

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "бакалавр"

на тему: Розробка веб-додатку для безпечної оренди житла

Виконала: студентка групи 4145



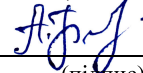
(підпис)

Коркасенко А. В.

Керівник роботи:

старший викладач

(посада, науковий ступінь вчене звання)



(підпис)

Павленко А. Ю..

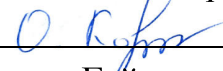
м. Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 124 "Системний аналіз"
Освітня програма "Системний аналіз"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми


к.т.н, доц. Гайдаєнко О. В.
" ____ " _____ 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "бакалавр"

Студенту Коркасенко Аліні Владиславівні

(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка веб-додатку для безпечної оренди житла

Керівник роботи Павленко Альона Юріївна

Затверджені наказом ректора № 295-уч від "27" березня 2025 року.

2. Термін подання роботи: 20 червня 2025 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем,

4. Перелік питань, що належать до розробки (найменування розділів)

1). Аналіз предметної галузі. Постановка задач. 2). Розробка проекту інформаційної підсистеми. 3). Реалізація проекту підсистеми. 4). Охорона праці.

5. Перелік презентаційних матеріалів Схема запропонованої концепції, Діаграма варіантів використання, Концептуальна модель даних, Фізична модель даних, Діаграма компонентів, Діаграма класів, Діаграма переходу станів, Діаграма розгортання системи, Результат розробки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Павленко А. Ю.	03.02.2025	22.02.2025
2	Павленко А. Ю.	22.02.2025	21.03.2025
3	Павленко А. Ю.	21.03.2025	25.04.2025
4	Павленко А. Ю.	25.04.2025	07.06.2025

7. Дата видачі завдання 03 лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Аналіз предметної галузі	13.02.2025	
2	Постановка задачі	15.02.2025	
3	Розробка концепції	22.02.2025	
4	Розробка проекту ІС	21.03.2025	
5	Реалізація проекту	25.04.2025	
6	Розробка питань з охорони праці	07.06.2025	
7	Розробка графічних матеріалів	12.06.2025	
8	Подання роботи на перевірку на плагіат	10.06.2025	
9	Подання роботи рецензенту	15.06.2025	
10	Подання роботи на захист	20.06.2025	

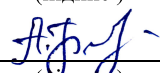
Студент


 (підпис)

Коркасенко А. В.

(ПІБ)

Керівник роботи


 (підпис)

Павленко А. Ю.

(ПІБ)

АНОТАЦІЯ

Коркасенко А.В. Розробка веб-додатку для безпечної оренди житла

Кваліфікаційна робота на здобуття ступеня вищої освіти «бакалавр» за спеціальністю 124 «Системний аналіз» галузі знань 12 «Інформаційні технології». – Національний університет кораблебудування імені адмірала Макарова, Миколаїв, 2025.

Пояснювальна записка: 95 с., 31 рис., 18 посилань.

У бакалаврській кваліфікаційній роботі виконується розробка та реалізація веб додатку для безпечної оренди житла. Основна мета проєкту полягає в створенні новітнього веб-додатку, який забезпечить зручний та безпечний процес оренди нерухомості.

Додаток має надати орендарям можливість швидко знаходити житло відповідно до власних потреб, переглядати характеристики осель, бронювати їх, здійснювати оплату та комунікувати з орендодавцями. В свою чергу орендодавці зможуть легко розміщувати свої оголошення і спілкуватися з клієнтами.

У роботі детально розглядаються питання безпечної оренди житла , безпеки користувачів сайту від шахраїв, поняття та структура нерухомості в Україні. Проєкт включає клієнтську частину платформи , яка буде реалізована з використанням HTML, CSS та JavaScript, що забезпечить зручний інтерфейс. Серверна логіка працюватиме на основі PHP у поєднанні з MySQL.

Висновком дипломної роботи було створено веб-додаток для безпечної оренди житла, що враховує проблеми шахрайства та незручностей у пошуку.

Ключові слова: веб-додаток, оренда, нерухомість, PHP, MySQL.

ABSTRACT

Korkasenko A.V. Development of a web application for secure housing rental

Qualification work for the degree of higher education “Bachelor” in the specialty 124 “System Analysis”, field of knowledge 12 “Information Technology.” - Admiral Makarov National University of Shipbuilding, Mykolaiv, 2025.

Explanatory note: 95 p., 31 figures, 18 references.

The bachelor's qualification work involves the development and implementation of a web application for secure housing rental. The main goal of the project is to create a state-of-the-art web application that will provide a convenient and secure rental process.

The application should allow tenants to quickly find housing according to their needs, view the characteristics of the apartments, book them, make payments, and communicate with landlords. In turn, landlords will be able to easily post their ads and communicate with customers.

The paper discusses in detail the issues of safe rental housing, the security of site users from fraudsters, the concept and structure of real estate in Ukraine. The project includes the client side of the platform, which will be implemented using HTML, CSS and JavaScript to provide a user-friendly interface. The server logic will be based on PHP in combination with MySQL.

The conclusion of the thesis was the creation of a web application for secure rental housing, taking into account the problems of fraud and inconvenience in the search

Keywords: web application, rental, real estate, PHP, MySQL .

ЗМІСТ

ЗМІСТ	6
ВСТУП.....	8
РОЗДІЛ I. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ. ПОСТАНОВКА ЗАДАЧІ.....	10
1.1 Дослідження та аналіз предметної галузі	10
1.2 Аналіз існуючих рішень.....	15
1.3 Визначення проблем.....	19
1.4 Постановка задачі.....	20
1.5 Розробка концепцій рішення проблем.....	22
РОЗДІЛ II. РОЗРОБКА ПРОЄКТУ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ.....	24
2.1 Загальносистемні рішення.....	24
2.1.1 Загальний опис системи	24
2.1.2 Розробка діаграми варіантів використання	26
2.2 Рішення з інформаційного забезпечення	28
2.2.1 Визначення сутностей інформаційної підсистеми	28
2.2.2 Розробка логічної моделі даних.....	31
2.2.3 Концептуальна модель даних.....	29
2.2.4 Розробка фізичної моделі даних.....	34
2.3 Рішення з технічного забезпечення.....	36
2.3.1 Вимоги до технічного забезпечення	36
2.3.2 Склад технічного забезпечення	37
2.4 Рішення з програмного забезпечення.....	38
2.4.1 Розробка схеми структури інформаційної підсистеми	38
2.4.2 Розробка діаграми компонентів.....	40
2.4.3 Розробка діаграми переходів станів.....	42
2.4.4 Розробка проекту інтерфейсу інформаційної підсистеми	44
2.4.5 Розробка діаграми розгортання підсистеми.....	46
РОЗДІЛ III. РЕАЛІЗАЦІЯ ПРОЄКТУ ПІДСИСТЕМИ	49
3.1 Аргументація вибору засобів розробки	49
3.2 Розробка структури програмного забезпечення	51
РОЗДІЛ IV. ОХОРОНА ПРАЦІ.....	55
4.1 Загальні питання охорони праці	55
4.2 Структура управління питаннями охорони праці на підприємстві	57
4.3 Шкідливі та небезпечні фактори, що супроводжують працю програміста.....	59

	7
4.4 Ергономічні вимоги до робочого місця при роботі за комп'ютером.....	61
4.5 Нормативні вимоги до робочої площі на одного працюючого за комп'ютером.....	63
ВИСНОВОК.....	65
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	68
ДОДАТОК А –Технічне завдання.....	70
ДОДАТОК Б – Програмний код.....	74
ДОДАТОК В – Інструкція користувача.....	92

ВСТУП

У сучасному цифровому світі технології суттєво змінюють традиційні підходи до пошуку та оренди житла. Все більше людей віддають перевагу онлайн-платформам, які надають змогу швидко та зручно знаходити необхідну нерухомість, взаємодіяти з орендодавцями та здійснювати фінансові операції. Проте разом із перевагами цифровізації виникає низка проблем, серед яких шахрайство, недостовірність інформації, відсутність механізмів безпеки та складність управління угодами. Тому розробка безпечного веб-додатку для оренди житла є актуальним завданням, що відповідає сучасним потребам ринку.

Метою цієї роботи є створення інноваційного веб-додатку, який забезпечить зручний та безпечний процес оренди нерухомості. Додаток має надати орендарям можливість швидко знаходити житло відповідно до власних потреб, переглядати характеристики об'єктів, бронювати їх, здійснювати оплату та комунікувати з орендодавцями. Водночас орендодавці отримають ефективний інструмент для розміщення своїх пропозицій, управління ними та взаємодії з потенційними клієнтами.

Головний акцент у розробці робиться на забезпеченні безпеки даних та захисту користувачів від шахрайства. Для цього передбачена система реєстрації та авторизації з верифікацією особистості, інтеграція безпечних методів платежів, а також засоби перевірки інформації про житло та його власників.

Технічна реалізація веб-додатку передбачає використання сучасних веб-технологій. Клієнтська частина буде створена на основі HTML, CSS та JavaScript, що забезпечить гнучкий і адаптивний дизайн. Серверна частина базується на PHP та MySQL, що дозволить ефективно зберігати та обробляти дані користувачів, а також підтримувати швидко та стабільну роботу платформи.

Розроблений веб-додаток сприятиме спрощенню процесу оренди житла, підвищенню рівня довіри між сторонами угоди та створенню комфортного середовища для взаємодії орендарів і орендодавців. У перспективі система може бути розширена додатковими функціями, такими як рейтинг користувачів,

автоматизований підбір варіантів за заданими параметрами та інтеграція з картографічними сервісами.

Таким чином, створення веб-додатку для безпечної оренди житла є важливим кроком у напрямку цифрової трансформації ринку нерухомості, що дозволить підвищити рівень безпеки, зручності та ефективності в цьому процесі.

РОЗДІЛ І. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ. ПОСТАНОВКА ЗАДАЧІ

1.1 Дослідження та аналіз предметної галузі

Нерухомість визначається як вид майна, який законодавчо визнано нерухомим. Це означає, що сюди належать земельні ділянки, об'єкти, пов'язані із землею, будівлі та споруди, переміщення яких неможливе без значних пошкоджень або втрати функціональності [1]. В Україні поняття "нерухомість" закріплено в Законі України «Про іпотеку», де нерухомим майном вважаються земельні ділянки та об'єкти, розташовані на них, переміщення яких призводить до їхнього знецінення або зміни призначення (стаття 1) [2]. Крім того, Закон України «Про збір на обов'язкове державне пенсійне страхування» також визначає нерухомість, включаючи житлові будинки, квартири, садові будинки, дачі, гаражі та інші об'єкти, віднесені до першої групи основних фондів за Законом України «Про оподаткування прибутку підприємств» (стаття 1) [3].

Зважаючи на вищенаведені законодавчі акти, нерухомість характеризується такими ознаками:

1. Вона включає земельні ділянки або будівлі і споруди, що нерозривно пов'язані із землею.
2. Переміщення об'єктів нерухомості є неможливим або призводить до їхньої втрати вартості чи зміни призначення.

З правової точки зору, нерухомість є об'єктом права власності, щодо якого укладаються угоди — купівля-продаж, оренда, передача прав на володіння та користування, припинення цих прав.

У Державному реєстрі прав зазначені наступні види нерухомого майна:

1. Житлові будинки.
2. Квартири.
3. Будівлі, що використовуються для перебування людей, розміщення рухомого майна, зберігання цінностей або виробництва.

4. Приміщення всередині житлових будинків, квартир та будівель.
5. Споруди, що виконують специфічні технічні функції, не віднесені до будівель та приміщень.

Класифікація об'єктів нерухомості з точки зору практичного управління може поділятися за кількома критеріями:

1. Спеціалізація:

- Спеціалізована нерухомість — це об'єкти, що мають особливий характер, через що рідко продаються на відкритому ринку. Для такого майна, через його специфічні конструктивні особливості, розміри та характеристики, часто немає ринку в певній місцевості або взагалі. Прикладом можуть бути заводи (нафтопереробні, хімічні, металургійні), а також гідро- чи електростанції.

- Неспеціалізована нерухомість — це всі інші об'єкти, на які є попит на загальному ринку, для використання в існуючих чи аналогічних цілях.

2. Ступінь готовності до експлуатації:

- Об'єкти, введені в експлуатацію.
- Об'єкти, що потребують реконструкції або капітального ремонту перед експлуатацією.

- Незавершені будівництвом об'єкти.

3. Походження:

- Природні об'єкти — земельні ділянки, насадження, водойми, ділянки надр; вони також відомі як "нерухомість за природою".

- Штучні об'єкти — всі інші об'єкти, віднесені до "нерухомості за законом".

4. Відтворюваність у натуральній формі:

- Відтворювані об'єкти — будівлі, споруди, насадження.
- Невідтворювані об'єкти — земельні ділянки, окремі водні об'єкти, ділянки надр.

На практиці об'єкти нерухомості часто поділяють за сегментами ринку, виходячи зі споживчих уподобань, що дозволяє виділити три основні категорії: земельні ділянки, житлова та нежитлова нерухомість.

Житлова нерухомість — це об'єкти, призначені для проживання. З комерційної точки зору цей сегмент включає відносини, пов'язані з орендою, купівлею-продажем, а також будівництвом і експлуатацією житлового фонду. Серед житлової нерухомості виділяються:

- малоповерхові (до 3 поверхів),
- багатоповерхові (від 4 до 9 поверхів),
- підвищеної поверховості (від 10 до 20 поверхів),
- висотні (понад 20 поверхів) будинки. Інші приклади — квартири, кімнати, секції, поверхи під'їздів, дачі.

Нежитлова нерухомість охоплює об'єкти комерційного, промислового, інституційного та рекреаційного призначення, а також готелі та мотелі.

- Комерційна нерухомість — офісні приміщення, торговельні об'єкти, автопарковки, АЗС та інші об'єкти для сфери послуг.
- Промислова нерухомість — промислові будівлі, інфраструктурні споруди, заводи, технопарки, склади.
- Інституційна нерухомість — приміщення для урядових, муніципальних органів, навчальних закладів, лікарень.
- Рекреаційна нерухомість — об'єкти для відпочинку, такі як клуби (наприклад, яхт-клуби), спортивні стадіони, басейни.
- Готелі та мотелі — особлива категорія, що фактично є житловою нерухомістю, але призначена для тимчасового проживання.

Ці класифікації є досить чіткими, але можуть не охоплювати всі наявні або нові види нерухомості. Тому додатково виділяється нерухомість змішаного типу, що поєднує характеристики кількох категорій.

Термін «ринок нерухомості» відображає сукупність правових і комерційних відносин, що виникають навколо угод з нерухомим майном, таких як оренда, ку-

півля-продаж, іпотека тощо. Інакше кажучи, ринок нерухомості — це система ринкових механізмів, які забезпечують створення, передачу та експлуатацію нерухомого майна.

Серед наукових визначень ринку нерухомості можна виділити формулювання Дж. Фрідмана і Н. Ордуей: ринок нерухомості — це набір механізмів, за допомогою яких передаються права власності та інтереси, встановлюються ціни та розподіляється простір між конкуруючими видами землекористування.

Суб'єктами ринку нерухомості є:

- Покупці (орендарі) — основні учасники ринку, що можуть виступати як фізичні, юридичні особи або державні органи, які мають повноваження здійснювати угоди.
- Продавці (орендодавці) — фізичні або юридичні особи, що володіють правом власності на нерухомість.
- Професійні учасники ринку — представники різних професій, що забезпечують виконання необхідних операцій на ринку.

Для подальшого розгляду ринку нерухомості необхідно розуміти поняття попиту та пропозиції, які є основою ціноутворення. У загальному вигляді попит і пропозиція — це універсальна економічна модель, що показує процес формування цін на ринку. Попит представляє потребу ринку у товарах, тоді як пропозиція відображає кількість товару, що доступна або може бути запропонована на ринку. Поняття "попит і пропозиція" вперше було використано Дж. Стюартом у 1767 році в його роботі «Принципи політичної економії».

Попит характеризує не лише потребу покупця у певному товарі чи послугі, але й його готовність сплатити за цей товар або послугу за ціною, яка входить у межі доступного діапазону. У кількісному сенсі, попит показує обсяг товару, який покупці бажають придбати за різними цінами. Закон попиту демонструє зворотну залежність між попитом і ціною: із зростанням ціни обсяг попиту зменшується.

Пропозиція, зі свого боку, відображає готовність та можливість продавців виставляти свої товари на ринок за певними цінами. Кількісно пропозиція — це обсяг товару, який продавці здатні та бажають продати за різними цінами протя-

гом певного періоду. Як і попит, пропозиція залежить від багатьох факторів, зокрема, від виробничих можливостей, цін на інші товари, споживчих уподобань, модних тенденцій, можливостей заміщення товару, інфляційних очікувань тощо. Закон пропозиції стверджує, що, за інших незмінних умов, збільшення ціни призводить до зростання обсягу пропозиції.

Однак слід зазначити, що теорія попиту і пропозиції відображає ідеальну модель, і отримані на її основі ціни є розрахованими в умовах ідеального ринку, що не завжди є досяжним в реальності.

Ринок нерухомості підлягає різноманітним класифікаціям. Наприклад, залежно від правових аспектів, ринок нерухомості можна поділити на ринок купівлі та ринок продажу. За формою укладання угод він поділяється на первинний і вторинний. Первинний ринок стосується об'єктів, що вперше вводяться в експлуатацію, тобто нових будівель або приватизованих об'єктів. Вторинний ринок, у свою чергу, охоплює об'єкти, які вже були в експлуатації й тепер можуть бути перепродані чи передані новим власникам через різні види угод. Важливо, що первинний і вторинний ринки є взаємопов'язаними: ціни на вторинному ринку часто визначають рентабельність будівництва нових об'єктів [4].

Серед найважливіших сегментів ринку нерухомості в Україні виділяють ринки житлової нерухомості, нежитлової нерухомості та земельних ділянок. Основними об'єктами на ринку нерухомості країни є багатоквартирні житлові будинки, зокрема квартири та кімнати. Слід зазначити, що ціни на нерухомість постійно зростають, що може пояснюватися політичною ситуацією або ж штучним підвищенням з боку посередників.

У цій роботі основна увага приділятиметься ринку оренди житлової нерухомості, хоча для загального розуміння також буде розглянуто купівлю та продаж житлового майна. За даними експертів у сфері оцінки нерухомості в Києві, за минулий рік середня вартість квадратного метра на вторинному ринку столиці зросла з 46 700 до 49 880 грн (+3 180 грн). Це зростання яскраво виражене у гривні, внаслідок інфляційного чинника, тоді як у доларовому еквіваленті ціна залишилася практично незмінною. Зазначається, що на ринку купівлі-продажу найбільший

попит мають однокімнатні квартири, а слідом за ними трикімнатні. Крім того, у зв'язку з поточною ситуацією, різко зріс інтерес до заміських будинків.

1.2 Аналіз існуючих рішень

У цьому розділі розглянуто наявні рішення для оренди житла, зокрема платформи на кшталт Airbnb, Booking.com та локальні сервіси оренди. Аналіз проводиться з акцентом на переваги та недоліки щодо забезпечення безпеки користувачів та захисту особистих даних, а також на функціональні можливості, які забезпечують простоту використання. Основною проблемою існуючих платформ є недостатній рівень перевірки орендодавців і відсутність централізованих інструментів для боротьби з шахрайством. Це свідчить про потребу у нових рішеннях, які приділяють особливу увагу верифікації користувачів та захисту транзакцій, забезпечуючи більший рівень безпеки оренди житла онлайн.

Airbnb — це міжнародна онлайн-платформа, яка надає можливість бронювати житло та інші послуги на короткостроковий період у різних куточках світу. Платформа, заснована в 2008 році, пропонує зручний інтерфейс і доступний вибір житлових варіантів для мандрівників, а також дозволяє власникам нерухомості здавати своє житло, отримуючи додатковий дохід. На рис. 1.1. наведено головну сторінку платформи.

Booking — це також міжнародна онлайн-платформа, яка надає можливість бронювати житло. Заснована в 1996 у Нідерландах. На рис. 1.2. наведено головну сторінку платформи

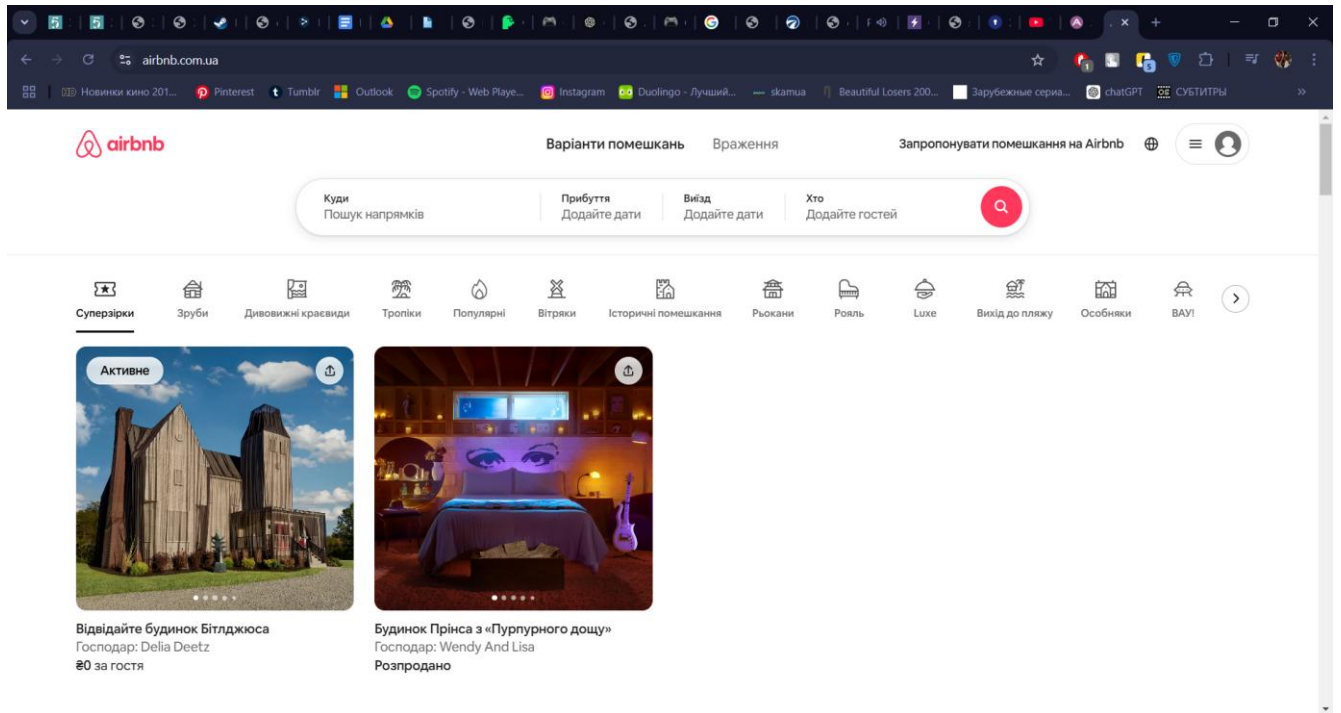


Рисунок 1.1 – Головна сторінка платформи Airbnb

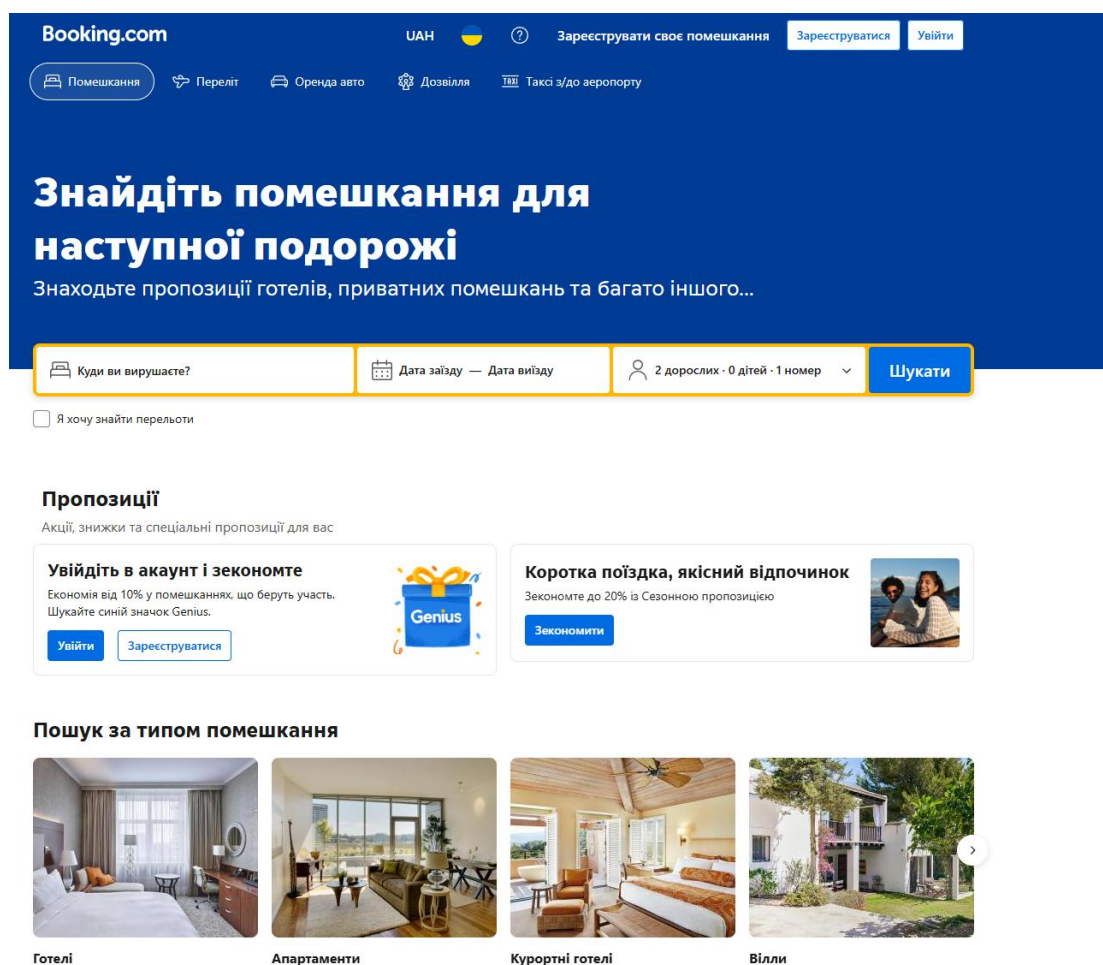


Рисунок 1.2 – Головна сторінка платформи Booking

Основні можливості Airbnb:

- Airbnb дозволяє користувачам легко шукати, порівнювати та бронювати різні типи житла, включаючи апартаменти, приватні кімнати, вілли, замки, бунгало на пляжі, намети, а іноді й унікальні варіанти, такі як будинки на дереві або фургони;

- як орендарі, так і орендодавці можуть залишати відгуки один про одного. Це підвищує довіру між користувачами і допомагає наступним орендарям краще розуміти якість та умови проживання;

Airbnb має комплексну систему безпеки, яка включає перевірку особистості користувачів, страховий захист для орендодавців (через програму "Гарантія господаря") та захист транзакцій. Попри це, платформа все ще стикається з деякими проблемами безпеки та випадками шахрайства, особливо у великих містах.

- Airbnb надає службу підтримки, яка допомагає вирішувати питання, пов'язані з бронюваннями, конфліктами між орендарями та орендодавцями, а також забезпечує швидке реагування у випадках непередбачених ситуацій;

- крім житла, Airbnb також пропонує функцію Airbnb Experiences, що дозволяє бронювати локальні заходи, екскурсії, майстер-класи та інші розваги, які проводять місцеві жителі. Це дозволяє туристам краще відчувати культуру та атмосферу міста.

Переваги платформи Airbnb:

- Airbnb пропонує мільйони варіантів житла у понад 220 країнах.
- у багатьох випадках оренда житла через Airbnb може бути дешевшою альтернативою готелям.

- Airbnb дозволяє вибрати саме той тип житла, який відповідає потребам і бюджету орендаря, включаючи незвичайні варіанти, що відрізняються від традиційного готельного сервісу.

Airbnb змінила підхід до короткострокової оренди житла, забезпечивши можливість не лише бронювати помешкання, але й з легкістю планувати подорожі з урахуванням індивідуальних потреб.

Недоліки платформи:

- оскільки житло здають різні люди з різним підходом до обслуговування, рівень комфорту та чистоти може суттєво відрізнятись. Користувачі можуть зіткнутися з невідповідністю опису або фото, що створює ризик неприємних сюрпризів при заселенні;

- іноді на Airbnb трапляються фейкові оголошення або шахрайські акаунти, особливо у популярних туристичних містах. Незважаючи на заходи безпеки, подекуди орендарі стикаються з обманом, коли оголошене житло не існує або в ньому відсутні обіцяні зручності;

Airbnb стягує комісію як з орендарів, так і з орендодавців, що може суттєво збільшити вартість. Окрім цього, багато оголошень мають додаткові збори, наприклад, за прибирання, що не завжди помітно при перегляді початкової ціни;

Правила скасування різняться від об'єкта до об'єкта, а у випадку конфліктів відшкодування може бути складним і тривалим процесом. Іноді Airbnb не повністю повертає кошти навіть у разі форс-мажорних обставин, що може бути розчаруванням для користувачів;

- у багатьох туристичних містах, таких як Барселона чи Венеція, Airbnb спричинила зростання вартості оренди житла та витіснення місцевих жителів, які не можуть дозволити собі платити вищу ринкову ціну. Це також може сприяти зменшенню доступного житла для довготривалої оренди;

- незважаючи на наявність служби підтримки, користувачі часто скаржаться на повільну реакцію та недостатню ефективність Airbnb у вирішенні проблем із бронюванням або конфліктів між орендодавцями та орендарями;

- бувають випадки, коли орендодавці без попередження встановлюють камери спостереження, що викликає занепокоєння щодо конфіденційності гостей. Навіть якщо камери дозволені правилами Airbnb, вони можуть знижувати відчуття приватності [5].

Якщо порівнювати Airbnb та Booking основні можливості дуже схожі. Також основні недоліки Airbnb притаманні майже усім сервісам з оренди житла.

1.3 Визначення проблем

У цьому розділі визначимо ключові проблеми, з якими стикаються користувачі при оренді житла на сучасних платформах, таких як Airbnb. Основні проблеми можна розділити на декілька категорій: безпека, відповідність очікуванням, прозорість транзакцій та вплив на місцеві громади. Розглянемо кожен з них детальніше.

На орендних платформах зустрічаються фейкові оголошення або недостовірні описи, що може призводити до неприємних ситуацій для орендарів. Недостатня перевірка профілів орендодавців іноді допускає шахраїв на платформу.

Випадки, коли в орендованому приміщенні встановлюють камери без попередження, ставлять під загрозу приватність гостей.

Невідповідність між описом і реальністю: Часто фотографії, опис і фактичний стан житла відрізняються, що викликає розчарування у гостей. Нестандартний рівень обслуговування та чистоти також знижує якість перебування.

Оскільки стандарти відрізняються, користувачі не можуть бути впевнені у рівні якості житла, що призводить до нерівномірного досвіду оренди.

Часто орендні платформи стягують комісію або додають приховані збори, наприклад, за прибирання. Це підвищує остаточну вартість і створює непорозуміння у користувачів.

Політика повернення коштів може бути нечіткою або складною для розуміння, а сама процедура часто займає багато часу та зусиль у разі конфліктних ситуацій.

У багатьох туристичних регіонах платформи на кшталт Airbnb сприяють підвищенню цін на житло, що створює труднощі для місцевого населення.

Внаслідок розвитку короткострокової оренди місцеві жителі часто втрачають можливість орендувати житло на тривалий термін через брак доступних варіантів та підвищення вартості.

У випадках, коли користувачі стикаються з проблемами, пов'язаними з бронюванням, вирішення питань через підтримку іноді займає багато часу або є малоефективним.

Ці проблеми підкреслюють необхідність створення безпечнішої та прозорішої платформи, що враховувала б потреби як орендарів, так і орендодавців, а також сприяла сталому розвитку місцевих спільнот [6].

1.4 Постановка задачі

Для розробки веб-додатку, що забезпечує безпечну оренду житла, необхідно сформулювати завдання, які охоплюють основні компоненти системи: фронтенд (HTML, CSS), бекенд (PHP), базу даних (MySQL). Основні етапи складаються з наступних пунктів:

1. Постановка задачі

Основні цілі:

- Забезпечити безпечну платформу для оренди житла.
- Створити зручний інтерфейс для користувачів.
- Реалізувати функціонал для перевірки орендодавців і орендарів.
- Підтримувати базу даних з можливістю зберігання та управління інформацією.

2. Функціональні вимоги

Головна сторінка:

- Вітрина доступних об'єктів нерухомості з фільтрами (локація, ціна, тип житла).
- Можливість швидкого пошуку.

Сторінка детального опису об'єкта:

- Інформація про житло;
- Кнопка для бронювання або запиту на оренду.

Реєстрація та вхід:

- Форма для реєстрації нового користувача (орендодавця/орендаря).
- Форма входу в обліковий запис.

Профіль користувача:

- Інформація про користувача.

Аутентифікація та авторизація:

- Реалізація системи входу та реєстрації користувачів.
- Захист облікових записів (шифрування паролів).

Управління оголошеннями:

- Функції для додавання, редагування та видалення оголошень.
- Можливість перевірки оголошень (модерація).

Бронювання:

- Обробка запитів на бронювання.
- Відправка підтверджень та сповіщень користувачам.

Структура бази даних:

Таблиця Users (користувачі): user_id, username (ім'я), email, password_hash (пароль), user_type (орендодавець/орендар), created_at (дата створення).

Таблиця apartments (оголошення): apartment_id (id), user_id (id користувача), description (опис), price (ціна), address (адреса), status (статус), created_at (дати доступності), image_path (зображення), category_id (категорія), city_id (місто).

Таблиця contracts (бронювання): id, apartment_id (id оголошення), user_id (id орендаря), created_at (дати бронювання), term_id (умови), status (статус).

Таблиця contract_terms (умови): id, term_text (текст умов).

Таблиця Categories (категорії): category_id, category_name (назва категорії).

Таблиця Cities (міста): city_id, city_name (назва міста).

3. Нефункціональні вимоги

- Забезпечення захисту даних користувачів, безпечна обробка фінансових транзакцій.

- Швидка реакція на запити користувачів, ефективна робота з базою даних.
- Можливість додавання нових функцій у майбутньому.

Технічне завдання на розробку представлене у додатку А.

1.5 Розробка концепцій рішення проблем

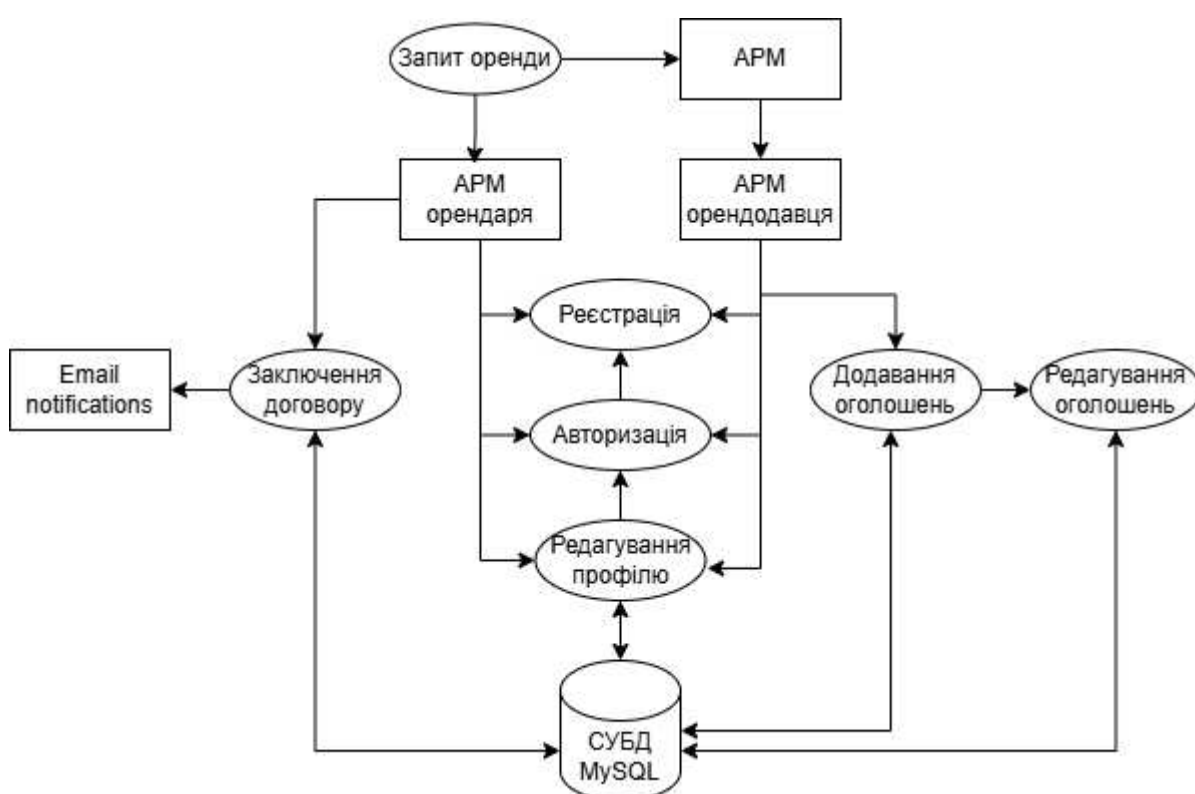


Рисунок 1.5 – Схема автоматизованої інформаційної системи оренди житла

Для ефективного вирішення проблем, пов'язаних із безпечною орендою житла, необхідно визначити концепції, які сприятимуть створенню надійної та зручної платформи. Необхідно врахувати наступні концепції:

Шифрування паролів за допомогою сучасних алгоритмів (наприклад, bcrypt), щоб запобігти їх компрометації.

Створення прозорої системи оцінювання, яка дозволяє користувачам залишати відгуки про орендодавців і орендарів.

Впровадження рейтингової системи на основі оцінок та коментарів, що допоможе новим користувачам приймати обґрунтовані рішення.

Створення зручного та зрозумілого інтерфейсу, що дозволяє користувачам легко орієнтуватися у функціоналі платформи.

Висновки за розділом I

У розділі проаналізовано ринок нерухомості та особливості винайму житла. Описано наявні сервіси (Airbnb, Booking), їх переваги й недоліки. Виявлено ключові проблеми: фейкові оголошення, низький рівень захищеності та не завжди зрозумілі умови винайму.

Для їх розв'язання поставлено завдання створити зручну та безпечну платформу з перевіркою користувачів, захистом даних і зрозумілим інтерфейсом. Це допоможе зробити винайм житла більш надійним для всіх учасників.

РОЗДІЛ II. РОЗРОБКА ПРОЕКТУ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ

2.1 Загальносистемні рішення

Використовується клієнт-серверна архітектура:

- HTML, CSS, JavaScript — для інтерфейсу користувача.
- PHP — для роботи на сервері.
- MySQL — для зберігання даних. Дані зберігаються у MySQL.

Щоб пошук був швидшим, використовуються індекси (наприклад, для полів `user_id` та `apartment_id`).

Дані не дублюються, таблиці пов'язані через ключі.

Паролі зберігаються у зашифрованому вигляді за допомогою функцій `password_hash()` та `password_verify()`.

Використовуються сесії для входу користувачів.

Щоб уникнути атак, застосовуються підготовлені запити (Prepared Statements) у PHP.

Дані, які вводять користувачі, очищаються функцією `htmlspecialchars()`.

2.1.1 Загальний опис системи

Розроблена система під назвою "Безпечна оренда житла" є інтерактивним веб-додатком, який забезпечує зручний та безпечний процес оренди житла для орендодавців та орендарів. Веб-додаток використовує HTML та CSS для створення інтерфейсу користувача, PHP для обробки серверних запитів, MySQL для зберігання даних, а також JavaScript для покращення взаємодії користувача з системою.

В системі передбачена реєстрація та аутентифікація користувачів, де кожен користувач має можливість зареєструватися як орендодавець чи орендар. Дані користувачів, такі як ім'я, email, пароль та тип користувача, зберігаються в базі да-

них, при цьому паролі зберігаються у вигляді хешів для забезпечення безпеки. Користувачі можуть додавати оголошення про оренду житла, заповнюючи опис, адресу, ціну, категорію, місто та завантажуючи фото. Уся ця інформація зберігається в таблиці оголошень, а категорії та міста житла організовані в окремі таблиці для зручності пошуку та фільтрації.

Орендарі можуть знайти житло за допомогою пошукової системи, яка дозволяє фільтрувати оголошення за ціною, категорією, містом та іншими параметрами. Пошук житла відбувається за допомогою інтерактивних фільтрів, що дає змогу користувачам швидко знаходити підходящі варіанти.

Після того як орендар знаходить підходяще житло, він може забронювати його. Бронювання відображається в таблиці контрактів, де зберігаються дані про дати бронювання, умови та статус контракту. Крім того, система передбачає можливість перегляду історії бронювань для кожного користувача.

Система умов оренди реалізована через таблицю, де зберігаються умови договору для кожного бронювання. В окремій таблиці зберігаються категорії житла, що дозволяє ефективно організувати та класифікувати оголошення. Для організації місця розташування житла є таблиця міст, що дозволяє легко здійснювати пошук за локацією.

Крім того, користувачі можуть залишати відгуки про орендоване житло, оцінюючи його якість. Це дозволяє формувати репутацію орендодавців, що додає довіри до них з боку потенційних орендарів.

Система розроблена з врахуванням високих стандартів безпеки. Дані користувачів захищені від несанкціонованого доступу, а паролі хешуються, що мінімізує ризик витоку особистої інформації. Інтерфейс додатку є інтуїтивно зрозумілим і зручним у використанні, а JavaScript додає динамічні елементи для зручності користувача.

Загалом, ця система дозволяє ефективно та безпечно орендувати житло, надаючи зручні інструменти як для орендодавців, так і для орендарів.

2.1.2 Розробка діаграми варіантів використання

Діаграма варіантів використання (Use Case Diagram) — це тип діаграми, що використовується в об'єктно-орієнтованому моделюванні, зокрема в методології UML (Unified Modeling Language). Вона призначена для опису взаємодії між користувачами (або іншими системами) і системою, яку проектують.

Основні елементи діаграми варіантів використання:

1. Актори — це користувачі або інші системи, що взаємодіють із розроблюваною системою. Актори можуть бути зовнішніми (наприклад, користувачі або інші системи) або внутрішніми (наприклад, різні модулі системи).
2. Варіанти використання (Use Cases) — це функції або дії, які система виконує у відповідь на запити акторів. Вони описують взаємодію між актором і системою для досягнення конкретної мети.
3. Зв'язки:
 - Association (Асоціація) — лінія, що з'єднує актора з варіантом використання, вказуючи на взаємодію.
 - Include (Включення) — відображає обов'язкові етапи в сценарії використання, які завжди виконуються разом з основним варіантом використання.
 - Extend (Розширення) — вказує на можливі додаткові або умовні дії, які можуть виконуватись у залежності від обставин.
 - Generalization (Успадкування) — коли один актор або варіант використання є підтипом іншого.

Цілі використання діаграми варіантів використання:

- Визначення основних функцій і ролей системи.
- Опис вимог і взаємодій користувачів із системою.
- Надання зрозумілої візуалізації для зацікавлених сторін (клієнтів, розробників, аналітиків) [6].

В рамках поставленої задачі розробимо наступну діаграму варіантів використання (рис. 2.1).

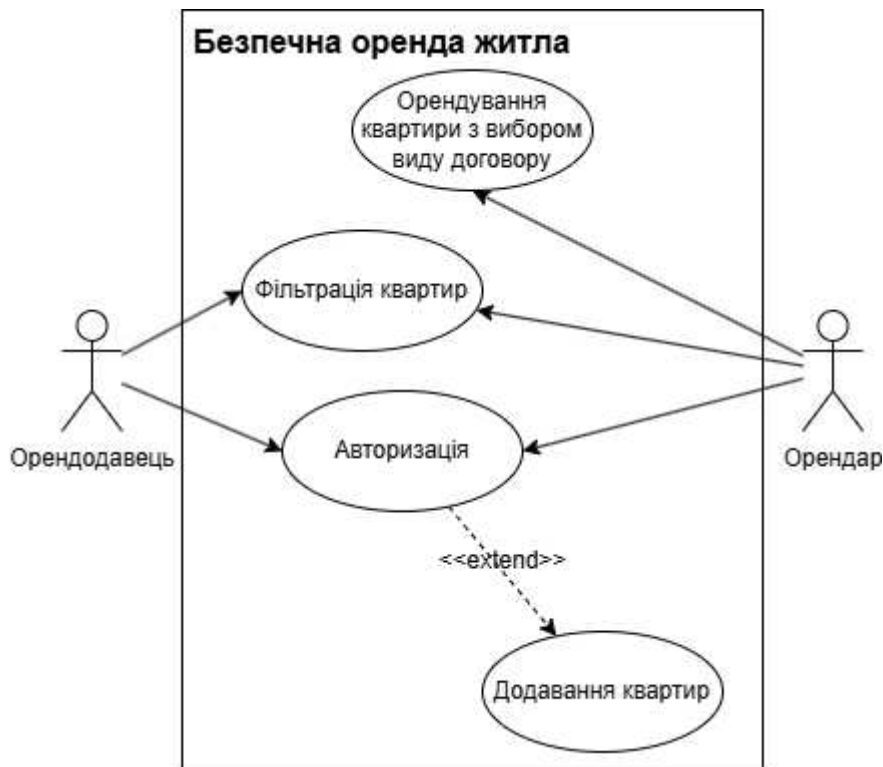


Рисунок 2.1 – Діаграма варіантів використання

Опишемо дану діаграму. Отже, ця діаграма відображає основні функції системи оренди квартир, а також взаємодію між двома типами користувачів: орендодавцями та орендарями.

Актори:

- Орендодавець - людина, яка здає квартиру в оренду.
- Орендар - людина, яка шукає квартиру для оренди.

Варіанти використання:

- Фільтрація квартир: Обидва актори можуть використовувати цю функцію для пошуку квартир, що відповідають їхнім критеріям.
- Авторизація: Обидва актори повинні авторизуватися, щоб отримати доступ до системи.
- Орендування квартири з вибором виду договору: Орендар може вибрати квартиру та тип договору оренди.
- Додавання квартир: Орендодавець може додати нову квартиру до бази даних.

Зв'язки:

- Обидва актори можуть використовувати функцію "Фільтрація квартир" та "Авторизація".
- Орендар може використовувати функцію "Орендування квартири з вибором виду договору".
- Орендодавець може використовувати функцію "Додавання квартир".
- Зв'язок <<extend>> між "Авторизація" та "Додавання квартир" означає, що орендодавець повинен бути авторизованим, щоб додати квартиру.

Діаграма чітко показує, що обидва актори мають доступ до фільтрації квартир та авторизації.

Орендар має можливість орендувати квартиру та вибрати тип договору.

Орендодавець має можливість додавати нові квартири до системи, але для цього йому потрібно бути авторизованим.

2.2 Рішення з інформаційного забезпечення

2.2.1 Визначення сутностей інформаційної підсистеми

Визначимо нефункціональні вимоги:

Система повинна обробляти не менше 1000 запитів на годину, а час відповіді на запит не повинен перевищувати 2 секунд. Для безпеки передбачене використання хешування паролів, наприклад, bcrypt, а також аутентифікація користувачів через JWT або OAuth. Реалізовані заходи захисту від SQL-ін'єкцій, XSS та CSRF-атак.

Система має підтримувати горизонтальне масштабування бази даних і використовувати кешування для швидшого доступу до даних. Гарантується доступність сервісу на рівні 99.9%, а для забезпечення безперервності роботи введена реплікація бази даних.

Інтерфейс повинен бути інтуїтивно зрозумілим для користувачів та адаптованим для мобільних пристроїв. Усі критичні події в системі фіксуються в логах, а

також передбачено моніторинг продуктивності та автоматичні сповіщення про помилки.

2.2.2 Концептуальна модель даних

На (рис. 2.2) зображена концептуальна модель даних, яка описує структуру інформаційної системи для керування оголошеннями про оренду чи продаж квартир. Вона включає основні сутності, їх атрибути та зв'язки між ними.

У центрі моделі розташована сутність **apartments**, яка є ключовою для системи. Вона містить такі атрибути: `apartment_id`, `image_path`, `created_at`, `area`, `bathrooms`, `bedrooms`, `price`, `address`, `description`. Квартири також зв'язані з кількома іншими сутностями:

1. **categories** - категорія, до якої належить квартира (наприклад, «новобудова», «вторинний ринок» тощо). Зв'язок позначено як *Фільтрується*. Атрибути: `category_id`, `category_name`.
2. **cities** - міста, в якому розташовані квартири. Зв'язок позначено як *Знаходиться*. Атрибути: `city_id`, `city_name`.
3. **users** - користувачі, які публікують оголошення про квартиру. Зв'язок позначено як *Викладається*. Атрибути: `user_id`, `username`, `password_hash`, `email`, `user_type`, `created_at`.
4. **contacts** – усі заявки, для генерації договорів. Зв'язок позначено як *Обирається* (мабуть, мається на увазі, що для кожної квартири можна згенерувати контактну заявку). Атрибути: `id`, `apartment_id`, `name`, `email`, `created_at`. Є додатковий зв'язок з **contract_terms** (типи контракту), що також *обирається*.
5. **contract_terms** - типи договору, з якими пов'язаний контакт. Атрибути: `id`, `term_text`.

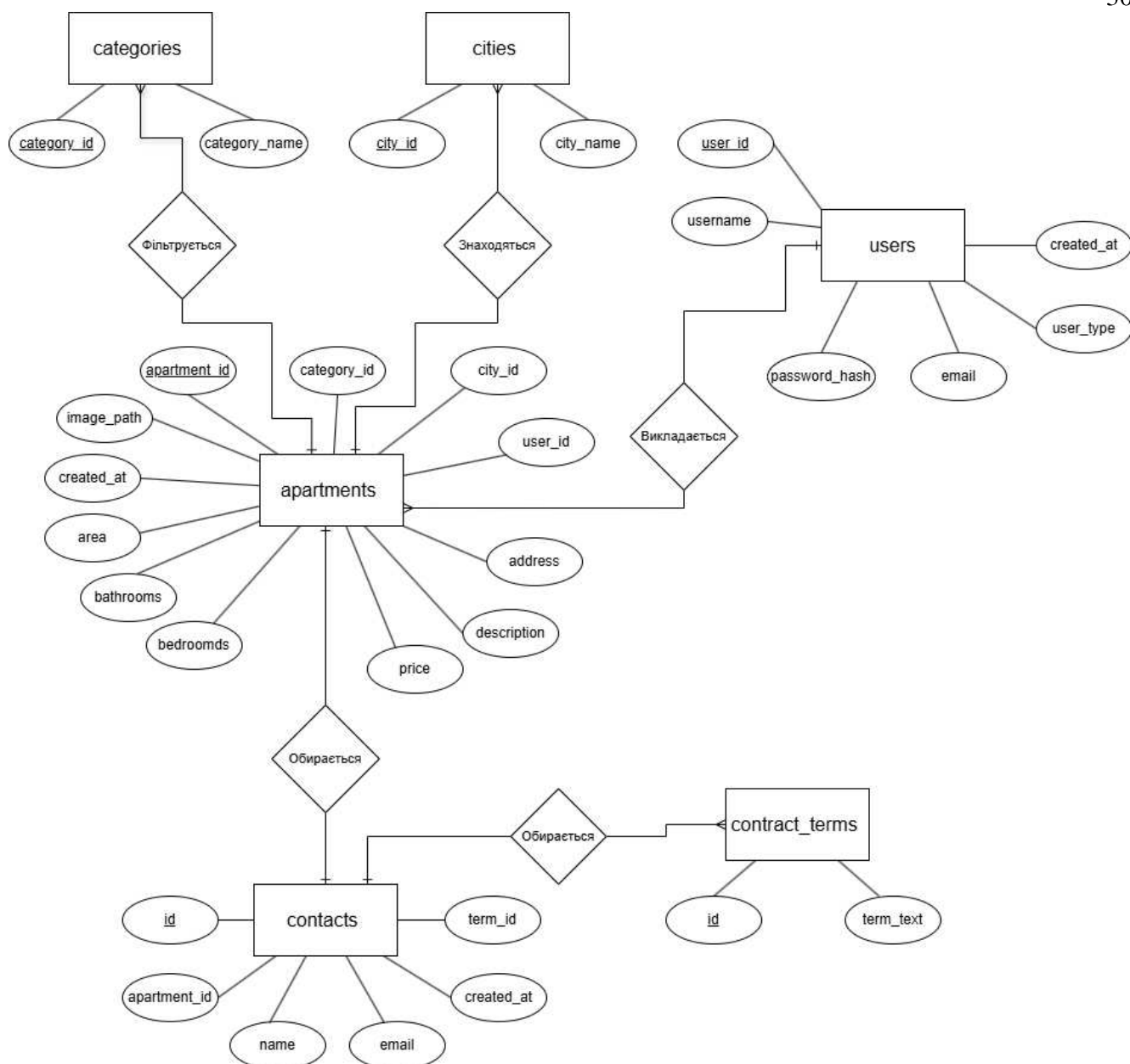


Рисунок 2.2 – Концептуальна модель даних

2.2.3 Розробка логічної моделі даних

Логічна структура даних — це спосіб організації і представлення даних таким чином, щоб вони були зрозумілі, зручні для обробки і доступу. Вона описує, як дані можуть бути логічно згруповані та пов'язані між собою. Логічна структура не залежить від фізичного розташування даних у пам'яті або способу їх зберігання.

База даних складається з наступних таблиць:

- apartments (рис. 2.3);
- categories (рис. 2.4);
- cities (рис. 2.5);
- contacts (рис. 2.6);
- contract_terms (рис. 2.7);
- users (рис. 2.8).

Структура таблиці		Вид відносин								
#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія	
1	apartment_id	int(11)			Ні	Немає		AUTO_INCREMENT		Більше
2	category_id	int(11)			Так	NULL				Більше
3	city_id	int(11)			Так	NULL				Більше
4	user_id	int(11)			Так	NULL				Більше
5	address	varchar(255)	utf8mb4_unicode_ci		Ні	Немає				Більше
6	description	text	utf8mb4_unicode_ci		Так	NULL				Більше
7	price	decimal(10,2)			Ні	Немає				Більше
8	bedrooms	int(11)			Ні	Немає				Більше
9	bathrooms	int(11)			Ні	Немає				Більше
10	area	decimal(10,2)			Ні	Немає				Більше
11	created_at	timestamp			Так	CURRENT_TIMESTAMP				Більше
12	image_path	varchar(255)	utf8mb4_unicode_ci		Так	NULL				Більше

Рисунок 2.3 – Структура таблиці `Apartments`

Переглянути Структура SQL Пошук Вставити Експорт Імпорт Привілеї Операції Тригери

Структура таблиці Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	category_id	int(11)		Ні	Немає		AUTO_INCREMENT	Більше
☐	2	category_name	varchar(100) utf8mb4_unicode_ci		Ні	Немає			Більше

Позначити все Вибрані:

Рисунок 2.4— Структура таблиці `Categories`

Сервер: 127.0.0.1:3306 / База даних: neryh / Таблиця: Cities

Переглянути Структура SQL Пошук Вставити Експорт Імпорт Привілеї Операції Тригери

Структура таблиці Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	city_id	int(11)		Ні	Немає		AUTO_INCREMENT	Більше
☐	2	city_name	varchar(100) utf8mb4_unicode_ci		Ні	Немає			Більше

Рисунок 2.5 – Структура таблиці `Cities`

Переглянути Структура SQL Пошук Вставити Експорт Імпорт Привілеї Операції Тригери

Структура таблиці Вид відносин

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1	id	int(11)		Ні	Немає		AUTO_INCREMENT	Більше
☐	2	apartment_id	int(11)		Ні	Немає			Більше
☐	3	name	varchar(255) utf8mb4_unicode_ci		Ні	Немає			Більше
☐	4	email	varchar(255) utf8mb4_unicode_ci		Ні	Немає			Більше
☐	5	created_at	timestamp		Так	CURRENT_TIMESTAMP			Більше
☐	6	term_id	int(11)		Ні	Немає			Більше

Позначити все Вибрані:

Рисунок 2.6 – Структура таблиці `contacts`

Структура таблиці		Вид відносин								
#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія	
☐ 1	id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT		Більше ▾
☐ 2	term_text	text	utf8mb4_unicode_ci		Ні	Немає				Більше ▾

☐ Позначити все Вибрані:

Рисунок 2.7 – Структура таблиці `contract\_terms`

Структура таблиці		Вид відносин								
#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія	
☐ 1	user_id 🔑	int(11)			Ні	Немає		AUTO_INCREMENT		Більше ▾
☐ 2	username 🔑	varchar(50)	utf8mb4_unicode_ci		Ні	Немає				Більше ▾
☐ 3	password_hash	varchar(255)	utf8mb4_unicode_ci		Ні	Немає				Більше ▾
☐ 4	email 🔑	varchar(100)	utf8mb4_unicode_ci		Ні	Немає				Більше ▾
☐ 5	user_type	enum('landlord', 'tenant')	utf8mb4_unicode_ci		Ні	Немає				Більше ▾
☐ 6	created_at	timestamp			Так	CURRENT_TIMESTAMP				Більше ▾

☐ Позначити все Вибрані:

Рисунок 2.8 – Структура таблиці `users`

Логічна модель представлена на рисунку 2.9.

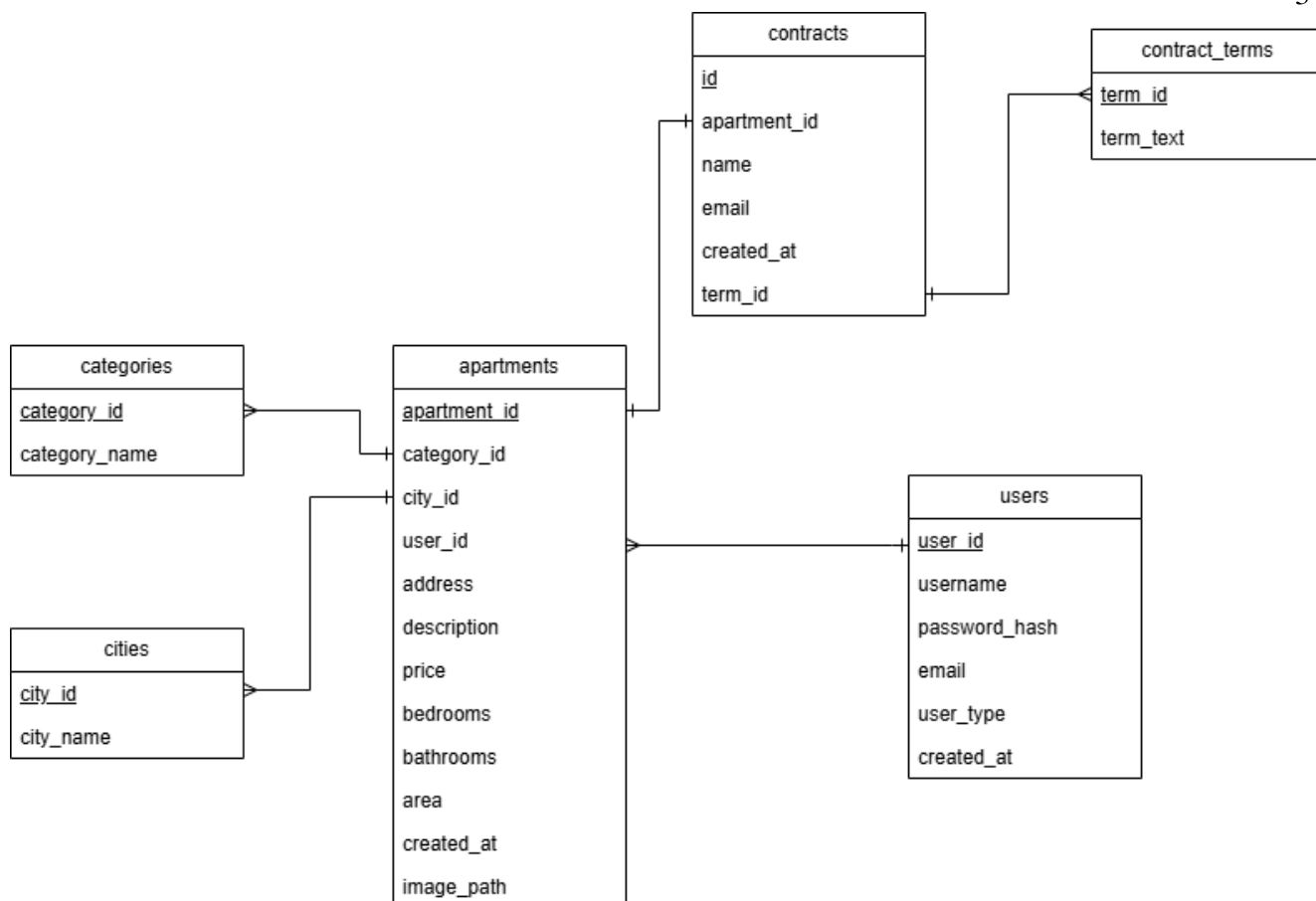


Рисунок 2.9 – Логічна модель бази даних

2.2.4 Розробка фізичної моделі даних

Фізична модель даних — це представлення даних у термінах фізичного зберігання та організації на комп'ютерних носіях. Вона описує, як дані фактично зберігаються в пам'яті комп'ютера або на інших фізичних носіях, таких як жорсткі диски чи сервери, з урахуванням обмежень апаратного забезпечення та ефективності доступу [8].

Основні аспекти фізичної моделі даних:

1. Фізичне зберігання — визначає, де саме будуть зберігатися дані: на жорсткому диску, в SSD, на мережевих носіях тощо. Фізичне зберігання включає такі елементи, як розмір блоків даних, структури файлів, механізми доступу до даних тощо.

2. Індеси — це спеціальні структури даних, які використовуються для оптимізації доступу до даних, що зберігаються в таблицях. Наприклад, при зберіганні даних у реляційних базах даних часто використовуються індеси для прискорення пошуку. Фізична модель описує, як індеси зберігаються на носіях.

3. Нормалізація та денормалізація — хоча нормалізація є частиною логічної моделі, фізична модель може включати денормалізацію для поліпшення продуктивності. Денормалізація — це процес створення додаткових копій даних або їх спрощення для прискорення операцій зчитування.

4. Шарування та кластеризація — це способи фізичного розподілу даних. Наприклад, в разі великих обсягів даних можуть використовуватися різні методи розподілу (шарування) даних між кількома серверами або дисками для забезпечення ефективності роботи бази даних.

5. Розміри та типи даних — в фізичній моделі визначаються точні типи даних, які будуть використовуватися для зберігання значень (наприклад, INT, VARCHAR, DATE). Це важливо для оптимізації пам'яті та ресурсів, оскільки різні типи даних займають різний обсяг пам'яті.

6. Методи доступу до даних — включають стратегії пошуку та організації даних, такі як послідовний доступ, індексований доступ, хешування тощо. Це визначає, як будуть оброблятися запити до бази даних.

Фізична модель даних спрямована на досягнення найкращої продуктивності і ефективності зберігання, забезпечуючи зручний доступ до даних при мінімальних витратах на ресурси. Вона залежить від апаратного середовища і може змінюватися в залежності від технічних обмежень або вимог до масштабованості системи [9].

У контексті розробленої системи фізичну модель бази даних наведено на рисунку 2.9.

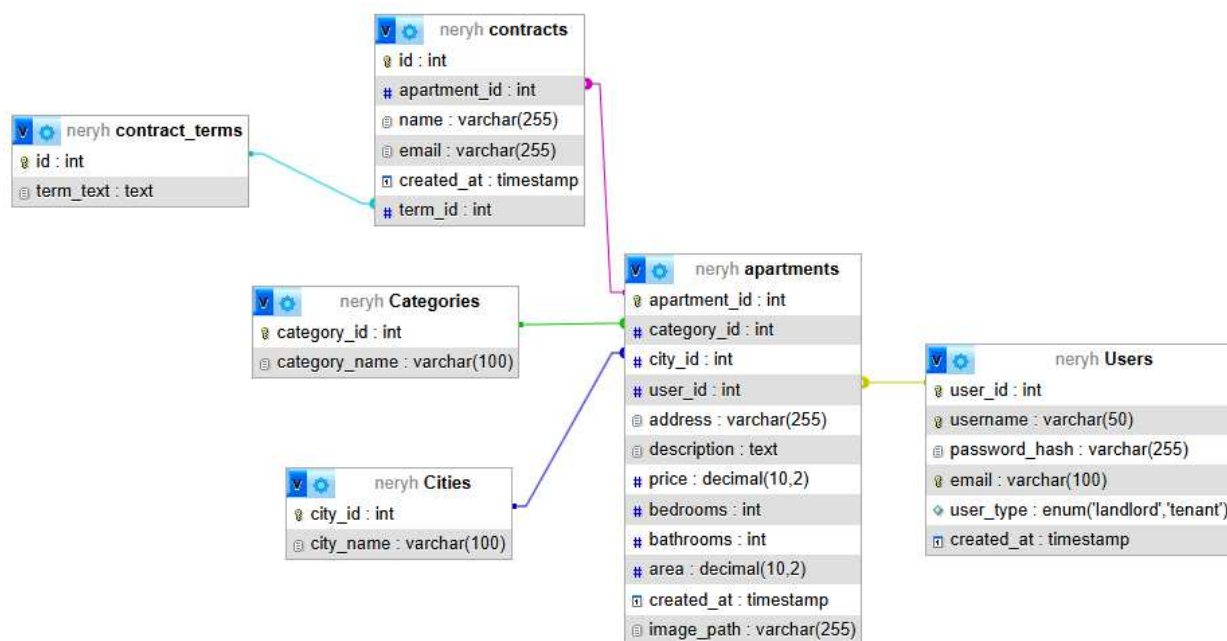


Рисунок 2.9 – Фізична модель бази даних

2.3 Рішення з технічного забезпечення

2.3.1 Вимоги до технічного забезпечення

Щоб встановити Open Server, потрібно, щоб версія операційної системи була Windows 7 або новішою (Windows 10 чи 11), з підтримкою як 32-розрядної, так і 64-розрядної версії. Також для коректної роботи Open Server можуть знадобитися деякі додаткові програмні компоненти, такі як Microsoft Visual C++ Redistributable і Microsoft .NET Framework. Зазвичай, якщо вони не встановлені в системі, програма запропонує їх інсталиувати під час запуску.

Важливо мати права адміністратора для інсталяції та налаштування Open Server, оскільки програма може потребувати змін у системі, таких як відкриття портів. Для доступу до локальних сайтів через Open Server достатньо будь-якого сучасного браузера, як Google Chrome, Mozilla Firefox або Microsoft Edge.

Якщо на комп'ютері встановлений антивірус або фаєрвол, переконайтеся, що вони не блокують порти, які використовує Open Server, наприклад, порт 80 для HTTP і 3306 для MySQL. Всі ці вимоги забезпечать правильну роботу Open Server на комп'ютері [10].

Мінімальні технічні характеристики для програмного забезпечення, які будуть працювати з даним програмним забезпеченням мають бути такі:

1. Дискового простору 15 Гб;
2. 2 Гб оперативної пам'яті;
3. Двух ядерний процесор з частотою 1.5 ГГц;

Рекомендовані характеристики:

1. Дискового простору 30 Гб;
2. 4 Гб оперативної пам'яті;
3. Два й більше ядерний процесор з частотою не менше ніж 2.0 ГГц;

2.3.2 Склад технічного забезпечення

Склад технічного забезпечення для системи чи проекту включає апаратне, програмне та мережеве забезпечення, а також механізми безпеки і резервування. Апаратне забезпечення складається з обчислювальних ресурсів, таких як процесори, оперативна пам'ять, жорсткі диски або SSD, що використовуються для зберігання даних, а також мережеві адаптери, що забезпечують з'єднання між компонентами системи. Також необхідно враховувати інтерфейси введення/виведення для підключення периферійних пристроїв.

Програмне забезпечення включає операційну систему, яка забезпечує управління апаратними ресурсами, а також системи управління базами даних для зберігання і обробки даних. Важливими компонентами є веб-сервери для обробки запитів від клієнтів, інструменти для моніторингу для спостереження за станом системи, а також системи управління версіями для відстеження змін в коді.

Мережеве забезпечення включає мережеве обладнання, таке як роутери і комутатори, які дозволяють забезпечити зв'язок між усіма елементами системи. Мережеві протоколи визначають порядок комунікацій між компонентами. Для захисту даних та системи використовуються механізми шифрування, антивірусне програмне забезпечення та фаєрволи для блокування несанкціонованого доступу.

Резервне та відновлювальне забезпечення включає системи резервного копіювання, які дозволяють зберігати важливі дані для відновлення їх у разі збоїв, а також системи для відновлення працездатності системи після аварійних ситуацій. Всі ці компоненти працюють разом, забезпечуючи ефективну та безпечну роботу системи або проекту.

2.4 Рішення з програмного забезпечення

2.4.1 Розробка схеми структури інформаційної підсистеми

Структурна схема відображає основні розділи та функціональність веб-сайту, ймовірно, пов'язаного з орендою або продажем нерухомості. Вона показує, як користувач може переміщатися між різними сторінками та виконувати певні дії.

На рис. 2.9 наведено структурну схему інформаційної підсистеми.

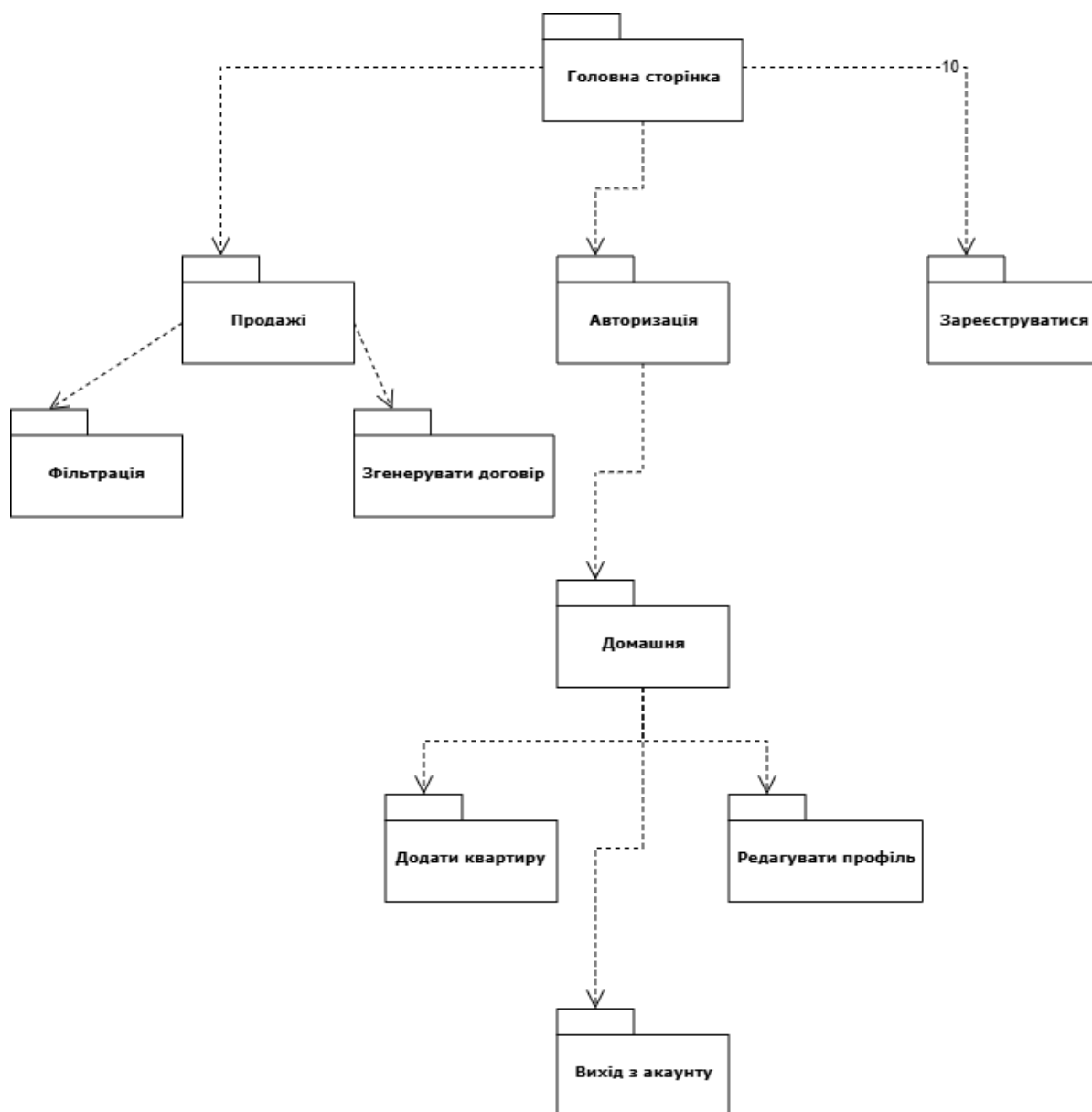


Рисунок 2.9 – Структурна схема інформаційної підсистеми

Основні елементи:

Головна сторінка - це центральний елемент, з якого можна перейти до інших розділів.

Продажі - розділ, де користувачі можуть переглядати оголошення про продаж нерухомості.

Фільтрація - Функція для звуження результатів пошуку за певними критеріями.

Заклучити договір - Можливість оформити угоду про купівлю-продаж.

Авторизація - Сторінка для входу в особистий кабінет.

Зареєструватися - сторінка для створення нового облікового запису.

Домашня - особистий кабінет користувача після авторизації.

Додати квартиру - функція для орендодавців, щоб додати нове оголошення.

Редагувати профіль - можливість змінити особисті дані.

Вихід - вихід з особистого кабінету.

Роль "Як орендодавець". На схемі є позначення "Як орендодавець" і людина, яка позначена як користувач. Таким чином, червона стрілка, яка виходить від орендодавця, веде до "Додати квартиру", що показує, що це дія, яку може зробити орендодавець.

Логіка роботи:

1. Користувач потрапляє на головну сторінку.
2. Він може вибрати розділ "Продажі", "Авторизація" або "Зареєструватися".
3. Якщо користувач вибирає "Продажі", він може використовувати фільтрацію або заключити договір.
4. Якщо користувач вибирає "Авторизація", він потрапляє на сторінку входу, а потім на "Домашню" сторінку.
5. На "Домашній" сторінці користувач може додати квартиру (якщо він орендодавець), редагувати профіль або вийти.

2.4.2 Розробка діаграми компонентів

Схема на рис.2.10 відображає основні файли РНР, які складають веб-додаток, а також їхні взаємозв'язки. Це показує, як користувач може переміщатися між різними сторінками та як дані обробляються.

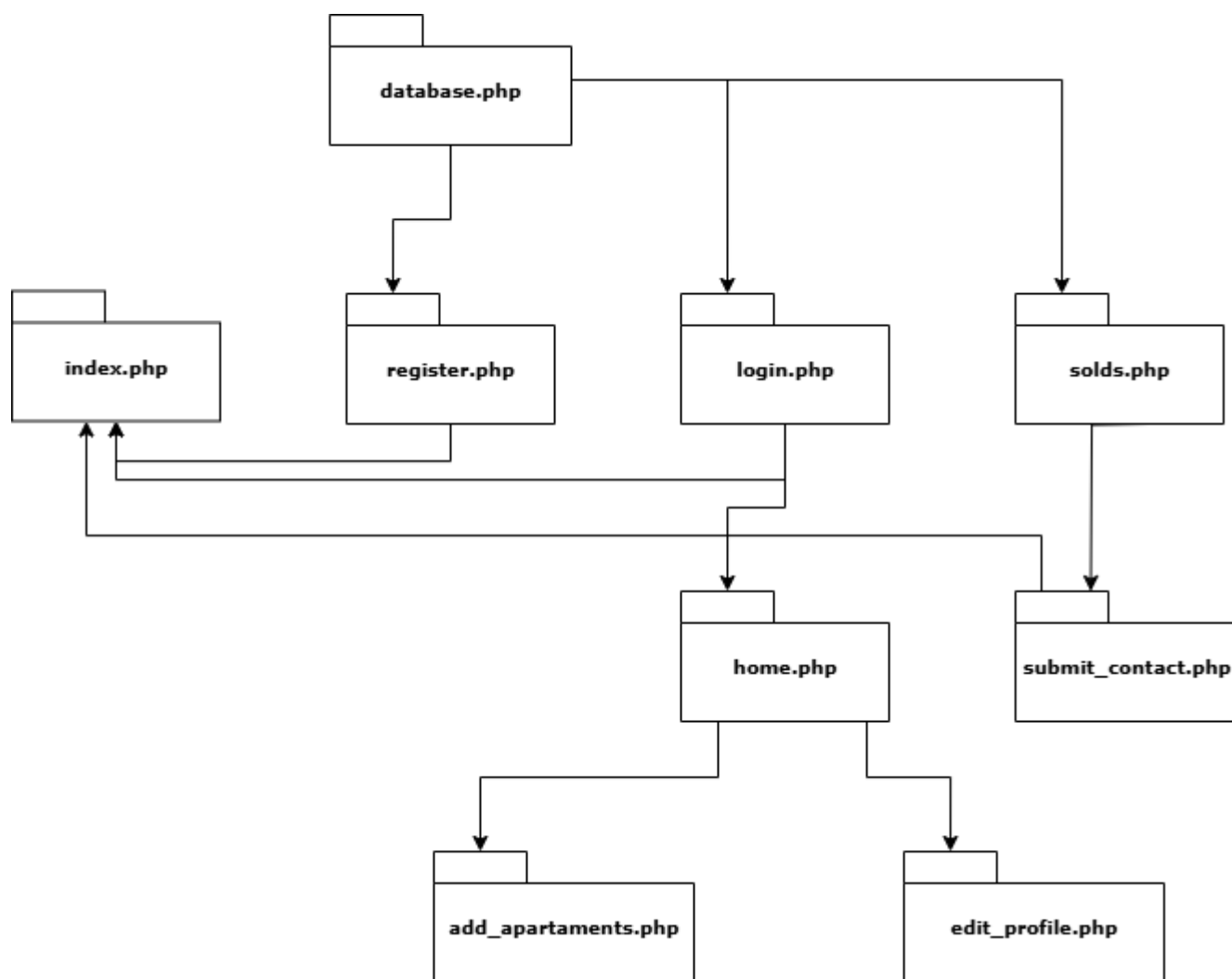


Рисунок 2.10 – Діаграма компонентів

Основні компоненти:

`database.php` - цей файл містить код для підключення до бази даних та виконання запитів. Він є центральним елементом, до якого звертаються інші файли для доступу до даних.

`index.php` - головна сторінка веб-сайту. Вона може містити посилання на інші сторінки або відображати основний контент.

`register.php` - сторінка для реєстрації нового користувача. Вона, ймовірно, містить форму для введення даних та код для збереження їх у базі даних.

`login.php` - сторінка для входу користувача. Вона перевіряє введені дані та дозволяє користувачеві увійти в систему.

`solds.php` - сторінка, яка відображає інформацію про продажі. Вона може отримувати дані з бази даних через `database.php`.

`home.php` - домашня сторінка користувача після успішного входу. Вона може містити посилання на інші сторінки, доступні лише авторизованим користувачам.

`submit_contract.php` - сторінка для подання договору. Вона може містити форму для введення даних та код для збереження їх у базі даних.

`add_apartments.php` - сторінка для додавання оголошення про квартиру. Вона доступна лише для певних користувачів.

`edit_profile.php` - сторінка для редагування профілю користувача. Вона дозволяє користувачеві змінювати свої особисті дані.

2.4.3 Розробка діаграми переходів станів

Діаграма станів відображає життєвий цикл об'єкта, починаючи з моменту його створення і проходячи через усі можливі стани, в яких він може перебувати, аж до остаточного видалення та припинення існування.

Кожен стан на діаграмі представлений у вигляді прямокутника із заокругленими кутами. Кількість таких станів не обмежена, а також можливий їхній поділ на підстани для деталізації процесів.

Перехід між станами позначається односпрямованою стрілкою, що вказує напрямок зміни. Ця стрілка може супроводжуватися назвою події, яка викликає зміну стану, а також додатковими умовами чи діями, пов'язаними з переходом.

Початок і завершення життєвого циклу об'єкта позначаються спеціальними символами: перший вказує на момент появи об'єкта, а другий – на його ліквідацію та завершення життєвого шляху [11].

Для даної системи маємо дві діаграми переходів станів. На рис. 2.11 наведено діаграму для орендаря, а на рис. 2.12 наведено діаграму для орендодавця.

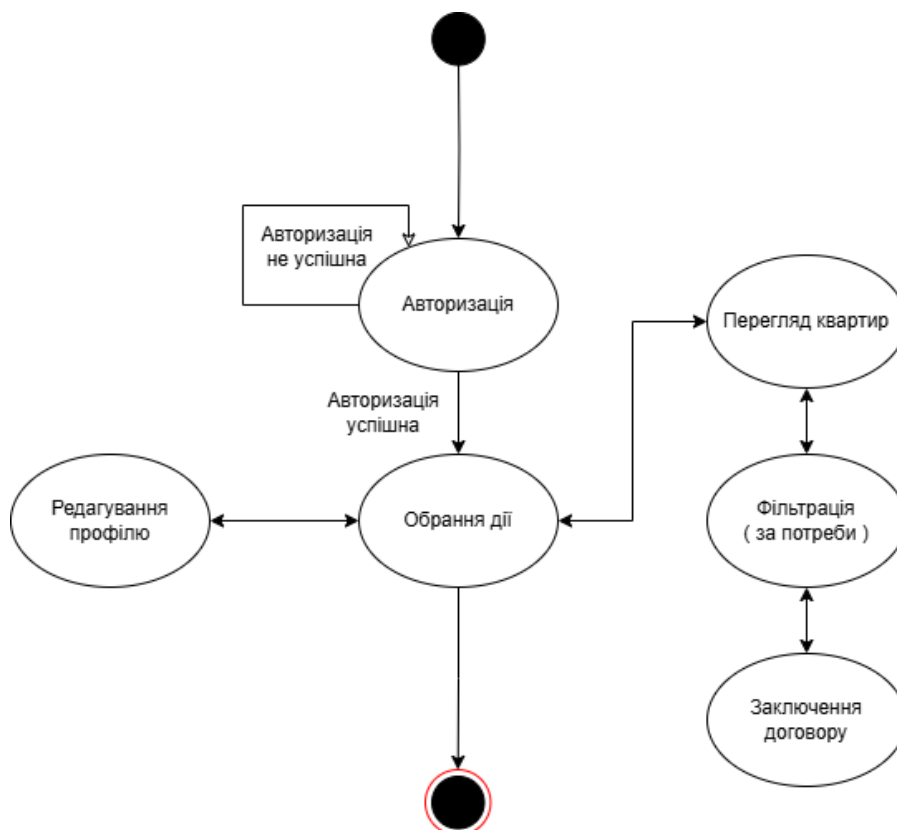


Рисунок 2.11 – Діаграма для орендаря

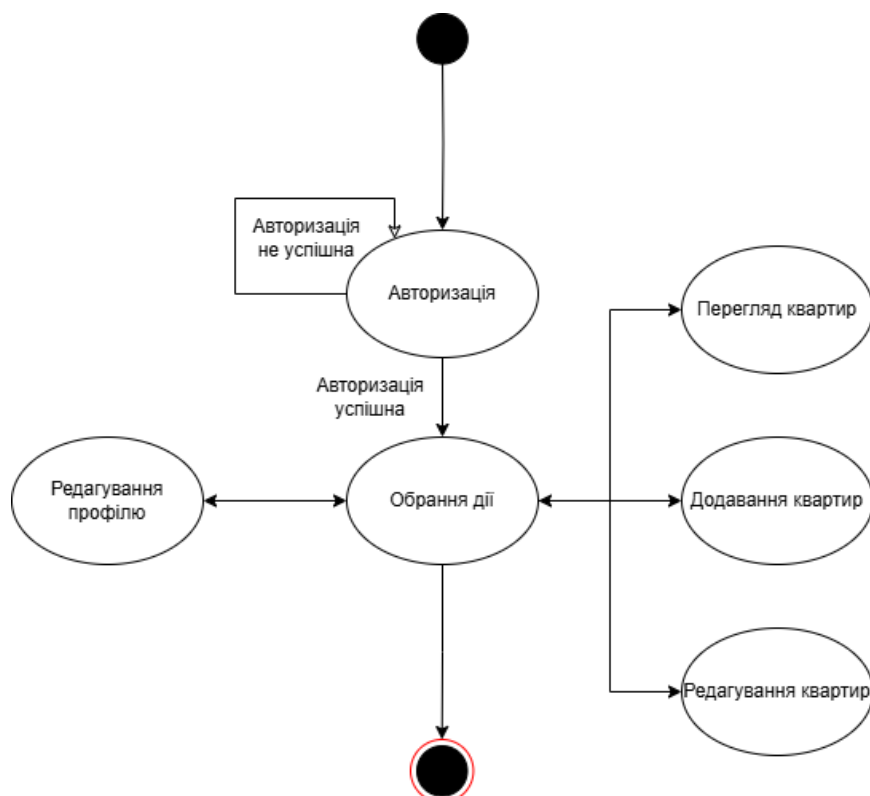


Рисунок 2.12 – Діаграма для орендодавця

У процесі розробки діаграми станів було проаналізовано життєвий цикл об'єктів системи, що дозволило чітко визначити можливі стани та умови їх переходів. Візуалізація цих змін у вигляді діаграми забезпечує структуроване представлення процесів, що відбуваються в системі.

Діаграми переходів станів для орендаря та орендодавця демонструють логіку зміни станів об'єктів у залежності від виконуваних дій та зовнішніх умов. Використання поділу на підстани дає змогу детальніше описати механізми взаємодії в межах системи, що сприяє її ефективному функціонуванню.

Таким чином, розроблені діаграми станів відіграють важливу роль у моделюванні процесів та забезпечують наочне відображення життєвого циклу об'єктів, що сприяє підвищенню розуміння та оптимізації роботи інформаційної системи.

2.4.4 Розробка проекту інтерфейсу інформаційної підсистеми

Інтерфейс розроблюваної інформаційної системи має бути мінімалістичний. Інтерфейс виконано за допомогою використання таблиці стилів CSS та мови програмування HTML. Остаточний інтерфейс наведено на рис. 2.13-2.15.

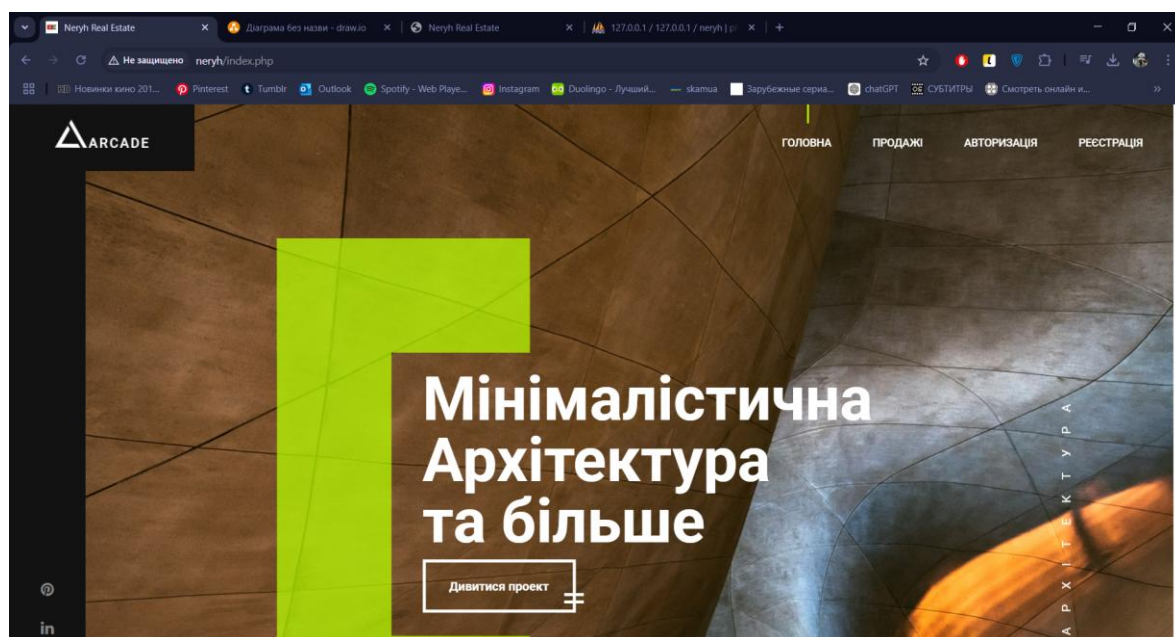


Рисунок 2.13 – Інтерфейс сторінки «Головна»

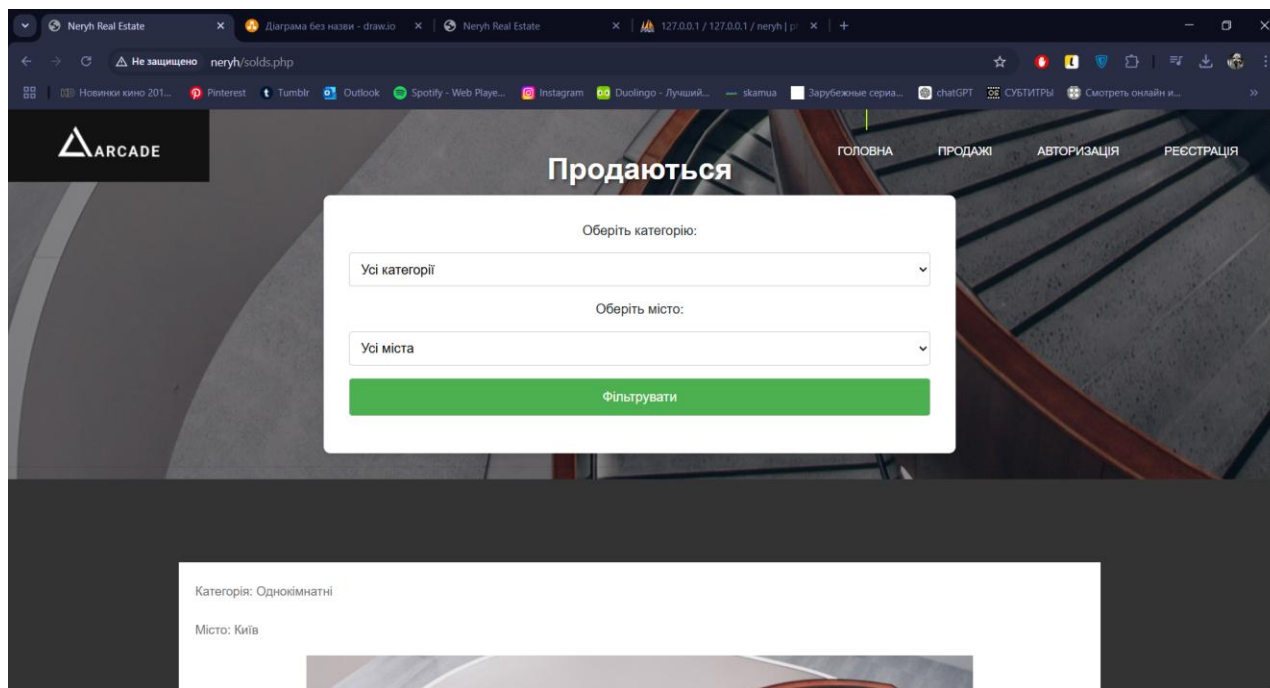


Рисунок 2.14 – Інтерфейс сторінки «Продажі»

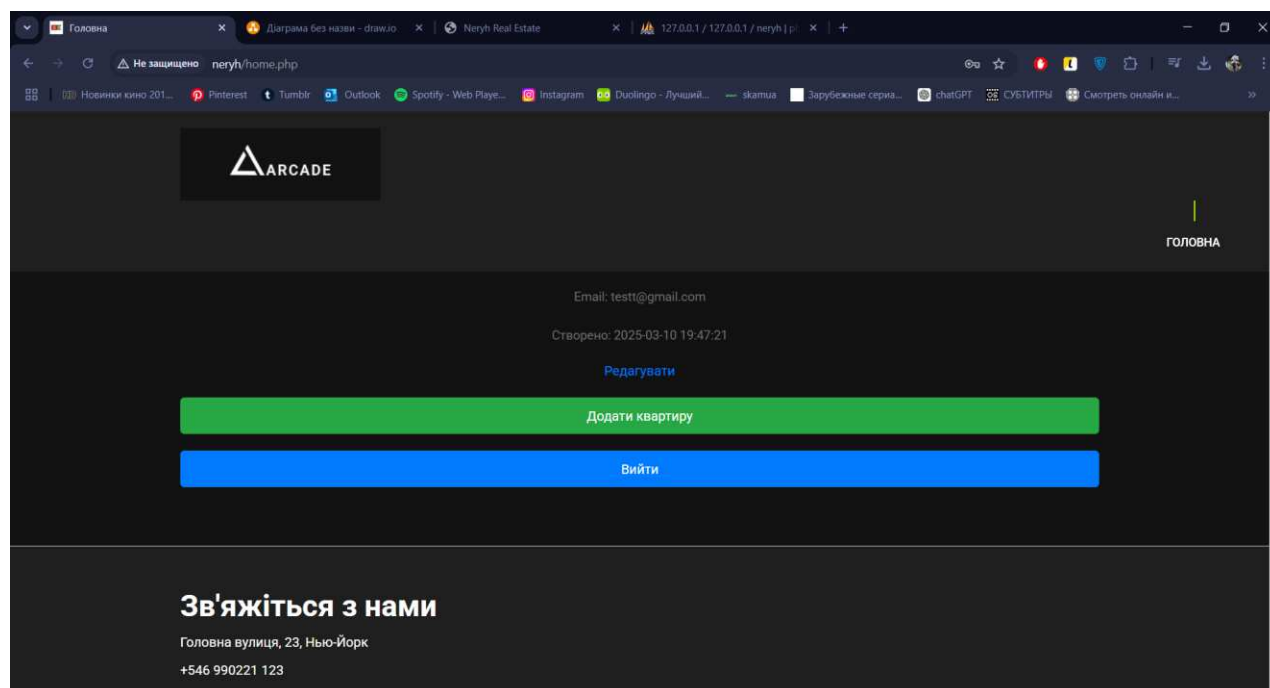


Рисунок 2.15 – Інтерфейс сторінки «Особистий кабінет»

У процесі розробки інтерфейсу інформаційної системи було дотримано принципів мінімалізму, що забезпечує зручність використання та інтуїтивну навігацію для користувачів. Візуальне оформлення реалізовано за допомогою HTML та CSS, що дозволило створити сучасний та адаптивний дизайн.

Представлені макети сторінок «Головна», «Продажі» та «Особистий кабінет» демонструють функціональність інтерфейсу, його логічну структуру та відповідність вимогам користувачів. Використання стилів CSS дало змогу оптимізувати відображення контенту, забезпечуючи гармонійне поєднання естетики та зручності взаємодії.

Таким чином, розроблений інтерфейс відповідає основним критеріям юзабіліті, є візуально привабливим і сприяє ефективній роботі з інформаційною системою.

2.4.5 Розробка діаграми розгортання підсистеми

Діаграма розгортання — це вид діаграми в UML, який відображає фізичну архітектуру програмного забезпечення, зокрема, як програмні компоненти розподіляються на апаратних пристроях (вузлах) та взаємодіють між собою.

Ця діаграма використовується для моделювання інфраструктури системи, демонструючи сервери, клієнтські пристрої, бази даних та інші елементи, а також канали зв'язку між ними. Основними складовими діаграми є вузли (сервери, комп'ютери, пристрої), компоненти (модулі програмного забезпечення) та з'єднання (мережеві канали, протоколи зв'язку).

Діаграма розгортання дозволяє зрозуміти, як саме працює система в реальних умовах, допомагає оптимізувати розподіл навантаження та забезпечити коректну роботу програмного забезпечення на різних платформах.

Діаграму розгортання підсистеми наведено на рис. 2.16.

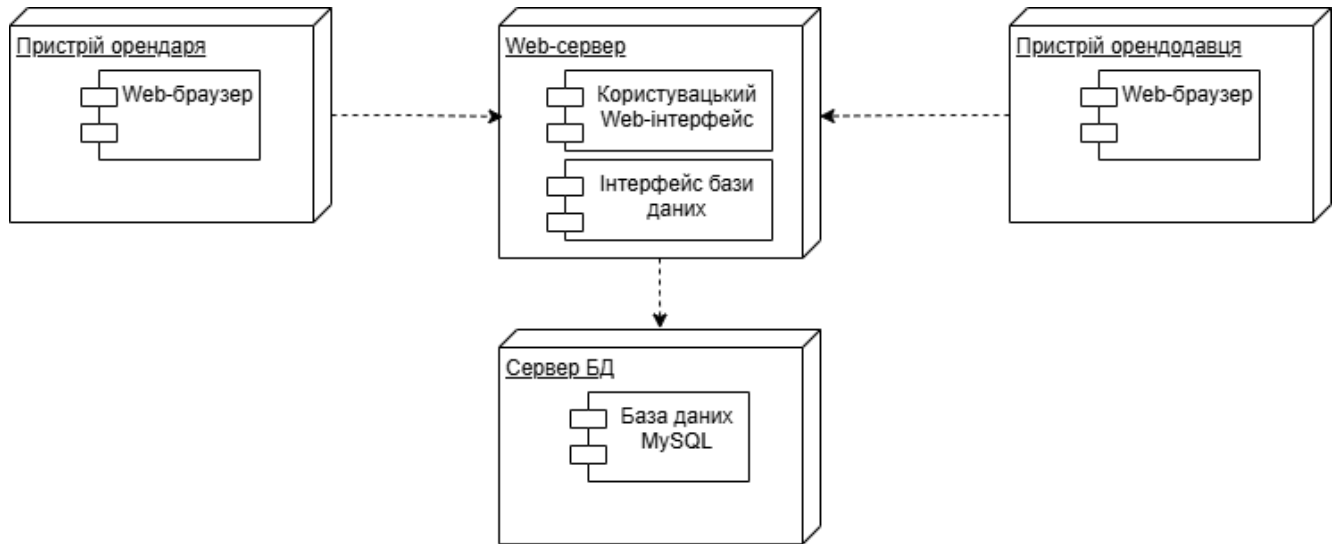


Рисунок 2.16 – Діаграма розгортання

Таким чином, можна зробити висновок, що взаємодія між орендарем, орендодавцем та сервером ґрунтується на клієнт-серверній архітектурі. Сервер виконує ключову роль у зберіганні, обробці та наданні даних, а користувачі взаємодіють із ним через клієнтську частину системи.

Орендар і орендодавець є клієнтами сервера, оскільки всі операції, пов'язані з переглядом, додаванням, редагуванням і бронюванням житла, здійснюються шляхом запитів до сервера. Сервер обробляє ці запити, взаємодіє з базою даних та надсилає відповідні відповіді клієнтам.

Орендарі отримують доступ до списку доступного житла, можуть переглядати його характеристики, здійснювати пошук за параметрами, бронювати обрані варіанти та керувати своїми бронюваннями. Усі ці дії передбачають взаємодію з сервером, який зберігає інформацію про користувачів, житло та статус бронювань.

Орендодавці, у свою чергу, використовують сервер для управління своїми об'єктами нерухомості: додають нові оголошення, редагують інформацію про житло, змінюють статус доступності та відстежують бронювання. Всі внесені зміни відправляються на сервер, де вони зберігаються та оновлюються в базі даних.

Завдяки серверній архітектурі забезпечується цілісність даних, їхня актуальність і безпека. Сервер виконує контроль доступу, забезпечує швидку обробку запитів та гарантує коректність інформації, що відображається клієнтам.

Таким чином, система оренди житла побудована на клієнт-серверній моделі, де сервер виступає центральним елементом для управління даними, а орендарі та орендодавці є його клієнтами, які через інтерфейс взаємодіють із серверною частиною, отримуючи необхідну інформацію та виконуючи операції в системі.

Висновки за розділом II

У другому розділі подано розробку інформаційної підсистеми веб-додатку "Безпечна оренда житла", збудованого на клієнт-серверній архітектурі з використанням HTML, CSS, JavaScript, PHP та MySQL. Система гарантує безпечну реєстрацію, фільтрацію оголошень, бронювання житла та керування користувацькими профілями. Докладно описано моделі даних, варіанти застосування, діаграми станів і розгортання, а також інтерфейс, що відповідає принципам зручності та мінімалізму. Враховано вимоги до продуктивності, безпеки й масштабованості, що забезпечують ефективну роботу системи.

РОЗДІЛ III. РЕАЛІЗАЦІЯ ПРОЕКТУ ПІДСИСТЕМИ

3.1 Аргументація вибору засобів розробки

При розробці веб-додатка важливо використовувати технології, які забезпечують зручність у розробці, ефективність у роботі та широкий спектр можливостей для реалізації функціоналу. Поєднання клієнтських і серверних рішень дозволяє створювати динамічні та інтерактивні веб-сторінки, які легко адаптуються під потреби користувачів.

Завдяки технологіям для створення інтерфейсу можна забезпечити коректне відображення контенту, стильне оформлення та зручну взаємодію з елементами сторінки. Це дає змогу створювати привабливий і зрозумілий інтерфейс для користувачів.

На серверному рівні використовуються засоби, що забезпечують обробку запитів, управління даними та зв'язок з базами. Це дає можливість реалізувати складну логіку, обробляти великі обсяги інформації та динамічно змінювати контент сторінки.

Для роботи з базами даних обираються інструменти, які дозволяють ефективно зберігати, структурувати та швидко отримувати необхідну інформацію. Це важливо для надійної та стабільної роботи веб-додатка.

Окрему увагу приділяють інтерактивності, яка забезпечує швидкий відгук інтерфейсу без необхідності перезавантаження сторінки. Це підвищує зручність використання та покращує загальний досвід взаємодії з веб-додатком.

Також важливим аспектом у виборі засобів розробки є їхня сумісність і підтримка сучасних стандартів. Це гарантує коректну роботу веб-додатка на різних пристроях та браузерах, а також полегшує його подальше розширення та оновлення.

Безпека відіграє ключову роль, тому використовуються перевірені технології, які дозволяють захистити дані користувачів, запобігти потенційним загрозам та забезпечити стабільну роботу системи.

Завдяки широким можливостям налаштування та оптимізації можна досягти високої продуктивності навіть при великій кількості користувачів. Гнучкість у розробці дозволяє легко змінювати функціонал, адаптувати його під нові вимоги та інтегрувати додаткові сервіси.

Важливим є й активне ком'юніті та наявність великої кількості документації. Це спрощує процес розробки, дає змогу швидко знаходити рішення для можливих проблем та використовувати перевірені практики для покращення роботи веб-додатка.

Таким чином, використання відповідних технологій у поєднанні з правильним підходом до їх інтеграції дозволяє створити ефективний, надійний і зручний веб-додаток, який відповідає сучасним вимогам як з боку розробників, так і користувачів.

Крім того, правильний вибір засобів розробки забезпечує масштабованість системи, що є важливим для її подальшого розвитку. Завдяки цьому веб-додаток можна легко адаптувати до зростаючого навантаження, збільшення кількості користувачів та додавання нових функцій без значних витрат часу та ресурсів.

Оптимізація коду та ефективна взаємодія між клієнтською та серверною частинами дозволяють мінімізувати затримки під час обробки запитів, що позитивно впливає на швидкість роботи веб-додатка. Це особливо важливо для покращення користувацького досвіду та підвищення загальної продуктивності системи.

Також варто зазначити, що вибрані технології дозволяють реалізувати адаптивний дизайн, що гарантує зручність використання веб-додатка на різних пристроях. Це підвищує його доступність для широкої аудиторії та забезпечує відповідність сучасним вимогам веб-розробки.

Завдяки можливостям інтеграції з іншими сервісами, такими як платіжні системи, сторонні API або сервіси аналітики, можна розширити функціональність веб-додатка та зробити його більш корисним і ефективним для користувачів [12].

У результаті використання правильно підібраних технологій не лише спрощує процес розробки, а й дозволяє створювати продуктивні, безпечні та гнучкі рішення, що відповідають потребам сучасного цифрового середовища.

3.2 Розробка структури програмного забезпечення

Розробка структури програмного забезпечення є важливим етапом у процесі створення системи. Правильна архітектура визначає не лише функціональність, але й ефективність, масштабованість, безпеку та подальшу підтримку програмного продукту. Успішне проектування структури потребує детального аналізу вимог до системи, врахування можливостей для розширення та організації структури таким чином, щоб вона була зручною і зрозумілою для команди розробників.

Першим етапом у розробці структури є детальний аналіз вимог до програмного забезпечення. Важливо визначити, які функціональні можливості повинна реалізовувати система, як вона буде взаємодіяти з користувачем і з іншими компонентами. Також потрібно враховувати вимоги до масштабування системи, можливості її інтеграції з іншими програмами і зовнішніми сервісами, а також важливі аспекти безпеки.

Наступним кроком є вибір архітектурного стилю, який визначить, як будуть організовані компоненти програмного забезпечення та як вони взаємодіятимуть між собою. Наприклад, можна вибрати монолітну архітектуру, коли всі компоненти працюють в єдиному блоці, або мікросервісну архітектуру, де система розподіляється на кілька незалежних сервісів, що дозволяє гнучко масштабувати окремі частини програми. Крім того, можна використовувати клієнт-серверну архітектуру, де клієнтська і серверна частини функціонують окремо.

Після цього важливо спроектувати структуру на рівні модулів та компонентів. Кожен компонент має відповідати за певну частину системи, що дозволяє організувати програму так, щоб її різні частини можна було окремо розробляти та тестувати. Наприклад, це можуть бути окремі модулі для користувацького інтер-

фейсу, для бізнес-логіки, для роботи з базою даних або для взаємодії з іншими системами.

Проектування бази даних є ще одним ключовим етапом, оскільки від структури даних залежить ефективність роботи програми. Необхідно продумати схему таблиць, зв'язки між ними та методи роботи з даними. Це дозволяє забезпечити не тільки ефективну обробку запитів, але й гарантує цілісність і безпеку збережених даних.

Вибір технологій для реалізації кожного з компонентів також має вирішальне значення. Це включає вибір мов програмування для серверної і клієнтської частини, фреймворків, бібліотек і систем управління базами даних. Також важливо враховувати сумісність вибраних технологій, оскільки вони повинні безперешкодно взаємодіяти між собою.

Наступним етапом є визначення взаємодії між компонентами системи. Всі модулі мають мати чітко визначені інтерфейси, що дозволяє їм ефективно комунікувати один з одним. Важливо, щоб кожен компонент мав можливість змінюватися або замінюватися без значного впливу на інші частини системи. Це дозволяє створити більш гнучку і зручну для розвитку систему.

На основі розробленої архітектури створюється прототип або мінімально життєздатний продукт (MVP), який дозволяє протестувати основні функції програми і отримати зворотний зв'язок від користувачів. Це дозволяє визначити, чи відповідає система вимогам, чи потрібно вносити зміни до структури.

Тестування програмного забезпечення є невід'ємною частиною процесу. Після завершення розробки основної частини програмного продукту необхідно провести тестування на різних етапах, щоб виявити помилки та неполадки. Важливо перевіряти як окремі компоненти системи, так і її взаємодію в цілому. Це дозволяє усунути можливі проблеми на ранніх етапах і забезпечити стабільну роботу продукту після випуску.

Таким чином, розробка структури програмного забезпечення є ключовим етапом у створенні системи, оскільки вона визначає ефективність, масштабованість і безпеку продукту. Відповідно до вимог до програмного продукту потрібно

обрати правильну архітектуру, технології та розподіл компонентів, що забезпечить успішну реалізацію проекту.

3.3 Результати розробки програмного забезпечення

Таким чином, у результаті розробки програмного забезпечення було створено функціональну інформаційну систему для оренди житла. Система дозволяє користувачам взаємодіяти з платформою в двох ролях: орендодавця та орендаря.

Орендодавець має змогу додавати нові об'єкти нерухомості, вносити зміни до їхніх характеристик, оновлювати інформацію щодо доступності житла та відстежувати стан бронювань. Орендар, у свою чергу, може переглядати доступні пропозиції, бронювати квартири відповідно до своїх потреб та отримувати підтвердження щодо статусу бронювання.

Для обох типів користувачів реалізовано особистий кабінет, що забезпечує доступ до персональних даних, історії операцій та дозволяє здійснювати необхідні маніпуляції з інформацією. Це включає редагування профілю, зміну деталей орендованих або запропонованих об'єктів, а також перегляд стану актуальних запитів.

Розроблена система має зручний та мінімалістичний інтерфейс, який забезпечує інтуїтивно зрозумілу навігацію та комфортне використання. Використання сучасних технологій, зокрема HTML, CSS, JavaScript, PHP та MySQL, дозволило створити ефективну та продуктивну платформу, що відповідає сучасним вимогам ринку оренди нерухомості.

Отже, реалізоване програмне забезпечення сприяє автоматизації процесів оренди житла, підвищує зручність для користувачів та забезпечує ефективну взаємодію між орендодавцями та орендарями, що значно покращує якість обслуговування та оптимізує управління орендними угодами.

Текст програми представлений у додатку Б, інструкція користувача представлена у додатку В.

Висновки за розділом III

У цьому розділі описано вибір технологій та побудову структури для створення зручного та швидкого веб-додатка. Завдяки сучасним інструментам система працює стабільно, швидко реагує на дії користувачів і легко доопрацьовується.

У підсумку розроблено сайт для оренди житла з особистими кабінетами для орендодавців та наймачів. Він дозволяє додавати, переглядати та бронювати житло, автоматизує основні процеси і спрощує взаємодію між користувачами.

РОЗДІЛ IV. ОХОРОНА ПРАЦІ

4.1 Загальні питання охорони праці

Сучасні виробничі процеси супроводжуються значною кількістю небезпечних факторів для людей та довкілля. Вони включають складні технологічні системи з численними рухомими елементами, високі рівні загазованості, запиленості, шуму та інших шкідливих чинників. Навіть робота в офісі може бути потенційно небезпечною через використання комп'ютерної техніки, факсів, ксероксів та інших приладів, що містять компоненти, здатні негативно впливати на здоров'я за певних умов.

Підготовка майбутніх фахівців і керівників виробничих колективів у закладах вищої освіти має критичне значення для забезпечення безпечних умов праці. Якість їхньої освіти безпосередньо впливатиме на рівень безпеки на виробництві. Відповідальність за організацію безпечних умов вимагає від спеціалістів глибоких знань у сфері охорони праці, що передбачає розробку заходів для запобігання небезпечним впливам на робочих місцях. Важливим аспектом є усунення ризиків, які можуть призвести до нещасних випадків, і мінімізація людського фактора у виробничих процесах.

Сучасний стан охорони праці в Україні викликає занепокоєння. Забезпечення безпечних умов є важливою складовою соціально-економічного розвитку держави та ключовою функцією органів влади й підприємств. Останні десять років спостерігається тенденція до зниження виробничого травматизму, що дозволило зменшити загальну кількість травмованих у чотири рази, а смертельних випадків — удвічі. Однак у перші роки XXI століття ситуація залишається стабільною без суттєвих змін. Незважаючи на те, що підприємства розширюють виробничу діяльність, економічні труднощі обмежують можливості значного покращення безпеки праці. Щомісяця в Україні отримують виробничі травми близько 2200 осіб,

із яких близько 100 гинуть. Фонд соціального страхування опікується понад 300 тисячами постраждалих на виробництві.

Окрім виробничих ризиків, значну загрозу становить побутовий і невиробничий травматизм, який має тенденцію до зростання. В Україні рівень нещасних випадків, дорожньо-транспортних пригод, пожеж та інших небезпечних ситуацій перевищує показники розвинених країн у кілька разів. Щорічно фіксуються десятки тисяч смертей та сотні тисяч випадків травмування. Президент України визнав проблему зниження травматизму пріоритетним завданням державної безпеки. Аналогічні виклики існують у всьому світі. За даними Міжнародної організації праці, щороку у світі реєструється 270 мільйонів випадків виробничого травматизму та 160 мільйонів професійних захворювань, а близько двох мільйонів людей гине внаслідок небезпечних умов праці. Втрати для глобальної економіки через ці фактори сягають 1,25 трильйона доларів США.

Попри те, що в багатьох країнах умови праці покращилися, загальна ситуація у сфері охорони праці залишається складною. Глобалізація впливає на трудові процеси, сприяючи впровадженню нових технологій і змін у фінансово-економічних умовах. Проте цей процес має двоїстий характер: одні країни отримують економічні переваги, тоді як інші стикаються з погіршенням соціальних умов і зростанням ризиків на виробництві. В умовах міжнародної конкуренції роботодавці часто розглядають заходи з безпеки не як частину управління якістю, а як додаткові витрати, що можуть знизити конкурентоспроможність продукції.

Необхідність вивчення дисципліни «Основи охорони праці» обумовлена тим, що будь-яка професійна діяльність пов'язана з небезпечними та шкідливими чинниками, які можуть призвести до травм, професійних захворювань або навіть загибелі працівників. Забезпечення безпечних умов праці, пріоритет здоров'я людини та зменшення рівня ризиків в усіх сферах діяльності потребує високого рівня підготовки фахівців. Викладання основ охорони праці у вищих навчальних закладах дозволяє майбутнім спеціалістам отримати необхідні знання та застосовувати їх для створення безпечних умов праці, підвищення ефективності виробниц-

тва, зниження рівня професійних захворювань та мінімізації виробничого травматизму [13].

4.2 Структура управління питаннями охорони праці на підприємстві

Охорона праці виконує одразу дві ключові функції.

Перша – інженерно-технічна, яка спрямована на мінімізацію ризиків під час трудового процесу шляхом:

- заміни небезпечних матеріалів на менш шкідливі аналоги;
- впровадження інноваційних технологій, що знижують ймовірність травмування та професійних захворювань;
- проєктування та створення обладнання відповідно до стандартів безпеки;
- розробки засобів захисту як для окремих працівників, так і для колективу в цілому.

Друга – соціальна, яка передбачає компенсацію матеріальних, моральних та соціальних збитків, завданих унаслідок нещасного випадку чи професійного захворювання. Її мета – забезпечити захист працівника та його прав.

Виходячи з цих завдань, охорона праці, спираючись на правові та організаційні механізми, охоплює такі сфери, як виробнича санітарія, промислова та пожежна безпека.

До структурних елементів охорони праці належать:

- правові та організаційні засади безпеки праці;
- фізіологія, гігієна праці та виробнича санітарія;
- промислова безпека;
- пожежна безпека та профілактичні заходи на виробництві.

З наведеної на рис. 4.1 схеми видно, що правові та організаційні основи є фундаментом, який забезпечує соціальний захист працівників. Саме на цій основі будуються санітарно-гігієнічні та інженерно-технічні складові системи охорони праці.

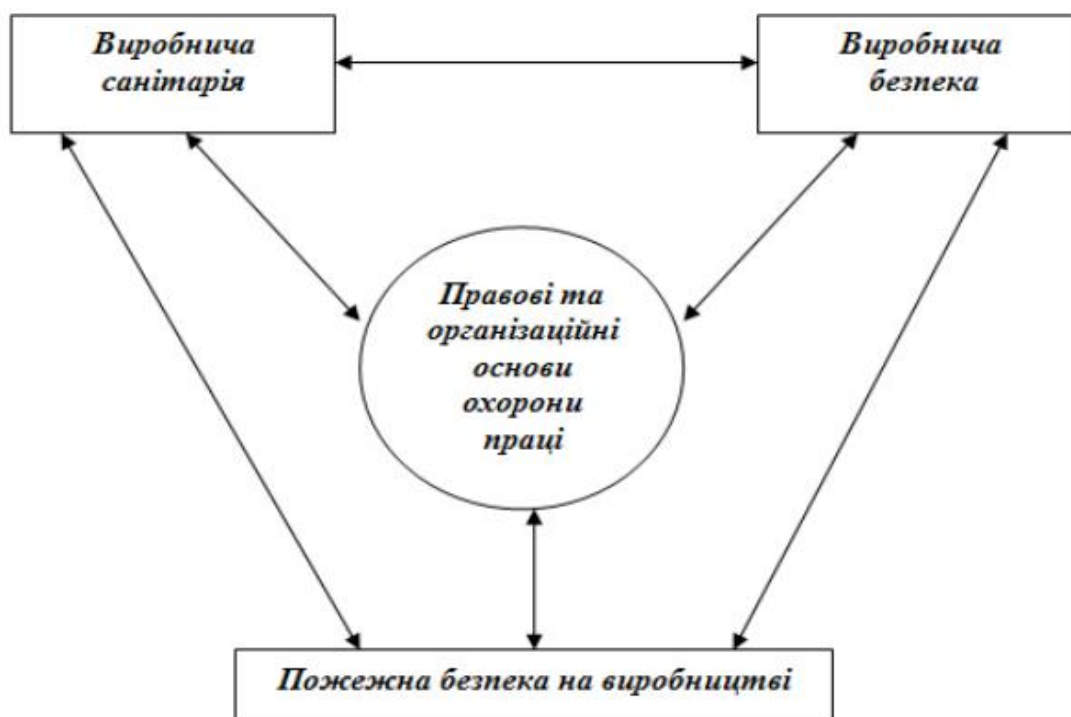


Рисунок 4.1 - Структурна схема охорони праці [14]

В Україні діє багаторівнева система управління охороною праці, яка охоплює державний, галузевий, регіональний та виробничий рівні.

Будь-яка система управління складається із суб'єкта та об'єкта управління, які взаємодіють у певному середовищі через інформаційні зв'язки. Суб'єкт управління включає керівний та виконавчий органи. Основна мета управління охороною праці – це забезпечення безпечних умов, збереження здоров'я та працездатності працівників у процесі їхньої діяльності.

Головне завдання впровадження системи управління охороною праці (СУ-ОП) – це створення безпечних і нешкідливих умов праці на всіх етапах виробничого процесу. При цьому важливо не тільки вчасно усувати порушення вимог охорони праці, а й запобігати їхньому виникненню.

Система управління охороною праці має кілька рівнів управління. На державному рівні управління здійснюють міністерства, галузеві відомства та науково-технічні відділи. Галузевий рівень включає об'єднання, концерни, комбінати та профільні відділи з охорони праці. На виробничому рівні управління здійснюють

підприємства та організації, де працюють відділи охорони праці та науково-технічні підрозділи. На цеховому рівні функціонують відділення, філії, інженери з охорони праці та спеціалісти. Останнім рівнем управління є робоче місце, де відповідальність за охорону праці несуть керівники лінійних служб та самі працівники.

Система управління охороною праці – це сукупність суб'єктів та об'єктів управління, які на основі нормативних документів здійснюють планомірну діяльність, спрямовану на забезпечення безпечних, здорових і продуктивних умов праці.

Основою охорони праці є законодавчі та нормативно-правові акти, які є ключовим джерелом інформації для СУОП. На рівні підприємства суб'єктом управління є роботодавець, а в межах цехів і виробничих підрозділів – керівники відповідних структур. Об'єктом управління є діяльність підприємства, спрямована на забезпечення безпечних умов праці.

Типова система управління охороною праці функціонує таким чином: керівник підприємства аналізує стан охорони праці, виявляє ризики та приймає рішення для підвищення рівня безпеки.

На функціонування СУОП впливають зовнішні чинники, такі як зміни у виробничому процесі, модернізація обладнання, коригування технологічних параметрів, випадки травматизму та професійні захворювання.

Система управління охороною праці керується законодавчими та нормативними актами. Роботодавець зобов'язаний розробляти, затверджувати та впроваджувати локальні нормативні акти – положення, інструкції та інші документи, що є обов'язковими для виконання на підприємстві [15].

4.3 Шкідливі та небезпечні фактори, що супроводжують працю програміста

Сьогодні професія програміста є однією з найпопулярніших у світі. Представники цієї сфери мають можливість працювати як в офісі, так і віддалено, од-

нак незалежно від місця роботи важливо забезпечити безпечні умови праці. Недотримання таких умов може призвести до впливу шкідливих факторів, які негативно позначаються на здоров'ї програміста та його продуктивності.

На роботу програміста впливають такі чинники, як невідповідний мікроклімат у приміщенні (температура, вологість), недостатнє освітлення, підвищений рівень шуму та електромагнітного випромінювання, погіршення іонного складу повітря, незручна організація робочого місця, перевантаження зору, фізичний дискомфорт, стрес тощо.

Оптимальними умовами для комфортної роботи вважаються температура повітря від 20 до 24 °С, відносна вологість у межах 40-60 %, атмосферний тиск близько 750 мм рт. ст., рівень запиленості не вище 10 мг/м³, швидкість руху повітря – до 0,1 м/с. Оскільки комп'ютерне обладнання є джерелом тепла, необхідно подбати про якісну вентиляцію, кондиціонування, а також підтримку оптимального рівня опалення. Для комфортної роботи обсяг приміщення має складати щонайменше 20 м³ на одну людину.

Однією з головних проблем програмістів є високе навантаження на зір, оскільки робота передбачає постійний зоровий контакт із монітором. Для роботи високої точності рівень загального освітлення має становити 300 лк, а при комбінованому освітленні (поєднання загального та локального джерела світла) – 750 лк. Важливо використовувати якісне штучне освітлення, наприклад люмінесцентні лампи, а також уникати відблисків на екрані монітора.

Окрім того, комп'ютерний монітор є джерелом випромінювання, що може негативно впливати на здоров'я. Оптимальна відстань від очей до екрана має бути не менше 50 см. Варто використовувати монітори зі зниженим рівнем випромінювання, а також регулярно робити перерви – 15 хвилин після кожної півтори години роботи. Для покращення якості повітря рекомендується встановлювати іонізатори, зволожувачі та нейтралізатори шкідливих випромінювань.

Ще одним негативним фактором є шум від комп'ютерів та периферійних пристроїв, який може впливати на слух і психоемоційний стан працівника. Допустимий рівень шуму не повинен перевищувати 50 дБА. Для його зменшення можна

використовувати звукопоглинальні матеріали для стін і встановлювати віброізолятори на поверхнях.

Важливе значення має і правильна організація робочого місця. Оптимальна висота робочого столу має бути 720 мм, а його поверхня – не менше 1600 x 1000 мм. Простір під столом повинен мати глибину не менше 650 мм, а для ніг слід передбачити підставку з нахилом 15°. Відстань між клавіатурою та краєм столу не повинна перевищувати 300 мм, а між очима та монітором – 40-80 см. Стілець повинен мати підйомно-поворотний механізм, регульовану висоту сидіння (400-500 мм), глибину не менше 380 мм і ширину не менше 400 мм. Висота опорної поверхні спинки має становити щонайменше 300 мм, а її ширина – не менше 380 мм. Кут нахилу спинки до сидіння повинен змінюватися в межах 90-110°.

Таким чином, для забезпечення безпечних умов праці програміста необхідно дотримуватися наведених норм і стандартів, що допоможе знизити ризики негативного впливу факторів робочого середовища та підвищити продуктивність праці [16].

4.4 Ергономічні вимоги до робочого місця при роботі за комп'ютером

При розташуванні технічних засобів на робочому місці, а також при налаштуванні крісла оператора, забезпечується зручний доступ до основних функціональних компонентів і модулів обладнання, що необхідно для технічного обслуговування, діагностики та ремонту. Робоче місце організоване так, щоб оператор міг легко займати та залишати його, уникати випадкового натискання на елементи керування, а також забезпечувати комфортну позу як під час роботи, так і в періоди відпочинку.

Розташування технічного та допоміжного обладнання відповідає санітарно-гігієнічним нормам, встановленим чинним законодавством.

Алфавітно-цифровий дисплей встановлюється на столі таким чином, щоб відстань до його екрана не перевищувала 700 мм, при цьому оптимальним вважа-

ється діапазон 500–700 мм. Екран розміщується так, щоб кут між лінією погляду та нормаллю до центру екрана становив 20° .

Кут огляду екрану та інших інформаційних засобів у горизонтальній площині не повинен перевищувати 60° .

Клавіатурний пульт встановлюється на висоті 650–720 мм від підлоги. Якщо пульт розташований на стаціонарному столі висотою 750 мм, необхідно використовувати крісло з регульованою висотою сидіння та підставкою для ніг.

Розміщення монітора, клавіатури та документів забезпечує мінімальний перепад яскравості їхніх поверхонь, що залежить від положення джерел світла. Цей перепад не перевищує 1:10 при рекомендованому значенні 1:3. При яскравості зображення на екрані в межах 50–100 кд/м² рівень освітленості документів має становити 300–500 лк. Важливо уникати ефекту засліплення, відблисків і віддзеркалень на екрані. Планування робочого простору здійснюється таким чином, щоб напрямок погляду оператора був паралельним до розташування вікон та світильників.

Елементи керування вентиляцією та панелі сигналізації розташовуються безпосередньо перед оператором у зоні легкої доступності. Телефонні апарати та засоби зв'язку встановлюються зліва, що дозволяє оператору користуватися правою рукою для записів.

Пристрої документування розташовані праворуч у межах досяжності. Для зручності організації робочого місця клавіатурний пульт і екран є окремими елементами, з'єднаними за допомогою кабелів.

Робочий процес на ЕОМ у межах восьмигодинної зміни передбачає регулярні перерви. Для розробників програм рекомендується 15-хвилинний відпочинок після кожної години роботи, оператори персональних комп'ютерів роблять перерву по 15 хвилин після кожних двох годин роботи, а оператори комп'ютерного набору – по 10 хвилин після кожної години [17].

4.5 Нормативні вимоги до робочої площі на одного працюючого за комп'ютером

При організації робочого місця для працівників, які використовують екранні пристрої, слід надавати перевагу обладнанню, що не створює зайвого шумового навантаження та не продукує надмірного тепла. Дотримання допустимого рівня шуму на робочих місцях регулюється відповідно до ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку», затверджених постановою Головного державного санітарного лікаря України від 01.12.1999 № 37. На необхідність виконання цих вимог також вказує пункт 4 розділу II НПАОП 0.00-7.15-18.

Роботодавець зобов'язаний організовувати регламентовані внутрішні перерви для відпочинку працівників, компенсуючи навантаження, що виникає при роботі з екранними пристроями. Це здійснюється відповідно до ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин», затверджених постановою Головного державного санітарного лікаря України від 10.12.1998 № 7. Пункт 5 розділу II НПАОП 0.00-7.15-18 визначає, що ці норми застосовуються виключно щодо тривалості робочої зміни. Водночас вимоги пункту 2.3 ДСанПіН 3.3.2.007-98, які передбачають мінімальну площу 6 м² та об'єм 20 м³ на одне робоче місце, не є обов'язковими.

Додаткові умови та нормативи

- Працівники, які працюють із рідкокристалічними відеотерміналами ЕОМ, можуть мати право на додаткові відпустки.
- Роботодавець повинен визначати необхідність атестації робочого місця для осіб з інвалідністю, які працюють за комп'ютером.
- Організація медичних оглядів для працівників, які постійно використовують персональні комп'ютери, регламентується окремими нормативними документами.

Відповідно до пункту 1 розділу III НПАОП 0.00-7.15-18, робочі місця користувачів екранних пристроїв повинні бути розраховані так, щоб забезпечувати можливість змінювати положення тіла та рухатися без перешкод.

Щодо площі приміщень, то:

- Для адміністративних та побутових приміщень мінімальна площа на одне робоче місце повинна становити 6 м².
- Для працівників конструкторських бюро та осіб з інвалідністю, які користуються кріслами-колясками, площа має бути не менше ніж 7,65 м².
- У разі використання великогабаритного обладнання або розміщення в приміщенні пристроїв колективного користування площа може бути збільшена відповідно до технічних умов експлуатації обладнання.

Ці вимоги визначені у пункті 5.1.1.1 ДБН В.2.2-28:2010 «Будинки і споруди. Будинки адміністративного та побутового призначення».

Також згідно з пунктом 5.1.1.2 ДБН В.2.2-28:2010 площа кабінетів керівників не повинна перевищувати 15% від загальної площі робочих приміщень управління [18].

Висновки за розділом IV

Охорона праці є невід'ємною частиною сучасного виробничого процесу та ключовим чинником забезпечення здоров'я робітників у будь-якій сфері діяльності. Попри позитивну динаміку зменшення виробничого травматизму в Україні, загальний стан безпеки праці вимагає подальшого вдосконалення. Система управління охороною праці передбачає багаторівневий підхід, що охоплює як державний, так і виробничий щабель, з акцентом на профілактику ризиків і створення безпечних умов праці. Особливу увагу варто приділяти організації робочих місць для робітників, які використовують комп'ютерну техніку, з врахуванням ергономічних та санітарно-гігієнічних вимог, що сприятиме збереженню їхнього здоров'я та підвищенню продуктивності.

ВИСНОВОК

У процесі виконання дипломної роботи було здійснено повний цикл розробки веб-додатку для безпечної оренди житла. Основна увага приділялася аналізу предметної галузі, визначенню актуальних проблем, формулюванню задач, розробці та реалізації інформаційної системи, а також питанням охорони праці під час роботи з програмним забезпеченням.

На першому етапі роботи було проведено ґрунтовне дослідження ринку оренди житла, виявлено ключові проблеми, з якими стикаються орендарі та орендодавці. Було проаналізовано існуючі рішення у цій сфері, їхні переваги та недоліки. Основними проблемами, які виявилися найбільш актуальними, стали:

- Відсутність надійних механізмів перевірки орендодавців та орендарів.
- Ризики шахрайства та неправомірного використання даних.
- Складність пошуку відповідного житла через недостатню фільтрацію та структурованість інформації.

- Відсутність зручних механізмів для комунікації між користувачами.

На основі виявлених проблем було сформульовано завдання для розробки системи, яка має забезпечити безпечний процес оренди житла, зручність у користуванні та високу продуктивність.

Під час розробки проєкту інформаційної підсистеми було розглянуто та обґрунтовано загальносистемні рішення, створено необхідні діаграми, які деталізують роботу системи:

- Діаграми варіантів використання для візуалізації основних ролей користувачів та їхніх можливостей у системі.

- Діаграми компонентів для моделювання структури програмного забезпечення та взаємодії між складовими.

- Діаграми переходів станів для опису життєвого циклу об'єктів у системі.

- Діаграми розгортання для визначення технічної інфраструктури, на якій буде розгорнуто систему.

Було розроблено інформаційну модель системи, яка включає логічну та фізичну моделі даних. Визначено основні сутності, їхні характеристики та зв'язки між ними, що дозволило ефективно спроектувати базу даних та забезпечити коректну роботу всієї системи.

На етапі реалізації було обґрунтовано вибір технологій, які використовувалися при створенні веб-додатку. Вибір технологічного стека був зроблений на основі вимог до продуктивності, безпеки та зручності використання:

- Фронтенд реалізовано за допомогою HTML, CSS та JavaScript, що дозволило створити мінімалістичний, інтуїтивно зрозумілий та адаптивний інтерфейс.

- Бекенд побудовано на основі PHP, що забезпечує динамічну генерацію сторінок, обробку запитів користувачів та взаємодію з базою даних.

- База даних розроблена на основі MySQL, що дозволяє ефективно зберігати, обробляти та забезпечувати доступ до необхідної інформації.

Було реалізовано наступний функціонал:

- Можливість орендодавцям додавати об'єкти нерухомості, редагувати та видаляти їх.

- Функція пошуку та фільтрації житла для орендарів.

- Реалізація особистих кабінетів для користувачів, у яких можна переглядати історію бронювань, управляти оголошеннями та вести комунікацію між учасниками процесу.

- Система реєстрації та авторизації користувачів з можливістю перевірки особистих даних для зменшення ризику шахрайства.

Окремим розділом дипломної роботи було розглянуто питання охорони праці при роботі програмістів. Були визначені шкідливі та небезпечні фактори, що можуть впливати на здоров'я під час тривалої роботи за комп'ютером, такі як:

- Висока зорове навантаження.
- Неправильна організація робочого місця.
- Вплив електромагнітного випромінювання.
- Малорухливий спосіб життя.

На основі проведеного аналізу були розглянуті заходи щодо оптимізації робочого місця програмістів відповідно до нормативних вимог, зокрема, ергономічні аспекти, норми площі робочого місця та регламентовані перерви для зменшення впливу шкідливих факторів.

Результатом роботи стало створення веб-додатку для безпечної оренди житла, який дозволяє значно спростити процес взаємодії між орендарями та орендодавцями, мінімізувати ризики шахрайства та забезпечити зручний механізм пошуку та бронювання житла.

Головні досягнення проєкту:

- Реалізація безпечного та зручного механізму оренди житла.
- Розробка структурованої бази даних, яка ефективно зберігає інформацію про орендарів, орендодавців та об'єкти нерухомості.
- Використання сучасних веб-технологій для створення продуктивної, адаптивної та захищеної системи.
- Застосування концепцій безпеки, таких як верифікація користувачів та захист персональних даних.

У перспективі можливий подальший розвиток системи, зокрема:

- Інтеграція картографічних сервісів для більш зручного пошуку житла за локацією.
- Впровадження рейтингової системи користувачів для підвищення довіри між учасниками орендних угод.
- Автоматизовані рекомендації на основі попередніх пошуків та уподобань користувачів.

Отримані результати підтверджують актуальність і практичну значущість створеного рішення, яке може бути використане у реальному житті та вдосконалене відповідно до вимог користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Господарський кодекс України від 16.01.2003 // Відомості Верховної Ради, 2003. – № 18-22
2. Дзера О.В. Цивільне право України: Підручник: У 2-х кн. / О. В. Дзери (кер. авт. кол.), Д. В. Боброва, А. С. Довгерт та ін.; За ред. О. В. Дзери, Н. С. Кузнецової. – К.: «Юрінком Інтер», 2004. – Кн. 1. – С. 147.
3. Закон України «Про іпотеку» від 05.06.2003 р. № 898-ІУ // Відомості Верховної Ради України. – 2003. – № 38. – С. 313.
4. Закон України «Про оренду державного та комунального майна» від 16.10.04.1992р. // ВВР України. – 1992. – № 30. – 416 с.
5. Береза А.М. Електронна комерція: навчальний посібник. К.: КНЕУ, 2002. – 26 с.
6. Бобко Т.В. Особливості ринку нерухомості в Україні / Т.В. Бобко // Економіка та держава. – 2018. – № 3. – С. 69-72.
7. Use Cases і сценарії. URL: <https://www.maxzosim.com/use-cases-and-scenarios/> (дата звернення: 05. 10.2024)
8. Лосєв М. Ю. Бази даних : навчально-практичний посібник для самостійної роботи студентів. Харків : ХНЕУ ім. С. Кузнеця, 2018. – 233 с
9. Моделі даних. Реляційна модель даних. URL: https://elib.lntu.edu.ua/sites/default/files/elib_upload/BD_2016_3/page5.html (дата звернення: 05. 10.2024)
10. Open Server. URL: <https://ospanel.io/> (дата звернення: 05. 02.2025)
11. Моделювання станів. URL: <https://www.maxzosim.com/state-modelling/> (дата звернення: 05. 02.2025)
12. Пустюльга С.І., Самчук В.П. Технології вебдизайну : Навчальний посібник. – Луцьк : Вежа, 2023. – 604 с.
13. Вступ. Загальні питання охорони праці. URL: <http://studfile.net/preview/10421935/page:2/> (дата звернення: 05. 02.2025)

14. Загальні питання з охорони праці. URL: <https://opcb.kpi.ua/wp-content/uploads/2014/08/Binder21.pdf> (дата звернення: 05. 02.2025)

15. Заїченко В. І. Менеджмент охорони праці : конспект лекцій (для здобувачів денної та заочної форм навчання першого (бакалаврського) рівня вищої освіти напряму підготовки спеціальності 6.170202 Охорона праці, спеціальність 263 – Цивільна безпека, освітня програма «Охорона праці» / В. І. Заїченко ; Харків. нац. унів. міськ. госп-ва. ім. О. М. Бекетова – Харків : ХНУМГ ім. О. М. Бекетова, 2021. – 82 с.

16. Небезпечні та шкідливі фактори, які впливають на програміста і засоби їх усунення. URL: <https://openarchive.nure.ua/server/api/core/bitstreams/94079df3-086d-4f0d-b501-af17ada9c1ec/content> (дата звернення: 05. 02.2025)

17. Ергономічні вимоги до робочого місця. URL: <https://studfile.net/preview/8746943/page:4/> (дата звернення: 05. 02.2025)

18. Облаштування робочого місця працівника офісу. URL: <https://pro-op.com.ua/question/702-vimogi-do-robochogo-mstsy-pri-robot-z-pk> (дата звернення: 05. 02.2025)

ДОДАТОК А –Технічне завдання

Вступ

Завданням даної дипломної роботи є розробка web застосунку для безпечної оренди житла. В таблиці 1 представлено загальні відомості про систему.

Таблиця 1

Загальні відомості про систему

Повна назва системи	Розробка WEB додатку для безпечної оренди житла
Умовне позначення системи	РД Д БОЖ
Розроблювач	Коркасенко Аліна Владиславівна
Замовник	Кафедра інформаційних управляючих систем та технологій

1. Підстави для розробки

РД Д БОЖ розробляється на підставі завдання для дипломного проектування за спеціальністю 124 – «Системний аналіз».

2. Призначення розробки

Основним завданням WEB додатку є:

- легка оренда та здача житла;

3. Мета створення системи

Метою даного проекту є створення веб-додатку, який дозволяє орендарям шукати та бронювати житло, а орендодавцям додавати та керувати своїми об'єктами нерухомості. Додаток має бути простим у використанні, інтуїтивно зрозумілим, а також забезпечувати високий рівень безпеки даних та комунікацій між користувачами.

4. Вимоги до програми

4.1 Вимоги до функціональних характеристик:

можливість авторизуватися за допомогою електронної пошти;
орендодавці можуть додавати нові об'єкти для оренди;
пошук житла за різними критеріями з можливостями комбінувати кількість критеріїв;
шифрування даних при реєстрації (паролі);
адаптивність під різні пристрої.

4.2 Вимоги до надійності

При проектуванні повинні бути використані типові проектні рішення там, де це можливо. На стадії технічного проектування повинні прийматися рішення, що зменшують трудомісткість експлуатації системи.

4.3 Умови експлуатації

РД Д БОЖ використовує наступний стек технологій:

для фронтенду – HTML, CSS, JavaScript;

для бекенду – PHP;

бази даних – MySQL.

4.4 Вимоги до інформаційної та програмної сумісності

Потрібно, щоб версія операційної системи була Windows 7 або новішою (Windows 10 чи 11), з підтримкою як 32-розрядної, так і 64-розрядної версії. Також для коректної роботи можуть знадобитися деякі додаткові програмні компоненти, такі як Microsoft Visual C++ Redistributable і Microsoft .NET Framework. Зазвичай, якщо вони не встановлені в системі, програма запропонує їх інсталиувати під час запуску.

Рекомендовані технічні характеристики для програмного забезпечення, які будуть працювати з даним програмним забезпеченням мають бути такі:

Дискового простору 30 ГБ;

4 ГБ оперативної пам'яті;

Два й більше ядерний процесор з частотою не менше ніж 2.0 ГГц;

4.5 Вимоги до технічного забезпечення

Технічні вимоги до клієнтських комп'ютерів визначаються вимогами операційної систем.

5. Вимоги до програмної документації

Усі регламентуючі документи є елементами РД Д БОЖ, які безпосередньо входять до цього. Регламент системи описує правила її ефективного використання. В таблиці 2 наведено склад документів.

Таблиця 2

Склад документів організаційно-методичного та експлуатаційного забезпечення

	Назва	Стадія
	Опис організаційної структури ІПГІА	Технічне проектування
	Керівництво користувача	Реалізація системи

Стадії та етапи розробки

В таблиці 3 представлено стадії та етапи по створенню РД Д БОЖ.

Стадії та етапи робіт по створенню АРМ СВК

	Стадія	Документи
	Аналіз предметної області	Звіт з описом предметної області
	Системне проектування	Технічне завдання
	Ескізне проектування	Ескізний проект
	Технічне проектування	Технічний проект
	Робоче проектування	Робочий проект

ДОДАТОК Б – Програмний код

Лістинг бази даних

```
-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Хост: 127.0.0.1:3306
-- Час створення: Бер 12 2025 р., 17:09
-- Версія сервера: 5.7.39
-- Версія PHP: 7.2.34

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- База даних: `neryh`
--

--
-- Структура таблиці `apartments`
--

CREATE TABLE `apartments` (
  `apartment_id` int(11) NOT NULL,
  `category_id` int(11) DEFAULT NULL,
  `city_id` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL,
  `address` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `description` text COLLATE utf8mb4_unicode_ci,
  `price` decimal(10,2) NOT NULL,
  `bedrooms` int(11) NOT NULL,
  `bathrooms` int(11) NOT NULL,
  `area` decimal(10,2) NOT NULL,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `image_path` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп даних таблиці `apartments`
--
```



```

INSERT INTO `apartments` (`apartment_id`, `category_id`, `city_id`,
`user_id`, `address`, `description`, `price`, `bedrooms`,
`bathrooms`, `area`, `created_at`, `image_path`) VALUES
(1, 1, 1, 1, 'ghfgfg', 'fgfgh', '55.00', 1, 2, '34.00', '2024-05-31
15:16:05', 'uploads/2.jpg'),
(4, 1, 2, 1, 'папап', 'прпап', '4454.00', 3, 2, '33.00', '2024-05-31
15:43:09', 'uploads/1.jpg'),
(5, 2, 1, 1, 'ппапап', 'прапа', '443434.00', 1, 1, '4234.00', '2024-
05-31 15:43:27', 'uploads/3.jpg'),
(6, 1, 2, 2, 'аава', 'аапап', '34334.00', 1, 1, '34343.00', '2025-
03-10 20:55:59', 'uploads/appartement-moderne-a-katowice-pologne-
kast-design-img~d7a1430d0979a5ed_4-7861-1-1371fbc.jpg'),
(7, 1, 4, 7, 'fdfdf', 'ggfg', '34343.00', 1, 1, '34434.00', '2025-
03-10 21:08:00', 'uploads/211131631.jpg'),
(8, 1, 3, 2, 'нова кв', 'ррррп', '454545.00', 2, 1, '4545.00',
'2025-03-12 14:05:45', 'uploads/images.jpg');

```

```
-- -----
```

```
--
-- Дублююча структура для представлення `apartments_view`
-- (Див. нижче для фактичного подання)
--
```

```

CREATE TABLE `apartments_view` (
`apartment_id` int(11)
, `category_id` int(11)
, `category_name` varchar(100)
, `city_id` int(11)
, `city_name` varchar(100)
, `user_id` int(11)
, `address` varchar(255)
, `description` text
, `price` decimal(10,2)
, `bedrooms` int(11)
, `bathrooms` int(11)
, `area` decimal(10,2)
, `created_at` timestamp
, `image_path` varchar(255)
);

```

```
-- -----
```

```
--
-- Структура таблиці `Categories`
--
```

```

CREATE TABLE `Categories` (
`category_id` int(11) NOT NULL,
`category_name` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

```

```
--
-- Дамп даних таблиці `Categories`

```

```
--

INSERT INTO `Categories` (`category_id`, `category_name`) VALUES
(1, 'Однокімнатні'),
(2, 'Двокімнатні');

-- -----

--
-- Структура таблиці `Cities`
--

CREATE TABLE `Cities` (
  `city_id` int(11) NOT NULL,
  `city_name` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп даних таблиці `Cities`
--

INSERT INTO `Cities` (`city_id`, `city_name`) VALUES
(1, 'Київ'),
(2, 'Суми'),
(3, 'Полтава'),
(4, 'Рівне'),
(5, 'Вінниця'),
(6, 'Хмельницький');

-- -----

--
-- Структура таблиці `contracts`
--

CREATE TABLE `contracts` (
  `id` int(11) NOT NULL,
  `apartment_id` int(11) NOT NULL,
  `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `term_id` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп даних таблиці `contracts`
--

INSERT INTO `contracts` (`id`, `apartment_id`, `name`, `email`,
`created_at`, `term_id`) VALUES
(1, 1, 'test', 'gfgg@gmail.com', '2024-05-31 16:45:53', 2),
(2, 1, 'авав', 'fglkfgf@gmail.com', '2025-03-10 21:00:34', 1),
(3, 8, 'ffdfld', 'fgfgfg@gmail.com', '2025-03-12 14:07:58', 1);
```

```

-- -----
--
-- Структура таблиці `contract_terms`
--

CREATE TABLE `contract_terms` (
  `id` int(11) NOT NULL,
  `term_text` text COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп даних таблиці `contract_terms`
--

INSERT INTO `contract_terms` (`id`, `term_text`) VALUES
(1, 'купля-продажа'),
(2, 'гарантія'),
(3, 'іпотека'),
(4, 'оренда');

-- -----
--
-- Структура таблиці `Users`
--

CREATE TABLE `Users` (
  `user_id` int(11) NOT NULL,
  `username` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `password_hash` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `email` varchar(100) COLLATE utf8mb4_unicode_ci NOT NULL,
  `user_type` enum('landlord','tenant') COLLATE utf8mb4_unicode_ci
NOT NULL,
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп даних таблиці `Users`
--

INSERT INTO `Users` (`user_id`, `username`, `password_hash`,
`email`, `user_type`, `created_at`) VALUES
(1, 'user', '$2y$10$k0YFfx.datjAUP7x7wOGobOAeSvlBXyYJsc8XfOtPLcPawqXyVp2Ce',
'test@gmail.com', 'landlord', '2024-05-31 11:26:27'),
(2, 'test', '$2y$10$IXTLzdIxsv9t07PvzaWG8.eMWYmnDglWrDWroztwobrhzj9BmJJHq',
'testt@gmail.com', 'landlord', '2025-03-10 16:47:21'),
(3, 'testordendator', '$2y$10$SgpEcE9N2rZ/qJHGjQ5wReOjQ9/CGA2hoMaL7dSa/atzxjX04HLdK',
'testordendator@gmail.com', 'tenant', '2025-03-10 20:26:13'),

```

```
(7, 'testorendodatel',
'$2y$10$n.5QHvZpbKxPj5jWOCAPNOEkZ9Z7d8bChPC3WnPxiQ3QVXwfZ7A6',
'testorendodatel@gmail.com', 'landlord', '2025-03-10 21:07:36');
```

```
-- -----
```

```
--
```

```
-- Структура для представлення `apartments_view`
```

```
--
```

```
DROP TABLE IF EXISTS `apartments_view`;
```

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`%` SQL SECURITY DEFINER
VIEW `apartments_view` AS SELECT `a`.`apartment_id` AS
`apartment_id`, `a`.`category_id` AS `category_id`,
`c`.`category_name` AS `category_name`, `a`.`city_id` AS `city_id`,
`ci`.`city_name` AS `city_name`, `a`.`user_id` AS `user_id`,
`a`.`address` AS `address`, `a`.`description` AS `description`,
`a`.`price` AS `price`, `a`.`bedrooms` AS `bedrooms`,
`a`.`bathrooms` AS `bathrooms`, `a`.`area` AS `area`,
`a`.`created_at` AS `created_at`, `a`.`image_path` AS `image_path`
FROM ((`apartments` `a` join `categories` `c` on((`a`.`category_id`
= `c`.`category_id`))) join `cities` `ci` on((`a`.`city_id` =
`ci`.`city_id`))) ;
```

```
--
```

```
-- Індекси збережених таблиць
```

```
--
```

```
--
```

```
-- Індекси таблиці `apartments`
```

```
--
```

```
ALTER TABLE `apartments`
  ADD PRIMARY KEY (`apartment_id`),
  ADD KEY `category_id` (`category_id`),
  ADD KEY `city_id` (`city_id`),
  ADD KEY `user_id` (`user_id`);
```

```
--
```

```
-- Індекси таблиці `Categories`
```

```
--
```

```
ALTER TABLE `Categories`
  ADD PRIMARY KEY (`category_id`);
```

```
--
```

```
-- Індекси таблиці `Cities`
```

```
--
```

```
ALTER TABLE `Cities`
  ADD PRIMARY KEY (`city_id`);
```

```
--
```

```
-- Індекси таблиці `contracts`
```

```
--
```

```
ALTER TABLE `contracts`
```

```

    ADD PRIMARY KEY (`id`),
    ADD KEY `apartment_id` (`apartment_id`),
    ADD KEY `fk_contract_term` (`term_id`);

--
-- Індекси таблиці `contract_terms`
--
ALTER TABLE `contract_terms`
    ADD PRIMARY KEY (`id`);

--
-- Індекси таблиці `Users`
--
ALTER TABLE `Users`
    ADD PRIMARY KEY (`user_id`),
    ADD UNIQUE KEY `username` (`username`),
    ADD UNIQUE KEY `email` (`email`);

--
-- AUTO_INCREMENT для збережених таблиць
--

--
-- AUTO_INCREMENT для таблиці `apartments`
--
ALTER TABLE `apartments`
    MODIFY `apartment_id` int(11) NOT NULL AUTO_INCREMENT,
    AUTO_INCREMENT=9;

--
-- AUTO_INCREMENT для таблиці `Categories`
--
ALTER TABLE `Categories`
    MODIFY `category_id` int(11) NOT NULL AUTO_INCREMENT,
    AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT для таблиці `Cities`
--
ALTER TABLE `Cities`
    MODIFY `city_id` int(11) NOT NULL AUTO_INCREMENT,
    AUTO_INCREMENT=7;

--
-- AUTO_INCREMENT для таблиці `contracts`
--
ALTER TABLE `contracts`
    MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT для таблиці `contract_terms`
--
ALTER TABLE `contract_terms`

```

```

MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT для таблиці `Users`
--
ALTER TABLE `Users`
  MODIFY `user_id` int(11) NOT NULL AUTO_INCREMENT,
  AUTO_INCREMENT=8;

--
-- Обмеження зовнішнього ключа збережених таблиць
--

--
-- Обмеження зовнішнього ключа таблиці `apartments`
--
ALTER TABLE `apartments`
  ADD CONSTRAINT `apartments_ibfk_1` FOREIGN KEY (`category_id`)
  REFERENCES `Categories` (`category_id`),
  ADD CONSTRAINT `apartments_ibfk_2` FOREIGN KEY (`city_id`)
  REFERENCES `Cities` (`city_id`),
  ADD CONSTRAINT `apartments_ibfk_3` FOREIGN KEY (`user_id`)
  REFERENCES `Users` (`user_id`);

--
-- Обмеження зовнішнього ключа таблиці `contracts`
--
ALTER TABLE `contracts`
  ADD CONSTRAINT `contracts_ibfk_1` FOREIGN KEY (`apartment_id`)
  REFERENCES `apartments` (`apartment_id`),
  ADD CONSTRAINT `fk_contract_term` FOREIGN KEY (`term_id`)
  REFERENCES `contract_terms` (`id`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Лістинг сторінки index.php

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Neryh Real Estate</title>
  <meta charset="UTF-8">
  <meta name="description" content="Arcade - Architecture
Template">
  <meta name="keywords" content="arcade, architecture, onepage,
creative, html">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <!-- Favicon -->
  <link href="img/favicon.ico" rel="shortcut icon"/>

```

```

<!-- Google Fonts -->
<link
href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700
" rel="stylesheet">

<!-- Stylesheets -->
<link rel="stylesheet" href="css/bootstrap.min.css"/>
<link rel="stylesheet" href="css/font-awesome.min.css"/>
<link rel="stylesheet" href="css/animate.css"/>
<link rel="stylesheet" href="css/owl.carousel.css"/>
<link rel="stylesheet" href="css/style.css"/>

<style>
.modal {
  display: none;
  position: fixed;
  z-index: 9999;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  overflow: auto;
  background-color: rgba(0,0,0,0.4);
  padding-top: 60px;
}

/* Вміст модального вікна */
.modal-content {
  background-color: #fefefe;
  margin: 5% auto;
  padding: 20px;
  border: 1px solid #888;
  width: 80%;
}

/* Кнопка закриття модального вікна */
.close {
  color: #aaa;
  float: right;
  font-size: 28px;
  font-weight: bold;
}

.close:hover,
.close:focus {
  color: black;
  text-decoration: none;
  cursor: pointer;
}
</style>
</head>
<body>
<!-- Page Preloder -->

```

```

<div id="preloader">
    <div class="loader"></div>
</div>
<!-- Header section start -->
<header class="header-area">
    <a href="index.php" class="logo-area">
        
    </a>
    <div class="nav-switch">
        <i class="fa fa-bars"></i>
    </div>

    <nav class="nav-menu">
        <ul>
                                <li class="active"><a
href="index.php">Головна</a></li>

                                <li><a href="solds.php">Продажі</a></li>

                                <li><a href="#"
id="openLoginModal">Авторизація</a></li>
                                <li><a href="#"
id="openRegisterModal">Реєстрація</a></li>
        </ul>
    </nav>
</header>
<!-- Login Modal -->
<!-- Login Modal -->
<div id="loginModal" class="modal">
    <div class="modal-content">
        <span class="close" id="closeLoginModal">&times;</span>
        <h2>Авторизація</h2>
        <form action="login.php" method="post">
            <label for="username">Логін</label>
            <input type="text" id="username" name="username"
required>

            <label for="password">Пароль</label>
            <input type="password" id="password" name="password"
required>

            <!-- Додавання вибору типу користувача на новій стрічці
-->
            <label for="user_type">Тип користувача</label><br>
            <div>
                <input type="radio" id="tenant" name="user_type"
value="tenant" required>
                <label for="tenant">Орендар</label>
            </div>
            <div>
                <input type="radio" id="landlord" name="user_type"
value="landlord" required>

```



```

        <label for="landlord">Орендодавець</label>
    </div>

    <button type="submit">Авторизація</button>
</form>
</div>
</div>

<!-- Register Modal -->
<div id="registerModal" class="modal">
    <div class="modal-content">
        <span class="close" id="closeRegisterModal">&times;</span>
        <h2>Реєстрація</h2>
        <form action="register.php" method="post">
            <label for="reg_username">Логін</label>
            <input type="text" id="reg_username" name="username" required>

            <label for="reg_password">Пароль</label>
            <input type="password" id="reg_password" name="password"
required>

            <label for="reg_email">Email</label>
            <input type="email" id="reg_email" name="email" required>

            <label for="user_type">Тип користувача</label>
            <select name="user_type" id="user_type" required>
                <option value="tenant">Орендатор</option>
                <option value="landlord">Орендодаватель</option>
            </select>

            <button type="submit">Зареєструватися</button>
        </form>
    </div>
</div>

<!-- JavaScript для відкриття та закриття модальних вікон -->
<script>
    document.addEventListener('DOMContentLoaded', (event) => {
        var loginModal = document.getElementById("loginModal");
        var registerModal =
document.getElementById("registerModal");
        var openLoginModal =
document.getElementById("openLoginModal");
        var openRegisterModal =
document.getElementById("openRegisterModal");
        var closeLoginModal =
document.getElementById("closeLoginModal");
        var closeRegisterModal =
document.getElementById("closeRegisterModal");

        openLoginModal.onclick = function() {
            loginModal.style.display = "block";
        }
    });

```

```

openRegisterModal.onclick = function() {
    registerModal.style.display = "block";
}

closeLoginModal.onclick = function() {
    loginModal.style.display = "none";
}

closeRegisterModal.onclick = function() {
    registerModal.style.display = "none";
}

window.onclick = function(event) {
    if (event.target == loginModal) {
        loginModal.style.display = "none";
    }
    if (event.target == registerModal) {
        registerModal.style.display = "none";
    }
}
});
</script>

<!-- Hero section start -->
<section class="hero-section">
    <!-- left social link bar -->
    <div class="left-bar">
        <div class="left-bar-content">
            <div class="social-links">
                <a href="#"><i class="fa fa-pinterest"></i></a>
                <a href="#"><i class="fa fa-linkedin"></i></a>
                <a href="#"><i class="fa fa-instagram"></i></a>
                <a href="#"><i class="fa fa-facebook"></i></a>
                <a href="#"><i class="fa fa-twitter"></i></a>
            </div>
        </div>
    </div>
    <!-- hero slider area -->
    <div class="hero-slider">
        <div class="hero-slide-item set-bg" data-
setbg="img/bg.jpg">
            <div class="slide-inner">
                <div class="slide-content">
                    <h2>Мінімалістична <br>Архітектура <br> та
більше</h2>
                    <a href="#" class="site-btn sb-light">Дивитися
проект</a>
                </div>
            </div>
        </div>
    </div>

```

```

        <div class="hero-slide-item set-bg" data-
setbg="img/bg.jpg">
        <div class="slide-inner">
            <div class="slide-content">
                <h2>Мінімалістична <br>Архітектура <br> та
Більше</h2>
                <a href="#" class="site-btn sb-light">Дивитися
проект</a>
            </div>
        </div>
    </div>
    <div class="slide-num-holder" id="snh-1"></div>
    <div class="hero-right-text">архітектура</div>
</section>
<!-- Hero section end -->

<!-- Intro section start -->
<section class="intro-section pt100 pb50">
    <div class="container">
        <div class="row">
            <div class="col-lg-7 intro-text mb-5 mb-lg-0">
                <h2 class="sp-title">Ми творча <span>Компанія</span>
нерухомості</h2>
                <a href="#" class="site-btn sb-dark">Дивитися
проект</a>
            </div>
            <div class="col-lg-5 pt-4">
                
            </div>
        </div>
    </div>
</section>
<!-- Intro section end -->

<!-- Service section start -->
<section class="service-section spad">
    <div class="container">
        <div class="section-title">
            <h2>Послуги</h2>
        </div>
        <div class="row">
            <div class="col-lg-4 col-md-6">
                <div class="service-box">
                    <div class="sb-icon">
                        <i class="fa fa-paint-brush"></i>
                    </div>
                    <h3>Архітектура</h3>
                </div>
            </div>
            <div class="col-lg-4 col-md-6">
                <div class="service-box">
                    <div class="sb-icon">

```

[illegible]

```

<!--===== Javascripts & JQuery =====>
<script src="js/jquery-2.1.4.min.js"></script>
<script src="js/bootstrap.min.js"></script>
<script src="js/isotope.pkgd.min.js"></script>
<script src="js/owl.carousel.min.js"></script>
<script src="js/jquery.owl-filter.js"></script>
<script src="js/magnific-popup.min.js"></script>
<script src="js/circle-progress.min.js"></script>
<script src="js/main.js"></script>
</body>
</html>

```

Лістинг сторінки home.php

```

<?php
session_start(); // початок сесії
?>
<!DOCTYPE html>
<html lang="uk">
<head>
    <title>Головна</title>
    <meta charset="UTF-8">
    <meta name="description" content="Шаблон Аркади - Архітектура">
    <meta name="keywords" content="аркада, архітектура,
односторінковий, креативний, html">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <!-- Іконка сайту -->
    <link href="img/favicon.ico" rel="shortcut icon"/>

    <!-- Шрифти Google -->
    <link
href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700
" rel="stylesheet">

    <!-- Таблиці стилів -->
    <link rel="stylesheet" href="css/bootstrap.min.css"/>
    <link rel="stylesheet" href="css/font-awesome.min.css"/>
    <link rel="stylesheet" href="css/animate.css"/>
    <link rel="stylesheet" href="css/owl.carousel.css"/>
    <link rel="stylesheet" href="css/style.css"/>
    <style>
    /* Додаткові стилі для оформлення секцій */
    .header-area {
        background-color: #1f1f1f; /* Темний фон для шапки */
        padding: 20px;
    }
    .content {
        padding: 50px 0;
        text-align: center;
        background-color: #121212; /* Темний фон для контенту */
        color: #fff; /* Білий текст для контрасту */
    }

```

```

.logout-button, .search-button, .add-apartment-button, .view-
contracts-button {
    display: block;
    margin: 20px auto;
    background-color: #28a745; /* Темно-зелений фон для кнопок
*/
    color: #fff;
    padding: 10px 20px;
    text-decoration: none;
    border-radius: 5px;
}
.logout-button {
    background-color: #007bff; /* Синій фон для кнопки виходу */
}
.footer-section {
    background-color: #1f1f1f; /* Темний фон футера */
    color: #fff;
    padding: 50px 0;
}
.footer-section h2 {
    color: #fff;
}
.footer-section ul {
    list-style-type: none;
    padding: 0;
}
.footer-section ul li {
    margin-bottom: 10px;
}
html, body {
    height: 100%;
    margin: 0;
    padding: 0;
    background-color: #121212; /* Темний фон для всього сайту */
    color: #fff; /* Білий текст для всього сайту */
}
.wrapper {
    display: flex;
    flex-direction: column;
    min-height: 100vh;
}
.content {
    flex: 1;
}
.nav-menu ul {
    background-color: #1f1f1f; /* Темний фон меню навігації */
}
.nav-menu ul li a {
    color: #fff; /* Білий текст для меню навігації */
}
.nav-menu ul li a:hover {
    color: #28a745; /* Зелений текст при наведенні */
}

```

```

</style>
</head>
<body>
<div class="wrapper">
    <header class="header-area">
        <div class="container">
            <div class="row">
                <div class="col-md-6">
                    <a href="index.php" class="logo-area">
                        
                    </a>
                </div>
                <div class="col-md-6 text-right">
                    <div class="nav-switch">
                        <i class="fa fa-bars"></i>
                    </div>
                </div>
            </div>
        </div>
        <nav class="nav-menu">
            <ul>
                <li class="active"><a
href="index.php">Головна</a></li>
            </ul>
        </nav>
    </header>

    <section class="content">
        <div class="container">
            <div class="row">
                <div class="col-md-12">
                    <h1>Ласкаво просимо на головну сторінку</h1>
                    <?php
include_once 'database.php';
connectDB();

// Перевірка, чи користувач авторизований
if (isset($_SESSION['user_id'])) {
    $user_id = $_SESSION['user_id'];
    $sql = "SELECT * FROM users WHERE user_id =
'$user_id'";

    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        echo "<p>Ім'я користувача: " .
$row['username'] . "</p>";
        echo "<p>Email: " . $row['email'] .
"</p>";
        echo "<p>Створено: " .
$row['created_at'] . "</p>";
        echo "<a href='edit_profile.php'
class='edit-button'>Редагувати</a>";
    }
}

```

```

        // Перевірка типу користувача
        if ($row['user_type'] === 'landlord') {
            // Покажемо кнопку "Додати квартиру"
            echo "<a href='add_apartments.php'
class='add-apartment-button'>Додати квартиру</a>";
        } else {
            // Якщо не орендодавець, не
            показуємо цю кнопку
            echo "<p>Ви не маєте доступу до
функції додавання квартири.</p>";
        }
    } else {
        echo "Дані користувача не знайдені!";
    }
} else {
    echo "Ви не увійшли в систему!";
}

$conn->close();
?>
        <a href="logout.php" class="logout-
button">Вийти</a>
    </div>
</div>
</div>
</section>

<footer class="footer-section">
    <div class="container">
        <div class="row">
            <div class="col-md-4">
                <h2>Зв'яжіться з нами</h2>
                <ul>
                    <li>Головна вулиця, 23, Нью-Йорк</li>
                    <li>+546 990221 123</li>
                    <li>contact@industryalinc.com</li>
                </ul>
            </div>
        </div>
    </div>
</footer>

<script src="js/jquery-2.1.4.min.js"></script>
<script src="js/bootstrap.min.js"></script>
<script src="js/isotope.pkgd.min.js"></script>
<script src="js/owl.carousel.min.js"></script>
<script src="js/jquery.owl-filter.js"></script>
<script src="js/magnific-popup.min.js"></script>
<script src="js/circle-progress.min.js"></script>
<script src="js/main.js"></script>
</body>

```


</html>

ДОДАТОК В – Інструкція користувача

Запустимо безпосередньо сайт (рис. В-2).

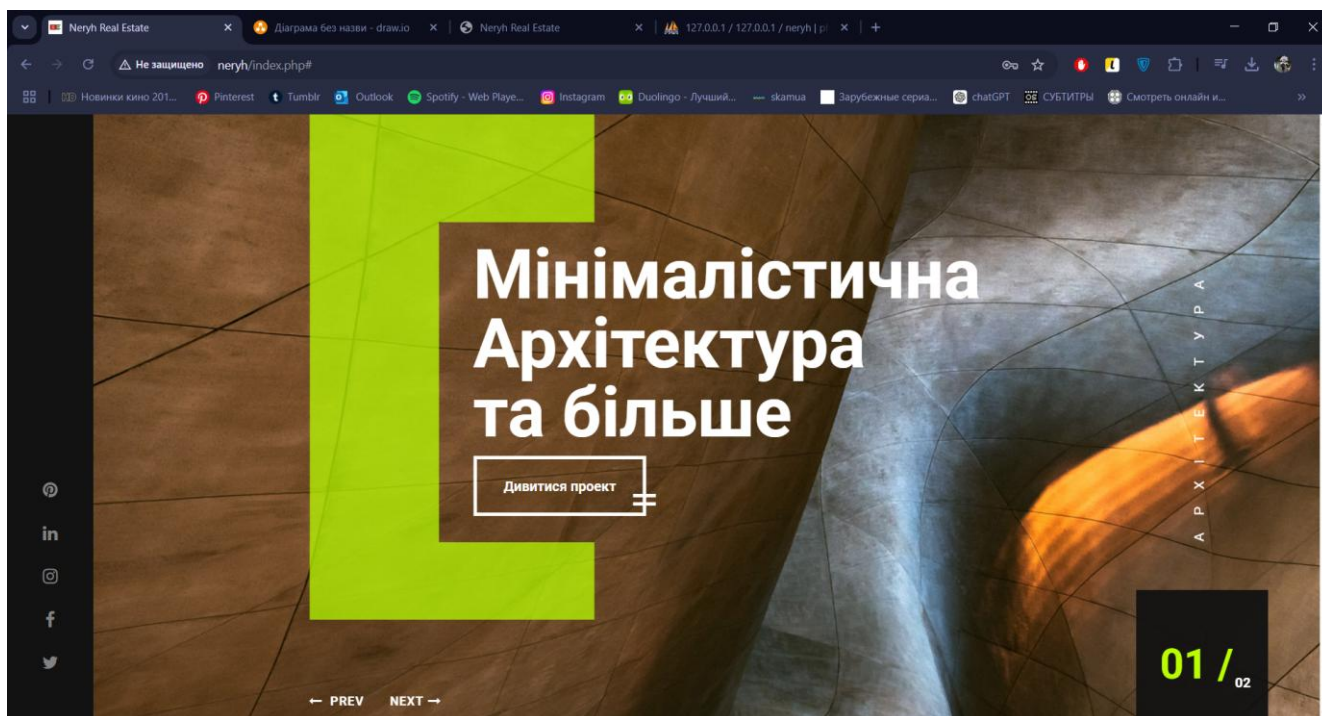


Рисунок В-2 – Запущенный сайт

Перейдемо в реєстрацію (рис.В-3), створимо орендодавця.

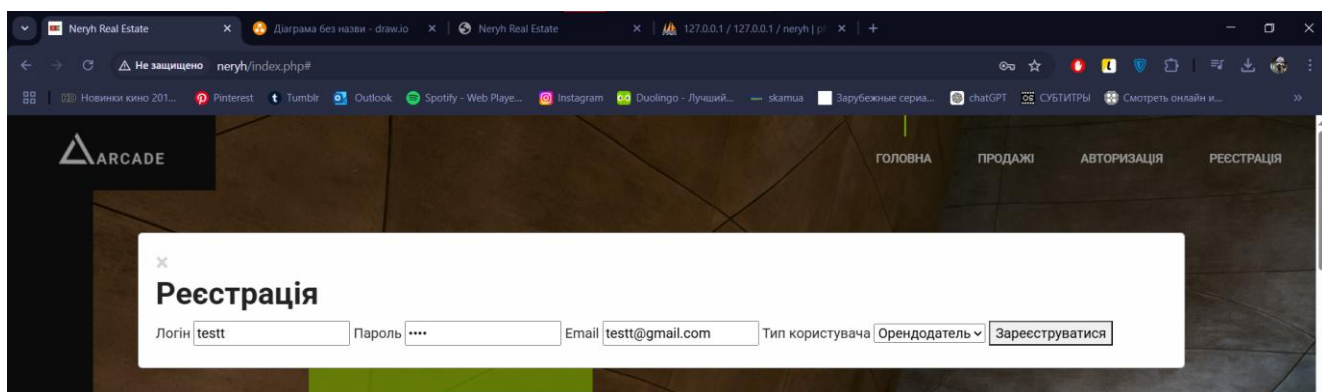


Рисунок В-3 – Створення орендодавця

Авторизуємося як орендодавець (рис.В-4).

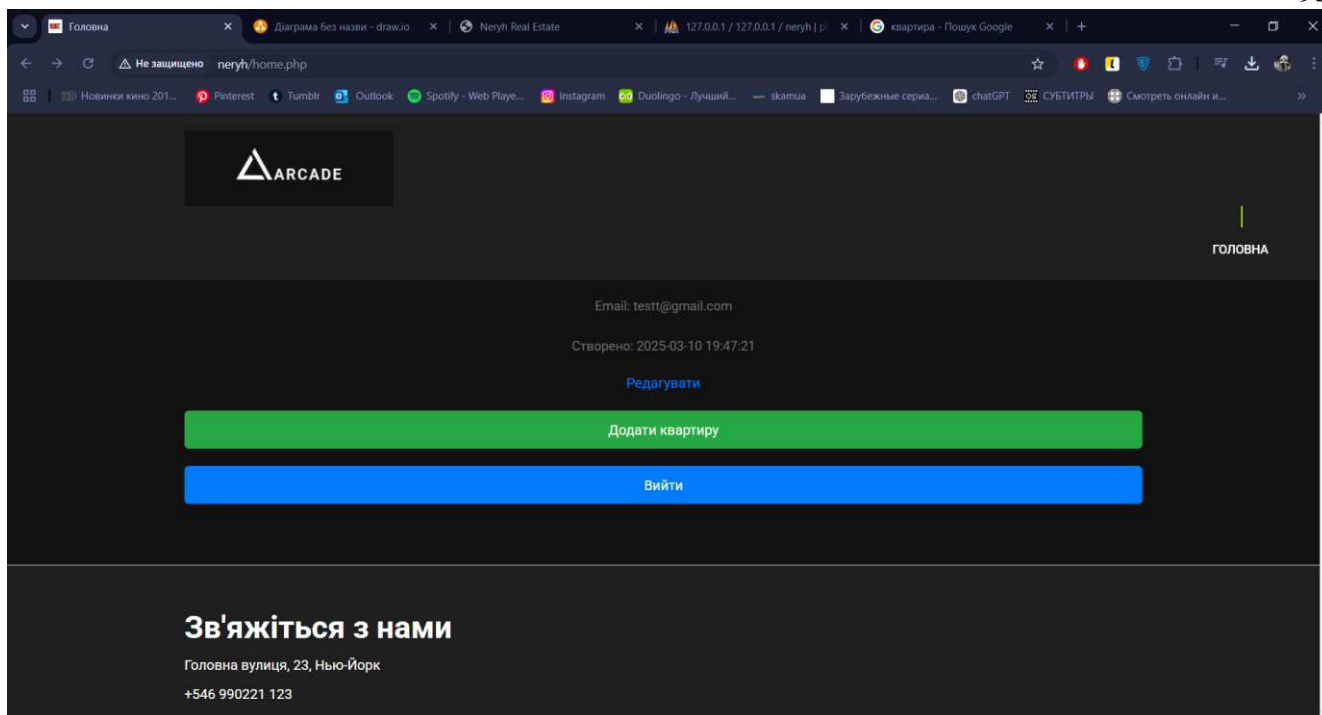


Рисунок В-4 – Авторизація як орендодавець

Додамо квартиру (рис.В-5).

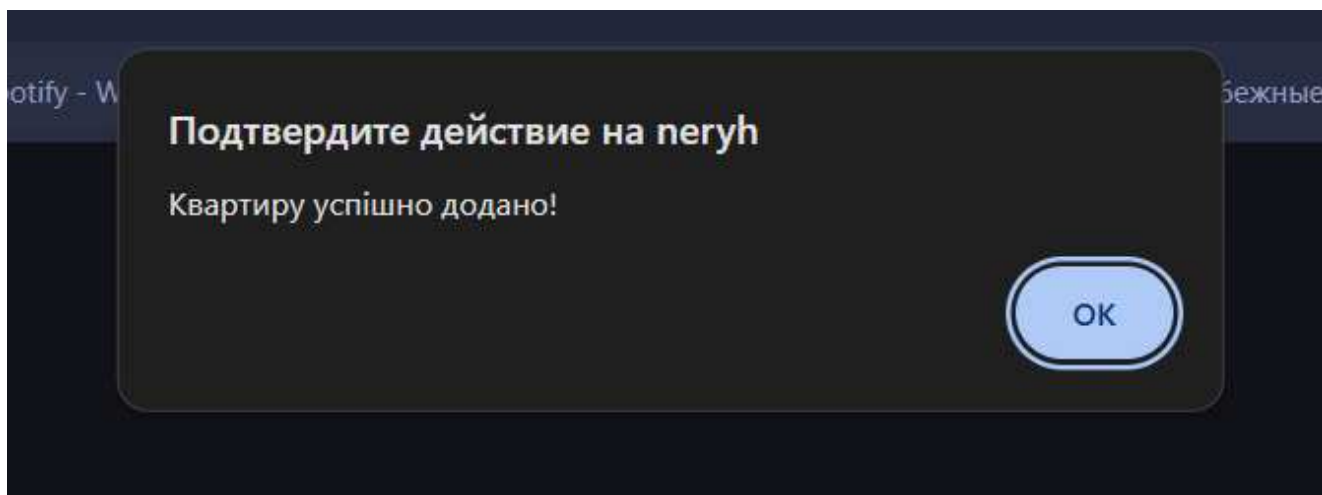


Рисунок В-5 – Додавання квартири

Переглянемо в продажах (рис.В-6).

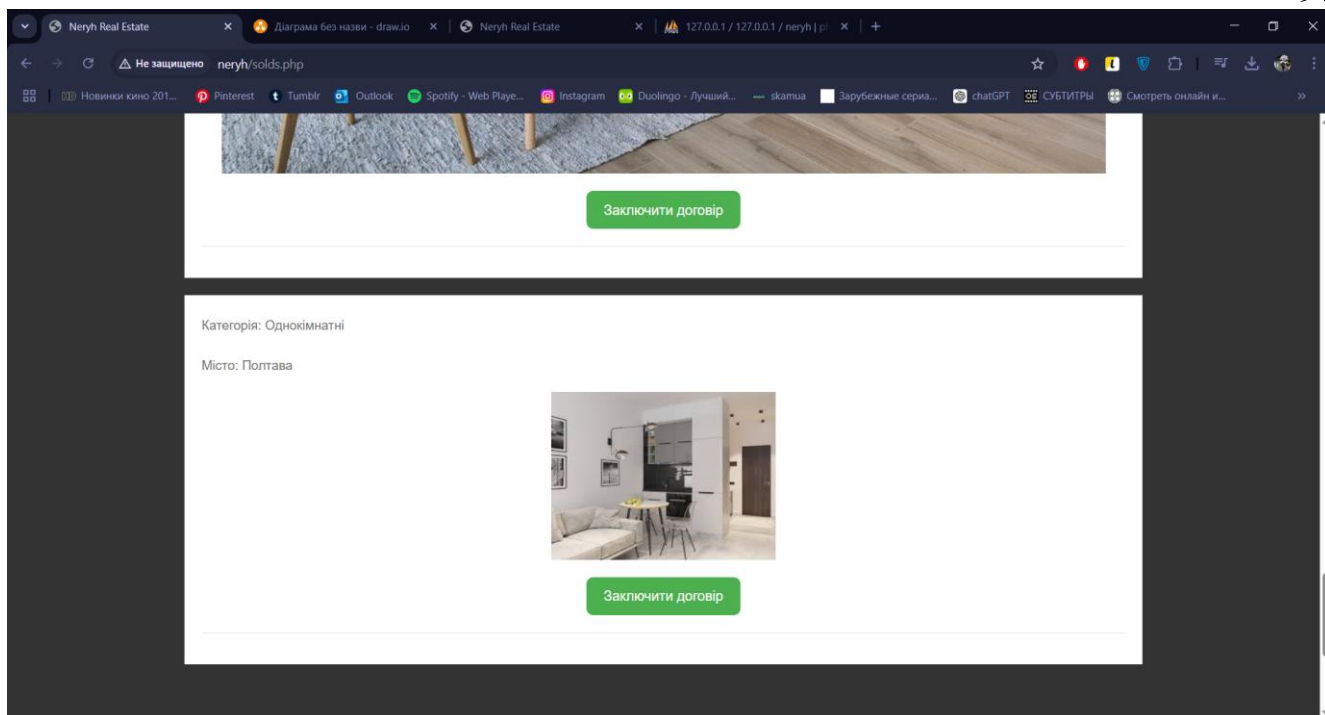


Рисунок В-6 – Нова квартира у продажах

Здійснимо оренду (рис. В-7).

Рисунок В-7 – Створення оренди

Перевіримо механізм фільтрації (рис.В-8-В-9).

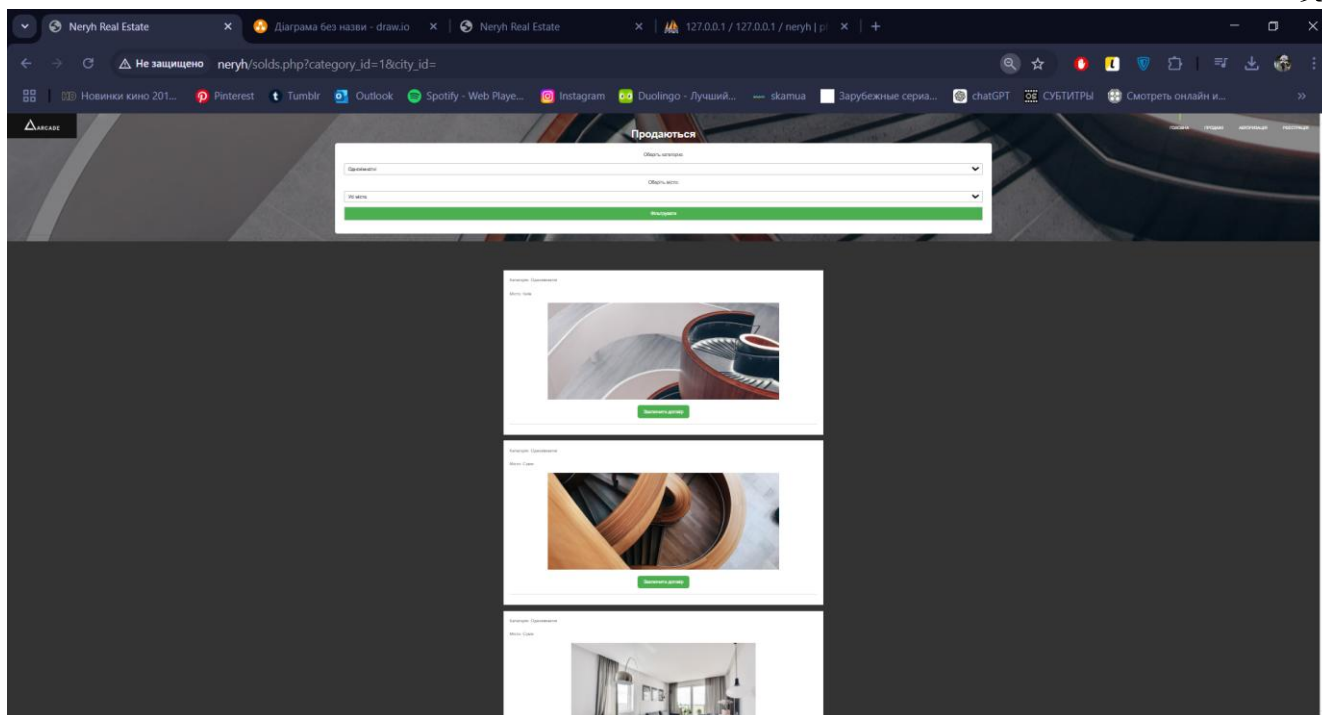


Рисунок В-8 – Фільтрація за типом квартири

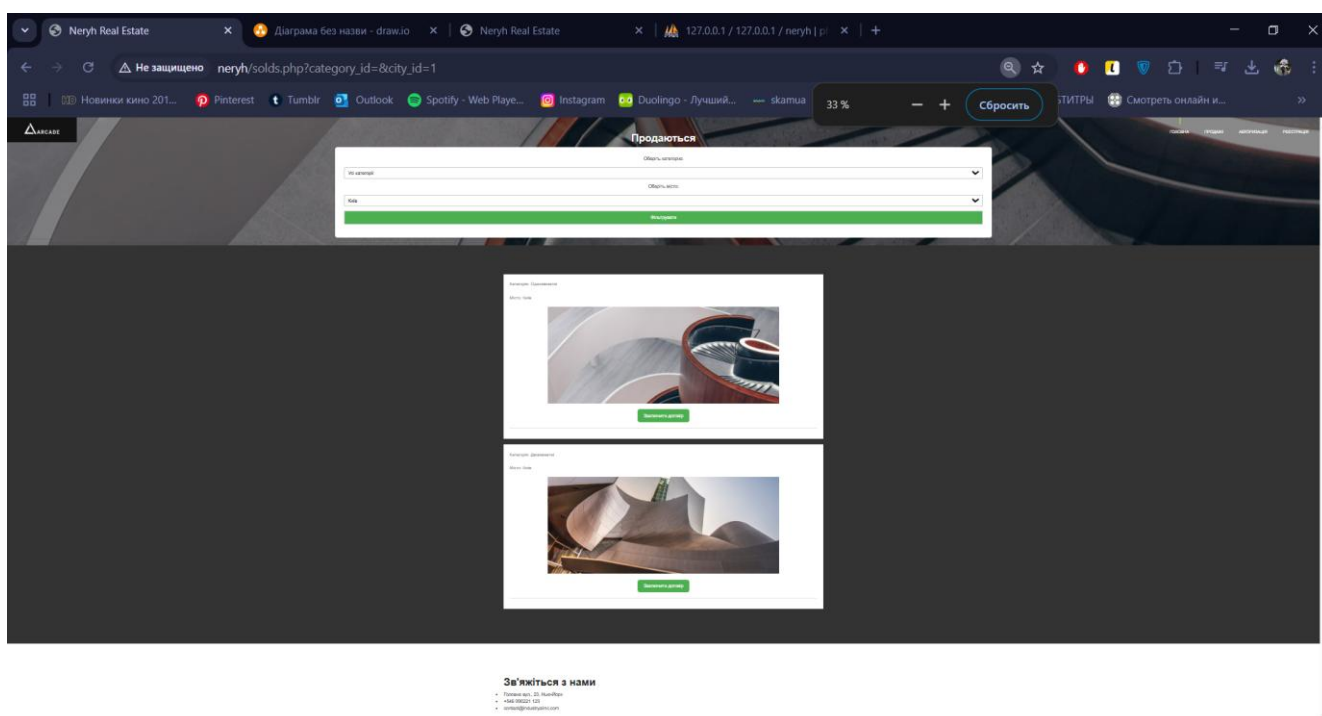


Рисунок В-9- Фільтрація за містом