

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

«Платформа оголошень з продажу товарів»

Виконав студент групи ІПЗ-21-4
ДАНИЛЬЧУК Володимир Богданович

Керівник роботи:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
ТОЛСТОЙ Ігор Анатолійович

Житомир – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії програмного
забезпечення

Тетяна Вакалюк

«26» лютого 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ДАНИЛЬЧУК Володимир Богданович**

Керівник роботи: **ЄФРЕМОВ Юрій Миколайович**

Тема роботи: **«Платформа оголошень з продажу товарів»,**

затверджена Наказом закладу вищої освіти від **«14» лютого 2025 р., №61/с**

Вихідні дані для роботи: **платформа оголошень як цифровий інструмент, що
використовується для комунікації між продавцями та потенційними покупцями**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх
стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Юрій ЄФРЕМОВ	26.02.2025р.	26.02.2025р.
2	Юрій ЄФРЕМОВ	26.02.2025р.	26.02.2025р.
3	Юрій ЄФРЕМОВ	26.02.2025р.	26.02.2025р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з веб-додатку для оголошень з продажу товарів та пояснювальної записки. Пояснювальна записка до випускної роботи містить 64 сторінок, 46 ілюстрацій та 13 таблиць.

Метою роботи є створення платформи оголошень з продажу товарів з використанням сучасних технологій.

Архітектура платформи з продажу товарів базується на сервері на якому розміщена база даних, веб-додатку, який використовується як графічний інтерфейс та на серверній частині, яка займається обрахунками даних.

Додаток має зручний інтерфейс, який дозволяє користувачам легко взаємодіяти з товарами інших користувачів та виставляти свої товари.

КЛЮЧОВІ СЛОВА: ВЕБ-ДОДАТОК, ПЛАТФОРМА, БАЗА ДАНИХ, СЕРВЕР.

					ІПЗ.КР.Б – 121 – 25 – ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата					
Розроб.		Данильчук В.Б.			Платформа оголошень з продажу товарів	Літ.	Арк.	Аркушів	
Керівник.		Єфремов Ю.М.					3	89	
Рецензент.		Толстой І.А.				Житомирська політехніка, група ІПЗ-21-4			
Зав. каф.		Вакалюк Т.А.							

ABSTRACT

The bachelor's final qualifying work consists of a web application for product sales announcements and an explanatory note. The explanatory note for the final work contains 64 pages, 46 illustrations and 13 tables.

The goal of the work is to create an advertising platform for the sale of goods using modern technologies.

The architecture of the product sales platform is based on a server hosting the database, a web application used as a graphical interface, and a server-side component that handles data calculations.

The application has a user-friendly interface that allows users to easily interact with other users' products and display their own products.

KEYWORDS: WEB APPLICATION, PLATFORM, DATABASE, SERVER.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПЛАТФОРМИ ОГолошень з ПРОДАЖУ ТОВАРІВ	9
1.1 Постановка завдання	9
1.2 Аналіз аналогів програмного продукту	11
1.3. Вибір архітектури і обґрунтування інструментальних засобів веб- орієнтованої платформи оголошень з продажу товарів	18
Висновки до першого розділу	20
РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАТФОРМИ ОГолошень з ПРОДАЖУ ТОВАРІВ	22
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	22
2.2 Розробка бази даних	31
2.3 Проектування та реалізація алгоритмів роботи платформи	35
2.4 Реалізація функціоналу платформи з оголошень з продажу товарів	37
Висновки до другого розділу	46
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ з ОНЛАЙН ПЛАТФОРМОЮ ОГолошень з ПРОДАЖУ ТОВАРІВ	47
3.1 Порядок встановлення та налаштування параметрів платформи	47
3.2 Структура функціонального інтерфейсу платформи оголошень з продажу товарів	48
3.3 Тестування роботи програмного комплексу	56

Висновки до третього розділу	59
ВИСНОВКИ	61
ЛІТЕРАТУРА	62
ДОДАТКИ	65

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми платформ оголошень з продажу товарів є доволі високою, особливо враховуючи тенденцію глобальної цифровізації. За останні роки триває підвищення продажів через інтернет, тобто все більше людей купують товари онлайн, що робить платформи оголошень дуже важливим інструментом для бізнесу та приватних продавців. Такі платформи, як "OLX", "Prom", "AliExpress" чи "Rozetka", дозволяють як великим підприємствам, так і приватним особам легко продавати різноманітні товари, охоплюючи велику кількість аудиторії, що робить їх більш доступнішими для всіх. Завдяки зручним та зрозумілим інтерфейсам і автоматизації процесів, продавці можуть доволі швидко розміщувати свої оголошення, а покупці — знаходити необхідні їм товари, що дозволяє економити ресурси і час. Платформи оголошень допомагають підприємцям розширювати свою область торгівлі і залучати більшу кількість клієнтів, що позитивно впливає на масштабування власного бізнесу. У період економічних змін і зростання конкуренції, такі платформи стають надзвичайно важливими для підтримки і покращення бізнесу, а також надають змогу до адаптування в сучасних умовах.

Метою випускної кваліфікаційної роботи бакалавра є розробка платформи оголошень з продажу товарів.

Для досягнення поставленої мети потрібно виконати наступні завдання:

Провести огляд функціональності подібних торговельних платформ, методів фільтрування даних про товар та способи реєстрації користувачів на платформі.

Здійснити вибір засобів об'єктно-орієнтованого програмування.

Розробити платформу оголошень з продажу товарів.

Об'єктом дослідження є сама платформа оголошень як цифровий інструмент, що використовується для комунікації між продавцями та потенційними покупцями.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

Предметом дослідження є технологічні аспекти платформи стосовно особливостей розробки і функціонування різних систем та взаємодія користувачів із платформою, тобто способи та моделі використання платформи продавцями та покупцями.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПЛАТФОРМИ ОГОЛОШЕНЬ З ПРОДАЖУ ТОВАРІВ

1.1. Постановка завдання

Метою випускної кваліфікаційної роботи бакалавра є розробка платформи оголошень з продажу товарів, використовуючи сучасні технології, що дозволить користувачам легко і швидко використовувати її.

Сучасні платформа оголошень з продажу товарів надають підприємцям необхідні умови для швидкого старту: мільйонну аудиторію, зручне і швидке розміщення товарів, прості рекламні інструменти та прозорі і зрозумілі умови співпраці.

Платформи оголошень для продажу товарів, такі як OLX, Rozetka та Prom, почали набирати популярність із швидким розвитком інтернету та електронної комерції. В Україні ці платформи стали особливо популярними приблизно у 2010-х роках, коли доступ до інтернету став значно більший, а онлайн-шопінг став неймовірно зручним і доступнішим для потенційних клієнтів.

Платформа оголошень з продажу товарів має представляти собою декілька програмних модулів, що забезпечують різноманітні функції та можливості для покупців та продавців:

- Модуль реєстрації та авторизації: Модуль надає можливість користувачу створювати новий аккаунт, авторизуватися на платформі та редагувати свій профіль.

- Модуль виставлення товару: Модуль забезпечує користувача можливістю, після заповнення полів відповідної форми, створювати своє оголошення на платформі.

- Модуль пошуку оголошення. Модуль дозволяє фільтрувати відображені оголошення за датою виставлення, мінімальною і максимальною ціною, категорією і тегами.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		9

– Модуль комунікації. Модуль надає засоби для взаємодії між продавцем товару та покупцем. Він дозволяє користувачам спілкуватися, задавати питання, та коментувати різних продавців, товари або інші оголошення.

– Модуль адміністрування. Модуль забезпечує адміністрування додатку, що дає можливість перевіряти кожне оголошення та переглядати статистику.

В результаті повинно бути створено веб-додаток на якому користувач може створити своє оголошення або знайти відповідний товар. Також важливою складовою є комунікація користувачів один між одним.

Основними етапами розробки платформи є:

1. Розробка дизайну платформи:

- створення загальної концепції дизайну веб-додатку;
- створення UI компонентів;
- розробка дизайну сторінок.

2. Побудова архітектури клієнтської частини додатку

3. Побудова архітектури серверної частини додатку.

4. Розробка системи авторизації та ролей користувачів.

5. Розробка основного функціоналу:

- створення логіки авторизації користувачів з відповідними ролями;
- створення віддаленої бази даних для зберігання інформації;
- імплементація бекенду на клієнтській стороні;
- організація спілкування серверної частини з клієнтською;
- фільтрація оголошень;
- створення оголошень;
- редагування оголошень.

6. Тестування додатку:

- функціональне тестування додатку;
- тестування графічного інтерфейсу.

Результатом виконання плану всіх вище перерахованих дій є створення платформи оголошень з продажу товарів, за допомогою якої користувачі можуть створювати відповідні оголошення, щоб в подальшому покупці змогли придбати товар за цими оголошеннями.

1.2. Аналіз аналогів програмного продукту

Почнемо аналіз аналогічних платформ, для того щоб дізнатися які функціональні можливості повинні бути розроблені для нашої платформи та які недоліки ми повинні обміркувати і виправити під час розробки програмного забезпечення.

Першою платформою оголошень яку ми будемо аналізувати є веб-сторінка під назвою OLX (Рис 1.1), посилання на цю веб-сторінку: <https://www.olx.ua/uk/>.

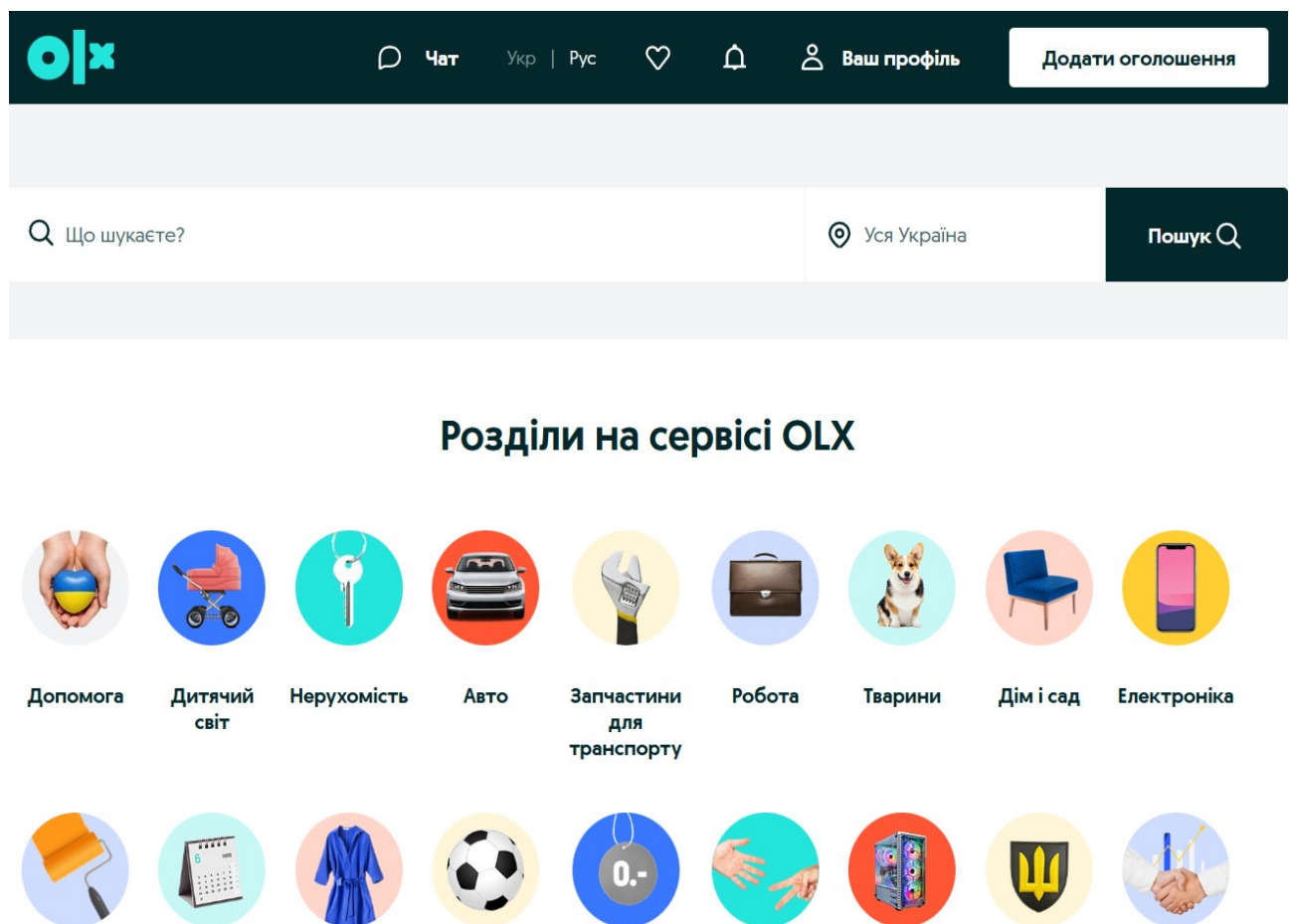


Рис. 1.1 Платформа оголошень OLX

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		11

OLX — це платформа онлайн-оголошень, яка дозволяє користувачам купувати, продавати або обмінювати товари та послуги. Вона була заснована у 2006 році і стала частиною міжнародної мережі OLX Group, яка представлена у понад 40 країнах світу. В Україні OLX є одним із найпопулярніших сервісів оголошень, де кожен може знайти або запропонувати практично все — від нерухомості до хобі та спорту.

Основними перевагами платформи OLX є:

Платформа має мільйони користувачів щомісяця, що забезпечує великий потенціал для продажів.

Інтуїтивно зрозумілий інтерфейс дозволяє легко розміщувати власні оголошення.

OLX пропонує різні формати реклами, включаючи таргетинг та ремаркетинг.

Платформа надає інструменти для створення власних бізнес-сторінок та магазинів.

Використання OLX не потребує значних витрат на веб-розробку.

Недоліками платформи є:

Велика кількість продавців може ускладнити видимість вашого оголошення.

Деякі функції доступні лише за додаткову оплату.

Як і на будь-якій платформі, існує ризик зіткнутися з недобросовісними користувачам.

Далі ми розглянемо платформу під назвою prom (Рис.1.2) за посиланням <https://prom.ua>.

Prom — це найбільший маркетплейс в Україні, створений для роздрібно́ї та оптової торгівлі товарами та надання послуг. На платформі представлено понад 100 мільйонів товарів від десятків тисяч підприємців. Prom дозволяє продавцям створювати власні інтернет-магазини, а покупцям — легко знаходити потрібні товари за доступними цінами.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		12

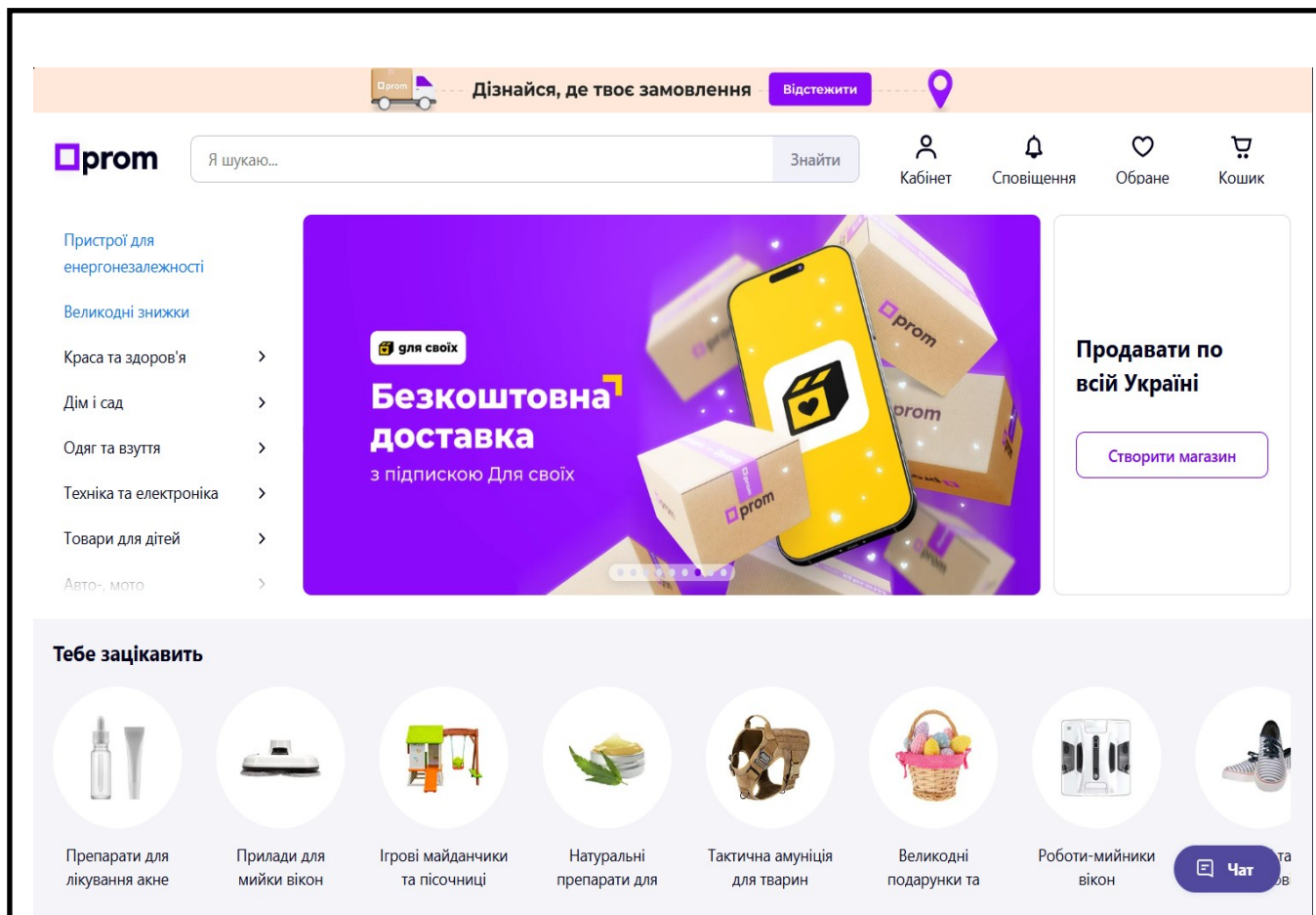


Рис. 1.2 Платформа оголошень prom

Основними перевагами платформи prom є:

На платформі представлено мільйони товарів, що дозволяє знайти практично все що завгодно.

Можливість створення власного інтернет-магазину та зручна інтеграція з іншими маркетплейсами.

Система рейтингу та відгуків допомагає завоювати довіру потенційних покупців.

Персональні менеджери супроводжують продавців на кожному етапі створення своєї сторінки.

Гнучкі варіанти оплати та доставки наприклад, функція "ПромОплата", яка гарантує безпеку транзакцій.

Недоліками платформи є:

Велика кількість продавців на платформі може ускладнити видимість вашого магазину.

Для повноцінного використання платформи оголошень потрібні додаткові витрати.

Функціонал для користувачів, які не є підприємцями значно обмежений.

Деякі користувачі скаржаться на дуже велику тривалість перевірки товарів.

Тепер ми можемо переходити до аналізу третьої. Її назва Rozetka (Рис 1.3) та посилання на неї <https://rozetka.com.ua>.

Rozetka — це один із найбільших онлайн-ритейлерів України, заснований у 2005 році. Спочатку платформа спеціалізувалася на продажу техніки, але з часом стала маркетплейсом, де можна знайти практично все: від електроніки до продуктів харчування. В 2016 році платформа додала функціональність торговельного майданчика, з того часу на сайті інші користувачі мають можливість продавати власні товари. Rozetka також має офлайн-магазини та пропонує зручні варіанти доставки й оплати.

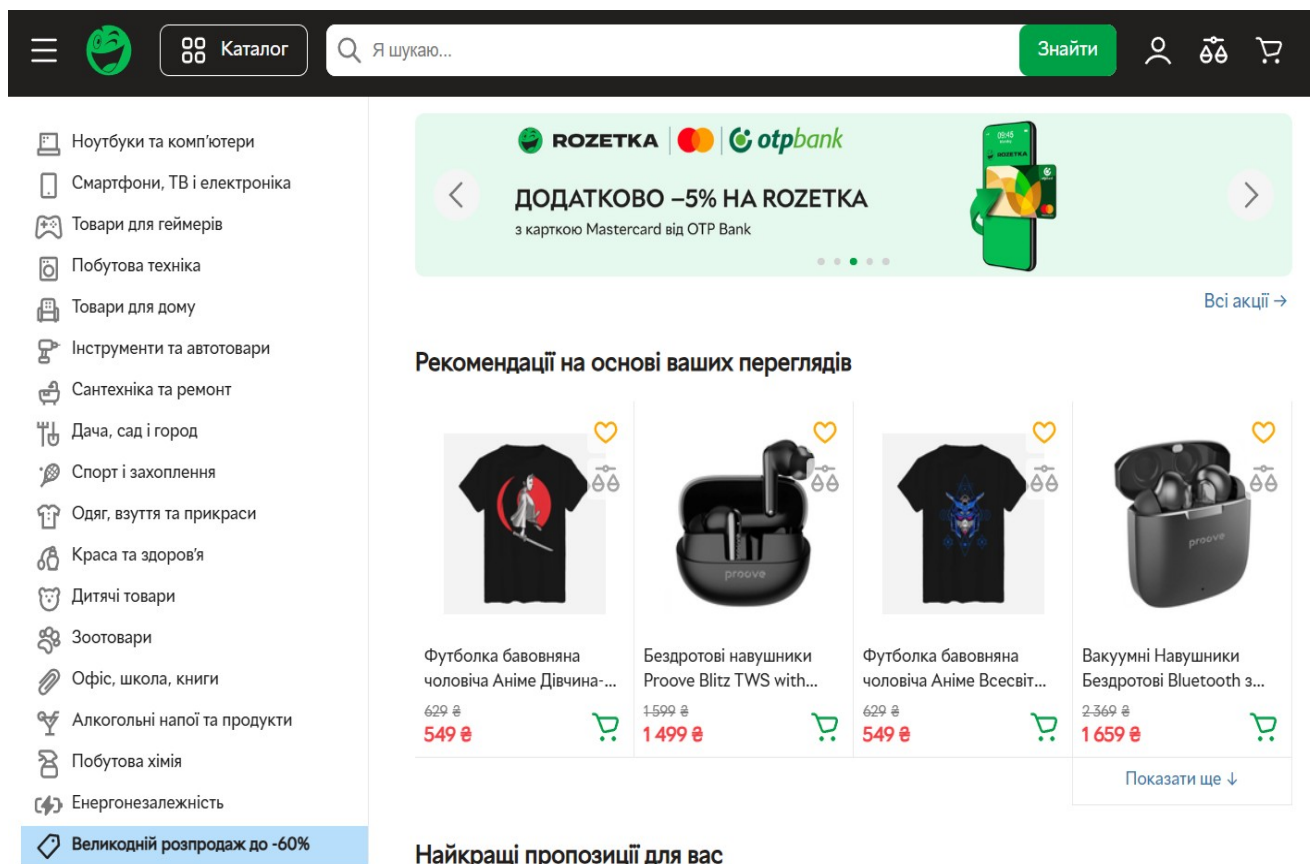


Рис. 1.3 Платформа оголошень Rozetka

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		14

Основними перевагами платформи Rozetka є:

На платформі представлено дуже велика клікість категорій, що дозволяє знайти практично все що завгодно.

Інтуїтивно зрозумілий дизайн сайту та мобільний додаток спрощують процес придбання товару.

Rozetka має власні пункти видачі та співпрацює з різними службами доставки.

На платформі часто проводяться акції та знижки, що робить покупки дуже вигідними.

Можливість, за потреби, повернення товару та гарантія на більшість продукції.

Недоліками платформи є:

Велика кількість продавців може ускладнити вибір.

Деякі товари можуть бути дорожчими порівняно з іншими платформами.

Проблеми з модерацією можуть призвести до затримки у перевірці товарів.

Обмеження безкоштовної доставки, оскільки безкоштовна доставка доступна лише для дорожчих товарів.

Наступна четверта платформа оголошень буде останньою яку ми проаналізуємо. Її назва Instagram (Рис 1.4) та посилання на неї <https://www.instagram.com>.

Instagram — це популярна соціальна мережа, яка дозволяє користувачам ділитися фотографіями та відео, застосовувати різні фільтри для їх обробки, а також взаємодіяти з іншими користувачами через коментарі, лайки та приватні повідомлення. Instagram є чудовим інструментом для продажу товарів, особливо якщо ви продаєте модний одяг, аксесуари чи речі ручної роботи. Бізнес-профіль у цій соціальній мережі можна створити без жодних витрат. Ця платформа належить компанії Meta Platforms і є одним із найпопулярніших сервісів для обміну різноманітним візуальним контентом.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		15

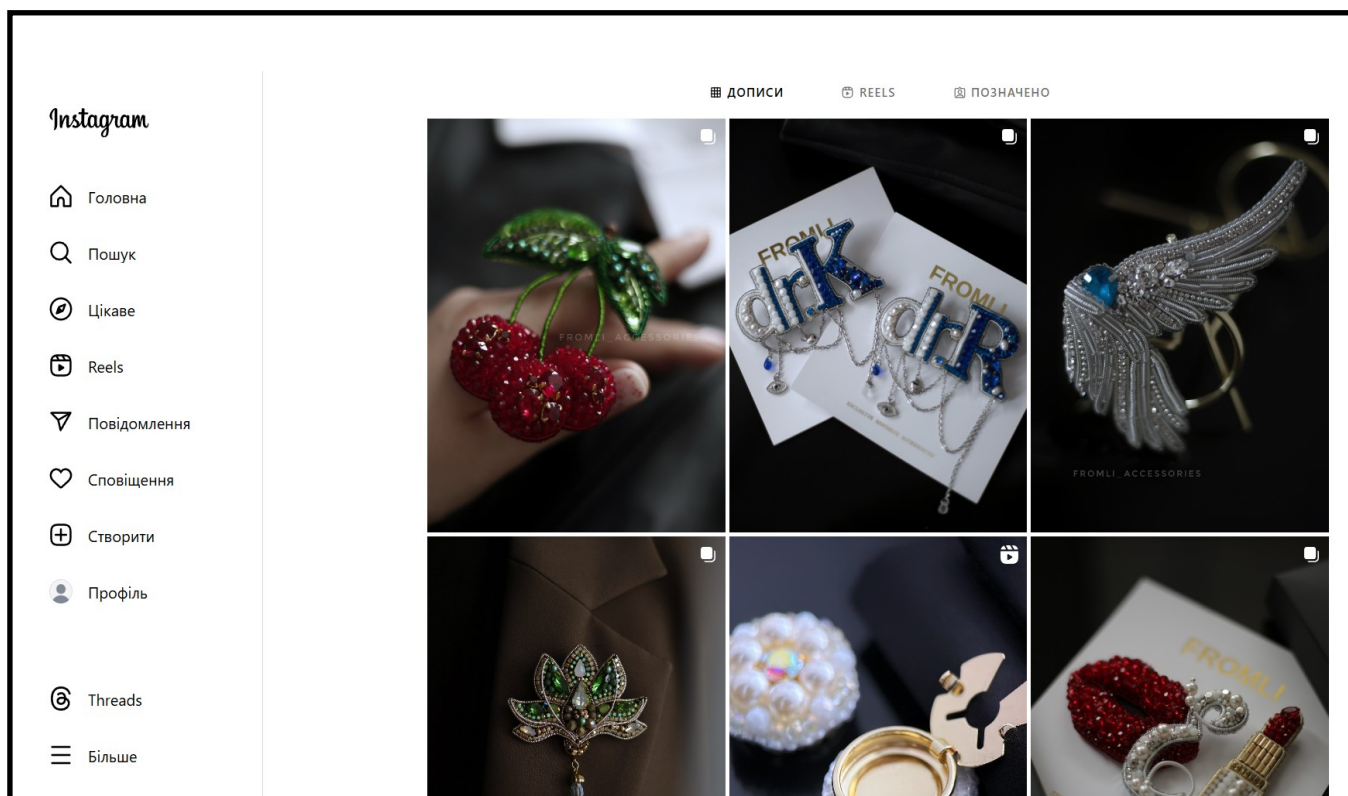


Рис. 1.4 Платформа оголошень Instagram

Основними перевагами платформи Instagram є:

Instagram має мільярди користувачів, що дозволяє охоплювати широку аудиторію.

Платформа орієнтована на фото й відео, що дає можливість показати товар у привабливому світлі.

Є можливості для використання таргетованої реклами та бізнес аналітики на платформі, які допомагають ефективно знаходити потрібну аудиторію під товари.

Інтеграція магазину прямо у профіль, де користувачі можуть купувати товари безпосередньо.

Лайки, коментарі та Direct Messages сприяють комунікації та побудові відносин із клієнтами.

Недоліками платформи є:

Велика кількість продавців може ускладнити видимість вашого профілю.

Постійні зміни в алгоритмах можуть впливати на охоплення вашого контенту.

Ефективна реклама може бути дорогою, особливо в конкурентних областях.

Підтримка активного профілю, створення контенту та відповіді на повідомлення можуть забирати багато часу.

У порівнянні з повноцінними онлайн-магазинами, можливості Instagram для управління продажами є дещо обмеженими.

Виходячи з наявних переваг та недоліків, складемо порівняльну таблицю вищезгаданих програмних аналогів (табл. 1.1).

Таблиця 1.1

Таблиця порівняння платформ оголошень

Характеристика	OLX	prom	Rozetka	Instagram
Зрозумілий дизайн	+	+	+	-
Проста в використанні	+	+	+	-
Система комунікації	+	+	-	+
Гнучкі варіанти оплати	+	+	+	-
Розширений інструментарій за додаткову оплату	-	-	-	+

Переглянувши різні платформи оголошень можна виділити основні відмінності: деякі з них дозволяють отримати повний інструментарій по розміщуванню і створенню оголошень лише після придбання повного доступу до цього інструментарію, у деяких платформ проблема з дуже довгою модерацією оголошень, всі платформи мають дуже великий список оголошень із-за чого складно знайти конкретний товар, а також їх всіх об'єднує легкий і інтуїтивно зрозумілий дизайн.

Таким чином, дана платформа має бути дуже простою у використанні, з інтуїтивно зрозумілим дизайном, велика кількість категорій для зручного пошуку потрібного оголошення, впроваджена система комунікації між продавцем і покупцем, а також гнучкі варіанти оплати, які гарантують безпеку транзакцій.

1.3. Вибір архітектури і обґрунтування інструментальних засобів веб-орієнтованої платформи оголошень з продажу товарів

Платформа оголошень буде побудований на клієнт-серверній архітектурі. Серверна частина буде представлена у вигляді API та буде написана з використанням сучасного вільного фреймворку laravel. Для зберігання інформації буде використовуватись база даних MariaDB.

Laravel — це сучасний, розширюваний та потужний фреймворк для веб-розробки на PHP, який значно спрощує створення складних веб-застосунків. Він базується на архітектурі MVC (Model-View-Controller), що забезпечує розділення логіки застосунку, полегшує підтримку коду та покращує організацію проекту.

Переваги Laravel:

Laravel робить роботу з кодом набагато простішим і інтуїтивно зрозумілим завдяки чітким, лаконічним та виразним стандартизованим конструкціям.

Laravel використовує Eloquent ORM — це зручна об'єктно-реляційна модель, що дозволяє працювати з базами даних за допомогою PHP-об'єктів, а також використовує міграції та фабрики даних, що надають змогу без великих зусиль змінювати структуру бази безпосередньо з коду.

Laravel включає вбудований захист від: SQL-ін'єкцій, XSS-атак, CSRF-атак та захист даних через хешування та шифрування.

Laravel має свій потужний механізм Blade, який дозволяє легко створювати шаблони з вбудованими динамічними можливостями.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		18

Laravel використовує черги, що дозволяють керувати фоновими процесами і покращують продуктивність та використовує планування завдань, що надає змогу запускати скрипти в заданий час без використання зовнішніх засобів типу cron.

Laravel має власну платформу для пакетів (Packagist), що надає змогу легко додавати різноманітні нові функції до вашого застосунку.

Фреймворк містить зручні механізми для створення REST API, обробки запитів та роботи з JSON-даними.

Узагальнюючи можна сказати що, Laravel є потужним PHP-фреймворком для веб-розробки, заснований на архітектурі MVC. Він забезпечує зручний синтаксис, ефективну роботу з базами даних через Eloquent ORM, високий рівень безпеки, гнучку шаблонізацію Blade, автоматизацію процесів та просту інтеграцію з API. Laravel підходить як для невеликих проектів, так і для масштабних корпоративних рішень завдяки своїй розширюваності та продуктивності [13].

MariaDB — це реляційна система керування базами даних, яка була створена у 2009 році як відгалуження MySQL. Вона розробляється під відкритою ліцензією GNU GPL і підтримує високу сумісність із MySQL, що дозволяє легко і швидко здійснити міграцію між цими двома системами. MariaDB включає покращені рушії баз даних, такі як Aria та XtraDB. Крім того, вона має додаткові можливості, зокрема віртуальні стовпці, покращену аутентифікацію та підтримку мікросекунд.

Переваги MariaDB:

MariaDB зберігає ту саму функціональність, API та команди, що й MySQL, що полегшує перехід з MySQL та використання вже існуючих інструментів та скриптів.

MariaDB розповсюджується за ліцензією GPLv2 з відкритим вихідним кодом і має велику та активну спільноту, що сприяє підтримці та розвитку бази даних.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		19

MariaDB активно розробляється, з регулярними оновленнями та виправленнями, що забезпечує стабільну та безпечну роботу з нею.

MariaDB підтримує кілька механізмів зберігання, включаючи InnoDB та ColumnStore, що дозволяє вибирати оптимальний для використання варіант згідно ваших потреб.

MariaDB часто показує кращі результати у швидкості запитів та масштабованості порівняно з MySQL, особливо для роботи з дуже великими обсягами даних.

MariaDB додає нові функції, такі як звіти про хід виконання операцій та підтримку не блокуючих операцій, які не завжди є доступними в MySQL.

MariaDB підтримує формат зберігання даних в колонках, що може бути корисним для аналітики та роботи з дуже великими обсягами даних.

MariaDB може виконувати деякі запити паралельно на різних серверах, що підвищує її продуктивність.

MariaDB добре масштабується, як вертикально, так і горизонтально.

MariaDB — це відкрите та потужне відгалуження MySQL, яке зберігає сумісність із MySQL, пропонує покращену продуктивність та розширені можливості. Воно має нові рушії баз даних, підтримку віртуальних стовпців, удосконалену аутентифікацію та багато інших корисних функцій [14].

Висновки до першого розділу

У першому розділі було проведено аналіз предметної області за допомогою якого були визначені основні вимоги до платформи оголошень, основним функціоналом якої є можливість створення відповідного оголошення та можливість пошуку потрібного оголошення за допомогою фільтрації.

В процесі дослідження аналогічних і схожих за ідеєю продуктів були проаналізовані та порівняні різні платформи з різними принципами роботи, основним функціоналом яких є комунікація покупця і продавця, створення оголошень та їх фільтрація. Крім цього, були сформовані основні вимоги для власної платформи.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		20

На основі відповідних вимог та критеріїв був сформований набір основних технологій і засобів реалізації для розробки платформи.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						21
Змн.	Арк.	№ докум.	Підпис	Дата		

Функціональність платформи оголошень залежить від активності користувачів. Користувачі можуть бути розділені на дві основні групи: звичайні користувачі, які бажають створити власне оголошення або знайти товар по вже створеному оголошенню, та адміністратори, які будуть взаємодіяти з замовленнями користувачів. Оголошення містять деталі про предмет продажу, його зображення, його описані характеристики, ціну на цей предмет, місто в якому знаходиться володар оголошення та інша важлива інформація, яка може зацікавити потенційного покупця.

Більш детальний опис діаграми варіантів використання наведено в таблицях з 2.1 по 2.7.

Таблиця 2.1

Опис прецеденту «Реєстрація/Авторизація»

Назва варіанту використання	Реєстрація/Авторизація
Основні актори	Адміністратор, авторизований користувач, користувач
Короткий опис	Забезпечує користувачам доступ до платформи через створення облікового запису або вхід в існуючий за допомогою електронної пошти чи сторонніх сервісів, таких як соціальні мережі, для швидкого та безпечного використання можливостей платформи оголошень

Таблиця 2.2

Опис прецеденту «Створення оголошення»

Назва варіанту використання	Створення оголошення
Основні актори	Адміністратор, авторизований користувач
Короткий опис	Користувачам можуть легко і інтуїтивно розміщувати свої оголошення, вводити потрібну інформацію, додавати зображення товару, публікувати, а в подальшому, за потреби, редагувати на сайті свої оголошення, що забезпечує швидкий доступу до потенційних покупців.

Таблиця 2.3

Опис прецеденту «Пошук оголошення»

Назва варіанту використання	Пошук оголошення
Основні актори	Адміністратор, авторизований користувач, користувач
Короткий опис	Забезпечує користувача швидким і ефективним доступом до потрібних оголошень, дозволяючи користувачам вводити ключові слова, застосовувати фільтри та сортувати результати за відповідними критеріями

Таблиця 2.4

Опис прецеденту «Видалення оголошення»

Назва варіанту використання	Видалення оголошення
Основні актори	Адміністратор, авторизований користувач
Короткий опис	Забезпечує користувача зручною і швидкою можливістю видаляти раніше створене оголошення, якщо така потреба в нього колись виникне

Таблиця 2.5

Опис прецеденту «Створення замовлення»

Назва варіанту використання	Створення замовлення
Основні актори	Авторизований користувач
Короткий опис	Якщо користувачу сподобається якесь оголошення і він буде авторизований на сайті у нього буде можливість створити замовлення цього оголошення, для подальшого його придбання

Таблиця 2.6

Опис прецеденту «Створення особистого запису»

Назва варіанту використання	Створення особистого запису
Основні актори	Адміністратор, авторизований користувач

Короткий опис	Забезпечує користувача створювати свій власний обліковий запис для подальшої взаємодії з функціоналом сайту, а також забезпечує можливість редагувати та, за потреби, видаляти свій профіль
---------------	---

Таблиця 2.7

Опис прецеденту «Додавання оголошення в список збережених»

Назва варіанту використання	Додавання оголошення в список збережених
Основні актори	Адміністратор, авторизований користувач
Короткий опис	Забезпечує користувачів можливістю додавати різні оголошення які сподобалися в список збережених оголошень, для подальшого швидкого доступу до цих оголошень і їх майбутнього перегляду

Основна ціль розробки платформи оголошень полягає у створенні зручного простору для ефективного пошуку, публікації та управління оголошеннями, сприяючи взаємодії між користувачами, підвищенню доступності товарів і послуг, а також забезпеченню безпечної комунікації між продавцями і покупцями. Платформа має допомагати економити час, розширювати можливості користувачів і підтримувати прозорість угод.

Основна мета платформи полягає у створенні інструменту, який дозволяє людям легко взаємодіяти, обмінюватися оголошеннями, знаходити необхідні товари чи послуги та безпечно здійснювати угоди. Платформа орієнтована на зручність, ефективність та підтримку прозорості у процесі комунікації між користувачами.

На нашій платформі повинні бути користувач адміністратор, неавторизований користувач і авторизований користувач.

Користувач адміністратор повинен мати можливість переглядати замовлення користувачів та видаляти їх

Авторизований користувач повинен мати можливість реєстрації і авторизуватися на платформі, створювати, редагувати і видаляти оголошення, шукати потрібного оголошення завдяки різноманітним фільтрам на сайті та можливість зв'язку з власникам оголошень для подальшого його уточнення.

Неавторизований користувач повинен мати можливість перегляду оголошень на платформі, пошуку потрібного оголошення завдяки різноманітним фільтрам на сайті та можливість створити власний обліковий запис.

Згідно з функціональними вимогами в системі повинний бути представлений функціонал реєстрації та авторизації користувача, комунікації користувачів, створення, редагування, видалення і відображення оголошень, додавання оголошень в список збережених та редагування особистого запису.

Згідно з не функціональними вимогами час, необхідний для навчання звичайних користувачів – 3-5 хвилин, а для навчання просунутих користувачів 2 хвилини, час відгуку платформи на прості дії користувача не повинен перевищувати 3 секунд, а для складних дій час не повинен перевищувати 15 секунд, інтерфейс платформи оголошень повинен бути зручним та інтуїтивно зрозумілим для всіх користувачів, та не вимагати від них додаткової підготовки, час, що витрачається на обслуговування всієї загальної системи не повинно перевищувати 2% від загального часу роботи платформи оголошень, середній час безвідмовної роботи – 30 робочих днів, максимальна норма помилок та дефектів в роботі системи – 1 помилка на десять тисяч рядків коду, система повинна підтримувати мінімум одну тисячу одночасно підключених користувачів, пов'язаних із загальною базою даних системи, а також велика кількість користувачів мають змогу переглядати веб-ресурс одночасно.

Згідно з системними вимогами платформа повинна задовільнять пристрої, що знаходиться в комплектації 2 Гб оперативної пам'яті, процесора з тактовою частотою не нижче 1.4 GHz та пристрої з виходом до мережі інтернет.

Опираючись на функціональні характеристики системи адміністраторам повинен бути наданий постійний доступ до інформації про оголошення, вхідні

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		26

дані формуються на інформації з бази даних що отримуються із веб-додатку та орієнтовна кількість записів, що зберігаються в базі даних – 100000.

Аналіз функціональних вимог дозволив виділити наступні сутності, що забезпечать програмну реалізацію додатку. На рисунку 2.2. наведено Діаграму класів, яка використовуються для його роботи.

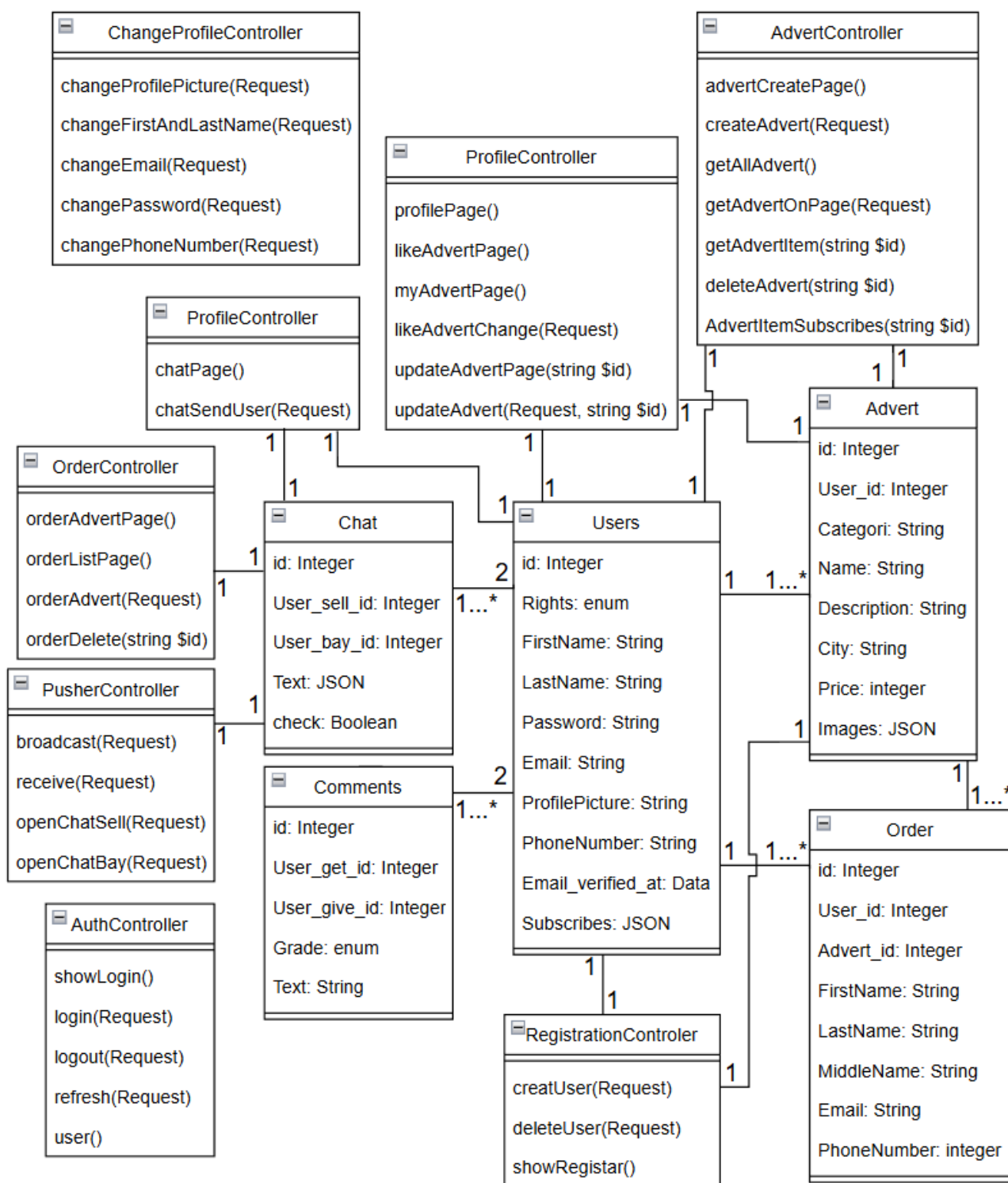


Рис. 2.2. Діаграма класів

Одною з особливостей Laravel є розділення класів на різні сутності. Ми будемо переважно використовувати сутності моделей та контролерів [13].

Класи моделей використовується, як представлення таблиць з бази даних з їх полями та іншими параметрами.

Класи контролерів використовується для обрахунку параметрів, які передаються в функції класів, а також для виведення даних в відповідні шаблони.

Опишемо діаграму класів (рис.2.2).

ProfileController – клас контролер який відповідає за власний профіль користувача:

- profilePage() – відповідає за вивід сторінки з профілем користувача;
- likeAdvertPage() – відповідає за вивід сторінки на якій будуть відображатися всі оголошення, які зберіг користувач;
- myAdvertPage() – відповідає за вивід сторінки на якій будуть відображатися всі оголошення поточного користувача;
- likeAdvertChange(Request) – відповідає за додавання оголошення в список збережених користувачем на сторінці з збереженими оголошеннями;
- updateAdvertPage(string \$id) – відповідає за вивід сторінки з формою для зміни інформації про оголошення;
- updateAdvert(Request, string \$id) – відповідає за збереження даних про оголошення якщо користувач захоче його змінити, за валідацію введених даних, за збереження завантажених зображень на сервері, а також за видалення старих зображень оголошення з серверу.

AdvertController – клас контролер який відповідає за взаємодію з оголошеннями на сайті:

- advertCreatePage() – відповідає за вивід сторінки з формою, за допомогою якої користувач може створити власне оголошення;
- createAdvert(Request) – відповідає за створення власного оголошення користувача та валідацію введених даних;

- getAllAdvert() – відповідає за вивід всіх оголошень на головній сторінці;
- getAdvertOnPage(Request) – відповідає за фільтрацію та вивід всіх оголошень на головній сторінці;
- getAdvertItem(string \$id) – відповідає за вивід конкретного оголошення на новій сторінці;
- deleteAdvert(string \$id) – відповідає за видалення оголошення, а також за видалення зображень оголошення з серверу;
- AdvertItemSubscribes(string \$id) – відповідає за додавання в список збережених оголошень користувача та видалення з нього на сторінці оголошення.

RegistrationControler – клас контролер який відповідає за реєстрацію на сайті:

- createUser(Request) – відповідає за створення нового користувача, валідацію введених даних та його авторизацію;
- deleteUser(Request) – відповідає за видалення поточного користувача, видалення всіх зв'язаних класів моделей з цим користувачем, а також видалення зображень з оголошень користувача та зображення профілю з серверу;
- showRegistar() – відповідає за вивід сторінки з реєстрацією користувача.

PusherController – клас контролер який відповідає за наявні чати користувачів та відправляння повідомлень:

- broadcast(Request) – відповідає за створення та оновлення чату користувачів, а також за відображення власного повідомлень в чаті в реальному часі;
- receive(Request) – відповідає за відображення повідомлення співбесідника в чаті в реальному часі;
- openChatSell(Request) – відповідає за відкриття чату з продавцями на сайті;

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						29
Змн.	Арк.	№ докум.	Підпис	Дата		

- openChatBay(Request) – відповідає за відкриття чату з покупцями на сайті.

OrderController – клас контролер який відповідає за замовлення оголошень:

- orderAdvertPage() – відповідає за вивід сторінки з формою для замовлення оголошення;
- orderListPage() – відповідає за вивід сторінки з списком всіх замовлень користувачів;
- orderAdvert(Request) – відповідає за створення нового замовлення, а також за валідацію отриманих даних з форми
- orderDelete(string \$id) – відповідає за видалення замовлення користувача адміністратором.

ChatController – клас контролер який відповідає за відображення сторінки з чатами користувачів:

- chatPage() – відповідає за вивід сторінки з чатами користувачів;
- chatSendUser(Request) – відповідає за вивід сторінки з чатами користувачів і відображення поточного чату з продавцем.

AuthController – клас контролер який відповідає за деякі можливості авторизованого користувача:

- showLogin() – відповідає за вивід сторінки з формою для авторизації;
- login(Request) – відповідає за авторизацію поточного користувача, генерацію нового ідентифікатора сесії, а також за валідацію отриманих даних з форми;
- logout(Request) – відповідає за вихід користувача з системи сайту, очистку даних сесій, а також за генерування нового CSRF-токену;
- refresh(Request) – відповідає за генерування нового CSRF-токену і за генерацію нового ідентифікатора сесії;
- user() – відповідає за отримання даних про поточного користувача.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						30
Змн.	Арк.	№ докум.	Підпис	Дата		

ChangeProfileController – клас контролер який відповідає за редагування профілю поточного користувача:

- changeProfilePicture(Request) – відповідає за валідацію, видалення старого зображення користувача якщо воно існувало і збереження зображення профілю користувача;
- changeFirstAndLastName(Request) – відповідає за валідацію і збереження нового імені та фамілії;
- changeEmail(Request) – відповідає за валідацію і збереження нової електронної пошти;
- changePassword(Request) – відповідає за валідацію, збереження і хешування нового паролю;
- changePhoneNumber(Request) – відповідає за валідацію і збереження нового номера телефону;

2.2. Розробка бази даних

Для збереження файлів та інформації на платформі оголошень використовуються MariaDB, це популярна система управління реляційними базами даних (СУБД), яка дозволяє зберігати, організовувати та отримувати дані у вигляді взаємопов'язаних таблиць [8, 14, 30].

База даних системи складається з трьох таблиць:

1. Users – таблиця, яка зберігає данні про користувачів;
2. Advert – таблиця, яка зберігає дані про оголошення;
3. Comments – таблиця, яка зберігає дані про коментарі;
4. Chat – таблиця, яка зберігає дані пов'язані з чатами користувачів;
5. Order – таблиця, яка зберігає замовлення.

Опис кожної з таблиць бази даних наведений у таблицях 2.8 – 2.12.

Таблиця 2.8

Опис таблиці Users

Users		
Поле	Тип	Призначення

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

Продовження таблиці 2.8

Id	Integer	Ідентифікатор користувача
Rights	Enum	Права користувача
FirstName	String	Ім'я користувача
LastName	String	Фамілія користувача
Password	String	Пароль користувача
Email	String	Електронна адреса користувача
ProfilePicture	String	Зображення профілю користувача
PhoneNumber	String	Номер телефону користувача
Email_verified_at	Data	Підтвердження електронної пошти користувача
Subscribes	JSON	Список збережених оголошень користувача
created_at	timestamp	Дата створення рядка
updated_at	timestamp	Дата оновлення рядка

Таблиця 2.9

Опис таблиці Comments

Comments		
Поле	Тип	Призначення
Id	Integer	Ідентифікатор коментаря
User_get_id	Integer	Ідентифікатор прокоментованого
User_give_id	Integer	Ідентифікатор коментатора
Grade	Enum	Оцінка за п'ятибальною шкалою
Text	String	Текст коментаря
created_at	timestamp	Дата створення рядка
updated_at	timestamp	Дата оновлення рядка

Таблиця 2.10

Опис таблиці Advert

Advert		
Поле	Тип	Призначення
Id	Integer	Ідентифікатор оголошення
User_id	Integer	Ідентифікатор власника оголошення
Categori	String	Назва категорії
Name	String	Назва оголошення
Description	String	Опис оголошення
City	String	Назва міста
Price	Integer	Ціна на товар з оголошення
Images	JSON	Список зображень
created_at	timestamp	Дата створення рядка
updated_at	timestamp	Дата оновлення рядка

Таблиця 2.11

Опис таблиці Order

Order		
Поле	Тип	Призначення
Id	Integer	Ідентифікатор оголошення
User_id	Integer	Ідентифікатор покупця
Advert_id	Integer	Ідентифікатор оголошення
FirstName	String	Ім'я користувача
LastName	String	Фамілія користувача
MiddleName	String	Ім'я по батькові користувача
Email	String	Електронна адреса користувача

Продовження таблиці 2.11

PhoneNumber	String	Номер телефону користувача
created_at	timestamp	Дата створення рядка
updated_at	timestamp	Дата оновлення рядка

Таблиця 2.12

Опис таблиці Chat

Chat		
Поле	Тип	Призначення
Id	Integer	Ідентифікатор оголошення
User_sell_id	Integer	Ідентифікатор продавця
User_buy_id	Integer	Ідентифікатор покупця
Text	JSON	Список з повідомленнями користувачів
check	Boolean	Прапорець про переглянуті повідомлення
created_at	timestamp	Дата створення рядка
updated_at	timestamp	Дата оновлення рядка

База даних є критично важливим компонентом для будь-якої платформи, включаючи платформу оголошень з продажу товарів. Вона забезпечує впорядковане зберігання всіх даних — від інформації про користувачів до оголошень, замовлень, коментарів та чату. Це дозволяє легко отримувати доступ до необхідної інформації та ефективно керувати даними.

Безпека бази даних є надзвичайно важливою, оскільки вона зберігає конфіденційну інформацію, як-от дані користувачів, історію покупок чи паролі. Шифрування й сучасні методи захисту даних допомагають гарантувати їх недоторканність і збереження довіри користувачів. Крім цього, добре спроектована база даних дозволяє платформі масштабуватись, обробляючи більші обсяги даних без втрати продуктивності.

База даних також є джерелом аналітики, надаючи можливість аналізувати активність користувачів. Вона інтегрує дані з різних елементів системи, забезпечуючи зв'язок між ними, наприклад, визначаючи, хто створив конкретне оголошення. Це справжнє серце системи, яке є основою успішної роботи платформи.

2.3. Проектування та реалізація алгоритмів роботи платформи

Веб-додаток є простим у використанні, має всі основні функції платформи, що дозволяє користувачам легко взаємодіяти з платформою. Алгоритм роботи з платформою зображено на рисунку 2.3, також описано детально його пункти [15].

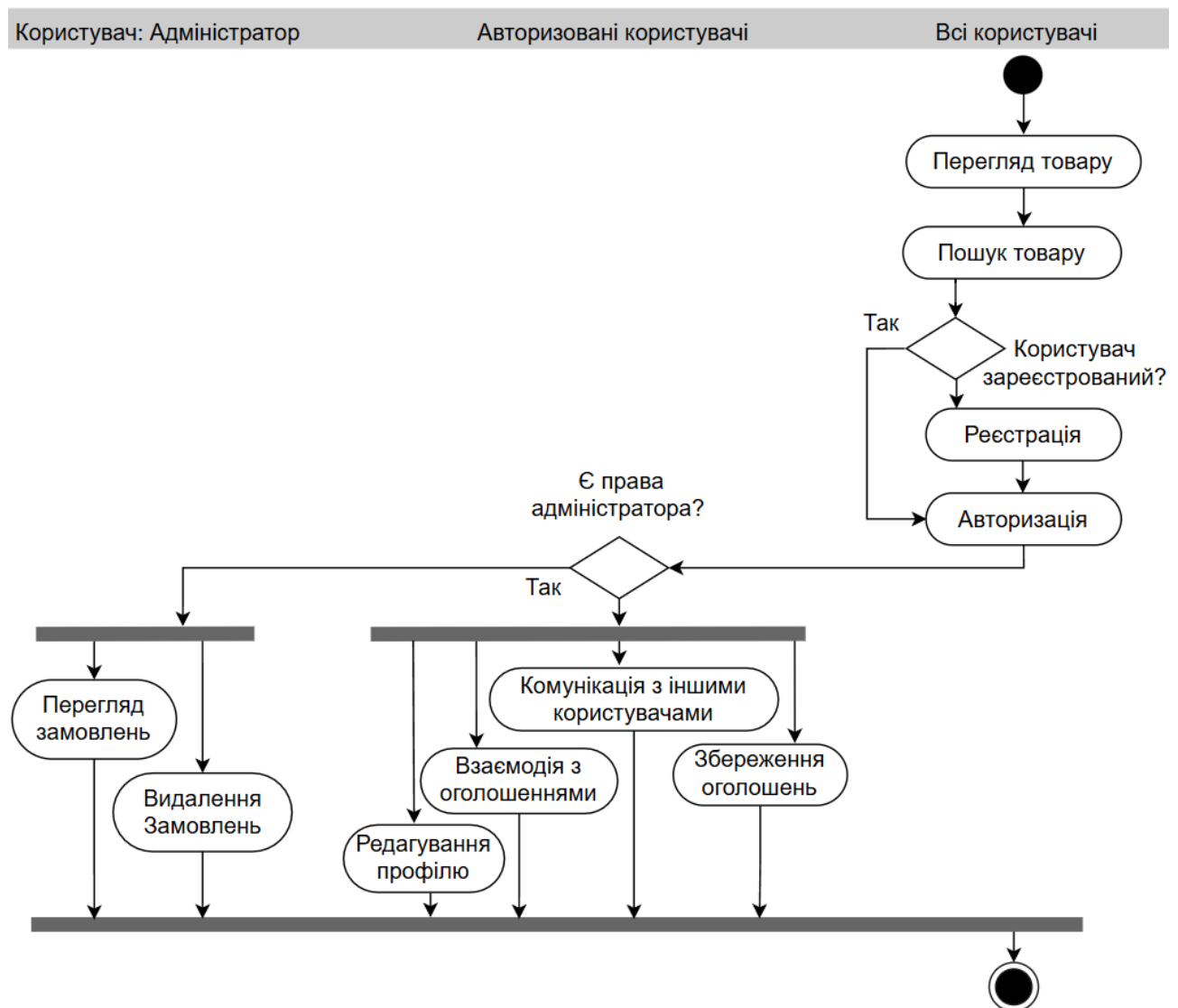


Рис. 2.3. Діаграма активності

Якщо користувач не авторизувався:

- неавторизований користувач після потрапляння на головну сторінку має можливість переглядати всі оголошення, які були раніше розміщені на платформі оголошень;
- неавторизований користувач має можливість користуватися пошуком і фільтрацією для подальшого пошуку і переглядання конкретних оголошень;
- неавторизований користувач має можливість зареєструватися на платформі;
- неавторизований користувач має можливість авторизуватися на платформі.

Якщо користувач авторизувався:

- авторизований користувач має можливість створювати нові оголошення, редагувати та видаляти власні оголошення, а також замовляти оголошення;
- авторизований користувач має можливість оновлювати інформацію свого особистого запису, та якщо у нього виникне потреба видалити його з платформи;
- авторизований користувач має можливість додавати різні оголошення в список збережених оголошень;
- авторизований користувач має можливість комунікувати з іншими користувачами платформи оголошень для уточнення умов конкретного оголошення.

Якщо авторизувався адміністратор:

- авторизований адміністратор має можливість переглядати всі поточні замовлень користувачів;
- авторизований адміністратор має можливість видаляти замовлення конкретного користувача.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						36
Змн.	Арк.	№ докум.	Підпис	Дата		

2.4. Реалізація функціоналу платформи з оголошень з продажу

товарів

Розглянемо основний функціонал платформи оголошень для продажу товару. Першочергово кожен користувач, при потраплянні на платформу, може спостерігати за головною сторінкою сайту на якій відображаються всі доступні на даний момент оголошення в порядку від нового до старого, а також є поля форми для фільтрації цих оголошень. Функція яка відображає головну сторінку, а також відповідає за фільтрування оголошень на головній сторінці зображена на рисунку 2.4.

```
public function getAdvertOnPage(Request $request): mixed
{
    $queryParams = $request->all();

    $Data = $queryParams['Data'] ?? 'DESC';
    $PriceMin = $queryParams['PriceMin'] ?? '';
    $PriceMax = $queryParams['PriceMax'] ?? '';
    $Category = $queryParams['Category'] ?? '';
    $City = $queryParams['City'] ?? '';
    $Text = $queryParams['Text'] ?? '';

    $Advert = $this->AdvertRepository->getAdvert(
        direction: $Data,
        PriceMin: $PriceMin,
        PriceMax: $PriceMax,
        Category: $Category,
        City: $City,
        Text: $Text
    );

    return view(view: 'HomePage', data: [
        "data" => $Advert,
        'oldParam' => $queryParams,
    ]);
}
```

Рис. 2.4. Відображення головної сторінки з фільтрацією

Також користувач має можливість натиснути на будь-яке потрібне йому оголошення, що автоматично перемістить його на сторінку цього оголошення. У випадку якщо оголошення з даним ідентифікатором не існує відобразиться відповідне повідомлення. Функція що відповідає за сторінку конкретного оголошення зображена на рисунку 2.5.

```

public function getAdvertItem(string $id): mixed
{
    $Advert = Advert::find(id: $id);

    if (!$Advert) {
        abort(code: 404, message: "Оголошення з цим індифікатором ($id) не знайдено");
    }

    $Users = Users::find(id: $Advert->User_id);
    $chek = false;

    if (Auth::check() && Auth::user()->Subscribes != null) {
        $subscribes = json_decode(json: Auth::user()->Subscribes);
        if (array_search(needle: $Advert->id, haystack: $subscribes) !== false)
            $chek = true;
    }

    return view(view: 'advertPage', data: [
        "data" => $Advert,
        "Users" => $Users,
        "chek" => $chek
    ]);
}

```

Рис. 2.5. Відображення конкретного оголошення

Далі всі відвідувачі платформи мають можливість авторизації і реєстрації нового аккаунту. Функція реєстрації нового аккаунту на рисунку 2.6.

```

public function createUser(Request $request): mixed|RedirectResponse
{
    $requestData = $request->validate(rules: [
        'FirstName' => 'required|max:255',
        'LastName' => 'required|max:255',
        'email' => 'required|email|unique:users|max:255',
        'password' => 'required|min:3|max:255|confirmed:password-confirm',
    ], params: [
        'email.unique' => 'Користувач з такою поштою вже існує'
    ]);

    $Users = new Users();

    $user = $Users->create(attributes: [
        'FirstName' => $requestData['FirstName'],
        'LastName' => $requestData['LastName'],
        'email' => $requestData['email'],
        'password' => Hash::make(value: $requestData['password'])
    ]);

    Auth::login(user: $user);

    return redirect()->route(route: 'homePage');
}

```

Рис. 2.6. Реєстрація аккаунту

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		38

Після того як поточний користувач створив свій власний обліковий запис, він може авторизуватися на платформі за допомогою функції яка зображена на рисунку 2.7.

```
public function login(Request $request)
{
    $credentials = $request->validate(rules: [
        'email' => 'required|max:255',
        'password' => 'required|max:255',
    ]);

    if (!Auth::attempt(credentials: $credentials)) {
        return view(view: 'auth.login', data: [
            'error' => 'Неправильна пошта або пароль'
        ]);
    }
    $request->session()->regenerate();

    return redirect()->route(route: 'homePage');
}
```

Рис. 2.7. Авторизація аккаунту

У авторизованого користувача з'являється можливість вийти з свого облікового запису в будь-який час з будь-якого місця на платформі оголошень, після чого його переадресує на головну сторінку використовуючи наступний функціонал (рис 2.8).

```
public function logout(Request $request)
{
    Auth::logout();

    $request->session()->invalidate();
    $request->session()->regenerateToken();

    return redirect()->route(route: 'homePage');
}
```

Рис. 2.8. Вихід з системи

Також після авторизації в свій обліковий запис користувач має можливість створити власне оголошення за допомогою наступної функції зображеній на

рисунку 2.9. А також ця функція зберігає зображення для оголошення на сервері [7].

```
public function createAdvert(Request $request)
{
    $requestData = $request->validate(rules: [
        'Name' => 'required|max:255',
        'Description' => 'required',
        'Category' => 'required',
        'Price' => 'required|numeric|decimal:0,2|min:0',
        'Images' => 'required|array',
        'City' => 'required|city_list_check',
        'Images.*' => 'image'
    ], params: [
        'Images.*.image' => 'Це поле повинно містити зображення',
    ]);

    $arr = [];

    foreach ($request['Images'] as $image) {
        $ImagesPath = $image->store('uploads', 'public');
        $manager = new ImageManager(driver: Driver::class);
        $manager->read(input: "storage/{$ImagesPath}")->scaleDown(width: 800, height: 800)->save();
        array_push(array: &$arr, values: $ImagesPath);
    }

    $Advert = new Advert();

    $Advert->insertGetId(values: [
        'Name' => $requestData['Name'],
        'Description' => nl2br(string: $requestData['Description']),
        'Price' => $requestData['Price'],
        'Images' => json_encode(value: $arr),
        'Category' => $requestData['Category'],
        'City' => $requestData['City'],
        'User_id' => Auth::user()->id,
        'created_at' => date(format: 'Y.m.d h:i:s')
    ]);

    return redirect()->route(route: 'homePage');
}
```

Рис. 2.9. Створення власного оголошення

Користувач може перейти на сторінку з власними оголошеннями для подальшого їх редагування (рис 2.10).

```
public function updateAdvertPage(string $id)
{
    $Advert = Auth::user()->advert()->find($id);

    if (empty($Advert))
        return redirect()->route(route: 'homePage');

    return view(view: 'pages.updateAdvert', data: ["data" => $Advert]);
}
```

Рис. 2.10. Сторінка власних оголошень

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		40

На сторінці з власними оголошеннями авторизований користувач може редагувати свої раніше створені оголошення. Після того як користувач змінить оголошення всі старі зображення будуть видалені з серверу і на їх місці будуть збережені нові (рис 2.11) [7].

```
public function updateAdvert(Request $request, string $id)
{
    $requestData = $request->validate(rules: [
        'Name' => 'required|max:255',
        'Description' => 'required',
        'Category' => 'required',
        'Price' => 'required|numeric|decimal:0,2|min:0',
        'Images' => 'required|array',
        'City' => 'required|city_list_check',
        'Images.*' => 'image'
    ], params: [
        'Images.*.image' => 'The Images field must be an image.',
    ]);

    $Advert = Advert::find(id: $id);
    foreach ($Advert->Images as $image) {
        unlink(filename: 'storage/' . $image);
    }

    $arr = [];

    foreach ($request['Images'] as $image) {
        $ImagesPath = $image->store('uploads', 'public');
        $manager = new ImageManager(driver: Driver::class);
        $manager->read(input: "storage/{$ImagesPath}")
            ->scaleDown(width: 800, height: 800)->save();
        array_push(array: &$arr, values: $ImagesPath);
    }

    $Advert->update(attributes: [ ...
    ]);

    return redirect()->route(route: 'myAdvertPage');
}
```

Рис. 2.11. Редагування оголошення

Також на сторінці з власними оголошеннями користувач має можливість видаляти свої власні оголошення, що також автоматично видалить всі зображення які ці оголошення містять (рис 2.12) [7].

```

public function deleteAdvert(string $id): mixed
{
    $Advert = Advert::find(id: $id);

    if (!$Advert) {
        abort(code: 404, message: "оголошення з цим індифікатором ($id) не знайдено");
    }

    foreach ($Advert->Images as $image) {
        unlink(filename: "storage/{$image}");
    }

    $Advert->delete();

    $Advert = Advert::where(column: 'User_id', operator: Auth::user()->id)->get();

    if (empty($Advert))
        return view(view: 'pages.myAdvert', data: ["data" => false]);

    $Advert = $Advert->toQuery()
        ->orderBy(column: 'advert.created_at', direction: 'DESC')->paginate(perPage: 4);

    return view(view: 'pages.myAdvert', data: ["data" => $Advert]);
}

```

Рис. 2.12. Видалення оголошення

Користувач має можливість перейти на сторінку свого профілю для подальшого редагування деяких його параметрів. Він може змінити свою пошту, номер телефону, ім'я, фамілію, пароль та зображення профілю (рис 2.13) [7].

```

public function changeProfilePicture(Request $request)
{
    $requestData = $request->validate(rules: [
        'profilePicture' => 'required|image'
    ]);

    unlink(filename: 'storage/' . Auth::user()->ProfilePicture);

    $ImagesPath = $requestData['profilePicture']->store('profilePicture', 'public');
    $manager = new ImageManager(driver: Driver::class);
    $manager->read(input: "storage/{$ImagesPath}")
        ->cover(width: 600, height: 600)->save();

    $Users = Auth::user();

    $Users->update(attributes: [
        'ProfilePicture' => $ImagesPath
    ]);

    return redirect()->route(route: 'profilePage');
}

```

Рис. 2.13. Редагування особистого профілю

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		42

Функціонал редагування інших параметрів подібний між собою, тому ми його розглядати не будемо.

Останній функціонал розміщений на сторінці профілю це можливість видалити свій особистий запис. Під час його видалення також видаляються всі зображення з серверу, які зв'язані з цим особистим записом, а також видаляються всі записи в базі даних які пов'язані з поточним користувачем. Потім користувач виходить з системи, далі очищуються ключі поточної сесії, регенерується CSRF-токен і в кінці відбувається переадресація на головну сторінку (рис 2.14) [7].

```

public function deleteUser(Request $request): mixed
{
    $user = Auth::user();
    $Adverts = Advert::where(column: 'User_id', operator: Auth::user()->id)->get();

    if ($user->ProfilePicture != null)
        unlink(filename: 'storage/' . $user->ProfilePicture);

    foreach ($Adverts as $Advert) {
        foreach ($Advert->Images as $image) {
            unlink(filename: 'storage/' . $image);
        }
    }

    $user->advert()->delete();
    $user->chatGet()->delete();
    $user->chatGive()->delete();
    $user->comments()->delete();
    $user->order()->delete();
    $user->delete();

    Auth::logout();

    $request->session()->invalidate();
    $request->session()->regenerateToken();

    return redirect()->route(route: 'homePage');
}

```

Рис. 2.14. Редагування особистого профілю

Останнє що ми розглянемо це буде функціональність відправлення і отримання повідомлень.

Для отримання повідомлення використовується функція зображена на рисунку 2.15 [11, 26].

```

public function broadcast(Request $request): Factory|View
{
    $currentChat = Chat::find(id: $request['id']);

    if (empty($currentChat)) {
        $Chat = new Chat();

        $array = [ ...
            ],
        ];
        $Chat->insertGetId(values: [ ...
            ]);
    } else {
        $mainArray = $currentChat->Text;
        $arrayAdd = [ ...
            ];

        array_push(array: &$mainArray, values: $arrayAdd);

        $currentChat->Text = $mainArray;
        $currentChat->save();
    }

    broadcast(event: new PusherBroadcast(
        message: $request['message'],
        id: $request['id'],
        idChat: $request['idChat']
    ))->toOthers();

    return view(view: 'components.chat-message-broadcast', data: [
        'message' => $request['message'],
        'data' => date(format: 'Y.m.d H:i:s')
    ]);
}

```

Рис. 2.15. Відправлення повідомлення

Ця функція оновлює поточний чат, а якщо його немає то створює новий. Далі ми починаємо транслювати поточне відправлене повідомлення з ідентифікатором поточного чату і в випадку коли у іншого користувача відкритий цей чат у нього виводиться відповідне повідомлення в реальному часі. Також ми виводимо у поточного користувача повідомлення в чаті.

Функція яка відповідає за отримання повідомлення зображена на рисунку 2.16 [26].

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						44
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public function receive(Request $request): Factory|View
{
    $chat = Chat::find(id: $request['id']);

    $user = (Auth::user()->id == $chat->User_sell_id)
        ? Users::find(id: $chat->User_buy_id)
        : Users::find(id: $chat->User_sell_id);

    return view(view: 'components.chat-message-receive', data: [
        'message' => $request['message'],
        'user' => $user,
        'data' => date(format: 'Y.m.d H:i:s')
    ]);
}

```

Рис. 2.16. Відправлення повідомлення

Ця функція отримує поточний чат, визначає якому користувачу відправлено повідомлення, для його подальшого відображенні в чаті, і повертає елемент який буде відображатися в чаті.

Підсумовуючи, було створено діаграму компонентів краудфандингової платформи (рис 2.17) і визначено основні компоненти збору [15].

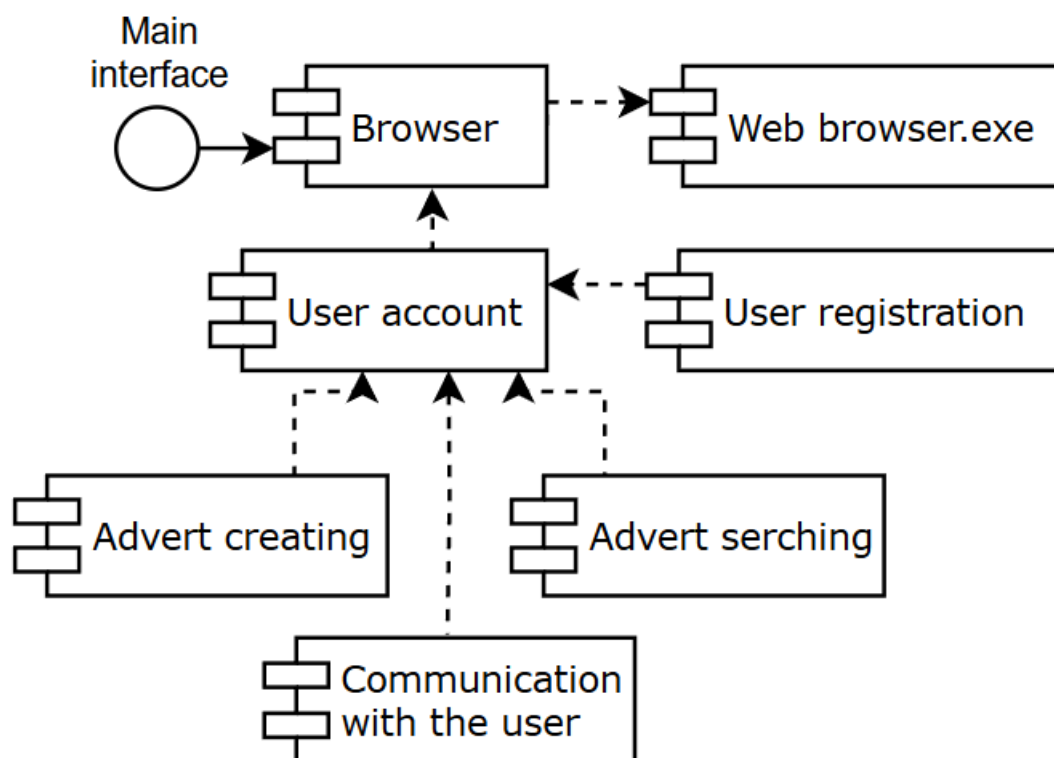


Рис. 2.17. Діаграма компонентів

На діаграмі визначені такі елементи:

- User account – компонент користувача;
- Advert serching – компонент за допомогою якого користувачі можуть шукати відповідні оголошення;
- User registration – компонент за допомогою якого користувачі можуть зареєструватися на платформі;
- Communication with the user – компонент за допомогою якого користувачі можуть комунікувати один між одним;
- Advert creating – компонент за допомогою якого користувачі можуть створювати власні оголошення.

Висновки до другого розділу

В процесі проектування платформи оголошень були проаналізовані усі варіанти використання програмного продукту відповідно до вимог програмного забезпечення. На основі отриманого результату була спроектована структура платформи. Проектування включало в себе моделювання структури проекту та взаємодії між їх складовими частинами.

Також було спроектовано базу даних для зберігання інформації про оголошення, та для даних які будуть зберігатися на серверній частині додатку, таких як користувачі та категорії оголошень.

Було описано алгоритм роботи з платформою та описані послідовні пункти варіантів його використання.

Також був наведений опис основного функціоналу платформи – це створення нового оголошення користувачем та його редагування, також були розглянуті основні функції по взаємодії користувачів із оголошеннями.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						46
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ОНЛАЙН ПЛАТФОРМОЮ ОГОЛОШЕНЬ З ПРОДАЖУ ТОВАРІВ

3.1. Порядок встановлення та налаштування параметрів платформи

Встановлення та налаштування платформи оголошень вимагає розгортання всіх його компонентів та налаштування як загальної системи в цілому, так і кожного елементу окремо.

Для подальшого опису архітектури платформи оголошень було створено діаграму розгортання (рис. 3.1) [15].

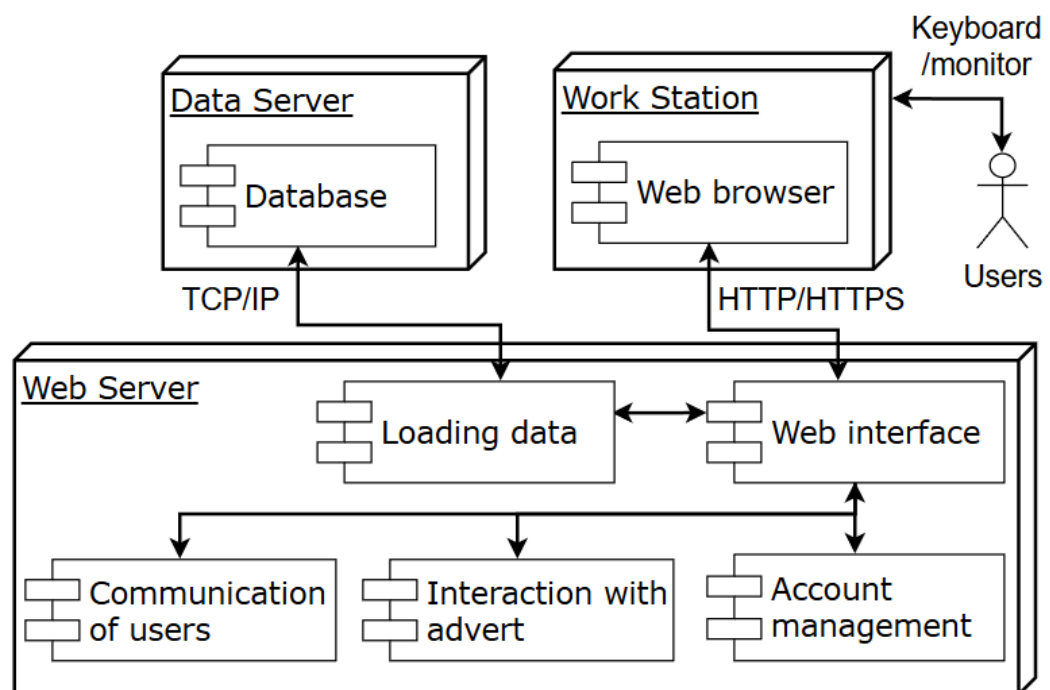


Рис. 3.1. Діаграма розгортання платформи оголошень

Опис процес розгортання.

Work Station – робоча станція користувача, через яку він користується своїм браузером.

Web browser – браузер користувача за допомогою якого він користується платформою оголошень.

Data Server – сервер на якому буде розміщуватися база даних.

Database – база даних, яка зберігає всю інформацію для подальшого використання на платформі оголошень.

Web Server – сервер на якому будуть розміщений функціонал платформи.

Web interface – набір методів який відповідає за візуалізацію інтерфейсів і взаємодію з ними.

Loading data – методи які підвантажують інформацію з бази даних і оновлюють її.

Account management – методи які відповідають за використання облікового запису користувача.

Interaction with advert – методи які відповідають за взаємодію з оголошеннями.

Communication of users – методи які відповідають за комунікацію користувачів між собою.

3.2. Структура функціонального інтерфейсу платформи оголошень з продажу товарів

При переході на платформу користувач бачить головну сторінку (рис. 3.2).

Оголошення UA | Створити оголошення | Авторизація | Реєстрація

Категорія: Місто: Ціна: від: до: Сортування по даті: ☒ Спочатку старі

Назва оголошення:

Кросівки
Ціна: 1550 грн.
2025-06-13
Коростень

Одяг

Штани
Ціна: 750 грн.
2025-06-12
Черняхів

Одяг

лінійка
Ціна: 40 грн.
2025-06-12
Черняхів

Канцелярія

Чашка
Ціна: 200 грн.
2025-06-12
Топорище

Посуд

Назад 1 2 Вперед

Головна сторінка | Часті запитання | Про нас

© 2025 Оголошення UA, Inc.

Рис. 3.2. Головна сторінка

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		48

На головній сторінці ми можемо спостерігати за шапкою сайту, на якій знаходиться логотип і назва сайту, кнопка яка переносить нас до сторінки на якій можна створити власне оголошення, а також кнопки які переносять нас на сторінки з авторизацією і реєстрацією. Далі на головній сторінці розміщений форма для подальшого фільтрування оголошень. В формі присутні поля фільтрування за категорією, містом, максимальною і мінімальною ціною, сортування по даті, пошук по назві оголошення та кнопка пошуку. Далі розміщений блок з самими оголошеннями та пагінація. У кожному оголошенню є перше зображення з списку зображень, назва оголошення, його ціна, дата, категорія і місто в якому це оголошення розміщено. В кінці сторінки розміщений футер.

При натиснені на оголошення нас переносить на сторінку цього оголошення (рис. 3.3)

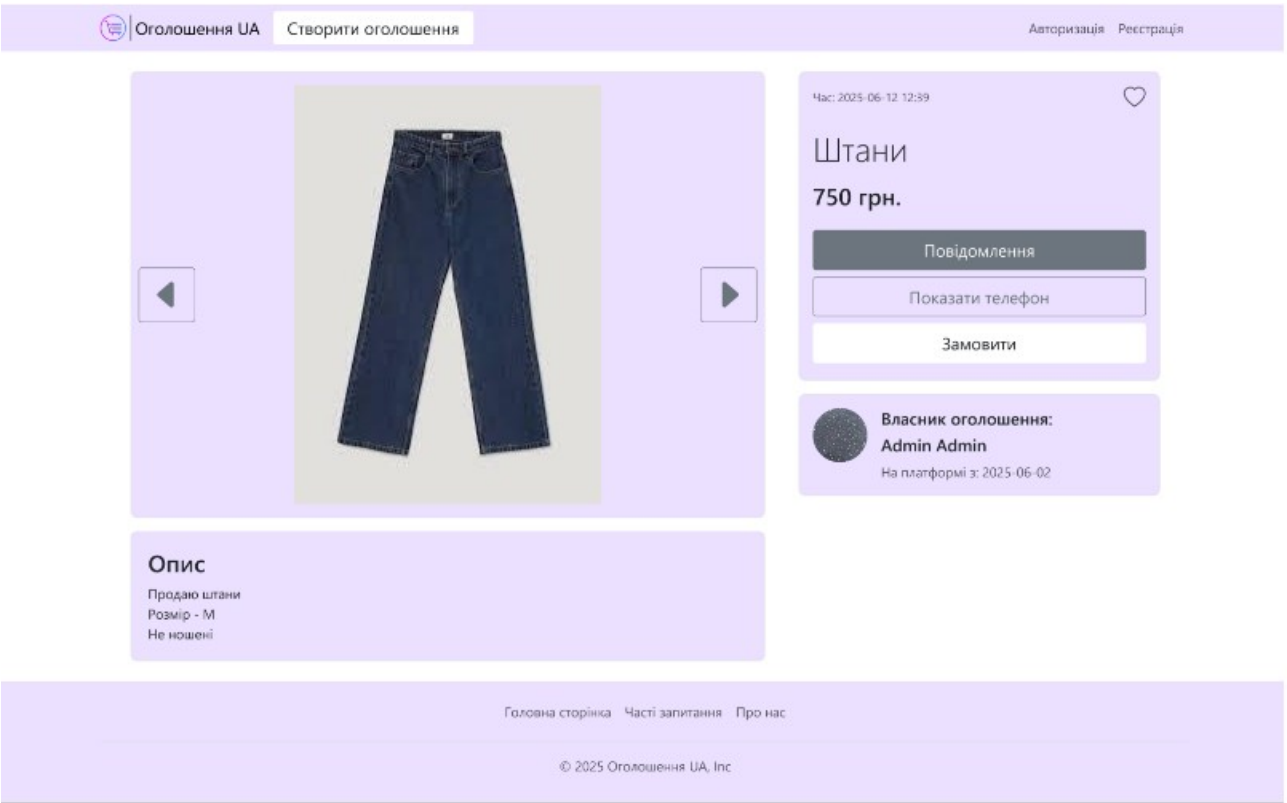


Рис. 3.3. Сторінка оголошення

На сторінці оголошення розмішена карусель з зображеннями товару цього оголошення, під зображеннями знаходиться блок з описом оголошення, справа від каруселі елемент в якому вказано час створення оголошення, можливість

додати оголошення до списку збережених, його назва та ціна, а також кнопки для відображення номеру власника оголошення, можливість замовити це оголошення і кнопка яка переносить користувача в чат з володарем оголошення. Під кнопками розміщений блок з короткою інформацією про власника оголошення, зображення його особистого запису, ім'я, фамілія та дата коли був зареєстрований його особистий запис.

Щоб користувач міг взаємодіяти з веб-додатком йому потрібно зареєструватися або авторизуватися на платформі оголошень (рис. 3.4).

Рис. 3.4. Вікно реєстрації

На сторінці реєстрації знаходиться форма яку потрібно заповнити відповідними даними.

У випадку якщо дані в формі введено не вірно відобразяться відповідні помилки (рис. 3.5).

Рис. 3.5. Помилки в вікні реєстрації

Якщо всі поля заповнені правильно, новий обліковий запис буде створене і користувача авторизує в ньому.

Після авторизації у користувача з'являється можливість створювати власне оголошення (рис. 3.6).

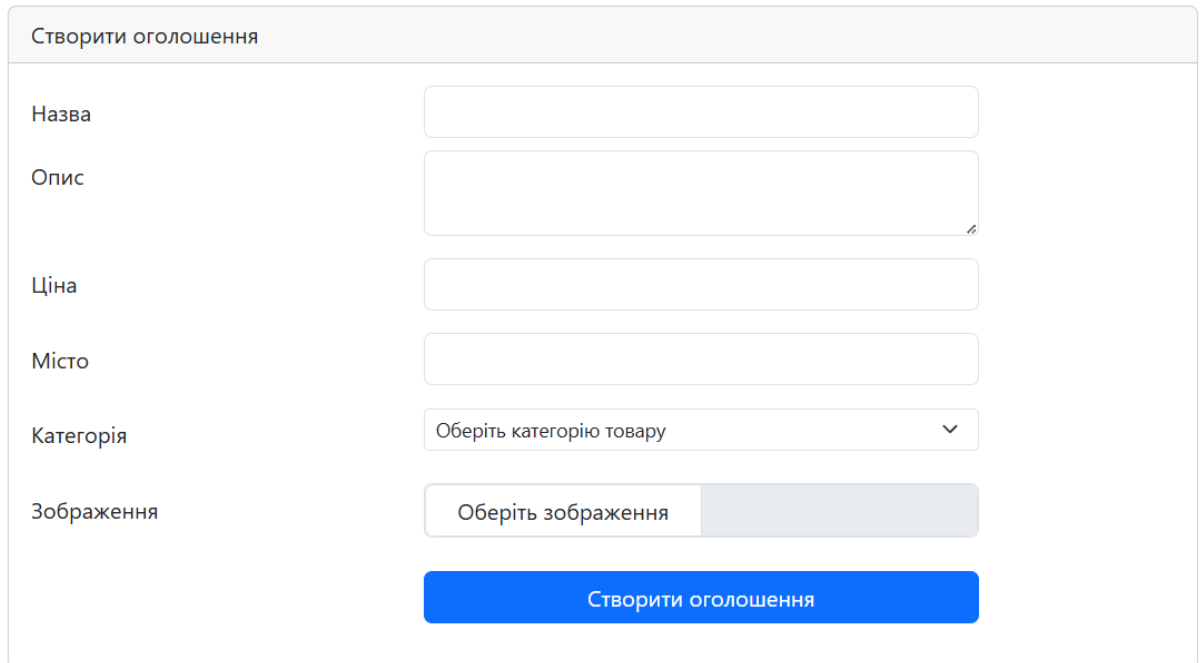


Рис. 3.6. Сторінка для створення власного оголошення

Після заповнення і надсилання форми, у випадку якщо всі данні введено правильно, створюється нове оголошення.

Після успішної авторизації на шапці сторінки починає відображатися елемент з поточним користувачем (рис. 3.7).

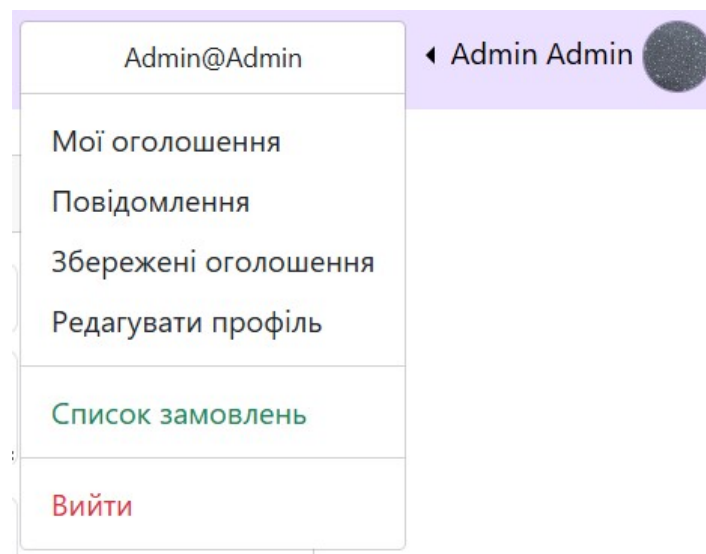


Рис. 3.7. Випадаюче меню користувача

При натисненні на елемент відображеного поточного користувача з'явиться випадаюче меню з різними можливостями. В меню відображається електронна пошта користувача, посилання на сторінку з його власними оголошеннями, посилання на чати з іншими користувачами, посилання на збережені оголошення, посилання на сторінку з редагуванням особистого запису, у випадку якщо у користувача є права адміністратора відобразиться посилання на список всіх замовлень, а також кнопка для того щоб вийти з свого особистого запису.

Першочергово розглянемо сторінку власних оголошень авторизованого користувача (рис. 3.8).

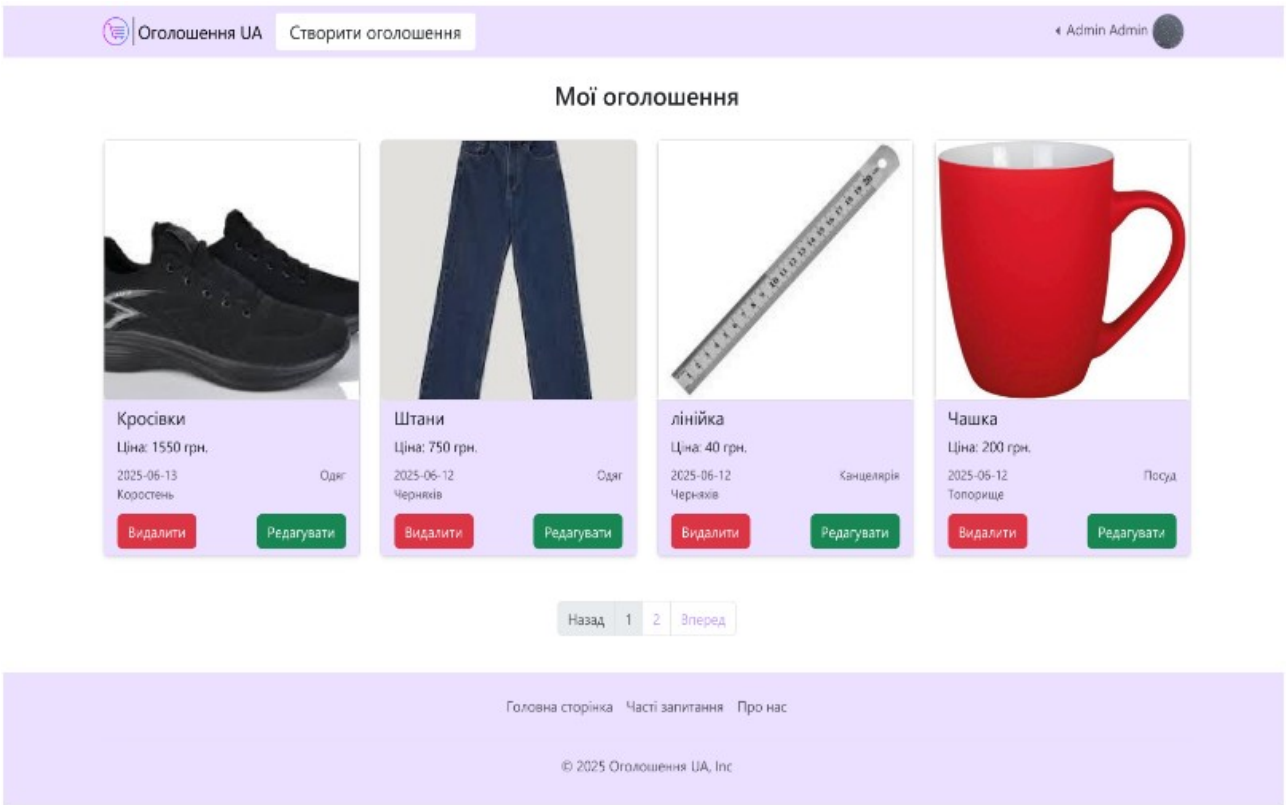


Рис. 3.8. Сторінка власних оголошень

На цій сторінці відображаються всі оголошення поточного користувача, а також кнопки з можливостями редагувати конкретне оголошення та видалити їх.

Далі ми будемо переглянемо сторінку з усіма чатами різних користувачів, які спілкувалися з поточним користувачем (рис. 3.9) [11].

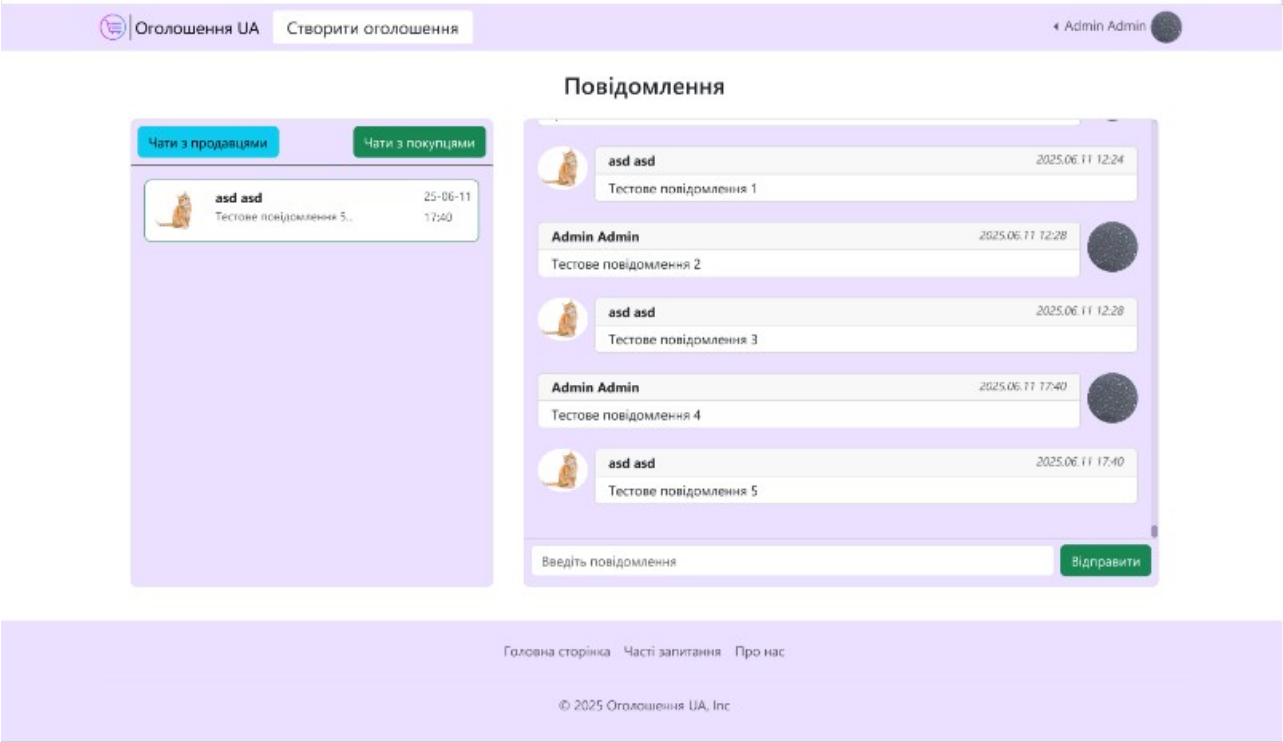


Рис. 3.9. Сторінка з чатами

На сторінці зліва зображено блок з кнопками для відображення чатів продавців та покупців, а справа відображається поточний відкритий чат з усіма його повідомленнями, а також відображається елемент в який потрібно ввести текст і кнопка для його відправлення.

Тепер розглянемо сторінку з збереженими оголошеннями (рис. 3.10).

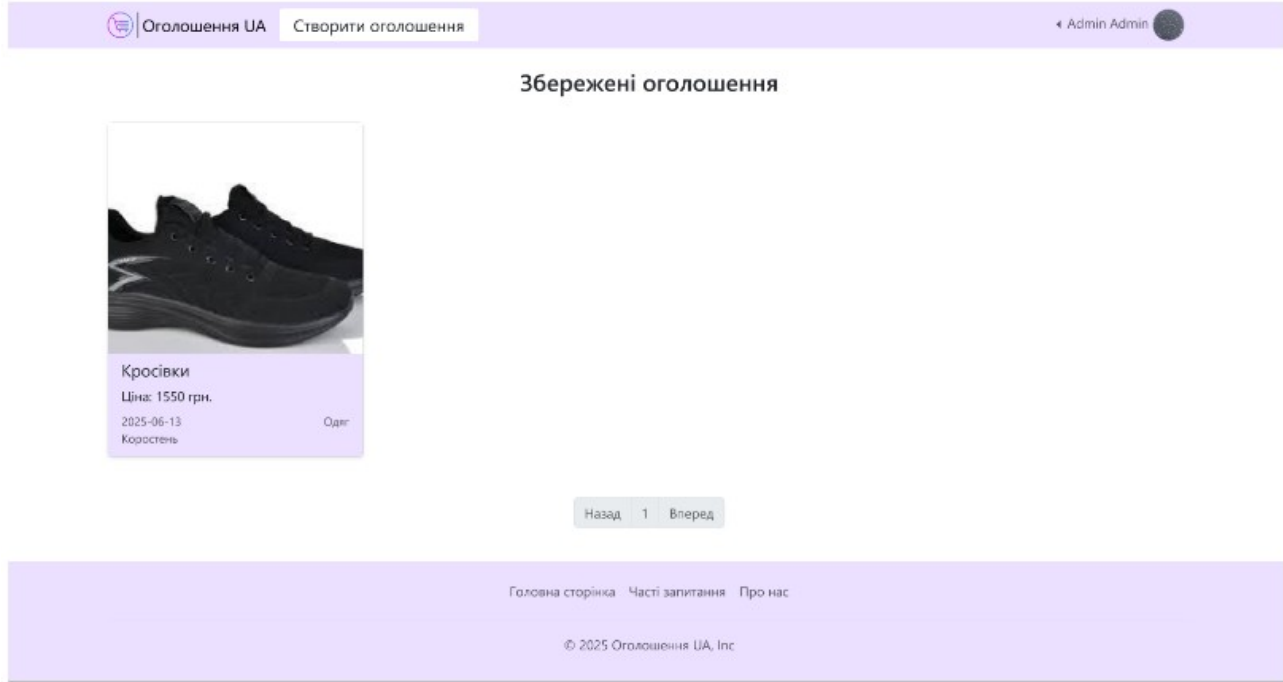


Рис. 3.10. Сторінка з збереженими оголошеннями

На сторінці відображаються всі оголошення які користувача зберіг раніше.

Зараз розглянемо сторінку з налаштуваннями поточного облікового запису користувача (рис. 3.11).

Рис. 3.11. Сторінка з налаштуваннями

На даній сторінці відображаються шість блоків при на тисненні на які з'являється відповідна форма для зміни певних даних особистого запису поточного користувача. Тут є такі блоки як зміна зображення особистого запису, зміна імені або фамілії, зміна пошти, паролю та номеру телефону. Останній шостий блок відповідає за видалення особистого запису користувача та даних, які пов'язані з цим записом.

Остання сторінка яку ми розглянемо, це сторінка зі всіма замовленнями користувачів (рис. 3.12).

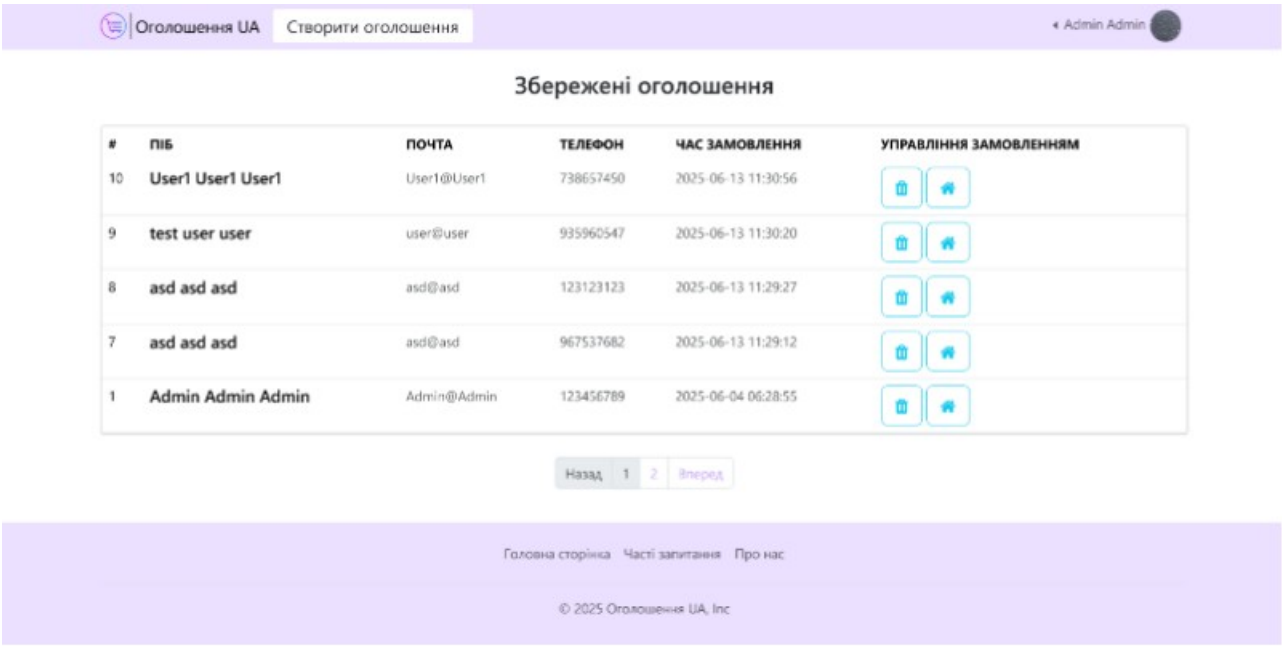


Рис. 3.12. Сторінка з замовленнями

На цій сторінці відображається інформація про замовлення користувачів. В цій таблиці знаходиться інформація про ідентифікатор замовлення, ім'я, фамілію та ім'я по батькові, пошту, номер телефону, дату коли це замовлення було зроблено і кнопки для видалення замовлення та переходу до замовленого оголошення. А також до цієї сторінки мають доступ лише користувачі з правами адміністратора.

Після переходу на сторінку оголошення, користувач може замовити його, що відобразить сторінку з відповідною формою для замовлення оголошення (рис. 3.12).

Створити замовлення

Ім'я

Admin

Фамілія

Admin

Ім'я по батькові

Електронна пошта

Admin@Admin

Номер телефону

+380

Створити замовлення

Рис. 3.13. Сторінка для створення замовлення

Після заповнення і надсилання форми, у випадку якщо всі данні введено правильно, створюється нове замовлення, а також до цієї сторінки автоматично підтягуються дані користувача.

3.3. Тестування роботи програмного комплексу

За допомогою інструменту lighthouse було проведено тестування платформи оголошень.

Lighthouse — це безкоштовний інструмент від Google, який використовується для аналізу якості різних веб-сайтів. Він допомагає виявити проблемні місця та недоліки веб-застосунку, оптимізувати швидкість завантаження сторінок, покращити доступність і зробити сайт більш зручним і зрозумілим для користувачів [12].

Після завершення тестування було отримано звіт, який містить різноманітну інформацію про продуктивність сайту та різні поради як її покращити (рис. 3.20).

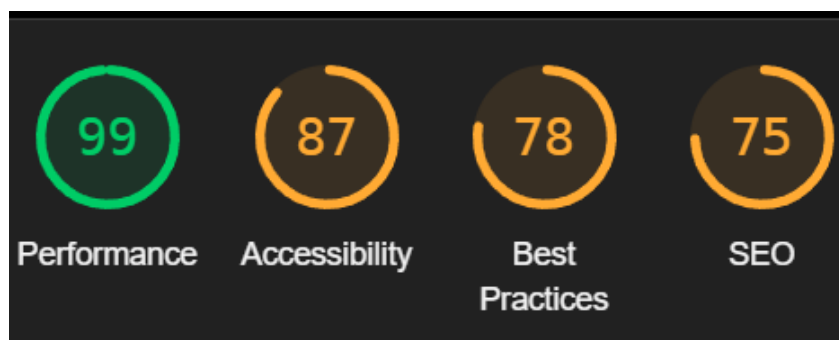


Рис. 3.20. Тестування веб-додатку за допомогою lighthouse

Звіти lighthouse містять важливі дані, такі як продуктивність даної сторінки, її доступність, досвід взаємодії користувача з сторінкою, пошукову оптимізацію, швидкість завантаження різних елементів та поради для покращення кожного з вище названих пунктів. Ця інформація може допомогти вам зрозуміти, як ваш веб-сайт працює з точки зору продуктивності та швидкості, і дозволяє вам вжити необхідних заходів для покращення цих показників (рис. 3.21).

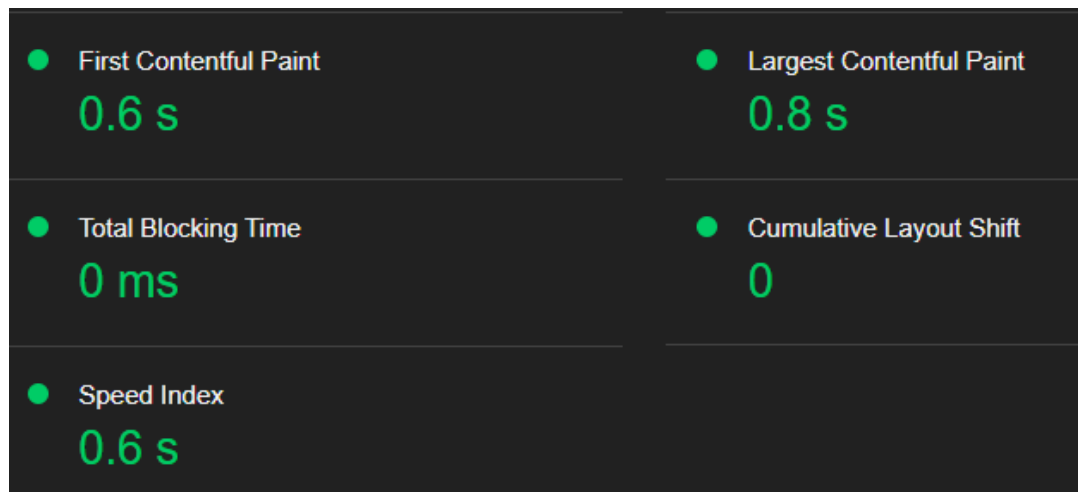


Рис. 3.21. Тестування швидкості завантаження елементів платформи за допомогою lighthouse

Першочергово розглянемо поради стосовно продуктивності сторінки (рис. 3.22).

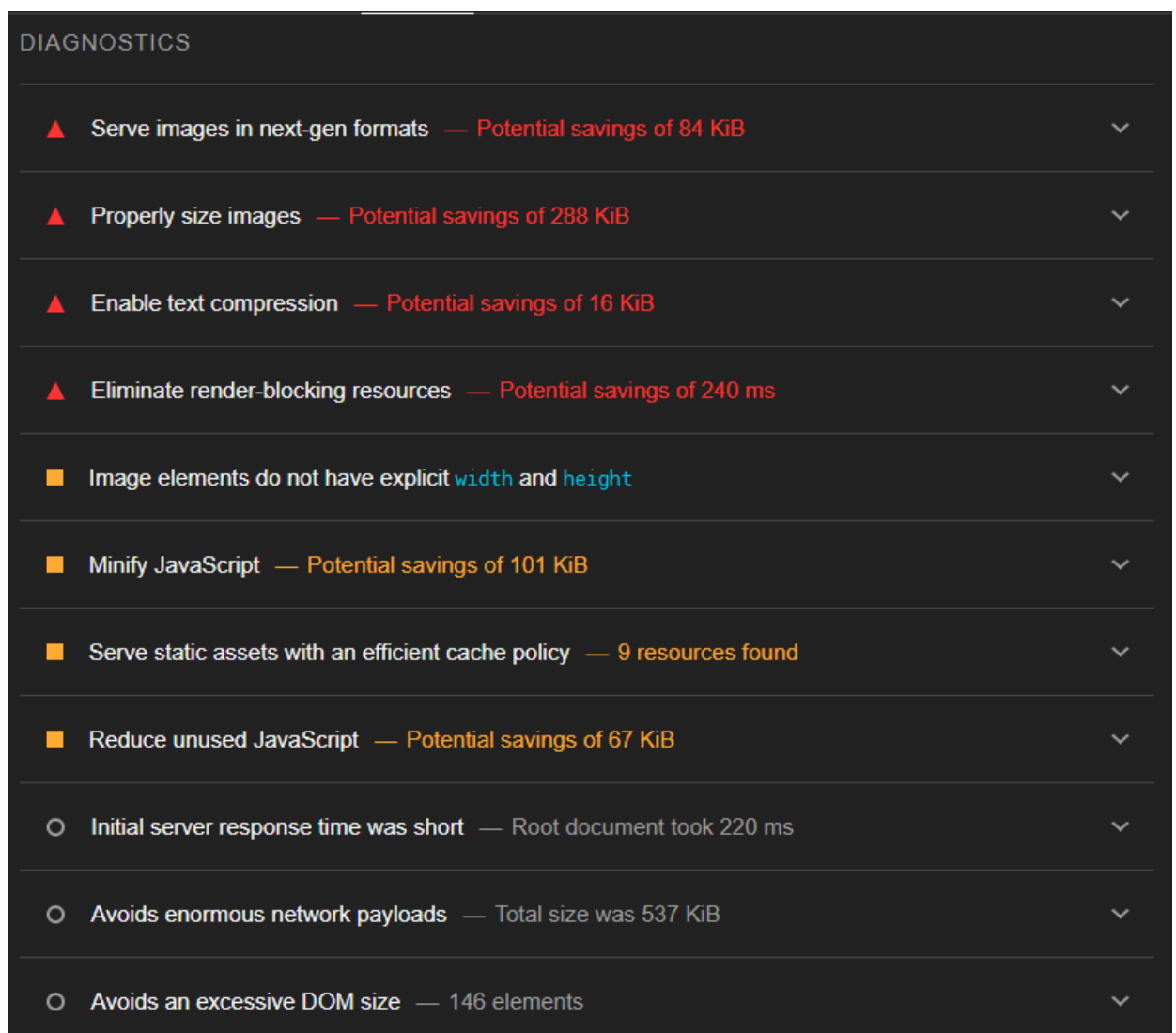


Рис. 3.22. Поради для покращення продуктивності

Для покращення продуктивності lighthouse рекомендує змінити формат використаних зображень на WebP або AVIF для зменшення їх розміру, відображати зображення відповідно до їх розмірів, щоб заощадити стільниковий трафік і пришвидшити завантаження, обробляти зі стисненням текстові ресурси, щоб мінімізувати загальний обсяг мережевих байтів, оптимізувати додавання JS та CSS файлів для прискорення завантаження сторінки та надає інші рекомендації для сторінки.

Далі перейдемо до переглядання порад пов'язані з доступністю даної веб-сторінки (рис. 3.23).

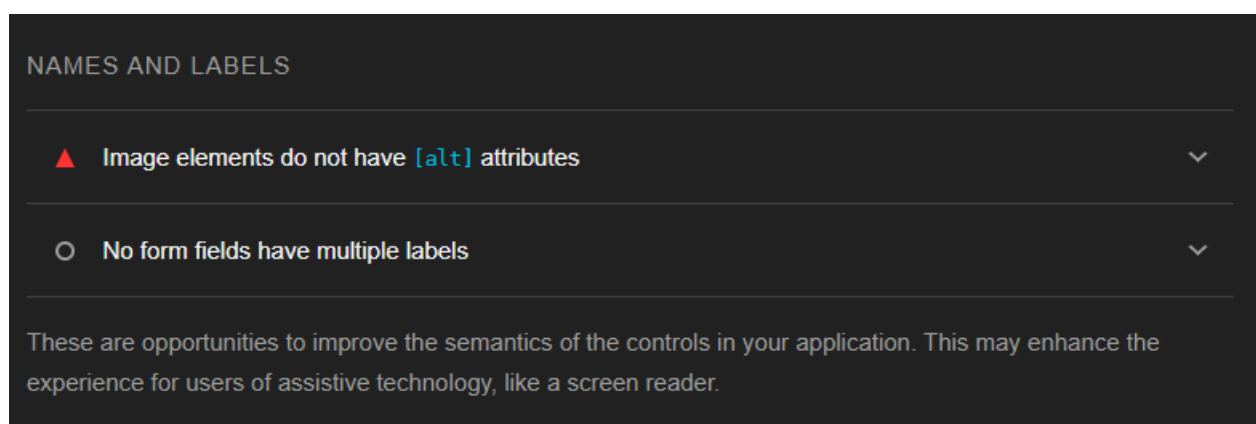


Рис. 3.23. Поради для покращення доступності

Для покращення доступності lighthouse рекомендує додати до елементів зображень атрибут alt, а також попереджує, що поля форми з кількома підписами можуть заплутати користувача який використовує допоміжні технології для зчитування даних з екрану.

Тепер переглянемо рекомендації стосовно досвіду користувацької взаємодії з сторінкою (рис. 3.24).

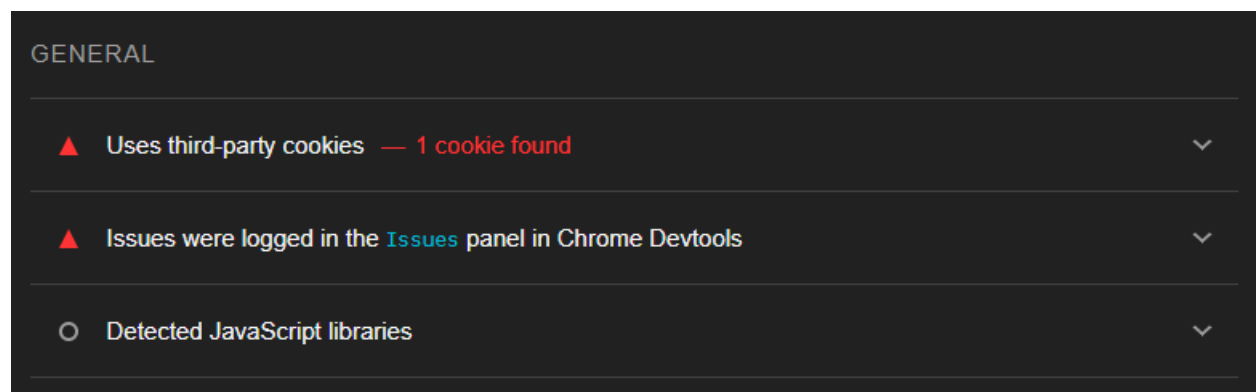


Рис. 3.24. Поради для покращення користувацької взаємодії

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		58

Для покращення користувацької взаємодії lighthouse рекомендує як краще використовувати куки файли, вирішити проблеми, які було зареєстровано за допомогою інструментів розробника Chrome та повідомляє що було знайдено JS бібліотеку яка не використовується на даній веб-сторінці.

Останні рекомендації які залишилися переглянути відносяться до покращення пошукової оптимізації (рис. 3.25).

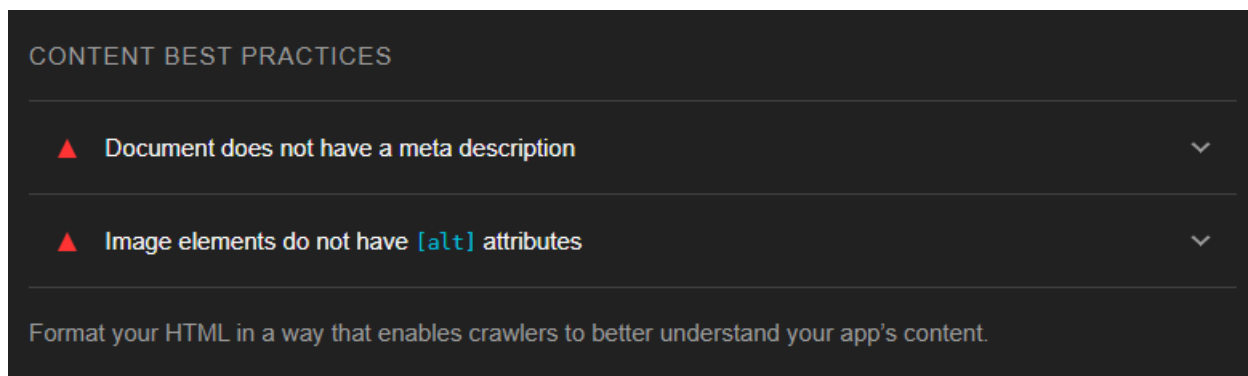


Рис. 3.25. Поради для покращення пошукової оптимізації

Для покращення пошукової оптимізації lighthouse рекомендує додати метаопис до даної веб-сторінки та додати інформативні елементи до зображень розміщених на сторінці.

За допомогою інструменту lighthouse було проведено тестування продуктивності та швидкості завантаження веб-сторінки. За результатами тестування отримано різні метрики, продуктивність сторінки, її доступність, досвід взаємодії з сторінкою, пошукову оптимізацію, Largest Contentful Paint (LCP), Cumulative Layout Shift (CLS), Total Blocking Time (TBT) та Speed Index (SI) та поради для покращення кожного з цих пунктів.

Висновки до третього розділу

В даному розділі було проаналізована структура та взаємодія компонентів головної програмної системи, побудована і розібрана архітектура програмного забезпечення та його розгортання. Кожен структурний елемент системи був описаний, а також було продемонстровано його взаємодію з іншими частинами платформи оголошень. Це дозволило зрозуміти архітектуру системи та взаємозв'язки між її компонентами.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		59

Далі було надано опис інтерфейсу розробленого програмного забезпечення, та продемонстровано всі функціональні можливості веб-додатку, залежно від ролі користувача.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		60

ВИСНОВКИ

Під час виконання кваліфікаційної роботи бакалавра було спроектовано та реалізовано платформу з оголошень, проведено аналіз предметної області та порівняно різні подібні платформи. На основі цього аналізу були визначені основні вимоги до програмного продукту. В процесі дослідження була розроблена архітектура, підходи та основні технології реалізації програмного забезпечення.

Проектування платформи оголошень включало аналіз варіантів використання програмного продукту та проектування структури, включаючи моделювання взаємодії між складовими частинами. Також було спроектовано базу даних для зберігання інформації про користувачів, оголошення, замовлень та інші дані на серверній частині додатку.

Описано алгоритм роботи з веб-додатком та послідовні пункти варіантів його використання. Розглянуто основний функціонал, включаючи створення облікового запису, створення оголошення та комунікацію користувачів між собою, а також основні функції взаємодії користувачів з оголошеннями.

Також була надана детальна схема розгортання платформи з оголошень, описано взаємодію структурних елементів системи та наведено опис інтерфейсу розробленого програмного продукту.

Додаток має інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам легко створювати власне оголошення, шукати потрібне оголошення та комунікувати з іншими володарями оголошень. Користувачі можуть редагувати свої оголошення та власні особисті записи.

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						61
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Інформаційні технології. Процеси життєвого циклу програмного забезпечення (ISO/IEC 12207:1995): ДСТУ 3918-1999. – [Чинний від 2000-01-01]. – К.: Держстандарт України, 2000. – 50 с. – (Національний стандарт України).
2. Інформаційні технології. Основні напрямки оцінювання та відбору CASE-інструментів (ISO/IEC 14102:1995) – ДСТУ 3919-1999 [Чинний від 2000-01-01]. – К.: Держстандарт України, 2000. – 470 с. – (Національний стандарт України).
3. Жильцов О.Б., Кулян В.Р., Юнькова О.О. Математичне програмування (з елементами інформаційних технологій). – Київ: ДП "Видавничий дім "Персонал", 2008. – 184 с.
4. Зайченко Ю.П. Дослідження операцій: Підручник. – Київ: Видавничий Дім "Слово", 2001. – 688 с.
5. Мартін С. М. Чистий код: створення, аналіз і рефакторинг. Фабула, 2019. 416 с.
6. В. В. Лапінський, В. Ю. Габрусєв, Н. Я. Бачинська Основи операційних систем. Посібник для студентів. Тернопіль: Навчальна книга – Богдан, 2002. 80 с.
7. PHP: PHP Manual - Manual. PHP: Hypertext Preprocessor. URL: <https://www.php.net/manual/en>
8. Ласкаво просимо до phpMyAdmin документації! – phpMyAdmin 6.0.0-dev documentation. URL: <https://docs.phpmyadmin.net/uk/latest/>
9. Стандарт UML 2.0 [Електронний ресурс:]. Режим доступу // <http://www.omg.org/spec/UML/2.0/>
10. Лавріщева К.М. Програмна інженерія / Лавріщева К.М. – Київ: КНУ ім. Тараса Шевченка, 2008. – 319 с.
11. Laracasts. [Електронний ресурс]. – Режим доступу: <https://laracasts.com/browse/all>

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		62

12. Lighthouse. [Електронний ресурс]. – Режим доступу: <https://developer.chrome.com/docs/lighthouse>
13. Офіційна документація Laravel [Електронний ресурс]. – Режим доступу: <https://laravel.com/docs/12.x>
14. Офіційна документація MariaDB [Електронний ресурс]. – Режим доступу: <https://mariadb.org>
15. Офіційна документація Visual Paradigm [Електронний ресурс]. – Режим доступу: <https://www.visual-paradigm.com>
16. Глинський, Я. М. Інтернет : Сервіси, HTML і web-дизайн: Навч. посіб. для студентів та учнів шкіл / Я. М. Глинський, В. А. Ряжська. – Львів : Деол, 2002. – 168 с.
17. Єрохін А.Л. Методи та засоби Інтернет – технологій : Навч. Посіб./ А.Л. Єрохін, В.В. Самсонов. – Харків : Компанія СМІТ, 2006 – 400 с.
18. Матвієнко О.В. Internet-технології: проектування Web-сторінки : Навч. посіб. для студ. вузів / О. В. Матвієнко, І. Л. Бородкіна. – 2-ге вид., доробл. і доп. – К. : Центр навч. літ-ри, 2004. – 154 с
19. Спірін О.М. Короткий курс інформатики: Інформаційнокомп'ютерні технології : Метод. посіб. – Житомир: Видавництво ЖДУ, 2006. – 198 с.
20. Рудненко В.Д. Практичний курс інформатики/ В.Д. Рудненко, О.М. Макарчук, М.О. Патланжоглу, за ред. Мадзігона В.М. - К.: Фенікс, 2007. - 304 с
21. CSS Tutorial. [Електронний ресурс]. – Режим доступу: www.w3schools.com/css
22. Пушкар О. І. Мультимедійні видання : навч. посібник / Пушкар О. І., Клименюк В. Є., Браткевич В. В. – Х. : Вид. ХНЕУ, 2012. – 144 с.
23. Скотт Хокінс. Адміністрування веб-сервера Apache і керівництво по електронній комерції. / С. Хокінс. — М.: «Вільямс», 2001. — 336 с.
24. W3schoolsua. [Електронний ресурс]. – Режим доступу: <https://w3schoolsua.github.io/index.html#gsc.tab=0>
25. Javascript Info. [Електронний ресурс]. – Режим доступу: <https://uk.javascript.info>

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		63

26. Laravel Broadcasting. [Електронний ресурс]. – Режим доступу:
<https://laravel.com/docs/12.x/broadcasting>

27. W3schoolsua jQuery. [Електронний ресурс]. – Режим доступу:
<https://w3schoolsua.github.io/jquery/index.html#gsc.tab=0>

28. UA PHP. [Електронний ресурс]. – Режим доступу:
<https://php.org.ua/manual/uk/index.md>

29. Пасічник О. В. Веб-дизайн. / О. В. Пасічник, В. В. Пасічник. – Львів: Магнолія, 2019. – 520 с.

30. Laravel Migrations [Електронний ресурс] – Режим доступу:
<https://laravel.com/docs/12.x/migrations>

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						64
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

AdvertController.php

```
class AdvertController extends Controller
{
    /**
     * @var AdvertRepository
     */
    private AdvertRepository $AdvertRepository;

    /**
     * @param AdvertRepository $AdvertRepository
     */
    public function __construct(AdvertRepository $AdvertRepository)
    {
        $this->AdvertRepository = $AdvertRepository;
    }

    /**
     * @return mixed
     */
    public function advertCreatePage(): mixed
    {
        return view('advertCreatePage');
    }

    /**
     * @return mixed
     */
    public function getAdvertOnPage(Request $request): mixed
    {
        $queryParams = $request->all();

        $Data = $queryParams['Data'] ?? 'DESC';
        $PriceMin = $queryParams['PriceMin'] ?? "";
        $PriceMax = $queryParams['PriceMax'] ?? "";
        $Category = $queryParams['Category'] ?? "";
        $City = $queryParams['City'] ?? "";
        $Text = $queryParams['Text'] ?? "";

        $Advert = $this->AdvertRepository->getAdvert(
            $Data,
            $PriceMin,
            $PriceMax,
            $Category,
            $City,
            $Text
        );

        return view('HomePage', [
            "data" => $Advert,
        ]);
    }
}
```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						65
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'oldParam' => $queryParams,
        'Category' => [
            "Електроніка",
            "Одяг",
            "Запчастини",
            "Меблі",
            "Канцелярія",
            "Спортивне приладдя",
            "Для тварин",
            "Посуд"
        ]
    });
}

/**
 * @return mixed
 */
public function getAllAdvert(Request $request): mixed
{
    $Advert = Advert::all()->sortBy('created_at')->reverse();

    return view('HomePage', ["data" => $Advert]);
}

/**
 * @param string $id
 * @return mixed
 */
public function getAdvertItem(string $id): mixed
{
    $Advert = Advert::find($id);

    if (!$Advert) {
        abort(404, "Оголошення з цим індифікатором ($id) не знайдено");
    }

    $Users = Users::find($Advert->User_id);
    $chek = false;

    if (Auth::check() && Auth::user()->Subscribes != null) {
        $subscribes = json_decode(Auth::user()->Subscribes);
        if (array_search($Advert->id, $subscribes) !== false)
            $chek = true;
    }

    return view('advertPage', [
        "data" => $Advert,
        "Users" => $Users,
        "chek" => $chek
    ]);
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						66
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public function AdvertItemSubscribes(Request $request, string $id)
{
    $arr = [];
    $Advert = Advert::find($id);
    $Users = Users::find(Auth::user()->id);
    $Subscribes = json_decode(Auth::user()->Subscribes);

    if ($Subscribes == [] || Auth::user()->Subscribes == null) {
        array_push($arr, $id);

        $Users->update([
            'Subscribes' => json_encode($arr)
        ]);
    } else {
        if (array_search($Advert->id, $Subscribes) !== false) {

            foreach ($Subscribes as $selem) {
                array_push($arr, $selem);
            }

            array_splice($arr, array_search($Advert->id, $Subscribes), 1);

            $Users->update([
                'Subscribes' => json_encode($arr)
            ]);
        } else {
            foreach ($Subscribes as $selem) {
                array_push($arr, $selem);
            }
            array_push($arr, $id);

            $Users->update([
                'Subscribes' => json_encode($arr)
            ]);
        }
    }
}
/**
 * @param Request $request
 * @return mixed
 * @throws Exception
 */
public function createAdvert(Request $request)
{
    $requestData = $request->validate([
        'Name' => 'required|max:255',
        'Description' => 'required',
        'Category' => 'required',
        'Price' => 'required|numeric|decimal:0,2|min:0',
    ]);
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		67

```

        'Images' => 'required|array',
        'City' => 'required|city_list_check',
        'Images.*' => 'image'
    ], [
        'Images.*.image' => 'Це поле повинно містити зображення',
    ]);

    $arr = [];

    foreach ($request['Images'] as $image) {
        $ImagePath = $image->store('uploads', 'public');
        $manager = new ImageManager(Driver::class);
        $manager->read("storage/{$ImagePath}")->scaleDown(800, 800)->save();
        array_push($arr, $ImagePath);
    }

    $Advert = new Advert();

    $Advert->insertGetId([
        'Name' => $requestData['Name'],
        'Description' => nl2br($requestData['Description']),
        'Price' => $requestData['Price'],
        'Images' => json_encode($arr),
        'Category' => $requestData['Category'],
        'City' => $requestData['City'],
        'User_id' => Auth::user()->id,
        'created_at' => date('Y.m.d h:i:s')
    ]);

    return redirect()->route('homePage');
}

/**
 * @param string $id
 * @return mixed
 */
public function deleteAdvert(string $id): mixed
{
    $Advert = Advert::find($id);

    if (!$Advert) {
        abort(404, "оголошення з цим індифікатором ($id) не знайдено");
    }

    foreach ($Advert->Images as $image) {
        unlink("storage/{$image}");
    }

    $Advert->delete();

    $Advert = Advert::where('User_id', Auth::user()->id)->get();
}

```

```

        if (empty($Advert))
            return view('pages.myAdvert', ["data" => false]);

        $Advert = $Advert->toQuery()
            ->orderBy('advert.created_at', 'DESC')->paginate(4);

        return view('pages.myAdvert', ["data" => $Advert]);
    }
}

```

AuthController.php

```

class AuthController extends Controller
{
    public function showLogin()
    {
        if (Auth::check())
            return redirect()->route('homePage');

        return view('auth.login');
    }

    public function login(Request $request)
    {
        $credentials = $request->validate([
            'email' => 'required|max:255',
            'password' => 'required|max:255',
        ]);

        if (!Auth::attempt($credentials)) {
            return view('auth.login', [
                "error" => 'Неправильна пошта або пароль'
            ]);
        }
        $request->session()->regenerate();

        return redirect()->route('homePage');
    }

    public function user()
    {
        return response()->json(Auth::user());
    }

    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();
        $request->session()->regenerateToken();

        return redirect()->route('homePage');
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						69
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    public function refresh(Request $request)
    {
        $request->session()->regenerateToken();
        $request->session()->regenerate();

        return view('HomePage');
    }
}

```

ChangeProfileController.php

```

class ChangeProfileController extends Controller
{
    public function changeProfilePicture(Request $request)
    {
        $requestData = $request->validate([
            'profilePicture' => 'required|image'
        ]);

        unlink('storage/' . Auth::user()->ProfilePicture);

        $ImagesPath = $requestData['profilePicture']->store('profilePicture', 'public');
        $manager = new ImageManager(Driver::class);
        $manager->read("storage/{$ImagesPath}")
            ->cover(600, 600)->save();

        $Users = Auth::user();

        $Users->update([
            'ProfilePicture' => $ImagesPath
        ]);

        return redirect()->route('profilePage');
    }

    public function changeFirstAndLastName(Request $request)
    {
        $requestData = $request->validate([
            'FirstName' => 'required|max:255|old_value',
            'LastName' => 'required|max:255|old_value'
        ]);

        $Users = Auth::user();

        $Users->update([
            'FirstName' => $requestData['FirstName'],
            'LastName' => $requestData['LastName']
        ]);

        return redirect()->route('profilePage');
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public function changeEmail(Request $request)
{
    $requestData = $request->validate([
        'Email' => 'required|email|max:255|old_value'
    ]);

    $Users = Auth::user();

    $Users->update([
        'email' => $requestData['Email']
    ]);

    return redirect()->route('profilePage');
}

public function changePassword(Request $request)
{
    $requestData = $request->validate([
        'password-old' => 'required|current_password',
        'password' => 'required|min:3|max:255|confirmed|old_password_check'
    ]);

    $Users = Auth::user();

    $Users->update([
        'password' => Hash::make($requestData['password'])
    ]);

    return redirect()->route('profilePage');
}

public function changePhoneNumber(Request $request)
{
    $requestData = $request->validate([
        'phoneNumber' => 'required|digits:9|numeric|decimal:0|old_value'
    ]);

    $Users = Auth::user();

    $Users->update([
        'PhoneNumber' => $requestData['phoneNumber']
    ]);

    return redirect()->route('profilePage');
}
}

```

ChatController.php

```

class ChatController extends Controller
{
    public function chatPage()
    {
        $ChatBay = Chat::where('User_bay_id', Auth::user()->id)
            ->orderBy('updated_at', 'DESC')->get();
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		71

```

$ChatSell = Chat::where('User_sell_id', Auth::user()->id)
->orderBy('updated_at', 'DESC')->get();

return view('pages.chat', ['ChatBay' => $ChatBay, 'ChatSell' => $ChatSell]);
}

public function chatSendUser(Request $request)
{
    $user = Users::find($request['id']);

    $ChatBay = Chat::where('User_bay_id', Auth::user()->id)
->orderBy('updated_at', 'DESC')->get();
    $ChatSell = Chat::where('User_sell_id', Auth::user()->id)
->orderBy('updated_at', 'DESC')->get();

    $currentChat = Chat::where('User_bay_id', Auth::user()->id)
->where('User_sell_id', $request['id'])->get();

    if ($request['id'] == Auth::user()->id) {
        $currentChat = false;
    } else {
        if (empty($currentChat[0])) {
            $Chat = new Chat();

            $Chat->insertGetId([
                'Text' => json_encode([]),
                'User_sell_id' => $request['id'],
                'User_bay_id' => Auth::user()->id,
                'created_at' => date('Y.m.d H:i:s'),
                'updated_at' => date('Y.m.d H:i:s'),
                'check' => 0
            ]);

            $currentChat = Chat::where('User_bay_id', Auth::user()->id)
->where('User_sell_id', $request['id'])->get()[0];

            $ChatBay = Chat::where('User_bay_id', Auth::user()->id)
->orderBy('updated_at', 'DESC')->get();
            $ChatSell = Chat::where('User_sell_id', Auth::user()->id)
->orderBy('updated_at', 'DESC')->get();
        } else {
            $currentChat = $currentChat[0];
        }
    }

    return view('pages.chat', [
        'user' => $user,
        'currentChat' => $currentChat,
        'ChatBay' => $ChatBay,
        'ChatSell' => $ChatSell
    ]);
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		72

```
}
```

OrderController.php

```
class OrderController extends Controller
{
    public function orderAdvertPage(string $id)
    {
        return view('pages.orderPage', ['id' => $id]);
    }
    public function orderListPage()
    {
        $order = DB::table('Order');

        if (empty($order))
            return view('pages.orderListPage', ['data' => false]);

        $order = $order->orderBy('created_at', 'DESC')->paginate(5);

        return view('pages.orderListPage', ['data' => $order]);
    }
    public function orderAdvert(Request $request, string $id)
    {
        $requestData = $request->validate([
            'FirstName' => 'required|max:255',
            'LastName' => 'required|max:255',
            'MiddleName' => 'required|max:255',
            'Email' => 'required|email|max:255',
            'PhoneNumber' => 'required|digits:9|numeric|decimal:0'
        ]);

        $order = new Order();

        $order->insertGetId([
            'FirstName' => $requestData['FirstName'],
            'LastName' => $requestData['LastName'],
            'MiddleName' => $requestData['LastName'],
            'Email' => $requestData['Email'],
            'PhoneNumber' => $requestData['PhoneNumber'],
            'User_id' => Auth::user()->id,
            'Advert_id' => $id,
            'created_at' => date('Y.m.d H:i:s')
        ]);

        return redirect()->route('homePage');
    }
    public function orderDelete(string $id)
    {
        $order = Order::find($id);

        if (!$order) {
            abort(404, "Замовлення з цим індифікатором ($id) не знайдено");
        }
    }
}
```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		73

```

$Order->delete();

$Order = DB::table('Order');

if (empty($Order))
    return view('pages.orderListPage', ["data" => false]);

$Order = $Order->orderBy('created_at', 'DESC')->paginate(5);

return view('pages.orderListPage', ['data' => $Order]);
}
}

```

ProfileController.php

class ProfileController extends Controller

```

{
    public function profilePage()
    {
        return view('pages.profile');
    }
    public function likeAdvertPage()
    {
        $subscribes = json_decode(Auth::user()->Subscribes);

        if (empty($subscribes))
            return view('pages.likeAdvert', ["data" => false]);

        $arr = [];
        $Users = Users::find(Auth::user()->id);
        $Advert = Advert::whereIn('id', $subscribes)->get();

        foreach ($Advert as $elem) {
            array_push($arr, strval($elem->id));
        }

        $Users->update([
            'Subscribes' => json_encode($arr)
        ]);

        $Advert = $Advert->toQuery()->orderBy('advert.created_at', 'DESC')->paginate(4);

        return view('pages.likeAdvert', ["data" => $Advert]);
    }

    public function myAdvertPage()
    {
        $Advert = Advert::where('User_id', Auth::user()->id)->get();

        if (empty($Advert[0]))
            return view('pages.myAdvert', ["data" => false]);
    }
}

```

```

$Advert = $Advert->toQuery()->orderBy('advert.created_at', 'DESC')->paginate(4);

return view('pages.myAdvert', ["data" => $Advert]);
}

public function likeAdvertChange(Request $request)
{
    $Users = Users::find($request['id']);

    $Users->insertGetId([
        'Subscribe' => json_encode($request['Subscribe'])
    ]);

    return view('pages.likeAdvert');
}

public function updateAdvertPage(string $id)
{
    $Advert = Auth::user()->advert()->find($id);

    if (empty($Advert))
        return redirect()->route('homePage');

    return view('pages.updateAdvert', ["data" => $Advert]);
}

public function updateAdvert(Request $request, string $id)
{
    $requestData = $request->validate([
        'Name' => 'required|max:255',
        'Description' => 'required',
        'Category' => 'required',
        'Price' => 'required|numeric|decimal:0,2|min:0',
        'Images' => 'required|array',
        'City' => 'required|city_list_check',
        'Images.*' => 'image'
    ], [
        'Images.*.image' => 'The Images field must be an image.'
    ]);

    $Advert = Advert::find($id);
    foreach ($Advert->Images as $image) {
        unlink('storage/' . $image);
    }

    $arr = [];

    foreach ($request['Images'] as $image) {
        $ImagesPath = $image->store('uploads', 'public');
        $manager = new ImageManager(Driver::class);
        $manager->read("storage/{$ImagesPath}")
            ->scaleDown(800, 800)->save();
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						75
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        array_push($arr, $ImagesPath);
    }

    $Advert->update([
        'Name' => $requestData['Name'],
        'Description' => nl2br($requestData['Description']),
        'Price' => $requestData['Price'],
        'Images' => $arr,
        'Category' => $requestData['Category'],
        'City' => $requestData['City']
    ]);

    return redirect()->route('myAdvertPage');
}
}

```

PusherController.php

```

class PusherController extends Controller
{
    public function broadcast(Request $request)
    {
        $currentChat = Chat::find($request['id']);

        if (empty($currentChat)) {
            $chat = new Chat();

            $array = [
                [
                    'key' => 1,
                    'message' => $request['message'],
                    'data' => date('Y.m.d H:i:s'),
                    'chaek' => 0,
                ],
            ];

            $chat->insertGetId([
                'Text' => json_encode($array),
                'User_sell_id' => $request['id'],
                'User_bay_id' => Auth::user()->id,
                'created_at' => date('Y.m.d H:i:s'),
                'updated_at' => date('Y.m.d H:i:s'),
                'check' => 0
            ]);
        } else {
            $mainArray = $currentChat->Text;

            $arrayAdd = [
                'key' => ($currentChat->User_bay_id == Auth::user()->id) ? 1 : 0,
                'message' => $request['message'],
                'data' => date('Y.m.d H:i:s'),
                'chaek' => 0,
            ];
        }
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						76
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        array_push($mainArray, $arrayAdd);

        $currentChat->Text = $mainArray;
        $currentChat->save();
    }

    broadcast(new PusherBroadcast(
        $request['message'],
        $request['id'],
        $request['idChat']
    ))->toOthers();

    return view('components.chat-message-broadcast', [
        'message' => $request['message'],
        'data' => date('Y.m.d H:i:s')
    ]);
}

public function receive(Request $request)
{
    $chat = Chat::find($request['id']);

    $user = (Auth::user()->id == $chat->User_sell_id)
        ? Users::find($chat->User_buy_id)
        : Users::find($chat->User_sell_id);

    return view('components.chat-message-receive', [
        'message' => $request['message'],
        'user' => $user,
        'data' => date('Y.m.d H:i:s')
    ]);
}

public function openChatSell(Request $request)
{
    $currentChat = Chat::where('User_buy_id', Auth::user()->id)
        ->where('User_sell_id', $request['id'])->get()[0];

    return view('components.chat-message-block', ['currentChat' => $currentChat]);
}

public function openChatBuy(Request $request)
{
    $currentChat = Chat::where('User_buy_id', $request['id'])
        ->where('User_sell_id', Auth::user()->id)->get()[0];

    return view('components.chat-message-block', ['currentChat' => $currentChat]);
}
}

```

RegistrationControler.php

```

class RegistrationControler extends Controller
{

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public function creatUser(Request $request)
{
    $requestData = $request->validate([
        'FirstName' => 'required|max:255',
        'LastName' => 'required|max:255',
        'email' => 'required|email|unique:users|max:255',
        'password' => 'required|min:3|max:255|confirmed:password-confirm',
    ], [
        'email.unique' => 'Користувач з такою поштою вже існує'
    ]);

    $Users = new Users();

    $user = $Users->create([
        'FirstName' => $requestData['FirstName'],
        'LastName' => $requestData['LastName'],
        'email' => $requestData['email'],
        'password' => Hash::make($requestData['password'])
    ]);

    Auth::login($user);

    return redirect()->route('homePage');
}

public function deleteUser(Request $request): mixed
{
    $user = Auth::user();
    $adverts = Advert::where('User_id', Auth::user()->id)->get();

    if ($user->ProfilePicture != null)
        unlink('storage/' . $user->ProfilePicture);

    foreach ($adverts as $advert) {
        foreach ($advert->Images as $image) {
            unlink('storage/' . $image);
        }
    }

    $user->advert()->delete();
    $user->chatGet()->delete();
    $user->chatGive()->delete();
    $user->comments()->delete();
    $user->order()->delete();
    $user->delete();

    Auth::logout();

    $request->session()->invalidate();
    $request->session()->regenerateToken();

    return redirect()->route('homePage');
}

```

```

    }

    public function showRegistar()
    {
        if (Auth::check())
            return redirect()->route('homePage');

        return view('auth.register');
    }
}

```

Advert.php

```

class Advert extends Model
{
    protected $table = "Advert";

    protected $fillable = [
        "Categori",
        "Name",
        "Description",
        "Price",
        "Category",
        "City",
        "Images"
    ];

    protected $casts =[
        'Images' => 'array'
    ];

    public function users(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_id', 'id');
    }
}

```

Chat.php

```

class Chat extends Model
{
    use HasFactory;
    protected $table = "Chat";

    protected $fillable = [
        "check",
        "Text"
    ];

    protected $casts = [
        'Text' => 'array',
    ];

    public function UserGet(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_bay_id');
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						79
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    public function UserGive(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_sell_id');
    }
}

```

Comments.php

```

class Comments extends Model
{
    protected $table = "Comments";

    protected $fillable = [
        "check",
        "Text"
    ];

    public function UserGet(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_get_id');
    }

    public function UserGive(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_give_id');
    }
}

```

Order.php

```

class Order extends Model
{
    use HasFactory;
    protected $table = "Order";

    protected $fillable = [
        "FirstName",
        "LastName",
        "MiddleName",
        "Email",
        "PhoneNumber"
    ];

    public function users(): BelongsTo
    {
        return $this->belongsTo(Users::class, 'User_id');
    }

    public function advert(): BelongsTo
    {
        return $this->belongsTo(Advert::class, 'Advert_id');
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						80
Змн.	Арк.	№ докум.	Підпис	Дата		

Users.php

```
class Users extends Authenticatable
{
    protected $table = "Users";

    protected $fillable = [
        "Rights",
        "FirstName",
        "LastName",
        "email",
        "Email_verified_at",
        "ProfilePicture",
        "PhoneNumber",
        "Subscribes",
        "password"
    ];

    protected $casts = [
        'Subscribe' => 'array'
    ];

    protected $hidden = [
        'password'
    ];

    public function hasRole($role)
    {
        return $this->Rights == $role ? true : false;
    }

    public function advert(): HasMany
    {
        return $this->hasMany(Advert::class, 'User_id');
    }

    public function chatGet(): HasMany
    {
        return $this->hasMany(Chat::class, 'User_bay_id');
    }

    public function chatGive(): HasMany
    {
        return $this->hasMany(Chat::class, 'User_sell_id');
    }

    public function comments(): HasMany
    {
        return $this->hasMany(Comments::class, 'User_give_id');
    }

    public function order(): HasMany
```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    {
        return $this->hasMany(Order::class, 'User_id');
    }
}

```

AdvertRepository.php

```

class AdvertRepository
{
    public function getAdvert(string $direction, string $PriceMin, string $PriceMax, string
$Category, string $City, string $Text): LengthAwarePaginator
    {
        $query = Advert::query();

        $query = $query->orderBy('advert.created_at', $direction);

        if ($Category != "")
            $query = $query->where('advert.Category', $Category);

        if ($City != "")
            $query = $query->where('advert.City', $City);

        if ($PriceMin != "")
            $query = $query->where('advert.Price', '>=', $PriceMin);

        if ($PriceMax != "")
            $query = $query->where('advert.Price', '<=', $PriceMax);

        if ($Text != "")
            $query = $query->where('advert.Name', 'like', "%{$Text}%");

        return $query->paginate(4);
    }
}

```

CheckUserRole.php

```

class CheckUserRole
{
    /**
     * Handle an incoming request.
     *
     * @param \Closure(Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\
Response) $next
     */
    public function handle(Request $request, Closure $next, string $role): Response
    {
        $user = Auth::user();

        if (isNull($user)) {
            if ($user->hasRole('ROLE_ADMIN')) {
                return $next($request);
            }

            abort(403, 'У користувача недостатньо прав');
        }
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						82
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }

    return redirect()->to("/");
}
}

```

web.php

```

Route::match(['post', 'get'], '/', [AdvertController::class, 'getAdvertOnPage'])->name("homePage");

Route::get('advertPage/{id}', [AdvertController::class, 'getAdvertItem'])->name("getAdvertItem");

Route::get('login', [AuthController::class, 'showLogin'])->name("loginPage");
Route::post('login', [AuthController::class, 'login'])->name("login");

Route::get('register', [RegistrationController::class, 'showRegistrar'])->name("registerPage");
Route::post('register', [RegistrationController::class, 'creatUser'])->name("register");

Route::middleware("auth")->group(function () {

    Route::post('logout', [AuthController::class, 'logout'])->name("logout");

    Route::post('advertPage/{id}/Subscribes', [AdvertController::class, 'AdvertItemSubscribes']);

    Route::get('createAdvert', [AdvertController::class, 'advertCreatePage'])->name("createAdvertPage");
    Route::post('createAdvert', [AdvertController::class, 'createAdvert'])->name("createAdvert");

    Route::post('deleteAdvert/{id}', [AdvertController::class, 'deleteAdvert'])->name("deleteAdvert");

    Route::post('deleteUser', [RegistrationController::class, 'deleteUser'])->name("deleteUser");

    Route::get('orderAdvert/{id}', [OrderController::class, 'orderAdvertPage'])->name("orderAdvertPage");
    Route::post('orderAdvert/{id}', [OrderController::class, 'orderAdvert'])->name("orderAdvert");

    Route::prefix('account')->controller(ProfileController::class)->group(function () {
        Route::get('profile', 'profilePage')->name("profilePage");
        Route::prefix('profile')->controller(ChangeProfileController::class)->group(function () {
            Route::post('changePicture', 'changeProfilePicture')->name("changeProfilePicture");
            Route::post('changeName', 'changeFirstAndLastName')->name("changeFirstLastName");
            Route::post('changeEmail', 'changeEmail')->name("changeEmail");
            Route::post('changePassword', 'changePassword')->name("changePassword");
            Route::post('changeNumber', 'changePhoneNumber')->name("changePhoneNumber");
        });

        Route::get('chat', [ChatController::class, 'chatPage'])->name("chatPage");
        Route::post('chat', [ChatController::class, 'chatSendUser'])->name("chatSendUser");
        Route::post('chat/openChatSell', [PusherController::class, 'openChatSell'])->name("openChatSell");
    });
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

Route::post('chat/openChatBay', [PusherController::class, 'openChatBay'])->name("openChatBay");
Route::post('chat/receive', [PusherController::class, 'receive'])->name('receive');
Route::post('chat/broadcast', [PusherController::class, 'broadcast'])->name('send');

Route::get('likeAdvert', 'likeAdvertPage')->name("likeAdvertPage");
Route::post('likeAdvert', 'likeAdvertChange')->name("likeAdvertChange");

Route::get('myAdvert', 'myAdvertPage')->name("myAdvertPage");
Route::get('myAdvert/updateAdvert/{id}', 'updateAdvertPage')->name("updateAdvertPage");
Route::post('myAdvert/updateAdvert/{id}', 'updateAdvert')->name("updateAdvert");
});

Route::middleware("role:ROLE_ADMIN")->controller(AdvertController::class)->group(function () {
    Route::get('orderListAdvert', [OrderController::class, 'orderListPage'])->name("orderListPage");
    Route::get('orderListAdvert/{id}', [OrderController::class, 'orderDelete'])->name("orderDelete");
});
});

```

app.js

```

function changeIcon() {
    circuit = 'm8 2.748-.717-.737C5.6.281 2.514.878 1.4 3.053c-.523 1.023-.641 2.5.314 4.385.92 1.815 2.834 3.989 6.286 6.357 3.452-2.368 5.365-4.542 6.286-6.357.955-1.886.838-3.362.314-4.385C13.486.878 10.4.28 8.717 2.01zM8 15C-7.333 4.868 3.279-3.04 7.824 1.143q.09.083.176.171a3 3 0 0 1 .176-.17C12.72-3.042 23.333 4.867 8 15';
    fill = `M8 1.314C12.438-3.248 23.534 4.735 8 15-7.534 4.736 3.562-3.248 8 1.314`;
    element = document.getElementById('elementFollow');

    if (element.getAttribute('d') == fill) {
        element.setAttribute('d', circuit);
    }
    else {
        element.setAttribute('d', fill);
    }
};

function showPhoneNumber(number) {
    if (" +380" + number == document.getElementById('numberButton').innerHTML)
        document.getElementById('numberButton').innerHTML = 'Показати телефон';
    else
        document.getElementById('numberButton').innerHTML = " +380" + number;
};

function showCities(city) {
    blockCities = document.getElementById('blockCities');

    if (city.length >= 2) {
        fetch("./storage\\Cities\\CitiesAndVillages.json")
    }
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

.then((response) => response.json())
.then((json) => {
    block = "";
    count = 0;

    json.filter((element) => {

        if (count < 10 && new RegExp(`^${city}`, 'i').test(element['object_name'])) {
            if (count == 0)
                block = '<div class="z-1 overflow-y-auto position-absolute" style="height: 175px; width: auto;"><div class="card rounded-0">';

            count++;
            return true;
        }
        return false;
    }).forEach(element => {

        objectName = element["object_name"].charAt(0) +
            element["object_name"].slice(1).toLowerCase();
        region = element["region"].charAt(0) + element["region"].slice(1).toLowerCase();

        block += `<div class="card-body p-2 py-1">
            <a class="dropdown-item" role="button" onclick="choiceCities(` + objectName +
                `)">

                <p class="card-text mb-0">` + objectName + `</p>
                <small class="text-body-secondary">` + region + `</small>
                <hr class="my-1" />
            </a>
        </div>`;

    });

    block += '</div></div>'
    blockCities.innerHTML = block;
});
}
else {
    blockCities.innerHTML = "";
}
};

function choiceCities(city) {
    document.getElementById('City').value = city;
    document.getElementById('blockCities').innerHTML = "";
};

function sendForm(event) {
    event.preventDefault();
    document.getElementById('logout-form').submit();
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

}

function showElement(idblock) {
    element = document.getElementById(idblock);

    if (element.className == 'card-body')
        element.className = 'card-body d-none';
    else
        element.className = 'card-body';
}

function showchatSell() {
    document.getElementById('ChatSell').className = 'list-unstyled mb-0';
    document.getElementById('ChatBay').className = 'list-unstyled mb-0 d-none';
    document.getElementById('bthSell').className = 'btn btn-info';
    document.getElementById('bthBay').className = 'btn btn-success'
}

function showchatBay() {
    document.getElementById('ChatBay').className = 'list-unstyled mb-0';
    document.getElementById('ChatSell').className = 'list-unstyled mb-0 d-none';
    document.getElementById('bthBay').className = 'btn btn-info';
    document.getElementById('bthSell').className = 'btn btn-success'
}

function clickElement() {
    document.getElementById('Images').click();
}

function showFiles(name, length) {
    document.getElementById('FileName').value = name + length;
}

```

addStyle.css

```

.btn-main {
    background-color: #fff;
}

.btn-main:hover {
    background-color: #a175ed;
    color: #fff;
}

.btn-advert {
    background-color: #C19FFA;
}

.btn-advert:hover {
    background-color: #a175ed;
}

.font-color a {

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						86
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    color: #C19FFA;
}

.font-color a:hover {
    color: #5E34A3;
}

.bg-main {
    background-color: #ebe0ff;
}

.pp-sm {
    border: solid #9C88BB 1px;
    height: 38px;
    border-radius: 100%;
}

.pp-mm {
    border: solid #9C88BB 1px;
    height: 52px;
    border-radius: 100%;
}

.pp-lg {
    height: 138px;
    border-radius: 100%;
}

.logo {
    height: 30px;
    border-right: 1px solid #333;
    margin-right: 5px;
    padding-right: 6px
}

.checked {
    color: orange;
}

::-webkit-scrollbar {
    width: 8px;
    color: black;
}

::-webkit-scrollbar-track {
    -webkit-box-shadow: inset 0 0 6px rgba(0, 0, 0, 0.3);
    border-radius: 10px;
}

::-webkit-scrollbar-thumb {
    border-radius: 10px;
    background-color: rgba(156, 136, 187, 0.9);
}

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
						87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

}

input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
    -webkit-appearance: none;
    margin: 0;
}

```

PusherBroadcast.php

```

class PusherBroadcast implements ShouldBroadcast
{
    use Dispatchable, InteractsWithSockets, SerializesModels;

    public string $message;
    public string $id;
    public string $idChat;

    public function __construct(string $message, string $id, string $idChat)
    {
        $this->message = $message;
        $this->id = $id;
        $this->idChat = $idChat;
    }

    public function broadcastOn()
    {
        return "chat.{ $this->idChat}";
    }

    public function broadcastAs(): string
    {
        return 'chat';
    }
}

```

AppServiceProvider.php

```

class AppServiceProvider extends ServiceProvider
{
    /**
     * Register any application services.
     */
    public function register(): void
    {
        //
    }

    /**
     * Bootstrap any application services.
     */
    public function boot(): void
    {
        Validator::extend('city_list_check', function ($attribute, $value, $parameters, $validator) {

```

					ІПЗ.КР.Б – 121 – 25 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

$cities = json_decode(file_get_contents('.\storage\Cities\CitiesAndVillages.json'));

foreach ($cities as $city) {
    $listCity = mb_convert_case(mb_strtolower($city->object_name), MB_CASE_TITLE,
"UTF-8");
    $valueCity = mb_convert_case($value, MB_CASE_TITLE, "UTF-8");

    if ($listCity == $valueCity)
        return true;
    }
    return false;
}, 'Місто повинно бути з списку');

Validator::extend('old_value', function ($attribute, $value, $parameters, $validator) {
    $userValue = Auth::user()->$attribute;

    if ($userValue == $value)
        return false;

    return true;
}, 'Нове значення не повинно бути схоже на старе');

Validator::extend('old_password_check', function ($attribute, $value, $parameters, $validator)
{
    $userValue = Auth::user()->password;

    if (Hash::check($value, $userValue))
        return false;

    return true;
}, 'Новий пароль не повинно бути схожим на старий');
}
}

```