МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Навчально-науковий інститут новітніх освітніх технологій
Кафедра інформаційно-обчислювальних систем і управління

ЯЦЮК Юлія Романівна

Рекомендаційна система вибору житла з використанням технологій машинного навчання/ Recommendation system for housing selection using machine learning technologies

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

Кваліфікаційна робота

Виконала студентка групи КН-42
Ю. Р. Яцюк

Науковий керівник:
к.т.н., доцент П.Є. Биковий

Кваліфікаційну роботу допущено до захисту
«___»_____ 2025 р.

В.о. завідувача кафедри
_____ Н.М. Васильків

Тернопіль – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
ЯЦЮК Юлія Романівна
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Рекомендаційна система вибору житла з використанням технологій машинного навчання/ Recommendation system for housing selection using machine learning technologies
керівник роботи П.Є. Биковий, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 20 грудня 2024 р. № 938.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2025 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:

- проаналізувати проблеми, з якими стикаються користувачі під час онлайн-пошуку житла, і визначити функціональні вимоги до системи, що має їх вирішити;
- дослідити існуючі рішення в галузі пошуку житла, зокрема сервіси Airbnb, Booking.com, OLX та Realtor.com, та виявити рівень персоналізації в них;
- розглянути підходи до побудови рекомендаційних систем (контентне, колаборативне, гібридне фільтрування) й обґрунтувати вибір методів для системи нерухомості;
- спроєктувати архітектуру веб-системи, включаючи інтерфейс користувача, базу даних, серверну логіку та модуль машинного навчання;
- реалізувати систему SmartHomeFinder з підтримкою пошуку, персоналізованих рекомендацій та автоматичного агента SearchAgent;
- забезпечити тестування системи на ефективність: оцінити точність рекомендацій, зручність інтерфейсу, швидкість знаходження релевантних об'єктів;

5. Перелік графічного матеріалу в роботі:

- контекстна DFD-діаграма системи;
- Інфологічна модель у вигляді ER-діаграми;
- UML-діаграма класів;
- UML-діаграма станів.

6. Консультанти розділів кваліфікаційної роботи

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

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2025 р.	

Студент_____ Ю.Р. Яцюк
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ П.Є. Биковий
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Рекомендаційна система вибору житла з використанням технологій машинного навчання» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» написана обсягом 47 сторінок і містить 15 ілюстрацій, 2 таблиці, 2 додатки та 27 використаних джерел.

Метою роботи є розроблення веб-системи SmartHomeFinder – рекомендаційної системи вибору житла із використанням технологій машинного навчання. Для реалізації системи використано Flask (мова Python) для розробки веб-застосунку, базу даних PostgreSQL для зберігання даних та алгоритми машинного навчання для формування рекомендаційної моделі.

Внаслідок виконання роботи створено веб-систему SmartHomeFinder, яка на основі алгоритмів машинного навчання надає користувачам персоналізовані рекомендації щодо вибору житла. Розроблену систему можна застосовувати на онлайн-платформах з пошуку нерухомості або в агентствах нерухомості для автоматизації та підвищення ефективності процесу добору житла відповідно до потреб користувачів.

Ключові слова: РЕКОМЕНДАЦІЙНА СИСТЕМА, ПОШУК ЖИТЛА, МАШИННЕ НАВЧАННЯ, ВЕБ-СИСТЕМА, НЕРУХОМІСТЬ, SMARTHOMEFINDER, FLASK, POSTGRESQL.

ANNOTATION

Qualification work on the topic «Recommendation system for housing selection using machine learning technologies» for the Bachelor's degree in specialty 122 «Computer Science», educational program «Computer Science», is written on 47 pages and contains 15 figures, 2 tables, 2 annexes and 27 sources.

The purpose of the work is to develop the SmartHomeFinder web system – a recommendation system for housing selection using machine learning technologies. For the implementation of the system, technologies such as the Flask microframework (Python) for web development, a PostgreSQL database for data storage, and machine learning algorithms for building the recommendation model were used.

As a result of the work, the SmartHomeFinder web system was developed, which provides users with personalized housing recommendations based on machine learning algorithms. The developed system can be applied on online real estate platforms or by real estate agencies to automate and improve the process of selecting appropriate housing for users.

Keywords: RECOMMENDATION SYSTEM, HOUSING SELECTION, MACHINE LEARNING, WEB APPLICATION, REAL ESTATE, SMARTHOMEFINDER, FLASK, POSTGRESQL.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API — Application Programming Interface (інтерфейс прикладного програмування)

HTTP — Hypertext Transfer Protocol (протокол передачі гіпертексту)

UML — Unified Modeling Language (уніфікована мова моделювання)

DFD — Data Flow Diagram (діаграма потоків даних)

JS — JavaScript (мова програмування для вебу)

CSS — Cascading Style Sheets (каскадні таблиці стилів)

HTML — HyperText Markup Language (мова розмітки гіпертексту)

SQL — Structured Query Language (структурована мова запитів)

DB — Database (база даних)

UX — User Experience (досвід користувача)

UI — User Interface (користувацький інтерфейс)

ML — Machine Learning (машинне навчання)

AI — Artificial Intelligence (штучний інтелект)

ВСТУП

На сучасному ринку нерухомості, який виступає об’єктом проєктування, виникає потреба у підвищенні ефективності процесу вибору житла. Існуючі платформи для пошуку нерухомості, як-от OLX, Dom.ria, Lun.ua, надають базову фільтрацію, однак не враховують поведінкову модель користувача. Це ускладнює прийняття рішень та знижує релевантність отриманих результатів.

У зв’язку з цим актуальною є розробка системи, що не лише здійснює пошук за параметрами, а й пропонує персоналізовані рекомендації на основі інтересів користувача. Таке рішення можливе завдяки впровадженню методів машинного навчання в інфраструктуру веб-системи.

Метою класифікаційної роботи є створення веб-системи SmartHomeFinder, яка надає персоналізовані рекомендації щодо вибору житла з використанням алгоритмів машинного навчання.

Для досягнення мети було поставлено такі завдання:

- проаналізувати потреби та проблеми в існуючих рішеннях;
- обрати та реалізувати ефективні підходи до побудови рекомендацій;
- спроєктувати архітектуру системи та реалізувати її веб-інтерфейс;
- впровадити модуль SearchAgent для автоматизованого пошуку;
- протестувати функціональність і точність рекомендаційної системи.

Система SmartHomeFinder має клієнт-серверну архітектуру. Веб-інтерфейс реалізовано з використанням HTML, CSS та JavaScript. Бекенд написаний мовою Python із застосуванням Flask. Дані зберігаються у базі PostgreSQL. Для реалізації рекомендаційного модуля використано методи машинного навчання. Інтерфейс підтримує розширений пошук, перегляд об’єктів, збереження вподобань та роботу з автоматизованими агентами.

До складу системи входить:

- веб-інтерфейс для взаємодії з користувачем;
- модуль рекомендацій з машинним навчанням;
- SearchAgent для моніторингу нових об’єктів;
- серверна логіка обробки запитів та управління даними;

- база даних для збереження профілів, фільтрів, об’єктів.

Сфера застосування: онлайн-платформи пошуку житла, агентства нерухомості, електронні маркетплейси. Система може бути інтегрована в існуючі сервіси з метою покращення взаємодії з користувачами.

Основні результати апробовані в межах наукової доповіді «Рекомендаційна система вибору житла як інструмент управління користувацькими рішеннями», представленої на конференції у секції «ШІ в управлінні проєктами».

1 АНАЛІЗ РЕКОМЕНДАЦІЙНИХ СИСТЕМ ВИБОРУ ЖИТЛА

1.1 Опис предметної області

Онлайн-пошук житла сьогодні є одним із ключових етапів процесу оренди чи купівлі нерухомості [1]. В сучасних умовах більшість користувачів розпочинають пошук нового дому саме через інтернет-сервіси. За статистикою, понад 93% покупців нерухомості звертаються до онлайн-порталів для збору інформації про потенційні об’єкти, причому більше половини угод з нерухомістю відбувається після того, як покупці знаходять об’єкт в інтернеті [21].

Однак, попри таку популярність цифрових платформ, сам процес пошуку житла залишається складним: пошук нерухомості входить до трійки найскладніших кроків на шляху до купівлі житла. Середня тривалість онлайн-пошуків перед тим, як користувач звертається до ріелтора, становить близько трьох тижнів. Це свідчить про наявність низки проблем та неефективностей у сучасних платформах для пошуку житла, які потребують вирішення.

Основні учасники процесу пошуку житла на онлайн-платформі – користувач, орендодавець та сама веб-система. Взаємодія між ними відбувається таким чином: користувач вводить певні критерії пошуку і отримує список доступних оголошень; платформа забезпечує фільтрацію та відображення результатів, а орендодавці надають інформацію про об’єкти та відповідають на запити.

Попри стрімкий розвиток інтернет-платформ, існують характерні проблеми, з якими стикаються користувачі під час пошуку житла онлайн.

Типова платформа може містити тисячі оголошень, через що користувачу важко швидко знайти потрібний варіант. Якщо інструменти фільтрації недосконалі, користувач змушений переглядати сотні нерелевантних пропозицій. Наприклад, на сайтах з обмеженими фільтрами процес пошуку може перетворитися на справжнє випробування: недостатня кількість критеріїв або неточні фільтри роблять пошук значно складнішим. У результаті користувач витрачає багато годин, переглядаючи зайві варіанти і відсіваючи більшість із них вручну. Дослідження підтверджують, що середньостатистичний покупець не переглядає далі першої сторінки або кількох сторінок результатів пошуку. Тому без

рекомендаційних механізмів існує ризик, що користувач просто не дійде до оголошення, яке найкраще відповідає його потребам.

Багато порталів нерухомості надають лише базові фільтри, які не враховують усіх потреб користувача. Якщо ж на сайті занадто мало критеріїв або вони надто загальні, користувач не може звузити вибір до прийнятного обсягу результатів. З іншого боку, навіть якщо доступно багато фільтрів, вони можуть не давати точних результатів через неповні дані у оголошеннях. Наприклад, далеко не всі портали дозволяють відфільтрувати житло за стилем інтер'єру чи наявністю специфічних зручностей, хоча для певних користувачів такі параметри критично важливі. У підсумку навігація стає виснажливою: родині, яка шукає будинок з конкретним набором характеристик, доводиться переглядати сотні фотографій і читати десятки описів, перш ніж знайти декілька придатних варіантів.

Сучасні платформи нерідко пропонують усім користувачам однаковий підхід до пошуку, не враховуючи їхню індивідуальну ситуацію чи вподобання. Сайт не "навчається" на діях користувача – наприклад, не запам'ятовує, яке житло той переглядав чи яке вподобав. Через це користувачу доводиться щоразу вручну налаштовувати фільтри і переглядати значний обсяг нерелевантної інформації. Пошуковий процес стає одноманітним і рутинним, що знижує залученість: досвід пошуку не приносить задоволення. Хоча люди зазвичай сповнені ентузіазму, обираючи нове житло, монотонний інтерфейс та відсутність «розумних» підказок можуть швидко їх знесилити. Як відзначають дослідники, пошук житла має бути цікавим і захопливим, а не нудним і стомлюючим процесом.

Перелічені проблеми вказують на необхідність удосконалення платформ з оголошеннями про нерухомість. Зокрема, актуальними є рішення, що зменшують навантаження на користувача, покращують навігацію та надають персоналізовані рекомендації. Вважається, що застосування сучасних технологій, таких як машинне навчання та штучний інтелект, може суттєво підвищити ефективність пошуку. Наприклад, алгоритми комп'ютерного зору здатні аналізувати фотографії в оголошеннях і автоматично визначати наявність певних характеристик для покращення фільтрації. Крім того, все більшого поширення набувають рекомендаційні системи, що вчаться на поведінці користувача та підлаштовують

результати під його вподобання [14]. Суть таких рішень – зробити так, щоб кожен користувач бачив саме ті варіанти житла, які йому найбільш підходять, без потреби переглядати сотні інших.

Персоналізація за допомогою машинного навчання розглядається як один із найефективніших способів покращити взаємодію користувача із сервісом нерухомості [13]. Аналізуючи дії відвідувача на сайті, система може прогнозувати, які об’єкти найбільше його зацікавлять, і рекомендувати їх автоматично. Дослідження в сфері e-commerce підтверджують, що рекомендовані системи підвищують задоволеність користувачів, надаючи більш точні та різноманітні пропозиції, які відповідають їхнім вподобанням. Іншими словами, персоналізовані підказки економлять час користувача і створюють відчуття, ніби платформа “розуміє” його потреби.

У підсумку, предметна область пошуку житла в інтернеті характеризується великим обсягом даних, активною взаємодією користувачів з системою та високими вимогами до зручності інтерфейсу. Подолання наявних проблем можливе шляхом впровадження сучасних технологій – зокрема, рекомендаційних систем на базі ML та інтелектуальних агентів, що автоматизують пошук нових оголошень для користувача. Далі в роботі розглянуто існуючі платформи та їх функціональні можливості, аби виявити найкращі практики і недоліки, на яких варто зосередитися при розробці власної системи.

1.2 Огляд і аналіз існуючих аналогів

На ринку існує багато онлайн-платформ, які надають можливості пошуку та підбору житла. Серед найбільш відомих міжнародних сервісів можна виділити Airbnb та Booking.com, а серед платформ для довгострокової оренди чи купівлі житла – класифайд-сайти на кшталт OLX Нерухомість та спеціалізовані портали як Realtor.com. Розглянемо коротко функціональні можливості та особливості цих аналогів, приділяючи увагу інструментам персоналізації і рекомендацій.

Airbnb – глобальна платформа для короткострокової оренди житла. Користувачам доступний зручний пошук за місцем і датами, численні фільтри,

інтерактивна карта, оцінки, відгуки, фотографії та позначки типу “Superhost”.

Платформа повинна використовувати алгоритми машинного навчання для персоналізованого сортування результатів: враховувати поведінку користувача і показувати йому релевантні варіанти. Також забезпечувати можливість надання рекомендацій, як-наприклад “Similar listings”.

Для ілюстрації сучасного інтерфейсу пошуку на Airbnb на рисунку 1.1 наведено копію екрану сторінки результатів: видно список пропозицій з фото та ціною, карту праворуч і панель фільтрів [23]. Пошуково-рекомендаційна система генерує до 99% бронювань, що свідчить про її ефективність.

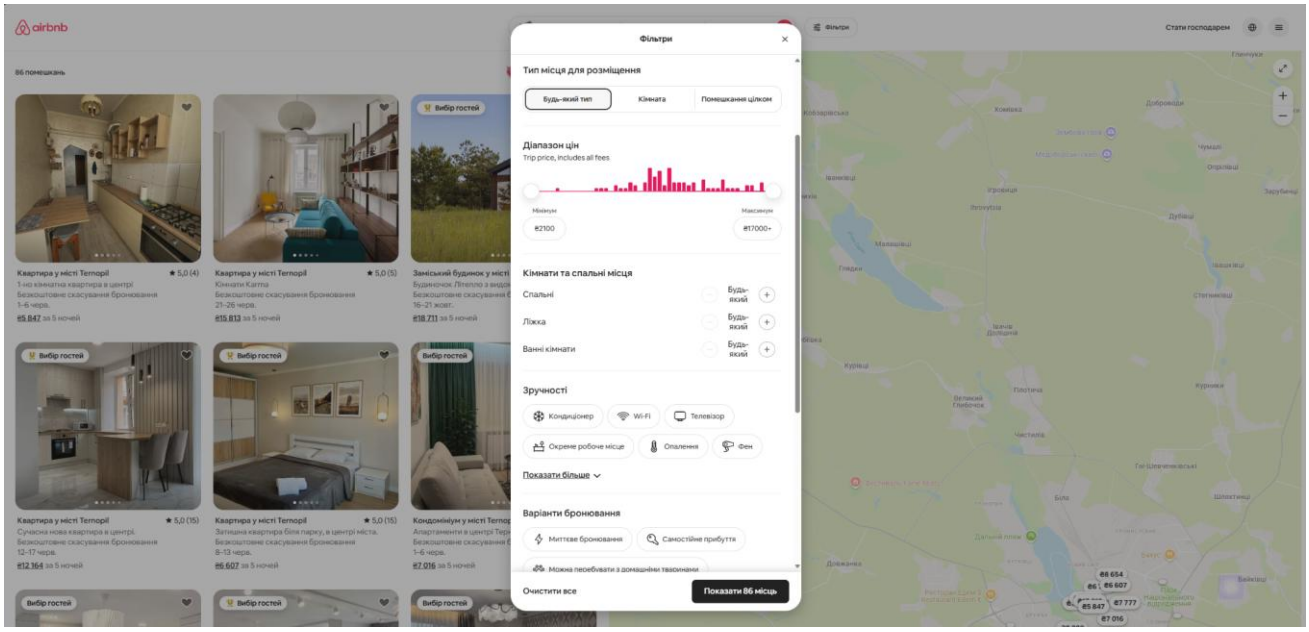


Рисунок. 1.1 – Приклад інтерфейсу пошуку на платформі Airbnb

Booking.com є одним із провідних світових сервісів для бронювання готелів, апартаментів та іншого житла [23]. Платформа надає користувачам широкі можливості для пошуку та вибору: доступна фільтрація за ціною, рейтингом гостей, розташуванням, зручностями, типом об’єкта тощо. Крім того, Booking має систему відгуків, програму лояльності та інструменти для порівняння цін.

Сервіс активно впроваджує персоналізацію: результати пошуку за замовчуванням сортуються за критерієм “Наші рекомендації”, який враховує популярність об’єкта, його рейтинг, відповідність запиту та поведінку користувача. При цьому використовується підхід колаборативного фільтрування, коли система аналізує уподобання схожих користувачів і на цій основі пропонує додаткові

варіанти. Персоналізовані елементи інтегровані у сам пошук і в email-розсилки з добірками житла, яке може зацікавити конкретного користувача.

OLX — один із найпопулярніших ресурсів в Україні для пошуку житла [21]. Це класифікований розділ оголошень, де користувачі самостійно розміщують пропозиції оренди чи продажу. Інтерфейс платформи дозволяє здійснювати базовий пошук за ключовими словами та застосовувати стандартні фільтри. Результати сортуються за обраним критерієм, наприклад, за датою або ціною.

Система не реалізує персоналізованого ранжування або рекомендацій. Усі користувачі бачать однаковий список результатів за одним і тим самим запитом. Проте зареєстровані користувачі можуть зберігати пошукові запити та підписуватись на автоматичні сповіщення про появу нових оголошень. Це дозволяє автоматизувати процес моніторингу без складної аналітики чи адаптації до поведінки користувача.

Realtor.com — відомий американський сайт для пошуку житла з великим набором фільтрів і додаткової інформації [22]. Платформа надає інтерактивну карту, на якій можна вручну позначати бажану зону пошуку. Користувачам доступні додаткові інформаційні шари на карті, наприклад, рівень цін у районі, наявність шкіл поблизу, рівень безпеки.

На платформі Realtor.com відсутня повноцінна рекомендаційна система, яка б аналізувала поведінку користувача, здійснювала глибоку персоналізацію та автоматично пропонувала релевантні варіанти на основі його інтересів або попередньої активності. Проте сайт надає базовий функціонал, який частково виконує роль рекомендаційної підтримки. Зокрема, користувач може зберігати параметри пошуку — обраний район, ціновий діапазон, кількість кімнат — і надалі отримувати електронні сповіщення про появу нових об’єктів, що відповідають цим умовам. Це дозволяє не пропустити потенційно цікаві варіанти без потреби щоденного моніторингу вручну. Такий механізм нагадує роботу простого агента пошуку, однак без складної логіки адаптації до змін у вподобаннях користувача.

Для узагальнення ключових характеристик і можливостей згаданих платформ, пов’язаних із пошуком і рекомендаціями, інформацію зведено у таблицю 1.1.

Таблиця 1.1 – Порівняльна характеристика платформ пошуку житла

Платформа	Функціональні можливості	Наявність рекомендацій
Airbnb	Пошук короткострокового житла по всьому світу; гнучкі фільтри; відгуки гостей; інтерактивна карта результатів.	Персоналізоване сортування результатів за допомогою ML; блоки схожих пропозицій на сторінках оголошень.
Booking.com	Бронювання готелів і апартаментів; багато критеріїв пошуку; карта; відгуки; програма лояльності.	Персоналізована видача з урахуванням поведінки користувача і статистики схожих клієнтів; рекомендації напрямків на основі історії пошуків; email-розсилки з пропозиціями.
OLX	Класифайд для оренди та продажу нерухомості; стандартні фільтри; прямий контакт з авторами оголошень; безкоштовне розміщення оголошень.	Відсутня ML-персоналізація; користувачі отримують однакові результати при однакових запитах. Є можливість зберігати пошук і отримувати сповіщення про нові оголошення за заданими критеріями.
Realtor.com	Пошук житла; дуже багато фільтрів; розширена карта; аналітичні дані.	Явна рекомендаційна система відсутня. Натомість – збереження пошукових налаштувань і email-сповіщення про появу нових відповідних об’єктів; демонстрація схожих об’єктів на базі атрибутів.

Жодна з наявних платформ не вирішує всі проблеми пошуку житла. Airbnb та Booking.com впровадили персоналізацію, але зосереджені на короткотерміновій оренді. OLX популярний в Україні [21], проте не пропонує інтелектуального підбору – пошук здійснюється вручну. Realtor.com має розширені фільтри, але не використовує машинне навчання для рекомендацій. Це створює передумови для розробки власної веб-платформи, що поєднує переваги аналогів і доповнюється сучасною рекомендаційною системою.

1.3 Постановка задачі дослідження

Мета роботи – розробити веб-платформу для пошуку житла з інтегрованою рекомендаційною системою на основі технологій машинного навчання. Ця система повинна забезпечувати користувачам можливість швидко знаходити релевантні варіанти житла відповідно до їхніх потреб та вподобань, мінімізуючи ручне переглядання зайвих оголошень. Проєкт передбачає створення не лише стандартного функціоналу пошуку та фільтрації, а й впровадження персоналізованого підбору об’єктів і автоматизованого агента, що інформуватиме користувача про нові цікаві пропозиції.

Відповідно до поставленої мети були сформовані основні задачі проєкту. Спочатку необхідно дослідити процес пошуку житла онлайн, виявити проблеми існуючих рішень та сформувати вимоги до функціоналу системи. Визначити, які параметри житла і критерії відбору є найбільш важливими для користувачів, і мають бути підтримані. Обґрунтувати доцільність впровадження рекомендаційної системи з точки зору підвищення зручності для користувача.

Далі потрібно вивчити різні підходи до побудови рекомендаційних систем і обрати ті, що найбільше підходять для задачі підбору нерухомості. Особливу увагу звернути на врахування специфіки доменної області: геолокація об’єктів, численні категоричні атрибути, важливість зображень тощо. Розглянути можливість застосування алгоритмів кластеризації або ранжування оголошень за релевантністю.

Розробити загальну архітектуру платформи: клієнтська частина, серверна частина. Спланувати структуру даних для зберігання інформації про оголошення і профілі користувачів та їхні уподобання. Передбачити інтеграцію модуля машинного навчання в загальну систему.

Створити інтерфейс, що дозволить користувачу здійснювати фільтрацію оголошень за основними параметрами. Забезпечити зручну навігацію результатами: відображення списку та карти, сортування за різними критеріями. Інтерфейс має бути інтуїтивно зрозумілим і не перевантажувати користувача.

Створити алгоритм, що буде генерувати персональні рекомендації житла для

користувача. Це може включати: (a) рекомендації подібних об’єктів при перегляді конкретного оголошення; (b) персональну вибірку нових або актуальних оголошень на головній сторінці на основі історії переглядів або вподобань. Модель повинна навчатися на даних про взаємодію користувачів із системою і виявляти приховані патерни – наприклад, розуміти, що користувач віддає перевагу певному району або типу планування, навіть якщо він це явно не фільтрував.

Реалізувати компонент, який буде моніторити появу нових оголошень за заданими користувачем критеріями або на основі профілю переваг, і надсилати повідомлення. Такий агент працюватиме постійно у фоновому режимі, щоб користувач не втрачав цікаві пропозиції, що з’явилися після його останнього сеансу пошуку. Наприклад, якщо користувач активно шукає 2-кімнатну квартиру у центрі міста, агент раз на день або в режимі реального часу відправлятиме добірку нових оголошень, що відповідають цьому запиту. Подібна функціональність присутня в деяких сервісах, але у нашому проєкті пропонується поєднати її з рекомендаційним підходом – тобто агент може пропонувати не тільки точні відповідності фільтрам, а й об’єкти, які алгоритм вважає потенційно цікавими для користувача.

Останнім завданням є перевірка розробленої системи на реальних або змодельованих даних. Потрібно оцінити, наскільки впроваджена рекомендаційна система полегшує користувачам пошук: можна виміряти скорочення часу на знаходження підходящого варіанту, підвищення клікабельності рекомендованих об’єктів, коефіцієнт конверсії, рівень задоволеності користувачів. Особливу увагу приділити ситуаціям, коли звичайний пошук дає дуже багато результатів – чи допомагають рекомендації знайти «голку в копиці сіна». Якщо можливо, порівняти результати роботи системи з наявними аналогами або з сценаріями, коли рекомендації вимкнені.

Отже, розробка рекомендаційної системи вибору житла є своєчасною і затребуваною. Поєднання класичних підходів з інтелектуальними технологіями дозволить створити якісно новий користувацький досвід у сфері пошуку нерухомості. Така система зменшить навантаження на користувача, прискорить процес підбору оптимального житла і підвищить ефективність роботи сервісу загалом.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1 Загальна структура проектованої системи

У роботі розроблено рекомендаційну систему вибору житла з використанням технологій машинного навчання. Загальна архітектура цієї системи має модульну побудову та включає кілька взаємопов'язаних компонентів.

Користувацький інтерфейс – відповідає за взаємодію з користувачем, надає засоби введення пошукових параметрів та відображає результати рекомендацій. Це може бути веб-інтерфейс або мобільний додаток, через який користувач вводить свої вподобання і переглядає рекомендовані об'єкти нерухомості.

Рекомендаційний модуль – ядро системи, яке реалізує бізнес-логіку і алгоритми машинного навчання для формування персоналізованих рекомендацій. Він отримує запити та дані від інтерфейсу, обробляє їх, запускає алгоритм рекомендації та формує список найбільш релевантних об'єктів. Рекомендаційний модуль містить налаштовану модель машинного навчання, яка на основі зібраних даних визначає рейтинг або вагу кожного потенційного варіанта.

База даних – централізоване сховище інформації про доступні об'єкти житла, користувачів та пов'язані дані. У базі даних зберігаються властивості об'єктів нерухомості, дані про користувачів та інша службова інформація. База даних забезпечує доступ рекомендаційного модуля до актуальної інформації та підтримує запити на вибірку даних за заданими критеріями.

Адміністративний модуль – призначений для керування даними системи. Зокрема, адміністративна частина дозволяє уповноваженим особам додавати або оновлювати інформацію про об'єкти житла в базі даних, стежити за актуальністю даних, керувати обліковими записами користувачів. Цей модуль може бути реалізований як окремий інтерфейс або інструмент адміністрування і безпосередньо взаємодіє з базою даних.

Усі згадані компоненти взаємозв'язані між собою та функціонують у єдиному програмному середовищі. Взаємодія між модулями відбувається наступним чином: користувач через інтерфейс формує запит. Інтерфейсний модуль передає ці дані до рекомендаційного модуля, який аналізує запит, звертається до

бази даних для вибірки кандидатних об’єктів та застосовує алгоритм машинного навчання для ранжування або відбору найкращих результатів. Після цього результати повертаються до модуля інтерфейсу і відображаються користувачу у зручному вигляді.

На рисунку 2.1 показано загальну архітектуру системи, яка візуалізує всі основні компоненти та їхню взаємодію. Зокрема, зображено, як користувач взаємодіє з веб-інтерфейсом, що передає запит до серверного застосунку, реалізованого на Flask. У цьому серверному модулі відбувається обробка запиту, виклик бізнес-логіки рекомендаційної системи та звернення до бази даних PostgreSQL для отримання необхідної інформації. Після обробки даних сервер повертає результати у вигляді рекомендацій, які відображаються користувачу. Така структура забезпечує чітке розділення відповідальностей між модулями та спрощує масштабування або модифікацію окремих компонентів у майбутньому.

Рисунок 2.1 – Структура архітектури системи SmartHomeFinder

На DFD-діаграмі класів (рисунок. 2.2) виділено ключові об’єкти системи та асоціації між ними. Класи характеризуються атрибутами і операціями, однак у рамках проєктування інформаційної системи основна увага приділяється саме атрибутам і зв’язкам, що згодом ляжуть в основу структури бази даних.

Рисунок 2.2 – Контекстна DFD-діаграма системи

Для формалізованого опису структури та поведінки проєктованої системи побудовано об’єктну, функціональну та динамічну моделі. Об’єктна модель представлена у вигляді діаграми класів, яка відображає основні класи предметної області та зв’язки між ними.

На рисунку 2.3 представлено інфологічну модель у вигляді ER-діаграми з п’ятьма сутностями: Users (ID, ім’я, email, пароль), Properties (ID, заголовок, опис, ціна, площа, кімнати, адреса, зовнішні ключі на Cities і Property type), Cities (ID, назва), Property type (ID, назва, тип), Favorites (UserID, PropertyID, Address, Date, Rating) — зв’язок «багато до багатьох» між користувачами та оголошеннями.

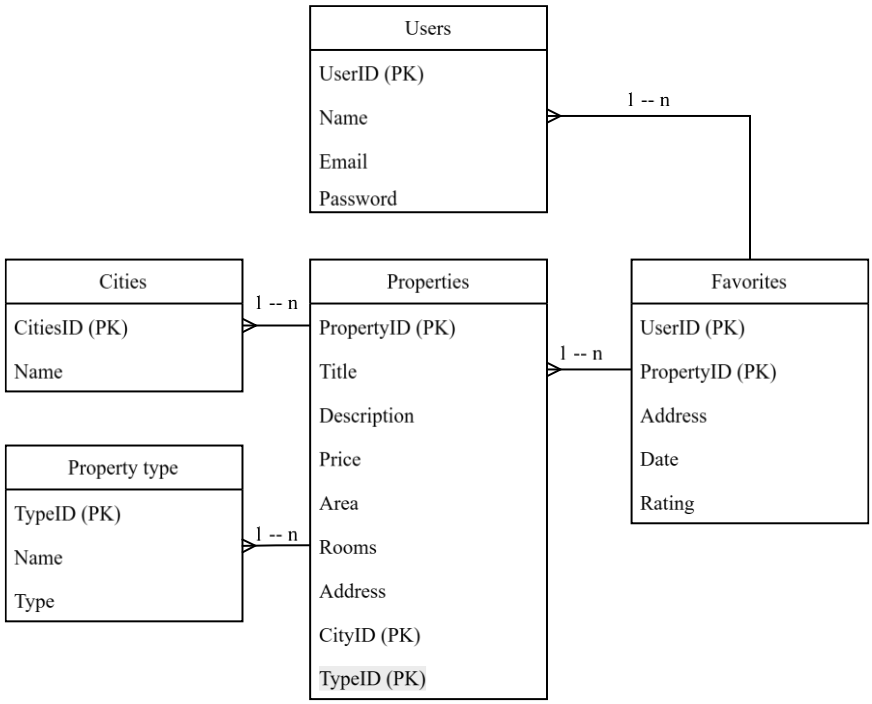


Рисунок 2.3 – Інфологічна модель у вигляді ER-діаграми

Отже, загальна структура проєктованої системи представлена сукупністю модулів, що реалізують визначені функції, та набором моделей, що описують архітектуру, інформаційні потоки і поведінку системи. Це забезпечує цілісне бачення системи «Рекомендаційна система вибору житла» перед безпосередньою реалізацією її алгоритмів і баз даних.

2.2 Алгоритмічне забезпечення

Алгоритмічне забезпечення описує логіку функціонування розробленої системи та послідовність дій для досягнення її основної мети – надання користувачеві персоналізованих рекомендацій щодо вибору житла. У моїй роботі розроблено узагальнений алгоритм функціонування системи, який охоплює всі основні етапи обробки запиту від його отримання до формування результату.

Алгоритм роботи рекомендаційної системи можна описати такими кроками:

1. Користувач запускає сеанс роботи з системою через інтерфейс. На цьому етапі проводиться автентифікація користувача та початкова ініціалізація необхідних структур даних для поточної сесії. Якщо користувач новий і в системі відсутня інформація про його вподобання, алгоритм враховує сценарій – для таких користувачів можуть застосовуватися загальні популярні рекомендації або вимагатися детальніші критерії відбору.

2. Користувач формулює свої вимоги до житла через інтерфейс. Вхідними даними можуть бути явні та неявні вподобання. Алгоритм отримує ці вхідні параметри та перевіряє їх на валідність і повноту. У випадку відсутності якихось важливих параметрів система може запросити уточнення у користувача.

3. Рекомендаційний модуль надсилає запит до бази даних для отримання початкового списку об’єктів нерухомості, що відповідають базовим критеріям користувача. На цьому етапі здійснюється фільтрація даних за жорсткими умовами: наприклад, вибираються всі оголошення, що відповідають зазначеному місту та діапазону цін, відфільтровуються варіанти з невідповідною кількістю кімнат чи іншими критичними параметрами. Якщо база даних не містить жодного об’єкта, що задовольняє початкові умови, алгоритм передбачає обробку цього

випадку: може бути запропоновано розширити критерії пошуку.

4. На відібраному наборі кандидатів виконується основний алгоритм машинного навчання, щоб визначити рейтинг або релевантність кожного об’єкта для конкретного користувача. У моїй роботі застосовано підхід, що поєднує елементи контентного та колаборативного фільтрування. Для кожного об’єкта нерухомості формується вектор ознак, а також враховується профіль користувача. Алгоритм обчислює оцінку відповідності – наприклад, прогноз ймовірності того, що даний користувач зацікавиться конкретним об’єктом. Це може бути реалізовано через модель машинного навчання, навченої на зібраних даних про взаємодію користувачів з об’єктами. Якщо користувач не має історії, система може більше покладатися на контентні ознаки та загальну популярність об’єктів.

5. Після обчислення рейтингових балів для всіх кандидатів, система сортує список об’єктів за цими балами у спадаючому порядку релевантності. Алгоритм відбирає топ-N об’єктів з найвищими рейтингами. Також, на цьому етапі може проводитися додаткова пост-обробка результатів: усунення дуже схожих об’єктів щоб урізноманітнити список, перевірка актуальності, тощо. Результатом кроку є сформований список рекомендованих об’єктів.

6. Відібрані рекомендації передаються від бізнес-логіки назад до користувацького інтерфейсу. Інтерфейс відображає користувачу список рекомендованих оголошень про житло у зручному форматі – наприклад, як перелік карток з фотографіями, ціною, коротким описом і посиланням на деталі. Користувач має змогу переглядати детальну інформацію по кожному з рекомендованих варіантів.

7. Після отримання рекомендацій користувач може здійснити певні дії, що слугують зворотним зв’язком для системи: наприклад, вибрати одне з рекомендованих оголошень для перегляду детально, додати вподобаний об’єкт до списку «Обране» або залишити оцінку. Ця інформація відстежується системою та може використовуватися для поліпшення майбутніх рекомендацій. Після завершення сесії користувач може вийти із системи; алгоритм переходить у кінцевий стан або очікує на новий запит, повертаючись до початкового стану готовності.

2.3 Інформаційне забезпечення

Інформаційне забезпечення має включати структуру та організацію даних, необхідних для реалізації її функцій. Воно охоплює опис вхідної та вихідної інформації, а також проєктування моделі даних на концептуальному, логічному та фізичному рівнях. Важливо, щоб інформаційна модель повністю відповідала потребам алгоритму рекомендації та забезпечувала цілісність і узгодженість даних. Основні функції системи передбачають обробку таких операцій:

а) генерація рекомендацій житла. Вхідні дані: параметри запиту користувача. Вихідні дані: персоналізований список рекомендованих об’єктів нерухомості, кожен з яких містить назву або заголовок, короткий опис, ціну, місце розташування та інші ключові характеристики.

б) пошук об’єктів нерухомості. Вхідні дані: фільтри пошуку за атрибутами. Вихідні дані: список знайдених об’єктів, що згідно заданих критеріїв. Ця функція схожа на рекомендаційну, але не враховує персональних вподобань користувача.

в) перегляд детальної інформації про об’єкт. Вхідні дані: ідентифікатор обраного об’єкта. Вихідні дані: повний набір інформації про цей об’єкт із бази даних, яка виводиться користувачеві.

г) управління вподобаннями користувача. Вхідні дані: дії користувача, що фіксують його вподобання. Вихідні дані: оновлені дані профілю користувача та повідомлення про результат операції. Наприклад, після додавання в “обране” оновлюється відповідний список у БД, а при зміні оцінки перераховується середній рейтинг об’єкта. На основі цих дій система формує внутрішній профіль користувача для майбутніх рекомендацій.

ґ) адміністрування каталогу нерухомості. Вхідні дані: інформація про нові або оновлені оголошення, які вводяться адміністратором системи. Вихідні дані: підтвердження успішного додавання або оновлення об’єкта та оновлений стан бази даних. Система перевіряє правильність введених даних і видає повідомлення про результат операції.

Таким чином, для кожної функції визначено набір необхідних вхідних даних і очікуваних виходів. Ці вимоги враховано при проєктуванні бази даних: структура

БД має забезпечувати збереження всіх вхідних параметрів і формування відповідних вихідних вибірок.

Концептуальне проєктування бази даних полягало у створенні інфологічної моделі предметної області «вибір житла». На цьому етапі визначено основні сутності та їх атрибути без прив’язки до конкретної СУБД. Було сформовано локальні моделі для різних аспектів системи: дані про користувачів і їх уподобання, дані про оголошення, довідкові сутності тощо. Після цього локальні моделі інтегровано в єдину інфологічну діаграму «сутність-зв’язок», яка відображає всі ключові сутності предметної області та їх зв’язки. Наприклад, зв’язок «Обране» демонструє відношення багато-до-багатьох між «Користувачем» і «Оголошенням», а кожне «Оголошення» пов’язане з одним «Містом» і одним «Типом_житла», при цьому кожне місто або тип житла може відповідати багатьом оголошенням.

У таблиці 2.1 визначено первинні ключі для унікальної ідентифікації записів та зовнішні ключі для забезпечення цілісності даних. Наприклад, поле CityID у таблиці Properties пов’язане з Cities.CityID зовнішнім ключем, а TypeID – з PropertyTypes.TypeID; поля UserID і PropertyID у Favorites посилаються відповідно на Users.UserID і Properties.PropertyID. Для кожного FK задано правила ON DELETE/ON UPDATE, що унеможливлює появу «висячих» записів.

Таблиця 2.1 - Структура таблиці Properties бази даних

Назва поля	Тип даних	Опис
PropertyID	INT PRIMARY KEY	Унікальний ідентифікатор оголошення
Title	VARCHAR(150)	Заголовок оголошення
Description	TEXT	Детальний опис об’єкта
Price	DECIMAL(10,2)	Ціна
Area	DECIMAL(5,1)	Площа
CityID	INT (FK)	Ідентифікатор міста
TypeID	INT (FK)	Ідентифікатор типу житла

Отже, розроблена фізична модель забезпечує ефективне зберігання та швидкий доступ до всіх необхідних даних системи. Використання реляційної СУБД MySQL гарантує підтримку транзакцій, індексацію та обмежень цілісності, що сприяє надійній роботі рекомендованого сервісу.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Реалізація програмного забезпечення

Розроблена рекомендаційна веб-система побудована за клієнт-серверною архітектурою. Клієнтська частина реалізована у вигляді веб-інтерфейсу, з яким взаємодіє користувач через браузер. Серверна частина створена на мові Python з використанням фреймворку Flask для обробки HTTP-запитів та формування відповіді. У якості системи управління базами даних використовується реляційна PostgreSQL, де зберігаються всі дані системи – інформація про користувачів, об’єкти нерухомості, вподобання, результати роботи пошукових агентів тощо.

Взаємодія компонентів відбувається наступним чином: користувач надсилає запит через веб-інтерфейс, запит передається на сервер, де обробляється бізнес-логікою додатка. Сервер звертається до бази даних PostgreSQL для отримання або збереження потрібних даних. Результати обробки формуються у HTML-сторінку і повертаються клієнту. Така розподілена архітектура забезпечує розмежування презентації та логіки обробки даних і рекомендацій (див. підрозділ 2.1).

У серверній частині реалізовано декілька основних бізнес-класів, що відповідають ключовим сутностям і функціям системи:

User – клас користувача. Містить ID, ім’я, email, пароль, а також список обраних об’єктів і налаштування. Реалізує методи реєстрації, автентифікації, управління «обраним». Відповідає таблиці users у БД.

Property – клас об’єкта нерухомості. Містить ID, заголовок, адресу, ціну, площу, кімнати, тип, опис та інші характеристики. Може включати методи форматування або обчислення. Відповідає таблиці properties.

SearchAgent – клас пошукового агента. Зберігає критерії пошуку користувача й автоматично виконує запити до БД або API, відбираючи нові оголошення. Має методи запуску пошуку та формування повідомлень. Відповідаабліці search_agents.

RecommendationEngine – клас, що формує рекомендації для користувача. Використовує дані про вподобання. Має метод get_recommendations(user) для видачі списку об’єктів, які можуть зацікавити. Працює з БД і алгоритмами ML або

внутрішнім ранжуванням.

Взаємозв’язки між бізнес-класами такі: користувач (User) може мати кілька пов’язаних об’єктів нерухомості через список обраного або історію переглядів; між User та Property реалізовано зв’язок "багато-до-багатьох" через окрему сутність Favorites – користувач може додати багато оголошень в обране, і одне оголошення може бути в обраному у багатьох користувачів. Користувач також може мати кілька пошукових агентів, кожен з яких налаштовується на певний сценарій пошуку. На UML-діаграмі класів (рисунок 3.1) графічно зображено перераховані класи, їх атрибути, методи та асоціації.

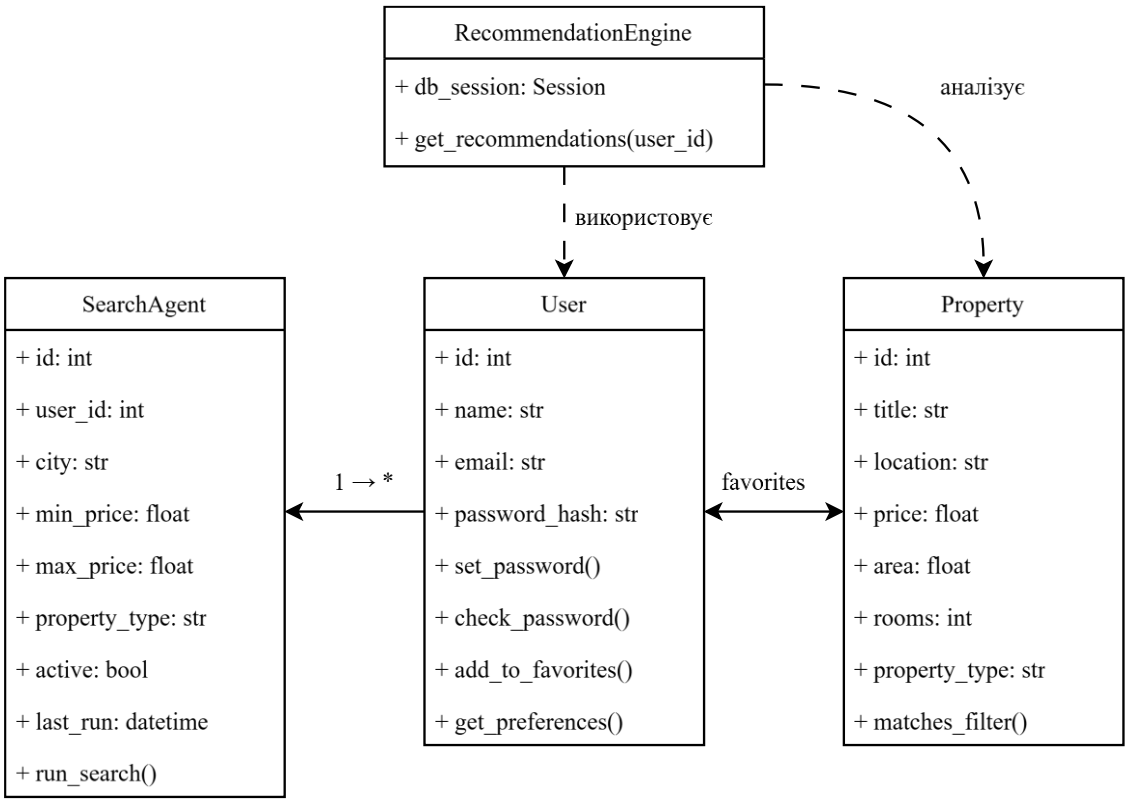


Рисунок 3.1 – UML-діаграма класів

Серверна частина реалізована мовою Python 3 з використанням мікрофреймворку Flask, який забезпечує маршрутизацію URL, обробку HTTP-запитів і рендеринг сторінок. У проєкті створено RESTful-API з маршрутами на кшталт /, /search, /property/id, /profile, які викликають відповідні функції контролера. Вони звертаються до бізнес-логіки та бази даних. Для зберігання даних використано PostgreSQL, яка підтримує складні SQL-запити. Зв’язок з базою забезпечує бібліотека psycopg2, а ORM SQLAlchemy дозволяє працювати з

таблицями як з Python-об’єктами, спрощуючи доступ до даних.

Для відображення поведінки системи в динаміці розроблено UML-діаграму станів (рисунок 3.2). На ній показано зміну станів основних компонентів/модулів під час роботи системи. Зокрема, ця діаграма ілюструє цикл роботи модуля SearchAgent: початковий стан, активний стан, стан повідомлення користувача про знайдені результати та повернення до стану очікування. Також відображено можливі стани для сесії користувача та стани життєвого циклу оголошення. Діаграма станів допомагає проілюструвати реакцію системи на дії користувача і внутрішні події.

Рисунок 3.2 – UML-діаграма станів

Для підтвердження працездатності реалізованої системи та демонстрації ключових аспектів її функціонування було створено програмний код, що охоплює основні компоненти: моделі даних, логіку рекомендацій та взаємодію з базою даних. Реалізація здійснена мовою Python із використанням фреймворку Flask, ORM-бібліотеки SQLAlchemy та PostgreSQL. Нижче наведено фрагменти коду, які ілюструють архітектурну та логічну побудову системи, а також принципи її роботи з даними.

У наведеному рисунку 3.3 клас User визначений як модель SQLAlchemy, що відповідає таблиці users. Він містить поля та методи для роботи з паролями: встановлення з хешуванням та перевірка. Також налаштовано зв’язок favorites – він вказує, що користувач має відношення "багато-до-багатьох" з моделлю Property через асоціативну таблицю favorites. Це дозволяє легко отримувати список обраних оголошень користувача через вираз user.favorites у коді.

```
class User(db.Model):  # SQLAlchemy model
    __tablename__ = 'users'
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    email = db.Column(db.String(100), unique=True, nullable=False)
    password_hash = db.Column(db.String(200), nullable=False)
    favorites = db.relationship('Property', secondary='favorites',
                                backref='liked_by')  # many-to-many relationship

    def set_password(self, raw_password):
        # Генерація гешу пароля
        self.password_hash = generate_password_hash(raw_password)

    def check_password(self, raw_password):
        # Перевірка пароля при вході
        return check_password_hash(self.password_hash, raw_password)
```

Рисунок 3.3 – Реалізація класу користувача (User)

На рисунку 3.4 показано реалізацію логіки рекомендацій: спочатку отримуються обрані об’єкти користувача. Якщо їх немає — система повертає останні додані в базу. Якщо вподобання є, система шукає "схожих користувачів", які також обрали хоча б один з цих об’єктів.

```
class RecommendationEngine:
    def __init__(self, db_session):
        self.db = db_session  # сесія доступу до БД (SQLAlchemy)

    def get_recommendations(self, user_id, limit=5):
        # Отримуємо вибрані об'єкти даного користувача
        user = self.db.query(User).get(user_id)
        favorite_props = set([prop.id for prop in user.favorites])
        if not favorite_props:
            # Якщо у користувача немає обраних об’єктів – рекомендуємо останні додані
            return self.db.query(Property).order_by(Property.id.desc()).limit(limit).all()

        # Знаходимо інших користувачів зі схожими вподобаннями (перетин обраних)
        similar_users = (self.db.query(User)
                         .join(User.favorites)  # join по таблиці favorites
                         .filter(Property.id.in_(favorite_props), User.id != user_id)
                         .all())
        # Збираємо всі об'єкти, які сподобались цим користувачам
        similar_fav_props = set()
        for u in similar_users:
            for prop in u.favorites:
                if prop.id not in favorite_props:
                    similar_fav_props.add(prop)
        # Якщо знайдено, повертаємо топ-рекомендації (наприклад, 5 об’єктів)
        recommendations = list(similar_fav_props)[:limit]
        return recommendations
```

Рисунок 3.4 – Реалізація базового алгоритму рекомендацій

Далі збираються інші об’єкти, вподобані цими користувачами, але відсутні у поточного — вони формують набір потенційних рекомендацій. Наприкінці обирається обмежений список для показу. Метод get_recommendations викликається контролером і передає результати у шаблон. Навіть у спрощеному вигляді ця логіка демонструє базовий принцип collaborative filtering.

У рисунку 3.5 демонструється, як застосунок працює з PostgreSQL через SQLAlchemy. Перший блок створює новий об’єкт нерухомості Property і додає його до сесії бази даних, після чого фіксує зміни – в результаті відповідний запис з’явиться в таблиці properties. Другий блок показує приклад використання критеріїв пошуку, подібний до того, що реалізує SearchAgent: формується запит до таблиці Property з фільтрацією по місту та максимальній ціні. Результатом є список об’єктів, що задовольняють умови; далі виводиться кількість знайдених оголошень і їх заголовки з ціною. Подібні запити використовуються для реалізації сторінки пошуку нерухомості: користувач задає критерії через форму, а сервер будує відповідний SQL-запит і відображає результати.

```
# Додавання нового оголошення нерухомості в базу
new_property = Property(title="Квартира на Центральній 5",
                        description="Двокімнатна квартира, 60 кв.м, новобудова",
                        city="Київ", price=50000, area=60, rooms=2, property_type="квартира")
db.session.add(new_property)
db.session.commit()

# Виконання пошуку об'єктів за критеріями (приклад SearchAgent)
city = "Київ"
max_price = 60000
result_properties = (db.session.query(Property)
                     .filter(Property.city == city, Property.price <= max_price)
                     .all())
print(f"Знайдено об'єктів у місті {city} до ${max_price}: {len(result_properties)}")
for prop in result_properties:
    print(f"- {prop.title}, ціна: ${prop.price}")
```

Рисунок 3.5 – Приклад роботи з базою даних

Таким чином, програмно-технологічна частина системи поєднує сучасний бекенд на Python, надійну PostgreSQL та класичний фронтенд. Усі основні компоненти інтегровані між собою відповідно до розробленої архітектури, що дозволило реалізувати повністю функціональну рекомендаційну систему вибору житла.

3.2 Інтерфейс користувача

Користувацький інтерфейс системи розроблено як веб-застосунок, доступний через браузер. Інтерфейс забезпечує зручну навігацію по основних розділах сайту та дає користувачеві можливість виконувати всі необхідні дії: пошук і перегляд об’єктів нерухомості, управління власним профілем, перегляд рекомендованих пропозицій тощо. Структура веб-сайту (див. додаток А) включає такі основні сторінки і розділи схему зображення структури:

На головній сторінці (рисунок 3.6) розміщено привітання, короткий опис сервісу та форма швидкого пошуку житла. Нижче відображаються популярні або нові оголошення. Для авторизованих користувачів також доступний блок “Рекомендовано вам” з персоналізованими пропозиціями. Із головної сторінки можна перейти до розширеного пошуку або перегляду конкретного об’єкта.

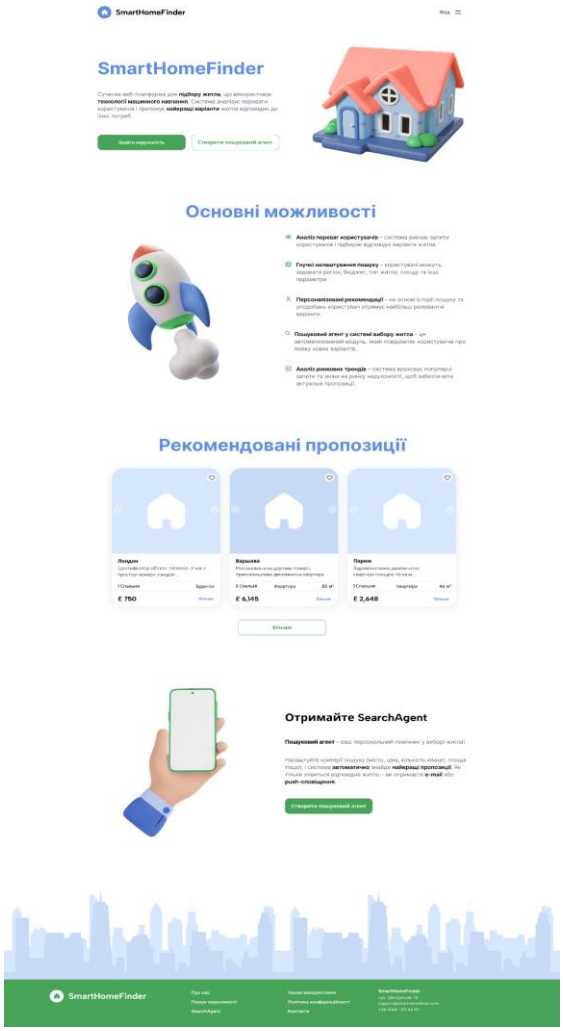


Рисунок 3.6 – Макет головної сторінки сайту

Сторінка пошуку нерухомості (рисунку 3.7). Ця сторінка відображається після того, як користувач здійснив пошук. Вона містить форму фільтрів у верхній частині та список результатів, що відповідають заданим умовам. Оголошення відображаються у вигляді списку або сітки карток: для кожного елемента показано основне фото, заголовок, короткий опис або ключові параметри, іконки чи мітки. Передбачено також пагінацію або динамічне підвантаження результатів, якщо їх багато. Користувач може сортувати результати і додавати вподобані об’єкти до “Обраного” прямо зі списку.

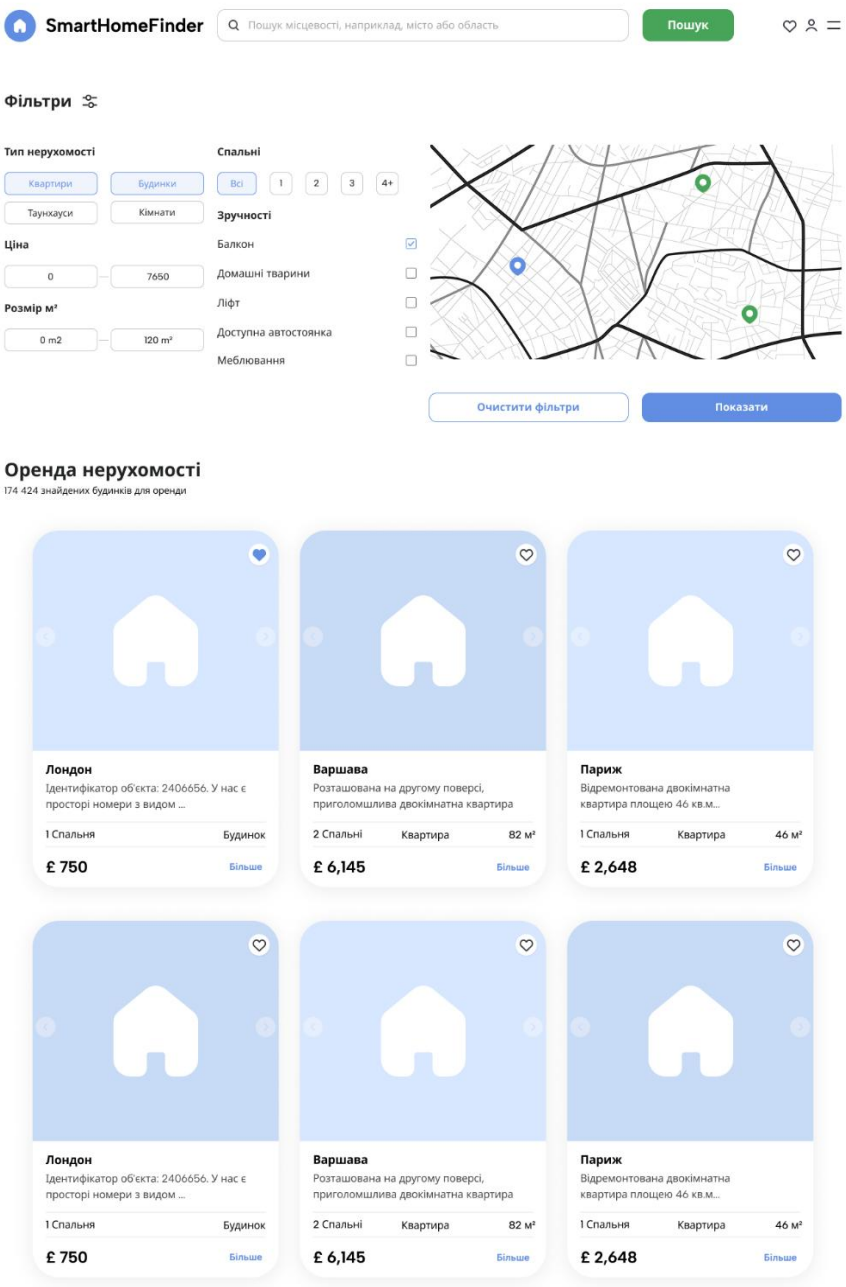


Рисунок 3.7 – Макет сторінки пошуку нерухомості

Сторінка деталізації об’єкта (рисунок 3.8). Коли користувач обирає конкретне оголошення нерухомості, відкривається окрема сторінка з повною інформацією про цей об’єкт. Тут відображаються всі фотографії, детальна інформація: опис, характеристики, ціна, контактні дані власника або ріелтора. Також є кнопка “Додати в обране” або “Видалити з обраного”, якщо вже додано, щоб користувач міг зберегти об’єкт. Інтерфейс цієї сторінки містить можливість переходу до наступного/попереднього результату пошуку без повернення назад. Додатково може відображатися блок “Схожі пропозиції” – кілька інших об’єктів, схожих за характеристиками.

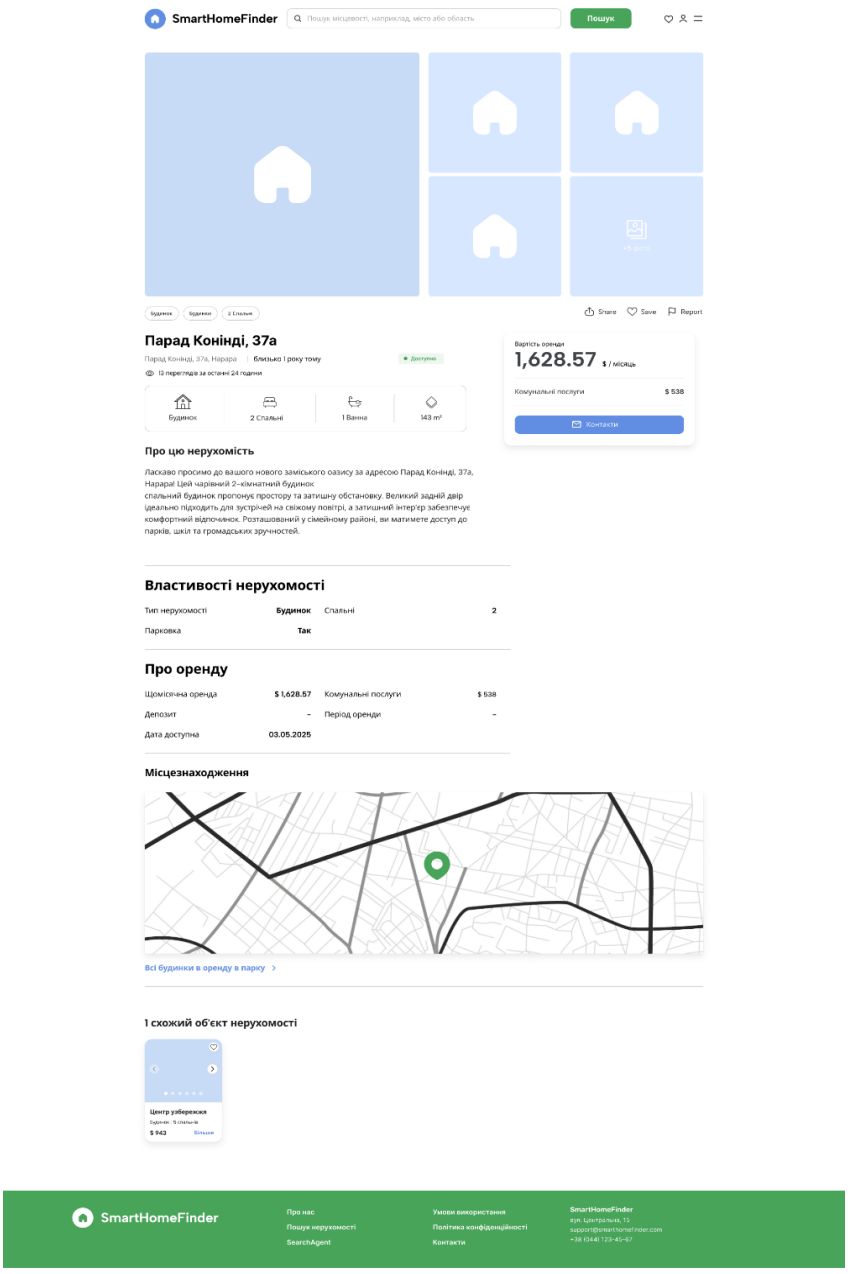


Рисунок 3.8 – Макет сторінки перегляду об’єкта нерухомості

Профіль користувача (рисунок 3.9) доступний після авторизації і містить особисті дані та кілька розділів. У “Мої обрані” відображається список об’єктів із таблиці favorites, кожен з яких можна переглянути або видалити. У розділі “Пошукові агенти” (рисунок 3.10) користувач створює, редагує та видаляє агенти з заданими критеріями пошуку, бачить їхній статус, дату останньої перевірки та кількість нових знайдених об’єктів.

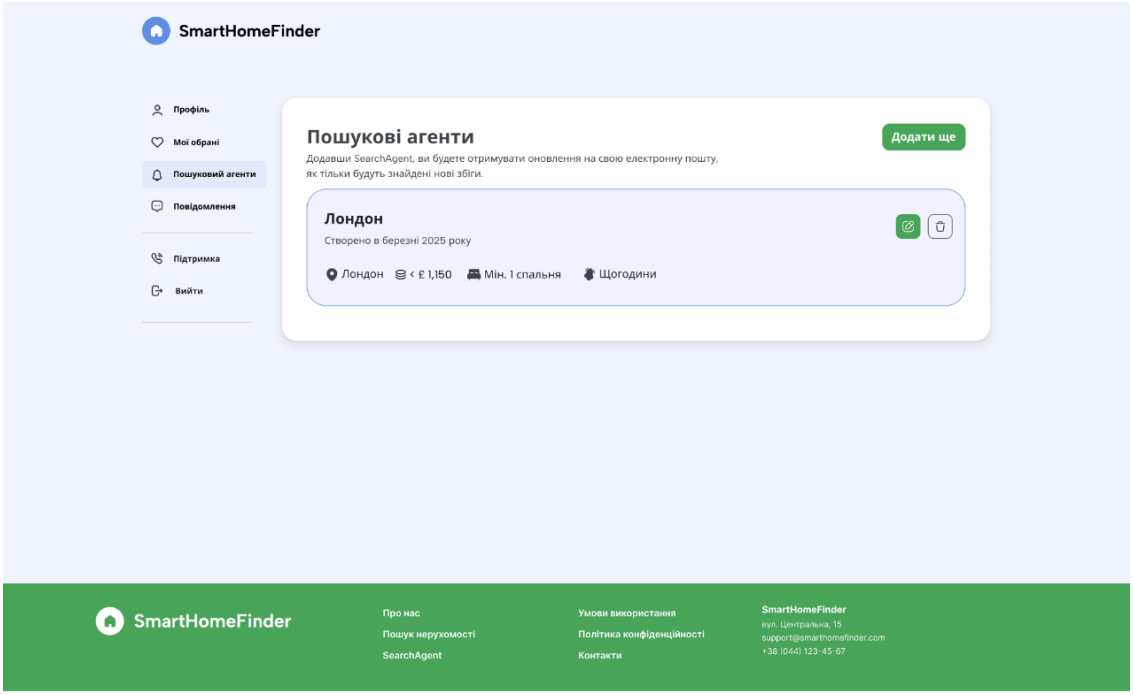


Рисунок 3.9 – Макет профілю користувача

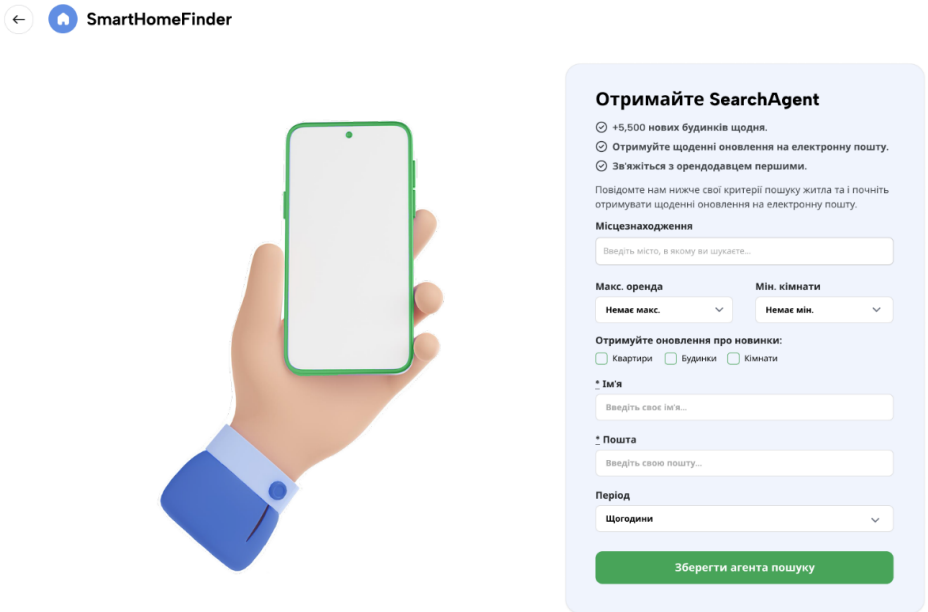


Рисунок 3.10 – Макет SearchAgent

Окрім основних, система має стандартні допоміжні сторінки та функції: Сторінка входу та реєстрації, де нові користувачі можуть створити акаунт, а існуючі – увійти. Сторінка про сервіс з інформацією про проект, контакти служби підтримки. Адміністративна панель, де адміністратор може додавати/редагувати оголошення, модерація відгуків тощо. Всі сторінки мають єдиний стиль оформлення і спільні елементи інтерфейсу – шапку сайту з логотипом і меню, та нижній колонтитул з контактами та копірайтом.

Розроблено прототипи і макети дизайну для вказаних сторінок, що забезпечило єдність стилю та зручність використання. Дивитися додаток А там показано приклади структури сайту. Дизайн витримано в спокійних тонах, з акцентами на важливі елементи. Використано сучасні підходи до верстки – адаптивний дизайн, що дозволяє веб-сторінкам коректно відображатися як на великих екранах десктопів, так і на мобільних пристроях.

При розробці інтерфейсу головну увагу приділено зручності користування та сучасним UX-принципам:

а) усі сторінки адаптивні — меню на смартфонах стає “гамбургером”, картки в одну колонку, шрифти і кнопки масштабуються для зручності;

б) зрозуміла навігація — головне меню у шапці з посиланнями на основні розділи, виділений заголовок сторінки, логотип повертає на головну;

в) єдині піктограми з підказками, уніфікована кольорова схема і шрифти для читабельності та професійного вигляду;

г) миттєвий зворотний зв’язок — повідомлення після дій, інформування про помилки, валідація форм, статус пошукових агентів.

Інтерфейс протестовано реальними користувачами, що підтвердило його зручність і сучасність, забезпечуючи приємний досвід вибору житла.На завершення, інтерфейс було протестовано реальними користувачами в режимі пробної експлуатації, і їхні відгуки підтвердили загальну зручність та привабливість дизайну. Завдяки дотриманню принципів UI/UX та адаптивності, веб-система виглядає сучасно і забезпечує позитивний досвід користувача при виборі житла.

3.3 Тестування розробленої системи

Розроблена система пройшла комплексне тестування, яке включало модульні тести, інтеграційні випробування та перевірку роботи інтерфейсу вручну. Метою тестування було впевнитись, що всі компоненти працюють згідно з вимогами, а також виявити і виправити можливі помилки перед фінальним впровадженням.

На етапі розробки бекенду для критичних функцій було створено набір модульних тестів. Тестувалися окремі методи класів та функції: зокрема, методи класу User, алгоритми RecommendationEngine, логіка пошукового агента та функції роботи з базою даних. Для написання тестів використовувався фреймворк pytest. Наприклад, було створено тест, який перевіряє генерацію рекомендацій для користувача з відомими вподобаннями: у тестовій базі даних попередньо додано кілька користувачів та об’єктів, встановлено їх взаємні "обрані". Після виклику RecommendationEngine.get_recommendations(test_user) перевіряється, що отриманий список містить очікувані об’єкти. Усі модульні тести успішно пройдені; при виявленні розбіжностей в очікуваному та фактичному результаті, код виправлявся. Так, на ранніх етапах тестування знайдено помилку: функція рекомендацій повертала дублікати об’єктів при певних вхідних даних. Цей дефект усунули шляхом використання множини (set) для уникнення повторень, що було підтверджено повторним проходженням тесту.

Після того, як окремі модулі були перевірені, проводилося тестування сценаріїв використання системи в цілому. Були складені тестові випадки для основних функціональних вимог, наприклад: “реєстрація нового користувача”, “вхід та вихід з системи”, “пошук об’єктів за різними критеріями”, “додавання об’єкта в обране”, “створення пошукового агента та отримання сповіщення”. Кожен сценарій виконувався вручну тестувальником або розробником у тестовому середовищі. Результати відповідали очікуванням:

- При введенні валідних даних новий користувач успішно створюється, з’являється повідомлення, а дані додаються в таблицю users.

- Пошук нерухомості працює коректно, включно з граничними випадками — система або повідомляє про помилку, або повертає 0 результатів без збою.

- Додавання до обраного працює: об’єкт з’являється у “Мої обрані”, іконка змінюється; повторне натискання видаляє об’єкт.

- Пошуковий агент коректно виконує запити до БД, знаходить нові оголошення, генерує сповіщення. Повторні сповіщення не створюються без нових результатів.

- RecommendationEngine рекомендує об’єкти відповідно до вподобань: очікувані результати підтверджено на сторінці профілю.

Окремо здійснювалося тестування веб-інтерфейсу в різних браузерах і на різних пристроях. Перевірялося, чи коректно відображаються всі елементи дизайну і чи працює адаптивність. В результаті такого тестування було внесено дрібні правки в CSS: наприклад, виправлено ширину деяких блоків на малих екранах, щоб текст не виходив за межі, збільшено розмір шрифту для кнопок на мобільних для зручності натискання. Також проводилось ручне тестування сценаріїв взаємодії з UI: натискання кнопок, заповнення форм, навігація меню. Це дозволило впевнитись, що, наприклад, після натискання кнопки “Пошук” справді відкривається сторінка результатів, що посилання “Профіль” перенаправляє на сторінку авторизації якщо користувач ще не увійшов, тощо. Всі виявлені неузгодженості були оперативно усунуті. Для прикладу, було знайдено помилку, коли після реєстрації нового користувача не відбувався автоматичний вхід – вимагалося вручну логінитися. Це було виправлено: після реєстрації система виконує авторизацію нового користувача і перенаправляє його до анкети профілю.

Після проведення повного циклу тестувань система продемонструвала стабільну роботу. Всі модулі задовольняють функціональним вимогам та коректно взаємодіють один з одним. Рекомендаційний механізм генерує релевантні пропозиції, пошукові фільтри працюють точно, інтерфейс не містить критичних багів і зручний для користувача. Проєкт відповідає поставленим задачам: реалізовано всі заплановані можливості, а тестування підтвердило їх правильну роботу у реальних сценаріях. Це означає, що веб-система готова до експлуатації користувачами – функціональні вимоги виконані, продуктивність і надійність знаходяться на прийнятному рівні.

ВИСНОВКИ

У межах кваліфікаційної роботи на тему «Рекомендаційна система вибору житла з використанням технологій машинного навчання» було повністю реалізовано повнофункціональну веб-систему SmartHomeFinder, яка поєднує інструменти персоналізованого пошуку нерухомості та сучасні засоби автоматизації, зокрема рекомендаційні алгоритми та автоматизованого агента пошуку житла.

У процесі роботи було виконано всі етапи створення інформаційної системи – від постановки задачі та аналізу предметної області до практичного впровадження та тестування:

1. У теоретичному розділі здійснено огляд існуючих підходів до реалізації рекомендаційних систем у галузі електронної комерції та пошуку товарів, виділено найбільш ефективні підходи до контентного й колаборативного фільтрування, а також обґрунтовано доцільність їх використання у сфері пошуку нерухомості.
2. У розділі алгоритмічного та інформаційного забезпечення було детально описано архітектуру системи, її основні функціональні модулі, механізми обробки даних, а також алгоритми роботи рекомендаційного механізму та агента пошуку. Складено діаграми потоків даних (DFD), діаграми станів (STD), UML-діаграми класів і структурні ERD-моделі, що відобразили логіку функціонування системи.
3. Веб-застосунок реалізовано з використанням мови програмування Python, системи управління базами даних PostgreSQL, а також HTML/CSS/JavaScript для розробки користувацького інтерфейсу. Система складається з модулів управління користувачами, об’єктами нерухомості, збереженими вподобаннями, а також рекомендаційного ядра та автономного сервісу SearchAgent, що моніторить нові оголошення за заданими критеріями.
4. Реалізовано адаптивний веб-інтерфейс, що забезпечує зручну навігацію, інформативну структуру сторінок та привабливий дизайн. Користувач має змогу здійснювати розширений пошук об’єктів за багатьма параметрами (локація, тип житла, площа, ціна, додаткові зручності), переглядати деталі оголошень, зберігати вподобані варіанти, а також користуватись персоналізованими рекомендаціями.

5. Під час тестування система продемонструвала високу стабільність, коректність обробки запитів та відповідність функціональним вимогам. Усі модулі пройшли модульне, функціональне та UI-тестування. Виявлені недоліки були усунені, а система стабільно виконує пошук та надає персоналізовані рекомендації без втрати продуктивності.

Таким чином, розроблена система SmartHomeFinder повністю відповідає поставленій меті, демонструє практичне застосування сучасних технологій у сфері автоматизованих інформаційних систем і є придатною для подальшого масштабування та вдосконалення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. WorldBank**.** ТОП-25 сайтів для оренди житла в Європі і за кордоном. 2022. URL: https://worldbank.org.ua/3802-top-25-saytiv-dlya-orendi-zhitla-v-ievropi-i-za-kordonom-.html (дата звернення: 14.06.2025).
2. MDN Web Docs. HTML: Гипертекстовый язык разметки. 2025. URL: https://developer.mozilla.org/ru/docs/Web/HTML (дата звернення: 14.06.2025).
3. MDN Web Docs. Adding interactivity to your first website. 2025. URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Getting_started/Your_first_website/Adding_interactivity (accessed: 14.06.2025).
4. W3Schools UA. Онлайн-ресурс для вивчення CSS (каскадних таблиць стилів). 2025. URL: https://w3schoolsua.github.io/css/index.html#gsc.tab=0 (дата звернення: 14.06.2025).
5. Вовк П. Розділ 1. Загальна архітектура комп’ютерів. 5 типів структур обчислювальних машин і систем. 2025. URL: https://sites.google.com/view/vovkpetro/розділ-1-загальна-архітектура-компютерів-1-5/5-типи-структур-обчислювальних-машин-і-систем (дата звернення: 14.06.2025).
6. Max Zosim. Data Flow Diagrams: A Complete Guide. 2025. URL: https://www.maxzosim.com/data-flow-diagrams/ (accessed: 14.06.2025).
7. Lucidchart. Entity Relationship Diagram (ERD). 2025. URL: https://www.lucidchart.com/pages/i18n/erd (дата звернення: 14.06.2025).
8. ScienceDirect Topics. Database Structure (структура бази даних). 2025. URL: https://www.sciencedirect.com/topics/computer-science/database-structure (дата звернення: 14.06.2025).
9. Microsoft Support. Створення схеми — діаграми станів UML. 2025. URL: https://support.microsoft.com/uk-ua/office/створення-схеми-діаграми-станів-uml-2e46fd66-e861-4e8c-9188-36255395ebf3 (дата звернення: 14.06.2025).
10. Flask Documentation. Flask — Lightweight WSGI web application framework (stable version). 2025. URL: https://flask.palletsprojects.com/en/stable/

(accessed: 14.06.2025).

11. W3Schools UA. HTTP методи запитів. 2025. URL: https://w3schoolsua.github.io/tags/ref_httpmethods.html#gsc.tab=0 (дата звернення: 14.06.2025).

12. MindOnMap. «What is UML Class Diagram» — введення в діаграми класів UML: визначення, компоненти (класи, атрибути, операції), типові зв’язки, переваги та рекомендації щодо створення діаграми онлайн. 2023. URL: https://www.mindonmap.com/uk/blog/what-is-uml-class-diagram/ (дата звернення: 14.06.2025).

13. Ricci F., Rokach L., Shapira B. Recommender Systems: Handbook. Springer, 2015. 1000 p.

14. Adomavicius G., Tuzhilin A. Toward the next generation of recommender systems: a survey. IEEE Transactions on Knowledge and Data Engineering. 2005. Vol. 17, № 6. P. 734–749.

15. PostgreSQL Documentation. 2025. URL: https://www.postgresql.org (дата звернення: 27.05.2025).

16. SQLAlchemy. Python SQL Toolkit and Object Relational Mapper. 2025. URL: https://www.sqlalchemy.org/ (дата звернення: 14.06.2025).

17. Python Software Foundation. Python Programming Language. 2025. URL: https://www.python.org (дата звернення: 27.05.2025).

18. Scikit-learn: Machine Learning in Python. 2025. URL: https://scikit-learn.org (дата звернення: 27.05.2025).

19. Figma. Онлайн-інструмент для спільної розробки інтерфейсів, прототипування та дизайну продукту. 2025. URL: https://www.figma.com (дата звернення: 14.06.2025).

20. Figma. Різниця між UI та UX дизайном. 2025. URL: https://www.figma.com/resource-library/difference-between-ui-and-ux/ (дата звернення: 14.06.2025).

21. OLX. Платформа для купівлі, продажу та оренди нерухомості. 2025. URL: https://www.olx.ua (дата звернення: 27.05.2025).

22. DOM.RIA. Сервіс пошуку та купівлі нерухомості. 2025. URL:

https://dom.ria.com (дата звернення: 27.05.2025).

23. Airbnb. Платформа для короткострокової оренди житла. 2025. URL: https://www.airbnb.com (дата звернення: 27.05.2025).

24. Booking.com. Платформа для пошуку житла. URL: https://www.booking.com (дата звернення: 14.06.2025).

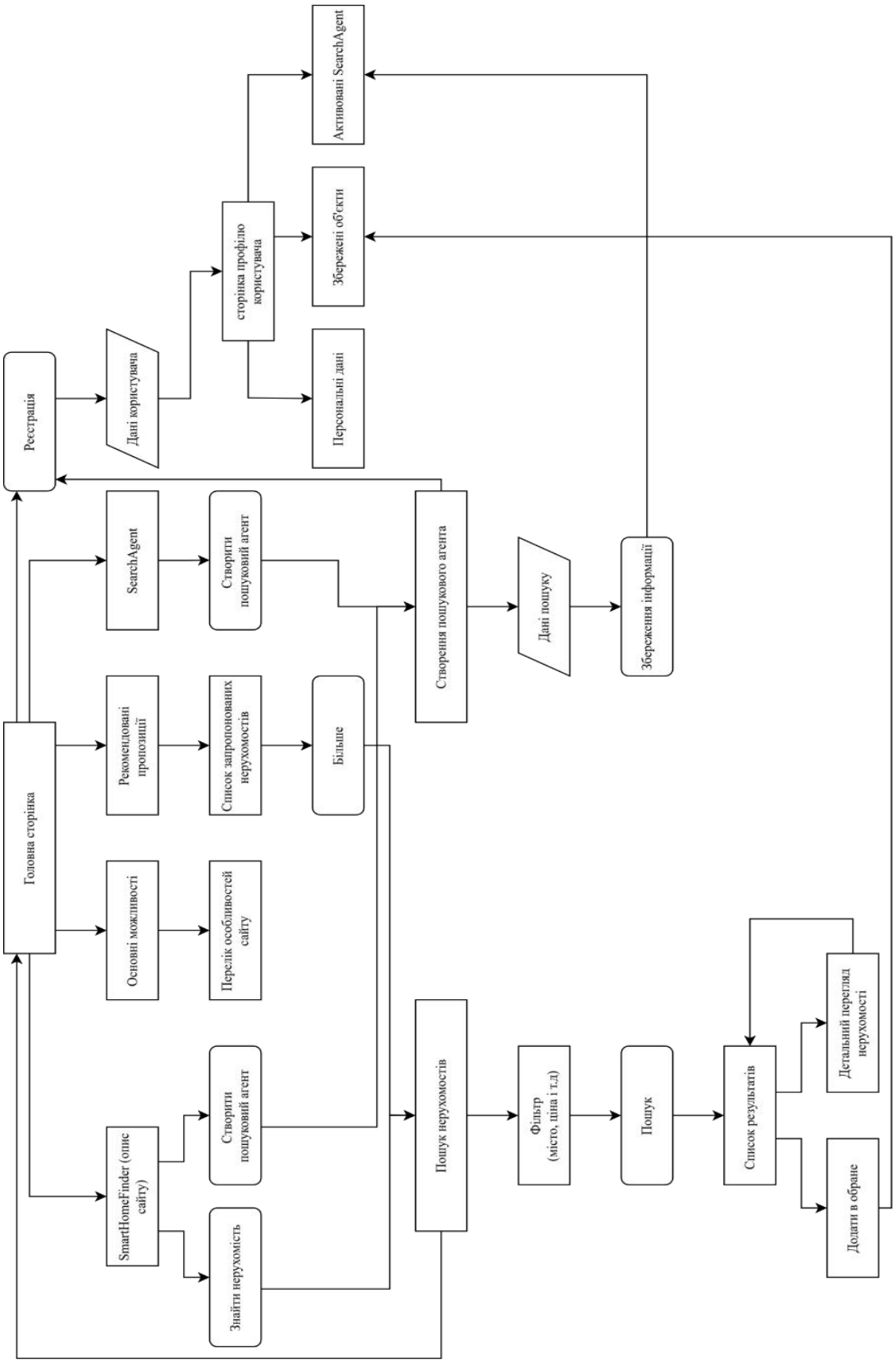
25. Medium. How Airbnb uses machine learning. URL: https://medium.com/airbnb-engineering/how-airbnb-uses-machine-learning-8f3b7ac5a42e (дата звернення: 14.06.2025).

26. Яцюк Ю.Р., Биковий П.Є. Рекомендаційна система вибору житла як інструмент управління користувацькими рішеннями. Інтелектуальні інформаційні технології в прикладних дослідженнях (IITAR-2025): збірник тез доповідей студентської науково-практичної конференції (Тернопіль, 27–29 травня 2025 р.). Тернопіль: ЗУНУ, 2025. С. 342-345.

27. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп'яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

Додаток А

Структура веб-сайту



Додаток Б

Копії опублікованих результатів

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ

Студентської науково-практичної конференції

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРИКЛАДНИХ ДОСЛІДЖЕННЯХ

(IITAR-2025)

27-29 травня 2025 року

Тернопіль

2025

Збірник тез доповідей студентської науково-практичної конференції «Інтелектуальні інформаційні технології в прикладних дослідженнях» (ІІТАR-2025). (Тернопіль, 27-29 травня 2025 року). Тернопіль: ЗУНУ, 2025. 372 с.

До збірника увійшли тези доповідей учасників студентської науково-практичної конференції "Інтелектуальні інформаційні технології в прикладних дослідженнях" (ІІТАR – 2025), що відбувалась у рамках AI Week на базі кафедри інформаційно-обчислювальних систем і управління Західноукраїнського національного університету. Мета конференції — об'єднати науковців, освітян, представників ІТ-бізнесу та здобувачів освіти для обміну досвідом, презентації сучасних досліджень і впровадження інноваційних рішень у сфері комп'ютерних наук, штучного інтелекту та суміжних галузей.

За зміст наукових праць та достовірність наведених фактологічних і статистичних матеріалів відповідальність несуть автори публікацій та їхні наукові керівники. У збірнику зберігається стилістика та орфографія авторів матеріалів.

Склад організаційного комітету конференції

Керівництво оргкомітету

Мирослав Комар, д.т.н., професор, професор кафедри інформаційно-обчислювальних систем і управління

Христина Ліп'яніна-Гончаренко, д.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Надія Васильків, к.т.н., в.о. завідувача кафедри інформаційно-обчислювальних систем і управління

Члени програмного комітету

Василь Коваль, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління, заступник декана факультету комп'ютерних інформаційних технологій

Олександр Осолінський, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Павло Биковий, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Діана Загородня, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Ігор Майків, к.т.н., доцент кафедри інформаційно-обчислювальних систем і управління

Члени оргкомітету

Андрій Івасечко, викладач кафедри інформаційно-обчислювальних систем і управління

Дмитро Дюг, викладач кафедри інформаційно-обчислювальних систем і управління

Христина Юрків, студентка ФКІТ ЗУНУ, технік лабораторії з проблем інформаційних технологій

Мар'яна Соя, студентка ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління

Микола Телька, студент ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління, студентський декан факультету комп'ютерних інформаційних технологій

Яцюк Юлія
студентка групи КН-42
yulya.yatsyuk@gmail.com
Биковий Павло
к.т.н., доцент
pb@wunu.edu.ua
Західноукраїнський національний університет
Тернопіль, Україна

РЕКОМЕНДАЦІЙНА СИСТЕМА ВИБОРУ ЖИТЛА ЯК ІНСТРУМЕНТ УПРАВЛІННЯ КОРИСТУВАЦЬКИМИ РІШЕННЯМИ

У сучасному цифровому середовищі процес вибору нерухомості стає дедалі складнішим через велику кількість оголошень, динамічні зміни на ринку та індивідуальні вподобання користувачів. Традиційні платформи, такі як OLX, Dom.ria чи Lun.ua [6-8], часто не враховують поведінкові патерни користувачів і не пропонують персоналізованих рішень інформація у таблиці 1 використана для демонстрації переваг SmartHomeFinder над існуючими сервісами на основі відкритих даних [6-8]. Це знижує ефективність пошуку житла та ускладнює прийняття рішень.

Метою розробки SmartHomeFinder є створення веб-системи з використанням алгоритмів машинного навчання для формування персоналізованих рекомендацій у сфері пошуку житла [1, 2]. Такий підхід дозволяє адаптувати процес до індивідуальних потреб кожного користувача, автоматизуючи моніторинг ринку й аналізуючи тренди [3].

Система поєднує контентну та колаборативну фільтрацію [1], аналіз історії переглядів, фільтрацію за обраними параметрами та інші критерії, формуючи рекомендації на основі переваг користувачів. У структурі SmartHomeFinder передбачено фронтенд на основі HTML/CSS/JS, бекенд на Python, базу даних PostgreSQL [5], а також використання бібліотеки TensorFlow для реалізації моделей машинного навчання [4].

Проблема поновлення на ринку власності має широкий обсяг, створює навантаження та заплутаність для користувачів. З метою спрощення процесу рекомендується використання рекомендаційної системи [2].

Завданням є розробити інструмент, який би автоматизував вибір для задоволення користувацьких запитів (послуги, характеристики, переваги) (рисунок 1).

Рисунок 1 – Архітектура системи SmartHomeFinder

Таблиця 1 – Порівняльна характеристика платформ пошуку житла

Функція	OLX	Dom.ria	Airbnb	Lun.ua	SmartHomeFinder
Пошук за фільтрами	☑	☑	☑	☑	☑
Персоналізовані рекомендації	✗	⚠	☑	✗	☑
Аналіз ринкових трендів	✗	⚠	☑	☑	☑
Автоматичний моніторинг оголошень	✗	✗	✗	✗	☑
Гібридна модель рекомендацій	✗	✗	☑	✗	☑

Отже, впровадження SmartHomeFinder дозволяє оптимізувати процес вибору житла, підвищити релевантність результатів пошуку та покращити користувацький досвід [1, 2]. Це інноваційне рішення в управлінні інформацією, що ґрунтується на персоналізації та інтелектуальній аналітиці [3]. Подальший розвиток передбачає інтеграцію з реєстрами, мобільні додатки та розширення підтримки API для зовнішніх джерел.

Список використаних джерел

1. Ricci F., Rokach L., Shapira B. Recommender Systems Handbook. Springer, 2015.
2. Aggarwal C. Recommender Systems: The Textbook. Springer, 2016.
3. Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems. Computer, 2009.
4. TensorFlow Documentation. URL: https://www.tensorflow.org
5. PostgreSQL Documentation. URL: https://www.postgresql.org
6. OLX. Платформа оголошень. URL: https://www.olx.ua
7. Dom.ria. Платформа пошуку житла. URL: https://dom.ria.com
8. Lun.ua. Український сервіс нерухомості. URL: https://www.lun.ua