</a></li>
                    <li> <a href="/customerService">Обслуговування клієнтів</a></li>
                    <li> <a href="#" class="come_In">Вхід</a></li>
```

```
                    <li> <a href="#" class="registration">Реєстрація</a></li>
                    <li> <a href="#" id="menu" class="icon">&#9776;</a></li>
                </ul>
            </nav>
        </div>
        <div class="chat">
            <a href="/chat" target="_blank">
                <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
            </a>
        </div>
    </header>
    <main>
        <div class="square">
            <a href="/main">
                Головна
            </a>
            <img src="img/shoes/woman_first_shoes.jpg" alt="">
            <div class="about_shoes">Зручні та легкі кросівки на високій підошві
відмінно поєднуються з <br>
                будь-яким стилем і придають виразності твоїй нозі.<br>
            </div>
            <h2>Кросівки Fllowers</h2>
            <div class="prise">
                Ціна <br>
                1500 &#8372;
            </div>
            <div class="size">Розмір</div>
            <select>
                <option value="1">35</option>
                <option value="2">36</option>
                <option value="3">37</option>
                <option value="4">38</option>
```

```
                <option value="5">39</option>
                <option value="6">40</option>
            </select>
            <div class="color">
                Колір
            </div>
            <div class="radio">
                <input     class="radio_input"     name="shoes_color"     type="radio"
id="radio_1">
                <label class="radio_label_first" for="radio_1">Коричневий</label>
            </div>
            <div class="radio">
                <input     class="radio_input"     name="shoes_color"     type="radio"
id="radio_2">
                <label class="radio_label_second" for="radio_2">Білий</label>
            </div>
            <div class="description">
                <h3>Про товар</h3>
                Матеріал: еко-шкіра/текстиль, підошва Eva має відмінну амортизацію,
легкість,
                зносостійкість і не жовтіє з часом. <br>
                Деталі та крій: висота підошви: 2 см, підошва прошита по всьому
периметру виробу,
                класична система шнурівки, м'який язичок з петлею для шнурків. <br>
                Сезон: демісезон/літо. <br>
                Догляд: тільки ручна чистка.
            </div>
        </div>
        <div class="adres">
            Адреса
        </div>
        <div class="write_us">
```

```
            Напиши нам
        </div>
        <div class="keep_in_touch">
            Будем на звязку
            <form th:action="@{/sendMail}" method="post">
                <input    type="text"    placeholder="Добавити    електрону    пошту"
name="userMail">
                <button type="submit">OK</button>
            </form>
        </div>
        <div class="adrrees">
            м. Хмельницький,</br>
            вул. Інститутська 11</br>
            vladfedorenko@gmail.ua</br>
            Телефон +380-68-123-45-78</br>
        </div>
        <form class="write_us_form">
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Електрона пошта">
            </label>
            <label>
                <input type="text" placeholder="Напиши нам">
            </label>
            <label>
                <input type="text" placeholder="Повідомлення">
            </label>
            <button type="submit">Відправити</button>
        </form>
    </main>
```

```
<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>

<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
```

```
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>
<script src="js/first_woman_shoes.js"></script>
</body>

</html>
```

Файл chat.html

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-scale=1.0">
    <title>Магазин взуття чат</title>
    <link rel="stylesheet" href="/css/chat.css" />
</head>
<body>
<noscript>
    <h2>Sorry! Your browser doesn't support Javascript</h2>
</noscript>

<div id="username-page">
    <div class="username-page-container">
        <h1 class="title">Введіть своє ім'я</h1>
        <form id="usernameForm" name="usernameForm">
            <div class="form-group">
                <input type="text" id="name" placeholder="Введіть своє ім'я" autocomplete="off" class="form-control" />
            </div>
            <div class="form-group">
                <button type="submit" class="accent username-submit">Початок спілкування</button>
            </div>
        </form>
    </div>
</div>

<div id="chat-page" class="hidden">
    <div class="chat-container">
```

```
        <div class="chat-header">
            <h2>Менеджер по роботі з клієнтами магазину взуття</h2>
        </div>
        <div class="connecting">
            Підключення...
        </div>
        <ul id="messageArea">

        </ul>
        <form id="messageForm" name="messageForm">
            <div class="form-group">
                <div class="input-group clearfix">
                    <input     type="text"     id="message"     placeholder="Введіть
повідомлення..." autocomplete="off" class="form-control"/>
                    <button type="submit" class="primary">Відправити</button>
                </div>
            </div>
        </form>
    </div>
</div>

<script                          src="https://cdnjs.cloudflare.com/ajax/libs/sockjs-
client/1.1.4/sockjs.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/stomp.js/2.3.3/stomp.min.js"></script>
<script src="/js/chat.js"></script>
</body>
</html>
```

Файл sale.html

```
<!DOCTYPE html>
<html                  lang="en"                  xmlns="http://www.w3.org/1999/xhtml"
xmlns:th="https://www.thymeleaf.org"
      xmlns:sec="https://www.thymeleaf.org/thymeleaf-extras-springsecurity3">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="css/sale.css">
    <title>Магазин взуття</title>
</head>

<body>
<header>
    <div class="header_logo">
        <a href="/main">MB</a>
    </div>
    <div class="header_text">
        <a href="/main">Магазин взуття</a>
    </div>
    <div class="header_navigation responsive" id="headerNavigation">
        <nav>
            <ul>
                <li><a href="/main">Головна</a></li>
                <li>
                    <a>Магазин</a>
                    <ul>
                        <li><a href="/forMan">Для чоловіків</a></li>
                        <li><a href="/forWoman">Для жінок</a></li>
                        <li><a href="/sale">Знижки</a></li>
                    </ul>
```

```
                </li>
                <li><a href="/aboutUs">Про нас</a></li>
                <li><a href="/customerService">Обслуговування клієнтів</a></li>
                <li><a href="#" class="come_In">Вхід</a></li>
                <li><a href="#" class="registration">Реєстрація</a></li>
                <li><a href="#" id="menu" class="icon">&#9776;</a></li>
            </ul>
        </nav>
    </div>
    <div class="chat">
        <a href="/chat" target="_blank">
            <img src="img/chat_icon.jpg" alt="" width="60px" height="60px">
        </a>
    </div>
</header>

<main>
    <h1>Знижки на взуття</h1>
    <div>
        <div class="square">
            <div class="shoes">
                <a  href="/thirdManShoes">  <img  src="img/shoes/man_third_shoes.jpg"
alt=""></a>
                <a                    href="/secondWomanShoes">                    <img
src="img/shoes/woman_second_shoes.jpg" alt=""></a>
                <a                     href="/thirdWomanShoes">                    <img
src="img/shoes/woman_third_shoes.jpg" alt=""></a>
            </div>
        </div>
    </div>
    <div class="adres">
        Адреса
```

```
</div>
<div class="write_us">
    Напиши нам
</div>
<div class="keep_in_touch">
    Будем на звязку
    <form th:action="@{/sendMail}" method="post">
        <input type="text" placeholder="Добавити електрону пошту" name="userMail">
        <button type="submit">OK</button>
    </form>
</div>
<div class="adrrees">
    м. Хмельницький,</br>
    вул. Інститутська 11</br>
    vladfedorenko@gmail.ua</br>
    Телефон +380-68-123-45-78</br>
</div>
<form class="write_us_form">
    <label>
        <input type="text" placeholder="Ім'я">
    </label>
    <label>
        <input type="text" placeholder="Електрона пошта">
    </label>
    <label>
        <input type="text" placeholder="Телефон">
    </label>
    <label>
        <input type="text" placeholder="Повідомлення">
    </label>
    <button type="submit">Відправити</button>
</form>
```

```
</main>


<div class="popup_bg_comeIn">
    <form class="popup_comeIn">
        <img src="img/close.jpg" class="close_popup_comeIn" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Увійти</button>
    </form>
</div>


<div class="popup_bg_registration">
    <form class="popup_registration">
        <img src="img/close.jpg" class="close_popup_registration" width="20px">
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Ім'я
            </div>
        </label>
        <label>
            <input type="text" name="text">
```

```
            <div class="label_text">
                Фамілія
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="text" name="text">
            <div class="label_text">
                Логін
            </div>
        </label>
        <label>
            <input type="password" name="password">
            <div class="label_text">
                Пароль
            </div>
        </label>
        <button type="submit">Зареєструватися</button>
    </form>
</div>
<script src="js/for_man.js"></script>
</body>

</html>
```

Код JavaScript chat.js

```
'use strict';

var usernamePage = document.querySelector('#username-page');
var chatPage = document.querySelector('#chat-page');
var usernameForm = document.querySelector('#usernameForm');
var messageForm = document.querySelector('#messageForm');
var messageInput = document.querySelector('#message');
var messageArea = document.querySelector('#messageArea');
var connectingElement = document.querySelector('.connecting');

var stompClient = null;
var username = null;

var colors = [
    '#2196F3', '#32c787', '#00BCD4', '#ff5652',
    '#ffc107', '#ff85af', '#FF9800', '#39bbb0'
];

function connect(event) {
    username = document.querySelector('#name').value.trim();

    if(username) {
        usernamePage.classList.add('hidden');
        chatPage.classList.remove('hidden');

        var socket = new SockJS('/ws');
        stompClient = Stomp.over(socket);

        stompClient.connect({}, onConnected, onError);
    }
    event.preventDefault();
```

```
}


function onConnected() {
    // Subscribe to the Public Topic
    stompClient.subscribe('/topic/public', onMessageReceived);

    // Tell your username to the server
    stompClient.send("/app/chat.addUser",
        {},
        JSON.stringify({sender: username, type: 'JOIN'})
    )

    connectingElement.classList.add('hidden');
}


function onError(error) {
    connectingElement.textContent = 'Could not connect to WebSocket server. Please
refresh this page to try again!';
    connectingElement.style.color = 'red';
}


function sendMessage(event) {
    var messageContent = messageInput.value.trim();
    if(messageContent && stompClient) {
        var chatMessage = {
            sender: username,
            content: messageInput.value,
            type: 'CHAT'
        };
```

```
        stompClient.send("/app/chat.sendMessage", {}, JSON.stringify(chatMessage));
        messageInput.value = '';
    }
    event.preventDefault();
}


function onMessageReceived(payload) {
    var message = JSON.parse(payload.body);

    var messageElement = document.createElement('li');

    if(message.type === 'JOIN') {
        messageElement.classList.add('event-message');
        message.content = message.sender + ' joined!';
    } else if (message.type === 'LEAVE') {
        messageElement.classList.add('event-message');
        message.content = message.sender + ' left!';
    } else {
        messageElement.classList.add('chat-message');

        var avatarElement = document.createElement('i');
        var avatarText = document.createTextNode(message.sender[0]);
        avatarElement.appendChild(avatarText);
        avatarElement.style['background-color'] = getAvatarColor(message.sender);

        messageElement.appendChild(avatarElement);

        var usernameElement = document.createElement('span');
        var usernameText = document.createTextNode(message.sender);
        usernameElement.appendChild(usernameText);
        messageElement.appendChild(usernameElement);
```

```
    }

    var textElement = document.createElement('p');
    var messageText = document.createTextNode(message.content);
    textElement.appendChild(messageText);

    messageElement.appendChild(textElement);

    messageArea.appendChild(messageElement);
    messageArea.scrollTop = messageArea.scrollHeight;
}


function getAvatarColor(messageSender) {
    var hash = 0;
    for (var i = 0; i < messageSender.length; i++) {
        hash = 31 * hash + messageSender.charCodeAt(i);
    }
    var index = Math.abs(hash % colors.length);
    return colors[index];
}

usernameForm.addEventListener('submit', connect, true)
messageForm.addEventListener('submit', sendMessage, true)
```

Код JavaScript main_script.ja

```
menu.onclick = function myFunction(){
    var x = document.getElementById('headerNavigation')

    if(x.className === "header_navigation"){
```

```
        x.className += "responsive"
    }else{
        x.className = "header_navigation"
    }
}

let popupBgComeIn = document.querySelector('.popup_bg_comeIn');
let popupComeIn = document.querySelector('.popup');
let openPopupButtonsComeIn = document.querySelectorAll('.come_In');
let closePopupButtonIn = document.querySelector('.close_popup_comeIn');


openPopupButtonsComeIn.forEach((button) => {
    button.addEventListener('click', (e)=>{
        e.preventDefault();
        popupBgComeIn.classList.add('active');
    })
});

closePopupButtonIn.addEventListener('click', ()=>{
    popupBgComeIn.classList.remove('active');
    popupComeIn.classList.remove('active');
});

document.addEventListener('click', (e)=>{
    if(e.target === popupBgComeIn){
        popupBgComeIn.classList.remove('active');
        popupComeIn.classList.remove('active');
    }
})

let popupBgRegistration = document.querySelector('.popup_bg_registration');
```

```
let openPopupButtonsRegistration = document.querySelectorAll('.registration');
let                          closePopupButtonRegistration                          =
document.querySelector('.close_popup_registration');


openPopupButtonsRegistration.forEach((button) => {
    button.addEventListener('click', (e)=>{
        e.preventDefault();
        popupBgRegistration.classList.add('active');
    })
});

closePopupButtonRegistration.addEventListener('click', ()=>{
    popupBgRegistration.classList.remove('active');
    popupRegistration.classList.remove('active');
});

document.addEventListener('click', (e)=>{
    if(e.target === popupBgRegistration){
        popupBgRegistration.classList.remove('active');
        popupRegistration.classList.remove('active');
    }
})
```

Клас Application

```
@SpringBootApplication
public class Application {
    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Клас UserService

```
@Service
public class UserService {
    User user = new User();
    @Autowired
    private UserRepository userRepository;

    public void findUser(String firstName){
        user.setFirstName(firstName);
        user.setLastName(firstName);
        userRepository.save(user);
    }
}
```

Клас UserMailService

```
@Service
public class UserMailService {
    UserMail userMail = new UserMail();
    @Autowired
    private UserMailRepository userMailRepository;

    public void sendMailToDatabase(String userMails) {
        userMail.setUserMail(userMails);
        userMailRepository.save(userMail);
    }
}
```

Інтерфейс UserRepository

```
@Repository
public interface UserRepository extends CrudRepository<User, Integer> {
    User findByUsername(String username);
}
```

Інтерфейс UserMailRepository

```
@Repository
public interface UserMailRepository extends CrudRepository<UserMail, Integer> {
}
```

Клас UserMail

```
@Getter
@Setter
@Entity
@Table(name = "user_mail")
public class UserMail {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Integer id;
    private String userMail;
}
```

Клас User

```
@Getter
@Setter
@Entity
@Table(name = "usr")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    public Integer id;
    private String username;
    private String firstName;
    private String lastName;
    private String eMail;
    private String password;
    private boolean active;

    @ElementCollection(targetClass = Role.class, fetch = FetchType.EAGER)
    @CollectionTable(name = "user_role", joinColumns = @JoinColumn(name = "user_id"))
    @Enumerated(EnumType.STRING)
    private Set<Role> roles;
}
```

Клас Role

```
public enum Role {
    USER;
}
```

Клас ChatMessagePojo

```
public class ChatMessagePojo {
```

```
    private MessageType type;
    private String content;
    private String sender;

    public MessageType getType() {
        return type;
    }

    public void setType(MessageType type) {
        this.type = type;
    }

    public String getContent() {
        return content;
    }

    public void setContent(String content) {
        this.content = content;
    }
    public String getSender() {
        return sender;
    }
    public void setSender(String sender) {
        this.sender = sender;
    }

    public enum MessageType {
        CHAT,
        JOIN,
        LEAVE
    }
}
```

Клас WebSocketEventListener

```
@Component
public class WebSocketEventListener {
    private static final Logger logger = LoggerFactory.getLogger(WebSocketEventListener.class);

    @Autowired
    private SimpMessageSendingOperations messagingTemplate;

    @EventListener
    public void handleWebSocketConnectListener(SessionConnectedEvent event) {
        logger.info("Отримане нове підключення до веб-сокета!");
    }

    @EventListener
    public void handleWebSocketDisconnectListener(SessionDisconnectEvent event) {
        StompHeaderAccessor headerAccessor = StompHeaderAccessor.wrap(event.getMessage());

        String username = (String) headerAccessor.getSessionAttributes().get("username");
        if (username != null) {
            logger.info("Користувач відключився : " + username);

            ChatMessagePojo chatMessagePojo = new ChatMessagePojo();
            chatMessagePojo.setType(ChatMessagePojo.MessageType.LEAVE);
            chatMessagePojo.setSender(username);

            messagingTemplate.convertAndSend("/topic/public", chatMessagePojo);
        }
    }
```

```
}
```

Клас UserController

```
@RestController
public class UserController {

    @Autowired
    private UserService userService;

    @PostMapping("/main")
    public void getUser(@RequestParam String firstName){
        userService.findUser(firstName);
    }

}
```

Клас RegistrationController

```
@Controller
public class RegistrationController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("/registration")
    public String registration() {
        return "registration";
    }

    @PostMapping("/registration")
    public String addUser(User user, Map<String, Object> model) {
```

```
        User userFromDB = userRepository.findByUsername(user.getUsername());
        if (userFromDB != null) {
            return "registration";
        }
        user.setActive(true);
        user.setRoles(Collections.singleton(Role.USER));
        userRepository.save(user);
        return "redirect:/login";
    }
}
```

Клас MainController

```
@Controller
public class MainController {
    @Autowired
    private UserMailService userMailService;

    @GetMapping("/")
    public String indexPage() {
        return "index";
    }

    @GetMapping("/main")
    public String mainPage() {
        return "main";
    }

    @GetMapping("/forMan")
    public String forManPage() {
        return "for_man";
```

```
}

@GetMapping("/forWoman")
public String forWomanPage() {
    return "for_woman";
}

@GetMapping("/sale")
public String salePage() {
    return "sale";
}

@GetMapping("/aboutUs")
public String aboutUsPage() {
    return "about_us";
}

@GetMapping("/customerService")
public String customerServicePage() {
    return "customer_service";
}

@GetMapping("/firstManShoes")
public String firstManShoesPage(Model model) {
    return "first_man_shoes";
}

@GetMapping("/secondManShoes")
public String secondManShoesPage() {
    return "second_man_shoes";
}
```

```
    @GetMapping("/thirdManShoes")
    public String thirdManShoesPage() {
        return "third_man_shoes";
    }

    @GetMapping("/firstWomanShoes")
    public String firstWomanShoesPage() {
        return "first_woman_shoes";
    }

    @GetMapping("/secondWomanShoes")
    public String secondWomanShoesPage() {
        return "second_woman_shoes";
    }

    @GetMapping("/thirdWomanShoes")
    public String thirdWomanShoesPage() {
        return "third_woman_shoes";
    }
    @GetMapping("/chat")
    public String chatPage(){
        return "chat";
    }
    @PostMapping("/sendMail")
    public String sendMail(@RequestParam String userMail) {
        userMailService.sendMailToDatabase(userMail);
        return "main";
    }
}
```

Клас ChatController

```
@Controller
public class ChatController {
    @MessageMapping("/chat.sendMessage")
    @SendTo("/topic/public")
    public ChatMessagePojo sendMessage(@Payload ChatMessagePojo chatMessagePojo) {
        return chatMessagePojo;
    }

    @MessageMapping("/chat.addUser")
    @SendTo("/topic/public")
    public ChatMessagePojo addUser(@Payload ChatMessagePojo chatMessagePojo,
SimpMessageHeaderAccessor headerAccessor) {
        headerAccessor.getSessionAttributes().put("username",
chatMessagePojo.getSender());
        return chatMessagePojo;
    }
}
```

Клас WebSocketConfig

```
@Configuration
@EnableWebSocketMessageBroker
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws").withSockJS();
    }

    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry) {
```

```
        registry.setApplicationDestinationPrefixes("/app");
        registry.enableSimpleBroker("/topic");
    }
}
```

Клас WebSecurityConfig

```
@Configuration
@EnableWebSecurity
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {
    @Autowired
    private DataSource dataSource;

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http

                .authorizeRequests()
                .antMatchers("/", "/registration").permitAll()
                .anyRequest().authenticated()
                .and()
                .formLogin()
                .loginPage("/login")
                .permitAll()
                .and()
                .logout()
                .permitAll();
    }

    @Override
    public void configure(WebSecurity web) {
```

```
        web.ignoring()
                .antMatchers(
                        "/css/**", "/fonts/**",
                        "/img/**");
    }

    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {
        auth.jdbcAuthentication()
                .dataSource(dataSource)
                .passwordEncoder(NoOpPasswordEncoder.getInstance())
                .usersByUsernameQuery("select username, password, active from usr
where username=?")
                .authoritiesByUsernameQuery("select u.username, ur.roles from usr u
inner join user_role ur on u.id=ur.user_id where u.username=?");

    }
}
```

Клас MvcConfig

```
@Configuration
public class MvcConfig implements WebMvcConfigurer {
    public void addViewControllers(ViewControllerRegistry registry) {
        registry.addViewController("/login").setViewName("login");
    }
}
```

ДОДАТОК Г

(обов’язковий)

ПРЕЗЕНТАЦІЙНІ МАТЕРІАЛИ

Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

Дипломна робота на тему: Інтернет-платформа магазин з продажу взуття.

Виконав: Федоренко В.В
Керівник: Бедратюк Л.П

Актуальність дипломного проекту:

На теперішній час цифровізація так поспішно увійшла у наше життя, що не можливо уявити надання послуг без застосування сучасних технологій особливо це стосується галузі продажу товару.

Актуальність теми полягає в тому, щоб розвивати і зберігати бізнес прибутковим потрібно долучати технології і цифрувати свій товар і продавати його через інтернет магазини. Отже актуальним є питання розробки програмного забезпечення, яке конкурувало за рахунок зручності та якості надання послуг.

Мета проекту і завдання проекту

Метою дипломного проекту є розробка інтернет-платформи магазину з продажу взуття. Для досягнення поставленої мети були сформовані завдання на дипломне проектування:

- Дослідити предметну область торгівельного бізнесу та виявити потреби потенційних користувачів програмного продукту;
- Проаналізувати наявне програмне забезпечення в цій галузі та сформулювати вимоги на розробку нового програмного забезпечення;
- Спроектувати структуру системи та розробити базу даних;
- Вибрати технології для розробки інтернет-платформи;
- Розробити та протестувати інтернет-платформу магазину з продажу взуття.

Етапи створення

Робота над інтернет-платформою містить декілька етапів:

- Проектування;
- Інформаційний дизайн;
- Графічний дизайн;
- Створення проекту;
- Тестування.

Проект містить в собі

- Дослідження процесів, які належать автоматизації, кінцеве визначення цілей, ідей сайта;
- Розробка архітектури сайта, бази даних;
- Аналіз вимог і створення інтерфейсу, функціональних елементів, інформаційних наповнень;
- Уточнення вимог для створення сайта в відповідності до результатів проведених аналізів наявних платформ.

Аналіз існуючих рішень

На ринку України і за кордоном працює багато магазинів взуття. Відповідно кожен магазин нав'язує своє бачення ідеальної інтернет платформи, але більшість перелік функцій справжньої платформи збігається з тим варіантом, який вони реалізували в своєму програмному продукті.

Розглянемо існуючі інтернет-платформи:

- Staff
- Sezon
- Nike
- InterTop
- Estro

Графічний і інформаційний дизайн

- Під інформаційним дизайном інтернет-платформи можна розуміти розробку структури , а також художнього і ділового оформлення даної структури, які пов'язані між собою фізичним дисковим простором одного сервера.
- Основною цілю графічного дизайну є привернення уваги аудиторії до конкретної інформації за допомогою запам'ятовуючого дизайну і картинок.

Створення проекту інтернет-платформа

На етапі створення логічної структури і графічного дизайну. Організовуються посилання між сторінками, тобто окремі сторінки оформляють сайт в одне ціле. Йде створення сторінок, програмування, а саме написання функціональної частин. Процес не є творчим, тому що ідеї створення йдуть по шаблонам.

Важливим елементом є не тільки правильний вибір елементів дизайну, але і проектування структури.

Мови програмування

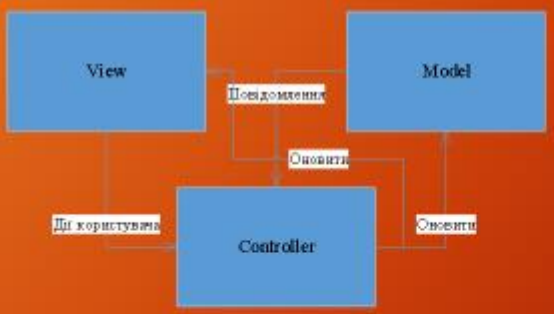
Після вибору структури сайта, можна приступати до його створення. Для цього потрібно вибрати мову програмування на якій буде створений сайт. Язики для створення інтернет-платформ діляться на дві групи:

- Клієнтські: HTML, CSS, JavaScript;
- Серверні: Java.

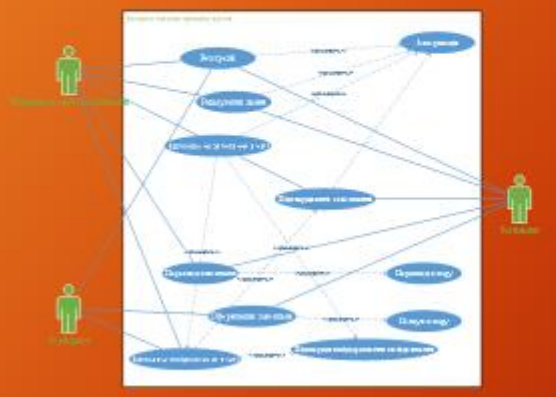
Також були використані фреймворки, які спрощують розробку ПЗ, а саме:

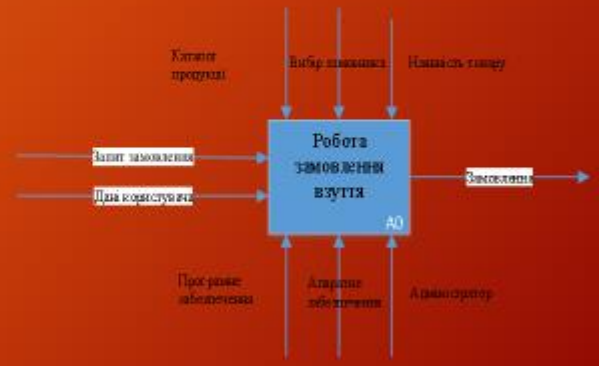
- Фреймворки Spring, Spring Boot, Spring Data, Hibernate, Lombock

Для реалізації програмного продукту вибрано шаблон MVC (Model-View-Controller)

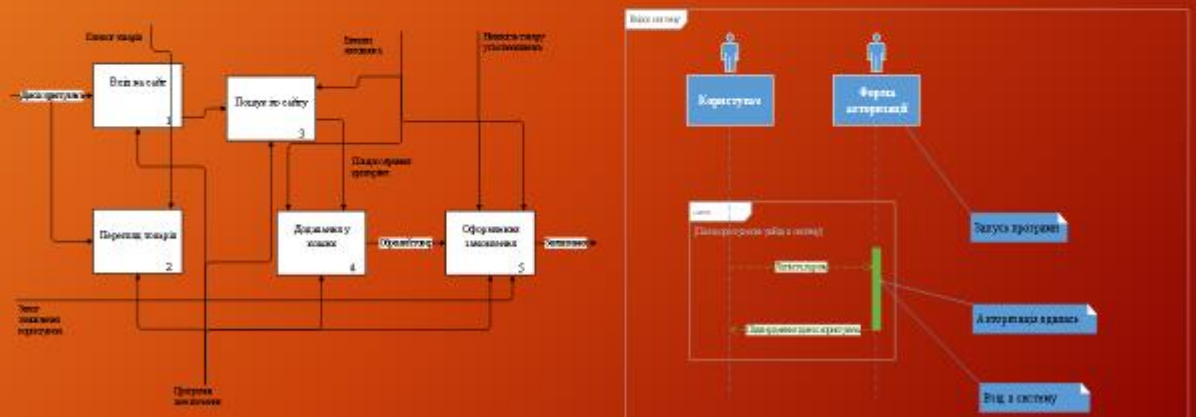


Діаграма варіантів використання і контекстна модель нульового рівня





Контекстна модель першого рівня і діаграма послідовності



Тестування інтернет-платформи

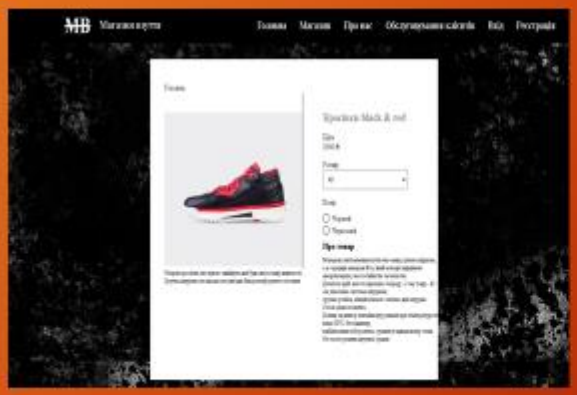
- Тестування на зручність інтерфейсу для використання;
- Перевірка посилань;
- Перевірка HTML-код сторінки сайта;
- Проходження ручного тестування.

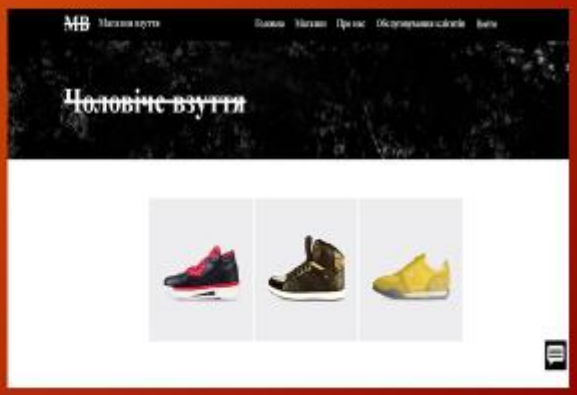
Графічне оформлення

В даній дипломній роботі дизайн інтернет-платформи був створений самостійно, але перед створенням було досліджено величезну кількість уже створених сайтів і шаблонів.

Результати роботи

Результати робота





Висновок

Під-час створення дипломної роботи був аналіз існуючих інтернет-магазинів, сучасних мов програмування та виявлення переваг і функцій з метою їх покращення, а також розробки інтуїтивного зрозумілого онлайн-магазину використовуючи всі набуті знання.

Створений онлайн магазин можна вже зараз використовувати. Окрім того завдяки гнучкості програмного коду можливо з легкістю продовжувати розвивати, оснащувати інтернет-платформу різним функціоналом.

Завідувачу кафедри інженерії програмного забезпечення проф. Бедратюку Л. П.

здобувача вищої освіти

Федоренко В. В.

Прізвище, ініціали

факультет ІТ, 4 курс, група ІПЗ-18-1

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

25.05.22

дата

підпис

Mon May 30 11:48:48 EEST 2022, Хіврич Володимир Русланович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 3.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 10%**

	Документ		Сумарний збіг по Базі Даних	
ID: 104205 Назва: Інтернет-платформа магазин з продажу взуття Додано в БД: 2022-05-30 Автора: В. В. Федоренко Керівники: Л. П. Бедратюк Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	74943	1126	4148 (6%)	58 (5%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK

Ім'я користувача:
Кафедра ІПЗ

ID перевірки:
1011375389

Дата перевірки:
30.05.2022 12:50:21 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
30.05.2022 12:51:27 EEST

ID користувача:
100005589

Назва документа: **Дипломна робота Федоренко В.В ІПЗ-18-1**

Кількість сторінок: **78** Кількість слів: **12904** Кількість символів: **103181** Розмір файлу: **3.17 MB** ID файлу: **1011259821**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

12.8%
Схожість

Найбільша схожість: **4.76%** з джерелом з Бібліотеки (ID файлу: **1011231415**)

7.76% Джерела з Інтернету	324	Сторінка 80
5.86% Джерела з Бібліотеки	80	Сторінка 82

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	19
Підозріле форматування	15 сторінок

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА ДИПЛОМНИЙ ПРОЕКТ
освітнього ступеня «Бакалавр»

Дипломник Федоренко Владислав Вікторович

Тема Інтернет-платформа магазин з продажу взуття

Спеціальність 121 – Інженерія програмного забезпечення

Обсяг дипломного проекту:

Кількість листів креслень _____; кількість сторінок записки _____

1.Короткий зміст пояснювальної записки та прийнятих рішень У дипломному проекті було досліджено і проаналізовано предметну область, усі функціональні та не функціональні вимоги. Було проведено аналіз наявного програмного забезпечення. За результатами аналізу було розроблено програмне забезпечення, яке доводить свою актуальність на сьогоднішній час. Розробка відбувалась з використанням найсучасніших інструментів для розробки інтернет-платформ. Також було проведене ручне тестування, за результатами, якого видно, що програмне забезпечення працює відповідно до вимог дипломного проекту.

2. Висновок про відповідність проекту поставленому завданню Дипломна робота освітнього ступеня «бакалавр» відповідай у повній мірі поставленому завданню, як в практичній частинні так і в теоретичній частинні роботи.

3. Характеристика виконання кожного розділу проекту, ступінь використання останніх досягнень науки і техніки та передових методів роботи У вступі доведено, що дана тема актуальна. У першому розділі дипломної роботи аналізується предметна область та аналіз наявного програмного забезпечення, які допомогли визначити завдання та цілі проекту, також у даному розділі описуються вимоги до програмного забезпечення. У другому розділі аналізуються та впроваджуються вимоги, проектується архітектура програмного забезпечення. У третьому розділі проведені всі можливі етапи програмної реалізації, підготовленні всі можливі залежності, описано їх особливості, в результаті чого був написаний програний продукт. У четвертому розділі виконано ручне тестування програмного забезпечення, що підтверджує працездатність роботи програмного продукту.

4. Позитивні сторони проекту Тематика дипломного проекту досить актуальна на сьогоднішній день, оскільки на сьогоднішній день оскільки всі стаціонарні магазину, які не продають свій товар в інтернеті втрачають дуже велику кількість потенційних клієнтів. Для розробки програмного забезпечення було застосовано сучасні технології розробки . Програмний продукт містить достатній функціонал, щоб задовільнити тему дипломної роботи.

5. Негативні сторони проекту У проекті було б доцільно створити форму для замовлення, щоб користувач міг заповнити данні для замовлення, щоб опростили замовлення для клієнта і не потрібно було підтверджувати статус замовлення з клієнтом, також було б доцільно створити меню бистрого пошуку товару.

6. Оцінка графічного оформлення та пояснювальної записки проекту Графічне оформлення виконано відповідно до теми дипломного проекту та подано у вигляді рисунки та діаграм. Пояснювальна записка оформлена згідно усіх вимог та стандартів.

7. Відгук про дипломний проект в цілому Дипломний проект повноцінно описує вирішення всіх задач та завдань. Матеріал чіткий та структурований, який повністю розкриває тему дипломного проекту. Графічний матеріал роботи дає змогу наочно побачити деталі системи.

8. Інші зауваження

9. Оцінка дипломного проекту Позитивні та негативні сторони дипломного проекту, дає можливість зробити висновок, що дипломний проект виконано в повному обсязі та відповідає поставленій задачі та заслуговує на оцінку «відмінно».

РЕЦЕНЗЕНТ (прізвище, ім'я, по-батькові, посада, місце роботи)
Чешун Віктор Миколайович, кандидат технічних наук, доцент кафедри КБ, Хмельницький національний університет.

"30" травня 2022 р.
(підпис)

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ ..

КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПРО ДОПУСК КФАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: «Інтернет-платформа магазину з продажу взуття»

Автор: Федоренко Владислав Вікторович

Спеціальність: 121 – Інженерія програмного забезпечення

Освітня програма: Освітньо-професійна програма «Інженерія програмного забезпечення»

Науковий керівник: Бедратюк Леонід Петрович, д-р фіз.-мат. наук, професор

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідальність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи.	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту(наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) у тексті дипломного проекту системами перевірки на плагіат виявлено схожість з деякими документами в частині загальновживаних обов'язкових словосполучень у стандартних бланках (титулка, бланк завдання на проектування, відомість документів, у структурі змісту, написах в рамках, назвах розділів/підрозділів тощо) та в назвах переліку джерел посилання;

2) в якості запозичень системою було зафіксовано деякі послідовності вихідного коду і посилання на бібліотеки, які є стандартними мовними конструкціями програмування та не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;

3) усі запозичення є фрагментарними або мають належним чином оформлені посилання;

4) виявлені модифікації тексту не впливають на відсоток схожості.

Сумарний обсяг всіх запозичень, визначений системою виявлених збігів ідентичності схожості, складає 12.8% і адресується до 324 джерел, що з урахуванням наведених обгрунтувань, відповідає характеру теми і свідчить на користь дипломного проекту

Керівник _____ Л. П. Бедратюк

Гарант ОП _____ Л. П. Бедратюк

Завідувач кафедри _____ Л. П. Бедратюк